

SOPAS - Submissão Online Para Análise de Software

José Pedro Silva Pedro Faria Ulisses Costa

Engenharia de Linguagens
Projecto integrado

March 13, 2011

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Até agora:

Concretizado até ao início da segunda fase:

- Descrição do sistema ✓
- Modelação formal e informal do problema ✓
- Modelo de dados ✓
- Início da implementação e respectivo tool demo ✓

Index

- 1 Até agora
- 2 **Objectivos**
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Motivação e Objectivos

Objectivos para segunda fase:

- Terminar a aplicação web
 - compilar e executar o código fonte submetido
 - guardar e apresentar resultados
- Investigação das métricas existentes
- Scripts auxiliares
- Exploração de um frontend

Até agora

Objectivos

Aplicação Web

Scripts Auxiliares

Frontend

Terminal Interface - Powered by Perl

Conclusão e trabalho futuro

Perguntas

Implementação: até à segunda fase

Implementação: linguagens de programação

Implementação: Compilação

Implementação: Execução

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Até agora

Objectivos

Aplicação Web

Scripts Auxiliares

Frontend

Terminal Interface - Powered by Perl

Conclusão e trabalho futuro

Perguntas

Implementação: até à segunda fase

Implementação: linguagens de programação

Implementação: Compilação

Implementação: Execução

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Implementação: até à segunda fase

Já implementado para o último checkpoint:

- Criação de contas de utilizador (grupo)
- Associação de concorrentes a determinado grupo
- Criação de concursos
- Criação de enunciados (através da interface web ou submetendo em formato xml)
- Inserção de baterias de teste para os enunciados
- Submissão de programas para avaliação

Index

- 1 Até agora
- 2 Objectivos
- 3 **Aplicação Web**
 - Implementação: até à segunda fase
 - **Implementação: linguagens de programação**
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Implementação: linguagens de programação

Configuração de linguagens de programação:

- Estando a linguagem correctamente configurada no servidor, é simples preparar o sistema de submissão para avaliar código submetido nessa linguagem
- Para isso basta inserir o comando usado para compilar e para executar, que por exemplo, em C seria:
String compilação: `gcc -O2 -Wall #{file}`
String de execução default: `./a.out`
String de execução para makefile: `./#{file}`

Até agora

Objectivos

Aplicação Web

Scripts Auxiliares

Frontend

Terminal Interface - Powered by Perl

Conclusão e trabalho futuro

Perguntas

Implementação: até à segunda fase

Implementação: linguagens de programação

Implementação: Compilação

Implementação: Execução

Index

1 Até agora

2 Objectivos

3 Aplicação Web

- Implementação: até à segunda fase

- Implementação: linguagens de programação

- **Implementação: Compilação**

- Implementação: Execução

4 Scripts Auxiliares

5 Frontend

6 Terminal Interface - Powered by Perl

7 Conclusão e trabalho futuro

Implementação: Compilação

- Caso seja necessário compilar o código fonte submetido, é usada a string de compilação definida aquando da configuração da linguagem
- Se for submetido um ficheiro comprimido que inclua um makefile, é executado o comando *make* e, o nome do executável criado é obtido a partir de um script perl

Até agora

Objectivos

Aplicação Web

Scripts Auxiliares

Frontend

Terminal Interface - Powered by Perl

Conclusão e trabalho futuro

Perguntas

Implementação: até à segunda fase

Implementação: linguagens de programação

Implementação: Compilação

Implementação: Execução

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - **Implementação: Execução**
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Implementação: Execução

- Para executar o programa para os diferentes inputs, é usada a string de execução simples (no caso de ser submetido apenas um ficheiro) ou a string de execução para makefile (no caso de ser submetido um makefile)
- Para cada input o comando é corrido uma vez
- O output é capturado e comparado com o esperado
- É guardada a percentagem de testes no qual o código submetido passou

Implementação: Apresentação de resultados

- A qualquer altura o utilizador pode consultar os resultados das últimas submissões (suas ou dos restantes participantes)
- Pode também consultar os seus melhores resultados, para cada enunciado

Num Tentativa	Grupo	Concurso	Enunciado	Resultado	Compilou	Terminou a execução
62	Example User	concurso 2 (Activo)	enunciado 1	0.0	true	false
61	Example User	test contest (Activo)	teste2	100.0	true	false
60	Example User	test contest (Activo)	teste2	100.0	true	false

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 **Scripts Auxiliares**
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Scripts auxiliares

- Script (em Perl) para obter o nome do executável gerado pelo makefile (para C)
- Script (em Perl) que gera estatísticas relativamente à quantidade de ficheiros submetidos para cada linguagem de programação

Script makefile.pl

- Utiliza o módulo perl *Makefile::Parser* para fazer parse do makefile, e obter o nome do executável gerado
- No caso de não ser definido um nome para o output, retorna *a.out*
- TODO: Suportar mais linguagens par além do C.

Script count.pl

- Dada uma pasta, explora recursivamente os seus directórios, e extraí várias estatísticas relativas á quantidade de número de linhas
 - Número de linhas de código por linguagem
 - Número de linhas comentadas por linguagem
 - Rácio entre linhas de código e número de ficheiros para cada linguagem
 - Percentagem de linguagem mais usadas no projecto
 - ...
- Utiliza o módulo perl *GD* para gerar gráficos

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Até agora
Objectivos
Aplicação Web
Scripts Auxiliares
[Frontend](#)
Terminal Interface - Powered by Perl
Conclusão e trabalho futuro
Perguntas

Language.C

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 Conclusão e trabalho futuro

Terminal Interface

- Criado para facilitar a manutenção e o acesso à base de dados.
- Interface utilizada apenas pelos **administradores** do sistema.

Terminal Interface - Perl

Escolheu-se esta linguagem devido:

- Rápida implementação
- Vasta diversificação de módulos

Desses módulos, usa-se e pretende-se usar:

- **DBIx::Class**
- **Term::Readline**
- **Digest::SHA2**
- **XML::DT**

Estado actual

Exemplo actual da interface:

```
File Edit View Terminal Help
Dados do User: zacarias

      Email → example-82@railstutorial.org
      Admin → f

Pretender efectuar alterações no user?[S/N]
s

Qualquer campo que não pretenda alterar deixe em branco

→ Inserir novo nome: Zella Douglas
→ Inserir novo email:
→ Admin? [sim]:

Novos dados do utilizador:

Nome: Zella Douglas
Email: example-82@railstutorial.org
Admin: f

Pretender confirmar as alterações? [S/N]
s

User alterado

<=====>
Sair? [S/N]
```

Index

- 1 Até agora
- 2 Objectivos
- 3 Aplicação Web
 - Implementação: até à segunda fase
 - Implementação: linguagens de programação
 - Implementação: Compilação
 - Implementação: Execução
- 4 Scripts Auxiliares
- 5 Frontend
- 6 Terminal Interface - Powered by Perl
- 7 **Conclusão e trabalho futuro**

Conclusão e trabalho futuro

- Implementar várias métricas para C com o FrontEnd
- Tornar a interface utilizador mais intuitiva e mais agradável de utilizar

Perguntas

?