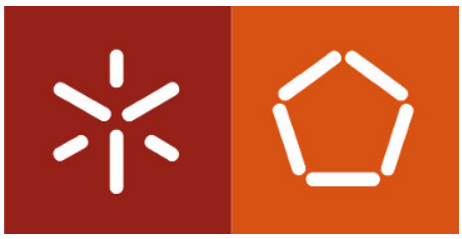
Gestão de Redes

28 de Fevereiro de 2021

Trabalho Prático 3

a83899	André Morais
a84485	Tiago Magalhães

Agente SNMP para monitorização de datas de eventos



Mestrado Integrado em Engenharia Informática
Universidade do Minho

Conteúdo

1 Introdução

Neste trabalho prático, tinhamos como objetivo criar um agente SNMP que implemente uma pequena MIB para monitorização de datas para eventos. As datas monitorizadas são configuradas diretamente no software do agente SNMP através dum ficheiro configuração.

Nestas próximas secções, aquilo que pretendemos é explicar cada um dos passos para a conceção deste projeto e para a sua estruturação.

2 Concepção/Desenho da resolução

Nesta secção secção serão explicados os passos que levaram à criação da ferramenta.

2.1 Tecnologia utilizadas

Primeiramente foram definidas as tecnologias a serem utilizadas.

As APIs de programação utilizadas foram o SNMP4J-Agent, e como consequência a linguagem utilizada foi java, isto deve-se ao facto desta API possuir uma maior documentação, manutenção e também ser compatível com a ferramenta AgenPro que permite através da definição de uma MIB gerar código para um agente já com a MIB estruturada ao nível de código e com suporte à API SNMP4J-Agent, em vez de sermos nós a ter de a estruturar desde o ínicio. Para definição da MIB que serviu de *input* para a ferramenta AgenPro foi utilizada a ferramenta MIB Designer que permite uma maior facilidade e correção para a construção da MIB de acordo com a norma SMI.

2.2 Descrição e justificação da MIB

Após escolhas das tecnologias a utilizar no desenvolvimento tivemos de definir a MIB de modo a podermos utilizar as ferramentas enumeradas anteriormente, assim nesta secção apresentaremos a descrição e a justificação da estrutura escolhida para a MIB, tendo em conta os requisitos apresentados no enunciado de modo a monitorizar as datas dos eventos.

2.2.1 Descrição

A nossa MIB está estruturada com duas tabelas:

- **grMIBTable**: Uma tabela composta com a descrição de cada evento, respetivo índice e mensagem a verificar se o evento ja ocorreu ou se ainda vai acontecer.
- **tempoTable**: Tabela que contém os objetos **"anos"**, **"meses"**, **"dias"**, **"semanas"**, **"horas"** e **"minutos"**.

Na imagem que se segue, permite exemplificar graficamente o que foi listado em cima:

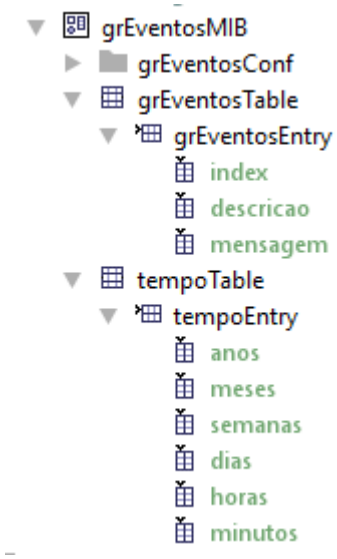


Figura 1: grMIB

2.2.1.1 grMIBTable

- **index**: ID de cada evento. Representado como *Counter32*, com permissões *read-only*.
- **descricao**: Descrição/Título de cada evento. Representado como *OCTET STRING*, com permissões *read-only*.
- **mensagem**: Mensagem sobre o espaço-temporal do evento, isto é, se já ocorreu, se ainda vai acontecer ou mesmo se está a decorrer. Representado como *OCTET STRING*, com permissões *read-only*.

2.2.1.2 tempoTable

- **anos**: Número de anos que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.
- **meses**: Número de meses que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.
- **semanas**: Número de semanas que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.

- **dias**: Número de dias que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.
- **horas**: Número de horas que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.
- **minutos**: Número de minutos que faltam ou já passaram depois do evento. Representado como *INTEGER*, com permissões *read-only*.

2.2.2 Justificação

De acordo com os requisitos presentes no enunciado, uma vez que a MIB irá monitorizar vários eventos, temos uma tabela principal com índice do evento, a sua descrição e a mensagem de alerta.

Após isto como uns dos requisitos é guardar objetos de forma a registar os anos, meses, semanas, dias, horas e minutos que faltam para um evento ou já passaram desde que ocorreu um evento, criamos uma tabela que tem como índice o da tabela principal que corresponde ao evento e como colunas os objetos anos, meses, semanas, dias, horas e minutos.

Optamos por ter apenas uma mensagem de alerta, em que o agente identifica o *frame* temporal e atualiza a mensagem de acordo com o *frame* identificado, para esta atualização e também dos tempos optamos por ler sempre do ficheiro o que é mais lento do que aceder a memória caso os dados estivessem em memória, no entanto como muitas das vezes os agentes estão presentes em dispositivos com algumas restrições como memória optamos por ler do ficheiro, no entanto não é necessário ler linha a linha, uma vez que estamos a utilizar uma API java-configparser, que utiliza uma estrutura INI para o ficheiro de configuração e permite aceder aos eventos através de indexações por secções.

A divisão em diferentes tabelas, deve-se ao facto de permitir uma melhor organização e também cada tabela possui um conceito próprio como tempo.

2.3 Descrição e arquitetura da ferramenta

2.3.1 Diagrama de classes

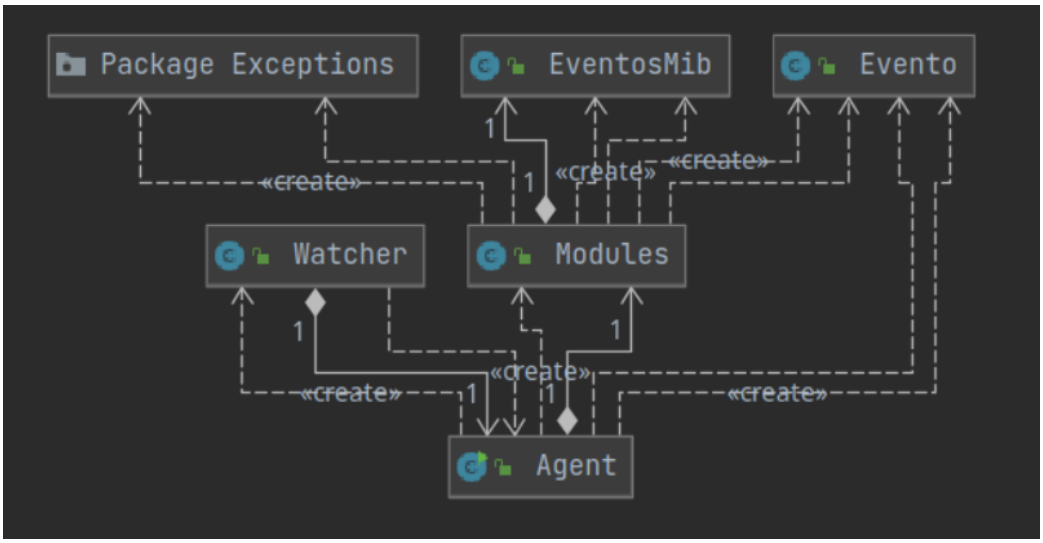


Figura 2: Diagrama de classes

2.3.2 Descrição da arquitetura

Através do diagrama de classes podemos observar que existem 4 classes principais sendo estas:

- **Agent**: Classe que representa o agente e onde é realizado *polling* de 1 minuto para atualizar os objetos da MIB, uma vez que é a menor unidade de medida de tempo presente na MIB é o minuto, também lida com os pedidos snmp.
- **Evento**: Classe que representa um evento com os campos relativos ao índice do evento, data de início e fim, bem como as mensagens de alerta para o passado, presente e futuro.
- **Modules**: Classe responsável pelas funcionalidades da MIB mais concretamente, onde são realizados os cálculos do tempo .
- **EventosMIB**: Classe que representa a estrutura da MIB, de acordo com a API SNMP4J-Agent.

- **Watcher**: Verifica se ficheiro de configuração foi alterado, de modo a que os objetos da MIB possam ser removidos ou atualizados.

2.3.3 Ficheiro de configuração

As datas monitorizadas devem ser configuradas diretamente no software do agente SNMP através dum ficheiro configuração (config.ini).

No ficheiro de configuração cada evento tem de ter uma secção denominada evento concatenado com o seu índice como exemplo: 'evento0' e dentro da sua secção ter os campos descrição, begin_date, end_date, mensagem_passado, mensagem_presente e mensagem_futuro seguido dos seus valores. Os valores atribuídos às datas têm de possuir a seguinte sintaxe 'DD/MM/YYYY-hh:mm'.

Pode ser acrescentado uma campo remove(opcional) para remover um evento de forma imediata ou apagar a secção correspondente a este, porém esta última só irá remover no *polling* seguinte.

Exemplo:

```
[evento0]
descricao = SEI 22
begin_date = 15/03/2022-10:00
end_date = 17/03/2022-18:30
mensagem_passado = O evento terminou ha
mensagem_presente = O evento esta a decorrer
mensagem_futuro = Esta quase a chegar, faltam

[evento1]
descricao = Porto vs Juventus - 1mao
begin_date = 17/02/2021-20:00
end_date = 17/02/2021-21:45
mensagem_passado = O encontro terminou ha
mensagem_presente = O jogo esta a decorrer
mensagem_futuro = Esta quase a chegar, faltam
remove = 1
```

3 Resultados

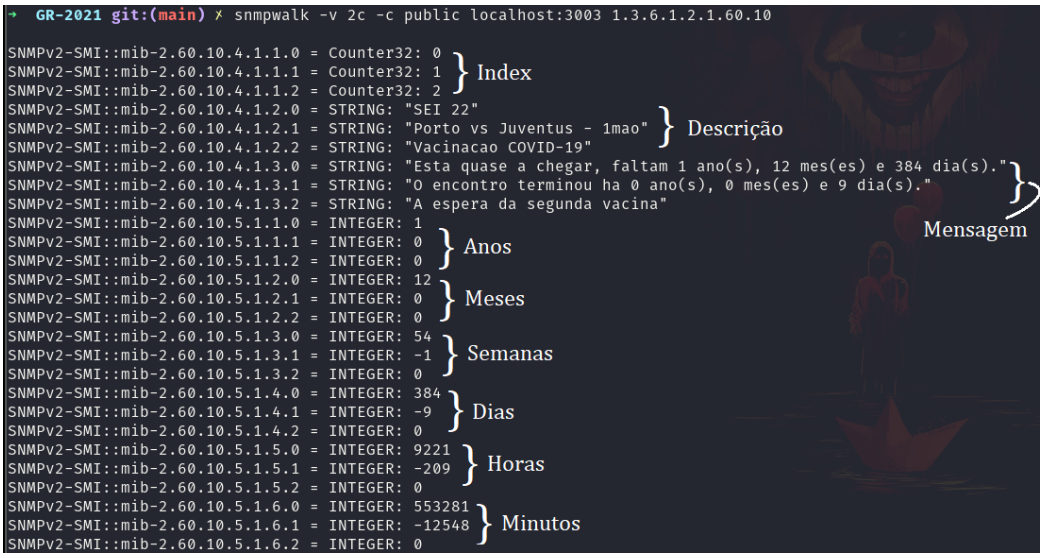


Figura 3: Comando snmpwalk

4 Conclusão

Em suma a realização deste projeto permitiu-nos consolidar a aprendizagem da UC de Gestão de redes, mais concretamente como proceder à realização de um agente SNMP que implemente uma MIB.