PROPOSTA DE UMA METODOLOGIA DE ESPECIFICAÇÃO PARA IMPLEMENTAÇÃO DE PROTOCOLOS DE COMUNICAÇÃO

Peter Eugene Lonergan (*)
Solon Benayon da Silva (*)
Stefania Stiubiener (**)
Wagner L. Zucchi (**)
Wilson V. Ruggiero (**)

Centro Científico Rio (*)
IBM Brasil
Caixa Postal 4624
20.071. Rio de Janeiro, RJ

Laboratório de Sistemas Digitais (**)
Universidade de São Paulo
Caixa Postal 11455
01000, São Paulo, SP

SUMÁRIO

O presente trabalho propõe uma metodologia de especificação para a implementação de protocolos de comunicação de dados. Nele é apresentada uma visão geral do que se entende por "especificação de protocolos", e por "especificação de implementação".

Em seguida é mostrada uma estrutura da especificação de implementação, suas características principais e medidas de desempenho e testes.

Este trabalho faz parte do ***Programa de Desenvolvimento Conjunto de Pesquisas em Comunicação de Dados*** do qual participam o Centro Científico Rio da IBM Brasil e o Laboratório de Sistemas Digitais do Departamento de Engenharia de Eletricidade da Escola Politécnica da Universidade de São Paulo.

1. INTRODUÇÃO

O permanente crescimento dos sistemas de comunicação de dados observado nas últimas décadas, constitui um desafio para as áreas de pesquisa e desenvolvimento de projetos avançados.

A atual penetração da comunicação de dados na sociedade, desde o nivel empresa/indústria, até o nivel de usuários de sistemas internacionalmente interconectados levou, não só a criação de novas técnicas em engenharia de comunicação, mas também a elaboração de uma sofisticada metodologia de organização para a troca de informações. Assim nasceu a "Engenharia de Protocolos", área que abrange o projeto, a especificação, a implementação e os testes de protocolos de comunicação de dados (1).

Pela natureza distribuida e heterogênea dos sistemas de comunicação de dados, se impõe a existência de protocolos bem estruturados e especificados, de tal forma que a sua implementação em diversos sistemas, se comporte de maneira a permitir a desejada intercomunicabilidade. Por isso se faz necessário uma especificação de implementação que leve em consideração a generalidade do desempenho das funcões de um determinado protocolo e as possíveis características locais de cada sistema.

2. ESPECIFICAÇÃO DE PROTOCOLOS PARA COMUNICAÇÃO DE DADOS

2.1. Especificação Informal e Especificação Formal de um Protocolo

Os primeiros projetos de protocolos de comunicação utilizaram como meio de especificação, as linguagens naturais, caracterizando-se por um gráu de liberdade de expressão muito grande. Um exemplo clássico nesse aspecto é a "especificação informal" utilizada na especificação do protocolo BSC da IBM (5).

A especificação informal de um protocolo se mostrou insuficiente em precisão (dando margem a diferentes interpretações para a mesma descrição de um objeto), ineficiente (tendendo a ser muito extensa) e imprópria para a verificação e geração automática do protocolo correspondente.

Um avanço em relação a especificação informal foi a especificação "semi-informal", que acrescentou às descrições informais, elementos de especificação formal, tais como diagramas de estado (6) ou diagramas de estado e tabelas de transição (7). No primeiro caso estão as descrições usadas inicialmente pela CCITT na definição do protocolo X.25 e no segundo caso está a descrição do protocolo de transporte da ISO.

A necessidade de utilização de métodos de verificação das propriedades de um protocolo (alcançabilidade, falta de impasses, etc), o aspecto da geração automática da implementação de um protocolo e o aparecimento da tecnologia de testes, constituiram a motivação do desenvolvimento da metodologia de especificação de protocolos, conhecida por "especificacões formais", que vem se consolidando a partir da década de 70.

A classificação das linguagens utilizadas na especificação formal de protocolos pode ser feita de acordo com o modelo de formalização escolhido, ou seja (8):

- modelos de autômato finitos ou transição de estados
  - máquinas de estado finitos
  - linguagens formais
  - grafos
- modelos baseados em linguagens de programação
- modelos baseados em lógica temporal

Observa-se atualmente a coexistência da especificação informal com a especificação formal de um protocolo, constituindo um consenso geral entre os profissionais da área que, para uma perfeita compreensão de um protocolo é necessária, tanto a disponibilidade da especificação formal quanto da disponibilidade da especificação informal.

A atuação dos órgaos de padronização na área de especificações formais teve como resultado a escolha de duas linguagens, a ESTELLE (9) e a LOTOS (10), desenvolvidas pela ISO, e a linguagem SDL (11) desenvolvida pela CCITT.

2.2. - Especificação de Protocolos Orientada a uma Implementação Compatível

Conforme mencionado na Introdução, a interconexão de sistemas heterogêneos com fins de interoperabilidade, levanta a problemática da implementação do mesmo protocolo em diversos sistemas, com características que atendam a essa exigência. Para o cumprimento dessa tarefa, faz-se necessário uma especificação de implementação bem estruturada e concreta, capaz de conduzir a um comportamento uniforme do protocolo de comunicação implementado em vários sistemas abertos, interconectados.

3. ESPECIFICAÇÃO DE IMPLEMENTAÇÃO

No contexto de definição de protocolo de comunicação, o termo ***especificação de implementação***, refere-se a um documento elaborado pelas partes interessadas na intercomunicação de suas máquinas contendo um ***conjunto mínimo*** de detalhes de implementação de um protocolo, necessários para a interoperação desejada.

Nesse sentido, uma especificação de implementação pode ser vista como uma espécie de "acordo de cavalheiros", pelo qual as partes envolvidas (duas ou mais) escolhem opções oferecidas por um determinado protocolo, preenchem lacunas em suas especificações e concordam a respeito de uma interpretação possível para a operação da entidade que implementa o protocolo, dentre várias aceitáveis.

A partir da especificação de implementação, as partes interessadas podem trabalhar isoladamente com a expectativa que seus produtos venham a se interconectar sem exagerado esforço conjunto e sem mudanças substanciais do produto envolvido.

Essa última observação permite definir a ***qualidade de uma especificação de implementação*** pela relação entre o tempo de desenvolvimento agragado e o tempo dispendido em ajustes que permitam a entrada em operação da rede, após a fase de implementação, ou seja:

$$Q(I) = \frac{n\,T(i)}{T(a)}$$

onde:

Q(I) = qualidade da especificação de implementação
T(i) = tempo para se obter uma implementação
T(a) = tempo de ajustes
n = número de partes envolvidas

É obvio que, nessa definição se admite que as implementações consideradas correspondam a tempos médios de desenvolvimento em condições semelhantes de "homens-máquina". Vê-se pela expressão que o esforço para a produção de uma especificação de implementação é tanto mais justificável quanto maiores as dificuldades de implementação e quanto mais numerosas as partes envolvidas. A arquitetura MAP (12) pode ser citada como o resultado de um esforço nesse sentido.

3.1. Especificação de Implementação e Arquitetura de Redes

Para discutir o conteúdo de uma especificação de implementação cumpre observar inicialmente a "insuficiência" da especificação de um protocolo em particular ou de uma camada em particular. Assim, por exemplo, uma especificação muito precisa da camada de transporte é inútil (no sentido de permitir a interoperabilidade) sem uma especificação tão ou mais precisa das camadas inferiores.

Generalizando esse raciocínio percebe-se que a especificação de uma camada deve sempre se inserir em um contexto de um documento maior que define uma arquitetura de rede, ou seja, os serviços oferecidos e as funções realizadas pelas diversas camadas tendo em vista uma aplicação específica. Como exemplos de documentos deste tipo pode ser citada a própria especificação MAP e a definição da arquitetura da rede de transferência eletrônica de fundos, recentemente aprovada pela ABNT (13). Nesse documento, são definidas as funções das diversas camadas, com referências a protocolos padronizados, quando existentes, ou com a definição do protocolo utilizado. No primeiro caso, a aplicação considerada é a automação do ambiente industrial e no segundo, pretende-se obter um sistema adequado para um serviço de automação comercial.

3.2. Conteúdo da Especificação de uma Implementação

Definida a arquitetura que se pretende implantar, pode-se iniciar o processo de especificação das camadas individuais, seguindo as seguintes etapas:

3.2.1. Escolha de Opções

Parte-se geralmente de uma especificação do protocolo, normalizada e em linguagem informal. Se tal especificação não existir, é mister elaborá-la nessa etapa. No entanto, esse caso está ficando cada vez mais raro, a medida em que os trabalhos de normalização da ISO avançam.

O protocolo normalizado tem a intenção de oferecer serviços a uma ampla gama de usuários com diferentes tipos de aplicações. A especificação de implementação deve escolher um conjunto de opcões adequadas a arquitetura em que ela está inserida e com a aplicação que se tem em vista. Dentre essas opções, podem ser destacadas:

1. classes de procedimentos oferecidas (por exemplo, classes do protocolo de transporte no padrão ISO IS 8073 (7))
2. Unidades funcionais disponíveis (por exemplo, unidade de acesso básico do protocolo FTAM (14))
3. Campos opcionais utilizados nas unidades de dados do protocolo (por exemplo, campos de dados no pacote CALL REQUEST do protocolo X.25 (6))
4. Tamanhos de campos variáveis nas unidades de dados do protocolo (por exemplo, tamanho do campo de endereço no protocolo de enlace para redes locais ISO 8802/2 (15))

É importante observar que, do ponto de vista da especificação de implementação, não basta que uma entidade possa informar sua parceira quais as opções que queira utilizar. Assim, por exemplo, o protocolo de transporte ISO incorpora um mecanismo pelo qual uma entidade de transporte informa a outra entidade parceira, qual o tamanho do campo de numeração de TPDU's que ela pretende utilizar. Porém, se a entidade remota não implementar o mesmo tamanho de campo, o mecanismo servirá apenas para constatar a incomunicabilidade das entidades.

A especificação de implementação pode deixar opcões ainda em aberto, porém com as seguintes restrições:

1. Se o protocolo prevê mecanismos pelos quais a opção escolhida em tempo de operação pode apenas ser informada a entidade remota, as opcões previstas pela especificação devem ser todas implementadas pelas entidades que pretendem ser conformes a ela
2. Se o protocolo prevê mecanismos de negociação de opcões, a especificação de implementação deve definir a modo preferencial de operação (caso o próprio protocolo não o faça) e exigir que todas as implementações que lhes são conformes ofereçam e aceitem esse modo de operação
3. Se o protocolo não oferecer nenhum dos mecanismos descritos não é possível manter opções na especificação de implementação

3.2.2. Definição dos Requisitos de Qualidade

O passo seguinte para a elaboração de especificações de implemantação é a definição dos requisitos de qualidade de serviço que devem ser atendidos. É importante que todas as implementações atinjam um índice mínimo nos parâmetros de desempenho (como a vazão, por exemplo), que permitam a sua interoperabilidade. Todavia, nesse caso, o valor exato do índice de qualidade não é importante, mas apenas o índice mínimo (isto é, de menor qualidade) deve ser especificado.

3.2.3. Resolução de Ambigüidades

Uma vez resolvidas as diferentes classes de opções oferecidas pelo protocolo, deve se passar à resolução das ambigüidades. Tal tarefa é particularmente difícil pois as diferentes interpretações da linguagem humana surgem em razão de sutilezas lingüísticas, dificuldades de tradução, diferenças culturais e outros fatores de difícil previsão e controle. A solução geralmemte adotada consiste em escrever novamente o protocolo em uma linguagem mais formal e menos sujeita a múltiplas interpretações. As próprias normas de definição dos protocolos vêm sendo modernamente completadas com descrições em linguagens semi-formais (tabelas de transição de estados, redes de Petri etc.) ou formais.

A especificação de implementação deve fornece ou completar uma descrição desse tipo, detalhando as opções escolhidas e relacionando cuidadosamente as operações realizadas com os parâmetros fornecidos nas interfaces do serviço e os valores das variáveis auxiliares na descrição.

Esse objetivo pode ser atingido em duas etapas:

1. definem-se as mensagens de interface com os níveis superiores, inferiores e com a função de gerenciamento, se necessário, em termos de parâmetros de entrada e de saída das primitivas. Todos os parâmetros devem ser definidos, incluindo identificadores locais, variáveis de controle de fluxo no interface etc.

2. define-se a operação do protocolo através de um autômato finito (possivelmente extendido). Os eventos que produzem transições de estados do autômato devem ser os definidos no item anterior e, em todos os estados deve ser especificada a ação realizada para cada um dos eventos anteriormente definidos. No caso de ações que provoquem o envio de primitivas para outras entidades, o valor de todos os parâmetros da primitiva devem ser especificados.

A lingugem que parece mais adequada para a descrição de protocolos em especificações de implementação são as tabelas de transição de estados. Tal ferramenta possui formalidade adequada para resolver ambiguidades e fácil leitura e implementação. A possibilidade de implementação automática oferecida por outras linguagens (como Estelle) será discutida na seção 4 deste trabalho.

3.2.4. Valores de Parâmetros

Uma vez definida a máquina de estados do protocolo a especificação deve definir valores iniciais de variáveis e contadores, bem como o estado inicial do autômato.

3.2.5. Cenários de teste

Finalmente a especificação deve conter cenários de teste que permitam exercitar os diversos estados previstos para a entidade que implementa o protocolo. É importante observar que, dada a metodologia até aqui utilizada, nenhum cenário de teste pode ser suficientemente detalhado para garantir a compatibilidade de uma implementação com a especificação.

Com efeito, a implementação pode ser pensada como uma caixa preta da qual se conhece o alfabeto de entrada e se pode verificar o alfabeto de saída. Os cenários de testes ideais devem permitir comprovar a equivalência dessa caixa preta com o autômato utilizado para a descrição do protocolo. Todavia, a teoria de autômato finitos nos garante que um tal teste não existe.

Para entender o porquê considere um exemplo simples. O protocolo de nível N ao receber uma mensagem X no estado A deve enviar uma mensagem Y e passar para o estado B. Numa tabela de transição de estados essa situação pode ser descrita como mostrado pela Figura 1.

ESTADO ATUAL	EVENTO	AÇÃO	PRÓXIMO ESTADO
A	Recebe X	Envia Y	B

Figura 1 - Exemplo de tabela de transição de estados.

Imagine-se agora que esse protocolo fosse implementado de acordo com a tabela mostrada pela Figura 2. Examinando as tabelas é óbvio que os dois autômata não são equivalentes, porém, entendendo-se a implementação como uma caixa preta, faz-se necessário repetir o mesmo teste 500 vezes para que a incongruência seja verificada.

ESTADO ATUAL	EVENTO	AÇÃO	PRÓXIMO ESTADO
A	Recebe X	Se n < 500 Envia Y n = n + 1	B
		Senão Envia Z	C
A	Outro evento	n = 0	A

Figura 2 - Exemplo de implementação do protocolo Figura 1.

É óbvio que o número 500 é arbitrário. Para qualquer número de testes realizados, sempre é possivel que a descrepância fique mascarada. Quando se considera que tais problemas podem ser introduzidos por esgotamento de áreas de memória dinâmica, insuficiência de pilhas de processos, variáveis não inicializadas e outros problemas, percebe-se quão limitadas são as capacidades de teste nesta metodologia.

A Figura 3 resume as etapas necessárias para a obtenção de uma especificação de implementação.

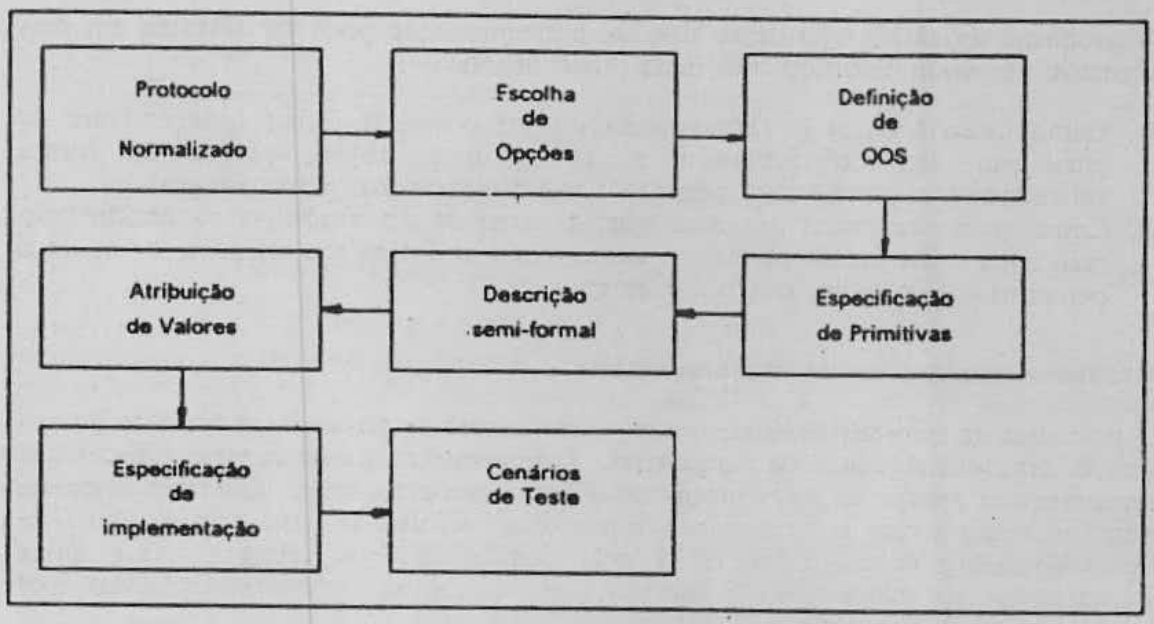

Figura 3 - Etapas da elaboração da especificação de implementação

4. ESPECIFICAÇÃO DE IMPLEMENTAÇÃO TRANSPORTÁVEL

Uma especificação de implementação, definida como na seção 3, possui a vantagem de conter o mínimo de informações necessárias para a intercomunicação das suas implementações. Os programadores que desenvolvem entidades de ume camada de acordo com essa especificação são constrangidos apenas nos detalhes que afetam a visibilidade externa do seu sistema; os detalhes internos da programação podem ser definidos de acordo com as caracteristicas da máquina onde ela é implementada ou com o gosto do implementador.

Ao lado dessa vantagem significativa, a seção 3 evidenciou algumas limitações da especificação de implementação que devem ser cuidadosamente ponderadas:

1. O usuário que implementa o protocolo de uma camada deve também implementar os protocolos das demais camadas (ao menos das camadas inferiores);
2. A capacidade de verificação dos cenários de testes é bastante limitada.

Por essas razões, a especificação de implementação pode estender o seu escopo, contendo informações que permitam não apenas a intercomunicação das implementações, mas também garantam que a implementação seja facilmente transportável de um sistema a outro e que ela seja correta (isto é, equivalente ao autômato que a define) por construção. Nesse caso a especificação de implementação é dita tranportável.

A especificação transportável oferece melhor divisão do trabalho entre as partes envolvidas, permitindo que o trabalho de desenvolvimento realizado para uma camada possa ser utilizado por outros implementadores sem esforço exagerado. Além disso, a

qualidade do trabalho importado fica, ao menos parcialmente, assegurada pelo método utilizado.

O problema da elaboração desse tipo de implementação pode ser dividido em dois aspectos, que serão discutidos nos ítens subseqüentes.

1. Como desenvolver a implementação do protocolo de forma independente de linguagem de implementação e de máquina objeto, porém de forma suficientemente precisa para permitir a sua utilização por outros programas.
2. Como gerar sequências de testes que, a partir de um autômato conhecido (não mais uma caixa preta) permitam excitá-lo em todos os seus estados, de modo a percorrer todo o código que o implementa.

4.1. Transportabilidade de Implmentações

O problema de transportabilidade de implementações de protocolo, é um caso particular de transportabilidade de programas. O interesse de transplantar entidades que implementam protocolos para máquinas e sistemas operacionais diferentes daqueles para os quais foram originalmente concebidos, justifica-se pela complexidade de desenvolvimento dessas entidades e pela possibilidade de integrar numa única arquitetura, entidades de protocolo de diferentes camadas criadas por implementadores diversos.

Além dessas vantagens, pelo seu maior formalismo e precisão, a especificação transportável pode servir de base para a elaboração de uma implementação de referência que é um elemento vital num ambiente de testes de protocolo. Sem uma implementação de referência cuja operação seja precisamente conhecida, os projetistas de protocolo não conhecerão contra o quê sua implementação será testada.

Programas que implementam protocolos são elementos relativamente fáceis de transportar devido à nítida divisão aí existente entre a parte do programa que define o comportamento do protocolo (uma máquina de estados finitos) e a parte que acessa o sistema operacional da máquina objeto e as entidades de protocolo das camadas adjacentes.

4.1.1. Especificações Transportáveis e Máquinas de Estados Finitos

A parte do protocolo correspondente à máquina de estados finitos (MEF) deve ser equivalente em todas as implementações compatíveis e pode, portanto, ser escrita numa linguagem de alto nível facilmente transportável (Modula-2, Pascal, C, etc.). Essa definição da MEF deve interfacear com o sistema operacional através de um conjunto de rotinas, unicamente definidas que possam ser facilmente implementadas nas máquinas objeto para as quais se pretende importar a MEF.

Esse conjunto de rotinas pode ser classificado em três grandes funções:

- Rotinas de controle de processos e de comunicação entre as camadas;
- Rotinas de gerenciamento de memória;
- Rotinas de temporização.

A comunicação entre camadas adjacentes pode ser modelada por troca de mensagens entre processos. Dado que esses processos podem ter sido implementados em diferentes linguagens é importante que a formatação dessas mensagens seja

independente dos tipos de dados existentes nas linguagens de alto nível. Para tanto o conteúdo das mensagens pode ser definido utilizando uma sintaxe abstrata de dados como a ASN.1 da ISO (16).

Finalmente é importante observar que se a especificação de implementação deve servir de base para a obtenção de uma implementação de referência ela deve ser obtida a partir da especificação do protocolo através de um método correto, pois sendo a implementação de referência um componente do sistema de testes, ela não pode ser testada por esse sistema.

A MEF, utilizando chamadas a um sistema operacional virtual, incluindo formatos abstratos de comunicação entre as camadas e obtida por um método correto, é definida como a especificação de implementação de um protocolo. A Figura 4 resume as etapas necessárias para a obtenção dessa especificação.

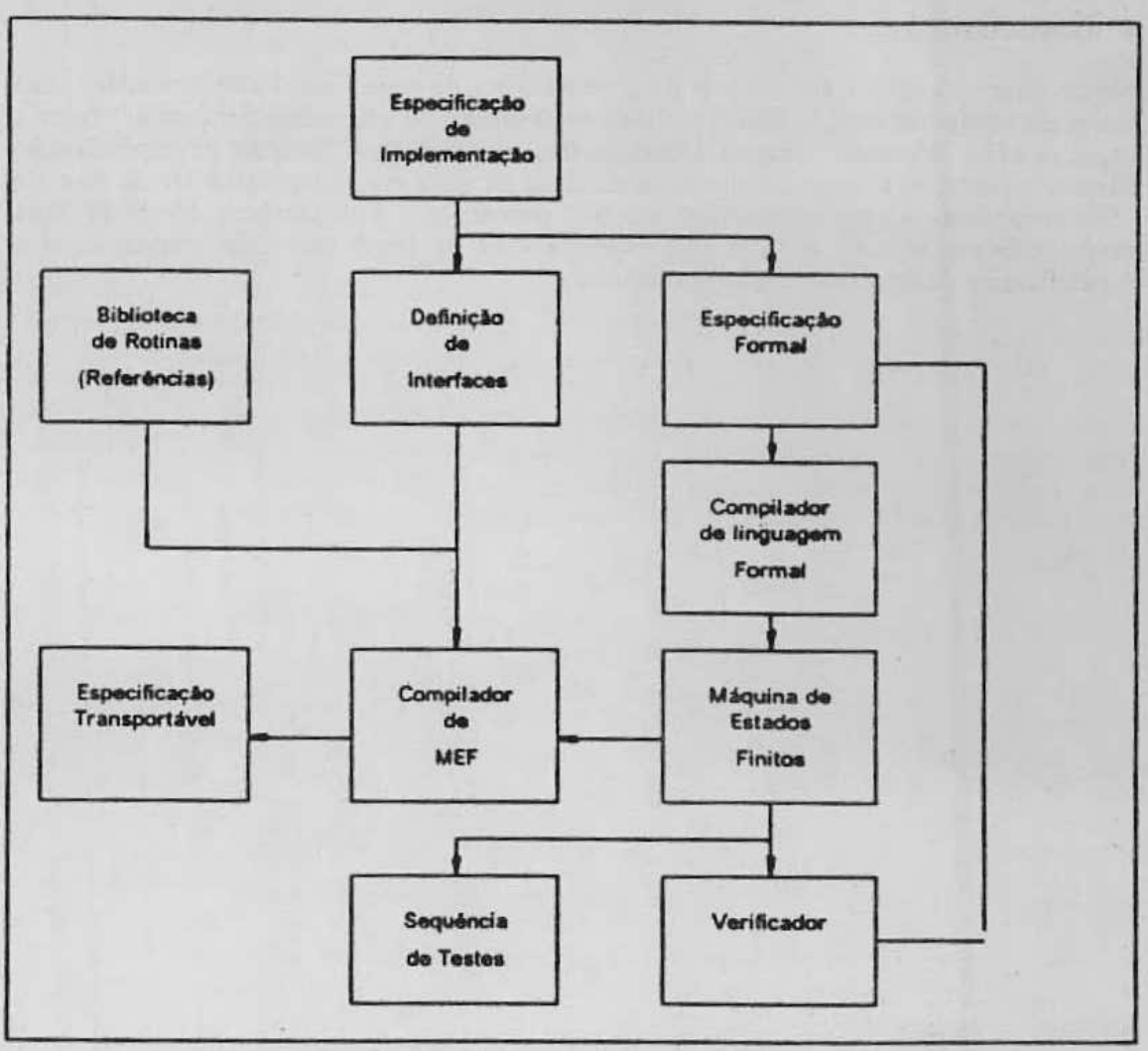

Figura 4 - Etapas da elaboração da especificação de implementação transportável.

4.2. Testes de Implementação

A partir da máquina de estados finitos, que pode ser obtida da especificação formal em qualquer linguagem (Estelle, Lotos, SDL, etc.) é possível obter, automaticamente, cenários de testes que permitam avaliar a compatibilidade de realizações da especificação de implementação (transportável ou não). Diversos métodos foram propostos para essa finalidade (17,18,19).

Esses métodos baseiám-se numa MEF não estendida, razão pela qual ela deve ser gerada a partir da especificação formal, por uma ferramenta automática. Além disso o uso dessa ferramenta permite que protocolos especificados em diferentes linguagens possam ser testados dentro da nossa metodologia.

5. CONCLUSÕES

Neste artigo foi feita a abordagem da problemática de especificação de protocolos para redes de computadores, considerando-se as soluções da especificação formal versus a especificação informal. Foram mencionadas as principais técnicas de especificação formal conhecidas e foi mostrada a necessidade de uma especificação informal, no ciclo "Desenvolvimento/Implementação" de um protocolo. Foi também oferecida uma orientação em relação ao que seja especificação de implementação convencional e especificação de implementação transportável.

6. REFERÊNCIAS

1. CFIP '88 - "Actes du Colloque Francophone sur l'Ingénerie des Protocoles", Bordeaux, França, 22-27 setembro, 1988

2. Paz,S.M. et al, "Experiências em Implementação de Protocolos para Redes de Computadores" - 5° SEMISH, Campinas, 1983

3. Stiubiener, Stefania "Especificação Formal, Verificação e Testes de Protocolos para Redes de Computadores" - 2° Encontro sobre Padronização de Protocolos, Organizado pela Secretaria Especial de Informática, Campinas, 1985

4. Ruggiero, Wilson V. "Programa de Trabalho da BRISA" - 4° Seminário de Normalização Técnica e Qualidade Industrial em Informática", ABNT, 26-28 setembro 1988

5. IBM, GA27-3004 "Binary Synchronous Communications", outubro 1970

6. CCITT X.25 "Interface Between Data Terminal Equipment (DTE) and Data Circuit Terminating Equipment (DCE) For Terminals Operating In The Packet Mode On Public Data Networks", Malaga-Torremolinos, 1984

7. ISO IS8073 "Connection Oriented Transport Protocol Specification", 1986

8. Benayon da Silva, Solon, "Uma Experiência em Validação Automática de Protocolos Aplicada a Redes Locais", Dissertação de Mestrado, PUC/RJ, 1986

9. ISO DP 8074 "Estelle: A Formal Description Technique Based On An Extended State Transition Model", 1986

10. ISO DP 8807 "Lotos - A Formal Description Technique Based On The Temporal Ordering Of Observational Behaviour", 1986

11. CCITT SDL 2.101, "Basic SDL", Genebra, Suiça, maio 1983

12. Hollingum, Jack, "The MAP Report", IFS Publication Ltd. - Bedford, England, 1986

13. Associação Brasileira de Normas Técnicas - ABNT "Especificação da Arquitetura da Rede de Transferência Eletrônica de Fundos", 1988

14. ISO DIS 8571 - File Transfer, Access And Management, 1986

15. ISO DIS 8802/2 "Logical Link Control", 1987

16. ISO DIS 8824 "Information Processing - Open Systems Interconnection - Specification Of Abstract Syntax Notation One (ASN.1)"

17. Naito, S., Tsunoyama, M. "Fault Detection For Sequential Machines By Transition Tour", IEEE Trans. on Software Engineering, 1981

18. Chow, T.S., "Testing Software Design Modeled By Finite State Machines", IEEE Transactions On Software Engineering, 1985

19. Gorec, G. "A Method For Design Of Fault Detection Experiment", IEEE Trans. on Computers, junho 1970