

## Excerpt

A projekt azonosító száma: GVOP-3.1.1.-2004-05-0345/3.0

A projekt címe: Megbízhatóbb, nyílt forráskódra épülő irodai szoftver előállítása céljából az OpenOffice.org nyílt forráskódú szoftver minőségének szisztematikus vizsgálata és a kód javítása

A projekt rövid címe:  
OpenOffice++: Nyílt forráskódú szoftverek minőségének javítása



# Contents

<b>1</b>	<b>Executive summary</b>	<b>1</b>
<b>2</b>	<b>Quality assessment methodology</b>	<b>2</b>
2.1	Overview . . . . .	2
2.2	Basic measurements and problem lists . . . . .	2
2.3	Baselines . . . . .	3
<b>3</b>	<b>System overview</b>	<b>5</b>
3.1	System Size . . . . .	5
3.2	Reusage, OOP design at first sight . . . . .	5
3.3	Evolution . . . . .	6
<b>4</b>	<b>Complexity analysis</b>	<b>9</b>
4.1	Most complex entities . . . . .	9
4.2	Comparison with baselines . . . . .	11
4.3	Evolution . . . . .	11
<b>5</b>	<b>Interdependence analysis</b>	<b>14</b>
5.1	Basic coupling measurement . . . . .	14
5.2	Comparison with baselines . . . . .	16
5.3	Evolution . . . . .	17
<b>6</b>	<b>Design problems</b>	<b>20</b>
6.1	Bad smells . . . . .	20
6.2	Evolution . . . . .	20
<b>7</b>	<b>Bugs and coding problems</b>	<b>23</b>
7.1	Rule violations . . . . .	23
7.2	Overview of specific rule violations . . . . .	24
7.3	Evolution . . . . .	26
<b>8</b>	<b>Summary of most problematic entities</b>	<b>29</b>
	<b>References</b>	<b>30</b>
	<b>Appendices</b>	<b>31</b>
<b>A</b>	<b>Metric descriptions</b>	<b>31</b>
<b>B</b>	<b>Bad smell descriptions</b>	<b>38</b>
<b>C</b>	<b>General rule descriptions</b>	<b>40</b>



# 1 Executive summary

This document is the report of the quality assessment of seven versions of the OpenOffice source code. The assessment has been performed by the Department of Software Engineering at the University of Szeged, Hungary.

This report is meant to be continuously updated as our assessment work with the source progresses. In its current form, it is the summarization of the results of an in-depth analysis performed on the seven versions of the OpenOffice source code. In the subsequent parts of this document we will refer to these seven versions with their version number. Table 1 shows some information about the seven versions.

Version	Date
1.1.4	December 22, 2004
1.1.5	September 14, 2005
2.0.0	October 20, 2005
2.0.1	December 21, 2005
2.0.2	March 8, 2006
2.0.3	June 29, 2006
2.0.4	October 13, 2006

Table 1: Summary of the analyzed versions and their release dates

The report incorporates two dimensions of source code quality. First, detailed measurable attributes (i. e., *metrics*) are computed, analyzed and presented, which can contribute to the quality, for instance size, complexity, and amount of interdependencies. Second, the source code is analyzed against possible design and coding flaws, which most importantly include bugs, dangerous constructs, and other possible design and coding problems (including coding *rule violations* and so-called *bad smells*). Where appropriate, comparison to metric baselines is provided based on the analysis of a number of other software systems. The methodology applied is described in detail in Section 2.

For the detailed analysis of the subject system a source code analysis framework called Columbus (<http://www.frontendart.com/>) has been used. It incorporates a detailed code analysis engine, and a metric storage and query subsystem, using which the measurement data and charts has been produced in this report.

The most important outcomes of this report are the identified classes and functions in the system, which are problematic from a specific point of view. Furthermore, classes and functions that are found to be problematic in more than one aspect are probably the most critical ones, so these are separately summarized (see Table 2 – the meaning of the columns will be explained later in the following sections). These need special attention in the future evolution of the system, like with the distribution of maintenance and testing efforts.

A significant part of the problems identified are related to serious coding problems (bugs, in fact), which will lead to certain or very probable problems when executed. For example, there are hundreds of general coding problems in the categories of bugs and dangerous constructs and memory handling problems.

In the next sections we will present results which show that **OpenOffice.org** has a source code with surprisingly **good complexity** but with **risky interdependencies** (see Tables 11 and 15, which deserve special attention).

Class	WMC	CBO	NOI	NII	BDC	MHP
SwDoc	4236	618	1837			
SwHTMLParser	2627		941			
ScDocument	2549	279	1107			
Window	2146			7240		
SwWW8ImplReader	2102	395	1020			
binfilter::SwDoc		407	1060			
SwView		387	1289			
ScDocShell		326	859			
binfilter::Sw3IoImp			1191	108		
ImpEditEngine			964		40	

Table 2: Most problematic classes in **OpenOffice.org**