4
A Solução

Com o objetivo de desenvolver um estudo para Telecomunicações utilizando a abordagem do DynaCIP, na disciplina de Sistemas Multi-Agentes, surgiu a idéia de se criar uma ferramenta para a analisar o processo de compra e venda de serviços em Redes Sem Fio de Nova Geração. O aprimoramento da ferramenta para o Projeto Final de Programação permitiu analisar diferentes cenários de uso. O ContextualFramework é um *framework* baseado em sistema multi-agentes que possibilita a análise do processo de negociação de serviços em redes sem fio de nova geração, observando as informações de contexto que influenciam neste processo de negociação.

A idéia de *framework* traz uma grande redução nos esforços por parte do desenvolvedor, além de embutir um alto grau de qualidade ao software (Fayad, 2000). A solução utiliza as principais vantagens do uso de *framework*, que incluem aumento de reuso e diminuição do tempo para produção de uma família de aplicações (Mattsson, 1996). O *framework* captura os conceitos mais gerais do domínio. Uma aplicação gerada a partir de *framework* especializa ou estende os conceitos capturados pelo *framework* podendo ainda adicionar novos conceitos.

O ContextualFramework pode ser classificado quanto a sua técnica de extensão como um framework *Graybox*, pois tem flexibilidade e extensibilidade, principalmente no que diz respeito aos comportamentos dos agentes. Aplicações podem definir Comportamentos Contextuais dos agentes, que estendam a classe Behaviour do JADE e implementem a interface ContextualBehaviour do ContextualFramework. Estes determinam os comportamentos dos agentes na negociação de serviços. Além disso, possui a habilidade de esconder informações não necessárias aos desenvolvedores da aplicação, como a utilização de scripts de instanciação de agentes e ambientes, como o processo de negociação dos agentes e mecanismo de leitura da instância da ontologia.

4.1. Visão Geral do Framework

O ContextualFramework visa a análise do processo de negociação de serviços em redes sem fio de nova geração. Para tanto, ele provê a modelagem genérica de um ambiente de simulação com a utilização de sistemas multi-agentes e de web semântica, utilizando o DynaCIP, que permitem a utilização e análise de informações de contexto que possam influenciar no processo de decisão de compra ou venda de um serviço em redes sem fio de nova geração.

Nesta solução, identificamos as seguintes partes fixas, *frozen spots* do *framework*:

- Modelagem da informação de contexto, utilização de ontologias segundo a proposta do DynaCIP;
- Instanciação dos Ambientes e dos Agentes da simulação;
- Processo de negociação baseado no Fipa ContractNet;
- O gerenciador de contexto.

Os *frameworks* possuem partes que são passíveis de extensão e customização (pontos de flexibilização), chamados *hots pots*. Nesta solução identificamos:

- A instância da ontologia e de regras do DynaCIP;
- Comportamento que implementa o algoritmo para tomada de decisão;
- Comportamentos Contextuais dos agentes que estendam a classe Behaviour do JADE e implementem a interface ContextualBehaviour do ContextualFramework;
- Acompanhamento da simulação.

Como principais características do framework, podemos identificar:

- Representação e analise de informações de Contexto. As informações de contexto são identificadas, seus valores e condições analisados e tarefas e ações realizadas por comportamentos dos agentes durante a simulação.
- Organizações e Ambientes são representados numa estrutura Top-Down na instanciação das classes do DynaCIP. Permitindo, por

exemplo, através da utilização de regras de composição, que organizações herdem propriedades de organizações superiores.

- Viabiliza o processo de negociação entre os Agentes da simulação. O processo de negociação é viabilizado através da utilização do protocolo Contract-Net proposto pela FIPA.
- Permite o comportamento migratório de Agentes clientes. Agentes podem mover entre localidades e ambientes da simulação. Este comportamento permite representação de um usuário real com um dispositivo móvel, por exemplo, a localização em que ele se encontra e as propriedades e características das redes também mudam.
- Auxilia os agentes a tomar decisões na negociação de um serviço. Tanto os agentes provedores quanto os agentes clientes utilizam um algoritmo baseado na leitura da instância do DynaCIP e na análise de informações de contexto para tomar suas decisões. Os provedores, por exemplo, no momento da definição de preços de seus serviços, e os clientes, no momento da definição do melhor serviço na situação em que ele se encontra.
- Gera um output/log para analisar o desempenho dos agentes no processo de negociação. Este output pode ser utilizado para geração de um script de inserção em um repositório de dados. Estes dados armazenados podem ser utilizados para a geração de relatórios e gráficos de desempenho.

4.2. Arquitetura

O Contextualframework, desenvolvido em JAVA sob a plataforma JADE, se divide em blocos identificados como Setup (processo de inicialização do ambiente de simulação), Market (onde ocorre a negociação dos serviços entre clientes e provedoras), Ontology (onde estão representados as classes e as instâncias do ambiente da simulação) e Monitoring (onde todas as ações realizadas pelos agentes clientes e provedoras são registradas para futura análise).

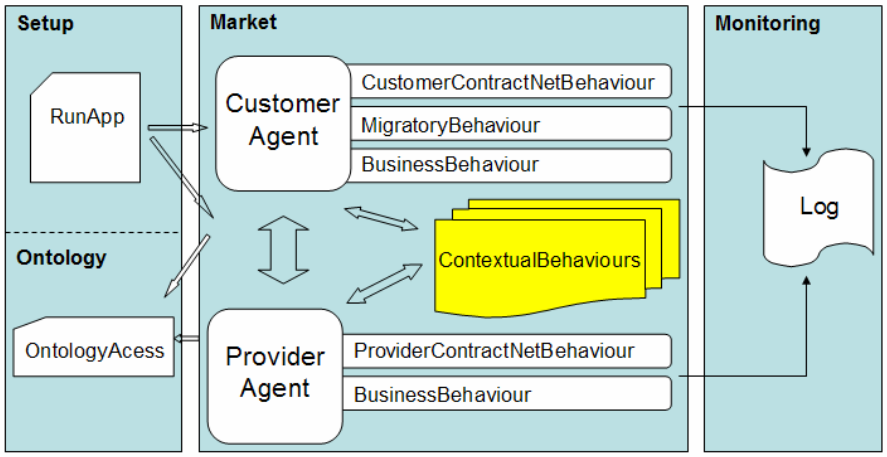

Figura 13. Arquitetura do ContextualFramework.

Na fase de inicialização identificada pelo bloco Setup, a classe RunApp se encarrega de instanciar as localidades (representadas por Containers do JADE), os agentes clientes (CustomerAgent) e os agentes provedores (ProviderAgent), através de uma leitura da ontologia da aplicação.

Depois de instanciados, os agentes iniciam o processo de negociação de serviços, realizado no bloco Market, onde os clientes através do comportamento CustomerContractNetBehaviour iniciam o processo de negociação para a compra de um determinado serviço baseado no protocolo FIPA ContractNet.

O processo de negociação se baseia num algoritmo de decisão definido no Comportamento BusinessBehaviour, responsável pela interpretação dos protocolos e suas diretivas. Este Comportamento pode ser estendido caso o algoritmo proposto pelo framework não atenda as necessidades da aplicação.

Os agentes clientes podem também utilizar o comportamento MigratoryBehaviour, caso o comportamento migratório seja interessante, para simular o deslocamento de uma localidade para outra de forma periódica e randômica. Já as provedoras utilizam o comportamento ProviderContractNetBehaviour para negociarem com os clientes os serviços que estão oferecendo, utilizando também o protocolo FIPA ContractNet. De forma semelhante aos agentes clientes, os agentes provedores utilizam o comportamento BusinessBehaviour para acessarem de forma cíclica a ontologia para interpretarem as informações de contexto (através do bloco Ontology com utilização da classe OntologyAccess) e as regras necessárias para simulação do ambiente.

O Comportamento BusinessBehaviour, tanto para os clientes quanto para as provedoras, será o responsável por chamar os comportamentos ContextualBehaviour implementados pelas aplicações, após a leitura das diretivas de um agente, descritas na ontologia do DynaCIP. Estes comportamentos são identificados através da informação de contexto que estão tratando. A informação de contexto é avaliada segundo uma condição, por instâncias de ContextualBehaviour,

Todas as ações, de todos os agentes podem ser acompanhadas em tempo de execução e são registradas em um arquivo de saída, realizado no bloco Monitoring para análise da dinâmica da simulação.

4.3. Pontos fixos

Neste tópico serão descritos os pontos fixos, pontos já implementados que facilitam as aplicações que instanciam o *framework*. Com as características de implementação e funcionamento do framework.

4.3.1. Modelagem da Informação de Contexto

O ContextualFramwork modela a informação de contexto segundo a proposta do DynaCIP. Porém, utiliza a abordagem do DynaCIP de uma maneira um pouco diferente da proposta.

A modelagem do domínio em uma estrutura Top-Down, e sua representação segundo a ontologia proposta, com o uso de regras de regras de composição, são utilizados no *framework*. Porém a ontologia inferida não é adicionada em um único comportamento a ser inserido nos agentes. A ontologia inferida é utilizada para a instanciação dos agentes e seus comportamentos, e das localidades da simulação. O *framework* realiza uma leitura constante da ontologia para monitorar e redefinir os comportamentos dos agentes durante a simulação.

No *framework* proposto as Organizações são representadas não só por agentes fornecedores, mas também por agentes clientes. Na ontologia proposta pelo DynaCIP podemos identificar os Ambientes, que aqui são mapeados em localidades, que serão habitadas pelos agentes.

Na solução os agentes clientes consomem os recursos oferecidos pelos agentes Fornecedores. Estes recursos são negociados como Serviços, com um preço base que pode ser alterado após analise de informações de contexto.

Um serviço pode ter uma utilização nominal de recursos da provedora definida. Na solução, a Descrição de Serviços, como o próprio nome diz, descreve os serviços de uma maneira padrão, para que os clientes sempre identifiquem os serviços de forma clara entre as diversas propostas das provedoras.

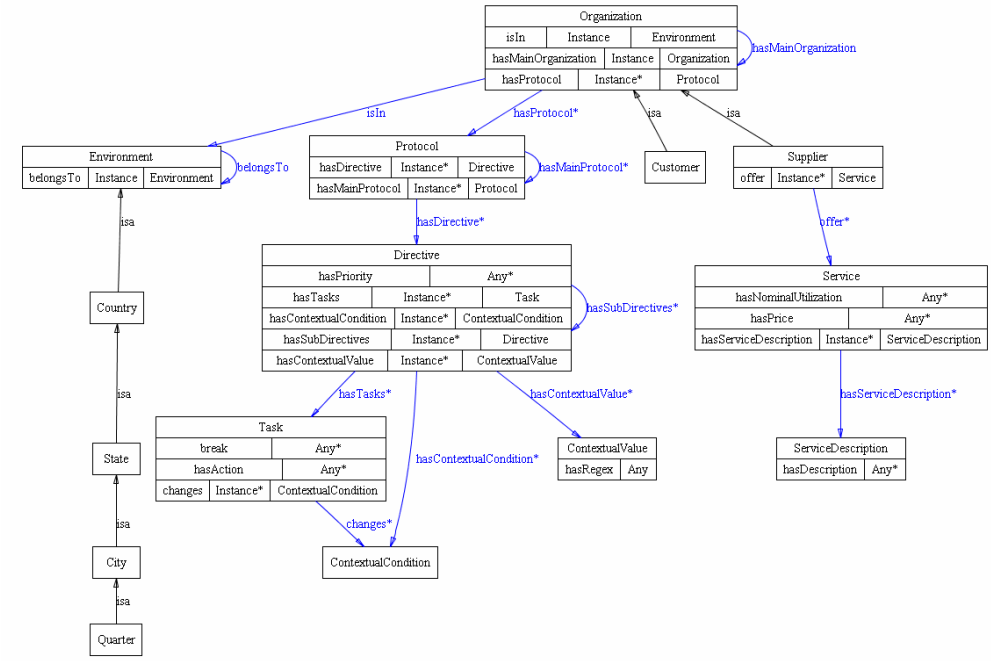

Figura 14. Ontologia de classes do DynaCIP.

O ContextualFramework adicionou também ao modelo de classes da ontologia proposta pelo DynaCIP, três novas entidades que caracterizam uma Diretiva: Condição Contextual (ContextualCondition), Valor Contextual (ContextualValue) e Tarefa (Task). O Valor Contextual define um ou um grupo de valores, que deve ser avaliado por uma Condição Contextual. Uma Condição Contextual identifica e mapeia um comportamento do agente, este comportamento é uma instância de ContextualBehaviour do *framework*. Este comportamento trata uma ou mais informações de contexto. Quando uma informação de contexto atinge os valores definidos por um Valor Contextual, um Comportamento Contextual definido pelo algoritmo de tomada de decisão deverá executar uma Tarefa.

A solução aqui proposta, usa o DynaCIP pois este viabiliza a utilização de informação de contexto em sistemas multi-agentes de uma maneira clara, objetiva e intuitiva. Com Organizações, Ambientes, Protocolos e Diretivas determinando o comportamento dos agentes no domínio da aplicação. Uma alteração na abordagem, com mais de um comportamento do agente utilizando o modelo proposto, e a adição de algumas classes na ontologia, foram feitos para permitir uma maior representatividade do domínio.

4.3.2. Instanciação do Ambiente e dos Agentes

No processo de inicialização do ambiente, a classe RunApp do bloco Setup irá identificar as propriedades da aplicação, definidas no arquivo de propriedades, e instanciará o as localidades, os agentes e os comportamentos dos agentes da simulação.

No arquivo de propriedades serão definidas algumas características da simulação, como o período cíclico de leitura da ontologia, isto permite que a instância da ontologia seja alterada e consequentemente o comportamento dos agentes também, a cada nova leitura. Os comportamentos podem ser instanciados em tempo de execução, por reflexão, proporcionando mais flexibilidade a simulação.

Pode-se definir também o tempo de migração dos agentes clientes, caso este comportamento seja interessante. O agente migra de uma localidade para outra simulando a mobilidade de um usuário com um dispositivo móvel. A nova localidade é definida aleatoriamente, sendo escolhida uma localidade dentro das disponíveis na simulação.

Definem-se os arquivos de ontologia e regras utilizados na simulação. Assim como o arquivo de saída da aplicação, onde serão exportados os resultados da simulação. Relatórios poderão ser gerados com estes resultados obtidos.

```
### ContextualFramework: Application properties ###

#Read ontology milisecond
ontologyTime=400000

#CustomerAgent migration time
migratoryTime=13300

#Application instance ontology
ontology=file:/C:/Programs/ContextualNGWN/ContextualNGWN.owl

#Application rule file
ruleFile=file:/C:/Programs/ContextualNGWN/composeConcepts.rules

#Application output file
appOutputFile=file:/C:/Programs/ContextualNGWN/ContextualNGWNOutputFile.txt

#Application path
appPath=file:/C:/Programs/ContextualNGWN/

#output classpath
appOutputclass=bin/

#Aplication behaviours path
appBehavioursPath=behaviour
```

Figura 15. Arquivo de propriedades.

Define-se também onde estão os arquivos da aplicação, as classes de comportamento que serão instanciadas para os agentes. A localização destes arquivos é importante, pois a instanciação dos comportamentos é realizada por reflexão com a identificação da localização/caminho das classes de comportamento, os Comportamentos Contextuais.

O bloco Ontology é o responsável pela leitura cíclica da ontologia, e instanciação dos Comportamentos Contextuais dos agentes definidas na ontologia. A ontologia define uma Condição Contextual que é mapeada para um Comportamento Contextual do agente.

A leitura da ontologia definirá os Ambientes mapeadas em localidades (containers do JADE) e Organizações que são mapeados em agentes, ambos são instanciados, iniciando-se a simulação.

4.3.3. O Processo de Negociação dos Serviços

Assim que o agente é instanciado, uma de suas primeiras ações é se registrar no DFService da Plataforma JADE, o serviço de páginas amarelas do JADE. O CustomerAgent ao chegar a uma localidade informa quem ele é, seu nome, e de que tipo ele é, um agente cliente. Sempre que migra de uma localidade para outra, o CustomerAgent atualiza sua localização no serviço de páginas amarelas. O ProviderAgent de forma semelhante informa quem ele é, seu nome, de que tipo

ele é, um agente provedor de serviços, e quais os serviços que está oferecendo (os serviços oferecidos podem variar de acordo com a localidade). Ao contrário do CustomerAgent, o ProviderAgent não tem um comportamento migratório, se registrando uma única vez no serviço de páginas amarelas.

O registro dos agentes no serviço de páginas amarelas é fundamental para a dinâmica da simulação. É através deste serviço que o CustomerAgent irá identificar todos os ProviderAgent que estão na localidade que ele se encontra e que oferecem um determinado tipo de serviço que está sendo requisitado por ele. Ou seja, o CustomerAgent solicita uma lista de provedoras que ofereça o serviço que ele estará solicitando naquela localidade, evitando desta forma enviar uma solicitação de proposta de serviço para provedoras que não atendam ao serviço que está sendo requisitado.

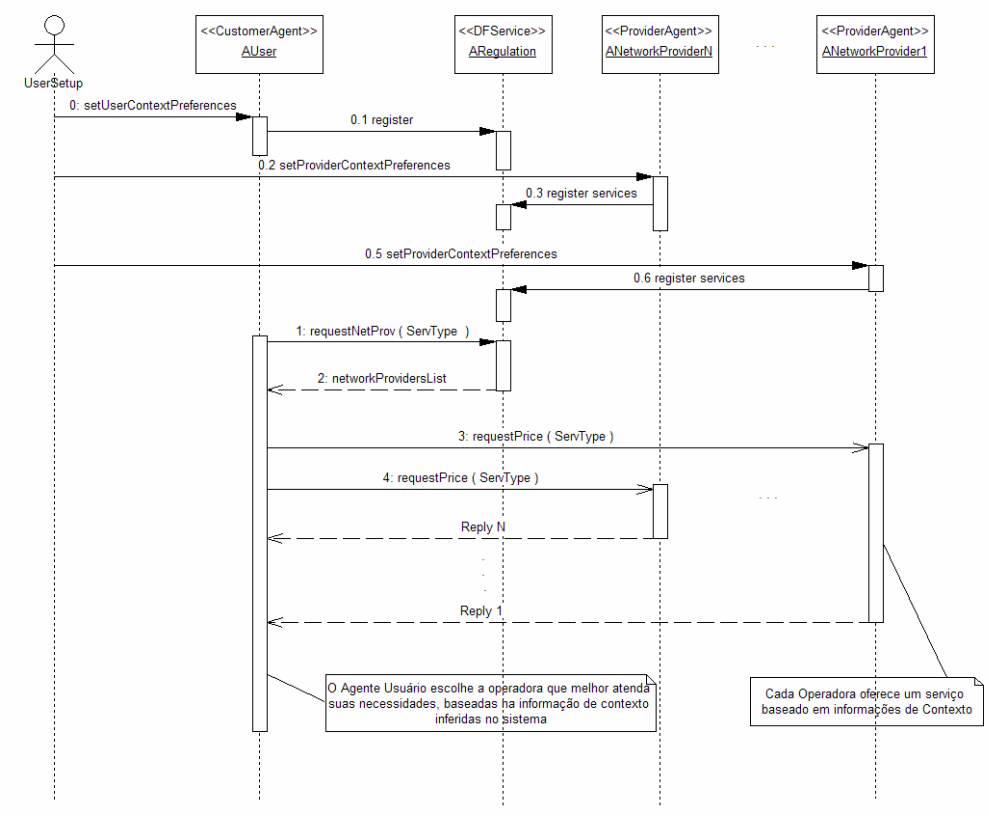

Figura 16. Dinâmica da simulação.

Com a lista de provedoras que podem atender sua requisição de serviço, o CustomerAgent inicia um processo de negociação baseado no protocolo ContractNet. Cada provedora calculará seu preço de acordo com as informações de Contexto, como por exemplo, o nível de congestionamento e utilização atual da

rede, o número de clientes na localidade de requisição do serviço e a qualidade de serviço que a provedora está oferecendo naquela localidade. Já o cliente aceitará uma solução que melhor atenda às suas necessidades. Ele avalia informações de Contexto como a localização que ele se encontra, o tipo de serviço que está requisitando e o número de clientes que se encontra em sua localidade. Após análise destas informações, define qual proposta escolher, como por exemplo, a de menor preço, melhor qualidade de serviço ou menor tempo de resposta e rejeitará as demais propostas recebidas.

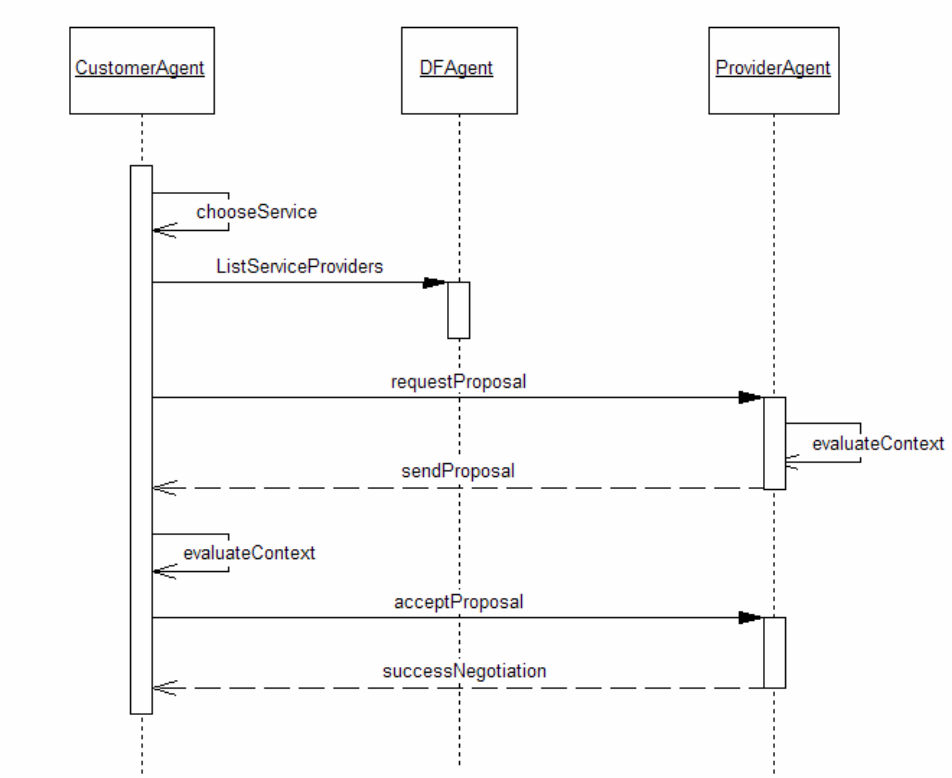

Figura 17. O processo de negociação baseado no ContractNet.

No processo de negociação, o CustomerAgent envia uma requisição de preço de um determinado tipo de serviço ao ProviderAgent, que então verifica se tem capacidade de rede disponível (resourceUtilization) para atender a solicitação, em caso afirmativo começa a calcular o preço alterando o preço base de acordo com informações contextuais, como por exemplo, o número de clientes (numOfCust), a qualidade de serviço (NetQoS) e a utilização da rede (netUtilization) enviando preço final como proposta ao cliente. O cliente então aguarda o recebimento de todas as propostas e escolhe aquela que melhor atenda às suas necessidades, enviando um aceite de proposta à provedora escolhida e

rejeitando as demais propostas. Ao receber o aceite, a provedora reserva capacidade para o cliente e envia uma mensagem confirmando a negociação do serviço com sucesso.

4.3.4. Gerenciador de Contexto

O Gerenciador de Contexto, no ContextualFramework, é um papel executado por um Comportamento de Negócio, chamado Business Behaviour, que tem a função de gerenciar as informações de contexto utilizadas por um agente.

O Comportamento de Negócio é o responsável por definir e aceitar propostas durante a negociação de serviços. A definição e o aceite de uma proposta normalmente são feitos após o gerenciador de contexto de um agente analisar as informações de contexto que influenciam no seu negócio.

O Gerenciador de Contexto é solicitado assim que uma mensagem de requisição de proposta ou de aceite de negócio é recebida pelo agente provedor ou cliente, respectivamente. O Gerenciador de Contexto irá então executar o seu algoritmo para tomada de decisão baseado na leitura dos Protocolos e Diretivas definidos na instância do DynaCIP e na análise das informações de contexto. O resultado da decisão do algoritmo, com a proposta ou aceite de negócio será enviado através de uma mensagem de resposta da negociação.

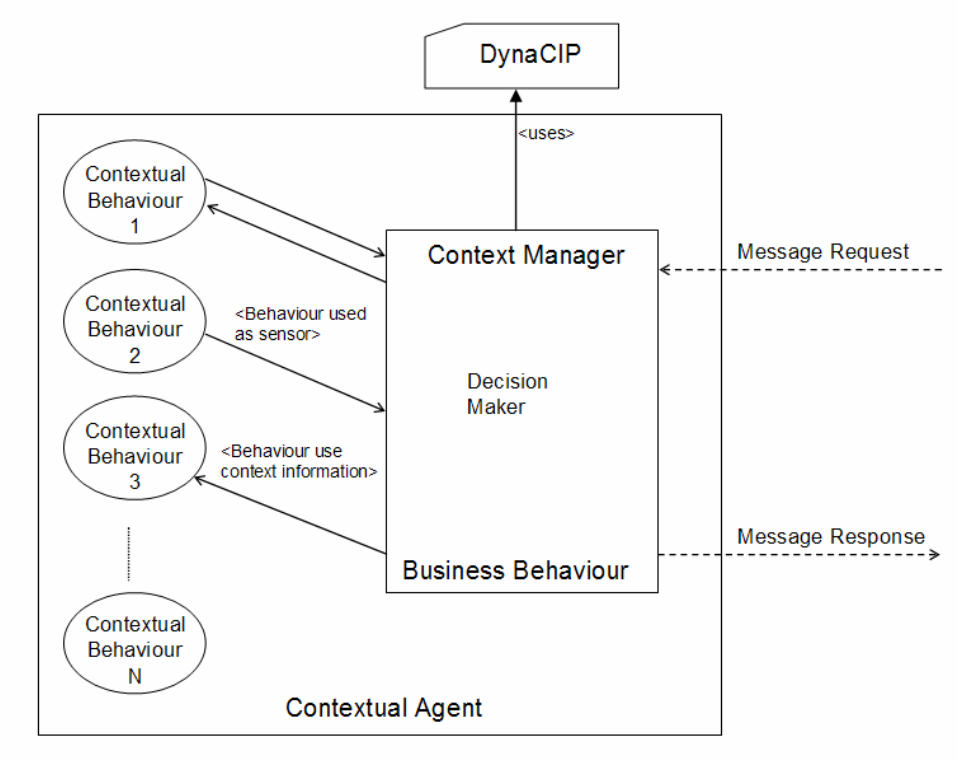

Figura 18. Gerenciador de Contexto.

Cada Diretiva ou Sub-Diretiva poderá ter uma Condição Contextual que está diretamente associada e mapeada para um Comportamento Contextual do agente. Sendo assim, quando o algoritmo identifica um Valor de Contexto a ser investigado, o Gerenciador de Contexto solicita ao Comportamento Contextual mapeado pela Condição Contextual a análise do mesmo.

Cada Comportamento Contextual poderá funcionar como uma fonte de contexto e/ou como um consumidor de contexto para a realização de uma Tarefa. Quando funcionar como fonte de contexto, o Comportamento Contextual deverá ficar monitorando uma informação de contexto, como um sensor de uma informação de contexto para o gerenciador. Quando funcionar como consumidor de contexto, este normalmente irá executar uma Tarefa após uma tomada de decisão do Gerenciador de Contexto. O Comportamento Contextual também poderá executar ambas as funções, de fonte e consumidor de contexto dependendo da aplicação e da implementação da instância do DynaCIP.

O Gerenciador de Contexto tem a função de agregar, armazenar e sintetizar as informações de contexto, permitindo que o agente tome decisões baseadas nestas informações de contexto. Permitindo consequentemente a comercialização de serviços entre agentes baseados em informação de contexto.

4.4. Pontos de Flexibilização

Neste tópico serão descritos os pontos flexíveis do framework. Pontos que devem ser implementados pelas aplicações que instanciam o framework. As características do domínio da aplicação são definidas nestes pontos de flexibilização.

4.4.1. A criação da instância da ontologia e regras do DynaCIP

Embora a estrutura de classes de ontologia seja fixa, e definida pela abordagem do DynaCIP com Organizações, Ambientes, Protocolos e Diretivas. Onde nenhuma outra classe pode ser definida pela aplicação. A instância da ontologia do DynaCIP é um ponto de flexibilização do framework.

Na instância, podem-se definir quais e quantos são os Ambientes, mapeados em localidades, que serão habitadas por agentes da simulação. Definem-se também quais e quantos são os agentes da simulação, onde se localizam inicialmente os agentes clientes e definitivamente os agentes provedores. Devido à mobilidade os agentes clientes são instanciados numa localidade, mas podem mudar de localidade durante a simulação. Já os agentes provedores ficam fixos em uma localidade até o final da simulação.

```
−<Supplier rdf:ID="CellNetProvider1">
  +<offer></offer>
   <offer rdf:resource="#CellNetProvider1InternetAccess_200K"/>
  −<isIn>
    −<State rdf:ID="RJ">
      −<belongsTo>
        −<Country rdf:ID="Brazil">
          −<belongsTo>
              <Country rdf:ID="SouthAmerica"/>
           </belongsTo>
         </Country>
       </belongsTo>
     </State>
   </isIn>
  −<hasMainOrganization>
    +<Supplier rdf:ID="CellNetProvider1_Corp"></Supplier>
   </hasMainOrganization>
  +<offer></offer>
  +<hasProtocol></hasProtocol>
 </Supplier>
−<Customer rdf:ID="Carlos">
   <hasProtocol rdf:resource="#Carlos1_ForChooseSupplier"/>
   <isIn rdf:resource="#Centro"/>
   <buy rdf:resource="#SendSMS"/>
 </Customer>
−<Quarter rdf:ID="Tijuca">
   <hasOrgId rdf:datatype="http://www.w3.org/2001/XMLSchema#string">4</hasOrgId>
   <belongsTo rdf:resource="#Rio_de_Janeiro"/>
 </Quarter>
```

Figura 19. Exemplo de Fornecedor, Cliente e Localidade de uma instância da ontologia do DynaCIP.

Regras de composição para a instância da ontologia também podem ser definidas pela aplicação. Regras de transitividade e simetria são definidas para permitir composição das classes definidas pelo DynaCIP.

```
[ruleForOrgWithMainOrgProt:
                (?Organization ontologyURI:hasMainOrganization ?MainOrg)
                (?MainOrg ontologyURI:hasProtocol ?MainOrgProt)
                 -> (?Organization ontologyURI:hasProtocol ?MainOrgProt) ]

[ruleForProtWithMainProtDir:
                (?Protocol ontologyURI:hasMainProtocol ?MainProt)
                (?MainProt ontologyURI:hasDirective ?MainProtDir)
                 -> (?Protocol ontologyURI:hasDirective ?MainProtDir) ]

[ruleForOrgWithOrgProtDir:
                (?Organization ontologyURI:hasProtocol ?Protocol)
                (?Protocol ontologyURI:hasDirective ?ProtDir)
                 -> (?Organization ontologyURI:hasDirective ?ProtDir) ]

[ruleForProtWithProtDirCond:
                (?Protocol ontologyURI:hasDirective ?Directive)
                (?Directive ontologyURI:hasContextualCondition ?DirCond)
                 -> (?Protocol ontologyURI:hasContextualCondition ?DirCond) ]
```

Figura 20. Exemplo de Regras de Composição da Ontologia.

As regras de composição permitem uma leitura direta de uma instância de uma classe da ontologia e das instâncias de classes que a compõem. Por exemplo, podem-se identificar com uma leitura simples, quais são os protocolos de uma Organização que foram herdados de uma Organização pai. Estas regras diminuem a necessidade de criação de algumas linhas de código para representar a composição das instâncias da aplicação e principalmente torna mais fácil e pratica esta composição.

4.4.2. Algoritmo para tomada de decisão

Durante o processo de negociação de um serviço, tanto o agente provedor quanto o agente cliente tomam suas decisões baseados num algoritmo para tomada de decisão do *framework*. A aplicação pode utilizar o algoritmo proposto pelo framework ou definir um novo algoritmo que atenda melhor as necessidades e características da aplicação.

O objetivo da proposta é permitir a análise de diferentes algoritmos de tomada de decisão, não forçando uma solução única no processo de decisão. O algoritmo proposto pelo framework foca na utilização da abordagem do DynaCIP,

com a utilização de informações de contexto para a tomada de decisão de uma maneira simples e objetiva. Desempenho não é o foco do algoritmo proposto, podendo esta ser uma propriedade implementada por algoritmos específicos e ser adicionado ao processo de execução do framework através de um de seus *hotspots*.

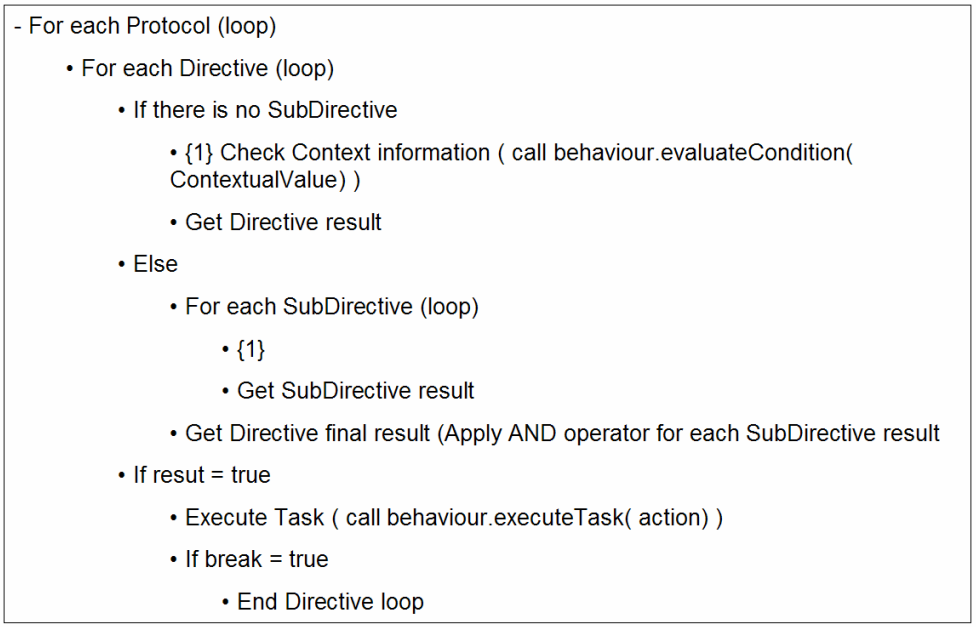

Figura 21. Algoritmo para tomada de decisão.

O algoritmo é executado no momento em que os agentes precisam tomar uma decisão. Normalmente quando o agente cliente precisa escolher qual é o melhor serviço disponível entre os oferecidos pelas provedoras. E as provedoras quando precisam definir uma proposta de serviço para uma requisição de um cliente.

O primeiro passo para tomada de decisão é avaliar os Protocolos dos agentes. Para cada Protocolo deve se fazer uma leitura de suas Diretivas, e para cada Diretiva deve-se avaliar a existência de sub-Diretivas.

Caso não exista uma sub-Diretiva, a Diretiva solicita a um Comportamento específico do agente, identificado pela Condição Contextual que se está avaliando, para analisar se a informação de Contexto desejada está dentro do limite definido pelo Valor Contextual.

Caso existam sub-Diretivas, para cada sub-Diretiva deve-se executar um passo semelhante ao descrito anteriormente, com a analise da informação de contexto comparando-a ao Valor Contextual, trabalho realizado pelo

Comportamento mapeado pela Condição Contextual que se está avaliando. Cada Comportamento solicitado por uma sub-Diretiva retornará um valor.

O resultado final obtido pela Diretiva será o retornado por um único Comportamento solicitado quando não existirem sub-diretivas, ou quando existirem, da combinação de todos os resultados obtidos pelas sub-diretivas segundo o operador "E" da lógica booleana.

Se o resultado final obtido for verdadeiro, uma ação definida por uma Tarefa específica será executada por um Comportamento determinado pela Diretiva. Esta ação pode ter um impacto parcial ou definitivo na decisão a ser tomada. Parcial quando a ação apenas altera uma variável, executa uma função de pontuação. Definitivo quando a decisão já foi tomada, criando-se neste caso uma condição de parada. Caso a Tarefa possua uma condição de parada, as demais Diretivas do Protocolo em execução não serão analisadas após a realização da ação.

O algoritmo propõe uma análise de informações de contexto, orientado pela leitura da instância de classes definida pelo DynaCIP, para que o agente cliente ou provedora tome uma decisão mais adequada a situação em que ele se encontra no momento de sua decisão.

4.4.3. Comportamentos Contextualizados

Um dos pontos flexíveis do ContextualFramework é a possibilidade e liberdade de criação de Comportamentos específicos para cada agente da simulação. Cada Comportamento pode ser implementado para tratar uma ou mais informações de contexto, este é chamado de Comportamento Contextualizado.

A criação destes Comportamentos são baseados na extensão da classe Behaviour da API do JADE e na implementação da interface ContextualBehaviour do ContextualFramework, com seus métodos para avaliação de uma condição e execução de uma tarefa.

Estes Comportamentos recebem solicitações do Comportamento de Negócio para avaliarem um Valor Contextual, segundo as informações de contexto que estão monitorando e/ou para executarem uma Tarefa após tomarem uma decisão.

Os Comportamentos utilizam o método de avaliação de condição para identificar se suas informações de contexto estão dentro do Valor Contextual esperado. O Valor Contextual é uma expressão regular que define os limites das informações de contexto que se deseja encontrar.

Os Comportamentos utilizam o método de execução de tarefas para realizarem ações após uma tomada de decisão. A ação deve ser implementada pelo Comportamento. Esta ação pode ser definitiva, ou seja, a decisão propriamente dita, com o resultado enviado através de uma mensagem de resposta ao solicitante. Ou ser uma ação parcial, ou seja, a ação apenas contabiliza pontos, atribui pesos ou altera uma variável.

Quando a ação é definitiva, a decisão propriamente dita, normalmente as demais Diretivas e Sub-Diretivas que ainda não foram lidas pelo algoritmo de tomada de decisão são abandonadas e não executadas. Neste caso, uma condição de parada, o Break, é definido na Tarefa solicitada.

Quando a ação é parcial, todas as Diretivas e Sub-Diretivas são executadas e nenhuma descartada. Neste caso, a Diretiva que tem o seu velor contextual satisfeito, solicitará a execução de uma tarefa com uma ação específica, como uma alteração de uma variável. A decisão neste caso será o resultado da observação das alterações sofridas por esta variável.

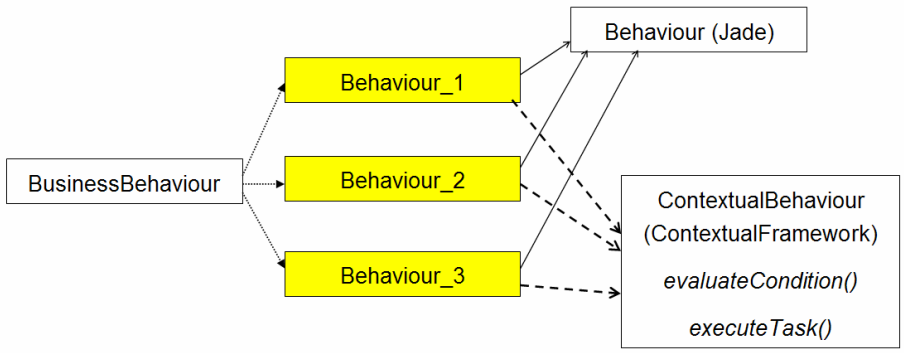

Figura 22. Comportamentos Contextualizados.

Uma das características do ContextualFramework é permitir a instanciação de Comportamentos Contextualizados em tempo de execução. Durante a simulação novos Comportamentos podem ser adicionados aos agentes permitindo que novas informações de contexto sejam analisadas, ou mesmo, informações de contexto não interessantes sejam descartadas.

O framework realiza constantemente uma leitura da instância da ontologia do DynaCIP. Qualquer alteração na instância da ontologia durante a simulação poderá introduzir mudanças no comportamento e na dinâmica da simulação.

Pode-se, por exemplo, adicionar uma nova Diretiva no Protocolo de um agente com uma nova Condição Contextual a ser analisada. Esta nova Condição Contextual, segundo o modelo proposto, deverá estar mapeada a um novo Comportamento que deverá ser inserido no agente.

O processo envolve a identificação de uma nova Condição Contextual definida na ontologia para um agente específico e por conseqüência a instanciação da classe de Comportamento correlacionada por reflexão, em tempo de execução, e na adição deste Comportamento ao agente instanciado, ativo na simulação.

A leitura e verificação constante das Condições Contextuais da ontologia, analisadas por um agente durante a simulação, permitem que este adquira novos Comportamentos Contextualizados e remova Comportamentos Contextualizados não interessantes também durante a simulação.

O Comportamento Contextualizado ao ser instanciado é adicionado ao agente através da API do JADE e também é adicionado a uma lista de Comportamentos Contextualizados gerenciada por cada agente. A idéia é que o Comportamento de Negócio solicite a avaliação de uma informação de contexto a um Comportamento Contextualizado presente em sua lista.

Esta lista é sempre recriada a cada leitura da instância da ontologia, os Comportamentos antigos não são retirados do agente através da API do JADE são apenas removidos da lista, dando lugar aos novos Comportamentos recém instanciados.

Os Comportamentos antigos podem estar realizando tarefas solicitadas pelos Comportamentos de Negócio durante a recriação da lista de Comportamentos Contextualizados, por isso estes não são totalmente removidos dos agentes.

Comportamentos Contextualizados antigos são totalmente removidos dos agentes quando terminam sua execução e identificam que não fazem mais parte da lista corrente de Comportamentos Contextualizados. Estes Comportamentos passam a não ser mais referenciados, apenas aguardando a ação do coletor de lixo do Java para serem apagados.

A remoção total dos Comportamentos Contextualizados antigos do agente e do sistema, podem ter uma implementação mais eficiente se necessário pela

aplicação, uma vez que estes Comportamentos são criados e instanciados pelas aplicações.

4.4.4. Acompanhamento da simulação

Os resultados obtidos pela simulação podem ser acompanhados em tempo de execução, por uma tela que exibe a negociação entre os agentes, ou através de um arquivo com as informações importantes da aplicação. Como características das propostas, aceitações e transações realizadas durante a simulação.

O formato das informações, com os resultados da simulação exportados para o arquivo de saída, deve ser definido pela aplicação. O arquivo gerado pode ser importado em uma planilha ou repositório de dados que possibilite a geração de relatórios da simulação.

Pode-se exportar, por exemplo, todas as propostas realizadas durante uma simulação, com os dados do provedor, a descrição do serviço, local e data da proposta e os detalhes desta, como o preço do serviço, a qualidade de serviço, entre outras informações solicitadas pelo cliente.

As requisições de propostas dos clientes também podem ser exportadas, com as informações do cliente, local e data da solicitação e a descrição do serviço solicitado, por exemplo.

Uma das principais análises de desempenho dos agentes pode ser obtida com a interpretação das propostas aceitas durante a simulação. O registro das aceitações de propostas, com os dados do provedor que fez a proposta, do cliente que aceitou a proposta, com a descrição do serviço, o motivo da aceitação da proposta (como, por exemplo, o menor preço ou a melhor qualidade de serviço), além da localidade e data da aceitação. São exemplos informações que podem ser utilizadas para análise.

O desempenho das provedoras com análise do total de propostas aceitas sobre o total de propostas realizadas, de preço e qualidade de serviço oferecido em propostas aceitas e recusadas, de serviços mais vendidos por localidade, do perfil do cliente por proposta aceita dependendo da localidade, do serviço e da qualidade oferecida, são exemplos de parâmetros de desempenho que podem ser comparados ao longo de uma ou várias simulações.

O desempenho dos clientes pode ser observado coma análise do motivo de aceitação das propostas e os valores das propostas aceitas. Por exemplo, podem-se analisar os preços pagos pelos clientes quando estes definem como motivo de aceitação o menor preço de uma proposta, ou a qualidade dos serviços aceitos quando o motivo for a melhor qualidade do serviço.

O *handoff* vertical dos clientes pode ser analisado, por exemplo, com a observação das propostas que os clientes aceitaram em uma localidade para um mesmo serviço sequencialmente, ou seja, analisar quais foram as provedoras escolhidas numa localidade durante a utilização de um serviço e quais foram os motivos de escolha destas provedoras, com os detalhes desta proposta. A análise da proposta escolhida num total de propostas realizadas para uma solicitação, permitirá identificar se os critérios para a tomada de decisão estão corretos e de acordo com a situação que um cliente se encontra.