José Jean-Paul Zanlucchi de Souza Tavares

Sistemas de Informação: Estudo de Caso no Fluxo de Materiais para a Fabricação de Blanks.

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

São Paulo
2000

José Jean-Paul Zanlucchi de Souza Tavares

Sistemas de Informação: Estudo de Caso no Fluxo de Materiais para a Fabricação de Blanks.

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

Área de Concentração:
Engenharia Mecânica

Orientador:
Prof. Dr. José Reinaldo Silva

São Paulo
2000

"Tudo posso naquele que me conforta." Fl. 4, 13.

Ofereço este trabalho à Vós, nosso Deus, pois sois Quem incentiva e auxilia na busca de novos conhecimentos através de Vossa paciência, incentivo, carinho e proteção.

Agradecimentos

Ao meu grande amigo e orientador Prof. Dr. José Reinaldo Silva pelo constante apoio e paciência na conclusão desse trabalho.

Ao meu grande amigo Prof. Dr. Gilmar Ferreira Batalha pelas horas despendidas e pelo grande incentivo nesse trabalho.

À minha família pelo carinho e infinito auxílio a mim dirigido.

À FAPESP pelo apoio financeiro em grande parte desse trabalho.

À CSN pelo auxílio e oportunidade de conhecimento.

Índice

Sumário:

Lista de Figuras

Resumo

Após a revisão sobre modelagem de sistemas automatizados e projeto da informatização da área de administração de materiais na fabricação de blank soldado, apresenta-se uma metodologia orientada a objetos dedutiva denominada AV/AO (Abstract View/ Abstract Object) para a concepção e desenvolvimento da representação formal de sistemas complexos, objetivando a reutilização de soluções e uma abordagem híbrida Top-Down e Bottom-Up.

Um estudo de caso na área de administração de materiais na fabricação de blank soldado é apresentado pelo fato da sua crescente adoção no processo de conformação de chapas e conseqüentes mudanças no processo de manufatura. Através da aplicação do sistemógrafo constata-se falhas no sistema de informação devido a falta de integração. Com a modelagem AV/AO é possível propor um sistema informacional integrado.

Resumo

Após a revisão sobre modelagem de sistemas automatizados e projeto da informatização da área de administração de materiais na fabricação de blank soldado, apresenta-se uma metodologia orientada a objetos dedutiva denominada AV/AO (Abstract View/ Abstract Object) para a concepção e desenvolvimento da representação formal de sistemas complexos, objetivando a reutilização de soluções e uma abordagem híbrida Top-Down e Bottom-Up.

Um estudo de caso na área de administração de materiais na fabricação de blank soldado é apresentado pelo fato da sua crescente adoção no processo de conformação de chapas e conseqüentes mudanças no processo de manufatura. Através da aplicação do sistemógrafo constata-se falhas no sistema de informação devido a falta de integração. Com a modelagem AV/AO é possível propor um sistema informacional integrado.

Abstract

After a review of automated systems modeling and design of information system for material management area of welded blank, it is showed a deductible object oriented methodology named AV/AO (Abstract View/ Abstract Object). This methodology was developed to concept and develops of complex system formal representation, looking for the reusing of solutions and a hybrid Top-Down and Bottom-Up approach.

A case study in material management area of tailored blank is showed by the fact of its great using in sheet metal forming and its consequent changes in the manufacturing process. The sistemógrafo application, at this case, shows faults in information system created by integration breaks. With AV/AO modeling is possible to purpose a integrated information system.

Capítulo 1

Introdução

A busca pela automação e integração de sistemas complexos gerou inúmeras metodologias localizadas, entretanto, sem uma análise do sistema global (operação, informação e gerência) uma melhora local pode amplificar os erros e falhas do sistema. Os sistemas tipo ERP (Enterprise Resource Planning – Planejamento de Recursos da Empresa) são um bom exemplo de melhora local. Através desse tipo de sistema é possível integrar as áreas de informação e gerenciamento de uma empresa, por exemplo administração de materiais e a área financeira; todavia, para que a operação possa ser integrada a esse sistema é necessário uma série de interfaces e outros sistemas, interfaces essas que geram falhas e perdas de comunicação acarretando na perda de integração. Nesse sentido é necessário modelar o sistema inicial, corrigir as falhas existentes para propor um sistema a ser implementado.

Esta se estudando especificamente uma classe de sistemas complexos pela facilidade de formalização, a saber, sistemas automatizados que, como apresentado no esquema OMT (Object Modeling Technique) de [Rumbaugh91], tem a arquitetura baseada em três modelos, respectivamente: funcional, estrutural e dinâmico.

O modelo funcional está relacionado com o planejamento de produção e aspectos gerenciais. O modelo estrutural está vinculado ao processo de manufatura, máquinas e equipamentos. O modelo dinâmico está relacionado com o comportamento do sistema e dos seus sub-sistemas de controle.

O modelo funcional e estrutural genericamente são modelados em conjunto, enquanto o modelo dinâmico é gerado após essa modelagem inicial. Dessa forma a conexão desse modelo inicial com os aspectos de controle do modelo dinâmico é, freqüentemente, um desafio e um problema ainda em aberto.

A metodologia apresentada explora a dualidade entre duas classes de objetos complexos: o AV e o AO [Tavares97], [Silva98]. Esta característica é, de fato, baseado na dualidade encontrada em sistemas dinâmicos discretos, tais como aquelas expressadas pelas Redes de Petri dentre outras [Silva94]. Dessa forma uma conexão entre a representação estrutural/funcional e a dinâmica ainda está em aberto.

1.1 Justificativas

Este estudo focaliza o processo de negócio da Filial São Paulo da Companhia Siderúrgica Nacional - CSN, indústria siderúrgica, localizada na cidade de São Paulo e responsável pela gestão da terceirização do corte de bobinas em blank.

A CSN procurando se manter competitiva e atender um importante segmento de mercado de aço, como a indústria automobilística, assumiu o desafio de produzir blank e blank soldado para grandes montadoras. Para isso foi importante o conhecimento das complexas necessidades dos clientes, a caracterização dos produtos, inicialmente importados, desenvolvimento dos materiais para os blanks, caracterizando-os juntamente com os produtos das peças importadas em termos de conformabilidade, e definição dos melhores parâmetros de soldabilidade para a tecnologia de solda à laser. A partir de julho de 1998, iniciou-se a produção dos blanks nacionalizados [Pincinini99]. Com isso surgiram problemas na fabricação dos produtos, ou seja, no chão de fábrica. Um dos problemas encontrados foi a fragilização da solda no momento do embutimento do blank soldado. Constatou-se que, por se tratar de aço com baixo teor de carbono, ocorre envelhecimento do material [Geiger95]. Por esse motivo, blanks soldados com tempos de estoque muito discrepantes, maior que 06 (seis) meses entre si, apesar de soldados com os parâmetros corretos, depois de embutidos apresentam trincas ou se rompem no local da solda.

A tecnologia do blank soldado está consolidada [Wagener97]. A união de materiais com especificações e espessuras diferentes que permitem a redução do peso do produto, o aumento de sua rigidez e segurança já é uma necessidade da indústria automobilística [Pincinini99].

Sendo assim a CSN desde 1998 tem-se reestruturado para automação de todos os seus processos de negócio, e implantou na Filial São Paulo um sistema do tipo ERP (Enterprise Resource Planning) desde maio de 1999. Em função da necessidade da operação necessitar saber o tempo das bobinas a serem soldadas para que não ocorram problemas em processos seguintes, está se desenvolvendo esse trabalho voltado a administração de materiais. Através do sistemógrafo pode-se verificar as falhas de informação existentes no sistema implantado, e, pela da modelagem AV/AO pode-se desenvolver um sistema integrado que atenda as exigências e necessidades da Filial.

1.2 Objetivo

O objetivo desta dissertação é apresentar um metodologia formal, chamada AV/ AO (Abstract View/ Abstract Object) que propicia a integração de sistemas automatizados, baseando-se na representação do sistema atual pelo Sistemógrafo [Bresciani97b], conforme mostra a Figura 1.

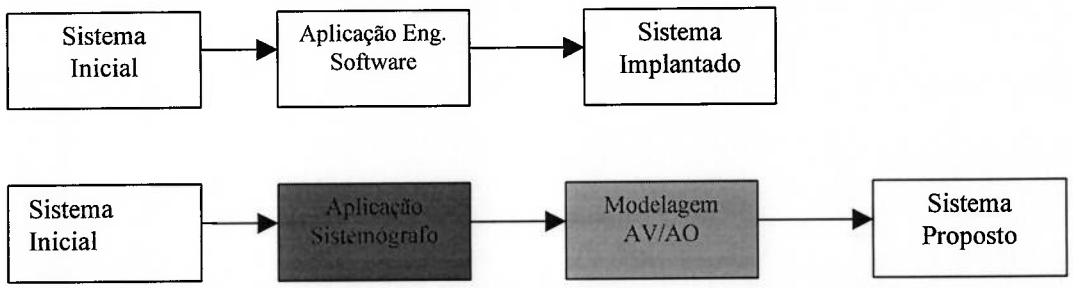


Figura 1 - Proposta de aplicação do Sistemógrafo e Modelagem AV/AO.

O Capítulo 2 apresenta conceitos básicos sobre administração de materiais e blank, o Capítulo 3 mostra um histórico sobre projeto de sistemas automatizados, o Capítulo 4 trata da modelagem de sistemas automatizados, o Capítulo 5 descreve o modelo AV/AO, o Capítulo 6 apresenta o estudo de caso para integração da administração de materiais na fabricação de blanks, e o Capítulo 7 mostra a discussão e conclusão.

Capítulo 2

Conceitos Básicos

2.1 Administração de Materiais

A organização empresarial é composta de áreas que interagem na busca do objetivos das empresas. Cada área é formada de vários segmentos (pessoal, material, informações) que se relacionam na busca de objetivos comuns [Carretoni93]. Na área de administração de materiais, o gerenciamento é realizado através das atividades como: planejar, comprar, estocar, fornecer, controlar.

A principal função da área de Administração de Materiais é ser responsável pelo abastecimento ininterrupto de materiais essenciais para manter a organização em funcionamento, levando em consideração especificações de materiais, prazos de entrega e custos de compra.

Os materiais precisam ser bem administrados para que não haja escassez que interrompa a produção, nem excessos que elevem os custos. A administração de materiais consiste em ter os materiais na quantidade e qualidade certas, no local certo e na hora certa, à disposição das áreas que fazem parte do processo produtivo da empresa [Chiavenato91].

Nas empresas industriais há, basicamente, os seguintes tipos de materiais:

a) matéria-prima: materiais utilizados ou agregados na produção de um produto.

b) produto em processo: são produtos que ainda não estão acabados, pois ainda estão em algum setor intermediário de produção.
c) produto acabado: são materiais que já foram produzidos, mas ainda não estão vendidos.
d) material de manutenção: são basicamente os materiais utilizados na manutenção dos equipamentos de produção.

Nos últimos anos tem aumentado a importância da área de materiais, podendo ser explicado principalmente pelas seguintes razões [Dias96]:

1. aumento dos custos, principalmente do transporte e do armazenamento;
2. melhoria da tecnologia (particularmente de tecnologia de informática) para utilização de informações sobre a logística.
3. complexidade da administração de materiais.
4. aumento da diversificação das funções da área de materiais.

Com essas premissas referentes à administração de materiais, serão vistos neste trabalho a racionalização e o desenvolvimento de um sistema de administração de materiais.

2.2 BLANK

O blank é um recorte plano de um metal, de propriedades e forma específica, que sendo submetido a uma operação de conformação será transformado em peça acabada. Tradicionalmente é uma peça homogênea, obtida a partir de bobinas de metal, processadas em Centro de Serviços ou máquinas de corte da própria empresa que produz a peça. Se uma peça maior, ao longo de sua extensão, requisesse diferentes propriedades mecânicas, resistência à corrosão, resistência à deformação depois de confeccionada ou conformabilidade optava-se sempre por um material que atendesse a condição mais importante ou a mais severa para a peça, em detrimento das outras qualidades que não poderiam ser atendidas de forma ótima, com o uso de Blank Soldado se obtive uma melhor solução para este problema.

2.2.1 O BLANK Soldado

A necessidade de reduzir custos e otimizar produtividade e qualidade intrínseca, levou à industria automobilística a desenvolver o Blank Soldado, que consiste em dividir uma peça, que era única, em parte menores, cada uma com características e exigências especificas, e uni-las por meio de solda de alta qualidade (solda por laser, resistência ou indução), formando uma peça única que será conformada inteiramente, sem necessitar de operação de união entre elas na fase de montagem dos carros.

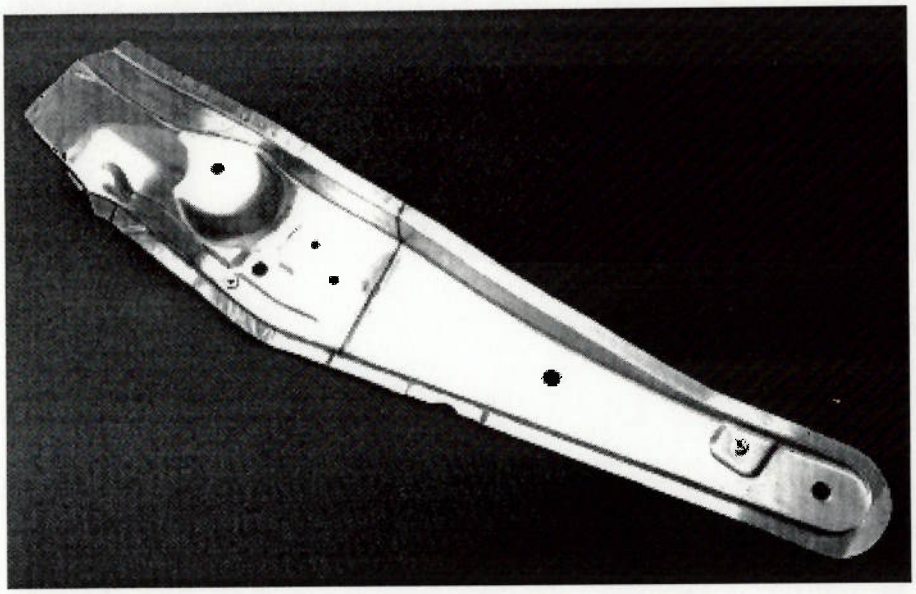


Figura 2: Reforço estrutural de um carro de passageiros. - Obtido a partir de um "Tailor Blank" conformado [MtlFrmHdk98].

Por outro lado, permite também agregar numa só peça plana, as partes que antes teriam por sua natureza diferenciada, de ser conformadas separadamente e unidas como peças acabadas, obtendo-se desse modo um número total menor das peças integrantes do carro e consequentemente melhorias no estoque das ensambladoras.

A título de ilustração a Figura 2 apresenta um Blank Soldado já conformado, o qual é o reforço estrutural de um carro de passageiros [MtlFrmHdbk98].

A Figura 3 apresenta um Blank Soldado complexo usado para fazer o painel lateral externo do veículo do projeto ULSAB (Ultra Light Steel Auto Body) [Pereira99]. Esse blank soldado reúne três diferentes graus de resistência mecânica, alem, de cinco diferentes espessuras numa mesma peça embutida.

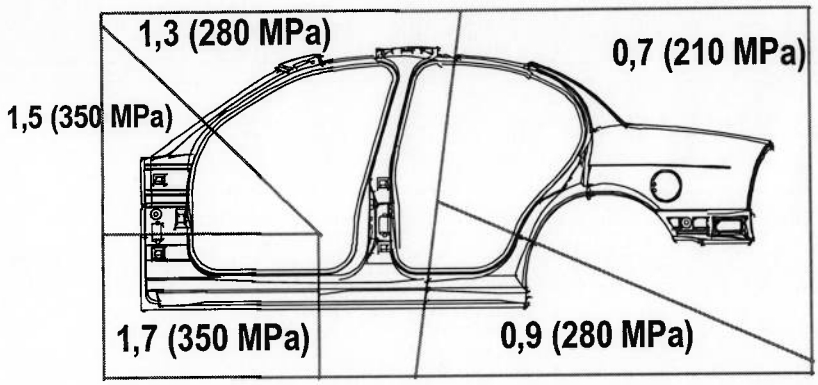


Figura 3: Painel lateral externo. – Obtido a partir da conformação de um "Tailor blank" soldado a Laser [Pereira99].

2.2.2 Processos de Produção e Conformação do BLANK Soldado

Os Blanks Soldados são produzidos em linhas preparadas para o corte e a soldagem de recortes de chapas metálicas. Na primeira etapa, isto é a ***produção dos recortes*** por guilhotinas e/ou prensas, a qualidade da superfície do corte (Figura 4), exerce uma grande influência sobre a qualidade do cordão de solda, e sobre os ciclos de produção na etapa de soldagem dos recortes.

O plano de controle da etapa de corte para definir os métodos de controle para as falhas potenciais podem ser identificadas na análise dos modos e efeitos de falhas potenciais (FMEA), tais como: ondulações, rebarba, esquadro, guias e outros.

- Ausência de material na área de costura.	- Ângulo do corte
- Posicionamento dos recortes (pontos de contato).	- Direção do cisalhado
- Retilinidade (fresta)	- Rebarba
- Relação de corte / cisalhamento	

Tabela 1: Fatores do processo de corte que influenciam a qualidade do blank soldado.

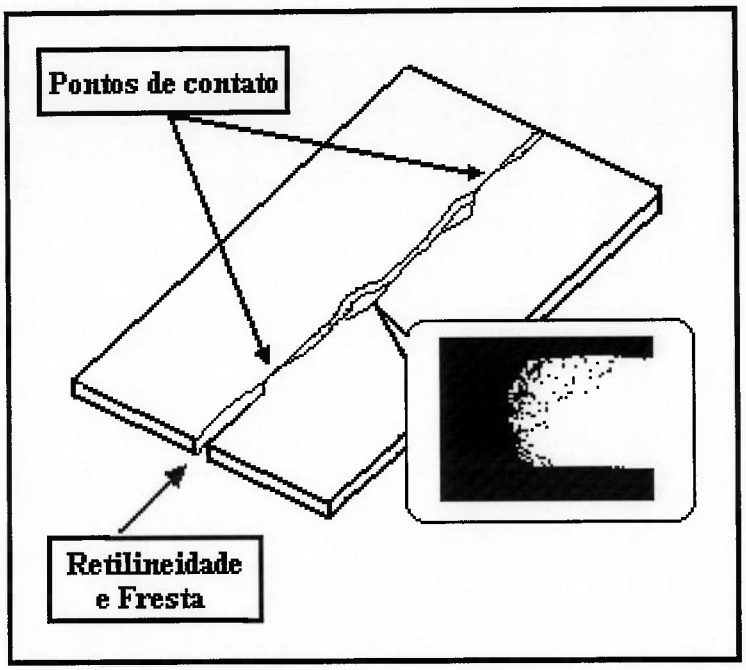


Figura 4 – Exemplos de defeitos na superfície de corte [Pincinini99].

Visando a ***etapa de soldagem***, os recortes são a seguir empilhados, alinhados, centrados, alimentados e fixados em uma máquina automática de soldagem (Figura 5), a qual pode empregar tecnologia de solda com laser, solda com resistência ou solda por indução.

No processo de produção de blank soldado estudado neste trabalho, utiliza-se uma máquina de solda a laser Soudronic Neftenbach AG, modelo LCL 2000 (Figura 6), a qual engloba um sistema de soldagem SOULAS, um dispositivo de preparação de bordas SOUKA (Figura 7), um sistema orientador do disparador de laser e um sistema de monitoramento da geometria da solda a laser, denominado SOUVIS, o qual inspeciona antes da solda, o formato do chanfro e após a soldagem os acabamentos superficiais da face e da raiz da solda e a ausência de solda. A velocidade máxima é de 80 m/min (posicionamento do material) e de 12 m/min em operação (soldagem a laser) dependendo das especificações e espessuras dos materiais.

Figura 5: Linha de produção de Blank Soldado com sistema de manuseio contínuo para a manufatura de partes laterais [MtlFrmHdbk98].

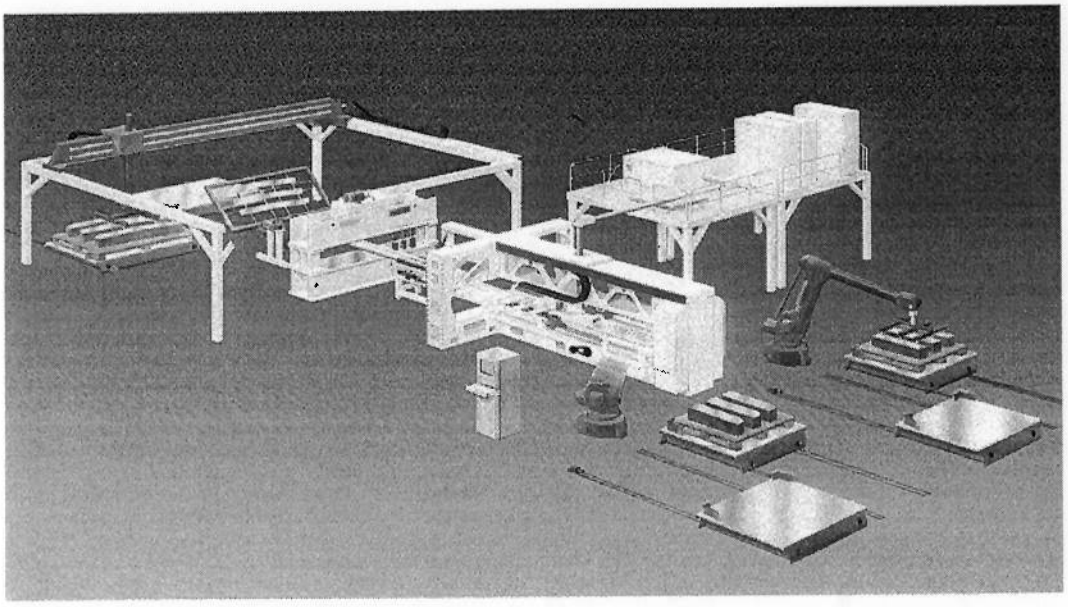


Figura 6 – Sistema de Soldagem Soulas – Lay-out do equipamento de solda a laser para fabricação dos blanks soldados [Pincinini99].

Além da inspeção pelo sistema SOUVIS, são realizados ainda:

- Exame visual da solda, avaliando o formato da face e da raiz da solda, ocorrência de porosidade e falhas de continuidade do laser (Figura 8).
- Ensaio de embutimento, visando ruptura fora da solda e da ZTA (Figura 9)
- Ensaio de tração transversal à solda, visando ruptura fora da região de solda.

- Exame metalográfico macrográfico: avaliando a concavidade, convexidade, penetração da raiz e mordedura.
- Exame metalográfico micrográfico: avaliando microconstituintes e granulação da zona termicamente afetada. (Figura 10).

Ensaios de microdureza, cujo critério é que a microdureza do metal de solda seja no máximo igual ou menor que duas vezes a do material de base mais duro (Figura 11).

Durante o processo de soldagem, os parâmetros da solda a laser são selecionados segundo a norma ISO 13919-1 e normas internas das montadoras. O plano de controle desta fase estabelece os parâmetros de soldagem (potência, velocidade, foco e outros) e as características, acima especificada, para avaliação da soldagem.

Após do processo de soldagem os Blanks Soldados passam pela estação de monitoramento e pela estação de armazenagem e proteção com óleo. O Blank Soldado concluído e empilhado novamente e à espera do seguinte processamento nas linhas de conformação. Na estampagem profunda, dependendo da posição da linha de solda, o Blank Soldado tem um comportamento diferente do que uma peça convencional. Por exemplo, se a espessura e inconstante, enquanto a força de sujeição do prensachapas é constante, rugas ou rupturas podem ocorrer na peça conformada. Nesses casos prensachapas especiais devem ser projetados de acordo com o produto a conformar, para utilizá-los no ferramental de conformação.

A Figura 12 mostra um prensa chapas modificado para a estampagem profunda de Blanks Soldados [MtlFrmHdbk98]. Na Figura 13 apresenta-se esquematicamente o processo de fabricação do Blank Soldado.

2.2.3 VANTAGENS DO BLANK SOLDADO

Como potenciais vantagens do uso do blank soldado pela indústria automobilística podem ser citadas as seguintes:

- Redução do peso da carroceria, uma vez que maiores espessuras só são usadas onde é necessário (com função estrutural - recebem impacto ou carga)

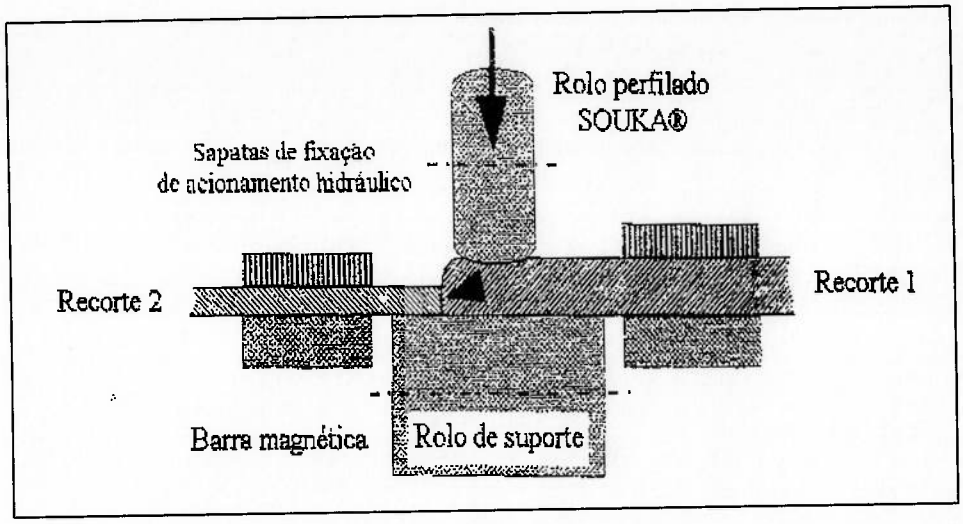


Figura 7 – Dispositivo de preparação de bordas – "SOUKA" [Pincinini99]

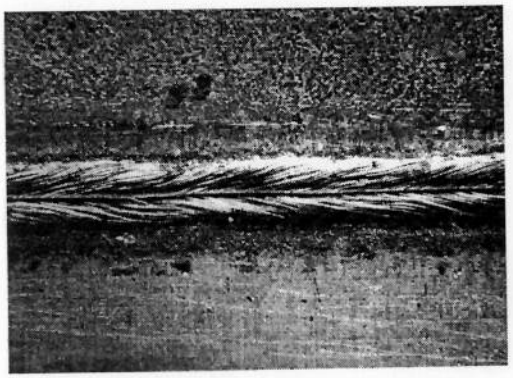


Figura 8 – Aparência visual da raiz da solda, apresentando uma raiz com bom acabamento. - Aumento 10X. [Pincinini99]

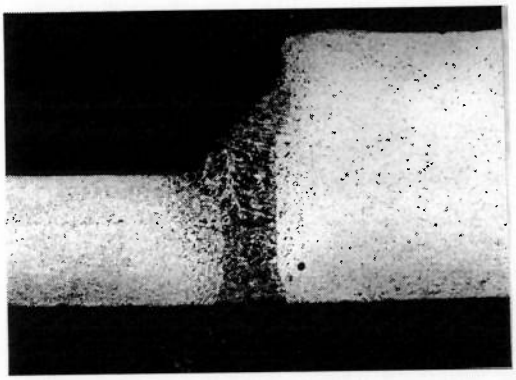


Figura 9 – Solda de qualificação de material BH com espessuras de 1,20x2,50 mm Aumento 25X – Nital 2% - Concavidade 6,3 % [Pincinini99]

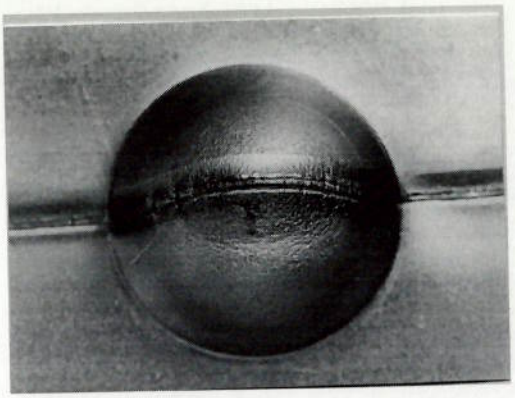
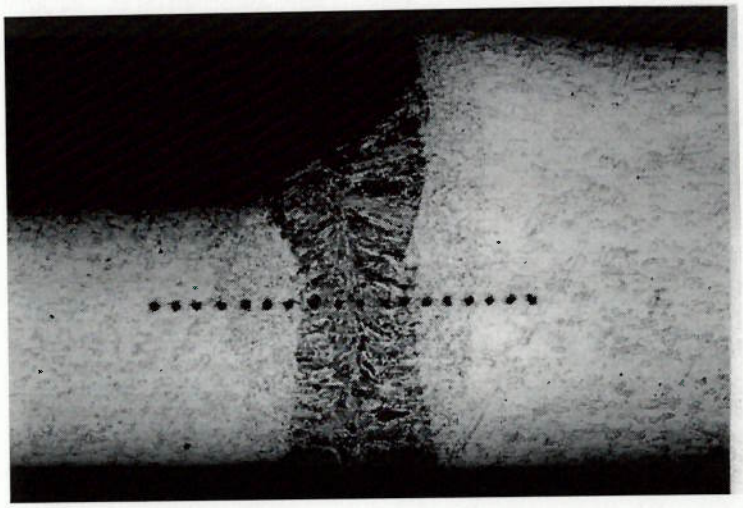
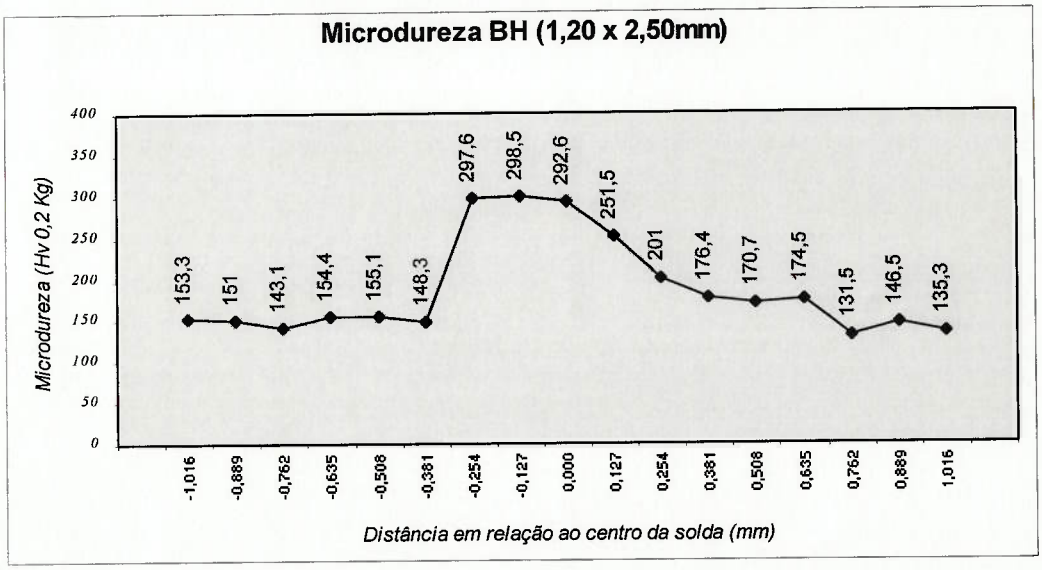


Figura 10 –Resultado do teste de embutimento, com trincas superficiais na solda e início de deformação localizada no material base. Laudo O.K. [Pincinini99]



(a)



(b)

Figura 11: (a) – Ensaio de microdureza (micrografia); (b) Curva de microdureza [Pincinini99]

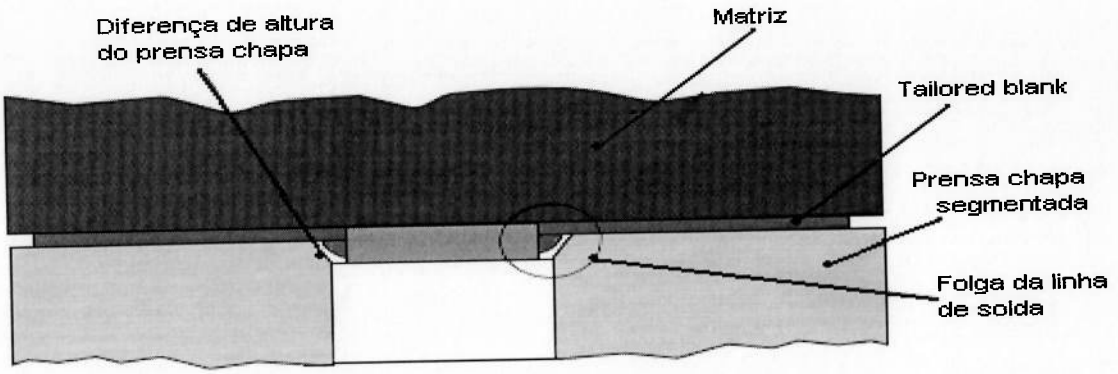


Figura 12: Prensa-Chapas modificado para estampagem de Tailored Blanks [MtlFrmHdbk98]

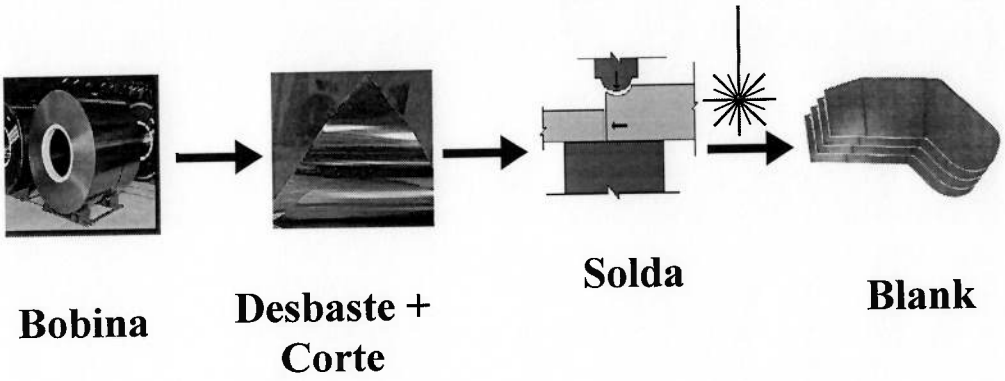


Figura 13: Processo de fabricação de Blank Soldado.

- Otimização do desenho da peça, permite compor blanks de materiais zincados com laminados a frio ou aço de alta conformabilidade com outro de maior resistência mecânica Redução do número de peças de reforço;
- Melhoria da resistência no teste de impacto.
- Melhoria da rigidez
- Redução do numero de componentes com a redução do número de soldas a pontos, consequentemente, também se obtém a redução a susceptibilidade a corrosão
- Sucata: eliminação da perda de grandes vãos (áreas para vidros e portas), através de um plano de montagem de partes adequadas.
- Logística: reduz estoque, diminuindo o número de peças, no caso em que antes eram peças separadas e passou a formar um blank único.
- Produtividade: reduz o numero de operações de estampagem em prensa, diminuindo o tempo de produção em prensa e montagem no carro, e reduzindo estoque intermediário antes da montagem.

2.2.4 Desenvolvimento do BLANK Soldado no Brasil

No desenvolvimento do blank soldado no Brasil podem ser destacadas as seguintes etapas:

- Em 29-01-96 foi acusado o recebimento na CSN dos primeiros desenhos e especificações técnicas dos materiais que iriam compor o blank soldado.
- Em 29-08-96 iniciou-se os entendimentos comerciais
- Entre agosto 97 e novembro 97 iniciou-se o envolvimento da Assistência Técnica e a composição do grupo de trabalho da CSN.
  - Em janeiro 98 chegou ao Brasil a Maquina de Solda a Laser (Fabr.: Soudronic Neftenbach AG; Mod.: LCL 2000) , adquirida pela Ferrolene.
  - Em junho 98 iniciou-se a introdução do blank nacionalizado (CSN/Ferrolene) na Ford.

As perspectivas do uso da tecnologia do blank soldado no futuro são otimistas porque está consolidada na atualidade, a união de materiais com especificações e espessuras diferentes que permitem a redução do peso do veículo, o aumento da sua rigidez e segurança já se faz presente em todos os novos modelos de carros a serem laçados nos próximos anos pelos fabricantes no Brasil e no mundo enteio.

1. Painel lateral interno;	2. Painel dianteiro do assoalho;
3. Reforço do Pilar "B" E/D;	4. Ossatura de porta;
5. Travessa frontal;	6. Longarinas;
7. Reforço do montante;	8. Painel interno tampa traseira.
9. Assoalho central;	

Tabela 2: Peças cuja concepção em Tailored Laser Blank serão produzidas no futuro [Pincinini99]

Um exemplo é mostrado, baseado numa base de dados dedutiva orientada a objetos chamada RLOG da Universidade de Regina no Canadá [Liu98-a], que possibilita a formalização do gerenciamento da informação e sua semântica através de cláusulas de Horn. Embora uma conexão completa entre a representação estática e

dinâmica não seja escopo desse trabalho, o exemplo prático apresentado é baseado num caso real de pedidos de compra de produtos com manufatura terceirizada.

Manufatura de blanks sob medida e blanks soldados é uma tecnologia em crescimento em processos de conformação de chapas, e a utilização de Sistema de Informação tipo ERP (Enterprise Resource Planning) visa aumentar a produtividade da fabricação. Entretanto, essa mesma técnica de aumento de produtividade pode ser revertida em desperdício de produto e recursos se uma modelagem correta do Sistema de Informação não for realizada.

O Capítulo 3 apresenta um histórico sobre projeto de sistemas automatizados, o Capítulo 4 trata da modelagem de sistemas automatizados, o Capítulo 5 descreve o modelo AV/AO, o Capítulo 6 apresenta o estudo de caso da otimização da informatização da administração de materiais na fabricação de blanks, e o Capítulo 7 mostra a discussão e conclusão sobre o trabalho.

Capítulo 3

Projeto de Sistemas Automatizados

A complexidade dos problemas encontrados pelas organizações, tais como mudanças tecnológicas, competitividade do mercado e os impactos causados por decisões governamentais, torna impossível a procura por soluções isoladas. É preciso ver o problema a partir de uma visão do sistema como um todo [Bresciani97-a].

Uma organização é um sistema complexo, não-linear e dinâmico, assim como qualquer operação de negócios envolvendo clientes, fornecedores e competidores [Leach96]. Diversos autores [Schoderbek90], [Murdock94], [Hardie95], [Melan88], [Snee93], [Hutchison94], [Wood93] apud [Kintscher98] têm tratado da necessidade da utilização da "sistêmica", ou seja, de métodos de tratamento de problemas complexos em diferentes atividades administrativas e tecnológicas.

A "sistêmica" define um novo método para estudos de sistemas complexos. Este método necessita de novas ferramentas para a concepção e modelagem de sistemas complexos em contraposição às ferramentas tradicionais baseadas no método cartesiano. O que se torna prioritário é o desenvolvimento de ferramentas que possam permitir a compreensão do comportamento do sistema complexo e, portanto, a solução do problema, através da construção, se possível, de um modelo matemático. [Lemoigne90] apud [Kintscher98] desenvolveu uma nova representação de sistemas complexos chamada sistemografia, que permite a representação gráfica de um sistema através de uma ferramenta chamada sistemógrafo. A função do sistemógrafo é construir um modelo de representação de um fenômeno percebido como complexo.

O objetivo deste capítulo é analisar os principais conceitos de "sistêmica" e apresentar o sistemógrafo. A proposta é a utilização desta ferramenta na análise da área de administração de materiais em uma empresa siderúrgica.

3.1 Teoria Geral de Sistemas e Teorias Específicas de Sistemas

A TGS (Teoria Geral de Sistemas) surgiu com o objetivo de compreender o comportamento dos sistemas complexos em termos da relação existente entre os seus diversos componentes e o meio ambiente, sendo uma teoria criada na tentativa de modelagem de sistemas abertos (que interagem com o meio ambiente), em contraposição ao método analítico tradicional (tratamento de sistemas fechados) [Schoderbek90].

Além da TGS, outras teorias de sistemas também foram desenvolvidas, das quais pode-se destacar a Cibernética, a Engenharia dos Sistemas, a Pesquisa Operacional e a Análise de Sistemas (Figura 14).

A cibernética é a ciência do controle ótimo e intencional aplicável aos sistemas complexos presentes na natureza e na sociedade. A cibernética defende a auto-regulação como ferramenta de controle de um sistema, ou seja, o sistema é controlado de forma a manter um estado desejado através da realimentação de informação de seu estado atual (controle em malha fechada).

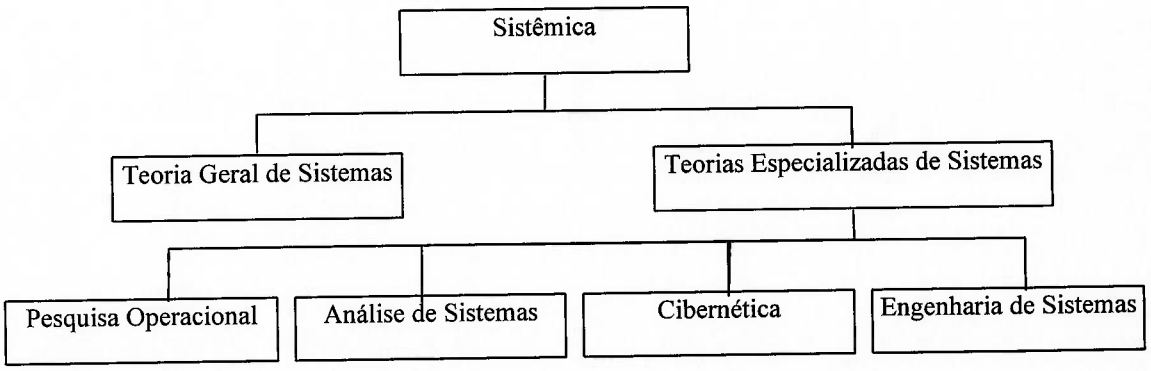


Figura 14 - Principais Teorias de Sistemas [Schoderbek90].

A engenharia de sistemas é definida pelas atividades de pesquisa, projeto, produção e assistência técnica de forma integrada com a finalidade de se atingir a realização ótima da missão do produto. A engenharia dos sistemas introduziu as noções de sistemas, subsistemas, objetos, relacionamentos, meio ambiente e outros conceitos tradicionais da teoria de sistemas. Seu principal conceito é que o objeto final deve ser projetado a partir da visão total do sistema.

A pesquisa operacional pode ser definida como a aplicação de métodos quantitativos da física e matemática para as operações representadas pelas atividades de homens e máquinas. A pesquisa operacional disponibiliza um corpo de técnicas computacionais que são utilizadas na tentativa de simulação de modelos de situações reais da vida do sistema.

A análise de sistemas é uma combinação de técnicas obtidas da área da engenharia e de outras disciplinas para a definição e o desenvolvimento de algoritmos necessários para resolução de um problema, particularmente de algoritmos que possam ser resolvidos por meios computacionais (computáveis).

Ao contrário das teorias de sistemas citadas acima que podem ser classificadas como casos particulares, a TGS é uma teoria que pode ser utilizada em todas as áreas do conhecimento. Essa disciplina vem sendo denominada mais recentemente de "sistêmica". Foi a partir da TGS ou da "sistêmica" que Le Moigne [Lemoigne90] apud [Kintscher98] concebeu e desenvolveu a sistemografia como modelo de representação de sistemas complexos.

3.2 Algumas Definições Básicas Relativas aos Sistemas

As definições são apresentadas abaixo:

3.2.1 Sistema

Um sistema é definido como um conjunto de objetos com relacionamentos entre eles seus atributos e seu contexto [Schoderbek90].

O sistema também pode ser definido como uma entidade unitária global, de natureza complexa e organizada, constituída de um conjunto de elementos que mantêm relações e realizam ações.

Os objetos do sistemas são considerados as partes, os componentes, os atores ou os agentes que realizam ações (bem como reações, retrações, proações e transações), conduzem processos e operações, produzem fenômenos, exercem atividades e são responsáveis por transformações, conversões e eventos que caracterizam os seus comportamentos. Os objetos podem ser classificados em três grupos principais: objetos de entrada, objetos dos processos de transformação interna e objetos de saída. Além disso, os objetos possuem características, propriedades, atributos, predicados e qualidades .

3.2.2 Organização e Complexidade

A organização é o conjunto das características estruturais e funcionais de um sistema, a qual representa as relações e as ações desse sistema e tem a capacidade de transformar, produzir, reunir, manter e gerar os comportamentos desse sistema. A estrutura de um sistema é o conjunto articulado de relações entre os seus objetos e pode ou não se constituir em um invariante desse sistema. O funcionamento de um sistema é conferido pelo conjunto articulado de atividades dos objetos; esses objetos conduzem o processo de transformação exercendo funções de forma dinâmica, mas condicionada por sua estrutura (relações).

Um sistema complexo é caracterizado pela imprevisibilidade de seu comportamento que é provocada pela ação conjunta e aleatória de fatores internos e externos. Pode ser compreendido com a simulação de seu comportamento, realizada através da observação do todo. O grau de complexidade de um determinado sistema pode ser identificado através do levantamento do número de seus elementos, do número de inter-relações entre os elementos, dos seus atributos e do seu grau de organização[Bresciani97-b].

3.2.3 Finalidades e Contexto

Os sistemas podem ter objetivos. Esses objetivos precisam ser explicitados e compreendidos, para que um dado conjunto de elementos da importação, processados pelo sistema, possam fornecer um conjunto de objetos de exportação compatível com esses objetivos.

A eficácia de um sistema é a expressão da sua capacidade em atingir um determinado objetivo. A eficiência de um sistema é a expressão da intensidade com que esse sistema atinge um determinado objetivo. A função de um sistema é a ação desenvolvida para atingir um objetivo.

O contexto é tudo aquilo que se convenciona que fica fora do sistema; dessa forma, determinam-se os limites de um sistema. Mas um sistema não é completamente isolado ou fechado do seu contexto, pois tudo o que entra ou sai do sistema vem do, passa por ou sai para o contexto, sendo a fronteira o lugar onde se dá a proibição ou a autorização da passagem.

Em decorrência da existência da fronteira, pode-se identificar os elementos internos e os externos ao sistema os quais mantêm relações entre si e podem exercer influências mútuas. Desse modo, pode-se distinguir uma categoria especial de elementos internos ao sistema, denominados elementos de fronteira que têm por incumbência estabelecer as relações do sistema com o contexto, sendo as responsáveis pelas entradas e saídas do sistema.

3.3 Representação de Sistemas Complexos – Sistemografia

[Schoderbek90] oferece um modelo de representação de um sistema complexo, conforme apresentado na figura 15.

Nota-se primeiramente que a entrada para um sistema é a saída para outro sistema, e que esta saída torna-se a entrada para outro sistema; desse modo é introduzido o conceito de realimentação do próprio processo. Nota-se também que a linha de demarcação do ambiente do sistema (isto é, o limite do sistema) não é sólida. Há duas razões para isso: primeiro, tal como a linha indica, há uma mudança contínua de informações entre o sistema e seu ambiente; a linha tracejada indica que as posições do limite do sistema são determinadas mais ou menos arbitrariamente pelo projetista, investigador ou observador da organização do sistema.

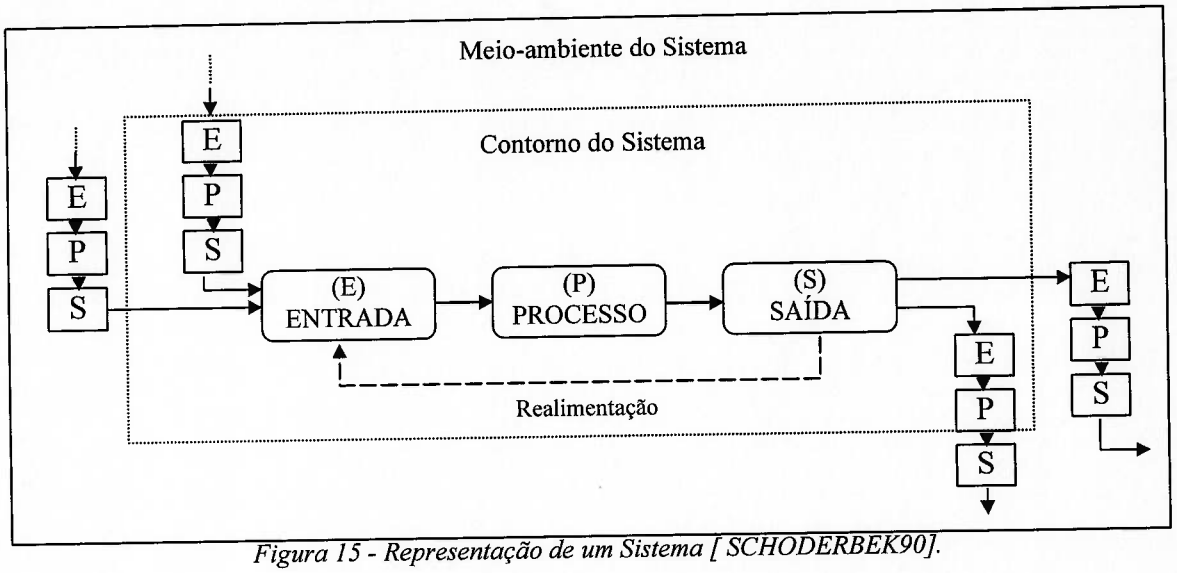


Figura 15 - Representação de um Sistema [SCHODERBEK90].

Pode-se aperfeiçoar esta representação através do sistemógrafo, que permite um maior detalhamento do modelo apresentado, principalmente em relação à classificação e descrição dos elementos que compõem um sistema. Ao invés de apenas diferenciar os elementos por elemento de entrada, de processo e de saída, propõe-se classificá-los por família (objetos processados ou objetos processadores), por categoria (operacional, informacional e decisional), por tipo (forma, espaço e tempo) e por nível de complexidade (1^o a 9^o):

Classificação por família

O elemento é representado pelo que ele faz (processador) ou pelo resultado de algo que foi feito nele (processado) . O elemento pode ser considerado como uma "caixa-preta", com a sua constituição interna não definida e, portanto, não necessária para sua compreensão.

Classificação por categoria

Os objetos (elementos) de processo constituem sistemas que, por sua vez, podem ser classificados em três categorias: sistema operacional, sistema informacional e sistema decisional. O sistema operacional é constituído pelas ações desenvolvidas e dos respectivos mecanismos de acionamento. O sistema informacional é constituído pelas informações disponíveis e seus mecanismos de acesso utilizadas no desenvolvimento das atividades. O sistema decisional é constituído pelo conjunto de decisões tomadas a partir das informações para realização das atividades.

Classificação por tipo

O objeto deve ser classificado pelos tipos: tempo, espaço e forma. A definição do tipo do objeto deve ser feita a partir das alterações no comportamento do objeto em relação aos referenciais tempo (se o tempo gasto para executar o processo foi relevante), espaço (se houve um deslocamento físico do objeto no processo) e forma (se o processo alterou ou converteu o objeto). Um determinado objeto pode ainda ser classificado como de um ou mais tipos em relação a esses três referenciais.

Classificação por nível de complexidade

O objeto deve ser classificado em um dos nove níveis progressivos de complexidade. Apresenta-se a seguir a classificação adaptada de [Bresciani97-a] a partir do texto de [Lemoigne90] apud [Kintscher98], seguido de exemplos retirados deste trabalho.

1º NÍVEL - Objeto Passivo: o objeto é inerte e não exerce qualquer processamento. Exemplo: alguma área da empresa recebe um documento e simplesmente repassa-o para outra sem qualquer verificação ou processamento.

2º NÍVEL - Objeto Ativo: o objeto processa, realiza e exterioriza um comportamento. Exemplo: todas as atividades que tenham algum tipo de processamento.

3º NÍVEL - Objeto Regulado: o objeto também processa, realiza e exterioriza um comportamento, porém, manifesta uma efetiva regularidade na sua atividade. Exemplo: verificação de dados de uma nota fiscal.

4º NÍVEL - Objeto Informado: o objeto também processa, realiza e exterioriza um comportamento de forma regular, porém, utilizando da informação. Exemplo: baixa de materiais no estoque.

5º NÍVEL - Objeto com Decisão: o objeto tem capacidade de tomar decisão com base em uma informação que provoca uma ação predefinida e conhecida; a representação é feita, pelo menos, com um processador decisional (detentor do projeto e finalidade do objeto). Exemplo: processo de autorizar uma compra de materiais no almoxarifado.

6º NÍVEL - Objeto com Memória: o objeto além de tomar decisão apoia-se em um processo de memorização; a representação é feita com processador decisional. Exemplo: consulta a área de compras de materiais para verificar procedimentos a serem tomados em relação a um fornecedor.

7º NÍVEL - Objeto com Pilotagem: o objeto (sistema geral) se articula segundo 03 subsistemas agregados e fundamentais: decisional, informacional e operacional; o sistema interno de pilotagem (que engloba coordenação) é de natureza hierarquizada no qual o processador decisional deve ter a capacidade de coordenação que implica: a capacidade relacional (ou seja, número de outros processadores com os quais se conecta) e a capacidade de tratamento de informação (no caso de seres humanos e a capacidade cognitiva); a representação pode ser complexa com cada subsistema contendo processadores conectados aos demais subsistemas. Exemplo: decisões rotineiras tomadas pela direção de uma empresa.

8º NÍVEL - Objeto com Inovação: o objeto tem a capacidade de inovação (imaginação, seleção, concepção, criação e invenção) de gerar informação simbólica, de aprendizagem, de inteligência, e de se auto-organizar. Exemplo: Decisões inovadoras tomadas pela direção de uma empresa.

9º NÍVEL - Objeto com Auto-Finalização: o objeto passa a ter no seu sistema de pilotagem um subsistema de finalização que lhe dá a capacidade de gerar os seus próprios objetivos e de ter consciência da sua existência e identidade, esse objeto no seu sistema de pilotagem engloba o sistema de diagnóstico e, no seu sistema de operação, o sistema de manutenção. Exemplo: decisões autônomas de estabelecimento de políticas tomadas pela direção de uma empresa.

3.4 Projeto em Engenharia

A filosofia clássica do projeto de engenharia, proposta por Asimow [Asimow68], baseia-se em princípios e conceitos que resultam das experiências coletivas da humanidade, sendo a escolha dos princípios, ou sua formulação, influenciada por experiências pessoais e idiossincrasias.

Os princípios devem formar um conjunto consistente que possa ser ampliado por combinação e extensão lógica onde o projeto possa se fundamentar com segurança.

Para essa filosofia ser útil ela deve incorporar uma metodologia onde os princípios possam ser aplicáveis de forma disciplinada. Além disso, a filosofia deve incluir um esquema de avaliação que oriente e possibilite a formulação de critérios específicos de valor.

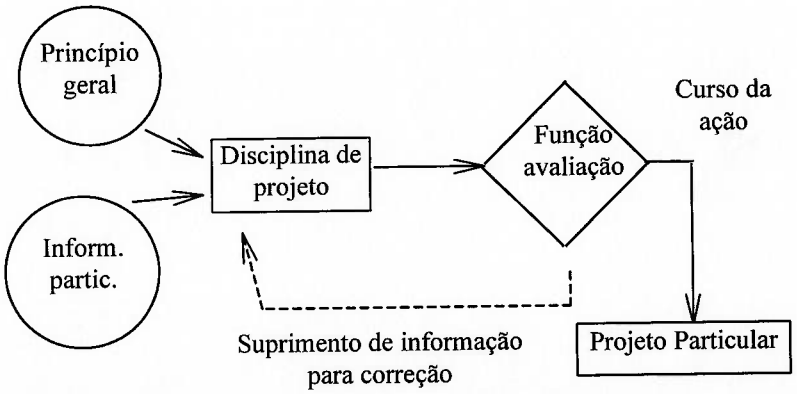


Figura 16. Filosofia do Projeto Clássico [Asimow68].

A Figura 16 representa esquematicamente a idéia da filosofia do projeto clássico de forma abreviada e simplificada. O envio de informações por retrocesso, para correção torna-se operável quando a solução é considerada inadequada e exige aperfeiçoamento.

Cada projeto [Asimow68] tem uma história individual que lhe é pertinente em todas as suas peculiaridades. Não obstante, à medida que um projeto é iniciado, e desenvolvido, desdobra-se uma seqüência de eventos, numa ordem cronológica, formando um modelo, o qual quase sempre é comum a todos os projetos.

Sobre essa filosofia foi criado um modelo geral onde o projeto é separado nas seguintes fases principais: estudo de exeqüibilidade; o projeto preliminar; o projeto detalhado; o planejamento do processo de produção; o planejamento da distribuição; o planejamento do consumo; e o planejamento da retirada do produto do mercado.

De um modo geral, as três primeiras etapas podem ser definidas como fase preliminar do projeto, onde grandes mudanças podem ser realizadas sem grandes perdas financeiras; enquanto as quatro etapas finais estão relacionadas com o ciclo de produção/consumo que envolve um compromisso econômico.

Cada etapa indicada no esquema da morfologia do projeto possui uma seqüência típica de operações, variando conforme a etapa. Denomina-se de estrutura vertical a morfologia de projeto e de estrutura horizontal a seqüência típica de cada etapa.

Para Asimow, a análise de exeqüibilidade é a etapa onde há a resolução do problema e a avaliação das soluções frente aos meios, custos e problemas organizacionais pertinentes. É elaborada a análise de atividades esquematizado na Figura 17. Os resultados desejáveis são derivados das necessidades do cliente final; os resultados indesejáveis são deduzidos quase inevitavelmente dos resultados desejados; os elementos de alimentação que o sistema transformará em resultados, são deduzidos das características do sistema, onde esses elementos podem ser classificados em cinco categorias: físicos (energia), humanos (esforço, controle, supervisão), informativos (sinais, dados), econômicos (custo) e relativos ao ambiente (calor, choque, umidade).

3.4.1. Processo de Projeto

O projeto em engenharia é um processo especializado de resolução de problemas. Embora apresente sua própria maneira, adaptada a um modelo tecnológico, seu processo assemelha-se ao de resolução de problemas.

3.4.2 Processo geral para resolução de problemas

Muitas espécies de palavras têm sido utilizadas para definir os passos principais da resolução de problemas.

Na morfologia viu-se que a estrutura vertical de um projeto de engenharia era um esqueleto, em redor do qual se podia planejar, organizar e desenvolver um projeto. Agora assinala-se que um projeto de engenharia tem analogamente uma estrutura horizontal. Analisa-se a estrutura horizontal, algumas vezes de modo completo, outras vezes somente em parte, em todos os passos da morfologia, à medida que se desce na sua estrutura vertical, partindo da análise das necessidades do estudo de exeqüibilidade até o passo final de revisão do projeto detalhado. Cada um desses passos possui, em forma quase completa, uma típica seqüência de operações. É esta seqüência que chamamos de processo de um projeto. É um processo para resolução de problemas de projetos de engenharia, do mesmo modo que o método científico é um processo para resolver os problemas de pesquisas. Visto que cada passo na morfologia contém um problema particular que deve ser resolvido, é claro que o processo de projetos deve aparecer em cada passo.

O processo do projeto clássico descreve a reunião, o manuseio e a organização criadora de informações relevantes à situação problema; prescreve a derivação de decisões que são otimizadas, comunicadas e testadas, ou mesmo avaliadas; tem um caráter interativo, pois, freqüentemente, na ação, surgem novas informações ou ganha-se nova compreensão, as quais exigem a repetição das operações anteriores. Algumas das operações são qualitativamente lógicas em caráter, como o raciocínio a partir de proposições verbais; algumas são baseadas em avaliações verbais, como na comparação

e combinação de valores diversos; muitas são sensíveis a análises quantitativas e a aplicações em computadores, como na otimização da representação analiticamente formulada de uma solução de problema. A maior parte das técnicas associadas com cada operação no processo de projetos é de tão grande generalidade que sua utilidade não se limita a nenhum caso particular.

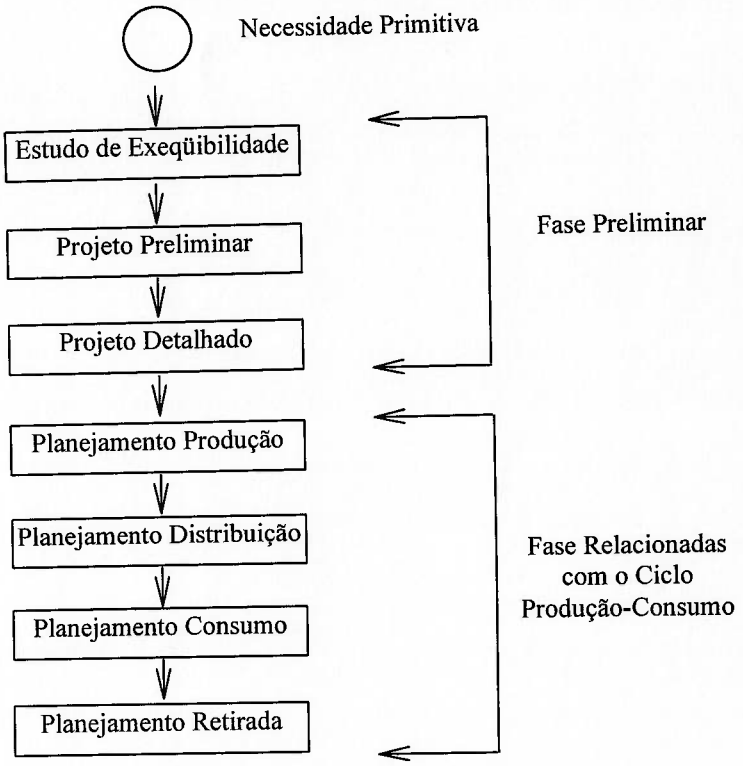


Figura 17 Morfologia do Projeto Clássico [Asimow68].

Assim o processo de projeto assemelha-se ao processo geral de resolução de problemas (GPS) quanto as características principais. Porém ele utiliza mais aguçados e, na maioria das vezes, mais analíticos instrumentos, que foram especificamente talhados e afiados para problemas de projeto em engenharia., inspirando vários métodos de formalização e automação do processo de projetos. Ele conduz o processo através da análise, síntese, avaliação e decisão, estendendo aos domínios da otimização, revisão e execução.

Os problemas raramente vêm já prontos com uma proposição apurada e clara dos fatores envolvidos e com, pelo menos, alguns indícios bem assinalados para indicar a

solução correta. De fato, geralmente não é claro se há um só ou vários problemas e, se houver vários, quais serão eles. O projetista se depara, não com um problema, mas com uma situação problema, situação esta que pode possuir muitos elementos intrincados, interrelacionados em formas complicadas e obscuras. É deste meio de confusão que definições claras dos problemas relevantes devem ser retiradas.

Quando há uma situação problema, usualmente verifica-se que pouco se sabe sobre os fatores que a compõe. Tenta-se focalizar os objetivos, mesmo embora eles tenham que ser revisado à luz de futuras evidências. Explora-se os conhecimentos sobre a situação, no que concerne aos objetivos e se identifica os tipos de informações que se necessita e as respectivas fontes, onde quer que se possa obtê-las. A informação é um artigo econômico, tem um custo; e a quantidade que se precisa é a que se exige do que se julga como fatores relevantes.

Logo que se adquire um entendimento da situação problema, habilita-se então a ver as reais dificuldades e se pode começar a formular as questões exatas.

As questões aleatórias podem finalmente ser agrupadas numa forma que constitua a proposição do problema. Essa proposição deixa claro quais os objetivos devem ser atingidos, que dificuldades devem ser superadas, que recursos existem, que limitações podem circunscrever qualquer situação aceitável e que critério deve ser usado para julgar-se a excelência de uma possível solução.

Através da proposição do problema começa-se a procurar soluções. A solução é uma síntese dos elementos componentes, que contorna as dificuldades restritivas e, sem exceder os recursos disponíveis nem desrespeitar os limites estabelecidos pelas limitações, atinge os objetivos prescritos. A maioria dos elementos componentes acumulam-se nas memórias dos projetistas, que aumenta seu "cabedal" à medida que suas experiências ampliam-se e suas habilidades de "ver" e "ouvir" o "potencialmente útil" se aguça. Os elementos podem ser de idéias ou de coisas físicas. Não existe ainda nenhuma forma definida, segundo Asimow, que habilite o projetista a retirar de seu "cabedal de experiências", justamente o conjunto exato de elementos e agrupá-los na combinação exata, de modo que possam ter um sentido de adaptação à situação.

O que caracteriza o projeto como empreendimento é o passo da síntese ou solução. Esse, mais que qualquer outro passo, exige esforços de criação e de invenção. O poder da criação é portanto um ingrediente essencial em um projeto de engenharia. Nenhum dos elementos individuais que compreendem a síntese necessita ser novo ou original. O desenvolvimento de elementos novos e originais é mais um objetivo da pesquisa do que de projeto. Após a descoberta de novos princípios, pelo pesquisador, a tarefa de desenvolver novos componentes vantajosos, possíveis, pela primeira vez, pelo novo princípio, é freqüentemente designada para o projetista. Em assim fazendo, ele deverá combinar, geralmente, o novo princípio com vários princípios existentes, a fim de conseguir os resultados finais.

Um projeto progride do abstrato para o concreto. Começa com uma concepção conjurada da mente; uma relação entre idéias ou formas geométricas que, de algum modo, amoldam-se às circunstâncias do problema. Tais abstrações mentais podem eventualmente manifestar-se em objetos físicos porém, a "ponte de ligação" é muito longa e o primeiro passo sobre ela é transformar a idéia original numa expressão comunicável. Conseguiu-se isto descrevendo a idéia em palavras, em ilustrações gráficas e em símbolos matemáticos. O simples ato de dar expressão a uma imagem mental, dota-a com certa essência da realidade e cria, a partir dela, um objeto que pode ser manipulado e combinado com outros. A concepção torna-se informação, a qual pode ser, em si mesma, verdadeira. A informação pode ainda ser abstrata, mas está a um passo mais próxima do concreto.

Uma única descrição, simbólica ou analítica, da concepção do projeto pode ser, e usualmente é, inadequada para retratar as várias facetas do seu comportamento. Portanto, são exigidas várias descrições, cada uma levando em consideração um aspecto físico e econômico; assim, é preservada a simplicidade do sistema tanto quanto possível. O conjunto de descrições é manipulado de uma maneira abstrata, como se elas fossem o objeto real, sendo os resultados assim obtidos atribuídos a situação real.

Por essa razão é que devem ser dadas formas simbólicas ou verbais às soluções, de modo a poderem ser manipuladas pelas regras da lógica e da matemática. Desta

maneira, pode-se testar a solução no abstrato, contra o espectro das exigências previstas, chegando assim a uma avaliação da mesma. De posse das avaliações, toma-se a decisão sobre qual solução adotar, levando em conta de que qualquer solução particular resulte em desfavorável.

[Asimow68] constatou a necessidade de um formalismo para a representação de projetos. Pelo fato da síntese realizar-se, principalmente, com princípios antigos, já conhecidos, em conjunto com algum princípio novo já desenvolvido, busca-se um formalismo que represente esses conhecimentos possibilite a remoção/ modificação/ adição de outros princípios e construa uma base que armazene as sínteses anteriores.

Dessa forma há possibilidade de um desenvolvimento de um sistema que auxilie a reutilização de projetos.

3.4.3. Suporte Computacional para o Processo de Projeto

O projeto inicia-se com um conjunto de requisitos de produtos. O projetista transfere o requisito de produtos para dentro de um conjunto de especificações funcionais. Então, as especificações funcionais são realizadas pela modelagem conceitual, que contém as relações de instalação e restrições entre representação de componentes significantes funcionalmente.

O suporte computacional para o processo pode ser top-down ou bottom-up. O ambiente top-down gera primeiro uma representação funcional do projeto. Essa representação pode ser na forma de um sistema de equações de restrições ou uma representação de projeto implícito que corresponde à alguma funcionalidade [Dieter91], [Gero89], [Pahl86], [Reich95]. A Figura 18 nos mostra o processo top-down.

No ambiente de projeto bottom-up, em contrapartida, o projetista desenvolve um modelo mental do projeto. A representação detalhada, correspondente a componentes individuais modelada pelas ferramentas existentes em ambientes computacionais. Só então as relações de agrupamento são definidas. A Figura 19 apresenta o processo bottom-up.

Devido ao fato da funcionalidade não ser representada num nível de componentes, o sistema de projeto bottom-up deixa a responsabilidade de manter a funcionalidade para o projetista.

A função de colocar esses dois itens, bottom-up e top-down, é justificar a necessidade de um módulo híbrido para o design de sistemas de automação e a necessidade de se ter uma representação estruturada formal, para esse modelo.

3.4.4 Requisitos para suporte computacional do projeto

Para gerar suporte a representação funcional de projeto, um sistema computacional deve auxiliar o projetista na realização de especificações funcionais através de ferramentas compatíveis com o vocabulário do projeto. Funcionalidade é raramente o resultado de ação de componentes únicos, o que implica que as ferramentas de modelagem têm que ser baseadas em conjuntos de componentes. Assim o modelo do conjunto será utilizado como uma superestrutura que representa a funcionalidade e os componentes a serem modelados.

O suporte computacional também deve assegurar que a funcionalidade seja mantida. Como um invariante do projeto várias estimações podem sugerir mudanças na característica dos componentes, mas tais modificações não devem violar a funcionalidade do modelo.

Como o modelamento da funcionalidade não precisa de uma representação com descrição completa, pode-se criá-la de forma abstrata e apenas detalhá-la no final do projeto.

O ambiente de projeto deve servir de estrutura para integração para diferentes aspectos de ciclos de vida de produção.

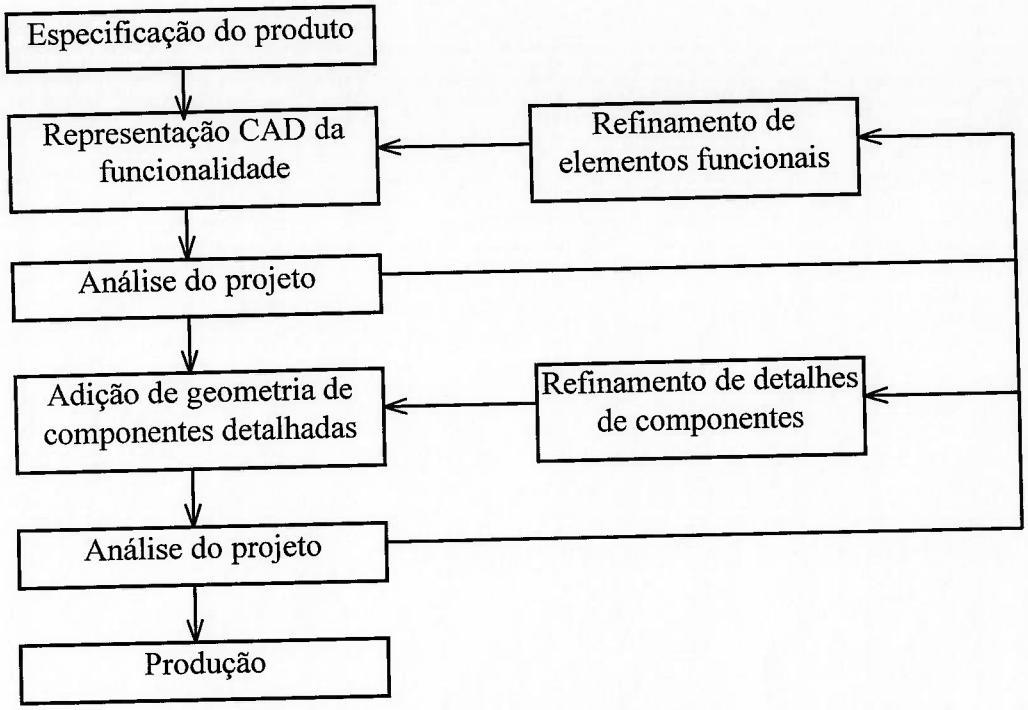


Figura 18. Representação esquemática do projeto Top-Down.

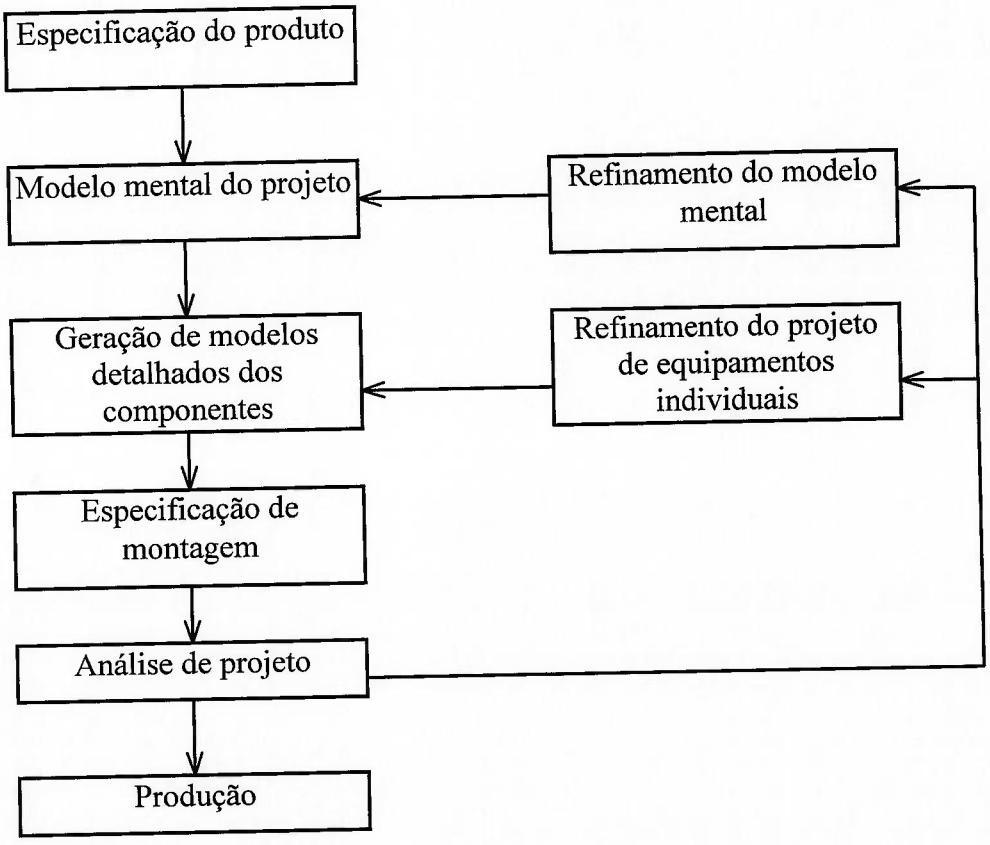


Figura 19. Representação esquemática do projeto Bottom-Up.

Em sistemas bottom-up para definição de produtos, o ambiente computacional permite ao projetista criar representações utilizando primitivas como base e se utilizar de operações de Boole nas mesmas. Não há suporte para verificar a funcionalidade. Assim, a reutilização de projetos baseados nessa estrutura é caracterizada por um conjunto de modelos primitivos que não são naturais para o processo do projeto e esta reutilização tem dificuldade para manter as restrições de funcionalidade entre os elementos.

A maioria dos projetos em engenharia tem características funcionais e estão relacionados com métodos top-down.

Esse trabalho busca uma estrutura de representação onde a funcionalidade pode ser implicitamente ou explicitamente utilizada como parâmetro de projeto. Com essa estrutura pode-se buscar componentes com mesma funcionalidade e, uma vez encontrado um candidato, reutilizar sua estrutura como base para elaboração daquela função, ou seja, reutilizar a estrutura de forma bottom-up, como se esta fosse um elemento da base de objetos que compõe o projeto.

As metodologias de desenvolvimento, tanto clássicas quanto moderna, se dirigem para o design em engenharia, ou seja, para o chão de fábrica ou sistema operacional. Com o advento da automação, os processos clássicos deram lugar a métodos modernos baseados em sistemas inteligentes, como é o caso da orientação à objeto.

Portanto a dicotomia entre o sistema operacional (chão de fábrica) e o sistema decisional (gerencial) ampliou-se muito. Esse trabalho propõe assim a interseção destes através da modelagem formal do sistema informacional, sobre o qual as decisões gerenciais são fundamentadas, assim como a previsibilidade e repetibilidade da parte operacional. O formalismo utilizado é baseado no modelo AV/AO e na Teoria de Metáforas [Gerric89], [Indurkia87], [Indurkia86] descritos nos Capítulos 6 e 7.

Capítulo 4

Modelagem de Sistemas Automatizados

Como discutido no Capítulo 1, o objetivo deste trabalho é propor uma formalização para o sistema de informação, para atuar como elemento de ligação entre o sistema operacional e o sistema decisional. Entretanto, antes de se tratar a base conceitual sobre a qual este sistema de informação será representado é preciso definir com maior clareza o nosso objeto de estudo que são os sistemas automatizados, notadamente os sistemas de manufatura.

Sistemas automatizados são sistemas complexos compostos por vários subsistemas ou módulos especializados, o que pressupõe uma divisão do sistema original em vários subsistemas, criando uma estrutura complexa onde as relações entre os subsistemas concentram a funcionalidade do sistema global integrado [Silva96]. Sendo assim, deve-se definir formalmente integração de subsistemas e como subsistemas cuja funcionalidade pode variar, se relacionam entre si. São chamados de sistemas flexíveis aqueles que podem variar sua funcionalidade conforme a necessidade existente, bem como de módulos funcionais o conjunto de funcionalidades de um dado sistema flexível.

4.1. Sistemas Integrados e Flexíveis

Seja M_1, M_2,..., M_n o conjunto de módulos funcionais independentes acoplados que compõe um grande sistema M. Cada módulo tem uma única funcionalidade igual a F_i, i=1,.., *n*. Esses módulos são ditos integrados se e somente se existe pelo menos uma

funcionalidade que seja composta de uma função $\Phi(M_1,.., M_k)$, $k \leq n$, sobre o conjunto de módulos.

$\Phi(M_1,.., M_k)$ pertence a um sistema fracamente acoplado se e somente se Φ é composta pela simples composição de módulos funcionais, a saber:

$$\Phi(M_1,..,M_k) = \bigcup_{i=1}^{k} F_i \qquad (1)$$

Similarmente, $\Phi(M_1,.., M_k)$ pertence um sistema com integração funcional se for o resultado de composição e também do compartilhando de dados e informações entre domínios.

A definição formal de sistema integrado é necessária para que se possa estabelecer uma forma de inter-relacionamento entre os subsistemas que juntos compõe um sistema integrado, pois no caso de apenas existir um acoplamento funcional fraco, não existe relação entre os subsistemas.

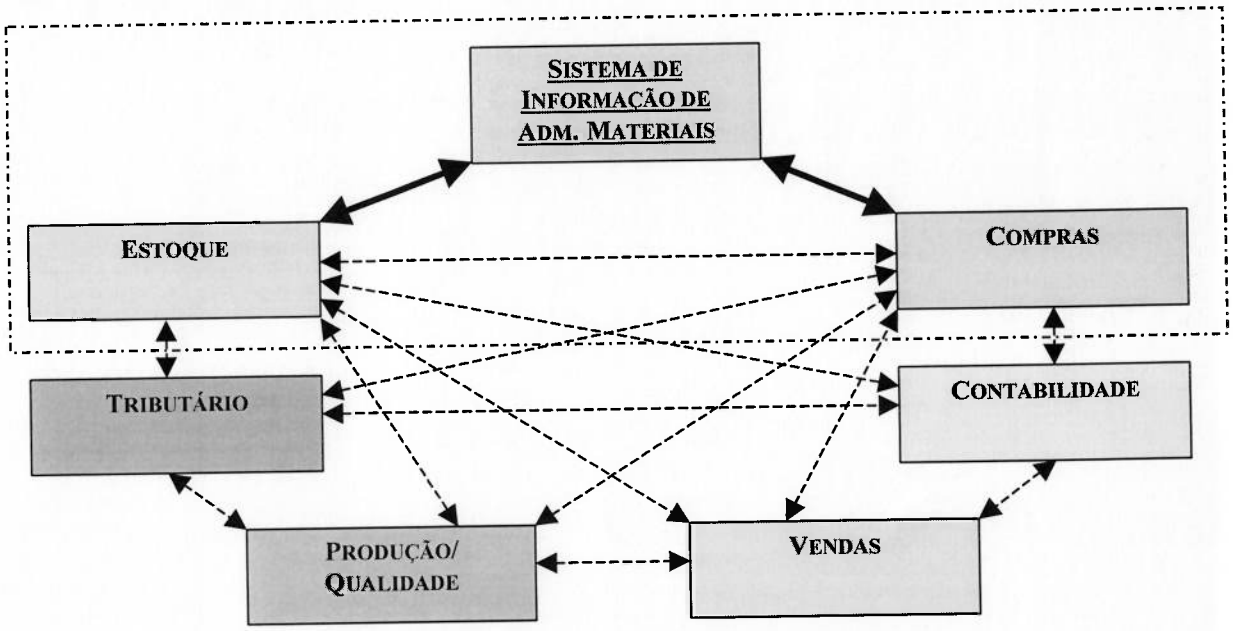


Figura 20: Esquema das inter-relações no sistema de informação de administração de materiais para o processo de fabricação de blank soldado.

Como exemplo de sistema integrado, na Figura 20 apresentamos as relações de vários subsistemas em um sistema de informação de administração de materiais, para o processo de fabricação de blanks. É possível identificar várias interdependências entre

seus subsistemas, a saber, o Estoque - que indica a existência do material - para os setores de Compras, Vendas, Contabilidade, Produção e Qualidade, respectivamente, no que tange os materiais necessários para fabricação e entrega dos produtos finais e seus respectivos custos. Outras dependências se verificam entre a área Tributária e as áreas de Compras, Vendas, e Contabilidade no que diz respeito aos impostos devidos e suas contabilizações; entre a área de Compras e a de Contabilidade, referente ao pagamento dos fornecedores; e finalmente a relação entre Vendas e Contabilidade, para contas a receber. A modelagem desse sistema deve ser integrada para otimização de recursos , redução do tempo de implementação dos sistemas e para minimizar as falhas de comunicação.

Esses relacionamentos determinam os tipos de implementação possíveis e podem determinar inclusive a funcionalidade e exeqüibilidade do projeto. A relação do sistema com os subsistemas é tal que a funcionalidade pretendida deve ser conseqüência da integração dos subsistemas. A complexidade do sistema aumenta pois cada subsistema possui um domínio específico que possuem relações e restrições particulares, e, para cada inter-relação existem diferentes funcionalidades, flexibilizando assim o sistema total

Se cada módulo M_i tem um conjunto de funcionalidades distintas $\{F_i\}$, $i \in N^*$, onde pelo menos uma funcionalidade é mais significativa em um dado instante, é dito que o módulo é flexível. Um sistema flexível é a agregação de módulos flexíveis, distribuídos, independentes e cooperativos, ou seja, módulos com uma integração funcional forte.

Um sistema flexível é fechado se a funcionalidade mais significativa em um dado instante faz parte de um conjunto de funcionalidades de base invariante no tempo. Reciprocamente, é dito que o sistema flexível está aberto se a funcionalidade significativa estiver uma base aberta, ou seja, um conjunto de funcionalidades que varia no tempo.

As definições acima caracterizam o tipo de sistema automatizado que nós pretendemos modelar através de uma representação baseada em objetos complexos, que

será apresentada no capítulo seguinte: a representação AV/AO (Abstract View/Abstract Object). Como um exemplo apresentamos um estudo de caso de um sistema de informação para administração do fluxo de materiais para fabricação de blanks, descrito no Capítulo 6.

Em linhas gerais um sistema (notadamente um sistema informacional) é caracterizado por *views* [Tomiyama97], [Umeda96] que por sua vez são associados a cada classe de possível usuário. Portanto os diferentes subsistemas se comunicam através de seus pórticos ou *views* e nunca diretamente, indo aos detalhes de composição destes.

Por outro lado, a parte essencial do subsistema pode ser caracterizada por um objeto genérico, aqui chamado de Abstract Object (AO), que além de ter uma definição paramétrica pode ter métodos de duas categorias : os comuns, destinados a alterar o valor dos atributos do AO e definições em lógica de primeira ordem, onde se encontra a descrição da sua funcionalidade. Além disto o AO - assim como o subsistema que este representa - pode enviar e receber mensagens, o que é essencial para compor sistemas integrados e flexíveis.

Cada AV pertence a um AO, formando um sistema dual, como veremos mais adiante no Capítulo 5. A representação dual é comum em Engenharia - tome-se por exemplo as redes de Petri [Silva95]. Neste caso a expressividade que queremos apresentar não está somente na contraposição dos elementos duais mas na relação semântica entre estes (que transcende inclusive as redes estendidas). A relação semântica entre estes elementos é representada como uma metáfora, cuja descrição formal se encontra na seção seguinte.

Esta representação é extremamente conveniente para o design dos sistemas de informação. Por outro lado, os sistemas de informação concentram todos os problemas de automação uma vez que são em geral sistemas flexíveis e cooperativos, embora não tenham os problemas de implementação característicos dos sistemas industriais.

Representações para sistemas de informação requerem um design híbrido "top-down" e "bottom-up" [Booch91], [Chan93], [Coleman94], [Gamma95], [Medland92], no sentido de que algumas partes do design já são conhecidos e poderão ser reutilizados de outros sistemas, enquanto outras partes - principalmente as funções de cooperação - devem ser geradas através de refinamentos sucessivos das funcionalidades principais.

O conceito de integração apresentado acima é mais freqüentemente aplicado no domínio do sistema operacional, isto é, no sistema que realiza ações: o chão de fábrica, controle de sistemas prediais, etc. Assim, uma célula de manufatura padrão, ou mesmo um sistema supervisório que controla um edifício são facilmente identificados como sistemas integrados. No entanto o mesmo conceito pode ser aplicado no âmbito do sistema decisional.

Através de refinamentos sucessivos do sistema é possível transformá-lo num conjunto de sub-sistemas com funcionalidades distintas, interligados entre si, de forma a separar claramente os objetivos e aspectos de cada parte integrante do sistema, como é o caso dos sistemas operacional, informacional e decisional. A conexão entre esses sistemas pode ser mapeada através de metáforas, definidas a seguir.

4.1.1 Metáforas e Relações de Coerência

Em princípio, são geradas metáforas quando um mesmo conteúdo semântico é representado em dois idiomas diferentes. Em casos práticos cada um destes idiomas é usado por classes diferentes de oradores [Silva92]. Nota-se que subsistemas integrados apresentam um mesmo conjunto semântico enquanto subsistemas fracamente acoplados não apresentam. Portanto esse estudo é voltado para sistemas integrados.

Indurkhya [Indurkhya87] [Indurkia86] propôs uma formalização de metáforas fundado em lógica de primeira ordem, classificando uma metáfora como uma transferência semântica entre domínios. Embora apresente algumas desvantagens [Gerrig89] apud [Silva92], é uma estrutura que pode representar símiles – um tipo especial de metáfora (ver Def. 3) mais adequado para ser utilizado no mapeamento dos sistemas operacional, informacional e decisional.

Uma metáfora é uma transferência semântica entre dois domínios onde:

Def.1.] Um domínio é uma quadrupla $< V, f, Sd, S >$ onde

V é um conjunto de símbolos

$f: V \rightarrow \Omega$, é um mapeamento no conjunto de tipos Ω

$Sd \subseteq S$ é um conjunto de derivações que definem condições

S é um jogo de fórmulas bem formadas em cima do vocabulário V = $< V, f >$

Um mapeamento $V_1 \rightarrow V_2$ entre dois conjuntos de símbolos é dito admissível se preservar aridade, isso é, só predicados com o mesmo número de argumentos ou instâncias são mapeados.

Def.2] Dados dois domínios:

$D_1 = < V_1, f_1, Sd_1, S_1 >$, $D_2 = < V_2, f_2, Sd_2, S_2 >$ e um mapeamento admissível $\phi: V_1 \rightarrow V_2$, uma metáfora é um par $< \phi, S >$ onde $S \subseteq S_1$ é um conjunto de sentenças transformáveis.

Def.3] A metáfora $< \phi, S >$ é dito coerente se $S'' = \phi(S) \cup S_2$ é consistente.

Um símile é um tipo especial de metáfora onde $V_1 \cap V_2 \neq \varnothing$, quer dizer, há um conjunto comum de condições que são usados em ambos os domínios. Por exemplo, estes domínios podiam ser instanciados por um comportamento e um modelo funcional do mesmo processo ou artefato onde os mesmos termos são usados em ambos.

Um mapeamento estendido admissível é aquele que aceita símbolos de alto nível, onde argumentos são procedimentos ou funções como "enviar mensagem", e esse mapeamento se aplica somente para argumentos contidos no conjunto estendido $V_2 \cup M$, onde V_2 são símbolos do modelo funcional e M são as especificações de um AO.

É dito que uma metáfora $\vartheta =< \phi^*, S >$, onde ϕ^* é um mapeamento estendido, é uma relação coerente entre se $S'' = \phi^*(S_1) \cup S_2$ é consistente.

Objetos complexos, mapeados por metáforas, são base para representar um modelo dual estrutural/funcional para sistemas integrados e flexíveis.

Na próxima seção apresenta-se as formas de representação de sistemas integrados e flexíveis.

4.2 Representação de Sistemas Integrados e Flexíveis.

Os sistemas automatizados podem ser representados através de três modelos, a saber, relacional, que apresenta as relações entre os subsistemas, funcional, o qual descreve as especificações funcionais dos sistemas e subsistemas, e dinâmico, responsável pelo controle do processo a ser implementado.

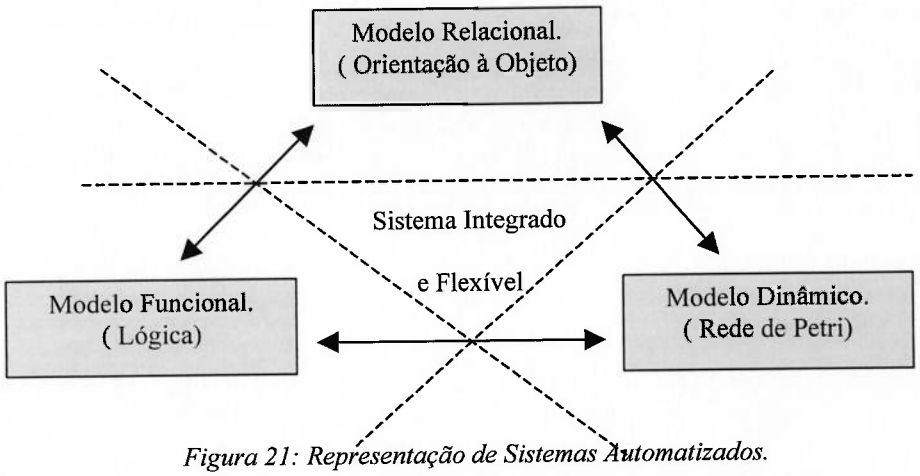


Figura 21: Representação de Sistemas Automatizados.

A Figura 21 apresenta uma solução para representação de cada um desses modelos separadamente.

Cada um desses modelos apresentam diferentes formas de representação. O modelo funcional é representado em linguagem coloquial no caso de requisições de sistemas automatizados, todavia se pode substituir a linguagem coloquial pela lógica de primeira ordem nessa representação.

O modelo relacional utiliza representações não formais, como é o caso da Orientação à Objetos que, apesar de ser estruturada, não apresenta um modelo matemático.

O modelo relacional em conjunto com modelo funcional compõe a representação estática ou estrutural do sistema automatizado (Figura 22).

O modelo dinâmico apresenta várias formas de representação como cadeia de Markov e redes de Petri. Essa última apresenta vários estudos formais avançados para otimização e desenvolvimento de sistemas.

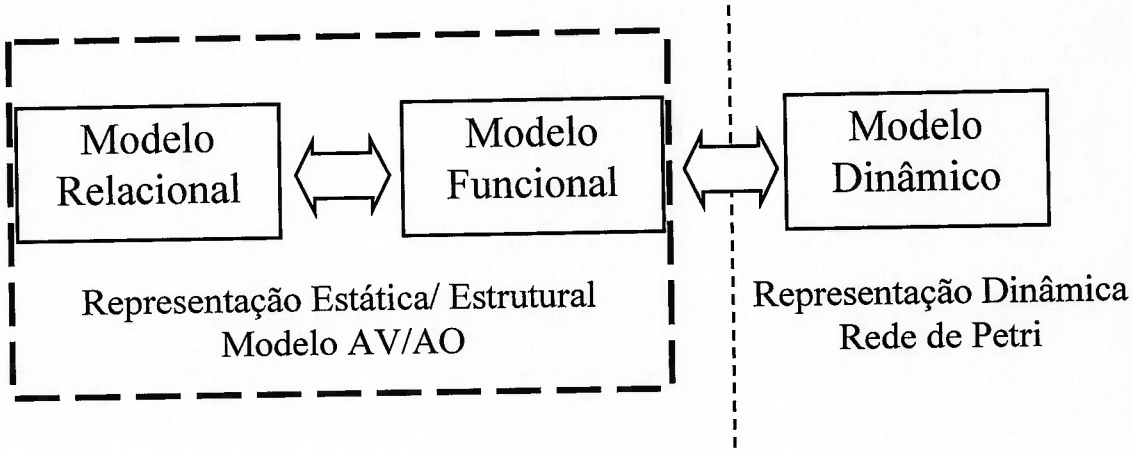


Figura 22: Representação Estática e Dinâmica de Sistemas Automatizados.

Dessa forma existe a possibilidade do sistema informacional, baseado nesta estrutura (interpretado como um conjunto de objetos formais que compõem uma representação), se ligar tanto à parte gerencial, como à parte operacional, isto é, o chão de fábrica, através de uma representação formal que pode se utilizar de mecanismos de análise da consistência do sistema, reduzindo as deficiências e falhas na troca de dados entre os modelos operacional, informacional e decisional.

O Capítulo 5 a seguir apresenta as definições formais do modelo AV/AO baseadas no formalismo descrito nesse capítulo.

Capítulo 5

Modelo AV/ AO.

O modelo proposto para a modelagem conceitual de sistemas automatizados, e que pode também ser utilizado até o projeto detalhado é baseado em objetos abstratos AV/AO, que por sua vez foram adaptados do modelo ADV/ADO [Alencar94], [Cowan95]. O *Abstract Data View* (ADV) foi criado para especificar claramente e formalmente a separação de funcionalidade ("separation of concerns") entre uma componente de software e o seu domínio de aplicação. Isto é particularmente importante quando a componente de software se destina a vários usuários, todos com características específicas e utilizando a mesma componente com diferentes propósitos. Essa aproximação para especificação de interfaces separa os componentes de aplicação dos demais dentro de uma abordagem cliente/servidor. Assim, os modelos das componentes de aplicação são chamados de *Abstract Data Objects* (ADO), que são projetados para minimizar o conhecimento do ambiente em que eles são usados e podem ser melhor reutilizados.

A diferença entre a proposta de Cowan e Lucena e a que apresentamos neste artigo vai além da supressão do termo "Data", característico das componentes de software. O modelo AV/AO, contém uma abordagem em lógica clássica para as relações entre componentes e para a conexão entre AV que é de fato a expressão da estrutura de uma rede de Petri do sistema integrado. Tem também uma proposta específica de análise de consistência entre cada modelo lógico parametrizado, AO, e seu respectivo comportamento (AV). Apesar do formalismo proposto ser simples, é o suficiente para a maior parte dos projetos de design em engenharia, principalmente para

os sistemas cujo processo de integração está intimamente associado a sistemas de controle supervisório, como mostraremos em um exemplo adiante.

A estruturação do projeto está associada à abordagem orientada a objetos (ambos AV e AO são de fato objetos. Além de responder pela parte funcional da abordagem, esta característica é um indicativo de que há uma boa possibilidade de reutilização de projetos representados por AV/AO, uma vez que os melhores resultados para a reutilização de projetos está associada a métodos orientados a objetos.

A relação entre AV e AO não é simétrica, uma vez que várias instâncias de AV podem estar associadas a um mesmo AO de forma a criar diferentes pontos de vista ou funcionalidades. Este relacionamento *muitos-para-um* significa que cada instância de AV deve ser coerente com o AO associado, o que é denominado de coerência vertical. Implica também que todas as instâncias dos AV devem ser consistentes entre si, o que é chamado de coerência horizontal, conforme Figura 23.

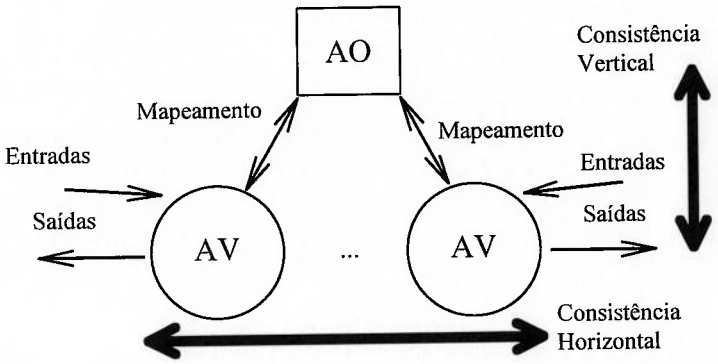


Figura 23: Relação muitos-para-um e propriedades de coerência do modelo AV/ AO.

O conceito da consistência garante a abstração correta de componentes de projeto e é fundamental no processo de reutilização, uma vez que as componentes reutilizadas devem ser adaptadas antes de serem incluídas em um novo projeto.

Nesse modelo, tanto o AV como o AO podem ser refinados, e a consistência horizontal e vertical deve ser mantida entre eles.

Com um modelo seguindo a metodologia a ser indicada, consegue-se um sistema que ao mesmo armazena tempo o objeto projetado, AO, e possibilita a busca de objetos numa base de dados através de especificações por meio de AV.

Pode-se iniciar o projeto com as características funcionais, sem necessariamente especificar de início as características do objeto a ser projetado. Uma vez identificado um objeto a ser reutilizado (e as inter-relações do novo projeto com ele) através de AV, esse pode ser "integrado" ao projeto pelo AO correspondente, depois de verificada a consistência horizontal. Tudo se passa como esse novo AO fosse um elemento de base para uma composição [Paton93], [Silva94].

O processo de reutilização empregado é uma versão simplificada do método baseado em metáforas descrito em [Silva 92].

Conceitualmente as relações de consistências possuem definições formais, a saber, a vertical indica que, caso exista um mapeamento de entradas e saídas de AV para um determinado AO, este deverá deduzir toda e qualquer entrada em uma saída. A horizontal, formaliza que, caso duas entradas indiquem a mesma saída de um AO, as entradas devem ser idênticas, gerando unicidade.

5.1 Formalização do Modelo AV/AO.

Formalizando as especificações de design tem-se sentenças lógicas, cuja dedução nos leva a funcionalidades específicas do projeto.

Utiliza-se a lógica de primeira ordem para modelagem de design para poder programar e implementar as funções e mapeamentos de relacionamentos entre AV e AO conforme descritos nesse capítulo.

Através de refinamentos é possível transformar um conjunto de funcionalidades desejadas em vários outros, interligados entre si, de forma a separar claramente os objetivos e aspectos de cada parte integrante do design.

Denomina-se AO (Abstract Object) objetos ou partes com funcionalidades definidas, enquanto as relações entre as funcionalidades das partes/ objetos são chamados AV (Abstract View). A Figura 24 esquematiza AO's e suas relações AV's.

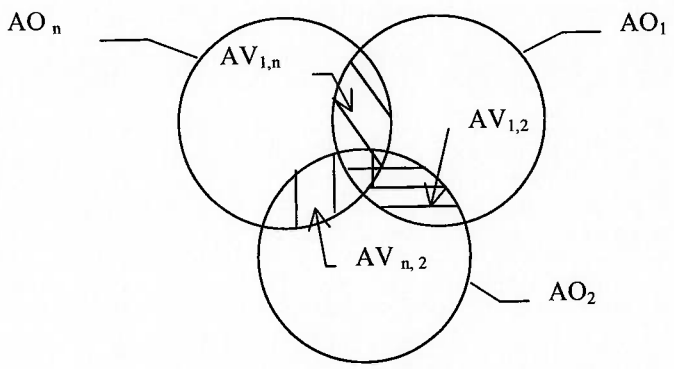


Figura 24. Esquema de AO's e suas relações AV's

Para formalização e tratamento desse modelo são necessárias algumas definições matemáticas iniciais.

5.1.1 Fundamentos Teóricos

Seja Ω o conjunto das sentenças lógicas.

Todo conjunto $\Sigma \subseteq \Omega$ de sentenças determinam unicamente uma classe K de sistemas matemáticos, na verdade, a classe de todos aqueles sistemas matemáticos em que toda sentença de Σ é válida. Σ é algumas vezes referenciado como um sistema postulado de K; sistemas matemáticos que definem K são chamados modelos de Σ [Chung77].

Definição 1) Define-se o conjunto Σ^* como:

$\Sigma^* = \Sigma - \varnothing$, onde $\varnothing$ é a cláusula vazia.

Definição 2) D_A é um conjunto de cláusulas consistente, se, e somente se:

$\{ D_A \subseteq \Sigma,\ F_A \subseteq \Sigma^* \ / \ D_A \vdash F_A\}$, onde o operador $\vdash$ indica que D_A deduz as cláusulas de F_A. F_A é denominado funcionalidade de D_A.

Definição 3) Dado um conjunto $D \subseteq \Sigma^*$ e $F \subseteq \Sigma$, os conjuntos $D_1, D_2, \ldots D_N$ são chamados de refinamentos de D, se, e somente se:

$D \vdash F_1 \wedge (D_1 \cup D_2 \cup \ldots \cup D_N) \vdash F_2 \mid F_1 \sim F_2$, onde o operador $\sim$ indica equivalência entre conjuntos. Analogamente, D é dito abstração de $D_1, D_2, \ldots D_N$.

Equivalência é definida da seguinte maneira:

para a seguinte notação: $D_1 \vdash F_1, D_2 \vdash F_2, \ldots, D_n \vdash F_n, D_i \subseteq \Sigma^*, F_i \subseteq \Sigma, 1 < i < n$, define-se que se $D_j \sim D_k \Rightarrow F_j \equiv F_k, 1 < j < k < n$.

Definição 4) Dados dois conjuntos $D_I, D_J \subseteq \Sigma$,se $C_{I,J} \nwarrow D_I \wedge C_{I,J} \nwarrow D_J$, $C_{I,J}$ denomina-se conexão entre os conjuntos D_I e D_J. O operador $a \nwarrow b$ denomina que a subsumes b, o que indica que para $f_1, f_2, a, b \subseteq \Sigma$, temos que $a \vdash f_1$ e $b \vdash f_2, f_1 \subseteq f_2$.

Essa conexão é integrada se $C_{I,J} \subseteq \Sigma^*$, analogamente, é fracamente acoplada se $C_{I,J} \vdash \varnothing$.

5.1.2 Design

O design de sistemas cada vez mais envolve áreas específicas que anteriormente não eram relacionadas. Sendo assim, um modelo de design deve ser capaz de que, transformando cada área específica num domínio D_i, cada domínio conhecido se relacione com os outros.

Outra característica importante para o modelo de design é o fato de que o conhecimento dos domínios ou se mantém constante ou se aprimora, determinando assim um modelo totalmente interativo capaz e que possa reconhecer as inter-relações entre domínios e ampliar a área de conhecimento dos mesmos. A Figura 25 apresenta graficamente a expansão de dois domínios e o surgimento de uma inter-relação.

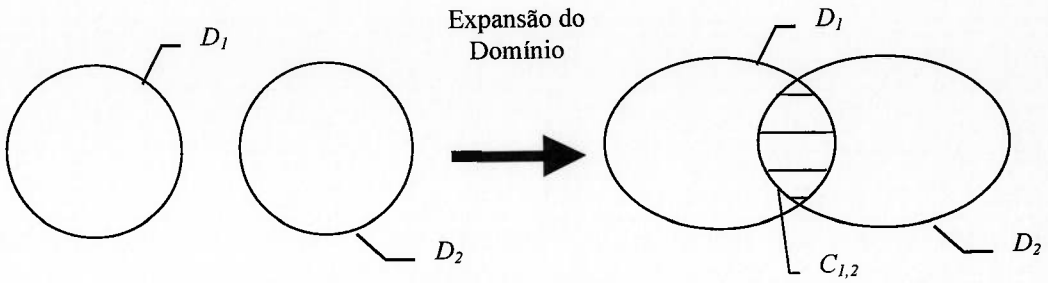


FIGURA 25. Representação da expansão de dois domínios e o surgimento de uma região de inter-relação mútua.

5.1.3 Modelo AV/AO

O modelo AV/AO foi desenvolvido para possibilitar a separação dos domínios através de Objetos Abstratos (AO) e suas conexões, as Vistas Abstratas (AV). Como já foi dito, denomina-se AO objetos, domínios ou partes com funcionalidades definidas, enquanto o AV é a relação entre as funcionalidades das partes/ domínios/ objetos.

Cada AO deve ser consistente conforme a Definição 2, ou seja, não deve deduzir a cláusula vazia. Caso isso ocorra, significa que a funcionalidade do domínio pode ser descartada, ou seja, o domínio não possui finalidade.

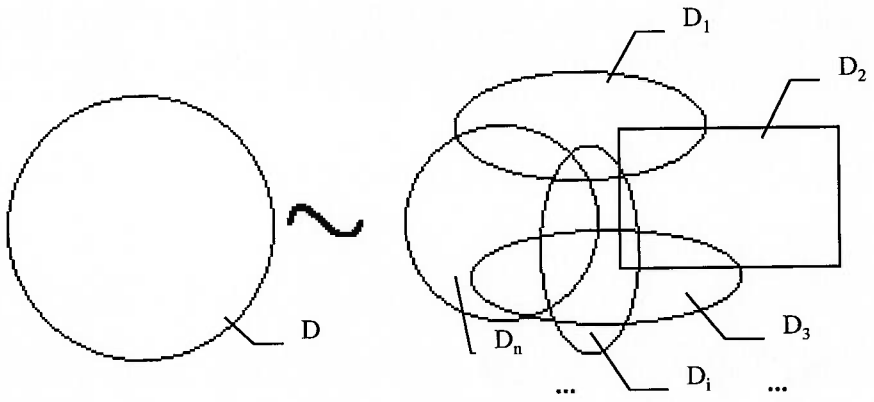


FIGURA 26. Representação do refinamento do domínio D *como a união dos domínios* D_i, $1 < i < n$.

Através da operação de refinamento, separa-se um AO em vários outros AO's, e a união desse refinamento deve ser consistente e conforme a Definição 3. A consistência do refinamento é denominada Consistência Vertical no modelo AV/AO. Na Figura 26

tem-se um esquema de refinamento de um domínio D nos domínios D_i, i, $n \in \aleph$, $1 < i < n$.

Nem sempre são conhecidas as relações entre domínios específicos e, em outras ocasiões, deseja-se uma correlação pré definida entre dois domínios. Nesses casos, o modelo do design precisa tratar separadamente os relacionamento dos domínios, a saber, os AV's.

Por se tratar de uma modelagem, ao se criar um domínio e suas interações com outros domínios, a base lógica criada pelo domínio e suas inter-relações deve ser uma só, que é o domínio completo. A Figura 27 nos mostra graficamente um domínio e suas interseções com outros domínios.

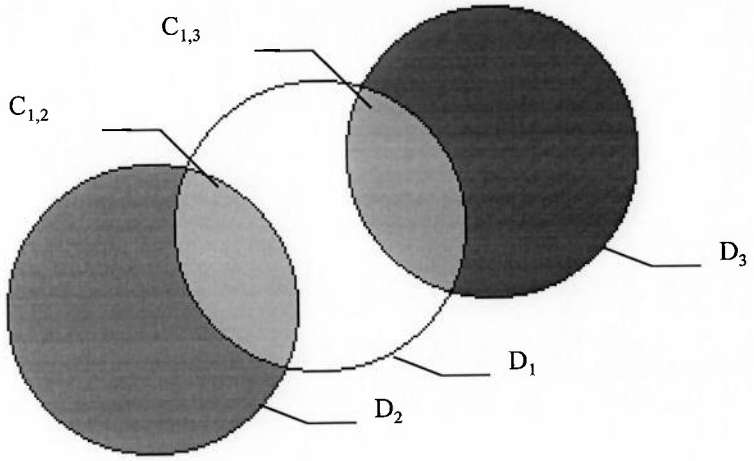


FIGURA 27. Representação de um domínio e suas interseções com outros dois.

Pode-se aplicar a consistência de um domínio para a união dele e suas inter-relações, gerando a operação de aglutinação, da seguinte forma:

Para a Figura 27, utilizando a Definição 4 tem-se que $C_{1,2}$, $C_{1,3} \not\subset D_1$, pode-se dizer *que* $D_1 \equiv C_{1,2} \cup C_{1,3} \cup D_1$.

Se D_1 é consistente, conforme Definição 2, $D_1 \vdash F_1$, $F_1 \subseteq \Sigma^*$. Conforme Definição 3, $D_1 \vdash F_1 \wedge (C_{1,2} \cup C_{1,3} \cup D_1) \vdash F_2 \mid F_1 \sim F_2$. *Portanto* $C_{1,2} \cup C_{1,3} \cup D_1$ é consistente.

Analogamente, para $i, j, \quad n \in \aleph, \quad 1 < i, j < n \quad \wedge \quad C_{ij} \not\subset D_A$, se D_A é consistente

$\Rightarrow \bigcup C_{i,j} \cup D_A$ também é consistente.

Essa consistência é chamada de Consistência Horizontal. Note que nesse caso, cada interseção não contradiz seu domínio, bem como a união das interseções; entretanto as interseções não geram consistência uma para outra, apenas para o todo.

As duas consistências apresentadas, horizontal e vertical, validam a integridade do modelo AV/AO. Na verdade, trata-se da expansão das Definições 2, 3 aplicados a Definição 4, refinando um domínio na união de vários outros e as suas respectivas interseções.

A definição formal das consistências está nas equações abaixo:

Definição 5) Consistência Vertical $\boldsymbol{R^V}$:

$\boldsymbol{R^V : [\,(x,y) \in AV^I{}_J \mid x \in S \wedge y \in P \wedge AO_J \quad (x \rightarrow y)],}$

onde $\boldsymbol{S}$ e $\boldsymbol{P}$ são respectivamente o conjunto de entradas e saídas do $\boldsymbol{AV^I}$ referente ao $\boldsymbol{AO_J}$, $\boldsymbol{x}$ é um dado de $\boldsymbol{S}$ e $\boldsymbol{y}$ um dado de $\boldsymbol{P}$.

Definição 6) Consistência Horizontal $\boldsymbol{R^H}$:

$\boldsymbol{R^H : [\exists\,(x,y) \in AV^I{}_J \wedge \exists\,(x,z) \in AV^K{}_J \Leftrightarrow z \cong y].}$

onde

As Figuras 28 e 29 representam graficamente as consistências Vertical e Horizontal.

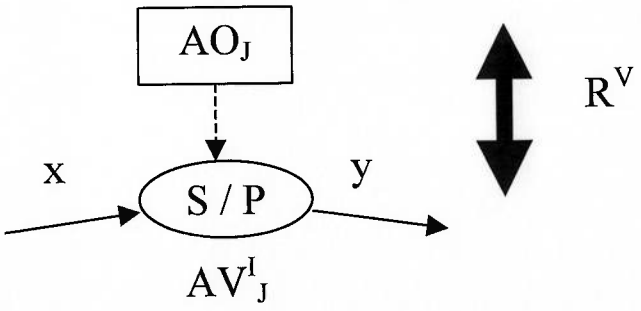


Figura 28: Diagrama Esquemático da Consistência Vertical entre AV e AO.

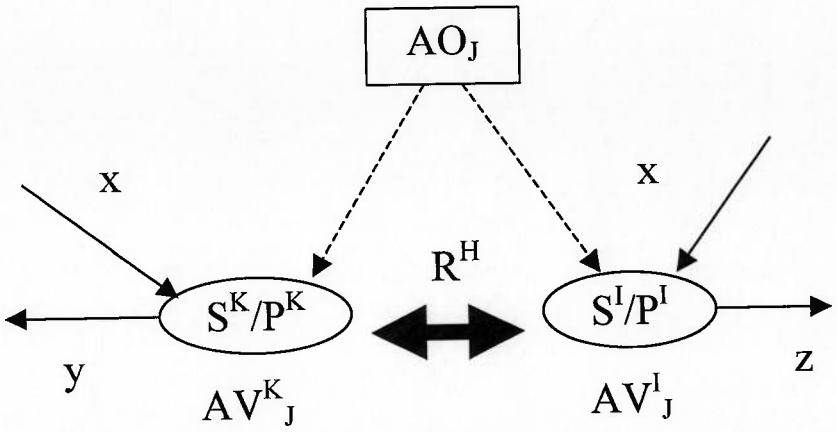


Figura 29: Diagrama Esquemático da Consistência Horizontal entre AV's do mesmo AO.

As duas operações refinamento e aglutinação podem ser aplicadas uma sobre a outra por inúmeras vezes, criando diferentes modelos de sistemas.

Aplicando a Definição 4 ao modelo AV/AO extrai-se que, para dois AO's, AO_1 e AO_2, se existe um AV, $AV_{1,2} \notin \varnothing$, tal que $AV_{1,2} \not\upharpoonright AO_1 \wedge AV_{1,2} \not\upharpoonright AO_2$, deduz-se que a inter-relação entre AO_1 e AO_2 é $AV_{1,2}$.

Apenas se tratam conexões integradas, já que a conexão fracamente acoplada é sempre válida, portanto trivial.

O modelo AV/AO também foi desenvolvido para possibilitar o desenvolvimento do design de uma forma híbrida, ou seja, BOTTON-UP e TOP-DOWN. Nas definições acima descritas, a operação de aglutinação, conforme a Figura 30, demonstra claramente essas características, uma vez que a união de vários AO pode ser tratada como sendo apenas um AO, e vice-versa, um AO pode ser reduzido a uma união de muitos. Sendo assim, pode-se modelar um design iniciando-se através de uma base de componentes existes, ou de sua funcionalidade final.

O fato de ser híbrido faz com que um AO desconhecido possa ser reduzido a uma base de AO's conhecidos, domínios esses que tenham, por exemplo, modelos em rede de Petri definidos, ponto esse a ser estudado não neste trabalho, mas em outra tese.

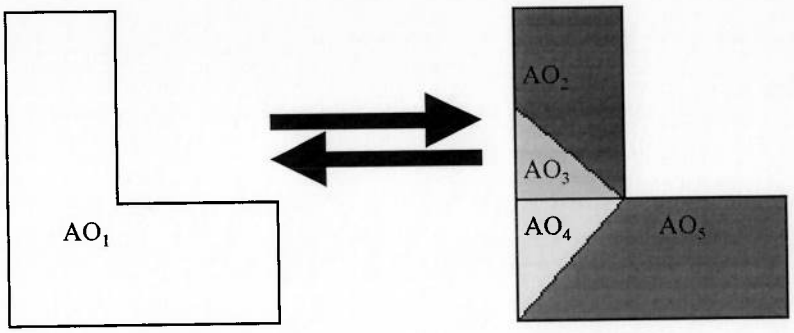


FIGURA 30. Modelagem Híbrida: TOP-DOWN e BOTTON-UP.

5.2 IMPLEMENTAÇÃO DO MODELO AV/AO.

Para ser implementado o modelo AV/AO é preciso de uma base de dados capaz de armazenar cláusulas lógicas, e que o conjunto dessas cláusulas possam ser trabalhados conforme as definições apresentadas no item 4.1.

Uma base de dados orientada a objeto possibilita as operações de refinamento e aglutinação; entretanto, não são formais nem tão pouco tratam de cláusulas lógicas.

Uma base de dados tipo Prolog [Bittencourt96], [Levesque86] é matematicamente formal e trata cláusulas lógicas [Gray93], mas não possui ferramentas de aglutinação e refinamento.

Uma solução para a implementação são as linguagens dedutivas orientadas a objeto as quais integram importantes características de base de dados dedutivas e base de dados orientadas a objeto com uma semântica bem definida. Exemplos dessas linguagens são RLOG [Liu96-a], Datalog [Ceri90] apud [Liu98-b], LDL [Naqvi89] apud [Liu98-b] e Coral [Abiteboul91] apud [Liu98-b].

Na última década um número de bases de dados dedutiva orientado a objeto foram propostas, como o Flogic [Kifer95] apud [Liu96-b], Rock&Roll [Barja95] apud [Liu96-b] e ROL [Liu96-a] [Liu98-b]. Entretanto, a maioria deles são apenas estruturalmente

orientados a objeto. Importantes características orientadas a objeto tais como métodos e encapsulamento não são suportadas. [Liu98-b]

Vários sistemas dedutivos orientados a objeto que também são estruturais e comportamentais simplesmente provêm dois tipos incompatíveis de linguagens: linguagem declarativa baseada em regras para dedução e busca, e linguagem imperativa para definição e manipulação de dados e tarefas complexas.

Dentre esses sistemas, os que mais se aproximam do Modelo AV/AO são o ROL [Liu96-a] e RLOG [Liu98-a]. Esses dois sistemas estão mostrados no Anexo 1 e Anexo 2 respectivamente.

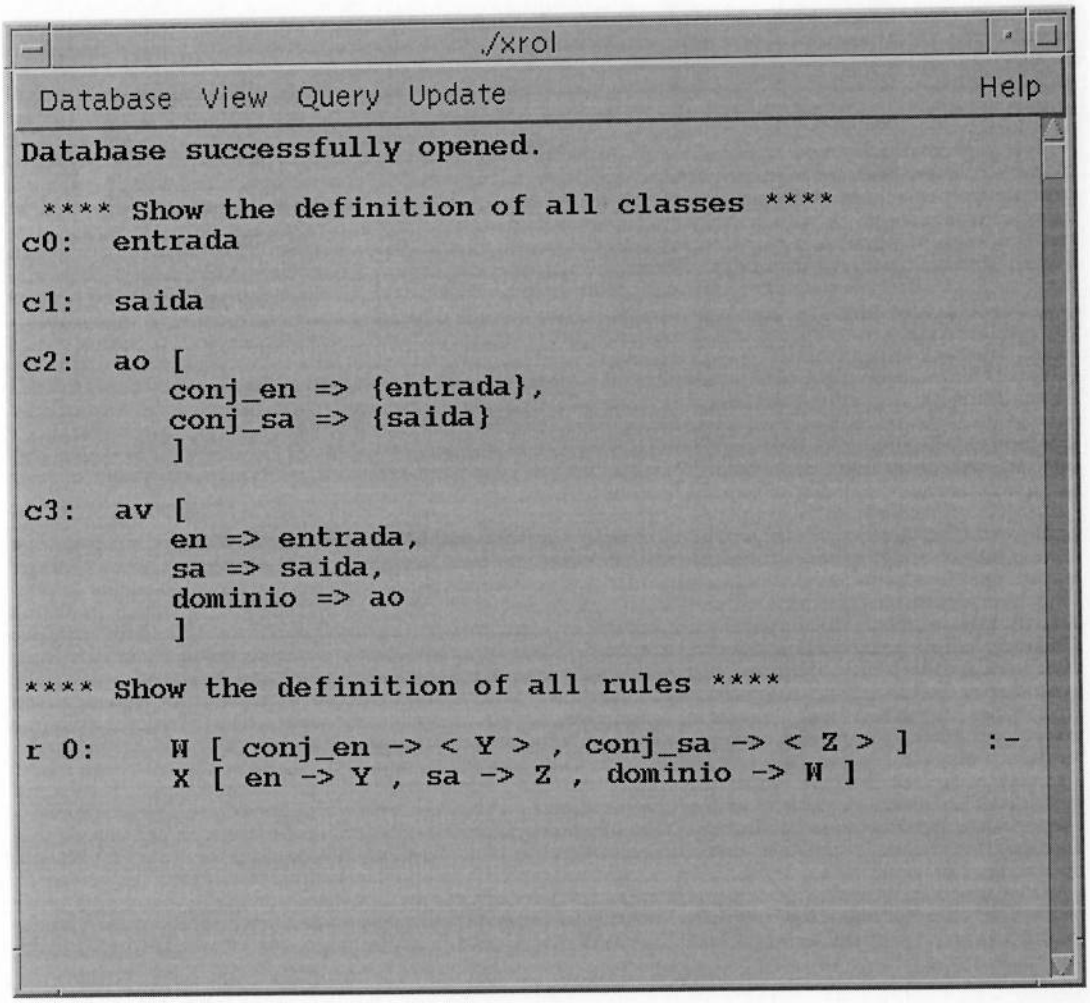


Figura 31: Relação de Consistência do Modelo AV/AO no ROL.

O RLOG é um sistema de base de dados dedutivas que suporta a armazenagem e inferência de dados com estruturas complexas, especialmente dados suportados em relações de encapsulamento e modelos de objeto complexos. Veja Anexo1.

O ROL foi estendido em dezembro de 99 para ROL2, o qual suporta métodos baseados em regras e encapsulamento [Liu98-c]. Veja Anexo2.

Como exemplo apresenta-se na Figura 31 as regras básicas de consistência do modelo AV/AO no ROL, e na Figura 32 as regras implementadas no RLOG.

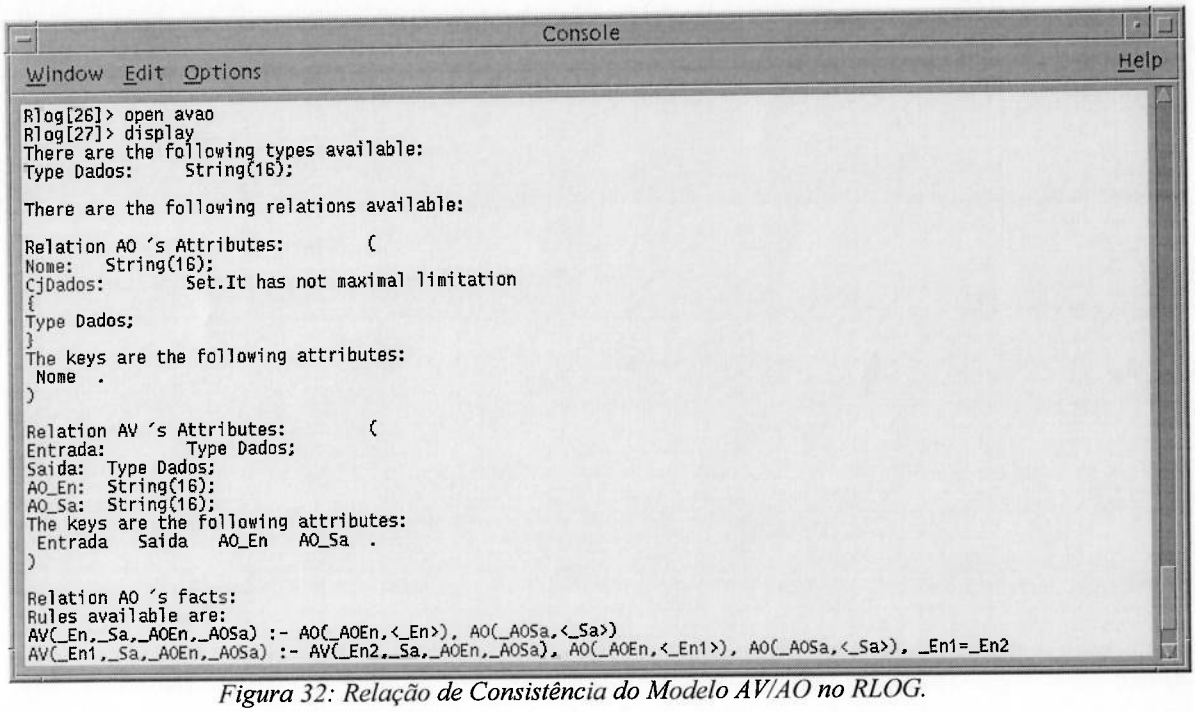


Figura 32: Relação de Consistência do Modelo AV/AO no RLOG.

Capítulo 6

Estudo de Caso: Otimização da Informatização da Administração de Materiais na Fabricação de Blanks.

Para este estudo de caso foi escolhida a área de administração de materiais na fabricação de blanks pelo fato da crescente adoção de blanks no processo de conformação de chapas promovendo algumas mudanças nítidas no processo de manufatura, destacando-se as mudanças logísticas e as mudanças de produtividade, uma vez que a redução do número de peças, antes separadas, estão agora formando um blank unido, reduzindo o número de operações de estampagem em prensa, por que ao conformar uma só peça em lugar de duas ou três, se reduz o consumo de energia, se otimiza a utilização de linhas, se diminui o tempo de produção em prensa e a montagem no carro e se reduz estoques intermediários ao estocar um só lote ao invés de várias peças não unidas.

Outra característica importante é o fato da fabricação do blank ser terceirizada a um centro de serviços, devido ao fato da empresa produtora de bobinas de aço dirigir seu foco à produção de aço, e não em equipamentos para o corte das bobinas em blanks e na solda à laser para blanks soldados. Aliado a isso tem-se o fato da fábrica de bobina ser distante do centro de serviços, o que faz com que haja a necessidade de existir uma filial junto ao centro de serviço que se responsabilize pela programação de corte dos blanks, pedidos de transferência das bobinas e remessa e faturamento de blanks para o cliente final.

6.1. Administração de Materiais na Fabricação de Blanks.

O gerenciamento da fabricação do blank soldado é dividido em três processos, detalhados na Tabela 3.

O processo de venda de Blank Soldado baseia-se no estoque do produto. A fabricação de Blank Soldado é "Make to Stock", ou seja, a sua produção é baseada no número de peças disponível em estoque. Para se fabricar o Blank Soldado é necessário a matéria-prima, a saber, as bobinas de aço, produzidas na matriz através de uma solicitação de transferência entre o centro produtor e a filial responsável pela fabricação dos Blanks Soldados. Depois que remeter as bobinas de aços para beneficiamento no fornecedor, inicia-se a fabricação do tipo "Make to Order", o qual produz e entrega o produto final à filial num prazo médio de 03 (três) meses. Não se faz a venda triangular de bobina com beneficiamento no fornecedor, porque a necessidade de se fabricar Blanks Soldados é semanal conforme um programa de remessa "Just in Time", e a quantidade a ser produzida é muito menor que a das bobinas que o compõe, visto que bobinas são produzidas em tonelada e Blanks Soldados em peças.

1. Processo de Venda.	♦ Programa de Remessa do Cliente; ♦ Remessa e Faturamento do Blank.
2. Processo de Compra	♦ Envio da bobina de aço para beneficiamento; ♦ Recebimento do blank soldado.
3. Processo de Transferência.	♦ Pedido de Transferência entre centros; ♦ Produção da bobina de aço; ♦ Remessa e faturamento da bobina; ♦ Recebimento da bobina.

Tabela 3 - Processo de Negócio do Blank Soldado

Os sistemas de informação para a administração de materiais existentes são modelados de forma não formal, e por isso, utilizam os processos sem vínculo direto entre eles, ou seja, não se baseiam na ordem de venda para criar os pedidos de compra, bem como possuem pedidos de transferência desvinculados da venda ao cliente final. Isso acaba gerando inconsistência no processo como um todo podendo criar situações onde se processa o corte de bobinas em quantidades diferentes da ordem de venda, se

produz bobinas diferentes das necessárias para atender o cliente, se gera ordens de venda sem se ater ao prazo de fabricação real do processo.

A proposta apresentada nesse estudo de caso está no levantamento dos processos existentes através do sistemógrafo, de forma a apresentar as falhas existentes e servir de base para a criação do modelo AV/AO correspondente.

Com o modelo inicial, faz-se o refino até atingir o nível de detalhe desejado, como por exemplo a busca de bobinas de tempo de fabricação com diferenças menos que 06 meses para que não ocorram quebras na etapa de embutimento dos blanks soldados.

6.2. Sistemógrafo

Através de um levantamento *in loco* foram criados os sistemógrafos das operações intrínsecas da área de administração de materiais na produção de Blank Soldado da empresa motivo deste estudo [Salazar99]

As operações da área de administração de materiais no processo de fabricação do Blank Soldado estão subdividida nas seguintes atividades básicas:

a) Recebimento físico e fiscal das bobinas de aço.
b) Recebimento físico de blank soldado.
c) Recebimento fiscal de blank soldado.
d) Pedido de compra de blank soldado.
e) Pedido de transferencias.
f) Remessa de bobina de aço para beneficiamento.

a) Recebimento físicos e fiscal das bobinas de aço.

a.1) Componentes do Sistema:

Entradas:

E1.1 – Bobina de Aço: A empresa recebe bobinas de aço para a fabricação de blank soldado.

Saída:

S1.1 – Envio Nota Fiscal a Área Legal: . Após a entrada da nota fiscal na empresa é enviada a primeira via para a área fiscal.

Processadores:

P1.1 – Pesar Bobinas de aço: Função desempenhada por uma pessoa na portaria que pesa o material e anexa a nota de pesagem à nota fiscal.

Categoria: operacional e informacional

Tipo : espaço e forma

Nível: 4

P1.2 – Conferir Peso Nota Fiscal: Funcionário responsável pelo recebimento de bobina verifica se o peso da Nota Fiscal está dento do aceitável com a relação com o peso medido.

Categoria: operacional, informacional e decisional

Tipo : tempo e forma

Nível: 5

P1.3 – Consultar Matriz: Caso ocorra discrepância de valor entre a Nota Fiscal e peso medido maior a 1%, é contatada a matriz para resolver o problema.

Categoria: informacional e decisional

Tipo : tempo e forma

Nível: 6

P1.4 – Entrada de material no Estoque com referência ao Pedido de Transferência: O funcionário responsável pelo recebimento de bobina digita dados da N.F. (pedido de transferência, lote de bobina, peso de bobina, características físicas, dados de imposto), e o sistema automaticamente faz a baixa do pedido e registro da N.F.

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 4

P1.5 - Definir Local de Estoque: É definido o local de armazenamento das bobinas de aço, que chegam em caminhão ou trem, quando for necessário armazenar os produtos.

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 5

a.2) Sistemógrafo Operacional:

A Figura 33 representa o fluxo operacional existente no processo de Recebimento físico e fiscal de bobinas.

O processador P1.1 pesa as bobinas de aço e emite nota de pesagem. O processador P1.2 confere se a nota fiscal está de acordo com o material recebido. Com sua liberação, inicia-se o processador P1.4 faz a entrada ao material. O processador P1.5 armazena o material no estoque ou envia para beneficiamento.

a.3) Sistemógrafo Informacional:

A Figura 34 representa o fluxo informacional existente no processo de Recebimento físico e fiscal de bobinas. O processador P1.1 informa o peso das bobinas de aço e emite nota de pesagem. O processador P1.2 confere se a nota fiscal está de acordo com o material recebido. O processador P1.3 é contatado para solucionar qualquer problema em relação ao peso das bobinas. Com sua liberação, inicia-se o processador P1.4 faz a entrada do material. O processador P1.5 armazena o material no estoque ou envia para beneficiamento.

a.4) Sistemógrafo decisional:

A figura 35 representa o fluxo decisional existente no processo de Recebimento físico e fiscal de bobinas. O processador P1.2 verifica se o peso das bobinas está de acordo com a nota fiscal. Caso haja algum problema, o processador P1.3 é contatado para solucionar qualquer problema com os dados na N.F.

(*) Os processos "Baixar Pedido" e "Registro N.F." automaticamente se atualizam no processador P1.4 por meio do sistema ERP [Enterprise Resource Planning] existente.

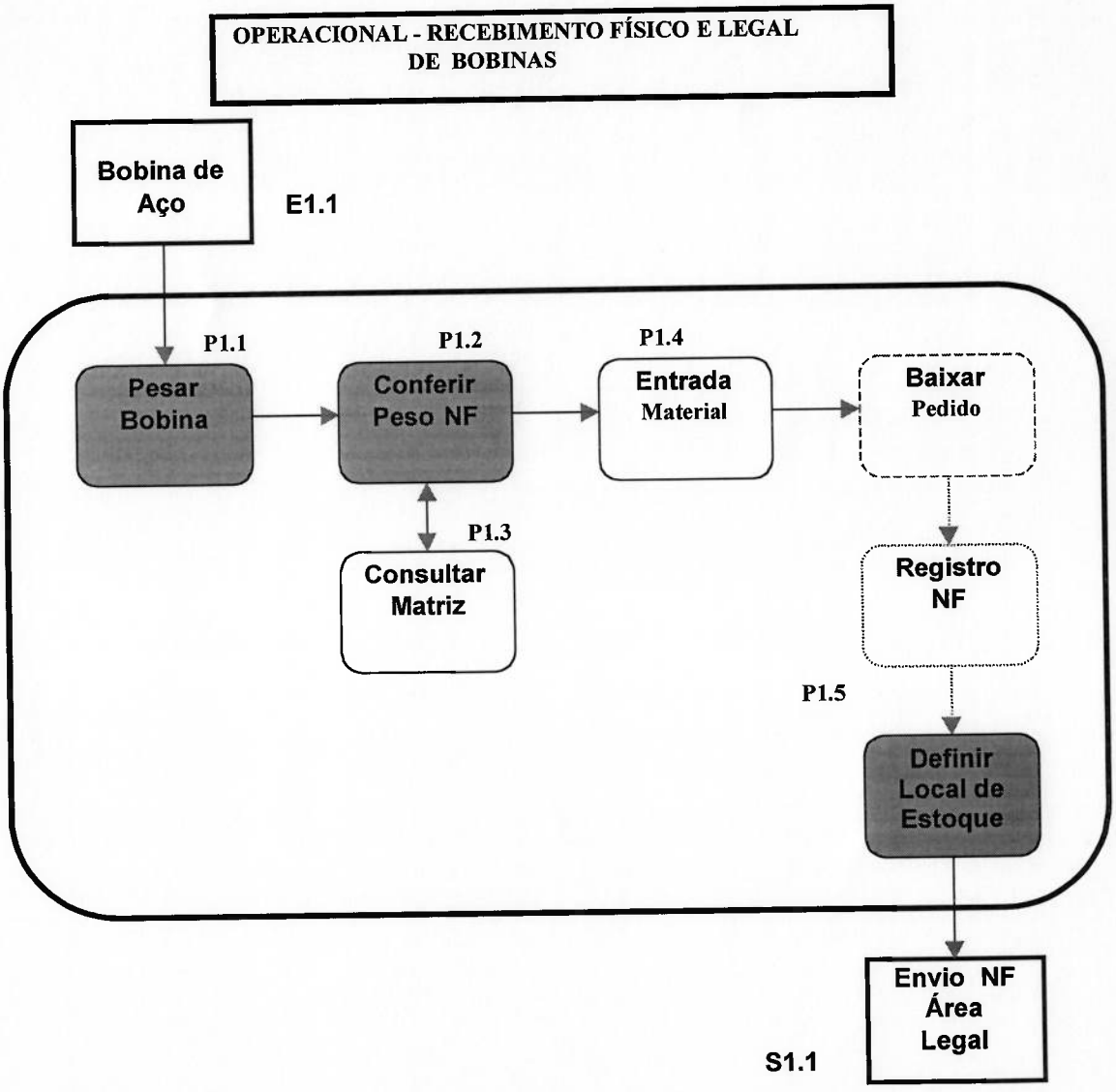


Figura 33: Sistemógrafo Operacional: Recebimento físico e legal das bobinas de aço.

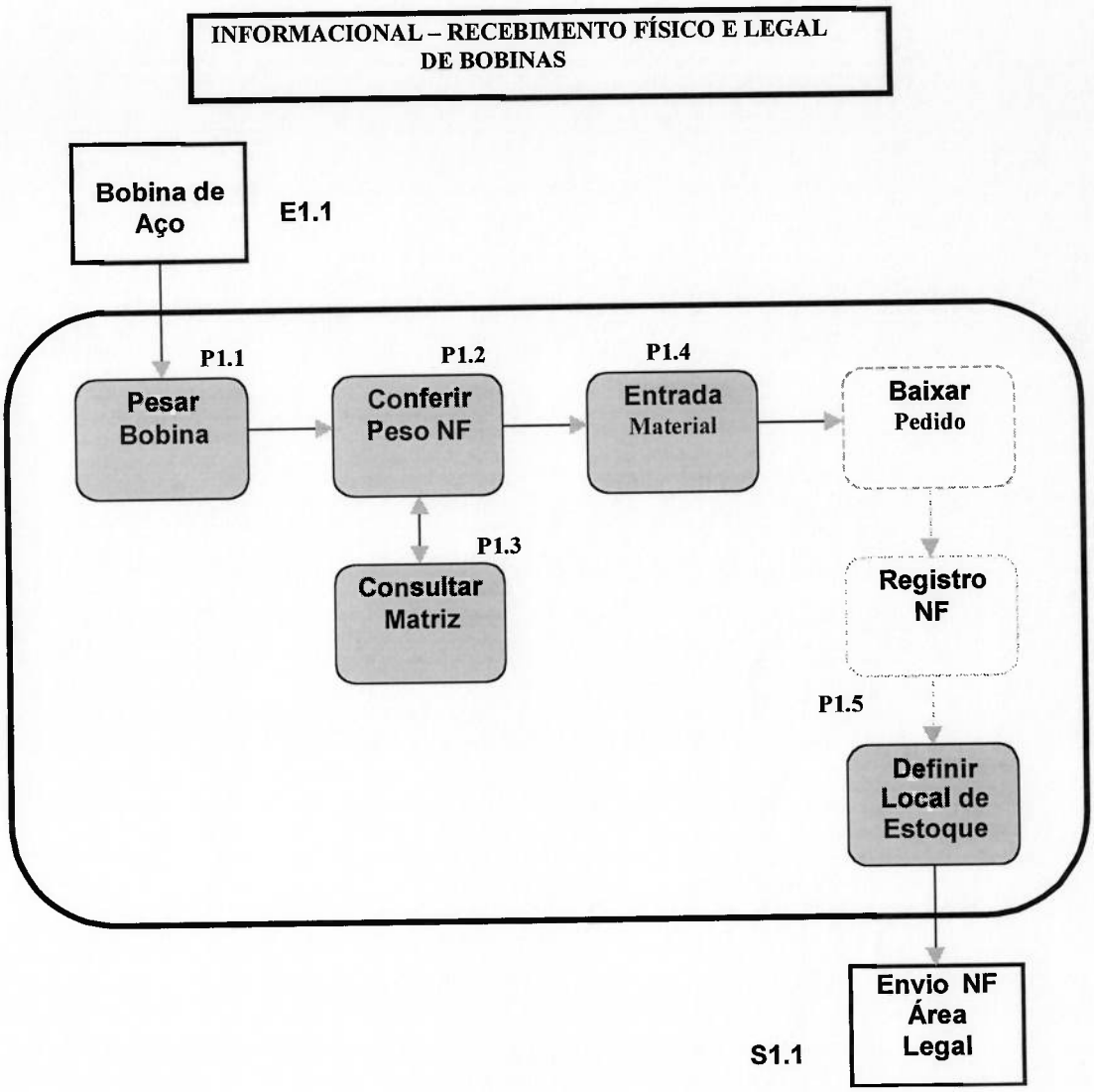


Figura 34: Sistemógrafo Informacional: Recebimento físico e legal das bobinas de aço

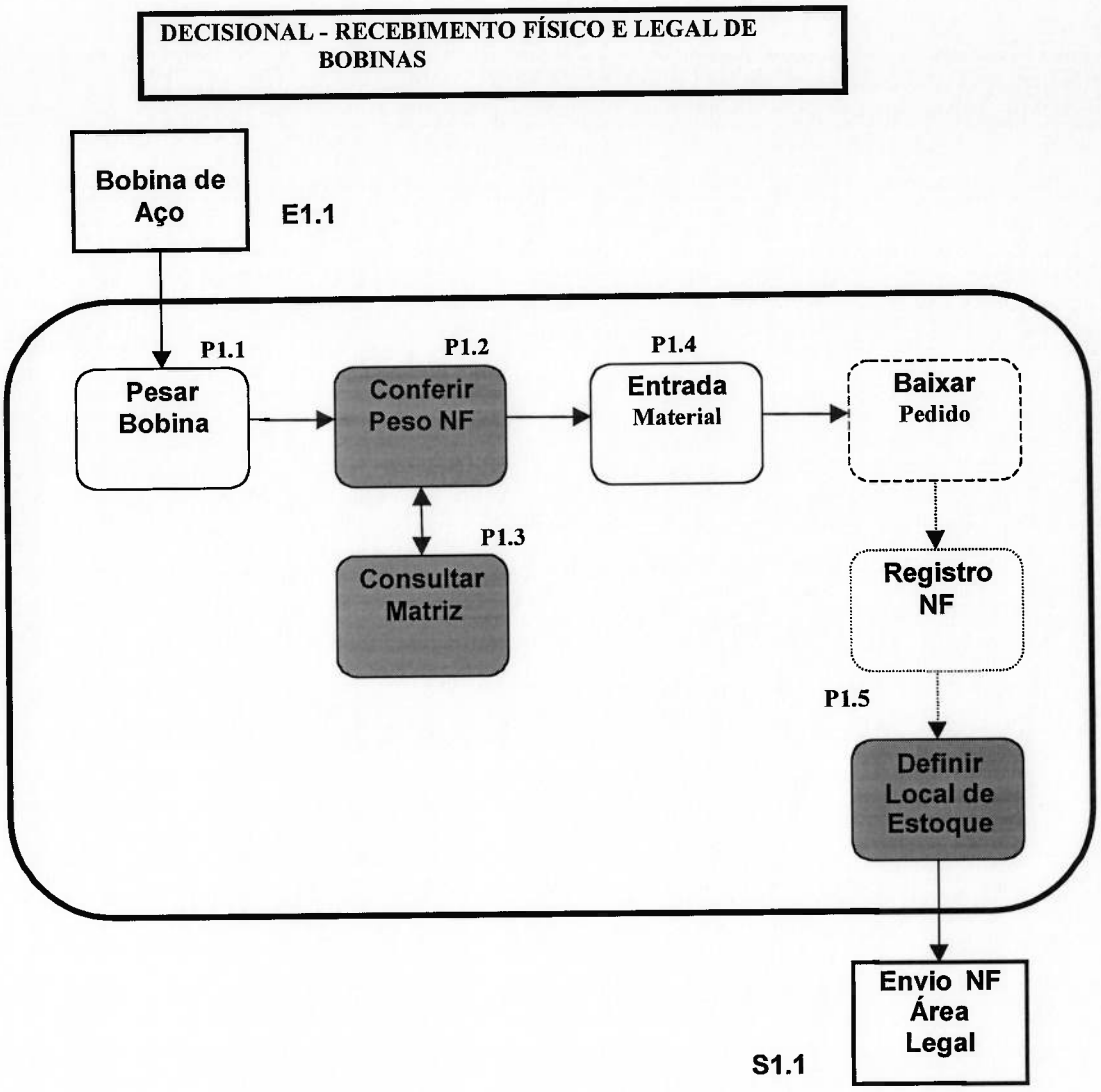


Figura 35: Sistemógrafo Decisional: Recebimento físico e legal das bobinas de aço

b) Recebimento físico de blank soldado

b.1) Componentes do Sistema:

Entradas:

E2.1 – Blank Soldado : A empresa recebe o blank soldado do centro de serviço para ser fornecido ao cliente.

Saída:

S2.1 – Envio de Nota Fiscal para Área Fiscal : Após a entrada da nota fiscal na empresa é enviada a primeira via para a área fiscal.

Processadores:

P2.1 - Entrada do Blank Soldado com referência ao Pedido do Corte: O funcionário responsável pelo recebimento do blank soldado através do sistema implantado, digita os dados da Nota Fiscal (número de pedido de corte, lote das bobinas beneficiadas, peso do blank, peso das bobinas beneficiadas, dados de impostos, características físicas) e o sistema, automaticamente faz a baixa do pedido, baixa das bobinas em poder do fornecedor e cria estoque de blank.

Categoria: operacional e informacional

Tipo : tempo, forma e espaço

Nível: 4

P2.2 – Consultar Programador de Corte: Caso ocorra discrepância de peso entre a Nota Fiscal o pedido (à maior ou menor do aceitável), é contatado o programador de corte para resolver o problema. (Restrição automática do sistema para peso maior).

Categoria: informacional e decisional

Tipo : tempo e forma

Nível: 6

b.2) Sistemógrafo Operacional:

A figura 36 representa o fluxo operacional existente no processo Recebimento físico do blank soldado. O processador P2.1 ingressa os dados da nota fiscal (número de pedido de corte, lote das bobinas beneficiadas, peso do blank, peso das bobinas beneficiadas, dados de impostos, características físicas) e, automaticamente cria o estoque.

b.3) Sistemógrafo Informacional:

A figura 37 representa o fluxo informacional existente no processo Recebimento físico do blank soldado. O processador P2.1 ingressa os dados da nota fiscal. Caso ocorra discrepância de peso entre a nota fiscal e o pedido, o processador P2.2 é acionado para solucionar eventuais os problemas

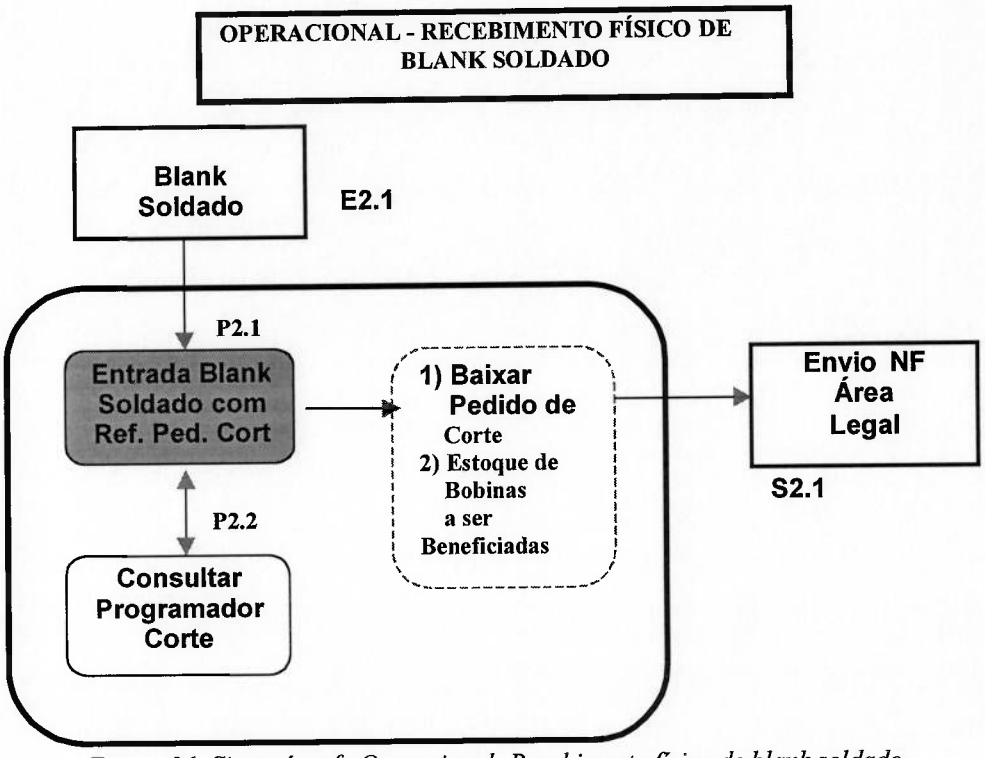


Figura 36: Sistemógrafo Operacional: Recebimento físico de blank soldado.

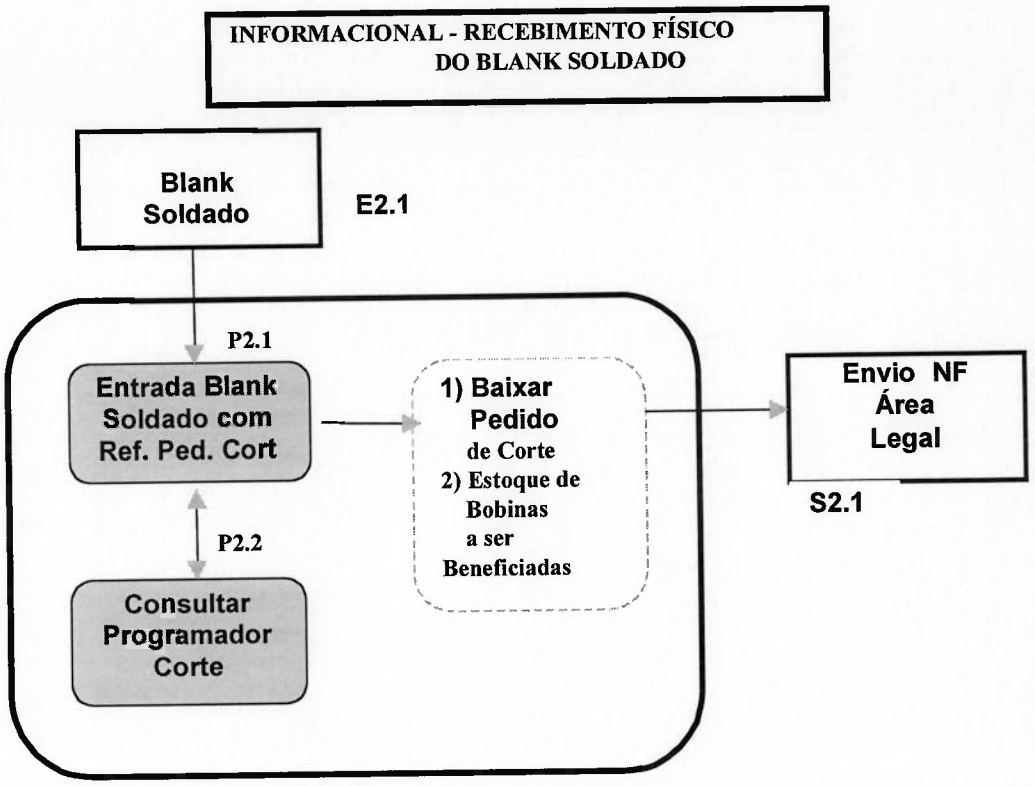


Figura 37: Sistemógrafo Informacional: Recebimento físico de blank soldado.

b.4) Sistemógrafo Decisional:

A figura 38 representa o fluxo decisional existente no processo Recebimento físico do blank soldado. O processador P2.2 resolve os problemas caso ocorra discrepâncias de peso entre a nota fiscal e o pedido de corte (restrição automática do sistema para pesos maior)

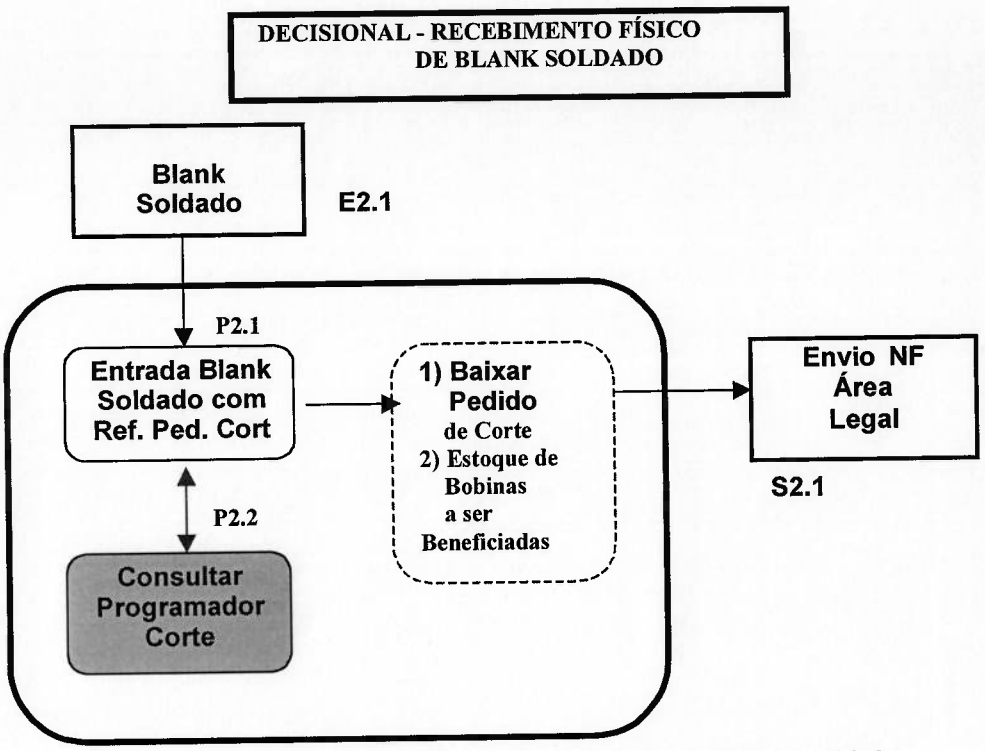


Figura 38: Sistemógrafo Decisional: Recebimento físico de blank soldado.

c) Recebimento legal de blank soldado

c.1) Componentes do Sistema:

Entradas:

E3.1 – Nota Fiscal: O documento legal é recebido com o blank soldado.

Saída:

S3.1 – Gravar Nota Fiscal: No caso de não existir observações na Nota Fiscal,

S3.1 – Glosar Nota Fiscal: No caso da Nota Fiscal estar errada..

Processadores:

P3.1 – Entrada de dados da Nota Fiscal: O funcionário responsável pelo recebimento legal do blank soldado, através do sistema implantado, digita os dados da Nota Fiscal.

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 4

P3.2 – Verificar Fatura: Caso o valor da Nota Fiscal seja igual ou menor que o do pedido, o sistema possibilita o registro da Nota Fiscal, no caso de discordância, é contatado o programador de corte.

Categoria: operacional, informacional e decisional

Tipo : tempo e forma

Nível: 5

P3.3 – Análise do Programador: Caso o valor da Nota Fiscal seja maior que o pedido, consulta-se o programador. Caso o erro esteja na Nota Fiscal, esta é glosada. Caso contrário o programador corrige o pedido para reiniciar a rotina.

Categoria: operacional informacional e decisional

Tipo : tempo forma e espaço

Nível: 6

c.2) Sistemógrafo Operacional:

A figura 39 representa o fluxo operacional existente no processo Recebimento legal do blank soldado.

O processador P3.1 digita os dados referente à nota fiscal recebida (data da nota, número de pedido, valor da nota) e o sistema retornará uma comparação entre o valor a ser pago pelo pedido e o valor da nota. O processador P3.2 verifica se o valor da nota fiscal é igual ou menor que o do pedido, logo esta é registrada. Caso o valor da nota fiscal seja maior que o pedido o processador P3.3 é consultado. Se o erro está na nota fiscal, esta é glosada, caso contrário o programador corrige o pedido para reiniciar a operação.

c.3) Sistemógrafo Informacional:

A figura 40 representa o fluxo informacional existente no processo Recebimento legal do blank soldado. O processador P3.1 registra os dados da nota fiscal. O processador P3.2 verifica se o valor da nota é igual ou menor que o do pedido, nesse caso a nota fiscal é registrada. O processador P3.3 é consultado se o valor da nota fiscal é maior que o pedido. Se o erro não está na nota fiscal, o programador corrige o pedido para reiniciar a operação, caso contrario, esta é glosada.

c.4) Sistemógrafo Decisional:

A figura 41 representa o fluxo decisional existente no processo Recebimento legal do blank soldado. O processador P3.3 é consultado se o valor da nota fiscal é maior que o pedido O programador corrige o pedido para reiniciar a operação, no caso do erro não estar na nota fiscal.

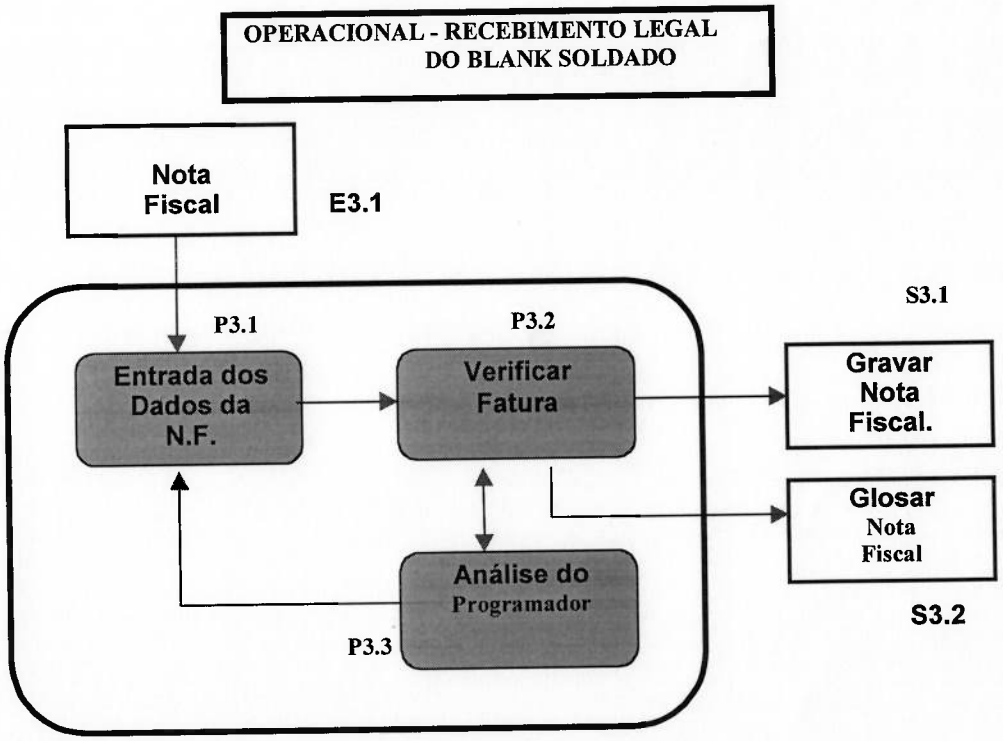


Figura 39: Sistemógrafo Operacional: Recebimento legal de blank soldado.

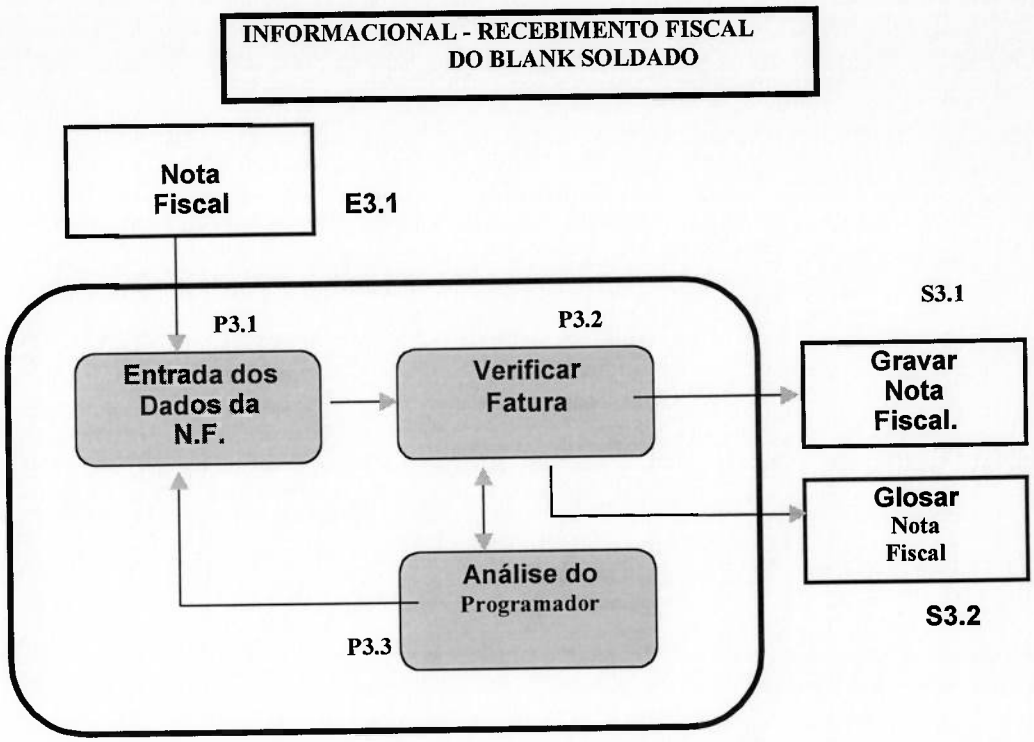


Figura 40: Sistemógrafo Informacional: Recebimento legal de blank soldado.

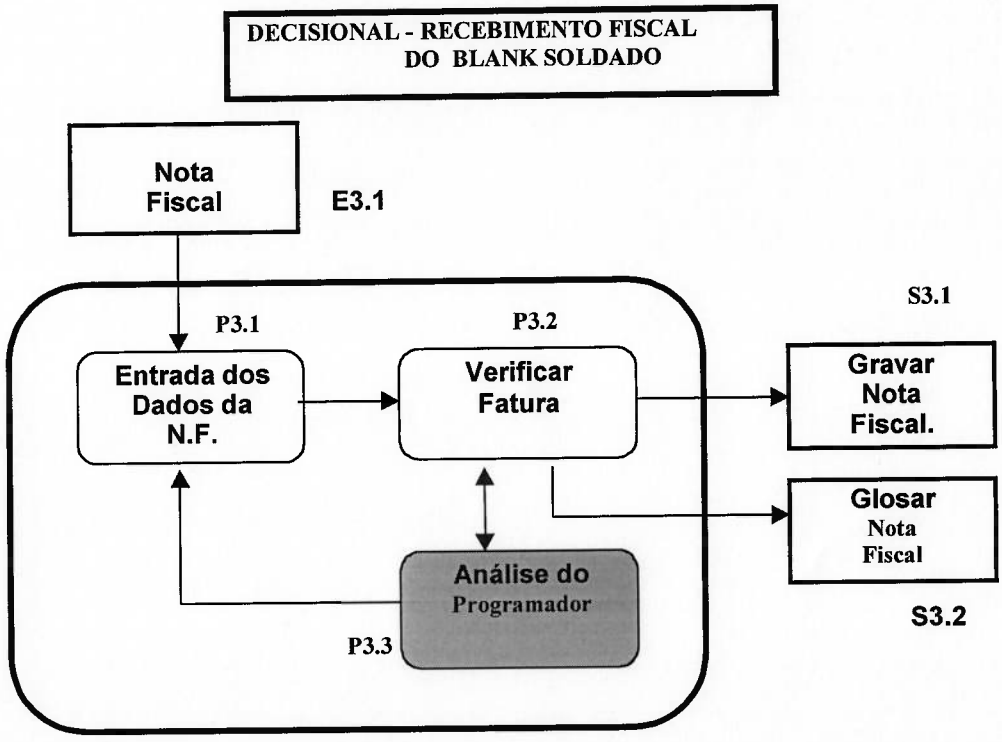


Figura 41: Sistemógrafo decisional: Recebimento legal de blank soldado.

d) Pedido de compra de Blank Soldado

d..1) Componentes do Sistema:

Entradas:

E4.1 – Programação de Entrega Blank Soldado do Cliente: O funcionário responsável, consulta o estoque e ao sistema de produção e realiza a programação de entrega de blank.

Saída:

S4.1 – Fornecedor: Executa a fabricação externa.

Processadores:

P4.1 – Cria Pedido de Corte de Blank Soldado: O funcionário informa as bobinas a serem cortadas, quantidades, medidas, preço e data de entrega do blank soldado. Emprega-se a programação de entrega de blank soldado do cliente, para similarmente fazer o pedido de fabricação para o fornecedor.

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 4

P4.2 – Aprovação do Pedido: O gerente de área aprova o pedido de compra após da análise. Esse é impresso e enviado ao fornecedor/fabricante.

Categoria: informacional e decisional

Tipo : tempo e forma

Nível: 6

d.2) Sistemógrafo Operacional:

A figura 42 representa o fluxo operacional existente no processo Pedido de compra do blank soldado. O processador P4.1 seleciona as bobinas a serem cortadas, assim como as quantidades, medidas, preço e data de entrega. Utiliza-se a programação de entrega de blank soldado do cliente para similarmente fazer o pedido de fabricação para o fornecedor.

d.3) Sistemógrafo Informacional:

A figura 43 representa o fluxo informacional existente no processo Pedido de compra do blank soldado. O processador P4.1 informa as bobinas a serem cortadas, assim como as quantidades, medidas, preço e data de entrega. Emprega-se a programação de entrega de blank soldado do cliente para fazer o pedido de fabricação, para o fornecedor. O processador 4.2 aprova o pedido de compra, esse é impresso e enviado ao fornecedor para sua fabricação.

d.4) Sistemógrafo Decisional:

A figura 44 representa o fluxo decisional existente no processo Pedido de compra do blank soldado. O processador P4.2 avalia o pedido de compra, que pode ser aprovado depois de um análise. Esse é enviado ao fornecedor para sua posterior fabricação.

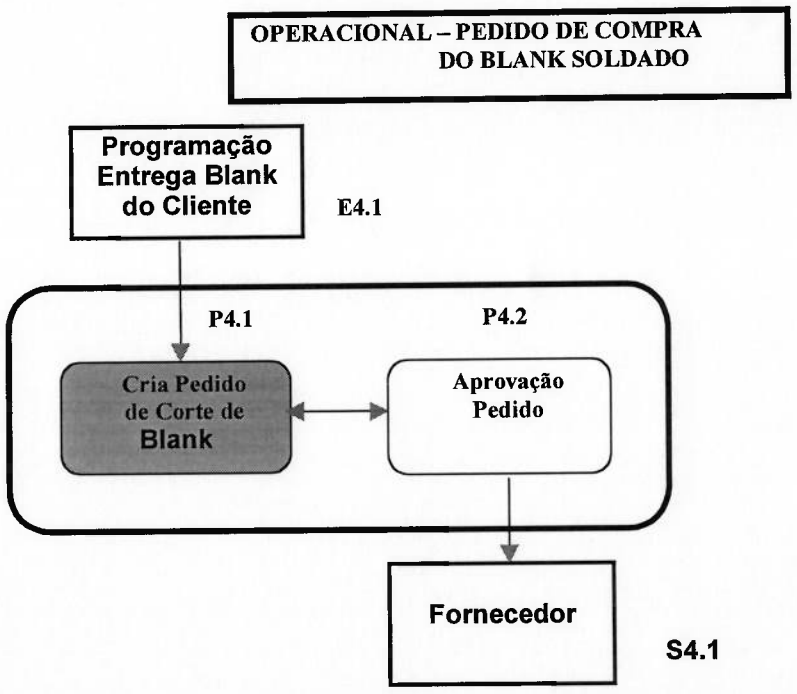


Figura 42: Sistemógrafo Operacional: Pedido de Compra do blank soldado

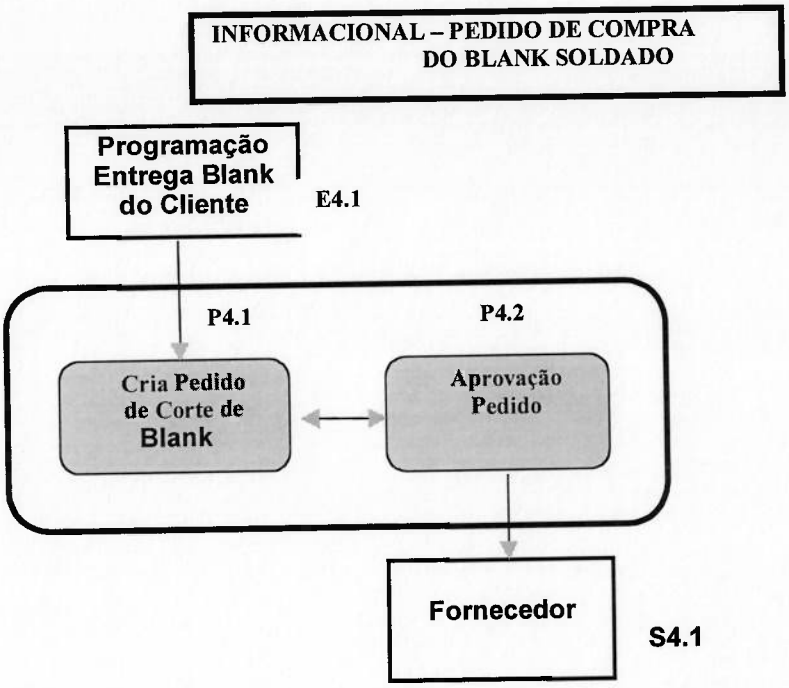


Figura 43: Sistemógrafo Informacional: Pedido de compra de blank soldado.

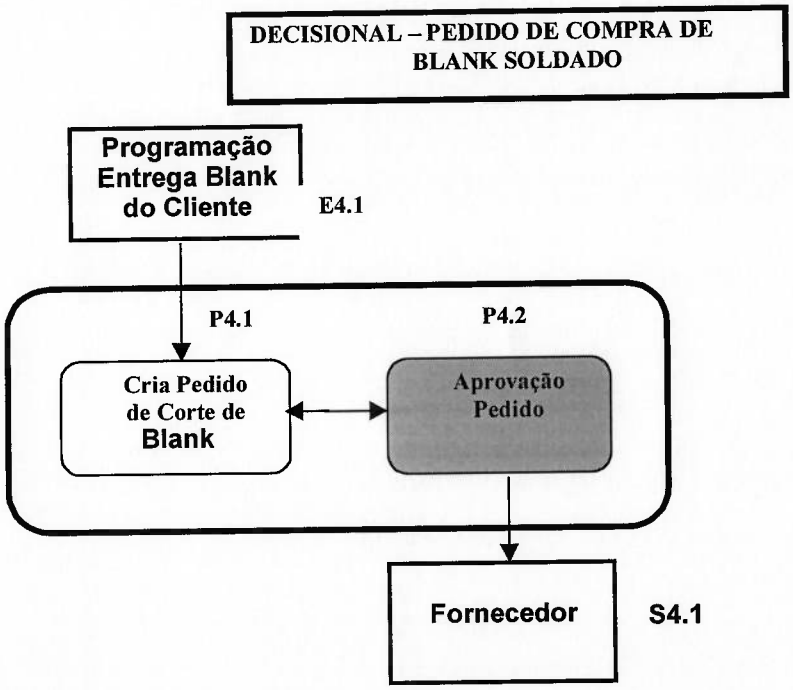


Figura 44: Sistemógrafo Decisional: Pedido de compra de blank soldado.

e) Pedido de Transferência

e.1) Componentes do Sistema:

Entradas:

E5.1 – Programação de entrega Blank soldado do cliente: O funcionário responsável, consulta ao sistema de produção e o estoque, realiza a programação de recebimento de blank no estoque.

Saída:

S5.1 – Produção de Bobinas Aço pela Matriz: Fabrica as bobinas de aço programadas.

Processadores:

P5.1 – Verificar as bobinas em estoque: O funcionário responsável verifica o estoque para escolher quais bobinas deverão ser produzidas.

Categoria: operacional informacional e decisional

Tipo : tempo e forma

Nível: 5

P5.2 – Fazer Pedido de Transferência: – O funcionário responsável preenche os campos do pedido de transferencia para produção de bobina pela matriz (material, quantidade, data de entrega).

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 4

e.2) Sistemógrafo Operacional:

A figura 45 representa o fluxo operacional existente no processo Pedido de transferência. O processador P5.1 faz um inventario do estoque, e escolhe as bobinas que deverão ser produzidas pela matriz. O processador 5.2 preenche os campos do pedido de transferencia para a produção das bobinas pela matriz.

e.3) Sistemógrafo Informacional:

A figura 46 representa o fluxo informacional existente no processo Pedido de transferência. O processador P5.1 informa quais são as bobinas insuficientes no estoque, e que deverão ser produzidas pela matriz. O processador 5.2 depois do informa do inventario, informa quais as bobinas, em quantidade e prazo a serem produzidas pela matriz.

e.4) Sistemógrafo Decisional:

A figura 47 representa o fluxo decisional existente no processo Pedido de transferência. O processador P5.1 depois do informe do inventario do estoque, decide quais bobinas devem ser produzidas pela matriz.

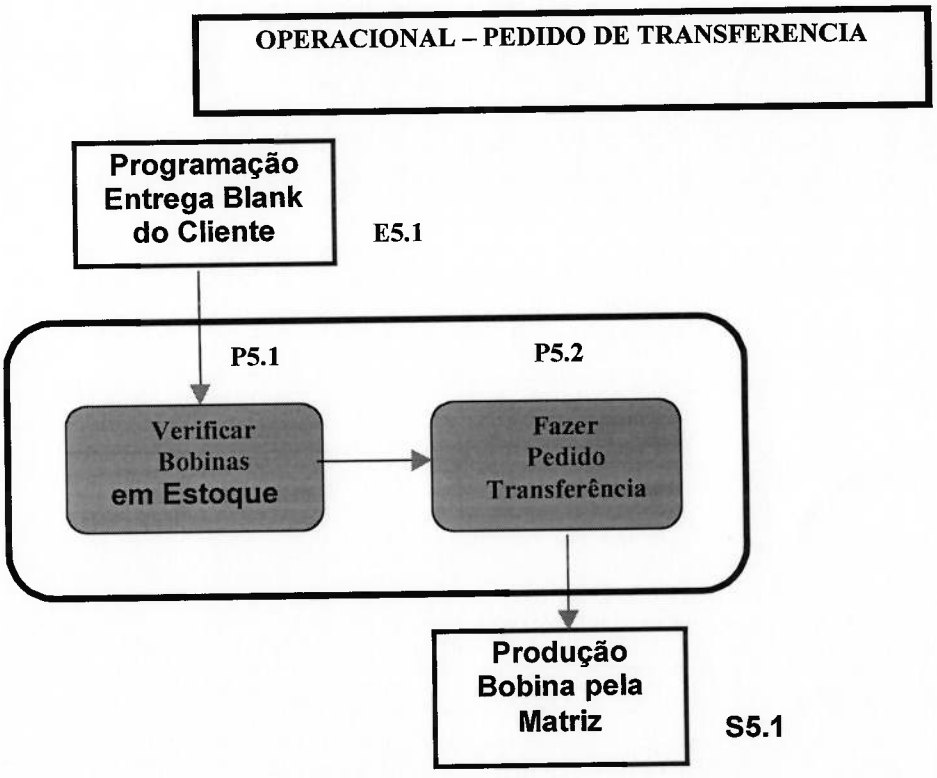


Figura 45: Sistemógrafo Operacional: Pedido de Transferência de bobina.

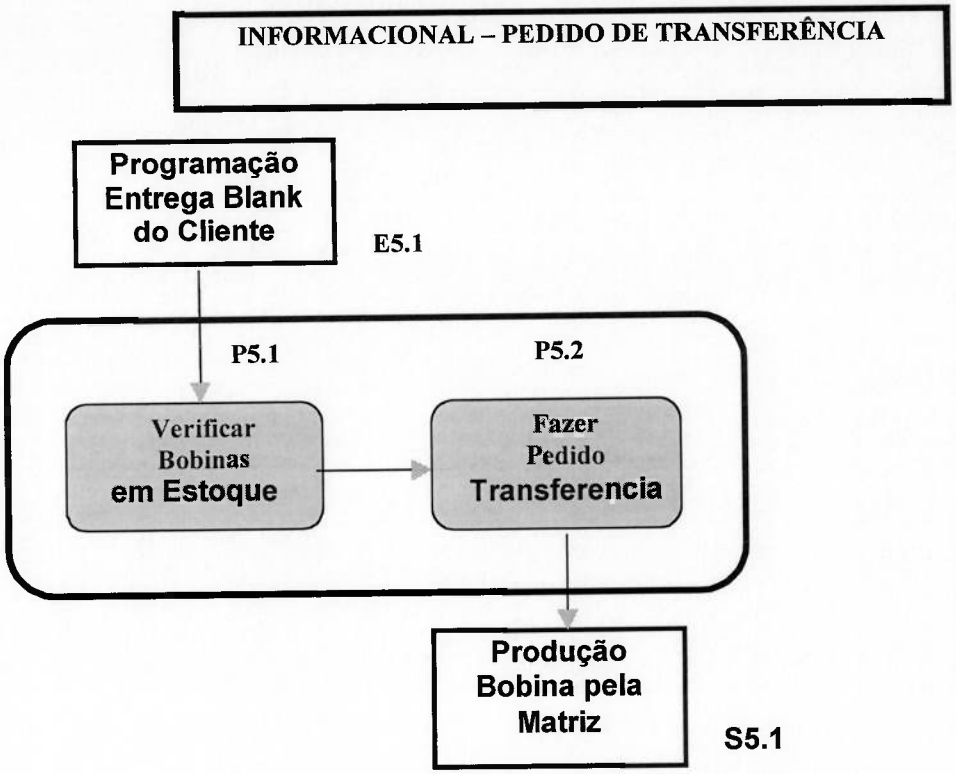


Figura 46: Sistemógrafo Informacional: Pedido de transferência de bobina.

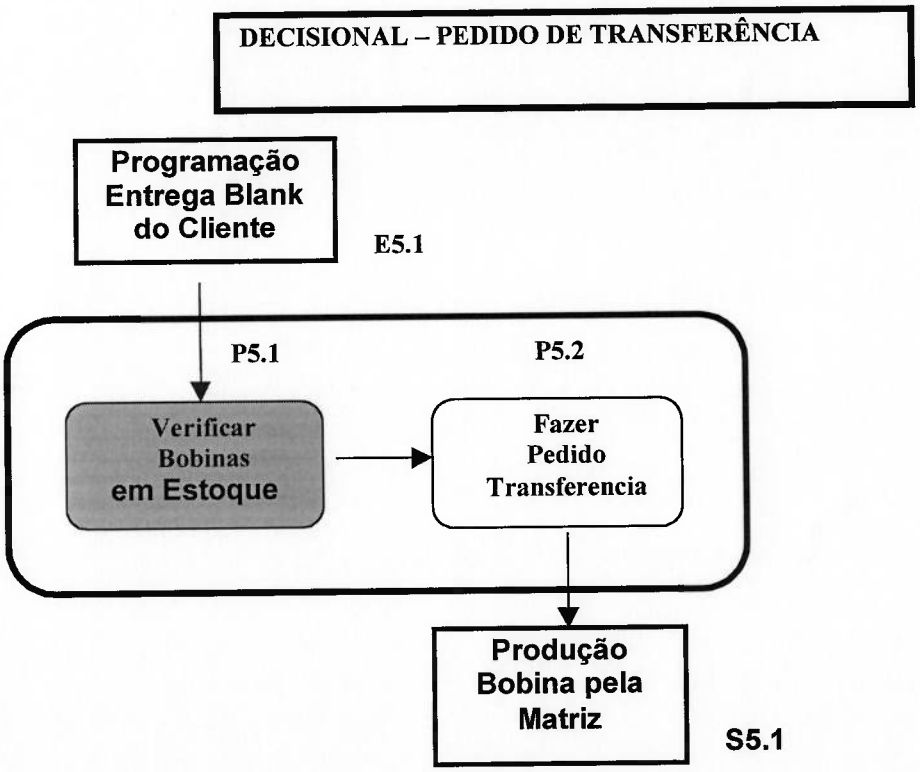


Figura 47: Sistemógrafo Decisional: Pedido de transferência de bobina.

f) Remessa de bobinas de aço para beneficiamento

f.1) Componentes do Sistema:

Entradas:

E6.1 – Programação de Entrega Blank Soldado do Cliente: O funcionário responsável, consulta ao sistema de produção e o estoque, realiza a programação de recebimento de blank no estoque.

Saída:

S6.1 – Gerar Nota Fiscal de Beneficiamento: O funcionário responsável digita os dados das bobinas de aço a ser beneficiadas.

Processadores:

P6.1 – Verificar Bobina de Aço em Estoque: O funcionário responsável verifica estoque para escolher quais bobinas serão beneficiadas.

Categoria: operacional informacional e decisional.

Tipo : tempo e forma

Nível: 5

P6.2 – Saída do Material: O funcionário responsável preenche os campos do pedido para beneficiamento, informando a etiqueta das bobinas.

Categoria: operacional e informacional

Tipo : tempo e forma

Nível: 4

f.2) Sistemógrafo Operacional:

A figura 48 representa o fluxo operacional existente no processo Remessa de bobina para beneficiamento. O processador P6.1 faz um inventario do estoque, e escolhe as bobinas que serão beneficiadas pelo fornecedor. O processador 6.2 preenche os campos do pedido de beneficiamento informando a etiqueta e peso das bobinas.

f.3) Sistemógrafo Informacional:

A figura 49 representa o fluxo informacional existente no processo Remessa de bobina para beneficiamento.

O processador P6.1 informa as bobinas que serão beneficiadas pelo fornecedor. O processador P6.2 depois do informe do inventario faz a baixa no estoque e credita o estoque disponibilizado ao fornecedor. Essas informações automaticamente compõem a nota fiscal a ser gerada.

f.4) Sistemógrafo Decisional:

A figura 50 representa o fluxo decisional existente no processo Remessa de bobina para beneficiamento. O processador P6.1 depois do informe do inventario do estoque, decide as bobinas que deverão ser beneficiadas pelo fornecedor.

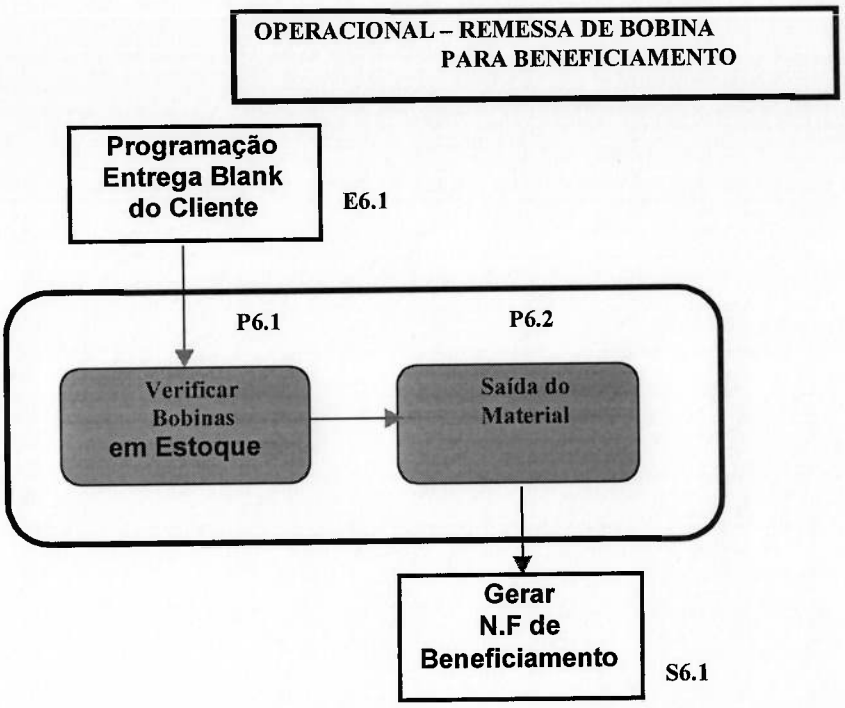


Figura 48: Sistemógrafo Operacional: Remessa de bobina para beneficiamento.

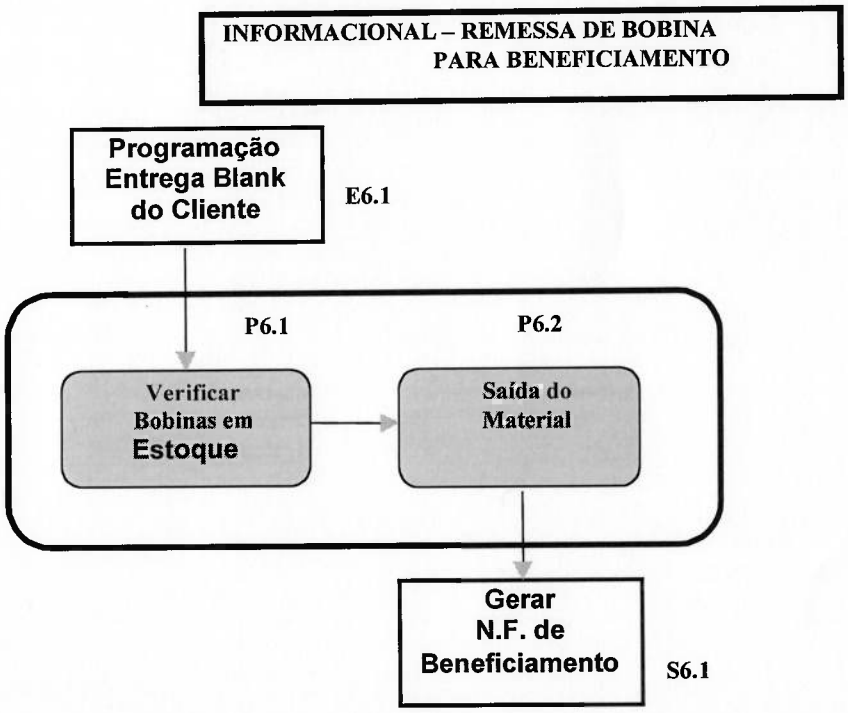


Figura 49: Sistemógrafo Informacional: Remessa de bobina para beneficiamento.

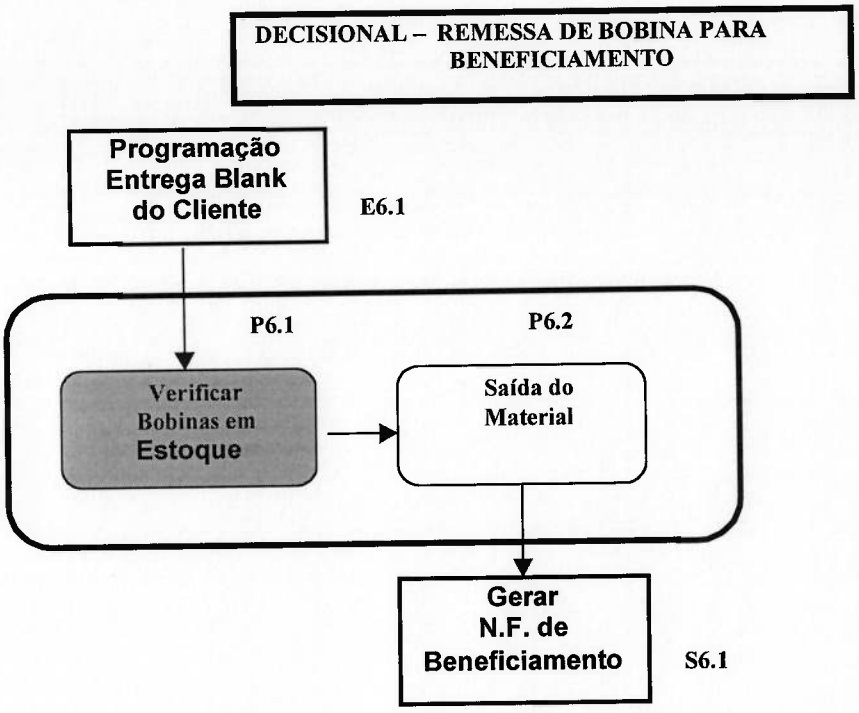


Figura 50: Sistemógrafo Decisional: Remessa de bobina para beneficiamento.

Processador	Operacional	Informacional	Decisional	Nivel	Espaço	Tempo	Forma
P1.1 Pesar bobinas de Aço	X	X		4	X	X	
P1.2 Conferir Peso Nota Fiscal	X	X	X	5		X	X
P1.3 Consultar Matriz		X	X	6		X	X
P1.4 Entrada de Material no estoque	X	X		4		X	X
P1.5 Definir Local de Estoque	X	X	X	5	X	X	X
P2.1 Entrada do Blank Soldado	X	X		4	X	X	X
P2.2 Consultar programador de corte		X	X	6		X	X
P3.1 Entrada dos dados Nota Fiscal	X	X		4		X	X
P3.2 Verificar Fatura	X	X	X	5		X	X
P3.3 Analise do Programador Corte	X	X	X	5	X	X	X
P4.1 Cria Pedido Corte de Blank S.	X	X		4		X	X
P4.2 Aprovação do Pedido.		X	X	6		X	X
P5.1 Verificar Bobinas no Estoque	X	X	X	5		X	X
P5.2 Fazer Pedido de Transferencia	X	X		4		X	X
P6.1 Verificar Bobinas no Estoque	X	X	X	5		X	X
P6.2 Saída do Material.	X	X		4		X	X

Tabela 4 – Tabela Comparativa entre Processadores do Sistema Atual [Salazar99].

Apesar da empresa em estudo possuir um sistema de informação integrado ERP, este apresenta falhas de comunicação entre os processos de vendas e compras (gerar um pedido de compra sem vínculo com ordem de venda), e de vendas e de transferência (gerar um pedido de transferência sem vínculo com ordem de venda).

A análise da tabela 4 , que demonstra e compara os processadores da situação atual e os sistemógrafos das sub-áreas envolvidas, permite observar que os processos apresentam um médio a alto grau de otimização por apresentar níveis acima de 4, o que significa dizer que as falhas possíveis de acontecer terão uma propagação veloz no processo, acarretando num alto grau de retrabalho e perda de recursos existentes. Dessa forma faz-se necessário um reestudo do processo como um todo para correção dessas falhas, para que a otimização reduza não só o tempo de processo, mas também os custos com os recursos envolvidos.

O sistemógrafo apresentado é a base para a modelagem AV/AO do sistema informacional, apresentada na seção seguinte, de forma a otimizar a integração entre os sistemas visando minimizar, ou até, eliminar as falhas de comunicação existente nos processos atuais.

6.3 Modelagem AV/AO

A modelagem se dará com duas abordagens, a saber, Top-Down e Bottom-Up. Inicialmente é necessário visualizar o processo de um ponto de vista mais abstrato, para em seguida refiná-lo até o nível desejado. Após se atingir esse nível, os processos principais serão gerados por meio de uma base de sub-processos existentes. Nesse exemplo, não iremos abordar as áreas de inventário, custos, contabilidade, contas ã pagar e receber, recursos humanos.

A visão geral do processo está apresentada na Figura 51 abaixo.

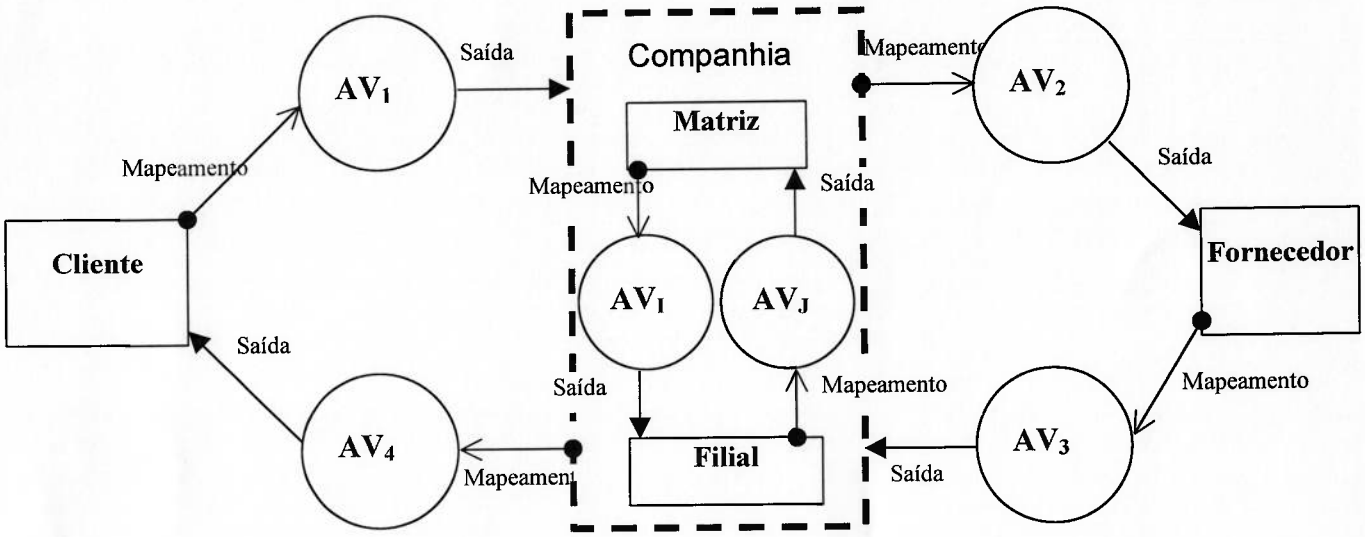


Figura 51: Visão Geral do processo através da modelagem AV/AO.

Note que os AO's representam as empresas envolvidas, enquanto os AV's modelam a interface entre elas.

O $AO_{Cliente}$ informa através do AV_1 o programa de remessas de blank desejado, como os pagamentos das entregas já realizadas, enquanto o AV_4 responsabiliza-se pelas informações das remessas de blank e nota fiscal para o Cliente, mapeados do $AO_{Companhia}$.

O AV_2 trata de todas as informações a serem passadas do $AO_{Companhia}$ para o $AO_{Fornecedor}$, ou seja, os pedidos de corte de chapa, as remessas para beneficiamento com

suas notas fiscais e o pagamento dos serviços realizados. O AV_3, por sua vez, informa as entregas de blank beneficiados e as notas fiscais dos serviços de corte, mapeados do $AO_{Fornecedor}$.

Ao se aplicar as regras de consistência, constata-se que para se criar um pedido de corte de chapa é necessário existir um programa de remessa e uma remessa de beneficiamento com nota fiscal. O pagamento dos serviços realizados só ocorre após a entrega do blank, nota fiscal dos serviços de corte, referenciados a um pedido de corte. O pagamento pelo produto entregue ao cliente ocorre após a remessa e faturamento do mesmo.

Pode-se refinar esse esquema separando o $AO_{Companhia}$ como sendo dois AO's referentes aos AO_{Matriz} e AO_{Filial} com dois AV's correspondente ao relacionamento entre eles.

Assim sendo, o AV_J refere-se ao Pedido de Transferência de bobina do AO_{Filial} para o AO_{Matriz}, e o AV_I diz respeito à remessa de bobina e sua nota fiscal de transferência. A consistência a ser cumprida é se gerar pedidos de transferência em função dos programas de remessa existentes advindos do AV_I. O envio de bobinas do AV_I deve ser vinculado ao pedido de transferência indicado no AV_J.

As interfaces entre as áreas devem ser consistentes com respeito a todo o sistema. O programa de remessas do cliente é comunicado à área de Vendas, a qual informa à Compras quais os produtos devem ser adquiridos do fornecedor. Para se iniciar o serviço de corte a área de compras precisa verificar o estoque da(s) bobina(s) que compõe(m) o blank. Caso não haja bobina em estoque, a área de compras inicia o processo de transferência por meio de um pedido de transferência à matriz. O fluxo de material é apresentado a seguir nas Figuras 52 e 53.

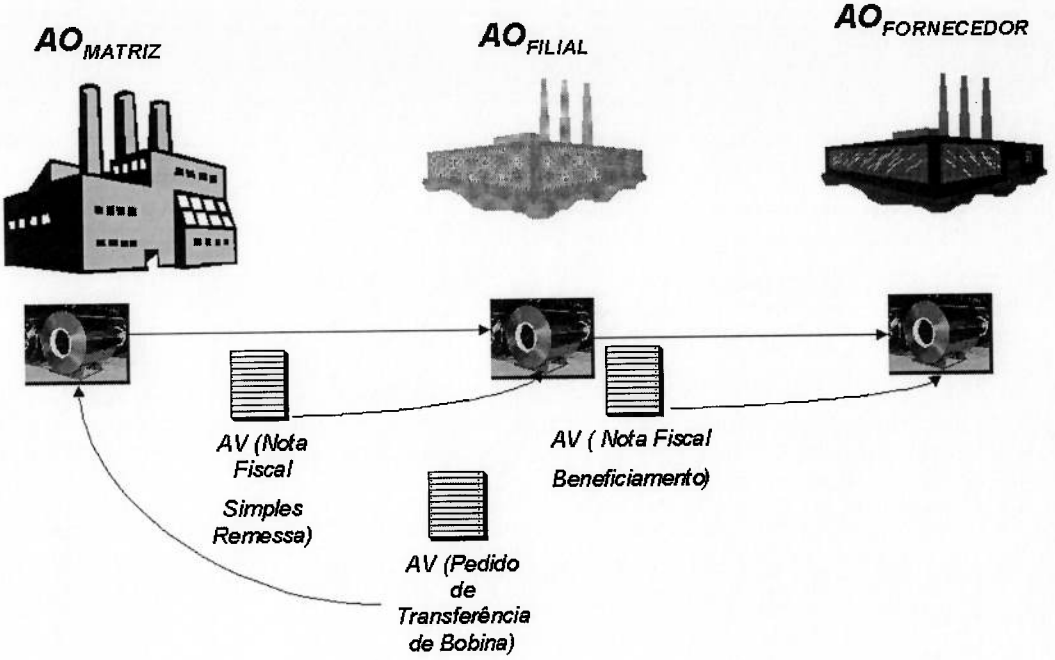


Figura 52: Fluxo de bobina entre AO_{Matriz}, AO_{Filial} e $AO_{Fornecedor}$

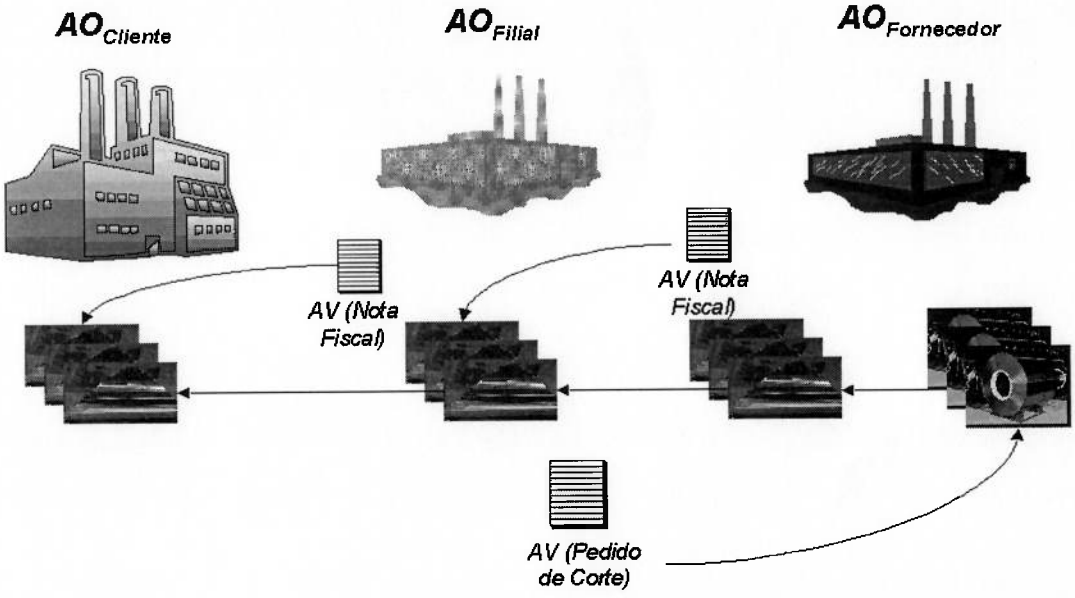


Figura 53: Fluxo de Blank entre $AO_{Cliente}$, AO_{Filial} e $AO_{Fornecedor}$

6.3.1. Programação no RLOG

Essa modelagem se inicia com 03 (três) AO: $AO_{cliente}$, $AO_{companhia}$ e $AO_{fornecedor}$. Cada AO terá um AV correspondente, que enviam mensagens entre si. $AO_{cliente}$ possui o AV_1, que transmite as informações dos programas de remessa de blank e os pagamentos a serem enviados para o $AO_{companhia}$ através do AV_4, bem como recebe as dados sobre os produtos e notas fiscais remetidos pelo $AO_{companhia}$. O $AO_{companhia}$ possui o AV_4 para receber e enviar as mensagens vindas do AV_1, e o AV_2 para se comunicar com o AV_3, responsável por enviar e receber informações do $AO_{fornecedor}$. As mensagens trocadas pelo AV_2 e AV_3 referem-se ao programa de corte de bobinas, pagamentos dos produtos recebidos, produtos e notas fiscais dos blanks. A Figura 54 representa o esquema inicial.

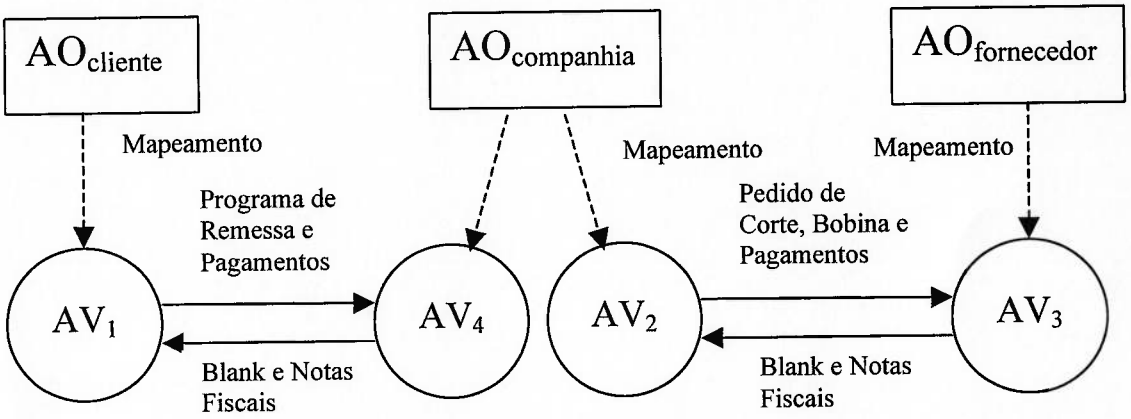


Figura 54: Modelo AV/AO no nível macro dos processos.

Nota-se que a forma de representação dos AV's são idênticas. Dessa forma é possível reutilizá-las, utilizando o mesmo esquema.

A base de dados no RLOG referente a Figura 54 está apresentada abaixo:

```
Relation AO 's Attributes:      (
Nome:  String(16);
Dados: { String(16)} * Conjunto sem limitação máxima *
The keys are the following attributes: Nome .)

Relation AV 's Attributes:      (
Entrada:      String(16);
Saida: String(16);
AO_En: String(16);
AO_Sa: String(16)
The keys are the following attributes: Entrada .)
```

```
Relation AO 's facts:
( Nome: "Cliente"; Dados:{"Programa_Remessa";"Pagamento";"Nota_Fiscal";"Remessa
        ;"Blank"})
( Nome: "Companhia"; Dados:{"Programa_Remessa"; "Pedido_Corte"; "Pagamento"
        ;"Nota_Fiscal" ;"Bobina"; "Blank"})
( Nome: "Fornecedor"; Dados:{"Pedido_Corte";"Pagamento";"Nota_Fiscal";"Bobina"
        ;"Blank"})

Rules available are:
AV("Nota_Fiscal","Pagamento","Cliente","Companhia") :- AO("Cliente",<"Pagamento">),
        AO("Companhia",<"Nota_Fiscal">)
AV("Nota_Fiscal","Pagamento","Companhia","Fornecedor") :-
        AO("Companhia",<"Pagamento">), AO("Fornecedor",<"Nota_Fiscal">)
AV("Programa_Remessa","Blank","Companhia","Cliente") :- AO("Companhia",<"Blank">),
        AO("Cliente",<"Programa_Remessa">)
AV("Pedido_Corte","Blank","Fornecedor","Companhia") :- AO("Fornecedor",<"Blank">),
        AO("Companhia",<"Programa_Remessa">)
AV("Bobina","Blank","Fornecedor","Companhia") :- AO("Fornecedor",<"Blank">),
        AO("Companhia",<"Bobina">)
```

As relações de coerência estão descritas nas regras (rules), que identificam as correlações entre os dados, a saber, para se ocorrer uma remessa de "blank" é necessário a existência de um "programa de remessas", para que um "pagamento" se efetue, a "nota fiscal" deve ser recebida, para que um "blank" seja produzido é necessário que seja enviado uma "bobina" e que exista um "pedido de corte". A reutilização está intrínseca ao processo, visto que a estrutura é a mesma para todos os objetos do tipo AV, quer dizer, o mapeamento e relacionamento entre os AO´s têm o mesmo esquema.

Algumas atividades não estão representadas nesse nível de implementação, sendo assim é necessário refinar modelo de forma a acrescentar a figura da Matriz, responsável pela produção das bobinas e da filial, responsável pelo gerenciamento das atividades com o "fornecedor" e o "cliente" da "companhia".

A Figura 55 apresenta o modelo refinado, acrescentando a Matriz e a Filial na Companhia.

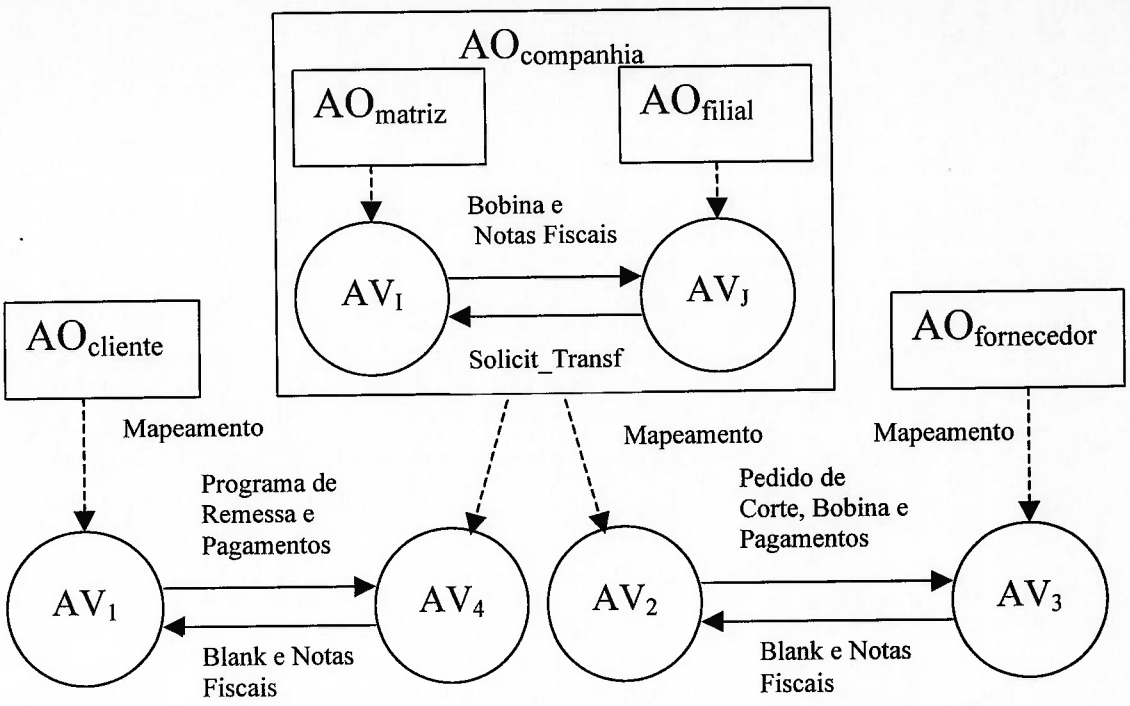


Figura 55: Modelo AV/AO refinado para acrescentar Matriz e Filial

Para programar uma base de dados no RLOG referente ao modelo refinado, é preciso modificar o *Schema* do AO, acrescentando um campo referente às partes que o compõe, como escrito abaixo:

```
Relation AO 's Attributes:      (
Nome:  String(16);
Dados:  { String(16)}; * Conjunto sem limitação máxima *
Parte_de:  { String(16)} * Conjunto sem limitação máxima *
The keys are the following attributes: Nome  .)
```

Com essa modificação é possível inserir *Facts* referentes à Matriz e a Filial, conforme descritos abaixo:

```
Relation AO 's facts:
( Nome: "Matriz"; Dados:{"Bobina";"Solic_transf";"Nota_Fiscal"}; Parte_de:{"Companhia"})
( Nome: "Filial"; Dados:{"Bobina";"Solic_transf";"Nota_Fiscal"}; Parte_de:{"Companhia"})
```

As regras de consistência referente aos AV_I e AV_J são:

```
AV("Solic_Transf","Bobina","Matriz","Filial") :- AO("Matriz",<"Bobina">),
        AO("Filial",<"Solic_Transf">).
```

Bibliografia

[Alencar94] **Alencar, P. S. C., Carneiro-Coffin M. F., Cowan, D. D. C., and Lucena C. J. P.** Towards a Formal Theory of Abstract Data Views. Technical Report 94--18, Computer Science Department, University of Waterloo, Waterloo, Ontario, Canada, April 1994.

[Asimow68] **Asimow, M.** Introdução ao projeto, 1ª ed., Editora Mestre, São Paulo, 1968.

[Bittencourt96] **Bittencourt, G.** Inteligência artificial: ferramentas e teorias, Campinas: Instituto de Computação, Unicamp, 1996.

[Booch91] **Booch, G.** Object Oriented Design with Applications. The Benjamin/Cummings Publishing Company, Inc., 1991.

[Bresciani97-a] **Bresciani Filho, E.** Sistema, Sistêmica e Sistemografia, monografia, UNICAMP, Puc-Campinas,1997.

[Bresciani97-b] **Bresciani, R.** Sistema de Qualidade de uma Empresa Industrial, dissertação de mestrado, UNICAMP,1997.

[Carretoni93] **Carretoni, E.** Administração de Materiais – Uma Visão Sistêmica, Editorial PSV, 1993, p.13 – 16.

[Chan93] **Chan, D.K.C., e Trinder, P.W.** Object comprehension: A query notation for object-oriented databases, Lectures notes in computer science nº752, p. 55-72., Springer, 1993.

[Chiavenato91] **Chiavenato, I.** Iniciação à Administração de Materiais, 1991, McGraw Hill , p. 35-39.

[Chung77] **Chung, C.C.** , Model Theory. 2nd ed., North-Holland Pub. Co., 1977.

[Coleman94] **Coleman, D. et al.** Object-Oriented Development: The Fusion Method, Prentice-Hall Object Oriented Series, Englewood Cliffs, NJ, 1994.

[Cowan95] **Cowan, D.D., Lucena, C.J.P.** Abstract data view: An interface specification concept to enhance design for reuse, IEEE Transactions of Software Engineering, vol. 21, nº3, march 1995.

[Dias96] **Dias, M. A. P.** Administração de Materiais, Altas 4ª Edição, 1996, p.10-22.

[Dieter91] **Dieter, G. E.** Engineering design : a materials and processing approach- 2nd ed., Mc Grall-Hill, 1991.

[Gamma95] **Gamma, E., Helm, R., Johnson, R., Vlissides, J.** Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley, 1995.

[Geiger95] **Geiger, M. Coremans, A.** Modern Sheet Metal Forming – Open to New Interdisciplinary Conceptions, Blech Rohre Profile, 42, 5, p. 284-8, 1995.

[Gero89] **Gero, J.S.** Metamodel: An integrated modeling framework for intelligent CAD, Artificial Intelligence in Design, Computacional Mechanics Publications, Southamption, p.429-449, 1989.

[Gerric89] **Gerrig, R.** Empirical Constraints on Computational Theories of Metaphor: Comments on Indurkhya, Cognitive Science, 13, 1989.

[Gray93] **Gray, P.M.D.** Knowledge reuse through networks of large KBS, Lectures notes in computer science n°752, p. 55-72., Springer, 1993.

[Hardie95] **Hardie, N.** *Complexity, Categories and Leadership.* ASQC Quality Congress Transactions, 49th Annual Quality Congress, 1995, p. 227-233.

[Hutchison94] **Hutchison, D.** Chaos Theory, Complexity Theory and Health Care Quality Management. Quality Progress, vol 27, no. 11, p.69-72, 1994.

[Indurkia86] **Indurkia, B.** Contrained semantic transference: A formal theory of metaphors, Sintese n°68, D. Reidel Publishing Company, 1986.

[Indurkia87] **Indurkhya, B.** Approximated Semantic Transference: A Computational Theory of Metaphors and Analogies, Cognitive Science, 11, 1987.

[Kintchner98] **Kintschner, F.** Metodologia de Reestruturação de administração de materiais em Empresa Industrial, Campinas, 1998. Disertação (Mestrado), Instituto de Informática, Pontifícia Universidade Católica de Campinas.

[Kintchner99] **Kintschner, F., Bresciani E.F.** Racionalização e informatização de área de administração de materiais, Campinas, UNICAMP, 1999, pp. 4-9.

[Leach96] **Leach, L.P.** TQM, Reengineering, and the Edge of Chaos, Quality Progress, vol. 29, n° 2, 1996, p. 85-9.

[Levesque86] **Levesque, H.J., Mylopoulos, J.** An overview of knowledge representation, On conceptual modeling, p. 3-17, Springer-Verlag, New York, 1986.

[Lemoinge90] **Le Moinge.** La modélisation des systèmes complexes. Paris, Editora Dunod, 1990.

[Liu96-a] **Liu, M.** ROL: A dedutive object base language, Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada, 1996a.

[Liu96-b] **Liu, M., Yu, W., Xing, M.** The ROL system user manual realease 2.1, University of Regina, Regina, Saskatchewan, Canada, May 1996b.

[Liu98-a] **Liu, M.** Relationlog: a Typed Extension to Datalog with Sets and Tuples, The Journal of Logic Programming, 1-30, 1998a.

[Liu98-b] **Liu, M., Shan, R.** The Relationlog System: User Manual, Release 1.0, University of Regina , August 1998b.

[Liu98-c] **Liu, M.** Logical Semantics and Language for Databases with Patial and Complete Tuples and Sets, In Proceedings of the 6th International Workshop on Deductible Database and Logic programming – DDLP'98 – Manchester, UK, pp 141-155, June, 1998c.

[Medland92] **Medland, A.J.** The computer-base design process- 2nd edition, Capman&Hall, London, 1992.

[Melan88] **Melan, E.H.** Focusing on The Process: Key to Quality Improviment, ASQC Quality Congress Transactions, 48th Annual Quality Congress, 1994, p. 170-717.

[MtlFrmHbk98] **Metal Forming Handbook / Schuler,** Berlin, Heidelberg, New York, Springer Verlag, 1998, pp. 310-313.

[Murdock94] **Murdock, M.** Extending The Boundaries of Quality in Primary Care, ASQC Quality Congress Transactions, 48th Annual Quality Congress, 1994, p. 710-717.

[Pahl86] **Pahl, G., e Beithz, W** Engineering design a systematic approach, Edited by Ken Wallace, London, 1986.

[Paton93] **Paton, W.N., Al-Qaimari, G., Doan, K.** On interface objects in object-oriented databases, Lectures notes in computer science nº752, p. 55-72., Springer, 1993.

[Pereira99] **Pereira, J.F.B., de Andrade, S.L., Rosa, L.K.** Projeto ULSAB Ajudando a Construir o Carro do Futuro, In: II WORKSHOP – QUALIFICAÇÃO DE CHAPAS PARA A INDÚSTRIA AUTOMOBILÍSTICA, São Paulo, 27 de Abril de 1999, PMC-EPUSP , pp. 20-31.

[Pincinini99] **Pincinini, M.F., Sampaio, A.P., Azambuja, S., Carvalho, J.R., Campbel, C.H., Carvalho, C.C., Suzuki, L.M., Dos Santos, J.C., Venturinnen, C.** Desenvolvimento Pioneiro de Blank Soldado na América Latina: O Projeto da CSN para a GMB e a Ford, In: II Workshop – Qualificação de Chapas Para a Indústria Automobilística, São Paulo, Abril 1999, PMC-EPUSP , p. 32-43.

[Reich95] **Reich, Y.** The study of design research methodology, Journal of mechanical design, pag. 211-214, vol. 117, june 1995.

[Rumbaugh91] **Rumbaugh, J. et al.** Object-Oriented Modeling and Design. Prentice Hall, 1991.

[Salazar99] **Salazar, C. E. V.** Sistemógrafo de um Sistema de Administração de Materiais para Blanks Soldados, Seminário apresentado na disciplina: Fundamentos da Engenharia dos Sistemas de Produção, ministrada pelo Prof. Dr. Ettore Bresciani Filho, na Faculdade de Mecânica, da Universidade Estadual de Campinas (UNICAMP), outubro de 1999.

[Silva92] **Silva, J. R.** Uma formalização para o processo de design baseado na teoria de metáforas: suas aplicações em automação de sistemas a eventos discretos, Tese de doutorado, Universidade de São Paulo, São Paulo, 1992.

[Silva94] **Silva, J. R.** An object-oriented approach to tha design of flexible manufacturing systems, IFIP, pag 91-106, 1994.

[Silva95] **Silva, J.R., Afsarmanesh, H., Cowan, D.D., Lucena, C.J.P.** An Object-Oriented Approach to the Design of Production Systems, in Balanced Automated Systems: Architecture and Design Methods, Camarinha-Matos, L. and Afsarmanesh, H. (eds.), IFIP, Chapman & Hall, pp.91-106, London, 1995.

[Silva96] **Silva, J.R., Miyagi, P.E.** Towards a Formal Approach to Functionality in the Design of Discrete Flexible Systems, ASIC'96, ICIMS-NOE, Toulouse, June, 1996.

[Silva98] **Silva, J.R.** Interative Design of Integrated Systems, in Inteligent Systems for Manufacturing: Munti-Agent Systems and Virtual Organizations, Camarinha-Matos, L., Afsarmanesh, H., Marik, V. (eds.), IFIP, -KAP, pp. 567-578, London, 1998.

[Snee93] **Snee, R.D.** Creating Robust Work Processes, Quality Progress, vol 26, n. 02, 1993, p.37-41.

[Shoderbek90] **Shoderbek, Shoderbek, Kefalas** Management System – Conceptual Considerations, BPI IRWIN, 1990, p.5-32.

[Tavares97] **Tavares, J.J.P.Z.S. Silva,J.R.** Fusão Entre Objetos e IA na Modelagem e Design de Sistemas Automatizados, III SBAI, p. 359-364 , 1997.

[Tomiyama97] **Tomiyama, T., Umeda, Y.** Functional Reasoning in Design, IEEE Expert, AI in Design, March-April, 1997.

[Umeda96] **Umeda, Y. et al.** Supporting Conceptual Design Based on the Function-Behaviour-State Modeler, AIEDAM, Vol. **10**, 4, September, 1996.

[Wagener97] **Wagener, H.** New developments in sheet metal forming: sheet materials, tools and machinery, Journal of Material Processing Technology 72, 342-357, 1997.

[Wood93] **Wood, T.J.** Teoria Sistêmica Avançada e terceira Onda da Qualidade.

Anexo 1

Relationlog

1 Introdução

Relationlog (ou RLOG) é uma nova base de dados dedutiva desenvolvida na Universidade de Regina, no Canadá.

O objetivo do sistema Relationlog é suportar o direcionamento, a armazenagem e a inferência de dados com estruturas complexas, especialmente dados suportados em relações tipo agregar e modelos de objetos complexos.

Houve várias versões do RLOG desde 1996. A primeira foi feita sobre plataforma ORACLE/ Ingres, utilizando programas tipo SQL/C. Essa versão possibilitava sistema de base de dados dedutivas persistentes que suporta dados com valores complexos. Entretanto, a performance não é satisfatória, com relações tipo agregar devem ser armazenados com relações simples e o programa não tem controle sobre otimização de buscas na base de dados.

A segunda foi elaborada sobre o gerenciador de armazenagem persistente EXODUS. Apesar do EXODUS pudesse ser uma boa escolha, ele não suporta nem trabalha sobre o sistema operacional Solaris, que é o único sistema operacional disponível para nós no desenvolvimento de sistemas. Ao lado disso, sua própria linguagem, E, não prevê um ambiente para correção o que impossibilitou o desenvolvimento do Relationlog.

Finalmente, o gerenciador de base de dados orientado a objetos quase relacional, PARODY, foi escolhido. Ele possui as seguintes vantagens:

1. objetos persistentes;
2. códigos C++ conciso e completamente implementado;
3. performance razoável para pequenas e médias base de dados.

que são necessários para o protótipo de pesquisa do Relationlog. Para fazer o PARODY mais aceitável para a implementação, os códigos originais foram modificados para inclusão definição de meta informações e gerenciamento de índices.

O sistema Relationlog foi completamente desenvolvido em SPARCstation da SUN, rodando Solaris. Para o ponto de vista de linguagem, o sistema Relationlog provê uma linguagem de busca declarativa baseada no Relationlog, e também uma linguagem de manipulação de base de dados declarativa baseada em DatalogU. Ele também provê a construção de conjuntos formais para representação e manipulação de informações completas e parciais de conjuntos.

Para a perspectiva da implementação, há muitas características novas no sistema Relationlog:

- Esquemas para relações extensivas e de intensivas;
- Tipos de domínio persistentes, esquemas relacionais, fatos e regras;
- Interface amigável e fácil de usar;
- Inferência direta e acesso para valores agregados em relações de objetos complexos ou tipo agregar como se fossem relações normais;
- Linguagem de definição de dados, de manipulação de dados e de busca tipo SQL;
- Gerenciamento de meta informações no arquivo de catálogo de sistema, que guarde o caminho para toda a base de dados existente no sistema;

- Combinação efetiva de várias estratégias baseadas na natureza das buscas e dados na base de dados sem intervenção do usuário tal que não haja necessidade para especificação de formas de busca como em ADITI, LDL e CORAL;
- Fatos e regras necessárias para a realização de buscas são dinamicamente selecionadas e o resultado temporariamente armazenado para as próximas buscas;
- Regras utilizadas para realização de buscas são dinamicamente rescritas baseadas nas buscas;
- Dados deduzidos podem ser materializados como também serem avaliadas dinamicamente todo tempo que eles forem procurados;
- Um mecanismo "least recently used"(LRU) é utilizado para remover dados extensivos e intensivos da memória quando espaço for necessário.

O sistema Relationlog pode ser utilizado em gerenciamento de negócios, realimentação de informação e indústrias. Ele também é uma ferramenta para cursos de base de dados e pesquisa acadêmica.

2. Linguagem de Definição de Dados

A linguagem de definição de dados do Relationlog possibilita a especificação de tipos de domínios, esquema de relações vistas e regras. É similar a linguagem de definição de dados SQL, que é utilizada no ORACLE, Sybase, Ingres dentre outros. Como um resultado, usuários familiarizados com Sistema de Gerenciamento de Base de Dados amplamente utilizados, podem utilizar esse sistema facilmente.

2.1. Tipos de Domínios:

Os tipos de domínio primitivos suportados são os seguintes:

a. string(n) para caracteres tipo string com comprimento fixo *n*;
b. string que é uma abreviação para string(16);

c. token(n) para strings alfanuméricas sem espaço, hífen e aspas com no máximo *n* caracteres;
d. token que é uma abreviação para token(16);
e. integer(1) para inteiros de 1 dígito;
f. integer(2) para inteiros de 2 dígitos;
g. integer(4) para inteiros de 4 dígitos;
h. integer(8) para inteiros de 8 dígitos;
i. integer que é uma abreviação de integer(4);
j. float(4) para um número real de 4 dígitos;
k. float(8) para um número real de 8 dígitos;
l. float que é uma abreviação de float(40.

Os seguintes são exemplo de valores aceitáveis e não aceitáveis para esses tipos de domínios primitivos:

Tipo	Aceitável	Não Aceitável
String	'Bob Sam', "Bob's Wife"	'Bob", "abc"def"q", 'Bob's Wife'
Token	BobSam, Bob_Sam	Bob Sam, bob-Sam, bob's_Wife, 2sons, _Bob
Integer	5, 8, 120, -40	1.5, 1e4, one, "two"
Float	1, 2.0, 3.14, 1.2e4	one, "two"

Como complemento aos tipos de domínios específicos, o sistema Relationlog também suporta tuplas, conjuntos, lista, e sacos, que podem ser definidos utilizando os tipos de domínios primitivos. Segue abaixo exemplos desses tipos.

- Tipo Tupla: [Last: string, First: string], [City: string, Country: string];
- Tipo Conjunto: { token}, {[Last: string, First: string]};
- Tipo Lista: | integer|, |[Last: string, First: string]|;
- Tipo Saco: *float*, *[Last: string, First: string]*.

Esses tipos não primitivos devem ser criados pelo usuário utilizando o comando *create domain* ou *create type*. Note que os tipos Saco e Lista são correntemente tratados do mesmo modo que Conjuntos no sistema Relationlog, com notações diferentes.

2.2. Regras e Views

Baseado em base de dados, informações dedutivas podem ser definidas utilizando regras em Relationlog. Uma regra é similar ao Datalog:

$$A{:-}\ L_1, ..., L_n.$$

Todo L_i ($1 \leq i \leq n$) no corpo da regra é também um literal positivo com variáveis, um literal negativo que consiste na chave de negação e um literal positivo, ou comparações de conjuntos ou aritméticas sobre variáveis e constantes. A cabeça A é um literal positivo. A regra pode ser utilizada para deduzir valores a atributos de objetos existentes, ou para descrever como construir objetos e definir seus valores. A regra deve ser isenta de erros, o que significa que toda variável que aparece na cabeça A também aparece no corpo em um literal não negado. Todas as variáveis no RLOG devem começar com _ e _ sozinho pode ser utilizado como variável anônima. Dessa maneira, palavras começando com letras maiúsculas podem ser tratadas como constantes tipo token.

A versão atual do Relationlog suporta expressões aritméticas, expressões de comparação de string, e expressões de comparação entre conjuntos podem ser utilizados no Relationlog. A versão atual apenas suporta as operações aritméticas +, -, *, /.

Para as expressões de comparação, Relationlog suporta as seguintes operações: <, <=, =, !=, >= e >.

Para a comparação de strings, relationlog suporta as seguintes operações: =, e !=.

Para a comparação de conjuntos, Relationlog somente suporta **in** (parte de).

Somente variáveis podem ser utilizadas como os operadores.

Por exemplo:

Seguindo são exemplos de aritmética, conjuntos e expressão de comparação:

Expressão aritmética: X + 10, _X * (_Y - _Z), _X/ _Y;

Comparação aritmética: _Y + _Z = _X, _X > _Y;

Associação aritmética: _X = _Y * Z (_X é associado e tem que estar na esquerda);

Comparações de string: _X = _Z, _X <> "ABC";

Associação e Comparação de Conjuntos: _X in _S.

Note, essas expressões devem estar no fim de uma regra ou busca. Expressão de conjuntos (in) deve estar antes de string ou expressões aritméticas. Pelo menos 4 (quatro) string e expressões podem ser colocadas numa regra e busca.

Views são definidos usando regras baseadas em relações base ou outras views. Pode existir dois tipos de views: materializada (estocável) ou não materializada. Views materializadas são persistentemente armazenadas na base de dados como uma relação básica e são mantidas correntes enquanto views não materializadas são utilizadas quando ocorre uma busca.

Views materializadas são criadas usando o comando create stored view, enquanto não materializadas são criadas com o comando create view. Relantionlog suporta views definidas recursivamente, diferente das linguagens relacionais tradicionais como SQL.

Relationlog necessita de todas as regras na base de dados para ser estratificada. A estratificação das regras é automaticamente checada toda hora que um novo view é criado. A estratifição das regras será discutida na seção 3.

Com regras e views, Relationlog suporta diretamente operadores de álgebra relacional extendidos. Considere o seguinte esquema relacional:

P1 (A : string; A1 : {[B : string; B1 : {[C : string; D : string]}]})
P2 (A : string; A1 : {[B : string; B1 : {[C : string; D : string]}]})
P3 (E : string; B : string; B1 : {[C : string; D : string]})
P4 (A : string; B : string; C : string; D : string)
P5 (A : string; A1 : {[B : string; B1 : {[C : string; D : string]}]})
P6 (A : string; A1 : {[B : string; B1 : {[C : string; D : string]}]})
P7 (E : string; B : string; B1 : {[C : string; D : string]})

Esse exemplo mostra as seguintes funções de operadores de álgebra relacional e relações de agregação: P 1 = nest(B; (C; D)(nest(C; D)(P4))), P4 = unnest(C; D)(unnest(B; (C; D))(P1)), P5 = P1 $\cup$ P2 , P6 = P1 $\cap$ P2 , P7 = P1 - P2 , P8 = P2 $\otimes$ P3 .

Em Relationlog essa álgebra pode ser representada diretamente utilizando regras tais como:

Agregar: P 1 (_A; <[_B; <[_C; _D]>]>) :- P 4 (_A; _B; _C; _D).
desagregar: P 4 (_A; _B; _C; _D) :- P 1 (_A; <[_B; <[_C; _D]>]>).
União: P 5 (_A; <[_B; < _E>]>) :- P 1 (_A; <[_B; < _E>]>)
P 5 (_A; <[_B; < _E>]>) :- P 2 (_A; <[_B; < _E>]>).
Intersecção: P 6 (_A; <[_B; < _E>]>) :- P 1 (_A; <[_B; < _E>]>),
P 2 (_A; <[_B; < _E>]>).
Diferença: P 7 (_A; <[_B; < _E>]>) :- P 1 (_A; <[_B; < _E>]>),
¬(P 2 (_A; <[_B; < _E>]>).
Junção: P 8 (_A; _C; _B; < _E>) :- P 2 (_A; <[_B; < _E>]>),
P 3 (_C; _B; < _E>).

Note que para as regras acima a variável _E é utilizada para representar uma tupla. Dessa forma, variáveis podem ser utilizadas para representar não somente valores atômicos, mas também conjuntos agregados e tuplas.

As views materializadas e não materializadas podem ser eliminadas pelo comando *drop view*. Entretanto, se uma view é utilizada por outra view, então a mesma não pode ser eliminada.

Se uma view a ser eliminada é não materializada, então as regras utilizadas para definir a view será deletada. Se a view a ser eliminada é materializada, então ambos regra e relações de derivação serão deletadas.

2.3. Linguagem de Busca

Relationlog suporta diretamente todas as buscas e regras Datalog. Ele também possibilita acesso direto e inferência de dados agregados profundamente pela criação de dois poderosos conjunto de termos: termos de conjunto parcial da forma { O_1, ... O_n}e um termo tupla da forma [O_1, ... O_n] ou [A_1: O_1, ... A_n: O_n], onde O_1, ... O_n, $2 \leq n$, são termos tipo variáveis constantes, conjuntos ou tuplas e A_1, ... A_n são nomes de atributos.

A busca do RLOG leva o usuário para buscar diretamente relações e views materializadas e não materializadas com comandos de procura. Uma busca é uma expressão da seguinte forma:

query L_1, ... L_n

onde L_1, ... L_n são uma seqüência de literais positivos, literais objetos negados, aritmética usual e literais de comparação da teoria de conjuntos. Um literal positivo em uma busca é organizado como:

relation_name (E_1, ... E_m),

enquanto E_1, ... E_m são uma combinação de variáveis, constantes, conjuntos e tuplas. Estes são muito similares a regras. Um literal negativo em uma busca consiste na chave não e um literal positivo. As comparações aritméticas e da teoria de conjuntos são as mesmas utilizadas nas regras. O Relationlog sempre irá dar todas as respostas de uma vez.

Anexo 2

Rol

ROL é um banco de dados orientado a objeto dedutível desenvolvido pela Universidade de Regina no Canadá. O sistema suporta objetos, objetos complexos, classe de objetos, herança de classes, herança múltipla suprimida e esquemas.

1. Estrutura

O programa do ROL consiste em duas partes: o *schema* e o *program*.

Schema contém as informações sobre as *classes*. *Classes* são utilizadas para representar uma coleção de objetos que possuem características comuns. Quatro *classes* de tipos são distintas:

•*Value Classes*. Ex.: *integer, real, string, integer(2..200), real(-30,00..30,00), string('Fácil','Dificil');*

•*Oid Classes*. Ex.: *fornecedor, parte, pessoa, estudante, departamento, curso;*.

•*Functor Object Classes*. Ex.: *fornece(fornecedor, parte), família({pessoa}), casado(person, person);*.

•*Set Classes*. Ex.: *{ pessoa}, { parte}, { quantidade(parte, inteiro)}*.

Uma *Functor Object Classes* corresponde a um esquema relacional numa base de dados relacional ou base de dados de valores complexos.

Classes podem ter atributos e *subclasses* de outras *classes*. A *classe* pode ser definida junto com seus atributos e *superclasses* imediatas no *squema* usando a seguinte notação:

$p\ [\ l_1 => q_1, \ldots, l_n => q_n\ ]\ isa\ p_1, \ldots, p_m$

onde p_i e q_i são *classes*, m, n ≥ 0. l_i é o nome do atributo i.

A *subclasse* herda todos os atributos definidos nas suas *superclasses*, a menos que sejam suprimidas.

Uma vez que uma *classe* é definida utilizando-se a forma acima, o *Set Classe* correspondente é automaticamente definido, e as *superclasses* imediatas correspondentes também são definidas. A menos que se deseje definir atributos no *Set Classe* não é utilizado a forma acima para o *Set Classes*.

Squema consiste no conjunto das definições das *classes*. É representado pela relação:

K= (B, C, A, Dc, **isa**, Da),onde:

B ⊆ Value Class;

C ⊆ Oid Class;

A ⊆ Set Class;

Dc ⊆ Class de conjunto definidos baseados em B e C tal que todo nome de classe objeto $f \in$ C está associado comum a classe em Dc;

isa é um conjunto de declarações de subclasses imediatas;

Da é um conjunto finito das declaração.

Baseada na relação imediata de *subclasses* e *atomic classes* , a conexão entre eles pode ser representada utilizando ***isa**. Para a relação entre instâncias e *classes* arbitrárias usa-se ***eisa**.

As *classes* de um program devem formar uma semi-reticulado. Isto é, para qualquer par de *classes* objeto c1 e c2, há uma única classe c3, chamada encontro de c1 e c2, tal que c3 ***isa** c1 e c3 ***isa** c: e para qualquer *classe* objeto c4, se c4 ***isa** c1 e c4 ***isa** c2, então c4 ***isa** c3.

Se a classe c herda definições de atributos *l* => *c1* e *l* => *c2* , então c tem as definições de atributos *l* => *c3*, onde c3 é o encontro de c1 e c2.

None é definido como uma *classe* de objeto base,. O encontro de *none* com qualquer outra *classe* é *none*.

Duas diferentes hierarquia de *classes* não pode ter atributos comuns, mas as *classes* internas a uma hierarquia de classes podem ter atributos comuns como *subclasses* podem ser herdados, suprimidos, ou refinados de atributos de suas *subclasses*.

O *program* contém informações sobre os objetos. Normalmente pode ser dividido em duas partes: *facts* e *rules*.

Como as classes, há quatro tipos de objetos no ROL:

- *Values.* Ex.: 5, 30, 210, 3.1416, -4.2, 'José';
- *Object Identifiers (oids).* Ex.: José, João, s1, p3;
- *Functor Objects.* Ex.: fornece(s1, p3), família({José, Maria});
- *Sets* que se dividem em:
- *Partial Sets.* Ex.: < José, Maria>;
- *Complete Sets.* Ex.: Família({ João, Pedro, Carla}).

Partial Sets denotam parte de um *Complete Sets* .

Não se admite *Partial Sets* aparecerem em *Functor Objects*, ou *Sets*.

Facts são pedaços da informação sobre um objeto: suas classes e/ou relacionamentos com outros objetos. Sua representação segue a seguinte forma:

$o : c\ [\ l_1 \rightarrow o_1, \ldots, l_n \rightarrow o_n\ ]$

$o\ [\ l_1 \rightarrow o_1, \ldots, l_n \rightarrow o_n\ ]$

onde $\underline{c}$ é um *atomic class*, $\underline{o_i}$ são objetos, $\underline{l_i}$ são nome de atributos, e $n \geq 0$. A primeira forma somente pode ser utilizada quando $\underline{o}$ é é um *object identifier* , enquanto a segunda forma, quando $\underline{o}$ for *object identifier, functor object* ou *set*. Para a segunda forma e sendo $\underline{o}$ um *set*, n deve ser maior que 0. Note que $\underline{o}$ não pode ser um *partial set*, mas $\underline{o_i}$ pode ser qualquer objeto.

Fatos de um programa devem ser *well-typed* com respeito ao *squema* como qualquer sistema de base de dados relacional. Quer dizer, para cada fato no programa, deve haver uma *class* correspondente na definição do *squema*. O atributo de um objeto num fato deve existir para sua *class*, definido diretamente, herdado ou suprimido e o atributo *values* deve ser consistente com as definições correspondentes.

Um variável deve iniciar com uma letra capital como em Prolog. Ele é apenas um registro.

Um *object expression* é similar ao fato exceto que se pode usar variáveis no lugar de objetos. Além disso, se f é um *functor object*, ao invés de dar os detalhes de sua estrutura, pode-se simplesmente utilizar a variável X com um *object expression* $X : f$.

Quando se tem variáveis em um *partial set*, então o *set* dever ser unitário (apenas uma variável no *set*).

Quando se tem variáveis em um *complete set*, então o *set* dever ser composto de variáveis, ou ser unitário contendo *functor object class*.

Regras são utilizadas para inferir informação sobre objetos: suas *classes* e/ou seus relacionamentos com outros objetos. São da seguinte forma:

A : - L_1, L_2, ... L_n

onde A é um expressão de objetos, e L_i , i > 0, é uma seqüência de objetos, aritmética, *set* ou expressões comparativas.

A estrutura geral de um programa no ROL é:

comments

Schema

class definitions

Program

facts

rules

Comments devem iniciar com % e podem aparecer em qualquer lugar do programa.

2. Buscas

No ROL pode-se usar três tipos de buscas:

1.Buscas em *Schema;*
2.Buscas em base de dados;
3.Buscas de alta ordem.

Normalmente, o resultado da busca é mostrado no padrão Prolog. Se esse padrão não é desejado, então pode-se adicionar o comando *write* após a expressão de busca para especificar o formato do resultado da busca, como mostrado abaixo

write (Filename, argument, ...)

A Busca em *Schema* é indicada para *schema* muito longos e tem a seguinte forma:

•C
onde C pode ser uma variável ou uma *class*;
•C_1 $[A => C_2]$
onde C_1, C_2 podem ser variáveis ou *classes* e A pode ser uma variável ou atributo;
•C_1 *isa* C_2 *ou* C_1 *!isa* C_2
onde C_1, C_2 podem ser variáveis ou *classes*, e ! representa negação;
•C_1 *isa** C_2 *ou* C_1 *!isa** C_2
onde C_1, C_2 podem ser variáveis ou *classes*, e ! representa negação;
•C_1 *eisa* C_2 *ou* C_1 *!eisa* C_2
onde C_1, C_2 podem ser variáveis ou *classes* ou *set classes*, e ! representa negação;
•C_1 *eisa** C_2 *ou* C_1 *!eisa** C_2
onde C_1, C_2 podem ser variáveis ou *classes* ou *set classes*, e ! representa negação;
•$C_1 = C_2$

onde C_1 e C_2 podem ser variáveis ou *classes* ou *set classes*.

A Busca em base de dados é uma seqüência de objetos, aritmética, *set* ou expressões comparativas.

Pode-se utilizar botton-up, top-down ou sua combinação para elaborar buscas.

Para se reduzir definições similares no programa, características de alta ordem podem ser usadas para escrever regras mais gerais e pesquisar buscas mais genéricas.

3. Arquitetura do ROL

O sistema ROL foi implementado como um sistema de base de dados para apenas um usuário. A arquitetura do sistema é mostrada na Figura 45.

O sistema é separado horizontalmente por 4 níveis. O primeiro é a interface com o usuário. Há dois tipos dessas interfaces, a primeira baseada em texto e, a segunda, em ambiente windows.

O segundo nível processa as buscas ou modificações. Esse nível se comunica com os gerenciadores de *facts, class* e *rules* respectivamente. O gerenciador de *facts* e *rules* consultarão o gerenciador de *class* com respeito as restrições do banco de dados, ou seja se o *fact* ou a *rule* é bem "tipada"; e a recíproca também é verdadeira.

O gerenciador de armazenamento gera um acesso rápido para *facts*, *squema* e *rules* guardadas em disco.

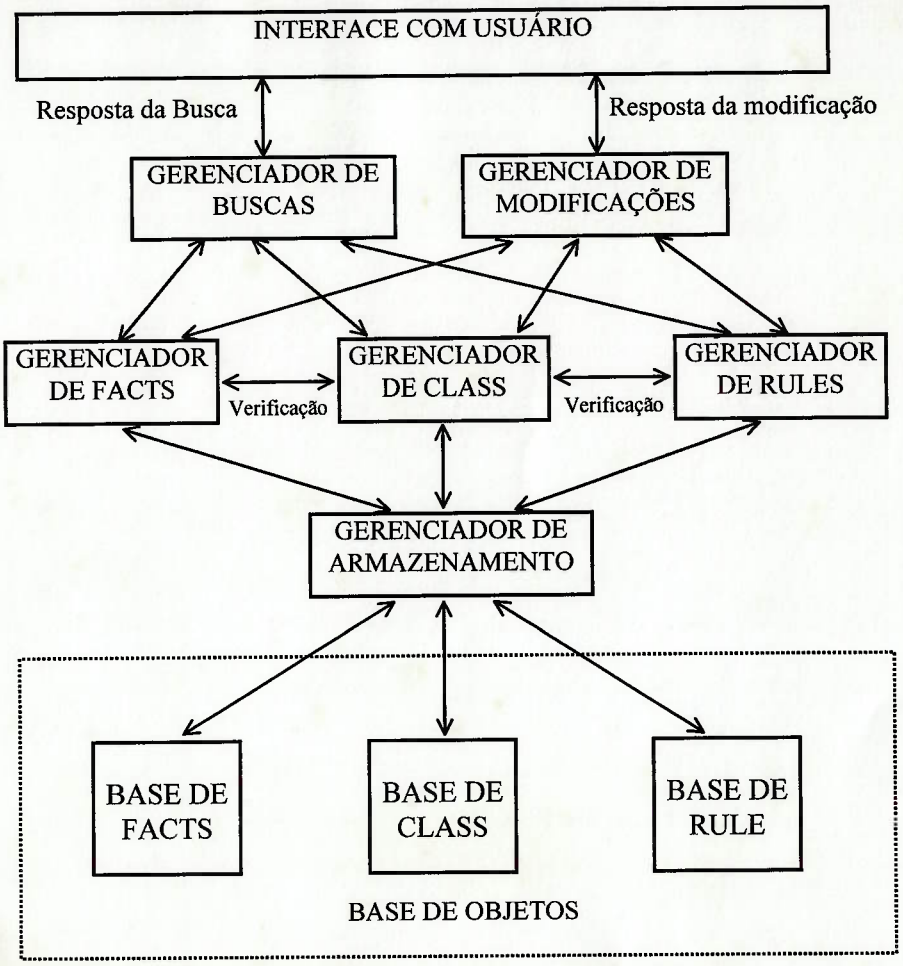


Figura 56 . Arquitetura do ROL