PUC

Série: Monografias em Ciência da Computação

Nº 16/77

ESPECIFICAÇÃO DO PROJETO COMCOM2

por

Arndt von Staa

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rua Marquês de São Vicente 225 — ZC 19

Rio de Janeiro — Brasil

Série: Monografias em Ciência da Computação

Nº 16/77

Editor da Série: Michael F. Challis

Outubro, 1977

ESPECIFICAÇÃO DO PROJETO COMCOM2*

por

Arndt von Staa

* Este trabalho foi financiado pelo Conselho Nacional do Desenvolvimento Científico e Tecnológico (CNPq), contrato nº 2222-0067/77 e pela Financiadora de Estudos e Projetos (FINEP), contrato 307/CT

Para obter cópias dirija-se a:

Rosane T.L. Castilho
Chefe, Setor Doc. Inf.
Deptº de Informática - PUC-RJ
Rua Marquês de São Vicente, 209 - Gávea
20000 - Rio de Janeiro - RJ
Brasil

Resumo:

É definido o sistema de auxílio à construção de compiladores e/ou pré-processadores COMCOM2. São caracterizados o hardware e software ambientais ao sistema COM OM2. São definidos os padrões de documentação e os principais documentos a serem produzidos durante a elaboração do projeto do sistema COMCOM2. São definidos os padrões mínimos de programação e teste de programas desenvolvidos neste projeto.

Palavras Chave:

Construção de Compiladores; Compilador de Compiladores; COMCOM2; Padrão de Documentação; Padrão de Programação; Padrão de Teste.

Abstract:

The compiler and/or pre-processor writing system COMCOM2 is described. The hardware and software environment for this system is caracterized. The documentation, programming and test standards are specified.

Key Words:

Compiler Construction; Compiler Compiler; COMCOM2; Documentation Standards; Programming Standards; Test Standards.

SUMÁRIO

1 Introdução.

Com o projeto COMCOM2 visamos criar um ferramental útil para a especificação de compiladores e de pré-processadores. Este ferramental é fundamental para a condução de experimentos de desenvolvimento de linguagens de acesso a bancos de dados, de linguagens de especificação de programas e de linguagens de processamento simbólico que vem sendo conduzidos no Departamento de Informática da PUC/RJ.

A especificação atual do sistema COMCOM2 é uma evolução de um compilador de compiladores - COMCOM - escrito em 1970 (v. Staa 70). Este sistema demonstrou sua utilidade em diversos projetos (Furtado 70, Azeredo 72). O sistema COM OM original possuia, porém, diversos erros de concepção e tinha restrições que reduziam sua aplicabilidade geral. Pior que isto, o sistema COMCOM original havia sido escrito em Assembler 7044 e tornou-se inútil com a desativação do computador IBM 7044 outrora instalado no Rio Datacentro da PUC/RJ.

Com base na experiência do sistema COMCOM original, e com base na evolução do estado da arte em ciência da computação, propomos a criação de um novo sistema - COMCOM2 - que seja moderno, flexível e de aplicabilidade geral. Como subproduto deste esfôrço almejamos, também, experimentar técnicas de desenvolvimento de programas e de técnicas de documentação.

2 Especificação do Projeto COMCOM2.

O sistema COMCOM2 constará de um conjunto de rot'nas e sistemas geradores de rotinas que visarão dar apoio à construção de compiladores e/ou pré-processadores. São seguintes os componentes do sistema COMCOM2:

1- Um sistema para a geração de co-rotinas que perfaçam a filtragem léxica, acrescidas de funções particulares fornecidas pelo usuário. Os filtros léxicos serão gerados a partir de especificações dadas pelo usuário;

2- Um sistema para a geração de co-rotinas que perfaçam a transdução sintática, baseada em linguagens do tipo SLR(1), acrescidas de rotinas de análise semântica fornecidas pelo usuário. Estas co-rotinas produzirão árvores sintáticas reduzidas. Os transdutores serão gerados a partir de especificações dadas pelo usuário;

3- Um sistema para a geração de co-rotinas que perfaçam a otimização de programas através da manipulação de árvores reduzidas. A otimização será efetuada a partir de especificações dadas pelo usuário;

4- Um conjunto de rotinas para a geração de código a partir de árvores sintáticas reduzidas, acrescido de rotinas fornecidas pelo usuário;

5- Um conjunto de rotinas para suporte ao pós-processamento da fase de compilação;

6- Um conjunto de rotinas para suporte aos programas em tempo de execução;

7- Um conjunto de rotinas para suporte ao pós-processamento da fase de execução.

Partes do COMCOM2 carecerão de pesquisas e/ou de criação de protótipos para o levantamento de características de comportamento. Os resultados destas pesquisas e/ou criação de protótipos serão parte da documentação do projeto COMCOM2.

3 Ambiente de Hardware e Software.

O sistema COMCOM2 será desenvolvido tendo por base o computador central instalado no RDC-PUC/RJ, um IBM/370-165. Inicialmente o sistema COMCOM2 será desenvolvido para um ambiente de programação em PL/1F. Futuramente será revisto e possivelmente adaptado a outro ambiente de programação.

Sistemas geradores poderão produzir programas (co-rotinas) em assembler (ou LKED) e poderão estar escritos em linguagens quaisquer. Porém, o ambiente de execução dos programas gerados e das bibliotecas de subrotinas é o da linguagem PL/1F.

A escolha deste ambiente deve-se ao fato de PL/1 possuir amplas facilidades de manuseio de informação e ser a linguagem utilizada com maior freqüência pelo corpo discente do Departamento de Informática. Em se tratando o COM OM2 um projeto experimental, é de importância maior haver um rápido retorno quanto à sua utilidade do que gerar compiladores altamente eficientes.

4 Documentação.

O desenvolvimento do sistema COM OM2 gerará diversos documentos. O uso desta documentação será facilitado através de três publicações de suporte:z(1) Rede, (2) Sumário e (3) Índice.

Além dessas publicações serão produzidos os documentos (4) Planejamento e, para cada subsistema, (5) Especificação do Subsistema, (6) Manual do Programa e (7) Manual do Usuário.

Algumas destas publicações são dinâmicas no sentido de sofrerem alterações e/ou acréscimos à medida que o projeto for evoluindo.

A seguir daremos uma vista geral do conteúdo de cada um dos documentos a serem produzidos.

4.1 Rede de Documentação do COMCOM2.

Este documento tem por objetivo demonstrar o interrelacionamento e a precedência de cada um dos documentos que compõe o resultado do sistema COMCOM2. Serão relacionadas também outras publicações (livros, teses, monografias) que tenham fornecido subsídios diretos e/ou sejam subprodutos do desenvolvimento do sistema COMCOM2.

A rede de documentação é um documento dinâmico, sofrendo revisões sempre que novas publicações forem anexadas ao conjunto documentos do COMCOM2. Ela é organizada sob a forma de um grafo, onde os nodos são os documentos, ou conjuntos de documentos, e as arestas definem o relacionamento de precedência.

4.2 Sumário do COMCOM2.

O Sumário do COMCOM2 tem por objetivo permitir o discernimento rápido quanto ao conteúdo de cada documento produzido durante a elaboração do COM OM2. Este documento visa tornar mais rápido o acesso à informação desejada por pessoas não familiarizadas com o projeto como um todo.

A organização deste documento é a de "folha solta" e sua atualização será feita sempre que novos documentos forem incorporados à coletânea de documentos do COMCOM2.

4.4 Planejamento do COMCOM2.

O objetivo do documento "Planejamento do COMCOM2" é o de fornecer informações quanto às tarefas já realizadas e por serem realizadas, permitindo uma rápida verificação do andamento do projeto.

Serão definidas metas intermediárias (pontos de controle) que constarão da apresentação de resultados específicos.

4.5 Manual de Especificação.

Para cada programa ou subsistema será fornecido um Manual de Especificação. Este manual tem por objetivo definir precisamente o que deverá ser feito pelo programa ou subsistema, sem porém entrar em detalhes quanto à sua implementação (como é feito).

As interfaces entre subsistemas terão sua especificação feita em separado como se fossem subsistemas autônomos.

Cada Manual de Especificação conterá:

a- objetivo da interface, programa ou subsistema

b- teoria de apoio

c- Características operacionais, tempos de resposta, volumes e nomes de arquivos

d- dados de entrada - sintaxe e semântica

e- resultados produzidos - listagens, organização e estrutura dos resultados

f- Índice.

4.6 Manual de Programa.

Para cada programa ou subsistema será produzido um manual de programa. O objetivo deste manual é dar informação sobre como o programa ou subsistema foi implementado.

O manual de programa constará de:

a- objetivo do programa ou subsistema

b- descrição das estruturas de dados internas utilizadas (tipos abstratos)

c- propriedades de desempenho dos algoritmos utilizados (estatísticas, gráficos, simulações)

d- listagem comentada do programa

e- mapas de teste (comprovação do plano de testes)

f- dados e resultados dos casos teste executados.

4.7 Manual de Operação.

Para cada programa ou subsistema será produzido um manual de operação. O objetivo deste manual é de dar informações quanto à implantação e uso do programa ou subsistema.

O manual conterá:

a- objetivo do programa ou subsistema

b- pontos de entrada ("entry points")

c- pontos externos referenciados ("external", "common")

d- mensagens de erro produzidas

- explicação
- causa
- consequência
- remédio

e- arquivos utilizados: formatos, organização, volumes, unidades.

f- JCL necessário para o uso

g- JCL necessário para a incorporação

h- propriedades de desempenho.

5 Normas de Programação.

Nesta seção descreveremos os padrões de apresentação e de teste a que deverão obedecer todos os programas escritos dentro do projeto COMCOM2.

5.1 Margens.

Exceto quando o programa for escrito em linguagem ASSEMBLER, dever-se-á obedecer às seguintes regras quanto à margem esquerda:

a- Para cada nível de aninhamento a margem esquerda deverá ser deslocada 5 caracteres para a esquerda. Alteram os níveis de aninhamento: DO, FOR, WHILE, IF, CASE, SELECT, PROCEDURE, BEGIN, FUNCTION, SUBROUTINE, etc.

b- Comentários deverão ser alinhados 2 caracters para a esquerda da margem esquerda no nível em que ocorrerem. Os comentários deverão estar precedidos e sucedidos de uma linha em branco.

c- Rótulos ("labels") que não são comentários (definição na próxima seção) deverão ser alinhados 5 caracteres para a esquerda da margem esquerda no nível em que ocorrem e devem estar isolados na linha em que ocorrem.

d- "THEN" e "ELSE" deverão ser colocados 2 caracteres à direita da margem esquerda do "IF" correspondente.

e- "END""s devem ser postos na margem esquerda do nível onde foi iniciado novo bloco ou grupo (IF, DO, PROC, BEGIN etc.) exceto no caso do END que termine um comando composto após um THEN, que deverá ser posto no mesmo alinhamento que o corpo do THEN, i.e. 5 caracteres para a direita do IF correspondente.

Na figura 5.1 apresentamos um exemplo em PL/I que demonstra as regras de alinhamento descritas nesta seção.

```
Y = D:
IF X    Y
  THEN DO;
       A = B;
       C = B;
       END;
  ELSE DO;
       A = D;
       C = D;
END;
X = A + C;
```

Figura 5.1 Exemplo de alinhamentos de margens.

5.2 Comentários.

Existem 4 tipos de comentário: (1) Explanatório, (2) Imperativo, (3) Assertivo e (4) Controle. Comentários deverão vir separados do texto do programa por uma linha em branco antes e após ao comentário.

A seguir caracterizaremos os diversos tipos de comentários e exemplificaremo-los através de trechos escritos em notação PL/1. Para outras linguagens deverão ser feitas as adaptações apropriadas.

O comentário explanatório descreve, em linhas gerais, o que uma porção maior de um programa computará caso venha a ser executada. Este tipo de comentário deverá estar contido num quadro formado por asteriscos . Deverá ser dado um comentário deste gênero no início de cada bloco e/ou procedimento. Ex:

```
/*=========================================*
*    ESTE PROCEDIMENTO COMPUTA A TABELA    *
*!   SLR(1) DE UMA GRAMATICA FORNECIDA     *
*    ATRAVES DO ARQUIVO "SYSIN"            *
*=========================================*/
```

O comentário imperativo determina o que deverá ser computado pelo trecho de programa a seguir O estilo do texto é imperativo e suscinto, ocupando 1 até no máximo 3 linhas.

Comentários imperativos deverão ser fornecidos para cada comando composto ("DO"..."END"), "Loop", "GO TO" e após cada "Label".

Comentários imperativos redundantes com o texto do programa devem ser evitados. Utilizando identificadores explicativos, o programa é capaz de, por si só, servir de explicação em muitos casos.

Comentários assertivos determinam condições que deverão estar satisfeitas no ponto do programa em que ocorrem. O estilo do texto é "lógico". Comentários assertivos poderão vir sucedidos de comentários imperativos. Comentários assertivos podem ser trechos de programa que testem as condições a serem satisfeitas e, mais tarde, serão transformados em comentários.

Comentários assertivos deverão figurar após cada "Label", no início de cada "Loop" (invariante do "loop") após "THEN", "ELSE", itens de "CASE", "SELECT" etc. e após o final de "Loops", "IFs", "CASEs" etc.

Comentários de controle são comentários e/ou trechos de programa contendo indicativos para o compilador e/ou contendo texto de programa para fins de depuração.

Para cada segmento de programa compilado independentemente, o primeiro registro deve ser um comentário de controle contendo o título do segmento a seguir.

Cada comando composto ("DO;".. "END;") e "Loop" deverá ser precedido de um rótulo ("Label") iniciando com 3 caracteres extraídos do nome do procedimento que o engloba e seguido de tantos dígitos quantos são os níveis de aninhamento. Estes dígitos dão o número de ordem do grupo no nível de aninhamento. Antes de "Loops" estes rótulos devem ser sucedidos da letra "L", após "THEN" sucedidos por "V" e após "ELSE" sucedidos por "F". O "END" que termina um conjunto de comandos deve ser sucedido do "label" do início do conjunto de comandos. Estes rótulos nunca figurarão em comandos "GO TO".

Para cada estrutura de informação complexa deverá ser escrita uma subrotina que imprima esta estrutura de maneira legível. Posteriormente, estas rotinas serão transformadas em comentários. Cada linha dessas rotinas de impressão deverá ser escrita na forma:

```
/==/      comando                    /= =/
```

Desta forma torna-se fácil transformar trechos de programa de depuração em comentários, através da eliminação automática da segunda barra ("/") no segmento "/==/".

O primeiro comando útil de cada segmento deverá ser precedido de um comando que imprima o nome do procedimento, bem como os parametros recebidos por este procedimento. Posteriormente, este, ou estes, comandos também serão transformados em comentários.

5.3 Testes

Os testes dos programas serão conduzidos em três níveis: (1) testes de segmentos; (2) testes de programas e (3) testes de sistema.

Os segmentos deverão ser testados de forma a permitir que cada aresta no grafo de controle do segmento tenha sido executada pelo menos uma vez no conjunto de testes (Huang 1975). No caso de programas dirigidos por tabela, deve-se garantir que toda a tabela tenha sido percorrida na totalidade dos testes.

Os programas deverão ser testados de tal forma que todas as mensagens sejam produzidas pelo menos uma vez, e que todos os procedimentos que compõe o programa tenham sido executados pelo menos uma vez na totalidade dos testes do programa.

O sistema deverá ser testado de tal forma que as opções significativas dos dados (máximo, médio, mínimo, todos os elementos de uma lista denumerada, alternativas, repetições, etc.) sejam exercitadas pelo menos uma vez na totalidade dos testes. Além disso, todos os itens de dados devem ser testados fora de seus valores permissíveis pelo menos uma vez na totalidade dos testes.

Prevê-se casos em que o teste do sistema obedecendo às especificações acima torne-se dispendioso em demasia. Nestes casos será feita uma seleção explícita de um subconjunto de testes considerado satisfatório. Em documento adicional será descrito este subconjunto e as opções que não foram testadas.

Referências.

Azeredo, P.A.
Otimização de Código Objeto e suas Aplicações em um Compilador para a Linguagem BASIC Usando um Compilador de Compiladores; Tese de Mestrado, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro; 1972

Bauer, F. L.; Eickel, J.
Compiler Construction: An Advanced Course; Springer Verlag; 1975

Furtado, A.L.
PUCMAT - A Programming Module for Arithmetic Pattern Matching, Monographs in Computer Science no. 2/71, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro; 1971

Huang, J.C.
"An Approach to Program Testing"; em Computing Surveys Association for Computing Machinery 7(3); September 1975; pp 113 - 128

Staa, A. v.
COMCOM - Compilador de Compiladores; Grupo de Aplicações, Rio Datacentro, Pontifícia Universidade Católica do Rio de Janeiro; 1970