TDP MECATEAM 2010

Cleber Pinelli Teixeira*, Orivaldo Vieira Santana Júnior†, Augusto Loureiro da Costa‡

*Av. Adhemar de Barros s/n, Ondina - 40170-110
Universidade Federal da Bahia (UFBA)
Departamento de Ciência da Computação - Instituto de Matemática
Salvador, Bahia, Brasil

†Av. Professor Luís Freire s/n, Cidade Universitária - 50740-540
Universidade Federal de Pernambuco (UFPE)
Centro de Informática (CIn)
Recife, Pernambuco, Brasil

‡Rua Prof. Aristides Novis, Federação - 40210-630
Universidade Federal da Bahia (UFBA)
Departamento de Engenharia Elétrica - Politécnica
Salvador, Bahia, Brasil

Emails: pinelli@dcc.ufba.br, ovsj@cin.ufpe.br, augusto.loureiro@ufba.br

Abstract— MecaTeam has been developed under an architecture that basically keep the focus on a Knowledge Based System. In 2009, the MecaTeam was used as base team by GPR-2D, a team from Paraná, such fact shows the relevance of its structure and the maturation that the team has gotten over the couple of years. For 2010, MecaTeam is shown with a new base, the agent2d that is the base used by Helios, the actual World Champion. This migration was proposed as priority to the continuity of the team, because of the obsoleteness of the Uva base. This change was reached without changing the multi-layer structure proposed to MecaTeam.

Keywords— Intelligent Multiagent systems, Planning, reasoning and smart making decision.

Resumo— O MecaTeam tem sido desenvolvido sob uma arquitetura que basicamente mantém como foco um Sistema Baseado em Conhecimento. Em 2009 o MecaTeam foi utilizado como time base pelo GPR-2D, um time do Paraná, tal fato mostra a relevância de sua estrutura e o amadurecimento que o time obteve ao longo dos anos. Para 2010, o MecaTeam apresenta-se com uma nova base, o agent2d, que é a base utilizada pelo Helios, o atual campeão mundial. Essa migração, foi imposta como prioridade a continuidade do time, devido a obsolescência da base Uva. Essa mudança no entanto, foi realizada sem modificar a estrutura multi-camada proposta para o MecaTeam.

Keywords— Sistemas multi-agentes inteligentes, Planejamento, raciocínio e tomada de decisão inteligente.

1 Introdução

O MecaTeam apresenta uma evolução na arquitetura do agente UFSC-Team-98 (Costa and Bittencourt, 1999b) na RoboCup'98, chamado Agente Autônomo Concorrente (Costa and Bittencourt, 1999a). Este baseia-se em um modelo híbrido para agente cognitivo (Bittencourt and Costa, 2001) e possui uma arquitetura de três camadas, cada uma delas representa um nível decisório distinto que complementa os demais para a construção de uma agente cognitivo. A arquitetura atual do MecaTeam é implementada re-utilizando o código do agent2d, que por vários anos foi o uva base, a Expert-Coop++ (Costa et al., 2003) e o um framework desenvolvido para modularizar a estrutura do código MecaTeam.

A equipe de futebol de robôs HELIOS desenvolveu uma bliblioteca base para implementar programas clientes para o RCSS chamada de *librcsc*. Para facilitar a implementação de novos agentes de fotebol de robôs simulado em duas dimensões, a equipe HELIOS disponibilizou um agente simples produzido com a *librcsc* denominado de *agent2d* (Akiyama et al., 2009).

A Expert-Coop++ é uma biblioteca orientada a objetos destinada a auxiliar o desenvolvimento de sistemas multiagentes sob restrição de tempo real do tipo melhor esforço(Costa et al., 2003), implementada na linguagem de programação C++.

O Framework MecaTeam propõe uma infraestrutura orientada a objetos (OO) definindo a arquitetura intra-agente de um agente de futebol de robôs. Assim, evita a reimplementação de partes básicas do agente e promove um tipo de reuso de código mais abrangente que a simples reutilização do código de outro time (de Santana Jr et al., 2008).

O MecaTeam 2010 é apresentado neste artigo, de acordo com a seguinte disposição: a seção 3 apresenta a arquitetura do agent2d usado como base para implementar o agente do MecaTeam; A Expert-Coop++ é mostrada na seção 4, junto com seu Sistema Baseado em Conhecimento; O Framework MecaTeam é ilustrado na seção 5; A mudança de base é ilustrada na seção 6; A configuração atual do MecaTeam é mostrada na seção 7; Na seção 8 está presente a conclusão e os trabalhos futuros.

2 Arquitetura do MecaTeam

A arquitetura do MecaTeam está dividida em três níveis o reativo, o instintiva e o cognitiva que se auto-complementam.

3 Arquitetura do agent2d

Por vários anos o MecaTeam vinha utilizando como base o time Uva Trilearn que foi disponibilizado em 2001. No entanto este framework apresenta-se obsoleto diante das mudanças sofridas pelo soccer server ao longo dos anos.

4 A Biblioteca Expert-Coop++

A **Expert-Coop++** (Costa et al., 2003) é uma biblioteca orientada a objetos destinada a auxiliar o desenvolvimento de sistemas multiagentes sob restrição de tempo real (Costa et al., 2003), implementada na linguagem de programação *C++*. Esta biblioteca oferece suporte a Sistemas Baseados em Conhecimento (SBC).

A expert-coop++ inicialmente integrada ao uva trouxe um modelo diferenciado, onde as heurísticas definidas pelo especialista são analisadas pelo motor de inferência através de regras de produção.

4.1 O Sistema Baseado em Conhecimento

O Sistema Baseado em Conhecimento utilizado na Expert-Coop++ é classificado como um Sistema Baseado em Regras de Produção (SBRP). Este sistema apresenta uma arquitetura de três módulos: uma *base de regras*, uma*base de fatos ou memória de trabalho* e um *motor de inferência* (de Santana Jr et al., 2006) como ilustra a figura 1.

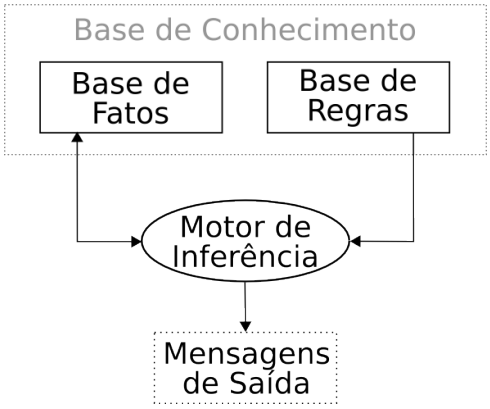

Figura 1: Arquitetura do Sistema Baseado em Conhecimento utilizado no MecaTeam

A base de regras e a base de fatos formam a base de conhecimento contendo uma representação do conhecimento do agente robótico jogador de futebol. O motor de inferência avalia as regras e aplica estas regras de acordo com as informações contidas na base de fatos.

4.2 Base de Fatos

A base de fatos contém uma descrição dos estados do ambiente. Ela é construída tomando como base árvores binárias, onde cada nó da árvore possui uma representação de conhecimento. O formalismo de representação de conhecimento utilizado é a lógica de primeira ordem (Bittencourt, 2001) usanda a forma implícita como ilustrado abaixo.

```
( logic (<objeto> <atributo> <valor>))
```

A base de fatos é atualizada de duas maneiras: uma com informações do ambiente e outra com as regras. As informações mais relevantes sobre os estados do ambiente são fornecidas por funções da *librcsc*.

4.3 Base de Regras

A base de regras é responsável pela leitura do arquivo de regras, previamente definido pela extração destas regras e pelo seu armazenamento em uma lista. Este processo de extração é realizado por um analisador de regras.

A base de regras relaciona estados às ações do robô. Logo, a inteligência do agente é formalizada em regras de produção. Elas são criadas com intuito de definir o comportamento do agente para um determinado conjuto de estados correntes.

4.4 Motor de Inferência

O motor de inferência é o principal componente do sistema. Seu funcionamento consiste em realizar inferências a partir do conhecimento contido na base de regras e da base de fatos. O modo de raciocínio usado no motor de inferência é o *forward chaining*(encadeamento progressivo) (Bittencourt, 2001), em que a parte esquerda da regra é comparada com a descrição da situação atual contida na base de fatos. Apenas as regras que satisfazem esta descrição são selecionadas. Um exemplo de regra é ilustrado na figura 2.

Ao final do processo de seleção das regras, o motor de inferência dispõe de um conjunto de regras que satisfazem a situação atual do problema (denominado *conjunto de conflito*). Se este conjunto for vazio, o processo de inferência é finalizado; caso contrário, torna-se necessária a definição das regras que serão efetivamente executadas e sua ordem de execução(Bittencourt, 2001).

```
(rule_kickToGoal
   (if   (logic ( player ball_kickable true ))
         (logic ( player localization AREA_ATTACK )))
   (then (logic ( reactive_behavior active kick_to_goal ))))
```

Figura 2: Um exemplo de regra utilizada no agente MecaTeam

As regras foram desenvolvidas de acordo com estratégias elaboradas pelos integrantes da equipe MecaTeam. As estruturas mais importantes da gramática que descreve as regras são o lado direito e lado esquerdo das regras. O lado esquerdo das regras contém a representação de um estado do ambiente e é identificado através do token if, já o lado direito contém informações para modificar o estado e o then identifica este lado. Para representar o conhecimento sobre o estado da partida, é utilizado o formalismo da lógica (Bittencourt, 2001); logo a informação é formatada em objeto, atributo e valor como na figura 2.

5 Framework MecaTeam

O framework MecaTeam (de Santana Jr et al., 2008). foi proposto em 2008 como uma forma de auxiliar a manutenção do time. Neste framework são utilizados dois padrões de projeto, onde cada um deles foca em um particular problema.

O *Strategy* fornece suporte ao ponto de extensão da estratégia de raciocínio. Isto possibilita a criação de vários agentes, cada um com sua estratégia de raciocínio, apenas estendendo e implementado a classe *Brain*.

O *Singleton* é aplicado, por exemplo, na classe encarregada de modelar o ambiente, pois é necessário que exista exatamente uma instância desta classe acessível para diversos componentes do Framework MecaTeam.

As principais motivações para criar o framework foram: a dificuldade de incorporar novos comportamentos, estratégias de raciocínio; avaliar os impactos de modificações em trechos de código; e principalmente o reuso do software.

6 Migração de Base

Obviamente, uma mudança de base traz algumas consequências à manutenção do time, pois o sistema desenvolvido para o MecaTeam não poderia ser simplesmente descartado, então o time passou por um processo de migração, onde toda a estrutura do time teve que ser adaptada à nova base.

O framework MecaTeam foi desenvolvido para dar mais modularidade ao código, devido a discrepância entre as arquiteturas, esta migração não foi uma tarefa tão simples. Os passos para alcançar tal objetivo foram primeiramente criar um agente simples utilizando-se o agent2d, ou seja a conexão entre o agent2d e o framework. Em seguida estabelecer a conexão entre o agent2d e a Expert-Coop++.

Nesta segunda etapa, a tarefa passou a ser mais complexa, foi-se necessário estudar ambas as bases, comparando-se suas arquiteturas, afim de criar uma interface com o framework, o qual foi projetado estritamente para a arquitetura do Uva.

No agent2d o raciocínio dos jogadores é divido em papéis e os comportamentos são separados por classes, a interface com o simulador é realizada pela biblioteca librcsc.

Já o uva trata o raciocínio de todos os jogadores em apenas uma classe, assim como os comportamentos. Essa diferença entre as arquiteturas é o que torna a tarefa da migração mais complexa.

As mudanças realizadas portanto foram encapsular os comportamentos, reimplementar o raciocínio do agente sob as regras de produção e conectar aos modulos cognitivo e instintivo.

Com as mesmas regras aplicadas ao time antes e após a substituição de base, foi possível notar uma melhora bastante significativa no seu desempenho.

7 MecaTeam 2010

Após alguns anos trabalhando-se na evolução do time, o MecaTeam hoje possui sua arquitetura de três camadas concluída, onde a camada reativa que era tratada pelo Uva Trilearn passou a ser tratada pelo agent2d, a camada instintiva tratada pela Expert-Coop++ que é responsável pela escolha do comportamento, e a camada cognitiva que é responsável pelo nível estratégico do time, possibilitando a cooperação entre os agentes. A complexidade de comportamento do agente é incrementada a cada camada, como pode ser visto na Figura 3.

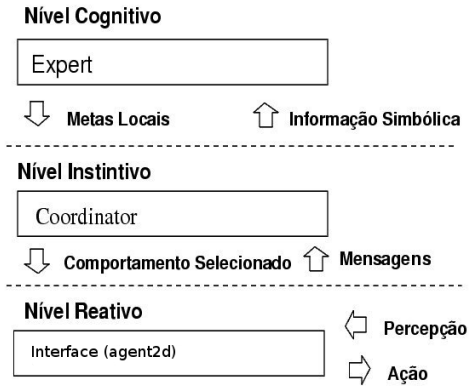

Figura 3: Arquitetura do Agente MecaTeam 2010

A proposta para o MecaTeam 2010 foi a migração de base, migração esta que tornou o desempenho do agente MecaTeam muito superior a sua versão anterior.

Com essa migração o MecaTeam mantém uma arquitetura robusta e com um time base de alto desempenho, ou seja, não apenas utilizando-se o time base, mas reaproveitando a arquitetura feita estritamente para o MecaTeam.

8 Conclusões e Trabalhos Futuros

A comunicação entre os agentes que deveria ser realizada foi adiada, pois por questões de desempenho a mudança de base foi priorizada por mostra-se muito mais eficaz. Através de um protocolo de comunicação, a camada cognitiva do agente será a próxima tarefa a ser realizada para o MecaTeam.

Um protocolo de comunicação é uma solução adotada para que haja cooperação entre os agentes, os quais devem atingir o objetivo através da distribuição de metas, planos e tarefas.

Esta etapa, apresenta-se como extremamente importante para o MecaTeam, pois é que deverá finalizar o modelo proposto para o MecaTeam, um agente cognitivo.

Após esta etapa, como trabalho futuro a um prazo mais longo, trabalhar-se na camada reativa ou até mesmo na camada cognitiva utilizando técnicas como redes neurais por exemplo é uma possível idéia.

Referências

Akiyama, H., Shimora, H., and Noda, I. (2009). Helios2009 team description, *RoboCup 2009*.

Bittencourt, G. (2001). *Inteligência Artificial Ferramentas e Teorias*, Editora da UFSC, ISBN 85-328-0138-2, 362 p., Florianópolis, SC, 2. edição.

Bittencourt, G. and Costa, A. L. d. (2001). Hybrid cognitive model, *The Third International Conference on Cognitive Science ICCS'2001 : Workshop on Cognitive Angents and Agent Interaction*. Pequim, China.

Costa, A. L. d. and Bittencourt, G. (1999a). From a concurrent architecture to a concurrent autonomous agents architecture., *IJCAI'99, Third International Workshop in RoboCup* pp. 85–90. Springer, Lecture Notes in Artificial Inteligence.

Costa, A. L. d. and Bittencourt, G. (1999b). Ufsc-team: A cognitive multi-agent approach to the robocup'98 simulator league., *RoboCup98 Workshop - Team description* pp. 371, 377. Springer, Lecture Notes in Artificial Inteligence, vol.1694.

Costa, A. L. d., Bittencourt, G., Gonçalves, E. M. N. and Silva, L. R. (2003). Expert-coop++: Ambiente para desenvolvimento de sistemas multiagente., *IV ENIA Encontro Nacional de Inteligência Artificial* pp. 597–606. XXIII Congresso da Sociedade Brasileira de Computação.

de Santana Jr, O. V., Sousa, J. P. R. P., Linder, M. S. and da Costa, A. L. (2006). Mecateam: Um sistema multiagente para o futebol de robôs simulado baseado no agente autônomo concorrente, *Encontro de Robótica Inteligente / XXVI Congresso da Sociedade Brasileira de Computação* pp. 146–152.

de Santana Jr, O. V., von Flach Garcia Chavez, C. and da Costa, A. L. (2008). Mecateam framework: An infrastructure for soccer agents development of simulated robots, *5th IEEE Latin American Robotics Symposium*, pp. 146–152.