UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Field Bus : Análise Comparativa das Propostas e Implementação da Proposta SP50 em um Ambiente Simulado de Controle de Processos

Dissertação submetida à Universidade Federal de Santa Catarina
para obtenção do grau de
MESTRE EM ENGENHARIA ELÉTRICA

IBERÊ LOCKS LIMA

FLORIANÓPOLIS, AGOSTO DE 1996

Field Bus : Análise Comparativa das Propostas e Implementação da Proposta SP50 em um Ambiente Simulado de Controle de Processos

Iberê Locks Lima

Esta dissertação foi julgada adequada para a obtenção do título de

Mestre em Engenharia

especialidade **Engenharia Elétrica**

área de concentração **Controle, Automação e Informática Industrial**

e aprovada em sua forma final pelo Curso de Pós-Graduação

Prof. Jean-Marie Farines, Dr.Ing.
Orientador

Prof. Enio Valmor Kassick,Dr.
Coordenador do Curso de Pós-Graduação em Engenharia Elétrica

Banca Examinadora :

Prof. Jean-Marie Farines, Dr.Ing. (Presidente)
Orientador

Profª. Elizabeth Sueli Specialski, Msc

Prof. Vitório Bruno Mazzola, Dr.

Prof. Marcelo Ricardo Stemmer, Dr. Ing.

À minha mãe Edésia,
À minha irmã Luciana
À minha tia Teresa

AGRADECIMENTOS

Ao Prof. Jean-Marie Farines pelo apoio, orientação, compreensão e amizade, pessoa sem a qual esta dissertação não seria possível.

Aos membros da banca examinadora, pela aceitação de participação e pelas críticas e sugestões que foram de fundamental importância para a melhoria desta dissertação.

A todos os membros e colegas do LCMI pelo apoio e incentivo dado, em especial doutoranda Keiko Verônica Ono Fonseca pelo apoio, ajuda técnica e discussões que tanto me ajudaram na realização deste trabalho.

A UFSC e a CAPES pelo suporte material e financeiro.

A todo o pessoal da Telesc, meus amigos e companheiros de trabalho do FAGI e ADPS nos úlltimos dois anos, que muito me incentivaram para o término desta dissertção. Em especial a Ana Lúcia, Jane, Valesca, Maurício e Daniel.

A todos os meus amigos pelo incentivo e compreensão em todos estes anos.

A minha família por todo o carinho e apoio, em especial para minha mãe que tanto lutou e luta por mim.

A Profª. Elizabeth Sueli Specialski pelo incentivo e "puxões de orelha".

A todos que de uma forma ou de outra contribuíram para esta dissertação.

Enfim a DEUS, companheiro de todas as horas. Força que me fez superar todos os obstáculos.

"Standing on a hill with my mountains of dreams
telling myself it's not as hard as it seems"

Going To California - Page & Plant

Sumário

Glossário

AE - Application Entity - Entidade de Aplicação.

AE-I - Application Entity Invocation - Invocação de Entidade de Aplicação.

AI - Analogic Input -Entrada Analógica

ALI - Application Layer Interface - Interface da Camada de Aplicação.

AO - Application Object - Objeto de Aplicação.

AP - Application Process - Processo de Aplicação.

AP-I - Application Process Invocation - Invocação de Processo de Aplicação.

AR - Application Relationship - Relacionamento de Aplicação

ASE - Application Service Element - Elemento de Serviço de Aplicação.

CNRS - Centre National de la Recherche Scientifique

CSP - Communicating Sequential Process

CSRD - Cyclic Send And Request Data - Envio e Pedido de Dados Cíclico.

DCS - Distributed Control System - Sistema de Controle Distribuído

DIN - Deutsches Institut Für Normung

DL - Data Link - Enlace de Dados.

DLC - Data Link Connection - Conexão de Enlace de Dados.

DLCEP - Data Link Connection End Point- Ponto Final da Conexão de Enlace de Dados

DLE - Data Link Entity - Entidade de Enlace de Dados.

DLPDU - Data Link Protocol Data Unit - Unidade de Dados de Protocolo de Enlace de Dados.

DLS - Data Link Service - Serviço de Enlace de Dados.

DLSAP - Data Link Service Access Point - Ponto de Acesso de Serviço de Enlace de Dados.

DLSDU - Data Link Service Data Unit - Unidade de Dados de Serviço de Enlace de Dados.

EC - Estelle to C Compiler

EDB - Estelle simulator/DeBugger

EDT - Estelle Development Toolset

ESTIM - Estelle SimulaTor based on an Interpretative Machine

FAC - Field Bus Access Control - Controle de Acesso de Barramento de Campo

FCB - Frame Count Bit - Bit de Contagem de Quadro

FDC DLE - Fractional Duty Cycle DLE - DLE com Ciclo De Obrigação Fracionado

FDL - Field Bus Data Link - Enlace de Dados Field Bus

FF - Fieldbus Foundation

FI - Forma Intermediária

FIP - Flux Information Processus ou Factory Instrument Protocol

FLC - Field Bus Link Control - Controle de Enlace de Barramento de Campo

FM - Field Bus Management - Gerenciamento de Barramento de Campo

FMS - Field Bus Message Specification - Especificação de Mensagens de Barramento de Campo

FSC - Fieldbus Segment Coupler - Acoplador do Segmento de Barramento de Campo

IEC - International Eletrotechnical Commission

ISA - Intrument Society of America

ISO - International Organization for Standardization

ISP - Interoperable System Project

KB - Kommunikationsbeziehung - Relação de Comunicação

KBL -Kommunikationsbeziehungsliste - Lista de Relações de Comunicação

KR -Kommunikationreferenz - Referência de Comunicação

LAAS - Laboratoire D'Automatique Et D'Analyse Des Systemes

LAS - Link Active Scheduler - Escalonador Ativo de Enlace.

LCMI - Laboratório de Controle e Microinformática

LLI - Lower Layer Interface - Interface dea Camada Inferior.

LLI-PDU - LLI Protocol Data Unit - Unidade de Dados de Protocolo LLI

LM - Link Master - Mestre de Enlace.

MAC - Medium Access Control - Controle de Acesso ao Meio.

MAU - Medium Attachment Unit - Unidade de Ligção ao Meio.

ML - Meta-Language

MMS - Manufacturing Message Specification - Especificação de Serviços de Manufatura.

MTG - Multiple Type Group - Grupo de Tipo Múltiplo.

NRZ - Non Return to Zero - Não Retorna a Zero

OB - Objektbeschreibung - Descrição de Objetos

OD ou OV - Object Dictionary ou Objektverzeichnis - Dicionário de Objetos

OSI -Open Systems Interconnection - Interconexão de Sistemas Abertos.

PhIDU - Physical Interface Data Unit - Unidade de Dados da Interface Física.

PID - Proporcional Integral Derivativo

PIPN - Prolog Intepreted Petri Nets

SPDU - Support Protocol Data Unit - Unidade de Dados de Protocolo de Suporte.

SDA - Send Data with Ackowledge - Envio de Dados Com Reconhecimento.

SDL - Specification and Description Language

SDN - Send Data with No Ackowledge - Envio de Dados Sem Reconhecimento.

SRD - Send and Request Data - Envio e Pedido de Dados.

STG - Simple Type Group - Grupo de Tipo Simples.

TCAP - Time Critical Application Process - Processo de Aplicação de Tempo Crítico.

TDF - Técnicas de Descrição Formal

UFSC - Universidade Federal de Santa Catarina

UTE - Union Techniques Electrotechnique

VFD - Virtual Field Bus Device - Dispositivo Virtual de Barramento de Campo.

Resumo

Este trabalho tem como área de estudos a comunicação em ambientes industriais, abordando a padronização e o desenvolvimento de protocolos para os níveis mais baixos da hierarquia de comunicação deste ambiente, mais especificamente chamado de campo ou chão de fábrica.

Inicialmente é apresentado um estudo sobre as três principais normas de padronização de Field Bus, mostrando também um estudo comparativo destas. Dá-se um ênfase especial a norma SP50, que se mostra como futuro padrão mundial.

Uma metodologia para a especificação e o desenvolvimento de protocolos de comunicação baseada em refinamentos sucessivos também é apresentada. Esta metodologia enfatiza as fases de validação no ciclo de vida de desenvolvimento de protocolos de comunicação.

Concluindo este trabalho, apresenta-se o uso desta metodologia sobre a norma SP50, de modo a mostrar o uso da metodologia passo-a-passo juntamente com seus resultados. Assim, torna-se mais claro a importância do uso de uma metodologia, bem como exemplifica que o SP50 é um protocolo possível de ser implementado e com real aplicabilidade.

Abstract

This research paper in the area of industrial communication systems deals with the standardization and development of communication protocols to the lowest levels of the hierarchy of this area called Field or Shop Floor.

Firstly the three main Field Bus standards are described and then a comparative study of them is presented. A great emphasis is given to the SP50 which seems to be the future standard to be adopted worldwide.

A methodology for the specification and development of communication protocols based on the successive refinement is also shown. This methodology strongly emphasizes the validation phases in the life cycle of the development of communication protocols.

Finally the application of this methodology to the SP50 standard is presented so as to show its step-by-step application along with its results. In this way, it is possible to reaffirm the importance of the use of a methodology as well as to exemplify the feasibility and applicability of the SP50 as a communication protocol.

Capítulo 1

Introdução

A integração e Automação da Manufatura vem tendo um crescimento rápido nestes últimos tempos. Esta integração somente é possível a partir da comunicação, através da interligação inteligente de seus diversos equipamentos. Surge, em conseqüência, a necessidade de padrões que permitam esta comunicação, pois o uso de protocolos e interfaces padrões em um ambiente aberto oferece várias vantagens, tais como :

- habilitar as companhias a desenvolver estratégias de manufatura baseadas em soluções de informações integradas;
- garantir um ambiente multi-fornecedor, com uso dos equipamentos que mais se adequam as necessidades dos usuários;
- garantir o menor custo para equipamentos de comunicação;
- garantir o menor custo de manutenção de software;
- tornar a configuração mais flexível.

A partir do desenvolvimento do modelo de referência OSI (RM-OSI - Reference Model - Open Systems Interconnection) pela ISO (International Organization for Stardization) [ISO84], foram iniciados inúmeros esforços, a nível mundial, com intuito de padronização em todos os aspectos da comunicação de dados. No campo da automação da manufatura, um importante passo neste sentido foi a definição dos padrões MAP (Manufacturing Automation Protocol)/TOP (Technical Office Protocol)[MAP88] [TOP88].

O MAP é uma seleção de protocolos OSI, voltado para problemas de comunicação nos níveis inferiores da hierarquia de um ambiente industrial, enquanto que o TOP está voltado para comunicação nas áreas administrativas das empresas. O padrão MAP definiu também as arquiteturas MAP/EPA (Enhanced Performance Architecture) e Mini-MAP, com intuito de abranger os requisitos encontrados nos sistemas de tempo real. Porém, estas demonstraram algumas deficiências especialmente para uso no nível mais baixo de uma planta industrial. Dentre estas deficiências podemos citar :

- a utilização de um poderoso protocolo de aplicação, o MMS (Manufacturing Message Specification), que foi projetado para utilizar as sete camadas do modelo MAP, sobre uma arquitetura de apenas três camadas, sem o devido suporte.
- o protocolo de enlace é determinístico, mas trata todas as estações de forma igual, o que pode ocasionar um desperdício de largura de banda e tempo com rotações de ficha para estações que não tenham nada a transmitir.
- não leva em consideração a existência de serviços cíclicos, que são de extrema importância no contexto de aplicações de chão de fábrica.

Como solução alternativa, surgiu o Field Bus (barramento de campo), que é um protocolo de comunicação destinado a interligar os dispositivos de baixo nível como sensores, atuadores e controladores locais. Um exemplo de utilização do Field Bus em uma hierarquia de um ambiente industrial pode ser visto na Figura 1-1.

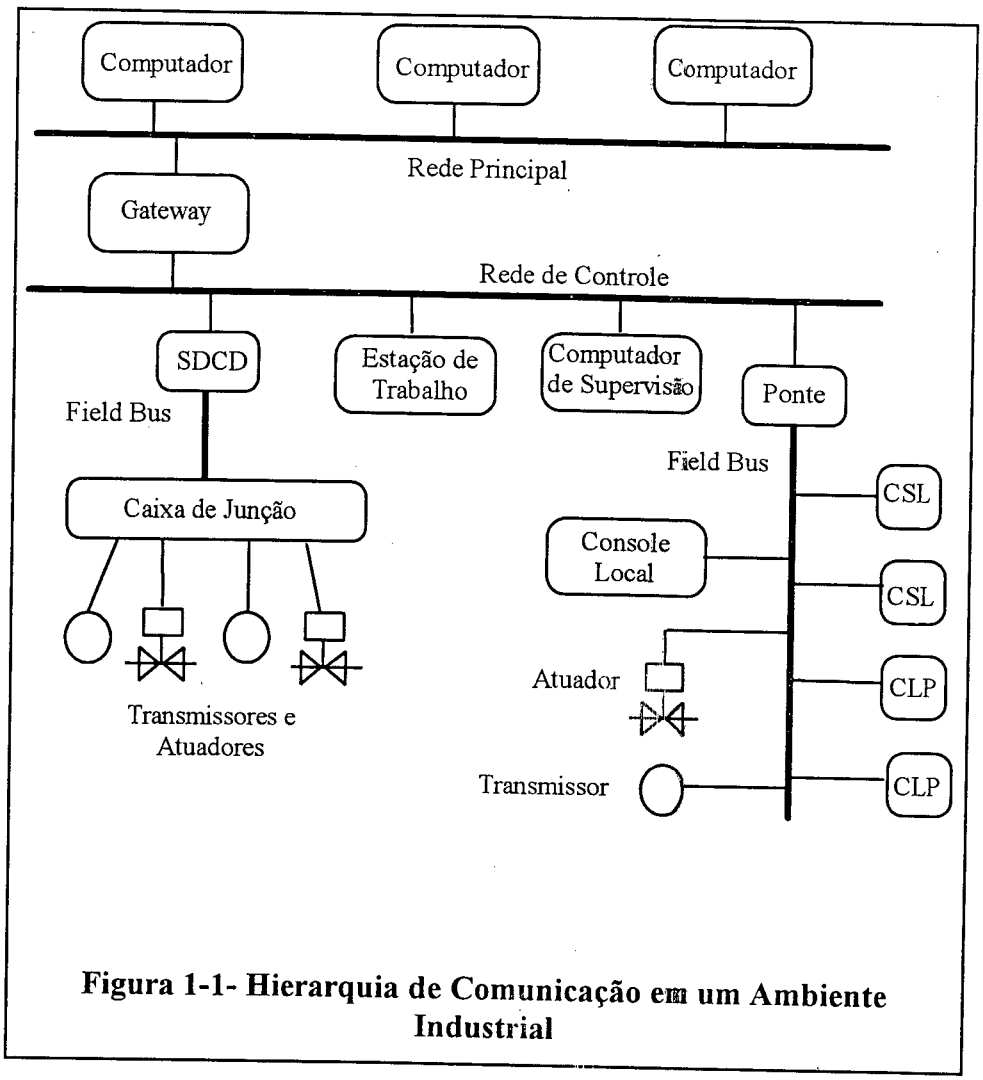


Figura 1-1- Hierarquia de Comunicação em um Ambiente Industrial

Existem várias propostas de padronização do Field Bus, sendo que dentre os padrões nacionais destacam-se o FIP (França) e o PROFIBUS (Alemanha). Já a nível mundial a definição do padrão está a cargo da IEC (International Eletrotechnical Commission) que tem como proposta o SP50[1]. Em 1992, surgiram dois grupos, compostos por várias companhias mundiais. O ISP (Interoperable System Project) baseado no PROFIBUS e o WorldFIP baseado no FIP. Em 1994, o WorldFIP e ISP se associaram para formar a Fieldbus Foundation (FF), num esforço para acelerar a padronização do fieldbus.

[1] - Esta norma foi inicialmente definida pela ISA (Instrument Society of America) que posteriormente passou a ter suas reuniões em conjunto com a IEC, tornando deste modo o SP50 o padrão mundial. Ao longo deste texto ela será referenciada como SP50 .

A presente dissertação visa inicialmente o estudo das propostas de padronização do Field Bus, bem como realizar uma análise comparativa destas. O estudo da norma SP50 (futuro padrão internacional) será apresentado com maior profundidade, uma vez que se constitui no enfoque principal do trabalho.

Outro objetivo deste trabalho é a escolha de uma metodologia para especificação e validação formal de protocolos de comunicação a partir de normas técnicas. Pretende-se aplicar esta metodologia sobre a norma SP50, com ênfase nas partes de simulação e verificação, com o intuito de retirar as ambigüidades inerentes de uma especificação informal e preparando esta para uma implementação.

A seguir, será realizada a simulação de uma planta industrial, cujo sistema de controle é composto de equipamentos interligados por um fieldbus SP50. O objetivo visado é duplo: demonstrar a possibilidade de realização da metodologia anteriormente apresentada e propor um ambiente de simulação da planta e do sistema de controle, que leva em conta os aspectos de comunicação.

No capítulo 2 serão apresentadas as três propostas de Field Bus, com mais detalhes para o futuro padrão internacional, o SP50.

No capítulo 3 estas três propostas são comparadas sob alguns critérios.

O capítulo 4 ressalta a importância do uso de uma metodologia para desenvolvimento de protocolos, e apresenta a metodologia adotada.

O capítulo 5 apresenta a especificação e validação de um fieldbus SP50 interligando a simulação de uma planta industrial, destacando os resultados obtidos com a aplicação da metodologia citada.

Por fim, o capítulo 6 faz algumas considerações finais sobre o trabalho, e apresenta perspectivas de novos trabalhos nesta área.

Capítulo 2

Os Padrões de Barramento de Campo

2.1 - Introdução

O desenvolvimento e o aumento do uso de dispositivos inteligentes foram o motor para a substituição da tradicional transmissão analógica 4-20 mA pela transmissão digital. A integração dos vários dispositivos inteligentes que aproveitam ao máximo esta tecnologia levou ao desenvolvimento dos padrões de comunicação do tipo Field Bus.

Um Field Bus (barramento de campo) é visto como um barramento de dados digital, serial, multiponto para comunicação em sistema de controle industrial de baixo nível (nível 0 ou chão de fábrica) com dispositivos de instrumentação como sensores, atuadores e controladores locais.

A substituição supracitada apresenta uma série de vantagens :

- o custo da cablagem é drasticamente reduzido, bem como o custo das interfaces de rede;
- o uso de um único cabo facilita a instalação, manutenção e expansão;
- a detecção das falhas nos cabos automaticamente durante o processo de varredura da interfaces de redes;
- a maior facilidade para a transmissão de informação para vários destinatários;
- a maximização da integração da informação;
- a maior tolerância a ruído, devido ao sinal digital;
- a maior precisão da medida, por causa da eliminação dos conversores digital/analógico e analógico/digital;

- a maior facilidade do emprego de dispositivos de campo inteligentes;
- a redução do custo da sala de controle devido a colocação de equipamentos de controle no campo.

De acordo com os requisitos que um Field Bus deve satisfazer definiu-se dois campos de aplicação :

- H1 : controle de processos tradicional de baixa velocidade usando a fiação existente, satisfazendo os requisitos de segurança intrínseca e fornecimento de energia através dos mesmos fios.
- H2 : controle lógico e de processos de alta velocidade, usando nova fiação e, se possível, fornecimento da energia através dos mesmos fios e satisfação dos requisitos de segurança intrínseca.

Existem várias propostas de padronização do Field Bus, sendo que dentre os padrões nacionais destacam-se o FIP (França) e o PROFIBUS (Alemanha). Já a nível mundial a definição do padrão está a cargo da IEC (International Eletrotechnical Commission) que tem como proposta o SP50[1]. Em 1992, surgiram dois grupos, compostos por várias companhias mundiais. O ISP (Interoperable System Project) baseado no PROFIBUS e o WorldFIP baseado no FIP. Em 1994, o WorldFIP e ISP se associaram para formar a Fieldbus Foundation (FF), num esforço para acelerar a padronização do fieldbus.

Neste capítulo serão apresentadas a três propostas citadas de padrão de Field Bus, com especial atenção para a norma SP50.

[1]- Esta norma foi inicialmente definida pela ISA (Instrument Society of America) que posteriormente passou a ter suas reuniões em conjunto com a IEC, tornando deste modo o SP50 o padrão mundial. Ao longo deste texto ela será referenciada como SP50

2.2 - FIP

2.2.1 - Introdução

O FIP (Flux Information Processus) é a proposta francesa de Field Bus (barramento de campo). Ela foi desenvolvida em um trabalho conjunto de empresas (usuários finais e fornecedores) e laboratórios.

O FIP é um sistema de comunicação, que segue o modelo OSI/ISO e possui 3 camadas , entre dispositivos de campo e de controle, substituindo a cablagem ponto-a-ponto tradicional.

A arquitetura geral do FIP pode ser vista na Figura 2-1 .

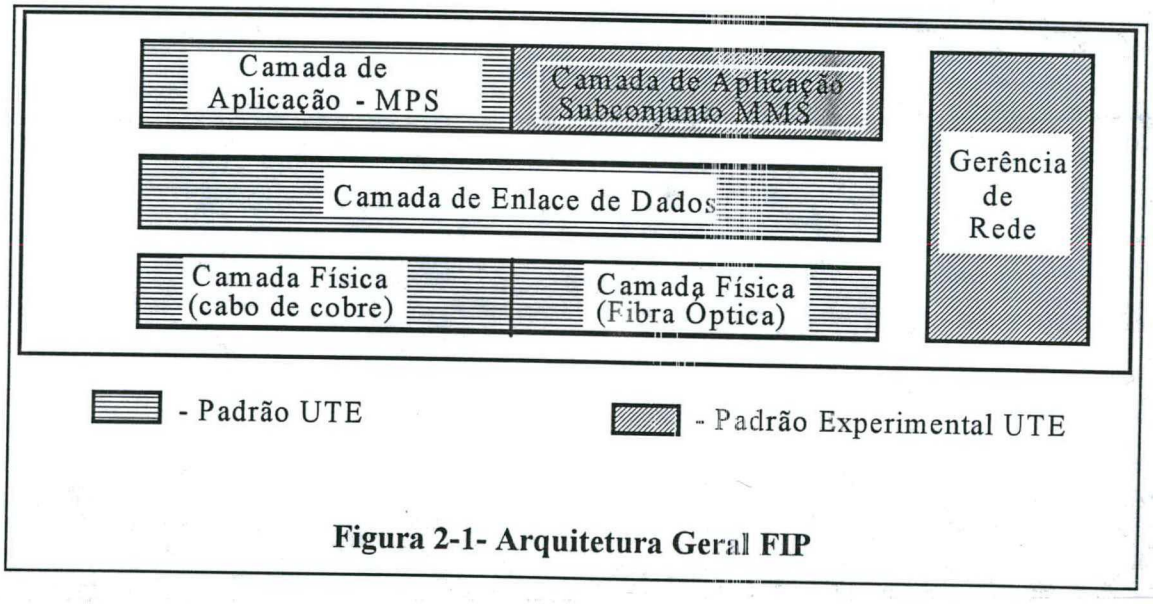


Figura 2-1- Arquitetura Geral FIP

A seguir veremos uma breve apresentação das camadas física e de enlace de dados.[2]

[2] - A camada de aplicação não foi incluída devido ao fato dos documentos relativos a esta que estavam a nossa disposição, estarem muito desatualizados

2.2.2 - Camada Física [FIP87]

A principal função da camada física do FIP é difundir para todos os assinantes a informação transmitida por qualquer usuário. Tendo como topologia básica um barramento no qual os assinantes estão conectados por meio de uma ligação (tap) passiva. O meio de transmissão é o par trançado blindado.

O FIP apresenta duas topologias, S1 e S2, que podem ser vistas nas figuras a seguir (Figura 2-2 e Figura 2-3).

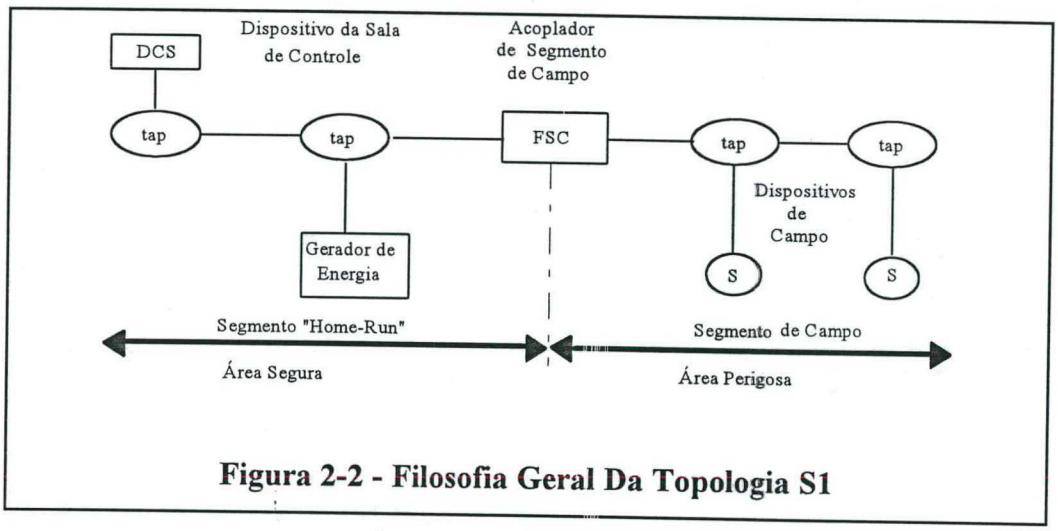


Figura 2-2 - Filosofia Geral Da Topologia S1

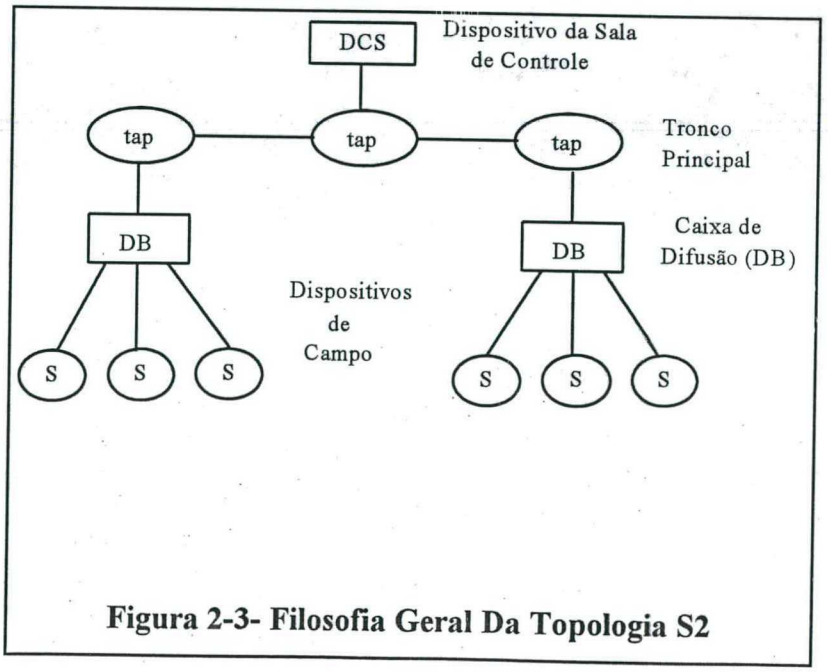


Figura 2-3- Filosofia Geral Da Topologia S2

Uma rede FIP deve seguir as seguintes regras:

- distância máxima entre dois assinantes é de 2 km;
- todos os assinantes devem trabalhar na mesma taxa de transmissão;
- o número de repetidores e caixas de difusão não deve ser maior que três;
- a operação da rede não deve ser interrompida quando um assinante é conectado ou desconectado da rede;
- a segurança intrínseca é possível nas topologias S1 e S2. No entanto, na S1 é limitada ao segmento de campo multiponto e na S2 ao segmento terminal ponto a ponto.
- codificação Manchester II (Bifase L).

No tocante a redundância de meio de transmissão, a emissão é feita simultaneamente em ambos os suportes de comunicação, mas deve haver uma escolha na recepção.

No que se refere a alimentação através do barramento, em ambas topologias S1 e S2, um dispositivo pode receber energia remotamente através do mesmo par de dados, respeitando algumas restrições.

2.2.3 - Camada de Enlace de Dados [FIP88]

A camada de enlace de dados do FIP apresenta algumas características relevantes:

- modelo de comunicação produtor/distribuidor/consumidor;
- a existência de uma estação especial, o árbitro de barramento que tem como principal função o controle de acesso ao barramento;
- não existe uma distinção formal entre a camada MAC e LLC.
- todos os endereços são globais.
- transferências de dois tipos :
  - * atualização de dados;
  - * mensagens.

As transferências de dados são sem conexão, sendo que os dados providos por um dispositivo são difundidos no barramento. As mensagens são trocadas ponto a ponto, com ou sem reconhecimento.

2.2.3.1 - Serviços

A camada de enlace de dados coloca à disposição da camada de aplicação os seguintes serviços :

- escrita de buffer;
- leitura de buffer;
- transferência de buffer;
- pedido explícito de transferência de buffer;
- pedido e transferência de mensagem sem reconhecimento;
- pedido e transferência de mensagem com reconhecimento.

O serviço de transferência de buffer é obrigatório enquanto que os demais são oferecidos de acordo com a classe de conformidade (10 classes).

O serviço de escrita de buffer permite colocar a informação da camada de aplicação na camada de enlace, isto é, permite a escrita do valor de uma variável identificada.

Já o de leitura permite a retirada de informação da camada de enlace pela camada de aplicação. É invocado pela camada de aplicação para ler o valor de uma variável.

A transferência de buffer permite a emissão ou recepção de uma variável através do meio. O pedido de transferência é feito durante a configuração do sistema para ser executado periodicamente.

O pedido explícito de transferência de buffer requisita ao árbitro de barramento, a colocação em circulação de um quadro de identificação, de modo a iniciar a transferência da variável associada. Ele será invocado pela camada de aplicação para provocar a circulação de um identificador qualquer. Este pedido será satisfeito durante a fase de verificação dos serviços aperiódicos do árbitro de barramento. A transferência de buffer que ocorre é similar ao serviço descrito anteriormente. Para cada pedido explícito, 2 níveis de prioridade estão a disposição: urgente e normal.

O serviço de pedido e transferência de mensagens sem reconhecimento permite iniciar a colocação em circulação, pelo árbitro de barramento, de um quadro identificador dedicado ao serviço de mensagens sem reconhecimento, ao nível de camada de enlace de dados, permitindo a troca efetiva de uma mensagem entre duas entidades.

De maneira similar existe o serviço de pedido e transferência de mensagens com reconhecimento. Este difere apenas no fato de que um reconhecimento é emitido pela entidade receptora da mensagem sem que seja necessário ao árbitro de barramento emitir um quadro identificador para o reconhecimento.

2.2.3.2 - Protocolo

No FIP o controle do acesso ao barramento é feito através de uma estação especial, o árbitro de barramento, caracterizando deste modo, um controle centralizado. Este método de acesso possui importantes características:

- difusão;
- eficiência do método de rastreamento (scanning) para transferência de dados cíclicas;
- compartilhamento do meio é parametrizado pelo usuário;
- tempo de acesso garantido para os dados cíclicos em qualquer circunstância.

Uma vez que o método de acesso não dá privilégio a mensagens e transferências de dados não periódicas, o tempo de acesso para estas não é garantido. Existem, porém espaços(slots) de tempo para ambas , dando um limite máximo para a realização destas.

Pode-se ver o funcionamento básico para transmissão de dados do FIP na Figura 2-4, onde cada identificador é ligado a um determinado dado do sistema FIP.

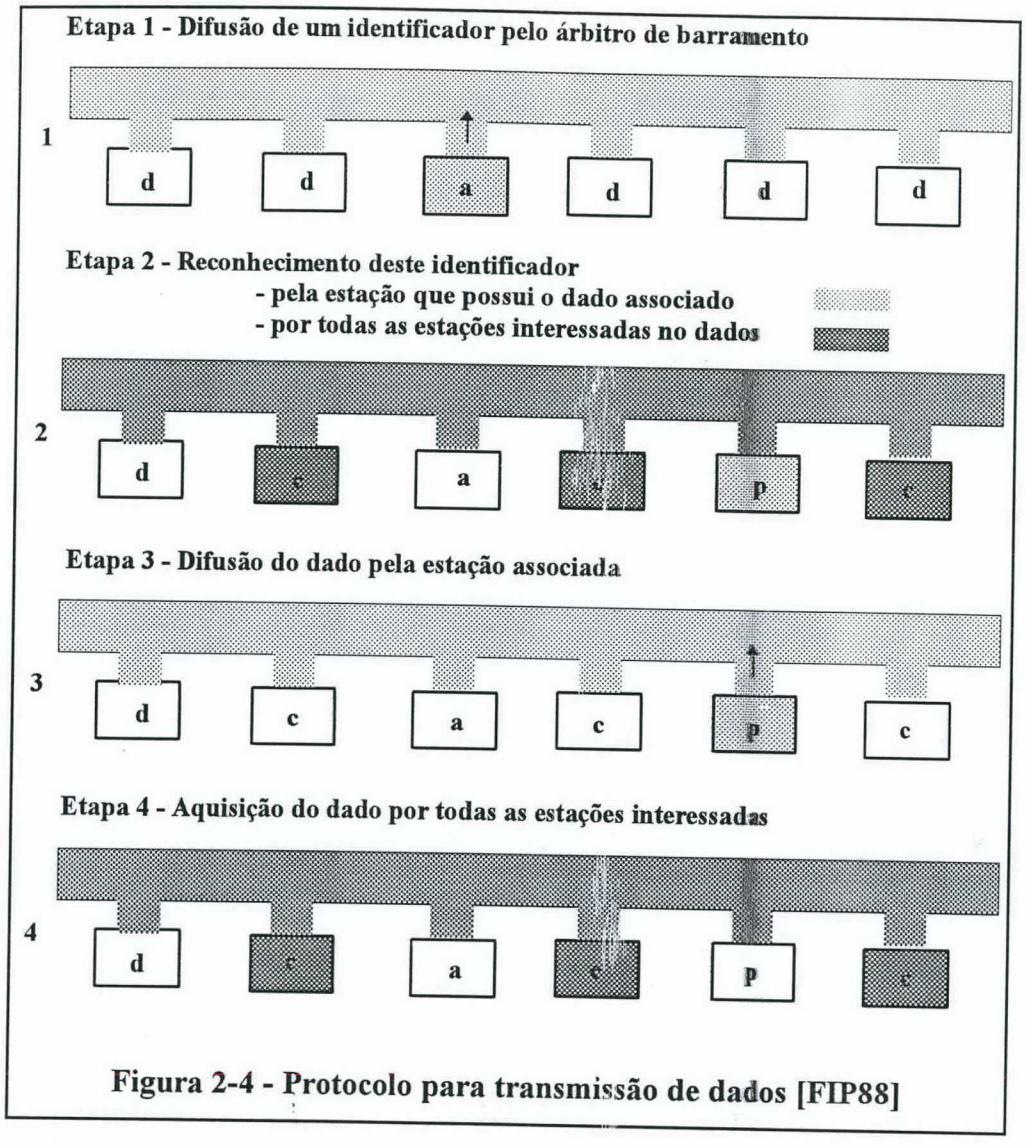


Figura 2-4 - Protocolo para transmissão de dados [FIP88]

Para atualizar um dado cíclico, o árbitro de barramento difunde um identificador de dado cíclico de acordo com a configuração do sistema. Então, o produtor difunde o quadro de resposta, o qual é recebido por todas as entidades consumidoras interessadas neste dado.

Para as mensagens e serviços acíclicos, a estação deve requisitar o serviço através do campo de controle de resposta, e este será atendido durante o tempo reservado para o tipo determinado de atividade.

O objetivo principal do FIP é prover o serviço de atualização cíclica de dados. Outros serviços são oferecidos usando transferências de mensagem e trocas de dados não periódicas, logo, as performances destes últimos serviços não são otimizadas.

Para a transferência de dados cíclicos e acíclicos o controle de fluxo é realizado em tempo de configuração, respeitando as restrições de períodos, espaço de memória alocado, número de transferências não periódicas autorizadas por ciclo do árbitro de barramento.

O controle de fluxo é parcialmente feito durante a transmissão, pois somente uma mensagem pode ser iniciada em um dado instante por um dado ponto de acesso de mensagem. O número máximo de mensagens por ciclo é definido durante a configuração.

Existem também mecanismos para detectar a perda ou duplicação de quadros. A detecção de perdas implica no uso de temporizadores.

No FIP existe um mecanismo de proteção que é o Protocolo de Introdução, que garante que uma entidade inserida no sistema FIP não participará das trocas de dados cíclicas ou acíclicas antes de checar a consistência de sua base de dados em relação a base de dados distribuída.

2.2.3.3 - Tipos de Estações

No que se refere a tipos estações, o FIP classifica-as em produtora e consumidora. Uma estação especial é designada como o árbitro de barramento.

Uma estação produtora/consumidora deve possuir :

- um mecanismo para analisar os tipos de quadros;
- uma tabela de identificadores dando o direito de transmissão;
- uma tabela de identificadores dando o direito de recepção;
- um mecanismo para ler ou escrever informação em espaços de memória alocados para variáveis e mensagens.

Já o árbitro de barramento tem o papel de conceder o direito de acesso a cada produtor da informação através das três funções seguintes :

- verificação cíclica de variáveis;
- verificação acíclica de variáveis;
- verificação de mensagens;

O árbitro garante igualmente funções de :

- encadeamento de seqüências elementares de quadros segundo os parâmetros de configuração;

- gestão das seqüências elementares de quadros e análise do campo de controle dos quadros de resposta.

2.2.3.4 - Macro-Ciclo e Ciclo Básico

A principal função do árbitro de barramento é dar o privilégio de acesso ao meio através da transmissão de identificadores. A configuração mínima que o árbitro de barramento deve receber da aplicação é uma lista com todos os identificadores que deve transmitir, de modo a gerenciar esta aplicação de usuário particular.

Ciclo Básico

O ciclo básico caracteriza-se pela verificação cíclica dos identificadores das variáveis da aplicação durante o período mais curto de atualização. O ciclo básico também pode cobrir as comunicações acíclicas. Isto porque ele pode ser composto de 5 fases , dentre as quais a primeira é obrigatória. São elas:

- varredura de variáveis cíclicas;
- reexecução da varredura de algumas variáveis cíclicas (retransmissão);
- varredura de mensagens com gerenciamento de retransmissões para as mensagens com reconhecimento;
- varredura das variáveis acíclicas com gerenciamento dos níveis de prioridade;
- espera do sinal de sincronização no caso de necessidade de um ciclo de varredura sincronizado.

Macro-Ciclo

O macro-ciclo é o período de varredura cíclica do conjunto de identificadores da aplicação. Ele cobre todas as necessidades de comunicação cíclica da aplicação do usuário, bem como todas as não periódicas que podem ocorrer durante a operação. O macro-ciclo é composto de pelo menos um ciclo básico, e pode ser com ou sem sincronização. A duração do macro-ciclo é normalmente baseada no maior tempo de atualização de uma variável do sistema. As figuras a seguir (Figura 2-5 e Figura 2-6) retratam estas definições.

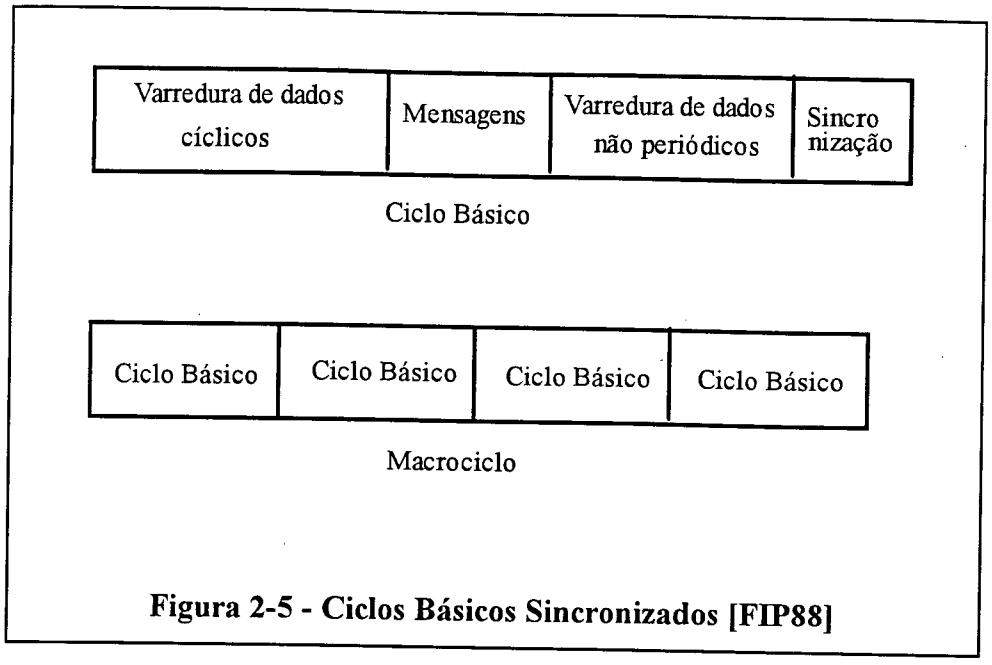


Figura 2-5 - Ciclos Básicos Sincronizados [FIP88]

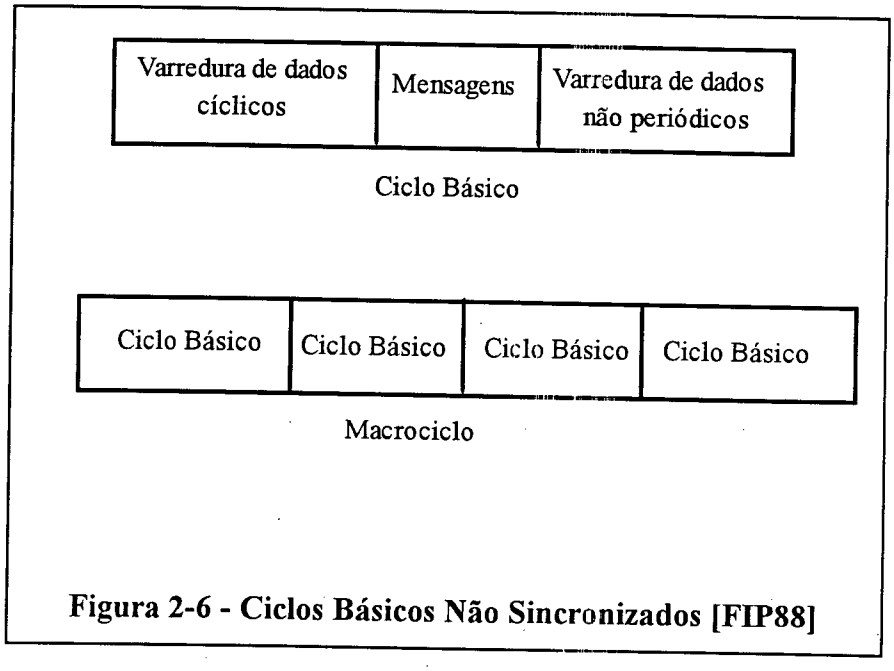


Figura 2-6 - Ciclos Básicos Não Sincronizados [FIP88]

2.3 -PROFIBUS

2.3.1 - Introdução

O PROFIBUS é a proposta alemã para padronização Field Bus. Este trabalho de padronização foi feito através de um trabalho conjunto de várias empresas e instituições de ensino e pesquisa alemãs.

Ela segue o modelo OSI/ISO, porém, possui uma arquitetura reduzida de 3 camadas:

- Física;
- Enlace de Dados;
- Aplicação.

No PROFIBUS cada estação deve ter um papel, Mestre ou Escravo.

Na Figura 2-7 podemos ver a arquitetura geral do PROFIBUS [PELL92]. Vale ressaltar que a Interface da Camada de Aplicação (ALI) não faz parte da camada de aplicação propriamente dita.

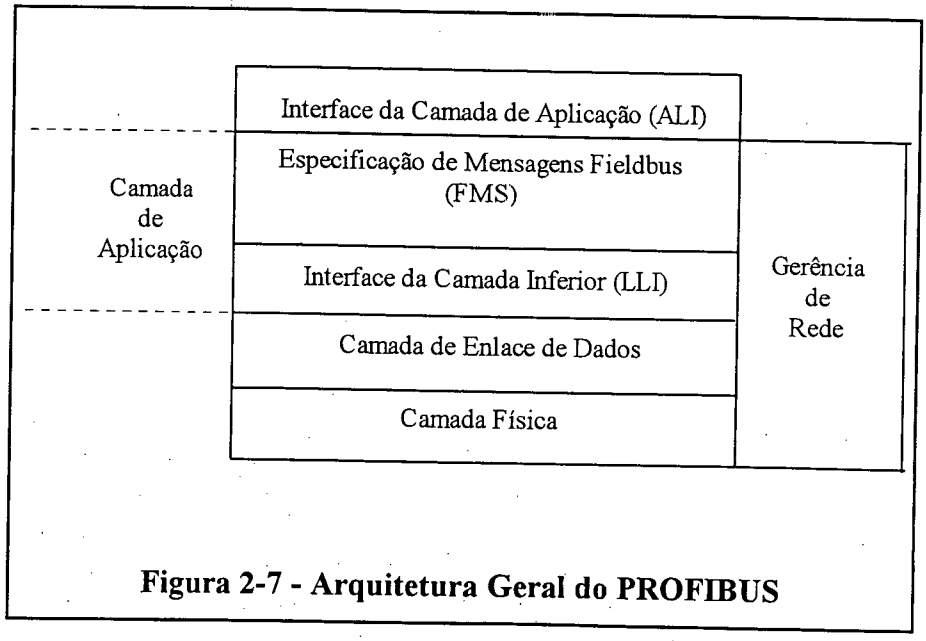


Figura 2-7 - Arquitetura Geral do PROFIBUS

A seguir, apresenta-se uma breve descrição de cada uma das camadas.

2.3.2 - Nível Físico [PROF91]

A nível físico o PROFIBUS segue a recomendação EIA/RS 485. Esta utiliza a transmissão assíncrona, com o meio de transmissão composto por um único par trançado blindado e codificação NRZ (Non Return to Zero).

No tocante a topologia, são possíveis o barramento ou árvore com estrela ativa.

No que se refere a taxa de transmissão, o PROFIBUS apresenta as seguintes taxas:

- 9.6, 19.2 e 93.75 Kbit/s se distância <= 1.2 Km;
- 187.5 Kbit/s se distância <= 0.6 Km;
- 500 Kbit/s se distância <= 0.2 Km.

O número máximo de repetidores, estações e comprimento máximo da rede na topologia de barramento linear são:

- sem repetidor : 1.2 km e 32 estações;
- 1 repetidor : 2.4 km e 62 estações;
- 2 repetidores : 3.6 km e 92 estações;
- 3 repetidores : 4.8 km e 122 estações.

Já na topologia árvore com estrela ativa pode-se usar até 5 repetidores e aumentar o número de estações para 127.

O PROFIBUS permite o uso de Camada Física e meio de transmissão redundantes, bem como o de estações redundantes (tanto para sistemas com múltiplos mestres como para sistemas com um único mestre).

2.3.3 - Camada de Enlace de Dados (Fieldbus Data Link - FDL) [ZEBE90] [PROF91]

A camada de enlace do PROFIBUS divide-se em Controle de Enlace de Barramento de Campo (Fieldbus Link Control - FLC) e Controle de Acesso ao Meio de

Barramento de Campo (Fieldbus Access Control - FAC), sendo que não há uma interface formal definida entre ambos.

São providos endereçamento ponto a ponto, grupo e por difusão com endereço máximo de 127 estações.

O controle de fluxo é provido pelo respondedor rejeitando o quadro recebido, indicando que não há recursos disponíveis.

Para os serviços confirmados, a detecção de perda de quadros é feita pelo iniciador, que retransmite o quadro quando não recebe uma resposta dentro de um espaço de tempo definido. Quadros duplicados são descartados pelo respondedor, utilizando, para isto, o mecanismo de Bit de Contagem de Quadro (Frame Count Bit - FCB).

O PROFIBUS utiliza formato de telegrama com distância Hamming 4.

2.3.3.1 - Controle de Acesso ao Barramento de Campo (Fieldbus Access Control - FAC)

PROFIBUS usa um método de acesso ao meio híbrido, com as transferências de dados sendo controladas por um princípio Mestre/Escravo(centralizado) e transferência da função de Mestre por passagem de ficha(descentralizado). Cada estação mestre pode acessar o barramento quando possui a ficha e as escravas só podem transmitir dados quando requisitadas. Todas as comunicações são iniciadas pelo mestre quando este recebe a ficha. O método de acesso do PROFIBUS pode ser visto na Figura 2-8.

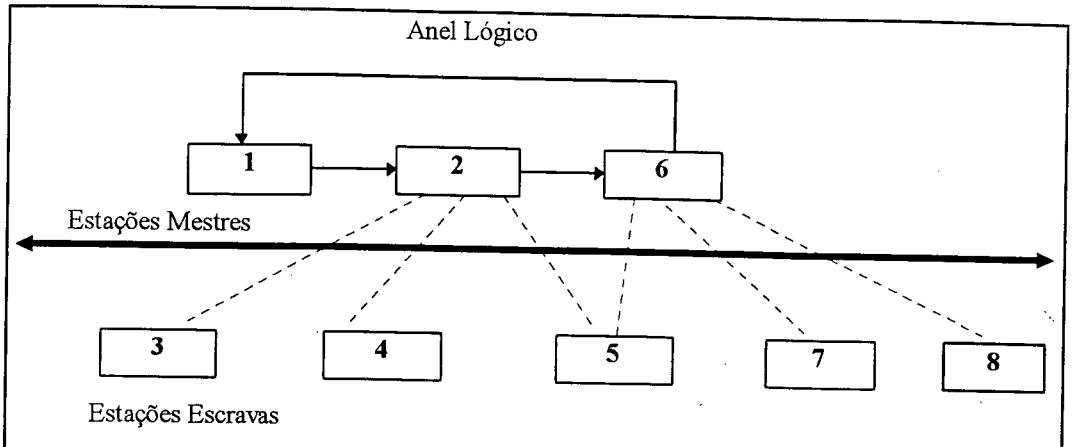


Figura 2-8 - Método de Acesso ao Barramento do PROFIBUS (Híbrido) [GÖDD90]

A ficha é passada de uma estação mestre para outra em ordem numérica ascendente de endereços, exceto para a estação com o endereço mais alto que passa para a estação com o endereço mais baixo para poder fechar o anel. A estação mestre que recebe a ficha torna-se o mestre corrente e o detentor da ficha.

Cada mestre mantém uma lista com todas as estações mestres ativas, que é constantemente verificada e dinamicamente modificada. Por outro lado, mantém também uma outra lista chamada de lista de GAP (GAPL) que compreende todas as possíveis estações entre ela e sua sucessora, onde é mantido o status destas estações. A lista de GAP é periodicamente verificada, fazendo-se as atualizações necessárias.

Se o anel lógico é composto por apenas uma estação mestre, o sistema trabalha como um mestre/escravo puro. Porém, a estação mestre continua a procurar por outros mestres que tenham sido ativados de modo a adicioná-los ao anel lógico automaticamente.

Transações Confirmadas são usadas para transferências de dados entre estações. Elas consistem de um quadro de ação (Dados e/ou Pedido) enviado pelo mestre corrente e o quadro de resposta correspondente (Reconhecimento ou Resposta) enviado pela estação endereçada (Mestre ou Escrava).

Transações Não Confirmadas são usadas para passar a ficha e para enviar mensagens sem reconhecimento (multicast e broadcast) e consistem de um quadro de ação (SDN).

Quando uma entidade recebe a ficha, primeiro ela executa os processos de alta prioridade pedidos pelo FLC local ou usuário FM (Field Bus Management). Após isto, se houver tempo de retenção de ficha disponível, a entidade processa os pedidos da Lista de Consulta(Poll List) em modo cíclico, e depois os processos acíclicos de baixa prioridade. Uma vez que uma transação começa a ser executada, incluindo qualquer tentativa de retransmissão, ela deve ser completada, mesmo que exceda o tempo de utilização da ficha que, neste caso, é artificialmente aumentado. Isto reduzirá automaticamente o tempo de retenção da ficha na próxima passagem.

Após completar suas transações ou após a expiração do tempo de retenção da ficha, o detentor da ficha passa a ficha para a estação sucessora. Se a estação sucessora não está ativa, ela tenta passar para a seguinte, e repete este processo até achar uma estação mestre ativa. Caso ela seja a única estação mestre ativa, então ela passa a ficha para si mesma e volta a usá-la. Entretanto, ela continua a monitorar as outras estações para poder detectar uma outra estação mestre que tenha sido ativada.

2.3.3.2 Controle de Enlace de Barramento de Campo (Fieldbus Link Control - FLC)

O controle de enlace possui os seguintes serviços confirmados :

- Send Data with Acknowledge (Envio de Dados Com Reconhecimento - SDA) : este serviço permite ao usuário FLC em uma estação mestre transmitir dados para um único usuário remoto através de um quadro de dados. O usuário local recebe uma confirmação indicando se os dados foram passados ou não para o usuário remoto.
- Send and Request Data (Envio e Pedido de Dados - SRD) : este serviço permite ao usuário FLC enviar dados a um único usuário FLC remoto bem como simultaneamente pedir dados que tenham sido previamente providos pelo mesmo usuário remoto, através da transmissão de um quadro de pedido de dados. Ele permite também ao usuário solicitar dados sem transmitir dados.
- Cyclic Send and Request Data (Envio e Pedido de Dados Cíclico - CSRD) : este serviço permite ao usuário FLC mandar dados

ciclicamente a vários usuários FLC remotos, bem como simultaneamente pedir dados que tenham sido providos previamente por estes mesmos usuários remotos através da transmissão de múltiplos quadros de dados. Os usuários remotos endereçados e o número e a seqüência de início de trocas nesta operação cíclica são determinados pelo usuário local através da lista de consulta (Poll-List) submetida junto com o pedido de ativação deste serviço. A operação cíclica continua, repetindo a Lista de Consulta, até que o usuário requisite a desativação deste serviço. Ele permite também ao usuário a solicitação cíclica de dados sem transmitir dados.

E um serviço não confirmado :

- Send Data with No Acknowledge (Envio de Dados Sem Reconhecimento - SDN) : este serviço permite ao usuário FLC transmitir dados para um usuário, grupo ou para todos usuários FLC em uma ou em todas as estações remotas simultaneamente através de uma única transmissão de um quadro de dados, sendo que o usuário local não recebe nenhuma confirmação da entrega dos dados.

Para serviços acíclicos, o cumprimento de uma transação leva ao término do serviço. Já nos cíclicos, várias transações ocorrem, logo múltiplas indicações serão recebidas e múltiplas respostas serão retornadas.

O usuário pode escolher duas prioridades para os serviços FLC : baixa ou alta.

2.3.4 - Camada de Aplicação [WILL92] [PELL92] [GÖDD90]

A camada de aplicação do PROFIBUS é composta de duas subcamadas:

- Field Bus Message Specification (FMS), que descreve os objetos de comunicação, serviços, bem como o modelo do dispositivo servidor (Virtual Fieldbus Device - VFD);
- Lower Layer Interface (LLI), que faz o mapeamento dos serviços da camada de aplicação para a camada de enlace de dados, além de prover as funcionalidades necessárias para suprir a não existência das camadas intermediárias do modelo OSI (3-6).

Existe também uma interface da camada de aplicação com os processos de aplicação, que é a Application Layer Interface (ALI).

Do ponto de vista de modelo de comunicação o PROFIBUS utiliza o modelo Cliente-Servidor, onde um cliente requisita um serviço que é realizado pelo servidor que pode ou não enviar uma resposta. O papel do cliente é controlado pela ALI, enquanto que o do servidor é descrito com a ajuda do modelo de Dispositivo Virtual de Barramento de Campo (VFD).

Uma vez que o modelo de comunicação do PROFIBUS trabalha com o paradigma de orientação a objetos, então toda a operação deste modelo é baseada em objetos. Visto que, num ambiente aberto um Processo de Aplicação (AP) é o elemento que executa o processamento de informações, no PROFIBUS tem-se ao nível de processo de aplicação os Objetos Processos (variáveis, programas, etc) , que contêm atributos, regras e descrições de operações sobre os objetos. Para que dois processos de aplicação remotos possam interagir, existe o mapeamento dos Objetos Processos nos Objetos de Comunicação, estes que são objetos virtuais, que são conhecidos pelos serviços PROFIBUS. Este serviço de mapeamento é realizado pela ALI, que também é responsável pelo envio orientado a objeto de pedidos de serviços pelo sistema de comunicação.

O VFD, citado anteriormente, representa a parte de um processo de aplicação que pode ser visível e gerenciada pelos serviços FMS. Este também tem como base um tipo de objeto, o Objeto VFD, que registra todos os Objetos de Comunicação representados, utilizando, para isto, as Descrições de Objetos (Objektbeschreibung - OB) que estão em um Dicionário de Objetos (Objektverzeichnis - OV), sendo que o Dicionário de Objetos pode consistir de até seis sub-dicionários de objetos, onde cada um destes contém descrições de uma certa classe de objetos de comunicação.

Vale ressaltar que um dispositivo pode conter mais que um VFD, podendo possivelmente agrupar Objetos Processos de operações distintas.

Os tipos de objetos conhecidos pelo PROFIBUS são :

- Variável (Simples, Arranjos, Registros);
- Domínio;
- Evento;
- Lista de Variável;
- Invocação de Programa.

No tocante a fase de definição, existem dois tipos de objetos no PROFIBUS :

- objetos estáticos, que são declarados durante a fase de projeto ou na de inicialização;
- objetos dinâmicos, que podem ser declarados durante a fase de comunicação.

Os Objetos Variáveis, Eventos, Domínios são apenas estáticos, enquanto que os Objetos Lista de Variáveis e Invocação de Programa podem ser declarados em qualquer fase. Estas restrições durante a fase de comunicação são feitas por questões de eficiência de transmissão.

A comunicação entre processos de aplicação ocorre através de canais lógicos, denominados de Relações de Comunicação (Kommunikationsbeziehung - KB). Um AP pode conter várias KB com o mesmo ou com diferentes APs. Todas as KBs são definidas durante a fase de projeto e registradas na Lista de Relações de Comunicação (Kommunikationsbeziehungsliste - KBL). Na KBL cada relação é identificada através de uma Referência de Comunicação (Kommunikationreferenz - KR).

A KBL é dividida em duas partes:

- FMS-KBL : contém informações sobre os serviços que podem ser usados dentro de cada KR, o número máximo de serviços pendentes, etc;
- LLI-KBL : contém os endereços do par remoto; LSAP local e remoto, conexão cíclica ou acíclica, conexão aberta ou definida e qualidade da conexão.

Os tipos de Relações de Comunicação podem ser vistos na Figura 2-9.

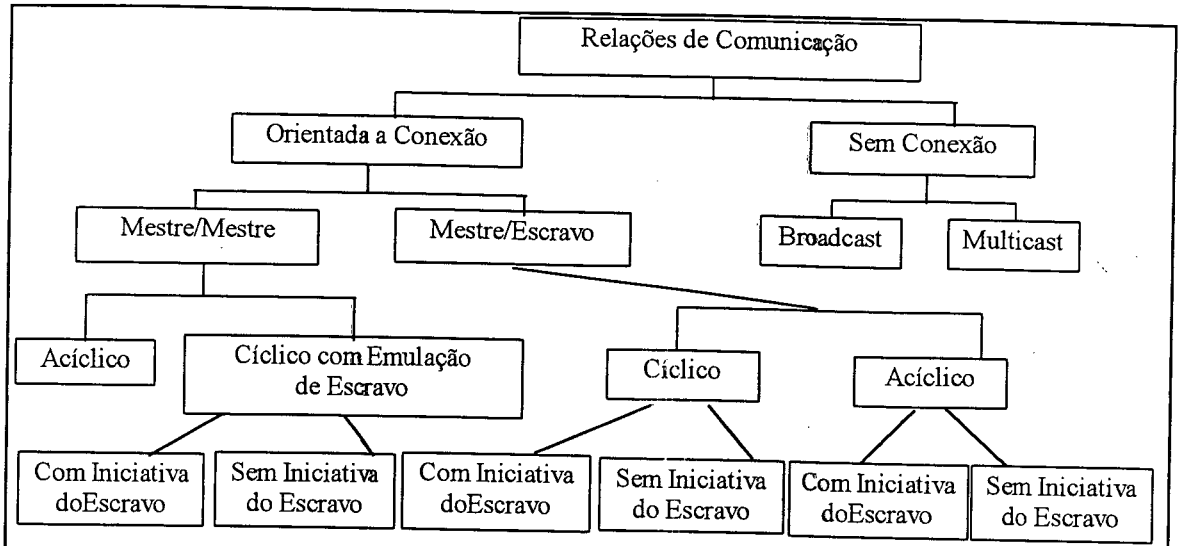


Figura 2-9 - Tipos de Relações de Comunicação do PROFIBUS [KATZ90]

As relações podem ser com ou sem conexão, sendo que as sem conexão podem ser "broadcast" (um-para-todos) ou "multicast" (um-para-alguns), sem confirmação. E na com conexão, esta deve ocorrer apenas entre dois processos (um-para-um), sendo possíveis os serviços confirmados e não confirmados. Na relação com conexão é necessário uma fase de inicialização, para troca de parâmetros. Ainda sobre esta última podemos subdividi-la em dois grupos :

- Conexão Definida, onde as partes envolvidas já estão previamente definidas, com endereço remoto e LSAP;
- Conexão Aberta , onde o dispositivo remoto não é conhecido.

O PROFIBUS fornece dois tipos de transmissão de dados:

- Cíclica, que permite que um pedido de leitura ou escrita de variável seja realizado a cada intervalo de tempo. Só pode ser realizado na relação Mestre/Escravo, ou na Mestre/Mestre, com um dos mestres emulando a função de escravo.
- Acíclica, onde não há a repetição temporal de serviços.

O usuário da ALI não pode distinguir se uma comunicação é cíclica ou acíclica, sendo que a LLI é a responsável pelo controle dos pedidos cíclicos.

No PROFIBUS um escravo só pode ter iniciativa para requisitar serviços não confirmados.

Quanto aos serviços FMS, eles são, em sua maioria, similares em suas funcionalidades aos serviços MMS, porém existem algumas extensões necessárias. As categorias de serviços de aplicação são :

- **Gerência de Contexto** : Inicialização, aborto e rejeição de conexão;
- **Gerência de Diretório de Objetos** : Leitura e escrita do Diretório de Objetos de um dispositivo de campo;
- **Suporte de VFD** : Oferece informação sobre os dispositivos;
- **Acesso a Variável :** Leitura e Escrita de variáveis de um dispositivo de campo, definido no Diretório de Objetos;
- **Invocação de Programa** : Permite a conexão de domínios a um programa e o controle do fluxo do programa;
- **Gerência de Eventos** : Oferece serviços para manuseio de alarmes controlado pelo usuário;
- **Gerência de Domínios** : Carga de partições de memória logicamente relacionadas (Domínios);

2.4 - SP50

2.4.1 - Introdução

A proposta SP-50 apresenta um modelo composto por 4 camadas :

- Física;
- Enlace de Dados
- Aplicação;
- Usuário.

A arquitetura geral do SP50 pode ser vista na Figura 2-10.

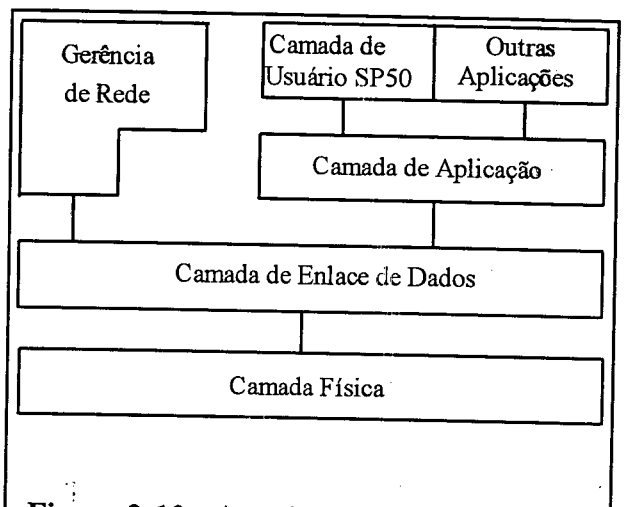


Figura 2-10 - Arquitetura Geral do SP50

Será mostrada uma descrição das camadas física, enlace de dados e aplicação. Vale ressaltar que a camada de aplicação está em fase de definição pela IEC, bem como a de usuário, sendo que para esta última apresenta-se apenas uma breve descrição.

2.4.2 - Camada Física [ISA92a]

A camada física especifica os requisitos Field Bus para as partes componentes e também os requisitos de configuração do meio e da rede, para garantir um nível aceitável de:

- integridade dos dados anterior a checagem de erros do enlace de dados;

- interoperabilidade entre dispositivos na camada física.

A camada física do Field Bus é igual a do modelo OSI da ISO, exceto que os delimitadores de quadro estão presentes na camada física.

A proposta SP50 define três tipos de meios de transmissão (fios, fibras ópticas e sinais de rádio). Todas as propostas apresentam características comuns :

- Transmissão de dados digital;
- Auto-sincronização (self-clocking);
- Comunicação half-duplex;
- Codificação Manchester.

A camada física também provê opções de combinações entre fornecimento ou não de energia através dos condutores do barramento e segurança intrínseca ou não.

Existem 2 tipos de acoplamento, paralelo e serial, e 3 velocidades de sinalização, 31.25 Kbit/s (H1), 1.0 Mbit/s e 2.5 Mbit/s (H2), sendo que o acoplamento paralelo combina-se com as 3 velocidades e a serial apenas com a de 1.0 Mbit/s.

Atualmente utiliza-se como meio de transmissão o par trançado, suportando as topologias em barramento e árvore, sendo que esta última é para apenas um caso. O número máximo de dispositivos que podem estar ligados a rede é 32, com um alcance de 1900m (31.25Kbit/s), 750m (1.0 Mbit/s) e 500m (2.5 Mbit/s).

2.4.3 - Camada de Enlace de Dados

2.4.3.1 Definição de Serviços [IEC93a]

A camada de Enlace de Dados do SP50 provê serviços de modo que possa ser usada pela camada de aplicação do Field Bus, bem como pela camada de rede do modelo OSI. Os serviços da camada de Enlace de Dados foram projetados para suportar comunicação com restrições temporais em ambiente de automação. Estes serviços são um superconjunto dos providos pelos protocolos OSI para camada de Enlace de Dados especificados no ISO 8886.

Os serviços da camada de Enlace de Dados são divididos em 4 tipos :

- Serviços de gerência de buffer, filas e endereços DLSAP;

- Serviços de transferência de dados com conexão;
- Serviços de transferência de dados sem conexão;
- Serviços de escalonamento de transação.

A - Serviços de gerência de buffer, filas e endereços DLSAP

Estes serviços permitem ao usuário DLS (Data Link Services):

- meios de associar e desassociar DLSAPs (Data Link Service Access Point);
- meios de criar, destruir, associar filas ou buffers para permitir a comunicação do usuário DLS para o provedor DLS e vice-versa;
- meios pelos quais DLSDUs (Data Link Service Data Unit) de tamanho limitado sejam escritas ou lidas de um buffer ou uma fila.

B - Serviços de Transferência de Dados Com Conexão

Estes serviços possibilitam o usuário DLS :

- estabelecer uma conexão bidirecional entre pares ou unidirecional multi-par;
- definir a qualidade do serviço desejado;
- transferir DLSDUs, que podem ser segmentadas;
- controle de fluxo;
- ressincronização de DLC (Data Link Connection);
- consulta sobre a existência de assinantes (multi-par);
- finalização incondicional de conexão;

C - Serviços de Transferência de Dados Sem Conexão

Estes serviços permitem :

- A transferência de dados de tamanho limitado de um DLSAP a outro em um único serviço, sem estabelecimento de conexão;
- Definição de qualidade do serviço pelo usuário emissor;
- Consulta sobre a existência de usuário escutando a um determinado endereço DLSAP;

D - Serviços de Guia de Escalonamento de Transações

As funções de guia de escalonamento permitem ao usuário influenciar na temporização ou na maneira como os serviços da camada de enlace de dados são providos.

As funções são as seguintes :

- pedido do tempo;
- liberação de um pedido de serviço previamente adiado;
- escalonamento de seqüências para executar uma vez ou ciclicamente;
- cancelamento deste escalonamento.

2.4.3.2 Definição do Protocolo [IEC93b]

A seguir, são apresentados definições e conceitos sobre estações, fichas, sub-níveis e classes que compõem o protocolo da camada de enlace do SP50.

A - Estações

A camada de enlace do SP50 apresenta três tipos de estações :

- **LAS - Link Active Scheduler - Escalonador Ativo de Enlace** : é uma entidade de enlace de propósito especial, a qual escalona o enlace local e serve como fonte para o tempo do enlace de dados (source of DL-time for the link). As funções de LAS estão presentes em todas as LM DLEs, porém apenas uma tem suas funções de LAS ativadas .
- **LM - Link Master - Mestre de Enlace** : é uma DLE que pode também prover as funções de LAS para o enlace, inicializando e escalonando o enlace.
- **DLE Comum - Data Link Entity - Entidade de Enlace de Dados** : é uma entidade simples que não possui as funções de LAS.

B - Fichas

O Token ou Ficha é a única permissão para ser o iniciador de transações no enlace local. Esta permissão é assumida pela LAS DLE quando o enlace é criado. Esta permissão pode ser delegada a DLEs individuais, sujeito a restrições no seu uso.

As fichas são implicitamente qualificadas pelo método de transmissão ou posse :

- **Ficha Escalonadora (Scheduler Token)** : é assumida e detida pela LAS DLE, e pode ser enviada para outra LM DLE no enlace local, para transferir a ativação das funções de LAS para a DLE receptora.
- **Ficha Delegada (Delegated Token)** : é enviada pela LAS para outra DLE no enlace e é retornada após seu uso, ou assumida na sua expiração.
- **Ficha de Resposta (Reply Token)** : é enviada pelo detentor atual da ficha delegada(ou ficha escalonadora, se não há detentor da ficha delegada) para uma DLE no enlace local, requerendo uma resposta imediata. Ela é retornada com a resposta imediata, ou assumida na expiração do período de resposta.

C - Sub-Níveis

A camada de enlace de dados é modelada em 3 sub-níveis :

- Baixo Nível de funções de escalonamento e acesso aos caminhos;
- Nível Intermediário de funções operação como ponte;
- Nível Superior de transferência de dados com e sem conexão, coordenação de pontes e funções de serviços de enlace de dados.

C.1 - Sub-Nível de Acesso ao Caminho e Escalonamento

O sub-nível de Acesso ao Caminho e Escalonamento tem a função de montar a DLPDU (Data Link Protocol Data Unit), a partir da informação de controle de protocolo e dos dados do usuário e da seqüência de verificação do quadro, sendo que esta última é calculada por esta subcamada. Então a subcamada passa a DLPDU como uma seqüência de PhIDUs (Physical Interface Data Units) para a entidade física. No lado da recepção ela retira os controles necessários, verifica a integridade da DLPDU e passa para a camada imediatamente superior.

Este sub-nível provê as funções básicas do iniciador e do respondente. Como respondente, ele prevê as funções necessárias :

- para receber uma DLPDU, possivelmente contendo uma ficha de resposta;

- no caso da recepção da ficha de resposta, para enviar uma DLPDU como resposta imediata a DLPDU que acabou de ser recebida.

Como iniciador, ele provê as funções necessárias para :

- receber a ficha delegada;
- enviar uma ou mais DLPDUs, incluindo aquelas que requisitam resposta imediata;
- receber esta resposta imediata, ou inferir sua ausência;
- devolver a ficha delegada.

Este sub-nível provê as funcionalidades de escalonamento de baixo nível, necessárias para escalonar a transmissão de DLPDUs em um caminho específico, incluindo as interações com o escalonador ativo de enlace (LAS), para coordenar o escalonamento com outras DLEs ou para requisitar a capacidade de transmissão necessária.

C.2 - Sub-nível de Pontes

O sub-nível de pontes provê as funcionalidades para :

- interconectar logicamente os enlaces locais formando o enlace estendido;
- prover a função de armazenamento-e-envio para permitir a comunicação entre DLEs no enlace estendido que não podem se comunicar de outra forma;
- prover um senso comum de tempo no enlace estendido;
- coordenar o escalonamento do enlace estendido.

C.3 - Sub-Nível de Transferência de Dados Com e Sem Conexão, Coordenação de Pontes e Serviços De Enlace de Dados

Este nível provê a funcionalidade de :

- gerenciar todas as interações da DLE com o usuário DLS, convertendo todos as primitivas de pedido e resposta na seqüência necessária de operações DLE, e gerando as primitivas de indicação e confirmação para o usuário quando necessário.
- gerenciar o sequenciamento de cada DLCEP (Data Link Connection End Point) ativo, incluindo :

* 1) determinação do tipo e seqüência das DLPDUs a serem transmitidas deste DLCEP;
* 2) negociação da qualidade de serviço;
* 3) segmentação e remontagem de grandes DLSDUs.

- processamento de todas DLPDUs de distribuição de informação de estado entre DLEs, tais como TIME DISTRIBUTION DLPDU e DLPDUs de configuração de pontes.
- gerenciar todas as interações DLPDUs de consulta e DLPDUs de resposta com outras DLEs, isto é aquelas que não ocorrem como resposta imediata.

D - Classes Funcionais

Existem 3 classes funcionais de DLE e estas determinam as capacidades para a atividade DLL (Data Link Level) autônoma, e a complexidade mínima para implementação, sendo que as classes superiores incluem as inferiores. As 3 classes em ordem crescente de complexidade são :

- Básica;
- Mestre de Enlace (LM) ;
- Ponte.

Todas as classes suportam todos os serviços de usuário DLS e são completamente interoperacionais.

D.1 - Classe Básica

Esta classe inclui os elementos de protocolo necessários para :

- prover interoperabilidade quando respondente a DLPDUs;
- iniciar, ressincronizar, terminar DLCs com um par;
- enviar e receber DLSDUs;
- pedir serviços de LAS;
- executar uma seqüência de operações de enlace continuamente;
- otimizar o uso do enlace.

D.2 - Classe Mestre de Enlace

Esta classe tem o necessário para :

- cooperar com DLEs similares para estabelecer e compartilhar a função de mestre do enlace;
- detectar a ausência dos LAS no enlace e acionar as funções de LAS do seu nodo;

e quando na função de LAS DLE

- manter um acesso ordenado ao recurso compartilhado de comunicação de enlace;
- servir de fonte do tempo interno a outras DLEs no enlace.

Esta classe é necessária para operação autônoma no enlace. Pelo menos uma DLE deve ser desta classe, porém apenas uma pode ter suas funções de LAS ativas.

D.3 - Classe Ponte

Esta classe adiciona os elementos necessários para :

- habilitar comunicações entre DLEs em um único enlace que são periodicamente incapazes de se comunicarem no enlace (ciclo de serviço fracionado - fractional duty cycle[1] (FDC - DLEs);
- interconectar dois ou mais enlaces locais para formar um enlace estendido;
- prover um senso comum de tempo interno Field Bus ajustado através do enlace estendido.

Esta classe é necessária quando se interconecta dois ou mais enlaces locais ou quando um ou mais DLEs no enlace local são do tipo FDC DLE.

[1] - Fractional Duty Cycle - é uma DLE que nem sempre recebe o sinal, normalmente está em um sistema afetado por grande carga elétrica. Ela necessita a assistência de uma ponte para se comunicar com outras DLEs no mesmo enlace.

2.4.4 - Camada de Aplicação [ISA92b]

Nesta seção serão apresentados conceitos gerais da camada de aplicação do SP50, o seu paradigma de orientação a objetos, este que norteia toda a sua definição. Serão apresentados também os modelos de objetos que já estavam definidos para esta camada nos documentos que estavam a nossa disposição.

2.4.4.1 Modelo de Objeto

A camada de aplicação do SP50 usa o paradigma de orientação a objetos, sendo que, com isto define um objeto como sendo uma coleção de seus dados, ações, comportamento e de seus relacionamentos com outros objetos.

Ela também usa a Estrutura de Informação de Enciclopédia, que define a estrutura da enciclopédia e a estrutura das definições dos objetos que estão contidos na enciclopédia. Ela é composta de 4 níveis de definições :

- **Descrição do Esquema** : composta de um conjunto primitivo de blocos de construção (building blocks);
- **Esquema de Enciclopédia** : composto de um conjunto de tipos entidade e tipos relacionamento, que permitem aos usuários definirem suas classes de objetos e instâncias de objetos;
- **Dicionário e Diretório** : contém definições de classes de objetos e instâncias de objetos. O conjunto de descrições de classes de objetos é chamado de Dicionário e o conjunto de descrições de instâncias de objetos é chamado de Diretório;
- **Objetos do "Mundo Real"** : contém as implementações dos objetos descritos na camada 3. Esta camada não é parte do dicionário e diretório físicos, mas é descrita por eles.

A - Descrição do Esquema

O primeiro nível usa 3 tipos de estruturas :

- tipos objeto do esquema, que identificam os objetos no nível 2;
- tipos atributo do esquema, usados para descrever os objetos do nível 2;
- tipos relacionamento do esquema, usados para associar os objetos do nível 2 com seus atributos.

B - Esquema de Enciclopédia

O segundo nível consiste dos seguintes objetos:

- tipos entidade, usados para definir entidades do nível 3;
- tipos relacionamentos, usados para definir relacionamentos entre entidades. São expressos na forma entidade-verbo-entidade;
- verbos de relacionamento, usados para prover um verbo comum para os tipos de relacionamento, citados acima;
- tipos atributos, usados para descrever a informação sobre as entidades e relacionamentos. Eles são os adjetivos das descrições.

Os relacionamentos entre os objetos do nível 2 são definidos usando os relacionamentos de esquema do nível 1. São eles:

- Tipo Entidade contém tipo atributo;
- Tipo Relacionamento conecta tipo entidade;
- Tipo Relacionamento possui verbo de relacionamento;
- Tipo Relacionamento contém tipo atributo;

Este nível possui um conjunto de estruturas de dados usadas para definir objetos e classes de objetos. Estas estruturas representam o Modelo de Objeto Field Bus que pode ser visto na Figura 2-11.

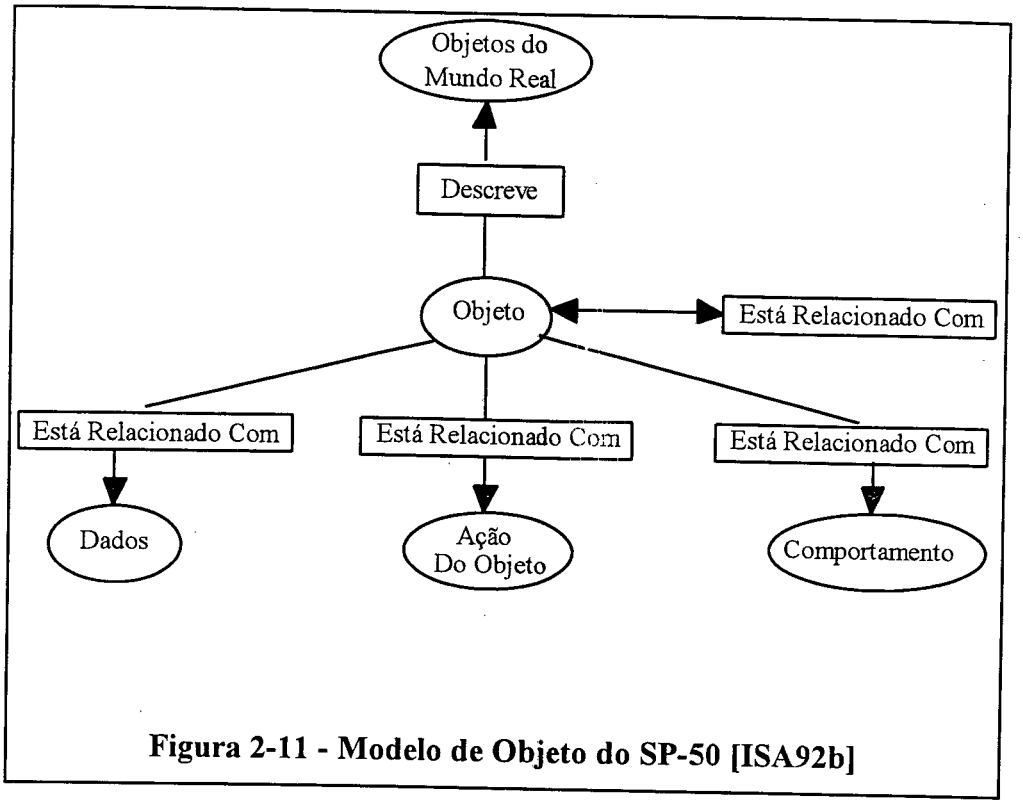


Figura 2-11 - Modelo de Objeto do SP-50 [ISA92b]

Neste Modelo os objetos são definidos através de sua interface, seus atributos, campos, comportamento, e por seus relacionamentos com outros objetos.

As definições de objeto incluem a definição de classes, onde são descritas as características genéricas de uma classe de objetos, e a definição de instâncias de objetos, na qual identifica-se a classe a que instância pertence e provê os valores necessários para dados e parâmetros de interface.

Os Dados de Objetos são seus atributos e dados internos, que são acessíveis a outros objetos. As definições de dados incluem definição de classes de dados e itens de dados.

Interfaces de Objetos, são as Ações, que podem ser :

- ações que um objeto provê;
- ações de outros objetos que este usa;
- ações que operam neste objeto;
- serviços de comunicação que o objeto usa.

As definições de ações são compostas de definições para classes de ações e ações de objeto. Classes de Ações são interfaces de objeto genéricas, definidas por seus

atributos e parâmetros de interface. Elas podem ser atômicas (não interruptíveis) e controláveis (interruptíveis).

Ações de objeto são instâncias das classes de ação. São definidas pelos objetos que as provêem e referenciadas pelos objetos que as usam.

O comportamento do objeto é a definição de como o objeto responde a pedidos de ações e a eventos internos, a partir de seu estado atual. Isto é, o comportamento externo do objeto.

B.1 - Verbos de Relacionamento

Os verbos de relacionamento, apresentados a seguir, definem os modos que as entidades podem se relacionar umas com as outras.

- É uma Subclasse de (Is a Subclass of) : especifica a herança orientada a objetos.
- É uma Instância de (Is Instance of) : é usado para identificar a classe de um objeto.
- Está Relacionado com (Is Related to) : define relacionamentos genéricos entre entidades.

B.2 - Tipos Atributo do Esquema de Enciclopédia

Existem vários tipos atributo, mas sua descrição está fora do escopo deste trabalho. Esta descrição pode ser vista em [ISA92b].

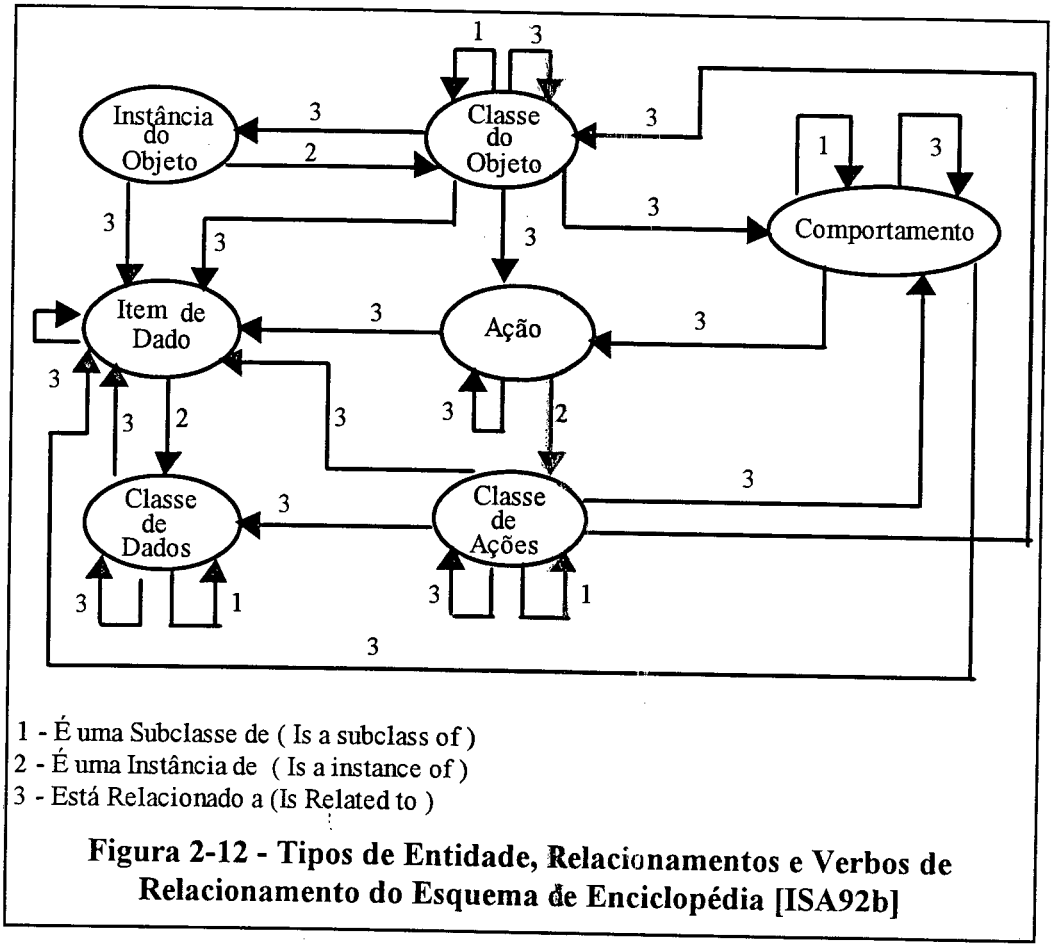
B.3 - Tipos Entidade de Enciclopédia

Tipos Entidade de Enciclopédia são usados para definir objetos. Cada tipo de entidade define um aspecto de um objeto, e é especificado por seus atributos e relacionamentos com outros tipos. Os tipos de entidade são :

- Instância de Objeto (Object Instance)
- Classe de Objeto (Object Class)
- Item de Dado (Data Item)
- Classe de Dados (Data Class)
- Ação (Action)

- Classe de Ações (Action Class)
- Comportamento (Behavior)

A seguir pode-se ver a Figura 2-12 que mostra os tipos de entidades, relacionamentos e verbos de relacionamento do esquema de enciclopédia.



1 - É uma Subclasse de (Is a subclass of)
2 - É uma Instância de (Is a instance of)
3 - Está Relacionado a (Is Related to)

Figura 2-12 - Tipos de Entidade, Relacionamentos e Verbos de Relacionamento do Esquema de Enciclopédia [ISA92b]

2.4.4.2 - Conceitos Gerais da Camada de Aplicação

A camada de aplicação do Field Bus define mecanismos para o suporte do processamento de informações distribuídas em tempo real.

A operação cooperativa em sistemas abertos de tempo crítico é modelada em termos de interações entre processos de aplicação de tempo crítico **(Time Critical Application Process - TCAPs)**

Um TCAP (ou AP) é uma representação abstrata dos modelos de um sistema aberto de tempo crítico real, que executa o processamento de informação para uma aplicação particular. Relacionados a um AP nós podemos definir :

- **Tipo Processo de Aplicação (Application Process Type)**, que representa a descrição da classe de processos de aplicação em termos de um conjunto de capacidades de inter-trabalho;
- **Invocação de Processo de Aplicação (Application Process Invocation - AP-I)**, que é uma utilização específica de parte ou de todas as capacidades de um processo de aplicação para realizar um determinado processamento de informação. A atividade de um AP é dada por uma ou mais AP-Is. A cooperação entre APs se dá através de relações estabelecidas entre AP-Is.

Para gerenciar as relações entre AP-Is é bom agrupar os elementos de serviço. Este agrupamento a nível de camada de aplicação define o **Tipo Entidade de Aplicação (Application Entity Type - AE)**, que é um subconjunto de um **Elemento de Serviço de Aplicação (Application Service Element - ASE)**, sendo que podem coexistir diferentes tipos de AE. Aqui também existem as **Invocações de Entidade de Aplicação (Application Entity Invocation - AE-Is)**, que são a utilização específica de parte ou de todas as capacidades de uma AE no suporte dos requisitos de comunicação de um processo de aplicação. Vale ressaltar que uma comunicação entre APs localizados em dispositivos diferentes só pode ser realizada se eles usarem AEs dos mesmo tipo.

A comunicação estabelecida entre AEs do mesmo tipo denomina-se **Relacionamento de Aplicação (Application Relationship - AR)**, que são modeladas como caminhos de comunicação entre AEs

Um **Objeto de Aplicação (Application Object - AO)** é a entidade que possui os dados a serem transferidos juntamente com seus atributos. Ele está contido em um AP e está acessível através de uma AE usando os serviços de uma ou mais ASEs, sendo que um AO pode conter outros AOs e o conjunto de AOs de comunicação relativos a um AP são reagrupados em **Dispositivos Virtuais de Campo (Virtual Field Devices - VFDs)**.

Um **Dispositivo Virtual de Campo** é uma representação abstrata de um conjunto de recursos e funções providos por um dispositivo real. Pode-se defini-lo também como a ferramenta que provê uma visão específica de um AP para os outros APs envolvidos em um relacionamento, sendo que um AP pode conter um ou mais

VFDs e um VFD pode conter outros VFDs. Além do mais, cada VFD está relacionado a pelo menos um AE. O VFD deve ser mapeado para o dispositivo real de modo a servir a aplicação local, sendo que este mapeamento é de responsabilidade da camada do usuário. Mas vale ressaltar que a vida de um VFD é independente do dispositivo real, isto porque ele é apenas uma representação abstrata.

A Figura 2-13 mostra um exemplo de uma possível estrutura de um dispositivo de campo.

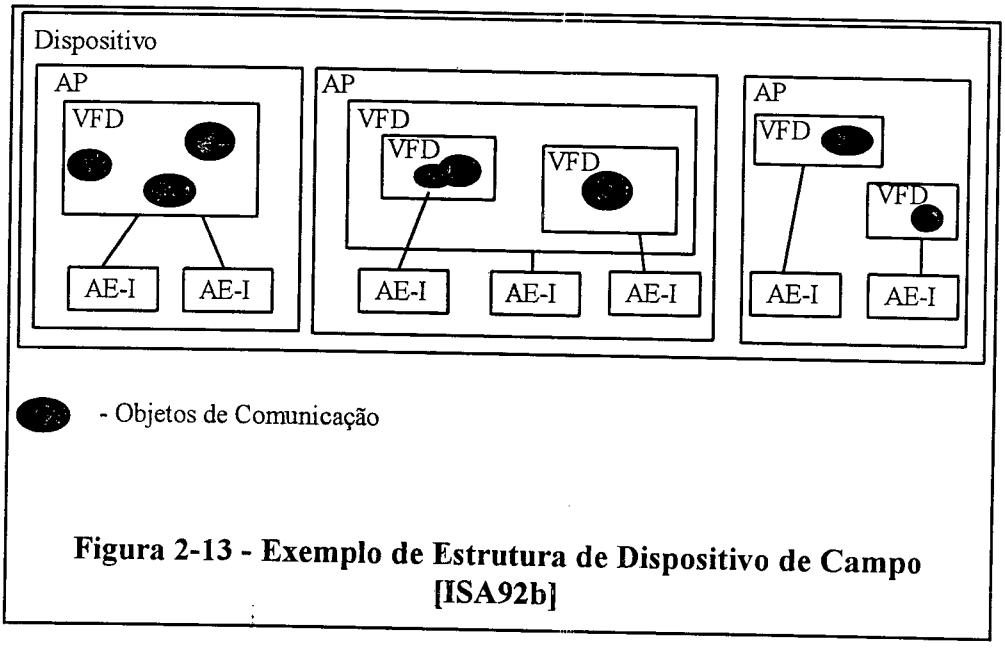


Figura 2-13 - Exemplo de Estrutura de Dispositivo de Campo [ISA92b]

2.4.4.3 Modelo de Objeto Gerenciado (Managed Object Model)

As funções de gerenciamento de objetos são usadas para manipular objetos e informações sobre estes. Estes objetos são instâncias das classes definidas na enciclopédia. Mas nem todos os objetos Field Bus podem ser acessados através destas funções. Estes serviços foram planejados para serem usados com instâncias de objetos cuja a definição da classe não mude durante o ciclo de vida do AP.

Esta funções possuem parâmetros de consistência de modo a manter a enciclopédia de objetos distribuídos consistente. Quando uma definição de objeto é modificada, o AP responsável pela modificação deve prover meios de notificar os outros APs cooperantes.

Os serviços de gerenciamento de objetos permite acessar os seguintes aspectos de um objeto gerenciado:

- atributos;
- ações;
- comportamento.

Os serviços são os seguintes:

- **Get** : é um serviço confirmado que permite ao usuário obter o conjunto de valores de certos atributos.
- **Set** : é um serviço confirmado que permite ao usuário prover os valores para um conjunto de atributos.
- **Create_Instance** : é um serviço confirmado que permite ao usuário da camada de aplicação criar uma ou mais instâncias de uma classe de objetos pré-definida em um AE-I destino.
- **Delete_Instance** : é um serviço confirmado que permite ao usuário da camada de aplicação remover uma instância de um objeto de um ambiente distribuído e invalidar todas as referências a este objeto.

2.4.4.4 Modelo de Relacionamento de Aplicação (Application Relationship Model)

Uma Relação de Aplicação (AR) é uma relação estabelecida entre dois ou mais AEs com o objetivo de transferir informação.

Uma AR-I é uma instância de comunicação entre AEs que participam de uma AR. Ela é composta do conjunto de AE-Is que representam as AE. Cada AE-I envolvida mantém um contexto local. O conjunto de todos estes contextos define o contexto de uma AR-I. A Figura 2-14 exemplifica este conceitos.

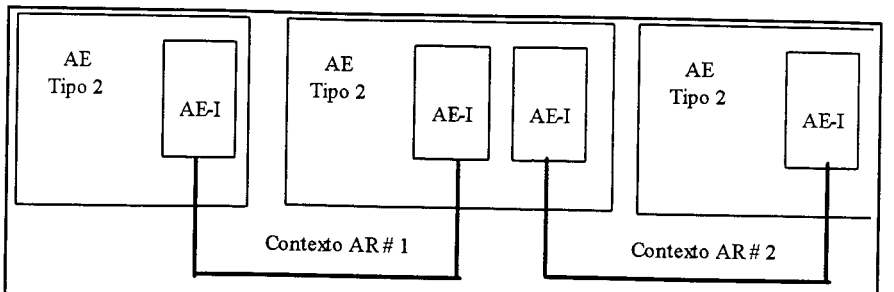


Figura 2-14 - Comunicação entre AEs utilizando ARs [ISA92b]

As ARs são classificadas de acordo com o número de participantes, seus papéis e da política de transferência.

Quanto ao tipo de política de transferência elas são divididas em :

- **AR com fila**, na qual a informação é preservada até ser entregue e removida;
- **AR com buffer**, na qual uma nova informação sobrepõe-se a anterior, independente se esta foi entregue ou não.

As ARs com fila podem ser de três tipos:

- **par-a-par**, a qual suporta serviços confirmados e não confirmados, e qualquer uma das AE-Is pode iniciar ou responder pedidos de serviço;
- **multi-parte não confirmada**, que provê comunicação entre dois ou mais AE-Is, e suporta somente serviços não confirmados. Isto é um editor e vários assinantes, onde o editor inicia os serviços e os assinantes não podem iniciar nem responder serviços;
- **multi-parte confirmada**, que é uma comunicação entre dois ou mais AEs, que suporta serviços confirmados e não confirmados. Esta última será objeto de estudos futuros da ISA.

No tocante as ARs com buffer, todas são pré-definidas, de modo que haja um buffer no lado receptor de cada caminho de transmissão. E as ARs com buffer podem ser estabelecidas entre um editor e um ou vários assinantes.

A Figura 2-15 mostra os tipos de relacionamento do SP50.

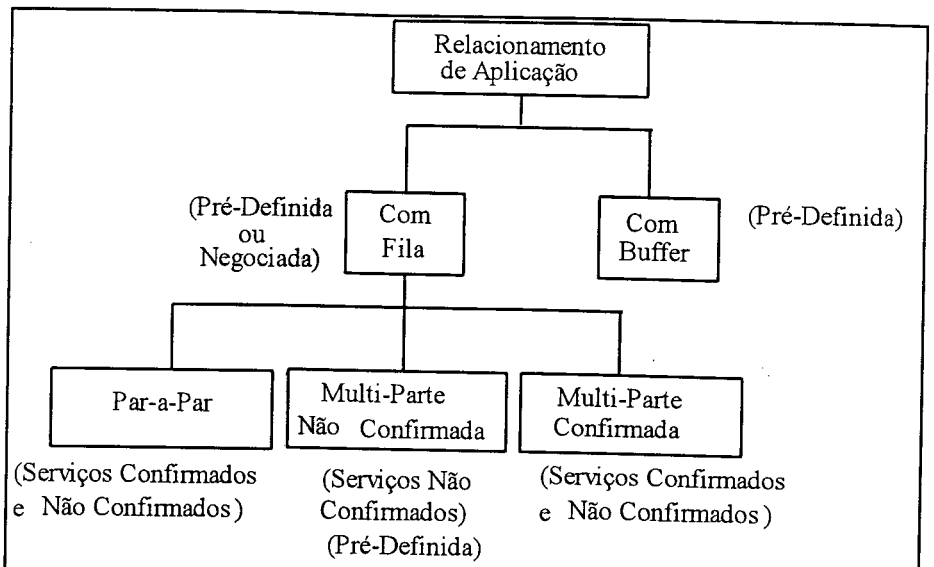


Figura 2-15 - Tipos de Relacionamentos de Aplicação do SP-50 [ISA92b]

Os serviços relacionados ao modelo AR são :

- **Establish** : usado para o estabelecimento, implícito ou explícito, de uma relação de comunicação com fila com ou sem confirmação.
- **DisEstablish** : é usado para terminar de modo normal um contexto de comunicação de uma relação de aplicação.
- **Abort** : é usado para terminar abruptamente uma relação de aplicação.
- **Reject** : indefinido.

2.4.4.5 Modelo de Tipo de Dado (Data Type Model)

O modelo define dois tipos de dados:

- **BÁSICO (BASIC)**, que são as estruturas de dados fundamentais e mais elementares;
- **CONSTRUÍDO (CONSTRUCTED)**, que são criados a partir dos tipos básicos. No entanto devem ser limitadas por questão de eficiência.

Os tipos construídos dividem-se em **ARRANJOS (ARRAYS)** e **ESTRUTURAS (STRUCTURES)**, onde os arranjos são limitados a uma dimensão e as estruturas não podem ser aninhadas.

E os tipos básicos se dividem em

- **simples** : tipos universais largamente utilizados;
- **específicos** : tipos definidos como padrão e usados para descrever variáveis complexas, funções de gerenciamento, etc.

2.4.4.6 Modelo de Variável (Variable Model)

O objeto Variável descreve o mapeamento entre um objeto de comunicação e uma variável real definida na aplicação.

Uma variável é um elemento abstrato do VFD, a qual é capaz de prover ou aceitar, ou ambos, um valor de dado com tipo.

Os serviços permitem acesso a variável inteira ou a um componente, se esta é do tipo construído.

Os serviços que operam sobre um objeto Variável são :

- **F_Read** : usado para obter o valor de uma ou mais variáveis;
- **F_Write** : usado substituir o conteúdo de uma ou mais variáveis com o valores fornecidos no pedido;
- **F_Information_Report** : usado por um usuário cliente para mandar o conteúdo de uma ou mais variáveis para um determinado receptor.

2.4.4.7 - Modelo de Grupo (Group Model)

Um objeto Grupo pode ser usado para relacionar objetos da camada de aplicação que pertençam ao mesmo AP como uma unidade. Eles provêem um mecanismo eficiente (número de mensagens) e confiável (de maneira atômica) para se comunicar com coleções de objetos, sendo que uma instância de objeto pode pertencer a vários grupos ao mesmo tempo.

Os Grupos são classificados de acordo com tipos de participantes ou de acordo com suas operações.

De acordo com o tipo dos participantes temos grupos heterogêneos e homogêneos, baseados nas classes.

Quanto aos serviços temos

- grupos que possuem somente serviços de gerenciamento de grupo;
- grupos que possuem serviços herdados das instâncias que os compõem, além dos serviços de gerenciamento de grupo.

Existem dois tipos de grupo padrão:

- **Grupo de Tipo Múltiplo (Multiple Type Group - MTGs),** que são heterogêneos no tocante a classe e possuem somente serviços de gerenciamento;
- **Grupo de Tipo Único (Single Type Group - STGs)**, que são homogêneos no tocante a classe e possuem operações de gerenciamento e operações herdadas de seus membros, porém com comportamento de grupo .

Vale ressaltar que se um grupo de objetos do mesmo tipo não herda os serviços ele é considerado um MTG.

2.4.5 - Camada de Usuário [ISA91b] [FURN92a]

De modo a apresentar mais transparência e simplicidade para as aplicações, desenvolveu-se uma nova camada, que é a de usuário. Ela pretende realizar a comunicação entre dispositivos existentes atualmente, bem como permitir a inclusão de novos dispositivos, que se beneficiarão de todas as vantagens apresentadas pelo Field Bus.

A camada de usuário possui como base arquitetural os blocos de funções, que realizam o trabalho de aquisição, controle e saída de dados. Os blocos de funções são formados por um algoritmo, uma base de dados e um nome único, que os identifica dentre os demais blocos de função.

A camada de usuário, no momento deste estudo, não se encontrava em um nível razoável de definição, sendo portanto, pouco descrita neste trabalho.

2.5 - Conclusão

Neste capítulo foram apresentadas as três propostas de padronização de Field Bus, mostrando as características das camadas componentes de cada proposta, ressaltando-se os serviços e protocolos da camada de enlace que tiveram um estudo mais aprofundado, pois principalmente sobre esta é que se baseia o restante deste trabalho. Deste estudo mais aprofundado, a norma SP50, tendo em vista sua futura adoção como padrão mundial, teve uma atenção especial. No capítulo 3, nós faremos uma análise comparativa destas três propostas Field Bus.

Capítulo 3

Uma Análise Comparativa dos Padrões de Field Bus

3.1 - Introdução

Neste capítulo apresenta-se uma análise comparativa entre as três propostas que foram expostas no capítulo anterior. O intuito é verificar suas similaridades e divergências, bem como possíveis vantagens e desvantagens de cada proposta. Estas comparações inicialmente abordarão aspectos gerais das propostas, e por fim cobrirão aspectos destas que concernem a sistemas de tempo real.

3.2 - Características Gerais

3.2.1 - Comparação das Camadas Física e de Enlace de Dados

Autores afirmam que o PROFIBUS [EM91] trabalha mais a nível de célula, bem como não apresenta características tempo real e uma simplicidade igual a do FIP. Os mesmos concluem que ele está menos apropriado que o FIP para os instrumentos de campo de controle de processos. Por outro lado, o SP50 procura abranger esta duas áreas, mas é uma proposta bastante complexa.

3.2.1.1 - Características Gerais

A seguir podemos ver dois quadros comparativos destas propostas.

Camada Física					
	SP50		FIP		PROFIBUS
	H1	H2	H1	H2	
Topologia	barramento ou árvore	barramento	home-run (estrela passiva)	Tronco com derivação em estrela	barramento ou árvore (estrela ativa)
Número Máximo de Nós	32		256		32 sem repetidor 62,92,122 e 127 c/ repetidores (1,2 ,3 e 5)
Meio Físico	par trançado blindado		par trançado blindado		par trançado blindado
Alcance Máximo (Km)	1.9	0.75 e 0.5	2	0.5	1.2 sem repetidor 2.4, 3.6 e 4.8 com
Taxa de Transmissão (Kbit/s)	31.25	1000 e 2500	31.25	1000 e 2500	9.6, 19.2, 93.75, 187.5 e 500
Redundância	Transmite em todas e seleciona na recepção		Transmite em todas e seleciona na recepção		Transmite em todas e seleciona na recepção
Codificação	Manchester II		Manchester II		NRZ (Non Return to Zero)
Alimentação via Barramento	Em alguns casos com restrições		Tanto na topologia S1 e S2, com restrições		Em versões futuras
Segurança Intrínseca	Em alguns casos		S1 limitada ao segmento de campo S2 limitada ao segmento terminal		Em versões futuras
Distância Hamming	4 se comprimento da msg <= 344 bytes 5 se comp <= 15 bytes		4		4

Tabela 3-1 - Características Gerais e Camada Física

Camada de Enlace de Dados			
	SP50	FIP	PROFIBUS
MAC	Híbrido	Centralizado	Híbrido
Comunicação ponto-a-ponto	sim	só para mensagens	sim
Broadcast e Multicast	sim	sim	sim
Prioridades	urgente, normal e tempo disponível	urgente e normal	alta e baixa

Tabela 3-2 - Camada de Enlace de Dados

3.2.1.2 - Serviços

No tocante a serviços oferecidos, o SP50 apresenta um grande número, tais como:

- criação e destruição de filas e buffers;
- associar filas ou buffers a DLSAPs;
- escrever ou ler filas e buffers;
- estabelecimento, ressincronização e liberação de conexão;
- transferência de dados com conexão ponto a ponto ou multiponto;
- transferência de dados sem conexão,
- consulta sobre a existência de entidades escutando o meio;
- consulta sobre a existência de assinantes;
- pedido do valor do tempo;
- liberação de um pedido de serviço previamente adiado ;
- escalonamento de seqüências para executar uma vez ou ciclicamente;
- cancelamento deste escalonamento.

Já o FIP apresenta os seguintes serviços :

- escrita/leitura de buffer;
- transferência de buffer;
- pedido explícito de transferência de buffer;
- pedido e transferência de mensagem com ou sem reconhecimento;

E o PROFIBUS possui os 4 serviços seguintes :

- SDA (Send Data with Acknowledge - Envio de Dados com Reconhecimento);
- SDN (Send Data with No Acknowledge - Envio de Dados sem Reconhecimento);
- SRD (Send and Request Data - Envio e Pedido de Dados);
- CSRD (Cyclic Send and Request Data - Envio e Pedido Cíclicos de Dados).

Da apresentação anterior pode-se ver que o SP50 tem uma gama muito maior de serviços, juntando serviços encontrados no FIP e no PROFIBUS, sendo mais abrangente. Porém é também muito mais complexa. Vê-se também que os serviços da camada de enlace de dados do FIP e PROFIBUS são todos sem conexão, enquanto que o SP50 oferece serviços com e sem conexão.

A - Serviços Cíclicos

Quanto aos serviços cíclicos que são de fundamental importância em aplicações industriais com tempo real podemos ver que no PROFIBUS o único serviço cíclico já é pré-definido, o

CSRD. Já no SP50 os serviços cíclicos são definidos durante o escalonamento estático, como no FIP onde os serviços de leitura, escrita e transferência de "buffer" são cíclicos e definidos estaticamente.

B - Serviços Acíclicos

Para os serviços acíclicos o SP50 e o FIP reservam um espaço de tempo de seu escalonamento para o tratamento destes, dentro deste espaço de tempo estes serviços são tratados de acordo com suas prioridades. No PROFIBUS estes serviços são tratados quando do recebimento da ficha de acordo com sua prioridade.

C - Serviços de Mensagens

O FIP é o único em que existe um serviço de mensagens, para o qual um tempo também é reservado no escalonamento

D - Tempo de Atendimento dos Serviços

No tocante ao tempo de atendimento dos serviços, no FIP os espaços (slots) de tempo são definidos a priori para os serviços dos três tipos (cíclicos, acíclicos e mensagens), enquanto no PROFIBUS não há a determinação de espaços de tempo para atender a cada tipo de serviço, mas estes são atendidos durante o tempo de retenção de ficha, e na ordem de prioridades definida, sendo que no PROFIBUS há uma definição de prioridades entre os tipos de serviço, definindo deste modo a seqüência em que estes são realizados quando da retenção da ficha por uma estação: processos acíclicos locais de alta prioridade, processos cíclicos e processos acíclicos locais de baixa prioridade. No SP50 os serviços cíclicos, que na norma SP50 são denominados serviços síncronos no tempo, são servidos de acordo com a tabela de escalonamento enquanto que os acíclicos, denominados de serviços de primeira oportunidade, são servidos durante os intervalos de tempo da tabela reservados para este fim.

E - Tempo de Acesso

No FIP o tempo de acesso é garantido para as transferências de dados cíclicos em qualquer circunstâncias, sendo que para mensagens e transferências de dados acíclicas o tempo de acesso não é garantido, mas como existem espaços de tempo dedicados a realização destes serviços, um limite máximo pode ser definido. No PROFIBUS devido ao seu tipo de abordagem

não existe esta distinção entre os tipos de transferências, mas a garantia é feita de acordo com as prioridades citadas anteriormente. Já no SP50 os serviços cíclicos (já escalonados) tem seu tempo de acesso garantido, sendo que isto não acontece com os acíclicos.

3.2.1.3 - Método de Acesso ao Barramento

No SP50 existe o Escalonador Ativo de Enlace (Link Active Scheduler - LAS), que é responsável pelo escalonamento global e controla a delegação da ficha, sendo portanto similar ao árbitro de barramento do FIP. Porém, o SP50 possui o mecanismo de pseudo ficha circulada, que consiste em uma circulação da ficha delegada por todas as estações ligadas a rede. Este procedimento é parecido com a rotação de ficha entre os mestres que ocorre no PROFIBUS.

3.2.1.4 - Subcamadas da Camada de Enlace de Dados

Vale ressaltar que em nenhuma das propostas há a definição de interface explícita entre as subcamadas do enlace de dados, não definindo deste modo os serviços oferecidos por cada uma. Este aspecto contraria a questão da importância do conceito de serviços defendida por [VISS86] e com a qual concordamos.

A - A Subcamada MAC

O MAC centralizado do FIP, facilita o controle de rastreamento (varredura) do serviço periódico, sendo mais apropriado para configurações onde há um único controlador (elemento centralizador do controle de fluxo de informação), o que corresponde à maioria das aplicações convencionais em plantas industriais. Contudo mostra-se problemático do ponto de vista de confiabilidade [AGUI89].

Já o MAC distribuído, na verdade híbrido[1], baseado na passagem de ficha do PROFIBUS, mostra-se mais confiável, por não possuir elementos centrais dos quais dependa a operação do barramento. Permite configurações multi-controlador, onde cada controlador gerencia o seu próprio fluxo de informação, atendendo a configurações complexas. Porém, regras de sincronização são necessárias para garantir que cada controlador mantenha sua temporização sem perturbar a estabilidade dos outros controladores. No MAC distribuído, pode-se configurar a rede

[1] - Híbrido porque as transferências de dados são controladas por um princípio Mestre/Escravo (centralizado) e transferência da função de Mestre por passagem de ficha (descentralizado).

de tal modo, que apenas um detentor de ficha retenha-a permanentemente, emulando deste modo um MAC centralizado[AGUI89].

O MAC[2] do SP50 é um modelo híbrido pois o controle de acesso ao meio é feito por posse de ficha porém existe uma estação, a LAS, que exerce um controle geral sobre estas fichas. Vale ressaltar que apenas uma estação pode ter suas funções de LAS ativa em um determinado instante, porém todas estações da classe mestre de enlace podem exercer a função de LAS, sendo que esta última característica provê maior confiabilidade ao sistema e é similar ao FIP no que se refere a redundância do árbitro de barramento.

3.2.2 - Comparação entre as camadas de aplicação do PROFIBUS e SP50

Inicialmente vale relembrar que a camada de aplicação do FIP, não foi incluída porque os documentos relativos a esta, que estavam a nossa disposição, estavam muito desatualizados.

Tanto o PROFIBUS quanto o SP50, apresentam um modelo de camada de aplicação orientado a objetos, onde se verificam várias similaridades entre as funcionalidades de alguns tipos de objetos definidos, diferindo apenas na nomenclatura.

3.2.2.1 - Processos de Aplicação

Em ambos tem-se o conceito de **Processo de Aplicação (Application Process - AP),** que é uma representação abstrata dos modelos de um sistema aberto de tempo crítico real, que executa o processamento de informação para uma aplicação particular.

No PROFIBUS os APs trabalham com os **Objetos Processos** (variáveis, programas, etc) , que contêm atributos, regras e descrições de operações sobre os objetos. Eles se assemelham ao **Objeto de Aplicação (Application Object - AO)** do SP50 onde os objetos são definidos através de sua interface, seus atributos, campos, comportamento, e por seus relacionamentos com outros objetos.

O **Objeto de Aplicação** do SP50 está contido em um AP e está acessível através de uma **Entidade de Aplicação (Application Entity - AE)** usando os serviços de uma ou mais **Entidades de Serviço de Aplicação (Application Entity Service Elements - ASEs)**, sendo que

[2]A camada de enlace de dados do SP50 de fato subdivide-se em três níveis com nomenclatura diferente desta.

um AO pode conter outros AOs e o conjunto de AOs de comunicação relativos a um AP são reagrupados em **Dispositivos Virtuais de Campo (Virtual Field Devices - VFDs)**.

As Entidades de Aplicação do SP50 , supracitados, são um agrupamento de elementos de serviço a nível de camada de aplicação, isto é, são um subconjunto de um ASE. Temos então uma **Invocação de AE (AE Invocation - AE-I)**, que é a utilização de parte ou de todas as capacidades de uma AE no suporte dos requisitos de comunicação de um AP. Estes elementos, têm sua correspondência no PROFIBUS nos **Objetos de Comunicação**, que são objetos virtuais conhecidos pelos serviços PROFIBUS, nos quais os Objetos Processos são mapeados.

3.2.2.2 - Canais de Comunicação

Outra similaridade é o das **Relações de Comunicação (Kommunikationsbeziehung - KB)** do PROFIBUS com as **Relacionamentos de Aplicação (Application Relationship - AR)** do SP50. Ambos são canais de comunicação lógicos. Porém estes diferem em alguns aspectos, pois no PROFIBUS todas as KBs são pré-definidas, isto é, são definidas na fase de projeto, enquanto no SP50 as ARs com buffer são pré-definidas e as ARs com fila não possuem esta restrição.

3.2.2.3 - Estruturas para Descrições de Objetos

O SP50 usa uma Estrutura de Informação de Enciclopédia, que define a estrutura da enciclopédia e a estrutura das definições dos objetos que estão contidos na enciclopédia. Ela é composta de 4 camadas de definições :

- Descrição do Esquema.;
- Esquema de Enciclopédia.;
- Dicionário e Diretório;
- Objetos do "Mundo Real".

O PROFIBUS utiliza um Dicionário de Objetos (Objektverzeichnis OV), que pode consistir de até seis sub-dicionários de objetos, onde cada um destes contém descrições de uma certa classe de objetos de comunicação.

3.2.2.4 - Virtual Field Bus Device

Ambos usam o conceito de VFD , bem como permitem que um dispositivo tenha mais que um VFD relacionado a ele e que VFDs contenham outros VFDs. No PROFIBUS, o VFD representa a parte de um processo de aplicação que pode ser visível e gerenciada pelos serviços FMS. Este também, tem como base um tipo de objeto, o Objeto VFD, que registra todos os Objetos de Comunicação representados, utilizando para isto, as Descrições de Objetos (Objektbeschreibung - OB) que estão em um Dicionário de Objetos (OV). Já no SP50, um VFD é uma representação abstrata de um conjunto de recursos e funções providos por um dispositivo real. Pode-se definí-lo também como a ferramenta que provê uma visão específica de um AP para os outros APs envolvidos em um relacionamento.

3.2.2.5 - Serviços

Existem alguns serviços que possuem praticamente a mesma funcionalidade. Na tabela a seguir pode ser vista a equivalência entre os serviços PROFIBUS e SP50. Vale ressaltar que o SP50 está ainda sendo definido, logo o conjunto de serviços está incompleto.

Classe	PROFIBUS	SP50
Gerência de Contexto	Initiate (1) Abort (2) Reject (3)	Establish (1) DisEstablish Abort (2) Reject (3)
Gerência de Domínios	Initiate Download Sequence Download Segment Terminate Download Sequence Initiate Upload Sequence Upload Segment Terminate Upload Sequence Request Domain Download Request Domain Upload	
Gerência de Objetos		Get , Set Create Instance Delete Instance
Gerência de Diretório de Objetos	Get_OV Initiate_Put_OV Put_OV Terminate_Put_OV	
Suporte de VFD	Status Unsolicited Status Identify	
Gerência de Eventos	Alter Event Condition Monitoring Event Notification Event Notification With Type Acknowledge Event Notification	
Acesso a Variável	Read (4)	F-Read (4)

Classe	PROFIBUS	SP50
	Write (5) Read With Type Write With Type Phys Read Phys Write Information Report (6) Information Report With Type Define Variable List Delete Variable List	F-Write (5) F-Information Report (6)
Invocação de Programa	Create Program Invocation Delete Program Invocation Start , Stop Resume ,Reset, Kill	

Tabela 3-3 - Comparação dos Grupos de Serviços da Camada de Aplicação

3.2.2.6 - Tipos de Objeto

Quanto aos tipos de objetos, no PROFIBUS tem-se os seguintes :

- Variável (Simples, Arranjos, Registros);
- Domínio;
- Evento;
- Lista de Variável;
- Invocação de Programa.

No tocante a fase de definição, existem dois tipos de objetos no PROFIBUS :

- objetos estáticos, que são declarados durante a fase de projeto ou na de inicialização;
- objetos dinâmicos, que podem ser declarados durante a fase de comunicação.

Os Objetos Variáveis, Eventos, Domínios são apenas estáticos, enquanto que os Objetos Lista de Variáveis e Invocação de Programa podem ser declarados em qualquer fase. Estas restrições durante a fase de comunicação são feitas por questões de eficiência de transmissão.

Os tipos de objetos que já foram definidos no SP50 são os objetos Variável e Grupo. Os objetos Event e Function Invocation são citados, mas ainda estão indefinidos. E não se sabe quais são os objetos restantes. Outro fato é que até o presente momento, pouco se definiu sobre a dinamicidade ou não dos objetos do SP50. Apenas é citado que existem classes pré-definidas, cujas instâncias são criadas dinamicamente.

3.2.2.7 - Tipos de Dados

Quanto aos tipos de dados os do PROFIBUS são os seguintes:

- Boolean, Integer, Unsigned, Floating Point, Octet String, Visible String, Date, TimeOfDay, TimeDifference.

Já o SP50 define dois tipos de dados:

- BÁSICO (BASIC), que são as estruturas de dados fundamentais e mais elementares, e se dividem em :
  - * simples : tipos universais largamente utilizados (Boolean, Bitstring, Integer, Unsigned, Octet String, Visible String, Universal Time, Floating Point, Binary Time, Bcd);
  - * específicos : tipos definidos como padrão e usados para descrever variáveis complexas, funções de gerenciamento, etc.
- CONSTRUÍDO (CONSTRUCTED), que são criados a partir dos tipos básicos. Os tipos construídos dividem-se em ARRANJOS (ARRAYS) e ESTRUTURAS (STRUCTURES). Onde os arranjos são limitados a uma dimensão e as estruturas não podem ser aninhadas.

3.2.2.8 - Modelo de Comunicação

No que se refere a modelo de comunicação, o PROFIBUS utiliza o modelo Cliente/Servidor, enquanto que o SP50 pode utilizar tanto um similar ao Produtor/Distribuidor/Consumidor (herdado do FIP), quanto o Cliente/Servidor (herdado do próprio PROFIBUS).

3.2.2.9 - Resumo Comparativo

A tabela a seguir é um resumo das comparações feitas anteriormente.

Característica	PROFIBUS	SP50
Processamento de Informação	Processo de Aplicação (AP)	Processo de Aplicação (AP)
Objetos que compõem um AP	Objetos Processos	Objetos de Aplicação
Suporte de Comunicação	Objetos de Comunicação	Invocação de Entidade Aplicação (AE-I)
Canais Lógicos de Comunicação	Relações de Comunicação (KB)	Relacionamentos de Comunicação (AR)
Descrições dos Objetos e	Dicionário de Objetos	Estrutura de Informação de

Característica	PROFIBUS	SP50
suas Classes		Enciclopédia
Representação dos Dispositivos	Dispositivo Virtual de Field Bus (VFD)	Dispositivo Virtual de Field Bus (VFD)
Tipos de Objetos e sua Dinamicidade	• Variável, Domínio e Evento (estáticos) • Lista de Variáveis e Invocação de Programas (estáticos ou dinâmicos)	• Váriável, Grupo, evento e Invocação de Programa (dinamicidade ainda não definida)
Tipos de Dados	• boolean, integer, unsigned, floating point, octet string, visible string, date, Time of Day, Time Difference	• Básico * simples (boolean, bitstring, integer, unsigned, octet string, visible string, universal time, floating point, binary time, bcd) * específicos • Construídos * arranjos * estruturas
Modelo de Comunicação	• Cliente/Servidor	• Produtor/Distribuidor/Consumidor • Cliente/Servidor

Tabela 3-4 - Resumo da Comparação entre as Camadas de Aplicação do PROFIBUS e do SP50

3.3 - Mecanismos de Controle e Escalonamento das Três Propostas

Nesta seção serão apresentados os mecanismos de controle e escalonamento das três propostas, que dizem respeito aos aspectos de tempo real. É realizada também, uma breve comparação sob a ótica de tempo real.

3.3.1 - FIP

3.3.1.1 - Mecanismos de Controle

A - Temporizadores

O FIP apresenta seus mecanismos de controle de modo a atender as necessidades de suas aplicações. Temporizadores são utilizados para detecção de perdas de quadros, bem como para os serviços aperiódicos.

O árbitro de barramento dispara temporizadores após a emissão de certos quadros. Por exemplo, após a emissão do quadro identificador de variável (ID_DAT) ou do quadro identificador de pedido explícito de transferência de buffer (ID_RQi) um temporizador é disparado e este expira caso o quadro de resposta não chegue.

Algo semelhante acontece após a emissão do quadro identificador de mensagem (ID_MSG), sendo que para mensagens com reconhecimento ele também dispara um temporizador para reconhecimento.

O FIP usa uma temporização, denominada T0, que inicia a transmissão do quadro identificador seguinte quando o quadro de resposta referente ao identificador corrente foi perdido. Isto acontece quando a condição seguinte é satisfeita:

- T0 > tempo máximo de resposta + atraso máximo de propagação

Caso ocorra a chegada do quadro, T0 é desabilitada.

B - Numeração

Outro mecanismo existente é o de numeração, este que é usado para evitar duplicação de quadros e também para controle de fluxo

C - Prioridades

Utilizam-se filas para o escalonamento dos serviços aperiódicos, distinguido os serviços pela sua prioridade (urgente ou normal)

D - Tempo de Resposta

No FIP existe o tempo de resposta (retorno) da estação, que é o intervalo de tempo que separa o fim do quadro identificador, do início da emissão ou recepção do quadro de resposta. Este tempo é um valor conhecido por todas as estações da rede para garantir a interoperabilidade dos elementos conectados.

Renovação (Refresh)

O FIP utiliza princípio de renovação(refreshment) do valor das variáveis periódicas, de modo a manter os valores destas atualizados

3.3.1.2 - Escalonamento

O FIP tem por filosofia a garantia das transferências cíclicas. Estas são conhecidas a priori e com base nestes dados é realizado o escalonamento estático. Durante este escalonamento é reservado um espaço de tempo para as acíclicas, assim como para as mensagens. Com isto define-se um número mínimo e máximo de transferências acíclicas e mensagens. A Figura 3-1 mostra um exemplo de agenda de escalonamento do FIP.

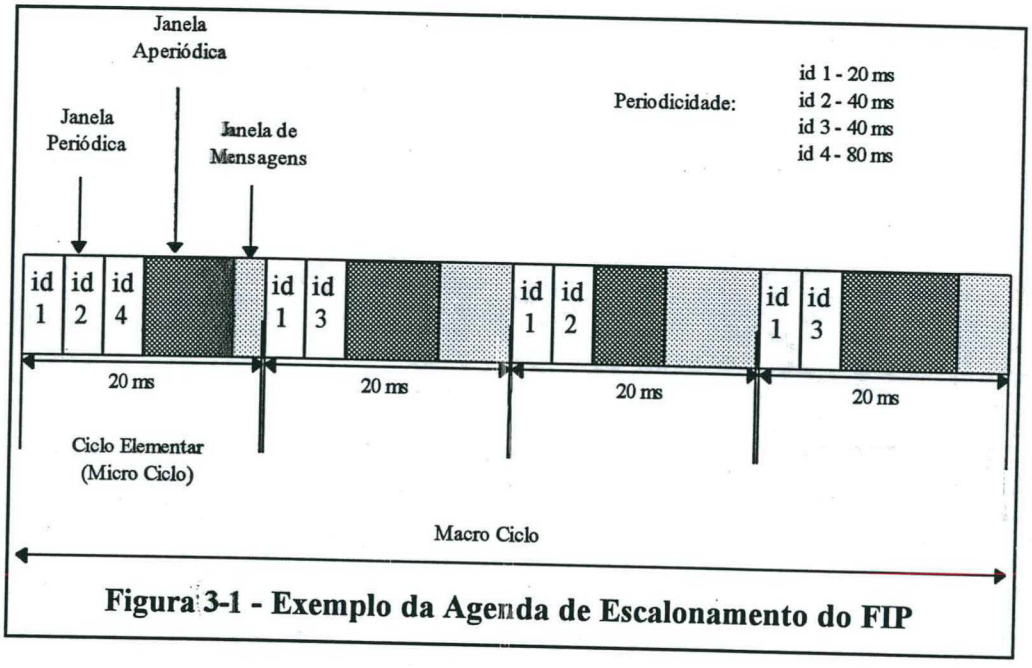


Figura 3-1 - Exemplo da Agenda de Escalonamento do FIP

Para a realização da configuração é necessário que :

- seja fornecido ao árbitro de barramento a tabela dos identificadores de serviços cíclicos, e indicar-lhe o tamanho das janelas de serviços acíclicos e de mensagens.
- seja fornecido a cada entidade produtora/consumidora a tabela de identificadores reconhecidos na emissão (variáveis produzidas) e na recepção (variáveis consumidas).

Também utiliza-se um mecanismo de configuração no qual todas as estações são consultadas e devem responder enviando suas características. Se o equipamento é pré-configurado, esta resposta deve conter apenas

- uma identificação não ambígua do equipamento
- uma indicação de que o equipamento é pré-configurado

- um identificador que permite ao configurador conhecer a configuração da estação de modo a verificar a coerência da instalação.

No caso das estações a configurar, o configurador vai carregar remotamente os seguintes parâmetros :

- a lista dos identificadores produzidos e consumidos;
- os tempos de prontidão[3] e renovação[4];
- o tamanho ou o tipo da variável associada a cada identificador;
- o tamanho do buffer de mensagens;
- um parâmetro de validação à nível de enlace de dados;
- um checksum global da tabela de configuração que permitirá verificar a coerência da configuração.

3.3.2 - PROFIBUS

3.3.2.1 - Mecanismos de Controle

O PROFIBUS apresenta uma série de mecanismos de controle que possibilitam sua operação, assim como auxiliam na tentativa de cumprir os requisitos de tempo necessários a aplicação de campo. Ressaltamos os seguintes :

A - Temporizadores

- Temporizador de Rotação da Ficha : usado para controlar o tempo real de rotação da ficha;
- Temporizador de Ociosidade : usado para controlar o envio de novos quadros de ficha ou pedidos (na estação detentora da ficha) ou para habilitar o receptor (nas outras);
- Temporizador de Slot : monitora se a estação receptora está ativa após a transmissão de um pedido ou passagem da ficha, controlando deste modo as retransmissões;

[3] - o tempo de prontidão é computado pela estação receptora quando recebe o dado e indica se a operação disparada por um pedido de atualização respeitou as restrições temporais definidas.
[4] - o tempo de renovação é computado pela estação produtora e indica se a operação de produção respeitou as restrições temporias definidas.

- Temporizador de Time-out : controla a atividade do barramento, reportando um erro, após um certo tempo de inatividade detectado;
- Temporizador de Intervalo de Sincronização : utilizado para a sincronização das estações;
- Temporizador de Atualização da GAP : quando expira, ele indica a necessidade da manutenção da GAP;

B - Listas

- Lista de Estações Ativas (LAS) : lista de estações mestres ou escravas que estão ligadas e operacionais na rede.
- Lista de GAP : lista que compreende a faixa de endereços entre uma estação e sua sucessora no anel;
- Lista de Consulta (Poll List) - para controle de consulta cíclicas, indica quais estações devem ser consultadas. As estações que não respondem, após as retransmissões, são marcadas como não operacionais, sendo que em consultas seguintes, retransmissões não serão feitas para estas estações.

C - Contadores

Contadores : para controle dos quadros enviados, recebidos e retransmitidos;

3.3.2.2 - Escalonamento

No PROFIBUS o escalonamento consiste em cálculos e monitoria de tempos que determinam o tipo e o número de mensagens permitidas a cada recepção da ficha.

- **Tempo de Rotação da Ficha (Token Rotation Time)** : é calculado entre duas recepções consecutivas da ficha e resulta na verdade no **Tempo Real de Rotação da Ficha (Real Token Rotation Time - T_{RR})**. O T_{RR} é importante para transmissão de mensagens de baixa prioridade.
- **Tempo de Rotação Alvo da Ficha (Target Rotation Time - T_{TR})** : é definido para manter o sistema dentro do tempo de reação requerido pela aplicação de campo.

O tempo de reação é definido em [PROF91] como o intervalo máximo (pior caso) entre dois ciclos de mensagens de alta prioridade de uma estação mestre na carga máxima do barramento

Independente do tempo real de rotação cada mestre deve sempre executar um ciclo de mensagens de alta prioridade para cada recepção da ficha, isto é, pelo menos um.

Para realizar o ciclo de mensagens de baixa prioridade o T_{RR} deve ser menor que o T_{TR}, senão as mensagens de baixa prioridade serão transmitidas nas recepções seguintes da ficha, quando possível.

O Tempo de Rotação Alvo (T_{TR}) mínimo do sistema depende :

- do número de estações mestres e consequentemente dos ciclos da ficha (Token Cycle - T_{TC})
- da duração dos ciclos de mensagem de alta prioridade (High T_{MC})

O Tempo de Rotação Alvo (T_{TR}) pré-definido deve conter tempo suficiente para a transmissão de mensagens de baixa prioridade e uma margem de segurança para retransmissões potenciais. Para conseguir o menor Tempo de Rotação Alvo(T_{TR}) possível deve-se, juntamente com a camada de aplicação, declarar somente os eventos importantes e os que raramente ocorrem como mensagens de alta prioridade e deve-se restringir seu tamanho

Para calcular o T_{TR} é utilizada a seguinte equação [PROF91]:

Min TTR = ne * (T_{TC} + high T_{MC}) + k * low T_{MC} + rm * RET T_{MC}

Onde

- ne : número de estações mestres
- k : número estimado de ciclos de mensagens de baixa prioridade por rotação d ficha
- T_{TC} - Tempo de Ciclo da Ficha (Token Cycle Time)
- T_{MC} - Tempo de Ciclo de Mensagem (Message Cycle Time), depende do comprimento do quadro
- rm - número de retransmissões de mensagens por rotação da ficha
- RET T_{MC} - tempo de ciclo de retransmissões de mensagens
- high - relativo a mensagem de alta prioridade
- low - relativo a mensagem de baixa prioridade

Na equação anterior o primeiro termo contém um ciclo de mensagens de alta prioridade por estação mestre e uma rotação de ficha. Logo, o tempo de reação máximo para ciclos de mensagens de alta prioridade sem retransmissões é garantido para todas as cargas do barramento. O segundo termo contém um número estimado de mensagens de baixa prioridade e o terceiro uma margem de segurança para retransmissões.

Quando da recepção da ficha calcula-se o Tempo de Retenção da Ficha: Token Holding Time - T_{TH} [PROF91]

$$T_{TH} = T_{TR} - T_{RR}$$

No PROFIBUS uma vez iniciada a transmissão, ela sempre será completada, incluindo retransmissões. Se T_{RR} exceder o T_{TR}, o tempo de transmissão será diminuído na próxima recepção da ficha, para compensar o tempo perdido.

O Ciclo de Consulta (Poll Cycle) depende do atraso máximo da estação, da duração do ciclo de mensagem, do tempo de rotação da ficha, do tamanho da lista de consulta e dos ciclos de mensagens de baixa prioridade.

No PROFIBUS a adição e remoção de estações é dinâmica, sendo que isto pode afetar o tempo de retenção da ficha e consequentemente a transmissão de mensagens.

3.3.3 - SP50

3.3.3.1 - Mecanismos de Controle

De modo a garantir o bom funcionamento do protocolo, existem alguns mecanismos de controle. São eles:

- temporizadores, que são usados para monitorar certos aspectos como atraso máximo de confirmação e atraso máximo de reconhecimento.
- monitoração : durante um intervalo de tempo especificado o enlace é monitorado para detectar se há atividade de enlace, com o intuito de verificar se a ficha foi recebida corretamente.
- atualização do tempo : as DLEs podem ter um sincronismo do tempo e o LAS periodicamente envia seu valor de tempo para as outras DLEs para a atualização

deste tempo. Caso o LAS não envie este valor até um determinado limite, a DLE que detectar este problema requisita ao LAS que faça o envio.

- contadores, usados para controlar alguns aspectos de tempo, como por exemplo, tempo de posse da ficha .
- variáveis para salvar aspectos importantes como o período de distribuição do tempo.
- Lista-Viva (Live-List), que é usada para gravar as DLEs ativas do enlace.
- filas, usadas para auxiliar nas funções de escalonamento.
- mecanismos de remoção : a LAS após quatro falhas consecutivas de uma DLE em receber a ficha, remove esta da Lista-Viva.

3.3.3.2 - Escalonamento

O escalonamento global pode ser visto como um ciclo repetitivo de três classes de atividades:

- atividades cíclicas (síncronas no tempo), que foram escalonadas para serem executadas em tempos específicos relativos ao tempo de inicialização da execução de todo o escalonamento. A maioria destes serviços são periódicos, mas alguns podem ser executados uma só vez.
- atividades acíclicas (de primeira oportunidade), que são executadas com base em uma fila de prioridades na primeira oportunidade que o escalonamento permite.
- atividades de "consulta", divididas em :
  - * atividades de verificação de endereços, usadas para verificar se uma nova DLE necessita entrar no enlace.
  - * atividades simulando uma ficha circulada, conhecida como pseudo ficha circulada, onde a circulação é feita na ordem da Lista-Viva [ISA93B].

A construção do escalonamento consiste primeiramente numa parte estática, isto é, o escalonamento das atividades síncronas no tempo. Como estas atividades representam a maior parte do tráfego existente e elas são conhecidas a priori, o LAS usa uma agenda de escalonamento para este fim [CAVA91] [SMAR93]. Na construção desta agenda reserva-se espaços de tempo, dentro dos quais serve-se a fila de primeira oportunidade.

Um escalonamento dinâmico é realizado durante a execução do escalonamento, sendo que neste as DLEs podem indicar a necessidade de novas ações de escalonamento através do subcampo de pedido de delegação de algumas DLPDUs. Estes pedidos quando chegarem ao LAS serão colocados em uma fila por prioridades.

Durante a execução do escalonamento é necessário servir as transações cíclicas (síncronas no tempo) e as acíclicas (de primeira oportunidade). As cíclicas são servidas de acordo com a agenda de escalonamento enquanto que as acíclicas são servidas durante os intervalos de tempo da agenda reservados para este fim. Um exemplo da agenda de escalonamento [SMAR93] pode ser visto na Figura 3-2.

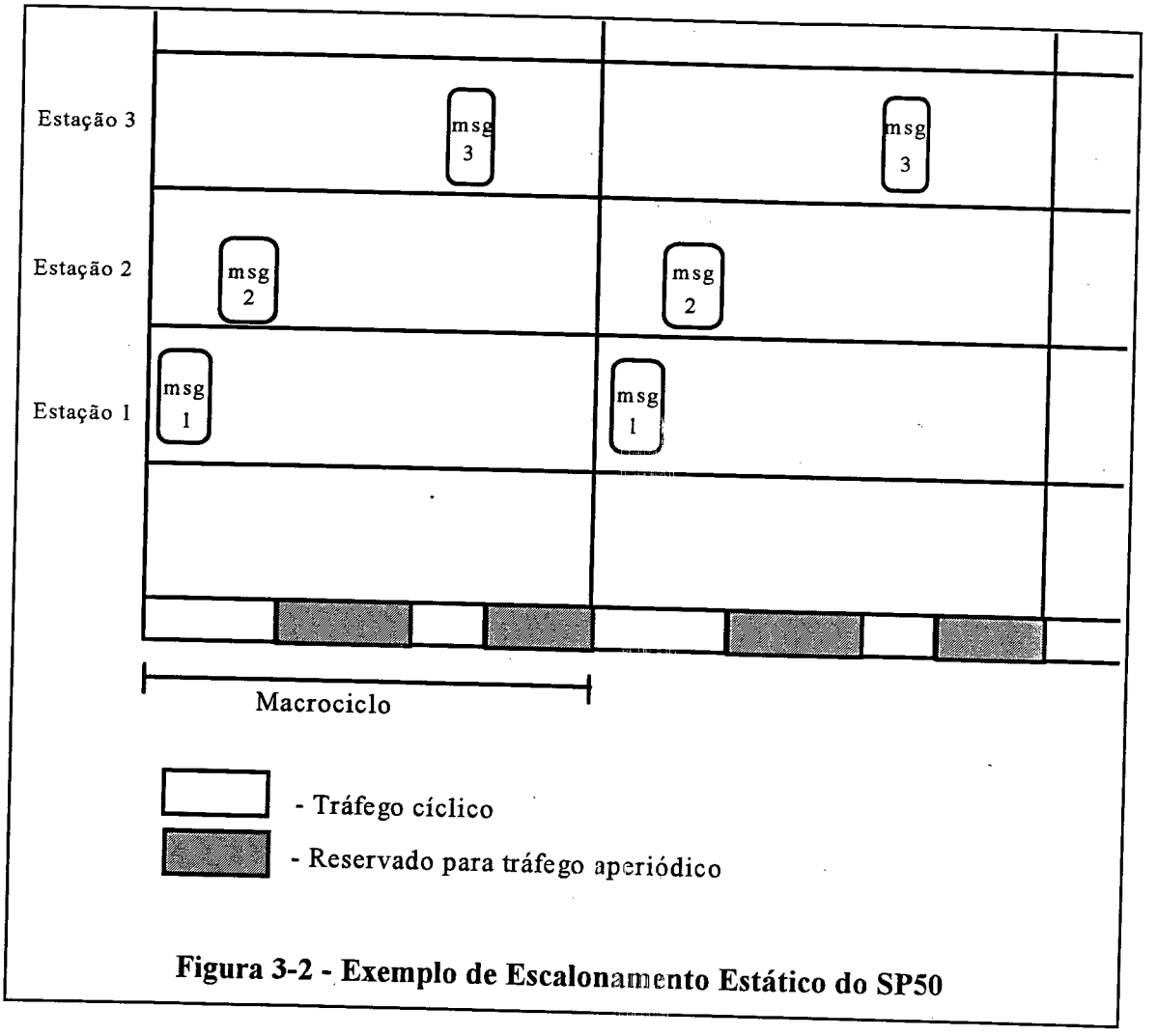


Figura 3-2 - Exemplo de Escalonamento Estático do SP50

As atividades de verificação de endereços e pseudo ficha circulada são realizadas nos intervalos de tempo em que não haja nem atividades cíclicas nem acíclicas a serem realizadas. As atividades de pseudo ficha circulada também podem ser escalonadas como as outras atividades.

Cada DLE faz seu escalonamento local e suas comunicações baseada em três filas:

- fila de pedido de usuário : uma fila de prioridades associada a um endereço DL específico de uma DLE contendo pedidos de usuário;

- fila de escalonamento de serviços : uma fila por prioridades associada a um endereço DL específico de uma DLE com referências a, pedidos na fila de pedidos de usuário, seqüências ativas, e SPDUs na fila de pedidos para a LAS;
- fila de pedidos para a LAS : uma fila por prioridades com SPDUs (Support PDUs) que a DLE deve transmitir a LAS ou referências a escalonamentos de um único elemento para executar uma única vez na primeira oportunidade.

3.3.4 - Algumas Considerações Complementares sobre a Comunicação Tempo Real

Os requisitos considerados a seguir estão baseados nos conceitos apresentados em [FONS93].

3.3.4.1 - Requisitos de Previsibilidade e Garantia de Entrega

A - Serviços Periódicos

Considerando-se serviços periódicos com alta prioridade, para o caso de um período pequeno, o PROFIBUS apresenta o problema que a ficha tem que rodar rapidamente, com tempo de retenção pequeno. Logo, muitas estações vão receber a ficha e não vão ter nada a transmitir. Já no FIP e no SP50 este serviço será atendido de acordo com o escalonamento estático.

Para o caso de um período grande o PROFIBUS não apresenta problemas, basta calcular o tempo de rotação da ficha para determinar que no momento de servir os periódicos a estação esteja de posse da ficha. Porém se no momento houver um outro serviço de mais alta prioridade, este é que será servido. No FIP e no SP50 também o serviço é atendido de acordo com o escalonamento estático.

B - Serviços Aperiódicos

Para o tratamento dos serviços aperiódicos os requisitos de previsibilidade e garantia de entrega não podem ser garantidos, no melhor caso podem ser caracterizados pela probabilidade de ocorrência deste serviços. No FIP, ele depende do tempo reservado para as acíclicas e do número máximo de acíclicas permitidas. No SP50 há a dependência do tempo reservado para as acíclicas e do tamanho da fila de primeira oportunidade. E por último no PROFIBUS depende do

tempo de rotação da ficha e do tempo de retenção desta. No PROFIBUS não há diferenciação quanto ao tratamento de serviços aperiódicos e periódicos exceto o fato que os periódicos são conhecidos a priori e pode-se usar isto para calcular o tempo de rotação e de retenção da ficha. No PROFIBUS pode-se calcular o tempo de rotação da ficha para que as estações recebam mais rapidamente a ficha (menor tempo de rotação) para poder atender eventuais aperiódicas. Porém existe o problema de a ficha circular rapidamente por muitas estações, e estas não terem nada para transmitir. Por outro lado não há o problema do tempo entre pedir a ficha e recebê-la, tal como pode existir no SP50.

No FIP e SP50 não há grande desperdício de tempo de utilização do barramento com o tempo reservado para as acíclicas, pois como o mecanismo de controle é centralizado somente as estações que requisitaram a ficha, receberão esta. Contudo há o problema do tempo de requisição até o tempo de execução.

No SP50 para as atividades acíclicas (de primeira oportunidade), o aspecto de previsibilidade não é coberto se adotarmos rigidamente o conceito definido em [FONS93][5], porém podemos dizer que a questão de previsibilidade, no SP50, é parcialmente realizado pelo atraso máximo de confirmação, isto porque existem temporizadores associados a cada primitiva de pedido de serviço. Quando estes temporizadores expiram, eles indicam a impossibilidade de completar o serviço.

3.4 - Conclusão

Neste capítulo realizamos uma comparação entre as propostas de Field Bus. Diante disto podemos verificar que a proposta SP50 teve uma forte influência das outras duas, e procurou abranger as áreas cobertas por estas. Foram analisadas e comparadas diversas características de mecanismos de controle e escalonamento visando o atendimento das restrições de tempo real. Estas que, nos levam a concordar com autores que dizem ser o FIP mais apropriado que o PROFIBUS para a realização destas tarefas no mais baixo nível da planta e ficando o PROFIBUS com sua atuação mais adequada a nível de célula, e o SP50 englobando estes dois mundos.

Após o estudo das três principais propostas de padronização de Field Bus realizado no capítulo 2 e da comparação destas realizada neste capítulo, e considerando que o SP50 tende a ser o futuro padrão mundial, traçou-se como objetivo o desenvolvimento de uma arquitetura de comunicação SP50. Para isto, torna-se necessário a utilização de uma metodologia de

[5] - Previsibilidade é analisar a capacidade do sistema em atender restrições de tempo da mensagem e caso contrário propor sua rejeição
- Garantia de entrega é assegurar que a mensagem vai ser entregue em tempo hábil [FONS93].

desenvolvimento. O capítulo 4 mostra a importância do uso de uma metodologia, os critérios de escolha de uma metodologia e por fim descreve a metodologia adotada.

Capítulo 4

Uma Metodologia para o Desenvolvimento de Protocolos de Comunicação

4.1 - Introdução

Os protocolos de comunicação, por sua natureza distribuída, apresentam uma complexidade muito maior quando comparados aos softwares para sistemas centralizados. Logo, é imperativo o uso de técnicas de engenharia de software para controlar esta complexidade, com intuito de produzir um software de qualidade. A definição dos passos a serem adotados na construção do software, usando essas técnicas, caracteriza a metodologia de desenvolvimento a ser seguida. Esta metodologia deve prover meios simples e eficientes de estruturar o processo de desenvolvimento, levando de uma descrição informal a uma especificação de projeto detalhada, antes de realizar as fases de implementação e testes.

O uso de uma metodologia se mostra importante, pois ela define uma abordagem sistemática para todas as fases da concepção de um software, definindo claramente os objetivos, resultados e marcos de cada fase. Uma boa metodologia deve abordar não somente as fases de especificação, projeto e implementação, mas também deve dar grande atenção a fase de validação. Além disto, esta metodologia deve considerar a importância da fase de manutenção do software.

Na definição de uma metodologia há vários fatores que influenciam. No nosso caso um ponto fundamental foi o uso de Técnicas de Descrição Formal (TDF).

Neste capítulo inicialmente ressalta-se a importância do uso de Técnicas de Descrição Formal dentro de uma metodologia, bem como o porque da escolha de Estelle como a TDF de suporte à metodologia adotada.

Depois será apresentada a metodologia de refinamentos sucessivos baseada na TDF Estelle. Dentro desta metodologia também destaca-se a importância da fase de validação em todos os níveis da metodologia, de modo a fornecer uma especificação e implementação correta dos serviços e protocolo.

4.1.1 - A Importância do Uso das Técnicas de Descrição Formal

As Técnicas de Descrição Formal são técnicas com embasamento matemático, que permitem descrever de maneira precisa e concisa o serviço esperado e o protocolo que o implementa, retirando as ambigüidades inerentes de uma linguagem informal e possibilitando o desenvolvimento de protocolos de comunicação mais confiáveis e de fácil manutenção.

A especificação formal não deve ser vista como uma substituta da análise de requisitos e especificação em linguagem natural, mas sim um complemento destas. Além disto ela ajuda a melhorar a qualidade da especificação informal, isto porque as notações formais levam o especificador a levantar questões que não haviam sido perguntadas ou estavam sem resposta na abordagem informal.

A realização de uma especificação formal facilita uma análise de sistemas detalhada, o que usualmente permite revelar eventuais erros de inconsistência na especificação informal ou mostrar que esta se encontra incompleta, além disto, provê um maior entendimento do comportamento do sistema. Uma especificação formal serve como um meio de comunicação preciso entre os

projetistas e o implementadores, e somente ela pode servir como referência comum para que diferentes implantações de um mesmo protocolo sejam realizados em redes heterogêneas.

Além disto, o uso de ferramentas automatizadas para os processos de verificação, simulação e emulação, possibilitam detectar mais facilmente os erros nas fases preliminares do desenvolvimento do protocolo. Visto que, quanto mais cedo detectam-se os erros menos custosos eles são, logo o uso de ferramentas para especificação formal é imprescindível. Não pode-se deixar de destacar também a possibilidade de geração parcial do código a partir da especificação formal, o que aumenta confiabilidade da implementação, bem como minimiza o tempo para sua realização. Outra possibilidade é a geração de seqüência de testes para verificar a conformidade da implementação.

4.1.2 - A Escolha de ESTELLE

Estelle (Extended State Transition Language) é uma técnica de descrição formal (TDF) baseada no modelo de Máquinas De Estados Finita Estendida e na linguagem de programação Pascal. Estelle é uma linguagem com grande poder de expressão do paralelismo e da comunicação, o que a caracteriza como uma boa ferramenta para a concepção de sistemas distribuídos. [BUDK87] diz que Estelle é uma linguagem para a especificação de sistemas distribuídos com uma aplicação particular em mente, que são os protocolos e os serviços de comunicação.

Estelle permite uma especificação modular, isto é, a estruturação de uma especificação em módulos, sub-módulos e assim por diante. Estelle permite separar a descrição das interfaces de comunicação, da descrição do comportamento interno de cada componente da especificação. Como em Pascal, todos os objetos manipulados são "fortemente tipados", o que permite detectar estaticamente inconsistências da especificação [DEMB89].

Além disto, Estelle permite uma abordagem de refinamentos sucessivos, na qual parte-se de uma abordagem bastante abstrata até uma especificação próxima a implementação.

Outro fator é que Estelle é um modelo formal híbrido, que combina as vantagens dos modelos de transição de estados e das linguagens de programação.

Por último, outro fator decisivo para a escolha de Estelle foi a existência e a disponibilidade de ferramentas automatizadas que auxiliassem a realização da metodologia escolhida.

O LCMI possui três ferramentas de suporte a Estelle : o ESTIM, o EDT e o ECHIDNA. Devido as características de Estelle suportadas por cada uma, apenas as duas primeiras foram escolhidas.

Uma descrição mais detalhada de Estelle e de Estelle*[1] é realizada no anexo I. O ESTIM e o EDT são descritos no anexo II.

4.2 - Metodologia

A metodologia adotada se baseia em uma metodologia de refinamentos sucessivos fundamentada na Técnica de Descrição Formal Estelle definida em [MAZZ91]. Esta metodologia também enfatiza a importância dos processos de validação das especificações a cada refinamento. Algumas mudanças foram feitas na metodologia, principalmente no que tange a parte da especificação orientada a implementação, onde adotou-se a metodologia apresentada em [SAQU92]. Os vários passos da metodologia escolhida serão descritos a seguir.

4.2.1 - Definição do Problema

Inicialmente deve-se levantar os requisitos e identificar os aspectos que precisam ser tratados.

[1] Estelle* é um dialeto de Estelle que inclui um mecanismo de rendez-vouz aumentando o poder de abstrações na especificação.

- Verificar os serviços que se pretende suprir para a aplicação desejada.
- Definir o conjunto de serviços básicos necessários para a aplicação (conjunto mínimo).
- Definir informalmente a especificação mínima necessária para a realização dos serviços definidos anteriormente, isto é, definir quais serão os serviços, PDUs, subcamadas, etc, que serão utilizados.

4.2.2 - Determinação da Arquitetura

Neste ponto define-se quais módulos irão compor a arquitetura do sistema. Além disto, quais os relacionamentos entre os módulos, isto é, pai-filho ou irmão. Determinam-se também os canais de comunicação, que podem existir, entre os módulos.

Neste nível, é realizada a diagramação em blocos de Estelle, pois facilita a visualização da arquitetura definida. Esta diagramação deve ser refinada sucessivamente até chegar-se a uma representação da estrutura no nível de abstração desejado.

4.2.3 - Especificação Funcional [MAZZ91]

A especificação funcional permite descrever a organização do software em termos de módulos e submódulos, etc. Nesta etapa, se fará a representação em Estelle da diagramação em blocos definida anteriormente.

Na especificação funcional é construído um "framework" da especificação. Este conterá apenas as declarações de canais, de cabeçalhos de módulos e os corpos destes módulos. Sendo que estes últimos poderão estar vazios. É possível também representar a decomposição em submódulos, sendo que todas as regras anteriores se aplicam durante esta decomposição. Por último podem haver as declarações de variáveis módulos, instanciação destes, bem como o estabelecimento das ligações entre os módulos.

4.2.4 - Definição dos Comportamentos

Este passo define o comportamento dos módulos. Neste ponto foi associado um significado às operações. Pode-se também, definir a ordem em que as operações serão realizadas.

Para o caso da especificação de protocolos de comunicação, se o comportamento é simples , deve-se fazer a Máquina de Estados Finitos (MEF), que representa o protocolo.

Caso o comportamento do sistema seja complexo [BOCH90], e um processo de refinamento possa ser aplicado, deve-se considerar o processamento como uma "aplicação" que pode ser especificada repetindo o processo de projeto a partir da fase de diagramação até a fase de definição do comportamento, até que todos os componentes estejam completamente especificados.

4.2.5 - Especificação Formal [MAZZ91]

4.2.5.1 - Especificação Orientada ao Modelo

A especificação orientada ao modelo permite representar os aspectos essenciais do funcionamento do software. Ela é o primeiro refinamento da especificação funcional, e difere desta principalmente pela definição dos corpos dos módulos, que estavam vazios ou só com as partes de declaração e inicialização.

Neste nível indica-se os tipos de interações trocadas em cada canal, isto é, os serviços definidos e mapeados aos módulos anteriormente. Porém, os parâmetros destes podem ser irrelevantes. Outra parte essencial neste nível de especificação é a declaração da parte das transições dos módulos, na qual representam-se os comportamentos que foram determinados em uma fase prévia. Porém, o uso de instruções PASCAL, procedimentos e funções deve ser restrito.

4.2.5.2 - Especificação Detalhada

Neste nível pode-se representar o funcionamento completo do software, dentro de uma óptica de implementação. Insere-se a parte seqüencial, isto é, as instruções, procedimentos e funções. Neste nível também são definidos os tipos e variáveis PASCAL a serem utilizadas. Aqui boa parte do não determinismo é resolvido com a introdução de novas cláusulas nas transições existentes. [MAZZ91].

4.2.5.3 - Especificação Orientada a Implementação

No momento da especificação orientada a implementação já se conhece qual será a organização física do software. Logo é necessário alterar a organização dos módulos da especificação e obviamente os atributos associados a estes. Deverá haver uma distinção entre módulos que executarão em máquinas diferentes.

Durante esta fase pode-se introduzir a metodologia proposta em [SAQU92]. Esta metodologia funciona em um ambiente Unix e TCP/IP, utilizando essencialmente o conceito de "sockets" [COME91] [SUN90]. Ela propõe que a partir de uma especificação Estelle sejam feitas algumas alterações e, com o auxílio de ferramentas, gere-se o código desejado.

Uma especificação Estelle caracteriza-se por ser um mundo fechado. Do ponto de vista da especificação de protocolos, isto quer dizer que este mundo fechado compreende a camada de protocolo alvo da especificação (camada N), e sua camada superior (camada N+1) e uma abstração das camadas inferiores (camada N - 1). Isto é representado na Figura 4-1.

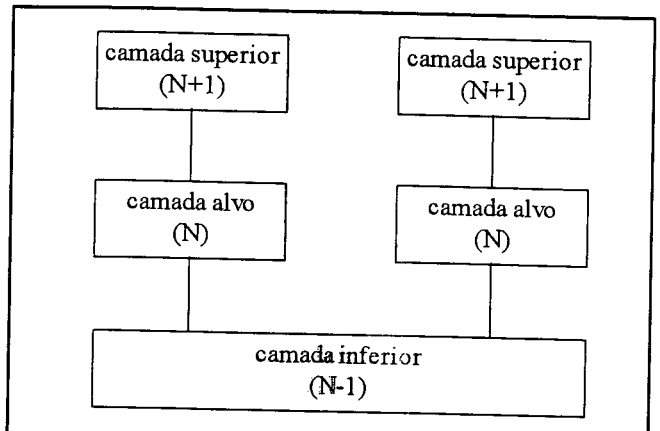


Figura 4-1- Arquitetura "Fechada" de uma Especificação Estelle

A metodologia sugere que sejam feitas transformações sucessivas na arquitetura supracitada, de modo a "abrir" o mundo da especificação. As figuras a seguir (Figura 4-2 e Figura 4-3) representam as transformações realizadas na especificação Estelle.

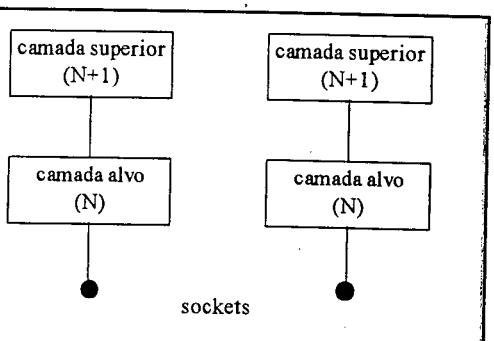


Figura 4-2 - Substituição das Camadas Inferiores por "Sockets"

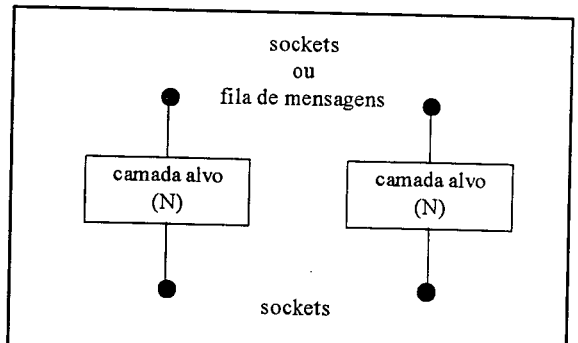


Figura 4-3 - Substiuição da Camada Utilizadora por "Sockets" ou Filas de Mensagens

Para a realização destas transformações sucessivas é necessário realizar algumas alterações na especificação Estelle. Para não afetar a especificação da camada alvo e para poder

permitir a comunicação com o mundo externo da especificação, realiza-se algumas alterações na especificação principal. Deve-se introduzir na especificação principal pontos de interação internos com papel oposto dos pontos de interação da camada alvo (N). Estes pontos de interação substituirão as funções dos pontos de interação das camadas N+1 e N-1. Juntamente deve-se colocar a comunicação com o mundo exterior através de "sockets". A Figura 4-4 mostra o resultado final das transformações realizadas sobre a especificação Estelle.

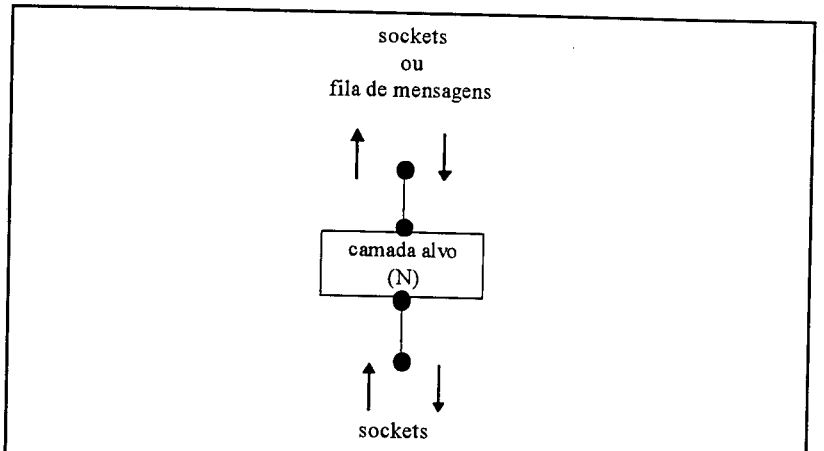


Figura 4-4 - Resultado das transformações realizadas na especificação Estelle

4.2.6 - Validação

Entende-se por validação todos os procedimentos realizados para certificar-se que uma especificação ou implementação está correta. A cada fase da concepção do software existe uma abordagem mais apropriada para a validação desta.

4.2.6.1 - Análise Estrutural Estática

Este primeiro nível de validação se aplica à especificação funcional, e consiste em verificar se todos os mecanismos de estruturação são utilizados de maneira apropriada e coerente. Ela permite eliminar unicamente os erros de estruturação estática da especificação[MAZZ91].

4.2.6.2 - Verificação por Abstração

Um método clássico de verificação de um sistema de transições consiste em construir o grafo de estados estáveis (grafo de acessibilidade) da especificação e a partir deste grafo deduzir as propriedades interessantes do modelo [MAZZ91].

O método de verificação por abstração, consiste em aplicar sobre o grafo de acessibilidade obtido as técnicas de redução baseadas nas relações de equivalência. O resultado da verificação é um autômato quociente[2], que dá uma visão abstrata do modelo. A escolha de uma boa relação de equivalência é determinante, o grafo reduzido deve ter poucos estados a serem explorados, e por outro lado, deve conservar certas propriedades interessantes do grafo original [COUR91]. A exemplificação de autômatos podem ser vistas na figuras a seguir (Figura 4-5 e Figura 4-6)

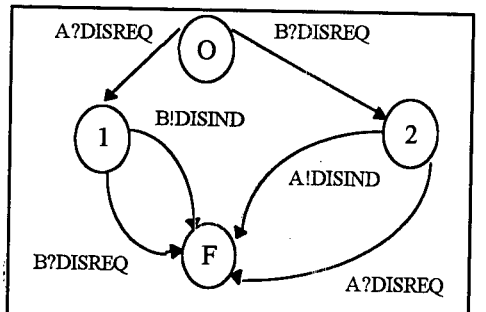


Figura 4-5 - Exemplo de Autômato de Serviço para a fase de Desconexão do protocolo Abracadabra [COUR92]

[2] Autômato Quociente é o autômato resultante da aplicação de técnicas de redução baseadas nas relações de equivalência sobre o grafo de acessibilidade

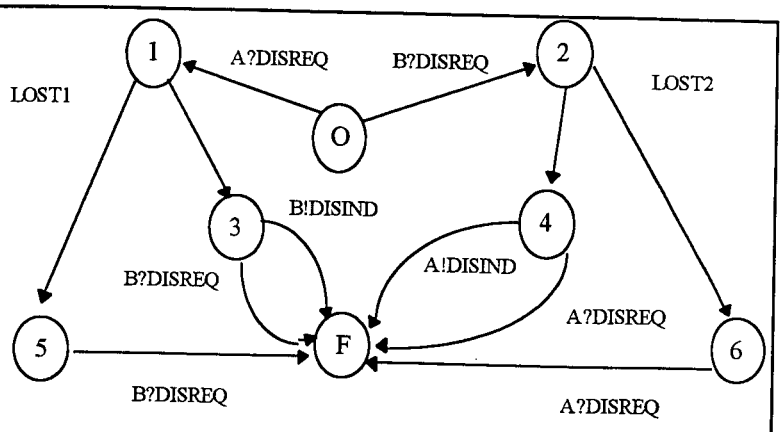


Figura 4-6 - Exemplo de Autômato Quociente pela aplicação da Equivalência Observacional para a fase de Desconexão do protocolo Abracadabra [COUR92]

A verificação por abstração é mais adequada para a especificação orientada ao modelo. Isto porque, se a aplicarmos a especificação detalhada, podemos ter o problema de explosão combinatória de estados. Além disto, a especificação orientada ao modelo já representa o funcionamento essencial do software.

As relações de equivalência existentes são :

- forte;
- observacional;
- teste;
- traço ou linguagem.

As definições destas podem ser vistas no Anexo III.

Partições

Neste nível é necessário realizar uma partição entre o *mundo observado* e o *ambiente*, sendo que os *eventos observáveis* são as trocas ocorridas entre estes dois mundos, sendo os outros eventos denominados de *eventos internos*.

Quanto aos tipos de partição, existem três tipos relevantes :

1 - os dois usuários do serviço possuem o mesmo papel, e fazem parte do mundo externo. Este tipo de partição permite obter a especificação do serviço global realizado por um protocolo de comunicação;

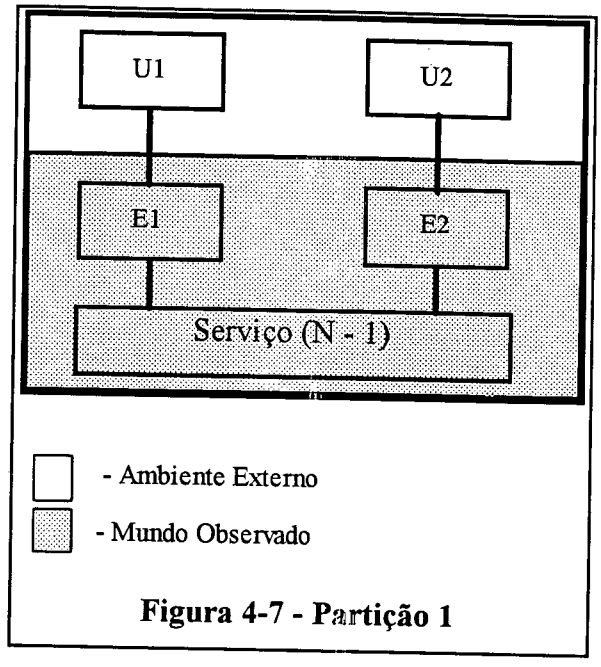


Figura 4-7 - Partição 1

2 - os dois usuários possuem papéis diferentes, onde um é o cliente e o outro é o servidor. Onde apenas o módulo cliente faz parte do mundo externo.

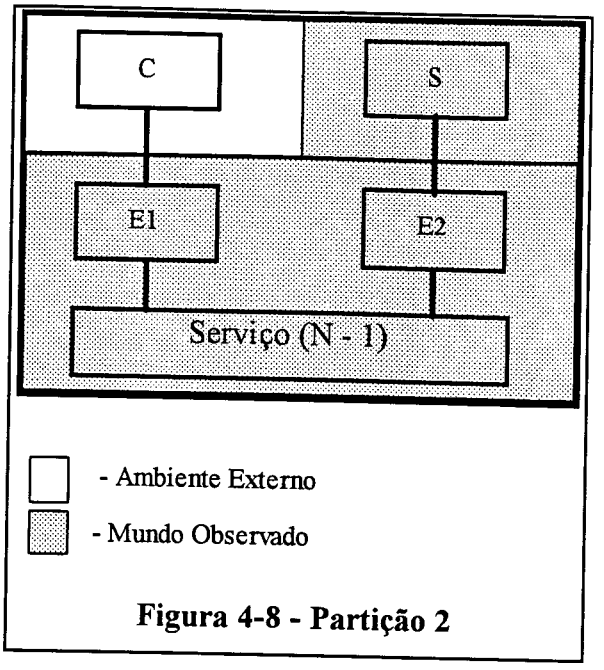


Figura 4-8 - Partição 2

3 - mais de dois usuários possuem papéis assimétricos. Esta partição pode ser vista como uma generalização da anterior. Aqui também, apenas os clientes fazem parte do mundo externo [COUR91].

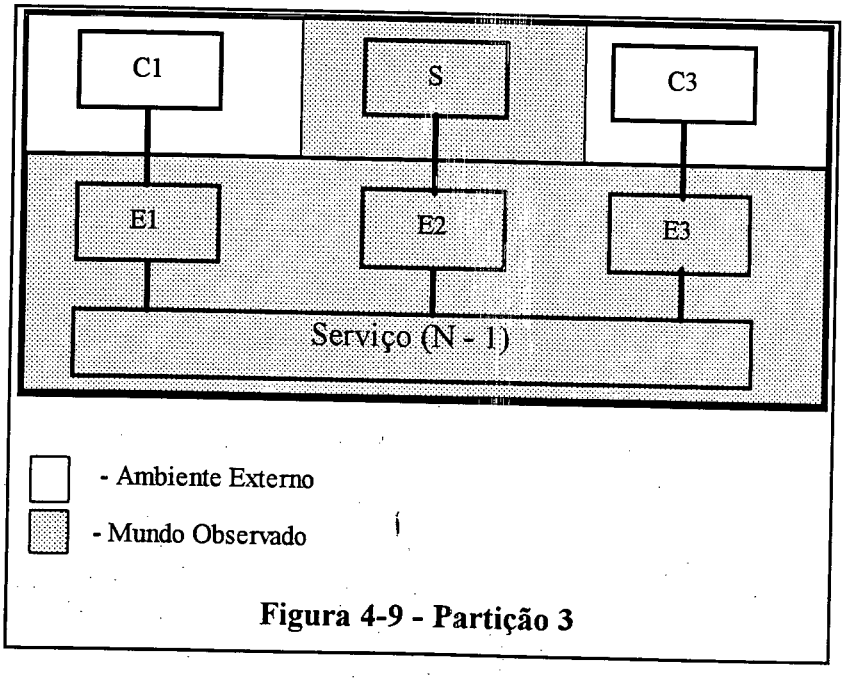


Figura 4-9 - Partição 3

4.2.6.3 - Simulação

A simulação é uma abordagem que pode ser utilizada na especificação orientada ao modelo, na detalhada, e na orientada à implementação.

A simulação primeiramente é um complemento da verificação. Ela permite seguir caminhos do grafo de acessibilidade, possibilitando analisar de maneira interativa os diferentes objetos da especificação, de modo a conhecer as causas do erros.

A simulação também permite validar a integração dos novos elementos introduzidos na especificação detalhada e de garantir a coerência do comportamento desta especificação em relação a especificação orientada ao modelo.

Quanto mais intensa for a simulação, mais eficaz ela será. Porém, normalmente a simulação não é exaustiva.

Na simulação sugere-se o uso da técnica de busca focalizada [BOCH80], na qual determina-se estados globais potenciais com certas propriedades e verifica-se se estes são alcançáveis.

4.2.7 - Implementação

Aqui pode-se optar por uma geração semi-automática de código ou manual. Na implementação semi-automática, as partes independentes de máquina são geradas automaticamente a partir da especificação formal, enquanto que o restante é codificado manualmente. Este tipo de implementação apresenta as seguintes vantagens [SIDH90]:

- conformidade da porção de código gerada automaticamente com o padrão do protocolo;
- facilidade de manutenção;
- maior confiabilidade;

- redução considerável do custo de implementação.
- o tamanho e a eficiência do código gerado é comparável com as implementações manuais em uma linguagem de alto nível.

4.2.8 - Teste de Implementação

O objetivo deste teste é verificar se todas as etapas anteriores foram bem efetuadas e de maneira eficaz e que o sistema atende as necessidades do usuário.

Uma abordagem clássica para testar as implementações dos protocolos OSI permite verificar que elas estão conforme sua especificação [RAFI90], [LINN89], [RAYN87].

No teste de conformidade a implementação do protocolo, chamada de "implementação sob teste" (implementation under test - IUT), é checada contra a especificação do protocolo, que serve como referência. A IUT é estimulada por entradas de testes emitidas por módulos testadores. A saída é comparada com a especificação para determinar se esta está de acordo.

A realização de um teste sem erros não indica que a IUT está totalmente conforme, visto que os testes não são exaustivos e que um erro pode ser encontrado por outro tipo de teste.

A seleção de um conjunto de testes apropriado é um aspecto importante do teste de conformidade. Vale ainda ressaltar que a geração de seqüências de testes pode em certos casos ser realizada de forma automática a partir da especificação formal.

4.2.9 - Resumo da Metodologia

A metodologia adotada pode ser vista na Figura 4-10.

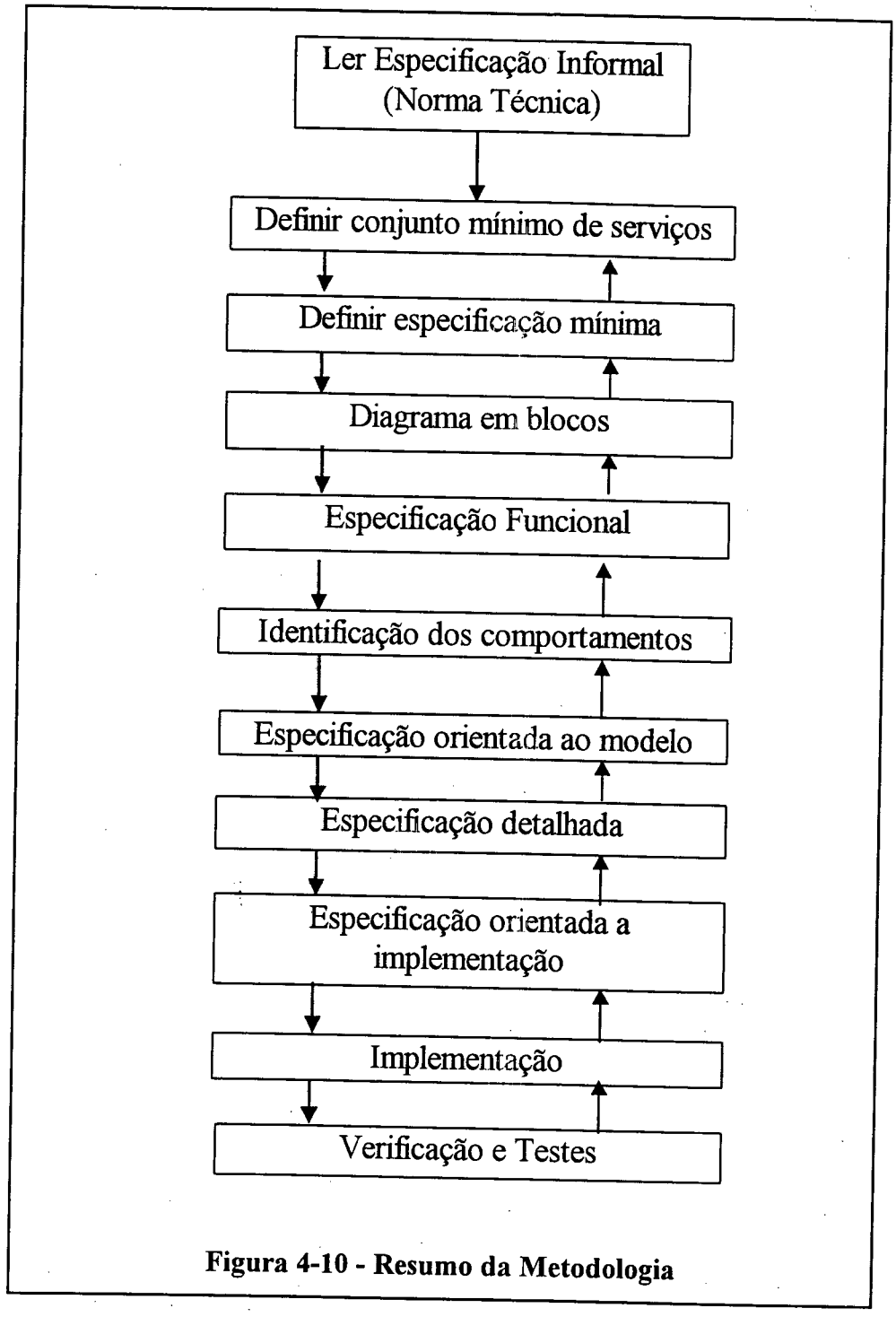


Figura 4-10 - Resumo da Metodologia

4.3 - Conclusão

Neste capítulo foi apresentada a metodologia adotada, bem como as justificativas para a escolha desta. Acredita-se que, sem a adoção de uma metodologia não é possível o desenvolvimento de sistemas distribuídos com uma mínima garantia de confiabilidade e correção. Acredita-se também que o uso de técnicas de descrição formal mellhora os resultados obtidos consideravelmente, bem como diminui o tempo para que estes sejam atingidos. É importante que no desenrolar de cada etapa da metodologia haja uma validação dos resultados desta, para que os erros sejam detectados o mais breve possível e que se tenha uma margem de confiabilidade dos resultados obtidos. Por fim, deve-se ter em mente que os softwares possuem uma característica que é a necessidade constante de evolução, logo é imperativo que uma metodologia adotada não desconsidere a fase de manutenção.

O capítulo seguinte mostra a aplicação da metodologia apresentada neste capítulo no desenvolvimento de uma arquitetura de comunicação SP50.

Capítulo 5

Um Ambiente de Simulação para Testes de Arquiteturas Distribuídas de Controle de Processos: Implementação e Experimentos

5.1 - Introdução

Este capítulo mostra a implementação e os primeiros experimentos de um ambiente de simulação para testes de arquiteturas distribuídas de processos. Este ambiente é o resultado da interconexão de um ambiente de simulação de controle de processos, o SADECA [KAMM92], desenvolvido no LCMI-UFSC e descrito sucintamente a seguir, com uma arquitetura de comunicação SP50. Esta arquitetura de comunicação, bem como a interconexão, foi especificada seguindo a metodologia descrita no Capítulo 4.

Um dos objetivos desta interconexão consiste em construir e experimentar um ambiente que simule uma aplicação de controle de processos, comunicando-se através de uma arquitetura SP50.

Num primeiro tempo, apresenta-se a arquitetura do ambiente de simulação, composto do simulador de processos SADECA já existente e do simulador da arquitetura de comunicação SP50. Descreve-se em detalhes a parte correspondente a comunicação, destacando-se a metodologia adotada, que prevê a validação do protocolo SP50 de forma a alcançar o nível de correção exigido do protocolo especificado.

A seguir apresenta-se a utilização deste ambiente para simular uma aplicação clássica da área de controle de processos. Por fim sugere-se uma metodologia para a utilização deste ambiente de simulação.

5.2 - Arquitetura do Ambiente de Simulação

O ambiente de simulação que será apresentado no decorrer deste capítulo é composto basicamente por uma simulação do processo realizado pelo SADECA, uma simulação da arquitetura de comunicação SP50 e uma simulação do controlador do processo. A simulação do processo realizada pelo SADECA interage com a simulação da arquitetura de comunicação. Nesta arquitetura estão representados os sensores, atuadores e o controlador do processo. Todos os elementos componentes da arquitetura de comunicação são compostos de uma camada de aplicação, camada de enlace de dados e camada física. As camadas de aplicação são representadas por programas em linguagem C, as camadas de enlace de dados são especificações Estelle, transformadas em linguagem C através da ferramenta EC e a camada física é representada por comunicação via "sockets". A comunicação entre a camada de aplicação e sua camada de enlace correspondente também é realizada via "sockets". As estações representando o sensor, atuador e controlador são estações do tipo DLE Comum, além destas existe uma estação especial que é do tipo LAS que realiza o escalonamento global para acesso ao meio de comunicação. No decorrer deste capítulo serão descritos mais detalhadamente os passos utilizados na construção do ambiente de simulação.

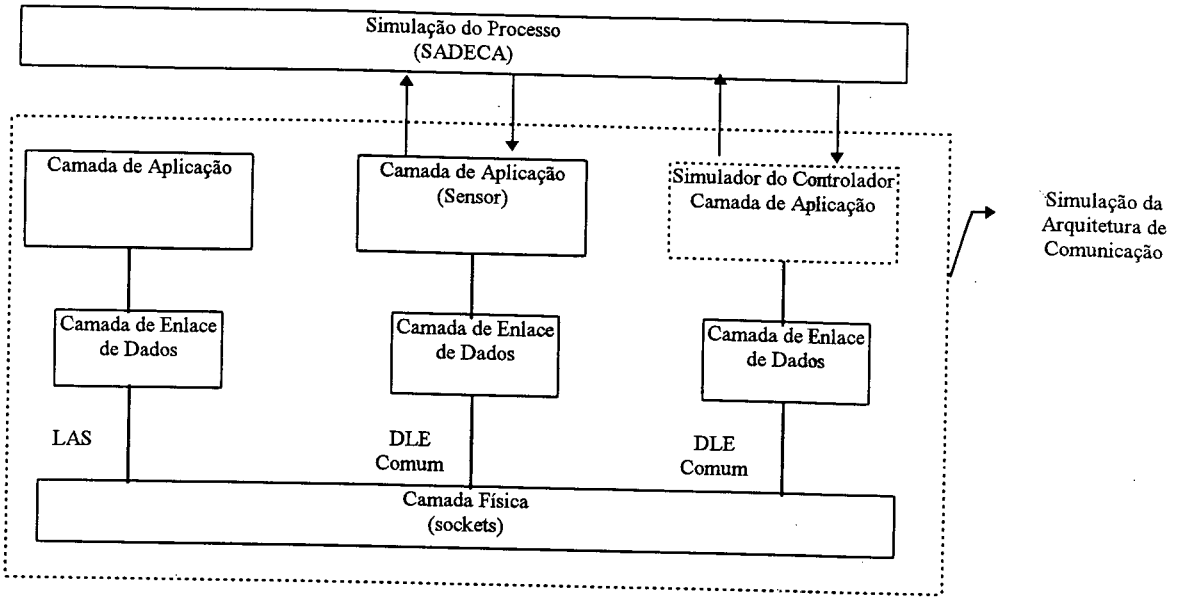


Figura 5-1 - Arquitetura do Ambiente de Simulação

5.2.1 - Simulação do Processo

O SADECA (Sistema para Avaliação de Desempenho de Controladores Adaptativos) é um ambiente de simulação computacional para a realização de avaliações de comportamento de diferentes estruturas de controle adaptativo [KAMM94] . Os controladores adaptativos podem ser testados sobre uma ampla gama de processos monovariáveis cuja dinâmica varia no tempo. Ele foi desenvolvido no LCMI-UFSC em linguagem C++ sobre ambiente UNIX, com interfaces gráficas implementadas com o pacote gráfico XView.

O SADECA possui três modos de operação : declaração, simulação manual e simulação automática. No modo declaração pode-se declarar tanto o controlador quanto o processo. Na declaração do controlador escolhe-se o tipo de estrutura a ser utilizada, bem como define-se alguns valores necessários para a fase de simulação. Já na declaração do processo define-se um modelo composto por até dez blocos em cascata, sem realimentações internas. Este blocos podem ser de cinco tipos diferentes : dinâmica linear contínua (DLC), dinâmica linear discreta (DLD), atraso de transporte (ATR), ganho não linear (GNL) e entrada de perturbação (PRT). Na fase de

simulação manual o usuário define e/ou altera alguns parâmetros e simula utilizando botões similares ao de um toca-fitas. Os resultados da simulação são observados em dois gráficos : o gráfico Y que mostra a sinal de saída do processo e o sinal de referência e o gráfico U que mostra o sinal de controle. A simulação automática consiste na execução de seqüências de eventos definidas em arquivos. Os gráficos observados são os mesmos da simulação manual. Um apresentação completa do SADECA pode ser vista em [KAMM92] e [KAMM94].

Para que o SADECA se comunique com programas externos, como por exemplo a simulação da arquitetura de comunicação definida neste trabalho, é necessário alterar o código fonte de um dos controladores existentes ou definir um novo controlador, de modo que este programa se comunique através de sockets com os programas externos. Em [KAMM94] são descritos todos os passos necessários para a introdução de novos controladores.

5.2.2 - Simulação da Arquitetura de Comunicação SP50

A simulação da arquitetura de comunicação do SP50 compreende duas fases: a fase de especificação, verificação e simulação de uma especificação fechada Estelle e fase de implementação e testes desta especificação.

Na primeira fase utilizou-se as ferramentas ESTIM e EDT. Nesta fase as três camadas são inteiramente especificadas em Estelle e simuladas nestes ambientes. Na segunda fase realiza-se a geração de código C a partir da especificação Estelle da camada de enlace de dados. A camada de aplicação é representada por programas escritos em linguagem C. A comunicação entre as camadas de aplicação e camada de enlace de dados, bem como a representação da camada física se dará via "sockets". Os comandos relativos aos "sockets" nos programas C são colocados diretamente no código fonte e na parte da especificação são embutidos como comentários de qualificação[1] que posteriormente serão convertidos em comandos em linguagem C pela ferramenta EC do EDT.

[1] A definição de comentários de qualificação pode ser vista no anexo II, que cobre as ferramentas Estelle utilizadas.

5.3 - A Camada de Enlace de Dados : Especificação Mínima para a Simulação

De acordo com o tipo de aplicação a ser atendida, deve-se definir uma configuração mínima da camada de enlace de dados do Field Bus SP50, que possibilita a realização dos serviços necessários.

5.3.1 - Tipos de Estações

No SP50 tem-se três tipos de entidades :

- LAS , LM e DLE Comum (descritas no Capítulo 2).

Nesta configuração, uma estação será do tipo LAS, porque o protocolo SP50 necessita de uma estação especial para o controle do escalonamento global de acesso ao meio de comunicação. As outras estações participantes serão do tipo DLE Comum.

5.3.2 - Tipos de Fichas

O Token (ou Ficha) é a única permissão para ser o iniciador de transações no enlace local. Esta permissão é assumida pela LAS DLE quando o enlace é criado. Esta permissão pode ser delegada a DLEs individuais, sujeito a restrições no seu uso.

Os tipos de fichas existentes no SP50 são :

- Ficha Escalonadora (Scheduler Token) ,
- Ficha Delegada (Delegated Token) ;
- Ficha de Resposta (Reply Token) .

Esta configuração vai apresentar a ficha escalonadora, que é necessária para que a LAS possa escalonar as tarefas a serem executadas, e a ficha delegada, que dá o direito de acesso ao meio a seu detentor.

5.3.3 - Sub-Níveis

A camada de enlace de dados é modelada em 3 sub-níveis :

- Baixo Nível de funções de escalonamento e acesso aos caminhos (subnível 1);
- Nível Intermediário de funções e operação como ponte (subnível 2);
- Nível Superior de transferência de dados com e sem conexão, coordenação de pontes e funções de serviços de enlace de dados (subnível 3).

Nesta especificação mínima serão necessários o sub-nível 1 e parte do sub-nível 3, sendo que a parte retirada do sub-nível 3, é a referente as pontes e transferências com conexão.

5.3.4 - Classes Funcionais

Existem 3 classes funcionais de DLE e estas determinam as capacidades para a atividade DLL autônoma, e a complexidade mínima para implementação, sendo que as classes superiores incluem as inferiores. As 3 classes em ordem crescente de complexidade são :

- Básica;
- Mestre de Enlace (LM) ;
- Ponte.

Todas as classes suportam todos os serviços de usuário DLS e são completamente interoperacionais. Na especificação proposta, tem-se somente necessidade das duas primeiras classes.

5.3.5 - Qualidade de Serviços

O usuário DLS pode selecionar vários aspectos dos serviços de enlace de dados. Os aspectos que estão sob o controle direto do provedor DLS são chamados de qualidade de serviço(Quality of Service - QoS). Os atributos QoS podem ser:

- dinâmicos, isto é, são selecionados independentemente para cada instância de serviço;
- estáticos, isto é, são selecionados uma só vez para uma classe inteira de serviços.

Os atributos QoS são descritos a seguir.

5.3.5.1 - Prioridade DLL

Cada serviço sem conexão e cada pedido e resposta de estabelecimento de conexão, especifica uma prioridade DLL, que é usada no escalonamento dos serviços de transferência de dados. Esta prioridade também determina a quantidade máxima de dados que pode ser carregada em uma DLPDU. As prioridades e respectivas quantidades são :

- Urgente : <= 64 bytes de dados de usuário por DLPDU;
- Normal : <= 128 bytes;
- Tempo Disponível : <= 256 bytes.

Devido às características de tempo real do sistema, as três possibilidades serão utilizadas.

5.3.5.2 - Autenticação da DLPDU

Existem duas possibilidades :

- Ordinary : cada DLPDU inclui a quantidade mínima necessária de informação de endereçamento;
- Extra : cada endereço DL inclui a quantidade máxima possível de informação de endereçamento.

Nesta especificação não será considerado este item.

5.3.5.3 - Atraso Máximo de Confirmação

É o tempo máximo para completar certos tipos de serviços. Este atributo pode possuir o valor ILIMITADO ou um intervalo em unidades de ms, de 1 ms até 1 minuto inclusive. Nesta especificação serão usados vários deste valores para teste de como o protocolo se comporta com cada um.

5.3.5.4 - Política de Escalonamento DL

Para cada endereço DLSAP, e cada DLCEP, o usuário DLS pode sobrepor a política de escalonamento normal (implicitamente escalonada).

Existem duas possibilidades :

- Implícita : qualquer comunicação requerida com usuário(s) DLS par deste endereço DLSAP ou deste DLCEP, ocorrerá logo que possível;
- Explícita : qualquer comunicação requerida com usuário(s) DLS par deste endereço DLSAP ou deste DLCEP, ocorrerá somente quando a causa do adiamento é explicitamente liberada por um usuário DLS envolvido.

Ressaltamos que nesta configuração será usada somente a alternativa implícita.

5.3.5.5 - Tamanho Máximo da DLSDU

Este atributo especifica o limite máximo para o tamanho da DLSDU. O valor é dependente do suportado pelos participantes da comunicação e pelos gerenciamentos associados a estes, sendo que nesta especificação mínima será utilizado o valor default que é o tamanho igual ao de uma DLPDU.

5.3.5.6 - Ligações de buffer e filas a DLSAP

A Figura 5-2 mostra mecanismos utilizados na transmissão e recepção de DLSDUs, destes o SP50 utiliza os buffers e as filas explícitas, com as seguintes características :

- um buffer persistente, cujo conteúdo não é afetado quando é lido, e que o valor existente é sobreposto quando acontece uma escrita;
- uma fila de tamanho máximo K, cujo conteúdo pode ser de 0 a K DLSDUs, e que rejeitará tentativas de leitura quando estiver vazia e de escrita quando estiver cheia.

Nesta alternativa serão utilizadas as filas explícitas.

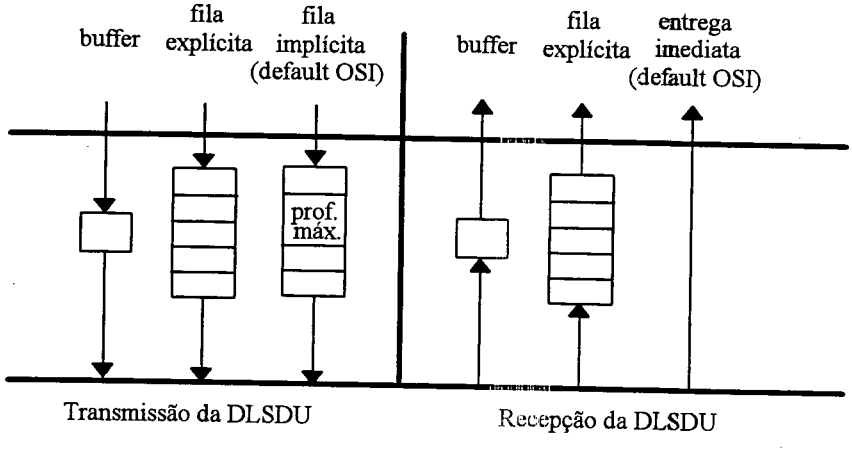


Figura 5-2- Tipos de Mecanismos Utilizados na Transmissão e Recepção de DLSDUs

5.3.6 - Serviços

Existem quatro tipos de serviços de enlace de dados :

- serviços de gerenciamento de buffers, filas e endereços DL(SAP);
- serviços de transferência de dados com conexão com quatro classes de serviços;
- serviços de transferência dados sem conexão;
- serviços de escalonamento de transações e tempo com pelo menos três classes de serviço.

Destas classes de serviços, foram escolhidos apenas alguns da classe gerenciamento de buffers, filas e endereços DL(SAP) e alguns serviços de transferência de dados sem conexão. Isto porque com estes serviços tem-se uma abordagem simplificada, mas que ao mesmo tempo atende às necessidades das aplicações visadas.

5.3.6.1 - Serviços de gerência de buffer, filas e endereços DLSAP

Desta subclasse de serviços serão necessários alguns serviços, pois estes irão liberar o usuário da camada de enlace do problema de gerenciamento das filas. Estes serviços são locais e só possuem a primitiva de requisição de serviço (request), porém esta inclui a possibilidade de troca de parâmetros. Os serviços necessários são os seguintes :

- Create : Permite criar um buffer ou uma fila de tamanho limitado para uma futura associação a um DLSAP.
- Delete : Permite destruir uma fila ou buffer.
- Bind : Possibilita conectar um buffer ou fila previamente criados, em uma direção da transferência de dados, a um único DLC ou a um identificador de endereço DLSAP especificado com um certo nível de prioridade.
- Unbind : Desconecta um buffer ou fila de um DLC ou de um endereço DLSAP.
- Get : Permite recuperar o conteúdo de um buffer ou o primeiro elemento de uma fila

5.3.6.2 - Transferência de Dados Sem Conexão

Desta classe será usado o serviço DL-UnitData, que permite a transferência de dados de tamanho limitado de um DLSAP a outro em um único serviço, sem estabelecimento de conexão. Este serviço compreende três primitivas :

- Unit Data Request (UDR)
- Unit Data Confirm (UDC)
- Unit Data Indication (UDI)

5.3.6.3 - Máquinas de Estados dos Serviços

Em [IEC93a] não há a definição de uma máquina de estados para os **Serviços de Gerência de DLSAP Fila ou Buffer**, porém é apresentada a Figura 5-3 - Seqüência das Primitivas para os Serviços de Gerência de DLSAP Fila ou Buffer. A partir desta nós podemos deduzir uma máquina de estados para estes serviços. Associando esta máquina de estados à máquina de estados dos **Serviços Sem Conexão** (Figura 5-4) tem-se como resultado a máquina de estados dos serviços da Figura 5-5.

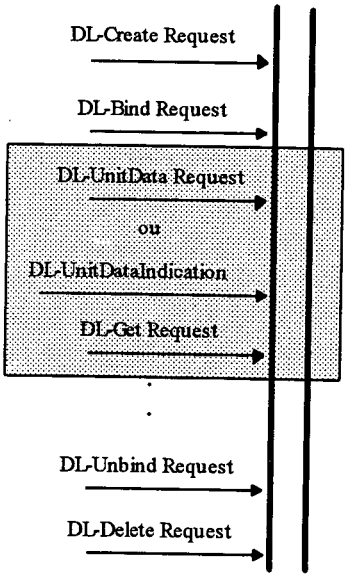


Figura 5-3 - Seqüência das Primitivas para os Serviços de Gerência de DLSAP Fila ou Buffer [IEC93a]

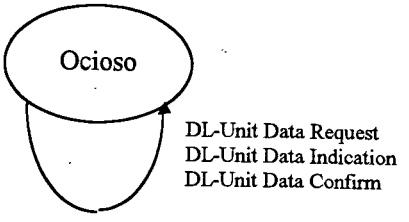


Figura 5-4 -Máquina de Estado de Serviços Sem Conexão[IEC93a]

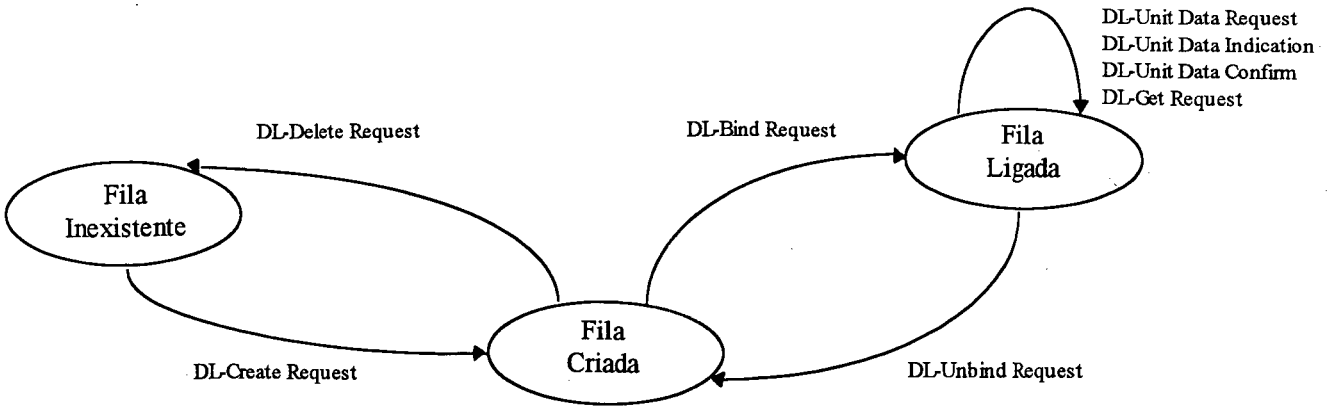


Figura 5-5 - Máquina de Estados dos Serviços

5.3.7 - Data Link Protocol Unit - DLPDUs

As DLPDUs necessárias para suporte dos serviços e funções descritos anteriormente são :

- DU-DLPDU - Data/UnitData DLPDU - é usada para transferência de uma quantidade limitada de dados de um usuário requisitante para um ou mais usuários DLS; ou para reconhecer a transferência deste tipo de dado dentro do contexto de uma DLC.
- ET-DLPDU - Execute Transaction DLPDU - é usada para passar a ficha delegada da LAS DLE para uma outra DLE no enlace local, que tenha previamente pedido ou escalonado uma transação, ou para a qual esta transação tenha sido configurada. É também usada pela LAS para verificar a existência de DLEs previamente não reconhecidas no enlace local. A ET DLPDU provê a DLE receptora o direito de iniciar uma única transação DL.
- EE-DLPDU - End Execution - é usada para devolver a ficha delegada para a LAS, e para pedir futuras delegações para uma transação, e também para pedir ações adicionais de escalonamento.

- RT-DLPDU - é usada para recuperar a ficha quando esta foi enviada para alguma estação e esta não a recebeu ou não devolveu corretamente.

Quanto a devolução da ficha delegada, vale ressaltar que ela também pode ser devolvida através das outras DLPDUs aqui citadas, bastando para isto colocar o valor FINAL no campo de controle de quadro da DLPDU. Porém definiu-se que a devolução somente será realizada através da EE DLPDU.

5.3.8 - Escalonamento

O escalonamento do SP50 consiste de um ciclo repetitivo de três classes de atividades :

- atividades síncronas no tempo;
- atividades de primeira oportunidade;
- atividades de investigação de endereços.

Nesta especificação serão utilizadas apenas as duas primeiras classes de atividades.

5.4 - A Simulação da Arquitetura de Comunicação SP50

5.4.1 - Diretivas para a Concepção da Simulação da Arquitetura de Comunicação SP50

A concepção do simulador da arquitetura de comunicação adotou a modularidade, de modo a facilitar a introdução de novos serviços ou grupos de serviços. A descrição dos serviços e protocolos definidos na norma técnica [IEC93a] [IEC93b], foram seguidos com a máxima fidelidade possível, retirando apenas os detalhes de menor importância.

Seguindo as diretrizes supracitadas, foram testadas várias arquiteturas, as quais foram validadas por meio de simulação, cada arquitetura nova sendo uma evolução da anterior, até atingir a simulação de uma arquitetura satisfatória seguindo os critérios estabelecidos.

5.4.2 - Descrição da Arquitetura de Comunicação SP50 Simulada

5.4.2.1 - Especificação Básica da Arquitetura de Comunicação SP50

A arquitetura de comunicação SP50 foi inicialmente descrita utilizando-se diagrama de blocos. A Figura 5-6 mostra o primeiro nível da especificação. Nela temos os macro componentes, que são:

- a camada de aplicação;
- a camada de enlace de dados, que é o alvo da especificação;
- e uma representação da camada física.

É bom ressaltar que dentre os módulos de camada de enlace de dados, já há uma diferenciação entre a LAS e a DLE Comum.

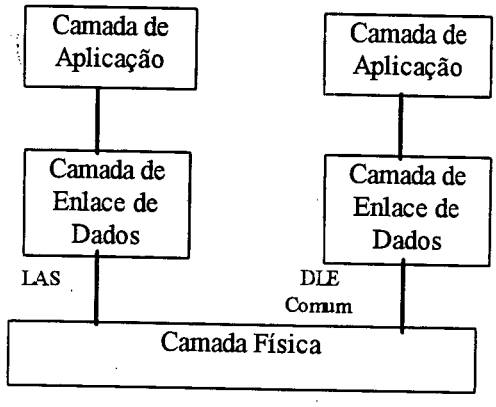


Figura 5-6 - Especificação do Simulador - Primeiro Nível

Sendo a camada de enlace o interesse maior na especificação que desejamos implementar, naturalmente esta sofreu vários refinamentos. O primeiro consiste da divisão da camada de enlace em dois níveis. Esta divisão da camada de enlace em dois sub-módulos (Alto Nível e Baixo Nível)

é justificada pelos subníveis definidos pela norma. Visto que não há a necessidade do subnível intermediário, devido a ausência de estações do tipo ponte no simulador, chegou-se a especificação que pode ser vista na Figura 5-7.

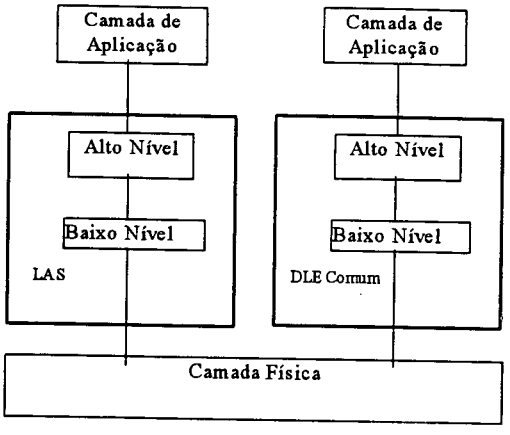


Figura 5-7 - Especificação do Simulador - Primeiro Refinamento

O módulo baixo nível do LAS necessitou ser dividido em dois módulos, isto ocorreu devido ao fato deste ser o responsável pelo escalonamento e a norma dividir o escalonamento em uma parte escalonadora e uma executora. A Figura 5-8 mostra este refinamento.

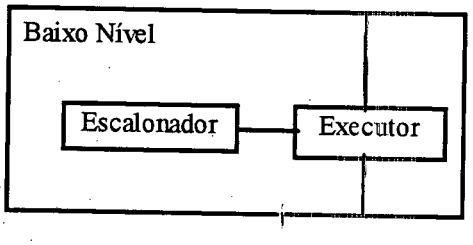


Figura 5-8 Módulo Baixo Nível da LAS

O módulo Baixo Nível da DLE Comum, bem como o módulo Executor da LAS, sofreram ainda uma outra subdivisão. Isto, devido ao fato de agruparem os serviços de acordo com sua categoria. As arquiteturas finais da DLE Comum e da LAS, resultantes deste último refinamento podem ser vistas nas figuras seguintes seguindo a representação clássica Estelle (Figura 5-9 e Figura 5-10)

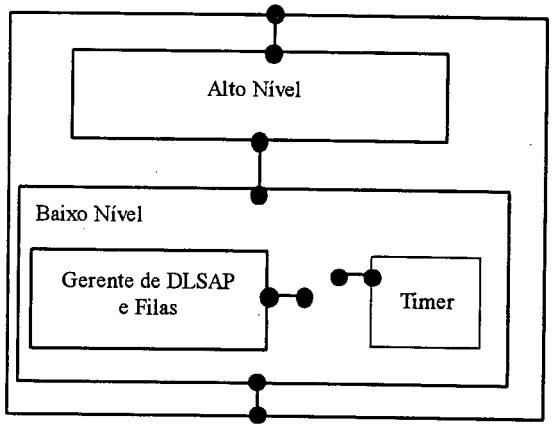


Figura 5-9 - Arquitetura Final da DLE Comum

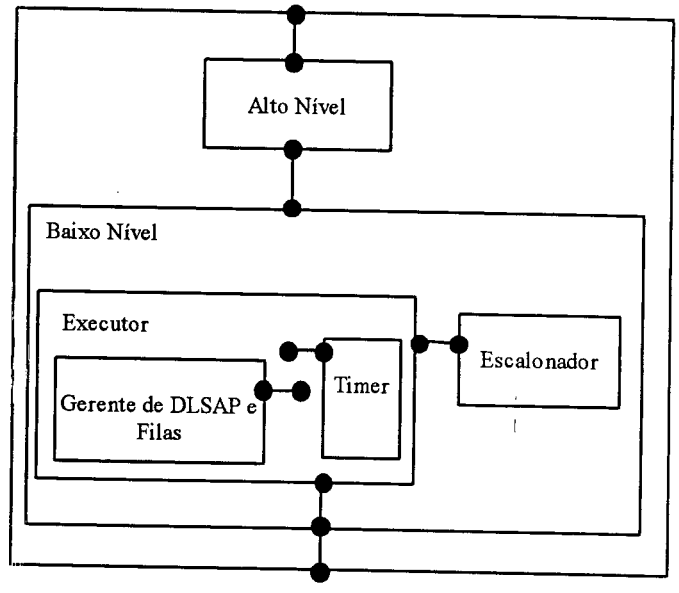


Figura 5-10 - Arquitetura Final da LAS

Os diagramas de blocos que descrevem a arquitetura foram especificados em Estelle, permitindo obter um arcabouço a ser refinado posteriormente.

A este nível, esta especificação funcional foi checada através do GENESTIM, ferramenta do ambiente ESTIM que permitiu verificar as estruturas léxicas, sintáticas e de semântica estática da especificação.

5.4.2.2 - Descrição dos Comportamentos na Arquitetura de Comunicação SP50

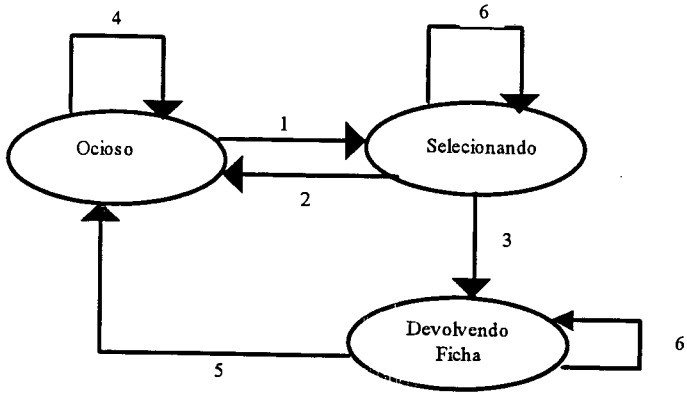
Na definição dos comportamentos optou-se por descrever as funções de cada módulo componente da arquitetura por meio de Máquinas de Estado Finito.

A - Módulo Alto Nível

O módulo Alto Nível é igual para os dois tipos de estações (LAS e DLE Comum). Ele é o responsável pelas interações com a camada de aplicação e mapeamento dos serviços nas DLPDUs necessárias. Ele também é responsável pela seqüenciamento das primitivas de enlace. Vale ressaltar que a máquina de estados para este módulo possui condições associadas as transições. A Figura 5-5 é representação desta máquina de estados, para uma instância de fila, entretanto a máquina de estados finita pode estar ocorrendo para mais de uma instância de fila paralelamente dentro do mesmo módulo, porém existem certas condições que controlam o funcionamento geral do módulo.

B - Módulo Baixo Nível

O módulo Baixo Nível é diferente nos dois tipos de estações. Na LAS ele subdivide-se nos submódulos Executor e Escalonador, sendo então suas funções desenvolvidas por estes submódulos. Já nas estações do tipo DLE Comum ele exerce as funções de controle de acesso ao meio e devolução da ficha. Além disto, o módulo Baixo Nível é o responsável pela montagem e desmontagem das DLPDUs. Na estação DLE Comum o módulo Baixo Nível possui o submódulo Gerente de DLSAP e Fila e o submódulo Timer. A Figura 5-11 mostra a máquina de estados do módulo Baixo Nível da DLE Comum.



1 - (? ET & ! Requisita Elemento Para Transmissão)

2 - (? Não Há Elemento Para Transmissão & ! EE) ou
(? RT)

3 - (? Elemento Para Transmissão & ! DU)

4 - (? UDR & ! Dispara Timer) ou
(? RT) ou
(? DU & ! UDI) ou
(? Time Out & ! UDC)

5 - (! EE & ! UDC) ou
(? RT)

6 - (? UDR & ! Dispara Timer) ou
(? DU & ! UDI) ou
(? Time Out & ! UDC)

Figura 5-11 - Máquina de Estados Módulo Baixo Nível da DLE Comum

B.1 - Módulo Gerente de DLSAP e Fila

Este módulo é responsável pelos serviços de gerenciamento de filas, bem como a associação destas aos DLSAPs. Isto é feito através dos serviços Create, Delete, Bind, Unbind e Get. Neste caso o procedimento é similar ao que ocorre para o módulo Alto Nível, no tocante a representação de uma instância de fila e controle geral do módulo. A máquina de estados correspondente a este módulo pode ser vista na Figura 5-12.

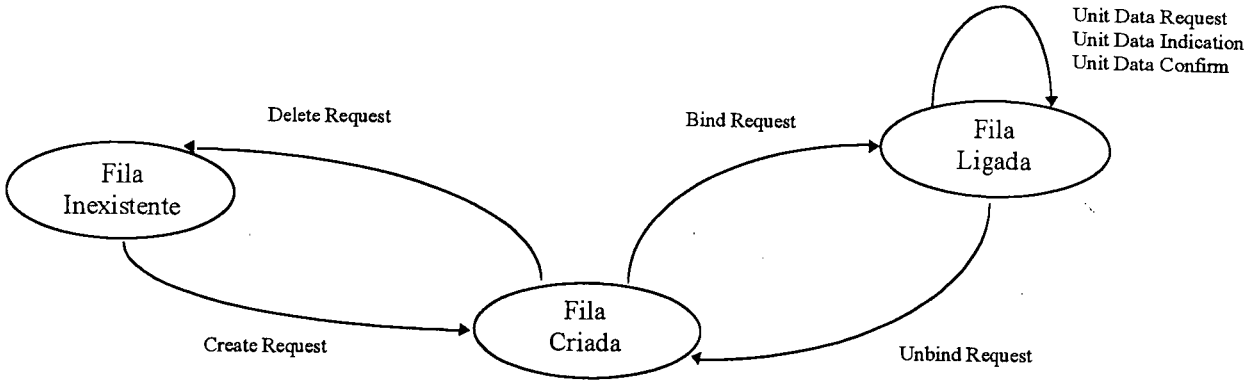


Figura 5-12 - Máquina de Estados do Módulo Gerente DLSAP e Fila

B.2 - Módulo Timer

O módulo Timer é o responsável pelo controle do tempo máximo de espera de uma DLPDU de dados (DU DLPDU) para ser transmitida, sendo que após este limite a DLPDU não será transmitida. A máquina de estados deste módulo pode ser vista na Figura 5-13.

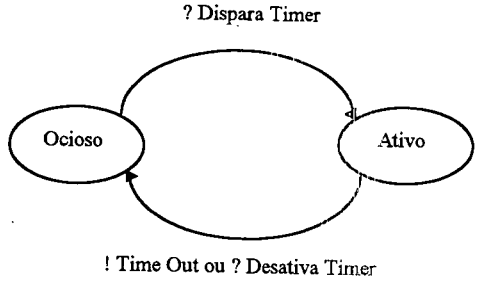


Figura 5-13 - Máquina de Estados do Módulo Timer

B.3 - Módulo Escalonador

Este módulo tem por objetivo realizar o escalonamento global no protocolo Field Bus. Este escalonamento é realizado a partir da tabela de escalonamento, que é previamente definida, e da fila de primeira oportunidade, construída dinamicamente durante a execução do

escalonamento. A máquina de estados que representa o módulo escalonador é apresentada na Figura 5-14.

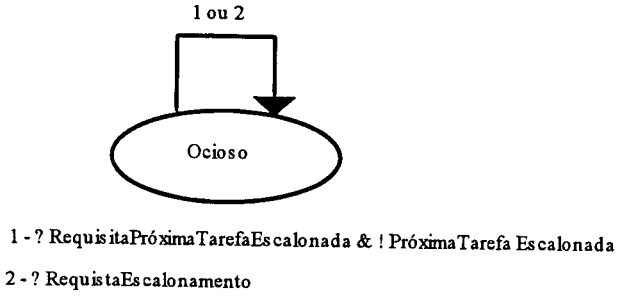
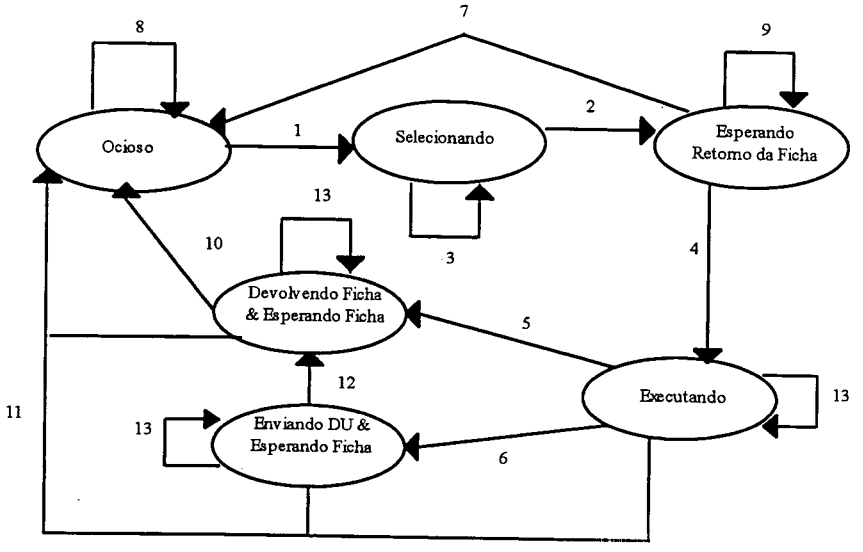


Figura 5-14 - Máquina de Estados do Módulo Escalonador (Baixo Nível da LAS)

B.4 - Módulo Executor

Este módulo realiza as mesmas funções do módulo Baixo Nível da DLE Comum, além disto ele exerce as funções de delegação e controle das fichas e executa o escalonamento de acordo com os dados passados pelo escalonador. Este módulo inclui o submódulo Gerente de DLSAP e Fila e o submódulo Timer. A máquina de estado representando este módulo é apresentada na Figura 5-15.



1 - Possui Ficha Escalonadora & Requisita Próxima Tarefa Escalonada

2 - ? Próxima Tarefa Escalonada & !ET

3 - ? ET

4 - ? ET* & Requisita Elemento Para Transmissão

5 - (? Não Há Elemento Para Transmissão & ! EE)

6 - ? Elemento Para Transmissão & ! DU

7 - (? EE) ou (! RT)

8 - (? RT & Emissor = Receptor) ou (? DU) ou (? UDR) ou (?ET & Possui Ficha Escalonadora) ou (? TimeOut & ! UDC)

9 - (? DU) ou (? UDR) ou (? TimeOut & ! UDC) ou (! UDI)

10 - ? EE

11 - (Estouro do Atraso Máximo Retorno Da Ficha & ! RT)

12 - (! EE & ! UDC)

13 - (? Time Out & ! UDC) ou (? RT & Emissor = Receptor) ou (? UDR)

* - Enviado pela própria estação

+ - Eh PDU Final

Figura 5-15 - Máquina de Estados do Módulo Executor (Baixo Nível da LAS)

C - Módulo Aplicação

Este é o módulo que utiliza os serviços de enlace de dados, sobre os quais ele baseia os serviços que vão formar a aplicação propriamente dita. Ele é o responsável por requisitar as primitivas de serviços da camada de enlace de dados. Na fase de especificação conterá todas as

primitivas de serviços que serão escolhidas no decorrer da simulação. Na fase de implementação será um programa que conterá as primitivas necessárias para o funcionamento do equipamento que estiver representando.

D - Módulo Camada Física

Este módulo é o repassador de DLPDUs, que recebe estas de uma entidade de enlace e a passa para outra, sem realizar nenhuma alteração. Neste módulo foi realizada uma simplificação, pois na verdade as DLPDUs são subdivididas em Unidades de Dado de Interface Física e então são transmitidas. Considerou-se que elas teriam o mesmo tamanho para facilitar o trabalho de especificação, nesta fase do trabalho.

5.4.3 - Especificação Formal para a Simulação da Arquitetura de Comunicação SP50

A especificação formal em ESTELLE utiliza todos os passos descritos no Capítulo 4, entretanto é necessário ressaltar alguns aspectos de cada nível de especificação, bem como os procedimentos de validação que deveriam ser utilizados a cada passo.

5.4.3.1 - Especificação Orientada ao Modelo

A simulação da arquitetura de comunicação SP50 inicia pela especificação orientada ao modelo e pela sua validação através da verificação e/ou simulação.

A - Características da Especificação Orientada ao Modelo

Neste nível de especificação, é utilizado Estelle*, um dialeto de Estelle que permite a comunicação por rendez-vous, o que torna a representação mais abstrata, sendo mais adequada para a representação da comunicação entre camadas adjacentes do modelo .

A arquitetura neste nível de especificação pode ser vista na Figura 5-16, onde pode-se ver que a comunicação entre a camada de aplicação e o módulo alto nível é realizado através de rendez-vous, enquanto que as outras comunicações se realizam por intermédio de filas FIFO.

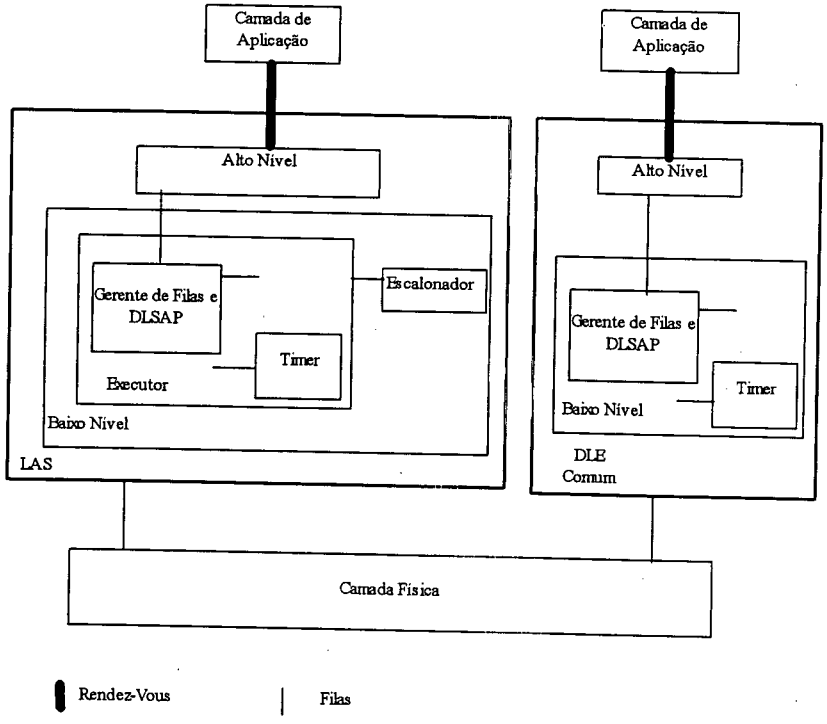


Figura 5-16 - Arquitetura da Especificação Orientada ao Modelo

A ferramenta a ser utilizada a este nível de especificação é o ESTIM, pois este suporta as construções de Estelle*. Além do mais, ele possui um ambiente de verificação, que foi utilizada aqui, pois a especificação orientada ao modelo já apresenta o funcionamento básico do protocolo, sendo menos sujeita ao problema de explosão combinatória do que a especificação detalhada.

B- Verificação da Especificação Orientada ao Modelo

Neste ponto da montagem da simulação da arquitetura, seguindo a metodologia proposta, é possível verificar a especificação.

A verificação como já foi exposto anteriormente, apresenta como grande obstáculo para sua realização o problema da explosão de estados do grafo de alcançabilidade. A sua utilização sobre a arquitetura apresentada na Figura 5-16 não é diferente. Optou-se então por realizar a verificação sobre diferentes fases do protocolo, separadamente. Abordagem similar é encontrado em [COUR92] no qual a verificação é realizada somente sobre a fase desconexão do protocolo ABRACADABRA [COUR92] [COUR88] [SAQU90b]. A ferramenta ESTIM permite inicialmente ao usuário escolher uma partição onde se define o ambiente e o mundo observado, o número máximo de estados gerados e um número de estados, que define que será gravado em arquivo o estado corrente do grafo de acessibilidade, a cada vez que este número for atingido.

Para exemplificar realizou-se a verificação sobre a fase de criação de filas e sua ligação aos DLSAPs da DLE Comum. Definiu-se uma partição na qual o módulo DLE Comum é visto como mundo observado e o resto como ambiente. A arquitetura representando esta fase pode ser vista na Figura 5-17.

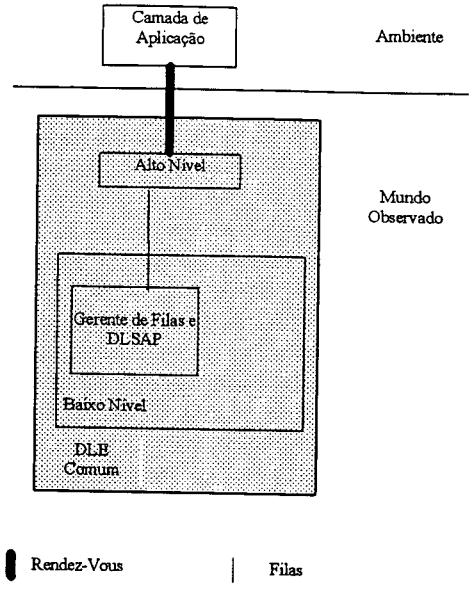
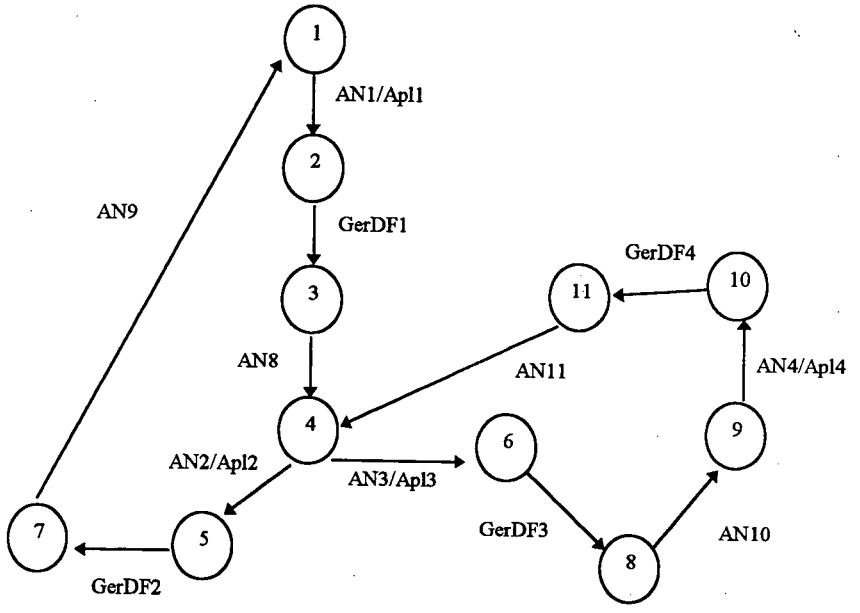


Figura 5-17 - Arquitetura para Verificação da Fase Criação e Ligação de Filas

. O ESTIM gera um grafo de acessibilidade, permitindo depois a realização das projeções através de ferramentas apropriadas. O grafo de acessibilidade desta fase pode ser visto na Figura 5-18 e as projeções de linguagem e observacional geradas podem ser vista na Figura 5-19. Vale ressaltar que neste caso as projeções de linguagem e observacional são iguais e que elas mostram que as reduções realizadas retiraram os eventos internos gerando um grafo que corresponde as seqüências das primitivas de serviços da máquinas de estados representando os serviços de Gerência de Filas e DLSAPs.



Eventos Externos (rendez-vous)	Eventos Internos (Filas FIFO)
AN1/Apl1 - Create Request	GerDF1 - ? Create Request_In & ! Create Request_Out
AN2/Apl2 - Delete Reuest	GerDF2 - ? Delete Request_In & ! Delete Request_Out
AN3/Apl3 - Bind Request	GerDF3 - ? Bind Request_In & ! Bind Request_Out
AN4/Apl4 - Unbind Reuest	GerDF4 - ? Unbind Request_In & ! Unbind Request_Out
	AN8 - ? Create Request_Out
	AN9 - ? Delete Request_Out
	AN10 - ? Bind Request_Out
	AN11 - ? Unbind Request_Out

GerDF - eventos do módulo Gerente de Filas e DLSAP
AN - eventos do módulo Alto Nível
AN/Apl - eventos ocorridos na interface entre módulo aplicação e o módulo enlace de dados

Figura 5-18 - Grafo de Acessibilidade

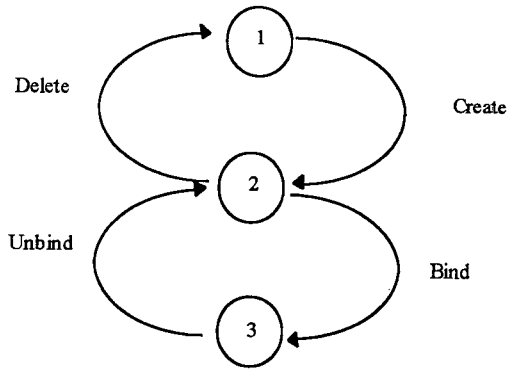


Figura 5-19 - Projeções de Linguagem e Observacional Geradas

C - Simulação

Na simulação deste nível de especificação foram utilizadas tanto a simulação randômica quanto a passo-a-passo. A simulação randômica foi utilizada com o intuito de detectar possíveis erros na especificação que causassem situações de bloqueio total (deadlock). Além do mais, após o término do número de disparos especificado, as facilidades do ESTIM permitem a análise das estatísticas de simulação e do conteúdo dos componentes do estado global da especificação (conteúdos das filas, variáveis de estado, etc).

Já a simulação passo-a-passo foi utilizada com dois objetivos :

- na busca da causa dos erros, quando estes foram detectados pela verificação ou pela simulação randômica;
- na checagem da realização de certas seqüências de eventos válidos e da não ocorrência de eventos inválidos.

Neste segundo método, seqüências de eventos, que chamamos de cenários de simulação, foram escolhidos e as transições foram disparadas passo-a-passo até ser alcançado o último evento desta seqüência ou até a ocorrência de um erro que impedisse a progressão da simulação. Durante o desenrolar da simulação também foi analisado o conteúdo dos componentes do estado global da especificação, confirmando assim se o protocolo estava funcionando de forma correta

para aquele cenário de simulação. Este tipo de simulação segue o princípio de busca focalizada [BOCH80].

5.4.3.2 - Especificação Detalhada

A - Características da Especificação Detalhada

Esta etapa corresponde a um detalhamento da especificação anterior, tomando entretanto cuidado em não refinar todas as partes no mesmo tempo, para evitar problemas de super-especificação. Limita-se então a refinar as partes que são de interesse no nosso estudo.

Na especificação, a parte referente ao escalonamento local recebeu uma atenção especial. Este escalonamento é normalmente realizado por um conjunto de 3 filas por DLSAP (Fila de Usuário por DLSAP, Fila de Pedidos para a LAS, Fila de Escalonamento de Serviço) e pela Fila do Nodo.

Entretanto utilizamos somente a Fila de Usuário DLSAP e a Fila do Nodo no estudo do escalonamento local.

Para efeito de testes, a Fila de Usuário por DLSAP também sofreu uma simplificação no tocante aos pedidos transmitidos. Pois, decorrente da especificação mínima definida, restaram apenas os pedidos que não haviam sido transmitidos, retirando, deste modo, as outras duas categorias (pedidos já transmitidos e ainda disponíveis para retransmissão e pedidos usando a política de escalonamento explícita).

A estrutura de filas de Estelle não permitia todos os procedimentos necessários para manipular as filas de usuário e a fila do nodo. Então optou-se por construir estas filas como estruturas de dados, juntamente com os procedimentos para manipulação destas.

Uma estrutura similar também foi utilizada para a fila de primeira oportunidade, esta que faz parte do módulo escalonador, e é responsável pelo escalonamento global dos serviços aperiódicos.

B - Validação da Especificação Detalhada

Na especificação detalhada o único meio de validação realizável é a simulação. Visto que o funcionamento básico da máquina de protocolos já é descrito pela especificação orientada ao modelo, a simulação foi realizada para testar se não foram introduzidos erros a medida que foram inseridos os comandos Pascal que representam o detalhamento do funcionamento do protocolo, isto é, quando foram inseridas suas características específicas de tratamento de dados, PDUs, etc. No caso em estudo, simulações foram feitas para validar principalmente a estrutura de filas e ligações que são usadas pelo SP50 para escalonamento e armazenamento local de DLPDUs a serem transmitidas. Os resultados obtidos permitiram concluir que a nossa especificação é válida.

5.4.3.3 - Especificação Orientada a Implementação

A - Características da Especificação Orientada a Implementação

Neste ponto a metodologia prevê que o protocolo já esteja funcionando corretamente, o que foi verificado pelas etapas anteriores, sendo necessário agora alterar a especificação para que esta funcione em um ambiente alvo específico.

No caso em estudo, foram realizadas as alterações previstas na metodologia apresentada no capítulo anterior e que foram propostas em [SAQU92]. As alterações básicas realizadas foram :

- introdução de uma especificação Estelle da camada de enlace de dados para cada elemento componente da aplicação. Por exemplo, para o exemplo testado e apresentado a seguir, especificou-se a LAS e quatro DLE Comum;

- retirada da especificação da camada de aplicação e desenvolvimento de programas em linguagem C para realizarem a função da camada de aplicação;
- retirada da especificação da camada física e a representação desta pela comunicação entre processos realizada através de sockets;
- declaração pontos de interação internos ao nível do módulo "specification" com papéis opostos aos dos pontos de interação externos da camada alvo (camada de enlace de dados);
- introdução de sockets para comunicação da especificação com o mundo externo, isto é, a simulação do processo.

Vale ressaltar que a partir das especificações foram gerados automaticamente códigos em linguagem C. Esta geração foi feita utilizando a ferramenta EC do EDT.

B- Testes da Especificação Orientada a Implementação

Os testes realizados aqui concernem à comunicação via sockets e à compatibilidade entre o código C gerado pela ferramenta EC e os programas C codificados manualmente para realizar as funções da camada de aplicação.

Destes problemas de compatibilidade, entre muitos, podemos citar por exemplo:

- a conversão dos limites de arrays de Estelle para C e a declaração de arrays em C;
- na conversão dos valores real (Pascal) era necessário verificar se ferramenta os convertia em double (C) ou em float (C), para então alterar manualmente os programas em C;

5.4.4 - Conclusão

Usando a metodologia de especificação/validação proposta no capítulo 4 e para uma dada aplicação, pode-se construir a simulação da arquitetura de comunicação, a ser interconectada ao simulador de processo SADECA.

Destaca-se que deverão ser especificados somente as partes do protocolo que fazem objeto do estudo que nos interessa.

5.5 - O Ambiente de Simulação para Testes de Arquiteturas Distribuídas de Controle de Processos

A construção do ambiente de simulação parte do princípio que a arquitetura de comunicação já foi testada. Então realiza-se os seguintes passos para a obtenção da simulação:

1. Construir programas C para cada elemento interconectado (sensores, atuadores, controlador), para realizar a função da camada de aplicação, isto é, entre outras funções, a emissão das primitivas de enlace de dados e tratamento da comunicação desta camada com a camada de enlace de dados via sockets.
2. Testar a comunicação destes programas via a simulação da arquitetura de comunicação SP50 previamente construída.
3. Declarar o processo a ser simulado no SADECA, bem as partes referentes a chamada do controlador, isto porque o SADECA em sua utilização habitual, contém também toda a parte referente a controlador do processo, sendo toda a sua operação realizada localmente. Neste trabalho é o nosso objetivo ter o controlador como parte da arquitetura distribuída de controle, para tal faz-se necessário realizar algumas alterações. Cria-se no SADECA as partes referentes a chamada do controlador, isto é, declarando um tipo de controlador, colocando-o no menu de controladores disponíveis, e introduzindo a parte do código que fará a comunicação com o controlador remoto via a simulação da arquitetura de comunicação. Os passos para adição de novos controladores no SADECA é detalhadamente descrita em [KAMM94].
4. Adaptar as estruturas utilizadas no passo 1 para serem utilizadas pelo SADECA, isto porque estas estruturas foram escritas em linguagem C e o SADECA é escrito em linguagem C++. Estas adaptações dizem respeito a tipos de dados, declaração de funções, etc.

A construção da parte de comunicação separadamente da utilização do SADECA é justificada por dois motivos, primeiro os passos 1 e 2 podem ser vistos como parte da especificação orientada a implementação, e segundo adotou-se uma abordagem que procura separar os erros de comunicação dos erros de utilização do SADECA, de modo que seja mais fácil identificá-los quando estes ocorrem.

5.6 - Aplicação do Ambiente de Simulação

5.6.1 - Descrição do Exemplo de Aplicação

Para a demonstração da utilização da metodologia apresentada no Capítulo 4, e da possibilidade de interconexão do SADECA com uma especificação Estelle da camada de enlace de dados do SP50, escolheu-se uma aplicação de controle de processos. Esta aplicação, cuja descrição pode ser vista na Figura 5-20, é composta por um tanque contendo uma solução líquida, um sensor para o controle do nível, uma válvula e um sensor de fluxo que controlam a entrada do líquido no tanque. Além disto, há um sensor que fornece a temperatura do líquido no interior do tanque. Por fim, todos estes equipamentos são supervisionados por uma estação de trabalho. A interligação destes equipamentos e da estação de trabalho é feita através de um barramento do tipo Field Bus SP50.

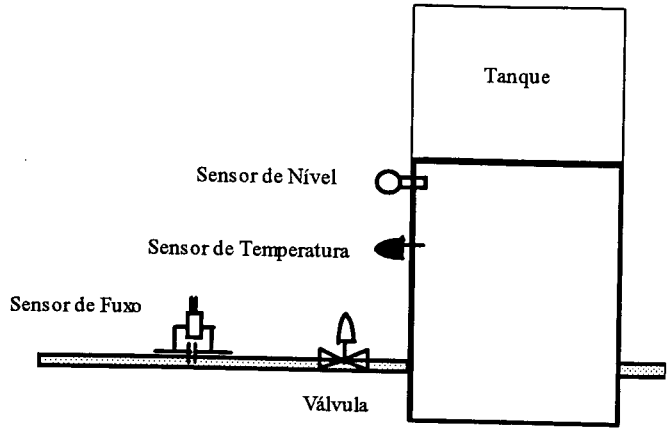


Figura 5-20 - Exemplo do Processo

5.6.2 - A Solução Proposta

5.6.2.1 - Construção da Simulação

Considerando que o intuito desta parte do trabalho é mostrar a possibilidade de um ambiente com representação do processo por um lado e uma simulação da arquitetura distribuída do tipo Field Bus SP50, decidiu-se simplificar o exemplo proposto.

Definiu-se uma equação de controle [2] simples para o tanque (Equação 5-1) levando em conta apenas o sensor de nível e a válvula. Considerou-se que a definição de uma equação para o exemplo completo seria complexo, fugindo do escopo deste trabalho.

$$\frac{h(s)}{F(s)} = \frac{Kp}{Zs+1}$$

Equação 5-1 - Equação Representando um Modelo Hipotético de Tanque

$$Kp = \frac{2\sqrt{h0}}{Kv}$$

Equação 5-2 - Equação definindo o ganho do processo

[2] - As equações foram definidas pelos engenheiros Leonardo César Kammer e Carla Cavalcante do LCMI-UFSC

Dada a Equação 5-1 e a Equação 5-2, e considerando

- Kp = 1, Z = 30 s, h_0 = 1 m, h_{Total} = 2 m

Onde :

- Kp = ganho do processo , Kv = ganho da válvula, Z = constante de tempo do processo

Temos como resultado a Equação 5-3, que foi utilizada na simulação do processo no SADECA.

$$\frac{h(s)}{F(s)} = \frac{1}{30s+1}$$

Equação 5-3 - Equação Representando um Modelo Hipotético de Tanque

A representação através de blocos de função do exemplo inicialmente proposto[SMAR93] que corresponde a um controle cascata : malha de nível seguido de malha de fluxo, pode ser vista na Figura 5-21.

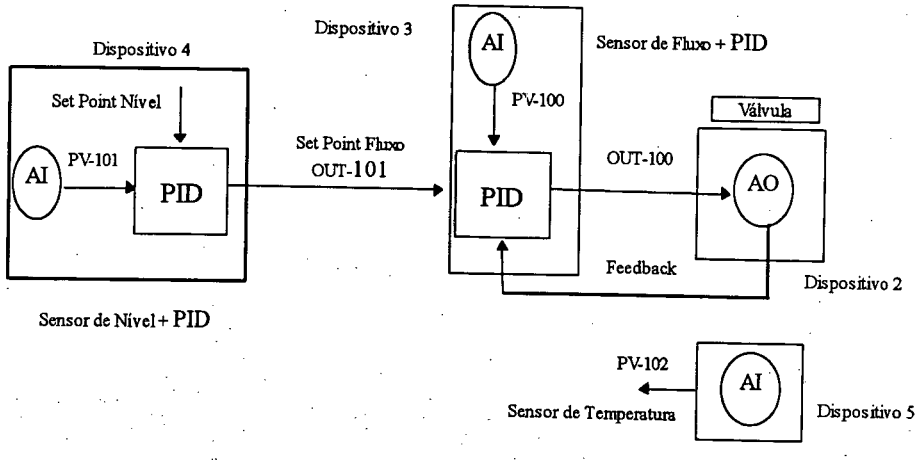


Figura 5-21 - Estratégia de Controle através de Blocos de Função do Exemplo Completo [SMAR93]

5.6.2.2 - A Arquitetura Distribuída

A Figura 5-22 mostra o exemplo completo, do ponto de vista da comunicação. Neste exemplo, o modelo produtor/consumidor é utilizado. Seu funcionamento é o seguinte:

- as estações 3 e 4 são sensores com um bloco de função de entrada analógica (AI) e um de PID;
- a estação 4 produz PV-101 (nível) e OUT-101 (saída do PID, que é o ponto de ajuste do bloco PID da estação 3).
- a estação 3 produz PV-100 (fluxo) e OUT-100 (saída do PID, que posicionará a válvula comandada pela estação 2).
- a estação 5 produz PV-102 (temperatura monitorada pela estação 1).
- estação 2 produz FEEDBACK (posição da válvula, que é usada como realimentação pelo bloco PID da estação 3).
- a estação 1 não produz nenhum dado.[SMAR93].

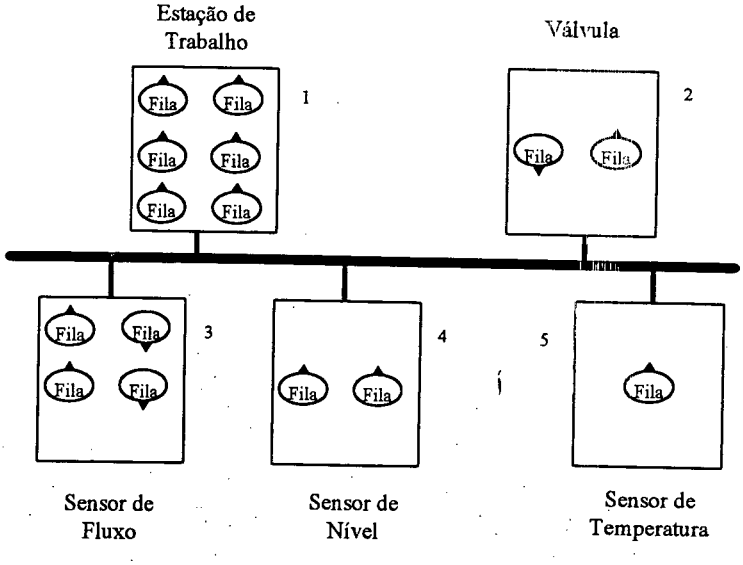


Figura 5-22 - Representação do Exemplo Completo do Ponto de Vista da Comunicação

5.6.2.3 - A Solução Simplificada

Tendo em conta as simplificações realizadas, os dispositivos 3 e 4 (sensor de fluxo e nível respectivamente) foram acoplados em um dispositivo hipotético formando então o ambiente sobre o qual foi realizado a interconexão da ferramenta SADECA com a simulação da arquitetura de comunicação SP50. A estratégia de controle deste exemplo simplificado pode ser vista na Figura 5-23.

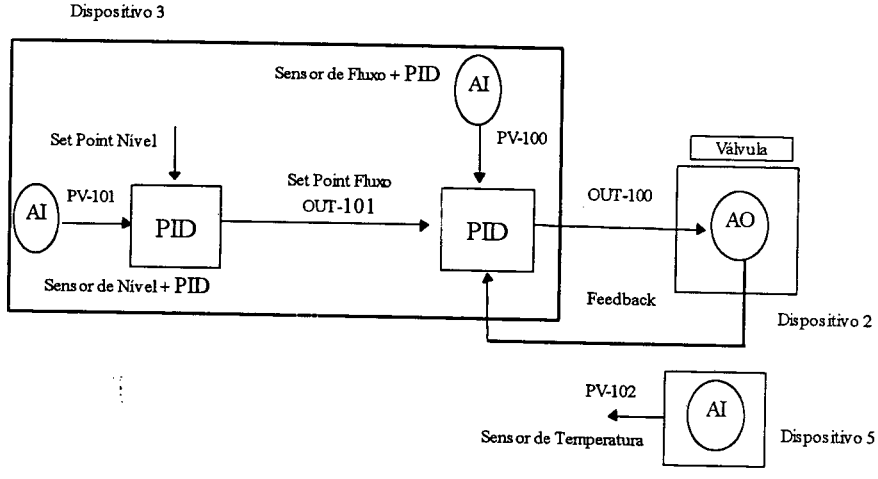


Figura 5-23 - Estratégia de Controle através de Blocos de Função do Exemplo Simplificado

Por último mais uma simplificação foi realizada, esta concerne ao controlador utilizado. Segundo o exemplo, o controlador utilizado seria um PID porém por questão de simplicidade resolveu-se utilizar um controlador de ganho fixo. Este controlador foi definido em [KAMM94]. A Figura 5-24 mostra a estratégia de controle do exemplo depois de todas as simplificações, como foi utilizada no simulador, nesta fase do teste do ambiente.

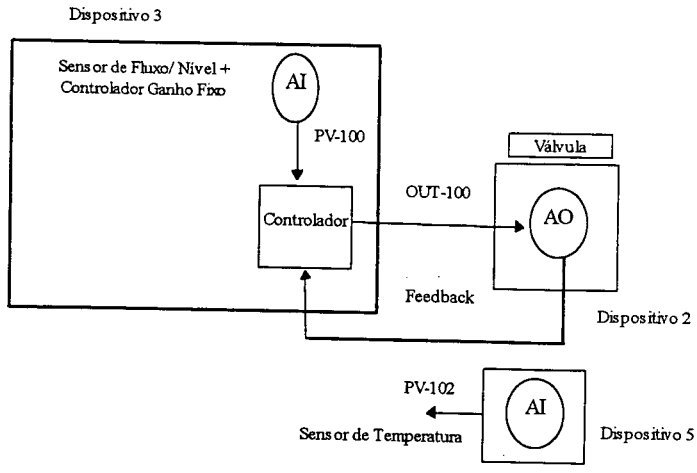


Figura 5-24 - Estratégia de Controle através de Blocos de Função do Exemplo Utilizado no Simulador

Do ponto de vista da comunicação as simplificações resultam que os elementos 3 e 4 (sensor de fluxo e sensor de nível respectivamente) da Figura 5-22 foram acoplados num único elemento.

5.6.3 - Testes e Resultados da utilização do Ambiente de Simulação

Os testes do ambiente de simulação integrado são testes de viabilidade que concernem principalmente a comunicação entre a ferramenta SADECA de simulação do processo e a simulação da arquitetura distribuída de controle. Verificou-se que o SADECA, no qual estava sendo representado o processo e também a válvula, comunicava-se com o módulo representando o dispositivo hipotético (nível e fluxo) da Figura 5-23, pois neste residia o controlador. Esta verificação da comunicação era com o intuito de certificar-se que o controlador ajustava o processo de acordo com os parâmetros recebidos.

Na Figura 5-25 apresentamos a declaração do processo no SADECA, onde é declarado um bloco de processo de dinâmica linear contínua (DLC) e é introduzido sua representação através de uma equação que corresponde a Equação 5-3.

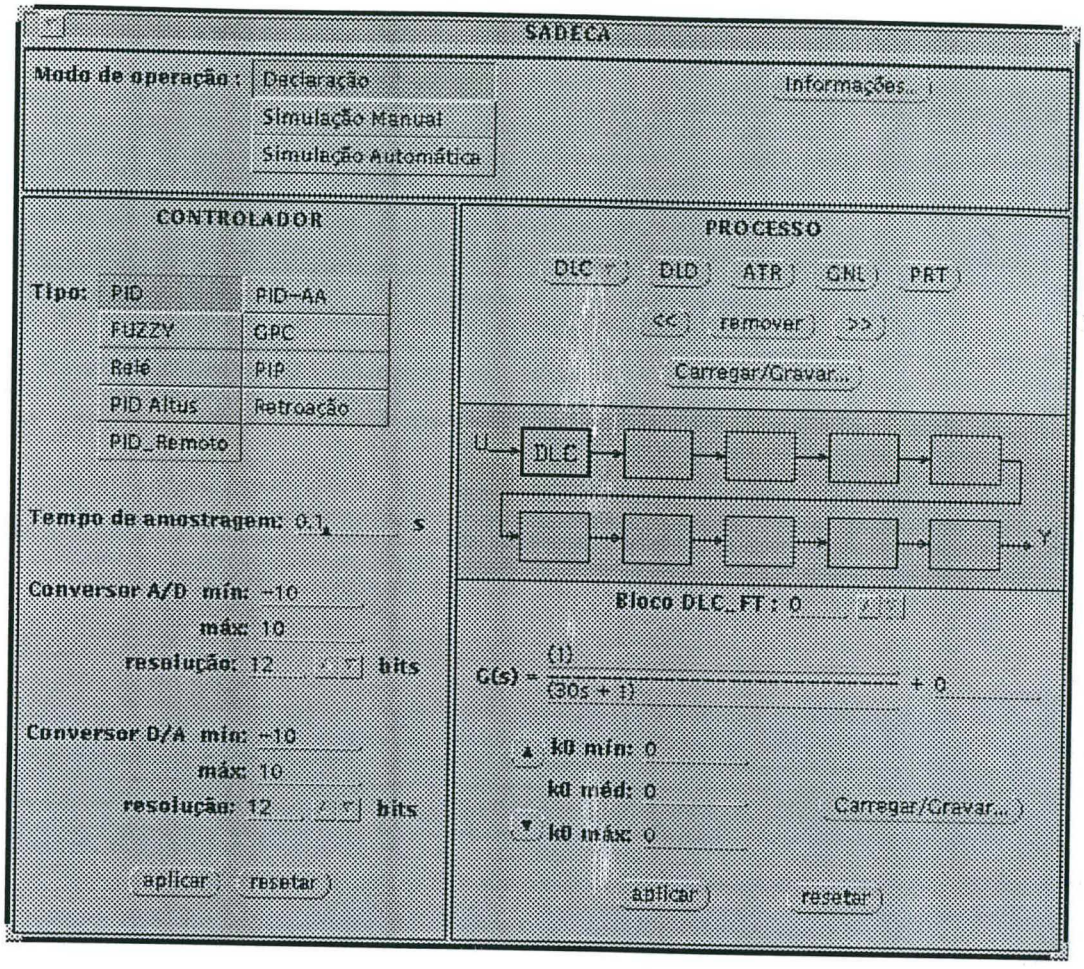


Figura 5-25 - Declaração do Processo

O processo foi executado por um determinado tempo, e observou-se os gráficos gerados pelo SADECA, bem como valores impressos pelos módulos comunicantes, para determinar que a comunicação estava sendo realizada como se pretendia.

Do ponto de vista da simulação da arquitetura de comunicação, foi verificado se o escalonamento realizado pela LAS atendia a demanda de transmissões dos módulos componentes. Este exemplo comprovou que a construção da agenda de escalonamento é de vital importância para o funcionamento de modo adequado do SP50. Durante esta fase de construção de agenda é fundamental analisar bem as reais necessidades de comunicação de cada elemento conectado ao Field Bus.

Foi realizada uma comparação entre uma simulação de controle de processos centralizado utilizando somente o SADECA, e outra com o controlador remoto utilizando a ferramenta SADECA interligada com a arquitetura distribuída de controle. Os resultados obtidos foram os mesmos, porém na arquitetura distribuída o tempo de processamento da simulação foi maior[3]. A Figura 5-26 mostra um exemplo de gráfico gerado pela ferramenta SADECA, o qual foi igual para as duas simulações . O gráfico Y representa o sinal de saída do processo (antes do conversor A/D) e o sinal de referência, enquanto que o gráfico U representa o sinal de controle (após o conversor D/A) [KAMM94].

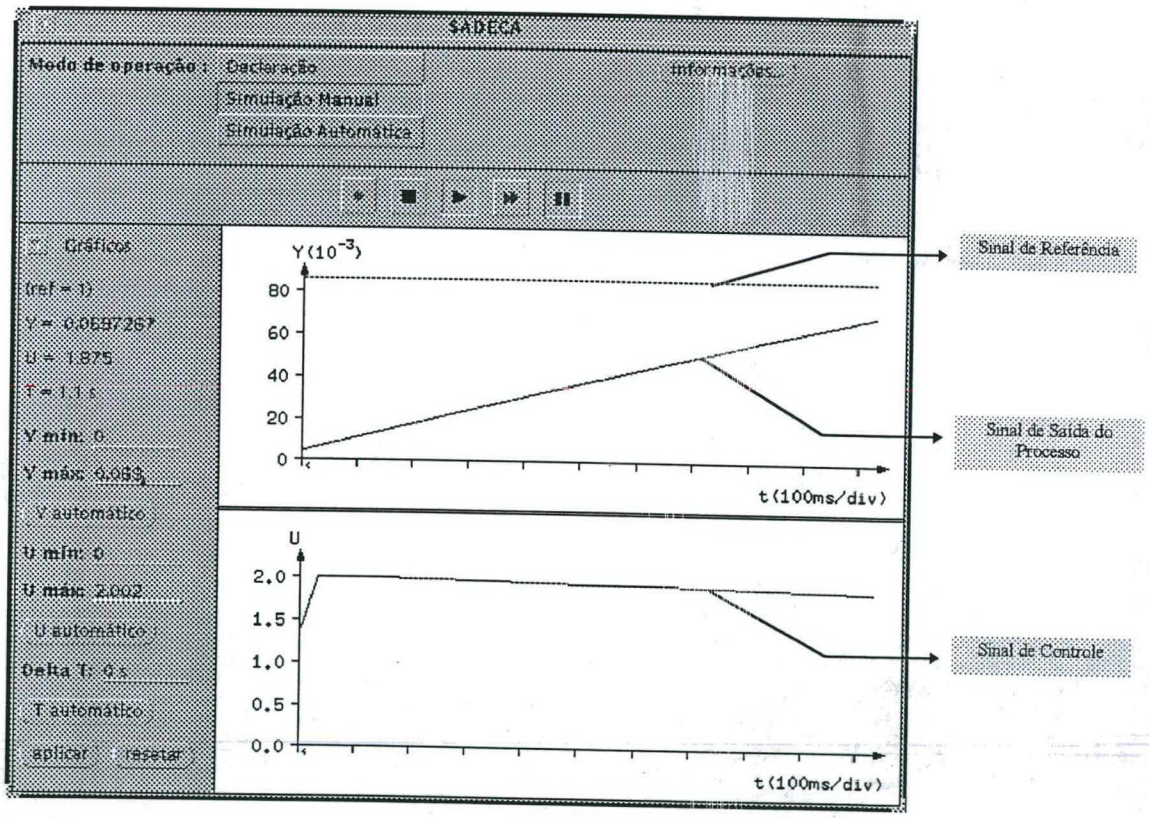


Figura 5-26 - Exemplo dos Gráficos de Saída do SADECA

[3] Deve-se considerar que na simulação da arquitetura distribuída haviam vários processos sendo executados em uma mesma máquina, além do que, há a sobrecarga do processamento da transmissão das mensagens,enquanto que na simulação centralizada há apenas o processo do SADECA.

5.7 - Metodologia para o Uso do Ambiente de Simulação

Tendo como ponto de partida uma descrição informal de uma aplicação a ser simulada neste ambiente, adota-se os seguintes passos para a realização desta:

1. Definir os equipamentos que comporão o ambiente a ser simulado e definir também a arquitetura de comunicação que dará suporte a estes equipamentos, isto é, definir que haverá um elemento do tipo LAS, quantas DLEs Comuns, a quantidade de filas de entrada e saída associadas a cada DLSAP, etc.
2. Construir e/ou adaptar a simulação da arquitetura de comunicação SP50 à aplicação dada e aos objetivos de simulação desejados. Esta fase deve seguir a metodologia apresentada no capítulo 4. Destaca-se que em alguns casos apenas a adaptação é necessária , pois pode-se usar os serviços da camada de enlace de dados que já tenham sido previamente especificados, e as mudanças se limitam ao número de filas e número de DLSAPs.
3. Definir as equações de controle, necessárias para a representação do processo no SADECA.
4. Utilizar os passos descritos no item 5.5 - O Ambiente de Simulação, para as tarefas que concernem ao SADECA e a integração deste ao ambiente de simulação da comunicação SP50, formando deste modo o ambiente de simulação.
5. Realizar testes, observando os gráficos resultantes do SADECA a partir da alteração dos parâmetros e da verificação da sua influência, e observando também, como se comporta a arquitetura de comunicação, principalmente no que se refere ao escalonamento e a delegação das fichas. Pode-se finalmente verificar-se o atendimento a demanda de comunicação dos elementos interconectados, de modo que a perda de dados seja mínima e que sejam atendidos no menor espaço de tempo.

5.8 - Conclusão

Este capítulo apresentou a construção de um ambiente que permite a simulação de arquiteturas distribuídas de controle de processos. Este ambiente é composto de um simulador da arquitetura distribuída de controle baseada no padrão de comunicação SP50 e de um simulador de processos.

Na construção do simulador de comunicação utilizou-se a metodologia apresentada no capítulo 4, o que nos permitiu testar e conhecer melhor o protocolo SP50 que foi uma das propostas de padronização estudada no capítulo 2. Procurou-se mostrar a cada passo da metodologia os resultados obtidos, principalmente no que diz respeito a validação das especificações realizadas.

O ambiente utilizado para simulação de processos foi a ferramenta SADECA, desenvolvido no LCMI-UFSC [KAMM92]. Este foi adaptado para que pudessemos interconectá-lo ao simulador de comunicação, formando então o ambiente de simulação para a arquitetura distribuída proposta.

Neste ambiente de simulação, a partir de uma aplicação sobre um processo simplificado mostrou-se a possibilidade de interconexão dos dois ambientes e determinou-se a melhor forma de especificar a camada de enlace de dados do SP50 para analisar o comportamento a nível de escalonamento e de comunicação.

Podemos citar como resultado, testes preliminares do ambiente de simulação de arquiteturas distribuídas de controle de processos utilizando uma arquitetura de comunicação do tipo Field Bus. Considerando o Field Bus como sendo o futuro para a comunicação no ambiente de controle de processos, este ambiente de simulação que deve ser ainda completado, poderá servir como instrumento de estudo deste vasto campo de aplicação.

Por fim, sugere-se uma metodologia no uso do ambiente de simulação para teste de arquiteturas distribuídas de controle de processos, facilitando, assim, novos estudos nesta área.

Capítulo 6

Conclusões

Este trabalho se desenvolveu seguindo dois eixos principais:

- o estudo da padronização da comunicação do tipo Field Bus e;
- a construção de um ambiente de simulação para testes de arquiteturas distribuídas de controle de processos, utilizando uma metodologia para especificação e desenvolvimento de protocolos de comunicação.

Do primeiro, podemos retirar como fruto deste trabalho, a apresentação das características das principais propostas de Field Bus e a realização de uma análise comparativa, a partir da qual destaca-se as características de maior interesse para a padronização Field Bus. Nota-se que o SP50 está servindo como campo de discussões entre os defensores das duas propostas FIP e PROFIBUS, com destaque para a tentativa de compatibilizar as características distintas de cada proposta.

No segundo, mostrou-se inicialmente a importância do uso de uma metodologia na concepção de protocolos de comunicação, em particular quando se apoia no uso de Técnicas de Descrição Formal, e das suas ferramentas associadas.

Desenvolveu-se e testou-se uma simulação de um ambiente de comunicação SP50 acoplado a uma simulação de controle de processos, permitindo assim que esta fosse realizada de modo distribuído gerando um novo ambiente de simulação de controle distribuído de processos, próximo da realidade futura na área de controle de processos, pois a tendência vai no sentido do uso de equipamentos de campo inteligentes, se comunicando através de Field Bus.

No decorrer do trabalho, surgiram algumas perspectivas, no que diz respeito a aspectos que não puderam ser abordados neste, ou que surgiram ao longo do desenvolvimento deste.

- O SP50 oferece também serviços orientados a conexão, que poderiam ser especificados de acordo com a metodologia apresentada neste trabalho, e então comparados com os serviços sem conexão do ponto de vista da sua utilização. Poderia-se ainda integrar este serviços orientados a conexão com os serviços já especificados neste trabalho, ampliando assim a gama de processos de aplicação suportados.

- Alguns trabalhos em âmbito mundial dizem respeito a realização do escalonamento do SP50, proporcionando atendimento não só as tarefas cíclicas como também as acíclicas com um "hard deadline".

- Enfim, a implementação dos serviços especificados neste trabalho seria importante para conhecer as reais performances e características oferecidas pelo padrão.

Bibliografia

[AGUI89] - M.W.C.de AGUIAR, ; "Análise Comparativa de Desempenho Da Camada de Enlace de Dados do PROFIBUS e do FIP, Propostas Candidatas ao Padrão Field-Bus"; Dissertação de Mestrado Em Engenharia Elétrica - UFSC; Florianópolis; Novembro de 1989.

[AGUI91] - M.V.C. AGUIAR, E. SPOHR, S.B.M. JOBERTO "Aspectos de Implementação de um Servidor MMS"; 9o Simpósio Brasileiro de Redes de Computadores; Florianópolis; Maio de 1991.

[AHO79] - A.V. AHO, J.D. ULLMAN; "Principles Of Compiler Design"; Addison-Wesley Publishing Company; Third Printing; April 1979.

[ALME89] - H.J. ALMEIDA et ali; "Implementação de um Sistema de Comunicação Industrial no CTI"; Seminário Franco-Brasileiro em Sistemas Informáticos Distribuídos; Pág 217-223; Florianópolis; Setembro 1989.

[BACH86] - M. J. BACH; "The Design of the UNIX Operating System"; Prentice-Hall International; 1986.

[BAST90] - A. L.da S. L BASTO, J.E. GIACOMELLI; "Linguagem de Especificação ESTELLE"; Resumo Técnico; Curso de Ciências da Computação - UFSC;1990;

[BERTH83] - B. BERTHOMIEU, M. MENASCHE;"An Enumerative Approach for Analysing time Petri Nets"; Proceedings of IFIP; Paris; 1983.

[BOCH80] - G.v. BOCHMANN, C.A.SUNSHINE; "Formal Methods in Communication Protocol Design"; IEEE Transactions on Communications; Vol 28; Number 4; April 1980.

[BOCH87] - G.v. BOCHMANN; "Usage of Protocol Development Tools : The Results of a Survey"; Protocol Specification, Testing and Verifications; Elsevier Science Publishers B. V.; IFIP 1987.

[BOCH90a] - G.v. BOCHMANN, D. QUIMET, G. NEUFELD; "Implementation support tools for OSI application layer protocols"; Publication #720; Département d'informatique et de recherche opérationelle; Université de Montreal; Canada; March 1990.

[BRIL91] - M. BRILL , U. GRAMM ; "MMS: MAP Application Services for the Manufacturing Industry"; Computer Networks and ISDN Systems;Number 21; 1991; North-Holland.

[BUDK87] - S. BUDKOWSKI, P. DEMBINSKI; "An Introduction to Estelle : A Specification Language for Distributed Systems"; Computer Networks and ISDN Systems 14; Elsevier Science Publishers B. V. (North-Holland); 1987.

[CARO] - R. A. CARO; "Benefits of the FieldBus";

[CAVA91] - CAVALIERI, Di STEFANO, MIRABELLA; "Some Remarks on the Scheduling Mechanism for Acyclic Traffic in the Field Bus Data Link Layer"; IEC_WG6; Milano - Italy; 1991.

[COME91] - D. E. COMER; "Internetworking with TCP/IP"; Volume I : Principles, Protocols and Architecture; Second Edition; Prentice-Hall International; 1991.

[COUR87] - J.P. COURTIAT, "How Could Estelle Become a Better FDT?"; Proceedings of the IFIP Conference on Protocol Specification, Testing and Verification VII, Zurich; 1987

[COUR88] - J.P. COURTIAT; "Estelle*: a powerful dialect of Estelle for OSI protocol description". Proceedings of the 8th IFIP Symposium on Protocol Specification, Testing and Verification, Atlantic City, 1988.

[COUR91] - J-P. COURTIAT, M. DIAZ V.B. MAZZOLA, P. de SAQUI-SANNES; "Description Formelle de Protocoles et de Services OSI en Estelle et Estelle* - Experience et Méthodologie"; CFIP'91; Pau; Septembre 1991.

[COUR92] - J-P. COURTIAT, P. de SAQUI-SANNES; "ESTIM: an integrated environment for the simulation and verification of OSI protocols specified in Estelle*"; Computer Networks and ISDN Systems; North Holland; Elsevier Science Publishers B. V.; 1992

[DEMB89] - P. DEMBINSKI, S. BUDKOWSKI; "Specification Language Estelle"; Elsevier Science Publishers B. V.; 1989

[DIAZ86] - M. DIAZ; "Nets in Protocols"; Presented at the Advanced Course on Petri Nets; September 1986; Bad Honnef; Germany.

[EC92] - "Estelle to C Compiler - EC - User Reference Manual"; Institute National des Telecomunications - Department Systèmes et Réseaux; Evry - France.1992.

[EDB92] - "Estelle Simulator/Debugger - EDB - User Reference Manual"; Institute National des Telecomunications - Department Systèmes et Réseaux; Evry - France.1992.

[ELLO91] - J.P. ELLOY; "Les Contraintes Du Temps Réel Dans Les Systèmes Industriels Répartis"; RGE N° 2/91; Février 1991.

[EM91] - "O Field Bus : A nova Tecnologia Que Revoluciona a Automação Indusrial e a Disputa Pelo Seu Padrão", Revista EM, 20 Julho de 1991.

[ERNB91] - P. ERNBERG, L-A FREDLUND, H. HANSSON, B. JONSSON, F. ORAVA, B. PEHRSON; "Guidelines for Specification and Verification of Communication Protocols"; SICS Perspective; Report N° 1; Swedish Institute of Computer Science; 1991; Sweden.

[FIAL] - S.V. FIALHO, J.L.S. LEÃO, A.C.D. PEDROZA; "Especificação e Verificação Formal de um Célula Flexível de Montagem"; 9° CBA; UFES; Vitória; Espírito Santo.

[FIP87] - FIP - A Standard Proposal For FieldBus

* Generality;
* Service Specification Of Application Layer;
* Data Link Layer Specifications
* Physical Layer Specifications;

1987; France.

[FIP88] - Systeme FIP - Couche Liaison De Donnèes; 1988; France.

[FONS93] - K.V.O. FONSECA e J.M. FARINES; "Uma Análise das Diversas Propostas de Atendimento dos Requisitos de Comunicação Para Sistemas Tempo Real", 11o Simpósio Brasileiro de Redes de Computadores (SBRC); 10 - 13 Maio de 1993; Campinas - São Paulo.

[FUR92a] - H. FURNESS; "Plugging into SP50's FieldBus"; Control Engineering; Mid-March 1992.

[FUR92b] - H. FURNESS; "FieldBus Benefits and Status"; Control Engineering; Mid-March 1992.

[GENR87] - H.J. GENRICH; "Predicate/Transition Nets" in Lectures Notes In Computer Science, vol 254, pp 207-247; New-York:Springer-Verlag; 1987.

[GIOZ86] - W.F. GIOZZA, J.F.M. ARAÚJO, J.A.B. MOURA , J.P. SAUVÉ ; "Redes Locais De Computadores - Tecnologia e Aplicações"; McGraw-Hill; 1986;

[GODD90] - J. GÖDDERTZ ; "PROFIBUS - Technical Publication for Industry and Commerce"; Klockner-Moeller; Bonn - Germany; 1990.

[GUIT96] - H. GUITTER, P. HUNNEL, M. C. VIALLATE; "Un Support á la Spécification de service-OSI"; 1996

[IEC89] - IEC/TC65WG6 "Programmable Controler Message Specification"; June 1989.

[IEC93a] - IEC DIS 1158-3 Digital Data Communications for Measurement and Control - Field Bus for Use in Industrial Control Systems - Part 3 : Data Link Service Definition; 1993.

[IEC93b] - IEC ACDV 1158-4 Digital Data Communications for Measurement and Control - Field Bus for Use in Industrial Control Systems - Part 4 : Data Link Protocol Specification; 1993.

[ISA91a] - ISA/SP50 Field Bus Standard For Use In Industrial Control Systems; Part 3 : Data Link Service Definitions - Version 8 - September 1991; Part 4 : Data Link Protocol Specification - Version 8 - September 1991.

ISA91b] - ISA/SP50 Field Bus Standard For Use In Industrial Control Systems - User Layer Technical Report; September 1991.

[ISA92a] - ISA/SP50 Field Bus Standard For Use In Industrial Control Systems - Part 2 : Physical Layer Specification And Service Definitions; February 1992.

[ISA92b] - ISA/SP50 Fieldbus Application Layer Specification; Draft 2; April 1992

[ISO84] - ISO, "Reference Model of Open Systems Interconnection - Basic Reference Model", IS 7498-1, 1994.

[ISO87] - ESTELLE DIS 9074; Junho de 1987.

[JARD88] - C. JARD; "Valider les Protocoles en Simulant"; Colloque Français sur l'Ingéniérie des Protocoles; Bordeaux - France; Septembre 1988.

[JARD89] - C. JARD, J-M JEZEQUEL; "A Multiprocessor Estelle-To-C Compiler to Prototype Distributed Algorithms on Parallel Machines"; 1989.

[JEZE91] - J-M. JEZEQUEL, C. JARD; "L'experimentation d'algorithms distrinués sur machines parallèles avec ECHIDNA"; Publication Interne 603; Institut de Recherche en Informatique et Systemes Aleatoires (IRISA) ; Septembre 1991.

[KAMM92] - L. C. KAMMER; "Desenvolvimento De Um Ambiente De Simulação Para Análise De Desempenho De Controladores Adaptativos"; Dissertação de Mestrado em Engenharia Elétrica - UFSC; Dezembro 1992.

[KAMM94] - L. C. KAMMER; "Sistema Para Avaliação De Desempenho De Controladores Adaptativos (SADECA) - Versão 1.0"; Manual do Usuário, de Programação e de Manutenção. LCMI - UFSC; 1994.

[KATZ90] - M. KATZ, G. BIWER, K. BENDER; "The PROFIBUS Application Layer "; Karlsruhe - Germany; 1990.

[KUMA89] - S. KUMARAN, J.D. DECOTIGNIE; " Multicycle Operations in a Field Bus : Application Layer Implications"; IECON'89 15th Annual Conference of IEEE - Industrial

Electronics Society, Volume III - Factory Automation; November 1989; Philadelphia - Pennsylvania - USA.

[MAGA86] - M.F. MAGALHÃES; "Software Para Tempo Real"; Campinas; Editora da UNICAMP;1986.

[MANH94] - E. B. MANHAS JÚNIOR. "Uma Metodologia Para O Desenvolvimento De Aplicações Distribuídas Baseada Na Técnica De Descrição Formal Estelle". Dissertação de Mestrado em Engenharia Elétrica; UFSC; Julho de 1994.

[MAP87] - Manufacturing Automation Protocol; Volume IV; Chapter 13 "The MAP Enhanced Performance Architecture (EPA)"; General Motors; July 1987.

[MAP88] - MAP 3.0. "Manufacturing Automation Protocol Specification". Implentation release subject to errata change. 1988

[MAZZ91] - V.B. MAZZOLA; "Contribution A La Conception De Systemes Flexibles De Production : Application De La Technique De Description Formelle Estelle"; Tese de Doutorado; Université Paul Sabatier; Toulouse; France;1991.

[MEYE85] - B. MEYER; "On Formalism in Specifications"; IEEE Software; 1985

[MEYE88] - B. MEYER; "Object-Oriented Software Construction"; Prentice-Hall International; 1988.

[MURA89] - T. MURATA; "Petri Nets : Properties, Analysis and Applications"; Proceedings of the IEEE, Vol 77; Number 4; April 1989.

[PAG89a] - A. PAGLIONE JR. et ali; "Protocolo RS-511 Um Modelo de Implementação";

[PAG89b] - A. PAGLIONE JR. et ali; "SISDI-MAP: Sistema Didático do Protocolo e da Interface de Aplicação MMS do MAP"; Seminário Franco Brasileiro em Sistemas Informáticos Distribuídos; Florianópolis - 11-14/09/89.

[PELL92] - D. PELLE, U. FRITZKE JR, R. WILRICH, E.S. SILVA; "Um Modelo de Implementação do Barramento de Campo PROFIBUS"; CERTI e LCMI - UFSC; Florianópolis; 1992.

[PEHR90] - B. PEHRSON; "Protocol Verification for OSI"Computer Networks and ISDN Systems; 18; pág 185-201; 1990

[PHAL88] - M. PHALIPPOU, R. GROZ; "Using Estelle for verifcation - An experience with the T.70 teletex transport protocol"; .Proceedings of the First International Conference on Formal Description Techniques; Edited by K.J. Turner; Stirling - Scotland ; 1988..

[PHIN91] - T. PHINNEY; "Field Bus - The Bottom-up Approach to an Open DCS"; Control Engineering; 2nd March 1991.

[PINH88] - A.N. PINHO; "Um Estudo da Especificação de Mensagem da Manufatura (MMS) do Protocolo de Comunicação para Ambientes Industriais MAP"; Dissertação de Mestrado em Engenharia Elétrica; UFSC; Dezembro de 1988

[PLEI88] - P. PLEINEVAUX, J.D. DECOTIGNIE; "Time Critical Communication Networks: Field Buses"; IEEE Network Vol. 2; Number 3; May 1988.

[PLEI] - P. PLEINEVAUX; "Field Bus: A New And Efficient Tool In Process Control".

[PRES87] - R.S. PRESSMAN; "Software Engineering - A Practitioner's Approach"; Second Edition; McGraw-Hill International; 1987.

[PROF88] - PROFIBUS Proposal to ISA SP50; February 1988; Germany.

[PROF91] - PROFIBUS Standard; April 1991; Germany

[RODD90] - M.G. RODD, G.F. ZHAO, I. IZIKOWITZ; "RTMMS - An OSI-Based Real-Time Messaging System"; The Journal of Real-Time Systems; 1990; Kluwer Academic Publishers.

[RUDI] - H. RUDIN; "The Dimension of Time In Protocol Specification";

[RZEH89] - H. RZEHAK, A.E. ELNAKHAL, R. JAEGER; "Analysis of Real-Time Properties and Rules for Setting Parameters of MAP Networks" ; The Journal of Real-Time Systems; 1989; Kluwer Academic Publishers.

[SAQU88] - P. de SAQUI-SANNES, J-P. COURTIAT; ESTIM - The Estelle Simulator Prototype of the Esprit SEDOS Project"; Proceedings of the First International Conference on Formal Description Techniques; Edited by K.J. Turner; Stirling - Scotland ; 1988.

[SAQU89] - P. de SAQUI-SANNES, J-P. COURTIAT; "From Simulation to the Verification of Estelle* Specifications"; FORTE 89, The 2nd International Conference on Formal Description Techniques; Vancouver, Canada; Dec 5-8; 1989; Ed North-Holland.

[SAQU90a] - P. de SAQUI-SANNES; "ESTIM User Manual"; LAAS - CNRS; Toulouse - France; 1990.

[SAQU90b] - P. de SAQUI-SANNES; "Prototypage D'un Environnement de Validation de Prototocoles : Aplication a L'Approache Estelle"; Tese de Doutorado; Université Paul Sabatier; Toulouse; France;1990

[SAQU92] - P. de SAQUI-SANNES , C. CHASSOT; "Implantation automatique d'une spécification Estelle en environnement UNIX et TCP-IP"; LAAS-CNRS;

[SHIN] - K.G. SHIN, Q. ZHENG; "Mixed Time Constrained and Non-Time Constrained Communications In Local Area Networks"; Real Time Computing Laboratory - Department

Of Electrical Engineering and Computer Science - The University of Michigan; Ann Arbor Michigam.

[SIDH89] - D. SIDHU, A. CHUNG, T.P. BLUMER; "Experience with Formal Methods in Protocol Development"; ACM SIGCOMM; Computer Communication Review; 1989.

[SIJE88] - R. SIJELMASSI, P. GAUDETTE; "An Object-Oriented Model for Estelle"; Proceedings of the First International Conference on Formal Description Techniques; Edited by K.J. Turner; Stirling - Scotland ; 1988

[SILV] - J.L. SILVEIRA, R. WILLRICH, J.M. FARINES, J.S. FRAGA; " Estudo da Utilização dos Serviços MMS numa Célula Flexível de Usinagem"

[SILV91] - J.L. SILVEIRA; "Estudo da Utilização do Padrão MMS em Aplicações Fabris"; Dissertação de Mestrado em Engenharia Elétrica - UFSC; Dezembro 1991.

[SMAR93] - "FieldBus Tutorial"; SMAR International Corporation; Houston - Texas - USA; 1993.

[SOUZ89] - W. L. de SOUZA; "Estelle: uma técnica para a descrição formal de serviços e protocolos de comunicação"; Revista Brasileira de Computação; Vol 5; Nº 1; Jul/Set 1989.

[SOUZ92] - L. C. de SOUZA; "Field Bus - 1992"; Instec; Setembro 1992.

[SUN90] - "SUN OS Reference Manual Vol. II. - Sun Release 4.1". 21 January 1990.

[TANE88] - A. S. TANENBAUM; "Computer Networks"; Second Edition; Englewood Cliffs; New Jersey; USA;Prentice-Hall International; 1989.

[THAC88] - B. THACKER; "The Computer Integrated Organization : Some Business and Technical Issues for the OSI-MAP/TOP Solution"; Michigan-USA.

[THOM89] - J. P. THOMESSE , Th. LAINE; "The Field Bus Application Services"; IECON'89 15th Annual Conference of IEEE - Industrial Electronics Society, Volume III - Factory Automation; November 1989; Philadelphia - Pennsylvania - USA.

[THOM90] - J. P. THOMESSE; "Time Critical Applications and Time Critical Communications"; ISA/SP50; 1990.

[THOM91] - J.P. THOMESSE, P. LORENZ, J.P. BARDINET, P. LETERRIER, T. VALENTIN; "Factory Instrumentation Protocol : Model, Products, and Tools"; Control Engineering; September 1991.

[TOP88] - TOP 3.0. "Technical Office Protocol". Implentation release subject to errata change. 1988

[VALA89] - J.C. VALADIER, R.C. BURNETT; "A Rede Experimental Quasimap de Comunicação Normalizada em Controles da Produção"; Seminário Franco-Brasileiro em Sistemas Informáticos Distribuídos; Florianópolis; Pág 209 - 216; Setembro 1989.

[VISS86] - C. A. VISSERS, L. LOGRIPPO; "The Importance of the Service Concept in the Design of Data Communications Protocols"; Protocol Specification, Testing And Verification; M Diaz (editor); Elsevier Science Publishing B.V. (North-Holland); IFIP 1986.

[WILL91] - R. WILLRICH; "Uma Proposta de um Modelo de Implementação de Serviços de Apoio a Aplicações Industriais Segundo o Padrão MMS"; Dissertação de Mestrado em Engenharia Elétrica - UFSC; Abril 1991.

[WILL92] - R. WILLRICH; "Implementação da Camada de Aplicação de um Protótipo PROFIBUS"; Relatório Técnico RT 92-23 LCMI-UFSC; Florianópolis; 1992.

[WING90] - J. M. WING; "A Specifier's Introduction to Formal Methods"; Computer; 1990.

Anexo I

ESTELLE e Suas Extensões

A - O Padrão Estelle

A.1 - Introdução

Estelle (Extended State Transition Language) é uma técnica de descrição formal (TDF) de segunda geração, visto que ela reflete a experiência ganha com experimentos no uso de técnicas descrição predecessoras. O seu desenvolvimento foi iniciado pela ISO (International Organization for Standardization) em 1981 e contou com a colaboração do CCITT (Comité Consultatif International Télégraphique et Téléphonique)[1], que definiu o SDL (Specification and Description Language) com o qual Estelle tem características em comum.

Estelle é baseada no modelo de Máquinas De Estados Finita Estendida e na linguagem de programação Pascal.

A.2 - Conceitos Básicos

Uma especificação Estelle é composta por módulos, que possuem um conjunto de pontos de interações através dos quais a comunicação é realizada. O comportamento interno do módulo é descrito por meio de uma máquinas de estado estendida com as ações através de comandos Pascal. Um módulo é dito ativo se possui pelo menos uma transição, caso contrário o módulo é inativo (ou passivo).

---

[1] Atualmente o CCITT chama-se International Telecomunications Union - ITU

A.2.1 - Hierarquia de Módulos

Os módulos que formam a arquitetura Estelle podem ser decompostos em sub-módulos (módulos filho), formando uma árvore hierárquica, cuja raiz é o módulo principal do sistema especificado que recebe o nome de "*specification*".

Os módulos em Estelle possuem um atributo associado que pode ser :

- *systemprocess*;
- *systemactivity*;
- *process*;
- *activity*.

A estruturação dos módulos deve seguir as seguintes regras [SOUZ89]:

- todo módulo ativo deve possuir um atributo;
- subsistemas (módulos *systemprocess* ou *systemactivity*) não são aninhados no interior de um módulo com atributo;
- módulos que incorporam subsistemas são todos inativos e sem atributos;
- módulos *process* e *activity* são aninhados no interior de um subsistema;
- módulos *systemprocess* e *process* podem ser refinados somente em módulos *process* e *activity*;
- módulos *systemactivity* e *activity* podem ser refinados somente em módulos activity;

Os atributos definem o modo pelo qual as transições de uma especificação serão executadas, no que se refere aos aspectos de paralelismo e não-determinismo.

A.2.2 - Paralelismo e Não Determinismo

Estelle permite expressar dois tipos de paralelismo :

- Paralelismo Assíncrono : é permitido somente entre subsistemas, onde cada um dos subsistemas seleciona um conjunto de transições prontas para disparar (no máximo uma por módulo) e as executa em paralelo, de forma independente.
- Paralelismo Síncrono : somente pode acontecer dentro um subsistema, ou mais precisamente entre ações de tarefas diferentes pertencendo ao mesmo subsistema. O subsistema seleciona as transições prontas para disparar (no máximo uma por módulo) e as executa simultaneamente. Somente após o término da execução de todas as transições é que ocorre uma nova seleção.
- Não Determinismo : ocorre entre transições de um mesmo subsistema. O subsistema seleciona e mantém , a cada vez, uma única transição em execução. O não determinismo ocorre entre módulos *activity* descendentes de um subsistema *systemactivity*.

A.2.3 - Comunicação

A comunicação entre módulos pode ser feita de através de dois mecanismos :

- troca de mensagens (interações) ;
- compartilhamento restrito de variáveis.

No primeiro caso, os módulos, os módulos devem estar ligados através de um canal e as mensagens serão enviadas e recebidas através de seus pontos de interação associados a este canal. Cada ponto de interação possui uma fila de tamanho ilimitado na qual ele coloca as interações recebidas. As filas podem ser dedicadas a um único ponto de interação (*individual queue*) ou compartilhadas por vários pontos de interação de um mesmo módulo (*common queue*).

No segundo caso, um módulo pai pode compartilhar variáveis com seu filho, podendo ler e escrever o conteúdo dessas variáveis. A declaração dessas variáveis é feita no módulo filho e declaradas como *exported*. A prioridade pai/filho, que favorece a execução das transições do módulo pai em relação ao filho, impede o acesso simultâneo as variáveis compartilhadas.

A.3 - Sintaxe

A.3.1 - Canais e Pontos de Interação

A.3.1.1 - Canais

Um canal é o meio pelo qual mensagens são transportadas de um módulo a outro. Sua definição determina dois papéis opostos, que indicam as interações que podem ser enviadas ou recebidas por uma módulo.

Sintaticamente define-se um canal da seguinte forma:

channel identificação-canal (PAPEL1,PAPEL2);
by PAPEL1 :
m1;
m2;
.
.
mN;
by PAPEL2 :
n1;
n2;
.
.
nK;

onde m1,...,mN, n1,...,nK são declarações das interações. Sendo que cada declaração de interação consiste de um identificador e possivelmente parâmetros tipados.

O módulo que desempenhar um certo papel , PAPEL1 por exemplo, poderá enviar o conjunto de mensagens deste papel e receber o conjunto de mensagens do papel oposto.

A determinação do papel é realizada através da declaração dos pontos de interação dos módulos.

A.3.1.2 - Pontos de Interação

Os pontos de interação são as portas de entrada e saída de um módulo, através das quais as mensagens são enviadas e recebidas. Existem dois tipos de pontos de interação : externos e internos. O externos são declarados no cabeçalho do módulo e destinam-se a comunicação com o ambiente externo, enquanto que os pontos de interação internos tem como função a comunicação de um módulo com seus filhos e são declarados no corpo do módulo.

A definição dos pontos de interação deve ser do seguinte modo:

***ip* identificação-ponto-interação : identificação-canal (papel) [tipo-de-fila]**

Onde :

- identificação-canal é a identificação de canal previamente definido;
- papel - define um dos sentidos da comunicação através do canal;
- tipo de fila = (*common queue* ou *individual queue*)

Também é possível a declaração de um conjunto pontos de interação do mesmo tipo :

***ip* identificação-ponto-interação : *array* [i..j] *of* identificação-canal (papel) [tipo-de-fila]**

A.3.2 - Módulo *Specification*

O módulo principal da especificação (*specification*) é declarado da seguinte forma :

specification identificação-especificação [classe-atributo-módulo];

[***default*** tipo-fila-default;]

[*timescale* unidade-tempo]

corpo-da-especificação

end.

Onde :

classe-atributo-módulo = (*systemprocess* ou *systemacivity*)

tipo-fila-default = (*common queue* ou *individual queue*)

unidade-tempo = (hours, minutes, seconds, miliseconds ou microseconds)

O corpo da especificação é definido da mesma forma que o de outros módulos, exceto pela inexistência de pontos de interação.

A.3.3 - Outros Módulos

Um módulo é constituído de duas partes :

- cabeçalho e;
- corpo.

A.3.3.1 - Cabeçalho de Módulo

A definição do cabeçalho possui a seguinte sintaxe:

module identificação-módulo [classe-atributo-módulo];
[lista-de-parâmetros;]
[lista-de-pontos-de-interação;]
[lista-de-variáveis-exportadas;]
end;

Onde :

classe-atributo-módulo = (*systemprocess*, *systemacivity*, *process* ou *activity*);

A lista-de-parâmetros contém a lista de variáveis com os respectivos tipos que deve ser passada ao módulo. A lista-de-pontos-de-interação contém os pontos de interação definidos para este módulo. E a lista-de-variáveis-exportadas contém a lista de variáveis com os respectivos tipos que serão compartilhadas com o módulo pai. Esta lista deve ser iniciada pela palavra-chave *export*.

A.3.3.2 - Corpo de Módulo

Já a definição do corpo pode ter duas alternativas :

1 - Declaração completa do corpo do módulo

body identificação-corpo-módulo *for* identificação-módulo;
declarações;
inicializações;
[transições;]
end;

2 - Indicação que o corpo do módulo foi definido em outra especificação ou que o mesmo será definido em refinamentos posteriores da sistema.

body identificação-corpo-módulo *for* identificação-módulo;
external;

Parte de Declarações

A parte de declarações inclui o seguinte :

- canais;
- módulos;
- variáveis do tipo módulo;
- estados de controle;
- pontos de interação internos;
- constantes, tipos, variáveis, procedimentos e funções da linguagem Pascal.

Declaração de Variáveis Módulos

Variáveis módulo servem como referência a instâncias de módulo de um certo tipo de módulo. como por exemplo

***modvar* X, Y, Z : A1**

indica que X, Y e Z são variáveis do tipo módulo especificado pelo cabeçalho de módulo A1.

Declaração de Estados de Controle

Os estados de controle representam o conjunto de estados da máquina de estados finita de um módulo. Eles são declarados da seguinte forma:

***state* lista-identificadores-estado;**

Pode-se juntar um conjunto de estados, de modo a simplificar a descrição das transições.

stateset

identificação-conjunto-estados = [lista-identficadores-estado];

Um exemplo apresenta estas definições ;

***state* V, X , Y;**

***stateset* Z = [X, Y];**

Parte de Inicializações

A parte de inicialização é indicada pela palavra ***initialize*** e pode conter:

- inicialização da variável de estado de controle ;
- inicialização de variáveis Pascal;
- criação de instâncias de módulos;
- estabelecimento de links de comunicação.

A variável de estado de controle é inicializada da seguinte forma ;

***to* estado-de-controle-inicial.**

As variáveis Pascal são inicializadas através do comando de atribuição Pascal. como por exemplo **i : = 10;**

Criação De Instâncias De Módulos

A criação de módulos segue a seguinte sintaxe :

init **identificação-variável-módulo** ***with*** **identificação-corpo-módulo;**

onde a **identificação-corpo-módulo** indica qual corpo do módulo a variável irá utilizar. Isto porque mais de um corpo pode ser especificado para um determinado cabeçalho de módulo.

O comando para destruição de módulos possui a seguinte sintaxe :

release **identificação-variável-módulo;**

Estabelecimento De Links De Comunicação

O estabelecimento de links de comunicação pode ser realizado de dois modos :

attach **identificação-ponto-de-interação1**
to **identificação-variável-módulo.identificação-ponto-de-interação2;**

ou

connect **identificação-variável-módulo1.identificação-ponto-de-interação1**
to **identificação-variável-módulo2.identificação-ponto-de-interação2;**

O comando ***attach*** liga um ponto de interação externo de um módulo a um ponto de interação externo de seu filho, proporcionando ao módulo filho a possibilidade comunicação com o ambiente externo ao seu pai. Os pontos de interação devem possuir papéis iguais.

O ***connect*** pode ser usado para ligar ;

- dois pontos de interação externos de dois filhos;
- um ponto de interação interno de um módulo a um ponto de interação externo de seu filho;
- dois pontos de interação internos de um módulo;
- dois pontos de interação externos de um mesmo módulo;

Os pontos de interação ligados através do ***connect*** devem possuir papéis opostos.

O estabelecimento de links de comunicação deve seguir um certo conjunto de regras :

- um ponto de interação pode participar de apenas uma ligação ***connect***;
- um ponto de interação não pode ter uma ligação ***connect*** com ele mesmo;
- um ponto de interação pode ter ligações ***attach*** simultâneas com um ponto de interação de seu módulo pai e um ponto de interação de seu módulo filho.
- um ponto de interação que possuir uma ligação ***attach***, poderá possuir uma ligação ***connect*** somente se for ele for o ponto de interação do módulo pai.

As comando ***attach*** e ***connect*** possuem seus comando duais que são ***detach*** e ***disconnect***. Estes que possuem a seguinte sintaxe :

detach **[identificação-variável-módulo]identificação-ponto-de-interação ;**

dis***connect*** **[identificação-variável-módulo].identificação-ponto-de-interação;**

ou

dis***connect*** **identificação-variável-módulo;**

onde a última forma desconecta todos os pontos de interação do módulo identificado por **identificação-variável-módulo**.

Transições

A parte de transições descreve o comportamento interno dos módulos. Ela é composta de um conjunto de declarações de transições. Uma transição pode ser do tipo simples ou aninhada, sendo esta última um modo simplificar a descrição das transições. A definição de uma transição é composta por um conjunto cláusulas condicionais e um conjunto de ações. A sintaxe é a seguinte ;

trans

[***priority*** expressão]

[***from*** identificação-estado-de-controle1]

[***to*** identificação-estado-de-controle2]

[***when*** identificação-ponto-interação.identificação-interação[(lista-de-parâmetros)]]

[***provided*** expressão-boolena]

[*delay* (t1[, t2])]
begin
bloco-ações
end;

As cláusulas podem ser colocadas em qualquer ordem entre a palavra ***trans*** e ***begin***. As cláusulas ***when*** e ***delay*** são mutuamente excludentes. A não existência de uma cláusula ***when*** caracteriza uma transição espontânea e cláusula ***delay*** uma transição retardada. A cláusula ***from*** caracteriza o estado atual que deve-se encontrar a instância do módulo. A cláusula ***when*** verifica se a interação identificada é a primeira da fila associada ao ponto de interação. A cláusula ***provided*** habilita ou não o disparo da transição dependendo do resultado da avaliação da expressão booleana ser verdadeiro ou falso. A cláusula ***priority*** determina a prioridade desta transição em relação as outras. A cláusula ***delay*** especifica um tempo mínimo e máximo entre os quais a transição pode ser disparada se tiver sido habilitada e permaneça habilitada pelo tempo mínimo especificado.

Bloco de Ações

O bloco de ações pode conter comando Pascal e comandos específicos de Estelle. Os comandos de específicos de Estelle são :

- ***init, release, attach, detach, connect*** e ***disconnect*** descritos anteriormente.
- ***output*** : o comando ***output*** é utilizado para enviar uma interação através de um ponto de interação. Ele possui a seguinte sintaxe ;

  output **identificação-ponto-de-interação.identificação-interação[(lista-de-parâmetros)];**
- ***nextstate*** : altera o estado de controle
- ***all*** : usado para fazer uma repetição de ações
- ***forone*** : executa as operações sobre o a primeira instância que satisfizer o seu critério de seleção;
- ***exist*** : verifica a existência de uma instância.

A.3.4 - Restrições no uso de Pascal

As principais restrições quanto ao uso de Pascal adotadas por Estelle são :

- as funções declaradas devem ser puras, isto é sem efeitos colaterais;
- ponteiros só podem ser usados em construções puramente Pascal;
- uso restrito do comando ***goto***;
- as declarações ***read*** e ***write*** não podem ser usadas
- o tipo ***file*** não pode ser usado
- não se pode usar ***conformance arrays***

B - A Extensão ESTELLE*

B.1 - Introdução

Estelle* é um dialeto de Estelle melhorado com o mecanismo de Rendez-Vous com a intenção de aumentar o potencial de Estelle para a descrição formal de protocolos. Este mecanismo de rendez-vous tem por objetivo exprimir de maneira mais abstrata as interações entre camadas de protocolo adjacentes. Estelle* tem por objetivo a abstração de algumas características de implementação impostas por Estelle. Estas características concernem a conceito de sistema, prioridade entre transições, a semântica do tempo e o mecanismo de fila FIFO

As diferenças apresentadas por Estelle* para o alcance destes objetivos :

- eliminação dos atributos ***process*** e ***systemprocess***;
- remoção da cláusula ***priority*** e da prioridade pai/filho;
- mudança da semântica da cláusula delay;
- introdução do mecanismo de rendez-vous.

Nós cobriremos apenas o mecanismo de rendez-vous e a cláusula delay, as outras mudanças podem ser vistas em [COUR87] [COUR88]

B.2 - O Mecanismo de Rendez-Vous

O rendez-vous é um mecanismo sincronizado e leva ao disparo concorrente de duas transições com uma possível passagem de parâmetros. Ele é caracterizado por dois pontos principais :

- a cláusula ***when*** e o comando ***output*** permanecem dedicados ao mecanismo de comunicação por filas FIFO.
- é introduzido uma cláusula de pedido de sincronização que expressa uma sincronização explícita entre duas instâncias de transição. Logo, o pedido de sincronização aparece somente ao nível de guarda de sincronização e só pode existir um pedido de sincronização no máximo por guarda de sincronização. A sincronização deve ocorrer em um ponto de interação que possua um disciplina de fila igual a ***no queue***.

A notação do rendez-vous é a mesma usada no CSP (Communicating Sequential Process):

- **IP?interação** para a recepção e;
- **IP!interação** para o envio.

B.3 - A Semântica da Cláusula *Delay*

A semântica do tempo da cláusula ***delay*** foi derivada da semântica do tempo definida para as Redes de Petri Temporais [BERT83] Nesta semântica é suposto que o tempo de disparo de uma transição (o qual não pode ser expresso em Estelle) é muito pequeno em relação aos valores de tempo especificados na cláusula ***delay*** e pode ser considerado nulo [COUR88].

Anexo II

As Ferramentas Estelle

A - ESTIM

A.1 - Introdução

ESTIM (Estelle Simulator based on an Interpretative Machine), é o protótipo do projeto ESPRIT-SEDOS[1], desenvolvido no LASS/CNRS[2] para validação de especificações Estelle*. Ele suporta tanto a verificação quanto a simulação. O ESTIM foi escrito em linguagem de prototipagem ML (Meta-Language) e funciona em ambiente UNIX.

A.2 - Simulação

O ESTIM oferece quatro tipos de funcionalidades em relação a simulação:

-
- acesso a valores de objetos declarados na especificação;
- escolha da política de disparo de transições;
- estatísticas de simulação;
- controle por meio de comentários de qualificação.

[1]- European Strategic Program for Research and Development in Information Technologies/Software Environment for the Design of Open Systems

[2]- Laboratoire D'Automatique Et D'Analyse Des Systemes - Centre National de Recherche Scientifique

A.2.1 - Acesso a Valores de Objetos Declarados na Especificação

O ESTIM possui comandos que permitem ter acesso a vários componentes da especificação global. Destes ressaltamos :

- a arquitetura de especificação e sua árvore hierárquica de instâncias;
- o estado global de cada instância;
- o valor de variáveis e constantes;
- o conteúdo das filas e das interações armazenadas nestas ;
- os intervalos de tempo de disparo dinâmicos.

A.2.2 - Escolha da Política de Disparo de Transições

Quanto ao disparo de transições o usuário pode optar dentre quatro alternativas:

1) **modo passo-a-passo** : este modo permite ao usuário escolher uma dentre a transições disparáveis da especificação. Ele permite ao usuário verificar o estado corrente da especificação, isto é, a cada disparo quais foram as alterações que ocorreram em relação ao estado anterior.
2) **modo randômico** : neste, o usuário escolhe um número de transições que serão disparadas. O controle da simulação retorna ao usuário quando este número é atingido ou quando ocorre um situação de bloqueio (deadlock). O traço da simulação pode ser guardado em arquivo para posterior análise.
3) **modo automático controlado pelo usuário** : aqui o usuário tem a possibilidade de desabilitar o não-determinismo na escolha das transições de modo a garantir "justiça" na seleção de instâncias de módulos que possuam transições disparáveis.
4) **modo execução** : este modo permite ao usuário disparar uma seqüência de eventos previamente armazenada em um arquivo.

A.2.3 - Estatísticas de Simulação

De modo a fornecer ao usuário dados para análise sobre a sessão de simulação, o ESTIM provê um conjunto de estatísticas, que incluem :

- a lista de transições disparadas desde o início da sessão de simulação;
- a lista de transições que nunca foram disparadas;
- a taxa de transições disparadas por instância de módulo;
- a lista de estados globais alcançados por cada instância de módulo.

A.2.4 - Controle por Meio de Comentários de Qualificação

Um comentário de qualificação (qualifying comment) é um modo de dirigir a simulação. Eles são comentários específicos delimitados por (*$ *) ou {$ }. Eles podem ser inseridos antes de uma declaração ou comando, sendo o último caso mais comum. Eles permitem :

- realizar traços da sessão de simulação através da impressão de mensagens e variáveis;
- inserir pontos de parada (breakpoints) dentro do bloco de transição, respeitando porém o princípio de atomicidade, isto que dizer, não é permitido alterar qualquer componente dentro de um ponto de parada. Em um ponto de parada só são permitidos um comando para mostrar as variáveis locais e um para sair do ponto de parada e retornar a para sessão de simulação.

Vale ressaltar que o GENESTIM não faz a checagem sintática dos comentários de qualificação.

A.3 - Verificação

A verificação utilizada no ESTIM baseia-se na geração do grafo de alcançabilidade a partir do estado inicial da especificação. Devido ao problema de explosão combinatória o tamanho do grafo gerado talvez seja inapropriado para a análise. Para contornar este problema o ESTIM oferece a possibilidade de interface com ferramentas que utilizam as técnicas de redução do grafo. A técnica utilizada é a análise por abstração (ou projeção) a qual reduz o grafo com base nas relações de equivalência.

A.4 - O Ambiente ESTIM

A Figura II-1 mostra o ambiente completo do ESTIM. Nela podemos ver o GENESTIM, que inicialmente invoca suas funções de tradutor, fazendo uma análise sintática e da semântica estática das construções Estelle*, reportando os erros quando estes ocorrem. Na inexistência de erros o tradutor gera uma Forma Intermediária (Intermediate Form - FI), que contém todas informações semânticas da especificação original. O GENESTIM depois converte esta forma intermediária em uma árvore abstrata na linguagem ML, esta que será a entrada para o ESTIM.

O ESTIM pode então, fazer as simulações como descrito anteriormente, ou gerar o grafo de acessibilidade para a verificação. Na segunda hipótese, o ESTIM requisita ao usuário definir :

- uma partição onde se define o ambiente e o mundo observado;
- o número máximo de estados gerados e;
- um número de estados, que define que será gravado em arquivo o estado corrente do grafo de acessibilidade a cada vez que este número for atingido.

Então o ESTIM gera o Grafo de Acessibilidade a partir do estado inicial da especificação. Este Grafo de Acessibilidade pode ser interfaceado com ferramentas que façam a redução do grafo através da projeções por equivalência. Estas ferramentas são o PIPN desenvolvido pelo LAAS/CNRS que realiza as projeções de traço e observacional e o ALDEBARAN desenvolvido pelo IMAG que realiza as projeções de bissimulação, observacional e teste.

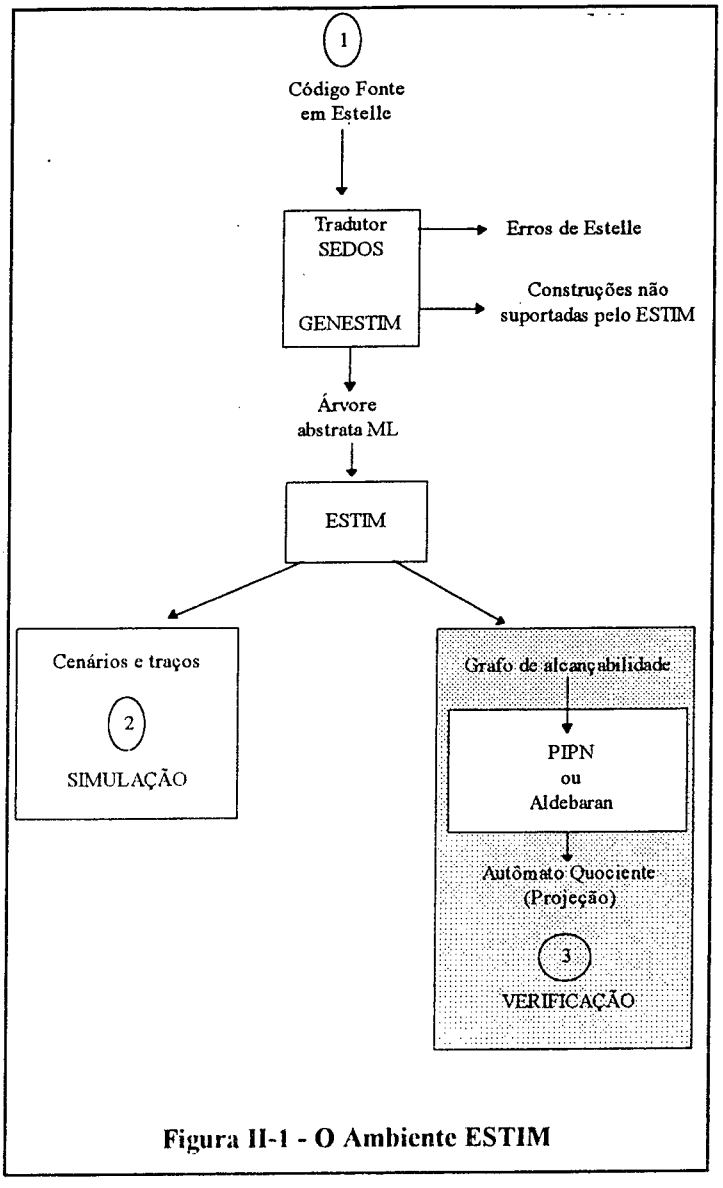


Figura II-1 - O Ambiente ESTIM

B - EDT

B.1 - Introdução

O **EDT (Estelle Development Toolset)** é um conjunto de ferramentas que compreende um compilador de Estelle para C chamado de **EC (Estelle to C Compiler)** e uma ferramenta para simulação e depuração, o **EDB (Estelle simulator DeBugger)**.

B.2 - EC (Estelle to C Compiler)

O EC gera a partir de uma especificação em Estelle um código fonte na linguagem C. Este compilador está funcionalmente dividido em dois componentes :

- um tradutor, responsável pelas análises léxica, sintática e semântica de uma especificação Estelle, gerando como saída as informações referentes a especificação Estelle estruturadas de forma conveniente para ferramentas de processamento automático. Esta saída é denominada de Forma Intermediária (Intermediate Form - IF).
- um gerador de código C, que tendo como entrada a Forma Intermediária constrói a codificação em linguagem C desta.

Estes dois componentes podem ser usados separadamente ou de forma encadeada. Vale ressaltar que para o uso do gerador é necessário a existência prévia da Forma Intermediária.

O EC, assim como o ESTIM, apresenta a possibilidade de inserção de comentários de Qualificação. Estes devem ser iniciados pela colocação do caracter **$** após os delimitadores de comentário de Estelle. Assim sendo os comentários de qualificação podem ser representados sob duas formas :

(*$ texto do comentário *)

ou

{$ texto do comentário }.

O tradutor do EC não faz análise do texto do comentários de qualificação. Ele apenas retira os delimitadores e coloca o texto na forma intermediária.

O EC provê três tipos de comentário de qualificação :

1) Inserção de comados C, sendo delimitados pelos caracteres **C$** , como por exemplo

(*C printf ("teste"); *)

resulta no código C

printf ("teste");

2) Inserção de declarações de tipos, constantes, etc. Estes comentários são delimitados pelos caracteres **W$**, como por exemplo :

```
type (*$W$ char *  *) data_type = ... ;
const (*$W$ 0  *) C = any INTEGER
```

resultariam no código C

```
typedef char* r<N>_data_type;
#define r<N+1>_C 0;
```

3) Sendo que o EC renomeia identificadores para uma forma própria, o último tipo de comentário tem por objetivo impedir esta renomeação. O delimitador deste comentário são os caracteres R$, como por exemplo :

```
type (*R$ *) (*$W$ char *  *) data_type = ... ;
const (*R$ *)  (*$W$ 0  *) C = any INTEGER
```

resultariam no código C

```
typedef char* data_type;
#define C  0;
```

B.3 - EDB (Estelle simulator DeBugger)

O EDB é um simulador/depurador interativo para a Técnica de Descrição Formal Estelle. Seu objetivo é ajudar ao usuário detectar erros que ocorrem durante a simulação de uma especificação Estelle. O usuário pode controlar, observar e rastrear esta simulação por intermédio de comandos.

O EDB tem como entrada a Forma Intermediária e vários arquivos C gerados pelo compilador EC.

B.3.1 - Modos de Simulação

O EDB permite que a simulação seja realizada dos seguintes modos:

- **passo-a-passo** : na qual o usuário vai interativamente coordenando a sessão de simulação, escolhendo o próximo passo a ser executado;
- **aleatória** : o usuário escolhe um determinado número de disparos ou um determinado tempo ou ambos., e motor de simulação executa a simulação escolhendo uma transição dentre as disparáveis em cada momento da simulação até que um dos limites seja alcançado ou ocorra uma situação de erro de execução ou bloqueio total (deadlock).

- **cenários de simulação** : o usuário pode definir através da linguagem de comando um cenário de simulação a ser seguido. Este pode incluir "observadores" (observers) que tem o poder de descrever uma situação de simulação e atuar sobre ela.

O EDB respeita o princípio de atomicidade, logo o usuário só pode interagir entre a execução de duas transições, nunca durante a execução.

B.3.2 - Acesso a Objetos

O EDB permite acessar duas categorias de objetos :

- objetos internos do EDB;
- objetos relacionados a especificação Estelle.

Os objetos da primeira categoria são acessados somente por funções pré-definidas e incluem os seguintes objetos:

- contexto das instâncias de módulo;
- flags e contadores;
- o valor da semente do gerador aleatório;
- variáveis definidas pelo usuário durante a sessão de simulação para uso próprio;
- o nome do arquivo contendo a especificação sendo simulada.

Já os objetos Estelle podem ser acessados pelo nome designado na especificação ou por funções pré-definidas quando não houver um nome associado. Os objetos nomeados incluem :

- parâmetros de instâncias de módulos;
- variáveis exportadas;
- variáveis de instâncias de módulos declaradas na parte de declaração do corpo do módulo, excluindo deste modo as declaradas nas transições, procedimentos e funções;
- variáveis módulos;
- informação sobre um determinado ponto de interação interno e externo;

Os não nomeados e acessados por funções pré-definidas incluem :

- estado de controle corrente de uma instância de módulo;
- informação sobre todos os pontos de interação de uma instância de módulo;

- informação sobre a fila associada a um ponto de interação de uma instância de módulo;
- a hierarquia de módulos;
- subsistemas e módulos habilitados a terminar ou iniciar uma transição
- transições habilitadas a terminar ou iniciar.

O EDB agrupa os seus comandos por categoria. Estas são :

- Sessão Geral;
- Navegação;
- Exibição;
- Modificação
- Observadores;
- Macro e chamadas de Macro;
- Seleção;
- *exit* e *break.*

Anexo III

As Relações de Equivalência [ENRB91]

A equivalência pode ser provada mostrando a existência de uma relação de bissimulação entre o conjunto de estados de dois sistemas de transições :

Definição : Sejam G_1 e G_2 dois sistemas de transicões de estados finitos com conjuntos de estados S_1 e S_2 respectivamente, e seja $S = S_1 \cup S_2$. Uma relação de bissimulação em G_1 e G_2 é uma relação $R \subseteq S \times S$ de modo que $\langle m,n \rangle \in R$ implica

1 - se $m \xrightarrow{\alpha} m'$ então $\exists n' : n \xrightarrow{\alpha} n'$ e $\langle m', n' \rangle \in R$ e

2 - se $n \xrightarrow{\alpha} n'$ então $\exists m' : m \xrightarrow{\alpha} m'$ e $\langle m',n' \rangle \in R$

Definição : Dois sistemas de transição G_1 e G_2 são equivalentes se existir uma bissimulação relacionando os estados iniciais do dois sistemas.

As relações de equivalência existentes, em ordem decrescente do grau de equivalência, são apresentadas a seguir :

1 - Forte : existe uma equivalência forte se for considerado todas as transições, incluindo as transições internas, e estas são igualmente importantes e realizam uma bissimulação no grafo de transições original. A equivalência forte requer que cada evento em um sistema de transição seja mapeado exatamente por um evento no outro sistema de transição. Devido as suas características, a equivalência forte dificilmente é utilizada para a verificação.

2 - Observacional : a equivalência observacional é mais fraca que a anterior, sendo necessário primeiro redefinir a noção de bissimulação para apresentá-la.

Sejam os estados m e m', escreve-se m $\xrightarrow{\delta}$ m' se m pode realizar o evento α de entrada ou saída precedido ou sucedido por um número finito (possivelmente zero) de eventos internos τ e terminar em um estado m'.

Escreve-se m $\xrightarrow{\delta}$ m' se m pode realizar zero ou mais eventos internos τ e terminar em um estado m'. Deste modo então redefinindo a noção de bissimulação:

Definição : Sejam G_1 e G_2 dois sistemas de transições de estados finitos com conjuntos de estados S_1 e S_2 respectivamente, e seja $S = S_1 \cup S_2$. Uma relação de bissimulação fraca em G_1 e G_2 é uma relação $R \subseteq S \times S$ de modo que $\langle m,n \rangle \in R$ implica

1 - se m $\xrightarrow{\alpha}$ m' então $\exists$n' : n $\xrightarrow{\delta}$ n' e $\langle m', n' \rangle \in R$ e

2 - se n $\xrightarrow{\alpha}$ n' então $\exists$m' : m $\xrightarrow{\delta}$ m' e $\langle m',n' \rangle \in R$

A equivalência observacional trata os eventos internos de forma especial, ela requer que estes sejam mapeados por zero ou mais eventos internos no outro grafo e vice-versa.

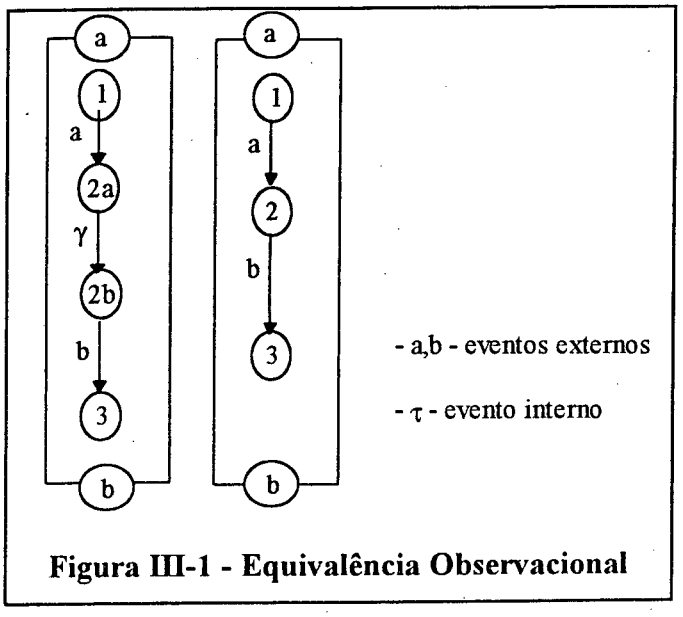


Figura III-1 - Equivalência Observacional

3 - Teste: As equivalências forte e observacional distinguem os sistemas de transição nos pontos exatos onde as escolhas não deterministas são feitas. A equivalência de teste, também faz a distinção entre sistemas que possuem um diferente grau de não determinismo, porém é mais insensível no que se refere aonde tais escolhas são realizadas. Aqui é necessário introduzir o conceito de grafo de aceitação.

Definição : Um grafo de aceitação é um grafo determinístico cujo estados possuem informações sobre possíveis deadlocks codificados como conjuntos de aceitação. O conjunto de aceitação *n.acc* de um estado *n* é um conjunto de transições : onde cada membro de n.acc é um conjunto de transições representando uma obrigação de poder continuar de *n* com qualquer transição do conjunto.

Uma equivalência de teste existe se provarmos a existência de uma relação de bissimulação entre dois grafos de aceitação. E que *m.acc* e *n.acc* sejam compatíveis, isto é, que cada elemento de m.acc seja um superconjunto de um elemento de *n.acc* e vice-versa. Um conjunto A é superconjunto de um conjunto B se todos os membros de B também forem membros de A. A Figura III-2 mostra um exemplo de equivalência de teste.

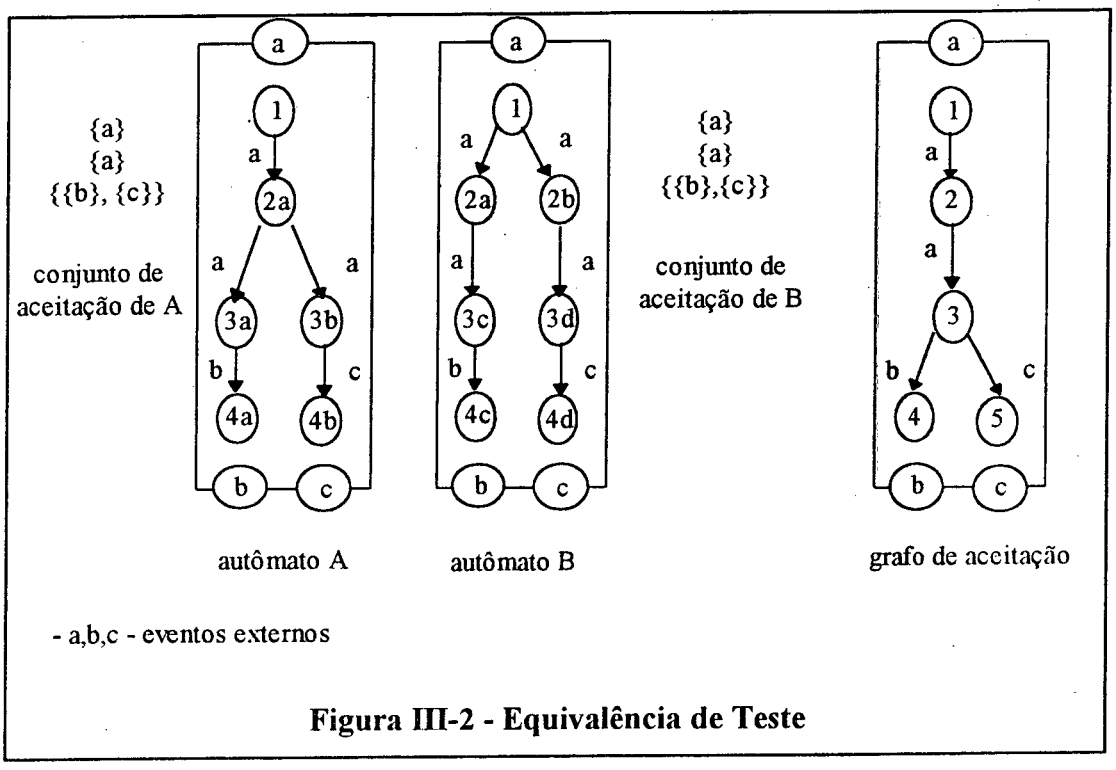


Figura III-2 - Equivalência de Teste

4 - Traço (ou de linguagem).: a equivalência de traço não distingue entre os diferentes graus de não determinismo e é insensitiva ao deadlock. Para que dois sistemas de transição possuam equivalência de traço, basta que exista uma bissimulação entre seus grafos de aceitação, desconsiderando os seus conjuntos de aceitação. Também pode-se definir a equivalência de traço de dois grafos, se eles realizam as mesmas seqüências de eventos. A Figura III-3 exemplifica a equivalência de traço.

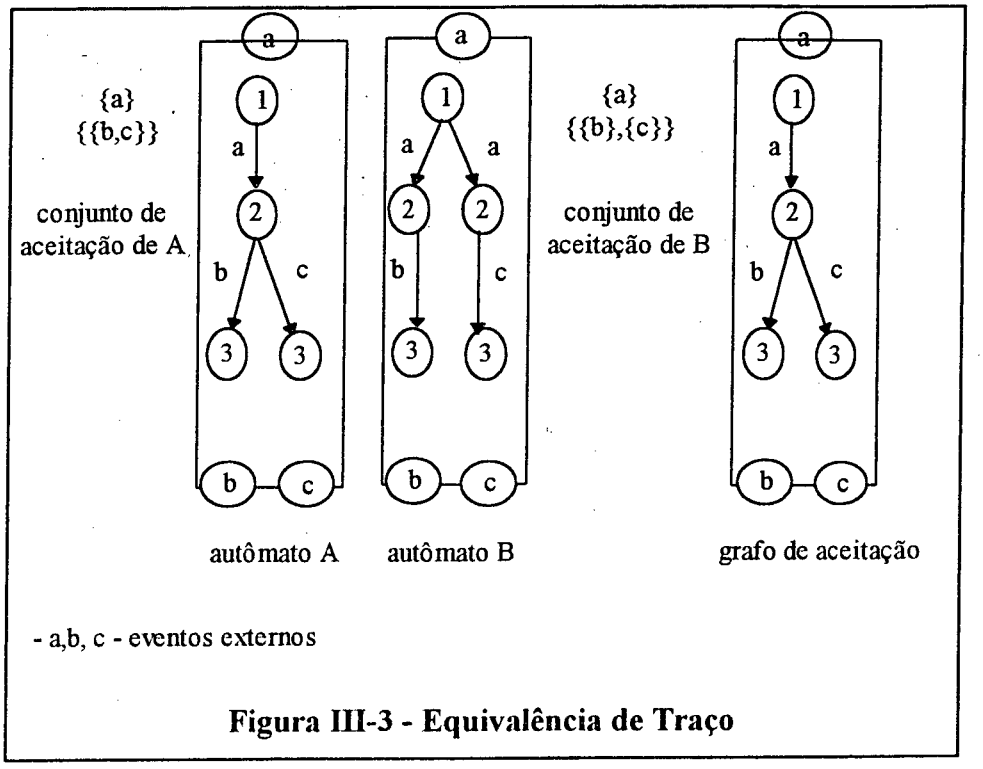


Figura III-3 - Equivalência de Traço