



FAGAMMON
FACULDADE PRESBITERIANA GAMMON

Relatório Técnico

Análise da evolução do código fonte do software Dolibarr

Qualidade de software

Professor: Victor Hugo Santiago

Alunos: Eduardo Augusto Santana Sales
Everton Henrique Batista Vilela

Lavras – MG

2019

1. Introdução

Hoje com a constante evolução tecnológica, os softwares estão sendo desenvolvidos cada vez mais rápido e utilizando cada vez mais metodologias ágeis, testes de aceitação, teste funcionais e de stress, e várias outras ferramentas afim de garantir a qualidade e excelência do que foi ou está sendo desenvolvido.

A melhoria do processo de software é um objetivo fundamental para as organizações e deve estar baseada em medições. Entretanto, definir, coletar e analisar um conjunto de métricas não é uma tarefa fácil.

Neste documento descreveremos uma abordagem da avaliação do software Dolibarr coletando métricas a partir das ferramentas PHPLOC E PHP_CodeSniffer. Será utilizado 5 versões do software Dolibarr afim de realizar uma análise de evolução do código fonte da aplicação.

2. Descrição das ferramentas PHPLOC e PHP_CodeSniffer

Neste capítulo, as ferramentas, medidas e recursos utilizados serão descritos.

2.1 Descrição da ferramenta - PHPLOC

A ferramenta phploc faz mais do que apenas contar linhas de código, ele conta toda uma seleção de recursos de uma base de código e os fornece como um relatório diretamente no console também com a opção de exportar o relatório para um arquivo.

Par usar está ferramenta no windows é necessário instalar o Xampp e o Composer, que será descrito na seção 2.3.

Para instalar o PHPLOC basta abrir o CMD e executar o seguinte comando

```
composer global require 'phploc/phploc=*
```

Após instalação basta abrir o cmd na pasta do projeto e executar o comando

```
phploc.
```

Abaixo temos um exemplo da saída que é gerada pelo software onde é possível observar quais métricas serão coletadas para análise.

1	phploc 4.0.1 by Sebastian Bergmann.
2	
3	Directories.....297
4	Files.....1317
5	
6	▼ Size
7	.. Lines of Code (LOC).....532296
8	.. Comment Lines of Code (CLOC).....111618 (20.97%)
9	.. Non-Comment Lines of Code (NCLOC).....420678 (79.03%)
10	▼ .. Logical Lines of Code (LLOC).....209718 (39.40%)
11	... Classes.....77240 (36.83%)
12	... Functions.....17780 (8.48%)
13	... Not in classes or functions.....114698 (54.69%)
14	
15	▼ Cyclomatic Complexity
16	.. Average Complexity per LLOC.....0.33
17	.. Average Complexity per Class.....49.27
18	.. Average Complexity per Method.....7.24
19	
20	▼ Dependencies
21	▼ .. Global Accesses.....23347
22	... Global Constants.....13363 (57.24%)
23	... Global Variables.....2676 (11.46%)
24	... Super-Global Variables.....7308 (31.30%)
25	.. Attribute Accesses.....118957
26	.. Method Calls.....6003
27	
28	▼ Structure
29	.. Namespaces.....0
30	.. Interfaces.....2
31	.. Traits.....0
32	▼ .. Classes.....480
33	... Abstract Classes.....51 (10.62%)
34	... Concrete Classes.....429 (89.38%)
35	▼ .. Methods.....3787
36	▼ ... Scope
37 Non-Static Methods.....3699 (97.68%)
38 Static Methods.....88 (2.32%)
39	▼ ... Visibility
40 Public Methods.....3729 (98.47%)
41 Non-Public Methods.....58 (1.53%)

2.2 Descrição da ferramenta PHP_CodeSniffer

PHP_CodeSniffer é uma ferramenta que auxilia no desenvolvimento de software e garante que seu código permaneça limpo e consistente.

PHP_CodeSniffer é composto por um conjunto de dois scripts PHP:

O script PHPCS é o principal, onde avalia projetos em PHP, JavaScript e CSS para detectar violações de um padrão de codificação definido.

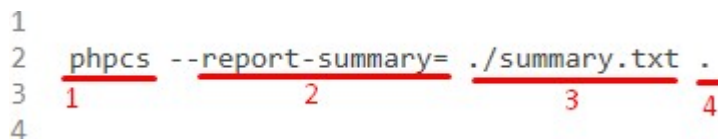
O segundo script PHPCBF tem a função de corrigir automaticamente as violações de padrões de codificação. Esta função não será utilizada, pois faremos somente a análise do código para se ter informações da evolução do software.

Para usar esta ferramenta no windows é necessário instalar o Xampp e o Composer, que será descrito na seção 2.3.

Para instalar o PHPLOC basta abrir o CMD e executar o seguinte comando

```
composer global require "squizlabs/php_codesniffer=*"
```

Após instalação basta abrir o CMD na pasta do projeto e executar o comando:



```
1  
2 phpcs --report-summary= ./summary.txt .  
3 1 2 3 4  
4
```

Legenda das numerações na imagem:

1. Phpcs □ Chamada principal do programa
2. --report-info □ Tipo de análise que será feita no software
3. =./info.txt □ Pasta onde será salvo o relatório, nome do arquivo que será gerado e tipo de saída, neste caso será .TXT
4. . □ Como o console foi aberto

Para análise será avaliada neste documento as seguintes chamadas para avaliar o software Dolibarr.

Na linha número 2 da imagem acima temos a chamada `PHPCS --REPORT-SUMMARY` que gera um relatório mostrando todos os arquivos que contem erro ou avisos.

Para informação de todos relatórios que podem ser gerados por essa ferramenta, acessar a documentação oficial da ferramenta

https://github.com/squizlabs/PHP_CodeSniffer/wiki/Reporting#printing-full-and-summary-reports

Nas imagens abaixo podemos observar um exemplo da saída gerada pela ferramenta.

Relatório do tipo SUMMARY

```
1
2 PHP·CODE·SNIFFER·REPORT·SUMMARY
3
4 FILE····· ERRORS· WARNINGS
5 -----
6 ...e-2\Nova-pasta\dolibarr-4.0.0\htdocs\disabled.php·2·····0
7 ...e-2\Nova-pasta\dolibarr-4.0.0\htdocs\document.php·118····26
8 ...Nova-pasta\dolibarr-4.0.0\htdocs\filefunc.inc.php·235····84
9 ...ware-2\Nova-pasta\dolibarr-4.0.0\htdocs\index.php·436····80
10 ...e-2\Nova-pasta\dolibarr-4.0.0\htdocs\main.inc.php·1793····432
11 ...2\Nova-pasta\dolibarr-4.0.0\htdocs\master.inc.php·289····43
12 ...2\Nova-pasta\dolibarr-4.0.0\htdocs\viewimage.php·70····33
13 ...libarr-4.0.0\htdocs\accountancy\admin\account.php·282····42
14 ...libarr-4.0.0\htdocs\accountancy\admin\card.php·520····24
15 ...arr-4.0.0\htdocs\accountancy\admin\categories.php·113····8
16 ...olibarr-4.0.0\htdocs\accountancy\admin\export.php·200····16
17
18
19 ...ibarr-4.0.0\htdocs\webservices\server_contact.php·934····30
20 ...ibarr-4.0.0\htdocs\webservices\server_invoice.php·920····71
21 ...olibarr-4.0.0\htdocs\webservices\server_order.php·1507····47
22 ...olibarr-4.0.0\htdocs\webservices\server_other.php·373····15
23 ...0\htdocs\webservices\server_productorservice.php·1022····92
24 ...ibarr-4.0.0\htdocs\webservices\server_project.php·91····23
25 ...0\htdocs\webservices\server_supplier_invoice.php·376····21
26 ...rr-4.0.0\htdocs\webservices\server_thirdparty.php·769····50
27 ...dolibarr-4.0.0\htdocs\webservices\server_user.php·972····34
28 ...dolibarr-4.0.0\htdocs\webservices\admin\index.php·64····13
29 ...pasta\dolibarr-4.0.0\htdocs\websites\frametop.php·9·····0
30 ...va-pasta\dolibarr-4.0.0\htdocs\websites\index.php·337····118
31 ...arr-4.0.0\htdocs\websites\class\website.class.php·1061····37
32 ...4.0.0\htdocs\websites\class\websitepage.class.php·1043····42
33 -----
34 A·TOTAL·OF·647738·ERRORS·AND·59038·WARNINGS·WERE·FOUND·IN·1338·FILES
35 -----
36 PHPCBF·CAN·FIX·622541·OF·THESE·SNIFF·VIOLATIONS·AUTOMATICALLY
37 -----
```


2.3. Recursos utilizados

2.3.1 Xampp

Muitas pessoas sabem por experiência própria que não é fácil instalar um servidor web Apache e torna-se mais difícil se você quer acrescentar MySQL, PHP. O objetivo do XAMPP é construir uma distribuição fácil de instalar para desenvolvedores entrarem no mundo do Apache. Para torná-lo conveniente para os desenvolvedores, o XAMPP é configurado com todos os recursos ativados. Do ponto de vista do XAMPP, o uso comercial também é gratuito. Há atualmente distribuições para Windows, Linux e OS X

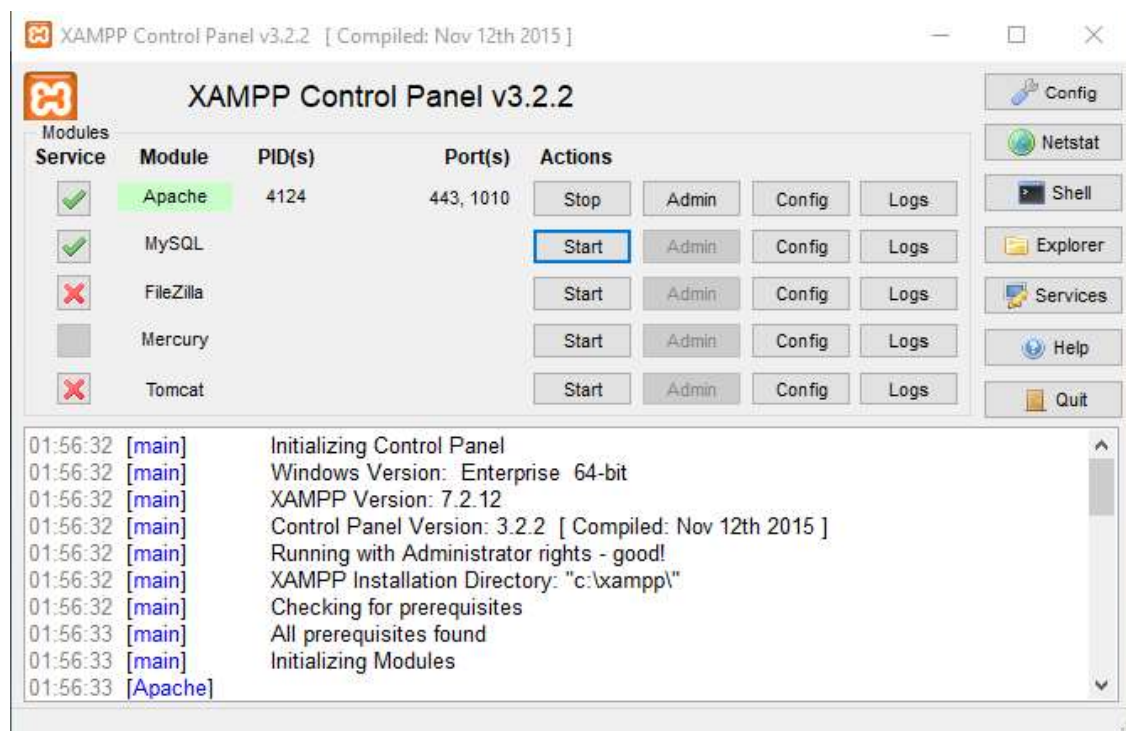
Instalando o Xampp

Acessar site oficial da ferramenta e baixar a versão mais atual

[Baixar XAMPP](#)

Após baixar, basta executar o instalador e seguir instruções na tela.

Após instalado a tela inicial da aplicação será esta:



2.3.2 Composer

Instalando o

Composer

Requerimentos

O composer requer o php 5.3.2 ou superior para rodar. Algumas configurações a mais serão necessárias com respeito ao php, porém o instalador irá avisar sobre eventuais incompatibilidades.

Composer é uma ferramenta para gerenciamento de dependências para o PHP que vem ganhando espaço e se tornando cada vez mais indispensável.

Com algumas poucas linhas de configurações você define todas as bibliotecas de terceiros ou mesmo suas que deseja/precisa utilizar em seu projeto, o composer encarrega-se de baixá-las e criar um autoloader deixando-as prontas para uso.

O composer é multiPlataforma e é feito para funcionar igualmente nas três plataformas existentes: Windows, Linux e OSX.

Instalação no

Windows Usando o

instalador

Esta maneira é rápida e simples, para se instalar o composer em seu windows. Baixe e execute o [Composer-Setup.exe](#), ele vai instalar a versão mais recente e configurar o path do windows para você, permitindo que você execute o composer em qualquer diretório a partir da linha de comando.

3. Sistema de software avaliado

Dolibarr ERP/CRM

Você ativa apenas o recurso de que precisa. Então, seja qual for a sua necessidade de gestão de negócios (vendas, recursos humanos, logística, estoque, faturamento, contabilidade, fabricação, etc), você é capaz de configurar o aplicativo para atender às suas necessidades. A integração entre os recursos / módulos que você decide usar está pronta 'na caixa', portanto, mesmo sem personalização, os usuários estão imediatamente prontos para trabalhar e não precisam fazer nenhuma entrada modificação no sistema.

Como as atualizações da nova versão são, por padrão, integradas ao processo de desenvolvimento, você pode atualizar a qualquer momento para a última versão, qualquer que seja sua versão atual, sem perder nenhum dado. Assim, os usuários sempre beneficiam os recursos e inovações mais recentes.

Um modelo FOSS (Free Open Source Software)

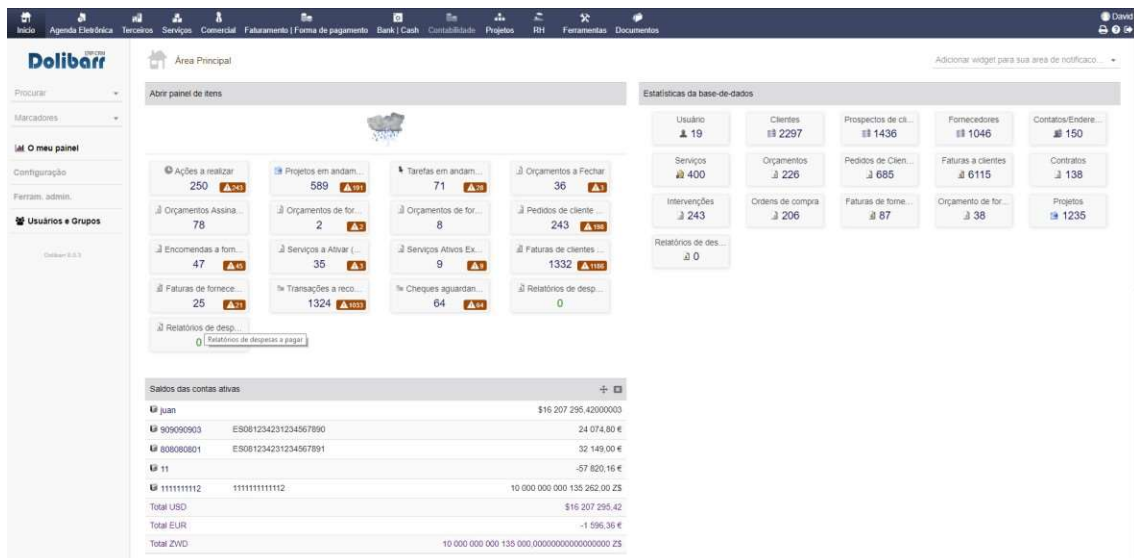
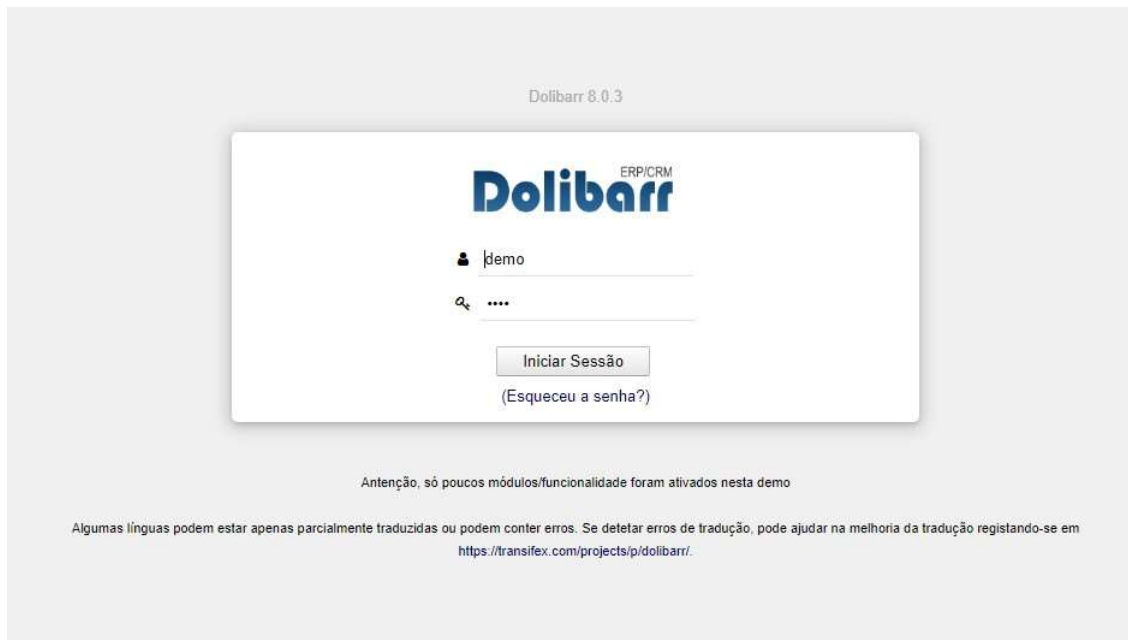
Por utilizar um modelo FOSS este software oferece uma solução competitiva: vários milhares de desenvolvedores, testadores e tradutores trabalham no projeto. É por isso Dolibarr é um software que está sempre na vanguarda da inovação.

Site oficial <https://www.dolibarr.org>

Instalação do software

Basta acessar o site da aplicação baixar o instalador e seguir instruções em tela [Dolibarr-installer](#)

Após instalar e configurar a aplicação seguindo as instruções a página inicial será esta:



4. Experimentos e Análises

Neste capítulo, são apresentados os experimentos realizados juntamente com a configuração do ambiente, resultados e análises.

4.1 Configuração do ambiente

Os experimentos foram realizados no ambiente apresentado na tabela abaixo:

Processador	Intel Core i5-2430M de 2.40GHz com 3mb de cache
HD	SSD 120GB Leitura até 500MB/s – Gravação até 450MB/s
Memória RAM	8GB DDR 3
Sistema Operacional	Microsoft Windows 10 Enterprise
Xampp	3.2.2
PHP	7.2.12
Composer	1.7.3
PHPLOC	4.0.1 by SebastianBergmann
PHP_CodeSniffer	version 3.3.2 (stable)

4.2 Experimentos

As versões do software foram obtidas no GitHub por meio de download, foi baixado as seguintes versões do software dolibarr 4.0.0, 5.0.0, 6.0.0, 7.0.0, 8.0.0 para cada versão baixada foi gerado 2 relatórios, 1 pelo PHP_CodeSniffer e 1 pelo PHPLOC

Link para acessar todas as versões no GitHub

<https://github.com/Eduardossales/Dolibarr>

4.3 Análise quantitativa

Após a coleta dos dados e geração dos gráficos, foram realizadas análises quantitativas avaliando a evolução dos dados a cada versão. Nas análises realizadas, foram considerados os valores totais obtidos com a ferramenta PHPLOC, a média de medidas em relação à complexidade ciclomática, o número de linhas de métodos, classes e funções.

A ferramenta PHP_CodeSniffer foi usada para coletar a quantidade de erros e avisos agrupadas por arquivos.

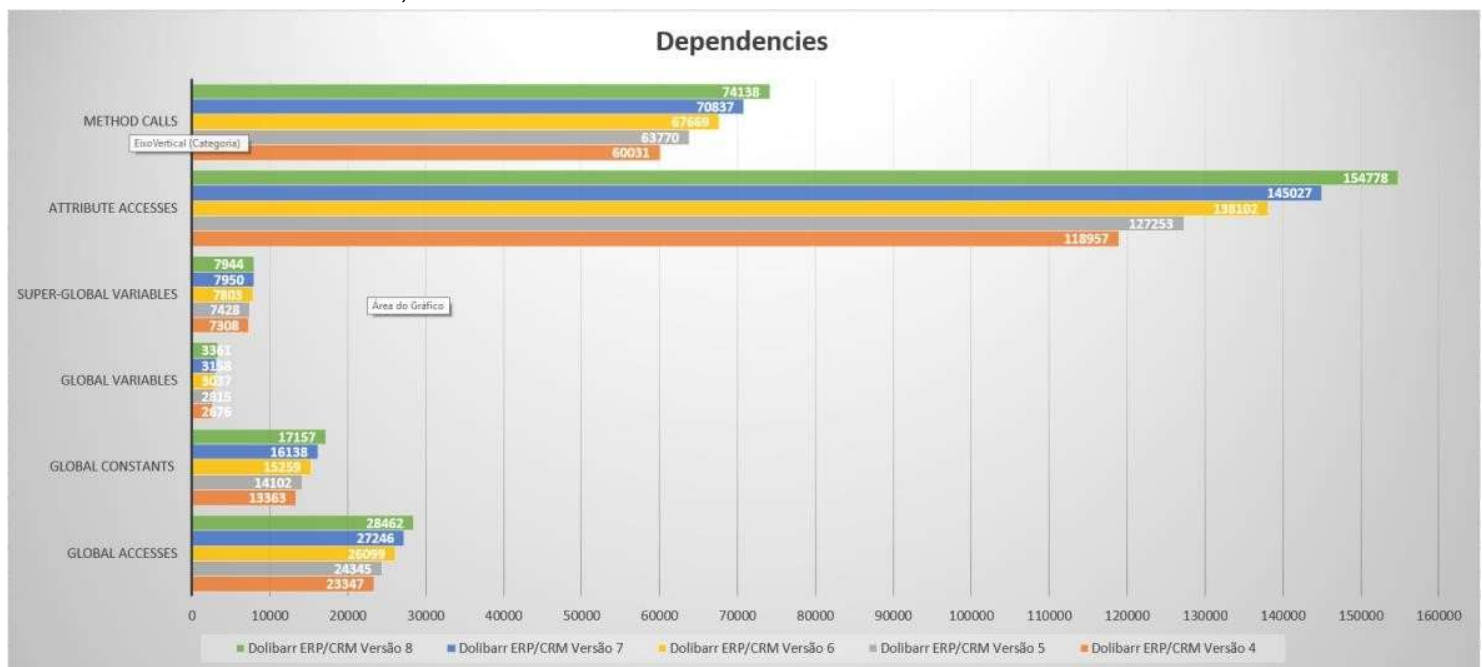
4.3.1 Analise do PHPLOC

Na análise realizada pela ferramenta PHPLOC foi possível observar que todos as métricas analisadas foram crescentes a cada versão, mostrando que a cada versão do software que é lançada o número de linhas, funções e complexidade ciclomática é crescente.

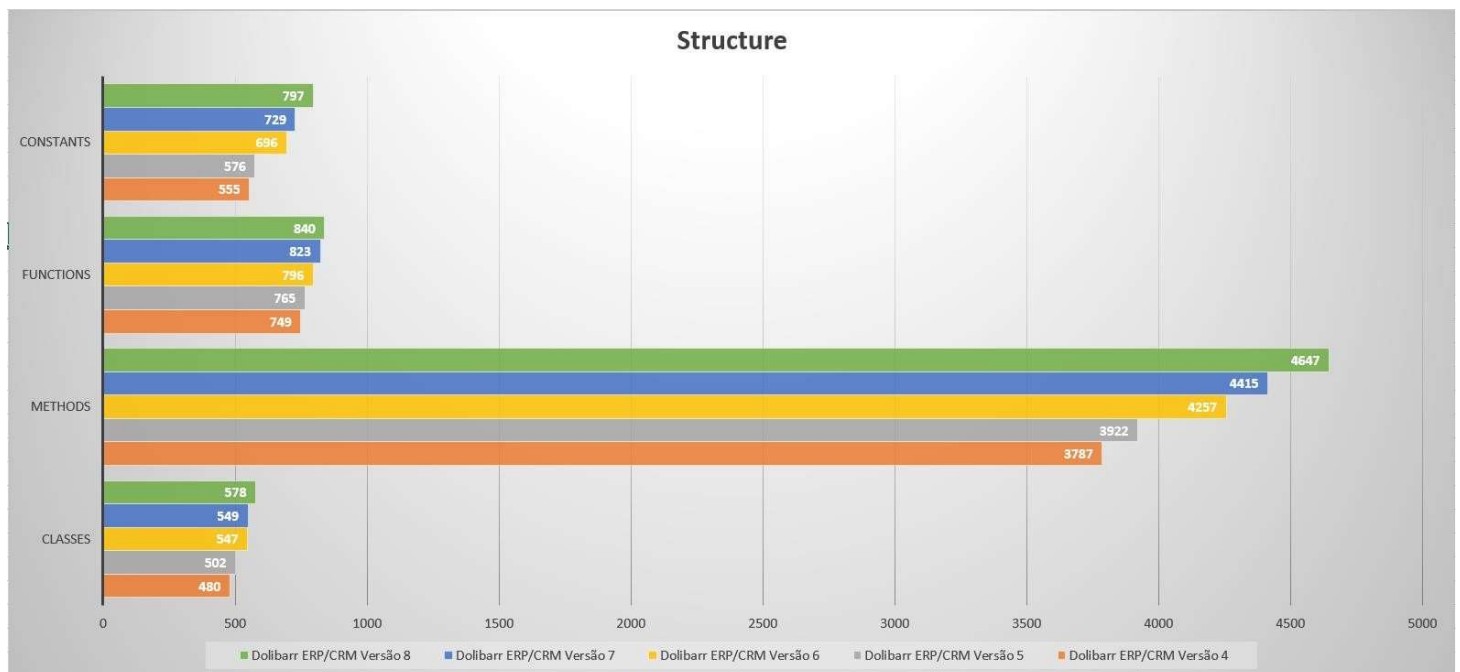
Dados gerados pelo software PHPLOC

Dolibarr ERP/CRM					
Medidas	Versão				
	4	5	6	7	8
Directories	297	301	342	346	362
Files	1317	1356	1464	1506	1580
Lines of Code (LOC)	532296	565375	614285	647475	686461
Comment Lines of Code (CLOC)	111618	118517	129385	135989	145133
Non-Comment Lines of Code (NCLOC)	420678	446858	484900	511486	541328
Logical Lines of Code (LLOC)	209718	223074	240579	253390	266856
Classes	77240	81355	87636	92726	99570
Functions	17780	18287	19457	20726	21713
Not in classes or functions	114698	123432	133486	139938	145573
Cyclomatic Complexity					
Average Complexity per LLOC	0,33	0,34	0,34	0,35	0,35
Average Complexity per Class	49,27	50,96	51,16	55,7	57,28
Average Complexity per Method	7,24	7,52	7,56	7,92	8,12
Dependencies					
Global Accesses	23347	24345	26099	27246	28462
Global Constants	13363	14102	15259	16138	17157
Global Variables	2676	2815	3037	3158	3361
Super-Global Variables	7308	7428	7803	7950	7944
Attribute Accesses	118957	127253	138102	145027	154778
Method Calls	60031	63770	67669	70837	74138
Structure					
Classes	480	502	547	549	578
Methods	3787	3922	4257	4415	4647
Functions	749	765	796	823	840
Constants	555	576	696	729	797

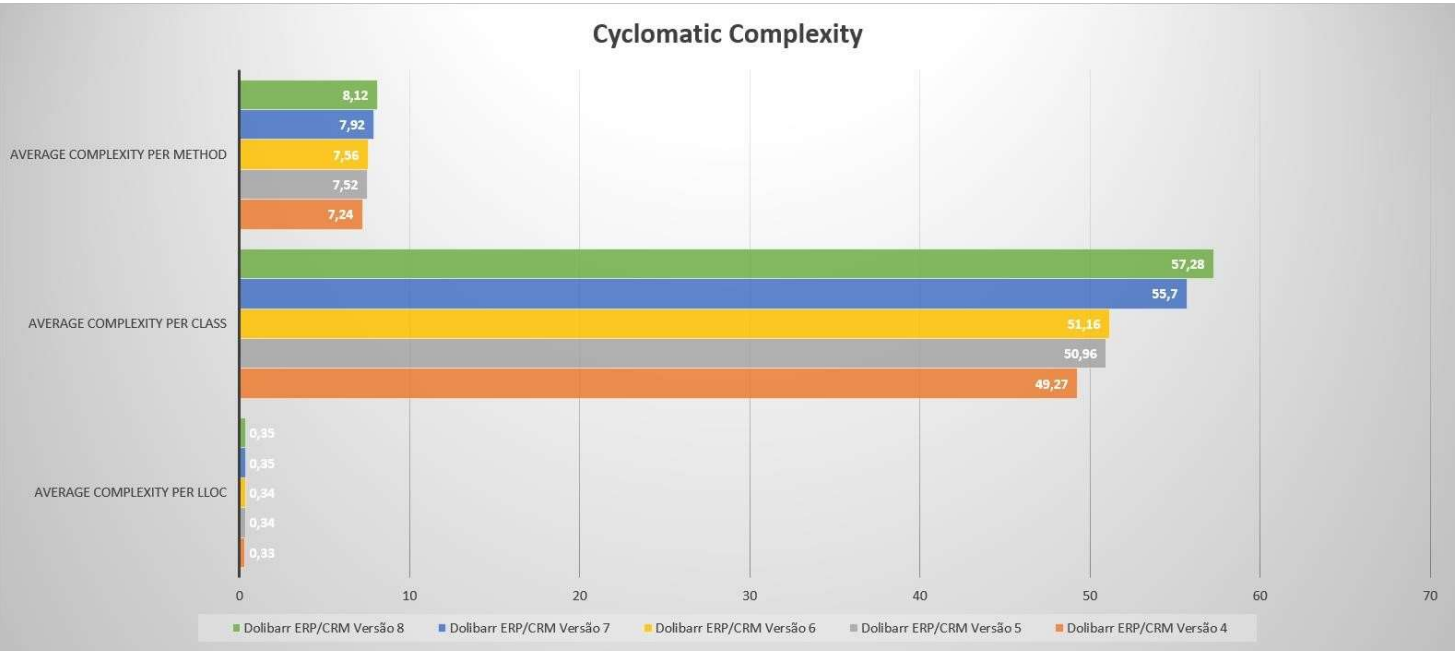
Todos os métodos, atributos e variáveis foram crescentes com as versões do software



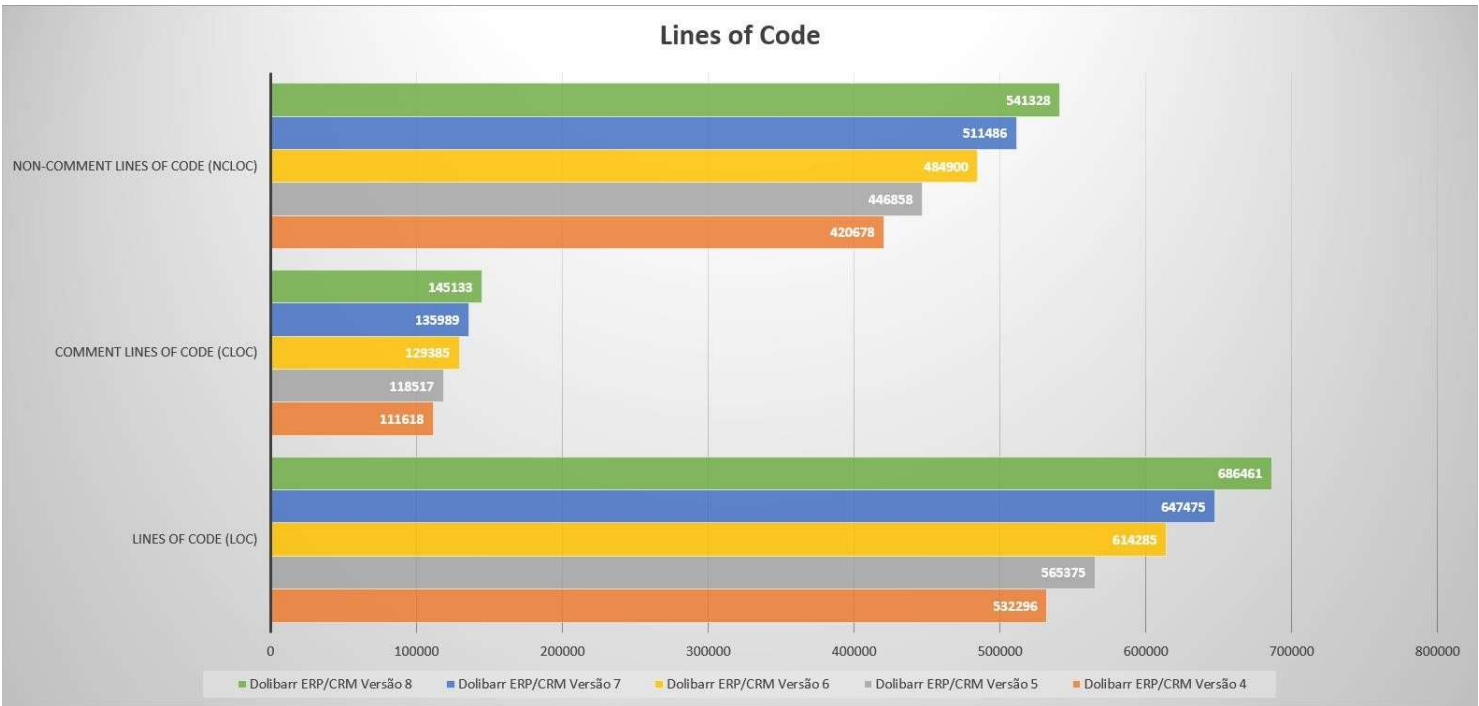
Analisando a estrutura podemos observar que a versão que mais acrescentou métodos, foi a versão 6 do software Dolibarr



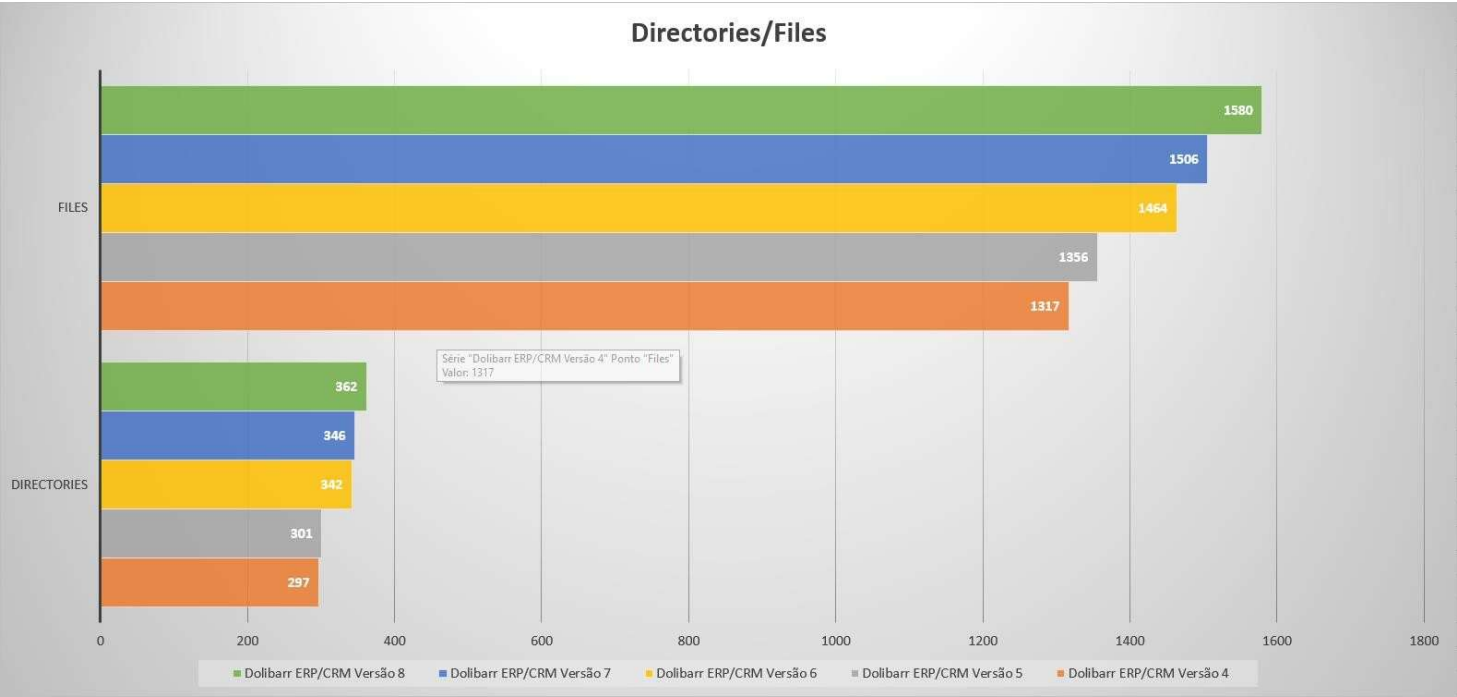
A versão que teve o maior aumento de complexidade ciclométrica foi a versão 7 do software Dolibarr



As linhas de código cresceram com uma média de 4000 linha por versão.



A versão que mais adicionou arquivos foi a versão 6 do software



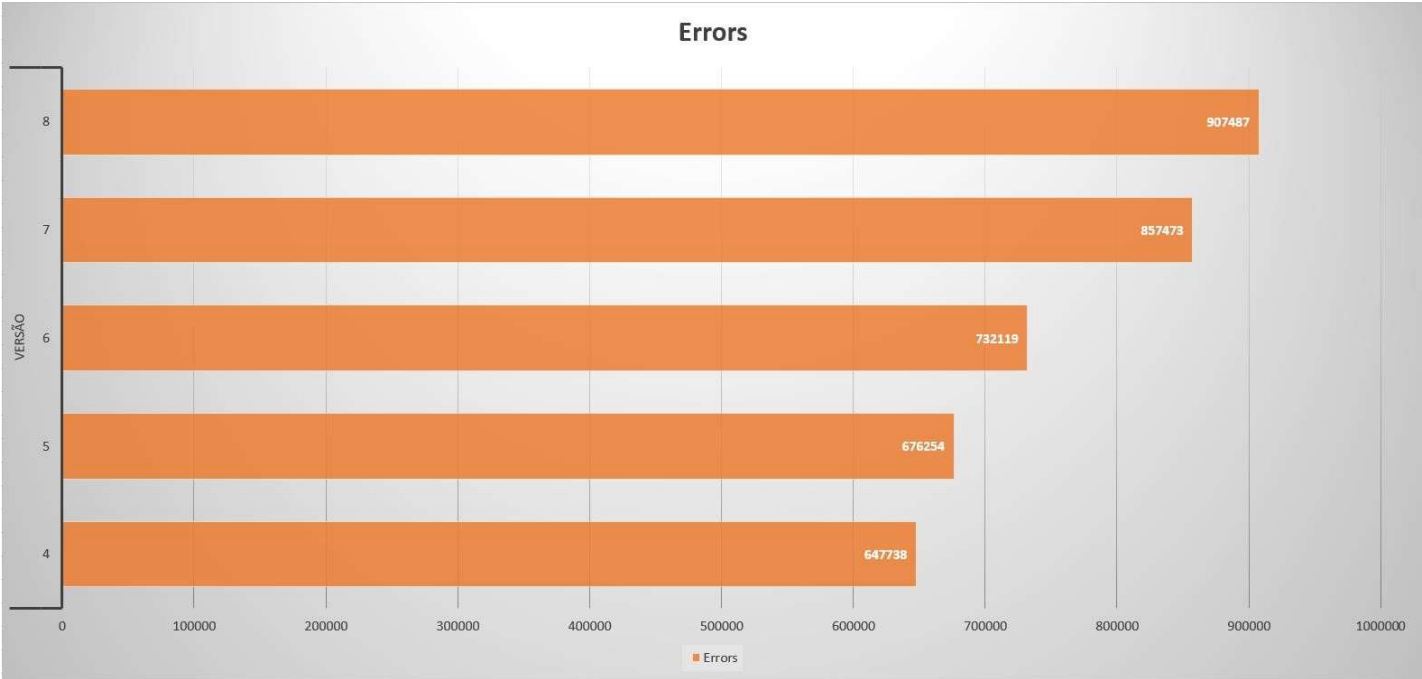
4.3.2 Analise doPHP_CodeSniffer

Abaixo podemos observar que a cada versão do software lançada, a quantidade de erros e avisos foi crescente, mostrando que não houve uma preocupação em relação a quantidade de erros no código da aplicação.

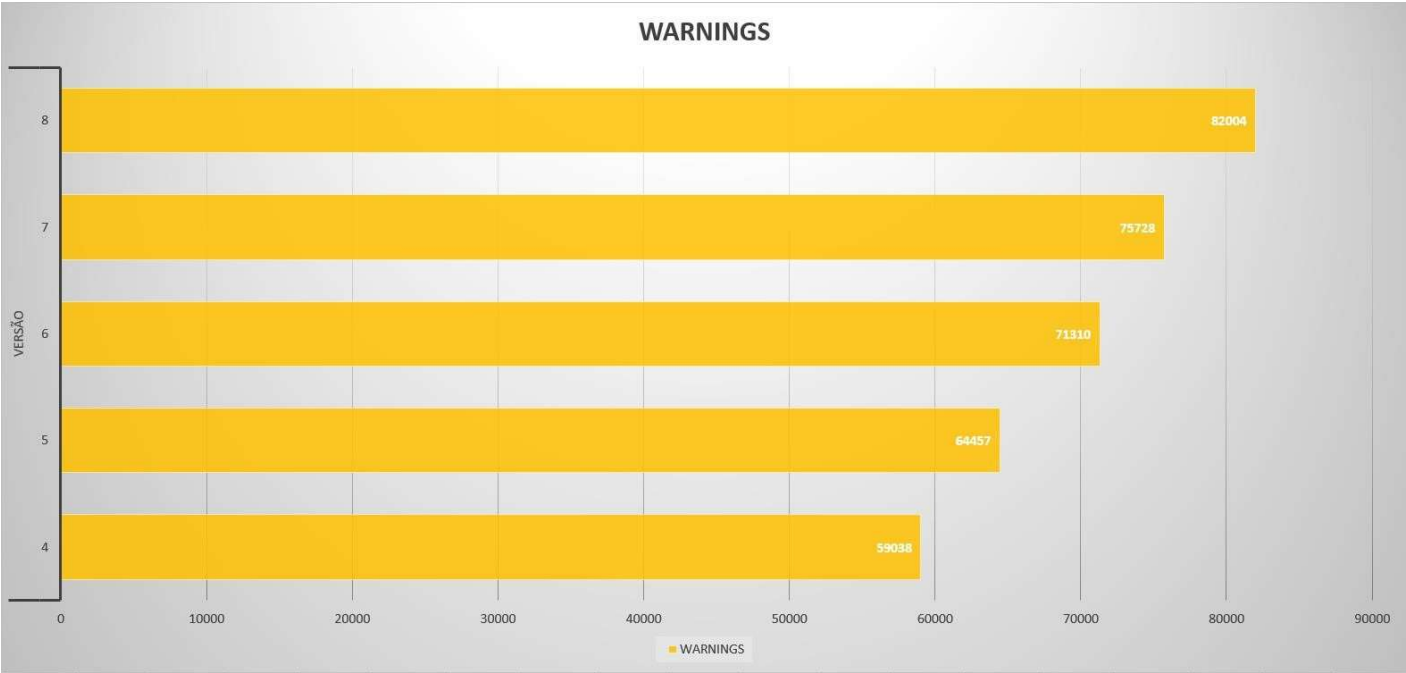
Dados gerados pela ferramenta PHP_CodeSniffer

Dolibarr ERP/CRM					
Medidas	Versão				
	4	5	6	7	8
Files	1338	1377	1493	1534	1607
Errors	647738	676254	732119	857473	907487
WARNINGS	59038	64457	71310	75728	82004

Quantidade de erros a cada versão, sendo a versão 6 que mais gerou erros.



A versão que mais gerou avisos no código, foi a versão 8 do software



4.4 Análise qualitativa

Analisando os ChangeLogs das versões analisadas neste documento pode-se observar que os desenvolvedores do software se preocupam bastante com a inovação do software, adicionando várias funcionalidades ao produto, o que justifica a quantidade de linha adicionadas a cada versão do produto e corrigindo erros que interferem na usabilidade do produto.

Para uma visão completa das mudanças de cada versão do software, acessar o link abaixo:

[ChangeLogs Dolibarr](#)

5 Conclusões

A utilização de análise e métricas de software tem se tornado cada vez mais importante para as empresas, usuários e desenvolvedores. Realizando medições nos produtos de software a qualidade do produto só tende a crescer, acrescentando valor ao produto e satisfação para os usuários.

Com as análises feitas pelas ferramentas PHPLOC E PHP_CodeSniffer pode observar que a cada versão do software, todas as métricas foram crescentes, com destaque para a versão 6 do software, já que foi a versão que teve a maior variação nas métricas.

Pode-se observar que os desenvolvedores do software não tiveram uma preocupação em relação a quantidade de erro e avisos gerados a cada versão, já que esta é uma métrica que tende a diminuir em softwares que tem uma preocupação com a qualidade do código fonte.

Quando se cuida destes tipos de métricas, todos têm a ganhar, os desenvolvedores ganham com a organização e entendimento do software, facilitando a implementação de novas funcionalidades e manutenibilidade do código. A equipe de qualidade consegue testar o software de forma mais eficiente aumentando a qualidade e confiança no software.