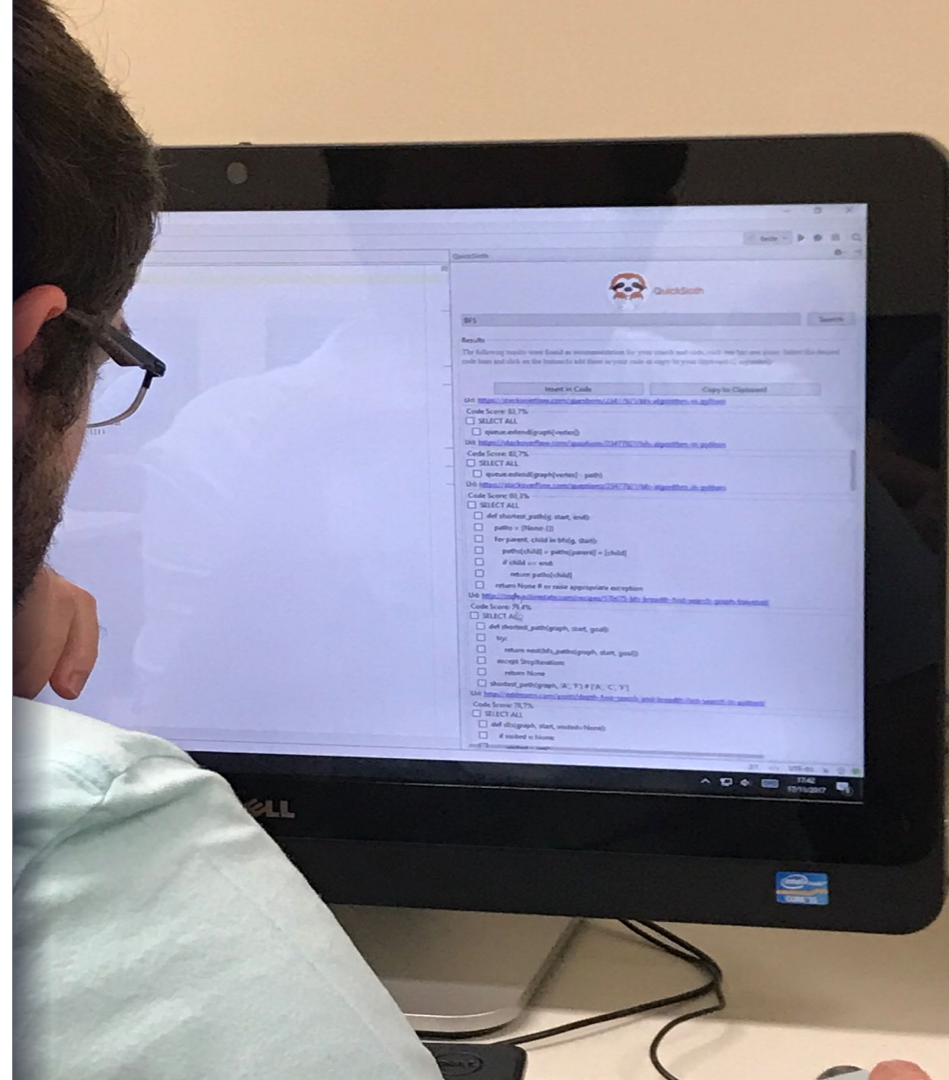


QuickSloth

Sistema modular de Recomendação de código-fonte

Ahmad Khatib e Pamela Peixinho

Orientador: Prof. Dr. Guilherme Wachs



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Sistemas de Recomendação

- Informações sobre o usuário
- Alta taxa de acerto
- Alta demanda em diversas áreas:
 - Filmes
 - Propaganda
 - Código-fonte (SCoReS)

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Internet é usada por desenvolvedores em

Média 12 vezes por dia

para buscar códigos



Mais de 19.4M

repositórios de código no Github em 2016



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Modelar e Implementar um sistema de recomendação de códigos-fonte que pode levar em consideração **métricas** provenientes da área de engenharia de software, assim como processamento de Linguagem Natural. Esse modelo deverá ser estruturado de tal forma que possibilite a **configuração e inclusão de novos módulos** capazes de avaliar códigos-fonte. Além disso, implementar uma interface para os usuários finais que possibilite a redução de tempo de desenvolvimento.

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

- Processamento de Linguagem Natural (NLP);
- Métricas de Engenharia de Software;
- Abstract Syntax Tree (AST);

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Processamento de Linguagem Natural

- Área multidisciplinar: estatística; linguística; ciência da computação; inteligência artificial;
- Usuários se comunicando com as máquinas através de linguagem natural;

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Processamento de Linguagem Natural

- Contexto;
- Modelagem contextual de palavra;
- *Gap Semântico*;

Introdução

Motivação

Objetivo

Conceitos

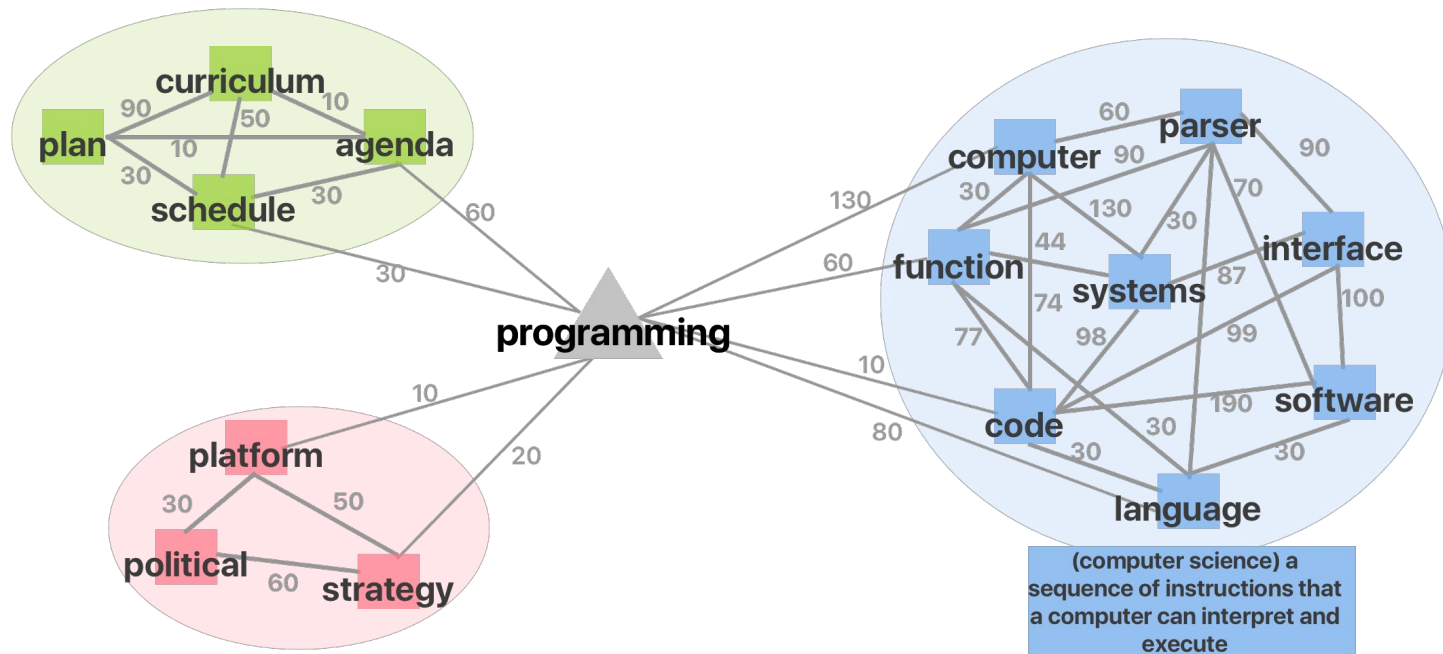
Metodologia

Resultados

Conclusão

Aplicação Final

Processamento de Linguagem Natural



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Métricas de Engenharia de Software

Introdução

Motivação

Objetivo

Conceitos

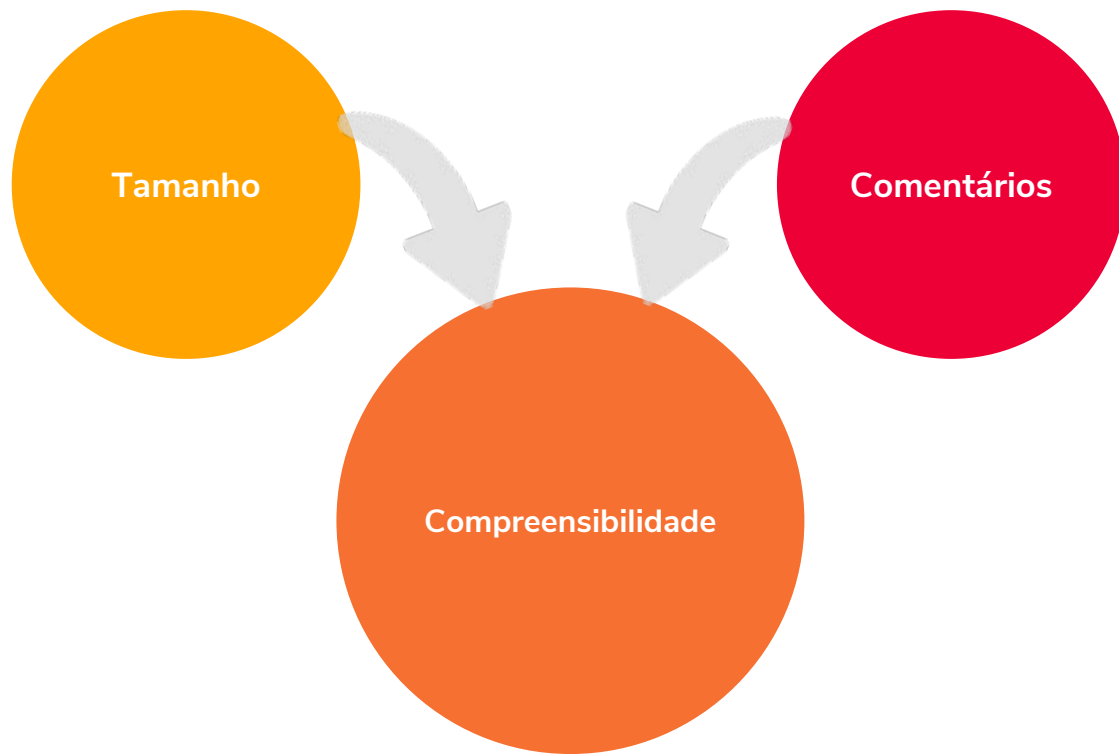
Metodologia

Resultados

Conclusão

Aplicação Final

Compreensibilidade de Código-fonte



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Compreensibilidade de Código-fonte

Muitas linhas de código e nenhuma linha de comentário

```
11 def f(n):
12     a = 0
13     b = 1
14     for i in range(0, n):
15         print(a)
16         temp = a
17         a = b
18         b = temp + b
19     return a
```

Poucas linhas de código e muitas linhas de comentários

```
2 def fibonacci(n):
3     #1st number in fibonacci sequence
4     if n == 0: return 0
5     #2nd number in fibonacci sequence
6     elif n == 1: return 1
7     #recursive fibonacci call to calculate the n-th value based on the 2 previous values
8     else: return fibonacci(n-1)+fibonacci(n-2)
9
```

Introdução

Motivação

Objetivo

Conceitos

Metodologia

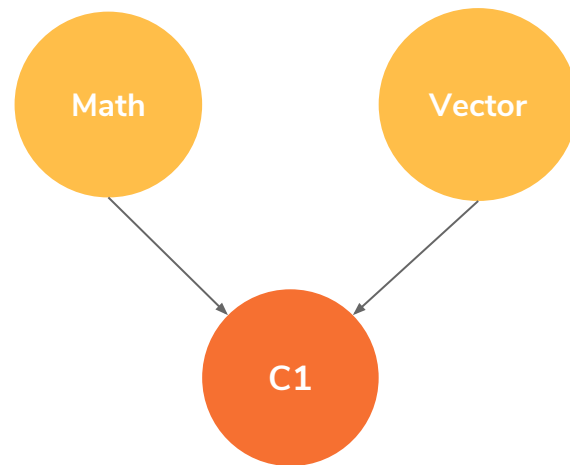
Resultados

Conclusão

Aplicação Final

Acoplamento

```
C1.java x
1  import java.util.Vector;
2  import java.lang.Math;
3
```



Introdução

Motivação

Objetivo

Conceitos

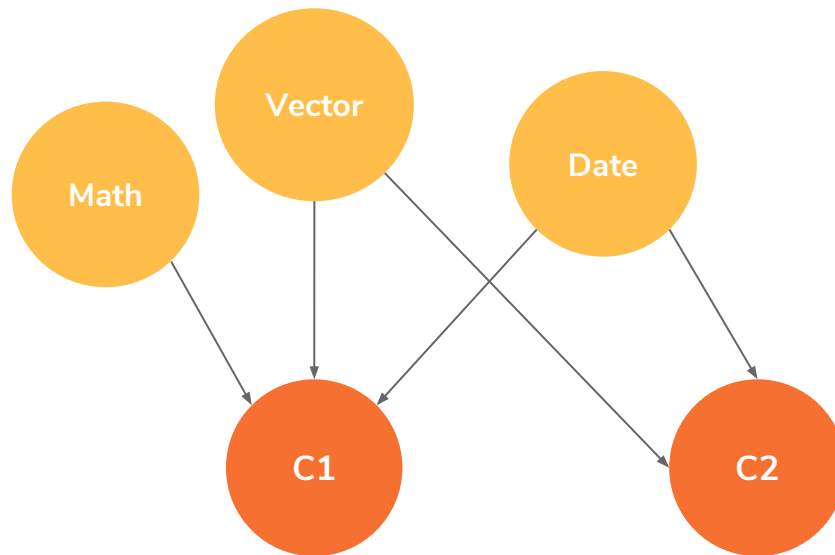
Metodologia

Resultados

Conclusão

Aplicação Final

Acoplamento



Fácil integração: C2 utiliza mesma dependência que C1

Introdução

Motivação

Objetivo

Conceitos

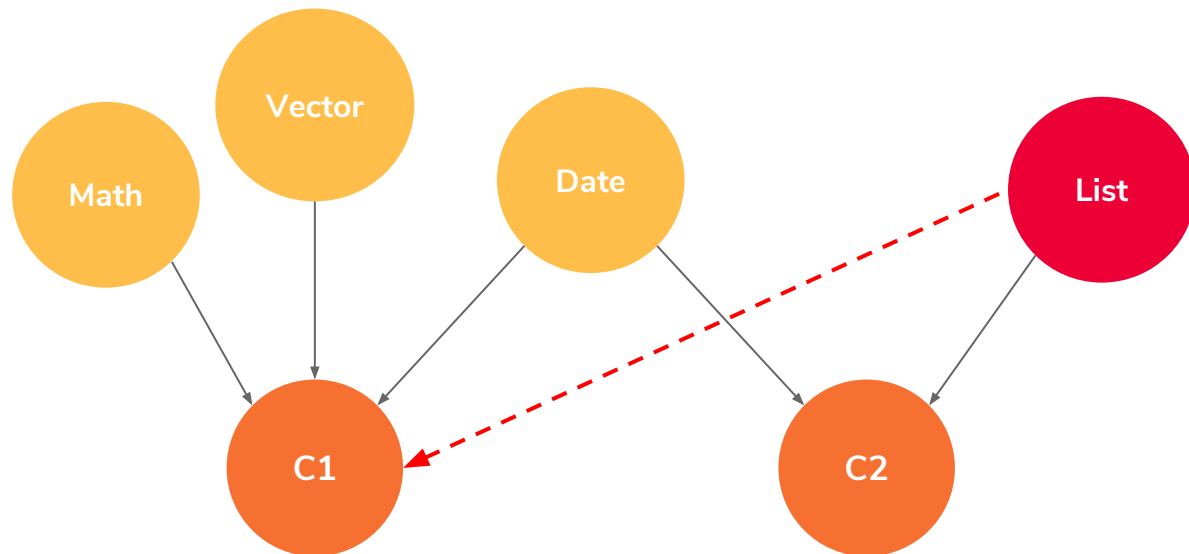
Metodologia

Resultados

Conclusão

Aplicação Final

Acoplamento



Difícil integração (custoso): C2 possui uma dependência diferente de C1

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

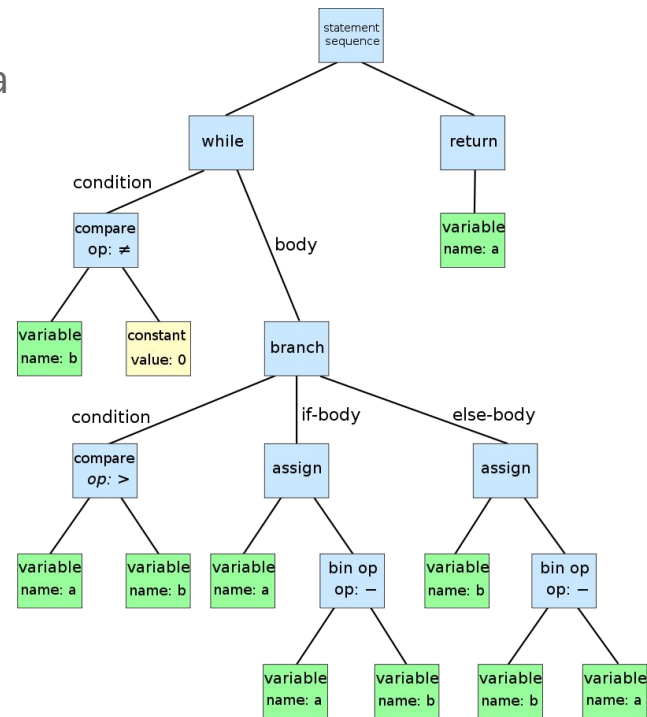
Conclusão

Aplicação Final

Abstract Syntax Tree

- Representação em árvore da estrutura sintática de um arquivo de código-fonte

```
while b ≠ 0
  if a > b
    a := a - b
  else
    b := b - a
return a
```



Introdução

Motivação

Objetivo

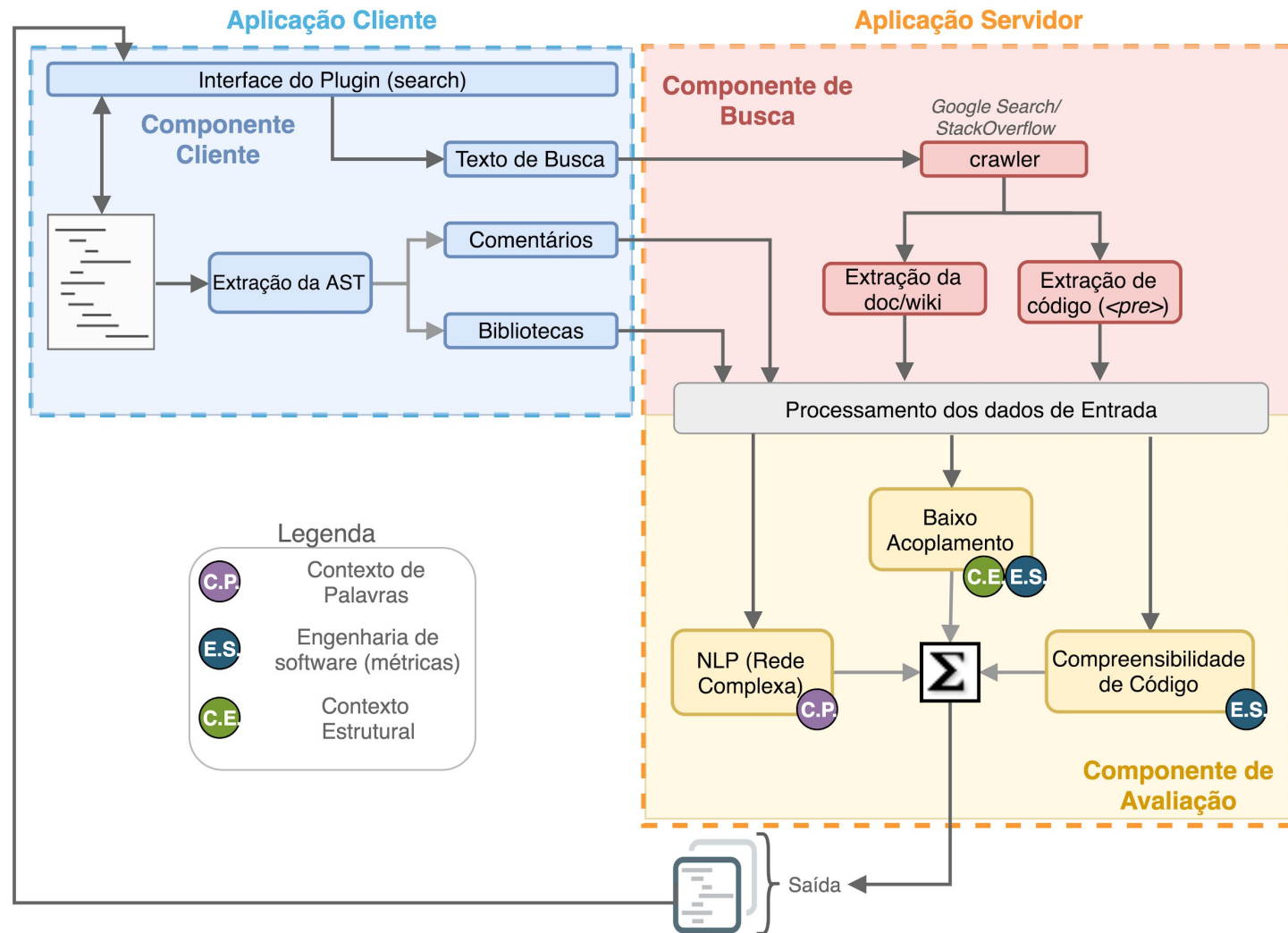
Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final



Introdução

Motivação

Objetivo

Conceitos

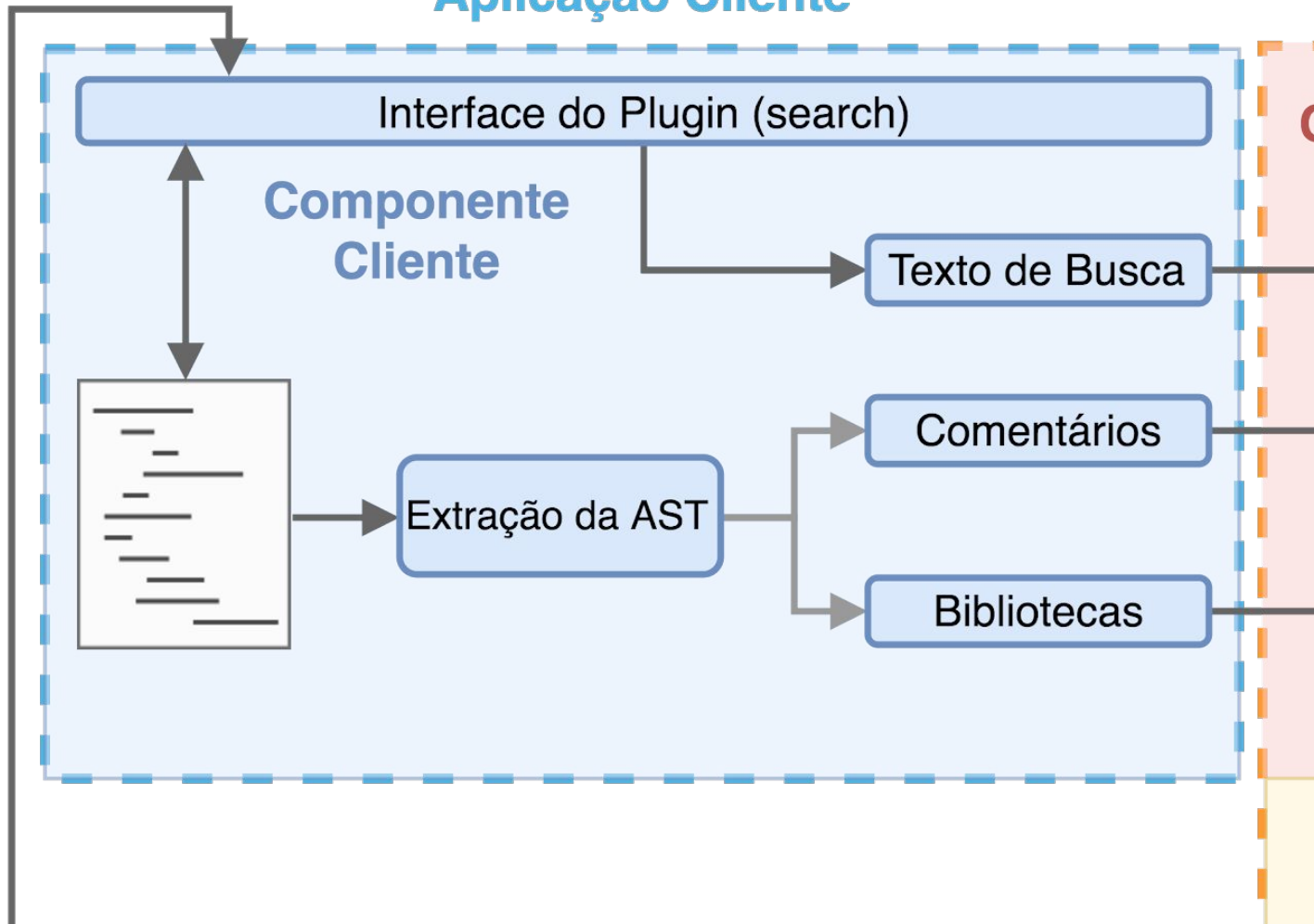
Metodologia

Resultados

Conclusão

Aplicação Final

Aplicação Cliente



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Aplicação Cliente

Interface do Plugin (search)

Fonte

Exatidão da AST

Texto de Busca

Comentários

Bibliotecas

Legenda

Contexto de
Palavras

Engenharia de
software (métricas)

Contexto
Estrutural

Aplicação Servidor

Componente de Busca

Google Search/
StackOverflow

crawler

Extração da
doc/wiki

Extração de
código (<pre>)

Processamento dos dados de Entrada

Baixo
Acoplamento

C.E.E.S.

NLP (Rede
Complexa)

C.P.

Compreensibilidade
de Código

E.S.

Componente de Avaliação

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Aplicação Servidor

Componente de Busca

*Google Search/
StackOverflow*

crawler

Extração da
doc/wiki

Extração de
código (<pre>)

Processamento dos dados de Entrada

Introdução

Motivação

Objetivo

Conceitos

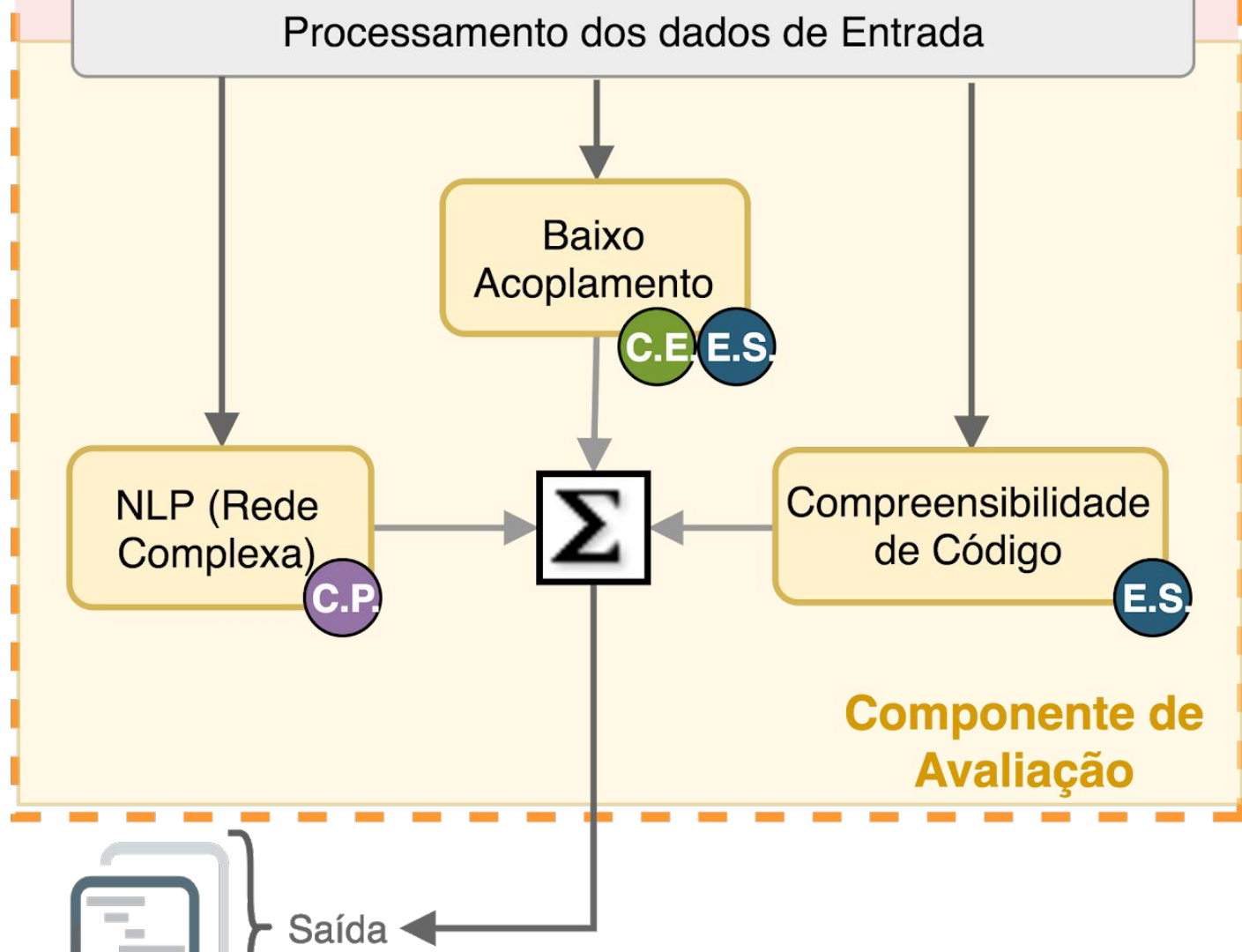
Metodologia

Resultados

Conclusão

Aplicação Final

Processamento dos dados de Entrada



Introdução

Motivação

Objetivo

Conceitos

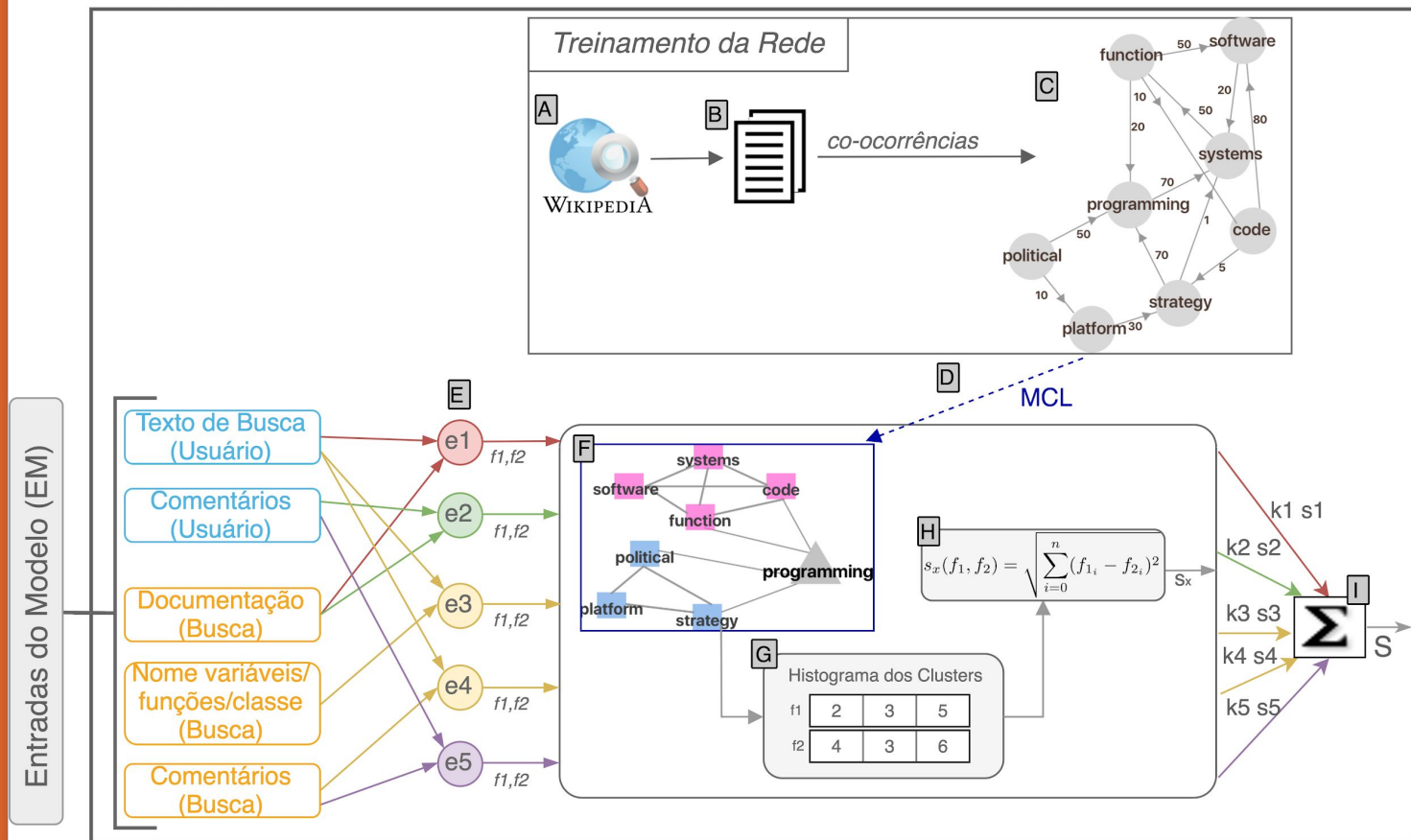
Metodologia

Resultados

Conclusão

Aplicação Final

Módulo NLP



Introdução

Motivação

Objetivo

Conceitos

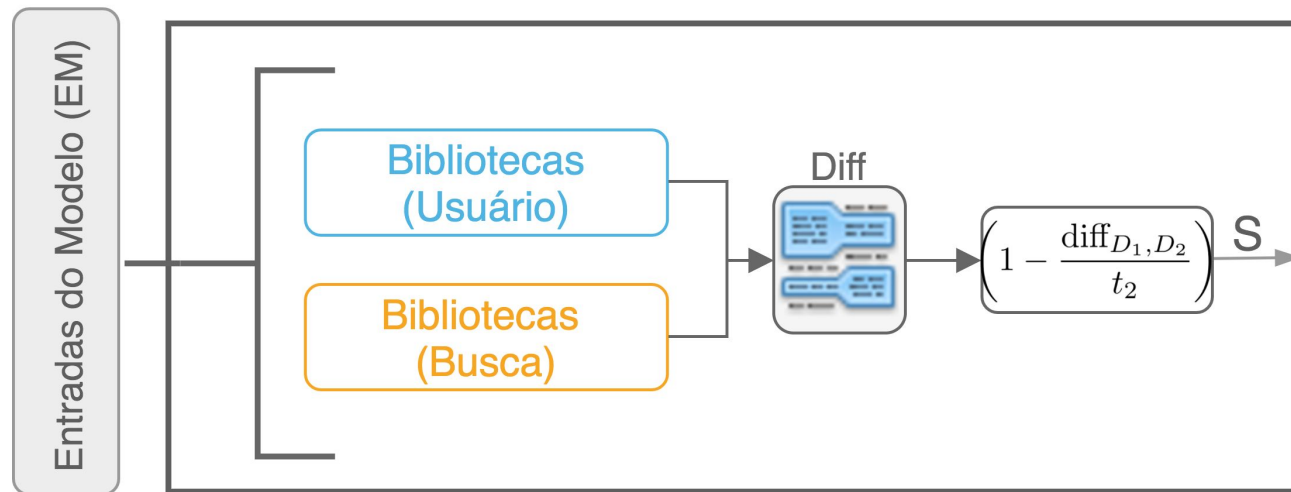
Metodologia

Resultados

Conclusão

Aplicação Final

Módulo Baixo Acoplamento



Introdução

Motivação

Objetivo

Conceitos

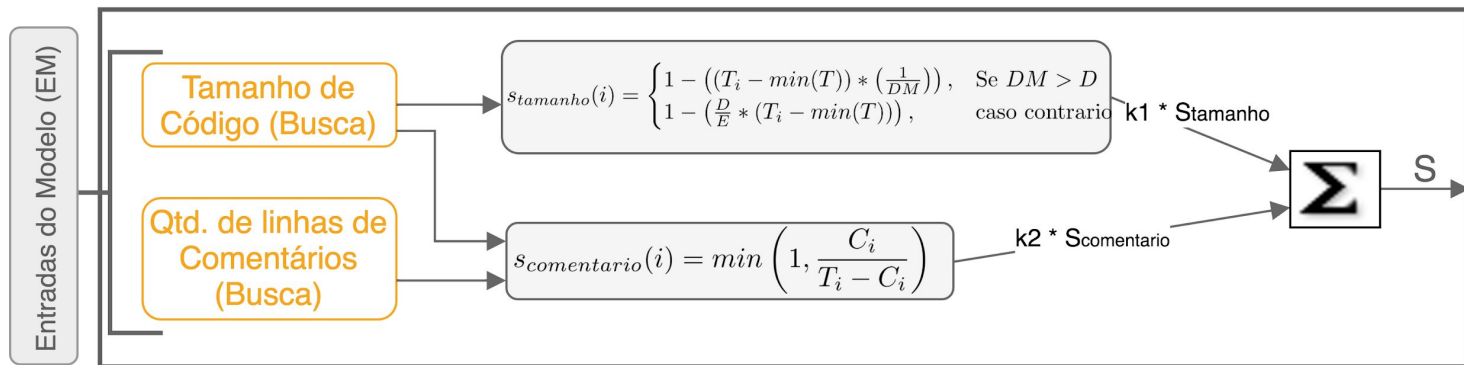
Metodologia

Resultados

Conclusão

Aplicação Final

Módulo Compreensibilidade de Código



Introdução

Motivação

Objetivo

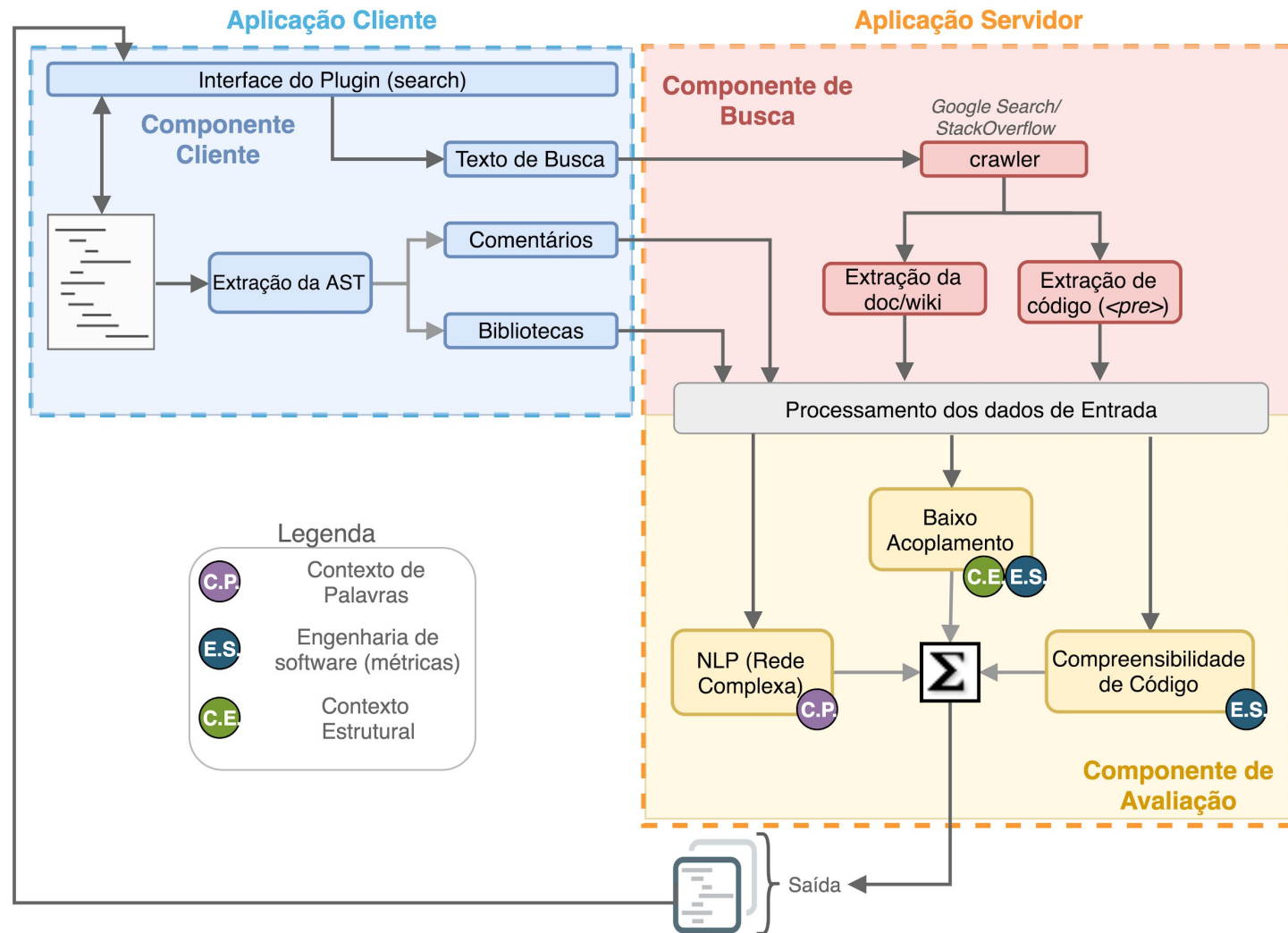
Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

- Resultados das Recomendações
 - Comparação com Google
- Teste com Usuários

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Resultados das Recomendações

- Comparação com Google ("read file")

read file

Search

Results

The following results were found as recommendation for your search and code, each one has one score. Select the desired code lines and click on the button to add these in your code or copy to your clipboard (5 segundos)

Insert in Code Copy to Clipboard

Code Score: 100%

- ☐ SELECT ALL
- ☐ # Open the file for reading.
- ☐ with open('my_file.txt', 'r') as infile:
- ☐ data = infile.read() # Read the contents of the file into memory.

Url: <https://stackoverflow.com/questions/3277503/how-do-i-read-a-file-line-by-line-into-a-list>

Code Score: 99.7%

- ☐ SELECT ALL
- ☐ # Return a list of the lines, breaking at line boundaries.
- ☐ my_list = data.splitlines()

Url: <https://stackoverflow.com/questions/3277503/how-do-i-read-a-file-line-by-line-into-a-list>

Code Score: 99.6%

- ☐ SELECT ALL
- ☐ fp = open('file.txt') # open file on read mode
- ☐ lines = fp.read().split("\n") # create a list containing all lines
- ☐ fp.close() # close file

Url: <https://stackoverflow.com/questions/3277503/how-do-i-read-a-file-line-by-line-into-a-list>

Code Score: 98.3%

- ☐ SELECT ALL
- ☐ # Open the file for reading.
- ☐ with open('my_file.txt', 'r') as infile:
- ☐ data = infile.read() # Read the contents of the file into memory.

9º resposta no StackOverflow

//stackoverflow.com/questions/3277503/h

Questions Developer Jobs Tags Us

27

Clean and Pythonic Way of Reading the

First and foremost, you should focus on op and pythonic way. Here is an example of th

```
infile = open('my_file.txt', 'r')
data = infile.read() # Read the co
infile.close() # Close the file si
```

Instead, I prefer the below method of open and does not require an extra step of clos below, we're opening the file for reading, a within this statement has finished running,

```
# Open the file for reading.
with open('my_file.txt', 'r') as in
    data = infile.read() # Read th
```

Now we need to focus on bringing this dat and flexible. In your case, the desired goal element. To accomplish this, we will use th

```
# Return a list of the lines, break
my_list = data.splitlines()
```

The Final Product:

```
# Open the file for reading.
with open('my_file.txt', 'r') as in
    data = infile.read() # Read th
# Return a list of the lines, break
my_list = data.splitlines()
```

Introdução

Motivação

Objetivo

Conceitos

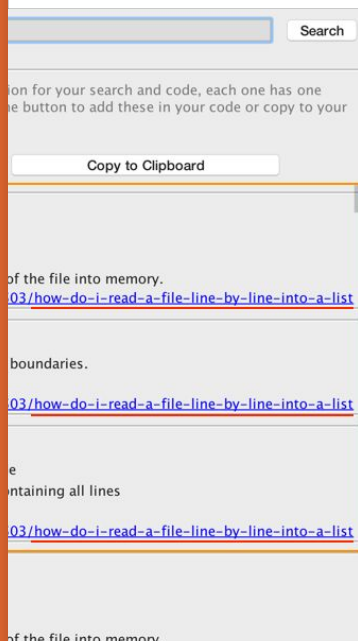
Metodologia

Resultados

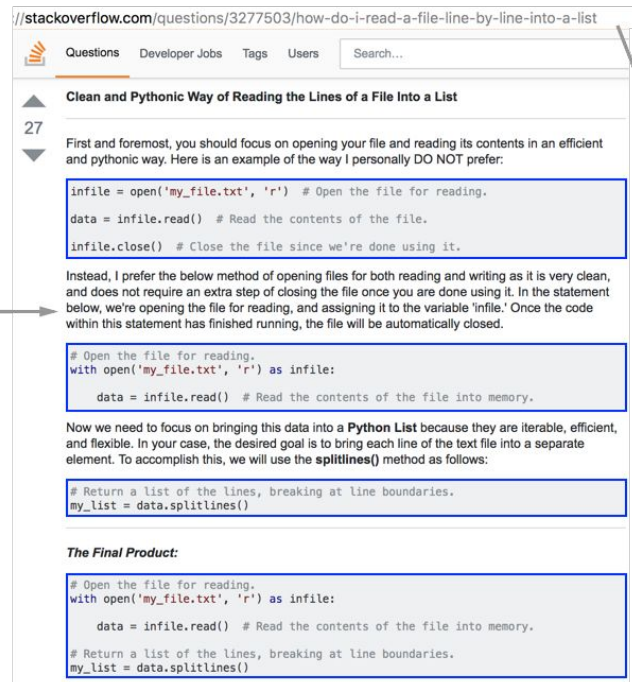
Conclusão

Aplicação Final

Resultados das Recomendações



9º resposta no StackOverflow



8º link do google



Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Resultados das Recomendações

//stackoverflow.com/questions/3277503/how-do-i-read-a-file-line-by-line-into-a-list

Questions Developer Jobs Tags Users Search...

Clean and Pythonic Way of Reading the Lines of a File into a List

27

First and foremost, you should focus on opening your file and reading its contents in an efficient and pythonic way. Here is an example of the way I personally DO NOT prefer:

```
infile = open('my_file.txt', 'r') # Open the file for reading.
data = infile.read() # Read the contents of the file.
infile.close() # Close the file since we're done using it.
```

Instead, I prefer the below method of opening files for both reading and writing as it is very clean, and does not require an extra step of closing the file once you are done using it. In the statement below, we're opening the file for reading, and assigning it to the variable 'infile.' Once the code within this statement has finished running, the file will be automatically closed.

```
# Open the file for reading.
with open('my_file.txt', 'r') as infile:
    data = infile.read() # Read the contents of the file into memory.
```

Now we need to focus on bringing this data into a **Python List** because they are iterable, efficient, and flexible. In your case, the desired goal is to bring each line of the text file into a separate element. To accomplish this, we will use the **splitlines()** method as follows:

```
# Return a list of the lines, breaking at line boundaries.
my_list = data.splitlines()
```

The Final Product:

```
# Open the file for reading.
with open('my_file.txt', 'r') as infile:
    data = infile.read() # Read the contents of the file into memory.

# Return a list of the lines, breaking at line boundaries.
my_list = data.splitlines()
```

8º link do google

read file python

Todas Vídeos Imagens Maps Notícias Mais Configurações Ferramentas

Aproximadamente 57.200.000 resultados (0,32 segundos)

Reading and Writing Files in Python - Pythonforbeginners.com
www.pythonforbeginners.com/files/reading-and-writing-files-in-... Traduzir esta página
8 de jul de 2013 - Overview When you're working with Python, you don't need to import a library in order to read and write files. ...

7. Input and Output — Python 2.7.14 documentation
<https://docs.python.org/2/tutorial/inputoutput.html> Traduzir esta página
(A third way is using the write() method of file objects; the standard output file To read a file's contents, call f.read(size), which reads some quantity of data ...

7. Input and Output — Python 3.6.3 documentation
<https://docs.python.org/3/tutorial/inputoutput.html> Traduzir esta página
(A third way is using the write() method of file objects; the standard output file 'b' appended to the mode opens the file in binary mode; now the data is read ...

Python Reading Files Tutorial - After Hours Programming
www.afterhoursprogramming.com/tutorial/Python/Reading-Files/ Traduzir esta página
The Python Reading Files tutorial explains how to read files with Python.

Exercise 15: Reading Files - Learn Python the Hard Way
<https://learnpythonthehardway.org/book/ex15.html> Traduzir esta página
Now you will learn about reading from a file. You may have to play with ... your checks. Working with files is an easy way to erase your work if you are not careful.

Exercise 16: Reading And Writing Files - Learn Python the Hard Way
<https://learnpythonthehardway.org/book/ex16.html> Traduzir esta página
Exercise 16: Reading and Writing Files. If you did the Study Drills from the last exercise you should have seen all sorts of commands (methods/functions) you ...

How To Read and Write Files in Python 3 | DigitalOcean
<https://www.digitalocean.com/.../how-to-handle-plain-text-files-in-...> Traduzir esta página
13 de out de 2016 - This tutorial will briefly describe some of the format types Python is able to handle. After a brief introduction to file formats, we'll go through how ...

string - How do I read a file line-by-line into a list? - Stack Overflow
<https://stackoverflow.com/.../how-do-i-read-a-file-line-by-line-into-...> Traduzir esta página
18 de jul de 2010 - with open('file.txt') as fp: lines = fp.read().split("\n") ... Now we need to focus on bringing this data into a Python List because they are iterable, ...

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Teste com Usuários

- Objetivo:
 - Validar efetividade
 - Validar aceitação
- Participantes:
 - Alunos do Curso de Computação
- 3 tarefas
- Questionário pós-teste

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Teste com Usuários

- Resultados Efetivos:

	QuickSloth	Google
Tarefa 1	80%	75%
Tarefa 2	100%	50%

Introdução

Motivação

Objetivo

Conceitos

Metodologia

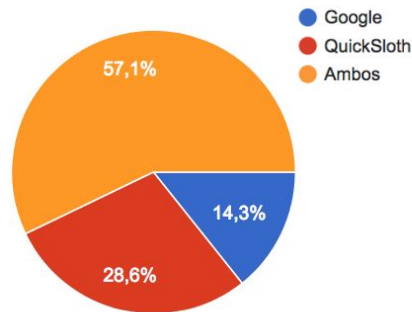
Resultados

Conclusão

Aplicação Final

Teste com Usuários

- Resultados de Aceitação
 - Escolha da Ferramenta na Tarefa 3



QuickSloth	Google
85,7%	71,4%

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Teste com Usuários

- Conclusão dos Resultados:
 - Recomendações atenderam as necessidades dos usuários;
 - A maioria dos usuários consideraram o sistema fácil de usar;
 - Usuários tiveram a impressão de que a aplicação economizou seu tempo de desenvolvimento;
 - Desenvolvedores inexperientes tiveram dificuldades em utilizar o QuickSloth.

Conclusão

- **Resultados Esperados:**

Evitar que o usuário saia do ambiente de desenvolvimento para encontrar o códigos-fonte;

Facilidade de incorporar códigos sugeridos pelo sistema;

Redução de tempo na busca por códigos-fonte;

Diminuição do gap semântico através do uso das Redes Complexas;

Sistema fácil de configurar e incluir conceitos (módulos).

Conclusão

- ✓ Evitar que o usuário saia do ambiente de desenvolvimento para encontrar o códigos-fonte;
- ✓ Facilidade de incorporar códigos sugeridos pelo sistema;
- ? *Redução de tempo na busca por códigos-fonte;*
- ✓ Diminuição do gap semântico através do uso das Redes Complexas;
- ✓ Sistema fácil de configurar e incluir conceitos (módulos);
- ✓ * Publicação do QuickSloth no Repositório Oficial da JetBrains.

Introdução

Motivação

Objetivo

Conceitos

Metodologia

Resultados

Conclusão

Aplicação Final

Aplicação



QuickSloth

Obrigado