Presentation of Results Experiment: Metrics & CodePro AnalytiX

Group 6:
Bruno Cardoso
Felipe César
Tarcísio Filó
William Marcondes

Empirical Software Engineering - 2014/1

Introduction

- Software Measure is important:
  - ‘Evaluate is Control’
  - Plan changes
  - Estimate resources
  - Software and process improvement
  - Plan evolution test
- Validated Metric
  - Tells something useful (correlation,causality, etc.)

How to Measure?

Introduction

- Most famous IDEs do not feature measurement tools! *(Why?)*
- Developers/Stakeholders want a fast and simple way to calculate them
- Eclipse has great & extensive plugin architecture
- Free! Eclipse Public License (~GPL)
- Active Community

Experiment

- Purpose: Evaluate and compare two Eclipse plugins to collect software metrics
  - Usability
  - Information Visualization
  - Export of Results
  - User Experience

Metrics

- Metrics & CodePro AnalytiX
  - http://metrics.sourceforge.net/
  - https://developers.google.com/java-dev-tools/codepro/doc/

Google CodePro AnalytiX™

Scoping

- Problem:
  - How ‘specialists sample’ evaluate two tools to gather software metrics?
  - Are they worth the usage?
  - What are their strengths and weakness?
- Goals:
  - Evaluate and compare two Eclipse plugins for Software Measurement
- Hypothesis:
  - One is better than the other?
  - Plugins are easy to use?

Scoping

- Analyze <Metrics, AnalyticX> from the purpose of <evaluation and comparison of Eclipse measurement plugins> with respect to their <usability, results visualization and export> from the point of view of the <developers> in the context of <software measurement>

Assumptions

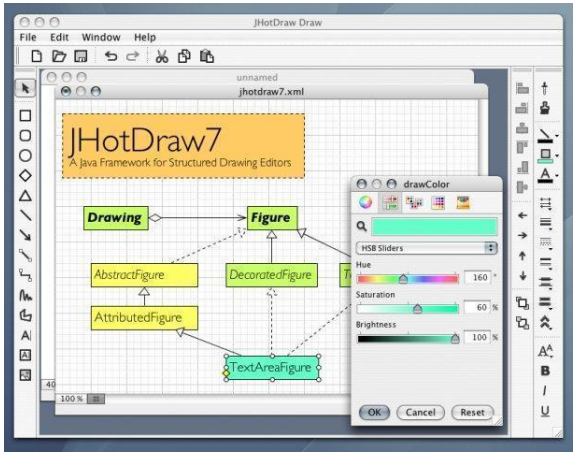
- Learnability (Nielsen(1994))
  - Users were not allowed to use documentation unless in case of failure performing the task
  - *Is it worth the time?*
- All computers desktops had the same specifications
- There were no evident bias during the experiment
  - eg. users logging in e-mail or facebook during it

Design

- Choose plugins
  - Get very familiar with them
  - Explore functionalities
- Prepare forms to collect information
- Prepare Eclipse IDE with plugins
  - Choose target project (JHotDraw)
- Set up computers desktops in the lab

JHotDraw

- JHotDraw is a Java GUI framework for technical and structured Graphics
- It has been developed as a "design exercise" but is already quite powerful
- Its design relies heavily on some well-known design patterns. GPL License
- It is not a simple and very small project.



Execution - step 1

- Run the plugin to extract some metrics
- Time to perform the task
- Difficulty to perform the task
  - Likert Scale (0 to 5)

Results - step 1

Metrics		
Tempo (m)	Utilizou doc ?	Dificuldade
12	1	4
3	0	1
42	1	2
3	0	0

AnalyticX		
2	0	0
3	0	0
2	0	0
4	0	1
9	0	1

Média	Documentação?	Dificuldade
15	40%	1,75

Média	Documentação?	Dificuldade
4	0	0,4

Execution - step 2

- Fill a table with the values extracted for the requested software artifacts
- Results Visualization
- Time to perform the task
- Difficult to perform the task
  - Likert scale (1 to 5)

Results - step 2

Metrics		
Tempo (m)	Utilizou doc ?	Dificuldade
23	1	3
17	0	1
14	0	5
12	0	2
11	0	1

AnalyticX		
Tempo (m)	Utilizou doc ?	Dificuldade
9	0	0
8	0	0
4	0	0
7	0	2
4	0	0

Média	Documentação?	Dificuldade
15,4	20%	2,4

Média	Documentação?	Dificuldade
6,4	0	0,4

Execution - step 3

- Export the results to a XML file
  - Acceptable way to share and compare results in production environment
- Time to perform the task
  - Is there an acceptable usage threshold?
- Difficulty to perform the task
  - Will developers use it again if it was too hard?

Results - step 3

Metrics		
Tempo (m)	Utilizou doc ?	Dificuldade
5	1	1
1	0	1
1	0	0
2	0	0
2	0	0

AnalyticX		
Tempo (m)	Utilizou doc ?	Dificuldade
10	1	2
4	0	1
4	0	1
2	0	2
10	1	3

Média	Documentação?	Dificuldade
2,2	20%	0,4

Média	Documentação?	Dificuldade
6	40%	1,8

Execution - step 4

- Overall User Experience in the use of the plugin
- General difficulty
  - Likert Scale
- Would recommend to other users?
  - Yes / No
- Relevance of the collected metrics
  - Yes / Yes, but miss some (Which?) / No
  - What did you like best/least ?

Results - step 4

Metrics		
Dificuldade	Recomendação	Relevância
2,4	80%	40%

AnalyticX		
Dificuldade	Recomendação	Relevância
1	100%	40%

Métricas Sugeridas
DIT
NOC
LCOM
Tamanho dos comentários

Métricas Sugeridas
LOC
DIT
FAN-IN
FAN-OUT

+Calcula automaticamente a cada build
-Usabilidade ruim

+Facilidade de uso
-Exportação ruim

*Dificuldade: de 0 a 5

Execution problems & Incidents

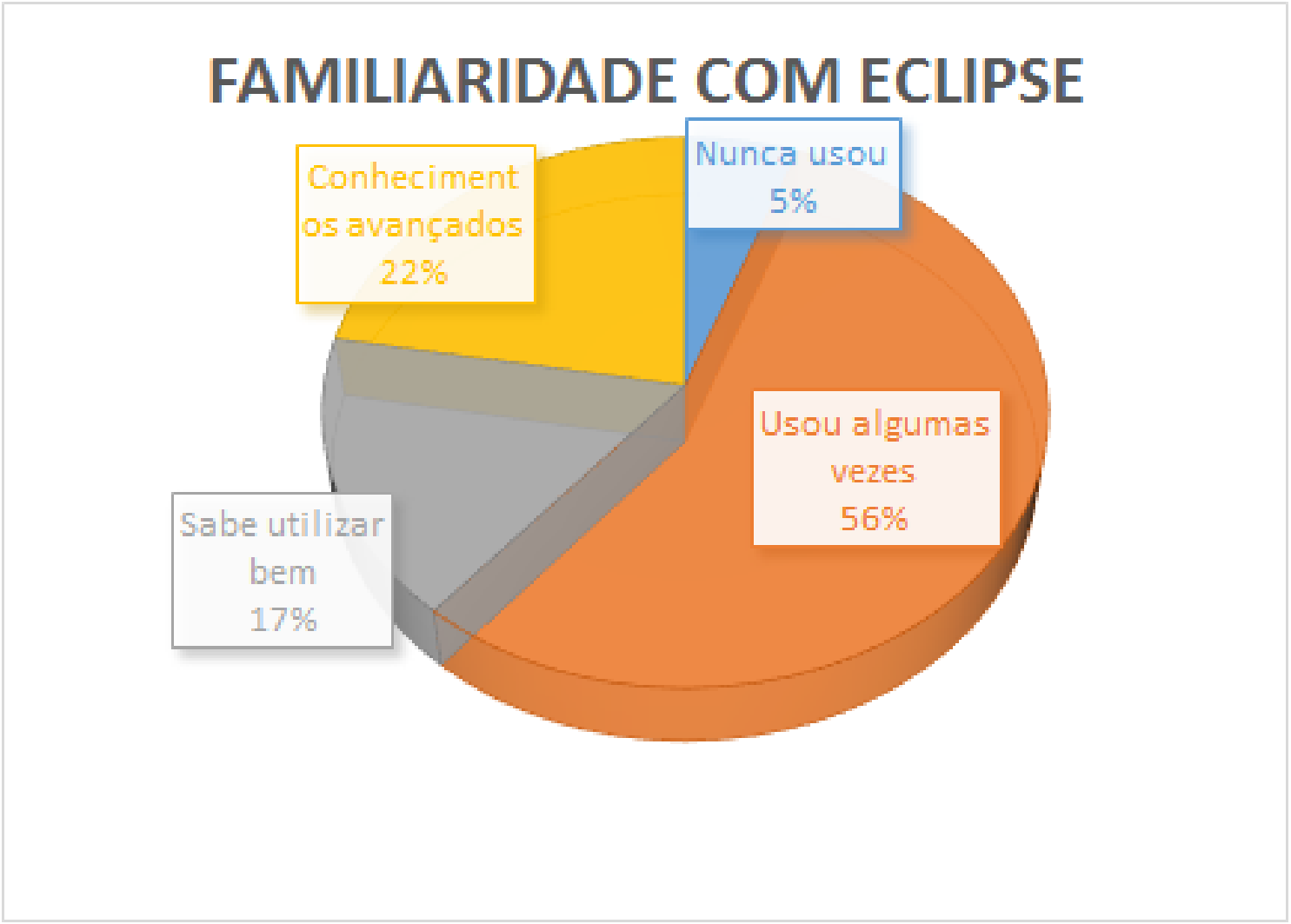
- Both plugins were already installed in Eclipse, but one pair got confused, despite instructions

- One machine could not compute metrics (stuck) during the experiment
  - Metrics plugin
  - Was it a bug or a machine problem?
- One of the artifacts was ordered with the wrong name in step 2

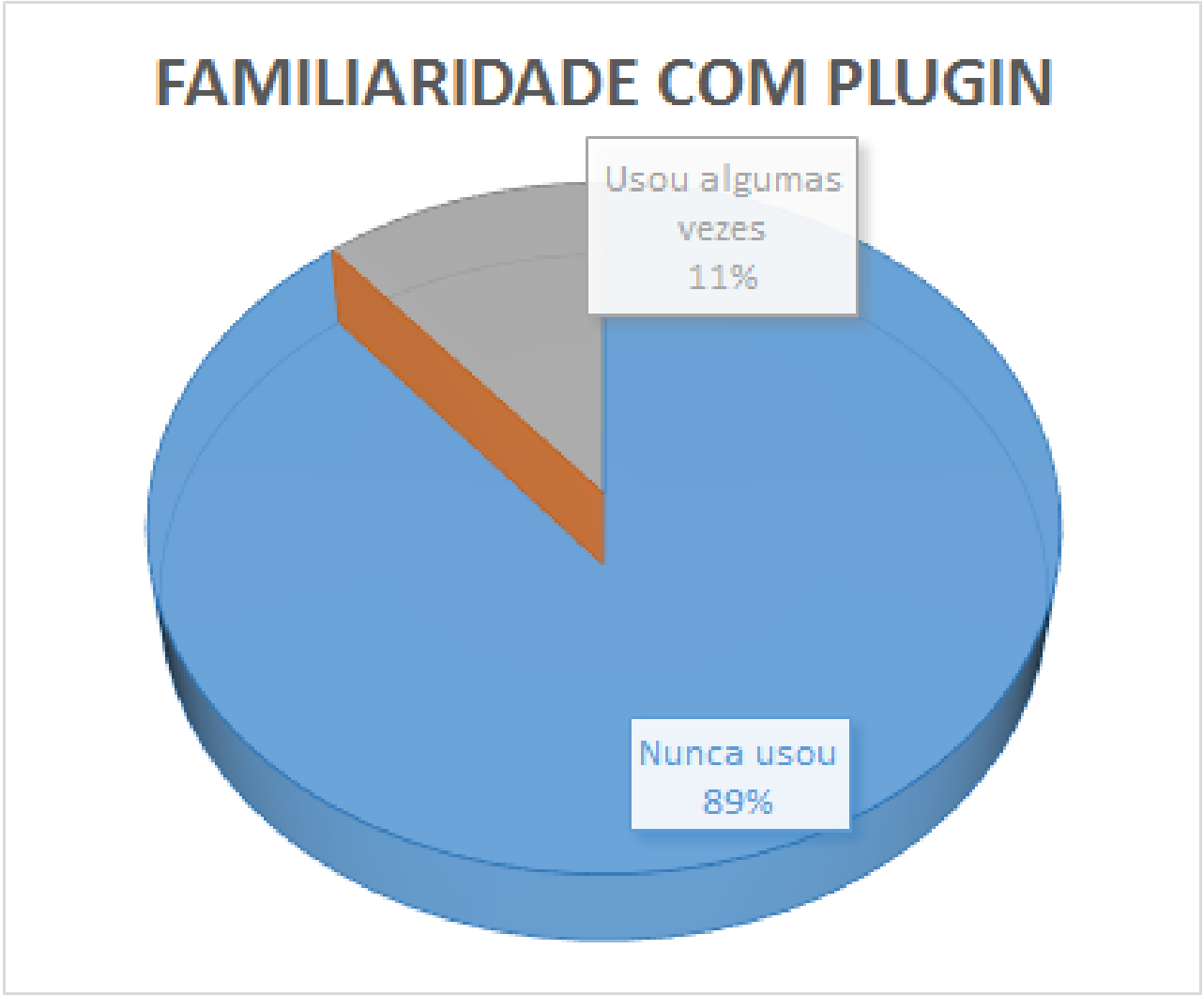
Prior Knowledge

- Eclipse IDE
  - Never used
  - Sometimes
  - Well
  - Advanced
- Metrics / CodePro AnalytiX
  - Used / Not Used
  - Used
    - Once
    - Sometimes
    - Well

Prior Knowledge



Prior Knowledge



Comments From Participants

- Lack of concern metrics was mentioned in the two plugins
- Metrics does not provide search tools and efficient navigation
- AnalytiX presents great ease of use and practicality
- AnalytiX has difficulty in exporting the results

Distribution of Knowledge

Metrics				
0	2	2	4	6

AnalytiX				
2	2	2	*4*	7

0 to 3 - Eclipse
0 to 2 - Plugin

Sum of the knowledge of each person in the group

Comments From Participants

- Lack of concern metrics was mentioned in the two plugins
- Metrics does not provide search tools and efficient navigation
- AnalytiX presents great ease of use and practicality
- AnalytiX has difficulty in exporting the results

Documentation Support

- Step 1
  - Metrics: two people used
  - AnalytiX: none
- Step 2
  - Metrics: one person used
  - AnalytiX: none
- Step 3
  - Metrics: one person used
  - AnalytiX: two people used

Conclusions

- Results suggest that AnalytiX has improved usability compared to Metrics
- They also suggest that AnalytiX provides a better results visualization
- However, Metrics was easier to use, providing more insight to export the results

Conclusions

- Both plugins should improve navigation on the results with artifacts filters
- In general, AnalytiX showed to be a better and more accepted tool
- However, both plugins showed ease of use. All pairs completed the task in a relatively short time, even with limited knowledge in Eclipse

Limitations

- What is a representative sample in Software Engineering?
- Does using students bias the results?
- Statistical Tests
  - Hard to perform Inferential Statistics
  - Only Descriptive Statistics
- Could have used mouse click counter, for example

References

- Course Website
  - www.dcc.ufmg.br/~figueiredo
- Notes from C. Wohlin et al. Experimentation in Software Engineering, Springer. 2012
- Jeff Offutt - George Mason University
  - http://www.cs.gmu.edu/~offutt/
- Jakob Nielsen Website
  - http://www.nngroup.com/people/jakob-nielsen/