Optimization of Distributed Software
ION IVAN, CATALIN BOJA, ADRIAN VISOIU, MIHAI DOINEA
Economic Informatics Department
Academy of Economic Studies
Romana Square No. 6, Bucharest
ROMANIA

Abstract: The paper analyses the software optimization process from the point of view of distributed applications. There are taken into consideration optimum criteria for this type of applications, structure optimization, source code optimization and maximizing the user satisfaction. The paper focuses on a particular type of distributed applications, one that is developed using Web technologies.
Key-words: optimization, software, distributed, optimum, criteria, code, structure

1. Distributed software

Information society is citizen oriented through a wide accessibility to hardware equipments, to Internet, to high interest e-applications in a secure environment. More than that, based on the quality standards implemented by the IT industry it aims to maximize the degree in which the citizen is satisfied by all this electronic services.

Distributed software presume that the following components CW1, CW2, ..., CW_n, are interacting through the links, references and calls at client-server level.

By components structure point of view, [2] the application is linear and simple, linear double connected, linear with multiple bindings and for each one of this class it is characterized by a sequential crossing of resources, dependent bindings between each one of the two components, arborescence, with assembly structure implemented with list of lists, lists with multiple connections, different arborescence structure and graphs.

From a Web perspective, a distributed application is a complex solution, referenced in figure 1, which includes:
- administrative systems;
- databases;
- resource access software;
- source texts for presentation and messages processing;
- human – computer component interface.

This architecture may be classified based on its components architecture [6], input data type, the scope or the problem to solve, methods used to access resources or implemented technologies [7].

For implementing the principles of distributed applications optimization it is considered the DistribEval application used to implement the evaluation process of different areas of knowledge. It uses the software instruments for processing and distributing data through the Internet and it is complementary to the educational process.

The complexity of the evaluation process implemented by DistribEval application requires a new architecture which will contain all the components presented in figure 1. The application’s structure is tree like dividing

Fig. 1 Components of Web Distributed application.
the components by type of process and type of user, figure 2.

![Application Processing Core](image)

Users | Processes
---|---
Evaluators | Project Owners | Authentication | Upload

Fig. 2 Structure of *DistribEval* application.

The interaction with users and the process of collecting the input data are done by using a large set of components.

### 2. Optimum criteria

Distributed applications oriented on implementing and solving problems of electronic evaluation processes, are characterized by optimum criteria, [1]:

- maximization of continuity level;
- minimization of the length of processing stream;
- minimization of the risk of having the stream unwind;
- maximization of the degree in which the users are satisfied;
- maximization of the level of flexibility in storing and searching data;
- maximization of the transparence level;
- maximization of the degree in which the data are reused;
- maximization of the global level of software quality;
- minimization of the time to access resources;
- shortcutting the way to selected resources;
- standardizing the way of inputting data;
- minimizing the input data volume;
- structure homogeneity associated to the modules;
- homogeneity maximization of simple connections between modules;
- minimization of application complexities;
- maximization of security level.

From the optimum criteria mentioned above will be selected a small number of them. The target of the distributed application optimization process is to minimize the overall effort and to reduce the costs by focusing on those characteristics that have the highest impact on the application quality level. The optimization process is done in connection with selected optimum criteria, step by step, leading to a multi-criteria optimization process in consecutive steps.

The continuation of the improving iterative process lids to a stabile solution which can maximize an optimum aggregate criteria.

### 3. Structure optimization

Applications components which interact directly with users and which affect directly their level of satisfaction are represented by:

- the interface which describes the manner the components are connected and which represent the way of access to different facilities: search, visualization and electronic payment;
- application structure which refers to the way the modules are connected between each others so they can form an uniform entity characterized by a limited number of paths;
- the processing data modules, the data volume and complexity of processing processes affect directly and significantly the time in which the results are obtained.

The analyzed distributed application is designed following the next steps:

- defining target group;
- defining the problem to be solved;
- defining the layers of security and access to the application resources;
- defining functions and application’s structure;
- designing the data flow diagrams;
- choosing the language for programming and database;
- programming;
- code testing;
- installing the application on server;
- normally utilization of application;
- maintenance process;
- reengineering process.

For the initial structure of application and for the optimized on there are determined the levels of complexity. The scope of the process is that the effect of the optimization process will lid to a decrease of the complexity level for final structure. Figure 3 describes the water fall model for building the *DistribEval*
application highlighting the area affected by the optimization process. This supports the idea that the software optimization process must apply on a functional product and the assessment process must take into consideration results of real running tests and not simulated ones.

Determination of the target group identity is strictly related with the definition process of application objective. The identity of target group must be well described, not ambiguous and must point out the scope for which the application is made.

The DistribEval application, intended for on-line evaluation of educational projects, defines a target group that is formed by two important elements, the project’s owners and the evaluators.

The common characteristics for the two types of users are described by:
- medium experience level of Internet utilization that will allow users to navigate in a tree like Web pages structure and to complete electronic forms;
- knowledge regarding secure access to Web resources through authentication pages and by using secure data transmission protocols;
- age between 23 and 60 years old;
- high level of knowledge in the field of evaluated projects.

Based on the sample analysis process, there are identified:
- structural elements;
- quality characteristics;
- the way information are organized;
- routines used to identify users.

The study of users collectivity is not made taking into consideration each one of the individuals because their number is too great to be analyzed.

The target group is reduced to a sample considered representative. Its size is established using statistic methods to determine the volume of the survey, [9]:

\[ n_{\text{sample}} = \frac{z_{\alpha}^2 \cdot \sigma_{\text{group}}^2}{\Delta_x^2}, \]  

(1)

for simple repeated survey;

\[ n_{\text{sample}} = \frac{N_{\text{group}} \cdot z^2_{\alpha} \cdot \sigma_{\text{group}}^2}{(N - 1)\Delta_x^2 + z^2_{\alpha} \cdot \sigma_{\text{group}}^2}, \]  

(2)

for simple unrepeated survey; where:
- \( n_{\text{sample}} \) – minimum dimension of sample required for the significance degree;
- \( N_{\text{group}} \) – real dimension of target group;
- \( z_{\alpha} \) – value found in Laplace tables for \( P = 1 - \alpha \) probability;
- \( \sigma_{\text{group}}^2 \) – value of statistical variance;
- \( \Delta_x \) – limit error, measures how much the sample average differs from the group average.
For DistribEval application, the collectivity is formed by the total number of evaluators and the total number of projects. The evaluator’s population has \( N_{\text{eval}} = 430 \) individuals.

The owner’s project population has \( N_{\text{proiecte}} = 1100 \) individuals.

Target group of application DistribEval is formed by \( N_{\text{DistribEval}} = 1530 \) individuals.

In the end it is defined a sample of \( n_{\text{sample}}^{\text{WebEval}} = 79 \) individuals who will test the application and measure the its level of quality. The \( n_{\text{quantion}}^{\text{WebEval}} \) indicator is calculated for \( z_{\alpha} = 0.05 \), the limit error concerning the average level of experience in using the on-line application \( \Delta x = 34 \) and a statistical variance equal to \( \sigma_{\text{Grup}}^2 = 4043 \).

Defining and building the target group facilitates the implementation of the application’s entire life cycle because it represents an important and required information source for development process in its analysis, project, test and implementation phases.

To optimize the DistribEval application structure means to:
- minimize the number of disposing modules;
- maximize the degree of modules homogeneity;
- maximize the level of flux automation.

The optimization process has an iterative quality. It will begin from an initial structure of application. It will be launched in execution for a period of time. There will be done measurements. These results are used to produce modifications to diminish the levels of variables which affect negatively the application performances.

### 4. Source code optimization

Optimizing source code aims the optimization of the source measured by amount of bytes occupied by the files, greatly influencing loading speed. The processing chain duration at user level is influenced by the multi-level structure, component referencing, component nature and referring frequency. That is why, minimizing the number of links from a processing chain, minimizing the length of each source file, homogenizing the components referred in the chain and minimizing the number of levels the chain is lengthen out are all aimed by source code optimization.

Some of the methods used for source code optimization, [2], [5], [4], in DistribEval are:
- trimming white space and unused characters;
- avoiding sending large amounts of data;
- using CSS style sheets or XSL;
- removing automatically included metadata;
- using techniques for memory consumption reduction;
- building complex menus based on graphical components;
- minimizing the duration of complex computations;
- implementing improved algorithms;
- implementing base components with minimum loading level;
- choosing an efficient application component access technology;
- implementing lossless image compression algorithms.

For the studied application, used for online project evaluation, the optimization process is influenced by:
- the type of process the application is built for;
- choosing programming language and database type;
- choosing a high performance authentication mode;
- obtaining a high level of accuracy for input data;
- external source referencing;
- assuring a homogenous way for referring components.

Table 1 describes the effects obtained by applying optimization methods to DistribEval web application.

<table>
<thead>
<tr>
<th>Optimum criterion</th>
<th>Initial version</th>
<th>Indicator</th>
<th>Improved version</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input data accuracy</td>
<td>inserting data into a single container</td>
<td>GNC ( \in [0;1] ) E( FU = 1 ); G( NV \in [0;1] )</td>
<td>- container defined for each attribute</td>
<td>GNC = 1 E( FU = \text{nat} ); G( NV = 1 )</td>
</tr>
<tr>
<td>Code size</td>
<td>specifications implementation</td>
<td>NLC = 15475</td>
<td>- improved specification implementation</td>
<td>NLC = 13550</td>
</tr>
</tbody>
</table>
where \( nat \) represents the number of evaluated attributes for each project.

The indicators used in table no. 1 to emphasize the effects of the optimization process are:

\[
GNC = \frac{NCC}{NTC} \quad (3)
\]

- **GNC** – weight of the fields defined in that procedure, filled in when submitting the form; the indicator range is \([0;1]\);
- **NCC** – number of filled in fields;
- **NTC** – total number of fields to be filled in;

\[
EFU = NCC \quad (4)
\]

- **EFU** – effort for using a certain component given by the number of web controls the user has to interact with;
- **NCC** – number of editable fields in the component;

\[
GNV = \frac{NCV}{NTC} \quad (5)
\]

- **GNV** – weight of the fields defined by the evaluation procedure that are also validated; the indicator range is \([0;1]\);
- **NCV** – number of validated fields;
- **NTC** – total number of fields to be filled in;

\[
GCA = \frac{NCMPA}{NTCMP} \quad (6)
\]

- **GCA** – the weight of the application components in which explicit authentication is done by reentering the data;
- **NCMPA** – number of components with authentified access using web forms;
- **NTCMP** – total number of application component;

The development cycle for *DistribEval* web application lasted 27 months and consists of 13550 source code lines. The duration of source code writing is 75% of the developing cycle, testing representing 25% of the developing cycle, due to the fact that the application is operated online and by heterogeneous users. Its quality depends strictly on the quality of user–application interactions. *DistribEval* is a distributed application based on the principle of decentralized database loading process, over 80% of the database contents is entered by 90% percent of the users.

Specialized application operators introduce only nonvolatile data meaning less than 20% of the database content.

5. Maximizing the degree of user satisfaction

Maximizing the degree of user satisfaction is the main objective for user oriented web applications. The degree of satisfaction \( GS \) is defined based on the number of accesses analysis

\[
GS = \frac{NA}{NTA} \quad (7)
\]

where:
- **GS** – degree of user satisfaction;
- **NA** – number of accesses for the working session;
- **NTA** – total of accesses.

A successful access means reaching an OK ending leaf from the application tree, by which the citizen certifies the fact that he is satisfied with the obtained result.

The first accessing is given by opening the home page of the application.

The maximization of user satisfaction degree is obtained by:
- displaying information gradually;
- using an understandable vocabulary;
- using the personal ID code as unique access key;
- entering username and password as it is done for the majority of web applications;
- including help files;
- doing corrections with little effort;
- partial saving of input data;
- loading the electronic format of the projects;
- sending evaluation results real time;

The average satisfaction degree, \( GSM \) is defined by:
\[
GSM = \frac{GSM_e + GSM_u + GSM_a}{3} \quad (8)
\]

where:

- \( GSM_e \) – the average satisfaction degree of the evaluators;
- \( GSM_u \) – the average satisfaction degree of the user as author of that project;
- \( GSM_a \) – the average satisfaction degree of the administrators.

Conclusions

Distributed application optimization produces positive effects for all users. If optimization refers to only one criterion, its effect is only partial. That is why, many criteria must be taken into account and the optimization process must be transformed into an objective the whole developing team must aim. A stage in the developing process of a distributed application is considered. Levels of performance characteristics are recorded. Optimization is performed according to a certain criterion. Measurements are redone assessing the effects generated by optimization. The process continues switching from performing optimization to recording measurements. The results obtained from implementing distributed applications must contribute to optimization process improvement for the following applications built by the developing team.

References


