UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Uma Proposta de Apoio Para Decisões de Grupo no Ambiente PROSOFT

por

RONNIE CLEY DE OLIVEIRA ALVES

Dissertação submetida à avaliação,
como requisito parcial para a obtenção do grau de Mestre
em Ciência da Computação

Prof. Dr. Daltro José Nunes
Orientador

Porto Alegre, junho de 2002.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Oliveira Alves, Ronnie Cley de

Uma Proposta de Apoio para Decisões de Grupo no Ambiente PROSOFT / por Ronnie Cley de Oliveira Alves. – Porto Alegre : PPGC da UFRGS, 2002.
174 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR – RS, 2002. Orientador: Nunes, Daltro José.

1. Sistemas de Apoio à Tomada de Decisão 2. Modelo de Processo de Software 3. Registro de Justificativas de Projeto. I. Nunes, Daltro José. II. Título

Aos
meus pais

Agradecimentos

Agradeço a Deus pela vida, pelas oportunidades, pela proteção e pelas luzes nos momentos mais difíceis.

Aos meus pais e familiares, agradeço por todo o estímulo e esforço em proporcionar este estudo com toda a tranqüilidade possível.

Agradeço a Ana, pela compreensão, amor, amizade e incentivo em todos os momentos alegres e difíceis da vida e deste trabalho.

Agradeço a Nossa Senhora de Nazaré e ao Santo Expedito pelas graças alcançadas.

Agradeço ao meu orientador, Prof. Dr. Daltro José Nunes, pela disposição para ajudar, também pela compreensão e confiança no meu trabalho.

Agradeço aos amigos e tutores, Rodrigo e Carla, pela amizade, conversas divertidas e por toda ajuda que me deram neste trabalho.

Agradeço a todos os colegas do PPGC, em especial à Bruno Jatene e José Otávio, pela amizade e companhia durante esses anos.

Agradeço aos amigos Isabel e Abraham, pelo companheirismo e incentivo durante o desenvolvimento deste trabalho.

Agradeço aos professores e funcionários do PPGC, que de uma forma ou de outra ajudaram na realização deste trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ADS	Ambiente de Desenvolvimento de Software
APSE	ADA Programming Support Environment – Ambiente de suporte à programação ADA
ATO	Ambiente de tratamento de objetos
GP	Gerenciador de Processos
IBIS	Issue Based Information System – Sistema de informação baseado em questões.
ICS	Interface de comunicação do sistema
SADT	Structured Analisys and Design Technique – Técnica de Análise e Projeto Estruturados
SADG	Sistemas de apoio à Decisão em Grupos
SAD	Sistemas de apoio à Decisão

Resumo

O processo de desenvolvimento de software implica na necessidade constante de tomadas de decisão. A cada etapa do processo, torna-se necessário estabelecer a comunicação e interação entre usuários, gerentes, analistas, programadores e mantenedores numa constante troca de informações.

O registro dos artefatos produzidos durante todo o processo é uma questão que norteia as pesquisas em ambiente de desenvolvimento de software. Quando se fala em suporte ao processo de colaboração entre os elementos de uma equipe de desenvolvimento, este registro torna-se ainda mais necessário. Neste contexto, a modelagem dos dados a serem armazenados se amplia para comportar outras informações provenientes da interação do grupo além dos artefatos gerados.

As informações trocadas durante este processo interativo que incluem fatos, hipóteses, restrições, decisões e suas razões, o significado de conceitos e, os documentos formais formam o que é denominado pela literatura especializada como memória de grupo.

A proposta da arquitetura SaDg PROSOFT visa fornecer suporte a memória de grupo, no que diz respeito ao registro das justificativas de projeto(*Design Rationale*), através de uma integração com o gerenciador de processos (GP) provido pelo ADS PROSOFT. Esta integração se dá através das ferramentas inseridas no modelo, assim desenhadas: Editor de Norma, Editor de Argumentação, Extrator de Alternativas, Editor de Votação.

O ADS PROSOFT integra ferramentas para desenvolvimento de software. Este ADS foi escolhido para o desenvolvimento do modelo SADG, pois baseia-se na construção formal de software, mas particularmente no método algébrico, por ser um ambiente estendível, possibilitando a inclusão do modelo SaDg PROSOFT ao seu conjunto de ferramentas, por ter características de um ambiente distribuído e cooperativo e por não dispor de nenhum suporte à discussões e decisões em grupos.

São apresentados os fundamentos de modelos SADG e algumas ferramentas. Alguns dos principais requisitos desses ambientes foram coletados e são apresentados a fim de embasar a proposta do trabalho. O modelo SADG é apresentado na forma de ferramentas PROSOFT(chamadas ATOs) e permite a definição de atividades como: Atividade de argumentação, atividade de extração e a atividade de votação. Além disso, permite a coordenação destas atividades através de um facilitador e do próprio GP, e também, possui um mecanismo para a configuração do processo decisório.

Palavras-Chaves: Ambiente de Desenvolvimento de Software, Sistemas de apoio à tomada de decisão, modelo de processo de software, modelos de argumentação, registro de justificativas de projeto, especificação formal.

TITLE: “A DECISION GROUP SUPPORT FOR THE PROSOFT ENVIRONMENT”.

Abstract

The software development process implies a sequence of decisions. Members of a software development team need to communicate, to interact, to negotiate with themselves about issues which came up during this process.

Storing all the information generated during software development is an important research issue in software environments. When one thinks about giving support in a collaborative process for all members involved in this task, storing software products are even more essential. In this context, not only software products are important but all documents generated from this interaction must be modeled to represent this interactive process completely.

During software development, information like facts, hypothesis, restrictions, decisions and their motivations, the concepts and the formal documents creates a huge rationale memory of group’s decisions.

This proposal implements a model named SaDg PROSOFT to provide rationale memory of group’s decisions. It is based on Design Rationale guided through the integration with the process centered software development environment, also called GP, supported by the PROSOFT SDE(software development environment). This integration is supported by the tools of the model, like: Norma Editor, Argumentation Editor, Alternatives Extractor and Voting Editor.

The PROSOFT SDE integrates software development tools, and was chosen for supporting the model because it is based on formal software construction, especifically in algebraic specification; it is extendable, allowing the decision model to share its set of tools; it is distributed and cooperative, and it currently does not support group decision during the software development process.

Models that support rationale memory of group’s decisions are presented, as some examples of tools. Some of the most important requirements were gathered and presented to provide the basis of this GDSS (Group Decision Support Systems) proposal. The GDSS model created is showed as some PROSOFT tools (named ATOs) and it supports decision tasks like: argumentation, extraction and voting. Besides that, decision tasks are guided through a coordenation mechanism using a facilitator role and the GP.

Keywords: Software Development Environment, Group Decision Support Systems, Software Process Model, Argumentation Models, design rationale, formal specification.

1 Introdução

O processo de desenvolvimento de software implica na necessidade constante de tomadas de decisão. A cada etapa do processo, torna-se necessário estabelecer a comunicação e interação entre usuários, gerentes, analistas, programadores e mantenedores numa constante troca de informações.

Na maioria dos processos de decisão, o tempo é um fator que se objetiva diminuir. As reuniões devem ser claras e objetivas, focalizando sempre o problema em questão, de maneira a administrar eficientemente as discussões entre os participantes para a chegada a uma conclusão final. Apesar da rapidez exigida, deve-se procurar levantar, considerar, avaliar e discutir o maio número de alternativas possíveis de solução a fim de que a decisão tomada seja a melhor. Assim, nenhuma alternativa pode ser negligenciada, sob o risco de se estar talvez negligenciando a melhor alternativa.

Pesquisadores da área de Engenharia de Software tem-se preocupado em criar ferramentas que auxiliem e facilitem a solução de problemas não estruturados por grupo de pessoas em ambientes de desenvolvimento de software. Dentre as ferramentas possíveis de serem utilizadas nestes ambientes, as ferramentas de apoio à decisão em grupo possuem um papel fundamental, já que as decisões tomadas no decorrer do processo de software refletem sobremaneira o produto final de desenvolvimento.

Estas ferramentas procuram minimizar os problemas de análise, acesso e compartilhamento de informações entre os componentes do grupo de decisão. Objetivam também reduzir a desorganização das atividades, a dominância do processo por determinadas elementos do grupo, pressões sociais, inibição e outras dificuldades facilmente encontradas nos trabalhos em grupo.

Para realizar cada uma das etapas de produção de software, os membros do grupo do desenvolvimento realizam tarefas cooperativamente. Estas tarefas envolvem: a gerência e o planejamento do processo, a elaboração e armazenamento de documentos, a discussão entre os participantes, a tomada de decisões e o gerenciamento do histórico do desenvolvimento.

Para realização destas atividades de forma automatizada e cooperativa, um ambiente de desenvolvimento deve prover mecanismos para suporte aos serviços básicos de: comunicação entre os participantes, coordenação de atividades e compartilhamento de informações.

Como exemplo, a gerência do processo de desenvolvimento envolve, além da coordenação das atividades da equipe, a análise dos custos e recursos. Para esta etapa, portanto, seria útil que o ambiente de desenvolvimento fornecesse ferramentas para o apoio à coordenação de tarefas e sistemas de suporte à decisão em grupo para auxiliar os coordenadores a decidir quanto aos recursos a serem utilizados no projeto.

Nas etapas de elicitação, especificação, análise de requisitos e projeto do sistema, ferramentas de apoio à discussão e decisão também se fazem importantes[BOR 95].

Não é difícil notar a importância que exercem as tarefas de discussão e tomada de decisão em grupo durante o desenvolvimento de software, uma vez que esta atividade aparece em quase todas as fases do processo.

1.1 Motivação e Objetivos do trabalho

O processo de decisão, por si só, é uma atividade bastante complexa. Levantar as alternativas possíveis de solução, analisá-las detalhadamente e por fim escolher a mais adequada é uma tarefa que exige, na maioria das vezes, o uso de métodos para este fim. Também é de grande importância registrar detalhadamente os passos, as discussões e as justificativas que levaram à chegada a tal conclusão. Particularmente no desenvolvimento de software, este registro se faz ainda mais necessário [POT 88][LUC 91].

O ambiente de desenvolvimento de software PROSOFT [NUN 92][NUN95], estabelecido no PPGC-UFRGS, foi construído com o objetivo de suportar o desenvolvimento de software através da utilização de técnicas de especificação formal. As ferramentas do ambiente são construídas com o uso de especificações algébricas [WAT 91] e o ambiente proporciona integração de dados, controle e apresentação entre as ferramentas. A necessidade de suportar atividades de tomada de decisão no processo de desenvolvimento de software para o PROSOFT motivou a proposta do presente trabalho[ALV 00].

O objetivo deste texto é apresentar um estudo a cerca de Sistemas para Apoio a Decisão em Grupos (SADG), e sua implicação no contexto de desenvolvimento de software. Este estudo resultou na especificação formal de um modelo SADG para o ambiente PROSOFT, definido na forma de conjunto ferramentas do ambiente chamada de SaDg PROSOFT.

O ADS PROSOFT foi escolhido para este estudo, pois baseia-se na construção formal de software, mais particularmente no método algébrico, por ser um ambiente estendível, possibilitando a inclusão de novos componentes, e por estar evoluindo para permitir a definição e execução de modelos de processo de software. As ferramentas especificadas neste trabalho estabelecem, portanto, a estrutura básica de *Design Rationale* necessária para registrar os processos de tomada de decisão durante os processos de software especificado em [LIM 98].

Por fim, deve-se ressaltar que uma nova categoria de aplicações voltadas ao apoio a tomada de decisão em grupo durante o desenvolvimento de software, podem ser desenvolvidas a partir deste trabalho.

1.2 Organização do texto

O texto está organizado como segue. O capítulo 2 descreve os Sistemas de Apoio a Decisão a Grupos, suas taxonomias e características para implementação de modelos.

O capítulo 3 apresenta algumas ferramentas e modelos SADGs, e suas principais funcionalidades. Apresenta também, alguns conceitos importantes sobre processos de software, bem como a utilização de mecanismos que apoiem tomada de decisão durante todo o processo.

O capítulo 4 descreve o ADS PROSOFT, as características que norteiam o desenvolvimento de software para o ambiente e suas limitações atuais.

O capítulo 5 apresenta uma descrição informal da motivação e características básicas acerca da extensão para o apoio a tomada de decisão em grupos proposta ao PROSOFT.

O capítulo 6 contém parte da especificação formal do SaDg PROSOFT, utilizando o paradigma do ambiente. De forma a facilitar o entendimento, todas as operações das ferramentas estarão detalhadas no anexo 1.

O capítulo 7 discute possíveis utilizações do modelo de SADG proposto neste trabalho em outros domínios.

No capítulo 8 são apresentadas conclusões e atividades futuras que podem ser realizadas a partir deste esforço.

O anexo 1 apresenta todas as operações definidas para as ferramentas do SaDg PROSOFT.

2 Sistemas de Apoio à Decisão a Grupos

O processo de tomada de decisão em grupo tem recebido muita atenção da comunidade de pesquisa visando a elaboração de métodos e disciplinas para sua realização. Toda esta preocupação é devida ao fato de que as decisões, atualmente, devem ser tomadas com bastante rapidez, segurança e correção.

Na maioria dos processos de decisão, o tempo é um fator que se procura minimizar. As reuniões devem ser objetivas, focalizando sempre o problema em questão, onde procura-se administrar eficientemente as discussões entre os participantes para a chegada a conclusão final. Apesar da rapidez exigida, deve-se procurar levantar, considerar, avaliar e discutir o maior número de alternativas possíveis de solução a fim de que a decisão tomada seja a mais correta. Assim, nenhuma alternativa pode ser negligenciada, sob o risco de se estar possivelmente negligenciando a melhor solução. Mais ainda, todas estas alternativas precisam ser analisadas cuidadosamente e com igual detalhe para que a conclusão final espelhe realmente a melhor solução.

Outra necessidade do processo de decisão é que as discussões desencadeadas sejam devidamente documentadas. A argumentação, as opiniões de cada participante, a forma de avaliação das alternativas e é claro, a decisão final, devem ser registradas a fim de permitir que futuramente seja possível rever os motivos que levaram à escolha e à não escolha de alternativas e utilizar os resultados em outros problemas de decisão similares.

Muito se tem pesquisado sobre a possibilidade de automação da atividade de decisão visando ampliar a capacidade de comunicação, argumentação e consenso entre pessoas envolvidas na tomada de decisão através da utilização de sistemas de suporte à decisão em grupo (SSDG). Estes sistemas, conforme apresentado por Kraemer et al [KRA 88], são sistemas interativos que facilitam a solução de problemas não estruturados por um conjunto de tomadores de decisão trabalhando em grupo.

A principal motivação para a construção destes sistemas reside na intenção de aumentar a produtividade das reuniões de decisão atualmente realizadas nas organizações em geral. Aumentar a produtividade das reuniões significa, neste caso, aumentar a velocidade de chegada a um consenso e aumentar a qualidade da decisão final.

Os sistemas de suporte à decisão em grupo procuram reduzir os problemas de análise, acesso e compartilhamento de informações entre os componentes do grupo de decisão. Objetivam ainda, reduzir a desorganização das atividades, a dominância do processo por determinados elementos do grupo, pressões sociais, inibição e outras dificuldades encontradas nos trabalhos em grupo.

Toda estas características atribuídas a sistemas de suporte à decisão e outras que serão apresentadas no decorrer deste trabalho, estimularam a presente proposta para o desenvolvimento de um modelo SADG para o PROSOFT, que é um ambiente formal orientado à processos de software, cujo foco é auxiliar o processo de desenvolvimento de software.

O processo de desenvolvimento de software implica na necessidade constante de

tomada de decisão. Os resultados destas decisões refletem o produto final do desenvolvimento principalmente no que tange à sua manutenção.

2.1 Taxonomias de SADGs

A dificuldade em definir SADG levou os pesquisadores a apresentar diferentes taxonomias que levam em conta uma série de aspectos que até então não foram reunidos em uma única definição. No estudo de [STO 92] ele verifica uma série de semelhanças e interseções entre SADG e outras áreas tais como: CSCW (*Computer Supported Cooperative Work*), *Groupware*, Automação de Escritórios, Conferência por Computador, dentre outros. Neste estudo pôde-se observar que algumas áreas poderiam ser sub-áreas de outras ou até sinônimas.

A fim de melhor entender o termo SADG, diversas taxonomias [STO 92][KRA 88][BUI 86][DES 87] são encontradas na literatura visando contribuir numa melhor compreensão deste tipo de sistema, de seu papel na tomada de decisões, e das funcionalidades necessárias para apoiar ao grupo e que dirigem muitas das pesquisas nesta área. A Tab. 2.1 abaixo resume alguma destas taxonomias:

TABELA 2.1 – Taxonomias de SADG.

Taxonomia	**Características**	**Referência**
Pinsoneault e Kraemer	Propõe 20 variáveis sociais e técnicas para melhor compreender SADGs	[STO 92]
Kraemer e King	Classifica bases tecnológicas levando em conta: *hardware*, *software*, *organizationware* e as pessoas	[KRA 88]
Bui e Jarke	Distingue seis tipos de arquiteturas de SADG	[BUI 86]
DeSanctis e Gallupe	Classifica as abordagens possíveis de suporte a grupos em três níveis de sistemas.	[DES 87]

O SADG proposto possui várias características mencionadas por [KRA 88] e [DES 87], no restante desta seção são detalhadas as taxonomias.

Em [KRA 88] é proposta uma classificação fundamentada no que denomina bases tecnológicas de SADG, em relação a quatro parâmetros: *hardware*, *software*, *organizationware* e pessoas.

No parâmetro *hardware* existe a preocupação de avaliar quais facilidades deverão ser disponibilizadas com relação a equipamentos de computação,

telecomunicações e audiovisual, bem como diferenças quanto configuração e sofisticações dos mesmos.

Nesta taxonomia o software de um SADG deve compreender características tecnológicas a fim de que ele possa ser usado para suportar processamento de informações, modelagem de decisão e/ou comunicação. O software para suportar processamento de informações engloba necessariamente um SGBD e ferramentas gráficas, planilhas eletrônicas para apresentação de informações e/ou resultados. O software para modelagem de decisão deve suportar tomada de decisão em grupo, bem como técnicas de estruturação de decisões do tipo *brainstorming*, análise de multi-atributos, dentre outros. Por fim, os software de comunicação devem suportar trabalho cooperativo, incluindo funcionalidades de áudio, vídeo, troca de dados e textos, para grupo de pessoas que estão ou em rede local, ou em redes de longa distância.

O parâmetro *organizationware* desta taxonomia tem como finalidade melhor organizar os dados, os processos de grupo no que diz respeito ao relacionamento entre os membros (autoritário, democrático, conflitante, consensual, dentre outros), ao direcionamento das discussões e escolhas efetuadas, e a gerência de procedimentos no trabalho em grupo colaborativo.

O ultimo parâmetro, pessoas, diz respeito aos membros do grupo e à equipe de suporte necessária, a fim de facilitar as atividades dos participantes. Às vezes faz-se necessário um facilitador. A figura do facilitador é de grande importância num SADG, pois é ele quem ensina os participantes do grupo a utilizar de forma adequada o hardware e software disponível. Outra atribuição do facilitador é a de conduzir as atividades dos membros do grupo de maneira que todos executem os passos estabelecidos para o processo decisório com intuito de obter-se um bom resultado da reunião.

Com base nestes quatros parâmetros, [KRA 88] propõe seis tipos de SADG, conforme Tab. 2.2 abaixo:

TABELA 2.2 – Tipos de SADG [KRA 88] .

Tecnologia	**Descrição**
Quadro eletrônico	Computador e audiovisual
Facilidade de Teleconferência	Computador e comunicação
Redes de grupos	Computador, redes e conferencia interativa
Centro de informação	Computador, banco de dados e ferramentas de recuperação
Salas de conferência	Computador e modelos de decisão
Laboratórios de colaboração	Computador e ferramentas de colaboração

Quadro eletrônico é um dos recursos mais relevantes de SADG. Estudado e implementado desde meados de 70 das mais variadas formas, o quadro eletrônico, por suas funções, assemelha-se bastante a um projetor de slides. O ponto chave é que o quadro eletrônico, por utilizar o computador, permite funções como armazenamento, recuperação e programação previa de apresentações.

A teleconferência é basicamente composta por dois tipos de participantes: os ativos e passivos. Os participantes ativos são aqueles que conduzem e discutem as questões levantadas pelos participantes passivos e, os participantes passivos são aqueles que absorvem e por vezes enviam algumas questões a serem discutidas através de fax, telex, e outros recursos que suportem transmissão digital do tipo voz, imagem, e dados.

Um dos pontos mais críticos para implantação de uma teleconferência é o custo. A aquisição dos recursos de áudio, vídeo, imagem, dentre outros, implica um custo muito alto que grande parte das organizações não dispõe e muitas vezes não pretende investir tão alto para utilização desta tecnologia.

Redes de grupo tem a mesma filosofia da teleconferência por computador. Entretanto, foi eliminada a necessidade das salas eletrônicas de reuniões. Também o uso do computador é mais amplo, pois as redes de grupos tem como foco da suporte por computador a pequenos grupos, dispersos fisicamente mas próximos geograficamente.

Desta forma, as redes de grupos tratam mais profundamente a habilidade dos participantes em ter acesso e manipular ferramentas baseadas no uso do computador. Nas redes de grupo, cada participante terá o seu computador para comunicar-se com os outros membros do grupo. Para controlar o fluxo de informações e a comunicação entre os participantes, pode existir um condutor da reunião que tenta facilitar ao máximo a reunião conduzindo da forma mais adequada, utilizando as funcionalidades disponíveis no ambiente.

Kraemer e King limitam este tipo de SADG a apoio de reuniões distribuídas/síncronas. Entretanto, acredita-se que, com os recursos de *hardware*, *software*, *oraganizationware* e pessoas, e sua proposta de coordenação, este tipo de SADG pode contemplar perfeitamente apoio à decisões distribuídas do tipo assíncronas. Neste trabalho, a base tecnológica de redes de grupo será considerada na sua forma estendida, dando apoio a decisões assíncronas e distribuídas.

Para o SADG denominado Salas de Conferencia, é necessária uma sala equipada com retroprojetor, computador, terminais para votação e principalmente um terminal de controle. Os programas para esse tipo de base tecnológica deverão utilizar técnicas do tipo arvore de decisão, análise de multí-atributos e métodos de votação, dentre outros. Faz-se necessário também, em nível de controle organizacional da reunião, protocolos para validar restrições de acesso e tipo de atividades permitidas para cada participante.

Centro de informações são recursos de processamento de dados organizados e dedicados a suportar usuários (gerentes e profissionais) em atividades do tipo geração de relatórios e manipulação de dados, através de pacotes de programas para gerenciamentos de dados, analise estatística, editores de texto, geradores de relatórios, dentro outros. Existe uma tendência de adaptar um centro de informações para um grupo de usuários especifico(ex: que prepara orçamentos, campanha de marketing, executivos de um empresa, dentre outros.) fornecendo ajuda para atividades de tomada de decisão daquele grupo, caracterizando portanto um centro de informações como um SADG. Um centro de informações necessita: ou terminais para acesso a dados, ou facilidades de salas de teleconferência (projetor de slides, retroprojetor, dentre outros) quando se trata de pequenos grupos.

Os laboratórios de colaboração permitem suporte por computador a um grupo trabalhando em interação face-a-face, permitindo dessa forma um contato direto com participantes do grupo. Os participantes do grupo irão interagir com os outros participantes através de estações de trabalho. Nelas estarão disponíveis ferramentas que

utilizam uma interface de modo a apresentar s informações publicas de forma compartilhada (WYSIWIS – *What You See Is What I See*) e as privadas em janelas separadas na estação permitida. Os protocolos adequados, assim como nas salas de conferencia, são necessários para suportar este tipo de interação.

A proposta de avaliar as bases tecnológicas apresentadas acima em relação aos parâmetros de *hardware*, *software*, *organizationware* e pessoas é muito importante não só para os SADGs já concebidos como para outros tipos que poderão surgir com o avanço da tecnologia da informação.

Embora apresente esta contribuição, as bases tecnológicas propostas não contemplam SADG com apoio a decisões que ocorrem em lugar e tempo diferentes (do tipo assíncrona distribuída). Esta observação parece ser um ponto fraco da contribuição de Kraemer e King porque existe uma quantidade bastante significativa de SADGs assíncronos distribuídos, dentre eles gIBIS[CON88], vIBIS[CES86][CES94],Co-Op[BUI86], QUORUN[ARA94]. Acredita-se que parâmetros como o SADG do tipo redes de grupo não precisam se restringir ao apoio à decisões assíncronas distribuídas, podendo ser estendidas para conceber decisões que ocorram em momentos e lugares distintos.

Outro fator importante é que as bases tecnológicas do tipo quadro eletrônico, salas de conferencia e teleconferência exigem um investimento muito alto para sua aquisição e grande parte das organizações não dispõem de recursos para tal investimento. Alem destes fatores, estas bases tecnológicas tem um propósito de reuniões mais restrito (apoiar teleconferência, apresentação através de quadro eletrônico, dentre outros). Dentre os SADGs mencionados anteriormente somente o de redes de grupo permite que a tecnologia seja utilizada para outros fins que não sejam exatamente para um SADG.

Em [DES 87] classificam as abordagens possíveis de suporte a grupos em três níveis de sistemas, visando destacar a troca de informações existentes no processo e decisão coletiva, O nível 1 diz respeito a SADGs que tem características únicas de suporte a comunicação. Os de nível 2 apresentam características de suporte analítico e/ou técnicas de estruturação de grupo. Por ultimo, os SADGs correspondentes ao nível 3 apresentam características de definição automática de padrões para condução de uma tomada de decisão.

Os SADGs correspondentes ao nível 1 oferecem funcionalidades visando remover barreiras de comunicação. Sabe-se que barreiras de comunicação entre os membros do grupo são bastante comuns, pois muitas vezes existem fatores como pressões hierárquicas, domínio do problema, opiniões contrárias, personalidades dominantes, dentre outros que podem afetar o andamento do processo. Portanto, estes SADGs tem por objetivo melhorar o processo de decisão, facilitando a troca de informações entre os membros do grupo através de transmissão eletrônica de dados, de recursos de exibição de telas públicas e agendas simuladas, preservando o anonimato na exibição de idéias e votos. Na Tab. 2.3 abaixo são citadas as funcionalidades básicas de SADGs deste nível.

TABELA 2.3 – Funcionalidades básicas de SADGs de nível 1 [DES 87].

Necessidades do Grupo	**Características**
Envio ou recebimento de informações entre todos participantes ou membros do grupo específicos.	Transmissão de mensagens eletrônicas
Ter acesso à banco de dados pessoais ou corporativos durante um processo de decisão.	Terminal de computador para cada membro do grupo; Ter acesso a uma rede local ou a um computador central
Apresentação das idéias, votos, dados gráficos ou tabelas para todos os membros do grupo simultaneamente.	SADG com uma tela comum para cada membro do grupo através de terminais
Relutância de alguns membros do grupo em falar por motivos de timidez, por ter um cargo de pouco status,dentre outros..	Anonimato para exibição de idéias e votos.
Falha de alguns membros em participar do processo por motivos de preguiça ou por estar fora de sintonia.	Solicitação de idéias ou votos para cada membro do grupo.
Falha em organizar e analisar idéias e votos de forma eficiente.	Sumario e exibição de idéias; sumario estatístico e exibição de votos;
Falha em quantificar preferências.	Fornecer avaliações de escalas e ou classificação de esquemas; solicitação e exibição de classificação e avaliações.
Falha em desenvolver uma estratégia ou plano de um processo de decisão.	Fornecer uma agenda simulada que o grupo possa completar
Falha em obedecer o plano de um processo de decisão	Continuamente exibir a agenda; fornecer um relógio; automaticamente exibir os itens da agenda no tempo apropriado.

Os SADGs referentes ao nível 2 fornecem técnicas de modelagem e/ou estruturação de grupos a fim de reduzir incertezas e ruídos que podem ocorrer no processo de decisão em grupo. Em SADGs que contemplam o nível 2, poderá aparecer a figura de um facilitador. Algumas das características que os SADGs referentes ao nível 2 são apresentadas na Tab. 2.4 abaixo:

TABELA 2.4 – Funcionalidades básicas de SADGs de nível 2 [DES 87].

Necessidades do Grupo	**Características**
Necessidade de estruturar, planejar e esquematizar um problema de decisão.	Modelos de planejamento, por exemplo PERT, COM, Gantt.
Auxilio de decisão analítica para eventos futuros incertos.	Modelos de avaliação de probabilidade e utilidade, por exemplo, arvores de decisão, avaliação de riscos.
Auxilio de decisão analítica para problemas de alocação de recursos.	Modelos de alocação de orçamentos.
Auxilio à decisão analítica para tarefas orientadas a dados.	Métodos estatísticos, modelos de decisão multicritérios.
Auxilio de decisão analítica para tarefas preferenciais.	Modelos de julgamento social.
Permite utilizar uma técnica de decisão estruturada, mas o conhecimento ou tempo para usar a técnica é insuficiente	Reunir e automatizar técnicas (Delphi, Nominal) que oferecem um tutorial on line para um grupo ou para um facilitador.

Por ultimo os SADGs referentes ao nível 3 são aqueles em que a própria máquina conduz o processo de decisão a grupo, estabelecendo padrões de comunicação, tempo e troca de informações, dentre outros mecanismos necessários quando existe uma discussão em grupo. Nestes tipos de sistemas a máquina executa exatamente as atividades que um facilitador faria. Exemplos de problemas e características necessárias de SADGs deste nível são mostrados na Tab. 2.5 abaixo.

TABELA 2.5 – Funcionalidades básicas de SADGs de nível 3 [DES 87].

Necessidades do Grupo	**Características**
Deseja forçar procedimentos de decisão formalizada.	Procedimentos parlamentares automatizados.
Deseja solicitar e combinar uma ordem de regras para discussão	Regulamento base; facilidade para a seleção e aplicação da lei
Incerteza sobre opções de procedimentos para reunião	Consultor automatizado, dando conselho sobre uma lei disponível e o uso apropriado
Deseja desenvolver regras para a reunião	Facilitada para escrever regras.

Assim como a classificação de [KRA 88], esta taxonomia também é uma forma interessante de identificar tipos de SADG de acordo com o nível de apoio em relação à troca de informações.

2.2 Efeitos do SADG

Nota-se nas taxonomias apresentadas acima, a variedade de contextos e de ambientes para o apoio à decisão, que pode acarretar numa serie de implicações e portanto, gerando efeitos variados sobre como organizações e pessoas trabalham, e o que estas esperam de um SADG [KAR 96][STO 92]. Em [STO92], é mencionado que pouco se sabe sobre os reais efeitos dos SADGs. No entanto, são delimitados alguns efeitos desses tipos de ferramentas que são esperados pelas pessoas e pela organização na qual está implantada, dentre eles:

1. efeitos sobre o desempenho de um indivíduo trabalhando em grupo;
2. efeitos sobre o desempenho do grupo;
3. efeitos sobre como as organizações trabalham;
4. efeitos sobre o relacionamento organizacional;
5. efeitos do software
6. efeitos do hardware.

Em um processo de decisão não informatizado as pessoas têm um espaço para apresentar suas idéias. A satisfação de uma pessoa trabalhando em grupo num SADG está ligada ao sucesso do mesmo conseguir desenvolver suas atividades neste tipo de ambiente, de tal maneira que, possa contribuir no mínimo igualmente que num ambiente convencional, e que fique satisfeito com os resultados atingidos. Por outro lado pode-se tornar desconfortável o trabalho em grupo para pessoas que nunca participam de um processo de decisão, pois não conseguem produzir quando interagem com outras, por questões de inibição ou ainda por não conseguir organizar suas idéias.

Quanto aos efeitos sobre o desempenho do grupo quando da utilização de um SADG, [STO92] afirma que se o desempenho de uma pessoa dentro do grupo melhora com o uso de SADG, a tendência é que o desempenho do grupo melhore proporcionalmente. Entretanto, existem alguns fatores que impedem que isto ocorra por completo, dentre as quais: tamanho do grupo, estrutura e protocolo do grupo e número de pessoas por equipamento.

O efeito de SADGs sobre como as organizações trabalham é ainda bastante difícil de ser tratado, pois poucas empresas utilizam esse tipo de tecnologia de forma que possa ser feito uma avaliação empírica adequada. No entanto, nota-se que pelo objetivo e a contribuição pretendida, que esse tipo de ferramenta tende a levantar um questionamento ou até proporcionar uma nova forma das organizações trabalharem.

Os efeitos do SADG no relacionamento organizacional está estritamente ligado na idéia de como grupos de indivíduos de um SADG podem acessar os dados e quais as restrições de acesso. Para que se alcance bons resultados, se torna necessário que o SADG possua mecanismos de anonimato de maneira que os usuários não se sintam constrangidos nas suas contribuições por fatores do tipo pressões hierárquicas.

As ferramentas SADGs devem ser projetadas de forma a apresentar uma interface amigável, corresponder aos requisitos requisitados pelos usuários do sistema, contemplar mecanismos para condução do processo de decisão e também mecanismos como, votação, métodos analíticos, negociação, dentre outros, com o propósito de auxiliar o processo de decisão propriamente dito.

Os efeitos do hardware num SADG é um dos aspectos que mais ocorrem variações. O hardware pode diferenciar o tipo de plataforma, equipamentos relacionados a velocidade, capacidade de armazenamento e software que poderão ser utilizados. Este são alguns dos aspectos que podem influenciar quanto ao custo do hardware necessário para a utilização de um SADG.

Existem muitos outros tipos de efeitos relacionados a SADGs na literatura especializada. Entretanto, não é objetivo deste trabalho, exaurir todos os tipos de efeitos. Os efeitos acima abordados estão diretamente ligados ao trabalho em grupo, e mais ainda, no foco ao desenvolvimento e comportamento de cada participante em determinados contextos de decisão.

Na seção seguinte serão apresentados alguns conceitos sobre processo de software, seu caráter cooperativo que servirão de fundamentos para a proposta do presente trabalho.

2.3 Processos de Software

O processo de desenvolvimento de software corresponde a um conjunto de atividades complexas que precisam ser executadas por um grupo de pessoas com habilidades distintas. Uma forma de analisar e evoluir tal processo é através da sua descrição, a qual consiste de um modelo de processo de software. A descrição formal de um processo de software é uma atividade que torna possível, a análise, a compreensão e a automatização do processo(execução). Desta forma, a modelagem e a execução de processos de software são de fundamental importância para o aumento da qualidade do produto de software e têm sido estudadas pela comunidade de engenharia de software com a finalidade de fornecer ferramentas que as suportem.

As atividades de um processo de software segundo [COR 94] produzem mudanças de estado visíveis externamente no produto de software. Atividades incorporam e implementam procedimentos, regras e políticas, e têm como objetivo gerar ou modificar um dado conjunto de artefatos. Elas podem ser organizadas em redes com duas dimensões e estão associadas com papéis, ferramentas e artefatos. Uma atividade aloca recursos (por exemplo, máquinas e orçamento), é escalonada, monitorada e atribuída a desenvolvedores (agentes), que podem utilizar ferramentas para executá-la. Uma atividade também pode ser executada somente por ferramentas automatizadas, sem intervenção humana. Toda atividade possui uma descrição, a qual pode especificar os artefatos necessários, as relações de dependência com outras atividades, a data de início e fim planejadas, os recursos a serem alocados e os agentes responsáveis pela mesma [DOW 91].

Construir software cooperativamente exige um gerenciamento e acoplamento constante das atividades sendo realizadas pelo grupo como um todo e daquelas realizadas individualmente por cada participante do grupo. É preciso, portanto, prover suporte para execução das tarefas do grupo e das tarefas individuais de cada membro através de mecanismos de definição, visualização e acompanhamento do estado do

processo.

A palavra chave relacionado à coordenação, portanto, refere-se a acompanhamento. Especificar como a interação se dará, definir regras e limites, estipular responsabilidades e controlar e acompanhar a execução de tarefas são questões que precisam ser apoiadas durante todo o processo. Estas questões quando organizadas e discutidas em grupo através de uma ferramenta SADG tendem a trazer uma melhor coordenação e execução das atividades envolvidas, bem como registro das justificativas de projeto *Design Rationale.*

A comunicação e percepção providas por um SADG podem ser importantes para o acompanhamento de tarefas ao contribuírem para que as questões de coordenação sejam resolvidas e decisões neste escopo sejam tomadas. Desta forma, os membros da equipe estarão envolvidos com o próprio acompanhamento do processo de trabalho, tornando a coordenação mais eficiente.

2.3.1 Modelagem de Processos de Software

Esta é uma questão de grande ênfase em relação aos ambientes de suporte ao projeto cooperativo de software. Muitas são as propostas para prover a modelagem e execução de processos levando em consideração o aspecto cooperativo do desenvolvimento.

Não sendo possível garantir a existência de um modelo de processo universal para suporte ao desenvolvimento de software, há uma evidente inclinação a oferecer a possibilidade de customização de processos para cada contexto de projeto através de linguagens para modelagem e representação de processos.

A modelagem de processo de software tem como objetivos principais:

Facilitar comunicação e compreensão entre as pessoas: a mesma representação do modelo pode ser compartilhada por todo o grupo de desenvolvimento;

Facilitar o aperfeiçoamento do processo: através de um modelo, é possível analisar o processo e descobrir pontos onde ele pode ser melhorado;

Reutilização: os processos não necessitam ser modelados todas as vezes que forem ser realizados. Suas descrições podem ser armazenadas e reutilizadas quando necessário;

Suportar gerência do processo: tendo um processo definido, é possível realizar estimativas e planejamentos.

Prover orientação automatizada do processo: um processo definido permite que ferramentas automatizem algumas partes do modelo e orientem os usuários no andamento do processo;

Suportar execução automatizada: um ambiente automatizado pode controlar o comportamento do processo definido, coletar métricas e reforçar as regras para garantir a integridade do processo.

As linguagens de modelagem de processos diferem nas informações que integram o processo e apresentam uma ou mais perspectivas diferentes relacionadas a estas informações. Segundo[CUR92], as perspectivas representadas mais comumente são:

Perspectiva funcional: representa quais atividades do processo estão sendo executadas, e quais os fluxos de informação são necessários para essas atividades;

Perspectiva comportamental: representa quando as atividades são executadas, assim como aspectos indicando como essas atividades são realizadas através da interação, condições de tomada de decisão, critérios de entrada e saída, dentre outros;

Perspectiva Organizacional: Representa onde e por quem (agentes) na organização as atividades são executados, os mecanismos físicos de comunicação usados para transferência;

Perspectiva Informacional: representa as entidades de informação produzidas ou manipuladas por um processo. Estas entidades incluem dados, produtos(intermediários e finais) e objetos. Esta perspectiva inclui a estrutura das entidades de informação e os relacionamentos entre elas.

Estas perspectivas, quando combinadas, produzem um modelo de processo de software integrado, consistente e completo.

Alguns ambientes oferecem uma linguagem de modelagem de processos exclusiva, como o ambiente SPADE (*Software Process Analysis, Design and Enactment*) [BAN 93], que define a linguagem SLANG (*Spade Language*), baseada em Redes de Petri de alto nível. O ambiente EPOS [EPO 96] segue a mesma idéia através da linguagem SPELL (*Software Process Evolutionary Language*), baseada num modelo de objetos contendo as classes que descrevem as entidades envolvidas na modelagem do processo. Outros, como o ambiente OZWeb [OZW 96], permitem o uso de qualquer paradigma para modelagem de processos.

Em algumas propostas, como em EPOS [EPO 96] e *Conversation Builder* [KAP 92], há também a preocupação com a modelagem de meta-processos, a definição de metodologias de modelagem e a criação de modelos de processo reutilizáveis.

A definição de meta-processos e sua customização também são as preocupações do ambiente CPCE [LON 96]. Sua proposta enfoca o processo através de um modelo de discussão onde questões, posições e argumentos são apresentados em relação a artefatos sendo manipulados pelos participantes ou em relação aos passos definidos para a execução da atividade em andamento. Portanto, processo e meta-processo se confundem numa mesma estrutura de argumentação. Processos são especializados (instanciados) a partir do modelo geral de processos e podem ser customizados para atender a requisitos específicos de determinado contexto de colaboração. Alterações no processo são realizadas através do lançamento de questões quanto ao meta-processo.

Uma filosofia similar segue o ambiente *Conversation Builder* [KAP 92]. Neste, as atividades realizadas são consideradas como “conversas”, seguindo a filosofia de Winograd e Flores de “conversa para ação”. Um protocolo de conversação provê um conjunto de regras que governam o comportamento das conversas instanciadas sob aquele protocolo, tais como as atividades que podem ser realizadas, por quem serão realizadas, sob quais circunstâncias e quais seus efeitos.

Numa tentativa de flexibilizar ainda mais a modelagem de processos, o ambiente RASP [SWE 93] se baseia na automação e reengenharia de processos de software, permitindo que membros de equipes tomem parte do planejamento e redefinição de processo. RASP evita o problema da complexidade na modelagem de processos por não exigir que os modelos estejam completos para serem encenados.

2.3.2 Execução de Processos de Software

As propostas para o controle da execução dos processos de software focalizam principalmente a questão da evolução dinâmica do processo (alterações no processo no decorrer de sua execução).

No ambiente SPADE [BAN 93], por exemplo, as atividades podem ser instanciadas dinamicamente durante a execução do processo. Analogamente, no ambiente OZWeb [OZW 96], os modelos de processo são encenados em uma máquina de *workflow* oferecida pelo ambiente que permite a associação dinâmica de usuários a atividades, permitindo, inclusive, a delegação de tarefas durante a execução do processo.

Outras questões são abordadas com o objetivo de livrar a execução de processos de um controle rígido e restritivo. O ambiente OZWeb, por exemplo, tem a preocupação em tolerar inconsistências na execução, quer que seja possível operar com informações incompletas e permitir fragmentações no processo. O projeto do ambiente OZWeb apresenta, ainda, um modelo para ambientes descentralizados centrados em processo. A tentativa é a de explorar a modelagem e a execução de colaboração entre grupos por processos independentes, autônomos e possivelmente preexistentes.

A mesma preocupação com execução flexível pauta os objetivos do ambiente Coo. Uma característica central deste ambiente surge do fato de que existem limites para uso de abordagens puramente baseadas no conhecimento prévio de todas as possibilidades de interação no decorrer do processo. A proposta do ambiente Coo está em basear a execução num modelo de transação seguro que, no momento em que o conhecimento sobre o processo seja insuficiente, possa definir um modo de sincronização padrão para sua execução [GOD 94].

2.3.3 Modelagem de Papéis

Em todos os ambientes e ferramentas apresentados, existe a preocupação na modelagem de seus usuários e delegação de responsabilidades aos diferentes tipos de usuários modelados.

Em geral é possível existir vários “papéis” em grupo de pessoas que trabalha junto. Por exemplo, em uma equipe de desenvolvimento de software pode-se identificar o coordenador, que em geral é atribuído ao chefe de equipe, o avaliador, responsável pela qualidade do software, o especialista, detentor de um conhecimento específico sobre a aplicação em construção, o analista e o programador, que são os executores da implementação do software respectivamente.

Alguns ambientes/ferramentas se preocupam em classificar explicitamente os usuários em grupos através de suas características, responsabilidades e papéis em relação ao processo [OZW 96][CAV 94][BEL 95]. Outros ambientes, como o EPOS [EPO 96] e *Conversation Builder* [KAP 92], também associam usuários à execução de atividades do processo mas não parecem se preocupar com a classificação e modelagem de usuários em papéis.

2.4 Processo de Decisão durante o Desenvolvimento de Software

Sobre o processo de decisão durante o desenvolvimento de software, é importante que sejam considerados os seguintes aspectos: a organização da equipe de decisão, as responsabilidades de cada participante, as deficiências do processo e suas necessidades.

Uma equipe de desenvolvimento pode ser organizada de duas formas. Os participantes podem ser formalmente indicados para exercerem um determinado papel ou podem, no decorrer do processo, mostrar espontaneamente tendências a assumir determinados papéis ou responsabilidades [DUA 92]. A organização formal é utilizada principalmente em organizações onde predomina a estrutura hierárquica de trabalho. Na maioria das vezes, entretanto, no decorrer do desenvolvimento são observadas incoerências na organização da equipe. Elementos podem não estar aptos a realizar as responsabilidades que assumiram, outros mostram interesses e aptidões maiores para outras tarefas que não as quais foi alocado e, observamos, sobretudo, alguns elementos acumulando tarefas e responsabilidades enquanto outros relegam as suas. Quando a estrutura da organização permite, estes elementos podem ser relocados na tentativa de superar estas dificuldades.

Dentro das equipes, a tomada de decisão costuma ser realizada por um subgrupo de pessoas envolvidas no processo de desenvolvimento. Novamente, este grupo pode ser formado através de imposições, quando os responsáveis pela decisão são os membros de níveis hierárquicos superiores ou especialistas no assunto em questão; ou espontânea e democraticamente, onde os membros discutem livremente a questão, buscam o consenso e a palavra final, ou seja, a decisão é daquele que conseguir convencer os outros com os seus argumentos.

Em ambos os casos, os elementos do grupo podem ser destacados de acordo com as responsabilidades assumidas perante o processo. Estas responsabilidades definem, basicamente, os seguintes papéis:

- Perito: os peritos são os especialistas do domínio do problema a ser resolvido. Sua principal função é a de apoiar a discussão com a apresentação de informações, fatos e opiniões sobre o assunto sendo discutido. Os peritos são identificados basicamente pelos participantes do projeto de desenvolvimento e usuários. Consultores externos ao projeto, chamados para apresentar seu parecer sobre determinados pontos da discussão também são considerados peritos.
- Facilitador: Os facilitadores se preocupam em estimular a troca de informações durante a decisão, solicitar a contribuição dos membros da equipe ou de elementos fora dela, ficar atento aos problemas que ocorrem durante a discussão e, possivelmente, resolvê-los. Trabalham também na organização como e no andamento do processo de decisão. Distribuem tarefas e controlam o processo de decisão, levando em consideração os prazos e garantindo a chegada à decisão final.
- Redator: Os redatores são responsáveis por documentar o processo de decisão e todas as conclusões as quais chega a equipe.

A organização do grupo e as responsabilidades assumidas por cada participante, como dito anteriormente, podem se manter estáticas ao longo do processo de decisão ou mudarem constantemente, ou podem também possuir outra configuração como tendo apenas um facilitador e agentes, que podem ser peritos, redatores ou até mesmo clientes.

O processo de decisão, por si só, é uma atividade bastante complexa. Levantar as alternativas possíveis de solução, analisá-las com detalhe e por fim escolher a mais adequada é uma tarefa que exige, muitas vezes, o uso de métodos para este fim. Porém, o processo de decisão ideal não se resume a encontrar uma melhor solução para o problema e aplicá-la. Também é de grande importância registrar detalhadamente os passos, discussões e as justificativas que levaram à chegada a tal solução. Ou seja, mais do que registrar o que foi escolhido, devemos ainda, registrar porque tal solução foi escolhida e porque outras foram descartadas.

Particularmente no desenvolvimento de software, este registro se faz ainda mais necessário [LUC 91][POT 88]. A ausência das informações relativas aos "porquês" de cada decisão e os resultados de cada uma prejudicam o próprio processo de desenvolvimento e a futura manutenção do sistema. É extremamente freqüente nos depararmos durante um projeto com as seguintes perguntas: "Porque isso ficou assim?", "Quem decidiu tal alteração?", "Em que se foi baseado a criação disso?", "O que tínhamos antes de mudar tal coisa". Estas perguntas são feitas com freqüência durante o desenvolvimento e costumam gerar problemas como: atrasos, pois o grupo precisa fazer uma recapitulação das decisões; desconfianças, pois se os motivos não conseguem ser esclarecidos, a segurança dos desenvolvedores pode diminuir dai em diante. Se durante o processo estas perguntas não encontram respostas imediatas, na manutenção do sistema este problema torna-se ainda mais crítico.

Estas perguntas poderiam, todavia, ser respondidas se fosse possível registrar cada passo das decisões realizadas. Quanto mais detalhado o histórico das decisões, maior o número de informações para serem consultadas no momento desejado. Além disso, não podemos nos esquecer que estes registros são fontes de grande importância para a reutilização em outros projetos.

As deficiências do processo de decisão, dentro do contexto do processo de desenvolvimento de software, são análogas às deficiências encontradas nos processos de decisão em grupo em geral. Os problemas da interação em grupo como, inibição e influências de poder e interesses pessoais, são freqüentes e de difícil solução principalmente quando o grupo possui uma organização hierárquica.

Os problemas de comunicação como: omissão de informações, apresentação de informações irrelevantes para o processo, divagação, negligência quanto às possíveis alternativas de solução, desperdício de tempo dentre outros, também aparecem com freqüência, prejudicando em muito a produtividade do projeto em desenvolvimento.

O suporte a cooperação é intrínseco ao processo de desenvolvimento e o mesmo se dá em três níveis: suporte à coordenação, suporte à comunicação e suporte ao compartilhamento de informações. A atividade de tomada de decisão também deve ser suportada tendo em vista estes três níveis de cooperação. Além dos três níveis citados, uma atividade de decisão deve oferecer um nível a mais de cooperação que vem a ser o suporte à decisão em si. Isto significa oferecer mecanismos para auxiliar o grupo a analisar as contribuições geradas e concluir quanto à melhor decisão.

No próximo capítulo serão apresentados alguns modelos de ferramentas SADG. Cada ferramentas possui características distintas quanto ao processo de decisão, de forma que nem todas as etapas mencionadas em [SIM 60][SIL 91] são contempladas nestes modelos. Entretanto, a discussão de tais modelos fornecem subsídios para o projeto do Modelo de Argumentação proposto neste estudo.

3 Ferramentas SADG

Neste capítulo serão apresentadas algumas ferramentas SADG caracterizada por [KRA 88] como do tipo redes de grupos, estendendo a concepção para o suporte assíncrono e distribuído. As ferramentas apresentam características que serviram de base para o modelo do SaDg PROSOFT.

Os SADGs são sistemas interativos que utilizam métodos que facilitam a solução de problemas não estruturados por um grupo de pessoas, auxiliando-os na tomada de decisão[ARA 94]. Dos métodos utilizados, os SADGs suportam por exemplo, mecanismos de votação, geração de idéias, identificação de alternativas, modelos de argumentação dentre outros...[ELL 91]

Em geral, os SADGs utilizam modelos de argumentação para capturar e estruturar melhor o raciocínio e as discussões, que formam a memória de grupo, gerados durante todo o processo decisório.

3.1 Modelos de Argumentação

O registro das interações, bem como os passos realizados para a tomada de decisão são questões que norteiam pesquisas em sistema de suporte à decisão de grupos. Quando se fala em suporte ao processo de colaboração entre elementos de uma equipe, seja para o desenvolvimento de um software, ou para o lançamento de uma campanha...este registro torna-se mais vital. Portanto, a modelagem dos dados a serem armazenados se amplia para comportar outras informações provenientes da interação em grupo além dos produtos a serem alcançados.

A forma de captura de conhecimento, conforme realizada na maioria dos processos de característica colaborativa, se concentra na preservação de documentos nas mais variadas formas de representação. Este conhecimento pode ser encarado como o conhecimento formal e é nele que o grupo se baseia como memória de trabalho.

Entretanto, o conhecimento dito informal, que é a descrição pela qual os produtos foram criados, compreendendo o registro das idéias, fatos, questões, pontos de vista, discussões, decisões, dentre outros, que aconteceram no decorrer do processo e acabaram por defini-lo, é difícil de ser capturado. Esta racionalização (*rationale*) do processo e outras formas de conhecimento informal devem estar intimamente relacionadas aos produtos sendo produzidos.

Esta captura de justificativas, racionalização, tem sido realizada através de modelos de argumentação. Na seções seguintes serão apresentados alguns destes modelos que serviram de base para o modelo de argumentação do SADG proposto.

3.1.1 IBIS

O modelo IBIS (*Issue Based Information Systems*), desenvolvido por [CON88], tem por objetivo auxiliar os participantes do grupo a apresentarem seus pontos de vista, utilizando-se de suas habilidades para resolução de uma questão do projeto. O modelo implica em classificar todos os elementos básicos de uma discussão em: tema, posição e argumento, organizados por relacionamentos pré-definidos. O nodo tema apresenta o problema de decisão a ser discutido. O nodo posição apresenta alternativas para o problema informado no nodo tema. No nodo argumento são justificadas as alternativas como objeção ou apoio a uma ou mais posições.

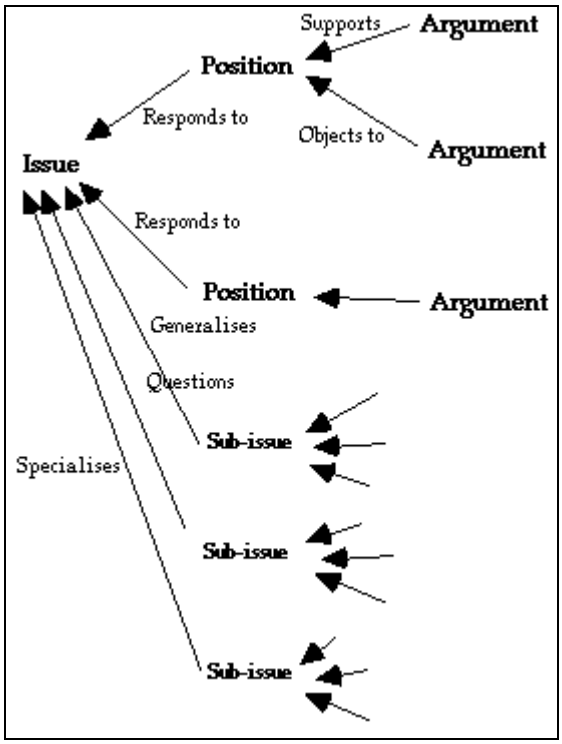


FIGURA 3.1 –Modelo de argumentação IBIS.

3.1.2 QOC

O modelo QOC, usado pelo *Design Space Analysis* (DAS) [SHU 94], envolve quatro elementos: questão, opção, critério, e argumento. As questões são os problemas a resolver, as opções são as alternativas apresentadas para os problemas identificados, os critérios justificam as opções existentes, e os argumentos são utilizados para conduzir a discussão sobre os demais elementos.

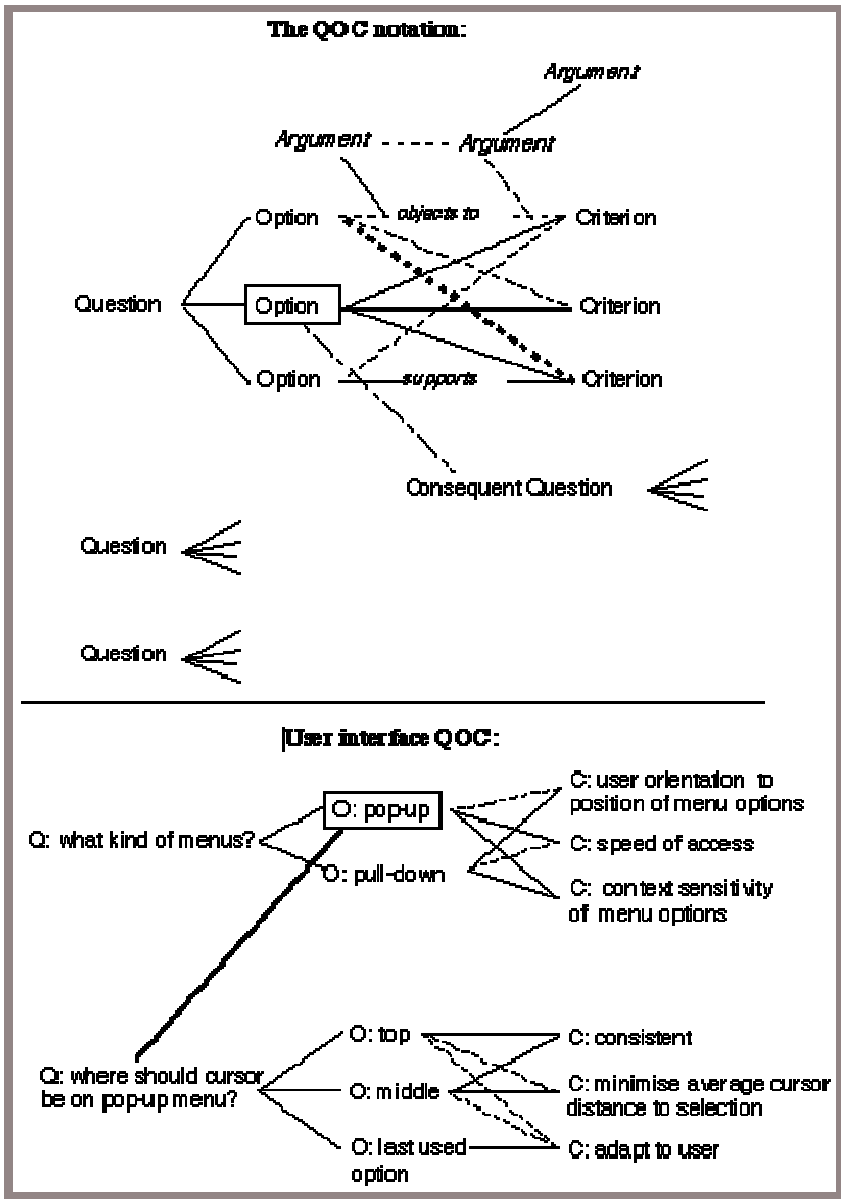


FIGURA 3.2 – Representação do modelo QOC.

3.1.3 DRL

O modelo DRL (*Design Rationale Language*), é considerado uma extensão do modelo IBIS, pois inclui, explicitamente, a noção de objetivos. Os objetivos estão relacionados ao problema em discussão, sendo as alternativas de soluções avaliadas através de alegações, em relação a estes objetivos. No modelo IBIS os objetivos estão implícitos nos argumentos que apoiam as alternativas (posições).

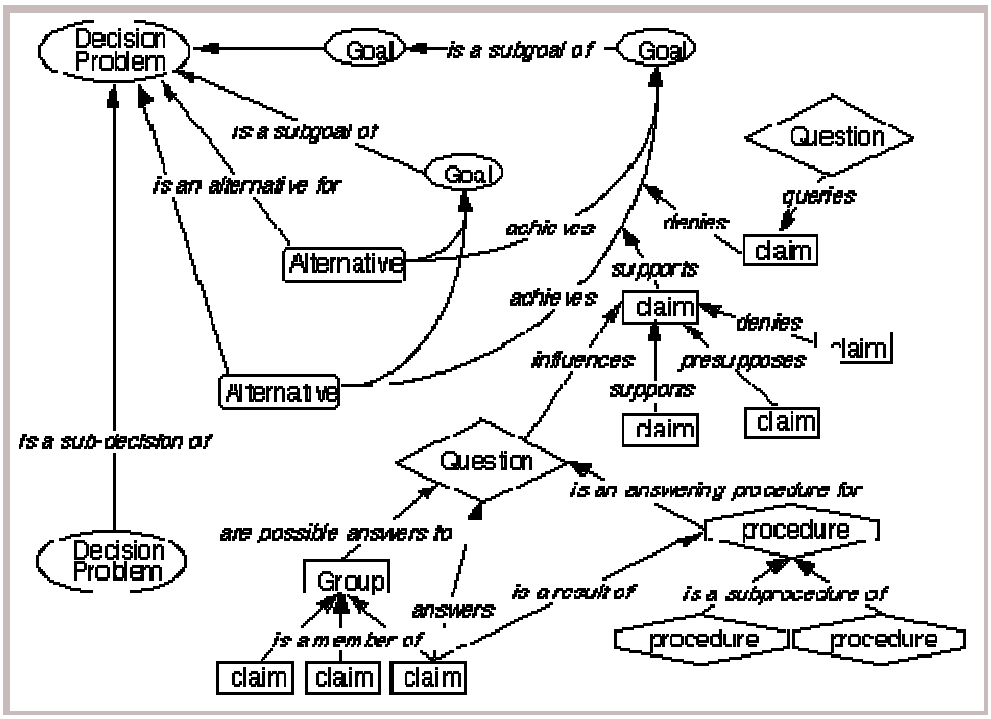


FIGURA 3.3 – Representação do modelo DRL.

Nem todas as reuniões são decisórias. Alguns grupos se reúnem para discutir um assunto, avaliar um documento, ou simplesmente estudar. Os sistemas que apoiam este tipo de reunião são conhecidos como Sistemas de Apoio à Discussão. O modelo DRL é mais complexo e voltado para a tomada de decisões, enquanto os modelo IBIS e QOC, são mais genéricos, podendo ser utilizados para apoiar discussões.

3.2 Ferramentas SADGs

3.2.1 gIBIS

A ferramenta gIBIS é um SADG onde os participantes de um processo de decisão podem estar dispersos geograficamente em tempos diferentes, ou seja, prover um mecanismo assíncrono distribuído.

A ferramenta utiliza um modelo de argumentação baseado no modelo IBIS, e também, um modelo hipertexto para o apoio a estruturação das idéias geradas durante um processo de decisão. Seu principal objetivo é fornecer mecanismos que auxiliem a geração e disseminação de idéias de forma a armazenar um histórico das decisões para estruturar a comunicação e a informação obtida.

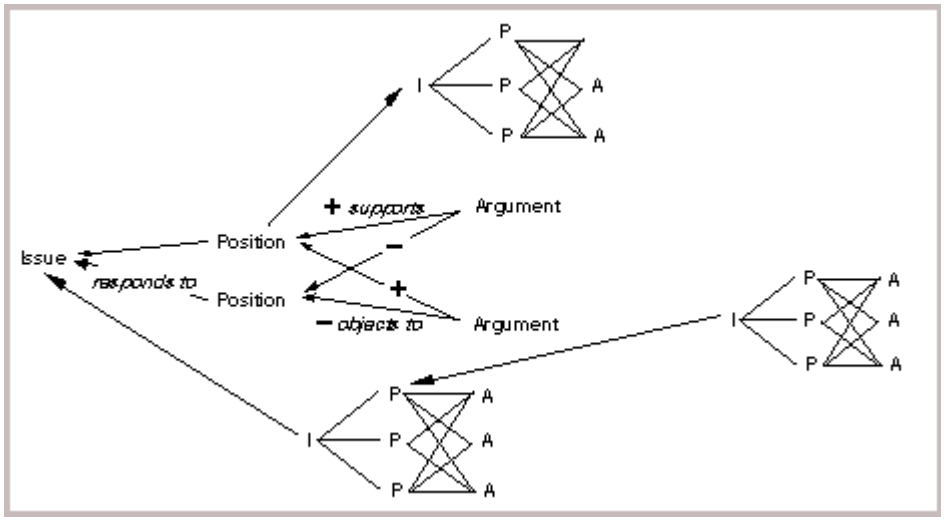


FIGURA 3.4 – Estrutura de argumentação do modelo gIBIS.

Os principais componentes da ferramentas estão distribuídas conforme segue:

- *Browser* gráfico permitindo manipular e visualizar a estrutura gráfica dos nodos (i.e. nodos de discussão e seus relacionamentos);
- Índice estruturado dos nodos, que fornece uma visão ordenada hierárquica dos nodos da rede em uso;
- Um painel de controle abaixo da estrutura indexada composta por um conjunto de botões que auxiliam nas funcionalidades da ferramenta - consulta, help, navegação entre nodos) e
- Um janela de inspeção para visualizar os atributos e conteúdos de nodos e links.

Em [CON88] são destacados dois tipos de usuários da ferramenta gIBIS. Um grupo usam-no como ferramenta de hipertexto isolada para estruturar suas idéias, e o outro grupo utiliza o gIBIS como mecanismo de comunicação estruturado. Conforme as

funcionalidades de SADGs discutidas no capitulo 1, gIBIS pode ser caracterizado como nível 1, por eliminar barreiras de comunicação, e de nível 2 por usar um modelo de argumentação que elimina os ruídos desta comunicação [DES 87].

Entretanto, no que diz respeito a etapa de decisão, o gIBIS possui algumas limitações. Não é provido nenhum mecanismo à fase de escolha do processo, já que limita-se a apoiar discussões remotas e assíncronas. A falta de um mecanismo que controle o processo, também dificulta a tomada de decisão na medida em que os usuários podem continuar discutindo indefinidamente, sem conseguir convergir, resultando em uma perda de tempo desnecessária.

O apoio oferecido pela ferramenta gIBIS foi descrito em [BEL95][BEL96] como de pré-reunião, ou seja, para uma fase anterior a reunião propriamente dita, a fim de que seus participantes tenham melhor conhecimento das idéias e tarefas que deverão ser abordadas durante a mesma.

3.2.2 vIBIS

A ferramenta vIBIS (*voting* IBIS) [CES94][CES96] é um sistema de discussão e votação. Tal como o gIBIS, ele é baseado no modelo de argumentação IBIS, e fornece um apoio assíncrono e distribuído. O destaque da ferramenta esta no processo decisório, no qual os membros podem, além de discutir e disseminar as idéia, ter apoio para escolher uma ou mais entre diversas alternativas levantadas, através de diferentes técnicas de votação [STR80][CES96][SCH83].

As principais etapas da ferramentas estão distribuídas conforme segue:

- Identifica-se o número de turnos (números de votações que serão executadas). Para cada turno deverão ser determinados os seguintes parâmetros: data de inicio, data final, lista de votantes, quem tem acesso aos resultados do processo;
- Define-se o método de votação para cada turno;
- Determina-se o peso do voto de cada participante;
- Obtém-se como saída um único vencedor, um conjunto de vencedores ou ainda uma lista ordenada de alternativas preferenciais.

O processo de votação permite que os participantes votem de maneira secreta, possibilitando que os membros solicitem recontagem de votos e ainda criem nós de votos de acordo com a hierarquia de seus participantes pois, no vIBIS, de acordo com a hierarquia, cada voto dado pelos participante tem pesos diferentes.

Durante o processo de decisão, aparece a figura do gerente que controla os parâmetros de votação, o que não é permitido a qualquer usuário e o coordenador que controla todo o processo decisório. Entretanto, não fica claro em que momento do processo de votação do vIBIS o método de votação(aprovação, pluralidade) é selecionado para ser empregado posteriormente pelos membros do grupo na hora de efetuarem seu voto.

A principal contribuição desta ferramenta é o emprego de técnicas de votação combinado com o uso de um modelo de argumentação. Sob o aspecto decisório, a utilização destas técnicas permitem que a ferramenta dê apoio à fase de escolha proposto no modelo de processo decisório de Simon. [SIM60][SIL91]

Pode-se caracterizar o vIBIS pelas funcionalidades apresentadas como de nível 1

e 2 proposto por [DES87] na etapa de deliberação.

Um detalhe importante apresentado neste SADG é que o autor dedicou uma atenção muito grande com o protocolo de comunicação entre as estações de trabalho. Este pode ser o motivo de algumas limitações que existem no vIBIS, entre as quais: não e explicada a interação entre os processos de discussão e votação, não fica claro a forma na qual as alternativas são escolhidas a partir do modelo de argumentação,dentre outros.

3.2.3 Co-Op

Ao contrário das ferramentas apresentadas nas seções precedentes, Co-Op [BUI 95] não utiliza um modelo de argumentação para estruturar a discussão. O objetivo da ferramentas e apoiar esta fase através de métodos de decisão multi-criterios(MDMC), permitindo que toda a comunicação apoiada seja dirigida a este objetivo.

O uso de MDMC permite a representação de múltiplos pontos de vista de um problema através de uma matriz de decisão, onde as linhas representam as alternativas de decisão e as colunas representam os critérios e pontos de vista. Permite também, a agregação de preferências dos múltiplos tomadores de decisão de acordo com as normas do grupo (dentre elas os pesos estabelecidos para cada critério).

As principais atividades de decisão da ferramentas estão distribuídas conforme segue:

- Definição do problema;
- Definição das normas de grupo;
- Priorização de critérios de avaliação;
- Seleção individual de alternativas;
- Seleção do grupo de alternativas e
- Busca de consenso e negociação.

O SADG Co-Op não apresenta funcionalidades que apóiem explicitamente a comunicação do grupo na etapa de definição do problema. No entanto está comunicação poderá ser realizada através de ferramentas de correio eletrônico, teleconferência ou telas publicas, que são independentes do Co-Op.

Durante o processo de decisão do Co-Op existe a figura de um gerente de comunicação ou gerente de grupo, que possui no mínimo três tarefas especificas, sendo elas: tarefa de coordenador, detetive e inventor.

Na etapa de definição de Normas o gerente de grupo define as pessoas que participarão do grupo e as atividades que deverão efetuar, além de efetuar o controle do andamento conforme definido anteriormente durante a definição da Norma. A tarefa implica basicamente em identificar os membros que participarão do processo, determinar senhas de acesso para cada membro, o acesso aos dados, conversação e acesso as técnicas de decisão em grupo.

No que diz respeito a priorização de critérios para avaliação, o Co-Op pode priorizá-los de em três modos:

- Combinada – onde membros participam de comum acordo uma prioridade a cada critério;
- Seqüencial – onde os membros priorizam um subconjunto de

critérios, e acordo com suas especialidades;

- Agregado – onde cada membro fixa primeiro pesos individuais, e depois, as prioridades individuais são agregados e computadas usando regras pré-determinadas.

As outras etapas do Co-Op envolvem o uso do MDMC.

A avaliação individual das alternativas pode ser feita por dois métodos de decisão multi-critérios: ELECTRE (seleção de uma das alternativas) e o AHP (processo hierárquico analítico de classificação de alternativas). A sumarização dos resultados do grupo é realizada utilizando corretamente técnicas de agregação e o peso da decisão do usuário, já definido na etapa de determinação das prioridades do critério de avaliação.

Caso não seja alcançada uma unanimidade dos resultados, o Co-Op sugere uma etapa de consenso e negociação, onde poderão ser utilizados os seguintes algoritmos: ELECTRE (onde é executada uma análise sensitiva baseado em seus parâmetros) e NAI (Negotiable Alternative Identifier) (como um mecanismo de negociação que busca alternativas que apresentam ao mesmo tempo o maior grau de aprovação e o menor de rejeição).

A complexidade do uso do MDMC restringe o tipo de usuários deste SADG, exigindo um conhecimento prévio de MDMC podendo ocasionar uma resistência a aprendizagem do mesmo. Entretanto, o emprego de uma Norma, que regula e controla todo o processo decisório, tais como a definição dos participantes, do MDMC, o acesso a dados permite uma melhor condução e controle do processo, favorecendo no resultado da mesma.

A utilização de uma Norma e de MDMC para avaliação de alternativas caracterizam este SADG como de nível 1 e 2, conforme [DES87], e também contempla as fases de inteligência, projeto e escolha propostas em [SIM60][SIL91].

3.2.4 QUORUM

A ferramenta QUORUM, desenvolvida por [ARA94][ARA95] é um SADG com o objetivo de apoiar as tarefas envolvidas no processo de desenvolvimento de software, que suporta ambiente remoto e assíncrono.

As tarefas de discussão e deliberação do grupo são realizadas utilizando o método de análise hierárquica (MAH) [ARA94], que tem por objetivo organizar o problema de decisão de forma estruturada.

Durante a atividade de discussão, é realizada a definição do problema, apresentando sua importância, o impacto e os objetivos a serem atingidos. Não só no MAH, mas como em outros modelos, esta etapa é de grande impacto em todo o processo pois, uma vez bem definido, os riscos de analisar e discutir questões irrelevantes torna-se menos arriscado.

Nesta etapa é permitido:

- Registro da discussão que envolve atributos pré-definidos: identificação do projeto de Software, fase de desenvolvimento, tarefa, assunto, descrição, data de início, prazo e data de término.
- Registro do problema de decisão, com os seguintes atributos: questão, data de início, prazo, data de término, descrição e prioridades de discussão.

- Registro dos participantes da discussão: nome, função e peso do julgamento).

Os elementos do grupo podem ser destacados de acordo com as responsabilidades assumidas perante o processo decisório. Estas responsabilidades definem, basicamente, os seguintes papéis:

- Perito: os peritos são especialistas do domínio do problema a ser resolvido. Sua principal função é a de apoiar a discussão com a apresentação de informações, fatos e opiniões sobre os assunto sendo discutido. Os peritos são identificados basicamente pelos participantes do projeto em desenvolvimento e usuários. Consultores externos ao projeto, chamados para apresentar seu parecer sobre determinados pontos da discussão também são considerados peritos.
- Facilitador: os facilitadores se preocupam em estimular a troca de informações durante a decisão, solicitar a contribuição dos membros da equipe ou de elementos fora dela. Ficar atento aos problemas que ocorrem durante a discussão e, porventura, tentar resolvê-los. Agem também na organização e no andamento do processo de decisão.
- Redator: os redatores são os responsáveis por documentar o processo de decisão e todas as conclusões as quais chega a equipe.

A etapa de argumentação é realizada pelos membros do grupo a fim de levantar as alternativas de ação para a solução o problema definido da etapa de definição. É nesta fase que o modelo de argumentação, que estende o modelo IBIS com outros tipos de nodos (fato, problema de decisão, alternativas, entre outros) e relações, é utilizado.

Na etapa seguinte, levantamento da hierarquia de critérios, o MAH propõe aos membros do grupo decompor o problema de decisão segundo os elementos: objetivos, critérios, subcritérios e alternativas.

No julgamento de prioridades todos os participantes do processo, através do MAH, avaliam as prioridades e preferências das alternativas e a importância de cada critério em relação ao objetivo, utilizando a abordagem de comparações em pares através de matrizes. Estes julgamentos são realizados sob a orientação do facilitador.

A tarefa de verificação de consistência será efetuada pelo próprio QUORUM logo após a entrada dos julgamentos dos membros do grupo em cada matriz de comparação, utilizando as diretrizes do MAH.

A última etapa do processo decisório é o consenso, que corresponde no MAH à resolução do problema de decisão através de uma alternativa “ótima” que foi sugerida dentre as muitas obtidas durante a discussão.

A ferramenta, através de suas características, facilita o processo de decisão durante as etapas para o desenvolvimento de software. O uso do MAH requer dos participantes do processo decisório um conhecimento prévio do mesmo. A utilização do MAH permite melhor organizar o problema de decisão e desenvolver alternativas de ação.

Estas características acima mencionadas permitem que a fase de escolha proposta no modelo de processo decisório de Simon e as características de SADG de nível 2 proposto por [DES 87], sejam contempladas no QUORUM. A utilização de um sistema de mensagens a parte do processo decisório, permitem a eliminação de ruídos,

atribuindo também características de SADGs de nível 1 à ferramenta.

3.2.5 ARCoPAS

O ARCoPAS é um ambiente para apoiar o processo de Engenharia Reversa, mais especificamente a Recuperação do Projeto Arquitetônico de Sistemas. As ferramentas existentes para apoiar este processo pouco investem na participação humana. Ao contrário destas ferramentas, o ARCoPAS alia extração automática da estrutura modular de um sistema, à extração cooperativa do conhecimento(a partir de um modelo de argumentação) dos especialistas sobre o mesmo. Este serviço oferecido pela ferramenta possibilita a recuperação de um documento mais completo, enriquecido por informações que só os especialistas possuem.

O ambiente divide a tarefa de recuperação em duas etapas: automática e cooperativa.

A etapa automática dificilmente consegue extrair dos códigos fonte uma descrição completa de cada módulo, baseando-se somente nos comentários encontrados. O enriquecimento da descrição de cada módulo depende da participação dos especialistas no sistema, quem em geral são os membros responsáveis pela manutenção.

Assim como ocorre na fase de projeto durante a engenharia progressiva, o processo reverso também exige que os membros da equipe discutam entre si. O registro desta discussão é importante para o processo, na medida em que, poderá facilitar o entendimento de como se originou o documento de Projeto. O modelo usado para capturar as discussões de Projeto é o IBIS na sua forma natural e intuitiva.

Para apoiar a etapa cooperativa, utiliza-se duas ferramentas: O FolioViews2 e a ItIBIS. O FolioViews é um sistema de hipertexto, através do qual é possível visualizar e complementar o documento de Projeto, assim como estabelecer uma discussão a cerca do mesmo. Para capturar esta discussão segundo o modelo IBIS, utiliza-se a ferramenta ItIBIS, implementada sobre o FolioViews.

Para apoiar a tarefa automática foi desenvolvida uma ferramenta chamada ExEM(Extrator da Estrutura Modular). Ela é responsável pela análise dos fontes: identificando os módulos e as suas relações, e gerando um texto com a estrutura modular no formato adequado para etapa cooperativa.

A ferramenta ItIBIS(Indented Text Issue-Based Information System), foi desenvolvida para permitir o uso independente de hardware e software, do modelo IBIS. A representação por endentação da estrutura hierárquica deste modelo pode ser implementada em qualquer editor de textos.

Na ItIBIS utiliza-se os códigos I, P, AS e AO para significar respectivamente os elementos Questão, Posição, Argumento de Suporte e Argumento de Objeção. Os símbolos "?", "*" e "-" são utilizados para significar respectivamente, pendência, aprovação/resolução e rejeição.

Para iniciar a discussão sobre a complementação do documento de Projeto, uma primeira Questão surge, a partir da análise dos códigos fonte, quando um módulo não possui uma descrição no formato esperado: "Qual a descrição deste módulo?". A ferramenta ExEM é capaz de encontrar nos códigos fonte, comentários que aparecem nas linhas correspondentes a cada módulo. Estes comentários são a primeira Posição para solucionar a Questão levantada. Como um Argumento a favor desta Posição, sabe-se que os comentários relativos a um módulo podem ajudar bastante na sua descrição.

Por outro lado, um argumento contra mesma posição é o fato dos comentários estarem fora de uma apresentação adequada. Este princípio de discussão é lançado pela própria ferramenta ExEM.

A introdução das característica cooperativa a uma ferramenta de apoio à Recuperação de Projeto traz uma importante contribuição para a Engenharia Reversa, já que poucas ferramentas a incluem.

A participação humana durante o processo, possibilita a captura do conhecimento dos especialistas, enriquecendo desta forma, o produto recuperado. A distribuição da responsabilidade desta tarefa entre os membros da equipe de manutenção, torna-a mais interessante e menos onerosa.

Os ambientes de apoio à Engenharia Reversa devem buscar a integração com ambientes de apoio à Engenharia Progressiva. Esta integração facilita a introdução de novas metodologias nos ambientes de manutenção e desenvolvimento de software.

O ARCoPAS introduz uma metodologia de Reprojeto Cooperativo de Sistemas. Da mesma maneira que o ARCoPAS permite que se registre a discussão sobre o Reprojeto de sistemas. Uma vez terminada a recuperação do Projeto de um sistema, os membros da equipe já estarão bastante familiarizados com a utilização da ferramenta ItIBIS. O Reprojeto deste sistema é uma conseqüência natural e indolor, pois esta mesma ferramenta poderá ser usada para capturar as decisões de Reprojeto.

O ARCoPAS pode ser encarado como uma tecnologia de suporte ao trabalho cooperativo. Segundo a classificação de groupwares apresentado por Olso et ali [OLS 93], o ARCoPAS é ao mesmo tempo um sistema de apoio a reuniões e de co-autoria, pois integra uma ferramenta de apoio à discussão (ItIBIS), e uma ferramenta de co-autoria(Folio Views).

No que tange a classificação de Simon para sistemas de apoio a decisão, a ferramenta não possuir um processo bem definido de escolha o que dificulta, e as vezes pode até confundir quanto a melhor solução à um problema em discussão. Entretanto, o uso do modelo IBIS em sua forma básica torna fácil a utilização, bem como a captura das discussões, pelos participantes.

3.2.6 GRADD

O Sistema GRADD – Grupo de Apoio à Discussão e Deliberação, é um SADG desenvolvido na PUCRS que combina o uso de modelos de argumentação para apoio à discussão, técnicas de votação para atividades de deliberação, e controle automático de normas para apoiar grupo em reuniões assíncronas e distribuídas. [BAC 97]

O desenvolvimento deste sistema foi baseado em duas grandes premissas:

- A objetividade é convergência dependem em grande parte de uma comunicação clara com o mínimo possível de “ruídos”, de suporte também as atividade de deliberação, bem como a de uma condução mais rígida de reunião;
- O emprego de técnicas simples dentro do processo decisório facilitam a utilização da ferramenta por um público mais amplo.

De forma a contemplar estas premissas, neste sistema, foram incorporadas as seguintes características:

- Utilização do modelo de argumentação IBIS. A simplicidade do modelo, é a chave do sucesso na sua ampla utilização, pois através dele, reuniões poderão acontecer com o mínimo de “ruído”;
- Técnicas de votação para apoio a escolha dentro do processo decisório, pela familiaridade que as pessoas tem com estas técnicas;
- Coordenação automática de vários aspectos da discussão e da deliberação pela ferramenta, através de uma norma que é definida por um facilitador.
- Interface orientada a formulários, com facilidades de navegação entre os vários subsistema que forma este SADG.

A arquitetura funcional do GRADD está dividida em três subsistemas: Subsistema de Norma, subsistema de Discussão e Votação.

O subsistema de Norma tem por finalidade propor funcionalidades que apoiem o processo decisório através de uma coordenação mais rígida da reunião, objetivando um melhor aproveitamento do processo de discussão e votação pelos usuários do SADG. Nesta primeira interação no sistema, existem dois papéis bem distintos, um facilitador (pessoa que apresenta um problema de decisão e coordena o processo decisório) e os membros do grupo (pessoas que participarão da reunião).

Na Norma são definidos os seguintes elementos:

- Participantes: cadastra o facilitador do processo, e os membros envolvidos.
- Regras para a discussão: são limites definidos dentro do processo que visam dar mais objetividade a reunião, evitando discussões intermináveis, dominância na discussão por um ou outro membro do grupo, dentre outros.
- Regras para a votação: define o método de extração de alternativas a partir da discussão efetuada, o número de turnos e suas características.
- Tema: o assunto a ser discutido e posteriormente votado.

Quando a norma detecta o tempo de início da discussão, esta é aberta pelo sistema. Os membros do grupo cadastrados na norma podem neste instante apresentar seus pontos de vista em relação ao tema proposto, respeitando as regras previamente definidas no subsistema de norma.

Como já mencionado, o nodo tema do modelo IBIS foi anteriormente preenchido pelo facilitador quando da definição da norma, pois é através dele que o problema de decisão é compartilhado para com os membros do grupo. Cada membro do grupo utiliza dos recursos dispostos neste subsistema de discussão para criar posições e argumentos relativos ao tema, os quais podem ser consultados a qualquer momento, ou a posteriori.

Após esta etapa do processo decisório é apresentado a atividade de votação que compreende duas etapas:

- Extração de alternativas de voto e

- Votação propriamente dita.

Em caso de empate, poderão ocorrer outros turnos, assim como configurados dentro da norma.

Na etapa de extração de alternativas de voto são selecionadas aquelas que farão parte do primeiro turno da votação, que corresponde às posições expressas durante a discussão. As posições que serão encaminhadas para a votação podem ser manipuladas diretamente pelo facilitador ou pelos membros do grupo.

Uma análise dos métodos de votação em [STR 80] mostrou que a maioria destes podem ser reunidos em três grandes grupos, considerando o número de alternativas que podem ser votadas por pessoa e a forma de expressão de voto (i.e. escolha ou escala de preferência). Os método se diferenciam apenas no cômputo dos votos. Dentro do subsistema de votação são utilizados os métodos de votação pluralidade, aprovação e Borda Kendall.

Detectado o inicio do primeiro turno pela norma, a votação é aberta, e cada participante pode votar na alternativas disponíveis, de acordo com a definição da norma. A norma também controla que cada participante vote uma única vez por turno. A interface fornecida pelo sistema permite navegar pela discussão, consultando detalhes das contribuições, a fim de melhor formar sua opinião. Ao término do processo, o sistema fecha a votação, realiza o cômputo dos votos de acordo com o método previamente selecionado na configuração da norma.

3.2.7 CPCE

A proposta do ambiente CPCE é de realizar modelagem de processos de software a partir de um modelo de discussão onde questões, posições e argumentos são apresentados em relação a artefatos sendo manipulados pelos participantes ou em relação aos passos definidos para a execução da atividade em andamento [LON 96]. Portanto, processo e meta-processo se confundem numa mesma estrutura de argumentação.

Processos são especializados a partir do modelo geral de processos e podem ser customizados para atender a requisitos específicos de determinado contexto de colaboração. Alterações no processo são realizadas através do lançamento de questões quanto ao meta-processo.

A proposta do ambiente está centrada nas seguintes premissas:

1. A maior parte dos processos colaborativos possui uma quantidade significativa de estruturas a serem modeladas, e suportadas através de PCEs (*Process Centered Environment*). PCEs são ambientes no quais são interpretados modelos de processo de maneira que:
    - O controle quanto a participação dos envolvidos durante as atividades que formam o processo é assegurado, monitorando estes quanto a violação das restrições previamente definidas,
    - A correção e/ou construção de um artefato é realizada a partir da execução de definições previamente configuradas para a realização de ações dentro do processo,
    - A Orientação dos participantes é provida, pela informação a estes de o que deve ser feito, o passo seguinte, pela

informação do *status* do processo em execução, pelo *start* de novas atividades, dentre outros.

2. As duas faces de colaboração precisam ser providas em ambientes PCEs
    - *Production Task* colaborativo,
    - *Meta Process* colaborativo, onde regras e procedimentos que orientam a *production task* (i.e. modelo de processo) surgem, são negociadas, modificadas e reinventadas.

Desta forma, o ambiente CPCE interpreta dois modelos de processos, como o demostrado na figura abaixo. É possível notar que não apenas a evolução dos modelos de processo é suportada, mas os mecanismos de interpretação também o são.

3. *Collective issue resolution* é a estrutura básica de processos colaborativos. Muitas abordagens incluem está estrutura como aspecto chave nas áreas de sistemas de argumentação (e.g. gIBIS [CON 88], *design rationale*, *design method modeling*, *requirement engineering process modeling*, dentre outros.

Em toda apresentação da proposta em [CPCE] está bem apresentado o significado de *collective issue resolution* e como está estrutura pode ser aplicado no contexto de processos colaborativos, seja no nível de atividade quanto no de meta processo.

Quanto ao funcionamento do ambiente, este é baseado no modelo de argumentação IBIS. Geralmente, todos os tipos de questões não podem ser levantados no início do processo: um processo bem estruturado dentro do ambiente consiste de fases bem definidas, cada fase compreendendo um subconjunto de um conjunto de questões. Normalmente, a finalização de uma fase é por ela mesma uma nova questão requerida para uma decisão coletiva. Fases, e questões dentro de fases, são executadas de maneira concorrente.

Cada participante dentro processo possui um (ou vários) "papel"(eis). Um papel define: que tipo de questões este pode levantar dentro do ambiente, quais deliberações ele pode participar e quais atividades ele pode realizar.

Todos os SADGs apresentados anteriormente possuem características bem distintas, quanto a sua aplicação final, bem como o desenho de passos (atividades) a serem realizados para suportar um processo decisório.

Na seção seguinte será apresentado um modelo muito utilizado na literatura, no contexto de processos colaborativos/cooperativos, para representar a segmentação (tipos) das atividades realizadas em grupo e a sua distribuição quanto a resolução de questões coletivas.

3.3 Modelo de McGrath

O modelo de McGrath é um modelo circumplexo de tipos de atividades de grupo, para representar os tipos de atividades de grupo, em momentos de decisão, para resolução de questões coletivas. *"um projeto é uma missão (...) Projetos consistem de*

atividades, conjunto de atividades definidas para terminar um projeto específico. Os resultados advindos das atividades possuem um valor como uma atribuição para o projetos...[GRA 84].

De acordo com modelo, existem quatro quadrantes, cada um dividido em dois tipos de atividades, como exemplificado na figura abaixo.

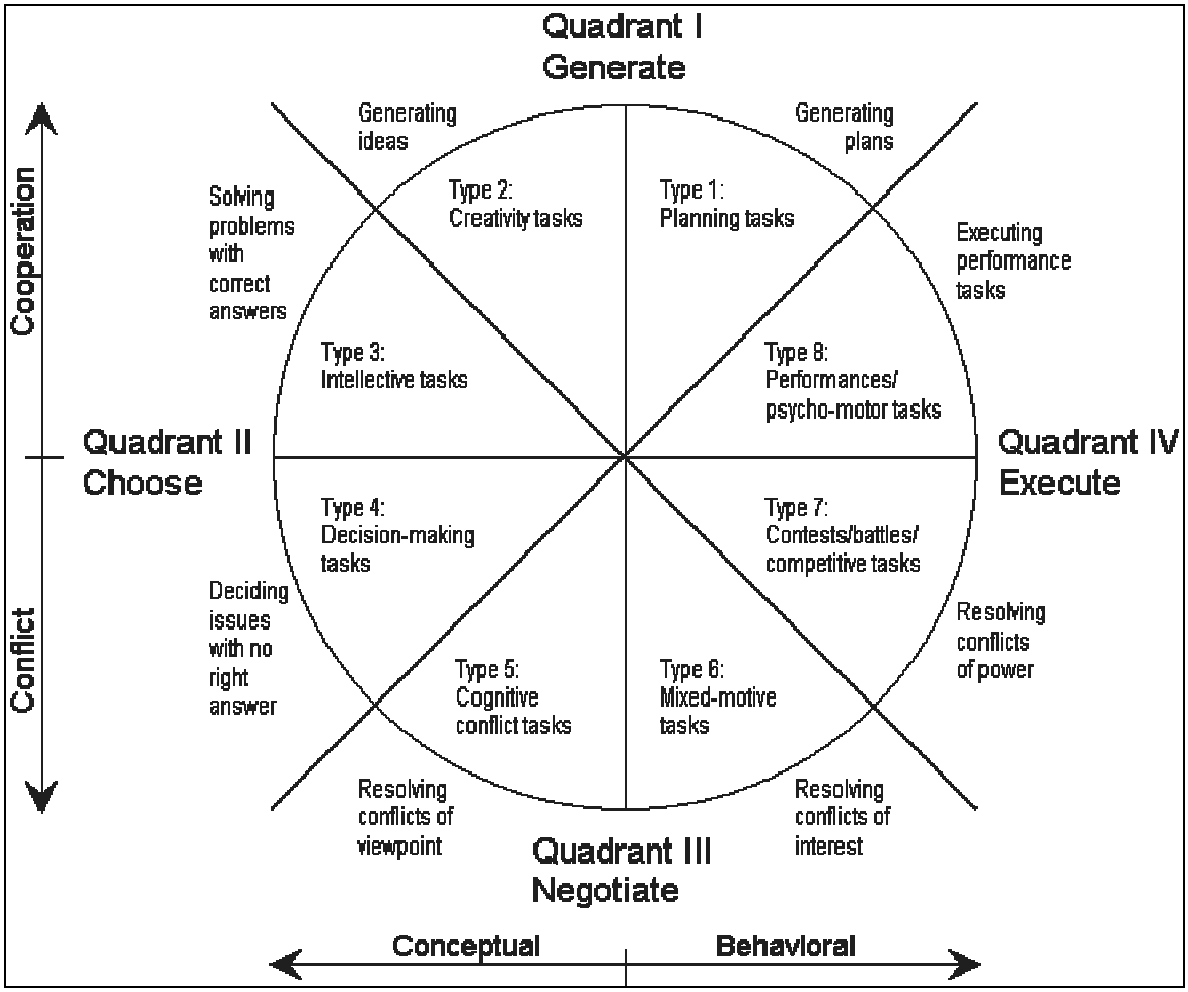


FIGURA 3.5 – Modelo Circumplexo de McGrath.

- Quadrante 1: ‘geração’.
  - Atividade 1: ‘planejamento’. Geração de planos para ação.
  - Atividade 2: ‘criativas’. Geração de idéias.
- Quadrante 2: ‘escolha’.
  - Atividade 3: ‘intelectuais’. Encontrar uma solução para um determinado problema.
  - Atividade 4: ‘preferências’. Não existe de forma definitiva uma resposta correta. A atividade do grupo consiste em alcançar selecionar, através de consenso, uma alternativa.
- Quadrante 3: ‘negociação’ (onde a escolha e feita sob conflitos).

- Atividade 5: ‘conflitos cognitivos’ . Os participantes possuem divergências quanto à preferências e estruturas.
- Atividade 6: ‘motivos mistos’. Compromisso ou acordo (negociação, barganha).

- Quadrante 4: ‘execução’
  - Atividade 7: ‘competição’. Os resultados são interpretados em termos de um vencedor e um perdedor.
  - Atividade 8: ‘coordenação’. Não requerem competição, mas a busca por padrões de excelência. Estas atividades acarretam um conjunto de atividades complexas e necessitam coordenação mais efetiva entre os membros e o tempo disposto [17] cpce

Esta forma de distribuição das atividades colaborativas propostas por Mc.Grath se contrastam entre atividades, de comportamento e conceito na dimensão horizontal, e entre cooperação e conflito na dimensão vertical.

Através da compreensão deste modelo se consegue atingir uma visão sistêmica do ponto de vista social e de cooperação, de forma que, a resolução coletiva de questões se apresenta como forma básica e importante para ambientes e cenários de tomada de decisão.

Na tabela abaixo, se utilizando do modelo apresentado é possível identificar as atividades dentro de cada quadrante e como estas estão distribuídas nas ferramentas comentadas na seção 3.2. A identificação destas atividades não demostra que uma ferramenta e inferior a outra e vice versa, apenas caracteriza a importância destas dentro do processo decisório, e como premissas básicas para proposta de resolução de questões coletivas.

TABELA 3.1 –SADGs em relação ao modelo circumplexo de McGrath.

	1º quadrante		**2º quadrante**		**3º quadrante**		**4º Quadrante**	
Atividade	planejam	criativas	intelectuais	preferen.	conflitos	Motivos	competição	coordenação
SADGs								
GIBIS	Sim	sim	Não	Não	não	Não	não	Não
VIBIS	Sim	sim	Sim	Não	não	não	sim	Não
Co-OP	Sim	não	Sim	Sim	não	sim	não	Sim
Quorum	Sim	sim	Sim	Sim	não	sim	não	Sim
Arcopas	Sim	sim	Não	Não	não	não	não	Sim
GRADD	Sim	Sim	Sim	Sim	não	sim	sim	Sim
CPCE	Sim	Sim	Sim	Não	não	sim	não	Sim

Como podemos notar na distribuição acima, não existe um modelo “ótimo”, mas em função da análise e relevância do próprio modelo de Mc.Grath, é possível identificar requisitos a serem implementados no trabalho em questão em função das atividades básicas de um processo de decisão.

No capítulo seguinte será apresentado o ambiente PROSOFT, com suas características, limitações, bem como os requisitos para inclusão de novas ferramentas, ou seja, novos ATOs.

4 AMBIENTE PROSOFT

O PROSOFT é um ambiente de desenvolvimento de software (ADS), que possui o objetivo principal de auxiliar o desenvolvimento formal de programas. Ele possibilita a definição e desenvolvimento de um conjunto de ferramentas a partir da análise de requisitos até a solução do problema [NUN 94].

Este ambiente é um projeto do Instituto de Informática da Universidade Federal do Rio Grande do Sul – UFRGS e está sob coordenação do Prof. Dr. Daltro José Nunes.

Os ADS em geral proporcionam ferramentas aos usuários para facilitar a construção de software fornecendo um ambiente computacional específico e ergonomicamente planejado para seu projeto. No PROSOFT há a determinação de satisfazer alguns requisitos, tais como:

- Interface de comunicação homem-máquina única e ergonômica, com o fim de tornar o ambiente padrão e de fácil utilização;
- Integração entre as várias ferramentas do ambiente, tanto em relação as operações como também os objetos. Por exemplo, uma determinada ferramenta utiliza os objetos resultantes de outras ferramentas, como ilustra a Fig. 4.1;
- Possibilitar ao usuário criar suas próprias ferramentas para atender melhor a seus objetivos;

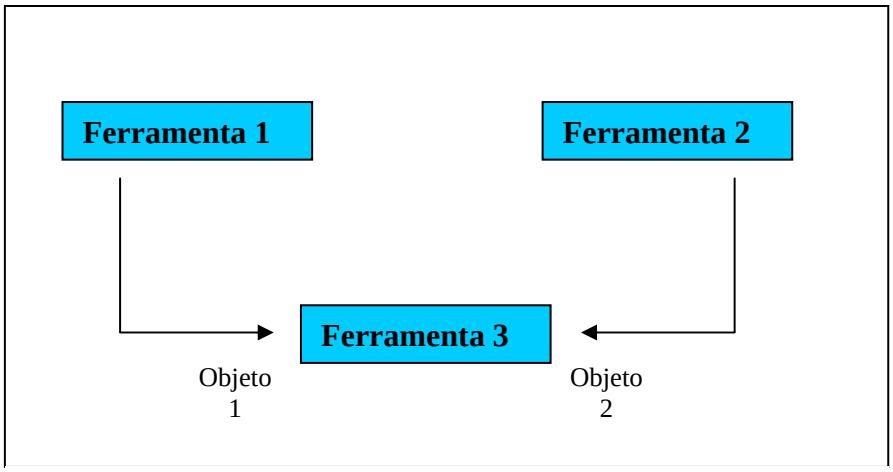


FIGURA 4.1 Integração entre as Ferramentas no Ambiente PROSOFT.

O ambiente PROSOFT é baseado nos seguintes conceitos [NUN94]:

- A estratégia “data-driven”, determina, que para encontrar a solução do problema, a definição das estruturas de dados devem ser definidas antes das operações [NUN92];
- O conceito de modelos, onde a solução de um problema é o modelo de alguma teoria, por exemplo o VDM [BJO78];
- O cálculo lambda;
- O conceito de Tipo Abstrato de Dados, onde tipos mais complexos podem ser definidos a partir da instanciação de tipos mais simples [WAT91];
- O paradigma de Orientação a Objetos, onde a reusabilidade é fortemente aplicada devido aos conceitos de troca de mensagens, de herança e de polimorfismo [COA90];
- O Método Algébrico [WAT91]

É importante salientar que a especificação algébrica, com assinatura e axiomas [WAT91], difere do formalismo usado no PROSOFT. Pode-se dizer que:

- A assinatura corresponde à instanciação e à interface do ATO e
- Os axiomas correspondem às operações (os conceitos de instanciação, interface e operações de um ATO serão vistos na próxima seção).

Essa diferença traz vantagens para o PROSOFT, como a visualização do tipo que está sendo definido através da classe apresentada graficamente e a especificação somente das operações de manipulação dos objetos da classe.

4.1 Estrutura do PROSOFT

O desenvolvimento de sistemas no ambiente PROSOFT consiste na construção de um ou mais ATOs – Ambiente de Tratamento de Objetos e na comunicação entre eles, através da ICS – Interface de Comunicação do Sistemas.

Os novos ATOs construídos podem usar ATOs já existentes no ambiente, porém como o PROSOFT é um ambiente homogêneo (fechado), ele poderá usar somente esses componentes (ATOs) desenvolvidos no seu paradigma [REI 98].

A estrutura do PROSOFT é formada pela integração (comunicação) entre os ATOs através da ICS, como ilustra a Fig. 4.2.

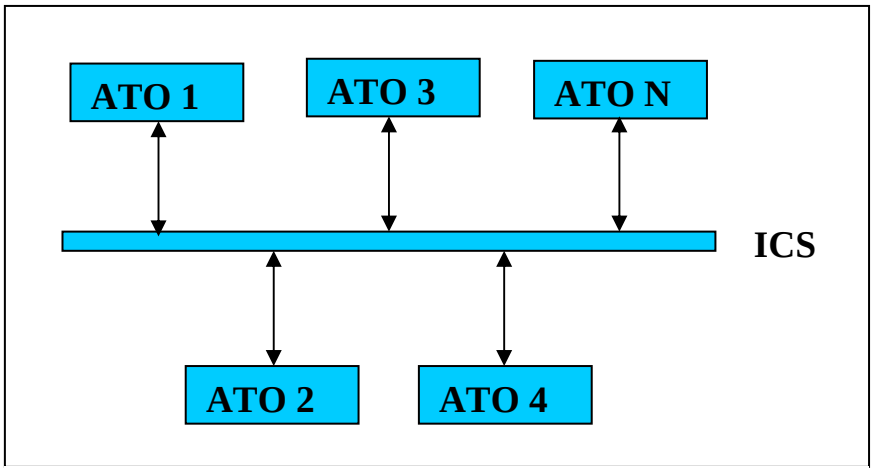


FIGURA 4.2 Estrutura do Ambiente PROSOFT.

4.1.1 Ambiente de Tratamento de Objetos

ATO – Ambiente de Tratamento de Objeto é o nome dado a qualquer sistema desenvolvido sob o paradigma PROSOFT. Um ATO PROSOFT implementa um tipo abstrato de dados e é composto por uma classe e um conjunto de operações que atuam sobre o objeto dessa classe. É importante ressaltar que o conceito de classe no PROSOFT é uma estrutura de dados que corresponde aos atributos do objeto ou variáveis de instância nas linguagens orientadas a objeto.

A estrutura de um ATO é formada por três partes, segundo [NUN94]:

- Classe – é a instanciação, representa graficamente a especificação algébrica dos tipos abstratos de dados;
- Interface – especifica as operações sobre a classe em termos de funcionalidade;
- Operações – representa a semântica das operações da Interface.

Conforme ilustrado na Fig. 4.3, cada ATO possui um nome. A interface é representada pelos nomes das operações (Op1, Op2, ...) e seus argumentos e as operações por suas especificações (semântica de cada operação).

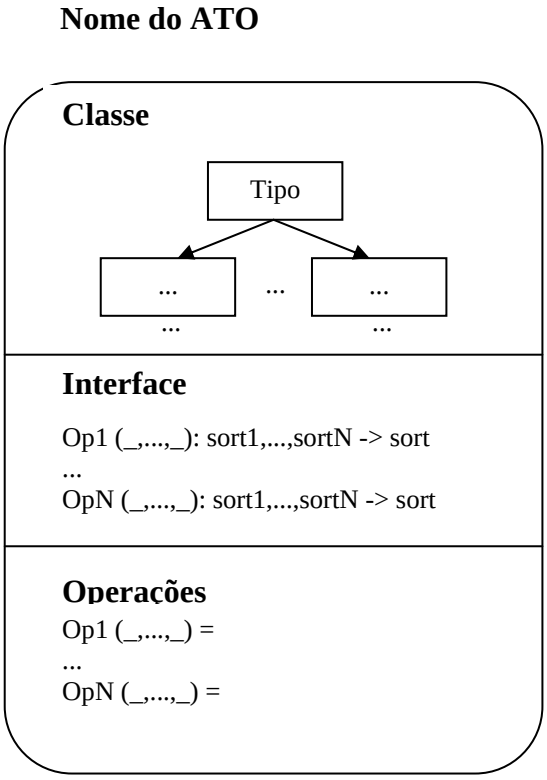


FIGURA 4.3 Estrutura do ATO.

Todo ATO possui um conjunto de objetos (instâncias da classe do ATO) e esses objetos poderão somente ser manipulados pelo ATO que o originou, pois cada ATO trata apenas termos do *sort* por ele definido. Nos casos em que um ATO necessita manipular um objeto criado por outro ATO ele envia uma mensagem, via ICS, para o ATO criador.

4.1.2 Interface de Comunicação do Sistema

A ICS é o meio de comunicação entre os ATOs, sendo responsável pelo fluxo de dados e pelo controle da execução das operações.

O formato da ICS é o mostrado na Fig. 4.4 e contém:

- Nome do ATO – é o nome do ATO da operação a ser utilizada;
- Nome da operação – é o nome da operação a ser utilizada;
- Seletor – é um objeto do tipo do ATO para qual a mensagem será enviada. Se na operação que está sendo chamada o seletor for um objeto, a ICS compara o seletor (objeto) enviado com o objeto da operação, no caso verdadeiro (se os objetos forem iguais) e os argumentos correspondentes, a operação é então executada. No caso do seletor na operação for uma

variável, a operação receberá o objeto enviado, como também os argumentos e executará a operação;

- Argumentos – lista de argumentos necessários para utilizar a operação desejada, porém esta lista não precisa ser tratada com operações referentes ao tipo de dados “Lista”.

ICS (nome do ATO, nome da operação, seletor, argumentos)

FIGURA 4.4 Estrutura da chamada ICS.

Um exemplo da utilização de uma chamada ICS de um ATO para outro é mostrado na Fig. 4.5. Como, segundo [NUN94], a pesquisa da operação é feita de cima para baixo dentro do ATO, se o nome do ATO coincidir, o nome da operação for uma operação pertencente ao ATO e o seletor for o mesmo, os argumentos serão ligados aos parâmetros formais e será executada a operação.

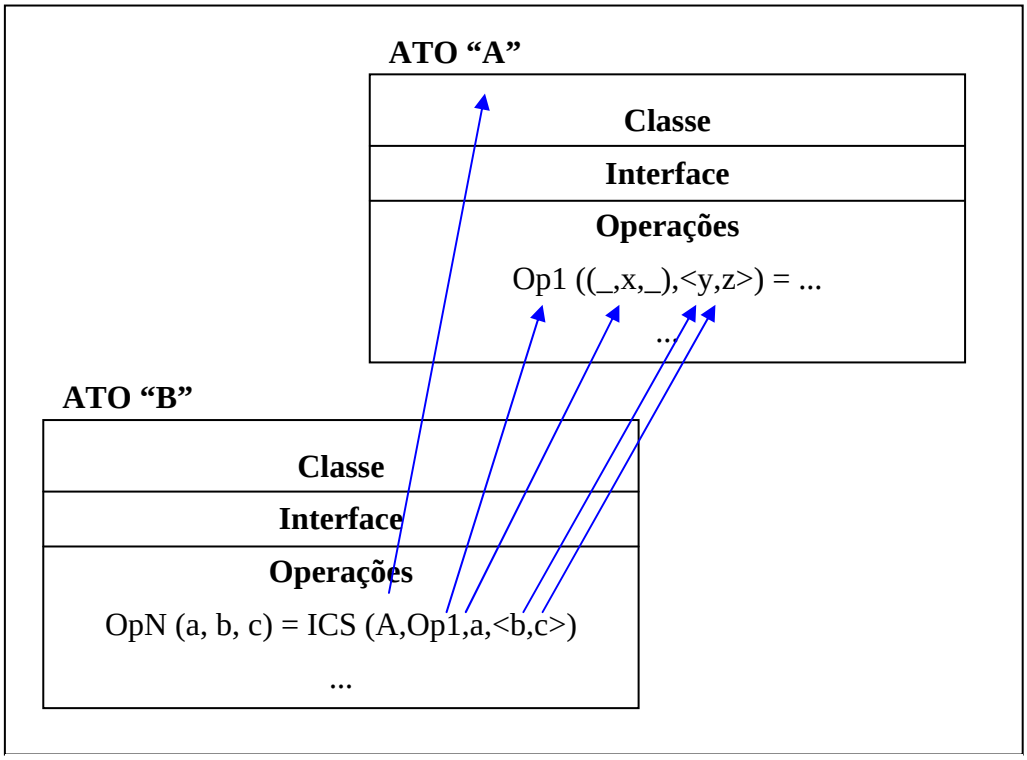


FIGURA 4.5 Exemplo de uma chamada ICS.

A operação chamada pela ICS será executada em função dos argumentos e podendo também utilizar o objeto (seletor) enviado. O objeto que foi enviado só será utilizado no lado direito da operação se na operação contiver uma variável no lugar do seletor e esta variável receber o objeto enviado. Sendo assim, como mostra a Fig. 4.6, dependendo do seletor (variável ou não) e do predicado (p, p’’), a operação executará um tipo de função (f ou f’, f’’ ou f’’’).

op (seletor, argumentos) =	**if**	p (seletor) **or** p’ (seletor, argumentos)
	then	f (argumentos) **or** f’(argumentos, seletor)
	else	f’’(argumentos) **or** f’’’ (argumentos, seletor)

FIGURA 4.6 Estrutura de uma operação no PROSOFT.

4.2 Tipos de Dados PROSOFT

Os tipos de dados utilizados no PROSOFT são classificados da seguinte forma:

- Primitivos – são os tipos derivados da linguagem hospedeira, o Pascal. São eles: *integer*, *real*, *string*, *boolean*, *char*, *time*, *date*;
- Compostos – são tipos definidos no método algébrico: conjunto, lista, mapeamento, registro e união disjuntiva;
- Definidos pelo Usuário – são tipos construídos a partir de outros tipos, e são representados pelos ATOs desenvolvidos no ambiente.

Com os tipos primitivos são triviais, a seguir serão mostrados os tipos compostos e os tipos desenvolvidos pelo usuário.

4.2.1 Tipos Compostos

Nesta seção serão apresentadas os tipos compostos, juntamente com exemplos de suas aplicações segundo sua representação diagramática, sua operação construtora e as possíveis operações utilizadas em cada tipo [REI98].

- Conjunto

Representa uma coleção de objetos, respeitando as características de conjunto, onde a ordem desses objetos não é importante.

Por exemplo, como ilustra a Fig. 4.7, uma Escola é formada por um conjunto (representado por **s**) de Sala. O tipo Sala é uma referência ao ATO Sala.

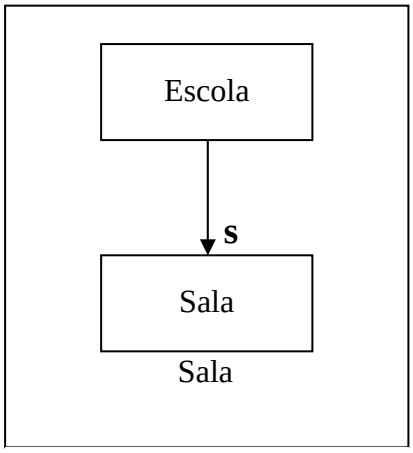


FIGURA 4.7 Classe Escola.

A operação construtora *add* e as outras operações do tipo Conjunto estão presentes na Tab. 4.1, onde contém suas funcionalidades e exemplos de aplicação.

TABELA 4.1 Operações do tipo composto Conjunto.

Operação	Funcionalidade	Exemplo
Add	Set x Component → Set	add ({x1,x2},x3) = {x1,x2,x3}
Belongs_to (∈)	Component x Set → Boolean	x1 ∈ {x1,x2} = TRUE
Cardinality (car)	Set → Nat	car ({x1,x2,x3}) = 3
Complement (\ )	Set x Set → Set	{x1,x2} \ {x2} = {x1}
Containtion (⊇)	Set x Set → Boolean	{x1,x2,x3} ⊇ {x1,x3} = TRUE
Delete	Set x Component → Set	delete ({x1,x2},x2) = {x1}
Empty-set	→ Set	empty-set = { }
Equal (=)	Set x Set → Boolean	{x1} = {x2} = FALSE
Intersection (∩)	Set x Set → Set	{x1,x2} ∩ {x2} = {x2}
Is_in	Set x Component → Boolean	is_in ({x1,x2},x1) = TRUE
Union (∪)	Set x Set → Set	{x1,x2} ∪ {x3} = {x1,x2,x3}

- Mapeamento

Representa uma função, onde ocorre a relação de um Tipo-Domínio com um Tipo-Imagem.

Por exemplo, como ilustra a Fig. 4.8, um Indivíduo pode ser representado pelo seu Nome e pelo número da sua Carteira de Identidade. Sendo que Nome é do tipo *string* e Carteira de Identidade do tipo *integer*.

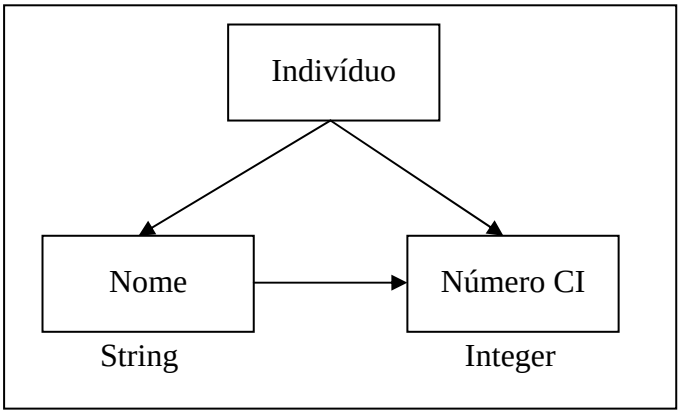


FIGURA 4.8 Classe Indivíduo.

A operação construtora *modify* e as outras operações do tipo Mapeamento estão presentes na Tab. 4.2, onde contém suas funcionalidades e exemplos de aplicação.

TABELA 4.2 Operações do tipo composto Mapeamento.

Operação	Funcionalidade	Exemplo
Composition (Θ)	Map x Map → Map	[x1 → y1] Θ [x2 → y2] = [x1 → y2]
Domain (dom)	Map → 2^{Set}	dom ([x1 → y1, x2 → y2]) = {x1,x2}
Empty-Mapping	→ Map	empty-mapping = { }
Image_of	Domain, Map → Rng	image_of (x1, [x1 → y1]) = y1
Merge (∪)	Map x Map → Map	[x1 → y1] ∪ [x2 → y2] = [x1 → y1,x2 → y2]
Modify	Domain x Rng x Map → Map	modify (x1, y1, [ ]) = [x1 → y1]
Override (+)	Map x Map → Map	[x1→y1,x2→y2]+[x2→y3]=[x1→y1,x2→y3]
Range (rng)	Map → Set	rng ([x1 → y1,x2 → y2]) = {y1, y2}
Restrict_to (|)	Map x 2^{Set} → Map	[x1 → y1,x2 → y2] | {x1} = [x1 → y1]
Restrict_with (\ )	Map x 2^{Set} → Map	[x1 → y1,x2 → y2] \ {x1} = [x2 → y2]

- Lista

Representa uma seqüência ordenada de zero ou mais componentes.

Por exemplo, como ilustra a Fig. 4.9, uma Sala é formada por uma lista (representada por *****) de Aluno. Sendo que Aluno é uma referência ao ATO Aluno.

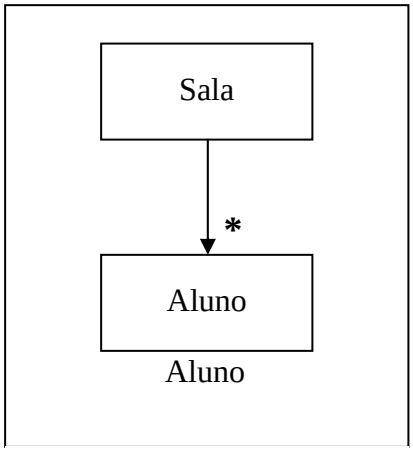


FIGURA 4.9 Classe Sala.

A operação construtora *cons* e as outras operações do tipo Lista estão presentes na Tab. 4.3, onde contém suas funcionalidades e exemplos de aplicação.

TABELA 4.3 Operações do tipo composto Lista.

Operação	Funcionalidade	Exemplo
Concatenation(^)	List x List → List	<x1, x2> ^ <x1> = <x1, x2, x1>
Cons	Component x List → List	cons (x1, <>) = <x1>
Elements (elems)	List → 2^{Set}	elems (<x1, x2, x1>) = {x1, x2}
Empty-list	→ List	empty-list = < >
Head (hd)	List → Component	hd (<x1, x2>) = x1
Index (inds)	List → 2^{Nat}	inds (<x1, x2, x1>) = {1, 2, 3}
Last	List → Component	last (<x1, x2, x1>) = x3
length (lng)	List → Nat	lng (<x1, x2, x1>) = 3
Projection (_,[ ])	List x Nat → Component	<x1, x2> [2] = x2
Replace (_,+,[_])	List x Nat x Component → List	<x1, x2> + [2, x3] = <x1, x3>
Tail (tl)	List → List	tl (<x1, x2>) = x2

- Registro

Define tuplas de tipos heterogêneos, expressando assim um produto cartesiano.

Por exemplo, como ilustra a Fig. 4.10, um Livro possui Nome, Autor, Editora, Ano e Número de Páginas. Sendo que Nome, Autor e Editora são do tipo *string* e Ano e Número de Páginas do tipo *integer*.

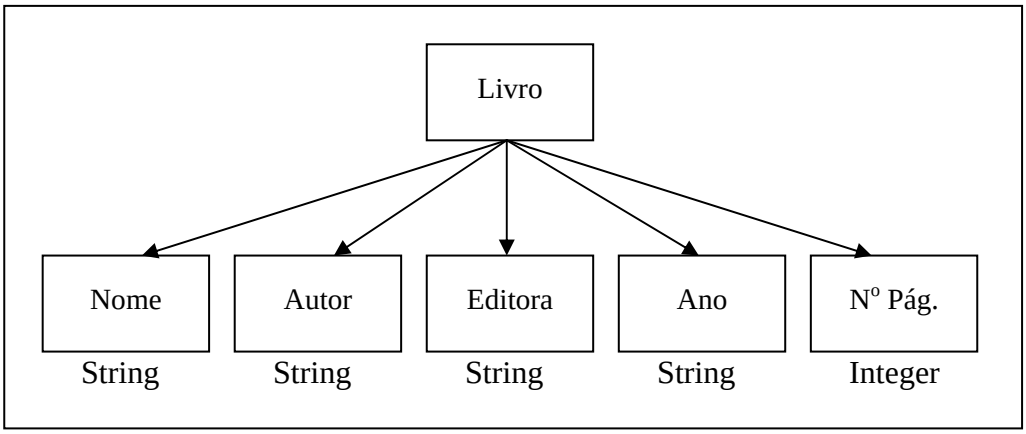


FIGURA 4.10 Classe Livro.

O tipo Registro possui uma operação construtora do tipo (_,_,...,_) e outra operação do tipo observadora, *select* (seleciona um dos campos do registro). Essas operações estão presentes na Tab. 4.4, onde contém suas funcionalidades e exemplos de aplicação. Para facilitar a seleção (e identificação) dos campos dos registros, os tipos dos seus campos são precedidos por uma *tag* (rótulo sublinhado).

TABELA 4.4 Operações do tipo composto Registro.

Operação e Funcionalidade	Exemplo
(_,_,...,_): C_1, C_2, ..., $C_n \rightarrow$ Record	(Rua rua, Bairro bairro, Cid cidade, Est estado, Cep cep)
Select-_: Tag Record $\rightarrow$ S_1 S_2 ... S_n	select-Bairro (_, Bairro Centro, _,_,_) = Centro

- União Disjuntiva

Define a união de tipos diferentes de elementos, sendo útil quando uma classe de dados necessita expressar valores alternativos.

Por exemplo, como ilustra a Fig. 4.11, um Professor pode ser Titular, Contratado ou Assistente. Cada tipo de Professor é do tipo *string*.

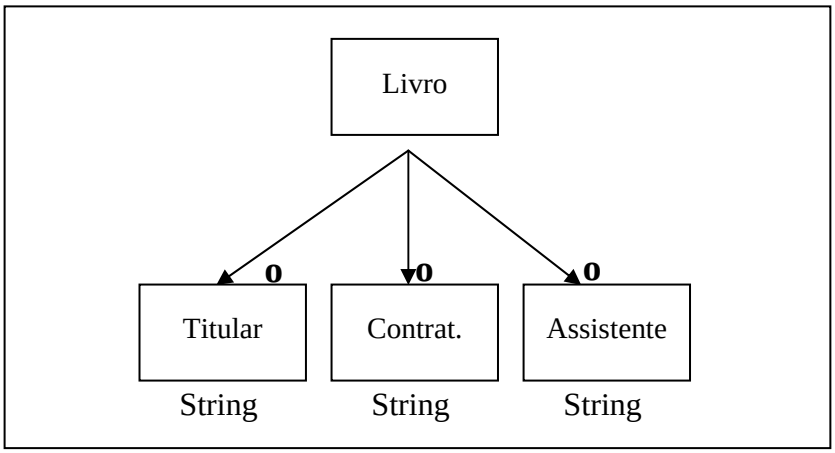


FIGURA 4.11 Classe Professor.

A construtora do tipo União Disjuntiva é o sinal de + e possui a seguinte funcionalidade: + (_,_,...,_): Tag_1 $Sort_1$, Tag_2 $Sort_2$, ..., Tag_N $Sort_N$ → Union

Um exemplo seria:

+ (titular Titular, contratado Contratado, assistente Assistente)

4.2.2 Tipos Definidos pelo Usuário

Os tipos definidos pelo usuário são os ATOs existentes no PROSOFT. Sendo que estes ATOs já especificados podem ser usados na criação de novos ATOs.

O ATO-ATO [RIB 91], permite incluir um novo ATO no ambiente a partir de sua especificação algébrica, eliminando assim a necessidade de recompilação de todo o ambiente.

Um ATO já implementado no PROSOFT que é muito importante é o ATO Diretório. Ele é responsável pelo armazenamento físico dos objetos do ambiente, fornecendo um conjunto de rotinas de tratamento de arquivos [CAP92].

Entre as ferramentas já implementadas, estão as de interface com o usuário:

- ATO Cenário – permite a definição de uma janela no PROSOFT, sendo a principal responsável pela interação do usuário com o sistema;
- ATO Quadro – auxilia na definição de uma quadro (janela de interface) dentro de um cenário (ambiente);
- ATO Menu – permite a construção de menus em quadros;
- ATO Opção – define as opções de um menu;
- ATO Texto – auxilia na edição de textos;
- ATO Edgraf – editor que auxilia na manipulação de objetos gráficos;
- ATO Figura – permite a definição de objetos gráficos;
- ATO Coord – define e localiza uma coordenada de tela.

Outros ATOs foram implementados para auxiliar no desenvolvimento de sistemas, tais como o ATO NSD (para especificação de diagramas Nassi-Schneidermann [NAS 73], o ATO SADT [ROS 85] e o Editor Larch (para especificação em Larch).

Há também a definição de um ambiente (*shell*) para construção de sistemas especialistas denominado Expert [MOR 97], um ambiente para trabalho cooperativo [REI 98] e um simulador de processos de software [SIL 01].

4.3 Gerenciador de Processos

O gerenciador de processos (GP) [LIM 98] é uma parte fundamental para o trabalho presente, por isso a ênfase maior para este tema.

O gerenciador permite coordenação de desenvolvedores, grupos de desenvolvedores e gerentes em um ambiente cooperativo é distribuído. Além disso, permite a coleta de métricas dos projetos sendo executados e reutilização de processos definidos.

A arquitetura do PROSOFT, apresentada na Fig. 4.12 mostra que o gerenciamento de processo de software foi implementado sobre os serviços do PROSOFT Cooperativo. O GP age como super-usuário do PROSOFT Cooperativo, possuindo controle sobre o acesso aos artefatos de software produzidos durante um projeto.

Os componentes principais do GP são agentes, cargos, recursos, processos, projetos e o Prosoft Cooperativo. Através da integração destes componentes foi especificado um mecanismo de execução de processos que interage com as ferramentas do ambiente Prosoft e com os usuários do ambiente. Um processo de desenvolvimento é definido como sendo um conjunto de atividades. As operações sobre um processo são refletidas diretamente nas suas atividades. As atividades de um processo podem ter sub-atividades. Entretanto, todas as atividades, em qualquer nível de granulosidade possuem as mesmas características básicas.

O GP adota como modelo de interação com os desenvolvedores a agenda de atividades. Cada desenvolvedor interage com o sistema através da agenda, aonde estão listadas as atividades em que ele está envolvido, o prazo determinado para completá-las, os profissionais que estão envolvidos na atividade (se for uma atividade de trabalho cooperativo) e uma descrição textual identificando quais tarefas que devem ser realizadas para a atividade (para cada um projetos nos quais o desenvolvedor está envolvido). Ao gerente, por sua vez, estão disponíveis operações para acompanhar a execução dos projetos da organização (visualizando, por exemplo, o andamento das atividades e a utilização dos recursos do ambiente) e a movimentação de objetos (como entrada e saída de atividades).

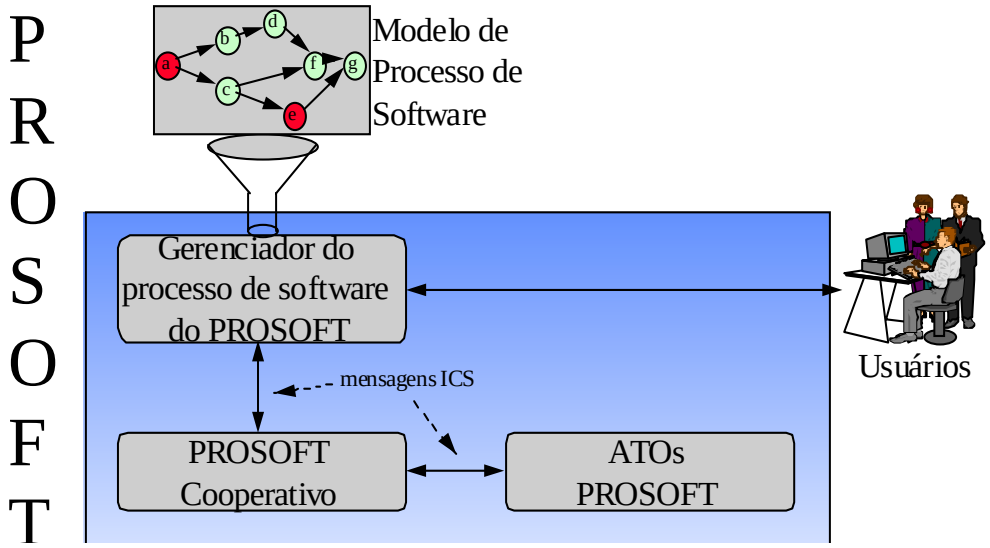


FIGURA 4.12 Integração do Prosoft Cooperativo e GP [LIM98].

Esses conceitos apresentados sobre o GP são importantes pois no capítulo 6 conterá a construção de novos ATOs para o ambiente PROSOFT (no qual foi baseado este trabalho), como também o funcionamento e comportamento dos objetos.

O ambiente vem se mantendo atualizado com as últimas tendências tecnológicas, sendo um dos precursores na utilização do paradigma de objetos no desenvolvimento de software, por apoiar a estratégia data-driven [NUN92] mesmo estando hospedado, na época, sob uma linguagem de programação que não apoiava tais construções (Pascal). Adicionalmente, a mudança da linguagem hospedeira do sistema (de Pascal para Java) [SCH97], realizada durante os anos de 1997-98, foi uma arrojada experiência prática da reengenharia de um sistema complexo que envolveu diversos estudantes e pesquisadores da UFRGS e Uni-Stuttgart. Como resultado, o ambiente implementa, hoje, diferentes ferramentas que cooperam no desenvolvimento de software para diferentes plataformas de hardware e software, beneficiando-se da portabilidade fornecida pela linguagem de programação adotada.

O próximo capítulo apresentará de maneira informal a proposta de modelo SADG para o ambiente PROSOFT.

5 Modelo de SADG proposto para o PROSOFT

O processo de desenvolvimento de software implica na necessidade constante de tomadas de decisão. A cada etapa do processo, torna-se necessário estabelecer a comunicação e interação entre usuários, gerentes, analistas, programadores e mantenedores numa constante troca de informações.

O registro dos artefatos produzidos durante todo o processo é uma questão que norteia as pesquisas em ambiente de desenvolvimento de software. Quando se fala em suporte ao processo de colaboração entre os elementos de uma equipe de desenvolvimento, este registro torna-se ainda mais necessário. Neste contexto, a modelagem dos dados a serem armazenados se amplia para comportar outras informações provenientes da interação do grupo além dos artefatos gerados[LUC 91][POT 88][YAK 90]. As informações trocadas durante este processo interativo que incluem fatos, hipóteses, restrições, decisões e suas razões, o significado de conceitos e, os documentos formais formam o que é denominado pela literatura especializada como memória de grupo.

A proposta do modelo SaDg PROSOFT, como exemplificado na Fig 5.1 abaixo, visa fornecer suporte a memória de grupo, no que diz respeito ao registro das justificativas de projeto(*Design Rationale*) através de uma integração com o Gerenciador de Processos (GP) disponível no ADS PROSOFT.

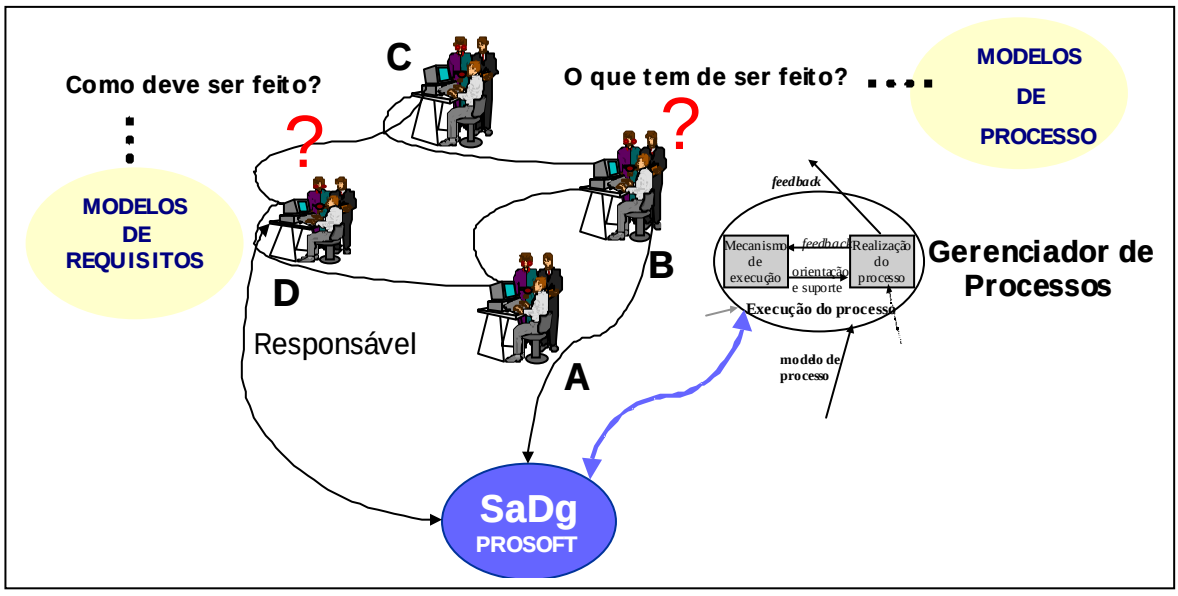


FIGURA 5.1 – Cenário de Utilização do modelo SaDg PROSOFT.

Esta integração se dá através das ferramentas inseridas na arquitetura, assim desenhadas: Editor de Problemas de Decisão, Editor de Normas, Editor de Argumentação, Extrator de Alternativas e Editor de Votação.

Qualquer processo de decisão dentro do SaDg PROSOFT só pode ser modelado por facilitadores, que são usuários do SaDg definidos pelos gerentes de desenvolvimento configurados dentro do GP. Os gerentes desempenham um papel importante no GP, modelando e a acompanhando a execução dos processos.

O facilitador inicia o processo de decisão desenhando-o com todas as suas características através do Editor de Problemas e do Editor de Normas. O Editor de Problemas contém todos os problemas a serem solucionados dentro do ambiente. O Editor de Norma é composto por um conjunto de formulários que configuram todas as atividades necessárias dentro de um processo decisório.

Um processo decisório dentro do SaDg PROSOFT está definido nas seguintes atividades: Atividade de argumentação (suportada pelo Editor de Argumentação), atividade de Extração (suportada pelo Extrator de alternativas) e por fim, a atividade de Votação (suportada pelo Editor de Votações).

No momento em que o facilitador termina de configurar a Norma, está definido um processo de decisão. O SaDg instanciará um processo de software, a partir do processo de decisão previamente registrado dentro da Norma, com todas as características de um processo decisório. Daí, o processo pode ser submetido ao GP para coordenação das atividades.

A partir deste ponto o GP começa a coordenar todo o processo, acionando as agendas dos agentes para as atividades definidas, bem como controlando a utilização das ferramentas dispostas no SaDg para a execução destas atividades.

Neste capítulo será apresentado o modelo de SADG proposto para o ambiente PROSOFT, suas características, bem como as atividades necessárias para execução de um processo decisório.

As setas de duplo sentido na Fig 5.2 abaixo indicam uma interação bilateral. Já as definidas com um sentido apenas, indicam interação unilateral. As linhas pontilhadas que ligam o processo de decisão para as ferramentas, indicam quais destas devem ser utilizadas dentro de cada atividade definida dentro de um processo decisório.

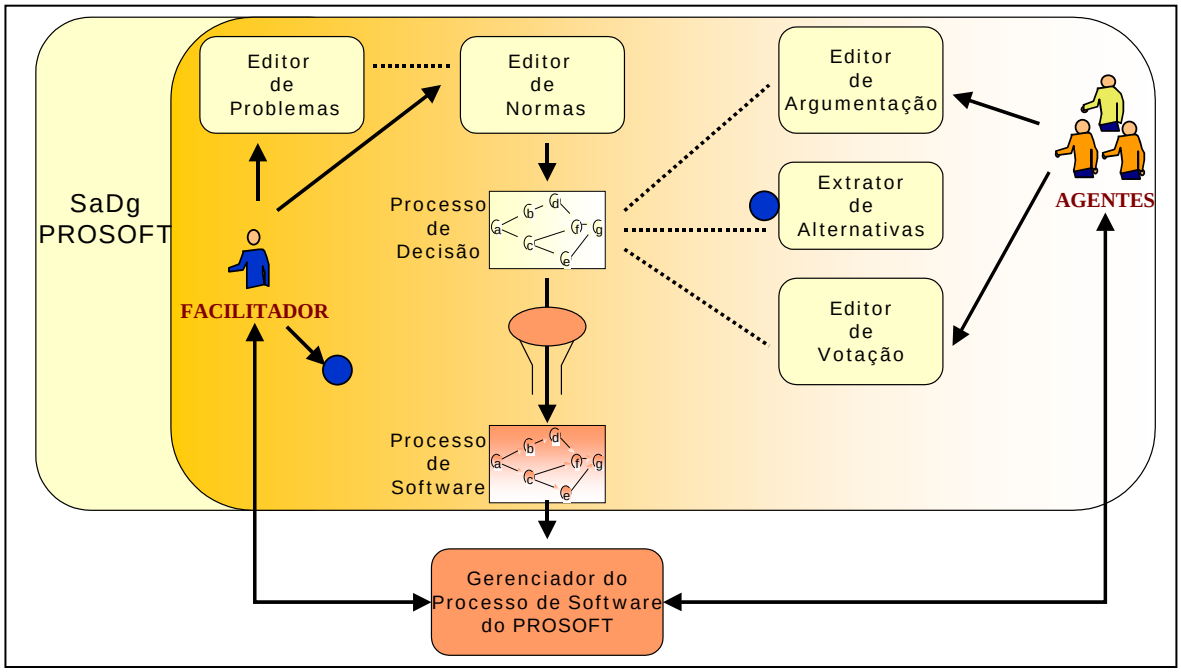


FIGURA 5.2 - Modelo SaDg PROSOFT.

Existem aspectos importantes a serem ressaltados sobre o modelo proposto. Por exemplo, é necessário que exista algum mecanismo para controlar e facilitar a comunicação dentro modelo, ou seja, uma condução mais rígida e com o mínimo de "ruído" durante todo processo.

Um outro aspecto é a uso de técnicas simples durante o processo decisório que facilitem o uso do modelo proposto em outros ambientes, não restringindo o mesmo apenas para processos de software. Entretanto, seriam necessárias modificações neste modelo de forma a enriquecer ainda mais o processo decisório em um novo contexto. Portanto, como primeira premissa, optou-se por descartar técnicas complexas de estruturação do processo decisório(e.g. DRH, MAH, AHP, MDMC), para flexibilizar a utilização do modelo por um público mais amplo.

Numa próxima seção será apresentado uma forma de utilização em outros contextos e quais seriam as modificações a serem realizadas para otimização do modelo.

A partir das funcionalidades comentadas acima, o presente modelo oferece suporte para:

- **Coordenação** - onde é possível definir um processo de decisão, através de uma norma, problemas de decisão, facilitador e o próprio gerenciador de processos GP .
- **Argumentação** – oferece apoio à discussão pelo grupo, estendendo o modelo IBIS com informações do gerenciador de processos[LIM 98].
- **Decisão** – possui técnicas de votação que auxiliam no processo de escolha.
- **Compartilhamento de Informações** – A base de informações geradas

durante todo o processo, é utilizada como memória da decisão.

5.1 Suporte à Coordenação

O suporte à coordenação tem como principal objetivo, acrescentar funcionalidades ao SaDg PROSOFT de forma a direcionar melhor processo, através de um documento de Norma e um Facilitador, proporcionando um melhor resultado do processo decisório como um todo. Fica bem claro, neste momento do processo que a interação entre pessoas com papéis diferentes e, entre estes e o facilitador, e delimitado no instante em que é criada uma Norma para o processo decisório.

A coordenação das atividades a serem executadas pelos participantes do processo decisório é solucionada pela distribuição de papéis e pela divisão do processo em atividades de decisão que serão posteriormente interpretadas e executadas pelos gerenciador de processos do ADS PROSOFT, oferecendo um suporte adicional à coordenação da atividades.

A Norma possui todas as regras e restrições para o andamento do processo, bem como os problemas de decisão ou temas, e as atividades pertencentes a cada tema de decisão. No ambiente PROSOFT, a Norma passa a ser um novo processo executável, contendo todas as informações sobre as atividades pertinentes ao processo decisão. Um cenário desta etapa do processo decisório pode ser visualizado da figura abaixo:

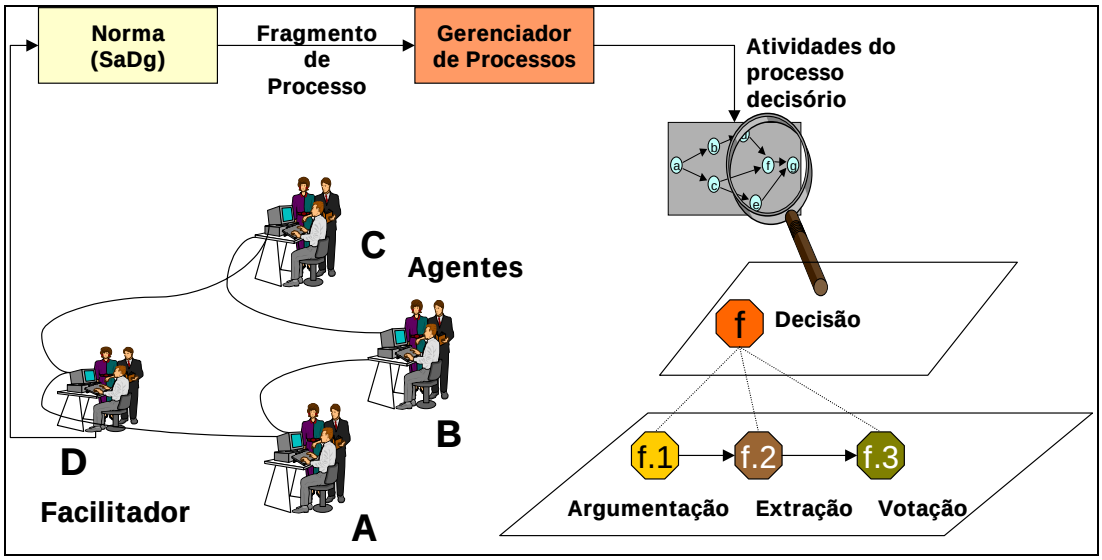


FIGURA 5.3 – Decomposição das atividades de um processo decisório.

A Norma deve ser manipulada somente pelo facilitador o qual deve ter conhecimento do processo de decisão, configurando-o com informações e regras para as atividades. A Fig 5.4 mostra os requisitos mínimos de uma Norma.

Um Norma é composta de:

- **Facilitador**: responsável por um processo decisório dentro do SaDg PROSOFT;
- **Agentes**: são usuários do ambiente, como projetistas, desenvolvedores, etc.;
- **Regras de Argumentação:** contém informações a cerca da atividade de argumentação ou discussão dentro do processo de decisão;
- **Regras de Extração:** contém informações a cerca da atividade de extração

dentro do processo de decisão;

- **Regras de Votação:** contém informações a cerca da atividade de votação;
- **Descrição:** um texto com observações sobre a Norma;
- **NomeProjGP:** contém a identificação do projeto referente ao GP;
- **IDPD:** contém a identificação do problema de decisão ao qual a Norma se refere.

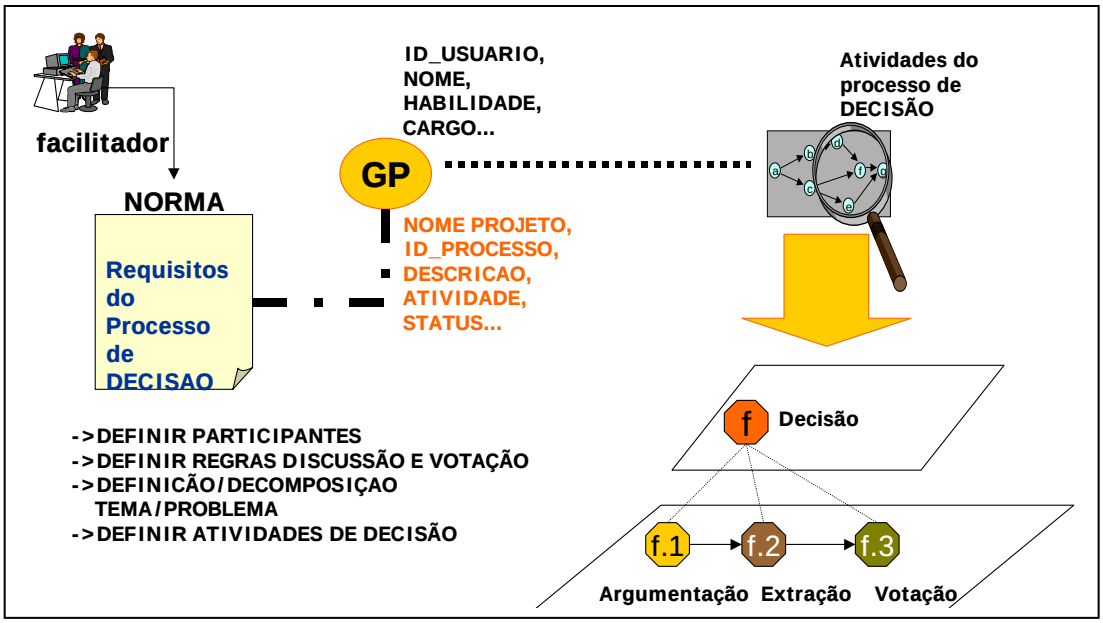


FIGURA 5.4 – Requisitos mínimos de uma Norma.

Um processo de decisão demanda por várias atividades. A primeira delas corresponde a definição de um problema de decisão, ou seja, onde se quer chegar, qual o problema a ser discutido. A definição do problema de decisão é importante para evitar que a atividade de argumentação caminhe por questões irrelevantes, mantendo os participantes nos objetivos agendados para execução da atividade, previamente configurados na Norma pelo facilitador.

No modelo de SADG proposto, um problema de decisão possui ligações para outros problemas de decisão. Um problema de decisão possui uma descrição detalhada do problema a ser discutido, datas de início, prazo e término e a prioridade dentro da atividade de argumentação, ou seja, sua importância dentro do processo de decisão, levando em consideração seus efeitos sobre o processo de desenvolvimento de software.

A divisão do processo em atividades facilitam o gerenciamento pelo GP, oferecendo mais um apoio à coordenação do processo, auxiliando também os participantes(agentes do ambiente PROSOFT) através de suas agendas, monitorando-as, sendo notificados das atividades de decisão já terminadas, as atividades sendo realizadas e as que se encontram pendentes.

O controle da transição de estados das atividades é automatizado, não sendo permitida manipulação por qualquer usuário do ambiente. Conforme apresentado em [LIM 98], a Fig. 5.5 demonstra a transição de estados de uma atividade de processo.

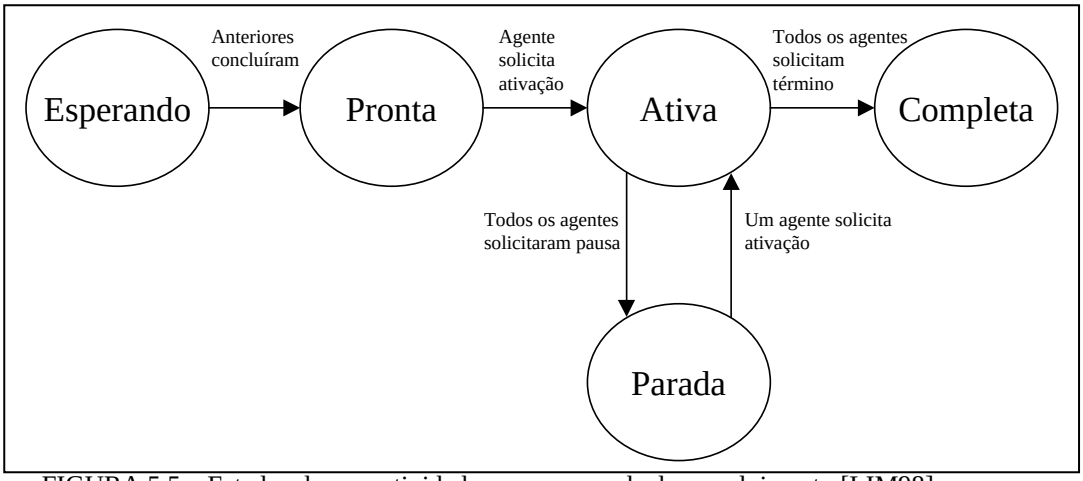


FIGURA 5.5 – Estados de uma atividade no processo de desenvolvimento [LIM98].

Na seção seguinte, serão apresentados os papéis desenhados pelos participantes do processo de decisão, bem como a configuração de uma Norma (ou modelo de processo decisório).

5.1.1 Papéis ou Responsabilidades

Cada participante é identificado dentro do processo decisório pelas suas informações oriundas do GP e pelo papel assumido durante as atividades de decisão.

Dois papéis são bem definidos dentro do modelo de SADG proposto, que são facilitador e agentes(agentes de desenvolvimento, definição GP-PROSOFT). Todo processo decisório representado pelo SADG possui somente um facilitador, e , o número de agentes deve ser suficientes para garantir a execução, para cada atividade de decisão.

A divisão de papéis é realizada na configuração da Norma, pelo facilitador do processo.

Existem responsabilidades que são atribuídas tanto para o facilitador quanto para os membros(agentes) participantes do processo decisório.

O facilitador é o participante que identifica algum problema de decisão que necessita ser discutido no ambiente. Este participante está encarregado de definir as regras para a condução do processo decisório. Assim cabe a este, configurar uma Norma com os participantes(agentes) do SADG, os controles para a discussão, processo de extração das alternativas de voto, votação, além de sua própria participação.

O facilitador também pode alterar as regras de discussão/votação definidas na norma, uma vez iniciado o processo, caso este estime que sejam benéficas para o processo. Por exemplo, aumentando ou diminuindo o tempo de discussão, o limite de intervenções por participantes, etc.

Outro tipo de papel definido é o de agente. Estes agentes são convidados a participar do processo decisório através da Norma definida pelo facilitador. Todos os agentes do processo possuem, os mesmos direitos de participação, ou seja, o mesmo tempo para discutir, votar, os mesmos limites para exposição de informações sobre o problema apresentado.

Os agentes poderão gerar novos problemas de decisão. Isto pode acontecer, por exemplo, quando uma apresentação de posições sobre um tema, dentro da atividade de argumentação, motiva um membro do grupo a discutir um novo problema. Todavia, será iniciado uma nova configuração - uma Norma - para o problema em questão.

Assim como no processo de argumentação, os agentes deverão participar do processo de votação. Neste processo, eles irão dar seus votos nas alternativas eleitas pelo facilitador no processo de extração. Da mesma forma como existe um tempo para que os membros consigam expor seus argumentos, existe um tempo para votar nas alternativas.

Caso exista um empate, os membros poderão participar do turno de desempate, obedecendo as regras de votação definidas na Norma.

Alguns fatores podem interferir na participação dos membros de grupo durante o processo decisório:

- Os agentes perdem os prazos para apresentar seus argumentos; por exemplo, por falha de comunicação na configuração da agenda do agente para participação na atividade de discussão, por esquecimento, etc.
- Alguns agentes poderão não participar na etapa de argumentação, ficando apenas como observadores do processo, estudando as alternativas que julgarem mais adequadas para voto;
- As alternativas eleitas para voto não fazem parte do conjunto de alternativas de voto esperadas pelo membro , e assim, este passa a anular seu voto.
- Os agentes não estão de acordo com nenhuma das alternativas de voto.

5.1.2 Norma e Regras para o Processo Decisório

A Norma ou Regras permitem uma melhor condução do processo decisório. A norma é definida segundo características do processo de decisão e do próprio gerenciador de processos. A definição do processo decisório deve estar de acordo com os requisitos mínimos definidos pelo Gerenciador de Processos [LIM 98].

Na Norma são definidos os agentes e o Facilitador que participarão de todo o processo. Através da configuração da Norma, o gerenciador de processos interpreta as atividades que precisam ser realizadas para execução do processo de decisão, e é neste momento que são acionadas as agendas dos agentes, informando-os do processo e as características de cada atividade.

A agenda do agente funciona como uma lista de tarefas a fazer e consiste do principal meio de comunicação entre o gerenciador de processos e o agente. O mecanismo de execução tem o poder de adicionar atividades a fazer nas agendas dos agentes e gerenciar os estados das mesmas.

O conceito de atividade na agenda do agente é diferente do conceito de atividade de um processo de desenvolvimento de software. Na agenda, a atividade representa uma atividade do processo, guardando informações sobre o andamento do ponto de vista do agente. Enquanto que a atividade do processo guarda a descrição do que deve ser feito, informações sobre todos os agentes envolvidos e quais os prazos da atividade, dentre outras informações.

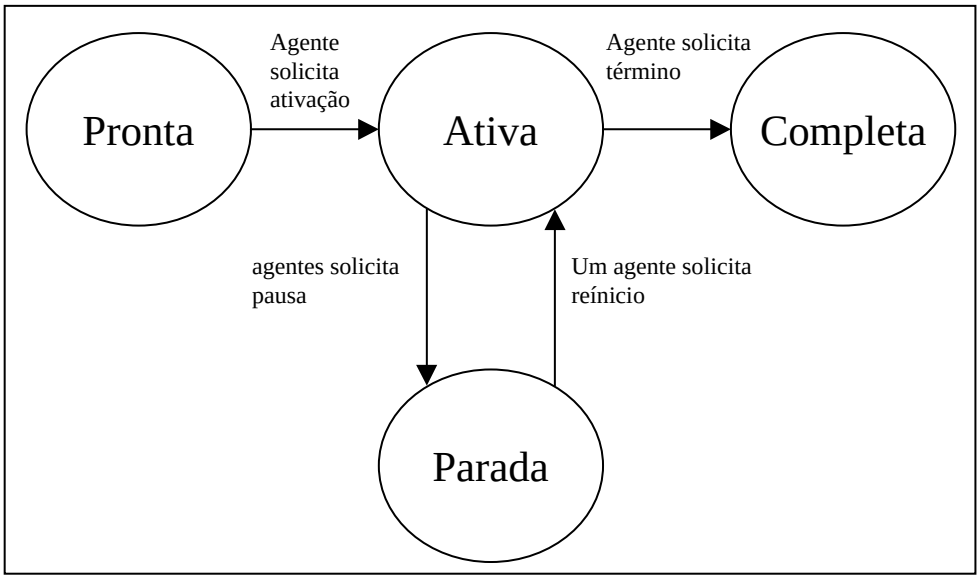


FIGURA 5.6 – Estados de uma atividade na agenda do agente [LIM98]

Após informados de suas atividades os agentes aguardam o inicio da atividade de argumentação para apresentarem suas contribuições ao problema de decisão.

Um fator importante em relação a definição da norma e escolha dos participantes, é que seria necessário uma preparação para antes do processo, para preparar os participantes, de forma a permitir um refinamento do problema a ser discutido, alteração dos limites, alteração dos métodos de escolha, entre outros. Entretanto, optou-se por não suportar estas características neste modelo.

Reconhecendo estas necessidades como um aspecto vital e importante para o processo decisório, o apoio ao mesmo não foi considerado pelos complexos aspectos de negociação e comunicação que podem envolver. O SADG proposto também não permite a realização de enquetes para permitir com que os participantes possam discutir quanto a definição da norma, dos próprios participantes, do problema e do objetivo do processo decisório. Entretanto, é possível a realização de troca de mensagens através do PROSOFT cooperativo [REI 98]

5.1.3 Regras para Argumentação

Os componentes das regras definidas para discussão são: data de inicio, data de término, número máximo de questões, número máximo de posições e número máximo de argumentos.

Os dois primeiros componentes, relativos ao período de argumentação, têm por objetivo organizar este processo de forma que, tanto o facilitador quanto os agentes do processo atentem ao período do processo onde poderão contribuir com suas idéias ao problema de decisão. Quanto a definição do período necessário para realização da atividade, caberá ao facilitador julgar o período necessário a fim de que os agentes

tenham tempo suficiente para apresentar suas idéias ao problema. Se não existisse este tipo de controle para a discussão, os agentes poderiam ficar discutindo durante um longo período, o que acarretaria desperdício de tempo e custos do projeto de software, prejudicando assim a tomada de decisão bem como, o processo de software.

Os outros componentes foram determinados com a finalidade de tornar a discussão mais objetiva e evitar a dominância na exposição de idéias. Desta forma, os agentes evitarão situações do tipo criar posições e argumentos semelhantes à outros já existentes, que não apresentem novas contribuições e que sejam pouco aproveitáveis durante o processo de discussão. Além de auxiliar na compreensão da discussão do tema sugerido, permitindo os participantes do processo tenham uma visão do todo.

Ao facilitador é atribuída a responsabilidade de limitar tais recursos durante a atividade de argumentação, nem extrapolando, nem restringindo a participação dos agentes, e também alterar os limites definidos na norma sempre que estimar que uma melhoria do processo de discussão pode ser alcançada através destas alterações.

Após a configuração das regras para atividade de argumentação, é possível definir as regras para a atividade de extração, que corresponde a seleção, feita pelo facilitador, a partir das posições que receberam pelo menos uma aprovação. A ação seguinte dentro do processo corresponde a etapa de votação dentro do processo. Esta etapa também possui uma configuração prévia com os requisitos mínimo a serem seguidos.

A Fig 5.7 mostra a configuração da atividade de argumentação e o fluxo das ações coordenadas pelo gerenciador de processos até estruturação do problema através do suporte a argumentação do modelo proposto.

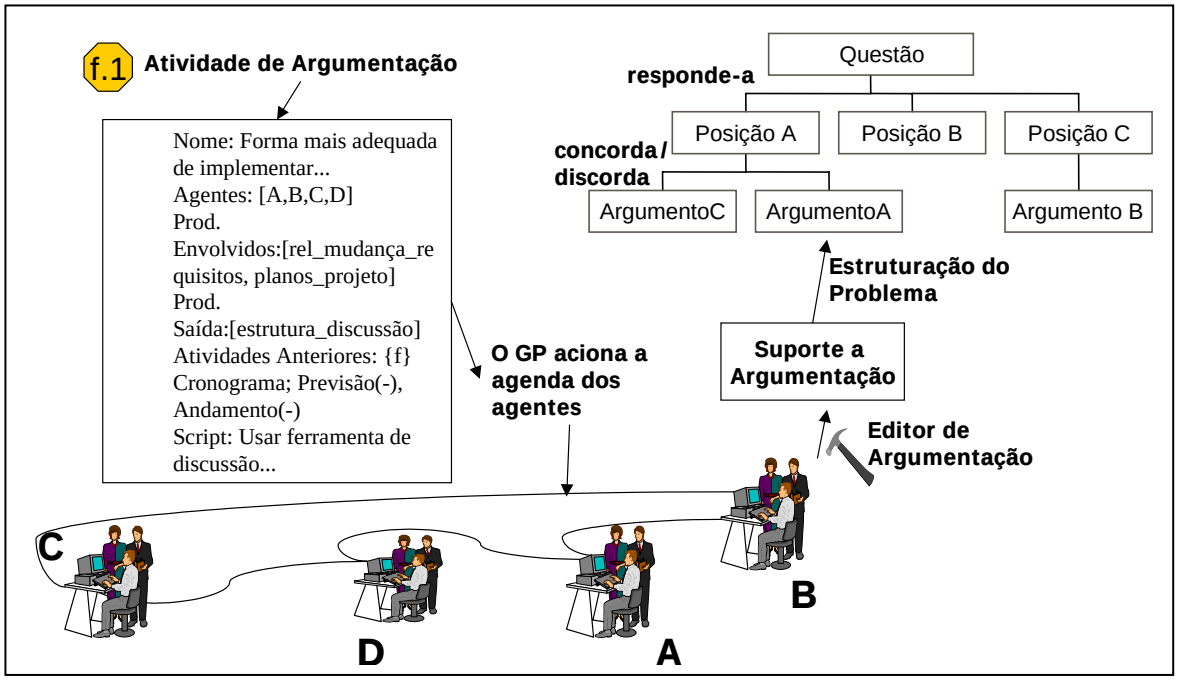


FIGURA 5.7 – Atividade de Argumentação na Agenda dos Agentes

5.1.4 Regras para Votação

Na norma também são definidas alguns componentes para controle da atividade de votação que devem ser determinadas pelo facilitador, sendo elas: extração de alternativas, número de turnos e procedimentos de desempate. Para cada turno são definidos, data de inicio e término , e o método de votação.

A extração de alternativas tem por objetivo determinar, dentre as alternativas apresentadas na discussão, aquelas que serão alternativas de voto durante a atividade de votação. Na realidade, antes mesmo de se iniciar o processo de votação e necessária a execução da atividade de extração das alternativas pelo facilitador, de forma a restringir o conjunto de alternativas de voto, na escolha por parte dos agentes. Acredita-se que quanto maior o conjunto de alternativas, menos excludentes elas se tornam, prejudicando a qualidade da escolha realizada. A Fig 5.8 abaixo mostra a extração de alternativas para votação pelo facilitador D a partir das posições apresentadas durante a atividade de discussão.

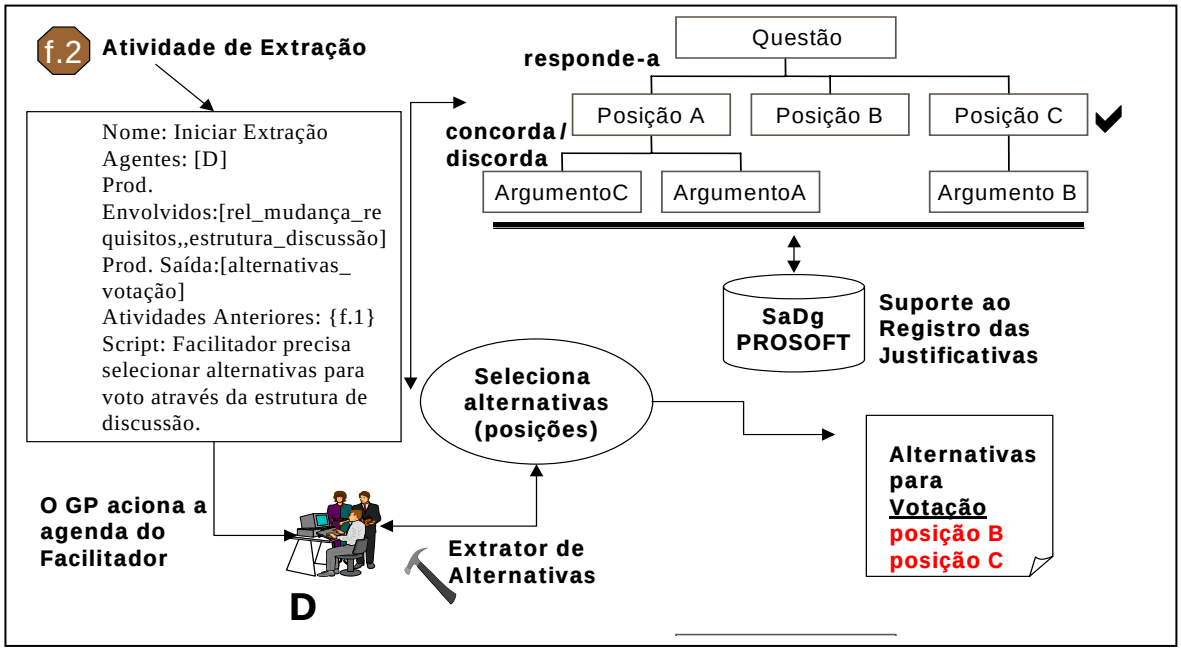


FIGURA 5.8 – Extração de Alternativas para Votação.

Os componentes, data de inicio, data de término, bem como o método de votação são específicos para cada turno. A atribuição de um controle de tempo para execução desta atividade foram determinadas pelo mesmo motivo apresentada na atividade de discussão. Assim os agentes saberão que podem votar nas alternativas referentes ao turno que mais lhe agradam. O facilitador escolhe quais dos métodos de votação disponíveis no SADG será utilizado para cada turno. A Fig. 5.9 mostra um cenário de execução da atividade de votação.

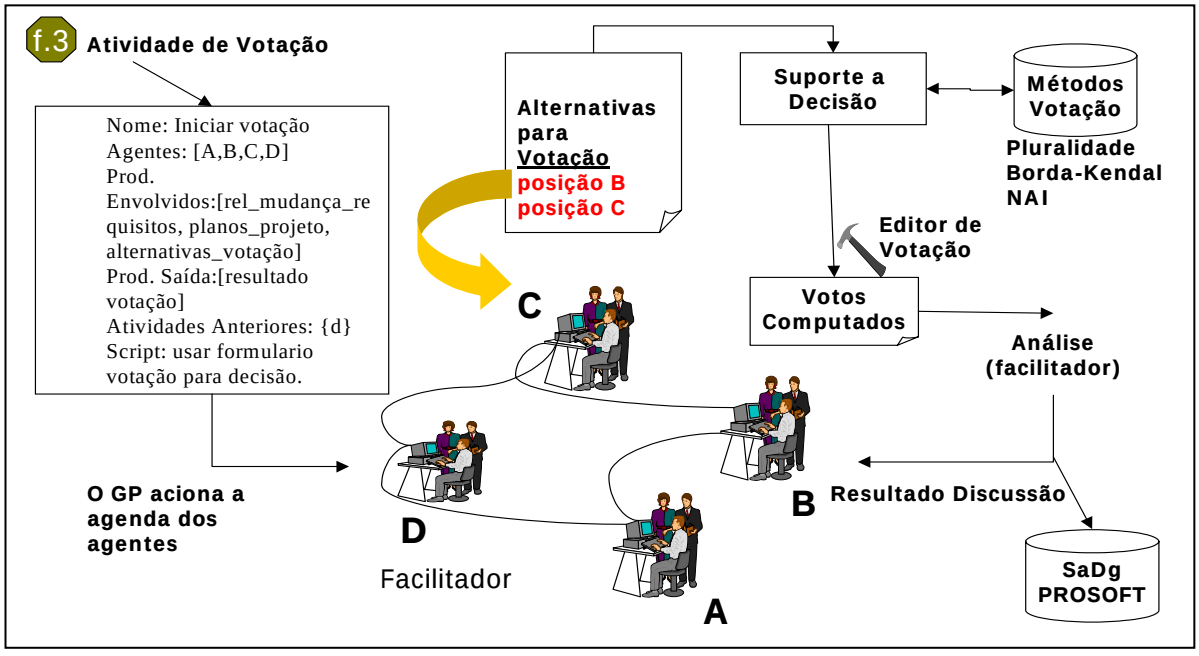


FIGURA 5.9 - Cenário de Execução da Atividade de Votação.

O segundo e subseqüentes turnos são realizados se mais de um turno for definido na norma, em caso de empate do turno anterior. Para as votações no caso de empate, as alternativas de voto poderão ser as seguintes: todas as alternativas e alternativas empatadas. Se ao final do número de turnos estipulados o empate permanecer, o desempate deverá ser feito pelo facilitador. Pode acontecer também que não seja necessário a realização de uma votação. Durante a atividade de argumentação os participantes podem demonstrar um consenso para resolução de um determinado problema, de maneira que, realizar uma atividade de votação só confundiria e atrasaria a tomada de decisão. Cabe ao facilitador atentar para esta possibilidade e a continuidade destas dentro do processo decisório.

No momento de configuração da norma, o facilitador deverá dar uma atenção especial aos intervalos de tempo entre a discussão, extração de alternativas e votação, bem como entre os diferentes turnos de votação. Quanto ao intervalo da discussão, este deverá ser iniciado após um período suficiente para que os agentes estejam cientes das atividades de discussão definidas em suas agendas.

Entre a discussão e a primeira votação deverá ter um intervalo para que o facilitador realize a extração das alternativas para votação. Os intervalos durante os turnos de votação devem permitir um tempo mínimo para apresentação dos dados levantados, bem como a preparação das alternativas para a etapa de votação seguinte, se assim for o caso.

Assim como na atividade de discussão, o facilitador também poderá efetuar alterações, sempre que estimar necessário, proporcionando o melhor aproveitamento da atividade de votação e da qualidade das escolhas realizadas pelos agentes.

5.1.5 Problema de Decisão

Os últimos componentes a serem configuradas pelo facilitador são referentes ao problema de decisão, tema da discussão/votação, sendo: problema de decisão, data, hora, palavra-chave, conteúdo e objetivos.

No componente problema de decisão o facilitador deverá preencher com o idpd, que corresponde ao problema de decisão proposto(e.g. man_rotina_com - manutenção da rotina de comunicação).

Os componentes data de inicio e termino, tem por objetivo registrar o momento que o problema de decisão foi gerado pelo facilitador. Neste caso o SADG proposto automaticamente define estas propriedades no momento da criação do problema.

O facilitador poderá preencher os componentes da forma que achar necessário(e.g, problemas com a sincronia dos processos, falhas na comunicação dos módulos, etc), visando que os membros do grupo possam melhor compreender o problema de decisão. Por último é definido o objetivo, onde se que chegar, ou seja, o que os agentes precisam discutir e posteriormente votar, para se alcançar uma melhor solução para o problema em questão.

Um problema de decisão pode ser composto por outros problemas e pode haver uma dependência entre estes, de tal maneira que para resolvê-los, será necessário resolver cada tema, começando pelo mais inferior até chegar ao nível superior(nível 0).

5.2 Suporte à Argumentação

Através do suporte à argumentação, fornecido pelo SaDg Prosoft, utilizando-se do Editor de Argumentação, cada participante do grupo de desenvolvimento responsável por participar do processo decisório pode, no momento em que for convocado em sua agenda, contribuir com suas idéias.

Os participantes envolvidos nesta atividade do processo estabelecem a comunicação incluindo suas contribuições na estrutura de argumentação fornecida pelo SaDg Prosoft. Esta estrutura é implementada segundo uma abordagem do modelo IBIS. O utilização do modelo IBIS auxilia o processo de decisão em grupo, pela possibilidade de, através da combinação de nós de informação, proporcionar liberdade para que idéias e contribuições sejam expressas de forma natural e organizada, favorecendo o armazenamento de informações a cerca das justificativas de projetos – *Design Rationale*.

Como os elementos manipulados durante as atividades de argumentação são pensamentos e idéias de cada participante e que é necessário transformar estas idéias geradas num documento organizado e estruturado, a abordagem hipertexto consegue associar estas contribuições estabelecendo representações para os objetos sendo manipulados e, consequentemente, o entendimento da argumentação em andamento.

Os elementos de argumentação podem conter as seguintes contribuições: alternativas, argumentos contra ou a favor outras contribuições, e posições sobre estes argumentos.

A base de informações geradas a cada discussão, ou seja, a cada atividade de argumentação, é utilizada como memória de justificativas de projeto. Em sua rede de associações são registrada todas as contribuições dos participantes bem como a dinâmica da discussão.

O registro das decisões é garantido, possibilitando que a qualquer momento, até mesmo fora de um processo decisório, seja possível realizar consultas a base, em busca de informações que possam esclarecer os porquês de determinadas decisões, bem como aprendizado dentro do ambiente de desenvolvimento.

O SaDg Prosoft oferece mecanismos de consulta à base de decisão. Esta pode ser pesquisada através de operações suportadas pelo Explorador de Decisões.

Este registro além de facilitar a consulta, torna possível que outros grupos reutilizem as informações armazenadas em discussões semelhantes às que precisam resolver. As informações também podem ser utilizadas como experiências para outros grupos analisarem e ter em mãos uma idéia das estratégias a utilizar para uma decisão e o que evitar ao longo do processo.

Os principais componentes dos elementos de discussão dentro do SADG proposto são:

- **IdElem** – identificação do elemento de discussão;
- **IdUsuario** – identificação do usuário;
- **TipoElemento** – podendo ser um dos tipos de elementos suportados pelo IBIS (questão, posição e argumento);
- **RotuloElemento** – uma palavra chave que identifique este elemento para

pesquisas (e.g. método formal);

- **TextoElemento** – descrição textual.
- **DataApresentação** – data da contribuição do elemento.
- **Relação** – qual a relação deste elemento para com outros elementos do modelo (questionamento, aprovação, rejeição e resposta).

A Fig. 5.10 abaixo, mostra os componentes relacionados durante a atividade de argumentação, armazenando todas as justificativas, quanto a resolução de questões pertinentes ao processo de software apresentados pelos agentes A e B.

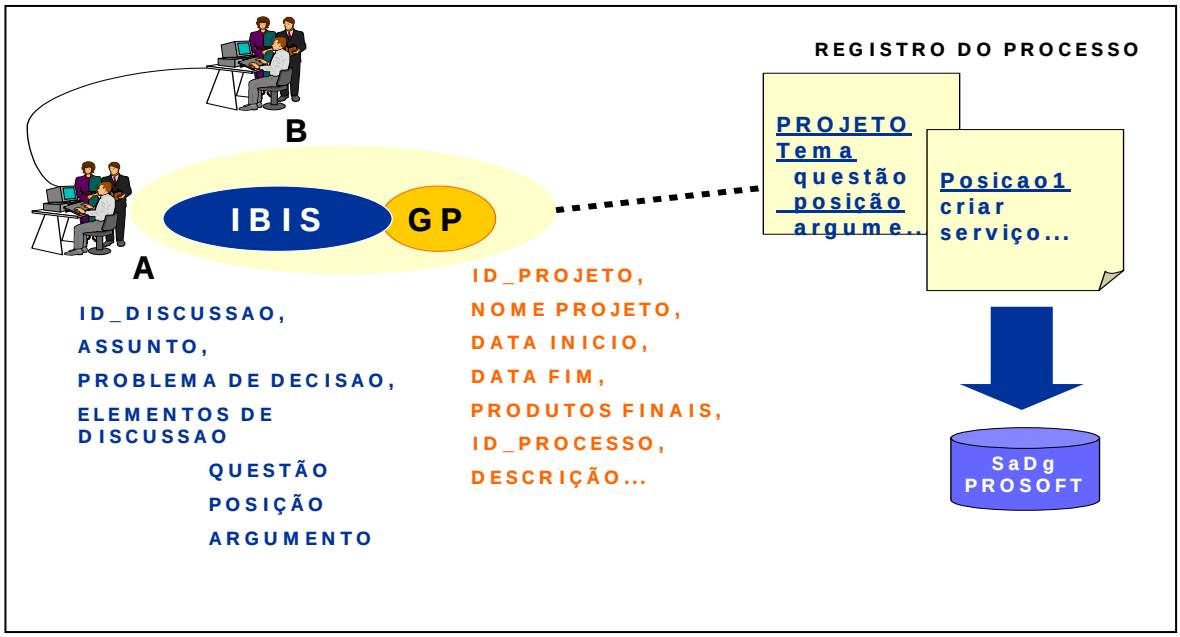


FIGURA 5.11 – Suporte à Argumentação Fornecido pelo SaDg PROSOFT.

5.3 Suporte à Decisão

Após o término da atividade de argumentação, os agentes estarão aptos a votar nas alternativas eleitas no processo de extração, bem como participar de outros turnos, seguindo as regras definidas na norma: numero de turnos, duração, e método de votação para cada turno.

É importante ressaltar a contribuição dos estudos realizados em [BAC 97] cujo qual utiliza métodos de votação em um sua forma de escolha ou decisão. Alguns deste métodos estudados foram implementados neste modelo SADG proposto, como pode ser notado nas seções seguintes.

Nesta seção serão apresentados o processo de extração, os métodos de votação com suas respectivas aplicações.

5.3.1 Métodos de Votação

Dos métodos de votação encontrados na literatura [STR80][CES94][COO82], nota-se que a maioria destes podem ser agrupados dentro de três grandes grupos, considerando o numero de alternativas que podem ser votadas por pessoa, e a forma de expressão de voto. A diferença entre eles está na forma de computar os votos vencedores.

No primeiro grupo de métodos de votação cada agente seleciona no máximo uma alternativa. São exemplos destes métodos Pluralidade, Copeland, e RunOff [STR80][CES94][COO82]. Será utilizado, deste grupo, no modelo SADG proposto, o método pluralidade.

No segundo grupo de métodos de votação cada agente seleciona quantas alternativas desejar. Exemplos destes procedimentos são Aprovação, Bids, Black [STR80]. Não será utilizado nenhum procedimento deste grupo.

No terceiro grupo, são identificados aqueles onde os agentes devem atribuir notas a cada alternativa. Dentre os exemplos encontrados na literatura podemos citar Hare [STR80], Borda-Kendal [COO88] e NAI[BUI95]. Os métodos NAI e Borda-Kendal serão utilizados pelo modelo proposto.

Os métodos foram escolhidos pelas suas características dentro de cada grupo além de sua vasta utilização pela literatura especializada.

Como trabalho futuro, neste SADG, poderiam ser implementados outros métodos de votação a fim de proporcionar um conjunto maior de opções aos agentes durante a atividade de votação.

5.3.2 Método de Votação Pluralidade

O método de pluralidade [STR80][CES94] é o mais conhecido entre os métodos disponíveis na literatura. Neste método cada agente poderá votar em uma única alternativa e, aquela que obtiver maior número de votos, vence.

Os passos para se alcançar os resultados são os seguintes:

- Verifica-se os candidatos (alternativas de voto) n, (n>1), e que será representado pelo índice i;
- Verifica-se o numero de votantes, m, m>1, e que será representado pelo índice j;
- Sabendo-se que o número máximo de possibilidades (x) por votante é igual a 1 (0<=x<=1), pode-se afirmar que cada eleitor j pode votar em no máximo uma alternativa i. Para representar os votos é usada uma matriz r MxN, onde para cada linha j(votante) existe no máximo uma coluna i(alternativa) com valor 1 (rji =1). As demais colunas da linha j recebe o valor 0 (rji=0).
- Com os votos realizados, é possível executar a formula do método que tem por objetivo selecionar a alternativa i que alcançar maior numero de votos.

$$MAX \quad \{ SI = \sum_{j=1}^{m} rji \ / \ 1 \leq i \leq n \}$$

A questão de desempate não é tratada por este modelo. Para isto, deverão ser realizadas várias votações até que o desempate ocorra, ou então entrar em consenso sem a utilização do método.

Outro detalhe importante é que este tipo de método poderá não expressar a vontade da maioria dos votantes. Em processos de votação onde existe apenas duas alternativas para voto, pode-se assegurar que alternativa vencedora expressa a vontade da maioria dos votantes. O mesmo não pode ser afirmado para processos que existam um número de alternativas maior ou igual a três. Neste caso, cabe ao facilitador estimar a necessidade de alterar as regras de votação para um outro tipo de método mais adequado a situação, ou propor outros turnos .

5.3.3 Método Borda-Kendall

Neste método[COO82][CES 94], os tomadores de decisão deverão aprovar todas as alternativas atribuindo notas ou pesos para cada uma delas. Terminada a votação, obtêm-se uma lista ordenada decrescentemente pelo grau de aprovações de cada alternativa candidata, vencendo aquela que obtiver maior grau.

Os passos para se alcançar os resultados são os seguintes:

- Verifica-se as alternativas n, n>1, e que será representado pelo índice i;
- Verifica-se o numero de votantes, m, m>1, e que será representado pelo índice j;
- Sabendo-se que o numero máximo de possibilidades de voto(x) por votante é igual ao número máximo de alternativas de voto n(0<=x<=n), pode-se afirmar que cada votante j pode votar em quantas alternativas i desejar, desde que seja uma única vez. Para representar os votos é usado uma matrix r MxN, onde para cada linha j(votante) deve receber no mínimo 0 e no máximo n colunas (alternativas) atribuindo uma nota/peso, obedecendo a escala prevista para apresentação destes valores (rji=peso/nota). As colunas da linha j que não foram votadas pelo eleitor j, receberão peso 0 (rji=0).
- Com os votos realizados, é possível executar a formula 2 que tem por objetivo selecionar a alternativa j que obtiver, o seu somatório, o maior peso/nota.

$$MAX \quad \{ SI \; = \sum_{j=1}^{m} rji \;\; / \, 1 \leq i \leq n \}$$

Neste método também não se trata a questão de empate. Mais uma vez cabe ao facilitador analisar o processo e tomar a melhor solução analisando a escala de atribuição para cada alternativa. Por exemplo selecionando a alternativa com valores peso/nota mais elevados, constantes e aproximados. Em [COO82] é apresentado um algoritmo alternativo para cômputo dos votos que tenta solucionar este problema.

5.3.4 Método NAI

O algoritmo NAI (Negotiable Alternative Identifier) é chamado por [BUI 95] como um mecanismo de negociação. Ele tem por objetivo apresentar aos tomadores de decisão um consenso das alternativas preferidas, de acordo com as pontuações atribuídas pelos mesmos.

Este algoritmo é uma alternativa aos métodos de votação apresentados anteriormente, porque visa encontrar um compromisso entre as alternativas com maior aprovação e menor rejeição. A busca por consenso de alternativas é obtida através de três operações básicas executadas pelo algoritmo, que são: expansão, contração e intersecção.

As operações de expansão, contração e intersecção podem ser executadas quantas vezes forem necessárias até que os tomadores consigam alcançar uma resultado. Uma das vantagens em utilizar este algoritmo em relação aos métodos de votação relatados anteriormente, é que o mesmo resolve a questão do desempate através das operações comentadas acima. O método multi-atributo utilizado neste algoritmo pode ser o mesmo a ser empregado no método de votação Borda-Kendall.

TABELA 5.1 – Cálculo do método NAI.

Verifica-se as alternativas de voto.	n / n >1, representado pelo índice i
Verifica-se o número de votantes.	m / m>1, representado por d
Vetor de preferências que cada votante estabelece.	rd = [r_{di}], onde $r_{di} \geq 0$
As preferências são normalizadas como ri, classificando-as por ordem de preferência.	$\sum_{j=1}^{n} rdi = 1$
Calcula-se a preferência cumulativa que cada votante dá as primeiras n alternativas.	rdn = $\sum_{j=1}^{n} rdi = 1$
Operação de expansão Calcula-se o índice estrutural de preferências entre subconjuntos de alternativas.	($SIj = \left(\frac{1}{j}\right)$ md(j), onde $md(j) = \left(\frac{1}{j-1}\right) \sum_{k=1}^{j-1} mdkj$, $mdkj) = \left(\frac{rdk}{\frac{k}{\frac{rdj - rdk}{j-k}}} \right)$, onde j=2,..,n k=1,..,j-1 e rdk é a preferência cumulativa para as primeiras k alternativas
Operação de contração Se obtém o conjunto de alternativas prediletas entre aquelas selecionadas na operação de expansão.	Cd, i* = ri / ri*
Operação de interseção Remover alternativas duplicadas após operação de contração.	
Se obtém o primeiro consenso entre as alternativas preferidas.	

5.4 Suporte ao Compartilhamento de informações

A base de informações gerada durante todo processo decisório pode ser utilizada como memória da decisão. Através da rede de associações registradas pelo Editor de

argumentação de argumentação, bem como os registros de votação, fica garantido o registro da dinâmica do processo de decisão, possibilitando que os participantes consultem a base quando necessário, em busca de informações que possam esclarecer os motivos de determinadas decisões.

O SaDg PROSOFT permite mecanismos de consultas a base de informações. Esta pode ser feita através da pesquisa oferecida pelas associações do modelo de argumentação, através dos rótulos dos elementos ou pela palavra chave que define toda atividade de decisão.

Este registro da decisão não apenas facilita a consulta como possibilita que os outros grupos em discussões seguintes, utilizem as informações registradas em discussões passadas, em problemas semelhantes aos resolvidos nestas discussões. Mesmo que não possam ser largamente utilizadas, as informações contidas durante o processo, podem servir como experiências para outros grupos analisarem e ter em mãos uma idéia das estratégias a adotar para uma decisão e o que evitar durante o processo.

5.5 Exemplo de Execução

Tomaremos uma situação hipotética onde existam quatro agentes desenvolvedores no ambiente, que são A, B, C, D. Eles estão conectados no ambiente PROSOFT e realizando cada um suas tarefas previamente agendadas pelo Gerenciador de Processo.

O Agente D tem privilégios diferentes dos demais, podendo ser considerado como desenvolvedor “senior”. O mesmo está cadastrado como usuário facilitador para o ambiente SADG.

O ponto de discussão é levantado e iniciado através de troca de mensagens pelo PROSOFT Cooperativo. A situação é desenhada pelo agente A, cujo qual, cria e atualiza periodicamente um módulo de programa para uma nova função do ambiente. Os agentes B, C, D utilizam este módulo em suas tarefas diárias. O agente A deve fornecer informações sobre as atualizações realizadas para cada novo serviço implementado neste módulo, o que dificulta e atrasa o projeto. A maneira como o agente A vem implementando estas alterações não é bem adequada. O agente D sugere uma tomada de decisão para se encontrar uma forma mais efetiva de se implementar esta *feature* no ambiente. A Fig. 5.1 apresentada no início deste capítulo pode ser usada como referência ao cenário levantado.

Como já apresentado na Fig. 5.2, o SaDg PROSOFT é composto por um conjunto de ferramentas que auxiliam a tomada de decisão assim desenhadas:

- **Editor de Problemas de Decisão**: é através desta ferramenta que serão cadastrados os problemas a serem discutidos dentro do ambiente. Apenas o facilitador poderá ter acesso a este serviço.
- **Editor de Normas**: através dos formulários dispostos dentro deste editor, o facilitador configura todas informações e as regras necessárias para a definição das atividades de um processo decisório, bem como os agentes envolvidos. Utilizando-se de operações disponíveis nesta ferramenta o facilitador deverá interagir com o GP mapeando a Norma para um processo executável, bem como abrindo uma sessão de decisão para a norma em questão.

- **Editor de Argumentação**: utilizando-se do modelo de argumentação fornecido pelo SaDG PROSOFT, os agentes poderão realizar suas contribuições usando dos elementos de argumentação e também criando relações entre estes elementos.
- **Extrator de Alternativas**: é um recurso disponibilizado pelo SADG de uso apenas pelo facilitador. No momento em que se encerra a atividade de argumentação, o facilitador realiza uma extração dos elementos relevantes para a atividade de votação.
- **Editor de Votação**: nesta ferramenta, os agentes participarão nas alternativas dispostas para votação. Após a finalização do atividade, será executado o método de votação para cômputo dos votos.

Conforme apresentado na seção 5.1.1, são dois os papéis principais dentro de um processo decisório. Cada papel possui responsabilidades distintas perante ao processo.

Os passos a serem realizados pelo agente D(facilitador) para realização da atividade de decisão devem ser:

- **Definir problemas de decisão**, com todas as informações pertinentes a estes, utilizando-se do Editor de Problemas de Decisão;
- **Configurar uma norma** com todas suas características e regras para o processo decisório, utilizando-se do Editor de Normas;
- **Transformar a norma em processo executável** para ser interpretado pelo GP, utilizando-se do Editor de Normas;
- **Abrir uma sessão de decisão** através do Editor de Normas;
- **Definir um conjunto de palavras chaves** para consultas futuras sobre o processo de decisão através do Editor de Normas;
- **Acompanhar e coordenar** as atividades que formam a atividade de decisão;
- **Selecionar alternativas para votação** após o término da atividade de argumentação, utilizando-se do Extrator de Alternativas;
- **Computar os votos** quando do término da atividade da votação, utilizando-se do Editor de Votação.

Após todas as definições iniciais do processo de decisão terem sido realizadas pelo agente D, o gerenciador de processos passará a fornecer um suporte adicional sobre a coordenação das atividades necessárias para a tomada de decisão dentro do ambiente PROSOFT.

Os passos seguintes correspondem as atividades inseridas nas agendas dos agentes pelo GP, informando detalhes sobre cada atividade, as ferramentas a serem utilizadas, bem como prazo previsto para conclusão.

Os agentes A, B, C deverão realizar as seguintes atividades:

- **Apresentar suas contribuições** (elementos de argumentação) para a atividade de argumentação, utilizando-se do Editor de Argumentação;
- **Participar da votação**, votando nas alternativas que achar relevante para a resolução do problema em questão, utilizando-se do Editor de Votação.

Os agentes poderão consultar, na medida em que acharem necessário, as informações geradas durante o processo de decisão. Estas informações poderão ser obtidas acessando as operações de pesquisa disponíveis no explorador de decisões.

No próximo capítulo será apresentada a implementação do modelo SADG proposto para tomada de decisão durante o processo de desenvolvimento de software sob o paradigma de ferramentas especificadas para o ambiente PROSOFT.

6 Especificação Formal do SaDg PROSOFT

O ambiente PROSOFT permite a construção formal de ferramentas que podem ser integradas a ele. O uso de linguagens de especificação formal provê uma base matemática que torna possível garantir propriedades como completeza, corretude e ausência de ambiguidade, que são difíceis de obter através de uma descrição informal.

Através da construção de ATOs PROSOFT, várias ferramentas já foram propostas para o ambiente. O capítulo 5 apresentou uma descrição informal do modelo de SADG proposto para o PROSOFT. Para gerenciar este modelo, foram construídas ferramentas na forma de vários ATOs, visando atender os requisitos já apresentados em capítulos anteriores.

Para facilitar o entendimento e as funcionalidades dos ATOs desenvolvidos, neste capítulo, serão apresentadas apenas as classes e as operações disponíveis em cada ATO. O anexo 1 deste trabalho fornece uma visão detalhada das operações implementada para cada ATO. Mais informações a cerca dos ATOs PROSOFT podem ser obtidas no capítulo 4.

Este capítulo apresenta a integração do novo componente à arquitetura do PROSOFT, o modelo SaDg PROSOFT e a especificação do SADG, através de classes construídas e suas operações algébricas.

6.1 Arquitetura do PROSOFT

O modelo SADG proposto utiliza a arquitetura atual do PROSOFT, o qual provê características de um sistema distribuído e cooperativo. Possui um Gerenciador de Processos, através do qual é possível coordenar atividades e usuários.

6.2 Atividades de um Processo de Decisão

Um processo de decisão é composto por um conjunto de atividades. Para cada atividade, dentro do processo de decisão, são definidos papéis com suas respectivas responsabilidades e ferramentas para manipulação e execução destas atividades. A Fig. 6.2 apresenta as atividades de um processo de decisão.

O modelo SaDg PROSOFT, possui as seguintes as atividades para tomada de decisão:

- **Atividade de Decisão –** corresponde a principal atividade de um processo de decisão. Define as diretrizes do processo e as atividades necessárias para resolução dos problemas a serem discutidos.
- **Atividade de Argumentação –** corresponde a atividade de *brainstorming*, onde cada participante apresenta suas idéias sobre o problema em questão.
- **Atividade de Extração –** nesta atividade apenas o facilitador poderá interagir, de forma a selecionar as principais contribuições apresentadas durante a atividade de argumentação.
- **Atividade de Votação –** corresponde a busca pela solução, onde os participantes elegem as alternativas que melhor resolvem o problema em questão.

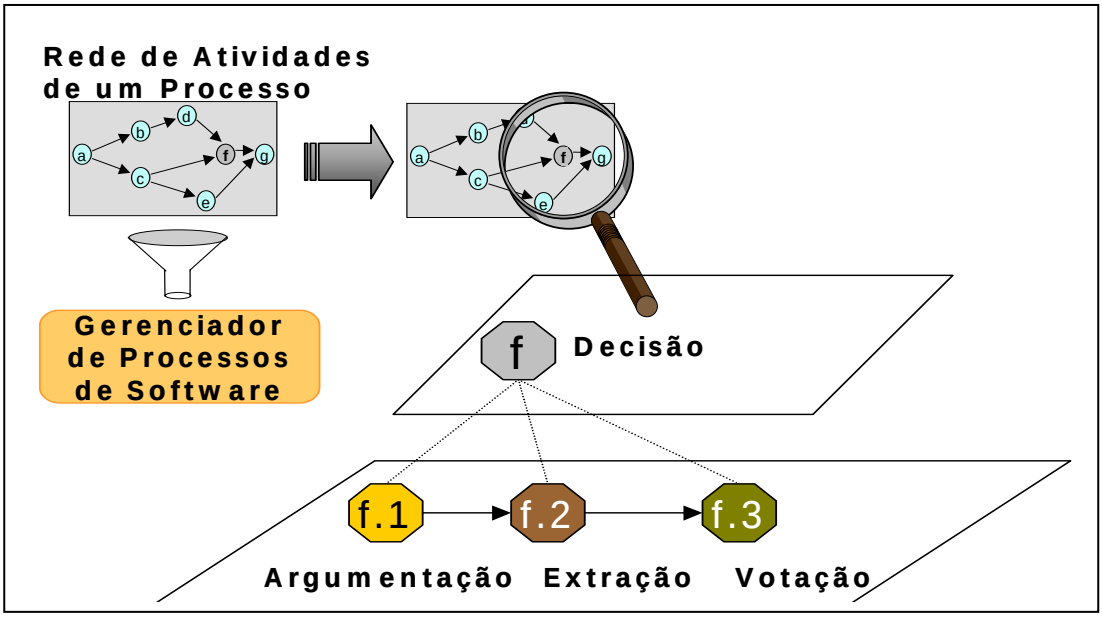


FIGURA 6.2 – Atividades de um processo de decisão.

6.3 Descrição do SaDg PROSOFT

Para o SaDg PROSOFT foram construídas 8 ferramentas, conforme pode ser visto na figura 6.3. As ferramentas utilizadas são:

- **ATO SaDg**: principal ferramenta do ambiente, composta de um conjunto de usuários (facilitadores), conjunto de problemas de decisão, conjunto de normas e conjunto de decisões. Esta ferramenta engloba todas as outras ferramentas do ambiente;
- **ATO Usuários**: ferramenta que permite a criação e a manipulação de usuários facilitadores dentro do SADG. Esta ferramenta está associada ao ATO SaDg;
- **ATO PD**: ferramenta que permite definição e manipulação de problemas de decisão, suas características e também a decomposição deste em outros problemas. Esta ferramenta está associada ao ATO SaDg. Corresponde ao Editor de Problemas de Decisão;
- **ATO Normas**: ferramenta que permite configuração e manipulação de normas, suas características, regras para as atividades do processo decisório e a marcação do objeto de decisão (e.g. processo de software). Esta ferramenta está associada ao ATO SaDg. Corresponde ao Editor de Normas;
- **ATO Decisões**: ferramenta que define e gerência as ferramentas que auxiliam no processo de tomada de decisão dentro do SADG. Esta ferramenta está associada ao ATO SaDg.;
- **ATO Arg**: ferramenta que permite a criação e manipulação de elementos de argumentação contribuídos dentro da rede de discussão. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Editor de Argumentação;
- **ATO Extração**: ferramenta que permite criação e manipulação das alternativas extraídas a partir dos elementos de argumentação. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Extrator de Alternativas;
- **ATO Votação**: ferramenta que permite criação e manipulação de votos para as alternativas selecionadas durante a atividade de extração. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Editor de Votação.

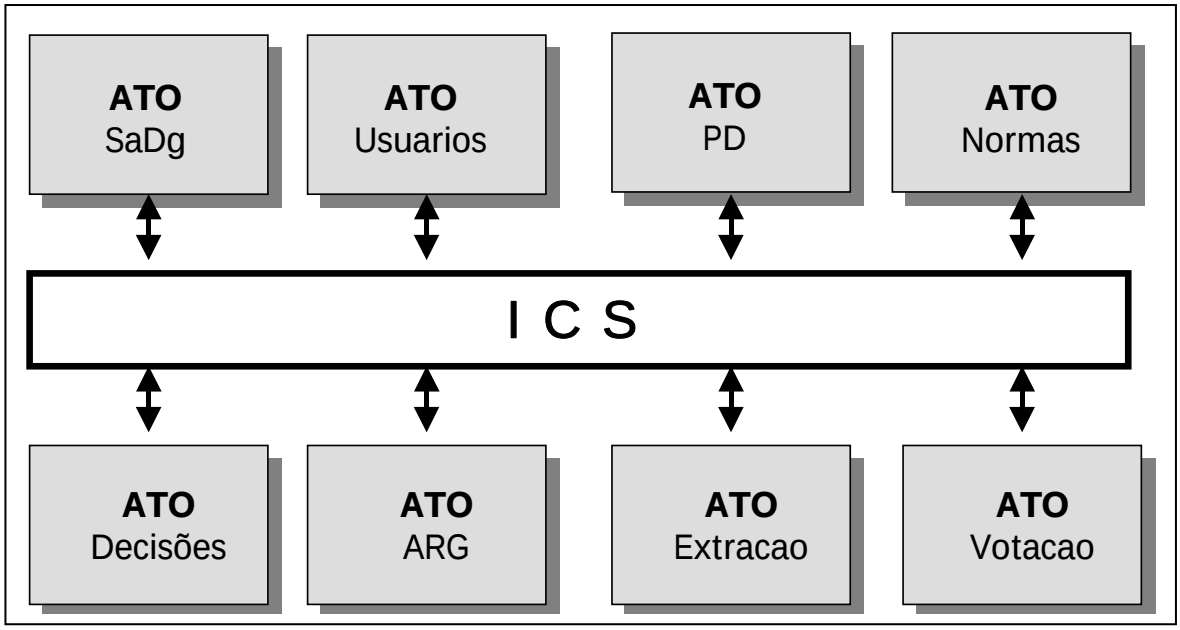


FIGURA 6.3 – Ferramentas do SaDg PROSOFT.

6.4 ATO SaDg PROSOFT

O ATO SaDg é a principal ferramenta do ambiente. Esta ferramenta provê todas as operações necessárias para a definição e manipulação de processos de decisão. As demais ferramentas do modelo proposto são invocadas através de chamadas ICS. A Fig. 6.4 apresenta representação gráfica [NUN 94] da classe deste ATO.

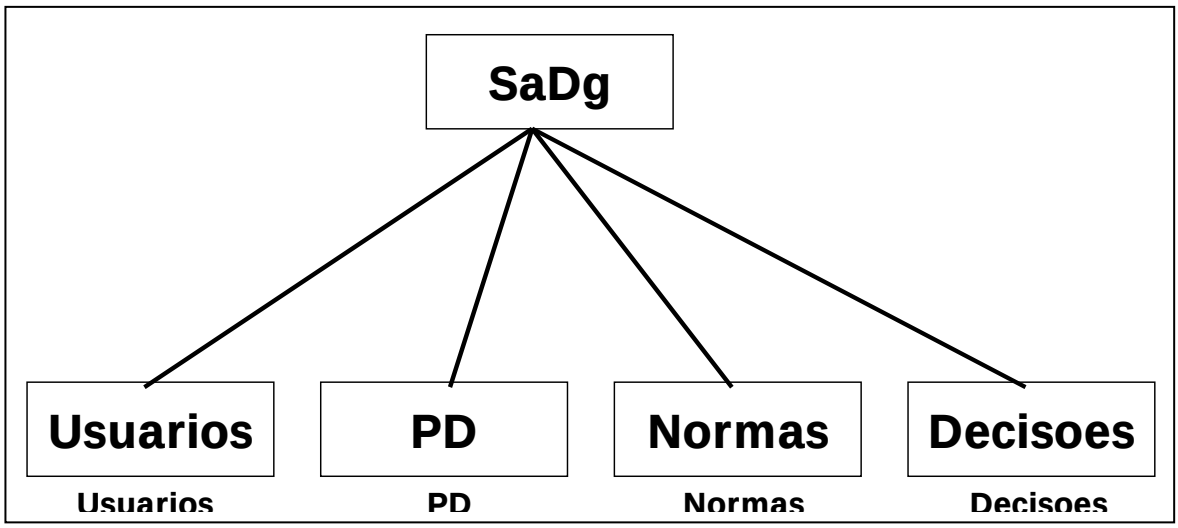


FIGURA 6.4 – ATO SaDg.

A instanciação do ATO SaDg foi definida a partir dos tipos de dados apresentados no capítulo (prosoft). A classe deste ATO possui quatro componentes: um conjunto de usuários (facilitadores), um conjunto de problemas de decisão, um conjunto de normas e um conjunto de decisões. As folhas da instanciação fazem referência a outros ATOS, que serão apresentados neste capítulo.

O componente Usuários contém todos os facilitadores cadastrados para o ambiente. O componente PD fornece os problemas de decisão cadastrados. O componente Normas estabelece as regras necessárias para a configuração das atividades do processo decisório. As decisões correspondem ao conjunto de rede de informações associadas as decisões realizadas dentro do ambiente.

A seguir serão apresentadas as operações disponíveis para o ATO SaDg. Uma parte das operações estão disponíveis apenas aos usuários facilitadores, principalmente as operações de manipulação e configuração das regras para execução de atividades de decisão. Entretanto, existem outras operações disponíveis, para usuários agentes, de acordo com suas necessidades. Além dessas operações, existem outras denominadas de observadoras, que fornecem informações gerenciais ou informações sobre os componentes do ambiente e operações internas utilizadas neste ATO.

6.4.1 Operações Criadoras

Um objeto do tipo SaDg é criado através da operação **c_cria_sadg**. No momento da criação apenas os gerentes cadastrados no GP podem manipular objetos deste ATO. Após a criação de usuários(facilitadores) estes podem manipular qualquer objeto dentro do ambiente.

6.4.2 Operações Modificadoras

- **m_inclui_facilitador_sadg** – cadastra um facilitador no SaDg.
- **m_remove_facilitador_sadg** – remove um facilitador do SaDg.
- **m_adiciona_pd_sadg –** cadastra um problema de decisão no SaDg.
- **m_remove_pd_sadg –** remove um problema de decisão do SaDg.
- **m_adiciona_comp_pd_sadg –** adiciona um componente para um problema de decisão no SaDg.
- **m_remove_comp_pd_sadg –** adiciona um componente para um problema de decisão no SaDg.
- **m_configura_norma_sadg –** configura uma Norma dentro do SaDg.
- **m_remove_norma_sadg –** remove uma Norma do SaDg.
- **m_adiciona_agente_sadg –** adiciona um agente para uma Norma do SaDg.
- **m_remove_agente_sadg –** exclui um agente para uma Norma do SaDg.
- **m_modifica_projeto_gp** – altera o nome do projeto do gerenciador de processos configurado para uma Norma do SaDg.
- **m_gera_atividade_decisao** – gera uma atividade de decisão para um projeto em execução no gerenciador de processos.
- **m_gera_atividade_argumentacao** – gera uma atividade de argumentação para um projeto em execução no gerenciador de processos.
- **m_gera_atividade_extracao** – gera uma atividade de extração para um projeto em execução no gerenciador de processos.

- **m_gera_atividade_votacao** – gera uma atividade de votação para um projeto em execução no gerenciador de processos.
- **m_gera_atividade_compvoto** – gera uma atividade de fechamento de votação para um projeto em execução no gerenciador de processos.
- **m_cria_ambiente_decisao –** prepara o ambiente com as ferramentas para tomada de decisão.
- **m_inclui_decisao_sadg –** inclui uma decisão no SaDg.
- **m_remove_decisao_sadg –** remove uma decisão do SaDg.
- **m_inclui_palavra_decisao_sadg –** adiciona uma palavra chave para a tomada de decisão.
- **m_remove_palavra_decisao_sadg –** remove uma palavra chave da tomada de decisão.

6.4.3 Operações Observadoras

- **o_facilitador_cadastrado_sadg** – verifica se um determinado facilitador está cadastrado no SaDg.
- **o_existe_problema_sadg** – verifica se existe um determinado problema cadastrado no ambiente.
- **o_componentes_problema** – recupera todos os componentes de um problema.
- **o_objetivo_problema** – recupera o objetivo a que se chegar na tomada de decisão de um determinado problema.
- **o_facilitador_resp_norma** – recupera o facilitador responsável por uma Norma dentro do ambiente
- **o_agentes_eresp_norma** – verifica se o agente está cadastrado para uma Norma.
- **o_agentes_norma** – recupera todos os agente selecionados para uma Norma.
- **o_problema_norma –** recupera o problema de decisão configurado para uma Norma.
- **o_eproblema_norma –** verifica se um problema está cadastrado para uma Norma.
- **o_projeto_gp_norma –** recupera um projeto do gerenciador de processos configurado para uma Norma.
- **o_regras_arg_norma –** verifica de existe regras de argumentação cadastradas para uma Norma.
- **o_regras_ext_norma –** verifica de existe regras de extração cadastradas para uma Norma.
- **o_regras_vot_norma –** verifica de existe regras de votação cadastradas para uma Norma.
- **o_normas_resp_facilitador** – verifica todas as normas configuradas por um determinado facilitador.

6.5 ATO Usuarios

O ATO Usuarios gerencia as informações sobre os facilitadores cadastrados para o ambiente SaDG PROSOFT. Sem o cadastro prévio destes facilitadores ficam impossível coordenar os atividades de um processo de decisão, ou até mesmo definir problemas de decisão a serem solucionados. A classe do ATO Usuarios é representado na Fig. 6.5.

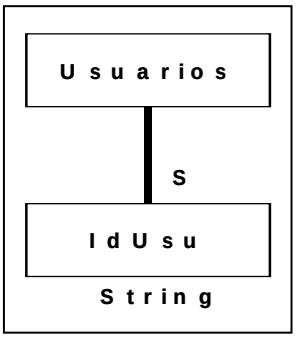


FIGURA 6.5 – ATO Usuarios.

6.5.1 Operações Criadoras

- **c_cria_usuario** – esta operação é responsável pela construção da estrutura inicial que servirá para a configuração dos facilitadores dentro do ambiente. Os facilitadores são criados por usuários gerentes do GP, e apenas por estes. Isto configurado, os facilitadores podem criar problemas de decisão, normas para o processo, bem como realizar decisões dentro SaDg.

6.5.2 Operações Modificadoras

- **m_inclui_usuario** – adiciona um novo facilitador dentro do conjunto de usuários facilitadores cadastrados no ambiente. Esta operação só pode ser realizada por usuários gerentes do GP, e os facilitadores que serão cadastrados precisam existir como agentes dentro do gerenciador de processos.
- **m_exclui_usuario** – remove um determinado facilitador do conjunto de usuários facilitadores cadastrados no ambiente.

6.5.3 Operações Observadoras

- **o_existe_id_usuario** – verifica se já existe definido dentro do SaDg um usuário com o identificador especificado.

6.6 ATO PD

O ATO PD permite a criação e manipulação de problemas de decisão. Um problema de decisão é composto por uma identificação, representado pelo campo *idpd*; por informações a cerca da data de previsão para resolução, representados pelos campos *Data inicio* e *Data fim;* pelo objetivo que se quer chegar, representado pelo campo *objetivo* e por componentes. Componentes são a decomposição do problema de decisão em sub-problemas. A classe do ATO PD é representado na Fig. 6.6.

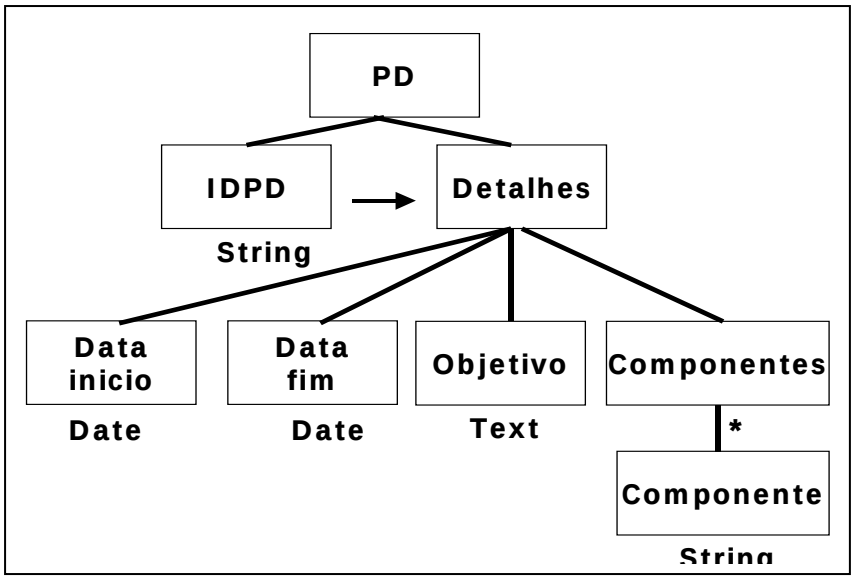


FIGURA 6.6 – ATO PD.

6.6.1 Operações Criadoras

- **c_cria_pd** – esta operação é responsável pela construção da estrutura inicial que servirá para o armazenamento das informações acerca dos problemas para decisão dentro do SaDg.

6.6.2 Operações Modificadoras

- **m_inclui_pd** – adiciona um novo problema de decisão dentro do conjunto de problemas cadastrados no ambiente. Esta operação só pode ser realizada por usuários facilitadores.
- **m_exclui_pd** – remove um determinado problema de decisão do conjunto de problemas cadastrados no ambiente.
- **m_inclui_comp** – adiciona uma dependência de um subproblema para a resolução de um problema de decisão.
- **m_exclui_comp** – remove um subproblema para a resolução de um problema.
- **m_altera_dt_inicio** – altera a data prevista para inicio da decisão sobre o problema.
- **m_altera_dt_fim** – altera a data prevista para termino da decisão sobre o problema.
- **m_altera_objetivo** – altera o objetivo de um problema.

6.6.3 Operações Observadoras

- **o_existe_pd** – verifica se já existe definido dentro do SaDg um problema de decisão com o identificador especificado.
- **o_data_inicio** – recupera a data inicio configurada para decisão de um problema.
- **o_data_fim** – recupera a data final configurada para decisão de um problema.
- **o_objetivo** – recupera o objetivo configurado para decisão de um problema.
- **o_comp_pd** - recupera um conjunto com todos os componentes de um problema.
- **o_existe_comp_pd** - verifica se existe componentes para um problema.

6.7 ATO Normas

O ATO Normas permite a criação e configuração de Normas para processos de decisão. Cada processo de decisão é composto por um conjunto de componentes que caracterizam as atividades a serem realizadas dentro do processo decisório. Estes componentes estão ditribuídos dentro deste ATO de maneira a configurar cada atividade contida dentro do processo. A classe do ATO Normas é representado na Fig. 6.7.

A configuração de uma Norma é composta pelos componentes:

- **Facilitador** – identificação do facilitador responsável pela Norma, consequentemente de um processo decisório.
- **Agentes** – conjunto de agentes que participarão das atividades dentro do processo de decisão.
- **Regra Argumentacao** – contém as diretrizes para a atividade de argumentação. Este componente é um ATO que será detalhado mais adiante.
- **Regra Extracao** – contém as diretrizes para a atividade de extração. Este componente é um ATO que será detalhado mais adiante.
- **Regra Votacao** – contém as diretrizes para a atividade de votação. Este componente é um ATO que será detalhado mais adiante.
- **Descrição** – uma descrição textual da Norma.
- **Nome ProjGP** – identificação do projeto de software cadastrado no Gerenciador de Processos.
- **IDPD** – identificação do problema de decisão, que será trabalhado durante as atividades de decisão.

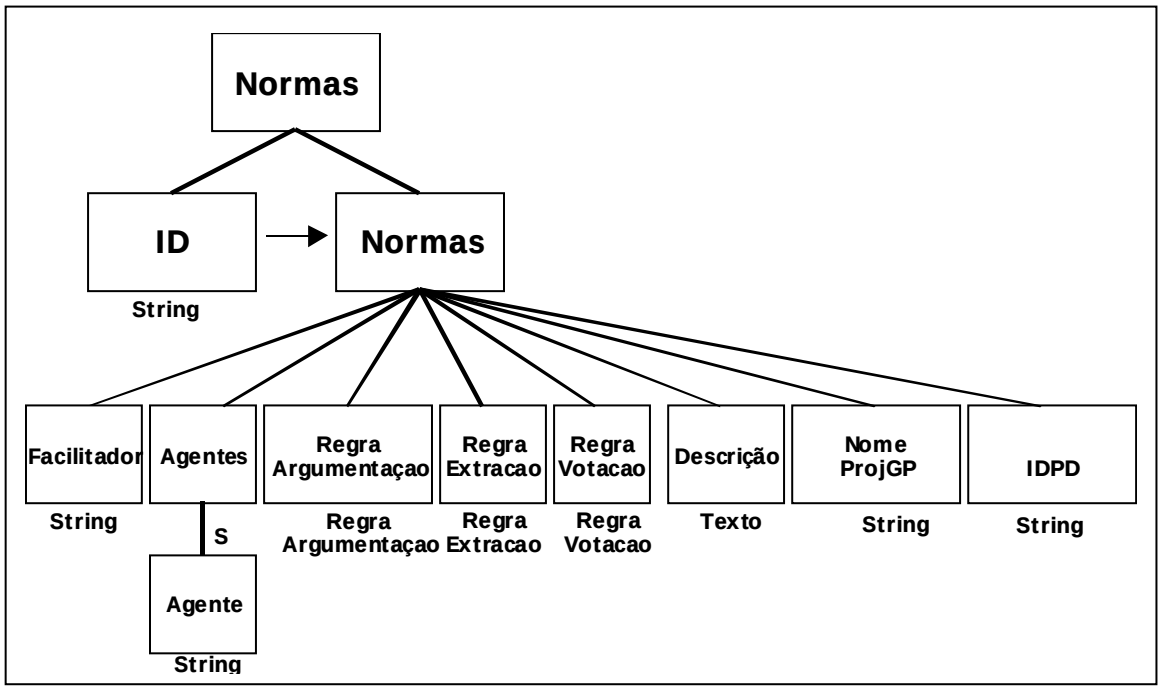


FIGURA 6.7 – ATO Normas.

6.7.1 Operações Criadoras

- **c_cria_norma –** cria no ambiente a estrutura inicial que servirá para o armazenamento das normas, que por sua vez guardam todas as informações a cerca do processo decisório bem como as ferramentas a serem utilizadas em cada atividade do processo**.**

6.7.2 Operações Modificadoras

- **m_inclui_norma** – cadastra uma nova norma no ambiente.
- **m_exclui_norma** – remove uma norma do ambiente.
- **m_altera_facilitador** – altera um facilitador responsável por uma norma especifica. Uma norma possui apenas um facilitador. Entretanto, é possível que um mesmo facilitador esteja participando de mais de uma norma.
- **m_altera_descricao** – altera o texto da descrição da norma.
- **m_altera_regra_arg_dt_prev** – altera a data de inicio e fim de uma regra de argumentação configurada para uma norma.
- **m_altera_regra_arg_nmax** – altera o número máximo de questões, posições e argumentos de uma regra de argumentação configurada para uma norma.
- **m_altera_regra_ext_dt_prev** – altera a data de inicio e fim de uma regra de extração configurada para uma norma.
- **m_altera_regra_ext_facilitador** – altera o facilitador de uma regra de extração configurada para uma norma.

- **m_altera_regra_vot_dt_prev** – altera a data de inicio e fim de uma regra de votação configurada para uma norma.
- **m_altera_regra_vot_metodo** – altera um método de votação de uma regra de votação configurada para uma norma.
- **m_altera_pd** – altera o problema de decisão configurado para uma norma.
- **m_altera_nome_projeto_gp** - altera o projeto GP configurado para a norma.
- **m_inclui_agente** – adiciona os usuários que participarão do processo de decisão. Os mesmos só podem ser incluídos por facilitadores e devem existir como agentes cadastrados no ambiente GP.
- **m_exclui_agente** – remove um agente da norma.
- **m_gera_inclui_atv_decisao** – adiciona uma nova atividade de decisão para um projeto do GP a partir das informações configuradas dentro de uma norma.
- **m_gera_inclui_atv_argumentacao** - adiciona uma nova atividade de argumentação para um projeto do GP a partir das informações configuradas dentro de uma norma.
- **m_gera_inclui_atv_extracao** - adiciona uma nova atividade de extração para um projeto do GP a partir das informações configuradas dentro de uma norma.
- **m_gera_inclui_atv_votacao** - adiciona uma nova atividade de votação para um projeto do GP a partir das informações configuradas dentro de uma norma.
- **m_gera_inclui_atv_compvoto** - adiciona uma nova atividade para se computar os votos para uma decisão dentro de um projeto do GP a partir das informações configuradas na norma.

6.7.3 Operações Observadoras

- **o_existe_facilitador_norma** - verifica se existe um facilitador cadastrado na norma.
- **o_existe_agente_norma** – verifica se um determinado agente existe na norma.
- **o_agentes_norma** – recupera um conjunto que possui os identificadores dos agentes pertencentes a uma norma.
- **o_existe_problema_norma** – verifica se já existe um problema de decisão definido para uma norma.
- **o_id_projeto_gp_norma** – recupera o identificador do projeto gerenciado pelo GP configurado na norma.
- **o_existe_regra_argumentacao** – verifica se já existe uma regra de argumentação configurada para uma Norma.
- **o_existe_regra_extracao** – verifica se já existe uma regra de extração configurada para uma Norma.
- **o_existe_regra_votacao** – verifica se já existe uma regra de votação configurada para uma Norma.
- **o_id_problema_norma** – recupera o problema de decisão configurado para uma norma.

- **o_facilitador_normas** – recupera as normas configuradas por um facilitador.

6.8 ATO Regra Argumentação (REGRAS_ARG)

O ATO Regras_Arg permite a criação e configuração de diretrizes para atividade de argumentação de um processo de decisão. Cada regra para atividades de argumentação é composta por um período previsto para realização, *Data Inicio* e *Data Fim*, por uma delimitação do número máximo de contribuições para questões, argumentos e posições, respectivamente, *Nmax Questão*, *Nmax Argumento*, *Nmax Posição*. A classe do ATO Regras_Arg é representado na Fig. 6.8.

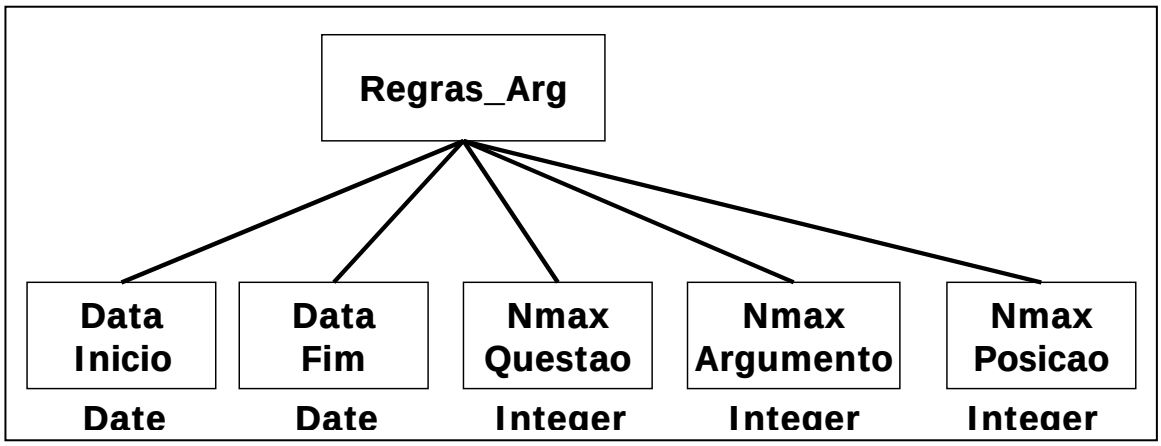


FIGURA 6.8 – ATO Regras_Arg.

6.8.1 Operações Criadoras

- **c_cria_regra_arg –** cria no ambiente a estrutura inicial que servirá para o armazenamento da regra de argumentação configurada para uma norma. Esta operação só pode ser executada por facilitadores.

6.8.2 Operações Modificadoras

- **m_altera_data_inicio** – modifica data inicio prevista configurada em um regra de argumentação para uma norma.
- **m_altera_data_fim** – modifica data fim prevista configurada em um regra de argumentação para uma norma.
- **m_altera_nmax_questao** – altera o número máximo de questões configurado na regra de argumentação de uma norma.
- **m_altera_nmax_argumento** – altera o número máximo de argumentos configurado na regra de argumentação de uma norma.
- **m_altera_nmax_posicao** – altera o número máximo de posições configurado na regra de argumentação de uma norma

6.8.3 Operações Observadoras

- **o_dt_prev_ini_arg** – recupera a data prevista para inicio de uma atividade de argumentação configurado em uma norma.

- **o_dt_prev_fim_arg** – recupera a data prevista para termino de uma atividade de argumentação configurado em uma norma.
- **o_nmax_questao** – recupera o número máximo de questões configurado na regra de argumentação de uma norma.
- **o_nmax_argumento** – recupera o número máximo de argumentos configurado na regra de argumentação de uma norma.
- **o_nmax_posicao** – recupera o número máximo de posições configurado na regra de argumentação de uma norma.

6.9 ATO Regra Votacao (REGRAS_VOT)

O ATO Regras_Vot permite a criação e configuração de diretrizes para atividade de votação de um processo de decisão. Cada regra para atividades de votação é composta por conjunto de zero, um ou mais turnos. Cada turno possui um identificador, *n_turno*, um método para cômputo dos votos, *método votação*, um período previsto para execução da atividade, *Data Inicio* e *Data Fim*. A classe do ATO Regras_Vot é representado na Fig 6.9.

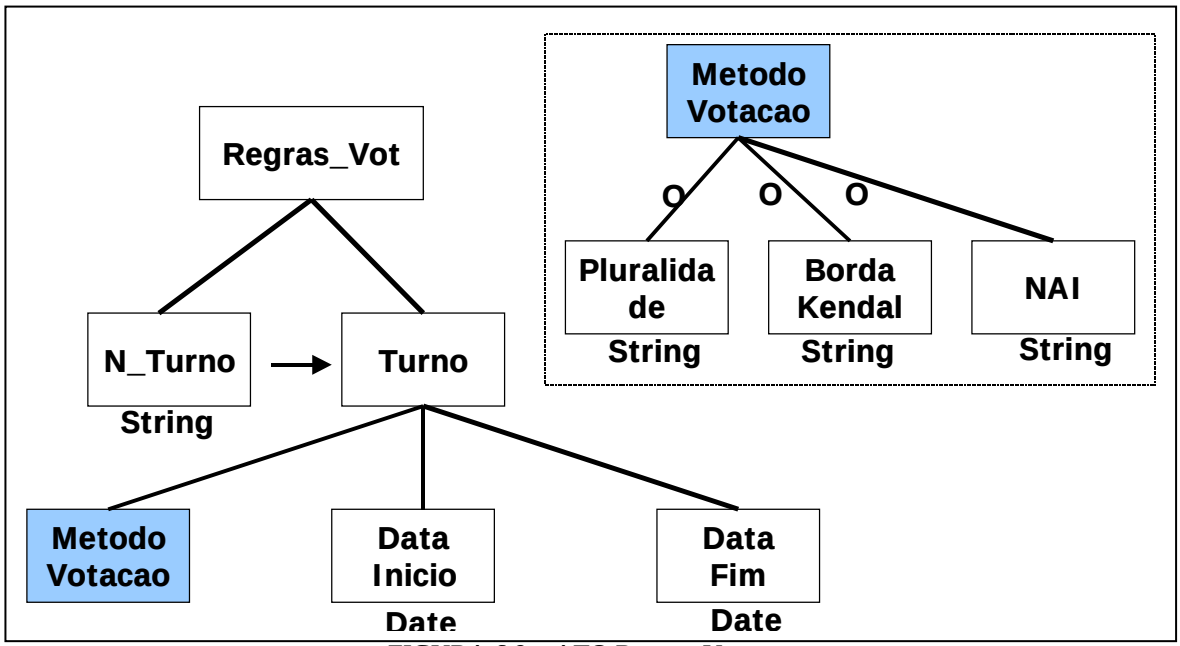


FIGURA 6.9 – ATO Regras_Vot.

6.9.1 Operações Criadoras

- **c_cria_regra_vot –** cria no ambiente a estrutura inicial que servirá para o armazenamento das regras de votação configuradas dentro de uma norma. Esta operação só pode ser executada por facilitadores.

6.9.2 Operações Modificadoras

- **m_inclui_turno** – inclui novo turno de votação para uma regra de votação.

- **m_exclui_turno** – remove um determinado turno configurado na regra de votação.
- **m_altera_data_inicio** – modifica data de inicio prevista configurada para um turno.
- **m_altera_data_fim** – modifica data de termino prevista configurada para um turno.
- **m_altera_metodo_votação** – altera o método que será usado em um determinado turno configurado na regra de votação.

6.9.3Operações Observadoras

- **o_existe_turno** – verifica se existe um turno configurado na regra de argumentação.
- **o_metodo_turno_votação** – recupera o método que será utilizado para cômputo dos votos em um determinado turno.
- **o_existe_regra_votação** - verifica se já existe uma regra de votação configurada para uma Norma.
- **o_data_prevista_inicio_votação** – recupera a data prevista para inicio de uma atividade de votação configurado em uma norma.
- **o_data_prevista_termino_votação** – recupera a data prevista para termino de uma atividade de votação configurado em uma norma.
- **o_numero_turnos** – recupera numero de turnos configurados para a votação.

6.10 ATO Regras Extração (REGRAS_EXT)

O ATO Regras_Ext permite a criação e configuração de diretrizes para atividade de extração de um processo de decisão. Cada regra para atividades de extração é composta por um facilitador, *facilitador*, um período previsto para a realização, *Data Inicio* e *Data Fim*. A classe do ATO Regras_Vot é representado na Fig. 6.10.

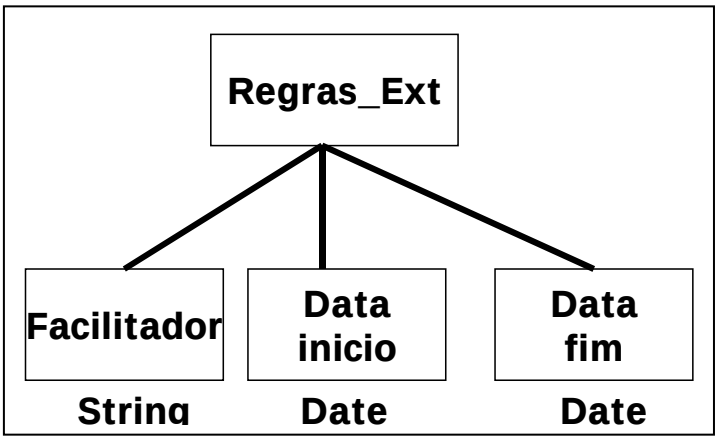


FIGURA 6.10 – ATO Regras_Ext.

6.10.1 Operações Criadoras

- **c_cria_regra_ext** – cria no ambiente a estrutura inicial que servirá para o armazenamento da regra de extração configurada dentro de uma norma. Esta operação só pode ser executada por facilitadores.

6.10.2 Operações Modificadoras

- **m_altera_facilitador** – modifica o facilitador responsável pela extração das alternativas para votação.
- **m_altera_dt_inicio** – modifica data de inicio prevista configurada para extração.
- **m_altera_dt_fim** – modifica data de termino prevista configurada para extração.

6.10.3 Operações Observadoras

- **o_facilitador_ext** – recupera o facilitador responsável pela extração
- **o_dt_inicio_ext** – recupera a data prevista para inicio de uma atividade de extração configurado em uma norma.
- **o_dt_fim_ext** – recupera a data prevista para termino de uma atividade de extração.

6.11 ATO Decisao

O ATO Decisao permite a criação e manipulação das decisões, corresponde as ferramentas dispostas no ambiente a serem utilizadas nas atividades de decisão.

Os componentes deste ATO são:

- **IdDecisao** – uma identificação para um processo de decisão.
- **IdNorma –** corresponde a Norma configurada para uma decisão.
- **Argumentacao –** corresponde ao controle e manipulação das argumentações realizadas dentro do ambiente. Este ATO será mais detalhado mais a frente.
- **Extracao –** corresponde ao controle e manipulação das extrações realizadas dentro do ambiente pelo facilitador. Este ATO será mais detalhado adiante.
- **Votacao –** corresponde ao controle e manipulação das votações realizadas dentro do ambiente. Este ATO será mais detalhado mais a frente. Este ATO será mais detalhado adiante.
- **Palavras Chave –** armazena um conjunto de palavras chaves que poderão ser utilizadas para pesquisa dentro do ATO Decisoes. Este ATO será mais detalhado adiante.

Ao ATO Decisoes estão associadas as ferramentas, comentadas no capítulo anterior, Editor de Argumentação, Editor de Extração e Editor de Votação, respectivamente relacionados ATO Argumentacao, ATO Extracao e ATO Votacao. A classe do ATO Decisoes é representado na Fig. 6.11.

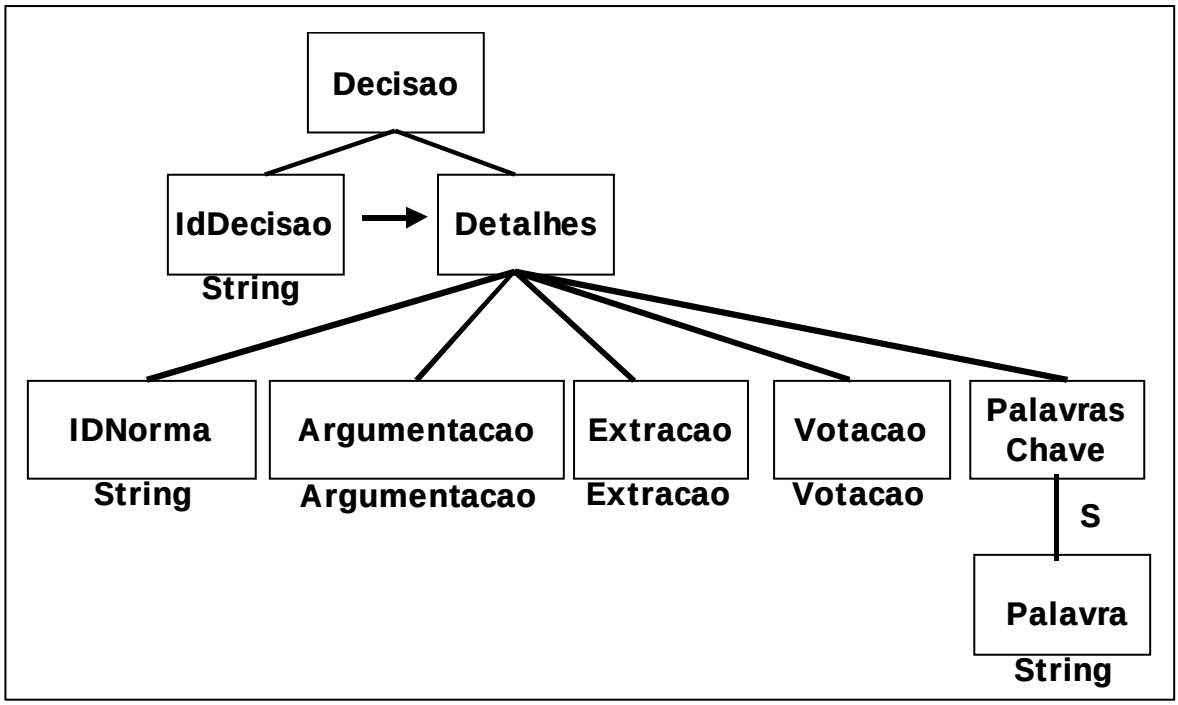


FIGURA 6.11 – ATO Decisao.

6.11.1 Operações Criadoras

- **c_cria_decisao** – cria no ambiente a estrutura inicial que servirá para o armazenamento dos decisões realizadas dentro do ambiente SaDg. Esta operação é executada após a geração do processo de decisão a partir da Norma previamente configurada.

6.11.2 Operações Modificadoras

- **m_inclui_decisao** – cadastra um processo de decisao no ambiente.
- **m_exclui_decisao** – remove um processo de decisao do ambiente.
- **m_altera_idnorma** – modifica norma configurada para o processo de decisao.
- **m_altera_ arg** – altera a argumentação.
- **m_altera_ext** – altera a extração.
- **m_altera_ vot** – altera a votação.
- **m_inclui_pal_chave** – adiciona as palavras chaves que servirão como base para a recuperação de informações dentro do SaDg.
- **m_exclui_pal_chave** – remove uma palavra chave.

6.11.3 Operações Observadoras

- **o_norma_decisao –** recupera a norma configurada para uma decisão.
- **o_decisao_palavra_chave** – recupera as decisões cadastradas para uma determinada palavra chave de pesquisa.
- **o_palavras_chave** – recupera as palavras chaves definidas para uma decisão.
- **o_existe_decisao** – verifica se já existe cadastrada uma determinada decisão.
- **o_existe_norma_decisao** – verifica se existe uma norma que orienta a decisão.
- **o_existe_arg_decisao** – verifica se existe argumentações para uma determinada decisão.
- **o_existe_ext_decisao** – verifica se existe extrações para uma determinada
- **o_existe_arg_decisao** – verifica se existe votações para uma determinada

6.12 ATO Argumentacao

O ATO Argumentacao permite a criação e manipulação de elementos de discussão para um processo de decisão. Este ATO controla todas as discussões realizadas dentro de uma decisão, bem como sua data de realização. A classe do ATO Argumentacao é representado na figura 6.12.

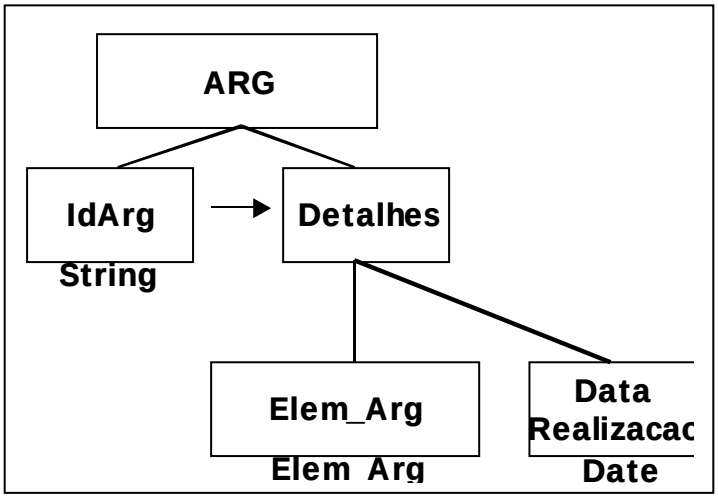


FIGURA 6.12 – ATO Argumentacao.

6.12.1 Operações Criadoras

- **c_cria_arg** – cria no ambiente SaDg a estrutura para armazenar as discussões/argumentações realizadas durante uma atividade de argumentação.

6.12.2 Operações Modificadoras

- **m_inclui_arg** – inclui um mapeamento para o registro de todas as discussões a serem realizadas para um processo de decisão.

- **m_exclui_arg** – exclui um mapeamento de argumentação.
- **m_inclui_elem_arg** – inclui um elemento de argumentação na discussão.
- **m_altera_elem_arg** – altera um elemento de argumentação já inserido no mapeamento de argumentação.
- **m_exclui_elem_arg** – exclui um elemento de argumentação da discussão.
- **m_relaciona_elementos** – define uma relação entre elementos de discussão.
- **m_altera_dt_realizacao** – modifica a data de realização da atividade de argumentação.

6.12.3 Operações Observadoras

- **o_existe_arg** – verifica se já existe criado um mapeamento para uma atividade de argumentação.
- **o_pesquisa_arg** – retorna um conjunto com elementos de argumentação que possuem uma mesma palavra-chave(rotulo).
- **o_participa_arg** – recupera um conjunto com todos os usuários participantes de uma atividade de argumentação.
- **num_contribuições_usuario** – recupera o numero de elementos contribuídos na discussão por determinado usuário.
- **o_tot_elem_arg** – recupera o total de elementos contribuídos durante a discussão.
- **o_tot_posicoes** – recupera o total de posições inseridas na discussão.
- **o_tot_questoes** – recupera o total de questões inseridas na discussão.
- **o_tot_argumentos -** recupera o total de argumentos inseridas na discussão.

6.13 ATO ELEM_ARG

O ATO ELEM_ARG permite a criação e manipulação das discussões a serem realizadas dentro do ambiente. A Fig 6.13 apresenta a Classe do ATO ELEM_ARG.

Os componentes deste ATO são:

- **IdElem** – identificação para um elemento de argumentação.
- **IdUsuario –** identificação do agente dentro da discussão.
- **Tipo Elemento –** identifica um tipo de elemento de argumentação podendo ser: um questão ou uma posição ou um argumento.
- **Rotulo Elemento –** uma marcação para facilitar a busca de informações dentro do ATO. Corresponde a uma palavra chave. (e.g. Engenharia de Software).
- **Texto Elemento –** uma descrição textual do elemento de argumentação.
- **Data Apresentacao –** a data de apresentação de um elemento.

- **Relacao –** os elementos de argumentação estão associados entre si.

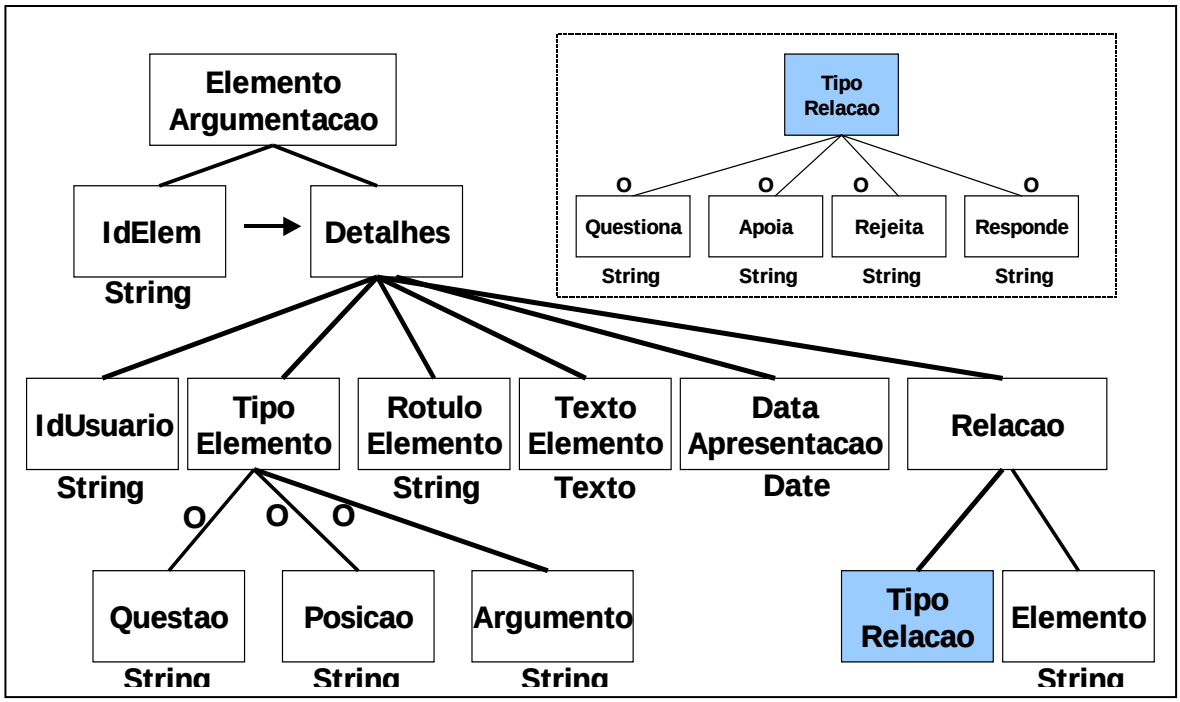


FIGURA 6.13 – ATO ELEM_ARG

A relação entre os elementos de argumentação permite o desenho da rede de discussões segundo o modelo IBIS. As relações estão assim desenhadas:

TABELA 6.1 – Relação entre elementos de argumentação.

Elemento 1	**Elemento2**	**Relação**
Questão	Posição	Questiona
Argumento	Posição	Apoia
Argumento	Posição	Rejeita
Questão	Argumento	Questiona
Posição	Questão	Responde

6.13.1 Operações Criadoras

- **c_cria_elem_arg** – cria no ambiente a estrutura que servirá para o armazenamento dos nodos de discussões realizadas dentro do ambiente SaDg.

6.13.2 Operações Modificadoras

- **m_inclui_elem_questao** – insere um nodo questão na estrutura de discussão. (problemas em discussão) .

- **m_inclui_elem_posicao** – insere um nodo posição na estrutura de discussão. (possíveis soluções) .
- **m_inclui_elem_argumento** – insere um nodo argumento na estrutura de discussão. (opiniões favoráveis ou não) .
- **m_exclui_elem_arg** – remove um elemento de argumentação da estrutura de discussão.
- **m_relacao_elem_arg** – defini uma relação entre elementos de argumentação.
- **m_altera_elem_arg_aux** - operação auxiliar, interna ao ATO ELEM_ARG.

6.13.3 Operações Observadoras

- **o_existe_id_elem** – verifica se já existe um id definido para um determinado elemento de argumentação.
- **o_ids_elems** – recupera um conjunto com todos os elementos de argumentação inseridos na estrutura de discussão.
- **o_e_questao** – verifica se o elemento de argumentação é um elemento do tipo questão.
- **o_e_posicao** - verifica se o elemento de argumentação é um elemento do tipo posição.
- **o_e_argumento** - verifica se o elemento de argumentação é um elemento do tipo argumento.
- **o_ids_questoes** – recupera um conjunto com todas as questões contribuídas na discussão.
- **o_ids_posicoes** – recupera um conjunto com todas as posições contribuídas na discussão.
- **o_ids_argumentos** – recupera um conjunto com todos os argumentos contribuídos na discussão.
- **o_texto_questao** – recupera o texto descritivo de uma questão.
- **o_texto_argumento** – recupera o texto descritivo de um argumento.
- **o_texto_posicao** – recupera o texto descritivo de uma posição.
- **o_argumentos_favor_posicao** – recupera um conjunto com todos os argumentos que apoiam uma determinada posição.
- **o_argumentos_contra_posicao** - recupera um conjunto com todos os argumentos que rejeitam uma determinada posição.
- **o_posicoes_respondem_questao** – recupera um conjunto com todos as posições que respondem uma questão.
- **o_questoes_questionam_posicao** – recupera um conjunto com todos as questões que questionam uma posição.
- **o_questoes_questionam_argumento** – recupera um conjunto com todos as questões que questionam um argumento.

- **o_elementos_argumentaçao_usuario** – recupera um conjunto com os elementos de argumentação contribuídos por um determinado usuário.
- **o_elementos_argumentacao_datapres** – recupera um conjunto com os elementos de argumentação contribuídos numa determinada data de apresentação.
- **o_elementos_argumentacao_rotulo** - recupera um conjunto com os elementos de argumentação marcados por um determinado rotulo.
- **o_elemento_argumentacao_usuario_existe_aux** – operação auxiliar interna ao ATO ELEM_ARG.
- **o_elemento_argumentacao_datapres_existe_aux** – operação auxiliar, interna ao ATO ELEM_ARG.
- **o_elemento_argumentacao_rotulo_existe_aux** - operação auxiliar, interna ao ATO ELEM_ARG.

6.14 ATO EXTRACAO

O ATO EXTRACAO permite a criação e manipulação das extrações realizadas para um processo de decisão. Os componentes deste ATO são:

- **IdExt** – uma identificação de uma atividade de extração.
- **Data Realizacao** – data de realização da atividade.
- **Alternativas** – corresponde aos elementos de discussão selecionados pelo facilitador para o a atividade de votação.
- **Descricao** – corresponde a uma descrição textual sobre a atividade.

A figura 6.14 abaixo apresenta a classe deste ATO.

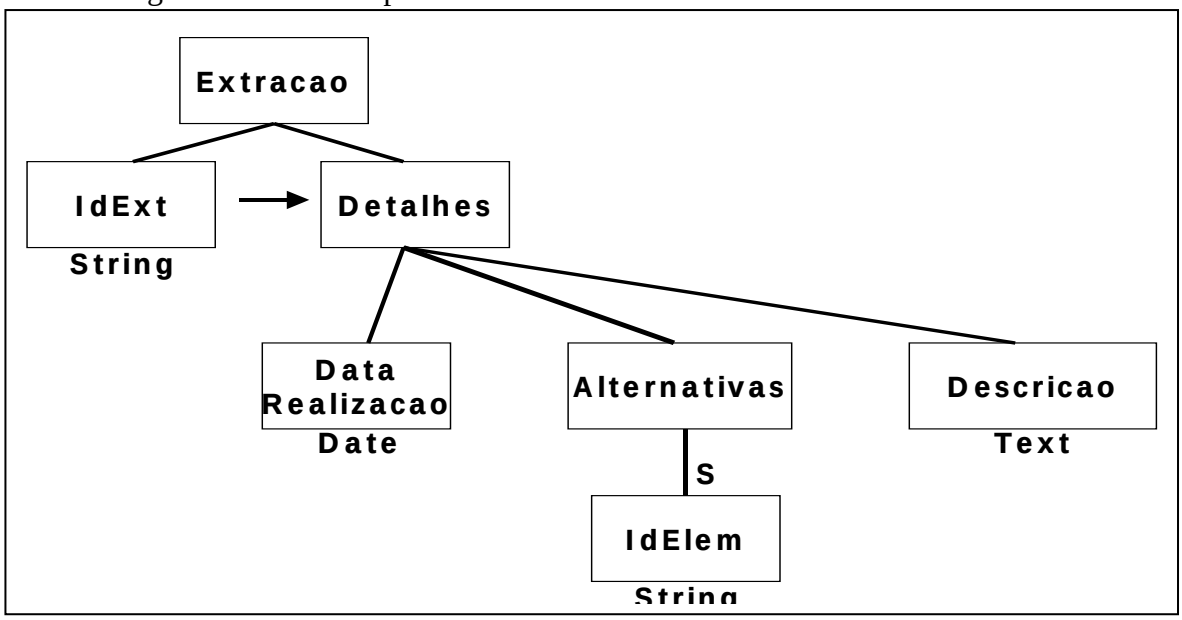


FIGURA 6.14 – ATO EXTRACAO.

6.14.1 Operações Criadoras

- **c_cria_extração** – cria no ambiente a estrutura inicial que servirá para o armazenamento das alternativas extraídas do processo de discussão, e que servirão de **alternativas** para o processo de votação.

6.14.2 Operações Modificadoras

- **m_inclui_extracao** – inclui um mapeamento de extração no ambiente.
- **m_exclui_extracao** – exclui um mapeamento de extração no ambiente.
- **m_insere_alternativa_extracao** – insere uma alternativa no mapeamento de extração.
- **m_remove_ alternativa_extracao** – remove uma alternativa do mapeamento de extração.
- **m_altera_descricao_extracao** – modifica a descrição de um mapeamento de extração.
- **m_altera_datare_extracao** – modifica a data de realização de um mapeamento de extração.

6.14.3 Operações Observadoras

- **o_pode_ser_alternativa** – verifica se um determinado identificador de um elemento de discussão pode usado como alternativa.
- **o_existe_id_extracao** – verifica se existe um id já configurado no processo de extração.
- **o_id_alternativas** – recupera um conjunto com todas as alternativas selecionadas dentro de um determinado mapeamento de extração.
- **o_id_alternativas_datare** – recupera as alternativas selecionadas dentro de uma determinada data de realização.

6.15 ATO VOTACAO

O ATO Votação permite a criação e manipulação de votos para um processo de decisão. Este ATO controla todas as votações realizadas dentro de uma decisão. A classe do ATO Argumentacao é representado na Fig. 6.15.

Os componentes deste ATO são:

- **IdVot** – corresponde a identificação de um voto.
- **Turno** – turno ao qual o voto se relaciona.
- **Escolha** – define a escolha ou não participação por uma determinada alternativa.
- **Data Voto** – a data de apresentação do voto.
- **Vencedor Voto** – após o cômputo dos votos, marca o voto vencedor.

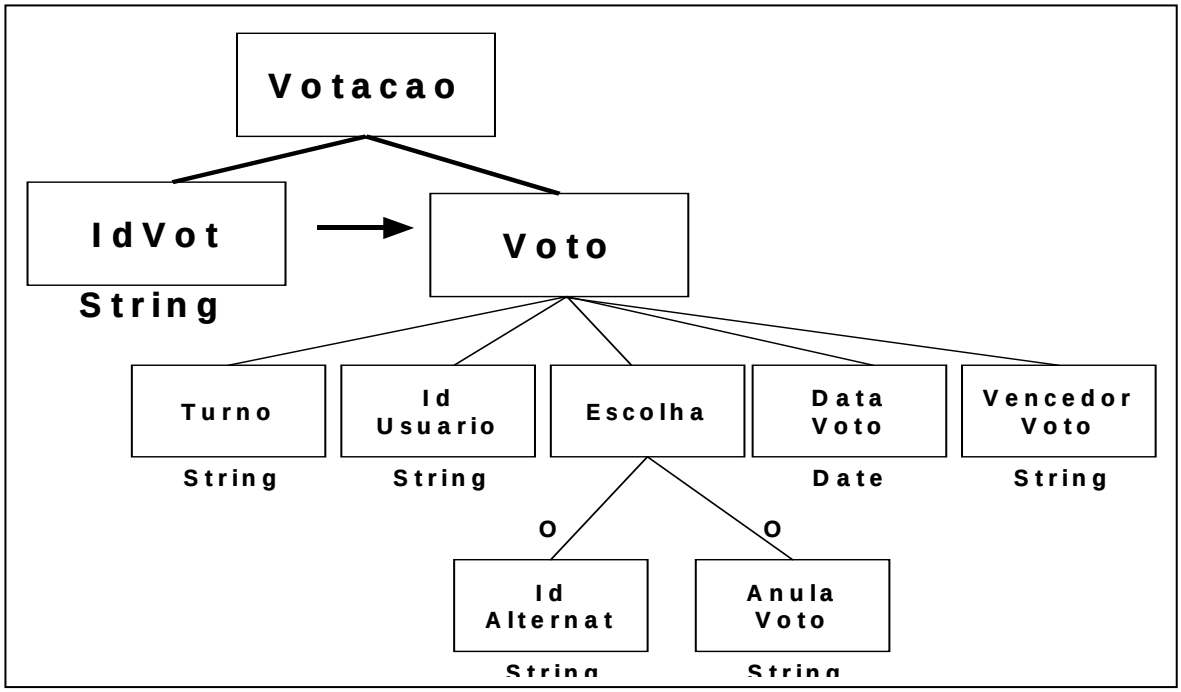


FIGURA 6.15 – ATO VOTACAO.

6.15.1 Operações Criadoras

- **c_cria_votacao**– cria no ambiente a estrutura inicial que servirá para o armazenamento dos votos.

6.15 .2 Operações Modificadoras

- **m_insere_voto** – adiciona um voto dentro do mapeamento de votação.
- **m_anula_voto** – anula um voto.
- **m_remove_voto** – exclui um voto dentro do mapeamento de votação.
- **m_computa_votos** – realiza o calculo dos votos.
- **m_metodo_pluralidade** – realiza calculo baseado no método de pluralidade.

- **m_metodo_bordaK** – realiza calculo baseado no método Borda Kendal.
- **m_metodo_nai** – realiza calculo baseado no método NAI.

6.15.3 Operações Observadoras

- **o_existe_id_voto** – verifica se existe voto.
- **o_alternativas_votadas** – recupera um conjunto com todas as alternativas votadas.
- **o_votos_anulados** – recupera um conjunto com todos os votos anulados.
- **o_votos_turno** – recupera um conjunto com todos os votos de um determinado turno.
- **o_votos_data** – recupera um conjunto com todos os votos em uma determinada data.
- **o_votos_usuario** – recupera um conjunto com votos de um determinado usuário.

7 Utilização do Modelo SaDg em outros Domínios

Durante os últimos anos, tem-se assistido uma investigação cada vez maior na área de trabalho cooperativo, em particular no que diz respeito à utilização de sistemas de apoio à decisão em grupo (SADG). Os SADGs combinam tecnologias de informação, de comunicação e decisão, para suporte de formulação e resolução de problemas em reuniões em grupo.

Reuniões são o meio mais usual das organizações discutirem seus problemas e tomar decisões. As pessoas acostumadas ao ambiente das empresas tem conhecimento de que este mecanismo é bastante adotado e por muitas vezes mal utilizado, tendo-se a sensação que as reuniões são improdutivas. Os motivos são vários e nem sempre fáceis de serem eliminados. Entretanto, mesmo em organizações modernas em que o computador, quase sempre conectado em rede, é um instrumento usual de trabalho, pouco suporte computacional existe para apoiar o entendimento, a discussão ou a decisão em grupo, objetivos básicos das reuniões.

Uma constatação referente aos aspectos sociais é que, dependendo do ambiente/cenário, os participantes tem comportamento e atitudes distintas. Portanto, a disciplina de divisão de papéis e de fundamental importância para coordenação do processo, bem como a adaptação destes ao recurso computacional utilizado para apoio a tomada de decisão.

O modelo SADG proposto neste trabalho apresenta uma divisão simples de papéis, mas poderão ser perfeitamente estendidos para complementar o cenário em que serão simulados. Além dos papéis, as etapas ou fases para se alcançar os objetivos, devem estar bem desenhadas dentro do processo.

A idéia e desenvolvimento deste modelo parte da premissa básica proposta no modelo de Simon [SIM 60]. O modelo de Simon é composto de quatro fases, que não necessariamente devem ocorrer seqüencialmente, sendo elas: inteligência, projeto, escolha e revisão. Cada fase pode constituir em si própria um processo decisório, sendo subdividida em fases, e assim recursivamente.

O modelo de Simon apresenta um estrutura simples, porém acredita-se que estabelecendo fases para o processo decisório se torna mais fácil identificar em que momento da evolução do processo se está localizado, bem como o tipo de apoio que pode ser proporcionado para torná-lo mais eficaz.

Realizando um paralelo com o modelo proposto, pode-se observar que as principais etapas apresentadas por Simon são contempladas. Fazendo-se uma ressalva apenas à etapa de revisão, cuja qual não é formatada dentro do modelo. Entretanto, esta etapa não é citada em discussões sobre o processo de tomada de decisão porque, após a escolha ter sido realizada, na maioria dos casos, não há outro processo a ser tratado na tomada de decisão.

Acreditamos, que independente do domínio a ser analisado para tomada de decisão, existem alguns propósitos comuns, e podem perfeitamente ajudar no processo como um todo, como:

- Compartilhar informações entre os participantes;
- Localizar um problema e defini-lo;
- Desenvolver novas idéias sobre o problema;
- Revisão de situações tais como: análise de variação sobre resultados financeiros ou adesão de plano e orçamento para um projeto particular, ou passos de análise e verificação de sistemas de informação;
- Revisão e aprovação (ou rejeição) de propostas tais como orçamento de capital;
- Tomada preliminar de escolhas de mais alto nível nas organizações tais como: um grupo de seleção para verificar três candidatos preferidos para uma posição;
- Negociação entre organizações (externar ou interna);
- Resolução de conflitos;
- Gerência de crises;
- Seleção formal, por votação, entre um conjunto de alternativas propostas;
- Planejamento.

Nas seções seguintes serão apresentados dois domínios distintos, suas características e quais os benefícios esperados quando apoiados por uma ferramenta SADG, utilizando como exemplo o modelo proposto neste trabalho.

7.1 Focus Group

O *Focus Group*, método qualitativo de pesquisa utilizado inicialmente pela sociologia, tem sido atualmente bastante empregado pela área de marketing, e começa a ganhar espaço, também , em diversas outras áreas, como educação, saúde, gestão, planejamento, decisão e sistemas de informação, entre outras.

Dependendo do objetivo da pesquisa, o *Focus Group* pode ser utilizado sozinho ou associado a outros métodos. Os resultados obtidos com a sua aplicação são particularmente efetivos em fornecer informações de como as pessoas pensam, sentem ou mesmo permitem depreender a forma como agem em relação a um tópico.

O *Focus Group* é um tipo de entrevista em profundidade realizada em grupo, cujas reuniões apresentam características quanto à proposta, tamanho, composição e procedimentos de condução. O foco ou objetivo da análise é a interação dentro do grupo. Os participantes influenciam uns aos outros através das repostas às idéias e colocações durante a discussão, estimulados por comentários ou questões que são fornecidas pelo moderador (pesquisador ou outra pessoa). Os dados fundamentais produzidos por esta técnica são transcritos das discussões do grupo, acrescidos das anotações e reflexões do moderador e de outro(s) observador(es), caso exista(m).

As características gerais do *Focus Group* são:

- envolvimento do pessoas;
- reuniões em série;
- homogeneidade dos participantes quanto a aspectos de interesse da pesquisa;
- geração de dados;
- natureza qualitativa;
- e discussão focada em um tópico, o qual é determinado pelo propósito da pesquisa.

Pode-se dividir o *Focus Group* em três etapas: **planejamento**; **condução das entrevistas** e **análise dos dados**. O planejamento é crítico para o sucesso do método, pois nesta fase o pesquisador considera a intenção do estudo e os usuários da informação, além de desenvolver um plano que guiará o restante do processo da pesquisa, incluindo a elaboração das questões e a seleção dos participantes. A fase de condução consiste na moderação das reuniões e, após as sessões, na fase de análise, realizam-se as transições, o tratamento dos dados e a elaboração do relatório.

No planejamento do *Focus Group*, deve-se desenvolver um plano cronológico, incluindo as seguintes atividades: desenvolvimento das questões, identificação das características dos participantes, obtenção da lista dos potenciais participantes, recrutamento dos participantes, realização das reuniões, obtenção da lista dos potenciais participantes, recrutamento dos participantes, realização das reuniões, *feedback* do planejamento, transcrição e análise do relatório.

Quanto ao número de participantes das sessões, a conclusão usual é utilizar grupos de tamanho moderado, os quais seriam constituídos de 6 a 10 pessoas. Em marketing, os pesquisadores são favoráveis a grupos de entre 6 e 8 pessoas, enquanto anos atrás eram entre 8 e 10 pessoas. Na definição do tamanho do grupo, deve-se ponderar para que o mesmo seja pequeno suficiente para que todos tenham oportunidade de partilhar suas percepções e grande o suficiente para permitir diversidade de percepções.

A determinação de quem participará do estudo é função do propósito da pesquisa. Além disso, deve-se considerar a necessidade de segmentar as pessoas em categorias, em função de fatores tradicionais, tais como localização geográfica, idade, tamanho da família, status, sexo, etc.

Quanto ao nível de envolvimento do moderador é sempre tratado como *continuum:* num extremo é baixo, onde tem um pequeno papel em fazer a discussão do grupo progredir e observar para que seus comentários sejam não diretivos tanto quanto possível; e no outro extremo é alto, onde o moderador controla os tópicos que são discutidos e a dinâmica da discussão.

Com relação ao conteúdo das entrevistas [MOR 88], os aspectos a serem observados são:

- **campo** – grupos com sucesso discutem um campo (área) de tópicos que não só cobre as questões que os pesquisadores já sabiam, mas também introduz um grupo de questões que os pesquisadores não tinham antecipado;
- **especificidade** – direcionar as discussões do *Focus Group*, para contribuições detalhadas e concretas das experiências dos participantes;
- **profundidade** – assegurar o envolvimento dos participantes com o material que eles estão discutindo;
- **contexto pessoal –** obter observações que dêem ao pesquisador um entendimento das perspectivas dos participantes sobre o tópico de interesse. O contexto pessoal pode ser baseado no papel social e categorias que os participantes ocupam ou pode ser enraizado nas experiências individuais.

Em relação as respostas, estas estão diretamente ligadas à qualidade das questões. As perguntas são a essência da entrevista do *Focus Group*. Elas devem parecer espontâneas para os participantes, mas devem ter sido cuidadosamente selecionadas e elaboradas em função da informação esperada. Tipicamente, uma entrevista incluirá cerca de 12 questões. As questões podem ser classificadas nas seguintes categorias [KRU 94]:

- **questões abertas** – primeira rodada de perguntas da sessão é feita a todos, de forma a permitir uma resposta rápida (10 a 20 segundos), e permite identificar características que os participantes têm em comum;
- **questões introdutórias –** introduz o tópico geral da discussão e fornece aos participantes uma oportunidade para refletir sobre experiências anteriores;
- **questões de transição –** estas questões movem a conversação para questões chave que norteiam o estudo;
- **questões chave –** direcionam o estudo, normalmente variam de 2 a 5 questões, são as que requerem maior atenção e análise;
- **questões finais –** fecham a discussão, consideram tudo o que foi dito até então. Permitem aos participantes considerar todos os comentários partilhados na discussão e identificar quais os aspectos mais importantes. Exemplo: “de todas as necessidades que nós discutimos, qual a mais a importante para vocês?”;
- **questões resumo –** o moderador faz um resumo de 2 ou 3 minutos das questões chave e grandes idéias que emergiram da discussão.
- **questão final** - questão padronizada perguntada ao final do *Focus Group.*

O uso prévio de *brainstorming* com colegas e usuários da informação pode ser útil para obtenção de questões e variações no vocabulário.

Por fim, a análise dos dados é um trabalho lento e consome muito tempo. Dependendo do número de grupos, da disponibilidade dos participantes e do tipo de análise pretendida para as transcrições, pode-se levar até 6 meses. Esta tarefa é dificultada ainda mais, visto que a discussão de grupo é conduzida várias vezes,

normalmente um mínimo de 3, com tipos específicos de participantes para identificar tendências e padrões na percepção dos participantes. Cuidadosa e sistemática análise das discussões fornecem sinais e percepções de como um produto, serviço ou oportunidade é percebida.

Na seção seguinte, apresentaremos como poderia ser incorporado a utilização do modelo SaDg proposto dentro deste método de pesquisa.

7.1.1 Utilizando o modelo SaDg em *Focus Group*

Dividindo as etapas do processo em atividades, segundo o modelo SADG proposto podemos observar:

- **Definição do Problema** – Através de todas as informações levantadas sobre o serviço ou produto desenha-se o problema, ou seja os tópicos das discussões do *Focus Group*.
- **Configuração de uma Norma –** através da norma são dividas as atividades dentro do processo decisório para solução dos tópicos. As atividades de extração e votação não precisam ser configuradas na Norma porque, o caráter do processo é de *brainstorming*, ou seja, todas as idéias são levadas em consideração no estudo final.
- **Atividade de Argumentação –** Através desta atividade e utilizando a ferramenta disposta no modelo, o Editor de Argumentações os participantes poderão contribuir com suas questões, posições e argumentos. O resultado da rede de discussões gerado durante a atividade, ajudam tanto na preparação das perguntas para entrevista – pré *Focus Group*, quanto para elaboração do relatório final.

Algumas observações quanto a configuração do processo:

- A configuração dos problemas de decisões, são avaliados como tópicos de discussão dentro do *Focus Group*.
- Para coordenação, neste tipo de domínio, será utilizado apenas o papel do coordenador, chamado de moderador dentro do *Focus Group*.
- Não é mais necessário a presença física dos participantes, todos terão um computador para participar do processo.
- Para efeito de busca e preparação do relatório final, seria importante, linkar palavras chaves da discussão dentro dos rótulos dos elementos.

Como já comentado anteriormente, as principais etapas da aplicação do método de Focus Group são: planejamento, condução das entrevistas e análise dos dados. Todas estas etapas podem ser configuradas dentro do modelo SaDg, como segue na Tab. 7.1:

TABELA 7.1 Etapas Focus Group x Modelo SaDg.

Etapas Focus Group	**Modelo SaDG**
Planejamento	Seguindo o fluxo de ações necessárias para a configuração dos tópicos da discussão: **Editor de Problemas** – para configuração dos tópicos da discussão; **Editor de Normas** – para configuração do tempo, participantes e atividades necessários para execução do Focus Group.
Condução das entrevistas	Seguindo o fluxo de ações para a tomada de decisão: **Editor de Argumentação** – todas as contribuições dos participantes. Novas questões, aprovações e rejeições quanto aos tópicos e alternativas. Todo a rede de informações geradas na discussão.
Análise dos Dados	Seguindo o fluxo de ações para a análise da discussão: Existe diversas operações dentro do modelo que permitem busca sobre todas as informações geradas dentro da rede de discussão. Isto pode facilitar o processo de geração do relatório, seja através de busca sobre a própria decisão, quanto aos elementos de argumentações e seus rótulos de pesquisa.

Como podemos notar na Tab. 7.1 nem todos os recursos do Modelo SaDg são utilizados para aplicação do *Focus Group*. De qualquer forma, qualquer alteração pode ser realizada de forma a melhor enriquecer o processo.

A avaliação da viabilidade de aplicação do modelo sobre este domínio esta em fase de desenvolvimento em uma empresa estabelecida em Porto Alegre (RS) que atua na área de marketing. Alguns serviços do modelo estão sendo alterados para melhor incorporarem as fases do método, ou seja, a adequação da ferramenta ao problema e não o contrário. O *Focus Group* é utilizado nesta empresa para análise de prestação de serviços de uma empresa de TV por assinatura.

Acredita-se que com os recursos oferecidos por um modelo SaDg será possível guardar todo o registro do processo de forma estruturada; recuperar quando necessário informações sobre um tópico específico para o enriquecimento e execução de outros *Focus Groups;* coordenar através de uma estrutura mais rígida (Norma) a discussão de forma a "focalizar mais o processo" e por fim, as justificativas para recomendações de uma determinada ação é assegurado. A Fig. 7.1 dar uma visão geral quanto a aplicação do modelo neste contexto.

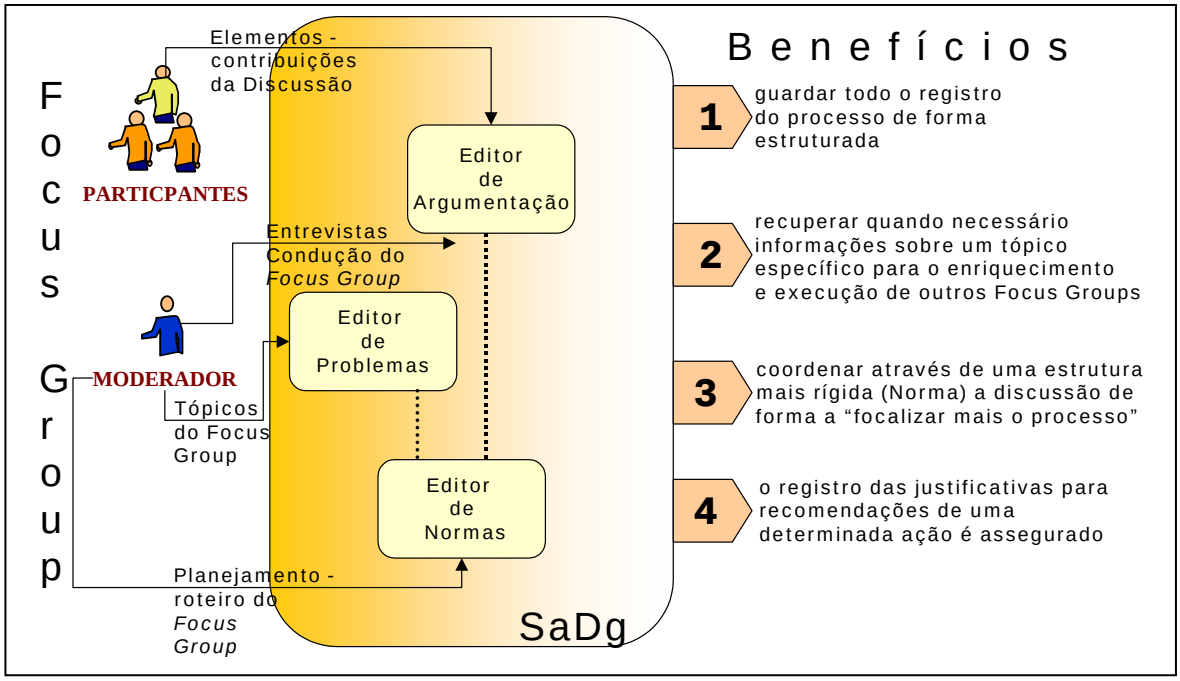


FIGURA 7.1 – Aplicação do modelo SaDg em *Focus Group*.

7.2 Ensino à Distancia

O processo de ensino à distância envolve diferentes estados de interação entre o formador e os formandos e os formandos entre si. Existem estados onde o tutor ministra sem interrupções uma sessão de ensino à distância, e outros onde se estabelece uma discussão aberta envolvendo o tutor e os formandos.

Na Fig. 7.2 abaixo podemos observar um modelo dos estados e transições que ocorrem no ensino à distância [MAR 99].

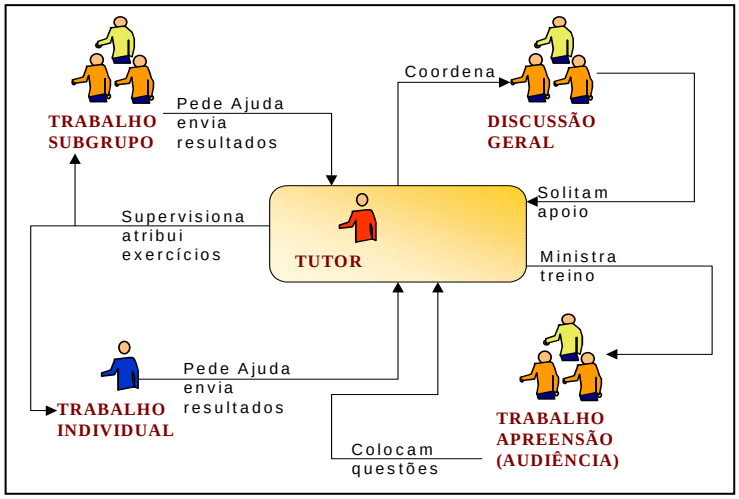


FIGURA 7.2 - Estados e Transições no Ensino à Distância.

Inicialmente o tutor ministra uma sessão de ensino ou treino para uma audiência. O trabalho individual ou de subgrupos resulta da atribuição de exercícios e divisão de tarefas entre formandos e estes entre si. Neste momento, torna-se essencial um controle remoto efetivo do trabalho ou pelo menos o acompanhamento efetuado localmente. Os formandos necessitam nestas fases de trabalho individual ou de subgrupo, de um apoio quase constante do tutor ou monitor local.

A existência de ferramentas de compartilhamento das aplicações a serem ensinadas e treinadas, facilita o processo de explicação e esclarecimento, podendo o tutor reforçar o seu discurso com ações sobre estas.

Assim o processo de ensino e treino (ciclo de ensino) considera as seguintes fases:

- **Ensino e treino** ministrado pelo tutor para uma audiência remota de formandos. Estes podem realizar intervenções esporádicas, desde que autorizadas pelo tutor;
- **Trabalho individual** do formando, que acontece quando este executa o curso de teletrabalho ou simplesmente realizam exercícios atribuídos pelo tutor. O formando pode sempre que possível contactar e compartilhar remotamente com o tutor os resultados ou colocar questões;
- **Trabalho de subgrupo**, acontece quando um grupo realiza exercícios em conjunto. Ferramenta de trabalho cooperativo e compartilhamento de aplicações e dados podem ser usadas durante esta fase para o apoio ao trabalho do grupo. O tutor poderá ser também acionado;
- **Discussão geral**, que acontece quando o tutor decide trazer para turma um tema comum para análise. Este processo é coordenado pelo tutor

Neste contexto de trabalho cooperativo que envolve controle das motivações e discussões realizadas durante o aprendizado, a utilização de ferramentas para o suporte, coordenação e registros das argumentações apresentadas durante o ensino se faz ainda mais necessário.

Na seção seguinte apresentaremos como poderia ser aplicado o modelo SaDg no contexto de ensino à distância para suporte as atividades de trabalho em subgrupo, discussão geral, bem como coordenação e apoio às questões levantadas durante o treino e aprendizado.

7.2.1 Utilizando o modelo SaDg em Ensino à Distância

Imaginemos uma situação onde o tutor apresenta um tema para ser discutido em profundidade e onde os participantes contribuem no processo de forma a gerar uma rede de discussões assertivas sobre cada tema. A idéia é que para análise do tutor ele possa captar o entendimento dos alunos perante ao tema, e também, formatar melhores abordagens para a condução destes temas com outros grupos.

Dividindo as etapas do processo em atividades, segundo o modelo SADG proposto podemos observar:

- **Definição do Problema** – Através de todas as informações levantadas sobre os temas aos quais o tutor precisa discutir, desenha-se o problema, ou seja os tópicos das discussões.
- **Configuração de uma Norma –** através da norma são dividas as atividades dentro do processo decisório para solução dos temas. As atividades de extração e votação não precisam ser configuradas na Norma porque, o caráter do processo é de *brainstorming*, ou seja, todas as idéias são levadas em consideração no estudo final.
- **Atividade de Argumentação –** Através desta atividade e utilizando a ferramenta disposta no modelo, o Editor de Argumentações os participantes poderão contribuir com suas questões, posições e argumentos. O resultado da rede de discussões gerado durante a atividade, ajudam na reflexão dos conteúdos apresentados e na fixação dos temas pelos participantes. Facilita também o processo contínuo do aprendizado, onde poderá ser recuperadas estas informações em novas discussões.

Algumas observações quanto a configuração do processo:

- A configuração dos problemas de decisões, são avaliados como temas de discussão dentro do ensino à distância.
- Para coordenação, neste tipo de domínio, será utilizado apenas o papel do coordenador, chamado de tutor.
- Para efeito de busca e análise, seria importante, linkar palavras chaves da discussão dentro dos rótulos dos elementos.

Acreditamos que com a utilização deste modelo SaDg, seria possível enriquecer o processo de ensino. A partir do registro destas discussões é possível realizar uma análise mais rica quanto ao entendimento do conteúdo aplicado, assim como a própria dinâmica e participação dos formandos.

Na seção seguinte será apresentado algumas considerações finais.

7.3 Considerações Finais

À medida que começam a existir evidências de que os SADG podem trazer grandes vantagens, torna-se também importante investigar o efeito da utilização destes sistemas.

Uma experiência realizada por Easton etl al [EAS 92] comparou os resultados de grupos de 4 pessoas que utilizaram ou um SADG não interativo (um computador por grupo) ou um SADG interativo (4 computadores ligados em rede por grupo). As conclusões a que chegaram foram:

- Os grupos interativos geraram uma maior número de idéias e tiveram uma maior igualdade de participação que os grupos não interativos.
- Os grupos interativos mostraram-se menos satisfeitos com o processo de decisão e os resultados alcançados que os grupos não interativos. Em particular, sentiram-se bastante frustados com a fase de votação.
- Apesar de ter existido muita investigação nesta área dos modelos SADGs nos últimos anos, os resultados ainda não são conclusivos. Muito deles acabam por ser contraditórios, não sendo possível definir por enquanto quais são as características mais importantes de um SADG. Foram definidos alguns modelos de investigação, sendo proposto por DeSanctis e Gallupe o mais seguido.
- A própria utilização dos SADGs não foi claramente demostrada como sendo uniformemente positiva, e o estudo de outros fatores e variáveis ainda não se encontra fortemente consolidado. No entanto, a meta análise de Benbasat e Lim [BEN 93] acaba por concluir que o fator tecnológico, em comparação com os outros, tem um grande impacto, e deve ser melhor investigado.

Uma investigação mais profunda destes aspectos poderá ajudar a esclarecer as inconsistências sentidas atualmente, ajudando a compreender melhor o papel dos SADGs no processo de tomada de decisão.

8 Conclusões e Trabalhos Futuros

Neste trabalho foi apresentada uma proposta de modelo de apoio a decisão em grupo para o ambiente PROSOFT, chamado SaDg PROSOFT, que visa fornecer mecanismos para apoiar as atividades de tomada de decisão, ocorridas durante o processo de desenvolvimento de software.

Para o desenvolvimento deste trabalho foi realizado um estudo a cerca do conceito SADG, suas características, bem como implementações em diferentes propostas de trabalho apresentadas no capítulo 3. Através desse levantamento foi possível definir funcionalidades para o modelo SaDg PROSOFT.

Como está intrínseco o caráter cooperativo e de tomada de decisão durante o processo de desenvolvimento, não apenas definir uma ferramenta SADG para o ambiente é importante, mas também entender a sua integração a este.

Assim como um processo de software é composto por um conjunto de atividades, com suas respectivas definições e executores, um processo de decisão também o é. Desta forma, foi possível utilizar a estrutura presente no ambiente, através do gerenciador de processos (GP)[LIM 98], para executar modelos de decisão. As atividades básicas do modelo levam em consideração fundamentos básicos como: uma atividade para planejamento, uma atividade de discussão e uma atividade de escolha.

A utilização do gerenciador de processos no apoio ao processo de decisão favorece a coordenação das atividades de decisão. A coordenação do processo decisório e essencial para que os participantes tenha “foco” nas suas atividades e na própria solução dos problemas que serão levantados no decorrer do processo de desenvolvimento. Além do gerenciador de processos, para a coordenação das atividades de decisão, foram definidos o papel de um facilitador(responsável pela definição dos requisitos mínimos para uma atividade de decisão) e uma Norma (configuração das atividades do processo decisório).

Após esse primeiro passo de definição e integração com o gerenciador de processos, foram especificadas algumas ferramentas para suportar as atividades definidas para uma processo decisório.

As ferramentas foram projetadas para facilitar a tomada de decisão, portanto, a utilização de mecanismos bastante usados em ambientes de discussão e decisão, como o modelo de argumentação IBIS e métodos de votação. Dos métodos de votação, definidos para o modelo SaDg, nem todos foram implementados sobre o paradigma do ambiente, mas acredita-se que a partir das definições das formulas apresentadas, poderão ser facilmente implementados em trabalhos futuros bem como a utilização de outros métodos.

A utilização de métodos de votação não implica em existir atividades de votação para qualquer atividade de decisão no ambiente. Por exemplo, o facilitador pode notar que em uma determinada atividade de discussão os participantes já demonstram um consenso para uma determinada alternativa, e este pode determinar a atividade de decisão como concluída.

Em linhas gerais, o modelo desenvolvido para o ambiente PROSOFT visa:

- Minimizar o nível de "ruído" na comunicação entre os desenvolvedores, através da definição de um mecanismo sistêmico para discussões, utilizando um modelo de argumentação;
- Facilitar a busca de uma solução comum, estruturando a fase de escolha, com métodos de votação;
- Fornecer uma estratégia para tomada de decisão, com mecanismo de controle que orientem o processo, bem como os participantes;
- Registrar todas as motivações e justificativas quanto aos "porquês" de cada decisão tomada durante o processo de desenvolvimento de software.

Além das contribuições expostas acima sobre o desenvolvimento deste trabalho, pode-se acrescentar a facilidade de se aplicar este modelo para contextos diferentes, assim como o verificado no capítulo 7. Este estudo proporcionou um melhor entendimento das características dispostas no modelo proposto, bem como um mecanismo para validar a aplicabilidade deste.

Uma importante contribuição a ressaltar é a especificação formal do modelo SADG proposto, que será implementado em breve no PROSOFT, o qual consiste de um modelo que serve como base para trabalhos futuros. Tais trabalhos poderão levar em consideração os históricos de decisão mantidos no ambientes para enriquecimento de modelos de processo de software, e também para o ensino de novos integrantes na equipe de desenvolvimento, quanto às melhores abordagens para determinados problemas encontrados durante o desenvolvimento de software em projetos semelhantes.

Não se pretende com este trabalho resolver todos os problemas relacionados a tomada de decisão em ambiente de desenvolvimento de software. Pretende-se sim, entender a relação deste ao processo de software, e como, estas atividades de decisão podem enriquecê-lo.

As deficiências do processo de decisão, dentro do contexto de desenvolvimento de software, são análogas às deficiências encontradas nos processos de decisão em grupo em geral. Os problemas da interação em grupo como: inibição e influências de poder e interesses pessoais, são freqüentes e de difícil solução principalmente quando o grupo possui uma organização hierárquica[BEM 93].

Os problemas de comunicação como: omissão de informações, apresentação de informações irrelevantes para o processo, divagação, negligência quanto às possíveis alternativas de solução, desperdício de tempo, etc, também aparecem com freqüência, prejudicando em muito a produtividade do projeto em desenvolvimento.

No modelo SADG desenvolvido, procurou-se reunir as características de vários modelos apresentados nos capítulos iniciais deste trabalho. Entretanto, alguns aspectos ficaram em aberto ou precisam ser melhorados.

Como trabalhos futuros, sugere-se o estudo das seguintes questões:

- Implementação do modelo, para validação dos requisitos quanto à facilitação do processo decisório;

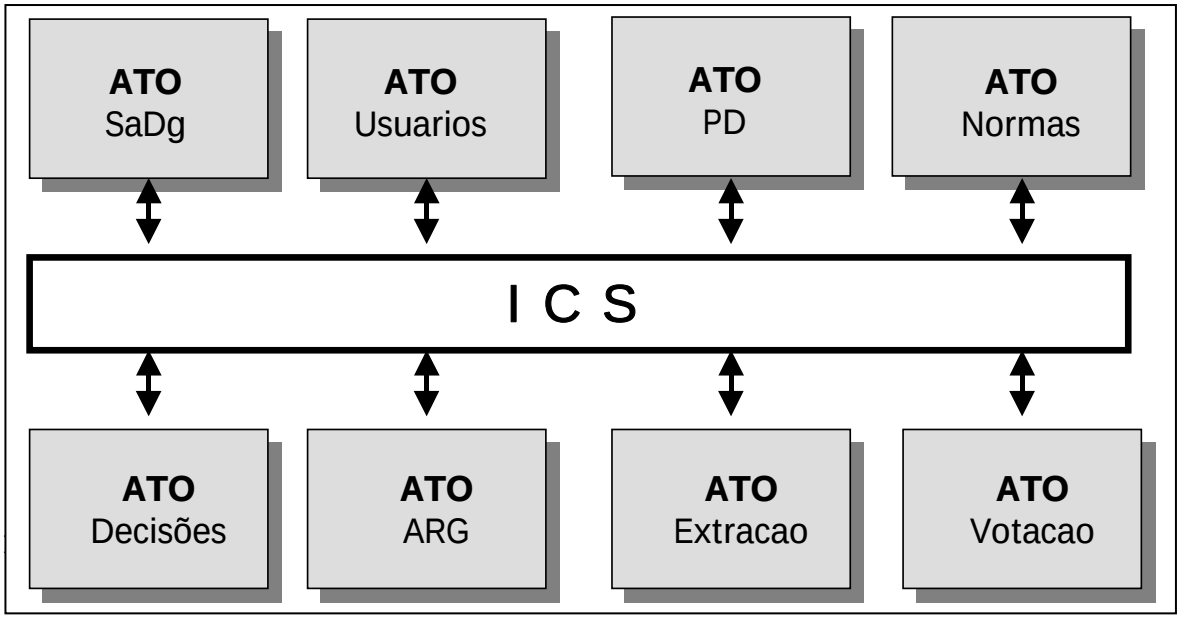
- Adição de novos métodos de votação, para que o facilitador possa ter maiores opções no momento da configuração da Norma;
- A integração e desenvolvimento de elementos de interface para utilização do modelo;
- Adição de mecanismos de busca ao processo de maneira a melhor identificar alternativas para questões baseadas no histórico de decisões anteriores;
- Implementação de mecanismos de busca para facilitar a pesquisa dentro da base de discussões;
- Adaptação do modelo SaDg, as demais ferramentas do ambiente PROSOFT, e não apenas ao Gerenciador de Processos.

Anexo 1 Operações do ATOs SaDg PROSOFT

1.1 Descrição do SaDg PROSOFT

Para o SaDg PROSOFT foram construídas 8 ferramentas, conforme pode ser visto na Figura 1.1. A especificação segue o padrão adotado por [MOR 97] para apresentação das funções definidas. As ferramentas utilizadas são:

- **ATO SaDg**: principal ferramenta do ambiente, composta de um conjunto de usuários (facilitadores), conjunto de problemas de decisão, conjunto de normas e conjunto de decisões. Esta ferramenta engloba todas as outras ferramentas do ambiente;
- **ATO Usuários**: ferramenta que permite a criação e a manipulação de usuários facilitadores dentro do SADG. Esta ferramenta está associada ao ATO SaDg;
- **ATO PD**: ferramenta que permite definição e manipulação de problemas de decisão, suas caracteríticas e também a decomposição deste em outros problemas. Esta ferramenta está associada ao ATO SaDg. Corresponde ao Editor de Problemas de Decisão;
- **ATO Normas**: ferramenta que permite configuração e manipulação de normas, suas características, regras para as atividades do processo decisório e a marcação do objeto de decisão (e.g. processo de software). Esta ferramenta está associada ao ATO SaDg. Corresponde ao Editor de Normas;
- **ATO Decisões**: ferramenta que define e gerência as ferramentas que auxiliam no processo de tomada de decisão dentro do SADG. Esta ferramenta está associada ao ATO SaDg.;
- **ATO Arg**: ferramenta que permite a criação e manipulação de elementos de argumentação contribuídos dentro da rede de discussão. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Editor de Argumentação;
- **ATO Extração**: ferramenta que permite criação e manipulação das alternativas extraídas a partir dos elementos de argumentação. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Extrator de Alternativas;
- **ATO Votação**: ferramenta que permite criação e manipulação de votos para as alternativas selecionadas durante a atividade de extração. Esta ferramenta está associada ao ATO Decisões. Corresponde ao Editor de Votação.



Interface

c_cria_sadg:
m_inclui_facilitador_sadg(_,_): SADG, STRING → SADG
m_remove_facilitador_sadg(_,_): SADG, STRING → SADG
m_adiciona_pd_sadg(_,_,_,_,_): SADG, STRING, DATE, DATE, TEXTO → SADG
m_remove_pd_sadg(_,_): SADG, STRING → SADG
m_adiciona_comp_pd(_,_,_): SADG, STRING, STRING → SADG
m_remove_comp_pd(_,_,_): SADG, STRING, STRING → SADG
m_configura_norma_sadg(_,_,_,_,_,_): SADG, STRING, STRING, TEXTO, STRING, STRING → SADG
m_remove_norma_sadg(_, _): SADG, STRING → SADG
m_adiciona_agente_sadg(_, _,_): SADG, STRING, STRING → SADG
m_remove_agente_sadg(_, _,_): SADG, STRING, STRING → SADG
m_modifica_projeto_gp(_,_,_): SADG, STRING, STRING → SADG
m_gera_atividade_decisao(_,_,_,_): SADG, GP, STRING, STRING → SADG
m_gera_atividade_argumentacao(_,_,_,_): SADG, GP, STRING, STRING → SADG
m_gera_atividade_extracao(_,_,_,_): SADG, GP, STRING, STRING → SADG
m_gera_atividade_votacao(_,_,_,_): SADG, GP, STRING, STRING → SADG
m_gera_atividade_compvoto(_,_,_,_): SADG, GP, STRING, STRING → SADG
m_cria_ambiente_decisao(_): SADG→ SADG
m_inclui_decisao_sadg(_,_,_): SADG, STRING, STRING → SADG
m_remove_decisao_sadg(_,_,_): SADG, STRING → SADG
m_inclui_palavra_decisao_sadg(_,_,_): SADG, STRING, STRING → SADG
m_remove_palavra_decisao_sadg(_,_,_): SADG, STRING, STRING → SADG
o_facilitador_cadastrado_sadg(_,_): SADG, STRING → BOOLEAN
o_existe_problema_sadg(_,_): SADG, STRING → BOOLEAN
o_componentes_problema(_,_): SADG, STRING → CONJUNTO
o_objetivo_problema(_,_): SADG, STRING → TEXTO
o_facilitador_resp_norma(_,_):SADG, STRING, STRING → BOOLEAN
o_agentes_resp_norma(_,_):SADG, STRING → BOOLEAN

o_agentes_ norma(_,_):SADG, STRING → CONJUNTO
o_problema_ norma(_,_):SADG, STRING → STRING
o_eproblema_ norma(_,_):SADG, STRING → BOOLEAN
o_projeto_gp_norma(_,_):SADG, STRING → STRING
o_regras_arg_norma(_,_):SADG, STRING → BOOLEAN
o_regras_ext_norma(_,_):SADG, STRING → BOOLEAN
o_regras_vot_norma(_,_):SADG, STRING → BOOLEAN
o_normas_resp_facilitador(_,_):SADG, STRING → CONJUNTO

Operações

Variáveis formais

v-sadg, v-novo-sadg	: SADG
v-gp	: GP
v-usuarios	: USUARIOS
v-norma	: NORMAS
v-id-norma, v-id-usuario, v-id-pd, v-componente	: STRING
v-facilitador, v-novo-facilitador, v-novo-idpd, v-palavra	: STRING
v-descricao, v-nova-descricao, v-objetivo	: TEXTO
v-nome-agente, v-nome-proj-gp, v-novo-nm-proj-gp	: STRING
v-id-decisao	: STRING
v-nome-agentes	: CONJUNTO
v-regra_arg, v-nova-regra-arg	: REGRA_ARG
v-regra_ext, v-nova-regra-ext	: REGRA_EXT
v-regra_vot, v-nova-regra-vot	: REGRA_VOT
v-pd	: PD
v-novo-nmaxQ, v-novo-nmaxQ, v-novo-nmaxQ	: INTEGER
v-dt-inicio, v-dt-fim	: DATE

c_cria_norma()
=(Usuarios **ICS**(Usuarios, c_cria_usuario), PD **ICS**(PD, c_cria_pd), Normas **ICS**(Normas, c_cria_norma), Decisoes **ICS**(Decisoes, c_cria_decisao))

m_inclui_facilitador_sadg((Usuarios v-usuarios,_,_,_),v-facilitador)
= (Usuarios **ICS**(Usuarios, m_inclui_usuario, v-usuarios <v-facilitador>),_,_,_)

m_remove_facilitador_sadg((Usuarios v-usuarios,_,_,_),v-facilitador)
=(Usuarios **ICS**(Usuarios, m_exclui_usuario, v-usuarios <v-facilitador>),_,_,_)

m_adiciona_pd_sadg((_, PD v-pd, _, _), v-id-pd, v-dt-inicio, v-dt-fim, v-objetivo)
=(_, PD **ICS**(PD, m_inclui_pd, v-pd, < v-id-pd, v-dt-inicio, v-dt-fim, v-objetivo>), _, _)

m_remove_pd_sadg((_, PD v-pd, _, _), v-id-pd)
=(_, PD **ICS**(PD, m_exclui_pd, v-pd, < v-id-pd >), _, _)

m_adicionar_comp_pd_sadg((_, PD v-pd, _, _), v-id-pd, v-componente)
=(_, PD **ICS**(PD, m_inclui_comp, v-pd, < v-id-pd, v-componente >), _, _)

m_remove_comp_pd_sadg((_, PD v-pd, _, _), v-id-pd, v-componente)
=(_, PD **ICS**(PD, m_exclui_comp, v-pd, < v-id-pd, v-componente >), _, _)

m_configura_norma_sadg((_, _, Normas v-norma, _), v-id-norma, v-facilitador, v-descricao, v-nome-proj-gp, v-id-pd)
=(_, _, Normas **ICS** (Normas, m_inclui_norma, v-norma, < v-id-norma, v-facilitador, v-descricao, v-nome-proj-gp, v-id-pd >),_)

m_remove_norma_sadg((_, _, Normas v-norma, _), v-id-norma)
=(_, _, Normas **ICS** (Normas, m_exclui_norma, v-norma, < v-id-norma>),_)

m_adiciona_agente_sadg((_, _, Normas v-norma, _), v-id-norma, v-nome-agente)
=(_, _, Normas **ICS** (Normas, m_inclui_agente, v-norma, < v-id-norma, v-nome-agente >),_)

m_remove_agente_sadg((_, _, Normas v-norma, _), v-id-norma, v-nome-agente)
=(_, _, Normas **ICS** (Normas, m_remove_agente, v-norma, < v-id-norma, v-nome-agente >),_)

m_modifica_projeto_gp((_, _, Normas v-norma, _), v-id-norma, v-nome-proj-gp)
=(_, _, Normas **ICS** (Normas, m_altera_nome_projeto_gp, v-norma, < v-id-norma, v-nome-proj-gp >),_)

m_gera_atividade_decisao((_, _, Normas v-norma, _), v-gp, v-id-norma, v-id-usuario)
=(_, _, Normas **ICS** (Normas, m_gera_inclui_atv_decisao, v-norma, < v-gp, v-id-norma, v-id-usuario >),_)

m_gera_atividade_argumentacao((_, _, Normas v-norma, _), v-gp, v-id-norma, v-id-usuario)
=(_, _, Normas **ICS** (Normas, m_gera_inclui_atv_argumentacao, v-norma, < v-gp, v-id-norma, v-id-usuario >),_)

m_gera_atividade_extracao((_, _, Normas v-norma, _), v-gp, v-id-norma, v-id-usuario)
=(_, _, Normas **ICS** (Normas, m_gera_inclui_atv_extracao, v-norma, < v-gp, v-id-norma, v-id-usuario >),_)

m_gera_atividade_votacao((_, _, Normas v-norma, _), v-gp, v-id-norma, v-id-usuario)
=(_, _, Normas **ICS** (Normas, m_gera_inclui_atv_votacao, v-norma, < v-gp, v-id-norma, v-id-usuario >),_)

m_gera_atividade_compvoto((_, _, Normas v-norma, _), v-gp, v-id-norma, v-id-usuario)
=(_, _, Normas **ICS** (Normas, m_gera_inclui_atv_compvoto, v-norma, < v-gp, v-id-norma, v-id-usuario >),_)

m_cria_ambiente_decisao (_, _, _, Decisoes v-decisoes)
=(_, _, _, Decisoes **ICS** (Decisoes, m_cria_decisao, v-decisoes))

m_inclui_decisao_sadg ((_, _, _, Decisoes v-decisoes), v-id-decisao, v-id-norma)
=(_, _, _, Decisoes **ICS** (Decisoes, m_inclui_decisao, v-decisoes, < v-id-decisao, v-id-norma >))

m_remove_decisao_sadg ((_, _, _, Decisoes v-decisoes), v-id-decisao)
=(_, _, _, Decisoes **ICS** (Decisoes, m_exclui_decisao, v-decisoes, < v-id-decisao >))

m_inclui_palavra_decisao_sadg ((_, _, _, Decisoes v-decisoes), v-id-decisao, v-palavra)
=(_, _, _, Decisoes **ICS** (Decisoes, m_inlcui_palavra, v-decisoes, < v-id-decisao, v-palavra >))

m_remove_palavra_decisao_sadg ((_, _, _, Decisoes v-decisoes), v-id-decisao, v-palavra)
=(_, _, _, Decisoes **ICS** (Decisoes, m_exclui_palavra, v-decisoes, < v-id-decisao, v-palavra >))

o_facilitador_cadastrado_sadg((Usuarios v-usuarios, _, _, _),v-id-usuario)
=Usuarios **ICS** (Usuarios, o_existe_id_usuario, v-usuarios, < v-id-usuario>), _, _, _)

o_existe_problema_sadg((_, PD v-pd, _, _), v-id-pd)
=(_, PD **ICS**(PD, o_existe_pd, v-pd, < v-id-pd>), _, _)

o_componentes_problema((_, PD v-pd, _, _), v-id-pd)
=(_, PD **ICS**(PD, o_comp_pd, v-pd, < v-id-pd>), _, _)

o_objetivo_problema((_, PD v-pd, _, _), v-id-pd)
=(_, PD **ICS**(PD, o_objetivo, v-pd, < v-id-pd>), _, _)

o_facilitador_resp_norma((_,_,Normas v-norma, _), v-id-norma, v-id-usuario)
=(_,_, Normas **ICS**(Normas, o_existe_facilitador_norma, v-norma, <v-id-norma, v-id-usuario>), _)

o_agentes_resp_norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_existe_agente_norma, v-norma, < v-id-norma>), _)

o_agentes_ norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_ agentes_norma, v-norma, < v-id-norma>), _)

o_problema_ norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_ id_problema_norma, v-norma, < v-id-norma>), _)

o_eproblema_ norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_ existe_problema_norma, v-norma, < v-id-norma>), _)

o_projeto_gp_norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_id_projeto_gp_norma, v-norma, < v-id-norma>), _)

o_regras_arg_norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_existe_regra_argumentacao, v-norma, < v-id-norma>), _)

o_regras_ext_norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_existe_regra_extracao, v-norma, < v-id-norma>), _)

o_regras_vot_norma((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_existe_regra_votacao, v-norma, < v-id-norma>), _)

o_normas_resp_facilitador((_,_,Normas v-norma, _), v-id-norma)
=(_,_, Normas **ICS**(Normas, o_facilitador_normas, v-norma, < v-id-norma>), _)

1.3 ATO Usuarios

Interface

c_cria_usuario:
m_inclui_usuario(_,_): USUARIOS,STRING → USUARIOS
m_exclui_usuario(_,_): USUARIOS,STRING → USUARIOS
m_existe_id_usuario(_,_): USUARIOS,STRING → BOOLEAN

Operações

Variáveis Formais

v-novo-usuario	: STRING
v-usuarios	: USUARIOS
v-gp	: GP

c_cria_usuario() = empty-set

m_inclui_usuario(v-usuarios,v-novo-usuario)=
if **ICS**(GP, existe_agente, v-novo-usuario) **and** **not**(existe_id_usuario(v-novo-usuario))
then add(v-usuarios, v-novo-usuario)
else v-usuarios

m_exclui_usuario(v-usuarios,v-id)=
if existe_id_usuario(v-id)
then delete(v-usuarios,v-id)
else v-usuarios

o_existe_id_usuario(v-usuarios,v-id)=
if is_in(v-usuarios,v-id)
then TRUE
else FALSE

1.4 ATO PD

Interface

c_cria_pd:
m_inclui_pd(_,_,_,_): PD, STRING, DATE, DATE, TEXTO → PD
m_exclui_pd(_,_): PD, STRING → PD
m_inclui_comp(_,_,_): PD, STRING, STRING → PD
m_exclui_comp(_,_,_): PD, STRING, STRING → PD

m_altera_dt_inicio(_,_,_): PD, STRING, DATE → PD
m_altera_dt_fim(_,_,_): PD, STRING, DATE → PD
m_altera_objetivo(_,_,_): PD, STRING, TEXTO → PD
o_data_inicio(_,_): PD, STRING → DATE
o_data_fim(_,_): PD, STRING → DATE
o_objetivo(_,_): PD, STRING → TEXTO
o_existe_pd(_,_): PD, STRING → BOOLEAN
o_comp_pd(_,_): PD, STRING → CONJ_STRING
o_existe_comp_pd(_):PD, STRING → BOOLEAN

Operações

Variáveis

v-dt_incio, v-nova-dt-incio, v-dt-fim, v-nova-dt-fim : STRING
v-idpd, v-busca-idpd : STRING
v-objetivo, v-novo-objetivo : TEXTO
v-pd : PD
v-componente : STRING
v-componentes : CONJUNTO

c_cria_pd() = empty-mapping

```
m_inclui_pd(v-id-pd, v-dt-incio, v-dt-fim, v-objetivo)=
modify   (v-idpd,
                (Data Inicio v-dt-inicio, Data Fim v-dt-fim, Objetivo v-objetivo,
                Componentes  empty-list
                ),
        v-pd
        )

m_exclui_pd(v-pd,v-idpd)=
Restrict_with(v-pd,add(empty-set,v-idpd)

m_inclui_comp(
modify(v-idpd,
                (_,_,_,Componentes v-componentes),
        v-pd
        ),
        v-busca-idpd,
        v-componente) =
if      v-busca-idpd=v-idpd
then    modify(v-idpd,
                        (_,_,_,_Componentes add(v-componentes, v-componente)),
                        v-pd
                        )
Else    modify(v-idpd,
                        (_,_,_,Componentes v-componentes),
                        m_inclui_comp(v-pd,v-busca-idpd,v-componente
                        )

m_inclui_comp(empty-mapping,_,_) = empty-mapping

m_exclui_comp(
modify(v-idpd,
                (_,_,_,Componentes v-componentes),
        v-pd
        ),
        v-busca-idpd,
        v-componente) =
if      v-busca-idpd=v-idpd
then    modify(v-idpd,
                        (_,_,_,_Componentes delete(v-componentes, v-componente)),
                        v-pd
                        )
Else    modify(v-idpd,
                        (_,_,_,Componentes v-componentes),
                        m_exclui_comp(v-pd,v-busca-idpd,v-componente
                        )

m_exclui_comp(empty-mapping,_,_) = empty-mapping

m_altera_dt_inicio(
```

```
modify(v-idpd,
                (Data Inicio v-dt-inicio,_,_,_),
        v-pd
        ),
        v-busca-idpd,
        v-nova-dt-inicio) =
if      v-busca-idpd=v-idpd
then    modify(v-idpd,
                        (Data Inicio v-nova-dt-inicio,_,_,_),
                        v-pd
                        )
Else    modify(v-idpd,
                        (Data Inicio v-dt-inicio,_,_,_),
                        m_altera_dt_inicio(v-pd,v-busca-idpd,v-nova-dt-incio)
                        )

m_altera_dt_inicio(empty-mapping,_,_) = empty-mapping

m_altera_dt_fim(
modify(v-idpd,
                (_,Data Fim v-dt-fim,_,_),
        v-pd
        ),
        v-busca-idpd,
        v-nova-dt-fim) =
if      v-busca-idpd=v-idpd
then    modify(v-idpd,
                                (_,Data Fim v-nova-dt-fim,_,_),
                        v-pd
                        )
Else    modify(v-idpd,
                                (_,Data Fim v-dt-fim,_,_),
                        m_altera_dt_fim(v-pd,v-busca-idpd,v-nova-dt-fim)
                        )

m_altera_dt_fim(empty-mapping,_,_) = empty-mapping

m_altera_objetivo(
modify(v-idpd,
                (_,Objetivo v-objetivo,_,_),
        v-pd
        ),
        v-busca-idpd,
        v-novo-objetivo) =
if      v-busca-idpd=v-idpd
then    modify(v-idpd,
                                (_,Objetivo v-novo-objetivo,_,_),
                        v-pd
                        )
Else    modify(v-idpd,
```

(_,Objetivo v-objetivo,_,_),
m_altera_objetivo(v-pd,v-busca-idpd,v-novo-objetivo)
)

m_altera_objetivo(empty-mapping,_,_) = empty-mapping

o_data_inicio(v-pd, v-busca-idpd) = select-Data Inicio(image_of(v-pd,v-busca-idpd))

o_data_fim(v-pd, v-busca-idpd) = select-Data Fim(image_of(v-pd,v-busca-idpd))

o_objetivo(v-pd, v-busca-idpd) = select-Objetivo(image_of(v-pd,v-busca-idpd))

o_existe_pd(v-pd, v-busca-idpd) = v-busca-idpd ∈ dom(v-pd)

o_comp_pd(v-pd, v-busca-idpd) =
add(select-Componentes(image_of(v-pd,v-busca-idpd)), empty-list)

o_existe_comp_pd(v-pd, v-busca-idpd) =
lng(*o_comp_pd*(v-pd,v-busca-idpd)) **not =** empty-list

1.5 ATO Normas

Interface

c_cria_norma:
m_inclui_norma(_,_,_,_,_): NORMA, STRING, STRING, TEXTO, STRING, STRING → NORMA
m_exclui_norma(_,_): NORMA, STRING → NORMA
m_inclui_agente(_,_,_): NORMA, STRING, STRING → NORMA
m_exclui_agente(_,_,_): NORMA, STRING, STRING → NORMA
m_altera_facilitador(_,_,_): NORMA, STRING, STRING → NORMA
m_altera_regra_arg_dt_prev(_,_,_,_): NORMA, STRING, DATE, DATE → NORMA
m_altera_regra_arg_nmax(_,_,_,_,_,_): NORMA, STRING, INTEGER, INTEGER, INTEGER → NORMA
m_altera_regra_ext_dt_prev(_,_,_,_): NORMA, STRING, DATE, DATE → NORMA
m_altera_regra_ext_facilitador(_,_,_): NORMA, STRING, STRING → NORMA
m_altera_regra_vot_dt_prev (_,_,_,_,_): NORMA, STRING, STRING, DATE, DATE → NORMA
m_altera_regra_vot_metodo (_,_,_,_): NORMA, STRING, STRING, STRING → NORMA m_altera_descricao(_,_,_): NORMA, STRING, STRING → NORMA
m_altera_nome_projeto_gp(_,_,_): NORMA, STRING, STRING → NORMA
m_altera_pd(_,_,_): NORMA, STRING, PD → NORMA
m_gera_inclui_atv_decisao(_,_,_,_): NORMAS, GP, STRING,STRING → GP
m_gera_inclui_atv_argumentacao(_,_,_,_): NORMAS, GP, STRING,STRING → GP
m_gera_inclui_atv_extracao(_,_,_,_): NORMAS, GP, STRING,STRING → GP
m_gera_inclui_atv_votacao(_,_,_,_): NORMAS, GP, STRING,STRING → GP
m_gera_inclui_atv_compvoto(_,_,_,_): NORMAS, GP, STRING,STRING → GP
o_existe_facilitador_norma(_,_): NORMA, STRING → BOOLEAN
o_existe_agente_norma(_,_): NORMA, STRING → BOOLEAN

o_agentes_norma(_,_): NORMA, STRING → CONJUNTO
o_id_problema_norma(_,_): NORMA, STRING → STRING
o_existe_problema_norma(_,_): NORMA, STRING → BOOLEAN
o_id_projeto_gp_norma(_,_): NORMA, STRING → STRING
o_existe_regra_argumentacao(_,_): NORMA, STRING → BOOLEAN
o_existe_regra_extracao(_,_): NORMA, STRING → BOOLEAN
o_existe_regra_votacao(_,_): NORMA, STRING → BOOLEAN
o_facilitador_normas(_,_): NORMA, STRING → CONJUNTO

Operações

Variáveis formais

v-norma	: NORMAS
v-id-norma, v-busca-id-norma, v-id-pd, v-novo-metodo	: STRING
v-facilitador, v-novo-facilitador, v-novo-idpd, v-nturno	: STRING
v-descricao, v-nova-descricao,	: TEXTO
v-nome-agente, v-nome-proj-gp, v-novo-nm-proj-gp	: STRING
v-nome-agentes	: CONJUNTO
v-regra_arg, v-nova-regra-arg	: REGRA_ARG
v-regra_ext, v-nova-regra-ext	: REGRA_EXT
v-regra_vot, v-nova-regra-vot	: REGRA_VOT
v-pd, v-novo-pd	: PD
v-novo-nmaxQ, v-novo-nmaxQ, v-novo-nmaxQ	: INTEGER
v-nova-dt-inicio, v-nova-dt-fim	: DATE

c_cria_norma() = empty-mapping

m_inclui_norma(v-norma, v-id-norma, v-facilitador, v-descricao, v-nome-proj-gp, v-id-pd)=
modify (v-id-norma,
(Facilitador v-facilitador, Agentes empty-set, Regra-Argumentacao **ICS** (REGRA_ARG, c_cria_regra_arg), Regra-Extracao **ICS** (REGRA_EXT, c_cria_regra_ext), Regra-Votacao **ICS** (REGRA_VOT, c_cria_regra_vot), Descricao v-descricao, Nome-Projeto-gp v-nome-proj-gp,
IDPD v-idpd
),
v-norma
)

m_exclui_norma(v-norma,v-id-norma)=
Restrict_with(v-norma,add(empty-set,v-id-norma)

m_inclui_agente (
modify(v-id-norma,
(_,Agentes v-nome-agentes, _,_,_,_,_,_),
v-norma
)
v-busca-id-norma,
v-nome-agente)=

```
if      v-busca-id-norma=v-id-norma
then    modify(v-id-norma,
                        (_,Agentes add(v-nome-agentes, v-nome-agente)
                         _,_,_,_,_,_),
                        v-norma
                        )
Else    modify(v-id-norma,
                        (_,Agentes v-nome-agentes_,_,_,_,_,_),
                        m_inclui_agente(v-norma,v-busca-id-norma,v-nome-agente
                        )

m_inclui_agente(empty-mapping,_,_) = empty-mapping


m_exclui_agente      (
modify(v-id-norma,
              (_,Agentes v-nome-agentes, _,_,_,_,_,_),
        v-norma
        )
        v-busca-id-norma,
        v-nome-agente)=
if      v-busca-id-norma=v-id-norma
then    modify(v-id-norma,
                        (_,Agentes delete(v-nome-agentes, v-nome-agente)
                         _,_,_,_,_,_),
                        v-norma
                        )
Else    modify(v-id-norma,
                        (_,Agentes v-nome-agentes_,_,_,_,_,_),
                        m_exclui_agente(v-norma,v-busca-id-norma,v-nome-agente
                        )

m_exclui_agente(empty-mapping,_,_) = empty-mapping

m_altera_facilitador(
modify(v-id-norma,
        (Facilitador v-facilitador,_,_,_,_,_,_,_),
        v-norma
        )
        v-busca-id-norma,
        v-novo-facilitador)=
if      v-busca-id-norma=v-id-norma
then    modify(v-id-norma,
                        (Facilitador v-novo-facilitador,_,_,_,_,_,_,_),
                        v-norma
                        )
Else    modify(v-id-norma,
                        (Facilitador v-facilitador,_,_,_,_,_,_,_),
                        m_altera_facilitador(v-norma,v-busca-id,v-novo-facilitador)
                        )
```

m_altera_facilitador(empty-mapping,_,_) = empty-mapping

m_altera_regra_arg_dt_prev(
modify(v-id-norma,
(_,_,Regra Arg v-regra_arg,_,_,_ ,_,_),
v-norma
)
v-busca-id-norma,
v-nova-dt-inicio,
v-nova-dt-fim)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,Regra Arg (
Data Inicio **ICS**(REGRAS_ARG, m_altera_data_inicio, v-regra_arg, v-nova-dt-inicio),
Data Fim **ICS**(REGRAS_ARG, m_altera_data_fim, v-regra_arg, v-nova-dt-fim)),_,_,_ ,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,Regra Arg v-regra_arg,_,_,_ ,_,_),
m_altera_regra_arg_dt_prev (v-norma,v-busca-id,v-nova-dt-inicio, v-nova-dt-fim)
)

m_altera_regra_arg_dt_prev (empty-mapping,_,_,_) = empty-mapping

m_altera_regra_arg_nmax(
modify(v-id-norma,
(_,_,Regra Arg v-regra_arg,_,_,_ ,_,_),
v-norma
)
v-busca-id-norma,
v-novo-nmaxQ,
v-novo-nmaxA,
v-novo-nmaxP)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,Regra Arg (
Data Inicio _,
Data Fim _,
Nmax Questao **ICS**(REGRAS_ARG, m_altera_nmax_questao, v-regra_arg, v-novo-nmaxQ) ,
Nmax Argumento **ICS**(REGRAS_ARG, m_altera_nmax_argumento, v-regra_arg, v-novo-nmaxA),
Nmax Posicao **ICS**(REGRAS_ARG, m_altera_nmax_posicao, v-regra_arg, v-novo-nmaxP)),_,_,_ ,_,_),
v-norma
)

Else modify(v-id-norma,
(_,_,Regra Arg v-regra_arg,_,_,_ ,_,_),
m_altera_regra_arg_nmax (v-norma,v-busca-id, v-novo-nmaxQ, v-novo-nmaxA, v-novo-nmaxP)
)

m_altera_regra_arg_nmax (empty-mapping,_,_,_,_) = empty-mapping

m_altera_regra_ext_dt_prev(
modify(v-id-norma,
(_,_,_,Regra Ext v-regra_ext,_,_ ,_,_),
v-norma
)
v-busca-id-norma,
v-nova-dt-inicio,
v-nova-dt-fim)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,Regra Ext (
Data_Inicio **ICS**(REGRAS_EXT, m_altera_dt_inicio, v-regra_ext, v-nova-dt-inicio),
Data_Fim **ICS**(REGRAS_EXT, m_altera_dt_fim, v-regra_ext, v-nova-dt-fim)),_,_ ,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,Regra Ext v-regra_ext,_,_ ,_,_),
m_altera_regra_ext_dt_prev (v-norma,v-busca-id,v-nova-dt-inicio, v-nova-dt-fim)
)

m_altera_regra_ext_dt_prev (empty-mapping,_,_,_) = empty-mapping

m_altera_regra_ext_facilitador(
modify(v-id-norma,
(_,_,_,Regra Ext v-regra_ext,_,_ ,_,_),
v-norma
)
v-busca-id-norma,
v-facilitador)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,Regra Ext (
Facilitador **ICS**(REGRAS_EXT, m_altera_facilitador, v-regra_ext, v-facilitador)),_,_ ,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,Regra Ext v-regra_ext,_,_ ,_,_),

m_altera_regra_ext_facilitador (v-norma,v-busca-id,v-facilitador)
)

m_altera_regra_ext_facilitador (empty-mapping,_,_) = empty-mapping

m_altera_regra_vot_dt_prev(
modify(v-id-norma,
(_,_,_,_,Regra Vot v-regra_vot,_, ,_,_),
v-norma
)
v-busca-id-norma,
v-nturno,
v-nova-dt-inicio,
v-nova-dt-fim)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,_,Regra Vot (modify(nturno,
Data Inicio **ICS**(REGRAS_VOT, m_altera_dt_inicio, v-regras_vot, nturno, v-nova-dt-inicio, v-nova-dt-fim),
Data Fim **ICS**(REGRAS_VOT, m_altera_dt_fim, v-regras_vot, nturno, v-nova-dt-inicio, v-nova-dt-fim)), v-regra_vot),_, ,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,_,Regra Vot v-regra-vot,_, ,_,_),
m_altera_regra_vot_dt_prev (v-norma, v-busca-id, v-nturno, v-nova-dt-inicio, v-nova-dt-fim)
)

m_altera_regra_vot_dt_prev (empty-mapping,_,_,_,_) = empty-mapping

m_altera_regra_vot_metodo(
modify(v-id-norma,
(_,_,_,_,Regra Vot v-regra_vot,_, ,_,_),
v-norma
)
v-busca-id-norma,
v-nturno,
v-novo-metodo)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,_,Regra Vot (modify(nturno,
Metodo **ICS**(REGRAS_VOT, m_altera_met_vot, v-regra_vot, nturno, v-novo-metodo),
Data Inicio _,
Data Fim _), v-regra_vot),_, ,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,_,Regra Vot v-regra-vot,_, ,_,_),

m_altera_regra_vot_metodo (v-norma, v-busca-id, v-nturno, v-novo-metodo)
)

m_altera_regra_vot_metodo (empty-mapping,_,_,_) = empty-mapping

m_altera_descricao(
modify(v-id-norma,
(_,_,_,_,_,Descricao v-descricao,_,_),
v-norma
)
v-busca-id-norma,
v-nova-descricao)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,_,_,Descricao v-nova-descricao,_,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,_,_,Descricao v-descricao,_,_),
m_altera_descricao(v-norma,v-busca-id,v-nova-descricao)
)

m_altera_descricao(empty-mapping,_,_) = empty-mapping

m_altera_nome_projeto_gp(
modify(v-id-norma,
(_,_,_,_,_,_ ,Nome Projeto GP v-nome-proj-gp,_),
v-norma
)
v-busca-id-norma,
v-nova-descricao)=
if v-busca-id-norma=v-id-norma
then modify(v-id-norma,
(_,_,_,_,_,_ ,Nome Projeto GP v-novo-nm-proj-gp,_),
v-norma
)
Else modify(v-id-norma,
(_,_,_,_,_,_ ,Nome Projeto GP v-nome-proj-gp,_),
m_altera_nome_projeto_gp(v-norma,v-busca-id,v-novo-nm-projeto)
)

m_altera_nome_projeto_gp(empty-mapping,_,_) = empty-mapping

m_altera_pd(
modify(v-id-norma,
(_,_,_,_,_,_ ,_,IDPD v-id-pd),
v-norma
)

```
        v-busca-id-norma,
        v-novo-idpd)=
if      v-busca-id-norma=v-id-norma
then    modify(v-id-norma,
                    (_,_,_,_,_,_ ,_,IDPD v-novo-idpd),
                    v-norma
                    )
Else    modify(v-id-norma,
                    (_,_,_,_,_,_ ,_,IDPD v-novo-idpd),
                    m_altera_pd(v-norma,v-busca-id,v-novo-idpd)
                    )

m_altera_pd(empty-mapping,_,_) = empty-mapping

m_gera_inclui_atv_decisao (
        modify(v-id-norma,
                    (Facilitador v-facilitador, Agentes v-agentes, Regra Argumentacao
                    (Data Inicio v-data-inicio, Data Fim v-data-fim), Regra Extracao v-
                    regra_ext, Regra Votacao v-regra_vot, v-descricao, Nome ProjGP
                    v-nome-proj-gp, idpd v-id-pd),
                v-norma),
                v-GP,
                v-busca-norma,
                v-id-usuario)=
        if      v-id-norma = v-busca-norma
        then    ICS(GP, inclui_atividade_projeto,
                            v-GP,
                            v-nome-proj-gp,
                     (Nome "1-preparar ambiente SaDg", Descrição v-descricao ,
                     Agentes v-facilitador, Prod_Envolvidos (ProdutosEnt v-id-norma,
                     ProdutosSaida empty-set),
                     Ativ_Ant empty-set, Condições (Precond ICS(CONDICAO, cria-
                     cond-vazia), Poscond ICS(CONDICAO, cria-cond-vazia)),
                     Esperando "esperando", Recursos empty-set,
                     Cronograma (Previsao(Inicio v-data-inicio, Fim v-data-fim),
                     Andamento (Inicio null-date, Fim null-date)),
                     Script DescInformal "Criar ambiente SaDg para tomada de
                     decisão dentro do processo de desenvolvimento de software" ),
                            v-id-usuario)
        else m_gera_inclui_atv_decisao (v-norma, v-GP, v-busca-norma, v-id-usuario)

m_gera_inclui_atv_decisao (empty-mapping,_,_,_) = empty-mapping

m_gera_inclui_atv_argumentacao (
        modify(v-id-norma,
                    (Facilitador v-facilitador, Agentes v-agentes, Regra Argumentacao
                    (Data Inicio v-data-inicio, Data Fim v-data-fim), Regra Extracao v-
                    regra_ext, Regra Votacao v-regra_vot, v-descricao, Nome ProjGP
                    v-nome-proj-gp, idpd v-id-pd),
                v-norma),
```

v-GP,
v-busca-norma,
v-id-usuario)=
if v-id-norma = v-busca-norma
then **ICS**(GP, inclui_atividade_projeto,
v-GP,
v-nome-proj-gp,
(Nome "2- Participar Discussão", Descrição v-descricao , Agentes v-agentes, Prod_Envolvidos (ProdutosEnt v-id-norma, ProdutosSaida elem_arg),
Ativ_Ant 1, Condições (Precond ICS(CONDICAO, cria-cond-vazia), Poscond ICS(CONDICAO, cria-cond-vazia)),
Esperando "esperando", Recursos empty-set,
Cronograma (Previsao(Inicio v-data-inicio, Fim v-data-fim),
Andamento (Inicio null-date, Fim null-date)),
Script DescInformal "Contribuir com elementos de discussão na atividade de argumentação utilizando do Editor de Argumentação"),
v-id-usuario)
else *m_gera_inclui_atv_argumentacao* (v-norma, v-GP, v-busca-norma, v-id-usuario)

m_gera_inclui_atv_argumentacao (empty-mapping,_,_,_) = empty-mapping

m_gera_inclui_atv_extracao (
modify(v-id-norma,
(Facilitador v-facilitador, Agentes v-agentes, Regra Argumentacao (Data Inicio v-data-inicio, Data Fim v-data-fim), Regra Extracao (Facilitador v-facilitador, Data Inicio v-dt-inicio, Data Fim v-dt-fim), Regra Votacao v-regra_vot, v-descricao, Nome ProjGP v-nome-proj-gp, idpd v-id-pd),
v-norma),
v-GP,
v-busca-norma,
v-id-usuario)=
if v-id-norma = v-busca-norma
then **ICS**(GP, inclui_atividade_projeto,
v-GP,
v-nome-proj-gp,
(Nome "3- Selecionar Alternativas", Descrição v-descricao , Agentes v-facilitador, Prod_Envolvidos (ProdutosEnt {v-id-norma, elem_arg} ProdutosSaida alternativas_votacao),
Ativ_Ant 2, Condições (Precond ICS(CONDICAO, cria-cond-vazia), Poscond ICS(CONDICAO, cria-cond-vazia)),
Esperando "esperando", Recursos empty-set,
Cronograma (Previsao(Inicio v-dt-inicio, Fim v-dt-fim),
Andamento (Inicio null-date, Fim null-date)),
Script DescInformal "Contribuir com elementos de discussão na atividade de argumentação utilizando do Editor de Argumentação"),

```
                    v-id-usuario)
        else m_gera_inclui_atv_extracao (v-norma, v-GP, v-busca-norma, v-id-usuario)

m_gera_inclui_atv_extracao (empty-mapping,_,_,_) = empty-mapping

m_gera_inclui_atv_votacao (
        modify(v-id-norma,
                (Facilitador v-facilitador, Agentes v-agentes, Regra Argumentacao
                v-regra_arg, Regra Extracao v-regra_ext, Regra Votacao (Data
                Inicio v-data-inicio, Data Fim v-data-fim), v-descricao, Nome
                ProjGP v-nome-proj-gp, idpd v-id-pd),
            v-norma),
            v-GP,
            v-busca-norma,
            v-id-usuario)=
        if  v-id-norma = v-busca-norma
        then ICS(GP, inclui_atividade_projeto,
                        v-GP,
                        v-nome-proj-gp,
                 (Nome "4- Participar Votação", Descrição v-descricao , Agentes
                 v-agentes, Prod_Envolvidos (ProdutosEnt v-id-norma,
                 ProdutosSaida v-votacao),
                 Ativ_Ant 1, Condições (Precond ICS(CONDICAO, cria-cond-
                 vazia), Poscond ICS(CONDICAO, cria-cond-vazia)),
                 Esperando "esperando", Recursos empty-set,
                 Cronograma (Previsao(Inicio v-data-inicio, Fim v-data-fim),
                 Andamento (Inicio null-date, Fim null-date)),
                 Script DescInformal "Contribuir com os votos na atividade de
                 argumentação utilizando do Editor de Votacão" ),
                        v-id-usuario)
        else m_gera_inclui_atv_votacao (v-norma, v-GP, v-busca-norma, v-id-usuario)

m_gera_inclui_atv_votacao (empty-mapping,_,_,_) = empty-mapping


m_gera_inclui_atv_compvoto (
        modify(v-id-norma,
                (Facilitador v-facilitador, Agentes v-agentes, Regra Argumentacao
                v-regra_arg, Regra Extracao v-regra_ext, Regra Votacao (Metodo
                Votacao v-metodo-votacao, Data Inicio v-data-inicio, Data Fim v-
                data-fim), v-descricao, Nome ProjGP v-nome-proj-gp, idpd v-id-
                pd),
            v-norma),
            v-GP,
            v-busca-norma,
            v-id-usuario)=
        if  v-id-norma = v-busca-norma
        then ICS(GP, inclui_atividade_projeto,
                        v-GP,
                        v-nome-proj-gp,
```

(Nome “4- Computar Votos”, Descrição v-descricao , Agentes v-facilitador, Prod_Envolvidos (ProdutosEnt v-id-norma, ProdutosSaida v-votacao),
Ativ_Ant 1, Condições (Precond ICS(CONDICAO, cria-cond-vazia), Poscond ICS(CONDICAO, cria-cond-vazia)),
Esperando “esperando”, Recursos empty-set,
Cronograma (Previsao(Inicio v-data-inicio, Fim v-data-fim),
Andamento (Inicio null-date, Fim null-date)),
Script DescInformal “Contribuir com os votos na atividade de argumentação utilizando do Editor de Votacão”),
v-id-usuario)

else *m_gera_inclui_atv_compvoto* (v-norma, v-GP, v-busca-norma, v-id-usuario)

m_gera_inclui_atv_compvoto (empty-mapping,_,_,_) = empty-mapping

o_existe_facilitador_norma(v-norma, v-busca-idnorma)
= **if not**(select-Facilitador(image_of(v-norma,v-busca-idnorma))) = {}
then TRUE
else FALSE

o_agentes_norma(v-norma, v-busca-idnorma) =
add(select-Componentes(image_of(v-norma,v-busca-idnorma)), empty-list)

o_existe_agente_norma(v-norma, v-busca-idpd)
= **if not**(lng(*o_comp_pd*(v-norma,v-busca-idnorma))) = empty-list
then TRUE
else FALSE

o_id_problema_norma(v-norma, v-busca-idnorma)
= (select-IDPD(image_of(v-norma,v-busca-idnorma)))

o_existe_problema_norma(v-norma, v-busca-idpd)
= **if not**(lng(*o_id_problema_decisao*(v-norma,v-busca-idnorma))) = empty-list
then TRUE
else FALSE

o_id_projeto_gp_norma(v-norma, v-busca-idnorma)
= (select-Nome ProjGP(image_of(v-norma,v-busca-idnorma)))

o_existe_regra_argumentacao(v-norma, v-busca-idnorma)
= **if not**(select-Regras_Arg(image_of(v-norma,v-busca-idnorma))) = {}
then TRUE
else FALSE

o_existe_regra_extracao(v-norma, v-busca-idnorma)
= **if not**(select-Regras_Ext(image_of(v-norma,v-busca-idnorma))) = {}

then TRUE
else FALSE

o_existe_regra_votacao(v-norma, v-busca-idnorma)
= **if not**(select-Regras_Vot(image_of(v-norma,v-busca-idnorma))) = {}
then TRUE
else FALSE

o_facilitador_normas(v-norma, v-id-facilitador) =
add(select-Norma(dom(v-norma,v-id-facilitador)), empty-list)

1.6 ATO Regra Argumentação (REGRAS_ARG)

Interface

c_cria_regra_arg(_,_,_,_,_): DATE, DATE, INTEGER, INTEGER, INTEGER → REGRAS_ARG
m_altera_data_inicio(_,_): REGRAS_ARG, DATE → REGRAS_ARG
m_altera_data_fim(_,_): REGRAS_ARG, DATE → REGRAS_ARG
m_altera_nmax_questao(_,_): REGRAS_ARG, INTEGER → REGRAS_ARG
m_altera_nmax_argumento(_,_): REGRAS_ARG, INTEGER → REGRAS_ARG
m_altera_nmax_posicao(_,_): REGRAS_ARG, INTEGER → REGRAS_ARG
o_dt_prev_ini_arg(_): REGRAS_ARG → DATE
o_dt_prev_fim_arg(_): REGRAS_ARG → DATE
o_nmax_questao(_): REGRAS_ARG → INTEGER
o_nmax_argumento(_): REGRAS_ARG → INTEGER
o_nmax_posicao(_): REGRAS_ARG → INTEGER

Operações

Variáveis formais

v-dt-inicio, v-nova-dt-inicio, v-dt-fim, v-nova-dt-fim	: DATE
v-nmax-quest,v-nova-nmax-quest	: INTEGER
v-nmax-arg,v-nova-nmax-arg	: INTEGER
v-nmax-pos,v-nova-nmax-pos	: INTEGER

c_cria_regra_arg(v-dt-inicio,v-dt-fim,v-namx-quest,v-nmax-arg,v-nmax-pos)=
(Data Inicio v-dt-inicio, Data Fim v-dt-fim, Nmax Quest v-nmax-quest, Nmax Arg v-nmax-arg, Nmax Pos v-nmax-pos)

m_altera_data_inicio((Data Inicio v-dt-inicio,_,_,_,_),v-nova-dt-incio)=
(Data Inicio v-nova-dt-inicio, _,_,_,_)

m_altera_data_fim((_,Data Fim v-dt-fim,_,_,_),v-nova-dt-fim)=
(_,Data Fim v-nova-dt-fim,_,_,_)

m_altera_nmax_questao((_,_,Nmax Questao v-nmax-quest,_,_),v-nova-nmax-quest)=
(_,_,Nmax Questao v-nova-nmax-quest,_,_)

m_altera_nmax_argumento((_,_,_,Nmax Argumento v-nmax-arg,_),v-nova-nmax-arg)=
(_,_,_,Nmax Argumento v-nova-nmax-arg,_)

m_altera_nmax_posicao((_,_,_,_,Nmax Posicao v-nmax-pos),v-nova-nmax-pos)=
(_,_,_,_,Nmax Posicao v-nova-nmax-pos)

o_dt_prev_ini_arg(v-dt-inicio) = select-Data Inicio(v-dt-inicio)

o_dt_prev_fim_arg(v-dt-fim) = select-Data Fim(v-dt-fim)

o_nmax_questao(v-nmax-quest) = select-Nmax Questao(v-nmax-quest)

o_nmax_argumento(v-nmax-arg) = select-Nmax Argumento(v-nmax-arg)

o_nmax_posicao(v-nmax-pos) = select-Nmax Posicao(v-nmax-pos)

1.7 ATO Regra Votacao (REGRAS_VOT)

Interface

c_cria_regra_vot:
m_inclui_regra_vot(_,_,_,_,_): REGRA_VOT, STRING, STRING, DATE, DATE → REGRA_VOT
m_exclui_regra_vot(_,_,_): REGRA_VOT, STRING → REGRA_VOT
m_altera_met_vot(_,_,_): REGRA_VOT, STRING, STRING → REGRA_VOT
m_altera_dt_inicio(_,_,_): REGRA_VOT, STRING, DATE → REGRA_VOT
m_altera_dt_fim(_,_,_): REGRA_VOT, STRING, DATE → REGRA_VOT
o_existe_regra_vot(_,_): REGRA_VOT, STRING → BOOLEAN
o_data_inicio(_,_): REGRA_VOT, STRING → DATE
o_data_fim(_,_): REGRA_VOT, STRING → DATE
o_metodo(_,_): REGRA_VOT, STRING → STRING

Operações

Variavéis Formais

v-regra-vot : NORMAS
v-nturno, v-busca-nturno : STRING
v-met-vot, v-novo-met-vot : STRING
v-regra_vot : REGRA_VOT
v-dt-inicio, v-nova-dt-inicio, v-dt-fim, v-nova-dt-fim : DATE

c_cria_regra_vot() = empty-mapping

m_inclui_regra_vot(v-id-turno,v-met-vot,v-dt-inicio,v-dt-fim)=
modify(v-nturno,
(Metodo Votacao v-met-vot, Data Inicio v-dt-inicio, Data Fim v-dt-fim),
v-regra-vot)

m_exclui_regra_vot(v-regra-arg,v-nturno)=
Restrict_with(v-regra-arg,add(empty-set,v-nturno)

m_altera_met_vot(
modify(v-nturno,
(Metodo Votacao v-met-vot,_,_),
v-regra-vot
)
v-busca-nturno

```
        v-novo-met-vot)=
if        v-busca-nturno=v-nturno and
          is_in({Pluralidade,Borda-Kendal,NAI}, v-novo-met-vot)
then      modify      (v-nturno,
                              (Metodo Votacao v-novo-met-vot,_,_),
                      v-regra-vot
                      )
Else      modify(v-nturno,
                              (Metodo Votacao v-met-vot,_,_),
                      m_altera_met_vot(v-regra-vot,v-busca-nturno,v-novo-met-vot)
                      )

m_altera_met_vot(empty-mapping,_,_) = empty-mapping

m_altera_dt_inicio(
modify(v-nturno,
              (_,Data Inicio v-dt-inicio,_),
        v-regra-vot
        )
        v-busca-nturno
        v-novo-met-vot)=
if        v-busca-nturno=v-nturno
then      modify      (v-nturno,
                              (_,Data Inicio v-nova-dt-inicio,_),
                      v-regra-vot
                      )
Else      modify(v-nturno,
                              (_,Data Inicio v-dt-inicio,_),
                      m_altera_dt_inicio(v-regra-vot,v-busca-nturno,v-nova-dt-inicio)
                      )

m_altera_dt_inicio(empty-mapping,_,_) = empty-mapping

m_altera_dt_fim(
modify(v-nturno,
              (_,_,Data Fim v-dt-fim),
        v-regra-vot
        )
        v-busca-nturno
        v-novo-met-vot)=
if        v-busca-nturno=v-nturno
then      modify      (v-nturno,
                              (_,_,Data Fim v-nova-dt-fim),
                      v-regra-vot
                      )
Else      modify(v-nturno,
                              (_,_,Data Fim v-dt-fim),
                      m_altera_dt_fim(v-regra-vot,v-busca-nturno,v-nova-dt-fim)
                      )
```

m_altera_dt_fim(empty-mapping,_,_) = empty-mapping

o_existe_regra_vot(v-regra-vot,v-nturno) = v-nturno ∈ dom(v-regra-vot)

o_data_inicio(v-regra-vot, v-busca-nturno) = select-Data Inicio(image_of(v-regra-vot, v-busca-nturno))

o_data_fim(v-regra-vot, v-busca-nturno) = select-Data Fim(image_of(v-regra-vot, v-busca-nturno))

o_metodo(v-regra-vot, v-busca-nturno) = select-Metodo Votacao(image_of(v-regra-vot, v-busca-nturno))

1.8 ATO Regras Extração (REGRAS_EXT)

Interface

c_cria_regra_ext(_,_,_): STRING, DATE, DATE → REGRA_EXT
m_altera_facilitador(_,_): REGRA_EXT, STRING → REGRA_EXT
m_altera_dt_inicio(_,_): REGRA_EXT, DATE → REGRA_EXT
m_altera_dt_fim(_,_): REGRA_EXT, DATE → REGRA_EXT
o_dt_inicio_ext(_): REGRA_EXT → DATE
o_dt_fim_ext(_): REGRA_EXT → DATE
o_facilitador_ext(_): REGRA_EXT → STRING

Operações

Variáveis Formais

v-facilitador,v-novo-facilitador : STRING
v-dt-inicio,v-dt-fim, v-nova-dt-inicio, v-nova-dt-fim : DATE

c_cria_regra_ext(v-facilitador,v-dt-inicio,v-dt-fim)=(Facilitador v-facilitador, Data Inicio v-dt-inicio, Data Fim v-data-fim)

m_altera_facilitador((Facilitador v-facilitador,_),v-novo-facilitador)=
(Facilitador v-novo-facilitador)

m_altera_dt_inicio((_,Data Inicio v-dt-inicio,_),v-nova-dt-incio)=
(_,Data Inicio v-nova-dt-inicio,_)

m_altera_dt_fim((_,_,Data Fim v-dt-fim),v-nova-dt-fim)=
(_,_,Data Fim v-nova-dt-fim)

o_dt_inicio_ext(v-dt-inicio) = select-Data Inicio(v-dt-inicio)

o_dt_fim_ext(v-dt-fim) = select-Data Fim(v-dt-fim)

o_facilitador_ext(v-facilitador) = select-Facilitador(v-facilitador)

1.9 ATO Decisoes

Interface

c_cria_decisao:
m_inclui_decisao(_,_): DECISAO, STRING, STRING → DECISAO
m_exclui_decisao(_,_): DECISAO, STRING → DECISAO
m_inclui_palavra(_,_,_): DECISAO, STRING, STRING → DECISAO
m_exclui_palavra(_,_,_): DECISAO, STRING, STRING → DECISAO
m_altera_idnorma(_,_,_): DECISAO, STRING, STRING → DECISAO
m_altera_arg(_,_,_): DECISAO, STRING, ARGUMENTACAO → DECISAO
m_altera_ext(_,_,_): DECISAO, STRING, EXTRACAO → DECISAO
m_altera_vot(_,_,_): DECISAO, STRING, VOTACAO → DECISAO
o_norma_decisao(_,_): DECISAO, STRING→ STRING
o_decisao_palavra_chave(_,_): DECISAO, STRING→ CONJUNTO
o_palavras_chave(_,_): DECISAO, STRING→ CONJUNTO
o_existe_decisao(_,_): DECISAO, STRING→ BOOLEAN
o_existe_norma_decisao(_,_): DECISAO, STRING→ BOOLEAN
o_existe_arg_decisao(_,_): DECISAO, STRING→ BOOLEAN
o_existe_ext_decisao(_,_): DECISAO, STRING→ BOOLEAN
o_existe_vot_decisao(_,_): DECISAO, STRING→ BOOLEAN

Operações

Variáveis formais

v-decisao	: DECISAO
v-iddecisao, v-busca-iddecisao, v-idnorma	: STRING
v-palavra, v-novo-idnorma	: STRING
v-pal-chaves	: CONJUNTO
v-argumentacao, v-nova-argumentacao	: ARGUMENTACAO
v-extracao, v-nova-extracao	: EXTRACAO
v-votacao, v-nova-votacao	: VOTACAO

c_cria_decisao() = empty-mapping

m_inclui_decisao(v-decisao, v-idnorma, v-iddecisao)=
modify (v-iddecisao,
(IDNorma v-idnorma, Argumentacao **ICS** (ARGUMENTACAO, c_cria_arg), Extracao **ICS** (EXTRACAO, c_cria_ext), Votacao **ICS** (VOTACAO, c_cria_vot), Palavras Chave empty-set)
,
v-decisao
)

m_exclui_decisao(v-decisao,v-iddecisao)=
Restrict_with(v-decisao,add(empty-set,v-iddecisao)

m_inclui_palavra(
modify(v-iddecisao,

```
                (_,_,_,_,Palavras Chaves v-pal-chaves)
        ,
        v-decisao
        )
        v-busca-iddecisao,
        v-palavra)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                                (_,_,_,_,Palavras Chaves add(v-pal-chaves, v-palavra)
                        ,
                        v-decisao
                        )
Else    modify(v-iddecisao,
                                (_,_,_,_,Palavras Chaves v-pal-chaves),
                        m_inclui_palavras(v-decisao, v-busca-iddecisao, v-palavra)
                        )

m_inclui_palavra(empty-mapping,_,_) = empty-mapping

m_exclui_palavra(
modify(v-iddecisao,
                (_,_,_,_,Palavras Chaves v-pal-chaves)
        ,
        v-decisao
        )
        v-busca-iddecisao,
        v-palavra)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                                (_,_,_,_,Palavras Chaves delete(v-pal-chaves, v-palavra)
                        ,
                        v-decisao
                        )
Else    modify(v-iddecisao,
                                (_,_,_,_,Palavras Chaves v-pal-chaves),
                        m_exclui_palavras(v-decisao, v-busca-iddecisao, v-palavra)
                        )

m_exclui_palavra(empty-mapping,_,_) = empty-mapping

m_altera_idnorma(
modify(v-iddecisao,
        (IDNorma v-idnorma,_,_,_,_),
        v-decisao
        )
        v-busca-iddecisao,
        v-novo-idnorma)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                                (IDNorma v-novo-idnorma,_,_,_,_),
```

```
                    v-decisao
                    )
Else    modify(v-iddecisao,
                        (IDNorma v-idnorma,_,_,_,_),
                    m_altera_idnorma(v-decisao,v-busca-iddecisao, v-novo-idnorma)
                    )

m_altera_idnorma(empty-mapping,_,_) = empty-mapping

m_altera_arg(modify (v-iddecisao,
        (_,Argumentacao v-argumentacao,_,_,_),
        v-decisao
        )
        v-busca-iddecisao,
        v-nova-argumentacao)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                        (_,Argumentacao v-nova-argumentacao,_,_,_),
                    v-decisao
                    )
Else    modify(v-iddecisao,
                        (_,Argumentacao v-argumentacao,_,_,_),
                    m_altera_arg(v-decisao, v-busca-iddecisao, v-nova-
argumentacao)
                    )

m_altera_arg(empty-mapping,_,_) = empty-mapping

m_altera_ext(modify (v-iddecisao,
        (_,_ ,Extracao v-extracao,_,_),
        v-decisao
        )
        v-busca-iddecisao,
        v-nova-extracao)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                        (_,_ ,Extracao v-nova-extracao,_,_),
                    v-decisao
                    )
Else    modify(v-iddecisao,
                        (_,_ ,Extracao v-nova-extracao,_,_),
                    m_altera_ext(v-decisao, v-busca-iddecisao, v-extracao)
                    )

m_altera_ext(empty-mapping,_,_) = empty-mapping

m_altera_vot(modify (v-iddecisao,
        (_,_ ,_ , Votacao v-votacao,_),
        v-decisao
        )
```

```
        v-busca-iddecisao,
        v-nova-votacao)=
if      v-busca-iddecisao=v-iddecisao
then    modify(v-iddecisao,
                        (_,_ ,_ , Votacao v-nova-votacao,_),
                v-decisao
                )
Else    modify(v-iddecisao,
                        (_,_ ,_ , Votacao v-votacao,_),
                m_altera_vot(v-decisao, v-busca-iddecisao, v-nova-votacao)
                )

m_altera_vot(empty-mapping,_,_) = empty-mapping

o_norma_decisao(v-decisao, v-busca-iddecisao)
= select-Norma(image_of(v-decisao,v-busca-iddecisao))

o_existe_norma_decisao(v-decisao, v-busca-iddecisao)
=       if not o_norma_decisao(v-decisao, v-busca-iddecisao) = {}
        then TRUE
        else FALSE

o_existe_arg_decisao(v-decisao, v-busca-iddecisao)
=       if not (select-Argumentacao(image_of(v-decisao,v-busca-iddecisao)))=empty-
        mapping
        then TRUE
        else FALSE

o_existe_ext_decisao(v-decisao, v-busca-iddecisao)
=       if not (select-Argumentacao(image_of(v-decisao,v-busca-iddecisao)))=empty-
        mapping
        then TRUE
        else FALSE

o_existe_vot_decisao(v-decisao, v-busca-iddecisao)
=       if not (select-Argumentacao(image_of(v-decisao,v-busca-iddecisao)))=empty-
        mapping
        then TRUE
        else FALSE

o_existe_decisao(v-decisao, v-busca-iddecisao)
= v-busca-iddecisao ∈ dom(v-decisao)

o_palavras_chave(v-decisao, v-busca-iddecisao)
= add(select-Palavras Chave(image_of(v-decisao,v-busca-iddecisao)), empty-list)

o_decisao_palavra_chave(v-decisao, v-busca-palavra)
=       if is_in(rng(v-decisao),v-busca-palavra)
        then add(dom(v-decisao),empty-set)
        else empty-set
```

1.10 ATO Argumentacao

Interface

c_cria_arg:
m_inclui_arg(_,_): ARG, STRING, DATE → ARG
m_exclui_arg(_,_): ARG, STRING → ARG
m_altera_elem_arg(_,_,_): ARG, STRING, ELEM_ARG → ARG
m_altera_dt_realizacao(_,_,_): ARG, STRING, DATE → ARG
m_inclui_elem_arg(_,_,_,_,_,_,_,_): ARG, ELEM_ARG, STRING, STRING, STRING, STRING, TEXTO, DATE → ARG
m_exclui_elem_arg(_,_,_): ARG, STRING, STRING → ARG
m_relaciona_elementos(_,_,_,_,_,_,_): ARG, STRING, ELEM_ARG, STRING, STRING,STRING, STRING → ARG
o_existe_arg(_,_): ARG, STRING → BOOLEAN

Operações

Variáveis formais

v-arg : ARG
v-idarg, v-busca-idarg, v-busca-idElem, v-idElem : STRING
v-idUsuario, v-tipoElem, v-rotulo-elem, v-idElem2 : STRING
v-relacao : STRING
v-textoElem : TEXTO
v-dt-realizacao, v-nova-dt-realizacao, v-dt-apresentacao : DATE
v-elem-arg, v-novo-elem-arg : ELEM_ARG

c_cria_arg() = empty-mapping

m_inclui_arg(v-arg, v-idarg, v-dt-realizacao)=
modify (v-idarg,
(Elemento Argumentacao **ICS** (ELEM_ARG, c_cria_elem_arg, v-elem-arg),
Data Realizacao v-dt-realizacao)
,
v-arg
)

m_exclui_arg(v-arg, v-idarg)=
Restrict_with(v-arg,add(empty-set,v-idarg)

m_altera_dt_realizacao(
modify(v-idarg,
(_, Data Realizacao v-dt-realizacao),
v-arg
),
v-busca-idarg,
v-nova-dt-realizacao)=

```
if      v-busca-idarg=v-idarg
then    modify(v-idarg,
                        (_, Data Realizacao v-nova-dt-realizacao),
                v-arg
                )
Else    modify(v-idarg,
                        (_, Data Realizacao v-dt-realizacao),
                m_altera_dt_realizacao(v-arg, v-busca-idarg, v-nova-dt-
realizacao)
                )

m_altera_dt_realizacao(empty-mapping,_,_) = empty-mapping

m_altera_elem_arg(
modify(v-idarg,
        (Elemento Argumentacao ((modify  (v-idElem,
                (Relacao ( Tipo Relacao v-relacao, Elemento v-idElem2)),
        v-elem-arg
        )),
    v-arg
    ),
    v-busca-idarg,
    v-busca-idElem,
    v-novo-elem-arg)=
if      v-busca-idarg=v-idarg
then    modify(v-idarg,
        (Elemento Argumentacao ICS(ELEM_ARG, m_ altera_elem_arg_aux , v-
        elem-arg, v-busca-idElem,  v-idElem, v-idElem2, v-relacao)),
          v-arg)

Else    modify(v-idarg,
                        (Elemento Argumentacao v-elem-arg, _),
                m_altera_elem_arg(v-arg, v-busca-idarg, v-novo-elem_arg)
                )

m_altera_elem_arg(empty-mapping,_,_) = empty-mapping

m_inclui_elem_arg(
    modify(v-idarg,
                Elemento Argumentacao v-elem-arg , _
          v-arg),
        v-elem-arg, v-idElem, v-idUsuario, v-tipoElem, v-rotulo-elem , v-
        textoElem, v-dt-apresentacao) =
    if    v-tipoElem = "Posicao" and  v-idarg = v-busca-idarg
        then    ICS(ELEM_ARG, m_inclui_elem_posicao,  v-elem-arg, v-
        idElem, v-idUsuario, v-tipoElem, v-rotulo-elem , v-textoElem, v-dt-
        apresentacao)
    else
          if    v-tipoElem = "Questao" and  v-idarg = v-busca-idArg
```

then **ICS**(ELEM_ARG, m_inclui_elem_questao, v-elem-arg, v-idElem, v-idUsuario, v-tipoElem, v-rotulo-elem , v-textoElem, v-dt-apresentacao)

else

if v-tipoElem = “Argumento” **and** v-idarg = v-busca-idArg

then **ICS**(ELEM_ARG, m_inclui_elem_argumento , v-elem-arg, v-idElem, v-idUsuario, v-tipoElem, v-rotulo-elem , v-textoElem, v-dt-apresentacao)

else *m_inclui_elem_arg*(v-arg, v-busca-idArg, v-elem-arg, v-idElem, v-idUsuario, v-tipoElem, v-rotulo-elem , v-textoElem, v-dt-apresentacao)

m_inclui_elem_arg(empty-mapping,_,_,_,_,_,_,_,_) = empty-mapping

m_exclui_elem_arg(
modify(v-idarg,
Elemento Argumentacao v-elem-arg,
v-arg),
v-busca-idarg,
v-idElem)=
if v-busca-idarg = v-idarg
then **ICS**(ELEM_ARG, m_exclui_elem_arg, v-elem-arg, v-idElem)
else *m_exclui_elem_arg*(v-arg, v-busca-idarg, v-idElem)

m_exclui_elem_arg(empty-mapping,_,_): empty-mapping

m_relaciona_elementos(
modify(v-idarg,
Elemento Argumentacao v-elem-arg,
v-arg),
v-busca-idarg,
v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao
)=
if v-busca-idarg = v-idarg
then **ICS**(ELEM_ARG, m_relacao_elem_arg, v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)
else *m_relaciona_elementos*(v-arg, v-busca-idarg, v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)

m_relaciona_elementos(empty-mapping,_,_,_,_,_,_) = empty-mapping

o_existe_arg(
modify(v-idarg,
Elemento Argumentacao v-elem-arg,
v-arg),
v-busca-idarg)=
if v-busca-idarg = v-idarg
then TRUE
else FALSE

1.11 ATO ELEM_ARG

Interface

c_cria_elem_arg:
m_inclui_elem_questao(_,_,_,_,_,_,_): ELEM_ARG, STRING, STRING, STRING, STRING, TEXTO, DATE→ ELEM_ARG
m_inclui_elem_posicao(_,_,_,_,_,_,_): ELEM_ARG, STRING, STRING, STRING, STRING, TEXTO, DATE→ ELEM_ARG
m_inclui_elem_argumento(_,_,_,_,_,_,_): ELEM_ARG, STRING, STRING, STRING, STRING, TEXTO, DATE→ ELEM_ARG
m_relacao_elem_arg(_,_,_,_,_): ELEM_ARG, STRING, STRING, STRING, STRING→
ELEM_ARG
m_exclui_elem_arg(_,_): ELEM_ARG, STRING → ELEM_ARG
m_altera_elem_arg_aux(_,_,_,_,_): ELEM_ARG, STRING, STRING, STRING, STRING→ ELEM_ARG
o_ids_questoes(_): ELEM_ARG → CONJUNTO
o_ids_posicoes(_): ELEM_ARG → CONJUNTO
o_ids_argumentos(_): ELEM_ARG → CONJUNTO
o_e_questao(_,_): ELEM_ARG, STRING → BOOL
o_e_posicao(_,_): ELEM_ARG, STRING → BOOL
o_e_argumento(_,_): ELEM_ARG, STRING → BOOL
o_texto_questao(_,_,_): ELEM_ARG, STRING, STRING → TEXTO
o_texto_argumento(_,_,_): ELEM_ARG, STRING, STRING → TEXTO
o_texto_posicao(_,_,_): ELEM_ARG, STRING, STRING → TEXTO
o_argumentos_favor_posicao(_,_,_): ELEM_ARG, STRING, STRING→ CONJUNTO
o_argumentos_contra_posicao(_,_,_): ELEM_ARG, STRING, STRING→ CONJUNTO
o_posicoes_respondem_questao(_,_,_): ELEM_ARG, STRING, STRING→ CONJUNTO
o_questoes_questionam_posicao(_,_,_): ELEM_ARG, STRING, STRING→ CONJUNTO
o_questoes_questionam_argumento(_,_,_): ELEM_ARG, STRING, STRING→ CONJUNTO
o_ids_elems(_): ELEM_ARG → CONJUNTO
o_existe_id_elem (_,_): ELEM_ARG, STRING → BOOLEAN
o_elementos_argumentacao_usuario(_,_):ELEM_ARG, STRING → CONJUNTO
o_elemento_argumentacao_usuario_existe_aux(_,_)ELEM_ARG, STRING → BOOLEAN
o_elementos_argumentacao_datapres(_,_): ELEM_ARG, DATE → CONJUNTO
o_elemento_argumentacao_datapres_existe_aux (_,_):ELEM_ARG, DATE → BOOLEAN
o_elementos_argumentacao_rotulo(_,_): ELEM_ARG, STRIN G → CONJUNTO
o_elemento_argumentacao_rotulo_existe_aux (_,_): ELEM_ARG, STRING → BOOLEAN

Operações

Variáveis formais

v-elem-arg, v-novo-elem-arg : ELEM_ARG
v-idElem, v-busca-idElem, v-rotulo-elem, v-idElem2 : STRING
v-dt-apresentacao : DATE
v-idUsuario, v-idElemRel : STRING
v-questao, v-posicao, v-argumento, v-busca-rotuloElem : STRING
v-textoElem : TEXTO
v-relacao : STRING

c_cria_elem_arg() = empty-mapping

m_inclui_elem_questao(v-elem-arg, v-idElem, v-idUsuario, v-questao, v-rotulo-elem, v-textoElem, v-dt-apresentacao) =

modify (v-idElem,
(IdUsuario v-idUsuario, Tipo Elemento v-questao, Rotulo Elemento v-rotulo-elem , Texto Elemento v-textoElem, Data Apresentacao v-dt-apresentacao)
,
v-elem-arg
)

m_inclui_elem_posicao(v-elem-arg, v-idElem, v-idUsuario, v-posicao, v-rotulo-elem , v-textoElem, v-dt-apresentacao) =
modify (v-idElem,
(IdUsuario v-idUsuario, Tipo Elemento v-posicao, Rotulo Elemento v-rotulo-elem , Texto Elemento v-textoElem, Data Apresentacao v-dt-apresentacao)
,
v-elem-arg
)

m_inclui_elem_argumento(v-elem-arg, v-idElem, v-idUsuario, v-argumento, v-rotulo-elem , v-textoElem, v-dt-apresentacao) =
modify (v-idElem,
(IdUsuario v-idUsuario, Tipo Elemento v-argumento, Rotulo Elemento v-rotulo-elem , Texto Elemento v-textoElem, Data Apresentacao v-dt-apresentacao)
,
v-elem-arg
)

m_exclui_elem_arg (v-elem-arg, v-idElem)=
Restrict_with(v-elem-arg,add(empty-set, v-idElem)

m_relacao_elem_arg(v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)=

if v-relacao = “Questiona” **and ICS**(ELEM_ARG, o_e_questao, v-idElem) and **ICS**(ELEM_ARG, o_e_posicao, v-idElem2)
then **ICS**(ELEM_ARG, m_ altera_elem_arg_aux, v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)
else
if v-relacao = “Apoia” **and ICS**(ELEM_ARG, o_e_argumento, v-idElem) **and ICS**(ELEM_ARG, o_e_posicao, v-idElem2)
then **ICS**(ELEM_ARG, m_ altera_elem_arg_aux, v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)
else
if v-relacao = “Rejeita” **and ICS**(ELEM_ARG, o_e_argumento, v-idElem) **and ICS**(ELEM_ARG, o_e_posicao, v-idElem2)
then **ICS**(ELEM_ARG, m_ altera_elem_arg_aux, v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)
else
if v-relacao= “Responde” **and ICS**(ELEM_ARG, o_e_posicao, v-idElem) **and ICS**(ELEM_ARG, o_e_questao, v-idElem2)
then **ICS**(ELEM_ARG, m_ altera_elem_arg_aux, v-elem- arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)
else *m_relacao_elem_arg*(v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)

m_relacao_elem_arg(empty-mapping,_,_,_,_) = empty-mapping

m_ altera_elem_arg_aux (v-elem-arg, v-busca-idElem, v-idElem, v-idElem2, v-relacao)=
(modify (v-idElem,
(Relacao (Tipo Relacao v-relacao, Elemento v-idElem2)),
v-elem-arg
),
v-busca-idElem,
v-novo-elem-arg)=
if v-busca-idElem=v-idElem
then modify(v-idElem,
(Relacao (Tipo Relacao v-relacao, Elemento v-idElem2)),
v-arg
)
Else modify(v-idElem,
(Relacao (Tipo Relacao v-relacao, Elemento v-idElem2)),
m_ altera_elem_arg_aux (v-idElem, v-busca-idElem, v-idElem2, v-relacao, v-novo-elem-arg)
)

m_altera_elem_arg_aux (empty-mapping,_,_,_,_) = empty-mapping

o_e_questao (v-elem-arg, v-busca-idElem) =

v-busca-idElem ∈ **ICS**(ELEM_ARG, o_ids_questoes, v-elem-arg)

o_e_posicao (v-elem-arg, v-busca-idElem) =
v-busca-idElem ∈ **ICS**(ELEM_ARG, o_ids_posicoes, v-elem-arg)

o_e_argumento (v-elem-arg, v-busca-idElem) =
v-busca-idElem ∈ **ICS**(ELEM_ARG, o_ids_argumentos, v-elem-arg)

o_ids_questoes (v-elem-arg) =
if add(select-Questao(rng(v-elem-arg), empty-set)) = empty-set
then empty-set
Else dom(modify(v-elem-arg,
add(select-Questao(rng(v-elem-arg), empty-set),
empty-mapping))

o_ids_posicoes (v-elem-arg) =
if add(select-Posicao(rng(v-elem-arg), empty-set)) = empty-set
then empty-set
Else dom(modify(v-elem-arg,
add(select-Posicao(rng(v-elem-arg), empty-set),
empty-mapping))

o_ids_argumentos (v-elem-arg) =
if add(select-Argumento(rng(v-elem-arg), empty-set)) = empty-set
then empty-set
Else dom(modify(v-elem-arg,
add(select-Argumento(rng(v-elem-arg), empty-set),
empty-mapping))

o_ids_elems(v-elem-arg) =
ICS(ELEM_ARG, o_ids_questoes, v-elem-arg) ∪
ICS(ELEM_ARG, o_ids_argumentos, v-elem-arg) ∪
ICS(ELEM_ARG, o_ids_posicoes, v-elem-arg)

o_existe_id_elem (v-elem-arg, v-idElem) =
if v-idElem ∈ **ICS**(ELEM_ARG, o_ids_elems, v-elem-arg)
then TRUE
Else FALSE

o_texto_questao(v-elem-arg, v-busca-idElem, v-idElem)=
o_texto_questao (
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
),
v-busca-idElem,
v-novo-elem-arg)=
ifv-busca-idElem=v-idElem **and** v-idElem ∈ **ICS**(ELEM_ARG,
o_ids_questoes, v-elem-arg)

then image_of (v-busca-idElem,
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
))
Else modify(v-idElem,
(Texto Elemento v-textoElem),
o_texto_questao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_texto_questao(empty-mapping,_,_)= empty-mapping

o_texto_argumento(v-elem-arg, v-busca-idElem, v-idElem)=
o_texto_argumento (
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
),
v-busca-idElem,
v-novo-elem-arg)=
ifv-busca-idElem=v-idElem **and** v-idElem ∈ **ICS**(ELEM_ARG, o_ids_argumentos, v-elem-arg)
then image_of (v-busca-idElem,
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
))
Else modify(v-idElem,
(Texto Elemento v-textoElem),
o_texto_argumento(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_texto_argumento(empty-mapping,_,_)= empty-mapping

o_texto_posicao(v-elem-arg, v-busca-idElem, v-idElem)=
o_texto_posicao (
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
),
v-busca-idElem,
v-novo-elem-arg)=
ifv-busca-idElem=v-idElem **and** v-idElem ∈ **ICS**(ELEM_ARG, o_ids_posicoes, v-elem-arg)
then image_of (v-busca-idElem,
modify(v-idElem,
(Texto Elemento v-textoElem),
v-elem-arg
))
Else modify(v-idElem,

(Texto Elemento v-textoElem),
o_texto_posicao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_texto_posicao(empty-mapping,_,_)= empty-mapping

o_argumentos_favor_posicao(
modify(v-idElem,
(Tipo Elemento "Argumento", Relação (Tipo Relação "Apoia", Elemento v-idElemRel)),
v-busca-idElem,
v-elem-arg
),
v-novo-elem-arg))=
if v-idElemRel=v-busca-idElem **and**
ICS(ELEM_ARG, o_e_posicao, v-elem-arg, v-busca-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_argumentos, v-elem-arg) =
then add(v-idElem, empty-set)
Else modify(v-idElem,
(Tipo Elemento "Argumento", Relação (Tipo Relação "Apoia", Elemento v-idElemRel)),
o_argumentos_favor_posicao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_argumentos_favor_posicao (empty-mapping,_,_)= empty-set

o_argumentos_contra_posicao(
modify(v-idElem,
(Tipo Elemento "Argumento", Relação (Tipo Relação "Rejeita", Elemento v-idElemRel)),
v-busca-idElem,
v-elem-arg
),
v-novo-elem-arg))=
if v-idElemRel=v-busca-idElem **and**
ICS(ELEM_ARG,o_e_posicao, v-elem-arg, v-busca-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_argumentos, v-elem-arg) =
then add(v-idElem, empty-set)
Else modify(v-idElem,
(Tipo Elemento "Argumento", Relação (Tipo Relação "Rejeita", Elemento v-idElemRel)),
o_argumentos_contra_posicao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_argumentos_contra_posicao (empty-mapping,_,_)= empty-set

o_posicoes_respondem_questao(

modify(v-idElem,
(Tipo_Elemento “Posicao”, Relação (Tipo_Relação “Responde”, Elemento v-idElemRel)),
v-busca-idElem,
v-elem-arg
),
v-novo-elem-arg))=
if v-idElemRel=v-busca-idElem **and**
ICS(ELEM_ARG, o_e_questao, v-elem-arg, v-busca-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_posicoes, v-elem-arg) =
then add(v-idElem, empty-set)
Else modify(v-idElem,
(Tipo_Elemento “Posicao”, Relação (Tipo_Relação “Responde”, Elemento v-idElemRel)),
o_posicoes_respondem_questao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_posicoes_respodem_questao (empty-mapping,_,_)= empty-set

o_questoes_questionam_posicao(
modify(v-idElem,
(Tipo_Elemento “Questao”, Relação (Tipo_Relação “Questiona”, Elemento v-idElemRel)),
v-busca-idElem,
v-elem-arg
),
v-novo-elem-arg))=
if v-idElemRel=v-busca-idElem **and**
ICS(ELEM_ARG, o_e_posicao, v-elem-arg, v-busca-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_questoes, v-elem-arg) =
then add(v-idElem, empty-set)
Else modify(v-idElem,
(Tipo_Elemento “Questao”, Relação (Tipo_Relação “Questiona”, Elemento v-idElemRel)),
o_questoes_questionam_posicao(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_questoes_questionam_posicao (empty-mapping,_,_)= empty-set

o_questoes_questionam_argumento(
modify(v-idElem,
(Tipo_Elemento “Questao”, Relação (Tipo_Relação “Questiona”, Elemento v-idElemRel)),
v-busca-idElem,
v-elem-arg
),
v-novo-elem-arg))=
if v-idElemRel=v-busca-idElem **and**

ICS(ELEM_ARG, o_e_argumento, v-elem-arg, v-busca-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_questoes, v-elem-arg) =
then add(v-idElem, empty-set)
Else modify(v-idElem,
(Tipo Elemento "Questao", Relação (Tipo Relação "Questiona", Elemento v-idElemRel)),
o_questoes_questionam_argumento(v-idElem, v-busca-idElem, v-novo-elem-arg)
)

o_questoes_questionam_argumento (empty-mapping,_,_)= empty-set

o_elemento_argumentacao_usuario_existe_aux(
modify(v-idElem, (IdUsuario v-IdUsuario), v-elem-arg),
v-busca-idUsuario)=
if v-idUsuario=v-busca-idUsuario
then TRUE
else *o_elementos_argumentacao_usuario_existe_aux*(v-elem-arg, v-IdUsuario)

o_elemento_argumentacao_usuario_existe_aux(empty-mapping,_) = FALSE

o_elementos_argumentacao_usuario(
modify(v-idElem, (IdUsuario v-IdUsuario), v-elem-arg),
v-busca-idUsuario)=
if **ICS**(ELEM_ARG, o_elemento_argumentacao_usuario_existe_aux,
,v-elem-arg, v-busca-idUsuario)
then add(dom(*o_elementos_argumentacao_usuario*(v-elem-arg, v-idUsuario)), v-idElem)
else *o_elementos_argumentacao_usuario*(v-elem-arg, v-idUsuario)

o_elementos_argumentacao_usuario(empty-mapping,_) = empty-set

o_elemento_argumentacao_datapres_existe_aux(
modify(v-idElem, (Data Apresentacao v-dt-apresentacao), v-elem-arg),
v-busca-dt-apresentacao)=
if v-dt-apresentacao = v-busca-dt-apresentacao
then TRUE
else *o_elemento_argumentacao_datapres_existe_aux* (v-elem-arg, v-dt-apresentacao)

o_elemento_argumentacao_datapres_existe_aux (empty-mapping,_) = FALSE

o_elementos_argumentacao_datapres (
modify(v-idElem, (Data Apresentacao v-dt-apresentacao), v-elem-arg),
v-busca-dt-apresentacao)=
if **ICS**(ELEM_ARG, o_elemento_argumentacao_datapres_existe_aux,
,v-elem-arg, v-busca-dt-apresentacao)
then add(dom(*o_elementos_argumentacao_datapres*(v-elem-arg, v-dt-apresentacao)), v-idElem)

else *o_elementos_argumentacao_datapres*(v-elem-arg, v-dt-apresentacao)

o_elementos_argumentacao_datapres(empty-mapping,_) = empty-set

o_elemento_argumentacao_rotulo_existe_aux(
modify(v-idElem, (Rotulo Elemento v-rotulo-elem), v-elem-arg),
v-busca-rotuloElem)=
if v-rotulo-elem = v-busca-rotuloElem
then TRUE
else *o_elemento_argumentacao_rotulo_existe_aux* (v-elem-arg, v-rotulo-elem)

o_elemento_argumentacao_rotulo_existe_aux (empty-mapping,_) = FALSE

o_elementos_argumentacao_rotulo (
modify(v-idElem, (Rotulo Elemento v-rotulo-elem), v-elem-arg),
v-busca-rotuloElem)=
ifICS(ELEM_ARG, o_elemento_argumentacao_rotulo_existe_aux, v-elem-arg, v-busca-rotuloElem)
then add(dom(*o_elementos_argumentacao_rotulo*(v-elem-arg, v-rotulo-elem)), v-idElem)
else *o_elementos_argumentacao_rotulo*(v-elem-arg, v-rotulo-elem)

o_elementos_argumentacao_rotulo(empty-mapping,_) = empty-set

1.12 ATO EXTRACAO

Interface

c_cria_extracao:
m_inclui_extracao(_,_,_,_): EXTRACAO, STRING, DATE, TEXTO → EXTRACAO
m_exclui_extracao(_,_): EXTRACAO, STRING → EXTRACAO
m_insere_alternativa_extracao(_,_,_): EXTRACAO, STRING, STRING → EXTRACAO
m_remove_alternativa_extracao(_,_,_): EXTRACAO, STRING, STRING → EXTRACAO
o_id_alternativas(_,_): EXTRACAO, STRING → CONJUNTO
m_altera_datare_extracao(_,_,_): EXTRACAO, STRING, DATE → EXTRACAO
m_altera_descricao_extracao(_,_,_): EXTRACAO, STRING, TEXTO → EXTRACAO
o_existe_id_extracao(_,_):EXTRACAO, STRING → BOOLEAN
o_id_alternativas_datare(_,_): EXTRACAO, DATE → CONJUNTO
o_pode_ser_alternativa(): ELEM_ARG, STRING, STRING → BOOLEAN

Operações

Variáveis formais

v-extracao, v-novo-extracao : EXTRACAO
v-idExt, v-idElem, v-busca-idExt, v-busca-idElem : STRING
v-descricao, v-nova-descricao : TEXTO

v-datare, v-nova-datare : DATE
v-alternativas : CONJUNTO
v-elem-arg : ELEM_ARG

c_cria_extracao() = empty-mapping

m_inclui_extracao(v-extracao, v-idExt, v-datare, v-descricao)=
modify(v-idExt,
(Data_Realizacao v-datare, Palavras_Chave empty-set, Descricao v-descricao),
v-decisao)

m_exclui_extracao(v-extracao,v-idExt)=
Restrict_with(v-extracao, add(empty-set,v-idExt)

m_insere_alternativa_extracao(
modify(v-idExt,
(Alternativas v-alternativas),
v-extracao),
v-busca-idExt,
v-idElem)
=
if v-busca-idExt=v-idExt **and**
ICS(ELEM_ARG, o_existe_id_elem, v-elem-arg, v-idElem) **and**
v-idElem ∈ **ICS**(ELEM_ARG, o_ids_posicoes, v-elem-arg) **and**
car(**ICS**(ELEM_ARG, o_argumentos_favor_posicao, v-elem-arg, v-idElem, v-busca-idElem) > 1 **and**
v-idElem **not** ∈ **ICS**(EXTRACAO, o_id_alternativas, v-extracao, v-busca-idExt)
then modify(v-idExt,
(Alternativas add(v-alternativas, v-idElem),
v-extracao)
Else modify(v-idExt,
(Alternativas v-alternativas),
m_insere_alternativa_extracao(v-extracao, v-busca-idExt, v-idElem))

m_insere_alternativa_extracao(empty-mapping,_,_) = empty-mapping

m_remove_alternativa_extracao(
modify(v-idExt,
(Alternativas v-alternativas),
v-extracao),
v-busca-idExt,
v-idElem)
=
if v-busca-idExt=v-idExt **and**
v-idElem ∈ **ICS**(EXTRACAO, o_id_alternativas, v-extracao, v-busca-idExt)

```
        then    modify(v-idExt,
                        (Alternativas delete(v-alternativas, v-idElem),
                        v-extracao)
        Else    modify(v-idExt,
                        (Alternativas v-alternativas),
                    m_remove_alternativa_extracao(v-extracao,  v-busca-idExt,  v-
                    idElem))

m_remove_alternativa_extracao(empty-mapping,_,_) = empty-mapping

o_id_alternativas (
modify(v-idExt,
                (Alternativas v-alternativas),
                v-extracao),
        v-busca-idExt)
=
        if      v-busca-idExt=v-idExt and
        then    add(image_of(v-idExt,
                                modify(v-idExt,
                        (Alternativas v-alternativas)),
                                v-extracao)
                        ), empty-set)
        Else    modify(v-idExt,
                        (Alternativas v-alternativas),
                    o_id_alternativas (v-extracao, v-busca-idExt))

o_id_alternativas (empty-mapping,_) = empty-set

o_existe_id_extracao (v-extracao, v-busca-idExt) =
        if      v-busca-idExt ∈ dom(v-extracao)
        then    TRUE
        else    FALSE

m_altera_datare_extracao (
modify(v-idExt,
                (Data Realizacao v-datare),
        v-extracao
        ),
        v-busca-idExt,
        v-novo-datare) =
if      v-busca-idExt=v-idExt
then    modify(v-idExt,
                                (Data Realizacao v-nova-datare),
                        v-extracao
                        )
Else    modify(v-idExt,
                                (Data Realizacao v-datare),
                        m_altera_datare_extracao        (v-extracao,v-busca-idExt,v-nova-
datare)
                        )
```

```
m_altera_datare_extracao (empty-mapping,_,_) = empty-mapping

m_altera_descricao_extracao (
modify(v-idExt,
            (Descricao v-descricao),
        v-extracao
        ),
        v-busca-idExt,
        v-nova-descricao) =
if      v-busca-idExt=v-idExt
then    modify(v-idExt,
                        (Descricao v-nova-descricao),
                    v-extracao
                    )
Else    modify(v-idExt,
                        (Descricao v-descricao),
                    m_altera_descricao_extracao     (v-extracao,v-busca-idExt,v-nova-
                    descricao)
                    )

m_altera_descricao_extracao (empty-mapping,_,_) = empty-mapping

o_id_alternativas_datare (
modify(v-idExt,
            (Data Realizacao v-datare, Alternativas v-alternativas),
        v-extracao
        ),
        v-busca-datare) =
if      v-busca-datare=v-datare
then    add(Selec-Alternativas(
        modify(v-idExt,
            (Data Realizacao v-datare, Alternativas v-alternativas),
        v-extracao
        )), empty-mapping)
Else    modify(v-idExt,
            (Data Realizacao v-datare, Alternativas v-alternativas),
        o_id_alternativas_datare(v-extracao, v-busca-datare))

o_id_alternativas_datare (empty-mapping,_) = empty-set


o_pode_ser_alternativa(v-elem-arg, v-idElem, v-busca-idElem)=
        if      car(ICS(ELEM_ARG, o_argumentos_favor_posicao, v-elem-arg, v-
idElem,                 v-busca-idElem) > 1
        then    TRUE
        else    FALSE
```

1.13 ATO VOTACAO

Interface

c_cria_votacao:
m_insere_voto(_,_,_,_,_,_): VOTACAO, STRING, STRING, STRING, STRING, STRING → VOTACAO
m_anula_voto(_,_,_,_,_): VOTACAO, STRING, STRING, STRING, STRING → VOTACAO
m_remove_voto(_,_): VOTACAO, STRING → VOTACAO
o_id_alternativas_votadas (_): VOTACAO → CONJUNTO
o_votos_anulados (_): VOTACAO → CONJUNTO
o_votos_turno (_,_): VOTACAO, STRING → CONJUNTO
o_votos_data (_,_): VOTACAO, DATE → CONJUNTO
o_votos_usuario (_,_): VOTACAO, STRING → CONJUNTO
o_existe_id_voto(_,_): VOTACAO, STRING → BOOLEAN

Operações

Variáveis formais

v-votacao, v-nova-votacao : VOTACAO
v-turno, v-idVot, v-idUsuario, v-escolha : STRING
v-dataVoto : DATE
v-buscaData, v-buscaUsuario, v-buscaTurno : STRING

c_cria_votacao() = empty-mapping

m_insere_voto(v-votacao, v-idVot, v-turno, v-idUsuario, v-escolha, v-dataVoto)=
if v-escolha <> "Anula Voto" **and**
v-escolha ∈ **ICS**(EXTRACAO, o_id_alternativas, v-extracao, v-escolha) **and**
not *o_existe_id_voto*(v-votacao,v-idVot)
then modify(v-idVot,
(Turno v-turno, Id Usuario v-idUsuario, Escolha v-escolha Data Voto v-dataVoto),
v-votacao)
else empty-mapping

m_anula_voto(v-votacao, v-idVot, v-turno, v-idUsuario, v-dataVoto)=
modify(v-idVot,
(Turno v-turno, Id Usuario v-idUsuario, Escolha "Anula Voto", Data Voto v-dataVoto),
v-votacao)

m_remove_voto(v-votacao,v-idVot)=
Restrict_with(v-votacao, add(empty-set,v-idVot)

```
o_existe_id_voto(v-votacao,v-busca-idVot)=
        if      v-busca-idVot ∈ dom(v-votacao)
        then    TRUE
        else    FALSE

o_alternativas_votadas(
          modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
          v-dataVoto),
                v-votacao) =
          if    Selec-Escolha(rng(modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
          v-dataVoto),
                v-votacao))) <> "Anula Voto"
          then  add(Selec-Escolha(rng(modify(v-idVot,
                      (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data
                      Voto v-dataVoto),
                v-votacao))),
                        o_alternativas_votadas(v-votacao))
        else    o_alternativas_votadas(v-votacao))

o_alternativas_votadas(_): empty-set


o_votos_anulados(
          modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
          v-dataVoto),
                v-votacao) =
          if    Selec-Escolha(rng(modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
          v-dataVoto),
                v-votacao))) = "Anula Voto"
          then  add(dom(modify(v-idVot,
                      (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data
                      Voto v-dataVoto),
                v-votacao)),
                o_votos_anulados(v-votacao))
        else    o_votos_anulados(v-votacao))

o_votos_anulados(_): empty-set

o_votos_turno(
          modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
          v-dataVoto),
                v-votacao),
                v-buscaTurno) =
          if    Selec-Turno(rng(modify(v-idVot,
```

```
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
            v-dataVoto),
                v-votacao))) = v-buscaTurno
            then add(dom(modify(v-idVot,
                        (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data
                        Voto v-dataVoto),
                    v-votacao)),
                    o_votos_turno(v-votacao))
          else  o_votos_turno(v-votacao))

o_votos_turno(_): empty-set

o_votos_data(
            modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
            v-dataVoto),
                v-votacao),
                v-buscaData) =
            if  Selec-Data Voto(rng(modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
            v-dataVoto),
                v-votacao))) = v-buscaData
            then add(dom(modify(v-idVot,
                        (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data
                        Voto v-dataVoto),
                    v-votacao)),
                    o_votos_data(v-votacao))
          else  o_votos_data(v-votacao))

o_votos_data(_): empty-set

o_votos_usuario(
            modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
            v-dataVoto),
                v-votacao),
                v-buscaUsuario) =
            if  Selec-Id Usuario(rng(modify(v-idVot,
                (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data Voto
            v-dataVoto),
                v-votacao))) = v-buscaUsuario
            then add(dom(modify(v-idVot,
                        (Turno v-turno,  Id Usuario v-idUsuario, Escolha v-escolha,  Data
                        Voto v-dataVoto),
                    v-votacao)),
                    o_votos_usuario(v-votacao))
          else  o_votos_usuario(v-votacao))

o_votos_usuario(_): empty-set
```

Bibliografia

[ALV 2000] ALVES, R.; NUNES, D.J. Uma Ferramenta para Apoiar Decisões de Grupo para o Ambiente PROSOFT. In: SEMANA ACADÊMICA DO PPGC, 5.,2000, Porto Alegre. **Anais**... Porto Alegre: PPGC da UFRGS, 2000.

[ARA 94] ARAUJO, Renata. **QUORUM:** Um Sistema de Suporte à Decisão em Grupo para o Desenvolvimento de Software. 1994. Dissertação (Mestrado) – COOPE, UFRJ, Rio de Janeiro.

[ARA 95] ARAUJO, Renata; FUKS, Hugo. Quorum-w: A Group Decision Support Tool for the internet environment. In: CYTED-RITOS INTERNATIONAL WORKSHOP ON GROUPWARE,1.,1995, Lisboa. **Proceedings of CRIGW 95**. Lisboa: Junta Nacional de Investigação Cientifica e Tecnológica, 1995. p.39-52.

[BAC 97] BACELO, T. P. A.; BECKER KARIN. **GRADD**: uma implementação Lotus-Domino de um Sistema de Reuniões Remotas. Porto Alegre: PUC-RS, 1997.

[BAN 93] BANDINELLI, S.; FUGGETTA, A.; GRIGOLLI, S. Process Modeling in-the-large with SLANG. In: INTERNATIONAL CONFERENCE ON SOFTWARE PROCESS, 1993. 2. **[Proceedings...]**, Los Alamitos: IEEE Press. 1993. p.75-83.

[BEL 95] BELASSAI, G. et al. SISCO: A tool to improve meetings productivity. In: CYTED-RITOS INTERNATIONAL WORKSHOP ON GROUPWARE,1.,1995, Lisboa. **Proceedings of CRIGW 95**. Lisboa: Junta Nacional de Investigação Cientifica e Tecnológica, 1995. p.149-161.

[BEL 96] BELASSAI, G. et al. SISCO: An IBIS Based Model to Support Group Discussions. In: **IFIP WG8**. 1996. **[Proceedings...]**. [S.l.: s.n.]., 1996.

[BEM 93] BENBASAT, Izak; LIM, Lai-Huat. The Effects of Group, Task, Context, and the Technology Variables on the Usefulness of Group Support Systems. **Small Group Research**., [S.l.], v.24, p. 430-462, 1993.

[BJØ 78] BJØRNER, D.; JONES, C.B. **The Vienna Development Method:** the meta-language. Berlin: Springer-Verlag, 1978. Lecture Notes in Computer Science.

[BOR 95] BORGES, M.R.S.; CAVALCANTI, M.C.R.; CAMPOS, M.L.M. **Suporte por Computador ao Trabalho Cooperativo**. Porto Alegre: Instituto de Informática da UFRGS, 1995. Curso ministrado na Jornada de Atualização em Informática, 14., 1995.

[BUI 95] BUI, Tung; YEN, Jerome. The Negotiable Alternatives Identifier (NAI) for Negotiation Support – An improved Algorithm. In: INTERNATIONAL CONFERENCE OF DECISION SUPPORT

SYSTEMS, 3., 1995. **Proceedings of Third International Conference of Decision Support Systems**. [S.l.: s.n.]., 1995. p.149-159.

[CAP 92] CARMO, P. R.S. **ATO Diretório**. Porto Alegre: PGCC-UFRGS, 1992. Relatório de Pesquisa interno.

[CES 94] CESAR, Flavio Lenz; WAINER, Jacques. vIBIS: A Discussion and Voting System. **Journal of Brazilian Computer Science**, Rio de Janeiro, n1, p. 36-43, 1994.

[CES 96] CESAR, Flavio Lenz. **vIBIS:** um modelo de Discussão e Votação. 1996. Dissertação (Mestrado) – UNICAMP, Campinas.

[CON 88] CONKLIN, Jeff; BEGEMAN, Michael L. gIBIS: A Hypertext Tool for Exploratory Policy Discussion. **ACM Transactions on Office Information Systems**, New York, n.6, p.303-331, 1988.

[COO 82] COOK, Wade D. and SEIFORD, Lawrence M. Borda-Kendall Consensus Method for Priority Ranking Problems. **Management Science**, New York, v.38, n.6, p.621-637, 1982.

[COR 94] CONRADI, R. et al. EPOS: Object-Oriented Cooperative Process Modelling. In: FINKELSTEIN, A. et al. (Ed.). **Software Process Modelling and Technology**. Somerset: John Wiley, 1994. p.33-70.

[CUR 92] CURTIS, B. et al. Process Modeling. **Communications of ACM**, New York, v.35, n.9, p.75-90, 1992.

[DAU 92] DAUDT, R. **Uma Proposta de Extensão do PROSOFT Básico.** 1992. Trabalho de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[DES 87] DESACNTIS, Gerardine.; GALLUPE, Brent. A Foundation for the Study of Group Decision Support System. **Management Science**, New York, n.33, p.589-609, 1987.

[DOW91] DOWSON, M.; NEJMEH, B.; RIDDLE, W. Fundamental Software Process Concepts. In: EUROPEAN WORKSHOP ON SOFTWARE PROCESS TECHNOLOGY, EWSPT, 1991. **Proceedings**... Milan, Italy. May, 1991.

[EAS 92] EASTON, ANNETTE C et al. Interactive Versus Stand-Alone Group Decision Support Systems for Stakeholder Identification and Assuption. **Decision Support Systems**, New York, v.8, n.2, p.159-168, 1992.

[ELL 91] ELLIS, C. A.; GIBBS, S. J.; REIN, G. L. GROUPWARE. Some issues and experiences. **Communication of the ACM**, New York, n.34, p.38-58. 1991.

[GRA 84] GRATH MC, J. E. **Groups:** Interaction and Performance. New York: Prentice Hall, 1984

[KAR 96] KARAN, V. et al. Information Technology Support for Collaborative Decision Making in Auditing: An Experimental Investigation. **Decision Support Systems**, New York, n.16, p.181-194, 1996.

[KRA 88] KRAEMER, KENNETH, KING, JOHN. Computer Based Systems For Cooperative Work and Group Decision Making. **ACM Computing Surveys**, New York, n.5, p.115-146, 1988.

[KRU 94] KRUEGER, R. A. **Focus Group:** A Practical for Applied Research. 2nd ed. Thousands Oaks: Sage Publicantions, 1994.

[LIM 98] LIMA REIS, C.A. Um gerenciador de processos de software para o ambiente PROSOFT. 1998. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[LON 96] LONCHAMP, J. A Kernel for Building Collaborative Process Centered Environment. **Proceedings SEE96**. [S.l.: s.n.]., 1996. p.95-105.

[LUC 91] LUCENA, C. J. P et al. **A Research Agenda on Software Design**. Rio de Janeiro: PUC/RJ, 1991. (Monografias em Ciência da Computação. n.29/91).

[MAR 99] MARCOS, A. et al. Ensino à Distância do Teletrabalho através das Tecnologias de Multimídia. In: WORKSHOP COMPUTAÇÃO GRÁFICA, MULTIMÍDIA E ENSINO, 1., 1999, Coimbra. **[Anais...]**. [S.l.: s.n.]., 1999

[MOR 88] MORGAN, D. L. **Focus Group as Qualitative Research**. Beverly Hills: SAGE Publications, 1988.

[MOR 97] MORAES, S. M. N. Um Ambiente Expert para Apoio ao Desenvolvimento de Software. 1997. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[NAS 73] NASSI, I; SCHNEIDERMAN, B. Flowchart Techniques for Structured Programming. **SIGPLAN Notices**, New York, v.1, n.5, p.36-76, Aug. 1973.

[NUN 92] NUNES, D. J. Estratégia Data-Driven no Desenvolvimento de Software. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, 6., 1992, Gramado. **Anais**... Porto Alegre: Instituto de Informática da UFRGS, 1992. v.1, p. 81-95.

[NUN 94] NUNES, D. J. **PROSOFT:** Descrição do ambiente. Relatório de Pesquisa, Porto Alegre: PPGC-UFRGS, 1994. Relatório de Pesquisa.

[POT 88] POTTS, C.; BRUNS, G. **Recording Decisions for Design Decisions**. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 10., 1988. **[Proceedings...]**. [S.l.: s.n.]., 1988. p.12-16.

[REI 98] REIS, R.Q. **Uma Proposta de Suporte ao Desenvolvimento Cooperativo de Software no Ambiente Prosoft**. 1998. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[RIB 91] RIBEIRO, LEILA. **Integração no PROSOFT de ambientes corretos obtidos a partir de especificações algébricas e executadas usando sistemas de reescrita**. 1991. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[ROS 85] ROSS, D. Applications and Extensions of SADT. **Computer**, New York v. 18, n. 4, 1985.

[SCH 97] SCHLEBBE, H.; SCHIMPF, S. **Reengineering of PROSOFT in Java**. Stuttgart: Institut für Informatik, Universitat Sttutgart, 1997. 15 p.

[SHU 94] SHUM, S.; HAMMOND, N. Argumentation Based Design Rationale: What use at what cost? **International Journal of Human Computer Studies**, New York, v.5, n.40, p.75-90, 1994.

[SIL 2001] SILVA, F.A.D. **Um Modelo de Simulação de Processos de Software Baseado em Conhecimento para o Ambiente PROSOFT**. 2001. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[SIL 91] SILVER, Mark S. **System that Support Decision Makers**. USA: John Wiley & Sons, 1991.

[SIM 60] SIMON, H . A. **The New Science of Management Decision**. New York: Harper and Row, 1960.

[STO 92] STOHR, E.; KONSYNSKY, B. **Information Systems and Decision Process**. New York: IEEE Computer Society Press, 1992.

[STR 80] STRAFFIN, P. D. **Topics in the Theory of Voting**. Stuttgart: UMAP Expository Monograph Series, 1980.

[WAT91] WATT, D. **Programming Language Syntax and Semantics**. New York: Prentice-Hall, 1991.

[YAK 90] YAKEMOVIC, B. K. C ; CONKLIN, E. J. Report on Development Project Use of an Issue Based Information System. In: CONFERENCE ON COMPUTER SUPPORTED COOPERATIVE WORK.1990. **[Proceedings...]** [S.l.:s.n.],1990. p.105-118.