LÉLIS TETSUO MURAKAMI

SOLUÇÃO DE PROBLEMAS DE OTIMIZAÇÃO
UTILIZANDO ARQUITETURA HÍBRIDA

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção do
título de Doutor em Engenharia

São Paulo
2008

LÉLIS TETSUO MURAKAMI

SOLUÇÃO DE PROBLEMAS DE OTIMIZAÇÃO
UTILIZANDO ARQUITETURA HÍBRIDA

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção do
título de Doutor em Engenharia

Área de Concentração:
Engenharia de Computação e Sistemas Digitais

Orientador: Prof. Titular
José Sidnei Colombo Martini

São Paulo
2008

FICHA CATALOGRÁFICA

Murakami, Lélis Tetsuo

Solução de problemas de otimização utilizando arquitetura híbrida / L.T. Murakami. -- São Paulo, 2008.

146 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Modelos de otimização 2.Modelos de geração de energia 3.Sistemas distribuídos 4.Arquitetura híbrida I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

DEDICATÓRIA

“ ...

Far better it is to dare mighty things, to win glorious triumphs, even though checkered by failure, than to take rank with those poor spirits who neither enjoy much nor suffer much, because they live in the gray twilight that knows not victory nor defeat...

...“

THEODORE ROOSEVELT
26th President of the United States of America
“THE STRENUOUS LIFE”, April 10, 1899 -
The Hamilton Club, Chicago

“ ...

É melhor lançar-se à luta em busca de triunfos, mesmo que sujeito ao fracasso, do que se juntar aos pobres de espírito que não festejam nem sofrem muito porque vivem na penumbra cinzenta dos que não conhecem vitória nem derrota.

...“

O conhecimento é um bem durável e junto com a evolução moral, faz parte do patrimônio daquele que o adquiriu. A sua busca é marcada por sacrifícios de dimensões intangíveis. Quem se embrenhou nesta empreitada sabe o quanto foi árdua esta missão que extrapola em muito seu próprio domínio, interferindo nas pessoas ao seu redor.

Dedico este trabalho à esposa e filhos que pacientemente aguardaram o término desta caminhada, privando-se muitas vezes do convívio em família.

AGRADECIMENTOS

À CESP – Cia Energética de São Paulo pela anuência em desenvolver este trabalho no âmbito de seus projetos e pela cessão de dados e informações;

Aos professores, em especial Prof^a. Dr^a. Líria Matsumoto Sato e Prof. Dr. Edison Spina pela relevante contribuição recebida;

Ao Prof. Dr. José Sidnei Colombo Martini pela preciosa orientação neste trabalho;

A todos que direta ou indiretamente me incentivaram a prosseguir principalmente nas horas mais difíceis, o meu sincero agradecimento.

RESUMO

A energia elétrica constitui um dos propulsores da economia de um país, assumindo um papel extremamente importante e estratégico, pois influi diretamente na capacidade produtiva. A expansão da produção de energia elétrica não se consegue somente com medidas de curto prazo, pois as obras deste setor demandam um longo tempo de execução, medido em anos e dependendo da magnitude da obra, o prazo pode até superar uma década.

O parque gerador nacional é constituído predominantemente por usinas hidroelétricas, complementado por usinas térmicas que utilizam diversas fontes de combustível, havendo a necessidade de minimizar a produção das térmicas, em virtude do alto custo de geração, em relação ao custo de geração hidroelétrica.

Garantir o suprimento da demanda futura de energia elétrica é uma tarefa complexa de planejamento que basicamente, depende da análise de dois cenários que se compõem: o primeiro cenário é o que desenha o crescimento futuro da economia e neste caso, desde que não ocorram fatos extraordinários como o recente crescimento econômico experimentado pela China, a previsão da demanda não acarreta surpresa de grande significância; o segundo cenário, traz como característica a incerteza, uma vez que a produção das hidroelétricas depende da quantidade de água disponível dos cursos de água, que por sua vez, depende do regime de chuvas passado e corrente. O índice pluviométrico é um dado estocástico, ocorrendo ao sabor da probabilidade, o que remete a um estudo de casos e seus desdobramentos, acarretando um leque de possibilidades de estados muito grande, dificultando as análises sobre a previsão futura.

Planejar o setor elétrico compreende prever um crescimento de demanda e equipar o setor com máquinas de geração, necessárias para atender a demanda, com uma margem de risco calculada. Para isto, utilizam-se modelos de simulação que possibilitem o exercício de previsão, combinando-se os dois cenários citados, visualizando os estados sub seqüentes, decorrentes de decisões tomadas. A dificuldade desta tarefa é devida à quantidade de alternativas da situação futura, resultante de um fenômeno combinatório de possibilidades que exige para simulação dos modelos, não só uma grande capacidade de processamento dos computadores

como também, uma estratégia de tratamento do problema, baseada em processos numéricos especializados e dirigidos a este tipo de problema.

Dada a importância e magnitude deste assunto, qualquer esforço que venha a contribuir para uma melhoria do planejamento do setor elétrico, traz benefícios significativos, o que corrobora com os propósitos desta tese, que busca em primeiro lugar, propor soluções técnicas viáveis e econômicas para o problema de otimização da geração de energia elétrica, e em segundo lugar, apresentar uma solução para este tipo de problema, com uma abordagem inovadora, provida de um potencial significativo para aplicação em muitos outros tipos de problemas similares.

Palavras-chave: Modelos de otimização, Modelos de geração de energia, Sistemas distribuídos, Arquitetura Híbrida.

ABSTRACT

Electrical power could be considered as one of the economy propulsion vector of a country, assuming extremely important and strategic role because it makes direct influence to the production capacity. The expansion of electrical energy production could not only be done in a short time because constructions in this area take many years and could require more then a decade depending on the magnitude of them.

The national power generation group is constituted mainly by hydro power plants complemented by thermal power plants which use several kinds of fuel which generation cost is high, if compared to hydro power generation, and should be minimized.

It is a complex planning issue to supply the future power demand which basically depends on the analysis of two compoundable scenarios: the first one refers to the forecast of future economy growing and in this case, unless unpredicted issues occur such as the recent high economy growing experimented by China, the future demand does not show any surprise and is easy to predict; the second one, has inside the uncertainty because the hydro plants productions depends on the water quantity of rivers which depends on the past and current rainfall regimen. The quantity of rainfall is a stochastic data and follows the rules of probability and this drives to the study of cases and its deployments which are numerous causing difficulties to forecast the future.

The planning of the electrical area has to examine the future demand and provide the necessary power generation equipment assuming a certain risk. To have it done, simulation models are used to predict the future, combining the two scenarios cited before, and viewing the results promoted by decision took in a step before. The difficult of this task is caused by the big amount of future alternatives provided by the combinatorial phenomena which require, to process the model, a computer with high processing capacity and specialized and specific methods that can resolve this kind of problem.

Because of the importance and magnitude of this issue, every effort which contributes to the improvement of power planning is welcome and this corroborates with this

thesis which has an objective to propose technical, viable and economic solutions to solve the optimization problems with a new approach and has potential to be applied in many others kind of similar problems.

Key words: Optimization Models, Power Generation Models, Distributed Systems, Hybrid Architecture.

LISTA DE ILUSTRAÇÕES

Figura 2-1 Sistema de Transmissão - Fonte ONS	6
Figura 3.2-1 Princípio de Optimalidade de Bellman.....	21
Figura 3.2-2 Prova	21
Figura 3.2-3 Protocolo de Comunicação FIPA.....	24
Figura 5.2-1 Fluxo de Processo do QFD	56
Figura 5.5-1 Rede P2P Híbrida.....	76
Figura 5.5-2 Arquitetura JADE	77
Figura 5.6-1 Diagrama de Intercâmbio - Fonte CESP	79
Figura 5.6-2 Estágios.....	83
Figura 6.1-1 Especificação Estrutural ASIC da Sociedade "Mercado de Energia"	89
Figura 6.2-1 Esquema Genérico de um Sistema Gerador	96
Figura 6.3-1 Balanço no nó i.....	101
Figura 6.3-2 Balanço Hídrico	103
Figura 6.3-3 Janela dos Agentes	110
Figura 6.3-4 Plataforma Eclipse.....	110

LISTA DE TABELAS

Tabela 5.2-1 Desdobramento dos Requisitos.....	57
Tabela 5.2-2 Valores de Importância Relativa	58
Tabela 5.2-3 Matriz de Importância Relativa	59
Tabela 5.2-4 Matriz de Importância Relativa Normalizada	60
Tabela 5.2-5 Requisitos em ordem de importância.....	61
Tabela 5.3-1 Casa da Qualidade	63
Tabela 5.3-2 Requisitos de Projeto Ordenados	64
Tabela 5.4-1 Matriz MC-HSQ	66
Tabela 5.4-2 Pontuação MC-HSQ	68
Tabela 5.4-3 Processos Numéricos	69
Tabela 6.3-1 Componentes do Protótipo	97
Tabela 6.3-2 Casos de Simulação da Vazão $i+1$	101
Tabela 7-1 Estágio 3.....	115
Tabela 7-2 Estágio 46.....	116
Tabela 7-3 Estágio 50.....	117
Tabela 7-4 Estágio 51.....	118
Tabela 7-5 Evolução das Vazões em UFD1	120
Tabela 7-6 Evolução das Vazões em URE1.....	121
Tabela 7-7 Evolução das Vazões e Volumes em URE2.....	122
Tabela 7-8 Evolução das Vazões e Volumes em URE3.....	123
Tabela 7-9 Tabela de Composição do Custo de Geração Hidráulica	124

LISTAS DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
ASIC	An architecture for social and individual control
BB	Branch and Bound
BI	Business Intelligence
CC-NUMA	Cache-Coherent NUMA
CHP	Combined Heat and Power Production
CMMI	Capability Maturity Model Integration
Cobit	Control Objectives for Information and related Technology
CORBA	Common Object Request Broker Architecture
CPU	Central Process Unit
CRM	Customer Relationship Management
DP	Dynamic Programming
DSS	Decision Support System
EPRI	Electric Power Research Institute
ES	Expert System
FIPA	Foundation for Intelligent Physical Agents
FT	Fuzzy Theorem
GA	Genetic Algorithm
GS	Generation Scheduling
HCT	Hydro Thermal Coordination
HD	Hard Disc
HOQ	House of Quality
HPC	High Performance Computer
IDL	Interface Definition Language
IIOB	Internet Inter – ORB Protocol
ISACF	Information Systems Audit and Control Foundation
ITIL	Information Technology Infrastructure Library
JVM	Java Virtual Machine
LAN	Local Area Network
LP	Linear Programming
LR	Lagrange Relaxation

MC – HSQ	Modelo Conceitual para Solução de Problemas de Otimização
MIP	Mixed Integer Programming
MPI	Message Passing Interface
MPP	Massive Parallel Processing
NP-Hard	Classe de problemas complexos
NUMA	Non Uniform Memory Access
OMG	Object Management Group
ONS	Operador Nacional do Sistema
OO	Object Oriented
ORB	Object Request Broker
OR/MS	Operational Research / Management Sciences
PATA	Parallel ATA
PC	Personal Computer
PMO	Programa Mensal da Operação
PO	Pesquisa Operacional
PVM	Parallel Virtual Machine
QFD	Quality Function Deployment
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
Rpm	Rotações por minuto
RUP	Rational Unified Process
SA	Simulated Annealing
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SEI	Software Engineering Institute
SMA	Sistema Multi Agentes
SMP	Symmetric Multi Processing
TCP-IP	Transmission Control Protocol - Internet Protocol
TI	Tecnologia da Informação
TQM	Total Quality Management
TS	Tabu Search
TSP	Team Software Process
UC	Unit Commitment

UCP	Unidade Central de Processamento
UHE('s)	Usina(s) Hidroelétrica(s)
UML	Unified Modeling Language
UTE('s)	Usina(s) Termoelétrica(s)
UTN	Usina Termoelétrica Nuclear
WAN	Wide Area Network
WWW	World Wide Web

SUMÁRIO

1. Introdução.....	3
2. Abordagem do Problema.....	6
2.1. Motivação.....	8
3. Conceitos.....	12
3.1. Problemas NP-Hard	12
3.2. Processos Numéricos (Software).....	16
3.3. Lista de Requisitos (Qualidade de Software)	28
3.3.1. Estratégia de Implementação da Qualidade em Software	32
3.3.2. Aspectos de Qualidade em Algoritmos	33
3.3.3. QFD – Quality Function Deployment.....	37
3.4. Infra-Estrutura (Hardware)	40
4. Estado da Arte	49
5. Desenvolvimento do Tema	53
5.1. Ineditismo.....	54
5.2. Lista de Requisitos.....	56
5.3. House of Quality.....	62
5.4. Modelo MC - HSQ.....	65
5.5. Arquitetura do Sistema.....	75
5.6. Descrição do Sistema de Geração de Energia	79
6. Implantação do Protótipo.....	86
6.1. Definição do SMA – Sistema Multi Agentes.....	86
6.2. Pesquisa Operacional	92
6.3. Implementação do Protótipo	97
7. Resultados Obtidos	114

8.	Conclusões	126
9.	Considerações Finais	128
10.	Bibliografia Citada	131
11.	Bibliografia Consultada	135
12.	Sites Consultados	138

1. INTRODUÇÃO

O planejamento energético é uma atividade, que alimenta o sistema de informações de uma empresa de geração de energia elétrica. Tem por objetivo, suprir áreas estratégicas da empresa com informações relevantes que possibilitem decidir sobre esta, ou aquela alternativa técnica e gerencial.

O objeto de negócio de uma empresa de geração de energia constitui-se da venda de *megawatts*, através de contratos firmados, ou pela venda no mercado *spot*. Os contratos são negociados em função de diversas informações, sobre disponibilidades e preços no mercado de energia. Surge então, a necessidade de possuir informações fidedignas para, baseado nelas, tomar boas decisões.

Faz parte também do planejamento, decidir sobre os investimentos necessários, seja em termos de novos empreendimentos de geração, ou em termos de novos equipamentos, aumentando a produção.

As retiradas de equipamentos, simulações de entrada em operação de máquinas, cronogramas de construção de novas usinas, construção de térmicas, operações especiais etc., são todos assuntos ligados ao planejamento e que necessitam de dados, para efetuar as simulações.

Dentro deste contexto de planejamento, identifica-se uma atividade importante referente ao comportamento futuro do sistema de produção de energia, obtido através de simulações, utilizando dados históricos e adotando-se diversos cenários da situação hidrológica. As simulações têm como objetivo, antever a demanda futura em cada sub sistema, otimizando o potencial hídrico e ao mesmo tempo, minimizando o despacho de usinas térmicas, devido ao seu alto custo de geração. As Usinas Hidro Elétricas - UHEs presentes no sistema tem outras finalidades, além da geração de energia, que precisam ser equacionadas, como é o caso dos reservatórios de acumulação, que também agregam a função de controle de cheias, navegação etc. Constitui-se também como uma das funcionalidades, a garantia da vazão de jusante, devido a fatores não energéticos, o que

corresponde a uma restrição operacional. Para atender às demandas, é preciso verificar a possibilidade de intercâmbio entre os diversos sub sistemas, pois existem limitações na capacidade de transmissão, além das perdas.

A simulação da geração de energia elétrica futura tem no seu bojo, um problema de otimização que apresenta uma complexidade elevada, sendo necessário, a utilização de processos numéricos específicos, pois constituem um tipo de problema muito conhecido, pela dificuldade de obter a solução ótima, devido ao fenômeno da explosão combinatória¹. Muitas idéias são sugeridas para resolver este tipo de problema, seja através da simplificação, transformando-o em problemas menores e conseqüentemente perdendo em qualidade ou então, descartando a solução ótima e aceitando uma solução considerada boa.

Os recursos computacionais, disponíveis para desenvolver o modelo de simulações são abundantes e proporcionam muitas alternativas de infraestrutura e processos numéricos, criando uma nova dificuldade sobre a escolha da melhor alternativa para compor o modelo de simulações. Este último tendo como finalidade o uso em simulações de planejamento, necessita que determinados requisitos sejam imprescindíveis em sua especificação. A maioria dos requisitos refere-se à qualidade, principalmente no quesito usabilidade².

O desenvolvimento do modelo de simulação depende das características dos recursos de computação, disponíveis no momento, e de suas potencialidades para executar os processos numéricos. É necessário conhecer com detalhes, qual é o estado da arte da computação e quais são

¹ Explosão combinatória é a característica de um problema em possuir diversas alternativas de solução que ocorrem em forma de ramificações em árvores, podendo ser binária (duas ramificações em cada nó) ou não (mais de uma ramificação).

² É um item de qualidade e tem por objetivo avaliar a facilidade com que um Usuário consegue executar uma determinada funcionalidade de um objeto, podendo inclusive ser um objeto de software.

as tendências, para atribuir maior ou menor capacidade de processar as alternativas pesquisadas. Algumas questões envolvendo o processamento paralelo ou distribuído precisam ser investigadas, para analisar a viabilidade das alternativas possíveis e a adequação a cada necessidade. Por este motivo, elaborou-se inicialmente uma verificação ampla e detalhada sobre os recursos computacionais atualmente disponíveis, identificando a situação em que se encontra a computação, para em seguida, analisar de que forma o potencial de processamento pode colaborar, junto com os recursos de software, na criação de um artefato de qualidade.

A solução proposta nesta tese apresenta uma formatação inovadora, onde direciona os esforços em três eixos, Hardware, Software e Qualidade, de forma controlada, obtendo uma solução de equilíbrio, através de um modelo conceitual criado especificamente para avaliar os diversos processos numéricos, e as correspondentes infra-estruturas requeridas, atendendo aos requisitos de qualidade estabelecidos para a solução.

Em relação ao Hardware, o processamento distribuído foi conseguido através de um recurso de software denominado de middleware, possibilitando uma abertura para implementação de processamento escalável, com baixos custos de implantação.

Os resultados obtidos comprovaram, através da implantação de um protótipo genérico, a validade da estratégia utilizada, demonstrando que existem alternativas de qualidade, para a solução de problemas complexos, como é o caso da otimização de geração de energia elétrica.

Os capítulos seguintes apresentam os conceitos inerentes ao trabalho, o modelo criado para estabelecer a solução de equilíbrio, finalizando com a implementação de um protótipo.

2. ABORDAGEM DO PROBLEMA

De acordo com os dados da PMO – Programa Mensal de Operação de Março de 2007 fornecidos pelo ONS – Operador Nacional do Sistema, órgão responsável pela coordenação da geração e transmissão de energia elétrica no Sistema Integrado Nacional, a simulação do setor energético brasileiro é efetuada com 146 usinas de geração hidroelétrica. Dessas, 118 são existentes e o restante das usinas, programados para entrar em operação no futuro. As usinas térmicas consideradas são 85 existentes e 35 em implantação. A figura 2-1 seguinte ilustra a rede de transmissão.

Figura 2-1 Sistema de Transmissão - Fonte ONS

O parque gerador é subdividido em 4 regiões, Sudeste, Sul, Nordeste e Norte, interligados através de linhas de transmissão. O grande desafio do

setor elétrico consiste em manter o sistema eletro-energético em funcionamento, de acordo com os padrões de qualidade e confiabilidade estabelecidos, atendendo a demanda e garantindo o suprimento ao longo do tempo. Para isso, utiliza as suplementações através de unidades de geração térmica, movidas a combustível de diversos tipos e custos variados. O *ONS – Operador Nacional do Sistema* é o órgão que tem por atribuição, garantir que as metas sejam atingidas ao mínimo custo.

Estabelecer a geração em cada usina ao mínimo custo é um problema de grande amplitude e de difícil solução, por apresentar alta complexidade, em função da grande quantidade de alternativas possíveis. Há duas características importantes neste caso: a) Trata-se de um problema multi-estágios onde a ação tomada em um estágio influi nos resultados do estágio seguinte; b) A geração nas usinas hidroelétricas depende da disponibilidade da água em rios e reservatórios, cuja previsão é não determinística, tornando o problema estocástico.

Há duas atividades importantes no setor energético que demandam estudos especiais:

- Uma primeira atividade tem como objetivo a geração de curto prazo, através do monitoramento mensal, semanal e diário, considerando as restrições operacionais vigentes em cada uma das usinas. No estudo mensal, são efetuadas todas as análises sobre a demanda e sobre os impedimentos operativos, configurando a solução de menor custo. Na programação semanal, o programa de geração é detalhado e monitorado, definindo a operação diária nas usinas.
- Uma segunda atividade diz respeito ao planejamento do setor. Tem como metas, além do atendimento à demanda, elaborar estudos sobre os cenários futuros de demanda, estabelecer as prioridades de investimentos, reprogramar a capacidade de geração em usinas e estabelecer a expansão da transmissão.

O planejamento do setor energético apresenta grandes desafios, não só sob o ponto de vista da solução do problema, mas também sob o ponto de vista da magnitude dos investimentos envolvidos. Em virtude desta dimensão, muitas pesquisas tem sido elaboradas, em busca de métodos que possibilitem soluções eficientes para o problema da programação de geração em sistemas de potência. Sob o ponto de vista computacional, as alternativas de solução observadas na bibliografia apresentam propostas de soluções diversificadas e em grande número, com o objetivo de resolver o problema de otimização, que continua a apresentar um grande desafio na busca de soluções eficientes.

A abordagem desta tese considera essencialmente a busca por uma solução inovadora, que permita obter bons resultados em tempos de processamento adequados, de modo a constituir-se em uma ferramenta útil e eficiente, servindo como um recurso adicional de qualidade para enfrentar o problema de planejamento do setor elétrico.

2.1. Motivação

Faz parte da natureza humana, a busca por desafios impulsionando a conquista de novos horizontes. Dada a complexidade inerente aos desafios, torna-se necessário analisar as peculiaridades de cada caso e traçar uma boa estratégia, para vencer os obstáculos que estarão presentes ao longo do caminho.

Como intróito para discorrer sobre o assunto é oportuno iniciar com um relato breve sobre Computação, recentemente renomeada para *TI - Tecnologia da Informação*, uma vez que este tópico tem estreita ligação com o desafio deste trabalho.

Poucas décadas atrás, a computação era concentrada em *bureaus* onde poucas pessoas capacitadas atuavam neste segmento. A formação profissional era obtida através da dedicação independente de cada pessoa, através de cursos específicos patrocinados pelos proprietários de hardware

ou software. Não havia nessa época, cursos universitários focados nesta área como existem hoje, inclusive com especializações.

Dois fatos importantes influíram no avanço exponencial da Computação: o evento do *downsizing* que provocou através da economia de escala a redução do valor de aquisição de recursos computacionais e que por sua vez, deflagrou um processo de disseminação da computação; o segundo ponto é o fenômeno mundial decorrente da rede www, ao qual denominamos de Internet, que constitui hoje um meio de comunicação e troca de arquivos dos mais utilizados. Com esta facilidade, a informação se espalha rapidamente e crescem de maneira incisiva, os servidores de domínio, repositórios de arquivos, servidores de *e-mails* e tudo que tenha relacionamento ou interesse com a interação entre pessoas e empresas.

Destas considerações, percebe-se que houve significativa explosão dos recursos de computação, seja em termos tecnológicos hoje conhecidos como infra-estrutura, como em termos de software. A evolução da infra-estrutura pode facilmente ser observada, pelo avanço da tecnologia nos computadores pessoais e tudo que os cercam, seja em termos de periféricos ou em termos de componentes de redes e comunicação. A elaboração do software também vem recebendo uma intensa dose de aprimoramento, através das inúmeras metodologias que surgem com grande frequência, sempre com o intuito de obter software com qualidade.

Mesmo diante de tantos recursos e inovações, nos deparamos com certos tipos de problemas, como da otimização de geração de energia elétrica, assunto central desta tese, que requer um poder de processamento por vezes considerado inviável para a obtenção de uma solução ótima e que não pode ser resolvido de maneira trivial. Para este tipo de problema, os métodos existentes fornecem uma solução boa ao invés da solução ótima e neste caso, o grande desafio, é a busca por métodos eficientes para obtenção de bons resultados. É comum também, o uso de modelos híbridos que atuam concomitantemente ou em seqüência, interagindo para buscar uma solução considerada satisfatória.

Sob o ponto de vista holístico do assunto, o suprimento de energia elétrica tem uma importância de vulto na economia de um país e afeta não só os Usuários consumidores, como também as empresas e indústrias. Investir em pesquisas neste segmento traz uma motivação muito forte, pois tem como metas, o bem estar da sociedade e o bom andamento da economia do país.

Atualmente, o planejamento do setor elétrico utiliza um modelo baseado em programação dinâmica estocástica dual, desenvolvido já há algum tempo, apresentando algumas dificuldades, apesar das atualizações constantes, resultantes de um processo de desenvolvimento essencialmente técnico e devido ao método da solução adotada que consome tempos de processamento elevados. Estas dificuldades, mesmo considerando-se o avanço tecnológico de hardware e software, apresentam um grau de complexidade elevada e constitui um ponto importante de motivação: a busca de soluções alternativas que resolvam as duas questões citadas, o da usabilidade e do tempo de resposta do modelo. O modelo matemático usado pelo ONS foi reformulado há pouco tempo em função de necessidades técnicas e como consequência, houve um acréscimo indesejável no tempo de resposta, o que compromete mais ainda a sua usabilidade.

O termo usabilidade, segundo Jacob Nielsen³, um dos expoentes no assunto, foi criado na época da segunda guerra mundial, quando a força aérea norte americana teve seu quadro de pilotos aumentado repentinamente com novos integrantes, tendo sido observado um acréscimo significativo de acidentes, devido ao fato de as aeronaves serem muito complicadas para se operar. Desde então, a facilidade de manuseio de um produto pelo ser humano, tem sido motivo de estudos e pesquisas. Na área de TI, a usabilidade refere-se à facilidade através da qual, o Usuário de um software consegue executar suas operações, através da interface gráfica preparada para este fim. Em se tratando de modelos, a usabilidade significa a facilidade em: operar o modelo, tratar erros, rastrear o algoritmo, extrair

³ www.useit.com

dados parciais, entrar e sair com dados de maneira facilitada e principalmente, apresentar tempos de processamentos adequados a cada tipo de uso.

3. CONCEITOS

Para o perfeito entendimento do assunto tratado nesta tese, é necessário discorrer sobre os conceitos utilizados, agrupando-se em três grandes temas que contribuem para a concepção da pesquisa.

A estratégia utilizada fundamenta-se na seguinte proposição:

Dado um problema X que apresenta um alto custo de processamento, quais processos numéricos fornecem soluções para este tipo de problema e qual a infra-estrutura necessária para que os requisitos estabelecidos sejam atendidos?

Tem-se, portanto, três questões importantes a serem consideradas:

- Quais são os processos numéricos (Software);
- Qual a lista de requisitos (Qualidade de Software);
- Qual a infra-estrutura necessária (Hardware).

O primeiro item requer o estudo dos principais processos numéricos que propõem soluções para o problema. O segundo é um importante item de qualidade que requer um procedimento bastante apurado, na identificação e determinação da lista de requisitos e o terceiro, necessita de uma análise sobre o hardware disponível.

Os conceitos assim agrupados estão descritos em seguida, iniciando com a caracterização do problema.

3.1. Problemas NP-Hard

Há uma classe de problemas que apresentam um custo computacional elevado para obtenção da solução ótima. Entenda-se por custo computacional, o número de passos que o algoritmo executa para encontrar a solução ou, o número de linhas de comando, presentes no código do programa. Em geral, o custo computacional está associado a um algoritmo que efetua um conjunto de procedimentos de forma cíclica. Ao número de

passos consumido pelo algoritmo para encontrar a solução, denomina-se de complexidade.

A definição para problemas NP-Hard ou NP-Difícil é:

“A classe de problemas de decisão que exigem para obtenção da solução, mais do que uma máquina de Turing⁴ não determinística levaria em tempo polinomial, denomina-se problemas NP-Difícil”

Isto significa, que a quantidade de alternativas possíveis para serem analisadas é tão elevada que não pode ser representada por uma função polinomial.

Os problemas que apresentam explosão combinatória, tipicamente pertencem a esta classe de problemas. Muitos são os exemplos explorados na área acadêmica, e até presentes em problemas reais. A maioria delas tem como objetivo, a minimização ou maximização de uma função-objetivo.

Para encontrar a solução ótima de um problema combinatório basta enumerar todas as possibilidades e determinar aquela que atenda ao objetivo. No entanto, esta estratégia nem sempre é possível, pois os problemas interessantes que apresentam alguma utilidade e estão presentes na realidade, possuem um número de combinações elevado, o que inviabiliza esta estratégia, mesmo utilizando o recurso de computadores. A solução para problemas desta natureza exige a utilização de métodos diferenciados.

Alguns exemplos de problemas:

- Problema da Mochila e suas variantes: Dado um conjunto de n objetos de peso w_i e valor agregado p_i e uma mochila de capacidade

⁴ Máquina de Turing é um dispositivo teórico criado por Alan M. Turing em 1936, capaz de processar dados de entrada (símbolos) e tomar decisões (ações) em função do estado corrente, migrando para um novo estado, através de uma função de transição. Quando esta função de transição leva a mais de um estado distinto, então a máquina de Turing é não determinística.

de carga W , deseja-se obter o conjunto de objetos, cuja somatória de valores agregados $\sum p_i$ seja máxima, sujeito a $\sum w_i < W$.

- **Caixeiro viajante:** Dada uma relação de n cidades C_i , deseja-se saber de que forma o caixeiro deverá visitar (uma só vez) cada uma das cidades, de forma que a somatória das distâncias percorrida seja mínima. São conhecidas as distâncias entre as cidades.
- **Fluxo máximo (mínimo) em rede:** Dada uma rede composta por nós e ramos, com um nó de origem e um nó de destino, deseja-se determinar qual é o fluxo máximo possível que passa pela rede, sendo conhecidos a rede e os valores dos fluxos em cada ramo.
- **Caminho mínimo:** Dada uma rede composta por nós e ramos, com um nó de origem e um nó de destino, deseja-se determinar o caminho de menor custo, sendo conhecidos os custos de cada ramo.
- **Roteirização de Veículos:** Dado um conjunto de clientes que possuem mercadorias a receber, a equipe de logística tem que decidir qual é a quantidade de carga a ser colocada em cada veículo e a lista de clientes a ser atendido. Além disso, há a necessidade de otimizar as rotas dos veículos e, em alguns casos, considerar a eventual necessidade de novo carregamento. Como exemplo, citam-se: distribuição de bebidas, produtos, entrega de documentos, coleta de lixo etc.
- **Otimização de Cortes:** Uma fábrica de roupas recebe tecidos em rolos e necessita efetuar os cortes das peças a serem costuradas. O tecido pode ser cortado de diversas maneiras, havendo sempre um desperdício de material, resultante dos pedaços não aproveitáveis. O objetivo é descobrir a melhor maneira de efetuar os cortes do tecido, de modo que o desperdício seja minimizado. Este é um problema de corte em duas dimensões que se aplica a diversos materiais, tais como pedras, vidro, metal, madeira etc.

- Otimização de empacotamento: Os problemas decorrentes de empacotamento podem ser vistos de forma análoga ao problema de corte. Neste caso, o objetivo é dispor da melhor maneira, um conjunto de objetos de modo que as sobras de espaço sejam minimizadas e, portanto, um número maior de objetos seja guardado. O exemplo mais comum deste caso é o trabalho de armazenamento de objetos em uma mudança, para minimizar o número de viagens.
- Alocação de funcionários: Dado um conjunto de atividades a serem realizadas e um conjunto de funcionários, deseja-se saber a melhor maneira de alocar seus funcionários nas atividades, de forma que todas as tarefas sejam cumpridas e os custos com a mão de obra sejam minimizados. Pode-se inserir no contexto, restrições trabalhistas e restrições operacionais da empresa que influem na forma como a alocação pode ser elaborada.
- Seqüenciamento de tarefas e suas variantes: Dada uma lista de tarefas de duração p_i com penalidades por atraso t_i e penalidades por adiantamento e_i , deseja-se estabelecer a ordem de execução destas tarefas em uma única máquina, de modo a minimizar a somatória de penalidades por atraso e por adiantamento. A data de entrega é conhecida e única.
- Planejamento Energético: Dado um conjunto de hidroelétricas e térmicas, produtoras de energia, deseja-se saber como e quanto gerar em cada unidade, de modo a minimizar o custo de produção, evitando o déficit futuro. São conhecidos os custos individuais das usinas térmicas, parâmetros das usinas hidroelétricas, o custo do déficit e dados de transmissão. O recurso hídrico é um parâmetro estocástico.

O último item dos exemplos acima citados constitui o foco principal, objeto de estudos desta tese. As soluções para os diversos problemas também são

variadas, sendo descritas em seguida as propostas que serão utilizadas no desenvolvimento deste trabalho.

3.2. Processos numéricos (Software)

A resolução de um problema de otimização compõe-se de duas etapas: a primeira consiste em formular o problema através de um modelo e, em uma segunda etapa, um algoritmo deve ser implementado para processar o modelo. Em geral, a solução de um problema de otimização não se resolve em tempo razoável utilizando-se algoritmos exatos⁵, devido ao grande número de combinações. Nestes casos, uma solução aproximada é considerada aceitável.

Os Métodos Heurísticos propiciam a construção de algoritmos capazes de encontrar soluções consideradas boas, mas não garantem encontrar a solução ótima de um problema. São muito dependentes das características do problema e seu desempenho na busca de uma solução é maior, na medida em que o espaço de soluções, possa ser restringido em função do conhecimento do problema.

Decorrentes do avanço das pesquisas para solução de problemas *NP-Hard*, diversos procedimentos heurísticos foram criados, constituindo as *meta-heurísticas*. Citam-se as mais conhecidas:

Algoritmos genéticos (AG), Scatter search, Meméticos, Ant system, Simulated Annealing (SA), Busca Tabu (BS), Grasp etc.

O problema de seqüenciamento de tarefas em máquina simples tem sido bastante explorado, como pesquisa de problemas NP-Difícil, em função do conceito *JIT – Just in Time* onde a entrega antecipada ou tardia, são desencorajadas por agregarem penalidades significativas. Consiste em um conjunto de n tarefas com tempos de processamentos definidos e uma data de entrega comum e que deve ser processado em uma única máquina. Se

⁵ Algoritmos exatos são aqueles que tem como objetivo a busca da solução exata ou solução ótima.

uma tarefa termina antes de sua data de entrega, caracteriza um adiantamento medido em unidades de tempo. Por outro lado, se a tarefa termina depois de sua data de entrega, caracteriza um atraso, igualmente medido em unidades de tempo de execução. Para cada tarefa é estipulada uma penalidade por unidade de tempo de atraso e uma penalidade por unidade de tempo de adiantamento. Um importante trabalho foi elaborado por (BISKUP 2001), acerca do seqüenciamento de máquinas simples com data de entrega comum, simulando 280 casos (com 10, 20, 50, 100, 200, 500 e 1000 tarefas), utilizando duas novas heurísticas para determinação dos limites superior e inferior da função objetivo. Este conjunto de casos com os limites calculados tem sido muito utilizados como *benchmark* em novas pesquisas sobre seqüenciamento de máquinas simples.

O mesmo problema de seqüenciamento de máquinas simples com data de entrega comum também foi pesquisado, desta vez utilizando como algoritmo a busca *Tabu* em (JAMES 1997). O trabalho de *Biskup*, citado no parágrafo anterior, refere-se a um seqüenciamento irrestrito da data de entrega, ou seja, não existe nenhuma restrição quanto à data de entrega. No entanto, James mostra também, como aplicar a busca *Tabu* em seqüenciamento de tarefas, com data de entrega restringida.

Em recente revisão da bibliografia, encontraram-se muitos trabalhos utilizando combinações de *meta-heurísticas*. (HUANG 2008) descreve uma experiência com uma estratégia híbrida, combinando a metaheurística *Ant Colony* com a *Busca Tabu*, para resolver o problema clássico de sequenciamento de máquinas. Segundo as colocações dos autores, é indiscutível o bom desempenho de métodos híbridos, em relação a métodos heurísticos simples. Em outro trabalho sobre o mesmo assunto, (AIEX 2003) explora a metaheurística *GRASP – Greedy Randomize Adaptive Search Procedure*, valendo-se de procedimentos paralelos e independentes que colaboram para a solução do problema. *GRASP* é um processo iterativo onde em cada iteração há duas fases distintas, a construção de uma solução possível e a busca local por soluções melhores. O melhor resultado dentre

as soluções obtidas em cada iteração é considerado a solução do problema. Neste trabalho, os autores propõem uma estratégia melhorada do *GRASP* tradicional, implementando uma intensificação na busca local, o que resultou em um desempenho melhor.

Um dos precursores em metaheurísticas (GLOVER 1995) utiliza o algoritmo *Genético* com a *Busca Tabu*, para resolver um problema de otimização. (PIRLOT 1996) descreve em seu artigo, um tutorial sobre a utilização de SA – *Simulated Annealing*, BT – *Busca Tabu* e AG – *Algoritmo Genético* com exemplos de aplicação, tecendo ao final, uma comparação entre os métodos, além de citações sobre técnicas de utilização de modelos híbridos BS e SA, BS e AG. (HINO, 2005) utilizou a *Busca Tabu*, juntamente com o algoritmo *Genético*, explorando as propriedades da solução ótima deste tipo de problema. Efetua também, uma análise de uma estratégia híbrida para melhorar o desempenho do método utilizado.

A utilização de meta-heurísticas pode ser combinada com a pesquisa operacional *OR/MS* através da programação linear. Um exemplo de utilização combinada destes métodos para o cálculo da função objetivo pode ser visto em (TAGHEM 1995). O problema de arranjo de capas de livros em uma mesma chapa de impressão é resolvido por uma programação linear mista e a busca combinatória é efetuada por qualquer um dos métodos AG, SA ou BT.

Há ainda, outras metodologias para solução de problemas de otimização, principalmente na área de transportes. A *Programação Linear* utiliza modelos matemáticos para formular um problema em termos de variáveis contínuas e um conjunto de restrições lineares sobre essas variáveis, representadas por equações ou inequações. A função-objetivo é representada pela fórmula de cálculo do custo que deve ser minimizado ou outro qualquer. Através de algoritmos apropriados (*Simplex*), o problema é resolvido de forma eficiente em tempo hábil. No caso real, freqüentemente as variáveis necessitam assumir valores inteiros e não contínuos. Neste caso, quando se impõe a restrição de que as variáveis assumam valores

inteiros (por ex. número de objetos ou pessoas) o problema pode ficar muito mais complicado, mas para isto, utiliza-se em conjunto com o *Simplex*, o método de *Branch and Bound* para determinação da solução inteira.

Aproveitando a ampla difusão da planilha Excel no mercado, a Microsoft encomendou uma função denominada de *Solver* que pode ser aplicada, na grande maioria dos problemas habituais, segundo seus criadores (FYLSTRA 1998). De acordo com *Fylstra*, os problemas freqüentes se situam até aproximadamente 200 variáveis de decisão. Um exemplo de utilização do otimizador Excel é citado em (SCHUSTER 1998), onde os autores utilizaram uma solução via programação linear para a *Welch's*, uma companhia de processamento de sucos de uva. O *MRP*⁶ – *Material Requirement Planning* fornecia originariamente um modelo com 8000 variáveis, impraticáveis para uso em planilhas, cuja versão em uso admitia somente 4000 variáveis e 2000 equações. Com o objetivo de diminuir a dimensão do problema, agregou-se a demanda por grupos de produtos, ao invés de produtos (esta é uma estratégia muito comum em otimização: transformar o problema em problemas menores ou *divide and conquer*), adotando o conceito de “famílias” de produtos, diminuindo para 324 variáveis de decisão com 361 equações.

O *Solver*, no entanto, não é o precursor na implementação de otimizadores de planilhas, cabendo a (SAVAGE 1985) com o algoritmo “*What's best*” em 1985. Este último é comercializado com suporte técnico pela “*Lindo System Inc*”⁷. O Microsoft Excel *Solver* usa o código de otimização não linear de gradiente reduzido genérico (*GRG2*), desenvolvido por *Leon Lasdon* da Universidade do Texas em Austin, e *Allan Waren*, da Universidade Estadual de Cleveland. A condição de parada do algoritmo

⁶ MRP é um sistema de inventário que tem como objetivo colocar à disposição da produção, o material certo no ponto certo e no momento certo, minimizando os investimentos em inventário. É uma atividade de planejamento que está intimamente ligada à linha de produção.

⁷ www.lindo.com

ocorre quando ele encontra a solução ótima; quando a função-objetivo não converge; quando não existe solução; ou quando o tempo máximo ou o número de interações é excedido.

Para a utilização de otimizadores, é necessário criar antes, um modelo representativo do problema. Um modelo em *OR/MS* segundo (PIDD 1999) é uma representação externa e explícita, de parte de uma realidade utilizada para entender, alterar ou gerenciar a mesma. Um modelo que represente a totalidade, se existir um, será tão caro complexo e de difícil manipulação, de onde se conclui que, a representação através da modelagem deve ser simples e parcial, direcionada para um objetivo específico. Um modelo simples não é necessariamente um modelo pequeno. De acordo com *Pidd*, sob o ponto de vista matemático, ser simples significa ser elegante e saber simplificar, exige grande conhecimento do problema e das ferramentas em uso. Os problemas de otimização, tipicamente exigem simplificações para obter soluções viáveis. *Pidd* cita ainda que “*um problema bem equacionado já é meio caminho andado*”

Muitos problemas possuem características de multi-estágios, ou seja, aqueles em que a situação de cada estágio depende do procedimento adotado no estágio anterior. Neste caso, o problema pode ser resolvido através da *PD - Programação Dinâmica*.

Richard Bellman criou o método da programação dinâmica para resolver os problemas de otimização, envolvendo processos decisórios seqüenciais. Baseia-se no princípio de optimalidade de *Bellman*:

Dada uma trajetória ótima T_{AC} de A a C, uma porção T_{BC} desta trajetória que parte de um ponto intermediário B até o ponto C é uma trajetória ótima de B a C.

Figura 3.2-1 Princípio de Optimalidade de Bellman

A prova deste princípio é obtida através de:

Dada a trajetória ótima $T_{AC} = T_{AB} T_{BC}$ de A a C, de acordo com o princípio de Bellman, T_{BC} é uma trajetória ótima de B a C. Suponha que exista uma outra trajetória ótima t_{BC} de B para C. Então, $T_{AB} t_{BC}$ deve ser uma trajetória melhor que T_{AC} . Isto contraria a hipótese de que T_{AC} é a trajetória ótima.

Figura 3.2-2 Prova

Os problemas combinatórios normalmente envolvem processos recursivos, onde um procedimento de cálculo é efetuado diversas vezes, na busca da solução ótima. A *PD* é um processo de otimização desta recursão, transformando-a em iteração, onde o problema é dividido em subproblemas e os resultados dos subproblemas são armazenados em memória, evitando efetuar um mesmo procedimento de cálculo, se este já foi executado anteriormente.

Uma característica importante da PD é a possibilidade de resolver problemas com representação estocástica, como é o caso da ocorrência de água na natureza. No entanto, a representação estocástica traz como consequência, um alto custo de processamento, o que inviabiliza a aplicação do método em problemas do tipo NP-Difícil. A solução para este caso é a simplificação do problema de grande porte, o que significa perda de qualidade do modelo na representação do caso real, transformando-o em um problema menor, de modo que o método possa ser aplicado.

Este fato representa um grande dilema a respeito do uso da PD, ou seja, para aplicar o método em problemas de alta complexidade, é necessário simplificar e como toda simplificação, há uma perda de precisão na representação do problema. O modelo NEWAVE utilizado atualmente pelo setor elétrico utiliza a PD, simplificando o sistema de geração brasileiro, agrupando as hidroelétricas em reservatórios equivalentes.

Em outra vertente, a Inteligência Artificial tem direcionado esforços no sentido de construir sistemas com características cada vez mais próximas da realidade em que estamos inseridos. Assim, a área de *SMA – Sistema Multi-Agentes*, baseia-se em conceitos de Sociologia que concebe sistemas, com características de sociedades ou grupos, onde o objetivo principal é o estudo do comportamento coletivo nestas sociedades. Alguns exemplos de sociedades: um formigueiro, uma colméia de abelhas, uma escola, um time de futebol, uma empresa, uma organização web etc. Nos exemplos citados, denotam-se características comuns que são objetos de análise em *SMA* tais como: a interação entre os seres, envolvendo uma linguagem e um protocolo de comunicação, o comportamento ativo e reativo de cada ser, a dinamicidade do ambiente onde a sociedade existe e a forma como elas são organizadas.

Os *SMA*s constituem-se de artefatos de software (Agentes) que atuam como se fossem elementos de software individuais. O conjunto de agentes, redes e base de conhecimentos é conhecido também, como sociedade de objetos. Tendo em vista que tais objetos têm a possibilidade de serem processados

em ambientes de processamentos diferentes, comunicando-se através de uma rede, pode-se considerar como um sistema distribuído utilizando objetos distribuídos. Desta forma, o conjunto de objetos distribuídos emula o comportamento da sociedade de objetos, o que corresponde a uma automação das atividades desta sociedade.

Dentro de uma sociedade, os agentes têm como característica interagir com outros agentes para atingir seus objetivos, utilizando um protocolo de comunicação como o da (FIPA 2005), conforme pode ser visto na figura 3.2-3 “*Protocolo de comunicação FIPA*”.

Neste diagrama, a iteração começa com o *Initiator* emitindo m chamadas com propostas para a ação *cpf-1*. Dos n participantes que responderam à proposta, k responderam estar capacitados a executar a ação proposta e $j=n-k$ participantes rejeitaram a proposta.

Figura 3.2-3 Protocolo de Comunicação FIPA

O *Initiator* pode decidir ser o final da interação, aceitando p propostas e rejeitando as restantes $k-p$. Alternativamente, poderá selecionar l propostas e solicitar uma nova revisão *cpf2* para os participantes, rejeitando as $k-l$ restantes. A intenção é obter respostas melhores, modificando a proposta anterior. O processo termina quando o *Initiator* rejeita todas as propostas dos proponentes ou, quando aceita p propostas rejeitando as demais ou, por fim, quando nenhum proponente respondeu à proposta efetuada.

Nesta interação, pode ocorrer uma colaboração, onde um agente se presta a operar em conjunto com outro agente, para atingir uma meta comum.

Outra particularidade dos agentes é a capacidade de adaptação de seus comportamentos decorrentes das mudanças que ocorrem no ambiente. O comportamento de um agente é regido por um mecanismo de “conduta social”, o que corresponde a utilizar informações de outros agentes para benefício próprio. Diz-se que um agente A depende de outro B , quando B de algum modo auxilia A em atingir seus objetivos. Esta troca de favores entre agentes permite levar dois ou mais agentes a se unirem (coalizão), em busca de um objetivo comum, obtendo com isto resultados até superiores ao que obteria, se todos se engajassem na luta isoladamente.

Os agentes são construídos de forma a assumirem um determinado papel dentro da sociedade. Cada papel é constituído de um conjunto de procedimentos, e uma base de conhecimentos que pode ser utilizada em seus *raciocínios* para tomar decisões. (SICHTMAN 2001) adota em seu trabalho alguns princípios importantes que facilitam a utilização de modelos SMA, considerando que o comportamento dos Agentes, obedece a alguns princípios que são tidos como verdadeiros a respeito de seu comportamento:

- *Princípio da não benevolência* – os agentes não têm como atitude assumir benevolência em seus atos, ou seja, sua decisão de “ajuda” a outro agente é baseada única e exclusivamente em seus princípios e objetivos pré-definidos, decidindo de forma destituída de qualquer tipo de sentimento;

- *Princípio da Sinceridade* – os agentes não têm atitudes deliberadas de fornecer informações errôneas sobre si mesmo. Transmite aos outros agentes, somente informações em que acredita serem verdadeiras;
- *Princípio do auto conhecimento* – possuem um conhecimento completo sobre si mesmo, acerca de seus objetivos, suas capacidades etc.;
- *Princípio da consistência* – os agentes não possuem informações dúbias sobre outros agentes. Quando percebem erros, adotam o procedimento de revisar seus conceitos e dados, de modo a buscar uma situação coerente.

A ação preponderante entre os agentes é a sua convivência entre eles, como se estivessem em uma sociedade, e juntos, se propõem a resolver problemas considerados difíceis. Como em toda sociedade, existem *indivíduos* com diversos perfis, qualificações, funções e responsabilidades e neste caso, de forma análoga e dependendo do problema a ser resolvido, criam-se diversos papéis e as correspondentes funcionalidades para atuar na sociedade.

Uma particularização do SMA é o denominado *EPS - Eco Problem Solving*, que se caracteriza como uma maneira diferente de solução de problemas, bastante parecido com a técnica *ALife*⁸, muito pesquisado em *Swarm Intelligence*⁹ ou sistemas auto organizados. Este método se aplica em um sistema dinâmico e utiliza agentes simples, isto é, agentes que não têm capacidade para elaborar planos, e se comportam de acordo com sua programação ao qual se denomina *eco-comportamento*, que nada mais é do

⁸ *ALife* – Artificial Life, enfatiza o desenvolvimento da inteligência e explora aspectos sociais ou de grupos. ² *Swarm* – Entende-se por *swarm* um conjunto de agentes móveis capazes de se comunicar entre si, e coletivamente buscar uma solução para um problema distribuído. (abelhas, formigas)

que uma combinação de *tropismos*, ou seja, constituídos de um conjunto de reações adotadas, diante de mudanças que ocorrem no meio.

Ao invés de considerar o problema como um todo, utiliza-se o conceito de transformar o problema em porções menores (uma vez mais: *divide and conquer*), que são resolvidos pelos agentes reativos independentes, cada um deles possuindo objetivos e comportamentos específicos.

A solução, se existir, obtém-se quando todos os agentes se encontram em situação de satisfação individual (objetivo atingido), uma vez que o estado final do sistema ou, satisfação global do sistema, corresponde a um estado de satisfação de todos os agentes, ou seja, a solução do problema.

Este método segundo (DROGOUL 1993) fornece soluções para problemas de explosão combinatória, como descreve em seu trabalho acerca do problema n-puzzle, cuja solução pode ser conseguida por esta técnica. N-puzzle é um jogo composto de $n \times n$ botões quadrados deslizantes sobre um tabuleiro, contendo apenas uma casa vazia para que os botões possam se movimentar nas quatro direções, para cima, para baixo, para a esquerda e para a direita. Cada botão possui um desenho, que pode ser um número ou uma letra do alfabeto. A solução do problema é efetuar uma série de movimentos, para colocar os botões em ordem numérica ou alfabética no menor tempo. Neste problema, a “casa vazia” pode estar em um dos quatro cantos, nas laterais ou no interior do tabuleiro. Para cada uma das situações, o número de possibilidades é de 2, 3 e 4 respectivamente, o que transforma o problema em um processo de busca em árvore, com mais de um ramo em cada nó, resultando em uma explosão combinatória. Cada elemento do puzzle constitui um agente reativo, sem planos, mas com comportamentos que se modificam em função do estado do conjunto. Assim, cada agente possui procedimentos, para atacar outro agente ou se defender, sempre procurando atingir seu objetivo. Quando todos os agentes encontram-se satisfeitos, a solução é obtida. Os experimentos elaborados neste trabalho resultaram para um puzzle de 30 por 30, ou seja, 899 elementos, um tempo de 1179.58 segundos, com 23009 movimentos fornecendo uma média de

25.6 movimentos por elemento. O autor implementou também um programa em Java, que para uma simulação de um puzzle de 15 elementos, obteve a solução em alguns segundos, incluindo o tempo gasto na parte gráfica de movimentação das peças, na janela da aplicação.

3.3. Lista de Requisitos (Qualidade de Software)

O software apesar de ser abstrato, pode ser considerado como um objeto “*manufaturado*”. A sua qualidade pode ser constatada, na medida em que executa de forma eficiente e eficaz, as funcionalidades especificadas segundo a ótica do Usuário. No entanto, esta lista pode ser extensa e desta forma, uma das dificuldades é garantir que todos os itens desta lista, estejam funcionando corretamente. O processo tradicional de desenvolvimento de software preconiza a utilização de testes de diversas naturezas e estratégias, com o intuito de inspecionar se o software atende aos requisitos estabelecidos. A fase de testes consome normalmente, 40% do tempo previsto para o desenvolvimento de um projeto e um erro descoberto em um sistema após implantação, provoca um acréscimo de 60%, nos custos do projeto segundo (PRESSMAN 2006). Nenhum programador ou analista, por mais experiente que seja, está imune a falhas de codificação. Em certos casos, o procedimento de testes se torna inviável, como nos casos onde a lista de funcionalidades é extensa, alcançando valores proibitivos. A garantia de que um software execute corretamente todas as funcionalidades segundo as quais foi projetado, é inversamente proporcional a sua dimensão. As empresas que utilizam práticas modernas de *TI* tais como o *CMMI* nível 5, possuem métricas que permitem estabelecer o nível de erros que seu produto irá apresentar ao final do desenvolvimento, baseado no histórico das ocorrências registradas ao longo do tempo. Desta forma, a qualidade de um desenvolvimento só fica estabelecida, na medida em que se apresentam valores sobre a incidência de erros, o que na verdade representa o seu nível de qualidade. Mesmo as empresas que possuem o *CMMI* nível 5 e se propõem a desenvolver grandes projetos utilizando diversas equipes, são surpreendidos com algumas dificuldades ao fabricar o software. Isto

acontece, porque as lideranças de equipes pensam de forma diferente e nem sempre, o critério ou procedimento adotado por um líder de equipe é igual ao de outro líder, o que acarreta diferenças que não comprometem, mas são indesejáveis.

Para os casos onde não há possibilidade de inspecionar toda a gama de funcionalidades de um software, é possível garantir que partes dele funcionem adequadamente, atuando no processo de fabricação. A estratégia “*divide and conquer*” se aplica quando se trata de componentização. Um componente de software é uma peça de um todo, que executa uma determinada função e tem utilidade genérica. É importante investir em sua produção, criando meios de inspecionar tais componentes, de modo que a sua qualidade seja próxima da ideal, ou seja, baixo índice de falhas. A evolução de um componente é gradativa e ocorre através de um processo de melhoria contínua, conforme pressupõe a metodologia *CMMI*. (BANKER 1991) argumenta que o nível de reuso do código é uma métrica da qualidade no processo de desenvolvimento. Ao investir em componentização, o grau de re-usabilidade e o nível de qualidade de partes do software é incrementado, o que significa, estar investindo no processo de fabricação do software, com o objetivo de produzir um artefato de melhor qualidade.

O uso de padrões, no processo de desenvolvimento de software, também proporciona aspectos positivos em diversos sentidos. O processo de fabricação do software ganha desempenho ao desenvolver uma aplicação, pois não se perde tempo, decidindo ou analisando a forma de implementação das rotinas. Na medida em que tais padrões estejam bem definidos, maior é a desenvoltura na elaboração do software. Quando se fala em padrões, surge também a necessidade de eleger ferramentas e os meios de comunicação entre os envolvidos com o processo de desenvolvimento. Normas de documentação de sistemas, de *follow up*, análise estatísticas de suporte, falhas observadas etc, são imprescindíveis para monitorar o processo de fabricação do software. Padrões de codificação também levam a um ganho de desempenho na atividade de programação e com certeza,

proporcionam economia de muitas horas de manutenção ou de alterações. Uma das principais vantagens decorrentes da adoção de padrões é a transparência do processo de desenvolvimento, tornando a atividade independente do executante. É necessário, no entanto, adotar uma forte administração, para que seja implantado um desenvolvimento baseado em normas e padrões. Não havendo controle adequado, o resultado obtido pode ser ineficiente.

A área de *TI* vem se tornando um dos importantes pilares das organizações que querem se manter competitivas no mercado atual. É inimaginável vislumbrar uma empresa, sem uma forte área de sistemas de informações, manipulando dados operacionais e provendo informações gerenciais ao corpo executivo, através dos *DSS – Decision Support System*. Para melhorar o processo de análise de riscos e tomada de decisão, é necessário um processo estruturado para gerenciar e controlar as iniciativas de *TI* nas empresas, para garantir o retorno de investimentos e adição de melhorias nos processos empresariais. A este novo movimento denomina-se Governança em *TI*, ou "*IT Governance*". O fato de a Informática ser necessária, com seus equipamentos, redes e software é clara, e o que se discute no momento, está focado na maneira como cada empresa gerencia seu aparato tecnológico. A partir dessa necessidade, modelos como o *ITIL – Information Technology Infrastructure Library* vem se firmando no mercado. Não são muitas as empresas que o adotam, mas a preocupação existe e a diversidade de metodologias é notória. O *ITIL* é uma coleção das melhores práticas e processos de prestação de serviços na área de *TI*. Foi criada na década de 80, pela *CCTA – Central Computer and Telecommunications Agency*, uma agência do Governo Britânico. A norma cobre diversas áreas, mas tem seu foco na qualidade do gerenciamento de serviços de *TI*, que quando aplicadas de forma homogênea, resultam em maior grau de eficiência na gestão de *TI* nas organizações.

Outro modelo, o *CobiT* é um guia para a gestão de *TI* sugerido pelo *ISACF (Information Systems Audit and Control Foundation, www.isaca.org)*. As

práticas de gestão do *CobiT* são recomendadas pelos peritos em gestão de *TI* que ajudam a otimizar os investimentos da área e fornecem métricas para avaliação dos resultados. O *CobiT* independe das plataformas de *TI* adotadas nas empresas e é orientado ao negócio. Fornece informações detalhadas para gerenciar processos, baseados em objetivos de negócios.

Ainda na linha de metodologias de gestão cita-se o *Rational Unified Process – RUP*, um processo de engenharia de software. Tem como principal característica delegar tarefas e responsabilidades, dentro de uma equipe ou empresa de desenvolvimento. Seu objetivo principal é garantir a produção de software dentro do prazo e custo planejado, e com a qualidade desejada pelo cliente. O ciclo de desenvolvimento do *RUP* é dividido em 4 fases: iniciação, elaboração, construção e transição.

O *CMMI – Capability Maturity Model Integration* é um “framework”¹⁰ ou arcabouço, para avaliação da maturidade de processos de desenvolvimento de software. O Instituto de Engenharia de Software (*SEI*) da universidade *Carnegie Mellon*, Estados Unidos, desde 1986 desenvolve e aperfeiçoa esse quadro de avaliação, cuja primeira versão era chamado simplesmente *CMM – Capability Maturity Model for Software*. O objetivo desse modelo é fornecer às organizações, um meio de medir a maturidade do processo de desenvolvimento de software, além de definir programas de melhoria desses processos. Por maturidade, pode-se entender a capacidade de replicar em projetos futuros, o sucesso obtido em projetos passados.

Todas estas metodologias proporcionam fortemente o controle e organização das atividades de *TI* e, portanto, a qualidade surge como um dos benefícios diretos. Na medida em que as organizações se estruturam e

¹⁰ No âmbito de desenvolvimento do software, um framework ou arcabouço é uma estrutura de suporte definida em que um outro projeto de software pode ser organizado e desenvolvido. Um framework pode incluir programas de suporte, bibliotecas de código, linguagens de script e outros softwares para ajudar a desenvolver e juntar diferentes componentes de um projeto de software. Fonte Wikipédia <http://pt.wikipedia.org/wiki/Framework>

adotam métodos e padrões, aumenta gradativamente a probabilidade de reuso dos componentes de software.

3.3.1. Estratégia de Implementação da Qualidade em Software

Software é um produto abstrato contendo conhecimento agregado, sendo utilizado para estabelecer o fluxo de operações de algum procedimento. Em uma visão simplificada, para que o software tenha qualidade, parece suficiente investir no processo de identificar e agregar o conhecimento. No entanto, as duas ações são complexas. A atividade de identificar os conhecimentos é complexa, devido a falhas de comunicação entre o detentor do conhecimento (Usuário) e o projetista de sistemas. Os principais problemas referem-se a:

- O projetista não domina o conhecimento do Usuário com a necessária abrangência (projeto limitado);
- O Usuário não interpreta corretamente a linguagem técnica do projetista (falha de especificação);
- O Usuário nem sempre consegue informar tudo que é necessário para que o projetista elabore o software (projeto incompleto).

Existem metodologias que tem por objetivo minimizar os erros de especificação de requisitos do software, garantindo um bom nível de qualidade. Por outro lado, há muitas formas de agregar o conhecimento especificado, de forma a garantir a funcionalidade especificada. As práticas de *TI* vigentes visam de maneira geral, atuar no processo de agregação do conhecimento, que acontece ao longo do desenvolvimento, também conhecido como processo de software. Desta forma, como estratégia de implantação da qualidade, sugere-se a adoção de uma administração de *TI* eficiente *CMMI*, *PMI*, *COBIT*, *RUP*, etc. conforme citadas anteriormente que possibilitem, além de uma forte organização, desenvolver as atividades com métodos e padrões bem delineados.

A utilização de um padrão de linguagem do tipo *UML – Unified Modeling Language*, proporciona uma comunicação eficiente evitando erros e conseqüentemente, aumentando o nível de qualidade dos projetos.

3.3.2. Aspectos de qualidade em algoritmos

Algoritmo é uma descrição seqüencial de procedimentos lógicos, apresentando um número finito de etapas ou passos que devem ser executados, para completar uma tarefa. A quantidade de passos que um algoritmo consome, para resolver um determinado problema, é denominada de complexidade.

Um algoritmo é uma abstração da realidade onde são considerados somente os aspectos relevantes do problema que está sendo modelado. Desta forma, eliminam-se do processo, todos os fatores que não tem importância para o estudo.

Sob o aspecto computacional, o algoritmo representa não a solução, mas a seqüência de procedimentos que devem ser executados, para que uma determinada situação seja resolvida. Um algoritmo que representa um fenômeno na forma numérica pode ser rotulado de modelo matemático e tem por objetivo, produzir números que constituem a solução do problema. Nem sempre os procedimentos são numéricos, mas sim lógicos, ou seja, constituem-se de processos decisórios que dependem de cálculos ou parâmetros adotados. No caso geral, o algoritmo engloba cálculos e processos decisórios.

O algoritmo serve como roteiro para que o Engenheiro de Sistemas elabore o software correspondente, segundo os métodos de desenvolvimento preconizados pela Engenharia de Software. A comunicação entre o Usuário e o projetista de software nem sempre é uma tarefa simples e da mesma forma, o perfeito entendimento do problema pelo projetista, em certos casos não é uma tarefa fácil. Para problemas complexos, há a necessidade de uma atuação conjunta do especialista com o projetista, pois este último

desconhece o problema e não tem como avaliar se os resultados parciais ou totais obtidos são aceitáveis ou, representam valores fora da realidade.

O algoritmo implantado é um artefato de software e como tal, para a sua elaboração, valem em princípio, todos os fundamentos descritos anteriormente quanto à sua qualidade. No entanto, é necessária uma atenção diferenciada de como implementar a qualidade.

Antes mesmo de passar para a tarefa de codificação do algoritmo, há a necessidade de se preocupar com a qualidade e conveniência do algoritmo. Se este apresenta um procedimento de cálculo inadequado, o software correspondente será igualmente inadequado. É preciso avaliar também o tipo de problema que está sendo considerado, pois nem todo algoritmo tem viabilidade computacional para sua implantação ou, apresenta uma dificuldade operacional, por ex. tempo de processamento considerado inadequado. Por outro lado, deve se atentar para o balanceamento equilibrado do conjunto Requisitos de Software, Processos Numéricos e Infra-Estrutura requerida. Isto quer dizer que a escolha do algoritmo (Software), deve ser compatível com os recursos de hardware necessários (Hardware), de modo que as necessidades do Usuário (Requisitos) sejam atendidas, apresentando uma usabilidade adequada para operar o software.

Compatibilizar a infra-estrutura (Hardware) significa verificar ou estabelecer em qual configuração de equipamento, o processo numérico (Software) deve ser implementado, ou seja, que tipo de processador será utilizado, em que ambiente operacional será processado, se haverá ou não processamento paralelo ou distribuído, se fará uso banco de dados, etc.

Compatibilizar Processo Numérico (Software) significa analisar alternativas de métodos numéricos que resolvam o problema, atendendo à lista de requisitos estabelecida, envolvendo também a determinação da plataforma de trabalho, definição da linguagem de programação, bibliotecas, etc. que produzam resultados satisfatórios, em consonância com o hardware adotado.

Sob o ponto de vista de qualidade de projetos de software científicos, o processo numérico constitui a parte principal da especificação de requisitos e, portanto, a conformidade é um item de qualidade obrigatório. A descrição em geral é efetuada através de pseudocódigos e a validação, se necessária, pode ser efetuada através de prototipação. Uma vez concebido o algoritmo, o mesmo deve ser codificado em linguagem de programação, retratando fielmente os passos nele constantes. Em geral, os algoritmos são complexos e para a sua elaboração, algumas providências são bem vindas, visando obter o máximo de qualidade no seu processo de desenvolvimento:

- a) Os algoritmos, em geral, apresentam quantidades de passos (ou complexidade) elevadas e por isso, o primeiro ponto que influencia na qualidade do software, é a escolha da linguagem de programação que proporcione processamentos rápidos. Há diversas opções de escolha de linguagem de programação para processamentos ostensivos: *Fortran*, *C/C++* e *Java*. A linguagem *Assembler* pode ser utilizada em rotinas críticas que exigem alto desempenho, mas em geral, esta alternativa não é freqüente. Estas linguagens apresentam alto desempenho, além de muitos recursos de cálculo, mas não apresentam facilidade, na elaboração de janelas para navegação ou apresentação de dados. Neste aspecto, há que se considerar, se a solução do problema em questão exige grande capacidade de processamento. Não sendo necessário, este requisito pode ser relaxado, utilizando talvez uma alternativa mais amigável para a linguagem de programação.
- b) A inspeção do software, no caso de este ser um algoritmo, constitui uma facilidade importante no projeto. Para isto, deve-se prever o acionamento paramétrico de inspeção, de partes do algoritmo ou do todo, proporcionando um recurso de grande utilidade para o Usuário. Com isto, torna-se possível o rastreamento direcionado do algoritmo, para efetuar uma simples verificação de conformidade ou, para analisar o comportamento do software, quando os resultados

apresentarem inconsistências. Se não existe a facilidade de inspeção do algoritmo, consome-se muito tempo em análise de resultados e dependendo da sutileza do erro, isto pode se tornar um problema crítico. A inspeção do software é um item de qualidade denominado de auditabilidade, o que significa, a possibilidade de aferir o software em relação às funcionalidades estabelecidas.

- c) Como todo software, um algoritmo também possui dados de entrada e dados de saída. A qualidade do software pode ser incrementada através de uma bateria de funções de análise de dados, que possibilite verificar a coerência, suficiência e representatividade dos dados. Não há como esperar bons resultados se os dados de entrada apresentam inconformidades. Portanto, esta etapa do processo é de importância relevante, para garantir a qualidade do software. Uma das formas de garantir em parte a integridade de dados é projetar um banco de dados, que contenha todas as características dos dados de entrada. Toda e qualquer regra imposta estará garantida pelo gerenciador do BD. No entanto, isto compreende somente a integridade em relação às regras impostas, não sendo possível a detecção de uma inconsistência fora destas regras.
- d) A elaboração de funcionalidades para análise dos resultados obtidos tem importância significativa no processo. A visualização dos resultados, na forma gráfica ou na forma de relatórios, com diversos níveis de estratificação, agregam qualidade ao software. Além disso, relatórios estratégicos de verificação são bastante produtivos, indicando inconsistências difíceis de serem diagnosticadas através dos relatórios de saídas usuais.

Para os algoritmos, a questão da exatidão nos cálculos e nos processos é extremamente importante e por isso, deve ser exaustivamente testado tanto sob o aspecto individual como no aspecto integrado. Todo e qualquer componente (parte do software) cuja função é executar uma tarefa, deve ser testado de forma a obter resultados absolutamente corretos e

documentados. Em se tratando de componentes de domínio público, deve-se buscar a implementação de melhor qualidade, em termos de codificação ou em termos de desempenho. Os componentes precisam ser testados no todo e também nas condições de contorno, documentando a atividade. É extremamente importante, a aplicação de testes individuais, avaliando inclusive o aspecto do desempenho. Em problemas de complexidade elevada que exigem grande capacidade de processamento, o desempenho de cada parte do software pode se constituir em um obstáculo, para o funcionamento adequado do algoritmo e desta forma, a rapidez de processamento deve ser analisada com cuidado. A estratégia indicada é garantir que as partes tenham confiabilidade (e desempenho), para que o todo também seja confiável.

3.3.3.QFD – Quality Function Deployment

A especificação de requisitos influi sobremaneira, na qualidade de um software e aumenta mais ainda, na medida em que a especificação retrata com fidelidade, as necessidades do Usuário. É uma tarefa de grande dificuldade e para isto, existem muitos métodos, mais freqüentemente vistos na indústria, que focam este assunto e se aplicam também ao desenvolvimento de software.

Dentre os métodos existentes, o *QFD – Quality Function Deployment* ou Desdobramento da Função de Qualidade, tem tido bastante sucesso no desenvolvimento de produtos e como mostra a pesquisa efetuada, existem muitos trabalhos, inclusive recentes, que utilizam o *QFD* para identificação de requisitos de software.

Este método surgiu no final da década de 60 no Japão, constituindo-se em um dos métodos mais eficientes utilizados na indústria de manufatura, para estabelecer o conjunto de requisitos desejáveis de um determinado produto ou serviço, segundo a ótica do cliente, e constitui uma das ferramentas integrantes do *TQM – Total Quality Management*. Ao longo da década de 70, esta técnica evoluiu tornando-se um instrumento cada vez mais robusto

disseminando-se pelos continentes, mais incisivamente nos *EUA* em décadas seguintes. (AKAO 2003) cita em seu trabalho que o método tem sido aplicado por indústrias líderes no Japão desde 1966. Na medida em que este método teve sua aplicação comprovada fora do território japonês, houve um movimento crescente de disseminação do método, através de inúmeros congressos. O primeiro deles, no Japão, aconteceu em 1983 patrocinado pelo Centro de Produtividade Japonês.

A proposta do *QFD* visa essencialmente, coletar e entender as necessidades ou expectativas do cliente, acerca de um determinado produto, transformando-os em especificações de projeto, que são utilizados durante o processo de fabricação. Como resultado decorrente deste procedimento são obtidos produtos, que apresentam uma qualidade e confiabilidade, que vão ao encontro dos anseios do cliente. O método vai mais além ainda, procurando descobrir outros requisitos não relatados pelos clientes, mas que são inferidos durante o processo, com o objetivo único de surpreendê-lo, recurso muito útil para garantir a sua satisfação.

Há diferenças entre produzir um produto e produzir um software, como relata (MURAKAMI 2005), assim como há diferença em especificar um produto e especificar um software. Um requisito de um determinado produto manufaturado fica muito claro, como por ex. um carro de cor preta. No entanto, quando se trata de um software, o requisito, como por ex. software de qualidade, pode-se tornar subjetivo e complexo.

Outros autores como (RUIZ 2002), apresentam uma visão diferente, do processo de desenvolvimento de software, e sugerem uma estratégia de como garantir a qualidade, baseada nas práticas modernas de modelagem. A estrutura do trabalho destes autores baseia-se nos métodos tradicionais de desenvolvimento de software, juntamente com o uso ostensivo de um modelo de simulação dinâmica do processo de desenvolvimento, com o objetivo de propiciar um embasamento quantitativo e qualitativo, para incremento do processo de desenvolvimento de software e da tomada de decisões, de modo a alcançar um alto valor da capacidade, no processo de

desenvolvimento de software de acordo com o método *CMMI – Capability Maturity Model Integration*.

(LIU 2005) relata em recente artigo, a importância de investir continuamente no processo de software, para continuar subsistindo neste ambiente competitivo de desenvolvimento, onde a inovação tecnológica é extremamente acentuada. Cita a conveniência de uso do *CMM* em diversas situações, o que provoca certa resistência em migrar para o *CMMI*. A grande dificuldade do *CMM*, segundo os autores, reside no fato de haver certa incongruência entre os objetivos do negócio de uma organização e os níveis de maturidade. O *CMM* indica o “*what to do*” deixando para a organização o “*how to do*”. Neste trabalho, os autores criaram um interessante *framework* para auxiliar no processo de mapeamento das regras de negócio e de outros requisitos para o *CMM*, além de propiciar subsídios para a elaboração de planos de ação, que atendam às metas do *CMM* e aos requisitos estabelecidos, utilizando para isto o método *QFD*.

Em outro trabalho recente, o método *QFD* é explorado com o objetivo de buscar a satisfação do cliente, em uma aplicação bancária via Internet. (GONSÁLEZ 2004) descreve uma experiência, onde a voz do consumidor foi obtida, através de uma pesquisa de 4 mil questionários enviados via email, tendo recebido como retorno, cerca de 824 respostas totalizando 20.6%. Os requisitos do Usuário foram sumarizados utilizando o diagrama de afinidade e de relacionamento do *TQM*, elaborando uma lista de 14 itens que na origem, constituíam cerca de 535 itens. Os requisitos foram avaliados na “Casa da Qualidade” conforme preconiza o método *QFD*. Como resultado da aplicação deste método, chegou-se à conclusão de que gerentes de projeto e gerentes de qualidade se beneficiaram dos resultados obtidos, possibilitando efetuar os ajustes, entre os requisitos dos Usuários e os procedimentos internos da organização.

Estabelecer metas em *SQFD – Software Quality Function Deployment*, segundo (LIU 2006), é extremamente importante, pois está diretamente relacionado com a elaboração de produtos de alta qualidade, e com elevada

satisfação do cliente. Normalmente, as metas são estabelecidas de modo subjetivo na prática, o que deixa de ser um método científico. Duas metodologias, para quantificação dos valores dos requisitos, foram propostas por (AKAO 2004): *benchmarking* e regressão linear primitiva. No entanto, os autores citam que estas metodologias, não se aplicam na avaliação do problema, causado pelos itens de satisfação do cliente. Por isto, propõem uma metodologia inovadora, para quantificar as metas técnicas do *SQFD* para resolver esta questão.

Não há dúvida de que, fabricar um produto ou fabricar um software, tem diferenças suficientes que tornam inviável o uso pleno do *QFD*. No entanto, muitos conceitos importantes desta técnica são úteis, principalmente na especificação de requisitos. A matriz *HOQ – House of Quality*, pode ser utilizada parcialmente, na identificação e priorização dos requisitos de software conforme a ótica do Usuário. Da mesma forma que o histórico mostra, o sucesso da aplicação do *QFD* na indústria de manufaturados, o método também se aplica, embora parcialmente, na fabricação de software.

3.4. Infra-Estrutura (Hardware)

A Computação está inserida na área de engenharia de sistemas e, é um segmento relativamente novo, se comparado com outras áreas. Experimenta uma evolução acentuada e contínua já há algumas décadas, infiltrando-se no cotidiano do homem, para controlar objetos que manuseia ou, para construir algo ao qual se denomina de software.

No mundo atual, muitos artefatos criados pelo homem se tornam imprescindíveis, como os veículos de transporte com todas as suas variações, os recursos que manipulam os meios de comunicação, invenções criadas para diferentes finalidades, tais como manufatura, monitoramento, lazer, saúde, etc. Acompanhando todo este processo de evolução, o objeto abstrato denominado de software tomou consistência e hoje, está inserido no contexto da vida humana, e constitui-se de um recurso que responde em linguagem conhecida, quando solicitado, ou produz resultados que levam a

executar algum tipo de ação em um objeto. Para isto, o software usa a informação e a trata, modifica, infere e dependendo da regra ou “segmento de código” definido pelo homem, produz resultados que podem se transformar em novas informações ou ações.

Cada vez mais, o software se faz presente na vida do homem, executando tarefas repetitivas a uma velocidade antes inimaginável, através de um processador, oscilando a uma frequência medida em *MHz*, que cresce a cada dia, em função da evolução tecnológica.

De uma forma geral, as partes integrantes envolvidas em um processamento são: dados de entrada, rotinas ou regras de manipulação dos dados e resultados. Pode parecer um contexto simples, mas até o momento, é desta forma simples que são desenvolvidos os software mais complexos. Similarmente, o processamento de dados, na maioria dos computadores, segue os postulados de *Von Neuman*¹¹, manipulando bits que assumem valores binários 0 e 1 e bytes que são constituídos de 8 bits, cuja combinação resulta em uma linguagem simbólica de comunicação humana.

Hardware e Software evoluem concomitantemente, mantendo uma dependência intrínseca e circular: Hardware potentes, permitem software mais complexos, e Software complexos, requerem maior capacidade de processamento. Entenda-se por avanço de hardware, principalmente sob o ponto de vista de capacidade de processamento, ou seja, a cada dia, os fabricantes de processadores incrementam de alguma forma a capacidade de processamento dos bytes que passam pela *UCP – Unidade Central de Processamento*. As indústrias de hardware valem-se do poder do processamento paralelo e confeccionam placas com mais de um

¹¹ Von Neuman foi um matemático húngaro de origem judaica naturalizado americano nos anos 30 do século XX, que desenvolveu importantes contribuições em Mecânica Quântica, Teoria dos conjuntos, Ciência da Computação, Economia, Teoria dos Jogos e praticamente todas as áreas da Matemática. John von Neumann foi quem formalizou o projeto lógico de um computador. Dentre seus postulados, frisamos: “As instruções devem fazer operações elementares sobre os dados”.

processador também conhecido como “core 2 (duo, quad,...)” para obter maior desempenho. Investe-se ainda, no aumento da banda da via de dados, construindo-se os processadores hoje rotulados como 64 bits, fato imprescindível, uma vez que os dados crescem de forma acentuada, criando a necessidade de transportar mais e mais dados, interligando os periféricos com a *UCP*. Outro recurso que incrementa a capacidade de manipular dados é o processo de dividir o trabalho de armazenamento dos dados no disco rígido, utilizando a tecnologia *RAID*¹². Como consequência, o tempo de salvamento dos dados, no disco rígido, não é exatamente a metade, mas se torna significativamente menor. Ocorre também uma investida, embora tenha limitações físicas, no aumento da rotação dos discos rígidos, hoje operando a *7200 rpm* com alguns modelos atingindo os *10000 rpm*.

A comunicação de dados entre o processador e o disco rígido se efetua através de padrões. O padrão *Serial ATA* ou *SATA - Serial Advanced Technology Attachment* é uma tecnologia para discos rígidos, que surgiu no mercado por volta do ano 2000, para substituir a tradicional interface *PATA - Paralell ATA* ou somente *ATA* ou, ainda, *IDE*. A tecnologia *PATA* faz transferência de dados, de forma paralela, transmitindo vários bits por vez e no *SATA*, a transmissão é em série. A transmissão paralela de dados causa um problema conhecido como “ruído”, que nada mais é do que a perda de dados ocasionada por interferência. No caso do padrão *SATA*, o ruído praticamente não existe, mesmo porque seu cabo de conexão ao computador possui apenas 4 vias e também é blindado.

De forma semelhante, a “mesa de trabalho” do processador ao qual denominamos de “memória *RAM*” também é utilizada de forma paralela ao

¹² Redundant Array of Independent Disks. Trata-se de um processo para comunicação em mais de uma via visando maior desempenho na leitura e gravação dos discos rígidos.

qual se denomina “*dual channel*”. Alternativa similar é a utilização de disco rígido, com tecnologia *SCSI*¹³ muito rápido, mas muito caro.

O processamento paralelo, sob o ponto de vista do hardware, é executado por máquinas paralelas, ou seja, computadores que possuem uma arquitetura paralela. Estas máquinas dividem-se em dois tipos: a) arquitetura com memória compartilhada e; b) arquitetura com memória distribuída. As arquiteturas de memória compartilhada podem ser *SMP – Symmetric Multi Processing* ou *NUMA – Non Uniform Memory Access / CC-NUMA – Cache-Coherent NUMA*. Por outro lado, as arquiteturas com memória distribuída podem ser agrupadas em *CLUSTER* ou *MPP – Massive Parallel Processing*.

A arquitetura *SMP* é uma extensão de uma arquitetura mono-processada, onde os processadores estão conectados à memória compartilhada. Os tempos de acesso à memória são iguais para todos os processadores.

As arquiteturas *NUMA* caracterizam-se pelo acesso não uniforme à memória, onde o tempo de acesso à memória para cada processador é variável, dependendo da região de memória. Mantém, no entanto, o mesmo conceito de compartilhamento de memória da arquitetura *SMP*.

Os *CLUSTERs*, constituem arquiteturas compostas por nós de unidades de processamento interligados através de uma rede de alto desempenho (*SCI - Scalable Coherent Interface, Gigabit Ethernet, Myrinet, etc.*), podendo constituir equipamentos com milhares de nós. A programação paralela, para este tipo de arquitetura, utiliza mecanismos de troca de mensagens do tipo *PVM – Parallel Virtual Machine* ou *MPI – Message Passing Interface*. A arquitetura *CLUSTER* constitui o modelo para compor computadores *HPC - High Performance Computer*, muito utilizado em processamento científico.

Nas arquiteturas *MPP*, o termo massivo tem a conotação de uso de milhares de processadores atuando em paralelo. É a base de constituição da maioria

¹³ Small Computer System Interface. Discos rígidos mais rápidos que os padrões, opera em 10000 rpm e usa controladora especial.

dos super computadores da atualidade, concomitante com as arquiteturas *CLUSTER*.

O uso de super computadores é uma solução de alto custo e sua aplicação, é direcionada para processamentos de grande envergadura. Estas máquinas de alta performance tem sido amplamente utilizadas em grandes projetos e desde 1993, a comunidade internacional mantém um projeto denominado de “*Top500 Project*”¹⁴ cujo objetivo, é proporcionar uma base confiável para fomentar o uso de computação de alta performance, publicando duas vezes por ano uma lista dos 500 super computadores mais rápidos em operação.

As máquinas paralelas contrastam com outra estratégia de processamento em paralelo denominado de sistemas de processamento distribuído. Nesta arquitetura, as unidades de processamento não necessitam serem iguais, sob o ponto de vista de hardware nem de software. Cada unidade tem sua memória própria e o objetivo é aproveitar o poder de processamento das unidades distribuídas. A conexão entre as unidades de processamento se efetua através de uma rede, e as aplicações se comunicam através de um protocolo de troca de mensagens. As unidades de processamento podem estar fisicamente distantes, e o número de unidades que compõem a arquitetura é ilimitado, o que caracteriza, um ponto a favor da escalabilidade, ou seja, a possibilidade de incrementar a capacidade de processamento. A dificuldade maior é atribuída à comunicação entre as unidades, dependente exclusivamente das características da rede, tais como velocidade, confiabilidade, ruídos, etc., o que traz certa perturbação do sincronismo, na execução de processamento conjugado. Sob o ponto de vista de montagem da arquitetura, não existe nenhuma dificuldade, pois as unidades remotas são *PCs* comuns e a rede de conexão utilizada pode ser uma *LAN – Local Area Network* ou uma *WAN – Wide Area Network*, utilizando o protocolo *TCP-IP*. Para a comunicação entre as unidades pode-se adotar o padrão

¹⁴ www.top500.org

CORBA e neste caso, as unidades remotas podem até utilizar ambientes de processamentos diferentes.

Analisando sob o ponto de vista do software, observa-se atualmente uma busca incessante pela qualidade. Os centros de pesquisa e os pesquisadores independentes investem em metodologias de desenvolvimento, que garantam um nível de qualidade dos artefatos de software produzidos, através de métricas. Por outro lado, os processos numéricos também evoluem acentuadamente, e as combinações de métodos, ao qual denominamos de modelos híbridos, são muito comuns. O objetivo dos modelos híbridos é similar ao de veículos com motores híbridos, ou seja, buscam a otimização através da interação concomitante ou seqüencial dos métodos.

Os modelos, híbridos ou não, podem ser medidos quanto a seu desempenho. Isto permite que sejam feitas análises conclusivas, sobre o ganho eventualmente obtido por um processo de paralelização do algoritmo, seja utilizando múltiplos processadores, ou *clusters* contendo nós de unidades de processamento. Um dos parâmetros de medida de desempenho de um algoritmo é o chamado *SpeedUP* (S), definido como a razão entre o tempo levado pelo computador executando o algoritmo serial mais rápido (t_s) e o tempo levado pelo mesmo computador executando o algoritmo paralelo com p processadores (t_p). Ou seja,

$$S = t_s / t_p \quad \dots (1)$$

Outro parâmetro decorrente é o chamado de Eficiência (E), definido como sendo a razão entre o *SpeedUp* e o número de processadores utilizados. Ou seja,

$$E = S / p \quad \dots (2)$$

Todo processamento paralelo tem a sua componente seqüencial que eventualmente, limita o *SpeedUp* que seria alcançado utilizando computadores paralelos.

A codificação de programas com processamento paralelo utilizam bibliotecas específicas como *MPI – Message Passing Interface*, para comunicação entre os computadores. O fluxo do processamento deve ser programado, através deste protocolo de comunicação, e o controle da comunicação, juntamente com o balanceamento de carga, deve ser efetuado no programa principal, através do uso de semáforos¹⁵, o que exige um esforço de programação bastante significativo. É importante salientar que todo este esforço só tem significado, se existirem na infra-estrutura mais de um processador, para que as *UCPs* atuem concomitantemente, proporcionando um ganho de desempenho. No entanto, estas *UCPs* devem ser simples executando tarefas simples, de modo que o conjunto, seja capaz de efetuar um processamento de grandes proporções. O paralelismo de processamento traz um ganho de desempenho, mas é importante observar que em uma aplicação, existem trechos de programa seqüenciais que não podem ser paralelizados, tornando inócuo neste segmento de código, a existência de múltiplos processadores.

Uma inovação que constitui uma solução de software é o projeto desenvolvido pela *Sun* denominado de projeto *CAJO*. Trata-se de uma pequena biblioteca de uso livre que permite um processamento dinâmico colaborativo entre unidades de processamento utilizando Java. A ferramenta apresenta uma facilidade de uso surpreendente, segundo a *Sun*, constituindo-se em um *framework* de fácil compreensão que simplifica o uso de *RMI*, proporcionando de maneira prática o processamento distribuído, sem necessitar de requisitos específicos ou de modificações da infra-estrutura. *CAJO* permite a criação de múltiplas *JVM – Java Virtual Machines*

¹⁵ Recurso utilizado para controlar a execução de tarefas nos núcleos de processamento

trabalhando em conjunto como se fosse uma unidade de processamento. É uma solução de software para construir um *Cluster* virtual, aproximando-se muito do conceito de *grid computing*.

Concluindo, pode-se dizer que hardware e software caminham lado a lado, interagindo de forma produtiva e evoluindo cada qual à sua maneira.

Outra questão que influi no desenvolvimento deste trabalho refere-se ao Usuário e suas necessidades. As indústrias de manufatura obtêm sucesso na venda de seus produtos porque inserem no processo de fabricação, as necessidades do Usuário do produto. Para o software não poderia ser diferente. Quanto mais os anseios do Usuário estiverem presentes no processo de fabricação do software, melhor a aceitação do mesmo o que equivale a conferir ao artefato, uma vida longa e pacífica com o Usuário. O grande segredo do sucesso de um software é compreender os desejos do Usuário e transformá-los em especificação de projeto, utilizando os conceitos de engenharia de software. Moldar um software às necessidades do Usuário significa criar recursos que o permitam, atingir seus objetivos de forma “*correta*” e “*confortável*”. O termo “*correta*” significa exatidão e isto, leva à idéia de que o software executa todas as suas funcionalidades, da forma como foi especificado. Quanto mais o software produz resultados de processamento em conformidade com o que foi estabelecido, maior é a sua qualidade. A predominância da qualidade de um software ao longo do tempo, isto é, funcionando e operando corretamente e de acordo com os requisitos especificados, conduz a um outro parâmetro denominado de confiabilidade. Conceitualmente, é uma métrica indicativa de quão correto, um software se apresenta ao longo do tempo. O termo “*confortável*” fornece uma idéia de facilidade no manuseio do software e talvez, seja o parâmetro mais importante sob o ponto de vista do Usuário, tendo sido considerado com muita ênfase, no desenvolvimento de produtos pela indústria de manufatura. Por que não utilizar estas idéias na produção de software? Este, também é um produto e da mesma forma, será utilizado por um Usuário. É preciso ouvi-lo com muita atenção, pois se não for atendido

satisfatoriamente, o mais provável é que não irá “*comprar/usar*” o software produzido.

Sob o aspecto usabilidade, é necessário preocupar-se também com o lado prático, evitando criar aplicações complexas que tornam o manuseio bastante difícil e inadequado. Em geral, os Usuários manifestam certa resistência com operações complicadas, e sempre que encontra dificuldades para operar um sistema, tendem a concluir que o mesmo não os atende. O argumento de que o software é melhor porque utiliza um complicado algoritmo X para resolver um problema Y não os convence. Se ele não perceber (estar convencido) a importância do algoritmo X, ele não irá pagar o preço cobrado.

Há muitas maneiras de administrar as atividades de TI, com o objetivo de melhorar os serviços que a área oferece às corporações. O desenvolvimento de sistemas constitui como um dos seus serviços e é beneficiado pelo uso destas práticas, que regulam e monitoram a forma de trabalho, elevando o nível de qualidade. As práticas de TI existentes tais como *CMMI*, *TSP*, *RUP*, *Cobit* ¹⁶ etc. são metodologias que tem como objetivo, a fabricação de software com qualidade, retirando a dependência pessoal, além de definir de forma clara e detalhada, atores e papéis (*roles*) que participam no processo de software, estabelecendo as responsabilidades de cada um.

¹⁶ São práticas de TI que visam a qualidade.

4. ESTADO DA ARTE

O PG – Programa de Geração elétrica é, em sua essência, um problema de otimização que continua a apresentar um grande desafio, na busca de soluções eficientes. Neste sentido, muitas pesquisas têm sido desenvolvidas, utilizando-se metodologias combinadas, como relatadas em (DAHALL 2007), onde são utilizados modelos híbridos, estratégia muito comum observada nas recentes pesquisas efetuadas sobre o assunto. O método utilizado para solucionar o problema *PG*, baseou-se no uso de diversas meta-heurísticas, heurísticas e recursos de programação matemática, para identificar rapidamente soluções muito próximas da solução ótima. Dentre as diversas alternativas híbridas aplicadas ao problema *PG*, os autores ressaltam o uso do procedimento híbrido *AG* – *Algoritmo Genético*, como um dos métodos recomendados para aplicação em problemas de grandes proporções.

Em um sistema misto envolvendo hidroelétricas e termelétricas a programação de geração energética é referenciada como Sistema Hidro-Térmico. O objetivo é determinar a configuração que apresente o menor custo de geração das usinas térmicas, para um determinado período de tempo. Nas considerações de (ZOUHAS 2004), este caso é um problema de otimização linear mista, sendo necessária a determinação das variáveis de decisão inteira e contínua. O resultado deste trabalho, segundo os autores, traz como benefício a redução do custo de geração global além de proporcionar maior confiabilidade ao sistema de geração, atendendo às restrições e objetivos de outras naturezas.

Uma outra importante alternativa de produção de energia elétrica é a combinação de produção de calor e energia elétrica, denominada de co-geração. A operação eficiente destas usinas, sob o ponto de vista de custo, pode ser modelada como um problema de otimização utilizando a PL – Programação Linear. Uma solução para o problema de otimização de uma usina com co-geração é descrita em (LAHDELMMA 2003) onde os autores modelaram a operação horária da usina, como um problema de

Programação Linear, utilizando diversas versões modificadas do método Simplex. Segundo os autores, resolver o problema de otimização horária (problema menor e solúvel) é interessante, uma vez que a otimização de médio prazo (problema maior, de difícil solução) pode ser resolvida, decompondo-a em problemas menores de curto prazo.

Como alternativa de solução, a Relaxação Lagrangeana tem sido aplicada em vários problemas de otimização combinatória, sendo válido também no problema de otimização da geração de energia elétrica. Essa relaxação é obtida multiplicando-se um conjunto de restrições por um vetor M de multiplicadores de Lagrange, adicionando-se o produto à função objetivo. A combinação deste método com a programação dinâmica é uma experiência diferente observada em (ENCINA 2006). Neste caso, o foco é direcionado para a determinação da programação diária ótima para o dia seguinte, em conformidade com o planejamento energético de médio prazo, constituindo desta forma, um recurso adicional para a operação em tempo real. A otimização proposta visa essencialmente minimizar os custos de perdas e os custos de parada/partida das unidades geradoras.

A metaheurística “Busca Tabu” também tem sido objeto de estudo, na programação de geração de energia elétrica, como descrito em (RAJAN 2004). Neste caso, o objetivo é identificar quais unidades geradoras devem atuar no sistema, de modo a minimizar o custo de operação total, ao longo de um período de tempo. Para este tipo de problema, a solução ótima só pode ser obtida através da enumeração completa de todas as possibilidades, o que torna sua solução inviável. Normalmente, é tratado como um problema de otimização inteira mista não linear de grandes proporções, onde os autores propõem soluções através da “Busca Tabu”, considerado um procedimento bastante utilizado em problemas deste tipo. Citam os autores os inúmeros métodos pesquisados na bibliografia para análise, tais como: *Programação Dinâmica*, *Programação Inteira mista*, *Relaxação Lagrangeana*, *Branch and Bound*, *Sistemas Especialistas*, *Método Fuzzy*, *Busca Tabu*, *Algoritmo Genético* e outros. Todos estes

métodos visam buscar soluções próximas da solução ótima de modo eficiente.

Em recente artigo, (YU 2005) tem diagnosticado como uma tendência atual em vários países, a privatização das companhias geradoras de energia elétrica, tornando-se independentes e por isso, atuando no mercado de acordo com seus próprios interesses, visando principalmente o lucro. Neste cenário, cada companhia de geração se preocupa em atingir seus objetivos, muitas vezes adotando um plano de geração indevido, gerando um mercado caótico de produção de energia. YU propõe a utilização de um sistema *Multi Agentes*, para auxiliar as empresas de geração, na atividade de programar a produção de energia elétrica. Os autores criaram três diferentes papéis para os *Agentes* que interagem entre si, para buscar uma solução próxima da solução ótima. De acordo com os autores deste trabalho, este estudo apresentou resultados comparáveis com os obtidos pelo método *LR-EP – Lagrange Relaxation – Evolutionary Programming*.

Em outra aplicação *Multi Agentes*, na área de geração de energia, é o que relata (NAGATA 2002), propondo resolver o problema de decidir, quais unidades geradoras devem estar operando, para suprir a demanda ao mínimo custo. Em geral, o custo é formado pelo custo de produção de energia elétrica, onde prepondera o custo do combustível, e pelo custo de inicialização da unidade geradora. O modelo de *Nagata* é constituído de um *Agente Facilitador*, de vários *Agentes Geradores* e dois tipos de *Agentes Móveis*. O *Agente Facilitador* tem a incumbência de atuar como coordenador do processo, e os outros *Agentes* possuem apenas estratégias de negociação. Para a comunicação entre os *Agentes*, foi utilizado um padrão do tipo *KQML – Knowledge Query and Manipulation Language*. De acordo com os autores, o estudo apresentou resultados compatíveis com o método de programação dinâmica, mostrando a viabilidade de utilização de sistemas *Multi Agentes* em problemas de otimização.

São inúmeras as propostas de soluções para o problema otimização da geração de energia. Todos os esforços buscam o mesmo propósito de

encontrar boas soluções com eficiência, de modo a suprir os tomadores de decisão com informações confiáveis, em tempo adequado.

5. DESENVOLVIMENTO DO TEMA

O tema estabelecido para esta tese compreende a solução de um problema de engenharia, considerado de alta complexidade, do setor de produção de energia elétrica, envolvendo a área de modelagem e desenvolvimento de software. O foco principal do tema refere-se à pesquisa e aplicação, de métodos e recursos computacionais que possibilitem, a solução do problema de otimização de geração de energia elétrica, com foco na qualidade.

Conforme visto no capítulo anterior, as soluções para este tipo de problema são variadas, mas a tese em questão, não se limita simplesmente à solução, como tem sido visto na bibliografia sobre o assunto, relatado em “Estado da Arte”. Agrega, além disso, uma metodologia que determina de que maneira se obtém um produto de qualidade, atendendo aos requisitos do Usuário, através do estudo dos processos numéricos, concomitantemente com a infra-estrutura necessária.

Para estabelecer os requisitos de projeto do software, utilizou-se a metodologia *QFD*, comprovadamente eficiente neste quesito, conforme descrito no capítulo 3.3.3.

Para auxiliar no processo de análise das diversas alternativas envolvendo os processos numéricos, foi necessária a criação de um modelo conceitual, detalhado em capítulo subsequente. O resultado decorrente deste modelo constituiu-se de um arcabouço do processo numérico a ser utilizado, sendo submetido, juntamente com os requisitos estabelecidos, ao processo de avaliação da importância de cada requisito de projeto, utilizando a *Casa da Qualidade*, conforme consta na metodologia *QFD*.

Com os resultados obtidos, elaborou-se uma arquitetura inovadora que possibilitou a montagem de um *cluster virtual*, valendo-se do conceito de processamento distribuído e das facilidades de um *middleware*, disponível para a plataforma de desenvolvimento.

Desta forma, obteve-se um produto de qualidade, com um poder de processamento escalável, de baixo custo e fácil implementação, utilizando métodos híbridos para solução do problema de otimização. A arquitetura elaborada tem a característica de permitir, de maneira ilimitada, expansão em duas direções: na capacidade de processamento e na sofisticação dos processos numéricos, o que representa um importante aspecto estratégico desta arquitetura, pois permite a evolução progressiva dos processos numéricos obtendo como consequência o refinamento dos resultados.

5.1. INEDITISMO

Constitui uma estratégia inovadora, a proposta formulada nesta tese, de utilizar uma metodologia de análise de processos numéricos em conjunto com a metodologia *QFD*, para definir uma arquitetura híbrida que apresente um poder de processamento escalável, de baixo custo e fácil implementação, visando a solução de um problema de otimização.

O modelo de análise de processos numéricos, detalhado em capítulo seguinte é uma abordagem inédita, concebida para dar suporte ao desenvolvimento deste trabalho podendo, no entanto, ser utilizada em problemas similares. A razão para adotar esta abordagem deve-se ao fato de existirem diversas alternativas de solução, sendo necessário um processo decisório para avaliar e mensurar a opção mais indicada.

A estrutura do tema enfatiza o aspecto da obtenção da qualidade do software, especificamente em modelos, o que não tem sido relatado na bibliografia, muito menos com um modelo como descrito adiante.

O processamento distribuído, antes implementado em *clusters* ou máquinas com *CPUs* múltiplas, constituíam soluções de hardware aglomerado. No entanto, algumas novas tendências têm sido observadas em relação ao processamento distribuído. Através de recursos de software, existem

soluções que distribuem processamentos em unidades remotas¹⁷, sem que ocorra a aglomeração do hardware. São as soluções intituladas *Grid Computing*. Muitas empresas estão investindo neste segmento, como a *Oracle*, a *IBM* e a *Sun*. A idéia do *Grid Computing* apareceu nos anos 90 como uma nova infra-estrutura de computação distribuída, concebida com o objetivo de auxiliar em projetos de pesquisa de grande porte, valendo-se de unidades remotas, muitas vezes heterogêneas para solução de parte de um problema maior. Vários ambientes de desenvolvimento já possuem bibliotecas para distribuição de processamento, como o projeto *CAJO* da *Sun Microsystems*, uma biblioteca que possibilita a distribuição de processamento criando um *cluster virtual* e o *middleware JADE*, uma plataforma de código aberto, que possibilita a utilização de aplicações, baseada em agentes que podem ser distribuídas em rede.

Na arquitetura proposta, além de um modelo híbrido, utilizou-se o conceito de processamento distribuído, valendo-se dos recursos de um *middleware*, para sincronizar as atividades nas unidades remotas e na principal, obtendo com isto, o poder do processamento paralelo ilimitado. As porções de código que são executadas nas unidades remotas constituem componentes de software, do tipo *multi threading*¹⁸.

Os capítulos seguintes descrevem com os detalhes necessários, toda a conceituação da metodologia concebida para o desenvolvimento desta tese.

¹⁷ unidades remotas são unidades de processamento que estão fisicamente separadas de uma unidade central

¹⁸ termo utilizado para referir-se a capacidade de processar programas independentes e concorrentes na memória.

5.2. Lista de Requisitos

A lista de requisitos foi elaborada, utilizando-se os fundamentos da metodologia *QFD*, descrita anteriormente no capítulo 3.3.3. O fluxo do processo de determinação dos requisitos proposto pela metodologia, está ilustrado na figura 5.2-1 seguinte.

Figura 5.2-1 Fluxo de Processo do QFD

Os requisitos foram levantados através de formulários eletrônicos, junto aos Usuários do sistema utilizado pelo *ONS*, constituindo a “Voz do Usuário”, conforme consta na metodologia. As considerações apresentadas pelos Usuários são variadas e em grande número, apresentando itens similares. Nem sempre os itens mencionados podem ser transformados em requisitos de software. Para resolver esta dificuldade, a metodologia propõe a criação de um grupo denominado de “*cross functional team*” que tem a atribuição de analisar as opiniões do Usuário, eliminando redundâncias e definindo os principais requisitos, desdobrando-os em diversos níveis de detalhamento, até que os itens resultantes possam ser transformados em requisitos de projeto. Além disso, este grupo também tem a incumbência de detectar itens

não relatados pelos Usuários, com o objetivo de criar um produto de qualidade, surpreendendo-os com requisitos que vão além do previsto.

Os requisitos levantados foram compilados e desdobrados em subitens até um nível que pudessem ser transformados em requisitos de software, conforme apresentado na tabela seguinte.

Tabela 5.2-1 Desdobramento dos Requisitos

Nível Primário	Nível Secundário	Nível Terciário
Produto prático de qualidade	Ser Eficiente	Apresentar Conformidade
		Consumir Tempo de Resposta Reduzido
		Produzir Resultados Desejados
	Apresentar Viabilidade Econômica	Custo de Desenvolvimento Reduzido
		Custo de Infra Estrutura Viável
		Custo de Implantação Baixo
	Ser Prático	Possuir Boa Usabilidade
		Permitir Rastreabilidade
		Optar por Algoritmos Simples
		Ser Fácil de Implementar
		Ser Fácil de Manter
		Permitir Interação
		Permitir Expansão
		Desenvolver em Módulos

Um dos parâmetros utilizados pelo *QFD* são os valores numéricos que representam a importância relativa entre requisitos. Segundo (AKAO 2004), isto pode ser levantado junto aos Usuários através de uma pesquisa, desde que a amostra seja suficientemente grande, para validar estatisticamente os parâmetros. Nem sempre a amostra é suficientemente grande e neste caso, Akao sugere a utilização do método *AHP – Analytic Hierarchic Process* desenvolvido por Thomas L. Saaty, para determinar a importância relativa dos requisitos. O mesmo processo é descrito em (MADU 2006) ilustrando de forma sucinta, casos práticos de utilização deste método.

Para utilizar o método *AHP*, é necessário utilizar os valores de importância relativa fornecidos pelo método, conforme a tabela abaixo:

Tabela 5.2-2 Valores de Importância Relativa

Critério	Valor
Requisito i e j são equivalentes	1
Requisito i é fracamente mais importante que j	3
Requisito i é mais importante que j	5
Requisito i é muito mais importante que j	7
Requisito i é absolutamente mais importante que j	9

Utilizando-se os valores acima, preenche-se a tabela de relacionamento dois a dois, atribuindo um valor para cada célula (i,j) , se o requisito da linha i for dominante ou, (j,i) em caso contrário. As células “opostas” em relação à diagonal são preenchidas com o inverso da célula preenchida. As células da diagonal obviamente assumem o valor unitário.

Assim, na tabela 5.2-3 seguinte, no cruzamento da linha “Custo de Infra Estrutura Viável” e da coluna “Possuir Boa Usabilidade”, foi atribuído o valor 3 (veja critério na tabela 5.2-2) indicando que o requisito “Custo de Infra Estrutura Viável” é dominante. De acordo com a tabela 5.2-2 anterior, entende-se que o requisito “Custo de Infra Estrutura Viável” é fracamente mais importante que “Possuir Boa Usabilidade”. A célula diagonalmente oposta na matriz, é preenchida com o inverso do valor atribuído, ou seja, $1/3 = 0.33$.

Os valores de importância relativa são somados na vertical conforme pode ser visualizado na linha “Somatória”.

Tabela 5.2-3 Matriz de Importância Relativa

	Apresentar Conformidade	Consumir Tempo de Resposta Reduzido	Produzir Resultados Desejados	Custo de Desenvolvimento Reduzido	Custo de Infra Estrutura Viável	Custo de Implantação Baixo	Possuir Boa Usabilidade	Permitir Rastreabilidade	Optar por Algoritmos Simples	Ser Fácil de Implementar	Ser Fácil de Manter	Permitir Interação	Permitir Expansão	Desenvolver em Módulos
Apresentar Conformidade	1	1	1	3	5	5	7	7	9	9	9	9	7	9
Consumir Tempo de Resposta Reduzido	1	1	1	3	1	3	5	7	7	9	9	9	9	9
Produzir Resultados Desejados	1	1	1	3	5	7	7	7	9	9	9	9	9	9
Custo de Desenvolvimento Reduzido	0,33	0,33	0,33	1	1	1	3	7	9	9	9	9	9	9
Custo de Infra Estrutura Viável	0,2	1	0,2	1	1	1	3	5	9	9	9	9	9	9
Custo de Implantação Baixo	0,2	0,33	0,14	1	1	1	9	3	5	7	1	9	9	9
Possuir Boa Usabilidade	0,14	0,2	0,14	0,33	0,33	0,11	1	0,33	5	3	1	5	7	7
Permitir Rastreabilidade	0,14	0,14	0,14	0,14	0,2	0,33	3	1	9	5	3	3	0,11	3
Optar por Algoritmos Simples	0,11	0,14	0,11	0,11	0,11	0,2	1	0,11	1	1	0,2	0,2	0,11	0,33
Ser Fácil de Implementar	0,11	0,11	0,11	0,11	0,11	0,14	5	0,2	1	1	1	5	9	9
Ser Fácil de Manter	0,11	0,11	0,11	0,11	0,11	1	7	0,33	5	1	1	1	0,11	0,33
Permitir Interação	0,11	0,11	0,11	0,11	0,11	0,11	9	0,33	5	9	1	1	0,2	3
Permitir Expansão	0,14	0,11	0,11	0,11	0,11	0,11	1	9	9	3	9	5	1	1
Desenvolver em Módulos	0,11	0,11	0,11	0,11	0,11	0,11	1	0,33	3	3	3	0,33	1	1
Somatória	4,72	5,71	4,63	13,1	15,2	20,1	62	47,6	86	78	65,2	74,5	70,5	78,7

Tabela 5.2-4 Matriz de Importância Relativa Normalizada

Requisito	Valores de importância relativa normalizados														Média
Apresentar Conformidade	0,21	0,18	0,22	0,23	0,33	0,25	0,11	0,15	0,10	0,12	0,14	0,12	0,10	0,17	
Consumir Tempo de Resposta Reduzido	0,21	0,18	0,22	0,23	0,07	0,15	0,08	0,15	0,08	0,12	0,14	0,12	0,13	0,14	
Produzir Resultados Desejados	0,21	0,18	0,22	0,23	0,33	0,35	0,11	0,15	0,10	0,12	0,14	0,12	0,13	0,18	
Custo de Desenvolvimento Reduzido	0,07	0,06	0,07	0,08	0,07	0,05	0,05	0,15	0,10	0,12	0,14	0,12	0,13	0,09	
Custo de Infra Estrutura Viável	0,04	0,18	0,04	0,08	0,07	0,05	0,05	0,10	0,10	0,12	0,14	0,12	0,13	0,09	
Custo de Implantação Baixo	0,04	0,06	0,03	0,08	0,07	0,05	0,15	0,06	0,06	0,09	0,02	0,12	0,13	0,07	
Possuir Boa Usabilidade	0,03	0,04	0,03	0,03	0,02	0,01	0,02	0,01	0,06	0,04	0,02	0,07	0,10	0,03	
Permitir Rastreabilidade	0,03	0,03	0,03	0,01	0,01	0,02	0,05	0,02	0,10	0,06	0,05	0,04	0,00	0,03	
Optar por Algoritmos Simples	0,02	0,03	0,02	0,01	0,01	0,01	0,02	0,00	0,01	0,01	0,00	0,00	0,00	0,01	
Ser Fácil de Implementar	0,02	0,02	0,02	0,01	0,01	0,01	0,08	0,00	0,01	0,01	0,02	0,07	0,13	0,03	
Ser Fácil de Manter	0,02	0,02	0,02	0,01	0,01	0,05	0,11	0,01	0,06	0,01	0,02	0,01	0,00	0,03	
Permitir Interação	0,02	0,02	0,02	0,01	0,01	0,01	0,15	0,01	0,06	0,12	0,02	0,01	0,00	0,03	
Permitir Expansão	0,03	0,02	0,02	0,01	0,01	0,01	0,02	0,19	0,10	0,04	0,14	0,07	0,01	0,05	
Desenvolver em Módulos	0,02	0,02	0,02	0,01	0,01	0,01	0,02	0,01	0,03	0,04	0,05	0,00	0,01	0,02	

Esta matriz de valores de importância relativa é recalculada dividindo-se o valor de cada célula pela somatória da coluna. Desta forma, os valores ficam todos normalizados entre 0 e 1.

A tabela 5.2-4 apresenta os valores recalculados desta forma, e na última coluna foi calculada a média (horizontal) de cada requisito. Esta coluna representa, portanto, a importância relativa de cada item.

Os requisitos são ordenados em ordem decrescente pela média dos valores normalizados de importâncias relativas, resultando na tabela 5.2-5 seguinte.

Os dois primeiros requisitos presentes na tabela são requisitos óbvios que devem ser atendidos. O terceiro item refere-se à necessidade de obter uma solução com tempo de resposta reduzido.

Os custos vêm em sequência, seguido de outros itens de qualidade.

Tabela 5.2-5 Requisitos em ordem de importância

Requisito	Importância
Produzir Resultados Desejados	0,183
Apresentar Conformidade	0,173
Consumir Tempo de Resposta Reduzido	0,143
Custo de Infra Estrutura Viável	0,093
Custo de Desenvolvimento Reduzido	0,092
Custo de Implantação Baixo	0,073
Permitir Expansão	0,051
Permitir Rastreabilidade	0,035
Possuir Boa Usabilidade	0,035
Permitir Interação	0,034
Ser Fácil de Implementar	0,031
Ser Fácil de Manter	0,027
Desenvolver em Módulos	0,019
Optar por Algoritmos Simples	0,011

Alguns requisitos nesta lista são de natureza não funcional. É o caso de “Custo de Desenvolvimento Reduzido”, “Custos de Infra-Estrutura Viável” e

“Custo de Implantação Baixo”. Por outro lado, há um conjunto de requisitos que dependem do processo numérico adotado e que devem ser analisados, em função das características de cada algoritmo.

5.3. House of Quality

O método *QFD* utiliza um gráfico de análise denominado de *House of Quality* ou *Casa da Qualidade*. Os requisitos estabelecidos pelo Usuário constituem um conjunto denominado de “*what to do*”, ou simplesmente “*WHAT*” e os meios, através dos quais os requisitos são resolvidos, formam o conjunto de requisitos de projeto ao qual se denomina “*how to do*” ou simplesmente “*HOW*”. O primeiro conjunto é disposto em linhas, à esquerda em ordem decrescente de importância relativa, conforme apresentada na tabela 5.2-5 constante no item anterior. O segundo conjunto é disposto em colunas na parte superior. Os cruzamentos das linhas e colunas constituem as células onde são inseridas as avaliações relativas entre o requisito do Usuário e o requisito de projeto.

A parte superior da *Casa da Qualidade* representa a *Matriz de Correlação* entre os requisitos de projeto. Tem por objetivo, identificar requisitos que apresentam efeitos contrários. A avaliação é efetuada utilizando-se a legenda “*Correlações*” constante na figura 5.3-1.

Neste caso, praticamente todos os requisitos são considerados positivos ou, fortemente positivos, quando comparados dois a dois.

Na tabela 5.3.1 a legenda “*Relacionamentos*” indica a pontuação para cada avaliação relativa, efetuada entre os requisitos do Usuário e os requisitos de projeto. Com os valores constantes na legenda, efetuam-se os cálculos para cada célula, multiplicando a importância do Usuário da primeira coluna com o valor correspondente do símbolo atribuído para a avaliação relativa. Assim, para a célula correspondente ao cruzamento da linha “*Possuir Boa Usabilidade*” com a coluna “*Interfaces IHM bem projetadas*” a avaliação foi “@”, o que segundo a legenda, indica “*Forte*” relacionamento, sendo

pontuado como 9. A primeira coluna de valores desta linha, indica a importância do Usuário como sendo 0.035 logo, o cálculo efetuado para esta

Tabela 5.3-1 Casa da Qualidade

Legenda Relacionamentos	@	Forte	=	9
	O	Médio	=	3
	Δ	Fraco	=	1

Legenda Correlações	⊙	Forte +
	O	Positiva
	X	Negativa
	*	Forte -

célula é: $9 * 0.035 = 0.315$. De maneira similar, calculam-se os valores para todas as células da coluna, efetuando a somatória da coluna, sendo esta, indicativo do peso da importância do requisito de projeto. Efetuando o mesmo procedimento para todas as colunas, foram obtidos os valores de todos os requisitos de projeto que, depois de ordenados em ordem decrescente, resultaram da tabela seguinte.

Tabela 5.3-2 Requisitos de Projeto Ordenados

Requisitos de Projeto	Pesos
Processamento Distribuído	13,66
Modelos Híbridos	13,45
Uso de Componentes	9,84
Relatórios de investigação	1,92
Desenvolvimento Modular	1,20
Pontos de controle	1,09
Interfaces IHM bem projetadas	1,05
Tratamento de erros	0,43
Testes Documentados	0,26

Os valores indicados na segunda coluna representam os pesos de importância de cada um dos requisitos, determinando o esforço que deve ser direcionado para cada item, para que os objetivos estabelecidos pelos Usuários sejam atingidos.

Os requisitos de projeto de maior peso referem-se ao uso de “*Processamento Distribuído*” e “*Modelos Híbridos*”, seguido por outros itens de peso menor. A composição dos modelos híbridos que fornecem soluções para problemas de otimização é muito diversificada, sendo uma tarefa complexa estabelecer a melhor combinação de métodos.

Para facilitar as análises dos diversos processos numéricos que poderiam compor um modelo híbrido, foi necessário criar um modelo de análise descrito no capítulo seguinte, cuja finalidade é proporcionar uma visualização comparativa das alternativas de modo a optar pela combinação mais adequada ao problema.

5.4. Modelo MC - HSQ

O termo MC – HSQ, significa *Modelo Conceitual* para solução de problema de otimização, analisando a infra-estrutura necessária (Hardware) para processar os algoritmos (Software), de modo a atender parte dos requisitos estabelecidos para a solução do problema.

Este modelo não se limita ao âmbito de otimização de geração de energia elétrica, podendo ser aplicado a qualquer tipo de problema.

A primeira providência neste modelo é estabelecer a lista de requisitos desejados, para o artefato a ser produzido. Recomenda-se listar no mínimo, os requisitos considerados relevantes ou, passíveis de análise diferenciada, uma vez que os requisitos mais simples são automaticamente atendidos, pelos métodos de desenvolvimento habituais.

No próximo passo, são efetuadas pesquisas sobre os métodos numéricos que fornecem soluções para o problema, independentemente do atendimento ou não aos requisitos estabelecidos. No entanto, ao efetuar as pesquisas, as particularidades das soluções de cada alternativa devem ser levantadas para alimentar o modelo.

Em seguida, efetua-se uma análise da infra-estrutura necessária para processar os diversos métodos numéricos selecionados, da mesma forma, identificando suas peculiaridades tais como custo, tempo de resposta etc.

A última etapa do modelo consiste em atribuir valores ou classes de valores, indicativos da aderência da alternativa, no atendimento ao requisito em questão.

Para facilitar a análise, foi elaborada uma matriz de avaliação e decisão onde são colocados à esquerda, os requisitos desejados. Na parte superior são colocados os processos numéricos e na parte inferior, a infra-estrutura necessária. Os cruzamentos das linhas e colunas indicam a avaliação do atendimento ao requisito, para o processo numérico constante na coluna e a infra-estrutura correspondente na parte inferior.

A tabela 5-4.1 Matriz MC-HSQ constitui o recurso gráfico para avaliar o conjunto de alternativas de software selecionados.

Tabela 5.4-1 Matriz MC-HSQ

MC-HSQ : Modelo Conceitual para solução de Problemas Complexos						
Qualidade		Software				
Especificação de Requisitos		Métodos que permitem solução do problema				
Objetivo	Item requerido (Exemplos)	Método 1	Método 2	Método 3	:	Método n
Requisito 1	Modularidade					
Requisito 2	Manutenabilidade					
Requisito 3	Rastreabilidade					
Requisito 4	Interatividade					
Requisito 5	Usabilidade					
Requisito 6	Rapidez de Resposta	C	C	A	A	A
Requisito 7	Complexidade reduzida	A	C	A	A	A
Requisito 8	Conformidade	A	A	B	A	A
...	...	...	...	...	...	...
Requisito n	Fácil Implementação	A	C	A	A	A
		Alta	Alta	Normal	Normal	Normal
Legenda		Capacidade de Processamento exigida				
Atende ao requisito requerido com suficiência	A					
Atende ao requisito requerido parcialmente	B					
Não atende ao requisito requerido		C				
		Hardware				

A avaliação pode ser efetuada por números, adotando-se a somatória como elemento decisor, ou por letras (A,B,C,...N) indicando uma graduação de aderência crescente ou decrescente e também por cores (VERMELHO: não atende; VERDE: atende plenamente; AMARELO:atende parcialmente, etc.) podendo também utilizar-se de recursos combinados.

Observe-se que as linhas correspondem aos requisitos, e estas podem ter avaliação vazia, subentendendo que independem do processo numérico e, portanto, não há necessidade de avaliação, pois serão plenamente

atendidos, pelos métodos habituais de desenvolvimentos de software, não tendo nenhuma influência na análise envolvendo os processos numéricos listados como “Software”.

Após a avaliação qualitativa/quantitativa das alternativas (células), podem-se tomar decisões considerando os tópicos seguintes:

- Fixar um método numérico. Nesta decisão, fica definida a infraestrutura necessária e alguns requisitos, podem não ser atendidos. Isto equivale a relaxar parte dos requisitos, ou seja, no caso acima, fixando-se o método 2, os requisitos 6, 7 e 9-n não são atendidos;
- Pode-se supor que a infraestrutura esteja previamente definida, por exemplo, “*Capacidade de processamento Alta*”, conforme consta na parte inferior direita da tabela 5.4-1. Neste caso, verificam-se quais requisitos são atendidos e decide-se pelo melhor método numérico. É possível também relaxar parte dos requisitos;
- Neste modelo, é possível também analisar sobre um determinado requisito. Isto equivale a fazer uma leitura da linha correspondente ao requisito. No exemplo acima, o requisito de número 8 é praticamente atendido e isto, pode levar a conclusões interessantes para a solução do problema;
- No caso de avaliar as alternativas por valores numéricos, a somatória das colunas indica a pontuação do método, utilizando a infraestrutura correspondente. Neste caso, é necessário adotar classes de valores para atribuir a pontuação em cada alternativa. A tabela 5.4-2 ilustra uma maneira de avaliar quantitativamente as alternativas, de 0 a 10.

Tabela 5.4-2 Pontuação MC-HSQ

		Pontuação		
		10	5	0
Requisito 1	Item 1	até 5 un.	até 10 un.	acima 10 un.
Requisito 2	Item 2	20 un.	100 un.	Acima 100 un.
Requisito 3	Item 3	Situação boa	Situação média	Situação ótima
Requisito 4	Item 4	De 1 a 10	De 10 a 100	Acima de 100
...	...	...	...	...

As seguintes indagações podem ser respondidas por este modelo:

- A lista de requisitos é viável?
- Quais requisitos podem ser relaxados?
- Qual a infra-estrutura mínima necessária?
- Qual é o nível de dificuldade de implantação da solução?

O modelo MC-HSQ acima descrito foi aplicado ao problema de otimização da geração de energia elétrica no médio prazo, obtendo-se os resultados descritos a seguir. Da lista completa dos requisitos definidos anteriormente, foram extraídos os seguintes requisitos que são influenciados pelos processos numéricos:

- Tempo de Resposta;
- Complexidade Reduzida;
- Custo de Desenvolvimento;
- Conformidade;
- Hardware Econômico;
- Facilidade de Implementação.

Os demais são atendidos pelo processo de fabricação do software e estão fora do escopo de análise deste modelo. Os métodos numéricos analisados que fornecem soluções para o problema de geração de energia foram selecionados, em função das pesquisas efetuadas no capítulo “*Conceitos*” e

da tendência atual dos estudos sobre o assunto. A tabela 5.4-3 seguinte constitui o conjunto de processos analisados:

Tabela 5.4-3 Processos Numéricos

Software				
Métodos que permitem solução do problema				
Branch and Bound	Programação Dinâmica	Meta Heurísticas	Pesquisa Operacional	Multi Agentes

Segue abaixo a descrição dos itens para montagem do modelo:

- Exatidão nos Cálculos: item Conformidade

Como já salientado em itens anteriores, a conformidade é imprescindível e este requisito, é necessário e deve ser implementado passo a passo, uma vez que os processos numéricos são complexos tornando difícil a verificação da conformidade global do modelo.

- Análise de Resultados: item Obtenção de Resultados

O conjunto de saídas do software tem necessidade de ser abrangente, tornando possível a extração de relatórios para análise e a exportação para planilhas de cálculo de modo facilitado.

- Operação facilitada: item Usabilidade

É necessário estar atento ao projetar as interfaces e a estrutura de navegação, de modo que a operação do modelo seja simples e funcional, permitindo maior desempenho nas tarefas.

- Desempenho em Simulações: item Tempo de Resposta

O modelo pretende ser utilizado em uma atividade de planejamento onde o Usuário interage com o mesmo executando diversos processamentos. Portanto, há uma expectativa de que o tempo de

resposta do modelo para as simulações, seja um tempo considerado adequado para este tipo de utilização.

- Viabilidade de Implantação: item Custo de Desenvolvimento

Dependendo do algoritmo, o custo de desenvolvimento pode se tornar bastante alto o que não é desejável. É um requisito estratégico e não funcional.

- Mecanismos de Verificação: item Rastreabilidade

No desenvolvimento de algoritmos, a conformidade é uma necessidade imperativa, ou seja, os resultados necessitam estar corretos e para isso, o recurso utilizado pelos Usuários é a rastreabilidade. Neste aspecto, são requeridos para o modelo, mecanismos de rastreamento de variáveis configuráveis, e que permita exportar os dados para análise gráfica ou, na forma de relatórios.

- Viabilidade da Infra-Estrutura: item Hardware econômico

Este item não é um requisito funcional, mas de viabilidade de implantação do modelo. É desejável que a infra-estrutura da alternativa escolhida tenha um custo condizente com a necessidade. Havendo possibilidade, alternativas de infra-estrutura mais barata agregam maior pontuação na escolha.

- Operacionalização: item Implantação

A implantação não é um item funcional, mas é desejável que esta atividade seja a mais simples possível.

- Expansão do Modelo: item Modularidade

Os processos numéricos quase sempre são complexos e a sua implantação normalmente é efetuada passo a passo, sendo verificada a conformidade, item imprescindível, em cada etapa do

desenvolvimento. Por este motivo, requer-se uma facilidade de expansão do modelo para que a implementação gradativa seja possível, garantindo a qualidade do desenvolvimento.

- Viabilidade no Desenvolvimento: item Fácil Implementação

Este requisito indica o grau de dificuldade para implementação do(s) modelo(s) escolhido(s) em termos de programação. Modelos mais complexos demandam tempos de análise, programação e depuração significativos e muitas vezes, inviabilizam a alternativa. Para modelos equivalentes, é conveniente a escolha pela implementação mais simples.

- Manutenção facilitada: item Manutenibilidade

Este item é um requisito de qualidade e tem por objetivo, dotar o modelo com recursos de manutenção de fácil operação, permitindo ao Usuário dedicar mais tempo em tarefas importantes.

- Compreensão do Modelo: item Complexidade Reduzida

Os processos numéricos em geral são complexos, mas em graus diferentes. Portanto, é de interesse do Usuário que esta complexidade seja o quanto menor possível, dentro das possibilidades. Um modelo muito complexo terá poucos Usuários habilitados, comprometendo o seu uso e isto, deve ser evitado, se houver uma alternativa disponível.

- Análises Direcionadas: item Interatividade

Muito comum em processos numéricos tipicamente científicos, a possibilidade de interagir com o modelo de modo a extrair parcialmente os resultados ou, interferir em determinado ponto da simulação.

Para avaliar cada alternativa de processo numérico, adotou-se um critério de pontuação conforme a tabela seguinte.

Tabela 5.4 4 Critério de Pontuação

		Pontuação		
		10	5	0
Desempenho em Simulações	Tempo de Resposta	até 5 min	até 1 h	acima 1 h
Compreensão do Modelo	Complexidade reduzida	20 h	100 h	acima 100 h
Viabilidade de Implantação	Custo de Desenvolvimento	Baixo	Médio	Alto
Exatidão nos Cálculos	Conformidade	Sol ótima	Sol boa	Sol ruim
Viabilidade da Infra-Estrutura	Hardware Econômico	PC	Servidor	Acima Servidor
Viabilidade no Desenvolvimento	Fácil Implementação	50 h	200 h	Acima 200 h

Observe-se que apenas os requisitos que dependem do processo numérico foram considerados na pontuação. Os valores indicados foram estabelecidos em função de estudos elaborados sobre os diversos métodos de solução de problemas de otimização, relatados nos capítulos anteriores.

De acordo com a conceituação do modelo MC-HSQ descrito anteriormente, a matriz de análise tem à esquerda a lista de requisitos, na parte superior os processos numéricos considerados e na parte inferior, a infra-estrutura requerida para executar o processamento dos modelos, atendendo aos requisitos.

Os métodos numéricos constantes na tabela 5.4.5 seguinte foram objetos de pesquisa para solução do problema de otimização da geração de energia elétrica. As duas primeiras alternativas “*Branch and Bound*” e “*Programação Dinâmica*” buscam a solução ótima e consomem tempos excessivos, tornando inviável a sua utilização. No entanto, métodos combinados podem minimizar o tempo de simulação, evitando a pesquisa completa das combinações. A *Meta Heurística* que fornece resultados rápidos para a função-objetivo, pode ser utilizada antes dos métodos de busca citados, como valor inicial, reduzindo o espaço de soluções.

Tabela 5.4 5 Aplicação do Modelo MC-HSQ

MC-HSQ : Modelo Conceitual para solução de Problemas Complexos						
Fóco na Qualidade		Software				
Especificação de Requisitos		Métodos que permitem solução do problema				
Objetivo	Item requerido	Branch and Bound	Programação Dinâmica	Meta Heurísticas	Pesquisa Operacional	Multi Agentes
Expansão do Modelo	Modularidade					
Manutenção facilitada	Manutenabilidade					
Mecanismos de Verificação	Rastreabilidade					
Análises Direcionadas	Interatividade					
Operação facilitada	Usabilidade					
Operacionalização	Implantação					
Desempenho em Simulações	Tempo de Resposta	0	0	8	8	8
Compreensão do Modelo	Complexidade reduzida	8	0	8	8	8
Viabilidade de Implantação	Custo de Desenvolvimento	6	0	9	8	8
Exatidão nos Cálculos	Conformidade	10	10	6	10	8
Viabilidade da Infra-Estrutura	Hardware Econômico	0	0	10	8	8
Viabilidade no Desenvolvimento	Fácil Implementação	10	0	8	9	9
Somatória->		34	10	49	51	49
		Alta	Alta	Normal	Normal	Normal
Legenda		Capacidade de Processamento exigida				
Atende ao requisito requerido com suficiência	Cor					
Atende ao requisito requerido parcialmente	A					
Não atende ao requisito requerido	B					
		C				
		Hardware				

De acordo com a matriz montada, os três métodos da direita: Meta Heurísticas, Pesquisa Operacional e Multi Agentes são mais indicados para serem utilizados, em função de suas pontuações. Pela somatória de pontos, a *Pesquisa Operacional* é a mais alta, mas todas os três foram avaliados como equivalentes, em termos de potencial para solução do problema. A combinação de métodos também fornece boas alternativas, mas a combinação *Multi Agentes + Pesquisa Operacional* se ajusta melhor aos propósitos da tese, conforme explicado nos tópicos abaixo:

- As *Meta-Heurísticas* podem ser configuradas de forma híbrida combinando diversos métodos heurísticos. Pode também ser utilizada como pré-processamento para encontrar rapidamente uma solução boa e passar o comando para um método *Branch & Bound* ou *Programação Dinâmica*, diminuindo significativamente o espaço de

soluções. No entanto, estes dois métodos são computacionalmente onerosos e, portanto inviáveis de acordo com a tabela 5.4-5.

As *Meta-Heurísticas* possuem uma estratégia de solução e objetivos diferentes de *Multi-Agentes* e *Pesquisa Operacional*, o que dificulta a montagem de um sistema híbrido desta forma;

- O problema de otimização requer processamento ostensivo e, portanto, é um fator importante, considerar a possibilidade de distribuir o processamento em várias unidades, onde o processo *Multi Agentes* se adapta com facilidade;
- A linha de pesquisa adotada faz parte do âmbito de *Objetos Distribuídos*.

O modelo híbrido adotado constitui-se de uma combinação de métodos utilizando um *Sistema Multi Agentes* que são objetos de software, passíveis de serem instanciados em unidades de processamentos diferentes, comunicando-se através de uma rede. Com isto, o potencial de processamento pode ser aumentado em muitas vezes, o que é extremamente importante para atender à demanda de processamento.

A otimização se resolve pela *Pesquisa Operacional*, muito difundida na atualidade, tendo diversas versões de bibliotecas para serem utilizados nos modelos.

Como resultado conclusivo das análises efetuadas através deste modelo, foi concebida uma arquitetura utilizando *Sistema Multi Agentes* e *Pesquisa Operacional* como processos numéricos, formando um modelo híbrido operando em um ambiente de processamento distribuído. A solução adotada para implementação do protótipo, transforma o grupo de geração de energia em uma sociedade composta por Agentes que tem diversas funções, e interagem entre si para resolver um problema comum que é o atendimento à demanda de energia elétrica.

5.5. Arquitetura do Sistema

No âmbito da Engenharia de Computação e Sistemas Digitais, a área de automação desenvolve e implanta projetos de sistemas de automação, integrando os recursos computacionais compostos de equipamentos de automação e artefatos de software e redes de comunicação. Dentre as linhas de pesquisa deste segmento, há uma vertente referente ao desenvolvimento de aplicações de objetos distribuídos, constituindo os sistemas distribuídos. Estes sistemas caracterizam-se como um conjunto de serviços, que estão alojados em computadores diferentes, conectados através de uma rede. Decorre disto, que os computadores podem ser de diferentes tecnologias e estarem configurados para operar em várias plataformas e por isso, a comunicação entre os componentes de software distribuídos se efetua através de interfaces de comunicação, que independam das tecnologias de hardware e software em uso ou, através de *middleware* baseado em protocolo conhecido.

A arquitetura do sistema concebida para este trabalho, é composta por componentes de software autônomos, representando os Agentes que se comunicam através de um protocolo de mensagem, executando cada qual seus métodos, em função de papéis assumidos.

O ambiente é constituído por unidades de processamentos remotas acessadas pelo TCP/IP. A ferramenta utilizada é o *JADE – Java Agent DEvelopment Framework*, totalmente desenvolvida em Java, utilizando o padrão *FIPA* para comunicação entre os Agentes. O controle pode ser efetuado através de uma janela remota, sendo possível re-configurar os Agentes, movendo de uma máquina para outra. Para seu funcionamento, *JADE* requer a versão 1.4 do Java. *JADE* é um software livre e é distribuído pela *Telecom Italia*, detentora dos direitos autorais. A última versão do *JADE* é a 3.5, liberada em 25 de Junho de 2007.

JADE opera em rede de três maneiras: *P2P* puro; *P2P* híbrido ou *Client/Server*.

P2P significa *Peer to Peer* e representa uma conexão entre duas unidades de computadores ligados através de uma rede. Uma rede *P2P* pura indica uma rede onde existem apenas conexões entre computadores. Uma rede *Client/Server* representa uma rede onde existe um computador central que se comunica com diversos outros computadores, em formato de estrela. As redes *P2P* híbridas são aquelas formadas por conexões *P2P* e *Client Server*. Para esta tese, a configuração utilizada é a *P2P* híbrida uma vez que há necessidade de um Agente se comunicar com outro Agente que não seja o servidor.

Figura 5.5-1 Rede P2P Híbrida

JADE utiliza o conceito de *Plataformas*, vide figura 5.5-2, para identificar um ambiente, coordenado por um Agente Servidor. Debaixo das plataformas, são criados os *Containers* onde os Agentes criados ficam alojados. Uma plataforma possui um container denominado “*Principal*” e um ou mais *containers* secundários. O container principal possui dois Agentes nativos denominados de *DF* – *Directory Facilitator* e *AMS* – *Agent Management System*.

Figura 5.5-2 Arquitetura JADE

O Agente AMS tem autoridade total no âmbito da plataforma e é responsável pelo serviço de atribuição de nomes, para garantir que um Agente de uma plataforma tenha um nome único.

O Agente DF tem a atribuição de executar o serviço de “páginas amarelas” através do qual, qualquer Agente pode encontrar outro Agente e oferecer seus serviços.

Um container pode ter mais de um Agente, que ao ser criado deve registrar-se junto ao AMS no container principal da plataforma. Os Agentes podem se comunicar com Agentes de qualquer plataforma, conforme suas necessidades.

5.6. DESCRIÇÃO DO SISTEMA DE GERAÇÃO DE ENERGIA

Para a simulação do sistema global, a rede de transmissão ilustrada no capítulo 2, figura 2.1 é representada de forma simplificada, criando-se nós fictícios e agrupando-as em sub sistemas, conforme pode ser visto na figura 5.6-1 “Diagrama de Intercâmbio - Fonte CESP”.

Figura 5.6-1 Diagrama de Intercâmbio - Fonte CESP

Os nós N, NE, SE/CO e S correspondem respectivamente às concentrações dos sistemas Norte, Nordeste, Sudeste/Centro Oeste e Sul. Várias usinas de geração estão em cascata em um mesmo curso de água havendo, portanto, uma reutilização da água para geração de energia. Cada usina possui um custo de geração e uma potência instalada, além de possuir regras específicas de operação, envolvendo restrições de montante, jusante,

navegação etc. Além disso, as usinas geradoras apresentam múltiplos usos tais como: regularização da vazão de jusante, restrições de navegação, restrições específicas (por ex. nível máximo de jusante para não inundar algum empreendimento) etc. sendo necessário considerar as prioridades de geração, em conjunto com estes objetivos. Cada usina interliga-se ao sistema, através da rede de transmissão, limitada pela sua capacidade, apresentando um custo inerente, além de outros parâmetros como perdas por transmissão.

A energia produzida por uma usina hidroelétrica i durante um intervalo de tempo t é obtida a partir da Energia Potencial E_{pot} da massa de água armazenada no reservatório. E_{pot} depende da diferença de nível de água no reservatório (montante) e o nível de jusante. O nível de montante é função não linear do volume de água armazenada no reservatório e o nível de jusante, da mesma forma, é função não linear da vazão defluente. A energia produzida depende também do rendimento da turbina e do grupo gerador.

$$E_i = E_{pot} * \eta_T * \eta_G \quad \dots (3)$$

onde:

E_i é a energia produzida na usina i durante um período de tempo t ;

E_{pot} é a energia potencial obtida através do produto da massa de água multiplicado pela aceleração da gravidade e pela diferença de nível:

$$E_{pot} = m_a * 9.8 * \Delta H \quad \dots (4)$$

onde: m_a : massa de água; ΔH : queda; 9.8: aceleração da gravidade;

η_T é o rendimento da Turbina e

η_G é o rendimento do Gerador.

A potência P_i da usina i é o valor da energia para uma unidade de tempo e a massa de água para a unidade de tempo é a vazão q_i

Portanto, considerando-se 10^{-3} o peso específico da água, tem-se:

$$P_i = 9.8 * 10^{-3} * q_i * \Delta H_i * \eta T_i * \eta G_i \quad \dots (5)$$

isolando q_i obtém-se:

$$P_i = \partial_i * q_i \quad \dots (6)$$

onde ∂_i é denominado de produtibilidade da usina i .

Foi adotada para o desenvolvimento do protótipo, a fórmula de cálculo acima (6) de transformação da vazão em potência utilizando a produtibilidade de cada usina. Isto significa, supor a altura de queda como constante que, em um modelo real, deveria considerar a flutuação da vazão.

Portanto, a quantidade de geração de energia em uma usina hidroelétrica depende da água nela disponível, que é alimentada por meio das precipitações diretas e pelo lençol freático. As precipitações quando ocorrem na natureza, percorrem um ciclo completo denominado de ciclo hidrológico. Da quantidade de água que cai sobre o solo, uma parte é evaporada; outra armazenada em superfícies impermeáveis; outra porção escorre pela superfície até o curso de água; outra parte infiltra alimentando o depósito de água superior e por fim, uma parte da água precipitada alimenta o depósito de água inferior. Este último é o que alimenta o curso de água na estiagem. A água disponível em uma usina depende, portanto, da precipitação sobre a área de influência, ou seja, da quantidade de água que precipita e escoar em direção ao curso d'água, que por sua vez, caminha para o reservatório onde fica acumulado (se houver capacidade para acumulação) e disponível, para ser utilizado na geração de energia. Se houver uma usina à montante, a afluência é constituída de uma contribuição da área entre as usinas que se

denomina de contribuição lateral, juntamente com a descarga da usina de montante, formando o volume de água disponível.

A quantificação de geração de energia nas usinas tem diversos interesses, conforme a finalidade da área de atuação. Para a operação diária das usinas, há a necessidade de saber quanto se deve gerar ao longo do dia, para que a demanda seja atendida, em conformidade com as restrições de transmissão e o suplemento, proveniente de outras fontes de energia térmica. Para manutenção de elementos operativos planejados ou emergenciais, como no caso de saída de uma máquina em uma usina, existe a necessidade de reprogramação da geração.

Para o planejamento são importantes as questões sobre o atendimento à demanda ao longo do tempo e a necessidade de investimento, considerando diversos cenários de crescimento econômico e situações hidrológicas. As atenções se voltam para a construção de novas usinas térmicas, instalação de máquinas adicionais, construção de novas usinas hidroelétricas etc. No ambiente mercadológico em que se encontra o setor elétrico, também é importante a simulação de cenários hidrológicos, já que este é o componente preponderante no problema, que forneçam subsídios para estabelecer o valor correto da energia produzida, possibilitando a concretização de negócios lucrativos.

Sob o ponto de vista computacional, o modelo de otimização da geração de energia elétrica tem por objetivo, simular o comportamento da geração considerando um horizonte de planejamento de n estágios, onde em cada estágio, há a necessidade de determinar a geração em cada unidade do sistema, de modo a atender a demanda, respeitando as restrições operacionais de cada componente de geração, a de um custo mínimo. Isto equivale a determinar em cada estágio, vide Fig 5.6-2, a configuração C_i estabelecendo as gerações em cada usina hidroelétrica e em cada usina térmica.

As interdependências entre as usinas hidroelétricas também devem ser consideradas, quando as unidades de geração estão dispostas em cascata.

Para as hidroelétricas, sem capacidade de reservação, o controle deve ser efetuado pela usina de montante caso exista, o que equivale a transformar as duas unidades em uma só para efeito de simulação.

Figura 5.6-2 Estágios

Para a construção de um protótipo representativo das usinas que compõem o parque gerador, foi criado um modelo composto por usinas hidroelétricas, usinas térmicas e uma rede de transmissão, que em conjunto, simulam um sistema gerador completo. Consideram-se para efeito de implantação do modelo as simplificações abaixo:

- A demanda utilizada é a demanda total, desconsiderando a distribuição por patamar;
- É feita uma abstração parcial de custos, desconsiderando os custos fixos de operação, custos variáveis de produção, custos de partida de uma unidade geradora e outros, uma vez que não são imprescindíveis para a montagem do protótipo;

- Desprezam-se os rendimentos das turbinas e do grupo gerador;
- Ignoram-se as perdas de transmissão e qualquer outro tipo de perda no sistema;
- O sistema de transmissão é encapsulado em grandes blocos, onde as usinas hidroelétricas e térmicas concentram a energia, em centros de carga que são interligados;
- Considera-se disponível o volume útil para efeito de simulação, desconsiderando as restrições.

Os dados de entrada são: vazões, demandas, capacidades de geração e de transmissão e restrições operativas das usinas. O objetivo almejado é o planejamento de longo termo (5 anos) e, portanto, a discretização do tempo adotada é mensal.

A função-objetivo consiste em determinar em cada estágio, a solução de menor custo de geração das térmicas, atendendo à demanda juntamente com a geração das hidroelétricas, e as restrições de transmissão.

Para solução do problema, dividiu-se o mesmo em módulos de processamento consecutivos. Em uma primeira etapa, utilizou-se um modelo *SMA – Sistema Multi Agente* onde cada usina geradora é representada por um Agente, que possui comportamentos e metas bem definidas. O parque gerador e o sistema de transmissão interagem entre si, buscando um objetivo comum que é o atendimento à demanda. Os Agentes Geradores chegam aos seus objetivos, quando conseguem determinar um intervalo de geração ou sua geração máxima. Para isto, efetua uma análise sobre as séries de vazões do histórico, para selecionar a melhor série ocorrida, em função de um critério de aderência estabelecido. No passo seguinte, através da pesquisa operacional, determina-se a configuração de geração, no conjunto de usinas de menor custo, envolvendo todas as variáveis do problema. Os valores determinados pela otimização são devolvidos para os Agentes Geradores que atualizam seus dados, para uma nova iteração. Este

processo é repetido até o fim do período, ressaltando-se que a decisão tomada em um estágio, influi nos estágios seguintes. Têm interesse no final da simulação, os custos obtidos e a verificação da existência de déficits no sistema.

6. IMPLANTAÇÃO DO PROTÓTIPO

A simulação da geração de energia elétrica é um processo conhecido como *multi-estágios* onde a decisão assumida em um estágio i influi no processo de simulação dos estágios seguintes.

Para cada estágio i ao longo do período de simulação, os agentes interagem entre si e fornecem como resultados, os limites máximo e mínimo de geração em que acreditam serem viáveis.

As informações são passadas então para um módulo de pesquisa operacional, que determina as gerações em cada unidade que resulte no custo mínimo, em função das restrições representadas pelas equações. O resultado é repassado para os Agentes que atualizam seus dados, e uma nova iteração é iniciada.

O processo de solução do problema é constituído de duas partes distintas descritas a seguir:

6.1. Definição do SMA – Sistema Multi Agentes.

Um sistema multi agentes é caracterizado por:

$$A = \{ a_1, a_2, \dots, a_n \} \quad \dots (7)$$

onde $a_1, a_2, \dots, a_n$ são agentes representativos dos elementos que intervêm no processo de geração de energia, tais como usinas hidroelétricas, térmicas e sistema transmissor. Os agentes assumem um papel pré-estabelecido e se comportam conforme os procedimentos criados para cada papel. Assim, uma usina que tenha capacidade para reservar água, exercerá dentre outros, um papel específico de regularizar a vazão de jusante. Possui neste caso, um comportamento bem definido como por ex., armazenar a água afluente em excesso para descarregar lentamente ao longo do tempo, para que não ocorra inundação à jusante.

Os agentes possuem a propriedade de interoperabilidade, ou seja, a capacidade de interagir com outros agentes. Em uma configuração de

usinas em cascata, a usina de jusante valoriza a informação de quanto se pretende gerar na usina de montante, representado por outro agente, pois a água descarregada (geração + vertimento) irá alimentar seu reservatório após o tempo de viagem da água, de um reservatório a outro.

O elemento *rede de transmissão* é representado por outro agente que interage com todos aqueles que dele utilizam, para transmitir a energia gerada. Em conjunto, decidem as restrições de transmissão, estabelecendo um intervalo de valores possíveis.

O totalizador de energia é representado por um *agente coordenador* que se comunica com todos os agentes geradores e transmissores, preocupando-se em suprir a demanda em cada estágio.

O SMA possui um objetivo global G e os agentes possuem cada um, objetivos distintos, ou seja:

$$G = \{ g_1, g_2, \dots g_n \} \quad \dots (8)$$

O objetivo global G é satisfeito quando os objetivos individuais de cada agente são satisfeitos.

Tem-se, portanto, em cada estágio, a função de atendimento aos objetivos, denotada por:

$$\forall a_i \forall g_i f(a_i, g_i) = \text{verdadeiro se } g_i = \text{verdadeiro} \quad \dots (9)$$

Os agentes criados são regidos pelos princípios de não benevolência, sinceridade, auto-conhecimento e consistência, conforme citado em (DROGOUL 1993). Assumem papéis distintos e para cada papel, estabelecem-se planos de ação para atingir seus objetivos.

Para atingir seus objetivos, os agentes contam ainda com uma base de conhecimento individual, que poderá ser consultada, para auxiliar no processo de decisão das possibilidades de ocorrência de fatos, com comportamentos típicos. Este conhecimento pode referir-se a uma situação

hidrológica futura ou, um conjunto de possibilidades baseado em fatos ocorridos no passado. Com esta informação, os agentes iniciam um processo de interação trocando informações para se chegar a uma decisão.

Em um SMA, os Agentes são organizados de acordo com uma estrutura bem definida, como a proposta por (BOISSIER 1993) através do *ASIC – An architecture for social and individual control*. Para o protótipo desta tese, a sociedade “Mercado de Energia” está ilustrada na figura 6.1-1 seguinte seguindo a arquitetura sugerida por Boissier, representando os papéis desempenhados pelos agentes.

Os papéis criados são agrupados por objetivos que são:

Grupo GERADOR – constituem os papéis reservados aos geradores de energia utilizando usinas hidroelétricas;

Grupo TERMICO – constituem os papéis reservados aos geradores de energia utilizando usinas térmicas.

Grupo TRANSMISSÃO – constituem os papéis reservados aos agentes encarregados da transmissão de energia elétrica pela rede.

Grupo GERAL – constituem os papéis reservados aos agentes que desempenham atividades de coordenação.

Em seguida estão descritos os diversos papéis e o grupo a que pertencem

Grupo GERADOR:

Este papel é atribuído às usinas hidroelétricas e seu objetivo é estabelecer para cada iteração, um intervalo de geração possível, em função do estado de cada reservatório (neste caso, representado pelo nível do volume em que se encontra), da vazão de contribuição lateral ou afluente, do histórico de vazões ocorrido até o momento e das descargas da usina de montante, se existir.

Figura 6.1-1 Especificação Estrutural ASIC da Sociedade "Mercado de Energia"

Cada agente que desempenha um papel neste grupo representa uma usina hidroelétrica e possui regras gerais, que servem para todos os papéis, e regras específicas, dependentes do tipo de papel a ser assumido.

Os agentes criados assumem um papel que rege o comportamento de uma determinada usina hidroelétrica. Para isso, algumas regras gerais direcionam o comportamento dos agentes:

- a) Os agentes vislumbram sempre chegar, em cada iteração, na faixa de volume compreendida entre o máximo e o mínimo;
- b) As restrições operativas existentes em cada usina, constituem crenças que devem ser obedecidas;
- c) Os agentes possuem o livre arbítrio para tomar decisões e fornecer ao agente Coordenador a melhor informação em que acredita serem viáveis;
- d) É possível inserir no sistema um papel específico com limitações rígidas e inflexíveis, para simular um caso específico. Isto significa diminuir um grau de liberdade;
- e) Para decidir uma situação, o agente deve consultar sua base de conhecimento, procurando identificar situações semelhantes ocorridas no passado e adotá-las como crença;
- f) Os agentes que representam usinas em cascata devem interagir entre si, para chegar a um bom termo sobre as defluências e gerações. A usina de montante influi na afluência da usina de jusante e por isso, as ações entre elas devem ser tomadas em conjunto. Nos casos de dependência entre dois elementos, os agentes repassam ao Agente Coordenador uma função de dependência.

Os papéis projetados para este grupo Gerador são:

PAPEL UFD – Usina Fio D'água: Constitui um papel a ser assumido por um agente representante de uma usina a fio d'água. Este tipo de usina se

caracteriza por não possuir (ou possuir um pequeno volume) volume de acumulação, não tendo, portanto, a capacidade de regularização de vazões. Todo volume de água que entra no reservatório precisa ser descartado para jusante, seja gerando energia através das turbinas ou, soltando água através dos órgãos de descarga. Quando muito, uma usina a fio d'água pode armazenar um pequeno volume por algumas horas, para atender à demanda de pico do sistema. Este tipo de usina apresenta viabilidade, apenas em cursos de água que possuam vazão mínima permanente ou, que exista a sua montante, outro reservatório de regularização que garanta uma vazão perene.

PAPEL URE – Usina com Capacidade de Acumulação: Constitui um papel a ser assumido por um agente representante de uma usina de acumulação. Tem como característica principal, a capacidade de amortecer ondas de cheias, utilizando para isso o volume de espera. Com isto, torna-se possível regularizar a vazão de jusante, e armazenar o volume excedente de água em época de chuvas, para defluir de forma controlada, no período de estiagem.

Os papéis UFD e URE podem estar dispostos de forma isolada ou, como freqüentemente ocorre, em cascata. Neste caso, existe uma forte influência da usina de montante no estado hidráulico da usina de jusante e por isso, as ações devem ser tomadas em conjunto.

Grupo TÉRMICO:

O objetivo dos papéis deste grupo é estabelecer para cada iteração, um intervalo de valores correspondentes às gerações máximas e mínimas que cada agente coloca à disposição para geração. Cada usina térmica possui dados referentes à sua capacidade e custos envolvidos. O objetivo de cada usina é suprir a demanda, sempre que necessário. O agente coordenador se encarrega de determinar qual é a geração desejada em cada usina térmica, que resulte no menor custo.

O papel definido para este grupo é constituído de apenas um tipo qual seja:

PAPEL TER – Usina Térmica: Constitui um papel a ser assumido por um agente representante de uma usina térmica. Possui uma capacidade de geração e um custo associado que é repassado para o agente Coordenador.

Grupo TRANSMISSÃO:

O objetivo dos papéis deste grupo é contabilizar a energia gerada e transferir para os nós através das conexões de transmissão. Possui uma capacidade máxima de transmissão e para definir valores, informa ao agente Coordenador suas limitações.

PAPEL RTR – Rede de Transmissão: Constitui um papel a ser assumido por um agente representante de um trecho da rede de transmissão.

Grupo COORDENADOR:

O objetivo dos papéis deste grupo é coordenar as ações junto aos agentes dos grupos GERADOR, TÉRMICO e TRANSMISSÃO, dando prosseguimento à simulação.

PAPEL COO – Coordenador da Geração: Constitui um papel a ser assumido por um agente que irá controlar todos os agentes e viabilizar a geração de energia necessária em função da rede de usinas hidroelétricas e térmicas.

Faz parte das atribuições deste papel, executar as iterações e em cada uma delas, interagir com os agentes e contabilizar a geração necessária. Em seguida, de posse das possibilidades de geração de cada agente, executa um procedimento de otimização através de pesquisa operacional, onde são impostas todas as restrições e condicionantes do problema, buscando como meta, a minimização do custo da função objetivo.

6.2. Pesquisa Operacional

Em cada iteração, seja $LimInf(u_n)$ e $LimSup(u_n)$ os limites inferior e superior de geração fornecidos pela usina u_n . A primeira restrição a ser considerada,

diz respeito à capacidade de geração de cada usina. Seja G_j a geração atribuída à usina j . Então, tem-se:

$$LimInf(u_j) \leq G_j \leq LimSup(u_j) \quad \text{com } j = 1, \dots, \text{num usinas} \quad \dots (10)$$

Considerando-se t usinas térmicas em operação e h usinas hidroelétricas em operação, seja T_i a geração na usina térmica i de custo C_i e G_j a geração da usina hidroelétrica j e D a demanda do sistema.

Então,

$$\sum_{i=1}^t T_i + \sum_{j=1}^h G_j \leq D \quad \dots (11)$$

Em cada nó de transmissão i , o módulo da somatória algébrica de carga C_i que chega (ou sai) deste nó deve ser menor que a sua capacidade de transmissão CT_k no trecho seguinte. Então:

$$\sum_{i=1}^m C_i \leq CT_k \quad \text{onde } i = 1, \dots, \text{num carga afluente ao nó } k \quad \dots (12)$$

Para as usinas dependentes da operação de uma usina de jusante, existe uma função que relaciona a afluência com a defluência de montante, ou seja,

$$afluência = Q_{mon \tan te} * CA_k \quad \dots (13)$$

onde CA_k é o coeficiente de abatimento do pico de onda da usina k

A afluência às usinas que possuem usina à montante se calcula como a somatória da vazão lateral mais a defluência (descarga + vazão turbinada) de montante considerando o abatimento do pico ou seja,

$$afluência = Q_{lateral} + Q_{mon\ tan\ te} * CA_k \quad \dots (14)$$

A função-objetivo é dada pela equação:

$$FO = (\min) \left(\sum_{i=1}^t T_i * CT_i + \sum_{j=1}^s G_j * CG_j \right) \quad \dots (15)$$

onde

T_i e CT_i são a geração e o custo da térmica de índice i ;

G_j e CG_j são a geração e o custo da hidroelétrica de índice j

Para a montagem de um protótipo genérico, algumas hipóteses foram adotadas, sem perda de generalidade do modelo. A discretização em um modelo de planejamento pode variar em função do horizonte adotado que pode ser de curtíssimo, curto ou médio prazos. Para curtíssimo prazo a discretização normalmente é diária, podendo até descer ao nível horário desde que os dados estejam disponíveis com esta fragmentação. Para planejamento de curto prazo, a discretização normalmente é semanal e para médio prazo a discretização é mensal. No protótipo a ser implementado, está fixado o horizonte como sendo de 5 anos e a discretização, como sendo mensal para efeito de exemplificação. A discretização é uma unidade de tempo genérica e neste caso, adotar como tempo de traslado da onda de um reservatório para outro, como sendo igual à unidade de tempo de um estágio, não compromete a implementação. Com isto, uma onda de cheia descarregada na usina de montante, estará disponível na usina de jusante no estágio seguinte. O abatimento foi fixado em 60% para efeito de simulação e isto também não invalida o protótipo porque, em um caso real basta substituir este valor, pelo valor correto, que depende principalmente do comprimento total do trecho de canal entre um reservatório e o seguinte.

Na figura 6.2-1 “Esquema Genérico de um Sistema Gerador”, está esquematizado um protótipo genérico formado por um conjunto de

reservatórios, térmicas e linha de transmissão, representando os elementos do grupo GERADOR, TÉRMICO E TRANSMISSÃO.

A usina UFD_1 é uma usina a fio d'água, as usinas URE_1 e URE_2 e URE_3 são as usinas com capacidade de acumulação, sendo que UFD_1 tem URE_1 como usina de montante e URE_3 tem URE_2 e UFD_1 como usinas de montante.

As térmicas envolvidas são as representadas por $TER_{1..3}$, além das térmicas virtuais $VIR_{1..3}$ de capacidade ilimitada e custos elevados, inseridas nos nós para atender a déficits no sistema. As demandas nos centros de carga estão identificadas por $D_{1..3}$.

As afluências naturais aos reservatórios e as contribuições laterais estão denominadas pela letra Q .

Para calcular as afluências totais em cada reservatório deve-se somar, se existir, o reservatório de montante, a afluência lateral e as defluências de montante, compostas pela vazão vertida e turbinada. Observar que a descarga de montante deve ser considerada no estágio seguinte, devido ao tempo de traslado da onda, fixado com duração de 1 estágio, por hipótese, e com um abatimento de 60% da onda.

Figura 6.2-1 Esquema Genérico de um Sistema Gerador

6.3. IMPLEMENTAÇÃO DO PROTÓTIPO

Para implementação do protótipo foram adotadas as seguintes considerações:

Hipóteses:

- O horizonte de planejamento é de $HP = 5$ anos com discretização mensal;
- O algoritmo que efetua a busca na base de dados do histórico de vazões afluentes aos reservatórios, segue o procedimento “Inferência da vazão afluente”, detalhado adiante e utiliza os coeficientes c_0 , c_1 , c_2 e c_3 , para a determinação da vazão do estágio seguinte, em função da vazão corrente, do mês $i-1$, do mês $i-2$ e da soma das vazões até então ocorridas;
- Parte dos dados de entrada foi extraída das usinas de Miranda, Capim Branco 2, Furnas e Itumbiara (Tabela 6.3-1), todos da Bacia do Rio Paranaíba, por apresentarem similaridade com o diagrama unifilar do protótipo. As demandas são fictícias e tem apenas a finalidade de validar a execução do protótipo. Para as térmicas, foram utilizados os dados de Cuiabá, Termo Rio e Norte Fluminense;
- As afluições e os demais dados utilizados para cada um dos elementos do protótipo estão ilustrados na tabela 6.3-1;

Tabela 6.3-1 Componentes do Protótipo

Parâmetro	Vmax	Vmin	Vol Útil	QT max	Pot Inst.	Produtibilidade	Posto	Nome do Posto
Usina	hm3	hm3	hm3	m3/s	MW			
Miranda	1120	974	146	669	408	0,6101	206	Miranda
Capim Branco2	879	879	0	174	70	0,4018	28	Capim Branco 2
Furnas	15000	5733	9267	1755	1312	0,7475	6+24	Furnas e Emborcação
Itumbiara	17027	4573	12454	3532	2280	0,6454	207+31	Capim Branco 1 e Itumbiara

- Para a simulação do protótipo, são utilizados o volume máximo e o volume mínimo, no cálculo do volume útil que é utilizado no balanço hídrico;
- Considera-se que o tempo de viagem da descarga de uma usina consome um tempo equivalente a um estágio, para atingir a usina seguinte;
- Considera-se um abatimento de 40% do valor da descarga de montante quando atinge a usina de jusante.

Dados de entrada:

Geral:

- Demanda nos nós em MW;

Grupo GERADOR:

- Histórico de Vazões Afluentes e Contribuições Laterais em m^3/s ;
- Produtibilidade das usinas em $\text{MW} / \text{m}^3/\text{s}$;
- Volume inicial dos reservatórios em hm^3 ;
- Volumes máximos e mínimos dos reservatórios em hm^3 ;
- Vazão turbinada máxima das usinas em m^3/s ;

Grupo TÉRMICO:

- Custo de geração em MWh;
- Capacidade máxima de geração em MW;

Grupo TRANSMISSÃO:

- Capacidade de Transmissão em MW;

Inferência da vazão afluyente: Cada agente do grupo GERADOR possui uma base de dados de afluições históricas utilizada na determinação de um conjunto de parâmetros de simulação.

O algoritmo que efetua a busca da melhor série na base de dados do histórico de vazões segue o procedimento abaixo:

- Dada uma série de vazões histórica mensal de $nAnos$ de um posto qualquer, a sua base histórica é representada por um conjunto H de n sub series de 5 anos com início i , variando de

$$i = 0 \text{ até } (i < nAnos - 5). \quad \dots (16)$$

- Para um estágio pos de uma simulação, o algoritmo irá analisar no sub conjunto H , qual a melhor série s que apresente o menor valor para :

$$((q_{i-2} - qhist_{i-2}^s) * c_3 + (q_{i-1} - qhist_{i-1}^s) * c_2 + (q_i - qhist_i^s) * c_1 + St * c_0) / 100 \dots (17)$$

onde:

- q e $qhist$ são vazões correspondentes às séries: simulada e histórica e na fórmula, seus índices indicam os valores referentes ao mês corrente, ao mês $i-1$ e ao mês $i-2$.
- St é a somatória de vazões históricas conhecidas até o estágio pos .

$$St = \sum_{i=0}^{pos} qhist_i \quad \dots (18)$$

- c_0, c_1, c_2, c_3 são parâmetros obtidos através de um processamento sobre a massa de dados, cuja somatória é 100. Representam fatores de ponderação da influência da soma das vazões e, dos valores de vazões antecessores ao valor corrente, na determinação do valor futuro. A somatória de vazões St foi introduzida entendendo-se que o volume de água

contabilizado até o momento, forneceria de certa forma, uma idéia de saturação do solo, o que traz influência na estimativa da vazão futura.

Foram efetuadas diversas simulações utilizando os procedimentos acima detalhados, sobre a massa de dados de vazões históricas, cujos resultados permitiram extrair as seguintes conclusões:

Conclusão 1 – Em uma análise efetuada somente visando estabelecer a melhor aderência da curva real com as curvas das séries históricas, verificou-se que, os meses $i-1$ e $i-2$, têm muito pouca influência na determinação do valor mínimo obtido pela expressão (17). Isto significa adotar $c_2=0$ e $c_3=0$;

Conclusão 2 – Em uma outra análise, incluindo nos cálculos a soma das vazões ocorridas, verificou-se que a qualidade da estimativa apresenta um decréscimo;

Conclusão 3 – Baseado somente no histórico de vazões ocorridas, conclui-se que não existem meios de estabelecer com precisão, a melhor série. Em outras palavras, a série de vazões futuras é incerta;

Conclusão Final – Para esta tese, optou-se por uma configuração intermediária para a estimativa da vazão futura do estágio seguinte, o que equivale a adotar a configuração:

$$c_0=0; c_1=90; c_2=10; c_3=0.$$

A tabela 6;3-2 seguinte contém o extrato das simulações efetuadas.

Tabela 6.3-2 Casos de Simulação da Vazão $i+1$

Caso	Coeficientes				Erro vazao			Erro v Futura		
	c0	c1	c2	c3	<2%	<5%	<10%	<2%	<5%	<10%
4	0	100	0	0	91,38%	98,28%	100,00%	1,72%	10,34%	22,41%
3	0	90	10	0	75,86%	94,83%	100,00%	1,72%	8,62%	27,59%
1	0	80	20	0	63,79%	89,66%	100,00%	1,72%	13,79%	25,86%
5	5	95	0	0	29,31%	72,41%	93,10%	3,45%	12,07%	37,93%
2	5	60	20	15	22,41%	56,90%	89,66%	3,45%	8,62%	32,76%

A alternativa selecionada corresponde ao caso 3 onde aproximadamente, em 95% dos casos, a diferença entre a vazão corrente e a vazão da série selecionada é inferior a 5%. Nesta alternativa, apenas em 8% dos casos, a diferença entre os totais da série histórica e a série escolhida para o período futuro é menor do que 5%.

Balanco no centro de carga:

Em cada centro de carga do sistema, a somatória das energias é nula para que o balanço energético seja verificado.

Figura 6.3-1 Balanço no nó i

Desta forma, da figura 6.3-1 se deduz:

$$\sum_{i=1}^n E_i = 0 \text{ ou seja, } T_i + G_i + T'_i = C_i + D_i \quad \dots (19)$$

onde D_i é a demanda no nó i ; C_i é a quantidade transferida para o nó seguinte; T_i e G_i são as gerações hidroelétrica e térmica respectivamente e T'_i é a térmica virtual de capacidade ilimitada e custo elevado, tendo a função de identificar o déficit de energia no nó i . Observar que a quantidade de intercâmbio C_i pode ser (somente esta grandeza neste nó) positiva ou negativa dependendo de, a direção do fluxo estar saindo ou entrando no nó. O nó seguinte deve conter a mesma componente obedecendo inclusive o sinal, conseqüentemente a direção. Isto significa que este valor irá compor o balanço energético do nó seguinte.

Cálculo do Balanço Hídrico e determinação da Disponibilidade Hídrica:

Seja V_i o volume inicial no estágio i , Q_i a vazão estimada para o estágio i , $Q_{mont_{i-1}}$ a vazão da usina de montante caso exista. A afluência é calculada em função da inferência Q_i baseada no histórico de vazões, somada com a defluência Q_{mont_i} da usina de montante, se existir, defasada de um estágio e com o pico abatido para 60% do valor original, conforme as hipóteses adotadas.

A disponibilidade energética é calculada utilizando-se o volume útil total ou seja,

$$V_{util} = V' - V_{min} \quad \dots (20)$$

O roteiro de cálculo é efetuado na seguinte ordem:

- Partindo-se do volume final do estágio anterior V_{i-1} , a afluência total ao reservatório $Q_i + q_{Mont_{i-1}} * 0.6$ leva a um volume V' ;
- $V_{disp} = V' - V_{min}$ representa o volume disponível para geração o que corresponde a uma vazão Q_{disp} . No entanto, esta vazão disponível

deve ser menor do que a capacidade máxima de geração Q_{tur} da usina;

$$\text{Afluência} := Q_i + qMont_{i-1} * 0.6$$

Figura 6.3-2 Balanço Hídrico

- O volume V'' corrigido é calculado por: $V'' = V' - V_{disp}$. Este deve ser confrontado com o V_{max} . Se for maior, há a necessidade de reduzir o volume até o máximo, através da descarga da usina Q_{desc} ;
- Q_{disp} é submetido a um processo de otimização e o valor da geração é devolvido pelo algoritmo de otimização. Os volumes são recalculados segundo o novo valor de V_{disp} , chegando-se então a:

$$V_{final} = V_i - V_{disp} - V_{desc} \quad \dots (21)$$

onde V_i é o volume inicial do estágio, V_{final} é o volume final, V_{disp} corresponde ao volume de geração e V_{desc} se existir, corresponde ao volume descarregado.

A transformação de vazão em volume se efetua através do coeficiente

$$nSeg / hm^3 \quad \dots (22)$$

onde $nSeg = 86400 * 30$ é o número de segundos de um mês e $hm^3 = 10^6$ é o transformador de unidade de metros para hectômetro.

A transformação de vazão em potência se efetua através da produtibilidade conforme explicado em capítulo anterior.

$$P = \partial * Q \quad \dots (23)$$

onde ∂ é a produtibilidade da usina, sendo expresso em $(MW / (m^3/s))$

Metas dos Agentes:

Um SMA – Sistema Multi Agentes possui um objetivo global G , que neste caso, corresponde a atender a demanda de energia minimizando o custo de geração das térmicas, e os agentes possuem cada um, objetivos distintos que dependem do seu papel.

O objetivo global G é satisfeito quando os objetivos individuais de cada agente sejam satisfeitos. Os objetivos individuais para cada papel são:

PAPEL COO – Coordenador da Geração: Este papel tem as seguintes metas que são implementadas como métodos de uma classe:

- Carregar dados;
- Instanciar os Agentes dos grupos GERADOR, TRANSMISSÃO e TÉRMICO;

Para cada estágio:

- Solicitar intervalo de geração;
- Executar processo de otimização;
- Informar geração adotada em cada Usina.
- Elaborar relatório com os resultados.
- Informar fim de atividade para os Agentes;
- Encerrar atividade.

PAPEL UFD/URE: Constitui um papel a ser assumido por uma usina a fio d'água ou de acumulação. Os métodos são:

- Carregar dados próprios;
- Interagir com o Agente Coordenador recebendo comandos para efetuar cálculos e devolver intervalo de geração;
- Interagir com usina de montante se existir, recebendo a descarga média esperada;
- Efetuar o Balanço Hídrico e determinar o intervalo de geração, respeitando todas as suas restrições operacionais;
- Interagir com o Agente Coordenador e notificar o intervalo
- Ficar em estado de espera, aguardando solicitação de ações do Agente Coordenador;
- Encerrar atividade.

PAPEL TER: Constitui um papel a ser assumido por uma usina térmica. Os métodos são:

- Carregar dados;
- Interagir com o Agente Coordenador recebendo comandos para efetuar cálculos e devolver intervalo de geração;
- Interagir com o Agente Coordenador e notificar o intervalo de geração possível;
- Ficar em estado de espera, aguardando solicitação de ações do Agente Coordenador;
- Encerrar atividade.

PAPEL RTR: Constitui um papel a ser assumido por um nó de transmissão. Os métodos são:

- Carregar dados;
- Interagir com Agentes do Grupo GERADOR informando sua capacidade de transmissão e a demanda no nó;
- Ficar em estado de espera, aguardando solicitação de ações do Agente Coordenador;
- Encerrar atividade;

As interações entre Agentes ocorrem através de mensagens baseadas no protocolo FIPA. As seguintes mensagens compõem exemplos de interações entre Agentes:

Agente COO requisitando intervalo de Geração:

Conjunto de mensagens em sequência correspondentes a um Usuário “COO” requisitando intervalos de geração esperada no mercado de energia:

```
ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
msg.setContent("ASKING");
msg.setReplyWith(Integer.toString(mesCorrente));
msg.addReceiver(new AID("URE1", AID.ISLOCALNAME));
msg.addReceiver(new AID("URE2", AID.ISLOCALNAME));
msg.addReceiver(new AID("URE3", AID.ISLOCALNAME));
msg.addReceiver(new AID("UFD1", AID.ISLOCALNAME));
msg.addReceiver(new AID("RTR1", AID.ISLOCALNAME));
msg.addReceiver(new AID("RTR2", AID.ISLOCALNAME));
msg.addReceiver(new AID("RTR3", AID.ISLOCALNAME));
send(msg);
```

onde:

ASKING representa o código de mensagem;

INFORM representa o tipo de mensagem, neste caso assíncrono;

mesCorrente representa o estágio da simulação;

URE1 representa a identificação do agente;

Agente gerador requisitando descarga média da usina de montante:

Conjunto de mensagens em sequência correspondentes a um Usuário URE3 requisitando a descarga do estágio anterior para UFD1:


```
msg = new ACLMessage(ACLMessage.INFORM);  
msg.setContent("ASKMONT_UFD1");  
bufao = Format.sprintf("%10d", p.add(indice));  
msg.setReplyWith(bufao);  
msg.addReceiver(new AID("UFD1", AID.ISLOCALNAME));  
send(msg);
```

onde:

ASKMONT_UFD1 representa o código de mensagem;

INFORM representa o tipo de mensagem, neste caso assíncrono;

indice representa o estágio da simulação;

UFD1 representa a identificação do agente para o qual está sendo solicitada a informação;

Resposta do Agente UFD1 informando o valor calculado:

```
msg = new ACLMessage(ACLMessage.INFORM);  
msg.setContent("VAZMONT_UFD1");  
bufao = Format.sprintf("%10.2f", p.add(aux1));  
msg.setReplyWith(bufao);  
msg.addReceiver(new AID("URE3", AID.ISLOCALNAME));  
send(msg);
```

onde:

VAZMONT_UFD1 representa o código de mensagem;

INFORM representa o tipo de mensagem, neste caso assíncrono;

aux1 representa o valor solicitado;

URE3 representa a identificação do agente para o qual está sendo enviada a informação;

Inicialmente, o protótipo foi implementado em uma planilha Excel, utilizando a função interna “solver”, simulando apenas uma iteração, variando os valores e verificando o seu comportamento. O resultado obtido mostrou-se adequado quanto à conformidade, e a questão mais importante, o tempo de resposta para este protótipo, constituído de 12 variáveis e um conjunto de inequações formando uma matriz de 27 x 12, consumiu alguns milissegundos. Para efeito de comparação, o mesmo problema foi codificado na linguagem

Java, utilizando a API “Lindo” versão 5.0 da *Lindo System*¹⁹ apresentando os mesmos resultados para a função-objetivo. Alguns testes mostraram resultados diferentes de configuração da solução, mas com o mesmo valor da função-objetivo.

Em um ensaio seguinte, o protótipo foi expandido para mais de uma centena de variáveis, utilizando o Solver da planilha Excel, com o objetivo de verificar se o tempo de resposta na busca da solução seria excessivo. A simulação envolvendo o processo de otimização consumiu alguns segundos, o que demonstrou não haver dificuldade em modelar sistemas maiores, da ordem de grandeza do parque gerador nacional.

Para consolidar o protótipo, o mesmo foi implementado em um desktop com processador dual com 2 GBytes de memória RAM, utilizando a linguagem Java através da plataforma de desenvolvimento *Eclipse JDK* versão 3.2.1 da *Apache Software Foundation*²⁰. Para o controle e monitoramento dos Agentes, foi utilizado o *framework JADE* descrito no capítulo 5.5 intitulado “Arquitetura do Sistema”. Para a pesquisa operacional foi utilizada a API “Lindo” versão 5.0.

Foram codificadas 8 classes, uma para cada agente integrante da sociedade, com diversos métodos perfazendo um total de 3546 linhas de código.

Para o caso de implementação do protótipo em múltiplas CPUs, o *framework JADE* permite a criação de diversas plataformas, operando em unidades de processamento distintas. Para monitorar as diversas plataformas, o *framework* utiliza dois agentes. O *ams – Agent Management System* para coordenar a criação de agentes e o *df – Directory Facilitator* para gerenciar a comunicação entre os agentes permitindo a comunicação inter-plataformas.

¹⁹ www.lindo.com/.

²⁰ <http://www.apache.org/>

A medida do desempenho em um sistema distribuído utilizando agentes, pode ser avaliado de forma análoga ao processamento paralelo, através do parâmetro *Speed UP*.

Seja T_{COO} o tempo total de processamento do Agente Coordenador e T_{AGi} o tempo de processamento do Agente i , sendo executado em unidades de processamento distintas de mesmo tipo.

Então, o desempenho do processamento distribuído poderia ser avaliado como a divisão entre: o tempo total que uma unidade de processamento levaria para executar as atividades do Agente Coordenador de todos os Agentes participantes, e o máximo dos tempos de processamento de todas as unidades de processamento, incluindo o Agente Coordenador.

$$\frac{T_{COO} + \sum_{i=1}^n T_{AGi}}{\max(T_{COO}, \max(T_{AGi}))} \quad \dots (24)$$

Os agentes foram codificados na linguagem Java e cada agente foi construído como uma classe. O agente *cooAgent* é criado junto com os demais, tendo como atribuição executar a simulação, controlando as ações dos agentes e efetuando o processamento da otimização, através da *API* do *Lindo*

O funcionamento do modelo é assíncrono, ou seja, o agente coordenador envia mensagens para os agentes participantes e espera a resposta de todos. Somente após o recebimento das respostas é que o processo segue adiante.

A figura 6.3-3 representa a janela de trabalho do framework Jade. Na parte da esquerda, estão listados todos os agentes criados.

A figura 6.3-4 representa a janela de desenvolvimento do Eclipse, onde podem ser visualizadas as classes correspondentes aos Agentes.

Figura 6.3-3 Janela dos Agentes

Figura 6.3-4 Plataforma Eclipse

A criação dos agentes é efetuada pelo método *Boot* do *framework* através da linha de comando com a seguinte estrutura:

-gui *URE1*:jade.*ure1Agent*

onde *URE1* é a identificação do agente criado e *ure1Agent* é a classe instanciada.

A criação de agentes utilizando o *framework* é bastante facilitado:

```
AgentContainer container = (AgentContainer)getContainerController();  
t1 = container.createNewAgent(t1AgentName, "pack.Agent", null);  
t1.start();
```

O *framework* permite a troca de mensagens através da janela de controle dos agentes de forma manual, mas para esta tese, todo o processo de mensagens foi efetuado através do código.

De forma resumida, o processo de simulação segue o roteiro abaixo;

1. Inicia o contador;
2. Agente Coordenador envia o estágio corrente através de uma mensagem, para os Agentes Geradores e aguarda até que todos respondam;
 - a. Cada agente participante efetua um processamento e envia ao final o valor da geração disponível;
3. Ao receber todas as mensagens dos Agentes participantes, o Agente Coordenador efetua o processo de otimização, determinando a configuração de geração de menor custo;
 - a. Antes de processar, o agente coordenador efetua um cálculo para verificar se a capacidade energética futura atende à demanda. Em caso negativo, é efetuado um novo cálculo, para atribuir à geração hidráulica um custo de déficit;

4. Após determinar a configuração de geração em cada usina, o Agente Coordenador informa aos Agentes participantes os valores resultantes da otimização;
5. Incremento do contador e teste de fim de simulação. Se fim vá para o passo 7;
6. Se não é o fim da simulação, vá para o passo 2;
7. Encerra a simulação.

Por se tratar de um desenvolvimento de um algoritmo, a exatidão dos procedimentos de cálculos deve ser garantida atendendo ao requisito *conformidade*, e para isto, ao longo do código foram implementados recursos para rastreamento da simulação, focalizando itens específicos. Para cada item importante, foi atribuído um código para que o sistema grave em arquivo, o processamento passo a passo, mostrando a evolução dos cálculos.

Desta forma, tornou-se possível analisar cada item de forma detalhada, verificando a exatidão nas passagens de parâmetros e dos cálculos efetuados, além de verificar também as condições limítrofes.

Foi adotado o seguinte conjunto de códigos e arquivos de saída associados, ou emitidos de forma incondicional, para auxiliar no processo de verificação do algoritmo:

DEBUGGER CODE

```
101 - RASTREAMENTO DO AGENTE COO - ARQUIVO traceCOO.txt
102 - PASSAGEM DE PARÂMETROS DO UFD1 PARA O COO - ARQUIVO traceCOO.txt
103 - PASSAGEM DE PARÂMETROS DO URE1 PARA O COO - ARQUIVO traceCOO.txt
104 - PASSAGEM DE PARÂMETROS DO URE2 PARA O COO - ARQUIVO traceCOO.txt
105 - PASSAGEM DE PARÂMETROS DO URE3 PARA O COO - ARQUIVO traceCOO.txt
106 - PASSAGEM DE PARÂMETROS DO RTR1 PARA O COO - ARQUIVO traceCOO.txt
107 - PASSAGEM DE PARÂMETROS DO RTR2 PARA O COO - ARQUIVO traceCOO.txt
108 - PASSAGEM DE PARÂMETROS DO RTR3 PARA O COO - ARQUIVO traceCOO.txt
```

```
201 - RASTREAMENTO DO AGENTE UFD1 - ARQUIVO traceUFD1.txt
301 - RASTREAMENTO DO AGENTE URE1 - ARQUIVO traceURE1.txt
401 - RASTREAMENTO DO AGENTE URE2 - ARQUIVO traceURE2.txt
501 - RASTREAMENTO DO AGENTE URE3 - ARQUIVO traceURE3.txt
```

ARQUIVOS GERADOS SEM NECESSIDADE DE IMPOR CÓDIGO

```
c:\\LindoAPI\\custo.txt
c:\\LindoAPI\\deficit.txt
c:\\LindoAPI\\trace.txt
c:\\LindoAPI\\vazaoUFD1.txt
```

```
c:\\LindoAPI\\vazaoURE1.txt  
c:\\LindoAPI\\vazaoURE2.txt  
c:\\LindoAPI\\vazaoURE3.txt
```

Para se rastrear um determinado item, atribui-se à variável *DEBUGAR* o valor do código. O programa está preparado para efetuar a exportação dos dados em arquivo específico, conforme ilustra o segmento de código abaixo.

```
if (DEBUGAR == 201){  
    comando 1;  
    ...;  
}
```

7. RESULTADOS OBTIDOS

O custo hidráulico em cada estágio é calculado em função da quantidade de energia futura em cada reservatório, representada pela vazão. Em cada estágio, as usinas efetuam um procedimento para identificar a série que melhor se adapta às condições vigentes. Com as séries selecionadas, pode-se calcular o montante de afluência futura em cada usina, possibilitando verificar se a energia hidráulica futura, juntamente com as capacidades de geração das térmicas, é suficiente para atender a demanda futura.

Havendo déficit, o montante é distribuído pelo período restante, resultando em um déficit médio mensal calculado conforme as fórmulas seguintes. Com este valor, distribui-se para a somatória da capacidade de geração das usinas hidroelétricas, para saber quanto o déficit mensal observado, incide com custo na geração hidráulica.

$$Dfm = \frac{GerHid + GerTerm - Demanda}{num_meses} \quad \dots(24)$$

$$Cmm = \frac{Dfm}{\sum_0^3 GerHid_i} * cTermMax \quad \dots(25)$$

onde:

Dfm = Déficit mensal médio;

$GerHid$ = Total da Geração Hidráulica Futura;

$GerTerm$ = Total da Geração Térmica Futura;

$Demanda$ = Total da Demanda Futura;

num_meses = Número de meses do período futuro;

e,

Cmm = Custo Médio Mensal;

$GerHid_i$ = Geração Máxima na Hidroelétrica i ;

$cTermMax$ = Custo da Térmica mais cara.

Os resultados obtidos são ricos em informações e promissores quanto à possibilidade de expansão do protótipo. Ao analisar as saídas de alguns dos estágios de simulação, várias conclusões são interessantes quanto aos resultados apresentados.

Tabela 7-1 Estágio 3

Estágio 3					
FO = 905376,15 d1=900,00 d2=700,00 d3=2200,00					
un. Geração	Geração	Custo Tot	Custo Un.	Cap. Máx	Cap. Mín.
	MW	\$\$\$	\$\$\$	MW	MW
URE1	205	0	0	173	0
UFD1	70	0	0	70	0
URE2	981	0	0	1327	0
URE3	1472	0	0	2280	0
TER1	480	2146	4	480	0
TER2	105	3771	36	105	0
TER3	400	20960	52	400	0
RTR1	-215	0	0	1200	-1200
RTR2	-740	0	0	800	-800
VIR1	0	0	10000	10000	0
VIR2	0	0	10000	10000	0
VIR3	88	878500	10000	10000	0

Na tabela 7-1 acima, resultante do estágio 3, as seguintes conclusões são importantes:

- As usinas hidroelétricas URE1, URE2 e URE3 apresentam custo zero. Isto significa que não foi computado déficit no futuro;
- As usinas térmicas virtuais VIR1 e VIR2 apresentam valores nulos, mas a usina VIR3 apresenta um custo não nulo o que significa que no estágio 3 houve um pequeno déficit;

- RTR1 e RTR2 apresentam valores de geração negativos. Indica apenas que o fluxo de energia na linha de transmissão tem sentido contrário ao adotado;
- Houve consumo das térmicas, o que significa que as hidroelétricas não conseguiram suprir a demanda no estágio 3 sendo necessário utilizar o recurso das térmicas.

No estágio 46 os dados estão apresentados na tabela 7-2 seguinte, e da mesma forma, pode-se concluir que:

- As usinas hidroelétricas URE1, URE2 e URE3 apresentam custo nulo. Isto significa que não foi computado um déficit no futuro;
- RTR1 apresenta valor de geração negativo. Indica apenas que o fluxo de energia na linha de transmissão tem sentido contrário ao adotado;

Tabela 7-2 Estágio 46

Estágio 46					
FO = 15448076,15 d1=900,00 d2=700,00 d3=2400,00					
un. Geração	Geração	Custo Tot	Custo Un.	Cap. Máx	Cap. Mín.
	MW	\$\$\$	\$\$\$	MW	MW
URE1	93	0	0	93	0
UFD1	70	0	0	70	0
URE2	981	0	0	981	0
URE3	329	0	0	329	0
TER1	480	2146	4	480	0
TER2	105	3771	36	105	0
TER3	400	20960	52	400	0
RTR1	-327	0	0	1200	-1200
RTR2	200	0	0	200	-800
VIR1	0	0	10000	10000	0
VIR2	1052	10517400	10000	10000	0
VIR3	490	4903800	10000	10000	0

- As disponibilidades hidráulicas e térmicas foram integralmente consumidas, além de requisitar as térmicas virtuais VIR2 e VIR3. Isto significa que neste estágio ocorre um déficit localizado. O conjunto de

todos os déficits ao longo do período de simulação fornece subsídios para o planejamento, sugerindo a implantação de novas térmicas.

Em outro exemplo, no estágio 50 os dados estão apresentados na tabela 7-3 seguinte, e da mesma forma, pode-se concluir que:

- As usinas hidroelétricas URE1, URE2 e URE3 apresentam custo não nulo. Isto significa que foi computado um déficit no futuro resultando em um custo de geração hidráulica diferente de zero;

Tabela 7-3 Estágio 50

Estágio 50					
FO = 75804.14		d1=700.00	d2=400.00	d3=2400.00	
un. Geração	Geração	Custo Tot	Custo Un.	Cap. Máx	Cap. Mín.
	MW	\$\$\$	\$\$\$	MW	MW
URE1	151	3326	22	151	0
UFD1	70	0	0	70	0
URE2	981	21558	22	981	0
URE3	1472	32344	22	1472	0
TER1	480	2146	4	480	0
TER2	105	3771	36	105	0
TER3	242	12657	52	400	0
RTR1	-69	0	0	1200	-1200
RTR2	-294	0	0	800	-800
VIR1	0	0	10000	10000	0
VIR2	0	0	10000	10000	0
VIR3	0	0	10000	10000	0

- Havendo déficit no futuro, o custo de geração hidráulica é não nulo, exceto no caso da UFD1 porque esta usina é a fio d'água, não possuindo volume útil para acumulação. Portanto, não há como taxar o valor da geração porque tudo que entra no reservatório está disponível e fora do processo de otimização de custo;
- As três usinas térmicas virtuais VIR1, VIR2 e VIR3 apresentam valores nulos. Significa que no estágio em questão não houve déficit;

- RTR1 e RTR2 apresentam valores de geração negativos. Indica apenas que o fluxo de energia na linha de transmissão tem sentido contrário ao adotado;
- Houve consumo máximo das térmicas TER1 e TER2, e uma geração parcial na TER3. Isto significa que o processo de otimização preferiu despachar a hidroelétrica de custo 22, do que a térmica TER3 que apresenta um custo de 52.

Na iteração 51, os dados estão apresentados na tabela 7-4 seguinte, e da mesma forma, pode-se concluir que:

- As usinas hidroelétricas URE1, URE2 e URE3 apresentam custo não nulo. Isto significa que foi computado um déficit no futuro;

Tabela 7-4 Estágio 51

Estágio 51					
FO = 4882671,96 d1=900,00 d2=700,00 d3=2400,00					
un. Geração	Geração	Custo Tot	Custo Un.	Cap. Máx	Cap. Mín.
	MW	\$\$\$	\$\$\$	MW	MW
URE1	275	17619	64	275	0
UFD1	70	0	0	70	0
URE2	981	62756	64	981	0
URE3	1219	78021	64	1472	0
TER1	480	2146	4	480	0
TER2	105	3771	36	105	0
TER3	400	20960	52	400	0
RTR1	-145	0	0	1200	-1200
RTR2	-200	0	0	200	-200
VIR1	0	0	10000	10000	0
VIR2	470	4697400	10000	10000	0
VIR3	0	0	10000	10000	0

- A disponibilidade hidráulica em URE3 **não** foi integralmente consumida o que contrapõe com o consumo da térmica VIR2 de alto custo. Isto ocorre porque a capacidade do fluxo RTR2 atingiu o valor máximo de 200.

As tabelas seguintes correspondem à evolução das vazões e volumes em cada uma das usinas do protótipo.

Tabela 7-5 Evolução das Vazões em UFD1

Estágio	qAflue m3/s	qTur m3/s	qDef m3/s
2	519	174	345
3	616	174	442
4	446	174	272
5	349	174	175
6	287	174	113
7	234	174	60
8	218	174	44
9	192	174	18
10	226	174	52
11	364	174	190
12	880	174	706
13	984	174	810
14	803	174	629
15	700	174	526
16	530	174	356
17	387	174	213
18	304	174	130
19	273	174	99
20	275	174	101
21	247	174	73
22	324	174	150
23	495	174	321
24	780	174	606
25	1219	174	1045
26	1258	174	1084
27	994	174	820
28	662	174	488
29	499	174	325
30	414	174	240
31	333	174	159
32	254	174	80
33	252	174	78
34	312	174	138
35	627	174	453
36	975	174	801
37	834	174	660
38	799	174	625
39	635	174	461
40	489	174	315
41	436	174	262
42	358	174	184
43	299	174	125
44	286	174	112
45	250	174	76
46	398	174	224
47	865	174	691
48	887	174	713
49	801	174	627
50	972	174	798
51	926	174	752
52	653	174	479
53	475	174	301
54	384	174	210
55	329	174	155
56	312	174	138
57	444	174	270
58	519	174	345
59	937	174	763

A usina UFD1 é uma usina **sem** capacidade de acumulação. Desta forma, todo o volume afluyente precisa ser descarregado. Se o volume afluyente é maior do que a capacidade das turbinas, neste caso 174 (coluna qTur), a diferença deve ser descarregada (coluna qDef).

Tabela 7-6 Evolução das Vazões em URE1

Estágio	volAnt hm3	volCorr hm3	qAflue m3/s	qTur m3/s	qDef m3/s
2	1000	974	487	497	0
3	974	974	283	283	0
4	974	974	237	237	0
5	974	974	175	175	0
6	974	974	156	156	0
7	974	974	118	118	0
8	974	974	126	126	0
9	974	974	97	97	0
10	974	974	152	152	0
11	974	974	248	248	0
12	974	974	662	662	0
13	974	974	496	496	0
14	974	974	435	435	0
15	974	974	363	363	0
16	974	974	244	244	0
17	974	974	186	186	0
18	974	974	162	162	0
19	974	974	152	152	0
20	974	974	160	160	0
21	974	974	127	127	0
22	974	974	222	222	0
23	974	974	319	319	0
24	974	974	531	531	0
25	974	1120	787	669	62
26	1120	1120	739	669	70
27	1120	974	482	538	0
28	974	974	314	314	0
29	974	974	262	262	0
30	974	974	219	219	0
31	974	974	168	168	0
32	974	974	126	126	0
33	974	974	154	154	0
34	974	974	190	190	0
35	974	974	469	469	0
36	974	974	593	593	0
37	974	974	397	397	0
38	974	974	448	448	0
39	974	974	291	291	0
40	974	974	262	262	0
41	974	974	236	236	0
42	974	974	182	182	0
43	974	974	162	162	0
44	974	974	165	165	0
45	974	974	127	127	0
46	974	974	303	303	0
47	974	974	619	619	0
48	974	974	487	487	0
49	974	1057	442	410	0
50	1057	1120	632	410	198
51	1120	1120	484	410	74
52	1120	1120	310	238	72
53	1120	1120	250	238	12
54	1120	974	213	269	0
55	974	974	177	177	0
56	974	974	184	184	0
57	974	974	333	333	0
58	974	974	301	301	0
59	974	1018	686	669	0

A usina URE1 é uma usina de baixa capacidade de acumulação. O volume oscila entre o máximo (1120) e o mínimo (974) defluindo o excedente quando necessário (estágios 25, 26, 50-53).

Tabela 7-7 Evolução das Vazões e Volumes em URE2

Estágio	volAnt	volCorr	qAflue	qTur	qDef
	hm3	hm3	m3/s	m3/s	m3/s
2	12000	12358	1913	1775	0
3	12358	10541	1074	1775	0
4	10541	8071	822	1775	0
5	8071	5733	648	1550	0
6	5733	5733	572	572	0
7	5733	5733	469	469	0
8	5733	5733	469	469	0
9	5733	5733	322	322	0
10	5733	5733	667	667	0
11	5733	5733	1319	1319	0
12	5733	9002	3036	1775	0
13	9002	9548	1986	1775	0
14	9548	9473	1746	1775	0
15	9473	8208	1287	1775	0
16	8208	5821	854	1775	0
17	5821	5733	648	682	0
18	5733	5733	606	606	0
19	5733	5733	468	468	0
20	5733	5733	446	446	0
21	5733	5733	464	464	0
22	5733	5733	667	667	0
23	5733	5733	1316	1316	0
24	5733	6464	2057	1775	0
25	6464	9847	3080	1775	0
26	9847	11350	2355	1775	0
27	11350	11873	1977	1775	0
28	11873	10606	1286	1775	0
29	10606	8722	1048	1775	0
30	8722	6324	850	1775	0
31	6324	5733	624	852	0
32	5733	5733	508	508	0
33	5733	5733	652	652	0
34	5733	5733	759	759	0
35	5733	6166	1942	1775	0
36	6166	10137	3307	1775	0
37	10137	11798	2416	1775	0
38	11798	14357	2762	1775	0
39	14357	13444	1423	1775	0
40	13444	12226	1305	1775	0
41	12226	10228	1004	1775	0
42	10228	7716	806	1775	0
43	7716	5733	639	1404	0
44	5733	5733	663	663	0
45	5733	5733	503	503	0
46	5733	5733	1059	1059	0
47	5733	8022	2658	1775	0
48	8022	8320	1890	1775	0
49	8320	8561	1868	1775	0
50	8561	10191	2404	1775	0
51	10191	9647	1565	1775	0
52	9647	7630	997	1775	0
53	7630	5733	799	1531	0
54	5733	5733	661	661	0
55	5733	5733	556	556	0
56	5733	5733	590	590	0
57	5733	5733	935	935	0
58	5733	5733	1172	1172	0
59	5733	7654	2516	1775	0

A usina URE2 se destaca pela grande capacidade de acumulação, o que permite armazenar o excedente sem necessidade de descarregar pelos vertedores. (coluna qDef =0). Quando há excedente, a usina gera a capacidade máxima (1775) e armazena o restante, aumentando o volume do reservatório.

Tabela 7-8 Evolução das Vazões e Volumes em URE3

A usina URE3 possui uma capacidade de acumulação significativa, o que permite utilizar o volume de manobra adequadamente, defluindo somente quando atinge o volume máximo (17027). Havendo folga no volume, a geração é máxima (2592). Caso contrário, a usina deflue quantidade suficiente para não ultrapassar o volume máximo estabelecido.

A tabela 7-9 seguinte corresponde aos cálculos sobre o custo hidráulico obtido no processamento.

Tabela 7-9 Tabela de Composição do Custo de Geração Hidráulica

Itera	nDias Futuro	Custo Térmica Máx. \$\$\$	Cap. Hidr. Máx MW	Custo Geração Hidráulica \$\$\$	Geração		Demanda MW	Déficit MW
					Hidráulica MW	Térmica MW		
2	58	52	3980	0	259383	57130	227600	88913
3	57	52	3849	0	220025	56145	223800	52370
4	56	52	3821	0	229798	55160	220000	64958
5	55	52	3615	0	204626	54175	216200	42601
6	54	52	2872	0	177330	53190	212400	18120
7	53	52	2772	0	192074	52205	208600	35679
8	52	52	1328	0	187899	51220	204800	34319
9	51	52	995	0	155964	50235	201000	5199
10	50	52	1456	0	186640	49250	197200	38690
11	49	52	2262	0	176075	48265	193400	30940
12	48	52	4080	0	166254	47280	189600	23934
13	47	52	3979	0	183649	46295	185800	44144
14	46	52	3942	0	160537	45310	182000	23847
15	45	52	3898	0	168536	44325	178200	34661
16	44	52	3825	3	120354	43340	174400	-10706
17	43	52	2973	0	161826	42355	170600	33581
18	42	52	2901	0	156747	41370	166800	31317
19	41	52	2792	0	156919	40385	163000	34304
20	40	52	1519	0	142366	39400	159200	22566
21	39	52	1218	0	145125	38415	155400	28140
22	38	52	1737	0	153944	37430	151600	39774
23	37	52	2585	0	135817	36445	147800	24462
24	36	52	4000	0	118203	35460	144000	9663
25	35	52	4084	0	137856	34475	140000	32331
26	34	52	4084	0	129463	33490	136000	26953
27	33	52	4005	0	121032	32505	132000	21537
28	32	52	3868	0	125018	31520	128000	28538
29	31	52	3836	0	100953	30535	124000	7488
30	30	52	3810	0	108249	29550	120000	17799
31	29	52	3089	0	113739	28565	116000	26304
32	28	52	2806	0	97731	27580	112000	13311
33	27	52	2423	0	107547	26595	108000	26142
34	26	52	1746	4	74567	25610	104000	-3823
35	25	52	3450	0	84679	24625	100000	9304
36	24	52	4038	0	95746	23640	96000	23386
37	23	52	3918	0	76891	22655	92000	7546
38	22	52	3950	0	89520	21670	88000	23190
39	21	52	3854	0	71872	20685	84000	8557
40	20	52	3836	0	69810	19700	80000	9510

41	19	52	3820	0	65202	18715	76000	7917
42	18	52	3787	0	67875	17730	72000	13605
43	17	52	3498	0	59318	16745	68000	8063
44	16	52	2946	5	43549	15760	64000	-4692
45	15	52	1925	0	54496	14775	60000	9271
46	14	52	2493	0	57904	13790	56000	15694
47	13	52	4054	0	41566	12805	52000	2371
48	12	52	3973	0	42042	11820	48000	5862
49	11	52	3946	6	27863	10835	44000	-5302
50	10	52	4081	4	26673	9850	40000	-3477
51	9	52	3972	2	25704	8865	36000	-1431
52	8	52	3865	7	19697	7880	32000	-4423
53	7	52	3646	9	16942	6895	28000	-4163
54	6	52	2974	10	14674	5910	24000	-3416
55	5	52	2873	2	14500	4925	20000	-575
56	4	52	2903	0	12016	3940	16000	-44
57	3	52	2595	0	13027	2955	12000	3982
58	2	52	2648	0	8658	1970	8000	2628
59	1	52	4084	0	7440	985	4000	4425

8. CONCLUSÕES

O *framework JADE* utilizado neste trabalho, fundamenta-se em duas tecnologias importantes que contribuem para os propósitos desta tese. Por um lado, utiliza a tecnologia de elementos de software independentes denominados de *threads* e para a comunicação entre as *threads* utiliza o padrão da *FIPA*. São duas características que permitem a implementação de *agentes* e o processamento distribuído. A ferramenta *JADE* possibilita a comunicação entre os *agentes*, que podem estar sendo processados em unidades fisicamente separadas, a qualquer distância, desde que estejam interligadas através de uma rede *TCP/IP*. Para a comunicação entre os *agentes*, utiliza-se um endereço do tipo ***http://IP:7778/acc*** ou ***http://Servidor:7778/acc***, o que significa, que a *thread* está sendo processada no endereço *IP* ou, no equipamento de nome ***Servidor***. A porta de comunicação padrão adotada no *JADE* é a indicada acima *7778/acc*.

Com a implementação do protótipo utilizando o *framework JADE*, fica comprovada a viabilidade de utilizar várias unidades de processamento conectadas através de uma rede, coordenando o processamento através de um *agente* central, comunicando-se através de mensagens. Não existe nenhuma limitação quanto ao número de unidades remotas, nem qualquer requisito especial de hardware, sendo necessário apenas existir uma conexão através de uma rede *TCP/IP*.

No diagrama do sistema de geração do protótipo, cada elemento (usinas hidroelétricas, térmicas, nós de transmissão) é representado por um *agente* que é implementado como um elemento de software, sendo desenvolvido como uma classe. Como parte desta classe, é possível criar tantos métodos quanto forem necessários, para expandir a funcionalidade do *agente* ou refinar seus procedimentos. No protótipo implementado, os *agentes* constituem uma sociedade simplificada que juntos, resolvem um problema específico interagindo entre eles. Este protótipo pode ser expandido, criando-se novos *agentes* ou desenvolvendo-se novos métodos, tornando

viável sua utilização em problemas maiores, simulando situações reais de problemas de otimização.

O tempo de simulação, um dos principais requisitos do Usuário, apresentou valores muito pequenos no protótipo. O tempo total de simulação incluindo gravações de dados em arquivos foi de 5 segundos aproximadamente. A otimização através da pesquisa operacional para um estágio consumiu menos de centésimos de segundos. Expandindo o número de variáveis para pouco mais de uma centena, os testes utilizando o Solver da planilha Excel forneceram 5 segundos para determinação da solução, em um estágio. Desta forma, uma expansão do modelo para simular um caso real do setor elétrico brasileiro, não representa uma dificuldade em termos de tempo de simulação para a pesquisa operacional, pois ainda se considera um problema de pequenas dimensões. Os maiores problemas de otimização estão na área de Logística e Transportes onde o número de variáveis são elevados, mas mesmo neste caso, ainda existe a alternativa de agrupá-las em famílias, para diminuir a quantidade de equações e variáveis.

Um dos recursos utilizados no protótipo, para evitar que a otimização gaste tempo de processamento tentando encontrar uma solução, quando ela não existe, refere-se à colocação de unidades térmicas fictícias de capacidade ilimitada (ou propositadamente elevada) e custo alto (já que a função-objetivo é de mínimo custo) nos nós. Desta forma, não havendo solução em um determinado estágio, a otimização utiliza estas térmicas obtendo com isto, dois benefícios quais sejam: a) O método sempre encontra uma solução e com isto, não consome tempo fazendo iterações no caso de insolvência. b) Ao utilizar a térmica fictícia, o método indica claramente onde e quando aconteceu o déficit e de quanto foi o valor do mesmo.

A aplicação do método QFD para identificar os requisitos do Usuário apresentou resultados satisfatórios e sua importância é maior, na medida em que a lista de requisitos é um dos alicerces que sustentam os propósitos

desta tese, juntamente com a pesquisa dos métodos numéricos e a especificação da infra-estrutura requerida.

Para a análise dos processos numéricos, foi necessária a criação de um modelo intitulado MC-HSQ com a finalidade de auxiliar no processo de identificação de soluções, considerando os requisitos do Usuário e as infra-estruturas necessárias. Como resultado da aplicação deste modelo, obteve-se um protótipo com características que permitem concluir como adequada, a configuração *Processo Numérico versus Hardware* atendendo à lista de requisitos estabelecida para o protótipo.

9. CONSIDERAÇÕES FINAIS

Com a conclusão deste trabalho, identificaram-se dois aspectos importantes que revelam novos caminhos, para a busca de alternativas de solução dos problemas de otimização.

Sob o ponto de vista de infra-estrutura, o desenvolvimento do protótipo mostrou a viabilidade de expandir o poder de processamento em função do aumento na dimensão do problema, sem que para isto, haja necessidade de grandes investimentos ou adoção de procedimentos complexos.

Por outro lado, sob o ponto de vista de software, não existe nenhuma limitação quanto ao nível de refinamento dos processos de simulação, o que pode ser conseguido, através da implementação de novos métodos específicos.

Em outras palavras, o protótipo assim desenvolvido apresenta uma importante característica, de permitir a expansão em relação à infra-estrutura e software, o que representa um diferencial importante. A configuração do protótipo pode ser moldada às necessidades de cada caso, com grau de complexidade e refinamento adequados a cada uso, tornando-se uma ferramenta versátil e flexível. Isto responde às questões formuladas pelas diversas áreas das empresas de geração de energia elétrica, envolvendo assuntos de planejamento, marketing, operação de usinas etc.

O modelo MC-HSQ, utilizado para desenvolver o protótipo, contribuiu de forma eficiente sendo, portanto, considerado como apropriado, para estabelecer as diretrizes na criação de protótipos voltados para problemas de otimização.

Os esforços direcionados em Infra Estrutura, Software e os Requisitos de Qualidade, notadamente em relação à usabilidade, demonstraram eficiência como procedimentos de uma nova abordagem, na produção de um software de qualidade. A matriz de decisão do MC-HSQ utilizada neste trabalho, mostrou ser viável como ferramenta, possibilitando visualizar soluções alternativas e configurações adequadas de recursos de hardware, além de projetar uma usabilidade adequada valendo-se inclusive de modelos combinados, denominados de híbridos.

A experiência adquirida na utilização de multi-agentes, com a possibilidade de efetuar processamento distribuído em aplicações de engenharia, forneceu uma noção dos benefícios e custos agregados de cada alternativa pesquisada. É um passo importante em direção ao futuro, onde a disponibilidade de processadores em *desktops* individuais ou em rede será aproveitada, formando um poderoso conjunto de processamento, capazes de resolver questões complexas, que exigem grande capacidade de processamento.

Muitas vertentes se abrem com a conclusão deste trabalho como objeto de pesquisa. O *framework JADE* utilizado, tem muitos recursos que podem ser explorados com mais profundidade, desenvolvendo sociedades de multi-agentes complexas para diversas finalidades na área de energia ou em outras áreas.

Os agentes que definem a disponibilidade energética é outro objeto de estudos e pesquisas, na busca de modelos abrangentes que considerem todas as restrições e limitações operacionais das usinas e termoeletricas. As interações entre agentes quando exploradas de maneira ampla e eficaz, produzem resultados globais de qualidade, comparáveis aos produzidos por modelos exatos, impróprios para serem utilizados em questões de

otimização de grande complexidade. Expansões desta estratégia podem ser objetos de pesquisas avaliando a possibilidade de efetuar a simulação do período mais de uma vez, considerando o custo futuro, admitindo como verdadeiro, refazendo a simulação.

No setor elétrico, o protótipo criado torna-se um objeto de pesquisa com grandes possibilidades para estudos de alocação de térmicas, identificação da suficiência de usinas geradoras, análise e avaliação de risco de déficits, modelos de negociação entre empresas geradoras, avaliação de custos de geração, valor de mercado, etc.

Sob o ponto de vista computacional, a questão do processamento distribuído pode constituir-se em mais um item relevante de pesquisas, analisando desempenho, confiabilidade, sincronismo e aplicabilidade em processamentos críticos.

Concluindo, fica registrado neste trabalho, um início de uma nova abordagem para solução de problemas de otimização que pode se expandir com novas implementações e idéias, uma vez que a estratégia do modelo admite uma infinidade de agregações de métodos e particularizações, adaptando-se às características de cada problema.

10. BIBLIOGRAFIA CITADA

- (AIEX 2003) Aix, R. M., Binato, S., Resende, M. G. C., Parallel GRASP with path-relinking for job shop scheduling, *Parallel Computing*, v. 29, p. 393–430, 2003.
- (AKAO 2003) Akao, Y., Mazur, G. H. The leading edge in QFD: past, present and future, *International Journal of Quality & Reliability Management* v. 20, n.1, p. 20-35, 2003.
- (AKAO 2004) Akao, J., *Quality Function Deployment: Integrating Customer Requirements into Product*, Yoji Akao Editor, p. 324, 2004.
- (BANKER 1991) Banker, R. D., Kauffman, R. J., Reuse and productivity in integrated computer-aided software engineering: an empirical study, *MIS Quarterly*, v.15 n. 3, p. 375-401, 1991.
- (BISKUP 2001) Biskup D., Feldmann M. Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers & Operations Research* v. 28, p. 787-801, 2001.
- (BOISSIER 1993) Boissier, O., "ASIC: An architecture for social and individual control and its application to computer vision", *European Workshop on Modeling Autonomous Agents in a Multiagent World*, p. 107-118, 1994.
- (DAHALL 2007) Dahal P. K., Aldridge C. J., Galloway S.J. Evolutionary hybrid approaches for generation scheduling in power systems. *European Journal of Operational Research*, v. 177, p. 2050-2068, 2007.
- (DROGOUL 1993) Drogoul, A., Dubreuil, C. A distributed approach to N-puzzle solving, *13th Workshop on Distributed Artificial Intelligence*, p. 13, 1993.
- (ENCINA 2006) Encina, A. S. A., *Despacho Ótimo de Unidades Geradoras em Sistemas Hidroelétricos via Heurística*

- Baseada em Relaxação Lagrangeana e Programação Dinâmica. Tese de Doutorado, FEEC/UNICAMP, 2006.
- (FIPA 2005) FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS. Disponível em <http://www.fipa.org/specs/fipa00061/SC00061G.html>.
- (FYLSTRA 1998) Fylstra, D., Lasdon L., Watson J., Waren A. Design and Use of the Microsoft Excel Solver, INTERFACES 28, v.5 September-October p. 29-55, 1998.
- (GLOVER 1995) Glover, F., Kelly, J.P., Laguna, M. Genetic algorithms and tabu search: Hybrids for optimization, Computers & Operations Research, v. 22, p. 111–134, 1995.
- (GONSÁLEZ 2004) Gonsález, M. E., et al., Customer satisfaction using QFD: an e-banking case, Managing Service Quality, v.14, n. 4, p. 317-330, 2004.
- (HINO, 2005) C. M. Hino, D. P. Ronconi , A. B. Mendes. Minimizing earliness and tardiness penalties in a single-machine problem with a common due date, European Journal of Operational Research. v. 160, p. 190-201, 2005.
- (HUANG 2008) Huang K. L., Liao, C. J., Ant colony optimization combined with Taboo Search for the Job Shop Scheduling Problem, Computers & Operations Research, v. 35, p. 1030–1046, 2008.
- (JAMES 1997) R. J. W. James. Using tabu search to solve the common due date early/tardy machine scheduling problem, Computers & Industrial Engineering v. 24, p. 199-208, 1997.
- (LAHDELMA 2003) Lahdelma R., Hakonen H., An efficient linear programming algorithm for cobined heat and power production. European Journal of Operational Research, v. 148, p. 141-151, 2003.

- (LIU 2005) Liu X., et al., QFD Application in software Process Management and Improvement Based on CMM, 27th International Conference on Software Engineering, St Louis, Missouri, USA, p. 1-6, 2005.
- (LIU F. 2006) Liu F., et al. A quantitative approach for setting technical targets based on impact analysis in software quality function deployment (SQFD), Software Quality Control, v. 14, n. 2, p. 113-134, 2006.
- (MADU 2006) Madu, C. N., House of Quality in a Minute, 6a Edição, Chi Publisser, New York, 2006.
- (MANZUR 1993) Manzur, G., H., QFD for Service Industries from Voice of Customer to Task Deployment, The Fifth Symposium on Quality Function Deployment, Novi Michigan, June, p. 485-503, 1993.
- (MURAKAMI 2005) Murakami, L. T., Martini, J. S. C., Spina, E., QFD applied to software development, The 2005 International Conference on Software Engineering Research and Practice (SERP'05), Las Vegas, USA, p. 27-30, 2005.
- (NAGATA 2002) Nagata, T., et al., A Multi-Agent Approach to Unit Commitment, Power Engineering Society Winter Meeting, v. 1, p. 64, 69, 2002.
- (PIDD 1999) Pidd M. Just Modeling Through: A Rough Guide to Modeling, INTERFACES 29, v. 2 March-April p. 118-132, 1999.
- (PIRLOT 1996) Pirlot M. General local search methods, European Journal of Operational Research v. 92, p. 493-511, 1996.
- (PRESSMAN 2006) Pressman, R., Engenharia de software 6a Edição. MacGraw Hill, São Paulo, 2006.
- (RAJAN 2004) Rajan, C. C., A., Mohan M. R. An Evolutionary Programming-Based Tabu Search Method for Solving the Unit Commitment Problem. IEEE Transactions on Power Systems, v. 19 n. 1 p. 577-585, 2004.

- (RUIZ 2002) Ruiz M., Ramos, I., Toro, M., A Dynamic Integrated Framework for Software Process Improvement, Software Quality Journal, v. 10, p. 181–194, 2002.
- (SAVAGE 1985) Savage L.S. What's Best – User Manual, General Optimization Inc, Chicago, Illionois, 1985.
- (SCHUSTER 1998) Schuster, E.W., Allen, S. J. Raw Material Management at Welch's, Inc., INTERFACES 28, v.5 September-October p. 13-24, 1998.
- (SICHMAN 2001) Sichman, J. S., Demazeau, Y. On social reasoning in Multi-Agents Systems, Revista Ibero-Americana de Inteligencia Artificial, v. 13, p. 68-84. Madrid: AEPIA, 2001.
- (TAGHEM 1995) Taghem, J., Pirlot M. Antoniadis, C. Embedding of Linear Programming in a simulated annealing algorithm for solving a mixed integer production planning problem. Journal of Computational and Applied Mathematics v.64, p.91-102, 1995.
- (YU 2005) Yu J., Zhou J., Hua B., Liao R., Optimal Short-Term Generation Scheduling with Multi-Agent System under a Deregulated Power, International Journal Of Computational, v. 3, n. 2, p. 61, 65, 2005.
- (ZOUMAS 2004) Zoumas, C. E., Bakirtzis, A.G., Theocharis, J.B. Petridis, V. A Genetic Algorithm Solution Approach to the Hydrothermal Coordination Problem. IEEE Transaction on Power Systems, v. 19 n. 3, p. 1356-1364, 2004.

11. BIBLIOGRAFIA CONSULTADA

- Abrahamsson, Pekka – “Commitment to Software Process Improvement—Development of Diagnostic Tool to Facilitate Improvement”, *Software Quality Journal*, 8, 63-76, 1999.
- Baker K.R., Scudder GD. – “Sequencing with earliness and tardiness penalties. A review”, *Operation Research* 1990, 38:22-36.
- Beasley J.E. – “Distributing test problems by electronic mail”, *Operational Research* 1990,41(11):1069-1072
- Ben-Ari, M. – “Principles of concurrent programming”, Prentice-Hall International, 1982
- Carlier, Jacques & Rebai, Ismail – “Two branch and bound algorithms for the permutation flow shop problem”, *European Journal of Operational Research* 90 (1996) 238-251
- Chae U. Lee & Seok J. Kim – “Parallel Genetic Algorithms for the earliness-tardiness job scheduling problem with general penalty wights”, *Computers & Industrial Engineering* 1995;28:231-43.
- Foster, Ian – “Designing and building parallel programs”, Addison-Wesley Publishing Company
- G. Amdahl - “Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities”, *AFIPS Conference Proceeding*, vol. 30, Abril, 1967, Thompson Books, Washington DC.
- Harry R. Lewis & Christos H. Papadimitriou – “Elementos de Teoria da Computação”. Ed. Bookman, 2a Edição,2000
- He, M. & Jennings, N. R. & Leung, H. – “On Agent-Mediated Electronic Commerce”, *IEEE Trans on Knowledge and Data Engineering*, Vol. 15 (04), pages 985-1003, 2003.
- Hübner, J. F. & Sichman, J. S. & Boissier, O. – “A Model for Structural, Functional and Deontic Specification of Organizations in Multiagent Systems”, In: *Proc. 16th Brazilian Symposium on Artificial Intelligence (SBIA'02)*, Porto de Galinhas, Brasil, 2002.

- Humphrey, Watts S. – “Managing the Software Process”. Addison-Wesley Publishing Company, Massachusetts, 1990.
- Matthew L., Siddiqi A. – “Can the case for CASE technology be advanced by Process Improvement?” Software Quality Journal 7,(1998) 3–10.
- Mayhew, Deborah J. – “The Usability Engineering LifeCycle”, Morgan Kaufmann Publishers, 1999
- Montemanni, R. & Gambardella, L.M. & Donati, A.V. – “A branch and bound algorithm for the robust shortest path problem with interval data”, Operations Research Letters 32 (2004)225 –232
- Morizawa, Kazuko & Sun, Xi & Nagasawa, Hiroyuki – “Squeezing branch and bound algorithm for the machine-fixed, machining-assembly flowshop scheduling problem”, Japan Manufacturing Technology and Management, Vol. 5, Nos. 1/2, 2003
- Nielsen, J. “Projetando Web Sites. Designing Web Usability”. Editora Campus, 2000.
- Nowicki E., Smutnicki C. – “A fast tabu search algorithm for the permutation flow-shop problem”. European Journal of Operational Research 1996, 91:160-175.
- Omar A.R., Harding J.A. and Popplewell K. – “Design for customer satisfaction: an information modelling approach”, Integrated Manufacturing Systems 10/4 [1999] 199±209.
- PACS Training Group – Introduction to MPI, University of Illinois.
- Paolo Toth – “Optimization engineering techniques for the exact solution of NP-hard combinatorial optimization problems” European Journal of Operational Research 125 (2000) 222±238
- Rabadia, Ghaith & Mollaghasemib, Mansooreh & Anagnostopoulosc , Georgios C. – “A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time”, Computer & Operation Research 31(2004) p.1727-1751

- Reiblein S., Symons A., – “SPI: ‘I can’t get no satisfaction’ – directing process improvement to meet business needs”, Software Quality Journal 6, (1997) 89–98.
- Rios-Mercado, Roger Z. & Bard, Jonathan F. – “A branch-and-bound algorithm for permutation flow shops with sequence-dependent setup times”, IIE Transactions (1999) 31, 721±731
- Sichman, J. S. – “Raciocínio Social e Organização em Sistemas Multiagentes: Avanços e Perspectivas”, Tese de Livre Docência apresentada a Escola Politécnica da Universidade de São Paulo, São Paulo, Brasil, 2003.
- Terninko, John – “Step by Step – QFD Customer-Driven Product Design” – 2nd Edition – 1997
- Miller, Russ & Boxer, Laurence – Algorithms Sequential & Parallel – A unified approach, Prentice Hall
- Foster, Ian T. – Designing and Building Parallel Programs, Addison – Wesley Publishing Company

12. SITES CONSULTADOS

- <https://cajo.dev.java.net/>
- www.ons.org.br
- www.aneel.gov.br
- www.cesp.com.br
- www.top500.com
- <http://www.inf.puc-rio.br/~inf1626/>
- <http://www.comsol.se/products/stateflow/>
- http://www.lurpa.ens-cachan.fr/csap/publi/publi_96_7/cesa96_agglae_1.html
- <http://www.ece.uc.edu/~ddel/projects/mss/iee92/print.html>
- <http://www.sei.cmu.edu/tsp/history.html>
- www.w3.org/WAI
- <http://www.qfdi.org/>
- www.top500.org
- <http://www.inf.puc-rio.br/~inf1626/>
- <http://www.comsol.se/products/stateflow/>
- http://www.lurpa.ens-cachan.fr/csap/publi/publi_96_7/cesa96_agglae_1.html
- <http://www.ece.uc.edu/~ddel/projects/mss/iee92/print.html>
- <http://java.sun.com/j2se/1.5.0/docs/guide/idl/index.html>
- <http://www.mundooo.com.br/php/mooartigos.php?pa=showpage&pid=17#1>
- <http://www.oracle.com/technologies/grid/index.html>
- www.usedit.com