Seminário de Integração Técnico-Científica
Julho, 2012

O uso de uma plataforma multiagentes baseada em Common Lisp para auxilio a gestão de projetos

Allan Caminha Trevisan e Milton Pires Ramos

Centro de Engenharia de Sistemas Inteligentes – Instituto de Tecnologia do Paraná

(TECPAR) – Curitiba, PR - Brasil

allan.trvsn@gmail.com, mpramos@tecpar.br

Resumo

Durante os últimos anos o aumento da utilização da tecnologia de Agentes de Software vem se tornando notável. Esse trabalho irá apresentar a plataforma multiagente OMAS e o seu uso no contexto de uma aplicação de auxilio à gestão de projetos dando destaque para a integração entre o sistema de diálogos da plataforma e da ontologia dos agentes.

Palavras-chave: Common Lisp, Sistema Multiagente, OMAS, Inteligencia Artificial

1. Introdução

1.1 Sistemas Multiagentes

O termo **Sistemas Multiagentes (SMA)** é um termo que vem ganhando espaço e sendo usado em uma diversidade de contextos dentro da Computação. Os SMA constituem um campo de estudo interdisciplinar em que dependendo do problema em questão podem cobrir disciplinas como: Engenharia de Software, Inteligencia Artificial, Sistemas Distribuídos, Filosofia, Sociologia, Biologia e muitas outras. A área teve inicio com a primeira conferencia internacional de inteligencia artificial distribuída, realizada nos EUA em 1980 (COELHO, 2008).

O conceito de **Agente** refere-se a uma entidade que opera sobre um ambiente, recebendo estímulos através de seus sensores e executando ações através de seus atuadores. Por muito tempo pensou-se que as arquiteturas de agentes se restringiam ao modelo percepção ação, em que o agente não possui representação explicita de seu ambiente e ao modelo cognitivo, onde há representação explicita do ambiente, além de capacidades de decisão, planejamento e raciocínio, porém outros modelos foram propostos como por exemplo os modelos BDI (Beliefs Desires Intentions), SOAR e ACT-R (Adaptive Control of Thought—Rational) (COELHO, 2008).

Segundo (RUSSELL & NORVIG, 2003) o trabalho da **IA (Inteligencia Artificial)** é construir agentes inteligentes constituídos de **arquitetura + programa**. A arquitetura é o dispositivo físico (sensores e atuadores). O programa implementa a função do agente, mapeando percepções em ações. Com isso em mente ele define quatro tipos de agentes, de acordo com seu comportamento:

1) Regras de produção sem estado interno (sem memoria): Agentes Reativos.
2) Regras de produção com estado interno (com memoria): Agentes Reativos Baseados em Modelos.
3) Mecanismos de decisão com seleção baseada em objetivos (técnicas de planejamento): Agentes Baseados em Objetivos.
4) Mecanismos de decisão racional com função de utilidade (capacidade de enfrentar incerteza): Agentes Baseados em Utilidades.

Os termos Sistema Multiagente e Agente são usados com frequência em: Processamento Inteligente de Informação, Descoberta do Conhecimento, Programação Orientada a Agentes e Simulação e Modelagem Baseada em Agentes. Esse trabalho se concentra na área da Programação Orientada a Agentes, esse estilo de programação se distingue dos demais paradigmas como por exemplo o Orientado a Objetos da seguinte maneira:

Objetos

- Encapsulam dados e comportamentos
- São definidos por serviços (métodos)
- São ativados por mensagens
- Partilham a mesma “thread” de controle

Agentes

- Possuem autonomia e racionalidade
- São guiados por objetivos
- Possuem iniciativa própria
- Executam ações
- Mantêm seu próprio “thread” de execução

O objetivo deste trabalho é utilizar uma plataforma multiagentes chamada **OMAS** (Open Multi Agent Systems) para construir uma aplicação que auxilie na gestão e na colaboração entre membros de projeto, implementando em partes a arquitetura proposta em (BARTHÈS et al, 2010). Trata-se de uma plataforma baseada em Common Lisp desenvolvida pelo professor Jean Paul Barthès e colaboradores na Universidade de Tecnologia de Compiègne. Há uma variedade de plataformas multiagentes disponíveis que possuem tecnologias e propósitos distintos como por exemplo a plataforma **JADE** baseada em Java cujo proposito principal é o de desenvolver aplicações orientadas a web, enquanto a plataforma **OMAS** tem como objetivo a implementação de agentes inteligentes complexos sobretudo para a área de pesquisa (BARTHÈS, 2012). Outro ambiente que merece destaque é o **Jason** que também é baseado em Java, ele se constitui em um interpretador para uma extensão da linguagem abstrata de programação de agentes **AgentSpeak(L)** que se baseia na arquitetura BDI, uma das arquiteturas mais conhecidas no desenvolvimento de agentes cognitivos usada principalmente em sistemas reativos de planejamento (BORDINI & HUBNER, 2007).

1.2 Plataforma OMAS

OMAS é uma plataforma multiagente para construção de aplicações que necessitam de agentes cognitivos complexos e dotados de uma interface humana flexível (BARTHÈS & RAMOS, 2010). Por ter sido desenvolvida em **Common Lisp**, uma linguagem de programação com alto poder de expressividade e extensibilidade (uma linguagem de programação programável), a tarefa de criar nossas próprias linguagens de domínio especifico para programar agentes inteligentes é em muito facilitada, um exemplo é o sistema de diálogos da plataforma que se constitui em uma **DSL (Domain Specific Language)**.

Aplicações são construídas com três tipos de agentes: Agentes de Serviço, Assistentes Pessoais e Agentes de Transferência (postmen).

Por se tratar de uma plataforma multiagente, os agentes devem poder ser distribuídos de forma que uma aplicação funcione de maneira não centralizada. A plataforma conta com o conceito de **Coterie** (uma rede de interesses comuns) para designar os agentes pertencentes a uma aplicação, que funcionam no mesmo loop de rede. Coteries distribuídas podem ser conectadas com o uso de Agentes de Transferência. Na plataforma OMAS os agentes não se distribuem automaticamente entre os hosts como em outras plataformas, eles precisam ser previamente instalados nos hosts que irão compor a aplicação. Os Agentes de Serviço são o tipo básico de agente sobre o qual os outros são construídos, eles são responsáveis por realizar serviços de maneira pró ativa ou reativa. Os Assistentes Pessoais são agentes a cargo de interfacear um humano com a plataforma, seu papel é facilitar as interações, de forma a oferecer uma interface padrão com seu mestre, dedicando-se a servi-lo e ajudá-lo a obter serviços da plataforma (implementados pelos agentes de serviço), eles agem como mordomos digitais (Negroponte, 1995). Esse modo de interação é feito com o uso de diálogos em linguagem natural, usando como entrada o teclado ou uma interface vocal.

2. A plataforma OMAS em uma aplicação de gestão de projetos

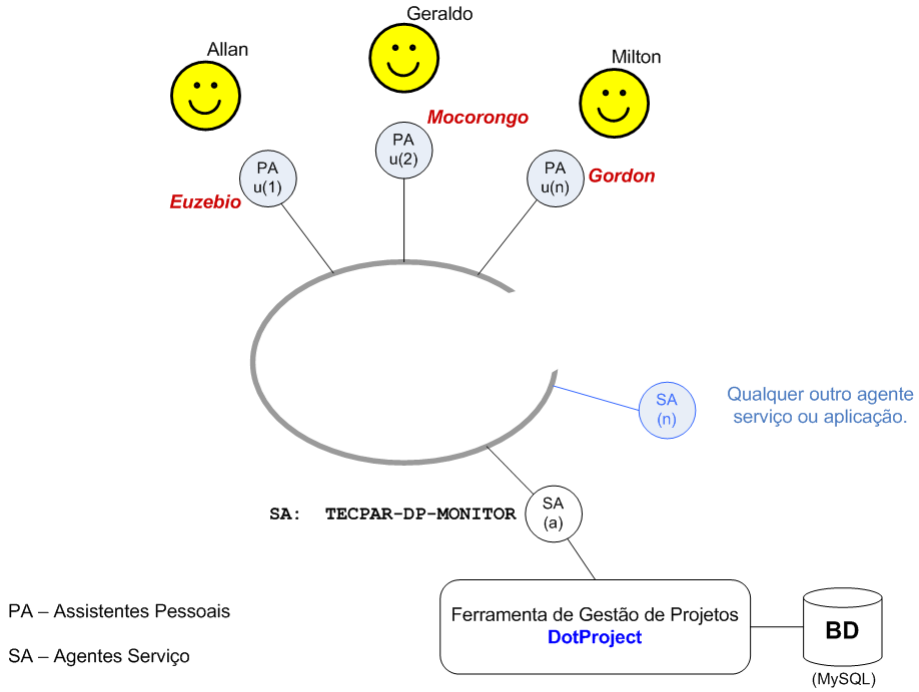
Quando usamos os computadores para trabalhar e nos comunicar se observa um crescimento da complexidade de nossos ambientes de trabalho, tal complexidade leva os usuários a recorrerem a diversas ferramentas diferentes (gerenciadores de e-mail, browsers, processadores de texto, sistemas GTD (Getting Things Done) para gerenciamento de atividades, etc). A sobrecarga cognitiva ao operar tantas ferramentas diferentes acaba levando os usuários a desorganização.

Com esse problema em mente podemos notar a necessidade da construção de interfaces melhores e mais simples para o usuário. Os Agentes Assistentes fazendo uso de diálogos em linguagem natural visam oferecer uma interface que seja ajustável as necessidades de cada usuário da plataforma.

Esse trabalho visa então implementar a arquitetura **CSCW-SD** (BARTHÈS et al, 2010). Aqui pretende-se usar o termo **CSCW** (computer suported colaborative working) em um contexto mais geral e não apenas no desenvolvimento de software em pequenas equipes. A arquitetura da aplicação desenvolvida é representada na figura 1. A aplicação constitui-se em um agente de serviço **TECPAR-DP-MONITOR** que esta a cargo de monitorar o banco de dados da ferramenta de gestão de projetos **dotproject**. O agente de serviço possui comportamento tanto reativo como pró ativo. O comportamento reativo é implementado pelas **skills** do agente (conjunto de capacidades e habilidades) que podem ser estáticas (o agente realiza sozinho), ou dinâmicas (o agente precisa da ajuda de

outros agentes para realizar a tarefa). O comportamento pró ativo é implementado com o uso de **objetivos**, que na plataforma OMAS nada mais são do que mensagens que o agente envia para si próprio dentro de um intervalo regular de tempo afim de disparar a execução de alguma skill.

Figura 1: Arquitetura da aplicação CSCW



O acesso aos serviços da aplicação é feito pelos **PA's** (Agentes Assistentes). Cada um deles foi implementado para interagir com um determinado usuário, oferecendo serviços que vão de encontro aos interesses do usuário. Um uso interessante para múltiplos PA's, cada um interagindo com o usuário em uma língua diferente, seria em uma aplicação de CSCW multicultural, com membros de projeto espalhados por diversos países.

Atualmente os serviços do **SA** (Agente de Serviço) se concentram em buscar informações relativas às atividades de um projeto. O usuário pode interagir com seu PA fazendo perguntas como por exemplo: “Quais as atividades de Allan Trevisan no projeto CSCW-TECPAR?”. Uma vez identificada a intenção da frase, uma requisição do PA para o SA é gerada, fazendo com que o SA execute uma skill que busca por atividades do membro de projeto no projeto especificado. É bom salientar que a frase introduzida pelo usuário não precisa ser exatamente como a anterior, nem gramaticalmente correta, o usuário poderia entrar com a frase: “Poderia me informar quais são as atividades de Allan

Trevisan no projeto CSCW-TECPAR?" e o PA deve ser capaz de entender da mesma maneira. Esse mecanismo de diálogos é implementado pelo sistema **MOSS** que vem integrado ao OMAS. O MOSS trata também dos aspectos referentes às **ontologias** dos agentes.

Na plataforma OMAS tanto Agentes Assistentes como Agentes de serviço podem possuir uma ontologia para representar "seu mundo". Na nossa aplicação as ontologias integram-se aos diálogos da seguinte maneira:

- Para se identificar a intenção do usuário na frase, usa-se uma abordagem baseada em padrões onde extrai-se apenas o que se interessa da frase. Essa abordagem é detalhada em (NORVIG, 1991). Se a frase corresponder ao padrão então o dialogo passa para seu próximo estado. Ela se constitui na seguinte construção MOSS:

```
(:patterns (((?* ?x) "atividades" "de" (?* ?y))
            ((?* ?y) "de" "atividades" "?" (?* ?x))
            ((?* ?x) "de" "atividades" (?* ?y) "?")
            )
           :keep ?y
           :target _gpma-ask-monitor-agent)
```

- No próximo estado do dialogo o PA envia uma mensagem para o SA, onde os argumentos são a frase entrada pelo usuário, e a informação de que se trata de um dialogo em português do Brasil, afim de que a resposta seja enviada no formato correto:

```
(:send-message :to :TECPAR-DP-MONITOR :self PA_EUZEBIO
               :action :search-what-activities-should-a-project-member-to-do-now
               :args `(((:data . ,(read-fact moss::conversation :input))
                    (:language . :br))))
```

- Ao receber a mensagem o SA deverá extrair as informações relevantes para a consecução da tarefa. Vamos supor que o usuário entrou com a frase: "Quais as atividades de **Allan Trevisan**?", nesse caso deve-se extrair o nome do membro de projeto, pois o agente precisa dessa informação para fazer a busca na base de dados. Isso é feito com a função MOSS l**ocate-objects**, onde dado um conjunto de

informações (a frase) e o nome de um conceito da ontologia (“pessoa”) é retornado um indivíduo da ontologia que contem as informações requeridas. O conceito de pessoa e o indivíduo “Allan Trevisan” são definidos da seguinte maneira:

```
(defconcept "pessoa"
  (:att "sobrenome" (:entry))
  (:att "nome"(:entry))
  (:rel "endereço" (:to "endereço"))
  (:rel "email" (:to "endereço de email"))
  (:rel "página web" (:to "endereço web"))
  (:rel "marido" (:to "pessoa")(:unique))
  (:rel "esposa" (:to "pessoa")(:unique))
  (:rel "mãe" (:to "pessoa"))
  (:rel "assistente pessoal" (:to "omas agent")))
```

```
(defindividual "pessoa"
  ("sobrenome" "Trevisan")
  ("nome" "Allan")
  ("assistente pessoal"
        ("omas agent"
              ("ag name" :is "EUZEBIO"))))
```

- Uma vez que o indivíduo “Allan Trevisan” é identificado o SA irá extrair as informações necessárias (nome e sobrenome) para fazer a busca na base de dados. O resultado da skill é então retornado para quem enviou a mensagem, nesse caso o PA, isso é feito automaticamente pelo uso da função **static-exit**.

Isso ilustra o processo de interação com o Agente Assistente o seu uso para a obtenção de serviços da plataforma, e como o mecanismo de diálogos se integra com a ontologia do agente.

3. Conclusão

Foi apresentada a construção de um protótipo de aplicação para auxilio a gestão de projetos fazendo-se uso da plataforma OMAS. Buscou-se explicar os componentes básicos que constituem o OMAS e como eles foram usados para a construção da aplicação, dando ênfase em como os Assistentes Pessoais são usados para se obter

serviços da plataforma e a integração entre a ontologia do Agente de Serviço e o mecanismo de diálogos definidos pelo MOSS.

Apesar do tempo limitado foi possível implementar o modelo para casos simples, servindo principalmente para recuperação da infraestrutura, validação do modelo e demonstração da viabilidade. As próximas etapas se concentrarão no desenvolvimento de tarefas mais complexas, por exemplo que envolvam mais de um skill do Agente de Serviço, ou a interação com outros agentes de serviço, bem como incentivar a colaboração entre os membros de uma equipe de projeto e a gestão de conhecimento tanto da equipe de projeto como corporativo.

Agradecimentos

Agradecimentos ao Instituto de Tecnologia do Paraná pela oportunidade e à Fundação Araucária pelo apoio financeiro (Programa PIBIC/2011).

Referências bibliográficas

COELHO, E. Teoria da Agencia: Arquitetura e Cenografia, LabMag e ICC, FCUL, 2008.

BARTHÈS, J. P. OMAS V9 - User Manual Issue 2, Compiègne, France, 2012.

BARTHÈS, J. P., RAMOS, M. P. OMAS, A Lisp platform for Prototyping Systems of Cognitive Agents. 2010

TACLA, C. A., PARAISO, E. C., SATO, G. Y., BARTHÈS, J. P., RAMOS, M. P. Dialog Construction in a Collaborative Project Management Enviroment. Proceding of the 2010 14th Conference on Computer Supported Cooperative Work in Design.

BORDINI, R. H., HUBNER, J. F. Jason – A Java based interpreter for an extended version of AgentSpeak. 2007

NORVIG, P., RUSSELL, S. J. Artificial Inteligence A Modern Approach. Second Edition. 2003.

NEGROPONTE, M. Being Digital. Westminster, Maryland, U.S.A. Alfred a Knopf Inc 1995.

NORVIG, P. Paradigms of AI Programming: Case Studies in Common Lisp 1991.