

UFRJ

TELECOMUNICAÇÕES

TRABALHO FINAL

SIGAPI APIs para a UFRJ

Autor

Bernardo AMORIM

Professor:

Ph.D. Fernando Gil

10 de Março de 2016



UNIVERSIDADE
FEDERAL DO
RIO DE JANEIRO

UFRJ

Resumo

This paper describes how to transform features available through UFRJ online services such as SIGA and SAP into Application Programmable Interfaces (API) so that other developers can use it to create more functionality on top of it with ease.

1 Introdução

O objetivo do trabalho era mostrar como é possível transformar serviços web já disponíveis em formatos para humanos em formatos para máquinas (formatos estruturados). Em especial, foram utilizados serviços da UFRJ e três serviços foram selecionados, por apresentarem características diferentes que podem demonstrar como a complexidade varia de acordo com como foi implementada a funcionalidade.

Os serviços selecionados em ordem de dificuldade são:

1. Lista de Cursos e Currículos
2. Sistema de Acompanhamento de Processos - SAP
3. Sistema Integrado de Gestão Acadêmica - SIGA.

Neste serviço foi apenas utilizada a emissão de documentos, como o Certificado de Registro em Inscrição de Disciplinas (CRID) por exemplo.

2 Sobre o protocolo HTTP

Com o objetivo de deixar claro como as coisas funcionam e porque algumas medidas foram tomadas, é importante primeiro entender um pouco sobre o protocolo HTTP.

O HTTP é um protocolo pequeno e flexível, portanto deixa em aberto muitas questões sobre como se criar serviços HTTP.

3 O WebServer e as APIs

Todas as APIs do projeto foram feitas para exibir tudo em um formato mais bem estruturado, em JavaScript Object Notation (JSON).

Para servir os dados foi criado um servidor HTTP utilizando *Node.JS*¹ e a biblioteca *Koa.JS*²

O KoaJS serve para criar um endpoint HTTP e realizar uma ação, o arquivo *index.js* realiza essa ligação (Ver anexo 8.1)

4 Lista de Cursos e Currículos

A parte mais simples do trabalho foi obter a lista de cursos e seus currículos. Por seus currículos entenda informações sobre o curso e as disciplinas obrigatórias em cada período.

4.1 O serviço existente

Para utilizar como fonte de dados, utilizei uma página disponibilizada no site da UFRJ.



Lista de Cursos	
Cursos e seus Desdobramentos	Versões Curriculares
Administração	1995/1 a 1997/2 1998/1 a 9999/9
Arquitetura e Urbanismo	1982/1 a 1995/2 1996/1 a 2005/2 2006/1 a 9999/9
Artes Cênicas	1983/1 a 2000/2 2014/1 a 9999/9
Artes Visuais - Escultura	2011/2 a 9999/9
Astronomia	1983/1 a 2007/2 2008/1 a 9999/9
Bacharelado em Ciência da Computação	2004/1 a 2009/2 2010/1 a 9999/9
Bacharelado em Letras	1983/1 a 1997/2 1998/1 a 9999/9
Bacharelado em Psicologia	1982/1 a 1987/1

Figura 1: Página da listagem de cursos

Por mais que pareça simples, na verdade essa página é consistida de outras

duas páginas. Uma contém apenas a caixa de seleção do nível do curso. Esta página é responsável por carregar uma segunda página no *frame* inferior.

Após escolhido o nível, é possível carregar a lista de cursos e suas especializações. Ao lado direito de cada curso existem as versões curriculares deles, que contêm cada um link para a página do currículo.

A página é composta de 3 partes, um *frame* decorativo (removido na imagem), um separado para mostrar os detalhes do curso, e um contendo a seleção da distribuição curricular, que após abrir algumas dezenas de cursos na mão, verifiquei que apenas 1 distribuição é usada (isto fez com que, na hora de produzir a API fosse utilizada apenas a primeira).

Ela contém alguns detalhes do curso, vide Figura 2, e também as informações da distribuição curricular contêm também as disciplinas do curso, conforme a figura 3.

Distribuição Curricular Engenharia de Computação e Informação-Integral - Cidade Universitária ▼

Atualizado em: 15/02/2016 12:50

Habilitação de Graduação em Engenharia de Computação e Informação			
Currículo a ser cumprido pelos alunos de 2012/1 a 9999/9			
Localização:	Escola Politécnica	Código:	3601012000
Durações		Estrutura	
Prazo máximo de integralização na UFRJ:	15 segmento(s)	Trabalho de Conclusão:	Projeto
Duração recomendada na UFRJ:	10 segmento(s)	Desenvolvido em Parceria:	não
Número mínimo de horas (CNE)	3600	Possui Banca Examinadora:	sim
Duração em anos (CNE):	Mínima:5 Média :0 Máxima:0	Pago:	não
Dados da coordenação		Características	
Responsável:	JANO MOREIRA DE SOUZA(Doutorado)	Modalidade:	Presencial
Matrícula:	24505820700	Denominação Oficial:	Engenharia de Computação e Informação
C.H.:	DE		
Email:	jano@cos.ufrj.br	Site:	http://www.poli.ufrj.br/

Figura 2: Página de detalhes do curso

Distribuição Curricular Engenharia de Computação e Informação-Integral - Cidade Universitária ▼

Atualizado em: 15/02/2016 12:50

6º Período					
Código	Nome	Créditos	C.H.G. Teórica/Prática		Requisitos
COC361	Inteligência Computacional	4.0	60	0	
COE363	Telecomunicações	4.0	60	0	
COS360	Otimização	4.0	60	0	COS110 (P), MAC238 (P), MAE125 (P)
EEL873	Engenharia de Software	4.0	45	30	
EEL879	Redes de Computadores II	4.0	60	0	
EEL882	Computacao Grafica	4.0	45	30	
Atividades Academicas Optativas (GrupoACE)		0.0	0	45	
Total de Créditos		24.0			

7º Período					
Código	Nome	Créditos	C.H.G. Teórica/Prática		Requisitos

Figura 3: Disciplinas do curso

4.2 Criando a API

Primeiro foi criado um módulo *curriculum.js* (Ver anexo 8.2) contendo os seguintes métodos:

1. *getLevels*: Obtem os níveis de cursos (Aperfeiçoamento, Graduação, Mestrado, etc...)
2. *getCourses*: Obtém, dado um nível, os cursos disponíveis e suas especializações.
3. *getAllCourses*: Um método que obtem todos os níveis utilizando o *getLevels* e para cada nível obtém os seus cursos.
4. *getCourse*: Dado um curso, obtém informações relevantes do curso, como orientador e disciplinas obrigatórias de cada período.

Cada um desses métodos faz requisições HTTP para as páginas do serviço original, utiliza algumas Expressões Regulares para extrair os dados e cria um objeto JavaScript para no futuro ser convertido em *JSON*.

Por último era preciso criar os *endpoints* no servidor HTTP, mapeando cada um deles para uma dos métodos implementados:

1. `/levels/` \Rightarrow `getLevels`
2. `/levels/:id/courses` \Rightarrow `getCourses`
3. `/courses` \Rightarrow `getAllCourses`
4. `/courses/:id` \Rightarrow `getCourse`

```
[  
  - {  
    title: "Aperfeiçoamento",  
    id: "AC3274D7-762A-497B-B5C5-EE2117F2A096"  
  },  
  - {  
    title: "Doutorado",  
    id: "16F17D95-5640-4F4C-B55F-631F6F0DF5C7"  
  },  
  - {  
    title: "Especialização",  
    id: "6B516876-CE08-4C73-A5A7-C6BE1E02F201"  
  },  
  - {  
    title: "Extensão",  
    id: "3F4F1BA5-2717-45BC-8CC0-C9FDC66E15BE"  
  },  
  - {  
    title: "Graduação",  
    id: "80167CF7-3880-478C-8293-8E7D80CEDEBE"  
  },  
  - {  
    title: "Mestrado",  
    id: "C26524B9-D1D4-4455-9395-7BF39195716E"  
  },  
  - {  
    title: "Mestrado Profissional",  
    id: "A3DFB7D1-92A4-F79C-016A-E0797B32F411"  
  },  
  - {  
    title: "Residência",  
    id: "09B92099-B020-4AFF-BBEC-91F11B733F45"  
  }  
]
```

Figura 4: Respostas da api de níveis

```
[
  - {
    name: "Administração",
    - specializations: [
      - {
        name: "Administração - Administração Internacional",
        - versions: [
          - {
            period: "1995/1 a 1997/2",
            id: "9BAE6355-92A4-F713-002D-7A10895C5212"
          },
          - {
            period: "1998/1 a 9999/9",
            id: "9BAE63B3-92A4-F713-002D-7A10F9796673"
          }
        ]
      },
      - {
        name: "Administração - Estratégia Empresarial",
        - versions: [
          - {
            period: "1995/1 a 1997/2",
            id: "9BAE6365-92A4-F713-002D-7A1062E7C307"
          },
          - {
            period: "1998/1 a 9999/9",
            id: "9BAE63C2-92A4-F713-002D-7A10C6C3821F"
          }
        ]
      },
      - {
        name: "Administração - Finanças e Controle",
        - versions: [
          - {
            period: "1995/1 a 1997/2".
          }
        ]
      }
    ]
  }
]
```

Figura 5: Respostas da api de cursos

```

{
  title: "Engenharia de Computação e Informação 2012/1 a 9999/9",
  localization: "Escola Politécnica",
  code: "3601012000",
  coordinator: "JANO MOREIRA DE SOUZA(Doutorado)",
  - periods: [
    - [
      - {
        code: "COS110",
        name: "Algoritmos e Programação",
        credits: "5.0",
        theory_hours: "60",
        practice_hours: "30",
        requirements: " "
      },
      - {
        code: "COS111",
        name: "Introd Eng Comput e Informação",
        credits: "2.0",
        theory_hours: "30",
        practice_hours: "0",
        requirements: " "
      },
      - {
        code: "EEL280",
        name: "Circuitos Logicos",
        credits: "5.0",
        theory_hours: "60",
        practice_hours: "30",
        requirements: " "
      },
      - {
        code: "FIS111",
        name: "Física Experimental I",
        credits: "3.0"
      }
    ]
  ]
}

```

[+ - View source](#) 

Figura 6: Respostas da api de curso

5 Sistema de Acompanhamento de Processos

O Serviço de Acompanhamento de Processos é a ferramenta online da UFRJ para que interessados possam consultar o andamento de processos dentro da faculdade, como dispensa de disciplinas e equivalência de créditos.

5.1 O serviço existente

O serviço consiste em duas formas de utilização: Uma é consultar direto pelo número do processo e outra permite pesquisar utilizando alguns campos (Ver figura 7)

Ao clicar em *pesquisar*, o usuário é apenas notificado de quantos resultados foram retornados e ele deve clicar em *resultados*, o que leva ele a página contendo os resultados (Ver figura 8)

Educação
Ministério da Educação

Opções: ▼

Universidade Federal do P

Pesquisa de Processos: Ordem: Número ▼ Critérios Resultado Ajuda

Critérios da Pesquisa

Campos com conteúdo variável (aceitam * e ?):

Número:

Interessado:

Campos com conteúdo exato (não aceitam * e ?):

Unidade: Documento:

Arquivo:

Assunto:

Data inicial: Data final:

Pesquisar Limpar

Figura 7: Interface de pesquisa do SAP

Educação

Ministério da Educação

Opções:

Universidade Federal do R

Pesquisa de Processos:

Ordem:

Número

Critérios

Resultado

Ajuda

Processos: registros 1 a 3 de 3 encontrados

	Número	Interessado / Unidade / Assunto	Doc. / Data
1	23079.000639/1996-39	BERNARDO AMORIM FTESM-Fundação Técnico Educacional Souza Marques (00.00.04.06) Registro de diploma/apostila (0698-0)	000034/1996 08/01/1996
2	23079.010843/2014-10	BERNARDO DORNELLAS CYSNEIROS GOMES DE AMORIM CT- Centro de Tecnologia (36.00.00.00) Assuntos academicos (0671-8)	000309/2014 25/02/2014
3	23079.042694/2014-58	BERNARDO DORNELLAS CYSNEIROS GOMES DE AMORIM CT- Centro de Tecnologia (36.00.00.00) Assuntos academicos (0671-8)	001174/2014 05/09/2014

dgdi

TIC

Figura 8: Interface com resultados de uma pesquisa SAP

5.2 Criando a API

O principal problema em criar uma API para o SAP é relacionado com ele **não** ser *stateless*, isto é, ele **guarda estado** de uma página para a outra.

Ele faz isso utilizando os chamados *Cookies*, e utiliza isso para guardar a ultima pesquisa. Portanto quando clicamos em *pesquisar* o sistema, ao invés de pesquisar e retornar em seguida os resultados, ele guarda um estado no servidor do SAP que é lembrado ao clicarmos em *resultados*.

Eles utilizam um conceito chamado *sessões* e faz isso criando um cookie *ASPSESSION* onde ele guarda um identificador de sessão que é guardado no servidor deles também. Portanto, quando o servidor nos retorna um header *Set-Cookie* o programa deve guardar esse cookie para fazer requisições futuras.

Dado isso, o nosso programa não pode, assim como no módulo de curriculos, simplesmente fazer uma requisição pedindo os resultados. Ele deve seguir um passo a passo:

1. Fazer uma requisição para `http://sap.ufrj.br` e obter o conteúdo do header *Set-Cookie* e guardar os cookies.
2. Utilizando os cookies obtidos, fazer uma requisição *POST* para `http://sap.ufrj.br/pesquisarCritR.asp`
3. Ainda utilizando os cookies obtidos, fazer uma requisição *POST* para `http://sap.ufrj.br/pesquisarR.asp` para em fim, obter uma resposta HTML
4. Extrair informações sobre os processos do HTML retornado.

Portanto o módulo *sap.js* foi criado (Ver anexo 8.3) e ao entrar no endpoint `/processes/search?q=QUERY` um JSON é obtido contendo a listagem de processos, vide figura 9.

```

[
  - {
    name: "BERNARDO AMORIM ",
    code: "23079.000639/1996-39",
    place: "FTESM-Fundaçao Tecnico Educacional Souza Marques (00.00.04.06)",
    subject: "Registro de diploma/apostila (0698-0)"
  },
  - {
    name: "BERNARDO DORNELLAS CYSNEIROS GOMES DE AMORIM",
    code: "23079.010843/2014-10",
    place: "CT- Centro de Tecnologia (36.00.00.00)",
    subject: "Assuntos academicos (0671-8)"
  },
  - {
    name: "BERNARDO DORNELLAS CYSNEIROS GOMES DE AMORIM",
    code: "23079.042694/2014-58",
    place: "CT- Centro de Tecnologia (36.00.00.00)",
    subject: "Assuntos academicos (0671-8)"
  }
]

```

Figura 9: Resposta da API do sap

6 SIGA e Emissão de documentos


6.1 O serviço existente

Para emitir um documento, o usuário deve seguir os seguintes passos

1. Fazer login utilizando CPF e senha (Ver figura 10)
2. Clicar na aba documentos
3. Clicar no documento desejado (Ver figura 11)

6.2 Criando a API

Embora o uso por um usuário normal seja extremamente simples, a aplicação em si é extremamente complicada. Basicamente porque utiliza muito código



UFRJ Sistema Integrado de Gestão Acadêmica

Identificação:

Senha:

[Recuperar Senha](#)

Instruções básicas de acesso

Acesso ao sistema

Entre com sua Identificação (CPF para brasileiros ou N° passaporte para estrangeiros) e sua senha. Após, clique em "Enviar".

Primeiro acesso ao Portal do Aluno?

Caso este seja seu primeiro acesso neste portal, você pode usar sua senha da Intranet UFRJ normalmente. Caso não lembre, use a opção "Recuperar Senha".


Esqueceu ou Perdeu sua Senha

Clique no link "Recuperar Senha" e siga o procedimento indicado na página.


Figura 10: Página de login do SIGA

Meus Documentos Autenticação


Selecione a matrícula:




Regularmente Matriculado




Histórico



Boletim



BOA



CRID

Figura 11: Seleção de documento no SIGA

javascript no cliente, dificultando que consigamos reproduzir o funcionamento apenas usando algumas requisições HTTP. Além disso muitas validações são feitas (provavelmente para evitar esse tipo de engenharia reversa) e essas validações tornam essa reprodução muito complicada.

Dada estas circunstâncias, um método diferente foi utilizado. Desta vez foi simulada a interação do usuário utilizando um navegador *headless*, isto é, sem interface gráfica e através de um script, automatizamos as ações que um

usuário normal faria para baixar um documento (as mesmas citadas acima)

Por fim este documento PDF deve ser transformado até extrairmos em formato estruturado as informações dele. Um exemplo está no diagrama abaixo:

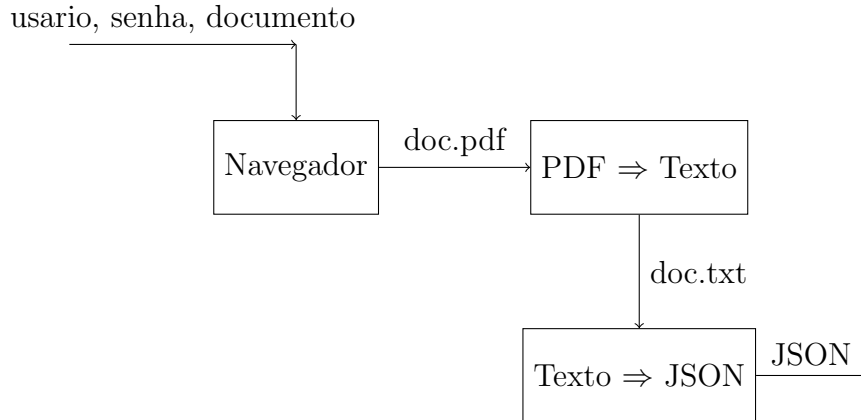


Figura 12: Diagrama do processo de obter documentos

No primeiro bloco, utilizamos como navegador o *PhantomJS*³ e além disso para tornar a criação de scripts para interagir com o navegador e facilitar o download de arquivos, foi utilizada a biblioteca *CasperJS*⁴.

O que foi criado foi um script para o *CasperJS*, o *get_documents.js* (Ver anexo 8.6). Este script baixa um arquivo pdf contendo o documento.

Para transformar o pdf em texto, utilizei a ferramenta *pdftotext*⁵. Para simplificar este comportamento foi criado um shell script, o *get_documents.sh* (Ver anexo 8.5), que salva o PDF para um arquivo temporário e depois utiliza o *pdftotext* para imprimir para a saída padrão o texto final.

Por fim, para converter o texto para um formato estruturado, foi criado o módulo *documents.js* (Ver anexo 8.4 que para motivos de demonstração só implementava a transformação do CRID na função *getCrid*).

Ao acessar o endpoint `/documents/crid?username=CPF&password=SENHA` ele obtinha um JSON contendo todas as disciplinas inscritas, vide figura 13.

6.3 Problemas

Tudo estava funcionando bem, até que na semana anterior a apresentação alguns eventos aconteceram:

```
[
  - {
    control: "6681",
    code: "FIN241",
    title: "Física Experimental IV - EPT2 ",
    hours: "30",
    credits: "1.0",
    situation: "Inscrição normal"
  },
  - {
    control: "10172",
    code: "COP509",
    title: "COP509 - Trabalho do Futuro - ECI/EEL ",
    hours: "75",
    credits: "5.0",
    situation: "Inscrição normal"
  },
  - {
    control: "12191",
    code: "COS351",
    title: "COS351 - Lógica Matemática - ECI ",
    hours: "60",
    credits: "4.0",
    situation: "Inscrição normal"
  },
  - {
    control: "6243",
    code: "FIM240",
    title: "Física-IV-A / ET1+ER1+NTA+IQG ",
    hours: "60",
    credits: "4.0",
    situation: "Inscrição normal"
  },
  - {
    control: "10160"
  }
]
```

Figura 13: Resposta da API do crid

1. O SIGA saiu do ar por algumas horas.
2. O SIGA mudou a forma como os documentos eram baixados, e essa nova forma, além de diferente (o que inutilizava o script do *CasperJS*) não funcionava.
3. O SIGA voltou com a versão antiga, mas talvez por estarmos no final do período, não dá para emitir o CRID, apenas os outros documentos.

Dada estas circunstâncias, o que eu fiz foi utilizar um crid já baixado e conectar ele diretamente ao módulo *documents.js*.

7 Conclusão

Embora o objetivo fosse criar bem mais APIs, o tempo tomado para algumas funcionalidades foi muito grande, principalmente pela necessidade de fazer

Engenharia Reversa com muitas coisas.

No geral a experiência foi positiva e até mesmo algumas funcionalidades complicadas (como emitir documentos) foi realizada com sucesso.

O maior problema de manter APIs desta forma é que fica dependendo de os fornecedores das bases de dados originais não mudarem o serviço deles, o que aconteceu inclusive durante o andamento do projeto com as mudanças no SIGA.

Entretanto, essa é exatamente a motivação para criar essas APIs, pois caso algo mude, apenas essa caixa preta deve ser mudada, possibilitando que muitos outros desenvolvedores poupem trabalho.

Obviamente os códigos produzidos não são 100% a prova de erros e prontos para serem colocados em produção. Eles são apenas experimentos que funcionam para casos de uso específicos, muito esforço deveria ser colocado para criar essas APIs no futuro.

Uma parte interessante que ficou faltando que foi prometida era o armazenamento de usuário e senha de forma criptografada para que a autenticação fosse feita apenas com um token, mas como a ligação com a função de obter o crid foi feita diretamente com um arquivo salvo e não sobrou muito tempo, esta funcionalidade ficou de fora.

8 Anexos

Todos os arquivos também podem ser encontrados no repositório do github⁶

8.1 index.js

```
1 var app = require('koa')();
2 var router = require('koa-router')();
3 var {getLevels, getCourses, getAllCourses, getCourse} = require('./curriculum');
4 var {getCrid} = require('./documents.js');
5 var {search} = require("./sap.js");
6
7 // Helper function to create routes using JSON
8 function mount(route, promiseFactory){
9   router.get(route, function(ctx, next){
10     return promiseFactory(this).then((resp) => {
11       this.body = JSON.stringify(resp);
```



```

12     this.set("Content-Type", "text/json; charset=utf-8");
13   });
14 });
15 }
16
17 // Mount the curriculum-related actions
18 mount('/levels', getLevels);
19 mount('/levels/:id/courses', (ctx) => getCourses(ctx.params.id));
20 mount('/courses', getAllCourses);
21 mount('/courses/:id', (ctx) => getCourse(ctx.params.id));
22
23 // Sample endpoint for processing the CRID document
24 mount('/documents/crid', (ctx) => getCrid(ctx.params.username, ctx.params.password));
25
26 // Mount the SAP search action
27 mount('/processes/search', (ctx) => search(ctx.request.query.q));
28
29 // Setup middlewares
30 app
31   .use(router.routes())
32   .use(router.allowedMethods());
33
34 app.listen(3000);

```

8.2 curriculum.js

```

1  var fetch = require('node-fetch');
2  var cheerio = require('cheerio');
3
4  // Helper for matching all occurrences of a regex
5  function matchAll(str, regex){
6    var match = str.match(regex);
7
8    if(!match)
9      return [];
10
11    return [match].concat(matchAll(str.slice(match.index+1), regex));
12  }

```

```

13
14 function getLevels(){
15     var url = 'https://siga.ufrj.br/sira/repositorio-curriculo/comboListaCursos.html';
16     var regex = /value="([A-Z0-9-]+\)\.html">([<]*)</;
17
18     return fetch(url)
19         .then( (resp) => resp.text() )
20         .then( (body) => (
21             matchAll(body,regex)
22             .map( (match) => ({title: match[2], id: match[1]}) )
23         ) );
24 }
25
26 function getCourses(id){
27     function parseCourse(tr, $, specializing){
28         var nameClass = specializing ? ".identacao2" : ".identacao1";
29         var nameEl = $('${nameClass} b', tr);
30         var name;
31
32         var versions = $("td", tr).not(nameClass).children(".linkNormal").toArray()
33             .map( (a) => {
34                 return {
35                     period: $(a).text(),
36                     id: $(a).attr('href').match(/repositorio\-\curriculo\/([A-Z0-9-]+\)\.html/);
37                 };
38             } );
39
40         if(nameEl.children("a").length == 0){
41             name = nameEl.text();
42         } else {
43             name = nameEl.children("a").text();
44         }
45
46         if(specializing) return {name, versions};
47
48         var specializations = $('tbody#p${nameEl.attr("title")} .tableBodyBlue2')
49             .toArray()
50             .map( (tr) => parseCourse(tr, $, true) )

```

```

51
52     return { name, specializations, versions }
53 }
54
55 function parseBody(body) {
56     var $ = cheerio.load(body);
57     return $("tr.tableTitleBlue")
58         .toArray()
59         .slice(1)
60         .map( (tr) => parseCourse(tr, $) )
61 }
62
63 return fetch('https://siga.ufrj.br/sira/repositorio-curriculo/${id}.html')
64     .then( (resp) => resp.text() )
65     .then( parseBody );
66 }
67
68 function getAllCourses(){
69     return getLevels().then( (levels) => Promise.all(
70         levels
71         .map( ({id,name}) => (
72             getCourses(id)
73             .then((courses) => courses.map( (props) => Object.assign({levelId: id, lev
74         ) )
75         .reduce((acc, arr) => acc.concat(arr), [])
76     ) );
77 }
78
79 function getCourse(id){
80     console.log("GETTING COURSE", id);
81     var periodRegex = /(\d+)º Período/
82
83     function parsePeriod($, table){
84         return $("[class^=tableBodyBlue]",table)
85             .toArray()
86             .map(function(d){
87                 return {
88                     code: $("td", d).eq(0).text(),

```

```

89         name: $("td", d).eq(1).text(),
90         credits: $("td", d).eq(2).text(),
91         theory_hours: $("td", d).eq(3).text(),
92         practice_hours: $("td", d).eq(4).text(),
93         requirements: $("td", d).eq(5).text()
94     }
95 })
96 }
97
98 function parseBody(body){
99     var $ = cheerio.load(body);
100
101     var tableTitle = (table) => {
102         return $(".tableTitle b",table).text()
103     };
104     var tables = $(".table.lineBorder");
105
106     var periods = tables
107         .toArray()
108         .filter( (table) => {
109             return tableTitle(table).match(periodRegex)
110         })
111         .map((table) => parsePeriod($,table))
112
113     return {
114         title: $(".title").text(),
115         localization: $("td", tables[0]).eq(4).text(),
116         code: $("td", tables[0]).eq(6).text(),
117         coordinator: $("td", tables[0]).eq(33).text(),
118         periods
119     }
120 }
121
122 return fetch('https://siga.ufrj.br/sira/repositorio-curriculo/combo${id}.html')
123     .then( (resp) => resp.text() )
124     .then( (body) => {
125         var $ = cheerio.load(body);
126         var uri = $(".option").val();

```

```

127     var url = 'https://siga.ufrj.br/sira/repositorio-curriculo/${uri}';
128     return fetch(url);
129   })
130   .then( (resp) => resp.text() )
131   .then( parseBody )
132 }
133
134 module.exports = {
135   getLevels,
136   getCourses,
137   getAllCourses,
138   getCourse
139 }

```

8.3 sap.js

```

1  var fetch = require('node-fetch');
2  var cheerio = require('cheerio');
3
4  const URL = "http://sap.ufrj.br/";
5  const SEARCH_URL = `${URL}pesquisarCritR.asp`;
6  const RESULTS_URL = `${URL}pesquisarR.asp`;
7
8  function search(interessado){
9    var cookies = {};
10
11    return (
12      fetch(URL)
13      .then((resp) => {
14        cookies = Object.assign(cookies, extractNewCookies(resp));
15
16        var search_str = `op=P&numero=&interessado=${interessado}&unidade=&documentos`;
17        var headers = { Cookie: cookiesToString(cookies), "Content-Type": "application/json" };
18        return fetch(SEARCH_URL, { method: 'POST', body: search_str, headers: headers });
19      })
20      .then((resp) => {
21
22        cookies = Object.assign(cookies, extractNewCookies(resp));

```

```

23
24     return resp.text();
25
26 }).then(function(body){
27     var headers = { Cookie: cookiesToString(cookies), "Content-Type": "applicati
28
29     var body = "pagina=1&total=3&ordem=N";
30
31     return fetch(RESULTS_URL, { method: 'POST', body, headers });
32 })
33 .then((resp) => resp.text())
34 .then((body) => {
35     var $ = cheerio.load(body);
36     var table = $("table").get(1);
37     var rows = $("tr", table).toArray();
38
39     rows.shift();
40     rows.shift();
41
42     var results = [];
43     for(var i = 0; i < rows.length/4; i++){
44         var name = $("td[colspan=2]", rows[i*4]).text();
45         var code = $("td a", rows[i*4]).text();
46         var place = $($("td", rows[i*4+1]).get(1)).text();
47         var subject = $($("td", rows[i*4+2]).get(1)).text();
48         results.push({name, code, place, subject});
49     }
50
51     return results;
52 })
53 );
54
55 }
56
57 // Helper functions to work with cookies
58 function cookiesToString(cookies){
59     return Object.
60         getOwnPropertyNames(cookies).

```

```

61     map((name) => `${name}=${cookies[name]}`).
62     join("; ");
63 }
64
65 function extractNewCookies(resp){
66     return resp.
67         headers.
68         getAll("set-cookie").
69         join("; ").
70         split(/; ?/).
71         map((x) => x.split("=")).
72         filter((x) => x[0] && x[0].length > 0 && x[1] && x[1].length > 0).
73         reduce((dict,x) => { dict[x[0]] = x[1]; return dict; }, {});
74 }
75
76 module.exports = { search };

```

8.4 documents.js

```

1  var denodeify = require('denodeify');
2  var exec = denodeify(require('child_process').exec, function(err, stdout, stderr)
3      return [err, stdout];
4  });
5
6  var cridRegex = /(\d+)\s+([A-Z0-9]+\s+(([^ ]+ ?)+)\s+(\d+)\s+(\d+\.\d+)\s+(.*)/;
7
8  function getCrid(username, password){
9      // Since siga changed, we can no longer use the old script
10     // In a normal scenario we would use the following code:
11     // return exec('./scripts/get_documents crid $username $password')
12
13     return exec('pdftotext -layout ../crid.pdf -')
14         .then(function(text){
15             var lines = text.split("\n")
16             while(!lines[0].match(/^Controle/)){lines.shift()}
17             lines.shift();
18             return lines
19                 .filter(function(line){return line.match(/^\d/)})

```

```

20     .map(function(line){ return line.match(cridRegex)})
21     .map(function(r){
22         return {
23             control: r[1],
24             code: r[2],
25             title: r[3],
26             hours: r[5],
27             credits: r[6],
28             situation: r[7]
29         }
30     });
31     });
32 }
33
34 module.exports = { getCrid };

```

8.5 get_documents.sh

```

1  #!/bin/sh
2
3  FNAME=$(mktemp)
4  casperjs get_documents.js $1 $2 $3 $FNAME
5  pdftotext -layout $FNAME -

```

8.6 get_documents.js

```

1  var casper = require('casper').create({
2      verbose: true,
3      logLevel: "debug",
4      pageSettings: {
5          loadImages: false
6      }
7  });
8
9  var documents = {
10     crid: 'j_id95',
11     comprovante: 'j_id78:0:j_id79',
12     historico: 'botaoHistorico',

```



```

13     boletim: 'botaoBoletim',
14     boa: 'j_id91'
15 }
16
17 phantom.cookiesEnabled = true
18
19 casper.start('https://gnosys.ufrj.br/Portal/home.seam');
20
21 casper.waitForSelector("#gnosys-login-form",function(){
22     this.echo("Logging in");
23
24     var username = casper.cli.raw.get(1);
25     var password = casper.cli.raw.get(2);
26     this.echo("Logging in as "+username+" / "+password);
27
28     this.fill("#gnosys-login-form", {
29         inputUsername: username,
30         inputPassword: password
31     }, true)
32 })
33
34 casper.waitForSelector(".gnosys-login-informacoes", function(){
35     this.echo("successfully logged in");
36 });
37
38 casper.thenOpen('https://gnosys.ufrj.br/Documentos/');
39
40 casper.waitForSelector(".documento.crid", function(){
41     this.echo("On documents page");
42     var document_name = casper.cli.raw.get(0);
43     this.echo("Document name: "+document_name);
44     var document_action = documents[document_name];
45     this.echo("Document action: " + documents[document_name]);
46
47     var resp = this.evaluate(function(doc_action){
48         var form = jQuery("#gnosys-decor-vis-seletor-matricula-form");
49         var reqData = form.serialize()+"&"+doc_action+"="+doc_action;
50         jQuery.post(form.attr('action'),reqData,function(){

```

```

51     window.myFinished = true;
52   });
53   return reqData;
54 },document_action);
55
56   this.echo("Req data: "+resp);
57 });
58
59 casper.waitFor(function check(){
60   return this.evaluate(function(){
61     return window.myFinished === true;
62   });
63 },function then(){
64   var cid = this.getCurrentUrl().split("cid=").reverse()[0].split("&")[0];
65   var fname = casper.cli.args[3];
66   var url = "https://gnosys.ufrj.br/Documentos/seam/docstore/document.seam?docId=1";
67   this.echo("CURRENT CID IS: "+cid);
68   this.echo("Saving file to: "+fname);
69   this.download(url, fname);
70 }, function timeout(){
71   this.echo("Taking too long");
72 }, 10000)
73
74 casper.run();

```

Notes

¹NodeJS: <https://nodejs.org/>

²Koa.JS: <http://koajs.com/>

³PhantomJS: <http://phantomjs.org/>

⁴CasperJS: <http://casperjs.org/>

⁵pdftotext: <http://linux.die.net/man/1/pdftotext>

⁶Repositório no GitHub: <https://github.com/bamorim/sigapi>