Testing the Application for Analyzing Structured Entities

Ion IVAN, Bogdan VINTILA
Academy of Economic Studies, Bucharest, Romania
ionivan@ase.ro, vb@vintilabogdan.ro

The paper presents the testing process of the application for the analysis of structured text entities. The structured entities are presented. Quality characteristics of structured entities are identified and analyzed. The design and building processes are presented. Rules for building structured entities are described. The steps of building the application for the analysis of structured text entities are presented. The objective of the testing process is defined. Ways of testing the application on components and as a whole are established. A testing strategy for different objectives is proposed. The behavior of users during the testing period is analyzed. Statistical analysis regarding the behavior of users in processes of infinite resources access are realized.

Keywords: *Structured Entity, Application, Analysis, Testing, Metrics*

1 Structured entities

The structured entities can be perceived as trees. These have nodes and links between them. A general structured text entity is represented in Fig. 1. The main structured entity is a tree that has many levels. On each level there are many sub structured text entities. The subdivision process continues until some text entities are elementary.

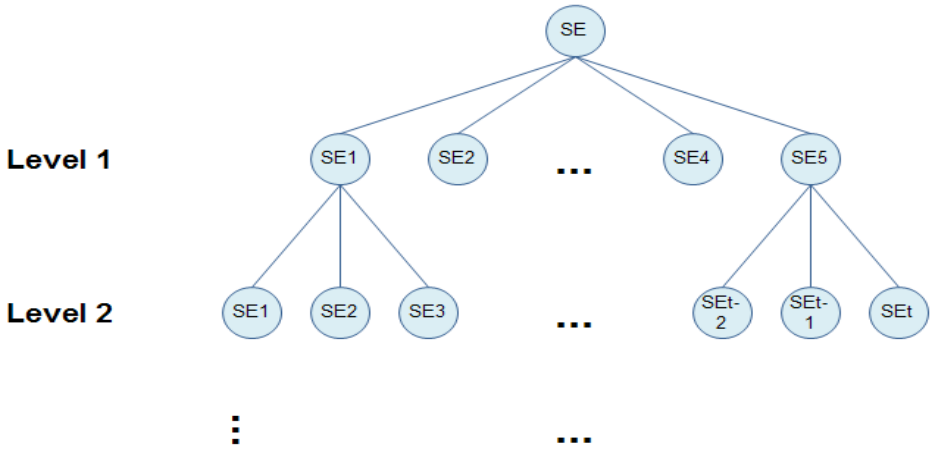

Fig. 1. Structured text entity

Let SE_i^k be the *i* entity from level *k* of the tree. It is obtained by concatenating all entities from level *k+1* that are linked to this one:

$$SE_i^k = SE_f^{k+1} || SE_{f+1}^{k+1} || SE_{f+2}^{k+1} || \dots || SE_{f+g-1}^{k+1}$$

where:

f - the index of the first element from the level *k+1* that is linked with SE_i^k;

g - the number of elements from *k+1* level that are linked with SE_i^k.

On all levels the structured entities are ordered by the links with superior entities. As all entities from the tree are built by concatenating inferior entities, the building process is recursive.

The structured text entity used by this paper is defined by:

- T_1 is the first component of the entity and it stands for the title; it is used as an identifier for the entity;
- T_2 is the second component of the entity and it contains the abstract; this

component is used to compress the contents of the structured entity and give the readers interesting information so that they read all the structured entity;

- T_3 is used to store the keywords of the entity; the indexing of the entities is done by these keywords; the searches of the readers are done by using the title and the keywords;
- T_4 … T_{ncse} are components that contain two sub-entities; the first sub-entity is the title of the component and the second one is the body; *ncse* is the total number of components of the structured text entity; these form the major part of the structured entity.

Each of these components must comply with rules.

For component T_1 the size requirements assume it is not shorter than three words and not longer than fifteen. The component T_2 must be longer than fifty words and shorter than one hundred. Component T_3 must be shorter than ten words and longer than three.

The structured text entity must also be equilibrated. This assumes that the length of the T_4...T_{ncse} must be approximately equal. The length of an element is given by the LG() function. A structured text entity is equilibrated if:

$$\forall 4 \le i \le ncha, \frac{LG\left(\sum_{j=4}^{ncse} T_j\right) - \theta}{ncha - 3} \le LG(T_i) \le \frac{LG\left(\sum_{j=4}^{ncse} T_j\right) + \theta}{ncha - 3}$$

where θ is a factor that determines how much the deviation from the average value can be.

By using this representation of the structured text entities, the orthogonality can be computed between entities on the same level and on different levels. The intra-level orthogonality is computed using entities from the same level: $H\left(ES_i^k, ES_{i+j}^k\right)$. The inter-level orthogonality is uses entities from different levels: $H\left(ES_i^k, ES_{i+j}^{k+z}\right)$. This ensures a high level of orthogonality of the structured text entity as all components can be checked to comply with orthogonality rules.

Structured text entities have quality characteristics that are used to determine the quality level. In an environment where there are more than one structured text entity there are quality characteristics that are determined in report to the already existent set of entities. Quality characteristics are:

- orthogonality is the measure in which the entities are different one from another; this characteristic can only be determined when there are other entities in the environment and it is computed in report to all available entities; the orthogonality can be computed in different ways; the simplest way is to report the common words in the vocabularies of the entities to the minimum length vocabulary:

$$I_o = 1 - \frac{LGA}{LGB}$$

where the lengths of vocabularies, *LGA* and *LGB*, are given by the relations:

$$LGA = LG(V_1 \bigcap V_2)$$

$$LGB = \min\{LG(V_1), LG(V_2)\}$$

- the completeness of the entities is given by the measure in which these are completed by the user [1]; the completeness is valid not only for entities but also for data; in the case of structured text entities we consider components to be data; data completeness is given by the measure in which the users input the required data in the application; let *DI* be the set of data entry methods used by the developers to cover the data requirements for the application; for each element in *DI*, the developers may use more than one instance; the number of instances for each element is given set *DER*; the total number of data entry requirements, *TDER*, is given, as a simple sum [9], by the formula:

$$TDER = \sum_{i=1}^{ndem} der_i$$

where:

ndem - number of data entry methods used

by the application;

der_i - element *i* from the DER set.

the data completeness indicator, *DCI*, is given by the formula:

$$\mathrm{DCI} = \frac{\sum_{i=1}^{ndem} \mathrm{sder}_i}{\sum_{i=1}^{ndem} \mathrm{der}_i}$$

where:

$sder_i$ - number of data entry requirements satisfied by the users;

der_i - element *i* from the set defining the number of elements of each type.

if the value of the indicator is lower than 0.7 the users haven't insert the required data not even in two thirds of the data input controls and they must review all controls; if the value of the indicator is lower than 0.95 but greater than 0.7 the users insert most of the data and they only have to input few data in order to complete the process; only if the indicator's value is 1 the users have completed the process and the application has complete data to process; the *DCI* indicator shows the measure in which the users covered all data entry requirements;

- the correctness of structured entities is given by the proportion of components that are correct [2]; the correctness of components is given by many rules; one of the most important is the membership of words from the component to the vocabulary; in the case of the structured text entities each component has a vocabulary associated; for the component to be correct, all words that compose it must exist in the associated vocabulary; this means that:

$$\forall i \leq nc, C_i - V_i = \emptyset$$

where *nc* is the number of component the entity has; in the case of data, the correctness is not given only by the membership to the vocabulary; in the case of numerical data the most common requirement is the membership to an interval; for calendar dates one of the most common problems is the format the user type it in; for measurements, one of the most problematic issue is typing measurement data using different measurement units;

- the consistency is very important for the structure text entities as authors must not argue with themselves; let us consider a structured text entity, *STE* formed of many phrases; $STE=\{P_1,P_2,P_3,\ldots\}$; let $\overline{STE}$ be the negation of the *STE* entity; the *STE* entity is consistent only if:

$$STE \cap \overline{STE} = \emptyset$$

this ensures that the author has been consistent when building the structured text entity; for informatics applications the consistency means that the application always deliver the same result for the same dataset [3].

In the case of structured text entities, the quality characteristics must be analyzed to determine the overall quality of the structures. As the components of structured entities are composed of words belonging to a vocabulary, the components and the associated vocabularies must possess the quality characteristics. The possible structures of text entities derive from the tree structure. By customizing the way the links are made between components and the number of child components, different structures are built.

2 Designing and building structured entities

Structured text entities are formed of many components and each of these components must be built following certain rules. The design process assumes the definition of the theme of the structured text entity, the structure of the entity and the rules that must be followed while building the entity. The building process assumes following the rules defined in the design stage to obtain a suitable text entity.

The design process of the structured text entity is complex and it must consider everything involved in the creation of the entities. The design process of structured text entities has the following steps:

Step A_0 – the team that is to design and build the structured text entity is defined; the team must include specialists that cover the subject of the structured text entity;
Step A_1 – the definition of the domain the entity is related to; a structured text entity can't lack the subject and be just a set of random words; each structured text entity has a theme that is related to a domain; a domain is a very broad delimitation in the knowledge environment; domains are formed of areas; each domain has a general area that defines general aspects about the contents of the domain; the other areas are more specific and detailed;
Step A_2 – the definition of the area the structured text entity is focused on is important as it can't focus on all areas of a domain; through the definition of an area, the structured text entity focuses only on some of the elements of the domain and can treat them in more detail;
Step A_3 – the definition of the issue from the selected area is vital as areas are too large for a structured text entity of low or average complexity; for high complexity structured text entities the selection is not limited to only one issue but to a part of them or even all of the issues from an area.
Step A_4 – the identification of the problem regarding the chosen issue is important as the structured text entity must state clear its purpose; the reader must know from the first lines if the structured text entity deserves reading;
Step A_5 – the definition of the problem and the identification of possible solutions are a must be for every structured text entity; even if the approached problem has been treated by others, the definition of the problem may be different and thus the solutions may differ; the definition of the problem must be clearly stated and the possible solutions must show their weaknesses and strong points;
Step A_6 – the identification of documentation sources is extremely important as one can find himself in the situation of trying to do something already done; documentation is vital as it shows what others have done before and gives an insight over the area; in the structured text entity there must be a component that refers what others have done in the area approached by the entity;
Step A_7 – defining the structure of the structured text entity gives the entity a skeleton to build on; if no structure is defined, the author will have nothing to build on and will just write his ideas in the order they appear; this will lead to a random set of words and not a structured text entity;
Step A_8 – defining the vocabularies for the components of the structured text entity makes the author use only certain words in the development of the entity; dictionaries of words that are forbidden must also be defined;
Step A_9 – the definition of construction rules that must be respected during development is important as, after the development is done without rules, complying to them is extremely difficult and resource consuming; the rules for the building of structured text entities must have some compulsory elements for them to be valid; if all rules are respected at the building stage, the structured text entity will be finished as planned.
The building process is the actual activity of building the structured text entity. This is the process of using the words from vocabularies to create components and using these to obtain a structured text entity. The building process must follow the following steps:
Step B_0 – the structure of the text entity is created; this will help the authors build in an organized manner and also guide them through the structured text entity; for structured text entities such as scientific papers the author creates the skeleton of the entity; the entity will have a title, abstract and keywords, classification, introduction with literature review, methodology, results, interpretation, conclusions; for each of these components a suitable title must be found; the structure definition step is the process that has the greatest impact on the future development of the structured entity;
Step B_1 – the current component of the structured text entity is constructed by the author; this means that the author writes the

information he desires; as the structured text entity has many components, this step will be repeated for all components;

Step B_2 – the list of vocabularies needed for the validation of the component is created; each component can be checked with multiple vocabularies; inclusion or exclusion of the words from the vocabularies is checked according to the type of vocabulary; in the case of vocabularies, the authors of papers don't create vocabularies with words they will use, but rather create vocabularies with words that are forbidden;

Step B_3 – for each vocabulary from the list created at step B_2 the component is checked to see if it complies with the inclusion or exclusion rules for the words; the author is notified if there are problems with the use of words from vocabularies; as vocabularies for large papers are large, as well, an application is used to determine if the rules that involve vocabularies are complied with. a risk that appears here is that if the algorithm that checks the vocabularies and the entity is not correct, words from the forbidden vocabularies might slip into the entities' components; the author must check to see if the algorithm works as it should;

Step B_4 – the list of rules that apply to the current component is created; not all rules apply to all components so the ones that apply must be selected and verified; as for a structured text entity there are many rules to comply with, the authors must select only the rules that are applicable for the current component;

Step B_5 – the rules from the list created at step B_4 are verified against the current component; the author is notified if there are problems with the compliance to rules; the messages must state clearly what problem occurred, where it is located and what the possible solutions are;

Step B_6 – if there were any problems with the vocabularies or the rules, the author is sent back to step B_1 to edit the component and solve the identified problems; if no problems occurred, the next component is selected and the author is sent to step B_1 to add contents to it; if there are no more components, the building process is done.

If all steps are respected by the author of the structured entity, it has a high quality level after the whole process is done.

3 Application for the development of structured text entities assisted

For assisting the development of structured text entities an application is developed. The application is aimed at regular citizens so it has to be citizen oriented. In order to guide the users through the building process of structured text entities, the application is structured in steps. Each step of the structured text entities building is available for users only for a limited period of time. The application allows users to access only the steps associated to the period of time the users access it. The application must be freely accessed so it is developed by using the client-server model [4]. The steps followed in the development of the application for assisting the development of structured text entities were:

Step C_0 – the analysis of the target group is essential for any citizen oriented informatics application; for the application for assisting the development process of structured text entities the target group is formed of all citizens as any of them might be the author of a text entity; as the target group is heterogeneous the training level is very different and the application must be designed so that all users can use it; the logic and the interface of the application are to be as simple as possible so that all users can use it without problems;

Step C_1 – the steps of building structured text entities were defined; for this application six steps were identified: title definition, abstract and keywords definition, bibliographic titles insertion, insertion of the chapters' titles, definition of figures and tables, insertion of full text; the users must follow these steps in order to create a complete structured text entity;

Step C_2 – the time intervals associated to these steps of the building process were defined; the testing process in which the

users accessed the application lasted six weeks; each week was associated to one step from the building process of the structured text entity;

Step C_3 – the platform and database system were chosen to be ASP.NET with C# code and Sql Server; as the application is addressed to citizens, it has to be free and accessible; through the use of ASP.NET, the only costs associated to the application are the costs of hosting [6]; the costs of the citizens are zero; the accessibility is ensured through the use of the client-server model; the application resides on the server and the users access it through an interface hosted by any internet browser [7]; the solution ensures maximum accessibility as users can access it using any hardware and software architecture from any given computer with internet access; Sql Server was chosen to be the database management system as it has the express version that is free and it can be installed on the development machines and used in the development and testing processes [8];

Step C_4 - the general structure of the text entity targeted by the application was defined; its structure include the title, the abstract and keywords, references, chapters that have title and body, figures and tables;

Step C_5 – the rules each of the components of the structured text entity must comply with are defined; for each component were defined rules aiming the size of the components, the vocabularies used, the inclusion and exclusion of elements, relations between components;

Step C_6 – the requirements of the application were defined; for each step of the building process of structured text entities the requirements were identified and detailed; each rule defined for the components of the structured text entity was translated in logic statements;

Step C_7 – the database structure was defined; each user must have an account; a table containing details about the users was created in the database; the database contains a table that stores information about the structured entities' titles, abstracts and keywords; references are stored in a dedicated table; the link with the entity is made through a unique code and the username; chapters' titles and text are stored in a table; the link is made through the same unique identifier; figures and tables are stored in a dedicated table; they are differentiated through a field that indicates their type; each figure and table has a chapter associated; they are defined in that particular chapter and they are also must be cited there;

Step C_8 – for the steps of the building process of the structured text entities the forms were designed; the error and confirmation messages for all situations were developed; these state what the problem is, where it is located and possible ways of dealing with it;

Step C_9 – the logic of the application was implemented; the class *articol* was written; this class has members to store the structured text entity's components; methods were also created to implement the rules that must be complied by the components; six versions of the application for the assisted development of structured entities were developed for each step the users must follow; each version is built on the previous one and the last version implements all required functionality;

Step C_{10} – the testing of the application is done by the development team and users; the testing done by the development team is done during the development process and in the testing phase; the testing process that involves users is done after the development is completed and the development team has conducted its testing routines; the testing identifies errors and the development team corrects them; the testing process is repeated until no errors are identified;

Step C_{11} – the maintenance process consists of corrections and addition of new features to the application; for the application that assists the users in the development of structured text entities the main tasks of the maintenance process were to keep the vocabularies updated with new terms and correct minor problems that users encountered.

The development process aimed at producing a citizen oriented informatics application. The interface was designed to be as simple as possible. The messages used common terms and specified as clear as possible where the problem occurred and which are the solutions for it. The client-server model used for the development of the application ensured accessibility. The chosen technologies implied no costs for the users. By using this application, the quality of the obtained structured text entities will be higher than if using the traditional building process [5].

4 Testing the application for assisting the development of structured entities

The objective of the testing process is the identification of faults in the informatics application and the observation of the users' behavior. For the application that assists authors in the development of structured text entities the testing process aimed at identifying problems in components and in the application as a whole. For the testing of the components users had time intervals for the testing of each component. For the testing of the whole application, users had to access the application and insert the whole structured text entity in one step. They had access to all steps and could access them in the required order. After they completed one step, the next one was made available for them.

The testing of the components has been done over six weeks by around seventy users. Each user built the structured text entity that he used to test the application.

In Table 1 are given the tasks of the users and the time intervals associated to these tasks.

Table 1. Tasks and associated time intervals

Task	Starting date	Ending date	Interval length (days)
Title definition	25/10/2010	03/11/2010	10
Abstract and keywords definition	04/11/2010	11/11/2010	8
Insertion of references	12/11/2010	17/11/2010	6
Insertion of the titles of chapters	18/11/2010	24/11/2010	7
Insertion of tables and figures	25/11/2010	01/12/2010	7
Insertion of full text	02/12/2010	08/12/2010	7
Total length (days)	-	-	45

The number of intervals for the testing of the application, *NIT*, is six. Each interval, T_i, is associated to one task. The tasks consist of testing different components of the application. These have approximately equal length. The total length of the testing period, *TLTP*, is given by the formula:

$$TLTP = \sum_{i=1}^{NIT} TI_i$$

For the developed application the value of *TLTP* is 45. During these forty five days the application recorded the actions of the users for the study regarding their behavior.

The testing process assumes the informatics application must deliver correct results for correct data and messages for incorrect or incomplete data. Let *AD* be a set of data that is both correct and complete and the application must deliver results for it and no error messages. Four situations appear. The first situation is when the application delivers the correct results and no error messages. The second situation is when the application delivers the results and no error messages, but the results are not correct. The third situation is when the application delivers no results and gives error messages regarding the data. The fourth situation is when there are no results, and all data from the set is reported to be incorrect. For the *AD* set only the first situation must be presented by the application and the others must be identified and corrected during the testing process.

Let *BD* be a bad dataset for which the application must give only error messages as

all data are incorrect. The single accepted situation for this set must be that in which the application gives only error messages for all data and no results. Any other situation must be identified and corrected in the testing process.

If the application can process batches of datasets and set *AD* and *BD* are concatenated, the application must deliver correct results for the first dataset and error messages for the second dataset.

Let:

- fw_{ij} be the absolute frequency of user *i* in week *j*;
- *nw* be the number of considered weeks and *nu* the number of users that participated to the testing process.

Then, the sum of actions for user *i*, $T_{i.}$, is given by the formula:

$$T_{i.} = \sum_{j=2}^{nw} fw_{ij}$$

And the total of actions from week j, $T_{.j}$, is given by the formula:

$$T_{.j} = \sum_{i=1}^{nru} fw_{ij}$$

where *nw* is the number of intervals the testing is done on and *nru* is the number of users that participated to the testing process. Users have been associated numbers in the alphabetical order of their username.

The maximum value of actions of an user during week *j*, $MV_{.j}$, is given by the formula:

$$MV_{.j} = \max_{i=1,nru} fw_{i,j}$$

The average value of the number of actions in week *j*, $ANA_{.j}$, is given by the formula:

$$ANA_{.j} = \frac{\sum_{i=1}^{nru} fw_{ij}}{nru}$$

In Table 2 are given the maximum value, average value and standard deviation values for the number of actions on weeks and total.

Table 2. Statistical indicators for the number of actions

	Week 2	Week 3	Week 4	Week 5	Week 6	TOTAL
$T_{.j}$	1805	4872	1370	722	1724	10493
$MV_{.j}$	141	199	113	32	196	465
$ANA_{.j}$	25.79	69.6	19.57	10.31	24.63	149.9
STANDARD DEVIATION	24.67	34.61	16.53	9.21	34.06	74.42

In the second week the number of actions was of almost two thousand. The maximum value for a user was of one hundred and forty one. The average value was of twenty five point seventy nine. The high value of the standard deviation shows that there were users that had very few actions in the application and also users that had much more actions than the average value.

For the third week the situation is similar as the maximum value is almost three times greater than the average value. The value of the standard deviation also indicates that the data set for the actions is widely distributed.

In the fourth week, even if the maximum value is almost six times greater than the average value, the standard deviation indicates that the values of the set are not as spread as in the previous weeks.

Week five is the most equilibrated one as the maximum value is only three times greater than the average and the value of the standard deviation is the smallest from all considered periods. This shows that the elements are not spread from the average value.

In week six the situation is very similar to week three, but the average value of the number of actions is much smaller. This is due to the fact that some users quit the process and didn't take any actions in the application.

The moving averages are defined differently for odd and even periods:

$$dm^{2k} = \frac{x_{l-2k} + \cdots + x_{l-k-1} + x_{l-k} + \cdots + x_l}{2k}$$

$$dm^{2k+1} = \frac{x_{l-2k-1} + \cdots + x_{l-k-1} + \boldsymbol{x_{l-k}} + x_{l-k+1} + \cdots + x_l}{2k+1}$$

The moving averages are used to remove the short term variations of the data series [10]. By following the moving averages data series one can avoid the short term variations and see the long term trend.

Table 3. The number of users' actions for the recorded days and the moving averages

Day	Actions	$dm^{(2)}$	$dm^{(3)}$	$dm^{(4)}$	$dm^{(5)}$	$dm^{(6)}$
1	59	-	-	-	-	-
2	12	35.5	-	-	-	-
3	20	16	25.75	-	-	-
4	54	37	26.5	26.12	-	-
5	71	62.5	49.75	38.12	32.12	-
6	202	136.5	99.5	74.62	56.37	44.25
7	129	165.5	151	125.25	99.93	78.15
8	1258	693.5	429.5	290.25	207.75	153.84
9	6	632	662.75	546.12	418.18	312.96
10	152	79	355.5	509.12	527.62	472.90
11	386	269	174	264.75	386.93	457.28
12	349	367.5	318.25	246.12	255.43	321.18
13	344	346.5	357	337.62	291.87	273.65
14	3635	1989.5	1168	762.5	550.06	420.96
15	318	1976.5	1983	1575.5	1169	859.53
16	55	186.5	1081.5	1532.25	1553.87	1361.43
17	31	43	114.75	598.12	1065.18	1309.53
18	96	63.5	53.25	84	341.06	703.12
19	28	62	62.75	58	71	206.03
20	82	55	58.5	60.62	59.31	65.15
21	760	421	238	148.25	104.43	81.87
22	55	407.5	414.25	326.12	237.18	170.81
23	46	50.5	229	321.62	323.87	280.53
24	116	81	65.75	147.37	234.5	279.18
25	29	72.5	76.75	71.25	109.31	171.90
26	73	51	61.75	69.25	70.25	89.78
27	68	70.5	60.75	61.25	65.25	67.75
28	335	201.5	136	98.37	79.81	72.53
29	70	202.5	202	169	133.68	106.75
30	21	45.5	124	163	166	149.84
31	60	40.5	43	83.5	123.25	144.62
32	45	52.5	46.5	44.75	64.12	93.68
33	141	93	72.75	59.62	52.18	58.15
34	355	248	170.5	121.62	90.62	71.40
35	1032	693.5	470.75	320.62	221.12	155.87

The number of actions users took each day is given in Table 3. The moving averages with two, three, four, five and six periods are calculated to eliminate the short term

variations. The number of daily actions is graphically presented in Fig. 2.

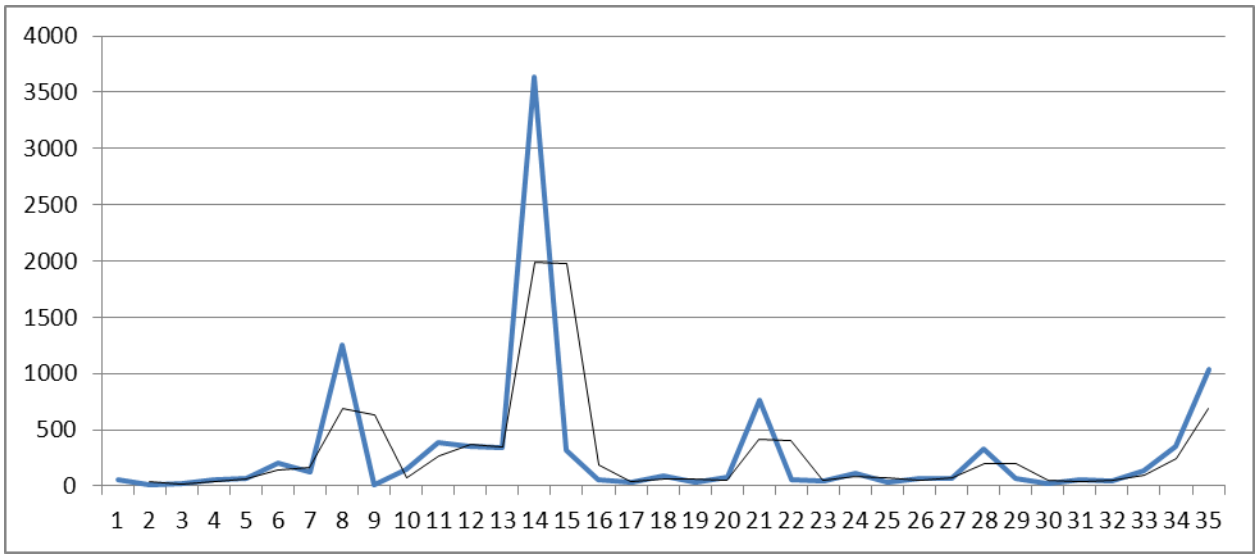

Fig. 2. Daily application actions and two periods moving averages

The daily actions graph shows that are days with great activity. These are the final days of the time intervals defined for each task. Users accessed the application mostly in the final days of the time interval associated for their task. With every application action, the users tested the functionality of the forms and the logic of the application. The application was modified when users identified problems with the forms or the logic. The trend using moving averages with two periods is represented by the black line.

The parallel testing of the application for the assisting of the development of structured text entities is showed in Fig. 3.

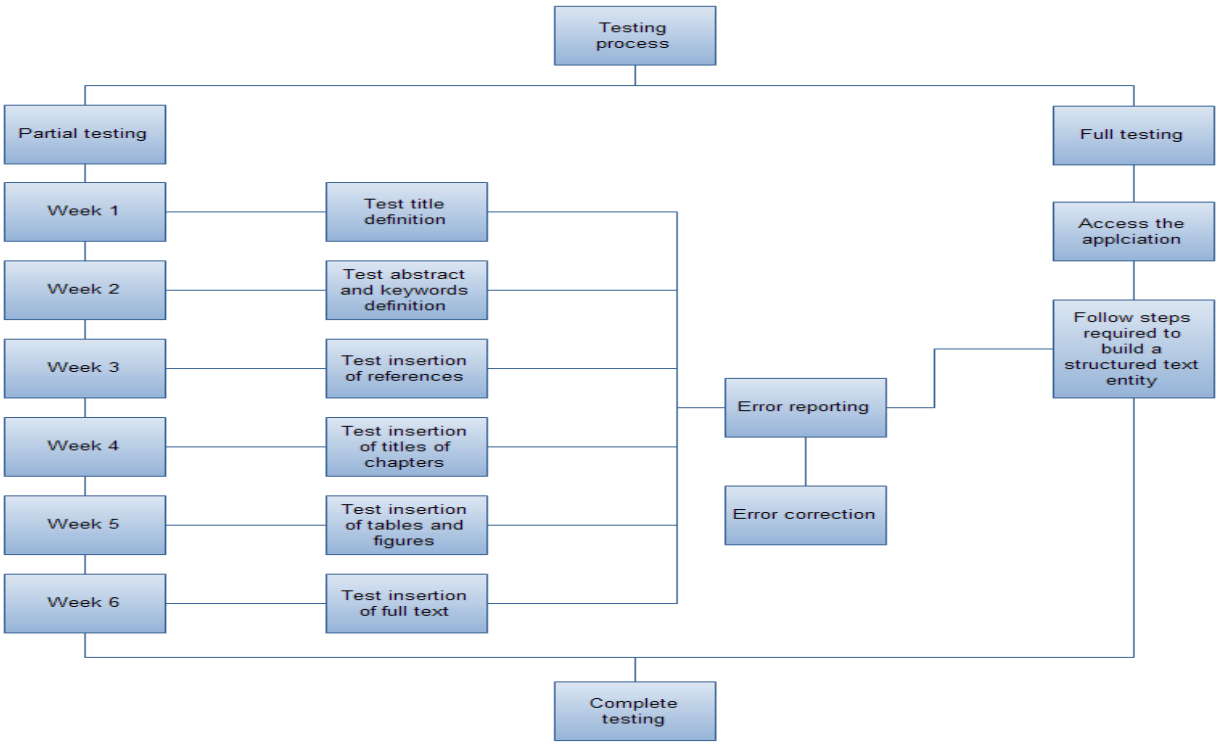

Fig. 3. Parallel testing process

The parallel testing process assumes the application is tested both on components and as a whole. For the application for assisted development of structured text entities the testing has been done in parallel with two versions of the application. The version for complete testing allowed users to access all steps needed for the development of the structured text entity. The objective was to see if the components function well when they are put together and identify the possible problems that appear. The version that had as objective the testing of the components restricted the access of users to just one component. The restriction was imposed so that users focus on just one component and ignore the relations between them.

The parallel testing of the application allowed the more rapid identification of faults and also the reaching of more than just one objective. The application was tested at the same time on components and as a whole. The code has been well documented to help the future maintenance process and the application has a modular structure for the same reason.

5 The behavior of users in the testing process

As historical data is very important for future projects, the application recorded all actions the users made in the test period.

Each time interval associated to a task has days in which no user completed his task. The last days have a high number of users that complete their tasks. Most users access the application in the last day of the time frame and complete their task. For the first two tasks all users completed them and then the percentage of users that completed the tasks lowers. The third and fourth tasks have been completed by sixty six users of seventy. For the fifth and sixth tasks the percentage drops even more as forty nine, respectively forty eight users completed them. The percentage of users that completed all tasks is sixty eight point fifty seven.

For the testing of the first component, the title definition one, the behavior of the users is given in Fig. 4. The users don't have a homogenous repartition on the days of the interval. They ignore their task until the last days of the interval when they access the application. In some of the days of the interval the application is not accessed once while in the last day almost half of those implied in the testing process access it. Half of the participants access the application in the first nine days of the time interval while the other half access it in the last day. At the half of the time interval only 11.43% of the participants accessed the application and fulfilled their task. After 70% of the time passed only 20% of the participants accessed the application. The last three days of the time interval totaled 80% of the participants to the testing process.

Possible causes of this behavior are:

- during week time the participants focused on other activities that defining the title of the structured text entity; after the weekend the number of users that defined the title of the structured text entity increased slightly; this shows that the few free days allowed participants to focus on working on the structured text entity; in the last day almost half of the participants to the testing process accessed the application and defined the title of their structured text entity;
- as all participants have more activities programmed on short term they have to find time for an additional one; it is in human nature to postpone something as long as possible; this explains why half of the participants accessed the application and defined the title for their structured text entity in the last day of the allocated time interval.

Solutions for normalizing the distribution of participants in time are:

- the allocation of smaller time intervals to groups of participants to the testing process; as they will be fragmented the distribution in time will normalize;
- closing the access to the application if it has been used by a certain number of users; as the participants to the testing process will know the access is

concurrent, they will try to fulfill their task as soon as possible.
-

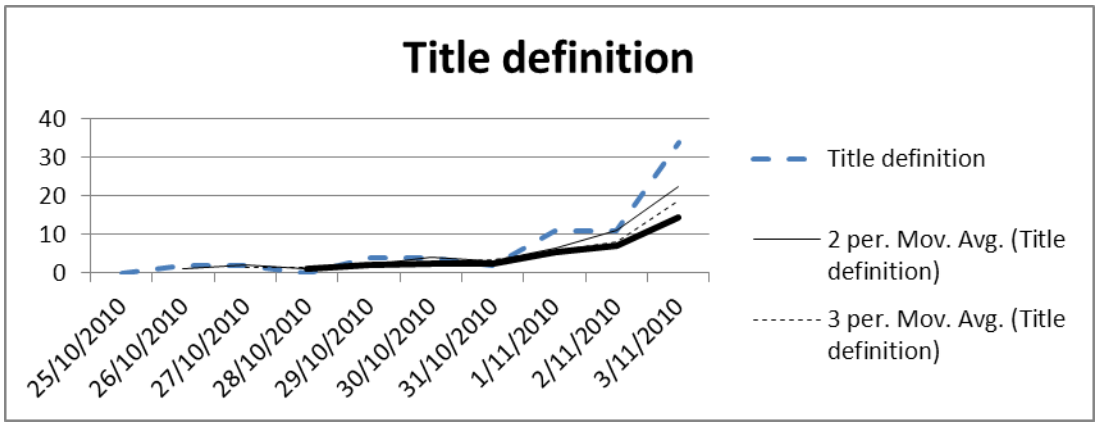

Fig. 4. Users' behavior for first component

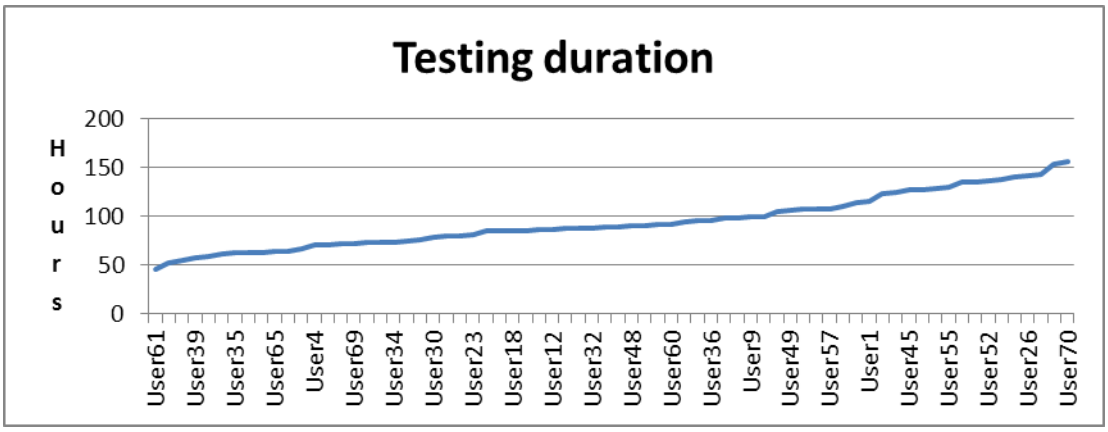

Fig. 5. - Hours between application hits

In Fig. 5 the time allocated by each user for the testing process is presented. The data were sorted ascending. The user with the lowest testing time scores about 45 hours. The user that accessed and tested the application the most used about 155 hours.
The recordings of the behavior of the users revealed that most users use the last day of the time interval associated to the task to do it. About seventy percent of the participants to the testing process ended it successfully by inserting a complete structured text entity.

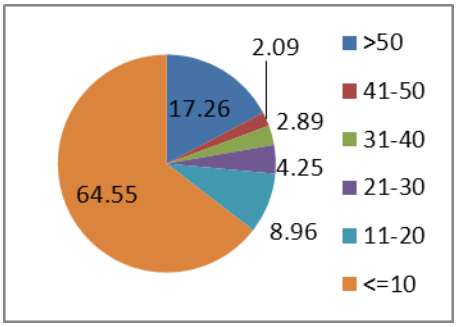

Fig. 6. Structure of interval lengths on groups of ten seconds

Fig. 6 shows the structure of interval lengths for groups with a difference of ten seconds between them. Almost two thirds of the intervals fall in the under ten seconds group. This shows that the users prefer to access the application at certain moments and use it intensively.

6 Users' tests and development errors

As the objective of the testing process is to identify faults in the logic or interface of the application the participants to the testing process reported abnormal behavior of the application. For the testing of the component responsible for the insertion of the abstract and keywords the participants to the testing process tried to insert problematic contents

almost three hundred times. They tested the functions that validate the rules that apply to the abstract and keywords. In Table 4 are given the rules that apply to the abstract and keywords and how many times they were tested by the users.

Table 4. Rules that apply to the abstract and keywords and number of tests

Rule	Number of tests
The abstract must be at most one hundred words long.	41
The abstract should be at least fifty words long.	5
Words from the title must exist in the abstract.	71
Words from the title must exist in the keywords.	126
All words from the abstract and keywords must be correct in English language.	173
There should be at least four keywords.	4
There should be at most ten keywords.	25
All keywords must be found in the abstract.	58
Keywords should have no duplicates.	25

The testing process totaled almost one hundred and seven hours for the abstract and keywords components.

Development errors were identified and reported by the participants to the testing process. These errors have been reproduced and identified by the developer. The identified and reported errors are:

- incorrect identification of similar text entities in the title definition form;
- unrecognized words specific to the domain;
- incomplete definition of the words separators;
- missing functionality of password recovery;
- incorrect analysis of the keywords for the edit form;
- incorrect selections in the View Paper State form;
- incomplete selections for the population of the selection lists from the forms for adding tables and figures;
- incomplete select syntax for full text insertion form chapters retrieval;
- the addition of source code in the full text caused problems to the users.

E1 – The incorrect identification of similar text entities was reported by a user that received the message that the title he tries to insert is similar with another title from the database, but the two titles were, undoubtedly, different. The correct function that established if the title is different from other titles from the database is given below.

E2 – There were unrecognized words specific to the domain. All users built structured text entities related to the informatics field. Many of them encountered words that were not recognized by the used dictionary for the English language. The problem was solved by completing the dictionary with words from the domain.

E3 – Incomplete definition of the words separators led to incorrect separation and analysis of words from title. In the validation routines the strings of the entity are split in order to obtain words that are analyzed. As the list of words separators was not complete initially, some of the words were not correctly separated and, as a consequence, were not validated. The cause of the error was easily identified as the messages delivered to users were clear and showed the location of the error. The list of words separators was updated and the application was retested to see the results of the modifications.

E4 – The missing functionality of password recovery was signaled by some of the users. The initial design included no password recovery feature for the testing users. After the problem was signaled by a few users, the feature was implemented. The users then had the possibility of receiving their username and password in the email they registered with. The form was implemented in ASP.NET, as the rest of the application.

E5 – Incorrect selections in the View Paper State form were made by the initial version

of the application. These were caused by the complex structure of multiple selections that depended on of another. The problem was solved by adding a new field to some tables in the database and simplifying the structure of the command used for retrieving the data.

E6 – Incomplete selections for the population of the selection lists from the forms for adding tables and figures were also caused by the mechanism used at first by the application to store multiple changes of data. The solving of the problem was the simplification and restructuration of the selection commands [11].

E7 – Incomplete select syntax for full text insertion chapters retrieval caused the users to have all the chapters of the structured text entity, but not in the correct order. The problem was corrected after the select command was completed with the order by clause.

E8 – Many users wanted to insert code in the text of the structured entity. The default behavior of the ASP.NET application is to halt the execution of the application at the detection of potentially harmful code. The problem was solved by changing the default behavior and encoding the text as HTML. This protected the integrity of the database and also allowed users to insert source code in the text of the structured entity. Future research includes the testing of the application using data generated with the use of genetic algorithms [12].

7 Conclusions

Structured text entities have many components. These are created by using words from vocabularies. There are rules that the components of the structured text entities must comply with. Rules for the structured text entities include the inclusion rules, the exclusion rules, the gradual approach rule, the homogeneity rule. Each author of structured text entities must comply with already existent rules or the ones defined by him.

Parallel testing is used as a mean of reaching more than one objective in short time duration. For the citizen oriented application for assisted development of structured text entities, the testing process reached the two proposed objectives: testing of the components and testing of full functionality. The testing was made in parallel. Two versions of the application were used. For the component testing time constraints were imposed. For the full functionality testing no restrictions were imposed. The parallel testing made possible the simultaneous identification of faults from components and interactions between them.

For this, time intervals for the tasks to be done must be defined. All actions of the users must be recorded along with additional information regarding the session, moment of time, details regarding the contents of the form the user is accessing, the controls accessed, errors received. An analysis of the time between actions is possible if the time moments are recorded. For this, the maximum and minimum values must be determined. The difference between the maximum and the minimum values is then split in the number of desired groups. The number of elements that fall in each group is then determined. If the elements concentrate in one group, that means the distribution of actions across the total time interval is regulate. If there are a significant number of elements in more than one group, it means that the users had a pattern of accessing the application. In the case in which the group containing the shortest intervals is large and another group is large as well, it means that the users worked in sessions at constant time intervals.

Future research is aiming at improving the percent of participants that finish successfully the testing process by identifying the causes that lead to failure.

Acknowledgement

This article is a result of the project „Doctoral Program and PhD Students in the education research and innovation triangle". This project is co funded by European Social Fund through The Sectorial Operational Programme for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies.

References

[1] Gh. Nosca, I. Ivan, S. Tcaciuc, O. Pârlog, R. Caciula, *Calitatea datelor*, Editura Inforec, Bucuresti 1999.

[2] I. Ivan, S. Capisizu, Gh. Nosca, *Auditul sistemelor informatice*, Editura ASE, Bucuresti, 2005.

[3] I. Ivan, E. Dumitrascu, D. Palaghiţă, D. Milodin, M. Popa, "Optimum criteria for developing defined structures," *Economic Informatics*, Vol. 12, No. 2, 2008, pg. 43 – 54.

[4] B. Vintila, S. Pavel, "Assisted Design, Development And Evaluation Of Citizen Oriented Collaborative Applications," *Journal of Applied Collaborative Systems*, vol. 2, no. 3, 2010, ISSN 2066-7450, pg. 64-72.

[5] B. Vintila, "Analyzing Structured Text Entities Using a Citizen Oriented Application," *Journal of Information Systems & Operations Management*, ISSN: 1843-4711, pp. 45-53, 2010.

[6] M. MacDonald, A. Freeman, *Pro ASP.NET 4 in C# 2010*, Apress, pp. 1616, 2010.

[7] A. Boehm, J. Murach, *Murach's ASP.NET 3.5 Web Programming with C# 2008*, Mike Murach & Associates, pg. 974, 2008.

[8] D. Esposito, *Programming Microsoft ASP.NET 3.5*, Microsoft Press, 2008, pg. 1128.

[9] V. Voineagu, A. Isaic-Maniu, C. Mitrut, T. Andrei, A. Costea, *Statistica, teorie si aplicatii*, Ed. CISON, pg. 303, 2010.

[10] E.M. Biji, *Statistica pentru economisti*, Ed. Economica, pg. 704, 2010.

[11] A. Velicanu, S. Olaru, "Optimizing spatial databases", *Informatica Economica*, Vol. 14, No. 2, pag. 61-71.

[12] L.A. Cotfas, A. Dioşteanu "Personalizarea serviciilor bazate pe locaţie utilizând algoritmi genetici şi agenţi inteligenţi," In *A 7-a Conferinţa Naţională de Interacţiune Om-Calculator*, 2010.

Ion IVAN has graduated the Faculty of Economic Computation and Economic Cybernetics in 1970. He holds a PhD diploma in Economics from 1978 and he had gone through all didactic positions since 1970 when he joined the staff of the Bucharest Academy of Economic Studies, teaching assistant in 1970, senior lecturer in 1978, assistant professor in 1991 and full professor in 1993. Currently he is full Professor of Economic Informatics within the Department of Computer Science in Economics at Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies. He is the author of more than 25 books and over 75 journal articles in the field of software quality management, software metrics and informatics audit. His work focuses on the analysis of quality of software applications.

Bogdan VINTILĂ graduated the Bucharest University of Economics, the Faculty of Cybernetics, Statistics and Economic Informatics. He is currently a PhD candidate in the field of Economic Informatics at University of Economics. He is interested in citizen oriented informatics applications, developing applications with large number of users and large data volumes, e-government, e-business, project management, applications' security and applications' quality characteristics.