Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Sistema Multiagente para Implementar Transparência na Distribuição de Processos Judiciais Eletrônicos

Denis José Sousa de Albuquerque

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Orientadora
Prof.ª Dr.ª Célia Ghedini Ralha

Coorientadora
Dr.ª Vanessa Tavares Nunes

Brasília
2016

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Curso de Computação — Licenciatura

Coordenador: Prof. Dr. Pedro Antonio Dourado Rezende

Banca examinadora composta por:

Prof.ª Dr.ª Célia Ghedini Ralha (Orientadora) — CIC/UnB
Prof. Dr. Rodrigo Bonifácio de Almeida — CIC/UnB
Dr.ª Vanessa Tavares Nunes — Pós-Doc PPGInf/UnB

CIP — Catalogação Internacional na Publicação

Albuquerque, Denis José Sousa de.

Sistema Multiagente para Implementar Transparência na Distribuição de Processos Judiciais Eletrônicos / Denis José Sousa de Albuquerque. Brasília : UnB, 2016.
106 p. : il. ; 29,5 cm.

Monografia (Graduação) — Universidade de Brasília, Brasília, 2016.

1. Sistemas Multiagentes, 2. Sistemas de Informação, 3. Distribuição de Processos Judiciais, 4. Transparência, 5. Governo Eletrônico, 6. Tribunal Superior do Trabalho, 7. JADE, 8. Tropos, 9. Drools.

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Sistema Multiagente para Implementar Transparência na Distribuição de Processos Judiciais Eletrônicos

Denis José Sousa de Albuquerque

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Prof.ª Dr.ª Célia Ghedini Ralha (Orientadora)
CIC/UnB

Prof. Dr. Rodrigo Bonifácio de Almeida
CIC/UnB

Dr.ª Vanessa Tavares Nunes
Pós-Doc PPGInf/UnB

Prof. Dr. Pedro Antonio Dourado Rezende
Coordenador do Curso de Computação — Licenciatura

Brasília, 24 de junho de 2016

Dedicatória

Este trabalho é dedicado aos meus amados pais, Pedro e Maria Lucia.

Agradecimentos

Agradeço

Em primeiro lugar, à minha esposa Mirella pelo amor, compreensão e apoio em todos os momentos.

À querida professora Dr.ª Celia Ghedini Ralha pelo incentivo, orientação e confiança.

À amiga Dr.ª Vanessa Tavares Nunes pelas valiosas colaborações e parcerias.

Ao professor Dr. Rodrigo Bonifácio de Almeida pelas aulas sempre motivadoras para o estudo de linguagens de programação.

Aos demais professores do Departamento de Ciência da Computação da UnB com os quais pude conviver durante esses anos de curso.

Aos muitos amigos do Tribunal Superior do Trabalho que direta ou indiretamente contribuíram para este trabalho.

Resumo

Neste trabalho descrevemos o projeto e a implementação de um sistema multiagente para realização das atividades de distribuição de processos judiciais eletrônicos. A distribuição é o ato em que se define o Órgão Jurisdicional responsável por conduzir e julgar um determinado processo judicial, sendo um dos elementos fundamentais para o bom andamento do processo judicial e a imparcialidade das decisões. Entretanto, atualmente são relatados problemas como a ausência de transparência nas atividades de distribuição de processos.

A transparência é um requisito essencial para oferecer um serviço de qualidade aos cidadãos e organizações que recorrem ao Poder Judiciário buscando a garantia de seus direitos. Foi estudado o caso do Tribunal Superior do Trabalho (TST), identificando como se dá a distribuição dos processos ajuizados naquela corte. O processo de trabalho foi mapeado e, seguindo a metodologia de desenvolvimento Tropos, foram levantados os requisitos de um sistema multiagente para tratar da questão.

O projeto da solução inclui desde a análise e definição dos requisitos iniciais até a implementação, utilizando o *middleware* JADE de sistemas multiagentes com linguagem de programação Java. O raciocínio dedutivo dos agentes foi implementado utilizando o formalismo de regras de produção com o motor de inferência JBoss Drools. Foram desenvolvidas interfaces gráficas para operação do sistema e uma aplicação *web* para implementar características de transparência na consulta de informações relacionadas à distribuição de processos judiciais. Uma base de dados com informações de mais de 300 mil processos judiciais foi construída, de forma semelhante a encontrada no TST, para fins de validação da solução.

Palavras-chave: Sistemas Multiagentes, Sistemas de Informação, Distribuição de Processos Judiciais, Transparência, Governo Eletrônico, Tribunal Superior do Trabalho, JADE, Tropos, Drools.

Abstract

This work describes the design and implementation of a multi-agent system to carry out the electronic lawsuits distribution activities. Distribution is the act that defines the Jurisdictional Authority responsible for conducting and judging a court case. It is one of the key elements for the smooth running of the judicial process and the impartiality of decisions. However, there are problems, currently reported, such as lack of transparency in the distribution process activities.

Transparency is an essential requirement to provide a quality service to citizens and organizations who seek the courts to guarantee their rights. We studied the case of the Superior Labor Court (TST), identifying how the distribution of lawsuits is performed in that court. The working process has been mapped and, following the Tropos development methodology, the requirements of a multi-agent system to address the issue were raised.

The solution's project includes various the phases, from the initial requirements definition up to the implementation, using the middleware JADE for multi-agent systems with the Java programming language. Deductive reasoning agents were implemented using the formalism of production rules with JBoss Drools' inference engine. In addition, graphical interfaces were developed for system operation and a web application to implement transparency characteristics in the query of information related to the distribution of lawsuits. A database with more than 300 thousand lawsuits information was constructed, similarly to what is found in the TST, for solution validation.

Keywords: Multiagent Systems, Information Systems, Distribution of Lawsuits, Transparency, e-Government, Superior Labor Court, JADE, Tropos, Drools.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Este trabalho apresenta o projeto e a construção de um Sistema Multiagente (SMA), utilizando a metodologia de desenvolvimento Tropos e o *framework* JADE (*JAVA Agent Development Framework*). O Sistema foi denominado LawDisTrA (*Lawsuit Distribution with Transparent Agents*), pois tem o objetivo de realizar a distribuição de Processos Judiciais Eletrônicos segundo requisitos de transparência.

De acordo com Jennings [19], o paradigma de agentes é adequado para a análise, o projeto e a construção de soluções de *software* para sistemas complexos, pois agentes representam um mecanismo de abstração natural para decompor e organizar tais sistemas.

Castro, Kolp e Mylopoulos [9] ressaltam que sistemas de informação corporativos possuem um ambiente operacional melhor entendido em termos de agentes, responsabilidades, objetivos, tarefas e recursos, mas a sua construção é geralmente concebida como uma coleção de programas, módulos, estruturas de dados e interfaces. É argumentado que esse descasamento entre o ambiente operacional das empresas e seus sistemas de informação seria um dos fatores para a baixa qualidade desses sistemas, bem como o frequente fracasso em projetos de desenvolvimento. Para resolver essas questões, os sistemas de informação corporativos deveriam ser organizados da maneira semelhante a organização das empresas.

Considerando que a tarefa de distribuição de processos judiciais nos dias atuais é, via de regra, realizada de forma automatizada por sistemas de informação em razão do advento dos chamados processos eletrônicos, um sistema orientado a agentes será desenvolvido de forma a identificar se essa abordagem é adequada para a solução do problema. Para tanto, foi utilizada a metodologia Tropos [5], uma metodologia de desenvolvimento dirigida por requisitos que busca dar suporte a diversas fases do desenvolvimento de sistemas baseados em agentes.

1.1 Problema

Sistemas de informação atualmente empregados na realização da distribuição eletrônica de processos judiciais apresentam falhas relacionadas a fraudes, eficiência e transparência [11, 21, 27]. A transparência nos atos da distribuição de processos judiciais é importante para que a sociedade possa fiscalizar se as regras estabelecidas na legislação estão sendo cumpridas pelo Poder Judiciário.

1.2 Motivação

De um modo geral, com o advento do Processo Judicial Eletrônico, a distribuição de processos ocorre mediante sorteio e é realizada de forma automatizada por computadores. Mas, de acordo com Lima [21], "apesar de a regra processual da livre distribuição ser de caráter cogente e de fácil aplicação, ela é violada diariamente, de forma velada ou às escâncaras".

Deseja-se burlar a distribuição principalmente quando o interessado na causa conhece previamente o entendimento de um juiz sobre determinada matéria, de forma que a vitória seja facilitada no caso do processo ser direcionado ao juízo de sua preferência. Lima [21] afirma que "advogados inescrupulosos, que fazem de tudo para ganhar a causa de seu cliente, sem qualquer crise de consciência, não hesitarão em fraudar a distribuição, se isso lhes propiciar a vitória na demanda". Diz ainda que "a maneira mais abominável de se malograr a livre distribuição é através da violação ao sistema de dados. Para a perpetração do ilícito, é necessário obter acesso à manipulação dos dados cadastrais, geralmente por meio de um funcionário do setor de distribuição (...) a fraude ao sistema de processamento de dados ocorre, na grande maioria das vezes, sem o conhecimento do juiz. Torna-se difícil, portanto, a sua repressão pelo magistrado a quem o processo foi distribuído".

A ocorrência desse tipo de fraude foi noticiada[1] pelo Conselho Nacional de Justiça (CNJ) em 11 de fevereiro de 2014, ocasião em que o órgão decidiu pela demissão de quatro servidores do Tribunal de Justiça do Estado do Maranhão em razão de fraude na distribuição de processos. Na ocasião, o relator do Processo Administrativo Disciplinar afirmou que "os servidores direcionavam processos para determinados juízos, contrariando as regras da distribuição por sorteio entre os juízos de mesma competência, em total violação ao princípio do juiz natural e às regras de competência de distribuição constantes do Código de Processo Civil".

[1] http://www.cnj.jus.br/noticias/cnj/61309-plenario-decide-pela-demissao-de-quatro-servidores-do-tjma-por-fraude-na-distribuicao-de-processos

Lima [21] relaciona diversas formas de interferência para violação a regra da livre distribuição. Um exemplo é o ingresso de ações em diversos estados da Federação, mesmo naqueles em que a parte autora não possui domicílio ou grafando propositalmente o nome das partes de forma diferente, de forma que o sistema não identifique a existência de outro processo relacionado. Outro exemplo é quando ocorre o ajuizamento de ações sucessivas cujos efeitos jurídicos pretendidos são os mesmos na prática e que serão distribuídas para dois juízes diferentes. "Caso um dos juízes defira o pedido liminar, a parte pede a desistência da outra ação, prosseguindo tão-somente o feito no juízo favorável ao autor" [21].

Caso semelhante ocorreu recentemente quando deputados federais impetraram mandado de segurança junto ao Supremo Tribunal Federal (STF) contra ato do presidente da Câmara dos Deputados. Ao identificar que o processo foi distribuído para relator que, em tese, teria posição contrária ao pleiteado, peticionaram requerendo a desistência da tramitação deste feito. O Ministro Relator em sua decisão escreveu: "Os impetrantes sequer disfarçam a tentativa de burlar o princípio do juiz natural e as regras atinentes à competência, em atitude flagrantemente ilegal, com a desistência imediatamente posterior à ciência do relator a quem foi distribuída a demanda. (...) Tal atitude configura-se como clara fraude à distribuição processual e constitui ato temerário e ofensivo, não a essa relatoria, mas ao Poder Judiciário. (...) Ninguém pode escolher seu juiz de acordo com sua conveniência, razão pela qual tal prática deve ser combatida severamente por esta Corte, de acordo com os preceitos legais pertinentes" [27].

Para o CNJ [11], "não é incomum constatar nos sistemas processuais ou de distribuição hoje existentes falhas que potencializam a emergência de desvios, sejam eles dolosos ou culposos. Redução da distribuição de um juízo em detrimento de outros, ausência de informações transparentes sobre a distribuição, omissão de dados relevantes sobre a distribuição, escolha de critérios que levam a pouca ou nenhuma aleatoriedade, entre outros, são alguns dos problemas que podem ser constatados".

Além de problemas relacionados a fraudes, advogados reclamam também da falta de transparência relacionada a distribuição de processos. Conforme relata o especialista em direito digital Alexandre Atheniense, em notícia[2] publicada em 12 de julho de 2014 no portal Consultor Jurídico, "é uma caixa preta os critérios utilizados para a distribuição de processos. Não traria nenhuma problema para a segurança nacional mostrar, por exemplo, quais os critérios para a distribuição dos processos. O Judiciário parte da premissa que nós devemos confiar sem podermos fazer qualquer tipo de controle".

Por fim, outra questão diz respeito a eficiência da distribuição. Se o processo ou sistema for ineficiente, a celeridade de tramitação do processo é prejudicada. Por exemplo, para

[2]http://www.conjur.com.br/2014-jul-12/advogados-exigem-transparencia-relacao-processo-eletronico

colocar em dia a distribuição de processos no STF, o presidente da corte chegou a criar uma força-tarefa com 50 servidores trabalhando inclusive aos sábados e domingos[3].

1.3 Objetivos

Busca-se projetar e construir um SMA que atenda os requisitos da distribuição de processos judiciais eletrônicos e trate os problemas identificados em soluções empregadas atualmente, especialmente quanto a ausência de transparência da informação, colaborando com os requisitos de combate à fraudes e manutenção da eficiência do processo.

Como objetivos específicos do trabalho citamos:

- Identificar os requisitos de um sistema para distribuição de processos judiciais;
- Realizar estudo de caso acerca da distribuição de processos empregada atualmente no Tribunal Superior do Trabalho (TST);
- Identificar vantagens e desvantagens no uso de sistema baseado em agentes para construção de sistemas de informação corporativos;
- Realizar experimentações para verificar a eficácia e a eficiência da solução desenvolvida;
- Discutir características relacionadas à transparência da informação e do processo para um sistema que trate da distribuição de processos judiciais.

1.4 Hipóteses

Como hipóteses deste trabalho podemos citar:

- O paradigma de agentes é adequado para a construção de sistemas de informação, podendo substituir meios mais tradicionais;
- Os requisitos funcionais e de qualidade – tais como confiabilidade e auditabilidade – exigidos por um sistema de distribuição de processos judicias serão atendidos mediante uso de transparência na interação entre agentes;
- A metodologia Tropos é adequada para modelar a decomposição do problema segundo o paradigma de agentes;
- O uso do *framework* JADE facilita o desenvolvimento de sistemas multiagentes, além de preservar características como flexibilidade e portabilidade da solução.

[3]http://www.stf.jus.br/portal/cms/verNoticiaDetalhe.asp?idConteudo=272449

1.5 Metodologia

A realização do trabalho segue um processo metodológico que preocupa-se com a concepção e a construção de artefatos tecnológicos para a solução de problemas organizacionais, permitindo sua posterior avaliação e análise. Nesse sentido, a pesquisa busca avaliar e analisar as características da construção de um SMA para a distribuição de processos judiciais. A metodologia pode ser entendida como um processo constituído de cinco fases, conforme apresentado na Figura 1.1:

1. Percepção do problema: investigação acerca da forma como é atualmente realizada a distribuição de processos judiciais pelos Tribunais do Poder Judiciário brasileiro, identificando suas principais características e problemas;

2. Análise da teoria: estudo da teoria de sistemas multiagentes, legislação relacionada a distribuição de processos judiciais e da transparência da informação e de processos;

3. Modelagem da solução: projeto do SMA com levantamento dos requisitos e modelagem do *software*;

4. Construção de artefato tecnológico: desenvolvimento do software projetado para atender os requisitos identificados;

5. Avaliação e análise: validação da arquitetura, análise e avaliação da solução desenvolvida para posterior crítica e evolução.

As fases de modelagem, construção e avaliação são repetidas de forma cíclica até que se obtenha uma solução satisfatória, adotando uma abordagem iterativa e incremental para o desenvolvimento da solução.

1.6 Apresentação do manuscrito

Este trabalho está dividido em cinco capítulos. No Capítulo 2 será apresentada a fundamentação teórica relativa aos sistemas multiagentes, à distribuição de processos judiciais e à transparência da informação. Os requisitos de um sistema informatizado para a distribuição são apresentados pela identificação de pontos importantes na legislação e de um estudo de caso.

No Capítulo 3, é apresentado o projeto do SMA, identificando os requisitos da solução, sua arquitetura e os modelos dos bancos de dados. Questões relativas a implementação do sistema e as ferramentas utilizadas também são apresentadas nesse capítulo.

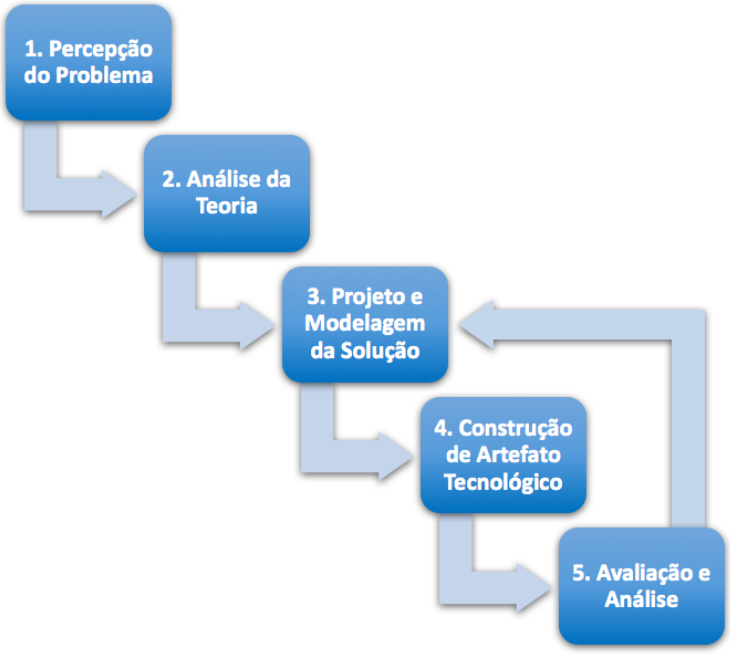

Figura 1.1: Processo Metodológico utilizado.

Os testes realizados e os resultados alcançados são discutidos no Capítulo 4. O Capítulo 5 conclui o trabalho e relaciona os trabalhos futuros. O Anexo I destaca os trechos do Regimento Interno do TST que identificam requisitos para a distribuição de processos. O Anexo II apresenta o dicionário de dados dos bancos de dados utilizados na experimentação e o Anexo III contém os *scripts* de criação das estruturas desses bancos de dados.

Capítulo 2

Fundamentação

O desenvolvimento do presente trabalho passa pelo estudo de conceitos nas áreas de conhecimento da Computação, do Direito e da Transparência, além da integração de diversas tecnologias para construção do SMA. Neste capítulo serão apresentados os conceitos nas áreas de conhecimento citadas.

2.1 Sistemas multiagentes

Para Wooldridge [34], um SMA é aquele que possui dois ou mais agentes que interagem uns com os outros, geralmente por meio de troca de mensagens em uma de rede de computadores, de forma a atingir um objetivo global.

Nesta seção, conceitos relativos aos sistemas multiagentes serão apresentados.

2.1.1 Agentes

Norvig e Russell [23] definem agente como uma entidade capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores. A ação do agente é escolhida de acordo com as condições percebidas do ambiente, onde a inteligência se dá pela escolha de ações mais adequadas para alcançar um objetivo.

Há ainda diversas outras definições para agentes de *software* na literatura, dentre as quais destacamos:

- um agente é um sistema computacional encapsulado que está situado em algum ambiente e é capaz de ação flexível autônoma neste ambiente, a fim de alcançar seus objetivos de projeto [34];
- agentes são unidades de software que podem lidar com mudanças ambientais e com os vários requisitos de redes abertas através de características como autonomia, mobilidade, inteligência, cooperação e reatividade [29].

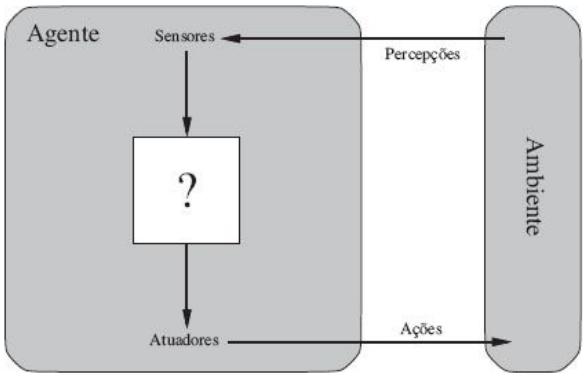

Figura 2.1: Agente ilustrado por Norvig e Russell [23].

No que se refere ao comportamento do agente, para Norvig e Russell [23] é possível classificá-lo em quatro tipos:

- Agente reativo simples: sua ação depende unicamente da percepção atual, ignorando o histórico de percepções e ações;
- Agente reativo baseado em modelos: o agente possui um modelo de como o ambiente funciona e mantém algum tipo de estado do mundo baseado no histórico de percepções, de forma que sua ação depende não só de sua percepção atual, mas também desse estado do mundo;
- Agente baseado em objetivos: possuem informações que descrevem situações desejáveis, utilizando-as para a tomada de decisões;
- Agente baseado em utilidade: utilizam uma função de utilidade para determinar qual ação resultará em um estado do mundo mais desejável e, dessa forma, escolher a ação a ser executada.

Os agentes podem ainda ser classificados como agentes com aprendizagem quando utilizam realimentação crítica sobre como o agente está funcionando para determinar de que maneira a escolha das ações pode ser modificada para a escolha de melhores ações no futuro [23].

O projeto de um agente inteligente passa pela identificação de quais são suas percepções, suas ações, seus objetivos e seu ambiente. A descrição desse projeto pode ser identificada como P.E.A.S. (*Performance, Environment, Actuators, Sensors*) ou P.A.G.E. (*Perceptions, Actions, Goals, Environment*). Para Norvig e Russell [23], ao projetar um agente, a primeira etapa deve ser especificar o ambiente de tarefa de forma tão completa quanto possível, sendo possível caracterizá-los como:

- Completamente observável ou parcialmente observável;

- Determinístico ou estocástico;
- Episódico ou sequencial;
- Estático ou dinâmico;
- Discreto ou contínuo;
- Conhecido ou desconhecido;
- Agente único ou multiagente.

2.1.2 Agentes Lógicos

Agentes baseados em conhecimento utilizam mecanismos de inferência lógica para derivar novas informações sobre o mundo a partir do conhecimento presente, usando essas derivações para deduzir suas ações. Esses agentes são compostos por uma base de conhecimento e um mecanismo de inferência. A inferência é o processo de derivação de novas sentenças a partir de sentenças antigas [23].

Para Norvig e Russell [23] o componente central desse tipo de agente é sua base de conhecimento (*knowledge base* – KB). Essa base é um conjunto de sentenças lógicas e fatos que representam asserções sobre o mundo. O agente incorpora suas percepções na base de conhecimento e consulta essa base para escolher a melhor ação a ser tomada, registrando a ação tomada também na base de conhecimento (Código 2.1).

```
1 função AGENTE-KB(percepção) retorna uma ação
2   persistente: KB, uma base de conhecimento
3                t, um contador, inicialmente igual a 0, indicando tempo
4   TELL(KB, CRIAR-SENTENÇA-DE-PERCEPÇÃO(percepção,t))
5   ação ← ASK(KB, CRIAR-CONSULTA-DE-AÇÃO(t))
6   TELL(KB, CRIAR-SENTENÇA-DE-AÇÃO(ação, t))
7   t ← t+1
8   retornar ação
```

Código 2.1: Função genérica para um agente baseado em conhecimento [23].

Agentes baseados em conhecimentos atuam de forma semelhante a sistemas especialistas. Um sistema especialista é aquele que é capaz de resolver problemas ou fornecer conselhos em algum domínio rico em conhecimento [18]. Um clássico exemplo de sistema especialista é o MYCIN, que foi projetado para auxiliar médicos no tratamento de infecções humanas. MYCIN funciona em um processo de interação com o usuário de forma a obter um conjunto de fatos simbolicamente representados que o sistema então utiliza para derivar conclusões.

Segundo Wooldridge [34], a mais importante diferença entre agentes e sistemas especialistas é que sistemas como o MYCIN não interagem diretamente com o ambiente, ou seja, elas não obtêm suas informações através de sensores, mas mediante a interação com o usuário. Além disso, em geral, sistemas especialistas não são capazes de se comunicar e interagir com outros agentes (cooperar ou competir).

Representação do conhecimento

O conhecimento dos agentes precisa de uma linguagem de representação simbólica para a elaboração das sentenças e regras que compõem a base de conhecimento. Ao processo de construção da base de conhecimento dá-se o nome de engenharia do conhecimento.

Norvig e Russell [23] definem o engenheiro de conhecimento como alguém que investiga um domínio específico, aprende quais conceitos são importantes nesse domínio e cria uma representação formal dos objetos e relações no domínio.

O processo de engenharia de conhecimento inclui etapas [23] de: identificação da tarefa; agregação do conhecimento relevante; definição de um vocabulário (ontologia) de predicados, funções e constantes; codificação do conhecimento geral sobre o domínio; formulação de consultas aos procedimentos de inferência e obtenção de respostas; depuração da base de conhecimento. Nesse processo, é preciso o envolvimento de especialistas no domínio, para fins de extrair e representar o conhecimento em um conjunto bem definido de sentenças e regras.

Uma linguagem de representação de conhecimento é definida por sua sintaxe – estrutura das sentenças – e sua semântica – define o valor de verdade de cada sentença. No campo da lógica computacional, a lógica proposicional e a lógica de primeira ordem são formalismos lógicos utilizados como linguagens de representação de conhecimento.

Raciocínio lógico baseado em regras

A aplicação de regras de inferência nas sentenças da base de conhecimento pode ser usada para derivar provas – uma cadeia de conclusões que conduzem ao objetivo desejado [23].

Inferência lógica pode ser alcançada por meio de algoritmos do tipo encadeamento para frente (*forward chaining*) e encadeamento para trás (*backward chaining*). De acordo com Norvig e Russell [23], ambos os algoritmos são naturais e, por isso, as etapas de inferência são óbvias e fáceis para as pessoas seguirem, onde:

- no encadeamento para frente, a inferência é iniciada a partir de fatos já conhecidos (registrados na base de conhecimento) e se todas as premissas de uma implicação forem conhecidas, sua conclusão será acrescentada ao conjunto de fatos conhecidos.

Segundo Norvig e Russell [23], o encadeamento para frente é um exemplo de raciocínio orientado a dados (raciocínio em que o foco da atenção começa com os dados conhecidos). Esse tipo de raciocínio pode ser usado em um agente para derivar conclusões a partir de percepções de entrada.

- no encadeamento para trás, a inferência é iniciada a partir do objetivo, encadeando regras até encontrar fatos conhecidos que apoiem a prova. O encadeamento para trás é uma forma de raciocínio orientado por metas [23], o tipo de inferência que é a base para a programação lógica.

Ferramentas

Motores de inferência baseados em regras podem empregar algoritmos de encadeamento para frente ou encadeamento para trás. Seu uso permite uma abordagem declarativa para processar a decisão inteligente do agente. Norvig e Russell [23] entendem que um agente bem-sucedido sempre deve combinar elementos declarativos e procedurais em seu projeto e que o conhecimento declarativo muitas vezes pode ser compilado em um código procedural mais eficiente. Existem alguns motores de inferência bem conhecidos na literatura, a saber:

- Drools[1] é um Sistema de Gerenciamento de Regras de Negócio (*Business Rule Management System – BRMS*) que disponibiliza um sistema de raciocínio híbrido (utiliza mecanismos de inferência tanto por encadeamento para frente quanto encadeamento para trás) implementado na linguagem Java. Seu motor de inferência é fortemente baseado no algoritmo Rete [6]. O algoritmo Rete, desenvolvido por Charles Forgy em 1974, foi concebido como um método eficiente para comparação de uma grande quantidade de padrões com uma grande coleção de objetos de forma a otimizar o casamento de padrões em sistemas de produções [14]. Uma regra de produção é uma estrutura de duas partes: condições e ações (Código 2.2). O motor de inferência casa os fatos da base de conhecimento com as condições de forma a inferir conclusões que resultem em ações.

```
1 rule ''Nome da regra''
2   when
3     <condições>
4   then
5     <ações>;
6 end
```

Código 2.2: Regra de produção em sintaxe utilizada pelo Drools.

[1] JBoss Drools – http://www.drools.org

- Jess[2] é um motor de inferência desenvolvido na linguagem Java. Assim como o Drools, Jess possibilita o uso de encadeamento para frente e encadeamento para trás, sendo também baseado no algoritmo Rete com melhorias para aumento de desempenho. Ele está disponível sem custos para o uso acadêmico, mas requer licenciamento específico para uso comercial.

- Prolog (*PROgrammation en LOGique*) é a linguagem de programação em lógica mais amplamente utilizada, com várias implementações livres e comerciais, além de interfaces para diversas outras linguagens de programação. Ela foi concebida no início da década de 1970, sendo uma das linguagem mais utilizadas na área de Inteligência Artificial. Muitos sistemas especialistas foram escritos em Prolog para os domínios jurídicos, médicos, financeiros, entre outros [23]. A execução de programas Prolog é feita por meio do encadeamento para trás, onde as cláusulas são experimentadas na ordem em que são escritas na base de conhecimento.

2.1.3 Características de SMA

Sycara [28] indica que um SMA é caracterizado por: (1) cada agente possui uma visão limitada, com informação e capacidade incompleta para resolver o problema; (2) não há controle global do sistema; (3) dados são descentralizados e (4) a computação é assíncrona.

Castro, Kolp e Mylopoulos [8] estabelecem diversas propriedades de agentes que precisam ser consideradas a fim de construir um *software* orientado a agente, são elas: autonomia, deliberatividade, reatividade, organização, socialização e interação.

Autonomia é a capacidade de agir de forma independente sem intervenção direta de humanos ou de outros agentes [36]. É uma propriedade inerente dos agentes que implica em um *software* mais robusto. De acordo com Yu [36], os agentes possuem iniciativa própria e não são necessariamente condescendentes com as demandas ou desejos externos.

Quanto a deliberatividade, para Castro, Alencar e Silva [8], na deliberatividade, um agente toma decisões considerando informações vindas do ambiente onde ele se situa, bem como as informações sobre experiências prévias.

A reatividade diz respeito a capacidade dos agentes perceberem seu ambiente e responderem aos estímulos externos, às mudanças que ocorrem nesse ambiente [8].

Organização está relacionada ao estabelecimento de um comportamento coeso para um grupo de agentes, onde o comportamento de cada um é restringido por um conjunto de normas sociais. Segundo Hübner e Sichman [17], a característica de organização é oposta à autonomia, porém muitas das propriedades desejadas nos sistemas multiagentes advêm do equilíbrio destas duas habilidades.

[2]Jess – http://herzberg.ca.sandia.gov

Socialização é a capacidade que os agentes possuem de cooperarem e coordenarem suas atividades com os demais agentes do ambiente. Os agentes são capazes de comunicar suas crenças, objetivos e planos uns com os outros.

A interação é a habilidade de se comunicar com o ambiente e com outros agentes [8]. Bellifemine et al. [2] afirmam que a comunicação é um dos componentes chaves dos SMA, uma vez que para atingir um objetivo, muitas vezes os agentes precisam ser capazes de se comunicarem com usuários, recursos ou outros agentes. A interação pode ser classificada como:

- cooperação – interação com outros agentes para atingir um objetivo comum;
- competição – interação com outros agentes onde o sucesso de um agente implica na falha de outros.
- coordenação – habilidade de executar alguma atividade em um ambiente compartilhado com outros agentes, determinando objetivos que eles compartilham e tarefas em comum, evitando conflitos desnecessários e gerenciando a utilização dos recursos [33];
- negociação – habilidade de interagir com outros agentes a fim de alcançar um acordo sobre algum problema [15].

2.1.4 Desenvolvimento de SMA

Para Castor e Castro [7], o paradigma de agentes envolve alguns desafios e obstáculos ao desenvolvimento dos sistemas, pois é difícil manter o equilíbrio entre comportamento proativo e reativo, considerando que os agentes precisam interagir continuamente com o ambiente e perseguir seus objetivos. Eles argumentam que para que o equilíbrio seja atingido é necessário que a tomada de decisão seja sensível ao contexto, o que resulta em um grau significativo de imprevisibilidade sobre quais objetivos o agente perseguirá, em quais circunstâncias e quais os métodos que serão usados.

Além das dificuldades inerentes ao próprio paradigma de agentes, algumas armadilhas surgem durante o desenvolvimento de sistemas multiagentes. Segundo Castor e Castro [7], a tecnologia de agentes representa uma inovação e uma importante maneira de conceituar e implementar *software*, mas é importante entender suas limitações, sendo uma armadilha apresentar com entusiasmo excessivo as soluções de agente ou falhar em entender onde agentes podem ser aplicados adequadamente.

Outra armadilha, segundo Castor e Castro [7], é o uso excessivo de inteligência artificial. Deve-se evitar a sobrecarga de técnicas experimentais como interfaces de linguagem natural, planejadores, provadores de teorema, sistemas de manutenção de razão, etc. Os

autores sugerem a construção de agentes com um mínimo de técnicas da inteligência artificial, considerando que alguns pesquisadores chegam a afirmar que, dependendo da aplicação, somente 1% de inteligência artificial é necessário.

Embora haja muitas aplicações adequadas ao uso de agentes, eles não são uma solução universal e há muitas aplicações para as quais os paradigmas convencionais de desenvolvimento de *software* são mais apropriados. É preciso entender quais casos de uso podem ser melhor atendidos com o uso de agentes, sendo um erro acreditar que usar agentes é a solução para todos os problemas (bala de prata). Para Castor e Castro [7], há bons argumentos que evidenciam que a tecnologia de agentes levará a melhorias no desenvolvimento de *software* distribuído complexo.

Para Wooldridge [34] existem vários fatores que devem ser considerados ao julgar a pertinência da abordagem orientada a agentes, são eles:

- ambiente aberto ou muito dinâmico, incerto ou complexo: em tais ambientes, sistemas capazes de ações autônomas e flexíveis são geralmente a única solução;
- agentes são uma metáfora natural: muitos ambientes são naturalmente modelados como sociedades de agentes, seja cooperando para resolver problemas complexos ou até competindo entre eles;
- dados, controle ou competência distribuída: em alguns ambientes, a distribuição dos dados, controle ou perícia significa que soluções centralizadas são extremamente difíceis de construir;
- sistemas legados: agentes podem ser utilizados como *wrappers* para componentes legados, possibilitando que eles possam se comunicar e cooperar com outros componentes de *software*.

Um SMA é naturalmente *software* com múltiplas linhas de controle (*multithreaded*). Para Castro e Castor [7], se há a necessidade de uma única linha de controle num sistema, então a utilidade de uma solução baseada em agentes deve ser seriamente questionada. Dessa forma, é importante entender as lições aprendidas das comunidades de sistemas concorrentes e distribuídos para lidar com os problemas inerentes aos programas *multithreaded*.

Deve-se ainda estar atento as boas práticas da engenharia de *software*, buscando utilizar técnicas testadas e confiáveis para dar assistência ao desenvolvimento do sistema. Agentes podem ser vistos como módulos utilizados para decompor o sistema, devendo atender as recomendações da engenharia de *software* para se beneficiar das vantagens relacionadas à programação modular.

Segundo Parnas [25], a modularização tem importância proporcional ao tamanho e complexidade do sistema desenvolvido. Ela busca aumentar a flexibilidade e a compre-

ensibilidade do *software*, permitindo reduzir o tempo de desenvolvimento e manutenção do sistema. A modularização permite que: (I) um módulo seja desenvolvido utilizando serviços de outro módulo, mas sem a necessidade de conhecer todo o comportamento de outro módulo; (II) um módulo seja modificado internamente ou até mesmo trocado sem a necessidade de adaptar todo o sistema; (III) o comportamento do sistema seja entendido a partir do estudo de um módulo por vez; (IV) programadores distintos possam desenvolver cada módulo simultaneamente com pouca necessidade de intercomunicação.

É importante encontrar um equilíbrio entre o número de agentes na plataforma e a quantidade de objetivos pretendidos. É recomendável evitar o uso de agentes para tudo, pois a sobrecarga gerada para gerenciar os agentes e a comunicação entre eles pesará mais do que os benefícios esperados. Mas também deve-se evitar concentrar muitos objetivos em poucos agentes, pois dessa forma o sistema não se beneficiará das vantagens do paradigma distribuído.

Diversas metodologias específicas para apoiar o desenvolvimento de sistemas baseados em agentes já foram propostas, tais como: Tropos [5], Prometheus [24], Gaia [35] e Desire [4]. Wooldridge [34] apresenta as características de várias dessas metodologias, a abordagem predominante é a adaptação de metodologias desenvolvidas para a análise e o projeto orientado a objetos. Nesse trabalho, optou-se por aplicar a metodologia Tropos no desenvolvimento da solução proposta, uma vez que sua proposta inclui diversas fases de desenvolvimento através de uma abordagem orientada a requisitos, modelos e objetivos, adequada com o tipo de SMA proposto.

2.1.5 Metodologia Tropos

Tropos é uma metodologia dirigida por requisitos que busca dar suporte a diversas fases do desenvolvimento de sistemas baseados em agentes. Ela é fundamentada na identificação de requisitos e elaboração de modelos.

Os modelos são construídos para capturar as intenções dos interessados (usuários, proprietários, etc.) e adotam o *framework* I* (ISTAR – *Intentional STrategic Actor Relationships modelling*) para representar os conceitos de atores (agentes, posições ou papéis), objetivos, tarefas, recursos e suas interdependências [36]. O *framework* I* define um conjunto de símbolos para diagramação dos modelos (Figura 2.2), bem como formas para identificar a dependência (Figura 2.3) entre os componentes.

Construir um SMA seguindo a metodologia Tropos, envolve a divisão do processo de desenvolvimento em cinco fases, quais sejam:

- Requisitos Iniciais: nesta fase, busca-se o entendimento do problema através da compreensão do contexto organizacional no qual o sistema a ser desenvolvido será

Figura 2.2: Componentes gráficos do *framework* I* [22].

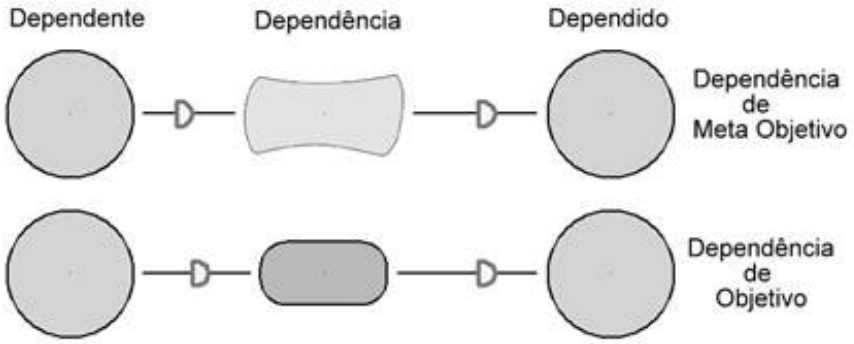

Figura 2.3: Dependência entre componentes [22].

implantado. Durante essa fase, os engenheiros de requisitos modelam os interessados como atores sociais que dependem uns dos outros e que têm intenções de atingir objetivos, realizar tarefas e fornecer recursos [8].

- Requisitos Finais: esta fase está preocupada com o refinamento dos modelos produzidos na fase anterior. Os componentes devem ser detalhados para apresentar as razões que estão por trás de suas dependências. Suas tarefas e objetivos precisam ser revisados, analisados e detalhados através de ligações de decomposição e de meios-fins [8].

- Projeto Arquitetural: busca-se definir a arquitetura do sistema, modelando sua estrutura em termos de subsistemas, interconectados através de fluxos de controle de dados. Em Tropos, subsistemas são representados como atores e interconexões são representadas como dependências entre os atores [8].

- Projeto Detalhado: nesta fase é introduzido detalhe adicional a cada componente arquitetural do sistema [8]. Envolve o uso de plataformas de desenvolvimento específicas e depende das características da linguagem de programação adotada. Assim, esta etapa geralmente está estritamente relacionada com as escolhas da implementação. Para documentar o projeto, pode-se ainda incluir a especificação da estrutura,

comunicação e comportamento dos agentes adotando modelos UML, utilizando diagramas de classe, sequência e comunicação.

- Implementação: a fase de implementação trata da construção do sistema projetado. Segundo Castro, Kolp e Mylopoulos [8], implementar sistemas multiagentes é inerentemente difícil. Para facilitar o desenvolvimento de SMA, diversas plataformas de implementação foram construídas, conforme veremos a seguir.

2.1.6 Ferramentas para SMA

Existem diversas ferramentas para apoio ao desenvolvimento de sistemas multiagentes, tais como: ABLE, JACK, JADE, JADEX, Jason, SeSam, SimAgent, ZEUS. Seu uso permite ao desenvolvedor concentrar-se na resolução do problema, simplificando o desenvolvimento do sistema na medida em que não será necessário ter que resolver questões complexas de infraestrutura. Passaremos a detalhar características de algumas das ferramentas citadas:

- ABLE[3] (*Agent Building and Learning Environment*): framework Java, biblioteca de componentes e ferramenta de produtividade, desenvolvido pelo *IBM T.J. Watson research laboratory*, para construção de agentes inteligentes com raciocínio e aprendizagem de máquina;

- JACK[4]: ambiente para construção e execução de sistemas multiagentes que foi desenvolvido para prover extensões orientadas a agentes para a linguagem de programação Java. Foi desenvolvido pelo *AOS Group* e aplica os conceitos do modelo BDI (*Beliefs, Desires, Intentions*) de forma a permitir a construção de SMA complexos;

- JADE[5] (*Java Agent DEvelopment Framework*): permite o desenvolvimento e execução de SMA na linguagem de programação Java em conformidade com a arquitetura padrão proposta por grupo ligado ao IEEE (*Institute of Electrical and Electronic Engineers*). JADE foi a ferramenta escolhida para apoiar o desenvolvimento da solução proposta neste trabalho e por isso será discutida em mais detalhes na seção seguinte.

- JADEX[6] (*JADE eXtension*): desenvolvido pela Active Components, fornece infraestrutura para implementação de agentes JADE segundo o modelo BDI.

[3]ABLE - http://researcher.watson.ibm.com/researcher/view$_group.php?id = 979$
[4]JACK - http://aosgrp.com/products/jack/
[5]JADE - http://jade.tilab.com/
[6]JADEX - https://www.activecomponents.org

JADE

Para Bellifemine et al. [2], o sucesso de sistemas multiagentes está fortemente relacionado com a disponibilidade de tecnologias que facilitem à implementação desses sistemas.

O JADE (*Java Agent DEvelopment framework*) é um *middleware* que objetiva simplificar o desenvolvimento de sistemas multiagentes ao mesmo tempo que garante a conformidade com padrões e fornece ferramentas que suportam a depuração e implantação do *software* [34].

Agentes JADE são implementados como *threads* Java, devendo ser uma instância da classe *jade.core.Agent*. A classe fornece todas as características e métodos necessários para realizar as interações básicas com a plataforma.

Os agentes devem realizar suas tarefas a partir da ativação de comportamentos, implementados em subclasses da classe *jade.core.behaviours.Behaviour*, e da troca de mensagens assíncronas com outros agentes disponíveis na plataforma.

O *framework* emprega um conjunto de serviços e agentes em conformidade com os padrões da FIPA (*Foundation For Intelligent Physical Agents*), além de suporte para linguagens de conteúdo e ontologias [2].

FIPA

A organização FIPA (*Foundation for Intelligent Physical Agents*) foi constituída com o objetivo de desenvolver e padronizar as tecnologias de sistemas multiagentes, incentivando o surgimento de *frameworks* que garantissem a interoperabilidade entre os sistemas. Ela definiu um conjunto de especificações para guiar a implementação de plataformas, sendo um modelo de referência que descreve o comportamento externo dos componentes do sistema. Ou seja, o padrão FIPA define um conjunto de serviços que precisam ser providos pelas plataformas, deixando em aberto detalhes relativos à estrutura interna e implementação [26].

O padrão FIPA define os serviços obrigatórios e opcionais de uma plataforma de agentes. Os serviços obrigatórios são o gerenciamento do ciclo de vida, o transporte de mensagens e o serviço de páginas amarelas e brancas. Os serviços opcionais são a integração agente-software, o serviço de ontologia e a interação agente-homem (Figura 2.4).

O serviço de transporte de mensagens (*Message Transport Service* – MTS) é o responsável por entregar as mensagens trocadas entre os agentes de uma mesma plataforma ou de plataformas diferentes. Todos os agentes FIPA têm acesso a no mínimo um MTS e somente mensagens endereçadas aos agentes podem ser enviadas através do MTS. Além disso, o MTS pode conter mecanismos de segurança [26].

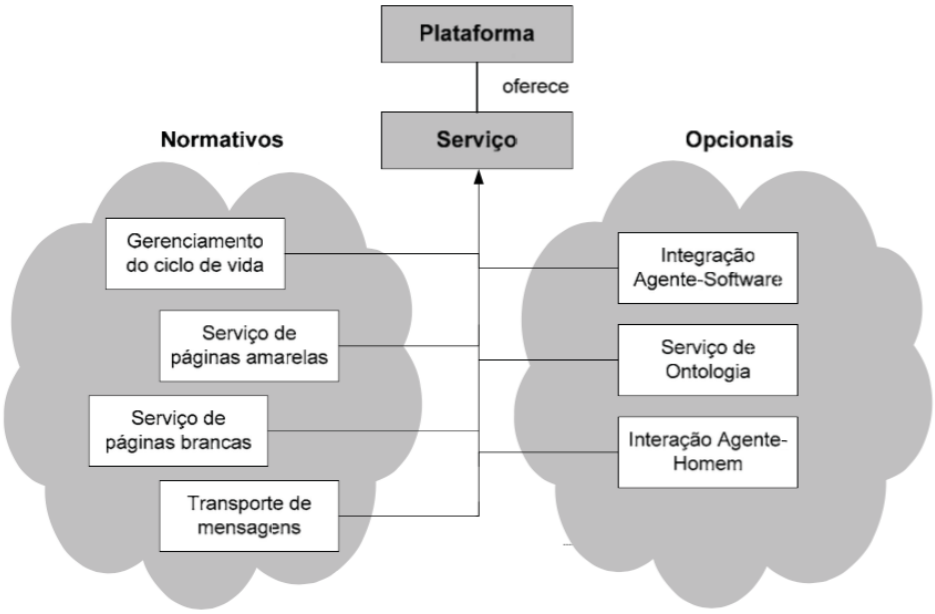

Figura 2.4: Serviços providos por uma plataforma compatível com o padrão FIPA [26].

O serviço de páginas brancas permite que um agente encontre outros agentes capazes de prover um determinado serviço de que necessita. O serviço de páginas amarelas lista os serviços oferecidos por um dado agente.

O serviço de ontologia proporciona o compartilhamento entre os agentes de um repositório com as ontologias utilizadas pelos agentes do sistema, evitando ambiguidades.

O serviço de gerenciamento do ciclo de vida do agente é responsável pela criação, remoção e migração de agentes entre plataformas [26]. Quando um agente é criado, este deve possuir um identificador único e fixo para permitir que um agente ou serviço possa se comunicar com outro agente de maneira não ambígua. O padrão define os possíveis estados no ciclo de vida de um agente e as transições que ocasionam as mudanças de estados (Figura 2.5).

No estado ativo, o MTS entrega as mensagens para o agente normalmente, já nos estados iniciado, em espera e suspenso o MTS armazena em um *buffer* as mensagens até que o agente retorne ao estado ativo ou encaminha as mensagens para um novo local (se o encaminhamento estiver definido para o agente). No estado em trânsito, o MTS armazena as mensagens em *buffers* até que o agente entre no estado ativo (ou seja, a função de movimento falhou na plataforma original ou o agente foi iniciado com sucesso na plataforma de destino) ou encaminha mensagens para um novo local. Somente os agentes móveis podem entrar no estado de trânsito, de forma a garantir que um agente estacionário execute todas as suas instruções na plataforma onde ele foi chamado [13]. No estado desconhecido, o MTS pode tanto armazenar as mensagens em *buffers* quanto rejeitá-las, isto depende da política do MTS e das necessidades de transporte da mensagem.

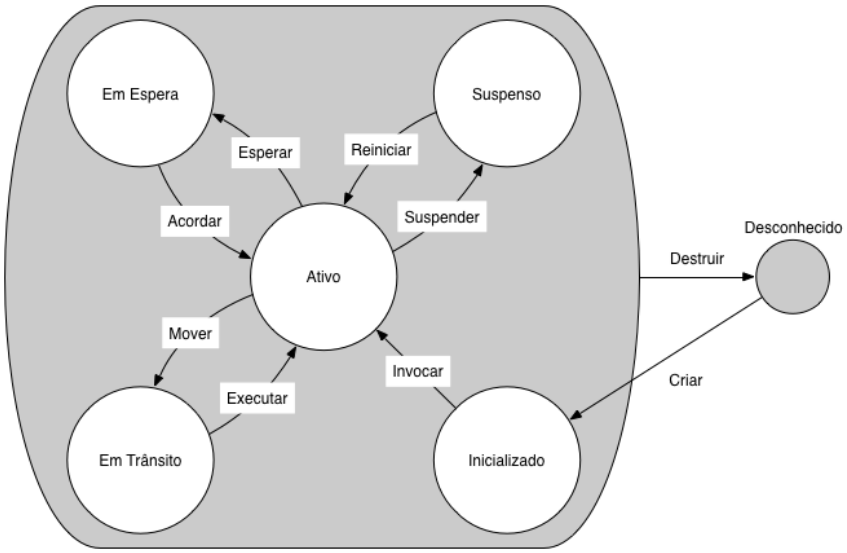

Figura 2.5: Ciclo de vida de um agente segundo padrão FIPA [26].

Além dos serviços especificados pela FIPA, existe uma arquitetura de referência (Figura 2.6) para plataformas de agentes e um conjunto de protocolos de interação definidos para padronizar as conversações entre os agentes.

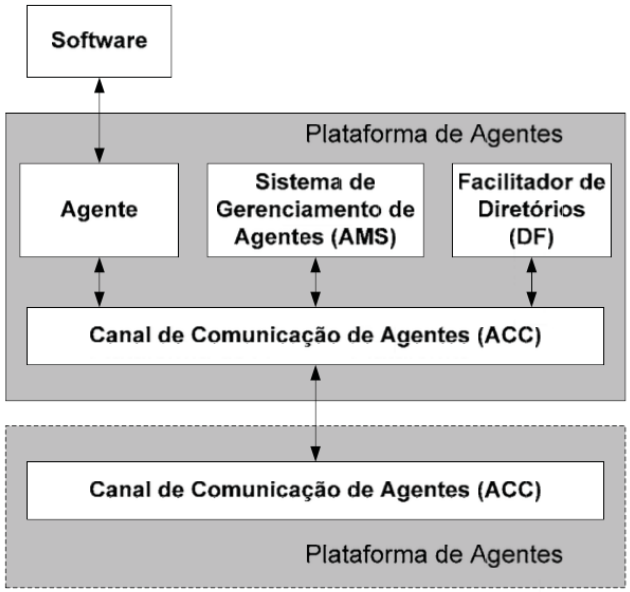

Figura 2.6: Arquitetura de referência FIPA para uma plataforma de agentes [26].

Segundo a especificação FIPA [13], a plataforma de agentes fornece a infraestrutura física em que os agentes podem ser implementados. A plataforma JADE inclui dois agentes especiais (AMS e DF) que são automaticamente ativados em sua inicialização.

O sistema gerenciador de agentes (*Agent Management System* – AMS) é um componente obrigatório e único que exerce o controle de acesso e uso da plataforma, tratando da criação, finalização e demais etapas do ciclo de vida dos agentes. O Facilitador de Diretórios (*Directory Facilitator* – DF) oferece o serviço de páginas amarelas para os outros agentes, ou seja, é um banco de dados centralizado cujas entradas associam um agente aos seus serviços.

2.2 Distribuição de processos judiciais

O processo judicial é um instrumento com o propósito de garantir direitos fundamentais do cidadão e proteger a ordem jurídica, sendo subordinado a diversos princípios do Direito, tais como o Princípio da Legalidade, o da Eficiência e o do Juiz Natural. Ele é composto por autos – peças constitutivas de um processo, tais como as petições, certidões, termos, etc. – e seu ciclo de vida pode ser entendido, de forma simplificada, conforme mostra a Figura 2.7.

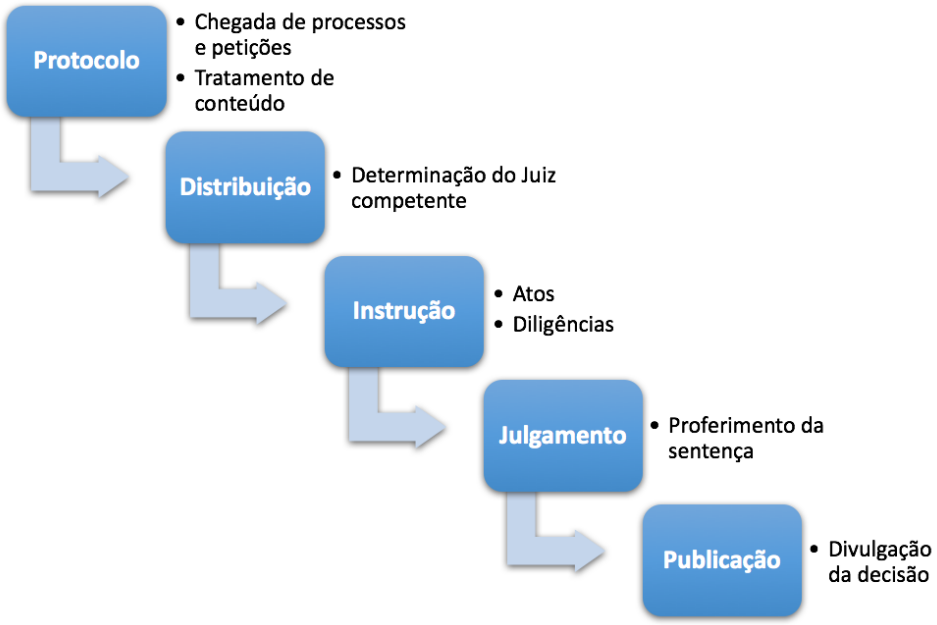

Figura 2.7: Fluxo simplificado de um processo judicial.

A distribuição é o ato em que se define o Órgão Jurisdicional responsável por conduzir e julgar um determinado processo judicial. Ela é um dos elementos fundamentais para o bom andamento do processo judicial e a imparcialidade das decisões, uma vez que busca preservar a garantia do Juiz Natural.

Segundo Delgado [12], o Princípio do Juiz Natural está consagrado no Art. 5º, Inciso XXXVII, da Constituição Federal do Brasil, sendo "uma garantia presente em todas as

Constituições dos povos cultos, refletindo a preocupação de não permitir que ninguém seja processado ou julgado senão por juízes componentes do Poder Judiciário e que sejam investidos de atribuições jurisdicionais fixadas e limitadas pela Lei Maior. O alcance do principio é proibir uma justiça de privilégios ou exceção, garantindo-se que todos os cidadãos tenham seus litígios julgados por juízes legais, juízes investidos nas suas funções de conformidade com as exigências constitucionais. A força dessa garantia constitucional não permite que os poderes constituídos criem juízos destinados a julgamentos de determinados casos ou de pessoas especificadas".

Conforme determina o Código de Processo Civil (CPC): todos os processos estão sujeitos a registro, devendo ser distribuídos onde houver mais de um juiz – Art. 284 [3]; a distribuição, que poderá ser eletrônica, será alternada e aleatória, obedecendo-se rigorosa igualdade – Art. 285 [3].

2.2.1 Tribunal Superior do Trabalho

O Poder Judiciário do Brasil possui diversos ramos e cortes com diferentes níveis de jurisdição. A Justiça do Trabalho, um desses ramos, é dividida em três estágios hierárquicos chamados de instâncias. Atualmente, a primeira instância é composta por 1.587 Varas do Trabalho, a segunda por 24 Tribunais Regionais do Trabalho (TRT) e a terceira pelo TST. A instância em que o processo deve ser iniciado depende da causa a ser tratada. Depois da sentença proferida, há casos em que é possível contestar a decisão mediante recurso à própria corte ou recorrendo para uma corte de instância superior até que o processo seja finalizado quando não ocorrerem – ou forem possíveis – novas apelações.

A função primária do TST é de uniformizar a jurisprudência trabalhista brasileira. A ele compete julgar as demandas que excedam a jurisdição dos TRT e outras controvérsias decorrentes de relações de trabalho. O TST é composto por vinte e sete Ministros, escolhidos – dentre os indicados segundo regras estabelecidas – e nomeados pela Presidência da República após aprovação pela maioria absoluta do Senado Federal. Para o julgamento dos processos judiciais, esses Ministros atuam de maneira colegiada em Órgãos Judicantes, quais sejam:

- Tribunal Pleno: composto pelos vinte e sete Ministros do TST;
- Órgão Especial: composto pelo Presidente do TST, pelo Vice-Presidente do TST, pelo Corregedor-Geral da Justiça do Trabalho e mais 11 Ministros;
- Seção Especializada em Dissídios Coletivos: composto pelo Presidente do TST, pelo Vice-Presidente do TST, pelo Corregedor-Geral da Justiça do Trabalho e mais 6 Ministros;

- Seção Especializada em Dissídios Individuais: dividida em duas subseções:
  - Subseção I: composto pelo Presidente do TST, pelo Vice-Presidente do TST, pelo Corregedor-Geral da Justiça do Trabalho e mais 11 Ministros;
  - Subseção II: composto pelo Presidente do TST, pelo Vice-Presidente do TST, pelo Corregedor-Geral da Justiça do Trabalho e mais 7 Ministros;
- Turmas: oito turmas com três Ministros cada. O Presidente do TST, o Vice-Presidente do TST e o Corregedor-Geral da Justiça do Trabalho não participam das Turmas.

É importante conhecer essa organização para entender a distribuição de processos no âmbito do TST, uma vez que cada um desses Órgãos possui competências específicas que serão determinantes na efetivação da distribuição.

2.2.2 Processo judicial eletrônico no TST

O Processo Judicial Eletrônico é regulamentado no âmbito do TST pelos Atos SEJUD.GP n.º 342/2010[7] e n.º 415/2010[8]. O Processo que tramita no TST pode ter sido iniciado em uma Vara do Trabalho, em um TRT ou no próprio TST. Quando o processo foi iniciado em uma Vara do Trabalho, dizemos que o TST atua como instância extraordinária. Quando foi iniciado em um TRT, dizemos que o TST atua como instância ordinária. Quando o processo foi iniciado no próprio TST, dizemos que ele atua como instância originária.

Enquanto instância originária, o processo é iniciado mediante entrega de uma petição inicial, o que pode ser feito pessoalmente por um advogado ao entregar o documento no departamento de protocolo do órgão ou remotamente utilizando um sistema chamado de e-DOC (Sistema Integrado de Protocolização e Fluxo de Documentos Eletrônicos).

Enquanto instância ordinária ou extraordinária, o processo é enviado eletronicamente, do TRT ao TST, por meio de rede de computadores utilizando um *software* chamado eRemessa (Sistema de Remessa de Peças Processuais).

Cada processo recebe uma identificação numérica de acordo com padronização, uniforme para todos os órgãos do Poder Judiciário, estabelecida na Resolução n.º 65/2008[9] do CNJ. A numeração observa a estrutura NNNNNNN-DD.AAAA.J.TR.OOOO, onde:

- NNNNNNN: 7 (sete) dígitos que identificam o número sequencial do processo de acordo com a unidade onde ele foi originado;
- DD: 2 (dois) dígitos numéricos utilizados para cálculo de verificação de integridade;

[7] Ato n. 342/SEJUD.GP, de 27 de julho de 2010 - http://aplicacao.tst.jus.br/dspace/handle/1939/7650
[8] Ato n. 415/SEJUD.GP, de 1 de setembro de 2010 - http://aplicacao.tst.jus.br/dspace/handle/1939/8230
[9] Resolução CNJ n.º 65/2008 - http://www.cnj.jus.br/images/stories/docs_cnj/resolucao/rescnj_65b.pdf

- AAAA: ano de ajuizamento do processo com 4 (quatro) dígitos;
- J: 1 (um) digito que identifica órgão ou segmento do Poder Judiciário conforme padronizado pelo CNJ. Para a Justiça do Trabalho, utiliza-se o número 5 (cinco);
- TR: 2 (dois) dígitos que identificam o Tribunal do respectivo segmento do Poder Judiciário. Para a Justiça do Trabalho, identificam o número do TRT. Utiliza-se o código 00 para identificar processos originários do TST e o código 90 para processos originários do Conselho Superior da Justiça do Trabalho;
- OOOO: número da unidade de origem (Vara do Trabalho) com 4 (quatro) dígitos. Nos processos de competência originária dos Tribunais, o campo deve ser preenchido com zeros.

Além dos já citados sistemas e-DOC e eRemessa, o acompanhamento e a tramitação do processo no TST envolvem a utilização de outros programas, quais sejam:

- Sistema Visualização de Autos[10]: permite aos usuários previamente cadastrados a visualização de peças processuais de seu interesse;
- Sistema de Consulta Processual[11]: disponível no portal do TST na Internet, permite a consulta pública da movimentação de processos e petições. Por meio dessa consulta é possível identificar para qual Órgão Judicante e qual Ministro relator um determinado processo foi distribuído (Figura 2.8);
- Diário Eletrônico da Justiça do Trabalho[12] (DEJT): é o instrumento oficial de divulgação e publicação dos atos do Poder Judiciário;
- Sistema de Apoio à Gabinetes (SAG): trata da informatização do fluxo do processo judicial que ocorre internamente nos gabinetes dos Ministros, controlando a confecção de documentos como os votos, acórdãos, despachos e planilhas de julgamento, além de integração com as sessões de julgamento;
- Base Gerencial Jurídica: solução de *Business Inteligente* (BI) que contempla um *data warehouse* com dados obtidos de diversas fontes para criação de relatórios gerenciais e estatísticos;
- O Sistema de Informações Judiciárias (SIJ) permite a autuação, a movimentação, o gerenciamento e o acompanhamento dos processos judiciais, além da elaboração de relatórios gerenciais e estatísticos, sendo utilizado por vários departamentos em

[10]Sistema Visualização de Autos - http://aplicacao3.tst.jus.br/visualizacaoAutos
[11]Consulta Processual no TST - http://www.tst.jus.br/processos-do-tst
[12]Diário Eletrônico da Justiça do Trabalho - https://aplicacao2.jt.jus.br/dejt/

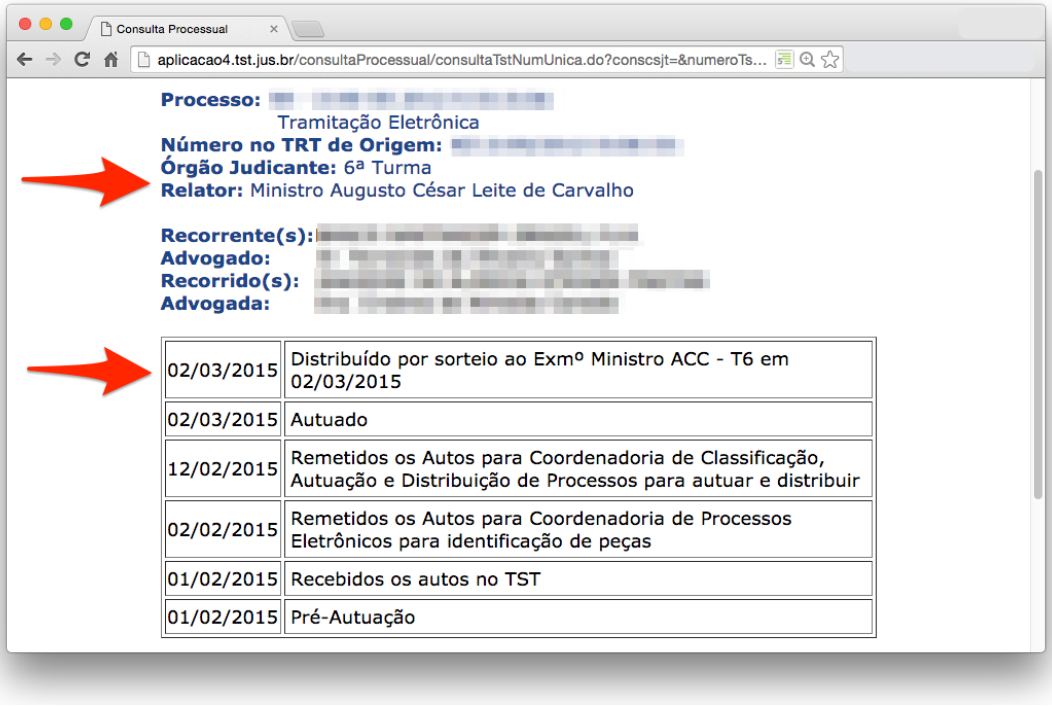

Figura 2.8: Sistema de Consulta Processual do TST, destacando informações relativas a distribuição.

diversas etapas do fluxo do processo. O SIJ é um grande sistema composto por vários módulos, sendo a distribuição de processos um desses módulos.

Há ainda outros sistemas que utilizam indiretamente as informações processuais, tais como o programa da Certidão Negativa de Débitos Trabalhistas (CNDT), o Sistema Único de Cálculos da Justiça do Trabalho (e-Calc), o programa de Informações Processuais por E-mail (TST-Push) e a Base de Jurisprudência do TST.

As informações utilizadas por esses sistemas são armazenadas em bancos de dados mediante uso de, principalmente, Sistemas Gerenciadores de Banco de Dados Relacional (SGBDR) Oracle e Sistema de Gestão de Documentos (GED) EMC *Documentum*.

O SGBDR gerencia os dados processuais gerais relativos à identificação do processo, classes e fases processuais, movimentações e andamentos, órgãos judicantes, magistrados, competências, partes interessadas, advogados, procuradores, assuntos, autuação, distribuição, pautas, sessões, petições, despachos, votos, acórdãos e muitos outros. São informações relativas a mais de 3 milhões de processos judiciais, um grande volume de dados distribuídos em mais de 950 tabelas relacionais, totalizando cerca de 1,69 *terabytes*, sem contar os dados armazenados em *logs* para fins de auditoria e réplicas dos bancos de dados. Por exemplo, a tabela que relaciona as tramitações dos processos armazena atualmente mais de 35 milhões de registros.

O GED cuida do gerenciamento das peças processuais (autos do processo), tratando do seu armazenamento, disponibilização, controle de acesso e outras atividades gerenciais. Atualmente, são mais 25 milhões de documentos armazenados, a maioria em formato PDF, totalizando 42 *terabytes* de dados.

2.2.3 Legislação relacionada à distribuição

A distribuição de processos judiciais segue regras gerais estabelecidas no CPC (Código de Processo Civil, Lei n.º 13.105 de 16 de março de 2015), além de regras específicas definidas em normativos dos Tribunais constituintes do Poder Judiciário.

O sorteio é a regra básica quando há mais de um Juiz competente para decidir sobre a questão. É o sorteio que define aleatoriamente o Juiz da causa (ou o relator do Processo nos Tribunais colegiados) na maioria dos casos, sendo um desdobramento dos princípios constitucionais da Isonomia, da Impessoalidade Administrativa e do Juiz Natural.

Mas, há casos em que não é adotado sorteio para distribuição do processo. A chamada distribuição por dependência (Art. 286 do CPC) ocorre quando um processo é direcionado à um Órgão Judicante que conduz outro processo legalmente interligado, de forma a permitir a decisão conjunta de ações conexas ou continentes, ou que possam gerar risco de decisões conflitantes caso decididos separadamente, mesmo sem conexão entre eles. A distribuição por dependência busca a eficácia e eficiência no julgamento de ações, mas é preciso tomar cuidado para que esse mecanismo não seja utilizado para direcionar processos ilegalmente.

Quanto a distribuição por dependência, a legislação prevê ainda que as causas deverão ser distribuídas à um mesmo Órgão Judicante "quando, tendo sido extinto o processo sem resolução de mérito, for reiterado o pedido". Isto objetiva coibir a prática da desistência e posterior reajuizamento de demandas na tentativa de obter vantagem caso o processo seja distribuído a outro juízo.

Consta no Art. 289 do CPC que a distribuição poderá ser fiscalizada pela parte, por seu procurador, pelo Ministério Público e pela Defensoria Pública [3]. Dessa forma, um sistema que realize a distribuição automatizada deve prover meios pelos quais essa fiscalização possa ser efetivada, sendo requisito funcional a transparência da informação relacionada as ações dos agentes distribuidores.

A distribuição deve considerar também os impedimentos legais que proíbem que um determinado magistrado atue em um processo. O Art. 144 do CPC determina que um Juiz está impedido de atuar no processo:

I em que interveio como mandatário da parte, oficiou como perito, funcionou como membro do Ministério Público ou prestou depoimento como testemunha;

II de que conheceu em outro grau de jurisdição, tendo proferido decisão;

III quando nele estiver postulando, como defensor público, advogado ou membro do Ministério Público, seu cônjuge ou companheiro, ou qualquer parente, consanguíneo ou afim, em linha reta ou colateral, até o terceiro grau, inclusive;

IV quando for parte no processo ele próprio, seu cônjuge ou companheiro, ou parente, consanguíneo ou afim, em linha reta ou colateral, até o terceiro grau, inclusive;

V quando for sócio ou membro de direção ou de administração de pessoa jurídica parte no processo;

VI quando for herdeiro presuntivo, donatário ou empregador de qualquer das partes;

VII em que figure como parte instituição de ensino com a qual tenha relação de emprego ou decorrente de contrato de prestação de serviços;

VIII em que figure como parte cliente do escritório de advocacia de seu cônjuge, companheiro ou parente, consanguíneo ou afim, em linha reta ou colateral, até o terceiro grau, inclusive, mesmo que patrocinado por advogado de outro escritório;

IX quando promover ação contra a parte ou seu advogado.

Há também os casos em que o magistrado é considerado suspeito para julgar a demanda. A suspeição do juiz é mais subjetiva do que o impedimento, podendo o juiz declarar-se suspeito por motivo de foro íntimo, sem necessidade de declarar suas razões. Segundo o CPC (Art. 145), há suspeição do juiz:

I amigo íntimo ou inimigo de qualquer das partes ou de seus advogados;

II que receber presentes de pessoas que tiverem interesse na causa antes ou depois de iniciado o processo, que aconselhar alguma das partes acerca do objeto da causa ou que subministrar meios para atender às despesas do litígio;

III quando qualquer das partes for sua credora ou devedora, de seu cônjuge ou companheiro ou de parentes destes, em linha reta até o terceiro grau, inclusive;

IV interessado no julgamento do processo em favor de qualquer das partes.

Caso o processo já tenha sido distribuído, os magistrados devem declarar-se impedidos ou suspeitos conforme os casos previstos em lei. Na suspeição ou no impedimento, o processo deverá ser redistribuído, sendo nulos os atos praticados pelo Magistrado suspeito ou impedido.

2.2.4 Distribuição de processos no TST

Conforme estabelecido no Regimento Interno do TST (RITST) [30], compete ao Presidente do Tribunal determinar a distribuição dos processos, bem como dirimir as controvérsias referentes à distribuição.

Os processos judiciais são distribuídos após os registros e as formalidades necessárias à sua identificação. "Os processos de competência do Tribunal serão distribuídos por classe, observada a competência e composição dos órgãos judicantes, assim como a ordem cronológica do seu ingresso na Corte, concorrendo ao sorteio todos os Ministros, excetuados os membros da direção" (Art. 89 RITST [30]). São membros da direção o Presidente do Tribunal, o Vice-Presidente do Tribunal e o Corregedor-Geral da Justiça do Trabalho.

A classe do processo identifica o tipo de processo de acordo com padronização taxonômica e terminológica estabelecida pelo CNJ para todo o Poder Judiciário. A classificação adequada do processo é condição para que a sua distribuição seja corretamente realizada. Para o TST, a grande maioria dos processos tratados pertencem as seguintes classes, em ordem decrescente do número de processos: Agravo de Instrumento em Recurso de Revista (AIRR); Recurso de Revista (RR); Embargos de Declaração (ED); Agravo (Ag); Embargos (E); Recurso Ordinário (RO). A Figura 2.9 apresenta um gráfico com a quantidade de processos por classe para os anos de 2013 e 2014.

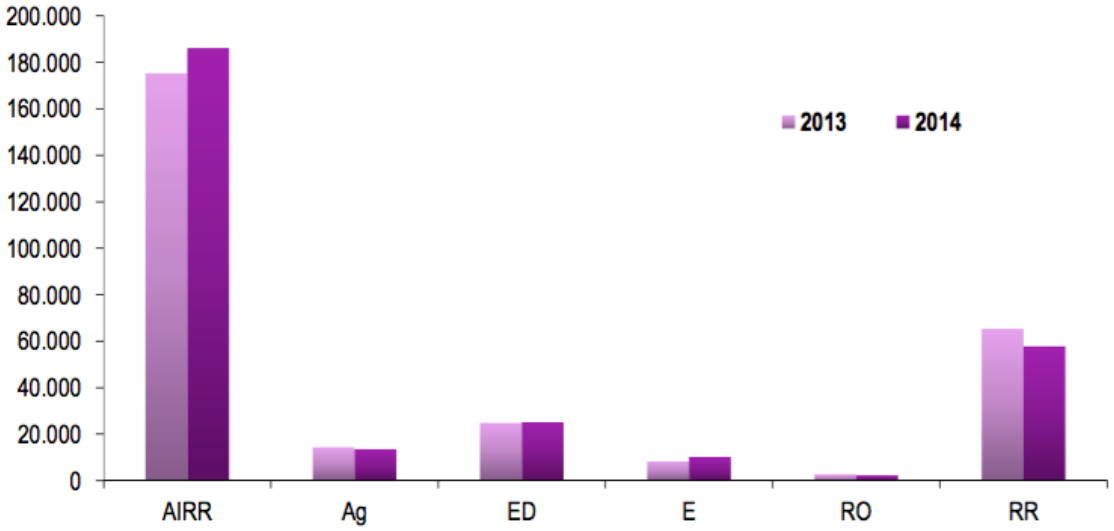

Figura 2.9: Processos recebidos por classe para as classes mais comuns, 2013-2014 [31].

No ano de 2015, o departamento de distribuição do TST realizou a distribuição de 195.867 processos (Figura 2.10). Há períodos de interrupção parcial da distribuição. "No período correspondente às férias dos Ministros, não haverá distribuição de processos, exceto os de dissídio coletivo, mandado de segurança, ações cautelares e habeas corpus" [30].

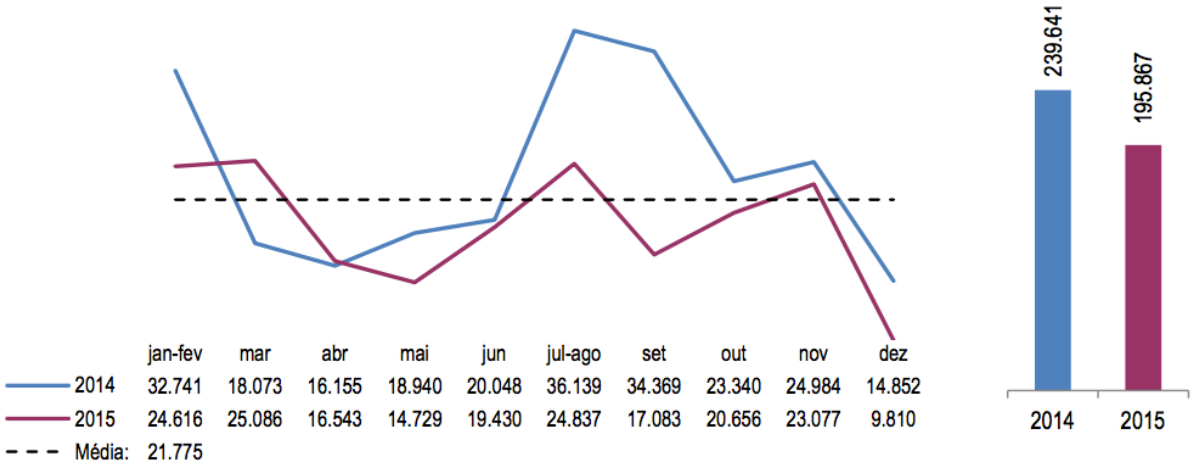

Figura 2.10: Processos judiciais distribuídos no TST, 2014-2015 [32].

Os programas que controlam e executam a distribuição são, em boa parte, procedimentos e funções que residem e atuam diretamente no SGDBR que gerencia os bancos de dados de processos judiciais. Eles estão em uso desde o ano de 1997 e seu desenvolvimento ao longo desses anos tratou dos requisitos definidos em regras constantes do RITST e em práticas de trabalho empregadas pelos departamentos administrativos que realizam as atividades de autuação e distribuição de processos desde a época em que os processos eram físicos e circulavam em papel.

No contexto dos processos físicos, havia a necessidade de uma única atividade de distribuição lidar com grandes quantidades de processos. Para que fosse possível tratar problemas frequentes, separar os processos em lotes, elaborar e juntar termos diversos, foi estabelecida uma etapa de preparação da distribuição que permitiu a operacionalização da atividade de distribuição seguindo agendamentos.

Com a implantação do processo eletrônico, uma distribuição passou a ser realizada individualmente para cada novo processo, mas a etapa de preparação para distribuição foi mantida, em razão da necessidade de revisão e conferência dos dados.

Para que o sistema funcione corretamente, o gestor do sistema precisa manter atualizadas algumas tabelas no banco de dados com informações sobre magistrados, órgãos judicantes, classes processuais, competências dos órgãos judicantes, impedimentos dos magistrados e outras. Problemas de qualidade dos dados podem impossibilitar o programa determinar o destino do processo com precisão, e pode causar atraso em seu trâmite. Em alguns casos, se o problema não for tratado, ocorrerá a distribuição ordinária por sorteio geral.

Para um processo estar pronto para distribuição, ele precisa atender alguns critérios como: estar na fila de distribuição e no departamento responsável pela distribuição; não estar com vistas para advogado; a classe do processo é de competência do Tribunal; há

órgão judicantes competentes e magistrados sem impedimentos para recebê-los; inexistência de irregularidades quanto às informações de CNPJ e CPF; ausência de petições pendentes; identificação de processos relacionados.

O sistema de distribuição contempla diversas funcionalidades, dentre as quais destacamos:

- distribuição por sorteio entre os magistrados aptos com base nas informações de composição do órgão judicante;
- verificação dos impedimentos dos magistrados quanto a processos, advogados e partes interessadas;

  "Reconhecida a suspeição ou o impedimento do Relator, declarar-se-ão nulos os atos praticados pelo Ministro suspeito ou impedido, e o processo será redistribuído, na forma regimental" (Art. 265, parágrafo único, do RITST [30]).
- manutenção de histórico de distribuição e de relatores do processo;
- retirada e liberação de processo da fila de distribuição para solução de pendências:

  O sistema permite que o usuário deixe o processo num estado "suspenso" para impedir que o mesmo seja distribuído sem que seja resolvida alguma possível pendência, sendo obrigatório informar o motivo dessa ação.
- atribuição de quantidades de processos em percentuais diferentes por magistrado:

  "Em face da atribuição contida no inciso IX do presente Art., o Presidente de Turma receberá 10% (dez por cento) a menos de processos distribuídos, respeitada a proporção quanto às classes processuais de competência da Turma" (Art. 81, parágrafo único, RITST [30]).
- agendamento de datas futuras para distribuição;
- tratamento de prevenção e dependência:

  A prevenção ocorre quando um processo retorna à Corte para novo exame. O sistema busca reconhecê-la baseado na numeração única do processo, embora ainda seja necessária a confirmação pelo distribuidor. Algumas exceções precisam ser tratadas, como por exemplo, a que ocorre quando a classe do processo que firmou a prevenção não é da competência do órgão julgador para o qual o processo foi distribuído anteriormente. Além disso, o sistema necessita das informações da data da distribuição, do órgão e do relator da época, bem como do histórico da composição dos órgãos para poder determinar o magistrado que atualmente ocupa àquela posição no órgão.

"O Colegiado que conhecer do processo terá jurisdição preventa para o julgamento dos recursos posteriores interpostos no mesmo processo, observada a competência. Parágrafo único: o processo que tramita na fase de execução será distribuído ao Ministro a quem coube a relatoria na fase de conhecimento, ou a quem o tenha substituído ou sucedido, devendo os processos tramitar conjuntamente, sempre que possível" (Art. 98. do RITST [30]).

- verificação de prevenção:

  "Os embargos interpostos contra decisão de Turma serão distribuídos entre os Ministros não integrantes do Colegiado prolator da decisão embargada" (Art. 104 do RITST [30]).

- cálculos dos saldos de distribuição de processos por Ministros para fins de compensação, levando em conta as distribuições, as redistribuições e seus cancelamentos:

  O sistema controla a compensação de processos promovendo a distribuição em quantidades iguais por Órgão Judicante, Ministro e classe processual. Os saldos da compensação são diretamente afetados pelas redistribuições no âmbito dos órgãos judicantes.

- lançamentos de débitos e créditos pelo gestor do sistema para compensação de saldos:

  A chamada compensação especial ocorre quando é necessário o controle de saldos por meio de lançamentos de quantidades de processos a serem compensados a débitos ou créditos. Esse recurso permite ao gestor do sistema atribuir mais ou menos processos para determinado magistrado. Busca corrigir disparidades entre magistrados que mudam de órgão judicante, quando a quantidade legada é diferente da herdada.

  "O relator que se afastar definitivamente da Turma ou da Seção Especializada, por motivo de remoção, receberá no órgão para o qual se removeu os processos vinculados ao antecessor em que este ainda não apôs o visto. Parágrafo único: Na hipótese de remoção de Turma, o Ministro que se removeu receberá no novo órgão, em compensação, a diferença entre o acervo processual deixado na Turma de origem, ao se remover, e o que recebeu na nova cadeira, observadas as classes processuais" (Art. 94-B do RITST [30]).

- cancelamento de distribuição com o respectivo estorno do saldo da compensação:

  Por meio de um programa criado especificamente para esta finalidade, o usuário da unidade que realizou a distribuição pode desfazê-la, sendo obrigatório informar o motivo da ação. O programa executa os procedimentos de inclusão de registro na

tabela de histórico de relatores excluídos dos processos, com todas as informações do cancelamento, como usuário, data e hora, motivo e os demais campos do registro excluído; estorno do saldo da compensação; lançamento de tramitação informando o cancelamento; etc.

- verificação de embargos infringentes:

  "À distribuição dos embargos infringentes não concorrerá o Ministro que já tenha atuado no processo como Relator e/ou redigido o acórdão embargado" (Art. 103 do RITST [30]).

- geração de documentos, peças de capa e termo da distribuição para juntada no processo eletrônico;

- geração de arquivo XML para publicação no DEJT.

As informações da distribuição só se tornam públicas após o devido tratamento das informações, mediante atualização do histórico de relatores e tramitações. Mais detalhes acerca das regras de distribuição estão normatizados no RITST, cujos trechos relevantes à distribuição encontram-se no Anexo I.

2.3 Transparência

Para Holzner [16], transparência é o valor social do acesso aberto, público e individual à informação mantida e disponibilizada por centros de autoridade, sendo crítica para as sociedades democráticas. É um conceito moderno especialmente relacionado ao contexto político-social.

É notório o crescimento de iniciativas, em organizações públicas e privadas, com fins de demonstrar transparência. O estabelecimento de transparência nas organizações se traduz nos processos organizacionais e nas informações geradas por meio da execução e instanciação desses processos.

Leite e Cappelli [20] definem transparência de processo como "a característica que possibilita ao cidadão acesso, facilidade de uso, qualidade de conteúdo, entendimento e auditoria aos/dos processos que tratam de informações de seu interesse, sob a tutela de centro de autoridade".

No que tange a distribuição de processos judiciais, a transparência da informação é especialmente importante para a adequada operacionalização da fiscalização estabelecida no Art. 280 do CPC [3] que define que "a distribuição poderá ser fiscalizada pela parte, por seu procurador, pelo Ministério Público e pela Defensoria Pública".

Leite e Cappelli [20] propuseram o *Transparency SIG (Softgoal Interdependency Graph)* que busca definir um catálogo de características de transparência a serem implementados em processos organizacionais. Na Figura 2.11, os nós são características de transparência a serem implementadas e os *links* identificam relacionamentos, onde as linhas de traço contínuo indicam as dependências entre as características de qualidade e suas operacionalizações, e as linhas tracejadas indicam contribuições entre as operacionalizações.

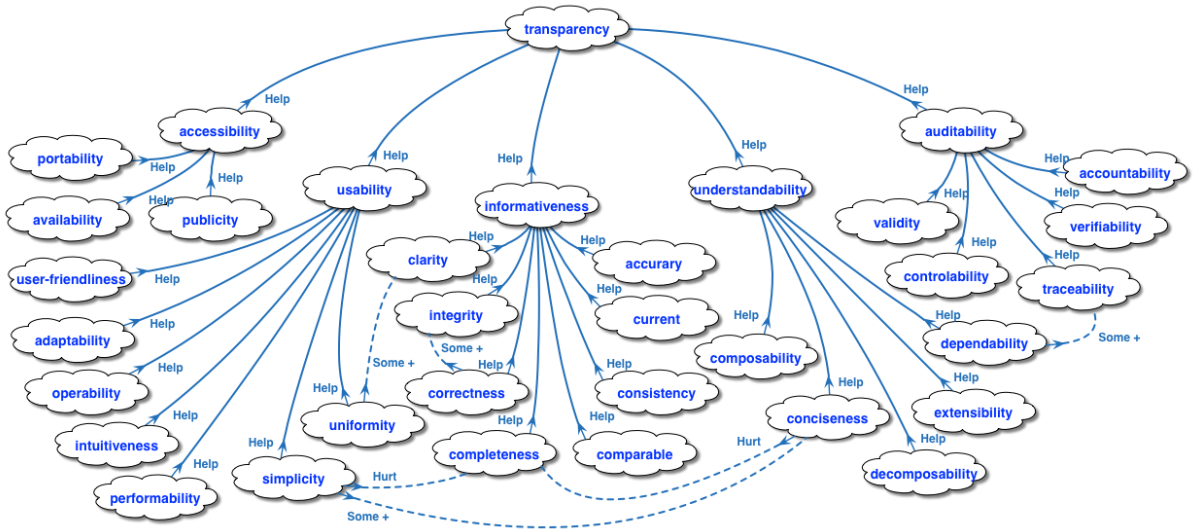

Figura 2.11: *Transparency SIG* [20].

Cada uma das características de transparência do *Transparency SIG* é definida conforme abaixo:

- Acessibilidade: capacidade de obtenção. Essa característica é implementada mediante:
  - Disponibilidade: capacidade de ser utilizado no momento em que se fizer necessário;
  - Portabilidade: capacidade de ser usado em diferentes ambientes;
  - Publicidade: capacidade de se tornar público.
- Usabilidade: capacidade de uso. Essa característica é implementada mediante:
  - Adaptabilidade: capacidade de mudar de acordo com as circunstâncias e necessidades;
  - Amigabilidade: capacidade de uso sem esforço;
  - Desempenho: capacidade de operar adequadamente;
  - Intuitividade: capacidade de ser utilizado sem aprendizado prévio;

  - Operabilidade: capacidade de estar operacional;
  - Simplicidade: capacidade de não apresentar dificuldades ou obstáculos;
  - Uniformidade: capacidade de manter uma única forma.

- Informativo: capacidade de prover informações de qualidade. Essa característica é implementada mediante:
  - Acurácia: capacidade de execução isenta de erros sistemáticos;
  - Atualidade: capacidade de estar no estado atual;
  - Clareza: capacidade de nitidez e compreensão;
  - Comparabilidade: capacidade de ser comparado;
  - Completeza: capacidade de não faltar nada do que pode ou deve ter;
  - Consistência: capacidade de resultado aproximado de várias medições de um mesmo item;
  - Corretude: capacidade de ser isento de erros;
  - Integridade: capacidade de ser correto e imparcial.

- Entendimento: capacidade de alcançar o significado e o sentido. Essa característica é implementada mediante:
  - Concisão: capacidade de ser resumido;
  - Composição: capacidade de construir ou formar a partir de diferentes partes;
  - Dependência: capacidade de identificar a relação entre as partes de um todo;
  - Divisibilidade: capacidade de ser particionado;
  - Extensibilidade: capacidade de utilização em mais de um caso.

- Auditabilidade: capacidade de exame analítico. Essa característica é implementada mediante:
  - Controlabilidade: capacidade de ter domínio;
  - Explicável: capacidade de informar a razão de algo;
  - Rastreabilidade: capacidade de seguir o desenvolvimento de um processo ou a construção de uma informação, suas mudanças e justificativas;
  - Validade: capacidade de ser testado por experimento ou observação para identificar se o que está sendo feito é correto;
  - Verificabilidade: capacidade de identificar se o que está sendo feito é o que deve ser feito.

2.3.1 Graus de transparência

Leite e Cappelli [20] identificaram dependências entre os grupos de transparência (acessibilidade, usabilidade, informativo, entendimento e auditabilidade), de forma que, para que as características de um determinado grupo sejam institucionalizadas, há necessidade de que outras, de outros grupos já tenham sido institucionalizadas antes (Figura 2.12).

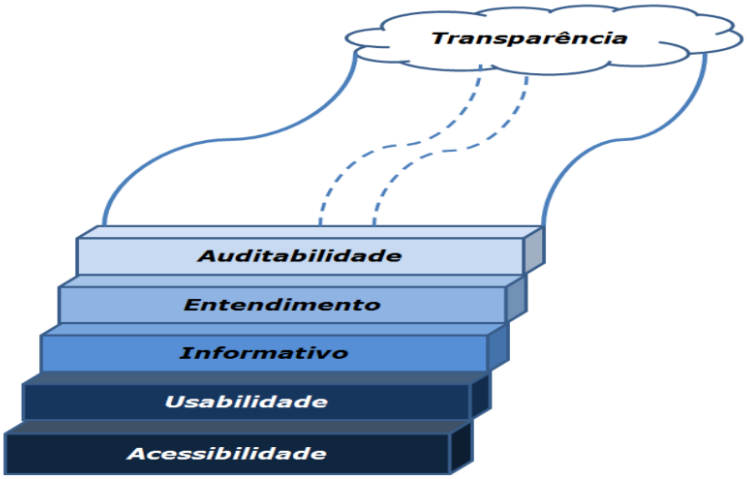

Figura 2.12: Degraus da Transparência [20].

Cada um dos graus de transparência pode então ser estabelecido através da institucionalização do conjunto de suas características [20], a saber:

- Grau 1 – Acessibilidade: a transparência é realizada através da capacidade de acesso. Esta capacidade é identificada através da aferição de práticas que implementam características de disponibilidade, portabilidade, publicidade;
- Grau 2 – Usabilidade: a transparência é realizada através das facilidades de uso. Esta capacidade é identificada através da aferição de práticas que implementam características de adaptabilidade, amigabilidade, desempenho, intuitividade, operabilidade, simplicidade, uniformidade;
- Grau 3 – Informativo: a transparência é realizada através da qualidade da informação. Esta capacidade é identificada através da aferição de práticas que implementam características de acurácia, atualidade, clareza, comparabilidade, completeza, consistência, corretude, integridade;
- Grau 4 – Entendimento: a transparência é realizada através do entendimento. Esta capacidade é identificada através da aferição de práticas que implementam características de concisão, composição, dependência, divisibilidade, extensibilidade;

- Grau 5 – Auditabilidade: a transparência é realizada através da auditabilidade. Esta capacidade é identificada através da aferição de práticas que implementam características de controlabilidade, explicável, rastreabilidade, validade, verificabilidade.

Modelos de maturidade têm sido propostos para como forma de institucionalizar práticas e avaliar as organizações em diferentes domínios. Diferentemente de outros modelos como o CMMI (*Capability Maturity Model Integration*) da Engenharia de *Software*, o modelo proposto por Leite e Cappelli [20] não preconiza a necessidade de alcance completo de todas as características de um determinado grau para que se possa passar para um próximo. Explicita apenas que há forte dependência entre os grupos de características, indicando que algumas características podem impossibilitar o alcance de outras caso não sejam estabelecidas.

Este Capítulo apresentou conceitos importantes relacionados à solução proposta, cujo detalhamento será apresentado no Capítulo 3.

Capítulo 3

Solução Proposta

O presente trabalho propõe o projeto, o desenvolvimento e a validação do SMA LawDisTrA, o qual atende requisitos da distribuição eletrônica de processos judiciais ao tempo em que resolve problemas encontrados nas soluções atualmente empregadas, especialmente quanto a transparência da informação.

3.1 Concepção do LawDisTra

A proposta de construção de um SMA para realização da atividade de distribuição de processos judiciais teve início durante a realização da disciplina Sistemas Multiagentes no primeiro semestre de 2015 no âmbito do Departamento de Ciência da Computação (CIC) da Universidade de Brasília (UnB). Na ocasião, sistemas multiagentes, metodologia Tropos e framework JADE foram estudados e empregados para construção de um protótipo do sistema. O trabalho desenvolvido foi apresentado no WPOS (*Workshop* da pós-graduação em computação do CIC/UnB) realizado em outubro de 2015.

Nos meses seguintes, continuou-se o trabalho de desenvolvimento do SMA para atender outros requisitos da solução, além de estudar os problemas relacionados a ausência de transparência na distribuição de processos. Esse trabalho culminou na publicação de [1]. A Figura 3.1 apresenta a interface gráfica até então desenvolvida para interação com os agentes do SMA que foi protótipo para o LawDisTra.

Prosseguiu-se com a evolução da arquitetura do LawDisTra de forma a torná-lo mais flexível e robusto; aumentar a autonomia dos agentes; integrá-lo com um banco de dados semelhante ao utilizado no TST; modelar o conhecimento envolvido; separar as regras de distribuição (regras de negócio) da programação lógica dos agentes; construir uma interface que permitisse a auditoria das distribuições de processos.

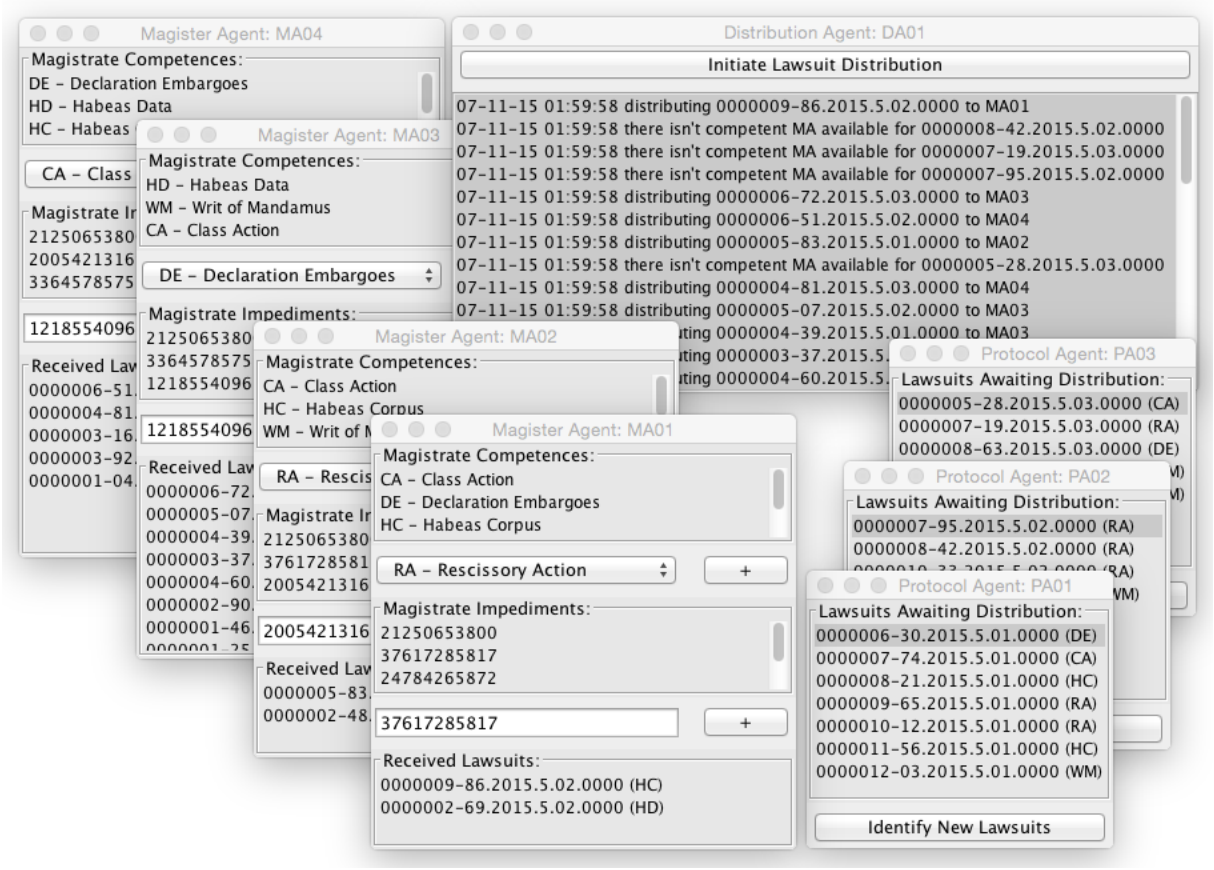

Figura 3.1: GUI para interação com os agentes no protótipo do LawDisTrA.

A solução atualmente proposta trabalha a integração de diversas tecnologias e uso de conhecimentos da área da computação de forma a atender os requisitos da distribuição eletrônica de processos judiciais e de transparência da informação (Figura 3.2).

(a) O processo judicial e sua distribuição é o domínio da aplicação em discussão;

(b) A solução proposta resolverá problemas atualmente encontrados ao aplicar conceitos relacionados a transparência da informação;

(c) A distribuição de processos será realizada mediante o uso de um SMA desenvolvido utilizando o *framework* JADE;

(d) O desenvolvimento de SMA com JADE é feito utilizando a linguagem de programação Java. A tecnologia Java Swing será empregada para construção de Interfaces Gráficas de Usuário (GUI) de forma a permitir o controle dos agentes por um humano;

(e) A metodologia Tropos será empregada para fins de auxiliar o projeto e a documentação do SMA;

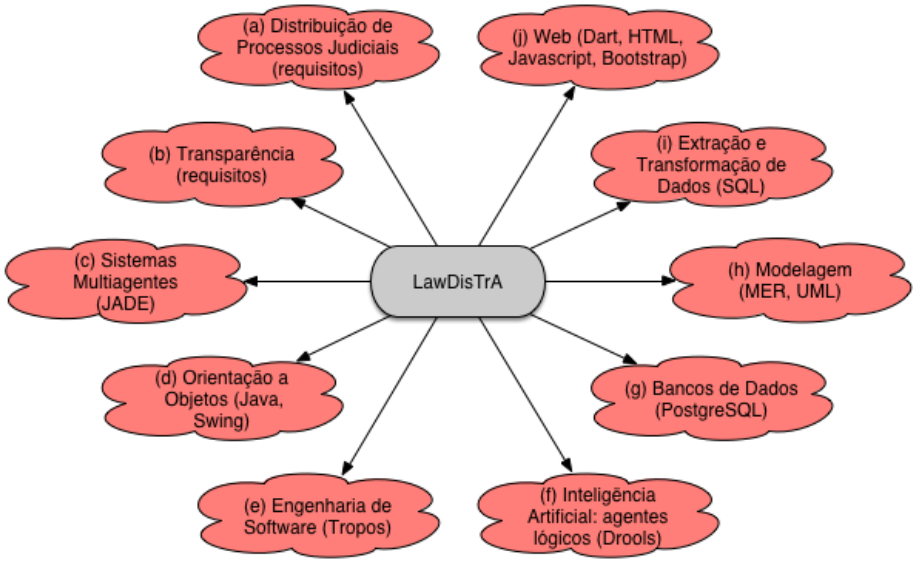

Figura 3.2: Áreas de conhecimento e tecnologias envolvidas no projeto.

(f) Os agentes responsáveis pela distribuição dos processos são agentes baseados em conhecimento e utilizarão a ferramenta Drools como forma de especificar e aplicar as regras de distribuição;

(g) Sistemas Gerenciadores de Bancos de Dados são utilizados para manter os dados relativos aos processos judiciais eletrônicos. LawDisTrA utiliza o SGBD relacional PostgreSQL;

(h) Modelos de Dados em níveis conceituais e físicos foram construídos de forma a auxiliar no projeto e documentação dos bancos de dados consultados e manipulados pelos agentes. Eles são simplificações dos bancos de dados encontrados no TST e contêm apenas os dados relevantes para a distribuição dos processos judiciais;

(i) A experimentação e validação da solução envolve a criação de bancos de dados com informações de processos judiciais, magistrados, partes, advogados, impedimentos, classes processuais, órgãos judicantes, competências, etc.;

(j) A solução envolve também a construção de um serviço *web* para permitir a consulta dos dados gerados durante a distribuição dos processos judiciais e, dessa forma, promover a transparência da informação. Esse serviço foi implementado utilizando a linguagem de programação Dart e fornece páginas HTML com *Javascript* e CSS (*Bootstrap*).

As escolhas das tecnologias utilizadas se basearam em critérios como: licenciamento gratuito, coerência com os requisitos e disponibilidade de boa documentação *online*.

O desenvolvimento do LawDisTrA seguiu as fases da metodologia Tropos, conforme seções a seguir.

3.2 Requisitos Iniciais

A Figura 3.3 apresenta os requisitos iniciais para o SMA de distribuição de processos judiciais. Nessa fase, foram identificados três atores: o Agente de Protocolo (AP), o Agente Magistrado (AM) e o Agente de Distribuição (AD).

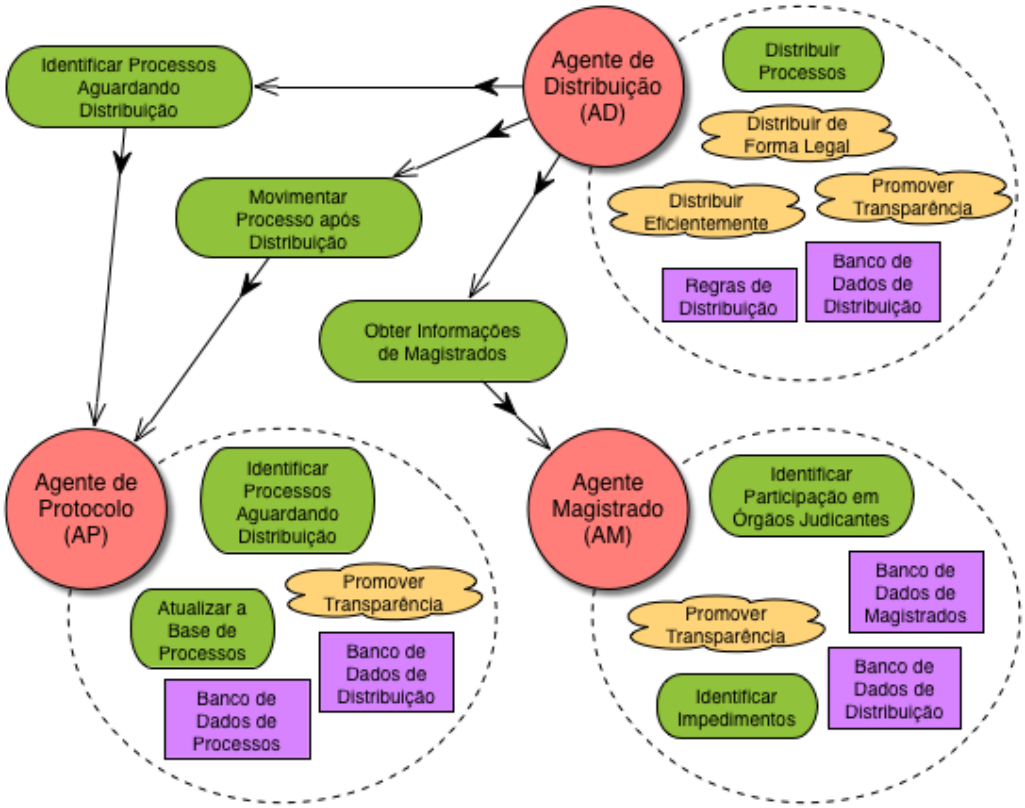

Figura 3.3: Requisitos Iniciais para o Sistema LawDisTrA.

O Agente de Protocolo (AP) representa a pessoa ou o departamento da organização que é responsável por autuar os processos. A autuação é o cadastro realizado para instauração de um processo e a sua preparação para a tramitação.

O Agente Magistrado (AM) representa o magistrado responsável por conduzir os processos judiciais. Os magistrados compõem os Órgãos Judicantes que são responsáveis por julgar um processo. Podem existir vários órgãos judicantes competentes para julgar um determinado processo. É preciso também conhecer os impedimentos dos magistrados, conforme apresentado na Seção 2.2.3.

O Agente de Distribuição (AD) representa a pessoa ou o departamento da organização responsável por realizar a distribuição do processo, ou seja, definir para qual magistrado o processo será atribuído. A distribuição deve ocorrer segundo regras preestabelecidas que visem uma distribuição legal e eficiente.

Para que seja possível realizar a distribuição, o AD deve conhecer os AM, quais órgãos judicantes o AM participa e quais seus impedimentos. Para que a distribuição seja realizada de forma legal é preciso que, via de regra, a escolha do AM seja realizada mediante sorteio e considere as competências do AM. Para que a distribuição seja eficiente, é importante identificar os impedimentos dos magistrados e se o processo em questão já foi distribuído em fase anterior ou se existem processo relacionados ou que tratem de causa semelhante e, dessa forma, o processo possa ser distribuído para o mesmo magistrado que já detém conhecimento da causa.

3.3 Requisitos Finais

As Figuras 3.4, 3.5 e 3.6 apresentam os requisitos finais para um agente de protocolo, um agente magistrado e um agente de distribuição, respectivamente. O detalhamento especifica os planos de ação compostos pelas tarefas para alcançar os objetivos (*hard-goals*), os *soft-goals* e os recursos necessários.

Todos os agentes possuem como *soft-goal* a necessidade de promover a transparência. O alcance desse objetivo é facilitado em sistemas multiagente na medida em que os agentes registrem de forma padronizada suas percepções e ações. LawDisTrA registra essas informações no banco de dados de distribuição de forma a permitir a consulta e auditoria das informações gravadas.

O agente de protocolo (Figura 3.4) possui dois objetivos: identificar processos aguardando distribuição e atualizar a base de processos.

A identificação de processos aguardando distribuição diz respeito a tarefa de buscar no banco de dados de processos judiciais um processo que precise ser distribuído, bem como todas as informações relativas ao processo que são necessárias para sua devida distribuição. É preciso identificar quais são as partes (as pessoas e organizações envolvidas) do processo; os advogados e procuradores – se houver – das partes; as fases anteriores do processo, se for o caso de se tratar de um processo já analisado no âmbito do tribunal; além de informações de processos relacionados.

Uma vez que é o agente de protocolo quem tem acesso ao banco de dados de processos judiciais, o agente de distribuição dependerá do AP para atualizar esse banco de dados quando for identificada a realização de uma nova distribuição.

Figura 3.4: Requisitos Finais para um Agente de Protocolo.

Por sua vez, um agente magistrado (Figura 3.5) deve ter acesso ao banco de dados de magistrados para que possa identificar os seus impedimentos, bem como a participação do magistrado nos órgãos judicantes existentes.

De forma semelhante ao sistema empregado atualmente no TST, LawDisTrA trata basicamente da verificação de três tipos de impedimentos: impedimentos declarados por magistrados no que diz respeito a processos específicos; impedimentos em relação as partes do processo; e impedimentos em relação aos advogados das partes.

As informações percebidas tanto pelos agentes de protocolo quanto pelos agentes magistrados serão passadas, quando solicitadas, ao agente de distribuição.

Já o agente de distribuição (Figura 3.6) possui como principal objetivo a distribuição de processos. Para alcançar esse objetivo, ele precisa interagir com o banco de dados de distribuição e com os demais agentes do LawDisTra de forma a: identificar quais são os órgãos judicantes do tribunal; identificar processos que aguardam distribuição; identificar os magistrados do tribunal, com as respectivas informações de impedimentos e composição de órgãos judicantes; aplicar regras de distribuição definidas; informar a distribuição para que o protocolo possa atualizar o banco de dados dos processos judiciais.

Figura 3.5: Requisitos Finais para um Agente Magistrado.

3.4 Projeto Arquitetural

LawDisTrA utiliza o *framework* JADE em seu desenvolvimento e, portanto, sua arquitetura foi projetada em conformidade com o modelo e as *constraints* do *framework*, buscando obter uma solução robusta e flexível.

No diagrama da Figura 3.7 é apresentada a arquitetura completa da solução, representando os seus componentes e relacionamentos. Essa arquitetura foi projetada para preservar o máximo de semelhança com o ambiente encontrado no TST. As setas no diagrama indicam o sentido do fluxo de dados entre esses componentes. Ou seja: o componente Auditor da Distribuição é consumidor dos dados do Banco de Dados de Processos Judiciais Eletrônicos e do Banco de Dados da Distribuição; um Agente de Distribuição é consumidor e produtor dos dados do Banco de Dados da Distribuição. Com essa diagramação da arquitetura, busca-se evidenciar que:

- Podem existir vários Agentes de Protocolo ($AP_1, AP_2, ..., AP_n$), cada qual especializado em tratar grupos de processos que podem ser classificados por tipo do processo ou por origem (em qual Tribunal o processo foi originado).

Figura 3.6: Requisitos Finais para um Agente de Distribuição.

– Os AP devem identificar nos Bancos de Dados de Processo Judiciais Eletrônicos quais processos necessitam de distribuição, obtendo as informações necessárias para a realização desta atividade. Eles devem ter inteligência para identificar, por exemplo, quando um processo está relacionado a outro de forma que eles possam ser distribuídos para o mesmo AM que tratou do processo anteriormente;
– As informações obtidas pelos AP serão repassadas para os AD quando solicitadas. Se necessário, os AP podem solicitar que os AD inicializem os procedimentos de distribuição;
– Cabe ao AP atualizar o Banco de Dados de Processos Judiciais com informações de movimentação dos processos (direcionar o processo para um departamento específico) a partir de informações de distribuição recebidas dos AD;
– Os AP atualizam o Banco de Dados da Distribuição de forma a registrar nele suas percepções e ações e, dessa forma, permitir que esses dados sejam utilizados em eventuais consultas do módulo Auditor da Distribuição;
– Agentes de Protocolo registram seus serviços no catálogo mantido pelo agente

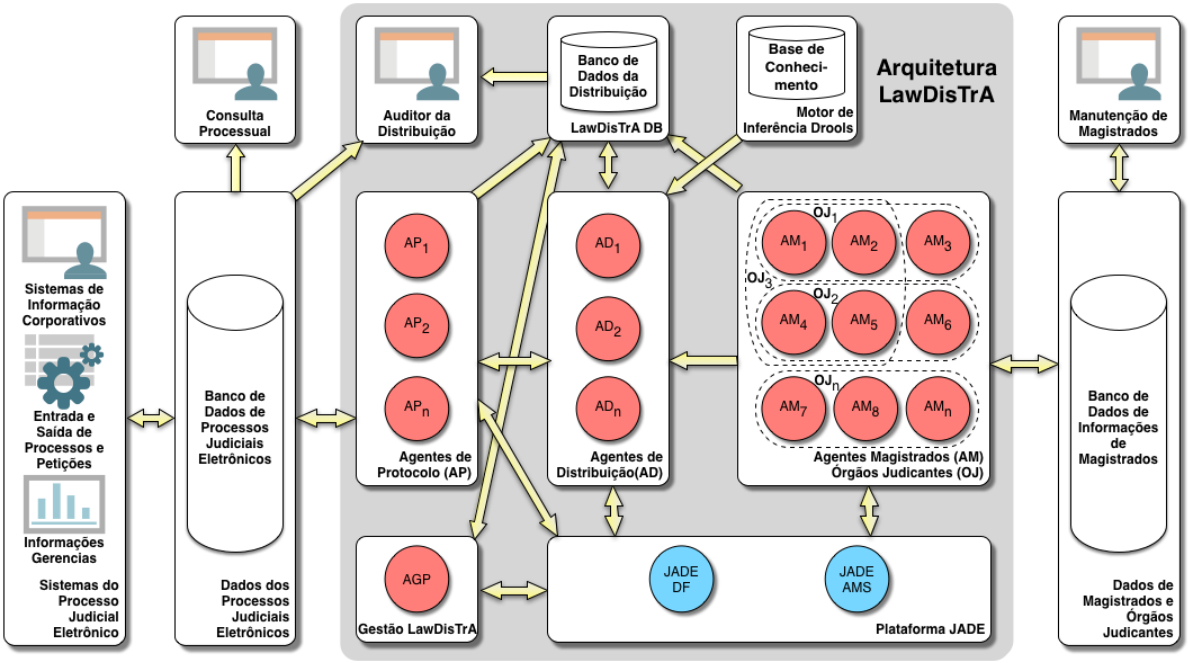

Figura 3.7: Arquitetura do sistema LawDisTrA.

DF e seu ciclo de vida é gerenciado pelo agente AMS, ambos providos pela plataforma JADE.

- Existem vários Agentes Magistrados ($AM_1, AM_2, AM_3, ..., AM_n$) e Órgãos Judicantes ($OJ_1, OJ_2, OJ_3, ..., OJ_n$) no Tribunal. No LawDisTrA, os OJ são agrupamentos lógicos dos AM. Os OJ definem as competências dos magistrados para julgar os processos. Cada AM pode ser membro de um ou mais OJ.
  - Os AM devem identificar os órgãos judicantes em que são membros e seus respectivos impedimentos no Banco de Dados de Informações de Magistrados. Essas informações serão repassadas aos AD quando solicitadas por eles;
  - Os AM podem possuir inteligência para identificar autonomamente novos impedimentos a partir de outras fontes de dados e atualizar o Banco de Dados de Informações de Magistrados;
  - Os AM também atualizam o Banco de Dados da Distribuição de forma a registrar nele suas percepções e ações;
  - Agentes Magistrados registram seus serviços no catálogo mantido pelo agente DF e seu ciclo de vida é gerenciado pelo agente AMS, ambos providos pela plataforma JADE.
- Mais de um Agente de Distribuição ($AD_1, AD_2, ..., AD_n$) podem ser empregados para fins de divisão da carga de processamento ou especialização da distribuição

onde cada AD trataria tipos específicos de processos. Por exemplo, poderia ser empregado um AD para cada AP disponível.

  - Os AD obtêm as informações necessárias à distribuição mediante interação com os AP e AM. Eles não devem acessar diretamente as bases de dados dos processos judiciais eletrônicos, dos magistrados e dos órgãos judicantes de forma evitar acoplamento e aumentar a flexibilidade da solução. Ou seja, os AD não precisam conhecer como essas bases estão estruturadas, devendo se basear apenas nos dados enviados pelos AP e AM.
  - A distribuição de processos pode ocorrer de forma automática, com os AD realizando suas tarefas de forma contínua ou agendada, ou ainda de forma automatizada a partir do recebimento de solicitações de outros agentes ou por interação direta com usuários;
  - AD registram no Banco de Dados de Distribuição as informações necessárias para a distribuição, bem como suas percepções e ações. Essas informações podem utilizada tanto pelos demais AD no auxílio às distribuições ou pelo módulo Auditor da Distribuição para fins de transparência;
  - As regras de distribuição utilizadas pelos AD estão especificadas separadamente em uma base de conhecimento. Essa base de conhecimento será utilizada pelos AD mediante aplicação do motor de inferência do Drools, que, combinado com as percepções do AD, indicará para qual magistrado um processo deve ser distribuído. As percepções dos AD, construídas a partir da interação deles com os demais agentes AP e AM, serão os fatos da base de conhecimento;
  - Agentes AD identificam os demais agentes disponíveis no LawDisTrA (AP e AM) a partir de consultas ao catálogo mantido pelo agente DF e seu ciclo de vida é gerenciado pelo agente AMS, ambos providos pela plataforma JADE. Agentes AD também podem utilizar os agentes DF para registrar serviços por eles providos.

- O agente AGP (Agente Gestor da Plataforma) foi incluído na solução para facilitar o gerenciamento do LawDisTrA pelo usuário do sistema. Por meio dele, o usuário poderá realizar a ativação ou desativação dos demais agentes (AP, AM e AD) na plataforma, bem como monitorar o funcionamento do sistema.

  - O AGP realiza suas ações mediante interação com os agentes da plataforma JADE;

  - O AGP consulta o Banco de Dados de Distribuição de forma a obter as configurações do LawDisTrA, identificando quais são os protocolos, magistrados e distribuidores disponíveis;
  - O AGP também registra suas percepções e ações no Banco de Dados da Distribuição. Essas informações podem ser utilizadas para rastrear quando o sistema e os agentes foram carregados ou descarregados, além de outras informações relativas a operação do LawDisTrA.

- O módulo Auditor da Distribuição foi concebido para promover a transparência da informação armazenada no Banco de Dados da Distribuição. Por meio dele, o usuário poderá auditar cada uma das distribuições realizadas. Esse módulo também utiliza informações extraídas do Banco de Dados de Processos Judiciais para que possa apresentar ao usuário informações detalhadas acerca dos processos.

- O componente Consulta Processual representa a ferramenta de consulta a informações de processos judiciais disponível publicamente no *website* do TST[1]. Essa consulta apresenta informações semelhantes as apresentadas pelo Auditor da Distribuição, sendo que este apresenta informações detalhadas acerca das distribuições realizadas e permite o rastreamento dos agentes do LawDisTrA.

- O componente Manutenção de Magistrados representa o sistema de informação corporativo responsável por manter as informações relacionadas aos magistrados do Tribunal em um banco de dados respectivo.

- Os Processos Judiciais Eletrônicos no TST são gerenciados e trabalhados por um conjunto de sistemas informatizados:

  - Sistemas Jurídicos Corporativos que trabalham o processo judicial durante seu ciclo de vida no Tribunal, ou seja, realiza o tratamento do conteúdo e a movimentação entre as diversas unidades administrativas que atuam no processo, desde sua autuação até o julgamento e publicação da decisão;
  - Sistemas que tratam da entrada e saída do processo no Tribunal. O processo pode ser advindo de um tribunal de instância inferior ou ser originário do próprio tribunal. A saída pode ser em retorno para a instância inferior ou para remessa à uma instância superior;
  - Sistemas que utilizam dados dos processos para tratamentos estatísticos e elaboração de relatórios gerenciais de apoio às decisões;

[1]Consulta Processual do TST - http://aplicacao4.tst.jus.br/consultaProcessual

– As informações dos processos eletrônicos são armazenadas em volumosos bancos de dados de forma centralizada. Muitos procedimentos e funções que implementam regras de negócio estão contidos nos sistemas gerenciadores desses bancos de dados (*stored procedures*).

3.5 Projeto Detalhado

A Tabela 3.1 apresenta a especificação P.A.G.E. que consolida informações dos agentes do LawDisTrA.

Tabela 3.1: Especificação P.A.G.E dos agentes.

Agente	**Percepções**	**Ações**	**Objetivos**	**Ambiente**
AP	Solicitações de agentes AD (agente reativo).	Obter informações de processos não distribuídos. Modificar banco de dados de processos judiciais. Registrar-se junto ao DF.	Identificar Processos Aguardando Distribuição. Atualizar Base de Processos.	Demais agentes do LawDisTrA e da plataforma JADE.
AM	Solicitações de agentes AD (agente reativo).	Descobrir impedimentos. Descobrir Órgãos Judicantes do Magistrado. Registrar-se junto ao DF.	Identificar Participação em Órgãos Judicantes. Identificar Impedimentos.	Bancos de dados de distribuição.
AD	Solicitações de execução de distribuição. Informações de agentes AP, AM e AGP.	Identificar Órgãos Judicantes. Identificar processos para distribuição mediante interação com agentes AP. Identificar magistrados mediante interação com agentes AM. Aplicar regras de distribuição. Registrar-se junto ao DF. Buscar outros agentes no DF.	Distribuir processos.	Ambiente completamente observável, multiagente, determinístico, episódico, estático, discreto e conhecido.
AGP	Requisições do usuário (agente reativo)	Ativar e desativar agentes no LawDisTra. Encerrar o sistema. Informar agentes AD de mudanças na plataforma.	Manter e monitorar os agentes no sistema LawDisTrA	

Para alcançar seus objetivos, os agentes devem realizar as tarefas identificadas nas fases de levantamento de requisitos (Figuras 3.3, 3.4, 3.5, e 3.6). No JADE, agentes realizam

tarefas a partir da especificação de comportamentos (subclasses de *Behaviour*) que serão executados de forma escalonada. A comunicação entre agentes é realizada mediante troca de mensagens assíncronas. A Figura 3.8 apresenta um diagrama de colaboração entre os componentes do LawDisTrA, especificando a troca de mensagens (rótulos na cor verde) para comunicação entre os agentes e a interação com os bancos de dados mediante ativação de comportamentos (rótulos na cor azul).

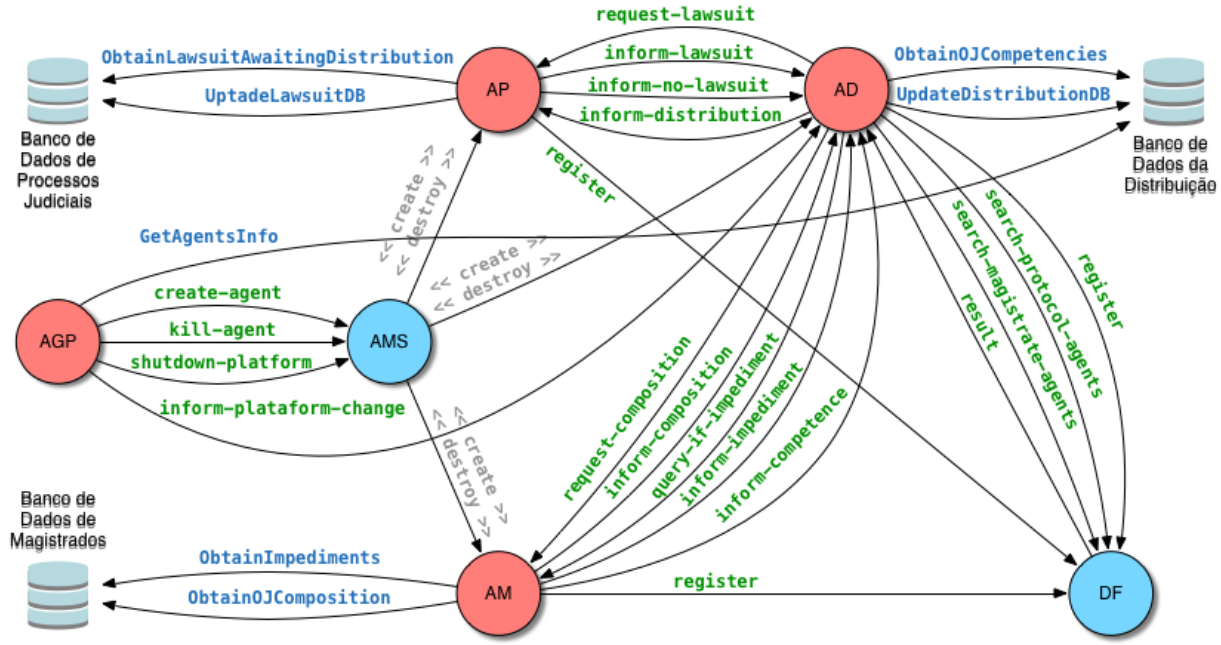

Figura 3.8: Colaboração entre componentes do LawDisTrA.

a. LawDisTrA é iniciado com o carregamento dos agentes do framework JADE (AMS, DF) e do AGP. O AGP busca, ativando o comportamento *GetAgentsInfo*, no banco de dados da distribuição, informações relativas a configuração do sistema, tais como quantos e quais são os agentes disponíveis no sistema.

b. O usuário pode então, utilizando a GUI do AGP, solicitar ao AMS que sejam ativados ou desativados cada um dos agentes (AP, AM e AD) que o usuário selecionar. A ativação é realizada pelo envio de mensagens do tipo *create-agent* e a desativação por mensagens do tipo *kill-agent*.

c. A criação (*«create»*) e destruição (*«destroy»*) de agentes na plataforma JADE fica a cargo do AMS.

d. O usuário pode ainda utilizar a GUI do AGP para solicitar ao AMS que encerre adequadamente o LawDisTrA enviando uma mensagem do tipo *shutdown-platform* ao AMS.

e. Agentes do tipo AP, AM e AD, ao serem iniciados, solicitam o registro dos seus serviços junto ao catálogo do agente DF mediante envio de mensagem do tipo *register*.

f. Cada AP obtém, ativando o comportamento *ObtainLawsuitAwaitingDistribution*, do banco de dados de processos judiciais, as informações dos processos que aguardam a realização da distribuição. Esse banco de dados poderia ser distribuído, por exemplo, para que cada AP buscasse o processo diretamente no sistema do Tribunal que originou o processo. É preciso ficar atento para evitar situações de concorrência onde mais de um AP trate um mesmo processo.

g. Cada AM obtém, ativando os comportamentos *ObtainImpediments* e *ObtainOJComposition*, no banco de dados de magistrados, informações acerca do magistrado que eles representam.

h. Cada AD obtém, ativando o comportamento *ObtainOJCompetencies*, no banco de dados da distribuição, quais são os Órgão Judicantes existentes, bem como suas respectivas competências. Saber a competência do Órgão judicante permite o AD identificar quais os OJ podem receber um processo de acordo com o seu tipo.

i. O AD busca no DF, mediante mensagens do tipo *search-protocol-agents* e *search-magistrate-agents*, quais os AP e AM ativados no LawDisTrA. É para os AM disponíveis que os processos serão distribuídos.

j. A plataforma deve permitir a ativação e desativação de novos agentes sem que isso afete a execução do sistema. Por isso, toda vez que um agente for ativado ou desativado, o AGP deve notificar os AD ativos. Essa notificação é feita por meio de mensagem do tipo *inform-plataform-change*. Ao receber mensagens desse tipo, os AD precisam se reconfigurar, buscando no DF quais AP e AM estão disponíveis.

k. Para cada um dos AM disponíveis, o AD envia requisição (mensagem tipo *request-composition*) solicitando a informação de quais Órgãos Judicantes o Magistrado atua. O AM questionado informa a sua participação em Órgão Judicante mediante mensagem tipo *inform-composition*. O distribuidor precisa saber a composição dos Órgãos Judicantes para identificar os Magistrados competentes para distribuição.

l. O AD solicita para um AP um processo para distribuir por meio de mensagem do tipo *request-lawsuit*. O AD realiza essa solicitação continuamente, intercalando os pedidos entre todos os AP disponíveis na plataforma. O AP questionado responde informando os dados de um processo pronto para distribuição (*inform-lawsuit*) ou indica que não há processos aguardando distribuição (*inform-no-lawsuit*).

m. O AD verifica quais Órgãos Judicantes são competentes para receber o processo e qual a composição de cada um desses órgãos. Para cada um dos magistrados que compõem o órgão judicante competente, o AD questiona esse AM (mensagem tipo *query-if-impediment*) se ele é impedido de julgar o processo em questão. O AM responde informando competência (mensagem tipo *inform-competence*) ou impedimento (mensagem tipo *inform-impediment*). O AD distribui um processo para um AM competente e desimpedido. Essa verificação não é necessária quando o processo tiver que ser distribuído por dependência ou prevenção, ou seja, se ele deve ser distribuído para algum magistrado específico em razão de já ter sido dirigido anteriormente a ele ou se está relacionado a um outro processo conexo que já encontra-se distribuído para um Magistrado.

n. O AD solicita, mediante mensagem tipo *inform-distribution*, ao AP que atualize o banco de dados de processos judiciais com a informação do Órgão Judicante e Magistrado relator identificado pela aplicação das regras de distribuição. Por sua vez, o AP atualiza o banco de dados, ativando o comportamento UpdateLawsuitDB, para indicar a movimentação do processo para o departamento do magistrado.

o. O AD atualiza, ativando o comportamento *UpdateDistributionDB*, o banco de dados da distribuição com informações acerca de uma distribuição realizada.

Conforme apresentado, para alcançar seus objetivos, os agentes precisarão interagir mediante a troca de mensagens assíncronas. A Tabela 3.2 relaciona os tipos de mensagens trocadas entre os agentes no LawDisTrA.

Tabela 3.2: Performativas de comunicação utilizadas pelos agentes no LawDisTrA.

Tipo de Mensagem	**Agente Origem**	**Agente Destino**	**Performativa FIPA**	**Semântica**
create-agent	AGP	AMS	REQUEST	solicita a criação de um agente específico e sua ativação no SMA.
kill-agent	AGP	AMS	REQUEST	solicita a exclusão de um agente ativo no SMA.
shutdown-platform	AGP	AMS	REQUEST	solicita o encerramento do SMA.
inform-platform-change	AGP	AD	INFORM	notifica acerca de ativação ou desativação de agentes no SMA.
register	AP, AM ou AD	DF	REQUEST	registra serviço no catálogo do DF.
search-protocol-agents	AD	DF	REQUEST	busca por serviços de agentes tipo AP no catálogo do DF.
search-magistrate-agents	AD	DF	REQUEST	busca por serviços de agentes tipo AM no catálogo do DF.
result	DF	AD	INFORM	informa resultado da busca por serviços no catálogo do DF.
request-lawsuit	AD	AP	REQUEST	solicita informações de algum processo que aguarda distribuição.
inform-lawsuit	AP	AD	INFORM	resposta com informações de um processo que aguarda distribuição.
inform-no-lawsuit	AP	AD	INFORM	informa que não há processos aguardando distribuição para o AP questionado.
inform-distribution	AD	AP	INFORM	notifica acerca de uma distribuição realizada.
request-composition	AD	AM	REQUEST	questiona quais são os órgãos judicantes que o magistrado faz parte.
inform-composition	AM	AD	INFORM	responde questionamento acerca de composição em órgãos judicantes.
query-if-impediment	AD	AM	QUERY-IF	questiona se o magistrado é impedido para julgar um determinado processo.
inform-impediment	AM	AD	INFORM	informa que o magistrado está impedido para julgar o processo.
inform-competence	AM	AD	INFORM	informa que o magistrado é competente para julgar o processo.

3.5.1 Bancos de Dados

O projeto do banco de dados iniciou com a identificação das principais entidades e atributos importantes para a distribuição de processos judiciais. O DER (Diagrama Entidade-Relacionamento) da Figura 3.9 apresenta o modelo de dados em nível conceitual conforme notação de Peter Chen [10].

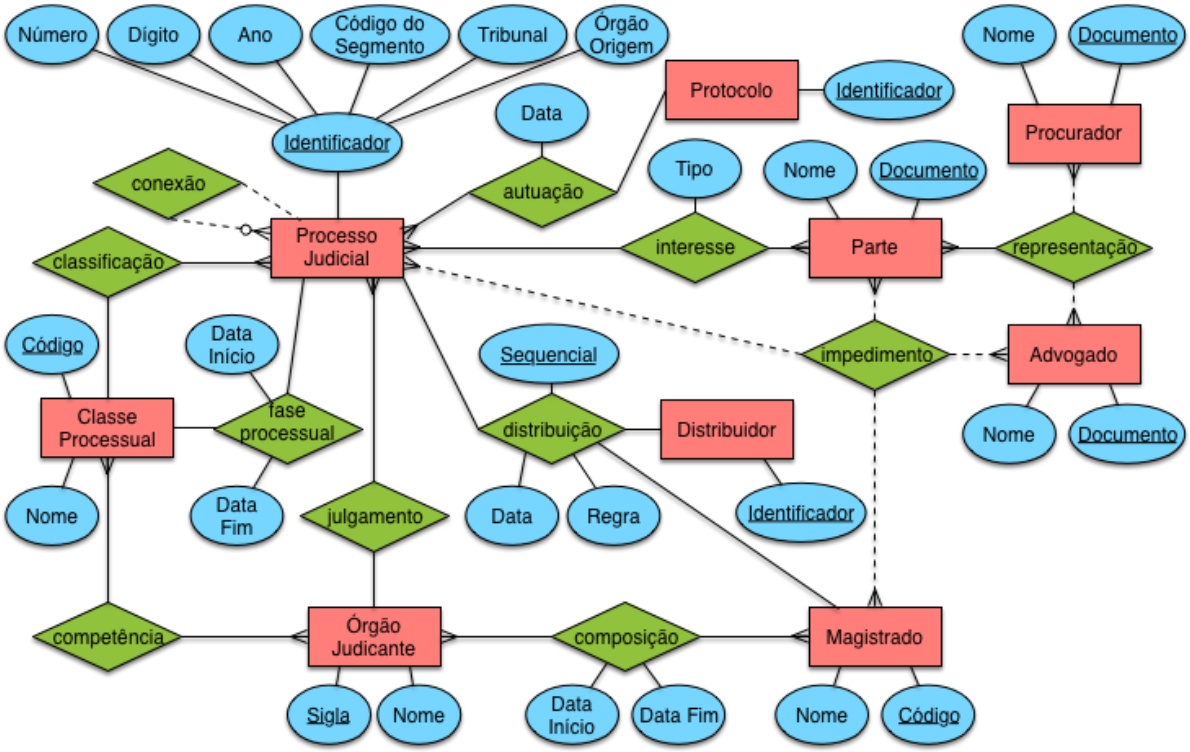

Figura 3.9: Diagrama Entidade-Relacionamento do modelo de dados conceitual para a distribuição de processos judiciais.

Conforme pode ser observado no DER (Figura 3.9), o **Processo Judicial** é *autuado* por uma entidade **Protocolo** que representa um departamento da organização. O processo é *julgado* por **Órgãos Judicantes** que possuem *competência* para julgar processos de acordo com sua **Classe processual**. Um **Órgão Judicante** é *composto* por **Magistrados** que serão responsáveis por conduzir o julgamento dos processos *distribuídos* à eles por um **Distribuidor** (departamento ou pessoa da organização responsável pela *distribuição*). O **Magistrado** pode possuir *impedimentos* relacionados à **Advogados**, **Processos Judiciais** específicos ou **Partes** do processo. As **Partes** são as pessoas ou organizações *interessadas* em determinados **Processos Judiciais**. Essas **Partes** são *representadas* por **Advogados** ou **Procuradores** (quando a **Parte** *representada* é uma entidade do governo).

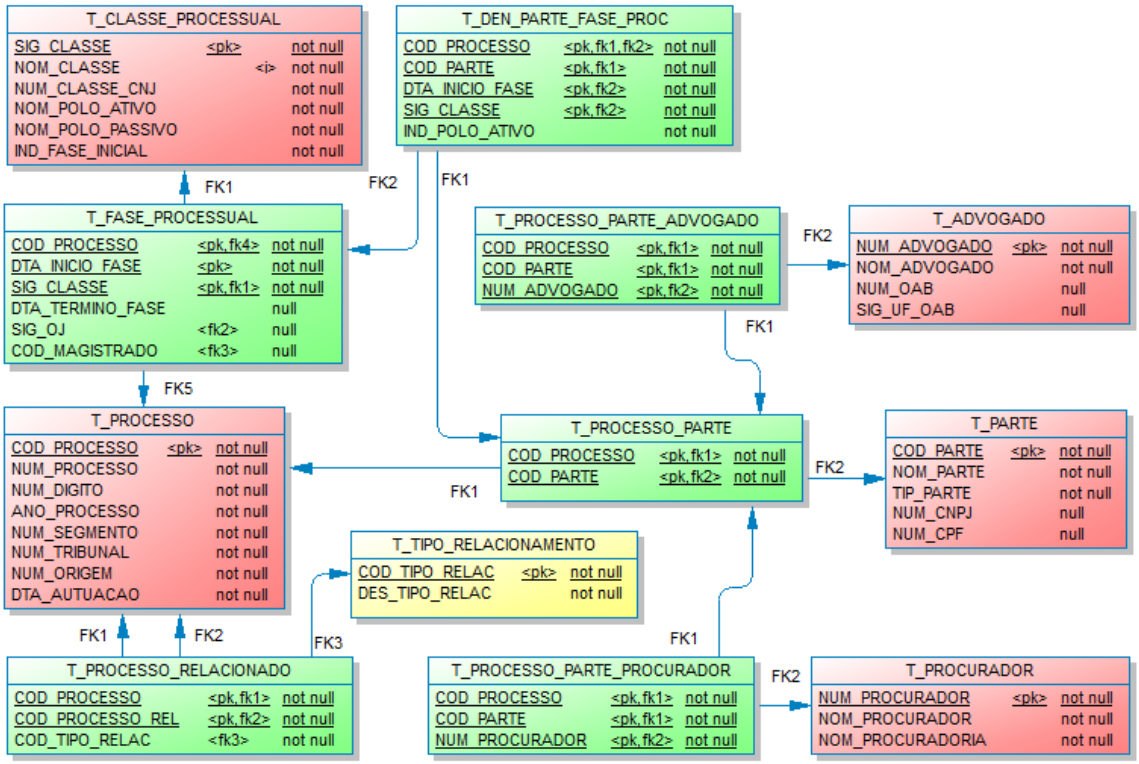

Figura 3.10: Banco de dados de processos judiciais. Modelo de dados em nível lógico.

A arquitetura do LawDisTrA (Figura 3.7) prevê que os dados podem estar distribuídos em três bancos de dados distintos: o banco de dados de processos judiciais, o banco de dados de magistrados e o banco de dados da distribuição. As Figuras 3.10, 3.11 e 3.12 apresentam diagramas do modelo de dados, em nível lógico, para esses bancos de dados. Os bancos projetados são do tipo relacional, coerentes com a tecnologia de banco de dados utilizada no TST. Cada banco será acessado por agentes e sistemas específicos de acordo com os dados nele contido.

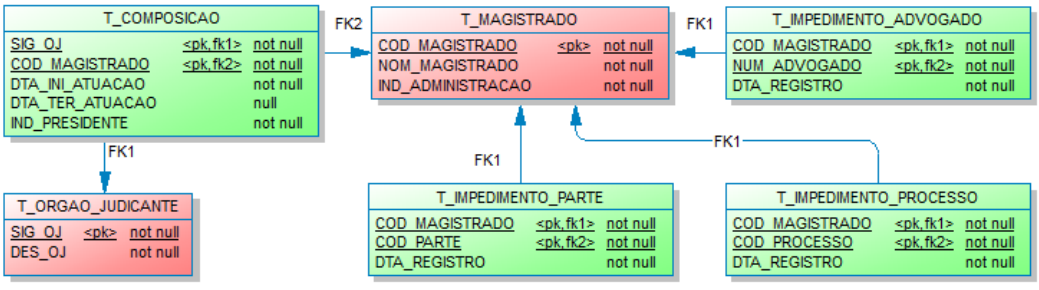

Figura 3.11: Banco de dados de magistrados. Modelo de dados em nível lógico.

O banco de dados de processos judiciais (Figura 3.10) armazena os dados gerais relativos aos processos. Os Agentes de Protocolo buscam nesse banco quais os processos que ainda não foram distribuídos a partir da análise das fases do processo. O módulo

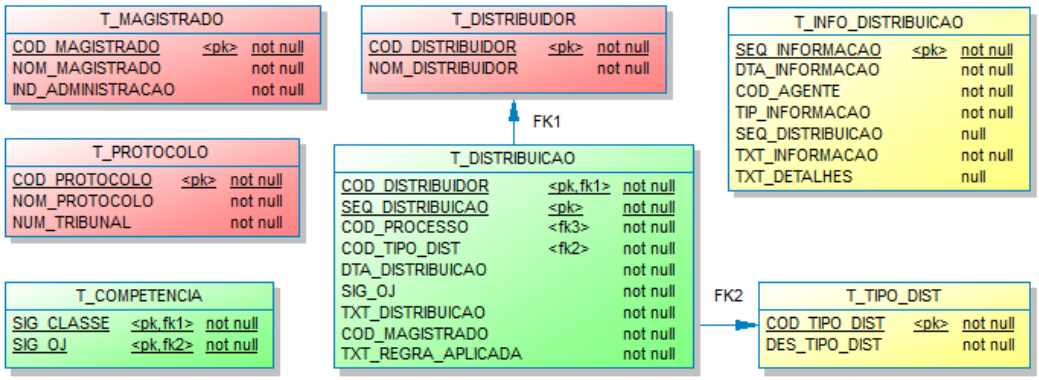

Figura 3.12: Banco de dados da distribuição. Modelo de dados em nível lógico.

Auditor da Distribuição precisa também acessar esses dados para apresentar informações ao usuário.

O banco de dados de magistrados (Figura 3.11) armazena dados relativos aos magistrados e órgãos judicantes do Tribunal. Os Agentes Magistrados utilizarão esse banco de dados para obter e manter as suas configurações, impedimentos dos magistrados, composição dos órgãos judicantes e outras informações.

O AGP utiliza o banco de dados da distribuição (Figura 3.12) como fonte das informações relativas aos demais agentes LawDisTrA. É nesse banco que todos os agentes registram suas percepções e ações para fins de posterior auditoria e transparência. Além disso, toda informação relativa a distribuição de processos realizada pelo LawDisTra será armazenada em tabelas desse banco.

Detalhes acerca dos bancos de dados encontram-se no Anexo II (Tabelas II.1 e II.2) e nos *scripts* de criação das estruturas dos bancos de dados (Anexo III).

3.6 Implementação

A implementação do LawDisTrA utilizou, em grande parte, a linguagem de programação Java, na versão 1.8 do SDK. O Módulo "Auditor da Distribuição" utilizou a linguagem de programação Dart, na versão 1.16 do SDK. Os códigos-fonte e documentação adicional estão disponíveis na Internet em **http://github.com/zidenis/LawDisTrA** e **http://github.com/zidenis/LawDisTrA-Auditor**.

Com objetivo de reduzirmos a complexidade de implementação de um SMA, na implementação do LawDisTrA, utilizamos o JADE[2] na versão 4.4.0. Para utilizar as facilidades do JADE, os agentes devem ser instâncias da classe *jade.core.Agent* e utilizar comportamentos, subclasses de *jade.core.behaviours.Behaviour*, para implementar as funcionalida-

[2]Java Agent DEvelopment Framework – http://jade.tilab.com/

des básicas para realizar suas ações. A Figura 3.13 apresenta a hierarquia e associação de classes do LawDisTra que definem os agentes e seus comportamentos.

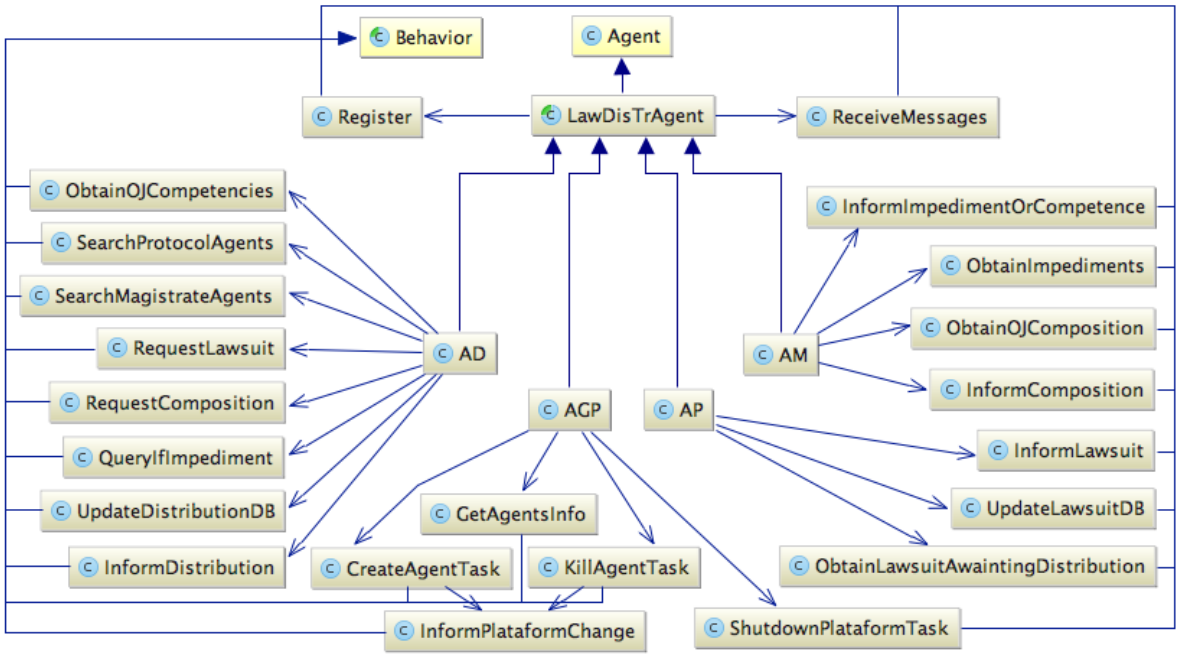

Figura 3.13: Classes que implementam os agentes e seus comportamentos no LawDisTrA.

A classe abstrata br.unb.sma.agents.LawDisTrAgent implementa as funcionalidades comuns aos quatro tipos de agentes (AD, AGP, AP e AM) utilizados no LawDisTrA. Essas funcionalidades incluem o comportamento *br.unb.sma.behaviors.Register*, utilizado para registrar os serviços do agente junto ao DF, e o comportamento *br.unb.sma.behaviors.ReceiveMessages*, utilizado para processar as mensagens recebidas pelos agentes. Além disso, a classe implementa métodos que auxiliam na conexão com os bancos de dados, no registro de *logs* e no carregamento das interfaces gráficas. As descrições de cada um dos comportamentos implementados estão relacionadas na Tabela 3.3.

Tabela 3.3: Comportamentos implementados.

Comportamento	Descrição
CreateAgentTask	Solicita ao AMS que um agente seja iniciado na plataforma.
GetAgentsInfo	Obtém as informações dos agentes disponíveis no sistema.
InformComposition	Envia resposta acerca da participação do AM em OJ para o agente que enviou mensagem tipo inform-composition.
InformDistribution	Informa, acerca de uma distribuição, o AP responsável pelo processo.
InformImpedimentOrCompetence	Responde se o AM é impedido ou competente para julgar um determinado processo.
InformLawsuit	Informa um AD solicitante sobre um processo pendente de distribuição.
InformPlataformChange	Informa que houve mudança no sistema (agentes iniciados ou removidos) de forma que os demais agentes possam se reconfigurar.
KillAgentTask	Solicita ao AMS que um agente específico seja removido da plataforma.
ObtainImpediments	Obtém informações sobre impedimentos do AM.
ObtainLawsuitAwaintingDistribution	Obtém informações de um processo ainda não distribuído.
ObtainOJCompetencies	Obtém as competências (quais tipos de processos podem ser julgados) para um OJ.
ObtainOJComposition	Obtém informações dos OJ de um magistrado.
QueryIfImpediment	Questiona se um AM é impedido para julgar um processo.
ReceiveMessages	Habilita o agente ao recebimento de mensagens assíncronas.
Register	Registra os serviços do agente junto ao catálogo do agente DF.
RequestComposition	Solicita do AM quais OJ ele compõe.
RequestLawsuit	Solicita um processo ainda não distribuído para um AP.
SearchMagistrateAgents	Busca por AM junto ao DF.
SearchProtocolAgents	Busca por AP junto ao DF.
ShutdownPlataformTask	Solicita ao AMS que inicie o desligamento apropriado do sistema.
UpdateDistributionDB	Atualiza o banco de dados de distribuição com informações acerca de uma distribuição realizada.
UpdateLawsuitDB	Atualiza o banco de dados de processos judiciais com informações acerca da movimentação do processo, designando o relator do caso.

O código-fonte de implementação desses comportamentos estão disponíveis em classes no pacote *br.unb.sma.behaviors*. Boa parte desses comportamentos são utilizados para o envio de mensagens e o processamento das mensagens recebidas. Outra grande parte é responsável pelo acesso aos bancos de dados. Por exemplo, a obtenção de informações acerca de um processo pendente de distribuição é realizada utilizando o comportamento

ObtainLawsuitAwaintingDistribution que executa a consulta SQL descrita no Código 3.1.

```
1  -- Obtém informações gerais de um processo não distribuído
2  SELECT p.cod_processo, num_processo, num_digito, ano_processo
3       , num_segmento, num_tribunal, num_origem, dta_autuacao
4    FROM t_processo p
5   INNER JOIN t_fase_processual f
6      ON p.cod_processo = f.cod_processo
7   WHERE num_tribunal = ? -- tribunal de autuação do processo
8     AND cod_magistrado is null
9     AND cod_motivo_redist is null
10  LIMIT 1
```

Código 3.1: Consultas SQL para obtenção de informações de um processo não distribuído.

Para implementação do LawDisTrA também utilizamos:

- PostgreSQL JDBC Driver[3], versão 9.4.1208, para conectar o SMA aos bancos de dados;
- PostgreSQL *database driver for Dart*[4], versão 0.3.3, para permitir o uso dos bancos de dados no módulo Auditor da Distribuição;
- jOOQ[5], versão 3.7.3, para simplificar a interação dos agentes com os bancos de dados;
- gson[6], versão 2.6.2, para transformação de objetos java em *strings* no formato JSON (*JavaScript Object Notation*) e vice-versa;
- Drools[7], versão 6.3.0, como motor de inferência para aplicação das regras de distribuição pelos Agentes de Distribuição. Drools foi escolhido em detrimento de Prolog, pois o mecanismo de raciocínio dedutivo dos agentes utiliza a abordagem de encadeamento para frente, enquanto que Prolog emprega algorítimo de encadeamento para trás. Jess não foi escolhido uma vez que requer licenciamento para fins comerciais;
- log4j[8], versão 1.2.17, para auxílio no registro de *logs* do SMA;
- Woomera[9], versão 1.0.2, para auxiliar na construção do servidor web para o Auditor da Distribuição;

[3]PostgreSQL JDBC Driver – http://jdbc.postgresql.org/
[4]PostgreSQL database driver for Dart – http://pub.dartlang.org/packages/postgresql
[5]jOOQ – http://www.jooq.org/
[6]gson – http://github.com/google/gson
[7]JBoss Drools – http://www.drools.org
[8]Apache log4j – http://logging.apache.org/log4j/1.2
[9]Woomera – http://pub.dartlang.org/packages/woomera

- mustache4dart[10], versão 1.0.10, no processamento de *templates* para construção das páginas HTML pelo Auditor da Distribuição;
- Bootstrap[11], versão 3.3.6, para criação dos componentes de interface de usuário do módulo Auditor da Distribuição.

3.6.1 Regras de Distribuição

As regras de distribuição utilizadas no processo decisório do AD foram especificadas separadamente do código de implementação do agente. Essas regras estão contidas no arquivo *src/br/unb/sma/rules/Distribution.drl* que é carregado no motor de inferência Drools para decidir qual magistrado e órgão judicante deverá receber o processo a ser distribuído a partir dos fatos registrados como percepções dos agentes. Foram implementadas quatro regras no banco de conhecimento do LawDisTrA, a saber:

- Regra 1: distribuição por dependência para processos que estão relacionados a outros que já foram distribuídos;
- Regra 2: distribuição por prevenção para processos em nova fase e que devem ser encaminhados ao mesmo Magistrado e OJ da fase anterior;
- Regra 3: distribuição de Embargos à SDI1 por divergência de decisões entre OJ;
- Regra 4: distribuição ordinária mediante sorteio dos Magistrados que compõem os OJ competentes, que é a forma geral de distribuição. As condições para ativação desta regra são apresentadas no Código 3.2, quais sejam:
  - Linha 4: há um processo com informação de sua classe na fase atual e que não tenha sido distribuído em fase anterior;
  - Linha 5: há informações sobre competências para julgamento de processos pertencentes a classe identificada;
  - Linha 6: uma relação de órgãos judicantes competentes para o julgamento é unificada com a variável *$listSigOJsCompetentes*;
  - Linha 7: informações acerca da composição – magistrados membros – dos órgão judicantes competentes são unificadas com a variável *$listComposicoesOJs*;
  - Linha 8: a relação de magistrados impedidos é unificada com a variável *$magsImpedidos*;
  - Linha 9: o objeto que conterá as informações da distribuição processada é unificado com a variável *$distribuicao*

[10]Mustache for Dart – http://pub.dartlang.org/packages/mustache4dart
[11]Bootstrap – http://getbootstrap.com/

– Linha 10: informações acerca dos impedimentos identificados são unificados com a variável *$listImpedimentos*

```
1  rule ''Regra 4 - Distribuição ordinária mediante sorteio dos Magistrados
       que compõem os OJ competentes''
2  activation-group ''mutex''
3      when
4          $pc : ProcessoCompleto(faseAnterior == null, faseAtual != null,
               $classe : faseAtual.sigClasse)
5          Competencia(sigClasse == $classe)
6          $listSigOJsCompetentes : List() from accumulate(Competencia(
               sigClasse == $classe, $sigOJ : sigOj), collectList($sigOJ))
7          $listComposicoesOJs : Set() from collect(ComposicaoOj(sigOj
               memberOf $listSigOJsCompetentes))
8          $magsImpedidos : Set() from accumulate(Impedimento($codMag :
               codMagistrado), collectSet($codMag))
9          $distribuicao : HistDistribuicao()
10         $listImpedimentos : Set() from collect(Impedimento())
11     then
12         /** Código java que realiza o sorteio de um dos órgãos judicantes
               competentes e de um dos magistrados membros do OJ escolhido. */
13 end
```

Código 3.2: Código com condições para ativação da regra de distribuição n.º 4.

3.6.2 Interfaces de Usuário

Para Bellifemine, Caire, e Greenwood [2], um problema comum que desenvolvedores encontram é fazer com que os agentes JADE interajam com suas respectivas interfaces gráficas e vice-versa. LawDisTra utiliza a tecnologia Java Swing para construção das GUI, onde para cada agente presente na plataforma é criada uma janela gráfica (Figura 3.14). As classes que definem as interfaces dos agentes estão contidas no pacote *br.unb.sma.agents.gui*.

As interfaces gráficas de usuário construídas para o LawDisTrA passaram por diversas mudanças ao longo do seu desenvolvimento, de forma que a versão atual (Figura 3.14) foi muito simplificada em relação à versão apresentada no início deste capítulo (GUI do protótipo, Figura 3.1), na medida em que os agentes foram evoluídos para trabalhar diretamente com os bancos de dados. Além disso, houve a introdução do agente gestor da plataforma (AGP) como forma de permitir um controle facilitado do sistema pelo usuário.

O AGP (Figura 3.15) permite o usuário controlar quais agentes de protocolo, de distribuição e magistrados estarão ativos durante a execução da distribuição, não sendo

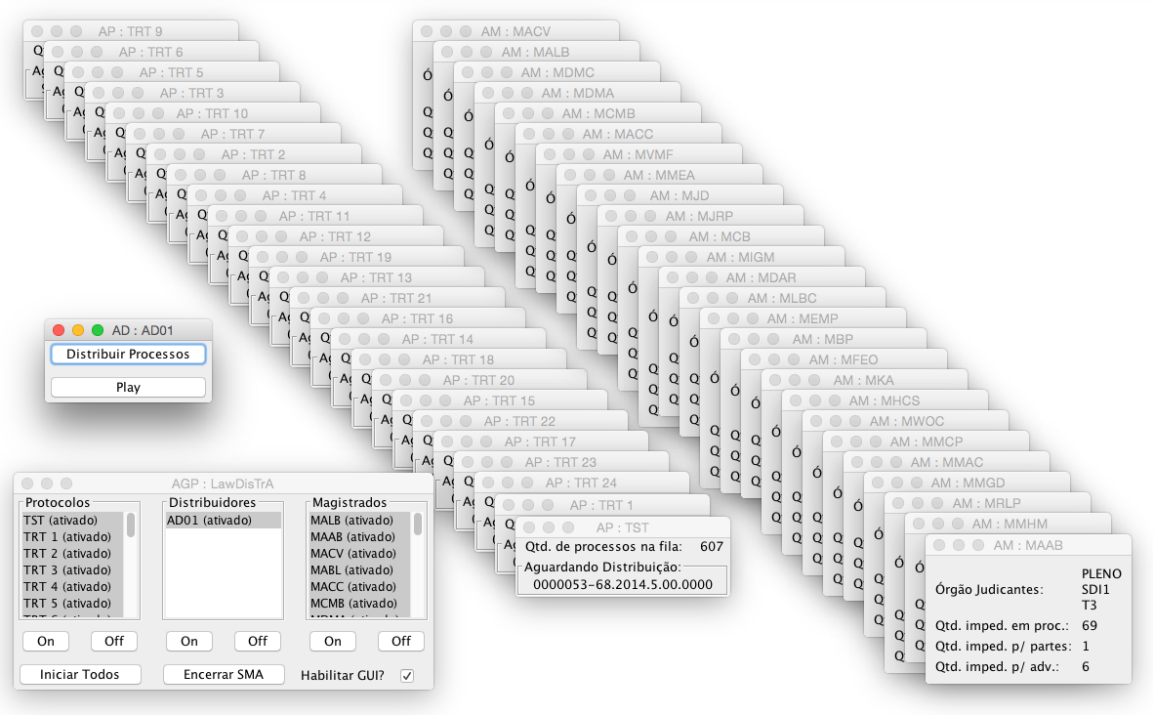

Figura 3.14: Interfaces gráficas desenvolvidas para agentes LawDisTrA.

necessário que todos os agentes estejam ativos para o sistema funcionar, pois os agentes se adaptam para trabalhar de acordo com o ambiente disponível. O AGP também facilita a ativação ou desativação de grupos de agentes selecionados. Serão criadas tantas janelas quantas forem necessárias para representar cada um dos agentes, mas o usuário pode também desabilitar a criação de interfaces gráficas, apenas carregando os agentes na plataforma que passarão então a atuar de forma autônoma.

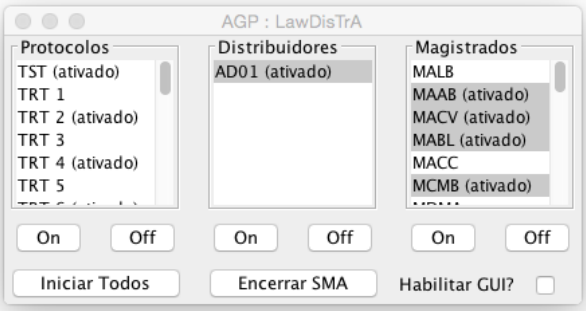

Figura 3.15: GUI desenvolvida para o agente gestor da plataforma.

A GUI do AP (Figura 3.16) é utilizada tão somente para apresentar informações gerais como o Tribunal que ele representa, a quantidade de processos aguardando distribuição e a numeração única do próximo processo a ser distribuído.

Figura 3.16: GUI desenvolvida para os agentes de protocolo.

A GUI do AM (Figura 3.17) apresentam informações gerais sobre um magistrado, tais como: seu código de identificação, a relação dos órgãos judicantes dos quais ele é membro e a quantidade de impedimentos encontrados para processos, partes e advogados.

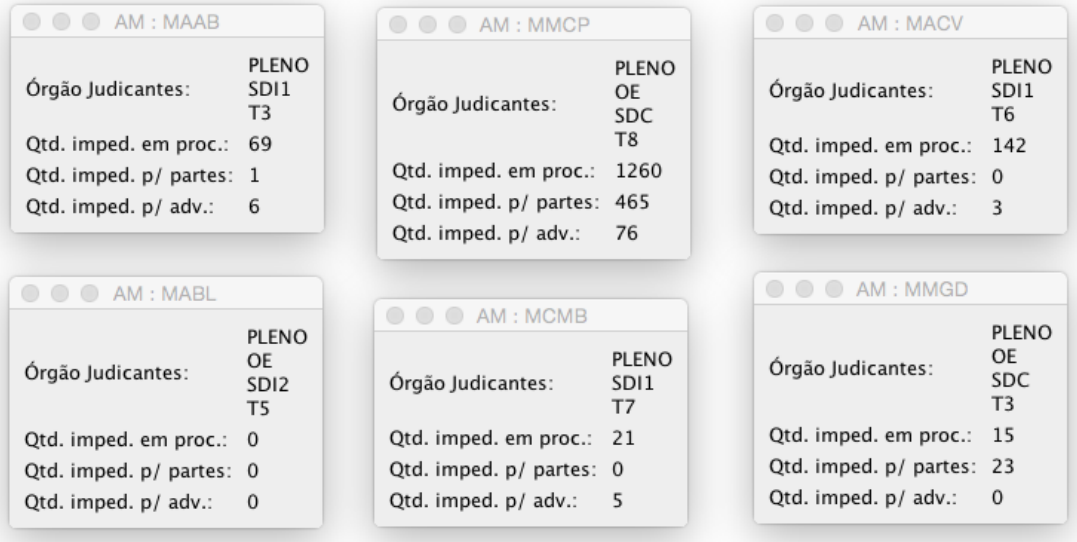

Figura 3.17: GUI desenvolvida para os agentes magistrados.

Em princípio, para os agentes de distribuição, não seriam necessárias quaisquer interfaces gráficas, uma vez que sua operação deveria ser completamente automatizada. Por exemplo, o agente poderia realizar as suas tarefas segundo uma periodicidade definida ou a partir de mensagens recebidas de outros agentes (ocorrência de eventos). Mas, uma interface simples (Figura 3.18) foi concebida para que o usuário pudesse controlar ma-

nualmente a iniciação ou a parada da tarefa de distribuição, o que também facilitou a realização dos experimentos e validação o sistema. Essa interface é composta de apenas dois botões, um para iniciar a distribuição de um lote de processos (um processo para cada agente de protocolo iniciado) e outro para iniciar a distribuição de forma contínua até que não haja mais nenhum processo para distribuir para os agentes de protocolo ativados na plataforma.

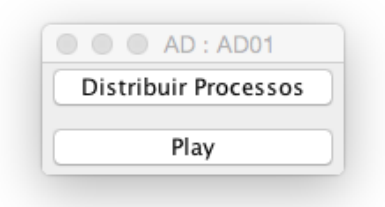

Figura 3.18: GUI desenvolvida para o agente de distribuição.

Módulo Auditor da Distribuição

As interfaces do módulo Auditor da Distribuição permitem a consulta aos dados dos processos e da distribuição de uma forma amigável, o que torna o sistema dotado de características desejáveis para a transparência da informação. Para maximizar essa capacidade, optou-se por construir essa interface em plataforma *web* (Figura 3.19), utilizando o mesmo formato da consulta no Sistema de Consulta Processual do TST.

Figura 3.19: Interface de consulta de processos pelo Auditor da Distribuição.

A Figura 3.20 ilustra uma consulta realizada para obter informações do processo AIRR 3019-56.2012.5.18.102. Podemos observar que se trata de um Agravo de Instrumento em Recurso de Revista que foi distribuído para a 5ª Turma (Órgão Judicante), sendo o Ministro Guilherme Augusto Caputo Bastos (MCB) responsável pela relatoria. Verificamos

também que esse processo passou por duas tentativas de distribuição, a de n.º 2121 e a de n.º 2292, sendo que a distribuição n.º 2121 foi cancelada pois o magistrado sorteado (MMCP) naquele momento possuía impedimentos relativos à uma parte e um advogado do processo. Já a distribuição n.º 2292 foi efetivamente realizada, momento em que a 5ª Turma e o ministro MCB foram sorteados.

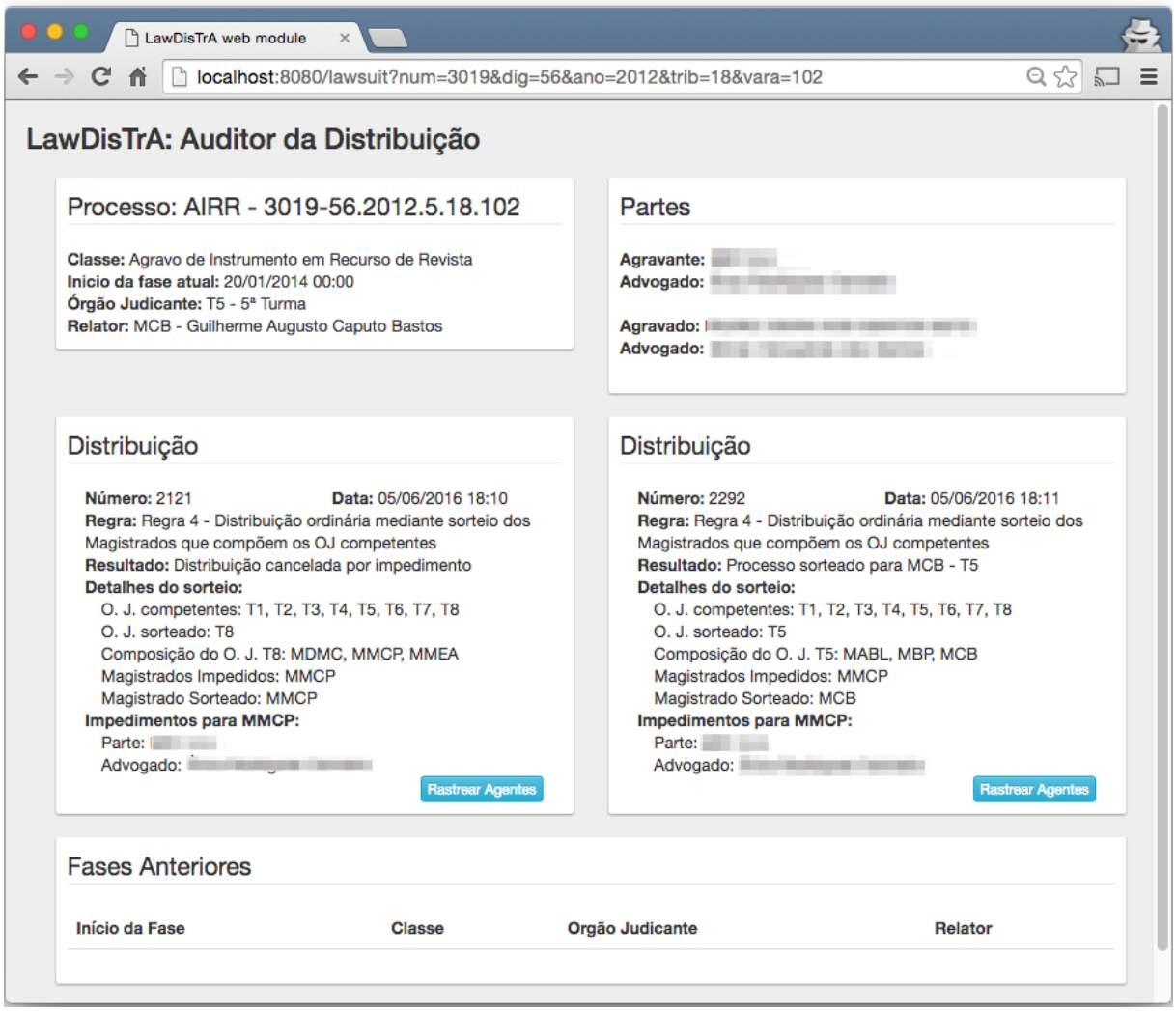

Figura 3.20: Resultado da consulta de um processo.

Para fins de promover a transparência, o sistema informa não somente o resultado da distribuição, mas também detalhes como: qual regra foi utilizada para distribuição, a data de realização da distribuição, os órgãos judicantes competentes, a composição desses OJ, além da existência de magistrados impedidos com detalhes de cada um dos impedimentos. Ainda, é possível rastrear os comportamentos ativados e mensagens trocadas entre os agentes do LawDisTrA que resultaram na distribuição específica, bastante que o usuário acione o botão *[rastrear agentes]*.

Por exemplo, a Figura 3.21 apresenta o rastreamento da distribuição n.º2292. As ações dos agentes – relativas a essa distribuição e que foram registradas no banco de dados – serão apresentadas na interface de forma que o auditor possa obter detalhes sobre o procedimento. Para essa distribuição específica, os agentes gravaram 93 registros no banco de dados. Esse número varia de acordo com o número de agentes disponíveis na plataforma e o tipo de distribuição realizada.

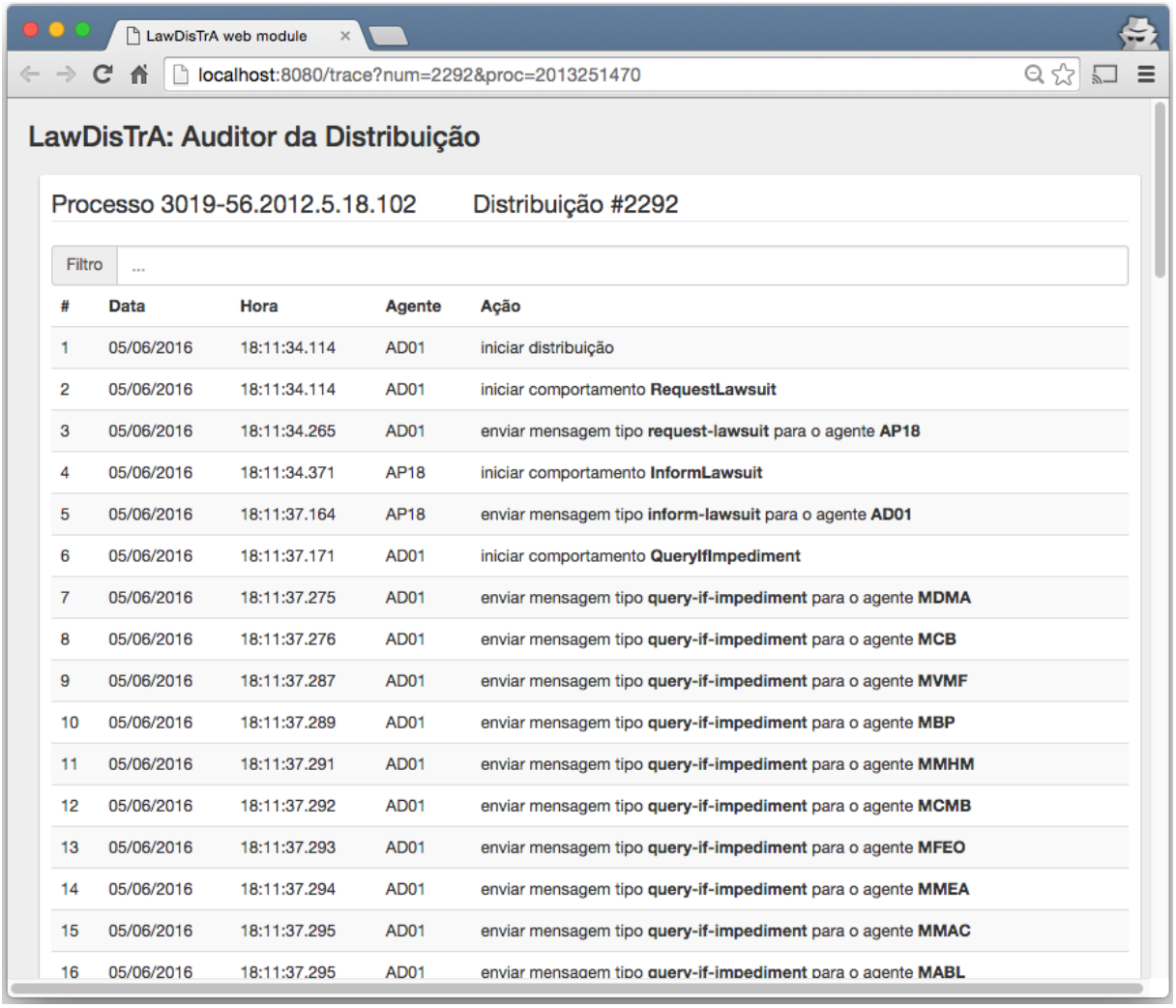

Figura 3.21: Rastreamento das ações dos agentes.

Para atestar o funcionamento do SMA, experimentos foram realizados, conforme descrito no Capítulo 4.

Capítulo 4

Experimentação e Resultados

A experimentação com o LawDisTrA passou pela construção do bancos de dados descrito no Anexo III) com informações de um total de 309.332 processos judiciais eletrônicos para serem distribuídos pelo sistema, o que é um pouco mais que o volume total de processos recebidos pelo TST no período de um ano. O TST recebeu 309.033 processos em 2014 e 291.454 em 2015, conforme estatística de movimentação processual divulgada[1].

A Tabela 4.1 apresenta a quantidade de processos contidos no banco de dados de experimentação de acordo com a classe processual. Embora os dados contidos no banco de experimentação não sejam iguais aos constantes das bases de dados do TST, buscou-se manter fidelidade quanto aos tipos e quantidades de processos recebidos pelo TST, onde a grande maioria dos processos são dos tipos Agravo de Instrumento em Recurso de Revista e Recurso de Revista, conforme apresentado anterior na Figura 2.9.

Em nossa simulação, foram ativados 55 agentes na plataforma, sendo:

- 25 agentes de protocolo, cada um representado um dos Tribunais do Trabalho (TST mais 24 TRT);
- 27 agentes magistrados, cada um representando um Ministro do TST;
- 1 agente de distribuição que execução as atividades de distribuição de forma centralizada;
- 1 agente JADE DF;
- 1 agente JADE AMS.

Com todos os agentes carregados, LawDisTrA foi capaz de distribuir o volume total de 309.332 processos em 28 horas, 50 minutos e 18 segundos de trabalho ininterrupto, ou seja, uma vazão de aproximadamente 3,15 processos distribuídos por segundo. Do total,

[1]Movimentação Processual no TST: http://www.tst.jus.br/tribunal-superior-do-trabalho1

94,78% dos processos foram distribuídos mediante sorteio, conforme Tabela 4.2, o que já era esperado, uma vez que esta é a regra geral a ser aplicada.

Tabela 4.1: Quantidade de processos no banco de dados por classe.

Classe do Processo	**Qtd.**	**Freq.**
Ação Rescisória	169	0,055%
Ação Trabalhista - Rito Ordinário	1	0%
Agravo	2	0,001%
Agravo de Instrumento em Recurso de Revista	221.750	71,687%
Agravo de Instrumento em Recurso Ordinário	95	0,031%
Agravo Regimental	2	0,001%
Argüição de Inconstitucionalidade	1	0%
Cautelar Inominada	326	0,105%
Conflito de Competência	220	0,071%
Dissídio Coletivo	4	0,001%
Dissídio Coletivo de Greve	6	0,002%
Embargos	1	0%
Embargos de Declaração	7	0,002%
Habeas Corpus	1	0%
Mandado de Segurança	107	0,035%
Pedido de Providências	1	0%
Petição	10	0,003%
Recurso de Revista	69.553	22,485%
Recurso de Revista com Agravo	13.419	4,338%
Recurso Extraordinário com Agravo	218	0,07%
Recurso Ordinário	3.439	1,112%

Tabela 4.2: Quantidade de processos distribuídos por regra utilizada.

Regra n.	**Descrição**	**Qtd.**	**Freq.**
1	Distribuição por dependência para processos que estão relacionados a outros que já foram distribuídos	475	0,15%
2	Distribuição por prevenção para processos em nova fase e que devem ser encaminhados ao mesmo Magistrado/OJ da fase anterior	15.580	5,04%
3	Distribuição de Embargos à SDI1 por divergência de decisões entre OJ	44	0,01%
4	Distribuição ordinária mediante sorteio dos Magistrados que compõem os OJ competentes	293.233	94,80%

Grande parte dos processos foram sorteados para os órgãos judicantes do tipo Turma, conforme Tabela 4.3. Esse também é o comportamento esperado, considerando que são as turmas que possuem competência para julgar os processos do tipo Recurso de Revista

e Agravo de Instrumento em Recurso de Revista que correspondem a mais de 94% dos processos cadastrados no banco de dados.

Tabela 4.3: Quantidade de processos distribuídos por Órgão Judicante.

Órgão Judicante	**Qtd.**	**Freq.**
1ª Turma	37655	12,173%
2ª Turma	37518	12,129%
3ª Turma	38492	12,444%
4ª Turma	38521	12,453%
5ª Turma	37867	12,242%
6ª Turma	38501	12,446%
7ª Turma	37805	12,221%
8ª Turma	38136	12,329%
Órgão Especial	21	0,007%
Seção Especializada em Dissídios Coletivos	35	0,011%
Subseção I Especializada em Dissídios Individuais	689	0,223%
Subseção II Especializada em Dissídios Individuais	4042	1,307%
Tribunal Pleno	50	0,016%

Apesar de boa parte da distribuição ser realizada por sorteio dos magistrados, a distribuição foi uniforme entre os magistrados, conforme Tabela 4.4. Já era esperado que os Ministros 12, 15 e 25 recebessem bem menos processos, pois eles compõem atualmente a Administração do TST (presidente, vice-presidente e corregedor) e, por isso, não integram as turmas e nem recebem novos processos.

O experimento registrou 27.574.903 *logs* de percepções e ações dos agentes. São essas informações que são expostas pelo Módulo Auditor de forma a permitir o rastreamento dos agentes para cada uma das distribuições realizadas. Por meio da experimentação foi possível verificar que LawDisTrA realizou a distribuição de processos da forma desejada, com resultados semelhantes aos encontrados no TST, mas cumprindo com os requisitos de transparência.

Tabela 4.4: Quantidade de processos distribuídos por Magistrado.

#	Magistrado	Qtd.	Freq.
1	MAAB	12.634	4,08%
2	MABL	12.559	4,06%
3	MACC	12.817	4,14%
4	MACV	12.966	4,19%
5	MALB	13.588	4,39%
6	MBP	12.126	3,92%
7	MCB	12.845	4,15%
8	MCMB	12.611	4,08%
9	MDAR	13.208	4,27%
10	MDMA	13.078	4,23%
11	MDMC	13.157	4,25%
12	MEMP	608	0,20%
13	MFEO	12.381	4,00%
14	MHCS	12.556	4,06%
15	MIGM	344	0,11%
16	MJD	12.910	4,17%
17	MJRP	12.760	4,13%
18	MKA	12.961	4,19%
19	MLBC	13.327	4,31%
20	MMAC	13.436	4,34%
21	MMCP	11.581	3,74%
22	MMEA	13.007	4,20%
23	MMGD	12.989	4,20%
24	MMHM	12.778	4,13%
25	MRLP	382	0,12%
26	MVMF	13.103	4,24%
27	MWOC	12.620	4,08%

4.1 Características de Transparência

Nesta seção, apresentaremos uma discussão acerca das características de transparência relacionadas no *Transparency SIG* (Figura 2.11) e que percebemos como relevantes à questão da distribuição de processos judiciais. Pretende-se avaliar empiricamente, na ausência de métricas objetivas na literatura, de que forma o LawDisTrA implementa mecanismos para promover transparência.

- Portabilidade: a solução foi implementada em linguagem Java, favorecendo sua portabilidade uma vez que seu código pode ser executado pela máquina virtual Java em diferentes plataformas. Utilizou-se o *framework* JADE que foi construído com

base no padrão FIPA que busca a portabilidade de agentes em diversas arquiteturas de SMA;

- Publicidade: todos os dados relativos à distribuição são disponibilizados pela interface do módulo Auditor da Distribuição a partir da consulta pela numeração única do processo. As regras utilizada para a distribuição também são públicas e de fácil acesso, pois encontram-se em arquivo Drools em repositório de código de livre acesso;

- Adaptabilidade: agentes podem ser ativados e desativados sem que isso afete o processo de distribuição como um todo. É possível modificar facilmente as regras de distribuição que estão em arquivo Drools específico, evitando a necessidade de modificação do código-fonte da lógica dos agentes;

- Amigabilidade: foram construídas interfaces gráficas para facilitar o uso do sistema e a apresentação das informações. O agente AGP foi construído especialmente com esse propósito, de forma que o usuário possui um mecanismo amigável para manter o sistema, sem a necessidade de conhecer detalhes de implementação do sistema e lidar com as interfaces mais complexas do JADE;

- Desempenho: LawDisTrA é capaz de processar automaticamente em aproximadamente 29 horas a quantidade de processos equivalente a um ano de processos recebidos pelo TST;

- Simplicidade: o uso de SMA torna mais complexo o processo de desenvolvimento e a própria solução. Buscou-se a simplicidade ao evitar a inclusão de agentes que não estão presentes no mundo real para a resolução do problema;

- Consistência: os resultados das distribuições realizadas pelo LawDisTrA são consistentes com os resultados obtidos pelo sistema de distribuição utilizado no TST;

- Divisibilidade: cada agente decompõe o processo de distribuição em passos específicos, independentes e únicos. Cada agente representa uma entidade específica do mundo real. Os relacionamentos entre os agentes foram especificados;

- Extensibilidade: é possível criar novos agentes e introduzi-los no sistema sem interromper os procedimentos em curso. A introdução de novas regras de distribuição também é facilitada em razão do uso do Drools;

- Auditabilidade: o Módulo Auditor do LawDisTrA foi construído especialmente para atender esta característica, apresentando vantagens em relação a solução atualmente empregada no TST.

- Controlabilidade: o controle de quais agente estão ativos e, dessa forma, participarão do processo, é realizada mediante interface do agente AGP. As informações de quais agentes estavam ativos e participaram das distribuições são todas registradas no banco de dados, permitindo identificar o início e termino de cada atividade, pois todas as ações realizadas pelos agentes também são registradas;
- Explicável: cada distribuição realizada é explicável a partir das informações armazenadas no banco de dados. É possível explicar quando a distribuição foi realizada, qual regra foi aplicada, quais os órgãos judicantes e magistrados participaram da atividade e quais impedimentos de magistrados foram considerados;
- Rastreabilidade: é possível rastrear quais agentes foram envolvidos em cada processo de distribuição, identificando de que forma, porque e quando cada um agiu.

Capítulo 5

Conclusões e Trabalhos Futuros

Este trabalho apresentou o projeto, a construção e a validação de um SMA para tratar a distribuição de processos judiciais eletrônicos de forma transparente. Foi realizado o levantamento dos requisitos do sistema a partir do estudo de caso do TST. O desenvolvimento da solução, utilizando a metodologia de desenvolvimento Tropos, permitiu um melhor entendimento de como um sistema de informação corporativo tradicional pode ser construído utilizando o paradigma de agentes. O uso de modelos Tropos se mostrou interessante para documentar e comunicar os requisitos do SMA, mas, considerando que a metodologia não é amplamente conhecida, sua eficiência pode ser questionada no caso da continuidade do projeto por novos analistas. Por outro lado, o paradigma de agentes favorece a decomposição do problema ao naturalmente representar a forma como a sociedade humana e as organizações são estruturadas e interagem.

Acreditamos que sistemas multiagentes possuem características – portabilidade, adaptabilidade, extensibilidade, comunicação – que favorecem o desenvolvimento de *softwares* bastante flexíveis e seu uso tem importância diretamente proporcional a complexidade do sistema a ser desenvolvido, sendo a arquitetura uma boa candidata para sistemas grandes e distribuídos. A complexidade advinda do emprego de sistemas multiagentes pode ser amenizada pela utilização de *middlewares*, como a plataforma JADE, que facilitam sua implementação, mas requer programadores mais especializados. Recomenda-se uma análise detalhada para verificar se a complexidade adicional no desenvolvimento é compensada pela flexibilidade alcançada em projetos específicos.

No sistema desenvolvido – LawDisTrA – buscou-se introduzir características que ampliassem a transparência do processo e da informação tratada. As organizações públicas estão crescentemente buscando formas de automatizar seus processos e os serviços que provêm aos cidadãos. A transparência parece ser um dos fatores cruciais para o aumento da eficiência dessas instituições, promovendo níveis de controle e aumento de sua performance, fortalecendo as iniciativas de governo eletrônico. Um dos resultados deste trabalho

foi um artigo que abordou a implementação de características de transparência no processo de distribuição, publicado na Revista Brasileira de Sistemas de Informação (iSys, Edição Especial: Governo Eletrônico) [1].

A transparência da informação é especialmente promovida em sistemas multiagentes se: os agentes desenvolvidos registrarem, em bancos de dados, suas percepções, decisões e ações; e o sistema possua uma interface de usuário que promova a publicidade, a acessibilidade, a portabilidade, a simplicidade, a amigabilidade, a uniformidade e a auditabilidade das informações registradas. LawDisTrA busca implementar essas características por meio de transparência na interação entre os agentes do sistema e do módulo Auditor da Distribuição. A separação das regras de negócio (regras da distribuição) da lógica de programação dos agentes, mediante construção de agentes lógicos que empregam um motor de inferência, ajudam na manutenção e evolução do sistema, ao tempo que também promove a transparência do processo.

A implantação do SMA desenvolvido, em substituição ao atualmente utilizado em Tribunais como o TST, requer que sejam implementadas funcionalidades adicionais, tais como: atribuição de quantidades de processos em percentuais diferentes por magistrado, de acordo com papeis atribuídos a eles; uso de saldos de distribuição de processos por magistrado para fins de compensação e equilíbrio da carga de trabalho, considerando a quantidade e a complexidade dos processos; tratamento automatizado de ocorrências de redistribuições; percepção e adaptação a eventos como férias, licenças e substituições dos magistrados. É preciso avaliar até onde o sistema pode evoluir para tratar a distribuição de forma automática sem a intervenção de humanos.

Um possível trabalho futuro é o aprimoramento no comportamento inteligente dos agentes. A autonomia e inteligência dos agentes pode ser melhorada para que, por exemplo, os agentes magistrados possam investigar e descobrir novos impedimentos dos magistrados a partir de mineração de dados na Internet ou no teor de processos judiciais existentes nas bases de dados, o que envolveria o processamento de linguagem natural.

Por fim, entendemos que o uso de sistemas multiagentes para construção de sistemas de informação corporativos é adequado quando o sistema a ser desenvolvido requerer portabilidade, escalabiliade e autonomia. Os artefatos de *software* e documentação, produzidos durante o desenvolvimento deste trabalho, estão publicamente disponíveis na Internet [1] [2].

[1] http://github.com/zidenis/LawDisTrA
[2] http://github.com/zidenis/LawDisTrA-Auditor

Referências

[1] Denis José Sousa Albuquerque, Vanessa Tavares Nunes, Claudia Cappelli, e Célia Ghedini Ralha. Implementing e-government processes distribution with transparency using multi-agent systems. *iSys - Revista Brasileira de Sistemas de Informação*, Edição Especial: Governo Eletrônico:118–138, 2016. 37, 73

[2] Fabio Bellifemine, Giovanni Caire, e Dominic Greenwood. *Developing multi-agent systems with JADE*, volume 7. John Wiley & Sons, 2007. 13, 18, 60

[3] Brasil. Lei n. 13.105, de 16 de março de 2015. Código de Processo Civil. `http://www.planalto.gov.br/ccivil_03/_Ato2015-2018/2015/Lei/L13105.htm`, 2015. [Online; acessado em 15-10-2015]. 22, 26, 32

[4] Frances M. T. Brazier, Barbara M Dunin-Keplicz, Nicholas R. Jennings, e Jan Treur. Desire: Modelling multi-agent systems in a compositional formal framework. *Int. Journal of Cooperative Information Systems*, 6(1):67–94, 1997. 15

[5] Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, e John Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004. 1, 15

[6] Paul Browne. *JBoss Drools business rules*. Packt Publishing Ltd, 2009. 11

[7] Andréa Pinto Castor e Freire Brelaz de Castro. Rastreamento de requisitos no processo de desenvolvimento de software orientado a agentes. Dissertação de mestrado, Universidade Federal de Pernambuco, 2004. 13, 14

[8] Jaelson Castro, Fernanda Alencar, e Carla Silva. Engenharia de software orientada a agentes. In *Atualizações em Informática*, pages 245–282. PUC-Rio, 2006. 12, 13, 16, 17

[9] Jaelson Castro, Manuel Kolp, e John Mylopoulos. Developing agent-oriented information systems for the enterprise. In *Enterprise Information Systems II*, pages 7–20. Springer, 2001. 1

[10] Peter Pin-Shan Chen. The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, 1(1):9–36, 1976. 53

[11] Conselho Nacional de Justiça. Distribuição. `http://www.cnj.jus.br/wikipje/index.php/Distribuicao`, 2015. [Online; accessado em 13/09/2015]. 2, 3

[12] José Augusto Delgado. *A supremacia dos princípios nas garantias processuais do cidadão*. Jurid Vellenich, 1994. 21

[13] FIPA. FIPA agent management specification. `http://www.fipa.org/specs/fipa00023/SC00023K.html`, 2004. [Online; acessado em 06/12/2015]. 19, 20

[14] Charles L Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial intelligence*, 19(1):17–37, 1982. 11

[15] Shaw Green, Leon Hurst, Brenda Nangle, e Pádraig Cunningham. Software agents: a review. Technical report, Trinity College Dublin, Department of Computer Science, 1997. 13

[16] Burkart Holzner e Leslie Holzner. *Transparency in global change: the vanguard of the open society*. Tech Report. University of Pittsburgh, 2006. 32

[17] Jomi Fred Hübner e Jaime Simão Sichman. Organização de sistemas multiagentes. *III Jornada de Mini-Cursos de Inteligência Artificial*, 8:247–296, 2003. 12

[18] Peter Jackson. Introduction to expert systems. 1986. 9

[19] Nicholas R Jennings. On agent-based software engineering. *Artificial Intelligence*, 117(2):277–296, 2000. 1

[20] Julio Cesar Sampaio do Prado Leite e Claudia Cappelli. Software transparency. *Business & Information Systems Engineering*, 2(3):127–139, 2010. x, 32, 33, 35, 36

[21] George Marmelstein Lima. Desrespeitos a regra da livre distribuição. *Revista do Centro de Estudos Judiciários*, 6(18):94–103, 2002. 2, 3

[22] Rafael Nascimento. Modelagem de software orientado a agentes utilizando a metodologia tropos. `http://www.devmedia.com.br/modelagem-de-software-orientado-a-agentes-utilizando-tropos/31683#ixzz3rePUTdYE`, 2010. [Online; acessado em 15/10/2015]. x, 16

[23] Peter Norvig e Stuart Russell. *Inteligência Artificial, 3ª Edição*. Elsevier Brasil, 2014. x, 7, 8, 9, 10, 11, 12

[24] Lin Padgham e Michael Winikoff. Prometheus: A methodology for developing intelligent agents. In *Agent-oriented software engineering III*, pages 174–185. Springer, 2002. 15

[25] David Lorge Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, 1972. 14

[26] Ismênia Galvão Lourenço da Silva e Freire Brelaz de Castro. Projeto e implementação de sistemas multi-agentes: O caso tropos. Dissertação de mestrado, Universidade Federal de Pernambuco, 2005. x, 18, 19, 20

[27] Supremo Tribunal Federal. Medida Cautelar em Mandado de Segurança 33.921. https://www.stf.jus.br/portal/autenticacao/abrirDocumento.asp?tipo=documentoGeral&numero=9950632, 2015. [Online; acessado em 23/01/2016]. 2, 3

[28] Katia P Sycara. Multiagent systems. *AI magazine*, 19(2):79, 1998. 12

[29] Yasuyuki Tahara, Akihiko Ohsuga, e Shinichi Honiden. Agent system development method based on agent patterns. In *Software Engineering, 1999. Proceedings of the 1999 International Conference on*, pages 356–367. IEEE, 1999. 7

[30] Tribunal Superior do Trabalho. Regimento Interno. Resolução Administrativa n. 1295/2008. Diário da Justiça da República Federativa do Brasil, Brasília, DF, 8 maio 2008, p. 20-30. 28, 30, 31, 32

[31] Tribunal Superior do Trabalho. Relatório geral da justiça do trabalho. http://www.tst.jus.br/documents/10157/5ea7ea0c-a245-4768-87c2-d1c72a4b8344, 2015. [Online; acessado em 07/03/2016]. x, 28

[32] Tribunal Superior do Trabalho. Movimentação processual do Tribunal Superior do Trabalho. http://www.tst.jus.br/documents/10157/350f2e53-bb87-4df0-9a90-126e6be1a85d, 2016. [Online; acessado em 07/03/2016]. x, 29

[33] Gerhard Weiss. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999. 13

[34] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009. 7, 10, 14, 15, 18

[35] Michael Wooldridge, Nicholas R Jennings, e David Kinny. The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and multi-agent systems*, 3(3):285–312, 2000. 15

[36] Eric Yu, Paolo Giorgini, Neil Maiden, e John Mylopoulos. Social modeling for requirements engineering: An introduction. *Social Modeling for Requirements Engineering*, pages 3–10, 2011. 12, 15

Anexo I

Trechos do Regimento Interno do TST que Identificam Requisitos para a Distribuição de Processos

Art. 35. Compete ao Presidente:

XXV – determinar a distribuição dos processos, segundo as regras regimentais e resoluções administrativas, aos Ministros do Tribunal, e dirimir as controvérsias referentes à distribuição;

XXVI - despachar as desistências dos recursos e das ações, quando se referirem a processo pendente de distribuição na Corte, bem como os demais incidentes processuais suscitados;

Art. 37. O Vice-Presidente participa das sessões dos órgãos judicantes do Tribunal, exceto de Turma, não concorrendo à distribuição de processos.

Art. 38. O Corregedor-Geral da Justiça do Trabalho não concorre à distribuição de processos, participando, quando não estiver ausente em função corregedora, das sessões dos órgãos judicantes da Corte, exceto de Turmas, com direito a voto.

Art. 81. Compete ao Presidente de Turma:

Parágrafo único. Em face da atribuição contida no inciso IX do presente artigo, o Presidente de Turma receberá 10% (dez por cento) a menos de processos distribuídos, respeitada a proporção quanto às classes processuais de competência da Turma.

Art. 89. Os processos de competência do Tribunal serão distribuídos por classe, observada a competência e composição dos órgãos judicantes, assim como a ordem cronológica do seu ingresso na Corte, concorrendo ao sorteio todos os Ministros, excetuados os membros da direção.

Parágrafo único. Não haverá distribuição de processos aos Ministros nos sessenta dias que antecederem a jubilação compulsória, nem a partir da data da apresentação do pedido de aposentadoria ao Órgão Especial.

Art. 90. No período correspondente às férias dos Ministros, não haverá distribuição de processos, exceto os de dissídio coletivo, mandado de segurança, ações cautelares e habeas corpus.

Art. 91. Todos os processos recebidos no Tribunal, independentemente da classe a que pertencerem, serão distribuídos logo após os registros e as formalidades necessárias à sua identificação.

Parágrafo único. Será fornecido a cada Ministro, por ocasião da distribuição, documento escrito ou transmissão computadorizada, contendo todos os dados da distribuição que lhe coube.

Art. 92. As redistribuições autorizadas expressamente neste Regimento serão feitas no âmbito da Secretaria do Colegiado em que tramita o processo, pelo respectivo Presidente, observada a compensação e publicidade, devendo ser fornecidos a cada Ministro integrante do Colegiado, mediante documento escrito ou transmissão computadorizada, todos os dados do repasse de feitos.

Art. 93. Os processos distribuídos aos Ministros permanecerão a eles vinculados, ainda que ocorram afastamentos temporários, ressalvada a hipótese de mandados de segurança originários, processos de dissídio coletivo, ações cautelares e habeas corpus que, a juízo da parte, reclamem solução inadiável. Nesse caso, ausente o Relator por mais de três dias, poderá ocorrer a redistribuição, observada a posterior compensação.

§ 1º Os processos de competência das Turmas, na hipótese de o Relator afastar-se temporariamente do Tribunal por período superior a 30 dias ou definitivamente, serão atribuídos ao Desembargador convocado para substituí-lo. Cessada a convocação, o Relator ou o novo Ministro Titular da cadeira receberá os processos, não solucionados, atribuídos ou distribuídos ao Desembargador convocado.

§ 2º Os processos de competência das Seções Especializadas serão redistribuídos no âmbito dos respectivos Órgãos fracionários, desde que não haja remoção de Ministro para a cadeira vaga. O Ministro que vier a ocupar a cadeira vaga receberá, em igual número, mediante compensação, o montante de processos redistribuídos por ocasião da vacância da cadeira.

§ 3º Os processos de competência do Órgão Especial, em caso de afastamento definitivo do Relator, serão atribuídos ao Ministro que o suceder no Órgão. Na hipótese de afastamento temporário, o Relator permanecerá vinculado a tais processos, observada, porém, a regra do art. 93, “caput”, do RITST.

Art. 94-B. O relator que se afastar definitivamente da Turma ou da Seção Especializada, por motivo de remoção, receberá no órgão para o qual se removeu os processos vinculados ao antecessor em que este ainda não apôs o visto.

Parágrafo único. Na hipótese de remoção de Turma, o Ministro que se removeu receberá no novo órgão, em compensação, a diferença entre o acervo processual deixado na Turma de origem, ao se remover, e o que recebeu na nova cadeira, observadas as classes processuais.

Art. 96. Se o afastamento do Relator for definitivo, em decorrência de haver assumido cargo de direção do Tribunal, seus processos serão atribuídos, conforme o caso, ao Juiz convocado, ou ao Titular da cadeira, que, em lugar do afastado, vier a integrar a Turma, inclusive em relação aos agravos e aos embargos de declaração.

Parágrafo único. Os processos de competência das Seções Especializadas serão atribuídos ao Titular da cadeira que, em lugar do afastado, vier a integrar a Seção Especializada, inclusive em relação aos agravos e aos embargos de declaração.

Art. 98. O Colegiado que conhecer do processo terá jurisdição preventa para o julgamento dos recursos posteriores interpostos no mesmo processo, observada a competência.

Parágrafo único. O processo que tramita na fase de execução será distribuído ao Ministro a quem coube a relatoria na fase de conhecimento, ou a quem o tenha substituído ou sucedido, devendo os processos tramitar conjuntamente, sempre que possível

Art. 99. O processo já apreciado pelo Órgão Especial ou por uma das Seções Especializadas, retornando a novo exame, será distribuído ao mesmo Colegiado e ao mesmo Relator ou Redator do acórdão. Na ausência definitiva do Relator ou do Redator do acórdão anterior, o processo será distribuído ao novo titular que vier a integrar o órgão prevento.

Parágrafo único. O processo já apreciado por uma das Turmas será distribuído ao mesmo Colegiado e ao mesmo Relator ou Redator do acórdão. Na ausência definitiva do Relator ou do Redator do acórdão anterior, o processo será distribuído ao Juiz convocado para a vaga ou ao novo titular que vier a integrar o órgão prevento.

Art. 100. Aplica-se a regra do artigo anterior à hipótese de processo no qual haja recurso submetido à apreciação do Tribunal em razão de provimento de agravo de instrumento.

Art. 101. O agravo de instrumento que tramitar, ou que deveria tramitar, anexado ao processo principal, será distribuído no mesmo Colegiado e ao mesmo Relator.

Art. 102. A ação cautelar será distribuída ao Relator do processo principal, salvo se a medida for requerida em procedimento preparatório, hipótese em que será sorteado Relator dentre os integrantes do Colegiado competente para o julgamento da matéria, o qual fica prevento para a ação principal.

Parágrafo único. Observar-se-á a mesma regra na hipótese de recurso ordinário em ação cautelar.

Art. 103. À distribuição dos embargos infringentes não concorrerá o Ministro que já tenha atuado no processo como Relator e/ou redigido o acórdão embargado.

Art. 104. Os embargos interpostos contra decisão de Turma serão distribuídos entre os Ministros não integrantes do Colegiado prolator da decisão embargada.

Art. 105. Da distribuição da ação rescisória originária será excluído o Ministro que tenha relatado o processo e/ou redigido o acórdão rescindendo.

Art. 156. O incidente de uniformização reger-se-á pelos preceitos dos arts. 476 a 479 do Código de Processo Civil.

§ 6º Será Relator no Tribunal Pleno, o Ministro originariamente sorteado para relatar o feito em que se verifica o incidente de uniformização; se vencido, o Ministro que primeiro proferiu o voto prevalecente. Caso o Relator originário não componha o Tribunal Pleno, o feito será distribuído a um dos membros deste Colegiado.

Art. 158. A revisão ou cancelamento da jurisprudência uniformizada do Tribunal, objeto de Súmula, de Orientação Jurisprudencial e de Precedente Normativo, será suscitada pela Seção Especializada, ao constatar que a decisão se inclina contrariamente a Súmula, a Orientação Jurisprudencial ou a Precedente Normativo, ou por proposta firmada por pelo menos dez Ministros da Corte, ou por projeto formulado pela Comissão de Jurisprudência e Precedentes Normativos.

§ 3º Será relator no Tribunal Pleno o Ministro originariamente sorteado para relatar o feito em que se processa a revisão ou o cancelamento da Súmula, da Orientação Jurisprudencial ou do Precedente Normativo; se vencido, o Ministro que primeiro proferiu o voto prevalecente. Caso o relator originário não componha o Tribunal Pleno, o feito será distribuído a um dos membros deste Colegiado.

Art. 204. O processo de conflito será autuado e distribuído, observada a competência dos órgãos judicantes do Tribunal.

Art. 214. A ação rescisória terá início por petição, acompanhada de tantas cópias quantos forem os réus e preenchidos os requisitos da legislação processual compatíveis com o processo do trabalho.

Parágrafo único. Registrada e autuada, a ação rescisória será distribuída, mediante sorteio, a um Relator, dentre os Ministros integrantes da Subseção II Especializada em Dissídios Individuais, e designado Revisor o Ministro que a ele se seguir na ordem decrescente de antiguidade no órgão.

Art. 222. Requerida a homologação de acordo em processo de dissídio coletivo, antes ou depois do julgamento, da apresentação de recursos ou da publicação do acórdão, adotar-se-á o seguinte procedimento:

II - o processo será redistribuído a um dos membros do Colegiado, se ausente, por qualquer motivo, o Relator;

Art. 227. O agravo de instrumento interposto contra despacho denegatório do processamento de recurso de competência desta Corte será autuado e distribuído, observada a competência dos órgãos do Tribunal, aplicando-se quanto à tramitação e julgamento as disposições inscritas nesta Seção.

Art. 236. O agravo regimental será concluso ao prolator do despacho, que poderá reconsiderá-lo ou determinar sua inclusão em pauta visando apreciação do Colegiado competente para o julgamento da ação ou do recurso em que exarado o despacho, salvo o previsto no art. 235, inciso X, que será diretamente distribuído entre os demais integrantes da Subseção I da Seção Especializada em Dissídios Individuais.

Art. 242. Registrado o protocolo na petição e após sua juntada, os autos serão conclusos ao Relator da decisão embargada, ressalvadas as situações previstas nos arts. 92 a 96 deste Regimento.

Parágrafo único. Não sendo possível a aplicação de nenhuma das regras previstas nos arts. 92 a 96, adotar-se-á critério de competência para a distribuição dos embargos de declaração ao Juiz convocado, na hipótese dos processos das Turmas, ou ao Ministro que tenha ocupado a vaga do antigo Relator, nas Turmas e nas Subseções, e, como último critério, distribuir-se-á o processo entre os integrantes do órgão.

Art. 247. Os procedimentos relativos à remessa do processo ao Tribunal Pleno, à distribuição e ao julgamento da argüição de inconstitucionalidade são regulados pelas normas estabelecidas neste Regimento.

Art. 253. O pedido cautelar será apresentado ao Presidente do Tribunal e distribuído ao Relator do processo principal, salvo se a medida for requerida em procedimento preparatório, caso em que será sorteado, dentre os integrantes do Colegiado competente, o Relator do feito, o qual ficará prevento para a ação principal.

Art. 260. Os Ministros declarar-se-ão impedidos ou suspeitos nos casos previstos em lei.

Art. 261. A suspeição ou o impedimento do Relator ou Revisor serão declarados por despacho nos autos. Se feita na sessão de julgamento, a argüição será verbal, devendo constar da ata e da certidão.

Parágrafo único. Na suspeição ou no impedimento do Relator, o processo será redistribuído pelo Presidente do órgão julgador entre os demais Ministros que o compõem, observada oportuna compensação.

Art. 263. O Relator, reconhecendo a suspeição ou o impedimento, determinará a juntada da petição aos autos, e, por despacho, submeterá o processo à Presidência do Colegiado, para sua redistribuição, na forma regimental.

Art. 265. Reconhecida a suspeição ou o impedimento do Relator, declarar-se-ão nulos os atos praticados pelo Ministro suspeito ou impedido, e o processo será redistribuído, na forma regimental.

Art. 274. O pedido de restauração de autos será apresentado ao Presidente do Tribunal e distribuído ao Relator do processo desaparecido ou ao seu substituto.

Art. 301. Quando o agravo de instrumento for processado nos autos principais, nos quais se encontra sobrestado julgamento de recurso de revista da outra parte, na autuação do processo será considerado o número originário do recurso de revista sobrestado e observada a classe de agravo de instrumento em recurso de revista e recurso de revista (AIRR e RR).

Parágrafo único. O processo será distribuído ao Relator do recurso de revista sobrestado. Se o Relator não se encontrar em exercício no órgão prevento, haverá a redistribuição no âmbito do Colegiado a um dos seus integrantes.

Anexo II

Dicionário de Dados

Tabela II.1: Tabelas dos bancos de dados.

Tabela	Descrição
T_ADVOGADO	Advogados que atuam representando partes em Processos Judiciais.
T_CLASSE_PROCESSUAL	Classes Processuais classificam o processo judicial de acordo com o tipo de causa que se discute.
T_COMPETENCIA	Mapeamento entre as Classes Processuais e os Órgãos Judicantes para onde processos da classe podem ser distribuídos.
T_COMPOSICAO	Composição dos Órgãos Judicantes: relaciona os magistrados com os OJ, indicando a data de inicio e término da atuação do Magistrados naquele OJ.
T_DEN_PARTE_FASE_PROC	Denominação das partes do processo em uma determinada fase. Relaciona as tabelas T_FASE_PROCESSUAL e T_PROCESSO_PARTE de maneira a indicar qual a forma de atuação da parte na fase processual determinada.
T_DISTRIBUICAO	Distribuições realizadas pelo LawDisTrA
T_DISTRIBUIDOR	Responsáveis pela realização de distribuições de processo.
T_FASE_PROCESSUAL	Fases pelas quais passou um processo judicial. Um processo pode passar por várias fases. Para cada fase, pode ser atribuída uma classe processual diferente.
T_IMPEDIMENTO_ADVOGADO	Impedimento de Magistrados em relação aos Advogados que atuam em processos.
T_IMPEDIMENTO_PARTE	Impedimento de Magistrados em relação às Partes de processos.
T_IMPEDIMENTO_PROCESSO	Impedimento de Magistrados em relação aos Processos específicos.

T_INFO_DISTRIBUICAO	Informações adicionais com detalhes da distribuição informados pelos agentes LawDisTrA.
T_MAGISTRADO	Magistrados do Tribunal.
T_ORGAO_JUDICANTE	Entidade diretamente responsável pelo julgamento de processos judiciais.
T_PARTE	Partes interessadas do Processo Judicial.
T_PROCESSO	Processos Judiciais Autuados. Os processos possuem uma identificação interna COD_PROCESSO utilizada no banco de dados e uma identificação padrão (numeração única) utilizada em todos os ramos do Poder Judiciário. A identificação padrão é utilizada desde janeiro de 2010 no TST e possui o formato numérico de 20 dígitos no formato NNNNNNN-DD.AAAA.J.TR.OOOO.
T_PROCESSO_PARTE	Relaciona as Partes com seus respectivos Processos Judiciais de interesse.
T_PROCESSO_PARTE_ADVOGADO	Relaciona os Advogados com as Partes dos Processos Judiciais.
T_PROCESSO_PARTE_PROCURADOR	Relaciona os Procuradores das Partes dos Processos Judiciais.
T_PROCESSO_RELACIONADO	Identifica conexão entre Processos de forma a permitir a Distribuição de processos relacionados.
T_PROCURADOR	Procuradores que atuam em Processos Judiciais.
T_PROTOCOLO	Informações acerca dos protocolos responsáveis por autuar processos.
T_TIPO_DIST	Relaciona os possíveis tipos de distribuição.
T_TIPO_RELACIONAMENTO	Relaciona os possíveis tipos de relacionamento entre dois processos.

Tabela II.2: Colunas das tabelas dos bancos de dados.

Coluna	**Tipo de dados**	**Descrição**
ANO_PROCESSO	NUMERIC(4)	AAAA - ano do ajuizamento do processo
COD_AGENTE	VARCHAR(8)	Código do agente LawDisTrA responsável pela informação
COD_DISTRIBUIDOR	CHAR(8)	Código que identifica o distribuidor do processo
COD_MAGISTRADO	VARCHAR(4)	Código Identificador do Magistrado
COD_PARTE	NUMERIC(10)	Código identificador da parte
COD_PROCESSO	NUMERIC(11)	Código numérico atribuído ao processo para identificá-lo no banco de dados
COD_PROCESSO_REL	NUMERIC(11)	Código numérico atribuído ao processo relacionado
COD_PROTOCOLO	VARCHAR(8)	Código que identifica o protocolo
COD_TIPO_DIST	CHAR(1)	Código do tipo de distribuição

COD_TIPO_RELAC	NUMERIC(2)	Código do tipo de relacionamento
DES_OJ	VARCHAR(64)	Descrição do Órgão Judicante
DES_TIPO_DIST	VARCHAR(64)	Descrição do tipo de distribuição
DES_TIPO_RELAC	VARCHAR(16)	Descrição do tipo de relacionamento
DTA_AUTUACAO	DATE	Data de autuação do processo
DTA_DISTRIBUICAO	DATE	Data de realização da distribuição do processo
DTA_INFORMACAO	TIMESTAMP	Data e hora em que a informação foi registrada
DTA_INI_ATUACAO	DATE	Data de início da atuação do Magistrado no OJ
DTA_INICIO_FASE	DATE	Data de início da fase
DTA_REGISTRO	DATE	Data de registro do impedimento
DTA_TER_ATUACAO	DATE	Data de termino da atuação do Magistrado no OJ
DTA_TERMINO_FASE	DATE	Data de término da fase do processo
IND_ADMINISTRACAO	BOOL	Indica se o magistrado compõe a Administração do Tribunal
IND_FASE_INICIAL	BOOL	Indica se a classe é atribuída na fase inicial do processo, ou diz respeito a fases de recursos
IND_POLO_ATIVO	BOOL	Indica se a parte atua como polo ativo nesta fase do processo
IND_PRESIDENTE	BOOL	Indica se o magistrado é o presidente do respectivo OJ
NOM_ADVOGADO	VARCHAR(64)	Nome Completo do Advogado
NOM_CLASSE	VARCHAR(64)	Nome completo da classe
NOM_DISTRIBUIDOR	VARCHAR(64)	Nome do distribuidor
NOM_MAGISTRADO	VARCHAR(64)	Nome Completo do Magistrado
NOM_PARTE	VARCHAR(512)	Nome completo da parte
NOM_POLO_ATIVO	VARCHAR(16)	Designação dada ao polo ativo da causa (o autor)
NOM_POLO_PASSIVO	VARCHAR(16)	Designação dada ao polo passivo da causa (o réu)
NOM_PROCURADOR	VARCHAR(64)	Nome completo do procurador
NOM_PROCURADORIA	VARCHAR(64)	Nome da procuradoria pela qual o procurador atua
NOM_PROTOCOLO	VARCHAR(64)	Nome dado ao protocolo
NUM_ADVOGADO	NUMERIC(8)	Número de identificação do Advogado
NUM_CLASSE_CNJ	NUMERIC(5)	Numeração da classe segundo classificação do CNJ
NUM_CNPJ	NUMERIC(14)	Número do CNPJ se pessoa jurídica
NUM_CPF	NUMERIC(11)	Número do CPF se pessoa física
NUM_DIGITO	NUMERIC(2)	dígitos verificadores da integridade do número do processo.
NUM_OAB	NUMERIC(8)	Número de inscrição do advogado na OAB
NUM_ORIGEM	NUMERIC(4)	OOOO - unidade de origem do processo, seguindo regras diversas para cada um dos segmentos do Judiciário, à exceção dos tribunais e conselhos, que terão esses dígitos preenchidos com zero (0000)

NUM_PROCESSO	NUMERIC(7)	NNNNNNN - número de ordem de autuação do processo, no ano de autuação e na unidade jurisdicional de origem
NUM_PROCURADOR	NUMERIC(8)	Número identificador do procurador
NUM_SEGMENTO	NUMERIC(1)	J - dígito identificador do segmento do Judiciário a que pertence o processo. 1 - STF, 2 - CNJ, 3 - STJ, 4 - JF, 5 - JT, 6 - JE, 7 - JM da União, 8 - Justiça dos Estados e do DF, 9 JM Estadual
NUM_TRIBUNAL	NUMERIC(2)	TR - tribunal ou conselho do segmento do Poder Judiciário a que pertence o processo; para os tribunais superiores (STF, STJ, TST, TSE e STM) e o CNJ, o código deverá ser preenchido com zero (00).
SEQ_DISTRIBUICAO	NUMERIC(8)	Número sequencial da distribuição de acordo com o Distribuidor
SEQ_INFORMACAO	INTEGER	Número sequencial que identifica a informação
SIG_CLASSE	VARCHAR(8)	Sigla identificadora da classe
SIG_OJ	VARCHAR(8)	Sigla identificadora do Órgão Judicante
SIG_UF_OAB	CHAR(2)	Sigla da UF da OAB onde o Advogado está registrado
TIP_INFORMACAO	VARCHAR(16)	Tipo de informação registrada (mensagem, tarefa, etc.)
TIP_PARTE	CHAR(1)	Tipo da Parte: F - pessoa Física, J - pessoa Jurídica
TXT_DETALHES	VARCHAR(256)	Texto com detalhes adicionais
TXT_DISTRIBUICAO	VARCHAR(2048)	Registra detalhes da distribuição para fins de posterior análise
TXT_INFORMACAO	VARCHAR(256)	Texto da informação
TXT_REGRA_APLICADA	VARCHAR(256)	Identifica a regra aplicada na distribuição

Anexo III

Scripts de Criação das Estruturas dos Bancos de Dados

III.1 Banco de Dados de Processos Judiciais

```
CREATE TABLE t_advogado (
    num_advogado numeric(8,0) NOT NULL,
    nom_advogado character varying(64) NOT NULL,
    num_oab numeric(8,0),
    sig_uf_oab character(2)
);
CREATE TABLE t_classe_processual (
    sig_classe character varying(8) NOT NULL,
    nom_classe character varying(64) NOT NULL,
    num_classe_cnj numeric(5,0) NOT NULL,
    nom_polo_ativo character varying(16) NOT NULL,
    nom_polo_passivo character varying(16) NOT NULL,
    ind_fase_inicial boolean NOT NULL
);
CREATE TABLE t_den_parte_fase_proc (
    cod_processo numeric(11,0) NOT NULL,
    cod_parte numeric(10,0) NOT NULL,
    dta_inicio_fase date NOT NULL,
    sig_classe character varying(8) NOT NULL,
    ind_polo_ativo boolean NOT NULL
);
CREATE TABLE t_fase_processual (
    cod_processo numeric(11,0) NOT NULL,
    dta_inicio_fase date NOT NULL,
    sig_classe character varying(8) NOT NULL,
    dta_termino_fase date,
```

```
    cod_magistrado character varying(4),
    cod_motivo_redist numeric(2,0),
    sig_oj character varying(8)
);
CREATE TABLE t_parte (
    cod_parte numeric(10,0) NOT NULL,
    nom_parte character varying(512) NOT NULL,
    tip_parte character(1) NOT NULL,
    num_cnpj numeric(14,0),
    num_cpf numeric(11,0)
);
CREATE TABLE t_processo (
    cod_processo numeric(11,0) NOT NULL,
    num_processo numeric(7,0) NOT NULL,
    num_digito numeric(2,0) NOT NULL,
    ano_processo numeric(4,0) NOT NULL,
    num_segmento numeric(1,0) NOT NULL,
    num_tribunal numeric(2,0) NOT NULL,
    num_origem numeric(4,0) NOT NULL,
    dta_autuacao date NOT NULL
);
CREATE TABLE t_processo_parte (
    cod_processo numeric(11,0) NOT NULL,
    cod_parte numeric(10,0) NOT NULL
);
CREATE TABLE t_processo_parte_advogado (
    cod_processo numeric(11,0) NOT NULL,
    cod_parte numeric(10,0) NOT NULL,
    num_advogado numeric(8,0) NOT NULL
);
CREATE TABLE t_processo_parte_procurador (
    cod_processo numeric(11,0) NOT NULL,
    cod_parte numeric(10,0) NOT NULL,
    num_procurador numeric(8,0) NOT NULL
);
CREATE TABLE t_processo_relacionado (
    cod_processo numeric(11,0) NOT NULL,
    cod_processo_rel numeric(11,0) NOT NULL,
    cod_tipo_relac numeric(2,0) NOT NULL
);
CREATE TABLE t_procurador (
    num_procurador numeric(8,0) NOT NULL,
    nom_procurador character varying(64) NOT NULL,
    nom_procuradoria character varying(64) NOT NULL
);
```

```
CREATE TABLE t_tipo_relacionamento (
    cod_tipo_relac numeric(2,0) NOT NULL,
    des_tipo_relac character varying(16) NOT NULL
);
ALTER TABLE ONLY t_advogado
    ADD CONSTRAINT advogado_pk PRIMARY KEY (num_advogado);
ALTER TABLE ONLY t_classe_processual
    ADD CONSTRAINT classe_processual_pk PRIMARY KEY (sig_classe);
ALTER TABLE ONLY t_parte
    ADD CONSTRAINT parte_pk PRIMARY KEY (cod_parte);
ALTER TABLE ONLY t_den_parte_fase_proc
    ADD CONSTRAINT pk_t_den_parte_fase_proc PRIMARY KEY (cod_processo,
        cod_parte, dta_inicio_fase, sig_classe);
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT pk_t_fase_processual PRIMARY KEY (cod_processo,
        dta_inicio_fase, sig_classe);
ALTER TABLE ONLY t_processo_parte_advogado
    ADD CONSTRAINT processo_parte_advogado_pk PRIMARY KEY (cod_processo,
        cod_parte, num_advogado);
ALTER TABLE ONLY t_processo_parte
    ADD CONSTRAINT processo_parte_pk PRIMARY KEY (cod_processo, cod_parte);
ALTER TABLE ONLY t_processo_parte_procurador
    ADD CONSTRAINT processo_parte_procurador_pk PRIMARY KEY (cod_processo,
        cod_parte, num_procurador);
ALTER TABLE ONLY t_processo
    ADD CONSTRAINT processo_pk PRIMARY KEY (cod_processo);
ALTER TABLE ONLY t_processo_relacionado
    ADD CONSTRAINT processo_relacionado_pk PRIMARY KEY (cod_processo,
        cod_processo_rel);
ALTER TABLE ONLY t_procurador
    ADD CONSTRAINT procurador_pk PRIMARY KEY (num_procurador);
ALTER TABLE ONLY t_tipo_relacionamento
    ADD CONSTRAINT tipo_relacionamento_pk PRIMARY KEY (cod_tipo_relac);
CREATE INDEX "FASE_PROCESSUAL_IN1" ON t_fase_processual USING btree (
    dta_termino_fase DESC);
CREATE UNIQUE INDEX classe_processual_in1 ON t_classe_processual USING
    btree (nom_classe);
ALTER TABLE ONLY t_den_parte_fase_proc
    ADD CONSTRAINT den_parte_fase_proc_fk1 FOREIGN KEY (cod_processo,
        cod_parte) REFERENCES t_processo_parte(cod_processo, cod_parte) ON
        UPDATE RESTRICT ON DELETE RESTRICT;
ALTER TABLE ONLY t_den_parte_fase_proc
    ADD CONSTRAINT den_parte_fase_proc_fk2 FOREIGN KEY (cod_processo,
        dta_inicio_fase, sig_classe) REFERENCES t_fase_processual(
```

```
        cod_processo, dta_inicio_fase, sig_classe) ON UPDATE CASCADE ON
        DELETE CASCADE;
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT fase_processual_fk1 FOREIGN KEY (sig_classe) REFERENCES
        t_classe_processual(sig_classe) ON UPDATE RESTRICT ON DELETE
        RESTRICT;
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT fase_processual_fk2 FOREIGN KEY (sig_oj) REFERENCES
        t_orgao_judicante(sig_oj) ON UPDATE SET NULL ON DELETE SET NULL;
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT fase_processual_fk3 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE SET NULL ON DELETE
         SET NULL;
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT fase_processual_fk4 FOREIGN KEY (cod_motivo_redist)
        REFERENCES t_motivo_redistribuicao(cod_motivo_redist) ON UPDATE
        RESTRICT ON DELETE RESTRICT;
ALTER TABLE ONLY t_fase_processual
    ADD CONSTRAINT fase_processual_fk5 FOREIGN KEY (cod_processo)
        REFERENCES t_processo(cod_processo) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_processo_parte_advogado
    ADD CONSTRAINT processo_parte_advogado_fk1 FOREIGN KEY (cod_processo,
        cod_parte) REFERENCES t_processo_parte(cod_processo, cod_parte) ON
        UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_processo_parte_advogado
    ADD CONSTRAINT processo_parte_advogado_fk2 FOREIGN KEY (num_advogado)
        REFERENCES t_advogado(num_advogado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_processo_parte
    ADD CONSTRAINT processo_parte_fk1 FOREIGN KEY (cod_processo) REFERENCES
         t_processo(cod_processo) ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_processo_parte
    ADD CONSTRAINT processo_parte_fk2 FOREIGN KEY (cod_parte) REFERENCES
        t_parte(cod_parte) ON UPDATE RESTRICT ON DELETE RESTRICT;
ALTER TABLE ONLY t_processo_parte_procurador
    ADD CONSTRAINT processo_parte_procurador_fk1 FOREIGN KEY (cod_processo,
         cod_parte) REFERENCES t_processo_parte(cod_processo, cod_parte) ON
        UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_processo_parte_procurador
    ADD CONSTRAINT processo_parte_procurador_fk2 FOREIGN KEY (
        num_procurador) REFERENCES t_procurador(num_procurador) ON UPDATE
        CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_processo_relacionado
```

```
    ADD CONSTRAINT processo_relacionamento_fk1 FOREIGN KEY (cod_processo)
        REFERENCES t_processo(cod_processo) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_processo_relacionado
    ADD CONSTRAINT processo_relacionamento_fk2 FOREIGN KEY (
        cod_processo_rel) REFERENCES t_processo(cod_processo) ON UPDATE
        CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_processo_relacionado
    ADD CONSTRAINT processo_relacionamento_fk3 FOREIGN KEY (cod_tipo_relac)
        REFERENCES t_tipo_relacionamento(cod_tipo_relac) ON UPDATE RESTRICT
        ON DELETE RESTRICT;
```

III.2 Banco de Dados de Magistrados

```
CREATE TABLE t_composicao_oj (
    cod_magistrado character varying(4) NOT NULL,
    dta_ini_atuacao date NOT NULL,
    dta_ter_atuacao date,
    ind_presidente boolean NOT NULL,
    sig_oj character varying(8) NOT NULL
);
CREATE TABLE t_impedimento_advogado (
    cod_magistrado character varying(4) NOT NULL,
    num_advogado numeric(8,0) NOT NULL,
    dta_registro date DEFAULT ('now'::text)::date NOT NULL
);
CREATE TABLE t_impedimento_parte (
    cod_magistrado character varying(4) NOT NULL,
    cod_parte numeric(10,0) NOT NULL,
    dta_registro date DEFAULT ('now'::text)::date NOT NULL
);
CREATE TABLE t_impedimento_processo (
    cod_magistrado character varying(4) NOT NULL,
    cod_processo numeric(11,0) NOT NULL,
    dta_registro date DEFAULT ('now'::text)::date NOT NULL
);
CREATE TABLE t_orgao_judicante (
    sig_oj character varying(8) NOT NULL,
    des_oj character varying(64) NOT NULL
);
ALTER TABLE ONLY t_composicao_oj
    ADD CONSTRAINT composicao_pk PRIMARY KEY (sig_oj, cod_magistrado);
ALTER TABLE ONLY t_magistrado
    ADD CONSTRAINT magistrado_pk PRIMARY KEY (cod_magistrado);
ALTER TABLE ONLY t_impedimento_advogado
```

```
    ADD CONSTRAINT impedimento_advogado_pk PRIMARY KEY (cod_magistrado,
        num_advogado);
ALTER TABLE ONLY t_impedimento_parte
    ADD CONSTRAINT impedimento_parte_pk PRIMARY KEY (cod_magistrado,
        cod_parte);
ALTER TABLE ONLY t_impedimento_processo
    ADD CONSTRAINT impedimento_processo_pk PRIMARY KEY (cod_magistrado,
        cod_processo);
ALTER TABLE ONLY t_orgao_judicante
    ADD CONSTRAINT orgao_judicante_pk PRIMARY KEY (sig_oj);
ALTER TABLE ONLY t_composicao_oj
    ADD CONSTRAINT composicao_oj_fk1 FOREIGN KEY (sig_oj) REFERENCES
        t_orgao_judicante(sig_oj) ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_composicao_oj
    ADD CONSTRAINT composicao_oj_fk2 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_impedimento_advogado
    ADD CONSTRAINT impedimento_advogado_fk1 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_impedimento_advogado
    ADD CONSTRAINT impedimento_advogado_fk2 FOREIGN KEY (num_advogado)
        REFERENCES t_advogado(num_advogado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_impedimento_parte
    ADD CONSTRAINT impedimento_parte_fk1 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_impedimento_parte
    ADD CONSTRAINT impedimento_parte_fk2 FOREIGN KEY (cod_parte) REFERENCES
         t_parte(cod_parte) ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_impedimento_processo
    ADD CONSTRAINT impedimento_processo_fk1 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_impedimento_processo
    ADD CONSTRAINT impedimento_processo_fk2 FOREIGN KEY (cod_processo)
        REFERENCES t_processo(cod_processo) ON UPDATE CASCADE ON DELETE
        CASCADE;
```

III.3 Banco de Dados da Distribuição

```
CREATE TABLE t_competencia (
    sig_classe character varying(8) NOT NULL,
```

```
    sig_oj character varying(8) NOT NULL
);
CREATE TABLE t_distribuicao (
    cod_distribuidor character(8) NOT NULL,
    seq_distribuicao numeric(8,0) NOT NULL,
    cod_processo numeric(11,0) NOT NULL,
    cod_tipo_dist character(1) NOT NULL,
    dta_distribuicao timestamp without time zone NOT NULL,
    cod_magistrado character varying(4),
    txt_distribuicao character varying(2048) NOT NULL,
    sig_oj character varying(8),
    txt_regra_aplicada character varying(256) NOT NULL
);
CREATE TABLE t_distribuidor (
    cod_distribuidor character varying(8) NOT NULL,
    nom_distribuidor character varying(64) NOT NULL
);
CREATE TABLE t_info_distribuicao (
    seq_informacao integer NOT NULL,
    dta_informacao timestamp without time zone
        DEFAULT ('now'::text)::timestamp without time zone NOT NULL,
    cod_agente character varying(8) NOT NULL,
    tip_informacao character varying(16) NOT NULL,
    seq_distribuicao numeric(8,0),
    txt_informacao character varying(256) NOT NULL,
    txt_detalhes character varying(256)
);
CREATE SEQUENCE t_info_distribuicao_seq_informacao_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
ALTER SEQUENCE t_info_distribuicao_seq_informacao_seq OWNED BY
    t_info_distribuicao.seq_informacao;
CREATE TABLE t_magistrado (
    cod_magistrado character varying(4) NOT NULL,
    nom_magistrado character varying(64) NOT NULL,
    ind_administracao boolean DEFAULT false NOT NULL
);
CREATE TABLE t_protocolo (
    cod_protocolo character varying(8) NOT NULL,
    nom_protocolo character varying(64) NOT NULL,
    num_tribunal numeric(2,0) NOT NULL
);
```

```
CREATE TABLE t_tipo_dist (
    cod_tipo_dist character(1) NOT NULL,
    des_tipo_dist character varying(64) NOT NULL
);
ALTER TABLE ONLY t_info_distribuicaoALTER COLUMN seq_informacao
        SET DEFAULT nextval('t_info_distribuicao_seq_informacao_seq'::
            regclass);
ALTER TABLE ONLY t_competencia
    ADD CONSTRAINT competencia_pk PRIMARY KEY (sig_classe, sig_oj);
ALTER TABLE ONLY t_distribuidor
    ADD CONSTRAINT distribuidor_pk PRIMARY KEY (cod_distribuidor);
ALTER TABLE ONLY t_distribuicao
    ADD CONSTRAINT hist_distribuicao_pk PRIMARY KEY (cod_distribuidor,
        seq_distribuicao);
ALTER TABLE ONLY t_magistrado
    ADD CONSTRAINT magistrado_pk PRIMARY KEY (cod_magistrado);
ALTER TABLE ONLY t_protocolo
    ADD CONSTRAINT pk_t_protocolo PRIMARY KEY (cod_protocolo);
ALTER TABLE ONLY t_info_distribuicao
    ADD CONSTRAINT t_info_distribuicao_pkey PRIMARY KEY (seq_informacao);
ALTER TABLE ONLY t_tipo_dist
    ADD CONSTRAINT tipo_dist_pk PRIMARY KEY (cod_tipo_dist);
CREATE INDEX info_distribuicao_in1 ON t_info_distribuicao USING btree (
    seq_distribuicao);
ALTER TABLE ONLY t_competencia
    ADD CONSTRAINT competencia_fk1 FOREIGN KEY (sig_classe) REFERENCES
        t_classe_processual(sig_classe);
ALTER TABLE ONLY t_competencia
    ADD CONSTRAINT competencia_fk2 FOREIGN KEY (sig_oj) REFERENCES
        t_orgao_judicante(sig_oj) ON UPDATE CASCADE ON DELETE CASCADE;
ALTER TABLE ONLY t_distribuicao
    ADD CONSTRAINT hist_distribuicao_fk1 FOREIGN KEY (cod_distribuidor)
        REFERENCES t_distribuidor(cod_distribuidor) ON UPDATE RESTRICT ON
        DELETE RESTRICT;
ALTER TABLE ONLY t_distribuicao
    ADD CONSTRAINT hist_distribuicao_fk2 FOREIGN KEY (cod_tipo_dist)
        REFERENCES t_tipo_dist(cod_tipo_dist) ON UPDATE RESTRICT ON DELETE
        RESTRICT;
ALTER TABLE ONLY t_distribuicao
    ADD CONSTRAINT hist_distribuicao_fk3 FOREIGN KEY (cod_processo)
        REFERENCES t_processo(cod_processo) ON UPDATE CASCADE ON DELETE
        CASCADE;
ALTER TABLE ONLY t_distribuicao
```

```
    ADD CONSTRAINT hist_distribuicao_fk4 FOREIGN KEY (cod_magistrado)
        REFERENCES t_magistrado(cod_magistrado) ON UPDATE RESTRICT ON DELETE
         RESTRICT;
ALTER TABLE ONLY t_distribuicao
    ADD CONSTRAINT hist_distribuicao_fk5 FOREIGN KEY (sig_oj) REFERENCES
        t_orgao_judicante(sig_oj) ON UPDATE RESTRICT ON DELETE RESTRICT;
```