UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ELETRICIDADE
ÁREA DE CIÊNCIA DA COMPUTAÇÃO

WILLIAM CORRÊA MENDES

ARQUITETURA BASEADA EM ONTOLOGIAS DE UM AGENTE RBC

São Luís
2013

WILLIAM CORRÊA MENDES

ARQUITETURA BASEADA EM ONTOLOGIAS DE UM AGENTE RBC

Dissertação de Mestrado apresentada ao curso de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão, como parte dos requisitos para a obtenção do título de Mestre em Engenharia de Eletricidade, na área de Ciência da Computação.

Orientadora: Profa. Dra. Rosario Girardi

São Luís

2013

Mendes, William Corrêa

Arquitetura baseada em ontologia de um agente RBC / William Corrêa Mendes. – São Luís, 2013.

127 f.

Orientador: Rosario Girardi.

Dissertação (Mestrado) – Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão, 2013.

1. Arquitetura de sistema - Ontologias. 2. Agente de software. I. Título.

CDU 004.4’2

ARQUITETURA BASEADA EM ONTOLOGIAS DE UM AGENTE RBC

William Corrêa Mendes

Dissertação aprovada em 04 de novembro de 2013.

Profa. María del Rosario Girardi Gutiérrez, Ph.D.
(Orientadora)

Prof. Evandro de Barros Costa, Dr.
(Membro da Banca Examinadora)

Prof. Francisco José da Silva e Silva, Dr.
(Membro da Banca Examinadora)

A meu primo e irmão, Fabrício Mendes (in memoriam).

AGRADECIMENTOS

A Deus, pela graça de ter me permitido concluir este trabalho.

À minha esposa Rafaela, que me deu incentivo e apoio durante esse trajeto, mesmo que tenha significado o adiamento de momentos especiais em nossa vida.

À minha família, pelo apoio. Em especial, aos meus pais Reginaldo e Janice que sempre me apoiaram e incentivaram na conclusão desta nova etapa. Aos meus irmãos Ronney, Hellen, Farah e Fred, pelo apoio, incentivo e torcida para a conclusão deste trabalho. Em especial, à minha irmã Farah que me ajudou revisando alguns dos meus textos deste trabalho.

À professora Dra. Rosario Girardi, cujo apoio e orientações foram fundamentais para a conclusão deste trabalho, pela paciência de me orientar, cobrando o máximo de mim e me guiando por todo o processo.

A Adriana Leite, cujas ponderações, sugestões e questionamentos muito ajudaram a enriquecer este trabalho, contribuindo diretamente para a realização do mesmo.

Aos colegas do mestrado, em especial, a Suzane e Fhillipi, pela ajuda mútua durante a realização das disciplinas. Ao Luís Eduardo, pela ajuda nas primeiras fases da pesquisa. Ao Paulo, pelas sugestões e ajuda na revisão do manuscrito.

A todos os integrantes e ex-integrantes do grupo GESEC que contribuíram direta ou indiretamente para realização desse trabalho.

Ao professor Dr. Sofiane Labidi, que me proporcionou a entrada no programa permitindo-me assistir às aulas de disciplinas isoladas, e pelo seu apoio e incentivo.

Ao programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão, pela oportunidade de realização do curso.

Aos amigos e colegas da Diretoria de Gestão da Tecnologia da Informação do IFMA, que me incentivaram e me apoiaram para a conclusão deste trabalho. Em especial ao diretor, prof. Claudio Fernandes, e ao coordenador Leonardo Rosa, por terem-me concedido horário especial necessário à realização de atividades do mestrado.

RESUMO

O Raciocínio Baseado em Casos (RBC) é um paradigma de resolução de problemas no qual é possível utilizar conhecimentos de experiências passadas para resolver novas situações. A abordagem de agentes RBC que combina a autonomia dos agentes e o modelo de resolução de problemas do RBC tem se mostrado adequada para o desenvolvimento de sistemas complexos. Este trabalho propõe a arquitetura de um agente RBC cujo principal diferencial é utilizar ontologias para representar a base de casos junto com todos os mecanismos que compõem um sistema RBC. A arquitetura proposta, além de promover o reúso da ontologia de representação dos casos, unifica as abordagens de agentes de software e RBC, um paradigma de raciocínio típico dos seres humanos. Estão presentes na arquitetura os mecanismos de representação dos casos, análise de similaridade para recuperação de casos, adaptação e aprendizado de casos, estes dois últimos ainda em fase de especificação. A arquitetura foi avaliada no domínio jurídico do Direito de Família brasileiro, sendo que para isso foi criada uma ontologia, representando casos RBC nesta área. Os resultados obtidos nos testes realizados demonstraram uma boa efetividade na recuperação de casos similares e a consequente viabilidade do uso da arquitetura com o modelo de similaridade semântico utilizado para recuperação de casos RBC.

Palavras-chave: Agentes de Software, Raciocínio, Ontologias, Direito de Família.

ABSTRACT

Case-Based Reasoning (CBR) is a problem-solving paradigm where it is possible to use knowledge from past experiences to solve new situations. The CBR agent approach that combines agent autonomy with the problem-solving model of CBR has been proven adequate for the development of complex systems. This paper proposes the architecture of a CBR agent whose main differential is the use of ontologies for representing the case base along with all the mechanisms that make up a CBR system. The proposed architecture besides promoting the reuse of the case ontology, unifies the software agent approach with CBR, a typical paradigm of human reasoning. All the CBR mechanisms are present in the proposed architecture: case representation, similarity analysis for cases retrieval, adaptation and cases learning, where the last two mechanisms are still being specified. The architecture was evaluated in the Brazilian Family Law legal domain. For that, a targeted ontology for the representation CBR cases of this area was created. The results obtained in the tests showed good effectiveness in retrieving similar cases and showing the feasibility of the architecture using the semantic model of similarity for retrieval of CBR cases.

Keywords: Software Agents, Reasoning, Ontologies, Family Law.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

BDI	Belief, Desire and Intentions
CBOR	Case-Based Ontology Reasoning
CBR	Case-Based Reasoning
CBRAA	Case-based Reasoning Agent Architecture
CPR	Caso -Problema Recuperado
EG	Episódio Generalizado
EI	Entidade de Informação
GESEC	Grupo de Pesquisa em Engenharia de Software e Engenharia do Conhecimento
IA	Inteligência Artificial
IBL	Instance-Based Learning
NCP	Novo Caso-Problema
RBC	Raciocínio Baseado em Casos
RBF	Radial Basis Function
RNA	Redes Neurais Artificiais
RRC	Rede de Recuperação de Casos

SUMÁRIO

1. INTRODUÇÃO

Os sistemas de conhecimento transitaram dos sistemas especialistas, suportando essencialmente a capacidade de raciocínio lógico para apoio às decisões de especialistas em domínios específicos de conhecimento, até os atuais sistemas multiagentes. Além das habilidades dos tradicionais sistemas especialistas, são compostos de entidades com autonomia, sociabilidade e habilidade de aprendizado, permitindo, assim, abordar de forma mais apropriada, a crescente complexidade dos sistemas de software (GIRARDI e LEITE, 2011).

Para atender a essa crescente complexidade, busca-se emular o comportamento inteligente do ser humano através de modelos de arquiteturas reativas que suportam comportamento reativo ou instintivo, e arquiteturas deliberativas que suportam diferentes formas de raciocínio, entre elas, o Raciocínio Baseado em Casos (RBC), um tipo de raciocínio utilizado para resolver problemas pelos seres humanos (AAMODT e PLAZA, 1994).

A combinação da autonomia de um agente permitindo a resolução de problemas pelo software a partir de percepções obtidas do ambiente através de sensores e o paradigma de raciocínio RBC, contribui, assim, para a construção de sistemas complexos mais efetivos na resolução de problemas.

1.1 Relevância e Motivação

O RBC é uma das tecnologias mais populares e disseminadas para o desenvolvimento de sistemas baseados em conhecimento (WANGENHEIM e WANGENHEIM, 2003).

É um paradigma de resolução de problemas através do qual é possível utilizar conhecimento de experiências passadas e reutilizá-lo para resolver situações novas. Essa forma de raciocínio é poderosa e frequente na resolução de problemas pelos seres humanos (AAMODT e PLAZA, 1994). Tal paradigma tem como preocupação solucionar problemas através da identificação e adaptação de situações similares armazenadas em uma base de experiências e soluções passadas (NAGAIAH, 2011).

Os quatro mecanismos que devem compor um sistema RBC são os mecanismos para representação de casos, recuperação e análise de similaridade, adaptação e aprendizado (WANGENHEIM e WANGENHEIM, 2003).

A abordagem de agentes racionais capazes de realização de inferências corretas e de interações com outros agentes de softwares é uma solução viável para o desenvolvimento de sistemas complexos. A autonomia que os caracteriza permite que as decisões sejam tomadas em tempo de execução e simplifica o desenvolvimento desse tipo de aplicação (GIRARDI, 2001).

Através da construção de ontologias para a representação dos casos é possível representar conhecimento semântico associado aos casos, que, além de viabilizar a recuperações de casos com uma melhor precisão, possibilita a reutilização da ontologia em outros sistemas baseados em conhecimento. Além disso, o uso de ontologias para representação de casos, em conjunto com uma medida de similaridade baseada em semântica para recuperação de casos, ajuda na construção de um método inteligente para a identificação e recuperação de casos similares, necessários para a resolução de problemas através do RBC, visto que, ao utilizar semântica, é possível obter valores de similaridade mais precisos na comparação de casos, proporcionando uma melhor efetividade na recuperação das soluções mais apropriadas para reutilização.

Por isso, um agente RBC capaz de executar ou propor soluções à novos problemas obtidos através dos sensores diretamente do ambiente, com base em experiências anteriores recuperadas através de um modelo de recuperação baseado no conhecimento, é uma boa solução para automação da resolução de problemas e sistemas de apoio à decisão de grande complexidade.

O desenvolvimento de uma arquitetura de um agente deliberativo que utiliza o RBC é uma forma de colaborar com as pesquisas que vêm sendo realizadas desde 2011 no grupo de pesquisa em Engenharia de Software (GESEC), pesquisas estas relacionadas ao desenvolvimento de sistemas multiagentes.

1.2 Objetivos

1.2.1 Objetivos Gerais

Contribuir para o desenvolvimento de sistemas complexos através da abordagem orientada a agentes e do paradigma RBC.

1.2.2 Objetivos Específicos

- Propor uma arquitetura baseada em ontologias de um agente RBC que englobe os quatro principais mecanismos de um sistema RBC.
- Especificar um formalismo baseado em ontologias para representação de casos RBC.
- Especificar um modelo semântico para a recuperação e análise de similaridade dos casos.
- Avaliar a efetividade do mecanismo de recuperação e análise de similaridade da arquitetura do agente RBC com um estudo de caso no domínio jurídico do Direito de Família brasileiro.

1.3 Estrutura da Dissertação

Este trabalho é composto de cinco capítulos, incluindo esta introdução. O capítulo 2 mostra o referencial teórico utilizado no trabalho, discutindo sobre os tipos de raciocínio, em particular o RBC, sobre as ontologias e a tecnologia de agentes de software. Este capítulo apresenta ainda um estudo do estado da arte de abordagens RBC e comparação com a arquitetura proposta neste trabalho. No capítulo 3, a arquitetura CBRAA é apresentada, destacando as características dos mecanismos de representação de casos, e o de recuperação e análise de similaridade. O capítulo 4 apresenta uma avaliação da arquitetura através de um estudo de caso que aborda casos do Direito de Família brasileiro, e o capítulo 5 expõe as conclusões sobre o trabalho realizado, enfatizando os resultados e os trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Raciocínio

O raciocínio é o processo de realizar inferências sobre determinada informação a fim de elaborar conclusões. Existem três espécies principais de raciocínio: dedução, indução e abdução. Para entender suas diferenças é necessário distinguir inicialmente a dedução da indução (BRANQUINHO, MURCHO e GOMES, 2006).

A dedução e a indução diferem na forma como suas conclusões são suportadas pelas premissas do raciocínio (COPI e COHEN, 2001). Na dedução, o processo de raciocínio acontece do geral para o particular (MANKTELOW, 1999), ou seja, as conclusões são suportadas pelas premissas (COPI e COHEN, 2001). Diferentemente, na indução, parte-se do particular para o geral, isto é, a partir de um conjunto de observações aplica-se um processo de busca de conclusões (MANKTELOW, 1999).

Por exemplo, no processo de raciocínio dedutivo, a partir da regra: “Todos os homens são mortais” e da observação “João é homem” é possível deduzir que “João é mortal”. No raciocínio indutivo, considerando um conjunto de premissas do tipo “Carlos é homem e é mortal” e “Luís é homem e é mortal” é possível generalizar uma regra baseada na experiência obtida que afirma que “Todos os homens são mortais”.

A abdução por sua vez, elabora hipóteses explicativas que são então avaliadas (THAGARD e SHELLEY, 1997). Por exemplo, a partir da observação “A rua está molhada” é possível, considerando outras observações prévias, elaborar as hipóteses de que “Choveu” ou “Um caminhão pipa passou derramando água”. Então procura-se validá-las a partir de outras observações e conhecimento obtido em experiências anteriores. Por exemplo, a partir da nova observação de que “O telhado da casa está molhado”, é possível descartar a hipótese do caminhão, pois é conhecido que o caminhão pipa não seria capaz de molhar o telhado, validando, assim, a primeira hipótese e encontrando nela a melhor explicação. Esse método de inferência também é conhecido como inferência pela melhor explicação (BRANQUINHO, MURCHO e GOMES, 2006).

Há ainda outro processo de raciocínio denominado analogia, no qual parte-se do particular para o particular. Este processo ocorre com a percepção de características similares e a classificação destas de acordo com um propósito (LLOYD, 2005).

Por exemplo, um conjunto de indivíduos A, B e C, que possuem o seguinte conjunto de características em comum: cor branca, magro, cabelo preto e olhos azuis. Considerando que o indivíduo A possui ainda como característica a estatura alta é possível afirmar que B e C provavelmente terão esta característica mesmo que não esteja explícita na descrição.

O raciocínio analógico contribui para a solução de problemas ou explicações respondendo perguntas como: “A é para X o que B é para Z? “. A analogia é uma importante forma de resolução de problemas em que soluções de problemas análogos são utilizadas em novos problemas.

A analogia não necessariamente precisa ter um conjunto de objetos para extrair características similares. A partir da observação e o estudo de características de apenas um objeto é possível tentar aplicar em outro problema de domínio diferente. Uma aplicação do uso da analogia seria como, por exemplo, um indivíduo A com característica X e um indivíduo B com característica Z e a tentativa de identificar o quão próximo A é de B mesmo que X e Z não sejam próximos ou ainda, considerando que A e B possuem características parecidas como posso definir um Z para B baseado na relação X para A. Como um exemplo mais real, a observação do movimento das barbatanas de um peixe poderia ser útil na elaboração de um mecanismo para uma canoa se mover na água, tal qual um remo.

A Figura 1 mostra como a analogia pode ser utilizada na resolução de problemas através de um método denominado transformação analógica (CARBONELL, 1985). Neste processo um novo problema identificado é mapeado parcialmente para a identificação de problemas resolvidos anteriormente com características similares. Então tais problemas são recuperados e suas soluções adaptadas de forma a satisfazer o novo problema.

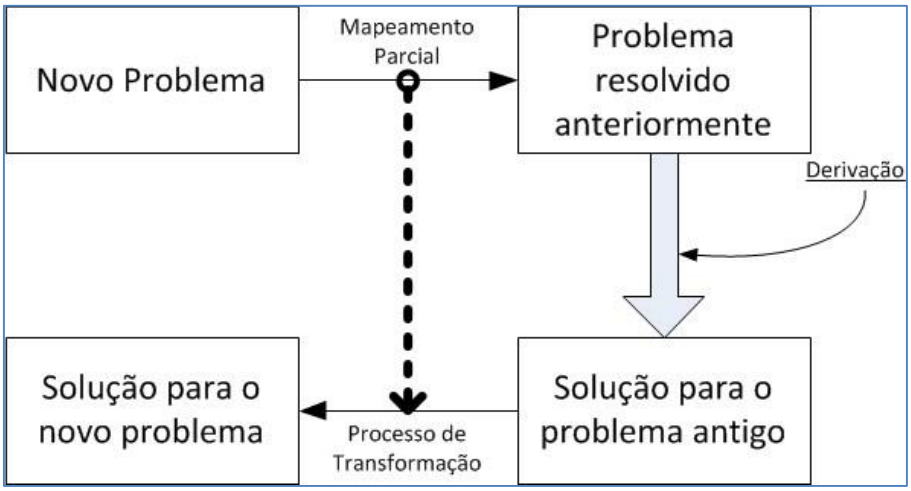


Figura 1: Transformação analógica para resolução de problemas (CARBONELL, 1985).

O raciocínio analógico é independente de domínio. Ele é capaz de elaborar novas soluções para um problema a partir de qualquer outro problema similar comparando suas características similares mesmo que de domínio diferentes como por exemplo a criação de uma asa de avião a partir do estudo da asa de um pássaro (WANGENHEIM e WANGENHEIM, 2003).

A Figura 2 ilustra de forma simplificada como funciona o processo em cada um dos tipos de raciocínio citados.

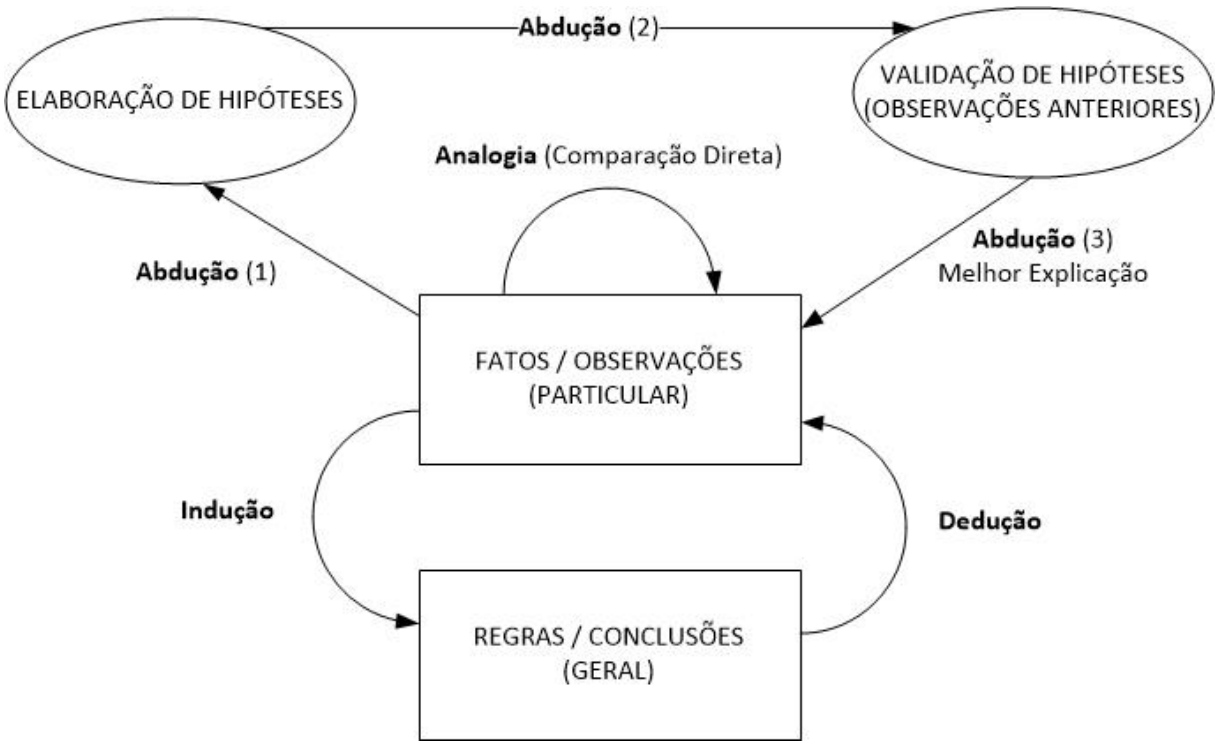


Figura 2: Tipos de raciocínio e seus fluxos.

Existe um tipo particular de raciocínio, denominado Raciocínio Baseado em Casos (RBC) que teve sua origem em modelos cognitivos de memória dinâmica (SCHANK, 1982), mas também em pesquisas originadas no estudo do raciocínio por analogia, em particular do estudo de Gick e Holyoak (1980), e que foram muito importantes para o avanço desta teoria. Motivo pelo qual alguns estudos consideram o RBC, portanto, uma especialização do raciocínio analógico (WANGENHEIM e WANGENHEIM, 2003).

O raciocínio analógico e o RBC diferenciam-se principalmente no que diz respeito a abrangência do domínio do novo problema a ser solucionado e do caso recuperado para reutilização da solução. O analógico baseia-se na extensão e transformação de conceitos similares entre diferentes domínios, como por exemplo, o conhecimento dos peixes e a forma como nadam pode ser utilizado na criação de um mecanismo naval com características ou objetivos similares tal qual um remo que ajuda um barco a se mover na água. Já o RBC é direcionado para encontrar situações similares dentro de um domínio e adaptar soluções anteriormente utilizadas como na sentença de casos jurídicos por um juiz que baseia-se na legislação, mas também, em sentenças anteriores de casos similares.

2.2 RBC – Raciocínio Baseado em Casos

Com origem no modelo de memória dinâmica de Schank (1982) e complementado por Kolodner (1983) através do desenvolvimento do primeiro sistema RBC baseado neste modelo cognitivo (SUN, HAN e DONG, 2008), o RBC é uma especialização do raciocínio analógico que utiliza experiências de casos anteriores para resolver novos problemas através da adaptação e reutilização de soluções anteriores, aprendidas de casos com problema similar ao novo (KOLODNER, 1991). É uma das abordagens mais populares e disseminadas para o desenvolvimento de sistemas baseados em conhecimento (WANGENHEIM e WANGENHEIM, 2003), já que é considerada similar à forma que os seres humanos resolvem problemas (AAMODT e PLAZA, 1994).

O RBC tem recebido bastante atenção nos últimos anos devido a algumas características, tais como a capacidade de adquirir conhecimento de novos casos e fato de não requerer um modelo de domínio explícito, possibilitando um simples recolhimento de casos anteriores (SRINIVASAN et al., 2011). Apesar disso, é a

similaridade do problema que possibilita o reúso de sua solução, e por isso os casos, cujas soluções serão reutilizadas deverão estar dentro do domínio do novo problema para que haja uma maior efetividade na resolução do problema.

A utilização de um sistema RBC no apoio à tomada de decisões é uma solução mutualmente benéfica, uma vez que o sistema pode ser capaz de apoiar decisões com base em suas próprias decisões e decisões tomadas anteriormente por humanos (SRINIVASAN et al., 2011).

Vários exemplos da vida diária podem ser utilizados para demonstrar como os seres humanos utilizam casos como uma forma de resolução de problemas: um profissional jurídico que reforça seus argumentos com base na jurisprudência que contempla a situação; um arquiteto que estuda as plantas de um prédio anteriormente construído para planejar um prédio similar; e um setor de suporte que tenta corrigir o problema de um computador baseando-se em solução anterior de problemas similares.

O enfoque RBC é capaz de utilizar o conhecimento específico de soluções de problemas concretos, experimentadas anteriormente, denominadas casos, ao invés de se basear unicamente em conhecimento generalizado de um domínio. A Figura 3 mostra o modelo da abordagem RBC, na qual, no surgimento de um novo problema, é realizada uma busca por casos com maior similaridade na base de casos aprendidos, e, após seleção do caso com maior similaridade a adaptação de sua solução é realizada, e, por fim, aplicada ao novo problema (WANGENHEIM e WANGENHEIM, 2003).

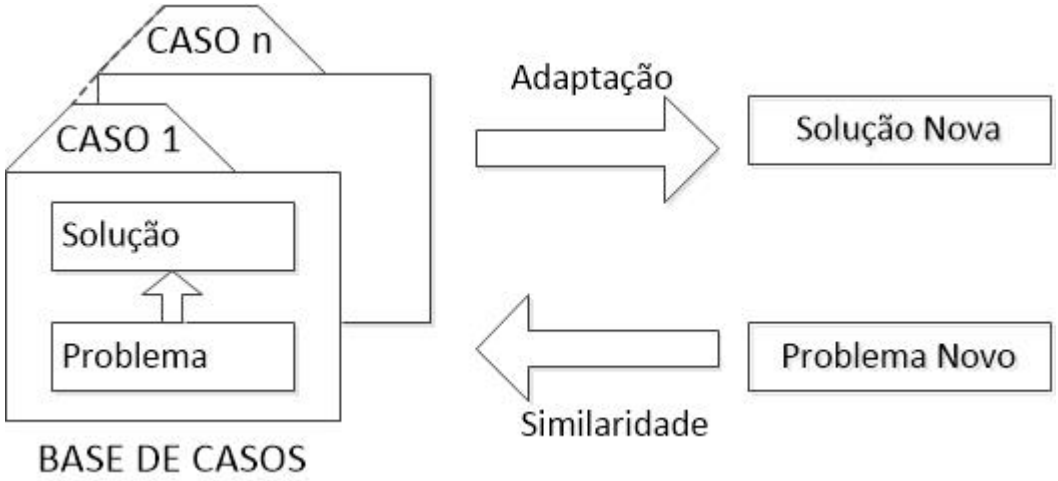


Figura 3: Modelo básico da abordagem RBC (WANGENHEIM e WANGENHEIM, 2003).

Um sistema RBC é extremamente dependente da estrutura e do conteúdo de sua base de casos, porém o conhecimento não está presente apenas nos seus casos, mas também nos seus elementos básicos (MATTOS, WANGENHEIM e PACHECO, 1999).

Os elementos básicos de um sistema RBC (WANGENHEIM e WANGENHEIM, 2003) são:

a) um mecanismo baseado no conhecimento para representação dos casos;

b) um mecanismo para recuperação de casos e uma medida de similaridade para identificar os casos relevantes para o problema atual, isto é, os casos aprendidos com maior similaridade;

c) mecanismos de adaptação que permitam adaptar os casos recuperados a fim de satisfazer a situação presente;

d) mecanismos de aprendizado para que o sistema seja capaz de lembrar os casos resolvidos com o objetivo de reutilizar sua solução em novos casos.

Sendo assim, para criar um sistema RBC é necessário definir como estará organizada a base de casos, como serão representados os casos, e como será realizada cada atividade do ciclo de raciocínio RBC. Para tanto, será explicado a seguir o ciclo de vida do RBC e serão mostradas as formas mais comuns de organização da base de casos RBC e as diferentes estratégias de execução das atividades dos mecanismos de representação de casos, recuperação e análise de similaridade, adaptação e aprendizado.

2.2.1 Ciclo de Vida RBC

Em relação ao ciclo de vida RBC, segundo Wangenheim e Wangenheim (2003), o mais aceito é o definido por Aamodt e Plaza (1994), que consiste em um ciclo de raciocínio contínuo composto por quatro tarefas principais: recuperação de casos similares; reutilização da informação e conhecimento daquele caso para resolver o problema após adaptação; revisão da solução proposta; retenção da experiência obtida na resolução do caso atual.

O ciclo se inicia com a obtenção de um novo caso-problema, e a partir dos seus dados é realizada a recuperação de casos similares. Em seguida, adapta-se a solução do caso mais similar para a situação do novo problema e, posteriormente,

essa solução é testada e avaliada. Por fim, a experiência que foi obtida com a resolução desse novo caso é retida e adicionada à base de casos, conforme ilustrado na Figura 4.

Figura 4: Ciclo do RBC (AAMODT e PLAZA, 1994).

2.2.2 Base de Casos

Para a criação de um sistema computacional que utiliza o método de raciocínio do RBC é necessário que haja a representação de uma base de casos a ser utilizada para armazenamento e organização dos casos, e recuperação futura para reutilização da solução.

Aamodt e Plaza (1994) citam em seu trabalho dois modelos de memória de casos bem influentes que correspondem à forma como os casos são organizados na base de casos RBC. O modelo de memória dinâmica de Schank (1982) e Kolodner (1983), e o modelo categoria-exemplar de Porter e Bareis (1986).

O modelo de memória dinâmica possui uma estrutura hierárquica que é denominada “pacotes de organização de memória episódica”. A ideia básica desse

modelo é organizar casos específicos de acordo com propriedades similares, em agrupamentos denominados Episódios Generalizados (EGs), sendo estes compostos de três diferentes tipos de objetos: normas, casos e índices (AAMODT e PLAZA, 1994).

As normas são traços comuns a todos os casos inseridos em um Episódio Generalizado (EG), os índices são traços que distinguem os casos do EG de outros. Um índice é composto por um nome e um valor, e pode apontar para um EG mais específico ou mesmo para um caso específico.

A Figura 5 ilustra um modelo de um EG composto de casos denominado “Episódio Generalizado 1”, que possui um índice que aponta para outro EG denominado “Episódio Generalizado 2”, mais específico.

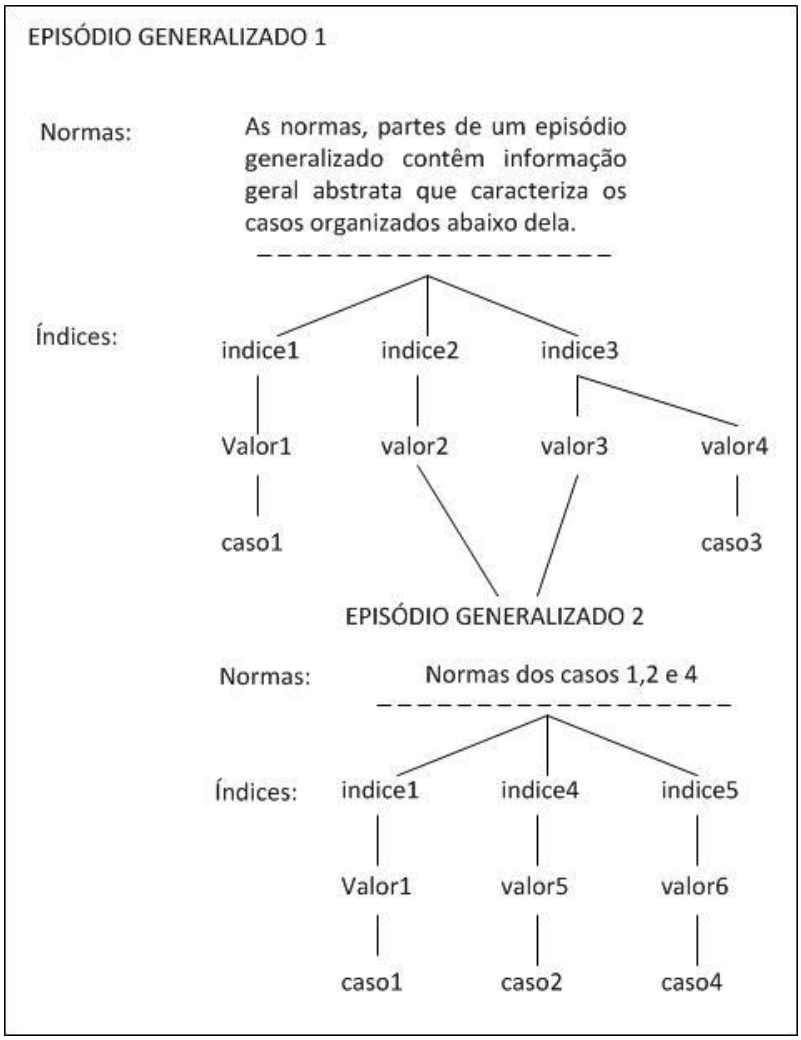


Figura 5: Estrutura de casos e EGs no modelo de memória dinâmica (AAMODT e PLAZA, 1994).

O modelo denominado Categoria e Exemplar, por sua vez, propõe um modo alternativo de organização de casos na memória (AAMODT e PLAZA, 1994). Nesse modelo os casos possuem a denominação de “Exemplar”. Diferentes características recebem diferentes importâncias na descrição de um “Exemplar” em relação a uma categoria (PORTER e BAREIS, 1986).

A Figura 6 ilustra essa forma de estrutura de memória de casos, na qual há um conjunto de quatro características em uma categoria denominada “Categoria-1”. Nessa categoria existem dois exemplares denominados “Exemplar-1” e “Exemplar-2”, sendo que o “Exemplar-2” possui prototipação fraca em relação a “Categoria-1”, pois possui menor similaridade ao seu conjunto de características, em particular, das características 1 e 4.

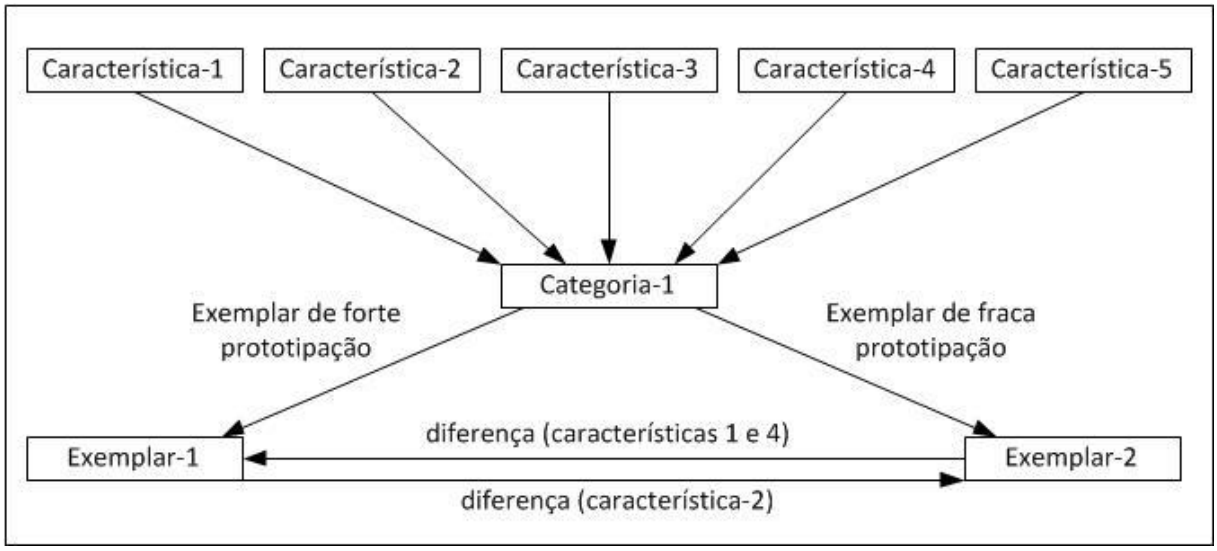


Figura 6: Estrutura Categoria-Exemplares (AAMODT e PLAZA, 1994).

2.2.3 Representação dos Casos

Um dos primeiros procedimentos ao se montar um modelo de sistema RBC é definir como cada caso será representado. O RBC possui grande dependência da estrutura da sua coleção de casos (AAMODT e PLAZA, 1994), por isso, é necessário que uma linguagem seja bem definida. Vários tipos de linguagem ou formalismos de representação de modelos de domínio ou casos, como grafos ou representações atributo-valor podem ser utilizados (WANGENHEIM e WANGENHEIM, 2003).

Inicialmente o problema é definir o que deve ser armazenado em um caso, definir uma estrutura apropriada e decidir como organizar a memória de casos de

forma tal que haja um eficiente meio de recuperação e reúso desses (AAMODT e PLAZA, 1994).

Um caso pode ser composto apenas da descrição do problema e da solução, ou pode ter como descrição o passo-a-passo de toda a tarefa de elaboração de nova solução. Essa descrição mais completa pode ser útil em atividades intermediárias da recuperação de casos (BURKHARD, 1998).

Os casos podem ser representados de diversas formas, dentre elas estão representações simples com a utilização de uma lista simples de pares do tipo atributo-valor, e representações pouco mais sofisticadas como a orientada a objetos, e as que utilizam árvores, grafos e redes semânticas. Sendo esta última um tipo específico de representação com grafos na qual o conhecimento é representado em uma estrutura similar a uma rede (WANGENHEIM e WANGENHEIM, 2003).

a) Representação Atributo-Valor

Neste tipo de representação os conjuntos de atributos normalmente são fixos para todos os casos na base e cada dado é representado por um par atributo-valor.

Na Figura 7, observa-se a representação de um possível mapeamento da representação do mundo real. No lado esquerdo consta o caso em linguagem natural e do lado direito a representação formal do mesmo problema em uma lista do tipo atributo-valor.

O cliente deu entrada no Helpdesk de uma empresa com uma impressora aparentemente com defeito, modelo Epson Lx-300+, o mesmo informou que o defeito da impressora é que a mesma não funciona. O técnico pediu que o cliente informasse o status das luzes do painel da impressora. O cliente informou que a luz de estado do papel estava apagada, as luzes de estado das tintas coloridas e prestas estavam apagadas e o estado do interruptor ligado. O técnico informou então que o problema era curto-circuito e que um técnico iria até o local para realizar a troca da fonte.	**Problema (Sintomas):** Problema: Impressora não funciona Modelo: Epson LX-300+ Luz de Estado do Papel: apagada Luz de Estado da Tinta colorida: apagada Luz de Estado da Tinta preta: apagada Estado do Interruptor: ligado **Solução:** Diagnóstico: Curto-Circuito Ação: Troca da fonte de alimentação

Figura 7: Mapeamento de um problema em linguagem natural para representação atributo-valor.

Neste modelo cada atributo é geralmente associado a um tipo de dados (domínio), tais como números, booleanos, datas, *string* e símbolos que podem ser não ordenados, ordenados ou taxonomia (WANGENHEIM e WANGENHEIM, 2003).

b) Representação orientada a objetos

Este modelo de representação possui características bem similares à forma de representação das conhecidas linguagens orientadas a objetos. A base de casos é formada por um conjunto de objetos, instâncias de uma classe. Neste tipo de representação é possível realizar a modelagem de relacionamentos entre diferentes tipos de objetos, tais como relações taxonômicas e composicionais. Os detalhes de cada caso são representados através de um conjunto de pares do tipo atributo-valor (WANGENHEIM e WANGENHEIM, 2003).

Com base nesse modelo é possível modelar o domínio da aplicação como pode ser observado através da Figura 8, na qual podem ser identificadas duas classes de um domínio jurídico denominadas Processo e Parte, ilustradas no lado direito, e uma instância da classe Caso, composta de propriedades correspondentes à instâncias das classes Processo e Parte, no lado esquerdo da figura.

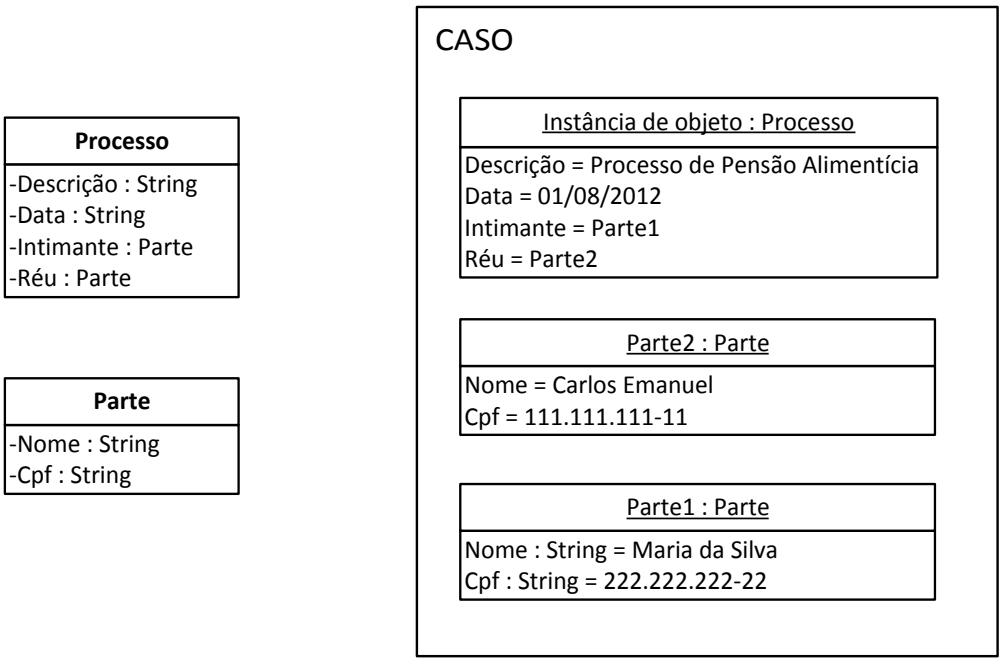


Figura 8: Exemplo de representação orientada a objetos.

Dentre as vantagens da modelagem de domínio orientada a objetos está a representação estruturada de casos, permitindo representação de informações estruturais e relacionais diretamente. Esse modelo possui ainda visualização mais compacta e permite a identificação de casos similares para reúso da solução a partir de comparação direta de instâncias (WANGENHEIM e WANGENHEIM, 2003).

c) Redes Semânticas

Nesse tipo de representação os nodos representam unidades conceituais e as arestas representam os relacionamentos (WANGENHEIM e WANGENHEIM, 2003).

Lenz e Buckhard (1996) propuseram um tipo específico de rede semântica para a recuperação de casos denominada Rede de Recuperação de Casos (RRC).

O item fundamental desse tipo de rede são as denominadas Entidades de Informação (EIs), que são capazes de representar qualquer conhecimento básico assim como as relações atributos-valor (LENZ e BUCKHARD, 1996).

Um caso corresponde a um conjunto de EIs e a base de casos é uma rede de nodos para as EIs. Nodos EIs similares são conectados através de arestas de similaridade, e os nodos de casos, correspondentes aos valores de atributos dos casos, são associados através de arestas de relevância. A Figura 9 ilustra um exemplo de representação de casos através de uma RRC, no domínio de uma empresa de viagens.

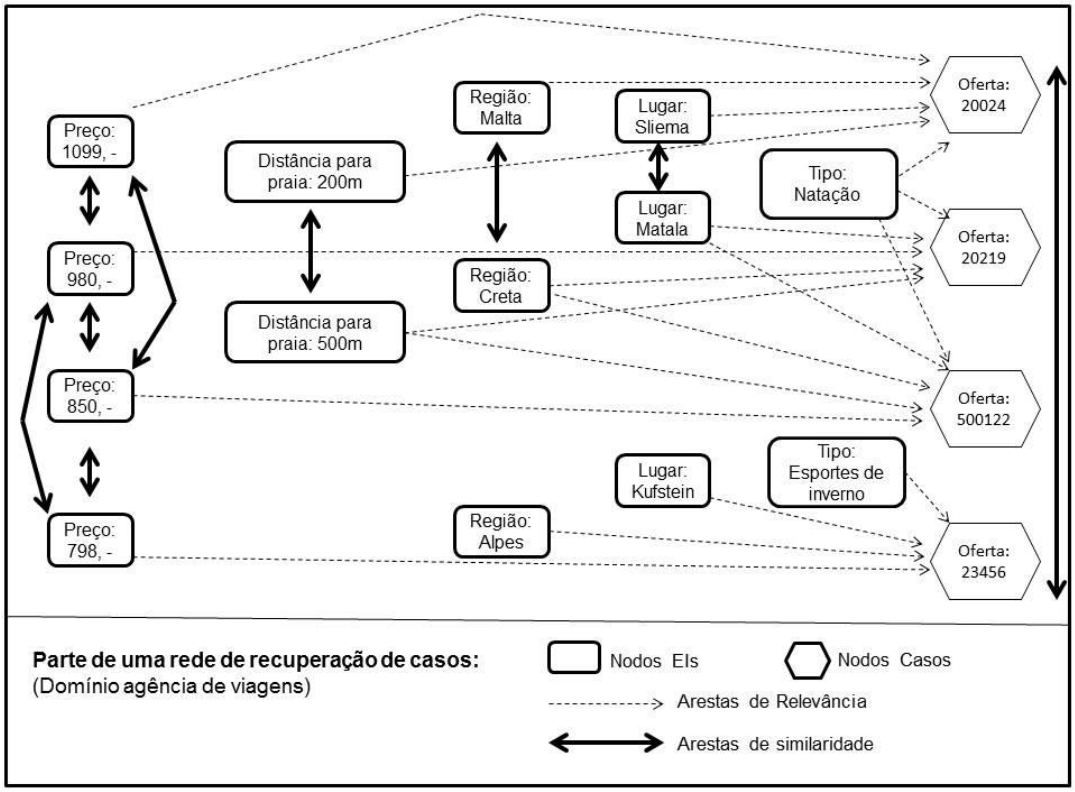


Figura 9: RRC no domínio de uma agência de viagens (LENZ e BUCKHARD, 1996).

d) Árvores K-D

As árvores K-D são uma forma de representação que utiliza grafos do tipo acíclico e são orientados com um nodo especial denominado nodo raiz. É uma árvore de pesquisa binária k-dimensional, que divide a base de casos em partes menores de forma interativa (WESS, ALTHOFF e DERWAND, 1994).

A Figura 10 ilustra casos representados em uma árvore K-D. Na Figura é possível observar como a base de casos é dividida. A parte I, subdivide-se na parte III e II que por sua vez possui uma subdivisão IV. Comparando-se as partes da árvore K-D aos EGs, essa forma de representação dos casos se assemelha muito ao modelo de memória de Schank (1982) e Kolodner (1983).

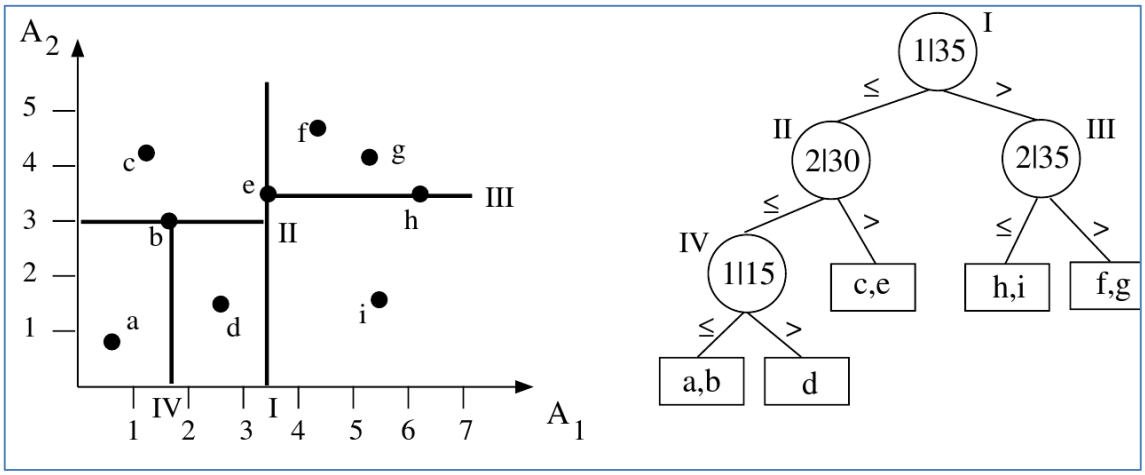


Figura 10: Espaço de busca bidimensional hipotético e correspondência com árvore de busca K-D (WESS, ALTHOFF e DERWAND,1994).

e) Ontologias

Outra forma pela qual os casos podem ser representados é através de instâncias em uma ontologia. Para isso é necessário que haja um corpo representativo do domínio para realizar o aprendizado das ontologias, o qual pode ser de forma manual, semiautomatizada ou automatizada. É possível ainda utilizar técnicas como o *dataminning* para enriquecê-las com mais relações e conceitos. Além disso, para a utilização de ontologias para a representação é importante que exista um meio que proporcione executar *querys*, isto é, consultas que permitam a identificação e recuperação de instâncias similares (MUSTAPHA et al., 2010).

Dentre as vantagens da representação com ontologias, destaca-se a capacidade de representação semântica do caso, além da reutilização da ontologia para outros sistemas baseados em conhecimento.

Como pudemos ver nesse tópico existem diversas formas de representar casos para utilização em um sistema RBC. Considerando que o sistema deverá ser capaz de recuperar os casos mais similares para adaptação da solução, é necessário que seja avaliado e escolhido o modo mais adequado para representação de casos para que haja uma maior facilidade e melhor efetividade no processo de recuperação e análise de similaridade de casos.

2.2.4 Recuperação de Casos e Análise de Similaridade

Outro mecanismo essencial de um sistema RBC é o de recuperação de casos e análise de similaridade. Esse mecanismo utiliza um modelo de recuperação e uma medida de similaridade que permite a recuperação de casos similares ao novo problema para reúso de sua solução.

Um caso é composto de entidades de informação (EIs), independente da forma como é representado, e que correspondem aos valores dos atributos do caso. Para que haja boa eficiência na recuperação de casos é necessário que sejam determinados os índices a partir das EIs identificadas. Esses índices são os atributos considerados como relevantes no problema e que irão permitir a distinção dos casos armazenados na base e proporcionarão a identificação dos mais úteis para resolução de um novo problema (WANGENHEIM e WANGENHEIM, 2003).

Para o julgamento de similaridade existem algumas suposições básicas a serem consideradas (WANGENHEIM e WANGENHEIM, 2003). São elas:

- Reflexividade: um fato ou objeto é similar a si mesmo.
- Simetria: se X é similar de Y, então Y é também similar de X.
- Transitividade: se X é similar de Y e Y é similar de Z, então X é similar de Z.
- Monotonicidade: as semelhanças entre dois objetos cresce monotonicamente à medida que há redução das diferenças e aumento das correspondências.

Além disso, ao projetar e implementar um sistema RBC deve-se considerar alguns conceitos em relação à similaridade: a similaridade entre uma consulta e um caso implica em utilidade; a similaridade deve ser baseada prioritariamente em fatos; devido ao fato de que casos podem ter diferentes níveis de utilidade, a similaridade deve ser mensurada (BURKHARD, 1998).

Existem diversas formas de definir uma medida de similaridade para a julgar a semelhança entre a descrição de casos. A forma mais conhecida é através de uma medida numérica de distância (WANGENHEIM e WANGENHEIM, 2003). A utilização de algoritmos para identificação do vizinho mais próximo, que calcula a distância entre os casos através de um somatório dos valores de similaridade entre as características que descrevem um problema, é a forma mais comum para obtenção

do valor de similaridade (FINNIE e SUN, 2002). Para tanto, utiliza-se um modelo matemático que deve ser capaz de estabelecer um valor de similaridade entre casos. Esse método pode ser acrescido de pesos através da aplicação de média ponderada (WANGENHEIM e WANGENHEIM, 2003).

Para o cálculo do valor de similaridade entre casos, as características correspondentes que descrevem os problemas devem ter sua similaridade calculada. Essas características podem ter diferentes tipos, dentre eles números e *strings*. Cada característica deve ter critérios estabelecidos para calcular o valor de similaridade. No caso dos números podem ser definidos intervalos, conjuntos ou diferenças que caracterizam um determinado valor de similaridade. Na comparação de *strings* podem ser consideradas correspondências exatas ou outros critérios como a contagem de palavras e comparação sintática das palavras (WANGENHEIM e WANGENHEIM, 2003). No entanto, é a comparação com base na semântica que garante uma melhor qualidade nos casos recuperados, visto que uma mesma palavra pode ter significados diferente de acordo com o domínio, e, por isso, deve ser considerado.

O mecanismo de recuperação de casos tem como objetivo encontrar um ou mais casos solucionados similares ao problema atual para recuperação e reúso da solução. O processo de recuperação de casos pode ser descrita por meio de um conjunto de atividades a serem realizadas (WANGENHEIM e WANGENHEIM, 2003).

a) Assessoramento da situação: formulação de uma consulta representada por um conjunto de valores de atributos relevantes (índices) do novo caso.

b) Casamento: Identificação de conjunto de casos similares em relação a descrição da consulta.

c) Seleção: Achar o melhor casamento dentre os casos similares selecionados.

A representação do processo de recuperação com as atividades supracitadas está ilustrado na Figura 11.

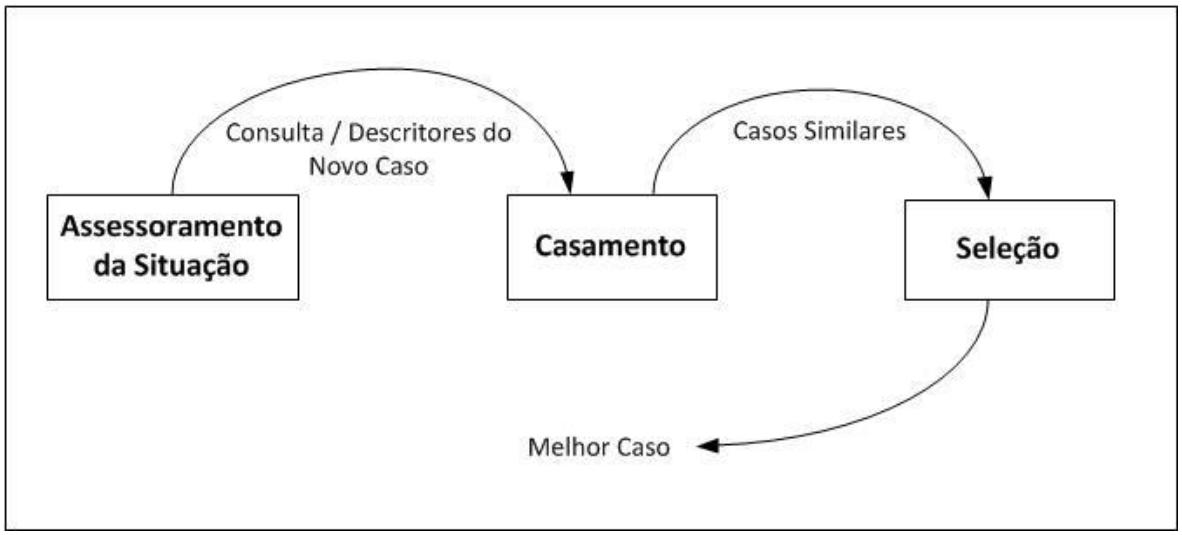


Figura 11: Processo de Recuperação de Casos em um sistema RBC.

O processo de recuperação pode ser realizado com diferentes técnicas (WANGENHEIM e WANGENHEIM, 2003).

a) Recuperação sequencial

Técnica simples pela qual todos os casos da base são comparados com o novo caso e ordenados de acordo com o valor de similaridade. Em seguida, são retornados os “n” casos mais similares.

b) Recuperação de dois níveis.

Essa técnica de recuperação é similar a anterior, com a diferença que o processo é realizado em duas etapas. Na primeira etapa é feita uma pré-seleção com base em uma seleção simples, em seguida, na segunda etapa é feita uma comparação mais complexa utilizando análise estrutural e similaridade mais ampla dos atributos, buscando-se os melhores casos (GENTNER e FORBUS, 1991).

c) Recuperação orientada a índices

Também é realizada em duas etapas assim como a recuperação de dois níveis. Na primeira fase, denominada pré-processamento, todos os casos são analisados e uma estrutura de índices é gerada com base nos critérios da consulta. Em seguida, na segunda fase, denominada recuperação, essa estrutura de índices é percorrida e os casos mais similares são recuperados.

d) Recuperação com Árvores K-D

Processo de recuperação específico para a representação de casos em grafos de árvores K-D. Nesse tipo de estrutura a recuperação é realizada de forma recursiva na árvore de busca binária de acordo com a medida de similaridade dos casos das folhas.

e) Recuperação em RRC

Assim como as árvores K-D, as redes de recuperação de casos também têm uma forma peculiar de recuperação de casos. Neste modelo de representação em redes semânticas, o processo recuperação ocorre através da ativação das entidades de informação dadas do caso de recuperação, isto é, da consulta. Em seguida a ativação se propaga pela rede de acordo com a similaridade, descrita através das arestas de similaridade, até que os nodos dos casos sejam alcançados. Por fim, a coleta da ativação é adquirida nos nodos dos casos em que houve associação.

f) Recuperação em Ontologias

Em bases de casos que utilizam ontologias para representação de casos, poderá haver conhecimento semântico associado aos casos, o que irá permitir a recuperação através dos índices e mecanismos de indexação similar aos outros processos de recuperação de casos, mas também pode ser utilizado um mecanismo baseado em semântica que pode melhorar a forma de elaboração das consultas e indexação através da semântica, permitindo uma maior precisão nos resultados obtidos (MUSTAPHA et al., 2010).

Uma técnica semântica de recuperação de casos ajuda na construção de uma técnica inteligente de recuperação de casos, visto que, ao utilizar semântica, é possível ter uma medida de similaridade mais precisa entre a descrição do novo problema e a descrição dos problemas dos casos solucionados na base de casos através da semântica, proporcionando uma melhor efetividade.

2.2.5 Adaptação dos Casos

Uma vez que um caso similar foi recuperado não significa que a solução desse encaixará perfeitamente ao novo problema. Muitas vezes haverá necessidade

de adaptação da solução recuperada ao problema atual (WANGENHEIM e WANGENHEIM, 2003).

Do ponto de vista do RBC há uma relação natural entre a complexidade da resolução de um problema e a complexidade de adaptação da solução recuperada. Os tipos de adaptação variam de acordo com a sua complexidade (WILKE E BERGMANN, 1998). A Figura mostra a hierarquia dos tipos de adaptação que serão explicados a seguir.

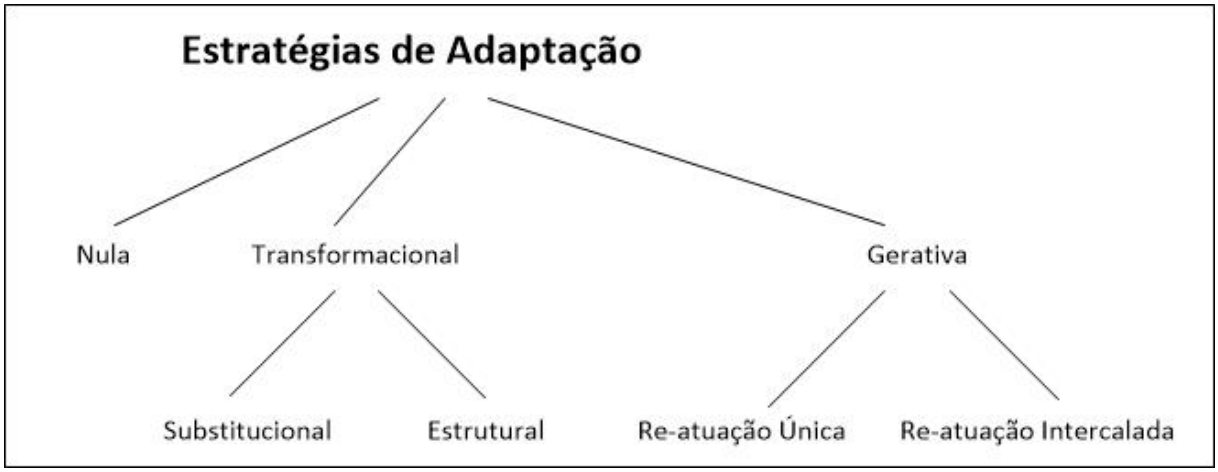


Figura 12: Estratégias de Adaptação de Casos. Adaptada de (WILKE e BERGMANN, 1998).

a) Adaptação Nula

Não há necessidade de adaptação.

b) Adaptação Transformacional

Há necessidade de adaptar a solução anterior para o novo problema. Permite a reorganização, exclusão ou inclusão de elementos da solução anterior. Requer normalmente utilização de um conjunto de regras de adaptação. Esse tipo subdivide-se ainda em outros dois tipos.

- Adaptação Substitucional: neste tipo mudam-se apenas valores de atributos da solução encontrada. A estrutura da solução permanece a mesma e não há troca de atributos. Normalmente utilizada quando a solução encontrada é bastante adequada ao novo problema.

- Adaptação Estrutural: permite alteração da estrutura da solução através da adição e exclusão de atributos. Essas alterações estruturais obedecem a regras e operadores de transformação.

c) Adaptação Gerativa

Este tipo de adaptação é mais apropriado para tarefas mais complexas de resolução de problemas, e pode levar a uma constante necessidade de adaptação durante a tentativa de resolução do problema (WANGENHEIM e WANGENHEIM, 2003).

O processo gerativo não só transfere a solução anteriormente à ser adaptada, mas todo o processo derivacional da solução anterior, isto é, todas as rotas seguidas para obtenção da solução anterior podem ser seguidas a fim de obter a solução para o novo problema.

Existem dois subtipos (WANGENHEIM e WANGENHEIM, 2003):

- reatuação única: Identifica-se primeiramente qual porção das ações traçadas para geração da solução anterior podem ser utilizadas, e, em seguida, são executadas pelo solucionador de problemas.

- reatuação intercalada: Este tipo por sua vez, intercala entre a repetição das ações da solução anterior e novas decisões do solucionador de problemas. O rastreio da solução anterior só é utilizado até o ponto em que possui benefício à elaboração da nova solução.

d) Adaptação Composicional

É um tipo de adaptação feita a partir da combinação de diversos tipos de adaptação, pela qual a solução para o novo caso é obtida através da recuperação e adaptação de uma composição das soluções de mais de um caso.

e) Adaptação Hierárquica

Também realizada a partir da combinação de outros tipos de adaptação. Nesse tipo de adaptação os casos são armazenados em uma estrutura hierárquica e a adaptação é realizada no sentido *top-down* utilizando a hierarquia de casos.

2.2.6 Aprendizado de Casos

O processo de retenção dos novos casos é o processo de incorporação do conhecimento do novo caso solucionado ao conhecimento já existente de casos anteriores. Aprendendo, assim, tudo que foi útil para resolução de um novo problema e as características desse problema (WANGENHEIM e WANGENHEIM, 2003).

Um sistema RBC pode ter três tipos de retenção: não ter retenção, reter novos casos solucionados pelo sistema e/ou retenção de casos a partir de documentos (WANGENHEIM e WANGENHEIM, 2003).

Não ter retenção é quando um sistema RBC desconsidera o aprendizado de novos casos. Para tanto, o sistema deve ter o domínio da aplicação bem definido e modelado, e composto de uma ampla variedade de casos solucionados do domínio previamente definidos.

Reter novos casos solucionados é quando são retidos os novos problemas e suas soluções que foram descritas com base na adaptação de problemas anteriores ou no conhecimento de domínio da aplicação.

A retenção de casos a partir de documentos acontece quando o conhecimento é retido de forma assíncrona ao processo de resolução de problemas, sendo assim inserido na base a partir de documentos disponíveis, tal como no exemplo jurídico, a inserção de novas jurisprudências em uma base de casos.

O processo de retenção de conhecimento em um sistema RBC pode ser divido em três etapas: extração de conhecimento, indexação de casos e integração na base de casos.

A fase da extração corresponde à seleção das estruturas do conhecimento, entidades de informação, a partir das fontes das novas experiências tais como documentos, manuais técnicos ou descrição de problemas com as respectivas soluções.

Em seguida, na fase da indexação de casos, as EIs obtidas devem ser identificadas para determinar quais serão índices, procedimento que, em um RBC, é um grande problema a ser realizado de forma automatizada, devido ao volume e complexidade dos dados. Por isso, uma solução que pode ser adotada é considerar todas elas como índices (WANGENHEIM e WANGENHEIM, 2003).

Por fim, a última etapa é a integração dos casos, que é a adição do mesmo à base de casos, podendo ser necessários ajustes de índices existentes.

No entanto, um sistema RBC não deve obrigatoriamente reter todos os novos casos solucionados, pois não implica necessariamente em uma melhoria na qualidade das soluções recuperadas e pode ainda reduzir a performance na recuperação de casos. Algumas políticas para identificar quais casos devem ser retidas devem ser definidas (ONTAÑÓN e PLAZA, 2003).

A retenção de casos é apenas o processo de retenção de conhecimento a partir de novos casos solucionados. O conhecimento pode ser ainda adquirido a partir do estudo de casos aprendidos como forma de aprimorar o comportamento do sistema, isto é, a forma como são executadas as atividades de um sistema RBC, como por exemplo, o aprimoramento do modelo de similaridade e definição dos pesos atribuídos aos índices.

O aprendizado baseado em casos é uma característica essencial em um sistema RBC, pois é essa etapa que vai permitir uma base de casos rica, uma melhora na performance do sistema e na sua capacidade de atender à diversas situações (WANGENHEIM e WANGENHEIM, 2003).

A Figura 13 exemplifica através de um modelo gráfico o funcionamento do aprendizado em RBC. Através da entrada de uma sequência de casos exemplares representados por $C_1,...,C_k$, o processo de aprendizado utiliza uma medida de similaridade sim_i para identificar um conceito a ser aprendido formando conjuntos de casos similares CB_i representado por tuplas (CB_n, sim_n) (WANGENHEIM e WANGENHEIM, 2003).

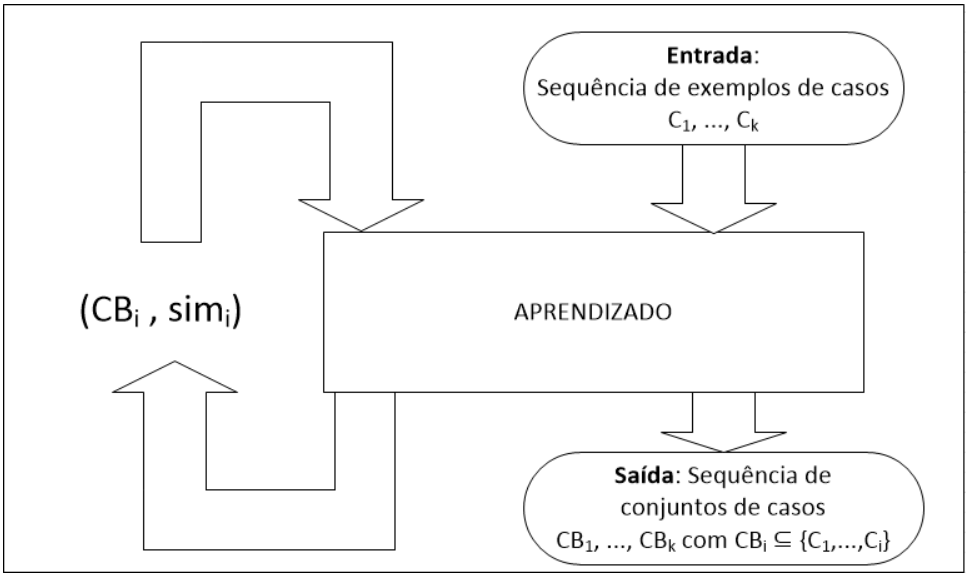


Figura 13: Modelo Gráfico do aprendizado baseado em casos (WANGENHEIM e WANGENHEIM, 2003).

2.3 Ontologias

A Inteligência Artificial (IA) considera ontologias como especificações formais de conceitos de um domínio de interesse, onde seus relacionamentos, restrições e axiomas são representados definindo um vocabulário comum para compartilhamento de conhecimento. São a especificação explícita de uma conceitualização (GRUBER, 1995).

Utilizadas para o formalismo de representação do conhecimento e com suporte ao processamento semântico, ontologias são utilizadas para representação do conhecimento de um domínio de uma aplicação. A utilização de ontologias permite que sistema modernos de conhecimento possuam maior efetividade na representação e recuperação de conhecimento (GIRARDI, 2001).

Ontologias representam conhecimento e informação em diferentes tipos de níveis de abstração permitindo, assim, a reutilização dessa estrutura. Uma forma de classificação hierárquica das ontologias de acordo com seu nível de dependência pode ser definida como: ontologias de alto-nível, que descrevem conceitos gerais que são independentes de um problema em particular ou domínio; ontologias de domínio, que descrevem o vocabulário referente ao domínio de uma forma genérica; ontologias de tarefa, que são ontologias direcionadas para uma tarefa específica dentro de um domínio; ontologias de aplicação, que descrevem especializações das ontologias de domínio e tarefas para uma aplicação em particular (GIRARDI, 2001).

A Figura 14 representa essa classificação das ontologias de acordo com seu nível de dependência.

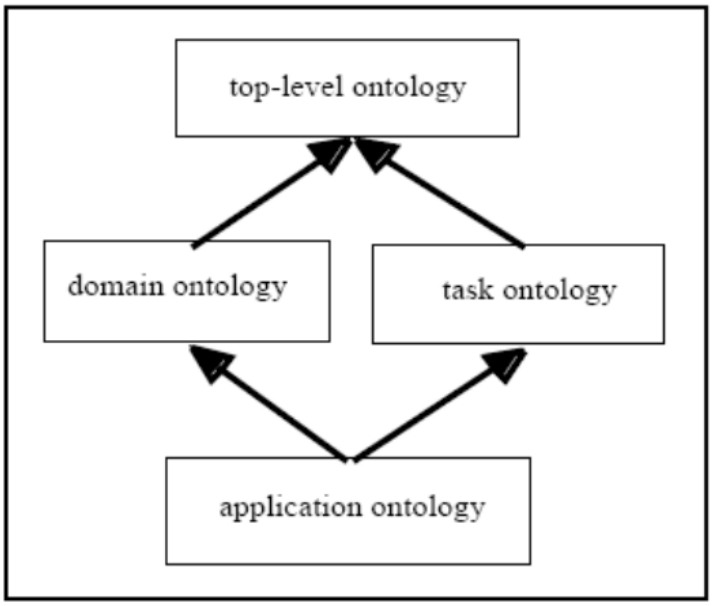


Figura 14: Taxonomia das Ontologias (GIRARDI, 2001).

Uma ontologia é composta por um lado de conceitos e relacionamentos taxonômicos, definindo uma hierarquia de conceitos, e também relacionamentos não taxonômicos, e por outro, por instâncias de conceitos e afirmações sobre as mesmas. Deve ser formal e, portanto, compreensível para os agentes de software e outras entidades computacionais. Desta forma, as ontologias podem fornecer um vocabulário comum entre várias aplicações e por isso também podem ser compartilhadas. Esta estrutura de representação de conhecimento, normalmente, consiste em um conjunto de classes organizadas hierarquicamente, descrevendo determinado domínio. Formalmente, uma ontologia pode ser definida de acordo com a equação:

$$\mathcal{O} = (C, H, I, R, P, A)$$

onde:

a) $C = C^C \cup C^I$ é o conjunto de entidades da ontologia. Representa as entidades do domínio sendo modelado e são designadas por um ou mais termos em linguagem natural. O conjunto C^C é formado por classes, ou seja, conceitos que representam entidades genéricas que descrevem um conjunto de objetos (por exemplo, “Pessoa” $\in C^C$). O conjunto C^I é formado por instâncias das classes, ou seja, entidades únicas no domínio (por exemplo “JoaoSilva” $\in C^I$);

b) $H = \{\text{tipo\_de}(c_1, c_2) |\ c_1 \in C^C \wedge c_2 \in C^C\}$ é o conjunto das relações taxonômicas entre os conceitos. Definem a hierarquia de conceitos e são denotadas por $\text{tipo\_de}(c_1, c_2)$ significando que c_1 é um tipo de c_2. Um exemplo desse relacionamento é $\text{tipo\_de}(Autor, Pessoa)$;

c) $I = \{\text{e\_um}(c_1, c_2) | c_1 \in C^I \wedge c_2 \in C^C\}$ é o conjunto de relacionamentos entre classes e suas instâncias (relacionamento "é um") de uma ontologia, por exemplo $\text{e\_um}(Joao\ Silva, Pessoa)$;

d) $R = \{\ rel_k\ (c_1, c_2, \dots, c_n)\ | \forall i,\ c_i \in C\}$ é o conjunto de relacionamentos não-taxonômicos de uma ontologia. Relacionamentos entre um objeto e instâncias de outras classes. Como exemplo $escreve(Autor, Livro)$ e $escreve(MonteiroLobato, OSaci)$;

e) $P = \{\ prop_k\ (c_i, tipo/valor)\ |\ c_i \in C\}$ é o conjunto de propriedades das entidades de uma ontologia. As propriedade relacionam conceitos a um tipo básico de dados, como *integer*, *real, boolean* ou *string*, ou podem relacionar instâncias a valores específicos de um tipo de dado. Alguns exemplos são $idade(Pessoa, Integer)$ e $idade(JoaoSilva, 30)$;

f) $A = \{Condition_x \implies conclusion_y\ (c_1, c_2, \dots\ c_n) | \forall_j,\ c_j \in C^C\}$ é um conjunto de axiomas, que são regras que permitem a checagem da consistência da ontologia, além de permitir a dedução de novos conhecimentos através de um mecanismo de inferência. O termo $Condition_x$ é dado por: $Condition_x := \{(\text{cond}_1, \text{cond}_2, \dots, \text{cond}_n) | \forall_z, \text{cond}_z \in H \cup I \cup R\}$. Por exemplo: $\{alugou(Cliente, Livro1), autor(Autor, Livro1\ ), autor(Autor, Livro2\ )\} \rightarrow$ provável-cliente $(Cliente, Livro2)$.

Para esclarecer essa definição, elaboramos uma ontologia de uma biblioteca como exemplo. A ontologia a seguir servirá para ilustrar os conceitos básicos de uma ontologia. Uma biblioteca aluga livros e lida com pessoas, que podem ser autores ou clientes. Os autores escrevem livros, enquanto os clientes alugam os livros e têm interesse por seus autores. A partir desta descrição identifica-se a ontologia a seguir, também mostrada na Figura 15.

a) $C^C = \{pessoa, autor, cliente, livro\}$

b) $C^I = \{MonteiroLobato, OSaci, JoaoSilva\}$

c) H = {tipo_de($raiz, pessoa$), tipo_de($raiz, livro$), ($pessoa, cliente$), ($pessoa, autor$)}

d) I = {é_um($MonteiroLobato, Autor$),é_um($OSaci, Livro$), é_um($Joao\ Silva, Cliente$)}

e) R = {tem_interesse($cliente, autor$), aluga($cliente, livro$), escreve($autor$,$livro$),escreve($MonteiroLobato$,$OSaci$)}

f) $P = \{valor(Livro, Real), valor(OSaci, 5.00)\}$

g) A = {aluga($cliente_1, livro_1$),escreve($autor_1, livro_1$)} ⇒ $tem\_interesse(cliente_1, autor_1)$

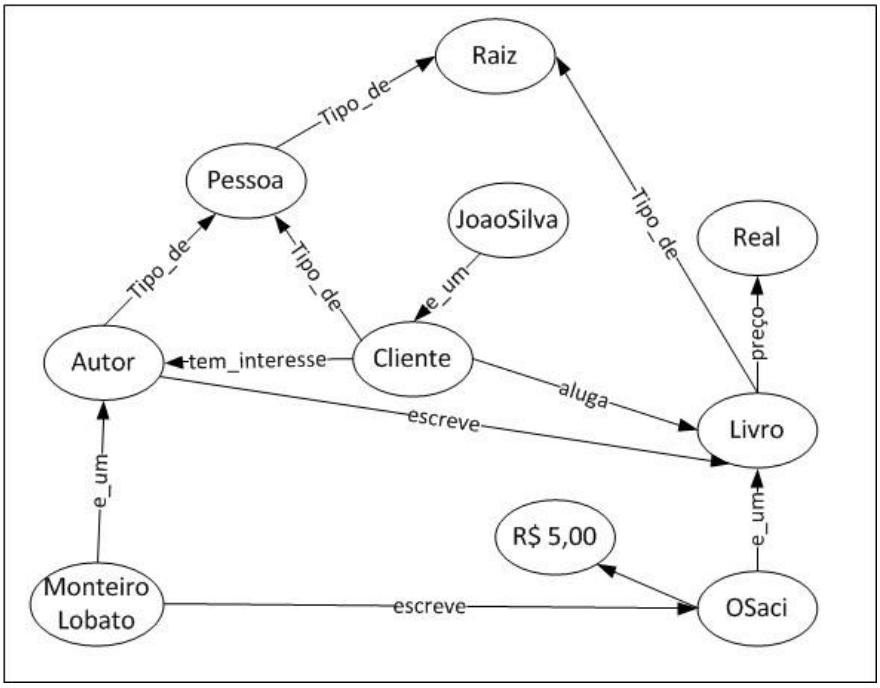


Figura 15: Exemplo de ontologia.

2.4 Agentes de Software

Agentes de software são entidades computacionais capazes de perceber o ambiente e agir sobre ele (WEISS, 1999). A percepção é realizada através de sensores e as ações dos agentes é realizada por meio de atuadores (RUSSEL e NORVIG, 2004). Esta interação está representada na Figura 16.

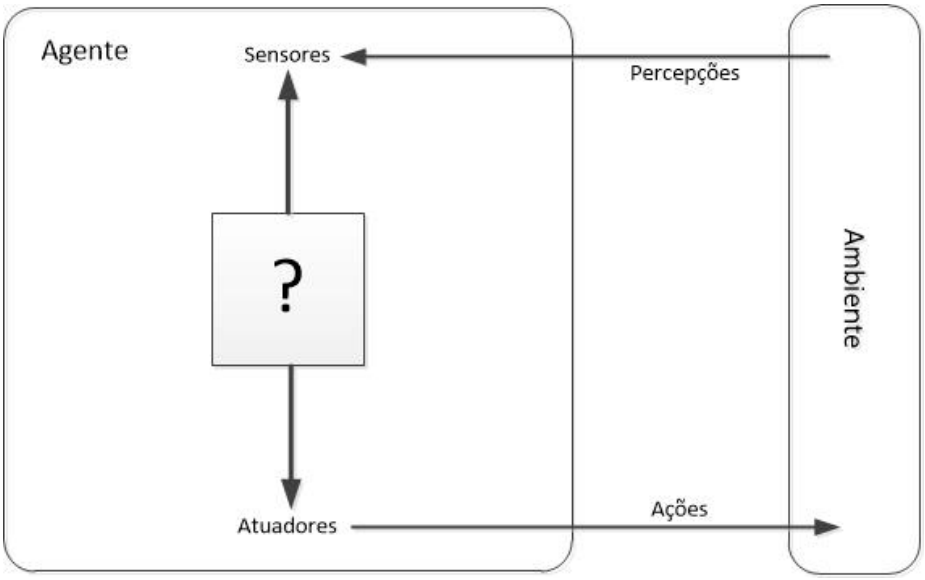


Figura 16: Agentes interagem com ambientes por meio de sensores e atuadores (RUSSEL e NORVIG, 2004).

Autonomia, capacidade de aprendizado e sociabilidade são as características em destaque de um agente de software, sendo a autonomia a principal característica que os difere de aplicações tradicionais, como por exemplo, sistemas especialistas (GIRARDI, 2001).

A autonomia permite que o agente possua controle sobre seu próprio comportamento, sendo um objeto ativo, capaz de agir a partir das percepções do ambiente e com base em seu conhecimento prévio ou adquirido a partir das percepções anteriores.

Um agente pode ser capaz ainda de ter a capacidade de aprendizado, sendo esta capacidade realizada com base nas observações do ambiente, atualizando seu estado e armazenando conhecimento adquirido das percepções e ações realizadas.

Um agente de software também é capaz de se relacionar com outros agentes de software em tempo de execução tanto para realizar seus objetivos quanto para ajudar outros agentes de uma sociedade de agentes.

O paradigma de agentes proporciona mecanismos como a decomposição, abstração e interações flexíveis, oferecendo, assim, excelentes meios para o desenvolvimento de sistemas complexos (GIRARDI, 2001).

Segundo Russel e Norvig (2004), os agentes podem ser classificados em: agentes reativos simples, agentes reativos baseados em modelos, agentes baseados em objetivos e agentes baseados na utilidade.

As arquiteturas de agentes reativos têm grande utilidade em casos em que há necessidade de uma ação rápida e imediata para uma determinada percepção no ambiente, sendo esta a sua principal vantagem (LEITE, GIRARDI, NOVAIS, 2013).

Agentes reativos simples não possuem capacidade de aprendizado, apenas reagem a partir de regras estáticas pré-definidas e a característica de sociabilidade é aplicável apenas quando um agente faz parte de uma sociedade de agentes. Através dos sensores obtém-se do ambiente a aparência atual do mundo, em seguida, a partir dessa observação e das regras condição-ação do agente, é obtida a ação correspondente ao estado atual que deve ser executada pelos atuadores (RUSSEL e NORVIG, 2004), conforme ilustrado na Figura 17**.**

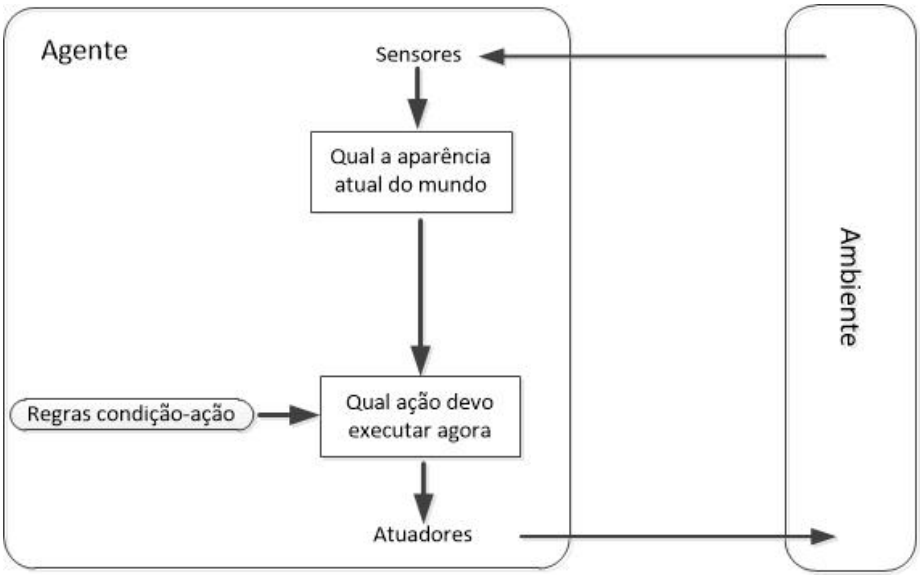


Figura 17: Esquema de um agente reativo simples (RUSSEL e NORVIG, 2004).

Agentes reativos baseados em modelos mantêm um estado interno dependente do histórico de percepções para que possa refletir alguns aspectos não observados do estado atual. Este tipo de agente controla o estado atual do mundo utilizando um modelo interno. Esse conhecimento é chamado de “modelo do mundo” (RUSSEL e NORVIG, 2004). Em seguida, ele escolhe a ação de maneira similar ao agente reativo simples, isto é, em uma lista de regras condição-ação pré-definidas conforme ilustrado na Figura 18.

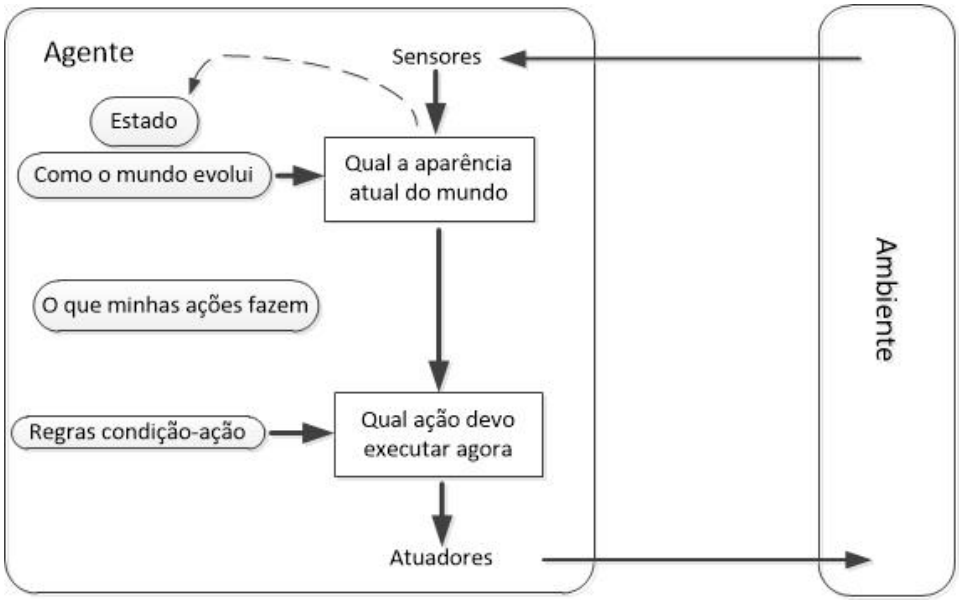


Figura 18: Esquema de um agente reativo baseado em modelos (RUSSEL e NORVIG, 2004).

Agentes deliberativos são aqueles que possuem um modelo de raciocínio (GIRARDI, 2001). Eles têm a capacidade de construção de planos de interação com o ambiente a fim de modificá-los. Possuem símbolos e rótulos que permitem que haja uma representação do mundo com o objetivo de planejar e decidir através de algum método de inferência as ações a serem executadas (NIU e HU, 2012).

Sendo assim, dentre os agentes da classificação de Russel e Norvig (2004), podemos considerar como deliberativos os agentes baseados em objetivos e os baseados em utilidade, visto que possuem um modelo de raciocínio.

Os agentes deliberativos baseados em objetivos, assim como os reativos baseados em modelo também conhecem o estado atual do ambiente, mas isso nem sempre é o suficiente para decidir o que fazer. Esse tipo de agente possui ainda informações sobre objetivos, isto é, situações desejadas, de forma que, combinando com informações de resultados de ações possíveis, possa escolher aquelas que possam alcançar o objetivo. Como pode ser observado na Figura 19, esse tipo de agente, além de controlar o estado do mundo, controla também um conjunto de objetivos que está tentando atingir, e é baseado nesse conjunto de informações que a ação é escolhida.

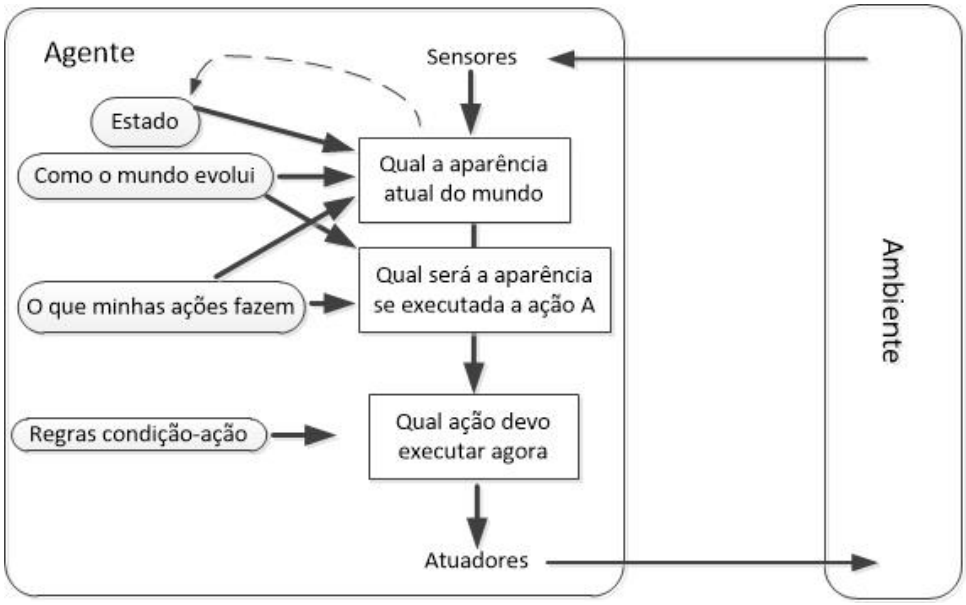


Figura 19: Esquema de um agente baseado em modelo e orientado para objetivos (RUSSEL e NORVIG, 2004).

Os agentes deliberativos baseados na utilidade possuem as mesmas características que a arquitetura de agentes orientada em objetivos, porém, para obter uma melhor qualidade do comportamento, utilizam uma função de utilidade, que é o mapeamento de um estado em um número real, que descreve o grau de utilidade associado. Baseado nessa função de utilidade, a ação a ser tomada é escolhida levando à melhor utilidade esperada (RUSSEL e NORVIG, 2004), conforme ilustrado na Figura 20.

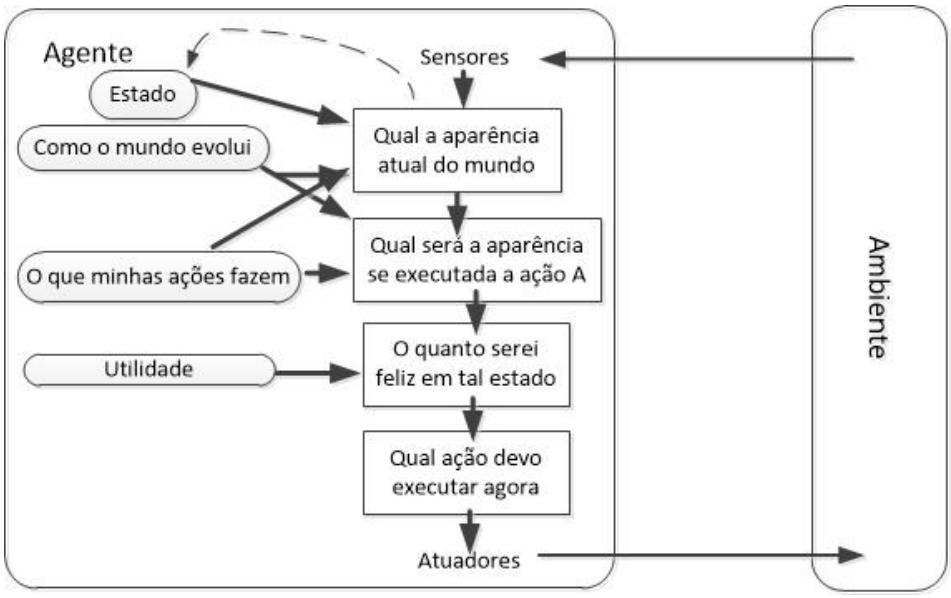


Figura 20: Esquema de um agente baseado em modelo e orientado para a utilidade (RUSSEL e NORVIG, 2004).

A arquitetura de um agente RBC como a proposta neste trabalho é considerada uma arquitetura de um agente deliberativo, visto que é um agente capaz de planejar e executar suas ações através do paradigma RBC. Um agente RBC é capaz de deliberar acerca das percepções obtidas e ações executadas realizando um processo de inferência com base em experiências anteriores que compõem a base de conhecimento do mesmo.

Existem ainda arquiteturas híbridas de agentes que unem as vantagens das arquiteturas reativas e deliberativas aumentando a flexibilidade do agente (LEITE, GIRARDI e NOVAIS, 2013). Este tipo de arquitetura proporciona uma reposta reativa mais rápida a eventos importantes mas também a capacidade de deliberar em outros tipos de eventos que exigem um processo de raciocínio (NIU e HU, 2012).

2.5 Estado da arte nas abordagens RBC

Vários trabalhos relacionados ao estudo do RBC têm sido apresentados na literatura. Alguns procuram definir os mecanismos, tarefas e o ciclo de vida de um sistema RBC (AAMODT e PLAZA, 1994).

Outros trabalhos relacionam esse paradigma de raciocínio à tecnologias, como sistemas multiagentes e/ou ontologias (CORCHADO e LAZA, 2012) (LESS e CORCHADO, 1997) (GUAN-YU, LI-NING e SHI-PEN, 2008), (SRINIVASAN et al, 2011) (GARRIDO et al., 2008).

O trabalho de Corchado e Laza (2012) apresentou um estudo sobre agentes BDI (Belief, Desire and Intention) que relacionam os conceitos do RBC, denominados agentes RBC-BDI. Less e Corchado (1997) apresentam a proposta de um sistema de aprendizado híbrido com RBC e Redes Neurais Artificiais (RNA). Uma framework, denominada CBOR, e que une o RBC com ontologias foi elaborada por Guan-Yu, Li-Ning e Shi-Pen (2008). O trabalho de Srinivasan et al. (2011) propõe a arquitetura de um sistema de suporte à decisão baseado em sistemas multiagentes utilizando mineração de dados e RBC. Garrido et al. (2008) propõem uma abordagem RBC baseada em ontologias para a recomendação e reúso de conhecimento compartilhado na tomada de decisões.

Nas seções seguintes serão descritas essas abordagens e feito um comparativo com a arquitetura CBRAA em relação às características referentes ao

uso de agentes, ontologias e modelo de similaridade utilizado para recuperação de casos.

2.5.1 Agentes RBC-BDI

Corchado e Laza (2012) propõem a criação de agentes deliberativos BDI (WOOLDRIDGE, 2009), que relacionam os componentes de um agente BDI às funções de um sistema RBC. A arquitetura de agentes BDI é uma visualização do sistema como agente racional com atitudes mentais de crença, desejo e intenção, que representam, respectivamente, a informação, motivacionais e estados deliberativos do agente (RAO e GEORGEFF, 1991).

Não há especificação do uso de ontologias na arquitetura, e o mecanismo de recuperação de casos é apoiado pelo uso de um método de núcleo que utiliza algoritmos para identificação de padrões (FYFE e CORCHADO, 2001) para recuperação de casos similares.

A proposta do trabalho é a de uma metodologia que fosse capaz de automatizar a criação desses agentes além de prover a eles capacidade de aprendizado e autonomia através dos conceitos do BDI. A utilização dos conceitos do RBC no trabalho tem como objetivo resolver o problema de aprendizado de um agente BDI. O relacionamento entre os conceitos de BDI e RBC se dá pela implementação das intenções como casos, em uma sequência ordenada de ações e estados.

A arquitetura de um sistema baseado em agentes para suporte a venda foi descrita com a utilização do agente RBC-BDI como agente de Planejamento ilustrado na Figura 21. Os agentes de planejamento geram planos de trabalhos com base seus sistemas RBC incorporados. Ele é responsável por estimar os custos, pessoal e material requerido para a construção de um projeto. O agente de busca na internet é um motor de busca que procura por clientes em potencial e informações sobre eles. Os agentes assistentes fazem o papel dos vendedores. São a interface entre o usuário e o agente de planejamento. Há ainda o agente de administração que é um agente de interface que facilita a interação entre usuários e o resto dos elementos do sistema.

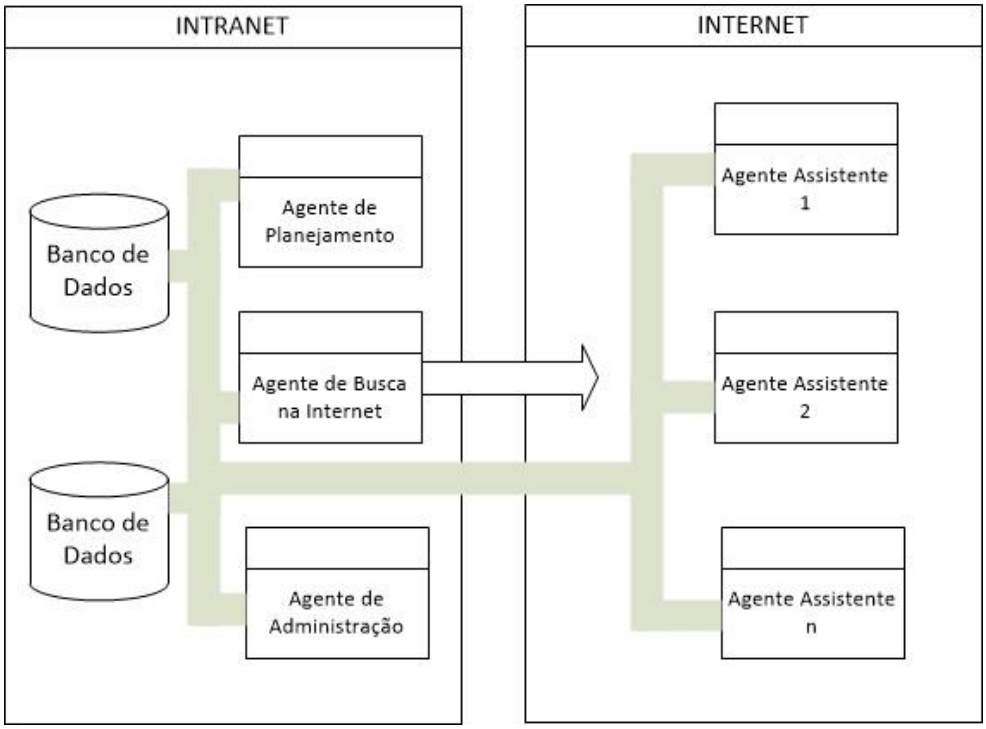


Figura 21: Arquitetura para comércio eletrônico orientada a agentes (CORCHADO e LAZA, 2012).

2.5.2 Sistema de Aprendizado Híbrido com RBC e RNA

O trabalho de Less e Corchado (1997) por sua vez propõe um sistema de aprendizado híbrido que combina RBC e Redes Neurais Artificiais (RNA) em uma arquitetura orientada a objetos. Nesse trabalho o RBC e RNA são utilizados com objetivos diferentes nos diversos estágios da aprendizagem e previsão, por meio de uma arquitetura de um agente adaptativo, isto é, um agente capaz de melhorar seu desempenho com o passar do tempo através de RNA. Um agente adaptativo adapta continuamente sua organização para alcançar as variações do ambiente (GUESSOUM, 2004), o que o torna bastante apropriado quando em ambientes complexos, onde mudanças de restrição são imprevisíveis. A recuperação e a identificação dos casos similares são realizadas através de RNA, buscando padrões de similaridade entre os casos.

A integração de RNA e RBC tem sido estudada por um número pequeno de pesquisadores (KROVVIDY e WEE, 1993) (KOCK, 1996) (LESS e CORCHADO, 1997). O RBC é efetivo na resolução de problemas devido à sua capacidade de lembrar de casos, no entanto não possui capacidade de generalização, a qual pode ser implementada através de uma RNA.

Segundo Reategui e Campbell (1994), é possível integrar RBC com outros métodos de raciocínio de quatro formas: controle central, em que um dispositivo central controla os diferentes mecanismos de raciocínio; controle distribuído, no qual cada mecanismo envolvido possui parte do controle; RBC dominante, em que o mecanismo RBC é o dominante e controla os outros mecanismos de acordo com a demanda; e RBC não dominante, cujo mecanismo RBC é controlado por outro mecanismo de raciocínio.

Redes neurais e RBC podem ser consideradas como técnicas complementares nesse trabalho. O RBC usa experiências passadas e possui capacidade de aprender, e RNA tem a capacidade de geração adaptativa de estruturas utilizando largo conjunto de dados. A rede neural de interesse dessa pesquisa é a *Radial Basis Function* (RBF). Em RBF a camada de entrada é um receptor de entrada de dados, enquanto que a camada oculta realiza uma transformação não linear dos dados inseridos. Os neurônios ocultos formam uma base para vetores de entrada; os neurônios de saída somente calculam a combinação linear das saídas dos neurônios ocultos (BISHOP, 1995).

Uma vez que o número de casos se torne grande, o agente utilizará os casos que tenham sido mais frequentemente explorados para treinar a RNA.

2.5.3 CBOR

Outro trabalho estudado foi o de Guan-Yu, Li-Ning e Shi-Peng (2008), no qual foi proposto o Raciocínio Baseado em Casos e Ontologias, que combina a utilização de ontologias para representação de conhecimento com RBC, porém o trabalho não envolve o uso de agentes. O trabalho especifica um framework que foi denominado *Case-Based Ontology Reasoning* (CBOR), que abrange os mecanismos necessários para inferência através do RBC, com o apoio de ontologias para a representação da base de casos. No CBOR a recuperação de casos utiliza o método de busca do vizinho mais próximo (PATTERSON, ROONEY e GALUSHKA, 2002) para medir a distância entre os casos. O valor de similaridade entre os conceitos é obtido através da comparação baseada em semântica. No entanto, não foi especificada a medida de similaridade para o cálculo.

Basicamente esse trabalho propôs a utilização das ontologias para representação dos casos. A arquitetura CBOR, ilustrada na Figura 22 é composta de

cinco etapas: Descrição dos Casos, Indexação dos Casos, Recuperação de Casos, Modificação dos Casos e Manutenção dos Casos, bem similar à arquitetura RBC tradicional (WANGENHEIM e WANGENHEIM, 2003).

A interface possui comunicação com o motor do CBOR que interage com as etapas de recuperação de casos (entradas) e descrição de casos (saídas). Além da base casos está presenta uma biblioteca de ontologias do domínio utilizada pelo mecanismo de representação de casos, obtida a partir de documentos de informações de domínio. Na figura e no trabalho, ilustradas e exemplificadas através informações médicas.

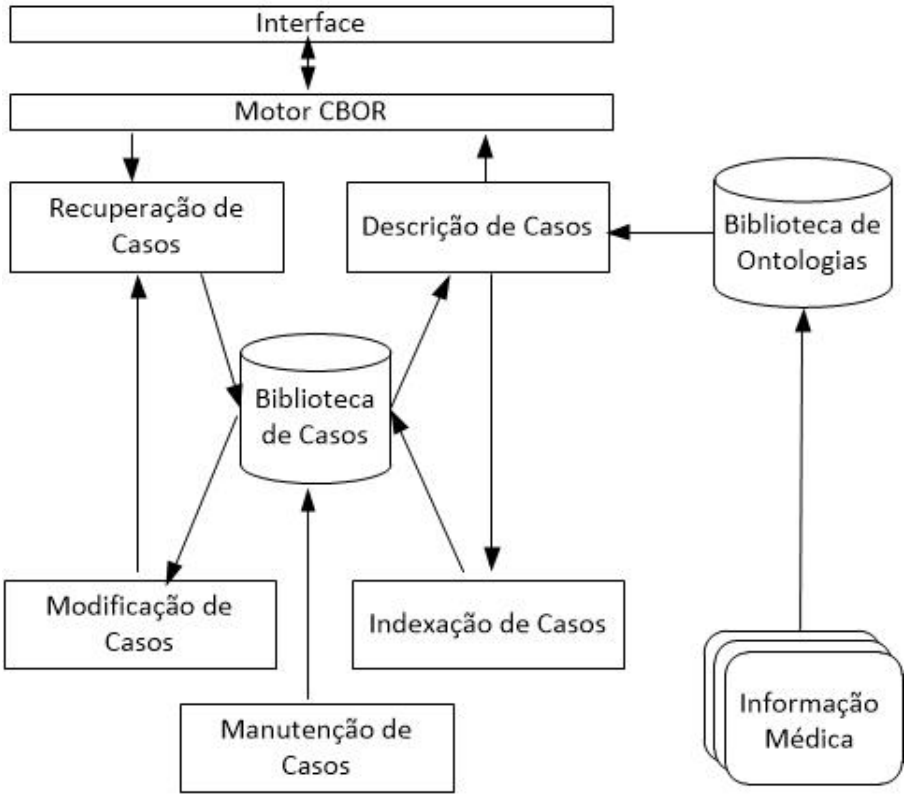


Figura 22: Arquitetura do CBOR (GUAN-YU, LI-NING, SHI-PENG, 2008).

A descrição dos casos é a base e o núcleo do modelo. É composta de três componentes: problemas e descrição dos cenários, soluções e resultados.

A descrição detalhada de um caso nessa solução é composta por: Código do Caso, Nome do Caso (para distinguir os diferentes casos), Tipo do caso (para indexação), Propriedades e Decisão (o que foi feito para resolver aquele caso).

2.5.4 Sistema de Suporte à Decisão baseado em sistemas multiagentes utilizando Mineração de Dados e RBC

No trabalho de Srinivasan et al. (2011) propôs-se a construção de um modelo que integrasse em um sistema de apoio à decisões, técnicas de mineração de dados, RBC e sistemas multiagente.

Sistemas de Apoio à decisão são programas computadorizados que auxiliam em um ambiente de tomada de decisões ou apenas auxiliam na resolução de um problema (SRINIVASAN et al, 2011).

Técnicas de mineração de dados são comumente utilizadas por estatísticos, analistas de dados e sistemas de gerenciamento de informação. Em termos genéricos, é a aplicação de algoritmos específicos para a extração de padrões dos dados (FAYYAD, PIATETSKY-SHAPIRO, SMYTH, 1996).

O modelo desse trabalho (Figura 23), possui um agente de interface do usuário para receber as requisições e então repassa ao agente coordenador do sistema de apoio à decisões, que por sua vez repassa ao gerenciador de diálogo que vai interpretar e passar a requisição ao sistema de mineração de dados ou ao sistema RBC.

O sistema de mineração de dados do modelo é utilizado quando há necessidade de identificação de padrões nos dados que sejam interessantes e válidos. É direcionado para ajudar os gerentes através de decisões estruturadas ou semiestruturadas de processos de tomadas de decisão. O sistema RBC, por sua vez, é utilizado para a resolução prioritária de problemas através de soluções previamente armazenadas em uma base de casos específicos de um domínio.

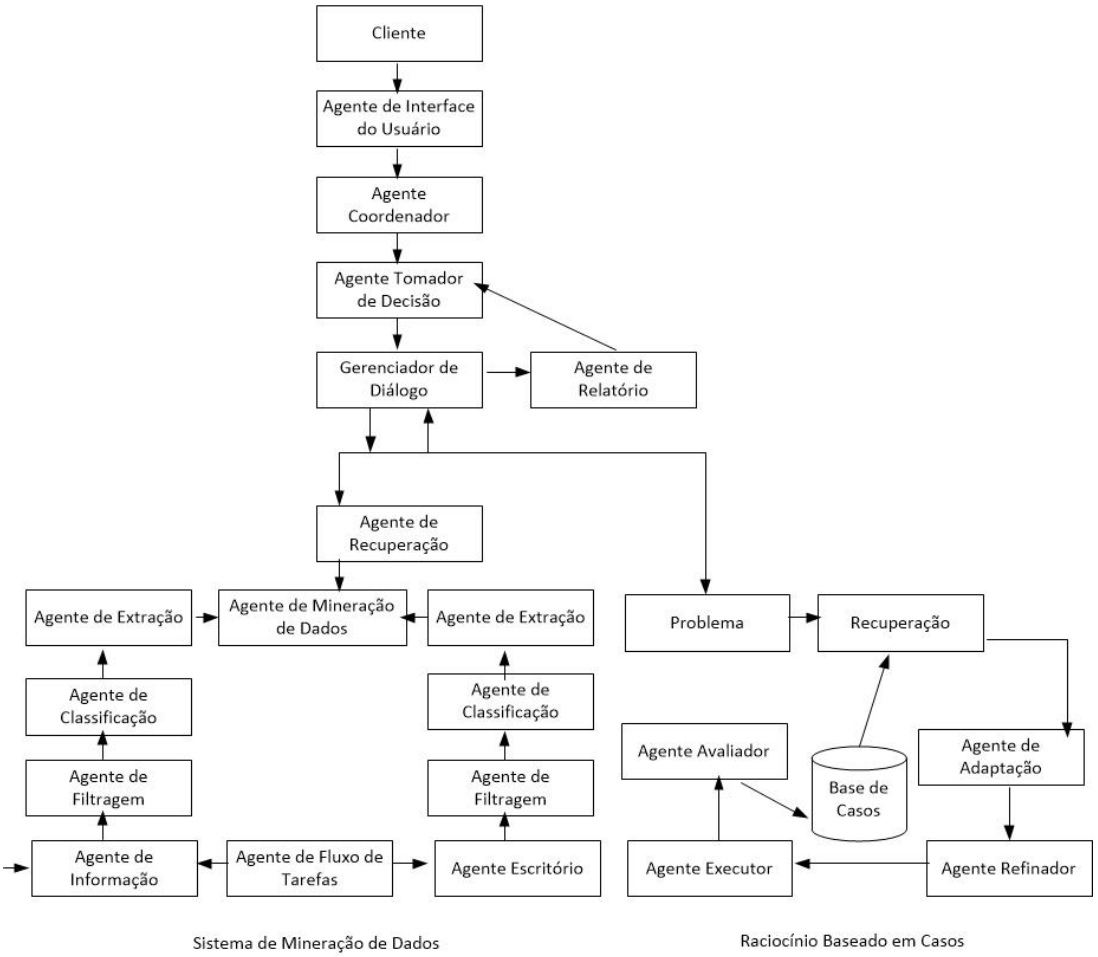


Figura 23: Sistema de Suporte à Decisão baseado em sistemas multiagentes utilizando Mineração de Dados e RBC (SRINIVASAN et al., 2011).

O sistema multiagente proposto possui agentes para cada atividade da tomada de decisões, atividades de mineração de dados e atividades do sistema RBC.

2.5.5 Uma abordagem RBC baseada em ontologias para a recomendação e reúso de conhecimento compartilhado na tomada de decisões

O trabalho de Garrido et al. (2008) é a arquitetura de um sistema RBC distribuído, baseado em ambiente multiagente. Consiste em uma arquitetura de um sistema RBC distribuído, direcionada para a reutilização de conhecimento e que se baseia em um ambiente multiagente para acesso baseado em semântica ao conhecimento.

Garrido et al. (2008) defende a utilização de uma abordagem RBC baseada em ontologias para o reúso de conhecimento compartilhado na tomada de decisões. A proposta desse trabalho foi direcionada para a reutilização de conhecimento,

durante atividades de decisão, traçando conceitos de RBC, gerência do conhecimento e ontologias.

Nessa arquitetura (Figura 24) várias organizações utilizam base de conhecimento implementada através de vários RBCs para o mesmo domínio de problema. Para isso, devem utilizar vocabulário similar e precisam cooperar, compartilhando conhecimento no intuito de melhorar o sistema de decisão. Cada agente deve combinar capacidades gerais de um agente de software com as capacidades presentes em um sistema RBC (busca, adaptação, raciocínio e aprendizado). A arquitetura é composta de dois tipos de agentes: agentes usuários e agentes RBC. Um agente usuário está no comando da manutenção da interação sobre as requisições de um usuário específico. São o *front-end* que fornecem a independência necessária do usuário em relação ao detalhamento interno do sistema. Fazem a comunicação entre usuários e base de conhecimento de forma transparente: possuem a localização dos agentes RBC, direcionando as requisições aos agentes RBC mais adequados; dão suporte a tolerância a falhas na comunicação e processamento, como por exemplo, redirecionando a requisição em caso de falha para outro agente RBC; distribuem as requisições para balanceamento de tarefas para melhor desempenho. Os agentes RBC, por sua vez, executam diferentes papéis nas quatro etapas do ciclo RBC e podem cooperar entre si através da troca de conhecimento (GARRIDO et al., 2008).

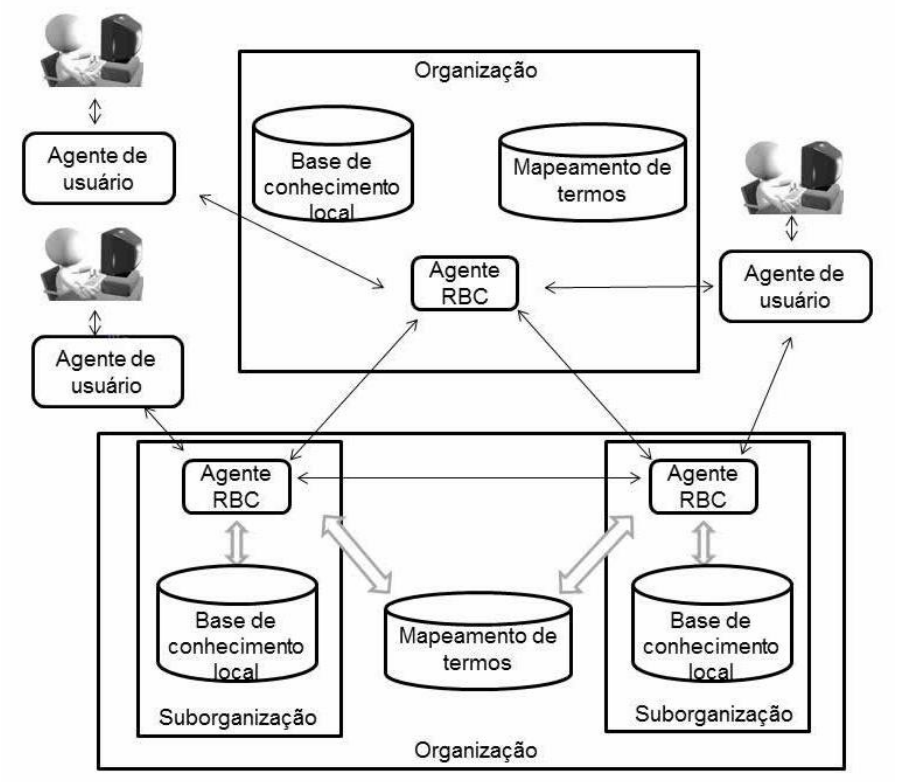


Figura 24: Arquitetura RBC baseada em ontologias para a recomendação e reúso de conhecimento compartilhado na tomada de decisões (GARRIDO et al., 2008)

É definida no trabalho como ocorre a etapa de recuperação de casos através de um modelo de similaridade semântica, e também as estratégias que poderiam ser utilizadas pelos mecanismos de adaptação e aprendizado de casos, baseados na representação em ontologia.

O modelo de similaridade semântica utilizado está definido na Equação 1.

$$Sim(C_C, C_M) = \sum_{i=1}^{n} \frac{w_i * sim(f_i^c, f_i^m)}{\sum w_i} \quad (1)$$

onde:

- C_C é o caso atual.
- C_M é o caso da memória de casos.
- w_i é o peso do atributo i.
- f_i é o valor do atributo i.
- sim(a,b) é a medida de similaridade dos atributos.

A função sim baseia-se no nível da hierarquia da ontologia entre os valores dos atributos dos casos comparados, e pode ser definida como:

$$\text{sim}(a,b)= \begin{cases} 1 \text{ se } a=b \\ \frac{3}{4} \text{ se a difere 1 nível de b} \\ \frac{1}{2} \text{ se a difere 2 níveis de b} \\ 0 \text{ se não está na ontologia ou mais que 2 níveis de b} \end{cases}$$

2.5.6 Análise Comparativa

As características mais relevantes destes trabalhos estão relacionadas na Tabela 1 com objetivo de comparar estes trabalhos com a arquitetura CBRAA proposta. A comparação foi realizada em relação às características correspondentes ao uso exclusivo de RBC, abordagem de agentes, uso de ontologias e modelo para recuperação de casos.

Tabela 1: Comparativo das características de trabalhos do estado da arte nas abordagens RBC.

Trabalho	**Uso de RBC**	**Uso de Agentes**	**Uso de Ontologias**	**Modelo de Recuperação de Casos**
Corchado e Laza (2012)	Sim + BDI	Sim	Não	Método de núcleo (FYVE e CORCHADO, 2001).
Less e Corchado (1997)	Sim + Redes Neurais Artificiais	Sim	Não	Identificação de Padrões através de Redes Neurais Artificiais.
Guan-Yu, Li-Ning e Shi-Peng (2008)	Sim	Não	Sim	Método de vizinho mais próximo baseado em semântica (PATTERSON, ROONEY e GALUSHKA, 2002).
Srinivasan et al. (2011)	Sim + Mineração de Dados	Sim	Não	Não especificado
Garrido et al. (2008)	Sim	Sim	Sim	Modelo Semântico (GARRIDO et al., 2008)

Os trabalhos de Corchado e Laza (2012), Guan-Yu, Ni-Ling e Shi-Peng (2008), Less e Corchado (1997), Srinivasan et al. (2011) e Garrido et al. (2008) foram contribuições importantes na área de estudo do RBC.

O principal diferencial da arquitetura CBRAA em relação aos trabalhos definidos em Less e Corchado (1997), Corchado e Laza (2012) e Srinivasan et al. (2011) é que a mesma possui suporte a ontologias para representação dos casos,

obtendo com isso as vantagens de uma representação semântica, com a facilidade de reutilização, adaptação e seleção de casos. Existem outras diferenças conceituais tais como a utilização de diferentes modelos de recuperação de casos e associação de RBC com BDI (CORCHADO e LAZA, 2012).

Já a arquitetura RBC definida em Guan-Yu, Li-Ning e Shi-Peng (2008) tem como principal diferença em relação à CBRAA a não utilização de agentes.

Por fim, a arquitetura CBRAA se diferencia do trabalho proposto em Garrido et al. (2008) pelo modelo de recuperação de casos e pela forma como as etapas do ciclo de raciocínio do RBC são distribuídas na arquitetura. Em CBRAA todos os componentes e mecanismos RBC estão inclusos na arquitetura do agente, que executa todas as etapas do RBC, já em Garrido et al. (2008) são executadas por diversos agentes que alternam os papéis executados durante o raciocínio.

2.6 Considerações Finais

Foram abordados neste capítulo os principais conceitos utilizados para a proposta da arquitetura do agente RBC deste trabalho: o conceito de raciocínio e seus tipos, em particular o RBC, que é considerado um tipo de raciocínio analógico voltado para resolução de problemas dentro de um domínio específico; conceitos básicos de ontologias, considerando-se que foi a opção utilizada para a representação da base de conhecimento da arquitetura; conceitos e classificação dos agentes de software, com destaque para os agentes deliberativos, considerando que um agente capaz de inferir sobre dados de um problema através de um paradigma de raciocínio como o RBC é um agente deliberativo.

Foi realizado, ainda, um estudo do estado da arte em algumas abordagens RBC e comparadas com a arquitetura CBRAA. Seus conceitos básicos e algumas arquiteturas foram ilustradas e descritas para fins comparativos. Os trabalhos de Guan-Yu, Li-Ning e Shi-Peng (2008) e Garrido et al. (2008) foram os que mais se aproximaram da nossa pesquisa, contudo, possuem algumas características que as diferenciam do nosso trabalho como a abordagem de agentes e distribuição das atividades do RBC.

3. A ARQUITETURA CBRAA

Neste capítulo é apresentada a proposta de uma arquitetura de um agente RBC que engloba as características de um agente deliberativo e o paradigma de resolução de problemas RBC.

3.1 Definição da arquitetura

A arquitetura proposta, denominada CBRAA ("*Case-based Reasoning Agent Architecture*"), foi elaborada de forma que o agente deliberativo tenha em sua composição todos os mecanismos de um sistema RBC (WANGENHEIM e WANGENHEIN, 2003): representação de casos, recuperação de casos e análise de similaridade, adaptação e aprendizado.

Um caso RBC consiste em duas partes básicas: uma correspondente à descrição de um problema utilizada para localização de problemas similares, e a outra especificando a solução para o problema.

A Figura 25 mostra a interação entre os principais componentes de um sistema RBC. A entrada do processo é uma consulta que consiste em um novo caso-problema a ser solucionado. Então, o mecanismo de representação de casos constrói uma representação interna do novo caso, o qual é comparado com casos da base de casos do sistema RBC para identificar similaridades. Em seguida, uma solução apropriada para o novo caso-problema é obtida a partir dos casos similares recuperados. Essa solução será eventualmente customizada através do mecanismo de adaptação para ser reutilizada. Caso essa solução seja adequada para o problema, o mecanismo de aprendizado irá avaliar a possibilidade de incluir o novo caso solucionado na base de casos.

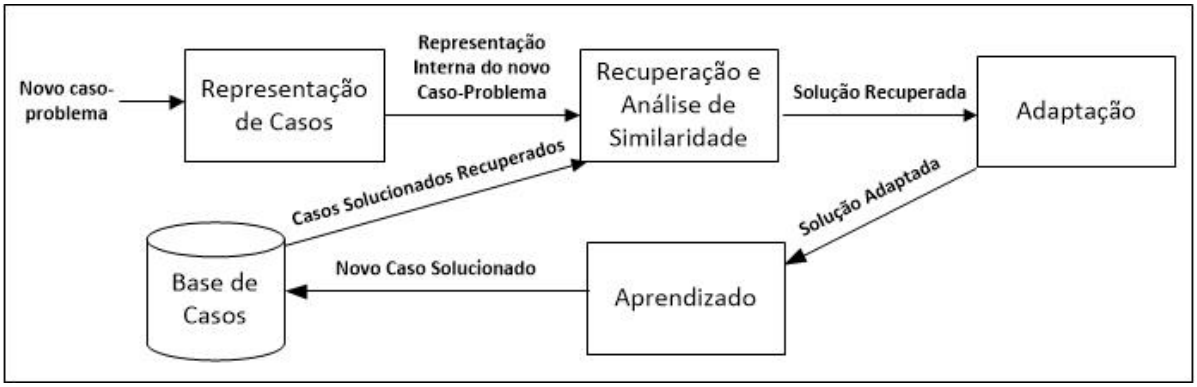


Figura 25: Principais componentes de um sistema RBC e suas interações.

O agente RBC proposto está baseado em uma composição dos mecanismos de um sistema RBC com o agente deliberativo baseado em modelos e orientado para utilidade (RUSSEL e NORVIG, 2004).

O objetivo do agente RBC é propor soluções para um novo caso-problema através do RBC. Para isso, o agente realiza um processo de inferência procurando as ações mais apropriadas para cada nova percepção baseando-se em casos solucionados e conhecimento de domínio. Os casos solucionados utilizados são aqueles selecionados que possuem maior similaridade na definição do problema em relação ao novo caso-problema, e sua solução em seguida deve ser recuperada e eventualmente adaptada, oferecendo, assim, a ação mais apropriada dada uma nova percepção do agente.

A Figura 26 ilustra a arquitetura do agente CBRAA proposta nesse trabalho, composta pelos quatro mecanismos de um sistema RBC e das interações que ocorrem entre eles. A base de conhecimento do agente é constituída de casos, conhecimento do domínio e regras de adaptação. Através de sensores o agente é capaz de perceber eventos do ambiente, e então identificar novos casos-problema a serem solucionados. Em seguida, uma representação interna da percepção correspondente ao novo caso-problema é construída em instâncias da ontologia de casos pelo mecanismo de representação de casos. Essa representação interna é então utilizada para comparar e mensurar a similaridade com casos recuperados da base de casos. Para isso, é utilizado pelo mecanismo de recuperação e análise de similaridade, um modelo de similaridade semântico (SILVA, GIRARDI E DRUMMOND, 2009) que permite identificar casos recuperados da base de conhecimento mais similares ao novo caso-problema. Em seguida, o agente de software deverá ser capaz de selecionar o caso ou conjunto de casos com maior similaridade para reúso da

solução após uma eventual adaptação. Essa adaptação utilizará um conjunto de regras de adaptação e conhecimento de domínio, e será realizada pelo mecanismo de adaptação automaticamente. Finalmente, a solução recuperada é executada no ambiente através dos atuadores.

Contudo, para que o processo RBC seja concluído é necessário aprender o novo caso solucionado. Para isso, o agente de software deverá ser capaz de perceber o impacto dessa solução no ambiente, isto é, avaliar se o problema foi resolvido com a solução aplicada. Através dos sensores, o agente é ser capaz de obter uma nova percepção acerca dos efeitos da ação da solução executada no ambiente e realizar uma avaliação, e, em caso de sucesso adicionar o novo caso solucionado à base de conhecimento do agente através do mecanismo de aprendizado.

A base de conhecimento do agente CBRAA consiste em casos solucionados, regras de adaptação para as soluções e conhecimento de domínio, que é utilizado pelos mecanismos de recuperação de casos e análise de similaridade, e de adaptação.

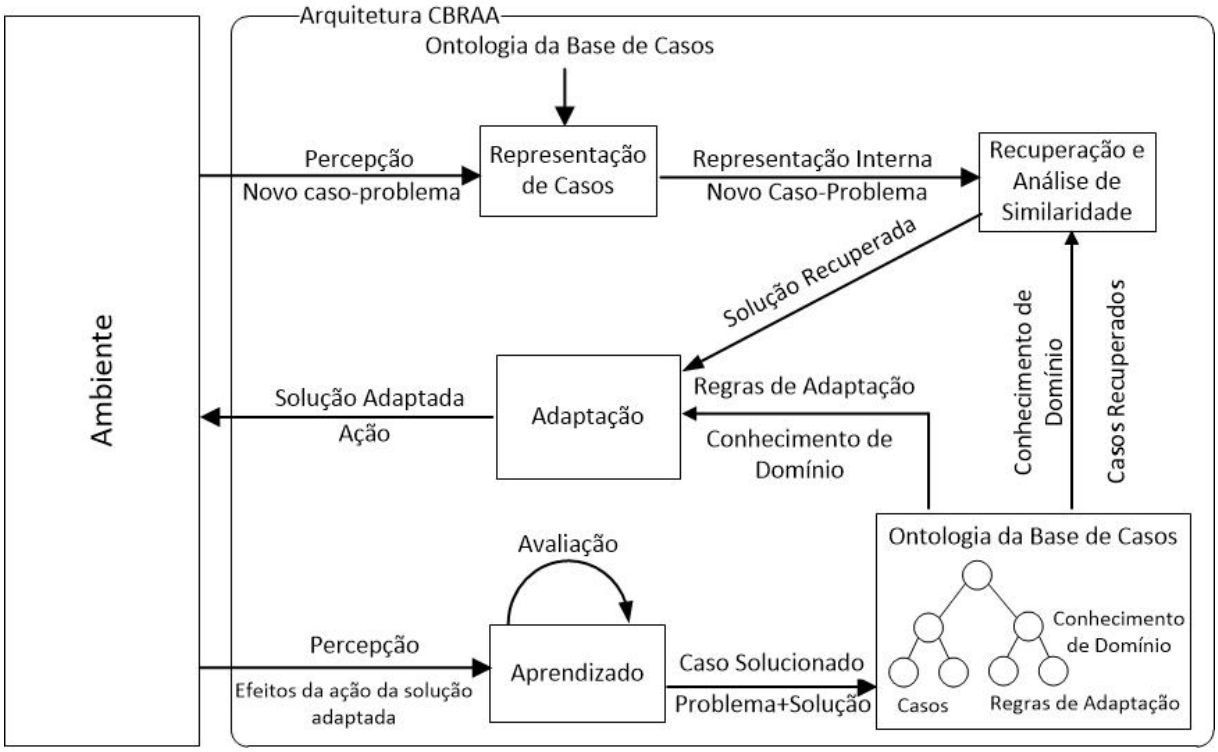


Figura 26: Arquitetura CBRAA

É essencial que uma base inicial de casos solucionados seja criada, podendo essa ser formada a partir de processamento da linguagem natural (ALLEN, 1995) ou mapeamento manual de documentos textuais referentes a casos do domínio, pois todo o processo de inferência do RBC depende da existência de casos similares para reúso.

Os componentes da arquitetura que representam os mecanismos do sistema RBC inclusos na arquitetura do agente serão especificados a seguir.

3.2 Componentes da arquitetura

3.2.1 Mecanismo de Representação de Casos

A representação interna de um novo caso-problema é criada através da instanciação da ontologia baseada em casos. A seguir há uma descrição das principais classes da ontologia e instâncias de exemplos do domínio jurídico do Direito de Família brasileiro.

A principal classe da ontologia, base de conhecimento do agente RBC, é a classe *Case*, estruturada de acordo com as duas partes de um caso RBC: o problema e sua solução, representados na ontologia pelas classes *Problem* e *Solution*, e os relacionamentos não-taxonômicos *hasProblem* e *hasSolution* (Figura 27).

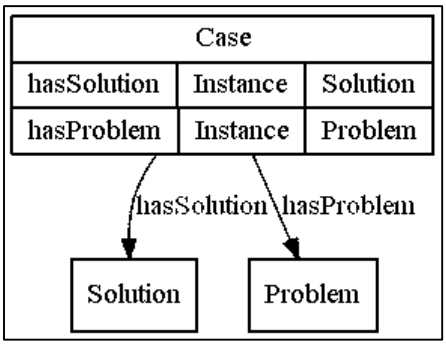


Figura 27: Ontologia de Casos simplificada.

Um sistema RBC resolve problemas a partir de soluções de problemas do mesmo domínio. Para tanto, é necessário que haja propriedades e relacionamentos nas classes que representem os atributos que compõem o problema e a solução de um caso.

A percepção de um novo caso-problema jurídico pode ser obtida a partir de um documento em linguagem natural, uma lista atributo-valor estruturada ou através de uma interação do advogado com apenas um ou ambos os consortes. O agente RBC deve ser capaz de extrair informações dessa percepção e então criar uma representação interna do novo caso-problema para que possa realizar o cálculo da análise de similaridade com os casos recuperados da base de conhecimento do agente.

LegalCaseProblem, uma subclasse de *Problem*, é a classe principal da ontologia que representa casos jurídicos do Direito de Família brasileiro. Instâncias dessa classe serão utilizadas pelo mecanismo de recuperação de casos para medir o valor de similaridade em relação a um novo caso-problema. A Figura 28 descreve classes correspondentes aos principais casos-problema jurídicos do Direito de Família brasileiro: Divórcio, Adoção e Casamento, através das classes *DivorceProblem*, *AdoptionProblem* e *MarriageProblem*. Semelhantemente, a figura mostra a hierarquia de classes correspondente à solução: *DivorceSolution*, *AdoptionSolution* e *MarriageSolution*.

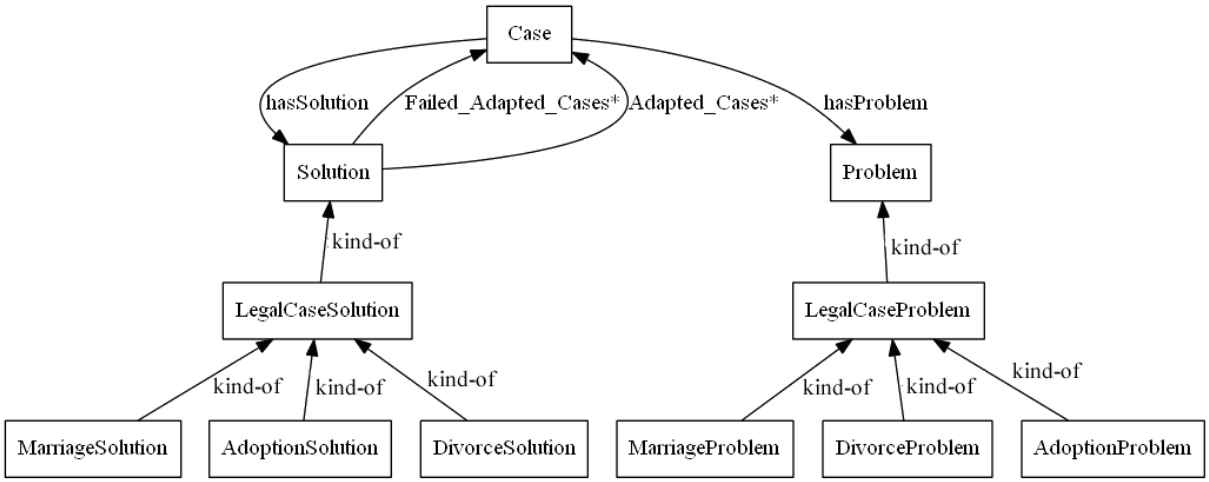


Figura 28: Principais classes da ontologia da base de conhecimento do agente CBRAA representando casos RBC do Direito de Família brasileiro.

A Figura 29 mostra a representação de classes de um caso de divórcio que representa seu problema e solução. Dentre os atributos de um problema de divórcio estão o regime de casamento, representado pela propriedade *matrimonialRegime*, na classe *DivorceProblem* e informação sobre a existência de requerimento acerca de algum dos termos do divórcio, como a propriedade *IsThereAlimonyRequirement* que

diz respeito a existência de solicitação de pensão alimentícia. Esses atributos serão melhor explicados no capítulo da avaliação.

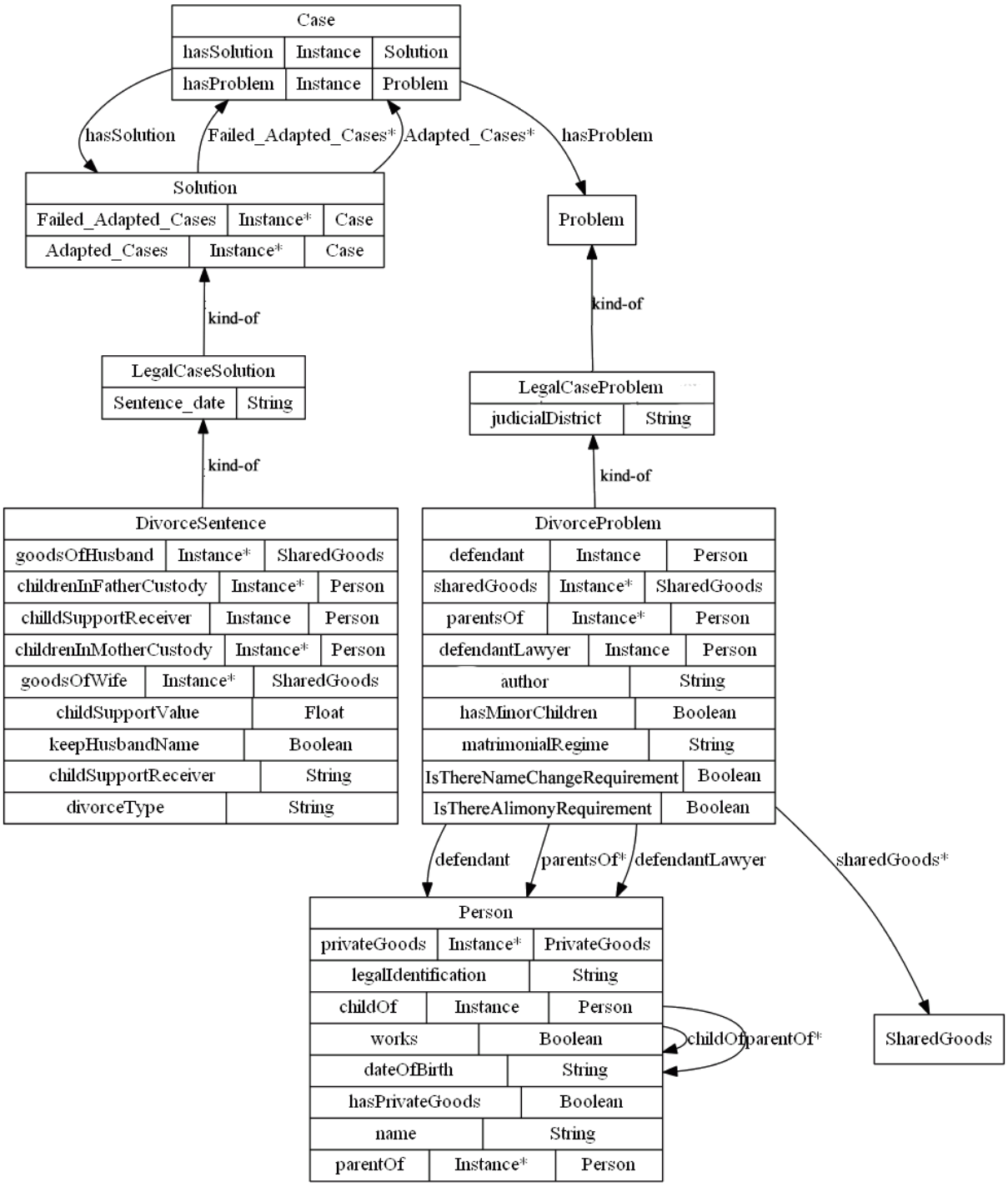


Figura 29: Classes da ontologia da base de conhecimento do agente CBRAA representando casos de divórcio.

Um exemplo de documento em linguagem natural utilizado para a elaboração das ontologias e instanciação do caso-problema é uma petição jurídica de

divórcio, um documento que descreve todo um caso jurídico de divórcio. Esse documento possui dados referentes às características de autores, partes, filhos, regime de casamento, dentro outras, conforme ilustrado em um trecho extraído de uma petição jurídica de divórcio representa na Figura 30.

EXMO (A) SR (A). JUIZ (A) DE DIREITO DA 2ª VARA DA COMARCA DE ROSÁRIO (MA).

Joao da Silva, brasileiro, maranhense, casado, professor, inscrito no RG sob nº1 e no CPF sob nº1, residente e domiciliado à Rua A e Maria da Silva, brasileira, maranhense, casada, do lar, inscrita no RG sob nº1 e no CPF sob nº1, residente e domiciliada à Rua A, cônjuges varão e varoa, respectivamente, pelo advogado e bastante procurador que constituíram (procuração em anexo), vêm, ajuizar **AÇÃO DE DIVÓRCIO CONSENSUAL**, com assentos nos fundamentos fáticos e jurídicos que doravante se expõe:

- I. DOS FATOS -

O casal contraiu matrimônio em 01 de janeiro de 2000, sendo lavrado o assento de matrimônio sob o Regime de Comunhão Parcial de Bens na 1ª Serventia Extrajudicial de Registro da Comarca de São Luís/MA, conforme se depreende da certidão apensa.

Dessa união adveio o nascimento dos menores Mariana Silva, nascida no dia 1 de outubro de 2001 (certidão de nascimento em anexo), e Joao Silva Filho, nascido no dia 1 de novembro de 2004 (certidão de nascimento em anexo).

Ocorre que, em razão das vicissitudes da vida, o matrimônio não mais subsiste plasmado pelo afeto necessário, de modo *more uxório*, pelo quê, em comum acordo, decidiram pôr fim ao vínculo conjugal, bem assim dispondo sobre a guarda dos filhos, alimentos, partilha de bens existentes e nome da cônjuge varoa.

- DA PARTILHA DE BENS –

O casal, na constância da união, não adquiriu bens suscetíveis de partilha. Os Requerentes não possuem dívidas a serem saldadas.

Figura 30: Trecho extraído de petição jurídica de divórcio.

A Figura 31 mostra a representação interna de um caso jurídico realizado de forma manual a partir de uma petição jurídica de divórcio em instâncias da ontologia.

As informações da petição são mapeadas para as propriedades correspondentes definidas na classe *DivorceProblem*. Para criar uma instância da classe *DivorceProblem* são identificados inicialmente quais dados serão representados através de instâncias na ontologia, por exemplo, a identificação de

todas as pessoas no documento como esposa, marido e filhos, e a criação de instâncias da classe *Person* para cada uma delas.

O procedimento é realizado para todas as informações representadas por instâncias. Em seguida, uma instância da classe *DivorceProblem* é construída com3 suas propriedades e relacionamentos não-taxonômicos associados às instâncias previamente criadas.

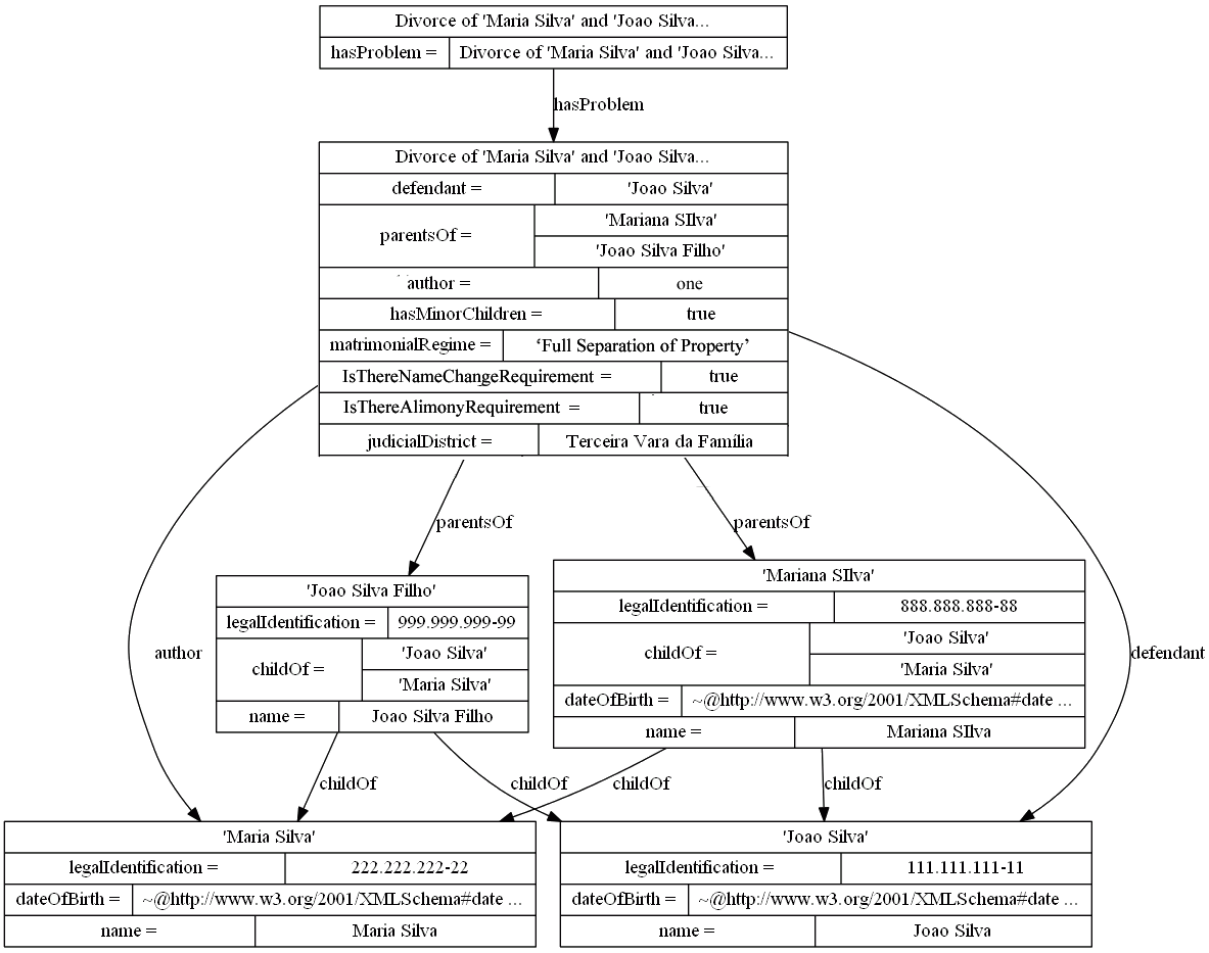


Figura 31: Parte da representação interna de um caso jurídico de divórcio em instâncias da ontologia de casos do agente.

3.2.2 Mecanismo de Recuperação e Análise de similaridade

Em um sistema RBC a informação está organizada em uma base de casos de onde o sistema deve ser capaz de recuperar os casos mais similares a uma nova consulta, isto é um novo caso-problema, o que na arquitetura proposta de um agente RBC corresponde a uma nova percepção deste. Esse procedimento corresponde a um processo de recuperação da informação, que no caso do sistema RBC

corresponde à obtenção de um caso similar com uma solução a ser adaptada. Para tanto, é necessário um modelo de recuperação de casos, que com uma medida de similaridade será capaz de calcular a relevância do caso recuperado em relação ao novo caso-problema.

Para uma melhor compreensão do processo de análise de similaridade utilizaremos um novo caso-problema jurídico de divórcio como exemplo. Considerando que o processo de divórcio pode envolver a disputa de guarda dos filhos e divisão de bens, estarão presentes na representação do novo caso-problema e no caso recuperado as mesmas propriedades que representam informações como as pessoas envolvidas, tais como marido, esposa e filhos, data, regime do casamento representados na Tabela 2. A coluna da esquerda ilustra em um conjunto de elementos do tipo atributo-valor a representação de uma nova percepção do agente RBC, isto é, um novo caso-problema, com suas propriedades e relações não taxonômicas e do lado direito um caso recuperado pelo mecanismo de análise de similaridade com o mesmo conjunto de propriedades e relações do novo caso-problema. Um conjunto de valores de atributos relevantes do novo caso-problema deve ser selecionado para o cálculo da medida de similaridade.

Como exemplo de uma possível análise de relevância da similaridade no exemplo ilustrado na Tabela 2, o fato do regime de casamento do caso recuperado ser igual ao da nova situação o torna mais similar em relação a outro caso de divórcio aonde o regime seja diferente, pois irá influenciar na separação de bens do casal. A informação a ser recuperada nesse processo de comparação e seleção de casos similares é a solução do problema a ser adaptada, mas a recuperação é realizada através do cálculo do valor de similaridade entre os problemas.

Tabela 2: Relação de atributos e valores de um novo caso-problema e um possível caso recuperado.

Percepção / Novo Caso-problema	**Caso Recuperado**
Problema: **Marido:** João Silva **Esposa:** Maria Silva **Data do Casamento:** 01/01/1980 **Regime de Casamento:** Comunhão Parcial **Autor(s):** Casal **Bens comuns:** Sim **Filhos Menores:** Sim	**Problema:** **Marido**: Alberto Limeira **Esposa**: Suzane Limeira **Data do Casamento**: 28/10/1975 **Regime de Casamento**: Comunhão Parcial **Autor(s)**: Marido **Bens comuns:** Sim **Filhos Menores:** Sim **Solução / Sentença Jurídico:** **Tipo de divórcio**: Judicial Consensual **Guarda dos Filhos:** Mãe

A recuperação de casos deve estabelecer o conjunto de atributos do problema a serem comparados e estabelecer os pesos que cada um irá possuir a fim de permitir a classificação dos casos recuperados de acordo com sua similaridade e consequente utilidade.

Em um sistema RBC a informação está organizada em uma base de casos de onde o mesmo deve ser capaz de recuperar os casos mais similares a uma consulta especificando um novo caso-problema. Na arquitetura proposta um novo caso-problema corresponde a uma nova percepção do agente CBRAA.

O principal objetivo do mecanismo de recuperação e análise de similaridade é identificar na base de casos o caso mais similar ao novo caso-problema para adaptar e reutilizar sua solução. Portanto, é necessário um modelo de recuperação de informação para comparar e recuperar casos solucionados, e através de um medida de similaridade calcular o valor de similaridade dos casos recuperados em relação ao novo caso-problema para reutilização da solução mais adequada. Para isso, o agente CBRAA utiliza um modelo de recuperação de casos inspirado no trabalho de Silva, Girardi e Drummond (2009) o qual utiliza o conceito de casos semânticos para representação dos itens de informação. Tal modelo é baseado em conhecimento, por isso as consultas e os documentos são representadas em instâncias de ontologias. Um caso semântico representa a característica de um item de informação, pelo qual os interesses de um usuário pode ser especificado (SILVA, GIRARDI E DRUMMOND, 2009).

Na arquitetura CBRAA o caso semântico corresponde aos atributos relevantes do problema de um caso RBC utilizados na recuperação, isto é, os índices do caso que permitem distingui-lo de outros (WANGENHEIM E WANGENHEIM, 2003). Os atributos relevantes são tópicos, sendo que, cada tópico descreve uma parte da ontologia do domínio, composta de um conceito mais geral, raiz da sua respectiva sub-hierarquia, e os seus sub-conceitos.

Portanto, formalmente, um atributo relevante de um caso RBC é definido como um par:

$$AR= (C_r, C_h) \quad (2)$$

onde C_r é o conceito raiz do atributo relevante e C_H é a hierarquia de conceitos de C_r.

Como exemplo de conceitos raízes e como se dá a formação dos atributos relevantes de um caso RBC, utilizaremos a classe *Problem* que representa o problema de um caso RBC na ontologia de casos do agente CBRAA como conceito raiz (C_r) e C_H a sua sub-hierarquia, no domínio em particular considerado. Sendo assim, a partir da ontologia ilustrada na Figura 28, *Problem* pode ser definido como na representação:

$$AR_{Problem}=(Problem,\{LegalCaseProblem\begin{Bmatrix}DivorceProblem,\ MarriageProblem,\\ AdoptionProblem\end{Bmatrix}) \quad (3)$$

Para representação interna de casos RBC utiliza-se uma estratégia baseada nos casos semânticos de Silva, Girardi e Drummond (2009), representando-se os conceitos e as instâncias que compõem a descrição de um caso RBC. Os atributos relevantes dos novos casos-problema devem ser mapeados em instâncias e conceitos da base de conhecimento do agente.

Um novo caso-problema no modelo de similaridade pode ser representado de acordo com a notação ncp $= \{P_1, \ldots, P_m\}$, na qual ncp corresponde ao novo caso-problema. O novo caso-problema é composto de um conjunto de pares P_j, onde j é o índice dos atributos relevantes do caso RBC utilizado para cálculo da similaridade do caso-problema. Cada par P_j contém o conjunto de conceitos representativos dos atributos relevantes e o conjunto de instâncias pertencentes à relação, podendo então ser representado por $P_j = \{CR_j, I_j\}$, onde I representa o conjunto das instâncias. Analogamente, um novo caso-problema é definido como:

$$ncp = \{(CR_1, I_1),..., (CR_m, I_m)\} \quad (4)$$

Da mesma forma, o problema dos casos recuperados da base de casos são definidos por:

$$cpr = \{(CR_1, I_1),..., (CR_m, I_m)\} \quad (5)$$

Assim, o novo caso-problema, “Mariana Lima, casada com Mauro Jorge em regime de comunhão parcial de bens, requer divórcio. O casal possui dois filhos menores.” e o caso-problema recuperado “Maria Silva e João Silva, casados em

regime de comunhão universal de bens, requerem divórcio. O casal não possui filhos.”, poderiam ser representados na seguinte forma:

$$ncp = \begin{Bmatrix} (\{DivorceProblem\},\{Divorce\ of\ Mariana\ Lima\ e\ Mauro\ Jorge\}),\ (\{One\},\{\ \}), \\ (\{Partial\ Community\ of\ Property\ Regime\},\{\ \}),\ (\{Yes\},\{\ \}) \end{Bmatrix}$$

$$cpr = \begin{Bmatrix} (\{DivorceProblem\},\{Divorce\ of\ João\ Silva\ e\ Maria\ Silva\}),\ (\{Both\},\{\ \}), \\ (\{Full\ Community\ of\ Property\ Regime\},\{\ \}),\ (\{No\},\{\ \}) \end{Bmatrix}$$

A similaridade entre os casos-problema no modelo proposto é dada pela fórmula:

$$sim\_case(ncp,cpr) = \frac{\sum_{j=1,n} W_j * sim\_attribute_j\left(Incp_j, Icpr_j\right)}{\sum_{j=1,n} W_j} \quad (6)$$

onde:

- j é o índice associado a cada atributo do problema RBC utilizado na recuperação;
- w_j é o peso atribuído ao atributo j.
- Incp é o conjunto de conceitos e instâncias dos valores dos atributos que representam o problema do novo caso-problema.
- Icpr é o conjunto de conceitos e instâncias dos valores dos atributos que representam o problema descrito no caso-problema recuperado da base de casos.
- $Sim\_attribute_j$ é a função para o cálculo de similaridade entre os valores dos atributos correspondentes do NCP e do CPR.

Desta forma, para o NCP e CPR exemplificados, considerando o mesmo peso para os atributos relevantes, a similaridade entre o novo caso-problema e o caso-problema recuperado poderia ser medida da seguinte forma:

$$\text{sim\_case}(ncp,cpr) = \sum_{j=1,n} w_j * \text{sim\_attribute}\begin{pmatrix}\text{Divorce of Mariana Lima e Mauro Jorge,}\\ \text{Divorce of João Silva e Maria Silva}\end{pmatrix} = (1 * \text{sim\_attribute}_{author}(one,both) + 1 * \text{sim\_attribute}_{matrimonialRegime}(\text{Partial Community of Property, Full Community of Property}) + 1 * \text{sim\_attribute}_{hasMinorChildren}(Yes,No)) / 3$$

A valor dos atributos que compõem o problema, representados na Equação 6, devem ter sua similaridade calculada de acordo com a Equação 7 definida para o cálculo do valor de similaridade entre conceitos (DRUMMOND, GIRARDI e SILVA, 2008), onde C corresponde ao conceito que representa o valor do atributo relevante ou o conceito representativo da instância correspondente no novo caso-problema e D ao valor do atributo correspondente no caso-problema recuperado:

$$\text{sim\_attribute}_j(C,D) = \frac{2 * |C_H \cap D_H|}{|C_H| + |D_H|} \quad (7)$$

onde:

- C_H é o conjunto formado por C e todos os superconceitos da hierarquia de C;
- D_H é o conjunto formado por D e todos os superconceitos da hierarquia de D;
- $|C_H|$ é o número de elementos do conjunto C_H;
- $|D_H|$ é o número de elementos do conjunto D_H;
- $|C_H \cap D_H|$ é o número de elementos na interseção dos conjuntos C_H e D_H;

Por exemplo, a comparação dos conceitos de autor entre os casos do exemplo é definida pela equação $\text{sim\_attribute}_{author}(one, both)$, e a similaridade deve ser calculada de acordo com a definição a seguir:

$$\text{sim\_attribute}_{author}(one,both) = \frac{2 \cdot |one_H \cap both_H|}{|one_H| + |both_H|}$$

O resultado da aplicação da medida de similaridade de cada conceito é multiplicado pelo peso do atributo no somatório que calcula a similaridade entre os casos-problema.

3.2.3 Mecanismo de Adaptação

Uma vez que uma solução de caso similar é recuperada, o agente RBC deverá ser capaz de adaptar a solução ao novo problema, por isso, a arquitetura do agente inclui ainda um mecanismo de adaptação de casos, necessário, visto que um caso similar não necessariamente vai ter uma solução que encaixará perfeitamente no novo caso, havendo, assim, necessidade de adaptá-la.

Ainda assim, na arquitetura CBRAA será possível com o aumento natural da base de casos, que ocorram adaptações do tipo nula, isto é, sem adaptação nenhuma da solução, no entanto vai depender da complexidade e do volume de casos aprendidos.

Dependendo da complexidade dos casos a serem adaptados poderão ser realizadas adaptações do tipo transformacional substitucional ou transformacional estrutural para casos mais simples, isto é ocorrendo apenas a substituição de valores e/ou elementos através de um conjunto de regras de adaptação pré-definidas representadas na base de conhecimento do agente e conhecimento do domínio. Nessa situação uma abordagem de adaptação hierárquica também pode ser utilizada tendo em vista a utilização de ontologias para a representação dos casos.

Em casos mais complexos a adaptação gerativa poderá ser utilizada. Desta forma, poderia ser utilizado todo o processo derivacional da solução anterior para tentar solucionar o novo caso-problema. Nas ontologias que representam a base da arquitetura proposta, as relações não-taxonômicas *Failed_Adapted_Cases* e *Adapted_Cases* irão colaborar com esse tipo de adaptação.

As regras a serem elaboradas para a adaptação poderão ser gerais e executadas de acordo com um modelo de raciocínio, como por exemplo, utilizando a seguinte regra geral:

“Se a esposa não trabalha deve receber pensão de pelo menos um salário mínimo”

Com a percepção de um novo caso-problema “Ação de divórcio na qual esposa e marido não possuem emprego”, o mecanismo de recuperação e análise de

similaridade poderá retornar um caso-problema anterior similar “Ação de divórcio na qual o marido possui emprego e a esposa não” e solução a ser adaptada “Pensão estabelecida em dois salários mínimos”. O novo caso-problema poderá ter como solução adaptada do caso recuperado com base na regra geral citada, isto é, independentemente da situação do marido há um valor mínimo de pensão a ser pago. Desta forma, a nova solução seria: “Pensão estabelecida em um salário mínimo”.

Outros conjuntos de regras que poderão ser utilizados pelo mecanismo, são regras de adaptação taxonômica. Essas regras irão ser definidas com base na hierarquia de classes da ontologia de casos podendo ainda utilizar o conhecimento do domínio e comparação semântica.

Regras semânticas podem compor o mecanismo, deixando ainda mais precisa a adaptação de casos. Essas regras poderiam se basear no grau de similaridade entre sinônimos, homônimos e parônimos, obtidos a partir de uma base de conhecimento linguístico como o Wordnet (MILLER et al., 2006), complementando, assim, o conjunto de regras.

O estudo desse mecanismo não foi aprofundado nesse trabalho e será abordado em trabalhos futuros. Dentre os desafios no estudo desse mecanismo, está a elaboração de forma automática ou semiautomática do conjunto de regras de adaptação a serem utilizadas.

3.2.4 Mecanismo de Aprendizado

Ainda como parte da arquitetura, o quarto mecanismo essencial para o funcionamento de um sistema RBC é o mecanismo de aprendizado de casos, que deverá ser capaz de incorporar o conhecimento do novo caso solucionado à base de conhecimento já existente. Sendo assim o agente deverá ser capaz de recuperar a solução desse caso para novos casos-problema.

Como visto no capítulo 2 um sistema RBC pode ter três tipos de retenção: Não ter retenção, reter novos casos solucionados e/ou retenção de casos a partir de documentos.

Na arquitetura CBRAA é necessária que haja uma retenção inicial de casos a partir de documentos, compostos de problemas e soluções de casos do domínio, isto é, retenção de forma assíncrona para que o agente possua uma base inicial de casos para recuperação. A inexistência de uma base inicial irá impossibilitar o

processo RBC, visto que, a etapa de recuperação de casos não poderá ser realizada, e, assim, não será possível a interação completa dos mecanismos e, por conseguinte, o RBC.

Considerando que a base de casos na arquitetura CBRAA é representada por uma ontologia de casos e os casos por instâncias dessa ontologia, o processo de retenção de conhecimento do agente proposto deverá ser capaz de extraí-lo de documentos ou percepções, mapear em instâncias das classes de ontologia que irão representar o novo caso, e, por fim, indexá-los e adicionar a base de casos.

Assim como o mecanismo de adaptação de casos, o mecanismo de aprendizado ainda não foi especificado em detalhe na arquitetura. No entanto, é possível afirmar que um estudo sobre técnicas de processamento de linguagem natural (ALLEN, 1995), povoamento de ontologias (FARIA, GIRARDI e NOVAIS, 2013) e aprendizagem baseada em instâncias (IBL, do inglês *Instance-Based Learning*) (MITCHEL, 1997) deverá ser realizado para composição do mecanismo.

3.3 Considerações finais

A arquitetura CBRAA inclui os quatro mecanismos de um sistema RBC, porém nesse trabalho detalhamos apenas os mecanismos de representação de casos, e o de recuperação e análise de similaridade. Estes mecanismos foram avaliados através da elaboração de um estudo de caso e de uma ontologia de casos no domínio do Direito de Família.

Nossa proposta diferencia-se de outros trabalhos do estado da arte (LESS e CORCHADO, 1997) (CORCHADO e LAZA (2012) (SRINIVASAN et al., 2011) devido ao suporte à representação de conhecimento semântico associado aos casos através do uso de ontologias para representação de casos e também a utilização de uma medida de similaridade semântica para comparação de casos. Outra proposta de arquitetura (GUAN-YU, LI-NING e SHI-PENG, 2008), por sua vez, utiliza ontologias no entanto não utiliza a abordagem de agentes.

A arquitetura mais similar à arquitetura CBRAA (GARRIDO et al., 2008) consiste na arquitetura de um agente RBC com especificações das ontologias e modelo de recuperação de casos diferentes. Além disso, na arquitetura CBRAA o agente possui todos os mecanismos de um sistema RBC em sua composição e ainda é responsável pela percepção do ambiente que irá identificar os novos casos-

problema a serem solucionados, além de obter a percepção *feedback* da utilização da solução no ambiente para que possa ser avaliada e armazenada.

A Tabela 3 mostra o conjunto destas características dos trabalhos relacionados utilizadas para comparação com a arquitetura CBRAA.

Tabela 3: Comparativo das características da arquitetura CBRAA com trabalhos relacionados

Trabalho	**Uso de RBC**	**Uso de Agentes**	**Uso de Ontologias**	**Modelo de Recuperação de Casos**
Corchado e Laza (2012)	Sim + BDI	Sim	Não	Um método de núcleo (FYVE e CORCHADO, 2001).
Less e Corchado (1997)	Sim + Redes Neurais Artificiais	Sim	Não	Identificação de Padrões através de Redes Neurais Artificiais.
Guan-Yu, Li-Ning e Shi-Peng (2008)	Sim	Não	Sim	Método de vizinho mais próximo baseado em semântica (PATTERSON, ROONEY e GALUSHKA, 2002).
Srinivasan et al. (2011)	Sim + Mineração de Dados	Sim	Não	Não especificado
Garrido et al. (2008)	Sim	Sim	Sim	Modelo Semântico (GARRIDO et al., 2008)
CBRAA	Sim	Sim	Sim	Modelo Semântico

A arquitetura CBRAA possui como vantagem a independência do agente RBC em relação às novas percepções do ambiente o que permite a execução completa do ciclo em caso de eventuais falhas de comunicação entre os agentes da sociedade que executam diferentes papéis, executando todas as atividades do ciclo RBC. E na arquitetura de Garrido et al. (2008), por sua vez, há a dependência da comunicação entre os agentes de interface e os diferentes agentes RBC das organizações. No entanto na arquitetura de Garrido et al. (2008) define-se o compartilhamento de conhecimento de um domínio entre organizações através da interação e troca de conhecimento entre agentes RBC proporcionando o enriquecimento de mais de uma base de conhecimento, o que na arquitetura CBRAA não foi abordado.

Os mecanismos de adaptação de casos e o de aprendizado necessário para retenção de novos casos e aprendizagem ainda são objetos de estudo dessa pesquisa, e, portanto, foram descritos de forma mais resumida. Contudo, também são essenciais para a execução das atividades de um sistema RBC.

4. AVALIAÇÃO

Neste capítulo é apresentado um estudo de caso na área jurídica do Direito de Família brasileiro para a avaliação da efetividade do mecanismo de recuperação e análise de similaridade da arquitetura CBRAA. Para tanto, foram projetados e implementados os mecanismos de representação e recuperação de casos da arquitetura.

O estudo de caso consistiu na construção de uma base de conhecimento composta de vinte casos (Tabela 4) e um conjunto de seis casos-problema utilizados como consulta no mecanismo de recuperação e análise de similaridade (Tabela 5). Os resultados obtidos foram avaliados em função da precisão dos casos relevantes recuperados para adaptação da solução. Para isso, foi utilizada a medida da precisão da área de recuperação de informação (DELLSCHAFT e STAAB, 2006), adaptada para o cálculo da precisão na recuperação de casos RBC, definida na Equação 8.

$$\text{Precisão} = \frac{|CPR_R \cap CPR_T|}{|CPR_T|} \quad (8)$$

onde:

- CPR_R é o conjunto de casos relevantes recuperados da base de casos.
- CPR_T é o conjunto de casos recuperados da base de casos.
- $|CPR_R \cap CPR_T|$ é o número de elementos do conjunto formado pela interseção dos conjuntos CPR_R e CPR_T.
- $|CPR_T|$ é o número de elementos do conjunto CPR_T.

A base de casos RBC que compõe a base de conhecimento do agente foi representada em ontologias e o projeto do mecanismo de análise de similaridade utilizado na recuperação de casos foi adaptado da proposta de Silva, Girardi e Drummond (2009).

Para realizar o estudo de caso foi necessária a elaboração de uma base de casos prévios, visto que um sistema RBC deve ser capaz de recuperar casos similares para adaptar suas soluções desde a sua primeira execução.

As próximas seções apresentam a especificação do projeto e implementação do agente CBRRA (seção 4.1), a base de conhecimento do agente

RBC (seção 4.2), a instanciação do modelo de similaridade para a recuperação de casos (seção 4.3), os resultados obtidos (seção 4.4), a discussão sobre a avaliação (seção 4.5), e, por fim, as conclusões a respeito do estudo de caso (seção 4.6).

4.1 Projeto e Implementação do agente

A representação da base de conhecimento inicial utilizada pelo agente foi elaborada através da ferramenta de edição de ontologias *Protégé* (GENNARI, MUSEN e FERGERSON, 2002).

Contudo, para que fosse possível realizar a implementação do mecanismo de similaridade utilizando-se de ontologias, elaborou-se uma representação destas através de fatos e regras em lógica de primeira ordem em Prolog utilizando-se a ferramenta WIN-PROLOG (GILMAN et al, 2010). Para elaboração dos agentes foi utilizado o CHIMERA (STEEL, 2005), uma ferramenta de desenvolvimento de agentes do WIN-PROLOG.

A ontologia foi construída na língua inglesa para facilitar a publicação de trabalhos relacionados à pesquisa. Por isso, a representação das classes, hierarquias de conceitos e instâncias, foram traduzidos e também estão representadas em língua inglesa.

4.2 A base de conhecimento do agente RBC

Os documentos utilizados para elaboração da base de conhecimento foram petições jurídicas de processos legais e certidões da área jurídica, em particular, do Direito de Família brasileiro (DINIZ, 2012).

Utilizamos nesta avaliação casos jurídicos de divórcio. Para isso, um estudo sobre os tipos de divórcio e suas características foi realizado para que fosse possível avaliar a efetividade do modelo de recuperação de casos e medida de similaridade semântica utilizados na arquitetura CBRAA, identificando dentre os casos recuperados quais são os relevantes.

Os tipos de divórcio atualmente existentes, de acordo com a emenda constitucional nº 66/2010 da constituição brasileira (DINIZ, 2012), podem ser classificados em um primeiro nível em divórcio extrajudicial e divórcio judicial. O primeiro tipo é o divórcio realizado através de escritura pública em cartório, que para

ser realizado deve atender a alguns pré-requisitos como: a inexistência de filhos menores e/ou incapazes, e a firme intenção do casal de interromper o vínculo matrimonial. O outro tipo de divórcio é o judicial, para os casos em que os pré-requisitos para o divórcio extrajudicial não são suportados. Esse tipo de divórcio subdivide-se ainda em litigioso e consensual. O litigioso ocorre quando há discordância entre os consortes em relação aos termos do encerramento do matrimônio como, por exemplo, assuntos referentes à partilha de bens, guarda de filhos, alimentos e modificação do nome, sendo necessário que ocorra uma discussão e definição em juízo acerca dos temas antes da conclusão do processo. Em quaisquer dos tipos de divórcio, quando da existência de bens comuns, há a necessidade de definição da partilha de bens que deve já ser definida previamente entre o casal e seus advogados nos casos do divórcio extrajudicial consensual e divórcio judicial consensual (DINIZ, 2012). A hierarquia dos tipos de divórcio vigentes no Brasil, segundo a EC nº 66/2010 (DINIZ, 2012), está representada na Figura 32.

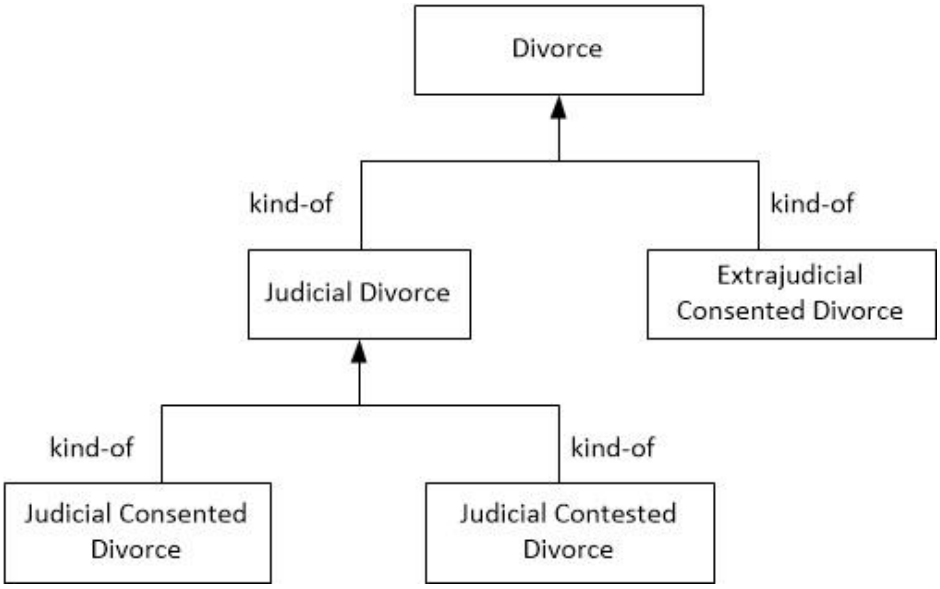


Figura 32: Hierarquia dos Conceitos de Tipos de divórcio.

O objetivo do agente CBRAA nesse estudo de caso é automatizar a tarefa feita manualmente pelos advogados de comparar casos similares para elaborar os documentos necessários para o divórcio de seu(s) novo(s) cliente(s) com menor esforço através da utilização de soluções anteriores que incluem características como o tipo de divórcio a ser realizado. Características como a existência de filhos menores e/ou incapazes, e a modalidade do regime de casamento irão influenciar na definição

do tipo de divórcio, de pensão alimentícia dos filhos e partilha de bens, que devem compor a descrição de tais documentos jurídicos. Essas características estão representados pelas propriedades *hasMinorChildren* e *matrimonialRegime* respectivamente, ilustradas na Figura 33. Essa figura representa a classe do problema de divórcio definida na ontologia de casos, destacando apenas os atributos do problema considerados como relevantes na avaliação.

DivorceProblem	
IsThereAlimonyRequirement	Boolean
IsThereNameChangeRequirement	Boolean
author	String
hasMinorChildren	Boolean
matrimonialRegime	String

Figura 33: Representação parcial da Classe *DivorceProblem* da Ontologia de Casos.

O regime de bens definido na casamento influencia na definição da divisão de bens do casal quando da dissolução do mesmo. Existem quatro tipos de regime de casamento no Brasil. O mais comum, utilizado como regime padrão quando da inexistência de um pacto pré-nupcial, é o regime de “Comunhão parcial de bens”. Nessa modalidade existe a separação entre os bens adquiridos antes do casamento por cada um dos consortes, classificados então como incomunicáveis e irrelevantes na divisão de bens, e os bens adquiridos durante o casamento, que serão divididos entre os consortes quando da homologação do divórcio. Existe ainda outro regime de comunhão de bens denominado “Comunhão universal de bens”, pelo qual, através de um pacto antenupcial, os bens de cada consorte adquiridos antes do casamento também entram na partilha em situação de divórcio (DINIZ, 2012).

O terceiro regime, denominado “Separação total de bens”, não compartilha as características dos regimes de comunhão. Nesse regime há um limite pré-estabelecido dos bens de cada consorte, independentemente de terem sido adquiridos antes ou durante o casamento. Os bens nesse regime são incomunicáveis, não havendo necessidade de especificação de divisão de bens na solução de divórcio.

Há ainda o regime de “Participação final nos aquestos”. Esse regime compartilha de características tanto do regime de “Comunhão parcial de bens” quanto do regime de “Separação total de bens”. Nesse tipo existem os bens de cada consorte

separados, adquiridos antes ou durante o casamento, e os bens comuns adquiridos em conjunto pelo casal enquanto casados. Em caso de dissolução do casamento, os bens comuns, quando provado que foram adquiridos em conjunto durante o casamento, deverão ser divididos entre as partes. A hierarquia dos tipos de casamento está representada através da Figura 34.

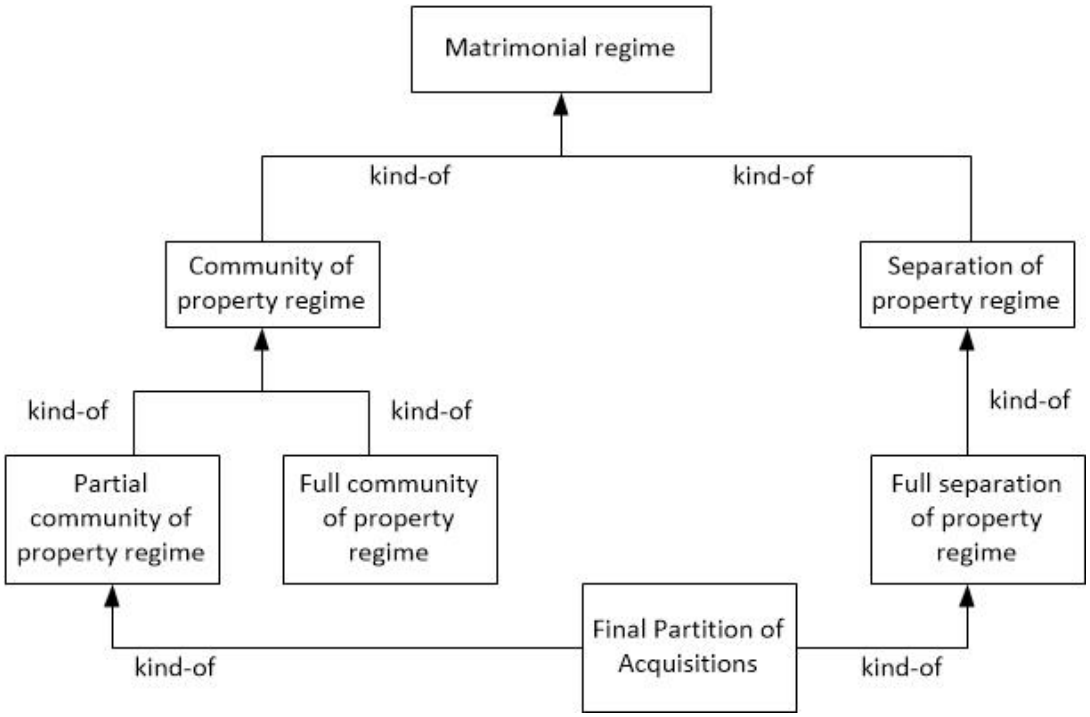


Figura 34: Hierarquia dos Conceitos de Regimes de Casamento.

Outro atributo do problema de divórcio considerado na avaliação é o autor do processo. Esse atributo representa o requerente, se é apenas o marido ou a esposa, isto é, uma ação individual, ou ambos os consortes, o que caracteriza uma ação conjunta. Se for uma ação individual a solução proposta pelo advogado será definida privilegiando os interesses de seu cliente. Em caso de ação conjunta, deve haver um equilíbrio entre os interesses de cada parte na definição do documento correspondente à solução de divórcio. Sendo assim, há apenas dois conceitos totalmente distintos na hierarquia que podem constar como valor desse atributo. São eles: individual ou ambos. Em caso de equivalência do valor do atributo do novo caso-problema em relação ao caso-problema recuperado, o valor de similaridade será igual a “1”, e, caso contrário, similaridade igual a “0”.

Os atributos que definem a existência de filhos menores e/ou incapazes, requerimento de pensão alimentícia, e mudança de nome são caracterizados por

valores booleanos, que indicam apenas “Sim” ou “Não”, e portanto, o valor de similaridade calculado para o atributo será similar ao do autor.

Esses conceitos são considerados relevantes na seleção do caso mais similar pois indicam a existência de certas características na solução, como por exemplo, o valor “Sim” para o atributo que representa a existência de filhos menores e/ou incapazes ocasionará em valor de similaridade maior em casos de divórcio do tipo judicial, pois não é possível realizar a modalidade extrajudicial nessa situação.

A Tabela 4 representa os valores dos atributos dos 20 casos RBC que compõem a base de conhecimento inicial do agente CBRAA implementado nessa avaliação. Cada caso está listado com os valores dos atributos do problema jurídico de divórcio considerados como relevantes, e que foram utilizados para o cálculo do valor de similaridade entre os casos RBC de divórcio. Esses, casos que denominamos caso-problema recuperado (CPR), terão sua similaridade calculada em relação a cada novo caso-problema e poderão ser recuperados. Na Tabela 4, cada CPR está listado ainda com o valor de um dos atributos da solução, que corresponde ao tipo de divórcio adotado.

Tabela 4: Casos da base de conhecimento inicial do agente CBRAA da avaliação.

ID	Consorts	matrimonial Regime	Author	hasMinor Children	Is there a Alimony Requirement?	Is there Name Change Requirement?	**divorceType (Solution Part)**
CPR_1	H_1 e W_1	Full separation of property	One	Yes	No	Yes	Judicial Consented
CPR_2	H_2 e W_2	Full community of property	One	Yes	No	No	Judicial Contested
CPR_3	H_3 e W_3	Final Partition of Acquisitions	One	Yes	Yes	Yes	Judicial Consented
CPR_4	H_4 e W_4	Partial community of property	One	No	No	No	Judicial Contested
CPR_5	H_5 e W_5	Full separation of property	Both	No	Yes	No	Judicial Contested
CPR_6	H_6 e W_6	Full community of property	Both	No	No	Yes	ExtraJudicial Consented
CPR_7	H_7 e W_7	Final Partition of Acquisitions	Both	No	Yes	No	ExtraJudicial Consented
CPR_8	H_8 e W_8	Partial community of property	Both	Yes	No	No	Judicial Contested
CPR_9	H_9 e W_9	Full separation of property	One	Yes	Yes	Yes	Judicial Contested
CPR_{10}	H_{10} e W_{10}	Full community of property	One	No	No	No	Judicial Consented
CPR_{11}	H_{11} e W_{11}	Final Partition of Acquisitions	One	No	Yes	No	Judicial Contested
CPR_{12}	H_{12} e W_{12}	Partial community of property	One	No	No	Yes	Judicial Consented
CPR_{13}	H_{13} e W_{13}	Full separation of property	Both	No	Yes	Yes	ExtraJudicial Consented
CPR_{14}	H_{14} e W_{14}	Full community of property	Both	Yes	No	No	Judicial Contested
CPR_{15}	H_{15} e W_{15}	Final Partition of Acquisitions	Both	Yes	Yes	Yes	Judicial Consented
CPR_{16}	H_{16} e W_{16}	Partial community of property	Both	No	No	No	ExtraJudicial Consented
CPR_{17}	H_{17} e W_{17}	Full separation of property	One	No	Yes	No	Judicial Contested
CPR_{18}	H_{18} e W_{18}	Full community of property	One	No	No	Yes	Judicial Contested
CPR_{19}	H_{19} e W_{19}	Final Partition of Acquisitions	One	Yes	Yes	Yes	Judicial Consented
CPR_{20}	H_{20} e W_{20}	Partial community of property	One	No	No	No	Judicial Consented

A instanciação dos CPRs que compõem a base de casos foi realizada de forma manual. Para isso, foram identificados nas petições de divórcio, documentos utilizados para definição dos atributos deste tipo de caso RBC, os valores dos atributos dos problemas de divórcio para instanciação da base de casos. A elaboração da ontologia de casos, utilizada para representação dos casos RBC, e o mapeamento da hierarquia de conceitos dos atributos dos casos, que também compõem a base de conhecimento do agente CBRAA, foram realizadas como auxílio de um profissional de Direito e foram modeladas através da ferramenta Protégé (GENNARI, MUSEN, FERGERSON, 2002).

4.3 Recuperação e Análise de Similaridade

A recuperação de casos similares com seus respectivos valores de similaridade utiliza o modelo de recuperação de casos proposto na arquitetura CBRAA.

Os experimentos foram realizados através de um agente implementado com uma base de casos inicial e o mecanismo de recuperação e análise de similaridade, a partir da instanciação do modelo de recuperação e medida de similaridade semântica propostos nesse trabalho. Foram selecionados casos da área jurídica do Direito de Família brasileiro, os quais tiveram seus valores de atributos mapeados na representação de novos casos-problema RBC de divórcio, e que foram submetidas ao agente. Os testes foram conduzidos com os seguintes objetivos:

- Geração da representação interna do novo caso-problema;
- Validação do modelo de recuperação e medida de similaridade semântica;
- Avaliação da precisão na recuperação de casos RBC.

4.3.1 Consultas

Para submissão das consultas ao agente CBRAA foi implementado um agente de interface para simulação do ambiente e percepções do agente correspondentes à novos casos-problema. Para cada NCP foram informados os valores dos atributos relevantes que o definem (Tabela 5), através da tela do agente de interface (Figura 35). Após submetida, a consulta é então mapeada para uma

representação interna e, por meio da aplicação da medida de similaridade é comparada com as instâncias dos casos da base e o valor de similaridade em relação a cada um deles é calculado.

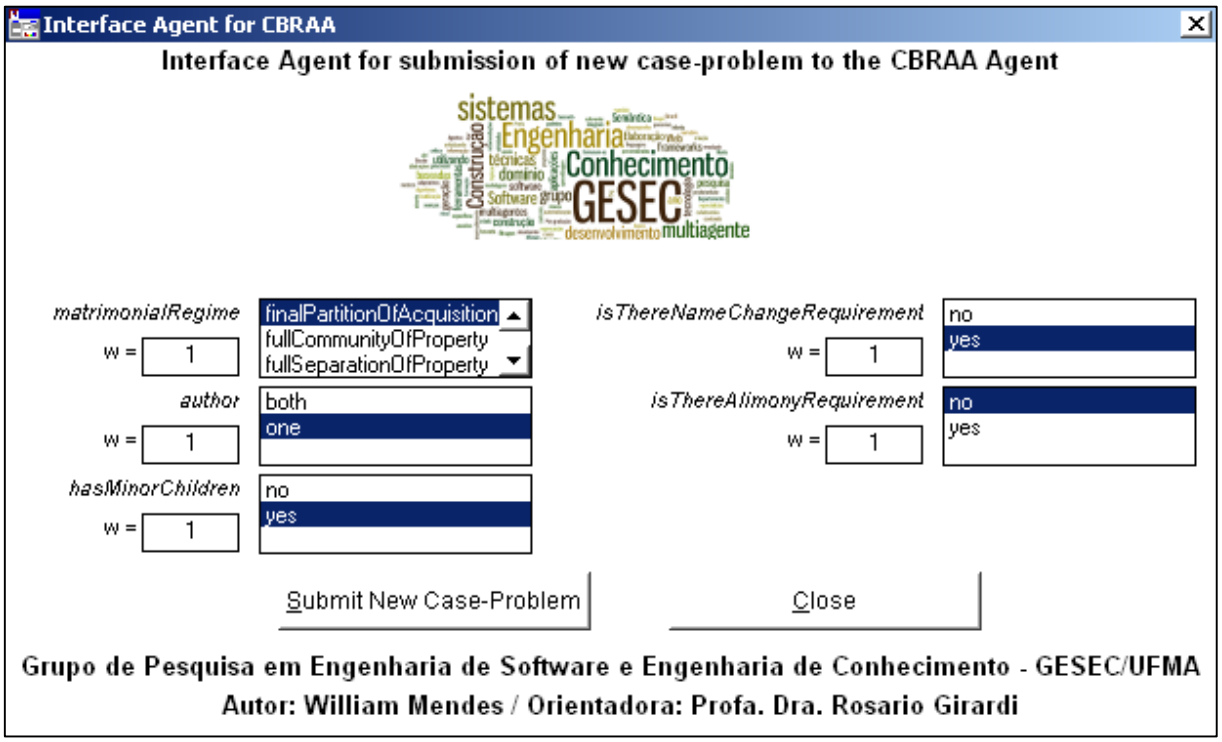


Figura 35: Tela de consulta de novos casos-problemas ao agente CBRAA.

Em seguida, o agente CBRAA retorna uma lista com os CPRs da base de conhecimento do agente que podem ter a solução reutilizada, com seus respectivos valores de similaridade e valores dos atributos da solução (Figura 36).

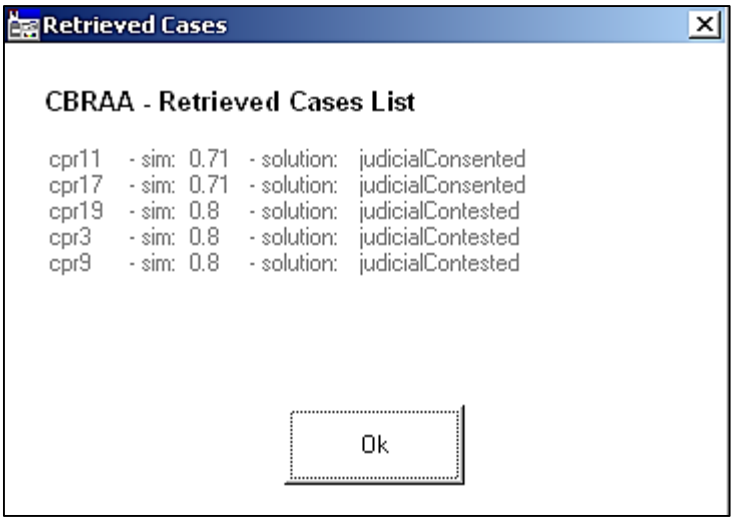


Figura 36: Exemplo da lista de CPRs recuperados pelo agente CBRAA

Os novos casos-problema submetidos com o mesmo conjunto de atributos que descrevem o problema dos CPRs estão representados na Tabela 5. Estão enumerados nas Tabelas 4 e 5 apenas os atributos do problema considerados como relevantes para o cálculo do valor de similaridade. Um total de seis novos casos-problema foi submetido na avaliação como consultas no mecanismo de recuperação e análise de similaridade de casos. Cada caso-problema submetido foi denominado de Novo Caso-Problema (NCP).

Tabela 5: Casos-Problema submetidos como consultas no estudo de caso.

ID	Consorts	matrimonial Regime	Author	hasMinor Children	Is there a Alimony Requirement?	Is there Name Change Requirement?
NCP_1	H_{21} e W_{21}	Final partition of Acquisitions	One	Yes	Yes	No
NCP_2	H_{22} e W_{22}	Full community of property	One	No	Yes	Yes
NCP_3	H_{23} e W_{23}	Full separation of property	Both	Yes	No	Yes
NCP_4	H_{24} e W_{24}	Partial community of property	One	No	No	Yes
NCP_5	H_{25} e W_{25}	Partial community of property	One	Yes	Yes	No
NCP_6	H_{26} e W_{26}	Full community of property	Both	No	No	No

4.3.2 Medida de Similaridade

O cálculo da similaridade entre um NCP e um CPR se dá pela Equação 6 na seção 3.2.2 que corresponde ao somatório do valor de similaridade dos atributos relevantes do caso-problema para cálculo de similaridade, dividido pela soma dos pesos dos atributos, que por sua vez possuem seus valores de similaridade calculados através da Equação 7 na seção 3.2.2.

Na avaliação, os atributos relevantes em um caso-problema de divórcio foram *matrimonialRegime*, *Author*, *hasMinorChildren*, *IsThereAlimonyRequirement e IsThereNameChangeRequirement* representados nas Tabelas 4 e 5 para os casos RBC da base de conhecimento do agente e para os novos casos-problema respectivamente. Sendo assim, para exemplificação, a similaridade entre NCP_1 e CPR_1 é calculada conforme detalhamento a seguir.

Calcula-se primeiramente a similaridade entre os atributos listados abaixo através da Equação 7 da seção 3.2.2.

a) *matrimonialRegime*

$$\text{Final partition of acquisitions}_H = \left\{ \begin{array}{c} \text{Final partition of acquisitions, Full separation of property regime,} \\ \text{Partial community property regime, Separation of property regime,} \\ \text{Community property regime} \end{array} \right\}$$

$$\text{Full separation of property}_H = \{\text{Full separation of property, Separation of property}\}$$

$$\text{Sim\_attribute}_{\text{matrimonialRegime}}(\text{Final partition of acquisitions, Full separation of property}) = \frac{2 * |\text{Final partition of aquisitions}_H \cap \text{Full separation of property}_H|}{|\text{Final partition of acquisitions}_H| + |\text{Full separation of property}_H|} = \frac{2 * 2}{5 + 2} = 0{,}57$$

b) *Author*

$$\text{One}_H = \{\text{One}\}$$

$$\text{Sim\_attribute}_{\text{Author}}(\text{One,One}) = \frac{2 * |\text{One}_H \cap \text{One}_H|}{|\text{One}_H| + |\text{One}_H|} = \frac{2 * 1}{1 + 1} = 1$$

c) *hasMinorChildren*, *IsThereAlimonyRequirement e IsThereNameChangeRequirement*

$$\text{Yes}_H = \{\text{Yes}\}$$

$$\text{No}_H = \{\text{No}\}$$

$$\text{Sim\_attribute}_{\text{hasMinorChildren}}(\text{Yes,Yes}) = \frac{2 * |\text{Yes}_H \cap \text{Yes}_H|}{|\text{Yes}_H| + |\text{Yes}_H|} = \frac{2 * 1}{1 + 1} = 1$$

$$\text{Sim\_attribute}_{\text{IsThereAlimonyRequirement}}(\text{No,Yes}) = \frac{2 * |\text{No}_H \cap \text{Yes}_H|}{|\text{No}_H| + |\text{Yes}_H|} = \frac{2 * 0}{1 + 1} = 0$$

$$\text{Sim\_attribute}_{\text{IsThereNameChangeRequirement}}(\text{Yes,No}) = \frac{2 * |\text{Yes}_H \cap \text{No}_H|}{|\text{Yes}_H| + |\text{No}_H|} = \frac{2 * 0}{1 + 1} = 0$$

Em seguida, o valor de similaridade entre o novo caso-problema NCP_1 e o caso-problema recuperado CPR_1, atribuindo-se peso igual à todos os atributos relevantes (w = 1), é então definido por:

$sim\_case(NCP_1, CPR_1)$ =

$(1 * Sim_{matrimonialRegime}(\text{Final partition of Acquisitions, Full Separation of Property}) +$

$1 * Sim_{Author}(One, One) + 1 * Sim_{hasMinorChildren}(Yes, Yes) +$

$1 * Sim_{IsThereAlimonyRequirement}(No, Yes) + 1 * Sim_{IsThereNameChangeRequirement}(Yes, No)) / 5$

$= 0,51$

O agente CBRAA calcula o valor de similaridade através dessa medida para cada um dos CPRs da base de conhecimento em relação aos NCPs submetidos. No entanto, somente recupera para possível adaptação os CPRs com um valor mínimo de similaridade e que poderão ter suas soluções adaptadas para reutilização. Para tanto, considera que deve recuperar apenas aqueles que possuem um valor de similaridade superior ou igual à 0,7.

A Tabela 6 representa uma lista com os valores de similaridade de cada CPR em relação aos NCPs submetidos e estão destacados em negrito apenas os casos recuperados pelo agente.

Tabela 6: Valor de Similaridade de cada novo caso-problema em relação aos casos recuperados da base para w=1 em todos os atributos.

	NCP1	NCP2	NCP3	NCP4	NCP5	NCP6
CPR1	0,51	0,40	**0,80**	0,60	0,40	0,20
CPR2	0,66	0,40	0,40	0,50	**0,70**	0,60
CPR3	**0,80**	0,66	0,51	0,51	**0,71**	0,06
CPR4	0,51	0,50	0,20	**0,80**	0,60	**0,70**
CPR5	0,51	0,40	0,40	0,20	0,40	0,60
CPR6	0,06	0,60	0,60	**0,70**	0,10	**0,80**
CPR7	0,60	0,46	0,31	0,31	0,51	0,66
CPR8	0,51	0,10	0,60	0,40	0,60	**0,70**
CPR9	**0,71**	0,60	0,60	0,40	0,60	0,00
CPR10	0,46	0,60	0,20	**0,70**	0,50	**0,80**
CPR11	**0,80**	0,66	0,11	0,51	**0,71**	0,46
CPR12	0,31	**0,70**	0,40	**1,00**	0,40	0,50
CPR13	0,31	0,60	0,60	0,40	0,20	0,40
CPR14	0,46	0,20	0,60	0,30	0,50	**0,80**
CPR15	0,60	0,46	**0,71**	0,31	0,51	0,26
CPR16	0,31	0,30	0,40	0,60	0,40	**0,90**
CPR17	**0,71**	0,60	0,20	0,40	0,60	0,40
CPR18	0,26	**0,80**	0,40	**0,90**	0,30	0,60
CPR19	**0,80**	0,66	0,51	0,51	**0,71**	0,06
CPR20	0,51	0,50	0,20	**0,80**	0,60	**0,70**

U3m sistema RBC deve ser capaz de adaptar a solução recuperada ao novo caso-problema, por isso não é requerido que o valor de similaridade entre os NCP e o CPR recuperado seja igual a “1”.

Através do cálculo dos valor similaridade entre CPR_1 e NCP_1 exemplificado, foi possível mensurar a utilidade da solução de CPR_1 na resolução do novo caso-problema NCP_1. O valor de similaridade entre CPR_1 e NCP_1 é 0,51 e por isso não foi recuperado pelo mecanismo de recuperação e análise de similaridade do agente. Esse valor foi abaixo de 0,7 devido à pequena similaridade entre os valores dos atributos que representam o problema de CPR_1 e de NCP_1. Isso ocorreu por que o valor do atributo referente ao regime de casamento possui um baixo valor de similaridade e os atributos correspondentes à existência de solicitação de troca de nome e pensão alimentícia não possuem nenhuma similaridade.

Para o novo caso-problema NCP_1, nota-se que há cinco casos recuperados, e destes, três com o valor de similaridade idênticos, igual a 0,80, correspondentes aos casos com maior valor de similaridade em relação à este novo caso-problema. São eles: CPR_3, CPR_{11} e CPR_{19}. Portanto, a solução de um desses casos seria apropriada para a adaptação e reutilização pelo NCP. No entanto, esses valores foram obtidos considerando apenas um conjunto restrito de atributos relevantes utilizados pela avaliação, e outros atributos poderiam ser considerados para melhorar essa classificação. Em uma implementação mais completa poderiam ser considerados como relevantes outros atributos do divórcio que poderiam diferenciar o valor de similaridade desses casos em relação a NCP_1 e consequentemente escolher o mais apropriado para adaptação e reúso da solução.

4.4 Resultados

Para o cálculo da precisão na recuperação de casos RBC, utilizando-se casos legais do Direito de Família brasileiro, em particular dos casos legais de divórcio, consideramos como casos relevantes os que possuem maior valor de similaridade nos atributos que correspondem às principais características da solução em uma petição de divórcio. São eles o atributo correspondente ao regime de casamento, que influencia fortemente na definição da divisão de bens do casal, uma das questões mais debatidas em um divórcio, e os atributos referentes ao autor e existência de filhos menores, que influenciam diretamente na modalidade de divórcio a ser aplicada, atributo da solução

representado nessa avaliação (Tabela 4). Por isso, consideramos como relevantes casos em que há similaridade prioritariamente nesses itens.

Como por exemplo, no cálculo da precisão de NCP_1 foram recuperados cinco casos-problema da base, no entanto, um deles, o CPR_{17}, não foi considerado como relevante por ter como valor do atributo de regime de casamento a “Separação total de bens”, e, portanto não possui detalhamento de partilha de bens, enquanto que NCP_1 possui regime “Participação final nos aquestos”, e, além disso, os outros atributos também não possuem similaridade o suficiente para adaptação da solução com menor esforço para o novo caso-problema NCP_1. Essa identificação de casos relevantes foi realizada com um profissional de Direito para essa avaliação, e portanto os casos que serão considerados relevantes deverão variar de acordo com o domínio e auxílio de um especialista de domínio.

A avaliação foi realizada em função do cálculo da medida de precisão de casos recuperados relevantes. A precisão foi calculada através da medida da precisão da área de recuperação de informação (Equação 8). O cálculo da precisão de NCP_1 é definido como:

$$\text{Precisão de } NCP_1 = \frac{|CPR_R \cap CPR_T|}{|CPR_T|} = \frac{4}{5} = 0{,}80$$

O valor da precisão na recuperação de casos para cada NCP utilizado como consulta na avaliação em relação ao número de CPRs relevantes recuperados da base de casos está listada na Tabela 7, com os respectivos quantitativos de casos recuperados e casos recuperados relevantes.

Tabela 7: Medida de precisão de recuperação de casos para cada novo caso-problema utilizado na avaliação.

NCP	***Nº de CPR Relevantes***	***Nº de CPR Recuperados***	***Precisão***
NCP_1	*4*	*5*	*0,8*
NCP_2	*2*	*2*	*1,0*
NCP_3	*2*	*2*	*1,0*
NCP_4	*6*	*6*	*1,0*
NCP_5	*4*	*4*	*1,0*
NCP_6	*7*	*7*	*1,0*

Para todos os novos casos-problema obteve-se um valor de precisão igual ou superior a 60%, e por isso consideramos como eficiente a recuperação de casos RBC da arquitetura CBRAA.

4.5 Discussão

A avaliação realizada teve como objetivo verificar a viabilidade de implementação do agente CBRAA e a verificação da precisão do mecanismo de recuperação e análise de similaridade, isto é, sua capacidade de recuperar dentre os casos da base de conhecimento, os casos mais relevantes em relação aos novos casos-problema (Tabela 5) para reutilização de sua solução após adaptação. Todos os atributos relevantes para o cálculo da similaridade entre os novos casos-problema e os casos-problema recuperados foram considerados como tendo pesos idênticos na aplicação da medida de similaridade, $\omega=1$, e obtiveram um valor de precisão na recuperação satisfatório. No entanto, para uma melhor precisão na recuperação de casos, alguns atributos poderiam ser considerados com um peso diferenciado, como por exemplo os atributos correspondentes ao regime de casamento, a existência de filhos menores e/ou incapazes e o autor. O valor desses atributos influencia diretamente em um parte da solução com grande relevância, que é a definição do tipo de divórcio. Por isso, os atributos referentes à existência de solicitação de pensão alimentícia e mudança de nome poderiam ter um peso menor, tal qual, $\omega=0,5$. Caso isso fosse realizado a precisão seria melhor. O valor de similaridade de cada NCP em relação aos CPRs nessa situação estão listados na Tabela 8. Como por exemplo, na submissão de NCP_1, não foi mais recuperado o caso NCP_{17}.

Tabela 8: Valor de Similaridade de cada novo caso-problema em relação aos casos recuperados da base com pesos diferenciados.

	NCP1	NCP2	NCP3	NCP4	NCP5	NCP6
CPR1	0,64	0,38	**0,75**	0,50	0,50	0,13
CPR2	0,70	0,50	0,38	0,50	**0,75**	0,50
CPR3	**0,88**	0,57	0,52	0,52	**0,77**	0,07
CPR4	0,52	0,63	0,13	**0,88**	0,63	0,63
CPR5	0,39	0,38	0,50	0,25	0,25	0,63
CPR6	0,07	0,63	0,50	0,63	0,13	**0,88**
CPR7	0,50	0,45	0,39	0,39	0,39	0,70
CPR8	0,52	0,13	0,63	0,38	0,63	0,63
CPR9	**0,77**	0,50	0,63	0,38	0,63	0,00
CPR10	0,45	**0,75**	0,13	**0,75**	0,50	**0,75**
CPR11	**0,75**	0,70	0,14	0,64	0,64	0,45
CPR12	0,39	**0,75**	0,25	**1,00**	0,50	0,50
CPR13	0,27	0,50	0,63	0,38	0,13	0,50
CPR14	0,45	0,25	0,63	0,25	0,50	**0,75**
CPR15	0,63	0,32	**0,77**	0,27	0,52	0,32
CPR16	0,27	0,38	0,38	0,63	0,38	**0,88**
CPR17	0,64	0,63	0,25	0,50	0,50	0,38
CPR18	0,32	**0,88**	0,25	**0,88**	0,38	0,63
CPR19	**0,88**	0,57	0,52	0,52	**0,77**	0,07
CPR20	0,52	0,63	0,13	**0,88**	0,63	0,63

Essa simulação utilizada com peso diferenciado obteve precisão na recuperação de casos para todos os NCPs, valor igual a 1. Contudo, a precisão deverá variar em função dos atributos relevantes considerados e dos pesos de cada um deles, definidos pelo especialista no domínio.

A tendência da base de conhecimento do agente é de crescimento, visto que, a cada caso solucionado e avaliado como útil pelo mecanismo de aprendizado, ele será adicionado à base de conhecimento. Dessa forma, mais casos com características diferentes, tanto referentes ao problema quanto a solução serão utilizados para favorecer a recuperação de soluções mais apropriadas à novos casos-problema, fornecendo, assim, uma maior diversidade de casos a serem comparados e possivelmente recuperados com características diferentes.

4.6 Considerações Finais

Esse capítulo apresentou um estudo de caso do agente CBRAA para avaliação do modelo de recuperação da informação adaptado de Silva, Girardi e Drummond (2009) para recuperação de casos similares RBC, aplicado à área do Direito de Família brasileiro (DINIZ, 2012), através da implementação parcial dos componentes da arquitetura. Por meio dos mecanismos de representação de casos, e de recuperação e análise de similaridade e, utilizando-se de casos de divórcio, o estudo cumpriu os objetivos de representar os casos RBC através de um formalismo baseado em ontologias e avaliar a efetividade da recuperação de casos através da instanciação do modelo semântico de recuperação de casos.

Nesse estudo de caso, mostrou-se a possibilidade de implementação de uma arquitetura de agente RBC que utiliza ontologias para representação de casos e utilizando a medida de similaridade semântica proposta, demonstrando, assim, sua aplicabilidade, e ficando pendente apenas a definição e implementação dos mecanismos de adaptação e aprendizado que compõem a arquitetura CBRAA.

5. CONCLUSÃO

Esse trabalho apresentou a proposta da arquitetura do agente CBRAA que inclui:

- autonomia de um agente de software permitindo a resolução de problemas a partir de percepções obtidas do ambiente.
- ontologias para representação da base de conhecimento do agente RBC, o que permite reutilizá-la, ter conhecimento semântico associado aos casos e, consequentemente, uma melhor precisão nos resultados obtidos no processo de recuperação de casos através de uma medida de similaridade semântica para calcular o valor de similaridade entre os casos;
- o paradigma de resolução de problemas RBC, um paradigma de raciocínio comum ao ser humano.

A arquitetura proposta foi avaliada através de um estudo de caso na área do Direito de Família brasileiro, com a implementação dos mecanismos de representação de casos, e de recuperação e análise de similaridade. A base de conhecimento utilizou uma ontologia desenvolvida para este fim, que descreveu os casos RBC, em particular, casos de divórcio e os elementos do domínio necessários para representá-los.

Esse trabalho contribui para a construção de sistemas complexos mais efetivos na resolução de problemas, com a especificação de uma arquitetura de agente RBC que une a tecnologia de agentes de software com o paradigma de raciocínio RBC, através da inclusão de todos os mecanismos de um sistema RBC na arquitetura, e, ainda, utiliza as estruturas de representação baseadas no conhecimento para a representação de casos, através de um formalismo baseado em ontologias.

A especificação da arquitetura CBRAA contribuiu ainda com a especificação de um modelo de recuperação de casos baseado em conhecimento para a recuperação de casos similares RBC, e demonstrando a aplicabilidade do modelo através da implementação do agente e do mecanismo de recuperação e análise similaridade para avaliação da efetividade da recuperação de casos.

5.1 Resultados

A avaliação conduzida neste trabalho foi realizada com foco apenas nos mecanismos de representação de casos para formação da base de casos RBC inicial e o de recuperação e análise de similaridade, e foi focada no domínio de casos jurídicos do Direito de Família brasileiro, mais especificamente com casos jurídicos de divórcio.

Através da avaliação mostrou-se ser possível implementar os mecanismo de representação, e recuperação e análise de similaridade do agente CBRAA. No entanto a avaliação teve limitações como por exemplo a utilização de um domínio específico na instanciação dos casos e portanto a avaliação não obteve resultados acerca da precisão da recuperação em uma diversidade maior de casos RBC. Contudo, isso ocorreu por que o aprendizado das ontologias que compõem a base de conhecimento do agente foi realizada de forma manual e orientada para esse domínio. Através da automação desse processo com a utilização de técnicas de aprendizado de ontologias será possível definir novas classes que representam problemas de outros domínios, assim como as hierarquias de conceitos dos atributos necessárias para a medida de similaridade dos atributos.

As principais contribuições dessa pesquisa foram:

- Definição da arquitetura de agente deliberativo que une as características do paradigma RBC e a autonomia de um agente de software. Consideramos que o uso de agentes com RBC é uma vantagem, pois permite evoluir dos sistemas RBC tradicionais para sistemas RBC autônomos.
- Adaptação de um modelo de recuperação de informação para o uso em sistemas RBC.
- Especificação de uma medida de similaridade semântica para a comparação de casos RBC.
- Definição de uma ontologia com a definição de uma hierarquia de casos RBC que servem de base para a representação de casos em qualquer domínio.

5.1.1 Publicações e Submissões

- Mendes, W., Girardi, R., Leite, A. Arquitetura Baseada em Ontologias de um Agente RBC, 8ª Conferência Ibérica de Sistemas e Tecnologias de Informação, v.1, pp. 776-781. Lisboa - Portugal. 19 a 22 de junho de 2013".

Nesta publicação foi apresentada a primeira especificação da arquitetura CBRAA, na qual descreveu-se os componentes da arquitetura e foi definido o modelo de recuperação da informação baseado no conhecimento a ser adaptado e utilizado na arquitetura para a recuperação de casos.

- Mendes, W., Girardi, R., Leite, A., Using Ontologies in CBR Agents, artigo submetido no dia 10.10.2013 para o Expert Systems with Applications journal. Qualis Capes A1 Engenharias IV (em análise).

Nesta publicação foi apresentada a arquitetura CBRAA, detalhando os mecanismos de representação de casos e o de recuperação e análise de similaridade, com o modelo semântico de recuperação de casos especificado. Foi apresentada ainda avaliação em função da precisão da recuperação de casos realizada através da implementação da base de casos e do mecanismo de recuperação e análise de similaridade da arquitetura.

5.2 Trabalhos Futuros

Os mecanismos de adaptação e aprendizado de casos ainda são tópicos de estudo desta pesquisa. O mecanismo de adaptação deverá ser desenvolvido de forma que seja capaz de utilizar conhecimento do domínio e regras de adaptação contidos na base de conhecimento do agente CBRAA para adaptar e definir soluções para o novo caso-problema. O mecanismo de aprendizado deverá ser capaz de armazenar na base de casos o novo caso resolvido, no entanto, antes deverá ser capaz de realizar uma avaliação de utilidade da solução utilizada (WANGENHEIM e WANGENHEIM, 2003) e da necessidade de retenção do novo caso (ONTAÑÓN e PLAZA, 2003) antes de armazená-lo.

O estudo para especificação do mecanismo de aprendizado já vem sendo realizado pelo GESEC, e terá como produto uma nova dissertação. Neste novo trabalho estará descrito o funcionamento do mecanismo na arquitetura do agente CBRAA através de IBL (MITCHELL, 1997). Dentre os tipos de algoritmos utilizados na IBL, o agente CBRAA utilizará a aprendizagem baseada em casos, típica do RBC, onde instâncias são consideradas casos.

Uma maior experimentação através do uso de uma base de casos composta de um número mais elevado de casos e um número maior de atributos relevantes considerados também se faz necessária. A análise de sinônimos e a utilização da generalização de termos através do uso de uma base de dados de conhecimento linguístico como o Wordnet (MILLER et al., 2006) também pode ser utilizada na experimentação para apoiar a recuperação de casos.

Além disso, está prevista uma melhoria no mecanismo de análise de similaridade através da qual será possível identificar e atribuir pesos automaticamente aos diferentes atributos relevantes do problema utilizados para o cálculo do valor de similaridade, através da avaliação do uso das soluções adaptadas dos casos recuperados. Através do estudo dos atributos dos casos armazenados e do histórico de reutilização dos casos, o mecanismo deverá ser capaz ao longo do tempo de identificar quais atributos devem possuir um maior peso no cálculo da análise de similaridade de um caso RBC, melhorando, assim, sua efetividade.

Durante o processo de desenvolvimento de um sistema multiagente, a arquitetura de cada agente da sociedade é definida na fase de projeto detalhado, de forma a satisfazer os requisitos funcionais e não funcionais do sistema. Um dos produtos obtidos na pesquisa do GESEC é um processo baseado no conhecimento para o desenvolvimento de famílias de aplicações multiagentes denominado MADAE-PRO (COSTA, 2009). Em trabalhos futuros está prevista a incorporação do desenvolvimento de agentes RBC ao MADAE-PRO através da inclusão do agente CBRAA no processo.

REFERÊNCIAS

AAMODT, A.; PLAZA, E. "**Case-Based Reasoning: Foundation Issues, Methodological Variations, and System Approachs**". AICOM, 7, nº 1, 1994. p. 39-59.

ALLEN, J. **"Natural Language Understanding"**. Redwood City, CA: The Benjamin/Cummings Publishing Company, Inc, 1995.

BISHOP, C. "**Neural Networks for Pattern Recognition**". 1st Edition, New York, USA: Oxford University Press, Inc., 1995.

BRANQUINHO, J., MURCHO, D., GOMES, N.G., "**Enciclopédia de Termos Lógico-Filosóficos**", São Paulo: Martins Fontes, 2006.

BURKHARD, H.D., **"Extending some Concepts of CBR – Foundations of Case Retrieval Nets".** In M. Lenz et al. (eds.). Case-Based Reasoning Technology from Foundations to Applications, Springer-Verlarg, 1998, p. 17-50.

CARBONELL, J.G., "**Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition"**, In R.S. Michalski et al. (eds.): Machine Learning: An Artificial Intelligence Approach, Morgan Kaufmann, 1985.

COPI, I.M., COHEN, E.C., "**Introduction to Logic**", Library Of Congress Cataloging-In-Publication Data, 11th Edition, USA, 2001.

CORCHADO, J., LAZA E.R., "**Constructing Deliberative Agents with Case-Based Reasoning Technology**", em International Journal of Intelligent Systems. Vol. 18, Nº 12, 2012, p. 1227-1241.

COSTA, A. L., "**MADAE-Pro: Um processo baseado no conhecimento para Engenharia de Domínio e de Aplicações"**. Dissertação (Mestrado em Engenharia da Eletricidade – Área de Ciências da Computação) – Universidade Federal do Maranhão, São Luís, 2009.

DELLSCHAFT, K., STAAB, S. “**On How to Perform a Gold Standard Based Evaluation of Ontology Learning**”. In: International Semantic Web Conference, volume 4273 of Lecture Notes in Computer Science, Springer, 2006, p. 228-241.

DINIZ, M., **“Curso de Direito Civil Brasileiro. Volume 5: Direito de Família”**. 27ª Edição, 2012. Editora Saraiva. São Paulo.

DRUMOND, L., GIRARDI, R., SILVA, F., **“A Similarity Analysis Model for Semantic Web Information Filtering Applications”**, In Proceedings of the Twentieth International Conference on Software Engineering and Knowledge Engineering (SEKE 2008), Ed. Knowledge Systems Institute Graduate School, Redwood City, California, USA. 01 a 03 de julho de 2008, p. 638-642.

FARIA, C., GIRARDI, R., NOVAIS, P. **“Analysing the Problem and Main Approaches for Ontology Population”,** 10th International Conference on Information Technology : New Generations. Las Vegas, Nevada, USA, 15 a 17 de abril de 2013.

FAYYAD, U., PIATETSKY-SHAPIRO, G., SMYTH, P. (1996). **"From Data Mining to Knowledge Discovery in Databases"**. AI Magazine, Vol. 17, 1996, p. 37-54.

FINNIE, G., SUN, Z., **“Similarity and metrics in case-based reasoning”**, International Journal of Intelligent Systems, Vol 17, Nº 4, p. 2002, p. 273-287.

FYFE, C., CORCHADO, J.M., “**Automating the Construction of CBR Systems using Kernel Methods**”. International Journal of Intelligent Systems. Vol 16, Nº. 4, 2001.

GARRIDO, J.L., HURTADO, M.V., NOGUERA, M., ZURITA, J.M., “**Using a CBR Approach Based on Ontologies for Recommendation and Reuse of Knowledge Sharing in Decision Making**”, Hybrid Intelligent Systems, His '08. Eighth International Conference, 2008, p. 837-842.

GENNARI, J., MUSEN, M. A., FERGERSON, R. W. et al.. "**The Evolution of Protégé: an Environment for Knowledge-Based Systems Development.",** Technical Report SMI-2002-0943, 2002.

GENTNER, D., FORBUS, K.D., **"MAC/FAC: A Model of Similarity Based Retrieval",** Proceedings the 13th Annual Conference of the Cognitive Science Society, v.19, 1991, p. 141-205.

GICK, M. L., HOLYOAK, K.J., **Analogic Problem Solving**. Cognitive Psychology, 12, 1980.

GILMAN, E., SÁNCHEZ, I., SALORANTA, T., RIEKKI, J., **"Reasoning for Smart Space Application: Comparing Three Reasoning Engines CLIPS, Jess and Win-prolog.**" In: 2010 IEEE 10th International Conference on 2010 Computer and Information Technology (CIT), p. 1340–1345 (2010)

GIRARDI, R., LEITE, A., "**Knowledge Engineering Support for Agent-Oriented Software Reuse**" In: M. Ramachandran. (Org.). Knowledge Engineering for Software Development Life Cycles: Support Technologies and Applications. Hershey: IGI Global, v. I, 2011, p. 177-195.

GIRARDI, R., **An Analysis of the Contributions of the Agent Paradigm for the Development of Complex Systems**, In: Joint Meeting of the 5th Wolrd Multiconference on Systemics, Cybernetics and Informatics (SCI 2001) and the 7th Internacional Conference on Information Systems Analysis and Synthesis (ISAS 2001). Orlando, Florida, 2001.

GRUBER, T.R., "**Toward Principles for the Design of Ontologies used for Knowledge Sharing**", International Journal of Human-Computer Studies. nº 43, 1995, p. 907-928.

GUAN-YU, L., LI-NING, Q., SHI-PENG, L., "**Design and Realization of Case-Based Ontology Reasoning**", Wireless Communications, Networking and Mobile Computing - WICOM '08 - 4th International Conference, 2008, p. 01-04.

GUESSOUM, Z., “**Adaptive Agents and Multiagente Systems**”, IEEE Distributed Systems Online, Volume 5 Issue 7, IEEE Educational Activities Department Piscataway, NJ, USA, 2004, p. 04.

KALOUSIS, A., THEOHARIS, M., **"NOEMON: Design, Implementation and Performance Results of an Intelligent Assistant for Classification Selection"**, Intelligent Data Analysis, Elsevier, 1999.

KOLODNER, J.L., “**Improving Human Decision Making Through Case-Based Decision Aiding**”, AI Magazine Volume 12 Number 12, AAAI, 1991, p. 52-68.
KOLODNER, J.L., “**Reconstructive Memory: a Computer Model**”, Cognitive Science, 1983, 7, p. 281–328.

KOCK, G., **“The Neural Network Description Language CONNECT, and its C++ Implementation“**, Technical report, CMD FIRST Berlin, Universitat Politecnica de Catalunya, 1996.

KROVVIDY, S., WEE, W.C., **“Wastewater Treatment Systems from Case-Based Reasoning”**, Machine Learning, 10:341-346, 1993.

LEITE, A., GIRARDI, R., NOVAIS, P., **“Using Ontologies in Hybrid Software Agent Architectures”,** The 2013 IEEE/WIC/ACM International Conference on Intelligent Agent Technology. Atlanta, USA. 17 a 20 de novembro de 2013.

LENZ, M., BURKHARD, H., **Case Retrieval Nets: Basic Ideas and Extensions,** Ki '96 Proceedings of the 20th Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence, Springer-Verlag London, UK, 1996, p. 227-239.

LESS, B., CORCHADO, J., “**Case Based Reasoning In a Hybrid Agent-Oriented System**”, 5th German Workshop on Case-Based Reasoning, 1997.

LLOYD, L.W., “**Legal Reason: The Use of Analogy in Legal Argument**”. New York: Cambridge University Press, 2005, p.192. Hardback.

MANKTELOW, K., "**Reasoning and Thinking**", Psychology Press, 1999.

MATTOS, M., WANGENHEIM, C., PACHECO, R., **"Aplicação de Raciocínio Baseado em Casos na Fase de Análise de Requisitos para Construção de Abstrações em Lógica de Programação"**. In: SEMINCO – Seminário de Computação, Blumenau, 1999.

MILLER, G. A., FELLBAUM C., TENGI, R., WAKEFIELD, P., LANGONE, H., HASKELL, B. R. **Wordnet - A Lexical Database For The English Language**. 2006. Disponível Em <Http://Wordnet.Princeton.Edu/>. Acessado Em 10/08/2013.
MITCHELL, T.M., **"Machine Learning"**, McGraw-Hill, Inc. New York, NY, USA, 1997.

MUSTAPHA, N.B., ZGHAL, H.B., AUFARE, M., GHEZALA, H., **Semantic Search Using Modular Ontology Learning and Case-Based Reasoning**, Edbt '10 Proceedings Of The 2010 EDBT/ICDT Workshops, Article Nº. 3, ACM New York, NY, USA, 2010

NAGAIAH, M.D., "**Agent-Based CBR for Decision Support System"**, International Journal of Scientific & Engineering Research, Volume 2, Issue 1, January-2011.

NIU, Q., HU, L., "**Design of Case-Based Hybrid Agent Structure for Machine Tools of Intelligent Design System**," 2012 IEEE 3rd International Conference on Software Engineering and Service Science (ICSESS), 22-24 June 2012, p.59,62.

ONTAÑÓN, S., PLAZA, E., **"Collaborative case retention strategies for CBR agents"**, ICCBR'03, Proceedings the 5th international conference on Case-based reasoning: Reaseach and Development, Springer-Verlag, Berlin, Heidelberg, 2003.

PATTERSON, D.W., ROONEY, N., GALUSHKA, M., "**Efficient Similarity Determination and Case Construction Techniques for Case-Based Reasoning**", ECCBR, 2002, p. 292-305.

PORTER, B., BAREISS, R., **"PROTOS: An Experiment in Knowledge Acquisition for Heuristic Classification Tasks"**, In: Proceedings of the First International Meeting on Advances in Learning (IMAL), Les Arcs, France, 1986, p.159-174.

RAO, A.S., GEORGEFF, M.P., "**Modeling Rational Agents within a BDI-Architecture**", In J. Allen, R. Fikes, and E. Sandewall, editors, Proceedings the Second International Conference on Principles of Knowledge Representing and Reasoning. Morgan Kaufmann Publishers, San Mateo, CA, 1991.

REATEGUI, E., CAMPBELL, J.A., **"A Classification System for Credit Card Transactions"**, Procs. Second European Workshop on Case-Based Reasoning, 1994, p. 167-174.

RUSSEL, S., NORVIG, P., "**Inteligência Artificial**", Rio de Janeiro: Elsevier, 2004.

SCHANK, R. "**Dynamic Memory: A Theory of Learning in Computers and People**", New York: Cambridge University Press, 1982.

SILVA, F., GIRARDI, R., DRUMMOND, L., "**A Knowledge-Based Retrieval Model**", In: International Conference on Software Engineering and Knowledge Engineering, Boston: Knowledge System Institute Graduate School, 2009, p. 558 – 563.

SRINIVASAN, S., SINGH, J., KUMAR, V., "**Multi-Agent Based Decision Support System Using Data Mining and Case Based Reasoning**", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, Nº 2, Julho de 2011, p. 340.

STEEL, B., "**Chimera Agents for WIN-PROLOG"**. 2005. Disponível Em <http://www.lpa.co.uk/chi.htm>. Acessado Em 15/05/2013.

SUN, Z., HAN, J., DONG, D., **Five Perspectives on Case Based Reasoning**, ICIC '08 Proceedings of the 4th International Conference on Intelligent Computing: Advanced Intelligent Computing Theories and Applications - With Aspects of Artificial Intelligence, 2008, p. 410 – 419.

THAGARD, P., SHELLEY, E.C., “**Abductive Reasoning: Logic, Visual Thinking, and Coherence**”, In: M.L, 1997.

WANGENHEIM, C.; WANGENHEIM, A. **“Raciocínio Baseado em Casos”**, 1. ed. Barueri-SP: Manole, 2003.

WEISS, G., **“Multiagent systems: a modern approach to distributed artificial intelligence”**, MIT Press Cambridge, MA, USA, 1999.

WESS, S., ALTHOFF, K., DERWAND, G., **Using K-D Trees to Improve the Retrieval Step in Case-Based Reasoning**, EWCBR '93 Selected Papers from the First European Workshop on Topics in Case-Based Reasoning, Springer-Verlag London, UK ,1994, p. 167–181.

WOOLDRIDGE, M., “**An Introduction to Multiagent Systems**”, Chichester, UK, 2 edition, 2009.

WILKE, W., BERGMANN, R., **Techniques and Knowledge Used for Adaptation During Case-Based Problem Solving**, IEA/AIE '98 Proceedings the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Tasks and Methods in Applied Artificial Intelligence, Springer-Verlag, London, UK, 1998, p. 497-506.

ANEXO A – EXEMPLO DE PETIÇÃO JURÍDICA DE DIVÓRCIO

EXMO (A) SR (A). JUIZ (A) DE DIREITO DA 2ª VARA DA COMARCA DE ROSÁRIO (MA).

JUSTIÇA GRATUITA – LEI N. 1.060/50

Joao da Silva, brasileiro, maranhense, casado, professor, inscrito no RG sob nº1 e no CPF sob nº1, residente e domiciliado à Rua A e Maria da Silva, brasileira, maranhense, casada, do lar, inscrita no RG sob nº1 e no CPF sob nº1, residente e domiciliada à Rua A, cônjuges varão e varoa, respectivamente, pelo advogado e bastante procurador que constituíram (procuração em anexo), vêm, ajuizar **AÇÃO DE DIVÓRCIO CONSENSUAL**, com assentos nos fundamentos fáticos e jurídicos que doravante se expõe:

- I. DOS FATOS -

O casal contraiu matrimônio em 01 de janeiro de 2000, sendo lavrado o assento de matrimônio sob o Regime de Comunhão Parcial de Bens na 1ª Serventia Extrajudicial de Registro da Comarca de São Luís/MA, conforme se depreende da certidão apensa.

Dessa união adveio o nascimento dos menores Mariana Silva, nascida no dia 1 de outubro de 2001 (certidão de nascimento em anexo), e Joao Silva Filho, nascido no dia 1 de novembro de 2004 (certidão de nascimento em anexo).

Ocorre que, em razão das vicissitudes da vida, o matrimônio não mais subsiste plasmado pelo afeto necessário, de modo *more uxório*, pelo quê, em comum acordo, decidiram pôr fim ao vínculo conjugal, bem assim dispondo sobre a guarda dos filhos, alimentos, partilha de bens existentes e nome da cônjuge varoa.

- DA GUARDA E DO REGIME DE VISITAS –

De forma convencionada, no que toca à guarda dos filhos, os cônjuges acordam nos seguintes termos:

A guarda da menor Joao da Silva Filho ficará sob a responsabilidade de Maria Silva.

A guarda do menor Mariana Silva, ficará sob a responsabilidade de Maria Silva.

Ressalte-se que, a guarda acima disposta, será unilateral nos moldes do art. 1.583, e § 1º do Código Civil.

O regime de visitas será acordado oportunamente entre os Requerentes, de maneira a que cada um tenha direito à visita no período de férias escolares.

- DA PARTILHA DE BENS –

O casal, na constância da união, não adquiriu bens suscetíveis de partilha. Os Requerentes não possuem dívidas a serem saldadas.

- DOS ALIMENTOS –

O Requerente X pagará, a título de pensão alimentícia em favor de Y, o valor mensal de R$ 100,00 (cem reais), que será depositado em conta a ser criada e informada neste juízo, de responsabilidade da mãe X.

- DO NOME–

A Requerente deseja voltar a usar o nome de solteira ou seja, Maria Costa.

- II. DO DIREITO–

O direito ao divórcio está previsto constitucionalmente, no art. 226, §6º da Constituição Federal: "*O casamento civil pode ser dissolvido pelo divórcio*."

Hodiernamente, prescinde-se da prévia separação de direito ou de fato.

No caso em tela, assiste direito aos Requerentes que seja homologado o presente acordo de divórcio consensual.

- III. DO PEDIDO –

Em virtude do exposto, postula-se:

a) a concessão dos benefícios da Justiça Gratuita, nos termos da Lei 1.060/50, por não poderem arcar com as despesas deste processo sem grave prejuízo de seu sustento e de sua família;

b) **NO MÉRITO**, seja homologado o presente **DIVÓRCIO CONSENSUAL**, para que se ponha termo ao vínculo matrimonial que os une, ordenando-se a seguir, a expedição do competente mandado de averbação à margem do assento de casamento de ambos;

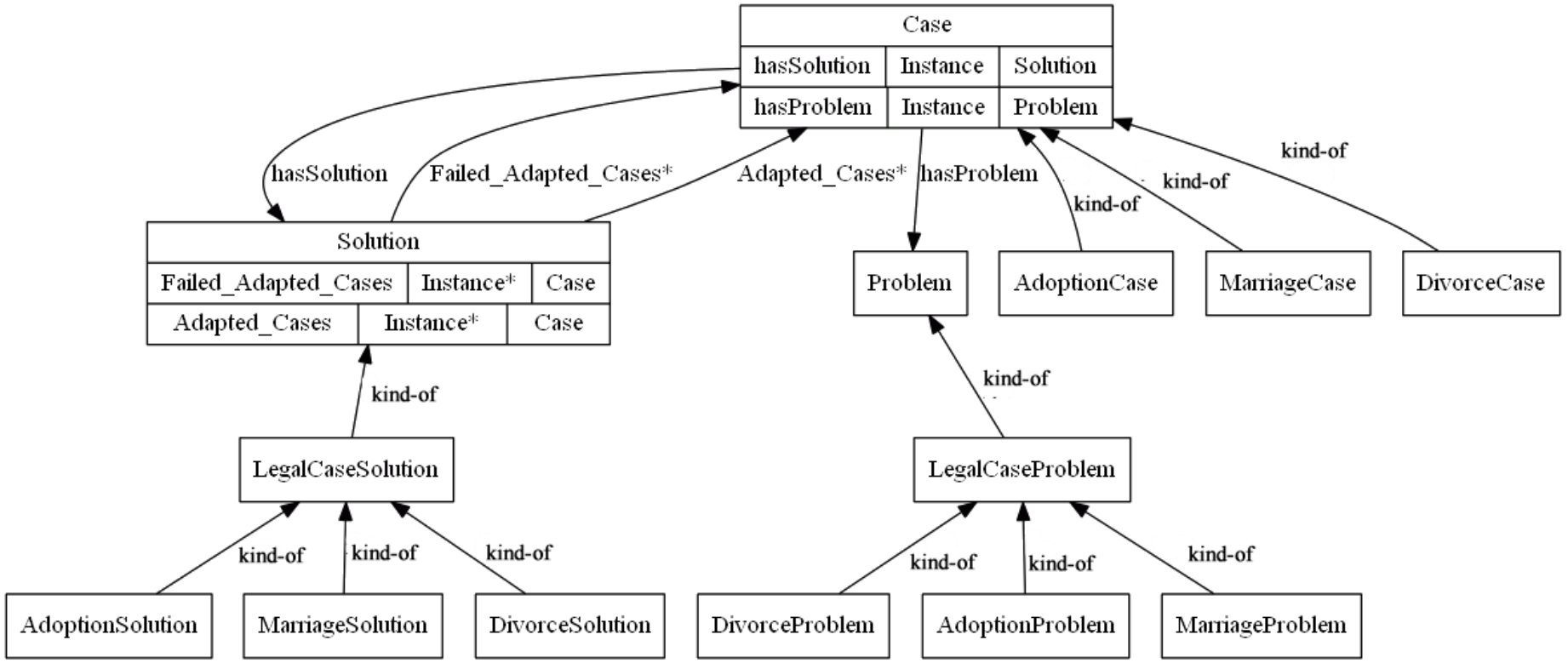
c) Requer-se, ainda, sejam deferidos todos os termos acordados entre as partes tal como acima explanados.

A fim de demonstrar o alegado, protesta-se a produção de todos os meios de provas, tais como depoimento das partes, oitiva de testemunhas, e juntada de novos documentos que se fizerem necessários.

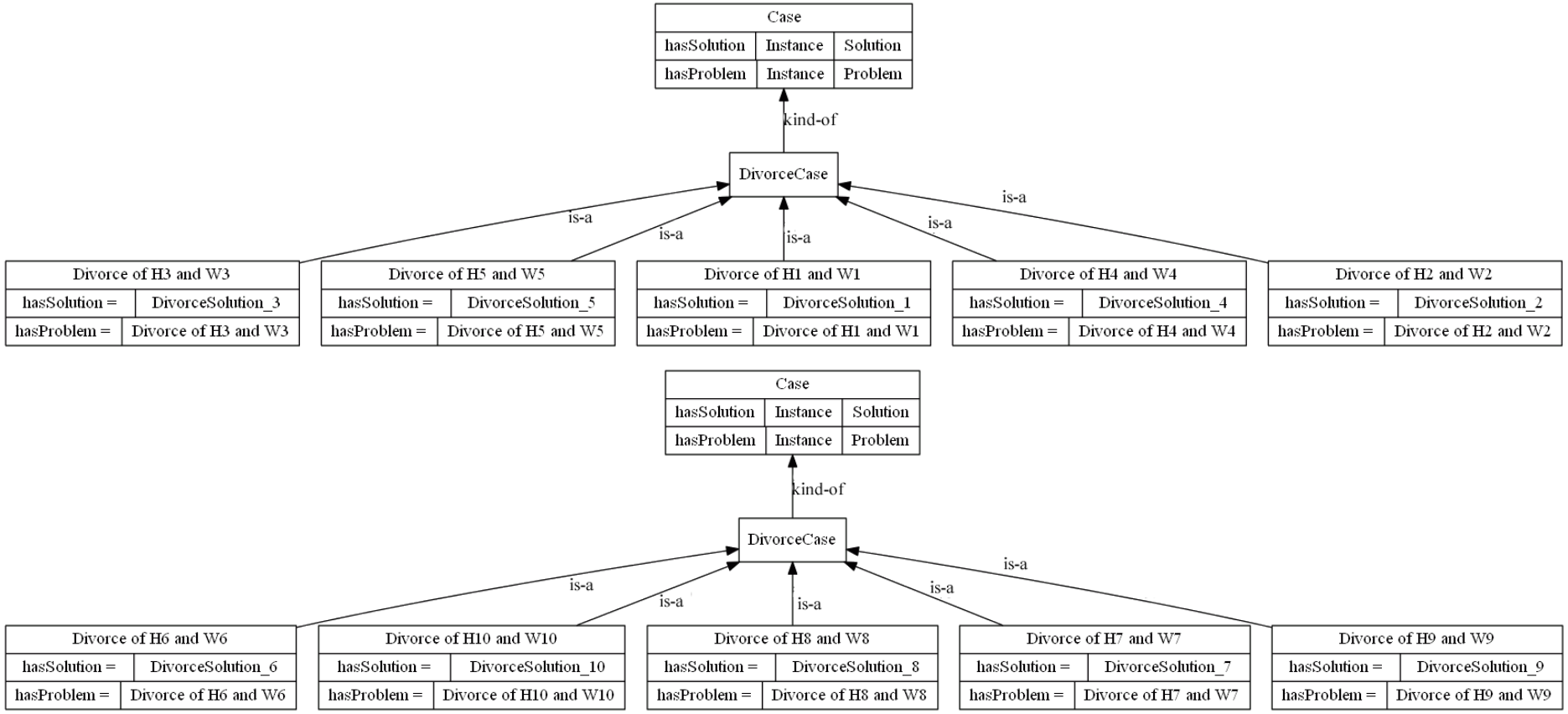
Dá-se à causa o valor de R$ 1.000,00 (mil reais).
Nestes termos,
Confia no deferimento.

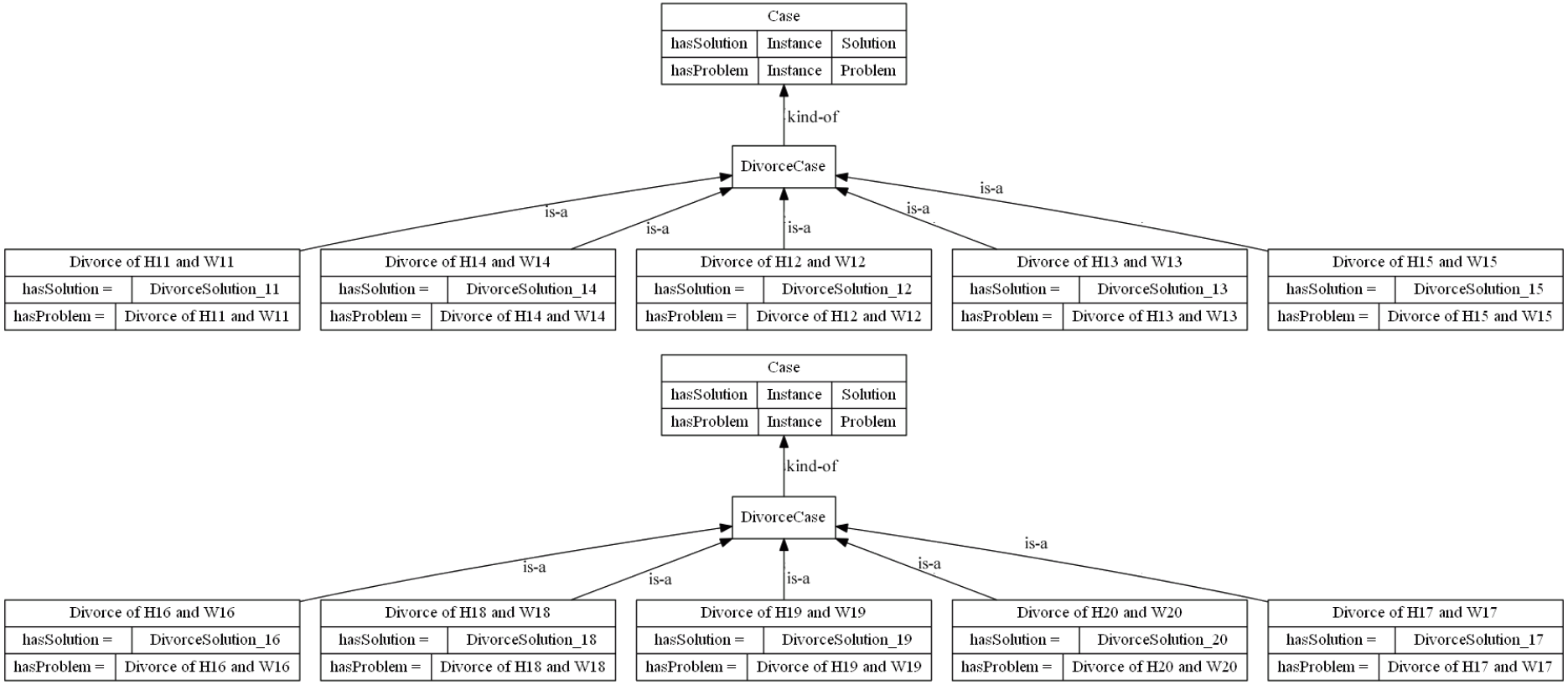
APÊNDICE A – ONTOLOGIAS DO AGENTE CBRAA UTILIZADAS NA AVALIAÇÃO

a) Ontologia de Casos RBC

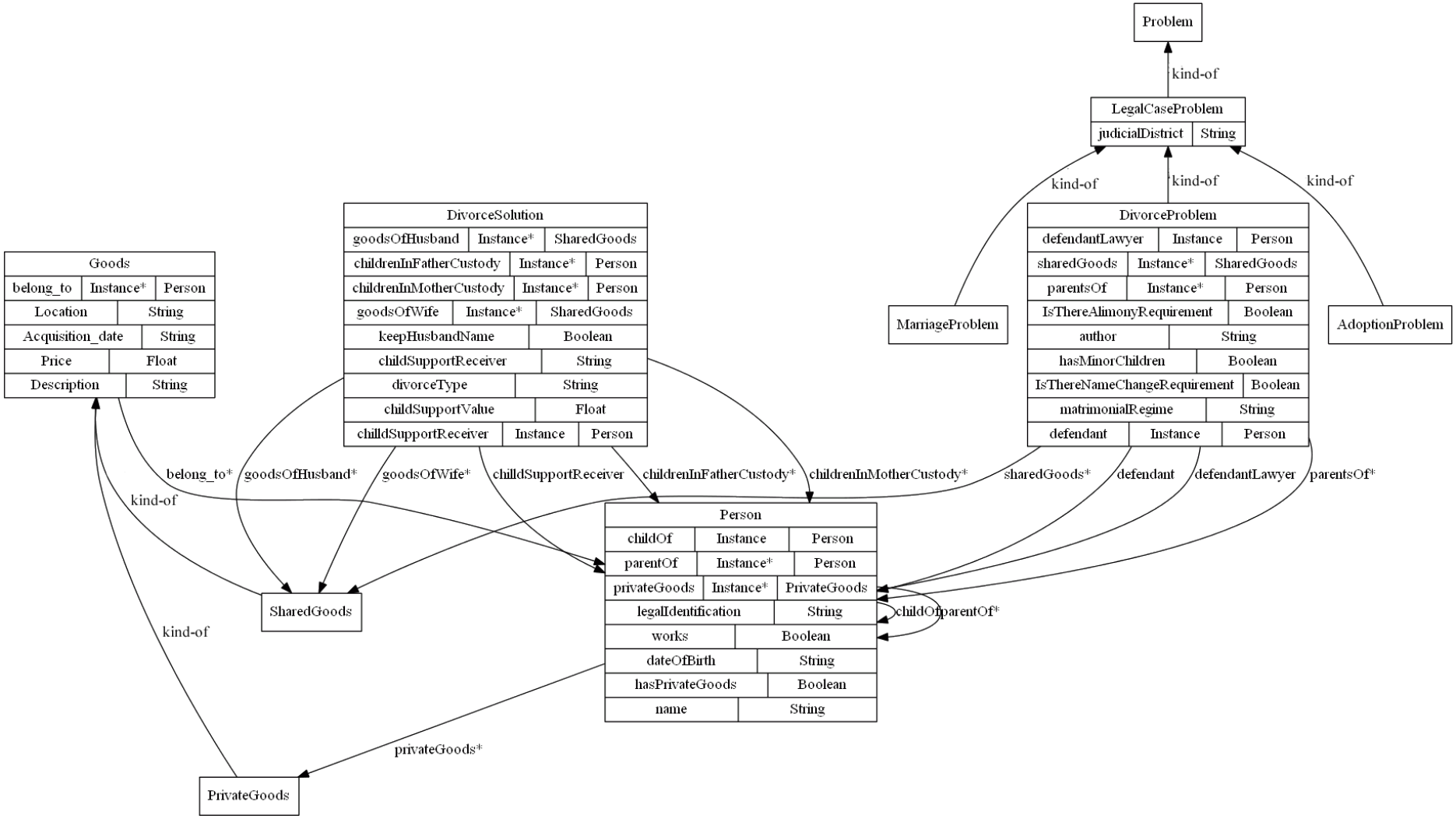


b) Instâncias dos Casos RBC de Divórcio (em grupos com cinco)

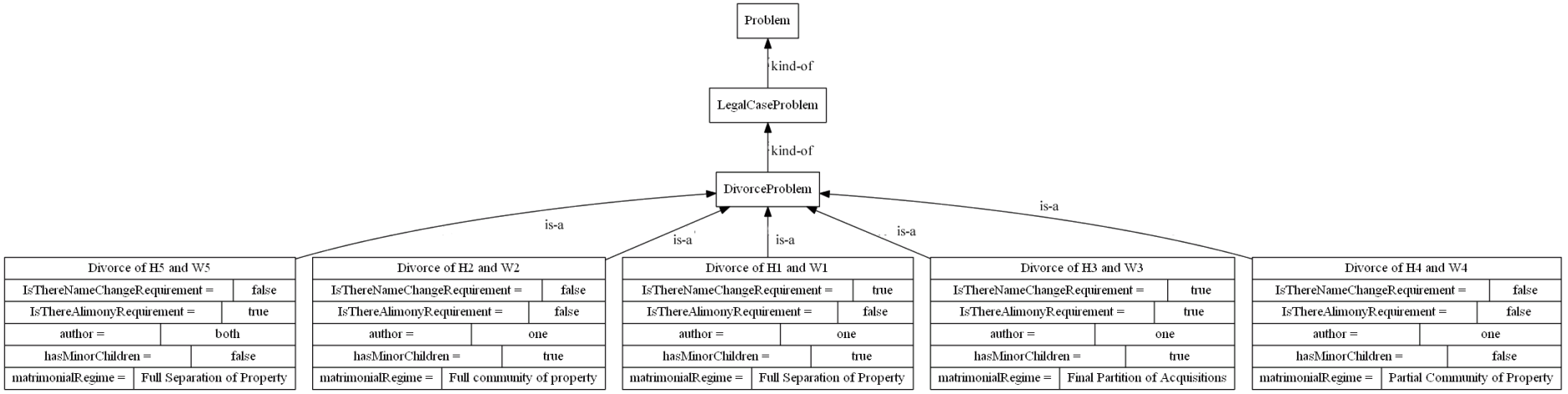


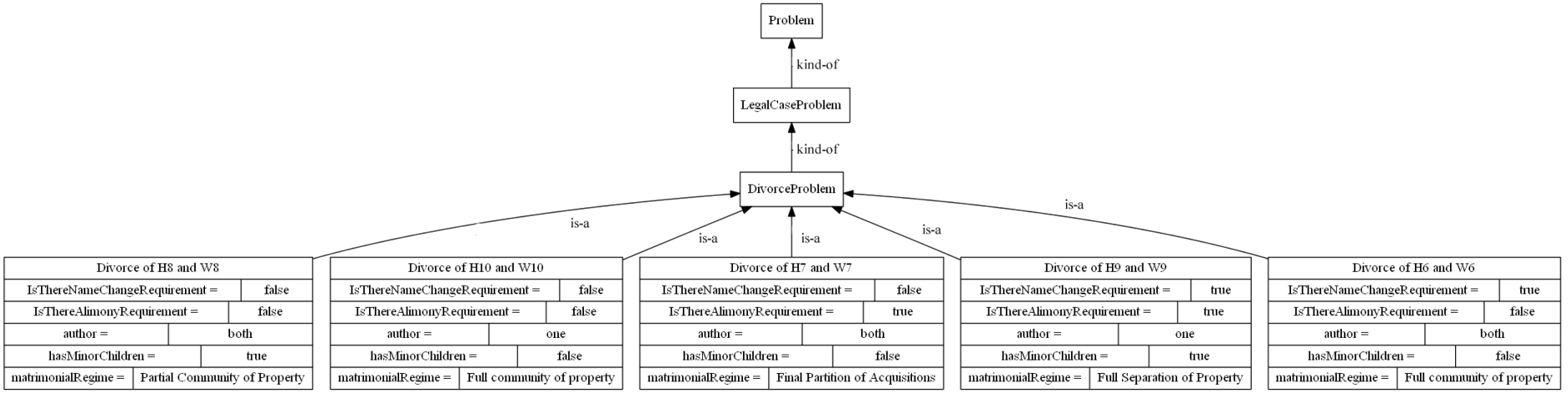

Case
hasSolution
Instance
Solution
hasProblem
Instance
Problem
kind-of
DivorceCase
is-a
is-a
is-a
is-a
is-a
Divorce of H11 and W11
hasSolution = DivorceSolution_11
hasProblem = Divorce of H11 and W11
Divorce of H14 and W14
hasSolution = DivorceSolution_14
hasProblem = Divorce of H14 and W14
Divorce of H12 and W12
hasSolution = DivorceSolution_12
hasProblem = Divorce of H12 and W12
Divorce of H13 and W13
hasSolution = DivorceSolution_13
hasProblem = Divorce of H13 and W13
Divorce of H15 and W15
hasSolution = DivorceSolution_15
hasProblem = Divorce of H15 and W15
Case
hasSolution
Instance
Solution
hasProblem
Instance
Problem
kind-of
DivorceCase
is-a
is-a
is-a
is-a
is-a
Divorce of H16 and W16
hasSolution = DivorceSolution_16
hasProblem = Divorce of H16 and W16
Divorce of H18 and W18
hasSolution = DivorceSolution_18
hasProblem = Divorce of H18 and W18
Divorce of H19 and W19
hasSolution = DivorceSolution_19
hasProblem = Divorce of H19 and W19
Divorce of H20 and W20
hasSolution = DivorceSolution_20
hasProblem = Divorce of H20 and W20
Divorce of H17 and W17
hasSolution = DivorceSolution_17
hasProblem = Divorce of H17 and W17

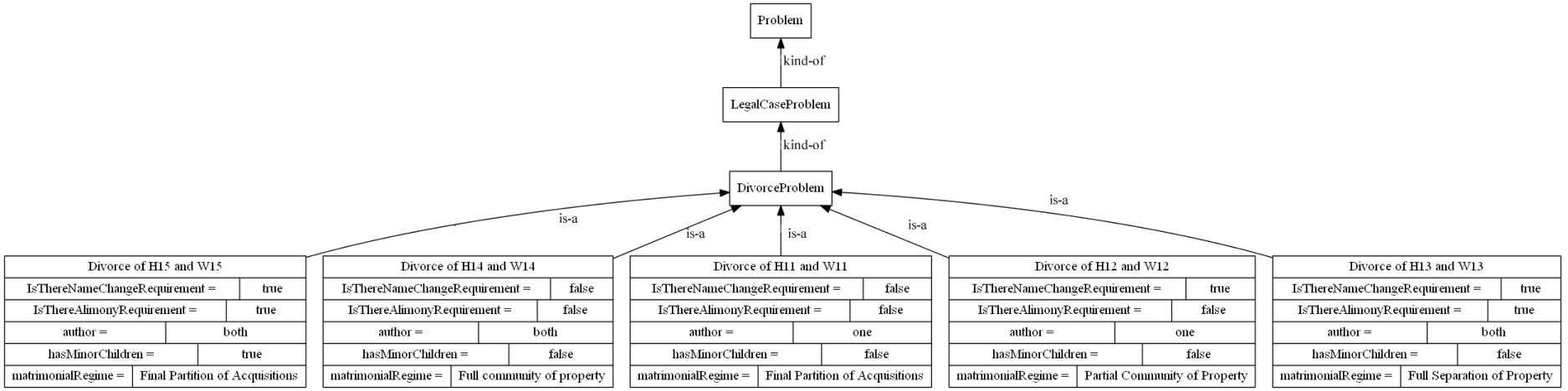
c) Hierarquia da classe *Problem*

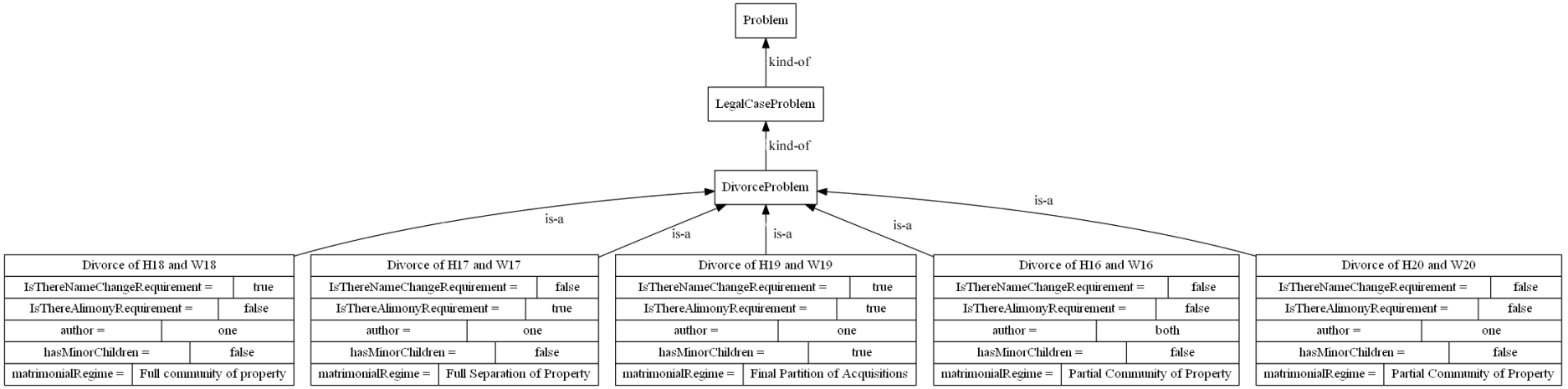


d) Instâncias de Problemas (em grupos com cinco)

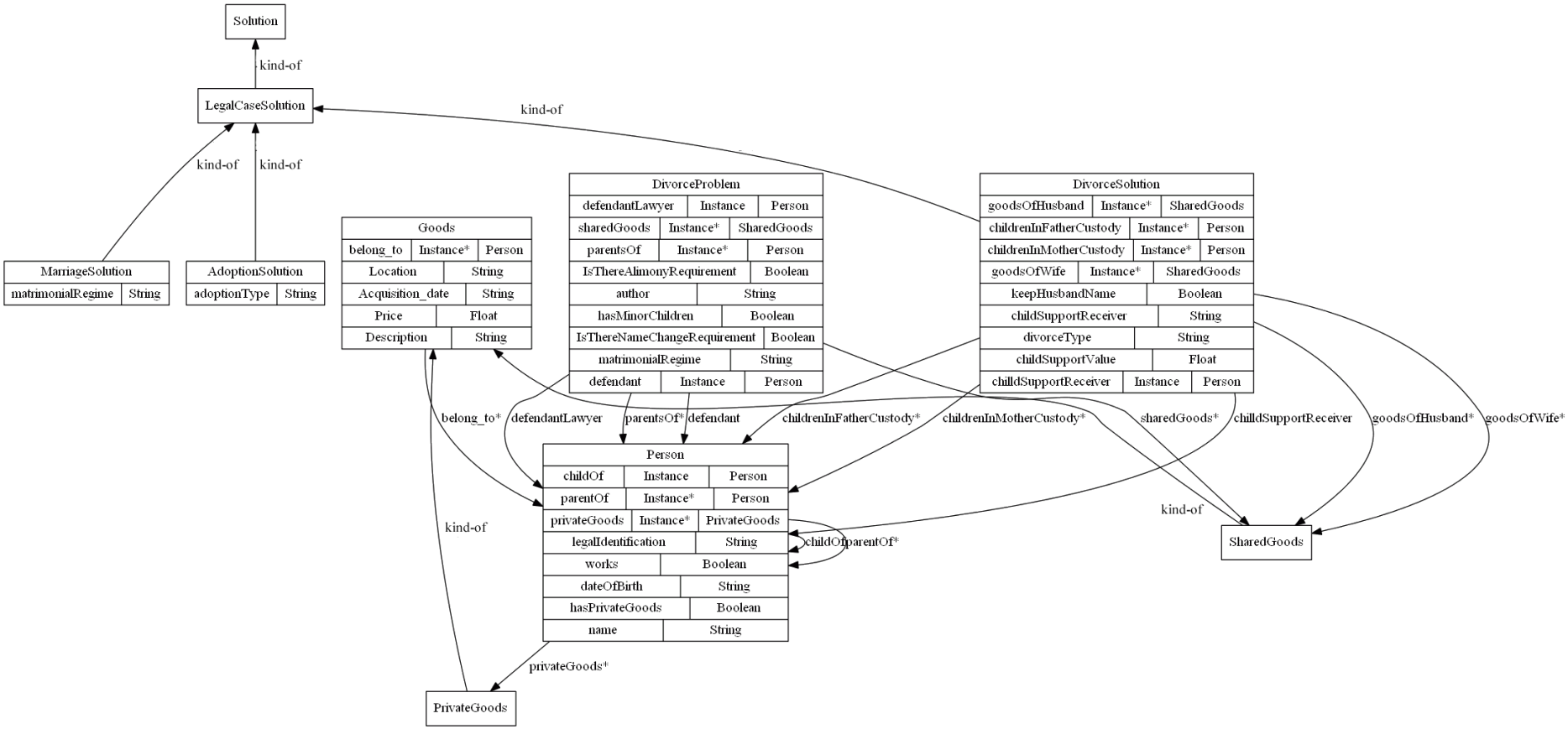




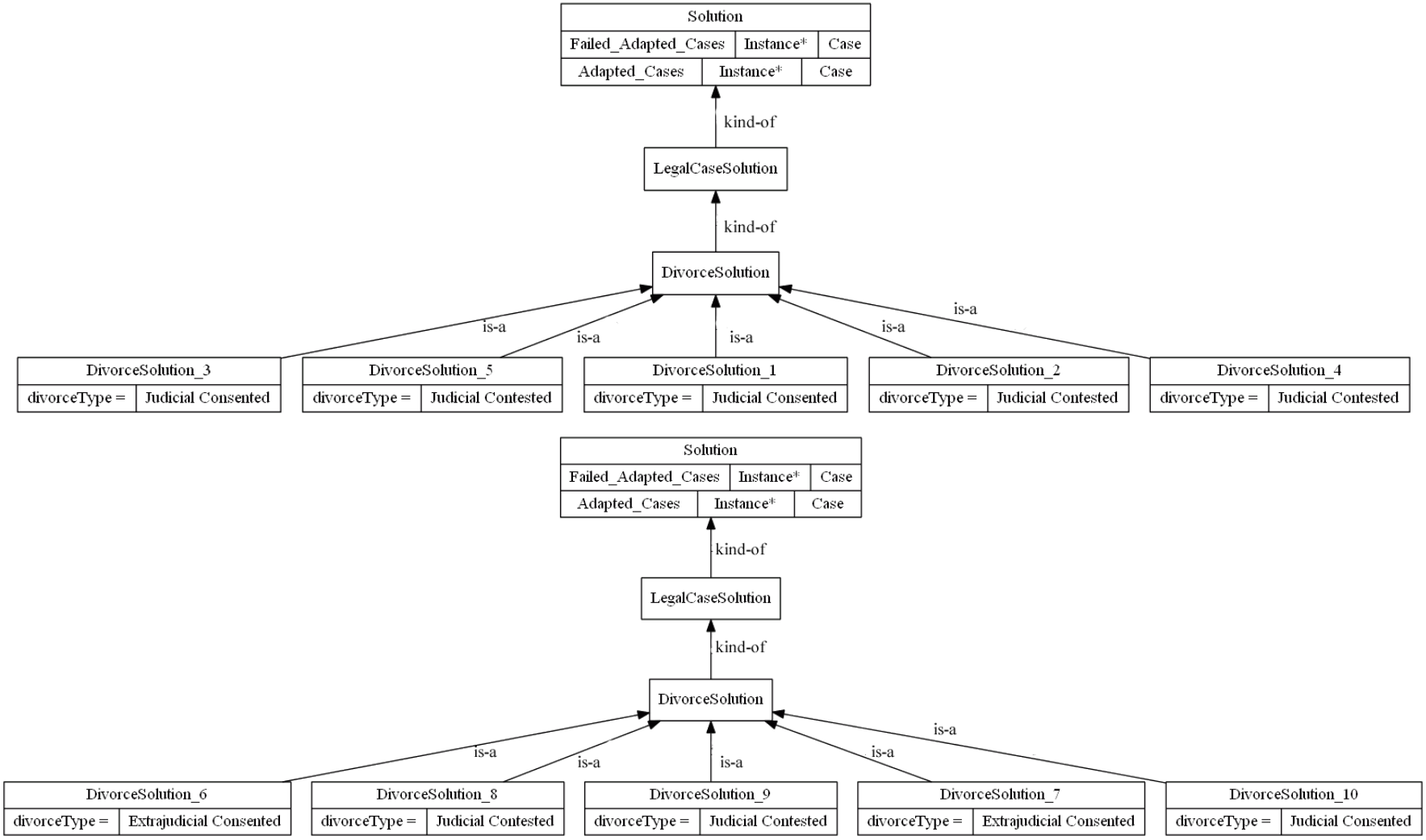

Problem
kind-of
LegalCaseProblem
kind-of
DivorceProblem
is-a
is-a
is-a
is-a
is-a
Divorce of H15 and W15
IsThereNameChangeRequirement = true
IsThereAlimonyRequirement = true
author = both
hasMinorChildren = true
matrimonialRegime = Final Partition of Acquisitions
Divorce of H14 and W14
IsThereNameChangeRequirement = false
IsThereAlimonyRequirement = false
author = both
hasMinorChildren = false
matrimonialRegime = Full community of property
Divorce of H11 and W11
IsThereNameChangeRequirement = false
IsThereAlimonyRequirement = false
author = one
hasMinorChildren = false
matrimonialRegime = Final Partition of Acquisitions
Divorce of H12 and W12
IsThereNameChangeRequirement = true
IsThereAlimonyRequirement = false
author = one
hasMinorChildren = false
matrimonialRegime = Partial Community of Property
Divorce of H13 and W13
IsThereNameChangeRequirement = true
IsThereAlimonyRequirement = true
author = both
hasMinorChildren = false
matrimonialRegime = Full Separation of Property

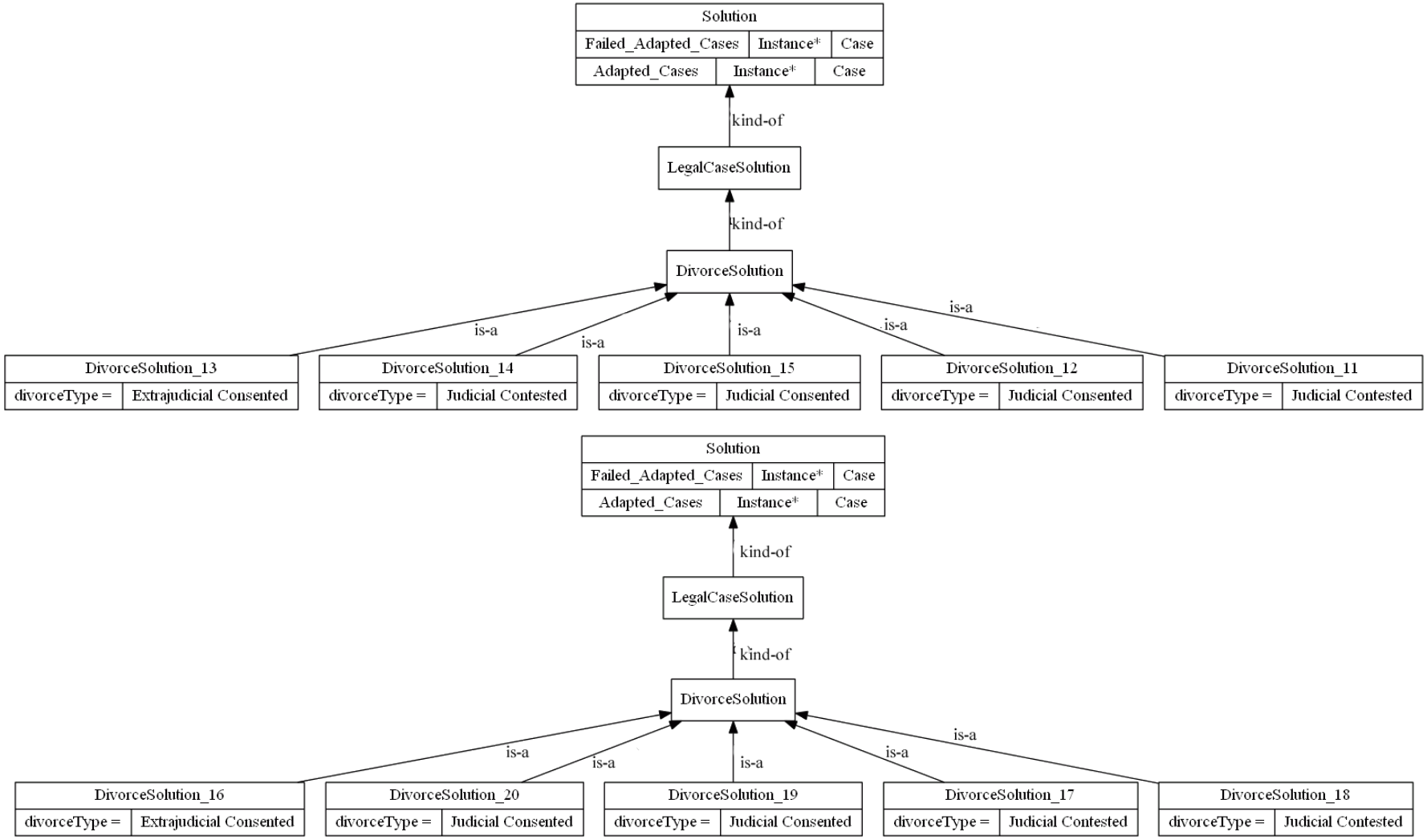

Problem
kind-of
LegalCaseProblem
kind-of
DivorceProblem
is-a
is-a
is-a
is-a
is-a
Divorce of H18 and W18
IsThereNameChangeRequirement = true
IsThereAlimonyRequirement = false
author = one
hasMinorChildren = false
matrimonialRegime = Full community of property
Divorce of H17 and W17
IsThereNameChangeRequirement = false
IsThereAlimonyRequirement = true
author = one
hasMinorChildren = false
matrimonialRegime = Full Separation of Property
Divorce of H19 and W19
IsThereNameChangeRequirement = true
IsThereAlimonyRequirement = true
author = one
hasMinorChildren = true
matrimonialRegime = Final Partition of Acquisitions
Divorce of H16 and W16
IsThereNameChangeRequirement = false
IsThereAlimonyRequirement = false
author = both
hasMinorChildren = false
matrimonialRegime = Partial Community of Property
Divorce of H20 and W20
IsThereNameChangeRequirement = false
IsThereAlimonyRequirement = false
author = one
hasMinorChildren = false
matrimonialRegime = Partial Community of Property

e) Hierarquia da Classe *Solution*

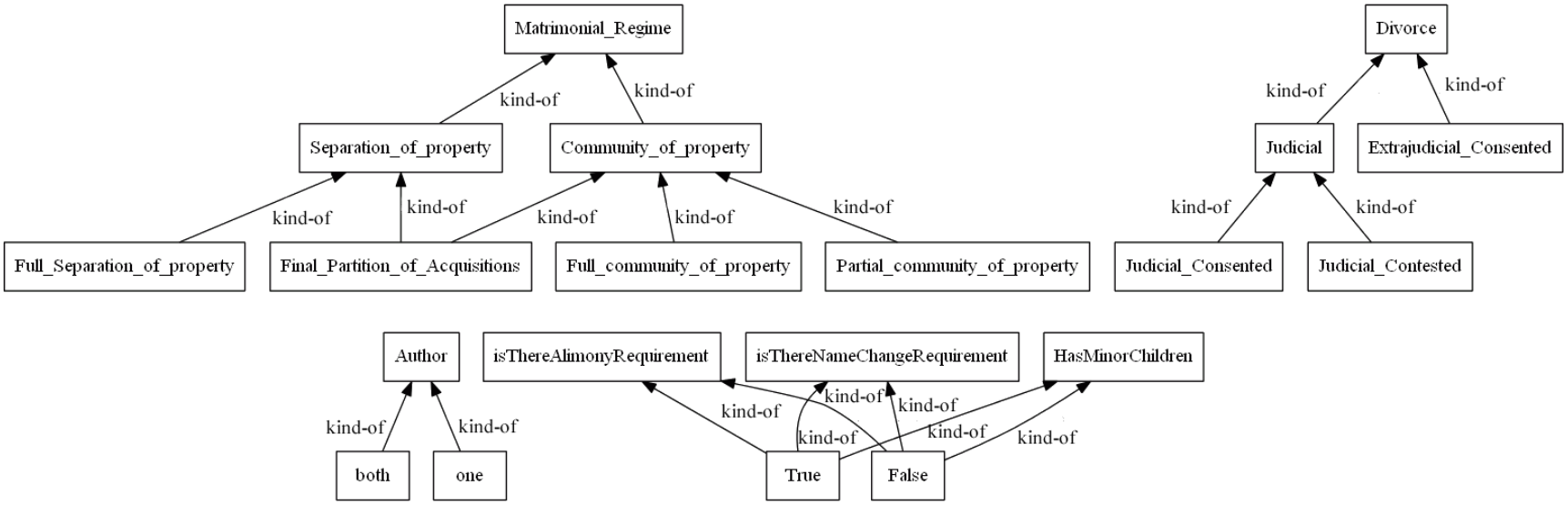


f) Instâncias de Soluções (em grupos com cinco)




Solution
Failed_Adapted_Cases
Instance*
Case
Adapted_Cases
Instance*
Case
kind-of
LegalCaseSolution
kind-of
DivorceSolution
is-a
is-a
is-a
is-a
is-a
DivorceSolution_13
divorceType =
Extrajudicial Consented
DivorceSolution_14
divorceType =
Judicial Contested
DivorceSolution_15
divorceType =
Judicial Consented
DivorceSolution_12
divorceType =
Judicial Consented
DivorceSolution_11
divorceType =
Judicial Contested
Solution
Failed_Adapted_Cases
Instance*
Case
Adapted_Cases
Instance*
Case
kind-of
LegalCaseSolution
kind-of
DivorceSolution
is-a
is-a
is-a
is-a
is-a
DivorceSolution_16
divorceType =
Extrajudicial Consented
DivorceSolution_20
divorceType =
Judicial Consented
DivorceSolution_19
divorceType =
Judicial Consented
DivorceSolution_17
divorceType =
Judicial Contested
DivorceSolution_18
divorceType =
Judicial Contested

g) Hierarquia de conceitos dos atributos de Divórcio



APÊNDICE B – CÓDIGO-FONTE EM PROLOG DO AGENTE CBRAA

a) Base de casos e medida de similaridade

```
/* Representação da Base de Casos */
/* Classificação de Casos */
ontology(legalCase,kindOf,case).
ontology(divorceCase,kindOf,legalCase).
ontology(marriageCase,kindOf,legalCase).
ontology(adoptionCase,kindOf,legalCase).

/* Classificação de Problema */
ontology(legalCaseProblem,kindOf,problem).
ontology(divorceProblem,kindOf,legalCaseProblem).
ontology(marriageProblem,kindOf,legalCaseProblem).
ontology(adoptionProblem,kindOf,legalCaseProblem).

/* Classificação de Solução */
ontology(legalCaseSolution,kindOf,solution).
ontology(divorceSolution,kindOf,legalCaseSolution).
ontology(marriageSolution,kindOf,legalCaseSolution).
ontology(adoptionSolution,kindOf,legalCaseSolution).

/*Relacionamentos Não-Taxonômicos de Caso */
ontology(case,hasRelation,hasProblem).
ontology(case,hasRelation,hasSolution).

/* para identificação dos tipos de casos */
caseproblem(divorceCase,divorceProblem).
caseproblem(adoptionCase,adoptionProblem).
caseproblem(marriageCase,marriageProblem).

/*Propriedades e Relacionamentos Não-Taxonômicos de Problema de Divorcio
(Atributos Relevantes) */
ontology(divorceProblem,hasRelation,matrimonialRegime).
ontology(divorceProblem,hasRelation,author).
ontology(divorceProblem,hasRelation,hasMinorChildren).
ontology(divorceProblem,hasRelation,isThereNameChangeRequirement).
ontology(divorceProblem,hasRelation,isThereAlimonyRequirement).

caseproblem(adoptionCase,adoptionProblem).
caseproblem(divorceCase,divorceProblem).
caseproblem(marriageCase,marriageProblem).

/* Representação dos Casos da Base de Conhecimento (Casos) - Instâncias */
/* CPR1 */
ontology(cpr1,isA,divorceCase).
ontology(cpr1,hasProblem,cpr1Problem).
ontology(cpr1Problem,isA,divorceProblem).
ontology(cpr1Problem,matrimonialRegime,`fullSeparationOfProperty`).
ontology(cpr1Problem,author,`one`).
ontology(cpr1Problem,hasMinorChildren,`yes`).
ontology(cpr1Problem,isThereNameChangeRequirement,`no`).
ontology(cpr1Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr1,hasSolution,cpr1Solution).
ontology(cpr1Solution,divorceType,judicialConsented).
```

```
/* CPR2 */
ontology(cpr2,isA,divorceCase).
ontology(cpr2,hasProblem,cpr2Problem).
ontology(cpr2Problem,isA,divorceProblem).
ontology(cpr2Problem,matrimonialRegime,`fullCommunityOfProperty`).
ontology(cpr2Problem,author,`one`).
ontology(cpr2Problem,hasMinorChildren,`yes`).
ontology(cpr2Problem,isThereNameChangeRequirement,`no`).
ontology(cpr2Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr2,hasSolution,cpr2Solution).
ontology(cpr2Solution,divorceType,judicialContested).

/* CPR3 */
ontology(cpr3,isA,divorceCase).
ontology(cpr3,hasProblem,cpr3Problem).
ontology(cpr3Problem,isA,divorceProblem).
ontology(cpr3Problem,matrimonialRegime,`finalPartitionOfAcquisitions`).
ontology(cpr3Problem,author,`one`).
ontology(cpr3Problem,hasMinorChildren,`yes`).
ontology(cpr3Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr3Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr3,hasSolution,cpr3Solution).
ontology(cpr3Solution,divorceType,judicialConsented).

/* CPR4 */
ontology(cpr4,isA,divorceCase).
ontology(cpr4,hasProblem,cpr4Problem).
      ontology(cpr4Problem,isA,divorceProblem).
ontology(cpr4Problem,matrimonialRegime,`partialCommunityOfProperty`).
ontology(cpr4Problem,author,`one`).
ontology(cpr4Problem,hasMinorChildren,`no`).
ontology(cpr4Problem,isThereNameChangeRequirement,`no`).
ontology(cpr4Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr4,hasSolution,cpr4Solution).
ontology(cpr4Solution,divorceType,judicialContested).

/* CPR5 */
ontology(cpr5,isA,divorceCase).
ontology(cpr5,hasProblem,cpr5Problem).
ontology(cpr5Problem,isA,divorceProblem).
ontology(cpr5Problem,matrimonialRegime,`fullSeparationOfProperty`).
ontology(cpr5Problem,author,`both`).
ontology(cpr5Problem,hasMinorChildren,`no`).
ontology(cpr5Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr5Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr5,hasSolution,cpr5Solution).
ontology(cpr5Solution,divorceType,judicialContested).

/* CPR6 */
ontology(cpr6,isA,divorceCase).
ontology(cpr6,hasProblem,cpr6Problem).
ontology(cpr6Problem,isA,divorceProblem).
ontology(cpr6Problem,matrimonialRegime,`fullCommunityOfProperty`).
ontology(cpr6Problem,author,`both`).
ontology(cpr6Problem,hasMinorChildren,`no`).
ontology(cpr6Problem,isThereNameChangeRequirement,`no`).
ontology(cpr6Problem,isThereAlimonyRequirement,`yes`).
```

```
%solution
ontology(cpr6,hasSolution,cpr6Solution).
ontology(cpr6Solution,divorceType,extrajudicialConsented).

/* CPR7 */
ontology(cpr7,isA,divorceCase).
ontology(cpr7,hasProblem,cpr7Problem).
ontology(cpr7Problem,isA,divorceProblem).
ontology(cpr7Problem,matrimonialRegime,`finalPartitionOfAcquisitions`).
ontology(cpr7Problem,author,`both`).
ontology(cpr7Problem,hasMinorChildren,`no`).
ontology(cpr7Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr7Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr7,hasSolution,cpr7Solution).
ontology(cpr7Solution,divorceType,extrajudicialConsented).

/* CPR8 */
ontology(cpr8,isA,divorceCase).
ontology(cpr8,hasProblem,cpr8Problem).
ontology(cpr8Problem,isA,divorceProblem).
ontology(cpr8Problem,matrimonialRegime,`partialCommunityOfProperty`).
ontology(cpr8Problem,author,`both`).
ontology(cpr8Problem,hasMinorChildren,`yes`).
ontology(cpr8Problem,isThereNameChangeRequirement,`no`).
ontology(cpr8Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr8,hasSolution,cpr8Solution).
ontology(cpr8Solution,divorceType,judicialContested).

/* CPR9 */
ontology(cpr9,isA,divorceCase).
ontology(cpr9,hasProblem,cpr9Problem).
ontology(cpr9Problem,isA,divorceProblem).
ontology(cpr9Problem,matrimonialRegime,`fullSeparationOfProperty`).
ontology(cpr9Problem,author,`one`).
ontology(cpr9Problem,hasMinorChildren,`yes`).
ontology(cpr9Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr9Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr9,hasSolution,cpr9Solution).
ontology(cpr9Solution,divorceType,judicialContested).

/* CPR10 */
ontology(cpr10,isA,divorceCase).
ontology(cpr10,hasProblem,cpr10Problem).
ontology(cpr10Problem,isA,divorceProblem).
ontology(cpr10Problem,matrimonialRegime,`fullCommunityOfProperty`).
ontology(cpr10Problem,author,`one`).
ontology(cpr10Problem,hasMinorChildren,`no`).
ontology(cpr10Problem,isThereNameChangeRequirement,`no`).
ontology(cpr10Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr10,hasSolution,cpr10Solution).
ontology(cpr10Solution,divorceType,judicialConsented).

/* CPR11 */
ontology(cpr11,isA,divorceCase).
ontology(cpr11,hasProblem,cpr11Problem).
ontology(cpr11Problem,isA,divorceProblem).
ontology(cpr11Problem,matrimonialRegime,`finalPartitionOfAcquisitions`).
```

```
ontology(cpr11Problem,author,`one`).
ontology(cpr11Problem,hasMinorChildren,`no`).
ontology(cpr11Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr11Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr11,hasSolution,cpr11Solution).
ontology(cpr11Solution,divorceType,judicialContested).

/* CPR12 */
ontology(cpr12,isA,divorceCase).
ontology(cpr12,hasProblem,cpr12Problem).
ontology(cpr12Problem,isA,divorceProblem).
ontology(cpr12Problem,matrimonialRegime,`partialCommunityOfProperty`).
ontology(cpr12Problem,author,`one`).
ontology(cpr12Problem,hasMinorChildren,`no`).
ontology(cpr12Problem,isThereNameChangeRequirement,`no`).
ontology(cpr12Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr12,hasSolution,cpr12Solution).
ontology(cpr12Solution,divorceType,judicialConsented).

/* CPR13 */
ontology(cpr13,isA,divorceCase).
ontology(cpr13,hasProblem,cpr13Problem).
ontology(cpr13Problem,isA,divorceProblem).
ontology(cpr13Problem,matrimonialRegime,`fullSeparationOfProperty`).
ontology(cpr13Problem,author,`both`).
ontology(cpr13Problem,hasMinorChildren,`no`).
ontology(cpr13Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr13Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr13,hasSolution,cpr13Solution).
ontology(cpr13Solution,divorceType,extrajudicialConsented).

/* CPR14 */
ontology(cpr14,isA,divorceCase).
ontology(cpr14,hasProblem,cpr14Problem).
ontology(cpr14Problem,isA,divorceProblem).
ontology(cpr14Problem,matrimonialRegime,`fullCommunityOfProperty`).
ontology(cpr14Problem,author,`both`).
ontology(cpr14Problem,hasMinorChildren,`yes`).
ontology(cpr14Problem,isThereNameChangeRequirement,`no`).
ontology(cpr14Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr14,hasSolution,cpr14Solution).
ontology(cpr14Solution,divorceType,judicialContested).

/* CPR15 */
ontology(cpr15,isA,divorceCase).
ontology(cpr15,hasProblem,cpr15Problem).
ontology(cpr15Problem,isA,divorceProblem).
ontology(cpr15Problem,matrimonialRegime,`finalPartitionOfAcquisitions`).
ontology(cpr15Problem,author,`both`).
ontology(cpr15Problem,hasMinorChildren,`yes`).
ontology(cpr15Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr15Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr15,hasSolution,cpr15Solution).
ontology(cpr15Solution,divorceType,judicialConsented).

/* CPR16 */
```

```
ontology(cpr16,isA,divorceCase).
ontology(cpr16,hasProblem,cpr16Problem).
ontology(cpr16Problem,isA,divorceProblem).
ontology(cpr16Problem,matrimonialRegime,`partialCommunityOfProperty`).
ontology(cpr16Problem,author,`both`).
ontology(cpr16Problem,hasMinorChildren,`no`).
ontology(cpr16Problem,isThereNameChangeRequirement,`no`).
ontology(cpr16Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr16,hasSolution,cpr16Solution).
ontology(cpr16Solution,divorceType,extrajudicialConsented).

/* CPR17 */
ontology(cpr17,isA,divorceCase).
ontology(cpr17,hasProblem,cpr17Problem).
ontology(cpr17Problem,isA,divorceProblem).
ontology(cpr17Problem,matrimonialRegime,`fullSeparationOfProperty`).
ontology(cpr17Problem,author,`one`).
ontology(cpr17Problem,hasMinorChildren,`no`).
ontology(cpr17Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr17Problem,isThereAlimonyRequirement,`no`).
%solution
ontology(cpr17,hasSolution,cpr17Solution).
ontology(cpr17Solution,divorceType,judicialContested).

/* CPR18 */
ontology(cpr18,isA,divorceCase).
ontology(cpr18,hasProblem,cpr18Problem).
ontology(cpr18Problem,isA,divorceProblem).
ontology(cpr18Problem,matrimonialRegime,`fullCommunityOfProperty`).
ontology(cpr18Problem,author,`one`).
ontology(cpr18Problem,hasMinorChildren,`no`).
ontology(cpr18Problem,isThereNameChangeRequirement,`no`).
ontology(cpr18Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr18,hasSolution,cpr18Solution).
ontology(cpr18Solution,divorceType,judicialContested).

/* CPR19 */
ontology(cpr19,isA,divorceCase).
ontology(cpr19,hasProblem,cpr19Problem).
ontology(cpr19Problem,isA,divorceProblem).
ontology(cpr19Problem,matrimonialRegime,`finalPartitionOfAcquisitions`).
ontology(cpr19Problem,author,`one`).
ontology(cpr19Problem,hasMinorChildren,`yes`).
ontology(cpr19Problem,isThereNameChangeRequirement,`yes`).
ontology(cpr19Problem,isThereAlimonyRequirement,`yes`).
%solution
ontology(cpr19,hasSolution,cpr19Solution).
ontology(cpr19Solution,divorceType,judicialConsented).

/* CPR20 */
ontology(cpr20,isA,divorceCase).
ontology(cpr20,hasProblem,cpr20Problem).
ontology(cpr20Problem,isA,divorceProblem).
ontology(cpr20Problem,matrimonialRegime,`partialCommunityOfProperty`).
ontology(cpr20Problem,author,`one`).
ontology(cpr20Problem,hasMinorChildren,`no`).
ontology(cpr20Problem,isThereNameChangeRequirement,`no`).
ontology(cpr20Problem,isThereAlimonyRequirement,`no`).
%solution
```

```
ontology(cpr20,hasSolution,cpr20Solution).
ontology(cpr20Solution,divorceType,judicialConsented).

/* Hierarquia de conceitos de matrimonialRegime */
ontology(`communityOfProperty`,kindOf,`matrimonialRegime`).
ontology(`separationOfProperty`,kindOf,`matrimonialRegime`).
ontology(`fullCommunityOfProperty`,kindOf,`communityOfProperty`).
ontology(`partialCommunityOfProperty`,kindOf,`communityOfProperty`).
ontology(`fullSeparationOfProperty`,kindOf,`separationOfProperty`).
ontology(`finalPartitionOfAcquisitions`,kindOf,`fullSeparationOfProperty`).
ontology(`finalPartitionOfAcquisitions`,kindOf,`partialCommunityOfProperty`
).

/* Hierarquia de conceitos de hasMinorChildren */
ontology(`one`,kindOf,`author`).
ontology(`both`,kindOf,`author`).

/* Hierarquia de conceitos de isThereAlimonyRequirement */
ontology(`yes`,kindOf,`isThereAlimonyRequirement`).
ontology(`no`,kindOf,`isThereAlimonyRequirement`).

/* Hierarquia de conceitos de isThereNameChangeRequirement */
ontology(`yes`,kindOf,`isThereNameChangeRequirement`).
ontology(`no`,kindOf,`isThereNameChangeRequirement`).

/* Hierarquia de conceitos de hasMinorChildren */
ontology(`yes`,kindOf,`hasMinorChildren`).
ontology(`no`,kindOf,`hasMinorChildren`).

/* peso dos atributos */
weight(matrimonialRegime,1.0).
weight(author,1.0).
weight(hasMinorChildren,1.0).
weight(isThereNameChangeRequirement,1.0).
weight(isThereAlimonyRequirement,1.0).

weight(aa,1.0).
weight(bb,1.0).

/* Conjunto dos Conceitos - Hierarquia de Superconceitos) */
hierarchy_up(Base,Next) :- ontology(Base,kindOf,Next).
hierarchy_up(Base,End) :- ontology(Base,kindOf,Next),
hierarchy_up(Next,End).

list_hierarchy_up(G,L) :- findall(X,hierarchy_up(G,X),[]),
append([G],[],L), !.
list_hierarchy_up(G,LLLL) :- setof(X,hierarchy_up(G,X),L2),
append([G],L2,L), findall(Z,ontology(_,hasRelation,Z),LLL),
deletelist(L,LLL,LLLL).

/* Conjunto dos Conceitos - Hierarquia de Subconceitos */
hierarchy_down(C1,C2) :- ontology(C2,kindOf,C1).
hierarchy_down(C1,CF) :- ontology(Next,kindOf,C1), hierarchy_down(Next,CF).

list_hierarchy_down(G,L) :- findall(X,hierarchy_down(G,X),[]),
append([G],[],L), !.
list_hierarchy_down(G,L) :- setof(X,hierarchy_down(G,X),L2),
append([],L2,L).

%remover elementos de uma lista de outra lista
deletelist([], _, []).
```

```
deletelist([X|Xs], Y, Z) :- atom_string(X1,X), member(X1, Y),
deletelist(Xs, Y, Z), !.
deletelist([X|Xs], Y, [X|Zs]) :- deletelist(Xs, Y, Zs).

%quantidade de elementos de uma lista
qtde([],0).
qtde([_|T],S) :- qtde(T,G),S is 1+G.

%findall recursivo para executar a partir de uma lista - one relation
ontology_list(Z,Relation,Y) :- member(X,Z), ontology(X, Relation, Y).
rec_findall(A,Relation,Lista) :- findall(Y, ontology_list(A, Relation, Y),
Lista), !.

%findall recursivo para executar a partir de uma lista - list of relation
ontology_list_2(X,Relation,Y) :- member(Rel,Relation), ontology(X, Rel, Y).
rec_findall_2(A,Relation,Lista) :- findall(Y, ontology_list_2(A, Relation,
Y), Lista), !.

%findall recursivo para executar a partir de uma lista - pesos
weight2(X,Y) :- member(Rel,X), weight(Rel, Y).
rec_findall_3(A,Lista) :- findall(Y, weight2(A, Y), Lista), !.

%interseção de duas listas
intersec([X | Y], L, [X |Z]) :- member(X, L), intersec(Y, L, Z).
intersec([_ |X], L, Y) :- intersec(X, L, Y).
intersec(_, _, []).

%tamanho da intersecao de duas listas
interseclength(Lista1, Lista2, Tamanho) :-
intersec(Lista1,Lista2,C),length(C,Tamanho).

%testa se é lista
isList([_|_]).
isList([_|Tail]) :- isList(Tail).

%calcula a similaridade entre dois conceitos
sim_attribute(Att1,Att2,W,S) :-
list_hierarchy_up(Att1,List1),list_hierarchy_up(Att2,List2),
interseclength(List1,List2,IntersectLenght), length(List1,L1),
number_string(W1,W), length(List2,L2), S is ((2 * IntersectLenght) / (L1 +
L2)) * W1, !.

%soma todos os elementos de uma lista
soma([],0).
soma([H|T],S):-soma(T,G),S is H+G.

%soma lista de elementos numéricos em uma lista de strings
soma_converte([],0).
soma_converte([H|T],S):- soma_converte(T,G), number_string(H1,H), S is
H1+G.

%calcula a similaridade entre duas listas de atributos
sim_attribute2([],[],_,0) :- !.
sim_attribute2(A,B,W,C) :- not(isList(A)),sim_attribute(A,B,W,C), !.
sim_attribute2(A,B,W,G) :- head(A,L), head(B,N), head(W,WH),
sim_attribute(L,N,WH,P), tail(A,J), tail(B,U), tail(W,WT),
sim_attribute2(J,U,WT,O), G is P + O.

%obtem cabeça da lista
head([A|B],X) :- X = A.
```

```
%obtem calda da lista
tail([A|B],X) :- X = B.

%arredonda valores numericos
round( Number, Places, Rounded ) :-
   (  fwrite( f, 0, Places, Number ),
      write( ` . ` )
   ) ~> String,
   read( Rounded ) <~ String.

%formata casos recuperados com a similaridade
formata_casos_sim([], [], [], []).
formata_casos_sim([C|Cs], [S|Ss], [D|Ds], Trios)  :- S < 0.7,
formata_casos_sim(Cs, Ss, Ds, Trios), !.
formata_casos_sim([C|Cs], [S|Ss], [D|Ds], [C-SX-D|Trios]) :- S >= 0.7,
round(S,2,SX),
formata_casos_sim(Cs, Ss, Ds, Trios), !.
formata_casos_sim([C|Cs], [S|Ss], [D|Ds], [C-S|Trios]) :-
formata_casos_sim(Cs, Ss, Ds, Trios).

%similaridade entre os casos
sim_case2(NCP,Type,SimList, WeightList) :- findall(X, ontology(X, isA,
Type),CaseList), rec_findall(CaseList,hasProblem, ProblemList),
head(ProblemList,FirstProblem),
ontology(FirstProblem, isA, T2),
findall(Y, ontology(T2, hasRelation, Y), Relations),
member(ZZ,ProblemList),rec_findall_2(ZZ,Relations,SL),
sim_attribute2(SL,NCP,WeightList,SimList2), soma_converte(WeightList,RWT),
SimList is SimList2/RWT.

sim_case(NCP,Type,FL, WeightList) :- findall(X, ontology(X, isA,
Type),CaseList), rec_findall(CaseList,hasSolution, SolutionList),
rec_findall(SolutionList,X, SolutionValuesList),
findall(SL,sim_case2(NCP,Type,SL, WeightList),SimList),
formata_casos_sim(CaseList,SimList,SolutionValuesList, FL2),
member(FL,FL2).
```

b) Agente CBRAA

```
:- ensure_loaded( system(chimera) ).
:- ensure_loaded(cbraa).

cbraa_agent(Port) :-
   agent_create( cbraa_agent, cbraa_retrieval, Port ),
   write( `Agente CBRAA!~M~J` ),
   write( `==================~M~J` ),
   write( `Universidade Federal do Maranhão ~M~J` ),
   write( `Grupo de pesquisa em Engenharia de Software e Engenharia de
Conhecimento~M~J` ),
   write( `Autor: William Mendes~M~J` ),
   write( `Orientadora: Profª Dr. Maria del Rosario Girardi~M~J` ),
   write( `Rodando na porta: ` ),
   write( Port ),
   write( `~M~J~M~J` ).

cbraa_retrieval( Name, Link, sim_cases(NCP,Problem, WeightList) ) :-
   write( `~M~JComputing Similarity and sending cases to Interface Agent`
),
   write( `~M~J` ),
   caseproblem(Case,Problem),
```

```
   findall(X,sim_case(NCP,Case,  X,  WeightList),Cases),  agent_post(  Name,
Link, interface_list(Cases) ).
```

c) Agente de interface e Formulário

```
:- ensure_loaded( system(chimera) ).
:- ensure_loaded(cbraa).

% agente de interface para comunicacao com o agente CBRAA
cbraa_agent(Port1) :-
   agent_create( cbraa_agent, retrieved, Port1 ),
   write( `Agente Interface CBRAA!~M~J` ),
   write( `==================~M~J` ),
   write( `Universidade Federal do Maranhão ~M~J` ),
   write( `Grupo de pesquisa em Engenharia de Software e Engenharia de
Conhecimento~M~J` ),
   write( `Autor: William Mendes~M~J` ),
   write( `Orientadora: Profª Dr. Maria del Rosario Girardi~M~J` ),
   write( `Rodando na porta: ` ),
   write( Port1 ),
   write( `~M~J~M~J` ).

retrieved( Name, Link, interface_list(Cases) ) :- retrieved_cases(Cases).

retrieved_cases(Text) :-
   _S1 = [ws_sysmenu,ws_popup,ws_caption,dlg_ownedbyprolog],
   _S2 = [ws_child,ws_tabstop,ws_visible,bs_pushbutton,
          bs_text,bs_center,bs_vcenter],
   _S3 = [ws_child,ws_visible,ss_left],
   _S4 = [ws_child,ws_disabled,ws_disabled,ws_visible,
          lbs_nosel,lbs_sort,lbs_hasstrings],
   wdcreate(  retcases, `Retrieved Cases`, 187,  60, 366, 259, _S1 ),
   wccreate( (retcases,1),  button, `Ok`, 140, 180,  90,  40, _S2 ),
   wfcreate( arialbold, arial, 16, 2),
   wccreate( (retcases,2), static, `CBRAA - Retrieved Cases List`,
20,  20, 320, 140, _S3 ),
   wfont((retcases,2),arialbold),
   wccreate( (retcases,3), listbox, `List1`, 20,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (arg(1,Item,ItemAtomX),
arg(1,ItemAtomX,ItemAtom),
   atom_string(ItemAtom,ItemString),wlbxadd((retcases, 3), -1, ItemString
))),
   wccreate( (retcases,4), listbox, `List2`, 60,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (wlbxadd((retcases, 4), -1, `- sim:` ))),
   wccreate( (retcases,5), listbox, `List3`, 90,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (arg(1,Item,ItemAtomX),
arg(2,ItemAtomX,ItemAtom),
   number_string(ItemAtom,ItemString),wlbxadd((retcases, 5), -1, ItemString
))),
   wccreate( (retcases,6), listbox, `List4`, 120,  50, 65, 130, _S4 ),
   forall(member(Item,Text), (wlbxadd((retcases, 6), -1, `- solution:` ))),
   wccreate( (retcases,7), listbox, `List5`, 175,  50, 165, 130, _S4 ),
   forall(member(Item,Text), (arg(2,Item,ItemAtom),
   atom_string(ItemAtom,ItemString),wlbxadd((retcases, 7), -1, ItemString
))), window_handler(retcases, retrieved_cases_handler),
   call_dialog( retcases, X ).
```

```
% formulario principal

cbr :-
   _S1 = [ws_sysmenu,ws_popup,ws_caption,dlg_ownedbyprolog],
   _S2 = [ws_child,ws_visible,ss_center,ws_ex_transparent],
   _S3 = [ws_child,ws_visible,ss_right,ws_ex_transparent],
   _S4 = [ws_child,ws_visible,ws_border,ws_tabstop,ws_vscroll],
   _S5 = [ws_child,ws_tabstop,ws_visible,bs_pushbutton,
          bs_text,bs_center,bs_vcenter],
   _S6 = [ws_child,ws_visible],
   wfcreate( arialbold, arial, 16, 2),
   wfcreate( arialitalic, arial, 17, 1),
   wfcreate( arialregular, arial, 15, 0),
   wdcreate(  cbr, `Interface Agent for CBRAA`, 188,  60, 626, 379, _S1 ),
   wccreate( (cbr,21000), static,  `Interface Agent for submission of new
case-problem to the CBRAA Agent`, 0, 0, 620,  20, _S2 ),
   wfont((cbr,21000),arialbold),
   wccreate( (cbr,11001), static,  `Author: William Mendes / Orientadora:
Profa. Dra. Rosario Girardi`, 0, 330, 630,  20, _S2 ),
   wfont((cbr,11001),arialbold),
   wccreate( (cbr,11002), static, `Grupo de Pesquisa em Engenharia de
Software e Engenharia de Conhecimento - GESEC/UFMA`, 0, 310, 620,  20, _S2
),
   wfont((cbr,11002),arialbold),
   wccreate( (cbr,11003), static,  `Which kind of problem do you need to
solve?`, 155,  140, 300,  20, _S3 ),
   wfont((cbr,11003),arialitalic),
   wccreate( (cbr,1),  listbox, `List1`, 245,  180, 140,  70, _S4 ),
   list_hierarchy_down(problem,Z),
   forall(member(N,Z), (atom_string(N,N1),wlbxadd((cbr,1), -1, N1))),
   wlbxsel( (cbr,1), 0, 1 ),
   wccreate( (cbr,1000),  button,  `&OK`, 140, 260, 160,  30, _S5 ),
   wccreate( (cbr,1100),  button,  `&Close`, 340, 260, 160,  30, _S5 ),
   wfont((cbr,1000),arialregular),
   wfont((cbr,1100),arialregular),
   wccreate( (cbr,9), grafix, ``,  0, 20, 626, 90, _S6 ),
   window_handler(cbr,my_handler_main), call_dialog( cbr, X ).

my_handler_main( _, msg_close, _, cancel ).

my_handler_main((cbr,1100),msg_button,_,cancel) :- write( `~M~JCancel Case
Retrieval~M~J` ).

my_handler_main((cbr,1000),msg_button,_,ok):- lb_getselitem((cbr,1), X),
cbraa(X).

my_handler_main( (GrafixControl,9), msg_paint, _, _ ):-
   gfx_paint( GrafixControl ),
   draw_bitmap,
   gfx_end(   GrafixControl ) .

% interface
cbraa(Problem) :-
   _S1 = [ws_sysmenu,ws_popup,ws_caption,dlg_ownedbyprolog],
   _S2 = [ws_child,ws_visible,ss_center,ws_ex_transparent],
   _S3 = [ws_child,ws_visible,ss_right,ws_ex_transparent],
   _S4 = [ws_child,ws_visible,ws_border,ws_tabstop,ws_vscroll],
   _S4x = [ws_child,ws_border,ws_tabstop,ws_vscroll],
   _S5 = [ws_child,ws_tabstop,ws_visible,bs_pushbutton,
          bs_text,bs_center,bs_vcenter],
   _S6 = [ws_child,ws_visible],
```

```
   _S7 = [ws_child,ws_visible,ws_tabstop,ws_border,
          es_center,es_multiline,es_autovscroll,es_autohscroll],
   wfcreate( arialbold, arial, 16, 2),
   wfcreate( arialitalic, arial, 14, 1),
   wfcreate( arialregular, arial, 15, 0),
   wdcreate( cbraa_form, `Interface Agent for CBRAA`, 188,  60, 626, 379,
_S1 ),
   wccreate( (cbraa_form,21000), static,  `Interface Agent for submission
of new case-problem to the CBRAA Agent`,  0,   0, 620,  20, _S2 ),
   wfont((cbraa_form,21000),arialbold),
   wccreate( (cbraa_form,11001), static,  `Autor: William Mendes /
Orientadora: Profa. Dra. Rosario Girardi`, 0, 330, 630,  20, _S2 ),
   wfont((cbraa_form,11001),arialbold),
   wccreate( (cbraa_form,11002), static,  `Grupo de Pesquisa em Engenharia
de Software e Engenharia de Conhecimento - GESEC/UFMA`, 0, 310, 620,  20,
_S2 ),
   wfont((cbraa_form,11002),arialbold),
   atom_string(ProblemS,Problem),
   wccreate( (cbraa_form,333),  listbox, `Hidden`,
130, 220, 140,  50, _S4x ),
   wlbxadd((cbraa_form,333), -1, Problem),
   wlbxsel( (cbraa_form,333), 0, 1 ),
   findall(X,ontology(ProblemS,hasRelation,X),ProblemAttributesList),
   qtde(ProblemAttributesList,E1), (E1>0->WeightField is 1/E1;WeightField
is 0),
   number_string(WeightField,WeightField2),(E1>0->
   head(ProblemAttributesList,Att1),
   atom_string(Att1,A1),
   wccreate( (cbraa_form,11003), static,  A1,  20,  130, 100,  20, _S3 ),
   wfont((cbraa_form,11003),arialitalic),
   wccreate( (cbraa_form,9000), static,  `w = `, 50, 150, 20, 20, _S2 ),
   wccreate( (cbraa_form,8000), edit, WeightField2,  70, 150,  50,  20, _S7
),
   list_hierarchy_down(A1,A1List),
   wccreate( (cbraa_form,1), listbox, `AttList1`, 130, 130, 140, 50, _S4 ),
   forall(member(V1,A1List), (wlbxadd((cbraa_form,1), -1, V1))),
   wlbxsel( (cbraa_form,1), 0, 1 )
   ;true),
   qtde(ProblemAttributesList,E2), (E2>1->
   tail(ProblemAttributesList,T2),
   head(T2,Att2),
   atom_string(Att2,A2),
   wccreate( (cbraa_form,11004), static,  A2, 20, 175, 100,  20, _S3 ),
   wfont((cbraa_form,11004),arialitalic),
   wccreate( (cbraa_form,9001), static,  `w = `, 50, 195, 20, 20, _S2 ),
   wccreate( (cbraa_form,8001), edit, WeightField2, 70, 195, 50, 20, _S7 ),
   list_hierarchy_down(A2,A2List),
   wccreate( (cbraa_form,2), listbox,`AttList2`, 130, 175, 140,  50, _S4 ),
   forall(member(V2,A2List), (wlbxadd((cbraa_form,2), -1, V2))),
   wlbxsel( (cbraa_form,2), 0, 1 )
   ;true),
   qtde(T2,E3), (E3>1->
   tail(T2,T3),
   head(T3,Att3),
   atom_string(Att3,A3),
   wccreate( (cbraa_form,11007), static,  A3, 20, 220, 100,  20, _S3 ),
   wfont((cbraa_form,11007),arialitalic),
   wccreate( (cbraa_form,9002), static,  `w = `, 50, 240, 20, 20, _S2 ),
   wccreate( (cbraa_form,8002), edit, WeightField2,  70, 240,  50,  20, _S7
),
   list_hierarchy_down(A3,A3List),
```

```
   wccreate( (cbraa_form,3),  listbox,`AttList3`, 130, 220, 140, 50, _S4 ),
   forall(member(V3,A3List), (wlbxadd((cbraa_form,3), -1, V3))),
   wlbxsel( (cbraa_form,3), 0, 1 )
   ;true),
   qtde(T3,E4), (E4>1->
   tail(T3,T4),
   head(T4,Att4),
   atom_string(Att4,A4),
   wccreate( (cbraa_form,11005), static,  A4, 290,  130, 180,  20, _S3 ),
   wfont((cbraa_form,11005),arialitalic),
   wccreate( (cbraa_form,9003), static,  `w = `, 400, 150, 20, 20, _S2 ),
   wccreate( (cbraa_form,8003), edit, WeightField2,  420, 150,  50,  20,
_S7 ),
   list_hierarchy_down(A4,A4List),
   wccreate( (cbraa_form,4), listbox, `AttList4`, 480, 130, 130, 50, _S4 ),
   forall(member(V4,A4List), (wlbxadd((cbraa_form,4), -1, V4))),
   wlbxsel( (cbraa_form,4), 0, 1 )
   ;true),
   qtde(T4,E5), (E5>1->
   tail(T4,T5),
   head(T5,Att5),
   atom_string(Att5,A5),
   wccreate( (cbraa_form,11006), static,  A5, 290,  175, 180,  20, _S3 ),
   wfont((cbraa_form,11006),arialitalic),
   wccreate( (cbraa_form,9004), static,  `w = `, 400, 195, 20, 20, _S2 ),
   wccreate( (cbraa_form,8004), edit, WeightField2,  420, 195,  50,  20,
_S7 ),
   list_hierarchy_down(A5,A5List),
   wccreate( (cbraa_form,5), listbox,`AttList5`, 480, 175, 130,  50, _S4 ),
   forall(member(V5,A5List), (wlbxadd((cbraa_form,5), -1, V5))),
   wlbxsel( (cbraa_form,5), 0, 1 )
   ;true),
   qtde(T5,E6), (E6>1->
   tail(T5,T6),
   head(T6,Att6),
   atom_string(Att6,A6),
   wccreate( (cbraa_form,11007), static,  A5, 290,  175, 180,  20, _S3 ),
   wfont((cbraa_form,11007),arialitalic),
   wccreate( (cbraa_form,9005), static,  `w = `, 400, 240z, 20, 20, _S2 ),
   wccreate( (cbraa_form,8005), edit, WeightField2,  420, 240,  50,  20,
_S7 ),
   list_hierarchy_down(A6,A6List),
   wccreate( (cbraa_form,6), listbox, `AttList6`, 480, 220, 130, 50, _S4 ),
   forall(member(V6,A6List), (wlbxadd((cbraa_form,6), -1, V6))),
   wlbxsel( (cbraa_form,6), 0, 1 )
   ;true),
   wccreate( (cbraa_form,1000),  button,  `&Submit New Case-Problem`,
140, 270, 160,  30, _S5 ),
   wccreate( (cbraa_form,1100),  button,  `&Close`,
340, 270, 160,  30, _S5 ),
   wfont((cbraa_form,1000),arialregular),
   wfont((cbraa_form,1100),arialregular),
   wccreate( (cbraa_form,9), grafix, ``,  0, 20, 626, 90, _S6 ),
   window_handler(cbraa_form,my_handler), cbraa_agent(9999),
   agent_create( cbraa_agent, 1, `localhost`, 8888),
   call_dialog( cbraa_form, X ).

retrieved_cases(Text) :-
   _S1 = [ws_sysmenu,ws_popup,ws_caption,dlg_ownedbyprolog],
   _S2 = [ws_child,ws_tabstop,ws_visible,bs_pushbutton,
          bs_text,bs_center,bs_vcenter],
```

```
   _S3 = [ws_child,ws_visible,ss_left],
   _S4 = [ws_child,ws_disabled,ws_disabled,ws_visible,
          lbs_nosel,lbs_sort,lbs_hasstrings],
   wdcreate(  retcases, `Retrieved Cases`, 187,  60, 366, 259, _S1 ),
   wccreate( (retcases,1),  button, `Ok`,  140, 180,  90,  40, _S2 ),
   wfcreate( arialbold, arial, 16, 2),
   wccreate( (retcases,2), static, `CBRAA - Retrieved Cases List`,
20,  20, 320, 140, _S3 ),
   wfont((retcases,2),arialbold),
   wccreate( (retcases,3), listbox, `List1`, 20,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (arg(1,Item,ItemAtomX),
arg(1,ItemAtomX,ItemAtom),
   atom_string(ItemAtom,ItemString),wlbxadd((retcases, 3), -1, ItemString
))),
   wccreate( (retcases,4), listbox, `List2`, 60,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (wlbxadd((retcases, 4), -1, `- sim:` ))),
   wccreate( (retcases,5), listbox, `List3`, 90,  50, 45, 130, _S4 ),
   forall(member(Item,Text), (arg(1,Item,ItemAtomX),
arg(2,ItemAtomX,ItemAtom),
   number_string(ItemAtom,ItemString),wlbxadd((retcases, 5), -1, ItemString
))),
   wccreate( (retcases,6), listbox, `List4`, 120,  50, 65, 130, _S4 ),
   forall(member(Item,Text), (wlbxadd((retcases, 6), -1, `- solution:` ))),
   wccreate( (retcases,7), listbox, `List5`, 175,  50, 165, 130, _S4 ),
   forall(member(Item,Text), (arg(2,Item,ItemAtom),
   atom_string(ItemAtom,ItemString),wlbxadd((retcases, 7), -1, ItemString
))),

   window_handler(retcases, retrieved_cases_handler), call_dialog(
retcases, X ).

retrieved_cases_handler( _, msg_close, _, cancel ).

retrieved_cases_handler((retcases,1),msg_button,_,cancel) :- write(
`~M~JList closed.~M~J` ),  agent_close(cbraa_agent).

retrieved_cases_handler((retcases,1),msg_button,_,ok) :- write( `~M~JList
closed.~M~J` ),  agent_close(cbraa_agent).

my_handler( _, msg_close, _, cancel ).

my_handler((cbraa_form,1100),msg_button,_,cancel) :- write( `~M~JCancel
Case Retrieval~M~J` ),  agent_close(cbraa_agent).

my_handler((cbraa_form,1000),msg_button,_,ok):-
   lb_getselitem((cbraa_form,333), ProblemType),
atom_string(PT,ProblemType),
   findall(X,ontology(PT,hasRelation,X),ProblemAttributesList),
   qtde(ProblemAttributesList,Qtd),
   (Qtd>0->
   lb_getselitem((cbraa_form,1), Val1), append([], [Val1], NCP1), wtext(
(cbraa_form, 8000), W1), append([], [W1], WT1) ;true),
   (Qtd>1->
   lb_getselitem((cbraa_form,2), Val2), append(NCP1, [Val2], NCP2), wtext(
(cbraa_form, 8001), W2), append(WT1, [W2], WT2); agent_post( cbraa_agent,
1, sim_cases(NCP1,PT, WT1) ), !),
   (Qtd>2->
   lb_getselitem((cbraa_form,3), Val3), append(NCP2, [Val3], NCP3), wtext(
(cbraa_form, 8002), W3), append(WT2, [W3], WT3);agent_post( cbraa_agent, 1,
sim_cases(NCP2,PT, WT2) ), !),
   (Qtd>3->
```

```
   lb_getselitem((cbraa_form,4), Val4), append(NCP3, [Val4], NCP4), wtext(
(cbraa_form, 8003), W4), append(WT3, [W4], WT4);agent_post( cbraa_agent, 1,
sim_cases(NCP3,PT, WT3) ), !),
   (Qtd>4->
   lb_getselitem((cbraa_form,5), Val5), append(NCP4, [Val5], NCP5), wtext(
(cbraa_form, 8004), W5), append(WT4, [W5], WT5);agent_post( cbraa_agent, 1,
sim_cases(NCP4,PT, WT4) ), !),
   (Qtd>5->
   lb_getselitem((cbraa_form,6), Val6), append(NCP5, [Val6], NCP6), wtext(
(cbraa_form, 8005), W6), append(WT5, [W6], WT6),
   agent_post( cbraa_agent, 1, sim_cases(NCP6,PT, WT6) ); agent_post(
cbraa_agent, 1, sim_cases(NCP5, PT, WT5) )).

lb_getselitem( Window, Item ) :-
   sndmsg( Window, lb_getcursel, 0, 0, R ),
   (  R = -1
   -> Item = ``
   ; wlbxget( Window, R, Item)
   ).

my_handler( (GrafixControl,9), msg_paint, _, _ ):-
   gfx_paint( GrafixControl ),
   draw_bitmap,
   gfx_end(   GrafixControl ) .

draw_bitmap:-
  bitmap( What, File ),
  gfx_bitmap_load( What, File ),
  gfx(bitmap(200,20,470,100,0,0,What)).

bitmap( test, 'gesec.bmp' ).

load_bitmap:-
      gfx_bitmap_load( X, Y ).
```