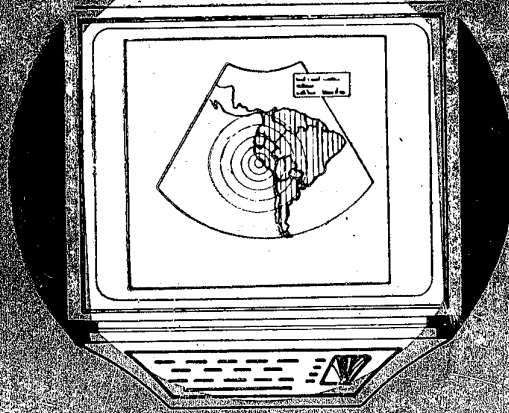


**PANEL  
INFO'82**



# **IX CONFERENCIA LATINOAMERICANA DE INFORMATICA**

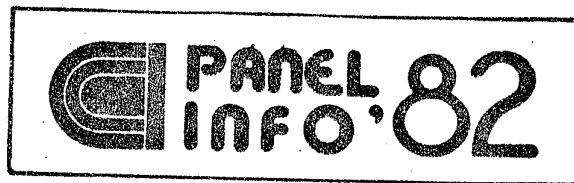
**Centro Latinoamericano de Estudios de Informática (CLEI)  
Asociación Peruana de Computación e Informática (APCI)  
Universidad Nacional de Ingeniería (UNI)**



**16 AL 20 DE AGOSTO DE 1,982  
LIMA - PERU**

**III  
Convención Peruana  
de  
Computación e  
Informática**

004.06  
C748  
V.1



**Anales de la  
IX Conferencia Latinoamericana  
de Informática**

**Centro Latinoamericano de Estudios de Informática (CLEI)**

**y**

**III Convención Peruana de Informática**

Organizado por:

**Asociación Peruana de Computación é Informática (APCI)  
Universidad Nacional de Ingeniería (UNI)**

Patrocinado por:

**Oficina Intergubernamental para la Informática (IBI)**

**Federación Internacional del Proceso de la Información (IFIP)**

**UNESCO**

16 al 20 de Agosto de 1,982

LIMA - PERU

Michael Stanton

Departamento de Informática  
Pontifícia Universidade Católica

Rio de Janeiro-Brasil

## 1. INTRODUÇÃO

No livro, "The Architecture of Concurrent Programs" [BH77,p192] Brinch Hansen discutiu o problema da estruturação de programas concorrentes, e propôs um roteiro para identificar uma estrutura ideal, em termos de atividades simultâneas e estruturas de dados. Mais recentemente Gentleman [GEN81] ressaltou a importância de identificar formas restritas e úteis de estruturar programas concorrentes, para que estas formas pudessem ser usadas rotineiramente como blocos básicos para a montagem de estruturas maiores. Gentleman ainda enfatizou o papel importante para a estruturação de programas concorrentes de uma visão antropomórfica, fazendo uma analogia entre conjuntos de processadores assíncronos e organizações de pessoas. Esta analogia ajuda escolher quais deverão ser as atividades paralelas e qual é a natureza das interações entre estas num programa concorrente.

Além de discussões sobre a estrutura abstrata de programas concorrentes, existem divergências quanto a sua realização. Há dois principais modelos, que são usados para combinar os conceitos de paralelismo, sincronização e comunicação, e que definem um estilo de programação. O mais conhecido utiliza monitores para controlar acesso a variáveis compartilhadas [BH77, HOA74, LAM80, WIR77], enquanto o segundo modelo utiliza o movimento explícito de dados na forma de troca de mensagens [BH70, BH79, HOA78, GEN81]. Este segundo modelo tem vantagens evidentes para um ambiente de memória distribuída, embora para implementação concorrente num ambiente de memória global qualquer dos dois modelos seja válido. Num estudo importante [LAU79], Lauer e Needham demonstraram que sob certas restrições existe uma dualidade entre os dois modelos, e que um programa realizado segundo um modelo tem seu correspondente direto segundo o outro.

O objetivo deste artigo é mostrar como a dualidade apontada por Lauer e Needham pode ser utilizada como ferramenta na estruturação de programas concorrentes, por possibilitar o projeto de uma estrutura usando um dos dois estilos mencionados, e realizá-la usando o estilo dual. Em particular, daremos exemplos de programas que foram projetados usando troca de mensagens e depois realizados usando monitores.

## 2. A DUALIDADE ENTRE OS DOIS MODELOS

Lauer e Needham [LAU79] fizeram uma comparação entre a estrutura de programas feitos usando os dois estilos de monitores, e de troca de mensagens. Como existe diversidade na especificação de troca de mensagens, resumimos aqui as assumpções feitas por Lauer e Needham. Seu modelo admite a existência de diversos p $\acute{o}$ rticos que possam receber mensagens. O remetente envia uma mensagem a um p $\acute{o}$ rtico e depois pode esperar a resposta desta mensagem. O recipiente, que tamb $\acute{e}$ m  $\acute{e}$  o dono dos p $\acute{o}$ rticos, pode esperar receber uma mensagem atrav $\acute{e}$ s de qualquer elemento num subconjunto dos p $\acute{o}$ rticos, que pode ser alterado dinamicamente. Esta alteraç $\tilde{a}$ o din $\tilde{a}$ mica permite atrasar arbitrariamente a recepç $\tilde{a}$ o de mensagens dirigidas a um dado p $\acute{o}$ rtico. Convencionalmente neste modelo, uma mensagem  $\acute{e}$  respondida antes da recepç $\tilde{a}$ o da pr $\acute{o}$ xima mensagem. A relaç $\tilde{a}$ o entre o remetente e o recipiente  $\acute{e}$  assim $\acute{e}$ trica, e o recipiente desconhece a identidade do remetente. Devemos notar ainda que quando um remetente se bloqueia esperando a resposta, este bloqueio ocorre (logicamente) na fila de mensagens no p $\acute{o}$ rtico do recipiente, e portanto antes da recepç $\tilde{a}$ o da mensagem.

Com estas assumpç $\tilde{o}$ es, Lauer e Needham concluíram que os dois estilos de programaç $\tilde{a}$ o usando monitores e troca de mensagens s $\tilde{a}$ o duais sob a transformaç $\tilde{a}$ o

processo	←-----→	mensagem
monitor	←-----→	processo
chamar/retornar	←-----→	enviar/responder

Isto implica que a cada programa, escrito em termos de processos e monitores, corresponde um programa dual, escrito em termos da troca de mensagens entre processos.  $\acute{E}$  interessante observar que a mesma palavra processo tradicionalmente tem sido usada com dois significados bem diferentes, como revelado pela transformaç $\tilde{a}$ o de dualidade. Esta observaç $\tilde{a}$ o talvez explique certa relut $\tilde{a}$ ncia contra a aceitaç $\tilde{a}$ o da tese de Lauer e Needham. Gentleman [GEN81], por exemplo, ao defender programaç $\tilde{a}$ o baseada na troca de mensagens, negou a relev $\tilde{a}$ ncia desta tese. Por $\acute{e}$ m pretendemos mostrar que o cont $\acute{e}$ udo do seu artigo n $\tilde{a}$ o suporta esta conclus $\tilde{a}$ o.

O modelo de troca de mensagens de Gentleman tem algumas diferenç $\tilde{a}$ s daquele assumido por Lauer e Needham. Primeiro, embora continue assim $\acute{e}$ trica a relaç $\tilde{a}$ o entre remetente e recipiente, o recipiente passa a conhecer a identidade do remetente, para poder responder  $\tilde{a}$ s mensagens em ordem arbitr $\tilde{a}$ ria. Segundo, a comunicaç $\tilde{a}$ o requer a sincronizaç $\tilde{a}$ o de ambas as partes, e logo, para n $\tilde{a}$ o bloquear nunca o recipiente, o envio de uma mensagem bloqueia o remetente at $\acute{e}$  a chegada da resposta. Terceiro, em vez de manter filas de mensagens para p $\acute{o}$ rticos diferentes, o recipiente recebe todas as mensagens na ordem de sua chegada. Em particular  $\acute{e}$  imposs $\tilde{i}$ vel bloquear a recepç $\tilde{a}$ o de mensagens de uma determinada fonte, e imposs $\tilde{i}$ vel bloquear mensagens de um determinado tipo. Devemos observar aqui que estas caracter $\tilde{i}$ sticas s $\tilde{a}$ o as mesmas que tem um monitor, que recebe chamadas das suas entradas em ordem de chegada, mas n $\tilde{a}$ o precisa retornar controle na mesma ordem, podendo bloquear a execuç $\tilde{a}$ o de uma entrada em qualquer ponto atrav $\acute{e}$ s do comando espera. Esta execuç $\tilde{a}$ o seria retoma

da mais tarde quando for conveniente. Na sua discussão, Lauer e Needham reconhece que esta característica de um monitor não tem correspondente no seu modelo de troca de mensagens. Na opinião deste autor, o modelo de Gentleman corresponde neste ponto. Embora um programa no estilo de Gentleman não seja transformado para uso de monitores por troca de certas palavras chave (LAU79 pl4), a transformação não é muito mais complexa. Em particular nos exemplos de administradores dados no artigo de Gentleman, o comando receber é sempre seguido de um comando de casos, que substitui o comando de casos de porticos de Lauer e Needham, e cujos braços correspondem às entradas de um monitor. Estamos portanto dispostos a reconhecer dualidade entre programas usando monitores e programas usando troca de mensagens no estilo de Gentleman.

A maneira mais vívida de ver a dualidade entre os estilos é de usar uma notação comum. Em particular pretendemos padronizar uma notação gráfica que representa as relações estruturais entre os componentes de um programa concorrente. O exemplo da fig. 1 é tirado da página 195 de BH77.

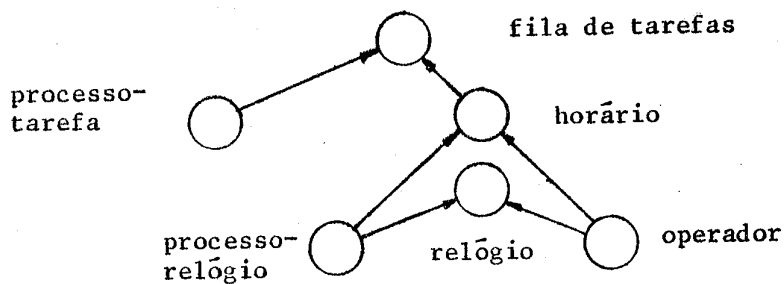


Figura 1: uma representação gráfica de uma estrutura concorrente.

Brinch Hansen [BH77] utiliza esta notação da figura 1 com o seguinte significado. Cada círculo representa ou um processo ou um monitor. Cada seta representa "direitos de acesso", ou seja o direito de chamar entradas do monitor apontado pela seta. Monitores então são distinguidos por serem apontados por pelo menos dois outros componentes. Propomos adotar esta mesma notação para o modelo usando troca de mensagens. Neste caso valerá a interpretação a seguir. Cada círculo representa um processo. Cada seta representa uma relação de comunicação assimétrica, com a cabeça apontando o recipiente das mensagens, que deverá respondê-las. (Por coincidência, ou não, Gentleman [GEN81] utiliza exatamente esta notação). Com estas convenções, podemos estabelecer a estrutura de um programa concorrente utilizando um ou outro modelo. O resultado da análise será um gráfico da estrutura, que agora possa ser interpretado de acordo com o modelo mais conveniente para fins de realização.

### 3. DUALIDADE COMO FERRAMENTA

No final da última seção sugerimos uma notação gráfica para programas concorrentes que incorpora a noção de dualidade. Daqui falta um passo pequeno para o resultado principal deste trabalho - o reconhecimento da dualidade como ferramenta poderosa para projetar um programa que será implementado usando um dos modelos aqui descritos. O primeiro passo é produzir um gráfico da estrutura na nossa notação. Deve-se observar que o raciocínio que gera este gráfico poderá usar características do modelo atual para chegar à estrutura desejada. A implementação segue direto de uma interpretação do gráfico em termos do modelo usada na implementação. Ou seja, o projeto pode ser feito em termos de um modelo, e imple-

mentado usando o outro. Aliás foi esta descoberta no contexto da implementação de um canal de comunicação duplex (veja na próxima secção) que motivou este trabalho.

Outra consequência de dualidade é a possibilidade de aproveitar os esforços de outros autores que trabalhavam em termos do modelo dual. Por exemplo, o administrador e os trabalhadores de Gentleman, desenvolvidos em termos da troca de mensagens [GEN81], têm seus correspondentes diretos em termos de monitores. Os gráficos de estrutura apresentados no seu artigo precisa meramente de interpretação segundo o outro modelo.

#### 4. UM EXEMPLO

Na implementação do protocolo X.25, Farran [FAR81A] empregou o modelo de monitores [FAR81B]. Parte do programa envolve a implementação de um canal duplex para transmissão de pacotes com controle de fluxo. A estrutura (simplificada) adotada é mostrada na figura 2.

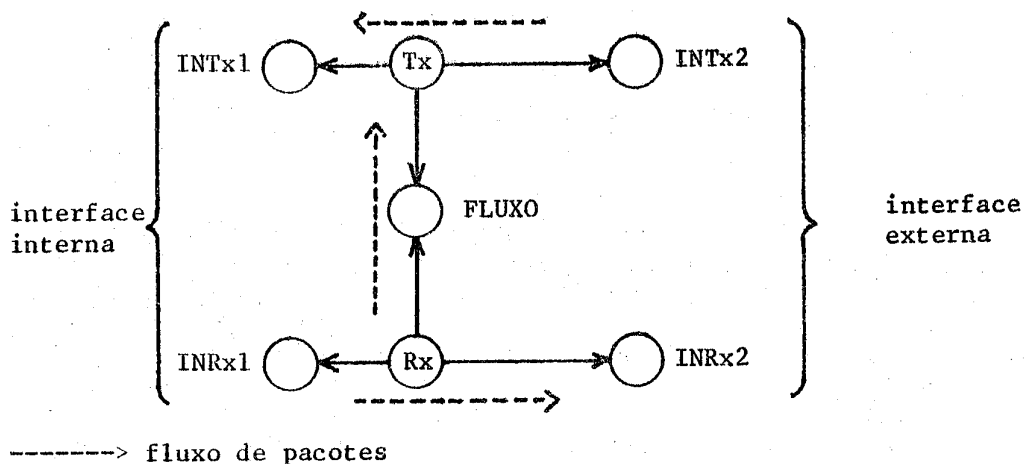


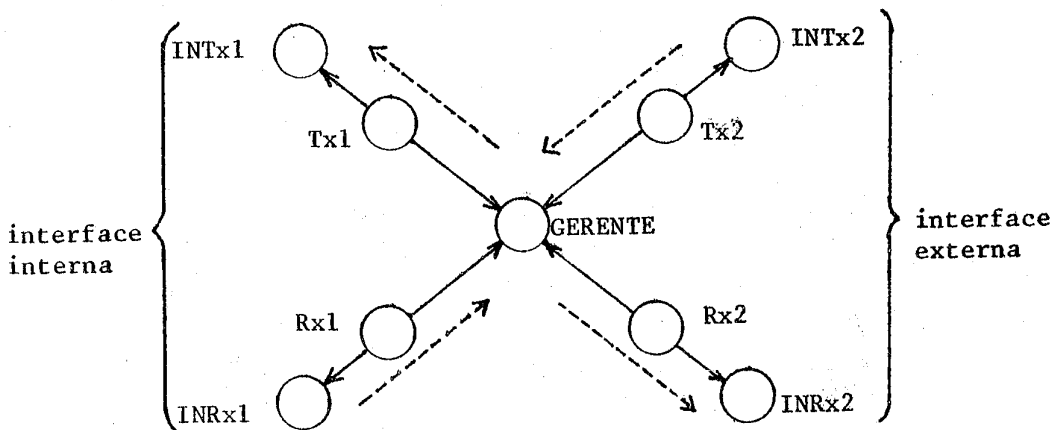
Figura 2: canal duplex de comunicação de pacotes (primeira versão)

Neste gráfico incompleto, as interfaces deste subestrutura com as vizinhas se faz através dos monitores INTx1, INRx1, INTx2 e INRx2. O monitor FLUXO é usado para controlar o fluxo de pacotes que passa pelo canal. Os processos Rx e Tx são responsáveis respectivamente para recepção e transmissão relativas à interface interna. Quando foi elaborada esta estrutura, o seguinte defeito foi reconhecido - o processo Tx, responsável para transmissão busca seu pacote ou de INTx2 ou de FLUXO. Se não tiver um pacote num destes monitores, o processo espera neste monitor, apesar da existência de um pacote no outro monitor, com conseqüente perda de paralelismo, e, logo, desempenho.

A solução correta foi percebida somente depois de conhecido o estudo de Lauer e Needham. Foi tentada uma solução em termos da troca de mensagens. Na verdade isto combina muito bem com a aplicação, por tratar de transmissão de pacotes. A figura 3 mostra a nova estrutura.

Deve-se notar que o complexo formado pelos componentes Rx, Tx e FLUXO agora foi substituído pelo que Gentleman chama de um administrador (GERENTE) e quatro correios (Rx1, Rx2, Tx1 e Tx2) para o envio de mensagens. Estes correios não processam os pacotes transmitidos; todo processamento é centralizado em GERENTE.

Uma vez conhecida a estrutura, sua implementação em termos de monitores se torna fácil.



-----> fluxo de pacotes

Figura 3: canal duplex de comunicação de pacotes (versão final)

## 5. CONCLUSÕES

No uso da ferramenta descrita aqui não há nenhuma consideração objetiva para preferir um ou outro dos modelos para a análise que gera o gráfico da estrutura. Por outro lado, para uma atividade intuitiva como projetar estruturas concorrentes, as considerações subjetivas são muito importantes. Este autor está convencido do que o modelo de troca de mensagens oferece mais apoio à intuição, devido ao seu maior grau de antropomorfismo, quando comparada com seu dual - organizações humanas funcionam na base da troca hierarquizada de mensagens. Gentleman [GEN81] também apoia esta visão.

## 6. AGRADECIMENTOS

Este trabalho resultou de longas discussões com Yussef Farran e Paulo Medeiros sobre a implementação de protocolos de comunicação. O autor agradece o apoio financeiro da FINEP, e a Reynaldo Lima pelos desenhos.

## 7. REFERÊNCIAS

- [BH70] Brinch Hansen, P., "The nucleus of a multiprogramming system", CACM 13 (4), 238-250, 1970.
- [BH77] Brinch Hansen, P., "The architecture of concurrent programs", Prentice-Hall, Englewood Cliffs, New Jersey, 1977.
- [FAR81A] Farran L., Y.E., "Especificação de uma implementação da recomendação X.25 do CCITT para um sistema IBM/370", tese de mestrado, PUC/RJ, 1981.
- [FAR81B] Farran L., Y.E. e Stanton, M.A., "Programação Concorrente usando monitores - uma implementação", Primeira Conferência Internacional em Ciência da Computação, Santiago, Chile, agosto de 1981.
- [GEN81] Gentleman, W.M., "Message passing between sequential processes: the reply primitive and the administrator concept", Software - Practice and Experience, 11(5), 435-466, 1981.

- [HOA74] Hoare, C.A.R., "Monitors: an operating system structuring concept", CACM 17(10), 549-557, 1974.
- [HOA78] Hoare, C.A.R., "Communicating sequential processes", CACM 21(8), 666-677, 1978.
- [LAM80] Lampson, B.W. e Redell, D.D., "Experience with processes and monitors in Mesa", CACM 23(2), 105-117, 1980.
- [LAU79] Lauer, H.E. e Needham, R.M., "On the duality of operating system structures", Second International Symposium on Operating Systems, IRIA, Rocquencourt, France, outubro de 1978, re impresso em Operating Systems Review 13(2), 3-19, 1979.
- [WIR77] Wirth, N., "Modula, a language for modular multiprogramming", Software - Practice and Experience 7(1), 3-35, 1977.

#### RESUMO

O objetivo principal deste trabalho é demonstrar a grande utilidade da dualidade entre o uso de monitores, e o uso de troca de mensagens na construção de programas concorrentes. Dualidade é vista como a ferramenta que permite o projetista de um programa concorrente escolher o estilo mais útil para facilitar sua compreensão e intuição, com a finalidade de escolher uma estrutura. A estrutura resultante, por dualidade, poderá ser transformada para o outro estilo para implementação. É mostrada a aplicação da ferramenta a uns exemplos.