4
A Solução

A solução proposta compreende em introduzir variabilidades no desenvolvimento de SMA, de modo a flexibilizar ainda mais o desenvolvimento e a abordagem de agentes de software. Para tanto é necessário que seja dado o suporte para a representação de tal variabilidade há modelagem dos agentes.

Apesar das linguagens atuais ajudarem no desenvolvimento de aplicações multi-agentes, estas não lidam com aspectos como: a representação de planos e ações abstratos. Um plano abstrato ocorre quando há um agente com características que determinam a existência de instâncias de planos distintos. A instância do plano que será utilizada na aplicação é determinada em função do escopo de atuação do sistema projetado. Desta forma é possível observar que o plano que será utilizado pelo agente varia em função da aplicação que está sendo instanciada, isto significa que estes planos possuem uma grande quantidade de ações distintas entre sí, o que determina a variabilidade interna a um agente de software.

Uma ação abstrata ocorre quando um agente pode executar uma ação de maneiras distintas dentro de um mesmo plano. Esta variabilidade de ações é determinada pelo escopo de atuação do plano do agente.

Apesar de a proposta ser independente da linguagem de modelagem de agente, será utilizada uma linguagem para que o sistema seja projetado o que tornara mais simples e direta a exposição do método proposto denominado GP-Method for Variability Agent Plan (GP-MVAP). A linguagem escolhida foi o Anote. Para a escolha da linguagem foi levada em consideração a dificuldade do uso em conjunto com o método GP-MVAP e a facilidade na utilização. Como o uso em conjunto do ANote e do GP-MVAP traz a simplicidade de apenas acrescentar *tag´s* em dois diagramas ANote para referência aos diagramas GP-MVAP, assim o uso em conjunto se tornou bastante simples e por isso a linguagem ANote foi escolhida para ilustrar a esta dissertação.

O GP-MVAP dará suporte à representação da variabilidade interna de um agente de software e proverá também um mecanismo para indicar os pontos de instanciação dentro do modelo de agentes. Além disso, é incluído um guia de como estes pontos devem ser instanciados a fim de facilitar o entendimento e o desenvolvimento de SMA com as características de variabilidade.

Para utilizar o GP-MVAP juntamente com uma linguagem devem ser analisados: os pontos de flexibilização de um agente, devem ser identificados nos diagramas da linguagem escolhida e estes pontos devem conter um gancho do modelo da linguagem para o modelo do método o qual se encarregará de representar as variabilidades do sistema.

Realizado tal estudo, é esperado que seja possível representar através da solução proposta linhas de produto de um sistema multi-agente através da representação de planos e ações abstratas de um agente, e com isto será possível representar tais variabilidades facilitando o desenvolvimento e o entendimento do sistema.

Técnicas de descrição e instanciação podem ser utilizadas para complementar a solução proposta, em agentes de software o método GP-MVAP visa desenvolver uma adaptação de uma dessas técnicas para ajudar na especificação de como deve ser feita à instanciação de um plano ou uma ação em agente.

Tais técnicas são em grande parte utilizadas para auxiliar na instanciação de um *framework*, já que a reutilização/instanciação de um *framework* está fortemente ligada ao tipo de *framework* (Oliveira, 2001) e geralmente o reutilizador do *framework* não é o desenvolvedor, o que faz necessária uma abordagem para facilitar o entendimento do *framework* e conhecimento de seus pontos de extensão.

A UML-F foi utilizada como base para criar as *tags* que servirão como marcação dos pontos de instanciação, a UML-F pode ser vista com mais detalhes em Fontoura (et al., 2002). A UML-F é uma extensão da UML para representar *frameworks* e especificar o que deve ser instanciado no *framework*.

4.1.
Problema Exemplo

A área médica foi escolhida como um ambiente de estudo, primeiro por se tratar de uma área extremamente rica no que diz respeito ao desenvolvimento de SMA e segundo por possibilitar uma interação relativamente complexa entre os agentes.

O objetivo do sistema desenvolvido é auxiliar um médico numa prescrição medicamentosa. Porém este tipo de procedimento é feito em setores distintos de um hospital que apresentam diferentes níveis de complexidade.

O setor de CTI (Centro de Tratamento Intensivo) de um hospital é o local onde são atendidos os pacientes mais graves; os demais setores são divididos em salas para o atendimento clínico, enfermaria e etc. Um hospital possui uma divisão clara entre pacientes. Essa divisão pode ser classificada de uma maneira informal como os "doentes de CTI" e os "doentes de não-CTI[2]". Nesta divisão pode-se perceber que geralmente os "doentes de CTI" necessitam de mais cuidados especiais que os demais pacientes, o que implica a execução de mais ações para o tratamento desses pacientes que os outros.

Os processos para o tratamento de uma doença são: análise, identificação da doença, definição de um medicamento, da sua dosagem e do tempo de tratamento; estes processos não mudam em função de um medicamento. Dentro do processo de tratamento de uma doença, o subprocesso que mostra uma boa variação é o processo de cálculo/definição de dosagem. Tal variação pode ser percebida quando o plano é alterado de CTI para "não-CTI".

A aplicação de controle de qualidade em prescrições médicas (CQPM) que será apresentada a seguir traz detalhes de uma abordagem "não-CTI" que serve como um ótimo estudo de caso para a abstração de planos de agentes de software. O anexo A traz mais detalhes do processo de prescrição e o ciclo de vida médico-paciente.

O projeto (CQPM) Controle de Qualidade em Prescrições Médicas tem como objetivo auxiliar o médico no desenvolvimento de uma prescrição de qualidade, para tanto o sistema tem conceitualmente quatro (4) agentes: "agente

[2] A nomenclatura doente não-CTI e doente CTI foi utilizada com o intuito de facilitar o entendimento, porém não é um termo médico.

medicamentoso" encarregado por identificar possíveis problemas que o medicamento pode causar ao paciente, "agente regulamentador" responsável por verificar se o medicamento está liberado para uso ou não, "agente cíclico" responsável por definir o período de uso de um medicamento, "agente dosador" encarregado por definir a dosagem adequada de um medicamento para um paciente. Porém a implementação do sistema é composta de cinco (5) agentes. Os cinco agentes implementados no sistema são denominados de: "agente medicamentoso", "agente regulamentador BD", "agente regulamentadorWeb", "agente cíclico" e "agente dosador".

A seguir serão definidos os pontos que o sistema deverar verificar para se ter uma prescrição de qualidade: ***regulamentação***, onde se deve observar se os medicamentos prescritos (genéricos ou não) estão em vigor e regulamentados[3]. A eficácia também deve ser verificada e para isto é levado em consideração o histórico do paciente e assim é possível evitar que, por exemplo, medicamentos que possam ter interação medicamentosa sejam prescritos concomitantemente, desta forma sempre que um paciente estiver fazendo uso de um medicamento será feita esta verificação com a finalidade de evitar algum tipo de problema. Além da verificação de interação medicamentosa é feita também a verificação de alergia, através do histórico do paciente, assim se o paciente é alérgico a algum medicamento o problema será detectado; ***períodos de administração*** devem ser observados, com base na posologia do medicamento, para evitar períodos longos e desnecessários de administração de determinado medicamento ou períodos muito curtos no qual o medicamento não alcance o efeito desejado e a ***dosagem*** que deverá levar em consideração a posologia do medicamento no caso de adultos, mas devera ser recalculada quando se tratar de crianças, e no caso de idosos a dosagem poderá ser recalculada ou apenas utilizar a posologia do medicamento dependendo do caso; para estes dois últimos maiores cuidados devem ser tomados.

Os agentes de software do sistema podem ser descritos segundo os processos de uma prescrição medicamentosa.

O processo de definição de um medicamento contido na fase de prescrição (figura 4) traz consigo o "agente medicamentoso" que é o encarregado de verificar

[3] A regulamentação pode ser verificada através da ANVISA.

possíveis interações medicamentosas e alergias do paciente, além de enviar os alertas ao médico, que são obtidos através das respostas dos demais agentes.

Este agente possui uma dependência relacionada aos "agentes regulamentadores BD e Web", que determina que se um dos agentes informar que um dado medicamento está suspenso o "agente medicamentoso", não fará as verificações de interação medicamentosa e de alergia.

O processo de análise contido na fase de diagnóstico traz ao sistema os "agente regulamentadorBD" e o "agente regulamentadorWeb".

O "agente regulamentadorBD" verifica no banco de dados se o medicamento está com o status suspenso, se estiver com este status o "agente regulamentador BD" retorna uma mensagem para o "agente medicamentoso" informando que o medicamento está suspenso. Caso o medicamento não tenha status suspenso o "agente regulamentador BD" delega a responsabilidade de verificar o status do medicamento ao "agente regulamentador Web".

Quando requisitado pelo "agente regulamentador BD", o "agente regulamentador Web" verifica o status do medicamento na "web". Caso o medicamento seja encontrado na lista de suspensos, é enviada uma mensagem informando ao "agente regulamentador BD" que o medicamento está suspenso. Este atualizará o banco de dados e enviará uma mensagem informando o "agente medicamentoso". Caso o medicamento não seja encontrado na lista de suspenso, enviar-se-á uma mensagem de falha ao "agente regulamentador BD" que informará ao "agente medicamentoso" a não suspensão do medicamento.

O processo de definição da dosagem na fase de prescrição traz à tona o "agente dosador".

O "agente dosador" que foi implementado, possui duas maneiras de calcular a dosagem de um medicamento: uma é a dosagem para crianças e a outra é a dosagem para idosos, além de verificar se a dosagem prescrita para pessoas na faixa etária de 13 a 64 está correta. Primeiro o agente verifica em que faixa etária o paciente se encontra. Após a verificação ele pode obter (no banco de dados) as fórmulas para calcular as dosagens mínimas e máximas de crianças e idosos ou apenas verificar no banco de dados qual é a posologia indicada do medicamento (para pessoas de 13 a 64 anos).

O "agente dosador" envia uma mensagem informando ao "agente medicamentoso" qual foi o resultado da verificação da dosagem.

Vale a pena ressaltar que este agente possui uma enorme variabilidade no cálculo das dosagens e no sistema (CQPM) foi abordado o mais comum, onde a dosagem é calculada em função do peso, porém em setores como CTI o cálculo da dosagem é mais complexo que o abordado aqui.

O processo de definição de tempo de uso na fase de prescrição traz consigo o "agente cíclico", que verifica se o período de uso do medicamento prescrito está dentro do padrão da sua posologia. Para tanto, faz uma consulta ao banco de dados. Após a verificação o agente envia uma mensagem informando ao "agente medicamentoso" se o período de uso prescrito está dentro do padrão ou não.

Neste ponto podem-se observar dois cenários para estudo da variabilidade do cálculo de dosagem: um paciente idoso chega a uma clínica e após analises o médico chega à conclusão de que deve ser utilizado um medicamento com uma dosagem de 15mg, um período de 3 semanas e uma freqüência de 2 vezes ao dia. Neste caso o médico definiu uma dosagem a partir da posologia do medicamento, ou seja, uma formula pré-definida. Num outro cenário um paciente idoso está no CTI e após analises o médico chega à conclusão de que deve ser utilizado o mesmo medicamento do cenário anterior, porém com uma dosagem de 20mg por 500ml de soro, um período de 3 semanas, uma freqüência de 2 vezes ao dia e com uma bomba de velocidade 10ml/mim. Neste caso o médico definiu uma dosagem a partir de cálculos mais complexos, pois este paciente esta em estado mais crítico.

Estes dois cenários apresentam uma variabilidade no plano do agente dosador um em CTI e outro em casos de não-CTI.

A seguir será apresentada toda a modelagem do sistema CQPM que foi feita com a linguagem ANote. Verificar prescrição é o principal objetivo do sistema por isso esta foi situada no topo do diagrama de objetivo do ANote. Para alcançar este objetivo será apresentado a modelagem para o caso de clínica médica.

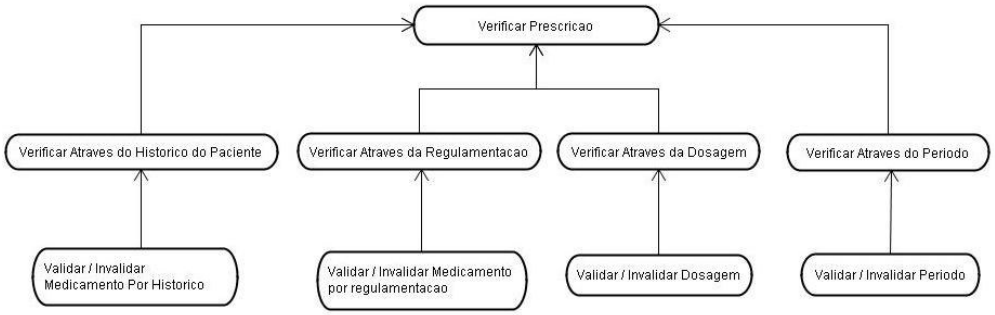

Figura 16. Decomposição dos objetivos relacionados com a prescrição.

A tabela a seguir serve de auxilio para a modelagem já que com ela é possível obter o mapeamento entre o agente e o subproblema que fica sobre sua responsabilidade.

Tabela 2. Mapeamento entre subproblemas e tipos de agentes.

Objetivo	**Agente**
Validar/Invalidar Medicamento por histórico do paciente	Agente Medicamentoso
Validar/Invalidar Medicamento por regulamentação	Agente Regulamentador BD e Agente Regulamentador Web
Validar/Invalidar Medicamento por dosagem	Agente Dosador
Validar/Invalidar Medicamento por período	Agente Cíclico

A seguir é apresentado o diagrama de agente do ANote onde é possível observar a como os vários tipos de agente se relacionam para obter uma melhor performance.

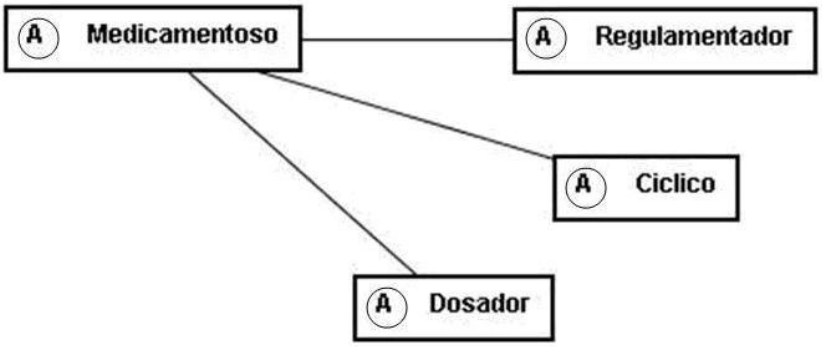

Figura 17. Arquitetura do sistema CQPM.

Seguindo a modelagem ANote será apresentado o diagrama *scenario view* dos agentes do sistema.

"Agente medicamentoso" recebe mensagem do "agente regulamentador BD" e do "agente regulamentador web" e a partir desta mensagem decide se continua o procedimento ou não. Caso afirmativo o "agente medicamentoso" deve verificar na base de dados a possibilidade de interação medicamentosa ou alergia.

Tabela 3. Diagrama *scenario view* do agente medicamentoso.

Validar/Invalidar Medicamento por Histórico	
Lead Agent:	Medicamentoso.
Precondition:	Existir um Medicamento e/ou um Princípio Ativo para validar.
Main Action Plan:	1. Selecionar Medicamento e Princípio Ativo. 2. Verifica interação medicamentosa do medicamento prescrito. 3. Verifica possível alergia do paciente ao medicamento prescrito. 4. Valida medicamento.
Interactions:	Regulamentador.
Variant Plan:	Variant PreCondition: Existir um Medicamento e/ou um Princípio Ativo prescrito. Plan Description: Se existe um Medicamento em uso em conjunto com o Medicamento prescrito pode gerar riscos a saude do paciente. 1.1. Agente Invalida Medicamento por Interação medicamentosa. Se o Medicamento prescrito está relacionado com alguma alergia do paciente. 1.1 Agente Invalida Medicamento por Alergia.

O "agente regulamentador" é dividido em dois agentes, o "agente regulamentadorBD" e o "agente regulamentadorWeb".

O “agente regulamentadorBD” verifica no banco de dados se o medicamento está com o status suspenso, já o “agente regulamentador Web” verifica o status do medicamento na “web”. Caso o medicamento seja encontrado na lista de suspensos, é enviada uma mensagem informando ao “agente regulamentador BD” para que seja atualizado o status do medicamento no banco.

Tabela 4. Diagrama *scenario view* do agente regulamentador.

Validar/Invalidar Medicamento por Regulamentação	
Lead Agent:	Regulamentador.
Precondition:	Existir um Medicamento e/ou um Princípio Ativo para validar.
Main Action Plan:	1. Selecionar Medicamento e Princípio Ativo. 2. Verificar no banco de dados o status do medicamento e/ou princípio ativo. 3. Verificar na web o status do medicamento e/ou princípio ativo. 4. Validar Medicamento.
Interactions:	Medicamentoso.
Variant Plan:	Variant PreCondition: O Agente não encontra o medicamento no banco de dados ou seu status está como suspenso. Plan Description: Se o medicamento esta com status suspenso no banco de dados. 1.1. Invalidar Medicamento. Se o medicamento não foi encontrado no banco e foi encontrado na lista de suspensos da Web. 1.2. O Banco de Dados é atualizado e o Medicamento invalidado.

O “agente dosador” é o responsável por calcular ou obter o valor da dosagem de um medicamento. O “agente dosador” envia uma mensagem informando ao “agente medicamentoso” informando a dosagem.

Vale a pena observer que este agente, já que, o calculo da dosagem possui uma variabilidade em parte das ações que compõem o calculo de uma dosagem.

Tabela 5. Diagrama *scenario view* do agente dosador.

Validar/Invalidar Medicamento por Dosagem	
Lead Agent:	Dosador.
Precondition:	Existir um Medicamento e/ou um Princípio Ativo para validar.
Main Action Plan:	1. Selecionar Peso, Idade, Dosagem. 2. Calcular a dosagem e ver se está de acordo com a dosagem Padrão. 3. Validar Dosagem.
Interactions:	Medicamentoso.
Variant Plan:	Variant PreCondition: O Agente encontra erros na dosagem. Plan Description: Se a dosagem está com problemas 1.1 Invalida o Dosagem.

O processo de definição de tempo de uso de um medicamento é desenvolvido pelo "agente cíclico", o período de uso é obtido em função da posologia do medicamento.

Tabela 6. Diagrama *scenario view* do agente cíclico.

Validar/Invalidar Medicamento por Período	
Lead Agent:	Cíclico.
Precondition:	Existir um Medicamento e/ou um Princípio Ativo para validar.
Main Action Plan:	1. Selecionar Período e freqüência. 2. Verificar se o Período de uso do medicamento esta de acordo com o Padrão. 3. Validar Período.
Interactions:	Medicamentoso.

Variant Plan:	Variant PreCondition: O Agente encontra erros na Definição de período. Plan Description: Se o período está com problemas 1.1 Invalida o Período.

Como foi observado o "agente dosador" é o principal agente do estudo de caso CQPM, pois tal agente contém os processos de cálculo da dosagem de um medicamento e estes processos variam segundo o plano que está sendo utilizado, ou seja, se o assunto em questão é o tratamento de um doente que está no CTI os processos de cálculo de dosagem são um caso contrário serão feitos outros processos para o cálculo de dosagem.

A idéia de abstração de planos de um agente será discutida em mais detalhes nas seções subseqüentes com base na aplicação médica (CQPM) descrita aqui nesta seção, o sistema CQPM pode ser visto mais a fundo em (Santos Pate, 2006).

4.2. Flexibilização de Plano

O estudo para introduzir a variabilidade no desenvolvimento de agentes de software traz consigo um pré-requisito, a inserção de noções de abstração, como ocorre no desenvolvimento de *frameworks* orientado a objetos.

A primeira idéia de abstração surge no domínio dos planos dos agentes.

Quando em um agente há a possibilidade de desenvolvimento de planos distintos para chegar a um objetivo comum, temos também a possibilidade de que as ações que compõem os planos sejam distintas.

Dentro de um plano pode ser visto um grande número de ações que devem ser executadas, a fim de que o objetivo do agente seja alcançado. Tais ações são muito bem definidas em função de um dado escopo. Este escopo geralmente é definido em função de uma aplicação que ser quer fazer. Porém aplicações distintas não implicam essencialmente em agentes distintos, ou seja, um agente pode ter o mesmo objetivo em varias aplicações e executar algumas ações

distintas para determinar as varias aplicações, o que nos leva a pensar em um mesmo núcleo de desenvolvimento para tais agentes.

Para resolver o problema de representação dos planos variáveis em agentes uma idéia é a utilização de restrições e esteriótipos nas visões do GP-MVAP. Os diagramas estendidos da linguagem ANote servem de gancho de modo a remeter a duas outras visões, que compõem o método GP-MVAP, denominadas de "*flexibility view*" e "*action note view*".

Para que o método GP-MVAP seja desenvolvido por completo os seguintes passos devem ser seguidos: inicialmente é preciso que, a partir da análise de requisito, seja verificado se realmente é possível ter duas ou mais aplicações para serem instanciadas em função de um mesmo objetivo (caso isto não seja confirmado o método não se aplica); em seguida é necessário definir claramente o plano base para as futuras instâncias, já que estas definirão as diferentes aplicações; o terceiro e ultimo passo será definir as ações que são utilizadas por todos os possíveis planos, ou seja, aquelas que serão utilizadas na construção do framework; no quarto passo serão definidas as ações para as instâncias das aplicações. Seguindo estes passos as visões definidas pelo método GP-MVAP são facilmente construídas.

A seguir são descritas características das visões que compõem o GP-MVAP (seus esteriótipos e restrições) e estão relacionadas à flexibilização de planos.

No nível de plano de agentes, pode-se utilizar a notação *{Incomplete}*, *Incomplete* é uma restrição utilizada para indicar que existe um plano abstrato e que este terá suas instâncias definidas por subplanos, ou seja, nem todos os subplanos foram ainda definidos e estes certamente são planos de instância que serão completados somente durante a instanciação.

O esteriótipo *{Instance Plan}* utilizado no diagrama *flexibility view* tem aplicação restrita apenas aos planos (assim como o *{Incomplete}*). Este esteriótipo mostra que o plano que deverá ser instanciado para uma dada aplicação. O plano que será instanciado deverá ser pintado de cinza, mostrando que ele é o plano que será concretizado e facilitando assim a percepção e minimizando a complexidade de quem vai desenvolver a aplicação. A figura a seguir representa o diagrama *flexibility view* que mostra um plano abstrato através do *flexibility plan* que é representado por vários retângulos e *plan1* e *plan2* representam os possíveis planos da aplicação, porém o que está sendo instanciado é pintado de cinza.

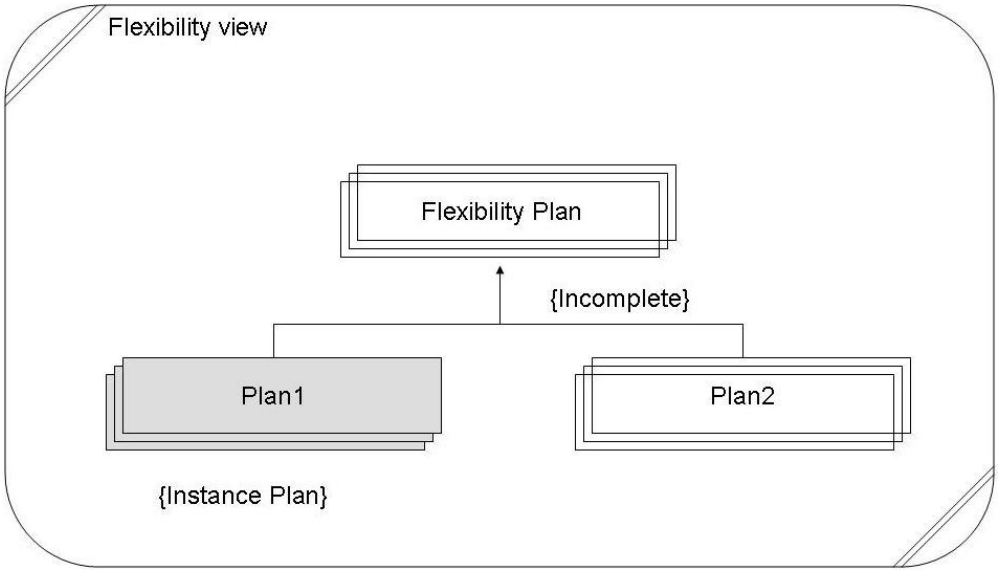

Figura 18. Diagrama *flexibility view*.

Ainda no nível de planos, o diagrama *action note view* pode ser utilizado como auxílio para a instanciação do plano, onde serão representados todos os processos necessários para que seja concluído satisfatoriamente o objetivo do agente, onde tais processos representam as ações do plano do agente. O *action note view* também utilizado na identificação de ações abstratas que serão discutidas mais à frente e com isto será possível determinar com mais clareza em que plano as ações devem ser executadas.

Uma restrição utilizada no *action note view* é *{both}*, onde sua semântica determina que a ação deve ocorrer em ambos os planos (abstratos e concretos). Assim os planos que serão instanciados herdarão do super plano uma implementação da ação. Esta restrição é utilizada em ações para determinar o comportamento que deve ser esperado e desta forma será possível auxiliar a instanciação dos planos.

A restrição *{concrete}* também é utilizada no diagrama *action note view*, o seu significado é que a ação que estiver indicada com esta restrição deve estar implementada no plano que será instanciado. Como a restrição anterior esta também é utilizada em ações, mas com o objetivo de auxiliar na instanciação dos planos.

A restrição *{relative}* também é utilizada no *action note view*, onde a sua semântica diz que a ação está presente em ambos os planos que poderão ser instanciados, porém a implementação da ação esta ligada ao plano que será

instanciado (daí identifica-se a ação abstrata). Como a restrição anterior esta também é utilizada em ações, mas com o objetivo de auxiliar na instanciação dos planos.

A seguir pode ser visto o diagrama/visão *action note view*, que auxilia o desenvolvedor quanto à instanciação do plano do agente, o diagrama mostra quais ações estão envolvidas neste ponto de flexibilização e mostra quais estão implementadas no plano abstrato, quais estão no plano concreto e qual pode estar em ambos os planos, todas seguindo um sentido de execução *top-down*.

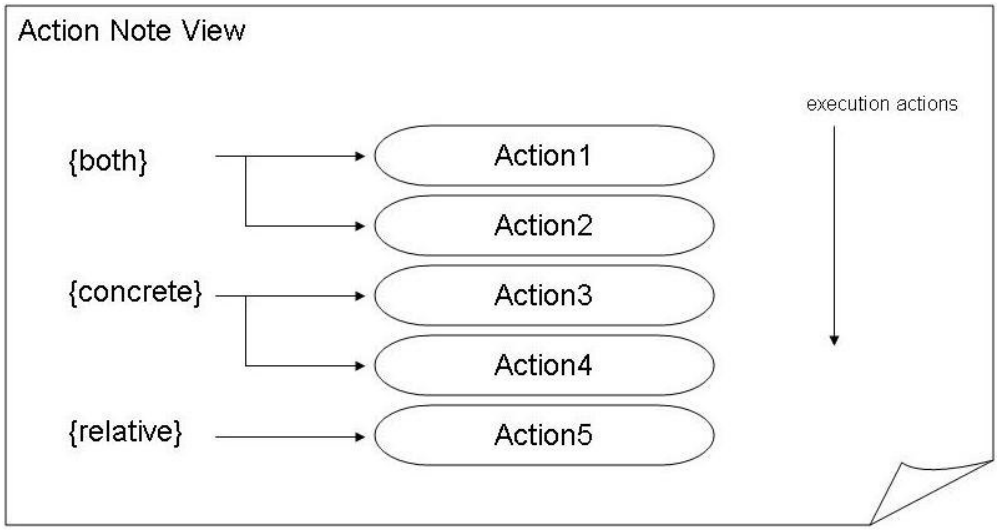

Figura 19. Diagrama *action note view*.

Os dois diagramas podem ser usados em separado, mas se utilizados juntos podem trazer um efeito complementares interessante, facilitando a compreensão da modelagem do sistema. A idéia inicial é que o *action note view* complemente o *flexibility view*, mas não há impedimento quanto o seu uso concomitante. A seguir pode ser vista uma representação dos dois diagramas.

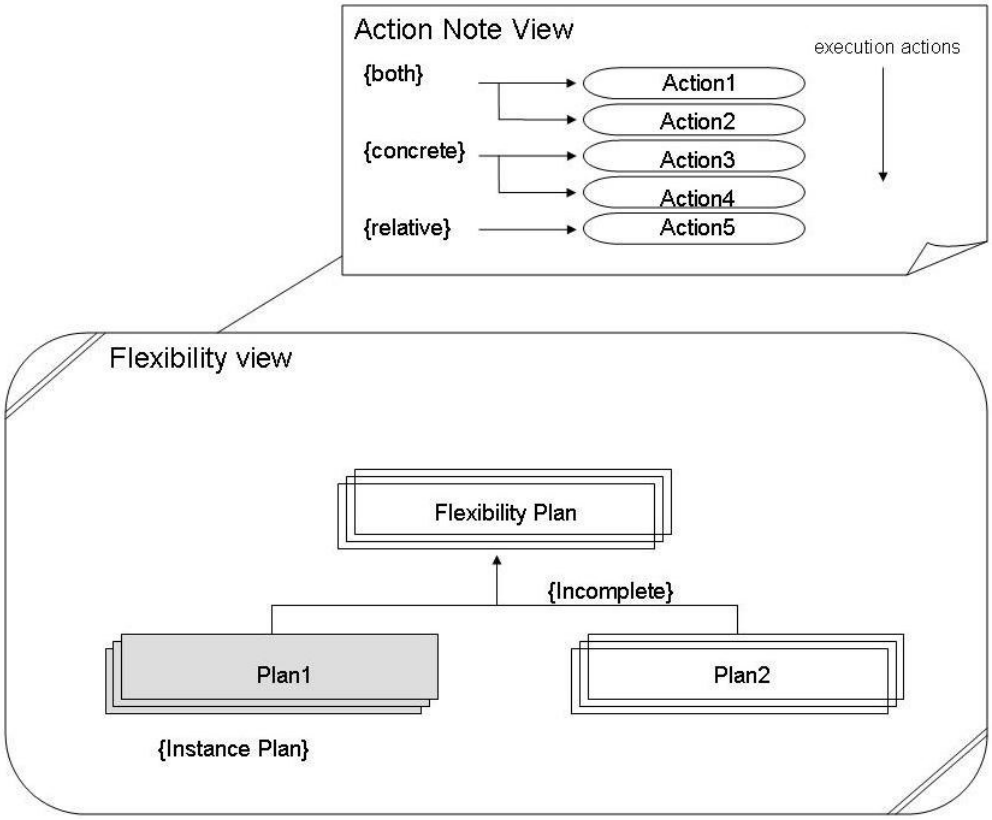

Figura 20. Uso conjunto dos diagramas *flexibility view* e *action note view*.

O diagrama a seguir mostra um ponto de indicação (*Flexibility Plan*) que representa onde deve ocorre à instanciação de um plano, a partir daí os novos diagramas (*Flexibility View* e *Action Note View*) guiarão o usuário determinando qual plano deve ser instanciado e como deverá ser feita tal instanciação.

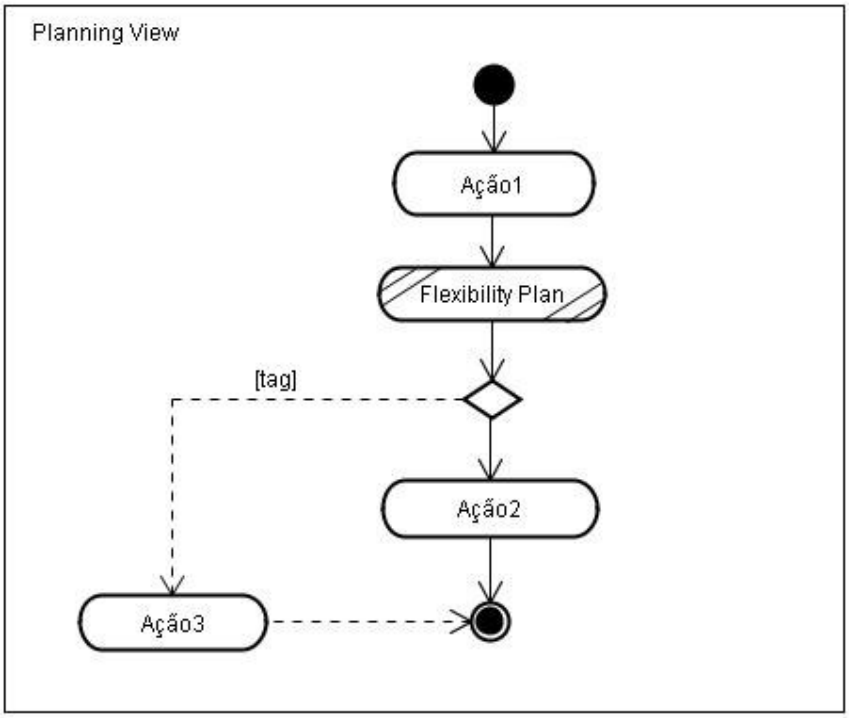

Figura 21. Diagrama *planning view* do ANote estendido para facilitar a integração com o método GP-MVAP.

Dependendo da linguagem de modelagem a integração com o método GP-MVAP pode ser mais trabalhosa, já que em algumas linguagens os diagramas com representações gráficas são atrelados a diagramas com representações descritivas e para determinar a integração será necessário ter uma indicação de onde ocorrerá à instanciação do plano em ambos os diagramas já que um é o espelho do outro, como pode ser visto a seguir no exemplo com o ANote.

Tabela 7. Diagrama *scenario view* do ANote estendido para facilitar a integração com o método GP-MVAP.

Objetivo do Agente	
Lead Agent:	Agente principal.
Precondition:	Precondição que é necessária para que sejam executadas as ações.
Main Action Plan:	Aqui é descria a seqüência de ações como no ANote. 1. Ação1.

	2. Ação2. → **Flexibility (Comando que se quer executar)** 3. Ação3.
Interactions:	Agente que ira interagir com o agente principal.
Variant Plan:	Aqui são descritas as ações secundárias como no ANote. Variant PreCondition: Pré-condição de execução de tais ações. Plan Description: → **Flexibility (Comando que se quer executar)** Se a condição foi satisfeita. 1.1 Ação4.

A seguir pode-se ver uma tabela de definições (informais) sobre o que foi utilizado para explorar a idéia de abstração para o mundo dos planos dos agentes, com isto, será possível facilitar o entendimento das modificações propostas.

Tabela 8. Tabela de definições dos elementos utilizados nos diagramas referentes à flexibilização de planos.

Elemento	Tipo	Semântica	Utilizar em	Com que diagrama
{Incomplete}	Restrição	Nem todos os sub-planos foram ainda definidos.	Plano	*Flexibility View*.
{Instance Plan}	Esteriótipo	Este é plano que deve ser instanciado.	Plano	*Flexibility View*.
{both}	Restrição	A ação deve ocorrer em ambos os planos.	Ação	*Action Note View*.
{concrete}	Restrição	Esta ação deve	Ação	*Action Note*

		estar implementada no plano que será instanciado.		*View*.
{relative}	Restrição	A ação está presente em ambos os planos, porém a maneira como a ação é feita está ligada ao plano que será instanciado.	Ação	*Action Note View*.
→ Flexibility (Comando que se quer executar)	*Hook*	Este ponto é um ponto flexível e tem como parâmetro o comando que se quer executar para obter tal flexibilidade.	Ação	*Scenario View.*

Em se tratando de flexibilização de planos o elemento → *Flexibility* deverá ser representado da seguinte forma → *Flexibility (Plan Creation)* e complementarmente a tabela de elemento pode ser vista a seguir. A nova tabela serve como uma ajuda mostrando o significado da indicação no diagrama *scenario view*.

Tabela 9. Tabela de definições dos elementos utilizados nos novos diagramas referentes à flexibilização de planos.

Elemento	Tipo	Semântica	Ponto de partida	Com que diagrama
→ Flexibility (Plan Creation)	*Hook*	Indica a ação de criação de um plano chamado "pName" no design. Retorna o	Ação	*Scenario View.*

		plano e suas ações para mais adiante ser manipulado.		

Neste caso o plano *pName* é o plano que será instanciado e terá sua representação complementada por outros diagramas da linguagem a ser utilizada.

4.2.1. Flexibilização de Plano no Exemplo

Para exemplificar a abstração de um plano e a idéia de um agente como um *framework* será apresentado como exemplo à aplicação CQPM que verifica e auxilia o médico no desenvolvimento de uma prescrição medicamentosa. Em particular será abordado o caso do cálculo de dosagem de medicamentos, que fica a cargo do "agente dosador".

O objetivo do "agente dosador" é o cálculo de dosagens de medicamentos sejam eles quais forem. Dentro destes grupos de medicamentos vemos dois bem definidos e que podem ser divididos em dois domínios distintos, ou seja, aplicações distintas. Para um ambiente hospitalar pode-se pensar em duas aplicações onde tais aplicações podem ser divididas da seguinte forma, uma aplicação dedicada ao cálculo de dosagem de "medicamentos de CTI" enquanto que a outra aplicação se dedicada ao cálculo de dosagem de medicamentos apelidados de "não-CTI". A seguir o diagrama *planning view* estendido do ANote mostra o ponto de flexibilização do plano.

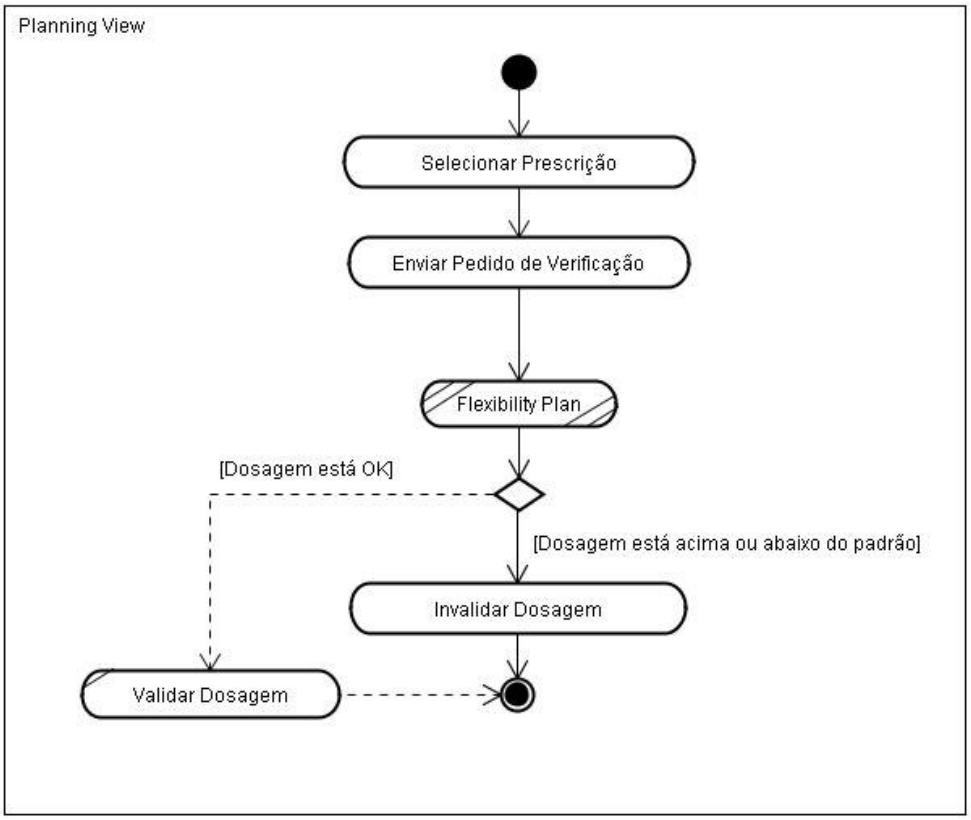

Figura 22. Agente dosador do sistema CQPM modelado com o *planning view* estendido.

O "agente dosador" continua com um único objetivo (o cálculo de dosagem), porém com dois planos distintos, "o cálculo de dosagem para medicamentos de CTI" e "o cálculo de dosagem para medicamentos não-CTI". Cada um dos planos traz ações distintas que devem ser executadas para obter o objetivo do agente e do sistema. Com isso, podem-se observar duas aplicações distintas formando uma linha de produto de software que poderão concretizadas por meio de *framework.*

A seguir pode ser visto o diagrama *scenario view* do ANote, já que os diagramas de *planning view* e de *scenario view* do ANote são complementares onde um representa a parte gráfica do fluxo das ações e o outro representa o fluxo das ações de uma forma descritiva.

Tabela 10. Agente dosador do sistema CQPM modelado com o *scenario view* estendido.

Validar/Invalidar Medicamento por Dosagem

Lead Agent:	Dosador.
Precondition:	Existir um Medicamento e/ou um Princípio Ativo para validar dosagem ou calculá-la.
Description:	Indica a ação de criação de um plano chamado "pName" no design. Retorna o plano e suas ações para mais adiante ser manipulado.
Main Action Plan:	1. Selecionar Prescrição. 2. Calcular a dosagem e ver se está de acordo com a dosagem Padrão. → **Flexibility (Plan Creation)** 3. Validar Dosagem.
Interactions:	Medicamentoso.
Variant Plan:	Variant PreCondition: O Agente encontra erros na dosagem. Plan Description: → **Flexibility (Plan Creation)**. Se a dosagem está com problemas. 1.1 Invalida a Dosagem.

Neste caso o plano a ser criado é o "Medicamento CTI" e terá sua representação a cargo dos diagramas *flexibility view* e *action note view*, como pode ser visto a seguir. Com estes dois diagramas fica claro que o sistema pode ter duas aplicações distintas, a que está sendo instanciada para a aplicação (Medicamento CTI) e a que pode vir a ser instanciada numa futura aplicação (Medicamento não-CTI).

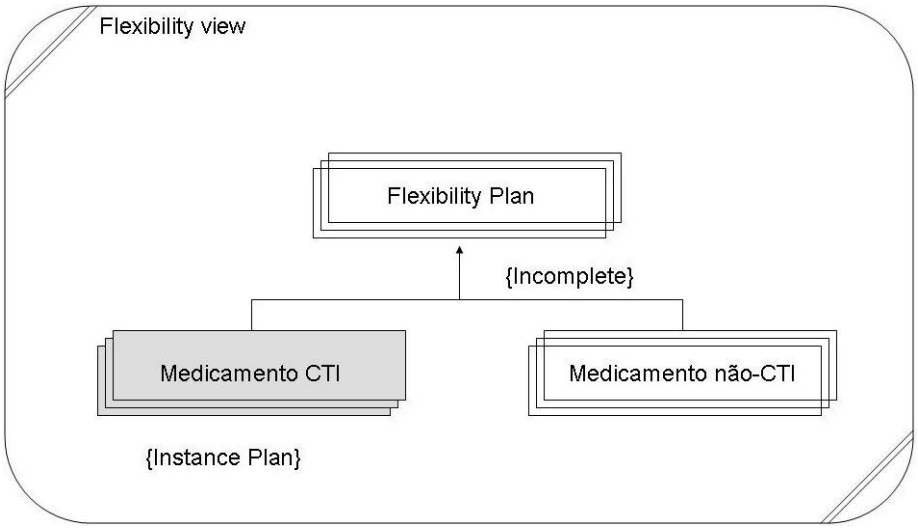

Figura 23. Diagrama *flexibility view* modelando o plano do agente dosador do sistema CQPM.

Na prática a diferença real entre os tipos de cálculo de dosagem define as aplicações. A seguir pode-se ver a diferença prática entre os dois procedimentos.

No plano de cálculo de dosagem para "medicamentos de CTI" devem ser feitos os seguintes procedimentos: definir medicamento, obter o peso do paciente, obter velocidade com que a bomba injeta o medicamento, fazer a diluição do medicamento com soro (já que o medicamento vem em ampolas), definir quanto o paciente deve receber da solução. Após estas ações de cálculo, a dosagem de um "medicamento de CTI" é concluída.

No plano de cálculo de dosagem para "medicamentos não-CTI" devem ser feitos os seguintes procedimentos: definir o medicamento, obter o peso do paciente, obter o valor padrão (em gramas) do medicamento que deve ser utilizado junto com o peso do paciente para o cálculo da dosagem. Após estas ações de cálculo, a dosagem de um "medicamento não-CTI" é concluída.

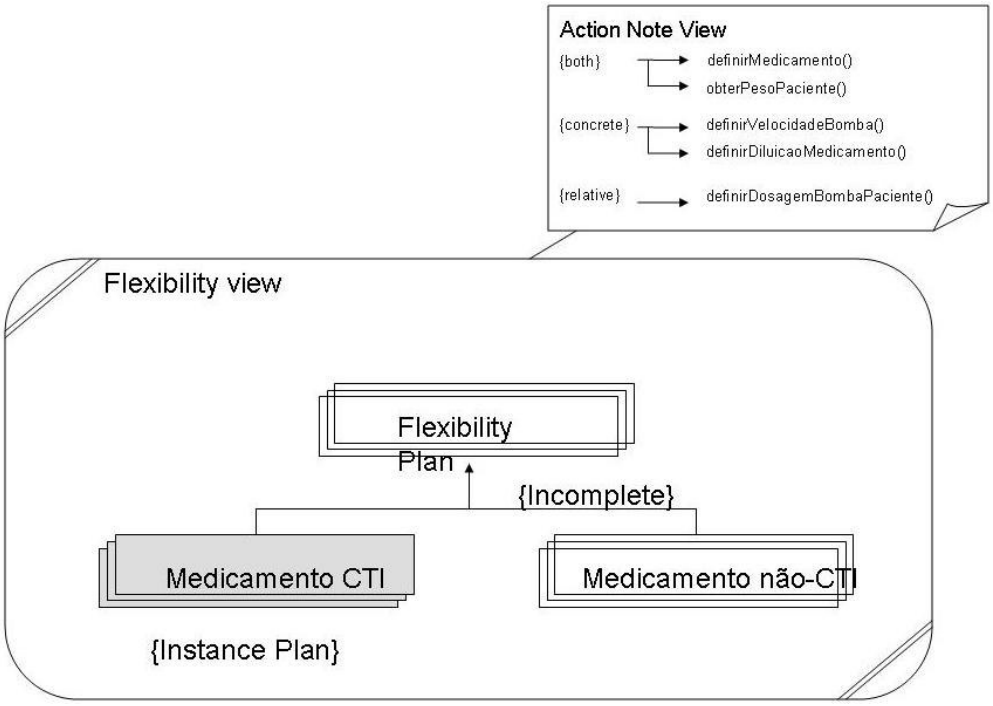

Figura 24. Diagrama *flexibility view* sendo utilizado em conjunto com o *action note view*.

Desta forma, pode-se pensar num plano abstrato, possibilitando a instanciação do plano em função do escopo do problema que ser quer resolver, flexibilizando ainda mais a idéia de agentes. Com isto, será possível a viabilização de uma linha de produtos para o desenvolvimento de agentes de software, o que atualmente não é feito.

Além disso, com a abstração de planos será possível dar uma maior coesão e clareza para a modelagem com relação ao código.

4.3. Flexibilização de Ação

O segundo ponto de abstração em agentes surge no domínio das ações. Muitas vezes as ações dos agentes de software podem ser executadas de maneiras diferentes, o que traz à tona a idéia de abstração no ramo das ações.

A idéia de ações abstratas pode ser observada de forma análoga à idéia de métodos abstratos, porém em orientação a objeto esta já é uma idéia conceituada e empregada há algum tempo.

Como foi visto nas seções anteriores, a linguagem de modelagem de agentes utilizada é o ANote com uma extensão para que o método GP-MVAP seja utilizado em conjunto.

Antes de toda a argumentação sobre ações abstratas é necessário apresentar a definição de uma ação abstrata. Uma ação pode ser definida com algo que não pode ser interrompido e é o último passo para se alcançar um objetivo que compreende a sua assinatura e implementação. Uma ação abstrata por sua vez ocorre quando não é possível prever de que maneira a ação deve ser executada, ou seja, tal ação terá apenas a sua assinatura e sua implementação ficará a cargo da aplicação (final) que será instanciada.

Na seção 4.3 foi visto que a restrição *{relative}* também serve para definir se uma ação é abstrata, no entanto uma ação com a marca *{relative}* quer dizer que a ação abstrata tem sua assinatura no plano abstrato e que sua implementação está dentro dos planos que podem ser instanciados (mesmo o que não será instanciado na ocasião).

Aqui será apresentada a restrição *{abstract}* que, ao ser identificada, indica que tal ação é abstrata, mas com uma semântica diferente da restrição *{relative}*. A restrição *{abstract}* diz apenas que a ação indicada é abstrata e que ela terá pelo menos uma instanciação.

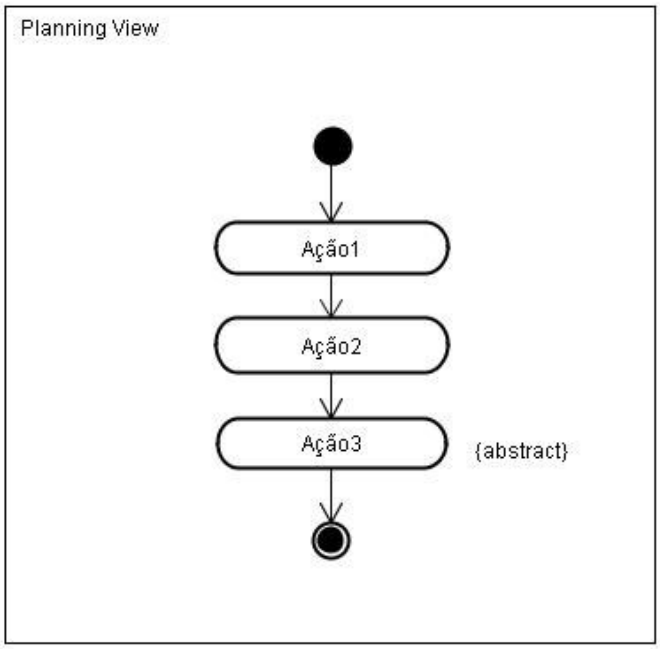

Figura 25. Diagrama *planning view* estendido para flexibilizar ações.

O diagrama *planning view* foi utilizado apenas por questão de comodidade, já que, será utilizada a linguagem ANote. Para esta representação o GP-MVAP provê o diagrama *action note view* observado nas seções anteriores.

Com isto é possível obter uma linha de produto para os agentes que tenham a característica de variabilidade em suas ações. A seguir o diagrama *scenario view* que completa a modelagem com ANote.

Tabela 11. Diagrama *scenario view* estendido para flexibilizar ações.

Objetivo do Agente	
Lead Agent:	Agente principal.
Precondition:	Pré-condição que é necessária para que sejam executadas as ações.
Main Action Plan:	Aqui é descrita a seqüência de ações como no ANote. 1. Ação1. 2. Ação2. → **Flexibility (Comando que se quer executar)** 3. Ação3.
Interactions:	Agente que irá interagir com o agente principal.
Variant Plan:	Aqui são descritas as ações secundárias como no ANote. Variant PreCondition: Pré-condição de execução de tais ações. Plan Description: → **Flexibility (Comando que se quer executar)** Se a condição foi satisfeita. 1.1 Ação4.

A seguir as tabelas de definições deixam mais claro para o leitor a semântica dos elementos utilizados na definição de ações abstratas.

Tabela 12. Tabela de definições dos elementos utilizados nos diagramas a fim de flexibilizar uma ação.

Elemento	Tipo	Semântica	Utilizar em	Com que diagrama
{abstract}	Restrição	A maneira com que a ação será implementada depende da aplicação final.	Ação	*Action Note View* OU *Planning View*.
→ Flexibility (Comando que se quer executar)	*Hook*	Este ponto é um ponto flexível e traz consigo o comando que se quer executar para obter tal flexibilidade.	Ação	*Scenario View.*

Em se tratando de flexibilização de ações, o elemento → Flexibility deverá ser representado da seguinte forma → Flexibility (Action Creation) e complementarmente a tabela de elemento pode ser vista a seguir a nova tabela que serve como uma ajuda mostrando o significado da indicação no diagrama *scenario view*.

Tabela 13. Tabela de definições dos elementos utilizados nos diagramas a fim de flexibilizar uma ação.

Elemento	Tipo	Semântica	Ponto de partida	Com que diagrama
→ Flexibility (Action Creation)	*Hook*	Indica a instanciação de uma ação chamada "aName" no design do agente. Retorna a ação para que mais adiante ela possa ser	Ação	*Scenario View.*

		manipulada.		

Uma representação através de exemplos será feita na seção seguinte, onde é descrito o cenário de compra e venda de produtos e através deste exemplo ficará mais claro como e qual ação deve ser instanciada. O anexo C traz um exemplo de flexibilização de ação.

4.4. Instanciando uma Aplicação

Como foi visto nas seções anteriores os planos e as ações recebem um ponto de indicação mostrando onde pode ser flexível, para ajudar no processo de instanciação fica intuitivo o uso de técnicas como *cookbook*, *hooks* ou RDL.

Dentro do conjunto de técnicas disponibilizadas, a RDL é a que apresenta mais mecanismos úteis para tal abordagem. A RDL foi escolhida por apresentar toda uma estrutura com campos significativos para o entendimento e que seraao herdados no método GP-MVAP, dessa forma será necessária apenas uma de tais estruturas. A RDL disponibiliza também um caminho mais preciso para instanciação, desde que não seja utilizada a linguagem natural. Para isto ela conta com uma estrutura bastante completa, com descrição, design e script para a instanciação (de *frameworks* orientados a objeto) (Oliveira et al., 2006). Como foi discutido no capítulo 3, o método GP-MVAP traz uma extensão da RDL para orientação a agentes.

Tabela 14. Diagrama *instantiation view* baseado em RDL e utilizado para guiar o processo de instanciação de planos e ações.

Instantiation View	
Description:	Indica a ação de criação de um plano chamado "pName" no design. Retorna o plano e suas ações para mais adiante ser manipulado.
Code:	**COOKBOOK** o que se quer fazer (objetivo) **RECIPE** main //adding a new plan

	NEW_PLAN (pName); // … **END_RECIPE**; **END_COOKBOOK**

Nas seções anteriores foi visto que o diagrama de *scenario view* é estendido para utilizar a solução proposta juntamente com a linguagem ANote. Tal diagrama contém uma indicação para o ponto de flexibilização do agente, onde tal ponto funciona como gancho para posteriormente ser apresentado o modo como deverá ser instanciado o novo plano como foi visto na figura anterior.

O Diagrama *instantiation view* foi feito com base na RDL, porém visando aplicações multi-agentes.

A RDL padrão utiliza as estruturas de *Command*, *Syntax*, *Description*, *Comment* (opcional), *Code* e *Design* para representar as tarefas de instanciação (Oliveira et al., 2006). Para documentar a instanciação de planos e ações serão utilizadas (e estendidas) algumas estruturas da RDL como *Description* e *Code.* A estrutura *Description* diz como funciona a instanciação e a estrutura *Code* tem a finalidade de indicar o ponto que faz com que o agente necessite da instanciação de um plano.

A estrutura *Code* é o elemento central do diagrama *instantiation view* na documentação da instanciação de um plano ou uma ação. Mais adiante será apresentada a semântica que está atrelada ao novo elemento.

Todo agente tem um objetivo a ser alcançado, o qual é obtido através da execução de ações que são organizadas por planos, porém algumas vezes a execução de certas ações faz necessária instanciação de um plano para o agente em função da aplicação que se deseja desenvolver.

Como visto nas seções anteriores, a linguagem ANote será utilizada em conjunto com o método GP-MVAP e será necessário indicar o ponto de flexibilização do agente; tal indicação mostra onde deve ocorrer a instanciação de um plano e é representada pela seguinte estrutura "→ Flexibility (Plan Creation)". A estrutura indica que há possibilidade de instanciação de planos distintos e que a instanciação de tal plano irá variar em função da aplicação que se deseja utilizar.

Do mesmo modo que a abstração de um plano a abstração de uma ação requer uma forma de indicação (na linguagem para que seja utilizada

concomitantemente com o método), a qual é feita através da seguinte estrutura "→ Flexibility (Action Creation)". A estrutura indica que tal ação tem maneiras distintas de ser executada e a instanciação de tal ação irá variar em função da aplicação que se deseja utilizar e das várias maneiras que ela pode executar tal ação.

Estas duas estruturas "→ Flexibility (Action Creation)" e "→ Flexibility (Plan Creation)" também são utilizadas como gancho para chegar ao diagrama *instantiation view* que serve de guia para instanciação de aplicações futuras.

A estrutura *Code* é fundamental, pois é onde será possível indicar qual o objetivo do agente através da estrutura "**COOKBOOK** objetivo" e para a realização deste objetivo faz-se necessário seguir a receita principal descrita em "**RECIPE** main" que traz em sua descrição de como deve ser feita a instanciação do plano através da adaptação da sintaxe da RDL "**NEW_PLAN** (pName);". A instanciação de uma ação requer algo semelhante utilizando para isso a sintaxe "**NEW_ACTION** (aName);".

Como já foi dito a estrutura *Code* traz consigo a estrutura *Description* que serve como uma ajuda ao desenvolvedor que utilizará o diagrama *instantiation view*.

A estrutura *Description* provê uma descrição textual para a tarefa, dizendo qual será o plano instanciado o que deverá ser esperado como retorno.

Através do diagrama *instantiation view* é possível indicar e descrever como deve ser feita a instanciação de um plano um agente. Na seção 4.5.1 é descrito o processo de instanciação de planos.

Isto também será feito para as ações abstratas através do mesmo elemento (*Code*) que é o guia da instanciação propriamente dito. Com o elemento *Code* será possível representar de que maneira a ação deve ser desenvolvida. Na seção 4.5.2 é descrito o processo de instanciação de ações.

Espera-se que a abordagem torne o entendimento do sistema mais fácil e rápido para o desenvolvedor.

A seguir pode-se ver uma tabela de definições (informais) sobre o que foi utilizado para explorar a idéia de ter uma documentação guia para as instanciações de planos e ações, com isto, será possível facilitar o entendimento das modificações propostas.

Tabela 15. Tabela de definição dos principais elementos utilizados no *instantiation view*.

Elemento	Tipo	Semântica	Utilizar em	Com que diagrama
Code:	Exemplo de instanciação	Traz o que deve ser instanciado, através de um pseudocódigo que segue o padrão baseado na RDL.	Plano e ações	*Instantiation View*.
Description:	Intenção	De forma textual define a intenção da tarefa, o que deve ser utilizado para a realização da tarefa e o que deve ser esperado após a realização da tarefa.	Plano e ações	*Instantiation View*.

4.4.1. Instanciando o Exemplo

Após o desenvolvimento dos diagramas *planning view* e *scenario view* providos pela linguagem ANote e dos diagramas *flexibility view* e *action note view* providos pelo método GP-MVAP o responsável pela modelagem terá em mãos o fluxo de execução das ações e o ponto onde é necessário instanciar o novo plano ou a nova ação, agora falta utilizar o diagrama *instantiation view* para determinar o guia para instanciação através das estruturas baseadas em RDL.

A seguir pode-se ver como é feito tal procedimento seguindo o exemplo visto anteriormente na seção 4.3.1 referente à aplicação CQPM. Foi visto neste exemplo que o "agente dosador" pode ser utilizado em dois ambientes distintos e por isso é necessária a idéia de abstração de plano.

O processo de instanciação a seguir mostra que o agente tem o mesmo objetivo independente do plano e tem funções distintas em seu interior dependendo do plano.

Tabela 16. Diagrama *instantiation view* guiando a instanciação de um plano.

Instantiation View	
Description:	Indica a ação de criação de um plano, que é o plano concreto "medicamentoCTI" que foi visto no design *flexibility view*. Retorna o plano e suas ações para mais adiante ser manipulado. SELECT_PLAN_EXTENSION(Flexibility Plan) retorna o plano concreto para o sistema CQPM. ADD_ASSIGN(SELECT_PLAN_EXTENSION (Flexibility Plan), *Action Note View*); mostra que o plano concreto receberá a assinatura das ações contidas em *Action Note View*.
Code:	**COOKBOOK** Calcular dosagem **RECIPE** main //adding a new plan (medicamentoCTI) **LOOP** PLAN_EXTENSION(Flexibility Plan, ?) **END_LOOP** ADD_ASSIGN(SELECT_PLAN_EXTENSION (Flexibility Plan), *Action Note View*); // … **END_RECIPE**; **END_COOKBOOK**

4.5. Discussão

O método GP-MVAP que foi apresentado é composto de diagramas e documentação de modo a facilitar a representação de novas instâncias de

aplicação no caso de sistemas multi-agentes que contenham agente com característica de variabilidade.

A utilização de diagramas e *tags* permitem ao GP-MVAP representar de forma clara os planos abstratos e suas instâncias. As *tags* também permitem definir claramente onde cada ação está sendo executada e onde deverá ser implementada.

A documentação que segue com o método GP-MVAP traz uma forma de guiar o desenvolvedor e com isto minimizar as dúvidas quanto a melhor forma de implementação do sistema e sua arquitetura. A documentação utiliza uma técnica baseada em RDL que faz uma espécie de ponte entre a modelagem e a implementação facilitando a interpretação do desenvolvedor.

A solução proposta visa solucionar o problema de representação da variabilidade de agentes de software através de um método que se basea em idéias consagradas de orientação a objeto como *framework* e linha de produto. Desta forma, é possível diminuir o custo do desenvolvimento de SMA, ao mesmo tempo aumentar a qualidade no desenvolvimento do software e minimizar a possibilidade de erros.

Nas seções anteriores foi apresentado um sistema que serviu para demonstrar como a solução modela uma aplicação multi-agentes. Um deles foi o sistema CQPM, inicialmente projetado com uma linguagem de modelagem de agentes sem fazer uso do método GP-MVAP. Em seguida, o sistema CQPM foi remodelado utilizando o método proposto, o que serviu como base para avaliar as vantagens que a solução proporcionou ao projeto.

No sistema CQPM é fácil observar que a idéia de *framework* se encaixa muito bem e traz grandes vantagens para a modelagem e implementação. A flexibilidade documentada tornando possível a reutilização e a manutenção, além da possibilidade de reutilização do código são pontos que podem ser destacados dentre as vantagens.

A abordagem proposta visa inserir idéia de linha de produto implementada por meio de *framework* a fim de atingir o objetivo de redução dos custos mantendo a alta qualidade, simplificando o desenvolvimento de aplicações e com isso obter ganhos de produtividade.

O método proposto apresenta todas as possíveis instâncias de aplicações segundo o requisito do sistema e com isso o tempo de desenvolvimento pode ser otimizado.

A solução define o método GP-MVAP que tornará possível a representação de planos e ações abstratas e conseqüentemente torna possível a representação de uma linha de produto para sistema multi-agente e que poderão ser observados como uma idéia de *framework*.

Com a linha de produto será possível determinar uma família de produtos em função de um sistema multi-agente onde tais produtos poderão ser acoplados quando necessário. Desta maneira com linhas de produto é possível projetar a variabilidade em um agente de software e sua flexibilização.

Um outro atrativo da solução proposta é que esta pode ser aplicada em qualquer linguagem de modelagem de agentes, já que o método GP-MVAP é livre de qualquer meta-modelo.

O mundo da tecnologia tendendo cada vez mais para aplicações distribuídas faz com que o paradigma de agentes torne-se cada vez mais um sucessor direto do paradigma de orientação a objeto (Jennings, 2001). Porém a teoria de agentes de software traz uma abordagem extremamente nova e por isso em constante evolução, tal evolução surge tanto do ponto de vista de desenvolvimento como do ponto de vista de modelagem.

As linguagens de modelagens de agentes atuais visam representar sistemas baseados em agentes de software onde tais agentes são desmembrados em objetivos, planos e ações. Porém as linguagens atuais só possibilitam a representação de sistemas baseados em agentes de forma inteiriça não permitindo uma flexibilização baseada na idéia de *framework*.

Um dos grandes obstáculos para a adoção da idéia no mundo OO é a curva de aprendizado (Fayad, 2000) e provavelmente este também poderá ser um obstáculo no âmbito dos agentes de software.