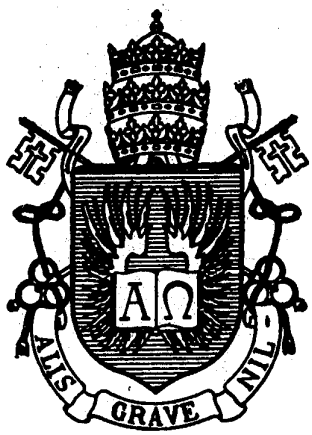


PUC

Série: Monografias em Ciência da Computação
Nº 9/79

DESENVOLVIMENTO ESTRUTURADO DE SISTEMAS AUTOMATIZADOS

por

Arndt von Staa

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 – CEP-22453
RIO DE JANEIRO – BRASIL

PUC/RJ - DEPARTAMENTO DE INFORMÁTICA

Série: Monografias em Ciência da Computação

Nº 9/79

Editor da Série: Daniel A. Menascé Junho, 1979

DESENVOLVIMENTO ESTRUTURADO DE SISTEMAS AUTOMATIZADOS*

por

Arndt von Staa

* Trabalho parcialmente financiado pela FINEP

Índice

Resumo

São examinados os problemas relacionados com o desenvolvimento de sistemas automatizados. O exame é conduzido observando este problema como um todo. Não serão estudados detalhes de métodos e técnicas de planejamento, de análise econômica, de especificação, de produção, de testes e de operação.

São definidos atributos de utilidade e de qualidade de sistemas automatizados. São descritos, também, fatores que condicionam o grau de alcance destes atributos, bem como a rentabilidade de sistemas automatizados. É descrito um ciclo de vida de sistemas automatizados onde são apresentados os objetivos e produtos a serem gerados por fase deste ciclo de vida.

Palavras Chave

Atributos de qualidade, atributos de utilidade, benefícios, ciclo de vida, custos, desenvolvimento controlado, entidades participantes, medidas de qualidade, medidas de utilidade, rentabilidade, qualidade, utilidade.

Abstract

The problems related to the development of automated systems are examined. The study is conduced observing always the whole development problem, without considering details such as methods and techniques for planning, economic analysis, specification, construction, test and operation.

Quality and utility attributes, software profitability and development conditions are described. Also a software system life cycle is described, where objectives and products per phase are specified.

Keywords

Benefits, controlled development, costs, life cycle, participating entities, profitability, quality, quality attributes, quality measurement, utility, utility attributes, utility measurement.

1 Introdução.

O objetivo deste estudo é o de identificar os problemas a serem resolvidos quando do desenvolvimento de sistemas automatizados rentáveis. Tais sistemas são complexos, tendem a ser caros e nem sempre mostram-se tão úteis quando prontos, quanto era imaginado ao início do desenvolvimento. Diversos são os fatores que levam os sistemas automatizados a terem utilidade e qualidade duvidosa. Nosso objetivo, no entanto, não é o de simplesmente identificar falhas encontradas em sistemas já desenvolvidos. Pelo contrário, o objetivo é o de identificar fatores que, se devidamente observados, levem naturalmente a sistemas úteis e rentáveis, minimizando, ou eliminando, a ocorrência de tais falhas.

O exame do desenvolvimento de sistemas automatizados será conduzido observando-se este problema como um todo. Não serão abordados detalhes com relação a métodos e técnicas tais como planejamento, análise econômica, especificação, construção, teste e operação de sistemas automatizados.

Sistemas automatizados são formados por diversos componentes, ou subsistemas, que são: software, hardware, arquivos, procedimentos não automatizados, documentação e, possivelmente, outros sistemas automatizados. Os componentes hardware, software e arquivos são, usualmente, chamados de sistemas de computação. Tais sistemas, porém, não podem existir de forma isolada, necessitando sempre de procedimentos não automatizados para que possam operar.

Sistemas automatizados são especificados e desenvolvidos para atender determinada necessidade de processamento de dados. São exemplos de sistemas automatizados: os componentes automatizados de sistemas administrativos; sistemas operacionais; sistemas de controle de processo; sistemas interligados formando uma rede de teleprocessamento; sistemas de processamento de dados distribuídos etc.

Atualmente tem-se tornado frequente o projeto de sistemas automatizados onde o componente hardware a ser utilizado não é previamente definido. Isto tem-se feito notar inclusive em aplicações comerciais. Esta tendência é a razão principal pela qual optamos por uma definição recursiva. Apesar do desenvolvimento e/ou seleção de hardware poder ser parte do desenvolvimento de um sistema de computação específico, enfatizaremos neste texto os aspectos relativos ao desenvolvimento de software.

Sistemas automatizados com finalidades industriais devem possuir elevado grau de qualidade, devendo ser úteis, rentáveis e duráveis. Entendemos neste texto rentabilidade como sendo o valor atual a cada instante da vida do sistema, onde valor atual é a diferença entre o valor acumulado de todos os benefícios auferidos e o valor acumulado de todos os custos

incorridos até este instante. A rentabilidade ao final da vida do sistema é o lucro, ou prejuízo, oriundo da utilização deste sistema.

Ao justificar-se a necessidade de um novo sistema, deve-se examinar a evolução esperada para a sua rentabilidade. Contudo, diversos sistemas serão desenvolvidos tendo por base decisões estratégicas, onde o exame da rentabilidade é relegado a um segundo plano. No entanto, mesmo estes sistemas devem satisfazer a critérios de utilidade e qualidade previamente estabelecidos, afim de que alcancem os objetivos estratégicos desejados.

Neste texto não abordaremos em detalhe os métodos e as técnicas para a análise econômica de projetos de desenvolvimento de sistemas automatizados. Tais métodos usualmente baseiam-se na determinação, ou avaliação, do valor presente de cada alternativa, tomando por base o comportamento estimado da rentabilidade, requerendo ainda uma análise da sensibilidade dos parâmetros e hipóteses utilizadas. Maiores detalhes quanto a tais métodos podem ser encontrados em [Bussey78].

Em estudos recentes, diversos autores propuseram soluções ao problema de desenvolvimento de sistemas automatizados, e cujos aspectos principais apresentamos a seguir [Metzger73, Gildersleeve74, Gildersleeve78, Brooks75, Benjamin71, Hice74, Mangold74, Zelkowitz78, Boehm76]:

a- definição do ciclo de vida do sistema a desenvolver;

b- produção continuada de documentação, desde o início do desenvolvimento, e utilização desta documentação como guia do desenvolvimento, ao invés de produzí-la após o desenvolvimento;

c- verificação contínua durante todo o desenvolvimento do grau de satisfação dos diversos requisitos de qualidade e utilidade desejados para o produto;

d- participação intensiva do usuário nas atividades de especificação e verificação;

e- desenvolvimento por refinamentos sucessivos, isto é, por adição de detalhes à medida que prossegue;

f- controle disciplinado e previamente estabelecido dos produtos (documentos, dados de teste, programas, especificações etc.);

g- determinação a priori dos pontos de controle de desenvolvimento que permitam efetivo acompanhamento e controle do desenvolvimento do sistema;

h- utilização de menos pessoas melhor qualificadas durante a definição, construção e implantação do sistema.

Os métodos apresentados na literatura, propõem um desenvolvimento "top down", tendo por base um plano de desenvolvimento. O método apresentado neste texto segue em essência estes mesmos princípios. Procuramos, porém, definir com mais rigor como é efetuado o desenvolvimento, como são controlados os produtos, e quais os produtos intermediários e finais a serem entregues por fase do desenvolvimento.

Observados em sua essência, a definição prévia dos requisitos de utilidade e qualidade; a posterior verificação se estes requisitos foram alcançados; a otimização dos requisitos dentro das limitações de ordem econômica; e o desenvolvimento por fases, em pouco diferem da prática atual de engenharia. Ou seja, o desenvolvimento de sistemas automatizados é análogo ao desenvolvimento de produtos industriais. Contudo, como ususalmente sistemas automatizados são internos à empresa, e como ainda inexiste uma tradição de indústria de software competitiva e eficiente, raras vezes propostas de desenvolvimento de sistemas automatizados são submetidas ao mesmo grau de análise econômica como é o caso no desenvolvimento de novos produtos.

É nosso interesse esboçar o que se entende por produção de sistemas automatizados, sob o ponto de vista de uma disciplina de engenharia, em contraste com o desenvolvimento artesanal destes sistemas. Os seguintes pontos devem ser observados ao produzir-se software de forma industrial:

a- otimização dos requisitos de utilidade e de qualidade dentro dos fatores limitantes do desenvolvimento, em especial os fatores custo e tempo.

b- existência de documentação fidedigna e que sirva efetivamente para orientar o desenvolvimento do software e sua posterior manutenção.

c- especificação do produto, produção de acordo com a especificação e certificação contínua do produto (confronto do produto com a especificação).

d- planejamento do desenvolvimento, acompanhamento e controle segundo o plano de desenvolvimento.

e- exame e controle do sistema quanto à satisfação dos requisitos de utilidade, de qualidade, de proteção e segurança, onde este controle é efetuado continuamente desde o início do projeto.

Nos capítulos a seguir iremo-nos aprofundar nos itens a (capítulos 2 e 3) e b (capítulo 4). Por razões de escopo, os demais itens não serão examinados neste texto.

2 Utilidade e Qualidade de Software.

Sistemas automatizados devem ser úteis. Apesar disto ser o óbvio, não são raros os sistemas cuja utilidade é no mínimo duvidosa.

A utilidade de um sistema automatizado não depende somente dos resultados que produz. Diversos outros fatores influenciam a utilidade destes sistemas, por exemplo: a capacidade do sistema em atender ao volume de processamento requerido dentro de determinados limites de tempo, a capacidade de armazenamento de dados, a facilidade com que se pode incrementar estas capacidades, a facilidade de operação, a clareza e inteligibilidade dos relatórios de saída, a completeza e correção destes relatórios, etc.

A utilidade é, portanto, um conjunto de propriedades a serem satisfeitas em maior ou menor grau, de forma que o sistema de computação atenda às necessidades do usuário durante toda a vida útil deste sistema.

Queremos enfatizar aqui que, de acordo com a definição acima, os atributos de utilidade não são medidos em unidades monetárias. Na seção 3.2 estudaremos em maior profundidade o interrelacionamento da qualidade e da utilidade com custos e benefícios.

De maneira semelhante, a qualidade de sistemas automatizados é um conjunto de propriedades a serem satisfeitas em maior ou menor grau, de forma a tornar a implementação do sistema adequada segundo o ponto de vista do usuário.

Deve ser observado, que tanto a utilidade quanto a qualidade de sistemas automatizados são atributos que não podem ser definidos à revelia do usuário, e sim requerem a participação deste, tanto para especificar os requisitos desejáveis e os mínimos toleráveis para os diversos atributos de utilidade e qualidade, quanto para avaliar os produtos a fim de determinar se estes requisitos foram atingidos.

Os atributos de utilidade e qualidade se confundem. De uma forma grosseira poderíamos separá-los, dizendo que os atributos de utilidade medem propriedades externas, enquanto que os de qualidade medem propriedades internas (implementacionais) ao sistema automatizado. Neste texto não procuraremos estabelecer uma distinção entre os atributos de qualidade e de utilidade. Ou seja, todos os atributos estudados devem sempre ser examinados, tanto como atributos de qualidade, quanto como atributos de utilidade.

Os atributos de utilidade variam de sistema para sistema, em casos extremados podendo variar de usuário para usuário. Foge ao escopo deste trabalho um exame pormenorizado de atributos de utilidade, o que nos limita ao exame de somente

alguns dos principais atributos. Detalhes quanto à determinação dos atributos de utilidade e da sua importância em um determinado sistema podem ser encontrados em [Allen78, Staa79].

As utilidade e qualidade de sistemas automatizados são medidas compostas por atributos. Cada um destes atributos possui peso próprio indicando a sua relevância no cômputo geral da qualidade e da utilidade de cada sistema em particular. Cada atributo deve ser objetivamente mensurável, ou avaliável.

A qualidade e a utilidade de um produto de software devem ser apresentadas listando-se os atributos, uma vez que a agregação dos atributos em uma única medida, por exemplo através de soma ponderada, pode causar uma perda excessiva de informação.

A ação de verificar se os requisitos de qualidade e de utilidade foram satisfeitos, será chamada de aprovação.

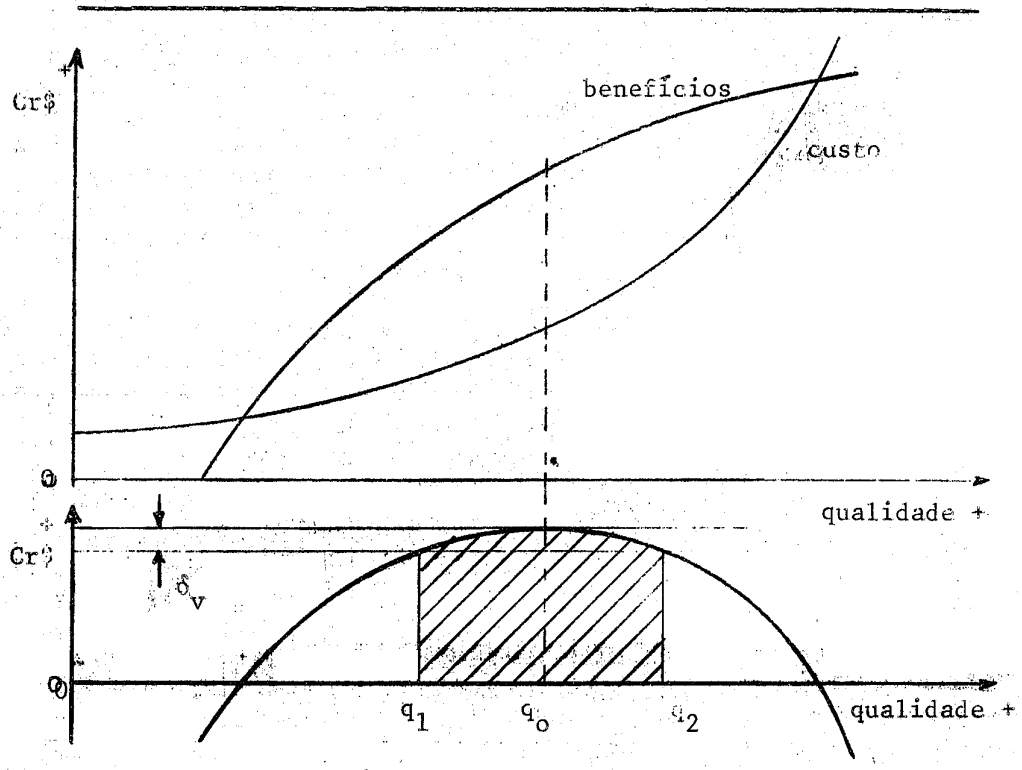
Ao iniciar-se um projeto de desenvolvimento de um sistema automatizado, devem ser determinados quais os atributos de qualidade e utilidade a serem enfatizados, quais os pesos destes atributos, e quais os níveis desejáveis e mínimos que estes atributos deverão possuir para que se possa determinar se os objetivos do sistema foram efetivamente alcançados. Os valores mínimos e os pesos requeridos para os atributos de qualidade e utilidade variam de produto para produto(1), sendo que, quanto mais elevados forem estes requisitos, mais custará o desenvolvimento do componente em questão.

O desejo de alcançar um elevado grau de qualidade traz consigo maiores custos de desenvolvimento, e uma redução dos custos operacionais (custos devido a falhas, custos de manutenção, custos devido a fraudes, etc.). Como os custos operacionais tendem a ser os de maior peso (60% ou mais [Zelkowitz78]), é de se esperar que uma redução dos custos operacionais (operação e manutenção) mais do que compense o aumento dos custos de desenvolvimento e, assim, contribua para tornar mais rentável o sistema. Além disso é esperado que um sistema de elevado grau de qualidade seja mais durável e, portanto, tenha uma rentabilidade maior também por esta razão.

Na figura 2.1 apresentamos o aspecto genérico das variações dos custos de desenvolvimento e dos benefícios com relação à qualidade. Na seção 3.2 examinaremos em maior profundidade os aspectos de custos e benefícios relacionados com níveis de qualidade e de utilidade.

---

(1) Entendemos produtos como sendo quaisquer componentes de um sistema automatizado. Estes componentes podem ser compostos e, eventualmente, ser o próprio sistema automatizado.



q_o grau de qualidade ótimo

δ_v tolerância de valor q_1 q_2 tolerância do grau de qualidade

Figura 2.1 Aspecto genérico da variação de custos e benefícios com relação à qualidade.

A curva da figura 2.1 deve ser traçada por atributo, uma vez que a qualidade é uma medida composta. Por exemplo, se a qualidade agregada for medida como soma ponderada dos diversos atributos, é perfeitamente possível uma variação de pesos e requisitos tal que, para valores diferentes de um determinado conjunto de requisitos, o valor da qualidade agregada permaneça o mesmo. Como, porém, a variação dos custos é diferente para cada um dos atributos, poderemos obter custos diferentes relativos a alternativas almejando um mesmo grau de qualidade agregada.

Diversos são os fatores que dificultam medir-se a utilidade e a qualidade de sistemas de computação. Em particular, estes sistemas não podem ser vistos de forma isolada. Eles sempre estarão correlacionados com outros sistemas, muitos dos quais não são automatizados. Por exemplo, ao projetar-se e implementar-se uma linguagem de programação, deve-se levar em conta o "sistema de preparação e de certificação de programas" no qual estará inserida esta linguagem. Pode ocorrer, então, que a otimização de um determinado sistema de software não corresponda, também, uma otimização do sistema automati-

zado do qual este sistema é componente. Ou seja, a utilidade e a qualidade de um sistema automatizado dependem do grau de resolução com o qual o sistema é observado e medido.

Sistemas automatizados poderão atender satisfatoriamente às necessidades de uma ou mais classes de usuários, porém não às de todas as classes. Por exemplo, um sistema financeiro poderá estar atendendo adequadamente às necessidades imediatas da gerência financeira, porém não estar atendendo satisfatoriamente aos auditores que desejam examinar e controlar este sistema de tempos em tempos. Ou seja, a utilidade e a qualidade de um sistema automatizado dependem do observador que mede estes atributos. Por esta razão é imprescindível identificar-se quais são os usuários do sistema e fazê-los determinar os requisitos de utilidade e qualidade mínimos, bem como fazê-los participar da especificação e avaliação do sistema.

A medida, ou avaliação, dos atributos de qualidade e de utilidade pode ser mais ou menos difícil. Alguns atributos poderão ser efetivamente medidos, outros poderão ser avaliados com facilidade, outros, no entanto, somente poderão ser avaliados por pessoas altamente qualificadas, uma vez que esta avaliação requer conhecimentos especializados por parte do avaliador [Boehm78].

Medidas sempre são feitas em termos de faixas de tolerância. Isto decorre da imprecisão própria ao instrumento utilizado para efetuar a medida. As faixas de imprecisão de medidas são mais amplas quanto maior a dificuldade encontrada ao efetuar a medida ou avaliação. Isto implica em que os limites mínimos dos atributos de utilidade e de qualidade estejam suficientemente acima do mínimo tolerável, para que se possa levar em conta uma parcela razoável do risco de erro de medida ou avaliação.

Atributos de utilidade e de qualidade podem ser difíceis de medir ou avaliar devido à falta de conhecimento de como fazê-lo. Por exemplo, como medir correção ou robustez? Em geral são propostas diversas técnicas, não necessariamente coincidentes para efetuar estas medidas. Nestes casos é sugerido medir-se utilizando estas diversas técnicas e comparar-se os resultados, esperando obter desta forma uma indicação do grau de confiança da medida.

Do dito até agora conclui-se que muito ainda precisa ser feito para que se possa definir precisamente o que vem a ser utilidade e qualidade de sistemas automatizados e como efetuar medidas e avaliações dentro de limites de custo e precisão adequados. Por estas razões, a medição da qualidade e da utilidade é, segundo alguns, considerada ser de pouca valia.

Defendemos aqui um ponto de vista oposto. Para nós, apesar de ser difícil medir ou quantificar e, apesar de dispormos de insuficiente precisão de definição, a utilidade e a qualidade de sistemas automatizados devem ser avaliadas, e não somente isto, os níveis a serem atingidos por estes atributos devem ser determinados antes mesmo de iniciar-se o desenvolvimento do sistema automatizado [Gilb77, Boehm78, Royce75, Zelkowitz78].

Definindo-se a priori os requisitos de qualidade e utilidade, torna-se mais precisa a estimativa de quanto custará o desenvolvimento do sistema automatizado, quais os benefícios que trará quando em operação, e qual será o grau de satisfação que os diversos usuários poderão vir a ter com relação ao sistema uma vez que este esteja em operação.

A seguir descreveremos os atributos mais importantes que compõem a qualidade e a utilidade de sistemas automatizados. A lista não pretende ser exaustiva, ou seja, certamente existirão atributos não relacionados nesta lista.

Diversos atributos de qualidade e de utilidade são, na realidade, atributos compostos (por exemplo: eficácia, flexibilidade, e c.) e devem ser apresentados sob a forma de uma lista de sub-atributos. Esta decomposição é muitas vezes dependente do próprio sistema sendo proposto ou extendido. Além disso um estudo pormenorizado destes sub-atributos obliteraria a visão geral do problema de medição de atributos devido ao excesso de detalhe assim introduzido. Os atributos compostos devem ser definidos em função de seus sub-atributos, uma vez que, também neste caso, uma agregação numa única medida causaria excessiva perda de informação.

Os atributos listados não são necessariamente independentes, ou seja, à variação de um determinado atributo poderá corresponder a variação de outros atributos.

Para cada atributo apresentaremos uma definição e uma explicação resumida. Neste texto não procuraremos decrever como medir estes atributos, nem, tampouco, procuraremos definir a natureza da medida. Esforços neste sentido estão descritos em [Gilb77, Boehm78].

Em algumas ocasiões seremos forçados a lançar mão de atributos ainda não definidos ao explicarmos o significado de outro atributo. Cremos, porém, que isto não cause maiores inconvenientes, uma vez que as definições são próximas aos conceitos intuitivos destes atributos.

2.1 Atributos da Classe "Confiabilidade de Processos".

Esta classe de atributos mede o grau de confiança que um usuário pode depositar nos programas e procedimentos que compõem o sistema.

Sistemas automatizados deverão ter tolerância a falhas, imprecisões, a erros humanos etc. Devemos saber, porém, o limite destas falhas, imprecisões, erros etc., pois somente assim poderemos esperar que o usuário confie nos resultados produzidos pelo sistema quando em operação. Como estes atributos dependem do ponto de vista do observador, eles deveriam ser determinados por classe de usuário.

Correção

Definição

é a medida da equivalência entre o sistema implantado e a especificação deste sistema, quando submetido a condições de operação normais.

Discussão

um software 100% correto não possui erros de codificação e obedece exatamente às especificações. Caso estas estejam incorretas (atributo eficácia), isto não altera a propriedade do software estar 100% correto com relação a estas especificações, apesar de apresentar erros do ponto de vista do usuário. Ou seja, para que um software seja 100% "correto" do ponto de vista do usuário, é necessário que seja 100% eficaz e 100% correto. A justificativa para esta decomposição, é o fato de erros de programa serem atribuíveis a erros de implementação e a erros de especificação, onde estas duas classes de erros são de natureza distinta.

No estágio de desenvolvimento atual, não é razoável esperar-se que o software esteja 100% correto. A capacidade de tolerar falhas próprias e do meio em que este software está localizado, é medida pelos atributos robustez e ductilidade a serem definidos mais tarde.

A correção de software é medida através da operação de certificação. Esta operação é composta pelas operações verificação - prova parcial ou total da correção - e validação - comprovação da correção através de experimentação (testes) [Zelkowitz78].

Como já foi dito, um sistema automatizado é composto por programas e procedimentos, ambos devendo satisfazer a critérios de correção. Enquanto se pode admitir sem maiores danos, que a taxa de falhas de hardware é negligenciável, o mesmo não vale para o ser humano "executor" dos procedimentos. As falhas humanas podem advir de uma série de fatores, desde a simples negligência, até a tomada de decisão errada e/ou fraude

deliberada. Consequentemente, os procedimentos devem ser projetados com suficiente redundância para que o volume de falhas que passam despercebidas seja suficientemente pequeno. Cabe observar que os resultados de um sistema automatizado dependem de todos os seus componentes, portanto de nada vale um grande esforço para garantir a qualidade do software, se usuários podem facilmente comprometer ou destruir a qualidade dos resultados através de falhas de operação, não cumprimento de prazos, fraudes, negligência etc.

Robustez

Definição

é a medida da capacidade do sistema implantado em detectar e proteger-se contra condições de operação adversas.

Discussão

quando submetido a condições adversas, ou seja, dados errados, catástrofes, panes, falhas de equipamento, erros de software, erros de uso, fraudes, sabotagens, negligência etc., um sistema automatizado robusto mantem arquivos e resultados em estados adequados, alertando o usuário quanto à existência de condições adversas. Em casos extremados, condições adversas podem surgir como consequências de ações hostis, tais como fraudes, sabotagens etc.

A robustez não mede a capacidade de recuperação de erros, mede somente a capacidade de detecção da ocorrência de condições adversas durante a operação do sistema.

Erros de uso podem ser acidentais ou resultados de tentativas de fraude. Panes podem ser causadas por fraudes, por catástrofes e por falhas de software, hardware e/ou do ambiente de suporte no qual o sistema opera. Para que o sistema se torne robusto, a redundância necessária para gerar a capacidade de detectar condições de operação adversas deve ser incorporada ao sistema desde o início do projeto.

Um software robusto pode, por exemplo, suspender o processamento, desde que o usuário seja alertado quanto a esta suspensão, suas causas e, também, desde que seja possível recomeçar ou continuar o processamento a partir de um ponto próximo ao de suspensão, sem que as propriedades de qualidade de arquivos e resultados sejam afetadas [Magalhães79].

Suspender o "processamento" de um procedimento, ou seja, uma sucessão de operações manuais, já é bem mais difícil, uma vez que depende do julgamento e do desejo do usuário em executar corretamente este procedimento.

Isto reforça a necessidade da existência de redundância de tal forma que seja possível descobrir falhas de execução de procedimentos através de vias paralelas de processamento. É claro que o grau de redundância a ser introduzido depende dos riscos e das consequências para a instituição dos diferentes erros e fraudes na operação do sistema automatizado [Parente78].

A robustez é medida de forma análoga à correção, isto é, através de certificação. No entanto, o objetivo da medição de robustez difere do objetivo da medição de correção, no sentido em que, para medir robustez, desejamos determinar a existência e a extensão dos conjuntos de dados errados, panes e condições adversas que efetivamente levem o sistema a comportar-se de modo imprevisível ou inadequado. Se estes conjuntos não existirem, o sistema será 100% robusto.

Ductilidade

Definição

é a medida da habilidade do sistema em operar adequadamente mesmo quando ocorrerem erros de uso, ou condições de operação adversas.

Discussão

em muitas ocasiões deseja-se que as porções corretas do sistema continuem operando, mesmo quando descobertos erros de uso, panes de componentes e/ou do suporte, fraudes, catástrofes, erros de dados, etc.

A ductilidade difere da robustez, uma vez que esta mede a capacidade de determinar a ocorrência de condições adversas, enquanto que a ductilidade mede a capacidade de continuar operando sob estas condições ("graceful degradation").

A inclusão de redundâncias, previsões para contingências e a capacidade de recuperação de erros são exemplos de esforços no sentido de aumentar a ductilidade.

Disponibilidade

Definição

é a medida da capacidade do sistema em atender a estímulos de demanda de serviço tão logo estes ocorram.

Discussão

os componentes dos sistemas automatizados estão sujeitos a uma determinada demanda de serviço. Esta demanda deve ser satisfeita satisfatoriamente, mesmo no evento da ocorrência de condições adversas. Em alguns casos, pode ser prevista uma redução controlada da capacidade de atendimento à demanda.

A disponibilidade é usualmente medida em termos do percentual do tempo útil (não incluindo paradas programadas) durante o qual serviços são solicitados e prontamente atendidos. No entanto, esta medida pode variar de acôrdo com o propósito de cada componente. Por exemplo, em um sistema de controle de processo contínuo, a disponibilidade costuma ser medida em relação ao tempo total, e não em relação ao tempo útil, uma vez que, neste caso, mesmo paradas programadas (manutenção) são sujeitas a restrições.

Medir a disponibilidade é particularmente importante com relação aos componentes automatizados, uma vez que a disponibilidade mede a tolerância do sistema com relação a paradas normais e/ou devidas a degradações acidentais.

A disponibilidade difere da ductilidade no sentido em que esta mede a capacidade de operar em estado degradado, enquanto que a disponibilidade mede a "presença" do sistema.

A disponibilidade difere, também, da eficiência de processamento, no sentido em que esta mede a evolução dos tempos de resposta com relação à evolução da demanda, enquanto que a disponibilidade mede a capacidade do sistema em dar partida no atendimento a um estímulo, independentemente da carga do sistema. Cabe salientar que faz parte do atendimento a um estímulo de demanda, recebê-lo e registrá-lo em uma fila de espera para efetivo atendimento no futuro.

Eficiência de Processamento

Definição

é a medida (inversa) do esforço necessário para operar-se o sistema.

Discussão

sistemas automatizados devem ter custo operacional direto baixo. Este custo depende do consumo de recursos computacionais (porte do equipamento, consumo de memória principal e auxiliar, tempo de processamento, etc.), de recursos de comunicação (linhas de comunicação, transporte de formulários, relatórios e volumes, etc.), de recursos físicos (arquivos convencionais, formulários, dispositivos especiais, etc.), e de mão de obra (coleta, transcrição e correção de dados, execução de procedimentos, treinamento, etc.).

Além dos custos de operação diretos, existem outros, tais como a amortização do investimento, custo de manutenção, custos decorrentes de panes e/ou fraudes etc. O desejo de reduzir substancialmente estes custos indiretos, geralmente expressivos, pode levar a uma re-

dução da eficiência de processamento, causando uma pequena elevação dos custos operacionais diretos. Em outras palavras, a eficiência de processamento deve ser sempre examinada dentro do contexto de custos totais e não somente no contexto dos custos operacionais diretos, uma vez que a enfatização da eficiência de sistemas de software tende a levar a custos globais mais elevados ao invés de reduzí-los.

O custo operacional direto varia com relação à carga de processamento do sistema. Esta variação tende a ser não linear no caso de sistemas automatizados. Nestes sistemas ocorre frequentemente que, após determinado ponto de saturação, a cada incremento de demanda corresponda um incremento substancialmente maior no custo operacional.

Tempestividade

Definição

é a medida da adequação do sistema com relação a restrições de tempo.

Discussão

sistemas são de pouca valia caso não produzam os resultados desejados nas épocas desejadas. Falhas de tempestividade podem ocorrer tanto devido à antecipação demasiada, quanto devido ao atraso demasiado na produção de resultados. As dependências de tempo podem ser bastante exíguas, como no caso de sistemas altamente automatizados, por exemplo sistemas de tempo real ou "on-line", e podem ser relativamente amplas, como é o caso normal de sistemas pouco automatizados, por exemplo sistemas operando em "batch".

Para poder satisfazer a requisitos de tempestividade, muitas vezes é necessário prever-se operações contingentes, ou aumentar a ductilidade do sistema. A satisfação da tempestividade pode provocar, também, o uso não balanceado do sistema computacional. Problemas de tempestividade podem ocorrer tanto em sistemas essencialmente "batch" como em sistemas "on-line". Os problemas usualmente decorrem da exiguidade de tempos de resposta para atender à demanda e/ou ao volume de serviço.

Existem interdependências entre diversos atributos de qualidade e o atributo tempestividade. Por exemplo, a atualização de dados mantidos em arquivos requer maior ("batch") ou menor ("on-line") período de tempo para ser efetuada, a partir do momento em que tenha ocorrido a mudança no mundo real, até que esta mudança seja efetivamente registrada no sistema. Durante este período de tempo, a consistência dos dados é diminuída, uma vez que os valores contidos nos arquivos não refle-

tem o mundo real. Isto pode levar a contratempos em alguns casos, como por exemplo, pode levar à redução da fidelidade dos resultados ou de processamentos utilizando estes dados.

Determinismo

Definição

é a medida da habilidade do sistema em produzir os mesmos resultados quando submetido aos mesmos dados.

Discussão

sistemas automatizados deveriam produzir sempre os mesmos resultados quando submetidos aos mesmos dados. Isto deveria ocorrer mesmo se os dados puderem ser fornecidos em sequências arbitrárias, desde que a ordenação em si não seja um fator importante para o processamento. O determinismo depende, portanto, do intervalo de tempo durante o qual são fornecidas sequências de dados cuja ordem é arbitrária.

O determinismo frequentemente não é alcançado devido a erros contidos nos programas, a julgamentos efetuados de forma diferente em ocasiões diferentes, etc. Por outro lado existem sistemas nos quais é até desejado um certo grau de indeterminismo, como por exemplo, quando da seleção aleatória de alternativas, ou da seleção de alternativas baseada em julgamento.

O determinismo de um sistema automatizado pode ser de difícil definição, uma vez que a operação do sistema pode depender de eventos cuja sequência no tempo é impossível de ser controlada. Deve ser observado que o não determinismo dos eventos que influenciam a operação de sistemas (principalmente os sistemas "on-line") tornam virtualmente impossível avaliar-se o determinismo dos resultados por via experimental.

Ao especificar-se o grau de determinismo a ser alcançado, devem ser identificados os componentes inerentemente não determinísticos, além dos que deverão ser asseguradamente determinísticos. No caso em que estes dependem de sequências de eventos não determinísticas, deve ser especificado, também, a maneira pela qual será avaliado o determinismo do componente.

Eficácia

Definição

é a medida da habilidade do sistema em atender às necessidades do usuário.

Discussão

a eficácia mede a adequação das especificações às necessidades do usuário. Ou seja, a eficácia mede requi-

sitos de utilidade não relacionados neste texto, bem como mede a adequação dos valores atribuídos aos requisitos de utilidade aqui relacionados. Como já havia sido dito anteriormente, foge ao escopo deste trabalho um exame detalhado dos possíveis atributos de utilidade.

É bastante difícil separar a eficácia de outros atributos, tais como tempestividade, correção, robustez, consistência, ductilidade, etc. Isto decorre destes atributos afetarem diretamente a satisfação do usuário. Na sua essência, a eficácia mede a "correção" das especificações, ou seja, a utilidade dos resultados e a dificuldade de utilizar o sistema para que resultados úteis possam ser produzidos.

A dificuldade de utilizar o sistema automatizado não é somente fruto da ineficiência dos processos, pode ser fruto, também, da dificuldade em escolher e/ou produzir parâmetros que façam com que o sistema produza resultados úteis. Por exemplo, quando se deseja estatísticas relativas a dados armazenados no sistema, um sistema de baixo grau de dificuldade de utilização possui uma linguagem apropriada para especificar e obter as estatísticas, enquanto que um sistema de elevado grau de dificuldade requer a redação de programas em linguagem de aplicação geral, tais como FORTRAN ou COBOL.

A eficácia é medida em termos de sub-atributos a serem identificados, avaliados e relacionados individualmente ao elaborar-se a proposta de desenvolvimento [Allen78, Staa79].

Mensurabilidade

Definição

é a medida (inversa) do esforço necessário para medir ou avaliar os atributos de qualidade e utilidade do sistema.

Discussão

a mensurabilidade implica na presença de instrumentação e de justificativas, tanto nos programas e procedimentos, quanto na documentação, de forma que permita medir ou avaliar-se objetivamente o grau de satisfação dos diversos atributos de qualidade e utilidade.

Para que o sistema seja mensurável, é necessário que se defina como serão medidos os diversos atributos de qualidade e de utilidade, de tal forma que se obtenha a precisão de medida desejada.

Sistemas automatizados são frequentemente alterados e/ou expandidos, o que torna necessário medir de novo os atributos de qualidade e utilidade. Ou seja, a instrumentação utilizada para medir ou avaliar-se inicialmente o sistema, deve fazer parte integrante deste sistema, mesmo durante a fase de operação.

Auditabilidade

Definição

é a medida da habilidade de avaliar-se continuamente o grau de alcance dos requisitos de proteção e segurança.

Discussão

sistemas automatizados manipulam recursos valiosos, os dados. A perda ou a adulteração dos dados, a divulgação indevida destes, podem causar prejuízos consideráveis para a instituição. Além disso, a instituição depende da operacionalidade do sistema, sendo que qualquer interrupção do processamento pode levar a sérios prejuízos.

Não é suficiente medir o grau de proteção e segurança do sistema somente quando de sua instalação, pois prejuízos decorrentes de falhas de proteção e segurança, são usualmente causados pela própria operação do sistema, sendo necessária, portanto, uma verificação frequente da satisfação destes requisitos.

A auditabilidade requer a presença de instrumentação, processos, dados e resultados redundantes, permitindo avaliar-se o grau de proteção e segurança alcançado e o confronto deste com o desejado [Silva78, Parente79].

A auditabilidade é, em essência, uma medida da capacidade do sistema em prevenir acidentes, catástrofes, panes, fraudes e erros de uso, e/ou minorar os efeitos destas agressões.

A definição de auditabilidade aqui apresentada, difere um pouco de sua definição normal [Parente78]. A auditoria de sistemas requer, entre outras coisas, a observação de fraudes (robustez, consistência), a adequação dos programas às necessidades do usuário (eficácia, correção, integridade) e a capacidade de medir as diversas propriedades do sistema (mensurabilidade). Como tais medidas já foram definidas em separado, optamos aqui por definir a auditabilidade somente em termos da capacidade de estabelecimento da proteção e da segurança do sistema automatizado.

2.2 Atributos da Classe "Confiabilidade de Dados".

Os atributos desta classe medem a confiança que um usuário pode depositar nos dados contidos nos arquivos e nos relatórios produzidos. Os arquivos manipulados por sistemas automatizados podem ser arquivos operados manualmente, por exemplo pastas, fichários etc., arquivos automatizados convencionais, bancos de dados etc. De forma semelhante os relatórios produzidos e/ou utilizados por sistemas automatizados podem ser produzidos manualmente ou por processamento eletrônico. Para que o sistema automatizado funcione adequadamente, tanto os componentes automatizados, quanto os não automatizados devem funcionar adequadamente. Portanto, deve-se examinar a necessidade e prever controles de manipulação também para dados mantidos manualmente.

Existe uma forte interdependência entre esta classe de atributos e os da classe "confiabilidade de processos". Esta interdependência torna virtualmente impossível estabelecer-se um delineamento preciso entre estas duas classes. Assim, por exemplo, a tempestividade possui uma forte correlação com a consistência. Procuramos aqui classificar os atributos de acordo com o grau de influência dos processos ou dos dados sobre a medida.

Fidelidade

Definição

é a medida da tolerância do sistema quanto à precisão dos dados mantidos e produzidos pelos diversos processos.

Discussão

em muitos casos nao e possível trabalhar-se com valores ou dados possuindo precisão absoluta. É necessário, nestes casos, definir-se uma faixa de tolerância, que, se mantida, leve a resultados satisfatórios.

A fidelidade não deve ser confundida com o atributo correção uma vez que sistemas corretos, submetidos a dados também corretos (fiéis), podem, em determinadas circunstâncias, produzir resultados incorretos ou pelo menos excessivamente imprecisos (infiéis). Isto é comum, por exemplo, em rotinas de cálculo numérico.

Pela mesma razão, fidelidade não deve ser confundida com robustez, apesar da semelhança dos métodos utilizados para assegurar tanto uma quanto a outra. Ou melhor, a robustez mede a capacidade de detectar falhas de processamento, enquanto que a fidelidade mede a capacidade do sistema em detectar e/ou evitar imprecisões intoleráveis nos dados e nos resultados.

Em muitos casos é impossível assegurar a precisão dos dados mantidos em arquivos. Por exemplo, a grafia do nome de uma pessoa pode conter pequenos desvios da realidade e, mesmo assim, servir de identificação desta pessoa. Ou seja, é perfeitamente possível ter-se um sistema de elevado valor, mesmo quando alguns dados possam ser incorretamente registrados e/ou mantidos. Isto tudo, desde que estes desvios estejam dentro de limites toleráveis e desde que existam processos de correção, permitindo retornar-se a estes limites.

Integridade

Definição

é a medida da habilidade do sistema em manter os dados tal como intencionalmente registrados ou atualizados.

Discussão

o sistema deve tomar medidas de proteção e segurança que garantam a permanência inalterada dos dados, exceto quando esta alteração, ou incorporação, de dados seja efetuada por processo de atualização autorizado.

A integridade dos dados não implica na fidelidade, consistência, validade etc. destes dados. Ou seja, mesmo que o dado legalmente introduzido no sistema seja inadequado segundo alguma medida, o sistema deverá protegê-lo contra adulterações acidentais ou premeditadas (fraudes).

Consistência

Definição

é a medida do grau de equivalência entre os dados contidos no sistema e suas imagens no mundo real.

Discussão

dados manipulados por sistemas de computação, são, frequentemente, representações de entidades do mundo real. É necessário assegurar-se uma correspondência entre estes dados e as entidades correlacionadas. Por exemplo, em um sistema de controle de estoque, a quantidade de uma determinada peça registrada em arquivo deve corresponder à quantidade real com que esta peça ocorre nas prateleiras.

Cabe evidenciar a diferença entre consistência e fidelidade. A consistência mede a correção dos dados com relação ao mundo real dentro de limitações de tempestividade. Por outro lado, a fidelidade mede a precisão com que estes dados estão registrados.

Devido à natureza dos sistemas automatizados, sempre existe um período de tempo a partir do evento que causou a alteração do estado do mundo real até ao instante em que esta mudança de estado está devidamente registrada nos arquivos do sistema. A extensão deste período de tempo pode variar desde bem pequeno ("on-line") até bastante grande ("batch"). A variação da extensão deste período pode dever-se aos procedimentos de registro e relacão dos eventos do mundo real, e não somente à metodologia de processamento utilizada. Durante este período de tempo, alguns dados contidos em arquivos estão necessàriamente inconsistentes. A utilização destes dados pode, então, causar a produção de resultados inadequados, o que pode levar à rápida deterioração da qualidade e, consequentemente, à deterioração do sistema em si. Os períodos de inconsistência devem ser levados em conta durante a especificação do sistema por intermédio do atributo "tempestividade".

Validade

Definição

é a medida da obediência às relações de redundância próprias do dado e das existentes entre dados.

Discussão

dados estão sujeitos a restrições. Por exemplo, determinados dados somente podem tomar valores dentro de um conjunto limitado, outros necessitam satisfazer relações de controle (por exemplo, dígitos de controle, "sum checks" etc.), já outros requerem a presença nos arquivos de grupos de dados interrelacionados (por exemplo, registros possuindo chaves iguais), finalmente pode existir uma relação complexa entre todos os dados apresentados (por exemplo, dados ou programas devendo obedecer a uma sintaxe específica).

Deseja-se sempre obter validade absoluta nos dados de entrada. Isto poderá ser efetuado por aproximações sucessivas, onde, a cada passo do exame da validade, os dados são verificados utilizando critérios cujo custo de aplicação seja cada vez mais elevado.

A validade dos dados não implica em fidelidade nem tampouco em consistência. Ela é somente um indicativo de que não existem determinados erros, encontráveis por inspeção automatizada.

Interpretabilidade

Definição

é a medida (inversa) do esforço necessário para o receptor de um conjunto de dados converter este conjunto em informação.

Discussão

para sabermos o que o dado representa, precisamos conhecer diversas de suas propriedades. Por exemplo, necessitamos conhecer conceitos relativos a estes dados (por exemplo, peso de mercadoria, item de estoque, identificação), a dimensão do dado (por exemplo, kg, número de itens, nome de pessoa), o relacionamento do dado com outros dados etc. Sem o conhecimento destas propriedades ficamos incapacitados em adquirir informação através da consulta a estes dados.

O esforço de conversão do dado em informação é sempre efetuado pelo receptor do dado. Este esforço é maior à medida em que mais dados são utilizados para gerar a informação [Burch74].

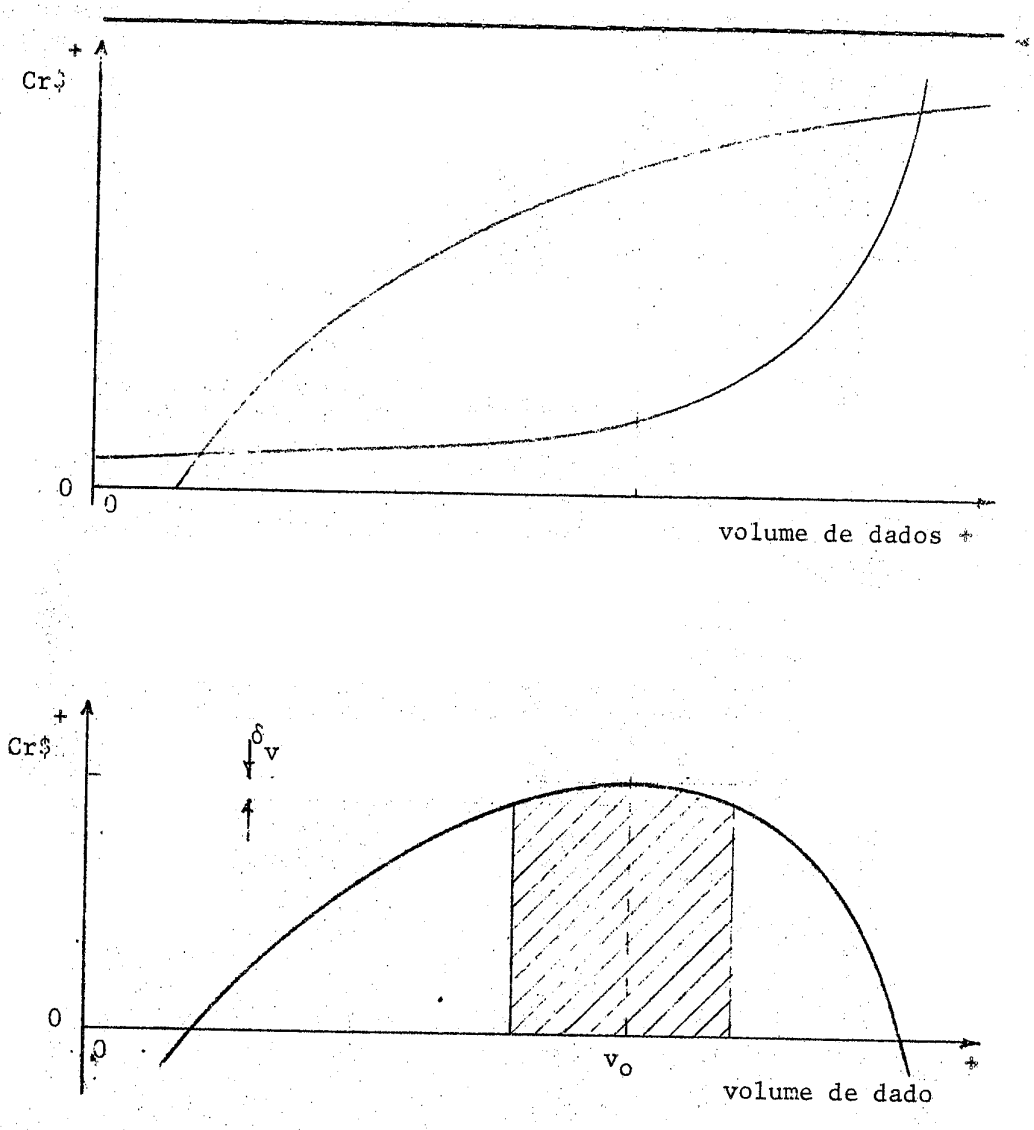
O esforço é limitado pela capacidade humana em processar dados. Este esforço pode ser medido em termos de custo, sendo que o custo total é o custo do esforço de interpretação acrescido do custo de produção dos dados.

Por outro lado, ao aumentar-se o volume de dados apresentados para a interpretação a precisão da informação é, em geral, incrementada. A precisão da informação pode ser medida em termos de benefícios auferidos, sendo estes benefícios decorrência de melhores decisões.

Também aqui existe um volume de dados que maximiza a diferença benefícios-custos. Isto leva à necessidade de estudos semelhantes ao da análise de custos e benefícios (rentabilidade), aplicado ao esforço de conversão versus volume de dados. A análise econômica relativa ao volume de dados, tanto de entrada, quanto os resultantes do processamento, deve ser efetuada para cada componente do sistema automatizado.

Na figura 2.2 apresentamos de forma esquemática a evolução dos custos e benefícios em função do volume de dados apresentados em diálogos homem/máquina específicos. Os benefícios decorrem do aproveitamento da informação extraída do relatório ou mensagem [Burch74]. Os custos decorrem do esforço necessário para produzir o relatório ou mensagem e do esforço necessário para extrair esta informação do relatório ou mensagem.

Cabe aqui uma pequena digressão em torno dos conceitos dado e informação. Dado é qualquer coisa registrada em algum meio físico, enquanto que informação é conhecimento, onde conhecimento é usualmente entendido como sendo o conjunto de todas as informações [Langefors73]. Para podermos converter dados em informação,



v_o volume de dados ótimos

$v_1 v_2$ tolerância de volume de dados

δ_v tolerância de valor .

Figura 2.2 Evolução dos custos e benefícios em função do volume de dados.

devemos, antes de mais nada, entender o dado, ou seja, necessitamos saber os conceitos associados a este dado (linguagem, vocabulário, sentido das palavras etc.).

O esforço de conversão de dados em informação deve-se à reflexão, associação, agregação etc., necessá-

rias para obtermos alguma informação útil tendo por base estes dados. Usando um exemplo simplista, ao gerente financeiro não interessa saber os valores pagos a cada funcionário da empresa. Interessa, porém, saber o total do que foi pago aos funcionários. Qualquer relatório que apresente valores de pagamentos individuais aumenta o esforço do gerente financeiro, ou por ter que encontrar o dado desejado no meio de outros, ou por ter que produzí-lo através de uma soma dos valores individuais.

Utilizamos aqui uma definição para informação estritamente vinculada ao ser humano. Muitas vezes, porém, a definição de informação é generalizada, permitindo que máquinas adquiram informação. Esta informação permitirá à máquina a seleção de alternativas de decisão previamente estabelecidas, ou programadas, como acontece, por exemplo, em alguns sistemas de informação sofisticados e em sistemas de comando e controle de malha fechada. Apesar desta possível generalização, preferimos manter a nossa definição de informação, o que implica em que tudo o que é manipulado e/ou produzido por computadores seja somente dado, mesmo quando o sistema de processamento de dados esteja capacitado para a escolha de decisões pré-definidas. A razão para isto é a diferença essencial da decisão tal como efetuada pelo homem e pela máquina, uma vez que a escolha da decisão pela máquina é estritamente pré-estabelecida, enquanto que a decisão tal como efetuada pelo homem leva em conta fatores de julgamento não quantificáveis.

Confidencialidade

Definição

é a medida da habilidade do sistema em proteger dados confidenciais contra acessos indevidos.

Discussão

muitos dos dados mantidos e/ou produzidos por sistemas automatizados devem ser protegidos contra o acesso por pessoas ou processos não autorizados. Esta proteção pode ser devida a condições contratuais (legislação), ou a riscos de perdas ou danos se os dados forem revelados a terceiros [Silva77, Parente78].

A confidencialidade pode estar sujeita a diversas gradações. Assim poderão coexistir em um mesmo sistema, dados cujo acesso é livre, outros de acesso pouco restrito e, ainda, outros de acesso fortemente restrito. O custo do estabelecimento e controle de restrições de acesso é tanto maior, quanto maior for o grau de restrição. Deve ser efetuada, então, uma análise econômica, com o intuito de determinar os níveis "ótimos" de restrição ao acesso [Silva77].

A confidencialidade complementa a integridade uma vez que esta mede a capacidade de proteção contra a adulteração ilegal dos dados, enquanto que a confidencialidade mede a capacidade de proteção contra o acesso ilegal aos dados.

Interrabilidade

Definição

é a medida da habilidade do sistema em compor-se com outros sistemas.

Discussão

como já foi mencionado, sistemas automatizados não existem isolados, compondo-se sempre com outros, com o intuito de formar sistemas mais abrangentes. Esta composição é efetuada através da troca de dados entre os sistemas componentes. A capacidade de trocar dados com outros sistemas, bem como o custo desta troca, é medida pela integrabilidade.

A troca de dados pode ocorrer entre diversos sistemas de diferentes instituições. Por exemplo, um sistema de folha de pagamento poderia produzir arquivos compatíveis com o sistema de controle de imposto sobre a renda retido na fonte, com o sistema de controle de ordens de pagamento do banco, etc. A troca de dados ocorrendo por meios mecânicos é, normalmente, menos sujeita a falhas operacionais, reduzindo, em geral, o custo operacional de forma significativa.

Cabe observar aqui, que mesmo quando do desenvolvimento de um sistema que a princípio não será integrado, existe a necessidade de comunicação entre este e outros sistemas. Devem existir, portanto, normas de utilização e representação de dados aplicadas a toda a instituição. Estas normas estarão contidas no diretório de dados a ser descrito mais tarde Furtado78 . De maneira semelhante, devem existir normas descrevendo dados e arquivos que poderão ser intercambiados entre sistemas de diferentes instituições.

2.3 Atributos da Classe "Manutenibilidade".

Esta classe de atributos mede a capacidade de modificar-se o sistema, seja para corrigí-lo, seja para adaptá-lo a novas condições ambientais, ou ainda para estendê-lo com o intuito de atender a mais necessidades do que originalmente especificadas.

Sistemas estão sujeitos a modificações constantes. Estas modificações têm por objetivo restabelecer, manter ou expandir a rentabilidade do sistema, contribuindo desta forma para o aumento da durabilidade deste. Estas frequentes alterações devem requerer pouco esforço (custo) para que possam ser efetuadas.

Compõem o esforço de alteração a dificuldade de localizar o que deve ser alterado, a dificuldade de efetuar a alteração, a dificuldade de certificar o sistema alterado e a dificuldade em manter o nível de utilidade e de qualidade ao efetuar-se as alterações. Estas dificuldades são evidenciadas em termos de tempo e recursos humanos, materiais e financeiros necessários para concluir-se cada uma dessas tarefas.

Como sistemas automatizados são caros, deve-se tomar cuidado para que tenham uma vida útil longa. A vida útil destes sistemas é fortemente afetada pelos atributos de manutenibilidade, e pela permanência destes atributos nos níveis especificados durante toda a vida útil do sistema.

Contribuem para a redução dos custos de manutenção, a modularidade dos processos, a qualidade da documentação, a mensurabilidade dos atributos, a certificabilidade dos procedimentos e programas, a existência de arquivos contendo dados para testes, etc.

Durante o desenvolvimento de um sistema automatizado, é frequente alterar-se componentes deste sistema para que este possa vir a atingir os seus objetivos. Portanto, já durante o desenvolvimento, os benefícios de elevada manutenibilidade fazem-se sentir, uma vez que as ações corretivas e adaptativas executadas durante o desenvolvimento são semelhantes àquelas executadas durante a operação do sistema.

A principal diferença entre a "manutenção" do sistema durante o desenvolvimento, e a manutenção na fase de operação, é que a primeira é efetuada pelas pessoas que participam do desenvolvimento dos procedimentos e programas. Estas pessoas possuem um elevado conhecimento memorizado do sistema. Por outro lado, as pessoas encarregadas de efetuar a manutenção propriamente dita, deverão ser (pelo menos deveriam poder ser) pessoas que não participaram do desenvolvimento. Estas pessoas somente têm à sua disposição a documentação do sistema, sendo seu trabalho fortemente afetado por propriedades tais como estruturação, legibilidade, fidelidade da documentação etc.

A manutenção de sistemas automatizados pode ocorrer devido a erros detectados no sistema, mudanças do meio ambiente em que reside o sistema, mudanças das necessidades do usuário etc. A seleção da ação a ser executada depende, no entanto, de uma análise de custo e benefício [Moura79]. Assim, após constatada uma necessidade de correção, alteração ou expansão, deve ser decidida qual a melhor solução a ser adotada, tendo por base esta análise econômica. Estas soluções podem ser, por exemplo, fazer nada, adaptar o sistema, alterar partes do sistema, substituir o sistema por outro novo etc.

Para permitir uma análise econômica adequada, deve-se, a princípio, produzir uma proposta de desenvovimento tal como é feito para um sistema novo [Staa79]. No entanto, o esforço para produzir esta proposta poderá ser considerável, quando comparado com o custo de implementação da alteração. Assim, antes de iniciar a confecção da proposta, deve ser estimado o custo da alteração, sendo o nível de detalhe da proposta adaptado à esta estimativa.

Localizabilidade

Definição

é a medida (inversa) do esforço necessário para localizar todos os pontos afetados por uma alteração no sistema.

Discussão

uma vez especificada uma alteração do sistema, seja para corrigir erros, seja para atender a mudanças no meio ambiente, ou ainda para atender a necessidades de expansão do sistema, torna-se necessário determinar quais as porções, ou pontos do sistema, que serão afetadas por esta alteração.

A localização dos pontos afetados pela alteração deve ser completa, pois, de outra forma, a alteração tornaria o sistema inconsistente, o que pode levar a desastres ou a alterações em cascata. Os pontos a localizar podem encontrar-se em diversas classes de componentes, por exemplo, podem ser localizados em programas, em procedimentos, na documentação, nos dados para teste, nos dados de produção, etc. Sistemas para a produção de índices hierárquicos, referências cruzadas, listas de relacionamentos entre os processos e a documentação, são exemplos de ferramentas que reduzem o esforço de localização.

Alterabilidade

Definição

é a medida (inversa) do esforço necessário para incorporar alterações ao sistema.

Discussão

ao corrigir-se erros, ao fazer-se adaptações, ou ao expandir-se um sistema automatizado, torna-se necessário substituir e/ou alterar componentes deste sistema e, após, aprovar estas modificações. As alterações podem variar desde bem simples, por exemplo a modificação de uma linha de código, até bastante complexas, por exemplo substituição de módulos, de documentos etc. Cabe observar que a complexidade da alteração independe do tipo de causa que a gerou, ou seja, independe de se é devida a uma correção, adaptação, aperfeiçoamento ou expansão.

O esforço necessário para incorporar alterações, advêm da necessidade de modificar textos (documentos, procedimentos, programas); manter bibliotecas de programas, dados para teste, documentação, etc.; aprovar modificações; disseminar as modificações aos usuários; controlar a efetiva incorporação das modificações; controlar versões do sistema e sequências de modificações; etc. O esforço necessário para incorporar alterações ao sistema, é tanto maior quanto menos sistematizado for o processo de alteração. Sistemas automatizados de auxílio ao desenvolvimento e à produção de documentos são exemplos de ferramentas que reduzem este esforço.

Cabe enfatizar que o esforço de incorporação de alterações advêm, também, da aprovação do sistema alterado. Ou seja, envolve a verificação dos níveis dos diversos atributos de qualidade e utilidade, tanto nas porções que foram efetivamente alteradas, quanto naquelas que não o foram. Se esta aprovação não for conduzida de modo sistemático, cedo o sistema tornar-se-á decrépito.

A alterabilidade não mede o esforço de desenvolvimento de componentes necessários para a correção, adaptação, aperfeiçoamento ou extensão. Este esforço deve ser estimado e medido caso a caso. Se o esforço for pequeno, a estimativa pode ser conduzida de modo ad hoc, porém se o esforço estimado for grande, deve ser empregada a mesma sistemática utilizada para propor-se novos desenvolvimentos, de maneira a permitir a avaliação rigorosa do esforço necessário [Staa79].

Detectabilidade

Definição

é a medida (inversa) do esforço necessário para a determinação de falhas existentes no sistema.

Discussão

ao detectar-se um erro, na realidade é detectado somente um sintoma. Por meio de análises conduzidas no sistema é localizada a causa do erro, ou seja, é localiza-

da a falha do sistema responsável por este sintoma. Após ter-se determinado a falha, torna-se necessário propor medidas corretivas, desenvolvê-las e, finalmente certificar o sistema corrigido. Esta certificação deve indicar que a falha foi eliminada de maneira completa e que nenhuma nova falha foi introduzida (ver "localizabilidade" e "alterabilidade" acima).

Como já foi mencionado, o esforço necessário para desenvolver medidas corretivas é muito variável, indo desde pequenos ajustes, até, em casos desastrosos, a necessidade do desenvolvimento de um sistema inteiramente novo. Portanto, antes de corrigir, deve ser estimado o esforço necessário para esta correção. Caso este esforço seja significativo, é conveniente conduzir um estudo de alternativas, evitando, desta forma, esforços de correção não rentáveis. Cabe observar que, após à análise de custo e benefício, pode ser concluído ser a melhor opção não corrigir a falha, introduzindo uma limitação no sistema que impeça a falha de ser exercitada.

A detectabilidade pode medir, também, o esforço necessário para especificar alterações devidas a mudanças ambientais, ou expansões do sistema. Isto pode ser feito através da "simulação" de um sintoma de erro. Ou seja, são modificadas as especificações e, a partir deste instante, o sistema não estará mais correto, procedendo-se então à correção dos "erros" assim introduzidos.

Os atributos localizabilidade, alterabilidade e detectabilidade, são fortemente interrelacionados, não possuindo delimitações evidentes. São, contudo, medidas independentes, uma vez que medem propriedades diferentes. Assim, dada a alteração, a localizabilidade mede a dificuldade de encontrar-se tudo o que pode ser afetado por esta alteração, enquanto que a alterabilidade mede o esforço de incorporação e/ou substituição e de teste. Por sua vez, a detectabilidade mede o esforço necessário para transformar um sintoma de erro em uma especificação de alteração.

As tres medidas são fortemente afetadas pela qualidade da documentação e pela modularidade e estruturação do sistema e dos processos.

Recuperabilidade

Definicaão

é a medida (inversa) do esforço necessário para repor o sistema em um estado adequado após a ocorrência de uma falha ou condição adversa.

Discussão

além de propor uma alteração que corrija a falha detectada, é necessário eliminar-se os danos introduzidos no sistema por esta falha. Da mesma forma, quando ocorrer uma condição adversa, é necessário recompor o sistema automatizado (execução de "back up"), assegurando a sua atualidade, para que possa retornar à operação [Magalhães79].

A recuperabilidade difere da robustez, uma vez que esta é uma medida preventiva, enquanto que a recuperabilidade depende da presença e da execução de processos de recuperação já durante a fase de operação.

Flexibilidade

Definição

é a medida da habilidade do sistema em atender a uma gama de condições ambientais e de necessidades do usuário, sem necessitar de alterações significativas em componentes deste sistema.

Discussão

o meio ambiente no qual reside o sistema está em constante alteração. Contudo é indesejável ter-se que modificar componentes do sistema sempre que aumente o volume de processamento ou de arquivamento necessário para operar o sistema, ou quando ocorrer uma mudança no meio ambiente ou em necessidades do usuário. É desejável então poder adequar os componentes, por exemplo, através do ajuste de parâmetros, ou substituição de componentes previamente definidos.

A flexibilidade é particularmente importante para atender à demanda de relatórios especiais, ou de perguntas específicas ("queries"). Tais relatórios e perguntas podem ser atendidos por intermédio de "programas" escritos em linguagem de fácil aprendizado. Estes "programas" são na realidade especificações simples dos relatórios ou das respostas desejadas.

Computadores são máquinas flexíveis, porém programas muitas vezes não o são. A falta de flexibilidade dos programas pode levar a elevados custos globais com relação à computação. Este custo é função direta do custo de alteração dos componentes do sistema automatizado, onde estas alterações (por exemplo alterações de programas, retreinamento de pessoal) são necessárias para atender às novas necessidades. O estabelecimento de níveis de flexibilidade leva a custos de alteração menores, apesar de causar custos de desenvolvimento maiores. É necessário, então estimar-se níveis ótimos para a flexibilidade.

De maneira semelhante, um sistema poderia ser desenvolvido para atender a diversos usuários com necessidades ligeiramente diferentes. Se o sistema for flexível, o esforço necessário para ajustar o sistema base aos diversos usuários é minimizado.

Portatilidade

Definição

é a medida (inversa) do esforço necessário para transferir o sistema para outro equipamento, configuração ou sistema de suporte.

Discussão

durante a vida de um sistema é frequente ocorrer mudanças de equipamento totais ou parciais e, também mudanças no software de apoio. É frequente, ainda, passar-se a dispor de equipamento ou sistema de suporte que não havia sido previsto ao iniciar-se o projeto de desenvolvimento do sistema automatizado. Estas mudanças, em geral, requerem alterações no sistema. Quanto mais portátil for o sistema, menores serão estas mudanças e menor será o esforço necessário para concluí-las.

A portatilidade pode ser encarada como sendo um sub-atributo de flexibilidade. Dada, porém, a natureza e a elevada influência da portatilidade na duração da vida útil de um sistema automatizado, julgamos conveniente a apresentação em separado desta medida.

Aproveitabilidade

Definição

é a medida do percentual do volume de componentes de um sistema que poderão ser aproveitados no desenvolvimento de um novo sistema.

Discussão

devido à evolução dos métodos empregados e das necessidades do usuário, sistemas automatizados serão eventualmente substituídos por outros possuindo características mais avançadas. Ao efetuar-se o desenvolvimento destes novos sistemas, deve ser possível aproveitar componentes de sistemas já existentes, reduzindo desta forma os custos de desenvolvimento e conversão para o sistema novo. O aproveitamento pode requerer adaptações e transporte para outro sistema de suporte tanto de programas quanto de conteúdos de arquivos.

Para que sistemas sejam aproveitáveis, é necessário que adiram a normas de desenvolvimento bem definidas. Em particular, é necessário que exista um dicionário de dados atualizado, completo e que abranja os di-

ferentes sistemas processando dados. São necessários, também, boa documentação e adequada modularização do sistema automatizado.

3 Condicionantes do Desenvolvimento.

O desenvolvimento de sistemas automatizados é regido por diversos fatores que limitam a escolha de alternativas e os níveis de qualidade e utilidade que podem ser atingidos. Muitos desses fatores não têm sido devidamente observados. Isto tem levado frequentemente a frustrações, insatisfações, perdas elevadas, etc. Por outro lado, quando sistemas são pequenos ou muito simples, a não observância desses fatores causa somente contratempos pequenos. Foi constatado que as dificuldades encontradas ao desenvolver-se um sistema automatizado aumenta pelo menos quadráticamente com o crescimento em tamanho e/ou complexidade deste sistema [Boehm73, Putnam77a].

O desejo de produzir sistemas automatizados de forma industrial, ao invés da sua criação artesanal, motivou estudos sobre o seu processo de desenvolvimento. Esses estudos conduziram a caracterizações, ainda embrionárias, de qualidade e utilidade de sistemas automatizados. Levaram, também, a reconhecer-se outros fatores que afetam a produção de sistemas automatizados e que limitam o grau de qualidade e utilidade atingível. Neste capítulo examinaremos alguns desses fatores.

3.1 Entidades Participantes.

Sistemas automatizados devem ser desenvolvidos tendo em vista o usuário. Isto é o óbvio, contudo a prática está repleta de exemplos de sistemas que não chegaram a ser utilizados ou, então, são hostilizados pelos usuários. Diversas são as razões para tal. Em particular existe frequentemente uma barreira de comunicação entre aqueles que desenvolvem o sistema e aqueles que irão utilizá-lo. Nos casos mais patológicos, o analista chega ao ponto de afirmar que "o usuário não sabe o que quer", passando, com base nesta afirmativa, a desenvolver o sistema que ele, analista, acha ser o desejado pelo usuário. A consequência quase infalível é, ao final de muito esforço, o sistema ser rejeitado pelo usuário por diferir demais daquilo por ele imaginado.

Para vencer esta barreira de comunicação, tanto os usuários, quanto os técnicos, precisam fazer algum esforço de aprendizado. O usuário precisa obter conhecimentos sobre o que o computador é capaz de fazer por ele, quais as limitações dos sistemas de computação, quais as restrições econômicas,

quais as necessidades organizacionais, etc. Este conhecimento não implica em saber programar, implica porém em saber expressar-se de uma forma clara e objetiva.

O técnico, por sua vez, necessita compreender o problema do usuário como um todo, e não somente aquelas facetas que eventualmente serão automatizadas. Isto requer a capacidade de aprender a linguagem (jargão) em que o usuário costuma expressar-se. Afinal, se o computador deve servir ao usuário, não é razoável esperar-se do usuário uma adaptação à máquina, muito pelo contrário, é esperado a máquina, dita ser flexível, adaptar-se ao usuário.

Apesar de sistemas automatizados deverem atender ao usuário, eles não devem ser desenvolvidos com o intuito de emular o sistema existente. Isto, porque, frequentemente, os sistemas existentes possuem defeitos nos componentes não automatizados e que, se copiados para o software, fatalmente tornã-lo-iam inflexível.

Disto tudo pode-se concluir que o usuário deve ter uma participação ativa no desenvolvimento do sistema. Esta participação deve fazer-se notar durante todas as fases do ciclo de vida do sistema automatizado. Deve estar claro que não se espera do usuário uma participação em atividades de programação e/ou de projeto de elementos de programas. Porém, espera-se dele o exame destes programas quanto ao atendimento às suas necessidades Benjamin71; Gildersleeve74; Gildersleeve78; Asteggiano78; [Weinberg75].

Diversas pessoas, ou grupos de pessoas, participam do desenvolvimento e da operação de sistemas automatizados. Estes grupos chamaremos aqui de entidades. Deve ser observado que não existe uma delimitação clara entre as diversas entidades. Em alguns casos é até possível que um mesmo grupo de pessoas participe de mais de uma entidade.

São seguintes as entidades:

i- entidade "usuário";

ii- entidade "cliente";

iii- entidade "coordenador";

iv- entidade "desenvolvedor".

No texto a seguir caracterizaremos sucintamente cada uma destas entidades.

A entidade usuário, além de receber e utilizar os resultados produzidos pelo sistema automatizado, também é responsável por sua operação, auditoria e manutenção. Esta amplitude de atividades e responsabilidades pode ser encarada como

um abuso do têrmo "usuário". Apesar de reconhecermos este abuso, continuaremos a utilizar este têrmo por falta de melhor.

A entidade usuário é composta por diversos grupos. Os sistemas automatizados devem atender às necessidades de todos estes grupos. Somente assim pode-se asegurar a satisfação plena com o uso do sistema. Por exemplo, um sistema difícil de operar, ou cujos diálogos homem/máquina foram mal projetados, fatalmente estará sujeito a uma incidência maior de erros de operação, ou de preparação de dados, o que, além de aumentar os custos operacionais (recuperação após a ocorrência de erros de uso), também pode levar à redução da eficácia, consistência, etc., o que, por sua vez, leva a um sistema de valor menor.

Sistemas automatizados devem ser examináveis quanto à sua qualidade, utilidade, benefícios, garantias de proteção e segurança, prevenção de acidentes e fraudes etc. Neste caso estará envolvido um grupo de pessoas que convencionamos chamar de usuários auditores.

Para que sistemas automatizados possam produzir benefícios, é necessário operar-se o equipamento computacional ou periférico que lhe serve de suporte, bem como é necessário manter-se operacionais os programas, procedimentos etc. Neste caso estão em jogo os grupos que convencionamos chamar, respectivamente, usuário produtor e manutenedor.

Embora sistemas automatizados devam ser mantidos pelos usuários ("usuário manutenedor"), a prática está repleta de exemplos de sistemas que exigem a contínua presença daqueles que o desenvolveram para efetuar as operações de manutenção. Isto tem levado à existência de sistemas de software que são da "propriedade" de determinado programador ou grupo de programadores. Estes sistemas muitas vezes são descontinuados prematuramente quando estes programadores abandonam a empresa ou mudam de atividade dentro da empresa. Em geral pode-se atribuir a "propriedade" de sistemas à falta de documentação adequada, à falta de estruturação desses sistemas, em suma à construção artesanal destes sistemas, ao invés de sua produção profissional [Brown74, Weinberg71, Yohe74, Yourdon75]. A tendência atual é a de exigir que o desenvolvimento seja feito de tal forma que seja possibilitada a manutenção do sistema por grupos diferentes dos que desenvolveram estes sistemas. Desta forma torna-se mais fácil o uso de software desenvolvido por empresas especializadas, seja este software feito sob medida ou genérico. A possibilidade do uso de empresas especializadas evidentemente tende a reduzir os custo de desenvolvimento e, através da diluição dos custos sobre diversos usuários, o custo de amortização do sistema por usuário. A possibilidade de manter-se operacional o sistema, aumenta, também, a durabilidade do sistema, o que, por sua vez, reduz

a necessidade de novos desenvolvimentos, contribuindo desta forma para a redução do custo global dos sistemas automatizados.

Na tabela 3.1 caracterizamos os diversos subgrupos da entidade "usuário".

gênero / sub-grupo	Operativo	Decisório
produtor	Manipula equipamentos de computação, de transcrição de dados, etc.	Determina sequências de execução, prioridades de execução, início de ações contingentes previstas
manutenedor	Corrige e adapta o sistema (manutenção propriamente dita)	Detecta, registra, reporta e toma medidas corretivas e preventivas quanto a erros e falhas
Direto	adquire, coleta e prepara os dados de entrada, preenche formulários	recebe e utiliza os resultados, toma decisões baseadas nos resultados, verifica a correção dos resultados, reporta erros
Auditor	Inspeciona e controla a qualidade total do sistema, a observância de normas técnicas e de proteção e segurança, detecta, registra e reporta erros de uso, fraudes, etc.	Determina cursos de ação corretiva e preventiva, determina procedimentos de recuperação de acidentes, erros de uso, falhas, fraudes, etc.
Indireto	Conjunto de pessoas que efetivamente utilizam o sistema, porém sempre com a interveniência de um usuário direto operativo e/ou decisório	
Existencial	Conjunto de pessoas que não utilizam o sistema, possivelmente nem sabem da existência deste, porém podem ser afetadas pela presença dele	

Tabela 3.1 Caracterização dos grupos componentes da entidade "usuário".

A entidade "cliente" é responsável pelo fornecimento, ou criação de condições de fornecimento, dos recursos financeiros para permitir o desenvolvimento do sistema automatizado.

A entidade "cliente" tem o poder de decisão quanto à vida do sistema. Em geral esta entidade é constituída pela alta gerência ou, em outros casos, por agências financiadoras ou de fomento.

A participação da entidade "cliente" no desenvolvimento tem sido pequena em geral, limitando-se à aprovação e eventual inspeção do projeto de desenvolvimento. Em alguns casos, mormente quando do desenvolvimento de grandes sistemas de informação integrados, é proposto que a entidade "cliente" participe ativamente das fases iniciais desse desenvolvimento. Justifica-se isto, uma vez que a alta gerência é usuário direto decisório neste caso [IBM75]. Não somente isto, sistemas desta natureza muitas vezes requerem substanciais mudanças organizacionais na empresa. São, além disso, caros e demorados, possuindo uma grande parcela de risco de não atingir adequadamente os objetivos.

A entidade "cliente" deve participar no desenvolvimento através de tomadas de decisão amparadas por auditagens feitas durante o desenvolvimento do sistema. Durante o desenvolvimento é importante examinar-se com frequência a rentabilidade esperada do sistema. Para evitar uma deturpação do exame de rentabilidade, é necessário dispor-se de controles que limitem estas possíveis deturpações. Estes controles devem ser efetuados através da auditoria de sistemas, a ser feita, portanto, já durante o próprio desenvolvimento [Parente78].

O exame do sistema por parte da entidade "cliente" deve ser razoavelmente frequente. Para que este exame seja possível, é necessário dispor-se de documentos corretos e que sejam adequados ao escrutínio por parte dos componentes da entidade "cliente". Cabe lembrar aqui que este exame deve ser efetuado durante todo o ciclo de vida do sistema e não somente durante a fase de definição. A necessidade de documentos direcionados para a entidade "cliente", impõe requisitos adicionais ao desenvolvimento do sistema.

A entidade "desenvolvedor" é responsável pelo desenvolvimento e implantação do sistema. É constituída por pessoal técnico e de suporte. Em muitos casos corresponde a empresas especializadas em desenvolvimento de sistemas automatizados.

A entidade "coordenador" é responsável pela condução (gerência) do projeto de desenvolvimento. As principais atribuições desta entidade são:

i- representar a entidade "cliente";

ii- controlar o uso de recursos;

iii- criar normas e padrões;

iv- observar a obediência a normas e padrões;

v- produzir junto com as entidades "usuário" e "cliente" a proposta de desenvolvimento do sistema [Staa79];

vi- selecionar e controlar a entidade "desenvolvedor";

vii- estabelecer a interface entre o desenvolvedor e o usuário;

viii- produzir junto com as demais entidades o plano de desenvolvimento [Rocha78, Metzger73];

ix- coordenar e controlar o desenvolvimento tendo por base o plano;

x- presidir a comissão de controle de alterações ativa durante o desenvolvimento do sistema;

xi- efetuar a aprovação de todos os produtos adquiridos;

xii- conduzir exames de aprovação de todos os produtos entregues, mesmo os intermediários;

xiii- garantir a defesa dos interesses de todos os que constituem a entidade "usuário";

xiv- determinar as alterações organizacionais que se fizerem necessárias para permitir a operação do sistema.

A entidade "coordenador" é a entidade central, da qual depende, em última análise, o sucesso ou o fracasso do desenvolvimento do sistema automatizado. Via de regra, a entidade "coordenador" tem sido ocupada por uma única pessoa: o gerente de desenvolvimento. Quando se trata do desenvolvimento de sistemas maiores, esta pessoa pode ser insuficiente para tomar a si todos os encargos e as responsabilidades próprias desta entidade. Nestes casos ocorre uma estratificação da entidade "coordenador", subdividindo-a em grupos de gerência de desenvolvimento, grupos de desenvolvimento de ferramentas e suporte, grupos de controle de qualidade, grupos de produção de normas e padrões e grupos de auxílio à produção de documentos.

No caso de sistemas maiores, ou de sistemas desenvolvidos para atenderem a diversos grupos de usuários, é recomendável dispor-se de um arquiteto de sistema [Brooks75]. Este seria responsável pela organização geral do sistema automatizado, pela adequação dos requisitos de qualidade e utilidade deste sistema em particular, pela generalização da aplicabi-

lidade do sistema, pelo projeto das interfaces homem/máquina ("human engineering") etc. O arquiteto de sistema deve fazer parte da entidade coordenador.

3.2 Custos e Benefícios

Sistemas automatizados são essencialmente investimentos de capital, por vezes consideràvelmente grande. Este investimento não advêm somente da aquisição de equipamento, advêm principalmente da produção e aquisição de software. A tendência atual é a participação percentual cada vez mais ampla do componente software nos sistemas de computação. Na figura 3.2 mostramos graficamente a evolução no tempo da relação percentual no custo total dos componentes hardware e software conforme estimado por Boehm [Boehm76].

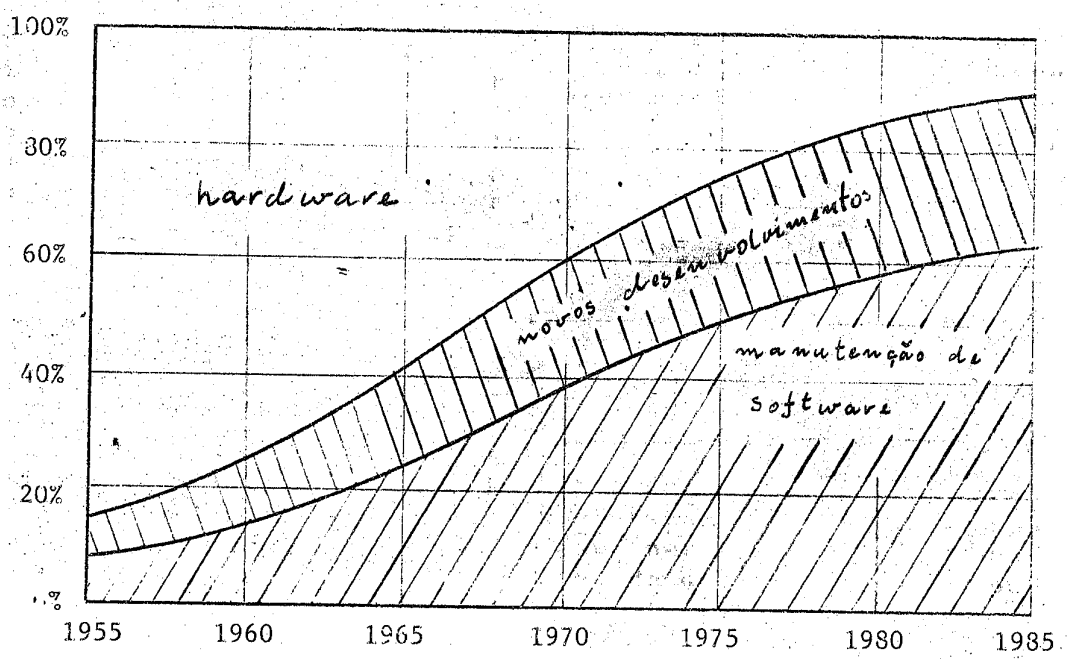


Figura 3.2 Evolução da participação percentual dos componentes hardware e software de um sistema de computação.

A evolução mostrada deixa claro que, se desejarmos reduzir os custos de sistemas de computação, devemos preocuparnos em reduzir a parcela custo de software.

Entendemos aqui benefícios como sendo fluxos de caixa causados pela operação do sistema. Os benefícios de um sistema automatizado podem decorrer da redução de custos, capital e operacional, e/ou do aumento da receita.

Usualmente os diversos atributos de utilidade determinam também benefícios. Ou seja, normalmente é possível estimar-se o fluxo de caixa gerado por determinado atributo de utilidade. Contudo, os atributos de utilidade devem ser especificados em termos de unidades diferentes de moeda. Por exemplo, a capacidade de atender a uma demanda de serviço deve ser determinada em termos de volume por unidade de tempo e não em termos de cruzeiros auferidos ou perdidos se esta demanda for ou não atendida.

Admitimos também a possibilidade da existência de atributos de utilidade, que, embora importantes, resultem em fluxos de caixa negligenciáveis. Tomando por base esta premissa, concluímos que um sistema automatizado é caracterizado por atributos de qualidade, de utilidade, custos e benefícios. Estes últimos determinam a evolução esperada para a rentabilidade do sistema automatizado.

Conclui-se então, que para podermos analisar e escolher alternativas de desenvolvimento, devemos possuir uma descrição dos diversos atributos de qualidade e utilidade, a importância relativa (peso) destes atributos e a relação dos benefícios esperados bem como dos valores estimados para estes. Cabe ressaltar que os atributos de utilidade, de qualidade e os benefícios formam conjuntos que se interceptam, não sendo comparáveis no caso geral, conforme ilustrado gráficamente na figura 3.3.

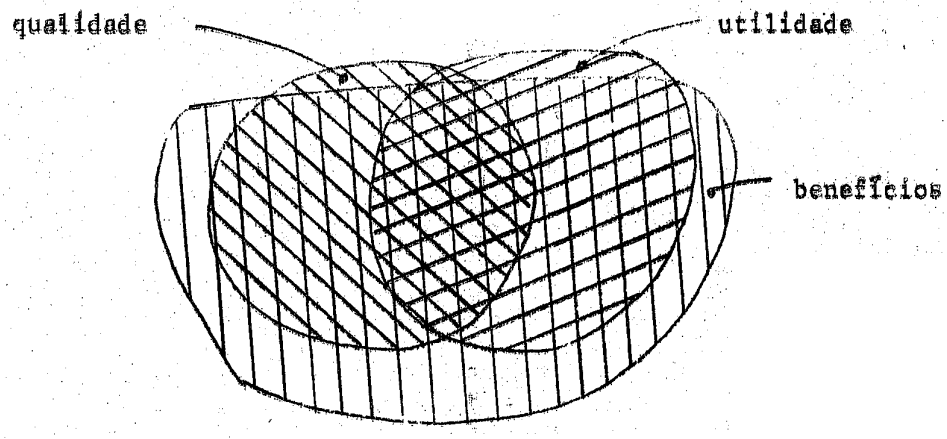


Figura 3.3 Representação gráfica das relações entre os atributos de utilidade, de qualidade e os benefícios.

O fato de sistemas automatizados serem investimentos de capital, faz com que devamos encarar com rigor a sua aquisição ou desenvolvimento, aplicando critérios de rentabilidade semelhantes aos utilizados em outras formas de investimento. Ou seja, além das especificações dos requisitos de qualidade e de utilidade, devemos ter uma projeção clara dos custos e uma estimativa adequada dos benefícios a serem produzidos pelo sistema.

Usualmente sistemas automatizados são sistemas de prestação de serviços à instituição que os utiliza, não constituindo, em si mesmos, produtos comercializáveis. Por esta razão, é frequente a incorporação dos custos decorrentes do desenvolvimento e uso de sistemas automatizados aos custos de administração ("overhead") da empresa, sem que seja questionada a rentabilidade destes sistemas.

Muitos dos benefícios gerados por sistemas automatizados não podem ser evidenciados através de sua contribuição direta aos lucros da empresa. Estes benefícios intangíveis são difíceis de quantificar. De fato alguns benefícios intangíveis são pouco amenos à quantificação, por exemplo, beleza, elegância etc. Contudo muitos benefícios ditos intangíveis podem ser quantificados através de estimativas. Esta quantificação pode ser estabelecida na medida em que se espera uma redução de custos (capital, operacional) e/ou um incremento de receita. Por exemplo, "melhoria da capacidade de gerência" pode representar uma redução de necessidade de capital da empresa, uma redução da necessidade de mão de obra e, possivelmente, um incremento da receita. Através de estimativas destes parâmetros, podemos chegar à quantificação dos benefícios intangíveis.

É claro que a precisão dos valores de quantificação de benefícios intangíveis é pequena. Melhorará, porém, com a aquisição de experiência em efetuar estas estimativas e através da comprovação do acerto destas estimativas em análises de pós-instalação [Gottlieb78]

Cabe enfatizar aqui que os benefícios tangíveis e intangíveis devem ser continuamente avaliados já desde o início do projeto de desenvolvimento. Somente desta forma é que poderemos assegurar a rentabilidade real de um sistema e tomar decisões objetivas quanto a continuar assim como está, aprimorar o sistema existente ou abandonar o sistema em favor de um novo.

Ao iniciar-se um novo desenvolvimento, ou uma extensão a um sistema já existente, deve ser examinada a rentabilidade deste novo sistema. Deve ser examinada, também, a rentabilidade do sistema antigo, ou do anterior à extensão. O sistema antigo pode possuir diversos defeitos que, se sanados, melhorariam suficientemente a sua rentabilidade. É importante então que, antes de partir para o desenvolvimento de um novo sistema automatizado, sejam examinadas as alternativas "dei-

xar assim como está", "reorganizar o que existe", e "aprimorar o que existe". Isto é novamente o óbvio, que na prática frequentemente não é observado. Além destas alternativas, devem ser examinadas, também, outras alternativas levando a graus variáveis de automação e/ou de alcance de objetivos secundários.

Os defeitos existentes no sistema antigo devem ser explicitados e o sistema novo, ou a extensão ao sistema antigo, deve removê-los. Devem ser evidenciados, também, os atributos de qualidade e utilidade do sistema antigo a serem mantidos em nível pelo menos igual no sistema novo. Uma vez tendo se feito isto, parte-se para a quantificação dos benefícios esperados. Desta forma pode-se obter uma estimativa da rentabilidade esperada com a operação do novo sistema.

Na figura 3.4 mostramos o aspecto genérico da evolução no tempo da rentabilidade de um sistema automatizado.

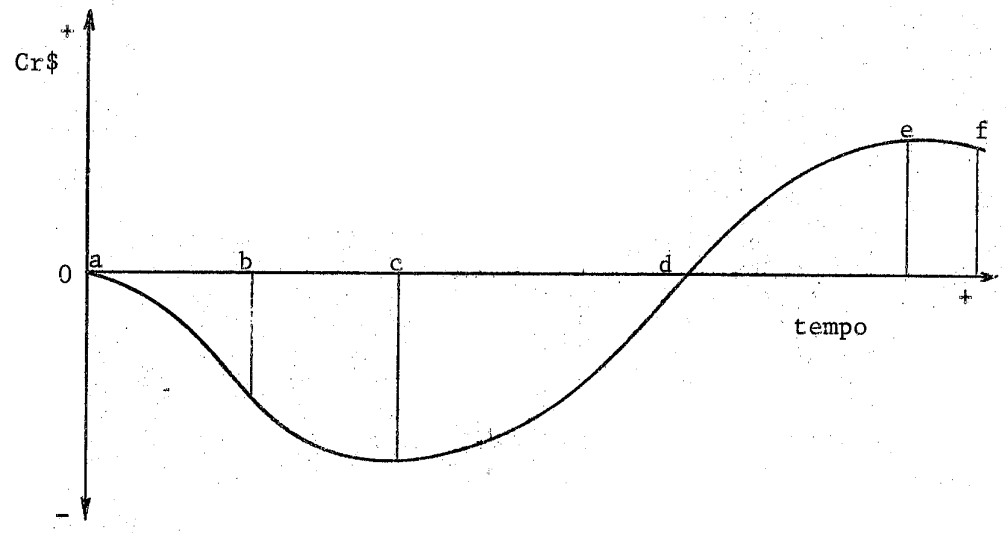


Figura 3.4 Aspecto genérico da evolução da rentabilidade.

São seguintes os significados dos pontos constantes da figura 3.4:

a- início do projeto de desenvolvimento. Isto ocorre durante a fase de concepção.

b- é o instante em que começam a ocorrer os primeiros benefícios. Sistemas maiores devem ser desenvolvidos e postos em operação parceladamente, componente por componente. Agindo desta forma, benefícios começarão a surgir bem antes do término total do desenvolvimento previsto.

c- é o instante em que é máximo o investimento. Este valor é um fator que, por si só, poderá impedir o desenvolvimento do sistema. O valor máximo pode ser reduzido através do desenvolvimento e implantação parcelados de subsistemas do sistema automatizado.

d- é o instante em que todo o investimento feito é retornado. Em muitos casos este ponto nem sequer é atingido. Um dos fatores que contribuem para que o ponto de retorno do capital não seja atingido, é a rápida obsolescência de sistemas automatizados, que, por sua vez, é acelerada pela baixa qualidade e baixa utilidade destes sistemas.

e- é o instante em que é máxima a rentabilidade. Com o decorrer do tempo e da continuada manutenção, é frequente ocorrer uma redução dos níveis dos diversos atributos de qualidade e utilidade do sistema. Isto é refletido em um aumento do custo operacional do sistema. Além disso, os requisistos de utilidade sofrem mudanças periódicas, o que pode provocar uma redução dos benefícios do sistema. Em casos extremados, o custo instantâneo poderá passar a ser maior do que o valor instantâneo dos benefícios do sistema, resultando em uma redução progressiva da rentabilidade.

f- é o instante em que o sistema é abandonado. O período de tempo desde o início até ao término, depende de vários fatores, tais como qualidade, obsolescência, manutenibilidade, número de vezes que os programas deverão ser operados etc. O período de vida de um sistema depende de atributos de qualidade tais como "portatilidade", uma vez que é altamente provável ocorrerem alterações ou mudanças de equipamento de suporte durante a vida de um sistema com período de vida longo.

Ao procurar-se soluções para substituir ou aprimorar sistemas, obviamente devemos optar por alternativas rentáveis. Nem sempre, porém, as alternativas mais sofisticadas serão as que apresentam a maior rentabilidade. Esta falha de julgamento leva muitos a optarem pelo uso de computadores, quando uma simples reorganização do sistema antigo seria suficiente para sanar os principais defeitos. Por outro lado, muitas vezes computadores não são utilizados devido ao desconhecimento da economia a ser obtida com o novo sistema. Este aspecto é sentido particularmente na reduzida aplicação de ferramentas de auxílio ao desenvolvimento de software. Finalmente, diversos sistemas automatizados são desenvolvidos com o intuito de solucionar problemas impraticáveis sem o auxílio de computadores, tais como, por exemplo, comando e controle em tempo real. O estudo da rentabilidade destes sistemas ocorre no nível do sistema envolvente, sendo, usualmente, limitado unicamente pelo custo máximo do desenvolvimento.

Ao iniciar-se um projeto de desenvolvimento ou de extensão, devem ser determinados critérios que evidenciem o sucesso do desenvolvimento ou da extensão. Estes critérios são tanto os critérios de utilidade e qualidade descritos no capítulo anterior, quanto critérios de rentabilidade. Portanto, devemos determinar:

i- os atributos de qualidade e utilidade relevantes, os níveis desejados para estes atributos, os limites inferiores dos níveis, bem como os pesos de cada um destes atributos;

ii- os atributos que afetam a rentabilidade, a estimativa dos custos e dos benefícios em função da variação destes atributos e de seus pesos no cômputo geral da rentabilidade.

iii- os benefícios que podem ser esperados a longo prazo através da participação de outros sistemas no uso de componentes do sistema a ser desenvolvido ou extendido.

Estimativas sempre são imprecisas. Esta imprecisão será tão maior quanto maior for o desconhecimento a respeito do sistema a ser desenvolvido. Ou seja, durante as primeiras atividades do desenvolvimento, são cometidos erros de estimativa maiores do que os cometidos nas últimas fases. Estes erros podem ser da ordem de 100% [Gildersleeve74]. Existe portanto um fator de risco considerável ao dar-se partida no desenvolvimento de um sistema automatizado, onde este risco diminui à medida que o desenvolvimento prossegue.

Na figura 3.5 mostramos o comportamento genérico dos riscos técnico e de utilidade com relação à época no projeto de desenvolvimento. Risco técnico é a probabilidade de especificar-se algo que seja tecnicamente inviável (tecnologia inexistente, ou até impossível), ou que seja economicamente inviável. Risco de utilidade é a probabilidade de especificar-se e, consequentemente, implantar-se algo que não atenda às necessidades do usuário. Ou seja, é o risco de desenvolver-se o produto errado. Para permitir comparações, deve ser utilizado o risco financeiro. O risco financeiro é o produto do risco pelo custo de desenvolvimento.

O risco técnico tem o comportamento como apresentado na figura 3.5, uma vez que, durante a fase de definição, frequentemente são incluidos e/ou excluidos requisitos do sistema sem haver uma justificativa de viabilidade técnica ou econômica adequada para tal. Com o decorrer da definição, tais requisitos são examinados, sendo justificada a inclusão ou eliminação destes. Desta forma o risco de falha técnica é reduzido à medida que estas justificativas são produzidas.

O risco de utilidade é reduzido no decorrer do desenvolvimento do sistema, somente se houver participação do usuário. Como já foi dito, esta participação deve ocorrer pelo menos na fase de definição e nas sub-fases de aprovação dos produtos de cada fase do ciclo de vida do sistema.

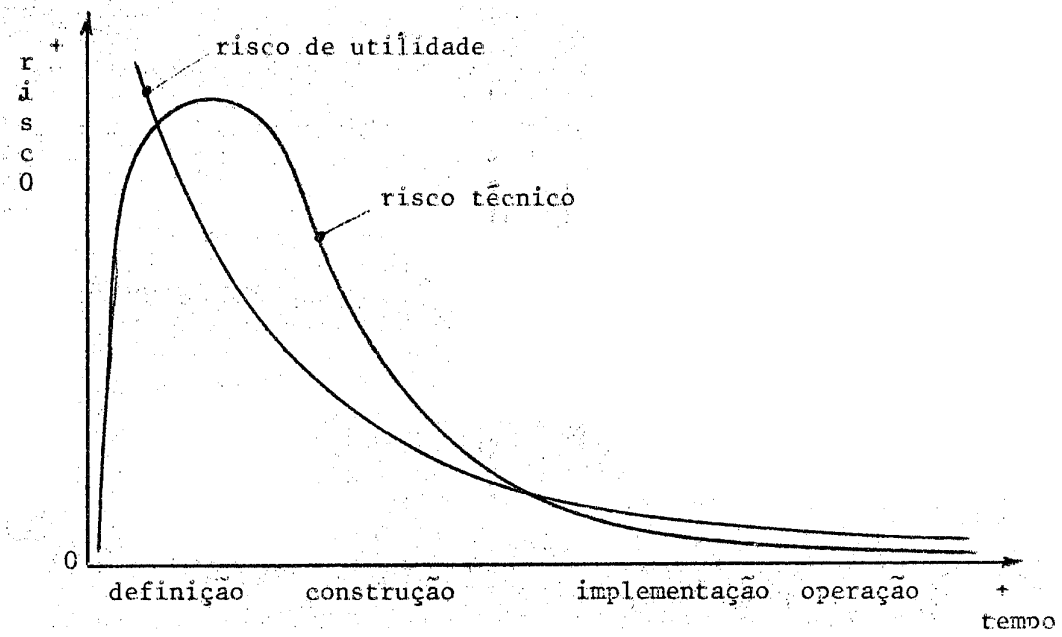


Figura 3.5 Riscos técnico e de utilidade.

Devido aos erros de estimativa, a curva da rentabilidade estimada deveria ser traçada sob a forma de uma "faixa de rentabilidade". Com o decorrer do tempo e o consequente aumento do conhecimento relativo ao sistema sendo desenvolvido, ou da extensão sendo efetuada, esta faixa de rentabilidade é tornada mais estreita.

Na figura 3.6 é apresentada uma faixa de rentabilidade típica. O aspecto da figura 3.6 é bastante comum, apesar do sistema poder levar a um lucro considerável (limite superior), ou a um prejuízo (limite inferior). A discrepância entre os dois limites é diminuida com o decorrer do tempo. Consequentemente, devemos reavaliar com frequência a estimativa da faixa de rentabilidade do sistema sendo desenvolvido ou estendido. Durante estas reavaliações poderá ser reconhecido que o ponto de retorno total do investimento (ponto $\underline{d}$ da figura 3.4) jamais será alcançado. Neste caso, o melhor que se faria é, ou cancelar o projeto, ou examinar quais os requisitos que, se eliminados ou incluidos, tornariam o sistema novamente rentável. Cabe observar, porém, que o custo do desenvolvimento do projeto cancelado deve ser incorporado ao custo

do sistema antigo, ou do sistema substituto a ser definido. Somente após ter-se feito isto, é que poderemos determinar efetivamente se o projeto deve ser cancelado ou não. Ou seja, os custos de amortização de um sistema X são "herdados" pelo sistema Y que virá a substituir o sistema X, mesmo que o sistema X não tenha sido completado ou posto em operação.

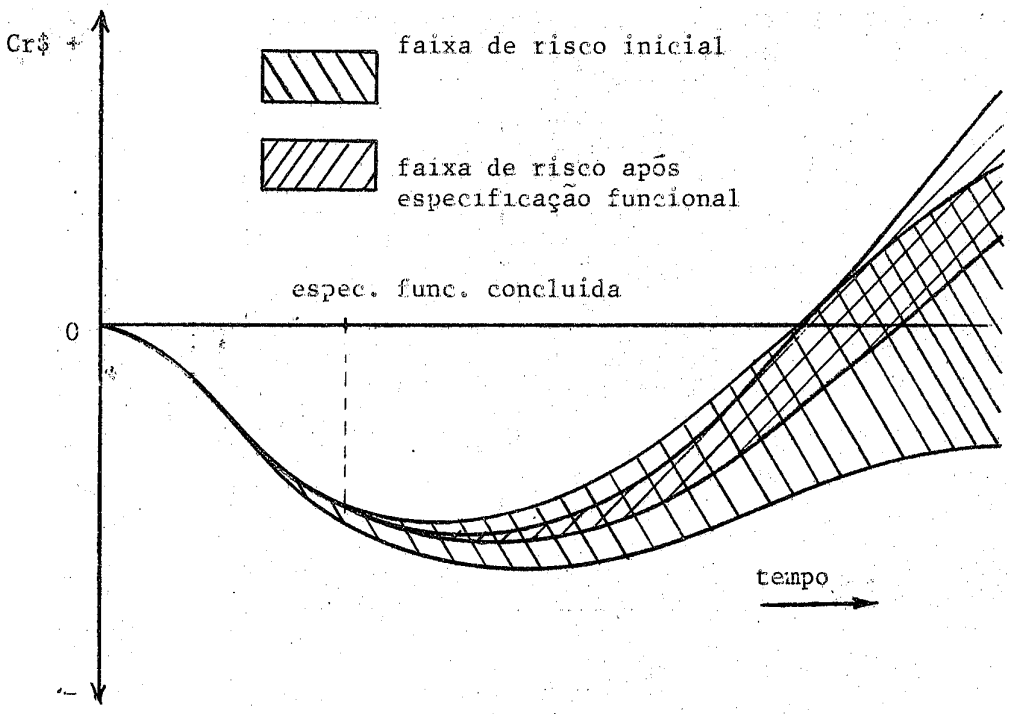


Figura 3.6 Aspecto genérico de faixas de rentabilidade.

3.3 Desenvolvimento Controlado.

Sistemas automatizados tendem a levar bastante tempo para serem desenvolvidos. Necessitam também diversos recursos que, se não disponíveis, podem causar sérios transtornos ao desenvolvimento. Devido a sua natureza, o desenvolvimento de sistemas automatizados possui restrições quanto ao grau de paralelismo com que pode ser levado a cabo. Consequentemente necessita-se de um plano de desenvolvimento que determine, entre outras coisas:

i- as atividades de desenvolvimento;

ii- as datas em que estas atividades deverão iniciar e terminar;

iii- os recursos e produtos necessários para iniciar-se cada atividade;

iv- os produtos resultantes de cada atividade;

v- a data em que os produtos resultantes estarão disponíveis e, também, a data em que estarão aprovados (controle de qualidade e de utilidade completos);

vi- a evolução do consumo de recursos discriminado por recurso (financeiro, pessoal, equipamentos, material etc.) e por atividade.

O plano de desenvolvimento nada mais é do que uma descrição da sequência de atividades que, se executadas, nos levarão ao produto final desejado. Existe então a necessidade de contínuo controle da execução segundo o plano. Para que isto seja possível deverão ser incluidos pontos de controle no plano. Para permitir a objetividade na determinação do alcance dos pontos de controle, estes devem ser definidos em termos de produtos acabados e aprovados, e não em termos de esforço, ou consumo de recursos.

Para permitir um acompanhamento efetivo, devemos poder determinar, com frequência, que pontos de controle foram alcançados. Isto requer a criação de uma hierarquia de atividades, em que as atividades dos níveis mais baixos desta hierarquia tenham uma duração estimada de duas a tres semanas. Cada uma destas atividades deve levar a produtos acabados e aprovados. Desta forma serão tornadas raras as crises, uma vez que poder-se-á determinar com antecedência a possibilidade de chegar-se a uma. Em geral esta antecedência é suficiente para tomar medidas que evitem crises, ou pelo menos medidas que reduzam o seu impacto.

Para ser útil, o plano deve ser antes de mais nada realista. De nada adianta produzir-se um plano, mesmo que impecável quanto à forma e consistência interna, se o plano não for exequível. O plano deve ser examinado, portanto, com o máximo de rigor e somente ser aprovado quando houver acordo entre todos que participaram de sua confecção. A confecção do plano deve ser efetuada por todos aqueles que serão responsáveis por executá-lo. De outra forma não será possível garantir-se a exequibilidade do plano.

É claro que poderão ser encontrados erros no plano ao executá-lo. Estes erros advêm principalmente de erros de estimativas devidos à falta de conhecimento de detalhes relativos ao sistema a ser desenvolvido quando da confeção do plano. A existência de tais erros não é nociva em geral. Nociva é a existência de erros conhecidos já antes da aprovação, e nociva é principalmente a falta de um plano de desenvolvimento, uma vez que isto fatalmente levará a uma falta de objetividade.

O plano de desenvolvimento é elaborado logo após à aprovação da proposta de desenvolvimento (ver seção 4.3). Nesta ocasião é impossível determinar-se detalhes de planejamento quanto às fases posteriores à fase de especificação funcional, pois estes detalhes dependem desta especificação. As fases posteriores constarão do plano inicial como se fossem constituidas por uma única atividade. À medida que o desenvolvimento for progredindo, mais detalhes quanto a estas fases posteriores vão se tornando disponíveis e, portanto, maior detalhe poderá ser dado aos sub-planos destas fases.

Quando a incerteza quanto ao plano global criado durante a fase de planejamento inicial for muito grande, a época em que os sub-planos das fases posteriores forem detalhados, são também ocasiões para uma revisão cuidadosa do cronograma, dos custos de desenvolvimento e da análise de custos e benefícios.

Na figura 3.7 mostramos a linha limite do grau de detalhe do plano ao final de algumas fases. Ao iniciarmos o projeto as primeiras fases estarão detalhadas e as últimas estarão pouco detalhadas. Ao final do projeto, o plano estará redigido em seu nível de detalhe total.

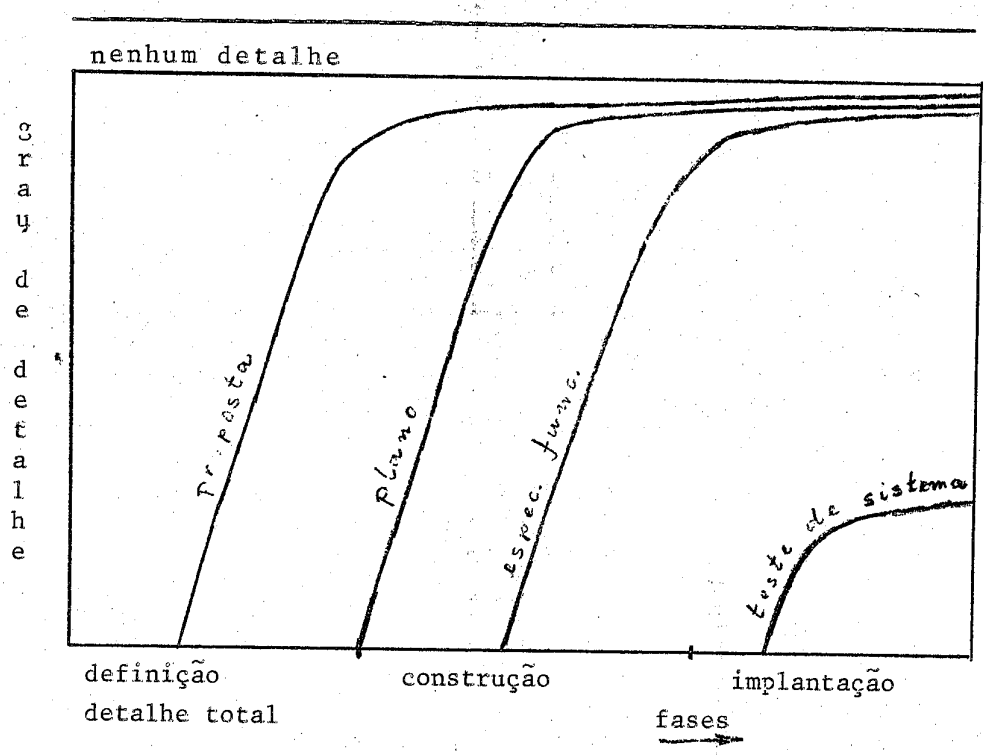


Figura 3.7 Evolução do nível de detalhe do plano de desenvolvimento.

Para permitir o planejamento inicial ser efetuado com suficiente credibilidade, necessitamos dispor de estatísticas relativas a projetos passados. Estas estatísticas permitirão a produção de estimativas mais precisas para o projeto como um todo. A coleção destas estatísticas requer a existência de uma metodologia de coleta de dados confiável, de baixo custo e produzindo dados úteis para a confecção de futuros planos [Leite79].

Os produtos, quaisquer que sejam as suas naturezas, resultantes de uma atividade, ou conjunto de atividades, devem sempre passar por uma aprovação. Desta forma é assegurado que o volume de erros residuais e o custo de eliminação destes erros seja diminuído. O controle de qualidade continuado assegura, também, melhores práticas de desenvolvimento. Estas consequências da aprovação continuada do sistema em desenvolvimento levam a uma redução do custo de desenvolvimento, apesar de terem a tendência de aumentar as fases de especificação. A redução do custo advêm principalmente de uma substancial redução do esforço necessário para testes, reajuste ("retrofit"), atividades de adaptação e correção durante o desenvolvimento.

Em alguns casos é desejável iniciar-se uma atividade mesmo quando os produtos necessários para tal ainda não estejam aprovados formalmente, porém já estejam prontos. Algumas das razões para isto poderão ser:

i- o desejo de antecipar a descoberta de dificuldades com o produto acabado, provocando a antecipação das consequentes correções. O aumento do custo decorrente da correcão destes produtos é pequeno, uma vez que as equipes que os produziram ainda existem, estando empenhadas em efetuar a aprovação. Quando o desenvolvimento é feito de maneira "top down", podem ser gerados problemas que afetem frontalmente atividades posteriores. A antecipação destas atividades procura reduzir a incidência destas dificuldades.

ii- a necessidade de reduzir atrasos. Cabe alertar aqui, que deve ser conduzida uma cuidadosa análise dos riscos, uma vez que uma frequência elevada de correcões e reajustes nos produtos de entrada de uma atividade, podem causar um aumento demasiado no esforço previsto para efetuar esta atividade. Foi constatado, também, que a tentativa de forçar a redução de prazos de desenvolvimento de sistemas automatizados, causa consideráveis aumentos de custo de desenvolvimento, se é que esta redução é conseguida de todo [Brooks75, Putnam77a].

É boa praxe tentar evitar-se problemas com atrasos, ou com aumentos nas necessidades de recursos, através da previsão de contingências. Desta forma pequenos desvios do planejado são absorvidos sem maiores dificuldades.

Nas referências seguintes são propostos diversos modelos para planos de desenvolvimento [Metzger73, Gildersleeve74, Rocha78].

Por mais que se procure acertar de saída, ocorrerão sempre eventos imprevistos que obrigam a modificar, durante o desenvolvimento, o plano e/ou as especificações de componentes do sistema automatizado. Estas necessidades de alteração podem ser devidas à percepção de necessidades do usuário não previstas antes, mudanças do meio ambiente em que o sistema operará (por exemplo mudanças de legislação), mudanças de disponibilidade de recursos etc. Em outras palavras, o processo de desenvolvimento tende a ser um processo iterativo, em que, algumas vezes, precisamos optar por técnicas do gênero tentativa e erro. Contudo, devemos estar sempre preocupados com a minimização do número de iterações. Para conseguir isto, será necessário, em alguns casos, desenvolver protótipos, desenvolver modelos de simulação, ou conduzir experimentos simples, para que possamos reduzir o grau de incerteza. Novamente, o investimento com a condução de tais experimentos deve ser analisado com cuidado, investindo-se mais nesta análise quanto maior for a incerteza [Mangold74].

Erros, ou inadequações, de um produto podem advir de falhas de produção, ou de falhas de especificação. Na realidade a especificação de um produto é o resultado de uma atividade anterior. Ao deparar-se com um erro de especificação, é normal ter-se que recompor o ambiente no qual esta especificação foi desenvolvida. Isto pode ser caro, e o custo aumenta com o aumento do intervalo de tempo decorrido desde a aprovação da especificação até a observação da sua inadequação. O aumento do custo é devido à necessidade de corrigir-se outros produtos que foram desenvolvidos a partir da especificação descoberta estar em erro. O desejo de evitar este incremento em cascata do custo, é a principal razão para que se efetuem aprovações frequentes, iniciando já nas primeiras atividades do desenvolvimento do sistema automatizado.

Na figura 3.8 mostramos o ciclo de produção. O ciclo consta de dois sub-ciclos, um de produção e correção do produto, e outro de revisão e correção da especificação deste produto.

A especificação de um produto foi, por sua vez, o produto elaborado durante algum ciclo de produção anterior ao que agora está em curso, exceto, é claro, quando da confeção da proposta de desenvolvimento que possui um ciclo de produção próprio [Staa79]. Uma vez pronta a especificação, ela poderá servir como partida para o desenvolvimento de diversos outros produtos que, por sua vez, poderão vir a ser especificações. Existe portanto uma hierarquia e uma ordenação parcial de atividades de produção com relação à época em que ocorrem os ciclos de produção. Esta ordenação parcial é o principal fator limitante da capacidade de desenvolvimento em paralelo.

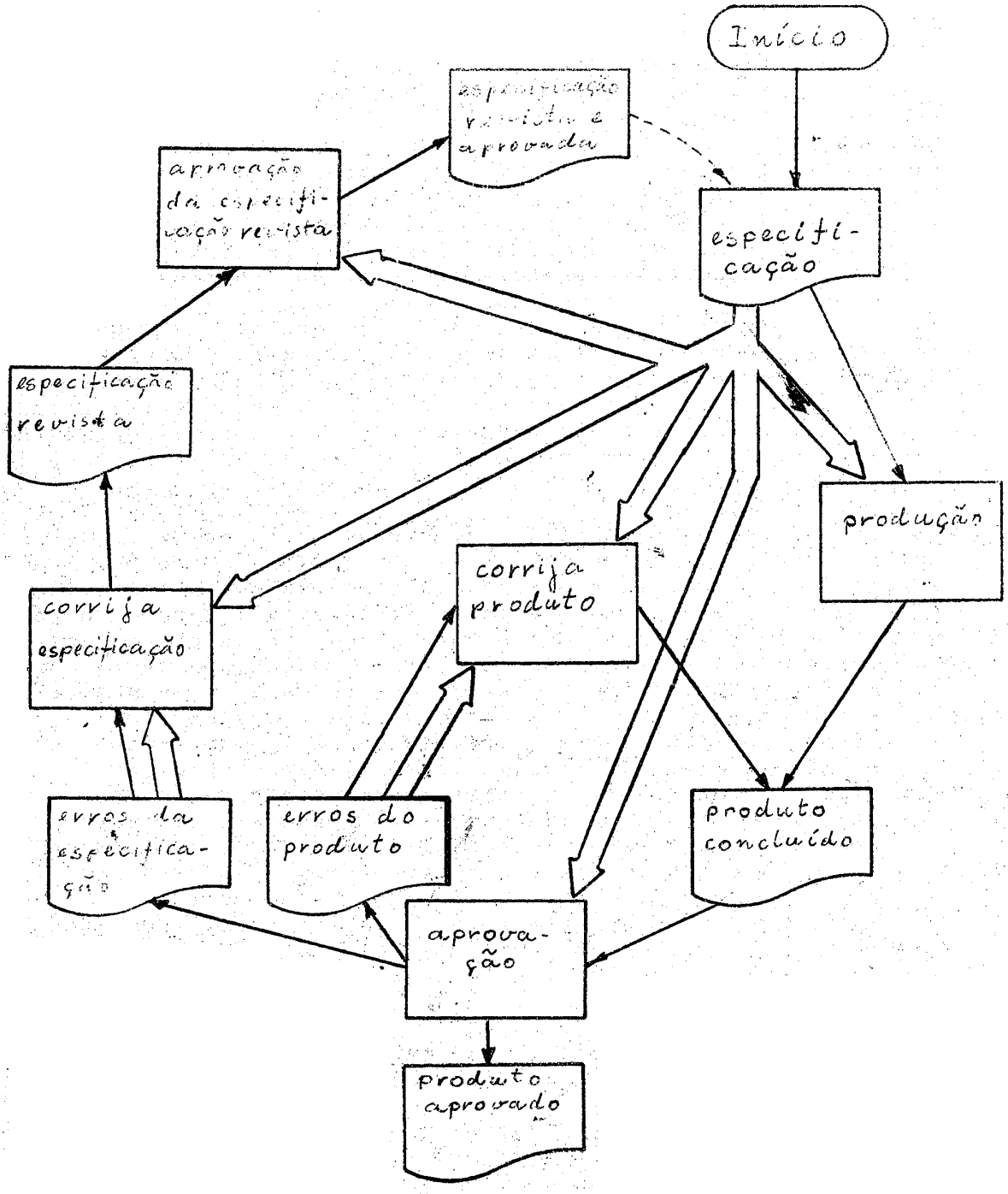


Figura 3.8 Ciclo de produção.

Em um determinado instante estarão ativos diversos ciclos de produção dependentes de uma mesma especificação. Qualquer alteração nesta especificação inicial, tem como con-

sequência alterações nos ciclos de producação efetuados ou em progresso e que dependem desta especificação inicial. Deve ser evidente que o custo de efetuar a alteração da especificação é proporcional ao número de ciclos de produção completados ou ativos e dependentes desta especificação.

Como já foi dito, apesar de todos os nossos esforços para evitá-las, ocorrerão alterações em especificações. Deve existir, então, um controle sobre pedidos de alterações. Neste controle deve ser feita uma análise dos efeitos (custo-benefício) da alteração, e deve ser garantida uma disseminação rápida e eficaz das alterações aceitas.

Alterações durante o desenvolvimento são efetuadas, em geral, da mesma forma como o são correções, adaptações e extensões durante a fase de operação. Para que o custo destas alterações seja mantido pequeno, desde cedo os requisitos de qualidade da classe "manutenibilidade" devem ser observados. Isto literalmente força a existência de documentação, critérios de aprovação, bem como de dados para efetuá- a, já desde as primeiras atividades do projeto de desenvolvimento.

A atividade de controle de alterações deve ser cuidadosamente definida e planejada para evitar oscilações durante o desenvolvimento e incrementos substanciais nos custos do desenvolvimento [Metzger73, Rocha78]. As oscilações podem decorrer de correções em cascata, ou de correções repetidas em um mesmo produto. Estas oscilações são eliminadas, desde que as alterações sejam sempre examinadas em todos os seus aspectos, em particular, seus efeitos colaterais.

Como já foi dito, o controle de alterações e a constante medida de qualidade dos produtos, exigem a presença de documentação. Ou seja, a documentação passa a existir antes do desenvolvimento do componente. Desta forma a documentação passa a ser a guia de desenvolvimento e não somente algo a ser entregue no final.

Várias vantagens podem ser enumeradas. A primeira certamente é a própria existência da documentação, que, devido a ter sido utilizada durante o desenvolvimento possui alto grau de qualidade.

Outra vantagem da existência de documentação é a possibilidade de poder-se trocar e/ou contratar novo pessoal sem afetar excessivamente o desenvolvimento. Finalmente, a existência de documentação leva a uma maior objetividade ao medir-se a qualidade, a utilidade e os benefícios de sistemas de computação.

Segundo alguns, a preocupação com documentação e medição da qualidade, da utilidade e dos benefícios, aumenta o tempo necessário para implantar-se o sistema. Na prática, porém, o tempo usualmente não é aumentado, mesmo em se tratando de projetos pequenos. A razão principal para a redução do esfor-

ço e do tempo necessário ao desenvolver sistemas em ambiente controlado é a sensível redução de tentativas e erros, e a maior objetividade quanto aos caminhos a tomar para conseguir-se chegar ao produto desejado [Yohe74].

O desenvolvimento de sistemas automatizados é uma atividade que requer abstração e criatividade, ou engenhosidade se preferirem. Sendo efetuada de forma industrial, ao invés de artesanal, é também uma atividade que requer disciplina e método. Somente por estas razões pode-se afirmar existirem poucas pessoas qualificadas para o desenvolvimento industrial de sistemas automatizados.

A afirmação "se funciona está certo" não se aplica no caso de sistemas automatizados, devido à necessidade do sistema ser adequado ao usuário, devido ao custo elevado do desenvolvimento, da adaptação e da alteração destes sistemas, e devido à necessidade de durabilidade. Se sistemas não forem desenvolvidos com cuidado e esmero, rapidamente tornar-se-ão decrépitos, difíceis de adaptar ao ambiente em constante mutação, sendo fatalmente descontinuados prematuramente, sem sequer terem componentes parcialmente reaproveitados pelos novos sistemas que os virão substituir.

O problema central é então o da disponibilidade de pessoas capazes e de ferramentas que tornem elevada a produtividade destas pessoas, e maior a qualidade dos produtos gerados. Ou seja, o desenvolvimento de sistemas automatizados está hoje mudando do estágio inicial "labor intersive para um estágio mais maduro "capital intensive", onde poucas pessoas, melhor qualificadas, dispondo de poderosas e eficazes ferramentas, desenvolvem melhores sistemas automatizados [Wegner78].

3.4 Aspectos Técnicos.

Como qualquer outro sistema, também sistemas automatizados não existem isoladamente. Ou seja, qualquer sistema automatizado é componente de um sistema maior.

Por sua vez, sistemas automatizados são compostos por componentes automatizados (programas) e não automatizados (procedimentos). Procedimentos não têm a haver somente com computadores e equipamentos periféricos. Um sistema automatizado deveria ser observado em primeira instância, englobando todas as atividades que o poderiam afetar. Por exemplo, um balconista que, apesar de exclusivamente preencher notas de venda, é uma pessoa que participa ativamente do sistema de controle de estoque. A maior ou menor dificuldade do balconista em preparar a nota de venda levará a resultados respec-

tivamente piores ou melhores. Sendo mais precisos, ao observarmos um sistema, deveremos identificar todos os seus usuários em potencial.

Uma vez observado o sistema global, podemos determinar os seus subsistemas, identificando os componentes a serem automatizados e os que serão processados manualmente. A qualidade do sistema automatizado depende de todos os seus componentes, mesmo aqueles que não foram automatizados. Portanto, torna-se necessário especificar, produzir e certificar procedimentos com o mesmo rigor que dispensamos à produção dos programas. Da mesma forma como são introduzidos controles de qualidade e auditoria em programas, procedimentos deverão possuir suficientes redundâncias para permitir controles de qualidade e auditoria [Benjamin71].

Ao terminar-se o desenvolvimento do sistema novo, torna-se necessário passar a operá-lo e, oportunamente, descontinuar o sistema antigo. Para poder-se operar o sistema novo, todos os arquivos (banco de dados) devem estar criados e corretos, os usuários deverão estar treinados, os formulários devem estar impressos etc. Conseguir-se chegar a este ponto, requer o desenvolvimento de um sistema de conversão. Este sistema entrará em operação durante a fase de desenvolvimento, desaparecendo, pelo menos parcialmente, ao iniciar-se a fase de operação. Apesar de ter uma esperança de vida curta, o sistema de conversão deve ser desenvolvido com cuidado, pois a operação do sistema novo depende diretamente da qualidade dos resultados produzidos pelo sistema de conversão. Se estes resultados forem de baixa qualidade, o sistema novo corre o risco de ter uma morte prematura, pois os resultados que produzirá serão também, de baixa qualidade, independentemente do nível de qualidade real do sistema novo.

O desenvolvimento de sistemas automatizados pode ser conduzido de inúmeras maneiras. Da mesma forma a documentação e programação poderão ser feitas de muitas maneiras, levando a resultados de qualidade variável. É necessário então desenvolver-se técnicas adequadas e tornar estas técnicas normas industriais da instituição. A própria existência de normas melhora a capacidade de comunicação, reduz gastos com treinamento de pessoal, torna mais disciplinada a atividade de desenvolvimento e, se baseada em princípios adequados, contribui para o aumento da durabilidade e, consequentemente, da rentabilidade do sistema.

4 Ciclo de Vida.

Qualquer sistema passa por um ciclo de vida. Em particular sistemas automatizados passam pelas fases de definição, construção, implantação e operação. A tomada de consciência do ciclo de vida de sistemas automatizados, trouxe consigo todo um arsenal de técnicas mais ou menos efetivas para o planejamento, controle de produção, desenvolvimento, controle de qualidade etc., vinculadas ao ciclo de vida. O objetivo destas ferramentas é procurar maior fidelidade nas estimativas de tempo e esforço de desenvolvimento, garantir a visibilidade do desenvolvimento, permitir a antecipação da aprovação, enfim, permitir ter-se confiança de estar desenvolvendo o que se deseja dentro dos limites de tempo, esforço, utilidade e qualidade previstos.

Diversos autores descreveram ciclos de vida para produtos de software [Benjamin71, Metzger73, Gildersleeve74, Rocha78, Thausworthe78]. Neste texto procuramos identificar os produtos de atividades fortemente correlacionados e os pontos em que é mais intensa a atividade de aprovação desses produtos. Utilizando este objetivo chegamos a um ciclo que difere ligeiramente dos apresentados na literatura corrente.

4.1 Produção e Aprovação dos Resultados.

Cada fase do ciclo de vida de um sistema é composto por um número variável de atividades. O conjunto de produtos desenvolvidos por estas diversas atividades é, então, o "produto" da fase.

As atividades de uma fase podem ser decompostas em sub-atividades e estas, por sua vez, podem ser decompostas em sub-sub-atividades e assim por diante até chegarmos a tarefas indivisíveis. Similarmente para cima podemos agregar fases em super-fases etc. As composições e decomposições são, a princípio, totalmente arbitrárias. Durante a decomposição somente deve ser observado que as sub-atividades cubram completamente a respectiva atividade. Devido a esta liberdade de compor e decompor fases e atividades é que são tantas as definições de ciclo de vida de sistemas automatizados. Cabe observar que, para cada uma das atividades descritas, existe um ciclo de produção.

Sistemas por sua vez são decomponíveis em sub-sistemas, ou componíveis em super-sistemas. Quando este desenvolvimento é dirigido por uma especificação do sistema envolvente, teremos o que se chama convencionalmente de desenvolvimento "top-down". Quando o desenvolvimento do sistema envolvente é através da composição de seus sub-sistemas componentes, sem que exista de antemão uma definição do sistema envolvente, falamos de desenvolvimento "bottom up".

Apesar de ser desejável, o desenvolvimento "top down" nem sempre será possível. Primeiro devido à própria envergadura do sistema envolvente e, segundo, por ser sempre possível encontrar-se um sistema envolvente do qual o sistema presente é componente. Pode-se também desejar criar um sistema envolvente a partir de sistemas já existentes, ou pelo simples fato de isto tornar-se possível tecnologicamente, ou por ter-se obtido conhecimentos melhores a respeito deste sistema envolvente. Cabe observar ainda, que na prática sistemas não são desenvolvidos exclusivamente de maneira "top-down" ou "bottom-up".

Cada atividade gera um produto, por extensão, cada fase (conjunto de atividades) gera um ou mais produtos. Cabe aqui conceituar melhor o que se entende por atividade. Durante o desenvolvimento de um produto, podem ser efetuadas diversas tarefas, que podem ser por exemplo: a produção de esboços de especificações, a redação de textos preliminares, a transcrição de textos de um meio para outro (digitação, datilografia), a condução de exames de requisitos de qualidade, etc. Do ponto de vista do controle e acompanhamento do desenvolvimento, os resultados destas tarefas não têm importância, tendo importância somente o resultado final a ser entregue. Uma atividade é, portanto, um conjunto de tarefas e/ou outras atividades, que leve a um produto aprovado e que tenha interesse do ponto de vista do controle e acompanhamento do desenvolvimento (execução do plano).

A cada atividade deve corresponder um ponto de controle, sejam estas atividades terminais (não decomponíveis) ou não terminais (decomponíveis em outras atividades). Já foi dito que atividades podem ser decompostas em sub-atividades, formando uma hierarquia de atividades. Como, para cada atividade, temos um ponto de controle, existe, também, uma hierarquia de pontos de controle. Pontos de controle sucedem as atividades a que dizem respeito, e correspondem a tarefas de aprovação de todos os produtos destas atividades e de suas sub-atividades caso existam. Devido ao seu grau de importância maior, o ponto de controle ao final de uma fase é chamado de marco.

Nem todos os pontos de controle requerem a participação de membros de todas as entidades envolvidas. No caso de marcos, porém, esta participação mais ampla é importante, uma vez que somente desta forma será possível assegurar-se a qualidade, utilidade e rentabilidade do sistema.

A aprovação de um marco requer tempo. É necessário então ser definido um ponto de controle onde o resultado da fase é entregue após ter sido aprovado pelo responsável por esta fase. Após, é efetuada uma pseudo-atividade na qual os resultados são examinados por membros de todas as entidades. Ao final desta pseudo-atividade o produto estará formalmente apro-

vado e o marco alcançado. Correções que porventura se fizerem necessárias para a aceitação do produto, serão efetuadas durante esta pseudo-atividade.

Como sistemas podem ser desenvolvidos através do desenvolvimento de sub-sistemas, cada qual com seu ciclo de vida próprio, poderá ser definida uma hierarquia de marcos para o controle do desenvolvimento destes sub-sistemas e, consequentemente, do sistema envolvente.

Como consequência da contínua aprovação de produtos e conjuntos de produtos à medida que o desenvolvimento prossegue, as atividades de teste e aprovação passam a ser primordialmente comprovações da qualidade do sistema desenvolvido, ao invés de constatações da existência de inúmeros erros e/ou inadequações. Isto decorre de uma redução significativa dos erros e inadequações introduzidas nos produtos, e não de uma possível superficialidade dos testes. Desta forma é reduzido o número de iterações de correção, sendo também antecipadas as dificuldades que seriam descobertas somente quando o esforço de recuperação já tiver-se tornado excessivo.

Cabe observar aqui, que a contínua aprovação de produtos ao final da cada atividade e fase, requer a presença de critérios de teste (plano de teste), de documentação e de especificações, tal como evidenciado no ciclo de produção (figura 3.8).

4.2 Documentação.

Os produtos do desenvolvimento de sistemas automatizados são equipamentos, por exemplo terminais de propósito particular, e documentos. Estes documentos poderão existir sob forma impressa, ou seja legível por seres humanos, ou estarão registrados em algum meio legível mecanicamente. Em outras palavras, constituem documentos não só os textos descritivos, como também os programas e arquivos resultantes do desenvolvimento.

Quando desenvolvemos um sistema por refinamentos sucessivos, partimos do geral para o detalhe. Isto reflete-se na produção, também por refinamentos sucessivos, dos documentos. Isto pode causar a produção repetida de trechos de documentos à medida que mais detalhes são incorporados. Para evitar tornar-se repetitiva a redação ou transcrição de documentos é necessário um planejamento cuidadoso da sequência com que serão adicionados detalhes aos documentos. Idealmente dever-se-ia contar com suporte automatizado para a produção de documentos [Mantley74, Teichroew77].

Documentos devem ser concisos, objetivos, corretos, completos, certificáveis, legíveis, etc. É claro que é difícil atingir-se a todos estes objetivos. Documentos devem ser sobretudo compreensíveis por todos aqueles que os deverão ler. Existe, portanto uma barreira de comunicação. Para vencermos esta barreira, é necessário utilizar-se uma linguagem adequada para transmitir as informações que desejamos transmitir por intermédio dos documentos. Cabe observar aqui que desejamos transmitir informação com precisão, portanto não podem existir ambiguidades nem tampouco itens que dêem margem a interpretações diferentes por diferentes leitores. Isto requer um estilo de redação apropriado a textos técnicos. Este estilo, aliado à linguagem utilizada, deve levar a documentos fáceis de ler e entender, e suficientemente formais para que possamos examinar e aprovar com maior rigor e menor esforço estes documentos [Wolverton77].

As linguagens utilizadas devem ser sucintas, ou seja, devem ser adequadas em número de detalhes ao nível de detalhe em que está sendo produzido o documento. Assim nos níveis de menor detalhe desejamos ver o sistema como um todo, enquanto que nos níveis de detalhe elevado desejamos ver as ações a serem efetuadas ao operar-se o sistema. Estas últimas devem ser apresentadas sob a forma de programas devidamente comentados.

As hierarquias de linguagem deveriam permitir uma transição fácil de um nível hierárquico para outro. As propostas mais radicais neste sentido, propõem uma única linguagem de especificação possuidora da propriedade de ser hierárquica [Ross77, IBM74].

Nos primeiros níveis as linguagens mais adequadas são as gráficas, por permitirem, em pouco espaço, conduzir um volume considerável de informações e, também, permitirem um exame mais aprimorado da sua qualidade. São exemplos de linguagens gráficas: fluxos de dados; relações entre entrada, processamento e saída; diagramas de transição; diagramas estruturados; diagramas hierárquicos; etc.

Nos níveis de maior detalhe, as linguagens mais adequadas são os próprios programas com comentários adequados [Lucena78]. Os comentários incluídos devem descrever, entre outras coisas, as condições que devem ser satisfeitas antes e após grupos de comandos contidos em blocos estruturados.

Além dos documentos formais, é requerido, em muitas ocasiões, texto explanatório que facilite a compreensão dos documentos. Também este texto deve ser sujeito a estilo e linguagem próprios.

Na literatura atual são propostas diversas linguagens de documentação mais ou menos formais [Ross77, IBM74, Lucena78, Yeh77, Teichroew77]. Uma abordagem mais detalhada destas linguagens foge ao escopo deste texto. Falta, porém, uma hierar-

quia homogênea de linguagens, possuindo o devido suporte computacional e permitindo a produção de documentos melhores com menor esforço.

Além de linguagens de documentação, devem existir também normas técnicas e glossários, que tornem homogêneos os documentos produzidos pelas diversas equipes ativas nos diversos projetos de desenvolvimento da instituição. A existência de normas facilita o entendimento dos documentos, uma vez que existirá somente um jargão utilizado para a comunicação. Desta forma é reduzido o custo de produção, aprovação e compreensão dos diversos documentos.

Documentos devem ser completos. A verificação da completeza tende a ser uma operação demorada e repetitiva. São sugeridos, então, o desenvolvimento e a normalização de roteiros de documentação que levem a documentos completos (roteiros "prêt-à-porter"). Estes roteiros possivelmente exigirão mais dados do que os necessários para determinados produtos. Estes requisitos excessivos devem ser indicados com "não se aplica", ao invés de serem simplesmente omitidos. Desta forma estará assegurada a completeza do documento, enquanto o roteiro levar a documentos completos.

Sistemas automatizados podem ser visualisados como sendo compostos de processos (procedimentos e programas) que recebem, transformam e geram dados, sendo comum dados produzidos em um processo servirem de entrada para outro. É evidente que tais dados precisam ser especificados cuidadosamente, antes mesmo de iniciar-se o desenvolvimento dos processos entre os quais fluem.

Para poderem operar corretamente, os processos necessitam receber dados que atendam às especificações de entrada. Mais uma vez isto implica na existência de uma especificação rigorosa dos dados de entrada antes de iniciar-se o desenvolvimento do processo que recebe estes dados.

Um dos documentos a serem produzidos durante o desenvolvimento, é o diretório de dados [Furtado78]. Este documento contém informações a respeito do fluxo de dados, das especificações dos dados, das normas relativas à representação dos dados, das restrições de utilização dos dados etc. O diretório de dados é, portanto, o documento central em torno do qual gravita a restante documentação do sistema automatizado.

O diretório de dados é, também, um instrumento de padronização, o que traz consigo uma maior facilidade de integração e comunicação entre os diversos sistemas e grupos de desenvolvimento de sistemas automatizados.

4.3 Fase de Definição.

Durante a fase de definição são especificados os atributos de qualidade e utilidade, bem como os benefícios esperados, sendo ainda produzidos o plano de desenvolvimento do sistema automatizado, a especificação funcional e o diretório de dados lógico. A fase de definição possui as seguintes subfases:

Concepção

Objetivo
- identificação de necessidades ou de possíveis melhorias, que poderiam ser atendidas através de um novo sistema ou através da extensão de um sistema já existente.

Produtos
- descrição das dificuldades observadas;
- descrição das principais condições a serem observadas pela solução;
- versão preliminar da proposta de desenvolvimento.

Especificação de Requisitos

Objetivo
- produção da versão final da proposta de desenvolvimento;
- determinação dos atributos de qualidade e utilidade:
- determinação dos pesos e dos níveis a serem atingidos pelos atributos de qualidade e utilidade;
- estimativa dos benefícios esperados;
- esboço de uma possível solução.

Produtos
- exame geral do sistema atual. O nível desta observação deve assegurar constantemente a visão geral do sistema e não a observação de detalhes, devendo indicar os principais defeitos deste, defeitos estes que deverão ser sanados pelo novo sistema ou pela extensão ao sistema;
- esboço de solução evidenciando os requisitos de qualidade, de utilidade e os benefícios desejados;
- condições de relacionamento com outros sistemas;
- condições de operação e desenvolvimento do sistema novo ou da extensão;
- condições contratuais gerais;
- avaliação da proposta.

Aprovação da Proposta de Desenvolvimento

Objetivo
- assinatura do contrato, ou do memorando de início de desenvolvimento.

Produtos
- análise da rentabilidade;
- documento de aceitação assinado por representantes dos diversos grupos da entidade "usuário";
- documento de aceitação assinado por representantes da entidade "cliente";
- documento de aceitação assinado por representantes da entidade "coordenador";
- contrato de desenvolvimento, ou memorando, assegurando a disponibilidade de recursos para o desenvolvimento.

Planejamento Inicial

Objetivo
- escolha da entidade "desenvolvedor";
- determinar como será desenvolvido o sistema;
- estabelecer objetivos e cronogramas de desenvolvimento, de consumo de recursos e de entrega de produtos.

Produtos
- plano de desenvolvimento;
- normas técnicas: documentação, programação, aceitação por fase, controle de alterações, diretório de termos técnicos, segurança operacional, segurança de desenvolvimento, etc. Estas normas podem requerer o desenvolvimento de ferramentas de suporte ao desenvolvimento.

Aprovação do Plano de Desenvolvimento

Objetivo
- revisão da análise de rentabilidade;
- explicitação dos requisitos de qualidade e utilidade e dos benefícios prioritários;
- aprovação dos resultados por membros das diversas entidades.

Produtos
- plano de desenvolvimento aprovado;
- normas técnicas aceitas e divulgadas;
- proposta de desenvolvimento revista.

Especificação Funcional

Objetivo
- determinação das alterações de organização e métodos necessários para permitir a operação do novo sistema;
- especificação funcional completa da solução proposta;
- criação da parte lógica do diretório de dados.

Produtos
- levantamento detalhado do sistema atual;
- seleção de software;
- determinação do fluxo de dados no sistema como um to-

do. Os dados de que se trata aqui, são na realidade os documentos e os arquivos que transitam e são mantidos pelo sistema. Estes documentos e arquivos devem ser descritos quanto ao seu conteúdo, sem especificar detalhes de organização física, uma vez que o objetivo do fluxo de dados é definir o que o sistema faz e não como faz;

- especificação funcional dos elementos componentes do fluxo de dados;
- especificação preliminar dos resultados a serem produzidos pelos diversos testes;
- dicionário de termos técnicos;
- parte lógica do diretório de dados. Compreende os nomes dos elementos e agregados de dados, seus limites de validade, sua interpretação, responsabilidades pela coleta dos dados, critérios de segurança e previsão dos custos e benefícios destes dados. Não compreende a organização dos dados, tamanhos, representações internas, codificações etc. [Furtado78];
- início da criação do manual do programador. O manual do programador é dirigido aos diversos programadores ativos no desenvolvimento do sistema automatizado, onde cada programador possuirá o seu manual próprio. O manual contém informações gerais (resumo do sistema, resumo do plano, normas técnicas, etc.) e informações dirigidas a cada programador (especificações, formatos, normas técnicas especiais, nomes de arquivos para os programas fonte, objeto e dados teste, etc.). O manual do programador é criado à medida que o sistema vai sendo desenvolvido, através da incorporação de mais detalhes. A sua criação finda na fase de codificação;
- organização e métodos necessários para operar o sistema sendo desenvolvido. Sistemas automatizados, para serem eficazes, devem ser simples, coesos e devem intercambiar o mínimo indispensável de dados entre seus componentes e o ambiente externo [Myers75]. O sistema antigo, principalmente se tiver sido criado por evolução histórica, frequentemente possui vícios e ineficiências internas que, se emuladas no sistema automatizado, comprometerão seriamente a eficácia deste. Torna-se necessário, então, criticar e modificar os procedimentos do sistema antigo, afim de que a nova forma de proceder permita a introdução do sistema automatizado ou de uma extensão ao sistema antigo;
- versão inicial, ainda incompleta, do manual do usuário. O manual do usuário contém informações sobre a utilidade do sistema, sobre dados a serem fornecidos, resultados que poderão ser obtidos, critérios de qualidade, utilidade e rentabilidade, etc. O manual do usuário é dirigido aos usuários diretos e auditores;
- especificação preliminar dos diálogos homem/máquina. Aqui também interessa somente a parte lógica, ou seja, o conteúdo destes diálogos e não sua disposição

física;
- especificação dos relatórios de auditoria;
- plano detalhado da fase de especificação lógica. Como já foi dito, ao criar-se o plano de desenvolvimento não devem ser fornecidos detalhes de fases posteriores à fase de especificação funcional, uma vez que grande parte das previsões dependem diretamente desta especificação. Os detalhes das fases posteriores devem ser fornecidos nas fases que imediatamente as antecedam.

Aprovação da Especificação Funcional

Objetivo
- revisão e fixação dos requisitos de qualidade e utilidade;
- revisão e fixação do plano de desenvolvimento;
- revisão da análise de rentabilidade;
- aprovação dos produtos da fase de especificação funcional;
- acordo de "congelamento" dos valores esperados e mínimos tolerados aos atributos de qualidade e utilidade, dos benefícios esperados, do plano de desenvolvimento e da especificação funcional.

Produtos
- revisão, correção e aprovação dos produtos gerados até agora;
- documento dando o acordo do "congelamento" das especificações, dos requisitos de qualidade e utilidade, dos benefícios esperados, e do plano de desenvolvimento. Este documento deve ser assinado por representantes das diversas entidades envolvidas. Cabe observar que continuam sendo possíveis alterações nos produtos gerados até agora, somente que o controle será formal, requerendo a aprovação de todos os signatários do acordo.

Neste instante existe uma especificação funcional do sistema a ser desenvolvido. Esta especificação determina os resultados que o sistema produzirá, os requisitos de qualidade e utilidade mínimos, e os benefícios esperados. Além disso estará definido como se pretende chegar ao produto final e quanto esforço será necessário para tal.

Alterações em produtos gerados até este instante tendem a ter repercussão fortemente negativa sobre o desempenho do desenvolvimento. Tais alterações devem, portanto, ser restritas àquelas que sejam amplamente justificadas. Sempre que forem feitas alterações nestes produtos, deve ser revisto o plano, devendo ser dada atenção particular às parcelas de consumo de recursos. Por esta razão é imprescindível um acordo de "congelamento", mesmo quando as diversas entidades pertencerem à mesma instituição.

O esforço consumido até este ponto é da ordem de 30% do esforço total do desenvolvimento [Brooks75, Boehm73]. Em alguns casos poderá ser mais. Isto é particularmente o caso quando existe pouco conhecimento a respeito do sistema a ser desenvolvido, seja por representar uma inovação tecnológica, seja por inexistir suficiente conhecimento do sistema em si. Nestes casos é sugerido incluir-se uma fase de desenvolvimento de protótipos anterior à fase de especificação funcional conforme discutido na seção 3.3.

4.4 Fase de Construção.

Nesta fase o sistema será construído em conformidade com a especificação. A fase de construção é constituida pelas seguintes sub-fases:

Especificação Lógica

Objetivo
- identificação dos programas e procedimentos;
- especificação rigorosa das interfaces entre os diversos processos (procedimentos e programas);
- diretório de dados, parte física completa;
- diálogos homem/máquina completamente especificados;
- manual do usuário completo.

Produtos
- especificação funcional dos programas e procedimentos elementares. Durante a especificação funcional do sistema, são desenvolvidas especificações para processos do sistema. Estes frequentemente são processos compostos, ou seja, são constituidos a partir de diversos programas e procedimentos. Por exemplo, um processo de atualização é, usualmente, composto por vários processos elementares, tais como: crítica de dados, correção de dados errados, ordenação, atualização propriamente dita etc. Durante a especificação lógica do sistema, cada um destes processos elementares será especificado funcionalmente;
- especifição dos arquivos (bancos de dados);
- parte física do diretório de dados. A parte física do diretório de dados descreve a organização física dos dados, sejam estes itens, agregados, registros, arquivos ou conjuntos de arquivos. A parte física não descreve procedimentos de validação, recuperação, manutenção, etc. A parte física descreve detalhadamente a interface entre cada processo elementar. Desta forma é facilitado o desenvolvimento em paralelo dos componentes do sistema automatizado;
- critérios de aceitação por processo;
- programa de treinamento;
- primeira versão completa do manual do usuário;

- versão inicial e incompleta do manual do operador. O manual do operador contém informações sobre como operar os diversos equipamentos e como instalar o sistema nestes equipamentos. Fazem parte deste manual as descrições dos diversos procedimentos, requisitos de equipamento, mensagens de erro, operações contingentes previstas, etc. O manual do operador é dirigido ao usuário produtor;
- plano detalhado da fase de especificação física;
- diálogos homem/máquina. Estes diálogos contêm as descrições das interações entre o homem e a máquina. Estes diálogos podem ser efetuados via terminais, ou através de formulários e relatórios utilizados e/ou produzidos pelo sistema. Quando for prevista uma "linguagem" para facilitar a produção de relatórios ad hoc, esta linguagem faz parte da interface homem-/máquina;
- especificação dos controles de segurança operacional;
- especificação dos controles de auditoria;
- definição (requisitos, plano, especificação funcional) do sistema de conversão completa.

Aprovação da Especificação Lógica

Objetivo
- revisão da análise de rentabilidade;
- aprovação da definição do sistema de conversão;
- aprovação dos produtos da fase de especificação lógica.

Produtos
- documento de aprovação dos produtos da fase de especificação lógica.

Especificação Física

Objetivo
- definição dos procedimentos completa;
- modularização dos programas.

Produtos
- especificação funcional dos módulos de programa;
- especificação das mensagens de erro e das operações de contingência;
- definição dos procedimentos completa;
- critérios de aceitação por módulo e composto de módulos;
- primeira versão completa do manual do operador;
- plano detalhado da fase de codificação;
- especificações lógica e física do sistema de conversão.

Aprovação da Especificação Física

Objetivo
- revisão da análise de rentabilidade;
- aprovação dos produtos da fase de especifição física.

Produtos
- documento de aprovação dos produtos da fase de especificação lógica.

Codificação

Objetivo
- produção de programas;
- produção de dados para testes;
- integração dos programas;
- execução do plano de teste de integração do sistema;
- aceitação do sistema de conversão.

Produtos
- listagens los módulos de programas;
- biblioteca de módulos;
- procedimentos e programas de prevenção e recuperação em caso de acidente durante o desenvolvimento documentados (documentos distribuidos), implantados e em operação (norma de segurança de desenvolvimento);
- listagens de resultados de testes;
- biblioteca de dados para testes;
- mensagens de erro revistas;
- manual do programador completo. O manual do programador é, na realidade, um conjunto de manuais, contendo os documentos relativos a todos os procedimentos programas criados durante o desenvolvimento. O manual do programador é dirigido ao programador de desenvolvimento e ao usuário manutenedor;
- parte de código do diretório de dados completa. O diretório de dados, além de normalizar nomes e propriedades físicas do dado, normaliza, também, o código para efetuar o acesso, a validação, a atualização, o controle de uso, a proteção, etc. [Furtado78]. A utilização deste código pelos programas simplifica a programação do sistema e a posterior produção de relatórios e respostas não previstas durante o desenvolvimento;
- sistema de conversão desenvolvido e aprovado, ou seja, codificação, teste de sistema e teste de aceitação do sistema de conversão completos.

Aprovação da Codificação

Objetivo
- revisão da análise de rentabilidade;
- comprovação da suficiência dos testes de módulos;
- comprovação da suficiência dos testes de integração

do sistema;
- aprovação dos resultados da fase de codificação;
- aprovação do sistema de conversão;
- acordo de disponibilidade de recursos para a instalação do sistema.

Produtos
- documento de aprovação dos produtos da fase de codificação;
- proteção de bibliotecas de programas e dados para teste;
- documento assegurando a disponibilidade dos recursos necessários para a instalação do sistema, ou da extensão.

No final da fase de construção o sistema já existe, não tendo sido aceito ainda para efeitos de operação. Além do sistema objeto, já existe e está aprovado o sistema de conversão.

Ao final da fase de construção, o usuário se compromete a tornar disponíveis os recursos necessários para as fases seguintes. Estes recursos são tipicamente pessoas a serem treinadas e recursos computacionais. Estes recursos podem tornar mais dispendiosas as atividades do usuário, uma vez que poderão ser necessárias mais pessoas para operar o sistema antigo e para receber o treinamento e operar o sistema novo. Caso estas atividades não sejam cuidadosamente planejadas, o caos poderá resultar.

4.5 Fase de Implantação.

Nesta fase são conduzidos diversos testes com o objetivo de verificar se o sistema efetua o que foi especificado, se estas especificações são satisfatórias e se ele efetivamente atende às necessidades dos diversos usuários. Os testes são conduzidos em diversos níveis. Primeiro é verificado se o sistema efetua o desejado utilizando-se para isto dados criados exclusivamente para teste. Após, o sistema é exercitado com dados reais, porém ainda executa em ambiente controlado, ou seja, instrumentado. Finalmente o sistema passa a ser operado em condições totalmente reais. No final desta última sub-fase o sistema antigo é descontinuado. São seguintes as sub-fases da implantação:

Teste de Sistema

Objetivo
- confrontar o sistema produto com as especificações;
- preparação dos arquivos operacionais;
- revisão da documentação;
- treinamento dos usuários.

Produtos
- manual do usuário revisto;
- manual do operador revisto;
- manual do programador revisto;
- listagens de testes e comprovação da suficiência destes;
- diretório de dados revisto;
- arquivos contendo dados para teste de sistema para utilização futura;
- arquivos operacionais criados e corretos (resultados da operação do sistema de conversão);
- sub-sistema de segurança operacional aprovado e em operação normal:
- treinamento dos usuários concluido.

Aprovação do Teste de Sistema

Objetivo
- revisão da análise de rentabilidade;
- comprovação do teste de sistema:
- aprovação dos produtos da fase de teste de sistema.

Produtos
- documento de aprovação do teste de sistema;
- arquivos operacionais aprovados e em manutenção.

Teste de Aceitação

Objetivo
- operação experimental pelo usuário utilizando o sistema ainda instrumentado. O sistema possivelmente ainda reside no equipamento, ou instalação, utilizados para o desenvolvimento. Estes não necessariamente são os que serão utilizados durante a operação;
- manutenção dos arquivos operacionais;
- forma final da documentação.

Produtos
- avaliação do sistema pelo usuário;
- manuais do operador, do programador e do usuário em forma final;
- porção do diretório de dados relativo a este sistema em forma final. O diretório de dados é um documento que aplica a todos os sistemas da instituição. Quando da produção do diretório, algumas inconsistências com relação a outros sistemas podem ter sido introduzidas. Neste ponto, no entanto, estes defeitos já foram todos sanados e o diretório obedece às normas estabelecidas;
- documentação de desenvolvimento e manutenção em forma final. Os manuais de sistema, manutenção etc., estão embutidos na documentação produzida durante o desenvolvimento;
- banco de dados (arquivos) operacional mantido e em

atividade;
- usuário capacitado a utilizar o sistema independentemente da presença de membros das outras entidades.

Aprovação do Teste de Aceitação

Objetivo
- revisão da análise de rentabilidade;
- aprovação final da documentação;
- aprovação do sistema produto.

Produtos
- documento de aprovação da documentação;
- documento de aprovação do sistema produto;
- aprovação do conteúdo do banco de dados operacional.

Teste no Local

Objetivo
- operação experimental do sistema novo em paralelo com o sistema antigo utilizando equipamentos e instalações em que residirá o sistema durante a fase de operação;
- avaliação do sistema e do desenvolvimento do sistema.

Produtos
- resultados de operação normais;
- avaliação do sistema pelo usuário;
- resumo da história do desenvolvimento do sistema;
- arquivamento dos produtos intermediários.

Aprovação da Operação no Local

Objetivo
- descontinuação do sistema antigo.

Produtos
- documento autorizando a descontinuação do sistema antigo;
- documento de aprovação da avaliação do sistema;
- documento de aprovação da história do desenvolvimento do sistema.

Neste instante o sistema encontra-se em operação normal. Possivelmente é previsto um período de garantia durante o qual o desenvolvedor deverá estar disponível para corrigir quaisquer erros e dificuldades porventura encontrados.

4.6 Fase de Operação.

Durante esta fase o sistema é operado e mantido normalmente. A manutenção e a operação correm por conta dos grupos produtor e manutenedor da entidade usuário. Possivelmente existirá um período inicial durante o qual o desenvolvedor prestará auxílios em caso de falhas, erros ou dificuldades de operação.

Durante a fase de operação serão percebidas também novas necessidades que poderiam ser satisfeitas através de extensões ao sistema. Como já foi dito, estas extensões devem ser encaradas como sendo projetos especiais, devendo ser examinadas da mesma forma como se um novo sistema estivesse sendo proposto.

5 Epílogo.

Este texto tinha por objetivo apresentar uma visão abrangente do problema "desenvolvimento de sistemas automatizados". Procuramos ater-nos a este objetivo sem entrar em detalhes demasiados. Isto pode ter dado, por vezes, a impressão de superficialidade. Apesar de existir este risco, o nível de abordagem é justificado uma vez que visa ao estudo do problema como um todo, evitando soluções de problemas relativos a detalhes das fases de desenvolvimento, o que poderia levar à perda de visão do conjunto.

Certamente remanesce um número elevado de perguntas carecendo de respostas. Estas perguntas podem ser respondidas através de estudos detalhados dos problemas relativos a cada fase ou classe de atividades. Ou seja, deve ter ficado evidente que ainda existe muito por fazer para que consigamos tornar o desenvolvimento de sistemas automatizados realmente em uma indústria com padrões de qualidade, utilidade e rentabilidade adequados. Deve ter ficado claro, também, que necessitamos para isto do desenvolvimento de ferramentas poderosas e econômicas, bem como do desenvolvimento de padrões e métodos industriais que comprovadamente contribuam para a melhoria destes índices.

Agradecimentos.

Agradeço aos senhores Eduardo Taquece Moura, Fernando Malburg da Silveira, José Antonio Fabiano Mendes, Oscar Marques Cardim, Osmar Boavista da Cunha Júnior, que, instados a revisarem as primeiras versões deste texto, contribuiram com inúmeras sugestões e correções, sem as quais este texto certamente seria ainda mais incompleto e enigmático.

Referências Bibliográficas.

Ao referenciarmos a bibliografia, procuramos indicar as referências mais recentes de nosso conhecimento, com o intuito de facilitar a pesquisa bibliográfica no assunto. Por esta razão, em alguns casos não será pago o devido tributo às pessoas que lançaram novas idéias.

[Allen78] Allen, J.; Lientz, B. P.
Systems in Action, a Managerial and Social Approach; Goodyear Publishing Co., Inc., Santa Monica, Ca, EEUU; 1978

[Asteggiano78] Asteggiano, S. L.
Análise de Sistemas: Uma Abordagem Sistêmica; Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; Fev 1978.

[Benjamin71] Benjamin, R. I.
Control of the Information System Development Cycle; Wiley Interscience, 1971.

[Boehm73] Boehm, B. W.
"Software and its Impact: A Quantitative Assessment"; Datamation 19(5), maio 1973; pags 48-59

[Boehm75] Boehm, B. W.
"The High Cost of Software"; in [Horowitz75] ed.; 1975; pags 3-14

[Boehm76] Boehm, B. W.
'Software Engineering'; IEEE Transactions On Computers C-25(12); dez 1976; pags 1126-1241

[Boehm78] Boehm, B. W.; Brown, J. R.; Kaspar, H. et alii
Characteristics of Software Quality; North Holland Publishing Co.; 1978

[Boehm74] Boehm, B. W.; Gaynor, E. eds.
Proceedings of a Symposium on Reliable, Cost-Effective, Secure Software; TRW Systems Group; Ca, EEUU; 1974

[Brooks75] Brooks, F. P.
The Mythical Man-Month; Addison Wesley Publishing Co.; 1975

[Brown74] Brown, P. J.
"Programming and Documenting Software Projects"; ACM Computing Surveys 6(4), dez 1974; pags 213-220

[Burch74] Burch, J. G.; Strater, F. R.
Information Systems: Theory and Practice; Hamilton Publishing Co., Santa Barbara, California, EEUU; 1974

[Bussey78] Bussey, L. E.
The Economic Analysis of Industrial Projects; Prentice Hall, Englewood Cliffs, NJ, EEUU; 1978

[Furtado78] Furtado, A. L.; Staa, A. v.
Diretório de Dados: Utilidade e Especificação; Monografias em Ciência da Computação no. 11/78, Departamento de Informática, Pontifícia Universidade Católica; Rio de Janeiro; jul 1978

[Gilb77] Gilb, T.
Software Metrics; Whintrop Publishing Inc., Cambridge, Mass., EEUU; 1977

[Gildersleeve74] Gildersleeve, T. R.
Data Processing Project Management; van Nostrand Reinhold Co.; 1974

[Gildersleeve78] Gildersleeve, T. R.
Succesful Data Processing System Analysis; Prentice Hall, Englewood Cliffs, NJ, EEUU; 1978

[Gottlieb78] Gottlieb, C. C.
Software Profitability; Seminário apresentado ao Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; nov 1978

[Hice74] Hice, G. F.; Turner, W. S.; Cashwell, L. F.
Systems Development Methodology; North Holland/American Elsevier; 1974

[Horowitz75] Horowitz, E. ed.
Practical Strategies for Developing Large Software Systems; Addison Wesley Publishing Company; 1975

[IBM74]
HIPO - A Design and Documentation Technique; IBM form GC20-1851; IBM, New York, EEUU; 1974

[IBM75]
Business System Planning, Information Systems Planning Guide; IBM, New York, EEUU; 1975

[Langefors73] Langefors, B.
Theoretical Analysis of Information Systems; Studentlitteratur, Lund, Suécia; 1973

[Leite79] Leite, J. C. S. P.
Contabilidade de Custos no Desenvolvimento de Software; Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; Mar 1979

[Lucena78] Lucena, C. J. P.; v. Staa, A.
Projeto de Programas; SERPRO, Série Cadernos de Programação, no. 1; Rio de Janeiro, RJ; 1978

[Magalhães79] Magalhães, J. A. C.
Técnicas para o Projeto e a Implementação de Sistemas de Software Confiáveis; Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; mar 1979

[Mangold74] Mangold, E. R.
"Software Visibility and Management"; in [Boehm74] ed.; 1974; pags 2.1-2.25

[Mantley74] Mantley, F. C.
"Software Development Tools"; in [Boehm74] ed.; 1974; pags 3.1-3.58

[Metzger73] Metzger, P. W.
Managing a Programming Project: Prentice Hall, Englewood Cliffs, NJ, EEUU; 1973

[Moura79] Moura, E. T.
Manutenção de Software; Dissertação de Mestrado; Departamento de Informática; Pontifícia Universidade Católica; Rio de Janeiro; jul 1979

[Myers75] Myers, G. J.
Reliable Software Through Composite Design; Petrocelli/Charter, New York, NY, EEUU; 1975

[Parente78] Parente, F. A. F.
Auditoria de Sistemas Automatizados; Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; dez 1978

[Putnam77a] Putnam, L. H.
"The Software Life Cycle: Evidence and Foundation"; in [Putnam77b] ed.; 1977; pags 1-112

[Putnam77b] Putnam, L. H.; Wolverton, R. W. eds.
Quantitative Management. Software Cost Estimating; Computer Software and Applications Software Tutorial (COMPSAC77); IEEE, catalog no. EHO 129-7; nov 1977

[Queiroz78] Queiroz, J. D.
Metodologia para a Elaboração de Proposta de Desenvolvimento de Sistemas de Informaçao Automatizados; Dissertaçao de Mestrado; Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; mar 1978

[Rocha78] Rocha, A. R. C.
Plano de Desenvolvimento de Projetos de Sistemas Automatizados; Dissertaçao de Mestrado, Departamento de In-

formática, Pontifícia Universidade Católica, Rio de Janeiro; dez 1978

[Ross77] Ross, D. T.
"Structured Analysis (SA): A Language for Communicating Ideas"; IEEE Transactions on Software Engineering SE-3(1), jan 1977; pags 16-34

[Royce75] Royce, W. W.
"Software Requirement Analysis: Sizing and Costing"; in [Horowitz75] ed.; 1975; pags 57-71

[Silva77] Silva, E. O.
Controle de Acesso e Uso das Informações de um Banco de Dados; Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; fev 1977

[Staa79] Staa, A. v.
Proposta de Desenvolvimento, o Primeiro Passo no Desenvolvimento de Sistemas Automatizados; Monografias em Ciência da Computação 79; Departamento de Informática, Pontifícia Universidade Católica, Rio de Janeiro; 1979, em preparação

[Tausworthe77] Tausworthe, R. C.
Standardized Development of Computer Software; Prentice Hall, Englewood Cliffs, NJ, EEUU; 1977

[Teichroew77] Teichroew, D.; Hershey, E. A.
"PSL/PSA: A Computer Aided Technique for Structured Documentation and Analysis of Information Processing Systems"; IEEE Transactions on Software Engineering SF-3(1), jan 1977; pags 41-49

[Wegner78] Wegner, P.
"Research Directions in Software Technology"; a ser publicado, MIT Press; 1978

[Weinberg71] Weinberg, G. M.
The Psychology of Computer Programming; van Nostrand Reinhold Co.; 1971

[Weinberg75] Weinberg, G. M.
An Introduction to General Systems Thinking; John Wiley and Sons; 1975

[Wolverton74] Wolverton, R. W.
"The Cost of Developing Large Scale Software"; IEEE Transactions on Computers jun 1974;

[Wolverton77] Wolverton, R. W.
"Management by Objectives"; em [Putnam77b] ed.; 1977; pags 239-256

[Yeh77] Yeh, R. T. ed.
Current Trends in Programming Methodology volumes 1 a 4; Prentice Hall, Englewood Cliffs, New Jersey, EEUU; 1977

[Yohe74] Yohe, J. M.
"An Overview of Programming Practices"; ACM Computing Surveys 6(4); dez 1974; pags 221-246

[Yourdon75] Yourdon, E.
Techniques of Program Structure and Design; Prentice Hall, Englewood Cliffs, New Jersey, EEUU; 1975

[Zelkowitz78] Zelkowitz, M. V.
"Perspectives on Software Engineering"; ACM Computing Surveys 10(2); jun 1978; pags 197-216