PROTEUM e PROTEUM/IM

HELSON LUIZ JAKUBOVSKI FILHO
SILVIA REGINA VERGILIO

Critério Análise de Mutantes

- O critério Análise de Mutantes (ou Teste de Mutação) é conhecido pela sua alta capacidade em revelar defeitos.
- Entretanto, é considerado uma atividade custosa e complexa.
  - Grande quantidade de mutantes e casos de teste devem ser gerados e executados.
- Nesse contexto, a utilização de ferramentas de suporte é fundamental.
- Diversas ferramentas estão disponíveis pra diferentes linguagens, como c, C++ e Java.

PROTEUM

- PROTEUM (PROgram Testing Using Mutants).
- Ferramenta desenvolvida no Instituto de Ciências Matemáticas e de Computação – ICM/USP.
- Tem como foco apoiar o Teste de Mutação em programas C, mas pode ser configurada para outras linguagens.
- Utilizada em SOs como SunOS, Solaris e Linux.

PROTEUM

- Possui 71 operadores de mutação divididos em quatro classes:
  - Mutação de comandos (statement mutations);
  - Mutação de operadores (operator mutations);
  - Mutação de variáveis (variable mutations);
  - Mutação de constantes (constants mutations);

PROTEUM

- Oferece ao testador recursos para, através da aplicação do Teste de Mutação, gerar um conjunto de casos de teste T para um programa P ou avaliar a adequação do conjunto T.
- Assim, com as informações fornecidas pela ferramenta, o testador pode melhorar a qualidade de T até um conjunto adequado ao critério.
- Portanto, PROTEUM pode ser utilizada como instrumento de avaliação bem como seleção de casos de teste.

Principais funcionalidades da PROTEUM

- 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.
- A diferença existente, é basicamente:
  - Conjunto de operadores de mutação que cada uma utiliza;
  - Foco da aplicação do teste de cada ferramenta:
    - PROTEUM destina-se ao teste de unidade.
    - PROTEUM/IM oferece características para testar a conexão entre as unidades, ou seja, teste de integração.

PROTEUM/IM

- 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.
- Ao todo, possui 33 operadores de mutação separados por dois grupos:
  - Grupo-I com 24 operadores; e
  - Grupo-II com 9 operadores.

PROTEUM e PROTEUM/IM

- Ambas as ferramentas, PROTEUM e PROTUEM/IM, são ambientes compilados em interfaces com janelas e scripts.
- Contudo, essas ferramentas não possuem um gerador automático de casos de teste e também não realizam 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.
- Integra em um único ambiente a ferramenta PROTEUM e PROTEUM/IM.

Aula prática

- 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/HelsonLJF/proteum1.4.1.git
- git clone https://github.com/HelsonLJF/proteumIM2.0.git
- git clone https://github.com/HelsonLJF/software-testing-examples-c.git

Aula prática

- Na pasta de exemplo, GitHub, realizar o seguinte comando: chmod –R 777 *
- Criar uma pasta para os experimentos, dentro do diretório da ferramenta.
- Copiar do repositório software-testing-examples-c o arquivo getcmd.c.
- Compilar o arquivo: gcc –o executavel getcmd.c –w
- Executar o “run” da PROTEUM: sh run.sh &

Aula prática

- Criar uma nova seção de teste:
  - Directory: indicar o caminho para a pasta com o getcmd
  - Program Test Naem: qualquer nome para testar
  - Sorce Program: arquivo fonte
  - Executable program: executável gerado
  - Compilation Command: gcc –o executavel getcmd.c –w
- Gerar mutantes (default 100%)
- Adicionar casos de teste
- Executar mutantes
- Visualizar Status e Relatório

Aula prática

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

Referências

- Apresentação adaptada de: https://jacksonpradolima.github.io/teaching/proteum&proteumIM.pdf.
- Maldonado, J. C., Vincenzi, A. M. R. e Delamaro M. E. 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. e Jino, M. (2000). Introdução ao teste de software. Minicurso apresentado no XIV Simpósio Brasileiro de Engenharia de Software (SBES 2000).

Obrigado pela atenção!

Extras-ii

- Caso ocorram problemas com a dimensão do layout, os arquivos da interface podem ser editados. Tais arquivos são encontrados em "/Proteum1.4.1./LINUX/bin/interface"
- Ou ainda, utilize outra versão da PROTEUM.
- git clone https://github.com/HelsonLJF/adjusted-proteum1.4.1.git
- Renomeie a pasta de "adjusted-proteum1.4.1" para "proteum1.4.1".
- Adicione as Variáveis de Ambiente:
  - export PATH={PATH}:/seu_diretório/proteum1.4.1/LINUX/bin
  - export PROTEUM14HOME=/seu_diretório/proteum1.4.1/LINUX/bin
- Entre na pasta "/Proteum1.4.1./LINUX/bin/interface"
  - executar ./proteum
- O restante do processo é o mesmo que foi apresentado no Slide 14.

Extras-ii

- Depois do carregamento do programa, podem aparecer algumas mensagens notificando que um determinado arquivo não foi encontrado ou não pode ser executado.
- Tal erro pode ser decorrente da compatibilidade do SO com aplicações 32bits.
- Nesse caso, deve-se verificar como é realizada a instalação de bibliotecas para suportar aplicações 32bits e instala-las.