PROTEUM & PROTEUM/IM

JACKSON ANTONIO DO PRADO LIMA

SILVIA REGINA VERGILIO

FERRAMENTAS PARA O TESTE DE MUTAÇÃO

Realizar atividades relacionadas ao critério Análise de Mutantes, mesmo considerando a utilização de estratégias para redução de custo, é uma atividade complexa. Por isso, a utilização de ferramentas de suporte é fundamental. Diversas ferramentas estão disponíveis para diferentes linguagens como C, C++ e Java.

PROTEUM

- Proteum (PROgram Testing Using Mutants)
- Ferramenta desenvolvida no Instituto de Ciências Matemáticas e de Computação – ICMC/USP
- Apoia o teste de mutação principalmente para programas C, mas pode ser configurada para outras linguagens
- Utilizada em SO SunOS, Solaris e Linux

PROTEUM (CONT.)

Possui 71 operadores de mutação divididos em quarto classes:

- Mutação de comandos (*statement mutations*);
- Mutação de operadores (*operator mutations*);
- Mutação de variáveis (*variable mutations*);
- Mutação de constantes (*constant mutations*).

PROTEUM (CONT.)

- Oferece ao testador recursos para, através da aplicação do critério Análise de Mutantes, avaliar a adequação de ou gerar um conjunto de casos de teste *T* para determinado programa *P*.
- Dessa maneira, com nas informações fornecidas pela *Proteum*, o testador pode melhorar a qualidade de *T* até obter um conjunto adequado ao critério.
- Assim, a ferramenta pode ser utilizada como instrumento de avaliação bem como de seleção de casos de teste.

PROTEUM (CONT.)

- Definição de casos de teste;
- Execução do programa em teste;
- Seleção dos operadores de mutação que serão utilizados para gerar os mutantes;
- Geração dos mutantes;
- Execução dos mutantes com os casos de teste definidos;
- Análise dos mutantes vivos;
- Cálculo do escore de mutação.

PROTEUM/IM

Semelhante a *Proteum*, sendo a diferença existente é, basicamente, o conjunto de operadores de mutação que cada uma utiliza e o fato de que a *Proteum* destina-se ao teste de unidade enquanto a *Proteum/IM* oferece características para testar a conexão entre as unidades, ou seja, teste de integração.

PROTEUM/IM (CONT.)

- Dada uma conexão entre duas unidades *F* e *G* (*F* chamando *G*). A mutação é tipicamente realizada nos pontos onde a unidade *F* faz chamada à unidade *G*, por exemplo, incrementando o argumento sendo passado para *G*.

- Possui um total de 33 operadores de mutação separados por dois grupos. O Grupo-I com 24 e o Grupo-II com 9 operadores.

PROTEUM E PROTEUM/IM

Ambas as ferramentas, *Proteum* e *Proteum/IM*, são ambientes compilados baseadas em interfaces com janelas e scripts. Contudo, essas ferramentas não possuem um gerador automático de casos de teste nem a determinação automática de mutantes equivalentes.

PRINCIPAIS CARACTERÍSTICAS

	PokeTool	*Proteum*	*PROTEUM/IM*
Linguagem	C, COBOL, FORTRAN	C	C
Geração automática de casos de teste	Não	Não	Não
Edição de casos de teste	Sim	Sim	Sim
Registro sobre caminhos não executáveis ou mutantes equivalentes	Sim	Sim	Sim
Restrição de tamanho do programa a ser testado	Não	Não	Não
Eliminação de casos de teste redundantes	Sim	Sim	Não
Interface	Menu, Janelas e *Scripts*	Janelas e *Scripts*	Janelas e *Scripts*
Sessões de teste	Sim	Sim	Sim
Apoio a experimentos	Sim	Sim	Sim
Importação de casos de teste	Sim	Sim	Sim
Geração seletiva de mutantes	Não se aplica	Sim	Sim
Ambiente compilado/interpretado	Compilado	Compilado	Compilado
Execução distribuída	Não	Não	Não
Determinação automática de mutantes equivalentes ou caminhos não executáveis (heurísticas)	Sim	Não	Não

PROTEUM/IM 2.0

- Proteum/IM 2.0 é uma ferramenta, para linguagem C, que aplica tanto o teste de unidade quanto o teste de integração, pois integra em um único ambiente a ferramenta Proteum e Proteum/IM.

AULA PRÁTICA

- Requisitos:

```
sudo apt-get install default-jdk
sudo apt-get install default-jre
```

AULA PRÁTICA (CONT.)

- sudo apt-get install git
- sudo apt-get install build-essential libssl-dev libcurl4-gnutls-dev libexpat1-dev gettext unzip
- Vá a um diretório onde deseja realizar os clones abaixo, por exemplo, um diretório denominado GitHub.
- git clone https://github.com/jacksonpradolima/proteum1.4.1.git
- git clone https://github.com/jacksonpradolima/proteumIM2.0.git
- git clone https://github.com/jacksonpradolima/software-testing-examples-c.git

AULA PRÁTICA (CONT.)

- Na pasta de exemplo, GitHub, realizar o seguinte comando: **chmod –R 777 ***
- Cria uma pasta para os experimentos, nesse caso iremos denominar a pasta com o mesmo nome do arquivo a ser testado (*getcmd*).
- Copiar do repositório software-testing-examples-c o arquivo *getcmd*.
- Compilar o arquivo: *gcc -o getcmd getcmd –w*
- Executar o “run” da *Proteum: sh run.sh &*

AULA PRÁTICA (CONT.)

- Criar um novo caso de teste
- Criar os mutantes (*default* 100)
- Executar os mutantes
- Verificar o status
- Executar o relatório
- Definir mutantes equivalentes

AULA PRÁTICA (CONT.)

- Proteum/IM possui a mesma interface, entretanto, para gerar os mutantes possui outros operadores e padrão.

REFERÊNCIAS

J. C. Maldonado, A. M. R. Vincenzi, e M. E. Delamaro. Proteum/IM 2.0: *An integrated mutation testing environment*. Mutation 2000 Symposium, San Jose, CA: Kluwer Academic Publishers, páginas 91–101. Springer, 2000.

Barbosa, E. F., Maldonado, J. C., Vincenzi, A. M. R., Delamaro, M. E., Souza, S. R. S., & Jino, M. (2000). Introdução ao teste de software. *Minicurso apresentado no XIV Simpósio Brasileiro de Engenharia de Software (SBES 2000)*.

MATERIAL

- www.inf.ufpr.br/japlima