DESIGN DE AMBIENTES COMPUTACIONAIS PARA MODELAGEM EM UM CONTEXTO EDUCACIONAL

Osvaldo Luiz de Oliveira

UNICAMP
BIBLIOTECA CENTRAL

UNIDADE BC
N.º CHAMADA:
T/UNICAMP
OL3d
V. Ex.
TOMBO BC/ 26168
PROC. 433/95
C [ ] D [X]
PREÇO R$ 11,00
DATA 01/12/95
Nº CPD

CM-00081026-4

FICHA CATALOGRÁFICA ELABORADA PELA

BIBLIOTECA DO IMECC DA UNICAMP

Oliveira, Osvaldo Luiz de

OL3d Design de ambientes computacionais para modelagem em um contexto educacional/ Osvaldo Luiz de Oliveira. -- Campinas, [SP : s.n.], 1995.

Orientador : Maria Cecília Calani Baranauskas.
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Ciência da Computação.

1. Interação Homem-Computador. 2. Ensino auxiliado por computador. 3.* Design (Estruturas para suporte a interação Homem-Computador). 4.* Software para educação. 5.* Modelagem educacional. I. Baranauskas, Maria Cecília Calani.II.Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Ciência da Computação. III. Título.

Tese defendida e aprovada em, 14 de setembro de 1995

Pela Banca Examinadora composta pelos Profs. Drs.

Prof(a). Dr(a). MARIA DAS GRAÇAS VOLPE NUNES

Prof(a). Dr(a). HELOISA VIEIRA DA ROCHA

Prof(a). Dr(a). MARIA CECILIA BARANAUKAS

DESIGN DE AMBIENTES COMPUTACIONAIS PARA MODELAGEM EM UM CONTEXTO EDUCACIONAL

Osvaldo Luiz de Oliveira

Departamento de Ciência da Computação
IMECC - UNICAMP

Banca Examinadora:

- Maria Cecília Calani Baranauskas (Orientadora)[1]
- Maria das Graças Volpe Nunes[2]
- Heloísa Vieira da Rocha[3]
- Eliane Martins[4]

[1] Professora do Departamento de Ciência da Computação - IMECC - UNICAMP.
[2] Professora do Departamento de Computação - ICMSC - USP São Carlos.
[3] Professora do Departamento de Ciência da Computação - IMECC - UNICAMP.
[4] Professora do Departamento de Ciência da Computação - IMECC - UNICAMP.

Para:

minha esposa Ana Cristina e
meus irmãos Edgard Antônio, Luiz Antônio, Maria Célia e Paulo Cézar.

AGRADECIMENTOS

Fui inspirado e influenciado por tantas pessoas ao longo destes anos que tentar relacionar todas a quanto sou grato seria um convite a erros e omissões. No entanto, para conclusão deste trabalho agradeço a algumas pessoas em especial:

à Ana Cristina por ter-me encorajado frente a numerosas contrariedades, pelo incentivo durante tanto tempo e pela compreensão da minha ausência nos momentos em que me dedicava a este trabalho;

ao Luisinho Carneiro por ter-me indicado o Departamento de Ciência da Computação da UNICAMP;

ao Paulo Centoducatte, Rose, Luisinho Carneiro, Marcos Euzébio, Rogério (Zig), Míriam (Mineira), Zieghard e Vicente pela convivência agradável ao longo destes anos em Campinas;

à Profa. M. Cecília, minha orientadora. Sempre me lembrarei da sua orientação firme, da sua sabedoria, do seu amor pelo que faz, da dedicação a este trabalho e da valorização das minhas idéias;

sou duplamente grato à Subcomissão de Pós-Graduação em Ciência da Computação pela oportunidade de ingresso e pelo meu retorno ao programa de Mestrado em Ciência da Computação depois de vários anos em que estive afastado;

a meu professor de Matemática no segundo grau Silvio Ramos e aos meus professores da Universidade Federal de Viçosa José Luís Braga, José Geraldo Teixeira e Roberto Daledone Machado pela mensagem transmitida e pelo caminho indicado.

aos meus amigos da Divisão de Informática da Universidade São Francisco pelo apoio e estímulo;

ao Luís, cuja lembrança nos trás uma grande saudade. Na secretaria de pós-graduação do DCC sempre tomou o partido da solução e não o do problema, desde o primeiro instante em que nós alunos lhe passávamos nossas dúvidas e dificuldades burocráticas com a UNICAMP.

Agora estou voltando ao meu caminho; agradeço a Deus por isto.

RECONHECIMENTO DE MARCAS COMERCIAIS

CorelDraw é marca registrada da *Corel Corporation.*

Excel, MS-DOS, PaintBrush, Windows e Word for Windows são marcas comerciais da *Microsoft Corporation.*

Harvard Graphics é marca registrada da *Software Publishing Corporation.*

SqlPlus é marca registrada da *Oracle Corporation.*

QMF é marca registrada da *International Business Machine Corporation.*

ToolBook e OpenScript são marcas registradas da *Asymetrix Corporation.*

RESUMO

Software para modelagem tem sido tradicionalmente dirigido a profissionais experientes, com competência formal em matemática. O uso de modelagem baseada no computador no contexto educacional é relativamente recente e tem acontecido a partir do desenvolvimento de teorias de interação Homem-Computador e da evolução na tecnologia de interface. Tendo como pano de fundo as teorias da ação (Norman, 1986) e o construcionismo (Papert, 1986), este trabalho discute os benefícios e desvantagens dos ambientes para modelagem atuais e propõe um conjunto de estratégias para o *design* do que denominamos Ambientes para Modelagem Orientados a um Domínio. Um protótipo de ambiente para modelagem no domínio da Mecânica é desenvolvido para ilustrar os principais aspectos discutidos.

ABSTRACT

Modelling tools have been directed to expert users, with formal competence in mathematics. The use of computer based modelling tools in the educational context is relatively recent and it has been improved with theories of Human-Computer interaction and with evolution in the technology of interfaces. Having as background the theory of action (Norman, 1986) and the constructionism (Papert, 1986), in this work we discuss the benefits and drawbacks of recent modelling environments and we propose strategies for the design of what we term Domain Oriented Modelling Environments. A prototype of an environment in the domain of Mechanics is developed, in order to illustrate the main aspects discussed.

SUMÁRIO

Resumo xiii

Abstract xv

1 - Introdução 1

2 - O estado da arte e a problemática 9

2.1 - De simulação à linguagem de programação 11

2.2 - Ambientes de suporte à modelagem em um contexto educacional 13

2.3 - Discussão e proposta 23

3 - Objetivos 27

4 - Estratégias para o *design* de ambientes para modelagem orientados a um domínio 31

4.1 - Introdução 33

4.2 - Restrição de domínio 34

4.3 - Identificando e escolhendo os conceitos atômicos do ambiente para modelagem 36

4.4 - Desenvolvendo o *design* de cada conceito atômico e do ambiente para modelagem como um todo 41

4.5 - Resumo das estratégias 46

4.6 - As estratégias discutidas diante de alguns métodos tradicionais para análise, projeto e implementação de sistemas 47

5 - Emprego das estratégias propostas ao *design* de um ambiente para modelagem 51

5.1 - Introdução 53

5.2 - Formalismos para representação dos conceitos atômicos 54

5.3 - Os elementos do domínio e a identificação dos conceitos atômicos subjacentes 56

5.4 - Desenvolvimento do *design* de cada conceito atômico 85

5.5 - Especificando a infra-estrutura do ambiente 92

6 - NEWTON: Um protótipo de ambiente para modelagem 97

6.1 - Introdução 99

6.2 - Generalidades do NEWTON 101

6.3 - Desenhando corpos 107

6.4 - Diagramas de comportamento mecânico 114

6.5 - Editando diagramas de comportamento mecânico 121

6.7 - Executando modelos e instalando os instrumentos de *feedback* 126

6.8 - Um exemplo de modelagem com o NEWTON 131

6.9 - Idéias e comentários sobre o trabalho de se criar modelos com o NEWTON 134

6.10 - O suporte dado pelo NEWTON aos estágios da execução e de avaliação de uma ação 138

6.11 - Melhorias 139

7 - O NEWTON do ponto de vista de sua implementação 143

7.1 - Tecnologia empregada na implementação 145

7.2 - Estrutura geral do NEWTON e dos modelos por ele criados 150

7.3 - A edição de corpos, de comportamento mecânico e dos instrumentos de *feedback* 155

7.4 - Diagramas de comportamento mecânico e equações diferenciais 157

7.5 - A execução dos modelos 167

8 - Discussão, conclusões e perspectivas 179

8.1 - Síntese 181

8.2 - O que há nas proximidades do que fizemos 181

8.3 - As estratégias propostas e a modelagem em outros contextos 187

8.4 - Considerações sobre o protótipo 190

8.5 - O quê fizemos e como isto pode ajudar no *design* de ambientes para modelagem 192

Apêndice A - Referência rápida ao uso do NEWTON 195

A.1 - Elementos gerais 197

A.2 - Menus 198

A.3 - Comandos dos menus 200

A.4 - Palhetas 204
A.5 - Barra de ícones 205
A.6 - Elementos básicos sobre a edição de comportamento mecânico 205
A.7 - Instrumentos de *feedback* 212
A.8 - Algumas dicas para se trabalhar com o NEWTON 213

Apêndice B - Alguns modelos criados com o NEWTON 215
B.1 - Cinemática de dois automóveis 217
B.2 - Sistema de amortecimento usado para reduzir o recuo de certos tipos de armas 217
B.3 - Canhão antiaéreo 218
B.4 - Lançamento vertical de um projétil para grandes altitudes 219
B.5 - Impulsão de um bloco sobre uma superfície sem atrito 219
B.6 - Impulsão de um bloco sobre uma superfície com atrito 220
B.7 - Bloco sobre um plano inclinado 221
B.8 - Tubo de raios catódicos 222
B.9 - Uma bola quicando sobre uma superfície plana 223
B.10 - Cravamento de uma estaca 224
B.11 - Viga de dois apoios 225
B.12 - Turbina 226
B.13 - Sistema motor 227

Referências Bibliográficas 229

Capítulo 1

Introdução

Este capítulo discute os termos modelo e modelagem aplicados em um contexto educacional, explicita o significado do tema da dissertação e apresenta uma visão geral do trabalho proposto.

Este trabalho tem como tema principal o *design* de ambientes computacionais para modelagem em um contexto educacional. Preferimos não traduzir o termo *design.* Neste trabalho, ele não é empregado no sentido de uma metodologia para especificação do projeto ("de programa") de ambientes para modelagem mas em um contexto mais amplo, de tal modo que as traduções como desenho, projeto ou concepção não captariam o seu significado. Empregamos o termo *design* no sentido sugerido por Winograd (1995) cujo significado engloba o objeto de *design* , os seus usuários e a situação de uso todas juntas como ponto de partida. Quando dizemos que um objeto de *design* funciona queremos dizer que ele funciona dentro de um contexto de valores e necessidades das pessoas que irão utilizá-lo. Para praticar *design* é preciso pensar e criar estruturas que participem e dêem suporte à inter-relação Homem-Computador. Diferente de um projetista de *software,* um *designer* de *software* deve ultrapassar a barreira das estações de trabalho e conceber estruturas que se adaptem ao pensamento das pessoas que irão usar o *software.*

Desde o primórdio de sua existência, o homem criou modelos na tentativa de explicar e predizer o funcionamento do mundo e na intenção de comunicar um modelo a outra pessoa necessitou expressá-lo segundo algum formalismo. Nos dias atuais, os termos "modelagem" e "modelo" são usados para descrever uma grande variedade de atividades e de objetos. Mesmo na ciência, um **modelo** pode significar desde um aparato físico até uma equação (Millwood e Stevens, 1990). No seu sentido mais geral, **modelo** pode ser definido como uma coleção de elementos formais que obedecem a regras bem definidas (Miller, R. *et alli*, 1993), regras estas propostas com o objetivo de explicar e predizer o funcionamento de algum aspecto do mundo: o domínio do modelo. E **modelagem** é a atividade de criar o modelo a partir do uso de uma linguagem formal.

O processo de modelar um fenômeno real ou hipotético e observar/analisar o comportamento de um modelo é constituído de três fases principais: (1) a construção de um modelo que representa aspectos relevantes do fenômeno que está sendo estudado, (2) execução e análise do modelo construído e (3) comparação entre o modelo construído e o fenômeno em estudo. Podemos chamar de **simulação** à segunda fase desse processo enquanto que a modelagem engloba as três (figura 1.1).

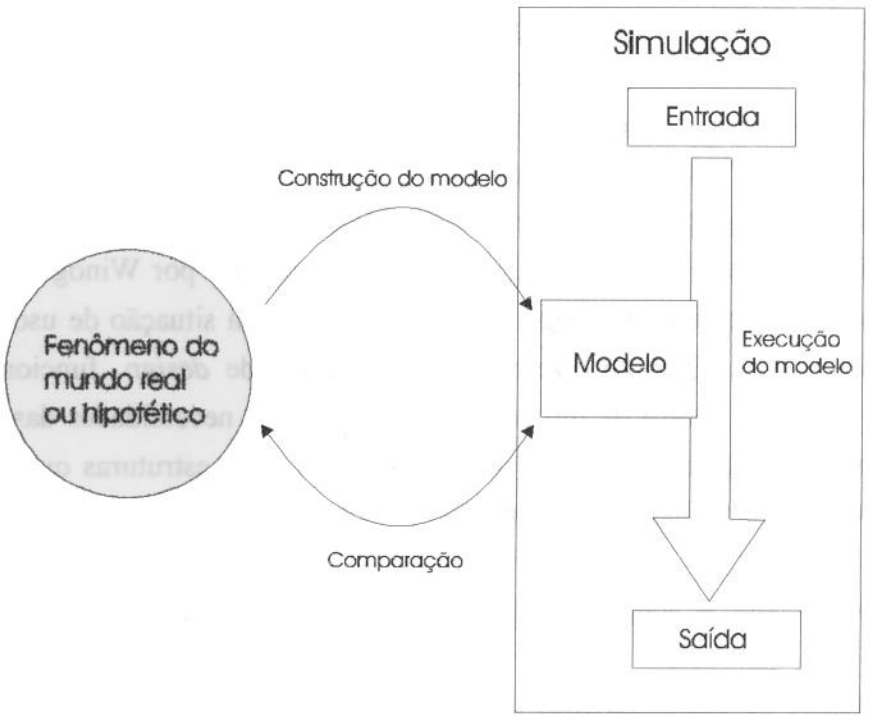


Fig. 1.1: O processo de modelagem

Modelos formais necessitam ser expressos segundo uma linguagem formal. E computadores, dada a sua vocação para manipulação de símbolos, são bons candidatos a ferramentas para se construir e executar modelos (Miller, R. *et alli*, 1991). Além disso, pode-se tirar partido extra do uso de computadores como ferramentas para modelagem: o modelo pode ser construído usando-se, por exemplo, recursos de imagem, animação e som; o computador pode dar suporte à criação do modelo, por exemplo, coibindo construções sintaticamente erradas ou mostrando as primitivas para modelagem disponíveis; dependendo do modelo, é possível que o computador possa executá-lo. Chamamos, portanto, de **modelagem computacional** a atividade de usar o computador para expressar o modelo, com o objetivo subsequente de explorar possíveis conseqüências do modelo e reavaliar a partir de um *feedback*, não apenas o modelo construído, mas o próprio conhecimento sobre o sistema/fenômeno alvo.

Sistemas computacionais para modelagem podem constituir ambientes de aprendizado poderosos, por envolver o aprendiz no ciclo básico de expressão, avaliação e reflexão sobre o domínio considerado. A exigência do computador pela expressão formal de um modelo leva o aprendiz a definir mais precisamente seu conhecimento sobre o assunto. Além disso, a execução do modelo pela máquina possibilita uma avaliação que pode

motivar o aprendiz a questionar o modelo, reavaliar seu conhecimento e, talvez, expressar novamente o modelo, continuando o ciclo de ações ao estilo construcionista[1] de aprendizagem (Papert, 1986).

Tem sido encontrado na literatura (Hassel, 1990; Miller *et alli,* 1991e 1993; Millwood e Stevens, 1990; Riley, 1990; Schecker, 1993) considerável evidência dos benefícios educacionais de ambientes baseados em modelagem. Modelagem tem sido defendida para o desenvolvimento de habilidades como resolução-de-problemas, tomada-de-decisão, apresentação e comunicação do entendimento e do conhecimento que o estudante tem em um dado domínio. Hassell (1990) sugere que modelagem encoraja os estudantes na clarificação de suas idéias, expressando-as de forma concreta.

O uso de modelagem baseada no computador no contexto educacional é relativamente recente e tem acontecido a partir do desenvolvimento de teorias de interação Homem-Computador e da evolução na tecnologia de interface. *Software* para modelagem tem sido tradicionalmente dirigido a profissionais experientes, com competência formal em matemática (Millwood e Stevens, 1990). A proposta de *software* de modelagem para propósitos educacionais desloca o foco de atenção do *designer* para os aspectos de interação usuário-sistema que possibilite ao aprendiz construir, testar e refinar seu modelo, na forma mais concreta possível.

Um dos primeiros sistemas para modelagem projetados para o contexto educacional foi o Dynamic Modelling System-DMS (Riley, 1990; Schecker, 1993). DMS requer a descrição do modelo através de equações e algoritmos numéricos numa linguagem semelhante ao Basic. Stella (LindField, 1992; Riley, 1990; Schecker, 1993), talvez o mais significativo, é um ambiente para exploração de como sistemas dinâmicos funcionam. Em Stella, modelos são um amálgama de diagramas (grafos dirigidos) e fórmulas algébricas. Em um nível mais alto, as estruturas do modelo são descritas por diagramas que mostram como as grandezas envolvidas em um fenômeno se relacionam

[1] Piaget citado em Papert (1980) observou que, desde os primeiros anos de vida, a criança já possui mecanismos de aprendizagem que ela desenvolve sem ser ensinada. A criança aprende conceitos, sem ser explicitamente ensinada, como por exemplo o de **par**, através da interação com objetos de seu mundo: os sapatos, as meias, homem-mulher, pai-mãe, as mãos etc.. Assim, segundo Piaget a criança desenvolve sua capacidade intelectual interagindo com objetos do mundo onde ela vive, fazendo uso de seu próprio mecanismo de aprendizagem. Papert (1986) acrescentou dois novos componentes ao trabalho de Piaget e usou o termo construcionismo para definir um processo de construção do conhecimento através do computador. Primeiro, para Papert, a construção do conhecimento acontece quando o aluno constrói alguma coisa de seu interesse (um desenho, conta uma estória, faz um programa de computador) e para o qual ele está bastante motivado. Trata-se do aprendizado do "fazer". Segundo, por fazer algo de seu interesse e para o qual está motivado, o aluno se envolve afetivamente com o seu trabalho. A amplitude de possibilidades de aplicação destas idéias é possível graças ao computador e segundo Papert (1980) isto se dá pelo fato do computador poder assumir milhares de formas, servir a milhares de finalidades e, assim, poder atrair milhares de gostos.

qualitativamente. Um modelo Stella é construído num ambiente de manipulação direta de objetos através de ícones. Num segundo nível são definidas relações algébricas quantitativas entre as as grandezas envolvidas. Como saída, além de gerar gráficos das variáveis no tempo, gera animação no diagrama do modelo. Extend é um requintado *software,* também, para modelagem de sistemas dinâmicos. Adicionalmente ao Stella, o Extend permite que seu usuário construa novos blocos (as suas primitivas) ou altere o comportamento de blocos predefinidos através de uma linguagem chamada ModL, bastante semelhante à linguagem C. Vários protótipos de novos ambientes vem sendo desenvolvidos recentemente, alguns dirigidos a estudantes mais novos como os sistemas IQON, SMOD, QMOD (Miller *et alli,* 1993) e Expert Builder (Webb, 1994).

Ross (1986) argumenta que ambientes para modelagem exigem um esforço cognitivo do usuário maior do que ambientes para simulação. Ross observa que, além das dificuldades associadas à definição do que vai ser modelado e ao projeto do modelo, as fases de implementação e de testes exigem bastante do usuário. Ao mesmo tempo em que o aprendiz deseja conhecer sobre um determinado domínio, a apropriação das técnicas de programação e de particularidades das linguagens para modelagem geram, no mínimo, dois problemas: aumentam o esforço cognitivo do aprendiz e desviam a sua atenção para aspectos nem sempre relacionados ao domínio de seu interesse.

Todos os ambientes citados para modelagem são versões computadorizadas de algum sistema formal. Por exemplo, o formalismo subjacente ao Q-MOD é a Álgebra; Expert Builder se baseia nos cálculos de predicado de primeira ordem e, como observa Schecker (1993), o Stella foi desenvolvido com base na teoria de sistemas dinâmicos de Forrester (1968). A questão que se coloca é: dado um domínio, como estabelecer um ambiente que permita a modelagem de fenômenos deste domínio? Nossa proposta é contribuir nesta direção.

Baranauskas e Oliveira (1995) observam que a apropriação pelo usuário não sofisticado dos ambientes para modelagem é dificultada pelo grau de generalidade de suas linguagens. As primitivas destes ambientes apresentam um nível de granulosidade diferente das primitivas conceituais utilizadas para descrever fenômenos dentro de um domínio. Ambientes para modelagem de propósito geral, apresentam-se para o aprendiz como um intermediário entre este e o domínio de seu interesse. Ao mesmo tempo em que o aprendiz deve se preocupar com as questões inerentes a um domínio, ele deve desenvolver esforços extras para se apropriar de uma linguagem para modelagem que lhe

permita traduzir os fenômenos de seu interesse (Ross, 1986). A nossa proposta é a de restringir o *design* de ambientes para modelagem a um domínio específico. Fazendo isto, o aprendiz interessado em um domínio comporá os seus modelos através do uso dos próprios elementos do domínio representados no ambiente para modelagem.

Este trabalho discute a modelagem computacional orientada a um domínio e propõe estratégias para o *design* de ambientes para modelagem no contexto educacional tendo como pano de fundo as teorias da ação (Norman, 1986) e o construcionismo (Papert, 1986). Um ambiente para modelagem no domínio da Mecânica é proposto como meio de se verificar a aplicação das estratégias propostas, bem como para viabilizar análises futuras sobre o *design* de ambientes para modelagem.

No capítulo 2 apresentamos os ambientes para modelagem mais significativos, voltados para aplicação em um contexto educacional, que estão referenciados na literatura. Discutimos a problemática que envolve estes ambientes e apresentamos nossa proposta. Formalizamos, no capítulo 3, os objetivos específicos deste trabalho. No capítulo 4 discutimos as bases teóricas para o *design* de ambientes computacionais para modelagem orientados a um domínio. Propomos um conjunto de princípios e os organizamos devidamente no sentido de compor um conjunto de estratégias que possam guiar o *design* dos referidos ambientes. O capítulo 5 exemplifica a aplicação das estratégias propostas ao *design* de um ambiente para modelagem. Os resultados descritos neste capítulo foram usados para desenvolver um protótipo de ambiente para modelagem. Os capítulos 6 e 7 discutem este protótipo segundo dois pontos de vista diferentes. O capítulo 6 descreve a funcionalidade do protótipo segundo a ótica de seu usuário. Já o capítulo 7, mostra o protótipo internamente destacando suas estruturas de dados, alguns de seus algoritmos e a tecnologia utilizada para desenvolvê-lo. Ou seja, este capítulo apresenta o protótipo sob a ótica da sua implementação. Por fim, o capítulo 8 expõe nossas conclusões e apresenta perspectivas de trabalhos futuros. Dois apêndices complementam e podem ser utilizados, principalmente, no transcorrer da leitura dos dois capítulos que tratam do protótipo: o apêndice A que é uma referência rápida ao uso do protótipo e o apêndice B que constitui uma pequena mostra da variedade de modelos que podem ser criados com o protótipo desenvolvido.

Capítulo 2

O Estado da Arte e a Problemática

Pretendemos atingir dois objetivos com este capítulo: (1) discutir a problemática envolvida com os ambientes para modelagem, dirigidos a um contexto educacional, encontrados atualmente na literatura e (2) apresentar nossa proposta de ambiente para modelagem. Para isto apresentamos, inicialmente, os ambientes para modelagem educacional mais significativos descritos na literatura. Após esta apresentação, discutimos a problemática que os envolve e expomos nossa proposta.

Tópicos deste capítulo:

Tópico		*Página*
2.1 -	De simulação à linguagem de programação	11
2.2 -	Ambientes de suporte à modelagem em um contexto educacional	13
2.3 -	Discussão e proposta	23

2.1 - DE SIMULAÇÃO À LINGUAGEM DE PROGRAMAÇÃO

Ambientes computacionais para aprendizagem diferem uns dos outros de várias formas, mas dois fatores de divergência são particularmente importantes: o esforço cognitivo solicitado ao usuário para que ele execute suas ações e o nível de controle embutido no *design* do sistema. Goodyear (1987) observa que, em geral, estes dois fatores relacionam-se em proporção direta, havendo um gradativo aumento do esforço cognitivo do usuário e do nível de controle (do usuário sobre o ambiente) quando consideramos um espaço de aplicações que se estende desde os ambientes puramente de simulação até os ambientes de programação de propósito geral. Ou seja, quanto maior for a autonomia oferecida por um ambiente ao aprendiz maior, também, será o esforço demandado para operá-lo. A figura 2.1 apresenta um espectro de ambientes que se prestam à modelagem e/ou simulação. Este espectro se estende desde ambientes puramente de simulação até as linguagens de programação de propósito geral que também podem ser utilizadas para criar modelos.

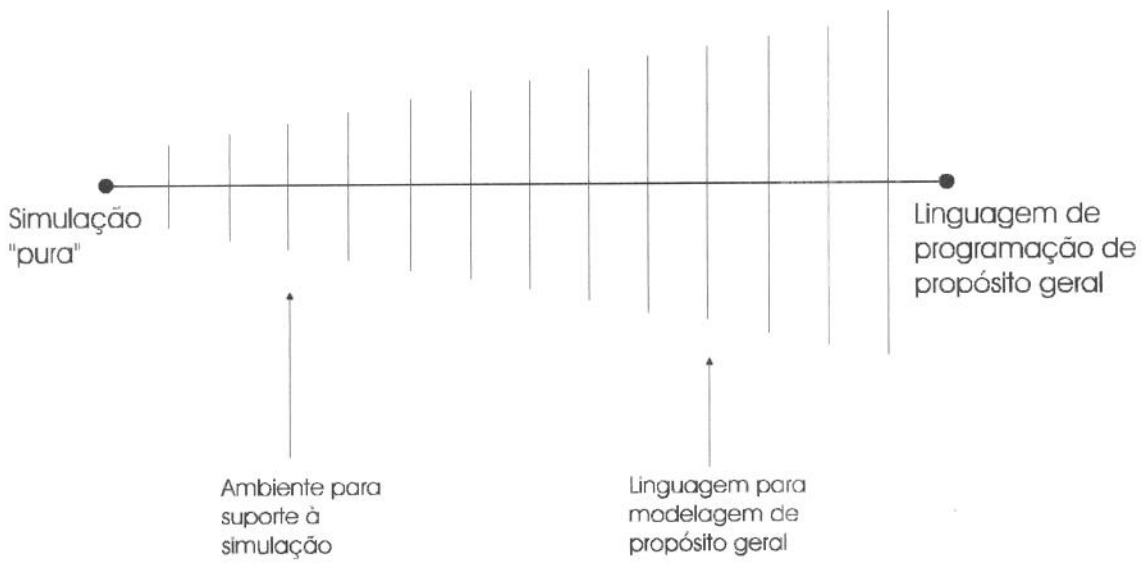


Fig. 2.1: Espectro de ambientes para modelagem e/ou simulação

Em um dos pólos do espectro, a simulação "pura", o usuário está limitado a alterar os parâmetros de entrada e saída de um modelo e observar os resultados de saída. Neste pólo não há atividade de modelagem. Nenhum controle sobre a "lógica" do modelo é dado ao

usuário. Recentemente, com o desenvolvimento de teorias de interação Homem-Computador e com a evolução da tecnologia de interface, tem surgido interesse pelo *design* de ambientes de suporte à simulação aplicados em um contexto educacional. Novos trabalhos colocam mais controle nas mãos do usuário da simulação (Baranauskas e Weller, 1994-b; deJong, 1991).

No pólo do espectro onde se situam as linguagens de programação de propósito geral, a idéia é permitir que o usuário crie a sua própria "simulação" através da construção de modelos que podem ser executados pelo computador. Ou seja, o usuário é encorajado a praticar modelagem. Linguagens de programação de propósito geral podem ser vistas como ambientes para modelagem, uma vez que permitem que o seu usuário desenvolva, execute e verifique os resultados do processamento dos modelos estabelecidos. Sob uma perspectiva histórica, os primeiros sistemas de modelagem computacional foram construídos usando linguagens de propósito geral, como o FORTRAN, por exemplo (1957-1960). Logo e SmallTalk trouxeram a modelagem para o contexto educacional. A década de 70 foi marcada pela separação lógica das fases de modelagem e simulação. Na década de 80 foi marcante a inclusão de novas tecnologias como, por exemplo, bancos de dados, recursos gráficos etc.. Os anos 90 tem sido marcados por uma mudança de ênfase nos sistemas para modelagem: da programação para a construção, simulação e refinamento de modelos. Linguagens de programação de propósito geral certamente possuem facilidades para a construção de estruturas de dados contendo conhecimento sobre domínios diferentes. Tais estruturas podem ser consideradas como modelos estabelecidos em um domínio. Entretanto, ao contrário de ferramentas específicas para modelagem, normalmente oferecem pouco suporte para a separação das fases de construção e processamento do modelo (Miller *et alli*, 1993). Além disso, as primitivas nas linguagens de programação são mais gerais e possuem um nível de granulosidade diferente do nível de primitivas conceituais que são usadas para descrever o domínio. O uso da linguagem de programação para modelagem é, em geral, vista pelo usuário como instruções à máquina e não no nível mais alto de descrição de um modelo num dado domínio.

A complexidade da tarefa de desenvolver um modelo utilizando uma linguagem de programação de propósito geral pode ser reduzida pelo uso de ambientes para modelagem de propósito geral. Estes ambientes contém uma variedade primitivas que podem diminuir o esforço cognitivo inerente à tarefa de programar. **Stella** (ou o seu sucessor comercial **Ithink**) e **Extend** são exemplos que se aproximam bem do que estamos chamando de

ambientes para modelagem de propósito geral. Ambos sistemas prestam-se para modelagem de fenômenos dinâmicos e podem ser aplicados para estabelecer modelos em diferentes domínios. Lindfield (1992) afirma que o Stella tem sido aplicado para estabecer modelos nas áreas de Economia, Biologia, Física, Engenharia Elétrica entre muitas outras. A próxima seção analisa estes e outros ambientes para modelagem. Alguns micromundos também podem ser considerados ambientes para modelagem de propósito geral. Squires (1987) ressalta que várias definições para micromundos tem sido propostas segundo diferentes pontos de vista. Para deJong (1991) um micromundo é visto como um ambiente computacional no qual o aprendiz pode manifestar comportamento exploratório dentro de um conjunto bem definido de limites (os limites do micromundo) sendo o Logo um exemplo desses ambientes. Também, a *DynaTurtle,* proposta por diSessa (1982) para aprendizagem sobre movimento, é frequentemente referenciada como outro exemplo de micromundo. Em alguns micromundos, como no Logo, o usuário realiza trabalhos de modelagem. Em outros micromundos, como o da *DynaTurtle,* o trabalho do usuário é tipicamente de simulação conforme sugerem diSessa (1982) e diSessa e White (1982).

2.2 - AMBIENTES DE SUPORTE À MODELAGEM EM UM CONTEXTO EDUCACIONAL

Um ambiente para modelagem, em geral, provê suporte às fases de criação do modelo, execução e análise/interpretação de saídas. A criação do modelo emprega alguma linguagem formal. Esta linguagem formal, consiste de um conjunto de símbolos, juntamente com um conjunto de regras para combiná-los. No contexto de uma linguagem para modelagem, os símbolos constituem as primitivas de modelagem, a partir das quais modelos podem ser construídos. Tal vocabulário pode consistir de elementos simbólicos e/ou entidades gráficas como nós e conexões entre eles. Um ambiente para modelagem provê um suprimento de tais elementos. Uma vez estabelecidos os componentes e relações entre os elementos de um modelo, ele pode ser executado e apresentado como uma simulação interativa.

Um dos primeiros *softwares* para modelagem desenvolvidos com propósito educacional foi o *Dynamic Modelling System* - DMS (Schecker, 1993). O DMS presta-se para modelagem de sistemas dinâmicos e requer a expressão de seus modelos em uma linguagem semelhante ao Basic. A exemplo do que acontece com outros ambientes para modelagem de propósito geral, apresentados nesta seção, o DMS permite modelar uma

infinidade de fenômenos que apresentam um comportamento dinâmico. Para se ter uma idéia da variedade de fenômenos dinâmicos que podem ser modelados Lindfield (1992), Imagine That (1993), High Performance Systems (1994), Riley (1990), apresentam exemplos de modelos para os seguintes fenômenos:

- na Biologia: o crescimento de uma planta, a descarga de um neurônio, as batidas do coração, poluição de um lago, inter-relação presa-predador;
- na Economia ou no comércio: efeito da propaganda na venda de um produto, as filas típicas de um sistema de atendimento bancário; a inter-relação entre clientes, quantidade de funcionários, volume de trabalho e qualidade de serviço prestado por uma companhia de consultoria;
- na Geografia: crescimento populacional;
- na Hidrologia: os efeitos da evaporação, chuva, condensação e transpiração de plantas sobre o volume de água de um lago;
- na Matemática: resolver numericamente equações diferenciais;
- em processos industriais: linha de montagem de um produto, sistemas de distribuição.

O problema com o DMS é que ele foi desenvolvido para ser utilizado por usuários com conhecimento formal de física, dos quais se espera habilidade em manipular as estruturas simbólicas usadas para representar um modelo e, adicionalmente, familiaridade com uma linguagem de programação semelhante ao Basic (Riley, 1990).

O *software* Stella ou o seu sucessor comercial Ithink (High Performance Systems, 1994), prestam-se, a exemplo do DMS, para modelagem de sistemas dinâmicos. Contudo, o Stella possibilita a construção de modelos utilizando menos estruturas simbólicas do que o DMS. Os modelos no Stella são uma composição de diagramas e fórmulas algébricas construídos em dois níveis. Em um nível mais alto, as estruturas do modelo são descritas por diagramas em termos de primitivas representando tanques (ou estoques), fluxos e conversores que podem ser unidos através de conectores (Lindfield, 1992). Em um segundo nível, a partir de um duplo *click* em cada um dos elementos envolvidos no modelo, são definidas as relações algébricas entre eles. Cada primitiva tem um ícone correspondente associado, sendo que o diagrama pode ser composto a partir de ações com o *mouse* (figura 2.2-a). Uma primitiva do tipo tanque possui um valor inicial e é capaz de acumular valores quando a ela são ligados elementos do tipo fluxo (figura 2.2-b). Um fluxo pode ser positivo ou negativo: fluxos positivos incrementam o valor do tanque; fluxos negativos, por sua vez, o decrementam.

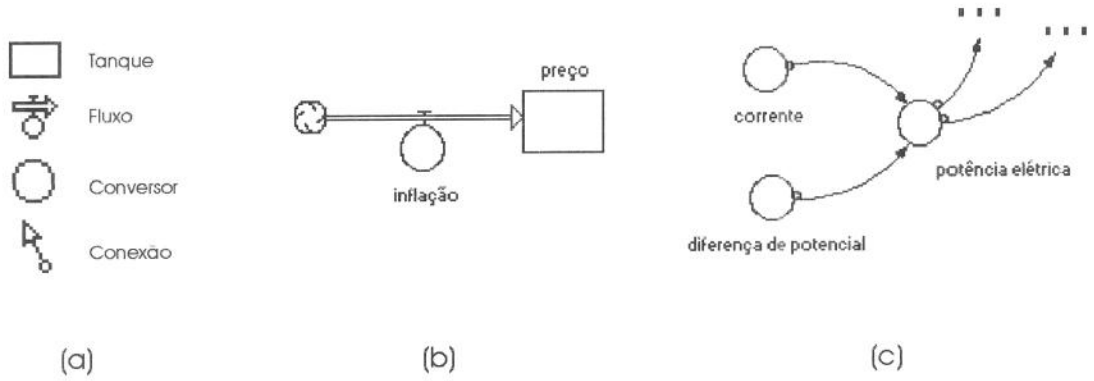


Fig. 2.2: Primitivas usadas para descrever modelos no Stella

Um conversor é uma primitiva capaz de transformar o valor de suas conexões de entrada em um valor que é colocado em suas conexões de saída. Tipicamente realizam uma transformação algébrica como, por exemplo, na figura 2.2-c o conversor chamado potência elétrica torna disponível em suas conexões de saída o valor da potência elétrica calculada como o produto de suas conexões de entrada. A fórmula que permite a transformação realizada por um conversor pode ser vista e/ou editada dando-se um duplo *click* no mesmo. Por exemplo, dando um duplo *click* no conversor potência elétrica pode-se descrever a potência elétrica como

corrente * diferença de potencial.

O Stella permite que um modelo seja executado. Como saída, o usuário pode solicitar a exibição de tabelas, *display* numérico ou gráficos mostrando a variação das quantidades envolvidas ao longo do tempo. Ou, ainda, o usuário pode solicitar que o Stella anime os diagramas construídos. A animação mostrará a variação, do conteúdo dos elementos dos tipos tanque e conversor imitando o enchimento ou esvaziamento de uma caixa d'água. A figura 2.3-a mostra um modelo Stella estabelecido para a inter-relação presa-predador em um ecosistema de 100 hectares contendo, inicialmente, 6000 presas e 125 predadores. O modelo supõe: (1) que cada presa procria a uma taxa tal que sua população aumenta em 25 porcento por mês. Já os predadores, se reproduzem a uma taxa menor equivalente a 2.5 porcento a cada mês; (2) que a densidade populacional de presas mantém uma relação

direta com a mortalidade das mesmas (0.3*densidade_presa) e inversa com a mortalidade dos predadores[1] (1.2/densidade_presa). A figura 2.3-b mostra um gráfico que foi obtido executando o modelo em um intervalo de tempo igual a 200 meses.

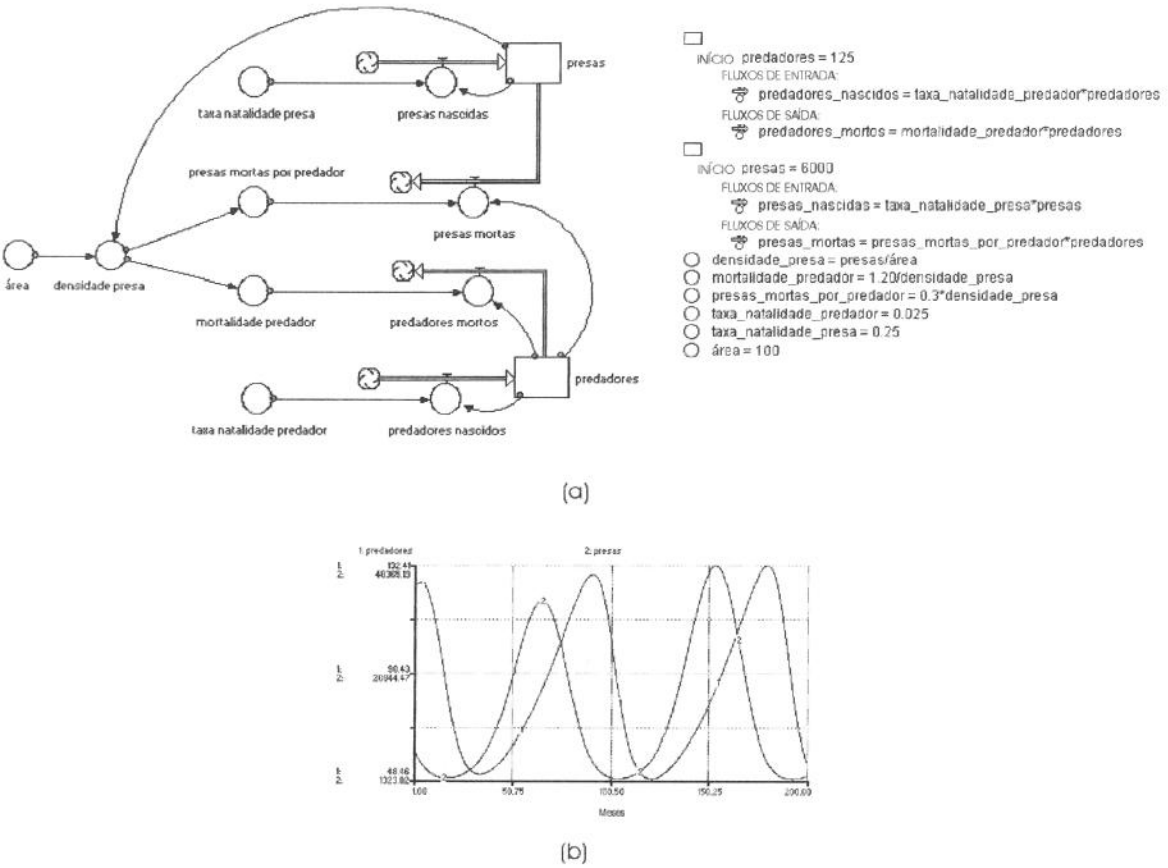


Fig. 2.3: Um modelo Stella para inter-relação presa-predador: (a) o diagrama e os valores estabelecidos para cada um de seus elementos; (b) um gráfico mostrando a variação da quantidade de presas e predadores em 200 meses

Extend (Imagine That, 1993) é um requintado ambiente para modelagem de propósito geral, também, para sistemas dinâmicos. Tal como o Stella, Extend possui um conjunto de primitivas gráficas, na denominação dada pelo ambiente "blocos", que podem ser utilizados para se construir modelos gráficos através de operações com o *mouse.*

[1] A lógica do modelo no que diz respeito à mortalidade de presas e predadores reside nos seguintes aspectos: (1) quanto maior for a densidade das presas, maior será a mortalidade das mesmas pois serão capturadas mais facilmente pelos predadores (relação direta); (2) também, uma maior densidade de presas ocasiona menor mortalidade de predadores pois os mesmos terão maior oferta de alimentos (relação inversa).

Diferente do Stella, os blocos não possuem nomes. Os nomes dos elementos envolvidos no modelo são conferidos às linhas que conectam os blocos. Numa analogia aos circuitos eletrônicos, as conexões são os fios através dos quais pode-se fazer trafegar a corrente elétrica, no caso, o dado. Os blocos executam funções específicas sobre os dados que são conduzidos aos seus conectores de entrada e tornam disponíveis, em seus conectores de saída, dados transformados segundo a sua especificidade. Por exemplo, o bloco da figura 2.4-a possui a função de multiplicar os dados que chegam aos seus conectores de entrada (a e b) colocando o resultado em seu conector de saída (c). O bloco da figura 2.4-b permite que o usuário digite uma equação que relacione os dados que chegam aos seus conectores de entrada (à esquerda do bloco) para compor a sua única saída (à direita do bloco). O bloco da figura 2.4-c assemelha-se ao elemento do tipo tanque do Stella: dados podem incrementar o valor corrente do bloco se forem conduzidos a este pelo seu conector da "torneira" localizada no topo do bloco; o valor do bloco pode ser decrementado em quantidades iguais ao valor do dado conectado ao seu conector **want**; ainda, o valor corrente acumulado pelo bloco pode ser conhecido através de seu conector **c**. O bloco da figura 2.4-d presta-se para tornar disponível um valor constante em seu conector de saída ou uma transformação algébrica realizada sobre o seu único conector de entrada.

(a) (b) (c) (d)

Fig. 2.4: Alguns blocos (primitivas) disponíveis no Extend

A figura 2.5 apresenta o correspondente modelo Extend para a inter-relação presa-predador discutida anteriormente.

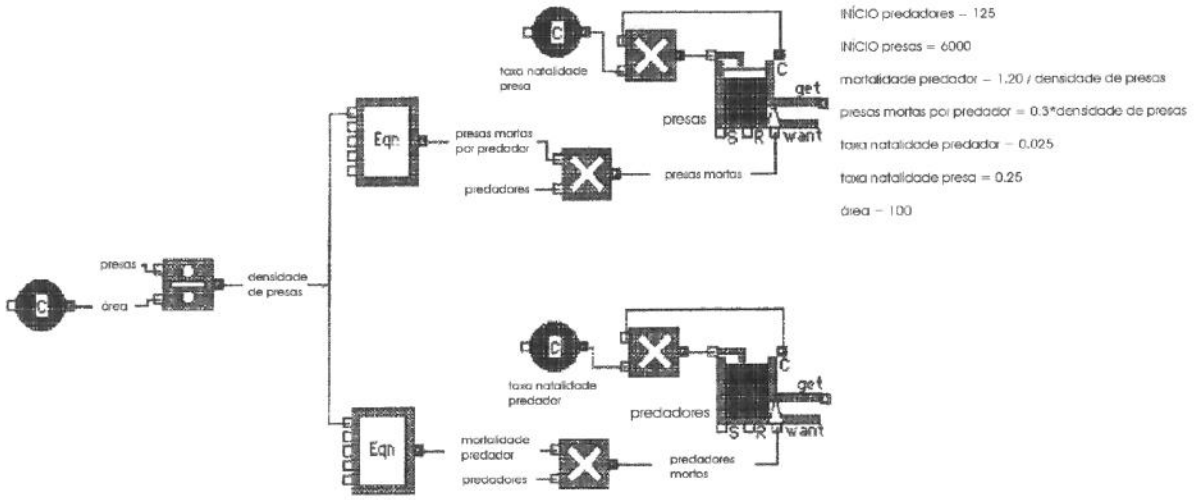


Fig. 2.5: O correspondente modelo para a inter-relação presa-predador estabelecido no Extend

O Extend permite, também, a animação dos blocos envolvidos em um modelo: os blocos do tipo tanque enchem-se e esvaziam-se durante a execução e é possível, na modelagem de fenômenos dinâmicos discretos, visualizar sinais luminosos correspondentes aos dados trafegando pelo modelo. O usuário pode construir blocos que contém em seu interior um submodelo, criando algo que pode ser comparado a um subprograma nas linguagens de programação. É possível, também, construir blocos novos ou alterar o comportamento dos blocos predefinidos através de uma linguagem compilada denominada ModL bastante semelhante à linguagem C. Segundo a Imagine That (1993) o Extend é absolutamente extensível não havendo, virtualmente, nenhum limite à criação de modelos.

Planilhas eletrônicas como Excel podem ser vistas como ambientes para modelagem (Miller *et alli,* 1993). Originalmente, as planilhas eletrônicas foram desenvolvidas com propósito de criar modelos financeiros. Os modelos criados por esta categoria de *software* em geral envolvem uma série de valores, tais como custos e lucro, que são calculados, normalmente, de forma não iterativa. Em Excel uma planilha é um modelo. E uma planilha é um conjunto de linhas e colunas. O campo correspondente a uma determinada linha e coluna é comumente denominada de célula. Qualquer célula pode conter uma cadeia de caracteres, um número ou um dado estruturado tal como data ou tempo. O conteúdo de uma célula pode ser fixo ou pode ser calculado a partir do conteúdo de outras células aplicando fórmulas compostas de diversos operadores e funções. A forma

mais comum de se referir a uma célula é através das suas coordenadas matriciais. Qualquer alteração em uma célula independente é imediatamente propagada para as células que dela dependem. Para Miller, as planilhas eletrônicas suportam modelos algébricos sendo que, um vasto espaço de soluções de modelagem pode ser explorado, porém, apenas por trocar os valores das variáveis de entrada de uma certa forma sistemática e por gravar os valores de saída. Contudo, versões recentes já suportam solução de problemas de forma automática, isto é, dado um modelo e dado o valor de uma célula dependente o sistema pode calcular os valores das células independentes (Miller *et alli,* 1993). Tais cálculos, é claro, podem não convergir para uma solução única.

As planilhas, por apresentarem uma estrutura de dados bastante genérica (uma matriz de células), são absolutamente neutras no que diz respeito à estrutura e tipo de modelos que podem ser estabelecidos. Logo, sua estrutura precisa ser adaptada ao *design* de determinado modelo. Esta adaptação inclui a denominação e formatação do conteúdo das células, a criação de bordas para enfatizar certas células e o ocultamento daquelas cujos valores são necessários para efeito de cálculo mas que não tem relevância no visual do modelo. Outra maneira de adaptar uma planilha ao *design* de determinado modelo consiste em escrever pequenos programas chamados macros. Entretanto, o uso de macros requer considerável esforço de programação.

Recentemente, vários protótipos de novos ambientes para modelagem estão sendo propostos, alguns deles dirigidos a estudantes mais novos, como o sistema Expert Builder (Millwood e Stevens, 1990) e os sistemas IQON, SMOD e QMOD (Miller *et alli,* 1993).

Expert Builder é um sistema especialista baseado em regras que explicita claramente a estrutura de conhecimento e o mecanismo de inferência através de uma interface gráfica (Webb, 1994). O programa funciona de maneira semelhante aos sistemas especialistas disponíveis comercialmente. Webb afirma que a maior diferença entre o Expert Builder e os sistemas especialistas tradicionais reside em sua interface gráfica, que permite que um modelo seja facilmente construído por uma criança. Usando Expert Builder um modelo pode ser feito pela construção de uma estrutura lógica em forma de diagramas constituídos através de ações com o *mouse.* O diagrama consiste de retângulos contendo cláusulas textuais conectadas umas às outras por meio dos operadores lógicos AND, OR e NOT (figura 2.6).

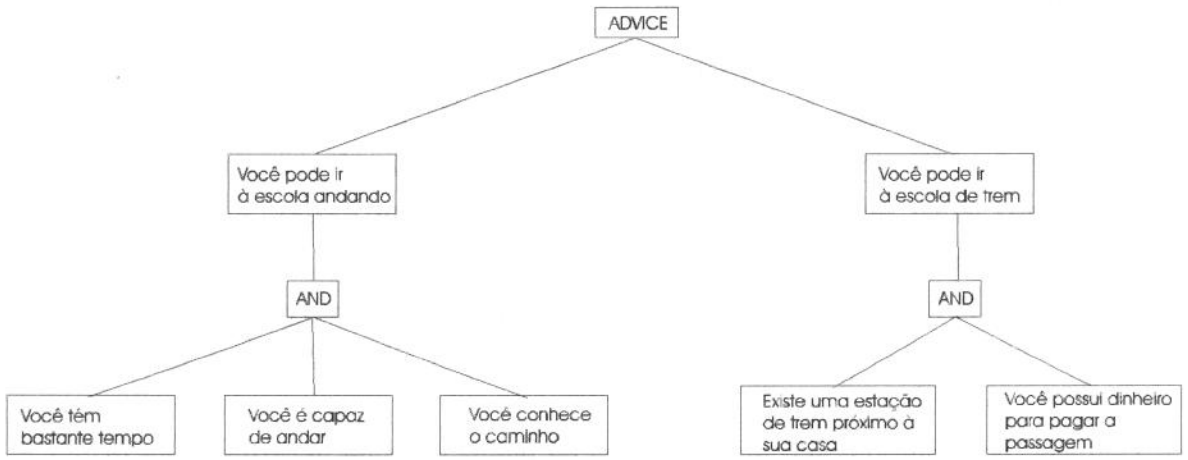


Fig. 2.6: Um modelo típico estabelecido com o Expert Builder (adaptado de MillWood e Stevens (1990), p. 253)

O protótipo de ambiente para modelagem IQON permite uma modelagem semi-quantitativa de sistemas dinâmicos (Miller *et alli,* 1991). Por semi-quantitativa devemos entender que as quantidades envolvidas em um fenômeno, bem como as relações entre estas quantidades, não são descritas por seu valor absoluto. Os valores das variáveis envolvidas em um fenômeno são descritas a partir de termos tais como: "positivo", "negativo", "zero", "alto", "baixo", "normal", "desprezível", "incrementando", "decrementando", "grande", "pequeno". Similarmente, os relacionamentos entre as variáveis podem ser descritos como "positivo" ou "negativo" de maneira que os efeitos destes sejam "forte", "fraco", "desprezível", "esperado", "imediato" ou "gradual". O vocabulário semi-quantitativo corresponde ao intuitivo e ao senso comum que as pessoas formam sobre o estado de um sistema. O IQON permite a criação de modelos por meio de diagramas de laços de causa. A figura 2.7 apresenta dois diagramas de laços causais bastante simples: o diagrama 2.8-a indica que quanto maior for o tempo de uso de uma pilha eletroquímica, menor será a carga armazenada por ela; o diagrama 2.8-b mostra que quanto maior for a renda de uma pessoa maiores serão, também, as suas despesas e a sua poupança. Ainda, quanto maior forem as despesas menor será a poupança e vice-versa.

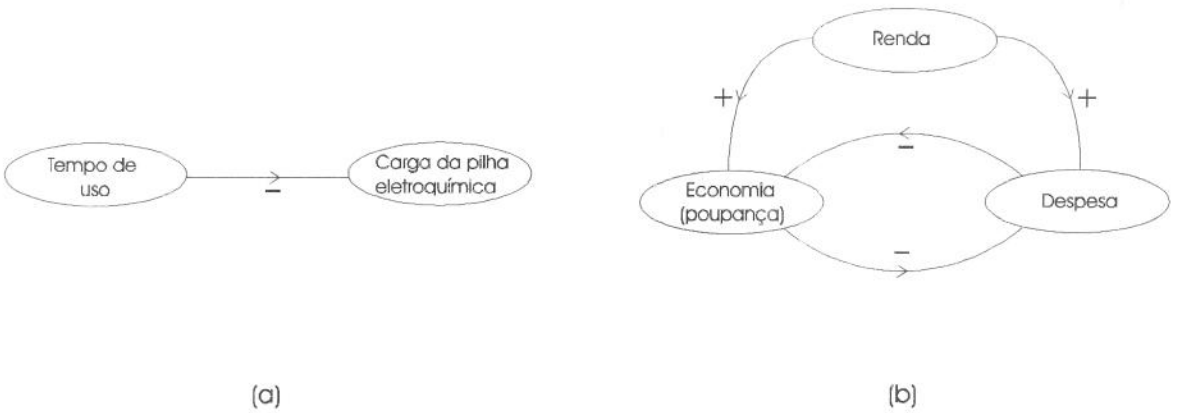


Fig. 2.7: Diagrama de laços de causa

O IQON possui um conjunto de ferramentas que permite criar diagramas de laços de causa. No IQON, as variáveis envolvidas em um modelo são representadas por um retângulo e as relações entre as variáveis por uma linha direcionada cujo centro apresenta um círculo com os sinais de "+" ou "-", significando relações positivas ou negativas. A figura 2.8 apresenta um modelo para o ciclo do carbono.

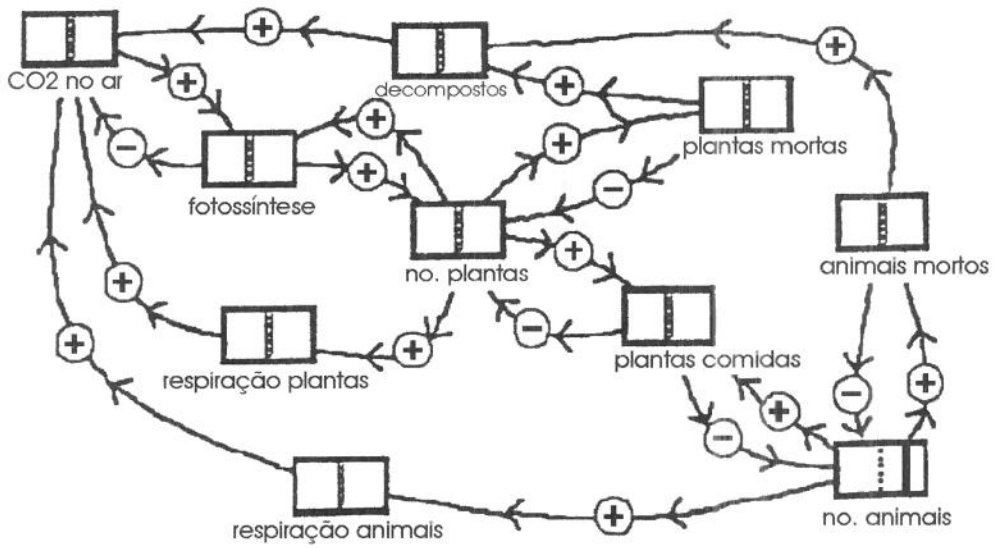


Fig. 2.8: Um modelo para o ciclo do carbono estabelecido com o IQON (traduzido de Miller *et alli* (1993), p. 222)

O valor ou o nível de um retângulo (leia-se de uma variável envolvida no modelo) é indicado por uma linha vertical que pode se mover durante a execução. O lado direito do retângulo representa valores acima de zero e o lado esquerdo representa valores abaixo de zero. Assim, uma variável está qualitativamente no nível normal se apresentar uma linha no meio do retângulo; uma variável que apresenta uma linha vertical no lado direito possui valor acima do normal e, de maneira similar, uma variável que apresenta uma linha vertical no lado esquerdo apresenta valor abaixo do normal. O poder (ou força) da relação entre duas variáveis é sinalizada por diferentes tamanhos dos sinais de "+" e de "-" existentes dentro do círculo das relações entre elas. Estes diferentes tamanhos significam uma relação fraca, média ou forte negativamente ou positivamente. Além da animação dos elementos do diagrama, o IQON permite ao usuário a observação de gráficos que mostram o comportamento da quantidades envolvidas.

O protótipo de ambiente para modelagem S-MOD (Miller *et alli*, 1993) permite criar modelos semi-quantitativos da mesma forma que o IQON. Os modelos criados pelo IQON e pelo S-MOD são intercambiáveis, isto é, podem ser criados e executados em um ou em outro ambiente. Na prática a diferença do IQON para o S_MOD reside em dois aspectos: (1) o S_MOD (o "S" é abreviatura de *static*) não permite a simulação dinâmica do modelo. O modelo apresenta-se estaticamente; (2) o S-MOD ajusta automaticamente os níveis das variáveis afetadas por uma alteração feita, interativamente pelo usuário, sobre as variáveis independentes do modelo.

O protótipo de ambiente para modelagem Q-MOD (Miller *et alli*, 1993) permite expressar modelos que representam graficamente expressões algébricas envolvendo as quatro operações básicas ""+","-","×", "÷". Em Q-MOD quadrados representam variáveis e as direções das setas que as unem indicam o fluxo de cálculo. A figura 2.9 apresenta um modelo Q-MOD, adaptado de Miller *et alli* (1993), que inter-relaciona exercício e dieta. Q-MOD automaticamente calcula os valores das variáveis quando se altera o valor de uma variável independente. Gráficos podem ser gerados mostrando a relação entre uma variável dependente e uma variável independente, para a situação em que se considera mantidas todas as demais variáveis envolvidas.

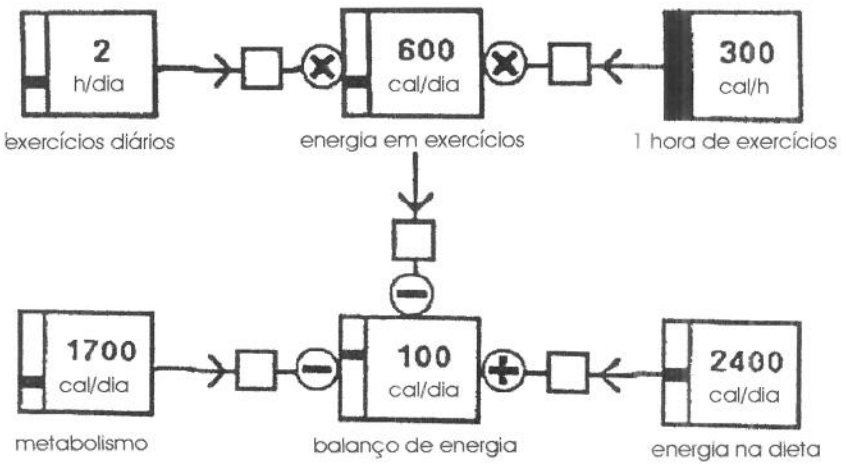


Fig. 2.9: Um modelo para a inter-relação exercício e dieta estabelecido com o Q-MOD (traduzido de Miller *et alli* (1993), p. 232)

2.3 - DISCUSSÃO E PROPOSTA

Se por um lado, ambientes para modelagem de propósito geral, como os que foram mostrados na seção anterior, oferecem um alto nível de controle quando comparados com ambientes para simulação, por outro lado eles demandam maior esforço cognitivo do usuário na sua operação. Como observou Ross (1986), além das dificuldades inerentes à definição do que vai ser modelado e ao projeto do modelo, as fases de criação e teste do modelo são especialmente exigentes para com o usuário. Para criar o modelo, o usuário necessita conhecer a linguagem para modelagem subjacente ao ambiente que está utilizando. A aprendizagem de técnicas de programação e caraterísticas da linguagem para modelagem, simultaneamente ao exercício da atividade de modelar, aumenta o esforço cognitivo do usuário, ao mesmo tempo em que desvia sua atenção dos aspectos relativos ao domínio de interesse. No teste do modelo, o usuário deve ser capaz de julgar os resultados. Mais importante ainda, ele deve ser capaz de avaliar um resultado inesperado como conseqüência de um erro de projeto, um erro de implementação, um erro de ambos ou, ainda, associar o erro ao seu entendimento do fenômeno modelado.

Um ambiente para modelagem de propósito geral solicita ao usuário atenção com aspectos nem sempre vinculados ao domínio sobre o qual um modelo está sendo desenvolvido. Isto se deve à generalidade do ambiente e, por vezes, pode confundir o aprendiz de um certo domínio. Por exemplo, tanto o Stella como o Extend possuem um conjunto especial de

primitivas para modelagem de eventos dinâmicos discretos. Ora, que importância tem as primitivas para modelagem de eventos discretos, por exemplo, para um aprendiz interessado no estudo da relação presa-predador? Ou, generalizando, que importância tem as referidas primitivas para a descrição de fenômenos em um domínio que delas não necessita?

De fato, ambientes para modelagem de propósito geral, como os que apresentamos na seção anterior, possuem primitivas bastantes genéricas, não se comprometendo com nenhum domínio específico. Nós argumentamos que um aprendiz interessado em um determinado domínio não deve ter que "programar" em uma linguagem para modelagem de propósito geral para criar, executar e avaliar o seu modelo. Nós entendemos que um aprendiz interessado em Ótica Geométrica deva usar uma "linguagem para Ótica Geométrica", linguagem esta composta de objetos tais como espelhos, lentes, pontos de luz e primitivas como a que emite um raio de luz sobre uma lente. De forma similar, aprendizes interessados em Genética devem usar uma "linguagem para Genética" composta de objetos tais como genes e fatores e primitivas tais como a que permite fazer um cruzamento.

O problema da generalidade dos ambientes para modelagem atuais tem como raiz os formalismos genéricos nos quais o seu *design* foi concebido. Todos os ambientes para modelagem de propósito geral apresentados na seção anterior são versões computadorizadas de algum formalismo tradicional de modelagem. Por exemplo, o formalismo subjacente ao Q-MOD é a Álgebra (ou Matemática); Schecker (1993) cita que o Stella segue a teoria de sistemas dinâmicos de Forrester (1968); IQON e S-MOD estão baseados nos trabalhos de Roberts *et alli* (1983) sobre diagramas de laços de causa, observa Miller *et alli*, (1993); Expert Builder está fundamentado nos cálculos de predicado de primeira ordem.

Do fato de serem versões computadorizadas de algum formalismo tradicional, advém um outro problema cuja solução não é referenciada na literatura: dado um domínio de conhecimento, como estabelecer um ambiente que permita a modelagem de fenômenos deste domínio? Ou, descrevendo o problema de um outro modo: que estratégias podem ser utilizadas para se estabelecer o *design* de um ambiente que permita a modelagem em um determinado domínio?

Baranauskas e Oliveira (1995) argumentam que o problema com os ambientes para modelagem de propósito geral está associado à generalidade de suas primitivas, que possuem um nível de granulosidade diferente do nível das primitivas conceituais que são utilizadas para descrever um domínio. A nossa proposta é a de fazer uma síntese entre os pólos extremos mencionados na seção 2.1 (De simulação à linguagem de programação) deslocando a ênfase dos sistemas para modelagem de programação para a construção de modelos em um domínio específico (figura 2.10)

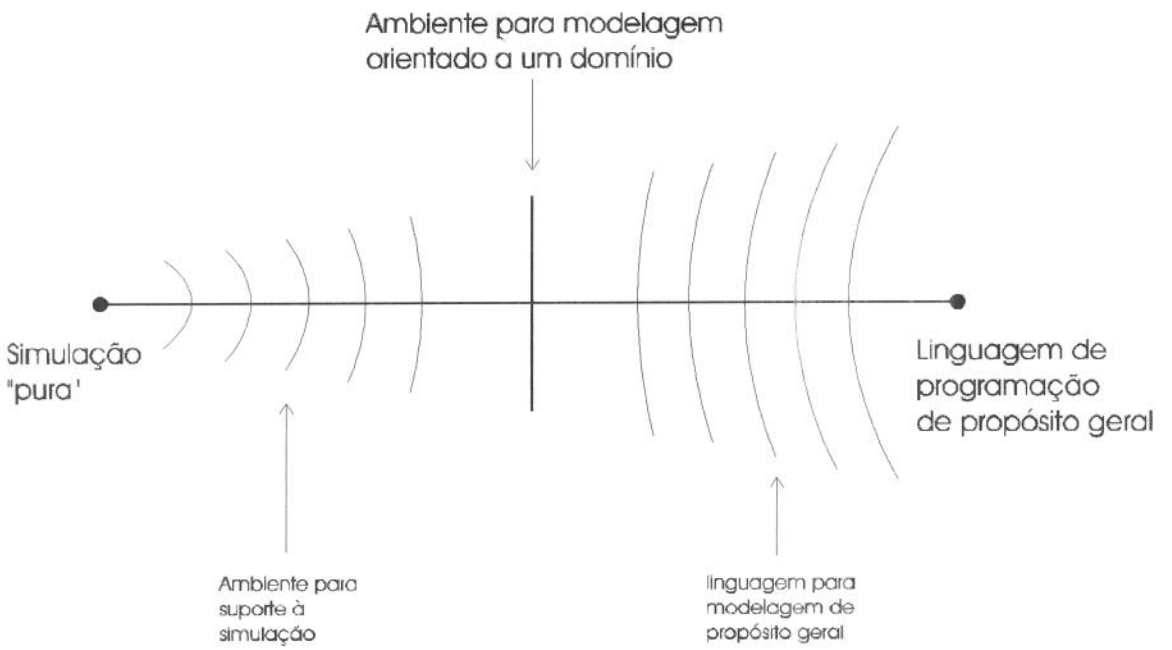


Figura 2.10: O ambiente para modelagem orientado a um domínio

Por restringir o ambiente para modelagem a um domínio específico, nós objetivamos estabelecer um equilíbrio entre os benefícios e dificuldades encontradas nos dois extremos do espectro de ambientes que se prestam à modelagem e simulação. Fazendo isto, estaremos estabelecendo o que, a partir de agora, passaremos a denominar de Ambiente para Modelagem Orientado a um Domínio (AMOD). Um AMOD representa, portanto, um equilíbrio entre o esforço cognitivo (do usuário) e o nível de controle (do usuário sobre o ambiente) encontrados em todo um espectro de ambientes que podemos imaginar inseridos em uma linha imaginária que se estende desde os ambientes puramente de

simulação (baixo esforço cognitivo mas baixo nível de controle) até as linguagens de programação de propósito geral (alto nível de controle mas alto esforço cognitivo).

Capítulo 3

Objetivos

Este capítulo expõe os objetivos de nosso trabalho.

Temos dois grandes objetivos com este trabalho: (1) definir um conjunto de estratégias que possam guiar o *design* de ambientes computacionais para modelagem, orientados a um domínio específico, para aplicação segundo um contexto educacional (o que chamamos, no capítulo anterior, de AMOD) e (2) experimentar estas estratégias no desenvolvimento de um destes ambientes.

O trabalho envolve basicamente:

- a definição de estratégias para o *design* dos AMOD;
- a seleção e o estudo de um domínio de conhecimento que possa ser utilizado na experimentação das estratégias propostas;
- a seleção e o estudo de um ambiente computacional que possa ser utilizado na implementação do protótipo do ambiente para modelagem no domínio de conhecimento escolhido e cujo *design* seguirá as estratégias propostas;
- o *design*, utilizando as estratégias que serão propostas e implementação de um protótipo de ambiente para modelagem no domínio de conhecimento escolhido.

O nosso objeto de estudo é, portanto, o *design* de ambientes computacionais para modelagem orientados a um domínio específico. Basicamente, estes ambientes combinam uma variedade de formas para expressão, pelo usuário, do modelo que ele (usuário) tem em mente. Também, apresentam o resultado da execução do modelo criado, para ser "percebido" e interpretado pelo usuário. As questões que objetivamos investigar são: quais devem ser as primitivas do ambiente para modelagem em um determinado domínio de conhecimento? Por exemplo, se o domínio for a genética mendeliana, serão genes/fatores as primitivas? Quais as conseqüências de se escolher um ou outro conjunto de primitivas? A definição de um domínio de conhecimento e de uma classe de usuários que potencialmente possam usar o ambiente contribui efetivamente para determinação de um conjunto de primitivas adequado? Como a tecnologia disponível influencia o desenvolvimento do ambiente para modelagem? Enfim, as questões anteriormente apresentadas fazem parte da questão maior que traduz o nosso objeto de estudo e que podemos colocar assim: como estabelecer o *design* de ambientes computacionais para modelagem, orientados a um domínio específico, para aplicação em um contexto educacional?

Sob a perspectiva do uso educacional de um ambiente para modelagem, estamos interessados em investigar que características ele deve ter de forma a facilitar ao aprendiz a expressão, a avaliação de suas próprias idéias e a evolução de seu conhecimento sobre um domínio, ao estilo construcionista de aprendizagem (Papert, 1986).

Por fim, mesmo não sendo nosso objetivo direto, acreditamos que o entendimento do *design* dos AMOD (nosso objetivo) poderá, adicionalmente, nos fornecer importante *feedback* ao *design* de *software* de um modo geral.

Capítulo 4

Estratégias para o *design* de Ambientes para Modelagem Orientados a um Domínio

Propusemos, no capítulo 2, os Ambientes para Modelagem Orientados a um Domínio (AMOD). Tais ambientes representam, simultaneamente, um equilíbrio entre o esforço cognitivo (do usuário) e o nível de controle (do usuário sobre o ambiente), quando consideramos um intervalo que se estende desde os ambientes puramente de simulação até as linguagens de programação de propósito geral. O presente capítulo discute as bases teóricas para o desenvolvimento de tais ambientes. Cinco princípios são enunciados e são devidamente organizados para compor nossas estratégias para o *design* destes ambientes. Ao final, confrontamos as estratégias discutidas com alguns métodos tradicionais para análise, projeto e implementação de sistemas.

Tópicos deste capítulo:

Tópico		*Página*
4.1 -	Introdução	33
4.2 -	Restrição de domínio	34
4.3 -	Identificando e escolhendo os conceitos atômicos do ambiente para modelagem	36
4.4 -	Desenvolvendo o *design* de cada conceito atômico e do ambiente para modelagem como um todo	41
4.5 -	Resumo das estratégias	46
4.6 -	As estratégias discutidas diante de alguns métodos tradicionais para análise, projeto e implementação de sistemas	47

4.1 - INTRODUÇÃO

De um lado, temos um domínio de conhecimento; de outro lado, um ambiente para construção de modelos para fenômenos que ocorrem neste domínio. De um lado uma casa e uma ponte; de outro lado, os modelos para uma casa e para uma ponte construídos a partir de um conjunto de peças, por exemplo, do jogo "Lego *System*" da Lego do Amazonas Brinquedos Ltda. Dois desafios: (1) dadas as peças, como construir a casa e a ponte? (2) dada a casa e a ponte, quais peças deve ter o jogo? Compete ao modelador vencer o primeiro desafio. Mas, muito antes do modelador ser levado ao trabalho, compete ao *designer* vencer o segundo desafio e é deste assunto que tratamos neste capítulo. Evidentemente, não estamos interessados no *design* de brinquedos mas no *design* de ambientes computacionais para modelagem. Baranauskas e Oliveira (1994-a) descrevem um conjunto de estratégias para o *design* desses ambientes organizadas segundo três etapas (figura 4.1).

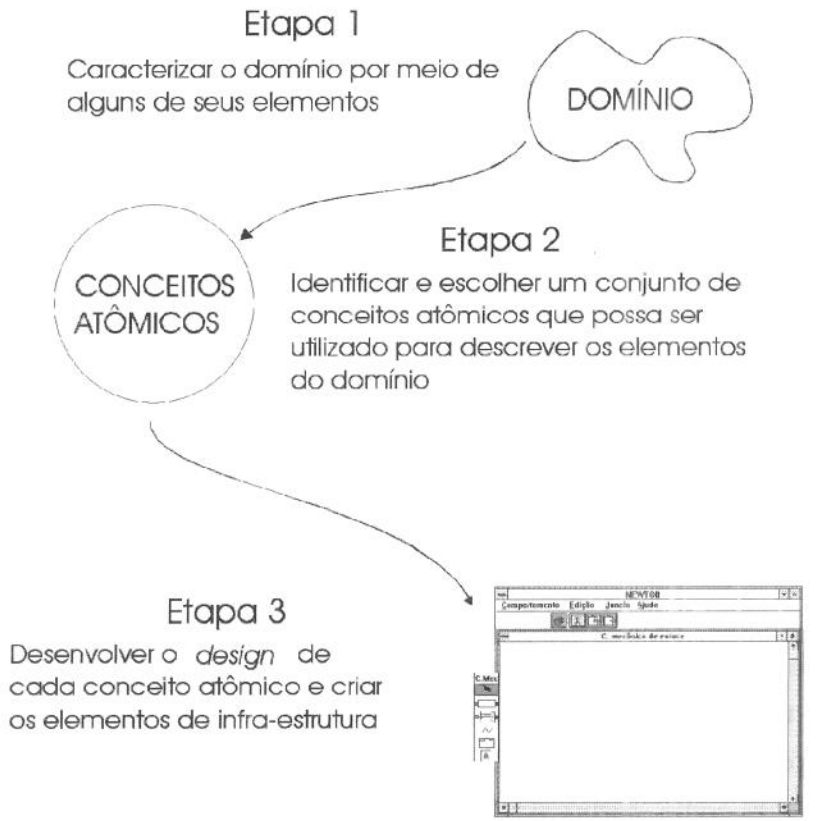


Fig. 4.1: Visão geral das estratégias para o *design* de Ambientes para Modelagem Orientados a um Domínio

A partir da escolha do domínio para o qual se quer desenvolver um ambiente para modelagem, a primeira etapa consiste em delimitar este domínio de conhecimento. Isto pode ser feito caracterizando o domínio por meio de alguns dos elementos que o compõem. Na segunda etapa, deve-se analisar e "fatorar" cada elemento usado na caracterização do domínio, isto é, deve-se obter um conjunto de "sub-elementos" (conceitos atômicos, na terminologia que será empregada), dentro do domínio, tais que possam ser utilizados para, no sentido inverso, descrevê-los. Em outras palavras, deve-se identificar conceitos atômicos, dentro do domínio, de tal sorte que quando devidamente combinados, possam ser utilizados para descrever os elementos do domínio de interesse. Nesta etapa, o *designer* não deve se preocupar com a forma de representação que será dada para o conceito atômico no ambiente para modelagem que ele pretende desenvolver. Apenas, deve descobrir e se assegurar de que o conjunto de conceitos atômicos identificado serve para descrever os elementos usados para caracterizar o domínio. Finalmente, na terceira etapa, o *designer* deve desenvolver o *design* de cada conceito atômico. Estruturas adicionais àquelas que visam implementar os conceitos atômicos podem ser necessárias como, por exemplo gravar, ler um modelo, fornecer ajuda ao uso do ambiente etc.. As atividades envolvidas na terceira etapa dependem, entre outras coisas, da tecnologia disponível para implementação: CPU, memória, discos, periféricos de entrada e saída etc.. É importante, também, compreender como as pessoas interagem com os sistemas físicos, no caso os ambientes para modelagem, e neste aspecto a teoria da ação (Norman, 1986) assumirá papel relevante nas discussões que faremos.

4.2 - RESTRIÇÃO DE DOMÍNIO

Eisenberg (1991) considera mais fácil para um usuário, conhecedor ou interessado em um determinado domínio, manipular uma aplicação programável voltada para o referido domínio do que usar uma linguagem de propósito geral para resolver problemas associados a este domínio. Eisenberg propõe a criação de **aplicações programáveis** com vocação para o tratamento de problemas pertencentes a um domínio específico. Isto é, uma aplicação programável para auxiliar o projeto de circuitos digitais, por exemplo, deve conter objetos como portas AND e OR e possuir primitivas que as manipulem diretamente. Uma aplicação programável para um músico, por exemplo, deve conter bibliotecas de sons, bibliotecas de acordes e primitivas que as manipulem diretamente.

Concordamos com Eisenberg e de forma similar, não esperamos que o usuário de um ambiente para modelagem saiba programar em uma linguagem como C++ ou Pascal. Esperamos, no entanto, que um usuário interessado no estudo de genética, por exemplo, consiga usar uma linguagem de propósito específico para a genética, linguagem esta contendo objetos como genes e fatores e primitivas como a que efetua um cruzamento. Portanto, propomos restringir o *design* de ambientes para modelagem a um domínio específico. Este é um de nossos princípios para o *design* de ambientes para modelagem o qual denominamos de **princípio da restrição de domínio**. Este princípio nos conduz ao estabelecimento de ambientes que se prestam à construção de modelos para fenômenos pertencentes a um domínio específico ou, usando a terminologia introduzida no capítulo 2, Ambientes para Modelagem Orientados a um Domínio (AMOD).

A rigor, todo domínio de conhecimento é infinito e possui detalhes infinitesimais. Tecnologicamente, não podemos tratar um domínio de conhecimento desde sua dimensão infinitesimal até sua dimensão infinita e, no entanto, pretendemos desenvolver ambientes para modelagem em um domínio de conhecimento. Mas, como? Se, na verdade, sequer podemos entender e delimitar precisamente o domínio que é nosso objeto de trabalho. Para tornar este problema tratável na condição de ser humano que somos e na condição de *designers* que possuem capacidade, tempo e orçamento limitados, resta-nos uma solução aproximada ao problema da caracterização e delimitação da extensão do domínio. Ou tentamos uma solução aproximada, ou estaremos condenados a passar o resto de nossas vidas tentando caracterizar um domínio de conhecimento e, talvez, não restará tempo para desenvolver o ambiente para modelagem. A solução aproximada que sugerimos consiste em expressar o domínio de conhecimento a partir de um conjunto de elementos (amostragem de objetos e/ou fenômenos do domínio) que o representem satisfatoriamente. Reconhecemos a fragilidade das aproximações e entendemos que, se não forem bem feitas, pode-se distorcer o resultado, ou seja, o ambiente para modelagem. Quanto mais representativo for o conjunto de elementos escolhido melhor será a caracterização do domínio e, portanto, menor a distorção. Uma vez escolhido um conjunto de elementos para representar o domínio, o próximo desafio é encontrar um conjunto adequado de "peças", que devidamente combinadas, possam ser utilizadas para descrever cada elemento do domínio. Isto será tratado na próxima seção.

A suposta desvantagem da restrição de domínio, o alto nível de especialização do ambiente, não é tão significativa se considerarmos que os AMOD são capazes de modelar fenômenos do domínio a que pertence. No capítulo 6 conheceremos o protótipo que

desenvolvemos para modelagem no domínio da Mecânica, o NEWTON. É admirável reconhecer que somos incapazes de dizer quantos fenômenos o NEWTON pode modelar dada a infinidade de possibilidades existentes.

4.3 - IDENTIFICANDO E ESCOLHENDO OS CONCEITOS ATÔMICOS DO AMBIENTE PARA MODELAGEM

Um ambiente para modelagem em um domínio de conhecimento específico é um ambiente que possui estruturas (itens de interface de manipulação direta, comandos em linguagem simbólica etc.) que permitem ao usuário construir modelos dentro deste domínio. Sob o contexto educacional, podemos dizer que, ao interagir com as estruturas do ambiente, o usuário manipula conceitos pertencentes ao domínio de conhecimento subjacente ao ambiente para modelagem. Além disso, podemos entender a atividade de construir um modelo como a atividade de agregar estruturas mais elementares para produzir estruturas conceitualmente mais complexas. A título de analogia, isto é o que acontece com um aprendiz em um ambiente de programação como o Logo (Papert, 1980): a partir das estruturas que implementam o conceito de deslocamento e o conceito de rotação, o aprendiz pode construir estruturas conceitualmente mais complexas, como um retângulo.

Nosso papel, enquanto *designers* de ambientes para modelagem, é o inverso do trabalho do usuário. Ou seja, queremos partir das estruturas mais complexas, especificamente estaremos partindo do todo isto é, dos elementos que representam o domínio, para encontrar as "peças" que nos permitam montar os modelos dos objetos e fenômenos do domínio. Mas, como encontrar tais "peças"? Expressando cada um dos elementos do domínio de conhecimento em função de elementos conceitualmente mais simples. Nós chamamos a estes elementos, conceitualmente mais simples, de **conceitos atômicos**.

Para exemplificar como podemos encontrar um conjunto de conceitos atômicos, consideremos um pequeno domínio de conhecimento formado por, apenas, dois elementos: o retângulo e o quadrado. Utilizaremos, este pequeno domínio de conhecimento para ilustrar a discussão a seguir. Tudo o que precisamos fazer é expressar cada um dos elementos do domínio em função de conceitos mais elementares. Tomemos o retângulo. Ele pode ser descrito, em função de conceitos mais elementares, de diversas maneiras. Eis a primeira: **um retângulo é um quadrilátero equiângulo.** Aqui, o

retângulo foi expresso a partir dos conceitos atômicos de quadrilátero, de ângulo e de igualdade. Vejamos uma outra maneira: **um retângulo é uma figura composta por quatro linhas retas. Cada vértice de uma linha (ponta da linha) está conectado a um, e somente um vértice de outra linha. Os ângulos, formados pela conexão entre as linhas, são todos iguais**. Observamos, agora, que o retângulo foi expresso a partir de outros conceitos atômicos: linha reta, vértice, conexão, ângulo, igualdade, e quantidade (no caso, quatro). Podemos proceder da mesma forma para o quadrado para obter os dois conjuntos de conceitos atômicos apresentados na figura 4.2.

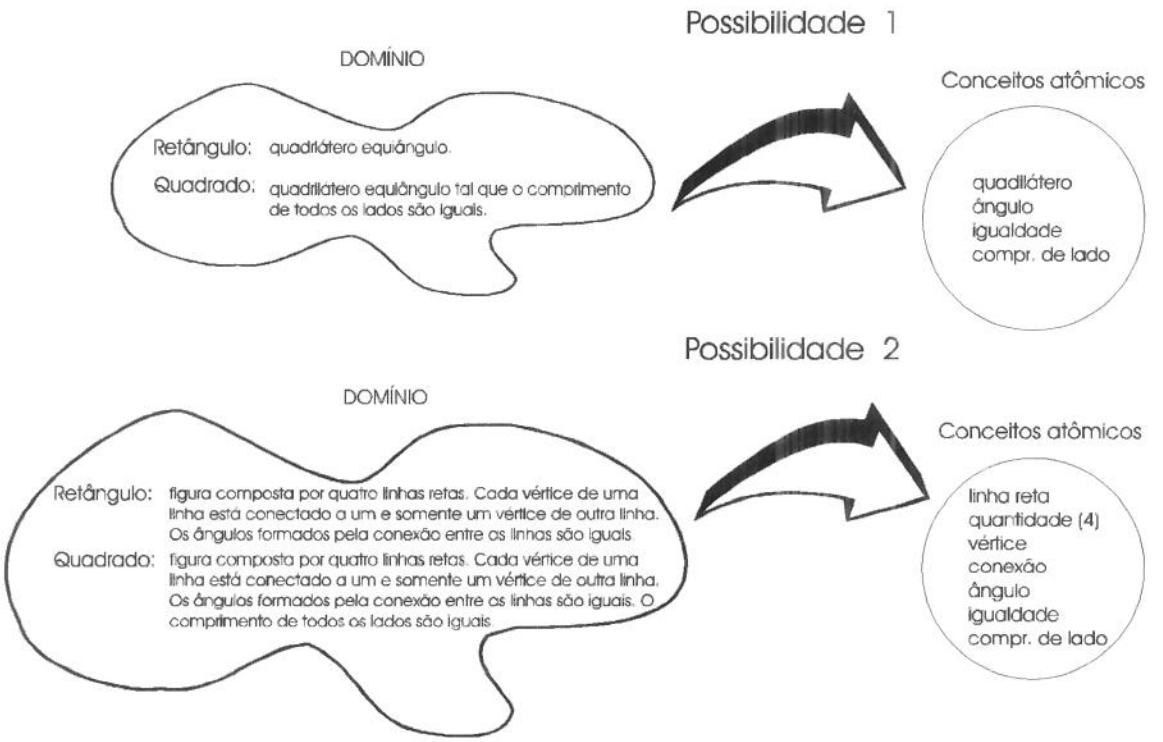


Fig. 4.2: Dois conjuntos distintos de conceitos atômicos para um domínio formado por dois elementos: o retângulo e um quadrado

Quando estamos identificando e escolhendo os conceitos atômicos para um determinado domínio de conhecimento não devemos pensar na forma de implementação dos mesmos em um ambiente para modelagem. A forma de implementação de cada conceito atômico já é um grande problema por si só. Devemos, portanto, tratá-la à parte e com um método próprio. Dedicamos a próxima seção a este assunto. Contudo, apenas para ilustrar, vejamos as possibilidades distintas de implementação de um ambiente para modelagem a

que poderia conduzir cada um dos dois conjuntos de conceitos atômicos da figura 4.2. A possibilidade 1 poderia levar à implementação de uma única primitiva. Trata-se de se implementar, no ambiente para modelagem, uma primitiva capaz de desenhar um quadrilátero e que aceitasse como parâmetros, também, os ângulos entre os lados e os comprimentos dos lados. Pode-se notar, que esta única primitiva implementa os conceitos atômicos presentes na possibilidade 1: quadrilátero, ângulo, comprimento de lado e igualdade. Já a possibilidade 2 poderia dar origem a um ambiente de modelagem composto de uma palheta de linhas de diferentes tamanhos, o que implementaria os conceitos atômicos de linha, vértice e comprimento de lado. O conceito atômico de conexão poderia ser implementado através da operação de se posicionar o vértice de uma linha sobre outra, após o que, um duplo *click* no botão do *mouse* poderia fazer a conexão. Ou desfazê-la, caso as linhas já estiverem conectadas. O conceito atômico de ângulo, poderia ser implementado criando-se uma operação, através do arraste do *mouse,* capaz de dar liberdade de giro à ponta livre de uma linha, conectada a outra por um vértice já fixado. Já os conceitos atômicos de quantidade e igualdade estariam implementados através da possibilidade de construção de figuras com diferentes quantidades de linhas de tamanhos variados[1] .

Antes de continuarmos, permita-nos revisar a terminologia que estamos usando. Empregamos o termo "elemento do domínio" ou, simplesmente "elemento", para descrever um objeto ou um fenômeno que compõe o domínio. O termo "conceito atômico" refere-se a uma unidade elementar que pode ser utilizada para modelar os elementos do domínio. O conceito atômico, como o próprio nome sugere, não tem existência física; ele existe no plano de nosso pensamento e representa nosso entendimento sobre um determinado elemento. Empregaremos o termo "primitiva" para designar o conceito atômico fisicamente implementado em um determinado ambiente para modelagem. Ou seja, a primitiva é o conceito atômico tornado concreto em um ambiente para modelagem.

Como afirmamos, vários conjuntos de conceitos atômicos podem ser estabelecidos para um mesmo domínio de conhecimento. A propósito, além dos conjuntos ilustrados na figura 4.2, poderíamos até ter definido um outro, baseado no conceito de deslocamento e

[1] Ainda que não seja nosso objetivo, poderiamos, também, usar o conjunto de conceitos atômicos da possibilidade 2 para desenvolver um brinquedo composto de peças retas de plástico de diferentes tamanhos, dotadas de elemento que permita a conexão das pontas e de tal modo que admitisse articulação entre duas peças conectadas, brinquedo este semenhante aos conjuntos para montagem LEGO System da LEGO do Amazonas Brinquedos LTDA ou o CRIE E MONTE da Bandeirantes Indústria de Brinquedos. Salientamos que todos os conceitos atômicos estão, novamente, implementados.

no conceito de rotação, o que poderia nos levar a desenvolver um ambiente para modelagem bem parecido com o ambiente de programação Logo (geometria) definido por Papert (1980). Contudo, fica a pergunta: de todos os conjuntos de conceitos atômicos que podemos obter, qual é o mais adequado?

Esta pergunta nos leva a pensar em uma classe de usuários a privilegiar no *design*. Usuários estão inseridos em cultura específica e trazem consigo certos conhecimentos anteriores. Assim, por exemplo, é possível que uma certa classe de usuários não possua o conceito de quadrilátero, que está presente no primeiro conjunto de conceitos atômicos, e possua todos os conceitos do segundo conjunto. Logo, para esta classe de usuários, pode ser preferível implementar um ambiente para modelagem baseado no segundo conjunto de conceitos atômicos do que no primeiro. Esta discussão introduz um novo e importante elemento na nossa estratégia de *design* de ambientes para modelagem: a **classe de usuários a privilegiar** no *design* do ambiente para modelagem. Cabe observar, que a escolha e a conseqüente implementação de um ambiente para modelagem que privilegie uma determinada classe de usuários, não implica na impossibilidade de uso deste ambiente por outros usuários que não conheçam alguns dos conceitos atômicos utilizados no desenvolvimento do mesmo. É bem verdade que usuários nesta condição terão que desenvolver esforços adicionais aos que, normalmente, serão desenvolvidos por usuários da classe privilegiada: eles terão que assimilar, interagindo com o ambiente, os conceitos atômicos que, por ventura, não possuam.

Embora a definição de uma classe de usuários a ser privilegiada pelo *design* possa contribuir no processo de escolha do conjunto de conceitos atômicos, esta definição, por si só, não é suficiente. Uma vez que vários conjuntos de conceitos atômicos podem ser de conhecimento da classe de usuários a ser privilegiada pelo *design* , perguntamos: qual, dentre estes vários conjuntos, é o mais adequado? **O princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos**, se propõe a dar uma possível resposta a esta pergunta. Por modelo conceitual mínimo entendemos que, de todos os conjuntos de conceitos atômicos estabelecidos para um domínio de conhecimento o mais adequado é aquele que apresenta o menor número de conceitos. E por máxima granulosidade dos conceitos atômicos entendemos que, de todos os conceitos atômicos que podem ser utilizados na expressão de um elemento do domínio de conhecimento, o mais adequado é o conceito mais próximo do elemento do domínio. Maximizando a granulosidade dos conceitos atômicos atuamos no sentido de diminuir o esforço cognitivo do usuário no desenvolvimento de modelos. E, minimizando o modelo

conceitual chegamos a estruturas atômicas que, pelos seus altos níveis de fragmentação, podem aumentar o controle do usuário sobre o ambiente para modelagem. Assim, o princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos se traduz na "válvula mestre" que permite regular o *design* dos ambientes de modelagem a Ambientes para Modelagem Orientados a um Domínio (AMOD). Vimos, no capítulo 2, que os AMOD possuem a virtude de apresentar simultaneamente, um equilíbrio entre o esforço cognitivo (do usuário) e o nível de controle (do usuário sobre o ambiente) encontrados em todo um espectro de ambientes, que podemos imaginar inseridos em uma linha que se estende desde os ambientes puramente de simulação (baixo esforço cognitivo mas baixo nível de controle) até as linguagens de programação de propósito geral (alto nível de controle mas alto esforço cognitivo).

O princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos nos leva a refletir sobre a própria natureza estrutural do conhecimento. Por exemplo:

- o Universo, a Via Láctea como uma galáxia dentro do Universo, o Sistema Solar dentro da Via Láctea, o planeta Terra no Sistema Solar, o Brasil dentro da Terra, ...;
- o Ecosistema, o ser humano dentro do Ecosistema, o sistema circulatório dentro do ser humano, o coração como órgão pertencente ao sistema circulatório, o ventrículo dentro do coração, o tecido muscular que compõe o ventrículo, as células que compõe o tecido muscular, o complexo de Golgi como uma organela celular, ...

O princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos tenta responder uma questão importante, relacionada à natureza do conhecimento, inerente a qualquer projeto: em que nível é conveniente parar? Adaptamos uma uma história, citada por Yourdon (1989) e que foi narrada por Eberhard (1970), para ilustrar a importância do referido princípio: "O pedido da maçaneta".

> Tive esta experiência em Washington, quando eu tinha dinheiro para gastar. Eu fiz um contrato com um *designer* e disse: "A maçaneta da porta do meu escritório foi elaborada com pouca imaginação e possui um *design* muito pobre. Você me desenvolveria outra maçaneta?" Ele respondeu "Sim" e após estabelecermos o preço, ele foi embora. Uma semana mais tarde ele voltou e disse-me: "Sr. Eberhard, antes de me preocupar com a maçaneta quero estudar a mão humana e o que o homem é capaz de fazer com ela." Eu disse: "Que bom, gosto de imaginação, prossiga". Ele voltou um tempo mais tarde e disse: "quanto mais eu penso sobre isto, surge o problema da adaptação. O que quero estudar, em primeiro lugar, é como o metal é

formado e quais as tecnologias para se fabricar peças metálicas, de modo que eu possa conhecer os verdadeiros parâmetros para o ajuste da mão". Eu respondi "Ótimo". Mas aí ele disse: "Estive estudando a constituição do metal e tudo depende das propriedades metalúrgicas. Preciso estudar metalurgia por três meses para compreender melhor o problema. "Excelente!", afirmei-lhe. Após três meses ele voltou e disse: "Sr. Eberhard, quanto mais estudo metalurgia, mais percebo que é a estrutura atômica que está no centro do problema". E, assim, o *designer* está estudando a física atômica a partir da maçaneta e, ainda, não resolveu o problema da maçaneta.

4.4 - DESENVOLVENDO O *DESIGN* DE CADA CONCEITO ATÔMICO E DO AMBIENTE PARA MODELAGEM COMO UM TODO

Uma vez conhecidos todos os conceitos atômicos a implementar, a tarefa agora é: como implementá-los com os recursos disponíveis? E esta pode não ser uma tarefa simples, dada as limitações impostas pela tecnologia corrente: tipos e natureza dos mecanismos de entrada, memória disponível, velocidade de CPU, forma e variedade de saídas necessárias etc..

O entendimento da maneira como as pessoas interagem com o computador é fundamental para se estabelecer o *design* do ambiente para modelagem. A teoria da ação (Norman, 1986) propõe que quando uma pessoa interage com um sistema, os seus objetivos são expressos em termos relevantes para ela (termos psicológicos) e os possíveis estados e mecanismos do sistema são expressos em termos relativos a ele (termos físicos). Norman vê esta discrepância como dois golfos existentes entre a pessoa e o sistema: o golfo da execução e o golfo da avaliação (figura 4.3). Estes golfos necessitam ser transpostos para que a interação entre a pessoa e o sistema seja efetiva.

Fig. 4.3: Os golfos existentes entre uma pessoa e um sistema (Norman, 1986)

A existência de dois golfos entre o sistema e o usuário conduz a um requerimento crítico no *design* do ambiente para modelagem: como atravessar estes golfos. Só existem duas formas de se fazer isto. Ou o usuário se especializa no uso do sistema ou o *designer* constrói estruturas no sistema tais que se adequem às necessidades do usuário. Como o que queremos, é o desenvolvimento de sistemas que são ambientes para modelagem em um contexto educacional e sabendo-se que não é objetivo do usuário aprender sobre o ambiente de modelagem mas, sim, através da atividade de modelar, concluímos que precisamos projetar sistemas que vençam boa parte dos golfos existentes.

Norman (1986) indica, ainda, sete estágios envolvidos no processo de ação da pessoa sobre um sistema. O primeiro estágio é o **estabelecimento do objetivo**. Então, para efetuar a ação a pessoa deve passar por três outros estágios: **formação da intenção**, **especificação da seqüência de ações** e **execução da ação**. Para receber o efeito da ação sobre o sistema, a pessoa precisa de mais três estágios: **perceber o estado do sistema**, através dos elementos deste, projetados para exibi-los, **interpretar o estado** e **avaliar o estado interpretado** com relação aos objetivos e intenções originais.

Para Norman (1986), apresentação visual pode auxiliar em vários estágios de atividade. Ela dá suporte à geração de intenção por lembrar ao usuário o que é possível fazer. Ajuda a especificação de ações por tornar visível os elementos respectivamente associados às ações. Auxilia a execução, especialmente se a execução pela ação de apontar é possível. Auxilia a avaliação por tornar possível a existência de lembranças visuais do que foi feito. Ainda, estruturas visuais podem auxiliar a interpretação. Mas, sobretudo, entendemos que, no mínimo dois princípios devem guiar o *design* de ambientes para modelagem centrados na ação do usuário: o princípio da redução da distância e o princípio do engajamento direto.

O **princípio da redução da distância** estabelece que o *design* do sistema (o ambiente para modelagem) deve ser tal que a distância entre este e o usuário deve ser mínima. Em outras palavras, os golfos da avaliação e execução, existentes entre usuário e ambiente de modelagem, devem ser mínimos. E, distância mínima significa que o trabalho de interação do usuário com o sistema é simples e direto, ou seja, que os pensamentos deste são prontamente traduzidos nas ações físicas requeridas pelo sistema, e que a saída do sistema se apresente em uma forma que pode ser facilmente interpretada em termos dos objetivos de interesse do usuário.

Hutchins (1986) referiu-se à redução de distância quando de seu estudo sobre interfaces de manipulação direta. Ele identificou duas formas de distância: (1) a **distância semântica**, que reflete o relacionamento entre a intenção do usuário e o significado de uma expressão na linguagem do sistema, seja na execução ou na avaliação e (2) a **distância articulatória**, que reflete o relacionamento entre a forma física da expressão na linguagem do sistema e o seu significado, novamente, nos dois golfos. A figura 4.4 ilustra isto.

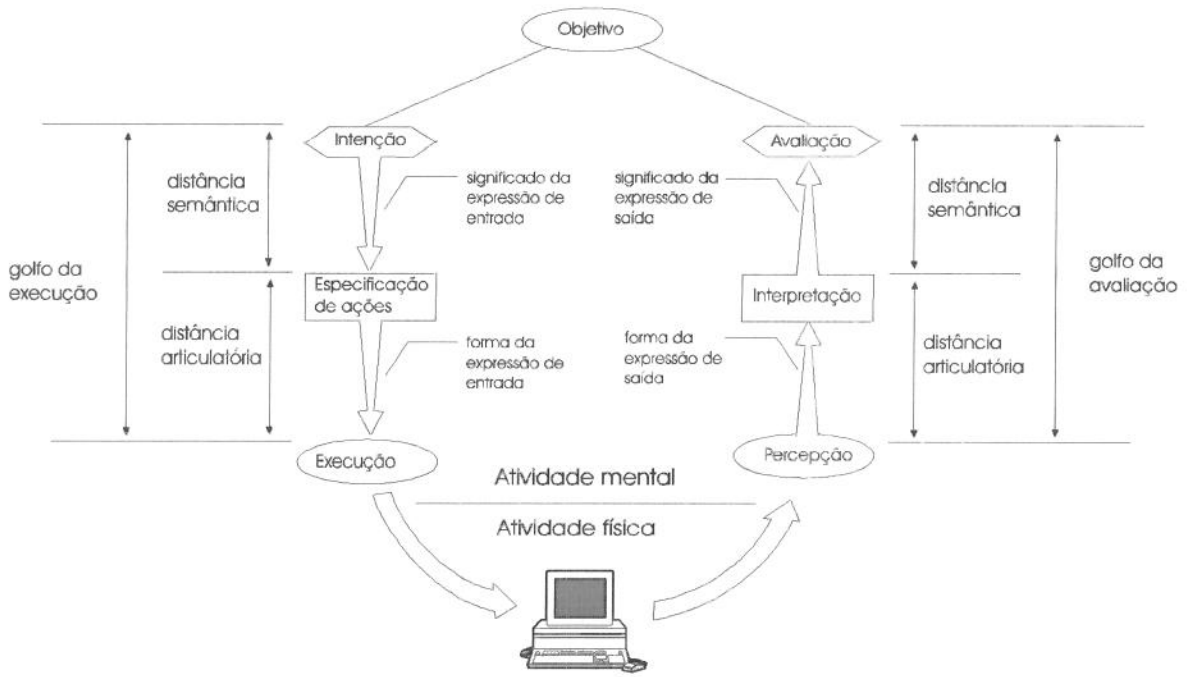


Fig. 4.4: As distâncias semântica e articulatória nos golfos de execução e avaliação (traduzida de Hutchins *et alli,* 1986)

A distância semântica pode ser reduzida propondo-se um conjunto de conceitos atômicos adequado para a classe de usuários a privilegiar no *design* . Assim, haverá um casamento da intenção do usuário com os elementos disponíveis no sistema, o ambiente para modelagem. Aliás, identificar e escolher um conjunto de conceitos atômicos adequado a uma classe de usuários, é exatamente o que prescreve as orientações da seção anterior. Portanto, seguindo-as entendemos que estaremos trabalhando no sentido de reduzir a

distância semântica. Para Hutchins, a distância articulatória pode ser reduzida criando-se vocábulos na linguagem do sistema tais que permitam especificar uma ação por meio de uma imitação, existindo, assim, uma similaridade articulatória entre o item do vocabulário e o seu significado. Em geral, a redução da distância articulatória pode depender da tecnologia de entrada/saída disponível. O *mouse,* por exemplo, é um dispositivo que permite mímica sobre espaço. Aliás, o *mouse* é, também, usual para uma grande variedade de ações porque nós podemos mapear vários fenômenos (mesmo os não intrinsecamente espaciais como, por exemplo, escolher, pela ação de apontar, uma entre várias opções) segundo a metáfora espacial. Neste aspecto, som em linguagem falada pode ser muito interessante como saída e/ou entrada, assim como dispositivos sensíveis ao toque, localização e tempo e, também, à pressão e ao torque (Buxton, W., 1986). O uso de linguagens pictoriais ou "icônicas"[2] é visto por Hutchins (1986) como uma outra maneira de se conseguir, nos dias atuais, similaridade articulatória. Adicionalmente, nós entendemos que o uso de dispositivos que permitam a participação virtual de um usuário (realidade virtual) em um ambiente para modelagem, em muito pode contribuir para a redução da distância aticulatória bem como pode estimular este usuário a um engajamento direto com os objetos do ambiente.

O **princípio do engajamento direto** estabelece que o *design* do sistema (o ambiente para modelagem) deve ser tal que convide o usuário (o modelador) para um engajamento direto com os objetos representados pelo sistema. Ou seja, o ambiente para modelagem deve se apresentar para o modelador como se fosse o próprio domínio de conhecimento. Em outras palavras, não deve existir intermediário entre o modelador e os objetos do domínio de conhecimento. O ambiente deve criar, para o modelador, a sensação dele estar agindo sobre os próprios objetos do domínio.

Acreditamos que Papert (1980) toca, ainda que implicitamente, no princípio do engajamento direto, quando se refere a invenção dos **objetos-de-pensar-sobre**, objetos estes que, segundo ele, criam uma possibilidade de identificação pessoal, onde a tartaruga Logo é um exemplo. Similar, também, ao princípio do engajamento direto é o **princípio de representação em primeira-pessoa** proposto por Laurel (1986). Laurel cita como exemplo, o jogo *Atari pole-position* onde o usuário dirige um carro de corrida em uma pista. O usuário controla a velocidade do carro pressionando um pedal que é análogo ao acelerador de um automóvel. A habilidade do usuário participar da fantasia de pilotar o

[2] Estamos derivando o termo "icônicas" do termo ícones, os quais são objetos gráficos que possuem a virtude de lembrar o seu significado.

carro de corrida com o sentimento de primeira-pessoa é, significativamente, menor se o meio de controlar a velocidade do carro deixar de ser através do pedal para passar a ser pelo teclado. Ou, igualmente, o sentimento de primeira-pessoa é diminuído se a ação de apertar o pedal, não for refletida por uma representação animada do carro, mas, por um *display* numérico de velocidade.

O engajamento direto ocorre quando o usuário experimenta a interação direta com os objetos do domínio. Existe um sentimento de envolvimento mais direto com o mundo dos objetos do que com a comunicação, que é uma intermediária. As ações são aplicadas sobre os objetos e os resultados destas ações, também, são observados nos próprios objetos, sendo que, o ambiente para modelagem e o computador se tornam invisíveis. Contudo, atualmente, muito pouco se sabe sobre metodologias para se aplicar este princípio. Ainda assim, Laurel (1986) apresenta algumas orientações: (1) o sistema (ambiente para modelagem) deve prover uma representação para os próprios objetos do domínio sobre o qual ele foi desenvolvido; (2) as linguagens de entrada e saída do sistema devem ser inter-referenciáveis, permitindo uma expressão de entrada incorporar ou fazer uso da expressão anterior. Isto é fundamental para que se crie, para o usuário, a ilusão de estar manipulando os próprios objetos do domínio; (3) o sistema deve reagir, prontamente, às ações do usuário. Nenhum intervalo de tempo deve haver entre a execução de uma ação e o resultado desta, exceto quando o intervalo de tempo for inerente ao domínio.

Hutchins (1986), acrescenta a estas orientações, que objetos complexos podem ser vistos de diferentes perspectivas, comportando-se completamente diferente em cada ponto de vista. Ele cita, como exemplo, a representação de um arquivo. Se a representação de um arquivo, criada por um sistema, for o seu nome ou um ícone associado a este, então esta irá trabalhar bem para operações tal como eliminar o arquivo, renomeá-lo etc.. Contudo, temos que reconhecer que esta representação não é boa para operações como a de trocar o atributo de proteção do arquivo. Logo, a reorganização de um objeto, em um ou mais pontos de vista, pode ser freqüentemente necessária.

Neste aspecto, podem nos ajudar as considerações de Bruner (1966). Ele afirma que qualquer objeto em um domínio pode ser representado de três formas: (1) por um conjunto de ações apropriadas para se obter determinado resultado (**representação ativa**), (2) por um conjunto de imagens resumidas ou gráficas que o representam, ainda que não completamente (**representação icônica**) e (3) por um conjunto de proposições lógicas ou

simbólicas, derivadas de um sistema simbólico e regido por normas ou leis para formar ou transformar as proposições (**representação simbólica**). Bruner (1966) concretiza a distinção entre estas representações empregando uma balança de travessão[3]. Pode-se entender a balança comparando-a com uma gangorra: sabe-se que para abaixar um dos seus lados, tudo o que se tem a fazer é deslocar-se para fora do seu centro. A balança pode, também, ser representada através de um desenho onde se mostre as hastes de sustentação, o travessão com os ganchos que se prestam para pendurar os pesos em forma de anéis. Temos finalmente que a balança pode ser descrita em linguagem corrente, sem o auxílio de diagramas, ou, ainda, matematicamente, com base na lei dos momentos de Newton. É desnecessário dizer que cada objeto ou conceito em um domínio, pode ter vocação para ser representado segundo uma ou outra classe de representação. Assim, objetos do domínio Legal dificilmente serão representados através de diagramas, enquanto que objetos do domínio da Geografia normalmente levam a pensar em ilustração.

4.5 - RESUMO DAS ESTRATÉGIAS

Resumimos, a seguir, as estratégias para o *design* de ambientes computacionais para modelagem em um contexto educacional.

1. **Definir o domínio de conhecimento** para o qual se quer desenvolver o ambiente de modelagem.

2. Analisar o domínio de conhecimento selecionado e **expressá-lo em termos de alguns dos elementos que o compõem**. Isto delimita o domínio de conhecimento selecionado do resto do mundo conhecido (princípio da restrição de domínio).

3. **Identificar e escolher um conjunto de conceitos atômicos** para guiar a implementação do ambiente de modelagem. Vários conjuntos de conceitos atômicos podem, em princípio, ser estabelecidos. Contudo, deve-se procurar estabelecer um conjunto que privilegie uma certa classe de usuários. Afinal, pode não ser possível obter um modelo conceitual que sirva, com sucesso, a usuários de diferentes culturas

[3] Balança formada por um travessão suspenso e preso no seu centro por duas hastes verticais. O travessão é composto por ganchos simétricamente dispostos em relação ao seu centro. Pode-se pendurar os pesos em forma de anéis nestes ganchos.

e de diferentes níveis de conhecimento. No entanto, para uma mesma classe de usuários é possível que haja várias possibilidades de escolha do conjunto de conceitos atômicos. Deve-se preferir o conjunto de conceitos atômicos que apresentar um equilíbrio entre duas vertentes: um o menor número de conceitos atômicos com maior granulosidade possível (princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos).

4. Finalmente, deve-se estabelecer que forma de implementação tomará, no ambiente para modelagem, cada conceito atômico. Ou, em outras palavras, deve-se **definir a forma física de cada conceito atômico**. Isto depende, entre outras coisas, da tecnologia disponível para a implementação: CPU, memória, discos, linguagens para implementação disponíveis, periféricos para entrada e saída etc.. É importante, nesta atividade, compreender como as pessoas interagem com os sistemas físicos. Deve-se criar estruturas que encurtem a distância entre o usuário e o ambiente para modelagem e que o convidem para um engajamento direto com o ambiente (princípio da redução da distância e princípio do engajamento direto). Estruturas adicionais àquelas que visam implementar os conceitos atômicos podem ser necessárias. Tais estruturas criam a infra-estrutura necessária à existência dos elementos que implementam os conceitos atômicos. Em geral, são exemplos de tais estruturas: gravar em disco um modelo criado e fornecer ajuda ao uso do ambiente (*help*).

4.6 - AS ESTRATÉGIAS DISCUTIDAS DIANTE DE ALGUNS MÉTODOS TRADICIONAIS PARA ANÁLISE, PROJETO E IMPLEMENTAÇÃO DE SISTEMAS

Linguagens de programação como COBOL não apresentam mecanismos capazes de isolar estruturas de dados definidas no interior de um programa. Isto é, o escopo de qualquer estrutura de dados é global ao programa onde está definida; os valores das estruturas de dados podem ser alterados em qualquer ponto do programa. Em contrapartida, linguagens verdadeiramente estruturadas, como Pascal ou C, oferecem mecanismos, denominados procedimentos ou funções, que são capazes de isolar estruturas de dados em seu interior, estruturas estas, normalmente caracterizadas na literatura, como locais ou privativas. As linguagens orientadas a objetos vão um pouco mais longe. Além dos dispositivos capazes de isolar dados em seu interior, elas apresentam um novo elemento, o objeto, que é capaz

de isolar dentro de si, código e estruturas de dados. Estrutura de dados e código são elementos relacionados ao computador e às linguagens de programação. Não encontramos tais elementos nos Ambientes para Modelagem Orientados a um Domínio (AMOD), a não ser que o domínio englobe o computador e as linguagens de programação. O que um AMOD apresenta é um conjunto de primitivas pertencentes ao domínio em que ele está orientado (figura 4.5).

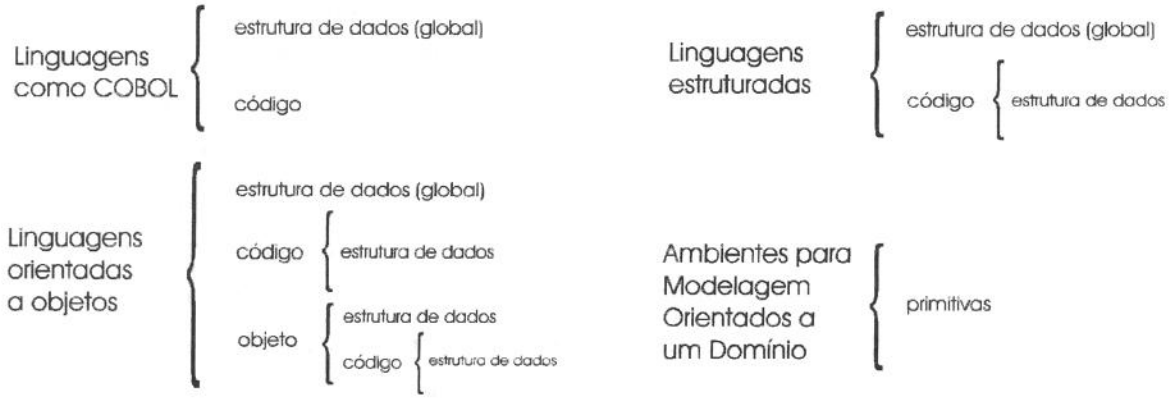


Fig. 4.5: Elementos utilizados no desenvolvimento de aplicativos em alguns ambientes

Neste aspecto o elemento "objeto", encontrado nas linguagens orientadas a objetos, é um ancestral das primitivas encontradas nos AMOD. O objeto é um elemento anfíbio que ainda não deixou o "mar do computador e suas linguagens de programação" mas que já pode habitar a "terra do domínio" para o qual foi desenvolvido. Dizemos que não deixou "mar do computador e suas linguagens de programação" porque possui código e estrutura de dados que precisam ser definidas. Dizemos que pode habitar a "terra de um domínio" porque, após programado, ele pode ser utilizado como uma primitiva do domínio, atuando de forma similar às primitivas dos AMOD. A programação, por exemplo, em uma linguagem estruturada ou orientada a objetos é completamente diferente da "programação" em um AMOD. O programador de uma linguagem estruturada ou orientada a objetos deve se preocupar com os elementos do domínio da aplicação e com os elementos do mundo destas linguagens, os quais são necessários para descrição da aplicação neste domínio. O "programador" de um ambiente para modelagem orientado a um domínio, o modelador, deve prender sua atenção, apenas, nos elementos do domínio.

Descobrir as primitivas de um domínio significa achar fragmentos deste domínio de tal sorte que os objetos e fenômenos do domínio possam ser expressos a partir destes fragmentos. Isto é inteiramente diferente, por exemplo, de se definir um domínio a partir dos objetos que o compõe (Análise Orientada a Objetos) ou de estabelecer o domínio a partir de seus processos, dados e comportamento tempo-dependente (Análise Estruturada). As estratégias descritas neste capítulo não se apoiam em qualquer paradigma tradicional para se analisar e projetar um sistema. Seguindo estas estratégias estaremos praticando um paradigma alternativo para se descobrir as primitivas do domínio e projetá-las. O próximo capítulo mostra como fizemos isto para desenvolver um protótipo de um ambiente para modelagem no domínio da Mecânica.

Capítulo 5

Emprego das estratégias propostas ao *design* de um ambiente para modelagem

Objetivamos, neste capítulo, exemplificar a aplicação das estratégias propostas ao *design* de um ambiente para modelagem orientado a um domínio. Escolhemos trabalhar com o domínio da Mecânica dos Corpos Rígidos. Após uma breve discussão sobre o poder de expressão de uma linguagem em descrever os conceitos subjacentes a um ou outro domínio, o restante do capítulo descreve a aplicação das estratégias discutidas, no capítulo 4, ao *design* um de ambiente para modelagem orientado ao domínio que escolhemos.

Tópicos deste capítulo:

Tópico		*Página*
5.1 -	Introdução	53
5.2 -	Formalismos para representação dos conceitos atômicos	54
5.3 -	Os elementos do domínio e a identificação dos conceitos atômicos subjacentes	56
5.4 -	Desenvolvimento do *design* de cada conceito atômico	85
5.5 -	Especificando a infra-estrutura do ambiente	92

5.1 - INTRODUÇÃO

Este capítulo exemplifica a aplicação das estratégias para *design* de ambientes para modelagem orientados a um domínio (AMOD) enunciadas no capítulo 4. Na exemplificação precisávamos de um domínio e para isto escolhemos o domínio da Mecânica; para ser mais exato, escolhemos trabalhar com um subdomínio da Mecânica conhecido como Mecânica dos Corpos Rígidos. A Mecânica é caracterizada, segundo Beer e Johnston (1980 e 1991), pela descrição das condições de repouso e de movimento dos corpos sendo que, na Mecânica dos Corpos Rígidos[1], os corpos são considerados perfeitamente rígidos ou seja, não sofrem deformações significativas quando estão sujeitos à ação de forças. Neste texto, estaremos chamando o domínio da Mecânica dos Corpos Rígidos, apenas, por Mecânica, a menos que a diferença entre os dois termos seja relevante e se faça necessária em algum ponto. Por razões de simplicidade de representação e implementação do ambiente para modelagem, assumiremos no trabalho de *design*, uma representação bidimensional para dos fenômenos.

Estaremos sempre considerando que a classe de usuários a ser privilegiada no trabalho de *design* do ambiente para modelagem domina conceitos de Álgebra elementar e possui noções de Geometria Plana.

Consideraremos, nos trabalhos de *design*, a seguinte tecnologia: (1) como *hardware*, um microcomputador AT 486 equipado com monitor de vídeo super VGA colorido, *winchester* e *mouse* e (2) como *software*, o sistema operacional Windows e os ambientes para desenvolvimento de *software* ToolBook e o Borland C++. Atribuímos a escolha deste ambiente para desenvolvimento, principalmente, pelo fato de ser possível construir, neste ambiente, uma interface gráfica com o usuário (GUI), que de antemão, sabemos que favorece a expressão dos elementos do domínio escolhido. Mais tarde, no capítulo 7, daremos mais detalhes sobre esta tecnologia.

A seguir fazemos uma reflexão sobre o poder de expressão de um formalismo em descrever os conceitos subjacentes a um ou outro domínio. Isto, porque precisamos de um formalismo adequado à descrição dos fenômenos da Mecânica e, talvez, não seja o Português, a exemplo do que fizemos para o pequeno domínio de conhecimento considerado na discussões do capítulo 4, a linguagem mais apropriada. Após esta

[1] Não são considerados corpos rígidos os fluidos (líquidos e gases) e os corpos que sofrem deformações tais que influenciam as suas condições de repouso ou equilíbrio dentro do fenômeno que participam.

reflexão, iniciaremos a aplicação das estratégias, enunciadas no capítulo 4, ao *design* de um ambiente para modelagem orientado ao domínio da Mecânica.

5.2 - FORMALISMOS PARA REPRESENTAÇÃO DOS CONCEITOS ATÔMICOS

O trabalho de identificação dos conceitos atômicos de um certo domínio de conhecimento consiste da expressão de cada um dos elementos do domínio, escolhidos para representá-lo, em função de conceitos mais elementares. Para expressar um elemento em função dos seus conceitos elementares nós necessitamos de alguma linguagem. Por exemplo, para o pequeno domínio considerado na seção 4.3 do capítulo 4, utilizamos a linguagem natural (Português) para expressar os elementos quadrado e retângulo em função de conceitos mais elementares. As linguagens naturais, como o Português, embora possam ser utilizadas para expressar os elementos de qualquer domínio, nem sempre possuem poder de expressão para defini-los de forma concisa. Concordamos com Bruner (1966) que alguns formalismos para representação são mais econômicos que outros. Assim, é mais econômico usar a linguagem matemática para dizer

$$2 + 3 = 5,$$

do que dizer

a soma de dois com três é igual a 5.

Assim, também, é bem provável que um gráfico expresse com muito mais economia os limites territoriais político-administrativo de um dado país do que algumas centenas de milhares de palavras em uma linguagem natural. Com o intuito de exemplificar isto, através do uso de um elemento no domínio de nosso interesse, consideremos o fenômeno Mecânico tradicionalmente conhecido como "movimento retilíneo uniforme de um corpo", ou seja, movimento em linha reta de um corpo sob velocidade constante, apresentado na figura 5.1 através de duas representações diferentes.

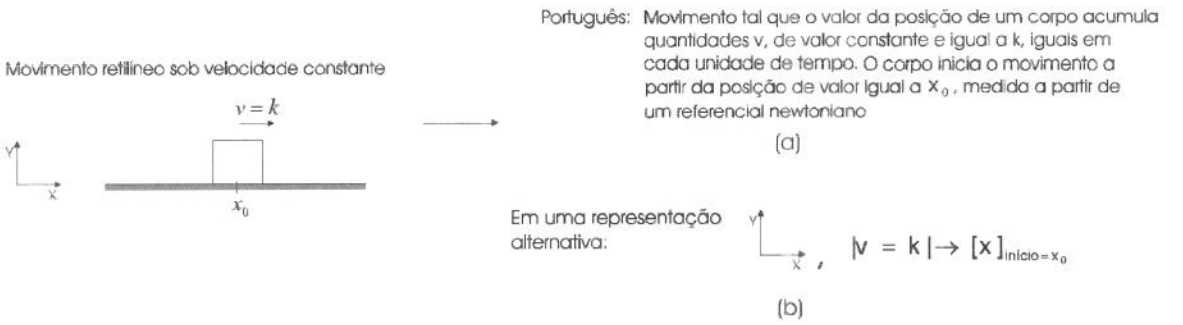


Fig. 5.1: Duas representações diferentes para definir o mesmo fenômeno

A figura 5.1-a mostra o movimento descrito em Português. Considerando que x representa a posição do corpo e que

representa o referencial newtoniano[2], a figura 5.1-b mostra o movimento segundo uma notação alternativa baseada na construção

$$|v = k| \rightarrow [x]_{início = x_0}$$

que indica que x é uma quantidade que acumula valores iguais a k em cada unidade de tempo e que o valor inicial de x é x_0. Mais tarde, neste capítulo, formalizaremos esta notação. Constatamos, novamente, que as duas representações apresentam formas economicamente distintas para descrever o mesmo fenômeno Mecânico e que a forma apresentada na figura 5.1-b é mais econômica do que a da figura 5.1-a.

Neste capítulo, não utilizaremos uma linguagem natural para descrever os conceitos atômicos envolvidos em cada elemento escolhido para representar o domínio da Mecânica. Criaremos representações próprias adaptadas à expressão que se fizer necessária e explicaremos o significado dos símbolos do formalismo que estaremos utilizando.

[2] Um referencial newtoniano é um referencial fixo, isto é, não possui movimento em relação aos fenômenos que descreve.

5.3 - OS ELEMENTOS DO DOMÍNIO E A IDENTIFICAÇÃO DOS CONCEITOS ATÔMICOS SUBJACENTES

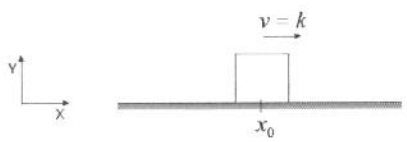
Pela nossa estratégia, necessitamos identificar um conjunto de elementos, dentro do domínio da Mecânica, que o represente satisfatoriamente. O domínio da Geometria Plana contém retângulos, triângulos, círculos etc. e certamente estes elementos são candidatos em potencial a representantes deste domínio. Mas quais são os elementos do domínio da Mecânica? Este domínio contém corpos que possuem comportamento mecânico, isto é, corpos que podem se movimentar ou permanecer em repouso à medida em que o tempo transcorre. E são exatamente estes corpos, com os seus comportamentos, que são os elementos do domínio da Mecânica. No entanto, dizer isto desta maneira não ajuda muito. Em nossa estratégia, precisamos identificar quais são os "retângulos", os "quadrados", os "triângulos" etc. do domínio da Mecânica. Pois, dizer que a Mecânica é composta de corpos com comportamento mecânico, é o mesmo que dizer que o domínio da Geometria Plana é composta de figuras. À primeira vista, os elementos da Mecânica não são tão aparentes quanto os elementos da Geometria Plana. Mas isto, é só à primeira vista. Um estudo na literatura pertinente conduziu-nos a uma série de classes de fenômenos mecânicos tradicionalmente nomeados, tais como: movimento retilíneo uniforme de um corpo, movimento retilíneo uniformemente acelerado de um corpo, movimento de um corpo sob a ação de forças etc.. Uma característica importante destes fenômenos é que eles se aplicam a qualquer corpo. Graças a esta característica expressaremos a seguir os fenômenos mecânicos independentemente dos corpos a eles associados. Isto é, estaremos interessados apenas no comportamento mecânico dos corpos sem levar em consideração os aspectos relacionados à aparência dos mesmos.

Temos dois objetivos, nesta seção: (1) mostrar os elementos do domínio da Mecânica que selecionamos para representá-lo, elementos estes relacionados ao comportamento mecânico dos corpos e (2) extrair destes elementos os conceitos atômicos necessários para a descrição de cada um dos elementos selecionados.

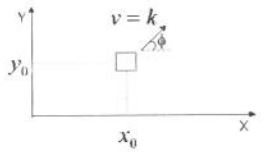
O estudo que fizemos do domínio permitiu-nos agrupar elementos (fenômenos da Mecânica) em algumas classes tradicionalmente referenciadas na literatura específica (Beer e Johnston, 1980 e 1991; Ratcliffe, 1992; Singer, 1977) . A figura 5.2 mostra, graficamente, os elementos que selecionamos. Estes elementos formam um conjunto bastante representativo do domínio. Cada um dos elementos da figura representa toda a

sua classe de fenômenos e não, apenas, um fenômeno. Isto é análogo ao que ocorreria se estivéssemos tratando do domínio da Geometria Plana e selecionássemos o elemento polígono como um de seus elementos representantes. Neste caso, o polígono representaria toda a sua classe: triângulos, retângulos, hexágonos etc.. Como dissemos, a Mecânica descreve as condições de repouso ou movimento dos corpos. Tanto repouso quanto movimento são termos relativos. Isto é, um corpo dotado de velocidade em relação a um referencial fixo é dito estar em movimento; em contrapartida, ele pode ser dito estar em repouso se estiver sendo observado a partir de um referencial que se movimenta com a sua velocidade. Daí, em todas as figuras, de 5.2-a até 5.2-y, aparece um par de eixos cartesianos que demarca a origem do sistema de medidas. Trata-se de um referencial fixo em relação aos fenômenos que descreve ou, usando uma terminologia apropriada, trata-se de um referencial newtoniano.

Movimento retilíneo sob velocidade constante

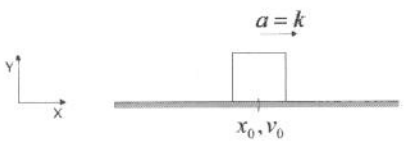


(a)

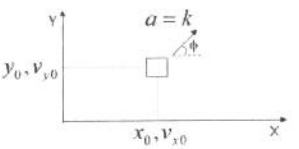


(b)

Movimento retilíneo sob aceleração constante

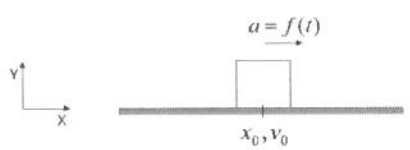


(c)

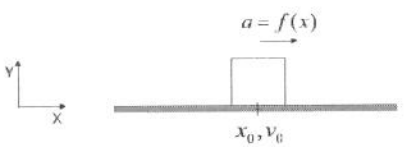


(d)

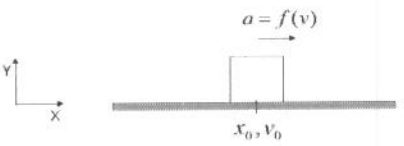
Movimento com aceleração e velocidade genéricas



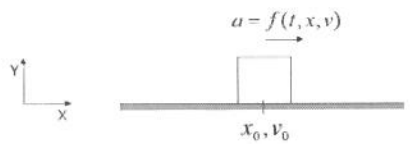
(e)



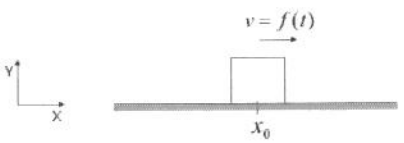
(f)



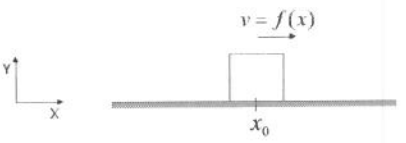
(g)



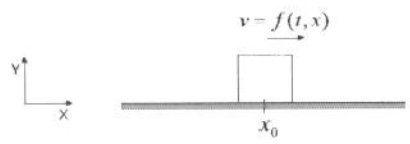
(h)



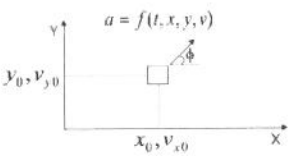
(i)



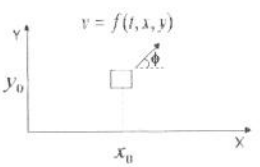
(j)



(k)

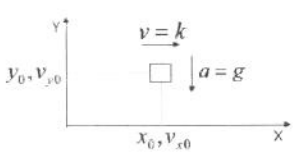


(l)



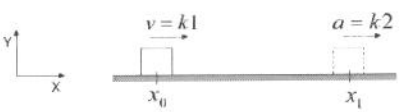
(m)

Movimento Composto 1

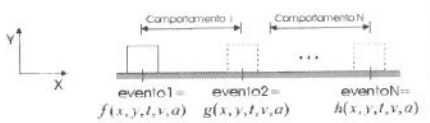


(n)

Vários comportamentos em um mesmo fenômeno



(o)



(p)

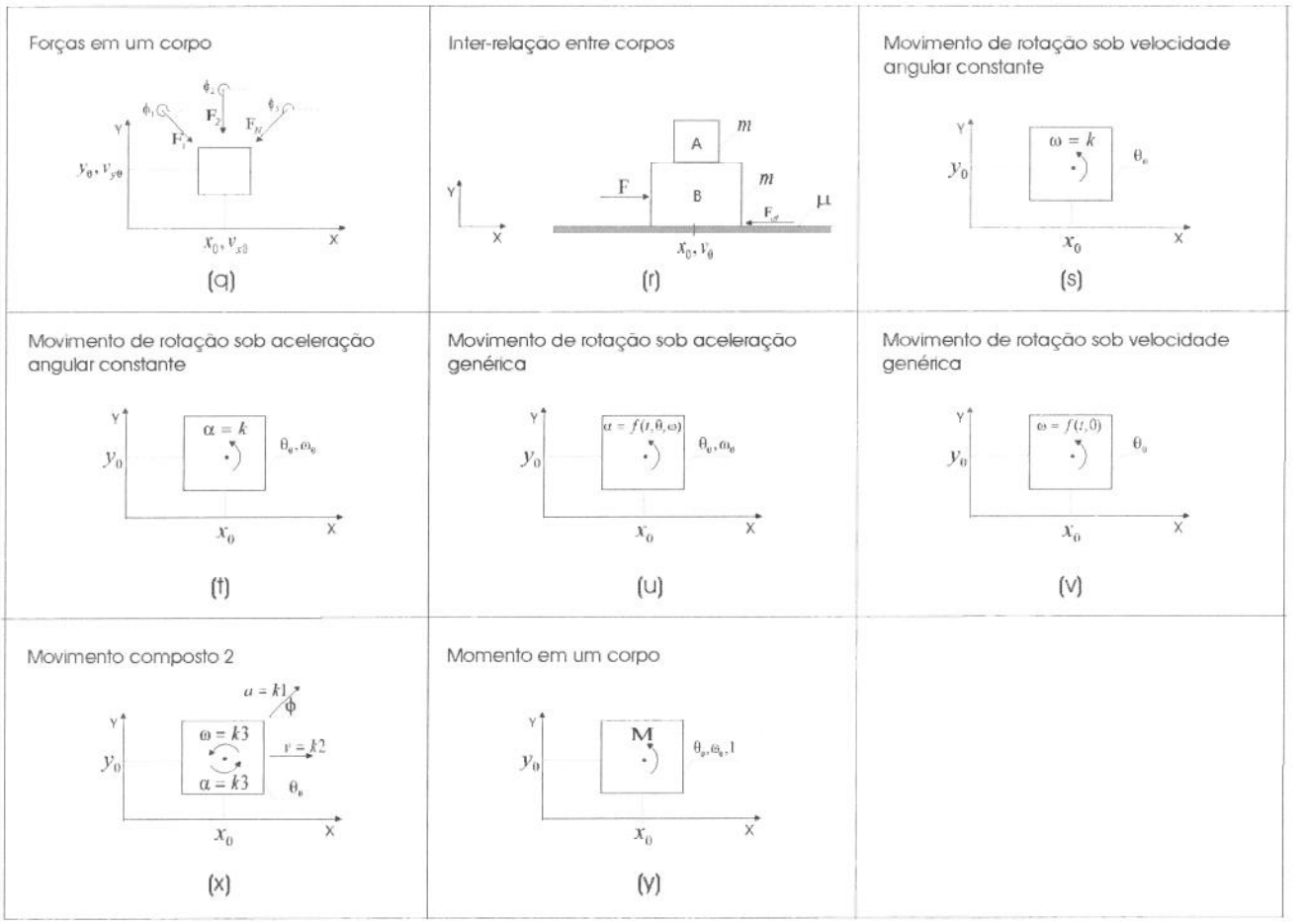


Fig. 5.2: Elementos do domínio da Mecânica selecionados para representa-la

Vejamos o que significa cada um dos elementos selecionados e que estão resumidos, graficamente, na figura 5.2:

ELEMENTO da figura 5.2	SIGNIFICADO
a e **b**:	é o clássico movimento retilíneo uniforme de um corpo, ou seja, movimento de um corpo em linha reta sob velocidade constante. A figura 5.2-a apresenta o movimento segundo uma das direções do par de eixos (X, no caso) que demarca a origem das medidas. Já a figura 5.2-b descreve o movimento do corpo segundo uma direção que forma um ângulo ϕ com o sentido positivo do eixo X. O valores x_0 e y_0 representam as posições iniciais do corpo ao longo das duas direções dos eixos cartesianos;
c e **d**:	representam o movimento retilíneo uniformemente acelerado, ou seja, movimento de um corpo em linha reta sob aceleração constante. A figura 5.2-c mostra o movimento segundo uma das direções do par de eixos (X, no caso) que demarca a origem e a figura 5.2-d segundo uma direção que forma um ângulo ϕ com o sentido positivo do eixo X. A figura 5.2-c mostra que o fenômeno mecânico se inicia com o corpo possuindo velocidade v_0 na posição x_0 e a figura 5.2-d mostra que o fenômeno se inicia com o corpo possuindo as componentes cartesianas da velocidade v_{x0} e v_{y0} na posição de coordenadas x_0 e y_0;
e até **m**:	descrevem o movimento de um corpo submetido a aceleração e velocidade genéricas. Todas estas figuras retratam o movimento retilíneo de um corpo segundo uma das direções do eixo que demarca a origem, exceto, as duas últimas, 5.2-l e 5.2-m, as quais apresentam um movimento segundo uma direção que forma um ângulo ϕ com o sentido positivo do eixo X. Os fenômenos descritos nas figuras de 5.2-e até 5.2-h mostram um corpo que, partindo de uma posição x_0 com velocidade v_0, apresenta uma aceleração que é, respectivamente, uma função do tempo ($a = f(t)$), uma função da posição do corpo ($a = f(x)$), uma função da velocidade instantânea do corpo ($a = f(v)$) e uma função das variáveis tempo, posição e velocidade ($a = f(t, x, v)$). Os fenômenos expostos pelas figuras 5.2-i até 5.2-k mostram um corpo que, partindo de uma posição x_0, apresenta velocidade que é, respectivamente, uma função do tempo ($v = f(t)$), uma função da posição do corpo ($v = f(x)$) e uma função das

variáveis tempo e posição ($v = f(t, x)$). A figura 5.2-l apresenta o movimento de um corpo cuja aceleração é função do tempo, posição e velocidade ($a = f(t, x, y, v)$). O fenômeno retratado nesta figura se inicia com o corpo possuindo as componentes de velocidade v_{x0} e v_{y0} na posição de coordenadas x_0 e y_0. A figura 5.2-m apresenta o movimento de um corpo que, partindo de uma posição de coordenadas x_0 e y_0, possui velocidade cujo valor é uma função do tempo e da posição assumida pelo corpo ($v = f(t, x, y)$);

n: mostra o comportamento mecânico de um corpo composto a partir de dois outros comportamentos: (1) mover com velocidade constante ($v = k$) ao longo do eixo X e (2) mover com aceleração constante de valor igual a **g**, ao longo da direção Y. O fenômeno se inicia com o corpo possuindo os componentes de velocidade v_{x0} e v_{y0} na posição de coordenadas x_0 e y_0;

o e **p**: cada uma destas figuras apresenta um fenômeno envolvendo vários comportamentos. A figura 5.2-o referencia o movimento retilíneo de um corpo sob velocidade constante $v = k1$ para o intervalo de posição compreendida entre x_0 e x_1. A partir da posição x_1 o corpo passa a se mover sob aceleração constante $a = k2$. A figura 5.2-p é bastante genérica. Ela mostra um corpo que possui diversos comportamentos mecânicos, que se iniciam e terminam em momentos sinalizados pela ocorrência de eventos que são função do tempo, velocidade e aceleração. Por exemplo, na referida figura a ocorrência do "evento1" marca o início do comportamento mecânico denominado "comportamento 1";

q: descreve um corpo submetido à ação de várias forças F_1, F_2, ..., F_n ; a figura mostra que cada força F_i forma, com o sentido positivo do eixo X, um ângulo de valor ϕ_i . A resultante das forças que agem sobre o corpo pode conferir a ele um estado de repouso ou de movimento;

r: apresenta a inter-relação entre um corpo **A** colocado sobre um corpo **B**. Cada um dos corpos possui uma certa massa m . O corpo **B** sofre a ação de uma força **F** e está sujeito ao atrito do plano sob o qual desliza. O coeficiente de atrito entre o corpo **B** e a superfície é igual a μ.

s: retrata o movimento de rotação de um corpo, a partir do ângulo θ_0, sob uma velocidade angular constante $\omega = k$;

t: descreve o movimento de rotação de um corpo submetido a uma aceleração angular constante $\alpha = k$. O fenômeno se inicia com o corpo possuindo velocidade angular ω_0 na posição angular θ_0;

u: trata-se de uma generalização do fenômeno descrito na figura 5.2-t; a aceleração angular é função do tempo, da velocidade angular e da posição angular ($\alpha = f(t, \omega, \theta)$);

v: esta figura apresenta uma generalização do fenômeno exposto na figura 5.2-s: a velocidade angular é função do tempo e da posição angular ($\omega = f(t, \theta)$);

x: descreve o comportamento mecânico de um corpo pela composição de quatro comportamentos: (1) o movimento com aceleração constante $a = k1$ cuja direção de ação forma um ângulo ϕ com o sentido positivo do eixo X; (2) o movimento sob velocidade constante $v = k2$ cuja direção de ação é a mesma do sentido positivo do eixo X; (3) movimento de rotação sob velocidade inicial $\omega = k3$ e aceleração angular $\alpha = k4$. O fenômeno tem início com o corpo na posição de coordenadas x_0 e y_0 e coordenada angular θ_0;

y: movimento de um corpo submetido a um momento constante M. O fenômeno tem início com o corpo possuindo a aceleração angular ω_0 na posição angular θ_0. O momento de inércia[3] do corpo é igual a I.

O próximo passo após a escolha dos elementos para representar o domínio é a identificação do conjunto de conceitos atômicos que guiará a implementação do ambiente para modelagem. E para isto, precisamos expressar cada um dos elementos da figura 5.2 em função dos conceitos que os compõem. Fizemos isto de três diferentes formas e as apresentamos nas figuras 5.3, 5.4 e 5.5. Chamaremos, a partir de agora, a estas diferentes formas de representação de: primeira forma de representação (figura 5.3), segunda forma de representação (figura 5.4) e terceira forma de representação (figura 5.5). Cada uma das figuras expressa os seus elementos em função de um certo conjunto de conceitos atômicos. Estas formas distintas se diferenciam, basicamente, pelo nível no qual os conceitos de aceleração e velocidade são explicitados. A primeira, segunda e terceira formas apresentam, nesta ordem, níveis decrescentes de granulosidade em relação aos conceitos de velocidade e aceleração. Ou seja, na figura 5.3 os conceitos de velocidade e aceleração estão no estado de maior granulosidade: velocidade e aceleração são conceitos atômicos. Nas figuras 5.4 e 5.5 os conceitos de velocidade e aceleração

[3] Assim como a massa de um corpo oferece resistência para um corpo se deslocar sob a ação de uma força, o momento de inércia de um corpo é uma medida para a resistência que o corpo apresenta para girar em torno de um certo eixo. O momento de inércia é função da geometria e da massa do corpo. Na prática da Engenharia e das soluções dos problemas de Mecânica, os momentos de inércia são tomados a partir de tabelas contendo corpos de geometrias comuns.

precisam ser compostos a partir de seus conceitos subjacentes sendo que, na figura 5.5, estes conceitos se encontram em um estado de menor granulosidade do que na figura 5.4.

Antes de falarmos sobre os conceitos atômicos que utilizamos nas três diferentes formas de representação dos elementos do domínio, vejamos em que as diferenças entre estas formas se fundamentam. Como dissemos na figura 5.3 velocidade e aceleração são apresentadas como conceitos atômicos. Assim por exemplo, o movimento retilíneo de um corpo sob velocidade constante, figura 5.3-a, é descrito, unicamente, pelo conceito atômico de velocidade

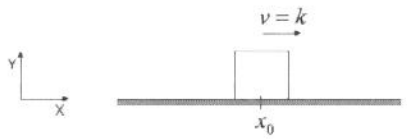
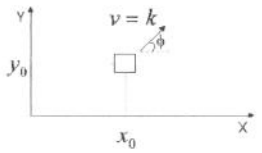
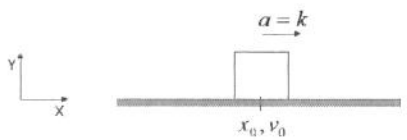
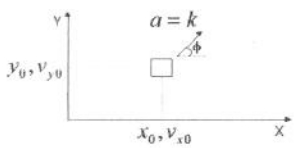
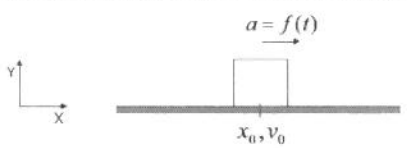
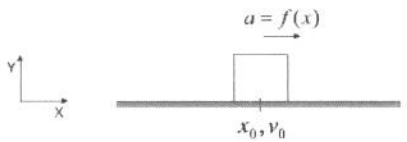
$$v(k,0,x_0,0)$$

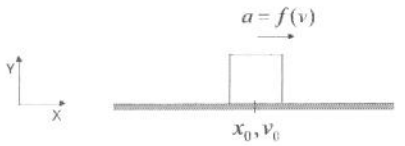
que significa velocidade[4] de valor igual a k e sentido que forma um ângulo de 0° com o sentido positivo do eixo X e que atua sobre um corpo a partir da posição de coordenadas x_0, 0. E o movimento retilíneo de um corpo sob aceleração constante, figura 5.3-c, pode ser descrito simplesmente por

$$a(k,0,x_0,0,v_0,0)$$

que significa aceleração de valor igual a k e sentido que forma um ângulo de 0° com o sentido positivo do eixo X e que atua sobre o corpo a partir da posição de coordenadas x_0, 0. O corpo possui as componentes cartesianas da velocidade inicial nas direções X e Y iguais, respectivamente, a v_0 e 0.

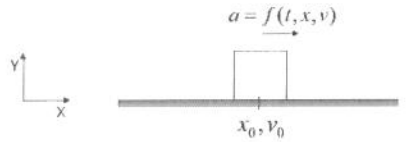
[4] Velocidade e aceleração são grandezas vetoriais, isto é, possuem módulo (ou valor ou intensidade), direção e sentido. A direção pode ser conhecida, por exemplo, pelo ângulo medido no sentido anti-horário, que o vetor forma com o sentido positivo do eixo X. O sentido pode ser conhecido, por exemplo, pelo sinal do seno do ângulo que define a direção. Daí, o conceito atômico de velocidade que estamos usando possui quatro parâmetros que definem, da esquerda para a direita, o seu módulo, através de um valor, a sua direção e sentido através de um ângulo e a posição a partir da qual o vetor de velocidade passa a atuar, através de dois números que representam as coordenadas desta posição. O conceito atômico de aceleração possui, adicionalmente, dois outros parâmetros que definem as componentes cartesianas da velocidade inicial nas direções X e Y.

Movimento retilíneo sob velocidade constante  (a)	$v(k,0,x_0,0)$
 (b)	$v(k,\phi,x_0,y_0)$
Movimento retilíneo sob aceleração constante  (c)	$a(k,0,x_0,0,v_0,0)$
 (d)	$a(k,\phi,x_0,y_0,v_{x0},v_{y0})$
Movimento com aceleração e velocidade genéricas  (e)	$\lvert t \rvert \longrightarrow a(f(t),0,x_0,0,v_0,0)$
 (f)	$\lvert x \rvert \longrightarrow a(f(x),0,x_0,0,v_0,0)$



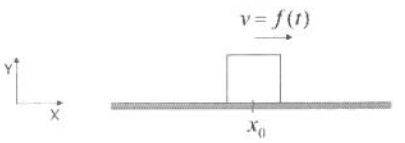
(g)

$$|v| \rightarrow a(f(v),0,x_0,0,v_0,0)$$



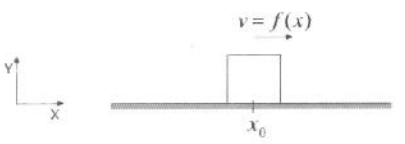
(h)

$$|t|\ |x|\ |v| \rightarrow a(f(t,x,v),0,x_0,0,v_0,0)$$



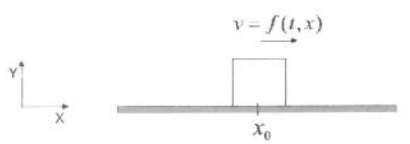
(i)

$$|t| \rightarrow v(f(t),0,x_0,0)$$



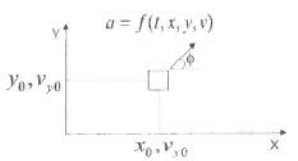
(j)

$$|x| \rightarrow v(f(x),0,x_0,0)$$



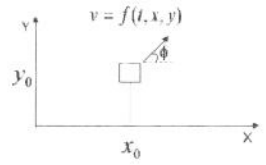
(k)

$$|t|\ |x| \rightarrow v(f(t,x),0,x_0,0)$$



(l)

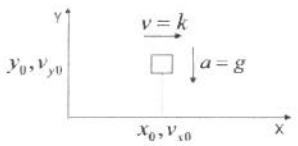
$$|t|\ |x|\ |y|\ |v| \rightarrow a(f(t,x,y,v),\phi,x_0,y_0,v_{x0},v_{y0})$$



(m)

$$|t|\,|x|\,|y|$$
$$v(f(t,x,y),\phi,x_0,y_0)$$

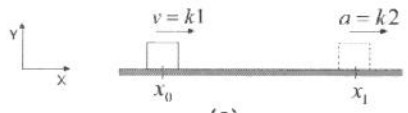
Movimento Composto 1



(n)

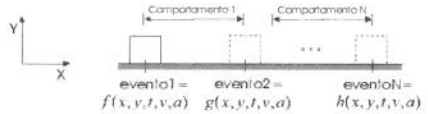
$$a(-g,270°,x_0,y_0,v_{x0},v_{y0})$$

Vários comportamentos em um mesmo fenômeno



(o)

Comportamento 1:
INÍCIO: $x = x_0$ FIM: $x = x_1$
$$v(k1,0,x_0,0)$$
Comportamento 2:
INÍCIO: $x = x_1$ FIM:---
$$a(k2,0,x_1,0,k1,0)$$



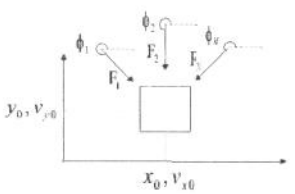
(p)

Comportamento 1:
INÍCIO: $f(x,y,t,v,a) = true$ FIM: $g(x,y,t,v,a) = true$

Comportamento 2:
INÍCIO: $g(x,y,t,v,a) = true$ FIM: $h(x,y,t,v,a) = true$

Comportamento N:
INÍCIO: $h(x,y,t,v,a) = true$ FIM:---

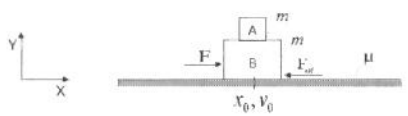
Forças em um corpo



(q)

$$|F_x = F_1.\cos\phi_1 + F_2.\cos\phi_2 + \ldots + F_N.\cos\phi_N| \longrightarrow a(F_x / m,0,x_0,y_0,v_{x0},v_{y0})$$
$$|F_1|,\ |F_2|,\ \ldots,\ |F_N| \quad\quad |m|$$
$$|F_y = F_1.\operatorname{sen}\phi_1 + F_2.\operatorname{sen}\phi_2 + \ldots + F_N.\operatorname{sen}\phi_N| \longrightarrow a(F_y / m,90°,x_0,y_0,v_{x0},v_{y0})$$

Inter-relação entre corpos



(r)

$$|\text{P de A} = m \text{ de A}.g| \quad\quad a(ac \text{ de B},0,x_0,0,v_0,0)$$
$$|F_{at} = \mu.(\text{P de A} + \text{P de B})| \longrightarrow |ac \text{ de B} = (F - F_{at}) / (m \text{ de A} + m \text{ de B})|$$
$$|\text{P de B} = m \text{ de B}.g| \quad\quad |ac \text{ de A} = ac \text{ de B}| \longrightarrow a(ac \text{ de A},0,x_0,0,v_0,0)$$

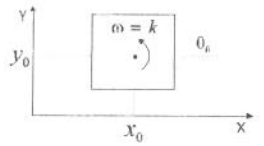
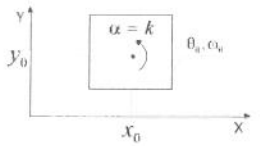
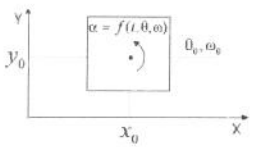
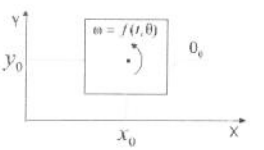
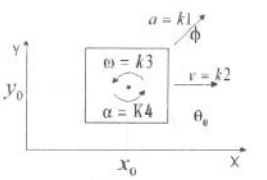
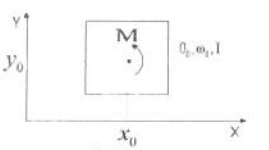
Movimento de rotação sob velocidade angular constante  (S)	$\omega(k, x_0, y_0, \theta_0)$
Movimento de rotaçao sob aceleração angular constante  (T)	$\alpha(k, x_0, y_0, \theta_0, \omega_0)$
Movimento de rotação sob aceleração genérica  (U)	$\|t\| \|\theta\| \|\omega\|$ $\alpha(f(t, \theta, \omega), x_0, y_0, \theta_0, \omega_0)$
Movimento de rotação sob velocidade genérica  (V)	$\|t\| \|\theta\|$ $\omega(f(t, \theta), x_0, y_0, \theta_0)$
Movimento composto 2  (X)	$a(k1, 0, x_0, y_0, k2, 0)$ $\alpha(k4, x_0, y_0, \theta_0, k3)$
Momento em um corpo  (Y)	$\alpha(M / I, x_0, y_0, \theta_0, \omega_0)$

Fig 5.3: Primeira forma de representação dos elementos do domínio da Mecânica

Na figura 5.4 os conceitos de velocidade e aceleração apresentam-se descritos a partir de conceitos mais elementares cujo fundamento se baseia no fato de que a velocidade[5] v é igual a taxa de variação da posição x no decorrer do tempo e que a aceleração a é igual à taxa de variação da velocidade ao longo do tempo. Escrevendo isto segundo uma notação matemática:

$$v = \frac{dx}{dt} \quad \text{e} \quad a = \frac{dv}{dt}. \qquad \text{(Exp. 5.1)}$$

Assim, podemos dizer que

$$\int_{x_0}^{x} dx = \int_{t_0}^{t_1} v \,.\, dt \quad \text{e, também que} \quad \int_{v_0}^{v} dv = \int_{t_0}^{t_1} a \,.\, dt \,. \qquad \text{(Exp. 5.2)}$$

O que implica em

$$x = x_0 + \int_{t_0}^{t_1} v \,.\, dt \quad \text{e} \quad v = v_0 + \int_{t_0}^{t_1} a \,.\, dt \qquad \text{(Exp. 5.3)}$$

Podemos entender as expressões

$$x_0 + \int_{t_0}^{t_1} v \,.\, dt \quad \text{e} \quad v_0 + \int_{t_0}^{t_1} a \,.\, dt \qquad \text{(Exp. 5.4)}$$

como as quantidades acumuladas entre os tempos t_0 e t_1, respectivamente, para x e para v. Usando uma notação alternativa para as expressões acima, podemos escrever que, em um tempo t

$$|v| \rightarrow [x]_{início = x_0} \text{ e que } |a| \rightarrow [v]_{início = v_0} \,. \qquad \text{(Exp. 5.5)}$$

[5] Para ser conceitualmente correto estamos nos referindo à velocidade e aceleração tradicionalmente conhecidas como velocidade intantânea e aceleração instantânea de um corpo, ou seja, a velocidade e a aceleração que o corpo possui em um determinado tempo. A velocidade e a aceleração instantânea de um corpo podem variar à medida que o tempo transcorre e isto é absolutamente diferente dos conceitos de velocidade média e aceleração média de um corpo, os quais se referem, respectivamente, à razão entre o espaço percorrido pelo corpo e o tempo gasto no percurso e a razão entre a diferença da velocidade final e inicial do corpo pelo tempo gasto para levar o corpo, desde a velocidade inicial até a velocidade final.

Devemos ler as expressões 5.5 como: (1) v é uma taxa de variação para a quantidade acumulada x cujo valor inicial é x_0 e (2) a é uma taxa de variação para a quantidade acumulada v cujo valor inicial é v_0. A notação usada nas expressões 5.5 é utilizada na segunda forma de representação dos elementos que selecionamos para representar o domínio de nosso interesse, na figura 5.4. Por exemplo, para o movimento retilíneo de um corpo sob velocidade v, constante, igual a k, observamos na figura 5.4-a

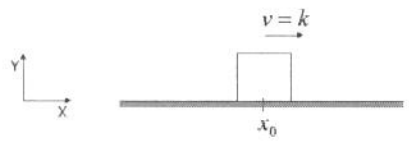
$$|v = k|\rightarrow [x]_{início=x_0} \qquad \text{(Exp. 5.6)}$$

que significa: a velocidade v é constante, tem o valor k e atua como uma taxa de variação para a posição que está representada pela quantidade acumulada x, cujo valor inicial é x_0. Para o movimento retilíneo de um corpo sob aceleração, constante, igual a k (figura 5.4-c), observamos

$$|a = k|\rightarrow [v]_{início=v_0} \rightarrow [x]_{início=x_0} \qquad (5.7)$$

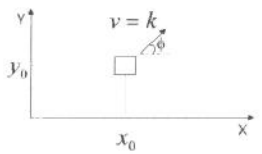
que significa: a aceleração a é uma constante de valor igual a k e atua como uma taxa de variação para a velocidade, representada pela quantidade acumulada v. A velocidade v, por sua vez, é uma taxa de variação para a posição, que está representada pela quantidade acumulada x.

Movimento retilíneo sob velocidade constante



(a)

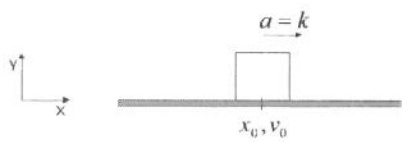
$$|v = k| \longrightarrow [x]_{início=x_0}$$



(b)

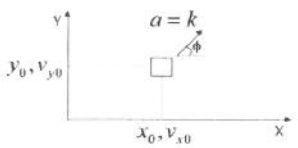
$$|v = k| \begin{cases} |v_x = v.\cos\phi| \longrightarrow [x]_{início=x_0} \\ |v_y = v.\operatorname{sen}\phi| \longrightarrow [y]_{início=y_0} \end{cases}$$

Movimento retilíneo sob aceleração constante



(c)

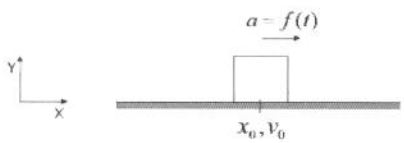
$$|a = k| \longrightarrow [v]_{início=v_0} \longrightarrow [x]_{início=x_0}$$



(d)

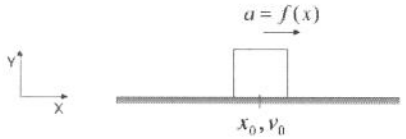
$$|a = k| \begin{cases} |a_x = a.\cos\phi| \longrightarrow [v_x]_{início=v_{x0}} \longrightarrow [x]_{início=x_0} \\ |a_y = a.\operatorname{sen}\phi| \longrightarrow [v_y]_{início=v_{y0}} \longrightarrow [y]_{início=y_0} \end{cases}$$

Movimento com aceleração e velocidade genéricas



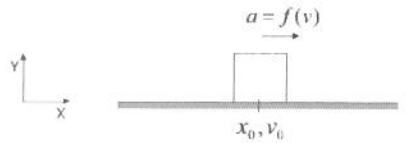
(e)

$$|t| \begin{cases} \\ |a = f(t)| \longrightarrow [v]_{início=v_0} \longrightarrow [x]_{início=x_0} \end{cases}$$

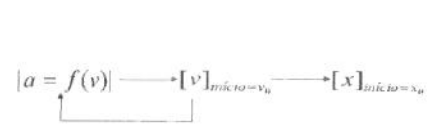


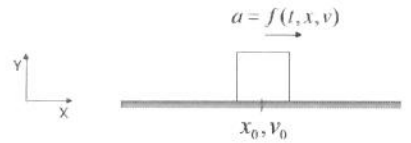
(f)

$$|a = f(x)| \longrightarrow [v]_{início=v_0} \longrightarrow [x]_{início=x_0}$$

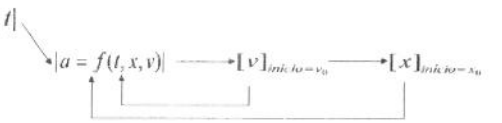

a = f(v)
Y
X
x0, v0

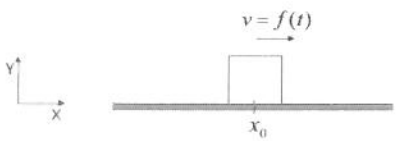
(g)


|a = f(v)|
[v]início=v0
[x]início=x0

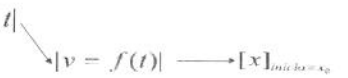

a = f(t, x, v)
Y
X
x0, v0

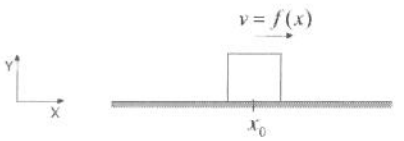
(h)


|t|
|a = f(t, x, v)|
[v]início=v0
[x]início=x0


v = f(t)
Y
X
x0

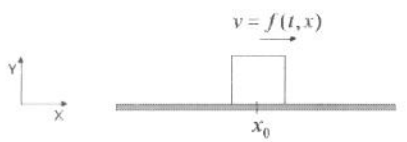
(i)


|t|
|v = f(t)|
[x]início=x0

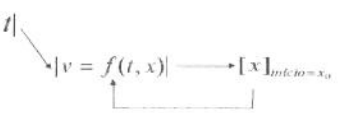

v = f(x)
Y
X
x0

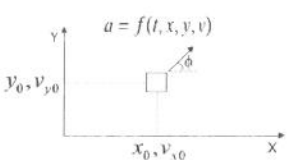
(j)

|v = f(x)|
[x]início=x0

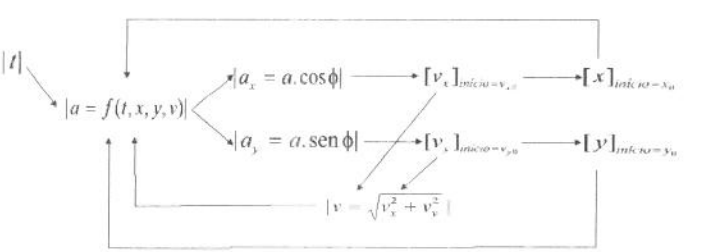

v = f(t, x)
Y
X
x0

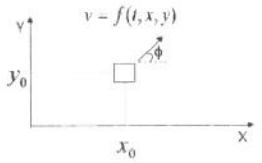
(k)


|t|
|v = f(t, x)|
[x]início=x0


a = f(t, x, y, v)
Y
y0, vy0
x0, vx0
X

(l)


|t|
|a = f(t, x, y, v)|
|ax = a.cosϕ|
[vx]início=vx0
[x]início=x0
|ay = a.senϕ|
[vy]início=vy0
[y]início=y0
|v = √(vx² + vy²)|

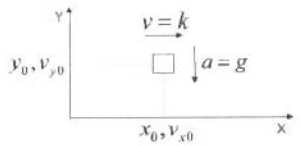


(m)

$|t| \rightarrow |v = f(t, x, y)| \rightarrow |v_x = v.\cos\phi| \longrightarrow [x]_{início = x_0}$

$|v = f(t, x, y)| \rightarrow |v_y = v.\text{sen}\,\phi| \longrightarrow [y]_{início = y_0}$

Movimento Composto 1

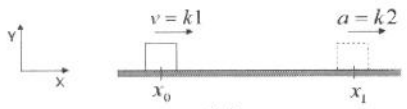


(n)

$|v = k| \longrightarrow [x]_{início = x_0}$

$|a = -g| \longrightarrow [v_y]_{início = v_{y0}} \longrightarrow [y]_{início = y_0}$

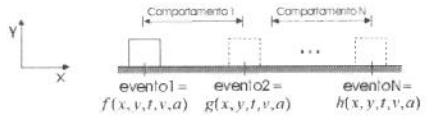
Vários comportamentos em um mesmo fenômeno



(o)

Comportamento 1:
INÍCIO: $x = x_0$ FIM: $x = x_1$
$|v = k1| \longrightarrow [x]_{início = x_0}$

Comportamento 2:
INÍCIO: $x = x_1$ FIM:---
$|a = k2| \longrightarrow [v]_{início = k1} \longrightarrow [x]_{início = x_1}$



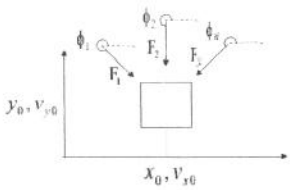
(p)

Comportamento 1:
INÍCIO: $f(x, y, t, v, a) = true$ FIM: $g(x, y, t, v, a) = true$

Comportamento 2:
INÍCIO: $g(x, y, t, v, a) = true$ FIM: $h(x, y, t, v, a) = true$

Comportamento N:
INÍCIO: $h(x, y, t, v, a) = true$ FIM:---

Forças em um corpo



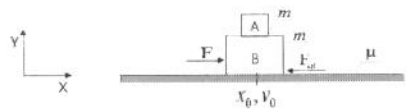
(q)

$|F_x = F_1.\cos\phi_1 + F_2.\cos\phi_2 + \ldots + F_N.\cos\phi_N| \longrightarrow |a_x = F_x / m| \longrightarrow [v_x]_{início = v_{x0}} \rightarrow [x]_{início = x_0}$

$|F_y = F_1.\text{sen}\,\phi_1 + F_2.\text{sen}\,\phi_2 + \ldots + F_N.\text{sen}\,\phi_N| \longrightarrow |a_y = F_y / m| \longrightarrow [v_y]_{início = v_{y0}} \rightarrow [y]_{início = y_0}$

$|F_1|, |F_2|, \ldots, |F_N|, |m|$

Inter-relação entre corpos



(r)

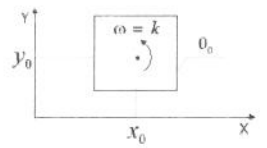
$|\text{P de A} = m \text{ de A} . g|$

$|F_{at} = \mu . (\text{P de A} + \text{P de B})| \longrightarrow |a \text{ de B} = (F - F_{at}) / (m \text{ de A} + m \text{ de B})| \longrightarrow [v \text{ de B}]_{início = v_0} \rightarrow [x \text{ de B}]_{início = x_0}$

$|\text{P de B} = m \text{ de B} . g|$

$|a \text{ de A} = a \text{ de B})| \longrightarrow [v \text{ de A}]_{início = v_0} \longrightarrow [x \text{ de A}]_{início = x_0}$

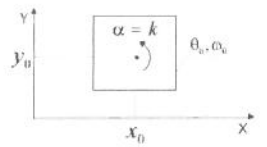
Movimento de rotação sob velocidade angular constante



(s)

$|\omega = k| \longrightarrow [\theta]_{início=0_0}$

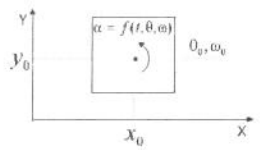
Movimento de rotação sob aceleração angular constante



(t)

$|\alpha = k| \longrightarrow [\omega]_{início=\omega_0} \longrightarrow [\theta]_{início=\theta_0}$

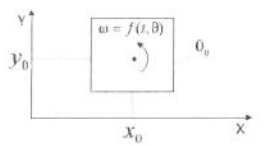
Movimento de rotação sob aceleração genérica



(u)

$|t|$
$|\alpha = f(t,\theta,\omega)| \longrightarrow [\omega]_{início=\omega_0} \longrightarrow [\theta]_{início=\theta_0}$

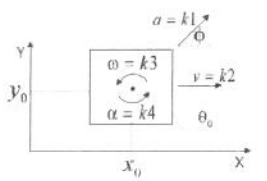
Movimento de rotação sob velocidade genérica



(v)

$|t|$
$|\omega = f(t,\theta)| \longrightarrow [\theta]_{início=\theta_0}$

Movimento composto 2

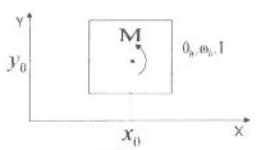


(x)

$|a = k1| \begin{cases} |a_x = a.\cos\phi| \longrightarrow [v_x]_{início=k2} \longrightarrow [x]_{início=x_0} \\ |a_y = a.\operatorname{sen}\phi| \longrightarrow [v_y]_{início=0} \longrightarrow [y]_{início=y_0} \end{cases}$

$|\alpha = k4| \longrightarrow [\omega]_{início=\omega_0} \longrightarrow [\theta]_{início=\theta_0}$

Momento em um corpo



(y)

$|\alpha = M / I| \longrightarrow [\omega]_{início=\omega_0} \longrightarrow [\theta]_{início=0_0}$

Fig. 5.4: Segunda forma de representação dos elementos do domínio da Mecânica

Na figura 5.3 os conceitos de velocidade e aceleração são descritos a partir de um único conceito atômico. Na figura 5.4, velocidade e aceleração são descritos a partir dos conceitos de taxa de variação e quantidade acumulada no tempo e que, na notação matemática, podem ser traduzidos pelas expressões 5.3 que fazem uso do conceito de integral. Pode-se dizer que "quebramos" os conceitos de aceleração e velocidade para encontrar os conceitos mais elementares de taxa de variação e quantidade acumulada no tempo. De forma análoga "quebraremos" os conceitos presentes nas equações 5.3 com o objetivo de obter conceitos mais elementares. Uma aproximação[6] para as equações 5.3 é dada por

$$x = x_0 + \sum_{t_0,dt}^{t_1} v\,.\,dt \quad \text{e} \quad v = v_0 + \sum_{t_0,dt}^{t_1} a\,.\,dt \;. \qquad \text{(Exp 5.8)}$$

Esta aproximação é conhecida como método de Euler[7] para resolução de equações diferenciais (Bronson, R., 1977). Devemos entender os sinais de somatório acima como: somatório do tempo variando de t_0 até t_1 em intervalos de valor igual a *dt*. As expressões 5.8 podem ser traduzidas na forma de algoritmo como abaixo:

$x \leftarrow x_0$;
repita para *tempo* $= t_0$ até t_1 passo *dt*
  $x \leftarrow x + v\,.\,dt$
fim repita

e

$v \leftarrow v_0$;
repita para *tempo* $= t_0$ até t_1 passo *dt*
  $v \leftarrow v + a\,.\,dt$
fim repita

É esta notação algorítmica que estamos utilizando para descrever os elementos da figura 5.5. Por exemplo, temos para os movimentos retilíneos sob velocidade e aceleração constante (figuras 5.5-a e 5.5-c) respectivamente:

$x \leftarrow x_0$;
repita para *tempo* $= t_0$ até t_1 passo *dt*
  $x \leftarrow x + k\,.\,dt$
fim repita

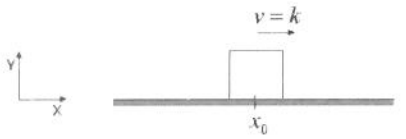
e

$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para *tempo* $= t_0$ até t_1 passo *dt*
  $v \leftarrow v + k\,.\,dt$;
  $x \leftarrow x + v\,.\,dt$
fim repita

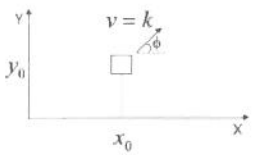
[6] Estamos falando de aproximação porque as equações 5.3 nem sempre possuem uma solução algébrica. No enteanto, existem diversas soluções numéricas aproximadas.

[7] O método de Euler é um método bastante simples para resolução de equações diferenciais. Existem outros métodos mais precisos: método de Runge-Kutta, método de Heun, método de Milne, método de Hamming etc.. Para o nosso objetivo, ou seja, verificar os conceitos mais elementares que o de integral, qualquer destes métodos conduziria ao mesmo resultado. No entanto preferimos, por simplicidade, utilizar o método de Euler em nossa discussão.

Movimento retilíneo sob velocidade constante



(a)

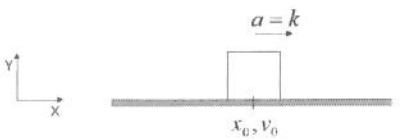


(b)

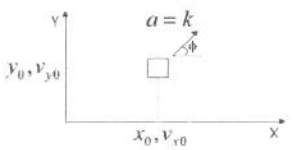
$x \leftarrow x_0$;
repita para $t = t_0$ até t passo dt
  $x \leftarrow x + k.dt$
fim repita

$x \leftarrow x_0$; $y \leftarrow y_0$;
repita para $t = t_0$ até t passo dt
  $x \leftarrow x + k.\cos\phi.dt$;
  $y \leftarrow y + k.\text{sen}\,\phi.dt$
fim repita

Movimento retilíneo sob aceleração constante



(c)

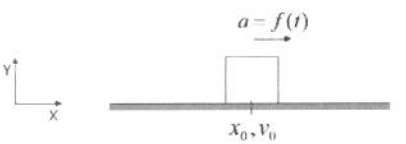


(d)

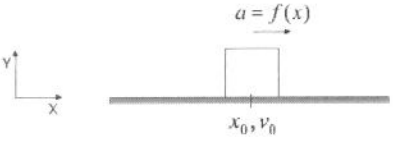
$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow v + k.dt$;
  $x \leftarrow x + v.dt$
fim repita

$x \leftarrow x_0$; $y \leftarrow y_0$; $v_x \leftarrow v_{x0}$; $v_y \leftarrow v_{y0}$;
repita para $t = t_0$ até t passo dt
  $v_x \leftarrow v_x + k.\cos\phi.dt$; $v_y \leftarrow v_y + k.\text{sen}\,\phi.dt$;
  $x \leftarrow x + v_x.dt$; $y \leftarrow y + v_y.dt$
fim repita

Movimento com aceleração e velocidade genéricas



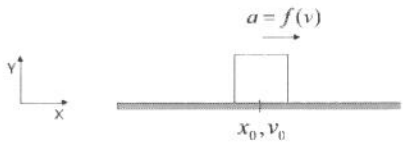
(e)



(f)

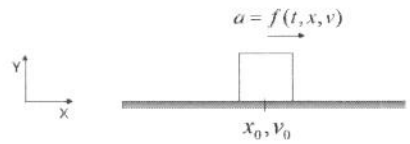
$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  $a \leftarrow f(t)$;
  $v \leftarrow v + a.dt$;
  $x \leftarrow x + v.dt$
fim repita

$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  $a \leftarrow f(x)$;
  $v \leftarrow v + a.dt$;
  $x \leftarrow x + v.dt$
fim repita



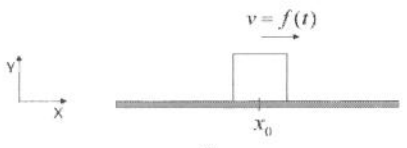
(g)

$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  $a \leftarrow f(v)$;
  $v \leftarrow v + a.dt$;
  $x \leftarrow x + v.dt$
fim repita



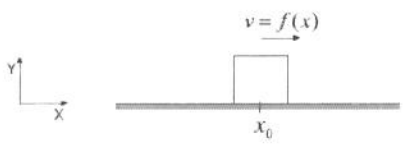
(h)

$x \leftarrow x_0$; $v \leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  $a \leftarrow f(t, x, v)$;
  $v \leftarrow v + a.dt$;
  $x \leftarrow x + v.dt$
fim repita



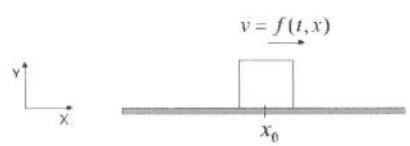
(i)

$x \leftarrow x_0$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow f(t)$;
  $x \leftarrow x + v.dt$
fim repita



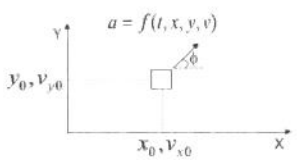
(j)

$x \leftarrow x_0$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow f(x)$;
  $x \leftarrow x + v.dt$
fim repita



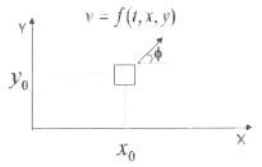
(k)

$x \leftarrow x_0$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow f(t, x)$;
  $x \leftarrow x + v.dt$
fim repita



(l)

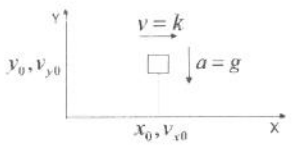
$x \leftarrow x_0$; $y \leftarrow y_0$; $v_x \leftarrow v_{x0}$; $v_y \leftarrow v_{y0}$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow \sqrt{v_x^2 + v_y^2}$; $a \leftarrow f(t, x, y, v)$;
  $a_x \leftarrow a.\cos\phi$; $a_y \leftarrow a.\operatorname{sen}\phi$;
  $v_x \leftarrow v_x + a_x.dt$; $v_y \leftarrow v_y + a_y.dt$;
  $x \leftarrow x + v_x.dt$; $y \leftarrow y + v_y.dt$
fim repita



(m)

$x \leftarrow x_0$; $y \leftarrow y_0$;
repita para $t = t_0$ até t passo dt
  $v \leftarrow f(t, x, y)$;
  $v_x \leftarrow v.\cos\phi$; $v_y \leftarrow v.\text{sen}\,\phi$;
  $x \leftarrow x + v_x.dt$; y $\leftarrow y + v_y.dt$
fim repita

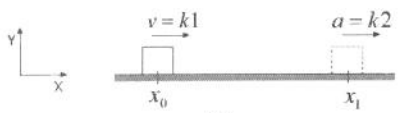
Movimento Composto 1



(n)

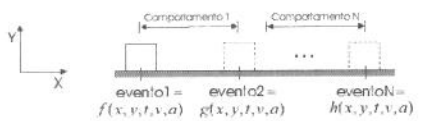
$x \leftarrow x_0$; $y \leftarrow y_0$; $v_x \leftarrow k$; $v_y \leftarrow v_{y0}$;
repita para $t = t_0$ até t passo dt
  $v_y \leftarrow -g.dt$;
  $x \leftarrow x + v_x.dt$; y $\leftarrow y + v_y.dt$
fim repita

Vários comportamentos em um mesmo fenômeno



(o)

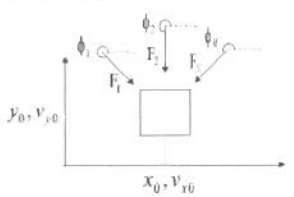
$x \leftarrow x_0$; v $\leftarrow v_0$;
repita para $t = t_0$ até t passo dt
  se ($x_0 \le x \le x_1$)
    $x \leftarrow x + k1.dt$
  senão
    $v \leftarrow k2.dt$;
    $x \leftarrow x + v.dt$
  fim se
fim repita



(p)

inicioiuC1 $\leftarrow$ *false*; *terminouC1* $\leftarrow$ *true*; ...; *inicioiuCN* $\leftarrow$ *false*; *terminouCN* $\leftarrow$ *true*;
repita para $t = t_0$ até t passo dt
  se (não *inicioiuC1*)
    inicioiuC1 $\leftarrow f(x, y, t, v, a)$ = *true*; se (*inicioiuC1*) *terminouC1* $\leftarrow$ *false* fim se
  fim se;
  se (*inicioiuC1* e não *terminouC1*) *terminouC1* $\leftarrow g(x, y, t, v, a)$ = *true*;
    se (não *terminouC1*) ... comportamento 1 ... senão *inicioiuC1* $\leftarrow$ *false* fimse
  fim se
  ... tratamento dos outros comportamentos ...
fim repita

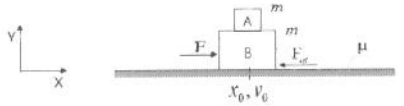
Forças em um corpo



(q)

$x \leftarrow x_0$; $y \leftarrow y_0$; $v_x \leftarrow v_{x0}$; $v_y \leftarrow v_{y0}$;
repita para $t = t_0$ até t passo dt
  $F_x \leftarrow F_1.\cos\phi_1 + F_2.\cos\phi_2 + \ldots + F_N.\cos\phi_N$;
  $F_y \leftarrow F_1.\text{sen}\,\phi_1 + F_2.\text{sen}\,\phi_2 + \ldots + F_N.\text{sen}\,\phi_N$;
  $a_x \leftarrow F_x / m$; $a_y \leftarrow F_y / m$;
  $v_x \leftarrow a_x.dt$; $v_y \leftarrow a_y.dt$;
  $x \leftarrow x + v_x.dt$; y $\leftarrow y + v_y.dt$
fim repita

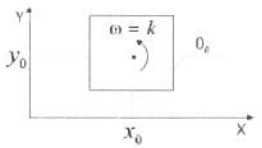
Inter-relação entre corpos



(r)

$x \leftarrow x_0$; v de A $\leftarrow v_0$; v de B $\leftarrow v_0$;
P de A $\leftarrow m$ de A $.g$; P de B $\leftarrow m$ de B $.g$;
$F_{at} \leftarrow \mu$. (P de A + P de B);
a de B $\leftarrow$ (F $- F_{at}$) / (m de A + m de B);
a de A $\leftarrow a$ de B;
repita para $t = t_0$ até t passo dt
  v de B $\leftarrow a$ de B.dt; v de A $\leftarrow a$ de A.dt;
  x de B $\leftarrow x$ de B + v de B.dt; x de A $\leftarrow x$ de A + v de A.dt;
fim repita

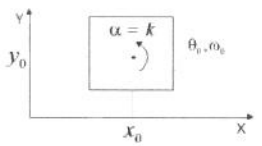
Movimento de rotação sob velocidade angular constante



(S)

$\theta \leftarrow \theta_0$;
repita para $t = t_0$ até t passo dt
  $\theta \leftarrow \theta + k.dt$
fim repita

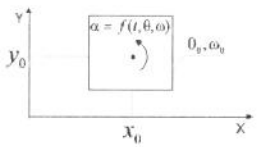
Movimento de rotação sob aceleração angular constante



(T)

$\theta \leftarrow \theta_0$; $\omega \leftarrow \omega_0$;
repita para $t = t_0$ até t passo dt
  $\omega \leftarrow \omega + k.dt$;
  $\theta \leftarrow \theta + \omega.dt$
fim repita

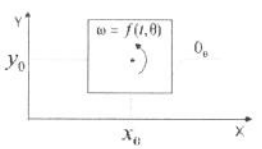
Movimento de rotação sob aceleração genérica



(U)

$\theta \leftarrow \theta_0$; $\omega \leftarrow \omega_0$;
repita para $t = t_0$ até t passo dt
  $\alpha \leftarrow f(t, \theta, \omega)$;
  $\omega \leftarrow \omega + \alpha.dt$;
  $\theta \leftarrow \theta + \omega.dt$
fim repita

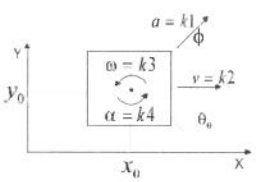
Movimento de rotação sob velocidade genérica



(V)

$\theta \leftarrow \theta_0$;
repita para $t = t_0$ até t passo dt
  $\omega \leftarrow f(t, 0)$;
  $\theta \leftarrow \theta + \omega.dt$
fim repita

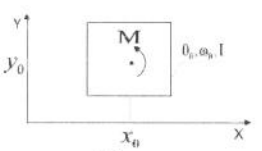
Movimento composto 2



(X)

$x \leftarrow x_0$; $y \leftarrow y_0$; $\theta \leftarrow \theta_0$;
$v_x \leftarrow k2$; $v_y \leftarrow 0$; $\omega \leftarrow k3$;
repita para $t = t_0$ até t passo dt
  $a_x \leftarrow k1.\cos\phi$; $a_y \leftarrow k1.\text{sen}\,\phi$;
  $v_x \leftarrow v_x + a_x.dt$; $v_y \leftarrow v_y + a_y.dt$;
  $\omega \leftarrow \omega + k4.dt$;
  $\theta \leftarrow \theta + \omega.dt$
fim repita

Momento em um corpo



(Y)

$\theta \leftarrow \theta_0$; $\omega \leftarrow \omega_0$; $\alpha \leftarrow M / I$;
repita para $t = t_0$ até t passo dt
  $\omega \leftarrow \omega + \alpha.dt$;
  $\theta \leftarrow \theta + \omega.dt$
fim repita

Fig. 5.5: Terceira forma de representação dos elementos do domínio da Mecânica

Podemos, agora, extrair os conjuntos de conceitos atômicos que aparecem nas três formas de representação apresentadas. A figura 5.6 ilustra o conjunto de conceitos atômicos que extraímos da primeira forma de representação.

- referencial newtoniano
- velocidade do corpo
- aceleração do corpo
- velocidade angular do corpo
- aceleração angular do corpo
- tempo
- posição do corpo
- posição angular do corpo
- valor da velocidade
- valor da aceleração
- valor da velocidade angular
- valor da aceleração angular
- função algébrica
- conexão
- duração do comportamento
- relação de dependência entre corpos
- nome do corpo
- composição de comportamentos

Fig. 5.6: Conceitos atômicos extraídos da primeira forma de representação dos elementos do domínio da Mecânica

Uma vez que a Mecânica trata das condições de repouso ou movimento dos corpos, todos os fenômenos mecânicos são expressos pela variação ou não das posições dos corpos envolvidos no fenômeno. E a posição de um corpo pode ser medida a partir de um referencial fixo em relação ao corpo. Daí, não é surpresa constatar que o conceito atômico que estamos denominando de referencial newtoniano, simbolizado por

apareça em todos os fenômenos. Afinal, todas as posições são medidas a partir deste referencial[8] que possui um par de eixos X e Y.

[8] O referencial newtoniano não precisa necessariamente ser um referencial cartesiano composto, no plano, pelo par de eixos X e Y. Ele poderia ser, por exemplo, um referencial polar composto de um ponto, denominado polo e por uma reta de referência para medida de ângulos, a partir do qual as posições são medidas. Em referencial polar a posição de um ponto **P** é dada por dois valores: a distância de **P** ao polo e o ângulo que a reta que passa por **P** e pelo polo forma com a reta de referência para medida de ângulos. Aqui, estamos considerando o referencial newtoniano como um referencial cartesiano. No entanto, o resultado, isto é, os conceitos atômicos que estamos extraindo, não mudariam caso estivessemos considerando outro tipo de referencial.

O conceito atômico de velocidade, simbolizado por $v(k, \phi, x_0, y_0)$, aparece em grande parte das descrições dos fenômenos e é revelado já no início pelas figuras 5.3-a e 5.3-b. Este conceito atômico possui 4 parâmetros que definem, da esquerda para a direita, para o vetor velocidade o seu módulo, através de um valor, a sua direção e sentido através de um ângulo e a posição a partir da qual o vetor de velocidade passa a atuar, através de dois números que representam as coordenadas desta posição. O conceito atômico de aceleração, simbolizado por $a(k, \phi, x_0, y_0, v_{x0}, v_{y0})$, apresenta os mesmos quatro primeiros parâmetros do conceito atômico de velocidade. Adicionalmente, existem dois outros parâmetros que definem as componentes cartesianas da velocidade inicial nas direções X e Y. Dizer que um corpo possui um comportamento mecânico representado por $v(k, \phi, x_0, y_0)$ é o mesmo que afirmar que ele possui uma velocidade constante de valor igual a k orientada segundo uma direção que forma um ângulo ϕ no sentido anti-horário com o lado positivo do eixo X e que este comportamento tem início no ponto de coordenadas x_0, y_0. Analogamente, um corpo que possui um comportamento mecânico representado por $a(k, \phi, x_0, y_0, v_{x0}, v_{y0})$ parte de uma posição de coordenadas x_0, y_0, possuindo as componentes cartesianas da velocidade iguais a v_{x0}, v_{y0} e está dotado de aceleração de valor igual a k orientada segundo a direção que forma um ângulo ϕ com o sentido anti-horário com o lado positivo do eixo X.

As figuras de 5.3-r a 5.3-y evidenciam os conceitos atômicos de velocidade angular, $\omega(k, x_0, y_0, \theta_0)$, e aceleração angular, $\alpha(k, x_0, y_0, \theta_0, \omega_0)$. A velocidade angular e aceleração angular representam para o movimento de rotação do corpo, o que representam a velocidade e aceleração para o movimento de translação. Um corpo com comportamento mecânico dado por $\omega(k, x_0, y_0, \theta_0)$ está em movimento de rotação ao redor do eixo de coordenadas x_0, y_0 sob velocidade angular de valor igual a k; o movimento tem início com o corpo possuindo posição angular de valor θ_0. De forma similar um corpo com comportamento mecânico dado por $\alpha(k, x_0, y_0, \theta_0, \omega_0)$ está em movimento de rotação ao redor do eixo de coordenadas x_0, y_0 sob aceleração angular de valor igual a k; o movimento tem início com o corpo possuindo posição angular de valor θ_0 e velocidade angular ω_0.

Os conceitos atômicos de tempo (t), posição do corpo (x, y), posição angular do corpo (θ), valor da velocidade (v), valor da aceleração (a), valor da velocidade angular (ω) e valor da aceleração angular (α) são usados nas representações de vários fenômenos com objetivo de compor uma expressão que é função das grandezas que representam. Por exemplo a figura 5.3-l descreve o comportamento mecânico de um corpo cuja aceleração

é função das grandezas tempo, posição e velocidade do corpo. Assim os conceitos atômicos tempo (t), posição do corpo (x, y) e velocidade (v) são utilizados para compor uma expressão que define o valor da aceleração como função dos mesmos.

Os conceitos atômicos de função algébrica e de conexão surgem da necessidade de se efetuar o cálculo de uma expressão e encaminhar o valor do resultado. Por exemplo, na descrição do fenômeno mostrado na figura 5.3-q, calcula-se o valor da componente cartesiana, na direção X, da resultante das forças que agem sobre o corpo por $F_x = F_1.\cos\phi_1 + F_2.\cos\phi_2 + \ldots + F_N.\cos\phi_N$. O valor do resultado, ou seja F_x, é encaminhado para o conceito atômico de aceleração para que o valor F_x / m possa ser calculado. Simbolizamos o conceito atômico que estamos chamando de função algébrica por meio de duas barras verticais[9] entre as quais escrevemos uma função algébrica. Assim na referida figura a função algébrica que calcula o valor de F_x está escrita como

$$|F_x = F_1.\cos\phi_1 + F_2.\cos\phi_2 + \ldots + F_N.\cos\phi_N|.$$

Já o conceito atômico de conexão, está simbolizado por uma flecha que conduz o valor de uma função algébrica a outro conceito atômico.

As figuras 5.3-o e 5.3-p mostram fenômenos cujo comportamento mecânico pode ser entendido como sendo composto por vários comportamentos individuais e de tal modo que, a passagem de um comportamento individual para outro, é marcada por um evento que é função das diversas grandezas envolvidas no fenômeno. Surge daí, a necessidade de um conceito atômico que descreva as condições de início de fim de cada comportamento mecânico individual: a este conceito atômico estamos dando o nome de duração do comportamento.

Alguns fenômenos envolvem vários corpos e a interação entre eles, tal como ocorre com fenômeno mostrado na figura 5.3-r, na qual um corpo A está sobre um corpo B que sofre a ação de uma força F e pode deslizar, com atrito, sobre uma superfície plana rugosa. O peso do corpo A interfere no movimento do corpo B e, por sua vez, o movimento do corpo B impõe movimento ao corpo A. Ou seja, existe uma inter-relação entre os comportamentos mecânicos dos corpos envolvidos no fenômeno. Isto, também, é o que demostra a descrição do comportamento mecânico apresentada na referida figura. Surge

[9] Salientamos que não estamos usando as duas barras verticais como na Matemática, onde elas significam valor absoluto.

um novo elemento capaz de se referir, com distinção, às propriedades de um e outro corpo como, por exemplo: (1) P de A e P de B que se referem, respectivamente aos pesos (P) do corpo A e do corpo B; (2) *ac* de A e *ac* de B que se referem à intensidade das acelerações (*ac*) do corpo A e B, respectivamente. A este conceito atômico, capaz de se referir, com distinção, a propriedades de um e outro corpo denominamos relação de dependência entre corpos. Surge, por conseqüência deste conceito atômico, outro conceito atômico: o de nome do corpo.

Às vezes o comportamento mecânico global de um corpo pode ser entendido como sendo composto pela combinação de diversos comportamentos mais simples. É o que ocorre, por exemplo, com o fenômeno ilustrado na figura 5.3-x: o corpo ilustrado sofre um movimento de rotação enquanto translada-se sob aceleração e velocidade cujas direções estão ilustradas na figura. O movimento de rotação pode ser descrito por $\alpha(k4, x_0, y_0, \theta_0, k3)$ e o movimento de translação pode ser descrito por $a(k1, 0, x_0, y_0, k2, 0)$ e entendemos que o movimento final do corpo é a composição dos movimentos individualmente descritos. A este conceito atômico que compõe o movimento global do corpo a partir das descrições de comportamentos mais simples e ao qual não associamos nenhum símbolo especial, chamamos de composição de comportamentos.

A segunda forma de representação dos elementos do domínio (figura 5.4) apresenta o conjunto de conceitos atômicos ilustrado na figura 5.7

- referencial newtoniano
- tempo
- posição do corpo
- posição angular do corpo
- função algébrica
- conexão
- duração do comportamento
- relação de dependência entre corpos
- nome do corpo
- composição de comportamentos
- quantidade acumulada

Fig. 5.7: Conceitos atômicos extraídos da segunda forma de representação dos elementos do domínio da Mecânica

Aparecem, novamente, os seguintes conceitos atômicos que já comentamos: referencial newtoniano, tempo, posição do corpo, posição angular do corpo, duração do comportamento, relação de dependência entre corpos, nome do corpo, função algébrica, conexão e composição de comportamentos. Simbolizamos por $[x]_{início=x_0}$ o conceito atômico quantidade acumulada; no caso, estamos representando a quantidade acumulada de nome x cujo valor valor inicial é x_0. Este conceito atômico está associado às variações, no decorrer do tempo, que x pode sofrer quando sobre ele incide uma taxa de variação. Por exemplo, quando uma função algébrica, digamos $|v = 5|$ atua como uma taxa de variação para uma quantidade x, cujo valor inicial é 3, escrevemos

$$|v = 5| \rightarrow [x]_{início=3}$$

e entendemos que, a cada unidade de tempo, o valor de x sofre acréscimos de valor igual a v, ou seja 5. Assim, x assumirá, sucessivamente, a cada unidade de tempo, os valores 3, 8, 13, 18, ...

A terceira forma de representação dos elementos do domínio revela, também, alguns conceitos atômicos já comentados anteriormente: referencial newtoniano, posição do corpo, posição angular do corpo e relação de dependência entre corpos e nome do corpo.

• referencial newtoniano	• expressão
• posição do corpo	• comando de repetição
• posição angular do corpo	• comando de atribuição
• relação de dependência entre corpos	• comando condicional
• nome do corpo	• variável

Fig. 5.8: Conceitos atômicos extraídos da terceira forma de representação dos elementos do domínio da Mecânica

Os conceitos atômicos de variável, comando de atribuição, comando condicional e expressão surgiram da necessidade de se implementar a aproximação para o conceito de integral dadas pelas expressões 5.8. Já o conceito atômico de comando condicional surgiu da necessidade de se descrever o início e o fim de cada comportamento mecânico individual que integra os fenômenos ilustrados nas figuras 5.5-o e 5.5-p.

Possuímos, agora, três conjuntos de conceitos atômicos que são capazes de descrever os fenômenos do domínio de nosso interesse. Nossa próxima tarefa é escolher qual, dos três conjuntos é o mais adequado para o estabelecimento do ambiente para modelagem. De todos os três conjuntos de conceitos atômicos estabelecidos, o que apresenta conceitos com maior granulosidade é o primeiro (figura 5.6). Em seqüência, a granulosidade diminui no segundo e terceiro conjuntos de conceitos atômicos (figuras 5.7 e 5.8, respectivamente). O terceiro conjunto de conceitos atômicos apresenta a menor quantidade de conceitos atômicos dentre os três que foram estabelecidos. O segundo e o primeiro conjuntos de conceitos atômicos apresentam, sucessivamente, quantidades crescentes de conceitos atômicos. Aplicando o princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos, enunciado na seção 4.3 do capítulo 4, concluímos que, dos três conjuntos de conceitos atômicos que estabelecemos, o mais adequado para o estabelecimento do ambiente para modelagem é o segundo conjunto (figura 5.7).

Podemos entender, de outra forma, a escolha a que conduziu a aplicação "nua e crua" do princípio do equilíbrio entre o modelo conceitual mínimo e a máxima granulosidade dos conceitos atômicos. O primeiro conjunto de conceitos atômicos apresenta conceitos mais próximos de um ambiente para simulação do que os outros dois conjuntos. Isto é verdade, dado que quase nenhum esforço é requerido do usuário para a descrição, por exemplo, de um movimento retilíneo de um corpo sob velocidade constante: um único conceito atômico é suficiente para descrever o movimento (figura 5.3-a e 5.3-b); este único conceito atômico encerra em si todo conhecimento sobre o movimento retilíneo de um corpo sob velocidade constante e deixa ao usuário, apenas, o trabalho de variar os parâmetros que constam no conceito atômico. Por outro lado, o terceiro conjunto de conceitos atômicos apresenta conceitos típicos de uma linguagem de programação de propósito geral que, como vimos no capítulo 2, ao mesmo tempo em que oferece ao usuário um alto nível de controle, solicita dele um esforço cognitivo maior nas descrições dos fenômenos do domínio. Os ambientes para modelagem orientados a um domínio (AMOD) o quais buscamos o *design* apresentam a virtude de representar um equilíbrio entre estes dois pólos. E, dos três conjuntos de conceitos atômicos estabelecidos, o que mais se aproxima disto é, justamente, o segundo conjunto.

5.4 - DESENVOLVIMENTO DO *DESIGN* DE CADA CONCEITO ATÔMICO

Hoje em dia conhecemos diversos tipos de ambientes que permitem a modelagem de fenômenos da mecânica. Examinemos dois deles. Consideremos, em primeiro lugar, um ambiente simples que permite a expressão de desenhos dos corpos em papel e a associação de textos, em português, que descrevem as condições de movimento ou de repouso de cada corpo. Embora este ambiente permita ao usuário pensar e descrever fenômenos da Mecânica, a falta de poder previsão torna-o de pouca importância para o estudioso de Mecânica. Não se pode, por exemplo, prever com precisão em quanto tempo um meteoro atingirá um dado planeta, dada alguma condição inicial. Outro ambiente para modelagem em Mecânica muito conhecido está presente nos livros de física: a grosso modo ele é composto por desenhos em papel, números e expressões matemáticas. Com o ferramental deste ambiente para modelagem é possível, agora, não só descrever os fenômenos da Mecânica mas, também, prever acontecimentos. Mas, o mundo da mecânica não é habitado, apenas, por números, equações matemáticas e desenhos estaticamente representados no papel. No mundo da mecânica, os corpos parecem ter vida dado que alguns deles movem-se e as pessoas podem vê-los movimentar; aos olhos de qualquer pessoa, observar a ocorrência de um fenômeno da Mecânica é provar da sensação de ver corpos que podem ou não se movimentar à medida em que o tempo transcorre. Qualquer ambiente para modelagem em Mecânica, que não permita ao modelador a experiência de ver as representações dos corpos se movimentar arremessa este modelador ao esforço adicional de associar as suas representações sobre um fenômeno e o fenômeno vivo no mundo e lhe nega o engajamento direto com os objetos do domínio.

Desejamos que o ambiente para modelagem convide o modelador para um engajamento direto com os objetos representados. Ou seja, queremos que o ambiente para modelagem se apresente para o modelador como se fosse o próprio domínio da Mecânica. O modelador deve experimentar a sensação de estar agindo sobre os próprios objetos do domínio e participar da fantasia de criar peças que possuem comportamento mecânico e que se inter-relacionam. Acreditamos que, para isto, o ambiente para modelagem deve prover meios para:

1- animar os corpos, uma vez que, na vida real, os corpos apresentam-se "animados";
2- permitir que o modelador construa os corpos que participarão do fenômeno tal como, na vida real, o engenheiro projeta e o fundidor forja as peças mecânicas;

3- definir o comportamento mecânico de cada corpo envolvido no fenômeno;
4- possibilitar que os corpos que já tiverem os seus comportamentos mecânicos definidos possam formar unidades mecânicas de modelos mais complexos. Isto cria, para o modelador, a sensação de estar produzindo peças mecânicas que podem ser utilizadas em estruturas ou máquinas mais complexas. Por exemplo, o modelador pode construir um motor; em seguida este motor pode ser utilizado para "tocar" uma moenda de cana de açúcar ou um trator de esteiras.

Retomando as orientações de Laurel (1986) para se conseguir engajamento direto, apresentadas na seção 4.4 do capítulo 4, podemos afirmar que os itens 1, 2 e 3 apresentam os requisitos para que o ambiente represente os próprios objetos do domínio e que o item 4 objetiva criar o que Laurel chama de linguagem de entrada e saída inter-referenciável, isto é, uma linguagem que admite que uma expressão de entrada faça uso ou incorpore uma expressão anterior.

Organizamos a complexidade de se estabelecer o *design* de cada conceito atômico que devemos considerar no desenvolvimento do ambiente para modelagem segundo a estrutura geral que apresentamos na figura 5.9.

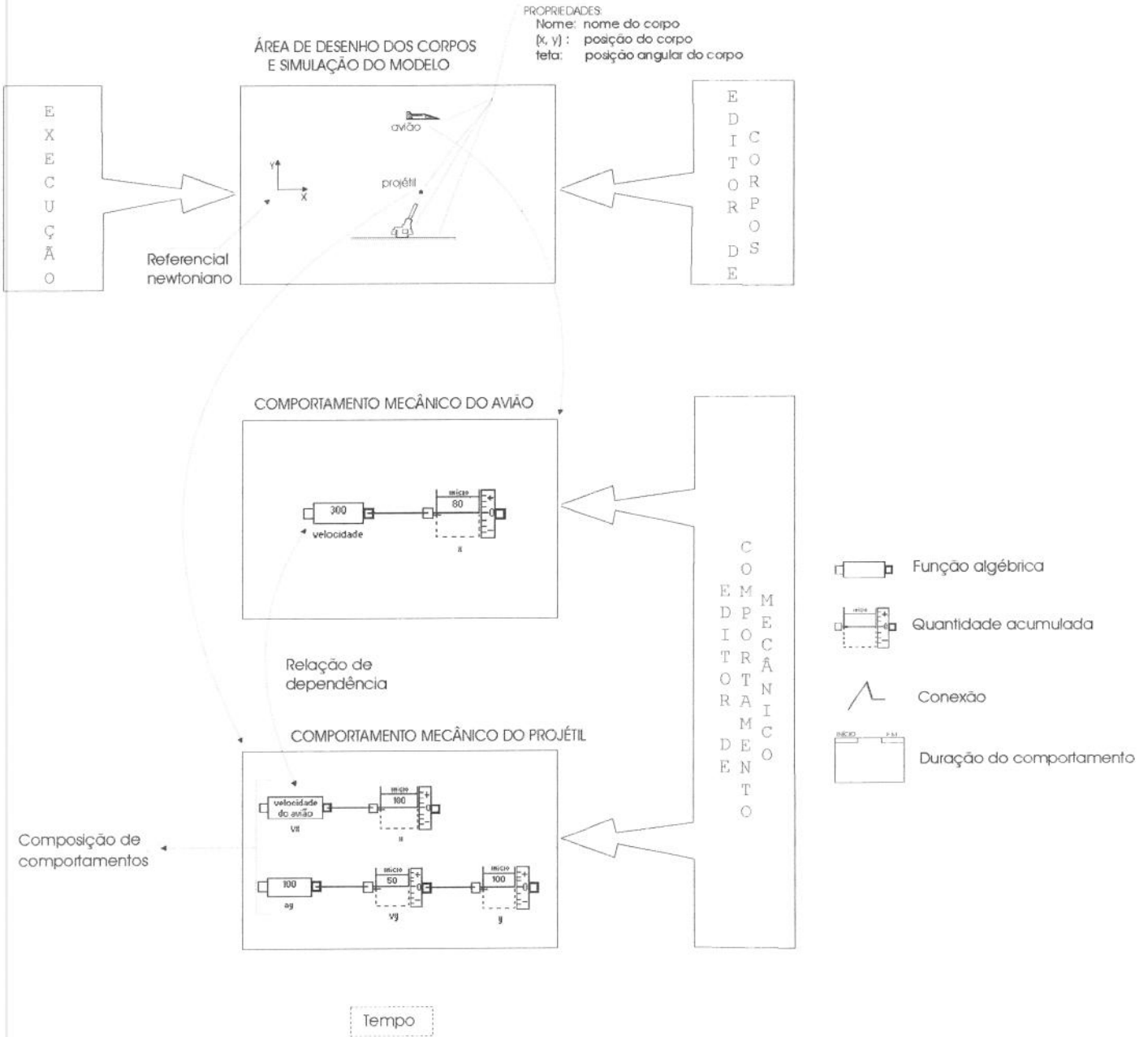


Fig. 5.9: Estrutura geral do ambiente

A figura 5.9 mostra que o ambiente para modelagem está composto por um conjunto integrado de ferramentas que permitem editar a aparência e o comportamento mecânico dos corpos e executar o modelo estabelecido. Um par de eixos cartesianos que, como veremos no próximo capítulo, pode ser colocado em qualquer posição da área de desenhos e simulação do comportamento dos corpos, demarca a origem do sistema de medidas. Este par de eixos é a primitiva do ambiente que implementa o conceito atômico de referencial newtoniano. Cada corpo tem uma aparência, possui propriedades e pode ter um comportamento mecânico associado. A aparência de um corpo pode ser expressa através de um desenho que o represente. Os comportamentos mecânicos são definidos por meio de diagramas compostos por primitivas que implementam os conceitos atômicos de função algébrica, quantidade acumulada, conexão, duração do comportamento, relação de dependência entre corpos e composição de comportamentos. As propriedades de um corpo são primitivas que implementam os conceitos atômicos de nome, posição (x, y) e posição angular de um corpo (teta). O *design* do ambiente é tal que ele reage, automaticamente, com uma variação de posição de um corpo toda vez em que o valor de uma de suas propriedades x, y ou teta é alterada. A inter-relação entre os comportamentos mecânicos dos corpos está implementada de forma simbólica por uma primitiva que permite construir referências tais como "velocidade do avião" ou "velocidade do corpo avião" e que significam, indistintamente, valor da velocidade do corpo de nome "avião". A primitiva que implementa o conceito atômico de composição de comportamentos não possui uma representação icônica ou simbólica mas, contudo, está implícita na atividade projetada para o ambiente, ou seja, se dois ou mais fragmentos de diagrama estabelecerem comportamentos mecânicos simples, o comportamento global do corpo será a composição dos comportamentos simples. Outra primitiva que não possui representação icônica ou simbólica é a que implementa o conceito atômico tempo. Durante a execução do modelo, o ambiente varia automaticamente o valor do tempo. No entanto, o valor corrente do tempo pode ser conhecido usando-se a palavra "tempo" dentro de uma expressão onde o seu valor for necessário.

O princípio da redução da distância, visto no capítulo 4, estabelece que devemos reduzir a distância entre o usuário e o ambiente para modelagem. Deve-se trabalhar o *design* do ambiente para modelagem para reduzir dois tipos de distâncias: a semântica e a articulatória. A distância semântica reflete o relacionamento entre a intenção do usuário em executar uma ação e o significado da expressão da ação na linguagem do ambiente para modelagem. Uma vez que, na seção anterior, trabalhamos para estabelecer um conjunto de conceitos atômicos adequado para a classe de usuários que queremos

privilegiar no *design,* esperamos estar contribuindo no casamento da intenção do usuário com os elementos disponíveis na linguagem do ambiente para modelagem. Por conseqüência, já fizemos o que podíamos para reduzir a distância semântica. A distância articulatória reflete o relacionamento entre a forma física da expressão na linguagem do ambiente para modelagem e o seu significado. As orientações de Hutchins (1986) para redução da distância articulatória, comentadas no capítulo 4, nos levaram a definir as primitivas apresentadas na figura 5.10. O *design* destas primitivas é tal que, as suas formas físicas guardam similaridades com os seus respectivos significados.

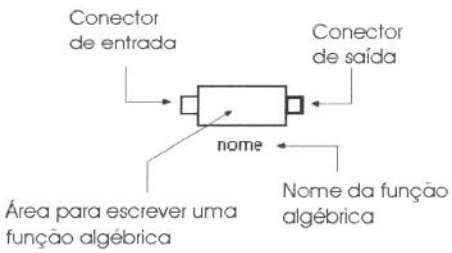
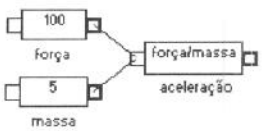
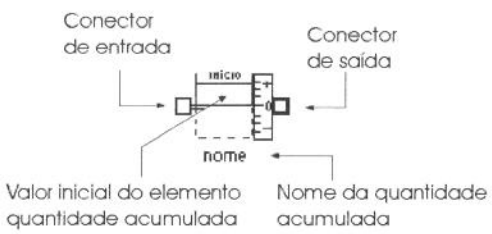
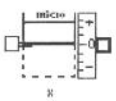
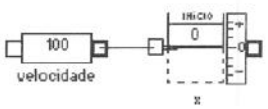
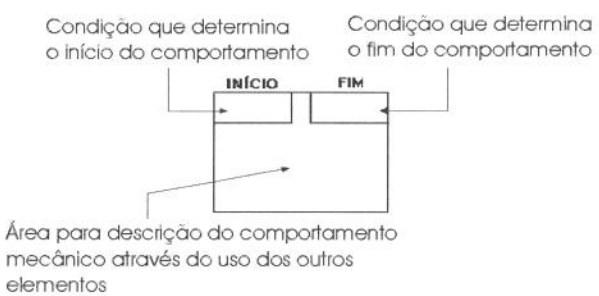
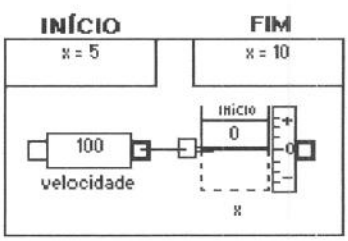
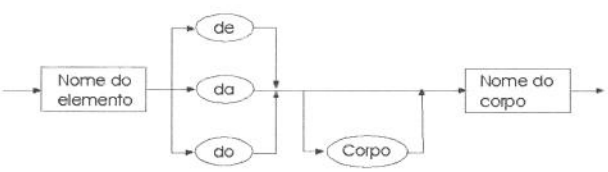
CONCEITO ATÔMICO	*PRIMITIVA*	*EXEMPLO*
Referencial newtoniano		
Nome, posição e posição angular do corpo	Propriedades de um corpo: nome, x, y e teta	
Função Algébrica		
Quantidade acumulada		
Conexão		
Duração do comportamento		
Relação de dependência entre corpos		x do avião

Fig. 5.10: Os conceitos atômicos e as primitivas que os implementam

Assim, as primitivas para expressão do comportamento mecânico lembram exatamente o que representam. A primitiva quantidade acumulada lembra uma caixa d'água. Isto porque, tais primitivas podem ter o seu valor aumentado ou diminuído, no decorrer do tempo, pela taxa de variação conectada à sua entrada, assim como, caixas d'água podem ter o volume de água em seu interior aumentado ou diminuído em função dos fluxos de água que chegam e saem da mesma. A primitiva conexão lembra fios ou tubulações que conduzem valores desde o conector de saída de um elemento do tipo quantidade acumulada ou função algébrica até o conector de entrada de outro. A primitiva duração do comportamento é um retângulo que envolve e demarca os fragmentos do diagrama cujo comportamento descrito tem início e fim estabelecidos pelos campos INÍCIO e FIM presentes nesta primitiva. A primitiva função algébrica é um retângulo dentro do qual se pode escrever uma função algébrica de forma similar à escrita de funções da Álgebra. A primitiva relação de dependência entre corpos é uma estrutura simbólica que utiliza a preposição "de" ou as contrações da preposição "de" com os artigos "a" ou "o" para expressar uma relação possessiva de um corpo sobre uma propriedade como em "velocidade do avião". No mundo real, o tempo transcorre sem que para isto tenhamos que fazer alguma coisa. Similarmente, a primitiva tempo implementa o conceito atômico de tempo automaticamente, isto é, sem que se tenha que explicitar os valores assumidos pelo tempo.

As primitivas cujo *design* descrevemos buscam uma similaridade articulatória com os seus respectivos significados e representam um esforço para reduzir a distância articulatória. O *design* destes elementos, em geral, depende da tecnologia de entrada e saída que está sendo empregada. Para que possamos sentir a dimensão desta afirmação, suponhamos que a tecnologia de realidade virtual fizesse parte do conjunto de tecnologias que estamos considerando em nosso trabalho de *design*. Se assim fosse, poderíamos estabelecer o *design* de um ambiente para modelagem em Mecânica onde fosse possível, por exemplo: (1) construir, virtualmente, os corpos com as próprias mãos. As peças que entram na composição "física" dos corpos poderiam ser tomadas de um depósito de suprimentos e as ferramentas para trabalhar as peças poderiam ser obtidas em uma caixa de ferramentas; (2) fabricar, virtualmente, o comportamento mecânico em placas sobre as quais os elementos de definição do comportamento poderiam ser soldados uns aos outros e, após a fabricação, as placas poderiam ser inseridas dentro dos corpos para conferir-lhes comportamento; (3) executar, parar e reiniciar a execução, virtualmente, tocando o dedo em um dos botões destinados a estas finalidades. Entretanto, com o uso da tecnologia que dispomos (vide seção 5.1), estabelecemos o *design* de um ambiente onde: (1) pode-se

construir os corpos, não com as mãos, mas através de um conjunto de operações realizadas com o auxílio do *mouse.* Os elementos que entram na composição do desenho dos corpos, não são tomados de um depósito de suprimentos de peças, mas de uma palheta contendo as primitivas apropriadas para o desenho dos mesmos; (2) pode-se definir o comportamento mecânico dos corpos, não em placas, mas através de diagramas de comportamento mecânico elaborados com o auxílio do *mouse* e de uma palheta contendo as primitivas apropriadas para definição do comportamento mecânico; (3) pode-se executar, parar e reiniciar a execução tocando em botões destinados a estas finalidades, não com o dedo, mas com o ponteiro do *mouse.*

O capítulo 6, cujo objetivo é o de caraterizar, sob o ponto de vista do usuário, o ambiente para modelagem que estamos propondo, detalha o *design* de cada um destas primitivas e do ambiente como um todo.

5.5 - ESPECIFICANDO A INFRA-ESTRUTURA DO AMBIENTE

Algumas estruturas adicionais àquelas que visam implementar os conceitos atômicos são necessárias para o funcionamento geral do ambiente. Estas estruturas apoiam as tarefas exercidas pelas primitivas que implementam os conceitos atômicos, criando o que chamamos de infra-estrutura do ambiente. O capítulo 6 discute em detalhes o *design* de cada uma delas. A seguir descrevemos quais são os elementos que compõe esta infra-estrutura e para que servem.

a - **Infra-estrutura para definição da aparência dos corpos.** Corpos possuem aparência e comportamento Mecânico. O comportamento mecânico será definido pelas primitivas que discutimos. A aparência poderá ser definida por desenhos em duas dimensões a partir dos elementos:

- retângulo;
- círculo;
- linha reta;
- arco;
- linha com ângulos;
- polígono.

Além destes elementos, o acabamento dos desenhos e a produtividade no trabalho de

desenhar podem ser melhorados por:

- **especificação de cores, padrões de preenchimento e espessura de linhas** para os corpos representados;
- **biblioteca de desenhos de corpos comuns.** Uma biblioteca de desenhos de corpos que aparecem freqüentemente nos fenômenos da Mecânica é útil para poupar o trabalho de desenho e deixar que o usuário se concentre na definição do comportamento mecânico dos corpos;
- **lupa.** Para permitir a edição de detalhes do desenho de um corpo, uma "lupa" poderá ser utilizada para ampliar ou reduzir os desenhos enquanto eles estão sendo compostos;
- **copiar/colar/cortar.** Estas estruturas estão presentes no *design* de todas as aplicações Windows. Elas permitem trocar informações com o *clipboard* (área de transferência) do Windows. A operação copiar permite que um objeto, previamente selecionado, seja copiado para o *clipboard;* a operação colar copia um objeto, previamente inserido no *clipboard,* para o modelo; a operação cortar retira um objeto do modelo e o insere no *clipboard.* Estas estruturas podem ser úteis, por exemplo para: (1) replicar desenhos e (2) eliminar desenhos indesejáveis;
- **agrupar/desagrupar.** Alguns corpos podem ser entendidos como formados por um agrupamento de dois ou mais corpos menores. Estas estruturas permitirão agrupar e desagrupar o desenho de tais corpos;
- **corpos especiais (molas e cabos).** Molas e cabos, freqüentemente, figuram nos modelos da mecânica dos corpos rígidos apesar de não serem corpos rígidos, já que sofrem deformações significativas quando forças agem sobre eles. Assim é desejável poder utilizá-los nos modelos a serem estabelecidos. Em nosso ambiente para modelagem tanto a mola quanto o cabo terão um comportamento mecânico predefinido. Tudo o que o modelador terá que fazer é estabelecer valores para os parâmetros da mola ou do cabo e inseri-los no contexto do modelo que está sendo estabelecido;
- **réguas e grade.** Um par de réguas e uma grade de pontos poderão ajudar nos ajustes das dimensões e posicionamento dos desenhos dos corpos;
- **ajuste do desenho.** Ajuda no arremate de um modelo poder tornar um corpo transparente, alinhar os corpos uns em relação aos outros e acertar um desenho já pronto através de um ajuste fino da posição dos vértices do mesmo.

b - Infra-estrutura para definição do comportamento mecânico:

- **comentário**. Permitirá associar um texto, em formato livre, com o objetivo, por exemplo, de esclarecer o significado de uma parte ou de todo o diagrama de comportamento mecânico;
- **teste de sintaxe**. Permitirá verificar se as construções diagramáticas estão corretas e se as expressões definidas estão sintaticamente corretas;
- **especificação de cores para os elementos do diagrama**. As cores podem ajudar o modelador a diferenciar partes conceitualmente distintas do diagrama de comportamento mecânico de um modelo. Isto pode constituir um excelente ferramental para ajudar na organização de um diagrama grande ou complexo;
- **copiar/colar/cortar**. Da mesma forma como no desenho dos corpos estas estruturas podem ser úteis para: (1) replicar desenhos e (2) eliminar desenhos indesejáveis;
- **imprimir** um diagrama estabelecido para fins de estudo ou documentação;
- **encerrar a edição de comportamento mecânico**. Permitirá encerrar a edição do comportamento mecânico com duas possibilidades: (1) consolidando as alterações que, por ventura, possam ter sido feitas e (2) não consolidando as alterações feitas, caso em que o comportamento mecânico do corpo permanecerá como estava antes da edição do mesmo.

c - **Infra-estrutura para avaliação do modelo criado**. Como vimos no capítulo 1, um ambiente computacional para modelagem deve prover meios para que o modelador possa executar e verificar os resultados do modelo criado. Neste sentido, o nosso ambiente para modelagem mecânica irá conter os seguintes elementos:

- **instrumentos de *feedback* (cronômetro, distanciômetro, conta-giros, valor de propriedade, tabela e gráfico)**. Uma vez "instalados" em um modelo, estes instrumentos permitirão acompanhar a sua execução. O cronômetro permitirá medir o tempo transcorrido entre dois eventos; o distanciômetro tornará possível medir a distância entre dois corpos; o conta-giros medirá a distância angular percorrida por um corpo; os valores assumidos por uma propriedade, no decorrer da execução, poderão ser exibidos pelo instrumento valor de propriedade; várias propriedades poderão ter os seus valores, assumidos ao longo da execução, apresentados juntos em uma tabela (instrumento tabela) ou mostrados em um gráfico (instrumento gráfico);
- **controle de execução**. A execução do modelo poderá ser interrompida, reiniciada ou cancelada em qualquer instante. A interrupção da execução pode ser especialmente útil para que se possa examinar, com tranqüilidade, os valores

registrados pelos instrumentos de *feedback* instalados, bem como a disposição geral dos corpos;

- **estabelecimento de pontos de parada.** A interrupção da execução de um modelo poderá ser previamente planejada pelo estabelecimento de condições de parada baseadas na ocorrência de determinados eventos. Por exemplo, interromper a execução quando o tempo for igual a 5 (cinco) ou quando a propriedade x do corpo "A" for igual à propriedade x do corpo "B".

d - **Infra-estrutura geral do ambiente para modelagem:**

- **Mais de um modelo aberto ao mesmo tempo.** A abertura, o desenvolvimento e a execução de mais que um modelo ao mesmo tempo poderá dar, ao modelador, a oportunidade de comparar e refletir sobre variações de versões distintas de um mesmo modelo;
- **Abrir um modelo já existente/ abrir um novo modelo/ gravar/gravar um modelo especificando um novo nome.** Este conjunto de estruturas, comum a qualquer aplicativo Windows, permitirá efetuar as tradicionais tarefas de manipular os arquivos associados à aplicação; neste caso, estes arquivos armazenam as informações sobre os elementos que compõem um modelo;
- **ajuda.** Poderá ser utilizada uma ajuda (*help*) ao uso do ambiente para modelagem;
- **imprimir** todo o modelo para fins de estudo ou documentação;
- **encerrar** a execução do ambiente para modelagem.

Capítulo 6

NEWTON: um protótipo de ambiente para modelagem

O NEWTON é um protótipo de ambiente para modelagem que desenvolvemos com base nas estratégias de *design* empregadas no capítulo anterior. Neste capítulo, procuramos caracterizar o ambiente para modelagem NEWTON do ponto de vista do usuário. Apresentamos o NEWTON como um produto comercial é mostrado para o seu usuário. Isto é, os elementos que compõem o NEWTON são apresentados e a função que cada um deles exerce é explicada sem que, para isto, tenhamos que apelar, por exemplo, para as fundamentações de Matemática e Física subjacentes ao seu *design* já abordadas em capítulos anteriores. Apresentamos, também, uma coleção de idéias e comentários sobre o trabalho de se criar modelos com o NEWTON e discutimos algumas melhorias para o ambiente. Dois apêndices complementam e podem ser utilizados no transcorrer da leitura deste capítulo. O apêndice A, que apresenta um guia de referência rápida ao uso do NEWTON e o apêndice B, que mostra exemplos de modelos que podem ser criados com este ambiente.

Tópicos deste capítulo:

Tópico		*Página*
6.1 -	Introdução	99
6.2 -	Generalidades do NEWTON	101
6.3 -	Desenhando corpos	107
6.4 -	Diagramas de comportamento mecânico	114
6.5 -	Editando diagramas de comportamento mecânico	121
6.6 -	Executando modelos e instalando os instrumentos de *feedback*	126
6.7 -	Um exemplo de modelagem com o NEWTON	131
6.8 -	Idéias e comentários sobre o trabalho de se criar modelos com o NEWTON	134
6.9 -	O suporte dado pelo NEWTON aos estágios da execução e de avaliação de uma ação	138
6.10 -	Melhorias	139

6.1 - INTRODUÇÃO

Conforme mencionado no capítulo anterior, o NEWTON é o protótipo de ambiente computacional para modelagem que desenvolvemos com base nos princípios enunciados. A nossa intenção, com o desenvolvimento deste protótipo, foi a de investigar a aplicação dos referidos princípios bem como viabilizar análises futuras sobre o *design* de ambientes computacionais para modelagem para aplicação em um contexto educacional.

O NEWTON permite a modelagem dentro do domínio da Mecânica. Para tal, NEWTON permite a representação de corpos através de desenho em duas dimensões, o estabelecimento do comportamento mecânico destes e a descrição de aspectos correspondentes à inter-relação dos corpos representados. O usuário pode desenhar os corpos ou importá-los de uma biblioteca de desenhos de corpos prontos. Cada corpo possui propriedades padrão, criadas automaticamente com ele tais como: nome, posição (**x**, **y**) em relação a um par de eixos cartesianos estabelecido como origem e posição angular (**teta**). Propriedades adicionais podem ser criadas pelo usuário. Estas propriedades podem ser referenciadas quando da definição do comportamento dos corpos ou na definição da inter-relação entre estes. O comportamento mecânico de um corpo pode ser definido através de uma linguagem diagramática composta pelos elementos ilustrados na figura 6.1.

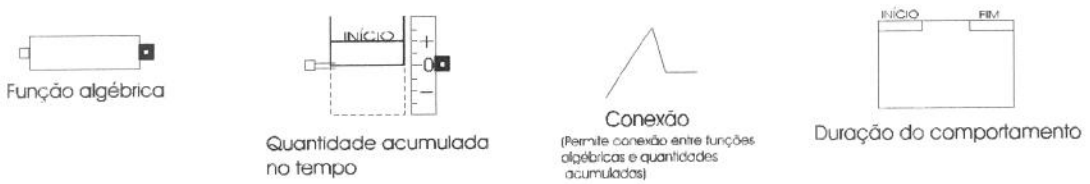


Fig. 6.1: Elementos para construção de diagramas para definição de comportamento mecânico

Para ilustrar o uso que pode ser feito do ambiente, consideremos a tarefa de modelar o movimento de dois automóveis: um deslocando-se com velocidade constante e outro com velocidade variada sob uma aceleração constante, ambos em linha reta. O que o usuário deve fazer? Desenhar os automóveis ou importá-los da biblioteca de desenhos de corpos (figura 6.2-a) e posicionar um par de eixos cartesianos para estabelecer a origem de

medida das posições. Estabelecer o comportamento mecânico de cada um deles, fazendo um duplo *click* em cada automóvel e construir diagramas que expressem o comportamento mecânico do automóvel usando um editor apropriado (figura 6.2-b). Um elemento do tipo **função algébrica** necessita de uma expressão que o defina e um elemento que é uma **quantidade acumulada** no transcorrer do tempo exige a definição de um valor inicial. Assim, uma vez editado, o diagrama se apresentará com alguns elementos que necessitam de definição de seu conteúdo. Podemos observar no exemplo, também, que as quantidades acumuladas denominadas **x**, presentes nos diagramas que definem o comportamento de cada um dos automóveis, já apresentam os seus conteúdos preenchidos (figura 6.2-b). Isto se deve ao fato de ser **x** a propriedade padrão que representa a abscissa de um corpo em relação ao par de eixos cartesianos estabelecido como origem pelo usuário e, desde que ele tenha desenhado e posicionado os automóveis na tela o valor inicial de **x** será conhecido automaticamente pelo ambiente. Entretanto a definição do conteúdo dos demais elementos poderá ser feita dando-se um duplo *click* nestes e editando, em seguida, uma expressão que estabelece o seu valor. A figura 6.2-c resume os valores que poderiam, hipoteticamente, ser definidos para cada elemento. Caso deseje, o usuário pode, ainda, definir aspectos vinculados à inter-relação entre os corpos e/ou a duração de determinado comportamento mecânico, como por exemplo, usar o elemento **duração do comportamento** para modelar a interrupção do movimento dos automóveis se o automóvel 1 da figura 6.2 for alcançado pelo automóvel 2.

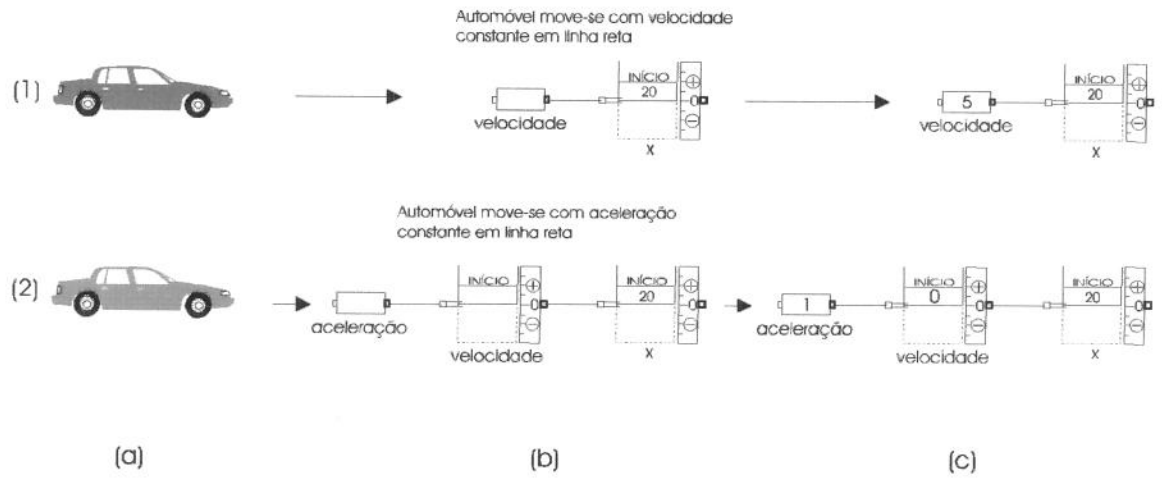


Fig. 6.2: Um modelo típico estabelecido no ambiente para modelagem NEWTON

Para execução do modelo, o usuário pode solicitar do ambiente, gráficos que demonstrem as relações entre os elementos usados na definição do comportamento dos corpos ou a relação destes com o tempo. Por exemplo, para execução do modelo da figura 6.2 o usuário pode solicitar a exibição de gráficos tais como: **velocidade** do automóvel 1 contra o **tempo**, **x** do automóvel 2 contra o **tempo**, **x** do automóvel 1 contra **x** do automóvel 2. O usuário poderá, ainda, solicitar para execução do modelo, o acionamento de cronômetros, o uso de distanciômetros para medir a distância entre os corpos ou a exibição dos valores assumidos, ao longo da execução, por qualquer elemento usado na definição do comportamento dos corpos. A execução do modelo mostrará os gráficos, cronômetros, distanciômetros e valores de elementos que o usuário tenha solicitado a exibição e, também, a animação do comportamento mecânico dos corpos envolvidos no modelo.

6.2 - GENERALIDADES DO NEWTON

Um fenômeno em Mecânica ocorre com a participação de um ou mais corpos que podem ou não se relacionarem entre si. Um ambiente para modelagem em Mecânica deve prover meios para representar os corpos bem como descrever o comportamento mecânico e as relações entre os corpos. Ainda, no sentido de permitir que o modelador observe os resultados e a validade do modelo criado, o ambiente para modelagem deve prover ferramentas adicionais que permitam acompanhar a execução do modelo. Assim, o NEWTON dispõe de ferramentas que permitem:

- modelar a aparência dos corpos envolvidos em um fenômeno, através de desenhos em duas dimensões. Deste assunto trata a seção 6.3;
- modelar o comportamento mecânico e as relações entre os corpos. Disto trata as seções 6.4 e 6.5;
- executar o modelo criado e instalar instrumentos (cronômetros, distanciômetros etc.) que permitam ao modelador acompanhar e verificar a validade do modelo criado. Na execução, o NEWTON mostrará a animação dos corpos envolvidos no modelo e os valores registrados pelos instrumentos instalados. Este assunto será abordado na seção 6.6.

A atividade de modelar um fenômeno no NEWTON pode ser entendida, portanto, como a soma de duas outras atividades: (1) modelar a aparência de um corpo e (2) modelar o comportamento mecânico e as relações entre os corpos. As demais atividades mencionadas anteriormente, execução e instalação de instrumentos, não são propriamente atividades de modelagem. Contudo, são de vital importância para a verificação dos resultados e validação do modelo criado. Antes de descrevermos os elementos do NEWTON que permitem desenvolver estas atividades, mostraremos como dar partida, abrir, fechar, gravar e sair deste ambiente.

O NEWTON foi desenvolvido para executar no sistema operacional **Microsoft Windows**. Logo, pode-se dar partida e encerrar a execução do NEWTON exatamente da mesma forma como qualquer outra aplicação **Windows**. Assim, para dar partida no NEWTON deve-se dar um duplo *click* no ícone do NEWTON presente no grupo NEWTON do *program manager* do **Windows** (figura 6.3-a). Após esta ação, a janela do NEWTON aparecerá como na figura 6.3-b.

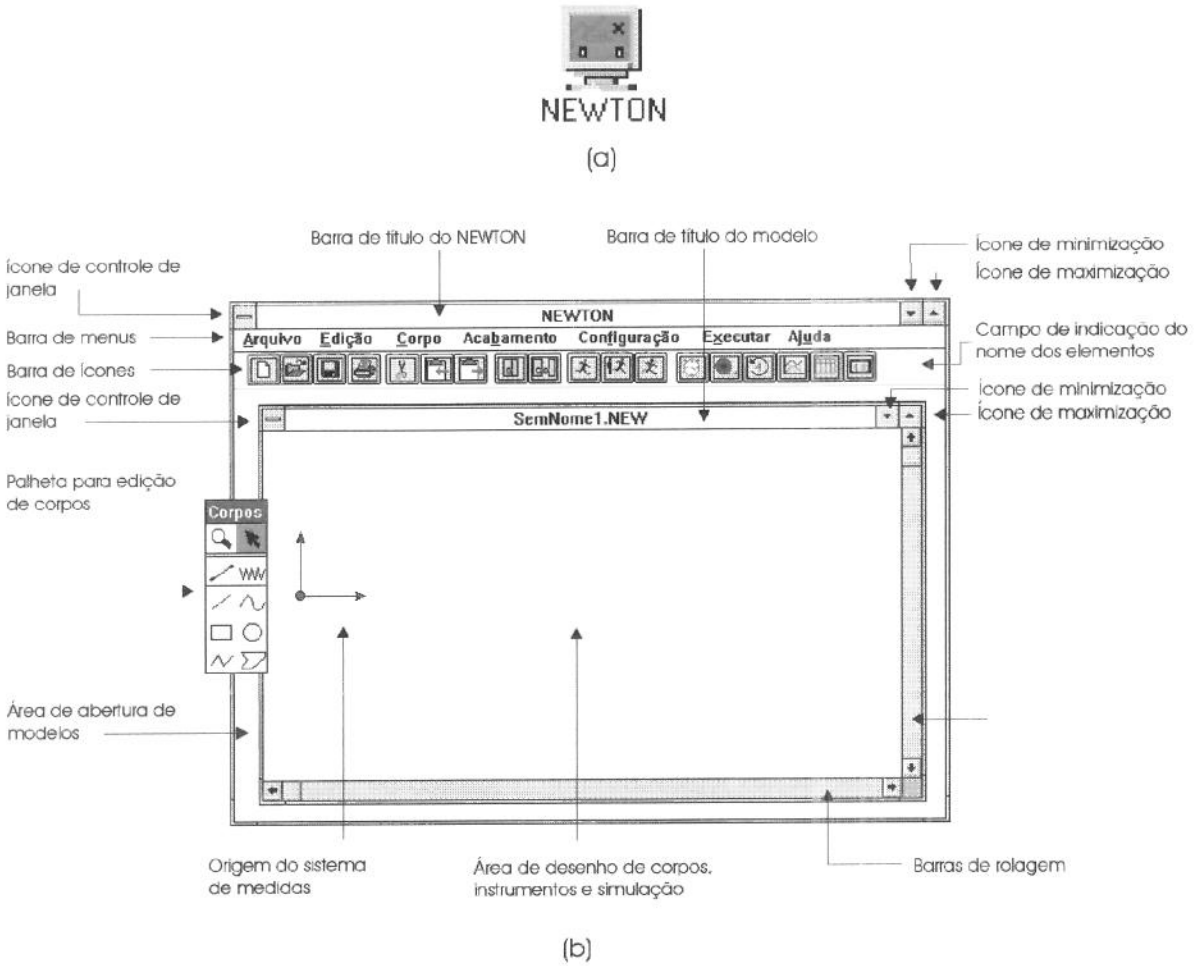


Fig. 6.3: (a) Ícone do NEWTON presente no *program manager* do Windows; (b) a janela do NEWTON com um modelo (SemNome1.NEW) aberto em seu interior

Ao ter a sua execução iniciada, o NEWTON coloca à disposição um modelo "em branco" denominado "SemNome1.NEW". Como se pode observar na figura 6.3, muitos elementos gráficos usados na janela do NEWTON são comuns a outros aplicativos Windows. Esta janela contém os seguintes elementos:

- barra de título do NEWTON: contém o título do aplicativo, neste caso, NEWTON;

- barra de menus: mostra os nomes dos menus de comandos disponíveis.

• ícone de controle da janela do NEWTON e da janela do modelo:	quando este ícone recebe um *click* ele apresenta os comandos para se trabalhar com janelas no **Windows**. Estes comandos são comuns a qualquer janela do Windows (minimizar, maximizar, restaurar, mover, fechar etc.);
• ícone de maximização da janela do NEWTON:	expande a janela do NEWTON ao tamanho máximo da tela;
• ícone de minimização da janela do NEWTON:	reduz a janela do NEWTON a um ícone;
• barra de ícones:	contém ícones para os principais comandos que podem ser dados pelo usuário. A maioria destes comandos podem ser acionados, também, através do uso dos menus. Entretanto, o uso destes ícones diminui o esforço do usuário;
• campo de indicação do nome dos elementos:	indica o nome do elemento sobre o qual o ponteiro do *mouse* está posicionado. Por exemplo, posicionando o cursor do *mouse* sobre um ícone da barra de ícones o nome deste ícone será exibido neste campo. Isto ajuda o iniciante a se familiarizar com o NEWTON e auxilia o veterano a lembrar da função de algum elemento;
• área de abertura de modelos:	área do NEWTON situada abaixo da barra de ícones que é usada para abertura das janelas dos modelos. Mais que um modelo pode ser aberto, ao mesmo tempo, neste ambiente;
• barra de título do modelo:	contém o título do modelo. No exemplo, o título é "SemNome1.NEW". O título de um modelo pode ser alterado;
• ícone de maximização do modelo:	expande a janela do modelo ao tamanho máximo da área de abertura de modelos do NEWTON;

• ícone de minimização do modelo:	reduz a janela do modelo a um ícone dentro da área de abertura de modelos do NEWTON;
• barras de rolagem:	uma página de um modelo pode ser maior que a janela do mesmo. As barras de rolagem podem ser utilizadas para trazer o resto da página à vista;
Origem do sistema de medidas	par de eixos cartesianos que determinam a origem (coordenada 0, 0) para um determinado modelo. Este par de eixos pode ser movimentado ou ter os seus sentidos alterados segundo conveniência do usuário. Geralmente, existem posições deste par de eixos que facilitam a expressão do modelo;
• área de desenho dos corpos e simulação	área da janela do modelo destinada ao desenho dos corpos. É nesta área, também, que o NEWTON realizará a execução do modelo e que o usuário instalará os instrumentos (cronômetro, distanciômetro etc.) que o permitirão acompanhar a execução do modelo;
• palheta para edição de corpos:	coleção de ferramentas que podem ser utilizadas para a criação do desenho dos corpos. Esta palheta pode ser "arrastada na tela" para a posição que se desejar.

Existem quatro formas diferentes de se acionar um comando no NEWTON: (1) através do *mouse*, dando um *click* no nome de um menu da barra de menus e escolhendo, em seguida, com outro *click* o comando desejado; (2) através do *mouse*, dando um *click* em algum dos ícones da barra de ícones; (3) pressionando a tecla ALT em conjunto com uma das letras sublinhadas do menu e, em seguida, uma das letras sublinhadas dos comandos dentro do menu; (4) através do uso de teclas de atalho como, por exemplo, CTRL G que aciona o comando "Gravar". Obviamente, não existem ícones e teclas de atalho para todos os comandos. Contudo, o NEWTON facilita o esforço do usuário tornando disponíveis ícones e teclas de atalho para os comandos mais comuns. Referência completa dos comandos, ícones, menus e teclas de atalho é apresentada no apêndice A.

Mais que um modelo pode ser aberto ao mesmo tempo e, no entanto, vários comandos só se aplicam a um modelo de cada vez. Para resolver este problema o NEWTON trabalha com o conceito de modelo corrente ou ativo ou selecionado. Para selecionar um modelo deve-se dar um *click* em qualquer ponto da janela do modelo. Após isto, a barra de título do modelo se apresentará destacada para diferenciar o modelo selecionado dos demais modelos que, por ventura, estejam abertos.

As janelas dos modelos podem ser movimentadas, redimensionadas, maximizadas ou transformadas em um ícone dentro da área destinada a abertura de modelos do NEWTON. Estas operações permitem organizar a área de trabalho de acordo com a necessidade que o usuário tem em um dado momento.

Pode-se gravar as trocas feitas em um modelo usando os comandos "Gravar" ou "Gravar como". Um modelo Pode ser gravado sob um nome diferente do apresentado em sua barra de título usando o comando "Gravar como". O comando "Novo" permite que seja aberto um novo modelo "em branco". Ele será criado, inicialmente, com o nome SemNome**i**.NEW, onde **i** é um número inteiro associado pelo NEWTON. Escolhendo o comando "Abrir", o NEWTON apresentará uma caixa de diálogo onde serão mostrados os modelos disponíveis. O usuário poderá, então, escolher o modelo a abrir e, se for o caso, poderá pesquisar o conteúdo de outros diretórios ou *drive*. O comando "Imprimir" pode ser utilizado para produzir uma documentação geral de um modelo selecionado. Pode-se fechar um modelo aberto dando-se um duplo *click* em seu ícone de controle ou pressionando ALT mais F4 quando o modelo estiver selecionado. Caso trocas tenham sido feitas no modelo e estas não tenham sido gravadas, o NEWTON emitirá uma mensagem de advertência. O usuário poderá, então, escolher entre gravar ou não gravar as trocas feitas, ou ainda, desistir da operação de fechar o modelo. A execução do NEWTON pode ser encerrada através do comando "Sair". Quando se encerra a execução do NEWTON a tela volta a apresentar o Windows. Caso o encerramento se dê sem gravar as trocas feitas em algum dos modelos abertos, o NEWTON apresentará mensagens de advertência para cada modelo onde isto tiver ocorrido. Para cada mensagem de advertência o usuário pode escolher gravar ou não gravar as trocas feitas. Referência completa dos comandos mencionados pode ser encontrada na seção A.3 do apêndice A.

6.3 - DESENHANDO CORPOS

Pode-se criar desenhos de corpos no NEWTON de forma similar a diversos *software* para desenho (**PaintBrush**, **CorelDraw**, **Harvard Graphics** for Windows etc.) disponíveis para ambiente Windows. A palheta **Corpos** contém todas as ferramentas para desenho de corpos (figura 6.4). Podemos dividir as ferramentas disponíveis na palheta Corpos em três categorias: ferramentas de ajuda na edição (lupa e seta de seleção), ferramentas com comportamento mecânico predefinido (mola e cabo inextensível) e ferramentas sem comportamento mecânico predefinido (linhas, arcos, retângulos, polígonos irregulares e círculos).

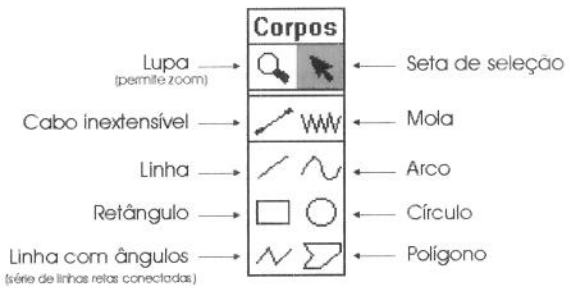


Fig. 6.4: A palheta de desenho de corpos

A palheta Corpos pode ser movimentada na tela para a posição que melhor convier ao usuário em um determinado momento. Basicamente, o desenho de um elemento gráfico contido na palheta pode ser feito: (1) selecionando o elemento na palheta (*click* no ícone deste); (2) editando o elemento, na área apropriada do modelo, por apertar e manter pressionado o botão esquerdo do *mouse* à medida em este é arrastado em uma direção aproximadamente diagonal (o desenho do elemento gráfico irá aparecer e crescerá à medida em que o *mouse* é deslocado); (3) soltando o botão esquerdo do *mouse* quando o tamanho desejado para o elemento gráfico for atingido. Com algumas técnicas adicionais, pode-se obter, por exemplo, desenhos de quadrados a partir da ferramenta retângulo ou obter-se desenho de linhas com ângulos múltiplos de 45 graus.

Muitos dos comandos do NEWTON recaem sobre os corpos selecionados. Para selecionar um corpo desenhado deve-se escolher a seta de seleção da palheta Corpos e dar um *click* no corpo desejado. Para selecionar mais que um corpo de uma só vez deve-se

pressionar a tecla SHIFT enquanto dá-se *click* em cada corpo. Para selecionar todos os corpos desenhados pode-se usar o comando "Selecionar tudo" do menu "Edição". Os corpos selecionados aparecerão com os marcadores de seleção ao seu redor (figura 6.5).

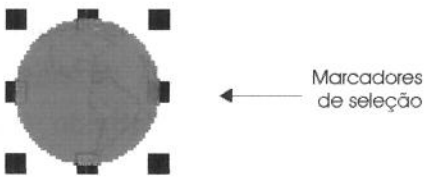


Fig. 6.5: Marcadores de seleção mostram que um corpo está selecionado

Quando vários corpos estão selecionados o próximo comando ou ação sobre o *mouse* afetará todos os corpos selecionados. Assim pode-se, por exemplo, remover, alterar a cor de preenchimento ou movimentar vários corpos de uma só vez. Um corpo permanece selecionado até que se cancele a sua seleção. Para cancelar uma seleção deve-se dar um *click* em algum ponto da área de edição que não possua um corpo desenhado.

Pode-se movimentar os corpos para qualquer posição da área de desenho e, até mesmo, promover um ajuste fino de sua posição *pixel* por *pixel.* A aparência de um corpo pode ser modificada alterando-se o seu tamanho. Atuando sobre os marcadores de seleção laterais centrais da direita e da esquerda (figura 6.5) pode-se alterar a largura do corpo. De forma similar, atuando sobre os marcadores de seleção centrais inferior e superior pode-se alterar a altura do corpo. Os marcadores situados nos vértices permitem que se altere, ao mesmo tempo, a largura e a altura do corpo. Pressionando-se a tecla CTRL enquanto se atua sobre um de seus marcadores de seleção situados nos vértices, pode-se alterar as dimensões do corpo mantendo-se a proporção entre a altura e largura do corpo.

A forma dos arcos, polígonos e linhas angulares pode ser ajustada, após terem sido desenhados, por meio de um comando "Acertar o desenho". Não se trata, simplesmente, de alterar o tamanho destas figuras o que, também, é possível. Trata-se de arrastar marcadores de acerto de desenho, colocados estrategicamente sobre um corpo, para alterar a sua forma básica. Por exemplo, quando se usa a operação de acertar o desenho para um polígono de 6 vértices, pode-se mover os vértices de cada ângulo. Isto significa

que é possível sobrepor dois vértices deste, fazendo com que o polígono passe a ter 5 vértices.

Para o NEWTON, o desenho de um simples retângulo representa um corpo. No entanto, os corpos podem ser agrupados para formarem corpos mais complexos. Por exemplo, um corpo cujo desenho é um retângulo pode ser agrupado a outro cujo desenho é um círculo. O NEWTON trata este conjunto como um novo corpo. Logo, tudo que se pode fazer com os corpos individualmente representados, também se pode fazer ao agrupamento deles. Assim, por exemplo, o agrupamento pode ser movimentado, ter suas dimensões alteradas ou, como veremos mais tarde, pode-se associar comportamentos mecânicos não só aos corpos que formam o grupo mas, também, ao corpo formado pelo agrupamento. Em princípio, pode-se construir grupos de grupos indefinidamente. Na prática, fica-se restrito aos recursos disponíveis no computador. Tal como se pode agrupar os corpos, pode-se desagrupá-los. O desagrupamento de um corpo remove o seu comportamento mecânico. Por segurança, caso um agrupamento possua comportamento mecânico definido, antes de fazer a referida remoção, o NEWTON apresenta uma caixa de diálogo que solicita do usuário a confirmação do desagrupamento.

Corpos podem ser alinhados precisamente com o auxílio de grade, réguas e do comando "Alinhar". A grade é um conjunto de pontos que mostram a intersecção de linhas verticais e horizontais e que podem ser exibidos na área de desenho dos corpos de um modelo aberto. Similarmente, réguas horizontal e vertical podem ser exibidas na referida área. O comando "Alinhar" permite que se alinhe os corpos um em relação aos outros e possui diversas opções de alinhamento (ao centro na horizontal, ao centro na vertical, pelo lado esquerdo, pelo lado direito, pelo topo, por baixo). A grade permite o alinhamento dos corpos através do *mouse* de uma forma similar ao alinhamento de desenhos à mão sobre um papel milimetrado (figura 6.6). Pode-se ajustar a malha da grade fazendo com que os seus pontos se apresentem mais perto ou mais longe uns dos outros. Tem-se a opção de permitir ou não o desenho entre duas linhas da grade. Entretanto, quando se habilita esta opção os corpos já representados não são afetados até que sejam movimentados ou até que sofram alterações em suas dimensões.

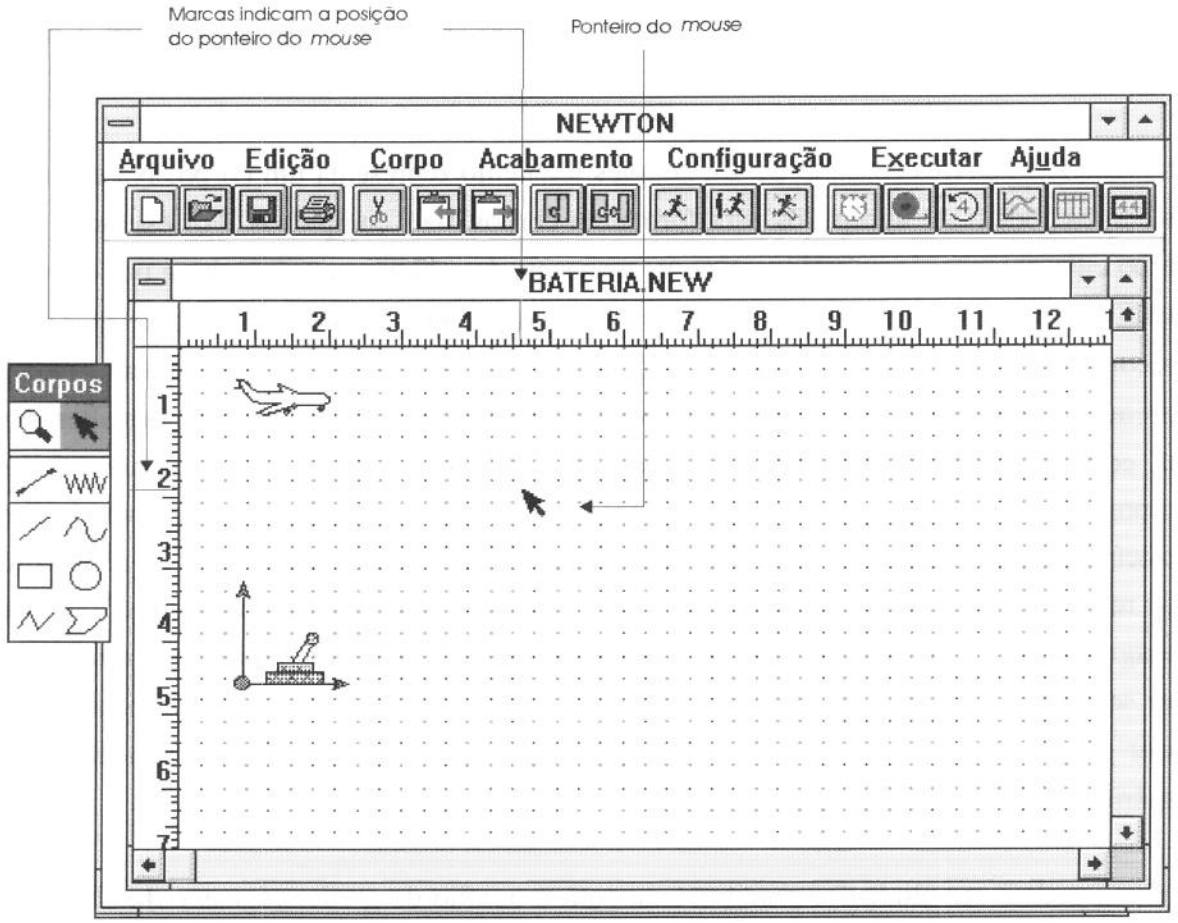


Fig. 6.6: Grade e réguas exibidas na área de desenho de corpos de um modelo

As réguas podem ser usadas para posicionar e medir os corpos (figura 6.6). O NEWTON permite inserir duas réguas na área de desenho de corpos do modelo correntemente selecionado: uma horizontal e outra vertical. Uma marca em cada régua indica a posição do ponteiro do *mouse*. A escala de cada régua é ajustada para cada nível de zoom selecionado. A escala das réguas é hipotética e não guarda, necessariamente, nenhuma relação com possíveis unidades utilizadas na descrição do modelo.

Usando-se a ferramenta lupa existente na palheta Corpos pode-se facilitar o alinhamento e, também, o desenho dos corpos por se produzir ampliações do desenho. Uma vez que o desenho esteja ampliado, pode-se trabalhar os detalhes deste com qualquer ferramenta da palheta Corpos. Pode-se conseguir ampliações de até dezesseis vezes o tamanho normal.

O NEWTON dispõe de uma grande variedade de cores para serem aplicadas aos corpos representados. A todo corpo estão associados dois tipos de cores: a cor de preenchimento (que colore o interior do corpo) e a cor de contorno (que colore as linhas que formam o contorno do corpo). Duas palhetas podem ser utilizadas para colorir um corpo: a palheta **Cores** e a palheta **Espectro de cores** (figura 6.7).

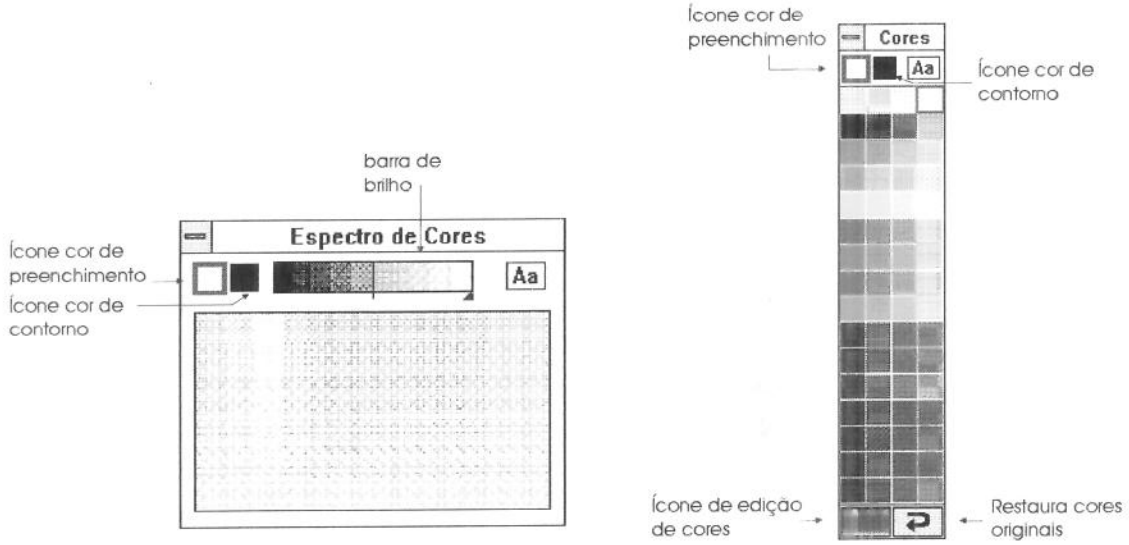


Fig. 6.7: As palhetas cores e espectro de cores

O NEWTON trabalha com o conceito de cor corrente para preenchimento e contorno. Um corpo sempre é desenhado tendo como cor de preenchimento a cor corrente para preenchimento e como cor de contorno a cor corrente para contorno. Por *default* a cor corrente para preenchimento é a branca e a de contorno é a preta. É possível alterar a cor corrente para preenchimento e contorno. Pode-se, também, alterar a cor de preenchimento ou de contorno de corpos já desenhados. O NEWTON permite, também, que o modelador configure as cores existentes na palheta Cores a partir da palheta Espectro de cores.

Quando se desenha corpos tais como um retângulo ou um círculo o NEWTON os preenche automaticamente com o padrão correntemente selecionado. Isto é inteiramente análogo ao que acontece em relação às cores correntes de preenchimento e de contorno. Pode-se especificar um novo padrão de preenchimento para um corpo após este ter sido

desenhado ou pode-se alterar o padrão corrente de preenchimento através da palheta de padrões (figura 6.8). Também, a espessura das linhas que formam o contorno dos corpos podem ser alteradas ou, pode-se alterar o padrão corrente de espessura de linha usando-se a palheta de linhas (figura 6.8).

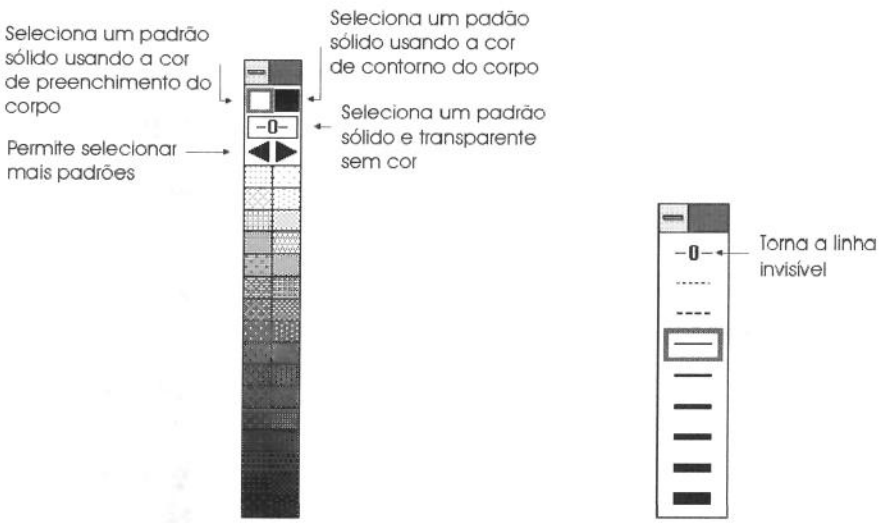


Fig. 6.8: (a) A palheta de padrões de preenchimento e (b) a palheta de espessura e estilo de linhas

Considerando que a representação no NEWTON é bidimensional a profundidade dos corpos não é representada, porém, o nível em que estes corpos são desenhados afeta a percepção visual uma vez que uns sobrepõem os outros (figura 6.9).

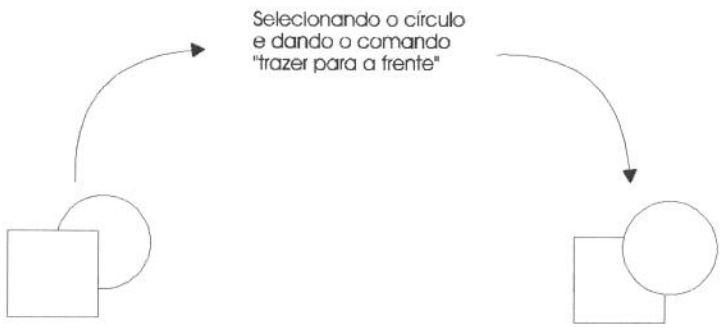


Fig. 6.9: Os corpos são apresentados em diferentes níveis

Toda vez que um novo corpo é criado o NEWTON o coloca na frente dos demais. Após ser criado pode-se alterar o nível em que um corpo se encontra através dos comandos "Trazer para a frente", "Enviar ao fundo", "Um nível à frente" e "Um nível ao fundo" os quais alteram o nível em que um corpo se encontra mostrando-o, respectivamente, na frente de todos os outros, atrás de todos os outros, um nível a frente do nível atual e um nível atrás do atual.

A área de transferência (*clipboard*) é um recurso oferecido pelo Windows que permite que dados (entre eles, desenho de corpos) sejam armazenados temporariamente. Isto possibilita, por exemplo, que aplicações distintas troquem dados entre si ou que, uma mesma aplicação possa armazenar dados na área de transferência, com a intenção subseqüente de copiá-los de volta: o resultado é um eficiente meio de se criar múltiplas cópias de um dado. Aplicativos Windows provêm sempre um mecanismo padrão para se operar com a área de transferência segundo 3 operações básicas. Uma operação de cópia na qual um aplicativo pode copiar dados para a área de transferência sem, contudo, removê-lo; uma operação, normalmente caracterizada como "Cortar", que remove o dado do aplicativo e o armazena na área de transferência. E, por fim, uma operação geralmente denominada "Colar" que permite que o dado seja copiado da área de transferência para o aplicativo. O NEWTON apresenta estas três operações básicas através de três comandos denominados "Copiar", "Cortar" e "Colar". Como um corpo no NEWTON possui uma aparência (o seu desenho) e pode possuir um comportamento mecânico, quando copia-se, corta-se ou cola-se um corpo, tanto o seu desenho quanto o seu comportamento mecânico serão, respectivamente, copiados, cortados ou colados. Isto cria um mecanismo fácil para se obter modelos cada vez mais complexos a partir de modelos mais simples que já estejam prontos.

Com o objetivo de facilitar a tarefa de desenvolver os modelos, o NEWTON possui uma biblioteca que contém desenhos de corpos comuns e que pode ser acionada através do comando "Biblioteca de corpos".

6.4 - DIAGRAMAS DE COMPORTAMENTO MECÂNICO

Um corpo possui uma aparência e um comportamento mecânico, ainda que o seu comportamento mecânico seja ficar absolutamente parado. Como vimos na seção anterior, o NEWTON permite que se modele a aparência dos corpos através de desenhos em duas dimensões. Já o comportamento mecânico de um corpo será definido através de diagramas criados a partir dos elementos apresentados na figura 6.1. Mas, qual o significado destes elementos? Como o próprio nome sugere, um elemento do tipo função algébrica presta-se para modelar uma função algébrica. Elementos deste tipo possuem um nome, um conector de entrada, um conector de saída (observe que ele se distingue do conector de entrada por se apresentar desenhado com uma linha mais grossa) e uma área, em seu interior, onde uma expressão algébrica pode ser escrita (figura 6.10-a). Pode-se conectar um elemento do tipo função algébrica a outro para construir funções algébricas mais complexas (figura 6.10-b). Esta figura apresenta três elementos do tipo função algébrica: a função algébrica constante de nome "força" cujo valor é igual a 5000, a função algébrica constante de nome "massa" cujo valor é igual a 100 e a função algébrica de nome "aceleração" cujo valor é a razão entre as quantidades presentes nos elementos "força" e "massa". Como deve ser interpretado este diagrama? O elemento "força" avalia a função presente no seu interior e coloca o resultado (5000) em seu conector de saída. De maneira similar, o elemento "massa" avalia a função presente no seu interior e coloca o resultado (100) em seu conector de saída. Estes valores são conduzidos à entrada do elemento "aceleração", visto que existem conexões entre os conectores de saída dos elementos "força" e "massa" e o conector de entrada do elemento "aceleração". O elemento "aceleração", por sua vez, avalia a função algébrica (força/massa) contida em seu interior e coloca o resultado (50) em seu conector de saída. Sucessivamente, o resultado do elemento "aceleração" poderia ser conduzido a outro elemento caso o seu conector de saída estivesse ligado ao conector de entrada de um outro elemento.

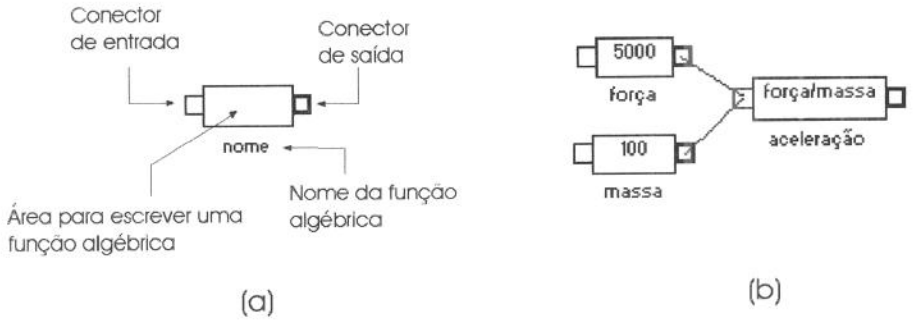


Fig. 6.10: O elemento função algébrica

Um elemento do tipo quantidade acumulada presta-se para acumular valores (positivos ou negativos) à medida que o tempo transcorre. Fazendo uma analogia hidráulica, elementos deste tipo atuam como uma caixa-d'água. Isto é, podem aumentar o volume de líqüido contido em seu interior à medida que recebem água. E podem diminuir o referido volume à medida que a água vai sendo retirada do seu interior. Contudo, devemos parar a analogia por aqui. Uma caixa-d'água jamais registra um volume negativo de água e elementos do tipo quantidade acumulada podem, em um dado momento, estar registrando níveis negativos. Além disso, uma caixa d'água comporta um volume máximo de água e elementos do tipo quantidade acumulada podem, em princípio, estar registrando valores tão grandes quanto se possa imaginar[1]. Elementos deste tipo possuem um nome, um conector de entrada, um conector de saída (novamente aqui, ele se distingue do conector de entrada por apresentar-se desenhado com uma linha mais grossa) e uma área, abaixo da palavra "INÍCIO", onde uma função algébrica pode ser escrita para definir um valor inicial para a quantidade acumulada (figura 6.11-a). Um elemento que se conecta à entrada de uma quantidade acumulada atua como uma taxa de variação para esta. Ou seja, a cada unidade de tempo, a quantidade acumulada sofre acúmulos correspondentes ao valor do elemento conectado na sua entrada. A figura 6.11-b mostra um elemento do tipo função algébrica de nome "velocidade" conectado à entrada de um elemento do tipo quantidade acumulada de nome "x" cujo valor inicial é 0 (zero). Isto significa que, para cada unidade de tempo, o valor da quantidade acumulada "x" sofre acréscimos de valor igual ao valor do elemento conectado à sua entrada, ou seja, 150. Em outras palavras, no início da contagem do tempo (tempo = 0) o valor do elemento "x" é 0 e à medida que o tempo transcorre o elemento "x" assume uma infinidade de valores crescentes a partir de 0 até chegar a ter o valor 150 quando a primeira unidade de tempo for atingida (tempo =

[1] É bem verdade que tais elementos estarão, também, limitados à capacidade de representação do computador e, na prática, quando implementados, possuem limites negativo e positivo.

1). Em seguida, o elemento "x" assume uma infinidade de valores crescentes a partir de 150 até chegar a ter o valor 300 quando a segunda unidade de tempo for atingida (tempo = 2). E assim, sucessivamente, o elemento "x" vai acumulando valores.

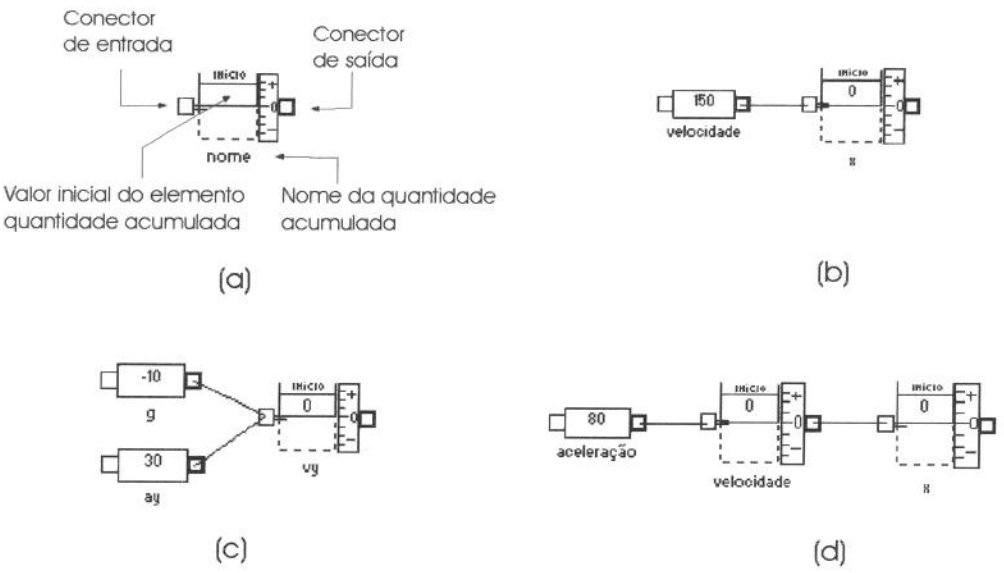


Fig. 6.11: O elemento quantidade acumulada

Mais que um elemento pode ser conectado à entrada de um elemento do tipo quantidade acumulada. Quando isto ocorre, a taxa de variação a qual fica submetido o elemento do tipo quantidade acumulada é igual à somatória dos valores de todos os elementos que se conectam à sua entrada. Por exemplo, o elemento de nome "vy", apresentado na figura 6.11-c, está submetido a uma taxa de variação de valor igual a 20. Um elemento do tipo quantidade acumulada pode atuar como uma taxa de variação para outro elemento do tipo quantidade acumulada. A figura 6.11-d mostra que o elemento de nome "aceleração" atua como uma taxa de variação para o elemento de nome "velocidade" que, por sua vez, é uma taxa de variação para o elemento de nome "x".

Até o momento, sabemos o significado de três (função algébrica, conexão e quantidade acumulada) dos quatro elementos apresentados na figura 6.1. Com estes três elementos é possível modelar uma grande variedade de comportamentos mecânicos. Utilizemos, pois, estes elementos para exemplificar a modelagem de alguns comportamentos mecânicos.

Comportamento mecânico 1- Um comportamento mecânico freqüentemente encontrado em aplicações práticas é o do movimento retilíneo de um corpo, cuja posição varia uniformemente ao longo do tempo. Ou seja, a posição do corpo (digamos, x) varia em quantidades iguais para quantidades iguais de tempo. Isto é o mesmo que dizer que existe uma taxa de variação de valor constante para a posição do corpo. Para modelar este comportamento mecânico podemos usar um elemento do tipo função algébrica de valor constante, por exemplo igual a 150, atuando como taxa de variação para a posição do corpo. A posição do corpo, neste caso, será modelada como um elemento do tipo quantidade acumulada. Este pequeno modelo é apresentado na figura 6.12.

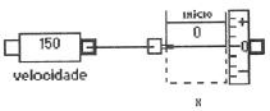


Fig. 6.12: Modelo para o comportamento mecânico de um corpo sob movimento retilíneo uniformemente variado

Chamamos a atenção para dois aspectos: (1) Apenas, para complemento do modelo, supusemos que a posição (x) inicial do corpo é 0. É claro que a posição inicial do corpo poderia ser 30, 70 ou qualquer outro valor. O que se mudaria, no modelo da figura 6.12, é o valor inserido dentro do elemento de nome "x"; (2) atribuímos ao elemento do tipo função algébrica, que atua como taxa de variação da posição, o nome "velocidade". Poderia ser qualquer outro nome. Preferimos adotar o nome clássico referenciado pelos livros de Mecânica. Entretanto, isto não quer dizer que o modelador não familiarizado com a Mecânica expressa na literatura não possa modelar comportamentos mecânicos da forma como estamos propondo. Modeladores sem conhecimento formal de Mecânica poderão usar o formalismo e o ambiente que propomos para aprendê-la. Nos modelos apresentados ao longo deste trabalho, apenas por questão de facilidade de comunicação com os leitores que possuem conhecimento formal de Mecânica, usaremos a nomenclatura consagrada pela literatura.

Comportamento mecânico 2 - Outro comportamento mecânico freqüentemente encontrado nas aplicações práticas é o do movimento retilíneo de um corpo cuja taxa de variação da posição varia uniformemente no tempo. Ou seja, a posição (digamos, x) é uma

quantidade que varia ao longo do tempo e será modelada, portanto, através de um elemento do tipo quantidade acumulada. Também, a taxa de variação da posição é uma grandeza que varia ao longo do tempo e será modelada através de outro elemento do tipo quantidade acumulada. Porém, a taxa de variação da taxa de variação da posição deve ser uma constante e, portanto, pode ser modelado através de um elemento do tipo função algébrica. A figura 6.13 mostra um modelo para este comportamento.

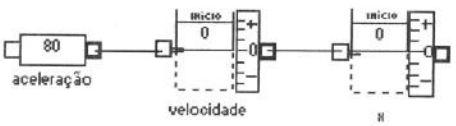


Fig. 6.13: Modelo para o comportamento mecânico de um corpo sob movimento retilíneo uniformemente acelerado

Os corpos, em geral, podem apresentar comportamentos mecânicos distintos dependendo das condições em que estão submetidos. Por exemplo, um automóvel pode movimentar-se sob aceleração constante durante um certo intervalo de tempo e, depois, viajar sob uma velocidade constante. Dois comportamentos distintos: primeiro, o automóvel movendo-se sob aceleração constante e segundo, o automóvel viajando sob velocidade constante. Para se modelar este conjunto de comportamentos deve-se lançar mão de elementos do tipo duração do comportamento. Elementos do tipo duração do comportamento prestam-se para modelar as condições nas quais um determinado comportamento mecânico se manifesta. Estes elementos apresentam campos para descrição das condições de início e fim do comportamento definido em seu interior (figura 6.14-a). Assim, poderíamos usar dois elementos do tipo duração do comportamento para estabelecer um possível modelo para o comportamento do automóvel citado acima (figura 6.14-b). O modelo desta figura estabelece que o automóvel parte de uma posição (x) igual a zero com velocidade igual a 0, viaja sob aceleração constante de valor igual a 150 no intervalo de tempo compreendido entre 0 e 5 e, após isto, continua movimentando-se sob a velocidade constante adquirida durante o intervalo de tempo em que sofria aceleração. Dois comentários: (1) deve-se observar o valor do elemento de nome "velocidade2". Ele é igual ao valor do elemento "velocidade1". Isto quer dizer que, quando o tempo for maior ou igual a 5 o automóvel se movimentará sob a velocidade adquirida no intervalo de tempo entre 0 e 5; (2) não descrever a condição de fim de um elemento do tipo

duração do comportamento significa dizer que o comportamento não tem final estabelecido. Um dos elementos do tipo duração do comportamento apresentados no modelo da figura 6.14-b apresenta tal situação: possui uma condição para início do comportamento (tempo > = 5) e não possui uma condição para encerramento do mesmo. De forma similar, elementos do tipo duração de comportamento podem não possuir condição de início sendo que, neste caso, deve-se entender que o comportamento se inicia no primeiro instante (tempo = 0). Não possuir ambas condições é o mesmo que dizer que o comportamento inicia-se no primeiro instante (tempo = 0) e que não possui final estabelecido. Em outras palavras, a presença ou a ausência em um modelo de elementos do tipo duração de comportamento nestas condições, não acrescenta absolutamente nada ao modelo.

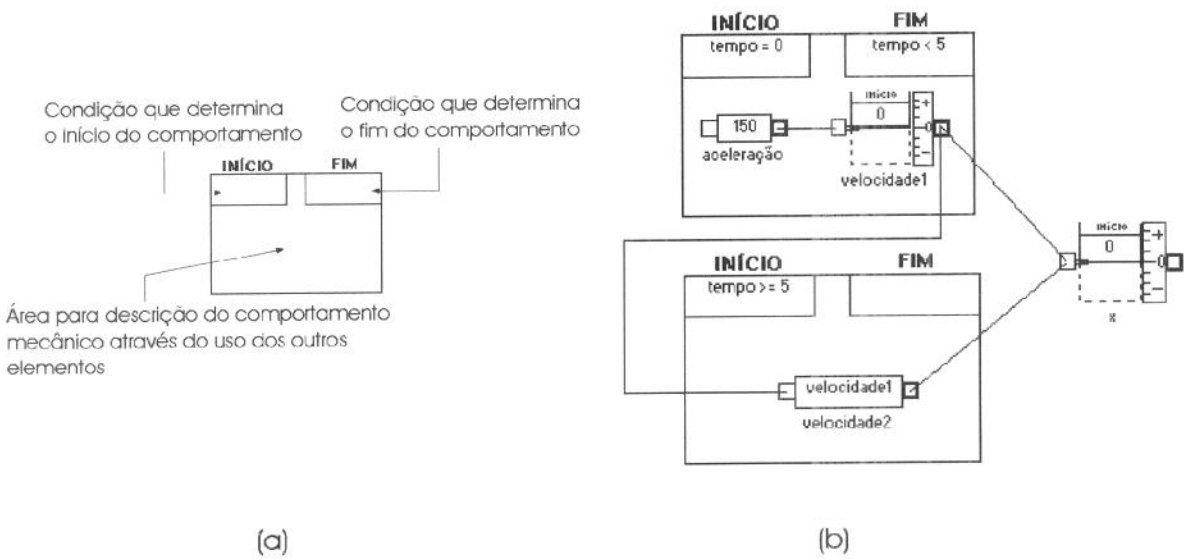


Fig. 6.14: O elemento duração de comportamento

As condições de início e fim de comportamento não precisam envolver, apenas, relações com o tempo. Elas podem envolver quaisquer grandezas utilizadas na descrição do modelo. Para um determinado modelo, por exemplo, a condição de início de um comportamento poderia ser **velocidade = 0** e a condição fim poderia ser x **> 300.** Não usar elementos do tipo duração de comportamento para descrever o comportamento mecânico de um corpo, é o mesmo que dizer que o corpo possui um único comportamento ao longo de toda execução do modelo. Os elementos do tipo duração de

comportamento não precisam, necessariamente, ser mutuamente exclusivos no que diz respeito ao intervalo de tempo em que os comportamentos descritos em seu interior são executados. Ou seja, dois ou mais comportamentos mecânicos, descritos ou não no interior de elementos do tipo duração de comportamento, podem estar ativos ao mesmo tempo. O comportamento final do corpo será a composição de cada um dos comportamentos individuais.

Quando o comportamento mecânico de um corpo depende do comportamento mecânico de outro corpo, um novo elemento, além daqueles apresentados na figura 6.1, precisa ser utilizado. Este elemento, o qual denominamos relação de dependência, é usado para descrever a inter-relação entre as grandezas utilizadas para definir o comportamento individual de cada corpo. Ao contrário dos outros elementos, um elemento do tipo relação de dependência não possui um desenho, mas pode ser definido pela seguinte estrutura simbólica descrita pelo diagrama sintático da figura 6.15.

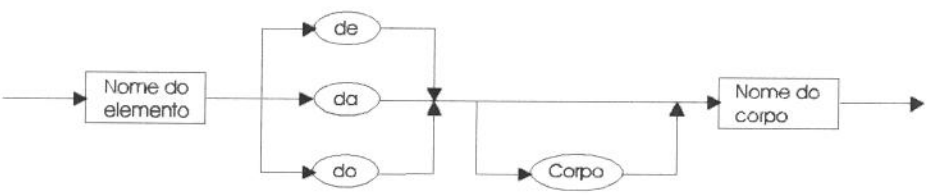
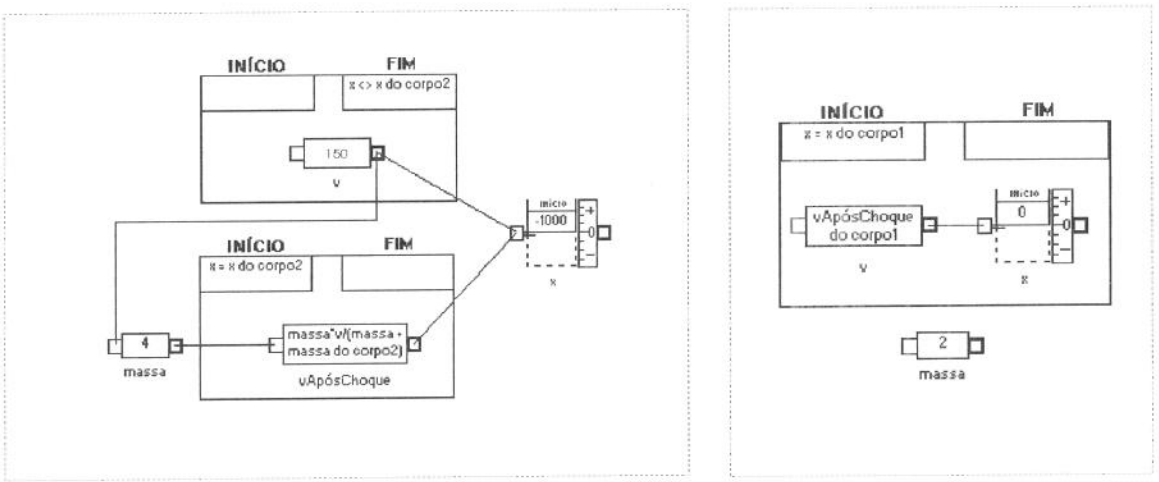


Fig. 6.15: Estrutura simbólica usada para definir a relação de dependência entre os comportamentos mecânicos dos corpos

Utilizando-se a estrutura sintática da figura 6.15 pode-se dizer que são referências válidas: velocidade do automóvel, aceleração do corpo A, x da estaca etc.. A figura 6.16 nos mostra como se pode utilizar elementos do tipo relação de dependência. Ela modela o choque plástico[2] entre dois corpos, um parado (corpo2) e outro movimentando-se sob velocidade constante (corpo1). Mostraremos, a seguir, a utilização dos elementos do tipo duração de comportamento. O que podemos observar? O modelo para o comportamento mecânico do corpo1 apresenta referências a elementos do corpo2 e elementos do corpo2 apresentam referências ao corpo1. Além disso, observamos que as relações de

[2] Um choque entre dois corpos rígidos é dito ser plástico se as velocidades dos corpos tornarem-se iguais após o choque. Isto é, os corpos passam a se movimentar juntos após o choque.

dependências estão expressas tanto em elementos do tipo duração de comportamento quanto em elemento do tipo função algébrica. Como podemos interpretar, por exemplo, a condição **x = x do corpo2** que aparece em um dos elementos usados na definição do comportamento mecânico do corpo1? Por estar escrito no comportamento mecânico do corpo1 esta expressão significa: valor do elemento de nome "x" do corpo1 igual ao valor do elemento "x" do corpo2. Da mesma maneira, a condição **x = x do corpo1** que aparece na definição do comportamento do corpo2 deve ser interpretada como: valor do elemento "x" do corpo2 (uma vez que está definido no comportamento mecânico do corpo2) igual ao valor do elemento "x" do corpo1.



Modelo para o comportamento mecânico do corpo1

Modelo para o comportamento mecânico do corpo2

Fig. 6.16: Um modelo para o choque plástico entre dois corpos

Conhecemos, agora, todos os elementos para a definição de diagramas do comportamento mecânico. Resta-nos saber como o NEWTON possibilita a edição destes diagramas. Isto é o que será mostrado na próxima seção.

6.5 - EDITANDO DIAGRAMAS DE COMPORTAMENTO MECÂNICO

Para todo corpo desenhado em algum modelo criado pelo NEWTON pode-se ou não definir um diagrama de comportamento mecânico. Um corpo sem comportamento

mecânico definido será mostrado em repouso absoluto quando da execução modelo. Tais corpos em geral compõem o cenário do modelo e, muitas das vezes, não guardam uma relação direta com fenômeno mecânico modelado; apenas figuram. Em contrapartida, corpos com comportamento mecânico definido podem mover-se no momento da execução do modelo. É claro que o movimento ou o repouso destes corpos será função do comportamento mecânico definido. Pode-se editar o comportamento mecânico de um corpo dando-se um duplo *click* neste. Então, o NEWTON abre uma caixa de diálogo que permite definir o nome do corpo e, também, abrir o editor de diagramas de comportamento mecânico através de um botão apropriado. A janela do editor de diagramas de comportamento mecânico tem a aparência mostrada na figura 6.17.

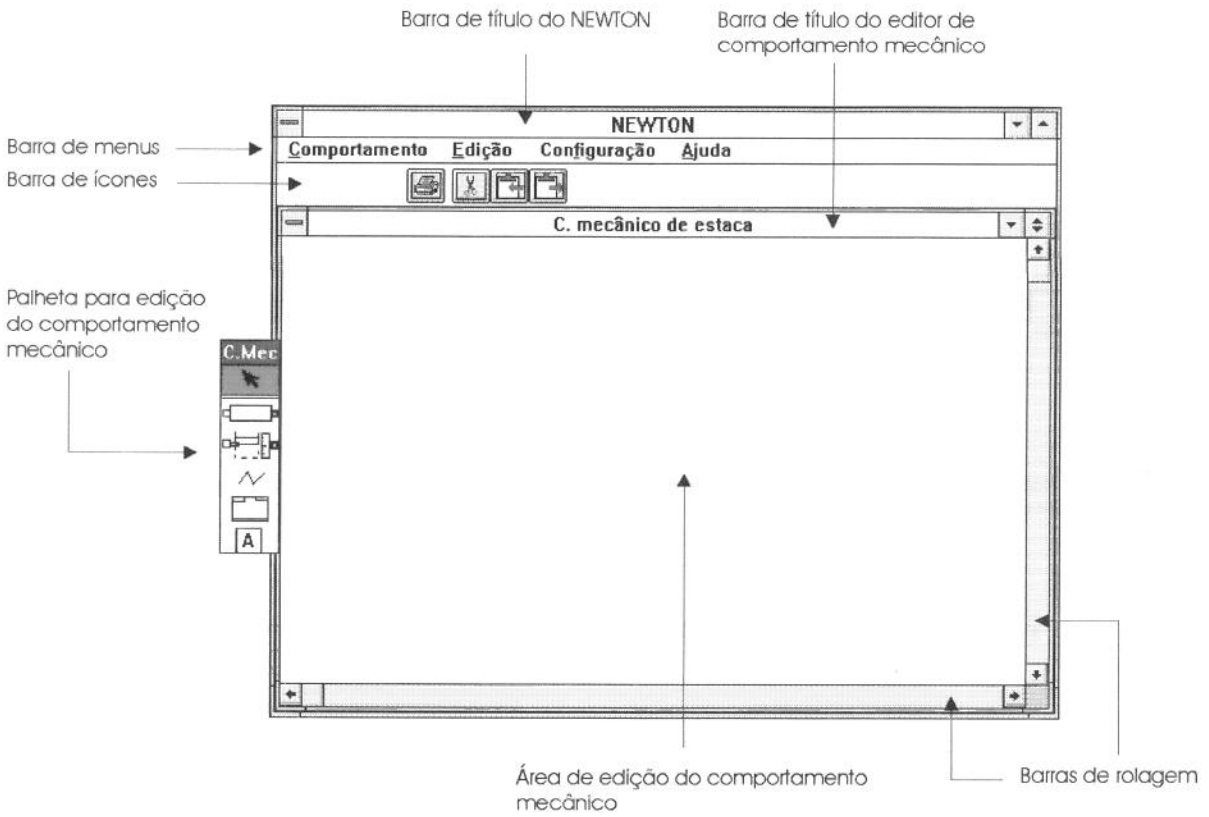


Fig. 6.17: A janela de edição de comportamento mecânico de um corpo. No caso o corpo cujo comportamento mecânico está sendo editado chama-se "estaca"

Observamos na figura 6.17 que a janela do editor de comportamento mecânico está contida na janela do NEWTON e que apresenta, em sua barra de título, o nome do corpo

cujo comportamento mecânico pode ser definido, ou seja, no exemplo, "estaca". Barras de rolagem permitem que a área para descrição do comportamento mecânico não se restrinja à área apresentada na tela. Os comandos podem ser acionados por qualquer das quatro formas mencionadas na seção 6.2. Uma palheta pode ser utilizada na edição dos elementos gráficos do diagrama de comportamento mecânico bem como, para comodidade do usuário, pode ser arrastada para qualquer ponto da tela. Trata-se da palheta para edição de diagramas de comportamento mecânico apresentada, em detalhes na figura 6.18, e referenciada, neste texto, apenas por palheta de comportamento mecânico.

Fig. 6.18: A palheta para edição de comportamento mecânico

A palheta de comportamento mecânico contém ícones para edição dos elementos gráficos que são utilizados nos diagramas de comportamento mecânico os quais foram amplamente discutidos na seção anterior: função algébrica, quantidade acumulada, conexão e duração do comportamento. A palheta contém, também, um ícone para seleção dos elementos desenhados (seta de seleção) e um ícone destinado à criação de comentários. Comentários são textos livres, escritos em linguagem natural, que podem ser inseridos em qualquer ponto do diagrama de comportamento mecânico. Em geral, podem ser utilizados pelo modelador para expor as características gerais do modelo, descrever os princípios sobre os quais ele se fundamenta, expressar as suas restrições ou, simplesmente, mostrar o que determinado trecho de diagrama faz. Usa-se a palheta de comportamento mecânico exatamente da mesma maneira como se usa a palheta para edição de corpos. Ou seja, seleciona-se um elemento na palheta (*click* no elemento da palheta) e este pode ser desenhado na área dedicada à edição do comportamento mecânico apertando e mantendo pressionado o botão esquerdo do *mouse*, à medida em este é arrastado em uma direção

aproximadamente diagonal. O NEWTON impõe, evidentemente, uma limitação ao desenho de uma conexão: ela deve iniciar no conector de saída de um elemento e terminar no conector de entrada de outro elemento. Fora isto, o procedimento para se desenhar estes elementos é semelhante ao procedimento para desenhar corpos. Assim, por exemplo, pode-se copiar, cortar, colar, movimentar ou alterar as dimensões dos elementos desenhados, individualmente ou coletivamente, antes ou depois de terem sido conectados uns aos outros. Pode-se editar o conteúdo de um item de algum elemento (por exemplo, a condição de início de um elemento do tipo duração do comportamento) dando-se um duplo *click* nele. Através do uso das palhetas de cores ou de espectro de cores pode-se colorir certos trechos do diagrama com o objetivo de destacá-los dos demais. Para fins de execução, o NEWTON permanece indiferente à cor atribuída a um determinado elemento. Entretanto, o uso de cores pode constituir um excelente recurso para separar partes conceitualmente distintas em diagramas grandes ou complexos. Referência aos comandos que podem ser utilizados no desenho de diagramas de comportamento mecânico pode ser encontrada no apêndice A.

As expressões algébricas que devem ser usadas no interior de alguns elementos podem usar um vasto conjunto de operadores e funções predefinidas tais como: "raizQuadrada ()" que calcula a raiz quadrada de um número, "sen ()" que calcula o seno de um número dado em radianos e "arredonda ()" que arredonda um número real. Referência completa dos operadores disponíveis, suas prioridades, funções predefinidas, tipo de expressão esperada (numérica ou booleana) por elementos do tipo quantidades acumuladas, função algébrica e duração de comportamento pode ser encontrada no apêndice A.

O NEWTON cuida para que não se estabeleçam construções diagramáticas incorretas ou expressões sintaticamente incorretas. Ao mesmo tempo, se preocupa em informar a incorreção em um tempo tal que não prejudique a produtividade do trabalho do modelador. Assim, algumas construções incorretas são coibidas em tempo de desenho. Outras, são detectadas apenas quando se usa o comando "Testar sintaxe" ou quando se quer deixar o editor de comportamento mecânico fazendo-se uso do comando "Sair alterando". Há até aquelas que só são detectadas em tempo de compilação do modelo. São exemplos de construções incorretas coibidas em tempo de desenho: ligar um conector de saída de um elemento ao seu conector de entrada; ligar duas vezes os mesmos conectores dos mesmos elementos; usar para nome de um elemento uma palavra reservada; escrever uma expressão cuja sintaxe é incorreta. São exemplos de construções incorretas detectadas pelo comando "Testar sintaxe" ou quando se quer deixar o editor de

comportamento mecânico fazendo-se uso do comando "Sair alterando": deixar de ligar ao conector de entrada de uma função algébrica algum dos elementos mencionados na expressão descrita em seu interior; mencionar na expressão descrita no interior de uma função algébrica um elemento não ligado ao seu conector de entrada; deixar de preencher o conteúdo de algum elemento do tipo função algébrica ou quantidade acumulada. São exemplos de construções incorretas detectadas em tempo de compilação: mencionar nomes de corpos que não existem; criar referências circulares em que dois elementos em corpos distintos referenciam-se mutuamente. Tais referências circulares podem envolver, também, três ou mais corpos.

Pode-se sair do editor de comportamento mecânico e voltar à área de edição de corpos de duas formas: através dos comandos "Sair alterando" ou "Sair cancelando" do menu "Comportamento". O comando "Sair alterando" permite que as trocas feitas no diagrama de comportamento mecânico sejam consolidadas, enquanto que o comando "Sair cancelando" não as consolida. O NEWTON apontará um erro de sintaxe ou de construção diagramática incorreta caso o diagrama editado contenha erros e se esteja usando o comando "Sair alterando".

Antes de encerrarmos esta seção precisamos definir o que é propriedade de um corpo para o NEWTON. Até agora, dissemos que os corpos possuem uma aparência e um comportamento mecânico. Acrescentamos, neste momento, que os corpos possuem propriedades. Define-se uma propriedade para um corpo quando se cria, em seu comportamento mecânico, elementos que podem conter valores, isto é, elementos do tipo função algébrica ou quantidade acumulada. Assim, dizemos que um corpo **A** possui uma propriedade chamada **aceleração** se ele contiver, em seu diagrama de comportamento mecânico, um elemento do tipo função algébrica ou quantidade acumulada cujo nome é **aceleração**. Esta terminologia nos será útil na seção seguinte.

Uma vez desenhados todos os corpos do modelo e tendo sido definidos os seus comportamentos mecânicos, o modelo está pronto para ser executado. É deste assunto que trata a próxima seção.

6.6 - EXECUTANDO MODELOS E INSTALANDO OS INSTRUMENTOS DE *FEEDBACK*

A execução de um modelo cria uma oportunidade para o modelador refletir sobre a validade do modelo criado. Qualquer modelo pode ser executado, ainda que apenas, mentalmente. A execução de um modelo pelo computador, além de ser precisa e rápida, tem a vantagem extra de liberar o tempo do modelador para as atividades inerentemente relacionadas à modelagem, isto é, criar e refletir sobre o modelo criado. O ambiente para modelagem NEWTON permite que se execute o modelo criado. No entanto, para que possamos entender como o NEWTON executa um modelo e, em especial, anima os corpos de acordo com os seus comportamentos mecânicos durante a execução, precisamos entender como ele reconhece e estabelece a posição dos corpos definidos em um modelo.

Todo modelo contém um par de eixos cartesianos que determina a origem do sistema de medidas (posição de coordenadas 0,0). Um corpo, em geral, é formado por um grande número de pontos que situam-se em distintas posições medidas a partir da origem. No entanto, por questão de representação e cálculo o NEWTON necessita definir a posição de um corpo em um ponto[3]. Por motivos análogos, a posição angular[4] de um corpo necessita ser definida previamente. Isto impõe que seja definido previamente, isto é, antes do início da execução do modelo, a posição do possível eixo de rotação do corpo. Dizemos, possível eixo de rotação, por que o modelo pode ou não definir rotação para um determinado corpo. Para demarcar a posição de um corpo e, também, o seu possível eixo de rotação o NEWTON dispõe do elemento gráfico denominado "posição e eixo de rotação" (figura 6.19). Todo corpo desenhado em algum modelo do NEWTON possui um elemento "posição e eixo de rotação" que demarca a sua posição e o seu possível eixo de rotação. Originalmente, isto é, quando o corpo é criado, o NEWTON coloca este elemento em uma posição padrão. O modelador, por conveniência do modelo que está desenvolvendo, pode arrastar este elemento para o ponto que desejar. Informações sobre como exibir e arrastar o elemento "posição e eixo de rotação" de um corpo pode ser encontrada no apêndice A.

---

[3] Não estamos, com isto, afirmando que o NEWTON só se presta à modelagem de fenômenos envolvendo corpos que podem ser tratados como pontos materiais, o que restringiria, do ponto de vista da mecânica a aplicação do NEWTON. O NEWTON permite modelar fenômenos envolvendo corpos tratados como pontos materiais ou como corpos rígidos. Para uma conceituação sobre pontos materiais e corpos rígidos pode-se consultar Beer e Johnston (1980 e 1991).

[4] O NEWTON permite que se represente os corpos através de desenhos em duas dimensões, isto é, no plano. Os possíveis movimentos que um corpo rígido pode sofrer no plano são: translação no plano e rotação em torno de um certo eixo perpendicular ao plano. Pode-se conheçer a posição de um corpo, no plano, conhecendo-se a sua posição em relação a um par de eixos cartesianos (isto permite acompanhar o movimento de translação do corpo) e a sua posição angular em relação ao seu eixo de rotação (isto permite acompanhar o movimento de rotação do corpo).

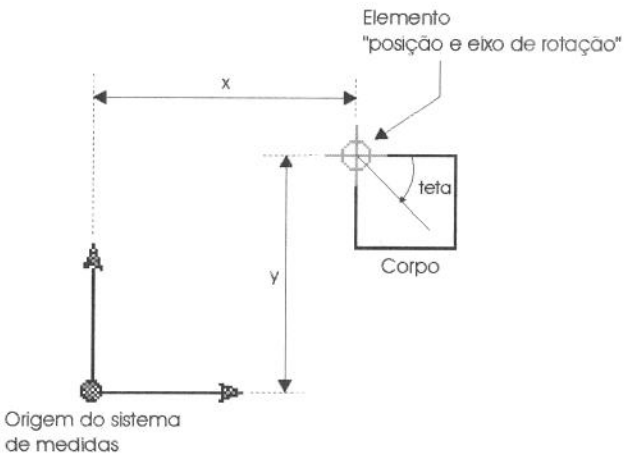


Fig. 6.19: Posição (x, y) e posição angular (teta) de um corpo

Quando um corpo é criado em algum modelo, o NEWTON associa a este corpo três propriedades padrão **x**, **y** e **teta** e, imediatamente, atribui valores iniciais a elas[5]. As propriedades x e y de um corpo representam a abscissa e a ordenada da posição do corpo em relação ao par de eixos cartesianos estabelecidos como origem das medidas. A propriedade teta é uma medida para a posição angular do corpo em relação ao seu eixo de rotação. Dizemos que estas propriedades são padrão porque o NEWTON as reconhece e atribui a cada uma delas um comportamento padrão do corpo. Isto é, se fizermos variar o valor da propriedade x de um corpo, o NEWTON fará variar a posição do corpo ao longo da direção da abscissa. Variando o valor da propriedade y de um corpo, o NEWTON fará variar a posição do corpo ao longo da direção da ordenada. E, variando o valor da propriedade teta de um corpo, o NEWTON irá girar o corpo em torno de seu eixo de rotação. Agora estamos prontos para entender como o NEWTON executa os seus modelos.

A execução de modelos no NEWTON mostra corpos animados de acordo com o comportamento mecânico definido para os mesmos. A animação é possível graças às propriedades padrão **x**, **y e teta** comum a todos os corpos criados neste ambiente. O NEWTON reage, com uma variação na posição de um corpo toda vez que os valores de suas propriedades x ou y são alterados. E reage com uma mudança na posição angular do

[5] O valor inicial das propriedades **x**, **y** e **teta** não é atribuido arbitrariamente pelo NEWTON. Tais valores dependem da geometria do corpo (teta) e do ponto no modelo onde este está sendo criado (x e y).

corpo toda vez que o valor de sua propriedade teta é alterada. A variação do valor das propriedades x, y e teta pode ser conseguida compondo diagramas de comportamento mecânico que contenham elementos do tipo quantidade acumulada ou função algébrica cujos nomes são x, y ou teta. A animação constitui um excelente *feedback* sobre o modelo criado por se assemelhar ao que uma pessoa vê quando observa fenômenos mecânicos reais no mundo. Muito se pode inferir sobre a validade do modelo criado observando, apenas, a animação do comportamento mecânico de seus corpos. Contudo, o modelador tem à sua disposição outros instrumentos de *feedback* (figura 6.20). São eles:

- cronômetro: mede o tempo transcorrido entre dois eventos. O modelador deve expressar a ocorrência dos eventos através de duas condições (expressões booleanas): uma é a condição que define a partida do cronômetro (campo início do cronômetro) e a outra é a condição que determina a parada do cronômetro (campo fim do cronômetro). Consideremos, por exemplo, a medida do tempo em que o automóvel de nome "auto1" leva para atingir o automóvel de nome "auto2" existentes em dado modelo. Supondo ser **x** o nome dos elementos que definem a posição dos automóveis, poderíamos escrever para a condição de partida do cronômetro **tempo = 0** e para a condição de parada do cronômetro **x do auto1 = x do auto2**;
- distanciômetro: mede a distância entre os dois corpos cujos nomes são definidos nos campos corpo1 e corpo2. Pode-se optar entre medir a distância real entre os corpos ou a projeção desta distância na direção da abscissa ou na direção da ordenada;
- conta-giros: mede a posição angular do corpo cujo nome está definido no campo corpo.
- gráficos: um único instrumento deste tipo é capaz de desenhar, simultaneamente, até dois gráficos em cores diferentes, que representam as relações entre os valores de elementos utilizados na descrição do comportamento mecânico dos corpos. Deve-se mencionar os nomes dos elementos nos campos de abscissa e ordenada para os gráficos 1 e 2. Um instrumento do tipo gráfico pode ser escondido enquanto não estiver sendo usado. Isto ajuda a "despoluir" a área de trabalho. Em seu lugar haverá um ícone que permitirá exibi-lo novamente no momento em que for necessário. Caso seja

importante a exibição simultânea de três ou mais gráficos, deve-se usar mais que um instrumento deste tipo;

- tabela: um único instrumento deste tipo é capaz de tabelar, à medida que o tempo transcorre, até quatro valores assumidos pelos elementos usados na definição do comportamento mecânico dos corpos de um modelo. Tal como instrumentos do tipo gráfico, pode-se "despoluir" a área de trabalho escondendo elementos deste tipo quando estes não estiverem sendo usados. Pode-se exibi-los novamente no momento em que for necessário através do uso de um ícone apropriado. Caso seja importante a exibição simultânea de três ou mais tabelas deve-se usar mais que um instrumento deste tipo;

- valor de propriedade: exibe o valor assumido por elementos dos tipos função algébrica ou quantidade acumulada definidos em algum dos diagramas de comportamento mecânico dos corpos do modelo. Para isto deve-se mencionar o nome do elemento no campo mais inferior do instrumento (figura 6.20). Supondo existirem os elementos de nomes **velocidade** e **y** dentro do diagrama de comportamento mecânico de um corpo chamado **auto1**, são exemplos de possíveis preenchimentos do campo inferior do instrumento: **velocidade do auto1**, **y do auto1** etc.

Pode-se obter, qualquer destes instrumentos dando um *click* no respectivo ícone da barra de ícones. Quando da execução do modelo, os instrumentos sinalizarão a todo momento os seus valores ou gráfico.

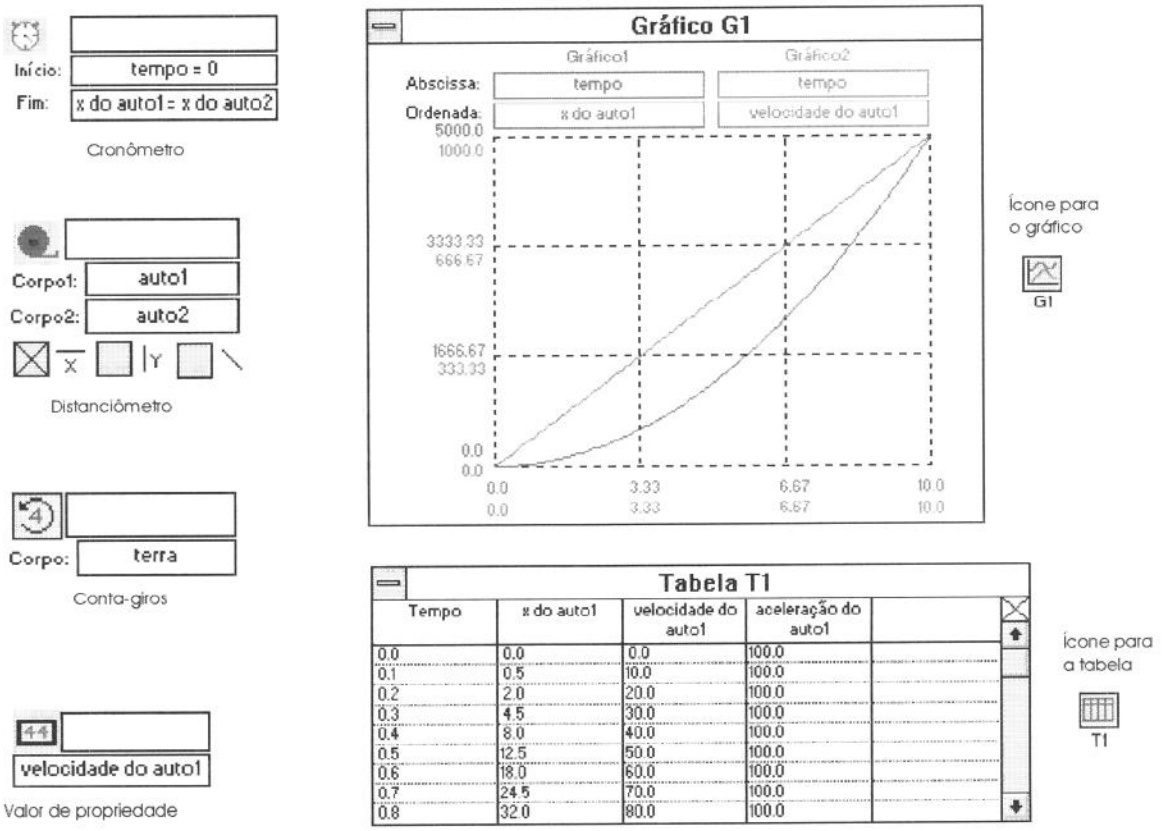


Fig. 6.20: Instrumentos de *feedback*

A execução de um modelo pode ser interrompida, em qualquer momento, de duas maneiras: (1) através de um simples *click* em um ícone disponível na barra de ícones e (2) através de condições de parada previamente estabelecidas como, por exemplo, quando **x do auto1 = x do auto2**. A interrupção da execução do modelo permite que o modelador explore detalhes da execução do modelo, através da observação dos valores registrados pelos instrumentos, gráficos ou disposição geral dos corpos. Pode-se reiniciar uma execução a partir do ponto onde ela foi interrompida através de um ícone disponível na barra de ícones. A execução pode ser cancelada, também, no momento que o modelador desejar. O tempo de início e de fim da execução de um modelo, também, podem ser controlados pelo modelador. Por *default*, o tempo de início da execução é 0 (zero) e o tempo final é 10 (dez) mas estes valores podem ser alterados.

6.7 - UM EXEMPLO DE MODELAGEM COM O NEWTON

Consideremos a tarefa de construir um modelo para uma bola que, movendo em uma superfície plana (sem atrito) com uma velocidade "vx", cai sobre um plano também horizontal e sem atrito. A primeira coisa que temos a fazer é desenhar os corpos envolvidos no fenômeno. A figura 6.21-a mostra o desenho que fizemos: a bola, um talude vertical e duas superfícies planas: uma superior de onde a bola vai cair e uma inferior sobre a qual a bola irá se chocar repetidas vezes. No modelo que estamos estabelecendo tanto o talude vertical quanto as superfícies planas horizontais não possuem comportamento mecânico, ou de outra forma, possuem o comportamento mecânico de ficar absolutamente parados (ou em repouso, usando a terminologia tradicional da Mecânica). Portanto, não precisaremos editar o comportamento mecânico destes corpos. Neste modelo eles servirão apenas como elementos de composição do cenário do fenômeno.

O próximo passo consiste em definir o comportamento mecânico da bola. Dando-se um duplo *click* sobre o desenho da bola o NEWTON permite editar o comportamento mecânico da mesma. Devemos observar que a bola possui dois movimentos:

- um horizontal, sob velocidade constante "vx", velocidade esta adquirida por ela antes da queda;

- outro vertical em queda livre sob efeito da aceleração da gravidade

$$g \cong -10\, m/s^2$$

até se chocar contra a superfície inferior (o sinal negativo da aceleração da gravidade deve-se ao fato de termos colocado o eixo da ordenada da origem do sistema de medidas apontando no sentido inverso ao movimento de descida da bola (figura 6.21-a)). Ao se chocar contra a superfície, a bola adquire velocidade no sentido oposto ao do seu movimento. Contudo, partindo do princípio que a bola não é perfeitamente elástica, a valor da velocidade da bola após o choque contra a superfície será uma fração do valor da velocidade que ela possuía antes do choque. Tudo se passa como se a velocidade da bola após o choque fosse igual a velocidade da bola antes do choque afetada por um certo coeficiente (CR) cujo valor está entre 0 e 1. Este coeficiente "CR" é o coeficiente de restituição da bola. Uma bola perfeitamente elástica possui

coeficiente de restituição igual a 1, isto é, a bola restitui a integralmente a sua velocidade após o choque. Chamando a velocidade da bola antes do choque de "vyDescida" e a velocidade da bola após o choque de "vySubida" podemos escrever que

$$vySubida = -CR*vyDescida.$$

O sinal de menos que afeta o produto se deve ao fato de que o sentido da velocidade se altera após o choque.

A figura 6.21-b mostra o comportamento mecânico que estabelecemos para a bola. Observamos ao topo a representação do movimento horizontal sob velocidade "vx", constante e igual a 50, atuando como taxa de variação para a abscissa "x" da bola. Logo abaixo, está representado o movimento vertical de descida da bola acelerado pela ação da gravidade. Foi utilizado um elemento do tipo duração de comportamento para limitar o comportamento de descida da bola que se inicia nos instantes onde esta possui velocidade vertical igual a zero e que termina nos instantes onde ela atinge a superfície inferior (ou seja, o valor da ordenada "y" da bola é igual a zero). Abaixo no diagrama está representado o movimento de subida da bola, desacelerado por efeito da gravidade. Também foi utilizado um elemento do tipo duração do comportamento para limitar o comportamento de subida da bola que se inicia nos instantes que a bola choca-se contra a superfície inferior (ou seja, o valor da ordenada "y" da bola é igual a zero) e que termina nos instantes onde a velocidade de subida da mesma torna-se zero.

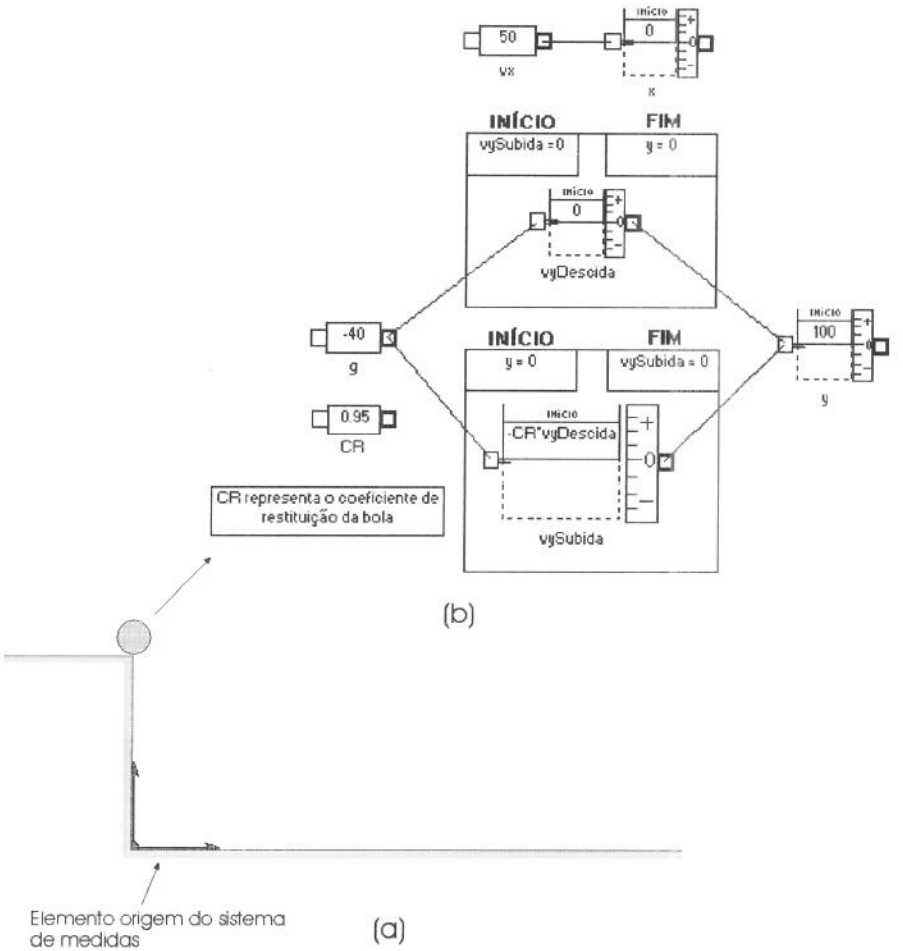


Fig 6.21: Um modelo para uma bola que quica sobre uma superfície plana horizontal

Uma vez modelado a aparência dos corpos envolvidos no fenômeno e o comportamento mecânico da bola, podemos executar o modelo estabelecido. O NEWTON apresentará a animação do comportamento mecânico referente ao modelo estabelecido. Este é importante *feedback* visto que sabemos pela nossa vivência qual deve ser o comportamento da bola. Se ela não quicar várias vezes contra a superfície inferior à medida em que, gradativamente perde a altura máxima a cada ciclo de choque contra a superfície é sinal de há algo errado com o modelo. Não é o caso deste modelo. Contudo, podemos instalar outros instrumentos de *feedback* para observar em detalhes o modelo. Instalamos um elemento tabela e elemento gráfico que mostram a relação da posição (x e y) da bola com o tempo. Como era esperado a abscissa "x" possui uma relação linear com o tempo enquanto que a ordenada "y" possui uma relação quadrática (na forma de

uma parábola) com amplitude decrescente a cada ciclo de choque da bola contra a superfície.

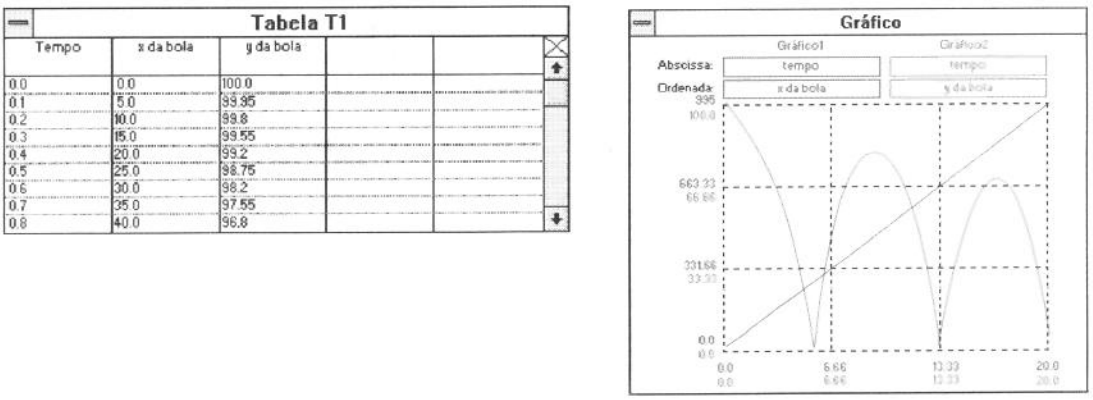


Fig. 6.22: Dois instrumentos de *feedback* instalados no modelo

Neste ponto, se quisermos, podemos executar o modelo e interromper a sua execução em momentos específicos com o objetivo de observar detalhes do fenômeno modelado. Podemos fazer isto de dois modos: (1) editando condições de parada previamente à execução como, por exemplo, parar a execução quando a ordenada "y" da bola for igual a zero (situação em que a bola choca-se contra a superfície inferior); (2) manualmente, dando-se um *click* no botão "Parar/reiniciar" da barra de ícones em meio à execução do modelo.

Este modelo pode ser facilmente incrementado para que calcule a energia potencial e a energia cinética da bola em cada instante ou, ainda, podemos aperfeiçoá-lo para que leve em consideração o atrito da superfície com a bola e a resistência que o ar impõe ao seu movimento.

6.8 - IDÉIAS E COMENTÁRIOS SOBRE O TRABALHO DE CRIAR MODELOS COM O NEWTON

O NEWTON contém todas as peças para se criar modelos para fenômenos da Mecânica dos corpos rígidos. Desde o mundo microscópico até a imensidão do universo, o

modelador pode soltar sua imaginação para criar modelos dos fenômenos que vê e, também, para os fenômenos que não vê mas supõe que existam. Se desejar, o modelador pode buscar inspiração nos modelos comentados no apêndice B. Esta seção constitui uma fonte adicional de inspiração para a criação de modelos com o NEWTON, mas ela não se resume apenas a isto: ela aborda, também, aspectos relacionados ao trabalho neste ambiente. Pode-se dizer que a presente seção é uma aquarela de idéias e comentários sobre o trabalho de se criar modelos com o NEWTON.

- **Versões distintas de modelos estabelecidos para um mesmo fenômeno.** O uso do recurso do NEWTON que permite abrir e executar mais que um modelo ao mesmo tempo, pode ser útil para comparar e estudar variações de versões distintas de modelos estabelecidos para um mesmo fenômeno.

- **Criação de modelos mais complexos a partir de modelos mais simples.** Quando se usa os comandos "Copiar", "Cortar" e "Colar" não só o desenho de um corpo é copiado, cortado ou colado na (da) área de transferência mas, também, o comportamento mecânico dele. Isto cria um fácil mecanismo para se obter modelos cada vez mais complexos a partir de modelos mais simples que já estejam prontos. Tendo sido estabelecido, por exemplo, o modelo de um motor pode-se, em seguida, copiá-lo para a área de transferência e colá-lo em um modelo de uma esteira rolante que gira sobre a ação de um motor. Sucessivamente, esteira rolante e motor podem ser copiados para compor o modelo de um trator de esteiras.

- **Modelos com vários pontos de vista de um fenômeno.** O comportamento mecânico de um automóvel e o seu respectivo motor podem ser descritos em um único modelo. Por exemplo, enquanto que o automóvel pode ser visto em movimento retilíneo em um modelo, em um detalhe à parte o seu respectivo motor pode ser visto parado, isto é, não em movimento retilíneo, mas mostrando o movimento de seus pistões e do eixo virabrequin.

- **Comportamento individual de um corpo e comportamento de seu agrupamento.** Consideremos o exemplo a seguir. Uma bicicleta possui rodas. As rodas giram em torno do seu centro e a bicicleta, como um todo, movimenta-se em linha reta. Logo, as rodas possuem dois comportamentos mecânicos: girar em torno de seu eixo central e movimentar-se em linha reta. Contudo, não é necessário modelar-se, para as rodas, estes dois comportamentos mecânicos. Pode-se modelar para elas, o comportamento

mecânico de girar em torno de seu eixo central, agrupá-las às outras peças que compõem bicicleta e, então, modelar o comportamento de movimentar em linha reta para o agrupamento.

- **Expressão qualitativa e quantitativa do diagrama de comportamento mecânico.** Os diagramas de comportamento mecânico apresentam relações qualitativas e quantitativas entre os seus elementos. O lado qualitativo dos diagramas de comportamento mecânico revela-se pelo desenho dos seus elementos, os seus nomes e as conexões entre eles. O lado quantitativo está descrito nas expressões contidas no interior dos elementos. Uma boa técnica para desenvolver diagramas que nos pareçam complicados, consiste em descrever os elementos envolvidas no fenômeno sem nos preocupar, num primeiro momento, com as relações quantitativas entre eles. Em outras palavras, descrever qualitativamenente o diagrama para, numa segunda etapa, preocupar-nos em descrevê-lo quantitativamente.

- **Posição da origem do sistema de medidas e posição de um corpo.** A posição e a orientação dos eixos que demarcam a origem do sistema de medidas podem ser alteradas (figura 6.23-a). Isto permite, entre outras coisas, (1) estabelecer a posição da origem e os sentidos de crescimento dos eixos ao sabor da conveniência de cada modelo e (2) simplificar a expressão do modelo por se escolher uma posição mais apropriada para a expressão do mesmo. Também, uma escolha apropriada da posição dos elementos "posição e eixo de rotação" de cada corpo envolvido no modelo, contribui para a simplificação da expressão do modelo. Por exemplo, observe na figura 6.23-b a posição destes elementos para os corpos "estaca" e "bate-estaca". Nestas condições pode-se descrever a posição de choque entre o "bate-estaca" e a "estaca" como: **y do bate-estaca = y da estaca**. De outra forma, isto é, se o elemento "posição e eixo de rotação" para a "estaca" tivesse sido posicionado na sua base, a expressão que define a posição de choque deveria ser: **y do bate-estaca = y da estaca - l,** onde **l** representa a altura da estaca. Esta última forma é, claramente, mais complicada que a primeira.

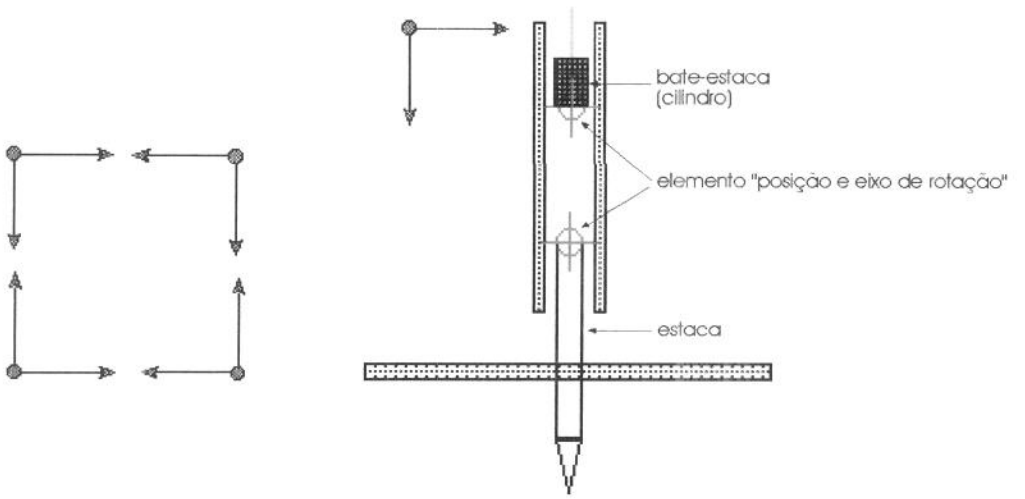


Fig. 6.23: (a) Várias posições e orientações dos eixos que demarcam a origem; (b) uma escolha adequada para o elemento posição e eixo de rotação pode simplificar a expressão do modelo

- **Usando o editor de comportamento mecânico para ajustar a posição linear e angular de um corpo.** Quando um elemento de nome x, y ou teta é criado em um diagrama de comportamento mecânico o NEWTON estabelece automaticamente o seu valor. Isto se dá por serem x e y as propriedades padrão para as posições lineares, nas direções da abscissa e ordenada respectivamente, e por ser teta a propriedade padrão para a posição angular do corpo e, desde que o modelador tenha desenhado e posicionado o corpo, os valores de x, y e teta serão conhecidos pelo NEWTON. O editor de comportamento mecânico permite que estes valores sejam alterados. E, toda alteração em um destes valores é refletida em uma correspondente alteração na posição do corpo. Isto pode ser usado sistematicamente para ajustar precisamente as posições lineares e angular dos corpos.

- **Corpos transparentes.** Pode-se tornar um corpo transparente usando-se o comando "Transparente" (mais informações, podem ser obtidas na seção A.3 do guia de referência rápida ao NEWTON no apêndice A). Se, por um lado, a transparência de um corpo (um vidro, por exemplo) sofistica o cenário, por outro lado, faz com que o NEWTON torne mais lenta a animação do modelo. Portanto, este recurso deve ser usado com cautela.

6.9 - O SUPORTE DADO PELO NEWTON AOS ESTÁGIOS DE EXECUÇÃO DE AVALIAÇÃO DE UMA AÇÃO

Vimos, na seção 4.4 do capítulo 4, que Norman (1986) indica sete estágios envolvidos no processo da ação de uma pessoa sobre um sistema: estabelecimento do objetivo, formação da intenção, especificação da seqüência de ações, execução da ação, percepção do estado do sistema, interpretação do estado do sistema e avaliação do estado interpretado com relação aos objetivos e intenções originalmente estabelecidos. A presente seção discute brevemente como o NEWTON auxilia a ação de seu usuário.

O NEWTON dá suporte à geração de intenção por lembrar ao usuário o que é possível fazer por meio de seus menus, da barra de ícones e de suas palhetas. O nome de cada elemento das palhetas e barra de ícones aparece escrito em um campo situado no lado direito da barra de ícones toda vez que o usuário move o cursor do *mouse* sobre um destes elementos. Além da forma gráfica de cada elemento lembrar o seu significado, os nomes a eles associados referenciam as suas funções e o fato destes nomes aparecerem escritos em um campo, constitui um mecanismo adicional que ajuda o usuário a descobrir e lembrar o que pode ser feito com o ambiente.

O NEWTON auxilia na especificação das ações por fazer visível os elementos respectivamente associados às ações. Auxilia, também, a execução de uma ação através de operações de apontamento feitas com o *mouse*. As principais primitivas podem ser facilmente localizadas e executadas a partir das palhetas e da barra de ícones do ambiente. As palhetas possuem, ainda, a facilidade adicional de poderem ser movimentadas para a posição da tela que melhor convier ao usuário. Reconhecemos que o ambiente poderia ajudar mais o usuário, caso contássemos com tecnologias que nos permitissem executar ações pelo toque de botões com o dedo, fazer as operações de montagem dos corpos e do comportamento mecânico utilizando as duas mãos (afinal, os usuários têm duas mãos) e executar comandos por meio da voz. No entanto, não são estas as tecnologias que utilizamos no *design* do NEWTON (vide seção 5.1 do capítulo 5).

O NEWTON auxilia a interpretação das ações executadas, como por exemplo, na ação de agrupar dois corpos, os marcadores de seleção que destacam individualmente cada um dos corpos antes da ação, passam a demarcar apenas o agrupamento após a ação. A

animação do comportamento mecânico dos corpos e os vários instrumentos de *feedback* disponíveis auxiliam a interpretação da execução do modelo estabelecido.

O NEWTON auxilia a avaliação por permitir lembranças visuais do que foi feito. Por exemplo, muitas vezes a simples observação da animação do comportamento mecânico dos corpos é suficiente para se avaliar se um dado modelo está correto. Assim, um projétil que, em sua animação, nunca "cai", pode revelar que o modelador não levou em consideração o efeito da força da gravidade sobre o projétil. Ou, a observação da animação do modelo estabelecido para uma bola que quica sobre uma superfície plana, sempre restituindo sua altura original a cada novo choque com a superfície, pode revelar que o usuário não levou em consideração que qualquer bola, na prática, não é perfeitamente elástica.

6.10 - MELHORIAS

Descrevemos a seguir algumas melhorias que achamos interessantes para o NEWTON:

- **Referencial em coordenadas polares.** Alguns fenômenos podem ser mais facilmente descritos a partir de um referencial polar[6] do que a partir de um referencial cartesiano. Um aprimoramento consiste em implementar, no NEWTON, um sistema de referencial polar. O modelador optaria entre o uso de um referencial cartesiano ou um referencial polar conforme as características do fenômeno cujo modelo ele quer desenvolver.

- **Biblioteca especializada de corpos com comportamento definido.** O NEWTON dispõe de uma biblioteca de desenhos de corpos. Não é a isto que estamos nos referindo. Imaginamos uma biblioteca que pudesse conter corpos com comportamento definido; uma coleção de corpos relacionados com determinada classe de fenômenos da Mecânica. Por exemplo, uma biblioteca de engrenagens composta de engrenagens de diversos tamanhos que pudesse ser utilizada na construção de modelos de caixas de câmbio, moendas etc.. Ou, uma biblioteca de corpos para o estudo da Estática[7] composta de vigas, apoios fixos, roletes, balancins, hastes etc., que pudesse ser

[6] Em um referencial polar a posição de um corpo no plano é dada por dois valores: (1) a distância do corpo até um ponto, que demarca a origem do referencial polar, e (2) o ângulo que o vetor de distância do corpo faz com uma linha convencionalmente tomada como linha de origem para medida dos ângulos.

[7] A Mecânica dos corpos rígidos é tradicionalmente subdividida em Estática e Dinâmica. A primeira refere-se a corpos em repouso e a segunda a corpos em movimento.

utilizada na construção de modelos para as estruturas compostas destas peças. É bem verdade que estamos alterando o nível de abstração dos modelos a serem criados. Tanto o NEWTON "puro" quanto o pretenso NEWTON equipado com uma determinada biblioteca especializada constituem ambientes para modelagem. A questão que colocamos e que, diga-se de passagem, não pertence ao escopo desta dissertação é: a que usuários servirá uma dada biblioteca especializada?

- **Janela de comandos para exibir valores.** A depuração ou análise dos resultados de um modelo é feita, em geral, interrompendo-se a execução do mesmo e verificando-se os valores das propriedades de cada corpo. Hoje, isto pode ser feito com o auxílio dos instrumentos de *feedback* disponíveis e cuja instalação deve preceder a execução do modelo. Uma janela que permitisse digitar o nome de uma propriedade e apresentasse, em seguida, o valor da mesma, constitui um meio alternativo para se analisar os resultados do modelo com a vantagem adicional de se poder determinar, em tempo de interrupção da execução, qual propriedade examinar.

- **Comparação de valores.** As grandezas envolvidas em um fenômeno da Mecânica, em geral, são contínuas enquanto que o NEWTON as representa de forma discreta. Isto cria problemas na comparação de valores. Por exemplo, entre o tempo **t = 0.5** e o tempo **t = 0.6** existem uma infinidade de valores de tempo. No entanto, dependendo do valor do passo de integração definido, o NEWTON entende que depois do tempo **t = 0.5** o próximo valor do tempo é **t = 0.6**. Isto cria uma complicação na avaliação das condições que aparecem nos modelos. Por exemplo, estabelecer uma condição de início do movimento acelerado de um corpo quando o valor do tempo for **t = 0.55**. O NEWTON tenta resolver este problema permitindo que o usuário estabeleça uma imprecisão na comparação de valores. Dependendo da imprecisão estabelecida, **0.55** fica igual, para fins de comparação, a **0.6** ou **0.5** e a referida complicação fica razoavelmente resolvida. No entanto, certas grandezas mal comportadas ultrapassam os limites aceitáveis de imprecisões e o NEWTON falha. Melhorias poderiam envolver interpolação das grandezas envolvidas nas comparações ou sinalizadores que indicassem, retomando o exemplo, que se o tempo era **t = 0.5** e o tempo é, no momento, **t = 0.6** então o tempo foi, em algum instante, **t = 0.55**.

- **Diversas.** Desfazer uma operação realizada, reorganizar (automaticamente) as janelas dos modelos abertos, poder acionar um tutorial com informações sobre o uso do ambiente, auxílio na edição das expressões dos elementos do editor de comportamento

mecânico através de uma caixa de diálogo que pudesse mostrar os operadores, as funções predefinidas e as regras de sintaxe para escrita de expressões, são mais algumas das melhorias que achamos interessantes para o NEWTON.

Capítulo 7

O NEWTON do ponto de vista de sua implementação

Neste capítulo, mostramos como o ambiente para modelagem NEWTON foi implementado; descrevemos a sua estrutura geral, a tecnologia utilizada no desenvolvimento, os principais problemas de implementação encontrados e as respectivas soluções adotadas.

Tópicos deste capítulo:

Tópico		*Página*
7.1 -	Tecnologia empregada na implementação	145
7.2 -	Estrutura geral do NEWTON e dos modelos por ele criados	150
7.3 -	A edição de corpos, de comportamento mecânico e dos instrumentos de *feedback*	155
7.4 -	Diagramas de comportamento mecânico e equações diferenciais	157
7.5 -	A execução dos modelos	167

7.1 - TECNOLOGIA EMPREGADA NA IMPLEMENTAÇÃO

Empregamos na implementação do NEWTON um microcomputador **AT 486**, equipado com *winchester, mouse* e monitor **super VGA** colorido, executando o sistema operacional **Microsoft Windows** versão 3.1. Fizemos uso do ambiente para desenvolvimento **ToolBook** versão 1.5 para programação do NEWTON. Escolhemos trabalhar com o *hardware* citado por ser este, hoje em dia, amplamente difundido. A escolha do sistema operacional recaiu sobre o Microsoft Windows, principalmente, pelo fato de sua interface gráfica com o usuário favorecer a expressão dos elementos do domínio escolhido. Optamos pelo ambiente de desenvolvimento ToolBook devido às facilidades que este ambiente oferece ao desenvolvimento de protótipos.

O ToolBook é ambiente de desenvolvimento para Windows, orientado a objetos e a eventos, que inclui "hipernavegação", ferramentas para criação de telas gráficas sem programação, uma rica linguagem orientada a objetos chamada OpenScript e ferramentas para edição e depuração das aplicações desenvolvidas (Assymetrix, 1991-a,b,c,d). No ToolBook os objetos gráficos podem ser criados, sem necessidade de programação, por meio de suas ferramentas, através de ações com o *mouse*. O comportamento destes objetos pode ser definido através de programas escritos na linguagem OpenScript. O ToolBook também é o ambiente para execução de seus aplicativos. Muitas das tarefas tradicionais de desenvolvimento são simplificadas pelas ferramentas de criação de objetos disponíveis no ToolBook. Para compreender o desenvolvimento de aplicações no ToolBook é importante entender que:

- por ser orientado a eventos, **nada acontece em uma aplicação ToolBook até que um evento ocorra.** Muitos eventos são gerados por ações do usuário, tais como apontar para um objeto e dar um *click* nele. Diferente do que ocorre com aplicações tradicionais que requerem que o usuário faça um conjunto específico de ações em uma certa ordem em lugar e tempo específicos, uma aplicação ToolBook pode responder a um evento se ele ocorrer;

- **o ToolBook possui objetos com comportamentos predefinidos.** Em vez de se programar todas as ações para o aplicativo ToolBook, pode-se definir, apenas, os comportamentos não predefinidos. Por exemplo, o ToolBook possui menus predefinidos para ações tais como abrir ou gravar um arquivo e formatar um texto; Isto torna o ToolBook atrativo para prototipação.

- **o ToolBook é extensível**. Pode-se definir mensagens, funções e propriedades para os objetos; adicionar, remover ou redefinir os comandos dos menus; trocar dados com outras aplicações, chamar funções de qualquer DLL[1] do Windows; traduzir e responder mensagens de sistema advindas do windows.

O Microsoft Windows é um sistema operacional que provê uma interface gráfica para o usuário comum para todas as suas aplicações através de janelas, menus, caixas de diálogos e controles gráficos (Microsoft Corporation, 1990). Pelo fato do ToolBook executar em Microsoft Windows, o programador pode contar com o Windows para muitas tarefas tradicionais no desenvolvimento de aplicações:

- o Windows provê a estrutura da janela e os respectivos ícones para controle, minimização e maximização. O comportamento da janela e de seus ícones fica automaticamente estabelecido no momento da sua criação. Assim, a aplicação se parecerá e atuará da mesma forma como outras aplicações Windows;

- o Windows fornece as fontes que uma aplicação usa para exibir textos. Se o aplicativo referencia uma fonte que não está disponível, o Windows imediatamente a substitui por uma fonte de características semelhantes;

- o Windows dispõe de *drivers* para diferentes cartões gráficos e dispositivos de saída. Isto significa que um aplicativo pode fazer uso de qualquer dispositivo aceito pelo Windows sem que seja necessário nenhuma programação adicional;

- o Windows é um sistema multitarefa. Logo, um aplicativo pode compartilhar dispositivos ou trocar dados com outros aplicativos na mesma seção de trabalho;

- o Windows gerencia a fila de ações executadas no teclado ou no *mouse* e envia a um aplicativo, apenas, as entradas destinadas a ele. Assim, um aplicativo não precisa se preocupar em descartar as entradas que são endereçadas a outros aplicativos;

---

[1] DLL é abreviatura, em inglês, de *Dinamic Link Libraries* que são bibliotecas de funções que podem ser utilizadas dinamicamente por um aplicativo Windows. Algumas destas DLLs são distribuidas com o Windows, como por exemplo, a "kernel.exe" que contém funções que permitem alocação de memória global por um aplicativo. Outras DLLs são distribuidas juntas com o ToolBook como a "TBKDLG.DLL" que possui funções para exibir e receber informações de caixas de diálogo ao estilo do Windows e a "TBKFILE.DLL" que possui funções que permitem copiar, eliminar e proceder a outras operaçãoes comuns a arquivos. Pode-se, também, escrever DLLs com auxílio de linguagens como C e, posteriormente, utilizá-las em um aplicativo ToolBook.

- o Windows gerencia a memória como um recurso compartilhado entre aplicativos. Um aplicativo pode contar com a cooperação entre o ToolBook e o Windows para gerenciamento da memória que necessitar.

Como dissemos nada acontece no ToolBook até que um evento ocorra. Alguns eventos resultam de ações tal como abrir um arquivo, pressionar uma tecla ou algum dos botões do *mouse* ou escolher um item do menu. Outros eventos incluem mover o ponteiro do *mouse* sobre a tela, criar ou destruir um objeto e, até mesmo, não fazer nada durante algum tempo. Por exemplo, um evento típico se dá pela ação de dar um *click* na tecla esquerda do *mouse* enquanto o cursor do mesmo está sobre um objeto gráfico ou um botão desenhado na tela. O ToolBook é notificado que um evento ocorreu por meio de uma mensagem. As mensagens são os elos de ligação entre os eventos e as respectivas ações tomadas por um aplicativo. Cada evento gera uma ou mais mensagens que são enviadas para um objeto. Por exemplo, pressionar o botão esquerdo do *mouse* gera a mensagem *buttonDown;* liberar o mesmo botão gera a mensagem *buttonUp;* Cada uma destas mensagens são enviadas para o objeto sob o cursor do *mouse* na tela. Quando uma mensagem é enviada a um objeto ele poderá ou não respondê-la. Ele a responderá, caso possua um *handler* para a mensagem enviada. Um *handler* é um programa escrito na linguagem OpenScript que especifica o que deve acontecer quando uma determinada mensagem é recebida por um objeto. A resposta pode ser, simplesmente, escrever alguma coisa na tela ou pode ser uma série complexa de ações que incluem a execução de outros *handlers.* A tarefa de desenvolver um aplicativo ToolBook consiste na escrita de *handlers* para responder a mensagens geradas por eventos. E para isto, deve-se decidir: (1) a quais eventos a aplicação irá responder, (2) quais mensagens serão respondidas, uma vez que um evento pode gerar várias mensagens e (3) se o *handler* que responderá a uma mensagem será escrito no *script* do objeto que inicialmente a receberá ou no *script* de outro objeto.

O ToolBook usa a metáfora de um livro para definir uma aplicação. Assim como um livro, um aplicativo ToolBook pode conter páginas. Estas páginas podem conter objetos gráficos tais como desenhos, campos e botões. Os campos admitem textos em seu interior. Os botões são elementos gráficos similares aqueles encontrados no ambiente Windows e que se prestam, normalmente, para fazer ações tal como a de virar uma página de livro quando recebem um *click* do *mouse.* Dois ou mais objetos podem ser reunidos para formar um grupo, sendo que o grupo formado é considerado um novo objeto. Uma página no ToolBook possui uma frente (a própria página) e um fundo. Uma

vez que é comum um livro ter páginas com o mesmo estilo de fundo, no ToolBook um fundo de página pode ser associado a várias páginas. Na prática, o conceito de fundo de página está relacionado com a facilidade de operação do usuário do ToolBook em produzir as páginas de um livro, uma vez que este não terá que criar vários fundos de páginas quando eles forem idênticos. Além disso, a utilização do conceito de fundo de página conduz a uma diminuição do volume de dados armazenados no arquivo que contém o livro. Genericamente, páginas, fundos, grupos, botões, campos e desenhos são denominados objetos. A figura 7.1 ilustra os objetos que podem ser criados no ToolBook.

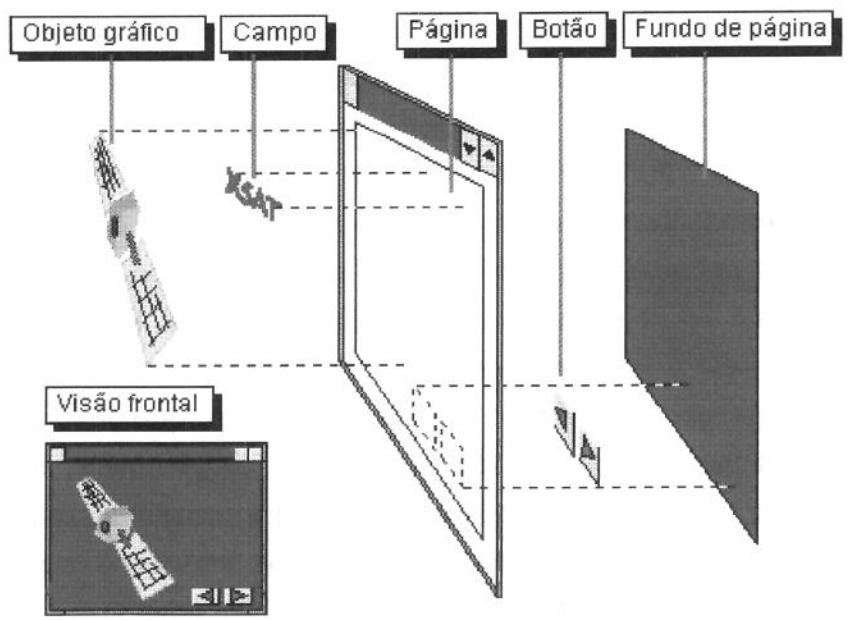


Fig. 7.1: Alguns objetos do ToolBook

Todo objeto definido no ToolBook pode conter um *script*. Um *script* é uma coleção de um ou mais *handlers* que definem como um objeto responde às mensagens. Quando uma mensagem é enviada, o ToolBook procura por um *handler* para respondê-la, começando pelo *script* do objeto que recebeu a mensagem (o alvo da mensagem). Se o *script* do objeto que recebeu a mensagem possui um *handler* para respondê-la, o ToolBook executa este *handler* e a mensagem não trafegará pela hierarquia de objetos do ToolBook (figura 7.2), a menos que o *handler* assim especifique. Caso não exista o *handler* para responder à mensagem no *script* do objeto alvo da mensagem, o ToolBook procurará por um *handler* no *script* de algum objeto, na ordem crescente da hierarquia dos objetos. O

ToolBook procura por um *handler* até encontrá-lo ou até encontrar o topo da hierarquia dos objetos.

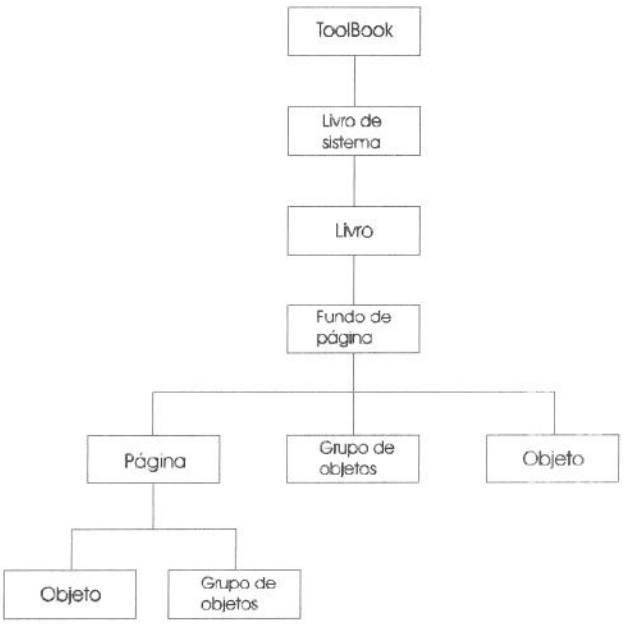


Fig. 7.2: A hierarquia de objetos para manipulação de mensagens

Um aplicativo ToolBook pode declarar um ou mais livros como sendo um **livro de sistema**. A declaração de um livro como um livro de sistema faz com que este seja adicionado na hierarquia de objetos do livro que o declarou. Quando uma mensagem não encontra um *handler* para respondê-la no *script* de um livro, ela procurará pelo *handler* em algum livro de sistema declarado. Livros de sistema são ideais para se definir *handlers* gerais para serem utilizados por vários livros. Ou ainda, eles podem ser úteis para se reunir *handlers* relacionados, formando algo semelhante a uma biblioteca de funções em linguagens como C.

As mensagens entram na hierarquia dos objetos em diferentes níveis, dependendo da classe da mensagem. Por exemplo, mensagens geradas pelo *mouse* são enviadas para o objeto sob o cursor do *mouse.* Já mensagens geradas pela manipulação do menu são enviadas para a página corrente. O ToolBook possui *handlers* predefinidos para muitas mensagens geradas por eventos. Por exemplo, a resposta predefinida do ToolBook para escolha de um comando no menu é exibir uma caixa de diálogo ou fazer a operação

indicada. Para muitas mensagens, a resposta predefinida do ToolBook é fazer nada. Por exemplo, o ToolBook não faz coisa alguma em resposta a eventos tais como mover o cursor do *mouse* ou dar um *click* em um de seus botões. Deixar o ToolBook dar respostas predefinidas a diversos eventos constitui um meio de liberar tempo do programador para que este se concentre na escrita de *handlers* que adicionem novos comportamentos a um aplicativo. Isto torna ToolBook ainda mais atrativo para utilização como uma ferramenta para desenvolvimento de protótipos. Adicionalmente às mensagens geradas pelo ToolBook a eventos relacionados com as ações do usuário, o programador pode definir novas mensagens e escrever *handlers* que as manipule.

7.2 - ESTRUTURA GERAL DO NEWTON E DOS MODELOS POR ELE CRIADOS

O NEWTON integra em seu ambiente:

- um **editor de desenhos de corpos**, que possibilita modelar a aparência dos corpos nos vários modelos abertos ao mesmo tempo no ambiente;

- um **editor de comportamento mecânico** que, em cada vez em que é acionado, permite modelar o comportamento mecânico de um corpo existente em um dos modelos abertos;

- um **editor de instrumentos de *feedback*** que possibilita "instalar" instrumentos para acompanhar a execução dos modelos abertos;

- um **módulo executor** que compila, calcula e simula, de uma só vez, os vários modelos abertos;

- um **módulo gerenciador de modelos** abertos responsável por operações tais como abrir, fechar, gravar, maximizar e minimizar as janelas dos modelos.

Na prática, projetamos o NEWTON como um conjunto aplicações ToolBook (livros) que trabalham de forma harmônica e integrada (figura 7.3).

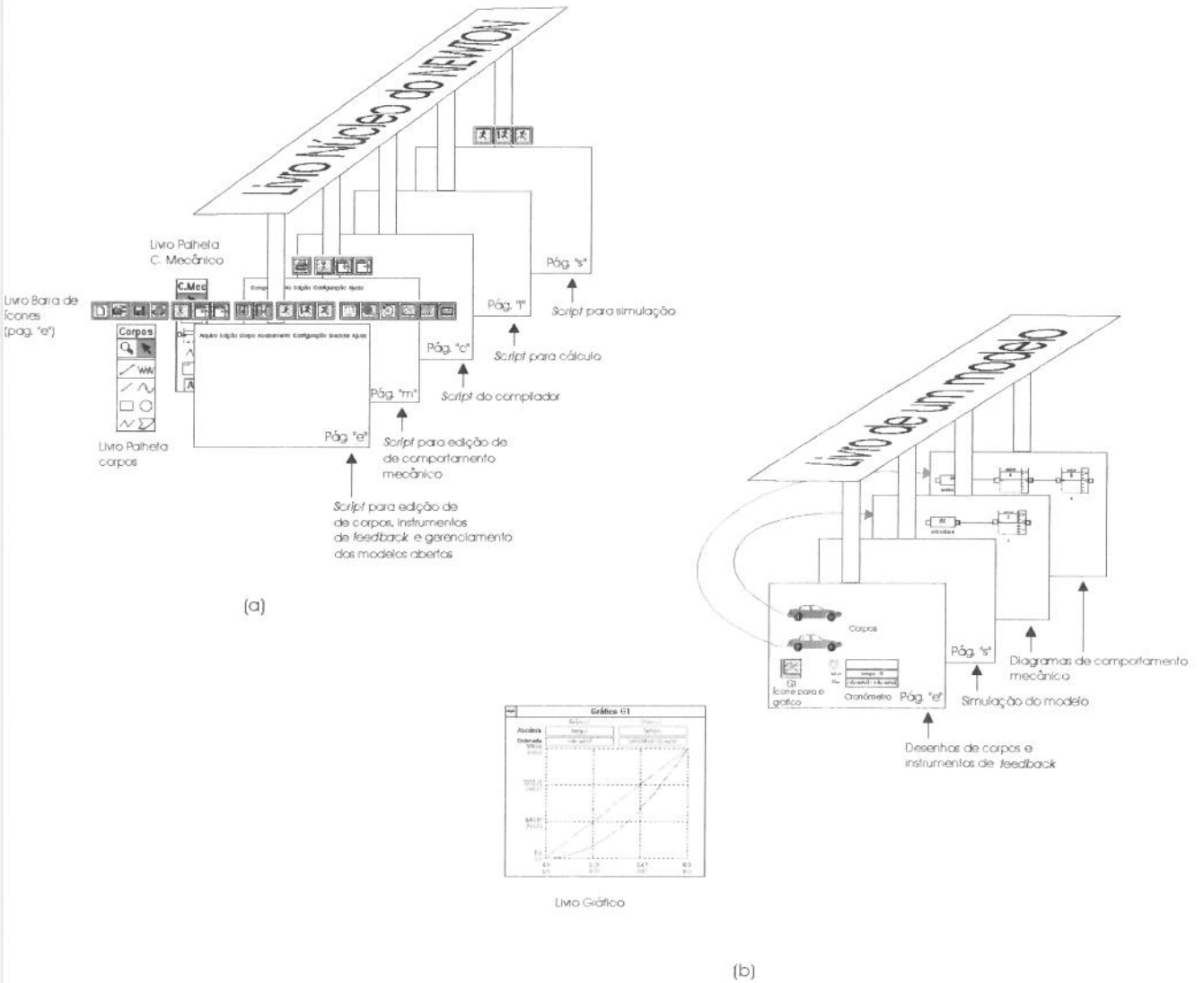


Fig. 7.3: Estrutura geral do ambiente para modelagem NEWTON: (a) o núcleo do NEWTON e os livros periféricos palheta corpos, palheta de comportamento mecânico e barra de ícones; (b) o livro de um modelo e o livro de um gráfico

O Núcleo do NEWTON é um livro em que o *script* de cada uma de suas páginas desempenha um papel específico relacionado à coordenação das atividades de: gerência dos modelos abertos, edição do desenho de corpos, edição de comportamento mecânico, edição de instrumentos de *feedback*, compilação, cálculo e simulação dos modelos. Um conjunto de livros trocam dados dinamicamente com o livro Núcleo do NEWTON através de protocolo DDE (*Dinamic Data Exchange*) do Windows. Estes livros desempenham papeis relacionados às ações de entrada e saída. É o caso da palheta para edição de corpos, da palheta para edição de comportamento mecânico, da barra de ícones e das

tabelas e dos gráficos que, por ventura, sejam "instalados" para acompanhar a execução dos modelos. Cada um deles representa um aplicativo Windows individual ou, usando a metáfora do ToolBook, cada um deles constitui um livro à parte. Também, os modelos que venham a ser desenvolvidos constituem livros ToolBook[2]. Tais livros possuem um conjunto de páginas destinadas a armazenar os corpos desenhados, os instrumentos de *feedback* "instalados" (cronômetro, distanciômetro, ícones para os gráficos e tabelas), os diagramas de comportamento mecânico definidos e, ainda, reservam uma página para a simulação do modelo. A simulação não ocorre na página reservada aos desenhos dos corpos e dos instrumentos de *feedback* (página "e"). Ela se dá na página "s". Imediatamente antes da simulação se iniciar, o NEWTON copia todos os corpos e instrumentos de *feedback* existentes na página "e" para a página "s" do modelo. Assim, o NEWTON preserva o conteúdo da página "e" de um modelo, mantendo as posições dos corpos exatamente como o modelador as definiu.

As páginas dos diversos livros que compõem o ambiente para modelagem NEWTON guardam, entre elas, uma relação muito específica quanto ao momento em que são visitadas. Assim, quando o usuário está editando o desenho de corpos, o livros Núcleo do NEWTON, Barra de Ícones e dos modelos abertos estarão em suas respectivas páginas "e"; neste momento, o livro palheta Corpos estará sendo apresentado (em sua única página) e o livro palheta Comportamento Mecânico estará escondido. Quando o usuário estiver editando o comportamento mecânico de algum corpo de um dos vários modelos que podem ser abertos, os livros Núcleo do NEWTON e Barra de Ícones estarão em suas respectivas páginas "m"; o livro palheta Corpos estará escondido e o livro palheta Comportamento Mecânico estará sendo exibido (em sua única página); o livro do modelo cujo corpo estiver tendo o seu comportamento mecânico editado, estará sendo mostrado em uma página que conterá o diagrama de comportamento mecânico associado ao referido corpo; quando a simulação estiver ocorrendo os livros Núcleo do NEWTON, Barra de Ícones e dos modelos abertos estarão sendo apresentados em suas respectivas páginas "s".

Embora não estejam sendo apresentados na figura 7.3 o ambiente para modelagem NEWTON conta, ainda, com um conjunto de livros de sistema cujos *script*: (1) permitem responder a ações do usuário durante a edição dos corpos ("edcorpos.sbk"); (2) permitem responder a ações do usuário durante a edição de comportamento mecânico e

[2] Não estamos querendo dizer com isto que um modelo desenvolvido pelo NEWTON pode ser executado pelo ToolBook sem a presença do NEWTON. Pelo contrário, modelos criados no NEWTON só podem ser executados corretamente por este.

compilação dos modelos ("edcmec1.sbk" e "edcmec2.sbk"); (3) dispõe de funções que auxiliam no cálculo dos modelos ("calculo.sbk"). De forma geral, a hierarquia dos objetos e a comunicação entre os diversos livros que formam o ambiente para modelagem NEWTON está mostrada na figura 7.4.

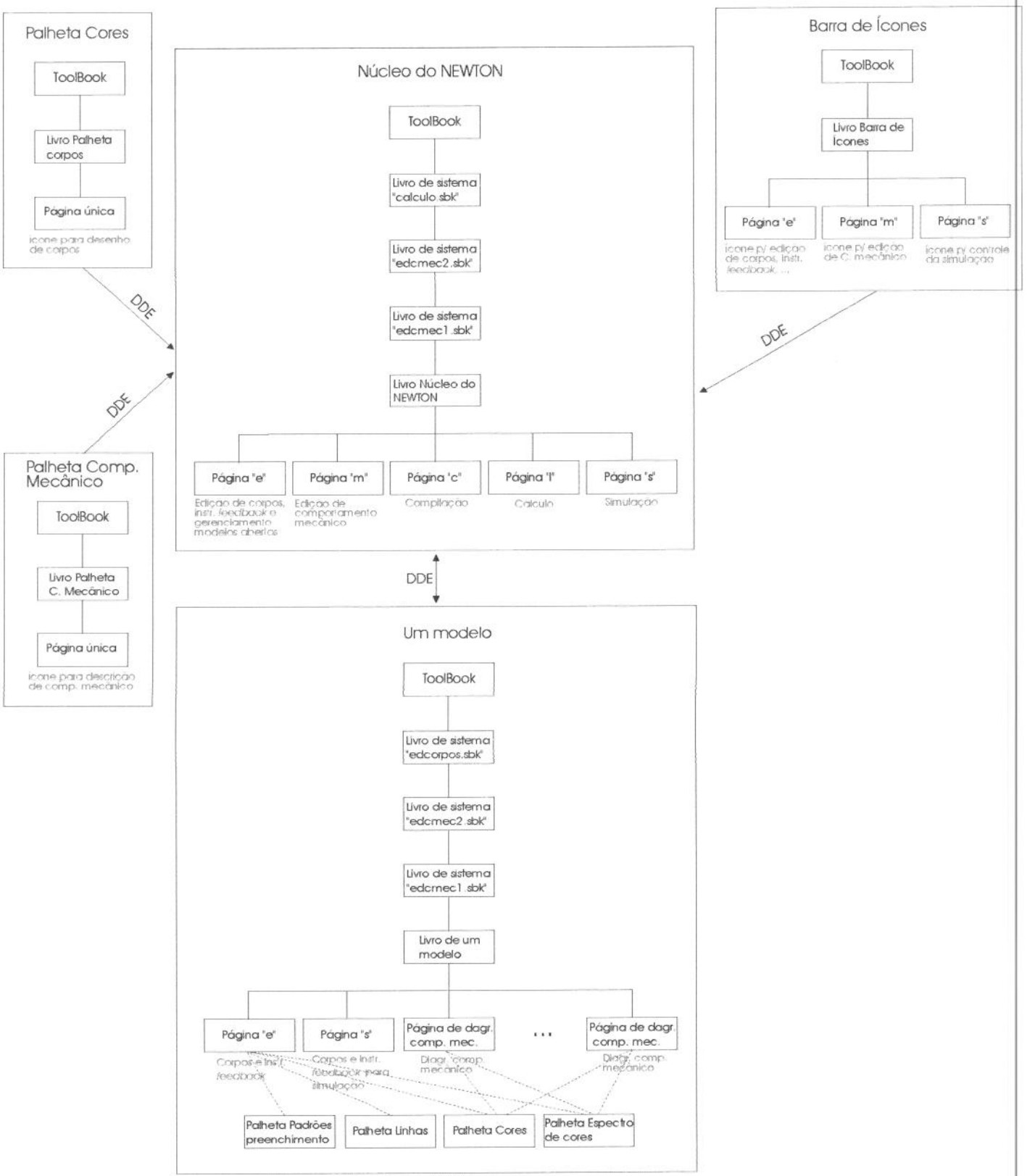


Fig. 7.4: Hierarquia de objetos e comunicação entre os diversos livros que formam o ambiente para modelagem NEWTON

7.3 - A EDIÇÃO DE CORPOS, DE COMPORTAMENTO MECÂNICO E DOS INSTRUMENTOS DE *FEEDBACK*

Na edição de corpos os livros Núcleo do NEWTON, Barra de Ícones e dos modelos abertos apresentam-se em suas respectivas páginas "e". Quando um usuário escolhe uma ferramenta na palheta Corpos, a ferramenta escolhida é comunicada ao livro Núcleo do NEWTON através de protocolo de intercâmbio dinâmico de dados entre aplicativos (DDE). Especificamente, a comunicação se dá com o *handler* controlarAçãoPalhetaCorpos da página "e" do livro Núcleo do NEWTON. Tal *handler* tem a função de controlar as ações executadas, pelo usuário, na palheta Corpos. Assim, por exemplo, quando o usuário escolhe desenhar um retângulo, este *handler* altera o cursor do *mouse* para uma cruz (a cruz significa que o NEWTON está preparado para desenhar) ou quando o usuário escolhe aumentar ou diminuir o zoom dos modelos, o *handler* altera o cursor do *mouse* para o desenho de uma lupa. Adicionalmente, o referido *handler* informa a cada modelo aberto a ferramenta correntemente selecionada na palheta. A partir daí, qualquer ação que o usuário realize em um dos modelos abertos refletirá o comportamento que se espera da ferramenta selecionada. Os *handlers* que perfazem o comportamento que se espera de uma ferramenta não estão colocados em cada uma das páginas "e" dos modelos abertos mas no livro de sistema "edcorpos.sbk", cujo *script* serve a qualquer dos modelos abertos. A página "e" do livro Núcleo do NEWTON é responsável pelas ações relacionadas à edição de corpos que estão disponíveis, para o usuário, na barra de menu. Esta página contém um *handler* para cada uma das seguintes ações de edição de corpos executadas a partir do:

- menu edição: cortar, copiar, colar, apagar, selecionar tudo, exibir a biblioteca de desenho de corpos;
- menu corpo: agrupar, desagrupar, trazer para a frente, enviar ao fundo, trazer um nível à frente, enviar um nível ao fundo;
- menu acabamento: tornar transparente um corpo, alinhar corpos, acertar o desenho de um corpo.

Os *handlers* que respondem às ações executadas a partir da barra de menus são os mesmos que respondem às respectivas ações que, também, podem ser acionadas através da barra de ícones. Assim, por exemplo, o *handler* que responde à ação de dar um *click* no botão "copiar" da barra de ícones é o mesmo *handler* que responde à ação de se escolher o comando copiar do menu edição. O mecanismo implementado no caso de ação

do usuário sobre a barra de ícones difere da respectiva ação por meio de menu, apenas, na forma em que o *handler* que executa a ação de copiar é acionado. No caso de ação sobre a barra de ícones, o *handler* é acionado por meio de intercâmbio dinâmico de dados entre o livro da Barra de Ícones e o livro Núcleo do NEWTON.

Quando se escolhe editar o comportamento mecânico de um corpo a palheta Corpos é escondida, a palheta Comportamento Mecânico é exibida, o menu existente na área de edição de corpos é substituído por outro menu relacionado à edição de comportamento mecânico e os livros Núcleo do NEWTON e Barra de Ícones são exibidos em suas respectivas páginas "m". O livro do modelo cujo corpo tem o seu comportamento editado apresenta-se em uma página que está vinculada ao corpo. O mecanismo implementado para o desenho dos diagramas de comportamento mecânico é similar ao de edição de corpos. Assim, existe na página "m" do Núcleo do NEWTON um *handler* denominado controlarAçãoPalhetaCMecânico que é responsável pelo controle das ações disponíveis na palheta Comportamento Mecânico. Estas ações são comunicadas ao livro Núcleo do NEWTON por intermédio do protocolo DDE. Este *handler* comunica ao modelo cujo comportamento está sendo editado qual a ferramenta que está correntemente selecionada e coloca no *clipboard* uma cópia do protótipo da ferramenta selecionada, obviamente se a ferramenta selecionada for um elemento usado no desenho dos diagramas (quantidade acumulada, função algébrica, conector etc.). O que estamos chamando de protótipo de elemento, é o desenho de um objeto com comportamento predefinido e cujas dimensões são pequenas. Por exemplo, o protótipo do elemento quantidade acumulada é um pequeno desenho do elemento quantidade acumulada. Este objeto possui um *script* capaz de lhe conferir comportamentos peculiares tais como permitir que o usuário edite o seu campo de valor inicial, o seu nome e altere as suas dimensões. Quando a ação de desenhar tem início o protótipo de elemento é colado na tela do modelo. Por ter dimensões pequenas, o protótipo é colado na tela sem que o usuário perceba. A partir daí as dimensões do protótipo são aumentadas pelo arraste do cursor do *mouse*. Para o usuário, tudo se passa como se o elemento estivesse sendo desenhado sob o cursor do *mouse* a partir do nada. Os *handlers* que respondem às ações relativas ao desenho dos diagramas encontram-se inseridos nos livros de sistema "edcmec1.sbk" e "edcmec2.sbk" (figura 7.4). Estes livros de sistema são responsáveis, também, pela resposta à ação de testar a sintaxe e verificar se determinada construção diagramática está correta. A página "m" do livro Núcleo do NEWTON contém *scripts* para responder às ações relacionadas à edição de comportamento mecânico que estão disponíveis, para o usuário, na barra de menu. Esta

página contém um *handler* para cada uma das seguintes ações de edição do comportamento mecânico de um corpo, executadas a partir do:

- menu comportamento: imprimir o diagrama e configurar a impressora;
- menu edição: cortar, copiar, colar, apagar, selecionar tudo;
- menu configuração: exibir ou esconder a barra de ícones, a palheta Comportamento Mecânico e a palheta Cores;
- menu ajuda: exibir ajuda através de um índice, de um tutorial ou de um glossário.

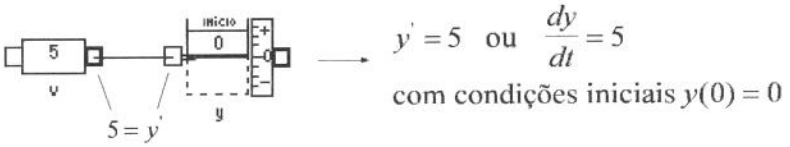
Instrumentos de *feedback* são editados juntamente com o desenho dos corpos. Durante a edição dos instrumentos de *feedback* os livros Núcleo do NEWTON, Barra de Ícones e dos modelos abertos exibem as suas respectivas páginas "e". Quando seleciona-se um instrumento de *feedback* na barra de ícones, o livro Barra de Ícones insere no *clipboard* um protótipo do instrumento selecionado e comunica, através de DDE, ao livro Núcleo do NEWTON que um instrumento deve ser recebido pelo modelo ativo. O Núcleo do NEWTON, então, solicita que o modelo ativo receba o protótipo do instrumento. Similarmente aos protótipos dos elementos para composição dos diagramas de comportamento mecânico, os protótipos de instrumentos de *feedback* são objetos com desenho e comportamento predefinidos. Assim, por exemplo, o protótipo de um cronômetro possui a forma do instrumento de *feedback* cronômetro (figura 6.20 do capítulo 6) e possui *script* que lhe confere o comportamento que permite ao usuário alterar as suas dimensões e editar as condições de início e fim da cronometragem.

7.4 - DIAGRAMAS DE COMPORTAMENTO MECÂNICO E EQUAÇÕES DIFERENCIAIS

Os diagramas de comportamento mecânico da forma como definimos representam o que, na matemática, denominam-se equações diferenciais ordinárias. Uma equação diferencial é uma equação que envolve uma função incógnita e suas derivadas (Bronson, 1977). Uma equação diferencial é dita ser ordinária[3] se a sua função incógnita depende, apenas, de uma variável independente. A variável independente envolvida em todas as nossas equações é o tempo . Por exemplo, o diagrama da figura 7.5-a é uma representação para

[3] Em contraposição às equações diferenciais ordinárias estão as equações diferenciais parciais, as quais apresentam funções incógnitas que dependem de duas ou mais variáveis independentes.

a equação diferencial[4] apresentada na figura 7.5-b e têm como variável independente o tempo (t). Por simplicidade, neste texto estaremos denominando as equações diferenciais ordinárias , apenas, de equações diferenciais.



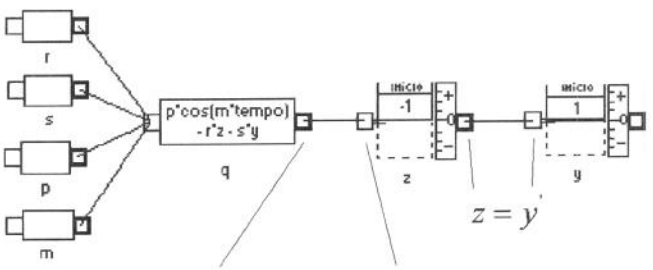
$y' = 5$ ou $\frac{dy}{dt} = 5$

com condições iniciais $y(0) = 0$

(a) (b)

$y'' + r.y' + s.y = p.\cos(m.tempo)$

com condições iniciais $\begin{cases} y(0) = 1 \text{ e} \\ y'(0) = -1 \end{cases}$



$p.\cos(m.tempo) - r.z - s.y = z'$

(c) (d)

Fig. 7.5: Diagramas de comportamento mecânico e equações diferenciais

Inversamente, dada uma equação diferencial podemos representá-la por meio dos diagramas de comportamento mecânico que propomos. Por exemplo, consideremos a equação diferencial de segunda ordem[5] completa dada por

$$y'' + r.y' + s.y = p.\cos(m.t) \qquad (7.1)$$

com condições iniciais $\begin{cases} y(0) = 1 \text{ e} \\ y'(0) = -1 \end{cases}$

[4] Estamos usando neste texto a notação y', y'', y''', $y^{(n)}$ para as derivadas, respectivamente, de primeira, segunda e n-ésima órdem de y em relação ao tempo. Outras notações equivalentes tratam de descrever estas respectivas derivadas por $\frac{dy}{dt}, \frac{d^2y}{dt^2}, \frac{d^3y}{dt^3}, \frac{d^ny}{dt^n}$ ou por $\dot{y}, \ddot{y}, \dddot{y}, y^{(n)}$.

[5] A ordem de uma equação diferencial é a ordem da mais alta derivada que nela comparece (Bronson, 1977).

onde t representa o tempo e r, s, p e m são constantes. Podemos escrever esta equação diferencial de segunda ordem por meio de um sistema de equações diferenciais de primeira ordem equivalente. Assim se fizermos,

$$y' = z \qquad (7.2)$$

podemos escrever que

$$z' + r.z + s.y = p.\cos(m.t) \qquad (7.3)$$

e isto nos conduz ao seguinte sistema de equações diferenciais de primeira ordem

$$\begin{cases} y' = z \\ z' = p.\cos(m.t) - r.z - s.y \end{cases} \qquad (7.4)$$

com condições iniciais $y(0) = 1$ e $z(0) = y'(0) = -1$.

O diagrama da figura 7.5-d representa este sistema de equações diferenciais de primeira ordem. O diagrama mostra que a expressão $p.\cos(m.t) - r.z - s.y$, representada pela função algébrica de nome **q**, é uma taxa de variação para a quantidade acumulada **z** (segunda das equações 7.4); **z**, por sua vez, atua como uma taxa de variação para a quantidade acumulada **y** (primeira das equações 7.4).

Uma vez que os diagramas de comportamento mecânico representam equações diferenciais, executar um diagrama é o mesmo que resolver as equações diferenciais correspondentes. Em testes que antecederam o projeto do NEWTON, experimentamos os seguintes métodos numéricos para resolução de equações diferenciais citados por Bronson (1977): Euler (método de primeira ordem), Runge-Kutta de segunda ordem, Runge-Kutta de quarta ordem, Milne (quarta ordem) e Hamming (quarta ordem). Um método numérico para resolução de equações diferenciais é de ordem **n** (**n** inteiro e positivo) se é exato para polinômios de grau menor ou igual a **n** (Bronson, 1977). Em outras palavras, se a solução verdadeira de uma equação diferencial é um polinômio de grau menor ou igual a **n**, então a solução verdadeira e a solução numérica coincidirão para um método de ordem **n**. Em geral, quanto maior a ordem de um método numérico, maior a sua

precisão. Em contrapartida, pode-se esperar, também, que quanto maior for a ordem de um método numérico, maior será a quantidade de cálculos envolvidos na sua aplicação. Em nossos testes observamos que:

- os três métodos de quarta ordem testados foram os mais precisos e apresentaram resultados equivalentes. Dos três métodos de quarta ordem testados, o de Runge-Kutta foi o mais lento; isto era esperado pois este método envolve mais cálculos para chegar à solução que os outros dois. Em contrapartida, os métodos de Milne e de Hamming exigem valores iniciais que necessitam ser calculados por outro método de quarta ordem como, por exemplo, o método de Runge-Kutta;
- seria inviável, com a tecnologia que estamos considerando, resolver as equações diferenciais e, simultaneamente, animar os corpos segundo o comportamento definido, mesmo para pequenos modelos (modelos envolvendo até oito elementos do tipo quantidade acumulada).

Assim, entendemos que deveríamos projetar o NEWTON de tal forma que a compilação, o cálculo das equações diferenciais e a simulação do modelo ocorressem em estágios distintos. Com o cálculo das equações diferenciais sendo feito antes da simulação, não precisávamos temer problemas de performance durante a animação e, assim, decidimos usar um dos métodos mais precisos dentre os que testamos. Escolhemos utilizar o método de Runge-Kutta de quarta ordem para simplicidade da implementação, uma vez que, a escolha do método de Milne ou Hamming obrigaria, adicionalmente, a implementação de um outro método de quarta ordem, talvez o próprio método de Runge-Kutta.

A solução de um sistema de duas equações diferenciais de primeira ordem[6]

$$\begin{cases} y' = f(t, y, z) \\ z' = g(t, y, z) \end{cases} \tag{7.5}$$

$$\text{com condições iniciais} \quad \begin{matrix} y(t_0) = y_0 \\ z(t_0) = z_0 \end{matrix} \quad \text{e}$$

pelo método Runge-Kutta de quarta ordem é descrita por Bronson (1977) como:

[6] Dizer que um sistema equações diferenciais de primeira ordem possui duas equações, é o mesmo que dizer que o seu diagrama de comportamento mecânico correspondente possui dois elementos do tipo quantidade acumulada.

$$y_{n+1} = y_n + \frac{1}{6}(k_{y1} + 2k_{y2} + 2k_{y3} + k_{y4})$$
$$z_{n+1} = z_n + \frac{1}{6}(k_{z1} + 2k_{z2} + 2k_{z3} + k_{z4})$$

onde

$$\begin{aligned}
k_{y1} &= hf(t_{n+1} - h, y_n, z_n) \\
k_{z1} &= hg(t_{n+1} - h, y_n, z_n) \\
k_{y2} &= hf(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y1}, z_n + \frac{1}{2}k_{z1}) \\
k_{z2} &= hg(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y1}, z_n + \frac{1}{2}k_{z1}) \\
k_{y3} &= hf(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y2}, z_n + \frac{1}{2}k_{z2}) \\
k_{z3} &= hg(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y2}, z_n + \frac{1}{2}k_{z2}) \\
k_{y4} &= hf(t_{n+1}, y_n + k_{y3}, z_n + k_{z3}) \\
k_{z4} &= hg(t_{n+1}, y_n + k_{y3}, z_n + k_{z3}).
\end{aligned} \tag{7.6}$$

Ao utilizar-se as equações (7.6), primeiro calcula-se os *k's* ($k_{y1}, k_{z1}, k_{y2}, k_{z2}, \ldots$); em seguida, determina-se y_{n+1} e z_{n+1}. E, como cada k depende de x_n e y_n , eles devem ser calculados, novamente, a cada *n.* O valor *h*, comumente chamado de passo de integração, representa a diferença entre dois valores sucessivos do tempo *t.* Isto é, quando n = 0 o valor de t é t_0, quando n = 1 o valor de t é $t_0 + h$, e assim, sucessivamente.

Genericamente, para um sistema de equações diferenciais de várias equações de primeira ordem

$$\begin{cases}
y' = f(t, y, z, \ldots, x) \\
z' = g(t, y, z, \ldots, x) \\
\quad \ldots \\
x' = j(t, y, z, \ldots, x)
\end{cases} \tag{7.7}$$

com condições iniciais
$$\begin{aligned}
y(t_0) &= y_0 \\
z(t_0) &= z_0 \\
&\ldots \\
x(t_0) &= x_0
\end{aligned}$$

o método Runge-Kutta de quarta ordem pode ser descrito como a seguir:

$$
\begin{aligned}
y_{n+1} &= y_n + \frac{1}{6}(k_{y1} + 2k_{y2} + 2k_{y3} + k_{y4}) \\
z_{n+1} &= z_n + \frac{1}{6}(k_{z1} + 2k_{z2} + 2k_{z3} + k_{z4}) \\
&\cdots \\
x_{n+1} &= x_n + \frac{1}{6}(k_{x1} + 2k_{x2} + 2k_{x3} + k_{x4})
\end{aligned}
$$

onde (7.8)

$$
\begin{aligned}
k_{y1} &= hf(t_{n+1} - h, y_n, z_n, \ldots, x_n) \\
k_{z1} &= hg(t_{n+1} - h, y_n, z_n, \ldots, x_n) \\
&\cdots \\
k_{x1} &= hj(t_{n+1} - h, y_n, z_n, \ldots, x_n) \\
k_{y2} &= hf(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y1}, z_n + \frac{1}{2}k_{z1}, \ldots, x_n + \frac{1}{2}k_{x1}) \\
k_{z2} &= hg(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y1}, z_n + \frac{1}{2}k_{z1}, \ldots, x_n + \frac{1}{2}k_{x1}) \\
&\cdots \\
k_{x2} &= hj(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y1}, z_n + \frac{1}{2}k_{z1}, \ldots, x_n + \frac{1}{2}k_{x1}) \\
k_{y3} &= hf(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y2}, z_n + \frac{1}{2}k_{z2}, \ldots, x_n + \frac{1}{2}k_{x2}) \\
k_{z3} &= hg(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y2}, z_n + \frac{1}{2}k_{z2}, \ldots, x_n + \frac{1}{2}k_{x2}) \\
&\cdots \\
k_{x3} &= hj(t_{n+1} - \frac{1}{2}h, y_n + \frac{1}{2}k_{y2}, z_n + \frac{1}{2}k_{z2}, \ldots, x_n + \frac{1}{2}k_{x2}) \\
k_{y4} &= hf(t_{n+1}, y_n + k_{y3}, z_n + k_{z3}, \ldots, x_n + k_{x3}) \\
k_{z4} &= hg(t_{n+1}, y_n + k_{y3}, z_n + k_{z3}, \ldots, x_n + k_{x3}) \\
&\cdots \\
k_{x4} &= hj(t_{n+1}, y_n + k_{y3}, z_n + k_{z3}, \ldots, x_n + k_{x3})
\end{aligned}
$$

Vimos que os os diagramas de comportamento mecânico representam sistemas de equações diferenciais de primeira ordem e apresentamos o método de Runge-Kutta de quarta ordem para resolver estas equações. Mas, como traduzir um diagrama de

comportamento mecânico em um sistema de equações diferenciais de primeira ordem? Consideraremos, por um momento, um diagrama composto por um elemento do tipo função algébrica conectado a um elemento do tipo quantidade acumulada (figura 7.6-a). Uma vez que v atua como uma taxa de variação[7] para o elemento y, este diagrama pode ser visto, matematicamente, como uma representação para

$$\frac{dy}{dt} = v \quad \text{ou, em outra notação,} \quad y' = v \ .$$

Ou seja, estamos diante de uma equação diferencial na qual

$$y' = f(t, y) = v$$

com condição inicial $y(t_0) = y_0$.

[7] Ou seja, taxa na qual a quantidade y varia em cada unidade de tempo (t).

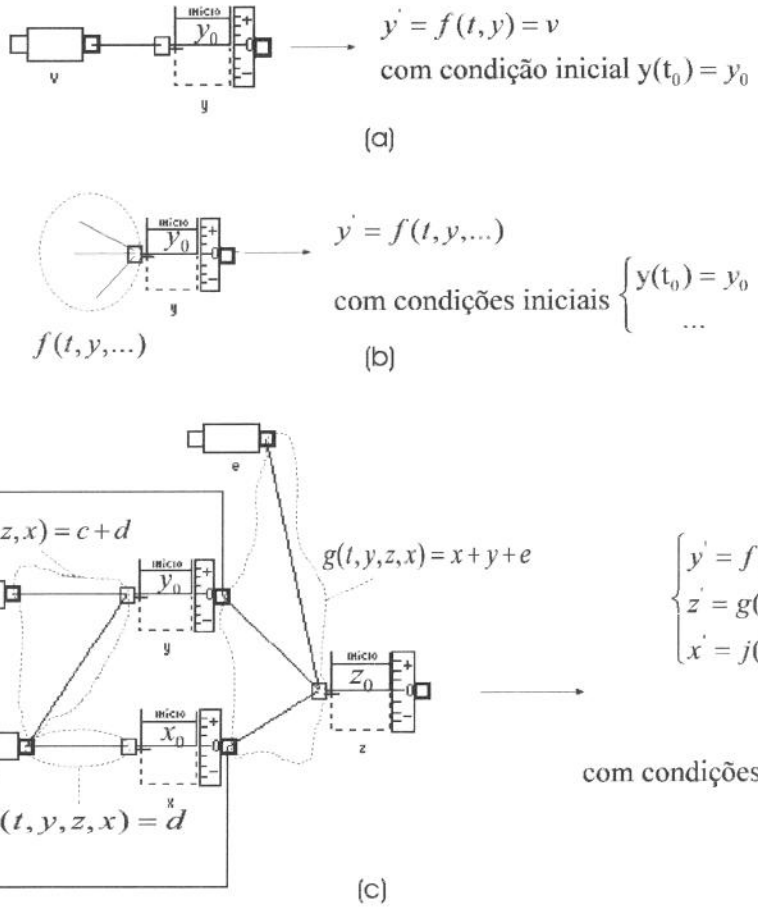


Fig. 7.6: Traduzindo o diagrama de comportamento mecânico em um sistema de equações diferenciais de primeira ordem

Genericamente, podemos entender o arranjo formado por um elemento do tipo quantidade acumulada (y) e suas respectivas conecções de entrada (figura 7.6-b) como sendo uma representação para a equação diferencial de primeira ordem

$$y' = f(t, y, \ldots)$$

com condições iniciais $\begin{matrix} y(t_0) = y_0 \\ \ldots \end{matrix}$.

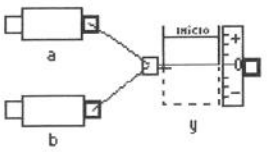
Aplicando as observações anteriores podemos traduzir o diagrama hipotético da figura 7.6-c no seguinte sistema de equações diferenciais de primeira ordem

$$\begin{aligned} y' &= f(t, y, z, x) = c + d \\ z' &= g(t, y, z, x) = x + y + e \\ x' &= j(t, y, z, x) = d \end{aligned}$$

$$\text{com condições iniciais} \quad \begin{aligned} y(t_0) &= y_0 \\ z(t_0) &= z_0 \\ x(t_0) &= x_0 \end{aligned} .$$

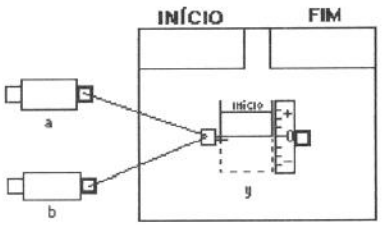
Até, agora, falamos da tradução de diagramas contendo elementos dos tipos quantidade acumulada, função algébrica e conexão. Mas, como traduzir diagramas contendo elementos do tipo duração de comportamento? A figura 7.7 sintetiza a influência de elementos do tipo duração do comportamento na tradução de diagramas de comportamento mecânico em equações diferenciais. Quando um elemento do tipo quantidade acumulada não sofre a influência de um elemento do tipo duração de comportamento (figura 7.7-a) a tradução se faz da forma como discutimos. Quando um elemento do tipo quantidade acumulada é envolvido por um elemento do tipo duração do comportamento (figura 7.7-b), tudo se passa como se a equação diferencial de primeira ordem não existisse nos momentos em que o comportamento ainda não se iniciou ou já terminou, ou seja,

$$y' = f(t, y) = 0.$$



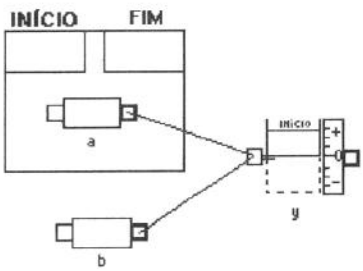
$$y' = f(t,y) = a+b, \text{ sempre}$$

(a)



$$y' = f(t,y) = \begin{cases} 0 & \text{se o comportamento não iniciou ou terminou} \\ a+b & \text{se o comportamento iniciou e não terminou} \end{cases}$$

(b)



$$y' = f(t,y) = \begin{cases} b & \text{se o comportamento não iniciou ou terminou} \\ a+b & \text{se o comportamento iniciou e não terminou} \end{cases}$$

(c)

Fig. 7.7: A influência do elemento duração de comportamento na tradução do diagrama de comportamento mecânico em um sistema de equações diferenciais de primeira ordem

Em contrapartida, isto é, nos momentos em que o comportamento iniciou e ainda não terminou tudo se passa como se o elemento duração de comportamento não existisse e a tradução pode ser feita da forma como discutimos, ou a exemplo da figura 7.7-b,

$$y' = f(t, y) = a + b.$$

Quando um dos elementos que atua como taxa de variação é envolvido por um elemento do tipo quantidade acumulada (figura 7.7-c) tudo se passa como se a taxa de variação não existisse nos momentos em que o comportamento ainda não se iniciou ou já terminou, ou no exemplo da figura 7.7-c,

$$y' = f(t, y) = b.$$

Caso contrário, é como se o elemento duração de comportamento não existisse. No exemplo da figura 7.7-c isto significa:

$$y' = f(t, y) = a + b.$$

7.5 - A EXECUÇÃO DOS MODELOS

Como dissemos anteriormente, projetamos o sistema de execução dos modelos pelo NEWTON para trabalhar segundo três estágios distintos que ocorrem, no tempo, um após o outro. O primeiro estágio compreende a compilação dos modelos abertos; o segundo estágio compreende o cálculo, pelo método Runge-Kutta de quarta ordem, das equações diferenciais definidas pelos comportamentos mecânicos dos corpos e a correspondente gravação, em um arquivo temporário, dos resultados das mesmas; o terceiro estágio consiste da leitura do arquivo temporário e a correspondente simulação dos modelos, ou seja animação do comportamento mecânico dos corpos e a apresentação dos resultados nos instrumentos que constarem dos modelos. O resultado do primeiro estágio, a compilação, é o código gerado, na linguagem OpenScript, para o cálculo das equações diferenciais (segundo estágio) e a para simulação dos modelos (terceiro estágio). Voltemos à figura 7.4 para acompanharmos o que ocorre com os objetos que compõem o NEWTON quando da execução dos modelos abertos. Quando o usuário decide executar os modelos, o Núcleo do NEWTON vai para a página "c". Esta página contém os *scripts* para compilação dos modelos; o resultado da compilação é a geração de código para as páginas "l "(página de cálculo dos modelos) e "s" (página que contém os *scripts* para simulação dos modelos). Em outras palavras, os *scripts* das páginas "l" e "s" do Núcleo do NEWTON são re-escritos a cada nova compilação e os seus conteúdos dependem da "lógica" dos modelos que estão sendo compilados.

Com o objetivo de observar o código gerado para as páginas de cálculo e simulação consideremos, por exemplo, o modelo representado na figura 7.8.

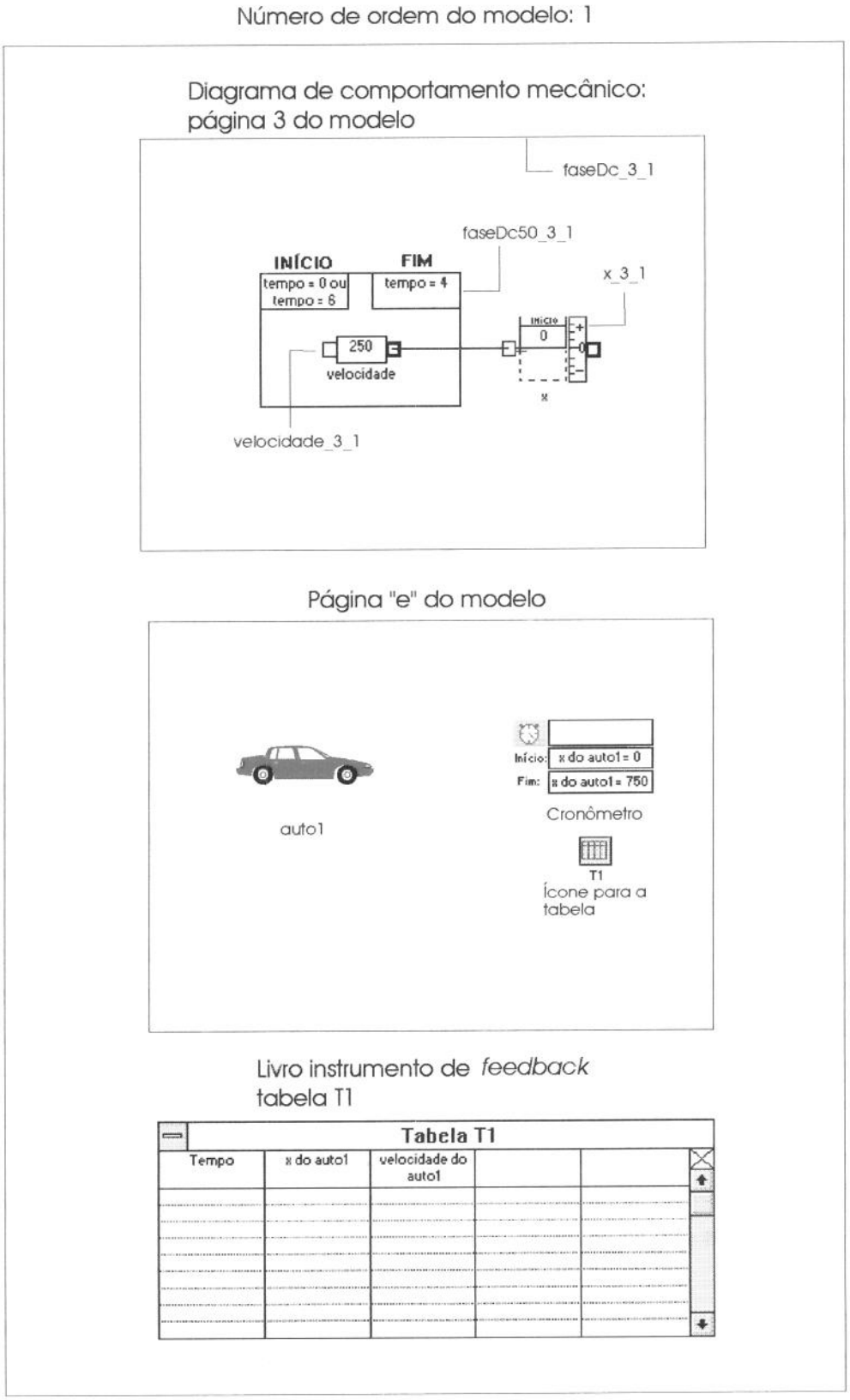


Fig. 7.8: Um modelo para exemplificar a geração de código pelo NEWTON

Trata-se do modelo de um automóvel que movimenta-se sob velocidade constante, de valor 250, no intervalo de tempo compreendido entre zero e quatro. No intervalo de tempo entre 4 e 6 o automóvel permanece em repouso, readquirindo velocidade de valor 250 após o tempo de valor 6. O modelo apresenta, ainda, dois instrumentos de *feedback:* uma tabela para exibir os valores da posição e da velocidade do automóvel (x e velocidade, respectivamente) e um cronômetro que mede o tempo que o automóvel leva para viajar da origem (posição zero) até a posição de valor 750. Abaixo apresentamos de maneira simplificada e em forma de um algoritmo o código gerado pelo NEWTON para a página de cálculo (página "1"). O algoritmo é apresentado em uma pseudo-linguagem que mistura um pouco da notação presente na linguagem OpenScript com um pouco da notação de uma linguagem como Pascal.

```
handler calcular

①       global h, tempoInícioExecução, tempoFimExecução,
              tempo, x_3_1, velocidade_3_1, faseDc50_3_1, faseDc_3_1;

②       criar arquivo temporário para armazenar o resultado das equações diferenciais;

③       tempo := tempoInícioExecução;
        x_3_1 := null; velocidade_3_1 := null;
        faseDc50_3_1 := 'finalizado'; faseDc_3_1 := 'finalizado';

        repita
                avaliarFaseDuraçõesComportamento ();

                inicializarQuatidadesAcumuladas ();

④               rungeKutta4 ();

                atribuirValoresFunçõesAlgébricas (tempo, x_3_1);

                escrever no arquivo temporário: x_3_1, velocidade_3_1;

                tempo := tempo + h

        até tempo > tempoFimExecução

⑤       fechar arquivo temporário que armazena os resultados das equações diferenciais;

end calcular

procedimento avaliarFaseDuraçõesComportamento ()

        global tempoInícioExecução, tempoFimExecução,
              tempo, x_3_1, velocidade_3_1, faseDc50_3_1, faseDc_3_1;

        condiçãoInício := igual (tempo, 0) ou igual (tempo, 6);
        condiçãoFim := igual (tempo, 4);

        condições
                quando faseDc50_3_1 = 'início':
                        se não condiçãoFim
                        então faseDc50_3_1 := 'execução' senão faseDc50_3_1 := 'finalizado'
                        fim se
```

```
                quando faseDc50_3_1 = 'execução' e condiçãoFim:
⑥                       faseDc50_3_1 := 'fim'

                quando faseDc50_3_1 = 'fim':
                        se não condiçãoInício
                        então faseDc50_3_1 := 'finalizado' senão faseDc50_3_1 := 'início'
                        fim se

                quando faseDc50_3_1 = 'finalizado' e condiçãoInício e não condiçãoFim:
                        faseDc50_3_1 := 'início'
        fim condições

        ...
⑦
        ...

fim avaliarFaseDuraçõesComportamento

procedimento inicializarQuantidadesAcumuladas ()

        global x_3_1, faseDc_3_1;
⑧
        se faseDc_3_1 = 'início' então x_3_1 := 0 fim se

fim inicializarQuantidadesAcumuladas

procedimento rungeKutta4 ()

        global h, tempo, x_3_1, velocidade_3_1, faseDc_3_1;

        se faseDc_3_1 = 'execução' ou faseDc_3_1 = 'fim'
        então   x_3_1K1 :=h * x_3_1F (tempo - h, x_3_1);
                x_3_1K2 :=h * x_3_1F (tempo - h/2, x_3_1 + x_3_1K1/2);
⑨               x_3_1K3 :=h * x_3_1F (tempo - h/2, x_3_1 + x_3_1K2/2);
                x_3_1K4 :=h * x_3_1F (tempo, x_3_1 + x_3_1K3);

                x_3_1 := x_3_1 + (x_3_1K1 + 2*x_3_1K2 + 2*x_3_1K3 + x_3_1K4)/6
        fim se

fim rungeKutta4

função x_3_1F (tempo, x_3_1)

        global faseDc50_3_1;

        resultado := 0;

⑩       se faseDc50_3_1 = 'execução' ou faseDc50_3_1 = 'fim'
        então resultado := resultado + velocidade_3_1(tempo, x_3_1)
        fim se

        retornar resultado

fim x_3_1F

    ...
⑪
    ...
```

O algoritmo mostra um *handler* de nome "calcular" que é responsável pelo cálculo dos modelos. Para fazer suas ações este *handler* utiliza um conjunto de procedimentos, funções e algumas variáveis criadas pelo compilador do NEWTON.

① O *handler* declara como global um conjunto variáveis, isto é, o valores destas variáveis podem ser utilizados em qualquer *script* do livro NEWTON. As variáveis **tempoInícioExecução** e **tempoFimExecução** representam, respectivamente, os valores do tempo de início e fim da simulação; estes valores são atribuídos pelo usuário e, neste contexto, apresentam-se como constantes. As variáveis tempo e h representam o tempo e o passo de integração. As demais variáveis estão, cada uma delas, associadas a um dos elementos presentes nos diagramas de comportamento mecânico. O nomes destas variáveis são conferidos de tal forma a identificar univocamente o elemento ao qual está associado. Assim, os nomes das variáveis associadas a elementos do tipo quantidades acumuladas e funções algébricas são compostos por três partes separadas com pelo caracter "sublinha": (1) o nome do elemento, (2) o número da página onde se encontra e (3) por um número de ordem associado pelo NEWTON a um modelo, uma vez que vários modelos podem ser abertos e executados ao mesmo tempo. Como exemplo, o compilador do NEWTON forma nomes tais como "x_3_1", "velocidade_3_1" (figura 7.8). Os nomes das variáveis associadas a elementos do tipo duração do comportamento são formados por um processo semelhante. No entanto, como os elementos do tipo duração de comportamento não possuem nome, o NEWTON lhes atribui um nome composto por duas partes: (1) uma parte fixa formada pela seqüência de caracteres "faseDc" e (2) uma parte composta pelo número de ordem do elemento duração de comportamento dentro da página. Com isto formam-se nomes de variáveis como "faseDc50_3_1" (figura 7.8). Um elemento do tipo duração de comportamento está associado a um nome de variável especial: por exemplo, na figura 7.8 o nome de variável faseDc_3_1 referencia o elemento duração de comportamento, geral ao diagrama, que tem as suas condições de início e fim associadas ao início e fim da execução; este elemento não é desenhado, mas está implícito no diagrama, uma vez que toda simulação possui um início e um fim.

Uma variável associada a um elemento do tipo quantidade acumulada ou função algébrica é utilizada para armazenar os valores que este elemento assume em cada tempo, durante o cálculo. Uma variável associada a um elemento do tipo duração do comportamento presta-se para sinalizar em que fase do ciclo de execução (ou ciclo de cálculo) este elemento se encontra. Um elemento do tipo duração de comportamento

apresenta fases distintas que podem se suceder em ciclo no decorrer do cálculo. Estas fases caracterizam os diversos estados que pode passar um elemento deste tipo: (1) fase "início" assinalada pelo momento no qual a condição de início do elemento avalia verdadeiramente; (2) fase "fim" demarcada pelo momento em que a condição de fim do elemento é avaliada como verdadeira; (3) fase "execução" que ocorre após a fase "início" mas antes da fase "fim" e na qual os diagramas no interior de uma duração de comportamento são executados (ou, se preferirmos, são calculados); (4) fase "finalizado", em que os diagramas no interior de um elemento duração de comportamento não são executados, e que ocorre após a fase fim (figura 7.9).

② O arquivo temporário para armazenamento de todos os resultados do cálculo dos diagramas é criado.

③ As variáveis que participam do processo de cálculo são inicializadas. Observamos que as variáveis que indicam em que fase se encontram os elementos do tipo duração do comportamento são inicializadas como "finalizado". Isto é, tudo se passa como se antes do início do cálculo, todos os elementos do tipo quantidade acumulada estivessem finalizados. Em outras palavras, todos os elementos estão esperando pela sua fase "início".

④ O laço **repita-até** promove o cálculo de cada uma das variáveis para cada valor de tempo desde o tempoInícioExecução até tempoFimExecução com variação igual ao passo de integração h. Em cada tempo, o algoritmo avalia em que fase se encontra cada um dos elementos do tipo duração do comportamento (avaliarDuraçõesComportamento ()). Dependendo da fase em que estão as durações de comportamento, algumas variáveis associadas a quantidades acumuladas são inicializadas com o valor do campo INÍCIO das mesmas (inicializarQuantidadesAcumuladas ()). Então, um procedimento que implementa o método Runge-Kutta de quarta ordem é chamado para resolver, para o valor de tempo corrente, as equações diferenciais representadas pelos diagramas dos modelos (rungeKutta4 ()). Como resultado da aplicação do método Runge-Kutta os valores das variáveis associadas a quantidades acumuladas são conhecidos. Resta, então, calcular os valores das funções algébricas presentes no modelo (procedimento atribuirValoresFunçõesAlgébricas ()) e gravar no arquivo temporário o valor das variáveis associadas aos elementos do tipo quantidade acumulada e função algébrica.

⑤ O arquivo temporário para armazenamento dos resultados do cálculo é, finalmente,

fechado.

⑥ Um elemento do tipo duração de comportamento apresenta um ciclo de execução (figura 7.9) que pode se repetir várias vezes ao longo do cálculo.

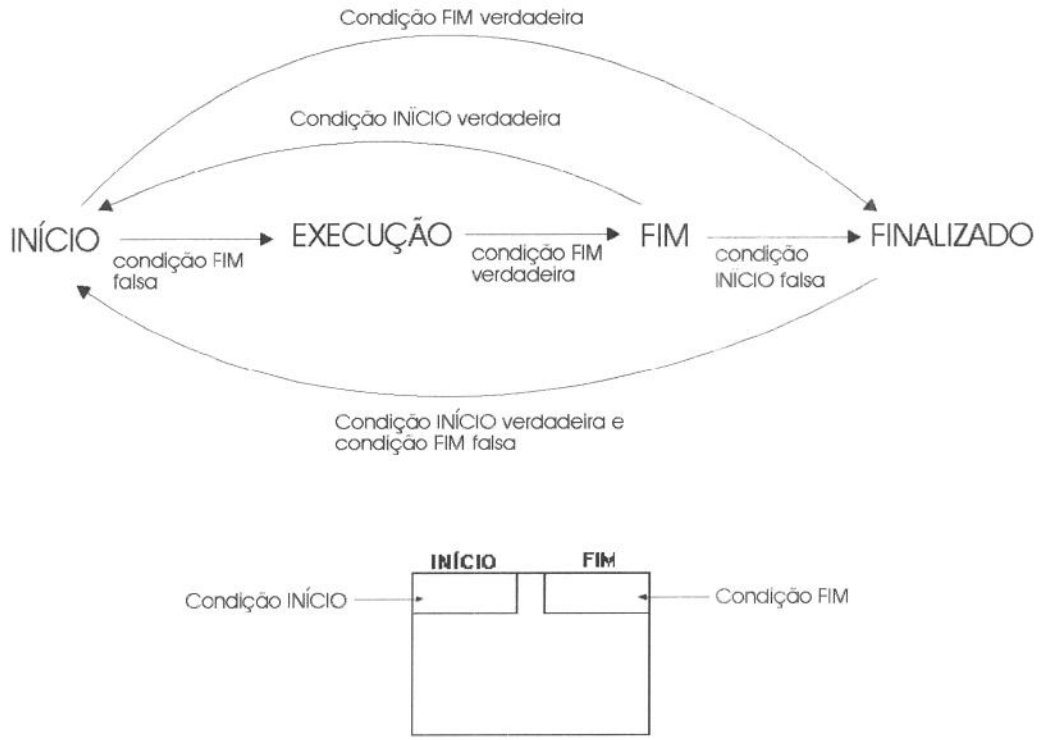


Fig. 7.9: O ciclo de execução de um elemento do tipo duração de comportamento

A figura 7.9 mostra que a fase "execução" sucederá a fase "início", apenas, se a condição de fim do elemento duração de comportamento for avaliada como falsa; caso contrário, a fase "finalizado" sucederá a fase "início". A fase "fim" sucederá a fase "execução" no momento em que a condição de fim do elemento duração do comportamento for avaliada como verdadeira. A fase "finalizado" sucederá a fase "fim" apenas se a condição de início for avaliada como falso; caso contrário a fase "início" sucederá a fase "fim", dando início à execução do elemento duração de comportamento sem que este tenha passado pela fase "finalizado". Estando o elemento duração de comportamento na fase "finalizado", a fase "início" só se

manifestará quando a condição de início do elemento duração do comportamento for avaliada verdadeira e, ao mesmo tempo, a condição de fim for avaliada como falsa. É exatamente este ciclo que está implementado pelo algoritmo para o elemento duração de comportamento que foi mostrado na figura 7.8. Observamos no algoritmo que o código gerado pelo NEWTON para a verificação da igualdade entre dois valores não faz uso do operador relacional de igualdade (=) que existe na linguagem OpenScript. Em seu lugar é utilizada uma função especial denominada igual (). Genericamente, o NEWTON necessita gerar código através das funções igual (), diferente (), maior () etc.. Estas funções pertencem ao livro de sistema "calculo.sbk" e retornam verdadeiro caso o valor de seu primeiro parâmetro seja, respectivamente, igual, diferente ou maior que o do segundo parâmetro; caso contrário, retorna falso. Tais funções representam uma solução para os problemas de comparação de valores que surgem do fato das grandezas da Mecânica terem uma natureza contínua enquanto que o NEWTON as representa de forma discreta (veja seção 6.10 do capítulo 6).

⑦ Códigos similares ao apresentado em ⑥ são gerados pelo compilador do NEWTON para a avaliação da fase em que encontram outros elementos do tipo duração de comportamento que componham os modelos. No modelo exemplificado na figura 7.8, pelo menos um outro elemento existe: o elemento duração de comportamento implícito e geral ao diagrama, cujas condições de início e fim demarcam o início e o fim do cálculo.

⑧ A variável x_3_1, que está associada à quantidade acumulada "x", é inicializada com o valor zero (veja figura 7.8) se o elemento duração de comportamento que a envolve estiver na fase "início". Especificamente neste caso, não há elemento duração de comportamento desenhado de tal forma a envolver a quantidade acumulada "x". O código se refere à fase do elemento duração de comportamento (faseDc_3_1), implícito e geral ao diagrama, que tem as suas condições de início e fim associadas ao início e fim do cálculo.

⑨ O procedimento rungeKutta4 () implementa a resolução do sistema de equações diferenciais representadas pelos diagramas de comportamento mecânico dos modelos (veja equações 7.8 da seção 7.4). A função x_3_1F () é a função que atua como taxa de variação para a quantidade acumulada "x".

⑩ A função taxa de variação x_3_1F () sofre a influência do elemento duração de

comportamento cuja variável faseDc50_3_1 está associada (veja figura 7.7-c, na seção 7.4). Assim a taxa de variação terá o valor zero (ou seja, não haverá taxa de variação) quando o elemento duração de comportamento que a envolve não estiver na fase "execução" e na fase "fim". Caso contrário, o valor da taxa de variação será o somatório de todos os elementos que se conectam na entrada da quantidade acumulada "x"; neste caso existe, apenas, um elemento do tipo função algébrica.

⑪ O algoritmo conta, ainda, com o procedimentos atribuirValoresFunçõesAlgébricas () que atribui valor às variáveis associadas às funções algébricas presentes nos modelos e a função velocidade_3_1() que retorna o valor da função algébrica associada à variável velocidade_3_1.

O código gerado pelo compilador do NEWTON para a página de simulação (página "s") para o modelo da figura 7.8 é mostrado, de forma simplificada, pelo algoritmo a seguir. O algoritmo apresenta três *handlers* : (1) o primeiro responde à mensagem *enterPage* que é enviada pelo ToolBook quando uma página é aberta; (2) o segundo *handler* responde à mensagem *leavePage* que é enviada pelo ToolBook quando uma página é fechada; (3) o terceiro *handler* responde à mensagem *idle* que é enviada pelo ToolBook sempre quando ele não está efetuando nenhuma ação.

```
handler enterPage

        global simulação, tempo, tempoInícioExecução, h

        simulação := 'nãoFim';
        tempo := tempoInícioExecução;

①       abre arquivo temporário que contém os resultados das equações diferenciais;

        inicializa cronômetro de número 475 da página de simulação;
        limpar tabela da página 0 do livro 't1.ins';

fim enterPage

handler leavePage

②       fechar arquivo temporário que contém os resultados das equações diferenciais;

fim leavePage

handler idle

        global simulação, tempo, h, x_3_1, velociadade_3_1

        condições
                quando simulação = 'nãoFim'
```

```
                ler arquivo temporário próxima unidade de tempo;

                se não fim de arquivo
                        colocar os valores lidos em x_3_1 e velocidade_3_1;

                        x do grupo de número 467 da página 's' := x_3_1;

③                       enviar mensagem pulso (igual (x_3_1, 0), igual (x_3_1, 750)) ao
                        cronômetro de número 475 da página de simulação do modelo;

                        enviar à tabela do livro 't1.ins' os valores x_3_1 e velocidade_3_1;

                        tempo := tempo + h

                senão
                        simulação := 'fim'
                fim if

        quando simulação = 'fim'
                voltar à página de edição de corpos;

    fim condições

④   avaliarCondiçõesParada ()

fim idle

...

...
```

① O *handler* enterPage é responsável por inicializar as variáveis, inicializar os valores registrados pelos instrumentos instalados e abrir o arquivo temporário que contém o resultado do cálculo do modelo. A variável denominada simulação é usada para definir o estado da simulação e pode receber um entre três possíveis valores: "nãoFim", que indica que a simulação está em curso; "fim", que indica que a simulação chegou ao fim ou foi cancelada por intervenção do usuário através do botão Cancelar da barra de ícones; "suspensa", que indica que a simulação foi suspensa por intervenção do usuário no respectivo botão da barra de ícones ou porque uma condição de parada previamente estabelecida foi encontrada. O valor inicial da variável simulação é "nãoFim", o que indica que a simulação estará em curso no momento que a primeira mensagem *idle* for enviada à página de simulação.

② O *handler* leavePage, apenas, fecha o arquivo temporário que contém os resultados do cálculo. Este *handler* é executado após a simulação, no momento em que o NEWTON se prepara para abandonar a página de simulação e voltar à página de edição de corpos.

③ O *handler* idle é o que, de fato, controla a simulação. Mensagens *idle* são enviadas para a página de simulação momento a momento pelo ToolBook. Cada resposta dada pelo *handler* idle corresponde a uma unidade de tempo. A estrutura Condições-fim discrimina duas possibilidades: (1) quando a simulação chegou ao fim (simulação = "fim") a ação a ser tomada é voltar à página de edição de corpos e (2) quando a simulação está em curso (simulação = "nãoFim") o arquivo que contém o resultado do cálculo é lido e os valores lidos são atribuídos às variáveis associadas às funções algébricas e quantidades acumuladas. A cada uma das propriedades padrão de um corpo, x, y ou teta (veja seção 5.4 do capítulo 5), se estiverem sendo referenciadas no modelo, será devidamente atribuído um valor. No exemplo que estamos tratando, o valor da variável x_3_1 está sendo atribuído à propriedade padrão x de um corpo que é reconhecido pelo NEWTON como o grupo de número 467 da página de simulação do modelo (página "s"). Aos instrumentos são enviados valores ou mensagens. No exemplo, os valores das variáveis x_3_1 e velocidade_3_1 são enviados ao instrumento de *feedback* tabela e a mensagem de nome pulso, que possui dois parâmetros, é enviada ao cronômetro: os parâmetros indicam através de valores booleanos (falso, verdadeiro) o momento em que o cronômetro deve ser inicializado (primeiro parâmetro) e finalizado (segundo parâmetro). Tanto os instrumentos de *feedback* quanto os corpos possuem *handlers* para responder aos valores atribuídos pelo *handler* idle. Como exemplo, um cronômetro contém um *handler* para responder à mensagem pulso enviada pelo *handler* idle o qual incrementa em h (passo de integração) o valor registrado pelo cronômetro caso ele tenha sido inicializado e ainda não tenha sido finalizado. E, um corpo contém *handlers* para responder à atribuição de valores às propriedades padrão x, y e teta. Por exemplo, a atribuição de um valor à propriedade x de um corpo faz com que este seja movimentado para a posição de abscissa igual ao valor atribuído.

④ Os usuários podem estabelecer condições de parada para a simulação, por exemplo, suspender a execução da simulação quando a posição (x) do automóvel ficar igual a 150 (veja seções 5.5 do capítulo 5 ou seção 6.6 do capítulo 6). O procedimento avaliarCondiçõesParada () verifica a cada instante se uma determinada condição de parada foi atingida. Caso seja atingida, este procedimento altera o valor da variável global simulação para o valor "suspensa". Quando isto ocorre o *handler* idle deixa de executar as suas ações. A execução pode ser reiniciada por ação do usuário sobre o botão parar/reiniciar existente na barra de ícones, o que faz com que o valor da variável simulação volte a ser "nãoFim".

Capítulo 8

Discussão, conclusões e perspectivas

O objetivo deste capítulo é discutir uma série de pontos de caráter geral relacionados ao *design* de ambientes para modelagem em um contexto educacional no que se refere ao trabalho desenvolvido. Paralelamente, sublinhamos perspectivas de trabalhos futuros.

Tópicos deste capítulo:

Tópico		*Página*
8.1 -	Síntese	181
8.2 -	O que há nas proximidades do que fizemos	181
8.3 -	As estratégias propostas e a modelagem em outros contextos	187
8.4 -	Considerações sobre o protótipo	190
8.5 -	O quê fizemos e como isto pode ajudar no *design* de ambientes para modelagem	192

8.1 - SÍNTESE

Este trabalho trata a modelagem computacional como uma atividade na qual um aprendiz pode usar o computador para expressar um fenômeno através de um modelo com o objetivo subseqüente de explorar as conseqüências do modelo e avaliar, através de um *feedback*, não somente o modelo construído, mas o seu próprio conhecimento do fenômeno.

A literatura tem mostrado consideráveis evidências dos benefícios educacionais advindos dos ambientes para modelagem. A atividade de modelar tem sido defendida como um rico ambiente no qual o usuário pode desenvolver habilidades na resolução de problemas, apresentação e comunicação do entendimento de uma pessoa sobre um determinado domínio. Contudo a literatura tem mostrado, também, dificuldades do aprendiz em dominar os ambientes para modelagem computacionais. Um dos principais problemas é o da relação entre o nível de controle embutido no *design* de um sistema e o esforço exigido do usuário para manipulá-lo (Goodyear, 1987). Ross (1986) refere-se às dificuldades dos usuários de ambientes para modelagem especialmente nas fases de criação e avaliação dos resultados de um modelo. Baranauskas e Oliveira (1995) entendem que o problema dos ambientes para modelagem atuais está na generalidade de suas primitivas, as quais apresentam um nível de granulosidade diferente do nível de granulosidade das primitivas conceituais utilizadas para descrever um domínio.

Nós propusemos um ambiente para modelagem orientado a um domínio (AMOD) como um equilíbrio entre o nível de controle colocado nas mãos do usuário e o esforço demandado (do usuário) para apropriação de conceitos de computação. Enunciamos um conjunto de princípios e os organizamos de modo a formar um conjunto de estratégias para o *design* de um AMOD. Desenvolvemos um protótipo para a modelagem Mecânica com o objetivo de experimentar as estratégias propostas, bem como para viabilizar análises futuras sobre o *design* de ambientes para modelagem.

8.2 - O QUE HÁ NAS PROXIMIDADES DO QUE FIZEMOS

As estratégias de *design* de ambientes para modelagem apresentadas neste trabalho compreendem três etapas: (1) caracterização do domínio a partir de alguns dos elementos que o compõem, (2) identificação de um conjunto de conceitos atômicos que possa ser

utilizado para descrever os elementos do domínio e (3) desenvolvimento do *design* de cada conceito atômico (primitiva) e criação dos elementos de infra-estrutura do ambiente (figura 8.1). O objetivo da presente seção é analisar as estratégias definidas sob a perspectiva da influência que elas sofrem de diversos fatores que vão desde a definição da extensão (tamanho) do domínio até as implicações do desenvolvimento da tecnologia de interface e do entendimento da interação Homem-Computador.

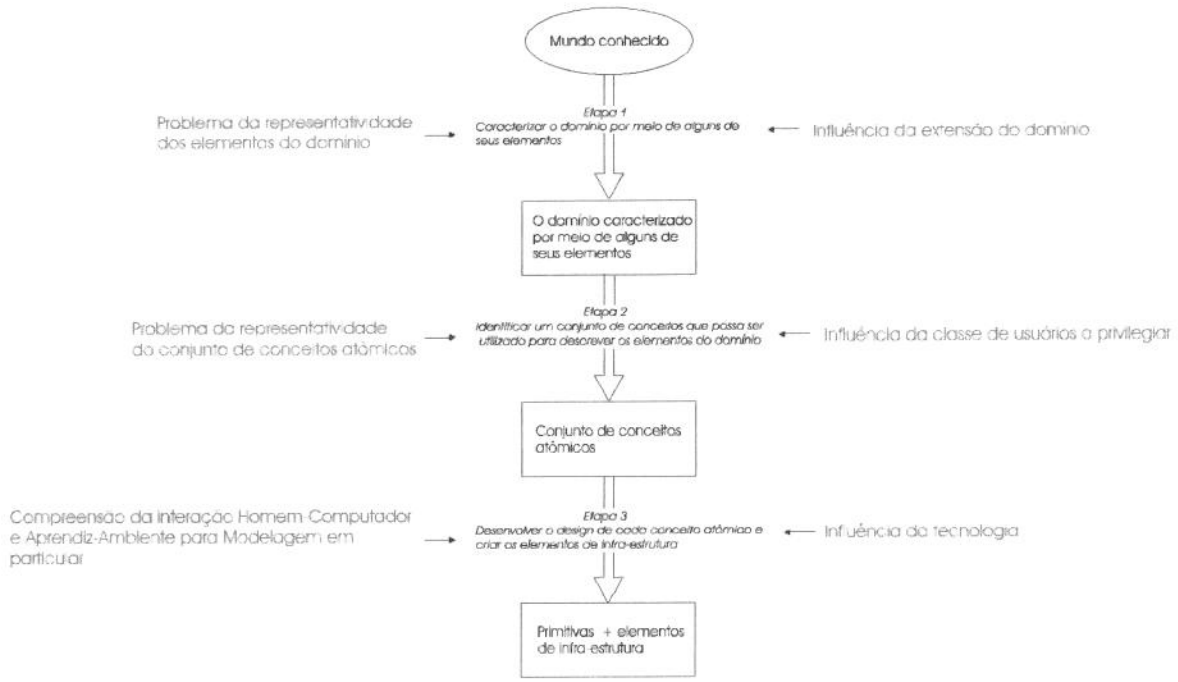


Fig. 8.1 - Influência de diversos fatores nas estratégias de *design* definidas

1 - **A influência da extensão do domínio.** Seguindo as estratégias discutidas, quando um *designer* se propõe desenvolver um AMOD a sua primeira tarefa consiste em definir o domínio de interesse, isto é, demarcar as linhas imaginárias que envolvem o domínio e delimitam os seus elementos (fenômenos, objetos) do resto do mundo conhecido. É bem provável que domínios grandes fascinem mais que os pequenos pela possibilidade de se estabelecer ambientes para modelagem amplos, aplicáveis em diversas situações. Contudo, devemos esperar que quanto maior for um domínio, menor será a granulosidade média das primitivas do respectivo ambiente para

modelagem que podemos obter aplicando as estratégias propostas. Afinal, se falamos do domínio das edificações precisaremos, talvez, de primitivas que representem tijolos, cimento, madeira etc. para compor modelos dentro deste domínio. Em contrapartida, se falarmos do domínio de todos os objetos concretos visíveis, talvez precisemos dos átomos como primitivas para descrevê-los.

A questão da extensão do domínio nos faz refletir sobre dois aspectos inter-relacionados. O primeiro aspecto diz respeito à necessidade fundamental de se limitar um domínio a um tamanho adequado como requisito de sucesso do ambiente para modelagem a ser desenvolvido. De nada adianta possuir um ambiente para modelagem que tudo é capaz, mas cujas primitivas se encontram tão longe de um fenômeno a ser modelado que obrigue o aprendiz a entender e modelar primeiro as estruturas dos átomos, segundo a das moléculas e suas ligações, terceiro as estruturas físico-químicas dos materiais para, enfim (e talvez), descrever um modelo para uma casa de alvenaria. É claro que isto é um exagero mas, guardadas as proporções admissíveis, o problema ainda existe e, em essência, é o mesmo.

O segundo aspecto é que não há garantias de que o aprendiz, ou qualquer outra pessoa, consiga descrever o modelo para uma casa de alvenaria a partir de um ambiente para modelagem que possua como primitivas representações para os átomos. Embora isto seja um problema intrigante e desafiante, é possível que nos dias atuais ninguém consiga resolvê-lo.

Sabemos que limitar o domínio a um certo tamanho é fundamental para o sucesso de um ambiente para modelagem cuja aplicação prevista está voltada para o contexto educacional. Porém não sabemos dizer em que nível esse limite deve ser estabelecido. Neste aspecto, entendemos que mais pesquisa precisa ser conduzida para que se possa entender melhor o problema e sua solução.

2 - **O problema da representatividade dos elementos do domínio.** Pelas estratégias propostas, um domínio é caracterizado por meio de alguns de seus próprios elementos (fenômenos, objetos). Quanto mais representativos forem estes elementos das diversas faces e nuanças do domínio, mais bem caracterizado ele fica sob a ótica do senso comum sobre o que ele envolve. Divergências à parte, mesmo para o que o senso comum dispõe sobre o que envolve um domínio, é difícil afirmar que um determinado conjunto de elementos o represente fielmente. É possível que para

determinados domínios existam métodos matemáticos capazes de assegurar que um conjunto de elementos encerre em si todos os germes capazes de gerar a diversidade do domínio. Uma possível solução seria avaliar um a um os elementos do domínio. Na prática, as proporções infinitas e infinitesimais de alguns domínios tornam não factível a tarefa de avaliar cada um de seus elementos. Esta é a questão que estamos endereçando com o nome de problema da representatividade dos elementos do domínio.

No caso do *design* do NEWTON tentamos resolver o referido problema de uma forma tipicamente tecnológica. Estudamos o domínio da mecânica e propusemos um conjunto de elementos que o representasse. Em seguida examinamos um número razoável de elementos dentro do domínio até nos convencermos de que o conjunto proposto o representava satisfatoriamente. Esta solução está longe de ser matemática, está distante de ser uma solução exata, perfeita. Mas entre ficar o resto da vida analisando a diversidade de fenômenos da Mecânica para buscar uma solução perfeita para o problema (que pode não existir com base na ciência atual) preferimos uma solução tipicamente de engenharia: aproximada, ainda que não seja exata.

Não discordamos de que é importante a condução de pesquisas no sentido de resolver o problema da representatividade dos elementos de um domínio. No entanto, neste momento de carência de ambientes para modelagem, argumentamos que pouco resultado prático pode trazer do ponto vista de *design* de ambientes para modelagem em um contexto educacional, a certeza de que um determinado conjunto de elementos representa fielmente um domínio. Afinal, se um conjunto de elementos representa, apenas um subdomínio do domínio de interesse, do ponto de vista educacional ainda pode ser muito interessante a experiência de possuir ambientes para modelagem para construir modelos neste subdomínio. Ademais, o que são realmente as linhas que demarcam um domínio senão limites arbitrariamente estabelecidos pelo homem: alguns metros para lá ou para cá podem, na prática, não fazer diferença.

3 - **O problema da representatividade do conjunto de conceitos atômicos.** Pelas estratégias propostas, um conjunto de conceitos atômicos deve ser escolhido para descrever os elementos usados na representação do domínio. Este conjunto é escolhido pela sua virtude de representar um equilíbrio entre o modelo conceitual

mínimo e a máxima granulosidade de seus conceitos atômicos (seção 4.3 do capítulo 4). Conceitos atômicos são derivados dos elementos escolhidos para representar um domínio. Se os elementos escolhidos para representar um domínio o representam fielmente, então o conjunto de conceitos atômicos deles derivado poderá ser usado para descrever qualquer elemento do domínio. Caso contrário, o conjunto de conceitos atômicos escolhido descreverá apenas um subdomínio dentro do domínio de interesse. Isto é o que estamos denominando de problema da representatividade dos conceitos atômicos. Na verdade este problema não é independente. Ele é resultante do problema da representatividade dos elementos do domínio (item 2 desta seção). Estamos explicitando-o pela sua importância e implicação: a menor ou maior abrangência, em relação ao domínio de interesse, do decorrente ambiente para modelagem a ser desenvolvido. Resolvendo o problema da representatividade dos elementos do domínio estaremos resolvendo este problema.

4 - **A influência da classe de usuários a privilegiar.** Uma classe de usuários define um perfil médio de usuário com uma cultura e conhecimento adquirido. Pelas estratégias propostas o *designer* deve, preferencialmente, escolher conceitos atômicos já assimilados pela classe de usuários que se quer privilegiar no *design*. E isto pode ser traduzido no seguinte requerimento crítico: como determinar se um conceito atômico é adequado à classe de usuários que se deseja privilegiar no *design?*

Conhecer as estruturas mentais atuais da classe a privilegiar pode ajudar em muito no *design* de ambientes para modelagem efetivos para esta classe. Entendemos que qualquer contribuição de pesquisa neste sentido é bem vinda. Entretanto, se determinado conceito atômico não é conhecido a *priori* por elementos da classe a privilegiar, isso não implica na impossibilidade de uso do decorrente ambiente para modelagem desenvolvido. É possível que usuários nesta condição devam desenvolver esforços extras para assimilar os conceitos que não possuam, paralelamente ao trabalho de modelagem que estiverem realizando.

5 - **Influência da tecnologia.** A evolução da tecnologia, especialmente a de interface, pode possibilitar *design* de conceitos atômicos e de elementos de infra-estrutura que se aproximem cada vez mais dos objetos do domínio de interesse. Isto ajuda a reduzir a distância entre o usuário e o domínio. Também, auxilia na promoção do engajamento direto do usuário com os objetos do domínio. Vimos na seção 5.4 do capítulo 5 que se dispuséssemos da tecnologia de realidade virtual para implementar

o protótipo NEWTON a solução seria bem diferente da que obtivemos e, certamente, teríamos alguns ganhos em relação a tecnologia utilizada. No entanto, concordamos com Draper e Norman (1986) e achamos que independentemente das limitações da tecnologia atual muito pode ser feito se compreendermos melhor a interação Homem-Computador; especialmente precisamos compreender melhor a interação Aprendiz-Ambiente para Modelagem.

6 - Compreensão da interação Homem-Computador e Aprendiz-Ambiente para Modelagem em particular.

> As pessoas são muito diferentes de computadores. Esta observação não é novidade, mas se pessoas usam computadores existe uma zona de acomodação mútua entre eles. Pessoas são adaptáveis e são capazes de assimilar toda a carga de acomodação a um artefato, mas *designers* habilidosos podem fazer com que esta carga seja diminuída por adequar o artefato a seus usuários (Draper e Norman, 1986, p. 1).

A interação Homem-Computador se dá nesta zona de acomodação mútua. O entendimento da interação Homem-Computador envolve dois campos complexos: Psicologia e Ciência da Computação (Norman, 1986). O desenvolvimento de um artefato computacional demanda não só o conhecimento do domínio ao qual ele se refere mas também o conhecimento de *design,* de programação, da tecnologia, das pessoas e seus princípios de processamento mental e comunicação.

Foi possível formular as estratégias de *design* que discutimos graças aos recentes estudos sobre a interação Homem-Computador e, em especial, à teoria da ação de Norman (veja seção 4.4 do capítulo 4). Contudo, o que cria uma boa relação entre pessoas e das pessoas com o computador ainda carece de boa fundamentação (Norman, 1986, p. 59). Nós ainda não entendemos nem mesmo as características globais que determinam a essência da interação. Argumentamos que é necessário mais pesquisa no sentido de compreender os princípios que guiam a ação do Homem sobre os artefatos da tecnologia.

Sabemos que ambientes computacionais para modelagem devem prover meios para que o aprendiz possa descrever, executar e avaliar os resultados de um modelo estabelecido. Por ser recente a aplicação da modelagem computacional em um contexto educacional temos pouca informação sobre a interação Aprendiz-Ambiente para Modelagem. Certamente pesquisas no campo da interação Aprendiz-Ambiente

para Modelagem em muito poderá contribuir para o *design* destes ambientes.

8.3 - AS ESTRATÉGIAS PROPOSTAS E A MODELAGEM EM OUTROS CONTEXTOS

As estratégias de *design* discutidas no capítulo 4 foram concebidas para serem aplicadas no desenvolvimento de ambientes para modelagem voltados para aplicação em um contexto educacional. Salvo algumas possíveis variações entendemos que, em essência, estas mesmas estratégias possam ser aplicadas, por exemplo, ao *design* de ambientes para modelagem em outros contextos. Dependendo do contexto, é possível que tais variações ocorram, principalmente, na menor ou maior influência da classe de usuários a privilegiar e no nível de importância dada ao *design* de mecanismos de *feedback* e de expressão dos modelos. Entendemos que estes aspectos são imprescindíveis no *design* de ambientes para modelagem para aplicação em um contexto educacional. Mesmo no caso de outros contextos é possível que os mecanismos de *feedback* afetem, por exemplo, a produtividade do usuário e o modelo mental que este faz do sistema. Entretanto, não temos a pretensão de analisar estas variações. O que desejamos fazer é refletir sobre as implicações da possibilidade de utilizar as estratégias propostas, com ou sem variações, ao *design* de ambientes para modelagem orientados a um domínio específico em um contexto diferente do educacional.

Para isto, consideremos nas discussões a seguir, por exemplo, um contexto típico de processamento empresarial (comercial) de dados. É bem provável que as discussões a seguir se apliquem, também, a outros contextos que não o de processamento comercial de dados. Contudo, limitaremos nossas discussões a este contexto. Na força de trabalho de processamento dos dados de uma empresa é possível identificar pelo menos dois grupos de pessoas além dos gerentes: o grupo dos desenvolvedores de aplicações[1] para processar os dados e o grupo dos usuários[2] destas aplicações (Mills *et alli*, 1986). Tipicamente os desenvolvedores de aplicações analisam os requisitos do usuário, projetam uma solução para o sistema[3] de interesse do usuário e implementam a parte automatizada do sistema

[1] Em algumas empresas, hoje em dia, uma parte ou a totalidade das atividades de desenvolvimento são repassadas a terceiros. Quando isto ocorre o grupo a que estamos nos referindo continua existindo, porém, fora da empresa.
[2] Estamos considerando como usuários de aplicações, também, os operadores de computadores.
[3] Existem diversas definições para o termo sistema, contudo, neste texto empregamos a palavra sistema no mesmo sentido descrito por Pressman (1987). Para Pressman um sistema envolve pessoas, métodos e máquinas que interagem entre si para atingir um objetivo específico.

por meio de uma linguagem de programação de propósito geral[4]. Em contrapartida, cabe ao grupo de usuários executar as aplicações para processar os dados da empresa. Em geral, o trabalho de um usuário se resume em acionar aplicativos e preencher um ou mais parâmetros de entrada solicitados por estes.

Consideremos, também, a possibilidade de poder aplicar as estratégias discutidas para estabelecer o *design* de um ambiente para modelagem orientado a um sistema[5] específico. Se assim fizermos, esperamos obter um ambiente para modelagem cujas primitivas se aproximam das estruturas conceituais do sistema de interesse do usuário. No decorrer deste trabalho dissemos que um domínio englobava diversos fenômenos ou, genericamente, dizíamos: elementos. No contexto dos sistemas empresariais os fenômenos, em geral, são serviços de processamento de dados que necessitam ser realizados.

Similarmente ao que acontece com os ambientes para modelagem em um contexto educacional, esperamos que os usuários de um ambiente para modelagem orientado a um sistema possam compor modelos que lhes permitam realizar atividades relacionadas ao seu trabalho. De certa forma as primitivas existentes em planilhas eletrônicas como Excel se aproximam das estruturas conceituais necessárias à execução de determinados trabalhos dentro de alguns sistemas de uma empresa. Excel pode ser utilizada para compor modelos financeiros bem à maneira com que um usuário trabalha as suas planilhas de papel. Também as operações fundamentais de projeção, seleção, produto cartesiano, união e diferença de conjuntos (Korth e Silberschatz, 1989, pp. 56-64) disponíveis em certos *software* (QMF e SqlPlus por exemplo) para acesso a sistemas gerenciadores de banco de dados através de linguagens como SQL ou Query By Example, constituem primitivas que se aproximam das estruturas conceituais necessárias à execução de determinados trabalhos dentro de alguns sistemas de uma empresa. Assim, não nos surpreendemos ao ver estes *software* serem indicados por seus fabricantes para serem utilizadas por usuários novatos na utilização de computadores (IBM, 1992, p. 1; Microsoft, 1994; Oracle, 1986). Na verdade Excel, QMF e SqlPlus lembram mas não refletem exatamente o que estamos considerando ambiente para modelagem orientado a um sistema. Excel, QMF e SqlPlus se aproximam do que podemos chamar de ambientes para modelagem de

[4] Rigorosamente isto não é verdade. Muitas das ferramentas utilizadas para desenvolvimento nos dias atuais não são o que se pode chamar de linguagem de programação de propósito geral. No entanto, para a reflexão que estamos fazendo nesta seção este aspecto não é relevante.

[5] Estamos usando o termo sistema de forma correlata a domínio, o qual nos referimos intensamente ao longo deste trabalho. No contexto que estamos tratando entendemos ser mais adequado usar o termo sistema, uma vez que este termo está consagrado pela literatura de Engenharia de Software.

propósito geral e não se comprometem com uma ou outra atividade que deve ser feita em um determinado sistema. Falamos de ambientes para modelagem voltados para modelagem especializada em um determinado sistema. Falamos de algo análogo ao AMOD NEWTON que é específico para o desenvolvimento de modelos dentro do domínio da Mecânica.

Argumentamos que ambientes para modelagem orientados a um sistema podem suprir uma série de requisitos dos usuários dentro das empresas. Mais ainda, acreditamos que ambientes para modelagem orientados a um sistema podem atuar como pivô de uma grande transformação[6]: a de trocar o papel do usuário dentro das empresas de operador de aplicativos para desenvolvedor de suas próprias soluções de processamento de dados. Se isto de fato começar a ocorrer, de forma gradativa haverá cada vez menos espaço para os desenvolvedores de aplicações (analistas de sistemas, projetistas e programadores) dentro das empresas. Não estamos afirmando que será o fim das atividades de análise, projeto e programação. Afinal, os próprios ambientes para modelagem orientados a um sistema necessitam de profissionais que sejam capazes de desenvolvê-los. O que argumentamos é que poderá haver um deslocamento na forma de praticar as análise, projeto e programação (veja seção 4.6 do capítulo 4).

Aqui cabe um esclarecimento: uma coisa é a análise de um sistema, projeto e programação (implementação[7]) de um *software;* outra coisa bem diferente são as atividades de análise de um objeto/fenômeno, projeto e "programação" de um modelo. Quando falamos de mudanças na análise e no projeto queremos nos referir à análise e projeto de ambientes para modelagem. Descobrir primitivas de um domínio ou sistema de tal modo que elas permitam construir modelos dentro de um domínio ou sistema é algo inteiramente diferente do que propõem os métodos tradicionais de análise e projeto. Quando falamos de mudanças na programação estamos nos referindo, especialmente, à construção de um modelo. A atividade de projetar e implementar um modelo em ambientes para modelagem como os que estamos tratando é algo tão diferente da atividade tradicional de programar que já não está claro se devemos continuar empregando a palavra programar.

---

[6] Em geral, todas as tentativas de previsão do futuro parecem se tornar ridículas com o passar do tempo. Não temos a pretensão de descrever o futuro. Apenas queremos esboçar algumas tênues fronteiras dentro das quais acreditamos que o futuro possa estar.

[7] Nesta seção estamos usando a palavra programação nos referindo ao que, em geral, na literatura de Engenharia de Software é caracterizado como implementação ou implantação. Independente da conotação distinta destas palavras, nesta seção programação está significando tanto as atividades de codificação quanto de testes.

8.4 - CONSIDERAÇÕES SOBRE O PROTÓTIPO

As estratégias discutidas neste trabalho se aplicam ao *design* de qualquer ambiente para modelagem. O capítulo 5 trata a aplicação destas estratégias ao *design* de um ambiente para modelagem em Mecânica, o NEWTON, o protótipo que desenvolvemos para verificar a aplicação das estratégias discutidas. Acreditamos que o NEWTON servirá, também, como **objeto-para-pensar-sobre** através do qual se viabilizarão análises futuras sobre o *design* de ambientes para modelagem em um contexto educacional e, em especial, ambientes para modelagem em Mecânica. Ao longo deste trabalho e principalmente no capítulo 5 tecemos os comentários que desejávamos sobre a aplicação das estratégias ao *design* do NEWTON. Nesta seção não pretendemos estender tais comentários, muito menos repeti-los. No entanto, gostaríamos de discutir a contribuição que o NEWTON pode dar à aprendizagem da Mecânica sob a ótica dos ambientes para modelagem. Gostaríamos de ver o NEWTON como se fosse um produto comercial, frente a frente com outros produtos disponíveis.

Consideremos, primeiramente dois ambientes não computacionais para modelagem em Mecânica bem semelhantes: o ambiente do quadro-negro e giz e o ambiente dos livros de Física, caderno e lápis. Estes ambientes permitem expressar um modelo por meio de desenhos dos corpos estaticamente representados, equações algébricas e/ou diferenciais. A execução de um modelo pode ser conseguida através de transformações algébricas aplicadas sobre as equações matemáticas que definem o fenômeno modelado. Os resultados do modelo executado se dão em forma de números e novas equações algébricas. Estes ambientes oferecem pouco suporte às fases de expressão, execução e avaliação dos resultados do modelo.

Software do tipo tutoriais ou simulações como os distribuídos com alguns livros de Mecânica e Física (por exemplo Beer & Johnston, 1991) não constituem ambientes para modelagem. Tutoriais são versões computadorizadas da instrução programada, método tradicional de ensino (Valente, 1993). Uma simulação, como vimos no capítulo 1, não permite que o usuário altere o seu correspondente modelo. Tipicamente, simulações permitem que o seu usuário execute o modelo subjacente e verifique os resultados da execução.

Ambientes para modelagem de propósito geral como Stella ou Extend (veja seção 2.2 do capítulo 2), dada a sua generalidade, normalmente ficam muito distantes do domínio sobre o qual um modelo está sendo desenvolvido. Sob o ponto de vista de expressão de um modelo da Mecânica, o Stella tipicamente representa elementos tais como a velocidade ou a posição dos corpos através de seus elementos gráficos e simbólicos (tanques, conversores, conexões etc.). E isto, é tudo. Mas, perguntamo-nos: onde estão os corpos, já que a Mecânica trata do movimento ou repouso de corpos? Se for necessário, como representar a interação dos corpos envolvidos em um fenômeno? Olhando um modelo Stella, o que se vê são tanques, conectores e outros aparatos. Na verdade, o Stella ou o Extend se apresentam como intermediários entre o aprendiz e o domínio da Mecânica. E isto não é privilégio apenas do Stella e do Extend. O ambiente dos livros de Física, lápis e papel, através de formulações algébricas e cálculo, há muito se presta à mesma finalidade: a diferença é que, no lugar de tanques e conectores, estão variáveis, integrais, equações diferenciais. Sob o ponto de vista de suporte à avaliação dos resultados de um modelo, o Stella gera animação sobre os tanques e conversores, exibe gráficos e tabelas de valores. No entanto, os corpos no mundo real podem se mover e o Stella priva seu usuário do interessante *feedback* de poder vê-los mover através de uma animação. No mundo real, cronômetros e instrumentos de medida de distância são usados para acompanhar e investigar a ocorrência de um fenômeno mecânico real. E novamente nos perguntamos: onde estão, no Stella ou no Extend, os instrumentos que permitem medir tempo e distância? Não existem. Se quisermos inferir a distância entre dois corpos representados em um modelo Stella, devemos subtrair o valor das respectivas posições dos mesmos simbolizadas em um diagrama, talvez, por um tanque ou um conversor.

O protótipo que desenvolvemos neste trabalho é uma resposta à indefinição destes ambientes por um domínio. O NEWTON toma o partido do domínio da Mecânica e permite que:

- se expresse o modelo através de representações para os corpos e de uma linguagem diagramática constituída de elementos do domínio da Mecânica;

- se execute o modelo de forma automática;

- os resultados do modelo possam ser observados na animação do comportamento dos corpos, em gráficos ou através do uso de instrumentos tais como cronômetros e

distanciômetros à semelhança da experimentação prática de fenômenos reais.

Por isto o usuário do NEWTON pode engajar-se com o domínio da Mecânica e experimentar da fantasia de criar modelos para fenômenos mecânicos com o sentimento de primeira pessoa descrito por Laurel (1986). Pode aprender sobre Mecânica quase sem notar a presença do computador. Enfim, praticar Mecânica antes de tudo. A figura 8.2 sintetiza esta discussão.

AMBIENTE	*SUPORTE A FASE DE*		
	DESCRIÇÃO DO MODELO	*EXECUÇÃO*	*ANÁLISE E INTERPRETAÇÃO DOS RESULTADOS*
Quadro-negro e giz ou livros de Física, caderno e lápis	Desenhos e equações algébricas e diferenciais	Transformações algébricas que conduzem à resolução das equações	Números e equações algébricas
Tutoriais	Não permite	-----	-----
Simulações	Não permite. Modelo é predefinido	Automática	Gráficos, números, animações
Stella ou Extend	Através de uma linguagem diagramática genérica	Automática	Gráficos, números, tabelas
Protótipo NEWTON	Representação para os corpos e suas inter-relações; linguagem diagramática constituída de elementos do domínio da Mecânica	Automática	Animação dos corpos, gráficos, números, tabelas, cronômetros, distanciômetro etc.

Fig. 8.2 - Suporte às fases da modelagem dada por alguns ambientes

8.5 - O QUÊ FIZEMOS E COMO ISTO PODE AJUDAR NO *DESIGN* DE AMBIENTES PARA MODELAGEM

Este é um trabalho de *design* de ambientes para modelagem. Por *design* não entendemos, neste trabalho, o projeto de *software* para modelagem no sentido de especificação dos módulos (procedimentos, funções etc.) que o compõem. Referimo-nos às estruturas do ambiente que participam e dão suporte à inter-relação Homem-

Computador ou, no caso específico, Aprendiz-Ambiente para Modelagem Computacional. Usando uma analogia semelhante à de Hooper (1986), podemos dizer que, neste trabalho, o *design* está para o projeto de *software* assim como a Arquitetura está para a Engenharia Civil. A Arquitetura diz respeito à organização dos espaços por meio do agenciamento urbano ou de edificações com o objetivo de servir aos diferentes tipos de atividades do Homem. A Engenharia Civil refere-se às construções, ao projeto das estruturas (pilares, lajes, vigas etc.) capazes de conferir forma concreta às estradas, edificações etc.. Ou seja, enquanto o arquiteto planeja o espaço para a inter-relação com o Homem, o engenheiro projeta a consolidação deste espaço por meio de estruturas e materiais conhecidos.

A principal contribuição deste trabalho se dá no sentido de guiar o *design* de ambientes para modelagem a partir da definição de um determinado domínio. E isto é feito a partir de um conjunto de princípios inter-relacionados e devidamente organizados para formar o que denominamos de estratégias. Estas estratégias não definem um caminho único, reto e plano entre a escolha de um domínio e o *design* de um ambiente para modelagem orientado a este domínio. Elas são bússolas que orientam o *designer* em seu trabalho de estabelecer o *design* de ambientes para modelagem com duas virtudes: (1) a de apresentar um equilíbrio entre o nível de controle do usuário sobre o ambiente e o respectivo esforço cognitivo demandado deste para apropriação dos conceitos de computação (seção 2.3 do capítulo 2 e seção 4.3 do capítulo 4); (2) a de conferir ao usuário um sentimento de ação direta sobre os objetos do domínio. O sentimento de ação direta se traduz na capacidade do ambiente em dispor de uma interface fácil de ser usada ao mesmo tempo em que convida o usuário (o aprendiz) a um engajamento direto com os objetos do domínio representado (seção 4.4 do capítulo 4). Além de contribuir para diminuir o esforço cognitivo do usuário para usar o ambiente, o sentimento de ação direta cria "laboratórios" virtuais responsáveis por permitir ao aprendiz participar da fantasia de estar manipulando os próprios objetos do domínio, estar construindo modelos "reais" dentro do "mundo real".

Entender o *design* de ambientes computacionais envolve entender a interação Homem-Computador. E interação Homem-Computador é um campo pluralista que envolve muitas disciplinas: Ciência da Computação, Psicologia, Inteligência Artificial, Lingüística, Antropologia e Sociologia (Draper e Norman, 1986). Considerando esta complexidade, em quê podem contribuir as estratégias para *design* colocadas por este trabalho? Entendemos que o mérito destas estratégias repousa no fato de poderem gerar um **objeto-**

para-pensar-sobre (Papert, 1980) para um conjunto de pessoas interessadas entender/possibilitar a construção do conhecimento através do computador.

"This computer is the proteus of machines. Its essence is its universality, its power to simulate. Because it can take on thousand forms and can serve a thousand function, it can appeal to a thousand tastes" (Papert, 1980, p. viii).

Apêndice A

Referência rápida ao uso do NEWTON

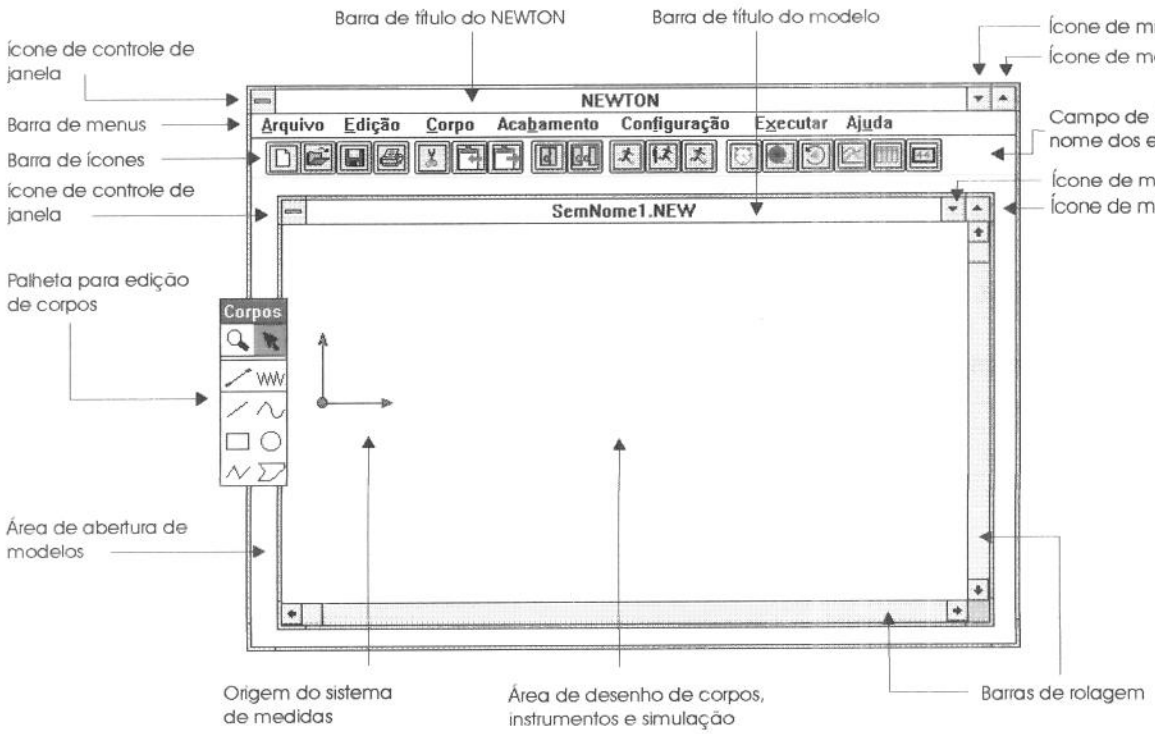
Esta referência rápida é um guia para utilização do ambiente para modelagem NEWTON. Este apêndice apresenta, de forma sucinta, a função de cada elemento do ambiente: menus, palhetas, botões da barra de ícones, elementos para definição dos diagramas de comportamento mecânico, expressões, operadores, funções predefinidas etc..

Tópicos deste apêndice:

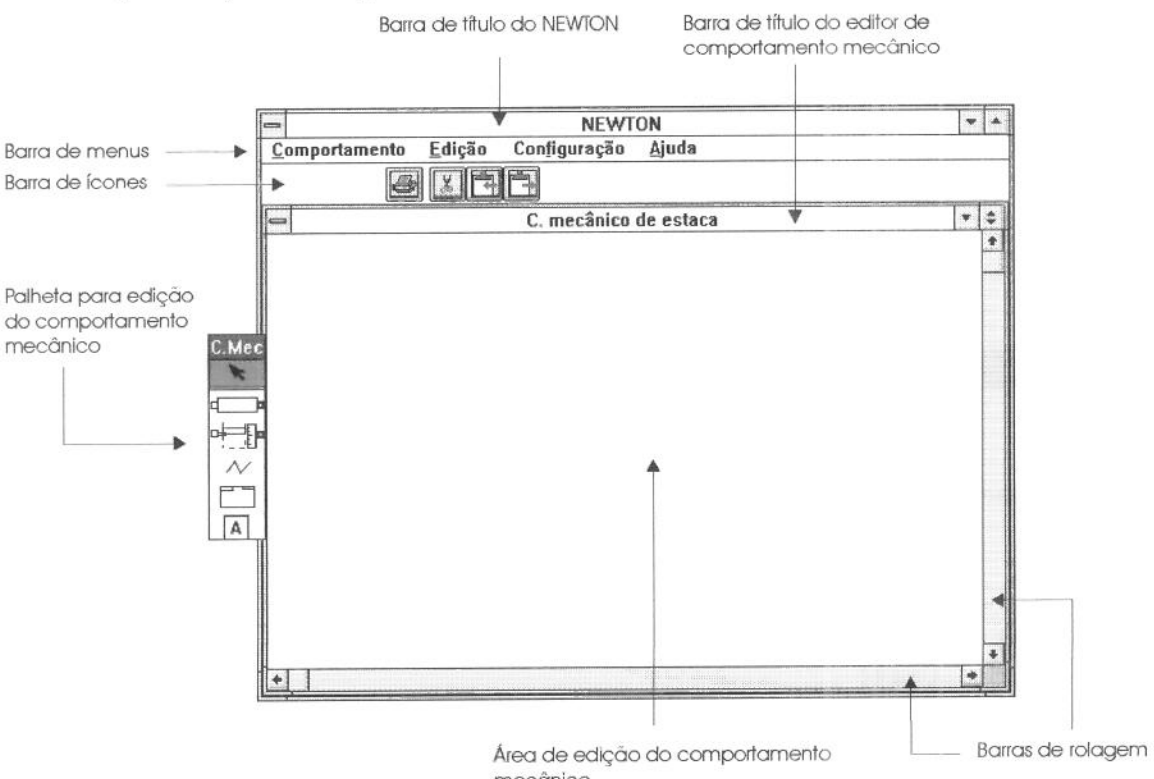
Tópico	*Página*
A.1 - Elementos gerais	197
A.2 - Menus	198
A.3 - Comandos dos menus	200
A.4 - Palhetas	204
A.5 - Barra de ícones	205
A.6 - Elementos básicos sobre a edição de comportamento mecânico	205
A.7 - Instrumentos de *feedback*	212
A.8 - Algumas dicas para se trabalhar com o NEWTON	213

A.1 - ELEMENTOS GERAIS

Área para edição de corpos, instrumentos de *feedback* e gerenciamento de modelos abertos



Área para edição do comportamento mecânico



A.2 - MENUS

Menus da janela de edição de corpos, instrumentos de *feedback* e gerenciamento de modelos abertos

Arquivo

Novo	Ctrl N
Abrir...	Ctrl A
Gravar	Ctrl G
Gravar como...	
Configurar impressora...	
Imprimir...	Ctrl I
Sair	Ctrl S

Edição

Cortar	Shift Del
Copiar	Ctrl Ins
Colar	Shift Ins
Apagar	Del
Selecionar tudo	Shift F9
Biblioteca de corpos	

Corpo

Agrupar	Ctrl R
Desagrupar	Ctrl D
Trazer para a frente	Ctrl T
Enviar ao fundo	Ctrl E
Um nível à frente	Ctrl +
Um nível ao fundo	Ctrl -

Acabamento

Transparente
Alinhar...
Acertar o desenho

Configuração

Origem	Ctrl O
Barra de ícones	
Réguas	
Grade...	
Palheta cores	Ctrl K
Palheta linhas	Ctrl H
Palheta padrões	Ctrl P

Executar

Executar	F3
Simulação...	

Ajuda

Índice	F1
Glossário	
Sobre...	

Menus da janela de edição de comportamento mecânico

Comportamento

Configurar impressora...	
Imprimir...	Ctrl I
Testar sintaxe	Ctrl T
Sair cancelando	Ctrl C
Sair alterando	Ctrl S

Edição

Cortar	Shift Del
Copiar	Ctrl Ins
Colar	Shift Ins
Apagar	Del
Selecionar tudo	Shift F9

Configuração

Barra de ícones	
Palheta cores	Ctrl K

Ajuda

Índice	F1
Glossário	
Sobre...	

A.3 - COMANDOS DOS MENUS

Comandos dos menus da área de edição de corpos, instrumentos de *feedback* e gerenciamento dos modelos abertos

Comando	*Efeito*
Abrir	Abre um modelo já existente.
Acertar o desenho	Exibe marcadores que permitem alterar o desenho de arcos, polígonos e linhas angulares.
Agrupar	Agrupa os objetos selecionados.
Alinhar	Alinha os corpos selecionados uns em relação aos outros (ao centro na horizontal, ao centro na vertical, pelo lado esquerdo, pelo lado direito, pelo topo, por baixo).
Apagar	Remove o corpo ou instrumento selecionado.
Barra de ícones	Esconde ou exibe a barra de ícones.
Biblioteca de corpos	Exibe a biblioteca de desenhos de corpos.
Colar	Insere no modelo ativo uma cópia de um objeto previamente armazenado no *clipboard.*
Configurar impressora	Configura a impressora *default* e suas opções.
Copiar	Insere uma cópia do corpo ou instrumento selecionado no *clipboard.*
Cortar	Remove o corpo ou instrumento selecionado e o insere no *clipboard.*
Desagrupar	Desagrupa o grupo selecionado.
Enviar ao fundo	Faz com que o corpo selecionado seja colocado atrás de todos os outros.
Executar	Executa os modelos abertos.
Glossário	Exibe um glossário de termos.

Continuação: comandos dos menus da área de edição de corpos, instrumentos de *feedback* e gerenciamento dos modelos abertos

Comando	*Efeito*
Grade	Permite exibir uma grade de pontos e configurá-la.
Gravar	Grava, em disco, o modelo ativo.
Gravar como	Grava, em disco, o modelo ativo sob o nome especificado.
Imprimir	Imprime a área de desenho de corpos e instrumentos do modelo ativo.
Índice	Exibe o índice de tópicos de ajuda (*help*).
Novo	Cria um novo modelo, vazio, na área para abertura de modelos.
Origem	Permite, no modelo ativo, travar ou destravar o par de eixos que demarcam a origem do sistema de medidas e/ou definir as dimensões da área de simulação.
Palheta cores	Esconde ou exibe a palheta de cores.
Palheta linhas	Esconde ou exibe a palheta de espessura e tipo de linhas.
Palheta padrões	Esconde ou exibe a palheta de padrões de preenchimentos.
Réguas	Esconde ou exibe, no modelo ativo, um par de réguas.
Sair	Termina a execução do NEWTON.
Selecionar tudo	Seleciona todos os corpos ou instrumentos existentes no modelo ativo.
Simulação	Permite estabelecer opções de simulação (tempo de duração, condições de parada, precisão de cálculo etc.).
Sobre	Exibe informações gerais sobre o protótipo NEWTON.
Transparente	Torna transparente o corpo selecionado.
Trazer para frente	Faz com que o corpo selecionado apareça na frente de todos os outros.

Continuação: comandos dos menus da área de edição de corpos, instrumentos de *feedback* e gerenciamento dos modelos abertos

Comando	*Efeito*
Um nível à frente	Faz com que o corpo selecionado seja colocado um nível à frente do nível em que se encontra.
Um nível ao fundo	Faz com que o corpo selecionado seja colocado um nível atrás do nível em que se encontra.

Comandos dos menus da área de edição de definição do comportamento mecânico

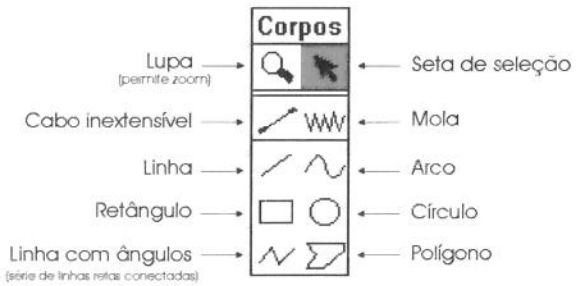
Comando	*Efeito*
Apagar	Remove o elemento selecionado no diagrama.
Barra de ícones	Esconde ou exibe a barra de ícones.
Colar	Insere na área de edição de comportamento mecânico uma cópia de um elemento previamente armazenado no *clipboard.*
Configurar impressora	Configura a impressora *default* e suas opções.
Copiar	Insere, no *clipboard*, uma cópia do elemento selecionado no diagrama.
Cortar	Remove o elemento selecionado no diagrama e o insere no *clipboard.*
Glossário	Exibe um glossário de termos.
Imprimir	Imprime o diagrama de comportamento mecânico.
Índice	Exibe o índice de tópicos de ajuda (*help*).
Palheta Cores	Esconde ou exibe a palheta de cores.
Sair alterando	Abandona a área de edição de corpos e retorna à área de desenho de corpos. A sintaxe do diagrama é testada e as alterações feitas são consolidadas.

Continuação: comandos dos menus da área de edição de definição do comportamento mecânico

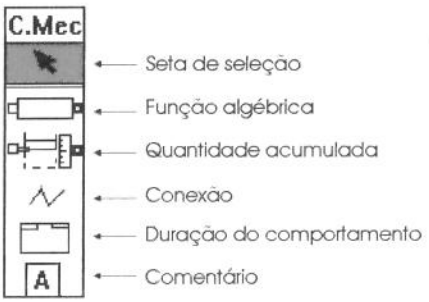
Comando	*Efeito*
Sair cancelando	Abandona a área de edição de corpos e retorna à área de desenho de corpos. A sintaxe do diagrama não é testada e as alteraçãoes feitas não são consolidadas.
Selecionar tudo	Seleciona todos os elementos existentes em um diagrama.
Sobre	Exibe informações genéricas sobre o protótipo NEWTON.
Testar sintaxe	Verifica a existência de construções diagramáticas incorretos ou expressões sintaticamente incorretas.

A.4 - PALHETAS

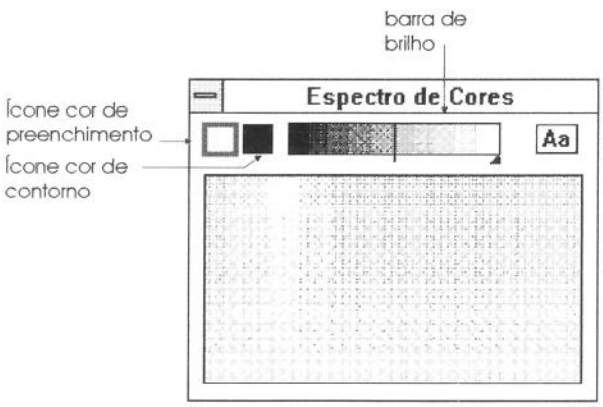
Desenho de corpos



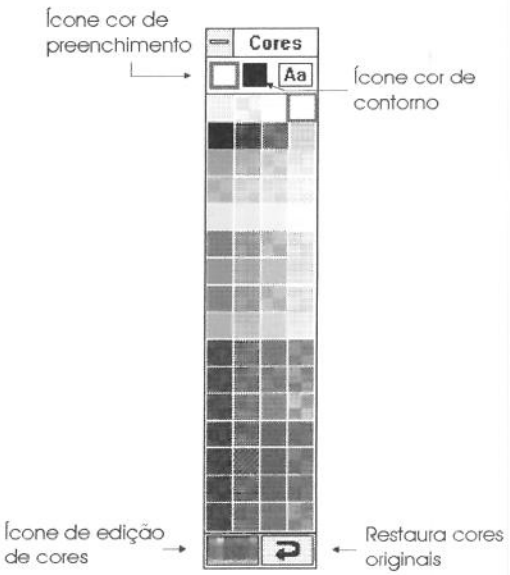
Edição de comportamento mecânico



Espectro de cores

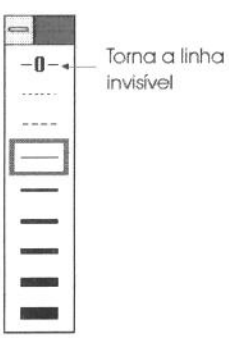


Cores



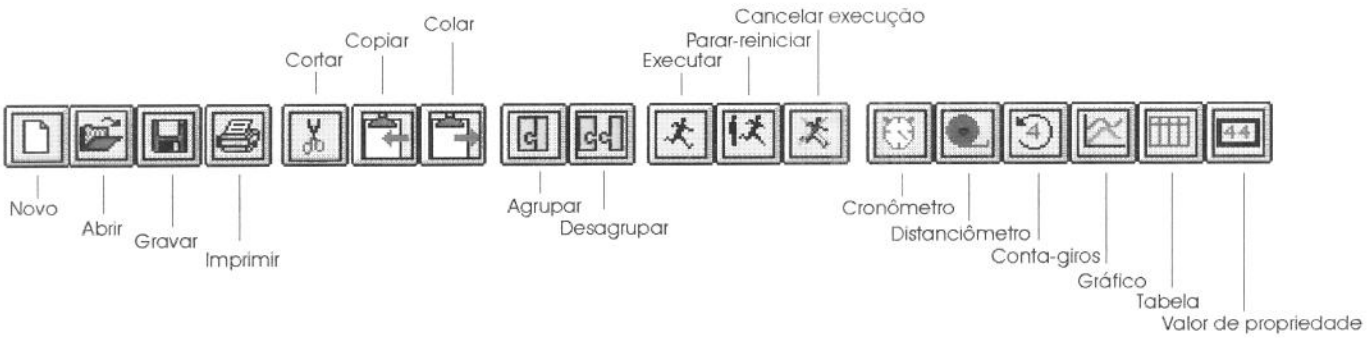
Padrões de preenchimento

Linhas

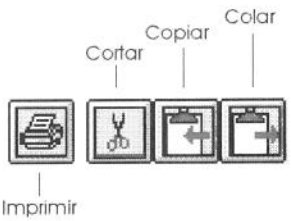


A.5 - BARRA DE ÍCONES

Área para edição de corpos, instrumentos de *feedback* e gerenciamento de modelos abertos



Área para edição do comportamento mecânico



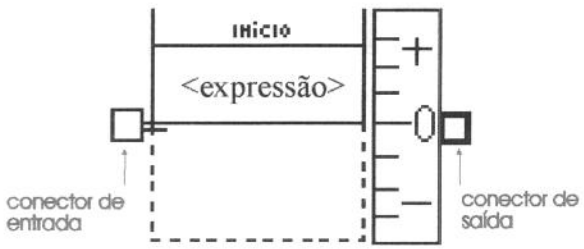
A.6 - ELEMENTOS BÁSICOS SOBRE A EDIÇÃO DE COMPORTAMENTO MECÂNICO

Símbolos usados na descrição da sintaxe

Símbolo	*Significado*
< >	Itens que devem substituídos por um valor literal ou uma expressão.
[ ]	Indica opcionalidade dos itens inseridos em seu interior.
{ }	Indica repetição de zero ou mais vezes dos itens inseridos em seu interior.
|	Qualquer um dos itens separados por este símbolo podem ser utilizados na expressão.

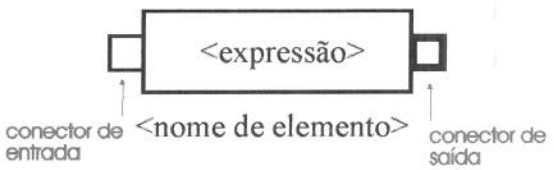
Elementos gráficos

Quantidade acumulada

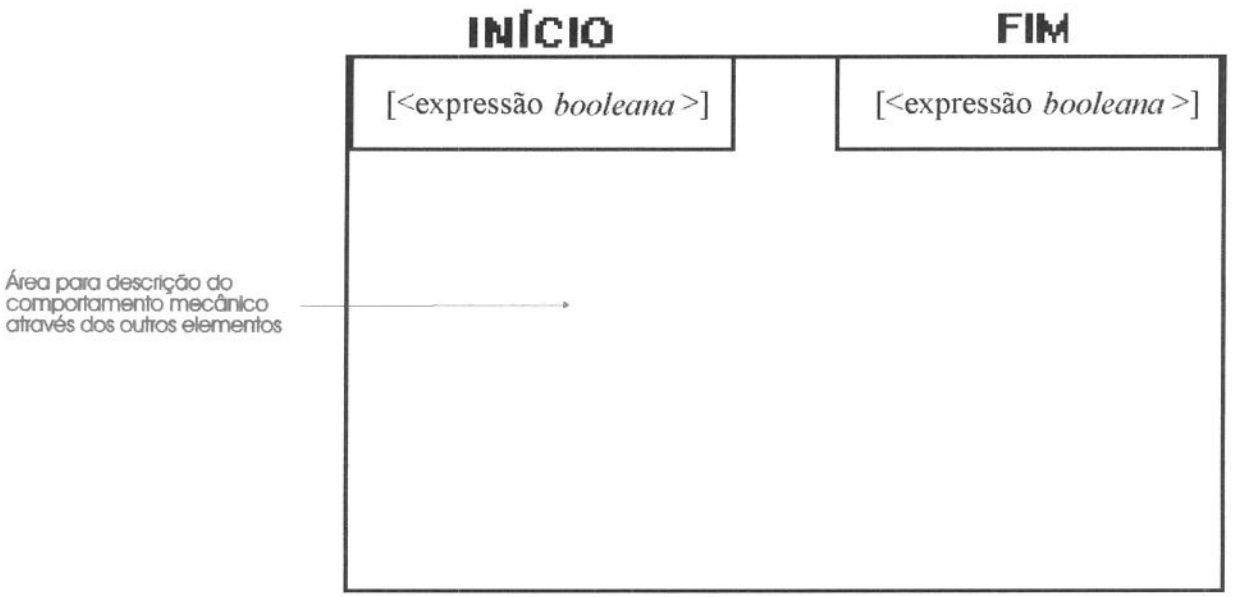


<nome de elemento>

Função algébrica



Duração do comportamento



Conexão

Comentário

[<texto livre>]

Elementos simbólicos

Elemento	*Significado*
<base>	<número inteiro>
<dígito>	0|1|2|3|4|5|6|7|8|9
<expressão booleana>	<subexpressão1> { ou <subexpressão1> }
<expressão>	[-] <subexpressão3> { +|- <subexpressão 3> }
<função>	<identificador função> (<expressão> [, <expressão>])
<identificador função>	abs|random|arredonda|trunca|raizQuadrada|acos|asen|atan|atan2|cos|cosh|sen|senh|tan|tanh|exp|log|ln
<identificador>	<letra> { <letra>|<dígito>}
<letra>	a|b|c|...|z|A|B|C|...|Z|_|à|á|ã|â|è|é|ê|ì|í|î|ò|ó|õ|ô|ù|ú|û|ç|ä|ë|ï|ö|ü|À|Á|Ã|Â|È|É|Ê|Ì|Í|Î|Ò|Ó|Õ|Ô|Ù|Ú|Û|Ç|Ä|Ë|Ï|Ö|Ü
<nome de corpo>	<identificador>
<nome de elemento>	<identificador>
<número>	<dígito> { <dígito> } [<separador>] [<dígito>] { <dígito> } [e|E [+|-] <dígito> <dígito>]
<número inteiro>	<dígito> { <dígito> }
<operador relacional>	=|<>|<|<=|>|>=
<propriedade>	<identificador> |tempo | dt | <identificador> de |da |do [corpo] <identificador>
<separador>	.
<subexpressão1>	<subexpressão2> { e <subexpressão2> }
<subexpressão2>	não|nao <expressão booleana> | (<expressão booleana>) | <expressão> <operador relacional> <expressão>

Continuação: elementos simbólicos

Elemento	*Significado*
< subexpressão3 >	< subexpressão4 > { *|/ |div |res < subexpressão4 > }
< subexpressão4 >	< subexpressão5 > { ^ < subexpressão5 > }
< subexpressão5 >	< propriedade > | < número > | < função > | (< expressão >)
< texto livre >	{ < letras > | < dígito > }

Palavras reservadas (não podem ser usadas como identificador)

corpo	e
da	nao
de	não
div	ou
do	res
dt	tempo

Operadores

Funções predefinidas por tipo

Funções aritméticas

Função	*Valor de retorno*
abs (<expressão>)	O valor absoluto de seu argumento.
arredonda (<expressão>)	O menor inteiro que é maior ou igual ao valor do seu argumento.
raizQuadrada (<expressão>)	A raiz quadrada de seu argumento.
random (<número inteiro>)	Um número "aleatório" entre 1 e o valor do argumento.
trunca (<expressão>)	A parte inteira de seu argumento.

Funções trigonométricas

Função	*Valor de retorno*
acos (<expressão>)	O arco, em radianos, cujo co-seno tem valor igual ao argumento.
asen (<expressão>)	O arco, em radianos, cujo seno tem valor igual ao argumento.
atan (<expressão>)	O arco, em radianos, cuja tangente tem valor igual ao argumento.
atan2 (<expressão>, <expressão>)	O arco, em radianos, cuja tangente tem valor igual à razão entre o primeiro e o segundo argumento. Os sinais dos argumentos são usados para determinar o quadrante do valor retornado.
cos (<expressão>)	O co-seno do angulo cujo valor, em radianos, é dado pelo argumento.
cosh (<expressão>)	O co-seno hiperbólico cujo valor, em radianos, é dado pelo argumento.
sen (<expressão>)	O seno do ângulo cujo valor, em radianos, é dado pelo argumento.

Continuação: funções trigonométricas

Função	*Valor de retorno*
senh (<expressão>)	O seno hiperbólico do ângulo cujo valor, em radianos, é dado pelo argumento.
tan (<expressão>)	A tangente do angulo cujo valor, em radianos, é dado pelo argumento.
tanh (<expressão>)	A tangente hiperbólica do angulo cujo valor, em radianos, é dado pelo argumento.

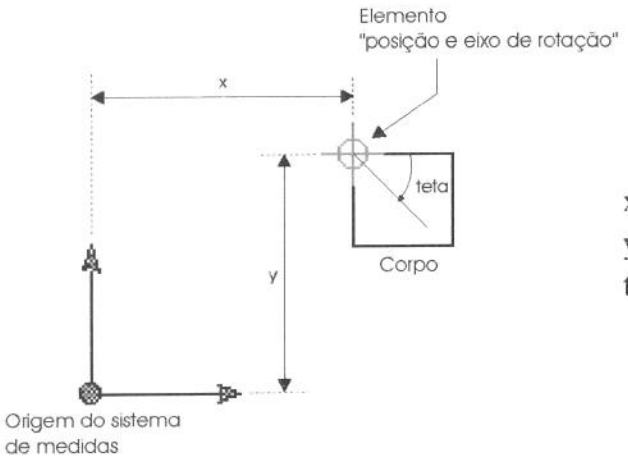
Funções logarítmicas

Função	*Valor de retorno*
exp (<expressão>)	O valor da constante **e** (2.7182818) elevada ao expoente dado pelo argumento. Em outras palavras $e^{<\text{expressão}>}$.
ln (<expressão>)	O logaritmo natural, ou logaritmo na base **e** (2.7182818), do valor dado pelo argumento.
log (<expressão>, <base>)	O logaritmo do número dado no primeiro argumento na base especificada pelo segundo argumento.

Propriedades

Corpos no NEWTON podem conter propriedades. O modelador cria propriedades para um corpo quando, na definição do diagrama do comportamento mecânico, cria elementos do tipo quantidade acumulada ou função algébrica.

Propriedades padrão (predefinidas) de um corpo



x: abscissa da posição do corpo
y: ordenada da posição do corpo
teta: coordenada (ou posição) angular do corpo

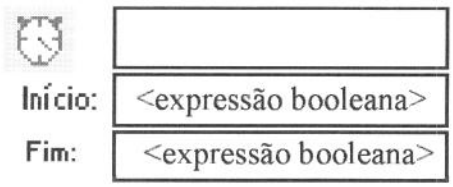
Observação:

1 - Para exibir e editar o elemento "posição e eixo de rotação" click no corpo pressionando simultaneamente as teclas CTRL e SHIFT.

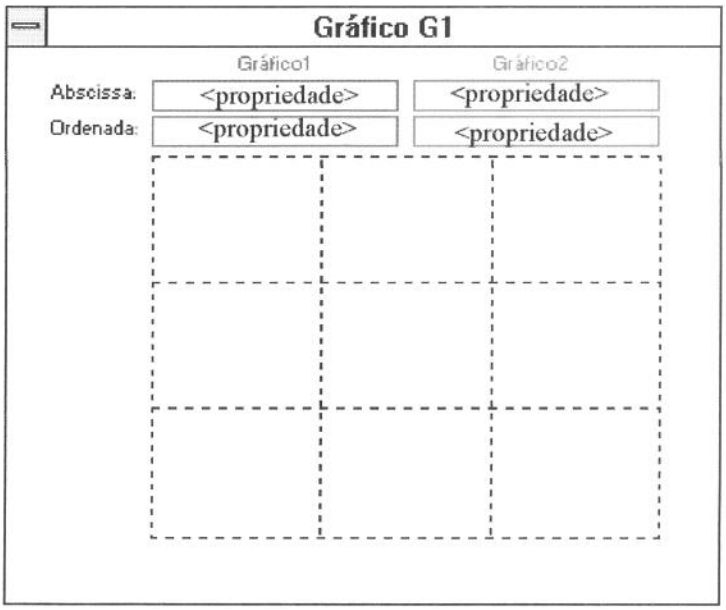
2 - Todos os corpos são criados com as propriedades padrão x, y e teta. Alterações de valor destas propriedades implicam na correspondente varição da posição do corpo na tela.

A.7 - INSTRUMENTOS DE *FEEDBACK*

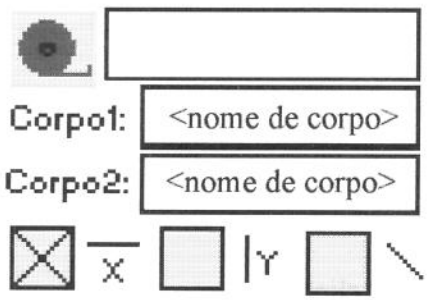
Cronômetro



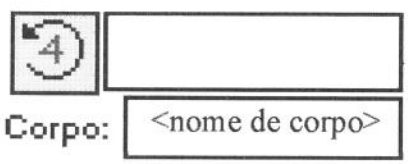
Gráfico



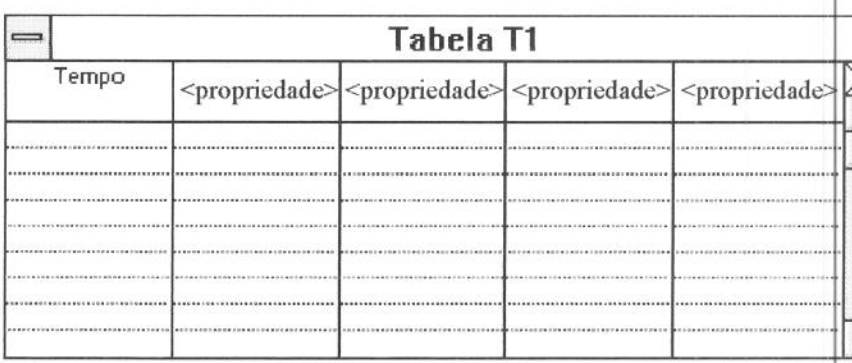
Distanciômetro



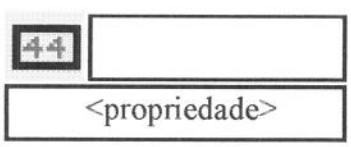
Conta-giros



Tabela



Valor de propriedade



A.8 - ALGUMAS DICAS PARA SE TRABALHAR COM O NEWTON

Para	*O que fazer*
Acionar a ajuda (*help*)	Pressionar F1.
Editar o comportamento mecânico de um corpo	Dar um duplo *click* no corpo.
Restringir o desenho de linhas apenas às direções horizontal e vertical	Pressionar a tecla Ctrl enquanto desenha.
Editar o elemento "posição e eixo de rotação de um corpo"	Pressionar simultaneamente as teclas Ctrl e Shift e dar um *click* no corpo desejado.
Acionar um comando	Existem, na maioria dos casos, quatro formas diferentes para se acionar um comando: (1) escolher o comando através do menu; (2) usar a tecla ALT em conjunto com uma das letras sublinhadas dos comandos dentro do menu; (3) dar um *click* no correspondente ícone na barra de ícones; (4) usar uma tecla de atalho (por exemplo Ctrl A aciona o comando Abrir).
Selecionar um corpo, um instrumento ou um elemento de edição de comportamento mecânico	Dar um *click* no mesmo. A seleção precede uma série de operações que se pode fazer com um corpo, um instrumento ou um elemento de edição de comportamento mecânico.
Tornar um modelo ativo	Dar um *click* em qualquer ponto da janela do mesmo.

Apêndice B

Alguns modelos criados com o NEWTON

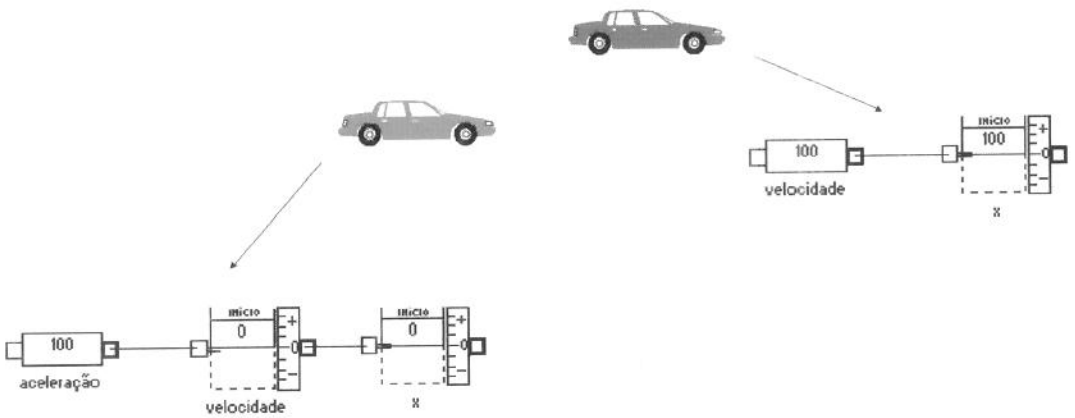
Este apêndice pretende mostrar um pouco da diversidade de modelos que podem ser criados com o NEWTON. Em cada caso os fenômenos são descritos e um correspondente modelo é apresentado. Foge do escopo deste apêndice a discussão detalhada das idéias e raciocínios que levaram a formulação dos modelos da maneira como estão apresentados. Entendemos, no entanto, que o raciocínio implícito na maioria dos modelos pode ser facilmente verificado.

Tópicos deste apêndice:

Tópico		*Página*
B.1 -	Cinemática de dois automóveis	217
B.2 -	Sistema de amortecimento usado para reduzir o recuo de certos tipos de armas	217
B.3 -	Canhão antiaéreo	218
B.4 -	Lançamento vertical de um projétil para grandes altitudes	219
B.5 -	Impulsão de um bloco sobre uma superfície sem atrito	219
B.6 -	Impulsão de um bloco sobre uma superfície com atrito	220
B.7 -	Bloco sobre um plano inclinado	221
B.8 -	Tubo de raios catódicos	222
B.9 -	Uma bola quicando sobre uma superfície plana	223
B.10 -	Cravamento de uma estaca	224
B.11 -	Viga de dois apoios	225
B.12 -	Turbina	226
B.13 -	Sistema motor	227

B.1 - CINEMÁTICA DE DOIS AUTOMÓVEIS

Um automóvel parte da origem (posição = 0) com aceleração constante e igual a 100; outro parte da posição 100 e viaja sob velocidade constante de valor 100.

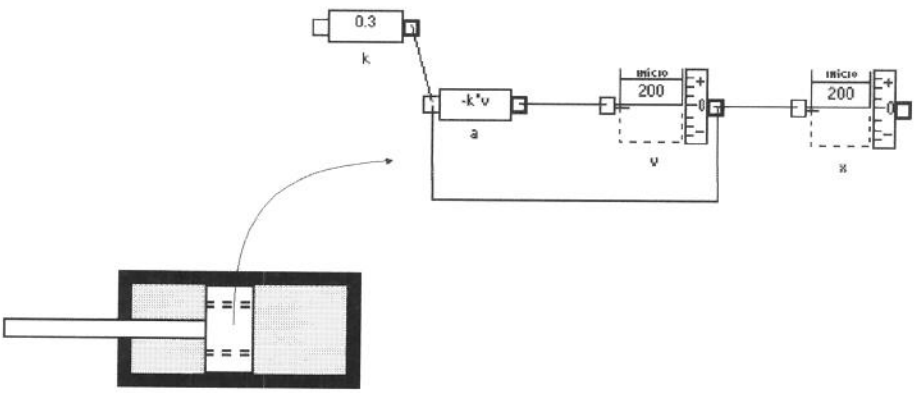


Este é um modelo típico de cinemática por envolver os conceitos de posição, tempo, velocidade e aceleração sem, contudo, explicitar as forças que atuam sobre os corpos. Variações interessantes deste modelo consistem em alterar o comportamento de cada automóvel em diferentes intervalos de tempo ou espaço percorrido. O uso dos instrumentos de *feedback* como cronômetros, distanciômetros, gráficos e tabelas além de contribuir na compreensão do fenômeno pode motivar o aprendiz a resolver com o computador os problemas tradicionais de cinemática descritos na literatura de Mecânica como por exemplo: determinar o tempo transcorrido até que um automóvel ultrapasse o outro (se a ultrapassagem ocorre); determinar a distância entre os automóveis após ter sido transcorrido um certo intervalo de tempo; calcular a velocidade dos automóveis no momento da ultrapassagem etc.

B.2 - SISTEMA DE AMORTECIMENTO USADO PARA REDUZIR O RECUO DE CERTOS TIPOS DE ARMAS

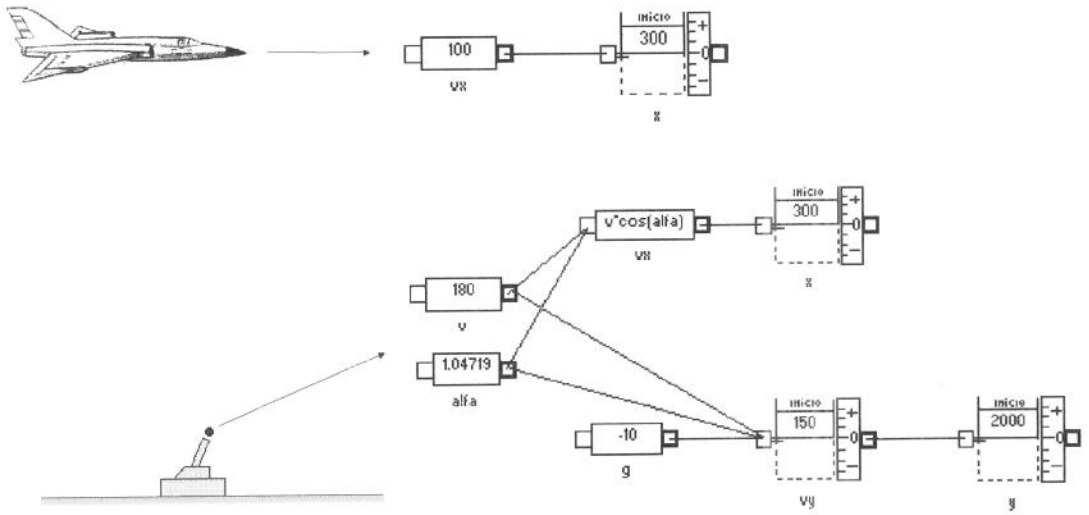
O mecanismo de amortecimento usado para reduzir o recuo em certos tipos de armas consiste, essencialmente, de um pistão que está preso ao cano da arma e que pode mover-se em um cilindro fixo cheio de óleo. Quando o cano recua com uma velocidade de

módulo igual a 200, o pistão movimenta-se e o óleo é forçado a passar através dos orifícios no pistão, provocando uma desaceleração do pistão e, por conseqüência, do cano. O modelo para este fenômeno parte do princípio que a desaceleração do pistão e do cano é proporcional à velocidade instantânea do mesmo, isto é, igual a *-k*v* onde *k* é uma constante e *v* é a velocidade.



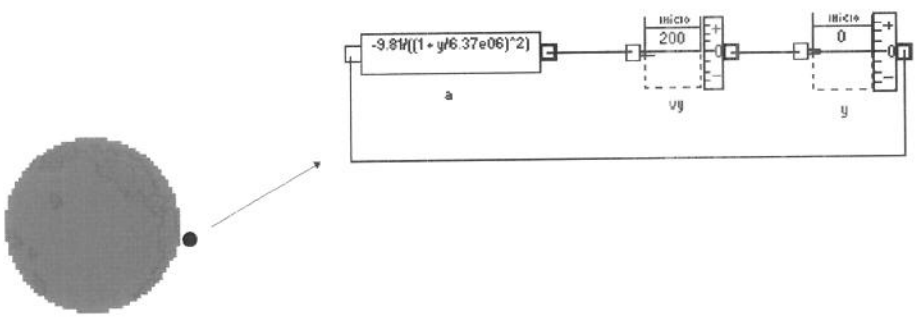
B.3 - CANHÃO ANTIAÉREO

Um canhão antiaéreo dispara uma granada no instante em que um avião passa sobre ele. O avião voa horizontalmente a uma certa altitude e a granada é disparada a uma velocidade que forma um ângulo *alfa* com a horizontal. O modelo define comportamento mecânico para o avião e para a granada.



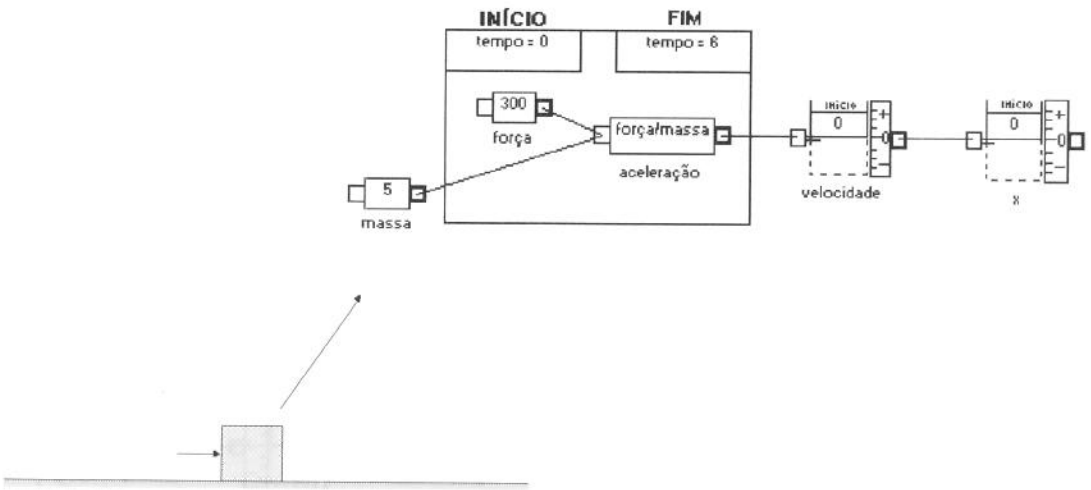
B.4 - LANÇAMENTO VERTICAL DE UM PROJÉTIL PARA GRANDES ALTITUDES

Um projetil é lançado verticalmente a partir da superfície da terra. O modelo apresentado leva em consideração a variação da gravidade da terra à medida que o projétil vai ganhando altura mas despreza o efeito da resistência do ar.



B.5 - IMPULSÃO DE UM BLOCO SOBRE UMA SUPERFÍCIE SEM ATRITO

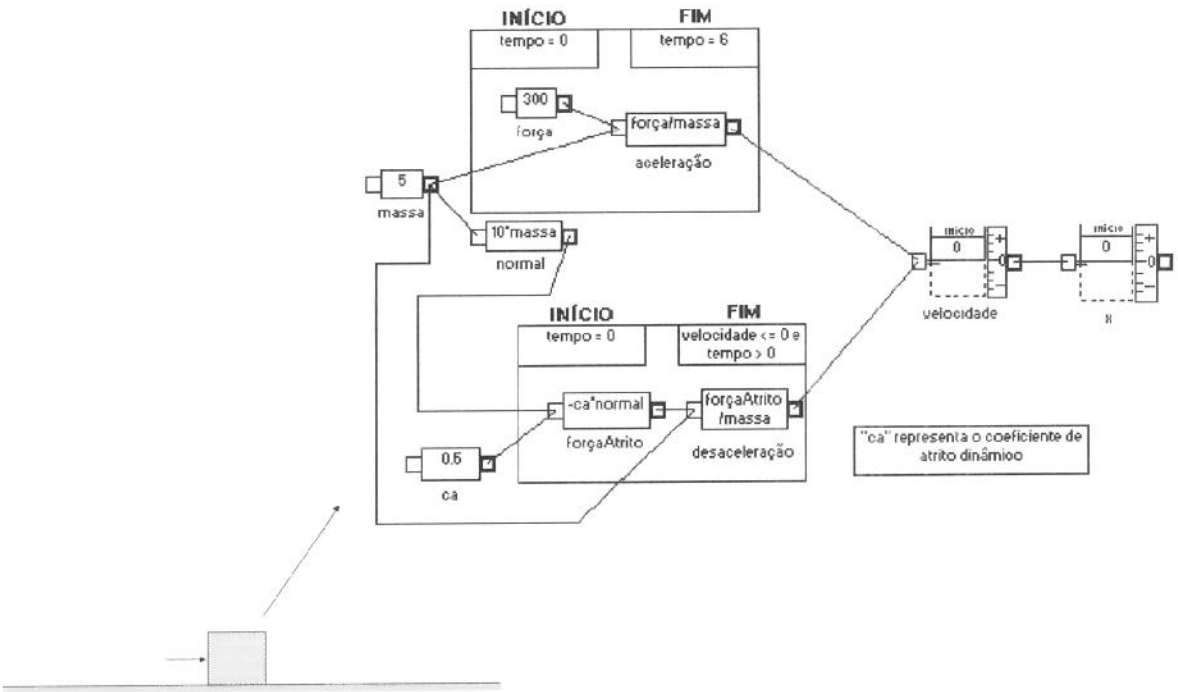
Um bloco sobre uma superfície livre de atrito é impulsionado por uma força durante 6 segundos a partir do repouso.



Um aspecto conceitual interessante deste modelo repousa no fato que após transcorrido a ação da força, o bloco continua a se movimentar sob velocidade constante de intensidade igual ao valor adquirido durante o período de impulsão. Embora simples, nem sempre as pessoas respondem corretamente questões envolvendo este aspecto conceitual (diSessa, A., 1982; diSessa e White, 1982; McCloskey, M., 1983; Squires, D., 1987).

B.6 - IMPULSÃO DE UM BLOCO SOBRE UMA SUPERFÍCIE COM ATRITO

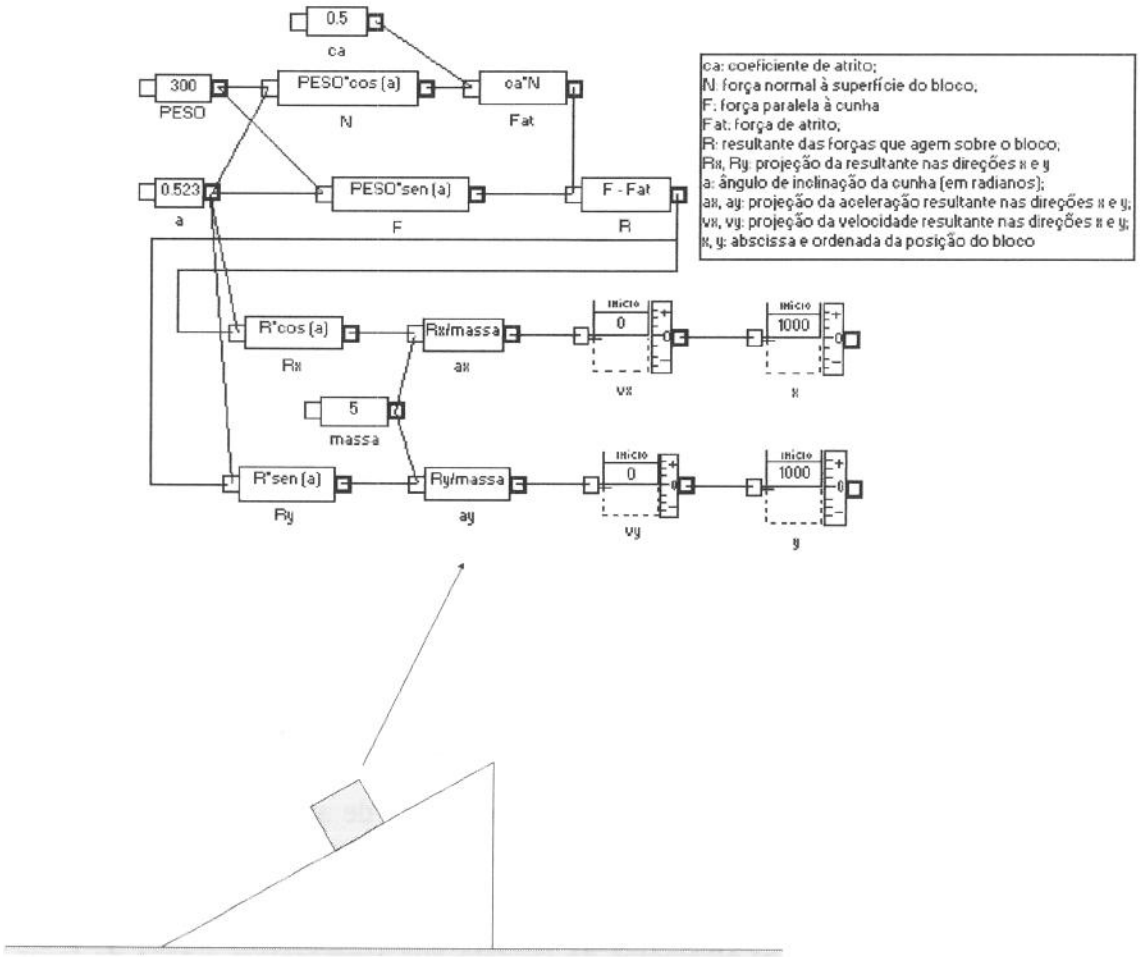
Idem fenômeno do item B.5, só que agora existe atrito entre o bloco e a superfície sobre a qual ele desliza.



Ao contrário do item B.5, o bloco para sob ação da força de atrito. É interessante notar que a força de atrito se opõe ao movimento do bloco e só atua sobre este enquanto ele possui movimento, isto é velocidade > 0. Por isto foi usado o elemento duração de comportamento que envolve os elementos do modelo que representam a ação da força de atrito. A não observação deste aspecto conduz ao estabelecimento do modelo segundo o qual o bloco, após parar sob ação da força de atrito, passa a se mover no sentido oposto ao movimento. Salientamos que erros conceituais desta natureza podem ser facilmente identificados no NEWTON pela observação da animação dos corpos.

B.7 - BLOCO SOBRE UM PLANO INCLINADO

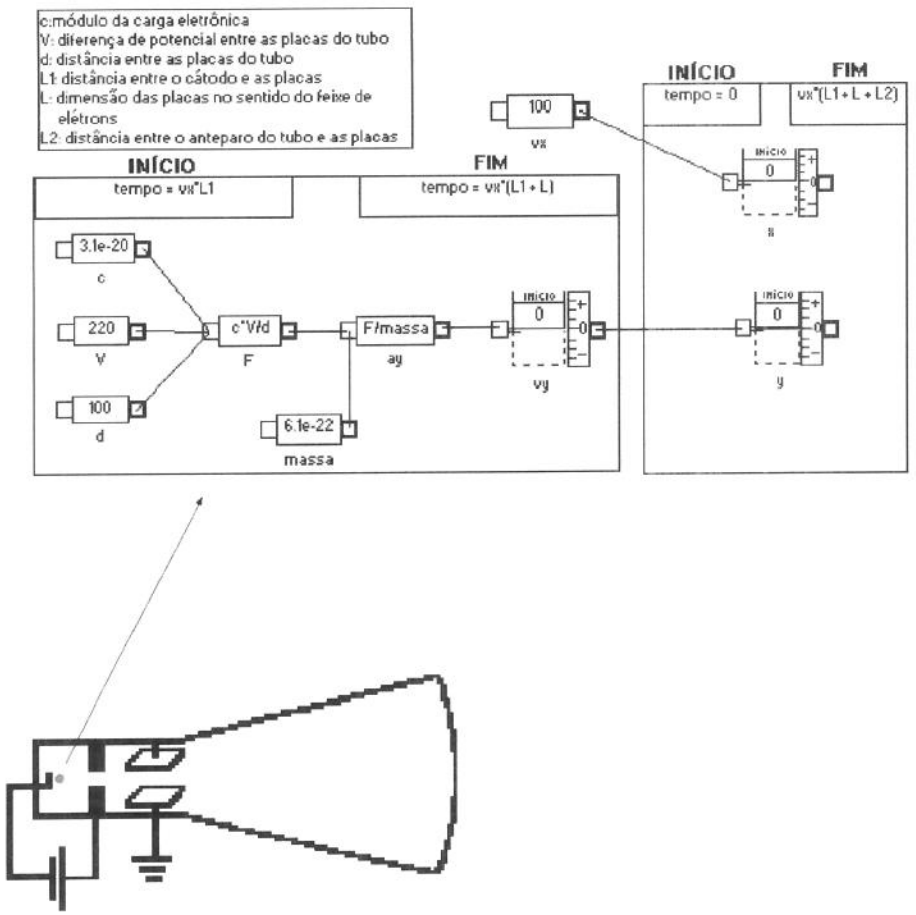
Um bloco desliza sobre um plano inclinado (cunha). O modelo representa a ação das forças da gravidade e de atrito entre o bloco e o plano inclinado.



B-8 - TUBO DE RAIOS CATÓDICOS

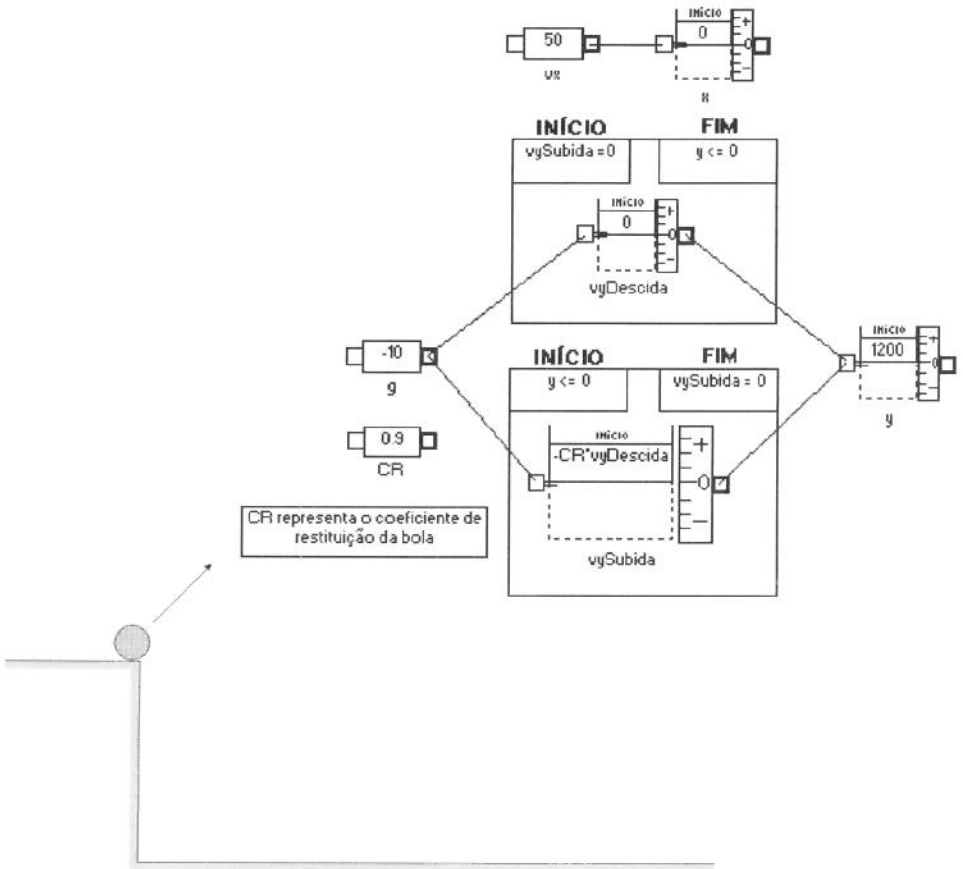
Em um tubo de raios catódicos os elétrons emitidos pelo cátodo[1] são atraídos pelo ânodo[2], passam por um orifício neste último e mantêm uma trajetória retilínea a uma velocidade constante, até atingirem o centro do anteparo da tela. Entretanto, quando se estabelece uma voltagem V, cada elétron é defletido quando passa entre duas placas horizontais pois ficam submetidos a uma força F de intensidade igual a cV / d, onde c é uma constante que representa a carga eletrônica e d é a distância entre as placas horizontais (Beer e Johnston, 1991, p. 151).

[1] Cátodo refere-se ao eletrodo negativo, isto é, de onde partem os elétrons e para onde se dirigem os íons positivos. Na figura o cátodo está ligado ao negativo da bateria.
[2] Ânodo é o eletrodo positivo. Na figura o ânodo está ligado ao positivo da bateria.



B.9 - UMA BOLA QUICANDO SOBRE UMA SUPERFÍCIE PLANA

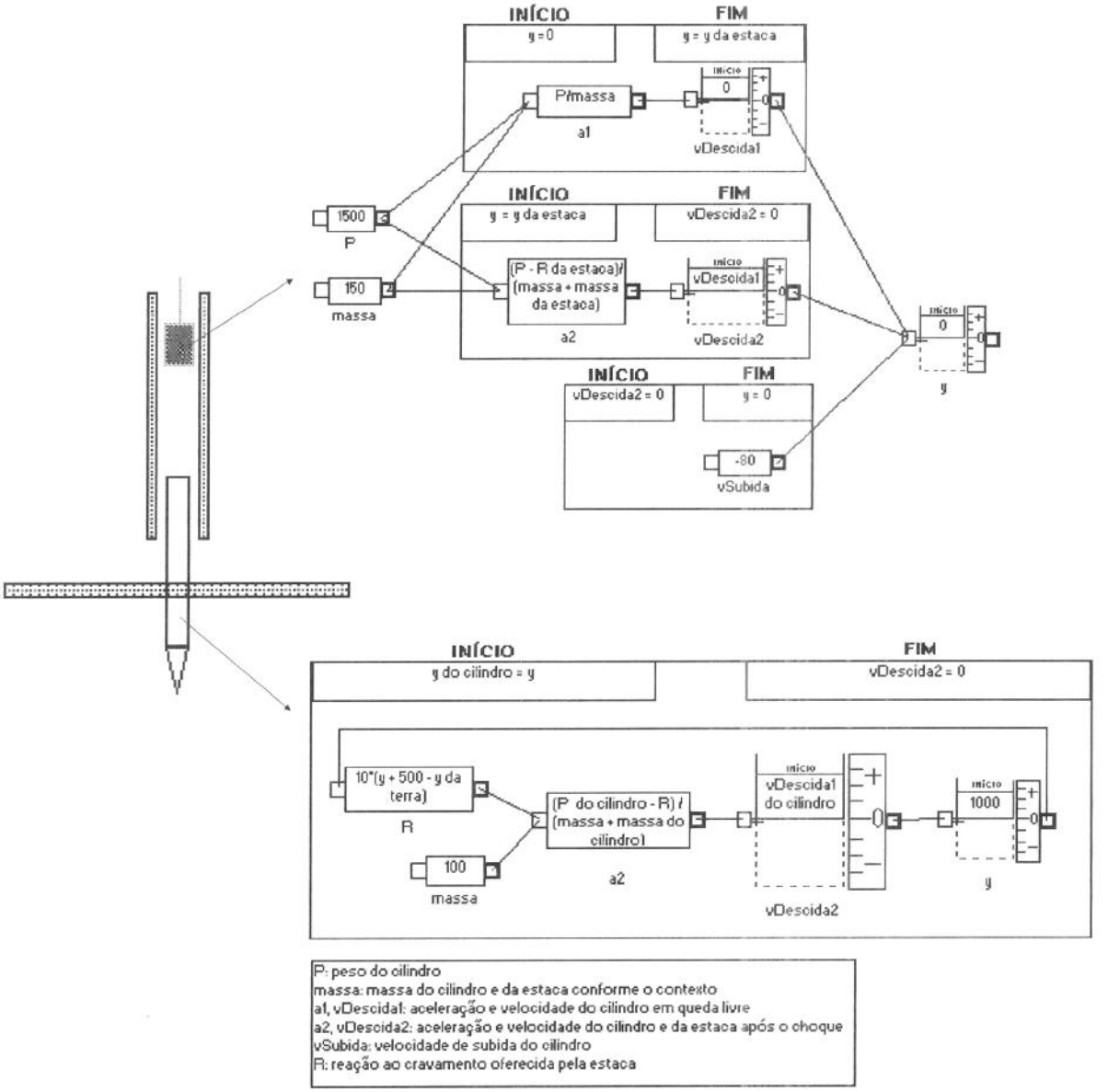
Uma bola movendo-se com velocidade horizontal constante cai sobre uma superfície plana sem atrito.



O modelo parte do princípio que a bola não é perfeitamente elástica, assim a bola vai perdendo altura à medida que vai se chocando com a superfície. Após cada choque com a superfície a sua velocidade é afetada na proporção de um coeficiente (CR). Um modelo mais realista levaria em consideração, também o atrito entre a bola e a superfície durante o choque bem como a resistência imposta pelo ar ao movimento da bola .

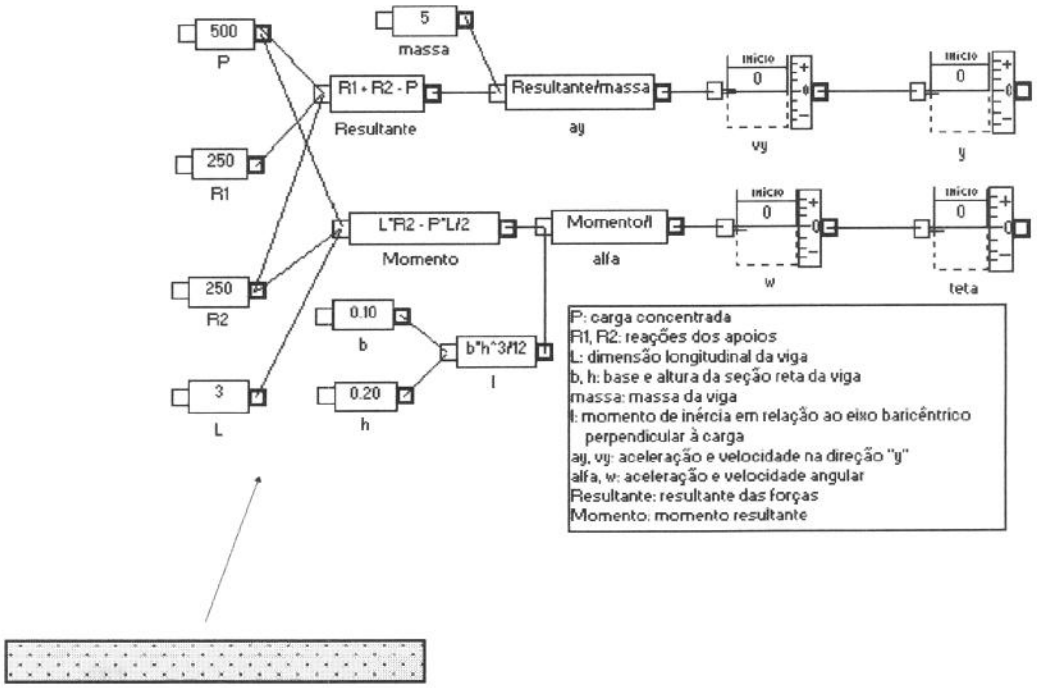
B.10 - CRAVAMENTO DE UMA ESTACA

Um cilindro de um bate-estacas cai de uma certa altura sobre uma estaca. O modelo supõe que a resistência que a terra oferece ao cravamento da estaca é proporcional altura encravada da estaca.



B.11 - VIGA DE DOIS APOIOS

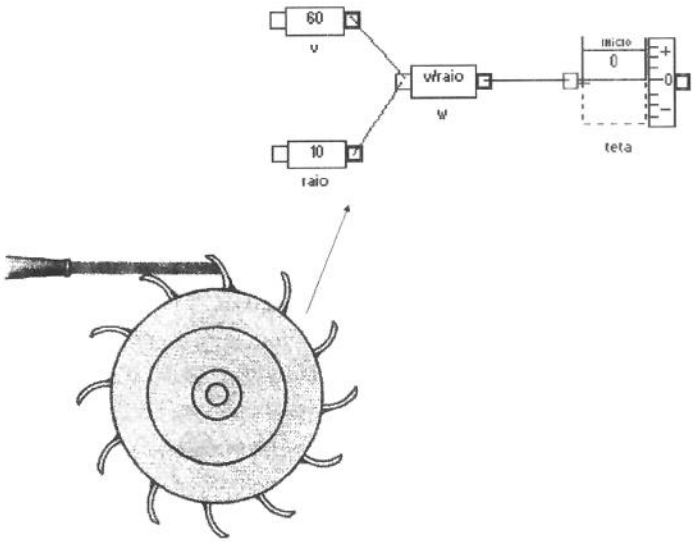
Uma viga bi-apoiada está submetida a uma carga concentrada em seu centro.



O termo viga refere-se a uma estrutura estática projetada para receber cargas, normalmente perpendiculares ao seu eixo, aplicadas em vários pontos de sua extensão maior (Beer e Johnston, 1980, p. 277). Por estrutura estática estamos entendendo estruturas que trabalham sem movimento. Todos os outros modelos que apresentamos neste apêndice são modelos para o que denominamos de mecanismos, isto é, corpos que podem se movimentar. Este é um modelo para o comportamento da viga mas, também pode ser considerado um modelo para um mecanismo correspondente no caso de não haver equilíbrio entre as cargas aplicadas.

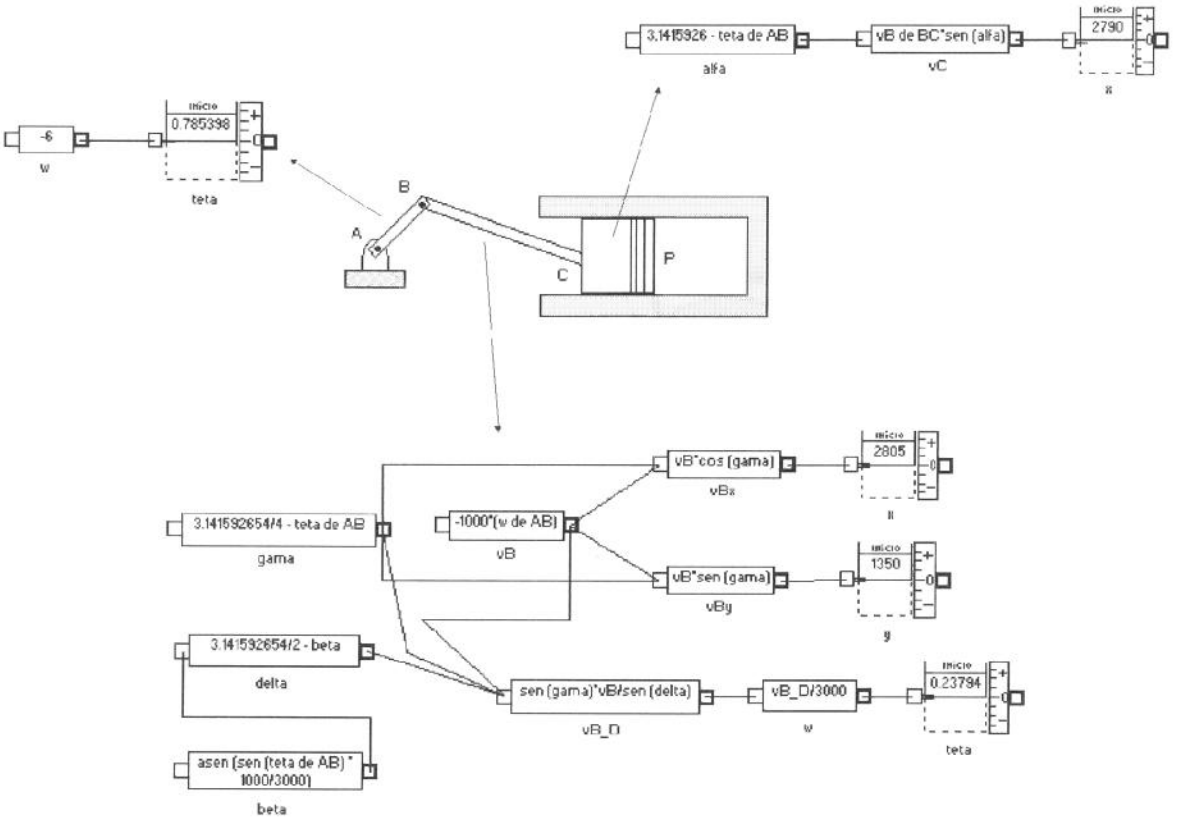
B.12 - TURBINA

Uma turbina recebe um jato de água a uma velocidade v perpendicular às suas palhetas. O modelo supõe que o fluxo de água que sai do bocal é totalmente encaminhado às palhetas.



B.13 - SISTEMA MOTOR

Em um sistema motor a manivela AB possui uma velocidade angular constante no sentido horário.



Este modelo tem um interesse especial por envolver diversos tipos de movimento: a manivela AB está sob movimento de rotação em torno do ponto A; a biela BC sofre um movimento plano geral, ou seja, sofre simultaneamente uma translação e uma rotação em torno do ponto B; o pistão P está submetido a uma translação horizontal.

Referências Bibliográficas

REFERÊNCIAS BIBLIOGRÁFICAS

Asymetrix Corporation (1991-a). *Using openScript: An Introduction and Reference to the openScript Language.* Washington, USA.

Asymetrix Corporation (1991-b). *Dll Reference: A guide to Using Dynamic Link Libraries.* Washington, USA.

Asymetrix Corporation (1991-c). *Using ToolBook: A Guide to Building and Working With Book.* Washington, USA.

Asymetrix Corporation (1991-d). *ToolBook Ideas: An Author's Introduction to Programming in ToolBook.* Washington, USA.

Baranauskas, M. C. C. e Oliveira, O. L. (1994-a). *Estratégias para o Design de Ambientes Computacionais para Modelagem.* Anais do Quinto Simpósio Brasileiro de Informática na Educação. Porto Alegre, RS, pp. 25 - 37.

Baranauskas, M. C. C. e Weller, D. (1994-b) *Extended Interfaces for Learning Environments Based on Simulation.* Accepted proposal for presentation at the ED-MEDIA/AACE, Vancouver, Canada.

Baranauskas, M. C. C. e Oliveira, O. L. (1995). *Domain-Oriented Modelling - a Balance between Simulation and Programming.* Procceedings of the Sixth IFIP World Conference Computers in Education. Birmingham, july, 1995.

Beer, P. e Johnston Jr., E. R. (1980). *Mecânica Vetorial para Engenheiros, Vol. 1, Estática.*. São Paulo, SP: McGraw-Hill.

Beer, P. e Johnston Jr., E. R. (1991). *Mecânica Vetorial para Engenheiros, Vol. 2, Cinemática e Dinâmica.* São Paulo, SP: Makron Books.

Bronson, R. (1977). *Moderna Introdução as Equações Diferenciais.* Traduzido por Farias, A. A.. São Paulo, SP: McGraw-Hill.

Bruner, J. S. (1966). *Toward a theory of Instruction.* Traduzido com o título *Uma nova teoria de aprendizagem* por Ribeiro, N. L.. Rio de Janeiro, RJ: Block Editores S/A.

Buxton, W. (1986). *There's More to Interaction Than Meets The Eye: Some Issues in Manual Input,* in *User Centered System Design: New Perspectives on Human-Computer Interation,* Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 319 - 337.

deJong, T. (1991). *Learning and Instruction with Computer Simulations.* Education & Computing, 6: 217 - 229.

diSessa, A. A. (1982). *Unlearning Aristotelian Physics: A Study of Knowledge-Based Learning.* Cognitive Science, 6: 37 - 75.

diSessa, A. A. e White B. Y. (1982). *Learning Physics from a Dynaturtle.* Byte, August: 324.

Draper, S. W. e Norman, D. A. (1986). *Introduction,* in *User Centered System Design: New Perspectives on Human-Computer Interation,* Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 1- 6.

Eberhard, J. P. (1970). *We Ought to Know the Diference,* in *Engineering Methods in Environmental Design and Planning,* Moore, G. T. (ed). Massachusetts: MIT Press, pp. 364 - 365.

Eisenberg, M. (1991). *Programable Applications: Interpreter Meets Interface.* Cambridge, MA: MIT Press.

Forrester, J. W. (1968). *Principles of Systems.* Cambridge, MA: Wright-Allen.

Goodyear, P. (1987). *A Toolkit approach to computer-aided systems modelling,* in *Trends in Computer Assisted Instruction,* Lewis, R. e Tagg, E. (eds). Oxford: Blackwells, pp. 88 - 96.

Hassel, D. J. e Webb, M. E. (1990). *Modus: The Integrated Modelling System.* Computers & Education, 15 (1-3): 265 - 270.

High Performance Systems Inc. (1994). *A guide to demonstration version of Ithink - The Visual Thinking Tool for the 90's:* Hanover, USA.

Hooper, K. (1986). *Architectural Design: An Analogy,* in *User Centered System Design: New Perspectives on Human-Computer Interation,* Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 9 - 23.

Hutchins, E. L., Hollan, J. D. e Norman, D. A. (1986). *Direct Manipulation Interfaces,* in *User Centered System Design: New Perspectives on Human-Computer Interation,* Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 87 - 124.

IBM (1992). *QMF - Query Management Facility version 3 release 1.1-* San Jose: CA, USA.

Imagine That Inc. (1993). *A guide to demonstration version of Extend - Performance modeling for decision support:* San Jose: CA, USA.

Korth, H. F. e Silberschatz, A. (1989). *Sistemas de Banco de Dados.* São Paulo, SP: Makron Books.

Laurel, B. K. (1986). *Interface as Mimesis,* in *User Centered System Design: New Perspectives on Human-Computer Interation,* Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 67 - 85.

LindField, G. R. (1992). *The Role of the STELLA SoftWare Package in Simulation and Modelling.* International Journal of Mathematical Education Science and Techonology, 23 (6): 865 - 880.

McCloskey, M. (1983). *Naive Theories of Motion,* in *Mental Models,* Gentner, D. e Stevens, A. L. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 299 - 324.

Microsoft Corporation (1990). *Microsoft Windows Software Development Kit version 3.0 - Reference for the MS-DOS and PC-DOS Operating Systems, Volume 1 and 2.* Washington, USA.

Microsoft Corporation (1994). *Microsoft Excel - Guia do Usuário.* Washington, USA.

Miller, R., Brough, D. R. e Briggs, J. H. (1991). *Declarative Software Tools for Learners.* Proceedings of Sixth International PEG Conference. Rapallo, Italy, pp. 177 - 184.

Miller, R., Ogborn, J., Briggs, J., Brough, D., Bliss, J., Boohan, R., Mellar, H. e Brosnan, T. (1993). *Educational Tools for Computational Modelling.* Computers & Education, 21(3): 205 - 261.

Mills, H. D., Linger, R. C. e Hevner, A. R. (1986). *Principles of Information Systems Analysis and Design.* Orlando, Florida: Academic Press.

Millwood, R. e Stevens, M. (1990). *What is The Modelling Curriculum?* Computers & Education, 15 (1-3): 249 - 254.

Norman, D. A. (1986). *Cognitive Engineering*, in *User Centered System Design: New Perspectives on Human-Computer Interation*, Norman, D. A. e Draper, S. W. (eds). Hillsdale, NJ: Lawrence Erlbaum Associate Publishers, pp. 31- 61.

Oracle Corporation (1986). *SqlPlus version 3.0 - Users Guide.* Belmonte, California, USA.

Papert, S. (1980). *Mindstorms: Children, Computer and Powerful Ideas.* New York, NY: Basic Books.

Papert, S. (1986). Construcionism: A New opportunity for Elementary Science Education. A Proposal to the National Science Foundation, Massachusetts Institute of Technology. Media Laboratory, Epistemology and Learning Group, Cambridge, MA.

Pressman, R. S. (1987). *Software Engineering.* New York: NY: McGraw-Hill.

Ratcliffe, C. P. (1992). *A Consistent and Understandable Method of Teaching Newton's Laws of Motion for the Solution of Rigid-Body Problems.* Physics Education, 27: 327-332.

Riley, D. (1990). *Learning About Systems by Making Models.* Computers & Education, 15 (1-3): 255 - 263.

Roberts, N., Anderson D., Deal, R., Garet, M. e Shaffer, W. (1983). *Introduction to Computer Simulation.* New York, NY: Addison-Wesley.

Ross, P. (1986). *Modelling as Method of Learning Physical Science and Mathematics. In Designing Computer-Based Learning Materials,* Weinstock, H. & Bork, A. (Eds.). Nato ASI Series, Series F: Computers And Systems Sciences (Vol. 23). Berlin/Heidelberg:Springer Verlag, Vol **23**, pp. 95-118.

Schecker, H. (1993). *Learning Physics by Making Models.* Physics Education, 28: 102 - 106.

Singer, F. L. (1977). *Mecânica Para Engenheiros - Dinâmica.* São Paulo, SP: Harper & Row do Brasil.

Squires, D. (1987). *Providing Computer-Based Experience for Learning Physics.* Physics Education, 22: 239 - 243.

Valente, J. A. (1993). *Diferentes Usos do Computador na Educação,* em *Computadores e Conhecimento: Repensando a Educação,* Valente, J. A. (organização). Campinas, SP: Gráfica Central da UNICAMP, pp. 1 - 23.

Webb, M. E. (1994). *Beginning Computer-Based Modelling in Primary Schools.* Computers & Education, 22 (1-2): 129 - 144.

Winograd, T. (1995). *From Programming Environments to Environments for Designing.* Comunications of the ACM, june, 38 (6): 65-74.

Yourdon, E. (1990). *Análise Estruturada Moderna.* Rio de Janeiro, RJ: Editora Campus.