Abstract:

A utilização de linguagens de especificação de muito alto nível na análise de sistemas informáticos é analisada levando em linha de conta os resultados teóricos e práticos dos principais projectos de investigação e desenvolvimento no campo. Para alem duma breve descrição da evolução nos últimos vinte anos deste ramo de investigação agora em grande actividade, os principais objectivos dessa investigação são indicados, bem como os possíveis resultados práticos Em particular, são expostos os principais requisitos duma linguagem de especificação e as linguagens desenvolvidas até ao momento são avaliadas face a esses requisitos.

Palavras chave

Analise de sistemas/ Linguagens de especificação / Bases de dados / linguagens de muito alto nível / Verificação da correcção

(1) Este trabalho foi realizado com o suporte financeiro da Fundação Calouste Gulbenkian e da Universidade de Lisboa

1 – INTRODUÇÃO

A ideia duma linguagem de muito alto nível para especificação de sistemas informáticos não é de maneira nenhuma nova. Os primeiros passos foram dados ainda nos anos cinquenta (YOUNG(1958)) mas os primeiros resultados dignos de atenção foram obtidos no âmbito do CODASYL Development Comittee uns anos depois (BOSAK (1962)). Neste ultimo artigo é reconhecida a necessidade duma linguagem não procedimental que permita a especificação dos sistemas informáticos duma maneira tão afastada quanto possível das restrições impostas ao nível da implementação. Os autores desse artigo vão mais longe e propõem uma linguagem (Information Algebra ou. simplesmente, IA) que introduz conceitos ao tempo revolucionários. Por exemplo IA é baseada numa semântica (informal) declarativa em termos de asserções e fundamenta-se num modelo do sistema informático muito próximo do modelo relacional das bases de dados.

Como seria de esperar o impacto destas propostas foi mínimo por alguns anos. Só nos anos setenta voltou a comunidade científica e profissional a interessar-se de novo pelo assunto isto acabou por acontecer pelas mais variadas razões. Primeiro, os analistas de sistemas nunca deixaram verdadeiramente de procurar um instrumento formal de trabalho (fosse uma linguagem ou não) por toda a década de sessenta (para um panorama deste desenvolvimento veja-
-se COUGER(1974)). Alguns dos projectos nascidos nesse contexto atingiram a maturidade no dealbar dos anos setenta (entre outros SYSTEMATICS (GRINDLEY(1972)) e PSL (TEICHROEW(1971))). Segundo, o advento da teoria e dos sistemas de bases de dados desencadeou o desenvolvimento de novas linguagens de especificação possíveis apenas devido às facilidades proporcionadas por esses sistemas (para citar apenas algumas, STREMA (CLARK(1977)), LEGOL (STAMPER (1978)) e SBA (ZLOOF (1977))). Terceiro as técnicas utilizadas neste campo foram usadas e desenvolvidas em projectos na Área da chamada "geração automática de programas" com base em linguagens. dirigidas ao utilizador final (as quais podem ser consideradas linguagens de especificação pois retem muitas das características destas, as quais são analisadas em detalhe na próxima secção). Os exemplos mais marcantes deste grupo são as linguagens SBA (já referida) e BDL (HAMM£R(1977)), ambas nascidas em centros de investigação da IBM. Quarto, a complexidade dos sistemas informáticos a implementar não tem parado de crescer, em particular devido a multiplicação e distribuição dos dados a processar. Por esta razão, a lógica dos sistemas informáticos é cada vez mais complexa pelo que se tornou imperioso introduzir instrumentos efectivos para a sua especificação e verificação de correcção. A utilização de linguagens de especificação e/ou desenvolvimento da implementação tem sida proposta aos mais diferentes níveis e nos tais diferentes campos (veja-se por exemplo YEH(1977) para uma breve introdução ao problema da validação de programas, área de pesquisa intensiva desde os trabalhos originais sobre o assunto (FLOYD(1967) e HOARE(1969))). Finalmente, recentes desenvolvimentos na area das linguagens de programação conducentes às chamadas linguagens de programação de muito alto nível vieram demonstrar a praticabilidade de muitos dos conceitos em germinação na área das linguagens de

especificação. Em particular no que se refere às capacidades de estruturação e abstracção, o benefício foi enorme. Para citar apenas dois dos exemplos mais flagrantes, as linguagens de especificação importaram o conceito de conjunto de processos concorrentes e o conceito de tipo abstracto de dados. Por outro lado, o desenvolvimento destes conceitos na área das linguagens de especificação levou a introdução de tipos temporais abstractos de dados e também à definição duma semântica declarativa (não executiva) dos processos concorrentes (em SERNADAS (1980 b,e)).

A terminar esta breve introdução ao problema da especificação dos sistemas informáticos torna-se necessário tratar uma linha (difusa) de demarcação entre programação e especificação. Hoje em dia, a demarcação feita em SCHWARTZ(1973) é ainda a mais aceite: “Programação é optimização e especificação é no que a programação se torna quando esquecemos a optimização e programamos na maneira apropriada a uma maquina infinitamente rápida com memória infinita"

Posto o problema deste modo, alguns interrogar-se-ão sobre o real interesse duma especificação. Isto é, se a optimização e esquecida o que resta? Estudos empírico têm demonstrado que 9O% do trabalho de concepção dum sistema informático diz respeito à lógica do problema (a qual é independente dos detalhes de implementação) - veja-se por exemplo WATERS(1979a). Efectivamente a grande maioria das opções a tomar (mesmo as referentes a procedimentos de restabelecimento) podem e devem ser feitas ao nível da analise de
sistemas e nas ao nível da implementação. Quanto a maioria dos outros são originados por especificações incompletas e/ou defeituosas (veja-se ALBERT (1976)) e quando o preço da correcção dum erro e tanto maior quanto mais tarde ele e detectado ao longo do ciclo de vida do sistema (ate 100 vezes maior de acordo com um
estudo da IBM citado em WATERS(1979b)), torna-te imperioso encontrar meios mais efectivos de especificação (e sua verificação) dos sistemas informáticos.
Voltando ao problema da demarcação programação/especificação. e útil recordar o já clássico quadro:

ESPECIFICAÇÃO

o quê
Não procedimental

Independente da im plementação

Colecção de asserções

Perspectiva do utilizador

Análise de sistemas

Programação

Como

procedimental
Definição da implementação
Sequência de comandos
Perspectiva do computador
Implementação

Outras diferenças de índole mais técnica serão indicadas na próxima secção. O que convém reter deste quadro pode resumir-se no seguinte principio:

Uma linguagem de especificação e um instrumento de trabalho do analista de sistemas que a usa para construir as regras de evolução (asserções) que o sistema pretendido deve satisfazer ao nível do entendimento da organização por parte dos utilizadores.

Note-se que o interesse das linguagens de especificação não esta de maneira nenhuma restrito ao campo das aplicações comerciais.

Por exemplo a sua utilidade para a concepção de "hardware" deve ser evidente (uma vez que os erros da

implementação serão ainda mais dispendiosos); No entanto. a análise ainda que breve desse problema sai completamente do âmbito deste artigo.

2 - PERFIL DUMA LINGUAGEM DE ESPECIFICAÇÃO

No âmbito restrito deste curto artigo não é possível fazer uma analise detalhada dos requisitos para uma linguagem de especificação minimamente eficaz. No entanto, é perfeitamente viável sumarizar esses requisitos e, ao mesmo tempo, indicar quais as linguagens que os satisfazem. Deste modo é possível dar uma panorâmica do estado actual da pesquisa no campo das linguagens de especificação objectivo primordial deste artigo.

Para alem das linguagens de especificação já citadas (com indicação de bibliografia) outras serão referidas oportunamente. A respectiva bibliografia será indicada aquando da sua primeira referência. Note-se que para cada linguagem só o trabalho mais importante e/ou fácil de obter é indicado. Para uma bibliografia exaustiva veja-se SERNADAS(l980e).

Convém também sublinhar que o perfil a desenvolver diz respeito sobretudo na especificação da dinâmica dos sistemas informáticos, isto é, a especificação das regras de evolução do sistema informático pretendido, área onde o autor tem desenvolvido o seu trabalho de investigação nos últimos dois anos. Uma vez que esse problema e o mais profundo e o que tem atraído mais atenção nos últimos anos a analise desenvolvida é facilmente justificável. De qualquer modo, as referências minimamente necessárias serão sempre feitas sobre um ou outro aspecto menos explorado.

2.1 Modelo adoptado (dos sistemas informáticos)

Toda a linguagem de especificação é baseada num determinado modelo dos sistemas informáticos. Em particular, esse modelo inclui um submodelo da componente de memorização (base de dados) e outro dos elementos funcionais (processos) responsáveis pela evolução do sistema. A relevância da teoria dos modelos de dados (da teoria das bases de dados) é evidente. Adoptado um modelo da base de dados (por exemplo. o relacional (CODD(1970))) resta escolher um
modelo para as componentes funcionais.

O modelo fornece os objectos básicos com que as especificações são construídas (tal como os elementos geométricos são usados pelos arquitectos para especificar edifícios).

Dois tipos de modelo estão hoje em voga:

- um baseado em mensagens e em processos de acordo com o qual o sistema informático é decomposto num reservatório de mensagens ("message pool") e numa colecção de processos concorrentes (veja-se RIDDiE (1979) e SERNADAS(1980a));
- o outro de natureza substantiva de acordo com o qual o sistema informático é especificado como parte integrante do sistema objecto que controla (veja-se STANPER(1979)).

Em qualquer dos casos o modelo adoptado deve satisfazer os seguintes requisitos:

2.1.1 - Independência da implementação;

2.1.2 - Enquadramento ao nível do entendimento dos utilizadores.

A maior parte das linguagens de especificação propostas ate hoje satisfaz minimamente estes requisitos. embora na maioria dos casos o modelo adoptado não seja, introduzido explicitamente. Os modelos mais cuidadosamente desenvolvidos em cada um dos dois grupos acima são respectivamente o modelo associado ao INFOLOG (SERNADAS(1980a) e ao LEGOL (já referido).

Como consequência destes requisitos outros tem de ser impostos os quais são oportunamente introduzidos na discussão que se segue.

2.2 - Exaustividade

A linguagem de especificação deve providenciar instrumentos capazes para a especificação exaustiva de todos os elementos do sistema informático. incluindo:

2.2.1 - A estrutura estática do sistema;

2.2.2 - A dinâmica do sistema;

2.2.3 - Os volumes do sistema;

2.2.4 - Os formatos dos documentos/mensagens trocados com o exterior.

Em WATERS(1979a) o interessado pode encontrar um estudo detalhado destes requisitos e uma avaliação dos métodos existentes de especificação e documentação no que lhes diz respeito.

E de sublinhar que algumas das linguagens de especificação propostas não cobrem todos os aspectos indicados. só aquelas que já atingiram a fase de implementação (discutida mais abaixo) tentam ser exaustivas. Na verdade, os elementos 2.2.3 e 4 não levantam os problemas de maior e são muitas vezes desprezados em projectos de investigação (pelo menos na fase inicial).

2.3 - Independência da estrutura de dados:

Naturalmente, a linguagem de especificação deve permitir a criação e modificação de especificações dum modo tão independente quanto possível da eventual estrutura de dados e ser escolhida para implementar a base de dados do sistema. As linguagens de especificação mais recentes adoptam o modelo relacional da base de dados (CODD(1970)) garantindo o necessário grau de independência dessas estruturas. Assim, SYSTEMATICS, STREMA, LEGOL, SBA e INFOLOG satisfazem este requisito. sem qualquer problema.

2.4- independência da organização de memória ("memory independence")

A especificação do sistema informático deve ser realizada independentemente da representação escolhida para o passado no estado presente do sistema. Na verdade, só o implementador deve escolher entre guardar uma mensagem ou reconstrui-la (a partir de outras) se e quando necessário. Este conceito inicialmente esboçado em GRINDLE (1972) foi formalizado em SERNADAS(1980b) com a introdução do conceito de base do dados temporal. Até ao momento apenas as linguagem SYSTEMATICS, PSL, LEGOL e INFOLOG satisfaçam este requisito.

2.5 - Poder de expressão

Dum ponto de vista teórico. um dos aspectos menos desenvolvidos das linguagens de especificação foi por muito tempo o do seu podar de expressão. Só com o desenvolvimento da teoria que levou ao INFOLOG foi este problema resolvido (SERNADAS(1980e) no enquadramento da teoria da computabilidade.

Vários aspectos do poder de expressão tem de ser examinados:

2.5.1 - Funcional
2.5.2 - Temporal
2.5.3 - Dinâmica
2.5.4 - Objectos primitivos

Nenhuma das linguagens de especificação é completa sob qualquer destes pontos de vista, com a excepção do INFOLOG que foi desenvolvido com estes objectivos em vista. Para uma discussão destes conceitos sua definição formal e avaliação das outras linguagens de especificação veja-se SERNADAS(1980e).
Na pratica isto significa que é sempre possível encontrar uma aplicação impossível de especificar com a linguagem incompleta escolhida.

Acidentalmente. a teoria desenvolvida é também aplicável às linguagens de consulta de bases de dados. O critério obtido revela-se melhor que o proposto em CODD(1972) e formalizado em BANCILHON(1978).

2.6 - Estruturação

Uma linguagem de especificação aceitável deve permitir a estruturação das especificações, a fim de aumentar a sua legibilidade e de facilitar a localização de possíveis erros, para já não falar na diminuição dos custos de possíveis modificações.

Dois tipos de estruturação são possíveis

2.6.1 - Estruturação vertical

Este tipo de estruturação exige a possibilidade de dividir a especificação em níveis de detalhe definidos em cima uns dos outros quer se progrida de baixo para cima ou de cima para baixo. Isto pressupõe a possibilidade de definição de novos objectos sempre que necessário eventualmente usando a aglomeração permitida por tipos ou tipos abstractos. Neste campo poucas linguagens atingem o grau de excelência minimamente necessário. Linguagens como HOS (HAMILTON(1976)) e DELTA (HOLBAEK-HANSEN (1977)) possuem já algumas das características desejadas, mas só o INFOLOG com o completo suporte de definições recursivas e de tipos abstractos temporais ou não satisfaz na totalidade os requisitos em questão.

2.6.2 - Estruturação horizontal

Esta estruturação é realizada decompondo o sistema em vários processos concorrentes que só interactuam através da base de dados do sistema. Num sentido restrito, a linguagem BDL e também a linguagem DELTA suportam este tipo de estruturação. Por outro lado, INFOLOG foi desenvolvido directamente sobre o conceito de colecção de processos concorrentes.

Note-se que a sincronização de componentes paralelas é fonte de muitos dos problemas dos sistemas actuais (compostos de inúmeros programas cuja interacção não é sempre clara pois não é formalmente definida). Usando uma linguagem de especificação capaz de definir processos concorrentes é possível explicitar as relações existentes entre os diferentes componentes do sistema.

2.7 - Estilo assertivo

Uma especificação deve ser uma colecção de asserções (regras) com uma semântica declarativa, no sentido de que cada regra deve poder ser interpretada como verdadeira ou falsa. Naturalmente, a pragmática de tais regras é a da satisfação: o sistema especificado é suposto s atisfazer todas as regras (isto é, é suposto tornar verdadeiras todas as regras).

Várias razões levam a este requisito. Primeiro, a maior parte das especificações tem este estilo (veja-se o caso das leis). Segundo, assim as especificações são mais legíveis (não há que levar em linha de conta um estado implícito de execução). Terceiro, tais regras podem ser encaradas como axiomas num sistema formal de demonstração (útil para a verificação da correcção de especificação e suas implementações).

Embora este requisito tenha sido identificado há quase 20 anos (em BOSAK((1962)), tem-se revelado muito difícil de satisfazer. Na verdade, só com o INFOLOG apareceu a teoria necessária á definição formal da semântica declarativa dum grupo de asserções. Todas as outras linguagens de especificação ficam muito aquém deste objectivo.

A imposição desta semântica declarativa (em termos de asserções) implica que a linguagem seja necessariamente muito não procedimental (qualquer que seja o critério usado). A analise detalhada deste ponto sai no entanto fora do âmbito deste curto artigo.

2.8 - Equivalência das duas semânticas

Uma vez que a noção de processo (actividade correspondente a execução dum programa) implica a consideração duma semântica executiva em termos de modificações a fazer em cada instante á base de

dados e, por outro lado o estilo assertivo implica a consideração duma semântica declarativa em termos de axiomas a serem satisfeitos
pelo sistema, torna-se necessário provar a equivalência das duas semânticas sob pena de incorrer em graves erros.

Acidentalmente a demonstração de tal equivalência leva a uma teoria declarativa dos processos concorrentes (de grande utilidade em campos não directamente relacionados com a presente discussão).

A linguagem INFOLOG é a única cuja semântica foi definida formalmente alternativamente em termos declarativos e em termos executivos. Mais ainda. a equivalência destas duas semânticas foi também provada formalmente (veja-se SERNADAS(1980e)).

2.9 - Sistema formal de demonstração

Toda a linguagem de especificação deve ser associada a um formalismo de demonstração o qual é absolutamente necessário para verificar a correcção de especificações e implementações. Embora tal requisito
tenha sido identificado há mais de 10 anos (nos trabalhos já referidos de Floyd e Hoare), nenhuma das linguagens de especificação até agora propostas o satisfaz, com a excepção do INFOLOG para o qual isso foi particularmente simples de conseguir graças á sua semântica declarativa.

2.10 - Metodologia de analise de sistemas

E evidente que uma linguagem de especificação definida à volta dum particular modelo do sistema informático deve ser associada a uma metodologia de analise de sistemas (análise de dados + análise funcional) desenvolvida com base no referido modelo. Tal metodologia serviria para guiar o analista de sistemas durante a especificação dum sistema informático de modo a tirar o máximo proveito da linguagem.

Até ao momento, só SISTEMATICS, LEGOL, DELTA e HOS se aproximam da satisfação deste requisito, embora algumas das outras linguagens (ainda em desenvolvimento) o possam conseguir num futuro mais ou menos próximo.

2.11 - Implementação

Uma linguagem de especificação só pode ser Implementada no âmbito dum sistema CASA (de "computer-Aided Systems Analysis") ou no âmbito muito limitado dum sistema do geração automática de programas (a partir das especificações). Só a primeira implementação interessa ao analista de sistema como instrumento de trabalho. Varias das linguagens de especificação já referidas são utilizadas na pratica com a ajuda dum sistema CASA. Esse é o caso da linguagem PSL (com cerca de 40000 Instalações por todo o mundo) que é sem duvida a mais popular. Outra linguagem de características semelhantes às da PSL foi implementada recentemente (ALFORD (1977) e DAVIS(1977)) para utilização no Departamento da Defesa dos E.U.A. Outras implementações de nível académico de linguagens mais avançadas são referidas em SERNADAS(1980e).

Um sistema CASA deve permitir a criação, modificação, anulação e verificação (usando as mais variadas técnicas) das especificações e suas implementações. Entre outros módulos deve incluir um interpretador um editor, um simulador, um emulador e um gerador de dados para teste. Para mais detalhes veja-se SERNADAS(1980). Uma parte das funções do sistema CASA corresponde às funções dum dicionário de dados, mas os dois não devem ser confundidos pois o sistema CASA é muito mais complexo uma vez que também analisa a dinâmica do sistema informático.

Entretanto, com os custos de programação a aumentarem e os custos de computação a diminuírem, não está longe o tempo em que a geração automática de implementações de rotina se torne banal. Este passo é tão certo como o foi o salto do nível máquina para o nível COBOL e o daí para os sistemas de bases de dados usando linguagens de consulta e produção de relatórios de muito alto nível;

3 - CONCLUSÕES

Em termos práticos, a lição a tirar deste breve estudo das linguagens de especificação é bem simples:

a) O analista de sistemas envolvido na concepção de aplicações não triviais não pode ignorar os desenvolvimentos nesta área. Na verdade, se não é possível adoptar um produto acabado (linguagem + sistema CASA) com as características desejáveis (ou porque ainda em desenvolvimento ou porque se já implementado é muito limitado), há toda a vantagem em tirar proveito das novas técnicas e conceitos que vão aparecendo a fim de melhorar a metodologia pessoal de analise de sistemas. A utilização de modelos independentes da implementação, de tipos abstractos, de processos concorrentes e de formalismos de verificação da correcção é francamente recomendada (do mesmo modo que aqueles que ainda não dispõem dum SGBD tentam utilizar os princípios da teoria das bases de dados a fim de organizarem melhor os seus ficheiros clássicos).

b) A adopção imediata dum produto acabado só deve ser encarada com a maior das reservas pois o campo está ainda a sofrer uma evolução muito rápida e os produtos já existentes no mercado deixam muito a desejar (em geral porque são construídos à volta de linguagens de especificação manifestamente inadequadas)

c) É inevitável que haja uma transferência do esforço de concepção da fase de implementação para a fase de analise à medida que instrumentos mais sofisticados de analise de sistemas forem introduzidos. Isto significa que no futuro mais ou menos próximo vão ser necessários menos recursos humanos ao nível da programação de rotina, mas, por outro lado vão aumentar as necessidades ao nível da concepção única dos sistemas. As opções respeitantes ao recrutamento e formação de pessoal devem já levar em linha de conta este facto, una vez que não é dum momento para o outro que se transforma um bom programador num especificador aceitável