DANIEL LUCAS DOS SANTOS

FELIPE BENI

GIOVANI PISA

MARCIO MARIANO DE PAULA

SANDRO NASCIMENTO LIMA

ClassFinder: Sistema de Classificação de bactérias por redes neuro-fuzzy com base em características extraídas das seqüências de genes de 16s rRNA

Trabalho de conclusão do curso de Tecnologia em Sistemas de Informação do Setor Escola Técnica da Universidade Federal do Paraná

Orientadores: Roberto Tadeu Raittz e Dieval Guizelini

CURITIBA
2008

SUMÁRIO

ÍNDICE DE FIGURAS

RESUMO

Neste projeto foi desenvolvida a aplicação ClassFinder para classificação taxonômica de bactérias com base na seqüência do gene de 16S rRNA (16S rDNA). As freqüências de trigrams foram extraídas destas seqüências e utilizadas no treinamento de Redes Neurais *Fuzzy*. Foram treinadas 1338 redes neurais utilizando-se 5165 seqüências de estirpes tipo obtidas do *Ribosomal Database Project*. O sistema aceita como entrada um arquivo no formato fasta contendo uma ou mais seqüências, com um limite máximo de 10 seqüências de 16S rDNA para classificação on-line. Mais de 10 seqüências são processadas em *batch*, sendo o resultado enviado posteriormente via *email*. Foi Possível classificar estas seqüências corretamente através de níveis taxonômicos superiores: Filo e classe.

ABSTRACT

In this Project the ClassFinder System is presented. The system classifies 16S rDNA sequences in agreement with the bacterial taxonomy. The characteristics of the 16S were extracted using the trigram code. This vector of 64 positions of trigrams was used in the training of fuzzy neural network. 1338 trained neural nets were obtained. A set of 5165 sequences downloaded from the Ribosomal Database Project were used in the training of these neural nets. As input, the ClassFinder System accepts a Fasta format file, and there is a threshold of 10 sequences for on line classification (processing). If one has more than 10 sequences, the classification utilizes a batch system, and the results of the classification is sent by email. Sequences of 16S rDNA were correctly classified in superior taxonomic levels: Phylum and Class.

LISTA DE ABREVIATURAS

AJAX - Asynchronous Javascript And XML

API - Application Programming Interface

CGI - Common Gateway Interface

CSS - Cascading Style Sheets

EJB - Enterprise JavaBeans

FTP - File Transfer Protocol

HTTP - Hypertext Transfer Protocol

IDE - Integrated Development Environment

JAAS - Java Authentication and Authorization Service

JSF - JavaServer Faces

JSP - JavaServer Pages

JSTL - JavaServer Pages Standard Tag Library

MVC - Model-view-controller

PAM - Pluggable Authentication Modules

PHP - PHP: Hypertext Preprocessor

SGBD - Sistema Gerenciador de Banco de Dados

SQL - Structured Query Language

SSL - Secure Sockets Layer

SWT - Standard Widget Toolkit

URL - Uniform Resource Locator

XML - eXtensible Markup Language

1 INTRODUÇÃO

Há uma necessidade constante de classificar bactérias em muitos tipos de laboratórios, tais como Microbiologia, Biologia Molecular, Ecologia, Análises Clínicas, entre outros. Esta necessidade procura atender aos mais diversos objetivos, por exemplo, identificar uma contaminação em algum tipo de alimento por bactéria patogênica, analisar a diversidade bacteriana em determinado ambiente, caracterizar novas espécies de bactérias, para citar algumas aplicações. Atualmente, existem muitos projetos de análise da diversidade bacteriana e de metagenômica, que procuram identificar a diversidade bacteriana e buscar novas enzimas de interesse comercial independentemente de cultivo de bactérias, somada a novas tecnologias de seqüenciamento, como o piroseqüenciamento (http://www.454.com/), tem depositado em bancos de dados públicos uma quantidade enorme de seqüências. O número de seqüências de nucleotídeos depositadas no GenBank do *site* NCBI já passa de 80 gigabases (BENSON et al, 2008), sendo que boa parte destas seqüências é oriunda de projetos de metagenômica, fazendo com que nossas idéias originais de diversidade de bactérias sejam extrapoladas enormemente. Estimativas recentes sugerem que um grama de solo contenha de 2000 a 8.3 milhões de espécies de bactéria (Gans et al., 2005; Schloss e Handelsman, 2006), portanto os sistemas de classificação de seqüências bacterianas devem oferecer opções cada vez mais robustas e precisas para auxiliar os pesquisadores nas análises laboratoriais de classificação. Este projeto apresenta uma alternativa aos sistemas convencionais de classificação de seqüências, baseados em alinhamento de seqüências e em modelos probabilísticos, utilizando para classificar estas seqüências a rede neuro-fuzzy FAN 2002 (RAITTZ, 2002) para reconhecimento de padrões extraídos das seqüências 16S rDNA de bactérias.

2 REVISÃO BIBLIOGRÁFICA

2.1 Hierarquia taxonômica

Taxonomia é a ciência da classificação. Os organismos conhecidos estão enquadrados em um esquema taxonômico. Abaixo estão mostrados os 8 principais níveis taxonômicos, do superior ao inferior, mostrando como exemplo a classificação de dois organismos: a bactéria *E. Coli* e a espécie humana, *Homo sapiens*:

Taxon	Espécie Humana	Bactéria *E. coli*
Domínio	Eucaria	Eubactéria
Reino	Animalia	Monera
Filo	Chordata	Proteobacteria
Classe	Mammalia	Gammaproteobacteria
Ordem	Primatas	Enterobacteriales
Família	Hominidae	Enterobacteriaceae
Gênero	*Homo*	*Escherichia*
Espécie	*Homo sapiens*	*Escherichia coli*

2.2 Taxonomia e classificação de bactérias

A taxonomia é tradicionalmente dividida em três partes (COWAN (1968); STALEY & KRIEG (1984)):

1) Classificação: ordenação dos organismos em grupos taxonômicos com base na similaridade entre eles.
2) Nomenclatura: rotulagem das unidades definidas na classificação.
3) Identificação dos organismos desconhecidos: Determinação de quais organismos pertencem às unidades taxonômicas (genômicas) e ecológicas pré-existentes.

A espécie é a unidade básica da taxonomia bacteriana, definida como um grupo de estirpes, incluindo a estirpe tipo, que dividam 70% ou mais de hibridização

DNA-DNA. Uma espécie de bactéria é uma categoria que contém um grupo de indivíduos (estirpes/isolados), que dividam um alto grau de similaridade em muitos aspectos independentes (taxonomia polifásica (VANDAMME et AL, 1996). A principal característica a ser avaliada é a seqüência de nucleotídeos do gene de 16S rRNA, sendo consideradas espécies diferentes aquelas que apresentam um limite inferior a 97% de similaridade nesta seqüências (HAGSTRÖM, 2000).

Análises de seqüências do gene que codifica para o 16S rRNA têm sido usadas para inferir relações filogenéticas entre muitas espécies de bactérias, sendo que homologia entre seqüências de 16S rDNA é o critério mais utilizado para estimar relações filogenéticas entre elas. Uma vantagem desta abordagem é que seqüências de DNA e produtos gênicos podem ser comparados em um contexto evolucionário (sistemática molecular) (van BERKUN et al, 2000).

A evolução de determinados genes bacterianos processa-se numa taxa constante. A história evolutiva do gene de 16S rRNA aproxima-se da história evolutiva do genoma total, tornando-se desta forma, aceitável reconstruir relações evolucionárias entre bactérias a partir da divergência das seqüências entre genes de 16S rRNA (van BERKUN et al, 2000. A homologia de DNA-DNA e a análise da seqüência do gene 16S rDNA é atualmente a abordagem adotada como consenso para estabelecer limites entre espécies bacterianas (STACKENBRANDT E GOEBEL, 1994).

2.3 Genes de rRNA

Os genes de rRNA (16S, 23S e 5S) são tidos como o melhor alvo para se estudarem relações filogenéticas, pois estão presentes em todas as bactérias, arqueobactérias, são funcionalmente constantes e a maior parte desta seqüência é altamente conservada (WOESE, 1987; STACKENBRANDT & GOEBEL, 1994),

sendo estas as principais características atribuídas a um relógio ou cronômetro molecular.

Os genomas bacterianos são compostos, sendo possível dividí-los em duas partes distintas. O genoma básico consiste de genes essenciais (*housekeeping*) são carregados nos cromossomos e de organização estável e sofrem pouca influência de transferência lateral ou horizontal (processo pelo qual um organismo incorpora material genético de outro organismo sem que estes dois tenham necessariamente relação filogenética muito próxima) e são estáveis, o que significa que possuem uma taxa evolutiva conhecida, o que os torna bastante apropriados para análises evolutivas. O gene de 16S rRNA é um exemplo de gene *housekeeping*.

O genoma acessório que possui genes que trazem adaptações em determinadas circunstâncias, como, por exemplo, algum tipo de restrição nutricional, permitindo que a bactéria possa sintetizar um composto necessário à própria nutrição. Estes genes, muitos deles também presentes no genoma, são carregados por plasmídeos, ilhas, transposons ou fagos, portanto possuem grande influência de transferência lateral. (YOUNG, 2000; SVYANEN, 1994).

2.4 Bioinformática

Bioinformática é a utilização de métodos computacionais, estatísticos e matemáticos para a resolução de problemas biológicos utilizando seqüências de DNA e aminoácidos e informações relacionadas a estas moléculas (Fredj Tekaia, Instituto Pasteur).

Uma das principais aplicações da Bioinformática atualmente é a comparação entre seqüências de aminoácidos (proteínas) e ácidos nucléicos (DNA e RNA). Esta comparação pode ser realizada com vários objetivos e finalidades, por exemplo, a busca por seqüências similares em bancos de dados públicos (www.ncbi.nlm.nih.gov/;

http://www.ebi.ac.uk/embl/; http://www.ddbj.nig.ac.jp/). Uma prática comum em laboratórios de Biologia Molecular e Microbiologia é a classificação de um *espécimen* em um determinado *taxon* (domínio, reino, filo, classe, ordem, família, gênero e espécie) de acordo com a seqüência do gene de 16S rRNA. Esta comparação pode ser feita utilizando-se basicamente dois tipos de algoritmos, os algoritmos baseados em alinhamento (chamados convencionais neste trabalho) e algoritmos que não utilizam as técnicas convencionais de alinhamento.

2.5 Algoritmos baseados em alinhamento de seqüências

Podemos fazer a comparação de seqüências com base nos algoritmos de alinhamento global e local. Esta seqüência de interesse é comparada com outras, visando encontrar aquelas mais próximas, com base no algoritmo de alinhamento local (SMITH e WATERMAN, 1981) (Figura 1), cuja principal implementação é o programa Blast (http://blast.ncbi.nlm.nih.gov/Blast.cgi), ou no algoritmo de alinhamento global (Figura 2) (NEEDLEMAN e WUNSCH, 1970) sendo que a implementação mais conhecida e utilizada é a série de programas chamada Clustal (THOMPSON et AL, 1994).

Se há muitas comparações a serem feitas estes métodos convencionais podem consumir muito tempo e recurso de *hardware*. Na comparação de seqüências, a experiência do especialista em manipular os parâmetros dos métodos de comparação é determinante na qualidade dos resultados de alinhamentos. Na ausência desta experiência, os resultados podem ser parciais ou inválidos. Por exemplo, entre duas seqüências pode haver *gaps* (também chamados *indels*, resultado de um evento de inserção ou deleção), que representam, normalmente, inserções ou deleções entre elas em termos biológicos, sendo que a penalidade entre estes *gaps* devem ser alteradas visando obter um melhor *score* entre estas seqüências, portanto o tipo de seqüências

envolvidas na análise deve ser conhecida pelo pesquisador. Uma maneira visual de comparação de seqüências é a construção de árvores filogenéticas que mostram caminhos evolutivos e relações de proximidade entre estas seqüências, como, por exemplo, o agrupamento de seqüências.

FIGURA 1: EXEMPLO DE ALINHAMENTO GLOBAL

```
CLUSTAL 2.0.10 multiple sequence alignment


EEA03709.1|          --MSFNNAGAAVVASSLAG--RAAEYVRMSTEHQQYSTENQRDRIR---DYAARRGFEIV 53
YP_001045366.1|      --MRGREASRETGASQSDPDVPAVAYVRMSTDHQKYSTENQLDVIR---SYAAARGLQIL 55
YP_001951643.1|      ---MLQTGTAYTPSQIMSLRLKGAMYIRMSTELQVESPENQERAIR---TYAAQYGIEII 54
ABV96540.1|          MTSTLPDWLRPPATTAPKRTIRVAFLGRTSTEEQQDPTLSIPRQLTNS-ERALLPSMVIV 59
ACA59334.1|          ------------------MSGVAIIYIRVSTEEQSRKGYSLPDQLEKCTQKAEEMGACDA 42
                                        * **: *     .    :      *   .

EEA03709.1|          RTYADEGKS----------------GLRIDGRQALQNLISDVASGKADFSVILVYDVSRW 97
YP_001045366.1|      RIFEDSGRS----------------GLRLDGREALQNLMGEVQSGRADFKAILVYDVSRW 99
YP_001951643.1|      KAYADLGVS----------------GINTEKREQFQSLIDDVEQGRNGYNIVLYLDESRW 98
ABV96540.1|          AWFWDVESSRKELSQRGRGTAWQKFDIAVQRDGGLADMLDEATSPDRRFDVVICESIDRI 119
ACA59334.1|          VLCTDYAET----------------AAYLDRPGLQRALELVATG--RYAYFIALDVDRL 83
                        *   :                       :   : .      : .:  . .*

EEA03709.1|          GRFQDADESAYYEYICRRAGIQVAYCAEQFENDGS---PVSTIVKGVKRAMAGEYSRELS 154
YP_001045366.1|      GRFQDADEGAYHEHVCSRAGIRVHYCGEQFENDGS---IGSNLLKTVKRVMAGEYSRELS 156
YP_001951643.1|      GRFVDSREAEYHRMRLERKNVLCQSCEKPLTLTSN---IADRIMTLLRDESASDYCRQLS 155
ABV96540.1|          ARWT--HQGTKIEHDLELAGVPLLAADEPVMQSNSRKRAAQILLRRTKQGVAEWYMIEML 177
ACA59334.1|          ARDLG-DQLYVTEEIEKRARVEFVTHRRGDPGNPE-----DTLFYHIKGAFSQYERAKTR 137
                     .*     :    .       :      .      .    . :.   :   :     :
```

A figura acima mostra um exemplo de alinhamento global (mostrando parte do alinhamento) com cinco seqüências de proteína Recombinase. Estão representados os números de acesso no GenBank (EEA03709.1: Burkholderia sp., ABV96540.1: Salinispora arenicola, YP_001951643.1: Geobacter lovleyi, YP_001045366.1: Rhodobacter sphaeroides, ACA59334.1: Desulforudis audaxviator).

Figura 2: Exemplo de alinhamento local

```
>AM490617 Ochrobactrum anthropi
          DSM 7216
       Length = 1387

  Plus Strand HSPs:

 Score = 1200 (186.1 bits), Expect = 6.4e-48, P = 6.4e-48
 Identities = 240/240 (100%), Positives = 240/240 (100%), Strand = Plus / Plus

Query:      1 CAGGCTTAACACATGCAAGTCGAGCGCCCCGCAAGGGGAGCGGCAGACGGGTGAGTAACG 60
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct:      1 CAGGCTTAACACATGCAAGTCGAGCGCCCCGCAAGGGGAGCGGCAGACGGGTGAGTAACG 60

Query:     61 CGTGGGAATCTACCTTTTGCTACGGAATAACTCAGGGAAACTTGTGCTAATACCGTATGT 120
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct:     61 CGTGGGAATCTACCTTTTGCTACGGAATAACTCAGGGAAACTTGTGCTAATACCGTATGT 120

Query:    121 GCCCTTTTGGGGAAAGATTTATCGGCAAAGGATGAGCCCGCGTTGGATTAGCTAGTTGGT 180
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct:    121 GCCCTTTTGGGGAAAGATTTATCGGCAAAGGATGAGCCCGCGTTGGATTAGCTAGTTGGT 180
```

A figura acima mostra um exemplo de alinhamento local (mostrando parte do alinhamento) com uma seqüência de consulta (query) 16S rDNA de Ochrobactrum thiophenivorans e o primeiro *hit* (seqüência mais similar) do banco, que é a própria seqüência de Ochrobactrum thiophenivorans. Está representado o número de acesso no GenBank (AM490617: Ochrobactrum thiophenivorans).

2.6 Aprendizagem de máquina, Reconhecimento de padrões e Inteligência artificial

Inteligência Artificial (IA) é uma ciência bastante abrangente, que inclui muitas subáreas, tais como lógica, probabilidade e matemática contínua: percepção, raciocínio, aprendizagem. Um dos primeiros conceitos sobre inteligência artificial está exposto no artigo "*Computing Machinery and Intelligence*" do matemático inglês Alan Turing (1950). Nele, Turing sugere que muitas das tarefas realizadas por um ser humano poderiam ser transferidas às máquinas, com regras pré-fixadas e sem prerrogativas de desviar-se sequer de um detalhe. A máquina, chamada por ele de *human computer*, ofereceria as vantagens de resultados mais rápidos e as regras poderiam ser modificadas de acordo com o trabalho a ser realizado.

2.6.1 Reconhecimento de Padrões

Ao se deparar com objetos ou situações desconhecidos, os seres humanos realizam comparações com acontecimentos de sua vivencia, associando sua experiência a fim de compreender, utilizar ou evoluir através da descoberta. O cérebro é capaz de realizar muitos processamentos paralelos, sendo considerada a máquina mais eficaz ao que se possa referir em reconhecimento de padrão (RP). Reconhecimento de Padrões é a ciência que tem como principal interesse a descrição ou classificação (reconhecimento) de medidas. A estrutura básica de um RP consiste em um sensor, um algoritmo para extração das características e filtragem de dados desnecessários e outro algoritmo para classificação (modelos estáticos) ou descrição (modelos sintáticos), concluindo a qual categoria pertence o determinado objeto estudado (SCHALKOFF, 1992). Obviamente, para usufruir das técnicas de RP e chegar a um resultado satisfatório, tanto usuários como desenvolvedores devem ter um conhecimento prévio do modelo de estudo proposto.

2.6.2 Inteligência Artificial na Bioinformática

O classificador bayesiano do RDP II, Ribosomal Database Project II (http://rdp.cme.msu.edu/) é o exemplo de aplicação baseada em IA mais conhecida para a classificação de bactérias com base em seqüências de 16S rDNA. No *site* do RDP são disponibilizadas as seqüências de 16S de virtualmente todas as espécies de bactérias descritas atualmente, com constantes atualizações. O algoritmo de classificação do RDP é o naïve Bayesian rRNA classifier, cujo treinamento foi feito utilizando-se seqüências de bactéria e arqueobactéria, agrupadas em 880 gêneros (WANG et al, 2007).

Outro exemplo da utilização da inteligência artificial em Bioinformática foi o emprego de redes neurais artificiais (back propagation e counter propagation) na classificação de seqüências de 16S rDNA e de famílias de proteínas (WU et al, 1994, 1995). O sistema de codificação utilizado para o treinamento foi o n-gram (http://www.w3.org/TR/ngram-spec/). Foi utilizada nestes dois exemplos a função de hashing n-gram (WU et al, 1994). O modelo de codificação baseia-se em extrair a freqüência de palavras n-gram de uma seqüencia biológica, transformando-a em um vetor numérico utilizado para treinamento das redes. Para a classificação das seqüências do 16S rDNA utilizou-se as freqüências de pentagrams (oligômeros ("palavras") de 5 caracteres do alfabeto de DNA = {a, c, g, t}), entre outras variações, para a classificação, com resultados superiores ao sistema de ranking de similaridade, que é uma razão entre a freqüência de oligômeros idênticos entre duas seqüências dividida pela freqüência de oligômeros únicos em uma das seqüências.

Este projeto de classificação de seqüências de 16S rDNA (WU e colaboradores) não parece ter sido continuado, pois não há registro de publicação mais recente deste tipo de abordagem pelo grupo de Cathy Wu (http://pir.georgetown.edu/pirwww/aboutpir/wubio.shtml).

Esta foi a inspiração inicial para desenvolvermos um sistema de classificação de seqüências com a utilização de redes neurais.

3 JUSTIFICATIVA

A classificação de bactérias é uma necessidade presente em muitos tipos de projetos que vão de análise da diversidade bacteriana em vários tipos de ambientes a análises laboratoriais (bactérias causadoras de infecção em humanos e animais e bactérias contaminantes de alimentos). A classificação baseada na seqüencia de DNA de moléculas conservadas entre os seres vivos e que represente sua história evolutiva, tal como o gene de 16S rRNA, é menos sujeita a variações, e permite o armazenamento de dados em bancos. Diariamente um número muito grande de novas seqüências é depositado em bancos de dados públicos, fazendo com que haja a necessidade de revisão constante de espécies descritas e criação de novos *taxa*.

4 OBJETIVOS

Este projeto pretende desenvolver um sistema classificador de seqüências de 16S rDNA de bactérias com base em códigos trigram, sendo uma alternativa viável aos softwares de classificação disponíveis atualmente.

5 MATERIAL E MÉTODOS

5.1 FAN – Free Associative Neurons

Desenvolvida em 1997, e aperfeiçoada em 2002, pelo Dr. Professor Roberto Tadeu Raittz, o sistema FAN é utilizado no reconhecimento de padrões através de redes neuronais difusas ou *neuro-fuzzy*.

Resumidamente, dado um universo de discurso, os algoritmos empregados no sistema realizam reconhecimento de padrão através de quatro módulos: extração de características, treinamento, testes e aplicação

A utilização desta ferramenta é voltada ao desenvolvimento de modelos computacionais com finalidade de resolver problemas extensos e complexos. Os primeiros trabalhos que utilizaram a *Free Associative Neurons* foram: Iris de Fisher, Reconhecimento de Cromossomos (Grupo de Denver), Problema do XOR, Mapa FAN (Dandolini 2000) e Construção de Agente para Supervisão de Alunos em Jogos de Empresas. Os ótimos resultados obtidos nestes casos, motivaram pesquisadores a utilizarem a rede, o que inevitavelmente a fez expandir, em suas aplicações, sobre as áreas mais diversas do conhecimento, encorajando pesquisadores a buscar, através deste conceito, modelos de pesquisas de maior acuidade e rapidez.

5.2 PostgreSQL

O PostgreSQL é um SGBD (Sistema Gerenciador de Banco de Dados) objeto-relacional de código aberto, com mais de 15 anos de desenvolvimento. É extremamente robusto e confiável, além de ser extremamente flexível e rico em recursos. Ele é considerado objeto-relacional por implementar, além das características de um SGBD relacional, algumas características de orientação a objetos, como herança

e tipos personalizados. A equipe de desenvolvimento do PostgreSQL sempre teve uma grande preocupação em manter a compatibilidade com os padrões SQL92/SQL99.

O código do POSTGRES foi aproveitado em um produto comercializado pela Illustra Information Technologies (posteriormente incorporada à Informix, que agora pertence à IBM). Em 1996 o nome Postgres95 tornou-se inadequado, o projeto foi rebatizado "PostgreSQL", para enfatizar a relação do POSTGRES original com a linguagem SQL.

O Grupo Global de Desenvolvimento do PostgreSQL é formado essencialmente por empresas especializadas em PostgreSQL, empresas usuárias do sistema, além dos pesquisadores acadêmicos e programadores independentes. Além da programação, essa comunidade é responsável pela documentação, tradução, criação de ferramentas de modelagem e gerenciamento, e elaboração de extensões e acessórios. Pela riqueza de recursos e conformidade com os padrões, o PostgreSQL é um SGBD muito adequado para o estudo do modelo relacional, além de ser uma ótima opção para empresas implementarem soluções de alta confiabilidade sem altos custos de licenciamento. É um programa distribuído sob a licença BSD, o que torna o seu código fonte disponível e o seu uso livre para aplicações comerciais ou não.

(http://www.postgresql.org.br/).

5.3 NetBeans

O IDE NetBeans é um ambiente de desenvolvimento integrado (IDE), modular e baseado em padrões, escrito na linguagem de programação Java. O projeto NetBeans consiste em um IDE de código-fonte aberto e em uma plataforma de aplicativo, utilizado como uma estrutura genérica para construir qualquer tipo de aplicativo. (http://www.netbeans.org/community/releases/60/index_pt_BR.html). NetBeans IDE é um conjunto de bibliotecas, módulos e APIs, formando um ambiente

integrado de desenvolvimento visual possibilitando ao desenvolvedor compilar, depurar e implantar suas aplicações.

Podemos destacar os seguintes recursos:

- Depurador e compilador de programas;
- Auto-completar avançado, depurador de erros, depurador de Servlets;
- Suporta linguagens Java, C, C++, entre outras;
- Suporte à XML e HTML, JSP, JSTL, Servlets, etc..;
- Recursos para desenvolvimento EJBs, Web Services;
- Total suporte ao ANT e TOMCAT integrado na IDE;
- Http Monitor para Monitoramento de aplicações WEB;
- Refatoração básica de código Java;
- Suporte a *Database*, *Data view*, *Connection wizard*;
- É um produto open source, 100% Java e possui vários módulos de expansão (modules)

(www.netbeans.org/)

5.4 Eclipse (IDE)

Eclipse é uma IDE de código aberto para a construção de programas de computador. O projeto Eclipse foi iniciado na IBM que desenvolveu a primeira versão do produto e doou-o como software livre para a comunidade. O gasto inicial da IBM no produto foi de mais de 40 milhões de dólares. Hoje o Eclipse é a IDE Java mais utilizada no mundo. Possui como características marcantes o uso da SWT e não do Swing como biblioteca gráfica, a forte orientação ao desenvolvimento baseado em plug-ins e o amplo suporte ao desenvolvedor com centenas de plug-ins que procuram atender as diferentes necessidades de diferentes programadores. O Eclipse, IDE inicialmente projetada para desenvolvimento em Java, hoje conta com poderosas e

inúmeras modificações e plug-ins para integrar ferramentas de desenvolvimento em várias linguagens, como PHP, C/C++, SQL, iPhone, todas as linguagens WEB e até algumas mais remotas, como COBOL. Por isto, tem se tornado indispensável para programadores e mesmo alguns web designers

(http://www.eclipse.org/)

5.5 JAVA

Java é uma linguagem de programação orientada a objetos. É desenvolvida pela companhia Sun Microsystems e hoje é uma linguagem com código aberto. Uma de suas principais características é sua portabilidade, o que a torna uma vantagem significativa para os desenvolvedores de software. Esta independência de plataforma juntamente com uma maior familiaridade da equipe com a linguagem foram as principais razões que a tornam interessante para o desenvolvimento do projeto ClassFinder.

Suas principais vantagens são:

plataforma portável;

Muitas opções de extensões, frameworks e ambientes de desenvolvimento;

Comunidade de desenvolvedores bastante ativa, que se mantém informada através de artigos, discussões, revistas, exemplos de códigos, etc.

(http://java.com/pt_BR/)

5.6 JAAS

O JAAS (*Java Authentication and Authorization Service*) é um conjunto de APIs que permite que as aplicações Java tenham um controle de autenticação e de acesso. O JAAS implementa uma versão Java do framework padrão *Pluggable*

Authentication Module (PAM), e suporta autorização baseada em usuário. Isso permite que a aplicação fique independente deste controle de segurança. Serve para controlar permissões de vários tipos de recursos: arquivos, diretórios, conteúdos e URLs. O JAAS está em nível de servidor de aplicação e não de aplicação, ou seja, a autenticação é executada pelo servidor de aplicação, antes mesmo de acessar a aplicação.

Para as necessidades do projeto, utilizou-se JASS na a autenticação do *login* de usuários no *site* do ClassFinder.

(http://java.sun.com/javase/6/docs/technotes/guides/security/jaas/JAASRefGuide.html)

5.7 JSF

Podemos dizer que o JSF (*Java Server Faces*) é uma infra-estrutura sobre a qual podemos criar frameworks de visão, análogo ao MVC (*Model View Controller*). O JSF define uma estrutura e especifica todo o ciclo de uma requisição sendo formado por várias partes, que vêm com implementações *default*, permitindo a troca de cada uma delas.

(http://java.sun.com/javaee/javaserverfaces/)

5.8 RichFaces

É uma biblioteca de componentes para aplicações web usadas no JSF originalmente desenvolvido por Exadel (http://exadel.com/web/portal/fiji) e atualmente é mantido por JBoss (http://www.jboss.org).

Possui um grande suporte a *skins* que deixam suas páginas padronizadas e vários componentes padrão para uma fácil implementação.

(http://www.jboss.org/jbossrichfaces/)

5.9 Ajax4JSF

É um framework que permite que seja usado AJAX (*Asynchronous* Javascrip*t And* XML) em páginas JSF sem o uso de Javascript.

Assim como o RichFaces (citado acima), atualmente é mantido por JBoss (http://www.jboss.org)

Esta tecnologia é usada para deixar as páginas mais dinâmicas, com mais interações com o usuário. No sistema ClassFinder, está sendo utilizada na página de resposta: como o processamento da rede não ocorre de forma rápida (em média) 1,5s por pesquisa, esta espera pode ser cansativa, Ajax4JSF está sendo utilizado para mostrar os resultados parciais obtidos com intervalos curtos, até o fim do processamento.

5.10 TomCat

É um servidor web, programa responsável por disponibilizar páginas, fotos e demais objetos ao navegador cliente. Opera recebendo dados do cliente, processando e enviando o resultado para que o cliente possa tomar a ação desejada como em aplicações CGI's, banco de dados web, preenchimento de formulários, etc.

Suas principais características são:

- Suporte a scripts CGI (*Common Gateway Interface*);
- Suporte a autorização de acesso;
- Autenticação de usuários e senhas;
- Permite exibição da página web no idioma requisitado pelo cliente
- Suporte a tipos mime;
- Personalização de *logs*;
- Mensagens de erro;

- Suporte a IP *virtual hosting*;
- Suporte a servidor Proxy FTP e HTTP, com limite de acesso configuráveis;
- Criptografia via SSL, Certificados Digitais.

(http://tomcat.apache.org/)

5.11 N-gram

Um n-gram é uma subseqüência de n itens de uma dada seqüência. Um n-gram de tamanho 1 é referido como um "unigram", tamanho 2 é um "bigram" (ou, menos comumente, um "digram"), tamanho 3 é um trigram, e tamanho 4 ou mais é simplesmente chamado um "n-gram".

(http://www.w3.org/TR/ngram-spec/).

5.12 Função n-gram utilizada no sistema de ClassFinder

Abaixo está representada a codificação trigram em seqüências de DNA utilizadas neste sistema (códigos-fonte das classes codificadoras de seqüências de 16S rDNA estão no CD que acompanha o trabalho):

Alfabeto: {A, C, G, T}

Total de códigos = m ^ n

n: grau de extração escolhido (2-gram, 3-gram, 4-gram, ... , n-gram);

m: número de símbolos diferentes do alfabeto

Codificação usando a seqüência de pares de base em DNA:

3-gram:

4 ^ 3 possíveis combinações = 64

Cada código contém a freqüência do respectivo 3-gram na string original de seqüência de pares de bases;

O padrão de entrada para rede neural terá 64 variáveis de entrada.

e. g. códigos: {AAA, AAC, AAG, ... , TTT} (A: adenina, C: citosina, G: guanina e T: timina).

6 RESULTADOS

6.1 Primeiros testes realizados utilizando-se árvores de decisão

Inicialmente três testes de classificação foram realizados com as seqüências de 16S rDNA de vinte e sete espécies de bactérias pertencentes ao filo Thermotogae. Foi testado monogram sozinhos, bigram sozionho e mono e bigram juntos. O método de codificação das seqüências escolhido foram os mono e bigram. No caso dos monogram foi feita uma contagem simples dos nucleotídeos (A, C, G, T) que integram a seqüência de cada uma das espécies e os bigram foi feita uma contagem do dinucleotídeos (AA, AC, AG, ... , TT). No anexo A estão representadas as 27 espécies pertencentes ao Filo Thermotogae utilizadas nestes experimentos iniciais.

6.1.1 Quatro atributos

Atributos utilizados: contagem simples de nucleotídeos (A C G T).

Algoritmo de árvore de decisão utilizado: c4.5 (http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/tutorial.html)

```
C4.5 [release 8] decision tree generator        Sun Apr 27 09:43:20 2008
----------------------------------------
    Options:
        File stem <especie>
Read 27 cases (4 attributes) from especie.data
Decision Tree:

G <= 510 :
|   T <= 258 :
|   |   T <= 248 : Fervidobacterium_nodosum
|   |   T > 248 :
|   |   |   A <= 349 : Fervidobacterium_gondwanense
|   |   |   A > 349 : Marinitoga_camini
|   T > 258 :
|   |   T > 266 : Petrotoga_mexicana
```

```
|   |   T <= 266 :
|   |   |   A <= 393 : Marinitoga_hydrogenitolerans
|   |   |   A > 393 : Geotoga_petraea
G > 510 :
|   A <= 327 :
|   |   G <= 539 : Fervidobacterium_islandicum
|   |   G > 539 : Thermotoga_maritima
continua

Continuação

|   A > 327 :
|   |   A <= 343 : Thermosipho_japonicus
|   |   A > 343 :
|   |   |   A <= 346 : Thermosipho_atlanticus
|   |   |   A > 346 : Geotoga_subterranea

Tree saved

Evaluation on training data (27 items):

         Before Pruning           After Pruning
         ----------------   ---------------------------
         Size      Errors   Size      Errors   Estimate

           21   16(59.3%)     21   16(59.3%)    (91.8%)   <<
```

6.1.2 Dezesseis atributos

Atributos utilizados: contagem dos dinucleotídeos (AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT).

```
C4.5 [release 8] decision tree generator        Sun Apr 27 09:56:35 2008
----------------------------------------

    Options:
        File stem <especie2>

Read 27 cases (16 attributes) from especie2.data

Decision Tree:

TT <= 36 :
|   TT <= 32 :
|   |   AT <= 39 : Fervidobacterium_islandicum
|   |   AT > 39 : Geotoga_petraea
|   TT > 32 :
|   |   TT <= 33 : Fervidobacterium_nodosum
|   |   TT > 33 :
|   |   |   AA <= 98 : Thermosipho_africanus
|   |   |   AA > 98 : Marinitoga_camini
TT > 36 :
|   TG <= 96 :
|   |   TC <= 53 : Marinitoga_hydrogenitolerans
|   |   TC > 53 : Thermotoga_hypogea
|   TG > 96 :
|   |   TG <= 98 :
|   |   |   AA <= 95 : Thermosipho_atlanticus
```

```
|   |   |   AA > 95 : Petrotoga_olearia
|   |   TG > 98 :
|   |   |   AA <= 110 : Fervidobacterium_gondwanense
|   |   |   AA > 110 : Geotoga_subterranea
Tree saved


Evaluation on training data (27 items):

         Before Pruning           After Pruning
         ----------------   ---------------------------
         Size      Errors   Size      Errors   Estimate

           21   16(59.3%)     21   16(59.3%)    (91.8%)   <<
```

6.1.3) Vinte atributos

Atributos utilizados: contagem dos nucleotídeos simples e dinucleotídeos (A, C, G, T, AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT).

```
C4.5 [release 8] decision tree generator        Sun Apr 27 10:00:14 2008
----------------------------------------
    Options:
        File stem <especie3>
Read 27 cases (20 attributes) from especie3.data
Decision Tree:
TT <= 36 :
|   TT <= 32 :
|   |   AT <= 39 : Fervidobacterium_islandicum
|   |   AT > 39 : Geotoga_petraea
|   TT > 32 :
|   |   TT <= 33 : Fervidobacterium_nodosum
|   |   TT > 33 :
|   |   |   A <= 357 : Thermosipho_africanus
|   |   |   A > 357 : Marinitoga_camini
TT > 36 :
|   TG <= 96 :
|   |   TC <= 53 : Marinitoga_hydrogenitolerans
|   |   TC > 53 : Thermotoga_hypogea
|   TG > 96 :
|   |   TG <= 98 :
|   |   |   A <= 345 : Thermosipho_atlanticus
|   |   |   A > 345 : Petrotoga_olearia
|   |   TG > 98 :
|   |   |   A <= 373 : Fervidobacterium_gondwanense
|   |   |   A > 373 : Geotoga_subterranea

Tree saved

Evaluation on training data (27 items):

         Before Pruning           After Pruning
         ----------------   ---------------------------
         Size      Errors   Size      Errors   Estimate
           21   16(59.3%)     21   16(59.3%)    (91.8%)   <<
```

Os resultados indicados anteriormente sugerem a impossibilidade de se utilizar árvores de decisão para a correta classificação de bactérias baseada em códigos mono e bigram seqüências de 16S rDNA, haja vista a alta ocorrência de erros (59.3%) na classificação obtidos com os três métodos de codificação.

6.2 Testes iniciais com redes neurais

Nesta abordagem inicial utilizamos 5165 seqüências de 16S rDNA de estirpes tipo depositadas no banco do RDP II para treinamento das redes neurais. Além da opção de se baixar seqüências de estirpe tipo (T), optou-se por baixar as seqüências de estirpes tipo com boa qualidade e de tamanho maior de 1200 pares de base. Todas as seqüências deste conjunto foram codificadas de acordo com as freqüências de trigram .

Na primeira tentativa um arquivo de treinamento único contendo o código de todas estas seqüências foi gerado. Sendo assim, cada uma destas seqüências foi considerada uma classe única. Com esta abordagem não foi possível obter êxito na classificação, pois houve um déficit considerável de memória para carregar este arquivo no programa EasyFAN (GARRET et al, 2006). Ficou claro que era preciso modularizar o arquivo em subunidades tratáveis taxonomicamente. Este mesmo arquivo de seqüências foi utilizado nesta abordagem posterior.

6.3 Classificação de seqüências de 16S rDNA com redes neurais artificiais

Nesta abordagem inicial utilizamos 5165 seqüências de 16S rDNA de estirpes tipo depositadas no banco do RDP II para treinamento das redes neurais. Além da opção de se baixar seqüências de estirpe tipo, optamos por baixar as seqüências com boa qualidade e de tamanho maior de 1200 pares de base. Todas as seqüências

deste conjunto foram codificadas de acordo com as freqüências trigram .

Estes são os filos a que pertencem todas as seqüências codificadas (entre parênteses está indicado o número de seqüências de cada um deles): Aquificae(18), Thermotogae(27), Thermodesulfobacteria(4), Deinococcus-Thermus(4), Chrysiogenetes(1), Chloroflexi(8), Thermomicrobia(1), Nitrospira(5), Deferribacteres(1), Cyanobacteria(14), Chlorobi(9), Proteobacteria(1925), Firmicutes(1178), Actinobacteria(1285), Planctomycetes(7), Chlamydiae(14), Spirochaetes(53), Fibrobacteres(2), Bacteroidetes(355), Fusobacteria(34), Verrucomicrobia(1), Dictyoglomi(1), Gemmatimonadetes(1), Lentisphaerae(2), Tenericutes(16).

A figura 3 traz o exemplo de uma das seqüências utilizadas no treinamento das redes neurais e seu correspondente código trigram.

FIGURA 3: ARQUIVO DE EXEMPLO MOSTRANDO A PRIMEIRA DAS SEQÜÊNCIAS CODIFICADAS E O CÓDIGO TRIGRAM CORRESPONDENTE.

```
>gi|37222674|gb|M83548.2|AQF16SRRN Aquifex pyrophilus 16S ribosomal RNA gene, partial sequence
TTCCCTGAAGAGTTTGATCCTGGCTCAGCGCGAACGCTGGCGGCGTGCCTAACACATGCAAGTCGTGCGC
AGGCTCGCTCCCTCTGGGAGCGGGTGCTGAGCGGCAAACGGGTGAGTAACACGTGGGTAACCTACCCCCA
GGAGGGGGATAACCCCGGGAAACCGGGGCTAATACCCCATAAAGCCGCCCGCCACTAAGGCGAGGCGGCC
AAAGGGGGCCTCTGGGCTCTGCCCAAGCTCCCGCCTGGGGATGGGCCCGCGGCCCATCAGGTAGTTGGTG
GGGTAACGGCCCACCAAGCCTATGACGGGTAGCCGGCCTGAGAGGGTGGCCGGCCACAGCGGGACTGAGA
CACGGCCCGCACCCCTACGGGGGGCAGCAGTGGGGAATCGTGGGCAATGGGCGAAAGCCTGACCCCGCGA
CGCCGCGTGGGGGAAGAAGCCCTGCGGGGTGTAAACCCCTGTCGGGGGGGACGAAGGGACTGTGGGTTAA
TAGCCCACAGTCTTGACGGTACCCCCAGAGGAAGGGACGGCTAACTACGTGCCAGCAGCCGCGGTAATAC
GTAGGTCCCGAGCGTTGCGCGAAGTCACTGGGCGTAAAGCGTCCGCAGCCGGTCGGGTAAGCGGGATGTC
AAAGCCCACGGCTCAACCGTGGAATGGCATCCCGAACTGCCCGACTTGAGGCACGCCCGGGCAGGCGGAA
TTCCCGGGGTAGCGGTGAAATGCGTAGATCTCGGGAGGAACACCGAAGGGGAAGCCAGCCTGCTGGGGCT
GTCCTGACGGTCAGGGACGAAAGCCGGGGGAGCGAACCGGATTAGATACCCGGGTAGTCCCGGCCGTAAA
CCATGGGCGCTAGGGCTTGTCCCTTTGGGGCAGGCTCGCAGCTAACGCGTTAAGCGCCCCGCCTGGGGAG
TACGGGCGCAAGCCTGAAACTCAAAGGAATTGGCGGGGGCCCGCACAACCGGTGGAGCGTCTGGTTCAAT
TCGATGCTAACCGAAGAACCTTACCCGGGCTTGACATGCCGGGGAGACTCCGCGAAAGCGGAGTTGTGGA
AGTCTCTGACTTCCCCCCGGCACAGGTGGTGCATGGCCGTCGTCAGCTCGTGTCGTGAGATGTTGGGTTA
AGTCCCGCAACGAGCGCAACCCCTGCCCCTAGTTGCTACCCCGAGAGGGGAGCACTCTAGGGGGACCGCC
GGCGATAAGCCGGAGGAAGGGGGGGATGACGTCAGGTCAGTATGCCCTTTATGCCCGGGGCCACACAGGC
GCTACAGTGGCCGGGACAATGGGAAGCGACCCCGCAAGGGGGAGCTAATCCCAGAAACCCGGTCATGGTG
CGGATTGGGGGCTGAAACTCGCCCCCATGAAGCCGGAATCGGTAGTAACGGGGTATCAGCGATGTCCCCG
TGAATACGTTCTCGGGCCTTGCACACACCGCCCGTCACGCCACGGAAGTCGGTCCGGCCGGAAGTCCCCG
AGCTAACCGGCCCTTTTTGGGCCGGGGGCAGGGGCCGATGGCCGGGCCGGCGACTGGGGCGAAGTCGTAA
CAAGGTAGCCGTAGGGGAACCTGC
```

Codificação trigram:

AAA	AAC	AAG	AAT	ACA	ACC	ACG	ACT	AGA	AGC	AGG	AGT	ATA	ATC	ATG	ATT
16	28	32	14	17	27	25	11	11	37	28	18	8	8	19	5
CAA	CAC	CAG	CAT	CCA	CCC	CCG	CCT	CGA	CGC	CGG	CGT	CTA	CTC	CTG	CTT
16	21	25	9	16	60	54	22	23	31	55	25	16	13	25	8
GAA	GAC	GAG	GAT	GCA	GCC	GCG	GCT	GGA	GGC	GGG	GGT	GTA	GTC	GTG	GTT
34	20	25	13	24	50	40	20	39	48	101	29	23	26	21	10
TAA	TAC	TAG	TAT	TCA	TCC	TCG	TCT	TGA	TGC	TGG	TGT	TTA	TTC	TTG	TTT
24	12	13	4	14	15	15	9	19	18	32	8	5	4	13	5

É possível ver na figura anterior a seqüência em formato fasta da primeira seqüência do conjunto de seqüências utilizadas no treinamento das redes neurais. A codificação em trigram é mostrada logo em seguida.

De posse de todas as 5165 características extraídas em vetores de 64 posições, optou-se por treinar mais de uma rede com este conjunto, de acordo com a estrutura modelada no banco, que procura simular a taxonomia das bactérias (*Bergey's Taxonomic Outline* - http://dx.doi.org/10.1007/bergeysoutline200310). A hierarquia taxonômica abordada é a seguinte: Domínio, Filo, Classe, Ordem, Família, Gênero e espécie. A organização do Banco utilizado no sistema ClassFinder é baseada nesta hierarquia: os níveis taxonômicos que ocupam níveis superiores são chamados de nós pais e os níveis inferiores são chamados nós filhos. Exemplificando: todos os *taxa* pertencentes ao nível taxonômico Classe (e. g. Betaproteobacteria) são chamados filho em relação aos níveis taxonômicos que pertencem ao nível taxonômico Filo (e. g. Proteobacteria).

Resumidamente, quando uma seqüência é submetida à classificação no sistema, ela é convertida em um código trigram mantido em um vetor de 64

características como descrito anteriormente. Este vetor de freqüências é normalizado (para tratamento pela rede) e submetido à classificação pela primeira rede treinada. Esta rede classifica a seqüência de entrada em um dos Filos citados acima.

É assumido a priori, se o arquivo estiver em conformidade com o formato fasta e a regra do sistema (menor do que 2000 pb), que a seqüência pertença a uma bactéria, pois não há uma rede treinada neste nível taxonômico, ou seja, não há uma classificação prévia nos domínios Archea, Bactéria e Eucarioto.

7 DISCUSSÃO

Os testes realizados com árvores de decisão mostraram-se inviáveis para a classificação de bactérias com o tipo de codificação utilizada (mono e bigram). Ficou claro depois destes testes iniciais que outro tipo de abordagem era necessária para o sistema de classificação. Optamos por utilizar outro tipo de abordagem baseada em aprendizagem e máquina, as redes neurais artificiais, haja vista a disponibilidade de se utilizar a *Free Associative Neurons*, FAN, criada pelo Professor Roberto Raittz.

Os primeiros testes com redes neurais mostraram-se inviáveis sob o ponto de vista de utilização dos recursos de memória. Nesta etapa inicial tentou-se treinar uma única rede com as características de 5165 seqüências, extrapolando desta maneira a capacidade de manipulação de memória da aplicação EasyFAN (GARRET et al, 2006).

Posteriormente estas seqüências foram divididas em grupos menores (taxonômicos) para o treinamento de várias redes com um número de seqüências menor, possibilitando, desta maneira um tipo de treinamento em árvore.

A classificação de seqüências de 16S rDNA bacterianas utilizando redes neuro-fuzzy FAN2002 mostrou-se uma alternativa viável aos convencionais sistemas de classificação. Até o momento da publicação deste trabalho, as redes neurais

treinadas foram capazes de classificar corretamente as seqüências bacterianas nos níveis taxonômicos de filo e de classe.

8 CONCLUSÕES

O sistema de codificação n-gram (mono e bi) de seqüências 16S rDNA não mostrou ser o método mais eficiente para se classificar bactérias com a utilização em árvores de decisão.

Com a codificação trigram de seqüências de 16S, o sistema ClassFinder pôde classificar corretamente seqüências de bactérias em níveis taxonômicos mais superiores: filo e classe.

O sistema ClassFinder apresenta-se como uma aplicação viável ao problema de se classificar seqüências de 16S rDNA. Este sistema possibilita aos usuários um interface simples e intuitiva para a utilização, que lembra as aplicações de Bioinformática mais utilizadas atualmente, com o carregamento de seqüências em formato Fasta.

Apesar da interface do sistema ClassFinder ser simples e intuitiva, o processo interno de classificação das seqüências utiliza técnicas avançadas de aprendizagem de máquina, que são redes neurais artificias treinadas com um conjunto selecionado de seqüências. Mais de 1300 redes neurais foram treinadas, fazendo deste trabalho um esforço enorme de capacidade computacional e conjunto de desenvolvedores e especialistas.

9 REFERÊNCIAS BIBLIOGRÁFICAS

454 SEQUENCING

<www.454.com> Acesso em 27 de novembro de 2008.

ALTSCHUL, S.F.; MADDEN, T.L.; SCHÄFFER, A.A.; ZHANG, J.; ZHANG, Z.; MILLER, W.; LIPMAN, D.J. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research, Oxford, v. 25, n. 17, p. 3389-3402, 1997.

Ajax4JSF

< http://www.jboss.org/jbossrichfaces/ > Acesso em 8 de dezembro de 2008

BENSON, D. A.; KARSCH-MIZRACHI, I. ; LIPMAN, D. J.; OSTELL, J. ; WHEELER, D. L. GenBank. Nucleic Acids Research. 36(Database issue): D25–D30. 2008.

Bergey's Taxonomic outline

< http://dx.doi.org/10.1007/bergeysoutline200310 > Acesso em 27 de Abril de 2008

BRENNER, D., STALEY, J. E KRIEG, N. Bergey's Manual of Systematic Bacteriology (Springer, New York). 2000.

CATHY WU, Ph. D. professora do Departamento de Bioquímica, Biologia Molecular e Biologia Celular da Universidade de Georgetown, Washington, DC, USA

< http://pir.georgetown.edu/pirwww/aboutpir/wubio.shtml > Acesso em 4 de Dezembro de 2008

CLUSTALW

< http://www.ebi.ac.uk/clustalw/> Acesso em 27 de novembro de 2008.

COWAN, S.T.. A dictionary of microbial taxonomic usage. Edinburgh, Oliver & Boyd, 1968.

C4.5

< http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/tutorial.html> Acesso em dia 03 de dezembro de 2008

DANDOLINI, G. A. Um procedimento para avaliação da saúde financeira de pequenas empresas: estudo de um caso usando redes neuronais artificiais. Dissertação de Mestrado, Universidade Federal de Santa Catarina, Departamento de Engenharia de Produção, 1996.

DDBJ: DNA Data Bank of Japan

< http://www.ddbj.nig.ac.jp/> Acesso em 27 de novembro de 2008.

ECLIPSE

< http://www.eclipse.org/ > Acesso em 27 de novembro de 2008

The EMBL Nucleotide Sequence Database (EMBL-Bank)

< http://www.ebi.ac.uk/embl/> Acesso em 27 de novembro de 2008.

GANS, J., M. WOLINSKY, E J. DUNBAR. Computational improvements reveal great , bacterial diversity and high metal toxicity in soil. Science 309. 1387–1390. 2005.

GARRET, L. F.; IGNÁCIO, F. A.; KUSTER, C. W.; LENFERS, F. P.; ZOTTO, S. P. EasyFAN. Trabalho de conclusão de Curso de Tecnologia em Informática, Setor Escola Técnica, Ubuversidade Federal do Paraná, Curitiba. 2006

HAGSTRÖM, Å.; PINHASSI, J. E ZWEIFEL, U. L.. Biogeographical diversity among marine bacterioplankton. Aquatic Microbiology Ecology. 21:231-244, 2000.

JAAS

< http://java.sun.com/javase/6/docs/technotes/guides/security/jaas/JAASRefGuide.html > Acesso em 06 de Dezembro de 2008

JAVA

< http://java.com/pt_BR/ > Acesso em 06 de Dezembro de 2008

Java Server Faces

(http://java.sun.com/javaee/javaserverfaces/) Acesso em 06 de Dezembro de 2008

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology. Março. V. 48(3), p. 443-53. 1970.

NATIONAL CENTER FOR BIOTECHNOLOGY INFORMATION

< http://www.ncbi.nlm.nih.gov/> Acesso em: 05 de janeiro de 2006.

NETBEANS

< http://www.netbeans.org > Acesso em 27 de novembro de 2008.

PAUL, E. A. E CLARK, F. E. Soil microbiology and biochemistry. Academic Press, San Diego, 1989).

POSTGRESQL

< http://www.postgresql.org.br/ Introdução_e_histórico > Acesso em 27 de novembro de 2008

RAITTZ, R. T. FAN 2002: um modelo neuro-fuzzy para reconhecimento de padrões. Tese de doutorado. UFSC. 2002.

RDP - Ribosomal Database Project II

< http://rdp.cme.msu.edu/ > Acesso em 27 de novembro de 2008

RichFaces

< http://www.jboss.org/jbossrichfaces/ > Acesso em 8 de dezembro de 2008

SCHALKOFF, R. J. Pattern Recognition: Statistical, Structural and Neural Approches, John Wiley & Sons, Inc., 1992.

SMITH, T. F.; WATERMAN, M. S. Identification of Common Molecular Subsequences. Journal of Molecular Biology 147: 195-197. 1981

SCHLOSS, P. D. ; HANDELSMAN, J. Toward a census of bacteria in soil. PLoS Computational Biology 2(7). 2006

STACKENBRANDT, E.; GOEBEL, B.M. Taxonomic note: a place for DNA-DNA reassociation and 16S rRNA sequence analysis in the present species definition in bacteriology. International Journal of Systematic Bacteriology, v. 44, p. 846-849. 1994.

STOCHASTIC LANGUAGE MODELS (N-GRAM) SPECIFICATION

< http://www.w3.org/TR/ngram-spec/ >. Acesso em 27 de novembro de 2008

SVYANEN, M. Horizontal gene transfer: evidence and Possible Consequences. Annual Reviews Genetics. 28:237- 61. 1994.

THOMPSON, J. D.; HIGGINS, D. G. E GIBSON, T. J. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Research. November 11; 22(22): 4673–4680. 1994.

TOMCAT

(http://tomcat.apache.org/) Acesso em 28 de novembro de 2008

TURING, A.M.Computing machinery and intelligence. Mind, 59, 433-460. 1950.

van BERKUN, P.; FUHRMANN, J. J.; EARDLY, B.D. Phylogeny of Rhizobia. In: PEDROSA, F.O.; HUNGRIA, M.; YATES, G.; NEWTON, W.E. NITROGEN FIXATION: FROM MOLECULES TO CROP PRODUCTIVITY. Foz do Iguaçu: Kluwer Academic Publishers, p 3-8, 2000.

VANDAMME, P.; POT, B.; GILLS, M; DEVOS , P.; KERSTERS, K.; SWINGS, J. 1996. Polyphasic taxonomy, a Consensus Approach to Bacterial Systematics. Microbiology Reviews, Washinton, v.60, n.2, p. 407-437.

YOUNG, J. P. W. Molecular evolution in Diazotrophs: do Genes Agree? In: PEDROSA, F.O.; HUNGRIA, M.; YATES, G.; NEWTON, W.E. NITROGEN FIXATION: FROM MOLECULES TO CROP PRODUCTIVITY. Foz do Iguaçu: Kluwer Academic Publishers, 2000.

WANG, Q.; GARRITY, G. M.; TIEDJE, J. M. E COLE, J.R. Naïve Bayesian Classifier for Rapid Assignment of rRNA Sequences into the New Bacterial Taxonomy. Applied and Environmrental Microbiology.V. 73(16), p.5261-7. 2007.

WOESE, C. R. Bacterial Evolution. Microbiological Reviews, Washington, v. 51, n. 2, p 221-271, 1987.

WU, C. H., BERRY, M., SHIVAKUMAR, S. AND MCLARTY, J. Neural networks for full-scale protein sequence classification: Sequence encoding with singular value decomposition. Machine Learning, 21, 177-193. 1995.

WU, C. H. AND SHIVAKUMAR, S. Back-propagation and counter-propagation neural networks for phylogenetic classification of ribosomal RNA sequences. Nucleic Acids Research, 22, 4291-4299. 1994.

WU, C. H., WHITSON, G., MCLARTY, J., ERMONGKONCHAI, A. AND CHANG, T. Protein classification artificial neural system. Protein Science, 1, 667-677. 1992.

ZUCKERKANDL, E.; PAULING, L.. Molecules as documents of evolutionary history. Journal of Theorical Biology, v. 8, p. 357-366, 1965.

10 ANEXO A

Números de acesso (cabeçalhos FASTA) das seqüências de 16S rDNA de espécies Thermotogae utilizadas nos primeiros ensaios. Estas seqüências, constituídas somente por estirpes tipo, foram baixadas do *site* do RDP II (http://rdp.cme.msu.edu/)

>S000004399 Thermotoga elfii (T); SERB; X80790

>S000010700 Thermotoga petrophila (T); RKU-1; AB027016

>S000382805 Thermotoga naphthophila (T); RKU-10; AB027017

>S000383207 Thermotoga neapolitana (T); DSM 4359; AB039768

>S000383208 Thermotoga thermarum (T); DSM 5069; AB039769

>S000392200 Thermotoga lettingae (T); TMO; AF355615

>S000436057 Thermotoga maritima (T); M21774

>S000437280 Thermotoga subterranea (T); SL1; U22664

>S000438744 Thermotoga hypogea (T); SEBR 7054; U89768

>S000001985 Fervidobacterium gondwanense (T); AB39 (=ACM ?); Z49117

>S000007611 Fervidobacterium nodosum (T); M59177

>S000436525 Fervidobacterium islandicum (T); H-21; M59176

>S000414371 Geotoga petraea (T); T5; L10658

>S000414372 Geotoga subterranea (T); CC-1; L10659

>S000004844 Marinitoga camini (T); MV1075; AJ250439

>S000391620 Marinitoga piezophila (T); KA3; AF326121

>S000539628 Marinitoga hydrogenitolerans (T); type strain:AT1271; AP1; AJ786363

>S000002133 Petrotoga sibirica (T); SL25T; AJ311702

>S000010535 Petrotoga olearia (T); SL24T; AJ311703

>S000015010 Petrotoga mobilis (T); SJ95T; Y15479

>S000396590 Petrotoga mexicana (T); Met-12; DSM 14811; CIP 107371; AY125964

>S000414370 Petrotoga miotherma (T); 42-6; L10657

>S000002167 Thermosipho geolei (T); DSM 13256, type strain; AJ272022

>S000010674 Thermosipho africanus (T); M83140

>S000017523 Thermosipho melanesiensis (T); BI429; Z70248

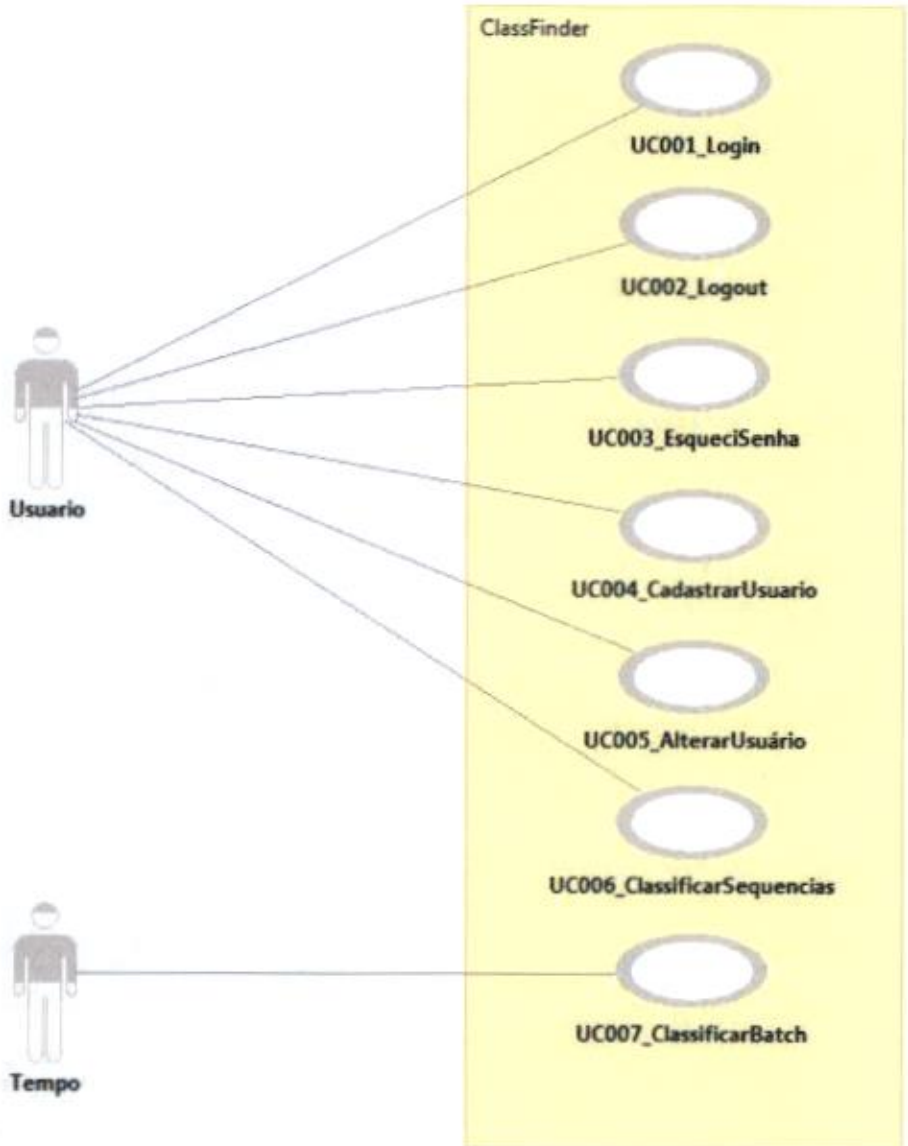
>S000382726 Thermosipho japonicus (T); IHB1; AB024932

>S000440162 Thermosipho atlanticus (T); type strain: DV1140; AJ577471

11 ANEXO B

11.1 Documentação do sistema classfinder

DIAGRAMA DE CASOS DE USO DO SISTEMA CLASSFINDER



11.1.1 Casos de uso (*Use Cases* - UCs)

UC001_Login

Controle do documento

Versão	Autor	Data	Descrição
1.0	Marcio Mariano	21/08/2008	Esboço inicial
1.1	Giovani Pisa	27/09/2008	Correções
1.2	Felipe Beni	27/11/2008	Correções

Descrição

Este Caso de Uso serve para efetuar o "Login" dos usuários no Sistema.

Pré-Condições

Este Case de Uso inicia somente se:

Usuário tiver acessado o link Login.

Pós-Condições

Após o término deste Caso de Uso, o sistema deve ter "Logado" o Usuário.

Ator

Usuário

Fluxo Principal de Eventos

1) Sistema apresenta tela DV02_Login com os campos disponíveis.
2) Usuário preenche os dados da tela.
3) Usuário clica no botão Enviar. (**A1**) (**A2**) (**E1**) (**E2**)
4) Sistema retorna para a página home do sistema.
5) Use case é finalizado.

Fluxo Alternativo

A1. Usuário escolhe a opção "Esqueci a Senha"

1) Sistema chama UC003_EsqueciSenha.
2) Caso de Uso é finalizado.

A2. Usuário escolhe a opção Cancelar

1) Sistema retorna para tela DV01_Inicial.
2) Caso de Uso é finalizado.

Fluxos de Exceção

E1. Nome de Usuário não confere

1) Sistema chama a tela DV02_Login com a mensagem "Usuário não encontrado".
2) Use case é finalizado.

E2. Senha incorreta

1) Sistema chama a tela DV02_Login com a mensagem "Senha não confere".
2) Use case é finalizado.

Regras de Negócio

N/A

Data Views

DV02_Login

UC002_Logout

Controle do documento

Versão	Autor	Data	Descrição
1.0	Felipe Beni	28/11/2008	Primeira Versão

Descrição

Este Caso de Uso serve para efetuar o "Logout" dos usuários no Sistema.

Pré-Condições

Este Case de Uso inicia somente se:

Usuário tiver acessado o link "Logout(login do usuario)".

Pós-Condições

Após o término deste Caso de Uso, o sistema deve ter efetuado o "Logout" do Usuário.

Ator

Usuário

Fluxo Principal de Eventos

1) Usuário preciona o link "logout" presente em todas as telas
2) Sistema efetua o Logout do Usuário.
3) Use case é finalizado.

Fluxo Alternativo

Não se aplica

Fluxos de Exceção

Não se aplica

Regras de Negócio

N/A

Data Views

DV00_Header

UC003_EsqueciSenha

Controle do documento

Versão	Autor	Data	Descrição
1.0	Felipe Beni	28/11/2008	Correções

Descrição

Este Caso de Uso serve para efetuar a solicitação de uma nova senha para o usuário.

Pré-Condições

Este Case de Uso inicia somente se:

Usuário tiver acessado o link "Esqueci a Senha".

Pós-Condições

Após o término deste Caso de Uso, o sistema deve ter enviado para o email do usuário sua nova senha.

Ator

Usuário

Fluxo Principal de Eventos

1) Usuário preenche dados referentes a sua conta.
2) Usuário clica no botão "Solicitar"; (**A1**) (**E1**) (**E2**) (**E3**)
3) Sistema gera uma nova senha para o Usuário;
4) Sistema envia a nova senha para o e-mail do usuário;
5) Use case é finalizado.

Fluxo Alternativo

A1. Usuário escolhe a opção Cancelar

1) Usuário clica no botão "Cancelar".
2) Sistema retorna para tela DV01_Inicial.
3) Caso de Uso é finalizado.

Fluxos de Exceção

E1. Campos obrigatórios não preenchidos

1) Usuário não preenche os campos da tela.
2) Sistema exibe a mensagem "Campo Obrigatório" ao lado do campo que é obrigatório. (R1)
3) Sistema retorna para passo 1 do fluxo principal.

E2. Valor menor que o permitido

1) Usuário preenche os campos com valores menores que o permitido.
2) Sistema exibe a mensagem "Valor menor que o permitido" ao lado do campo que é obrigatório.
3) Sistema retorna para passo 1 do fluxo principal.

E3. Usuário não encontrado com os dados digitados.

1) Usuário preenche os campos com valores diferentes dos contidos no banco de dados.
2) Sistema exibe a mensagem "Usuário não encontrado com os dados abaixo" acima dos campos
3) Sistema retorna para passo 1 do fluxo principal.

Regras de Negócio

R1. Todos os campos da tela são obrigatórios.

Data Views

DV03_EsqueciSenha

UC004_CadastrarUsuários

Controle do documento

Versão	Autor	Data	Descrição
1.0	Marcio Mariano	21/08/2008	Esboço inicial
1.1	Giovani Pisa	27/09/2008	Correções
1.2	Felipe Beni	27/11/2008	Correções

Descrição

Este Caso de Uso serve para cadastrar os Usuários do Sistema ClassFinder.

Pré-Condições

Este Case de Uso inicia somente se:

Usuário tiver escolhido a opção "Registro".

Pós-Condições

Após o termino deste Caso de Uso, o sistema deve ter cadastrado o Usuário no banco de dados do sistema.

Ator

Usuário

Fluxo Principal de eventos

1) Sistema apresenta tela DV04_CadastrarUsuario.
2) Usuário preenche os dados da tela.
3) Usuário clica no botão Salvar (A1) (E1) (E2)
4) Usuário verifica que os campos "Senha" e "Confirmação de Senha" são iguais. (**E3**)
5) Usuário verifica no banco de dados que o Login escolhido não existe.

(**E4**)

6) Sistema retorna para tela DV01_Inicial.
7) Caso de Uso é finalizado.

Fluxo Alternativo

A1. Usuário escolhe a opção Cancelar

1) Usuário clica no botão Cancelar.
2) Sistema retorna para tela DV01_Inicial.
3) Caso de Uso é finalizado.

Fluxos de Exceção

E1. Campos obrigatórios não preenchidos

1) Usuário não preenche os campos da tela.
2) Sistema exibe a mensagem "Campo Obrigatório" ao lado do campo que é obrigatório. (R1)
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

E2. Valor menor que o permitido

1) Usuário preenche os campos com valores menores que o permitido.
2) Sistema exibe a mensagem "Valor menor que o permitido" ao lado do campo que é obrigatório.
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

E3. O campo Confirmação de Senha é diferente do campo Senha.

1) Usuário preenche os dados "Senha" e "Confirmação de Senha" diferentes.
2) Sistema exibe a mensagem "A confirmação de senha é diferente da senha" no topo da página.
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

E4. Login já existente no Banco de dados.

1) Usuário preenche login já existente no banco de dados.
2) Sistema exibe a mensagem "Login de usuário já existente" no topo da página.
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

Regras de Negócio

R1: Os campos obrigatórios são: Nome, E-mail, Login, Senha, Confirmação de Senha.

Data Views

DV04_CadastrarUsuario

UC005_AlterarUsuário

Controle do documento

Versão	Autor	Data	Descrição
1.0	Marcio Mariano	21/08/2008	Esboço inicial
1.1	Giovani Pisa	27/09/2008	Correções
1.2	Felipe Beni	27/11/2008	Correções

Descrição

Este Caso de Uso serve para alterar o cadastro do usuário no Sistema ClassFinder

Pré-Condições

Este Case de Uso inicia somente se:

Usuário tiver efetuado Login no sistema.

Usuário tiver escolhido a opção "Minha Conta"

Pós-Condições

Após o término deste Caso de Uso, o sistema deverá ter modificado os dados do usuário.

Ator

Usuário

Fluxo Principal de eventos

1) Sistema realiza consulta do usuário a ser alterado.
2) Sistema apresenta tela DV05_AlterarUsuario com os dados do usuário preenchidos, menos os campos "Senha" e "Confirmação de Senha", e desbloqueados para edição.

3) Usuário altera os campos desejados. (**R1**)
4) Usuário clica no botão Salvar (**A1**) (**A2**) (**E1**) (**E2**)
5) Usuário verifica que os campos "Senha" e "Confirmação de Senha" são iguais. (**E3**)
6) Sistema retorna para tela DV01_Inicial.

Fluxo Alternativo

A1. Usuário escolhe a opção Cancelar

1) Sistema retorna para tela DV01_Inicial.
2) Caso de Uso é finalizado.

A2. Usuário escolhe a opção Excluir

1) Sistema executa o Logout do usuário;
2) Sistema exclui o Usuário que estava Logado;
3) Sistema retorna para tela DV01_Inicial;
4) Caso de Uso é finalizado.

Fluxos de Exceção

E1. Campos obrigatórios não preenchidos

1) Usuário não preenche os campos da tela.
2) Sistema exibe a mensagem "Campo Obrigatório" ao lado do campo que é obrigatório. (R1)
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

E2. Valor menos que o permitido

1) Usuário preenche os campos com valores menores que o permitido.
2) Sistema exibe a mensagem "Valor menor que o permitido" ao lado do campo que é obrigatório. (R2)
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

E3. O campo Confirmação de Senha é diferente do campo Senha.

1) Usuário preenche os dados "Senha" e "Confirmação de Senha" diferentes.
2) Sistema exibe a mensagem "A confirmação de senha é diferente da senha" no topo da página.
3) Retorna para passo 2 do fluxo principal, mantendo os dados já preenchidos, menos os campos senha e confirmação de senha.

Regras de Negócio

R1: Os campos "Senha" e "Confirmação de Senha" são de preenchimento obrigatório toda vez que o usuário solicitar uma alteração de cadastro.

Data Views

DV05_AlterarUsuario

UC006_ClassificarSeqüências

Controle do documento

Versão	Autor	Data	Descrição
1.0	Marcio Mariano	21/08/2008	Esboço inicial
1.1	Sandro Lima	23/08/2008	Correção
1.2	Giovani Pisa	27/09/2008	Correções
1.3	Felipe Beni	28/11/2008	Correções

Descrição

Este Caso de Uso serve para classificar taxonomicamente uma seqüência de pares de bases.

Pré-Condições

Este Case de Uso inicia somente se:

O usuário tiver acessado o link "Classificador".

Pós-Condições

Após o termino deste Caso de Uso, o sistema deve ter identificado a seqüência submetida pelo Usuário.

Ator

Usuário

Fluxo Principal de eventos

1) Sistema apresenta tela DV06_Classificador.
2) Usuário preenche campo de texto com a(s) seqüência(s) deseja(s). (**A1**)
3) Usuário clica no botão Enviar. (**A2**)
4) Sistema verifica que a seqüência é do tipo FASTA. (**E1**)
5) Sistema verifica que a quantidade de seqüências no arquivo é menor que

10. (**A3**)

6) Sistema codifica a seqüência submetida pelo usuário para um código tri-gram.
7) Sistema envia o código tri-gram para a rede neural treinada para classificação.
8) Sistema apresenta tela DV07_Resposta com os dados retornados pela rede neural.
9) Caso de Uso é finalizado.

Fluxo Alternativo

A1. Usuário seleciona um arquivo para fazer um *upload*.

1) Sistema retorna para o passo 3 do fluxo principal.
2) Caso de Uso é finalizado.

A2. Usuário clica Limpar.

1) Sistema retorna para passo 1 do Use case.
2) Caso de Uso é finalizado.

A3. Arquivo contém mais de 10 sequencias.

1) Sistema verifica que o usuário está "Logado" no sistema. (**E2**)
2) Sistema verifica que a quantidade de seqüências no arquivo é maior que 10.
3) Sistema armazena seqüências no banco de dados batch.
4) Caso de Uso é finalizado.

Fluxos de Exceção

E1. Seqüência não é do tipo FASTA.

1) Sistema exibe a mensagem "O formato do arquivo é inválido, certifique-se que é um arquivo fasta" no topo da página. (R1)
2) Sistema retorna para o passo 2 do fluxo principal.

E2. Usuário não está "Logado".

1) Sistema exibe a mensagem "Para enviar mais do que 10 seqüências simultâneas, você precisa estar Logado".
2) Sistema retorna para o passo 2 do fluxo principal.

Regras de Negócio

R1: Para validar um arquivo fasta, devemos conferir se a seqüência é dividida em duas linhas, e a primeira precisa iniciar com sinal de maior (>).

Data Views

DV06_Classificador

UC007_ClassificarBatch

Controle do documento

Versão	Autor	Data	Descrição
1.0	Felipe Beni	28/11/2008	Primeira Versão

Descrição

Este Caso de Uso serve para classificar taxonomicamente uma seqüência de pares de bases quando o arquivo enviado contém mais de 10 seqüências.

Pré-Condições

Este Case de Uso inicia somente se:

A hora do sistema for igual a 23:59.

Pós-Condições

Após o termino deste Caso de Uso, o sistema deve ter classificado as seqüências submetidas pelos Usuários e ter enviado o resultado por e-mail.

Ator

Tempo

Fluxo Principal de Eventos

1) Sistema verifica que o horário é 23:59.
2) Sistema pesquisa as seqüências enviadas no dia corrente.
3) Sistema codifica as seqüências pesquisadas para um código tri-gram.
4) Sistema envia o código tri-gram para a rede neural treinada.
5) Sistema envia e-mail ao respectivo usuário com as respostas para suas seqüências.

6) Use case é finalizado.

Fluxo Alternativo

Não se aplica

Fluxos de Exceção

Não se aplica

Regras de Negócio

Não se aplica

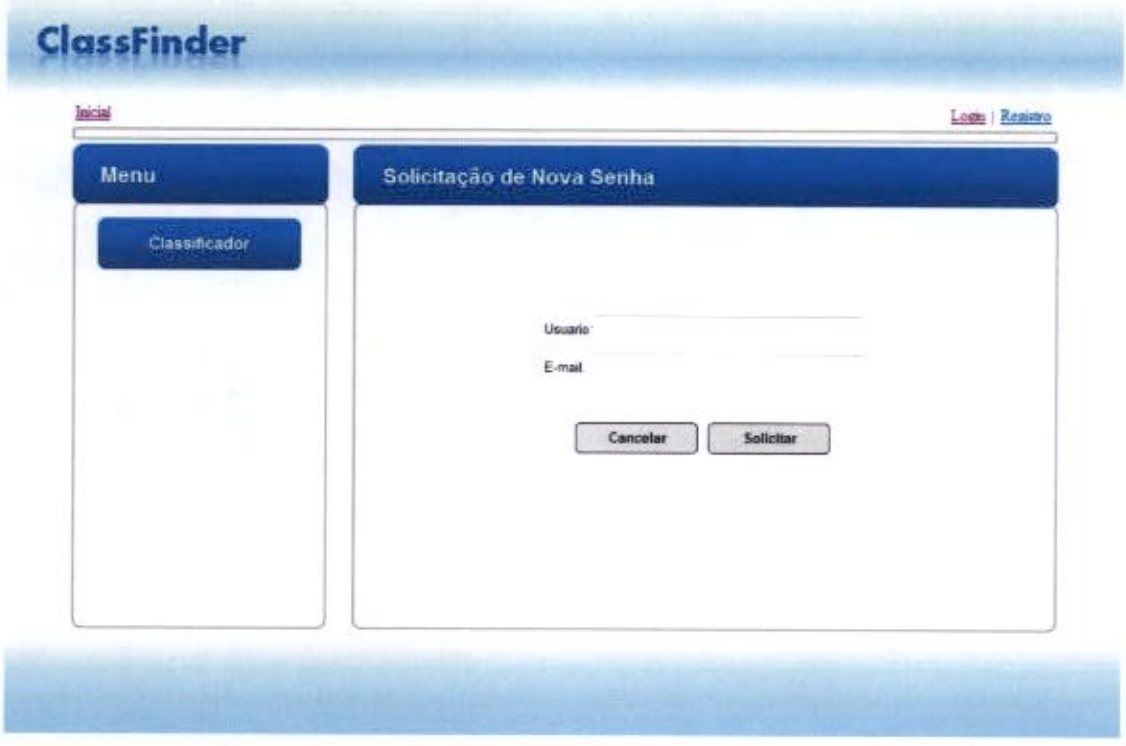
Data Views

Não se aplica

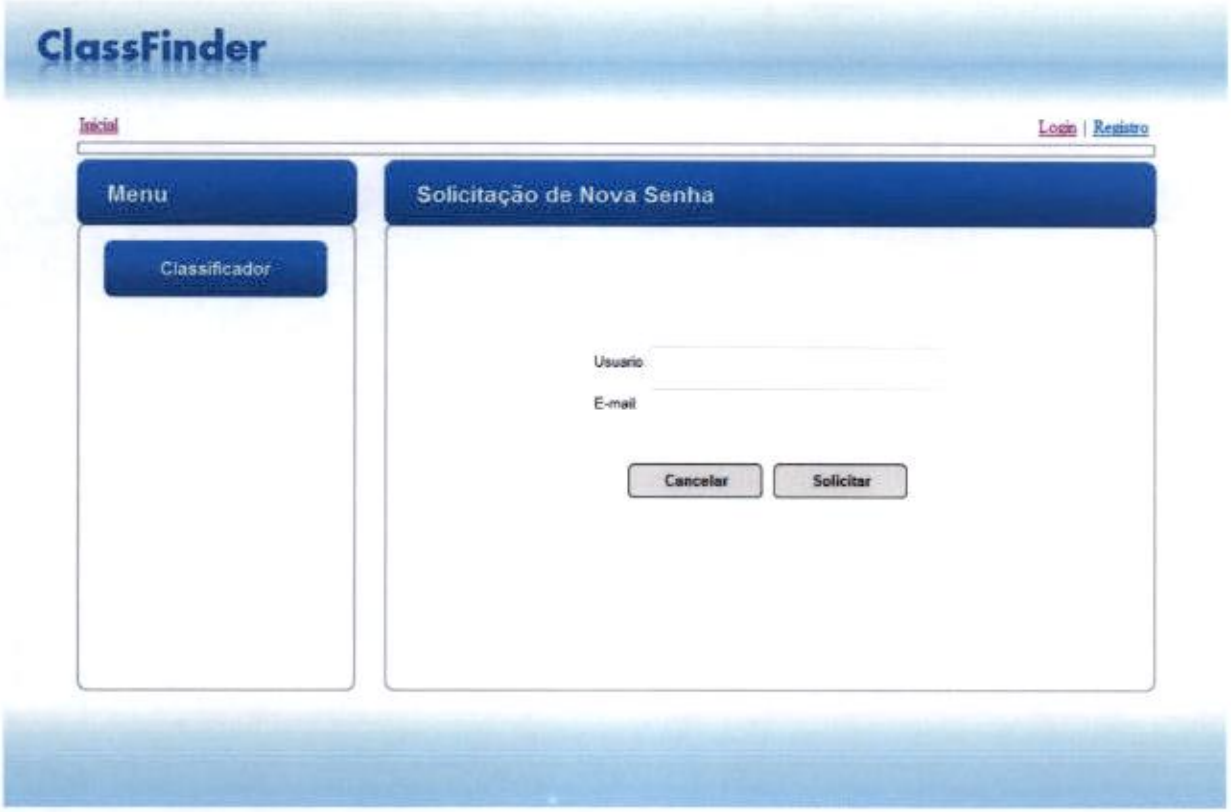
11.1.2 Telas (*data views*)

CABEÇALHO – DV00 - HEADER

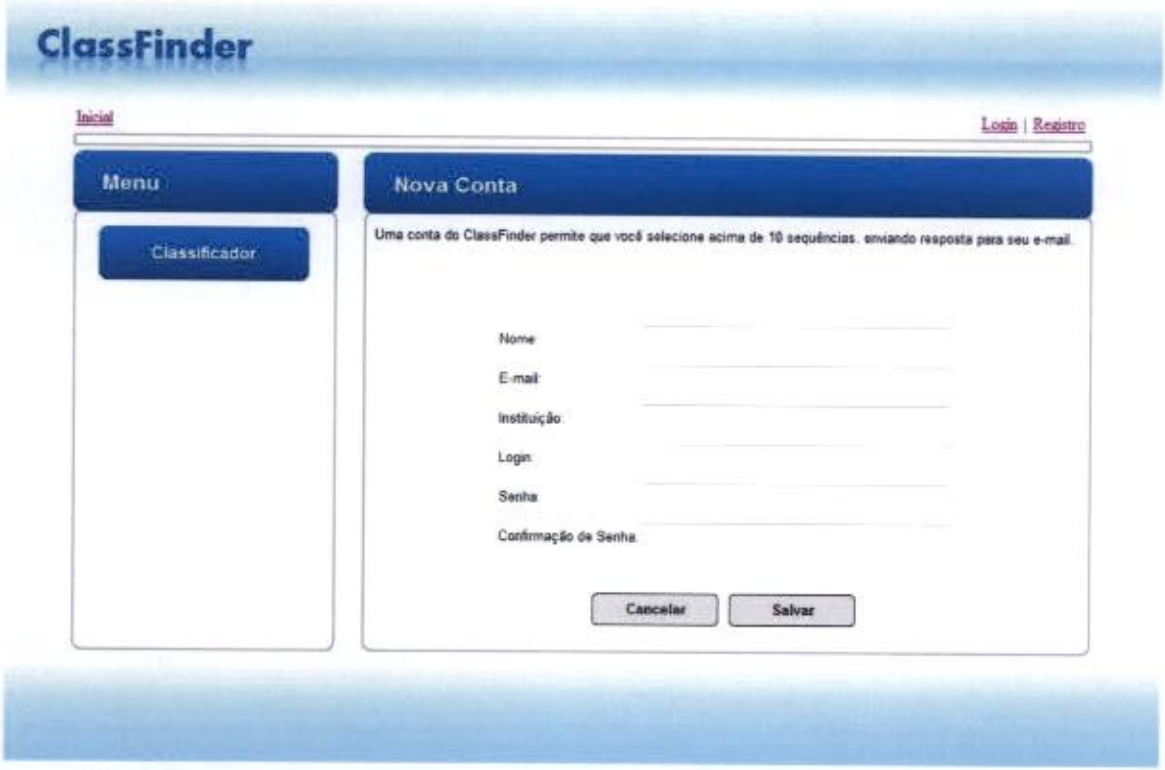
TELA DE LOGIN – DV02 - LOGIN



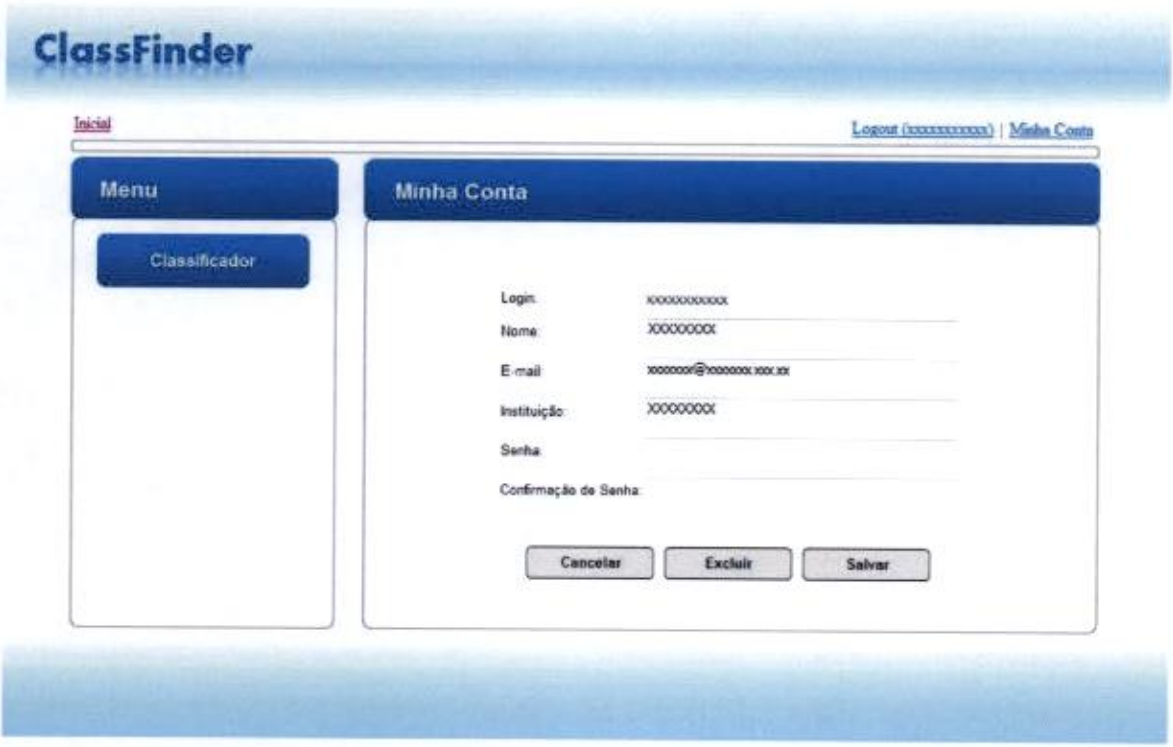
TELA DE SOLICITAÇÃO DE SENHA – DV03 – SOLICITAR SENHA



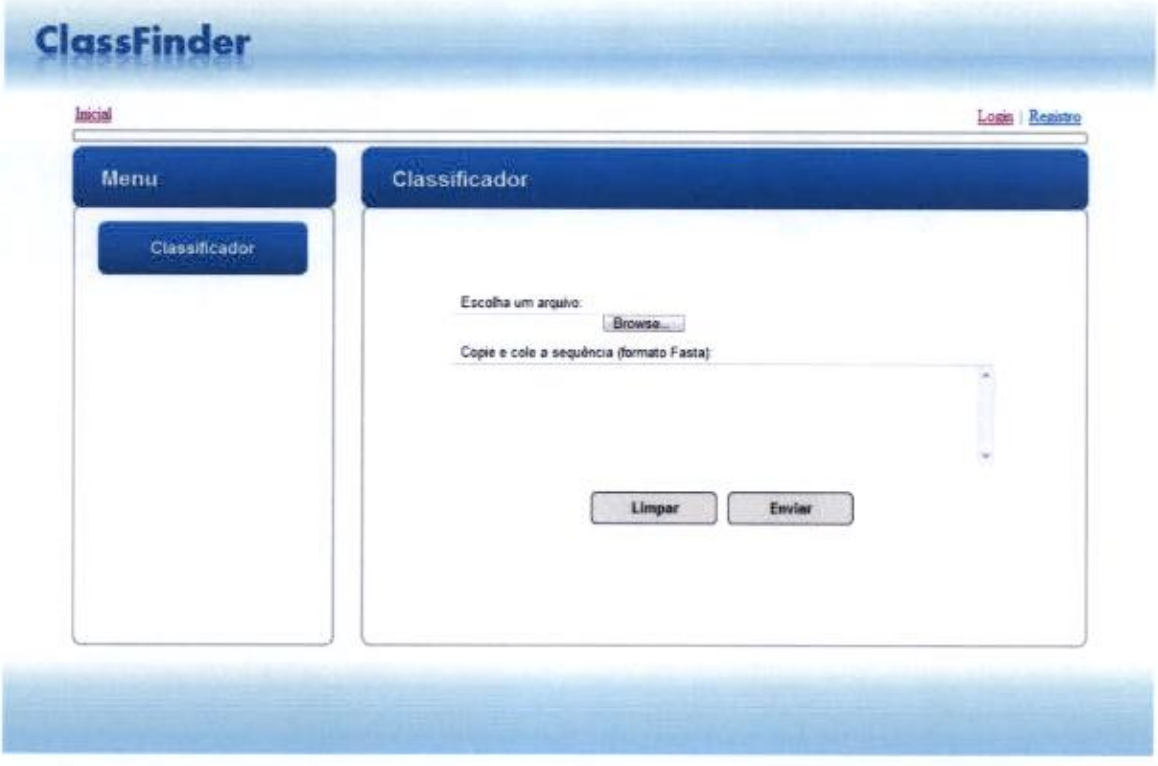
TELA DE CADASTRO DE USUÁRIOS – DV04 – CADASTRAR USUÁRIOS



TELA DE ALTERAÇÃO DE USUÁRIOS – DV05 – ALTERAR USUÁRIO



TELA DE CLASSIFICAÇÃO – DV06 - CLASSIFICADOR



TELA DE RESPOSTA – DV07 – RESPOSTA

11.1.3 Diagramas de classes

DIAGRAMAS DE CLASSES: MÓDULO USUÁRIO

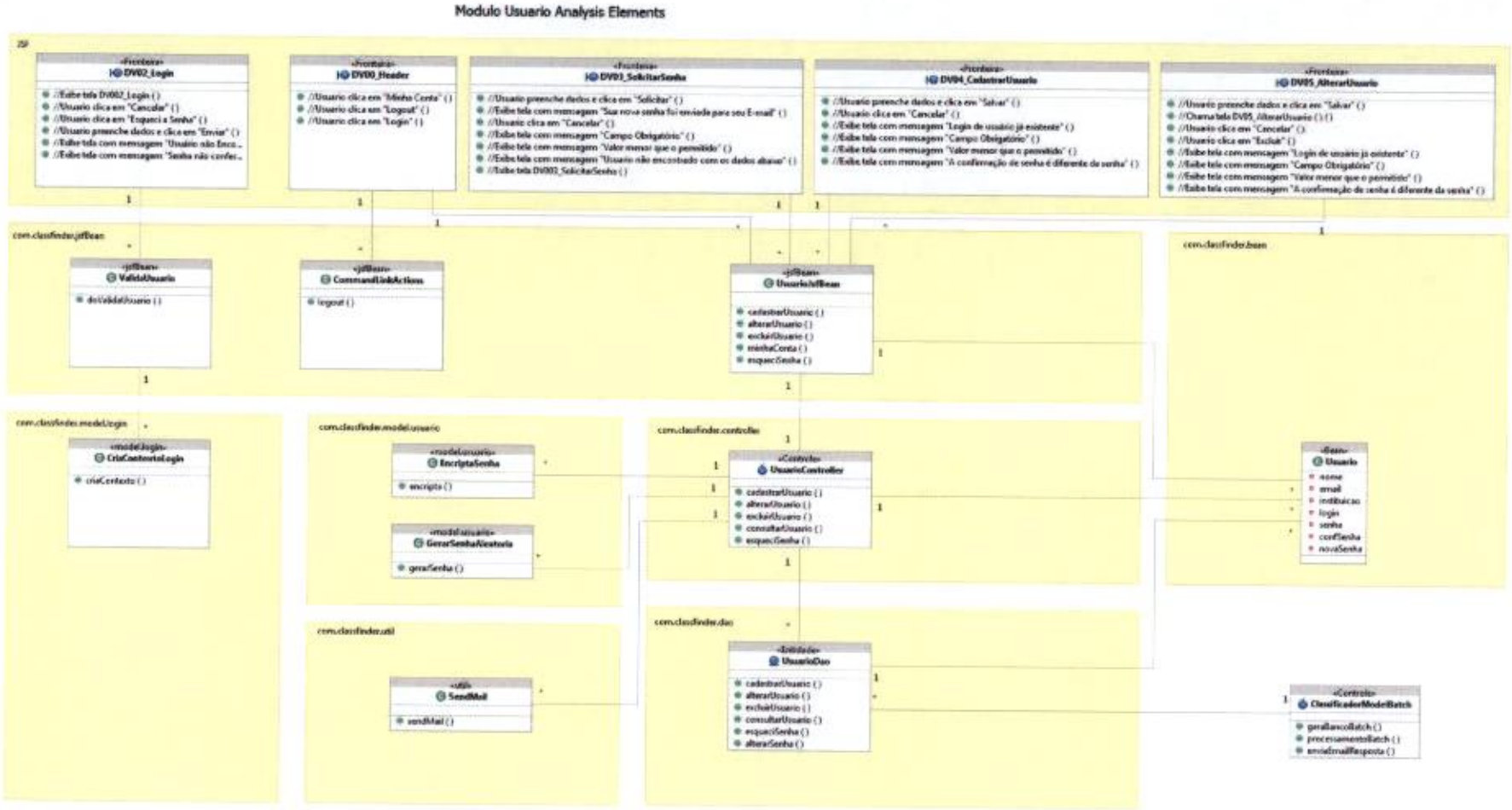
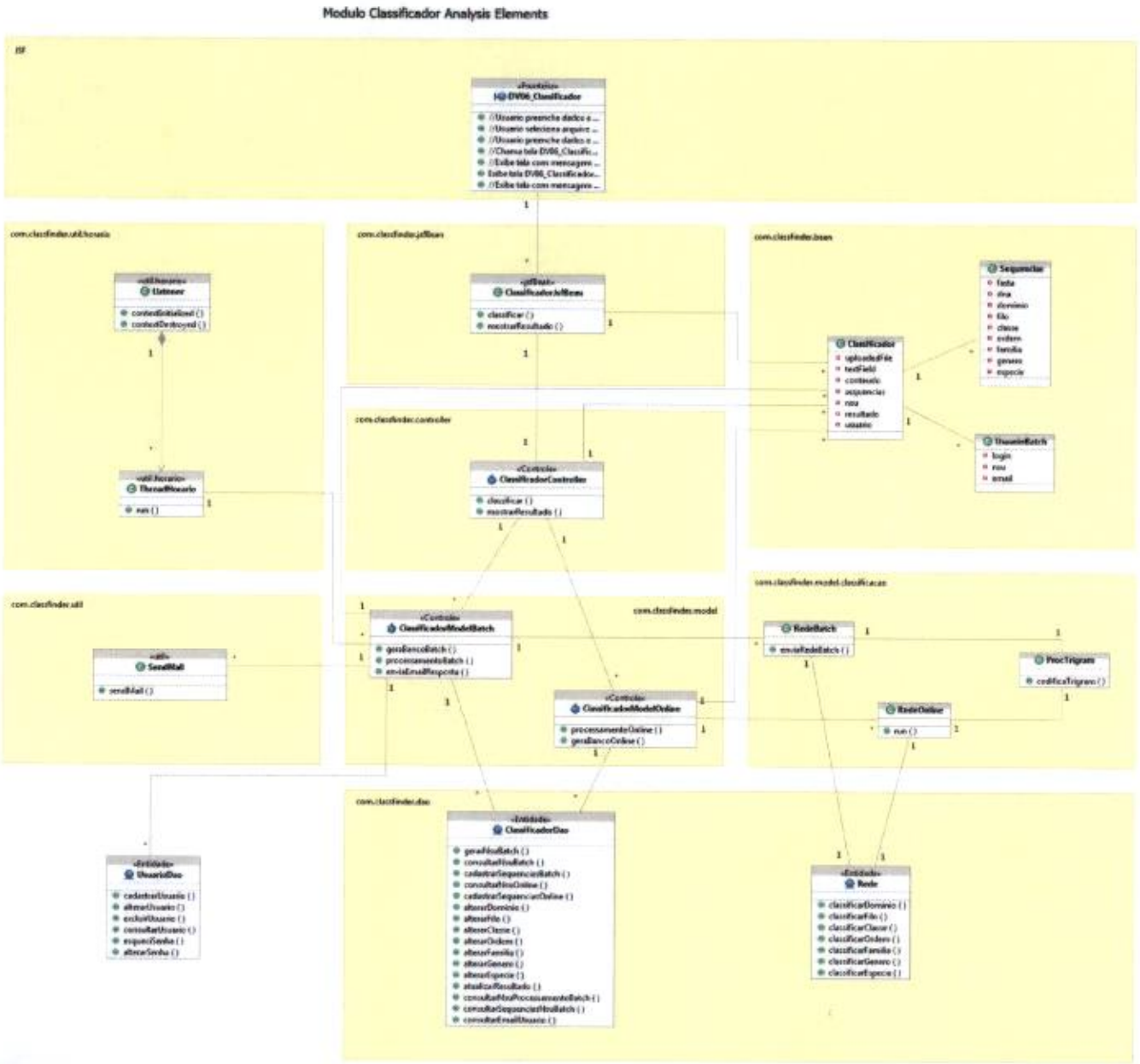
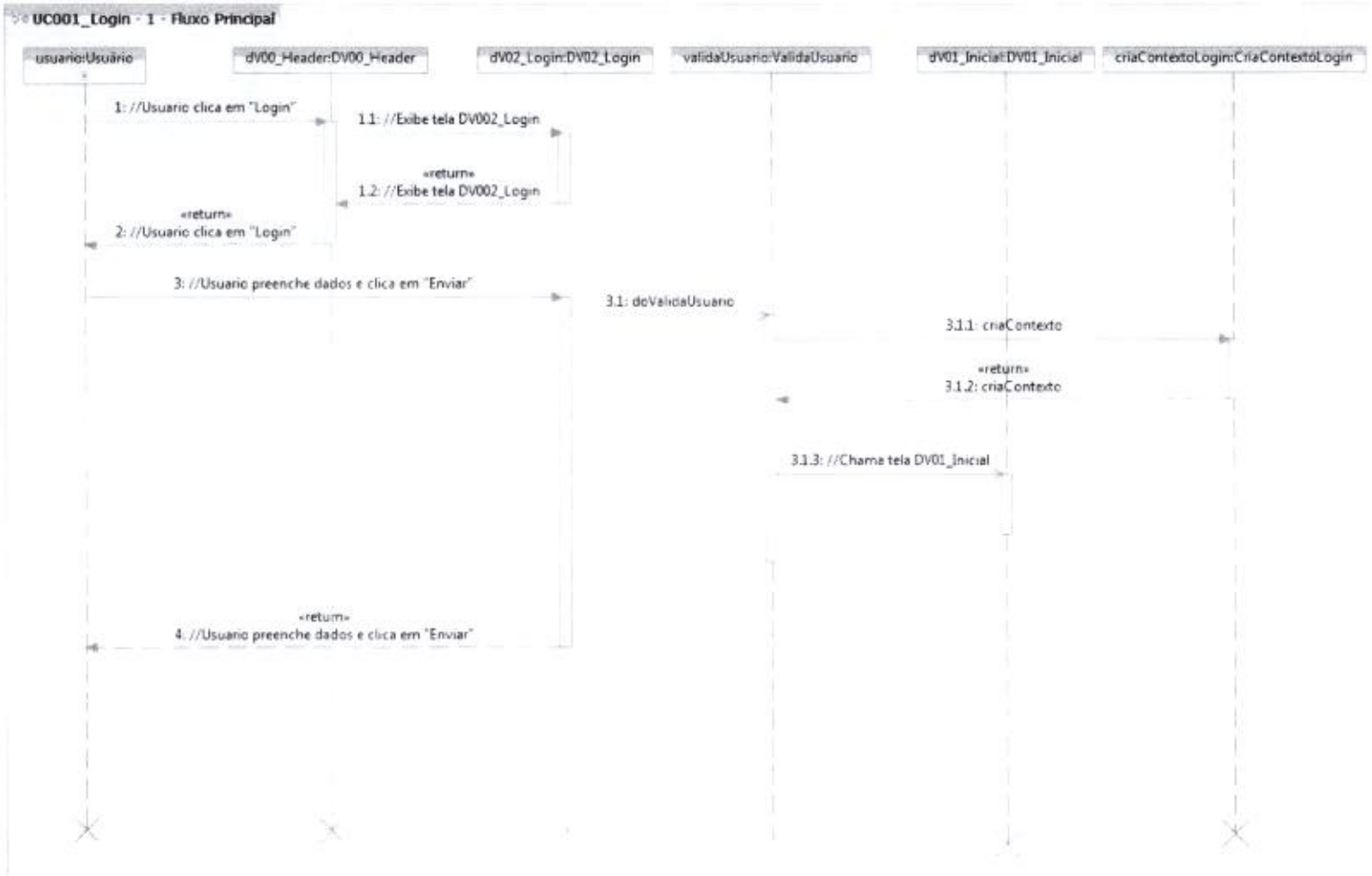
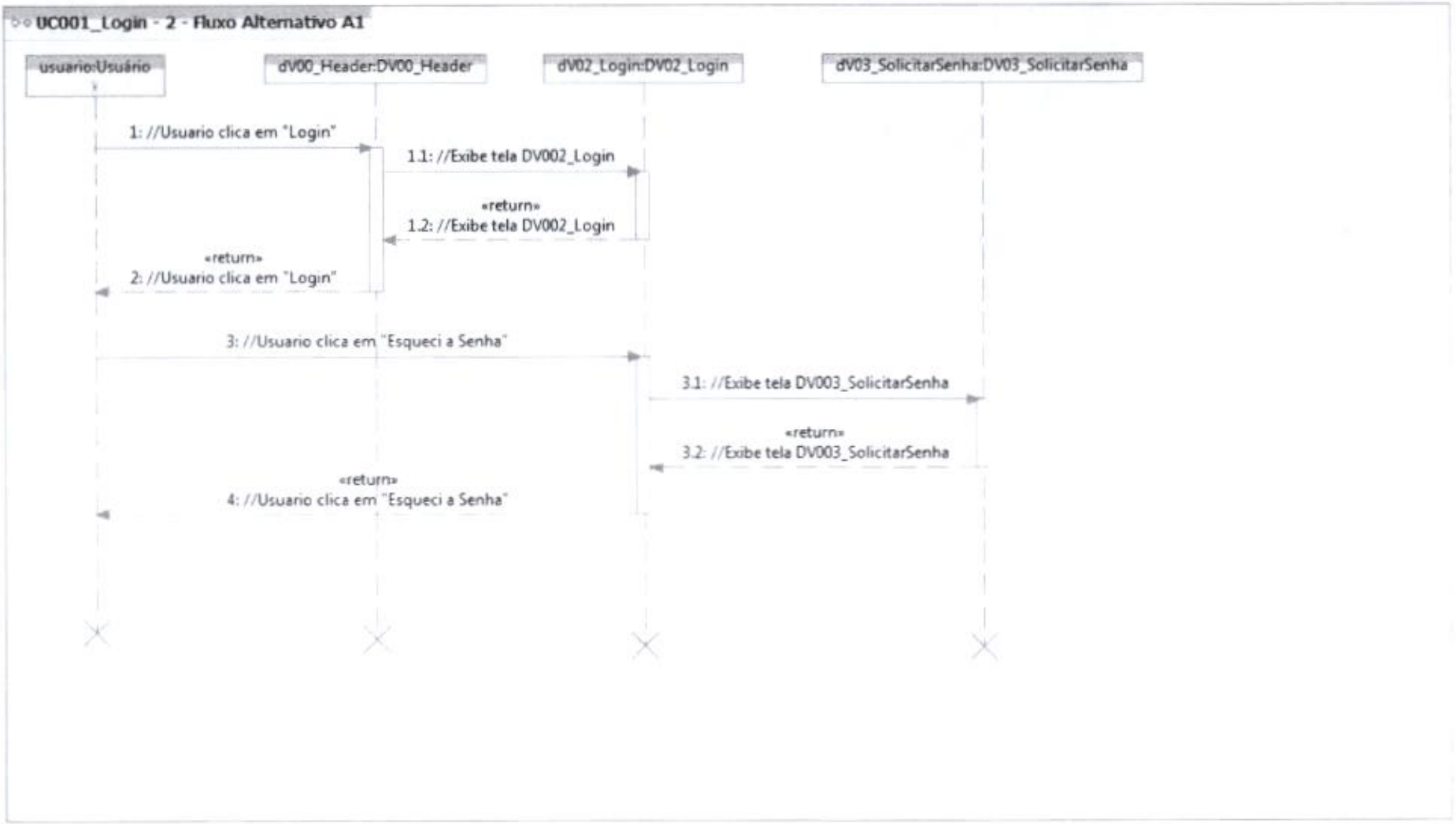


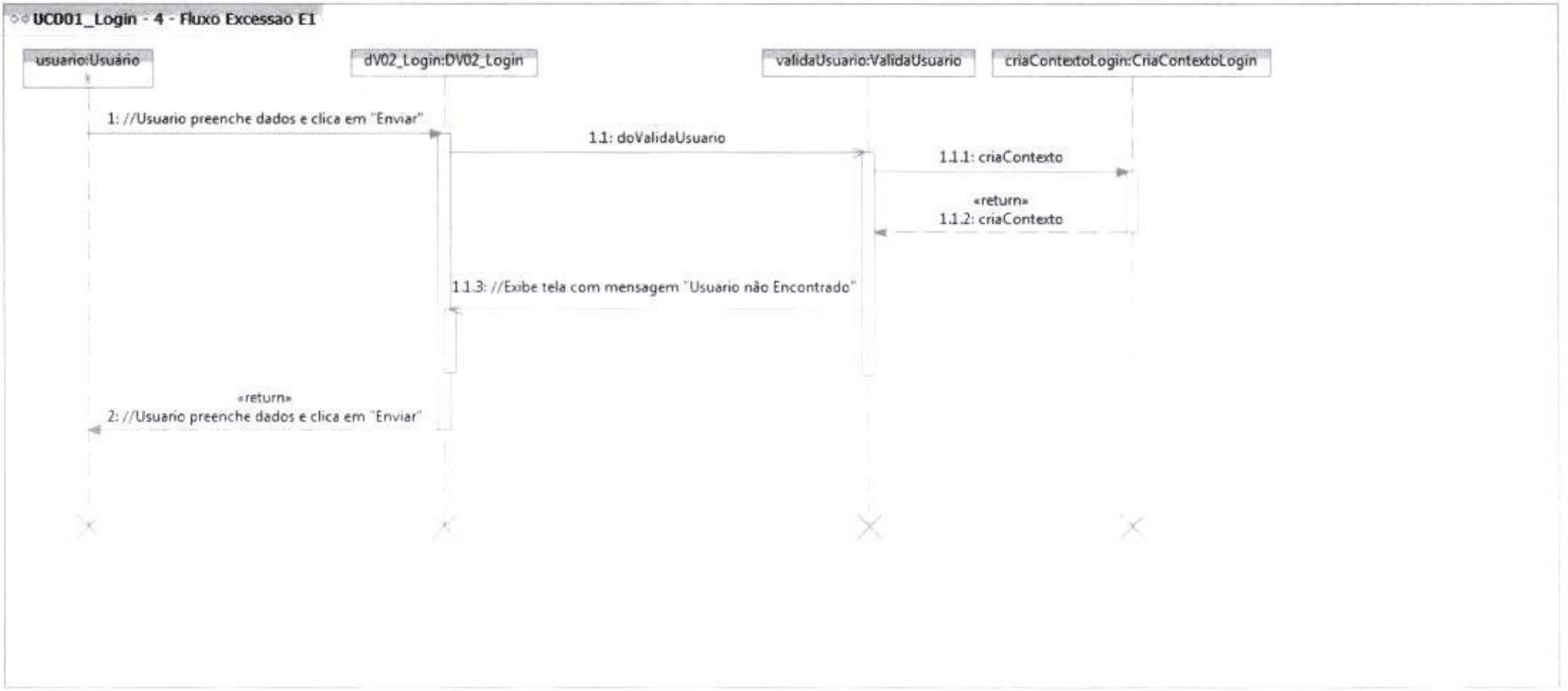
DIAGRAMA DE CLASSES: MÓDULO CLASSIFICADOR

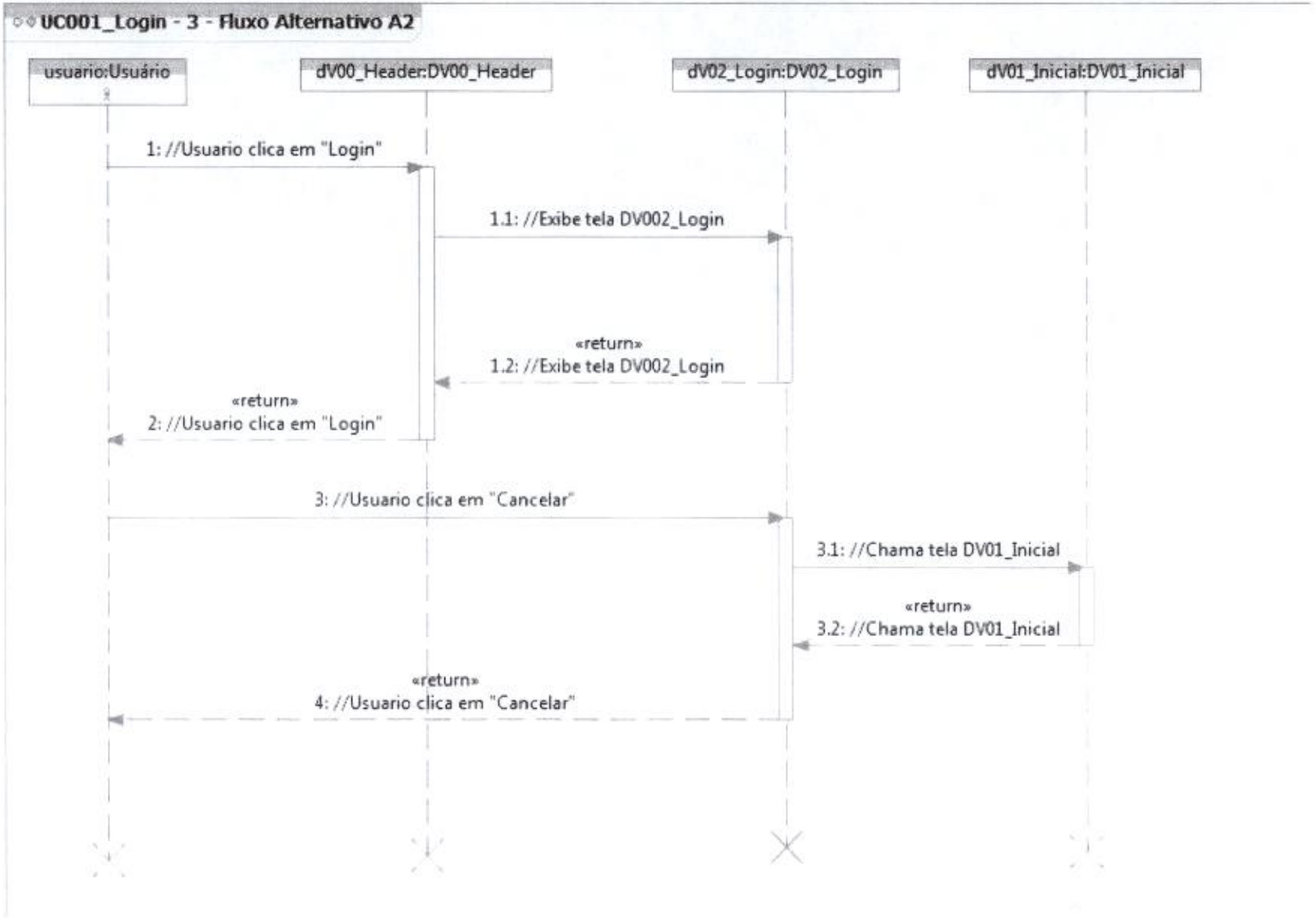


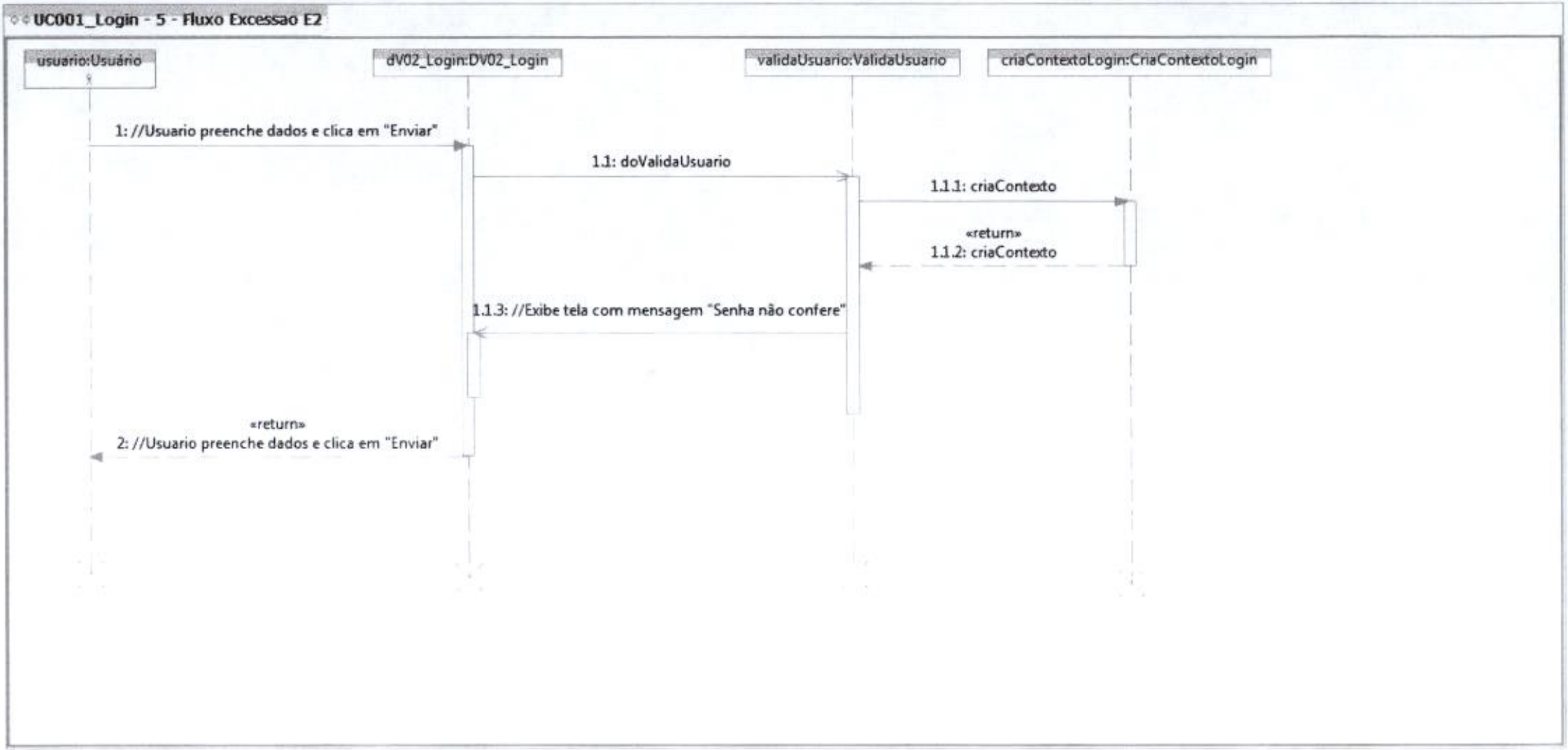
11.1.4 Diagramas de seqüências

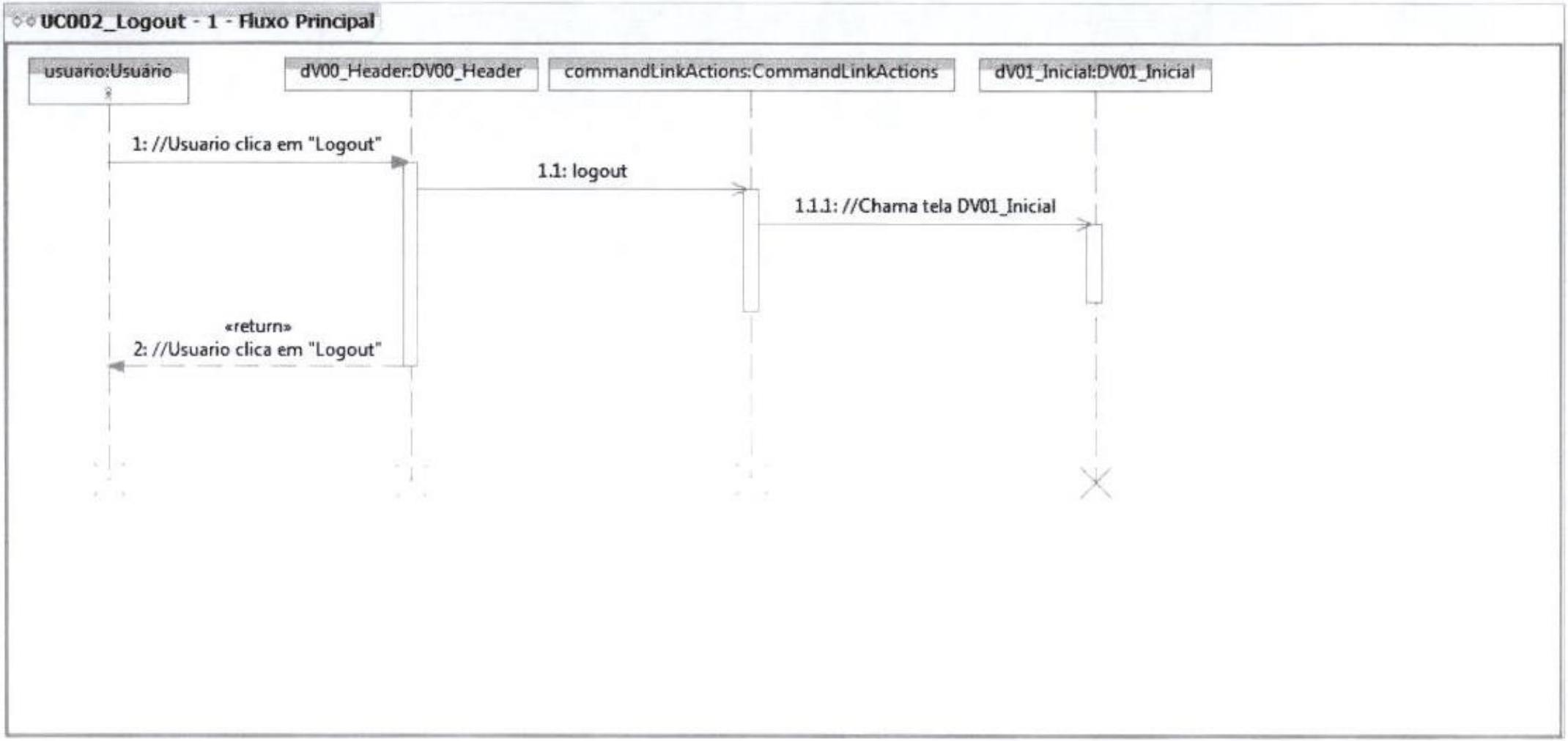


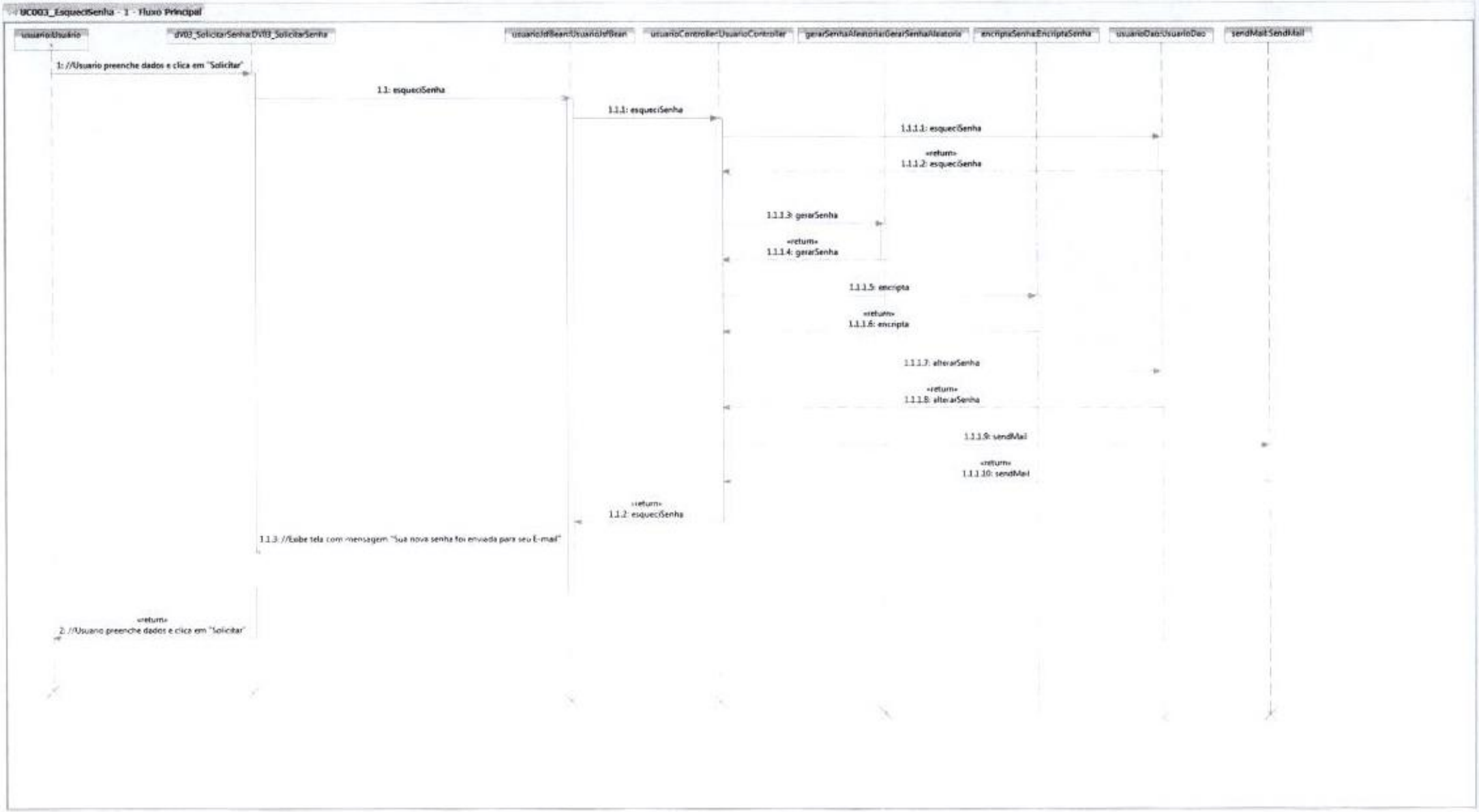

UC001_Login - 2 - Fluxo Alternativo A1
usuario:Usuário
dV00_Header:DV00_Header
dV02_Login:DV02_Login
dV03_SolicitarSenha:DV03_SolicitarSenha
1: //Usuario clica em "Login"
1.1: //Exibe tela DV002_Login
«return»
1.2: //Exibe tela DV002_Login
«return»
2: //Usuario clica em "Login"
3: //Usuario clica em "Esqueci a Senha"
3.1: //Exibe tela DV003_SolicitarSenha
«return»
3.2: //Exibe tela DV003_SolicitarSenha
«return»
4: //Usuario clica em "Esqueci a Senha"

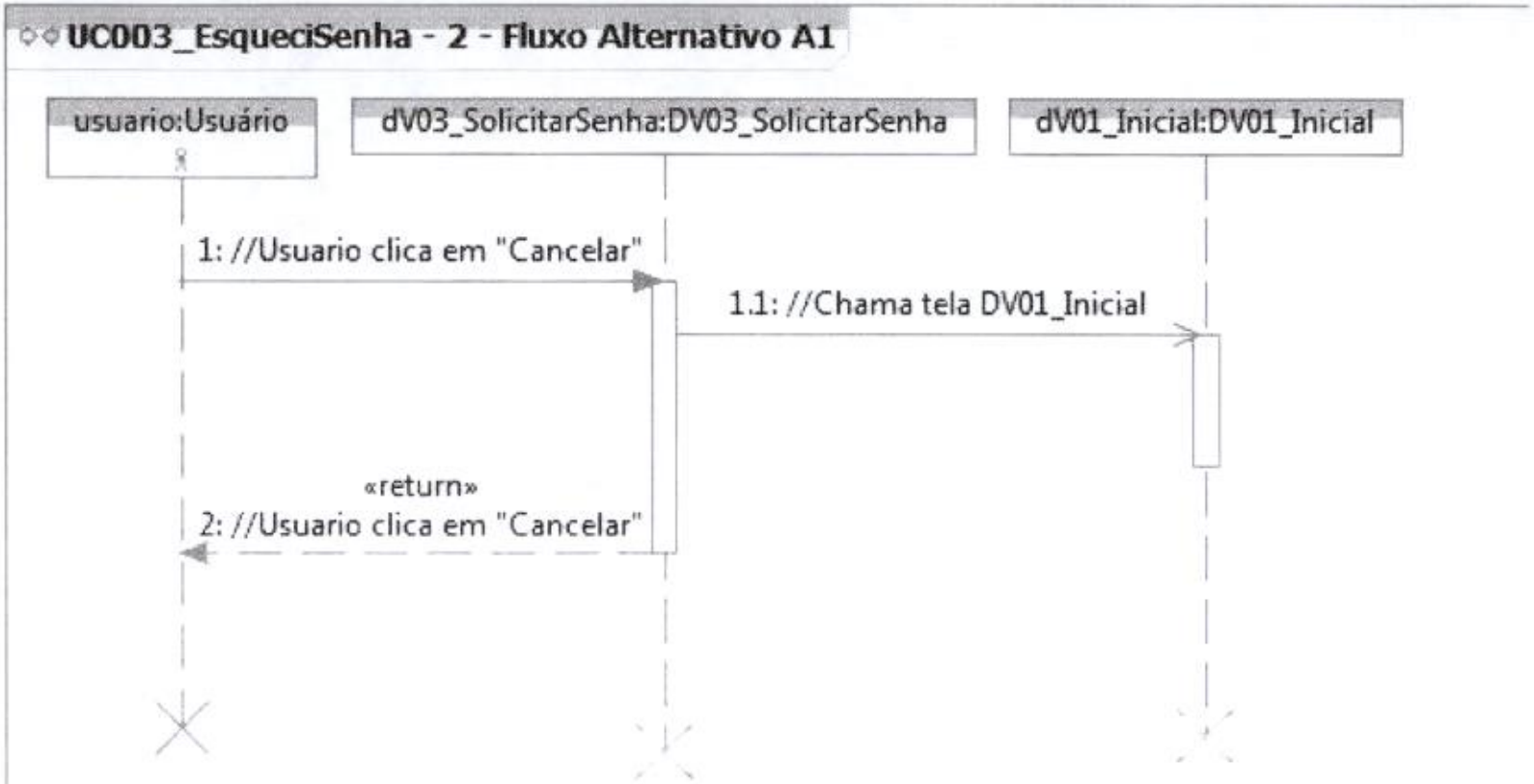

UC001_Login - 4 - Fluxo Excessao E1
usuario:Usuário
dV02_Login:DV02_Login
validaUsuario:ValidaUsuario
criaContextoLogin:CriaContextoLogin
1: //Usuario preenche dados e clica em "Enviar"
1.1: doValidaUsuario
1.1.1: criaContexto
«return»
1.1.2: criaContexto
1.1.3: //Exibe tela com mensagem "Usuario não Encontrado"
«return»
2: //Usuario preenche dados e clica em "Enviar"

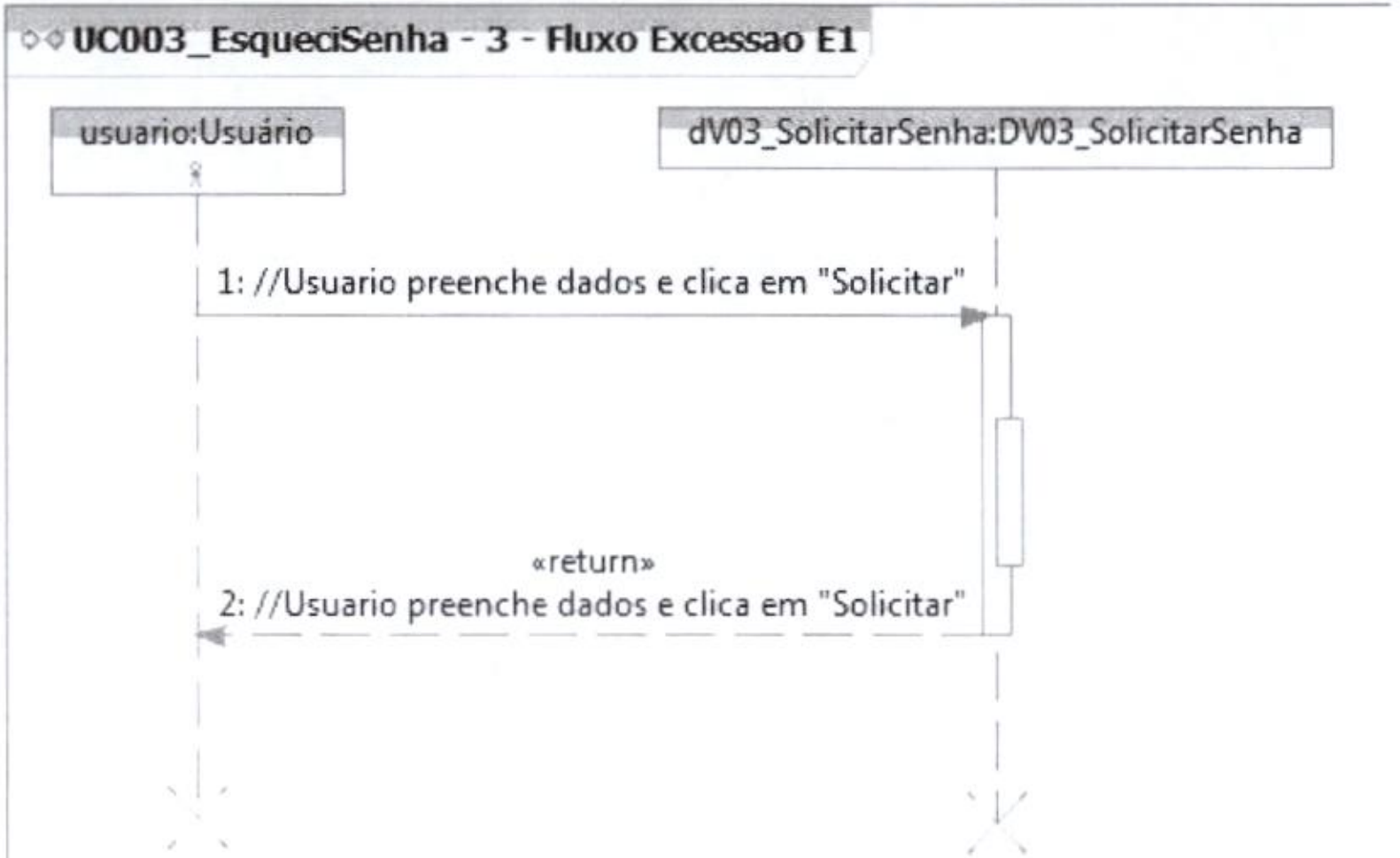

UC001_Login - 3 - Fluxo Alternativo A2
usuario:Usuário
dV00_Header:DV00_Header
dV02_Login:DV02_Login
dV01_Inicial:DV01_Inicial
1: //Usuario clica em "Login"
1.1: //Exibe tela DV002_Login
«return»
1.2: //Exibe tela DV002_Login
«return»
2: //Usuario clica em "Login"
3: //Usuario clica em "Cancelar"
3.1: //Chama tela DV01_Inicial
«return»
3.2: //Chama tela DV01_Inicial
«return»
4: //Usuario clica em "Cancelar"

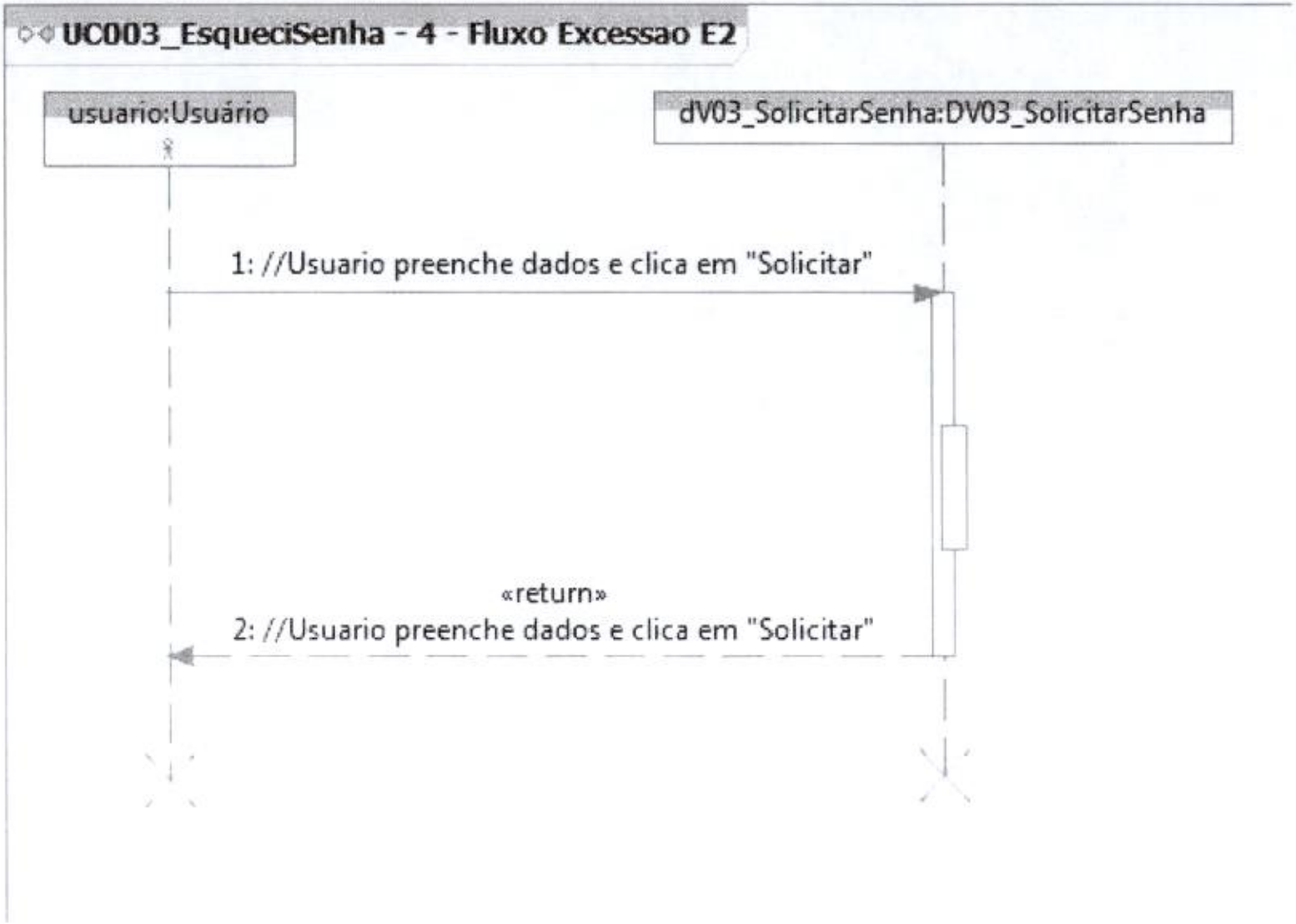

UC001_Login - 5 - Fluxo Excessao E2
usuario:Usuário
dV02_Login:DV02_Login
validaUsuario:ValidaUsuario
criaContextoLogin:CriaContextoLogin
1: //Usuario preenche dados e clica em "Enviar"
1.1: doValidaUsuario
1.1.1: criaContexto
«return»
1.1.2: criaContexto
1.1.3: //Exibe tela com mensagem "Senha não confere"
«return»
2: //Usuario preenche dados e clica em "Enviar"

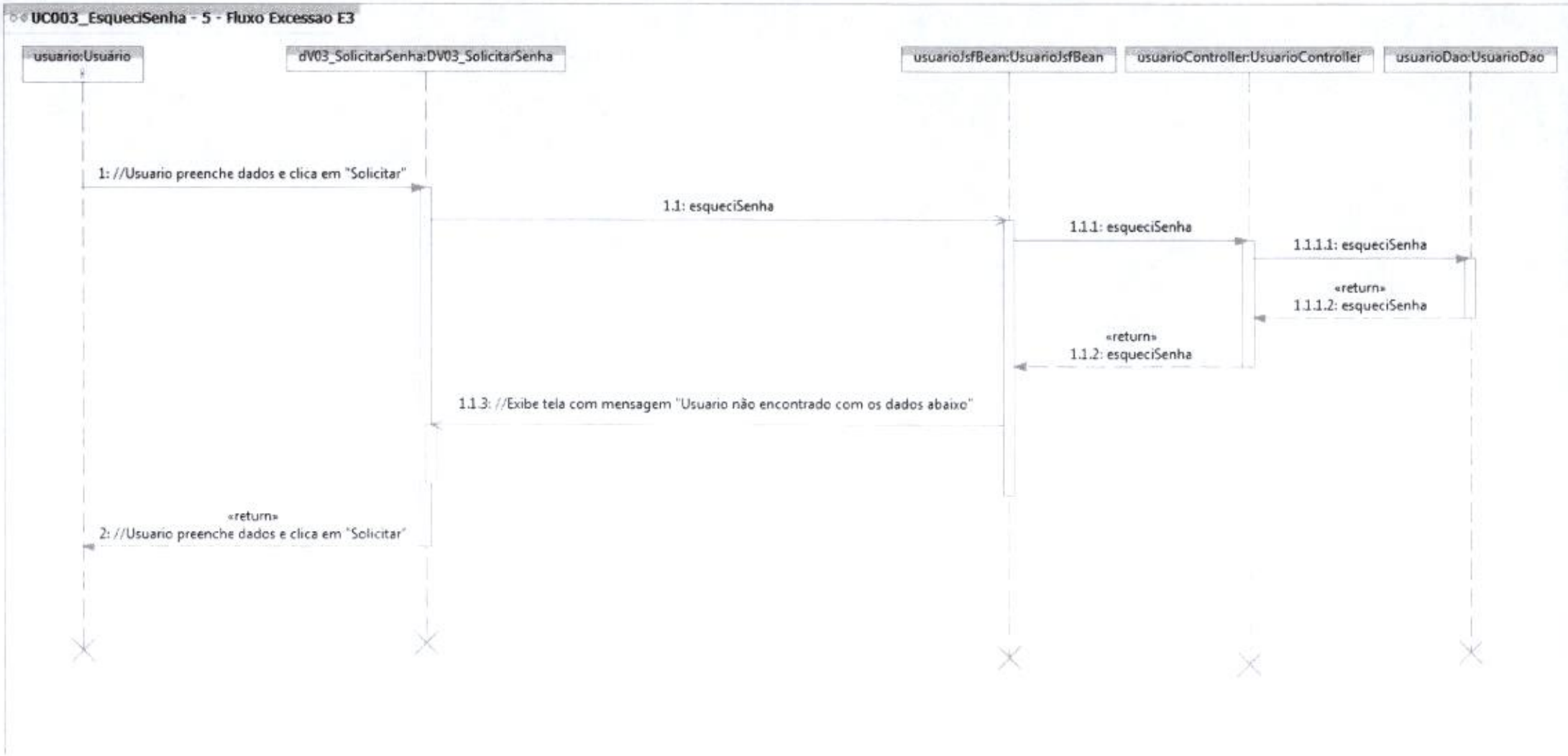

UC002_Logout - 1 - Fluxo Principal
usuario:Usuário
dV00_Header:DV00_Header
commandLinkActions:CommandLinkActions
dV01_Inicial:DV01_Inicial
1: //Usuario clica em "Logout"
1.1: logout
1.1.1: //Chama tela DV01_Inicial
«return»
2: //Usuario clica em "Logout"

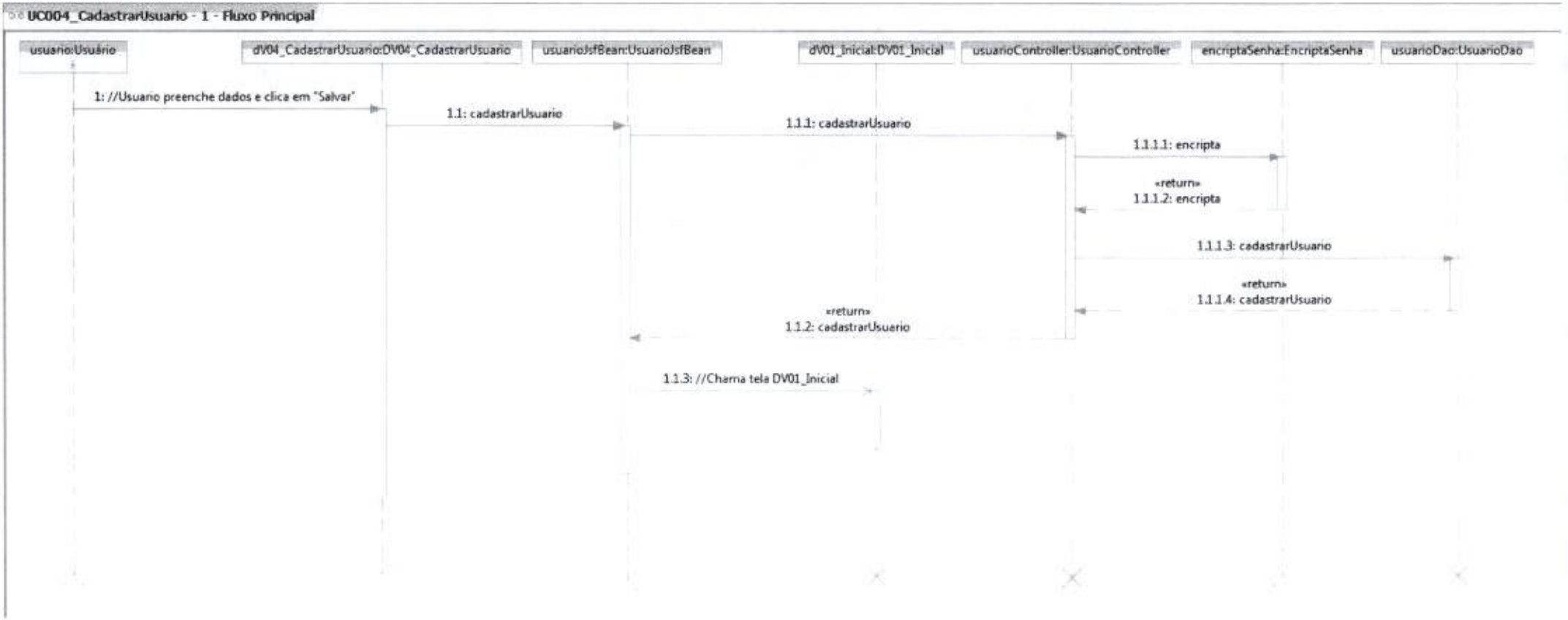

UC003_EsqueciSenha - 1 - Fluxo Principal
usuario:Usuario
dv03_SolicitarSenha:Dv03_SolicitarSenha
usuarioJdfBean:UsuarioJdfBean
usuarioController:UsuarioController
gerarSenhaAleatoria:GerarSenhaAleatoria
encriptaSenha:EncriptaSenha
usuarioDao:UsuarioDao
sendMail:SendMail
1: //Usuario preenche dados e clica em "Solicitar"
1.1: esqueciSenha
1.1.1: esqueciSenha
1.1.1.1: esqueciSenha
«return»
1.1.1.2: esqueciSenha
1.1.1.3: gerarSenha
«return»
1.1.1.4: gerarSenha
1.1.1.5: encripta
«return»
1.1.1.6: encripta
1.1.1.7: alterarSenha
«return»
1.1.1.8: alterarSenha
1.1.1.9: sendMail
«return»
1.1.1.10: sendMail
«return»
1.1.2: esqueciSenha
1.1.3: //Exibe tela com mensagem "Sua nova senha foi enviada para seu E-mail"
«return»
2: //Usuario preenche dados e clica em "Solicitar"

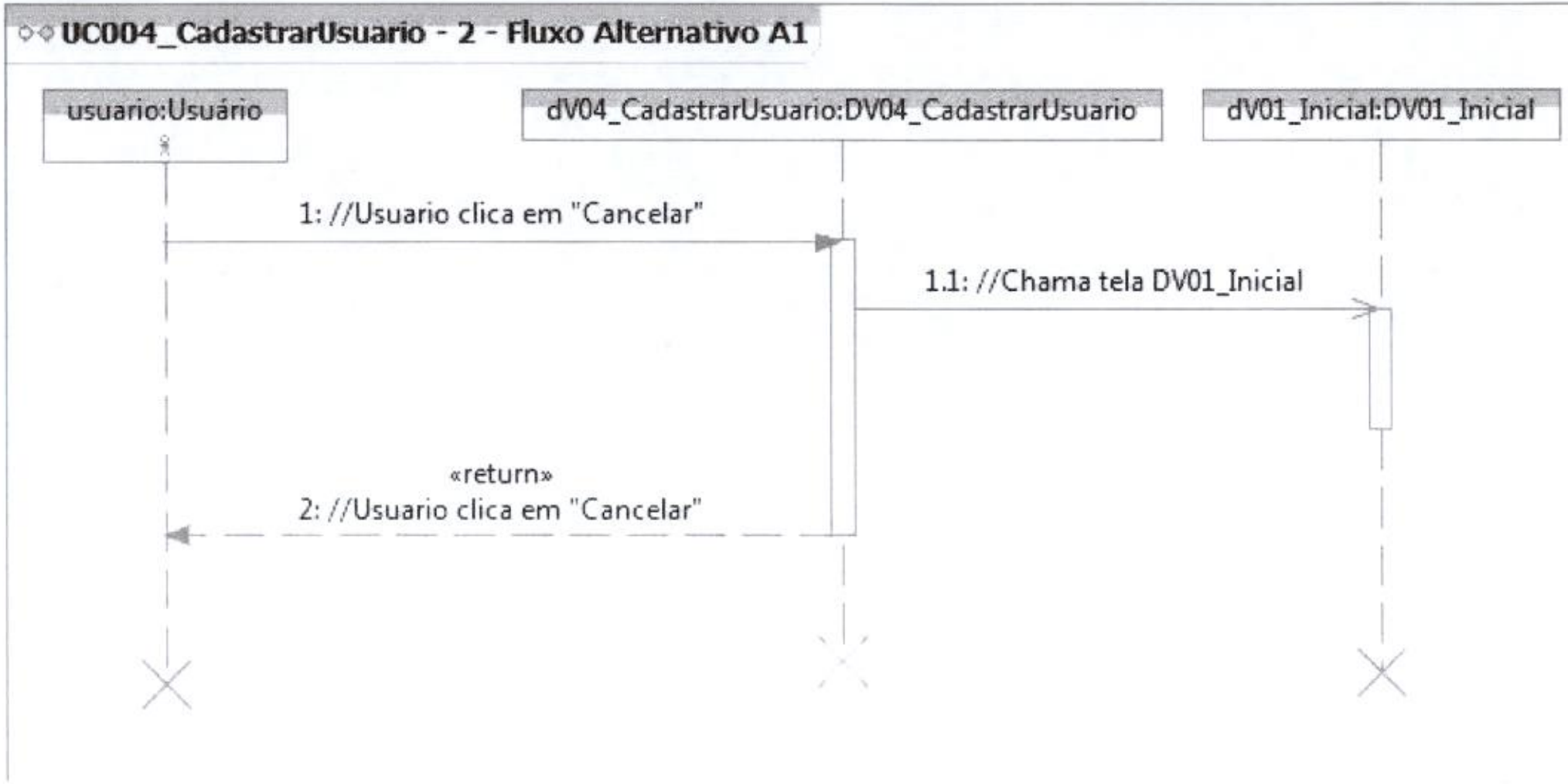

UC003_EsqueciSenha - 2 - Fluxo Alternativo A1
usuario:Usuário
dV03_SolicitarSenha:DV03_SolicitarSenha
dV01_Inicial:DV01_Inicial
1: //Usuario clica em "Cancelar"
1.1: //Chama tela DV01_Inicial
«return»
2: //Usuario clica em "Cancelar"

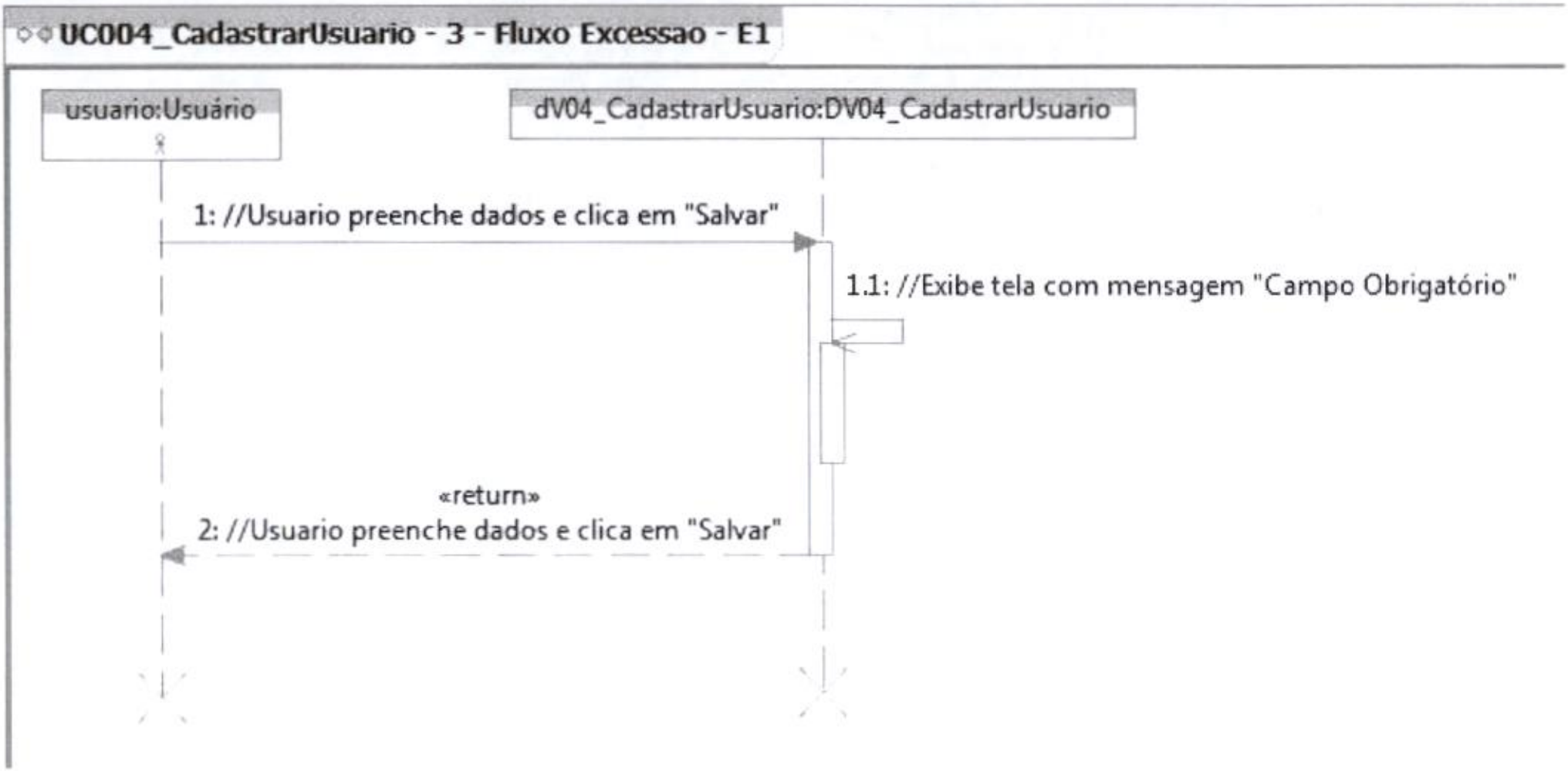

UC003_EsqueciSenha - 3 - Fluxo Excessao E1
usuario:Usuário
dV03_SolicitarSenha:DV03_SolicitarSenha
1: //Usuario preenche dados e clica em "Solicitar"
«return»
2: //Usuario preenche dados e clica em "Solicitar"

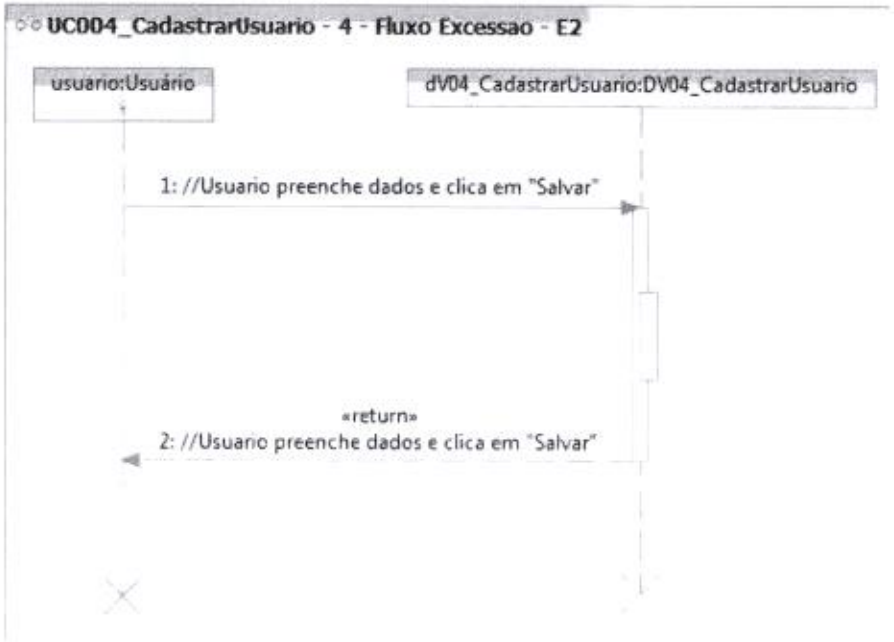

UC003_EsqueciSenha - 4 - Fluxo Excessao E2
usuario:Usuário
dV03_SolicitarSenha:DV03_SolicitarSenha
1: //Usuario preenche dados e clica em "Solicitar"
«return»
2: //Usuario preenche dados e clica em "Solicitar"

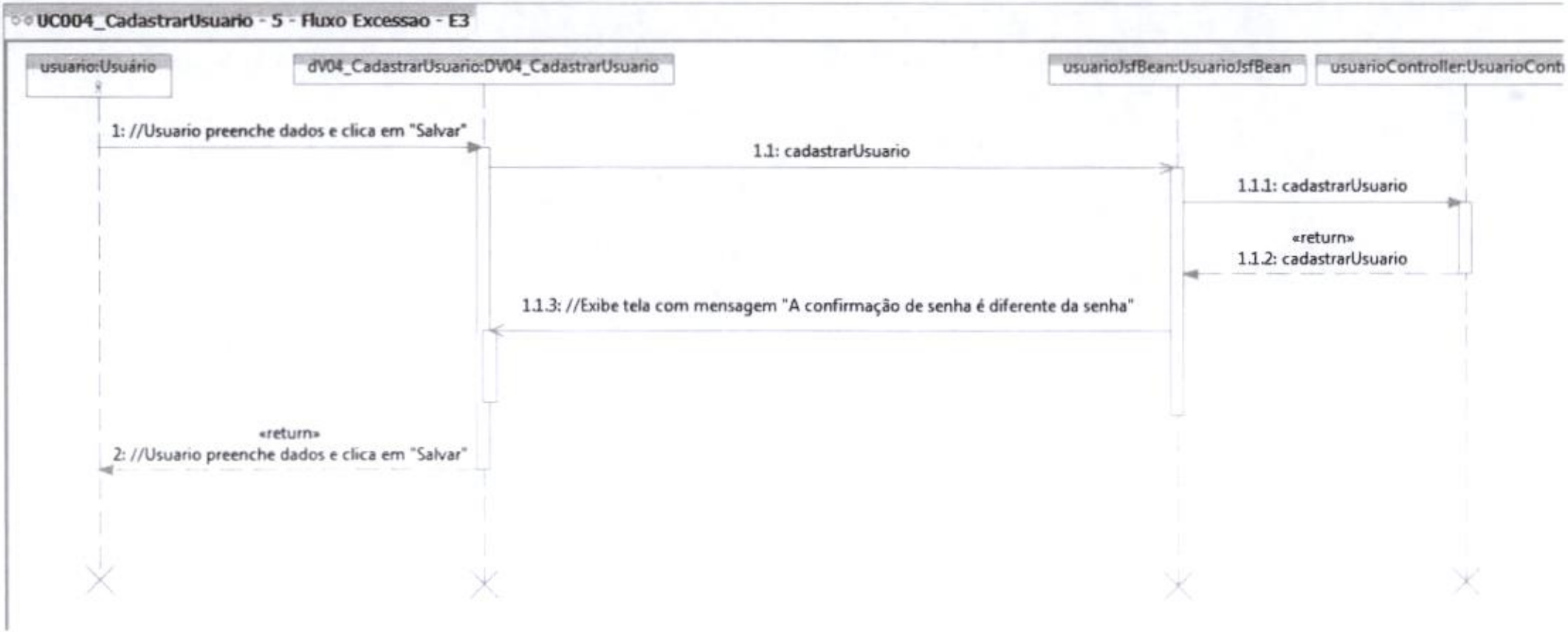

UC003_EsqueciSenha - 5 - Fluxo Excessao E3
usuario:Usuario
dV03_SolicitarSenha:DV03_SolicitarSenha
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario preenche dados e clica em "Solicitar"
1.1: esqueciSenha
1.1.1: esqueciSenha
1.1.1.1: esqueciSenha
«return»
1.1.1.2: esqueciSenha
«return»
1.1.2: esqueciSenha
1.1.3: //Exibe tela com mensagem "Usuario não encontrado com os dados abaixo"
«return»
2: //Usuario preenche dados e clica em "Solicitar"

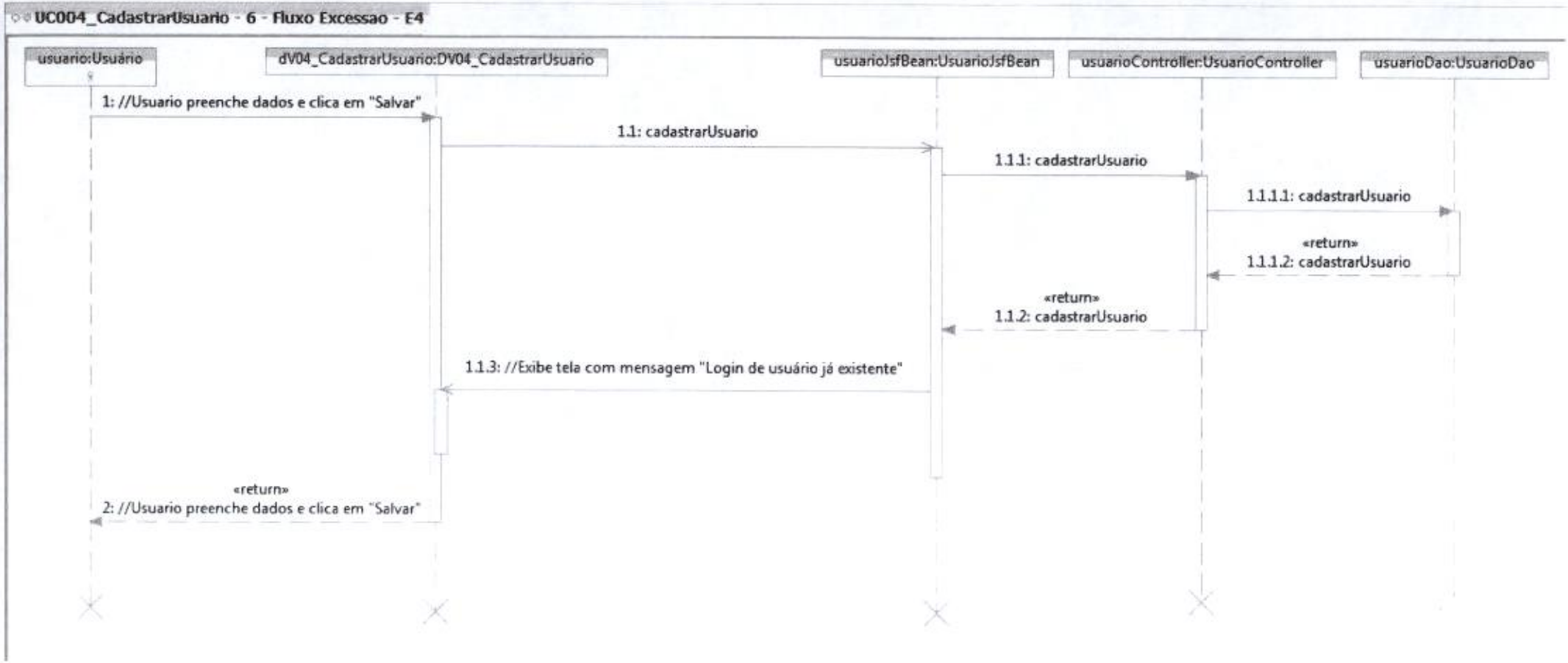

UC004_CadastrarUsuario - 1 - Fluxo Principal
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
usuarioJsfBean:UsuarioJsfBean
dV01_Inicial:DV01_Inicial
usuarioController:UsuarioController
encriptaSenha:EncriptaSenha
usuarioDao:UsuarioDao
1: //Usuario preenche dados e clica em "Salvar"
1.1: cadastrarUsuario
1.1.1: cadastrarUsuario
1.1.1.1: encripta
«return»
1.1.1.2: encripta
1.1.1.3: cadastrarUsuario
«return»
1.1.1.4: cadastrarUsuario
«return»
1.1.2: cadastrarUsuario
1.1.3: //Chama tela DV01_Inicial

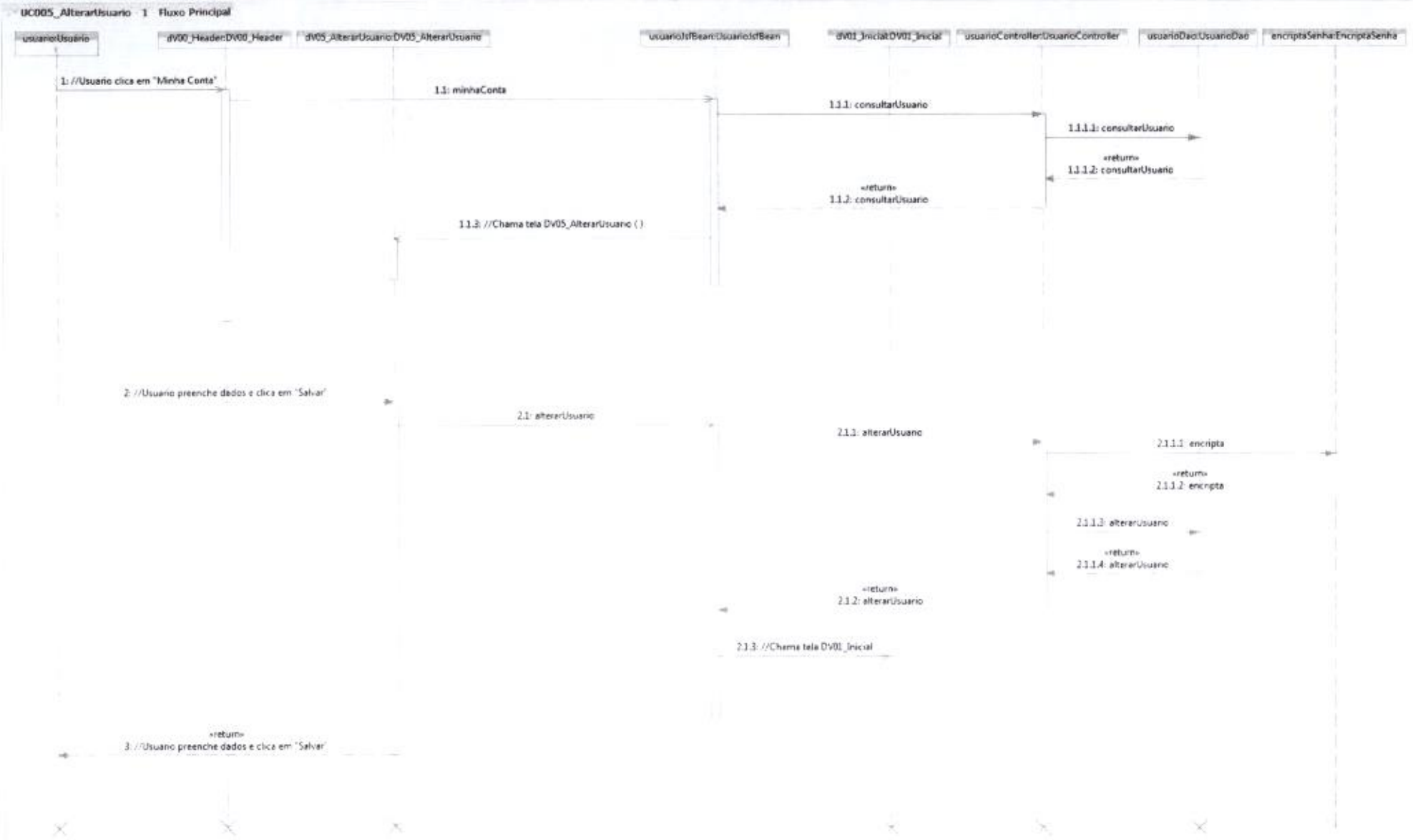

UC004_CadastrarUsuario - 2 - Fluxo Alternativo A1
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
dV01_Inicial:DV01_Inicial
1: //Usuario clica em "Cancelar"
1.1: //Chama tela DV01_Inicial
«return»
2: //Usuario clica em "Cancelar"

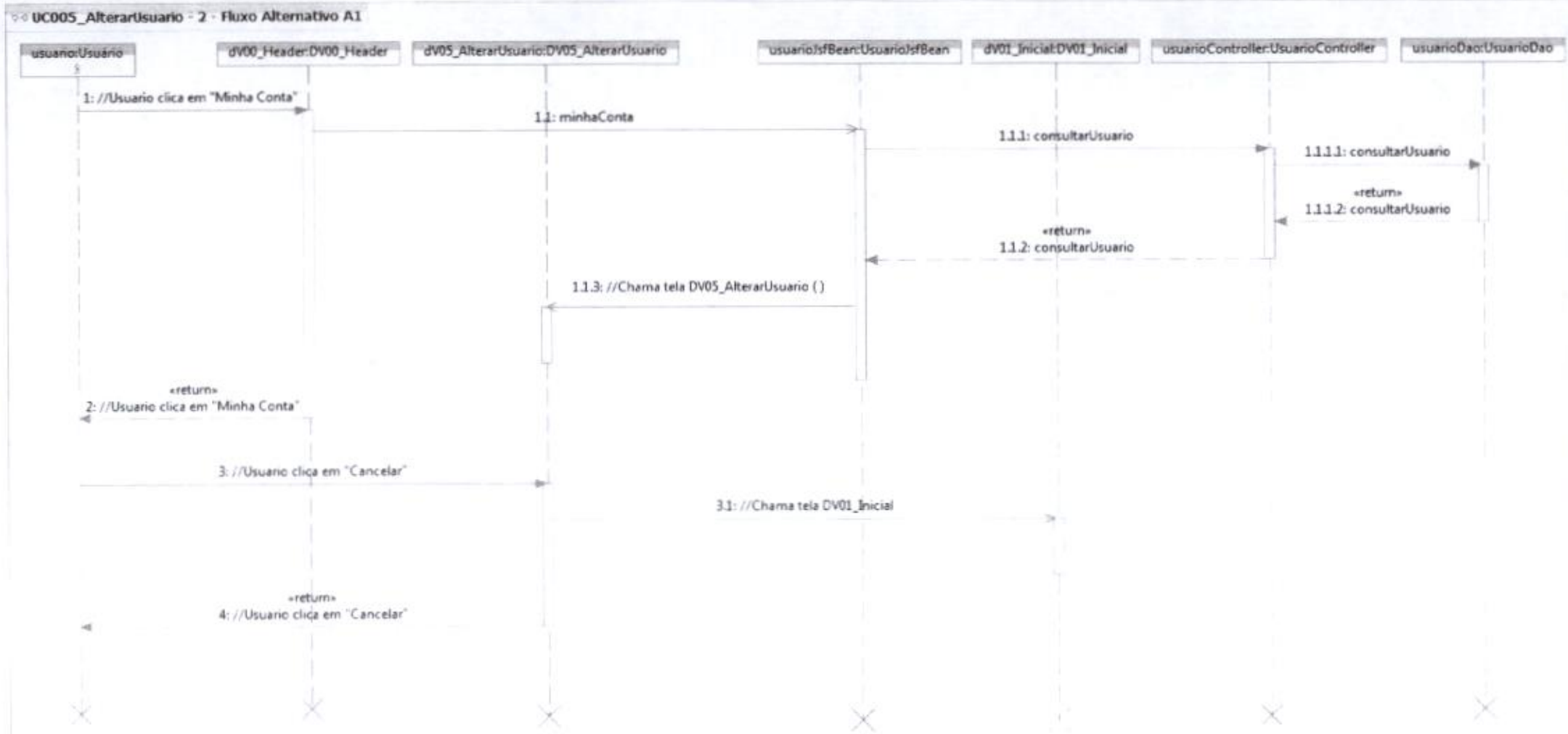

UC004_CadastrarUsuario - 3 - Fluxo Excessao - E1
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
1: //Usuario preenche dados e clica em "Salvar"
1.1: //Exibe tela com mensagem "Campo Obrigatório"
«return»
2: //Usuario preenche dados e clica em "Salvar"


UC004_CadastrarUsuario - 4 - Fluxo Excessao - E2
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
1: //Usuario preenche dados e clica em "Salvar"
«return»
2: //Usuario preenche dados e clica em "Salvar"

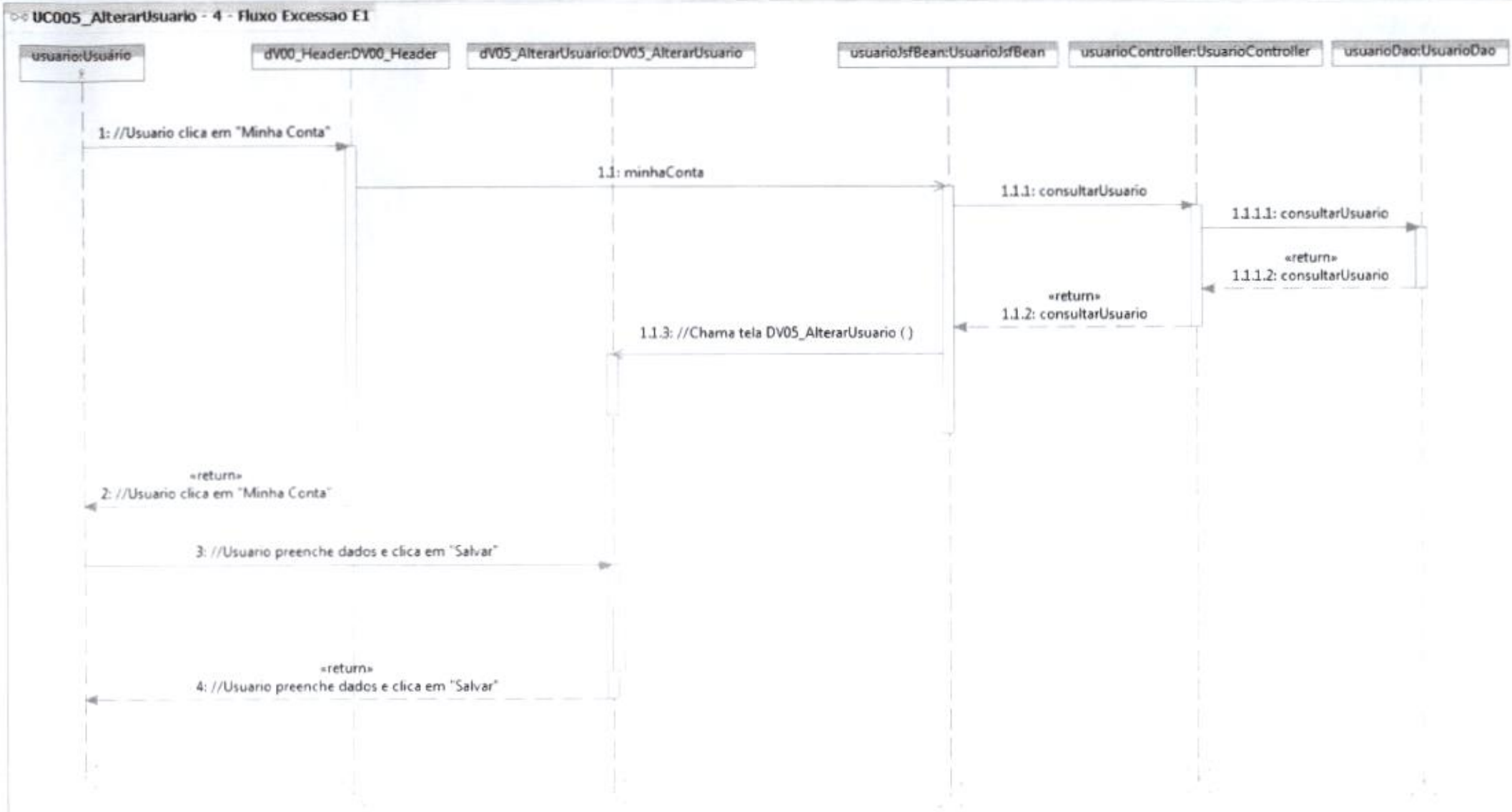

UC004_CadastrarUsuario - 5 - Fluxo Excessao - E3
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioConti
1: //Usuario preenche dados e clica em "Salvar"
1.1: cadastrarUsuario
1.1.1: cadastrarUsuario
«return»
1.1.2: cadastrarUsuario
1.1.3: //Exibe tela com mensagem "A confirmação de senha é diferente da senha"
«return»
2: //Usuario preenche dados e clica em "Salvar"

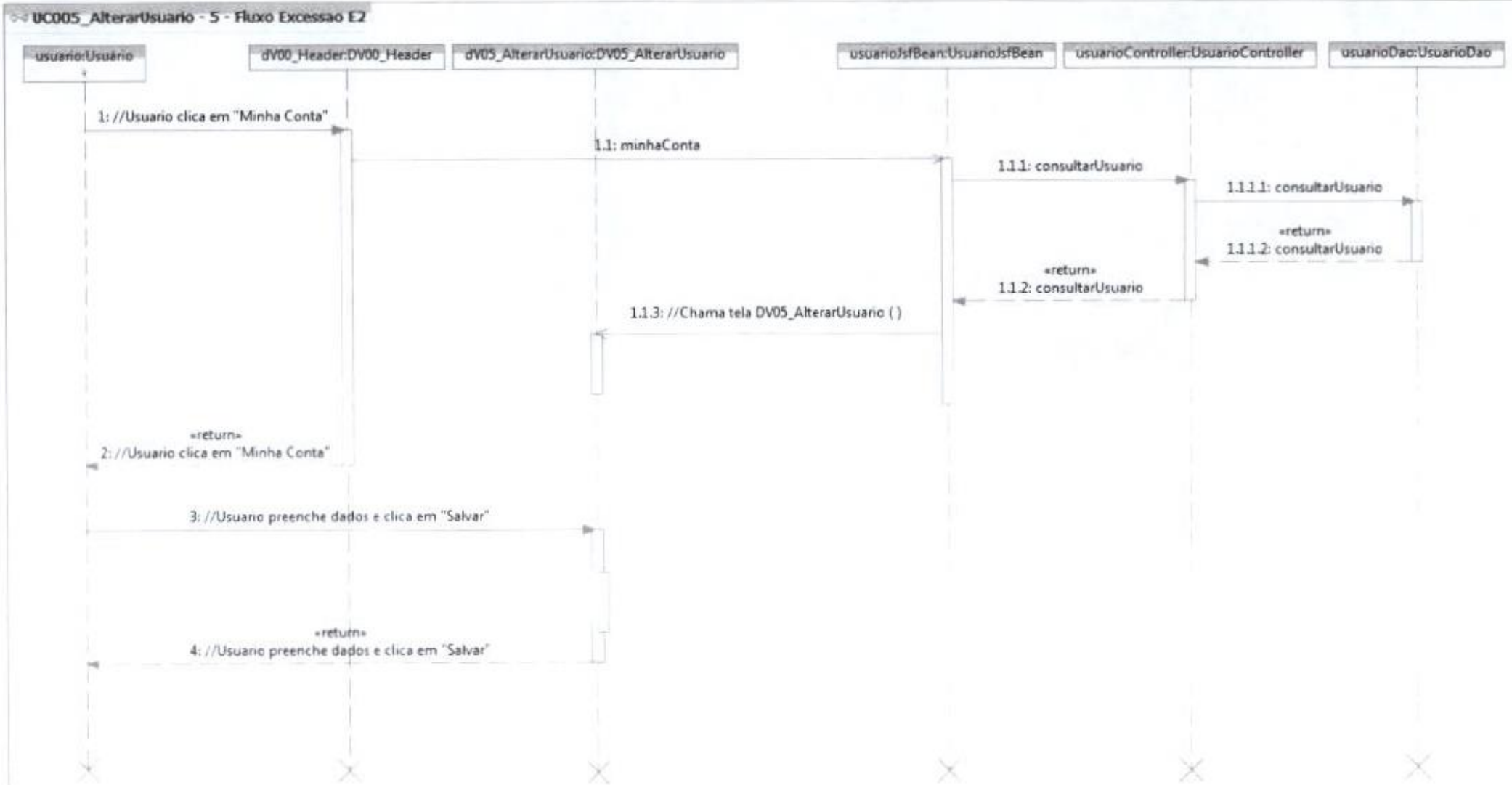

UC004_CadastrarUsuario - 6 - Fluxo Excessao - E4
usuario:Usuário
dV04_CadastrarUsuario:DV04_CadastrarUsuario
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario preenche dados e clica em "Salvar"
1.1: cadastrarUsuario
1.1.1: cadastrarUsuario
1.1.1.1: cadastrarUsuario
«return»
1.1.1.2: cadastrarUsuario
«return»
1.1.2: cadastrarUsuario
1.1.3: //Exibe tela com mensagem "Login de usuário já existente"
«return»
2: //Usuario preenche dados e clica em "Salvar"


UC005_AlterarUsuario 1 Fluxo Principal
usuario:Usuario
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
dV01_Inicial:DV01_Inicial
usuarioController:UsuarioController
usuarioDao:UsuarioDao
encriptaSenha:EncriptaSenha
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
2: //Usuario preenche dados e clica em "Salvar"
2.1: alterarUsuario
2.1.1: alterarUsuario
2.1.1.1: encripta
«return»
2.1.1.2: encripta
2.1.1.3: alterarUsuario
«return»
2.1.1.4: alterarUsuario
«return»
2.1.2: alterarUsuario
2.1.3: //Chama tela DV01_Inicial
«return»
3: //Usuario preenche dados e clica em "Salvar"

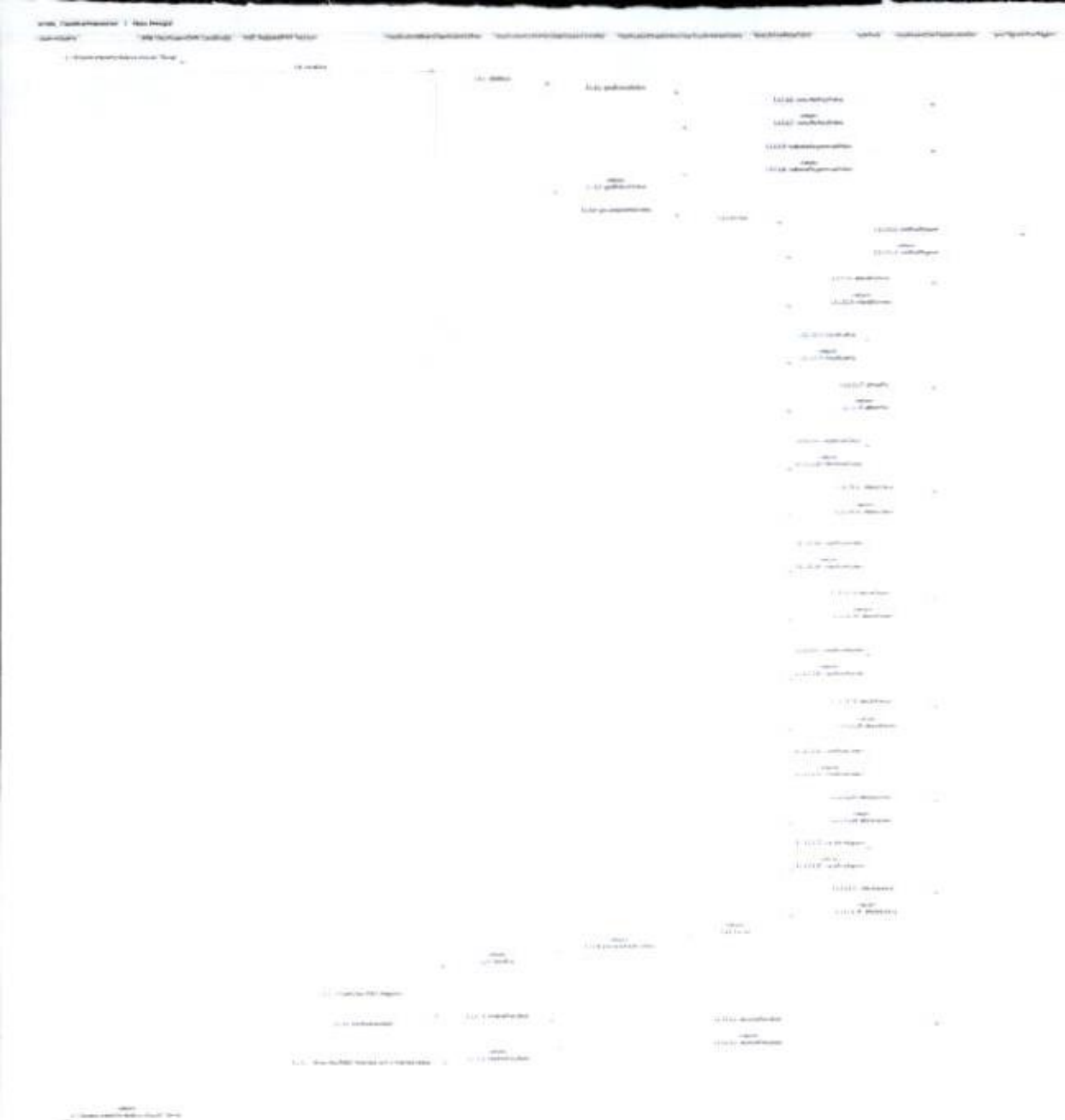

UC005_AlterarUsuario - 2 - Fluxo Alternativo A1
usuario:Usuário
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
dV01_Inicial:DV01_Inicial
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
«return»
2: //Usuario clica em "Minha Conta"
3: //Usuario clica em "Cancelar"
3.1: //Chama tela DV01_Inicial
«return»
4: //Usuario clica em "Cancelar"

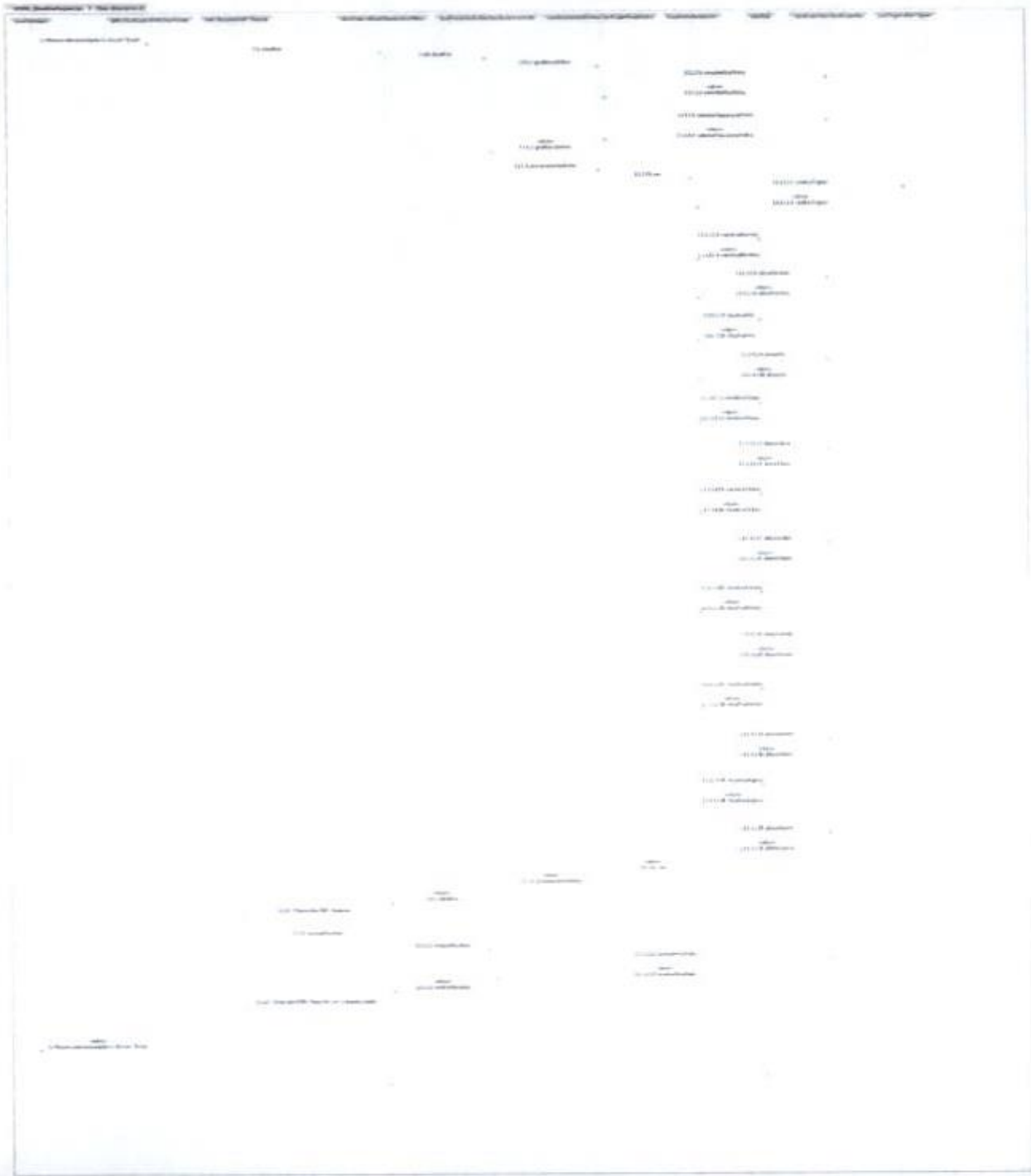

UC005_AlterarUsuario - 3 - Fluxo Alternativo A2
usuario:Usuario
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
dV01_Inicial:DV01_Inicial
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
«return»
2: //Usuario clica em "Minha Conta"
3: //Usuario clica em "Excluir"
3.1: excluirUsuario
3.1.1: excluirUsuario
3.1.1.1: excluirUsuario
«return»
3.1.1.2: excluirUsuario
«return»
3.1.2: excluirUsuario
3.1.3: //Chama tela DV01_Inicial
«return»
4: //Usuario clica em "Excluir"

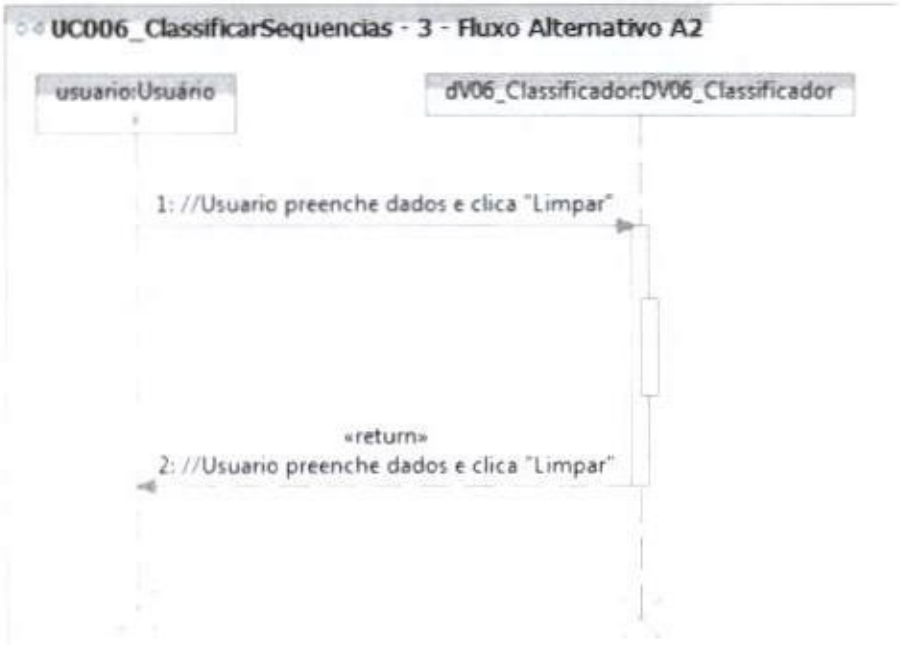

UC005_AlterarUsuario - 4 - Fluxo Excessao E1
usuario:Usuário
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
«return»
2: //Usuario clica em "Minha Conta"
3: //Usuario preenche dados e clica em "Salvar"
«return»
4: //Usuario preenche dados e clica em "Salvar"

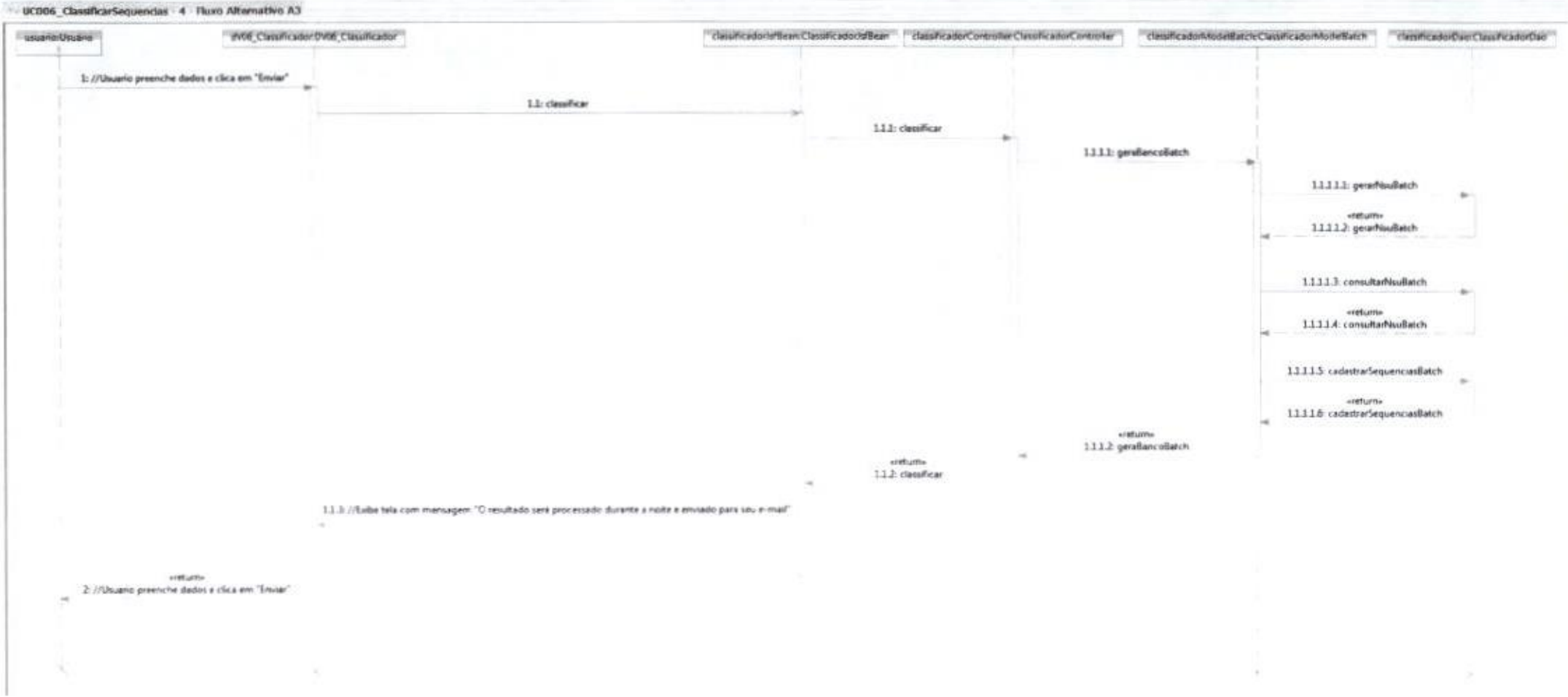

UC005_AlterarUsuario - 5 - Fluxo Excessao E2
usuario:Usuario
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioController
usuarioDao:UsuarioDao
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
«return»
2: //Usuario clica em "Minha Conta"
3: //Usuario preenche dados e clica em "Salvar"
«return»
4: //Usuario preenche dados e clica em "Salvar"


UC005_AlterarUsuario - 6 - Fluxo Excessao E3
usuario:Usuario
dV00_Header:DV00_Header
dV05_AlterarUsuario:DV05_AlterarUsuario
usuarioJsfBean:UsuarioJsfBean
usuarioController:UsuarioController
UsuarioDao:UsuarioDao
1: //Usuario clica em "Minha Conta"
1.1: minhaConta
1.1.1: consultarUsuario
1.1.1.1: consultarUsuario
«return»
1.1.1.2: consultarUsuario
«return»
1.1.2: consultarUsuario
1.1.3: //Chama tela DV05_AlterarUsuario ( )
«return»
2: //Usuario clica em "Minha Conta"
3: //Usuario preenche dados e clica em "Salvar"
3.1: alterarUsuario
3.1.1: alterarUsuario
«return»
3.1.2: alterarUsuario
3.1.3: //Exibe tela com mensagem "A confirmação de senha e diferente da senha"
«return»
4: //Usuario preenche dados e clica em "Salvar"

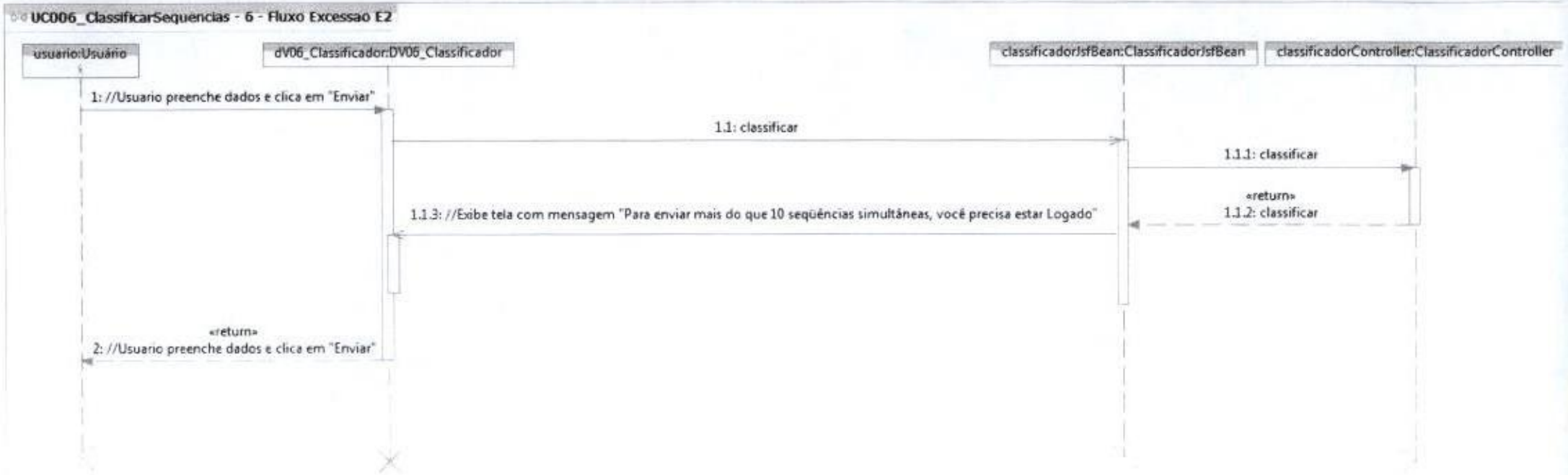


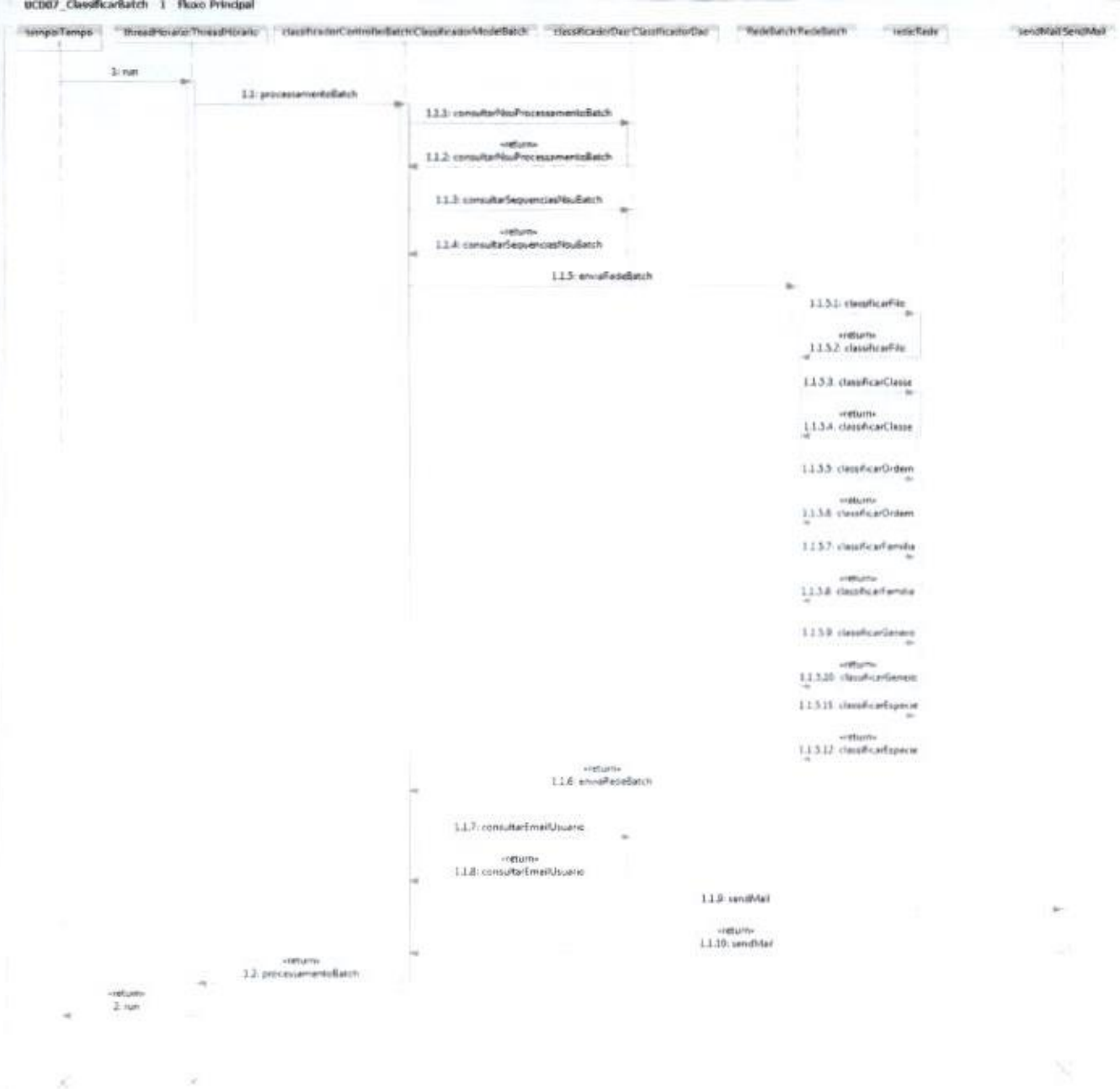



UC006_ClassificarSequencias - 3 - Fluxo Alternativo A2
usuario:Usuário
dV06_Classificador:DV06_Classificador
1: //Usuario preenche dados e clica "Limpar"
«return»
2: //Usuario preenche dados e clica "Limpar"

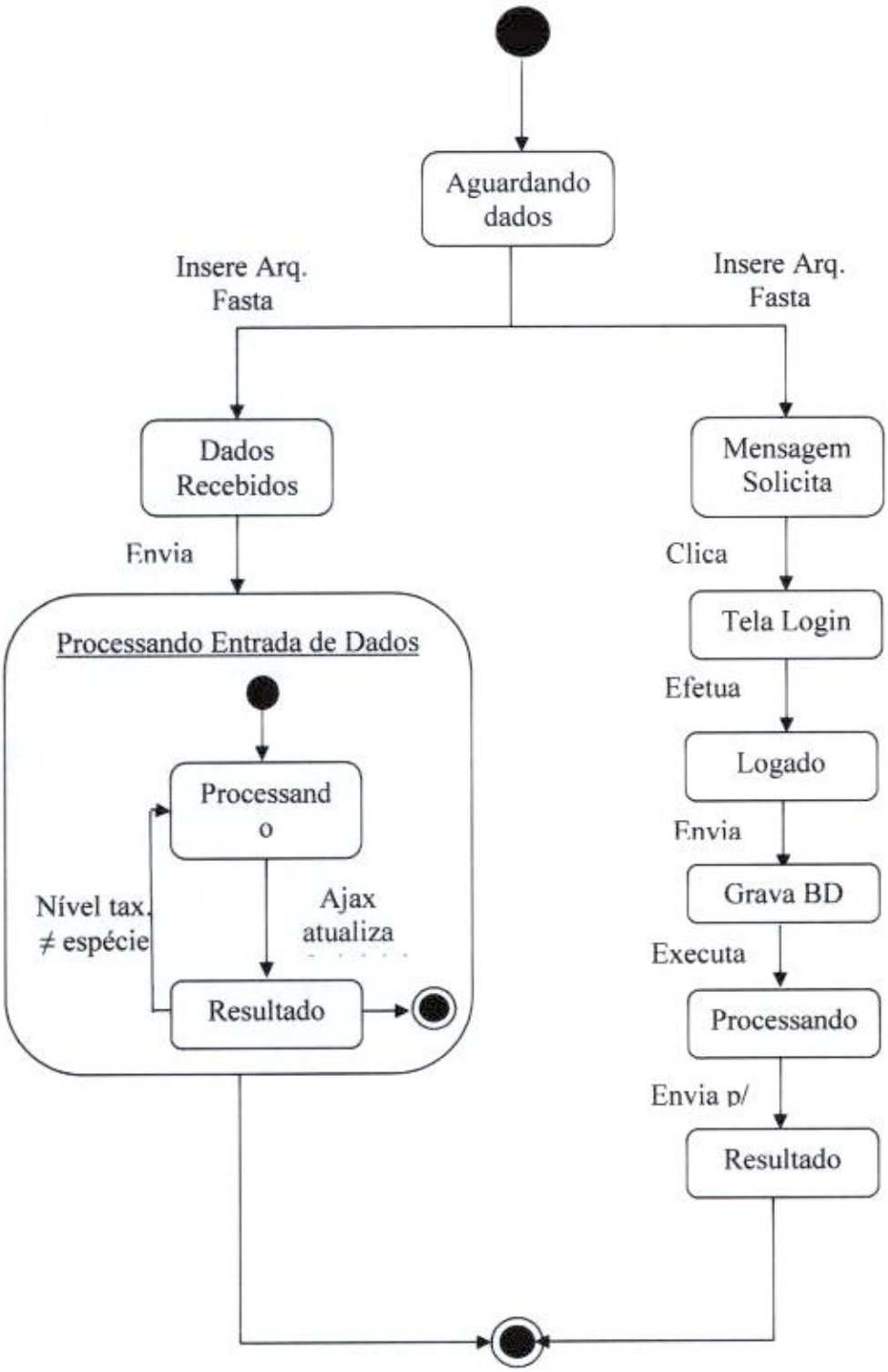

UC006_ClassificarSequencias - 4 - Fluxo Alternativo A3
usuario:Usuario
tV06_Classificador:tV06_Classificador
classificadorIefBean:ClassificadorIefBean
classificadorController:ClassificadorController
classificadorModelBatch:ClassificadorModelBatch
classificadorDao:ClassificadorDao
1: //Usuario preenche dados e clica em "Enviar"
1.1: classificar
1.1.1: classificar
1.1.1.1: geraBancoBatch
1.1.1.1.1: geraNsuBatch
«return»
1.1.1.1.2: geraNsuBatch
1.1.1.1.3: consultarNsuBatch
«return»
1.1.1.1.4: consultarNsuBatch
1.1.1.1.5: cadastrarSequenciasBatch
«return»
1.1.1.1.6: cadastrarSequenciasBatch
«return»
1.1.1.2: geraBancoBatch
«return»
1.1.2: classificar
1.1.3: //Exibe tela com mensagem: "O resultado será processado durante a noite e enviado para seu e-mail"
«return»
2: //Usuario preenche dados e clica em "Enviar"


UC006_ClassificarSequencias - 5 - Fluxo Excessao E1
usuario:Usuário
dv06_Classificador:DV06_Classificador
classificadorJsfBean:ClassificadorJsfBean
classificadorController:ClassificadorController
1: //Usuario preenche dados e clica em "Enviar"
1.1: classificar
1.1.1: classificar
«return»
1.1.2: classificar
1.1.3: Exibe tela DV06_Classificador com mensagem "formato do arquivo é inválido, certifique-se que é um arquivo fasta"
«return»
2: //Usuario preenche dados e clica em "Enviar"

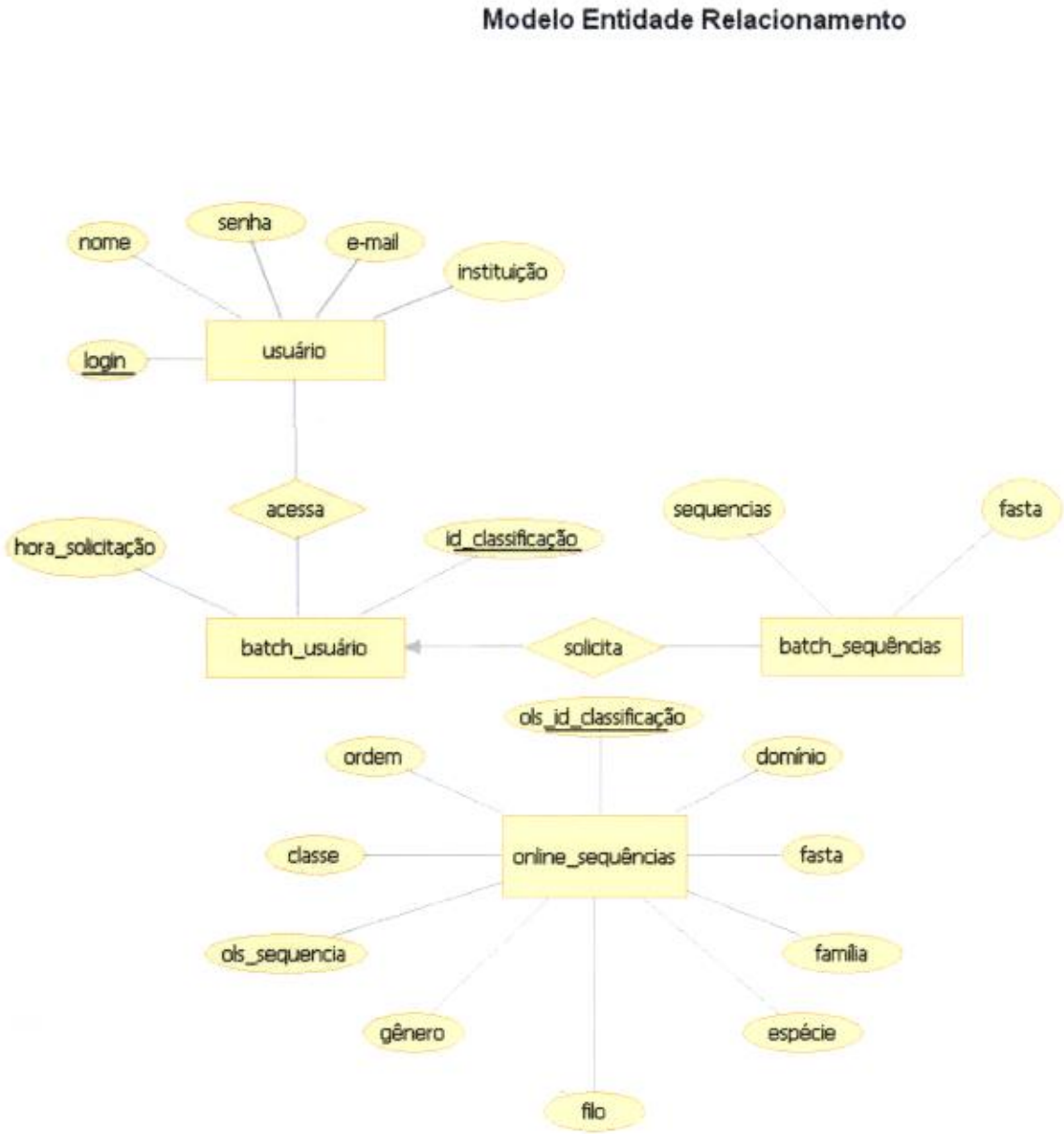

UC006_ClassificarSequencias - 6 - Fluxo Excessao E2
usuario:Usuário
dV06_Classificador:DV06_Classificador
classificadorJsfBean:ClassificadorJsfBean
classificadorController:ClassificadorController
1: //Usuario preenche dados e clica em "Enviar"
1.1: classificar
1.1.1: classificar
«return»
1.1.2: classificar
1.1.3: //Exibe tela com mensagem "Para enviar mais do que 10 seqüências simultâneas, você precisa estar Logado"
«return»
2: //Usuario preenche dados e clica em "Enviar"


UC007_ClassificarBatch : Fluxo Principal
sompoTempo
ThreadHorario:ThreadHorario
classificarControleBatch:ClassificadorModeloBatch
classificarDao:ClassificadorDao
RedeBatch:RedeBatch
redeRede
sendMail:SendMail
1: run
1.1: processamentoBatch
1.1.1: consultarNovoProcessamentoBatch
«return»
1.1.2: consultarNovoProcessamentoBatch
1.1.3: consultarSequenciasNovoBatch
«return»
1.1.4: consultarSequenciasNovoBatch
1.1.5: enviaRedeBatch
1.1.5.1: classificarFilo
«return»
1.1.5.2: classificarFilo
1.1.5.3: classificarClasse
«return»
1.1.5.4: classificarClasse
1.1.5.5: classificarOrdem
«return»
1.1.5.6: classificarOrdem
1.1.5.7: classificarFamilia
«return»
1.1.5.8: classificarFamilia
1.1.5.9: classificarGenero
«return»
1.1.5.10: classificarGenero
1.1.5.11: classificarEspecie
«return»
1.1.5.12: classificarEspecie
«return»
1.1.6: enviaRedeBatch
1.1.7: consultarEmailUsuario
«return»
1.1.8: consultarEmailUsuario
1.1.9: sendMail
«return»
1.1.10: sendMail
«return»
1.2: processamentoBatch
«return»
2: run

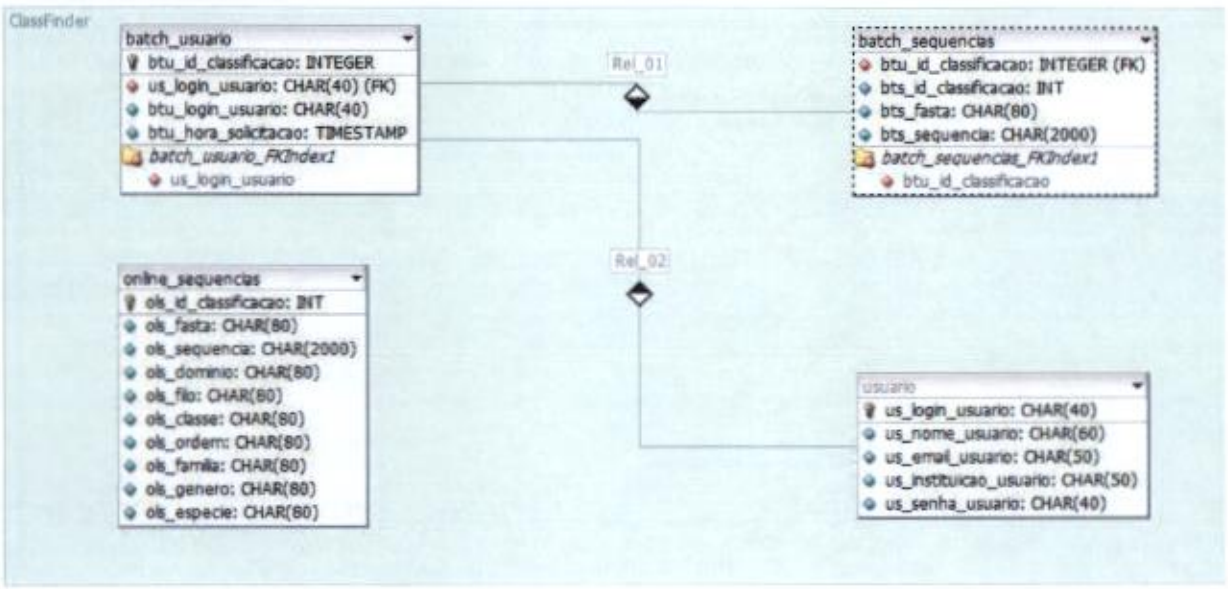
11.1.5 Diagramas de estados



11.1.6 Diagrama de Entidade de Relacionamento



11.1.7 Diagrama Relacional



11.2 Documentação do sistema classfinder

11.2.1 Requisitos Mínimos para Instalação como Servidor

Sistema Operacional Windows XP ou Superior

Servidor Web com seguintes softwares:

PostgreSQL versão 8.3.0 ou superior

Apache Tomcat versão 6.0.16 ou superior

JDK versão 1.6.014 ou superior

11.2.2 Requisitos Mínimos para Usuário Cliente

Sistema com Navegador Web com Javascript habilitado

11.2.3 Instruções para instalação

- Copiar o arquivo classfinder.war para a pasta webapps, que se encontra no diretório default de instalação do servidor Tomcat.
- Copiar o arquivo login.config para o diretório padrão do Tomcat
- Incluir comando para ser executado na inicialização do Tomcat:
- Djava.security.auth.login.config="local onde foi colocado o arquivo"\login.config
- Importar banco de dados taxonomia.sql
- Reiniciar o servidor Tomcat

12 PLANO GERAL DO PROJETO

12.1 Escopo e Propósito do Documento

Pretende-se desenvolver neste projeto uma metodologia de classificação de seqüências de 16S, com base na taxonomia de procariotos encontrada no manual Bergey's Taxonomic Outline (http://dx.doi.org/10.1007/bergeysoutline200310). As características serão extraídas da molécula de 16S rDNA das bactérias utilizando-se a metodologia de extração trigram. Estas características serão utilizadas no treinamento de Redes Neurais Artificiais (FAN – Raittz, 2002), que fazem a classificação destas seqüências codificadas através dos níveis taxonômicos contidos no manual Bergey (domínio. Filo, classe, ordem, família, gênero e espécie). O sistema aceitará como entrada um arquivo no formato fasta (ASC II) contendo uma ou mais seqüências, com um limite inicial de 50 seqüências. A saída deste sistema será uma página HTML revelando a hierarquia taxonômica. E quando possível a identidade da espécie.

12.2 Objetivos do Projeto

Analisar e reconhecer padrões em seqüências de 16Sr DNA para que se obtenha a classificação taxonômica bacteriana com base nestas seqüências.

12.2.1 Objetivos

O sistema de classificação bacteriana utilizará redes neurais artificiais para classificar seqüências de 16S rDNA de acordo com os níveis taxonômicos descritos no Manual Bergey, utilizado como referência na área de microbiologia bacteriana.

12.3 Funções Principais

A principal função do sistema de classificação bacteriana é revelar a hierarquia taxonômica e a identidade da espécie a que pertence esta seqüência.

O usuário do sistema entra com um arquivo no formato fasta, contendo uma ou mais seqüências de 16S rDNA,

O sistema fornecerá a hierarquia a que pertencem as seqüências previamente informadas. Se, e somente se a qualidade destas seqüências for muito boa e elas estiverem completas, ou seja, pelo menos 1500 pares de bases, o usuário então pode ter uma saída contendo a espécie a que pertencem estas seqüências.

12.4 Questões de Desempenho

O desempenho deste projeto está intimamente relacionado ao sucesso e a rapidez com que poderemos obter as redes treinadas na classificação de bactérias. Para tanto contaremos com um servidor de redes neurais compatível com o padrão SQL para o treinamento em bloco destas seqüências, dentro de cada nível taxonômico. Este servidor chama-se Sibila e está sendo desenvolvido pelo Professor Dieval Guizelini, da Escola Técnica da UFPR. Podemos imaginar a hierarquia taxonômica bacteriana como uma grande árvore, na qual os nós internos são os níveis taxonômicos superiores, por exemplo o filo, a classe, a ordem, a família e o gênero e os nós chamados folha (os terminais) são o nível específico. Quanto à interface do sistema planejamos desenvolvê-la utilizando Framework Java Server Faces.

12.4.1 Restrições Técnicas e Administrativas

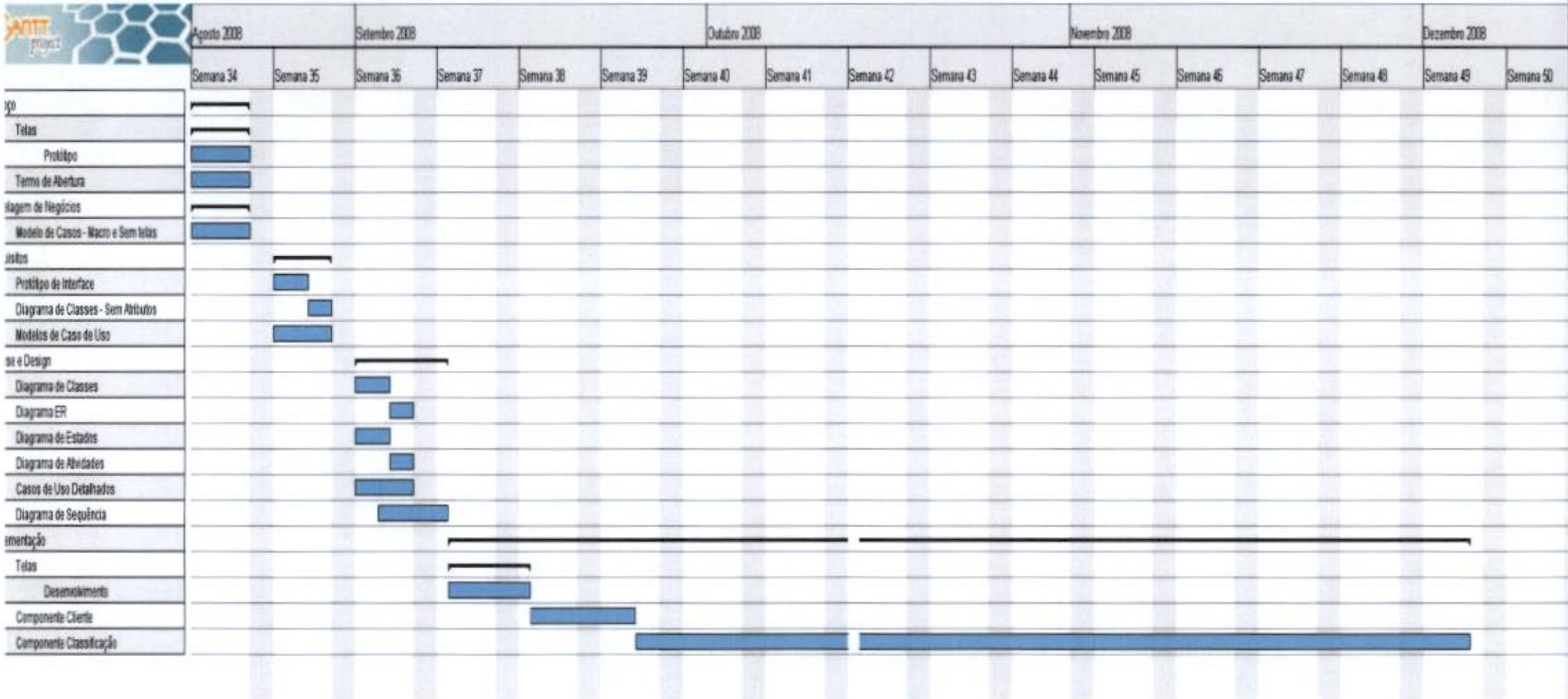
Neste projeto será utilizado o sistema de codificação trigram de seqüências de 16S rDNA e extração de características . Estas características serão utilizadas para o

treinamento de redes neurais artificiais (Raittz, 2002) para o reconhecimento de padrões e classificação de estas seqüências em níveis taxonômicos. A interface do sistema será construída com JSF, HTML e CSS. O processamento das seqüências submetidas e das previamente processadas e treinadas terá no desenvolvimento de classes Java e a utilização de JSF como principais ferramentas. Há a possibilidade de se utilizar módulos que não estejam diretamente relacionados com a tecnologia Java, pois existem atualmente módulos escritos para a linguagem Perl que foram desenvolvidos especificamente para serem utilizados no processamento de dados biológicos. Este conjunto de módulos recebeu o nome de BioPerl e é mantido pela Open Bioinformatics. O software Matlab será utilizado, principalmente na fase de reconhecimento de pardrões, pois a toolbox Bioinformatics disponíveis neste software foi desenvolvida para processar seqüências e outras toolbox são excelentes para o reconhecimento de padrões.

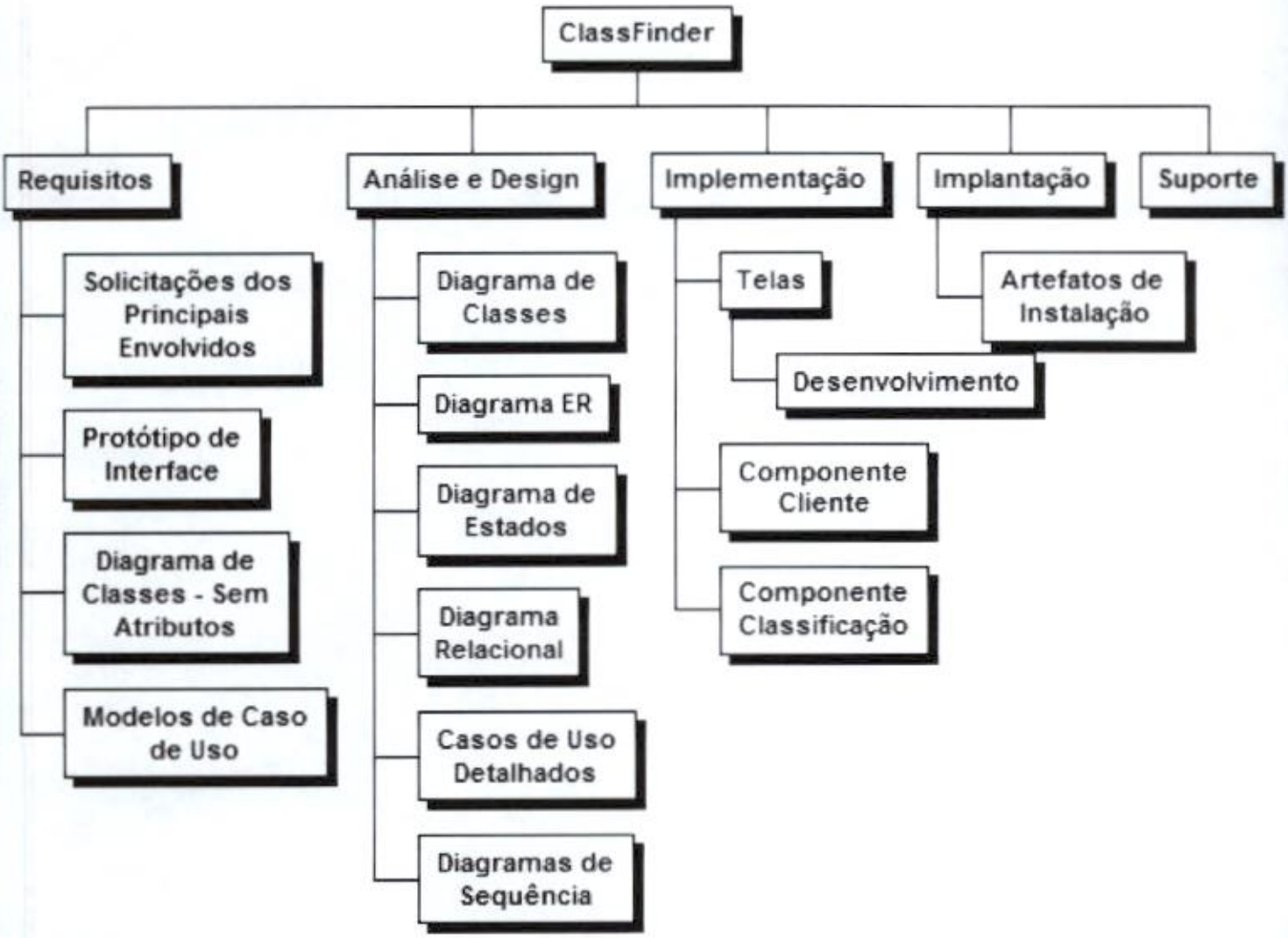
Quanto à documentação, os diagramas da UML serão utilizados para documentar todo o sistema, além de práticas e documentos da disciplina de gerenciamento de projetos e engenharia de software, como o RUP principalmente, além da consulta à documentação de gerência de projetos como o SWEBOK e PMBOK.

12.5 Cronograma

12.5.1 Gráfico de Gantt



12.6 Work Breakdown – Divisão de Trabalho no Projeto



13 Recursos do Projeto

13.1 Pessoal

A equipe de desenvolvimento deste projeto é composta pelos seguintes alunos do curso de Tecnologia em sistemas de Informação:

Daniel Lucas dos santos

Felipe Beni

Giovani Pisa

Marcio Mariano de Paula

Sandro Nascimento Lima

13.2 Hardware e software

O hardware a ser utilizado na realização deste projeto será composto principalmente pelos PCs portéteis dos integrantes, sendo que a configuração básica dos Laptops é a seguinte:

Processador Intel Core 2 Duo 1.7 GHz

Memória RAM de 2000 Mb

Hd 160 Gb

DVD-RW

Entradas USB (para conexão com Pen-drives)

Acesso a internet com conexão de 2000 Kbps