1 GODAcademics '15

Grupo:*Pedro Bruel, António Martins Miranda, António Castro Júnior*
Contato:{pedro.bruel, amartmiranda, to.junior.25}@gmail.com

O GODAcademics é um agregador de informações acadêmicas. A partir de um perfil do Google Scholar, ou de um currículo Lattes, esta aplicação constroi um relatório resumindo as informações obtidas de diversas fontes.

1.1 Objetivos

Os objetivos para o primeiro semestre de 2015 foram:

- Refatoração de Testes;
- Aumentar a Cobertura de Testes;
- Aprimorar a página *web* do projeto;
- Implementar a busca *fuzzy* de texto (distância de Levenshtein e cálculo da similaridade);
- Implementar uma interface com o Sistema Lattes.

1.2 Refatorações e Cobertura de Testes

Adicionamos mais casos de teste para a busca por *strings* relativas a conferências e *journals*. Separamos alguns testes de unidade que agrupavam vários métodos no mesmo caso de teste.

1.3 Página do Projeto

Foi implementada uma nova interface para a página *web* do GODAcademics. A *Issue* 450 no *redmine* descreve alguns pedidos de alterações feitos ao grupo *GODWeb*, para que nossa implementação pudesse ser utilizada.
Habilitamos as visualizações para as buscas no *Google Scholar* e currículo *Lattes*. A busca agora é feita através de um único campo, e a distinção entre as fontes é feita baseando-se na assinatura do *link* submetido.

1.4 Busca *Fuzzy*

Foi implementado um algoritmo para a busca aproximada de texto. O algoritmo utilizado foi o cálculo da distância de *Levenshtein*, que permite calcular distâncias entre *strings* e fazer a busca *fuzzy*. Por exemplo, a busca por "*plos comp bio*"deve retornar resultados para o *journal PLOS Computational Biology*.
O método implementado foi `levenshteinDistanceBetween:and:`, que recebe duas *strings* e calcula a distância entre elas.
Tivemos alguns problemas com a implementação dos testes para esse algoritmo, pois o arcabouço de testes para *Smalltalk* utilizado impõe limites para a duração dos testes de unidade.

1.5 Interface com o Sistema Lattes

Tivemos problemas com o interfaceamento e obtenção de informações de Currículos *Lattes*, pois o *site* adotou, recentemente, um sistema de *CAPTCHA*. Uma tentativa de contornar esse problema foi utilizar o *scriptLattes*, uma ferramenta para obtenção desses dados. No entanto, a ferramenta também não funcionou por conta do *CAPTCHA*.
Decidimos então implementar o *parsing* de arquivos html correspondentes a perfis do Currículo Lattes. Desta forma, quando for possível contornar as restrições impostas pelo sistema, já teremos a estrutura para obtenção de informações sobre os pesquisadores pronta.
Criamos a classe `ACADInput`, que permite a criação de uma entrada genérica para o *GODAcademics*. Um objeto `ACADInput` armazena a *url* de um perfil do *Google Scholar* ou Currículo *Lattes*, e referencia um *handler* que contém informações sobre as *tags html* correspondentes ao perfil.

1.6 Trabalhos Futuros

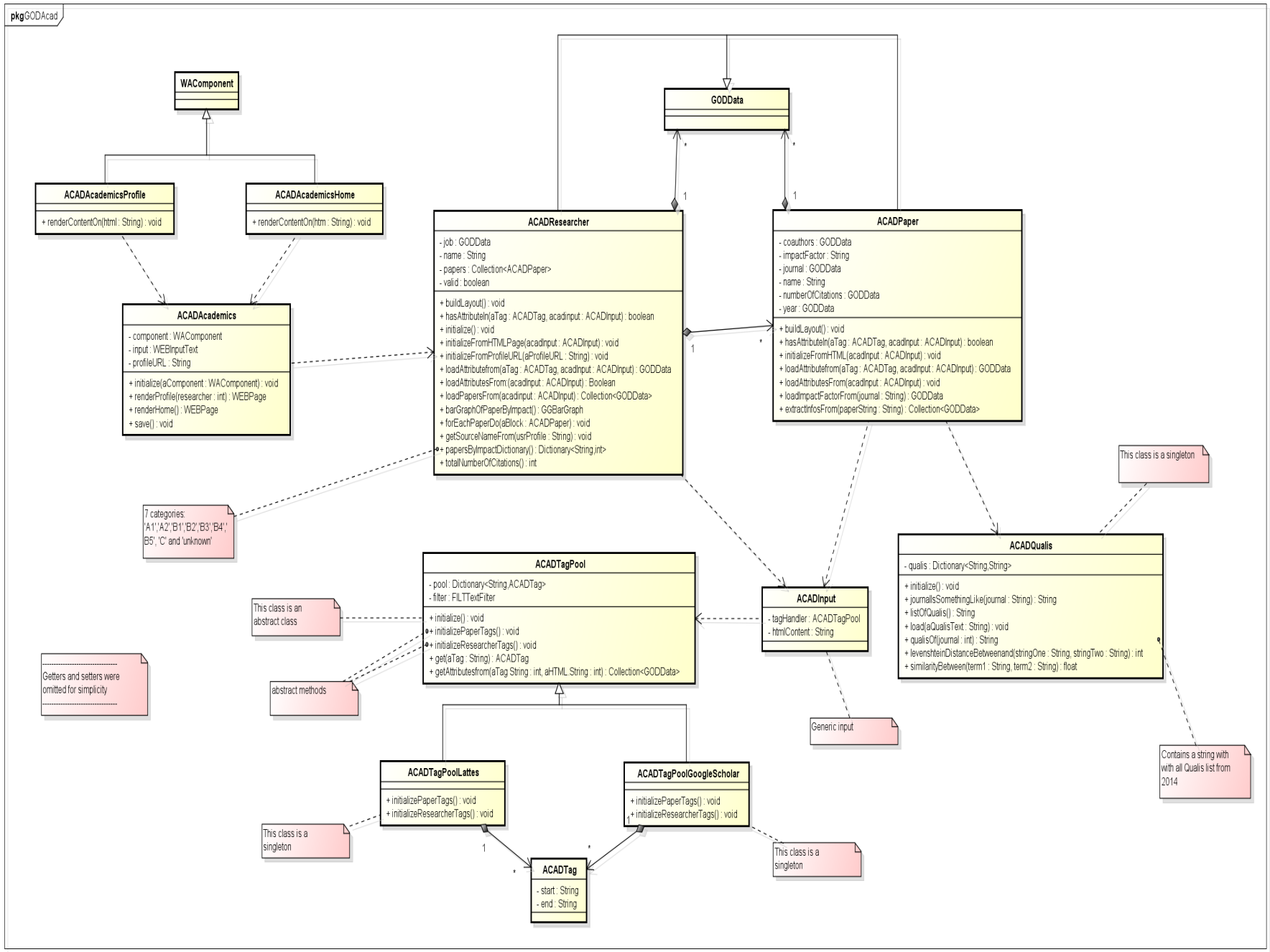
Seria interessante conseguir contornar o *CAPTCHA* para o acesso ao Currículo *Lattes*. Uma ideia para isso seria repassar ao usuário a tarefa de preenchê-lo, seja redirecionando à página do currículo e depois retornando, ou lendo e mostrando o mesmo *CAPTCHA* à partir do GODAcademics. Uma vez que o *parser* para o *Lattes* já está implementado, contornar o *CAPTCHA* permitiria o acesso às informações disponíveis no currículo.
Gostaríamos de implementar mecanismos de análise dos grupos de pesquisa ao redor do Brasil, agrupando temas, laboratórios e pesquisadores geograficamente. Isso poderia ser feito através de informações disponíveis nos Currículos *Lattes* de pesquisadores.
Essas informações geográficas poderiam ser utilizadas na forma de um mapa exibido pelo GOD, que mostraria a distribuição e o interesse em determinado tema de pesquisa no Brasil.
Outra possibilidade poderia envolver a implementação de um *webcrawler* para o Sistema *Lattes*, que agregasse informações continuamente sobre os currículos.

Algumas correcções e otimimazões a serem levandas em conta:

- na home do *GODAcademics*, validar os campo para a inserção de links;
- melhorar eficência do atual algoritmo *fuzzy* (cálcula da similaridade entre duas strings por meio da distância do *Levenshtein*) ou implementar outra estratégia de busca *fuzzy* mais elaborada(algoritmos do Smith–Waterman, Needleman–Wunsch,Jaro–Winkler, etc);
- aumentar a precisão do *Levenshtein*, fazendo um tratamento mais rigoroso das strings a serem comparadas ;
- estudar e determinar qual a melhor porcentagem de similaridade a ser usado como critério de aceitação ou não entre duas strings;
- implementar mais tags para o *Lattes*, como por exemplo, nível de produtividade;

1.7 Diagrama de classes

A seguir apresntamos o diagrama de classes (final) do *GODAcademics* após as alterções:


pkg GODAcad
WAComponent
GODData
ACADAcademicsProfile
+ renderContentOn(html : String) : void
ACADAcademicsHome
+ renderContentOn(htm : String) : void
ACADResearcher
- job : GODData
- name : String
- papers : Collection<ACADPaper>
- valid : boolean
+ buildLayout() : void
+ hasAttributeIn(aTag : ACADTag, acadinput : ACADInput) : boolean
+ initialize() : void
+ initializeFromHTMLPage(acadInput : ACADInput) : void
+ initializeFromProfileURL(aProfileURL : String) : void
+ loadAttributefrom(aTag : ACADTag, acadInput : ACADInput) : GODData
+ loadAttributesFrom:(acadInput : ACADInput) : Boolean
+ loadPapersFrom(acadinput : ACADInput) : Collection<GODData>
+ barGraphOfPaperByImpact() : GGBarGraph
+ forEachPaperDo(aBlock : ACADPaper) : void
+ getSourceNameFrom(usrProfile : String) : void
+ papersByImpactDictionary() : Dictionary<String,int>
+ totalNumberOfCitations() : int
ACADPaper
- coauthors : GODData
- impactFactor : String
- journal : GODData
- name : String
- numberOfCitations : GODData
- year : GODData
+ buildLayout() : void
+ hasAttributeIn(aTag : ACADTag, acadInput : ACADInput) : boolean
+ initializeFromHTML(acadInput : ACADInput) : void
+ loadAttributefrom(aTag : ACADTag, acadInput : ACADInput) : GODData
+ loadAttributesFrom(acadInput : ACADInput) : void
+ loadImpactFactorFrom(journal : String) : GODData
+ extractInfosFrom(paperString : String) : Collection<GODData>
ACADAcademics
- component : WAComponent
- input : WEBInputText
- profileURL : String
+ initialize(aComponent : WAComponent) : void
+ renderProfile(researcher : int) : WEBPage
+ renderHome() : WEBPage
+ save() : void
7 categories:
'A1','A2','B1','B2','B3','B4','B5', 'C' and 'unknown'
This class is a singleton
ACADQualis
- qualis : Dictionary<String,String>
+ initialize() : void
+ journallsSomethingLike(journal : String) : String
+ listOfQualis() : String
+ load(aQualisText : String) : void
+ qualisOf(journal : int) : String
+ levenshteinDistanceBetweenand(stringOne : String, stringTwo : String) : int
+ similarityBetween(term1 : String, term2 : String) : float
ACADTagPool
- pool : Dictionary<String,ACADTag>
- filter : FILTTextFilter
+ initialize() : void
+ initializePaperTags() : void
+ initializeResearcherTags() : void
+ get(aTag : String) : ACADTag
+ getAttributesfrom(aTag String : int, aHTML String : int) : Collection<GODData>
This class is an abstract class
abstract methods
ACADInput
- tagHandler : ACADTagPool
- htmlContent : String
Generic input
Getters and setters were omitted for simplicity
ACADTagPoolLattes
+ initializePaperTags() : void
+ initializeResearcherTags() : void
ACADTagPoolGoogleScholar
+ initializePaperTags() : void
+ initializeResearcherTags() : void
This class is a singleton
This class is a singleton
ACADTag
- start : String
- end : String
Contains a string with with all Qualis list from 2014
1
*