CID-10  
Escopo e requisitos do projeto

22/05/2017

# Visão geral

## Descrição e histórico do projeto

Projeto proposto durante a disciplina de Integração de Aplicações do curso de Engenharia de Software – UFG.

## Escopo do projeto

Desenvolver um módulo em Java que permita realizar buscas na tabela CID-10, para encontrar o código de uma doença, bem como sua descrição, presentes na CID-10.

## Requisitos de alto nível

No novo módulo:

* A busca pode ser feita apenas por parte do código, parte da descrição ou de ambos. Por exemplo, a busca por "dengue" deve trazer todas as entradas da CID-10 que contêm "dengue" como parte da descrição. Nesse caso, as entradas cujos códigos são A90 e A91. Por outro lado, se a entrada inclui "90" e "dengue", então apenas a entrada de código A90 é a resposta correta para a consulta. Observe que não é necessário fornecer nem o código nem a descrição completa.
* O módulo **cid10.jar** deve incluir o código que implementa a busca e os dados propriamente ditos sobre os quais a busca é realizada. Ou seja, os arquivos que contêm a CID10 deverão estar embutidos nesse arquivo jar.
* O módulo **cid10.jar** deve incluir a classe CID10 que deve conter os métodos públicos **load** e **unload**. O primeiro carrega e mantém em RAM os códigos e eventuais estruturas para a busca eficiente e o segundo remove da memória RAM qualquer índice e/ou dados para busca eficiente (cache). O método relevante dessa classe é o método **search**, que recebe como argumento um vetor de sequências de caracteres (**String[]**). Esse método retorna **null** caso a entrada fornecida não identifique algum código correspondente e, caso contrário, um vetor de sequências de caracteres, contendo tantas entradas quanto os códigos localizados. A resposta sempre é o código seguido de um espaço em branco seguido da descrição. Por exemplo, um retorno possível é **A90 Dengue (dengue clássico)**, ou seja, o código A90 seguido de espaço que é seguido da descrição correspondente ao código A90.

## Requisitos Funcionais

* RF1 : A busca pode ser feita apenas por parte do código, parte da descrição ou de ambos, não é necessário fornecer nem o código nem a descrição completa.
* RF2: O módulo cid10.jar deve incluir o código que implementa a busca e os dados propriamente ditos sobre os quais a busca é realizada.
* RF3: O módulo cid10.jar deve incluir a classe CID10 que deve conter o método público load() que carrega e mantém em RAM – através de um ArrayList de String – os códigos e eventuais estruturas para a busca eficiente.
* RF4: O módulo cid10.jar deve incluir a classe CID10 que deve conter o método público unload() que remove da memória RAM qualquer índice e/ou dados para busca eficiente (cache).
* RF5: Método search(), que recebe como argumento um vetor de sequências de caracteres (String[]). Esse método retorna null caso a entrada fornecida não identifique algum código correspondente e, caso contrário, um vetor de sequências de caracteres, contendo tantas entradas quanto os códigos localizados. A resposta sempre é o código seguido de um espaço em branco seguido da descrição.
* RF6: O método load() deve ser chamado antes de utilizar o método search(), caso contrário uma exceção em tempo de execução será lançada, indicando que o método search() foi chamado em hora irregular.

## Resultados finais

Embora seja um projeto acadêmico, há possibilidade de utilização desse módulo por outros sistemas que queiram realizar buscas junto ao DATASUS (Departamento de Informática do SUS) na tabela CID-10.

Um provável sistema a utilizar esse módulo é o [Barramento SUS](https://github.com/kyriosdata/db/wiki/Barramento-SUS).

## Partes afetadas

Quaisquer outros sistemas que quiserem utilizar este software como ferramenta para consulta de dados da tabela CID-10, assim como a própria tabela CID-10.

# Aprovação e autoridade para proceder

Aprovamos o projeto como descrito acima e autorizamos a equipe a prosseguir.

|  |  |  |
| --- | --- | --- |
| Nome | Título | Data |
| Erik Raphael | Desenvolvedor e Analista de processos | 22/05/2017 |
| Erivan Barbosa | Gerente e Desenvolvedor | 22/05/2017 |
| Henrique Cardoso | Testador e Desenvolvedor | 22/05/2017 |
| Igor Queiroz | Desenvolvedor e Engenheiro de Requisitos | 22/05/2017 |
| Osmar Cavalcante | Desenvolvedor | 22/05/2017 |
| Rodolpho Hiroshi | Desenvolvedor e Gerente de Configuração | 22/05/2017 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
| Aprovado por e-mail |  |  | Data |  | Aprovado por e-mail |  |  | Data |
| Erik Raphael |  |  | 22/05/2017 |  | Igor Queiroz |  |  | 22/05/2017 |
| Erivan Barbosa |  |  | 22/05/2017 |  | Osmar Cavalcante |  |  |  |
| Henrique Cardoso |  |  | 22/05/2017 |  | Rodolpho Hiroshi |  |  | 22/05/2017 |