FRANCISCO RICARDO TABORDA AGUIAR
JOSÉ LEONARDO SIVEK
MOYSÉS KALEB RIBAS DE LIMA

ESCOLA TÉCNICA DA UFPR
BIBLIOTECA

SISTEMA ESPECIALISTA PARA A CRIAÇÃO DE EDITAIS DE CONCURSOS PÚBLICOS

Trabalho de Graduação apresentado à Disciplina de Trabalho de Conclusão de Curso, ministrada pelo orientador Prof. Irapuru Haruo e co-orientador Prof. Roberto Raittz, no Curso de Tecnologia em Informática da Universidade Federal do Paraná.
Setor: Escola Técnica

MG
005.1
A282
2007
ex.1

CURITIBA
2007

AGRADECIMENTOS

Primeiramente, agradecemos a Deus a energia positiva para enfrentar todos os obstáculos desta árdua caminhada e ao final cada um dizer com orgulho Eu Venci, Valeu a Pena Lutar.

A nossos familiares, pela paciência sabendo aceitar nossa ausência, incentivando-nos porque acreditam no nosso potencial. O apoio e carinho foram os fatores primordiais neste período de sacrifícios a fim de realizar um ideal esperado.

Aos orientadores e professores, pela orientação, acompanhamento, apoio e dedicação, indicando sempre os melhores caminhos para que pudéssemos chegar até o fim com sucesso, em especial aos professores Roberto, Irapuru e Mario.

Aos nossos amigos, pela amizade e companheirismo.

Aos especialistas Rosângela e principalmente Thais pelas informações e conhecimentos passados, fundamentais para o desenvolvimento do sistema.

A Universidade Federal do Paraná (UFPR) e a Escola Técnica, por proporcionarem o curso de Tecnologia em Informática e toda uma estrutura para a realização do projeto.

"Não são os grandes planos que dão certo,
são os pequenos detalhes"
Stethen Kanitz

RESUMO

O projeto trata-se de um sistema especialista capaz de gerar editais de concursos públicos, baseado em técnicas de inteligência artificial. O propósito do sistema é implementar funções que visam resolver problemas relacionados com a criação de um edital de concurso público para o núcleo de concursos da Universidade Federal do Paraná. O software utiliza uma relação de regras e frames encadeados que juntamente com uma interface amigável fazem determinadas perguntas sobre os editais a serem desenvolvidos e no fim do processo emitem um arquivo com o edital completamente montado e preenchido.

ABSTRACT

The project is an expert system able to generate proclamations of public competitions. The goal of the system is to implement functions that they aim at to decide problems related with the creation of a proclamation of public competition for the nucleus of competitions of the Federal University of the Paraná. Software uses a relation of chained rules and frames that together with a friendly interface make definitive questions on the proclamations to be developed and in the end of the process they emit an archive with the proclamation completely mounted and filled.

SUMÁRIO

AGRADECIMENTOS 1
RESUMO 3
ABSTRACT 4
LISTA DE SIGLAS 9
1 INTRODUÇÃO 10
2 MOTIVAÇÃO 11
3 DELIMITAÇÃO 12
4 METODOLOGIA DE PESQUISA 13
4.1 ESPECIFICAÇÃO DO PROBLEMA 13
4.1.1 Objetivo Geral 13
4.1.2 Objetivos Específicos 13
5 FUNDAMENTAÇÃO TEÓRICA 14
5.1 SISTEMAS ESPECIALISTAS 14
5.2 CONHECIMENTO 15
5.3 ARQUITETURA DE UM SISTEMA ESPECIALISTA 16
5.4 BASE DE CONHECIMENTO 16
5.5 MEMÓRIA DE TRABALHO 17
5.6 MÁQUINA DE INFERÊNCIA 17
6 DESENVOLVIMENTO DA PESQUISA 19
6.1 DESENVOLVIMENTO 19
6.2 FASES DO DESENVOLVIMENTO 19
6.3 DEFINIÇÃO DO PROBLEMA 20
6.3 DEFINIÇÃO DO PROBLEMA 21
6.4 ESPECIFICAÇÃO DO PRODUTO 21
6.5 AQUISIÇÃO DE CONHECIMENTO 21
6.6 PROJETO DA ARQUITETURA DO SISTEMA E DOS DADOS 22
6.7 CODIFICAÇÃO 22
6.8 TESTE 22
6.9 DOCUMENTAÇÃO 23
6.10 IMPLANTAÇÃO 23
6.11 MODELAGEM 23
6.12 MODELAGEM ORIENTADA A OBJETOS 23
6.13 LINGUAGEM DE MODELAGEM 25
6.14 LINGUAGEM DE PROGRAMAÇÃO 25
7 IMPLEMENTAÇÃO 28
8 MODELAGEM DO CONHECIMENTO 29
9 ARQUITETURA 34
10 FERRAMENTAS UTILIZADAS 35
Java 1.5 35
Eclipse 3.2 35
JESS 7.0 36
JUDE 3.0.3 36
Diagram Designer 1.17 36
Microsoft Project 36
Db Designer 4 36
Microsoft Office 2003 36
Acrobat Reader 5 37
11 DOCUMENTAÇÃO 38

12 TESTES 39
13 ANÁLISE DE RESULTADOS 40
13.1 RESULTADOS DA EQUIPE 40
13.2 RESULTADOS DO PROJETO 40
14 CONCLUSÃO 41
REFERÊNCIAS 42
DIAGRAMAS 44
DIAGRAMA DE CASOS DE USO 44
CASO DE USO CRIA EDITAL 44
Fluxo de Eventos 45
Fluxo Principal 45
Cenário Principal 45
Cenário Alternativo 46
DIAGRAMA DE CLASSES 46
DIAGRAMA DE ESTADOS 48
DIAGRAMA DE ENTIDADE E RELACIONAMENTO 48
FLUXOGRAMAS E REDES DE INFERÊNCIAS 50
FUNDAMENTAÇÃO TEÓRICA (AVANÇADA) 75
SISTEMAS ESPECIALISTAS 75
HISTÓRICO 75
COMPARAÇÃO ENTRE O CONHECIMENTO HUMANO E O CONHECIMENTO ARTIFICIAL 77
CONCEITOS 77
UTILIZAÇÃO 79
PRINCIPAIS BENEFÍCIOS 80
PROBLEMAS ENFRENTADOS 81
FERRAMENTAS PARA IMPLEMENTAÇÃO DE SE 81
SHELL de Sistemas Especialistas 81
CLIPS 83
APRESENTAÇÃO 84
CRÍTICA 85
JESS 86
APRESENTAÇÃO 86
CRÍTICA 87
ALGORITMO RETE 88
CONFIGURAÇÃO / INSTALAÇÃO DO SISTEMA 90
INSTALAÇÃO MANUAL 90
INSTALAÇÃO AUTOMÁTICA 90
MANUAL DO SISTEMA 91
CÓDIGO FONTE 94
Aplicativo.java 94
Assembler.java 94
Edital.java 99
EditalRecorder.java 104
Engine.java 106
TelaPrincipal.java 109
Background 116
CarregaFatos 118
Frames 119
Funções 121

Manager 128
SessionRules 141
Startup 149
BASE DE CONHECIMENTO 150
Afro 150
Cargos 157
Classificacao 163
Convocacao 167
DaInscricao 178
DeficientesFisicos 192
DispoGerais 200
DispoPreliminares 212
Inscricao 219
Introducao 234
Investidura 242
MetaRules 252
Modelo 286
Nomeacao 286
Provas 292
Recursos 305
Resultado 317
MEMÓRIA DE TRABALHO 326
JAVADOC – Padrão Sun 339
Class Aplicativo 339
Aplicativo 339
main 340
Class Assembler 341
Assembler 342
setFonte 342
montaEdital 342
setTexto 342
Class Edital 343
Edital 344
setNumItens 344
setNumParagrafos 344
setNumTopicos 344
getNumItens 345
getNumParagrafos 345
getNumTopicos 345
getTextoParagrafo 345
getTextoTopico 345
putTexto 345
salvaEdital 346
Class EditalRecorder 347
EditalRecorder 347
gravarEdital 348
fechaEdital 348
Class Engine 349
Engine 349
startup 350

rout 350
getTr 350
Class TelaPrincipal 351
TelaPrincipal 354
getTxtPergunta 355
getTxtResposta 355
getTxtExpert 355
getResposta 355
setTxtPergunta 355
actionPerformed 355
PERGUNTAS PARA O EDITAL UNIVERSAL 366

LISTA DE SIGLAS

AI	-	Artificial Intelligence
IA	-	Inteligência Artificial
JESS	-	Java Expert System Shell
NC-UFPR	-	Núcleo de Concursos da Universidade Federal do Paraná
SE	-	Sistema Especialista
UML	-	Unified Modeling Language

1 INTRODUÇÃO

O objetivo da disciplina de Trabalho de Conclusão de Curso é colocar em prática todos os conhecimentos adquiridos sobre o processo de desenvolvimento e gerenciamento de softwares durante o curso de Tecnologia em Informática, isso é feito através do desenvolvimento de um projeto selecionado pelas equipes juntamente com a colaboração de coordenadores e professores da área em áreas de real necessidade de desenvolvimento.

Durante o período de escolha do projeto, alguns temas foram surgindo para serem implementados e após um debate entre os integrantes da equipe e também com os orientadores de projeto uma proposta chamou maior atenção: desenvolver um sistema capaz de gerar com o menor esforço possível e sem falhas um edital de concurso público, haja visto que até o momento todo o trabalho era realizado manualmente e sem nenhuma ferramenta de apoio. A escolha do tema foi baseada no fato de ser uma aplicação totalmente nova na área e que seria uma poderosa ferramenta para o Núcleo de Concursos da Universidade Federal do Paraná conseguir criar seus editais com eficácia e eficiência. Além disso, o sistema acabou tornando-se um desafio para a equipe, pois para realizar o projeto seria necessário um grande conhecimento de sistemas especialistas e ainda pesquisar e compreender ferramentas que tornassem possível atender todos os requisitos.

2 MOTIVAÇÃO

A criação de editais de concursos públicos é uma tarefa extremamente trabalhosa. Demanda uma análise criteriosa em todos os aspectos jurídicos e institucionais que compreendem o concurso. Leis que garantem vagas especiais para afro-descendentes e deficientes físicos, inclusão social para candidatos oriundos de ensino público, conteúdos programáticos para as provas e as condições para que o candidato questione a respeito do critério utilizado para a correção, do conteúdo das questões e dos resultados das provas são apenas algumas facetas de um concurso público que devem estar claramente dispostos em um edital.

3 DELIMITAÇÃO

O trabalho terá como foco o processo de criação de editais de concursos públicos externos para o Núcleo de Concursos da Universidade Federal do Paraná (NC-UFPR).

O NC-UFPR busca criar os editais dentro do menor tempo possível, com o mínimo de esforço e com ausência de falhas, sendo que não há atualmente nenhuma ferramenta computacional que possa auxiliar de forma efetiva a este processo pois normalmente é realizado de forma quase que inteira manual e baseando-se em outros editais criados anteriormente.

4 METODOLOGIA DE PESQUISA

4.1 ESPECIFICAÇÃO DO PROBLEMA

4.1.1 Objetivo Geral

O objetivo geral do trabalho é desenvolver um sistema computacional que automatize e padronize a tarefa de criar editais de concursos públicos.

4.1.2 Objetivos Específicos

Para se atingir o objetivo geral do trabalho o sistema computacional deverá possuir o conhecimento e o raciocínio que um especialista humano detém com relação ao processo de criar editais.

É necessário criar um edital universal padrão para desenvolver o sistema, este edital deve ser capaz de incorporar todas as necessidades do especialista e abranger o máximo de possibilidades de criações de editais.

Com um edital padrão criado é preciso definir quais são as variáveis existentes para assim poder criar as perguntas e trabalhar as possíveis respostas do usuário

Esse conjunto de informações permite a criação de regras baseadas no raciocínio do especialista e um tratamento dos dados criando assim a base de conhecimento do sistema.

Através do menor número possível de perguntas deve ser capaz de interpretá-las e desenvolver o edital de maneira clara e sem deixar possíveis "brechas" para interpretações diferentes.

Outro aspecto específico a ser atingido diz respeito a desenvolver uma interface homem-máquina amigável. Isto quer dizer que o processo de interação entre o usuário e o sistema deve se dar uma linguagem bem parecida com a linguagem natural e de fácil compreensão

5 FUNDAMENTAÇÃO TEÓRICA

O trabalho será fundamentado em técnicas de Artificial Intelligence [1] (AI). DURKIN (1994, p. 3) define AI como sendo: "A field of study in computer science that pursues the goal of making a computer reason in a manner similar to humans." [2]

De acordo com JAMES (1985, p. 11): "[...] AI é desenvolvimento de software [...]" . Mas DURKIN (1994, p. 5), por sua vez, especifica um pouco mais sobre as contribuições que a AI pode oferecer para a ciência da computação:

> From a practical standpoint, the goal of AI is to make computers more useful for humans. This can be achieved by producing computer programs that assist human in decision making, intelligent information search, or simply by making computers easier to use with natural language interfaces. A second goal of AI, but an equally important one, is to better understand human intelligence. Building an intelligent computer system requires us to understand how humans capture, organize, and use knowledge during their problem solving.[3]

No contexto do desenvolvimento de um software que automatize o desenvolvimento de editais de concursos públicos, atividade esta que possui um especialista humano, a AI pode dar a sua contribuição através de técnicas para o desenvolvimento do programa, das quais afirma JAMES (1985, p. 63): "[...] têm recebido muita atenção da comunidade e do público em geral no momento, como uma das melhores coisas resultantes dos esforços em AI [...]" . Tais técnicas estão compreendidas em uma área de estudo da AI denominada: sistemas especialistas.

5.1 SISTEMAS ESPECIALISTAS

Sistemas especialistas correspondem a uma área da AI que trata do desenvolvimento de sistemas computacionais capazes de resolver problemas do

---

[1] Inteligência Artificial

[2] Um campo de estudo em ciências da computação que se dedica ao objetivo de fazer um computador raciocinar de maneira semelhante aos humanos.

[3] A partir de um ponto de vista prático, o objetivo da Inteligência Artificial é fazer computadores mais úteis para os humanos. Isto pode ser obtido através do desenvolvimento de programas de computadores que auxiliem os humanos nas tomadas de decisão, procura de informações inteligentes, ou simplesmente tornando os computadores mais fáceis para serem utilizados com interfaces de linguagem natural. O segundo objetivo da Inteligência Artificial, mas igualmente importante, é melhorar o entendimento acerca da inteligência humana. Construir um sistema computacional inteligente requer, ao desenvolvedor, um entendimento da maneira como os humanos capturam, organizam e usam o conhecimento enquanto resolvem um determinado problema.

mundo real através de métodos semelhantes aos utilizados pelos seres humanos. DURKIN (1994, p.7) dá a seguinte definição para sistemas especialistas: "A computer program designed to model the problem-solving ability of a human expert." [4]

O projeto de um sistema especialista deve estar voltado a codificar o conhecimento e o raciocínio de um especialista humano. Neste contexto o especialista humano torna-se um personagem de extrema importância para o desenvolvimento do software.

Uma característica importante de um sistema especialista é o seu alto grau de conhecimento sobre uma área bem específica. É o chamado domínio de conhecimento. DURKIN (1994, p. 53) define domínio como sendo: "A well-focused subject area." [5]

Um ponto importante a ser observado durante o projeto de um sistema especialista é o foco do domínio. Problemas complexos devem ser divididos em problemas menores e em domínios de conhecimento bem focados.

5.2 CONHECIMENTO

O conhecimento corresponde a um termo abstrato que busca identificar o entendimento que um indivíduo possui acerca de um determinado assunto. DURKIN (1994, p. 53) dá a seguinte definição para conhecimento: "Understanding of a subject area." [6]. Após a aquisição de conhecimento junto a um especialista da área se faz necessário codificar este conhecimento para o sistema especialista. A maneira como se estrutura o conhecimento em um sistema especialista define o que se chama de representação do conhecimento.

DURKIN (1994, p.53) afirma categoricamente: "Knowledge is power [...]" [7]. Esta declaração demonstra a importância do conhecimento para um sistema especialista. A qualidade do conhecimento que um sistema especialista possui a

---

[4] Um programa de computador projetado para modelar a habilidade de resolução de problemas de um especialista humano.

[5] Uma área de assunto bem focada.

[6] Entendimento de uma área objeto.

[7] Conhecimento é poder.

respeito de um determinado problema irá influenciar de forma decisiva na qualidade do software.

5.3 ARQUITETURA DE UM SISTEMA ESPECIALISTA

Um Sistema Especialista é formado basicamente pelos seguintes módulos: base de conhecimento, máquina de inferência e memória de trabalho. A figura 1 ilustra uma estrutura geral para um sistema especialista. Cabe aqui observar que a modelagem desta estrutura pode variar de acordo com o problema a ser tratado pelo sistema.

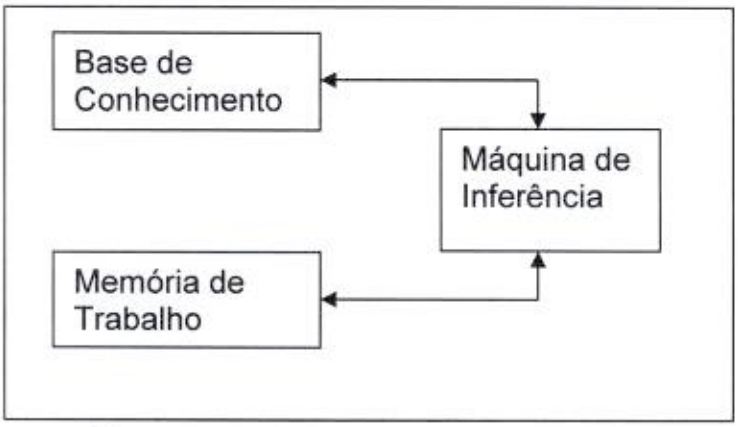


Figura 1 – Arquitetura de um Sistema Especialista

5.4 BASE DE CONHECIMENTO

DURKIN (1994, p. 29) define base de conhecimento como sendo: "Part of an expert system that contains the domain knowledge." [8]. Este módulo do sistema especialista é responsável por armazenar o conhecimento, o qual é altamente especializado.

Para se montar uma base de conhecimento se faz necessário a aquisição de conhecimento junto a um especialista humano e posteriormente codificar tal conhecimento adquirido. Segundo DURKIN (1994, p. 29): "One typical way of representing knowledge in an expert system is rules." [9]

---

[8] Parte de um sistema especialista que contém o domínio de conhecimento.

5.5 MEMÓRIA DE TRABALHO

A definição de DURKIN (1994, p. 29) para memória de trabalho é a seguinte: "Part of na expert system that contains the problem facts that are discovered during the session." [10] .

Durante uma interação o usuário fornece fatos ao sistema especialista sobre o problema. Estes fatos são armazenados na memória de trabalho. O sistema, então, busca associar estes fatos com o conhecimento armazenado na base de conhecimento para poder inferir novos fatos. Estes novos fatos, por sua vez, são, também, armazenados na memória de trabalho e o processo de consulta à base de conhecimento e interação com o usuário prossegue até que o sistema alcance alguma conclusão. Esta conclusão, também, é armazenada na memória de trabalho.

Diante desta situação, pode-se dizer que a memória de trabalho contém todas as informações sobre o problema, sejam elas fornecidas pelo usuário ou inferidas pelo sistema. Além disto, os sistemas especialistas podem, também, consultar informações externas (como aquelas armazenadas em bancos de dados, por exemplo) e armazená-las na memória de trabalho, podendo a interação ser exclusivamente desta forma.

5.6 MÁQUINA DE INFERÊNCIA

DURKIN (1994, p. 29) define máquina de inferência da seguinte forma: "Processor in an expert system that matches the facts contained in the working memory with the domain knowledge contained in the knowledge base, to draw conclusions about the problem." [11] . A máquina de inferência corresponde ao

---

[9] Uma maneira típica de se representar o conhecimento em um sistema especialista é através de regras.

[10] Parte de um sistema especialista que contém os fatos do problema que são discutidos ao longo da sessão.

[11] Processador em um sistema especialista que associa os fatos contidos na memória de trabalho com o domínio de conhecimento contido na base de conhecimento, para chegar a conclusões sobre o problema.

módulo do sistema que simula o raciocínio do especialista humano e fornece conclusões ou recomendações ao usuário. É a máquina de inferência quem estabelece a comunicação entre as informações contidas na memória de trabalho e o conhecimento armazenado na base de conhecimento para que novos fatos, conclusões ou recomendações sejam inferidos pelo sistema para o posterior armazenamento na memória de trabalho.

6 DESENVOLVIMENTO DA PESQUISA

6.1 DESENVOLVIMENTO

O desenvolvimento de um sistema especialista é um processo iterativo, no qual o sistema é concebido a partir de um pequeno conhecimento a respeito do problema. DURKIN (1994, p.43) afirma: "The designer develops the system in parallel with obtaining a better understanding of the problem [...] " [12] . Tal conhecimento vai aumentando a medida que o projeto vai sendo desenvolvido. DURKIN (1994, p. 43) declara: "[...] an expert system is grown." [13] . Tal afirmação enfatiza a importância de se desenvolver um protótipo inicial que deve ser refinado em paralelo com a aquisição de um melhor entendimento a respeito do problema.

Para se atingir o objetivo geral do trabalho será necessário adquirir conhecimento suficiente sobre a criação de editais. Isto envolve absorver todo o conhecimento fundamental (regras, relacionamentos, fatos, entre outros) e o conhecimento empírico (heurísticas) que o especialista humano adquiriu através da sua experiência. Após este processo será necessário codificar o conhecimento adquirido e a forma como um especialista humano resolve os problemas.

6.2 FASES DO DESENVOLVIMENTO

---

[12] O projetista desenvolve o sistema em paralelo com a obtenção de um melhor entendimento a respeito do problema.

[13] Um sistema especialista amadurece.

A figura 2 ilustra as fases que serão seguidas para o desenvolvimento do projeto.

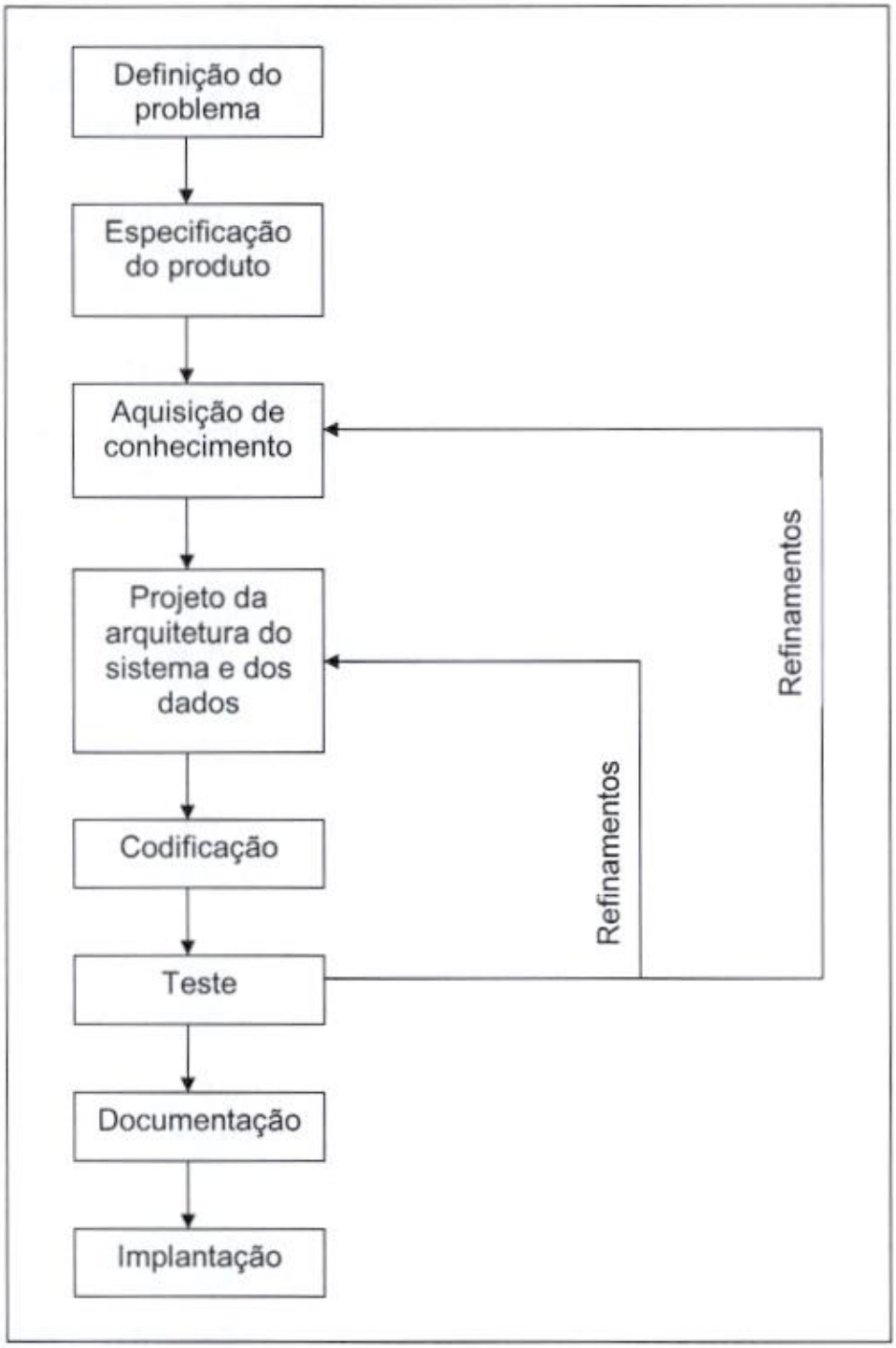


Figura 2 – Fases do desenvolvimento do sistema

BIBLIOTECA DA ESCOLA TECNICA DA UFPR

6.3 DEFINIÇÃO DO PROBLEMA

Nesta fase serão determinados os objetivos gerais, as características importantes e o foco do projeto.

6.4 ESPECIFICAÇÃO DO PRODUTO

Nesta fase será desenvolvido um documento que contenha todos os requisitos do sistema computacional. Este documento deverá ser assinado pelo cliente e pelo desenvolvedor para demonstrar o comum acordo entre ambas as partes no que diz respeito ao conteúdo de tal documento.

6.5 AQUISIÇÃO DE CONHECIMENTO

DURKIN (1994, p. 41) dá a seguinte definição para aquisição de conhecimento: "The process of acquiring, organizing, and studying knowledge." [14] .

Em um primeiro momento, nesta fase deve-se absorver os conceitos e os métodos gerais para a criação de editais. Este conhecimento inicial será utilizado para o desenvolvimento de um protótipo inicial. Posteriormente a aquisição de conhecimento será utilizada para refinar o conhecimento e o protótipo. Este processo de refinar o conhecimento adquirido e o protótipo será realizado até que o sistema computacional alcance os objetivos estabelecidos.

A aquisição de conhecimento corresponde a uma fase extremamente delicada, a qual deverá ser realizada em conjunto com um especialista humano da área. Da forte proximidade entre o desenvolvimento e o especialista, durante a fase de aquisição de conhecimento, dependerá o sucesso do projeto. JAMES (1985, p. 64) faz o seguinte comentário a respeito da fase de aquisição de conhecimento:

> O maior problema com todos os sistemas especialistas consiste na seleção da coleção inicial de regras que irá formar o banco de dados, o que é normalmente executado trabalhando-se juntamente com um especialista humano e tentando descobrir que regras ele usa em seu trabalho.

---

[14] O processo de adquirir, organizar e estudar o conhecimento.

6.6 PROJETO DA ARQUITETURA DO SISTEMA E DOS DADOS

Nesta fase a estrutura e a organização do sistema serão definidas. O objetivo inicial será construir um protótipo do sistema. DURKIN (1994, p. 41) afirma sobre a importância do protótipo: "The purpose of the prototype is to provide a vehicle for obtaining a better understanding of the problem." [15] .

Outro aspecto importante a ser observado é que esta fase é iterativa. DURKIN (1994, p. 41) afirma esta situação através da seguinte declaração: "System design is inherently an iterative process where findings from system testing are used to refine the system's knowledge and structure." [16]

6.7 CODIFICAÇÃO

Esta fase corresponde a codificação da estrutura definida na fase de projeto em uma linguagem de programação escolhida pelo desenvolvedor.

6.8 TESTE

A palavra-chave nesta fase é validação. A proximidade entre o desenvolvedor e o especialista é uma característica importante desta fase. Outro aspecto importante é que a fase de teste não é uma tarefa separada, mas sim um processo que se estende ao longo do projeto. DURKIN (1994, p. 41) afirma:

> Following each interview with the domain expert, new knowledge is added to the system. This is followed by additional testing where again the system's knowledge may be modified. The major objective of testing is to validate the overall structure of the system and its knowledge. In addition, this phase studies the acceptability of the system by the end-user. [17]

---

[15] O propósito do protótipo é fornecer um veículo para a obtenção de um melhor entendimento do problema.

[16] O projeto do sistema é naturalmente um processo iterativo onde as descobertas obtidas na fase de teste são utilizadas para refinar o conhecimento do sistema e a sua estrutura.

[17] Após cada entrevista com o especialista do domínio, novo conhecimento é adicionado ao sistema. Posteriormente novos testes adicionais são realizados onde o conhecimento pode novamente ser modificado. O maior objetivo da fase de teste é validar a estrutura geral e o conhecimento do sistema. Além disto, esta fase estuda a aceitabilidade do sistema pelo usuário final

6.9 DOCUMENTAÇÃO

É a fase em que se reúnem todas as informações do projeto em uma documentação que serve tanto para o usuário quanto para o desenvolvedor. O usuário necessita de informações sobre a forma de utilizar o sistema e as suas características funcionais. O desenvolvedor, por sua vez, encontra nestes documentos uma apresentação bem organizada a respeito do conhecimento do sistema. DURKIN (1994, p. 43) afirma sobre esta fase: "It is augmented throughout the project as new knowledge is obtained." [18].

6.10 IMPLANTAÇÃO

Corresponde à fase de entrega do sistema computacional, já devidamente concluído e testado mediante aos requisitos presentes na especificação do produto.

6.11 MODELAGEM

A modelagem trata da construção de modelos para visualizar e controlar a arquitetura de um determinado sistema. BOOCH e RUMBAUGH e JACOBSON (2000, p. 6) dão a seguinte definição para modelo: "Um modelo é uma simplificação da realidade." . Sendo assim, a modelagem proporciona uma melhor compreensão acerca do sistema que se deseja construir, e a sua importância é caracterizada por BOOCH e RUMBAUGH e JACOBSON (2000, p. 6) da seguinte maneira:

> Com a modelagem, alcançamos quatro objetivos.
> 1. Os modelos ajudam a visualizar o sistema como ele é ou como desejamos que seja.
> 2. Os modelos permitem especificar a estrutura ou o comportamento de um sistema.
> 3. Os modelos proporcionam um guia para a construção do sistema.
> 4. Os modelos documentam as decisões tomadas.

6.12 MODELAGEM ORIENTADA A OBJETOS

---

[18] Amadurece ao longo do projeto enquanto novo conhecimento for obtido.

A modelagem orientada a objetos corresponde a uma metodologia para desenvolvimento de software, cujo principal bloco de construção é o objeto. Um objeto corresponde a um elemento do mundo real. DEITEL e DEITEL (2003, p. 67) exemplificam o termo objeto através da seguinte afirmação:

> Pessoas, animais, plantas, carros, aviões, construções, computadores, etc. Seres humanos pensam em termos de objetos. Temos a maravilhosa habilidade de abstração, que nos permite visualizar imagens numa tela, como pessoas, aviões, árvores e montanhas, como objetos, em vez de pontos coloridos isolados (chamados píxels – abreviatura de picture elements [19]). Podemos, se desejarmos, pensar em termos de praias em vez de grãos de areia, florestas em vez de árvores e casas em vez de tijolos.

Esta declaração demonstra exemplos de objetos e introduz um importante conceito da orientação a objetos, a abstração de dados. Através da abstração de dados o programador pode ter uma visão mais generalista sobre o problema, o que é importante para a construção de sistemas computacionais complexos.

Objetos possuem um identificador, ou um nome que lhe é atribuído para diferenciá-los dos demais objetos, atributos que correspondem aos dados a eles associados, um estado para descrever os dados e um comportamento que corresponde a ações que o objeto pode realizar. BOOCH e RUMBAUGH e JACOBSON (2000, p. 11) procuram demonstrar objetos relacionados a sistemas computacionais através da seguinte declaração:

> Por exemplo, considere uma arquitetura simples com três componentes para um sistema de cobrança, incluindo a interface para o usuário, uma camada intermediária e um banco de dados. Na interface para o usuário, você encontrará objetos concretos, como botões, menus e caixas de diálogo. No banco de dados, haverá objetos concretos como tabelas, que representam entidades provenientes do domínio do problema, incluindo clientes, produtos e pedidos. Na camada intermediária, você encontrará objetos como transações e regras de negócios, além de visões de alto nível relacionadas às entidades do problema, como clientes, produtos e pedidos.

Outro elemento importante para a orientação a objetos é a classe. Esta corresponde a uma descrição de objetos comuns, como a classe do seres humanos e dos automóveis, por exemplo.

DEITEL e DEITEL (2003, p. 67) afirmam sobre orientação a objetos: "[...] suas principais tecnologias – herança e polimorfismo." . A herança corresponde a uma forma de reutilização de código, pois novas classes são criadas a partir de classes já existentes, absorvendo seus atributos e comportamentos. Desta forma o

---

[19] Elementos da imagem

programador não necessita reescrevê-los, basta herdá-los. A classe da qual se herda é chamada de superclasse, enquanto que a classe que herdou é denominada subclasse. A vantagem prática da herança seria a capacidade de definir na subclasse adições ou substituições aos recursos herdados da superclasse.

O polimorfismo, por sua vez, permite ao programador escrever softwares [20] com um escopo geral, para tratar de uma ampla variedade de classes relacionadas já existentes ou, ainda, a serem criadas. Desta forma o polimorfismo torna mais fácil a manutenção do programa e o acréscimo de novos recursos ao sistema.

6.13 LINGUAGEM DE MODELAGEM

A linguagem a ser utilizada para modelagem do sistema será a Unified Modeling Language [21] (UML).

Nos anos 80, quando um crescente número de empresas passou a utilizar o projeto orientado a objetos, surgiu a necessidade de um processo adequado para tratá-lo. DEITEL e DEITEL (2003, p. 69) dão um breve histórico sobre o desenvolvimento da UML:

> Em 1994, James Rumbaugh juntou-se a Grady Booch na Rational Software Corporation e os dois começaram a trabalhar para unificar seus populares processos. Em seguida, juntou-se a eles Ivar Jacobson. Em 1996, o grupo liberou versões preliminares da UML para a comunidade de engenharia de software e pediu realimentação.

Atualmente UML é a linguagem de modelagem mais amplamente utilizada para o desenvolvimento de projetos orientados a objeto. Ela possui vários diagramas que facilitam a compreensão e a documentação do sistema.

6.14 LINGUAGEM DE PROGRAMAÇÃO

A linguagem de programação adotada para a geração do código da parte convencional do sistema é a linguagem Java.

---

[20] Parte lógica de um computador, correspondente aos programas.

[21] Linguagem de modelagem unificada.

Java foi desenvolvida por uma equipe de pesquisadores da Sun Microsystems. Inicialmente destinava-se para a utilização em dispositivos eletrônicos compactos, em um projeto chamado de Green, em 1991. O seu criador, James Gosling, chamou a linguagem criada de Oak [22], em homenagem a uma árvore que dava para a janela do seu escritório, na Sun Microsystems. Porém, mais tarde descobriu-se que já existia uma linguagem de programação chamada Oak. O nome Java surgiu quando a equipe da Sun Microsystems visitou uma cafeteria local. O nome do local de origem do tipo do café importado, cidade de Java, foi a inspiração para o novo nome da linguagem desenvolvida. Porém o projeto Green atravessou dificuldades e por pouco não foi cancelado. Em 1993, novas oportunidades surgiram com a Internet [23], a qual é definida por DEITEL e DEITEL (2003, p. 61) da seguinte maneira:

> A Internet foi desenvolvida há mais de três décadas, financiada pelo Departamento de Defesa dos EUA. Originalmente projetada para conectar os principais sistemas de computadores de cerca de uma dezena de universidades e organizações de pesquisa, a Internet atualmente é acessível a centenas de milhões de computadores no mundo.

Apesar de a sua utilização inicial ser destinada para a criação de páginas dinâmicas para a Internet, Java cresceu tanto que os programas codificados nesta linguagem podem ser executados virtualmente em qualquer tipo de computador, independente do hardware [24], ou até mesmo em outros tipos de aparelhos (como celulares, por exemplo).

FURGERI (2005, p. 18) declara: "A partir de 1995, a Sun anunciou o Java, não apenas como mais uma linguagem de programação, mas como uma nova plataforma de desenvolvimento."

Pode-se citar como vantagens da linguagem Java, o fato de ser uma tecnologia livre, orientada a objetos, a portabilidade dos códigos e do programa gerado (capacidade de serem utilizados em diferentes plataformas), grande quantidade de classes utilitárias que reforçam o conceito de reutilização de códigos, gerenciamento automático da memória e fácil integração a sistemas de bancos de dados.

---

[22] Carvalho.

[23] Rede internacional.

[24] Parte física de um computador, correspondente aos periféricos.

DEITEL e DEITEL (2003, p. 61) declara sobre o futuro do desenvolvimento de sistemas e a importância da Java Application Programming Interface [25] (Java API) neste contexto:

> Talvez a mais animadora seja a noção de que novas classes podem herdar de classes abundantes, como aquelas oferecidas com a Java API. As organizações desenvolvem suas próprias bibliotecas de classe e podem tirar proveito de outras bibliotecas disponíveis no mundo. Algum dia, a maior parte do software poderá ser construída de componentes reutilizáveis padronizados, exatamente como se constrói o hardware. Isso ajudará a superar os desafios de se desenvolver os softwares cada vez mais poderosos que precisaremos no futuro.

---

[25] Interface de programação de aplicações Java.

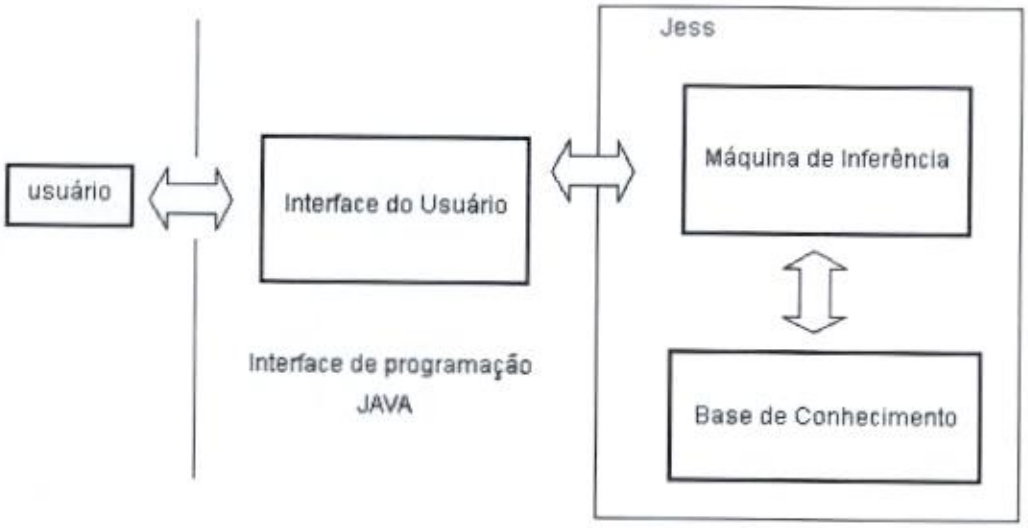
7 IMPLEMENTAÇÃO

Para a implementação do código-fonte foi utilizada a linguagem Java em conjunto com a shell Jess. O desenvolvimento foi realizado utilizando o software Eclipse.

O código referente ao sistema especialista foi implementado através do Jess (Java Expert System Shell), um "*script shell*" para construção de sistemas especialistas baseados em regras. O JESS foi inteiramente desenvolvido em Java. Originalmente baseado na linguagem CLIPS, mas adiciona uma série de facilidades. Entre elas o Jess integra a linguagem de scripts, a base de conhecimento e sua máquina de inferência ao ambiente Java e suas tecnologias.O Jess utiliza-se da tecnologia de "*beans*" (componentes Javabeans) para permitir que as regras tenham acesso aos métodos Get () / Set () das instâncias beans como simples propriedades dos objetos. Assim as propriedades são lidas e alteradas pela máquina de inferência do Jess. O padrão JavaBeans ainda possibilita que as modificações nas propriedades sejam automaticamente notificadas para a máquina de inferência quando ocorrem.

A máquina de inferência do JESS implementa um algoritmo Rete que permite grande eficiência no processamento de regras pelo sistema. Sua máquina de inferência suporta tanto o encadeamento para trás (*backward chaining*) como o encadeamento de regras para frente (*foreward chaining*). O encadeamento para trás é o mais empregado e destaca-se como mais intuitivo para o desenvolvedor, sendo a sua base o mecanismo de recursão.

A estrutura básica da aplicação é uma base de conhecimento que armazena um conjunto de regras, fatos e heurísticas que juntos correspondem ao conhecimento do especialista no domínio sobre o qual foi construído o sistema. O motor de inferência é responsável pelo processamento do conhecimento. De forma geral, podemos afirmar que o processo permite tirar conclusões a partir do conhecimento armazenado, explorando o encadeamento lógico existente na base de conhecimento.



Estrutura do sistema especialista

O Jess diferentemente de outros *scripts shell*, fornece suporte mais amplo às funções de linguagem de programação. Essa característica parece bastante importante para a maior popularização do uso de sistemas especialistas em aplicações comerciais, pois facilita o trabalho do programador em diversas tarefas, como a entrada e saída de dados e a manipulação tradicional de dados, permitindo ainda a adoção de soluções de programação tradicional quando requeridas. Desse modo acreditamos ter agregado vantagens no desenvolvimento e superado algumas das limitações da programação tradicional.

O sistema foi divido em vários scripts, se comportando como sistemas especialistas menores. Os scripts possuem funções bem definidas, são elas: um script para gerenciar o questionário, um para iniciar o sistema, um contendo as funções específicas para manipular um edital, um para montar os frames específicos do edital. Estes últimos, do ponto de vista do JESS, correspondem a fatos, ou seja, a menor unidade de informação que pode ser, separadamente, adicionada ou removida da memória de trabalho. Além disto existem vários scripts para formar a base de conhecimento, um script contendo regras de validações do sistema, classes Java para fornecer a interface gráfica e o output do edital em disco. Ao longo do código há comentários explicando o funcionamento das regras, funções, frames e classes.

8 MODELAGEM DO CONHECIMENTO

Como foram utilizadas técnicas de IA para o desenvolvimento do projeto foi necessária a modelagem do conhecimento sobre os editais de concurso criados pelo NC-UFPR, para a geração da base de conhecimento.

Os editais criados pelo núcleo de concursos são normalmente gerados em um processador de texto padrão, assim, foi necessário criar um mecanismo para criar o edital de forma a manter a compatibilidade com tais processadores. O sistema ao concluir a criação do edital gera um documento com as extensões .pdf e .rtf que são visualizadas nos mais diversos softwares de edição e visualização de documentos de texto existentes no mercado principalmente os gratuitos como o Acrobat Reader (.pdf) e o OpenOffice (.pdf e .rtf).

Basicamente foi desenvolvido um edital genérico (universal), através da total colaboração do especialista, que serviu como base para a modelagem do conhecimento do sistema. . Com isto o sistema deve ser capaz de se enquadrar na maioria dos casos. A base de conhecimento é formada pelo layout deste edital, regras e frames através de vários scripts

O script Manager.clp foi desenvolvido de forma muito especial. A sua função é gerenciar o questionário do sistema. O grande mérito deste script é que ele funciona de forma independente da sua base de conhecimento, ou seja, ele pode ser perfeitamente utilizado para administrar o questionário de qualquer outro sistema especialista, ficando de acordo com o princípio da reutilização de código. Este resultado foi alcançado pela aplicação de herança sobre os frames do sistema. O exemplo abaixo demonstra o funcionamento deste mecanismo:

Regra do script Manager.clp para validar a colocação de perguntas na memória de trabalho

```
(defrule putTriggerPerguntar

  ;Goal-seeking
  (MAIN::need-Resposta (id ?id))

  ;Existe a variável id
  (MAIN::Key (id ?id))

  ;Pergunta ainda não foi respondida
  (not (MAIN::Resposta (id ?id)))

  ;Não há solicitação de disparar perguntas
```

```
   (not (MAIN::Perguntar ?))

   =>

   ;Coloca a trigger
   (assert (MAIN::Perguntar ?id))

)
```

Regra do script Introducao.clp (componente da base de conhecimento) que testa a hipótese da colocação de perguntas na memória de trabalho

```
(defrule putPerguntasIntroducao

        (MAIN::Pergunta (id ch_Publicador) (tipo fraseComArtigo)
        (texto "Quem torna pública a realização do Concurso? (Inicie com o artigo
adequado)"))

        (MAIN::Pergunta (id ch_TipoDecretoIntroducao) (tipo escolha) (teste 1 2 3 -1 -
2 -3 -4 -5 -6)
        (texto "Entre com o tipo da resolução legal que torna pública a realização do
concurso.
(1 Decreto, 2 Lei, 3 Portaria)"))

        (MAIN::Pergunta (id ch_NumDecretoIntroducao) (tipo numero)
        (texto "Entre com o número da resolução legal que torna pública a realização
do concurso."))

        (MAIN::Pergunta (id ch_DataDecretoIntroducao) (tipo data)
        (texto "Entre com  a data da resolução legal que torna pública a realização do
concurso. (dd-mm-aa)"))

        (MAIN::Pergunta (id ch_NumAnexos) (tipo numero)
        (texto "Quantos anexos o Edital possui? (0 – Nenhum)"))

   =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item introducao) (tipo pergunta)))

)
```

Regra do script Introducao.clp que testa a hipótese da colocação das variáveis do item Introdução.

```
(defrule putVariaveisIntroducao

        (MAIN::Variavel (id ch_Publicador) (item introducao)
```

```
        (descricao "O Concurso torna-se aberto ao público mediante as atribuições
legais de alguma
pessoa física ou jurídica. A resposta a esta pergunta irá indicar ao sistema
quem é esta pessoa. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta: A Comissão Executiva do Concurso Público"))

        (MAIN::Variavel (id ch_TipoDecretoIntroducao) (item introducao)
        (descricao "O sistema necessita saber se a resolução legal que torna pública
a
realização do concurso é do tipo Decreto, Lei ou Portaria."))

        (MAIN::Variavel (id ch_NumDecretoIntroducao) (item introducao)
        (descricao "A resposta a esta pergunta irá definir o número da resolução legal
que torna
pública a realização do concurso. Exemplo de resposta: 1387"))

        (MAIN::Variavel (id ch_DataDecretoIntroducao) (item introducao)
        (descricao "A resposta a esta pergunta irá indicar ao sistema a data da
resolução
legal que torna pública a realização do concurso. Esta deve ser fornecida no formato
dia-mês-ano. Exemplo de resposta: 14-02-06"))

        (MAIN::Variavel (id ch_DecretoIntroducao) (item introducao)
        (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento das variáveis ch_TipoDecretoIntroducao,
ch_NumDecretoIntroducao
e ch_DataDecretoIntroducao. Será utilizada na sentença ch_DecretoIntroducao."))

        (MAIN::Variavel (id ch_QuadroServidores) (item introducao)
        (descricao "Variável utilizada internamente pelo sistema, resultante da
inferência
sobre o a abrangência do edital. Será utilizada na sentença ch_QuadroServidores."))

        (MAIN::Variavel (id ch_NumAnexos) (item introducao)
        (descricao "Variável que indica ao sistema quantos anexos o edital possui.
Caso o
edital não possua anexos, basta o usuário digitar 0."))

        (MAIN::Variavel (id ch_Anexos) (item introducao)
        (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento da variável ch_NumAnexos. Será utilizada na sentença
ch_Anexos."))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item introducao) (tipo variavel)))

)
```

No exemplo acima o frame Variável, que compõe o edital, corresponde a uma subclasse do frame Key. A regra do Manager.clp faz uma chamada a superclasse Key para poder validar a colocação da pergunta na memória de trabalho. Pode-se dizer que para se utilizar a funcionalidade do script Manager.clp, basta criar subclasses dos frames deste.

Esta funcionalidade correspondeu a um desafio para a equipe, pois em princípio todo o sistema estava sendo desenvolvido em um único script, que se relacionava com a base de conhecimento. Utilizando-se deste script e com um conhecimento sobre Jess e Sistemas Especialistas uma infinidade de outros sistemas podem ser criados com maior facilidade.

9 ARQUITETURA

Para desenvolver o sistema, foi necessário identificar as possibilidades e recursos para sua implantação. No desenvolvimento do trabalho acadêmico utilizamos o sistema operacional Windows XP pois os equipamentos no cliente são providos por este sistema operacional, a plataforma de desenvolvimento utilizada foi Java através do Eclipse. Todo desenvolvimento foi baseado em sistemas especialistas e utilizando a shell JESS devido a grande complexidade e pelo grande número de fatos e casos possíveis que descartam a possibilidade de um programa convencional.

10 FERRAMENTAS UTILIZADAS

Java 1.5

Linguagem de programação orientada a objeto desenvolvida na década de 90 pelo programador James Gosling, na empresa Sun Microsystems. Diferentemente das linguagens convencionais, que são compiladas para código nativo, a linguagem Java é compilada para um "bytecode" que é executado por uma máquina virtual.

Foi projetada tendo em vista os seguintes objetivos:

Orientação a objeto - Baseado no modelo de Smalltalk e Simula67;

Portabilidade - Independência de plataforma - "write once run anywhere";

Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com protocolos TCP/IP, como HTTP e FTP;

Segurança - Pode executar programas via rede com restrições de execução;

Além disso, podem-se destacar outras vantagens apresentadas pela linguagem:

Sintaxe similar a Linguagem C/C++.

Facilidades de Internacionalização - Suporta nativamente caracteres Unicode;

Simplicidade na especificação, tanto da linguagem como do "ambiente" de execução (JVM);

É distribuída com um vasto conjunto de bibliotecas (ou APIs);

Possui facilidades para criação de programas distribuídos e multitarefa (múltiplas linhas de execução num mesmo programa);

Desalocação de memória automática por processo de coletor de lixo (garbage collector);

Carga Dinâmica de Código - Programas em Java são formados por uma coleção de classes armazenadas independentemente e que podem ser carregadas no momento de utilização.

Eclipse 3.2

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.

JESS 7.0

É um "script shell" para construção de sistemas especialistas baseados em regras inteiramente desenvolvidas em Java. Baseada em CLIPS.

JUDE 3.0.3

Java and UML Developer Environment é uma das ferramentas grátis para UML mais poderosas disponíveis atualmente. Com características que não são encontradas nas outras ferramentas grátis, como adicionar métodos no diagrama de sequência e a alteração se refletir no diagrama de classes. Sua performance impressiona, principalmente tratando-se de uma ferramenta 100% Java/Swing.

Diagram Designer 1.17

É um software gratuito que possibilita a elaboração de diagramas – como organogramas e fluxogramas. É um software de fácil uso e de grande utilidade na elaboração de desenhos, fluxos de trabalho e organização.

Microsoft Project

Software de gestão de projetos (ou gerência de projetos) produzido pela Microsoft.

Db Designer 4

Sistema de design que integra modelagem, criação e manutenção de sistemas e banco de dados, entre outras funcionalidades.

Microsoft Office 2003

Um pacote de aplicativos para escritório líder de mercado que contém programas como processador de texto, planilha de cálculo, banco de dados, apresentação gráfica e gerenciador de tarefas, e-mails e contatos.

Acrobat Reader 5

Software que permite que o usuário do computador visualize, navegue e imprima arquivos no formato PDF. Este tipo de arquivo é muito comum em documentações gerais (manuais de instrução, apostilas, eBooks). Por ser multiplataforma, está disponível para diversos sistemas operacionais.

11 DOCUMENTAÇÃO

O processo de documentação do desenvolvimento foi realizado a fim de facilitar a utilização, avaliação e manutenção do sistema. Ele se encontra nos anexos do projeto e há comentários por todo o código para facilitar a interpretação, as classes Java estão documentadas em javadoc nos padrões sun e o restante dos scripts e clps seguem um mesmo padrão para facilitar o entendimento.

12 TESTES

Durante toda a fase de desenvolvimento foram realizados testes pela equipe, o projeto foi dividido em vários módulos e através do Eclipse utilizando console do Java foi possível avaliar o seu desempenho e realizar a correção de falhas.

Durante as execuções dos testes dos módulos puderam ser verificadas diversas falhas lógicas, na modelagem do conhecimento e da máquina de inferência além de alguns problemas na utilização do JESS. Os testes foram importantíssimos para que esses problemas fossem solucionados a tempo, e em alguns casos, obrigaram o grupo a tomar medidas bruscas na implementação como reiniciar todo o processo do módulo e fazê-lo de maneira diferente.

13 ANÁLISE DE RESULTADOS

13.1 RESULTADOS DA EQUIPE

O desenvolvimento do projeto exigiu da equipe uma grande busca de novos conhecimentos e ferramentas. Nenhum dos integrantes já tinha trabalhado na prática com sistemas especialistas, assim a experiência com o projeto amadureceu o nível técnico da equipe em IA de maneira geral.

Isso ocorreu também durante a parte de programação, pois a maioria do código foi desenvolvido em JESS, o que tornou o desenvolvimento mais complexo e exigiu muito da equipe.

Foram utilizadas técnicas de programação orientada a objetos, programação estruturada e programação em linguagem declarativa, além das técnicas para o desenvolvimento de sistemas especialistas empregando técnicas de sistemas baseados em regras e sistemas baseados em frames.

Um dos "gargalos" foi a fase de aquisição de conhecimento, pois o desenvolvimento de editais de concursos públicos apresentou-se extremamente abstrato para os membros da equipe. Foram empregadas muitas horas de pesquisa e várias de seções de entrevista com o especialista do assunto para a aquisição e modelagem do conhecimento.

13.2 RESULTADOS DO PROJETO

Todos os objetivos principais do sistema planejados pela equipe foram atingidos e os resultados muito gratificantes tanto para o cliente que antes não possuía nenhum software para auxiliar na geração de editais quanto para os desenvolvedores pelo desafio e o resultado de todos os esforços. Com o software, todo o trabalho realizado pelo especialista tornou-se muito menos trabalhoso e mais ágil além de garantir que falhas não sejam realizadas durante o processo de desenvolvimento.

14 CONCLUSÃO

Através do Trabalho de Conclusão de Curso (TCC) e com a grande colaboração dos orientadores, foi possível atingir os objetivos esperados durante a elaboração do projeto final. Todo o trabalho necessário para desenvolver o projeto fez com que a equipe percebesse a total importância de todas as etapas necessárias para se desenvolver um bom projeto como por exemplo fazer um planejamento, criar cronogramas, elaborar diagramas entre outros.

Tudo isso amadureceu o conceito do processo de desenvolvimento de software, tanto no âmbito técnico quanto no gerencial. O aprendizado maior de novos conceitos como os sistemas especialistas só vieram a engrandecer o conhecimento da equipe e acreditamos que se tornou uma ótima ferramenta para que outros alunos também possam desfrutar de tudo e até a melhorar ou desenvolver projetos muito mais sofisticados do que o criado por nós.

REFERÊNCIAS

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **Uml, guia do usuário.** Rio de Janeiro: Campus, 2000.

CIS587, **The RETE Algorithm**. Disponível em <http://www.cis.temple.edu/~ingargio/cis587/readings/rete.html>. Acesso em: outubro, 2006.

DEITEL, H. M.; DEITEL, P. J. **Java, como programar.** 4. ed. Porto Alegre: Bookman, 2003.

DEVX. Disponível em: <http://www.devx.com/Java/Article/17651> Acesso em: Setembro, 2006.

DURKIN, J. **Expert systems:** design and development. New Jersey: Prentice Hall, 1994.

FORGY, C.L. **Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem Artificial Intelligence**, 3. ed. Winston: P.H, 1994.

FRIEDMAN-HILL, E. J. **Jess in action: rule-based systems in java**. Greenwich: Manning Publications Co., 2003.

FRIEDMAN-HILL, E.J. **Jess, The expert system shell for the Java platform.** Disponível em: <http://herzberg.ca.sandia.gov/jess/>. Acesso em: maio, 2006.

FURGERI, S. **Java 2: ensino didático**: desenvolvendo e implementado aplicações. São Paulo: Érica, 2002.

GIARRATANO,J., RILEY,G.: **Expert Systems**: Principles and Programming PWS Publishing Co., 1994.

INARGIOLA. **Cis587: the rete algorithm.** Disponível em: <http://www.cis.temple.edu/~ingargio/cis587/readings/rete.html> Acesso em: Outubro, 2006.

JAMES, M. **Inteligência artificial em basic.** Rio de Janeiro: Campus 1985.

JESS, **JESS**: The expert system shell for the Java platform. Disponível em <http://www.jessrules.com/>, Acesso em: Agosto, 2006 .

LOWAGIE, B. **Tutorial: iText by Example.** Disponível em <http://itextdocs.lowagie.com/tutorial/ > Acesso em: Novembro, 2006

MANCHINI, D. P.; PAPPA, G. L. **Sistemas Especialistas**. Disponível em: <http://www.din.uem.br/ia/intelige/especialistas/especialistas/index.html>. Acesso em: outubro, 2006.

NOS : **Certificate in Computer Applications**. Disponível em: <http://www.nos.org/htm/course.htm>. Acesso em: Novembro, 2006

PROTEGE. Disponível em: <http://protege.stanford.edu/download/prerelease/full/>

RILEY, G. **What is clips?.** Disponível em: <http://www.ghg.net/clips/CLIPS.html> Acesso em: Novembro, 2006.

RUSSEL, S.; NORVIG, P. **Artificial Intelligence:** a Modern Approach. Englewood Cliffs: Prentice Hall,1997

SERRA NEGRA, C. A.; SERRA NEGRA, E. M. **Manual de trabalhos monográficos de graduação, especialização, mestrado e doutorado.** 2. ed. São Paulo: Atlas, 2004.

SUN. Disponível em: <http://java.sun.com/developer/technicalArticles/J2SE/JavaRule.html> Acesso em: Setembro, 2006.

SUN. Disponível em: < http://java.sun.com/jcp/> Acesso em: Setembro, 2006.

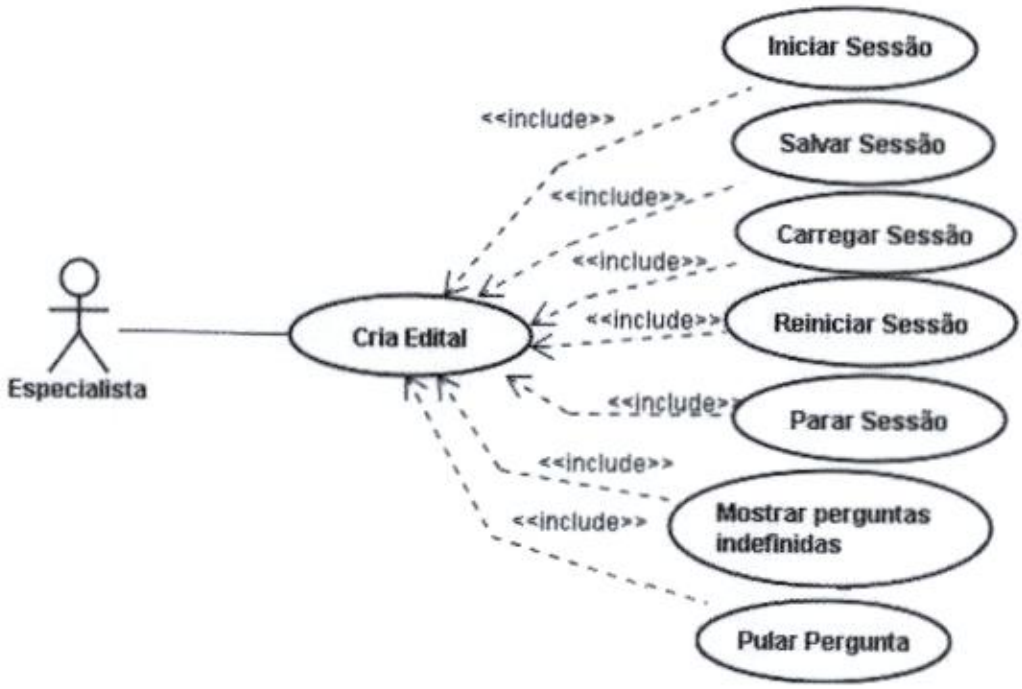
UNINOVA. 4. **Sistemas periciais de tempo real.** Disponível em <http://www.uninova.pt/~cam/si/SI4.pdf> Acesso em: outubro, 2006.

ANEXOS

DIAGRAMAS

DIAGRAMA DE CASOS DE USO

O diagrama de casos de uso descreve o que o sistema faz do ponto de vista de um observador externo. A ênfase é no que o sistema faz em vez de como. Cada caso de uso possui uma descrição resumida, um fluxo de eventos e um cenário.



Por tratar-se de um sistema especialista todo o trabalho é realizado no desenvolvimento interno do sistema, isso torna o diagrama de casos de uso algo extremamente simples do ponto de vista de um observador externo.

CASO DE USO CRIA EDITAL

Descrição Resumida

Este caso de uso é iniciado pelo especialista. Ele possibilita dar inicio ao processo de criação de edital.

Fluxo de Eventos

Fluxo Principal

1 - O especialista acessa a interface principal do sistema onde seleciona o tipo de edital a ser montado. Todo o processo de criação é realizado através de uma caixa onde o sistema faz as perguntas necessárias e o especialista descreve o que deseja.

2 – Quando iniciada as Meta Perguntas é preciso responde-las para o sistema definir quais são os itens que serão incorporados para o desenvolvimento e em seguida o sistema dispara todas as perguntas necessárias para criar o edital.

Fluxo Secundário

1 - Caso especialista deseje salvar o estado da memória em algum momento clica no botão Salvar Sessão.

2 - Caso deseje abrir uma seção incompleta salva anteriormente clica no botão Carregar Sessão.

3 - Caso deseje refazer o processo do item clica em Reiniciar Sessão.

4 - Caso opte por parar de responder clica em Parar Sessão.

5 - Caso não queira responder a pergunta no momento clica em Pular Pergunta.

6 - Caso tenha pulado alguma pergunta usuário pode clicar em Listar Perguntas Indefinidas e ver quais ainda faltam responder.

7 – Caso tenha parado a Sessão pode clicar em Iniciar Sessão para iniciar o processo.

Cenário Principal

Ator Especialista: Thaís clica no ícone e o sistema inicia a sessão, o sistema mostra as primeiras perguntas na caixa de texto que são chamadas de "meta perguntas". Com as respostas destas perguntas o sistema define o tipo de edital a ser criado e passa a interagir com Thaís através de perguntas específicas até finalizar a criação do edital e salvar em um arquivo no formato texto em um diretório específico no raiz do computador.

Cenário Alternativo

1- Thaís clica em Salvar Sessão e o edital que está em desenvolvimento e as etapas que já foram passadas durante a criação do edital são salvas.

2- Thaís clica em Carregar Sessão e o sistema abre uma sessão que estava em desenvolvimento e salva dentro de um diretório no computador.

3- Thaís utiliza o botão Reiniciar Sessão e o sistema limpa a memória de trabalho, ou seja, apagar toda e qualquer informação inserida até o momento e reinicia o sistema.

4- Thaís clica em Parar Sessão e o aplicativo interrompe definitivamente todo o sistema.

5- Ao ler pergunta Thaís resolve clicar no botão Pular Pergunta e responde-la depois.

6- Como Thaís pulou algumas perguntas e deseja saber quais eram clica no botão Listar Perguntas Indefinidas e o sistema verifica e lista essas perguntas.

7- Thaís após parar a sessão resolve começar de novo, então clica no botão Iniciar Sessão.

DIAGRAMA DE CLASSES

O diagrama de classes dá uma visão geral do sistema mostrando suas classes e os relacionamentos entre elas. São estáticos, pois mostram as interações mas não o que acontece quando elas ocorrem.

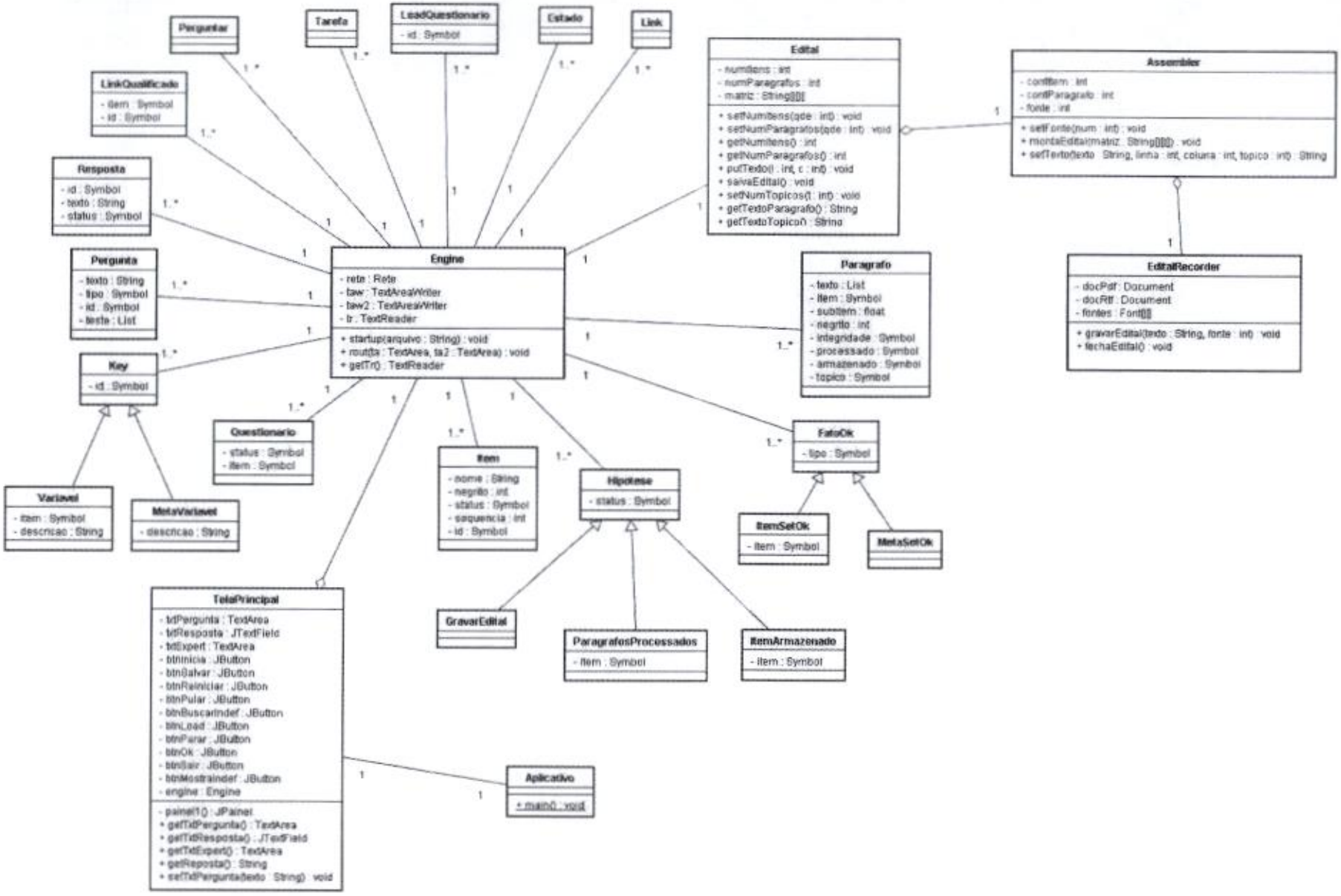

Perguntar
Tarefa
LeadQuestionario
- id : Symbol
Estado
Link
Edital
- numItens : int
- numParagrafos : int
- matriz : String[][]
+ setNumItens(qde : int) : void
+ setNumParagrafos(qde : int) : void
+ getNumItens() : int
+ getNumParagrafos() : int
+ putTexto(l : int, c : int) : void
+ salvaEdital() : void
+ setNumTopicos(t : int) : void
+ getTextoParagrafo() : String
+ getTextoTopico() : String
Assembler
- contItem : int
- contParagrafo : int
- fonte : int
+ setFonte(num : int) : void
+ montaEdital(matriz : String[][]) : void
+ setTexto(texto : String, linha : int, coluna : int, topico : int) : String
LinkQualificado
- item : Symbol
- id : Symbol
Resposta
- id : Symbol
- texto : String
- status : Symbol
Engine
- rete : Rete
- taw : TextAreaWriter
- taw2 : TextAreaWriter
- tr : TextReader
+ startup(arquivo : String) : void
+ rout(ta : TextArea, ta2 : TextArea) : void
+ getTr() : TextReader
Pergunta
- texto : String
- tipo : Symbol
- id : Symbol
- texto : List
Paragrafo
- texto : List
- item : Symbol
- subItem : float
- negrito : int
- integridade : Symbol
- processado : Symbol
- armazenado : Symbol
- topico : Symbol
EditalRecorder
- docPdf : Document
- docRtf : Document
- fontes : Font[]
+ gravarEdital(texto : String, fonte : int) : void
+ fechaEdital() : void
Key
- id : Symbol
Questionario
- status : Symbol
- item : Symbol
Item
- nome : String
- negrito : int
- status : Symbol
- sequencia : int
- id : Symbol
Hipotese
- status : Symbol
FatoOk
- tipo : Symbol
Variavel
- item : Symbol
- descricao : String
MetaVariavel
- descricao : String
ItemSetOk
- item : Symbol
MetaSetOk
TelaPrincipal
- txtPergunta : TextArea
- txtResposta : JTextField
- txtExpert : TextArea
- btnInicia : JButton
- btnSalvar : JButton
- btnReiniciar : JButton
- btnPular : JButton
- btnBuscarIndef : JButton
- btnLoad : JButton
- btnParar : JButton
- btnOk : JButton
- btnSair : JButton
- btnMostrarIndef : JButton
- engine : Engine
- painel1() : JPainel
+ getTxtPergunta() : TextArea
+ getTxtResposta() : JTextField
+ getTxtExpert() : TextArea
+ getResposta() : String
+ setTxtPergunta(texto : String) : void
GravarEdital
ParagrafosProcessados
- item : Symbol
ItemArmazenado
- item : Symbol
Aplicativo
+ main() : void

DIAGRAMA DE ESTADOS

Os diagramas de estados apresentam os estados possíveis dos objetos, e as transições que causam uma mudança de estado.

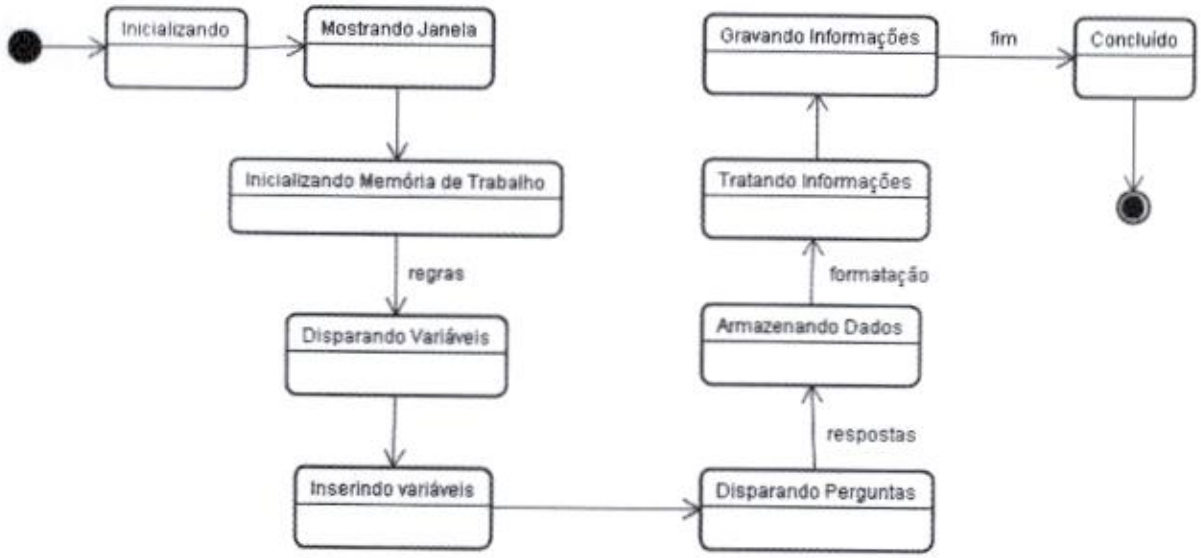
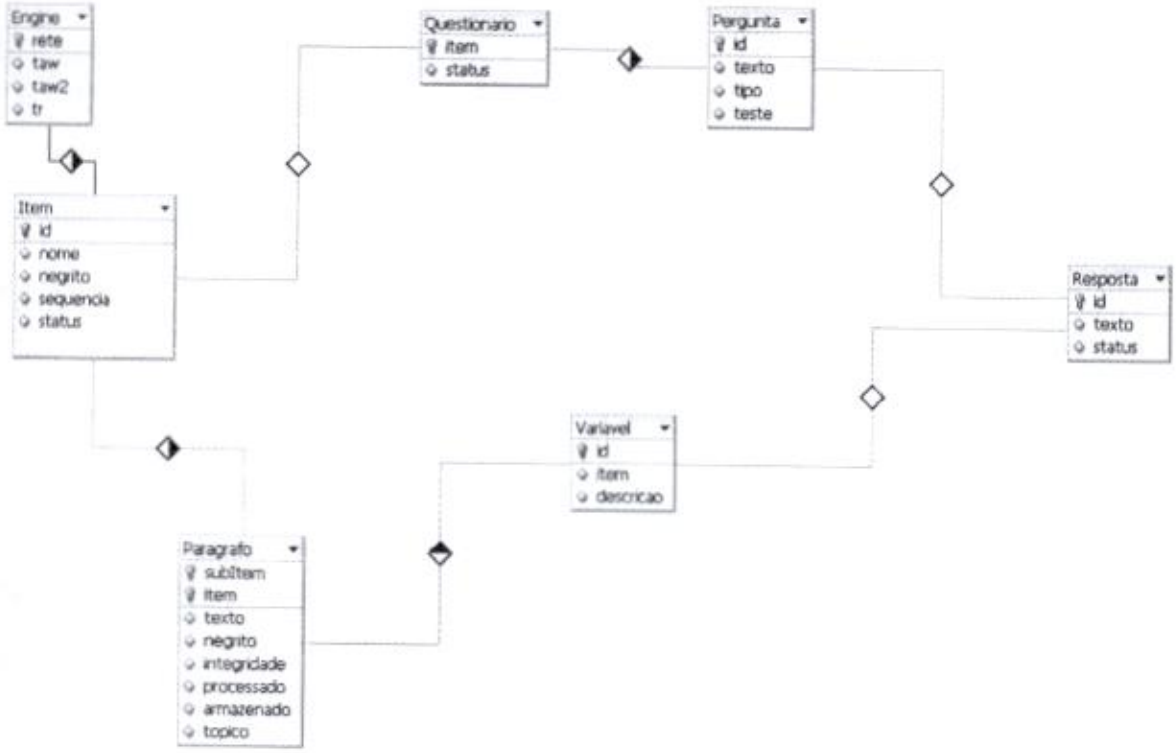
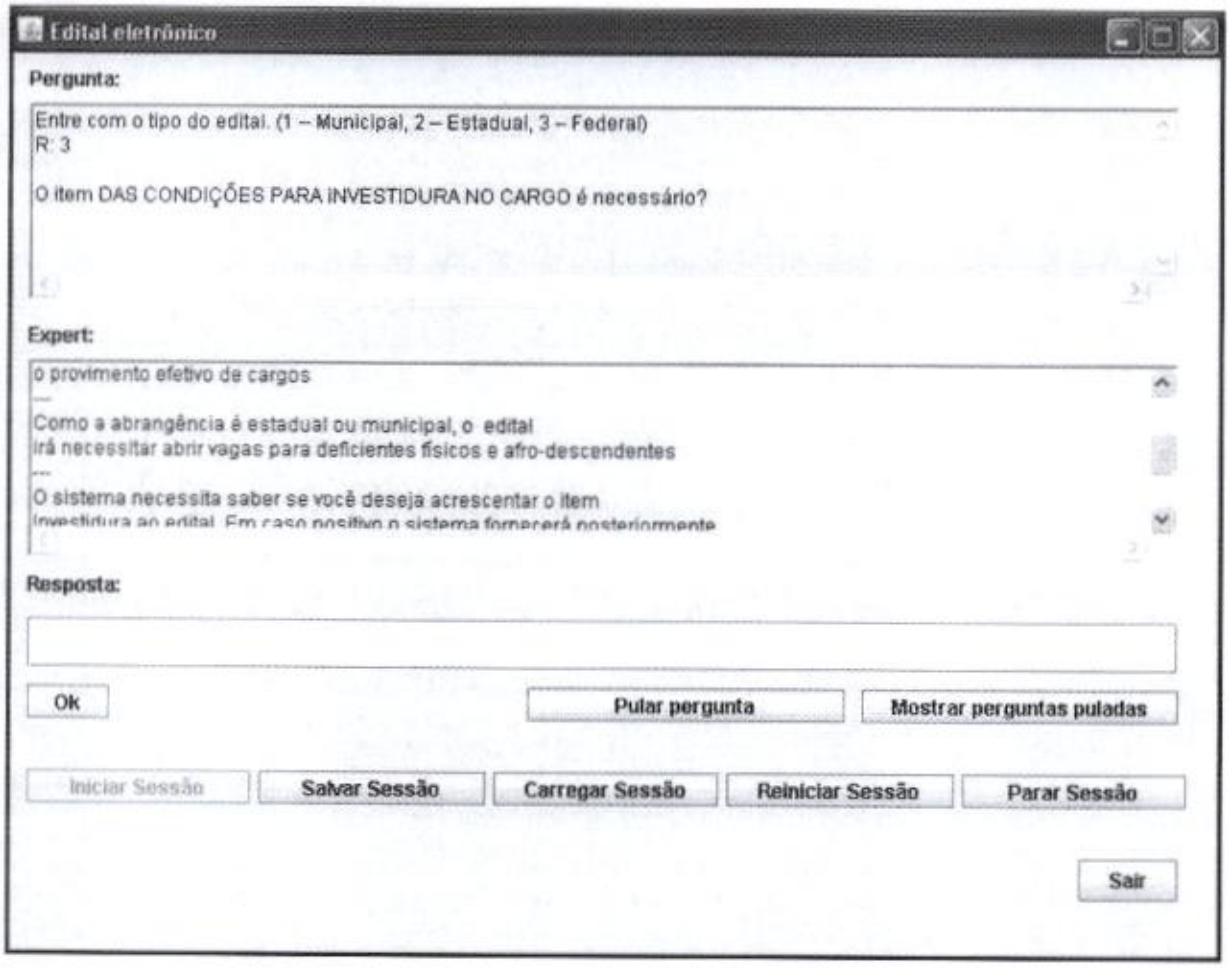


DIAGRAMA DE ENTIDADE E RELACIONAMENTO

O diagrama entidade-relacionamento (DER) demonstra a estrutura física, compreendida pelas tabelas e os relacionamentos entre elas, do banco de dados.



TELA DO SISTEMA



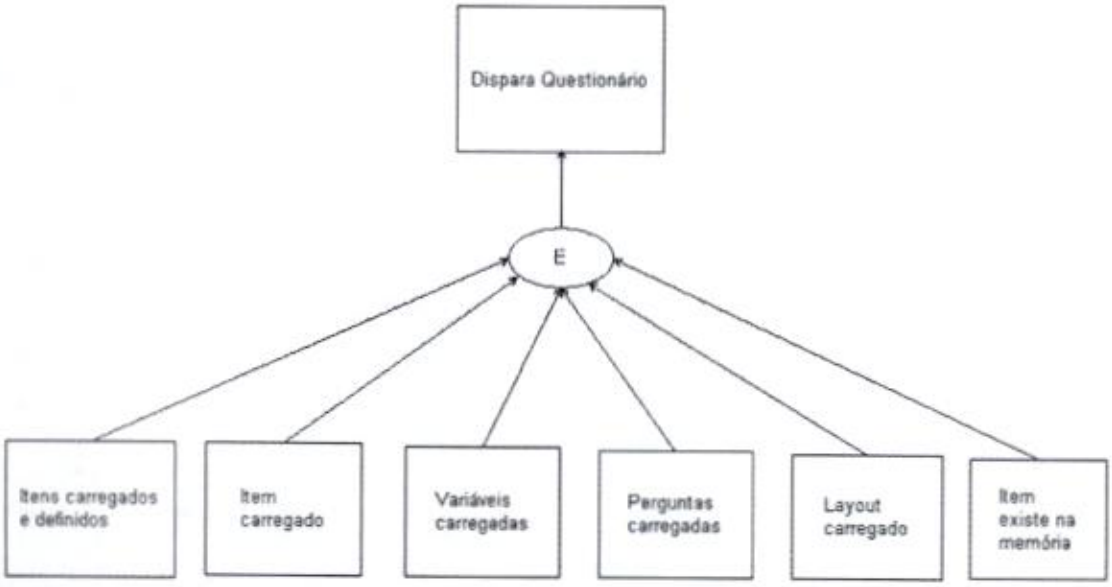
Todos os diagramas foram gerados utilizando o software Jude 3.0.3 com exceção do diagrama de entidade e relacionamento que foi criado utilizando o Db Designer 4.

Outro diagrama existente na UML é o de Atividades que no caso do nosso sistema é representado através de Fluxogramas e Redes de Inferências, para a criação foi usado o software Diagram Designer 1.17.

FLUXOGRAMAS E REDES DE INFERÊNCIAS

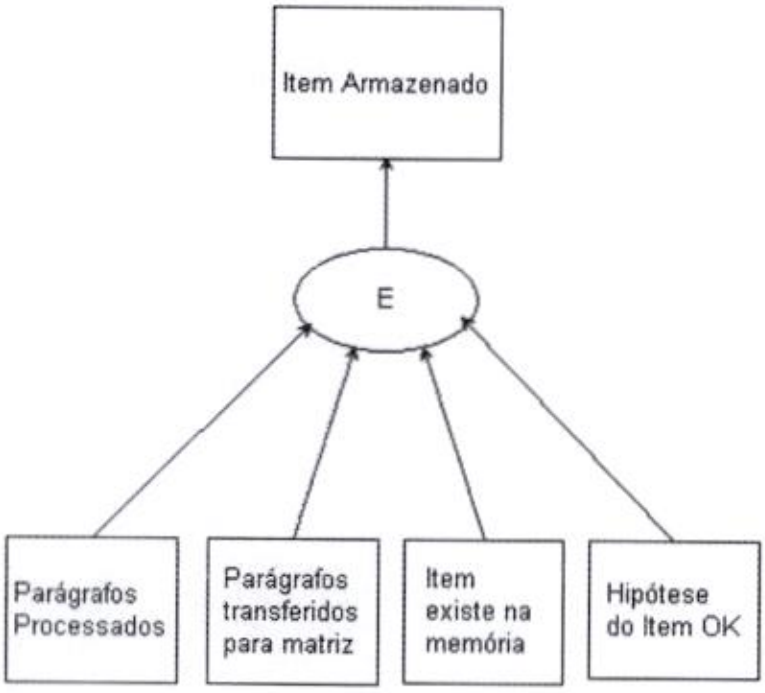
REDE DE INFERÊNCIAS PARA ARMAZENAR ITENS

REDE DE INFERÊNCIAS PARA DISPARAR QUESTIONÁRIO

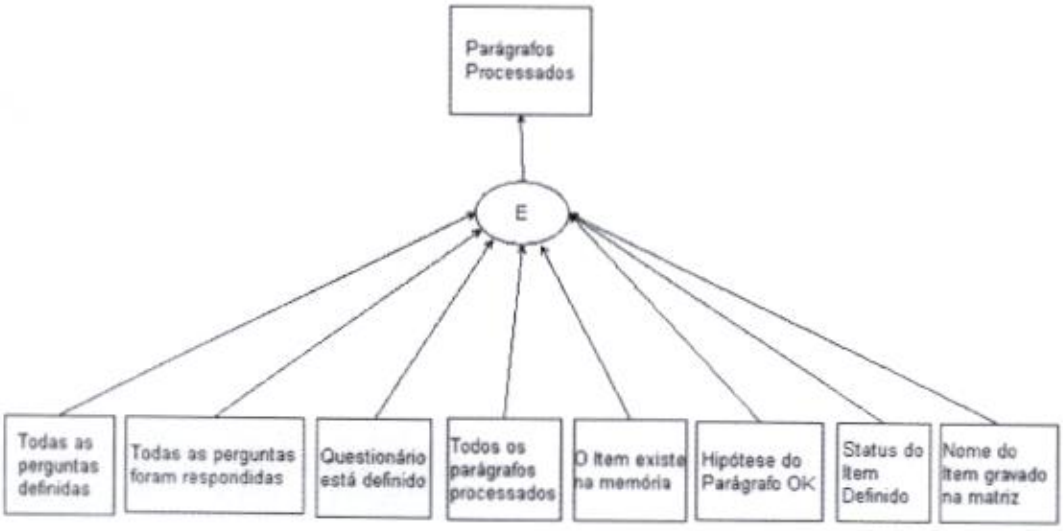


BIBLIOTECA DA ESCOLA TECNICA DA UFPR

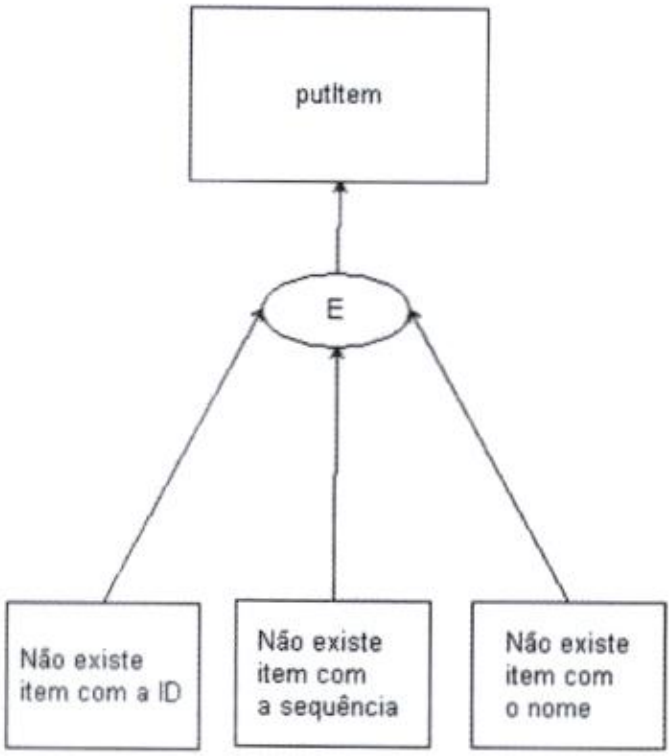
REDE DE INFERÊNCIAS PARA ITEM ARMAZENADO



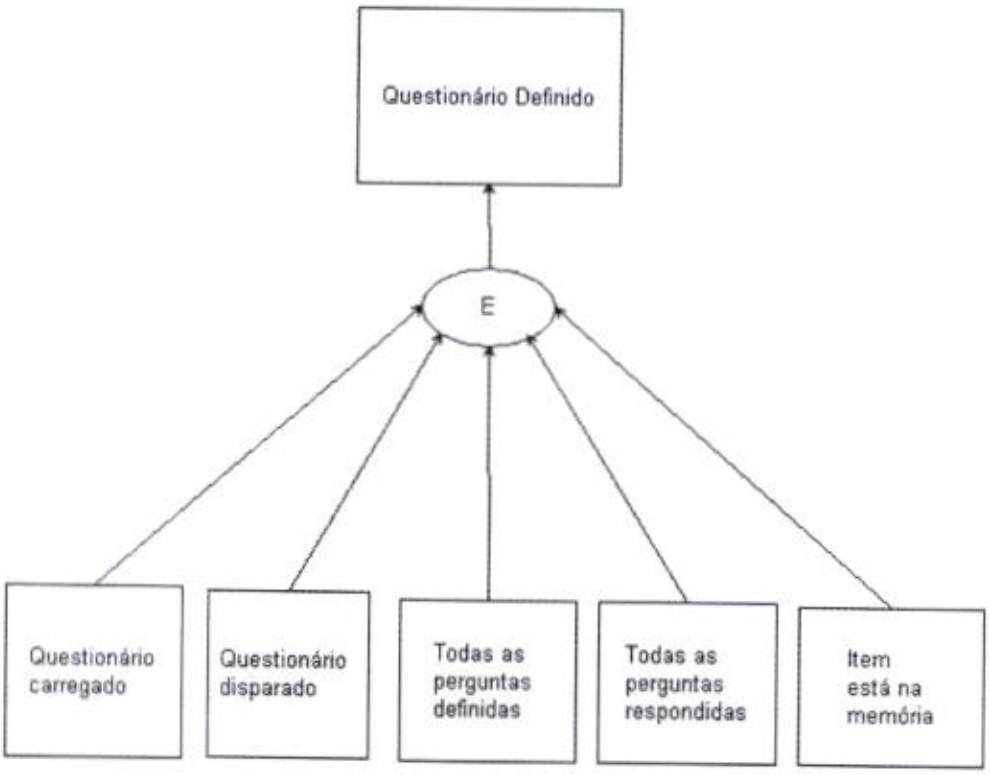
REDE DE INFERÊNCIAS PARA PROCESSAR PARÁGRAFO



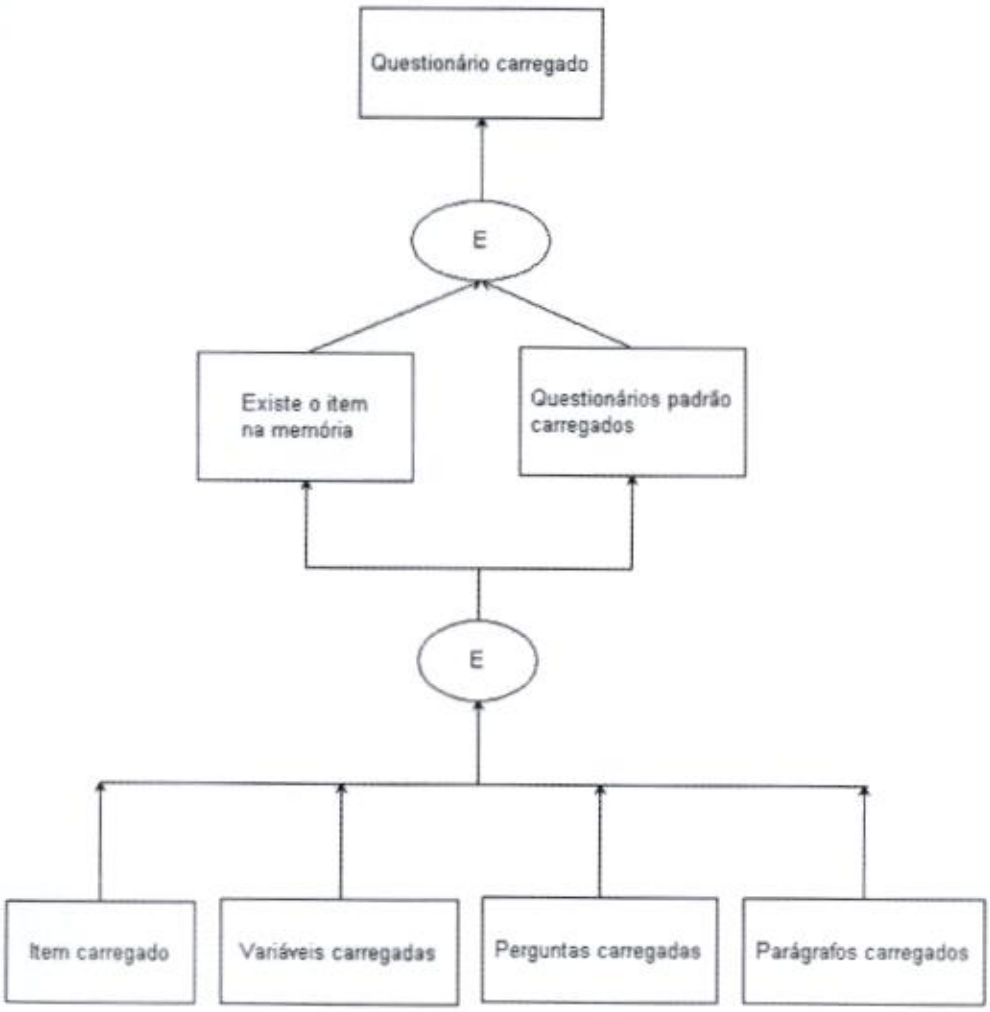
REDE DE INFERÊNCIAS PARA PUTITEM



REDE DE INFERÊNCIAS PARA QUESTIONÁRIO DEFINIDO

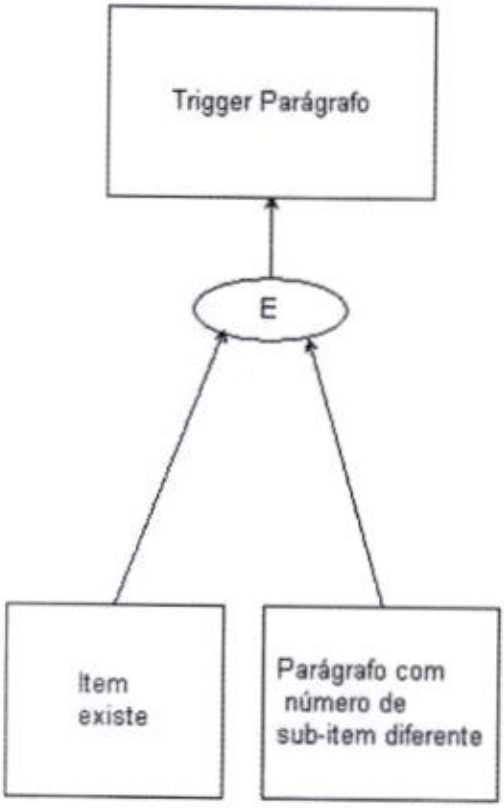


REDE DE INFERÊNCIAS PARA QUESTIONÁRIO CARREGADO

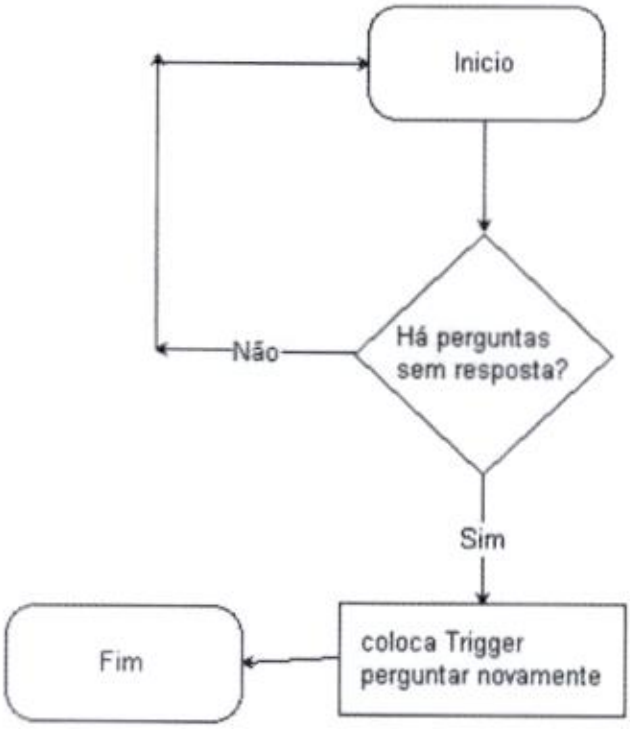


REDE DE INFERÊNCIAS PARA GRAVAR O EDITAL

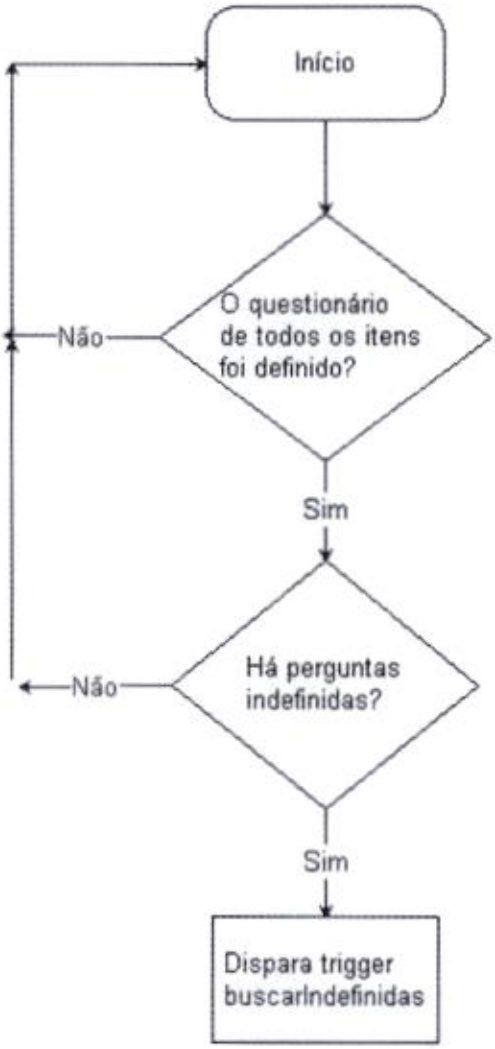
REDE DE INFERÊNCIAS PARA TRIGGER PARÁGRAFO



FLUXOGRAMA BACKGROUND FALTARESPOSTA

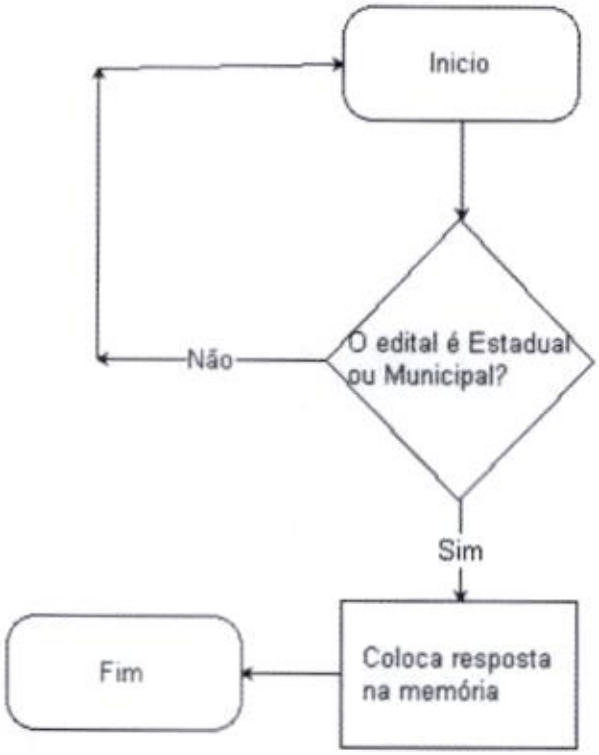


FLUXOGRAMA BACKGROUND INDEFINIDAS

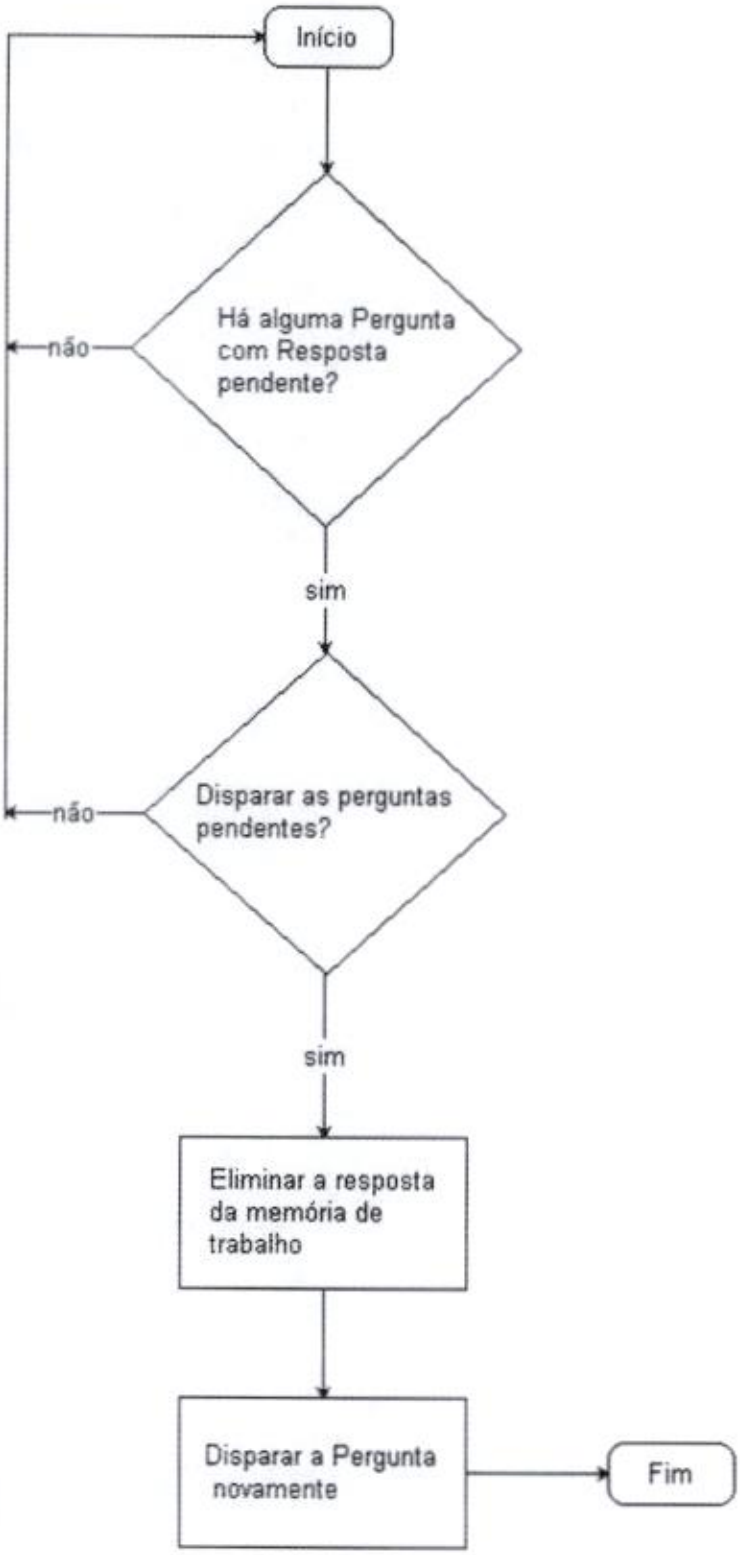


FLUXOGRAMA DEFINIR EDITAL

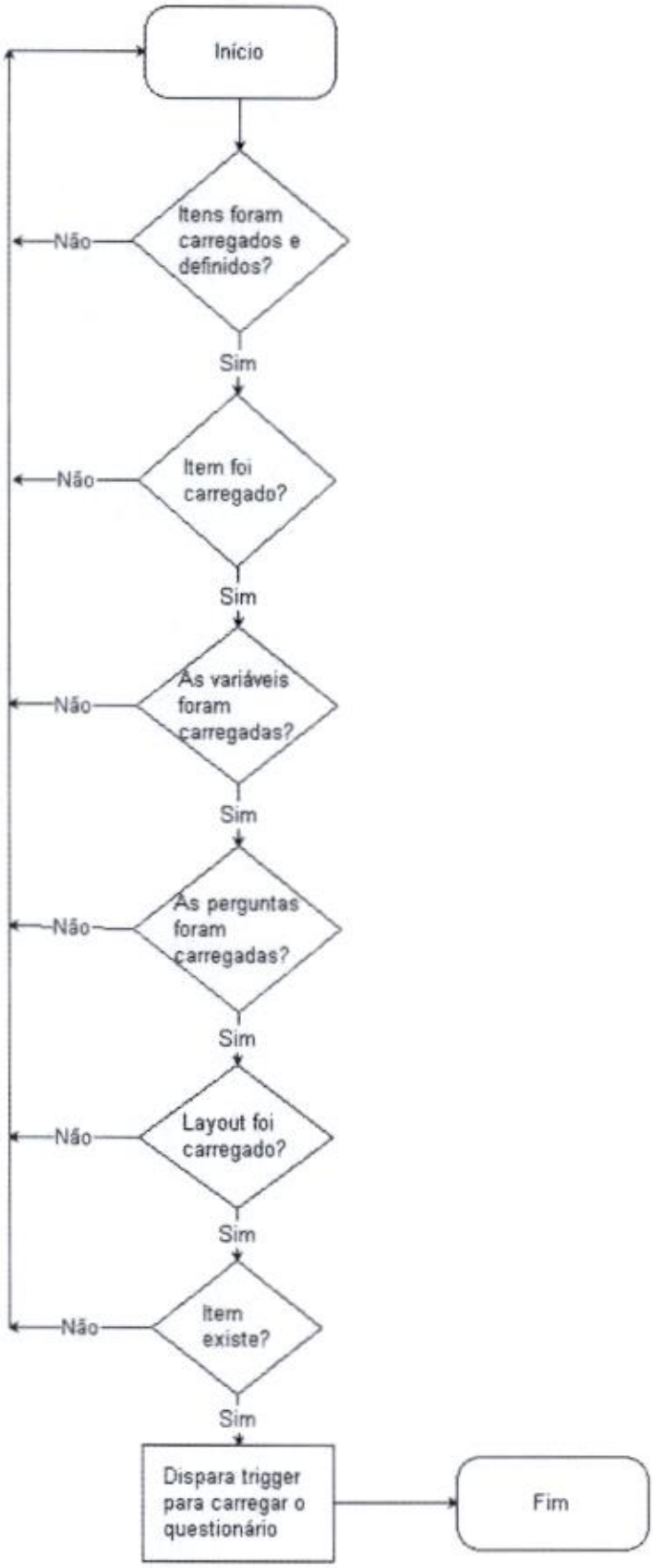
FLUXOGRAMA DETERMINA AFRO

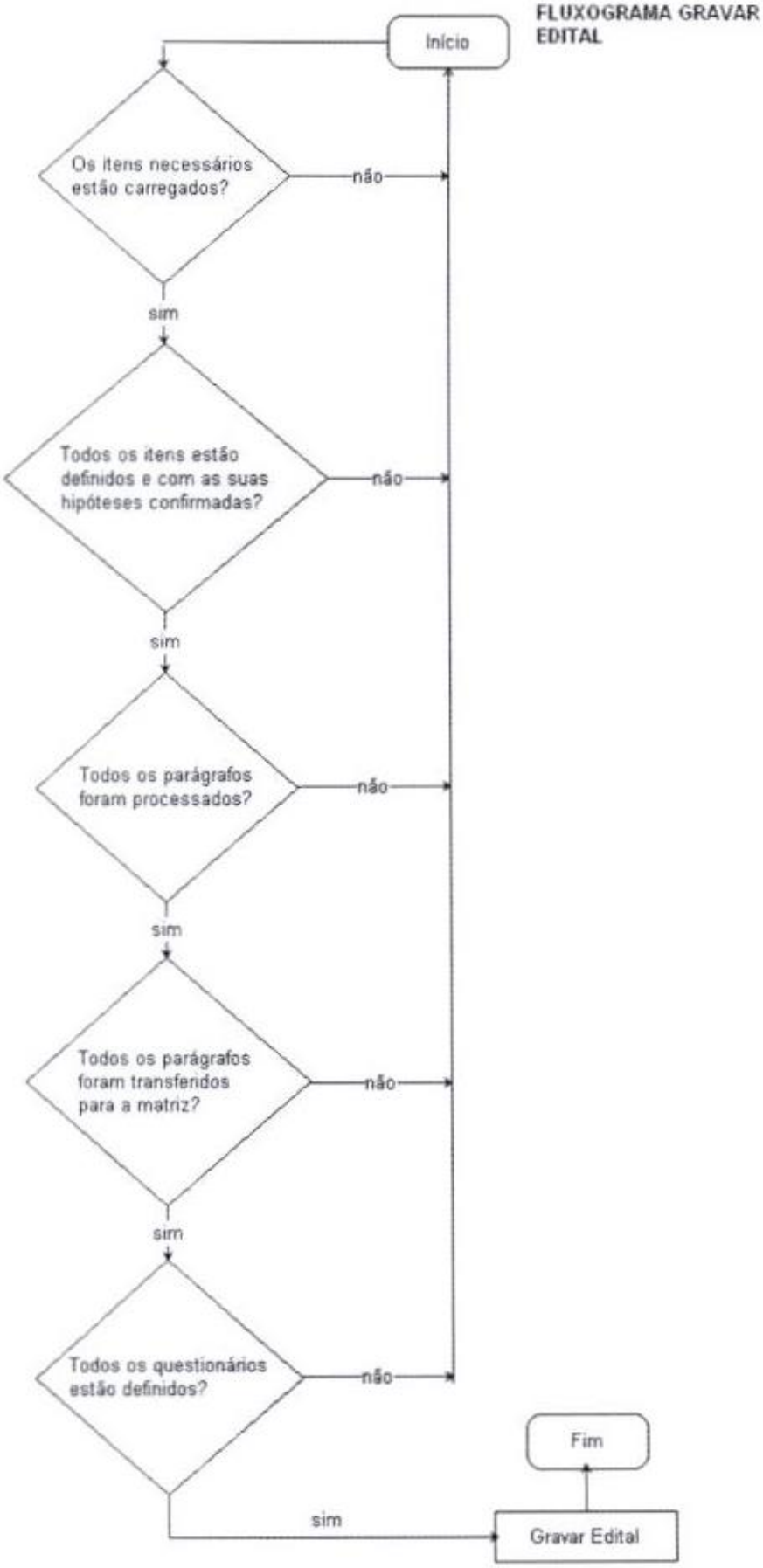


Fluxograma para disparar perguntas não respondidas

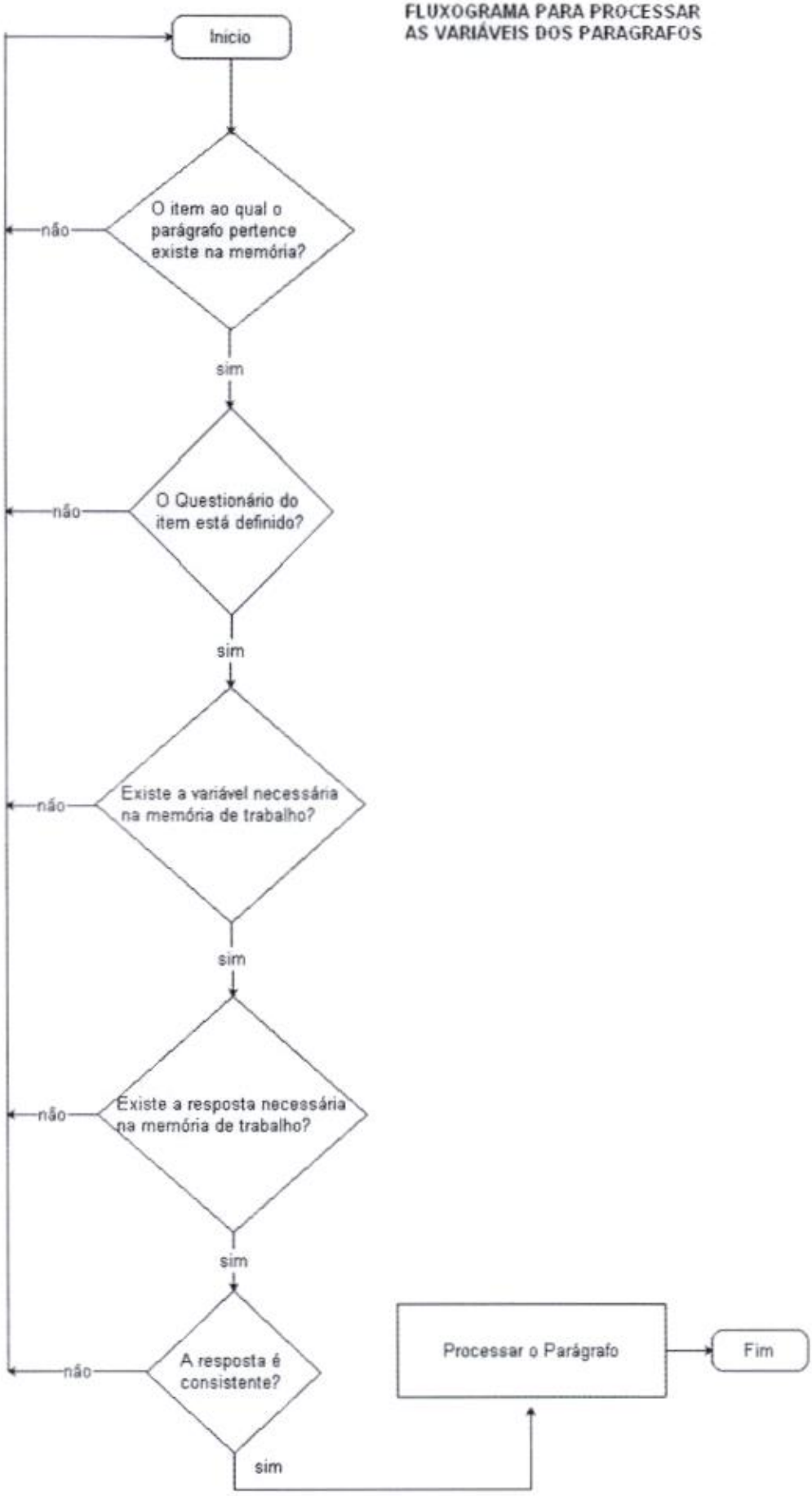


FLUXOGRAMA DISPARA QUESTIONÁRIO

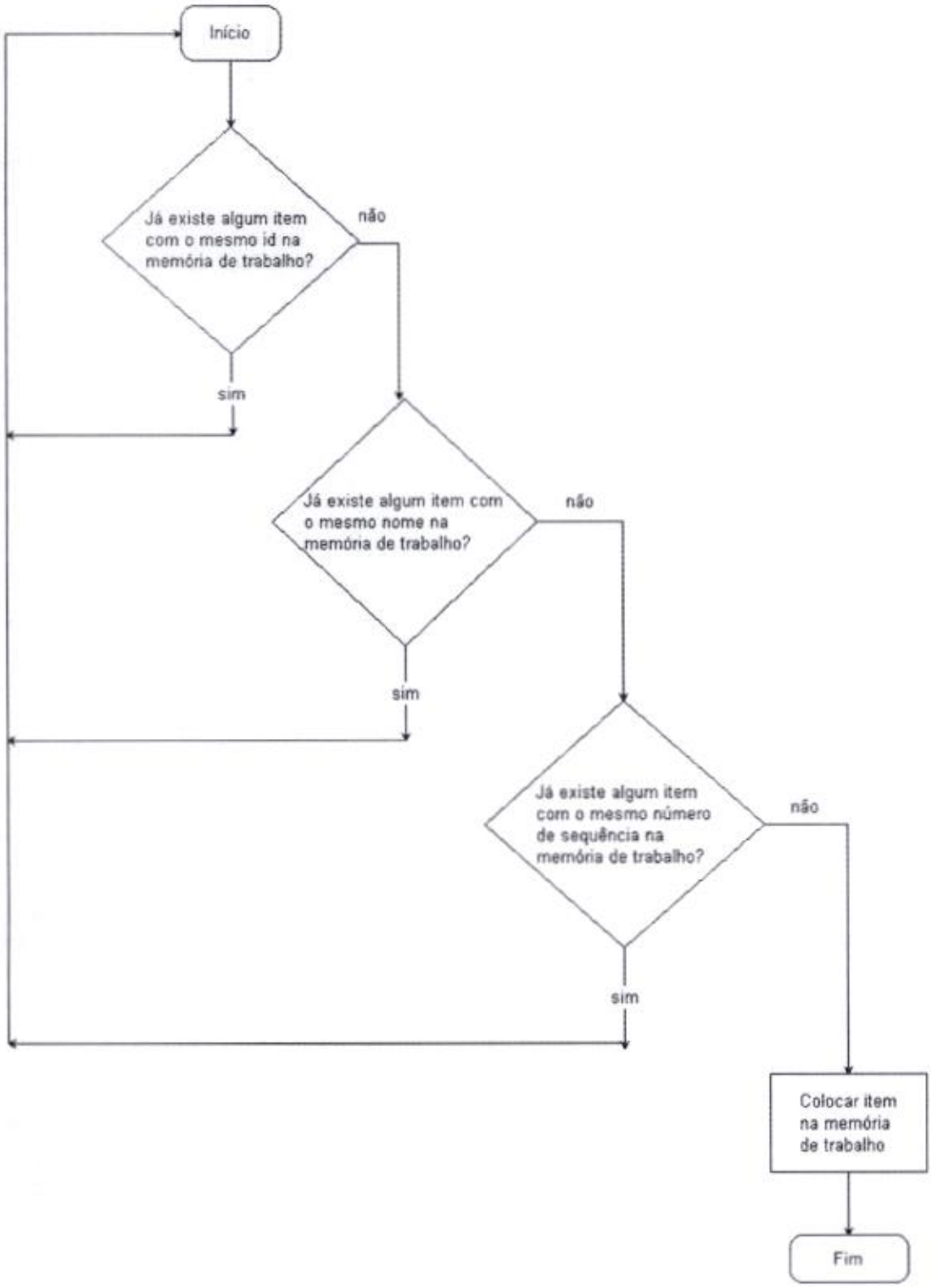



FLUXOGRAMA GRAVAR EDITAL
Início
Os itens necessários estão carregados?
não
sim
Todos os itens estão definidos e com as suas hipóteses confirmadas?
não
sim
Todos os parágrafos foram processados?
não
sim
Todos os parágrafos foram transferidos para a matriz?
não
sim
Todos os questionários estão definidos?
não
sim
Gravar Edital
Fim

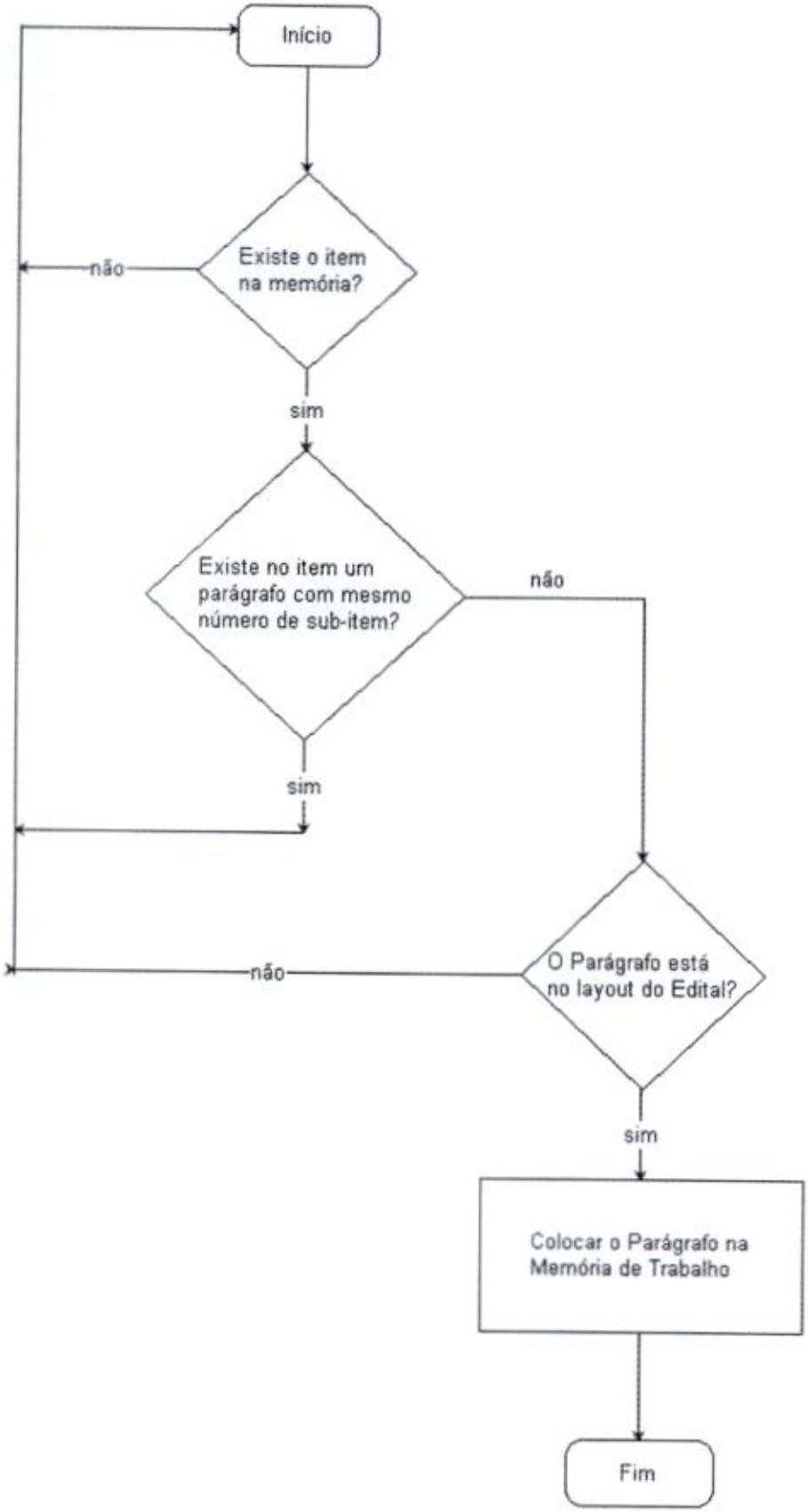
FLUXOGRAMA PARA PROCESSAR AS VARIÁVEIS DOS PARAGRAFOS

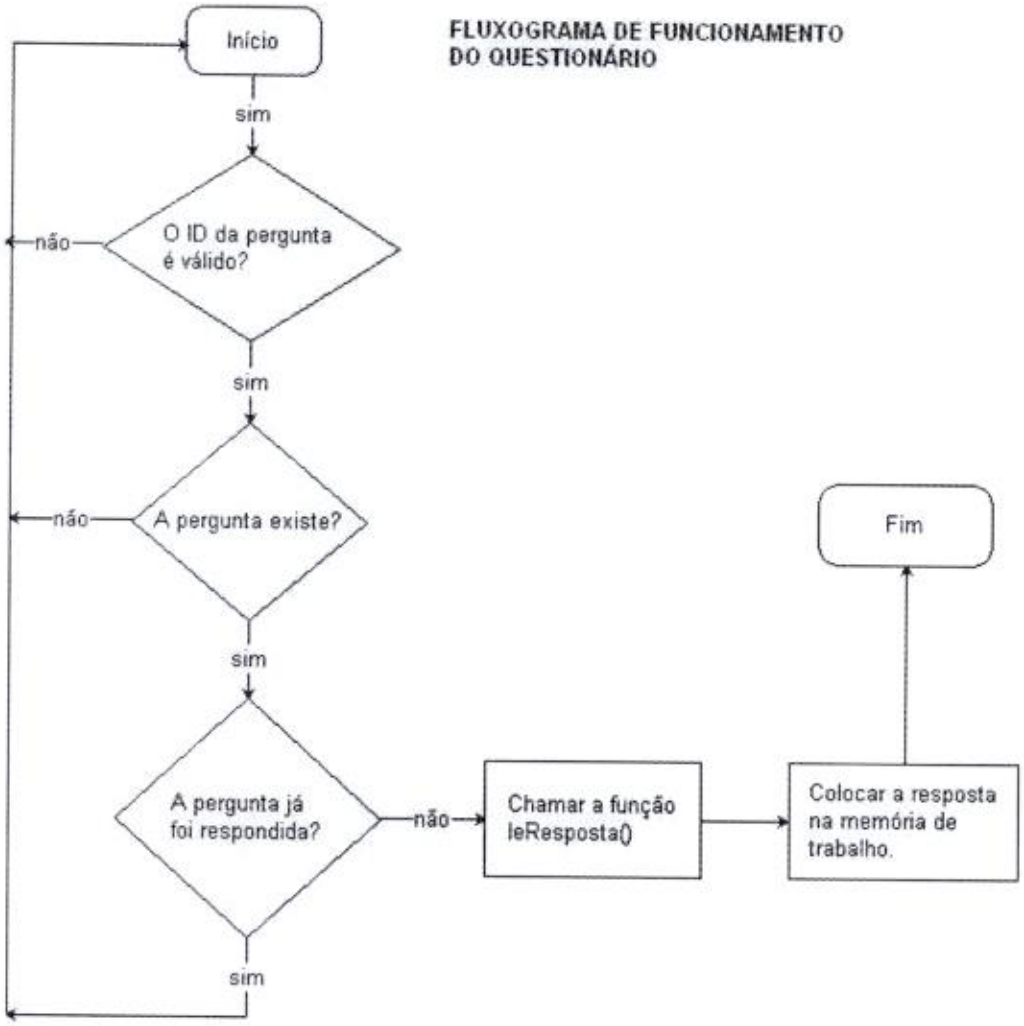


FLUXOGRAMA PARA COLOCAR NA MEMÓRIA DE TRABALHO OS ITENS DO EDITAL

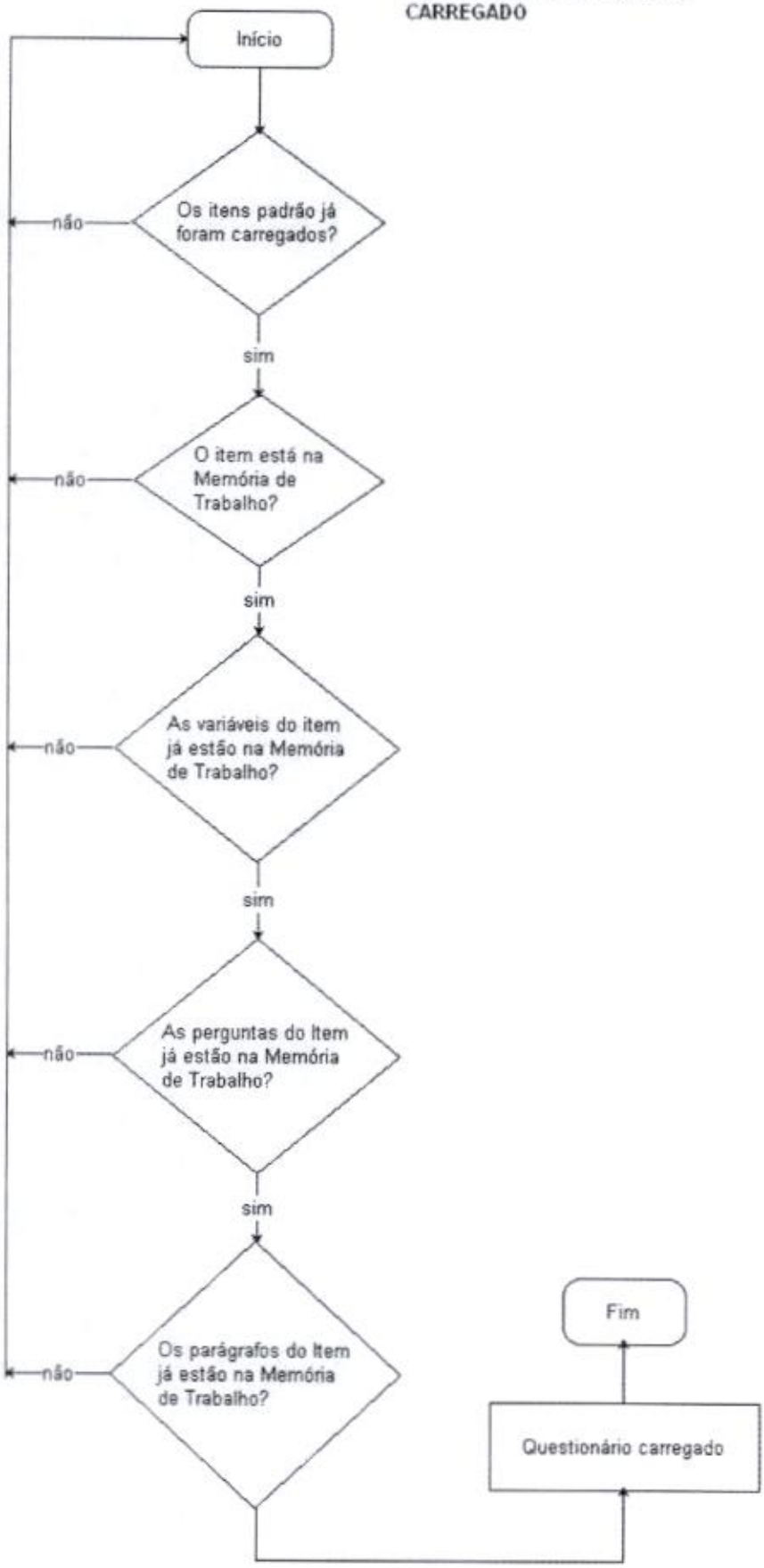


FLUXOGRAMA PARA COLOCAR O PARÁGRAFO NA MEMÓRIA

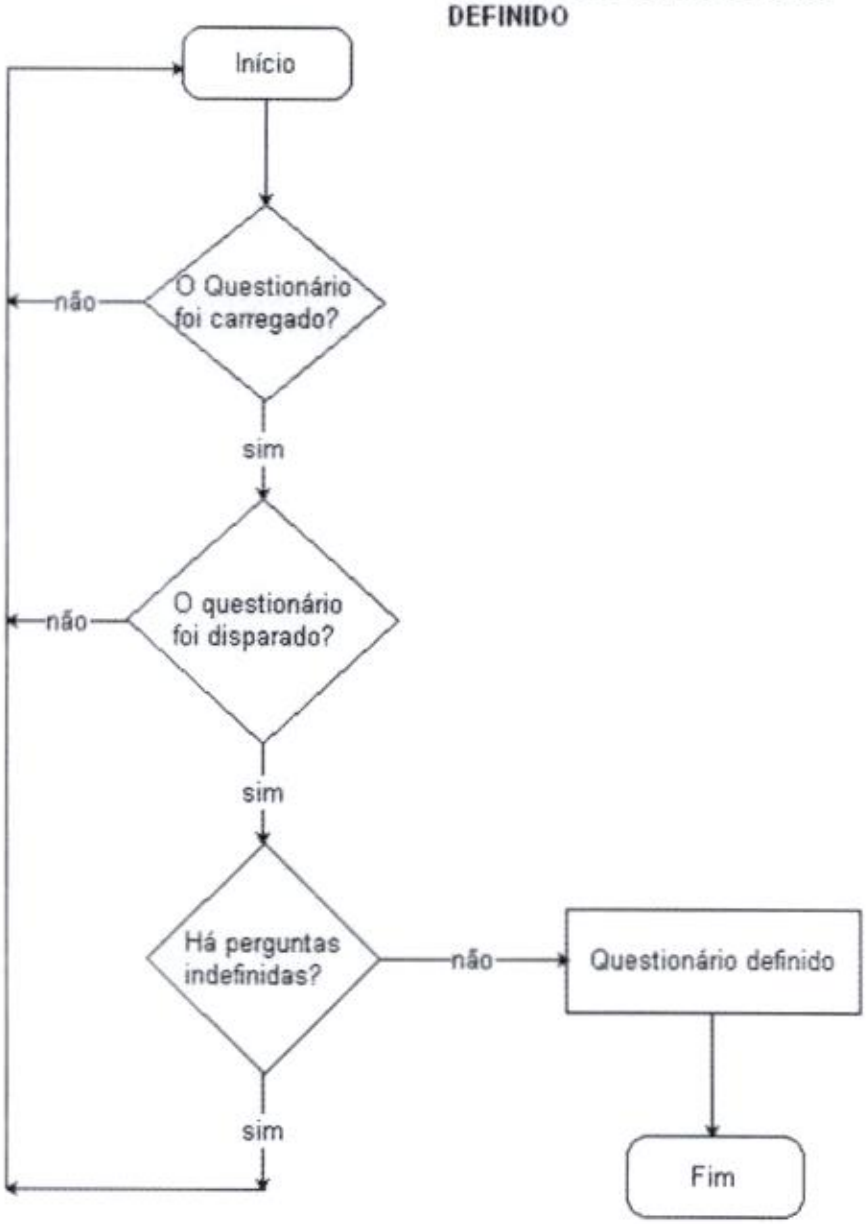



FLUXOGRAMA DE FUNCIONAMENTO
DO QUESTIONÁRIO
Início
sim
O ID da pergunta é válido?
não
sim
A pergunta existe?
não
sim
A pergunta já foi respondida?
não
sim
Chamar a função leResposta()
Colocar a resposta na memória de trabalho.
Fim

FLUXOGRAMA PARA DEFINIR QUESTINÁRIO COMO STATUS CARREGADO



FLUXOGRAMA PARA DEFINIR QUESTINÁRIO COMO STATUS DEFINIDO



CRONOGRAMA

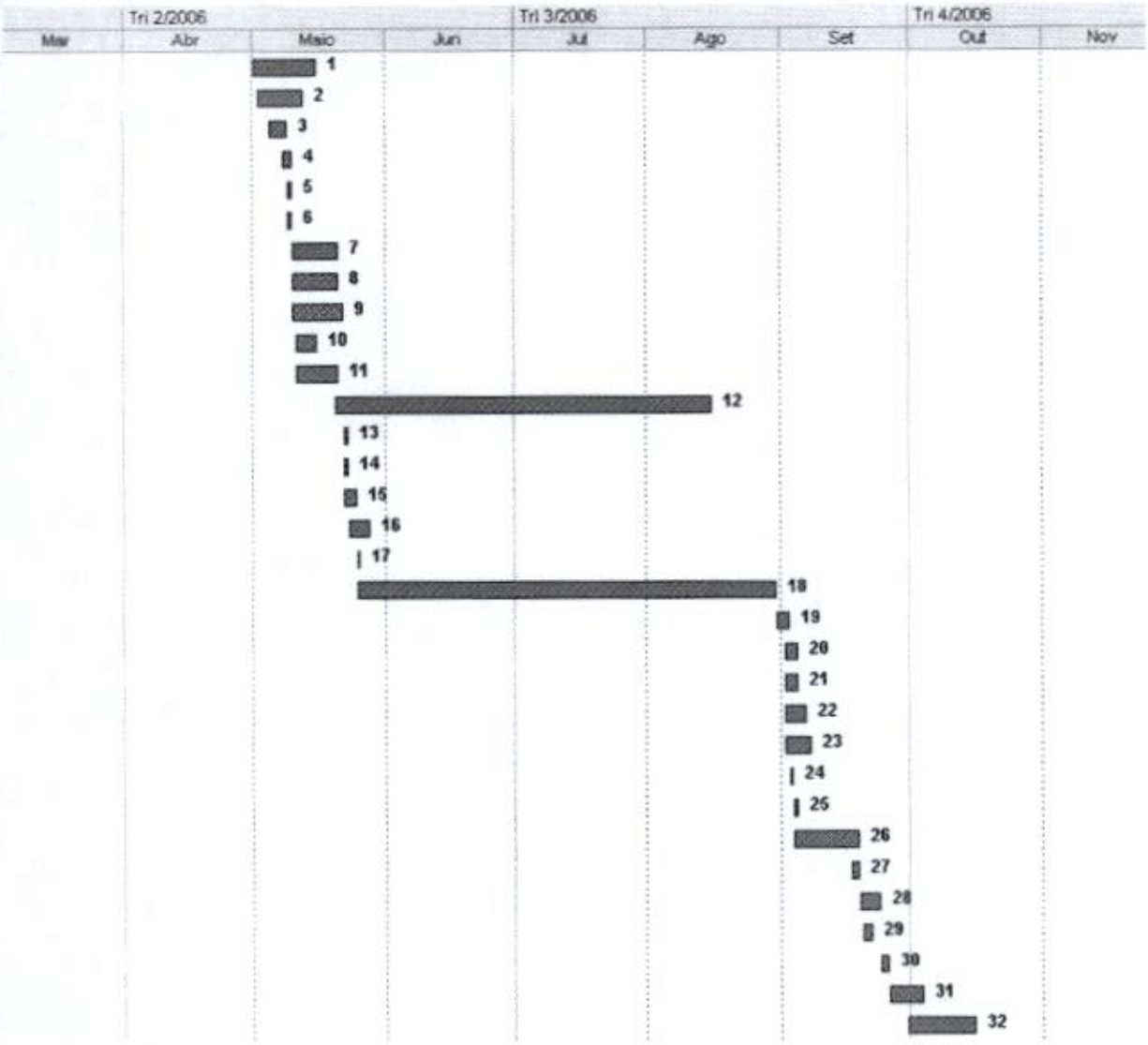
Todas as atividades que ocorreram durante a elaboração e desenvolvimento do sistema estão definidas e datadas na planilha abaixo, em seguida há um gráfico criado através do Microsoft Project que mostra todo o desenvolvimento no decorrer dos meses.

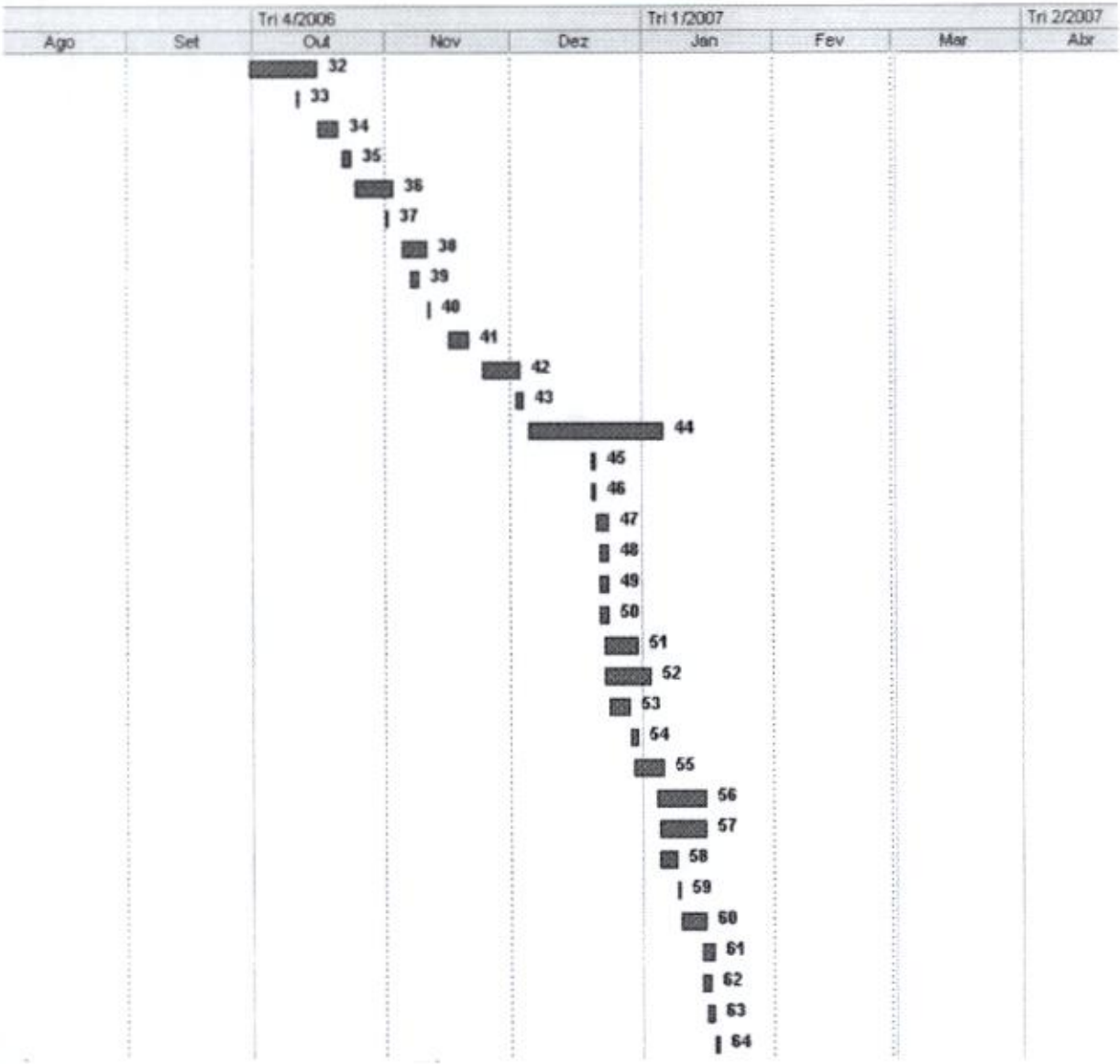
Nº	Atividade	Início	Fim
1	Estudo sobre Sistemas Especialistas	01/05/06	15/05/06
2	Teoria	02/05/06	12/05/06
3	Ferramentas disponíveis	05/05/06	08/05/06
4	Aplicações práticas	08/05/06	09/05/06
5	Primeiro encontro com o especialista	09/05/06	09/05/06
6	Análise de necessidade do especialista	09/05/06	09/05/06
7	Estudo de Editais de Concursos Públicos	10/05/06	20/07/06
8	Análise de vários editais publicados	10/05/06	20/07/06
9	Definição principais abrangências do sistema	10/05/06	21/07/06
10	Shell para desenvolvimento de Sistemas Especialistas	11/05/06	15/08/06
11	Spirit	11/05/06	20/05/06
12	JESS	20/05/06	15/08/06
13	Reunião com o especialista	22/05/06	22/05/06
14	Divisão de principais informações para o edital	22/05/06	22/05/06
15	Criação de primeiro formato padrão de edital	22/05/06	24/05/06
16	Definição da estrutura do sistema	23/05/06	27/05/06
17	Visita ao local de trabalho do especialista	25/05/06	25/05/06
18	Definição dos templates necessários	25/05/06	30/08/06
19	Definição técnicas de inferência (Backward e Forward-Chaining)	31/08/06	02/09/06
20	Definição de variáveis em edital universal	02/09/06	04/09/06
21	Inicio do planejamento do projeto	02/09/06	04/09/06
22	Criação das possíveis perguntas do edital	02/09/06	06/09/06
23	Definição de possíveis respostas	02/09/06	07/09/06
24	Reunião com o especialista	03/09/06	03/09/06
25	Alterações no edital universal	04/09/06	04/09/06
26	Mecanismo p/ disparar perguntas e obter as respostas do usuário	04/09/06	18/09/06
27	Teste do módulo criado	17/09/06	18/09/06
28	Funções para validação das respostas	19/09/06	23/09/06
29	Adaptação e criação de perguntas e respostas	20/09/06	21/09/06
30	Definição mecanismo p/ colocar perguntas necessárias na memória	24/09/06	25/09/06
31	Elaboração apresentação projeto para orientadores	26/09/06	03/10/06
32	Definição quantos e quais serão os questionários	30/09/06	15/10/06
33	Reunião com o especialista	11/10/06	11/10/06

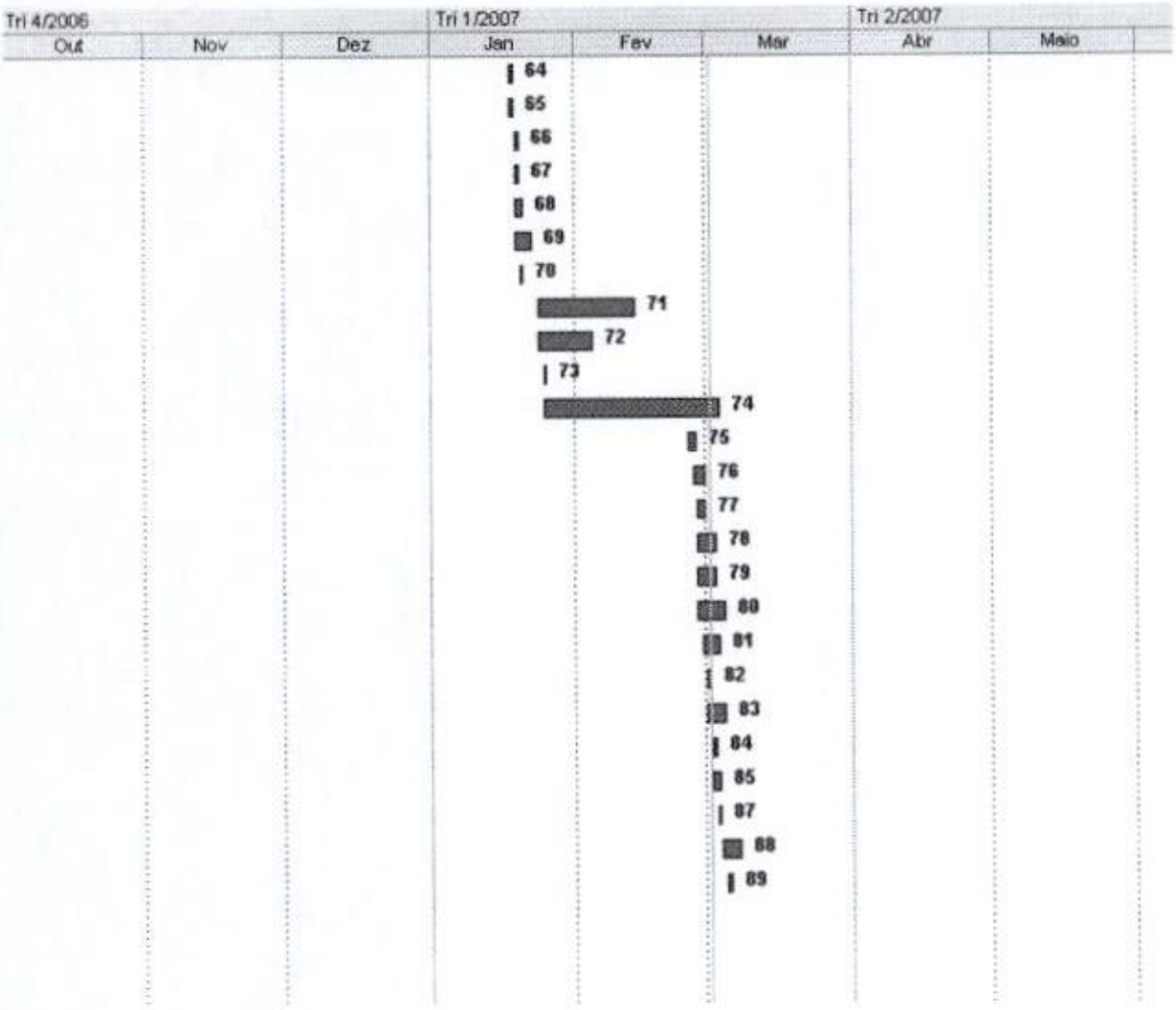
34	Definição de encadeamento lógico do sistema	16/10/06	20/10/06
35	Teste de utilização de herança para os fatos	22/10/06	23/10/06
36	Definição módulo para interação com código escrito em Java	25/10/06	02/11/06
37	Apresentação e demonstração de parte do sistema especialista	01/11/06	01/11/06
38	Definição mecanismo p/ gerar edital baseado na memória de trabalho	05/11/06	10/11/06
39	Escopo diagrama de classes e relacionamentos	07/11/06	08/11/06
40	Reunião com orientador para esclarecimento de dúvidas	11/11/06	11/11/06
41	Pesquisa de métodos e ferramentas para saída final do edital	16/11/06	20/11/06
42	Definição de classe para gravar o edital em formato RTF e PDF	24/11/06	02/12/06
43	Testes de criação de edital	02/12/06	03/12/06
44	Inicio criação de diagramas	05/12/06	05/01/07
45	Reunião com o especialista	20/12/06	20/12/06
46	Testes do sistema com o especialista	20/12/06	20/12/06
47	Sessão de consulta	21/12/06	23/12/06
48	Mecanismo p/ que possa pular perguntas e retomá-las posteriormente	22/12/06	23/12/06
49	Mecanismo p/ que o usuário possa interromper a sessão quando desejar	22/12/01	23/12/06
50	Mecanismo p/ que o usuário possa reiniciar a memória de trabalho	22/12/06	23/12/06
51	Fluxogramas (Diagramas de Atividades)	23/12/06	30/12/06
52	Estrutura para a modelagem do edital	23/12/06	02/01/07
53	Desenvolvimento da estrutura de modelagem para o layout do edital	24/12/06	28/12/06
54	Definição dos templates necessários	29/12/06	30/12/06
55	Definição técnicas de inferência (Backward e Forward-Chaining)	30/12/06	05/01/07
56	Montagem da documentação	04/01/07	15/01/07
57	Modularização do sistema	05/01/07	15/01/07
58	Módulo p/ gerenciar questionário de forma independente	05/01/07	08/01/07
59	Módulo para inicializar o sistema	09/01/07	09/01/07
60	Módulo com regras específicas para a manipulação de editais	10/01/07	15/01/07
61	Implementação de herança para o funcionamento dos módulos	15/01/07	17/01/07
62	Integridade das variáveis do sistema	15/01/07	16/01/07
63	Alteração do edital universal	16/01/07	17/01/07
64	Criação template para o fato Variável	18/01/07	18/01/07
65	Garantir a não utilização de variáveis inexistentes	18/01/07	18/01/07
66	Garantir que uma variável não identifique mais do que uma pergunta	19/01/07	19/01/07
67	Garantir que uma variável não identifique mais do que uma resposta	19/01/07	19/01/07
68	Regras para correta criação de itens e parágrafos	19/01/07	20/01/07
69	Testes de implementação do edital	19/01/07	22/01/07
70	Tratando respostas iniciando com artigos adequados	20/01/07	20/01/07
71	Modelagem da Base de Conhecimento	24/01/07	14/02/07
72	Um sistema especialista para cada item do edital	24/01/07	04/03/07
73	Salvar memória de trabalho	25/01/07	25/01/07
74	Meta-regras	25/01/07	04/03/07
75	Mecanismo para gravar os fatos da memória de trabalho	25/02/07	26/02/07
76	Mecanismo para recuperar os fatos da memória de trabalho	26/02/07	28/02/07
77	Definição da classe que implementa a interface gráfica do sistema	27/02/07	28/02/07
78	Item Especificação de cargos e vagas	27/02/07	02/03/07

79	Testes de memória de trabalho	27/02/07	02/03/07
80	Interface Gráfica	27/02/07	04/03/07
81	Integração da classe Java com o mecanismo de inferência	28/02/07	03/02/07
82	Respostas do tipo data	01/03/07	01/03/07
83	Testes de Interface	01/03/07	04/03/07
84	Validações das respostas	02/03/07	02/03/07
85	Respostas do tipo hora / web site	02/03/07	03/03/07
86	Testes de validação do edital	03/03/07	03/03/07
87	Mecanismo p/ descrição de perguntas e inferências ao longo da sessão	04/02/07	07/02/07
88	Testes do sistema com o especialista	05/03/07	05/03/07

Gráfico


Tri 2/2006
Tri 3/2006
Tri 4/2006
Mar
Abr
Maio
Jun
Jul
Ago
Set
Out
Nov
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32


Tri 4/2006
Tri 1/2007
Tri 2/2007
Ago
Set
Out
Nov
Dez
Jan
Fev
Mar
Abr
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64


Tri 4/2006
Tri 1/2007
Tri 2/2007
Out
Nov
Dez
Jan
Fev
Mar
Abr
Maio
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
87
88
89

FUNDAMENTAÇÃO TEÓRICA (AVANÇADA)

SISTEMAS ESPECIALISTAS

HISTÓRICO

Nos anos 50, os pesquisadores já haviam estabelecido as fundações da Inteligência Artificial (IA), incluindo Lógica Matemática e Teoria das Funções Recursivas, guiando a formulação de processamento de listas e da própria linguagem LISP que fornece um interpretador para desenvolver expressões simbólicas recursivas. Tais capacidades suportaram o surgimento de sistemas práticos de computação simbólica. Ao mesmo tempo, emergiram computadores interativos tornando possíveis ambientes computacionais para desenvolvimento e depuração de programas incrementais. Aproximadamente nesta mesma ocasião, psicólogos cognitivos - estudantes da forma de pensar humana - criaram caminhos padrão do processo de investigação do raciocínio, modelando o aparente processo de tomada de decisão em termos de regras de produção condicionais.

Nos anos 60, os pesquisadores de Inteligência Artificial tentaram simular o complexo processo do pensamento procurando métodos gerais para resolver uma ampla classe de problemas. Então durante a década de 70 concentraram esforços em técnicas como Representação - isto é, modo de formular o problema de maneira a tornar sua solução mais fácil e como controlá-la inteligentemente dentro da capacidade de memória do computador.

Somente no final da década fizeram a descoberta mais importante: o poder do programa em resolver problemas depende mais do conhecimento que possui do que do formalismo ou esquema de inferência empregado. Esta realização levou ao desenvolvimento de programas de computador de propósito particular, sistemas que são peritos em alguma área limitada. Estes programas são chamados "Sistemas Especialistas".

No campo de IA aplicada, após uma década de trabalho, emergiram três subcampos principais: Sistemas Especialistas, Linguagem Natural e Robótica - incluindo visão, fala e locomoção. Os sistemas especialistas começaram a surgir

comercialmente entre 1980 e 1981. A primeira companhia formada exclusivamente para produzir sistemas especialistas foi a Intelli Genetics, no campo de engenharia genética e com técnicos oriundos do Projeto de Programação Heurística da Universidade de Stanford. Assim os sistemas especialistas após terem se desenvolvido por 15 anos como mera curiosidade de IA aplicada em laboratórios de pesquisa, tornaram-se alvo de significativos esforços de desenvolvimento, tanto técnicos como comerciais. Abaixo segue algumas especificações de sistema:

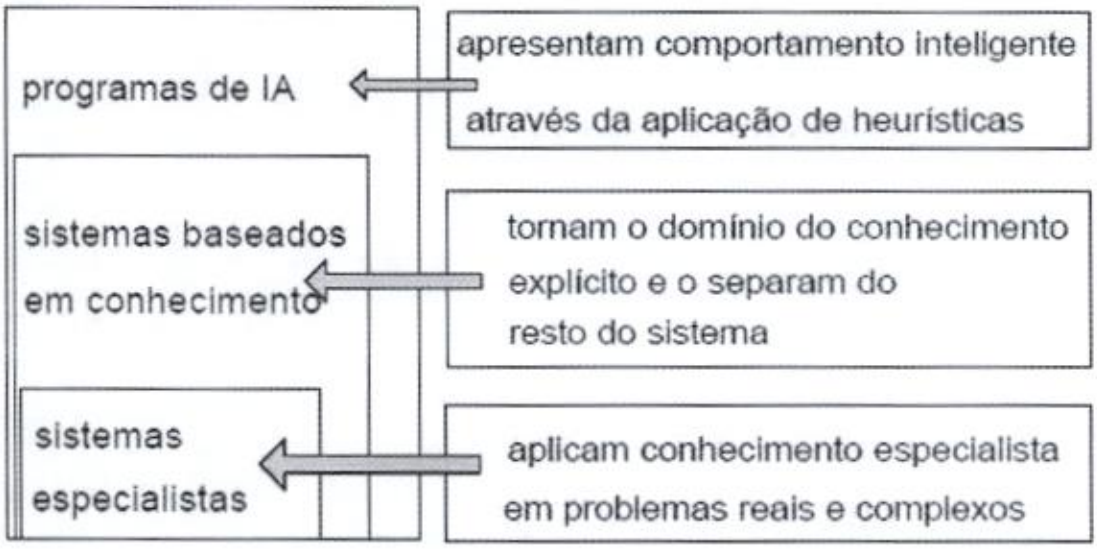


Figura I – Especificações de sistema

Recentemente, muitas organizações têm explorado esta tecnologia, ampliando suas pesquisas e começando a adaptar suas atividades para tanto. Embora sistemas especialistas e peritos reais possam em alguns casos desempenhar tarefas idênticas, as características de ambos são criticamente diversas. Mesmo havendo algumas vantagens evidentes dos sistemas especialistas, eles não poderão substituir os peritos em todas as situações devido a algumas limitações inerentes.

COMPARAÇÃO ENTRE O CONHECIMENTO HUMANO E O CONHECIMENTO ARTIFICIAL

Perecível	Permanente
Difícil de Transferir	Fácil de Transferir
Difícil de Documentar	Fácil de Documentar
Imprevisível	Consistente
Caro	Razoável
Discriminatório	Imparcial
Social	Individualizado
Criativo	Sem Inspiração
Adaptável	Inflexível
Enfoque Amplo	Enfoque Restrito
Baseado em Senso Comum	Técnico

CONCEITOS

Sistema: conjunto de elementos, materiais ou ideais, entre os quais se possa encontrar ou definir alguma relação.

Especialista: pessoa que se consagra com particular interesse e cuidado a certo estudo. Conhecedor, perito.

Sistemas especialistas são sistemas que solucionam problemas que são resolvíveis apenas por pessoas especialistas (que acumularam conhecimento exigido) na resolução destes problemas. Criados para resolver problemas e que devem apresentar um comportamento semelhante a um especialista em um determinado domínio, cujo conhecimento utilizado é fornecido por pessoas que são especialistas naquele domínio.

Um sistema especialista é um programa inteligente de computador que se utiliza de métodos inferenciais para a resolução de problemas técnicos e altamente especializados. Por utilizar-se da Inteligência Artificial, um ramo da computação que estuda a capacidade de uma máquina raciocinar e aprender como um ser humano, os sistemas especialistas interagem com seu usuário numa linguagem natural de perguntas e respostas, sugerindo e auxiliando na solução de problemas complexos. Programas de computador que tentam resolver problemas que os seres humanos resolveriam emulando o raciocínio de um especialista, aplicando conhecimentos específicos e inferências são ditos "Sistemas Especialistas".

O sistema convencional é baseado em um algoritmo, emite um resultado final correto e processa um volume de dados de maneira repetitiva enquanto que um sistema especialista é baseado em uma busca heurística e trabalha com problemas para os quais não existe uma solução convencional organizada de forma algorítmica disponível ou é muito demorada. Um sistema especialista é aquele que é projetado e desenvolvido para atender a uma aplicação determinada e limitada do conhecimento humano. É capaz de emitir uma decisão, apoiado em conhecimento justificado, a partir de uma base de informações, tal qual um especialista de determinada área do conhecimento humano. Abaixo segue uma comparação entre sistemas convencionais e especialistas:

Sistemas Especialistas	Sistemas Convencionais
Representação de Conhecimento	Estrutura d Dados
Conceitos, Relações entre Conceitos e Regras	Dados e Relações entre os Dados
Procura Heurística	Usam Algoritmos Determinísticos
Conhecimento representado explicitamente e separado do programa que o manipula e interpreta	Conhecimento embebido no código do programa
Podem e devem explicar o seu reciocínio	Explicação do raciocínio é difícil

Figura II – Comparação de sistemas

Para tomar uma decisão sobre um determinado assunto, um especialista o faz a partir de fatos que encontra e de hipóteses que formula, buscando em sua memória um conhecimento prévio armazenado durante anos, no período de sua formação e no decorrer de sua vida profissional, sobre esses fatos e hipóteses. E o faz de acordo com a sua experiência, isto é, com o seu conhecimento acumulado sobre o assunto e, com esses fatos e hipóteses, emite a decisão. Durante o processo de raciocínio, vai verificando qual a importância dos fatos que encontra comparando-os com as informações já contidas no seu conhecimento acumulado

sobre esses fatos e hipóteses. Neste processo, vai formulando novas hipóteses e verificando novos fatos, que influenciaram no processo de raciocínio. Este raciocínio é sempre baseado no conhecimento prévio acumulado. Um especialista com esse processo de raciocínio pode não chegar a uma decisão se os fatos de que dispõe para aplicar o seu conhecimento prévio não forem suficientes. Pode, por este motivo, inclusive chegar a uma conclusão errada, mas este erro é justificado em função dos fatos que encontrou e do seu conhecimento acumulado previamente.

O conhecimento necessário para atuar a esse nível, mais os procedimentos inferenciais empregados, pode considerar-se um modelo da perícia aos melhores profissionais do ramo. O conhecimento de um sistema especialista consiste em fatos e heurísticas. Os fatos constituem um corpo de informação que é largamente compartido, publicamente disponível e geralmente aceito pelos especialistas em um campo. As heurísticas são, em sua maioria privadas, regras poucos discutidas, de bom discernimento (regras de raciocínio plausível, regras de boa conjectura), que caracterizam a tomada de decisão a nível de especialista na área. O nível de desempenho de um sistema especialista é função principalmente do tamanho e da qualidade do banco de conhecimento que possui."

Um sistema especialista deve, além de inferir conclusões, ter capacidade de aprender novos conhecimentos e, desse modo, melhorar o seu desempenho de raciocínio, e a qualidade de suas decisões. Os sistemas especialistas são construções de software que os peritos em campos específicos enriquecem com seu conhecimento, capacitando um computador a auxiliá-lo num processo de tomada de decisão. A intenção em usar conhecimento simbólico para simular o comportamento dos especialistas humanos é justamente desenvolver programas que possibilitem a utilização dos conhecimentos dos especialistas através de uma máquina que permita o armazenamento e o seqüenciamento de informações e a auto-aprendizagem.

UTILIZAÇÃO

De um modo geral, sempre que um problema não pode ser algoritmizado, ou sua solução conduza a um processamento muito demorado, os sistemas especialistas podem ser uma saída, pois possuem o seu mecanismo apoiado em

processos heurísticos. Preservar e transmitir o conhecimento de um especialista humano em uma determinada área. Um sistema especialista não é influenciado por elementos externos a ele, como ocorre com o especialista humano, para as mesmas condições deverá fornecer sempre omesmo conjunto de decisões.

Para que um sistema especialista seja eficaz, as pessoas têm de ser capazes de interagir com ele facilmente. Para facilitar esta interação os sistemas devem ser capazes de explicar seu raciocínio e consequentemente o processo de raciocínio deve proceder em etapas compreensíveis em que o metaconhecimento suficiente (conhecimento sobre o processo de raciocínio) esteja disponível para que as explicações dessas etapas possam ser geradas;

PRINCIPAIS BENEFÍCIOS

- Velocidade na determinação dos problemas;
- A decisão está fundamentada em uma base de conhecimento;
- Segurança. A confiabilidade do software é um importante fator. A possibilidade de exame da montagem da regras geradoras de uma conclusão, permite encontrar as inferências indutivas e suas justificações. Quando uma pessoa faz o mesmo trabalho intelectual, mesmo que tenha uma perícia e um certo grau de inteligência, chega-se que ela esquece ou distorce a realidade, cometendo assim um erro. Se o software comete uma falha, esta pode ser consertada, pois foi introduzida durante a concepção e assim a validação não é feita corretamente e o resultado não é apresentado;
- Exige pequeno número de pessoas para interagir com o sistema;
- Estabilidade;
- Dependência decrescente de pessoal específico.
- Flexibilidade;
- Evita interpretação humana de regras operacionais. Softwares construído com linguagens clássicas do processo de dados, requerem um conhecimento rígido, encaixados uns aos outros, em uma determinada ordem, enquanto as regras de uso de um sistema especialista são flexíveis, graças ao seu motor de inferência;
- A separação da informação e o seu tratamento, facilita a restrição à regras inadequadas ou obsoletas, e a sua modificação inclui a evolução e a introdução

de novas regras. O conhecimento considerado no fato geralmente pode ser atualizado;

- Reduz os custos com funcionários especializados. Com sistemas especialistas uma tarefa de grande complexidade pode ser realizada por funcionários que não sejam *experts*, mas funcionários especializados apenas. Com isso, os especialistas podem ficar dedicados a tarefas que realmente sejam de altíssima complexidade;
- Sistemas especialistas podem freqüentemente fazer melhor que um especialista. Eles podem combinar a capacidade de um especialista com a velocidade e precisão de uma máquina. Eles podem fazer uma análise detalhada e completa de uma situação que dificilmente um humano seria capaz de fazer. Segundo, se um sistema foi bem estruturado, ele não cometerá erros que eventualmente um humano pode cometer.

PROBLEMAS ENFRENTADOS

- Fragilidade. Como os sistemas especialistas somente têm acesso a conhecimento altamente específicos do seu domínio, não possuem conhecimentos mais genéricos quando a necessidade surge;
- Falta de metaconhecimento. Geralmente não possuem conhecimentos sofisticados sobre sua própria operação, portanto não conseguem raciocinar sobre seu próprio escopo e restrições. A aquisição do conhecimento continua sendo um dos maiores obstáculos à aplicação de tecnologia dos sistemas especialistas a novos domínios;
- É difícil extrair conhecimento do especialista;
- Validação. A medição do desempenho de sistemas especialistas é muito difícil porque não sabemos quantificar o uso de conhecimento;

FERRAMENTAS PARA IMPLEMENTAÇÃO DE SE

SHELL de Sistemas Especialistas

Inicialmente, os sistemas especialistas eram criados do nada. Mas devido ao fato de os sistemas serem construídos como um conjunto de representações declarativas (em sua maioria, regras) combinadas com um interpretador dessas representações, era possível separar o interpretador do conhecimento específico do domínio da aplicação e assim criar um sistema que podia ser usado para elaborar novos sistemas especialistas através da adição de novos conhecimentos, correspondentes ao novo domínio do problema. Estes interpretadores resultantes são chamados de "SHELLs", que servem de base para muitos dos sistemas especialistas que estão sendo desenvolvidos. O nome shells vem do fato de tornar transparente as dificuldades inerentes à implementação de uma aplicação em Inteligência Artificial: escolha da representação de conhecimento, do método de busca, ferramentas para encontrar erros, etc.

A construção de sistemas especialistas envolvem investimentos em profissionais de diversas áreas, implicando em riscos maiores para a produção. Uma solução para este problema é o uso de ferramentas capazes de prototipar, avaliar e implementar o projeto de um sistema, os chamados "SHELLs". Procura-se, assim, diminuir a necessidade de recursos necessários, entre eles o tempo envolvido e tenta-se conciliar a alta produtividade e a versatilidade de ferramentas qualificadas.

Um shell é uma ferramenta genérica para implementação de bases de conhecimento. Que de tal modo, o desenvolvedor não se preocupa com a máquina de inferência, apenas com o conjunto de regras que modelam o conhecimento desejado. Normalmente um SHELL inclui:

- Ferramentas para introduzir conhecimento na base de conhecimento;
- Um ou vários motores de Inferência;
- Mecanismos para tratar imprecisão;
- Mecanismos para tratar incertezas;
- Mecanismos de seguir passo a passo o funcionamento do programa de modo a encontrar possíveis erros;
- Uma comunicação homem-máquina.

Exemplos de SHELLs para a construção de sistemas especialistas:

Expert Sinta: sua finalidade é embutir dentro de seus próprios ambientes, meios para construção de interfaces gráficas com o usuário final, além de mecanismos de troca de dados com outras aplicações.

KappaPC: permite escrever aplicações em um ambiente gráfico e gerar códigos padrão ANSI C. Ele é uma ferramenta de auxílio na construção de sistemas baseados em conhecimento, usando frames, regras de produção, e programação orientada a objetos.

VP-Expert: sistema baseado em regras. Usa o raciocínio para frente e possui mecanismos embutidos para o tratamento de imprecisão com o paradigma dos fatores de confiança.

Advisor: ambiente de desenvolvimento baseado na tecnologia de regras de negócio, construído 100% Java.

Expert: conhecido também como Nexpert, é uma ferramenta de desenvolvimento para ambientes Windows e Unix, que fornece a gerentes e desenvolvedores a capacidade de integrar a experiência do negócio.

FuzzyClips: utiliza-se da teoria e dos cálculos da Lógica Fuzzy, para representar o conhecimento.

Clips: é a abreviação de 'C Language Integrated Production System', foi projetado pela NASA/Johnson Space Center.

Jess: Evolução do CLIPS. Baseado em regras inteiramente desenvolvidas em Java.

CLIPS

Por volta de 1985, a NASA decidiu unificar o desenvolvimento interno de sistemas especialistas. O sistema usado até então, não estava definido para todas as plataformas utilizadas pela companhia, especificamente PCs e Macintoshs. Como resultado, o setor de tecnologia de software do Johnson Space Center criou um clone das capacidades de encadeamento progressivo e da sintaxe de ART, introduzindo o CLIPS (*C Language Integrated Production System*) no domínio público.

CLIPS rapidamente conseguiu um grande número de usuários, principalmente no meio acadêmico. Sua sintaxe, parecida com a de OPS5, em nada se assemelhava à da linguagem hospedeira (C). Entretanto, a sua semelhança com a sintaxe de outras linguagens usadas pela comunidade de Inteligência Artificial (notadamente LISP) ajudou na proliferação de seus usuários.

APRESENTAÇÃO

A representação de conhecimento em CLIPS pode ser feita de três formas: regras (de produção), usadas principalmente para a representação de conhecimento heurístico baseado em experiência; funções específicas e funções genéricas, para representação de conhecimento procedimental; ou programação orientada a objetos, também usada na representação de conhecimento procedimental (com uma certa carga declarativa, dada a possibilidade da definição de relações entre as classes). Apesar de poder ser usado em conjunto com linguagens de programação convencionais (como C, Ada ou C++), CLIPS define uma linguagem de programação própria (COOL, ou *CLIPS Object-Oriented Language*). De fato, uma aplicação pode ser definida inteiramente em CLIPS (ou COOL), que também provê funções de linguagens convencionais, tais como funções de entrada/saída e formatação de texto. Com esta concepção, o mundo de objetos CLIPS pode ser completamente dissociado do mundo dos objetos da linguagem de programação hospedeira.

Em CLIPS há uma diferença entre tipos estruturados de dados e classes, de forma similar à diferença entre os construtores struct e class em C++. Tipos, ou *templates*, definem um agrupamento de valores, como o construtor literalize de OPS5 (com algumas novas funcionalidades, como a possibilidade do valor dos atributos serem fatos, e não apenas valores atômicos).

Fatos são adicionados à memória de trabalho através da utilização do comando assert. Da mesma forma que OPS5, atributos não inicializados recebem o valor nulo, representado nil.

Classes são definidas através do construtor defclass, através do qual o usuário pode informar maiores detalhes sobre a estrutura sendo criada, como a hierarquia da classe sendo criada (através do comando is-a), o tipo da classe (i.e., se ela é abstrata ou concreta), métodos de acesso para os campos, valores *default*, entre outros. Além disso, ainda é possível se definir novas mensagens a que objetos da classe irão responder (ou métodos de acesso, dependendo da taxonomia utilizada).

Objetos CLIPS são criados através do comando make-instance. A contrário dos fatos, os objetos criados possuem um *nome*, pelo qual podem ser referenciados posteriormente.

Fatos e objetos convivem em uma certa harmonia no mundo CLIPS. Classes e *templates* podem ter o mesmo nome, em um dado instante podem existir na memória de trabalho tanto fatos e objetos, as regras do sistema podem filtrar tanto fatos quanto objetos, enfim, os dois mundos (fatos e objetos) coexistem naturalmente em CLIPS.

Regras CLIPS são definidas de forma similar a OPS5, através do comando defrule.

O comando printout de CLIPS é usado para o envio de dados para a saída, no dispositivo especificado em seu primeiro argumento. A saída padrão (console) é representada por t.

O controle de disparo das regras CLIPS é baseado em uma das 7 estratégias de resolução de conflitos suportadas pelo sistema. Para cada regra, pode-se especificar uma prioridade (ou *saliência*, no vocabulário de CLIPS), e dentre as regras de mesma saliência, o sistema utiliza a estratégia escolhida para escolher a próxima regra a ser disparada. As estratégias existentes incluem a de profundidade (em que regras recém ativadas têm maior prioridade que as mais antigas), a de largura (o contrário da anterior, apesar infeliz escolha de terminologia), a de simplicidade (em que regras em que precisam ser feitas menos comparações adquirem maior prioridade), a de complexidade (o contrário da anterior), a aleatória, a MEA (onde regras com objetos mais recentes adquirem maior prioridades) e a LEX (o contrário da anterior).

CRÍTICA

CLIPS é uma referência importante em EOOPS por diversos motivos. Foi o primeiro sistema amplamente aceito (especialmente no meio acadêmico) que superasse as limitações de OPS5. Através da utilização do algoritmo de casamento de padrões Rete, o sistema apresentava um ótimo desempenho para aplicações de porte razoável. Além disso, seu número de usuários cresceu bastante, talvez por ser patrocinado por um órgão forte no desenvolvimento de sistemas especialistas, a NASA. No entanto, com o corte de fundos para o desenvolvimento da linguagem há alguns anos, CLIPS raramente é usado para fins comerciais, principalmente devido

à falta de suporte. O sistema ainda é, todavia, bastante usado em aplicações acadêmicas, especialmente em cursos de graduação.

A integração de CLIPS com as linguagens hospedeiras, apesar de existir, está longe de ser completa. A sintaxe das regras está muito mais próxima à de LISP que a de C, dificultando o seu aprendizado por programadores desta linguagem (problema conhecido por *Impedance Mismatch*).

Um outro problema de integração é o fato dos objetos do sistema existirem em um mundo à parte do mundo dos objetos da linguagem, apesar de poder haver ligação entre eles. Isso é decorrente da filosofia de CLIPS de que tudo pode ser feito em COOL. O grande problema desta abordagem (que é o mesmo da criação de uma nova linguagem) é que todos os serviços necessários para o desenvolvimento de aplicações tem que ser reconstruídos nesta linguagem. Mesmo utilizando os serviços da linguagem hospedeira, o usuário tem que realizar alguma conversão entre os dois mundos, dificultando a reutilização.

JESS

No início de 1997, o pesquisador Ernest J. Friedman-Hill, do laboratório Sandia da Califórnia, criou um clone de CLIPS para a linguagem Java. Foi a primeira proposta divulgada de integração entre sistemas de produção e a linguagem, que na época já aparecia como uma idéia viável, principalmente pelo apelo da Internet.

APRESENTAÇÃO

A idéia de JESS (*The Java Expert System Shell*) é a conversão de um subconjunto (inicialmente) de CLIPS para Java, mantendo a mesma filosofia do sistema desenvolvido pela NASA. Inicialmente, o sistema apresentava diversas limitações: não suportava naturalmente a herança de objetos, apresentava diversos *bugs* documentados, entre outras. Entretanto, a versão atual (6.0a2) já resolveu grande parte destes problemas, possibilitando, inclusive, a utilização de raciocínio com encadeamento regressivo, serviço não suportado por nenhum dos outros sistemas estudados.

A popularidade de JESS cresceu grande parte devido à base de usuários de CLIPS que desejavam migrar de soluções baseadas em C/C++ para Java (e utilizar os serviços oferecidos por esta linguagem). Existem milhares de usuários do sistema, que conta com uma lista de discussão relativamente movimentada, fazendo com que o suporte ao uso deste sistema seja bastante efetivo.

Assim como CLIPS, JESS define toda uma linguagem para ser usada na criação de aplicações, e não apenas de regras. Comandos de criação de interfaces gráficas, comunicação, ou entrada/saída são disponibilizados nesta linguagem, enfim, toda uma aplicação pode ser desenvolvida dentro do *shell* de JESS.

JESS pode utilizar como elementos de sua memória de trabalho objetos Java. No entanto, existe uma restrição para os objetos Java que podem ser usados. Como não existe nenhuma forma explícita de se avisar ao sistema quando um objeto é modificado (como o modified de NéOpus ou o modify de RAL/C++), os objetos devem ser capazes de avisar quando uma de suas propriedades é alterada. Em outras palavras, para uma propriedade poder ser utilizada na verificação da condição de alguma regra, ela deve ser uma *bound property*, de acordo com a nomenclatura do padrão de componentes de Java, JavaBeans.

O sistema JESS utiliza a estratégia de interpretação de regras, fazendo com que ele seja também um excelente ambiente de *scripting* para Java. A partir da sua linha de comando, usuário é capaz de criar objetos Java ou invocar métodos sem ter que compilar nenhuma linha de comando.

CRÍTICA

Apesar de ser o primeiro sistema especialista definido para Java, JESS sofre dos mesmos problemas de CLIPS, basicamente a baixa integração do sistema com a linguagem hospedeira. Ainda existe a separação dos mundos de objetos Java e o de objetos (ou fatos) JESS, e a sintaxe utilizada na definição das regras em nada se assemelha à da linguagem hospedeira, fazendo com que o usuário, tipicamente um programador Java, tenha que passar um tempo maior no entendimento do sistema.

A necessidade que os objetos sejam capazes de avisar sobre modificações ocorridas em suas propriedades também limita a utilização do sistema, que fica impossibilitado, por exemplo, de utilizar algumas classes definidas por terceiros

(aquelas que não notificam a mudança de suas propriedades), uma prática bastante comum no desenvolvimento orientado a objetos.

A eficiência de JESS também é um pouco prejudicada por ser um sistema interpretado: ele não toma proveito dos compiladores JIT que já se tornaram padrão para Java. A utilização de um mundo de objetos a parte dos objetos de Java também causa problemas para JESS. Um String em JESS consiste em um objeto da classe jess.Value que contém uma String de Java. Com isso, uma chamada ao método substring em Java cria um objeto como resultado, enquanto que em JESS dois objetos são criados. O número 5 em Java é um tipo primitivo – pode ser armazenado em um registro – enquanto que em JESS é um objeto que contém o número 5 nele. Com isso, se em Java a operação "2 + 3" não cria nenhum objeto, (+ 2 3) em JESS cria um. Como a criação de objetos é uma operação bastante cara em Java, a performance do sistema fica prejudicada.

ALGORITMO RETE

Há uma estratégia definida no algoritmo de Markov em função das regras de maior prioridade. Tanto quanto se necessite, as regras de maior prioridade são utilizadas, caso contrário, é selecionada a regra de prioridade imediatamente inferior e assim sucessivamente. Esse procedimento traz uma grande ineficiência para Sistemas Especialistas com muitas regras, uma vez que para cada inferência, todas as regras serão verificadas, perdendo-se em tempo de processamento. Esse problema ficará mais acentuado se quisermos implementar Sistemas Especialistas com centenas ou milhares de regras. A solução desse impasse passaria pela criação de um sistema que possua algum conhecimento das regras de vez que não precise passar por todas elas de forma seqüencial para fazer a inferência.

Charles L. Forgy, na Carnegie-Mellon University, em 1979, em sua tese de Phd, desenvolveu o algoritmo de RETE. O algoritmo de Rete é um modelo muito veloz, que obtêm velocidade pelo armazenamento da informação sobre as regras, na rede. Ao invés de buscar as semelhanças dos fatos em todas as regras em todos os ciclos de reconhecimento das ações, o algoritmo Rete somente procura por trocas semelhantes em todos os ciclos. Isto aumenta em muito a velocidade da

busca das semelhanças dos fatos para os antecedentes desde que os dados estáticos que não mudam de ciclo para ciclo possam ser ignorados.

Em sua implementação clássica, apresenta duas vantagens em relação à maneira "tradicional" de casamento de padrões, na qual cada novo objeto inserido na base é testado com todas as demais combinações de objetos já presentes para se encontrar unificações com as condições das regras. A primeira deve-se ao fato de que, criando um grafo de propagação de novos fatos, as avaliações de uma condição que ocorram em mais de uma regra podem ser feitas apenas uma vez. Uma outra vantagem do algoritmo é que ele permite que as avaliações que já foram realizadas sejam armazenadas, de modo que não precisem ser refeitas sem necessidade. Por exemplo, se um novo fato "mãe(Maria, José)" for inserido na memória de trabalho do mesmo exemplo, o sistema só irá testá-lo com os fatos que já passaram pelo primeiro nó da rede ("pai(X,Y)") e cuja variável Y está unificada com o termo "Maria", evitando desta forma um grande número de testes que não levariam à ativação da regra.

CONFIGURAÇÃO / INSTALAÇÃO DO SISTEMA

INSTALAÇÃO MANUAL

Para que o aplicativo possa ser iniciado no micro a forma manual de instalação ocorre da seguinte maneira (Windows XP):

- Primeiramente é necessário instalar o software Java jre1.5.0_11 (Máquina Virtual Java);
- Em seguida é necessário fazer a configuração do micro, isso ocorre clicando em Iniciar, Configurações, Painel de Controle. Abrindo a opção Sistema e selecionando a aba Avançado, clica-se no botão Variáveis de ambiente.
- Será necessário clicar em Criar variável e colocar o nome CLASSPATH;

No campo valor deve conter a seguinte informação:

Valor:

.;%JAVA_HOME%;.;c:\\Jess70\\Jess70b7\\lib\\Jess.jar;
c:\\Jess70\\Jess70b7\\lib\\iText-1.4.8.jar

- Será necessário criar mais uma variável agora com o nome JAVA_HOME

Valor (XP em português):

C:\\Arquivo de programas\\Java\\ jre1.5.0_11;

Valor (XP em inglês):

C:\\Program Files\\Java\\ jre1.5.0_11;

- Com isso feito é preciso editar a variável Path acrescentando no final o valor:

;%JAVA_HOME%\\bin

Lembrando que a pasta do aplicativo deve ser no diretório raiz do micro (C:\). Há um ícone específico chamado EditalX que inicia o programa.

INSTALAÇÃO AUTOMÁTICA

Basta o usuário clicar no ícone Instalar.bat e o sistema será instalado automaticamente com todos os requisitos necessários.

MANUAL DO SISTEMA

Este manual tem como objetivo descrever todas as funcionalidades existentes dentro do sistema, por tratar-se de um sistema especialista sua interface é de fácil compreensão e de total interação com o usuário.

Todo o processo de elaboração do Edital se passa na tela Principal (figura I) onde existem algumas ferramentas que auxiliam durante a execução e serão descritas na seqüência.

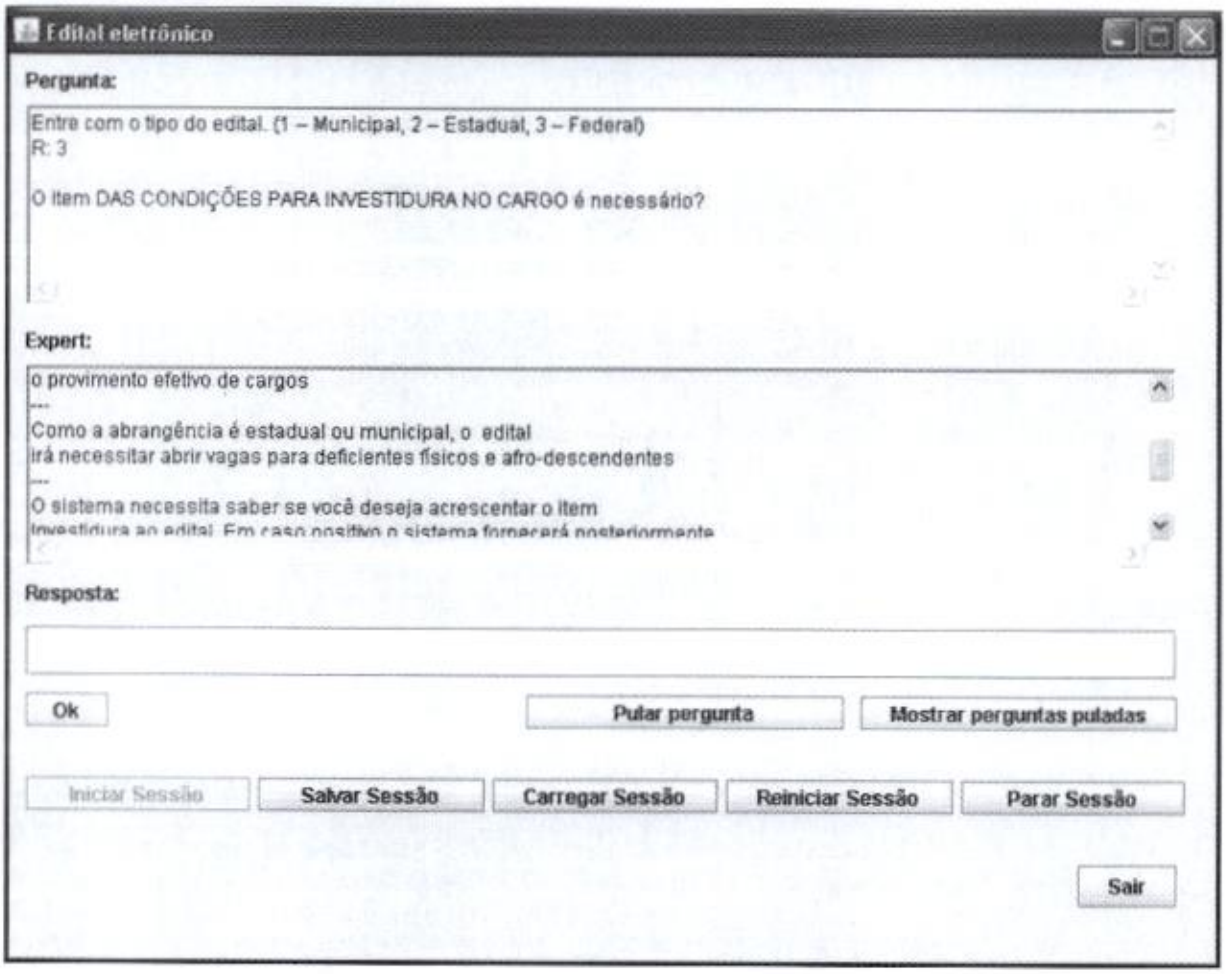


Figura I – Tela Principal

As três caixas de texto existentes na interface são fundamentais para a criação do edital e praticamente toda interação com o usuário é através delas, veja a seguir cada uma delas:

Pergunta: Toda comunicação do sistema com o usuário ocorre através desta caixa. A função dela é apresentar as perguntas (uma a uma) elaboradas pelo sistema até

que seja possível concluir o edital de maneira adequada e com todas as informações necessárias.

Resposta: A caixa resposta é o local onde o usuário deve digitar as respostas relacionadas a cada pergunta gerada pelo programa. Estas respostas devem ser digitadas de forma coesa e com a escrita adequada pois após um tratamento serão incorporadas ao edital.

Expert: Essa caixa tem como principal objetivo auxiliar o usuário em caso de dúvida com as respostas a serem inseridas. O Expert mostra sempre um comentário sobre qual informação o sistema necessita naquele momento ou algumas conclusões sobre as decisões tomadas e em certos casos cita alguns exemplos de possíveis respostas para o usuário se basear.

Dentro da tela Principal há também botões que ajudam durante o desenvolvimento do edital e podem ser utilizados sempre que necessário. Abaixo segue a definição de cada um deles:

Salvar Sessão: Salva o edital que está em desenvolvimento, ou seja, a qualquer momento durante a utilização do programa é possível salvar as etapas que já foram passadas durante a criação do edital.

Carregar Sessão: Se existir uma sessão que estava em desenvolvimento salva dentro de um diretório no computador esse botão pode ser utilizado para carregar e abrir essa sessão.

Reiniciar Sessão: A função deste botão é limpar a memória de trabalho, ou seja, apagar toda e qualquer informação inserida até o momento e reiniciar o sistema.

Parar Sessão: Ao ativar este botão o aplicativo interrompe definitivamente todo o sistema.

Pular Pergunta: Com este botão é possível deixar de responder qualquer pergunta e seguir para a próxima (lembrando que para concluir o edital todas as perguntas devem ter sido respondidas).
Listar Perguntas Indefinidas: Como o usuário pode pular perguntas este botão se encarrega de verificar e listar todas as perguntas que foram puladas.

Iniciar Sessão: Caso a sessão tenha sido parada pode-se utilizar o botão Iniciar Sessão para que o aplicativo comece novamente todo o processo.

Sair: Este botão limpa a memória de trabalho e encerra o aplicativo.

Obs: Todas as funcionalidades descritas acima estão em um arquivo Help em html juntamente com executável do programa no raiz do computador e no cd.

CÓDIGO FONTE

Parte convencional do Sistema:

Aplicativo.java

```
import javax.swing.*;

/**
 * Aplicativo é classe que executa o Sistema Especialista EditalX
 * usando a shell Jess.
 */
public class Aplicativo {

        /**
         * Método principal
         */
        public static void main(String[] args) {
                // TODO Auto-generated method stub

      TelaPrincipal forma = new TelaPrincipal();



      forma.setDefaultLookAndFeelDecorated(true);
      forma.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


            }


        }
```

Assembler.java

```
/**
 * Classe responsável por gerar parágrafos IText a partir de uma matriz
 * tridimensional de strings.
 *
 */
public class Assembler {

        //contador para os itens
        private int contItem;

        //contador para os parágrafos
```

```
private int contParagrafo;

//atributo para seleção do tipo da fonte
private int fonte;

//objeto que implementa o output do edital nos formatos pdf e rtf
private EditalRecorder editalRecorder;


//Construtor

/**
 * Construtor sem argumentos da classe Assembler.
 */
public Assembler(){

	//inicializando os int para zero
	this.contItem = 0;
	this.contParagrafo = 0;
	this.fonte = 0;

	//criando objeto editalRecorder
	this.editalRecorder = new EditalRecorder();

}

//Métodos set

/**
 * Configura o tipo da fonte que será utilizada na geração do parágrafo.
 * Valida se o argumento passado e 0 ou 1. O caso default é 0.
 * @param num o tipo da fonte a ser utilizada:
 * 0 - Times New Roman, sem negrito
 * 1 - Times New Roman, negrito
 *
 */
public void setFonte( int num ){

	// teste
	if ( ( num == 0 ) || ( num == 1 ) )
		this.fonte = num;

	//caso default
	else
		this.fonte = 0;


}

//Acessando a matriz
```

```
/**
 * Grava em documentos pdf e rtf cada string presente na matriz, em forma de
parágrafos IText.
 * @param matriz referência para a matriz que contém o conteúdo do edital.
 *
 */
public void montaEdital( String[][][] matriz ){

        //número de itens (linhas)
        int linhas = matriz.length;

        //número de parágrafos (colunas)
        int colunas = matriz[0].length;

        //número de tópicos
        int topicos = matriz[0][0].length;

        //percorre os itens (linhas da matriz)
        for (int i = 0; i < linhas; i++ ) {

                //zerando o contador de parágrafos em cada item
                this.contParagrafo = 0;

                //percorre os parágrafos (colunas da matriz)
                for (int j = 0; ( j < colunas ); j++ ) {

                        for (int k = 0; ( k < topicos ); k++ ){

                                //ignora itens vazios
                                if  ( !( matriz[i][j][k].equals( "" ) ) )

                                        //convertendo texto em parágrafo e
gravando-o em disco
                                        this.editalRecorder.gravarEdital( setTexto(
matriz[i][j][k],
                                                i, j, k ),this.fonte);


                        } // for tópicos

                } //for parágrafos

        } //for item


        //fechando o edital
        this.editalRecorder.fechaEdital();
```

```
}


/**
 * Acrescenta o número do item, do parágrafo e tabulações ao
 * String passado como argumento.
 * @param texto o texto a ser manipulado
 * @param item o número do item do texto a ser manipulado
 * @param paragrafo o número do parágrafo do texto a ser manipulado
 * @param topico o número do tópico do texto a ser manipulado
 */
public String setTexto( String texto, int item, int paragrafo, int topico ){

        //buffer de strings que será manipulada
        StringBuffer buffer = new StringBuffer();

        //se não é o item introducao (linha zero)
        if ( item > 0 ){

                //coluna zero, topico zero, nome do item
                if ( ( paragrafo == 0 ) && ( topico == 0 ) ) {

                        buffer.append( "\n" );

                        //numero do item
                        buffer.append( ++this.contItem );

                        //ponto de separaçao
                        buffer.append( ". ");

                        //texto do SE
                        buffer.append( texto );

                        //pula linha
                        buffer.append( "\n" );

                        //configurando a fonte para negrito
                        setFonte( 1 );

                } //if paragrafo


                //coluna maior do que zero e tópico igual a zero, parágrafos
                if ( ( paragrafo > 0 ) && ( topico == 0 )) {

                        //número do item
                        buffer.append( this.contItem );
```

```
                //subitem
                buffer.append( '.' );
                buffer.append( ++this.contParagrafo );
                buffer.append( "\t");

                //texto do SE
                buffer.append( texto );

                buffer.append( "\n" );

                //configurando a fonte para normal
                setFonte( 0 );

        }

        //paragrafo maior do que 0 e topico maior do 0, tópicos
        if ( ( paragrafo > 0 ) && ( topico > 0 )) {


                //número do item
                buffer.append( this.contItem );

                //subitem
                buffer.append( '.' );
                buffer.append( this.contParagrafo );
                buffer.append( '.' );
                buffer.append( topico );
                buffer.append( "\t");

                //texto do SE
                buffer.append( texto );

                buffer.append( "\n" );

                //configurando a fonte para normal
                setFonte( 0 );

        }


} //if não introducao

//se item introducao (linha 0)
else
        if ( item == 0 ) {

                //texto do SE
                buffer.append( "\t\t" );
                buffer.append( texto );
```

```
                    buffer.append( "\n" );

                    //configurando a fonte para normal
                    setFonte( 0 );

            }

            //retorna a string processada
            return ( buffer.toString() );

        }


}
```

Edital.java

```
/**
 * Classe responsável por organizar o texto de um edital de concurso público em
 * uma matriz tridimensional ( i, j, k ), sendo que os índices i correspondem aos
 * aos itens, j aos parágrafos dos itens e k aos tópicos dos parágrafos.
 *
 */
public class Edital {

        //o número máximo de itens que o edital deverá possuir
        private int numItens;

        //o número máximo de parágrafos que cada item poderá possuir
        private int numParagrafos;

        //o número de máximo de tópicos que cada parágrafo poderá possuir
        private int numTopicos;

        //matriz que contém os itens, parágrafos e tópicos do edital
        private String[][][] matriz;

        //objeto que converte a matriz em parágrafos IText
        private Assembler assembler;

        //Construtor

        /**
         * Construtor da classe Edital: número de itens, parágrafos e tópicos devem
ser
         * fornecidos.
         * @param itens o número máximo de itens que o edital pode possuir
         * @param parágrafos o número máximo de parágrafos em cada item
```

```
 * @param topicos o número máximo de tópicos em cada parágrafo
 */
public Edital( int itens, int paragrafos, int topicos ){

        //definindo o número de itens, parágrafos e tópicos do edital
        setNumItens( itens );
        setNumParagrafos( paragrafos );
        setNumTopicos ( topicos );

        //criando e inicializando a matriz
        this.matriz = new
String[getNumItens()][getNumParagrafos()][getNumTopicos()];
        for ( int i = 0; ( i < getNumItens() ); i++ )
                for ( int j = 0; ( j < getNumParagrafos() ); j++ )
                        for ( int k = 0; ( k < getNumTopicos() ); k++ )
                                this.matriz[i][j][k] = "";


        //criando o objeto assembler
        this.assembler = new Assembler();

}


//Métodos Set

/**
 * Configura o número máximo de itens que o edital pode possuir.
 * Caso o parâmetro passado seja menor ou igual a zero, o valor
 * adotado será quinze.
 * @param qde o número de itens
 *
 */
public void setNumItens( int qde ) {

        if ( qde > 0 )

                this.numItens = qde;

        else {

                System.out.println( "Número de itens menor ou igual a zero.
Utilizando " +
                                "número padrão. ( 15 itens )" );

                this.numItens = 15;

        }
```

```
}

/**
 * Configura o número máximo de parágrafos que cada item pode possuir.
 * Caso o parâmetro passado seja menor ou igual a zero, o valor
 * adotado será cinquenta.
 * @param qde o número de parágrafos
 *
 */
public void setNumParagrafos( int qde ) {

        if ( qde > 0 )

                this.numParagrafos = qde;

        else {

                System.out.println( "Número de parágrafos menor ou igual a
zero. Utilizando " +
                        "número padrão. ( 50 itens )" );

                this.numParagrafos = 50;

        }

}

/**
 * Configura o número máximo de tópicos que cada parágrafo pode possuir.
 * Caso o parâmetro passado seja menor ou igual a zero, o valor
 * adotado será cinquenta.
 * @param qde o número de tópicos
 *
 */

public void setNumTopicos ( int qde ){


        if ( qde > 0 )

                this.numTopicos = qde;

        else {

                System.out.println( "Número de tópicos menor ou igual a zero.
Utilizando " +
                        "número padrão. ( 50 tópicos )" );
```

```
                this.numTopicos = 50;

            }


        }

        //Métodos get

        /**
         * Retorna o número máximo de itens que o edital pode possuir.
         * @return um <code>int</code> indicando o número de itens.
         *
         */
        public int getNumItens() {

            return this.numItens;
        }

        /**
         * Retorna o número máximo de parágrafos que cada item pode possuir.
         * @return um <code>int</code> indicando o número de parágrafos.
         *
         */
        public int getNumParagrafos() {

            return this.numParagrafos;
        }


        /**
         * Retorna o número máximo de tópicos que cada parágrafo pode possuir.
         * @return um <code>int</code> indicando o número de tópicos.
         *
         */
        public int getNumTopicos(){

            return this.numTopicos;
        }


        /**
         * Retorna o texto do parágrafo armazenado na posição indicada pelos
argumentos
         * @return um <code>String</code> referente ao texto.
         *
         */
        public String getTextoParagrafo(int l, int c) {
```

```
        return this.matriz[l][c][0];
    }

    /**
     * Retorna o texto do tópico armazenado na posição indicada pelos
argumentos
     * @return um <code>String</code> referente ao texto.
     *
     */
    public String getTextoTopico(int i, int j, int k){

        return this.matriz[i][j][k];
    }




    //Acrescentando textos

    /**
     * Coloca texto na matriz.
     * @param item a posição do item na matriz
     * @param paragrafo posição do parágrafo na matriz
     * @param topico posição do tópico na matriz
     * @param texto a ser colocado na matriz
     */
    public void putTexto(int item, int paragrafo, int topico, String texto) {

        this.matriz[item][paragrafo][topico] += texto;


    }

    //Gravando o edital

    /**
     * Chama o método do objeto assembler para gravar o edital em disco.
     * O edital será gravado nos formatos PDF e RTF.
     *
     */
    public void salvaEdital(){

        this.assembler.montaEdital( this.matriz );


    }
```

```
}
```

EditalRecorder.java

```
//Java API
import java.awt.Color;
import java.io.FileOutputStream;
import java.io.IOException;

//IText API
import com.lowagie.text.Document;
import com.lowagie.text.DocumentException;
import com.lowagie.text.Font;
import com.lowagie.text.FontFactory;
import com.lowagie.text.Phrase;
import com.lowagie.text.pdf.PdfWriter;
import com.lowagie.text.rtf.RtfWriter2;

/**
 * Classe que implementa a gravação de documentos em formato RTF e PDF.
 */
public class EditalRecorder{



	//arquivo em formato pdf
	private Document docPdf;

	//arquivo em formato rtf
	private Document docRtf;

	//fonte em que o edital será gravado
	private Font[] fontes;


	/**
	 * Construtor sem argumentos
	 */
	public EditalRecorder(){


		//Documentos
		this.docPdf = new Document();
		this.docRtf = new Document();

		//Abrindo os arquivos
```

```
            try {
                    //Formato pdf
                    PdfWriter writerPdf = PdfWriter.getInstance(this.docPdf, new
FileOutputStream("c:\\edital\\Final.pdf"));

        writerPdf.setViewerPreferences(PdfWriter.PageModeUseOutlines);
                    this.docPdf.open();


                    //Formato rtf
                    RtfWriter2 writerRtf = RtfWriter2.getInstance(this.docRtf, new
FileOutputStream("C:\\edital\\Final.rtf"));

        writerRtf.getDocumentSettings().setAlwaysGenerateSoftLinebreaks(true);
                    this.docRtf.open();

                    //Configurando as fontes
                    this.fontes = new Font[2];
                    this.fontes[0] =
FontFactory.getFont(FontFactory.TIMES_ROMAN, 12, Font.NORMAL, new Color(0,
0, 0));
                    this.fontes[1] = FontFactory.getFont(FontFactory.TIMES_BOLD,
12, Font.BOLD, new Color(0, 0, 0));



            }

            catch(DocumentException de) {
                System.err.println(de.getMessage());
            }
            catch(IOException ioe) {
               System.err.println(ioe.getMessage());
            }


        }


        /**
         * Grava a string passada como argumento em documentos pdf e rtf
         * @param texto o texto a ser gravado
         * @param fonte o estilo da fonte a ser utilizada no parágrafo
         * 0 - Times New Roman, Tamanho 12, Normal, Cor Preta
         * 1 - Times New Roman, Tamanho 12, Negrito, Cor Preta
         */
        public void gravarEdital( String texto, int fonte ){

            try {
```

```
                //convertendo a string em parágrafo
                Phrase phrase = new Phrase( texto, this.fontes[fonte] );

                //adicionando nos documentos
                this.docPdf.add( phrase );
                this.docRtf.add( phrase );

        }

        catch (DocumentException de) {
                System.err.println(de.getMessage());
        }

    }


    /**
     * Fecha os documentos
     */
    public void fechaEdital(){

        this.docRtf.close();
        this.docPdf.close();

    }

}
```

Engine.java

```
//JESS API
import jess.*;
import java.awt.TextArea;
import jess.awt.TextAreaWriter;
import jess.awt.TextReader;

//JAVA API
import java.io.*;

/**
 * Classe que implementa o mecanismo de inferência utilizando
 * JESS (Java Expert System Shell)
 */
public class Engine {

    //jess engine, baseado no algoritmo RETE
    private Rete rete;
```

```
//adaptadores gráficos da GUI
private TextAreaWriter taw;
private TextAreaWriter taw2;
private TextReader tr;


/**
 * Construtor da classe Engine: os adaptadores para output na GUI devem ser
fornecidos
 * @param ta o stream para adaptar output das perguntas para a GUI
 * @param ta2 o stream para adapatar o output das mensagens para a GUI
 */
public Engine ( TextArea ta, TextArea ta2 ) {


        //criando o objeto rete
this.rete = new Rete();

//roteando o output para a GUI
rout( ta, ta2 );

}


/**
 * Método para inicializar o sistema especialista
 * @param arquivo a localização do JESS script
 */
public void startup( String arquivo ){

        try {

                //carregando o script que inicia o sistema
                FileReader filename = new FileReader( arquivo );

                Jesp jesp = new Jesp( filename, this.rete );

                jesp.parse(false);


        }

        catch (JessException e) {

                e.printStackTrace();
        }

        catch (IOException io) {
```

```
            io.printStackTrace();
        }

}

/**
 * Método para direcionar o output stream e input stream para a GUI
 * @param ta output para as perguntas
 * @param ta2 output para as mensagens
 */
public void rout( TextArea ta, TextArea ta2 ) {

        //criando os objetos
        this.taw = new TextAreaWriter( ta );
        this.taw2 = new TextAreaWriter( ta2 );
        this.tr = new TextReader(false);

        //direcionando input/output stream
        this.rete.addInputRouter("t", this.tr , true);
        this.rete.addOutputRouter( "t", this.taw );
        this.rete.addOutputRouter( "WSTDOUT", this.taw2 );

}

/**
 * Retorna o adaptador do input stream
 * @return um <code>TextReader</code> indicando o objeto input stream
 *
 */
public TextReader getTr(){

        return this.tr;
}


/**
 * Paralisa a máquina de inferência e fecha o Java
 */
public void sair(){

        try{

            this.rete.eval( "(exit)" );

        }

        catch (JessException e) {
```

```
                e.printStackTrace();
            }

        }

}
```

TelaPrincipal.java

```
//JAVA API
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

//JESS API
import jess.awt.TextReader;


/**
 * Classe que implementa a interface gráfica do sistema (GUI)
 */
public class TelaPrincipal extends JFrame implements ActionListener {

        /**
         * Controle da serialização
         */
        private static final long serialVersionUID = 1L;

        //componentes gráficos
    private TextArea txtPergunta;
    private JTextField txtResposta;
    private TextArea txtExpert;
    private JButton btnInicia;
    private JButton btnSalvar;
    private JButton btnReiniciar;
    private JButton btnPular;
    //private JButton btnBuscarIndef;
    private JButton btnLoad;
    private JButton btnParar;
    private JButton btnOk;
    private JButton btnSair;
    private JButton btnMostraIndef;

    //JESS engine
    private Engine engine;

    /**
         * Construtor sem argumentos da classe TelaPrincipal
```

```
     */
public TelaPrincipal() {

    //configurando a GUI
   super("Edital eletrônico");
   getContentPane().add(painel1());
   setSize(730, 570);
   setResizable(false);
   setVisible(true);

   //instância de Engine
   this.engine = new Engine( getTxtPergunta(), getTxtExpert() );

   //incializando o Sistema Especialista
   this.engine.startup( "c:\\projeto\\workspace\\editalX\\expert\\startup.clp" );


}

/**
     * Método para adicionar os componentes gráficos ao painel
     */
private JPanel painel1() {

    JPanel Jpainel1 = new JPanel();
   Jpainel1.setLayout(null);

   btnOk = new JButton("Ok");
   btnOk.addActionListener(this);
   btnOk.setBounds(10, 380, 50, 20);
   btnOk.setToolTipText( "Confirmar a resposta" );

   btnInicia = new JButton("Iniciar Sessão");
   btnInicia.addActionListener(this);
   btnInicia.setBounds(10, 430, 135, 20);
   btnInicia.setEnabled( false );
   btnInicia.setToolTipText( "Inicia uma sessão abortada");

   btnSalvar = new JButton("Salvar Sessão");
   btnSalvar.addActionListener(this);
   btnSalvar.setBounds(150, 430, 135, 20);
   btnSalvar.setToolTipText( "Salva uma sessão em andamento" );

   btnLoad = new JButton("Carregar Sessão");
   btnLoad.addActionListener(this);
   btnLoad.setBounds(290, 430, 135, 20);
   btnLoad.setToolTipText( "Carrega uma sessão que foi previamente salva" );

   btnReiniciar = new JButton("Reiniciar Sessão");
   btnReiniciar.addActionListener (this);
```

```
btnReiniciar.setBounds(430, 430, 135, 20);
btnReiniciar.setToolTipText( "Reinicia a sessão" );

btnParar = new JButton("Parar Sessão");
btnParar.addActionListener(this);
btnParar.setBounds(570, 430, 135, 20);
btnParar.setToolTipText( "Aborta a sessão" );

btnSair = new JButton("Sair");
btnSair.addActionListener(this);
btnSair.setBounds(640, 480, 60, 25);
btnSair.setToolTipText( "Fecha o aplicativo" );


JLabel lblPergunta = new JLabel("Pergunta: ");
lblPergunta.setBounds(10, 5, 80, 20);
lblPergunta.setToolTipText( "Local para a visualização da pergunta" );
txtPergunta = new TextArea(2,3);
txtPergunta.setBounds(10, 30, 150, 50);
txtPergunta.setSize(690, 120);
//txtPergunta.setFont( new Font("Arial", Font.PLAIN, 12) );
txtPergunta.setEditable(false);

JLabel lblResposta = new JLabel("Resposta: ");
lblResposta.setBounds(10, 310, 80, 20);
lblResposta.setToolTipText( "Local para entrar com a resposta" );
txtResposta = new JTextField(2);
txtResposta.setBounds(10, 340, 170, 70);
txtResposta.setSize(690, 30);
txtResposta.setEditable( true );
txtResposta.addActionListener(this);


JLabel lblExpert = new JLabel("Expert: ");
lblExpert.setBounds(10, 160, 80, 20);
lblExpert.setToolTipText( "Local para mensagens do Sistema" );
txtExpert = new TextArea(5,5);
txtExpert.setBounds(10, 185, 150, 50);
txtExpert.setSize(690, 120);
txtExpert.setEditable( false );


btnPular = new JButton("Pular pergunta");
btnPular.addActionListener (this);
btnPular.setBounds(310, 380, 190, 20);
btnPular.setToolTipText( "Pula a pergunta corrente" );

btnMostraIndef = new JButton("Mostrar perguntas puladas");
btnMostraIndef.addActionListener (this);
btnMostraIndef.setBounds(510, 380, 190, 20);
```

```
    btnMostraIndef.setToolTipText( "Mostra as perguntas que foram puladas ao
longo da sessão" );

    //btnBuscarIndef = new JButton("Responder Perguntas indefinidas");
    //btnBuscarIndef.addActionListener(this);
    //btnBuscarIndef.setBounds(540, 300, 240, 20);

    //adicionando labels e text na tela
    Jpainel1.add(lblPergunta);
    Jpainel1.add(txtPergunta);
    Jpainel1.add(lblExpert);
    Jpainel1.add(txtExpert);
    Jpainel1.add(lblResposta);
    Jpainel1.add(txtResposta);
    Jpainel1.add(btnInicia);
    Jpainel1.add(btnSalvar);
    Jpainel1.add(btnReiniciar);
    //Jpainel1.add(btnbusca);
    Jpainel1.add(btnPular);
    //Jpainel1.add(btnBuscarIndef);
    Jpainel1.add(btnLoad);
    Jpainel1.add(btnParar);
    Jpainel1.add(btnOk);
    Jpainel1.add(btnSair);
    Jpainel1.add(btnMostraIndef);

    return Jpainel1;

  }

  //Métodos get

  /**
       * Retorna o TextArea que mostra as perguntas ao usuário
       * @return um <code>TextArea</code>
       */
  public TextArea getTxtPergunta() {

      return this.txtPergunta;

  }

  /**
       * Retorna o JTextField que recebe as respostas do usuários
       * @return um <code>JTextField</code>
       */
  public JTextField getTxtResposta() {

      return this.txtResposta;
  }
```

```
/**
     * Retorna o TextArea que recebe as mensagens do sistema
     * @return um <code>TextArea</code>
     */
public TextArea getTxtExpert(){

   return this.txtExpert;
}

/**
     * Retorna a resposta corrente
     * @return um <code>String</code>
     */
public String getResposta(){

   return getTxtResposta().getText();
}

//Método set

/**
     * Configura o TextArea referente a pergunta
     */
public void setTxtPergunta( String texto ){

    getTxtPergunta().append( texto );
   getTxtPergunta().append( "\n" );


}


/**
     * Eventos dos componentes da GUI
     */
public void actionPerformed(ActionEvent e) {

    //lê a resposta
   if (e.getSource() == btnOk ) {

    this.engine.getTr().clear();
    getTxtPergunta().append( "R: " + getTxtResposta().getText() + "\n\n" );
    this.engine.getTr().appendText( getTxtResposta().getText() + "\n" );
            getTxtResposta().setText( "" );



   }
```

```
  //lê a resposta
else if (e.getSource() == txtResposta ) {

  this.engine.getTr().clear();
  getTxtPergunta().append( "R: " + getTxtResposta().getText() + "\n\n" );
  this.engine.getTr().appendText( getTxtResposta().getText() + "\n" );
  getTxtResposta().setText( "" );


}



//salvar o estado da memória de trabalho
else if (e.getSource () == btnSalvar) {

  this.engine.getTr().appendText( "-3" + "\n" );
  getTxtResposta().setText( "" );

}


//carregar memória de trabalho
else if (e.getSource () == btnLoad) {

  this.engine.getTr().appendText( "-4" + "\n" );
  getTxtResposta().setText( "" );

}

//reiniciar o sistema
else if (e.getSource () == btnReiniciar) {

  this.engine.getTr().appendText( "-5" + "\n" );
  getTxtResposta().setText( "" );
  getTxtPergunta().setText( "" );

}

//parar o sistema
else if (e.getSource () == btnParar) {


  //this.engine.getTr().appendText( "-2" + "\n" );
  getTxtResposta().setText( "" );
  getTxtPergunta().setText( "" );
  getTxtExpert().setText( "" );
```

```
  //habilita o botão para iniciar o sistema
  this.btnInicia.setEnabled( true );

  //desabilitando os outros botões
  //this.btnBuscarIndef.setEnabled( false );
  this.btnMostraIndef.setEnabled( false );
  this.btnOk.setEnabled( false );
  this.btnPular.setEnabled( false );
  this.btnReiniciar.setEnabled(false );
  this.btnSalvar.setEnabled( false );
  this.txtResposta.setEnabled( false );


}

//inicia o sistema
else if (e.getSource() == btnInicia) {

  this.engine.getTr().appendText( "-5" + "\n" );
  getTxtResposta().setText( "" );

  //this.engine.startup( "c:\\projeto\\workspace\\editalX\\expert\\startup.clp" );

  //habilitando os botões
  //this.btnBuscarIndef.setEnabled( true );
  this.btnMostraIndef.setEnabled( true );
  this.btnOk.setEnabled( true );
  this.btnPular.setEnabled( true );
  this.btnReiniciar.setEnabled( true );
  this.btnSalvar.setEnabled( true );
  this.txtResposta.setEnabled( true );

  //desabilita o botão para iniciar o sistema
  this.btnInicia.setEnabled( false );

}

//pular perguntas
else if (e.getSource () == btnPular) {

  this.engine.getTr().appendText( "-1" + "\n" );
  getTxtResposta().setText( "" );
  getTxtPergunta().append( "Responder depois...\n\n" );

}

//mostrar perguntas indefinidas
else if (e.getSource () == btnMostraIndef) {
```

```
        this.engine.getTr().appendText( "-6" + "\n" );
        getTxtResposta().setText( "" );

      }



        //fecha o sistema
      if (e.getSource () == btnSair) {

        this.engine.sair();
        //dispose();
      }

    }

}
```

Parte especialista do Sistema (Scripts):

Background

```
;Script que realiza tarefas em background

;regra que dispara as perguntas indefinidas ao final da seção
(defrule backgroundIndefinidas

   ;não tenha itens sem disparar questionario
        (not (and (explicit (MAIN::Item (id ?item))) (not (MAIN::LinkQualificado (id
questionarioDisparado)
                (item ?item)))))

   ;há perguntas indefinidas
   (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))


   =>

        ;disparando trigger
        (assert (MAIN::Tarefa buscarIndefinidas))


)
```

```
;regra que informa que há perguntas a serem definidas
(defrule backgroundHaIndefinidas

   ;alta prioridade na agenda
   (declare (salience 10000))

   ;ainda não definiu estado
   (not (MAIN::Estado haIndefinidas))

   ;há perguntas indefinidas
   (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))

   =>

   ;coloca estado Há perguntas indefinidas no SE
   (assert (MAIN::Estado haIndefinidas))


)


;regra para corrigir o estado do SE
(defrule backgroundNaoHaIndefinidas


   ;não há perguntas indefinidas
   (not (or (explicit (MAIN::Resposta (status responderDepois)))
         (explicit (MAIN::Resposta (status carregarMemoria)))
         (explicit (MAIN::Resposta (status salvarMemoria)))
         (explicit (MAIN::Resposta (status mostrarAgenda)))))

   ;estado do SE
        ?trigger <- (MAIN::Estado haIndefinidas)

   =>

   ;corrigindo estado do SE
   (retract ?trigger)

)


;regra que corrige o estado do SE para não há perguntas indefinidas
(defrule backgroundFaltaResposta
```

```
   ;baixa prioridade na agenda
   (declare (salience -100))

        (not (and (explicit (MAIN::Item (id ?item))) (not (MAIN::LinkQualificado (id
questionarioDisparado)
                (item ?item)))))

   ;há perguntas sem responder
    (and (explicit (MAIN::Pergunta (id ?id)))
         (not (explicit (MAIN::Resposta (id ?id)))))

   =>

        ;colocando a trigger para perguntar novamente
        (assert (MAIN::Perguntar ?id))

)



;regra para recuperar a memória
(defrule BackRecuperarMemoria


   ;se resposta do tipo salvarMemoria
   (MAIN::Estado fatosCarregados)


   ;há perguntas indefinidas
   (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))


   =>

        ;disparando trigger
        (assert (MAIN::Tarefa buscarIndefinidas))


)
```

CarregaFatos

```
;Script para iniciar o sistema a partir de fatos carregados externamente

;Limpa a memória de trabalho
```

```
(reset)

;Desabilita a possibilidade de alteração das variáveis globais
(set-reset-globals FALSE)

;flag que indica que fatos foram carregados externamente
(assert (MAIN::Estado fatosCarregados))

;carregando fatos externos
(load-facts "C:\\projeto\\workspace\\editalX\\Expert\\fatos\\fatos.dat")

;iniciar o mecanismo de inferência
(run)
```

Frames

```
;Script para criar os frames especificos para Edital

;Class Frame para os itens do edital
(deftemplate MAIN::Item
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot nome) ;o nome que deve constar no edital para identificar o item
   (slot negrito) ;variável que indica se o nome deverá ser em negrito
   (slot sequencia) ;variável que especifica a posição do item no edital
   (slot status)  ;define o status do item:
   ;nil - estado indicial
   ;definido - carregado na memória, questionario respondido, parágrafos
processados
   (slot id) ;define o identificador do item
)


;Instance Frame de Key para as variáveis do edital
(deftemplate MAIN::Variavel extends Key
        (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot item)      ;item ao qual pertence a variável
   (slot descricao)) ;breve descrição sobre a variável


;Instance Frame de Key para as Meta-Variáveis do edital
(deftemplate MAIN::MetaVariavel extends Key
        (declare (slot-specific TRUE) (backchain-reactive TRUE))
)

;Class Frane para a lista de palavras que compõem o texto de cada item do edital
(deftemplate Paragrafo
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (multislot texto)   ;texto do parágrafo
   (slot topico (default 0)) ;topicos do parágrafo
```

```
   (slot item) ;item ao qual o paragrafo pertence
   (slot subItem) ;numero do sub-item ao qual o paragrafo pertence
   (slot negrito)  ;fonte que o paragrafo deve utilizar:
            ; 0 - Times New Roman, 12, Normal
            ; 1 - Times New Roman, 12, Negrito
   (slot integridade) ;teste para evitar colocações inválidas pelo mec. back.
   (slot processado) ;indica se o paragrafo foi processado (nil, ok)
   (slot armazenado) ;indica se o paragrafo foi armazenado (nil, ok)

)



;Class Frame para testar se determinados fatos foram carregados com sucesso
(deftemplate FatoOk
   (declare (slot-specific TRUE))
   (slot tipo) ;o tipo do fato
)


;Instance Frame de FatoOk para testar se o item foi carregado com sucesso
(deftemplate ItemSetOk extends FatoOk
   (declare (slot-specific TRUE))
   (slot item) ;o id do item
)


;Instance Frame de FatoOk para testar se o meta-questionário foi carregado com
sucesso
(deftemplate MetaSetOk extends FatoOk
   (declare (slot-specific TRUE))
)

;Class Frame para o fato relacionado a estados que os elementos do edital podem
apresentar
(deftemplate Estado
   (declare (ordered TRUE)))

;Class Frame para os questionários do edital
(deftemplate Questionario
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot item) ;identificador do questionario
   (slot status);status do questionario:
   ;definido (todas as perguntas estão definidas)
   ;indefinido (há perguntas indefinidas)
   ;nil (não foi totalmente respondido)

)
```

```
;Class Frame para a trigger que dispara questionário
(deftemplate LoadQuestionario
   (declare (ordered TRUE)))


;Class Frame para as hipóteses
(deftemplate Hipotese
   (declare (slot-specific TRUE))
   (slot status) ;status da hipótese
)

;Instance Frame de Hipotese para gravar o edital
(deftemplate GravarEdital extends Hipotese
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
)


;Instance Frame de Hipotese para testar se todos os itens estão armazenados no
conteiner
(deftemplate ItemArmazenado extends Hipotese
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot item) ;o item do edital
)


;Instance Frame de Hipotese para testar se todos os parágrafos do item estão
processados
(deftemplate ParagrafosProcessados extends Hipotese
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot item) ;o item do edital
)


;Class Frame do tipo trigger, que possui slots
(deftemplate MAIN::LinkQualificado
   (slot id) ;id da trigger
   (slot item) ;item ao qual pertence
)
```

Funções

```
;;;Script das funções específicas para edital

;Função que transforma a primeira letra da frase minúscula
(deffunction iniciaMinuscula(?frase)


      (bind ?primeira (sub-string 1 1 ?frase)) ;pegando a primeira letra
```

```
    (bind ?segunda  (sub-string 2 (str-length ?frase) ?frase)) ;pegando o restante da
frase

    (bind ?primeira (lowcase ?primeira)) ;convertendo a primeira letra para
minúscula

    (bind ?nova (str-cat ?primeira ?segunda)) ;juntando novamente as partes

    (return ?nova)

)

;Função que transforma a primeira letra da frase maiúscula
(deffunction iniciaMaiuscula(?frase)


    (bind ?primeira (sub-string 1 1 ?frase)) ;pegando a primeira letra
    (bind ?segunda  (sub-string 2 (str-length ?frase) ?frase)) ;pegando o restante da
frase

    (bind ?primeira (upcase ?primeira)) ;convertendo a primeira letra para
minúscula

    (bind ?nova (str-cat ?primeira ?segunda)) ;juntando novamente as partes

    (return ?nova)

)


;Função para processar os parágrafos do edital
(deffunction processar(?paragrafo ?argumentos ?variavel)

  ;contador para a lista de argumentos
  (bind ?cArgumentos 0)

  ;loop ao longo da lista de argumentos para converter strings em JessLists
  (foreach ?arg ?argumentos

    (++ ?cArgumentos) ;incrementa o contador

            (if (stringp ?arg) then ;testa se o argumento é uma string

      ;substitui por JessList
                    (bind ?argumentos (replace$ ?argumentos ?cArgumentos
?cArgumentos (stringToList ?arg)))

    )

  )
```

```
   ;identifica o índice em que se encontra a variável
   (bind ?cParagrafo (member$ ?variavel ?paragrafo))


   ;caso a variável exista na lista, será substituída pelos argumentos
   (if (not (eq ?cParagrafo FALSE)) then
          (bind ?paragrafo (replace$ ?paragrafo ?cParagrafo ?cParagrafo
?argumentos))
   )

   ;retorna a lista atualizada
   (return ?paragrafo)

)


;Função que testa se existe alguma variável na lista
(deffunction paragrafoProcessado (?paragrafo)

   ;substring que identifica uma variável
   (bind ?teste "ch_")

   ;convertendo a lista para string
   (bind ?paragrafo (listToString ?paragrafo))

   ;testando se o teste existe no parágrafo
   (bind ?resultado (str-index ?teste ?paragrafo))

   ;caso exista, será um número maior ou igual a 1
   (if (numberp ?resultado) then

      (if (>= ?resultado 1) then
      (return nil)

        )
   )

   ;caso não exista será um symbol
   (if (symbolp ?resultado) then

      ;caso não exista o resultado será false
      (if (eq ?resultado FALSE) then
         (return ok)
       )
    )

   ;default
   (return nil)
```

```
)

;função que retorna cargos
(deffunction putCargo(?descricao)

   ;criando as variáveis
   (bind ?cargo "")

   (while (< (str-length ?cargo) 2)

         (printout t "Entre com o cargo." crlf)
      (printout WSTDOUT ?descricao crlf)
      (printout WSTDOUT "---" crlf)

         (bind ?cargo (readline))

   )

   (return ?cargo)


)

;função que retorna requisitos
(deffunction putRequisitos(?descricao)

   (bind ?requisitos "")

   (while (< (str-length ?requisitos) 2)

         (printout t "Entre com os requisitos, separados por vírgula." crlf)
      (printout WSTDOUT ?descricao crlf)
      (printout WSTDOUT "---" crlf)

         (bind ?requisitos (readline))

   )

   (return ?requisitos)


)

;função que retorna a carga horária
(deffunction putCargaHoraria(?descricao)

      (bind ?cargaHoraria "")
```

```
   (while (< (str-length ?cargaHoraria) 1)

          (printout t "Entre com a carga horária semanal." crlf)
      (printout WSTDOUT ?descricao crlf)
      (printout WSTDOUT "---" crlf)

          (bind ?cargaHoraria (readline))

   )


   (return ?cargaHoraria)


)

;função que retorna as vagas
(deffunction putVagas(?descricao)

   (bind ?vagas "")

   (while (< (str-length ?vagas) 1)

          (printout t "Entre com o número de vagas." crlf)
      (printout WSTDOUT ?descricao crlf)
      (printout WSTDOUT "---" crlf)

          (bind ?vagas (readline))

   )

   (return ?vagas)


)

;função que retorna as vagas para afro-descendetes
(deffunction putVagasAfro(?descricao)

   (bind ?vagasAfro "")

   (while (< (str-length ?vagasAfro) 1)

          (printout t "Entre com o número de vagas para AFRO-DESCENDENTES."
crlf)
      (printout WSTDOUT ?descricao crlf)
      (printout WSTDOUT "---" crlf)

          (bind ?vagasAfro (readline))
```

```
    )

    (return ?vagasAfro)

)

;função que retorna as vagas para deficientes físicos
(deffunction putVagasDef(?descricao)

    (bind ?vagasDef "")

    (while (< (str-length ?vagasDef) 1)

            (printout t "Entre com o número de vagas para DEFICIENTES FÍSICOS"
crlf)
        (printout WSTDOUT ?descricao crlf)
        (printout WSTDOUT "---" crlf)

            (bind ?vagasDef (readline))

    )

    (return ?vagasDef)

)

;função que retorna a remuneração
(deffunction putRemuneracao(?descricao)

    (bind ?remuneracao "")

    (while (< (str-length ?remuneracao) 1)

            (printout t "Entre com a remuneração para o cargo" crlf)
        (printout WSTDOUT ?descricao crlf)
        (printout WSTDOUT "---" crlf)

            (bind ?remuneracao (readline))

    )

    (return ?remuneracao)

)

;função que retorna a taxa de inscrição
(deffunction putTaxa(?descricao)
```

```
    (bind ?taxa "")

    (while (< (str-length ?taxa) 1)

            (printout t "Entre com a taxa de inscrição" crlf)
        (printout WSTDOUT ?descricao crlf)
        (printout WSTDOUT "---" crlf)

            (bind ?taxa (readline))

    )

    (return ?taxa)

)


;Função para controlar a colocação de cargos na memória
(deffunction putEspecCargos(?numCargos ?contador)

    (return (<= ?contador ?numCargos))

)

;Função que elimina o ponto final das respostas
(deffunction tiraPontoFinal (?string)


)


;Função para definir o tipo de resolução
(deffunction putTipoResolucao (?tipo)

    ;tipo decreto, retornando o
    (if (eq ?tipo 1) then

        (return (stringToList "o Decreto"))

     )

    ;tipo lei, retornando a
    (if (eq ?tipo 2) then

        (return (stringToList "a Lei"))

     )
```

```
   ;tipo portaria, retornando a
   (if (eq ?tipo 3) then

      (return (stringToList "a Portaria"))

    )

)

;Função para definir anexos
(deffunction putAnexos (?numero)

        (if (eq ?numero 0) then

      (return (stringToList "."))
    )


   (if (eq ?numero 1) then

      (return (stringToList "e no seu Anexo I."))
    )

   (if (> ?numero 1) then
      (return (stringToList (str-cat "e nos seus " (listToString (create$ ?numero)) "
anexos.")))
   )

)
```

Manager

```
;Script para administrar o questionário

;DEFINIÇÃO DOS TEMPLATES*****************************************************

;Class Frame para as perguntas
(deftemplate MAIN::Pergunta
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot texto) ;O texto da pergunta
   (slot tipo) ;numero, frase, fraseComArtigo, escolha, data, hora, webSite
   (slot id) ;Identificador da Pergunta
   (multislot teste)) ;Respostas permitidas


;Class Frame para as respostas
(deftemplate MAIN::Resposta
   (declare (slot-specific TRUE) (backchain-reactive TRUE))
   (slot id) ;identificador da resposta
```

```
   (slot texto) ;o texto da resposta do usuário
   (slot status) ;define o status da resposta

)


;Class Frame para o trigger que dispara perguntas
(deftemplate MAIN::Perguntar
   "Trigger que solicita a execução de uma pergunta"
       (declare (ordered TRUE)))

;Class Frame para o fato que conecta as regras
(deftemplate MAIN::Link
   (declare (ordered TRUE)))

;Class Frame para o trigger que dispara tarefas
(deftemplate MAIN::Tarefa
   "Trigger que dispara uma tarefa para o sistema"
   (declare (ordered TRUE)))

;Class Frame que identifica as perguntas e respostas na memória
(deftemplate MAIN::Key
   (declare (slot-specific TRUE))
   (slot id) ;identificador do fato
   (slot descricao) ;descricao sobre o fato

)

;Class Frame para o fato relacionado a estados que o SE pode apresentar ao longo
da interação
(deftemplate Estado
   (declare (ordered TRUE)))


;*****************************************************************************
;

;DEFINIÇÃO DAS FUNÇÕES ******************************************************


;Função que transforma um Jess Value List em uma string
(deffunction listToString(?list)

   (return (implode$ ?list))

)


;Função que transforma uma String em um Jess Value List
(deffunction stringToList(?string)
```

```
    (return (explode$ ?string))

)


;Função que retorna verdadeiro caso a frase inicie com um artigo
(deffunction isArtigo(?frase)

    ;Testa se o argumento não é um número puro
    (if (or (numberp ?frase) (floatp ?frase)
          (integerp ?frase)) then
       (return FALSE))

    ;Testa se a primeira letra do argumento é um artigo (and lógico)
    (bind ?teste (* (str-compare (sub-string 1 1 ?frase) "o")
            (str-compare (sub-string 1 1 ?frase) "a")
            (str-compare (sub-string 1 1 ?frase) "O")
            (str-compare (sub-string 1 1 ?frase) "A")))

    (return (= ?teste 0))

)

;função para validar datas
(deffunction validaData (?data)

    ;convertendo para string
         (bind ?data (str-cat ?data))

    ;string vazia - início do loop
    (if (eq (str-length ?data) 0) then
       (return FALSE))

    ;testando o comprimento da string
    (if (<> (str-length ?data) 10) then
       (printout WSTDOUT "Utilize o formato dd-mm-aaaa" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )

    ;extraindo o dia
    (bind ?dia (integer (sub-string 1 2 ?data)))

    ;extraindo o mes
    (bind ?mes (integer (sub-string 4 5 ?data)))

    ;extraindo o ano
    (bind ?ano (integer (sub-string 7 10 ?data)))

    ;confirmando ano maior do que 1800
```

```
    (if (< ?ano 1800) then
       (printout WSTDOUT "Ano inválido" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )


    ;confirmando mês
    (if (or (< ?mes 1) (> ?mes 12)) then
       (printout WSTDOUT "Mês inválido" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )


    ;confirmando dia maior do que zero
    (if (or (< ?dia 1) (> ?dia 31)) then
       (printout WSTDOUT "Dia inválido" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )

    ;criando lista de dias / mês
    (bind ?vetor (create$ 31 28 31 30 31 30 31 31 30 31 30 31))

    ;testando anos bissextos
    (if (and (eq ?dia 29) (eq ?mes 2) (or (eq (mod ?ano 400) 0)
          (and (eq (mod ?ano 4) 0) (<> (mod ?ano 100) 0)))) then
               (return TRUE))

    ;testando o limite do dia / mês
    (if (> ?dia (nth$ ?mes ?vetor)) then
       (printout WSTDOUT "Dia inválido" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )

    ;não entrou em nenhum dos filtros
    (return TRUE)


)

;função para validar horas
(deffunction validaHora (?time)

    ;convertendo para string
          (bind ?time (str-cat ?time))

    ;string vazia - início do loop
```

```
    (if (eq (str-length ?time) 0) then
       (return FALSE))

    ;testando o comprimento da string
    (if (<> (str-length ?time) 5) then
       (printout WSTDOUT "Utilize o formato hh:mm" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )

    ;extraindo a hora
    (bind ?hora (integer (sub-string 1 2 ?time)))

    ;extraindo o minuto
    (bind ?minuto (integer (sub-string 4 5 ?time)))

    ;extraindo o separador
    (bind ?separador (sym-cat (sub-string 3 3 ?time)))

    ;testando o separador
    (if (not (eq ?separador :)) then
       (printout WSTDOUT "Separador inválido. Utilize :" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )


    ;confirmando hora
    (if (or (< ?hora 1) (> ?hora 24)) then
       (printout WSTDOUT "Hora inválida" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )


    ;confirmando minuto
    (if (or (< ?minuto 0) (> ?minuto 59)) then
       (printout WSTDOUT "Minutos inválidos" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )

    ;não entrou em nenhum dos filtros
    (return TRUE)

)


;função para validar entrada de sites
(deffunction validaSite (?site)
```

```
    ;criando o contador de campos
    (bind ?count 0)

    ;criando a string
        (bind ?site (str-cat ?site))

    ;string vazia - início do loop
    (if (eq (str-length ?site) 0) then
       (printout t "Vazio" crlf)
       (return FALSE))

    ;extraindo www
    (bind ?www (sym-cat (sub-string 1 3 ?site)))

    ;testando o www
    (if (not (eq ?www www)) then
       (printout WSTDOUT "---" crlf)
       (printout WSTDOUT "Inicie com www!" crlf)
       (printout WSTDOUT "---" crlf)
       (return FALSE)
    )


    ;testando campos
    (bind ?resultado (str-index "." ?site))

    ;caso não exista será um symbol
    (if (symbolp ?resultado) then

       ;caso não exista o resultado será false
       (if (eq ?resultado FALSE) then
              (printout WSTDOUT "---" crlf)
              (printout WSTDOUT "Utilize ponto para separar os campos!" crlf)
              (printout WSTDOUT "---" crlf)
              (return FALSE)
        )
     )

    ;segundo campo
    (bind ?segundo (sub-string (+ ?resultado 1) (str-length ?site) ?site))

    ;não passou pelos filtros
    (return TRUE)

)


;Função para testar se o argumento é um número
```

```
(deffunction testaNumero(?numero)
        "Retorna TRUE se ?numero é um número"
        (try
                (integer ?numero)
                (return TRUE)
        catch
                (return FALSE)))


;Função para validar a respota do usuário
(deffunction validaResposta (?resposta ?tipo ?teste)
        "Valida as respostas do usuário"

   ;;tipo escolha
        (if (eq ?tipo escolha) then
                (foreach ?item ?teste
                        (if (eq (sym-cat ?resposta) (sym-cat ?item)) then ;testa se está
na lista
                                (return TRUE)))
                (return FALSE))

   ;;tipo numero
        (if (eq ?tipo numero) then
                (return (testaNumero ?resposta))) ;testa se é número

   ;;tipo frase
   (if (eq ?tipo frase) then
      (return (or
            ;testa respostas triggers
              (or (eq ?resposta "-1") (eq ?resposta "-2") (eq ?resposta "-3") (eq
?resposta "-4")
              (eq ?resposta "-5") (eq ?resposta "-6"))
            ;testa respostas válidas
            (and (> (str-length ?resposta) 0);testa se usuário digitou alguma frase
            (stringp ?resposta)))) ;testa se é string
   )

   ;;tipo fraseComArtigo
   (if (eq ?tipo fraseComArtigo) then

      (return (or
            ;testa respostas triggers
              (or (eq ?resposta "-1") (eq ?resposta "-2") (eq ?resposta "-3") (eq
?resposta "-4")
              (eq ?resposta "-5") (eq ?resposta "-6"))
            ;testa respostas válidas
            (and (> (str-length ?resposta) 0);testa se usuário digitou alguma frase
            (isArtigo ?resposta) ;testa se é artigo
            (stringp ?resposta)))) ;testa se é string
   )
```

```
    ;;tipo fraseComPresposicao
  (if (eq ?tipo fraseComPreposicao) then

    (return  (stringp ?resposta))

    )


  ;;tipo data
  (if (eq ?tipo data) then

    (return (or (validaData ?resposta)
            ;testa respostas triggers
            (eq ?resposta -1) (eq ?resposta -2) (eq ?resposta -3) (eq ?resposta -4)
             (eq ?resposta -5) (eq ?resposta -6)))


   )


  ;;tipo hora
  (if (eq ?tipo hora) then

    (return (or (validaHora ?resposta)
            ;testa respostas triggers
            (eq ?resposta -1) (eq ?resposta -2) (eq ?resposta -3) (eq ?resposta -4)
             (eq ?resposta -5) (eq ?resposta -6)))
   )


  ;;tipo webSite
  (if (eq ?tipo webSite) then

    (return (or
            ;testa respostas triggers
            (or (eq ?resposta -1) (eq ?resposta -2) (eq ?resposta -3) (eq ?resposta -
4)
             (eq ?resposta -5) (eq ?resposta -6))

            (symbolp ?resposta)))


    )

)
```

```
;Função para efetuar perguntas ao usuário
(deffunction leResposta(?pergunta ?tipo ?teste)
        "Faz uma pergunta e retorna uma resposta"
        (bind ?resposta "")
        (while (not (validaResposta ?resposta ?tipo ?teste)) do

                (printout t ?pergunta)

      (printout t crlf)

      (if (or (eq ?tipo frase) (eq ?tipo fraseComArtigo) (eq ?tipo fraseComPreposicao))
         then ;se for do tipo frase usa readline para ler como string
         (bind ?resposta (readline))
         ;explode$ implode$!!!
```

```
   ;reiniciar memória
   (if (or (eq ?resposta -5) (eq ?resposta "-5")) then
      (return reiniciarMemoria)
   )

   ;mostrar agenda
   (if (or (eq ?resposta -6) (eq ?resposta "-6")) then
      (return mostrarAgenda)
   )

   ;responder depois
   (if (or (eq ?resposta -1) (eq ?resposta "-1")) then
      (return responderDepois)
   )

   ;caso chegou até aqui, resposta definida
   (return respostaDefinida)

)


;*****************************************************************************
;

;DEFINIÇÃO DAS REGRAS QUE COORDENAM O
QUESTIONÁRIO*******************************


;Regra goal-seeking para disparar a trigger necessária para a interação com o
usuário
(defrule putTriggerPerguntar
   ?p <- (MAIN::need-Resposta (id ?id)) ;goal-seeking
   (MAIN::Key (id ?id)) ;existe a variável id
   (not (explicit (MAIN::Resposta (id ?id)))) ;pergunta ainda não foi respondida
   (not (MAIN::Perguntar ?)) ;não há solicitação de disparar perguntas
   =>
   (assert (MAIN::Perguntar ?id)) ;colocar a trigger

   ;eliminar fato desnecessário
   (retract ?p)

)


;Regra que dispara a interação com o usuário
(defrule disparaPerguntaId
   ?pergunta <- (explicit (MAIN::Pergunta (id ?id) (texto ?texto) (teste $?teste) (tipo
?tipo))) ;a pergunta solicitada está na memória de trabalho
   (MAIN::Key (id ?id) (descricao ?descricao)) ;existe a variável id
   (not (explicit (MAIN::Resposta (id ?id)))) ;a pergunta ainda não foi respondida
```

```
   ?trigger <- (MAIN::Perguntar ?id) ;trigger para disparar a pergunta ?id
   =>
   (retract ?trigger)
   (printout WSTDOUT ?descricao crlf)
   (printout WSTDOUT "---" crlf)
   (bind ?resposta (leResposta ?texto ?tipo ?teste))
   ;coloca o status da resposta
   (bind ?status (statusResposta ?resposta))
   (assert (Resposta (id ?id) (texto ?resposta) (status ?status))))



;Regra goal-seeking para a inserção de perguntas
(defrule putPergunta
   ?p <- (MAIN::need-Pergunta (id ?id) (texto ?texto) (teste $?teste) (tipo ?tipo))
;goal-seeking
   (MAIN::Key (id ?id)) ;existe a variável id
   (not (explicit (MAIN::Pergunta (id ?id)))) ;não existe pergunta com este id
   =>
   (assert (MAIN::Pergunta (id ?id) (texto ?texto) (teste $?teste) (tipo ?tipo))) ;coloca
a pergunta

   ;elimina fato desnecessário
   (retract ?p)
)


;Regra para perguntar novamente uma pergunta "pulada"
(defrule askAgain

   (declare (salience 10000))
   ?trigger <- (MAIN::Tarefa buscarIndefinidas) ;trigger para entrar
        ;resposta indefinida
   ?resposta <- (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))
   =>
        (assert (MAIN::Perguntar ?id)) ;colocando a trigger para perguntar novamente
   (retract ?resposta)  ;eliminando a resposta
   ;(retract ?trigger)
)



;regra para disparar perguntas que foram puladas
(defrule dispararIndefinidas2

   ;baixa prioridade na agenda
   (declare (salience -1000))
```

```
   ;respostas indefinidas
   (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))
   =>

   ;dispara trigger
        (assert (MAIN::Tarefa buscarIndefinidas))
)


;regra para mostrar as respostas indefinidas
(defrule dispararTipoAgenda

   ;alta prioridade na agenda
   (declare (salience 10000))

   ;mostrarAgenda
   (explicit (MAIN::Resposta (status mostrarAgenda)))

   ;respostas indefinidas
   (or (and (explicit (MAIN::Pergunta (id ?id) (texto ?t)))
      (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))))

   =>
   (printout WSTDOUT "Pergunta que foi pulada:" crlf)
   (printout WSTDOUT ?t crlf crlf)



)

;Regra para reiniciar o sistema
(defrule resetar
   ;altíssima prioridade
   (declare (salience 10000))
        ; solicitação do usuário
        (explicit (MAIN::Resposta (id ?id) (status reiniciarMemoria)))
   =>
   (printout WSTDOUT "Reiniciando o sistema..." crlf crlf)
   ;limpando a memória
   (clear)
   (reset)
   ;reinciando o sistema
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Startup.clp")
)
```

```
;Regra para parar o sistema
(defrule parar
   ;altíssima prioridade
   (declare (salience 10000))
        ; solicitação do usuário
        (explicit (MAIN::Resposta (id ?id) (status pararSistema)))
   =>
        (printout WSTDOUT "Parando o sistema..." crlf crlf)
   (halt)

)


;regra para salvar estado da memória
(defrule salvarMemoria

   ;altíssima prioridade
   (declare (salience 10000))

   ;se trigger
   (MAIN::Link podeGravarMemoria)

   ;se não há fato que impeça
   (not (MAIN::Link naoPodeGravarMemoria))

   ;se resposta do tipo salvarMemoria
   ?resposta <- (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))

   =>

   ;(printout WSTDOUT "Salvando a sessão..." crlf crlf)

   ;salvado a memoria
   (save-facts "C:\\projeto\\workspace\\editalX\\Expert\\fatos\\fatos.dat")

   ;repetindo a pergunta
        (assert (MAIN::Perguntar ?id)) ;colocando a trigger para perguntar novamente
   (retract ?resposta) ;eliminando a resposta


)

;regra para recuperar a memória
(defrule recuperarMemoria

   (declare (salience 10000))

   ;se resposta do tipo salvarMemoria
   ?resposta <- (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
```

```
=>

(retract ?resposta)

(printout WSTDOUT "Carregando sessão..." crlf crlf)

;reinciando o sistema
(batch "C:\\projeto\\workspace\\editalX\\Expert\\CarregaFatos.clp")

)
```

SessionRules

```
;Script com regras específicas para a sessão de consulta para a geração do edital

;Coloca na memória a hipótese de que o sistema está pronto para gravar o edital
(defrule hipoteseGravarEdital

        (MAIN::GravarEdital (status ok))

=>

        ;chama o método da classe para gravar o edital
        (?*edital* salvaEdital)

(printout WSTDOUT "Edital gravado com sucesso!!" crlf)
(printout WSTDOUT "---" crlf)

(exit)


)


;Regra goal-seeking para a inserção de variaveis
(defrule putVariavel

 ;goal-seeking
?p <- (MAIN::need-Variavel (id ?id) (item ?item) (descricao ?descricao))

;não existe a variável
(not (explicit (MAIN::Variavel (id ?id))))

;existe o item
```

```
    (explicit (MAIN::Item (id ?item)))

    =>

    ;coloca a variável na memória
    (assert (MAIN::Variavel (id ?id) (item ?item) (descricao ?descricao)))

    ;eliminando fato desnecessário
    (retract ?p)
)


;Regra goal-seeking para a inserção de meta-variaveis
(defrule putMetaVariavel

    ;goal-seeking
    ?p <- (MAIN::need-MetaVariavel (id ?id) (descricao ?descricao))

    ;não existe a meta-variavel
    (not (MAIN::MetaVariavel (id ?id)))

    =>

    ;coloca a meta-variavel
    (assert (MAIN::MetaVariavel (id ?id) (descricao ?descricao)))

    ;eliminando fato desnecessário
    (retract ?p)

)



;Regra goal-seeking para a inserção de itens
(defrule putItem

    ;goal-seeking
    ?p <- (MAIN::need-Item (id ?id) (sequencia ?seq) (negrito ?negrito) (status nil)
       (nome ?nome))

    ;não existe item com este id
    (not (explicit (MAIN::Item (id ?id))))

    ;não existe item esta sequência
    (not (explicit (MAIN::Item (sequencia ?seq))))

    ;não existe item com este nome
    (not (explicit (MAIN::Item (nome ?nome))))

    =>
```

```
   ;coloca o item na memória
       (assert (MAIN::Item (id ?id) (sequencia ?seq) (negrito ?negrito) (status nil)
     (nome ?nome)))

       ;eliminando fato desnecessário
   (retract ?p)


)


;Regra goal-seeking para a inserção de parágrafos
(defrule putTriggerParagrafo


   ;goal-seeking
   ?p <- (MAIN::need-Paragrafo (texto $?texto) (item ?item) (subItem ?subItem)
     (negrito ?negrito) (integridade ok) (processado ?processado)
     (armazenado nil) (topico ?topico))

   ;existe o item
   (explicit (MAIN::Item (id ?item)))

   ;Não existe no item um parágrafo com o mesmo número de sub-item
   (not (explicit (MAIN::Paragrafo (item ?item) (subItem ?subItem) (topico ?topico))))

   =>

   ;coloca o paragrafo na memória
       (assert (MAIN::Paragrafo (texto $?texto) (item ?item) (subItem ?subItem)
     (negrito ?negrito) (integridade ok) (processado ?processado)
        (armazenado nil) (topico ?topico)))

   ;eliminando desnecessário
   (retract ?p)

)


;Regra para dar início ao sistema
(defrule mostraTela

   ;se trigger
   ?trigger <- (MAIN::Tarefa iniciarSistema)

   =>

   ;Carregando as meta-regras
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\MetaRules.clp")
```

```
    ;eliminando a trigger
    (retract ?trigger)

)

;Regra que faz verificações para disparar o questionário ?id
(defrule disparaQuestionario

    ;goal seeking
    (MAIN::need-Questionario (item ?item) (status carregado))

    ;se os itens foram carregados e definidos
    (MAIN::Estado itensDefinidosECarregados)

    ;se o item foi carregado
    (MAIN::ItemSetOk (item ?item) (tipo item))

        ;se as variáveis foram carregadas
    (MAIN::ItemSetOk (item ?item) (tipo variavel))

        ;se as perguntas foram carregadas
    (MAIN::ItemSetOk (item ?item) (tipo pergunta))

        ;se o layout foi carregado
    (MAIN::ItemSetOk (item ?item) (tipo layout))

    ;se o item existe
    (explicit (MAIN::Item (id ?item)))

    =>

        ;dispara trigger para carregar o quesionário
        (assert (MAIN::LoadQuestionario ?item))

)

;Regra goal para disparar o questionario ?item
(defrule goalQuestionario

        ;se trigger
    ?p <- (MAIN::LoadQuestionario ?item)

    ;se o questionário ainda não foi carregado
    (not (MAIN::Questionario (item ?item)))

    =>
```

```
   ;coloca o fato na memória
   (assert (MAIN::Questionario (item ?item) (status carregado)))

)


;Regra para definir status do questionário
(defrule statusQuestionarioDefinido

   ;estado do SE
       (not (MAIN::Estado haIndefinidas))

   ;disparou o questionário do item
   (MAIN::LinkQualificado (id questionarioDisparado) (item ?item))

   ;o questinário foi carregado na memória
   ?p <- (explicit (MAIN::Questionario (status carregado) (item ?item)))

   (and
      (explicit (MAIN::Resposta))
   ;não há resposta do tipo parar o sistema
   (not (explicit (MAIN::Resposta (status pararSistema))))
   ;não há perguntas pendentes
   (not (explicit (MAIN::Resposta (status responderDepois))))
   ;não resposta do tipo salvar a memória
   (not (explicit (MAIN::Resposta (status salvarMemoria))))
   ;não resposta do tipo carregar memória
   (not (explicit (MAIN::Resposta (status carregarMemoria))))
   ;não resposta do tipo reiniciar memória
   (not (explicit (MAIN::Resposta (status reiniciarMemoria))))
   ;não resposta do tipo mostrar agenda
   (not (explicit (MAIN::Resposta (status mostrarAgenda))))
   ;não resposta indefinida
   (not (explicit (MAIN::Resposta (status nil)))))

   ;todas as perguntas foram respondidas
   (not (and (explicit (MAIN::Pergunta (id ?id)))
         (not (explicit (MAIN::Resposta (id ?id))))))

   ;verifica se existe o item na memória
   (explicit (MAIN::Item (id ?item)))

   =>

   ;modifica o status para definido
   (modify ?p (status definido))

)
```

```
;Regra que prova a hipótese: todos os parágrafos do item estão processados
(defrule provaHipoteseParagrafosProcessados

   ;goal-seeking
        ?trigger <- (MAIN::need-ParagrafosProcessados (item ?item) (status ok))

   (MAIN::Estado itensDefinidosECarregados)

   ;estado do SE
        (not (MAIN::Estado haIndefinidas))

        (not (and (explicit (MAIN::Item (id ?id))) (not (MAIN::LinkQualificado (id
questionarioDisparado)
                (item ?id)))))

   ;não podem ter perguntas indefinidas
   (not (or (explicit (MAIN::Resposta (status pararSistema)))
        (explicit (MAIN::Resposta (status responderDepois)))
        (explicit (MAIN::Resposta (status salvarMemoria)))
        (explicit (MAIN::Resposta (status carregarMemoria)))
        (explicit (MAIN::Resposta (status reiniciarMemoria)))
        (explicit (MAIN::Resposta (status mostrarAgenda)))
        (explicit (MAIN::Resposta (status nil)))))

   ;todas as perguntas foram respondidas
   (not (and (explicit (MAIN::Pergunta (id ?id)))
         (not (explicit (MAIN::Resposta (id ?id))))))


   ;questionário definido
   (explicit (MAIN::Questionario (item ?item) (status definido)))


   ;não há parágrafos sem processar
   (and (not (explicit (MAIN::Paragrafo (item ?item) (processado nil)
                (integridade ok))))
      (explicit (MAIN::Paragrafo (item ?item))))

   ;existe o item
   ?pItem <- (explicit (MAIN::Item (id ?item) (status nil) (nome ?nome) (sequencia
?sequencia)))


   =>

   ;a medida que os parágrafos começam a ser trasnferidos para a matriz, a
memória não poderá ser salva
   (assert (MAIN::Link naoPodeGravarMemoria))
```

```
    ;comprova a hipótese
    (assert (MAIN::ParagrafosProcessados (item ?item) (status ok)))

    ;muda o status do item
    (modify ?pItem (status definido))

    ;grava o nome do item na matriz
    (?*edital* putTexto ?sequencia 0  0 ?nome)

    ;eliminando fato desnecessário
    (retract ?trigger)


)


;Regra para armazenar o conteúdo de itens processados
(defrule armazenarItens

    ;todos os itens estão definidos
    (not (explicit (MAIN::Item (status nil))))

    ;todos os parágrafos estão processados
    (explicit (MAIN::ParagrafosProcessados (item ?item) (status ok)))

    ;busca os parágrafos do item
    ?p <- (explicit (MAIN::Paragrafo (item ?item) (subItem ?subItem) (texto $?texto)
       (topico ?topico) (negrito ?negrito) (integridade ok) (armazenado nil)
       (processado ok)))

    ;não podem ter perguntas indefinidas
    (not (or (explicit (MAIN::Resposta (status pararSistema)))
         (explicit (MAIN::Resposta (status responderDepois)))
         (explicit (MAIN::Resposta (status salvarMemoria)))
         (explicit (MAIN::Resposta (status carregarMemoria)))
         (explicit (MAIN::Resposta (status reiniciarMemoria)))
         (explicit (MAIN::Resposta (status mostrarAgenda)))
         (explicit (MAIN::Resposta (status nil)))))

    ;todas as perguntas foram respondidas
    (not (and (explicit (MAIN::Pergunta (id ?id)))
          (not (explicit (MAIN::Resposta (id ?id))))))

    ;estado do SE
         (not (MAIN::Estado haIndefinidas))

    ;questionário definido
    (explicit (MAIN::Questionario (item ?item) (status definido)))

    ;existe o item
```

```
   (explicit (MAIN::Item (id ?item) (nome ?nome) (sequencia ?sequencia)))

   =>


   ;armazenando o conteúdo do item no conteiner
   (bind ?paragrafo (listToString ?texto))

   (?*edital* putTexto (integer ?sequencia) (integer ?subItem) (integer ?topico)
?paragrafo)


   (retract ?p)


)


;Regra que prova a hipótese: o item está armazenado
(defrule provaHipoteseItemArmazenado

   ;goal seeking
   (MAIN::need-ItemArmazenado (item ?item) (status ok))

   ;todos os parágrafos do item foram processados
       (explicit (MAIN::ParagrafosProcessados (item ?item) (status ok)))

   ;todos os parágrafos foram transferidos para a matriz
   (not (explicit (MAIN::Paragrafo)))

   ;existe o item
   (explicit (MAIN::Item (id ?item)))


   =>

   ;confirma a hipótese
   (assert (MAIN::ItemArmazenado (item ?item) (status ok)))

)


;Regra que prova a hipótese: Gravar o edital
(defrule provaHipoteseGravarEdital

   ;goal-seeking
   (MAIN::need-GravarEdital (status ok))

   ;não há itens com hipóteses a serem confirmadas
```

```
        (not (and (explicit (MAIN::Item (id ?id))) (not (MAIN::LinkQualificado (id
hipotesesOk)
                (item ?id)))))

   ;todos os itens necessários estão definidos e carregados
   (MAIN::Estado itensDefinidosECarregados)

   ;não há mais parágrafos sem processar
   (not (explicit (MAIN::Paragrafo)))
   (and (not (explicit  (MAIN::ParagrafosProcessados (status nil))))
      (explicit (MAIN::ParagrafosProcessados)))

   ;não há itens sem armazenar
   (and (explicit (MAIN::ItemArmazenado))
      (not (explicit (MAIN::ItemArmazenado (status nil)))))

   ;estado do SE
        (not (MAIN::Estado haIndefinidas))


   ;todas as perguntas foram respondidas
   (not (and (explicit (MAIN::Pergunta (id ?id)))
         (not (explicit (MAIN::Resposta (id ?id))))))

   ;não há questionários indefinidos
   (and (not (explicit (MAIN::Questionario (status indefinido))))
      (not (explicit (MAIN::Questionario (status nil))))
               (not (explicit (MAIN::Questionario (status carregado))))
      (explicit (MAIN::Questionario)))

   ;não há itens indefinidos
        (not (explicit (MAIN::Item (id ?id) (status nil))))

   =>

   ;hipótese GravarEdital comprovada
   (assert (MAIN::GravarEdital (status ok)))

)
```

Startup

```
;Script para iniciar o sistema

;Limpa a memória de trabalho
(reset)

;Desabilita a possibilidade de alteração das variáveis globais
(set-reset-globals FALSE)
```

```
;carregando o gerenciador do questionário
(batch "C:\\projeto\\workspace\\editalX\\Expert\\Manager.clp")

;carregando os frames específicos do edital
(batch "C:\\projeto\\workspace\\editalX\\Expert\\Frames.clp")

;carregando as funções específicas para a manipulação de editais
(batch "C:\\projeto\\workspace\\editalX\\Expert\\Funcoes.clp")

;carregando as regras da sessão definição do edital
(batch "C:\\projeto\\workspace\\editalX\\Expert\\SessionRules.clp")

;carregando as tarefas background
(batch "C:\\projeto\\workspace\\editalX\\Expert\\Background.clp")

;flag que indica que o sistema está pronto para gravar a memória
(assert (MAIN::Link podeGravarMemoria))

;trigger para desencadear o início do sistema
(assert (MAIN::Tarefa iniciarSistema))

;iniciar o mecanismo de inferência
(run)
```

BASE DE CONHECIMENTO

Afro

```
;Script que coloca fatos necessários para a montagem do questionário do item afro

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisAfro

   ;testa se colocou o item
   (MAIN::ItemSetOk (item afro) (tipo item))

        (MAIN::Variavel (id ch_PrazoAfro) (item afro)
        (descricao "O sistema necessita saber quantos dias úteis o candidato tem
para se
apresentar para a perícia para a verificação dos traços fenotípicos que o
caracterizam na sociedade como pertencente ao grupo racial negro e à entrevista
que
decidirá se o candidato atende aos requisitos. Exemplo de resposta: 5"))

        (MAIN::Variavel (id ch_DesignadorBancaAfro) (item afro)
```

```
        (descricao "O sistema necessita quem será o responsável por designar a
banca que
fará a entrevista com o candidato para decidir se o mesmo atende aos requisitos
necessários para concorrer as vagas reservadas para afro-descendentes. Você deve
iniciar a resposta
com o artigo adequado. Exemplo de resposta: A Comissão Executiva do Concurso
Público"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item afro) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasAfro

   ;testa se colocou as variaveis
   (MAIN::ItemSetOk (item afro) (tipo variavel))

        (MAIN::Pergunta (id ch_PrazoAfro) (tipo numero)
        (texto "Qual é o prazo para que o candidato convocado apresente-se
para a verificação dos traços fenotípicos?"))

        (MAIN::Pergunta (id ch_DesignadorBancaAfro) (tipo fraseComArtigo)
        (texto "Quem designará a banca para a entrevista com o candidato?"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item afro) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutAfro

  ;testa se colocou as perguntas
  (MAIN::ItemSetOk (item afro) (tipo pergunta))

        (MAIN::Paragrafo (texto São reservadas aos afro-descendentes as vagas
qualificadas no item 4.1 deste edital,
```

conforme previsto na Lei 14.274, de 26/12/2003.) (item afro) (integridade ok) (negrito 0) (subItem 1)
(processado ok) (topico 0))

(MAIN::Paragrafo (texto Considerar-se-á afro-descendente aquele que assim se declarar expressamente,
identificando-se como de cor preta ou parda, da raça etnia negra, devendo tal informação
integrar os registros cadastrais de ingresso dos servidores.) (item afro) (integridade ok) (negrito 0) (subItem 2)
(processado ok) (topico 0))

(MAIN::Paragrafo (texto O candidato afro-descendente que se inscrever para cargo cuja vaga está reservada,
conforme item 4.1, deverá declarar essa condição no ato da inscrição.) (item afro) (integridade ok) (negrito 0) (subItem 3)
(processado ok) (topico 0))

(MAIN::Paragrafo (texto Os candidatos às vagas reservadas aos afro-descendentes participarão do concurso em
igualdade de condições com os demais candidatos no que concerne às exigências estabelecidas neste edital.) (item afro) (integridade ok) (negrito 0) (subItem 4)
(processado ok)
(topico 0))

(MAIN::Paragrafo (texto Detectada a falsidade na declaração a que se refere o item anterior, sujeitar-se-á o infrator
às penas da Lei e ainda:)
(item afro) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Se candidato, à eliminação do Concurso Público e de todos os atos dai decorrentes,)
(item afro) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 1))

(MAIN::Paragrafo (texto Se já admitido no cargo efetivo para o qual concorreu na reserva de vagas, utilizando-se
da declaração inverídica, à pena disciplinar de demissão motivada.)
(item afro) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 2))

(MAIN::Paragrafo (texto No prazo de ch_PrazoAfro dias úteis, contados da divulgação dos resultados da aprovação, o
candidato afro-descendente será convocado, de acordo com o número de vagas, a classificação e o interesse do apoiador,
para submeter-se à perícia para a verificação dos traços fenotípicos que o caracterizam na
sociedade como pertencente ao grupo racial negro e à entrevista com banca a ser designada ch_DesignadorBancaAfro , que decidirá se o candidato atende aos requisitos.) (item afro) (integridade ok) (negrito 0) (subItem 6) (topico 0))

```
        (MAIN::Paragrafo (texto O concurso público será homologado depois da
realização da perícia e da entrevista de
que trata o item anterior, sem prejuízo da homologação prévia do resultado em
relação
aos candidatos da lista geral, que não concorrem às vagas reservadas.) (item afro)
(integridade ok) (negrito 0) (subItem 7)
      (processado ok) (topico 0))

        (MAIN::Paragrafo (texto As vagas reservadas serão liberadas aos demais
candidatos, caso não tenha ocorrido
inscrição para o concurso ou aprovação de candidato afro-descendente.) (item afro)
(integridade ok) (negrito 0) (subItem 8)
(processado ok) (topico 0))

        (MAIN::Paragrafo (texto Os candidatos que se inscreverem para a reserva de
vagas para afro-descendentes poderão
concorrer, além das vagas que lhes são destinadas por Lei, à totalidade das vagas,
desde
que aprovados, habilitados e observada rigorosamente a ordem geral de
classificação.) (item afro) (integridade ok) (negrito 0)
(subItem 9) (processado ok) (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item afro) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesAfro

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item afro))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item afro) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item afro) (status ok))


   =>

        (assert (MAIN::LinkQualificado (id hipotesesOk) (item afro)))
```

```
)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Iniciar Questionário
(defrule iniciaAfro
        ;testa hipótese
        (MAIN::Questionario (item afro) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_PrazoAfro) (texto ?resposta)) ;hipótese backward-
chaining

   =>

   ;trigger para prosseguir
        (assert (MAIN::Link designadorBancaAfro))

)

;Regra que verifica quem será o responsável por designar a banca que
;fará a entrevista com o candidato para decidir se o mesmo atende aos requisitos
;necessários para concorrer as vagas reservadas para afro-descendentes.
(defrule designadorBancaAfro
        ;trigger para entrar
        ?p <- (MAIN::Link designadorBancaAfro)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DesignadorBancaAfro) (texto ?resposta)) ;hipótese
backward-chaining

   =>

   ;caminho a seguir
   (assert (MAIN::LinkQualificado (id questionarioDisparado) (item afro)))

        (retract ?p) ;elimina fato desnecessário



)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L
```

```
;regra inicial
(defrule iniciaMontagemAfro

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item afro))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item afro) (status definido)))

   ;busca a resposta
        (explicit (MAIN::Resposta (id ch_PrazoAfro) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item afro) (subItem 6))) ;o parágrafo que será processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_PrazoAfro))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DesignadorBancaAfro) (item afro)))


)


;processar a variável ch_DesignadorBancaAfro
(defrule processarChDesignadorBancaAfro

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_DesignadorBancaAfro) (item afro))

   ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item afro) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DesignadorBancaAfro) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item afro) (subItem 6))) ;o parágrafo que será processado

   =>
```

```
        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "pel" (iniciaMinuscula
?resposta))) ch_DesignadorBancaAfro))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item afro)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)


;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S


;regra para alterar o status dos parágrafos para ok
(defrule finalizaAfro

   (declare (no-loop TRUE))

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item afro))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (armazenado nil) (item afro) (subItem 6)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
afro)))

)


;----------------------F I M ----------------------------------------
```

Cargos

```
;Script que coloca fatos necessários para a montagem do questionário do item
cargos

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisCargos

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item cargos) (tipo item))

        (MAIN::Variavel (id ch_NumeroDeCargos) (item cargos)
        (descricao "O sistema necessita saber quantos cargos o concurso irá prover.
Exemplo de resposta: 5."))

        (MAIN::Variavel (id ch_Cargos) (item cargos)
        (descricao "O usuário deve informar ao sistema o cargo. Exemplo de
resposta: Advogado."))

        (MAIN::Variavel (id ch_Requisitos) (item cargos)
        (descricao "O usuário deve informar quais são os requisitos para o cargo
inserido (Quando existir
mais de um requisito o usuário deverá separa-los por vírgula). Exemplo de resposta:
Curso Superior em Direito,
Registro na Ordem dos Advogados do Brasil."))

        (MAIN::Variavel (id ch_CargaHorariaSemanal) (item cargos)
        (descricao "O usuário deve informar ao sistema qual é a carga horária
semanal, em horas, para o cargo inserido.
Exemplo de resposta: 40."))

        (MAIN::Variavel (id ch_Vagas) (item cargos)
        (descricao "O usuário deve informar ao sistema qual é a quantidade de vagas
para o cargo inserido.
Exemplo de resposta: 10."))

        (MAIN::Variavel (id ch_VagasAfro) (item cargos)
        (descricao "O usuário deve informar ao sistema qual é a quantidade de vagas
destinadas ao item
Afro-Descendentes. Exemplo de resposta: 2."))

        (MAIN::Variavel (id ch_VagasDefi) (item cargos)
        (descricao "O usuário deve informar ao sistema qual é a quantidade de vagas
destinadas ao item
Pessoas com Deficiência. Exemplo de resposta: 2."))
```

```
        (MAIN::Variavel (id ch_Remuneracao) (item cargos)
        (descricao "O usuário deve informar ao sistema o valor da remuneração para
o cargo inserido.
Exemplo de resposta: 2.117,03"))

        (MAIN::Variavel (id ch_TaxaInscricao) (item cargos)
        (descricao "O usuário deve informar ao sistema o valor da taxa de inscrição.
Exemplo de resposta: 50,00."))


        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item cargos) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasCargos

   ;testa se colocou as variáveis
   (MAIN::ItemSetOk (item cargos) (tipo variavel))

        (MAIN::Pergunta (id ch_NumeroDeCargos) (tipo numero)
        (texto "Entre com o número de cargos que o concurso irá prover?"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item cargos) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutCargos

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item cargos) (tipo pergunta))

        (MAIN::Paragrafo (texto Os cargos, os requisitos exigidos, a carga horária
semanal, o total de vagas, a
remuneração e o valor da taxa de inscrição são as seguintes:)
```

```
(item cargos) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Ao número de vagas estabelecidas nos subitens
anteriores, serão acrescidas outras
que surgirem ou forem criadas durante o prazo de validade do concurso.)
(item cargos) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto As provas, os conteúdos programáticos que não
estiverem estabelecidos neste Edital,
deveram ser anunciados nos anexos.)
(item cargos) (integridade ok) (negrito 0) (subItem 3) (processado ok) (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item cargos) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesCargos

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item cargos))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item cargos) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item cargos) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item cargos)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaCargos
```

```
        ;testa hipótese
        (MAIN::Questionario (item cargos) (status carregado))

    ;busca a resposta na memória
    (MAIN::Resposta (id ch_NumeroDeCargos) (texto ?resposta))

        =>

        ;caminho a seguir
        (assert (MAIN::Link definirCargos))

)

;regra para montar as especificações de cargos
(defrule chamaFuncaoCargos

    ?trigger <- (MAIN::Link definirCargos)


    ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_NumeroDeCargos) (texto ?numCargos) (status
respostaDefinida)))

    ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaAfroDescendentes) (texto ?putAfro) (status
respostaDefinida)))

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto ?putDef) (status
respostaDefinida)))

    ;busca as variáveis
        (explicit (MAIN::Variavel (id ch_Cargos) (item cargos)
        (descricao ?descCargos)))

        (explicit (MAIN::Variavel (id ch_Requisitos) (item cargos)
        (descricao ?descRequisitos)))

        (explicit (MAIN::Variavel (id ch_CargaHorariaSemanal) (item cargos)
        (descricao ?descCargaHoraria)))

        (explicit (MAIN::Variavel (id ch_Vagas) (item cargos)
        (descricao ?descVagas)))

        (explicit (MAIN::Variavel (id ch_VagasAfro) (item cargos)
        (descricao ?descVagasAfro)))

        (explicit (MAIN::Variavel (id ch_VagasDefi) (item cargos)
        (descricao ?descVagasDef)))
```

```
        (explicit (MAIN::Variavel (id ch_Remuneracao) (item cargos)
        (descricao ?descRemuneracao)))

        (explicit (MAIN::Variavel (id ch_TaxaInscricao) (item cargos)
        (descricao ?descTaxa)))


        =>

    ;inicializando o contador
    (bind ?contador 1)

    ;;;inicializando o topico
    (bind ?topico 0)


        ;loop para definição das especificações
    (while (putEspecCargos ?numCargos ?contador)

      (++ ?topico)
              (assert (MAIN::Paragrafo (texto (create$ CARGO: (sym-cat (putCargo
?descCargos)))) (item cargos)
          (subItem 1) (negrito 0) (integridade ok) (processado ok)
          (armazenado nil) (topico ?topico)))


      (++ ?topico)
              (assert (MAIN::Paragrafo (texto (create$ REQUISITOS: (sym-cat
(putRequisitos ?descRequisitos)))) (item cargos)
          (subItem 1) (negrito 0) (integridade ok) (processado ok)
          (armazenado nil) (topico ?topico)))

      (++ ?topico)
              (assert (MAIN::Paragrafo (texto (create$ CARGA HORÁRIA: (sym-cat
(putCargaHoraria ?descCargaHoraria)))) (item cargos)
          (subItem 1) (negrito 0) (integridade ok) (processado ok)
          (armazenado nil) (topico ?topico)))

      (++ ?topico)
              (assert (MAIN::Paragrafo (texto (create$ VAGAS: (sym-cat (putVagas
?descVagas)))) (item cargos)
          (subItem 1) (negrito 0) (integridade ok) (processado ok)
          (armazenado nil) (topico ?topico)))


            ;se vagas para afro-descendentes
            (if (or (eq ?putAfro s) (eq ?putAfro S)) then

        (++ ?topico)
```

```
        (assert (MAIN::Paragrafo (texto (create$ VAGAS PARA PORTADORES DE
DEFICIÊNCIA:
         (sym-cat (putVagasAfro ?descVagasAfro)))) (item cargos)
        (subItem 1) (negrito 0) (integridade ok) (processado ok)
        (armazenado nil) (topico ?topico)))

          )

          ;se vagas para deficientes
          (if (or (eq ?putDef s) (eq ?putDef S)) then

       (++ ?topico)
            (assert (MAIN::Paragrafo (texto (create$ VAGAS PARA AFRO-
DESCENDENTES:
        (sym-cat (putVagasDef ?descVagasDef)))) (item cargos)
        (subItem 1) (negrito 0) (integridade ok) (processado ok)
        (armazenado nil) (topico ?topico)))

     )

     (++ ?topico)
          (assert (MAIN::Paragrafo (texto (create$ REMUNERAÇÃO: (sym-cat
(putRemuneracao ?descRemuneracao))))
     (item cargos) (subItem 1) (negrito 0) (integridade ok) (processado ok)
        (armazenado nil) (topico ?topico)))


     (++ ?topico)
          (assert (MAIN::Paragrafo (texto (create$ TAXA DE INSCRIÇÃO: (sym-cat
(putTaxa ?descTaxa))))
     (item cargos) (subItem 1) (negrito 0) (integridade ok) (processado ok)
        (armazenado nil) (topico ?topico)))

     (++ ?contador)

   )


       ;caminho a seguir
       (assert (MAIN::LinkQualificado (id questionarioDisparado) (item cargos)))

       (retract ?trigger)

)


;-----------------------------------------------------------------

;R E G R A S  P A R A  A  M O N T A G E M  D A S  S E N T E N Ç A S
```

```
;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemCargos

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item cargos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item cargos) (status definido)))


   =>


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
cargos)))



)

;-----------------------------------------------------------------
;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule finalizaCargos

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item cargos))


        =>


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
cargos)))

)

;----------------------F I M ----------------------------------------
```

Classificacao

```
;Script que coloca fatos necessários para a montagem do questionário do item
classificacao
```

```
;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisClassificacao

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item classificacao) (tipo item))


         =>

         ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item classificacao) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasClassificacao

   ;testa se colocou as variáveis
   (MAIN::ItemSetOk (item classificacao) (tipo variavel))


         =>

         ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item classificacao) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutClassificacao

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item classificacao) (tipo pergunta))

         (MAIN::Paragrafo (texto O número de questões objetivas para cada cargo e a
pontuação das provas
encontra-se nos Anexos.)
(item classificacao) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico
0))
```

```
        (MAIN::Paragrafo (texto O cálculo da pontuação da prova objetiva será igual a
soma das
respostas do candidato as quais estejam em concordância com o gabarito oficial
final da prova.)
(item classificacao) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico
0))

        (MAIN::Paragrafo (texto Será eliminado o candidato que não obtiver no
mínimo 50% -cinqüenta por cento-
dos pontos na prova objetiva.)
(item classificacao) (integridade ok) (negrito 0) (subItem 3) (processado ok) (topico
0))

        (MAIN::Paragrafo (texto Em caso de empate na classificação final, será dada
preferência ao
candidato que obtiver maior nota na prova de conhecimentos específicos.)
(item classificacao) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico
0))


        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item classificacao) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesClassificacao

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
classificacao))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item classificacao) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item classificacao) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item classificacao)))
```

```
)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaClassificacao
        ;testa hipótese
        (MAIN::Questionario (item classificacao) (status carregado))

        =>


        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item
classificacao)))


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemClassificacao

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item classificacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item classificacao) (status definido)))


   =>



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
classificacao)))



)

;-----------------------------------------------------------------
```

```
;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule finalizaClassificacao

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item classificacao))


        =>


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
classificacao)))

)

;----------------------F I M ----------------------------------------
```

Convocacao

```
;Script que coloca fatos necessários para a montagem do questionário do item
convocacao

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisConvocacao

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item convocacao) (tipo item))

        (MAIN::Variavel (id ch_QuemForneceraFormularioDeslocamento) (item
convocacao)
        (descricao "O usuário deverá informar ao sistema quem fornecerá o formulário
de deslocamento.
Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: o
Departamento de Recursos
Humanos da Secretaria Municipal de Administração."))

        (MAIN::Variavel (id ch_PrazoApresentacaoRh) (item convocacao)
        (descricao "O usuário deverá informar ao sistema qual será o prazo em dias
para os candidatos
se apresentarem no RH. Exemplo de resposta: 5."))

        (MAIN::Variavel (id ch_LocalApresentacaoRh) (item convocacao)
```

```
        (descricao "O usuário deverá informar ao sistema qual será o local para os
candidatos
aprovados se apresentarem. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta:
o Orgão de Administração do Poder."))

        (MAIN::Variavel (id ch_QuemRealizaraAvaliacaoSaudeFisicaMental) (item
convocacao)
        (descricao "O usuário deverá informar ao sistema quem realizará a avaliação
de saúde física mental.
Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: o Serviço
Médico Pericial Municipal."))

        (MAIN::Variavel (id ch_PrazoJuntadaDosDocumentos) (item convocacao)
        (descricao "O usuário deverá informar ao sistema qual será o prazo em dias
para que o candidato
reúna a documentação necessária para a nomeação. Exemplo de resposta: 5."))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item convocacao) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasConvocacao

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item convocacao) (tipo variavel))

        (MAIN::Pergunta (id ch_QuemForneceraFormularioDeslocamento) (tipo
fraseComArtigo)
        (texto "Quem fornecerá o formulário de deslocamento?"))

        (MAIN::Pergunta (id ch_PrazoApresentacaoRh) (tipo numero)
        (texto "Qual será o prazo para os candidatos se apresentarem no RH?"))

        (MAIN::Pergunta (id ch_LocalApresentacaoRh) (tipo fraseComArtigo)
        (texto "Qual será o local para os candidatos aprovados se apresentarem?"))

        (MAIN::Pergunta (id ch_QuemRealizaraAvaliacaoSaudeFisicaMental) (tipo
fraseComArtigo)
        (texto "Quem realizará a avaliação de saúde física mental?"))

        (MAIN::Pergunta (id ch_PrazoJuntadaDosDocumentos) (tipo numero)
```

```
        (texto "Qual é o prazo para que o candidato reúna a documentação
necessária para a nomeação?"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item convocacao) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutConvocacao

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item convocacao) (tipo pergunta))

        (MAIN::Paragrafo (texto O candidato classificado e desde que observado o
número de vagas existentes,
será convocado para a nomeação quando então, se for de sua conveniência, poderá
requerer o seu deslocamento
para o final da lista, uma única vez, mediante pedido a ser efetuado em formulário
próprio, fornecido
ch_QuemForneceraFormularioDeslocamento .)
(item convocacao) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Os candidatos convocados para nomeação terão o
prazo improrrogável de
ch_PrazoApresentacaoRh dias úteis da publicação na imprensa, do Edital de
chamamento para se apresentarem
ch_LocalApresentacaoRh , decorridos os quais e não havendo o registro do
comparecimento, o candidato será
considerado como desistente.)
(item convocacao) (integridade ok) (negrito 0) (subItem 2) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Comparecendo o convocado ao órgão de
administração do Poder, será
encaminhando para avaliação de saúde física e mental a ser realizada
ch_QuemRealizaraAvaliacaoSaudeFisicaMental
sendo que o não comparecimento na data marcada, implicará na eliminação do
convocado no processo de nomeação.)
(item convocacao) (integridade ok) (negrito 0) (subItem 3) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto O convocado aprovado na avaliação de saúde física
e mental terá
```

```
ch_PrazoJuntadaDosDocumentos dias úteis para juntada de documentos
necessários à nomeação, devendo
estes ser entregues no órgão de pessoal, em caso do não comparecimento no
citado prazo, o convocado
será considerado como desistente da vaga.)
(item convocacao) (integridade ok) (negrito 0) (subItem 4) (processado nil) (topico 0))


        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item convocacao) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesConvocacao

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
convocacao))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item convocacao) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item convocacao) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item convocacao)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaConvocacao
        ;testa hipótese
        (MAIN::Questionario (item convocacao) (status carregado))
   ;busca a resposta na memória
```

```
  (MAIN::Resposta (id ch_QuemForneceraFormularioDeslocamento) (texto
?resposta)) ;hipótese backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link prazoApresentacaoRh))

)

(defrule prazoApresentacaoRh
        ;trigger para entrar
        ?p <- (MAIN::Link prazoApresentacaoRh)
        ;buscar a resposta na memória
  (MAIN::Resposta (id ch_PrazoApresentacaoRh) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link localApresentacaoRh))

        (retract ?p) ;elimina fato desnecessário

)

(defrule localApresentacaoRh
        ;trigger para entrar
        ?p <- (MAIN::Link localApresentacaoRh)
  ;busca a resposta na memória
  (MAIN::Resposta (id ch_LocalApresentacaoRh) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link quemRealizaraAvaliacaoSaudeFisicaMental)) ;caminho a
seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

(defrule quemRealizaraAvaliacaoSaudeFisicaMental
        ;trigger para entrar
        ?p <- (MAIN::Link quemRealizaraAvaliacaoSaudeFisicaMental)
        ;buscar a resposta na memória
  (MAIN::Resposta (id ch_QuemRealizaraAvaliacaoSaudeFisicaMental) (texto
?resposta)) ;hipótese backward-chaining
```

```
        =>

        ;caminho a seguir
        (assert (MAIN::Link prazoJuntadaDosDocumentos))

        (retract ?p) ;elimina fato desnecessário

)

(defrule prazoJuntadaDosDocumentos
        ;trigger para entrar
        ?p <- (MAIN::Link prazoJuntadaDosDocumentos)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_PrazoJuntadaDosDocumentos) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item convocacao)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemConvocacao

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item convocacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item convocacao) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_QuemForneceraFormularioDeslocamento) (texto
?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 1))) ;o parágrafo que será
processado

   =>
```

```
        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$  (str-cat "pel" (iniciaMinuscula
?resposta))) ch_QuemForneceraFormularioDeslocamento))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_PrazoApresentacaoRh) (item
convocacao)))

)

;processar a variável ch_PrazoApresentacaoRh
(defrule processarChPrazoApresentacaoRh

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_PrazoApresentacaoRh) (item
convocacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item convocacao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_PrazoApresentacaoRh) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 2))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_PrazoApresentacaoRh))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_LocalApresentacaoRh) (item
convocacao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)
```

```
;processar a variável ch_LocalApresentacaoRh
(defrule processarChLocalApresentacaoRh

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_LocalApresentacaoRh) (item
convocacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item convocacao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_LocalApresentacaoRh) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 2))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "n" (iniciaMinuscula
?resposta))) ch_LocalApresentacaoRh))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id
ch_QuemRealizaraAvaliacaoSaudeFisicaMental) (item convocacao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_QuemRealizaraAvaliacaoSaudeFisicaMental
(defrule processarChQuemRealizaraAvaliacaoSaudeFisicaMental

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id
ch_QuemRealizaraAvaliacaoSaudeFisicaMental) (item convocacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item convocacao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_QuemRealizaraAvaliacaoSaudeFisicaMental)
(texto ?resposta)));hipótese backward-chaining
```

```
   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 3))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
         (bind $?newTexto (processar ?t (create$ (str-cat "pel" (iniciaMinuscula
?resposta))) ch_QuemRealizaraAvaliacaoSaudeFisicaMental))
         (modify ?p (texto ?newTexto))


         ;dispara trigger
         (assert (MAIN::LinkQualificado (id ch_PrazoJuntadaDosDocumentos) (item
convocacao)))


         ;eliminando o fato desnecessario
         (retract ?trigger)

)

;processar a variável ch_PrazoJuntadaDosDocumentos
(defrule processarChPrazoJuntadaDosDocumentos

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_PrazoJuntadaDosDocumentos) (item
convocacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item convocacao) (status definido)))

         ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_PrazoJuntadaDosDocumentos) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 4))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
         (bind $?newTexto (processar ?t (create$ ?resposta)
ch_PrazoJuntadaDosDocumentos))
         (modify ?p (texto ?newTexto))
```

```
        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
convocacao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaConvocacao

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item convocacao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item convocacao) (subItem 1)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoDois) (item convocacao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaConvocacaoDois

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoDois) (item convocacao))


   ;parágrafo que terá o status alterado
```

```
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
     (processado nil) (item convocacao) (subItem 2)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoTres) (item convocacao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaConvocacaoTres

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoTres) (item convocacao))


  ;parágrafo que terá o status alterado
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
     (processado nil) (item convocacao) (subItem 3)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoQuatro) (item convocacao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaConvocacao

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoQuatro) (item convocacao))
```

```
;parágrafo que terá o status alterado
?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
   (processado nil) (item convocacao) (subItem 4)))

      =>

      ;função que testa a existência de ch
      (bind ?resultado (paragrafoProcessado ?t))

      ;atribui o retorno do parágrafo ao slot processado
      (modify ?p (processado ?resultado))



      ;dispara trigger
      (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
convocacao)))

)

;----------------------F I M ----------------------------------------
```

DaInscricao

```
;Script que coloca fatos necessários para a montagem do questionário do item
inscrição

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisDaInscricao

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item daInscricao) (tipo item))

      (MAIN::Variavel (id ch_DataInicioDaInscricao) (item daInscricao)
      (descricao "O sistema necessita saber qual é a data inicial para a inscrição do
concurso. A resposta
deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))

      (MAIN::Variavel (id ch_HoraFimDaInscricao) (item daInscricao)
      (descricao "O sistema necessita saber qual é a hora final para a inscrição do
concurso.
A resposta deve ser no formato hora:minuto. Exemplo de resposta: 18:30."))

      (MAIN::Variavel (id ch_DataFimDaInscricao) (item daInscricao)
      (descricao "O sistema necessita saber qual é a data final para a inscrição do
concurso.
```

```
A resposta deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))

        (MAIN::Variavel (id ch_DataLimitePagamentoDaInscricao) (item daInscricao)
        (descricao "O sistema necessita saber qual é a data limite para efetuar o
pagamento da taxa de inscrição.
A resposta deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 24-04-
2007."))

        (MAIN::Variavel (id ch_DataVerificarSituacaoDaInscricao) (item daInscricao)
        (descricao "O sistema necessita saber qual é a data para o candidato verificar
a situação da inscrição,
ou seja, se foi realizada de forma efetiva ou se há alguma divergência. A resposta
deve ser fornecida no formato
dia-mês-ano. Exemplo de resposta: 24-05-2007"))

        (MAIN::Variavel (id ch_DataVerificarImpDoComprovante) (item daInscricao)
        (descricao "O sistema necessita saber qual é a data para que o candidato
possa imprimir o comprovante de
ensalamento, no qual estará indicado o local, horário e endereço de realização das
provas.
A resposta deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))


        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item daInscricao) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasDaInscricao

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item daInscricao) (tipo variavel))

        (MAIN::Pergunta (id ch_DataInicioDaInscricao) (tipo data)
        (texto "Entre com a data inicial para a inscrição do concurso."))

        (MAIN::Pergunta (id ch_HoraFimDaInscricao) (tipo hora)
        (texto "Entre com a hora final para a inscrição."))

        (MAIN::Pergunta (id ch_DataFimDaInscricao) (tipo data)
        (texto "Entre com a data final para a inscrição do concurso."))

        (MAIN::Pergunta (id ch_DataLimitePagamentoDaInscricao) (tipo data)
```

```
        (texto "Entre com a data limite para o pagamento da taxa de inscrição."))

        (MAIN::Pergunta (id ch_DataVerificarSituacaoDaInscricao) (tipo data)
        (texto "Entre com a data inicial para verificação da situação de inscrição?"))

        (MAIN::Pergunta (id ch_DataVerificarImpDoComprovante) (tipo data)
        (texto "Entre com a data para a impressão do comprovante de ensalamento"))


        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item daInscricao) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutDaInscricao

    ;testa se colocou as perguntas
    (MAIN::ItemSetOk (item daInscricao) (tipo pergunta))

        (MAIN::Paragrafo (texto As inscrições serão realizadas no período de
ch_DataInicioDaInscricao até às
ch_HoraFimDaInscricao de ch_DataFimDaInscricao somente via internet no site
www.nc.ufpr.br.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Antes de efetuar o recolhimento da taxa de inscrição
o candidato deverá
certificar-se de que preenche todos os requisitos exigidos para a participação no
Concurso Público e para
a investidura no cargo, pois a taxa, uma vez paga, não será restituída em hipótese
alguma.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Após o preenchimento do formulário de inscrição, o
candidato deverá confirmar
as informações, imprimir o boleto bancário e pagá-lo preferencialmente nas casas
lotéricas ou na rede bancária
até o dia ch_DataLimitePagamentoDaInscricao devendo ao candidato guardar o
comprovante de pagamento para a
eventualidade de atestá-lo junto ao Núcleo de Concursos.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 3) (processado nil) (topico 0))
```

(MAIN::Paragrafo (texto A inscrição somente será efetivada após a identificação eletrônica do
pagamento da taxa de inscrição.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O simples agendamento do pagamento no banco não é o suficiente
para efetivação da inscrição.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

(MAIN::Paragrafo (texto No ato da inscrição, o candidato deverá optar por um único cargo. Não
será aceito qualquer pedido de alteração de cargo para o qual o candidato se inscreveu.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 6) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Caso o candidato faça mais de uma inscrição, será considerada apenas a última.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O candidato com deficiência ou que necessite de atendimento diferenciado
para a realização da prova deverá encaminhar ao Núcleo de Concursos um laudo médico expedido no prazo de 90
dias antes do término das inscrições e um formulário próprio preenchido, disponível no site www.nc.ufpr.br.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O laudo médico e o formulário devem ser entregues pessoalmente ou
enviados pelo correio via sedex, durante o período de inscrição, ao Núcleo de Concurso – UFPR – Setor de
Ciências Agrárias – Rua dos Funcionários nº 1540 - CEP 80035-050, Curitiba – PR,)
(item daInscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 1))

(MAIN::Paragrafo (texto O laudo médico deve atestar a espécie e o grau ou nível da deficiência,
com expressa referência ao código correspondente da Classificação Internacional de Doença – CID, bem como
a provável causa da deficiência, expedido no prazo de 3 meses.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 2))

(MAIN::Paragrafo (texto Não será concedido atendimento especial a candidatos que não efetuarem o
comunicado ao Núcleo de Concursos até a data especificada.)
(item daInscricao) (integridade ok) (negrito 0) (subItem 9) (processado ok) (topico 0))

(MAIN::Paragrafo (texto A partir de ch_DataVerificarSituacaoDaInscricao , o candidato poderá verificar,

no site www.nc.ufpr.br, a situação da sua inscrição, devendo, em caso de alguma divergência, entrar em contato
com o Núcleo de Concursos, pessoalmente ou pelo telefone -0xx41- 3313-8800 Central de Atendimento ao Candidato – NC.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 10) (processado nil) (topico 0))

(MAIN::Paragrafo (texto A partir de ch_DataVerificarImpDoComprovante o candidato deverá acessar o
site www.nc.ufpr.br e imprimir o seu comprovante de ensalamento, no qual estará indicado o local, horário e
endereço de realização das provas.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 11) (processado nil) (topico 0))

(MAIN::Paragrafo (texto Não haverá isenção total ou parcial do valor da taxa de inscrição.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 12) (processado ok) (topico 0))

(MAIN::Paragrafo (texto É vedada a transferência do valor pago a título para terceiros,
assim como a inscrição para outrem.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 13) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Para efetuar a inscrição, é imprescindível o número de Cadastro de Pessoa
Física –CPF– do candidato.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 14) (processado ok) (topico 0))

(MAIN::Paragrafo (texto A inscrição será de inteira responsabilidade do candidato, assumindo
ele total responsabilidade pelas informações prestadas no formulário de inscrição e arcando com as consequências
de eventuais erros de preenchimento.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 15) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Ao se inscrever, o candidato aceita, de forma irrestrita, as condições
contidas neste Edital e, caso houver, nos seus Anexos, não podendo delas alegar desconhecimento.)
(item dalnscricao) (integridade ok) (negrito 0) (subItem 16) (processado ok) (topico 0))

=>

```
        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item daInscricao) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesDaInscricao

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
daInscricao))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item daInscricao) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item daInscricao) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item daInscricao)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaDaInscricao
        ;testa hipótese
        (MAIN::Questionario (item daInscricao) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DataInicioDaInscricao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link dataFimDaInscricao))

)
```

```
(defrule dataFimDaInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link dataFimDaInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataFimDaInscricao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link horaFimDaInscricao))

        (retract ?p) ;elimina fato desnecessário

)

(defrule horaFimDaInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link horaFimDaInscricao)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_HoraFimDaInscricao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link dataLimitePagamentoDaInscricao)) ;caminho a seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

;Regra que verifica qual é a hora final para protocolar recurso
(defrule dataLimitePagamentoDaInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link dataLimitePagamentoDaInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataLimitePagamentoDaInscricao) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataVerificarSituacaoDaInscricao))

        (retract ?p) ;elimina fato desnecessário

)
```

```
(defrule dataVerificarSituacaoDaInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link dataVerificarSituacaoDaInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataVerificarSituacaoDaInscricao) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataVerificarImpDoComprovante))

        (retract ?p) ;elimina fato desnecessário

)

(defrule dataVerificarImpDoComprovante
        ;trigger para entrar
        ?p <- (MAIN::Link dataVerificarImpDoComprovante)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataVerificarImpDoComprovante) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item daInscricao)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemDaInscricao

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_DataInicioDaInscricao) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
```

```
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataInicioDaInscricao))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataFimDaInscricao) (item
daInscricao)))

)

;processar a variável ch_DataFimDaInscricao
(defrule processarChDataFimDaInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataFimDaInscricao) (item daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataFimDaInscricao) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_DataFimDaInscricao))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_HoraFimDaInscricao) (item
daInscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)
```

```
)

;processar a variável ch_HoraFimDaInscricao
(defrule processarChHoraFimDaInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HoraFimDaInscricao) (item daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraFimDaInscricao) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_HoraFimDaInscricao))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataLimitePagamentoDaInscricao)
(item daInscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_DataLimitePagamentoDaInscricao
(defrule processarChDataLimitePagamentoDaInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataLimitePagamentoDaInscricao) (item
daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

        ;busca a resposta na memória
```

```
   (explicit (MAIN::Resposta (id ch_DataLimitePagamentoDaInscricao) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 3))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataLimitePagamentoDaInscricao))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataVerificarSituacaoDaInscricao) (item
daInscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_DataVerificarSituacaoDaInscricao
(defrule processarChDataVerificarSituacaoDaInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataVerificarSituacaoDaInscricao) (item
daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataVerificarSituacaoDaInscricao) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 10))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataVerificarSituacaoDaInscricao))
```

```
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataVerificarImpDoComprovante) (item
daInscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_DataVerificarImpDoComprovante
(defrule processarChDataVerificarImpDoComprovante

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataVerificarImpDoComprovante) (item
daInscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item daInscricao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataVerificarImpDoComprovante) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 11))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataVerificarImpDoComprovante))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
daInscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)
```

```
;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaDaInscricao

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item daInscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 1)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoTres) (item daInscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDaInscricaoTres

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoTres) (item daInscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 3)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))
```

```
        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoDez) (item daInscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDaInscricaoDez

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoDez) (item daInscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 10)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoOnze) (item daInscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDaInscricao

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoOnze) (item daInscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item daInscricao) (subItem 11)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))
```

```
        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
daInscricao)))

)

;----------------------F I M ------------------------------------
```

DeficientesFisicos

```
;Script que coloca fatos necessários para a montagem do questionário do item
deficientesFisicos

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisDeficientesFisicos

        ;testa se colocou o item
        (MAIN::ItemSetOk (item deficientesFisicos) (tipo item))

        (MAIN::Variavel (id ch_IndicaMultiprofissional) (item deficientesFisicos)
        (descricao "A resposta a esta pergunta irá informar ao sistema quem é a
pessoa
responsável pela indicação da Equipe Multiprofissional. Esta última corresponde a
uma equipe de profissionais que irá fornecer um parecer quanto ao atendimento das
condições diferenciadas solicitadas pelo candidato portador deficiência física.
Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: A
Comissão Executiva do
Concurso Público"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item deficientesFisicos) (tipo variavel)))

)

;-------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasDeficientesFisicos

        ;testa se colocou as variaveis
        (MAIN::ItemSetOk (item deficientesFisicos) (tipo variavel))
```

```
        (MAIN::Pergunta (id ch_IndicaMultiprofissional) (tipo fraseComArtigo)
        (texto "Quem designará a Equipe Multiprofissional que fornecerá o parecer ao
atendimento das
condições diferenciadas solicitadas pelo candidato portador de deficiência física?"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item deficientesFisicos) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutDeficientesFisicos

  ;testa se colocou as perguntas
  (MAIN::ItemSetOk (item deficientesFisicos) (tipo pergunta))

        (MAIN::Paragrafo (texto Às pessoas com deficiência, que pretendam fazer
uso das prerrogativas que lhes são
facultadas pelo inciso VIII do artigo 37 da Constituição Federal, de 5/10/1988, e do
artigo
3 do Decreto n. 3298/1999 e alterações posteriores, que regulamenta a Lei n.
7.853/1999,
é assegurado o direito de inscrição para os cargos oferecidos neste Edital, desde
que a
deficiência de que são portadoras seja compatível com as atribuições do cargo em
provimento.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 1)
(processado ok)
      (topico 0))

        (MAIN::Paragrafo (texto Em obediência ao disposto no item anterior fica
reservado o percentual de cinco por
cento das vagas existentes para cada cargo, de acordo com o inciso 2º do artigo 5º
da Lei n.
8.112/1990 e no inciso 1º do artigo 37 do Decreto n. 3.298/1999.) (item
deficientesFisicos) (integridade ok) (negrito 0)
      (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Considera-se pessoa com deficiência aquela que se
enquadra nas categorias discriminadas
no artigo 4º. do Decreto n.3.298/99 e suas alterações posteriores, no Decreto n.
5296/2004, assim definidas:) (item deficientesFisicos) (integridade ok)
      (negrito 0) (subItem 3) (processado ok) (topico 0))
```

(MAIN::Paragrafo (texto Deficiência física: alteração completa ou parcial de um ou mais segmentos do corpo
humano, acarretando o comprometimento da função física, apresentando-se sob a forma de paraplegia, parapesia, monoplegia, monoparesia, tetraplegia, tetrapesia, triplegia, triparesia, hemiplegia, jemiparesia, ostomia, amputação ou ausência de membro, paralisia cerebral, nanismo, membros com deformidade congênita, ou adquirida, exceto as deformidades estéticas e as que não produzam dificuldades para
o desempenho de funções,) (item deficientesFisicos) (integridade ok)
(negrito 0) (subItem 3) (processado ok) (topico 1))

(MAIN::Paragrafo (texto Deficiência auditiva: perda bilateral, parcial ou total, de quarenta e um decibéis
ou mais, aferida por audiograma nas freqüências de 500Hz, 1000Hz, 2000Hz e 3000Hz,) (item deficientesFisicos) (integridade ok)
(negrito 0) (subItem 3) (processado ok) (topico 2))

(MAIN::Paragrafo (texto Deficiência visual: cegueira, na qual a acuidade visual é igual ou menor que 0,05 no
melhor olho, com a melhor correção óptica, a baixa visão, que significa acuidade visual entre 0,3 e 0,05 no melhor olho, com a melhor correção óptica, os casos nos quais a somatória da medida do campo visual em ambos os olhos for igual ou menor que 60°, ou a ocorrência simultânea de quaisquer das condições anteriores,)
(item deficientesFisicos) (integridade ok)
(negrito 0) (subItem 3) (processado ok) (topico 3))

(MAIN::Paragrafo (texto Deficiência mental: funcionamento intelectual significativamente inferior à média,
com manifestação antes dos dezoito anos e limitações associadas a duas ou mais áreas
de habilidades adaptativas, tais como: comunicação, cuidado pessoal, habilidades sociais, utilização dos recursos da comunidade, saúde e segurança, habilidades acadêmicas, lazer e trabalho,)
(item deficientesFisicos) (integridade ok) (negrito 0) (subItem 3) (processado ok)
(topico 4))

(MAIN::Paragrafo (texto Deficiência múltipla: associação de duas ou mais deficiências.) (item deficientesFisicos) (integridade ok)
(negrito 0) (subItem 3) (processado ok) (topico 5))

(MAIN::Paragrafo (texto De acordo com o Art.1 da Convenção da Guatemala o termo deficiência significa uma
restrição física, mental ou sensorial, de natureza permanente ou transitória, que limita a
capacidade de exercer uma ou mais atividades essenciais da vida diária, causada ou
agravada pelo ambiente econômico e social - Convenção da Guatemala.) (item deficientesFisicos) (integridade ok) (negrito 0)
(subItem 4) (processado ok) (topico 0))

(MAIN::Paragrafo (texto No ato de inscrição, o candidato com deficiência poderá requerer condições diferenciadas
para a realização das provas durante o período de inscrição conforme o item 6.8 deste
edital.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 5) (processado ok)
(topico 0))

(MAIN::Paragrafo (texto São condições diferenciadas: prova ampliada, prova em Braile, prova em libras,
solicitação de ledor, transcritor, guia e intérprete de libras, intérprete para leitura labial,
mobiliário especial, equipamentos de informática, tempo adicional, etc.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 6)
(processado ok) (topico 0))

(MAIN::Paragrafo (texto Em caso de necessidade de tempo adicional para a realização das provas, o candidato
com deficiência deverá apresentar justificativa, acompanhada de parecer emitido por especialista da área de sua deficiência, durante o período de inscrição, juntamente com o
CID, conforme especificações no item 6.8 deste edital.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 7)
(processado ok) (topico 0))

(MAIN::Paragrafo (texto O atendimento às condições diferenciadas solicitadas, inclusive a possibilidade de tempo
adicional de que trata o item 6.8 ficará sujeito à análise e razoabilidade e ao parecer favorável da Equipe Multiprofissional designada ch_IndicaMultiprofissional .) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 8)
(processado nil) (topico 0))

(MAIN::Paragrafo (texto O candidato que se declarar deficiênte participará do Concurso Público em igualdade de
condições com os demais candidatos, no que diz respeito ao conteúdo das provas, à
avaliação das provas e aos critérios de aprovação, ao dia, horário e local de aplicação das
provas e à nota mínima exigida.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 9) (processado ok)
(topico 0))

(MAIN::Paragrafo (texto As vagas reservadas serão liberadas aos demais candidatos, caso não tenha ocorrido
inscrição para o concurso ou aprovação de candidato com deficiência.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 10)
(processado ok) (topico 0))

```
        (MAIN::Paragrafo (texto Os candidatos que, dentro do período de inscrição,
não atenderem aos dispositivos
mencionados nos itens 6.8 e 6.9 não serão considerados deficientes e não terão
provas
especiais preparadas.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem
11) (processado ok)
      (topico 0))

        (MAIN::Paragrafo (texto O candidato com deficiência que não realizar a
inscrição conforme as instruções dos itens
3.1, 3.2 e 3.3 perderá o direito de concorrer à reserva de vagas referido no item.)
(item deficientesFisicos) (integridade ok) (negrito 0)
      (subItem 12) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Encerrado o período de inscrições, a Equipe
Multiprofissional designada
ch_IndicaMultiprofissional analisará o laudo Médico encaminhado pelo candidato
, de acordo com o item 6.8 deste edital, e verificará se há correspondência entre o
laudo e o CID. Em caso
negativo, o candidato terá sua inscrição excluída da lista de candidatos com
deficiência,
passando a integrar a lista geral de candidatos.) (item deficientesFisicos)
(integridade ok) (negrito 0) (subItem 13)
      (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Após a investidura no cargo, a deficiência não poderá
ser argüida para justificar a
concessão de aposentadoria.) (item deficientesFisicos) (integridade ok) (negrito 0)
(subItem 14) (processado ok)
      (topico 0))

        (MAIN::Paragrafo (texto Por medida de segurança, as provas dos candidatos
com deficiência poderão ser gravadas
ou filmadas.) (item deficientesFisicos) (integridade ok) (negrito 0) (subItem 15)
(processado ok)
(topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item deficientesFisicos) (tipo layout)))


)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
```

```
(defrule hipotesesDeficientesFisicos

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
deficientesFisicos))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item deficientesFisicos) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item deficientesFisicos) (status ok))

   =>


        (assert (MAIN::LinkQualificado (id hipotesesOk) (item deficientesFisicos)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaDeficientesFisicos
        ;testa hipótese
        (MAIN::Questionario (item deficientesFisicos) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_IndicaMultiprofissional) (texto ?resposta)) ;hipótese
backward-chaining

         =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item
deficientesFisicos)))

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  A  M O N T A G E M  D A S  S E N T E N Ç A S

;regra inicial
(defrule iniciaMontagemDeficientesFisicos

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item deficientesFisicos))
```

```
   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item deficientesFisicos) (status definido)))

   ;busca a resposta
         (explicit (MAIN::Resposta (id ch_IndicaMultiprofissional) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item deficientesFisicos) (subItem 8))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
         (bind $?newTexto (processar ?t (create$ (str-cat "pel" (iniciaMinuscula
?resposta))) ch_IndicaMultiprofissional))
         (modify ?p (texto ?newTexto))


         ;dispara trigger
         (assert (MAIN::LinkQualificado (id ch_IndicaMultiprofissional) (item
deficientesFisicos)))

)

;processar a variável ch_DesignadorBancaAfro
(defrule processarChIndicaMultiprofissional

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_IndicaMultiprofissional) (item
deficientesFisicos))

   ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item deficientesFisicos) (status definido)))

         ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_IndicaMultiprofissional) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item deficientesFisicos) (subItem 13))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
         (bind $?newTexto (processar ?t (create$ (str-cat "pel" (iniciaMinuscula
?resposta))) ch_IndicaMultiprofissional))
```

```
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
deficientesFisicos)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)


;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaDeficientesFisicos

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
deficientesFisicos))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (armazenado nil) (item deficientesFisicos) (subItem 8)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo13) (item
deficientesFisicos)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDeficientesFisicos

        ;trigger
```

```
        (MAIN::LinkQualificado (id testarParagrafo13) (item deficientesFisicos))


    ;parágrafo que terá o status alterado
    ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
       (processado nil) (armazenado nil) (item deficientesFisicos) (subItem 13)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
deficientesFisicos)))

)

;----------------------F I M ----------------------------------------
```

DispoGerais

```
;Script que coloca fatos necessários para a montagem do questionário do item
dispoGerais

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisDispoGerais

    ;testa se o item já foi colocado
    (MAIN::ItemSetOk (item dispoGerais) (tipo item))

        (MAIN::Variavel (id ch_PrazoValidadeConcurso) (item dispoGerais)
        (descricao "O usuário deve informar qual é prazo de validade do Concurso em
anos. Exemplo de resposta: 2."))

        (MAIN::Variavel (id ch_QuemProrrogaConcurso) (item dispoGerais)
        (descricao "O usuário deve informar ao sistema a critério de quem fica
prorrogação do prazo de
validade do concurso. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta: a Administração Municipal. "))

        (MAIN::Variavel (id ch_NomeacaoCondicionada) (item dispoGerais)
```

```
        (descricao "O usuário deve informar ao sistema quem é responsável pela
concretização da nomeação
condicionada. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta: a Administração do Município. "))

        (MAIN::Variavel (id ch_LocalManterEnderecoAtualizado) (item dispoGerais)
        (descricao "O usuário deve informar ao sistema em qual local o candidato
deverá manter o seu endereço
atualizado. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta: a Prefeitura Municipal de São José dos Pinhais no
Departamento de Recursos Humanos. "))

        (MAIN::Variavel (id ch_PrazoRetiradaDocumentos) (item dispoGerais)
        (descricao "O usuário deve informar ao sistema qual é o prazo, em meses,
após a homologação do resultado,
para a retirada, por parte do contratante, dos materiais utilizados no concurso.
Exemplo de resposta: 6."))


        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item dispoGerais) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasDispoGerais

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item dispoGerais) (tipo variavel))

        (MAIN::Pergunta (id ch_PrazoValidadeConcurso) (tipo numero)
        (texto "Entre com o prazo de validade do concurso."))

        (MAIN::Pergunta (id ch_QuemProrrogaConcurso) (tipo fraseComArtigo)
        (texto "Quem é responsável pela prorrogação do prazo de validade do
concurso?"))

        (MAIN::Pergunta (id ch_NomeacaoCondicionada) (tipo fraseComArtigo)
        (texto "Quem é responsável pela concretização da nomeação
condicionada?"))

        (MAIN::Pergunta (id ch_LocalManterEnderecoAtualizado) (tipo
fraseComArtigo)
```

```
        (texto "Qual é o local onde o candidato deverá manter o seu endereço
atualizado?"))

        (MAIN::Pergunta (id ch_PrazoRetiradaDocumentos) (tipo numero)
        (texto "Qual é o prazo para a retirada dos materiais utilizados no concurso?"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item dispoGerais) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutDispoGerais

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item dispoGerais) (tipo pergunta))

        (MAIN::Paragrafo (texto Será automaticamente eliminado do Concurso
Público o candidato que não
cumprir as normas estabelecidas ou não preencher todos os requisitos previstos no
Edital.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Se o candidato necessitar de declaração de
participação no Concurso,
deverá dirigir-se à Inspetoria do seu local de realização de prova somente no dia do
Concurso.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O prazo de validade do presente concurso é de
ch_PrazoValidadeConcurso anos,
contados a partir da data de publicação da homologação do resultado final, podendo
ser prorrogado, uma
única vez, por igual período, a critério ch_QuemProrrogaConcurso .)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 3) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto A classificação no presente concurso não assegura
ao candidato o
direito de ingresso automático nas vagas ofertadas por este concurso, mas apenas a
expectativa de ser
nomeado, seguindo a rigorosa ordem classificatória, ficando a concretização do ato
de nomeação condicionada
às observâncias das disposições legais pertinentes e, sobretudo, ao interesse e
conveniência
```

ch_NomeacaoCondicionada .)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 4) (processado nil) (topico 0))

(MAIN::Paragrafo (texto Não será fornecido ao candidato qualquer documento comprobatório de
classificação no Concurso, valendo para este fim, a listagem publicada conforme esclarecido no item 1.3 .)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O NC poderá, por motivos relevantes, cancelar ou anular as provas em um
ou mais locais de prova, devendo os candidatos dos cursos afetados submeter-se a novas provas em datas e
locais fixados pelo NC, sem necessidade de pagamento de nova taxa de inscrição.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 6) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Quaisquer alterações nas regras fixadas neste Edital somente poderão
ser realizadas por intermédio de outro Edital.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O candidato deverá manter atualizado o seu endereço perante
ch_LocalManterEnderecoAtualizado , enquanto da validade deste concurso.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 8) (processado nil) (topico 0))

(MAIN::Paragrafo (texto A contratante terá o prazo de
ch_PrazoRetiradaDocumentos meses após a
homologação do resultado para a retirada de todo o material utilizado no concurso: provas, cartões-resposta,
etc. Decorrido esse prazo todo o material será incinerado.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 9) (processado nil) (topico 0))

(MAIN::Paragrafo (texto As despesas relativas à participação do candidato no Concurso serão
de responsabilidade do candidato.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 10) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Quaisquer alterações nas regras fixadas neste Edital somente poderão
ser realizadas por intermédio de outro Edital.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 11) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Os casos omissos deste Edital serão resolvidos pela Comissão Executiva
do Concurso Público, constituída pelo Decreto n. 1387 de 14-02-2006.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 12) (processado ok) (topico 0))

```
        (MAIN::Paragrafo (texto Este Edital entra em vigor na data de sua
publicação.)
(item dispoGerais) (integridade ok) (negrito 0) (subItem 13) (processado ok) (topico
0))


        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item dispoGerais) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesDispoGerais

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
dispoGerais))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item dispoGerais) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item dispoGerais) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item dispoGerais)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaDispoGerais
        ;testa hipótese
        (MAIN::Questionario (item dispoGerais) (status carregado))
   ;busca a resposta na memória
```

```
   (MAIN::Resposta (id ch_PrazoValidadeConcurso) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link quemProrrogaConcurso))

)

;Regra que verifica qual é a hora final para a entrega dos questionamentos
(defrule quemProrrogaConcurso
        ;trigger para entrar
        ?p <- (MAIN::Link quemProrrogaConcurso)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_QuemProrrogaConcurso) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link nomeacaoCondicionada))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data final para a entrega dos questionamentos
(defrule nomeacaoCondicionada
        ;trigger para entrar
        ?p <- (MAIN::Link nomeacaoCondicionada)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_NomeacaoCondicionada) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link localManterEnderecoAtualizado)) ;caminho a seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

;Regra que verifica qual é a hora final para protocolar recurso
(defrule localManterEnderecoAtualizado
        ;trigger para entrar
        ?p <- (MAIN::Link localManterEnderecoAtualizado)
        ;buscar a resposta na memória
```

```
   (MAIN::Resposta (id ch_LocalManterEnderecoAtualizado) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link prazoRetiradaDocumentos))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data final para protocolar recurso
(defrule prazoRetiradaDocumentos
        ;trigger para entrar
        ?p <- (MAIN::Link prazoRetiradaDocumentos)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_PrazoRetiradaDocumentos) (texto ?resposta)) ;hipótese
backward-chaining

        =>


        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item dispoGerais)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemDispoGerais

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item dispoGerais))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoGerais) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_PrazoValidadeConcurso) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
```

```
    (processado nil) (item dispoGerais) (subItem 3))) ;o parágrafo que será
processado

  =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_PrazoValidadeConcurso))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_QuemProrrogaConcurso) (item
dispoGerais)))

)

;processar a variável ch_QuemProrrogaConcurso
(defrule processarChQuemProrrogaConcurso

  ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_QuemProrrogaConcurso) (item
dispoGerais))

  ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item dispoGerais) (status definido)))

        ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_QuemProrrogaConcurso) (texto
?resposta)));hipótese backward-chaining

  ;processa somente parágrafos que ainda não foram processados
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
    (processado nil) (item dispoGerais) (subItem 3))) ;o parágrafo que será
processado

  =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "d" (iniciaMinuscula
?resposta))) ch_QuemProrrogaConcurso))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_NomeacaoCondicionada) (item
dispoGerais)))


        ;eliminando o fato desnecessario
```

```
        (retract ?trigger)

)

;processar a variável ch_NomeacaoCondicionada
(defrule processarChNomeacaoCondicionada

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_NomeacaoCondicionada) (item
dispoGerais))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoGerais) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_NomeacaoCondicionada) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoGerais) (subItem 4))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "d" (iniciaMinuscula
?resposta))) ch_NomeacaoCondicionada))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_LocalManterEnderecoAtualizado) (item
dispoGerais)))


        ;eliminando o fato desnecessario
        (retract ?trigger)




)

;processar a variável ch_LocalManterEnderecoAtualizado
(defrule processarChLocalManterEnderecoAtualizado

   ;trigger
```

```
   ?trigger <- (MAIN::LinkQualificado (id ch_LocalManterEnderecoAtualizado) (item
dispoGerais))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoGerais) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_LocalManterEnderecoAtualizado) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoGerais) (subItem 8))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_LocalManterEnderecoAtualizado))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_PrazoRetiradaDocumentos) (item
dispoGerais)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_PrazoRetiradaDocumentos
(defrule processarChPrazoRetiradaDocumentos

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_PrazoRetiradaDocumentos) (item
dispoGerais))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoGerais) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_PrazoRetiradaDocumentos) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
```

```
    (processado nil) (item dispoGerais) (subItem 9))) ;o parágrafo que será
processado

  =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_PrazoRetiradaDocumentos))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
dispoGerais)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaDispoGerais

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item dispoGerais))


  ;parágrafo que terá o status alterado
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
    (processado nil) (item dispoGerais) (subItem 3)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoQuatro) (item dispoGerais)))

)
```

```
;regra para alterar o status dos parágrafos para ok
(defrule finalizaDispoGeraisQuatro

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoQuatro) (item dispoGerais))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoGerais) (subItem 4)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoOito) (item dispoGerais)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDispoGeraisOito

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoOito) (item dispoGerais))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoGerais) (subItem 8)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoNove) (item dispoGerais)))
```

```
)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaDispoGerais

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoNove) (item dispoGerais))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoGerais) (subItem 9)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
dispoGerais)))

)

;----------------------F I M ----------------------------------------
```

DispoPreliminares

```
;Script que coloca fatos necessários para a montagem do questionário do item
dispoPreliminares

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variaveis na memória
(defrule putVariaveisDispoPreliminares

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item dispoPreliminares) (tipo item))

        (MAIN::Variavel (id ch_Apoiador) (item dispoPreliminares)
        (descricao "O sistema necessita saber quem irá apoiar o Núcleo de
Concursos para
a realização do concurso. Você deve iniciar a resposta com o artigo adequado
Exemplo: A Secretaria Municipal da Administração de São José dos Pinhais"))
```

```
        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item dispoPreliminares) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasDispoPreliminares

   ;testa se colocou as variáveis
   (MAIN::ItemSetOk (item dispoPreliminares) (tipo variavel))

        (MAIN::Pergunta (id ch_Apoiador) (tipo fraseComArtigo)
        (texto "Entre com a pessoa (física ou jurídica) que apoiará o NC para a
realização
do Concurso."))


        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item dispoPreliminares) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutdispoPreliminares

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item dispoPreliminares) (tipo pergunta))

        (MAIN::Paragrafo (texto O Concurso Público, será regido por este Edital e
executado
pelo Núcleo de Concursos da Universidade Federal do Paraná – NC da UFPR, com
o apoio
ch_Apoiador .) (item dispoPreliminares) (integridade ok) (negrito 0) (subItem 1)
(topico 0))

        (MAIN::Paragrafo (texto A seleção de que trata este Edital consistirá de prova
objetiva de
```

```
conhecimentos, de caráter eliminatório e classificatório, a ser realizada pelo
Núcleo de Concurso da Universidade Federal do Paraná.) (item dispoPreliminares)
(integridade ok) (negrito 0) (subItem 2)
(processado ok) (topico 0))

        (MAIN::Paragrafo (texto Os candidatos aprovados serão convocados através
de divulgação
ch_OutroSiteDivulgacaoResultado e ch_LocalDivulgacaoResultado .) (item
dispoPreliminares) (integridade ok) (negrito 0)
(subItem 3) (topico 0))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item dispoPreliminares) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesDispoPreliminares

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
dispoPreliminares))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item dispoPreliminares) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item dispoPreliminares) (status ok))


   =>

        (assert (MAIN::LinkQualificado (id hipotesesOk) (item dispoPreliminares)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaDispoPreliminares
```

```
        ;testa hipótese
        (MAIN::Questionario (item dispoPreliminares) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_Apoiador) (texto ?resposta)) ;hipótese backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item
dispoPreliminares)))

)



;-----------------------------------------------------------------

;R E G R A S  P A R A  A  M O N T A G E M  D A S  S E N T E N Ç A S

;regra inicial
(defrule iniciaMontagemDispoPreliminares

   ;terminou o questionário
   (MAIN::LinkQualificado (id questionarioDisparado) (item dispoPreliminares))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoPreliminares) (status definido)))

   ;busca a resposta
        (explicit (MAIN::Resposta (id ch_Apoiador) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoPreliminares) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "d" (iniciaMinuscula
?resposta))) ch_Apoiador))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_SiteDivulgacaoResultado) (item
dispoPreliminares)))

)

;processar a variável ch_SiteDivulgacaoResultado
```

```
(defrule processarChSiteDivulgacaoResultado

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_SiteDivulgacaoResultado) (item
dispoPreliminares))


   ;resposta definida
   (or (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto s)))
         (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto S))))


   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoPreliminares) (status definido)))


        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_SiteDivulgacaoResultado) (texto ?respostaSite)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoPreliminares) (subItem 3))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ nos sites www.nc.ufpr.br e
?respostaSite)
        ch_OutroSiteDivulgacaoResultado))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_LocalDivulgacaoResultado) (item
dispoPreliminares)))

        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_SiteDivulgacaoResultado
(defrule processarChNaoSiteDivulgacaoResultado

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_SiteDivulgacaoResultado) (item
dispoPreliminares))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item dispoPreliminares) (status definido)))
```

```
        ;resposta definida
   (or (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto n)))
         (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto N))))


   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoPreliminares) (subItem 3))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ no site www.nc.ufpr.br)
        ch_OutroSiteDivulgacaoResultado))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_LocalDivulgacaoResultado) (item
dispoPreliminares)))

        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_LocalDivulgacaoResultado
(defrule processarChLocalDivulgacaoResultado

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_LocalDivulgacaoResultado) (item
dispoPreliminares))

   ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item dispoPreliminares) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_LocalDivulgacaoResultado) (texto ?resposta)))
        ;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item dispoPreliminares) (subItem 3))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
```

```
        (bind $?newTexto (processar ?t (create$ (str-cat "n" (iniciaMinuscula
?resposta)))
        ch_LocalDivulgacaoResultado))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
dispoPreliminares)))

        ;eliminando o fato desnecessario
        (retract ?trigger)

)
;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciafinalizaDispoPreliminares

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
dispoPreliminares))

   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (armazenado nil) (item dispoPreliminares) (subItem 1)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo3) (item dispoPreliminares)))

)


;regra para alterar o status dos parágrafos para ok
(defrule finalizaDispoPreliminares

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo3) (item dispoPreliminares))

   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
```

```
(processado nil) (armazenado nil) (item dispoPreliminares) (subItem 3)))

=>

;função que testa a existência de ch
(bind ?resultado (paragrafoProcessado ?t))

;atribui o retorno do parágrafo ao slot processado
(modify ?p (processado ?resultado))

;dispara trigger
(assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
dispoPreliminares)))

)


;----------------------F I M ----------------------------------------
```

Inscricao

```
;Script que coloca fatos necessários para a montagem do questionário do item
inscricao

;-----------------------------------------------------------------

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisInscricao

  ;testa se o item já foi colocado
  (MAIN::ItemSetOk (item inscricao) (tipo item))

      (MAIN::Variavel (id ch_DataInicioInscricao) (item inscricao)
      (descricao "O sistema necessita saber qual é a data inicial para a inscrição do
concurso.
A resposta deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))

      (MAIN::Variavel (id ch_DataFimInscricao) (item inscricao)
      (descricao "O sistema necessita saber qual é a data final para a inscrição do
concurso.
A resposta deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))

      (MAIN::Variavel (id ch_HoraFimInscricao) (item inscricao)
      (descricao "O sistema necessita saber qual é a hora  para final para a
inscrição do concurso.
A resposta deve ser no formato hora:minuto. Exemplo de resposta: 18:30."))
```

```
        (MAIN::Variavel (id ch_DataLimitePagamentoInscricao) (item inscricao)
        (descricao "O sistema necessita saber qual é a data limite para efetuar o
pagamento
da taxa de inscrição. A resposta deve ser fornecida no formato dia-mês-ano.
Exemplo de resposta: 24-04-2007."))

        (MAIN::Variavel (id ch_DataVerificarSituacaoInscricao) (item inscricao)
        (descricao "O sistema necessita saber qual é a data para o candidato verificar
a situação da
inscrição, ou seja, se foi realizada de forma efetiva ou se há alguma divergência. A
resposta
deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 24-05-2007"))

        (MAIN::Variavel (id ch_DataVerificarImpComprovante) (item inscricao)
        (descricao "O sistema necessita saber qual é a data para que o candidato
possa imprimir o
comprovante de ensalamento, no qual estará indicado o local, horário e endereço de
realização
das provas."))

    (MAIN::Variavel (id ch_Anexos) (item inscricao)
        (descricao "Variável utilizada internamente pelo sistema, resultande do
processamento
da variável ch_NumAnexos . Será utilizada na sentença ch_Anexos "))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item inscricao) (tipo variavel)))


)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasInscricao

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item inscricao) (tipo variavel))

        (MAIN::Pergunta (id ch_DataInicioInscricao) (tipo data)
        (texto "Entre com a data inicial para a inscrição do concurso. dd-mm-aa."))

        (MAIN::Pergunta (id ch_DataFimInscricao) (tipo data)
        (texto "Entre com a data final para a inscrição do concurso. dd-mm-aa."))
```

```
        (MAIN::Pergunta (id ch_HoraFimInscricao) (tipo hora)
        (texto "Entre com a hora final para a inscrição. hh:mm."))

        (MAIN::Pergunta (id ch_DataLimitePagamentoInscricao) (tipo data)
        (texto "Entre com a data limite para o pagamento da taxa de inscrição. dd-
mm-aa."))

        (MAIN::Pergunta (id ch_DataVerificarSituacaoInscricao) (tipo data)
        (texto "Entre com a data inicial para verificação da situação de inscrição? dd-
mm-aa."))

        (MAIN::Pergunta (id ch_DataVerificarImpComprovante) (tipo data)
        (texto "Entre com a data para a impressão do comprovante de ensalamento.
dd-mm-aa."))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item inscricao) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutInscricao

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item inscricao) (tipo pergunta))

        (MAIN::Paragrafo (texto As inscrições serão realizadas no período de
ch_DataInicioInscricao até às
ch_HoraFimInscricao de ch_DataFimInscricao somente via internet, no site
www.nc.ufpr.br.)
(item inscricao) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 0))


        (MAIN::Paragrafo (texto Antes de efetuar o recolhimento da taxa de inscrição
o candidato deverá certificar-se de
que preenche todos os requisitos exigidos para a participação no Concurso Público
e para
a investidura no cargo, pois a taxa, uma vez paga, não será restituída em hipótese
alguma.)
(item inscricao) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Após o preenchimento do formulário de inscrição, o
candidato deverá confirmar as
informações, imprimir o boleto bancário e pagá-lo preferencialmente nas casas
lotéricas
```

ou na rede bancária até o dia ch_DataLimitePagamentoInscricao , devendo ao candidato
guardar o comprovante de pagamento para a eventualidade de atestá-lo junto ao Núcleo
de Concursos.)
(item inscricao) (integridade ok) (negrito 0) (subItem 3) (processado nil) (topico 0))

(MAIN::Paragrafo (texto A inscrição somente será efetivada após a identificação eletrônica do pagamento da taxa
de inscrição.) (item inscricao) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O simples agendamento do pagamento no banco não é o suficiente para efetivação da
inscrição.) (item inscricao) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

(MAIN::Paragrafo (texto No ato da inscrição, o candidato deverá optar por um único cargo. Não será aceito
qualquer pedido de alteração de cargo para o qual o candidato se inscreveu.)
(item inscricao) (integridade ok) (negrito 0) (subItem 6) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Caso o candidato faça mais de uma inscrição, será considerada apenas a última.)
(item inscricao) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O candidato com deficiência ou que necessite de atendimento diferenciado para a
realização da prova deverá encaminhar ao Núcleo de Concursos um laudo médico expedido no prazo de 90 dias antes do término das inscrições e um formulário próprio
preenchido - disponível no site www.nc.ufpr.br.)
(item inscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O laudo médico e o formulário devem ser entregues pessoalmente ou enviados pelo
correio via sedex, durante o período de inscrição, ao Núcleo de Concurso – UFPR – Setor
de Ciências Agrárias – Rua dos Funcionáros nº 1540 - CEP 80035-050, Curitiba – PR.)
(item inscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 1))

(MAIN::Paragrafo (texto O laudo médico deve atestar a espécie e o grau ou nível da deficiência, com expressa
referência ao código correspondente da Classificação Internacional de Doença – CID,
bem como a provável causa da deficiência, expedido no prazo de 3 meses.)
(item inscricao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 2))

```
(MAIN::Paragrafo (texto Não será concedido atendimento especial a
candidatos que não efetuarem o comunicado
ao Núcleo de Concursos até a data especificada.)
(item inscricao) (integridade ok) (negrito 0) (subItem 9) (processado ok) (topico 0))

(MAIN::Paragrafo (texto A partir de ch_DataVerificarSituacaoInscricao , o
candidato poderá verificar, no site
www.nc.ufpr.br, a situação da sua inscrição, devendo, em caso de
alguma divergência, entrar em contato com o Núcleo de Concursos, pessoalmente
ou pelo
telefone -0xx41- 3313-8800 Central de Atendimento ao Candidato – NC.)
(item inscricao) (integridade ok) (negrito 0) (subItem 10) (processado nil) (topico 0))

(MAIN::Paragrafo (texto A partir de ch_DataVerificarImpComprovante , o
candidato deverá acessar o site
www.nc.ufpr.br e imprimir o seu comprovante de ensalamento, no qual
estará indicado o local, horário e endereço de realização das provas.)
(item inscricao) (integridade ok) (negrito 0) (subItem 11) (processado nil) (topico 0))

(MAIN::Paragrafo (texto Não haverá isenção total ou parcial do valor da taxa
de inscrição.)
(item inscricao) (integridade ok) (negrito 0) (subItem 12) (processado ok) (topico 0))

(MAIN::Paragrafo (texto É vedada a transferência do valor pago a título para
terceiros, assim como a inscrição
para outrem.) (item inscricao) (integridade ok) (negrito 0) (subItem 13) (processado
ok) (topico 0))

(MAIN::Paragrafo (texto Para efetuar a inscrição, é imprescindível o número
de Cadastro de Pessoa Física -CPF-
do candidato.) (item inscricao) (integridade ok) (negrito 0) (subItem 14) (processado
ok) (topico 0))

(MAIN::Paragrafo (texto A inscrição será de inteira responsabilidade do
candidato, assumindo ele total
responsabilidade pelas informações prestadas no formulário de inscrição e arcando
com
as consequências de eventuais erros de preenchimento.)
(item inscricao) (integridade ok) (negrito 0) (subItem 15) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Ao se inscrever, o candidato aceita, de forma
irrestrita, as condições contidas neste Edital
ch_Anexos , não podendo delas alegar desconhecimento.)
(item inscricao) (integridade ok) (negrito 0) (subItem 16) (processado nil) (topico 0))

=>

;fato indicando sucesso na operacao
(assert (MAIN::ItemSetOk (item inscricao) (tipo layout)))
```

```
)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesInscricao

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item inscricao))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item inscricao) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item inscricao) (status ok))


   =>

        (assert (MAIN::LinkQualificado (id hipotesesOk) (item inscricao)))

)


;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule inicialnscricao
        ;testa hipótese
        (MAIN::Questionario (item inscricao) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DatalnicioInscricao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link dataFimInscricao))

)

;Regra que verifica qual é a data final para a inscrição do concurso
(defrule dataFimInscricao
        ;trigger para entrar
```

```
        ?p <- (MAIN::Link dataFimInscricao)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DataFimInscricao) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link horaFimInscricao)) ;caminho a seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

;Regra que verifica qual é a hora final para a inscrição
(defrule horaFimInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link horaFimInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HoraFimInscricao) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataLimitePagamentoInscricao))

        (retract ?p) ;elimina fato desnecessário
)

;Regra que verifica qual é a data limite para o pagamento da taxa de inscrição
(defrule dataLimitePagamentoInscricao
        ;trigger para entrar
        ?p <- (MAIN::Link dataLimitePagamentoInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataLimitePagamentoInscricao) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataVerificarSituacaoInscricao))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data inicial para verificação da situação de inscrição
(defrule dataVerificarSituacaoInscricao
```

```
        ;trigger para entrar
        ?p <- (MAIN::Link dataVerificarSituacaoInscricao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataVerificarSituacaoInscricao) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataVerificarImpComprovante))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data para a impressão do comprovante de ensalamento
(defrule dataVerificarImpComprovante
        ;trigger para entrar
        ?p <- (MAIN::Link dataVerificarImpComprovante)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataVerificarImpComprovante) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item inscricao)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemInscricao

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_DataInicioInscricao) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
```

```
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_DataInicioInscricao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataFimInscricao) (item inscricao)))



)


;processar a variável ch_DataFimInscricao
(defrule processarChDataFimInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataFimInscricao) (item inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataFimInscricao) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_DataFimInscricao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_HoraFimInscricao) (item inscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)
```

```
;processar a variável ch_HoraFimInscricao
(defrule processarChHoraFimInscricao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_HoraFimInscricao) (item inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;buscar a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraFimInscricao) (texto ?resposta))) ;hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)) ch_HoraFimInscricao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataLimitePagamentoInscricao) (item
inscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_DataLimitePagamentoInscricao
(defrule processarChDataLimitePagamentoInscricao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_DataLimitePagamentoInscricao) (item
inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;buscar a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataLimitePagamentoInscricao) (texto
?resposta))) ;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 3))) ;o parágrafo que será processado
```

```
=>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta))
ch_DataLimitePagamentoInscricao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataVerificarSituacaoInscricao) (item
inscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_DataVerificarSituacaoInscricao
(defrule processarChDataVerificarSituacaoInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataVerificarSituacaoInscricao) (item
inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;buscar a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataVerificarSituacaoInscricao) (texto
?resposta))) ;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 10))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta))
ch_DataVerificarSituacaoInscricao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataVerificarImpComprovante) (item
inscricao)))


        ;eliminando o fato desnecessario
```

```
        (retract ?trigger)

)

;processar a variável ch_DataVerificarImpComprovante
(defrule processarChDataVerificarImpComprovante

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataVerificarImpComprovante) (item
inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;buscar a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataVerificarImpComprovante) (texto
?resposta))) ;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 11))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta))
ch_DataVerificarImpComprovante))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_Anexos) (item inscricao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_Anexos
(defrule processarChAnexosInscricao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_Anexos) (item inscricao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item inscricao) (status definido)))

        ;buscar a resposta na memória
```

```
   (explicit (MAIN::Resposta (id ch_NumAnexos) (texto ?respostaNumAnexos)))
;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 16))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
         (bind $?newTexto (processar ?t (create$ (putAnexos ?respostaNumAnexos))
ch_Anexos))
         (modify ?p (texto ?newTexto))

         ;dispara trigger
         (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
inscricao)))


         ;eliminando o fato desnecessario
         (retract ?trigger)

)

;-----------------------------------------------------------------
;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S


;regra para alterar o status dos parágrafos para ok
(defrule IniciafinalizaInscricao

         ;trigger
         (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item inscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 1)))

         =>

         ;função que testa a existência de ch
         (bind ?resultado (paragrafoProcessado ?t))

         ;atribui o retorno do parágrafo ao slot processado
         (modify ?p (processado ?resultado))

         ;dispara trigger
```

```
        (assert (MAIN::LinkQualificado (id testarParagrafo3) (item inscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule IniciafinalizaInscricao3

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo3) (item inscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 3)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo10) (item inscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule IniciafinalizaInscricao10

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo10) (item inscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 10)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo11) (item inscricao)))
```

```
)

;regra para alterar o status dos parágrafos para ok
(defrule IniciafinalizaInscricao11

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo11) (item inscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 11)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo16) (item inscricao)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaInscricao

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo16) (item inscricao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item inscricao) (subItem 16)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
inscricao)))
```

```
)

;----------------------F I M ------------------------------------

Introducao

;Script que coloca fatos necessários para a montagem do item Introdução


;V A R I Á V E I S

;Regra backward-chaining para a colocação de variaveis na memória
(defrule putVariaveisIntroducao

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item introducao) (tipo item))

        (MAIN::Variavel (id ch_Publicador) (item introducao)
        (descricao "O Concurso torna-se aberto ao público mediante as atribuições
legais de alguma
pessoa física ou jurídica. A resposta a esta pergunta irá indicar ao sistema
quem é esta pessoa. Você deve iniciar a resposta com o artigo adequado.
Exemplo de resposta: A Comissão Executiva do Concurso Público"))

        (MAIN::Variavel (id ch_TipoDecretoIntroducao) (item introducao)
        (descricao "O sistema necessita saber se a resolução legal que torna pública
a
realização do concurso é do tipo Decreto, Lei ou Portaria."))

        (MAIN::Variavel (id ch_NumDecretoIntroducao) (item introducao)
        (descricao "A resposta a esta pergunta irá definir o número da resolução legal
que torna
pública a realização do concurso. Exemplo de resposta: 1387"))

        (MAIN::Variavel (id ch_DataDecretoIntroducao) (item introducao)
        (descricao "A resposta a esta pergunta irá indicar ao sistema a data da
resolução
legal que torna pública a realização do concurso. Esta deve ser fornecida no formato
dia-mês-ano. Exemplo de resposta: 14-02-2006"))

        (MAIN::Variavel (id ch_DecretoIntroducao) (item introducao)
        (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento das variáveis ch_TipoDecretoIntroducao,
ch_NumDecretoIntroducao
e ch_DataDecretoIntroducao. Será utilizada na sentença ch_DecretoIntroducao."))

        (MAIN::Variavel (id ch_NumAnexos) (item introducao)
        (descricao "Variável que indica ao sistema quantos anexos o edital possui.
Caso o
```

```
edital não possua anexos, basta o usuário digitar 0."))

        (MAIN::Variavel (id ch_Anexos) (item introducao)
        (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento da variável ch_NumAnexos. Será utilizada na sentença
ch_Anexos."))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item introducao) (tipo variavel)))

)


;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasIntroducao

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item introducao) (tipo variavel))

        (MAIN::Pergunta (id ch_Publicador) (tipo fraseComArtigo)
        (texto "Quem torna pública a realização do Concurso?"))

        (MAIN::Pergunta (id ch_DataDecretoIntroducao) (tipo data)
        (texto "Entre com  a data da resolução legal que torna pública a realização do
concurso."))

        (MAIN::Pergunta (id ch_NumDecretoIntroducao) (tipo numero)
        (texto "Entre com o número da resolução legal que torna pública a realização
do concurso."))

        (MAIN::Pergunta (id ch_TipoDecretoIntroducao) (tipo escolha) (teste 1 2 3 -1 -
2 -3 -4 -5 -6)
        (texto "Entre com o tipo da resolução legal que torna pública a realização do
concurso.
(1 Decreto, 2 Lei, 3 Portaria)"))

        (MAIN::Pergunta (id ch_NumAnexos) (tipo numero)
        (texto "Quantos anexos o Edital possui? (0 – Nenhum)"))

    =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item introducao) (tipo pergunta)))
```

```
)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O
(defrule putLayoutIntroducao

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item introducao) (tipo pergunta))


   ;testa a hipótese de parágrafo na memória
       (MAIN::Paragrafo (texto ch_Publicador no uso de suas atribuições legais e
        tendo em vista ch_DecretoIntroducao torna pública a abertura de inscrições
                        e as normas para a realização de Concurso Público para o
provimento de cargos
                        especificados no item: DA ESPECIFICAÇÃO DOS CARGOS,
VAGAS E OUTROS DADOS,
                        mediante as condições estabelecidas neste Edital ch_Anexos)
(item introducao) (negrito 0) (subItem 1)
                        (integridade ok) (topico 0))

   =>

   ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item introducao) (tipo layout)))



)


;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesIntroducao

       ;trigger
       (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item introducao))

       ;testa a hipótese dos parágrafos estarem todos processados
       (MAIN::ParagrafosProcessados (item introducao) (status ok))

       ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item introducao) (status ok))
```

```
   =>

        (assert (MAIN::LinkQualificado (id hipotesesOk) (item introducao)))


)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaIntroducao
        ;testa hipótese
        (MAIN::Questionario (item introducao) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_Publicador) (texto ?resposta)) ;hipótese backward-
chaining

   =>

   ;trigger para prosseguir
        (assert (MAIN::Link tipoDecretoIntroducao))


)

;Regra que verifica qual tipo do decreto que torna público o concurso
(defrule tipoDecretoIntroducao

        ;trigger para entrar
        ?p <- (MAIN::Link tipoDecretoIntroducao)

        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_TipoDecretoIntroducao) (texto ?t)) ;hipótese backward-
chaining

        =>

        (retract ?p) ;elimina fato desnecessário

)
```

```
;Regra que verifica qual o número e data do decreto que torna público o concurso
(defrule NumDataDecretoIntroducao

   ;confirmando existência de outro site
   (or (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao) (texto 1) (status
respostaDefinida)))
      (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao) (texto 2) (status
respostaDefinida)))
                (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao) (texto 3)
(status respostaDefinida))))

        ;buscar as respostas na memória
   (MAIN::Resposta (id ch_NumDecretoIntroducao) (texto ?t1)) ;hipótese backward-
chaining
   (MAIN::Resposta (id ch_DataDecretoIntroducao) (texto ?t2)) ;hipótese backward-
chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link numAnexos))

)



;Regra para reposta pulada (decreto que torna público o concurso)
(defrule PuladaNumDataDecretoIntroducao

   ;confirmando que a resposta está indefinida
   (and (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao)))
                (or (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao)
              (status responderDepois)))
                (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao)
                (status salvarMemoria)))
                (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao)
           (status mostrarAgenda)))))

        =>

        ;caminho a seguir
        (assert (MAIN::Link numAnexos))


)
```

```
;Regra que verifica se o edital necessita de anexos
(defrule numAnexos
        ;trigger para entrar
        ?p <- (MAIN::Link numAnexos)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_NumAnexos) (texto ?t)) ;hipótese backward-chaining

        =>

   ;caminho a seguir
   (assert (MAIN::LinkQualificado (id questionarioDisparado) (item introducao)))

        (retract ?p) ;elimina fato desnecessário


)


;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemIntroducao

   ;terminou o questionário
  (MAIN::LinkQualificado (id questionarioDisparado) (item introducao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item introducao) (status definido)))

   ;busca a resposta
        (explicit (MAIN::Resposta (id ch_Publicador) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item introducao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (iniciaMaiuscula ?resposta) ,)
ch_Publicador))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DecretoIntroducao) (item introducao)))
```

```
)

;processar a variável ch_DecretoIntroducao
(defrule processarChDecretoIntroducao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_DecretoIntroducao) (item introducao))

   ;questionário definido (todas respostas estão definidas)
        (explicit (MAIN::Questionario (item introducao) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TipoDecretoIntroducao) (texto
?respostaTipoDecretoIntroducao)));hipótese backward-chaining

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_NumDecretoIntroducao) (texto
?respostaNumDecretoIntroducao)));hipótese backward-chaining

        ;busca a resposta na memória
        (explicit (MAIN::Resposta (id ch_DataDecretoIntroducao) (texto
?respostaDataDecretoIntroducao)));hipótese backward-chaining


   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item introducao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (putTipoResolucao
?respostaTipoDecretoIntroducao) nº ?respostaNumDecretoIntroducao
             de ?respostaDataDecretoIntroducao ,) ch_DecretoIntroducao))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_Anexos) (item introducao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)


;processar a variável ch_DecretoIntroducao
```

```
(defrule processarChAnexosIntroducao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_Anexos) (item introducao))

   ;questionário definido (todas respostas estão definidas)
        (explicit (MAIN::Questionario (item introducao) (status definido)))

        ;buscar a resposta na memória
   (explicit (MAIN::Resposta (id ch_NumAnexos) (texto ?respostaNumAnexos)))
;hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item introducao) (subItem 1))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (putAnexos ?respostaNumAnexos))
ch_Anexos))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
introducao)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)


;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S


;regra para alterar o status dos parágrafos para ok
(defrule finalizaIntroducao

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item introducao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
```

```
(processado nil) (item introducao) (subItem 1)))

=>

;função que testa a existência de ch
(bind ?resultado (paragrafoProcessado ?t))

;atribui o retorno do parágrafo ao slot processado
(modify ?p (processado ?resultado))

;dispara trigger
(assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
introducao)))

)



;----------------------F I M ----------------------------------------
```

Investidura

```
;Script que coloca fatos necessários para a montagem do questionário do item
investidura

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisInvestidura

 ;testa se colocou o item
 (MAIN::ItemSetOk (item investidura) (tipo item))

      (MAIN::Variavel (id ch_TipoResolucaoDemissao) (item investidura)
      (descricao "Uma das exigências para que o candidato aprovado no concurso
público seja investido no cargo é que ele não tenha sido demitido em serviço
público municipal, estadual ou federal. Existe uma resolução legal que trata deste
assunto. O sistema necessita saber qual é o tipo desta resolução legal, se é decreto,
lei ou portaria."))

      (MAIN::Variavel (id ch_NumResolucaoDemissao) (item investidura)
      (descricao "A resposta a esta pergunta irá definir o número da resolução legal
que
trata da demissão de Serviço Público. Exemplo de resposta: 525."))

      (MAIN::Variavel (id ch_DataResolucaoDemissao) (item investidura)
```

```
        (descricao "A resposta a esta pergunta irá indicar ao sistema a data da
resolução
legal que trata da demissão de Serviço Público. Esta deve ser fornecida no formato
dia-mês-ano. Exemplo de resposta: 25-03-2004"))

        (MAIN::Variavel (id ch_ResolucaoDemissao) (item investidura)
        (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento das variáveis ch_TipoResolucaoDemissao,
ch_NumResolucaoDemissao  e ch_DataResolucaoDemissao. Será utilizada na
sentenca ch_ResolucaoDemissao."))

        (MAIN::Variavel (id ch_AnaliseFisicaMental) (item investidura)
        (descricao "O candidato aprovado no Concurso de que trata este edital deverá
ter
aptidão física e mental compatíveis às funções que irá exercer. A resposta a esta
pergunta irá indicar para o sistema quem será o órgão responsável pela verificação
do atendimento a estas condições. Você deve iniciar a resposta com o artigo
adequado.
Exemplo de resposta: o Serviço Médico Pericial"))

        (MAIN::Variavel (id ch_CarteiraHabilitacao) (item investidura)
        (descricao "Há casos em que possuir carteira de habilitação é uma condição
para que o
candidato aprovado no Concurso de que trata o edital possa ser investido no
cargo. A resposta a esta pergunta irá indicar ao sistema se esta condição se aplica a
este Concurso. Exemplo de resposta: (s) para Sim e (n) para Não (obs: em
minusculo)"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item investidura) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasInvestidura

  ;testa se colocou as variaveis
  (MAIN::ItemSetOk (item investidura) (tipo variavel))

        (MAIN::Pergunta (id ch_DataResolucaoDemissao) (tipo data)
        (texto "Entre com  a data da resolução legal que trata da demissão de Serviço
Público."))

        (MAIN::Pergunta (id ch_NumResolucaoDemissao) (tipo numero)
```

```
        (texto "Entre com o número da resolução legal que trata da demissão de
Serviço Público."))

        (MAIN::Pergunta (id ch_TipoResolucaoDemissao) (tipo escolha) (teste 1 2 3 -
1 -2 -3 -4 -5 -6)
        (texto "Entre com o tipo da resolução legal que trata da demissão de Serviço
Público (1 Decreto, 2 Lei, 3 Portaria)"))

        (MAIN::Pergunta (id ch_AnaliseFisicaMental) (tipo fraseComArtigo)
        (texto "Entre com o órgão responsável pela análise da aptidão física e mental
do
candidato aprovado."))

        (MAIN::Pergunta (id ch_CarteiraHabilitacao) (tipo escolha) (teste s n -1 -2 -3 -
4 -5 -6)
        (texto "É necessário possuir carteira de habilitação?"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item investidura) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutInvestidura

  ;testa se colocou as perguntas
  (MAIN::ItemSetOk (item investidura) (tipo pergunta))

        (MAIN::Paragrafo (texto O candidato aprovado no Concurso de que trata este
Edital, e convocado, será investido
no cargo, se atendidas as seguintes exigências:) (item investidura) (integridade ok)
(negrito 0) (subItem 1) (processado ok)
        (topico 0))

        (MAIN::Paragrafo (texto Ter nacionalidade brasileira, e aos estrangeiros os
requisitos na forma da Lei,
conforme a Constituição Federal,) (item investidura) (integridade ok) (negrito 0)
(subItem 1) (processado ok) (topico 1))

        (MAIN::Paragrafo (texto Gozar dos direitos políticos,) (item investidura)
(integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 2))

        (MAIN::Paragrafo (texto Estar em dia com as obrigações eleitorais e, se do
sexo masculino, também com as militares,)
```

(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 3))

(MAIN::Paragrafo (texto Comprovar por meio de Diploma ou Certificado o nível de escolaridade exigido para o exercício do cargo,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 4))

(MAIN::Paragrafo (texto Ter idade mínima de 18 anos,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 5))

(MAIN::Paragrafo (texto Não ter sido demitido do Serviço Público Estadual, Federal ou Municipal, ch_ResolucaoDemissao)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 6))

(MAIN::Paragrafo (texto Ter aptidão física e mental compatíveis às suas funções, através de verificação feita ch_AnaliseFisicaMental)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 7))

(MAIN::Paragrafo (texto Apresentar certidão negativa de antecedentes criminais,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 8))

(MAIN::Paragrafo (texto Apresentar declaração de bens e valores que constituam seu patrimônio e, se casado, a do cônjuge,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 9))

(MAIN::Paragrafo (texto Apresentar declaração quanto ao exercício ou não de outro cargo, emprego ou função
pública e de compatibilidade de horário para o exercício de mais de um cargo quando acumulável,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 10))

(MAIN::Paragrafo (texto Declaração sobre percepção de proventos de aposentadoria, por conta do regime geral de
previdência ou de qualquer outro regime próprio de previdência em âmbito federal, estadual ou municipal, decorrentes de cargos, emprego ou função pública, conforme artigo 40 da Constituição Federal e de serviço militar, conforme artigos 42 e 142 da Consituição Federal,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 11))

(MAIN::Paragrafo (texto Apresentar comprovante de residência,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 12))

(MAIN::Paragrafo (texto Apresentar outros documentos que se fizerem necessários à época da posse,)
(item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 13))

```
        (MAIN::Paragrafo (texto A falta de comprovação ou a constatação de
falsidade de qualquer um dos requisitos
        especificados e daqueles que vierem a ser estabelecidos, impedirá a
investidura do
        candidato ao cargo.) (item investidura) (integridade ok) (negrito 0) (subItem 2)
(processado ok)
        (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item investidura) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesInvestidura

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
investidura))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item investidura) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item investidura) (status ok))

   =>

        (assert (MAIN::LinkQualificado (id hipotesesOk) (item investidura)))

)

;-----------------------------------------------------------------


;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Iniciar Questionário
(defrule inicialnvestidura
        ;testa hipótese
```

```
        (MAIN::Questionario (item investidura) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto ?resposta)) ;hipótese
backward-chaining

   =>


)

;Regra que verifica qual é o número e a data da resolução legal que trata da
demissão de Serviço Público
(defrule numDataResolucaoDemissao


   ;confirmando existência do tipo
   (or (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto 1) (status
respostaDefinida)))
      (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto 2) (status
respostaDefinida)))
                (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto 3)
(status respostaDefinida))))

        ;buscar as respostas na memória
   (MAIN::Resposta (id ch_NumResolucaoDemissao) (texto ?respostaNum))
;hipótese backward-chaining
   (MAIN::Resposta (id ch_DataResolucaoDemissao) (texto ?respostaData))
;hipótese backward-chaining


        =>

        ;caminho a seguir
        (assert (MAIN::Link analiseFisicaMental))

)



;Regra para reposta pulada (tipo da resolução legal)
(defrule pulaNumResolucaoDemissao

   ;confirmando que a resposta está indefinida
   (and (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao)))
                (or (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao)
               (status responderDepois)))
                (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao)
                (status salvarMemoria)))
                (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao)
            (status mostrarAgenda)))))
```

```
   =>

        ;caminho a seguir
        (assert (MAIN::Link analiseFisicaMental))

)


;Regra que verifica qual é o órgão responsável pela análise da aptidão física e
mental do candidato aprovado
(defrule analiseFisicaMental
        ;trigger para entrar
        ?p <- (MAIN::Link analiseFisicaMental)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_AnaliseFisicaMental) (texto ?resposta)) ;hipótese
backward-chaining

   =>

        ;caminho a seguir
        (assert (MAIN::Link carteiraHabilitacao))

        (retract ?p) ;elimina fato desnecessário
)

;Regra que verifica se é necessário possuir carteira de habilitação
(defrule carteiraHabilitacao
        ;trigger para entrar
        ?p <- (MAIN::Link carteiraHabilitacao)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_CarteiraHabilitacao) (texto ?resposta)) ;hipótese
backward-chaining

   =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item investidura)))

        (retract ?p) ;elimina fato desnecessário

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  A  M O N T A G E M  D A S  S E N T E N Ç A S

;regra inicial
(defrule iniciaMontagemInvestidura
```

```
  ;terminou o questionário
      (MAIN::LinkQualificado (id questionarioDisparado) (item afro))

  ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item investidura) (status definido)))

  ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto
?respostaTipoResolucaoDemissao)));hipótese backward-chaining

      ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_NumResolucaoDemissao) (texto
?respostaNumResolucaoDemissao)));hipótese backward-chaining

      ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_DataResolucaoDemissao) (texto
?respostaDataResolucaoDemissao)));hipótese backward-chaining

  ;processa somente parágrafos que ainda não foram processados
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
     (processado nil) (item investidura) (subItem 1) (topico 6))) ;o parágrafo que será
processado

  =>

      ;processar o parágrafo
      (bind $?newTexto (processar ?t (create$ (putTipoResolucao
?respostaTipoResolucaoDemissao) nº ?respostaNumResolucaoDemissao
          de ?respostaDataResolucaoDemissao ,) ch_ResolucaoDemissao))
      (modify ?p (texto ?newTexto))

      ;dispara trigger
      (assert (MAIN::LinkQualificado (id ch_AnaliseFisicaMental) (item investidura)))


)

;processar a variável ch_AnaliseFisicaMental
(defrule processarChAnaliseFisicaMental

  ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_AnaliseFisicaMental) (item investidura))

  ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item investidura) (status definido)))

      ;busca a resposta na memória
```

```
  (explicit (MAIN::Resposta (id ch_AnaliseFisicaMental) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item investidura) (subItem 1) (topico 7))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ (str-cat "pel" (iniciaMinuscula
?resposta))) ch_AnaliseFisicaMental))
        (modify ?p (texto ?newTexto))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_CarteiraHabilitacao) (item investidura)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;solicitar CNH
(defrule putChCarteiraHabilitacao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_CarteiraHabilitacao) (item investidura))

   ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item investidura) (status definido)))

        ;busca a resposta na memória
        (explicit (or (MAIN::Resposta (id ch_CarteiraHabilitacao) (texto s))
               (MAIN::Resposta (id ch_CarteiraHabilitacao) (texto S))))

   =>


        (assert (MAIN::Paragrafo (texto Apresentar Carteira Nacional de Habilitação -
CNH.)
        (item investidura) (integridade ok) (negrito 0) (subItem 1) (processado ok)
(topico 14)))

        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
investidura)))
```

```
        ;eliminando o fato desnecessario
        (retract ?trigger)

)


;não solicitar CNH
(defrule NaoPutChCarteiraHabilitacao

   ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_CarteiraHabilitacao) (item investidura))

   ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item investidura) (status definido)))

        ;busca a resposta na memória
        (explicit (or (MAIN::Resposta (id ch_CarteiraHabilitacao) (texto n))
                (MAIN::Resposta (id ch_CarteiraHabilitacao) (texto N))))

   =>


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
investidura)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)



;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule finalizaInvestidura

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item investidura))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (armazenado nil) (item investidura) (subItem 1)))
```

```
        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
investidura)))

)

;----------------------F I M ----------------------------------------
```

MetaRules

```
;Script com meta regras para definir os itens necessários ao edital

;-----------------------------------------------------------------------------

;E D I T A L

;Regra para definir o modelo do edital
(defrule defineEdital

   ;não criou o Edital
        (not (MAIN::Estado modeloOk))

   =>

        ;Carregando o modelo do edital
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Modelo.clp")

   ;disparando trigger
   (assert (MAIN::Estado modeloOk))

)


;-----------------------------------------------------------------------------


;M E T A - V A R I Á V E I S

;Regra backward-chaining para a colocação de meta-variaveis na memória
```

```
(defrule CriaMetaVariaveis

  ;trigger
  (MAIN::Estado modeloOk)

      (MAIN::MetaVariavel (id ch_AbrangenciaEdital)
      (descricao "O sistema necessita saber qual é a abrangência do concurso,
se é municipal, estadual ou federal. O sistema utilizará a resposta para chegar
à algumas conclusões. Por exemplo, concursos com abrangência federal ou
estadual
deverão possuir vagas reservadas para afro-descendentes e deficientes físicos.
Além disto a abrangência do edital irá definir o tipo da lei que irá regulamentar
o provimento efetivo de cargos"))

      (MAIN::MetaVariavel (id ch_TestaIntroducao)
      (descricao "O sistema necessita saber se o item Introdução será necessário
no edital. A resposta deve ser fornecida na forma s ou n. Exemplo de resposta: s"))

      (MAIN::MetaVariavel (id ch_TestaDispoPreliminares)
      (descricao "O sistema necessita saber se você deseja acrescentar o item
Disposições Preliminares ao edital. Em caso positivo o sistema fornecerá
posteriormente um questionário específico para este item."))

      (MAIN::MetaVariavel (id ch_TestaAfroDescendentes)
      (descricao "O sistema necessita saber se o concurso abrirá vagas para
afro-descendentes. Em caso positivo o sistema fornecerá posteriormente um
questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em minusculo)"))

      (MAIN::MetaVariavel (id ch_TestaDeficientesFisicos)
      (descricao "O sistema necessita saber se o concurso abrirá vagas para
pessoas
com deficiência física. Em caso positivo o sistema fornecerá posteriormente um
questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em minusculo)"))

      (MAIN::MetaVariavel (id ch_TestaCargos)
      (descricao "O sistema necessita saber se você deseja acrescentar o item
Cargos ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em minusculo)"))

  (MAIN::MetaVariavel (id ch_TestaInvestidura)
      (descricao "O sistema necessita saber se você deseja acrescentar o item
Investidura ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em minusculo)"))

  (MAIN::MetaVariavel (id ch_TestaDaInscricao)
      (descricao "O sistema necessita saber se você deseja acrescentar o item
```

Inscrição ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaProvas)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Provas ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaRecursos)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Recursos ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaClassificacao)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Classificação ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaResultado)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Resultado ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaConvocacao)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Convocação ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaNomeacao)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Nomeação ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_TestaDispoGerais)
(descricao "O sistema necessita saber se você deseja acrescentar o item
Disposições Gerais ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item. Exemplo de resposta: (s) para Sim e
(n)para Não (Obs.: em mínusculo)"))

(MAIN::MetaVariavel (id ch_OutroSiteDivulgacaoResultado)
(descricao "O sistema necessita saber se além do site do núcleo de
concursos, o resultado

```
do concurso será divulgado em outra pagina de Internet. Exemplo de resposta: (s)
para Sim e
(n)para Não (Obs.: em mínusculo)"))

        (MAIN::MetaVariavel (id ch_SiteDivulgacaoResultado)
        (descricao "O sistema necessita saber qual é o endereço eletrônico que será
divulgado o resultado,
além do site do Núcleo de Concursos. Exemplo: www.sanepar.com.br ."))

        (MAIN::MetaVariavel (id ch_LocalDivulgacaoResultado)
        (descricao "O sistema necessita saber qual é local de divulgação do
resultado.
Inicie com artigo, exemplo de resposta: o Órgão de Imprensa do Município."))


        =>

        (assert (MAIN::MetaSetOk (tipo metaVariavel)))

)


;-----------------------------------------------------------------------------


;M E T A - P E R G U N T A S

;;Regra backward-chaining para a colocação de meta-perguntas na memória de
trabalho
(defrule CriaMetaPerguntas

        ;colocou as meta-variáveis
    (MAIN::MetaSetOk (tipo metaVariavel))

        (MAIN::Pergunta (id ch_TestaIntroducao) (tipo escolha)
        (texto "O item INTRODUÇÃO é necessário? (s ou n)")
    (teste s n -1 -2 -3 -4 -5 -6))

        (MAIN::Pergunta (id ch_TestaDispoPreliminares) (tipo escolha)
        (texto "O item DAS DISPOSIÇÕES PRELIMINARES é necessário?")
    (teste s n -1 -2 -3 -4 -5 -6))

        (MAIN::Pergunta (id ch_TestaAfroDescendentes) (tipo escolha)
        (texto "Há vagas para AFRO-DESCENDENTES?")
    (teste s n -1 -2 -3 -4 -5 -6))

        (MAIN::Pergunta (id ch_TestaDeficientesFisicos) (tipo escolha)
        (texto "Há vagas para PESSOAS COM DEFICIÊNCIA?")
    (teste s n -1 -2 -3 -4 -5 -6))
```

```
    (MAIN::Pergunta (id ch_TestaCargos) (tipo escolha)
    (texto "Há vagas para DAS ESPECIFICAÇÕES DOS CARGOS, VAGAS E
OUTROS DADOS?")
  (teste s n -1 -2 -3 -4 -5 -6))

    (MAIN::Pergunta (id ch_TestaInvestidura) (tipo escolha)
    (texto "O item DAS CONDIÇÕES PARA INVESTIDURA NO CARGO é
necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

    (MAIN::Pergunta (id ch_TestaDaInscricao) (tipo escolha)
    (texto "O item DA INSCRIÇÃO é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

    (MAIN::Pergunta (id ch_TestaProvas) (tipo escolha)
    (texto "O item DAS PROVAS é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaRecursos) (tipo escolha)
    (texto "O item DOS RECURSOS é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaClassificacao) (tipo escolha)
    (texto "O item DA APROVAÇÃO E CLASSIFICAÇÃO é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaResultado) (tipo escolha)
    (texto "O item DO RESULTADO é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaConvocacao) (tipo escolha)
    (texto "O item DA CONVOCAÇÃO é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaNomeacao) (tipo escolha)
    (texto "O item DA NOMEAÇÃO é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

  (MAIN::Pergunta (id ch_TestaDispoGerais) (tipo escolha)
    (texto "O item DAS DISPOSIÇÕES GERAIS é necessário?")
  (teste s n -1 -2 -3 -4 -5 -6))

    (MAIN::Pergunta (id ch_AbrangenciaEdital) (tipo escolha)
    (texto "Entre com o tipo do edital. (1 – Municipal, 2 – Estadual, 3 – Federal)")
  (teste 1 2 3 -1 -2 -3 -4 -5 -6))


  =>

    ;fato indicando sucesso na operacao
```

```
   (assert (MAIN::MetaSetOk (tipo pergunta)))

)


;-----------------------------------------------------------------------------
;R E G R A S  P A R A  D E T E R M I N A R  O S  Í T E N S  N E C E S S Á R I O S

;regra para determinar a abrangência do edital
(defrule iniciaMetaRules

   ;se carregou as perguntas e variáveis
        (MAIN::MetaSetOk (tipo pergunta))
        (MAIN::MetaSetOk (tipo metaVariavel))

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_AbrangenciaEdital) (texto ?resposta)) ;hipótese
backward-chaining

        =>


        ;trigger
        (assert (MAIN::Link determinarDispoGerais))

)

;regra que determina o item Resultado como sendo necessário para todos os casos
(defrule determinaDispoGerais

        ;se trigger
        ?trigger <- (MAIN::Link determinarDispoGerais)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaDispoGerais) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarResultado))
```

```
)

;regra que determina o item Resultado como sendo necessário para todos os casos
(defrule determinaResultado

        ;se trigger
        ?trigger <- (MAIN::Link determinarResultado)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaResultado) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarClassificacao))

)

;regra que determina o item Classificação como sendo necessário para todos os
casos
(defrule determinaClassificacao

        ;se trigger
        ?trigger <- (MAIN::Link determinarClassificacao)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaClassificacao) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarRecursos))

)

;regra que determina o item Recursos como sendo necessário para todos os casos
(defrule determinaRecursos

        ;se trigger
        ?trigger <- (MAIN::Link determinarRecursos)

        =>
```

```
        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaRecursos) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarProvas))

)

;regra que determina o item Provas como sendo necessário para todos os casos
(defrule determinaProvas

        ;se trigger
        ?trigger <- (MAIN::Link determinarProvas)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaProvas) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarCargos))

)

;regra que determina o item Cargos como sendo necessário para todos os casos
(defrule determinaCargos

        ;se trigger
        ?trigger <- (MAIN::Link determinarCargos)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaCargos) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarDispoPreliminares))
```

```
)

;regra que determina o item Disposições Preliminares como sendo necessário para
todos os casos
(defrule determinaDispoPreliminares

        ;se trigger
        ?trigger <- (MAIN::Link determinarDispoPreliminares)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link determinarIntroducao))

)


;regra que determina o item Introdução como sendo necessário para todos os casos
(defrule determinaIntroducao

        ;se trigger
        ?trigger <- (MAIN::Link determinarIntroducao)

        =>

        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaIntroducao) (texto s) (status
respostaDefinida)))

        ;eliminando a trigger
        (retract ?trigger)

        ;disparando outra trigger
        (assert (MAIN::Link metaRulesOk))

)

;-----------------------------------------------------------------------------
```

```
;M E T A - R E G R A S

;regra para determinar a necessidade de afro-descendentes mediante a abrangência
do concurso
(defrule determinaAfro

   ;alta prioridade na agenda
   (declare (salience 1000))

   ;se edital é estadual ou municipal
        (explicit (or (MAIN::Resposta (id ch_AbrangenciaEdital) (texto 2))
         (MAIN::Resposta (id ch_AbrangenciaEdital) (texto 3))))


        =>

   (printout WSTDOUT "Como a abrangência é estadual ou municipal, o  edital
irá necessitar abrir vagas para deficientes físicos e afro-descendentes" crlf)
   (printout WSTDOUT "---" crlf)


        ;coloca a resposta na memória
        (assert (MAIN::Resposta (id ch_TestaAfroDescendentes) (texto s) (status
respostaDefinida)))
        (assert (MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto s) (status
respostaDefinida)))


)


;regra para determinar a necessidade do site de divulgação de resultado
(defrule metaRegraSiteLocal

   ;alta prioridade na agenda
   (declare (salience 1000))

   ;itens
        (or (explicit (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto s)))
        (explicit (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto S)))
        )

        ;perguntas
        (MAIN::Pergunta (id ch_SiteDivulgacaoResultado) (tipo webSite)
        (texto "Entre com endereço eletrônico opcional para a divulgação do
resultado."))

        (MAIN::Pergunta (id ch_OutroSiteDivulgacaoResultado) (tipo escolha) (teste s
n S N -1 -2 -3 -4 -5 -6)
        (texto "Há outro site de divulgação do resultado?"))
```

```
        (MAIN::Pergunta (id ch_LocalDivulgacaoResultado) (tipo fraseComArtigo)
        (texto "Entre com o local da publicação do resultado. (Inicie com o artigo
adequado)"))

        =>

   (printout WSTDOUT "Como há o item Disposições Preliminares no edital
será necessário entrar com o site e o local para a divulgação do resultado.")
   (printout WSTDOUT "---" crlf)

   ;trigger para perguntar sobre o local de divulgação
   (assert (MAIN:Link temLocalDivulgacao))

        ;trigger para perguntar sobre o site
   (assert (MAIN::Link temSite))



)



;Regra se há endereço eletrônico (opcional) para a divulgação do resultado
(defrule metaOutroSiteDivulgacaoResultado

   ;alta prioridade na agenda
   (declare (salience 1000))

        ;trigger para entrar
        ?p <- (MAIN::Link temSite)

        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto ?resposta))
;hipótese backward-chaining

        =>

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é o endereço eletrônico (opcional) para a divulgação do
resultado
(defrule metaSiteDivulgacaoResultado

   ;alta prioridade na agenda
   (declare (salience 1000))
```

```
   ;confirmando existência de outro site
   (or (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto s)
(status respostaDefinida)))
      (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto S)
(status respostaDefinida))))

        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_SiteDivulgacaoResultado) (texto ?resposta)) ;hipótese
backward-chaining

        =>


)

;Regra para endereço eletrônico não existente para a divulgação do resultado
(defrule metaNaoSiteDivulgacaoResultado

   ;confirmando existência de outro site
   (or (explicit (MAIN::Resposta (id ch_OutroSiteDivulgacaoResultado) (texto n)
(status respostaDefinida)))
      (explicit (MAIN::Resposta (texto N) (id ch_OutroSiteDivulgacaoResultado)
(status respostaDefinida))))

        =>

        ;valor default
   (assert (MAIN::Resposta (id ch_SiteDivulgacaoResultado) (texto www.nc.ufpr.br)
            (status respostaDefinida)))

)



;Regra que verifica qual é o local para a publicação do resultado
(defrule metaLocalDivulgacaoResultado

   ;alta prioridade na agenda
   (declare (salience 1000))

        ;trigger para entrar
        ?p <- (MAIN::Link temLocalDivulgacao)

        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_LocalDivulgacaoResultado) (texto ?resposta)) ;hipótese
backward-chaining

        =>
```

```
        (retract ?p) ;elimina fato desnecessário

)



;-----------------------------------------------------------------------------

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;regra para verificar a necessidade do item Introdução
(defrule iniciaMetaQuestionario

   ;se trigger
   ?trigger <- (MAIN::Link metaRulesOk)

   ;se carregou as perguntas e variáveis
        (MAIN::MetaSetOk (tipo pergunta))
        (MAIN::MetaSetOk (tipo metaVariavel))

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaIntroducao) (texto ?resposta)) ;hipótese
backward-chaining

        =>


        ;disparando trigger
        (assert (MAIN::Link dispararDispoPreliminares))

)


;regra para verificar a necessidade do item Disposições Preliminares
(defrule MetaPerguntaDispoPreliminares

   ;se trigger
   ?trigger <- (MAIN::Link dispararDispoPreliminares)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
```

```
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararInvestidura))

)


;regra para verificar a necessidade do item Investidura
(defrule MetaPerguntaInvestidura

   ;se trigger
   ?trigger <- (MAIN::Link dispararInvestidura)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaInvestidura) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararCargos))

)

;regra para verificar a necessidade do item Cargos
(defrule MetaPerguntaCargos

   ;se trigger
   ?trigger <- (MAIN::Link dispararCargos)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaCargos) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararDaInscricao))

)

;regra para verificar a necessidade do item Inscrição
(defrule MetaPerguntaDaInscricao
```

```
   ;se trigger
   ?trigger <- (MAIN::Link dispararDaInscricao)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaDaInscricao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararProvas))

)

;regra para verificar a necessidade do item Provas
(defrule MetaPerguntaProvas

   ;se trigger
   ?trigger <- (MAIN::Link dispararProvas)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaProvas) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararRecursos))

)

;regra para verificar a necessidade do item Recursos
(defrule MetaPerguntaRecursos

   ;se trigger
   ?trigger <- (MAIN::Link dispararRecursos)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaRecursos) (texto ?resposta)) ;hipótese
backward-chaining

        =>
```

```
        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararClassificacao))

)

;regra para verificar a necessidade do item Classificação
(defrule MetaPerguntaClassificacao

   ;se trigger
   ?trigger <- (MAIN::Link dispararClassificacao)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaClassificacao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararResultado))

)

;regra para verificar a necessidade do item Resultado
(defrule MetaPerguntaResultado

   ;se trigger
   ?trigger <- (MAIN::Link dispararResultado)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaResultado) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararConvocacao))

)

;regra para verificar a necessidade do item Resultado
(defrule MetaPerguntaConvocacao
```

```
   ;se trigger
   ?trigger <- (MAIN::Link dispararConvocacao)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaConvocacao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararNomeacao))

)

;regra para verificar a necessidade do item Resultado
(defrule MetaPerguntaNomeacao

   ;se trigger
   ?trigger <- (MAIN::Link dispararNomeacao)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaNomeacao) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararDispoGerais))

)

;regra para verificar a necessidade do item dispoGerais
(defrule MetaPerguntaDispoGerais

   ;se trigger
   ?trigger <- (MAIN::Link dispararDispoGerais)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaDispoGerais) (texto ?resposta)) ;hipótese
backward-chaining

        =>
```

```
        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararDeficientesFisicos))

)


;regra para verificar a necessidade do item Deficientes Fisicos
(defrule MetaPerguntaDeficientesFisicos

   ;se trigger
   ?trigger <- (MAIN::Link dispararDeficientesFisicos)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link dispararAfro))

)

;regra para verificar a necessidade do item Afro
(defrule MetaPerguntaAfro

   ;se trigger
   ?trigger <- (MAIN::Link dispararAfro)

        ;busca a resposta na memória
        (MAIN::Resposta (id ch_TestaAfroDescendentes) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;eliminando trigger
        (retract ?trigger)

        ;disparando trigger
        (assert (MAIN::Link testaIndefinidasMeta))

)
```

```
;-----------------------------------------------------------------------

;R E G R A S  P A R A  I M P O R T A R  D A  B A S E  O S  Í T E N S  N E C E S S
Á R I O S


;regra para colocar o item Introdução na memória
(defrule metaRuleCarregarIntroducao

   (MAIN::Link metaPerguntasDefinidas)


   ;busca a resposta na memória
       (explicit (MAIN::Resposta (id ch_TestaIntroducao) (texto s)))

   ;tenta colocar o item introducao na memória
       (MAIN::Item (id introducao) (sequencia 0) (negrito 0) (status nil)
     (nome ""))

       =>

       ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item introducao) (tipo item)))

       ;Carregando o item introducao a partir da base de conhecimento
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Introducao.clp")

   ;disparando outra trigger
   (assert (MAIN::Link testaDispoPreliminares))

)


;regra para não colocar o item Introdução na memória
(defrule metaRuleNaoCarregarIntroducao

       ;se trigger
   (MAIN::Link metaPerguntasDefinidas)

   ;busca a resposta na memória
       (explicit (MAIN::Resposta (id ch_TestaIntroducao) (texto n)))

       =>
```

```
   ;disparando outra trigger
   (assert (MAIN::Link testaDispoPreliminares))

)



;regra para a colocação do item Disposições Preliminares na memória
(defrule metaRuleCarregarDispoPreliminares

        ;se trigger
        ?trigger <- (MAIN::Link testaDispoPreliminares)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto s)))

   ;tenta colocar o item afro na memória
        (MAIN::Item (id dispoPreliminares) (sequencia 1) (negrito 1) (status nil)
      (nome "DAS DISPOSIÇÕES PRELIMINARES"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item dispoPreliminares) (tipo item)))

   ;Carregando o item introducao a partir da base de conhecimento
   (batch
"C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\DispoPreliminares.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;dispara trigger
        (assert (MAIN::Link testaDeficientesFisicos))

)



;regra para a não colocação do item Disposições Preliminares na memória
(defrule metaRuleNaoCarregarDispoPreliminares

        ;se trigger
        ?trigger <- (MAIN::Link testaDispoPreliminares)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaDispoPreliminares) (texto n)))
```

```
        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaDeficientesFisicos))



)

;regra para a colocação do item Deficientes Fisicos na memória
(defrule metaRuleCarregarDeficientesFisicos

        ;se trigger
        ?trigger <- (MAIN::Link testaDeficientesFisicos)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto s)))

   ;tenta colocar o item deficentes fisicos na memória
        (MAIN::Item (id deficientesFisicos) (sequencia 3) (negrito 1) (status nil)
        (nome "DAS VAGAS DESTINADAS A PESSOAS COM DEFICIÊNCIA"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item deficientesFisicos) (tipo item)))

   ;Carregando o item introducao a partir da base de conhecimento
   (batch
"C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\DeficientesFisicos.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaAfroDescendentes))

)

;regra para a não colocação do item Deficientes Fisicos na memória
(defrule metaRuleNaoCarregarDeficientesFisicos

        ;se trigger
        ?trigger <- (MAIN::Link testaDeficientesFisicos)
```

```
        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaAfroDescendentes))



)



;regra para a colocação do item Afro-descendentes na memória
(defrule metaRuleCarregarAfro

        ;se trigger
        ?trigger <- (MAIN::Link testaAfroDescendentes)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaAfroDescendentes) (texto s)))

    ;tenta colocar o item afro na memória
        (MAIN::Item (id afro) (sequencia 2) (negrito 1) (status nil)
        (nome "DAS VAGAS DESTINADAS A AFRO-DESCENDENTES"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item afro) (tipo item)))

    ;Carregando o item introducao a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Afro.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaCargos))


)
```

```
;regra para a não colocação do item Afro-descendentes na memória
(defrule metaRuleNaoCarregarAfro

        ;se trigger
        ?trigger <- (MAIN::Link testaAfroDescendentes)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaAfroDescendentes) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaCargos))



)

;regra para a colocação do item Cargos na memória
(defrule metaRuleCarregarCargos

        ;se trigger
        ?trigger <- (MAIN::Link testaCargos)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaCargos) (texto s)))

   ;tenta colocar o item cargos na memória
        (MAIN::Item (id cargos) (sequencia 4) (negrito 1) (status nil)
        (nome "DAS ESPECIFICAÇÕES DOS CARGOS, VAGAS E OUTROS
DADOS"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item cargos) (tipo item)))

   ;Carregando o item introducao a partir da base de conhecimento
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Cargos.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaInvestidura))
```

```
)

;regra para a não colocação do item Cargos na memória
(defrule metaRuleNaoCarregarCargos

        ;se trigger
        ?trigger <- (MAIN::Link testaCargos)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaCargos) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaInvestidura))


)

;regra para a colocação do item Investidura na memória
(defrule metaRuleCarregarInvestidura

        ;se trigger
        ?trigger <- (MAIN::Link testaInvestidura)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaInvestidura) (texto s)))

   ;tenta colocar o item investidura na memória
        (MAIN::Item (id investidura) (sequencia 5) (negrito 1) (status nil)
        (nome "DAS CONDIÇÕES PARA INVESTIDURA NO CARGO"))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item investidura) (tipo item)))

   ;Carregando o item introducao a partir da base de conhecimento
   (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Investidura.clp")
```

```
        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaDaInscricao))


)


;regra para a não colocação do item Investidura na memória
(defrule metaRuleNaoCarregarInvestidura

        ;se trigger
        ?trigger <- (MAIN::Link testaInvestidura)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaInvestidura) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaDaInscricao))


)


;regra para a colocação do item Inscrição na memória
(defrule metaRuleCarregarDaInscricao

        ;se trigger
        ?trigger <- (MAIN::Link testaDaInscricao)

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_TestaDaInscricao) (texto s)))

   ;tenta colocar o item inscrição na memória
        (MAIN::Item (id daInscricao) (sequencia 6) (negrito 1) (status nil)
        (nome "DA INSCRIÇÃO"))

        =>
```

```
        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item daInscricao) (tipo item)))

    ;Carregando o item introducao a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\DaInscricao.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaProvas))


)


;regra para a não colocação do item Inscrição na memória
(defrule metaRuleNaoCarregarDaInscricao

        ;se trigger
        ?trigger <- (MAIN::Link testaDaInscricao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaDaInscricao) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaProvas))


)

;regra para a colocação do item Provas na memória
(defrule metaRuleCarregarProvas

        ;se trigger
        ?trigger <- (MAIN::Link testaProvas)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaProvas) (texto s)))

    ;tenta colocar o item provas na memória
```

```
        (MAIN::Item (id provas) (sequencia 7) (negrito 1) (status nil)
        (nome "DAS PROVAS"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item provas) (tipo item)))

    ;Carregando o item provas a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Provas.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;trigger para a finalização do edital
        (assert (MAIN::Link testaRecursos))


)


;regra para a não colocação do item Provas na memória
(defrule metaRuleNaoCarregarProvas

        ;se trigger
        ?trigger <- (MAIN::Link testaProvas)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaProvas) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaRecursos))


)

;regra para a colocação do item Recursos na memória
(defrule metaRuleCarregarRecursos

        ;se trigger
        ?trigger <- (MAIN::Link testaRecursos)
```

```
        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaRecursos) (texto s)))

    ;tenta colocar o item recursos na memória
        (MAIN::Item (id recursos) (sequencia 8) (negrito 1) (status nil)
        (nome "DOS RECURSOS"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item recursos) (tipo item)))

    ;Carregando o item provas a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Recursos.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaClassificacao))


)



;regra para a não colocação do item Recursos na memória
(defrule metaRuleNaoCarregarRecursos

        ;se trigger
        ?trigger <- (MAIN::Link testaRecursos)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaRecursos) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaClassificacao))

)

;regra para a colocação do item Classificação na memória
(defrule metaRuleCarregarClassificacao
```

```
        ;se trigger
        ?trigger <- (MAIN::Link testaClassificacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaClassificacao) (texto s)))

    ;tenta colocar o item recursos na memória
        (MAIN::Item (id classificacao) (sequencia 9) (negrito 1) (status nil)
        (nome "DA APROVAÇÃO E CLASSIFICAÇÃO"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item classificacao) (tipo item)))

    ;Carregando o item provas a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Classificacao.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaResultado))


)



;regra para a não colocação do item Classificação na memória
(defrule metaRuleNaoCarregarClassificacao

        ;se trigger
        ?trigger <- (MAIN::Link testaClassificacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaClassificacao) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaResultado))

)

;regra para a colocação do item Resultado na memória
```

```
(defrule metaRuleCarregarResultado

        ;se trigger
        ?trigger <- (MAIN::Link testaResultado)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaResultado) (texto s)))

    ;tenta colocar o item resultado na memória
        (MAIN::Item (id resultado) (sequencia 10) (negrito 1) (status nil)
        (nome "DO RESULTADO"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item resultado) (tipo item)))

    ;Carregando o item resultado a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Resultado.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaConvocacao))


)



;regra para a não colocação do item Resultado na memória
(defrule metaRuleNaoCarregarResultado

        ;se trigger
        ?trigger <- (MAIN::Link testaResultado)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaResultado) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaConvocacao))

)
```

```
;regra para a colocação do item Convocação na memória
(defrule metaRuleCarregarConvocacao

        ;se trigger
        ?trigger <- (MAIN::Link testaConvocacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaConvocacao) (texto s)))

    ;tenta colocar o item convocacao na memória
        (MAIN::Item (id convocacao) (sequencia 11) (negrito 1) (status nil)
        (nome "DA CONVOCAÇÃO"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item convocacao) (tipo item)))

    ;Carregando o item resultado a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Convocacao.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaNomeacao))


)


;regra para a não colocação do item Convocacao na memória
(defrule metaRuleNaoCarregarConvocacao

        ;se trigger
        ?trigger <- (MAIN::Link testaConvocacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaConvocacao) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)


        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaNomeacao))
```

```
)

;regra para a colocação do item Nomeação na memória
(defrule metaRuleCarregarNomeacao

        ;se trigger
        ?trigger <- (MAIN::Link testaNomeacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaNomeacao) (texto s)))

    ;tenta colocar o item convocacao na memória
        (MAIN::Item (id nomeacao) (sequencia 12) (negrito 1) (status nil)
        (nome "DA NOMEAÇÃO"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item nomeacao) (tipo item)))

    ;Carregando o item resultado a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\Nomeacao.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaDispoGerais))


)


;regra para a não colocação do item Nomeação na memória
(defrule metaRuleNaoCarregarNomeacao

        ;se trigger
        ?trigger <- (MAIN::Link testaNomeacao)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaNomeacao) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)
```

```
        ;estado do edital: itens necessários carregados
        (assert (MAIN::Link testaDispoGerais))

)

;regra para a colocação do item dispoGerais na memória
(defrule metaRuleCarregarDispoGerais

        ;se trigger
        ?trigger <- (MAIN::Link testaDispoGerais)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaDispoGerais) (texto s)))

    ;tenta colocar o item dispoGerais na memória
        (MAIN::Item (id dispoGerais) (sequencia 13) (negrito 1) (status nil)
        (nome "DAS DISPOSIÇÕES GERAIS"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item dispoGerais) (tipo item)))

    ;Carregando o item resultado a partir da base de conhecimento
    (batch "C:\\projeto\\workspace\\editalX\\Expert\\Knowledge\\DispoGerais.clp")

        ;eliminando a trigger
        (retract ?trigger)

        ;estado do edital: itens necessários carregados
        (assert (MAIN::Estado itensDefinidosECarregados))


)


;regra para a não colocação do item dispoGerais na memória
(defrule metaRuleNaoCarregarDispoGerais

        ;se trigger
        ?trigger <- (MAIN::Link testaDispoGerais)

        ;busca a resposta na memória
    (explicit (MAIN::Resposta (id ch_TestaDispoGerais) (texto n)))

        =>

        ;eliminando a trigger
        (retract ?trigger)
```

```
        ;estado do edital: itens necessários carregados
        (assert (MAIN::Estado itensDefinidosECarregados))

)

;-----------------------------------------------------------------------------

;R E G R A S  P A R A  T R A T A R  D A S  R E S P O S T A S  I N D E F I N I D A S

;dispara perguntas indefinidas do item
(defrule disparaIndefMetaRule

   ;trigger
   (MAIN::Link testaIndefinidasMeta)

        ;há respostas indefinidas?
   (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda))))

        =>

   ;dispara trigger
        (assert (MAIN::Tarefa buscarIndefinidas))

)

;verifica se há perguntas indefinidas
(defrule checaNaoRespondidasMeta

        ;alta prioridade na agenda
   (declare (salience -100))

        ;trigger
   (MAIN::Link testaIndefinidasMeta)

   ;há perguntas sem responder
    (and (explicit (MAIN::Pergunta (id ?id)))
         (not (explicit (MAIN::Resposta (id ?id)))))

   =>

        ;colocando a trigger para perguntar novamente
        (assert (MAIN::Perguntar ?id))

)
```

```
;dispara trigger informando que não há perguntas indefinidas
(defrule ChecaIndefMetaRule

        ;trigger
   ?trigger <- (MAIN::Link testaIndefinidasMeta)

        ;não há indefinidas
   (not (or (explicit (MAIN::Resposta (id ?id) (status responderDepois)))
         (explicit (MAIN::Resposta (id ?id) (status carregarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status salvarMemoria)))
         (explicit (MAIN::Resposta (id ?id) (status mostrarAgenda)))))

        =>

        ;disparando trigger
   (assert (MAIN::Link metaPerguntasDefinidas))

        ;eliminando fato desnecessário
        (retract ?trigger)

)

;---------------------------F I M-----------------------------------------------
```

Modelo

```
;Class Frame para a classe Java Edital
(deftemplate edital
   (declare (from-class Edital)
   (slot-specific TRUE)))

(defglobal ?*edital* = (new Edital 14 40 40))

(definstance edital ?*edital* static)
```

Nomeacao

```
;Script que coloca fatos necessários para a montagem do questionário do item
nomeacao

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisNomeacao
```

```
   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item nomeacao) (tipo item))

        (MAIN::Variavel (id ch_PrazoPosseCargo) (item nomeacao)
        (descricao "O usuário deve informar ao sistema qual é o prazo em dias que o
candidato
terá para tomar posse do cargo. Exemplo de resposta: 5."))

        (MAIN::Variavel (id ch_LocalNomeacao) (item nomeacao)
        (descricao "O usuário deve informar ao sistema qual é o local para a
nomeação do candidato.
Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: o Órgão
de Imprensa do Município."))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item nomeacao) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasNomeacao

   ;testa se colocou as variáveis
   (MAIN::ItemSetOk (item nomeacao) (tipo variavel))

        (MAIN::Pergunta (id ch_PrazoPosseCargo) (tipo numero)
        (texto "Qual é o prazo que o candidato terá para tomar posse do cargo?"))

        (MAIN::Pergunta (id ch_LocalNomeacao) (tipo fraseComArtigo)
        (texto "Qual é o local para a nomeação do candidato?"))


        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item nomeacao) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O
```

```
(defrule putLayoutNomeacao

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item nomeacao) (tipo pergunta))

        (MAIN::Paragrafo (texto O provimento efetivo será no nível inicial do cargo no
qual o candidato
se inscreveu, de conformidade com a Lei n. 41/89 e alterações e Lei Complementar
n. 02/04 e alterações,
obedecendo rigorosamente à ordem de classificação dos candidatos habilitados, de
acordo com a necessidade
da Administração, não havendo portanto, obrigatoriedade de admissão total dos
aprovados.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Quando da nomeação deverão ser chamados
alternada e proporcionalmente os
candidatos das duas listas, prosseguindo-se até a caducidade do concurso.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto A convocação de forma alternada iniciar-se-á com o
primeiro candidato da
lista geral, passando-se ao primeiro da lista especial. As demais convocações
seguirão pela lista geral e
especial até completar a porcentagem definida pelo certame.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 3) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Para efeitos de nomeação, os candidatos ficam
sujeitos, sob pena de
eliminação, a apresentação dos documentos constantes no Art. 17 da Lei n. 525, de
25-03-2004 e alterações
e demais, documentos conforme item 5 deste Edital.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto É vedada ao candidato aprovado e convocado, a
acumulação de cargos públicos,
salvo para os cargos previstos na Constituição Federal, artigo 37, inciso XVI, alíneas
a, b e c.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O candidato aprovado e convocado, terá
ch_PrazoPosseCargo dias contados
da publicação do ato da nomeação ch_LocalNomeacao para tomar posse do cargo,
conforme o Art. 16 da Lei 525,
de 25-03-2004 e alterações.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 6) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Caso haja demanda de novas investiduras acima do
número de vagas
```

```
estimado no item 4.1, dentro do prazo de validade do certame, a Administração
poderá convocar candidatos
aprovados, observada a ordem de classificação.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto A nomeação do candidato ao cargo fica condicionada
ao atendimento dos
requisitos básicos para investidura constante do item 5 deste Edital e da legislação
vigente.)
(item nomeacao) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto A nomeação seguirá rigorosamente a ordem
classificatória no concurso e
atenderá ao requisito de aprovação em Exame de Saúde Física e Mental, conforme
mencionado no item 11.3 .)
(item nomeacao) (integridade ok) (negrito 0) (subItem 9) (processado ok) (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item nomeacao) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesNomeacao

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item nomeacao))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item nomeacao) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item nomeacao) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item nomeacao)))

)

;-----------------------------------------------------------------

;R E G R A S
```

```
;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaNomeacao
        ;testa hipótese
        (MAIN::Questionario (item nomeacao) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_PrazoPosseCargo) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link localNomeacao))

)

(defrule localNomeacao
        ;trigger para entrar
        ?p <- (MAIN::Link localNomeacao)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_LocalNomeacao) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item nomeacao)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemNomeacao

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item nomeacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item nomeacao) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_PrazoPosseCargo) (texto ?resposta)))
```

```
   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item nomeacao) (subItem 6))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
       (bind $?newTexto (processar ?t (create$ ?resposta) ch_PrazoPosseCargo))
       (modify ?p (texto ?newTexto))


       ;dispara trigger
       (assert (MAIN::LinkQualificado (id ch_LocalNomeacao) (item nomeacao)))

)

;processar a variável ch_LocalNomeacao
(defrule processarChLocalNomeacao

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_LocalNomeacao) (item nomeacao))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item nomeacao) (status definido)))

       ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_LocalNomeacao) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item nomeacao) (subItem 6))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
       (bind $?newTexto (processar ?t (create$ (str-cat "n" (iniciaMinuscula
?resposta))) ch_LocalNomeacao))
       (modify ?p (texto ?newTexto))


       ;dispara trigger
       (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
nomeacao)))


       ;eliminando o fato desnecessario
```

```
        (retract ?trigger)

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule finalizaNomeacao

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item nomeacao))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item nomeacao) (subItem 6)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
nomeacao)))

)

;---------------------F I M -------------------------------------
```

Provas

```
;Script que coloca fatos necessários para a montagem do questionário do item
provas

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisProvas

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item provas) (tipo item))
```

```
        (MAIN::Variavel (id ch_DiaDeProvas) (item provas)
        (descricao "O sistema necessita saber qual é o dia para a realização das
provas. A
resposta necessita ser no formato dia-mês-ano. Exemplo A resposta
deve ser no formato dia-mês-ano. Exemplo de resposta: 13-09-2006."))

        (MAIN::Variavel (id ch_HoraInicioDeProvas) (item provas)
        (descricao "O sistema necessita saber em que horário as provas irão iniciar. A
resposta necessita ser no formato hora:minuto. Exemplo de resposta: 14:00"))

        (MAIN::Variavel (id ch_HoraFimDeProvas) (item provas)
        (descricao "O sistema necessita saber qual é o horário limite para que o
candidato
termine a realização das provas. A resposta necessita ser no formato hora:minuto.
Exemplo de resposta: 18:00"))

        (MAIN::Variavel (id ch_CidadeProvas) (item provas)
        (descricao "O sistema necessita saber em que cidade as provas serão
realizadas.
A resposta necessita ser no formato Cidade-UF. Exemplo: Curitiba-PR"))

        (MAIN::Variavel (id ch_HoraFechaPortoes) (item provas)
        (descricao "O sistema necessita saber qual é o horário limite em que as
portas de
acesso aos prédios onde serão realizadas as provas serão fechadas. A resposta
necessita ser no formato hora:minuto. Exemplo de resposta: 13:30"))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item provas) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasProvas

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item provas) (tipo variavel))

        (MAIN::Pergunta (id ch_DiaDeProvas) (tipo data)
        (texto "Entre com a data para a realização das provas."))

        (MAIN::Pergunta (id ch_HoraInicioDeProvas) (tipo hora)
        (texto "Entre com o horário do início da realização das provas."))
```

```
        (MAIN::Pergunta (id ch_HoraFimDeProvas) (tipo hora)
        (texto "Entre com o horário de término da realização das provas."))

        (MAIN::Pergunta (id ch_CidadeProvas) (tipo frase)
        (texto "Entre com a cidade onde as provas serão realizadas."))

        (MAIN::Pergunta (id ch_HoraFechaPortoes) (tipo hora)
        (texto "Entre com o horário de fechamento dos portões."))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item provas) (tipo pergunta)))

)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutProvas

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item provas) (tipo pergunta))

        (MAIN::Paragrafo (texto As provas serão realizadas no dia ch_DiaDeProvas
das ch_HoraInicioDeProvas às ch_HoraFimDeProvas , em ch_CidadeProvas .)
(item provas) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto O local de prova do candidato constará no respectivo
comprovante de ensalamento,
disponível no site www.nc.ufpr.br.)(item provas) (integridade ok) (negrito 0) (subItem
2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto As provas objetivas serão constituídas de questões
de múltipla escolha, com cinco
alternativas -a, b, c, d, e,- das quais apenas uma será correta.)
(item provas) (integridade ok) (negrito 0) (subItem 3) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto As respostas às questões objetivas deverão ser
transcritas para o cartão-resposta com
caneta esferográfica com tinta preta, devendo o candidato assinalar uma única
resposta
para cada questão.) (item provas) (integridade ok) (negrito 0) (subItem 4)
(processado ok) (topico 0))

        (MAIN::Paragrafo (texto Não serão consideradas questões não assinaladas
ou que contenham mais de uma
resposta, emenda ou rasura.)
(item provas) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))
```

(MAIN::Paragrafo (texto O candidato assume plena e total responsabilidade pelo correto preenchimento do cartão-resposta
e sua integridade. Não haverá substituição desse cartão, a não ser em caso de defeito em sua impressão.)
(item provas) (integridade ok) (negrito 0) (subItem 6) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Nas provas objetivas haverá para cada candidato, um caderno de prova e um cartão-resposta
identificado e numerado adequadamente. A correção dessas provas será feita por meio de leitura óptica do cartão-resposta e, quando necessário, onde couber, com a
verificação dos cálculos constantes no caderno de prova, os quais deverão atestar a consistência da resposta encontrada para cada questão.)
(item provas) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Com o objetivo de garantir a lisura e a idoneidade do Concurso, o candidato deverá
autenticar a ficha de identificação por meio de suas impressões digitais.)
(item provas) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 0))

(MAIN::Paragrafo (texto A ausência do candidato, por qualquer motivo, inclusive doença e atraso, implicará na
sua eliminação do concurso.)
(item provas) (integridade ok) (negrito 0) (subItem 9) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Para ingresso na sala de provas, o candidato deverá apresentar o comprovante de
ensalamento, juntamente com o original do documento oficial de identidade.)
(item provas) (integridade ok) (negrito 0) (subItem 10) (processado ok) (topico 0))

(MAIN::Paragrafo (texto São documentos oficiais de identidade: Carteiras e/ou Cédulas de Identidade expedidas
pelas Secretarias de Segurança, pelas Forças Armadas, pela Polícia Militar, pelo Ministério das Relações Exteriores, Cédulas de Identidade fornecidas por Ordens ou Conselhos de Classe que por lei federal valem como documento de identidade, como, por
exemplo, as do CREA e OAB, etc., a Carteira de Trabalho e Previdência Social, bem como a Carteira Nacional de Habilitação, com fotografia, na forma da Lei n. 9.053/97.)
(item provas) (integridade ok) (negrito 0) (subItem 11) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Os documentos para ingresso na sala de prova, especificados no subitem anterior, devem
estar em perfeitas condições e devem permitir, com clareza, a identificação dos candidatos.)
(item provas) (integridade ok) (negrito 0) (subItem 12) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Em caso de perda ou roubo de documentos, o candidato será admitido para realizar as

provas, desde que tenha se apresentado no local que lhe foi designado e que seja possível
fazer a verificação de seus dados junto ao Núcleo de Concursos antes da hora marcada
para início das provas.)
(item provas) (integridade ok) (negrito 0) (subItem 13) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Nas salas de prova, os candidatos não poderão manter em seu poder relógios, armas e
aparelhos eletrônicos: BIP, telefone celular, calculadora, agenda eletrônica, etc, devendo seguir as orientações fornecidas pelos aplicadores antes do início da prova. O
candidato que estiver portando qualquer desses instrumentos durante a realização da
prova será eliminado do Concurso.)
(item provas) (integridade ok) (negrito 0) (subItem 14) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Excepcionalmente e a critério médico devidamente comprovado, o candidato que estiver
impossibilitado de realizar a prova em sala de prova, fará a mesma em local especialmente designado pelo Núcleo de Concurso – NC.)
(item provas) (integridade ok) (negrito 0) (subItem 15) (processado ok) (topico 0))

(MAIN::Paragrafo (texto A candidata que tiver necessidade de amamentar durante a realização das provas deverá
levar um acompanhante, que ficará responsável pela guarda da criança. A candidata que
não levar acompanhante não realizará a prova.)
(item provas) (integridade ok) (negrito 0) (subItem 16) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Os casos citados nos itens 7.15 e 7.16 como também outro caso de emergência devem ser
comunicados ao Núcleo de Concurso pelo fone/fax -0xx41- 3313-8832.)
(item provas) (integridade ok) (negrito 0) (subItem 17) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Não será permitido o ingresso ou a permanência de pessoas estranhas ao Concurso
Público no local de aplicação das provas.)
(item provas) (integridade ok) (negrito 0) (subItem 18) (processado ok) (topico 0))

(MAIN::Paragrafo (texto As portas de acesso nas entradas onde serão realizadas as provas serão fechadas às
ch_HoraFechaPortoes . Os relógios da Comissão Organizadora do Concurso serão acertados pelo Serviço: Hora Certa Brasil Telecom – Telefone 130 .)
(item provas) (integridade ok) (negrito 0) (subItem 19) (processado nil) (topico 0))

(MAIN::Paragrafo (texto Poderá, a critério do Núcleo de Concurso, ocorrer a prorrogação no fechamento das

portas de acesso de um ou mais locais onde serão realizadas as provas, em função de
fatores externos.)
(item provas) (integridade ok) (negrito 0) (subItem 20) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Não será permitido ao candidato, durante a realização das provas, ausentar-se do recinto,
a não ser em caso especiais e acompanhando de um membro componente da equipe de
aplicação do Concurso.)
(item provas) (integridade ok) (negrito 0) (subItem 21) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Não será permitido, durante a realização da prova, o uso de bonés, gorros, chapéus, etc.)
(item provas) (integridade ok) (negrito 0) (subItem 22) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Não será permitido ao candidato alimentar-se dentro da sala de aula, devendo aqueles que
necessitar fazê-lo, por motivos médicos, solicitar ao aplicador de provas e encaminhamento à sala de inspeção.)
(item provas) (integridade ok) (negrito 0) (subItem 23) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Não será permitida, durante a realização da prova, comunicação entre os candidatos, nem
o uso de calculadoras e/ou similares, livros, anotações, réguas de cálculo, impressos ou
qualquer outro material de consulta.)
(item provas) (integridade ok) (negrito 0) (subItem 24) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O candidato somente poderá retirar-se da sala após decorrida 1:30h -uma hora e trinta
minutos- do início da prova, devendo entregar ao aplicador da prova, obrigatoriamente, o
caderno de provas e o cartão-resposta devidamente assinalado.)
(item provas) (integridade ok) (negrito 0) (subItem 25) (processado ok) (topico 0))

(MAIN::Paragrafo (texto O caderno de provas conterá um espaço designado para anotação das respostas das
questões objetivas, que poderá ser recortado e levado pelo candidato para posterior conferência do gabarito.)
(item provas) (integridade ok) (negrito 0) (subItem 26) (processado ok) (topico 0))

(MAIN::Paragrafo (texto É de responsabilidade exclusiva do candidato a identificação correta de seu local de
realização das provas e o comparecimento no horário determinado.)
(item provas) (integridade ok) (negrito 0) (subItem 27) (processado ok) (topico 0))

(MAIN::Paragrafo (texto Constatada a qualquer tempo, por meio eletrônico, estatístico, visual ou grafológico, a

```
utilização pelo candidato de procedimentos ilícitos, a prova será objeto de anulação
e
automaticamente o candidato será eliminado do Concurso, sem prejuízo das
cominações
legais civis e criminais deles decorrentes.)
(item provas) (integridade ok) (negrito 0) (subItem 28) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Será eliminado o candidato que, durante a aplicação
das provas, praticar atos contra as
normas ou disciplinas.)
(item provas) (integridade ok) (negrito 0) (subItem 29) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Os procedimentos e os critérios para correção das
provas são de responsabilidade do NC.)
(item provas) (integridade ok) (negrito 0) (subItem 30) (processado ok) (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item provas) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesProvas

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item provas))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item provas) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item provas) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item provas)))

)

;-----------------------------------------------------------------

;R E G R A S
```

```
;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaProvas
        ;testa hipótese
        (MAIN::Questionario (item provas) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DiaDeProvas) (texto ?resposta)) ;hipótese backward-
chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link horaInicioDeProvas))

)

;Regra que verifica qual é o horário do início da realização das provas
(defrule horaInicioDeProvas
        ;trigger para entrar
        ?p <- (MAIN::Link horaInicioDeProvas)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_HoraInicioDeProvas) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link horaFimDeProvas)) ;caminho a seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

;Regra que verifica qual é o horário de término da realização das provas
(defrule horaFimDeProvas
        ;trigger para entrar
        ?p <- (MAIN::Link horaFimDeProvas)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HoraFimDeProvas) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link cidadeProvas))

        (retract ?p) ;elimina fato desnecessário
)
```

```
;Regra que verifica qual é a cidade onde as provas
(defrule cidadeProvas
        ;trigger para entrar
        ?p <- (MAIN::Link cidadeProvas)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_CidadeProvas) (texto ?resposta)) ;hipótese backward-
chaining

        =>


        ;caminho a seguir
        (assert (MAIN::Link horaFechaPortoes))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é o horário de fechamento dos portões
(defrule horaFechaPortoes
        ;trigger para entrar
        ?p <- (MAIN::Link horaFechaPortoes)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HoraFechaPortoes) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item provas)))

        (retract ?p) ;elimina fato desnecessário

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemProvas

   ;terminou o questionário
   (MAIN::LinkQualificado (id questionarioDisparado) (item provas))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item provas) (status definido)))

   ;busca a resposta
```

```
   (explicit (MAIN::Resposta (id ch_DiaDeProvas) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
       (bind $?newTexto (processar ?t (create$ ?resposta) ch_DiaDeProvas))
       (modify ?p (texto ?newTexto))


       ;dispara trigger
       (assert (MAIN::LinkQualificado (id ch_HoraInicioDeProvas) (item provas)))

)

;processar a variável ch_HoraInicioDeProvas
(defrule processarChHoraInicioDeProvas

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HoraInicioDeProvas) (item provas))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item provas) (status definido)))

       ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraInicioDeProvas) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
       (bind $?newTexto (processar ?t (create$ ?resposta) ch_HoraInicioDeProvas))
       (modify ?p (texto ?newTexto))


       ;dispara trigger
       (assert (MAIN::LinkQualificado (id ch_HoraFimDeProvas) (item provas)))


       ;eliminando o fato desnecessario
       (retract ?trigger)

)
```

```
;processar a variável ch_HoraFimDeProvas
(defrule processarChHoraFimDeProvas

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HoraFimDeProvas) (item provas))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item provas) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraFimDeProvas) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 1))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_HoraFimDeProvas))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_CidadeProvas) (item provas)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_CidadeProvas
(defrule processarChCidadeProvas

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_CidadeProvas) (item provas))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item provas) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_CidadeProvas) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 1))) ;o parágrafo que será processado
```

```
=>

    ;processar o parágrafo
    (bind $?newTexto (processar ?t (create$ ?resposta) ch_CidadeProvas))
    (modify ?p (texto ?newTexto))


    ;dispara trigger
    (assert (MAIN::LinkQualificado (id ch_HoraFechaPortoes) (item provas)))


    ;eliminando o fato desnecessario
    (retract ?trigger)

)

;processar a variável ch_HoraFechaPortoes
(defrule processarChHoraFechaPortoes

  ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_HoraFechaPortoes) (item provas))

  ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item provas) (status definido)))

    ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_HoraFechaPortoes) (texto ?resposta)));hipótese
backward-chaining

  ;processa somente parágrafos que ainda não foram processados
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
     (processado nil) (item provas) (subItem 19))) ;o parágrafo que será processado

  =>

    ;processar o parágrafo
    (bind $?newTexto (processar ?t (create$ ?resposta) ch_HoraFechaPortoes))
    (modify ?p (texto ?newTexto))


    ;dispara trigger
    (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
provas)))


    ;eliminando o fato desnecessario
    (retract ?trigger)

)
```

```
;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaProvas

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item provas))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 1)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafo19) (item provas)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaProvas

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafo19) (item provas))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item provas) (subItem 19)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))
```

```
        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
provas)))

)
;----------------------F I M ----------------------------------------
```

Recursos

```
;Script que coloca fatos necessários para a montagem do questionário do item
recursos

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisRecursos

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item recursos) (tipo item))

        (MAIN::Variavel (id ch_QuantasHorasGabProvisorio) (item recursos)
        (descricao "O sistema necessita saber após quantas horas da realização das
provas
o gabarito provisório será divulgado. Exemplo de resposta: 24."))

        (MAIN::Variavel (id ch_HoraEntregaQuestionamentos) (item recursos)
        (descricao "O sistema necessita saber qual é a hora limite para a entrega do
formulário específico. Tal formulário é necessário para que o candidato apresente
algum questionamento sobre o gabarito ou à formulação das questões. A
resposta necessita ser no formato hora:minuto. Exemplo de resposta: 14:00"))

        (MAIN::Variavel (id ch_DataEntregaQuestionamentos) (item recursos)
        (descricao "O sistema necessita saber qual é a data final para a entrega do
formulário específico. Tal formulário é necessário para que o candidato apresente
algum questionamento sobre o gabarito ou à formulação das questões.
Esta deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 14-02-
2006"))

        (MAIN::Variavel (id ch_HoraFinalProtocolo) (item recursos)
        (descricao "O sistema necessita saber qual é a hora limite para protocolar o
recurso. Tal procedimento é necessário para que o candidato apresente algum
questionamento sobre o gabarito ou à formulação das questões.
A resposta necessita ser no formato hora:minuto. Exemplo de resposta: 14:00"))

        (MAIN::Variavel (id ch_DataFinalProtocolo) (item recursos)
        (descricao "O sistema necessita saber qual é a data limite para protocolar o
recurso.
Tal procedimento é necessário para que o candidato apresente algum
```

```
questionamento sobre o gabarito ou à formulação das questões.
Esta deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 14-02-
2006"))

        (MAIN::Variavel (id ch_DataGabaritoOficial) (item recursos)
        (descricao "O sistema necessita saber qual é a data de divulgação do gabarito
oficial. Sobre este gabarito não será aceito nenhum tipo de recurso.
Esta deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 14-02-
2006"))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item recursos) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasRecursos

    ;testa se colocou as variáveis
    (MAIN::ItemSetOk (item recursos) (tipo variavel))

        (MAIN::Pergunta (id ch_QuantasHorasGabProvisorio) (tipo numero)
        (texto "Quantas horas após a realização das provas será divulgado o gabarito
provisório?"))

        (MAIN::Pergunta (id ch_HoraEntregaQuestionamentos) (tipo hora)
        (texto "Entre com a hora final para a entrega dos questionamentos."))

        (MAIN::Pergunta (id ch_DataEntregaQuestionamentos) (tipo data)
        (texto "Entre com a data final para a entrega dos questionamentos."))

        (MAIN::Pergunta (id ch_HoraFinalProtocolo) (tipo hora)
        (texto "Entre com a hora final para protocolar recurso."))

        (MAIN::Pergunta (id ch_DataFinalProtocolo) (tipo data)
        (texto "Entre com a data final para protocolar recurso."))

        (MAIN::Pergunta (id ch_DataGabaritoOficial) (tipo data)
        (texto "Entre com a data de divulgação do gabarito oficial."))

        =>

        ;fato indicando sucesso na operacao
    (assert (MAIN::ItemSetOk (item recursos) (tipo pergunta)))
```

```
)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutRecursos

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item recursos) (tipo pergunta))

        (MAIN::Paragrafo (texto Até ch_QuantasHorasGabProvisorio horas após a
realização das provas, o NC publicará o gabarito provisório das
questões objetivas no site www.nc.ufpr.br.)
(item recursos) (integridade ok) (negrito 0) (subItem 1) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Serão aceitos questionamentos sobre o gabarito ou à
formulação das questões desde que
estejam em conformidade com o disposto nos subitens 3, 4, 5, 6, 7, 8 deste item.)
(item recursos) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Os questionamentos devem estar devidamente
fundamentados e apresentados em
formulário específico que estará disponível no site www.nc.ufpr.br, até as
ch_HoraEntregaQuestionamentos do dia ch_DataEntregaQuestionamentos .)
(item recursos) (integridade ok) (negrito 0) (subItem 3) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Os candidatos devem imprimir o formulário específico
e seguir rigorosamente as
instruções contidas no site www.nc.ufpr.br.)
(item recursos) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Cada questão ou item deve ser apresentado em um
formulário separado.)
(item recursos) (integridade ok) (negrito 0) (subItem 5) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O recurso deverá ser protocolado na Secretaria do
Núcleo de Concurso ou enviado
apenas 1 via por Sedex ao Núcleo de Concursos – UFPR - Setor de Ciências
Agrárias -
Rua dos Funcionários nº 1540  - CEP 80035-050, Curitiba – PR, desde que postado
até as
ch_HoraFinalProtocolo do dia ch_DataFinalProtocolo .)
(item recursos) (integridade ok) (negrito 0) (subItem 6) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Serão desconsiderados pelo NC questionamentos
que não estiverem redigidos no
formulário específico, não protocolados, fora do prazo ou que não estiverem
```

```
devidamente fundamentados, bem como encaminhados de forma diferente ao
estabelecido nos itens anteriores.)
(item recursos) (integridade ok) (negrito 0) (subItem 7) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Serão desconsiderados pelo NC questionamentos
relativos ao preenchimento do cartão-resposta.)
(item recursos) (integridade ok) (negrito 0) (subItem 8) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O recurso será apreciado por uma banca
examinadora, designada pelo Núcleo de
Concurso, que emitirá decisão fundamentada, a qual será colocada à disposição do
requerente na Secretária no Núcleo de Concursos.)
(item recursos) (integridade ok) (negrito 0) (subItem 9) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Uma vez julgados os recurso apresentados, será
emitido novo gabarito.)
(item recursos) (integridade ok) (negrito 0) (subItem 10) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto A Banca Examinadora constitui última instância para
recurso, sendo soberana em suas
decisões, razão pela qual não caberão recursos adicionais.)
(item recursos) (integridade ok) (negrito 0) (subItem 11) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Os pontos relativos às questões porventura anuladas
serão atribuídos a todos os
candidatos que fizerem a prova. Se houver alteração, por força de impugnações do
gabarito oficial provisório, de item integrante de provas, essa alteração valerá para
todos
os candidatos, independentemente de terem recorrido.)
(item recursos) (integridade ok) (negrito 0) (subItem 12) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Com exceção dos recursos previstos nos itens
anteriores, não se concederá revisão de
provas, segunda chamada, vistas, ou recontagem de pontos em qualquer das
provas,
devido às características do Concurso.)
(item recursos) (integridade ok) (negrito 0) (subItem 13) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O gabarito oficial das provas objetivas será divulgado
no site www.nc.ufpr.br até o dia ch_DataGabaritoOficial .)
(item recursos) (integridade ok) (negrito 0) (subItem 14) (processado nil) (topico 0))

        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item recursos) (tipo layout)))

)

;-----------------------------------------------------------------
```

```
;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesRecursos

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item recursos))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item recursos) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item recursos) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item recursos)))

)

;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaRecursos
        ;testa hipótese
        (MAIN::Questionario (item recursos) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_QuantasHorasGabProvisorio) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link horaEntregaQuestionamentos))

)

;Regra que verifica qual é a hora final para a entrega dos questionamentos
(defrule horaEntregaQuestionamentos
        ;trigger para entrar
        ?p <- (MAIN::Link horaEntregaQuestionamentos)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HoraEntregaQuestionamentos) (texto ?resposta))
;hipótese backward-chaining
```

```
        =>

        ;caminho a seguir
        (assert (MAIN::Link dataEntregaQuestionamentos))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data final para a entrega dos questionamentos
(defrule dataEntregaQuestionamentos
        ;trigger para entrar
        ?p <- (MAIN::Link dataEntregaQuestionamentos)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DataEntregaQuestionamentos) (texto ?resposta))
;hipótese backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link horaFinalProtocolo)) ;caminho a seguir

        ;fato tornou-se desnecessário
        (retract ?p)

)

;Regra que verifica qual é a hora final para protocolar recurso
(defrule horaFinalProtocolo
        ;trigger para entrar
        ?p <- (MAIN::Link horaFinalProtocolo)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HoraFinalProtocolo) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataFinalProtocolo))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data final para protocolar recurso
(defrule dataFinalProtocolo
        ;trigger para entrar
        ?p <- (MAIN::Link dataFinalProtocolo)
        ;buscar a resposta na memória
```

```
   (MAIN::Resposta (id ch_DataFinalProtocolo) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link dataGabaritoOficial))

        (retract ?p) ;elimina fato desnecessário

)

;Regra que verifica qual é a data de divulgação do gabarito oficial
(defrule dataGabaritoOficial
        ;trigger para entrar
        ?p <- (MAIN::Link dataGabaritoOficial)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_DataGabaritoOficial) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item recursos)))

        (retract ?p) ;elimina fato desnecessário


)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemRecursos

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item recursos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item recursos) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_QuantasHorasGabProvisorio) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 1))) ;o parágrafo que será processado
```

```
=>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_QuantasHorasGabProvisorio))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_HoraEntregaQuestionamentos) (item
recursos)))

)

;processar a variável ch_HoraEntregaQuestionamentos
(defrule processarChHoraEntregaQuestionamentos

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HoraEntregaQuestionamentos) (item
recursos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item recursos) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraEntregaQuestionamentos) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 3))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_HoraEntregaQuestionamentos))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_DataEntregaQuestionamentos) (item
recursos)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)
```

```
;processar a variável ch_DataEntregaQuestionamentos
(defrule processarChDataEntregaQuestionamentos

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataEntregaQuestionamentos) (item
recursos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item recursos) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataEntregaQuestionamentos) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 3))) ;o parágrafo que será processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataEntregaQuestionamentos))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_HoraFinalProtocolo) (item recursos)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_HoraFinalProtocolo
(defrule processarChHoraFinalProtocolo

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HoraFinalProtocolo) (item recursos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item recursos) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HoraFinalProtocolo) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
```

```
    (processado nil) (item recursos) (subItem 6))) ;o parágrafo que será processado

  =>

      ;processar o parágrafo
      (bind $?newTexto (processar ?t (create$ ?resposta) ch_HoraFinalProtocolo))
      (modify ?p (texto ?newTexto))


      ;dispara trigger
      (assert (MAIN::LinkQualificado (id ch_DataFinalProtocolo) (item recursos)))


      ;eliminando o fato desnecessario
      (retract ?trigger)

)

;processar a variável ch_DataFinalProtocolo
(defrule processarChDataFinalProtocolo

  ;trigger
  ?trigger <- (MAIN::LinkQualificado (id ch_DataFinalProtocolo) (item recursos))

  ;questionário definido (todas respostas estão definidas)
  (explicit (MAIN::Questionario (item recursos) (status definido)))

      ;busca a resposta na memória
  (explicit (MAIN::Resposta (id ch_DataFinalProtocolo) (texto ?resposta)));hipótese
backward-chaining

  ;processa somente parágrafos que ainda não foram processados
  ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
    (processado nil) (item recursos) (subItem 6))) ;o parágrafo que será processado

  =>

      ;processar o parágrafo
      (bind $?newTexto (processar ?t (create$ ?resposta) ch_DataFinalProtocolo))
      (modify ?p (texto ?newTexto))


      ;dispara trigger
      (assert (MAIN::LinkQualificado (id ch_DataGabaritoOficial) (item recursos)))


      ;eliminando o fato desnecessario
      (retract ?trigger)

)
```

```
;processar a variável ch_DataGabaritoOficial
(defrule processarChDataGabaritoOficial

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_DataGabaritoOficial) (item recursos))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item recursos) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_DataGabaritoOficial) (texto ?resposta)));hipótese
backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 14))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta) ch_DataGabaritoOficial))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
recursos)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaRecursos

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item recursos))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 1)))
```

```
        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoTres) (item recursos)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaRecursosTres

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoTres) (item recursos))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 3)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoSeis) (item recursos)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaRecursosSeis

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoSeis) (item recursos))


   ;parágrafo que terá o status alterado
```

```
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 6)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoQuatorze) (item recursos)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaRecursos

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoQuatorze) (item recursos))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item recursos) (subItem 14)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
recursos)))

)

;---------------------F I M ------------------------------------
```

Resultado

```
;Script que coloca fatos necessários para a montagem do questionário do item
resultado

;V A R I Á V E I S

;Regra backward-chaining para a colocação de variáveis na memória
(defrule putVariaveisResultado

   ;testa se o item já foi colocado
   (MAIN::ItemSetOk (item resultado) (tipo item))

        (MAIN::Variavel (id ch_DataDivulgacaoDesempenho) (item resultado)
        (descricao "O sistema necessita saber qual é a data limite em que os
candidatos
poderão consultar o seu desempenho. A resposta deverá ser fornecida no formato
dia-mês-ano.
Exemplo de resposta: 05-09-2007."))

        (MAIN::Variavel (id ch_HomologadorResultadoFinal) (item resultado)
        (descricao "O sistema necessita saber a quem caberá homologar o resultado
final do Concurso Público.
Lembre-se de utilizar os artigos adequados para a resposta a esta pergunta.
Exemplo de resposta: o Prefeito Municipal de São José dos Pinhais."))

        (MAIN::Variavel (id ch_LocalInformacoesSeisMeses) (item resultado)
        (descricao "Existe um local específico para obter informações referentes ao
concurso,
após o período de seis meses da realização das provas. O sistema necessita saber
qual é este local.
Lembre-se de utilizar os artigos adequados na resposta a esta pergunta.
Exemplo de resposta: A prefeitura municipal de São José dos Pinhais."))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item resultado) (tipo variavel)))

)

;-----------------------------------------------------------------

;P E R G U N T A S

;;Regra backward-chaining para a colocação de perguntas na memória de trabalho
(defrule putPerguntasResultado

   ;testa se colocou as variáveis
   (MAIN::ItemSetOk (item resultado) (tipo variavel))

        (MAIN::Pergunta (id ch_DataDivulgacaoDesempenho) (tipo data)
```

```
        (texto "Entre com a data limite para a divulgação do desempenho dos
candidatos."))

        (MAIN::Pergunta (id ch_HomologadorResultadoFinal) (tipo frase)
        (texto "A quem caberá homologar o resultado final do concurso?"))

        (MAIN::Pergunta (id ch_LocalInformacoesSeisMeses) (tipo fraseComArtigo)
        (texto "Entre com o local no qual o candidato poderá obter informações
referentes
ao concurso após 6 meses da realização das provas."))

        =>

        ;fato indicando sucesso na operacao
   (assert (MAIN::ItemSetOk (item resultado) (tipo pergunta)))


)

;-----------------------------------------------------------------

;L A Y O U T  D O  T E X T O

(defrule putLayoutResultado

   ;testa se colocou as perguntas
   (MAIN::ItemSetOk (item resultado) (tipo pergunta))

        (MAIN::Paragrafo (texto O resultado oficial do Concurso será divulgado nos
locais
definidos no Item 1.3, em duas listas, contendo a primeira a pontuação de todos os
candidatos,
inclusive, caso houver, a dos candidatos com deficiência e a segunda somente a
pontuação desses últimos.)
(item resultado) (integridade ok) (negrito 0) (subItem 1) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Serão emitidas duas listagens por cargo, sendo uma
em ordem alfabética e
outra classificatória geral.)
(item resultado) (integridade ok) (negrito 0) (subItem 2) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Somente será considerada oficial e válida a lista dos
classificados,
após publicada  nos locais definidos no item 1.3 .)
(item resultado) (integridade ok) (negrito 0) (subItem 3) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto Toda divulgação por outros meios será considera tão
somente como auxiliar
para os interessados, não sendo reconhecido nessa divulgação qualquer caráter
oficial.)
```

```
(item resultado) (integridade ok) (negrito 0) (subItem 4) (processado ok) (topico 0))

        (MAIN::Paragrafo (texto O resultado e o desempenho de cada candidato
serão divulgados até o
dia ch_DataDivulgacaoDesempenho .)
(item resultado) (integridade ok) (negrito 0) (subItem 5) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Caberá ao ch_HomologadorResultadoFinal , ou
pessoa por ele delegada,
homologar o resultado final do Concurso Público.)
(item resultado) (integridade ok) (negrito 0) (subItem 6) (processado nil) (topico 0))

        (MAIN::Paragrafo (texto Os candidatos que, após decorridos 6 meses da
realização das provas,
ainda desejarem alguma informação sobre o resultado do concurso devem entrar em
contato
com ch_LocalInformacoesSeisMeses .)
(item resultado) (integridade ok) (negrito 0) (subItem 7) (processado nil) (topico 0))


        =>

        ;fato indicando sucesso na operacao
        (assert (MAIN::ItemSetOk (item resultado) (tipo layout)))

)

;-----------------------------------------------------------------

;H I P Ó T E S E S

;coloca as hipóteses de finalização do edital
(defrule hipotesesResultado

        ;trigger
        (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item resultado))

        ;testa a hipótese dos parágrafos estarem todos processados
        (MAIN::ParagrafosProcessados (item resultado) (status ok))

        ;testa a hipótese de armazenar os parágrafos do item
   (MAIN::ItemArmazenado (item resultado) (status ok))


   =>

                (assert (MAIN::LinkQualificado (id hipotesesOk) (item resultado)))

)
```

```
;-----------------------------------------------------------------

;R E G R A S

;R E G R A S  P A R A  D I S P A R A R  O  Q U E S T I O N Á R I O

;Primeira busca
(defrule iniciaResultado
        ;testa hipótese
        (MAIN::Questionario (item resultado) (status carregado))
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_DataDivulgacaoDesempenho) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::Link homologadorResultadoFinal))

)

(defrule homologadorResultadoFinal
        ;trigger para entrar
        ?p <- (MAIN::Link homologadorResultadoFinal)
        ;buscar a resposta na memória
   (MAIN::Resposta (id ch_HomologadorResultadoFinal) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;caminho a seguir
        (assert (MAIN::Link localInformacoesSeisMeses))

        (retract ?p) ;elimina fato desnecessário

)

(defrule localInformacoesSeisMeses
        ;trigger para entrar
        ?p <- (MAIN::Link localInformacoesSeisMeses)
   ;busca a resposta na memória
   (MAIN::Resposta (id ch_LocalInformacoesSeisMeses) (texto ?resposta)) ;hipótese
backward-chaining

        =>

        ;trigger para prosseguir
        (assert (MAIN::LinkQualificado (id questionarioDisparado) (item resultado)))

        ;fato tornou-se desnecessário
```

```
        (retract ?p)

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  M O N T A R  O  E D I T A L

;regra inicial
(defrule iniciaMontagemResultado

   ;terminou o questionário
        (MAIN::LinkQualificado (id questionarioDisparado) (item resultado))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item resultado) (status definido)))

   ;busca a resposta
   (explicit (MAIN::Resposta (id ch_DataDivulgacaoDesempenho) (texto ?resposta)))

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item resultado) (subItem 5))) ;o parágrafo que será
processado

   =>

         ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_DataDivulgacaoDesempenho))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_HomologadorResultadoFinal) (item
resultado)))

)

;processar a variável ch_HomologadorResultadoFinal
(defrule processarChHomologadorResultadoFinal

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_HomologadorResultadoFinal) (item
resultado))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item resultado) (status definido)))
```

```
        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_HomologadorResultadoFinal) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item resultado) (subItem 6))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
        (bind $?newTexto (processar ?t (create$ ?resposta)
ch_HomologadorResultadoFinal))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id ch_LocalInformacoesSeisMeses) (item
resultado)))


        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;processar a variável ch_LocalInformacoesSeisMeses
(defrule processarChLocalInformacoesSeisMeses

   ;trigger
   ?trigger <- (MAIN::LinkQualificado (id ch_LocalInformacoesSeisMeses) (item
resultado))

   ;questionário definido (todas respostas estão definidas)
   (explicit (MAIN::Questionario (item resultado) (status definido)))

        ;busca a resposta na memória
   (explicit (MAIN::Resposta (id ch_LocalInformacoesSeisMeses) (texto
?resposta)));hipótese backward-chaining

   ;processa somente parágrafos que ainda não foram processados
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item resultado) (subItem 7))) ;o parágrafo que será
processado

   =>

        ;processar o parágrafo
```

```
        (bind $?newTexto (processar ?t (create$ (iniciaMinuscula ?resposta))
ch_LocalInformacoesSeisMeses))
        (modify ?p (texto ?newTexto))


        ;dispara trigger
        (assert (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item
resultado)))

        ;eliminando o fato desnecessario
        (retract ?trigger)

)

;-----------------------------------------------------------------

;R E G R A S  P A R A  V A L I D A R  O  S T A T U S  D O S  P A R Á G R A F O S

;regra para alterar o status dos parágrafos para ok
(defrule iniciaFinalizaResultado

        ;trigger
        (MAIN::LinkQualificado (id fimProcessamentoParagrafos) (item resultado))


    ;parágrafo que terá o status alterado
    ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
       (processado nil) (item resultado) (subItem 5)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoSeis) (item resultado)))

)

;regra para alterar o status dos parágrafos para ok
(defrule finalizaResultadoSeis

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoSeis) (item resultado))
```

```
   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item resultado) (subItem 6)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarParagrafoSete) (item resultado)))

)


;regra para alterar o status dos parágrafos para ok
(defrule finalizaResultado

        ;trigger
        (MAIN::LinkQualificado (id testarParagrafoSete) (item resultado))


   ;parágrafo que terá o status alterado
   ?p <- (explicit (MAIN::Paragrafo (texto $?t) (negrito ?negrito) (integridade ok)
      (processado nil) (item resultado) (subItem 7)))

        =>

        ;função que testa a existência de ch
        (bind ?resultado (paragrafoProcessado ?t))

        ;atribui o retorno do parágrafo ao slot processado
        (modify ?p (processado ?resultado))



        ;dispara trigger
        (assert (MAIN::LinkQualificado (id testarProcessamentoParagrafos) (item
resultado)))

)

;----------------------F I M ----------------------------------------
```

MEMÓRIA DE TRABALHO

(MAIN::initial-fact)
(MAIN::need-GravarEdital (status ok))
(MAIN::edital (class <Java-Object:java.lang.Class>) (numItens 14) (numParagrafos 40) (numTopicos 40) (OBJECT <Java-Object:Edital>))
(MAIN::Estado fatosCarregados)
(MAIN::edital (class <Java-Object:java.lang.Class>) (numItens 14) (numParagrafos 40) (numTopicos 40) (OBJECT <Java-Object:Edital>))
(MAIN::Link podeGravarMemoria)
(MAIN::Estado modeloOk)
(MAIN::need-MetaVariavel (id ch_AbrangenciaEdital) (descricao "O sistema necessita saber qual é a abrangência do concurso, se é municipal,
estadual ou federal. O sistema utilizará a resposta para chegar à algumas conclusões.
Por exemplo, concursos com abrangência federal ou estadual deverão possuir vagas reservadas
para afro-descendentes e deficientes físicos. Além disto a abrangência do edital irá definir
o tipo da lei que irá regulamentar o provimento efetivo de cargos"))
(MAIN::MetaVariavel (id ch_AbrangenciaEdital) (descricao "O sistema necessita saber qual é a abrangência do concurso, se é municipal,
estadual ou federal. O sistema utilizará a resposta para chegar à algumas conclusões.
Por exemplo, concursos com abrangência federal ou estadual deverão possuir vagas reservadas
para afro-descendentes e deficientes físicos. Além disto a abrangência do edital irá definir
o tipo da lei que irá regulamentar o provimento efetivo de cargos"))
(MAIN::need-MetaVariavel (id ch_TestaIntroducao) (descricao "O sistema necessita saber se item Introdução será necessário no edital.
A resposta deve ser fornecida na forma s ou n. Exemplo de resposta: s"))
(MAIN::MetaVariavel (id ch_TestaIntroducao) (descricao "O sistema necessita saber se item Introdução será necessário no edital.
A resposta deve ser fornecida na forma s ou n. Exemplo de resposta: s"))
(MAIN::need-MetaVariavel (id ch_TestaDispoPreliminares) (descricao "O sistema necessita saber se você deseja acrescentar o item
Disposições Preliminares ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaDispoPreliminares) (descricao "O sistema necessita saber se você deseja acrescentar o item
Disposições Preliminares ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaAfroDescendentes) (descricao "O sistema necessita saber se o concurso abrirá vagas para afro-descendentes.
Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."))

(MAIN::MetaVariavel (id ch_TestaAfroDescendentes) (descricao "O sistema necessita saber se o concurso abrirá vagas para afro-descendentes.
Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaDeficientesFisicos) (descricao "O sistema necessita saber se o concurso abrirá vagas para pessoas
com deficiência física. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaDeficientesFisicos) (descricao "O sistema necessita saber se o concurso abrirá vagas para pessoas
com deficiência física. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaInvestidura) (descricao "O sistema necessita saber se você deseja acrescentar o item
Investidura ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaInvestidura) (descricao "O sistema necessita saber se você deseja acrescentar o item
Investidura ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaInscricao) (descricao "O sistema necessita saber se você deseja acrescentar o item
Inscrição ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaInscricao) (descricao "O sistema necessita saber se você deseja acrescentar o item
Inscrição ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaProvas) (descricao "O sistema necessita saber se você deseja acrescentar o item
Provas ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaProvas) (descricao "O sistema necessita saber se você deseja acrescentar o item
Provas ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_TestaRecursos) (descricao "O sistema necessita saber se você deseja acrescentar o item
Recursos ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::MetaVariavel (id ch_TestaRecursos) (descricao "O sistema necessita saber se você deseja acrescentar o item
Recursos ao edital. Em caso positivo o sistema fornecerá posteriormente
um questionário específico para este item."))
(MAIN::need-MetaVariavel (id ch_OutroSiteDivulgacaoResultado) (descricao "O sistema necessita saber se além do site do núcleo de concursos, o resultado
do concurso será divulgado em outra pagina de Internet. Exemplo: s para sim, n para não."))
(MAIN::need-MetaVariavel (id ch_OutroSiteDivulgacaoResultado) (descricao "O sistema necessita saber se além do site do núcleo de concursos, o resultado

do concurso será divulgado em outra pagina de Internet. Exemplo: (s) para sim, (n) para não."))
(MAIN::MetaVariavel (id ch_OutroSiteDivulgacaoResultado) (descricao "O sistema necessita saber se além do site do núcleo de concursos, o resultado
do concurso será divulgado em outra pagina de Internet. Exemplo: (s) para sim, (n) para não."))
(MAIN::need-MetaVariavel (id ch_SiteDivulgacaoResultado) (descricao "O sistema necessita saber qual é o endereço eletrônico que será divulgado o resultado, além do site do Núcleo de Concursos. Exemplo: www.sanepar.com.br ."))
(MAIN::MetaVariavel (id ch_SiteDivulgacaoResultado) (descricao "O sistema necessita saber qual é o endereço eletrônico que será divulgado o resultado, além do site do Núcleo de Concursos. Exemplo: www.sanepar.com.br ."))
(MAIN::need-MetaVariavel (id ch_LocalDivulgacaoResultado) (descricao "O sistema necessita saber qual é local de divulgação do resultado.
Exemplo: Órgão de Imprensa do Município."))
(MAIN::MetaVariavel (id ch_LocalDivulgacaoResultado) (descricao "O sistema necessita saber qual é local de divulgação do resultado.
Exemplo: Órgão de Imprensa do Município."))
(MAIN::MetaSetOk (tipo metaVariavel))
(MAIN::need-Pergunta (texto "O item Introdução é necessário? (s ou n)") (tipo escolha) (id ch_TestaIntroducao) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item Introdução é necessário? (s ou n)") (tipo escolha) (id ch_TestaIntroducao) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "O item Disposições Preliminares é necessário?") (tipo escolha) (id ch_TestaDispoPreliminares) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item Disposições Preliminares é necessário?") (tipo escolha) (id ch_TestaDispoPreliminares) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "Há vagas para afro-descendentes?") (tipo escolha) (id ch_TestaAfroDescendentes) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "Há vagas para afro-descendentes?") (tipo escolha) (id ch_TestaAfroDescendentes) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "Há vagas para deficientes físicos?") (tipo escolha) (id ch_TestaDeficientesFisicos) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "Há vagas para deficientes físicos?") (tipo escolha) (id ch_TestaDeficientesFisicos) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "O item DAS CONDIÇÕES PARA INVESTIDURA NO CARGO é necessário?") (tipo escolha) (id ch_TestaInvestidura) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item DAS CONDIÇÕES PARA INVESTIDURA NO CARGO é necessário?") (tipo escolha) (id ch_TestaInvestidura) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "O item DA INSCRIÇÃO é necessário?") (tipo escolha) (id ch_TestaInscricao) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item DA INSCRIÇÃO é necessário?") (tipo escolha) (id ch_TestaInscricao) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "O item DAS PROVAS é necessário?") (tipo escolha) (id ch_TestaProvas) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item DAS PROVAS é necessário?") (tipo escolha) (id ch_TestaProvas) (teste s n -1 -2 -3 -4 -5 -6))

(MAIN::need-Pergunta (texto "O item DOS RECURSOS é necessário?") (tipo escolha) (id ch_TestaRecursos) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "O item DOS RECURSOS é necessário?") (tipo escolha) (id ch_TestaRecursos) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::need-Pergunta (texto "Entre com o tipo do edital. (1 – Municipal, 2 – Estadual, 3 – Federal)") (tipo escolha) (id ch_AbrangenciaEdital) (teste 1 2 3 -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "Entre com o tipo do edital. (1 – Municipal, 2 – Estadual, 3 – Federal)") (tipo escolha) (id ch_AbrangenciaEdital) (teste 1 2 3 -1 -2 -3 -4 -5 -6))
(MAIN::MetaSetOk (tipo pergunta))
(MAIN::need-Resposta (id ch_AbrangenciaEdital) (texto nil) (status nil))
(MAIN::Resposta (id ch_AbrangenciaEdital) (texto 1) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaRecursos) (texto s) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaProvas) (texto s) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaDispoPreliminares) (texto s) (status respostaDefinida))
(MAIN::need-Pergunta (texto "Entre com endereço eletrônico opcional para a divulgação do resultado.") (tipo webSite) (id ch_SiteDivulgacaoResultado) (teste))
(MAIN::Resposta (id ch_TestaIntroducao) (texto s) (status respostaDefinida))
(MAIN::Link metaRulesOk)
(MAIN::Resposta (id ch_TestaInvestidura) (texto s) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaInscricao) (texto n) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaDeficientesFisicos) (texto s) (status respostaDefinida))
(MAIN::Resposta (id ch_TestaAfroDescendentes) (texto s) (status respostaDefinida))
(MAIN::Link metaPerguntasDefinidas)
(MAIN::need-Item (nome "") (negrito 0) (sequencia 0) (status nil) (id introducao))
(MAIN::Item (nome "") (negrito 0) (sequencia 0) (status nil) (id introducao))
(MAIN::ItemSetOk (tipo item) (item introducao))
(MAIN::need-Variavel (id ch_Publicador) (item introducao) (descricao "O Concurso torna-se aberto ao público mediante as atribuições legais de alguma pessoa física ou jurídica. A resposta a esta pergunta irá indicar ao sistema quem é esta pessoa. Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: A Comissão Executiva do Concurso Público"))
(MAIN::need-Questionario (item introducao) (status carregado))
(MAIN::Item (nome "DAS DISPOSIÇÕES PRELIMINARES") (negrito 1) (sequencia 1) (status nil) (id dispoPreliminares))
(MAIN::ItemSetOk (tipo item) (item dispoPreliminares))
(MAIN::need-Variavel (id ch_Apoiador) (item dispoPreliminares) (descricao "O sistema necessita saber quem irá apoiar o Núcleo de Concursos para a realização do concurso. Você deve iniciar a resposta com o artigo adequado Exemplo: A Secretaria Municipal da Administração de São José dos Pinhais"))
(MAIN::need-Questionario (item dispoPreliminares) (status carregado))
(MAIN::Item (nome "DAS VAGAS DESTINADAS A PESSOAS COM DEFICIÊNCIA") (negrito 1) (sequencia 3) (status nil) (id deficientesFisicos))
(MAIN::ItemSetOk (tipo item) (item deficientesFisicos))
(MAIN::need-Variavel (id ch_IndicaMultiprofissional) (item deficientesFisicos) (descricao "A resposta a esta pergunta irá informar ao sistema quem é a pessoa responsável pela indicação da Equipe Multiprofissional. Esta última corresponde a uma equipe de profissionais que irá fornecer um parecer quanto ao atendimento das condições diferenciadas solicitadas pelo candidato portador deficiência física."))
(MAIN::need-Questionario (item deficientesFisicos) (status carregado))

(MAIN::Item (nome "DAS VAGAS DESTINADAS A AFRO-DESCENDENTES") (negrito 1) (sequencia 2) (status nil) (id afro))
(MAIN::ItemSetOk (tipo item) (item afro))
(MAIN::need-Variavel (id ch_PrazoAfro) (item afro) (descricao "O sistema necessita saber quantos dias úteis o candidato tem para se
apresentar para a perícia para a verificação dos traços fenotípicos que o caracterizam na sociedade como pertencente ao grupo racial negro e à entrevista que
decidirá se o candidato atende aos requisitos. Exemplo de resposta: 5"))
(MAIN::need-Questionario (item afro) (status carregado))
(MAIN::Item (nome "DAS CONDIÇÕES PARA INVESTIDURA NO CARGO") (negrito 1) (sequencia 5) (status nil) (id investidura))
(MAIN::ItemSetOk (tipo item) (item investidura))
(MAIN::need-Questionario (item investidura) (status carregado))
(MAIN::Item (nome "DAS PROVAS") (negrito 1) (sequencia 7) (status nil) (id provas))
(MAIN::ItemSetOk (tipo item) (item provas))
(MAIN::need-Questionario (item provas) (status carregado))
(MAIN::Item (nome "DOS RECURSOS") (negrito 1) (sequencia 8) (status nil) (id recursos))
(MAIN::ItemSetOk (tipo item) (item recursos))
(MAIN::need-Questionario (item recursos) (status carregado))
(MAIN::Estado itensDefinidosECarregados)
(MAIN::Variavel (id ch_QuantasHorasGabProvisorio) (item recursos) (descricao "O sistema necessita saber após quantas horas da realização das provas
o gabarito provisório será divulgado. Exemplo de resposta: 24."))
(MAIN::Variavel (id ch_HoraEntregaQuestionamentos) (item recursos) (descricao "O sistema necessita saber qual é a hora limite para a entrega do
formulário específico. Tal formulário é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões."))
(MAIN::Variavel (id ch_DataEntregaQuestionamentos) (item recursos) (descricao "O sistema necessita saber qual é a data final para a entrega do
formulário específico. Tal formulário é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões."))
(MAIN::Variavel (id ch_HoraFinalProtocolo) (item recursos) (descricao "O sistema necessita saber qual é a hora limite para protocolar o
recurso. Tal procedimento é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões."))
(MAIN::Variavel (id ch_DataFinalProtocolo) (item recursos) (descricao "O sistema necessita saber qual é a data limite para protocolar o recurso.
Tal procedimento é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões."))
(MAIN::Variavel (id ch_DataGabaritoOficial) (item recursos) (descricao "O sistema necessita saber qual é a data de divulgação do gabarito
oficial. Sobre este gabarito não será aceito nenhum tipo de recurso."))
(MAIN::ItemSetOk (tipo variavel) (item recursos))
(MAIN::Pergunta (texto "Quantas horas após a realização das provas será divulgado o gabarito provisório?") (tipo numero) (id ch_QuantasHorasGabProvisorio) (teste))
(MAIN::Pergunta (texto "Entre com a hora final para a entrega dos questionamentos. (hh:mm)") (tipo hora) (id ch_HoraEntregaQuestionamentos) (teste))

(MAIN::Pergunta (texto "Entre com a data final para a entrega dos questionamentos. (dd-mm-aa)") (tipo data) (id ch_DataEntregaQuestionamentos) (teste))
(MAIN::Pergunta (texto "Entre com a hora final para protocolar recurso. (hh:mm)") (tipo hora) (id ch_HoraFinalProtocolo) (teste))
(MAIN::Pergunta (texto "Entre com a data final para protocolar recurso. (dd-mm-aa)") (tipo data) (id ch_DataFinalProtocolo) (teste))
(MAIN::Pergunta (texto "Entre com a data de divulgação do gabarito oficial. (dd-mm-aa)") (tipo data) (id ch_DataGabaritoOficial) (teste))
(MAIN::ItemSetOk (tipo pergunta) (item recursos))
(MAIN::Paragrafo (texto Até ch_QuantasHorasGabProvisorio horas após a realização das provas, o NC publicará o gabarito provisório das questões objetivas no site www.nc.ufpr.br.) (topico 0) (item recursos) (subItem 1) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Serão aceitos questionamentos sobre o gabarito ou à formulação das questões desde que estejam em conformidade com o disposto nos itens 8.3, 8.4, 8.5, 8.6, 8.7, 8.8.) (topico 0) (item recursos) (subItem 2) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Os questionamentos devem estar devidamente fundamentados e apresentados em formulário específico que estará disponível no site www.nc.ufpr.br, até as ch_HoraEntregaQuestionamentos do dia ch_DataEntregaQuestionamentos .) (topico 0) (item recursos) (subItem 3) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Os candidatos devem imprimir o formulário específico e seguir rigorosamente as instruções contidas no site www.nc.ufpr.br.) (topico 0) (item recursos) (subItem 4) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Cada questão ou item deve ser apresentado em um formulário separado.) (topico 0) (item recursos) (subItem 5) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O recurso deverá ser protocolado na Secretaria do Núcleo de Concurso ou enviado apenas 1 via por Sedex ao Núcleo de Concursos – UFPR - Setor de Ciências Agrárias - Rua dos Funcionáros nº 1540 - CEP 80035-050, Curitiba – PR, desde que postado até as ch_HoraFinalProtocolo do dia ch_DataFinalProtocolo .) (topico 0) (item recursos) (subItem 6) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Serão desconsiderados pelo NC questionamentos que não estiverem redigidos no formulário específico, não protocolados, fora do prazo ou que não estiverem devidamente fundamentados, bem como encaminhados de forma diferente ao estabelecido nos itens anteriores.) (topico 0) (item recursos) (subItem 7) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Serão desconsiderados pelo NC questionamentos relativos ao preenchimento do cartão-resposta.) (topico 0) (item recursos) (subItem 8) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O recurso será apreciado por uma banca examinadora, designada pelo Núcleo de Concurso, que emitirá decisão fundamentada, a qual será colocada à disposição do requerente na Secretária no Nucleo de Concursos.) (topico 0) (item recursos) (subItem 9) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Uma vez julgados os recurso apresentados, será emitido novo gabarito.) (topico 0) (item recursos) (subItem 10) (negrito 0) (integridade ok) (processado ok) (armazenado nil))

(MAIN::Paragrafo (texto A Banca Examinadora constitui última instância para recurso, sendo soberana em suas decisões, razão pela qual não caberão recursos adicionais.) (topico 0) (item recursos) (subItem 11) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Os pontos relativos às questões porventura anuladas serão atribuídos a todos os candidatos que fizerem a prova. Se houver alteração, por força de impugnações do gabarito oficial provisório, de item integrante de provas, essa alteração valerá para todos os candidatos, independentemente de terem recorrido.) (topico 0) (item recursos) (subItem 12) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Com exceção dos recursos previstos nos itens anteriores, não se concederá revisão de provas, segunda chamada, vistas, ou recontagem de pontos em qualquer das provas, devido às características do Concurso.) (topico 0) (item recursos) (subItem 13) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O gabarito oficial das provas objetivas sera divulgado no site www.nc.ufpr.br até o dia ch_DataGabaritoOficial .) (topico 0) (item recursos) (subItem 14) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::ItemSetOk (tipo layout) (item recursos))
(MAIN::LoadQuestionario recursos)
(MAIN::Questionario (item recursos) (status carregado))
(MAIN::Resposta (id ch_QuantasHorasGabProvisorio) (texto 24) (status respostaDefinida))
(MAIN::Resposta (id ch_HoraEntregaQuestionamentos) (texto -1) (status responderDepois))
(MAIN::Estado haIndefinidas)
(MAIN::Resposta (id ch_DataEntregaQuestionamentos) (texto 24-04-06) (status respostaDefinida))
(MAIN::Resposta (id ch_HoraFinalProtocolo) (texto 17:00) (status respostaDefinida))
(MAIN::Resposta (id ch_DataFinalProtocolo) (texto 24-04-06) (status respostaDefinida))
(MAIN::Resposta (id ch_DataGabaritoOficial) (texto 24-04-06) (status respostaDefinida))
(MAIN::LinkQualificado (id questionarioDisparado) (item recursos))
(MAIN::Variavel (id ch_DiaDeProvas) (item provas) (descricao "O sistema necessita saber qual é o dia para a realização das provas. A
resposta necessita ser no formato dia-mês-ano. Exemplo: 20-08-06."))
(MAIN::Variavel (id ch_HoraInicioDeProvas) (item provas) (descricao "O sistema necessita saber em que horário as provas irão iniciar. A
resposta necessita ser no formato hora:minuto. Exemplo: 14:00"))
(MAIN::Variavel (id ch_HoraFimDeProvas) (item provas) (descricao "O sistema necessita saber qual é o horário limite para que o candidato
termine a realização das provas. A resposta necessita ser no formato hora:minuto. Exemplo: 18:00"))
(MAIN::Variavel (id ch_CidadeProvas) (item provas) (descricao "O sistema necessita saber em que cidade as provas discursivas e
objetivas serão realizadas. A resposta necessita ser no formato Cidade-UF. Exemplo: Curitiba-PR"))
(MAIN::Variavel (id ch_HoraFechaPortoes) (item provas) (descricao "O sistema necessita saber qual é o horário limite em que as portas de

acesso aos prédios onde serão realizadas as provas serão fechadas. A resposta necessita ser no formato hora:minuto. Exemplo: 13:30"))
(MAIN::ItemSetOk (tipo variavel) (item provas))
(MAIN::Pergunta (texto "Entre com a data para a realização das provas. (dd-mm-aa)") (tipo data) (id ch_DiaDeProvas) (teste))
(MAIN::Pergunta (texto "Entre com o horário do início da realização das provas. (hh:mm)") (tipo hora) (id ch_HoraInicioDeProvas) (teste))
(MAIN::Pergunta (texto "Entre com o horário de término da realização das provas. (hh:mm)") (tipo hora) (id ch_HoraFimDeProvas) (teste))
(MAIN::Pergunta (texto "Entre com a cidade onde as provas serão realizadas. (Cidade-UF)") (tipo frase) (id ch_CidadeProvas) (teste))
(MAIN::Pergunta (texto "Entre com o horário de fechamento dos portões. (hh:mm)") (tipo hora) (id ch_HoraFechaPortoes) (teste))
(MAIN::ItemSetOk (tipo pergunta) (item provas))
(MAIN::Paragrafo (texto As provas serão realizadas no dia ch_DiaDeProvas das ch_HoraInicioDeProvas às ch_HoraFimDeProvas , em ch_CidadeProvas .) (topico 0) (item provas) (subItem 1) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto O local de prova do candidato constará no respectivo comprovante de ensalamento, disponível no site www.nc.ufpr.br.) (topico 0) (item provas) (subItem 2) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto As provas objetivas serão constituídas de questões de múltipla escolha, com cinco alternativas -a, b, c, d, e,- das quais apenas uma será correta.) (topico 0) (item provas) (subItem 3) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto As respostas às questões objetivas deverão ser transcritas para o cartão-resposta com caneta esferográfica com tinta preta, devendo o candidato assinalar uma única resposta para cada questão.) (topico 0) (item provas) (subItem 4) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não serão consideradas questões não assinaladas ou que contenham mais de uma resposta, emenda ou rasura.) (topico 0) (item provas) (subItem 5) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O candidato assume plena e total responsabilidade pelo correto preenchimento do cartão-resposta e sua integridade. Não haverá substituição desse cartão, a não ser em caso de defeito em sua impressão.) (topico 0) (item provas) (subItem 6) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Nas provas objetivas haverá para cada candidato, um caderno de prova e um cartão-resposta identificado e numerado adequadamente. A correção dessas provas será feita por meio de leitura óptica do cartão-resposta e, quando necessário, onde couber, com a verificação dos cálculos constantes no caderno de prova, os quais deverão atestar a consistência da resposta encontrada para cada questão.) (topico 0) (item provas) (subItem 7) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Com o objetivo de garantir a lisura e a idoneidade do Concurso, o candidato deverá autenticar a ficha de identificação por meio de suas impressões digitais.) (topico 0) (item provas) (subItem 8) (negrito 0) (integridade ok) (processado ok) (armazenado nil))

(MAIN::Paragrafo (texto A ausência do candidato, por qualquer motivo, inclusive doença e atraso, implicará na sua eliminação do concurso.) (topico 0) (item provas) (subItem 9) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Para ingresso na sala de provas, o candidato deverá apresentar o comprovante de ensalamento, juntamente com o original do documento oficial de identidade.) (topico 0) (item provas) (subItem 10) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto São documentos oficiais de identidade: Carteiras e/ou Cédulas de Identidade expedidas pelas Secretarias de Segurança, pelas Forças Armadas, pela Polícia Militar, pelo Ministério das Relações Exteriores, Cédulas de Identidade fornecidas por Ordens ou Conselhos de Classe que por lei federal valem como documento de identidade, como, por exemplo, as do CREA e OAB, etc., a Carteira de Trabalho e Previdência Social, bem como a Carteira Nacional de Habilitação, com fotografia, na forma da Lei n. 9.053/97.) (topico 0) (item provas) (subItem 11) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Os documentos para ingresso na sala de prova, especificados no subitem anterior, devem estar em perfeitas condições e devem permitir, com clareza, a identificação dos candidatos.) (topico 0) (item provas) (subItem 12) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Em caso de perda ou roubo de documentos, o candidato será admitido para realizar as provas, desde que tenha se apresentado no local que lhe foi designado e que seja possível fazer a verificação de seus dados junto ao Núcleo de Concursos antes da hora marcada para início das provas.) (topico 0) (item provas) (subItem 13) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Nas salas de prova, os candidatos não poderão manter em seu poder relógios, armas e aparelhos eletrônicos: BIP, telefone celular, calculadora, agenda eletrônica, etc, devendo seguir as orientações fornecidas pelos aplicadores antes do início da prova. O candidato que estiver portando qualquer desses instrumentos durante a realização da prova será eliminado do Concurso.) (topico 0) (item provas) (subItem 14) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Excepcionalmente e a critério médico devidamente comprovado, o candidato que estiver impossibilitado de realizar a prova em sala de prova, fará a mesma em local especialmente designado pelo Núcleo de Concurso – NC.) (topico 0) (item provas) (subItem 15) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto A candidata que tiver necessidade de amamentar durante a realização das provas deverá levar um acompanhante, que ficará responsável pela guarda da criança. A candidata que não levar acompanhante não realizará a prova.) (topico 0) (item provas) (subItem 16) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Os casos citados nos itens 7.15 e 7.16 como também outro caso de emergência devem ser comunicados ao Núcleo de Concurso pelo fone/fax - 0xx41- 3313-8832.) (topico 0) (item provas) (subItem 17) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não será permitido o ingresso ou a permanência de pessoas estranhas ao Concurso Público no local de aplicação das provas.) (topico 0) (item provas) (subItem 18) (negrito 0) (integridade ok) (processado ok) (armazenado nil))

(MAIN::Paragrafo (texto As portas de acesso aos prédios onde serão realizadas as provas serão fechadas às ch_HoraFechaPortoes . Os relógios da Comissão Organizadora do Concurso serão acertados pelo Serviço Hora Certa Brasil Telecom – fone 130 .) (topico 0) (item provas) (subItem 19) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Poderá, a critério do Núcleo de Concurso, ocorrer a prorrogação no fechamento das portas de acesso de um ou mais locais onde serão realizadas as provas, em função de fatores externos.) (topico 0) (item provas) (subItem 20) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não será permitido ao candidato, durante a realização das provas, ausentar-se do recinto, a não ser em caso especiais e acompanhando de um membro componente da equipe de aplicação do Concurso.) (topico 0) (item provas) (subItem 21) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não será permitido, durante a realização da prova, o uso de bonés, gorros, chapéus, etc.) (topico 0) (item provas) (subItem 22) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não será permitido ao candidato alimentar-se dentro da sala de aula, devendo aqueles que necessitar fazê-lo, por motivos médicos, solicitar ao aplicador de provas e encaminhamento à sala de inspeção.) (topico 0) (item provas) (subItem 23) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não será permitida, durante a realização da prova, comunicação entre os candidatos, nem o uso de calculadoras e/ou similares, livros, anotações, réguas de cálculo, impressos ou qualquer outro material de consulta.) (topico 0) (item provas) (subItem 24) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O candidato somente poderá retirar-se da sala após decorrida 1:30h -uma hora e trinta minutos- do início da prova, devendo entregar ao aplicador da prova, obrigatoriamente, o caderno de provas e o cartão-resposta devidamente assinalado.) (topico 0) (item provas) (subItem 25) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto O caderno de provas conterá um espaço designado para anotação das respostas das questões objetivas, que poderá ser recortado e levado pelo candidato para posterior conferência do gabarito.) (topico 0) (item provas) (subItem 26) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto É de responsabilidade exclusiva do candidato a identificação correta de seu local de realização das provas e o comparecimento no horário determinado.) (topico 0) (item provas) (subItem 27) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Constatada a qualquer tempo, por meio eletrônico, estatístico, visual ou grafológico, a utilização pelo candidato de procedimentos ilícitos, a prova será objeto de anulação e automaticamente o candidato será eliminado do Concurso, sem prejuízo das cominações legais civis e criminais deles decorrentes.) (topico 0) (item provas) (subItem 28) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Será eliminado o candidato que, durante a aplicação das provas, praticar atos contra as normas ou disciplinas.) (topico 0) (item provas) (subItem 29) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Os procedimentos e os critérios para correção das provas são de responsabilidade do NC.) (topico 0) (item provas) (subItem 30) (negrito 0) (integridade ok) (processado ok) (armazenado nil))

(MAIN::ItemSetOk (tipo layout) (item provas))
(MAIN::LoadQuestionario provas)
(MAIN::Questionario (item provas) (status carregado))
(MAIN::Resposta (id ch_DiaDeProvas) (texto -1) (status responderDepois))
(MAIN::Resposta (id ch_HoraInicioDeProvas) (texto -1) (status responderDepois))
(MAIN::Resposta (id ch_HoraFimDeProvas) (texto 17:00) (status respostaDefinida))
(MAIN::Resposta (id ch_CidadeProvas) (texto "curitiba") (status respostaDefinida))
(MAIN::Resposta (id ch_HoraFechaPortoes) (texto -1) (status responderDepois))
(MAIN::LinkQualificado (id questionarioDisparado) (item provas))
(MAIN::Variavel (id ch_TipoResolucaoDemissao) (item investidura) (descricao "Uma das exigências para que o candidato aprovado no concurso
público seja investido no cargo é que ele não tenha sido demitido em serviço público municipal, estadual ou federal. Existe uma resolução legal que trata deste assunto. O sistema necessita saber qual é o tipo desta resolução legal, se é decreto, lei ou portaria."))
(MAIN::Variavel (id ch_NumResolucaoDemissao) (item investidura) (descricao "A resposta a esta pergunta irá definir o número da resolução legal que
trata da demissão de Serviço Público. Exemplo: 525."))
(MAIN::Variavel (id ch_DataResolucaoDemissao) (item investidura) (descricao "A resposta a esta pergunta irá indicar ao sistema a data da resolução
legal que trata da demissão de Serviço Público. Esta deve ser fornecida no formato dia-mês-ano. Exemplo: 25-03-04"))
(MAIN::Variavel (id ch_ResolucaoDemissao) (item investidura) (descricao "Variável utilizada internamente pelo sistema, resultante do
processamento das variáveis ch_TipoResolucaoDemissao,
ch_NumResolucaoDemissao e ch_DataResolucaoDemissao. Será utilizada na sentenca ch_ResolucaoDemissao."))
(MAIN::Variavel (id ch_AnaliseFisicaMental) (item investidura) (descricao "O candidato aprovado no Concurso de que trata este edital deverá ter
aptidão física e mental compatíveis às funções que irá exercer. A resposta a esta pergunta irá indicar para o sistema quem será o órgão responsável pela verificação do atendimento a estas condições."))
(MAIN::Variavel (id ch_CarteiraHabilitacao) (item investidura) (descricao "Há casos em possuir carteira de habilitação é uma condição para que o
candidato aprovado no Concurso de que trata este edital possa ser investido no cargo. A resposta a esta pergunta irá indicar ao sistema se esta condição se aplica a este Concurso."))
(MAIN::ItemSetOk (tipo variavel) (item investidura))
(MAIN::Pergunta (texto "Entre com a data da resolução legal que trata da demissão de Serviço Público. (dd-mm-aa).") (tipo data) (id ch_DataResolucaoDemissao) (teste
))
(MAIN::Pergunta (texto "Entre com o número da resolução legal que trata da demissão de Serviço Público.") (tipo numero) (id ch_NumResolucaoDemissao) (teste
))
(MAIN::Pergunta (texto "Entre com o tipo da resolução legal que trata da demissão de Serviço
Público (1 Decreto, 2 Lei, 3 Portaria)") (tipo escolha) (id
ch_TipoResolucaoDemissao) (teste 1 2 3 -1 -2 -3 -4 -5 -6))
(MAIN::Pergunta (texto "Entre com o órgão responsável pela análise da aptidão física e mental do

candidato aprovado. (Inicie com o artigo adequado)") (tipo fraseComArtigo) (id ch_AnaliseFisicaMental) (teste))
(MAIN::Pergunta (texto "É necessário possuir carteira de habilitação?") (tipo escolha) (id ch_CarteiraHabilitacao) (teste s n -1 -2 -3 -4 -5 -6))
(MAIN::ItemSetOk (tipo pergunta) (item investidura))
(MAIN::Paragrafo (texto O candidato aprovado no Concurso de que trata este Edital, e convocado, será investido no cargo, se atendidas as seguintes exigências:) (topico 0) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Ter nacionalidade brasileira, e aos estrangeiros os requisitos na forma da Lei, conforme a Constituição Federal,) (topico 1) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Gozar dos direitos políticos,) (topico 2) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Estar em dia com as obrigações eleitorais e, se do sexo masculino, também com as militares,) (topico 3) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Comprovar por meio de Diploma ou Certificado o nível de escolaridade exigido para o exercício do cargo,) (topico 4) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Ter idade mínima de 18 anos,) (topico 5) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Não ter sido demitido do Serviço Público Estadual, Federal ou Municipal, ch_ResolucaoDemissao) (topico 6) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Ter aptidão física e mental compatíveis às suas funções, através de verificação feita ch_AnaliseFisicaMental) (topico 7) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado nil) (armazenado nil))
(MAIN::Paragrafo (texto Apresentar certidão negativa de antecedentes criminais,) (topico 8) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Apresentar declaração de bens e valores que constituam seu patrimônio e, se casado, a do cônjuge,) (topico 9) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Apresentar declaração quanto ao exercício ou não de outro cargo, emprego ou função pública e de compatibilidade de horário para o exercício de mais de um cargo quando acumulável,) (topico 10) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Declaração sobre percepção de proventos de aposentadoria, por conta do regime geral de previdência ou de qualquer outro regime próprio de previdência em âmbito federal, estadual ou municipal, decorrentes de cargos, emprego ou função pública, conforme artigo 40 da Constituição Federal e de serviço militar, conforme artigos 42 e 142 da Consituição Federal,) (topico 11) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto Apresentar comprovante de residência,) (topico 12) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))

(MAIN::Paragrafo (texto Apresentar outros documentos que se fizerem necessários à época da posse,) (topico 13) (item investidura) (subItem 1) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::Paragrafo (texto A falta de comprovação ou a constatação de falsidade de qualquer um dos requisitos especificados e daqueles que vierem a ser estabelecidos, impedirá a investidura do candidato ao cargo.) (topico 0) (item investidura) (subItem 2) (negrito 0) (integridade ok) (processado ok) (armazenado nil))
(MAIN::ItemSetOk (tipo layout) (item investidura))
(MAIN::LoadQuestionario investidura)
(MAIN::Questionario (item investidura) (status carregado))
(MAIN::Resposta (id ch_TipoResolucaoDemissao) (texto 3) (status respostaDefinida))
(MAIN::Resposta (id ch_NumResolucaoDemissao) (texto -1) (status responderDepois))
(MAIN::Resposta (id ch_DataResolucaoDemissao) (texto -3) (status salvarMemoria))

JAVADOC – Padrão Sun

Aplicativo

Class Aplicativo

```
java.lang.Object
  Aplicativo
```

```
public class Aplicativo
extends java.lang.Object
```

Aplicativo é classe que executa o Sistema Especialista EditalX usando a shell Jess.

Constructor Summary

Aplicativo()

Method Summary

static void	main(java.lang.String[] args) Método principal

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Aplicativo

```
public Aplicativo()
```

Method Detail

main

```
public static void main(java.lang.String[] args)
```

Método principal

Assembler

Class Assembler

```
java.lang.Object
  Assembler
```

```
public class Assembler
extends java.lang.Object
```

Classe responsável por gerar parágrafos IText a partir de uma matriz tridimensional de strings.

Constructor Summary

`Assembler()`	Construtor sem argumentos da classe Assembler.

Method Summary

void	`montaEdital(java.lang.String[][][] matriz)` Grava em documentos pdf e rtf cada string presente na matriz, em forma de parágrafos IText.
void	`setFonte(int num)` Configura o tipo da fonte que será utilizada na geração do parágrafo.
java.lang.String	`setTexto(java.lang.String texto, int item, int paragrafo, int topico)` Acrescenta o número do item, do parágrafo e tabulações ao String passado como argumento.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Assembler

```
public Assembler()
```

Construtor sem argumentos da classe Assembler.

Method Detail

setFonte

```
public void setFonte(int num)
```

Configura o tipo da fonte que será utilizada na geração do parágrafo. Valida se o argumento passado e 0 ou 1. O caso default é 0.

Parameters:

`num` - o tipo da fonte a ser utilizada: 0 - Times New Roman, sem negrito 1 - Times New Roman, negrito

---

montaEdital

```
public void montaEdital(java.lang.String[][][] matriz)
```

Grava em documentos pdf e rtf cada string presente na matriz, em forma de parágrafos IText.

Parameters:

`matriz` - referência para a matriz que contém o conteúdo do edital.

---

setTexto

```
public java.lang.String setTexto(java.lang.String texto,
                                 int item,
                                 int paragrafo,
                                 int topico)
```

Acrescenta o número do item, do parágrafo e tabulações ao String passado como argumento.

Parameters:

`texto` - o texto a ser manipulado

`item` - o número do item do texto a ser manipulado

`paragrafo` - o número do parágrafo do texto a ser manipulado

`topico` - o número do tópico do texto a ser manipulado

EDITAL

Package Class Use Tree Deprecated Index Help
PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class Edital

```
java.lang.Object
  Edital
```

```
public class Edital
extends java.lang.Object
```

Classe responsável por organizar o texto de um edital de concurso público em uma matriz tridimensional (i, j, k), sendo que os índices i correspondem aos aos itens, j aos parágrafos dos itens e k aos tópicos dos parágrafos.

Constructor Summary

`Edital(int itens, int paragrafos, int topicos)`
Construtor da classe Edital: número de itens, parágrafos e tópicos devem ser fornecidos.

Method Summary

int	`getNumItens()` Retorna o número máximo de itens que o edital pode possuir.
int	`getNumParagrafos()` Retorna o número máximo de parágrafos que cada item pode possuir.
int	`getNumTopicos()` Retorna o número máximo de tópicos que cada parágrafo pode possuir.
java.lang.String	`getTextoParagrafo(int l, int c)` Retorna o texto do parágrafo armazenado na posição indicada pelos argumentos
java.lang.String	`getTextoTopico(int i, int j, int k)` Retorna o texto do tópico armazenado na posição indicada pelos argumentos
void	`putTexto(int item, int paragrafo, int topico, java.lang.String texto)` Coloca texto na matriz.
void	`salvaEdital()` Chama o método do objeto assembler para gravar o edital em disco.
void	`setNumItens(int qde)` Configura o número máximo de itens que o edital pode possuir.
void	`setNumParagrafos(int qde)` Configura o número máximo de parágrafos que cada item pode

	possuir.
void	**setNumTopicos**(int qde) Configura o número máximo de tópicos que cada parágrafo pode possuir.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Edital

```
public Edital(int itens,
              int paragrafos,
              int topicos)
```

Construtor da classe Edital: número de itens, parágrafos e tópicos devem ser fornecidos.

Parameters:

itens - o número máximo de itens que o edital pode possuir

parágrafos - o número máximo de parágrafos em cada item

topicos - o número máximo de tópicos em cada parágrafo

Method Detail

setNumItens

```
public void setNumItens(int qde)
```

Configura o número máximo de itens que o edital pode possuir. Caso o parâmetro passado seja menor ou igual a zero, o valor adotado será quinze.

Parameters:

qde - o número de itens

---

setNumParagrafos

```
public void setNumParagrafos(int qde)
```

Configura o número máximo de parágrafos que cada item pode possuir. Caso o parâmetro passado seja menor ou igual a zero, o valor adotado será cinquenta.

Parameters:

qde - o número de parágrafos

---

setNumTopicos

```
public void setNumTopicos(int qde)
```

Configura o número máximo de tópicos que cada parágrafo pode possuir. Caso o parâmetro passado seja menor ou igual a zero, o valor adotado será cinquenta.

Parameters:

`qde` - o número de tópicos

---

getNumItens

```
public int getNumItens()
```

Retorna o número máximo de itens que o edital pode possuir.

Returns:

um `int` indicando o número de itens.

---

getNumParagrafos

```
public int getNumParagrafos()
```

Retorna o número máximo de parágrafos que cada item pode possuir.

Returns:

um `int` indicando o número de parágrafos.

---

getNumTopicos

```
public int getNumTopicos()
```

Retorna o número máximo de tópicos que cada parágrafo pode possuir.

Returns:

um `int` indicando o número de tópicos.

---

getTextoParagrafo

```
public java.lang.String getTextoParagrafo(int l,
                                          int c)
```

Retorna o texto do parágrafo armazenado na posição indicada pelos argumentos

Returns:

um `String` referente ao texto.

---

getTextoTopico

```
public java.lang.String getTextoTopico(int i,
                                       int j,
                                       int k)
```

Retorna o texto do tópico armazenado na posição indicada pelos argumentos

Returns:

um `String` referente ao texto.

---

putTexto

```
public void putTexto(int item,
                     int paragrafo,
                     int topico,
                     java.lang.String texto)
```

Coloca texto na matriz.

Parameters:

`item` - a posição do item na matriz

`paragrafo` - posição do parágrafo na matriz

`topico` - posição do tópico na matriz

`texto` - a ser colocado na matriz

---

salvaEdital

```
public void salvaEdital()
```

Chama o método do objeto assembler para gravar o edital em disco. O edital será gravado nos formatos PDF e RTF.

EDITALRECORDER

Class EditalRecorder

```
java.lang.Object
  EditalRecorder
```

```
public class EditalRecorder
extends java.lang.Object
```

Classe que implementa a gravação de documentos em formato RTF e PDF.

Constructor Summary

`EditalRecorder()` Construtor sem argumentos

Method Summary

void	`fechaEdital()` Fecha os documentos
void	`gravarEdital(java.lang.String texto, int fonte)` Grava a string passada como argumento em documentos pdf e rtf

Methods inherited from class java.lang.Object
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

EditalRecorder

```
public EditalRecorder()
```

Construtor sem argumentos

Method Detail

gravarEdital

```
public void gravarEdital(java.lang.String texto,
                         int fonte)
```

Grava a string passada como argumento em documentos pdf e rtf

Parameters:

`texto` - o texto a ser gravado

`fonte` - o estilo da fonte a ser utilizada no parágrafo 0 - Times New Roman, Tamanho 12, Normal, Cor Preta 1 - Times New Roman, Tamanho 12, Negrito, Cor Preta

---

fechaEdital

```
public void fechaEdital()
```

Fecha os documentos

ENGINE

Class Engine

```
java.lang.Object
  Engine
```

```
public class Engine
extends java.lang.Object
```

Classe que implementa o mecanismo de inferência utilizando JESS (Java Expert System Shell)

Constructor Summary

`Engine(java.awt.TextArea ta, java.awt.TextArea ta2)` Construtor da classe Engine: os adaptadores para output na GUI devem ser fornecidos	

Method Summary

`jess.awt.TextReader`	`getTr()` Retorna o adaptador do input stream
`void`	`rout(java.awt.TextArea ta, java.awt.TextArea ta2)` Método para direcionar o output stream e input stream para a GUI
`void`	`startup(java.lang.String arquivo)` Método para inicializar o sistema especialista

Methods inherited from class java.lang.Object
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

Engine

```
public Engine(java.awt.TextArea ta,
              java.awt.TextArea ta2)
```

Construtor da classe Engine: os adaptadores para output na GUI devem ser fornecidos

Parameters:

`ta` - o stream para adaptar output das perguntas para a GUI

`ta2` - o stream para adapatar o output das mensagens para a GUI

Method Detail

startup

```
public void startup(java.lang.String arquivo)
```

Método para inicializar o sistema especialista

Parameters:

`arquivo` - a localização do JESS script

---

rout

```
public void rout(java.awt.TextArea ta,
                 java.awt.TextArea ta2)
```

Método para direcionar o output stream e input stream para a GUI

Parameters:

`ta` - output para as perguntas

`ta2` - output para as mensagens

---

getTr

```
public jess.awt.TextReader getTr()
```

Retorna o adaptador do input stream

Returns:

um `TextReader` indicando o objeto input stream

TELAPRINCIPAL

Class TelaPrincipal

```
java.lang.Object
  java.awt.Component
      java.awt.Container
          java.awt.Window
              java.awt.Frame
                  javax.swing.JFrame
                      TelaPrincipal
```

All Implemented Interfaces:

java.awt.event.ActionListener, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, java.util.EventListener, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

---

```
public class TelaPrincipal
extends javax.swing.JFrame
implements java.awt.event.ActionListener
```

Classe que implementa a interface gráfica do sistema (GUI)

See Also:

Serialized Form

Field Summary

Fields inherited from class javax.swing.JFrame
EXIT_ON_CLOSE

Fields inherited from class java.awt.Frame
CROSSHAIR_CURSOR, DEFAULT_CURSOR, E_RESIZE_CURSOR, HAND_CURSOR, ICONIFIED, MAXIMIZED_BOTH, MAXIMIZED_HORIZ, MAXIMIZED_VERT, MOVE_CURSOR, N_RESIZE_CURSOR, NE_RESIZE_CURSOR, NORMAL, NW_RESIZE_CURSOR, S_RESIZE_CURSOR, SE_RESIZE_CURSOR, SW_RESIZE_CURSOR, TEXT_CURSOR, W_RESIZE_CURSOR, WAIT_CURSOR

Fields inherited from class java.awt.Component
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface javax.swing.WindowConstants
DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE, HIDE_ON_CLOSE

Fields inherited from interface java.awt.image.ImageObserver
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Constructor Summary

TelaPrincipal() Construtor sem argumentos da classe TelaPrincipal

Method Summary

void	actionPerformed(java.awt.event.ActionEvent e) Eventos dos componentes da GUI
java.lang.String	getResposta() Retorna a resposta corrente
java.awt.TextArea	getTxtExpert() Retorna o TextArea que recebe as mensagens do sistema
java.awt.TextArea	getTxtPergunta() Retorna o TextArea que mostra as perguntas ao usuário
javax.swing.JTextField	getTxtResposta() Retorna o JTextField que recebe as respostas do usuários
void	setTxtPergunta(java.lang.String texto) Configura o TextArea referente a pergunta

Methods inherited from class javax.swing.JFrame
getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getJMenuBar, getLayeredPane, getRootPane, isDefaultLookAndFeelDecorated, remove, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setIconImage, setJMenuBar, setLayeredPane, setLayout, update

Methods inherited from class java.awt.Frame
addNotify, getCursorType, getExtendedState, getFrames, getIconImage, getMaximizedBounds, getMenuBar, getState, getTitle, isResizable, isUndecorated, remove, removeNotify, setCursor, setExtendedState, setMaximizedBounds, setMenuBar, setResizable, setState, setTitle, setUndecorated

Methods inherited from class java.awt.Window
addPropertyChangeListener, addPropertyChangeListener, addWindowFocusListener, addWindowListener, addWindowStateListener, applyResourceBundle, applyResourceBundle, createBufferStrategy, createBufferStrategy, dispose, getBufferStrategy, getFocusableWindowState, getFocusCycleRootAncestor, getFocusOwner, getFocusTraversalKeys, getGraphicsConfiguration, getInputContext, getListeners, getLocale, getMostRecentFocusOwner, getOwnedWindows, getOwner, getToolkit, getWarningString, getWindowFocusListeners, getWindowListeners, getWindowStateListeners, hide, isActive, isAlwaysOnTop, isFocusableWindow, isFocusCycleRoot, isFocused, isLocationByPlatform, isShowing, pack, postEvent, removeWindowFocusListener, removeWindowListener, removeWindowStateListener, setAlwaysOnTop, setBounds, setCursor, setFocusableWindowState, setFocusCycleRoot, setLocationByPlatform, setLocationRelativeTo, show, toBack, toFront

Methods inherited from class java.awt.Container
add, add, add, add, add, addContainerListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalPolicy, getInsets, getLayout, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, removeAll, removeContainerListener, setComponentZOrder, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, transferFocusBackward, transferFocusDownCycle, validate

Methods inherited from class java.awt.Component
action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, disable, dispatchEvent, enable, enable, enableInputMethods,

firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getBackground, getBounds, getBounds, getColorModel, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusListeners, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getLocation, getLocation, getLocationOnScreen, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, imageUpdate, inside, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, prepareImage, prepareImage, printAll, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setComponentOrientation, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeysEnabled, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, size, toString, transferFocus, transferFocusUpCycle

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Methods inherited from interface java.awt.MenuContainer
getFont, postEvent

Constructor Detail

TelaPrincipal

public **TelaPrincipal**()

Construtor sem argumentos da classe TelaPrincipal

Method Detail

getTxtPergunta

`public java.awt.TextArea` **`getTxtPergunta`**`()`

Retorna o TextArea que mostra as perguntas ao usuário

Returns:

um `TextArea`

---

getTxtResposta

`public javax.swing.JTextField` **`getTxtResposta`**`()`

Retorna o JTextField que recebe as respostas do usuários

Returns:

um `JTextField`

---

getTxtExpert

`public java.awt.TextArea` **`getTxtExpert`**`()`

Retorna o TextArea que recebe as mensagens do sistema

Returns:

um `TextArea`

---

getResposta

`public java.lang.String` **`getResposta`**`()`

Retorna a resposta corrente

Returns:

um `String`

---

setTxtPergunta

`public void` **`setTxtPergunta`**`(java.lang.String texto)`

Configura o TextArea referente a pergunta

---

actionPerformed

`public void` **`actionPerformed`**`(java.awt.event.ActionEvent e)`

Eventos dos componentes da GUI

Specified by:

`actionPerformed` in interface `java.awt.event.ActionListener`

MODELO EDITAL UNIVERSAL

Observação: As palavras em itálico iniciadas por "ch_ " se tratam das variáveis trabalhadas pelo SE.

ch_Publicador no uso de suas atribuições legais tendo em vista *ch_DecretoIntroducao* torna pública a abertura de inscrições e as normas para a realização de Concurso Público para o provimento de cargos conforme o item: DAS ESPECIFICAÇÕES DOS CARGOS, VAGAS E OUTROS DADOS, mediante as condições estabelecidas neste Edital *ch_Anexos* .

1. DAS DISPOSIÇÕES PRELIMINARES

1.1 O Concurso Público, será regido por este Edital e executado pelo Núcleo de Concursos da Universidade Federal do Paraná – NC da UFPR *ch_Apoiador* .

1.2 A seleção de que trata este Edital consistirá de prova objetiva de conhecimentos, de caráter eliminatório e classificatório, a ser realizada pelo Núcleo de Concurso da Universidade Federal do Paraná.

1.3 Os candidatos aprovados serão convocados através de divulgação *ch_OutroSiteDivulgacaoResultado* e *ch_LocalDivulgacaoResultado* .

2. DAS VAGAS DESTINADAS A AFRO-DESCENDENTES

2.1 São reservadas aos afro-descendentes as vagas qualificadas no item 4.1 deste edital, conforme previsto na Lei 14.274, de 26 /12/2003.

2.2 Considerar-se-á afro-descendente aquele que assim se declarar expressamente, identificando-se como de cor preta ou parda, da raça etnia negra, devendo tal informação integrar os registros cadastrais de ingresso dos servidores.

2.3 O candidato afro-descendente que se inscrever para cargo cuja vaga está reservada, conforme item 4.1, deverá declarar essa condição no ato da inscrição.

2.4 Os candidatos às vagas reservadas aos afro-descendentes participarão do concurso em igualdade de condições com os demais candidatos no que concerne às exigências estabelecidas neste edital.

2.5 Detectada a falsidade na declaração a que se refere o item anterior, sujeitar-se-á o infrator às penas da Lei e ainda:

2.5.1.1 Se candidato, à eliminação do Concurso Público e de todos os atos dai decorrentes,

2.5.1.2 Se já admitido no cargo efetivo para o qual concorreu na reserva de vagas, utilizando-se da declaração inverídica, à pena disciplinar de demissão motivada.

2.6 No prazo de *ch_PrazoAfro* dias úteis, contados da divulgação dos resultados da aprovação, o candidato afro-descendente será convocado, de acordo com o número de vagas, a classificação e o interesse do apoiador para submeter-se à perícia para a verificação dos traços fenotípicos que o caracterizam na sociedade como pertencente ao grupo racial negro e à entrevista com banca a ser designada pel *ch_DesignadorBancaAfro* que decidirá se o candidato atende aos requisitos.

2.7 O concurso público será homologado depois da realização da perícia e da entrevista de que trata o item anterior, sem prejuízo da homologação prévia do resultado em relação aos candidatos da lista geral, que não concorrem às vagas reservadas.

2.8 As vagas reservadas serão liberadas aos demais candidatos, caso não tenha ocorrido inscrição para o concurso ou aprovação de candidato afro-descendente.

2.9 Os candidatos que se inscreverem para a reserva de vagas para afro-descendentes poderão concorrer, além das vagas que lhes são destinadas por Lei, à totalidade das vagas, desde que aprovados, habilitados e observada rigorosamente a ordem geral de classificação.

3. DAS VAGAS DESTINADAS A PESSOAS COM DEFICIÊNCIA

3.1 Às pessoas com deficiência, que pretendam fazer uso das prerrogativas que lhes são facultadas pelo inciso VIII do artigo 37 da Constituição Federal, de 5/10/1988, e do artigo 3 do Decreto n. 3298/1999 e alterações posteriores, que regulamenta a Lei n. 7.853/1999, é assegurado o direito de inscrição para os cargos oferecidos neste Edital, desde que a deficiência de que são portadoras seja compatível com as atribuições do cargo em provimento.

3.2 Em obediência ao disposto no item anterior fica reservado o percentual de 5% (cinco por cento) das vagas existentes para cada cargo, de acordo com o inciso 2º do artigo 5º da Lei n. 8.112/1990 e no § 1º do artigo 37 do Decreto n. 3.298/1999.

3.3 Considera-se pessoa com deficiência aquela que se enquadra nas categorias discriminadas no artigo 4º do Decreto n.3.298/99 e suas alterações posteriores, no Decreto n. 5296/2004, assim definidas:

3.3.1 Deficiência física: alteração completa ou parcial de um ou mais segmentos do corpo humano, acarretando o comprometimento da função física, apresentando-se sob a forma de paraplegia, parapesia, monoplegia, monoparesia, tetraplegia, tetrapesia, triplegia, triparesia, hemiplegia, jemiparesia, ostomia, amputação ou ausência de membro, paralisia cerebral, nanismo, membros com deformidade congênita, ou adquirida, exceto as deformidades estéticas e as que não produzam dificuldades para o desempenho de funções,

3.3.2 Deficiência auditiva: perda bilateral, parcial ou total, de quarenta e um decibéis (dB) ou mais, aferida por audiograma nas freqüências de 500 Hz, 1000 Hz, 2000 Hz e 3000 Hz;

3.3.3 Deficiência visual: cegueira, na qual a acuidade visual é igual ou menor que 0,05 no melhor olho, com a melhor correção óptica; a baixa visão, que significa acuidade visual entre 0,3 e 0,05 no melhor olho, com a melhor correção óptica; os casos nos quais a somatória da medida do campo visual em ambos os olhos for igual ou menor que 60º.; ou a ocorrência simultânea de quaisquer das condições anteriores,

3.3.4 Deficiência mental: funcionamento intelectual significativamente inferior à média, com manifestação antes dos dezoito anos e limitações associadas a duas ou mais áreas de habilidades adaptativas, tais como: comunicação, cuidado pessoal, habilidades sociais, utilização dos recursos da comunidade; saúde e segurança; habilidades acadêmicas; lazer e trabalho,

3.3.5 Deficiência múltipla: associação de duas ou mais deficiências.

3.4 De acordo com o Art.1 da Convenção da Guatemala o termo deficiência significa uma restrição física, mental ou sensorial, de natureza permanente ou transitória, que limita a capacidade de exercer uma ou mais atividades essenciais da vida diária, causada ou agravada pelo ambiente econômico e social (convenção da Guatemala).

3.5 No ato de inscrição, o candidato com deficiência poderá requerer condições diferenciadas para a realização das provas durante o período de inscrição conforme o item 6.8 deste edital.

3.6 São condições diferenciadas: prova ampliada, prova em Braile, prova em libras, solicitação de ledor, transcritor, guia e intérprete de libras, intérprete para leitura labial, mobiliário especial, equipamentos de informática, tempo adicional, etc.

3.7 Em caso de necessidade de tempo adicional para a realização das provas, o candidato com deficiência deverá apresentar justificativa, acompanhada de parecer emitido por especialista da área de sua deficiência, durante o período de inscrição, juntamente com o CID, conforme especificações no item 6.8 deste edital.

3.8 O atendimento às condições diferenciadas solicitadas, inclusive a possibilidade de tempo adicional de que trata o item 6.8 ficará sujeito à análise e razoabilidade e ao parecer favorável da Equipe Multiprofissional designada pel *ch_IndicaMultiprofissional* .

3.9 O candidato que se declarar deficiente participará do Concurso Público em igualdade de condições com os demais candidatos, no que diz respeito ao conteúdo das provas, à avaliação das provas e aos critérios de aprovação, ao dia, horário e local de aplicação das provas e à nota mínima exigida.

3.10 As vagas reservadas serão liberadas aos demais candidatos, caso não tenha ocorrido inscrição para o concurso ou aprovação de candidato com deficiência.

3.11 Os candidatos que, dentro do período de inscrição, não atenderem aos dispositivos mencionados nos itens 6.8 e 6.9 não serão considerados deficientes e não terão provas especiais preparadas.

3.12 O candidato com deficiência que não realizar a inscrição conforme as instruções dos itens 3.1, 3.2 e 3.3 perderá o direito de concorrer à reserva de vagas referido no item.

3.13 Encerrado o período de inscrições, a Equipe Multiprofissional designada pel *ch_IndicaMultiprofissional* analisará o laudo Médico encaminhado pelo candidato (item 6.8 deste edital), e verificará se há correspondência entre o laudo e o CID. Em caso negativo, o candidato terá sua inscrição excluída da lista de candidatos com deficiência, passando a integrar a lista geral de candidatos.

3.14 Após a investidura no cargo, a deficiência não poderá ser argüida para justificar a concessão de aposentadoria.

3.15 Por medida de segurança, as provas dos candidatos com deficiência poderão ser gravadas ou filmadas.

4. DAS ESPECIFICAÇÕES DOS CARGOS, VAGAS E OUTROS DADOS

4.1 Os cargos, os requisitos exigidos, a carga horária semanal, o total de vagas, a remuneração e o valor da taxa de inscrição são os seguintes:

4.1.1 Cargo: *ch_Cargos* ,

4.1.2 Requisitos: *ch_Requisitos* ,
4.1.3 Carga Horária: *ch_CargaHoraria* ,
4.1.4 Vagas: *ch_Vagas* ,
4.1.5 Vagas para Portadores de Deficiência: *ch_VagasDeficientes* ,
4.1.6 Vagas para Afro-descendentes: *ch_VagasAfro* ,
4.1.7 Remuneração: *ch_Remuneracao* ,
4.1.8 Taxa de inscrição: *ch_TaxaInscricao* .
4.2 Ao número de vagas estabelecidas nos subitens anteriores, serão acrescidas outras que surgirem ou forem criadas durante o prazo de validade do concurso.
4.3 As provas, os conteúdos programáticos que não estiverem estabelecidas neste Edital, deveram ser anunciadas nos anexos.

5. DAS CONDIÇÕES PARA INVESTIDURA NO CARGO

5.1 O candidato aprovado no Concurso de que trata este Edital, e convocado, será investido no cargo, se atendidas as seguintes exigências:
5.1.1 Ter nacionalidade brasileira, e aos estrangeiros os requisitos na forma da Lei, conforme a Constituição Federal,
5.1.2 Gozar dos direitos políticos,
5.1.3 Estar em dia com as obrigações eleitorais e, se do sexo masculino, também com as militares,
5.1.4 Comprovar por meio de Diploma ou Certificado o nível de escolaridade exigido para o exercício do cargo,
5.1.5 Ter idade mínima de 18 anos,
5.1.6 Não ter sido demitido do Serviço Público Estadual, Federal ou Municipal ch_ ResolucaoDemissao .
5.1.7 Ter aptidão física e mental compatíveis às suas funções, através de verificação pel *ch_AnaliseFisicaMental* ,
5.1.8 Apresentar certidão negativa de antecedentes criminais;
5.1.9 Apresentar declaração de bens e valores que constituam seu patrimônio e, se casado, a do cônjuge,
5.1.10 Apresentar declaração quanto ao exercício ou não de outro cargo, emprego ou função pública e de compatibilidade de horário para o exercício de mais de um cargo quando acumulável,
5.1.11 Declaração sobre percepção de proventos de aposentadoria, por conta do regime geral de previdência ou de qualquer outro regime próprio de previdência em âmbito federal, estadual ou municipal, decorrentes de cargos, emprego ou função pública, conforme artigo 40 da Constituição Federal e de serviço militar, conforme artigos 42 e 142 da Constituição Federal,
5.1.12 Apresentar comprovante de residência,
5.1.13 Apresentar outros documentos que se fizerem necessários à época da posse,
5.1.14 *ch_CarteiraHabilitacao* ,
5.2 A falta de comprovação ou a constatação de falsidade de qualquer um dos requisitos especificados e daqueles que vierem a ser estabelecidos, impedirá a investidura do candidato ao cargo.

6. DA INSCRIÇÃO

6.1 As inscrições serão realizadas no período de *ch_DataInicioInscricao* até às *ch_HoraFimInscricao* de *ch_DataFimInscricao* somente via internet no site www.nc.ufpr.br.

6.2 Antes de efetuar o recolhimento da taxa de inscrição o candidato deverá certificar-se de que preenche todos os requisitos exigidos para a participação no Concurso Público e para a investidura no cargo, pois a taxa, uma vez paga, não será restituída em hipótese alguma.

6.3 Após o preenchimento do formulário de inscrição, o candidato deverá confirmar as informações, imprimir o boleto bancário e pagá-lo preferencialmente nas casas lotéricas ou na rede bancária até o dia *ch_DataLimitePagamentoInscricao* devendo ao candidato guardar o comprovante de pagamento para a eventualidade de atestá-lo junto ao Núcleo de Concursos.

6.4 A inscrição somente será efetivada após a identificação eletrônica do pagamento da taxa de inscrição.

6.5 O simples agendamento do pagamento no banco não é o suficiente para efetivação da inscrição.

6.6 No ato da inscrição, o candidato deverá optar por um único cargo. Não será aceito qualquer pedido de alteração de cargo para o qual o candidato se inscreveu.

6.7 Caso o candidato faça mais de uma inscrição, será considerada apenas a última.

6.8 O candidato com deficiência ou que necessite de atendimento diferenciado para a realização da prova deverá encaminhar ao Núcleo de Concursos um laudo médico expedido no prazo de 90 dias antes do término das inscrições e um formulário próprio preenchido, disponível no site www.nc.ufpr.br.

6.8.1.1 O laudo médico e o formulário devem ser entregues pessoalmente ou enviados pelo correio via sedex, durante o período de inscrição, ao Núcleo de Concurso – UFPR – Setor de Ciências Agrárias – Rua dos Funcionáros nº 1540 - CEP 80035-050, Curitiba – PR.

6.8.1.2 O laudo médico deve atestar a espécie e o grau ou nível da deficiência, com expressa referência ao código correspondente da Classificação Internacional de Doença – CID, bem como a provável causa da deficiência, expedido no prazo de 3 meses.

6.9 Não será concedido atendimento especial a candidatos que não efetuarem o comunicado ao Núcleo de Concursos até a data especificada.

6.10 A partir de *ch_DataVerificarSituacaoInscricao* , o candidato poderá verificar, no site www.nc.ufpr.br, a situação da sua inscrição, devendo, em caso de alguma divergência, entrar em contato com o Núcleo de Concursos, pessoalmente ou pelo telefone -0xx41-3313-8800 Central de Atendimento ao Candidato – NC.

6.11 A partir de *ch_DataVerificarImpComprovante* o candidato deverá acessar o site www.nc.ufpr.br e imprimir o seu comprovante de ensalamento, no qual estará indicado o local, horário e endereço de realização das provas.

6.12 Não haverá isenção total ou parcial do valor da taxa de inscrição.

6.13 É vedada a transferência do valor pago a título para terceiros, assim como a inscrição para outrem.

6.14 Para efetuar a inscrição, é imprescindível o número de Cadastro de Pessoa Física –CPF– do candidato.

6.15 A inscrição será de inteira responsabilidade do candidato, assumindo ele total responsabilidade pelas informações prestadas no formulário de inscrição e arcando com as consequências de eventuais erros de preenchimento.

6.16 Ao se inscrever, o candidato aceita, de forma irrestrita, as condições contidas neste Edital e, caso houver, nos seus Anexos, não podendo delas alegar desconhecimento.

7. DAS PROVAS

7.1 As provas serão realizadas no dia *ch_DiaDeProvas* das *ch_HoraInicioDeProvas* às *ch_HoraFimDeProvas* em *ch_CidadeProvas* .

7.2 O local de prova do candidato constará no respectivo comprovante de ensalamento, disponível no site www.nc.ufpr.br.

7.3 As provas objetivas serão constituídas de questões de múltipla escolha, com cinco alternativas -a, b, c, d, e-, das quais apenas uma será correta.

7.4 As respostas às questões objetivas deverão ser transcritas para o cartão-resposta com caneta esferográfica com tinta preta, devendo o candidato assinalar uma única resposta para cada questão.

7.5 Não serão consideradas questões não assinaladas ou que contenham mais de uma resposta, emenda ou rasura.

7.6 O candidato assume plena e total responsabilidade pelo correto preenchimento do cartão-resposta e sua integridade. Não haverá substituição desse cartão, a não ser em caso de defeito em sua impressão.

7.7 Nas provas objetivas haverá para cada candidato, um caderno de prova e um cartão-resposta identificado e numerado adequadamente. A correção dessas provas será feita por meio de leitura óptica do cartão-resposta e, quando necessário, onde couber, com a verificação dos cálculos constantes no caderno de prova, os quais deverão atestar a consistência da resposta encontrada para cada questão.

7.8 Com o objetivo de garantir a lisura e a idoneidade do Concurso, o candidato deverá autenticar a ficha de identificação por meio de suas impressões digitais.

7.9 A ausência do candidato, por qualquer motivo, inclusive doença e atraso, implicará na sua eliminação do concurso.

7.10 Para ingresso na sala de provas, o candidato deverá apresentar o comprovante de ensalamento, juntamente com o original do documento oficial de identidade.

7.11 São documentos oficiais de identidade: Carteiras e/ou Cédulas de Identidade expedidas pelas Secretarias de Segurança, pelas Forças Armadas, pela Polícia Militar, pelo Ministério das Relações Exteriores, Cédulas de Identidade fornecidas por Ordens ou Conselhos de Classe que por lei federal valem como documento de identidade, como, por exemplo, as do CREA e OAB, etc., a Carteira de Trabalho e Previdência Social, bem como a Carteira Nacional de Habilitação (com fotografia), na forma da Lei n. 9.053/97.

7.12 Os documentos para ingresso na sala de prova, especificados no subitem anterior, devem estar em perfeitas condições e devem permitir, com clareza, a identificação dos candidatos.

7.13 Em caso de perda ou roubo de documentos, o candidato será admitido para realizar as provas, desde que tenha se apresentado no local que lhe foi designado e que seja possível fazer a verificação de seus dados junto ao Núcleo de Concursos antes da hora marcada para início das provas.

7.14 Nas salas de prova, os candidatos não poderão manter em seu poder relógios, armas e aparelhos eletrônicos - BIP, telefone celular, calculadora, agenda eletrônica, etc. Devendo seguir as orientações fornecidas pelos aplicadores

antes do início da prova. O candidato que estiver portando qualquer desses instrumentos durante a realização da prova será eliminado do Concurso.

7.15 Excepcionalmente e a critério médico devidamente comprovado, o candidato que estiver impossibilitado de realizar a prova em sala de prova, fará a mesma em local especialmente designado pelo Núcleo de Concurso – NC.

7.16 A candidata que tiver necessidade de amamentar durante a realização das provas deverá levar um acompanhante, que ficará responsável pela guarda da criança. A candidata que não levar acompanhante não realizará a prova.

7.17 Os casos citados nos itens 7.15 e 7.16 como também outro caso de emergência devem ser comunicados ao Núcleo de Concurso pelo fone/fax -0xx41- 3313-8832.

7.18 Não será permitido o ingresso ou a permanência de pessoas estranhas ao Concurso Público no local de aplicação das provas.

7.19 As portas de acesso aos prédios onde serão realizadas as provas serão fechadas às *ch_HoraFechaPortoes* Os relógios da Comissão Organizadora do Concurso serão acertados pelo Serviço Hora Certa Brasil Telecom – fone 130.

7.20 Poderá, a critério do Núcleo de Concurso, ocorrer a prorrogação no fechamento das portas de acesso de um ou mais locais onde serão realizadas as provas, em função de fatores externos.

7.21 Não será permitido ao candidato, durante a realização das provas, ausentar-se do recinto, a não ser em caso especiais e acompanhando de um membro componente da equipe de aplicação do Concurso.

7.22 Não será permitido, durante a realização da prova, o uso de bonés, gorros, chapéus, etc.

7.23 Não será permitido ao candidato alimentar-se dentro da sala de aula, devendo aqueles que necessitarem fazê-lo, por motivos médicos, solicitar ao aplicador de provas e encaminhamento à sala de inspeção.

7.24 Não será permitida, durante a realização da prova, comunicação entre os candidatos, nem o uso de calculadoras e/ou similares, livros, anotações, réguas de cálculo, impressos ou qualquer outro material de consulta.

7.25 O candidato somente poderá retirar-se da sala após ter sido decorrida 1:30h - uma hora e trinta minutos- do início da prova, devendo entregar ao aplicador da prova, obrigatoriamente, o caderno de provas e o cartão-resposta devidamente assinalado.

7.26 O caderno de provas conterá um espaço designado para anotação das respostas das questões objetivas, que poderá ser recortado e levado pelo candidato para posterior conferência do gabarito.

7.27 É de responsabilidade exclusiva do candidato a identificação correta de seu local de realização das provas e o comparecimento no horário determinado.

7.28 Constatada a qualquer tempo, por meio eletrônico, estatístico, visual ou grafológico, a utilização pelo candidato de procedimentos ilícitos, a prova será objeto de anulação e automaticamente o candidato será eliminado do Concurso, sem prejuízo das cominações legais civis e criminais deles decorrentes.

7.29 Será eliminado o candidato que, durante a aplicação das provas, praticar atos contra as normas ou disciplinas.

7.30 Os procedimentos e os critérios para correção das provas são de responsabilidade do NC.

8. DOS RECURSOS

8.1 Até *ch_QuantasHorasGabProvisorio* após a realização das provas, o NC publicará o gabarito provisório das questões objetivas no site www.nc.ufpr.br.

8.2 Serão aceitos questionamentos sobre o gabarito ou à formulação das questões desde que estejam em conformidade com o disposto nos itens 8.3, 8.4, 8.5, 8.6, 8.7, 8.8.

8.3 Os questionamentos devem estar devidamente fundamentados e apresentados em formulário específico que estará disponível no site www.nc.ufpr.br, até as *ch_HoraEntregaQuestionamentos* do dia *ch_DataEntregaQuestionamentos* .

8.4 Os candidatos devem imprimir o formulário específico e seguir rigorosamente as instruções contidas no site www.nc.ufpr.br.

8.5 Cada questão ou item deve ser apresentado em um formulário separado.

8.6 O recurso deverá ser protocolado na Secretaria do Núcleo de Concurso ou enviado apenas 1 via por Sedex ao Núcleo de Concursos – UFPR - Setor de Ciências Agrárias - Rua dos Funcionáros nº 1540 - CEP 80035-050, Curitiba – PR, desde que postado até as *ch_HoraFinalProtocolo* do dia *ch_DataFinalProtocolo* .

8.7 Serão desconsiderados pelo NC questionamentos que não estiverem redigidos no formulário específico, não protocolados, fora do prazo ou que não estiverem devidamente fundamentados, bem como encaminhados de forma diferente ao estabelecido nos itens anteriores.

8.8 Serão desconsiderados pelo NC questionamentos relativos ao preenchimento do cartão-resposta.

8.9 O recurso será apreciado por uma banca examinadora, designada pelo Núcleo de Concurso, que emitirá decisão fundamentada, a qual será colocada à disposição do requerente na Secretária no Nucleo de Concursos.

8.10 Uma vez julgados os recursos apresentados, será emitido novo gabarito.

8.11 A Banca Examinadora constitui última instância para recurso, sendo soberana em suas decisões, razão pela qual não caberão recursos adicionais.

8.12 Os pontos relativos às questões porventura anuladas serão atribuídos a todos os candidatos que fizerem a prova. Se houver alteração, por força de impugnações do gabarito oficial provisório, de item integrante de provas, essa alteração valerá para todos os candidatos, independentemente de terem recorrido.

8.13 Com exceção dos recursos previstos nos itens anteriores, não se concederá revisão de provas, segunda chamada, vistas, ou recontagem de pontos em qualquer das provas, devido às características do Concurso.

8.14 O gabarito oficial das provas objetivas será divulgado no site www.nc.ufpr.br até o dia *ch_DataGabaritoOficial* .

9. DA APROVAÇÃO E CLASSIFICAÇÃO

9.1 O número de questões objetivas para cada cargo e a pontuação das provas encontra-se nos Anexos.

9.2 O cálculo da pontuação da prova objetiva será igual a soma das respostas do candidato as quais estejam em concordância com o gabarito oficial final da prova.

9.3 Será eliminado o candidato que não obtiver no mínimo 50% (cinqüenta por cento) dos pontos na prova objetiva.

9.4 Em caso de empate na classificação final, será dado preferência ao candidato que obtiver maior nota na prova de conhecimentos específicos.

10. DO RESULTADO

10.1 O resultado oficial do Concurso será divulgado nos locais definidos no item 1.3, em duas listas, contendo a primeira a pontuação de todos os candidatos, inclusive, caso houver, a dos candidatos com deficiência e a segunda somente a pontuação desses últimos.

10.2 Serão emitidas duas listagens por cargo, sendo uma em ordem alfabética e outra classificatória geral.

10.3 Somente será considerada oficial e válida a lista dos classificados, após publicada nos locais definidos no item 1.3.

10.4 Toda divulgação por outros meios será considera tão somente como auxiliar para os interessados, não sendo reconhecido nessa divulgação qualquer caráter oficial.

10.5 O resultado e o desempenho de cada candidato serão divulgados até o dia *ch_DataDivulgacaoDesempenho* .

10.6 Caberá ao *ch_HomologadorResultadoFinal* , ou pessoa por ele delegada, homologar o resultado final do Concurso Público.

10.7 Os candidatos que, após decorridos 6 meses da realização das provas, ainda desejarem alguma informação sobre o resultado do concurso devem entrar em contato com *ch_LocalInformacoesSeisMeses* .

11. DA CONVOCAÇÃO

11.1 O candidato classificado e desde que observado o número de vagas existentes, será convocado para a nomeação quando então, se for de sua conveniência, poderá requerer o seu deslocamento para o final da lista, uma única vez, mediante pedido a ser efetuado em formulário próprio, fornecido pel *ch_QuemForneceraFormularioDeslocamento*.

11.2 Os candidatos convocados para nomeação terão o prazo improrrogável de *ch_PrazoApresentacaoRh* dias úteis da publicação na imprensa, do Edital de chamamento para se apresentarem *ch_LocalApresentacaoRh*, decorridos os quais e não havendo o registro do comparecimento, o candidato será considerado como desistente.

11.3 Comparecendo o convocado ao órgão de administração do Poder, será encaminhando para avaliação de saúde física e mental a ser realizada pel *ch_QuemRealizaraAvaliacaoSaudeFisicaMental* sendo que o não comparecimento na data marcada, implicará na eliminação do convocado no processo de nomeação.

11.4 O convocado aprovado na avaliação de saúde física e mental terá *ch_PrazoJuntadaDosDocumentos* dias úteis para juntada de documentos necessários à nomeação, devendo estes ser entregues no órgão de pessoal, em caso do não comparecimento no citado prazo, o convocado será considerado como desistente da vaga.

12. DA NOMEAÇÃO

12.1 O provimento efetivo será no nível inicial do cargo no qual o candidato se inscreveu, de conformidade com a Lei n. 41/89 e alterações e Lei

Complementar n. 02/04 e alterações, obedecendo rigorosamente à ordem de classificação dos candidatos habilitados, de acordo com a necessidade da Administração, não havendo portanto, obrigatoriedade de admissão total dos aprovados.

12.2 Quando da nomeação deverão ser chamados alternada e proporcionalmente os candidatos das duas listas, prosseguindo-se até a caducidade do concurso.

12.3 A convocação de forma alternada iniciar-se-á com o primeiro candidato da lista geral, passando-se ao primeiro da lista especial. As demais convocações seguirão pela lista geral e especial até completar a porcentagem definida pelo certame.

12.4 Para efeitos de nomeação, os candidatos ficam sujeitos, sob pena de eliminação, a apresentação dos documentos constantes no artigo 17 da Lei n. 525, de 25-03-2004 e alterações e demais, documentos conforme item 5 deste Edital.

12.5 É vedada ao candidato aprovado e convocado, a acumulação de cargos públicos, salvo para os cargos previstos na Constituição Federal, artigo 37, inciso XVI, alíneas a, b e c.

12.6 O candidato aprovado e convocado, terá *ch_PrazoPosseCargo* dias contados da publicação do ato da nomeação em *ch_LocalNomeacao* para tomar posse do cargo, conforme o Art. 16 da Lei 525, de 25-03-2004 e alterações.

12.7 Caso haja demanda de novas investiduras acima do número de vagas estimado no item 4.1, dentro do prazo de validade do certame, a Administração poderá convocar candidatos aprovados, observada a ordem de classificação.

12.8 A nomeação do candidato ao cargo fica condicionada ao atendimento dos requisitos básicos para investidura constante do item 5 deste Edital e da legislação vigente.

12.9 A nomeação seguirá rigorosamente a ordem classificatória no concurso e atenderá ao requisito de aprovação em Exame de Saúde Física e Mental, conforme mencionado no item 11.3 .

13. DAS DISPOSIÇÕES GERAIS

13.1 Será automaticamente eliminado do Concurso Público o candidato que não cumprir as normas estabelecidas ou não preencher todos os requisitos previstos no Edital.

13.2 Se o candidato necessitar de declaração de participação no Concurso, deverá dirigir-se à Inspetoria do seu local de realização de prova somente no dia do Concurso.

13.3 O prazo de validade do presente concurso é de *ch_PrazoValidadeConcurso*, contados a partir da data de publicação da homologação do resultado final, podendo ser prorrogado, uma única vez, por igual período, a critério d *ch_QuemProrrogaConcurso*.

13.4 A classificação no presente concurso não assegura ao candidato o direito de ingresso automático nas vagas ofertadas por este concurso, mas apenas a expectativa de ser nomeado, seguindo a rigorosa ordem classificatória, ficando a concretização do ato de nomeação condicionada às observâncias das disposições legais pertinentes e, sobretudo, ao interesse e conveniência d *ch_NomeacaoCondicionada*.

13.5 Não será fornecido ao candidato qualquer documento comprobatório de classificação no Concurso, valendo para este fim, a listagem publicada conforme esclarecido no item 1.3 .

13.6 O NC poderá, por motivos relevantes, cancelar ou anular as provas em um ou mais locais de prova, devendo os candidatos dos cursos afetados submeter-se a novas provas em datas e locais fixados pelo NC, sem necessidade de pagamento de nova taxa de inscrição.

13.7 Quaisquer alterações nas regras fixadas neste Edital somente poderão ser realizadas por intermédio de outro Edital.

13.8 O candidato deverá manter atualizado o seu endereço perante a *ch_LocalManterEnderecoAtualizado*, enquanto da validade deste concurso.

13.9 A contratante terá o prazo de *ch_PrazoRetiradaDocumentos* após a homologação do resultado para a retirada de todo o material utilizado no concurso provas, cartões-resposta, etc. Decorrido esse prazo todo o material será incinerado.

13.10 As despesas relativas à participação do candidato no Concurso serão de responsabilidade do candidato.

13.11 Quaisquer alterações nas regras fixadas neste Edital somente poderão ser realizadas por intermédio de outro Edital.

13.12 Os caso omissos deste Edital serão resolvidos pela Comissão Executiva do Concurso Público, constituída pelo Decreto n. 1387 de 14-02-2006.

13.13 Este Edital entra em vigor na data de sua publicação.

DATA_____

_____

Presidente da Comissão Executiva do Concurso Público(tem que ser feita a pergunta da pessoa e do cargo que ocupa

_____

Agente Administrativo(tem que ser feita a pergunta sobre a pessoa e o cargo que ocupa

PERGUNTAS PARA O EDITAL UNIVERSAL

QUESTIONÁRIO PADRÃO
INTRODUÇÃO (introducao)

ITEM:

- id – introducao
- sequencia – 0
- nome –
- negrito – 0

- status – nil

VARIÁVEL:
- id – ch_Publicador
- item – introducao
- descricao – "O Concurso torna-se aberto ao público mediante as atribuições legais de alguma pessoa física ou jurídica. A resposta a esta pergunta irá indicar ao sistema quem é esta pessoa. Você deve iniciar a resposta com o artigo adequado. Exemplo de resposta: A Comissão Executiva do Concurso Público"

PERGUNTA:
- id – ch_Publicador
- tipo – fraseComArtigo
- texto – "Quem torna pública a realização do Concurso? (Inicie com o artigo adequado)"

VARIÁVEL:
- id – ch_TipoDecretoIntroducao
- item – introducao
- descricao – "O sistema necessita saber se a resolução legal que torna pública a realização do concurso é do tipo Decreto, Lei ou Portaria."

PERGUNTA:
- id – ch_TipoDecretoIntroducao
- tipo – escolha
- texto – "Entre com o tipo da resolução legal que torna pública a realização do concurso (1 Decreto, 2 Lei, 3 Portaria) "
- teste – 1 2 3 -1 -2 -3 -4 -5 -6

VARIÁVEL:
- id – ch_NumDecretoIntroducao
- item – introducao
- descricao – "A resposta a esta pergunta irá definir o número da resolução legal que torna pública a realização do concurso. Exemplo de resposta: 1387."

PERGUNTA:
- id – ch_NumDecretoIntroducao
- tipo – numero

- texto – "Entre com o número da resolução legal que torna pública a realização do concurso."

VARIÁVEL:

- id – ch_DataDecretoIntroducao
- item – introducao
- descricao – "A resposta a esta pergunta irá indicar ao sistema a data da resolução legal que torna pública a realização do concurso. Esta deve ser fornecida no formato dia-mês-ano. Exemplo: 14-02-06"

VARIÁVEL:

- id – ch_DecretoIntroducao
- item – introducao
- descricao – "Variável utilizada internamente pelo sistema, resultante do processamento das variáveis ch_TipoDecretoIntroducao, ch_NumDecretoIntroducao e ch_DataDecretoIntroducao. Será utilizada na sentença ch_DecretoIntroducao."

PERGUNTA:

- id – ch_DataDecretoIntroducao
- tipo – data
- texto – "Entre com a data da resolução legal que torna pública a realização do concurso. (dd-mm-aa)."

VARIÁVEL:

- id – ch_Anexos
- item – introducao
- descricao – "Variável utilizada internamente pelo sistema, resultante do processamento da variável ch_NumAnexos. Será utilizada na sentenca ch_Anexos."

PERGUNTA:

- id – ch_NumAnexos
- tipo – numero
- texto – "Quantos anexos o Edital possui? (0 – Nenhum)"
-

DISPOSIÇÕES PRELIMINARES (dispoPreliminares)

ITEM:

- id – dispoPreliminares

- sequencia – 1
- nome – DAS DIPOSIÇÕES PRELIMINARES
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_Apoiador
- item – dispoPreliminares
- descricao – "O sistema necessita saber quem irá apoiar o Núcleo de Concursos para a realização do concurso. Você deve iniciar a resposta com o artigo adequado. Exemplo: A Secretaria Municipal da Administração de São José dos Pinhais"

PERGUNTA:

- id – ch_Apoiador
- tipo – fraseComArtigo
- texto – "Quem apoiará o NC para a realização do Concurso. (Inicie com o artigo adequado)"

VARIÁVEL:

- id – ch_OutroSiteDivulgacaoResultado
- item – dispoPreliminares
- descricao – "Além do site do núcleo de concursos, o resultado do concurso pode ser divulgado em algum outro endereço eletrônico. Exemplo: Sim – s, Não - n

PERGUNTA:

- id – ch_OutroSiteDivulgacaoResultado
- tipo – escolha
- texto – "Entre com s para sim ou n para não"
- teste – (s S n N -1 -2 -3 -4 -5 -6)

VARIÁVEL:

- id – ch_SiteDivulgacaoResultado
- item – dispoPreliminares
- descricao – "Caso a variável ch_OutroSiteDivulgacaoResultado seja respondida com sim o sistema necessita saber qual é o endereço eletrônico opcional."

PERGUNTA:

- id – ch_SiteDivulgacaoResultado
- tipo – webSite
- texto – "Entre com o endereço eletrônico (opcional) para a divulgação do resultado."

VARIÁVEL:

- id – ch_LocalDivulgacaoResultado
- item – dispoPreliminares
- descricao – "O sistema necessita saber qual é o local de divulgação do resultado. Exemplo: Órgão de Imprensa do Município."

PERGUNTA:

- id – ch_LocalDivulgacaoResultado
- tipo - fraseComArtigo
- texto – "Entre com o local para a publicação do resultado."

VAGAS DESTINADAS A AFRO-DESCENDENTES (afro)

ITEM:

- id – afro
- sequencia – 2
- nome – DAS VAGAS DESTINADAS A AFRO-DESCENDENTES
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_PrazoAfro
- item – afro
- descricao – "O sistema necessita saber quantos dias úteis o candidato tem para se apresentar para a perícia para a verificação dos traços fenotípicos que o caracterizam na sociedade como pertencente ao grupo racial negro e à entrevista que decidirá se o candidato atende aos requisitos. Exemplo de resposta: 5"

PERGUNTA:

- id – ch_PrazoAfro
- tipo – numero
- texto – "Qual é o prazo (em dias úteis) para que o candidato convocado apresente-se para a verificação dos traços fenotípicos?"

VARIÁVEL:

- id – ch_DesignadorBancaAfro
- item – afro
- descricao – "O sistema necessita quem será o responsável por designar a banca que fará a entrevista com o candidato para decidir se o mesmo atende aos requisitos necessários para concorrer as vagas reservadas para afro-descendentes."

PERGUNTA:

- id – ch_DesignadorBancaAfro
- tipo – fraseComArtigo
- texto – "Quem é a pessoa (física ou jurídica) que designará a entrevista."

VAGAS DESTINADAS A PESSOAS COM DEFICIÊNCIA (deficientesFisicos)

ITEM:

- id – deficientesFisicos
- sequencia – 3
- nome – DAS VAGAS DESTINADAS A PESSOAS COM DEFICIÊNCIA
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_IndicaMultiprofissional
- item – deficientesFisicos
- descricao – "A resposta a esta pergunta irá informar ao sistema quem é a pessoa responsável pela indicação da Equipe Multiprofissional. Esta última corresponde a uma equipe de profissionais que irá fornecer um parecer quanto ao atendimento das condições diferenciadas solicitadas pelo candidato portador deficiência física."

PERGUNTA:

- id – ch_IndicaMultiprofissional
- tipo – fraseComArtigo
- texto – "Entre com a pessoa (física ou jurídica) que designará a Equipe Multiprofissional."

ESPECIFICAÇÃO DOS CARGOS, VAGAS E OUTROS DADOS (cargos)

ITEM:

- id – cargos
- sequencia – 4
- nome – DAS ESPECIFICAÇÕES DOS CARGOS, VAGAS E OUTROS DADOS
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_QuantidadeCargos
- item – cargos
- descricao – "O usuário deve informar ao sistema a quantidade de cargos que Concurso irá prover."

PERGUNTA:

- id – ch_QuantidadeCargos
- tipo – frase
- texto – "Quantos cargos o Concurso ira prover? "

VARIÁVEL:

- id – ch_Cargos
- item – cargos
- descricao – "O usuário deve informar ao sistema o cargo. Exemplo: Advogado."

PERGUNTA:

- id – ch_Cargos
- tipo – frase
- texto – "Entre com o cargo."

VARIÁVEL:

- id – ch_Requisitos
- item – cargos
- descricao – "O usuário deve informar quais são os requisitos para o cargo inserido (Quando existir mais de um requisito o usuário deverá separa-los por vírgula). Exemplo: Curso Superior em Direito, Registro na Ordem dos Advogados do Brasil."

PERGUNTA:

- id – ch_Requisitos
- tipo – frase
- texto – "Entre com o(s) Requisito(s)."

VARIÁVEL:

- id – ch_CargaHorariaSemanal
- item – cargos
- descricao – "O usuário deve informar qual é a carga horária semanal para o cargo inserido. Exemplo: 40."

PERGUNTA:

- id – ch_CargaHorariaSemanal
- tipo – numero
- texto – "Entre com a carga horára semanal."

VARIÁVEL:

- id – ch_Vagas
- item – cargos
- descricao – "O usuário deve informar qual é a quantidade de vagas para o cargo inserido. Exemplo: 10."

PERGUNTA:

- id – ch_Vagas
- tipo – numero
- texto – "Entre com a quantidade de vagas."

VARIÁVEL:

- id – ch_VagasAfro
- item – cargos
- descricao – "O usuário deve informar qual é a quantidade de vagas reservadas para Afro-Descendentes. Exemplo: 4."

PERGUNTA:

- id – ch_VagasAfro
- tipo – numero

- texto – "Entre com a quantidade de vagas reservadas para Afro-Descendentes."

VARIÁVEL:
- id – ch_VagasDefi
- item – cargos
- descricao – "O usuário deve informar qual é a quantidade de vagas reservadas para Pessoas com Deficiência. Exemplo: 4."

PERGUNTA:
- id – ch_VagasDefi
- tipo – numero
- texto – "Entre com a quantidade de vagas reservadas para Pessoas com Deficiência."

VARIÁVEL:
- id – ch_Remuneracao
- item – cargos
- descricao – "O usuário deve informar o valor da remuneração para o cargo inserido. Exemplo: 2.117,03"

PERGUNTA:
- id – ch_Remuneracao
- tipo – numero
- texto – "Entre com a remuneração."

VARIÁVEL:
- id – ch_TaxaInscricao
- item – cargos
- descricao – "O usuário deve informar o valor da taxa de inscrição. Exemplo: 50,00."

PERGUNTA:
- id – ch_TaxaInscricao
- tipo – numero
- texto – "Entre com o valor da taxa de inscrição."

CONDIÇÕES PARA INVESTIDURA NO CARGO (investidura)

ITEM:

- id – investidura
- sequencia – 5
- nome – DAS CONDIÇÕES PARA INVESTIDURA NO CARGO
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_TipoResolucaoDemissao
- item – investidura
- descricao – "Uma das exigências para que o candidato aprovado no concurso público seja investido no cargo é que ele não tenha sido demitido em serviço público municipal, estadual ou federal. Existe uma resolução legal que trata deste assunto. O sistema necessita saber qual é o tipo desta resolução legal, se é decreto, lei ou portaria."

PERGUNTA:

- id – ch_TipoResolucaoDemissao
- tipo – escolha
- texto – "Entre com o tipo da resolução legal que trata da demissão de Serviço Público (1 Decreto, 2 Lei, 3 Portaria)"
- teste – 1 2 3 -1 -2 -3 -4 -5 -6

VARIÁVEL:

- id – ch_NumResolucaoDemissao
- item – investidura
- descricao – "A resposta a esta pergunta irá definir o número da resolução legal que trata da demissão de Serviço Público. Exemplo: 525."

PERGUNTA:

- id – ch_NumResolucaoDemissao
- tipo – numero
- texto – "Entre com o número da resolução legal que trata da demissão de Serviço Público.

VARIÁVEL:

- id – ch_DataResolucaoDemissao
- item – investidura
- descricao – "A resposta a esta pergunta irá indicar ao sistema a data da resolução legal que trata da demissão de Serviço Público. Esta deve ser fornecida no formato dia-mês-ano. Exemplo: 25-03-04"

VARIÁVEL:

- id – ch_ResolucaoDemissao
- item – investidura
- descricao – "Variável utilizada internamente pelo sistema, resultante do processamento das variáveis ch_TipoResolucaoDemissao, ch_NumResolucaoDemissao e ch_DataResolucaoDemissao. Será utilizada na sentenca ch_ResolucaoDemissao."

PERGUNTA:

- id – ch_DataResolucaoDemissao
- tipo – data
- texto – "Entre com a data da resolução legal que trata da demissão de Serviço Público. (dd-mm-aa).

VARIÁVEL:

- id – ch_AnaliseFisicaMental
- item – investidura
- descricao – "O candidato aprovado no Concurso de que trata este edital deverá ter aptidão física e mental compatíveis às funções que irá exercer. A resposta a esta pergunta irá indicar para o sistema quem será o órgão responsável pela verificação do atendimento a estas condições."

PERGUNTA:

- id – ch_AnaliseFisicaMental
- tipo – fraseComArtigo
- texto – "Entre com o órgão responsável pela análise da aptidão física e mental do candidato aprovado."

VARIÁVEL:

- id – ch_CarteiraHabilitacao
- item – investidura
- descricao – "Há casos em que é obrigatório possuir carteira de habilitação para que o candidato aprovado no Concurso seja investido no cargo. A resposta a esta pergunta irá indicar ao sistema se esta condição se aplica a este Concurso."

PERGUNTA:

- id – ch_CarteiraHabilitacao
- tipo – escolha
- texto – "É necessário possuir carteira de habilitação?"

- teste – s S n N -1 -2 -3 -4 -5 -6

INSCRIÇÃO (inscricao)

ITEM:

- id – inscricao
- sequencia – 6
- nome – "DA INSCRIÇÃO"
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_DataInicioInscricao
- item – inscricao
- descricao – "O sistema necessita saber qual é a data inicial para a inscrição do concurso. A resposta deve ser no formato dia-mês-ano. Exemplo: 13-09-06. "

PERGUNTA:

- id – ch_DataInicioInscricao
- tipo – data
- texto – "Entre com a data inicial para a inscrição do concurso. (dd-mm-aa)."

VARIÁVEL:

- id – ch_DataFimInscricao
- item – inscricao
- descricao – "O sistema necessita saber qual é a data final para a inscrição do concurso. A resposta deve ser no formato dia-mês-ano. Exemplo: 13-09-06. "

PERGUNTA:

- id – ch_DataFimInscricao
- tipo – data
- texto – "Entre com a data final para a inscrição do concurso. (dd-mm-aa)."

VARIÁVEL:

- id – ch_HoraFimInscricao
- item – inscricao

- descricao – "O sistema necessita saber qual é a hora para final para a inscrição do concurso. A resposta deve ser no formato hora:minuto. Exemplo: 18:30."

PERGUNTA:

- id – ch_HoraFimInscricao
- tipo – hora
- texto – "Entre com a hora final para a inscrição (hh:mm)."

VARIÁVEL:

- id – ch_DataLimitePagamentoInscricao
- item – inscricao
- descricao – "O sistema necessita saber qual é a data limite para efetuar o pagamento da taxa de inscrição. A resposta deve ser fornecida no formato dia-mês-ano. Exemplo: 24-04-07."

PERGUNTA:

- id – ch_DataLimitePagamentoInscricao
- tipo – data
- texto – "Entre com a data limite para o pagamento da taxa de inscrição. (dd-mm-aa)"

VARIÁVEL:

- id – ch_DataVerificarSituacaoInscricao
- item – inscricao
- descricao – "O sistema necessita saber qual é a data para o candidato verificar a situação da inscrição, ou seja, se foi realizada de forma efetiva ou se há alguma divergência. A resposta deve ser fornecida no formato dia-mês-ano. Exemplo de resposta: 24-05-07 "

PERGUNTA:

- id – ch_DataVerificarSituacaoInscricao
- tipo – data
- texto – "Entre com a data inicial para verificação da situação de inscrição? (dd-mm-aa) "

VARIÁVEL:

- id – ch_DataVerificarImpComprovante
- item – inscricao

- descricao – "O sistema necessita saber qual é a data para que o candidato possa imprimir o comprovante de ensalamento , no qual estará indicado o local, horário e endereço de realização das provas."

PERGUNTA:

- id – ch_DataVerificarImpComprovante
- tipo – data
- texto – "Entre com a data para a impressão do comprovante de ensalamento (dd-mm-aa)."

PROVAS (provas)

ITEM:

- id – provas
- sequencia – 7
- nome – DAS PROVAS
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_NumDiasProvas
- item – provas
- descricao – "O sistema necessita saber quantos dias de provas o concurso consistirá. Exemplo de resposta: 2"

PERGUNTA:

- id – ch_NumDiasProvas
- tipo – numero
- texto – "As provas discursivas e objetivas serão realizadas em quantos dias?"

VARIÁVEL:

- id – ch_DiaDeProvas
- item – provas
- descricao – "O sistema necessita saber qual é o dia para a realização das provas. A resposta necessita ser no formato dia-mês-ano. Exemplo: 20-08-06."

PERGUNTA:

- id – ch_DiaDeProvas
- tipo – data
- texto – "Entre com a data para a realização das provas discursivas e objetivas. (dd-mm-aa)"

VARIÁVEL:

- id – ch_HoraInicioDeProvas
- item – provas
- descricao – "O sistema necessita saber em que horário as provas irão iniciar. A resposta necessita ser no formato hora:minuto. Exemplo: 14:00"

PERGUNTA:

- id – ch_HoraInicioDeProvas
- tipo – hora
- texto – "Entre com o horário do início da realização das provas discursivas e objetivas. (hh:mm)"

VARIÁVEL:

- id – ch_HoraFimDeProvas
- item – provas
- descricao – "O sistema necessita saber qual é o horário limite para que o candidato termine a realização das provas. A resposta necessita ser no formato hora:minuto. Exemplo: 18:00"

PERGUNTA:

- id – ch_HoraFimDeProvas
- tipo – hora
- texto – "Entre com o horário de término da realização das provas discursivas e objetivas. (hh:mm)"

VARIÁVEL:

- id – ch_CidadeProvas
- item – provas
- descricao – "O sistema necessita saber em que cidade as provas discursivas e objetivas serão realizadas. A resposta necessita ser no formato Cidade-UF. Exemplo: Curitiba-PR"

PERGUNTA:

- id – ch_CidadeProvas
- tipo – frase

- texto – "Entre com a cidade aonde as provas discursivas e objetivas serão realizadas. (Cidade-UF)"

VARIÁVEL:

- id – ch_HoraFechaPortoes
- item – provas
- descricao – "O sistema necessita saber qual é o horário limite em que as portas de acesso aos prédios onde serão realizadas as provas serão fechadas. A resposta necessita ser no formato hora:minuto. Exemplo: 13:30"

PERGUNTA:

- id – ch_HoraFechaPortoes
- tipo – hora
- texto – "Entre com o horário de fechamento dos portões. (hh:mm)"

RECURSOS (recursos)

ITEM:

- id – recursos
- sequencia – 8
- nome – DOS RECURSOS
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_QuantasHorasGabProvisorio
- item – recursos
- descricao – "O sistema necessita saber após quantas horas da realização das provas o gabarito provisório será divulgado. Exemplo de resposta: 24"

PERGUNTA:

- id – ch_QuantasHorasGabProvisorio
- tipo – numero
- texto – "Quantas horas após a realização das provas será divulgado o gabarito provisório?"

VARIÁVEL:

- id – ch_DataEntregaQuestionamentos

- item – recursos
- descricao – "O sistema necessita saber qual é a data final para a entrega do formulário específico. Tal formulário é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões. "

PERGUNTA:

- id – ch_DataEntregaQuestionamentos
- tipo – data
- texto – "Entre com a data final para a entrega dos questionamentos. (dd-mm-aa) "

VARIÁVEL:

- id – ch_HoraEntregaQuestionamentos
- item – recursos
- descricao – "O sistema necessita saber qual é a hora limite para a entrega do formulário específico. Tal formulário é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões. "

PERGUNTA:

- id – ch_HoraEntregaQuestionamentos
- tipo – hora
- texto – "Entre com a hora final para a entrega dos questionamentos. (hh:mm) "

VARIÁVEL:

- id – ch_DataFinalProtocolo
- item – recursos
- descricao – "O sistema necessita saber qual é a data limite para protocolar o recurso. Tal procedimento é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões. "

PERGUNTA:

- id – ch_DataFinalProtocolo
- tipo – data
- texto – "Entre com a data final para protocolar recurso. (dd-mm-aa) "

VARIÁVEL:

- id – ch_HoraFinalProtocolo

- item – recursos
- descricao – "O sistema necessita saber qual é a hora limite para protocolar o recurso. Tal procedimento é necessário para que o candidato apresente algum questionamento sobre o gabarito ou à formulação das questões. "

PERGUNTA:

- id – ch_HoraFinalProtocolo
- tipo – hora
- texto – "Entre com a hora final para protocolar recurso. (hh:mm) "

VARIÁVEL:

- id – ch_DataGabaritoOficial
- item – recursos
- descricao – "O sistema necessita saber qual é a data de divulgação do gabarito oficial. Sobre este gabarito não será aceito nenhum tipo recurso. "

PERGUNTA:

- id – ch_DataGabaritoOficial
- tipo – data
- texto – "Entre com a data de divulgação do gabarito oficial. (dd-mm-aa) "

APROVAÇÃO E CLASSIFICAÇÃO (classificacao)

ITEM:

- id – classificacao
- sequencia – 9
- nome – "DA APROVAÇÃO E CLASSIFICAÇÃO"
- negrito – 1
- status – nil

RESULTADO (resultado)

ITEM:

- id – resultado
- sequencia – 10
- nome – "DO RESULTADO"
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_OutroSiteDivulgacaoResultado
- item – dispoPreliminares
- descricao – "Além do site do núcleo de concursos, o resultado do concurso pode ser divulgado em algum outro endereço eletrônico. Exemplo: Sim – s, Não - n

PERGUNTA:

- id – ch_OutroSiteDivulgacaoResultado
- tipo – escolha
- texto – "Entre com s para sim ou n para não"
- teste – (s S n N -1 -2 -3 -4 -5 -6)

VARIÁVEL:

- id – ch_SiteDivulgacaoResultado
- item – dispoPreliminares
- descricao – "Caso a variável ch_OutroSiteDivulgacaoResultado seja respondida com sim o sistema necessita saber qual é o endereço eletrônico opcional."

PERGUNTA:

- id – ch_SiteDivulgacaoResultado
- tipo – webSite
- texto – "Entre com o endereço eletrônico (opcional) para a divulgação do resultado."

VARIÁVEL:

- id – ch_LocalDivulgacaoResultado
- item – dispoPreliminares
- descricao – "O sistema necessita saber qual é o local de divulgação do resultado. Exemplo: Órgão de Imprensa do Município."

VARIÁVEL:

- id – ch_LocalDivulgacaoResultado
- item – dispoPreliminares
- descricao – "Entre com o local para a publicação do resultado."

VARIÁVEL:

- id – ch_DataDivulgacaoDesempenho
- item – resultado

- descricao – "O sistema necessita saber qual é a data limite em que os candidatos poderão consultar o seu desempenho. A resposta deverá ser fornecida no formato dia-mês-ano. Exemplo de resposta: 05-09-07."

PERGUNTA:

- id – ch_DataDivulgacaoDesempenho
- tipo – data
- texto – "Entre com a data limite para a divulgação do desempenho dos candidatos. (dd-mm-aa)"

VARIÁVEL:

- id – ch_HomologadorResultadoFinal
- item – resultado
- descricao – "O sistema necessita saber a quem caberá homologar o resultado final do Concurso Público. Lembre-se de utilizar as preposições adequadas na resposta a esta pergunta. Exemplo de resposta: Ao prefeito municipal de São José dos Pinhais."

PERGUNTA:

- id – ch_HomologadorResultadoFinal
- tipo – fraseComPreposição
- texto – "A quem caberá homologar o resultado final do concurso? (Utilize as preposições corretas)"

VARIÁVEL:

- id – ch_LocalInformacoesSeisMeses
- item – resultado
- descricao – "Existe um local específico para obter informações referentes ao concurso, após o período de seis meses da realização das provas. O sistema necessita saber qual é este local. Lembre-se de utilizar os artigos adequados na resposta a esta pergunta. Exemplo de resposta: A prefeitura municipal de São José dos Pinhais."

PERGUNTA:

- id – ch_LocalInformacoesSeisMeses
- tipo – fraseComArtigo
- texto – "Entre com o local no qual o candidato poderá obter informações referentes ao concurso após 6 meses da realização das provas."

CONVOCAÇÃO (convocacao)

ITEM:

- id – convocacao
- sequencia – 11
- nome – DA CONVOCAÇÃO
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_QuemForneceraFormularioDeslocamento
- item – convocacao
- descricao – "O usuário deverá informar ao sistema quem fornecerá o formulário de deslocamento."

PERGUNTA:

- id – ch_QuemForneceraFormularioDeslocamento
- tipo – fraseComArtigo
- texto – "Quem fornecerá o formulário de deslocamento?"

VARIÁVEL:

- id – ch_PrazoApresentacaoRh
- item – convocacao
- descricao – "O usuário deverá informar ao sistema qual será o prazo (em dias) para os candidatos se apresentarem no RH. Exemplo: 5."

PERGUNTA:

- id – ch_PrazoApresentacaoRh
- tipo – numero
- texto – "Qual será o prazo para os candidatos se apresentarem no RH?"

VARIÁVEL:

- id – ch_LocalApresentacaoRh
- item – convocacao
- descricao – "O usuário deverá informar ao sistema qual será o local para os candidatos aprovados se apresentarem."

PERGUNTA:

- id – ch_LocalApresentacaoRh
- tipo – fraseComArtigo
- texto – "Qual será o local para os candidatos aprovados se apresentarem?"

VARIÁVEL:

- id – ch_QuemRealizaraAvaliacaoSaudeFisicaMental
- item – convocacao
- descricao – "O usuário deverá informar ao sistema quem realizará a avaliação de saúde física mental."

PERGUNTA:

- id – ch_QuemRealizaraAvaliacaoSaudeFisicaMental
- tipo – fraseComArtigo
- texto – "Quem realizará a avaliação de saúde física mental?"

VARIÁVEL:

- id – ch_PrazoJuntadaDosDocumentos
- item – convocacao
- descricao – "O usuário deverá informar ao sistema qual será o prazo (em dias) para que o candidato reúna a documentação necessária para a nomeação. Exemplo: 5."

PERGUNTA:

- id – ch_PrazoJuntadaDosDocumentos
- tipo – numero
- texto – "Qual é o prazo para que o candidato reúna a documentação necessária para a nomeação?

NOMEAÇÃO (nomeacao)

ITEM:

- id – nomeacao
- sequencia – 12
- nome – DA NOMEAÇÃO
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_PrazoPosseCargo
- item – nomeacao

- descricao – "O usuário deve informar ao sistema qual é o prazo (em dias) que o candidato terá para tomar posse do cargo. Exemplo: 5."

PERGUNTA:

- id – ch_PrazoPosseCargo
- tipo – numero
- texto – "Qual é o prazo que o candidato terá para tomar posse do cargo?

VARIÁVEL:

- id – ch_LocalNomeacao
- item – nomeacao
- descricao – "O usuário deve informar ao sistema qual é o local para a nomeação do candidato."

PERGUNTA:

- id – ch_LocalNomeacao
- tipo – fraseComArtigo
- texto – "Qual é o local para a nomeação do candidato?

VARIÁVEL:

- id – ch_QuemRealizaraAvaliacaoSaudeFisicaMental
- item – convocacao
- descricao – "O usuário deverá informar ao sistema quem realizará a avaliação de saúde física mental."

PERGUNTA:

- id – ch_QuemRealizaraAvaliacaoSaudeFisicaMental
- tipo – fraseComArtigo
- texto – "Quem realizará a avaliação de saúde física mental?"

DISPOSIÇÕES GERAIS (dipoGerais)

ITEM:

- id – dispoGerais
- sequencia – 14
- nome – DAS DISPOSIÇÕES GERAIS
- negrito – 1
- status – nil

VARIÁVEL:

- id – ch_PrazoValidadeConcurso
- item – dispoGerais
- descricao – "O usuário deve informar qual é prazo de validade do Concurso. Exemplo: dois anos."

PERGUNTA:

- id – ch_PrazoValidadeConcurso
- tipo – frase
- texto – "Entre com o prazo de validade do concurso."

VARIÁVEL:

- id – ch_QuemProrrogaConcurso
- item – dispoGerais
- descricao – "O usuário deve informar ao sistema a critério de quem fica prorrogação do prazo de validade do concurso."

PERGUNTA:

- id – ch_QuemProrrogaConcurso
- tipo – fraseComArtigo
- texto – "Quem é responsável pela prorrogação do prazo de validade do concurso?

VARIÁVEL:

- id – ch_NomeacaoCondicionada
- item – dispoGerais
- descricao – "O usuário deve informar ao sistema quem é responsável pela concretização da nomeação condicionada."

PERGUNTA:

- id – ch_NomeacaoCondicionada
- tipo – fraseComArtigo
- texto – "Quem é responsável pela concretização da nomeação condicionada?

VARIÁVEL:

- id – ch_LocalManterEnderecoAtualizado
- item – dispoGerais
- descricao – "O usuário deve informar ao sistema em qual local o candidato deverá manter o seu endereço atualizado."

PERGUNTA:

- id – ch_LocalManterEnderecoAtualizado
- tipo – fraseComArtigo
- texto – "Qual é o local onde o condidato deverá manter o seu endereço atualizado?

VARIÁVEL:

- id – ch_LocalDivulgacaoResultado
- item – dispoPreliminares
- descricao – "O sistema necessita saber qual é o local de divulgação do resultado. Exemplo: Órgão de Imprensa do Município."

PERGUNTA:

- id – ch_LocalDivulgacaoResultado
- tipo - fraseComArtigo
- texto – "Entre com o local para a publicação do resultado."

VARIÁVEL:

- id – ch_PrazoRetiradaDocumentos
- item – dispoGerais
- descricao – "O usuário deve informar ao sistema qual é o prazo, após a homologação do resultado, para a retirada, por parte do contratante, dos materiais utilizados no concurso. Exemplo: seis meses."

PERGUNTA:

- id – ch_PrazoRetiradaDocumentos
- tipo – frase
- texto – "Qual é o prazo para a retirada dos materiais utilizados no concurso?

VARIÁVEL:

- id – ch_LocalDivulgacaoResultado
- item – dispoPreliminares
- descricao – "O sistema necessita saber qual é o local de divulgação do resultado. Exemplo: Órgão de Imprensa do Município."

PERGUNTA:

- id – ch_LocalDivulgacaoResultado
- tipo - fraseComArtigo

- texto – "Entre com o local para a publicação do resultado."

QUESTIONÁRIO META

METAVARIÁVEL:

- id – ch_AbrangênciaEdital
- descricao – "O sistema necessita saber qual é a abrangência do concurso, se é municipal, estadual ou federal. O sistema utilizará a resposta para chegar à algumas conclusões. Por exemplo, concursos com abrangência federal ou estadual deverão possuir vagas reservadas para afro-descendentes e deficientes físicos. Além disto a abrangência do edital irá definir o tipo da lei que irá regulamentar o provimento efetivo de cargos."

PERGUNTA:

- id – ch_AbrangênciaEdital
- tipo – escolha
- texto – "Entre com o tipo do edital. (1 – Municipal, 2 – Estadual, 3 – Federal)"
- teste – 1 2 3 -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaIntroducao
- descricao – "O sistema necessita saber se item Introdução será necessário no edital. A resposta deve ser fornecida na forma s ou n. Exemplo de resposta: s "

PERGUNTA:

- id – ch_TestaIntroducao
- tipo – escolha
- texto – "O item Introdução é necessário? (s ou n)"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaDispoPreliminares
- descricao – "O sistema necessita saber se você deseja acrescentar o item Disposições Preliminares ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_ TestaDispoPreliminares
- tipo – escolha
- texto – "O item Disposições Preliminares é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaAfroDescendentes
- descricao – "O sistema necessita saber se o concurso abrirá vagas para afro-descendentes. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaAfroDescendentes
- tipo – escolha
- texto – "Há vagas para afro-descendentes? "
- teste – s n -1 -2 -3 -4 -5

METAVARIÁVEL:

- id – ch_TestaDeficientesFisicos
- descricao – "O sistema necessita saber se o concurso abrirá vagas para pessoas com deficiência física. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaDeficientesFisicos
- tipo – escolha
- texto – "Há vagas para pessoas com deficiência?"
- teste – s n -1 -2 -3 -4 -5

METAVARIÁVEL:

- id – ch_TestaCargos
- descricao – "O sistema necessita saber se você deseja acrescentar o item Cargos ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaDeficientesCargos
- tipo – escolha
- texto – "O item Cargos é necessário?"
- teste – s n -1 -2 -3 -4 -5

METAVARIÁVEL:

- id – ch_TestaInvestidura
- descricao – "O sistema necessita saber se você deseja acrescentar o item Investidura ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaInvestidura
- tipo – escolha
- texto – "O item Investidura é necessário?"
- teste – s n -1 -2 -3 -4 -5

METAVARIÁVEL:

- id – ch_TestaInscricao
- descricao – "O sistema necessita saber se você deseja acrescentar o item Inscrição ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaInscricao
- tipo – escolha
- texto – "O item Inscrição é necessário?"
- teste – s n -1 -2 -3 -4 -5

METAVARIÁVEL:

- id – ch_TestaProvas
- descricao – "O sistema necessita saber se você deseja acrescentar o item Provas ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaProvas
- tipo – escolha
- texto – "O item Provas é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaRecursos
- descricao – "O sistema necessita saber se você deseja acrescentar o item Recursos ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaRecursos
- tipo – escolha
- texto – "O item Recursos é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaClassificacao
- descricao – "O sistema necessita saber se você deseja acrescentar o item Classificacao ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaClassificacao
- tipo – escolha
- texto – "O item Classificação é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaResultado
- descricao – "O sistema necessita saber se você deseja acrescentar o item Resultado ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaResultado
- tipo – escolha
- texto – "O item Resultado é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaConvocacao
- descricao – "O sistema necessita saber se você deseja acrescentar o item Convocação ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaConvocacao
- tipo – escolha
- texto – "O item Convocação é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaNomeacao
- descricao – "O sistema necessita saber se você deseja acrescentar o item Nomeação ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_TestaNomeacao
- tipo – escolha
- texto – "O item Nomeação é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6

METAVARIÁVEL:

- id – ch_TestaDispoGerais
- descricao – "O sistema necessita saber se você deseja acrescentar o item Disposições Gerais ao edital. Em caso positivo o sistema fornecerá posteriormente um questionário específico para este item."

PERGUNTA:

- id – ch_ TestaDispoGerais
- tipo – escolha
- texto – "O item Disposições Preliminares é necessário?"
- teste – s S n N -1 -2 -3 -4 -5 -6