SALESIANO
Opção pelos Jovens

Faculdade de Tecnologia
da Alta Noroeste

CURSO DE TECNÓLOGO EM PROCESSAMENTO DE DADOS

Análise e Projeto de Sistemas II

Prof. Sérgio Luiz Tonsig

2000

FACULDADE DE TECNOLOGIA DA ALTA NOROESTE

Curso de Tecnólogo em Processamento de Dados

Disciplina: Análise e Projeto de Sistemas II

Prof.: Sérgio Luiz Tonsig*

Ementa:

Desenvolver a especificação de sistemas, através de estudos de casos, propiciar uma visão geral sobre as etapas que devem ser seguidas para o efetivo desenvolvimento de um projeto e seus aspectos gerenciais.

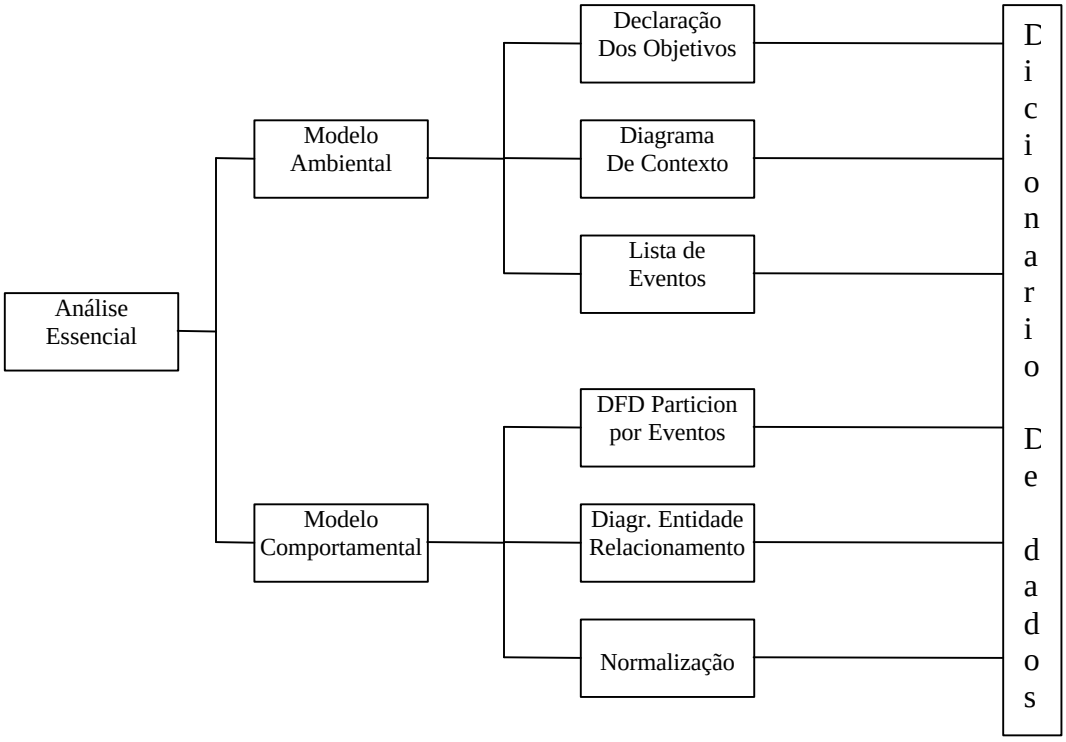
- *Docente da Faculdade de Tecnologia da Alta Noroeste. Especialista em Sistemas de Informação pela Universidade Federal de São Carlos. Mestrando em Gerência de Sistemas de Informação de PUC Campinas.*

Sumário

1. O Caminho do Desenvolvimento de Sistemas

Antes de se enfatizar qualquer aspecto relativo a *Projeto de Sistema,* especialmente em nossa etapa atual de formação profissional, é fundamental resgatar, através de uma síntese (visto que foi objeto de estudo do semestre anterior), o caminho que um Analista de Sistemas deverá seguir, ao utilizar o método da análise essencial.

1.1. A Análise Essencial



A premissa básica para o método de análise essencial é descrever o sistema de maneira independente de restrições tecnológicas.

Isto implica dizer que devemos considerar na confecção do modelo essencial a existência de uma *tecnologia perfeita*.

Devemos entender este aspecto como uma abstração em que se supõe uma tecnologia ideal, sem limitações, onde:

a) Os custos, consumo e desgaste dos equipamentos são zero
b) A capacidade de armazenamento de dados do sistema é infinita
c) A velocidade dos processadores é infinita
d) O tempo de acesso a dados é instantâneo
e) Zero Erro (não ocorrem falhas)

Através deste método faz-se um exame do domínio do problema (levantamento de dados acerca do sistema que será feito) inicialmente enfocando os aspectos mais essenciais pertinentes ao problema. A análise essencial, é constituída basicamente por duas fases ou modelos: Ambiental e o Comportamental.

Modelo Ambiental

No modelo ambiental tem-se a especificação macro do sistema (como se fosse uma caixa preta) inserido em um meio ambiente, busca-se representar a relação do sistema com o meio onde ele está, através de estímulos provenientes do meio e respostas a estes estímulos, que poderão ser internas ao sistema ou ainda serem devolvidas para o meio ambiente. Três grandes atividades são elaboradas neste modelo: Declaração dos Objetivos do Sistema, a elaboração do Diagrama de Contexto e a especificação da lista de eventos.

Declaração de Objetivos

Trata-se da especificação daquilo que o sistema deverá fazer, frente aos problemas existentes na organização para o qual ele será feito. Deve também, tanto quanto possível, refletir os desejos do usuário no que diz respeito as solicitações que ele tenha apresentado como alternativas de solução dos problemas.
Naturalmente antes da elaboração dos objetivos do sistema, o Analista deverá ter efetuado um minucioso levantamento de dados (checando inclusive requisitos do sistema), conhecendo profundamente o chamado domínio do problema. Se o sistema for referente a controle de uma biblioteca, o Analista precisa saber tudo sobre bibliotecas, regras gerais, linguagem utilizada, detalhes e exceções.

Diagrama de Contexto

Após a especificação formal dos objetivos do sistema, o Analista já estará em condições mais apropriadas para elaborar o diagrama de contexto. Ele reflete graficamente a relação do sistema com o meio ambiente onde está inserido. Esta relação dá-se através do recebimento de estímulos do meio ambiente, os quais ativam processos, e estes, por sua vez geram respostas, que podem vir a ser respostas externas ao sistema, ou seja, resposta ao meio ambiente.

Lista de Eventos

Trata-se da especificação das atividades (processos) essenciais que o sistema terá . Elas são ativadas por estímulos (fluxo de dados, fluxo temporal ou de controle), fazem algum processamento e geram respostas. Não há uma precedência estabelecida para a elaboração da lista de eventos e o diagrama de contexto, são atividade que podem estar acontecendo paralelamente inclusive.

Modelo Comportamental

O modelo comportamental é a fase em que o Analista passa a olhar para dentro do sistema. Irá detalhar como cada atividade existente na lista de eventos se comporta (como ela deve funcionar). Também fará um modelo de dados sobre o qual o sistema atuará, observando critérios para conseguir boa performance na sua utilização (através da normalização de dados). Acompanhando mais efetivamente este modelo (muito embora já possa existir antes dele) cria-se o dicionário de dados. Também nesta fase elabora-se o D.F.D. Hierárquico, que nada mais é do que o agrupamento de atividades essenciais afins (síncronas), que enfocam determinado aspecto no sistema.

Diagrama de Fluxo de Dados Particionado por Evento

Para cada item da lista de eventos o Analista de Sistemas fará um Diagrama de Fluxo de Dados, representando de forma gráfica, individualmente, cada evento existente no sistema. Desta forma, haverá tantos diagramas de fluxo de dados particionado por eventos, quantos forem os itens existentes na lista de eventos.

Diagrama Entidade Relacionamento

Para modelagem de dados, o Analista de Sistemas fará inicialmente o D.E.R.
Com este diagrama terá um poderoso instrumento para mapear como os dados estão organizados e como eles se relacionam. Em cima deste modelo, pode ser aplicado a teoria da Normalização, que permitirá extrair melhor performance quando da utilização da estrutura dos dados.

Diagrama Hierárquico de Macro Atividades

Trata-se de um D.F.D. onde se propiciará uma visão de Macro Atividades. Pega-se os diagramas de fluxo de dados particionados por eventos e verifica-se quais são aquelas atividades afins (síncronas – que tratam determinado assunto). Estes processos são aglutinados em um único, de tal forma que se obterá uma visão mais sintética da representação do sistema, cuja finalidade é, além da documentacional a possibilidade de examinar e definir interfaces e locais de processamento. A fim de facilitar a construção do DFD Hierárquico (através de uma visão mais global do sistema), pode-se antes elaborar o chamado Diagrama Preliminar, que consiste em pegar todos os DFDs particionados por evento e torná-lo um só (visão única de um DFD com todos os processos existentes).

Dicionário de Dados

Todos os dados referenciados na construção do sistema, deve ter sua definição no dicionário de dados.
Para a construção do dicionário existem alguns padrões, dos quais, vamos adotar o segue:

Símbolo	**Significado**
=	**É composto de**
+	**E**
( )	**Opcional (pode estar presente ou ausente)**
{ }	**Iteração (Repetição)**
[ ]	**Escolha uma das opções**
*** ***	**Comentário**
@	**Indica campo chave**
| ou /	**Separa Alternativas na construção [ ]**

1.1.3. Um Estudo de Caso

Domínio do Problema

A primeira grande missão do Analista de Sistema, ao ser chamado a desenvolver um sistema, é o de delimitar a área fronteiriça deste sistema a ser desenvolvido. Exatamente o que se deseja e até onde deverá ser feito tal coisa. Para iniciar este processo de identificação uma boa dica é começar pela pergunta fundamental: *Por que tal sistema deveria existir ?* – naturalmente quem contratou seu serviço está apto a responder este questionamento com uma riqueza de detalhes muito grande.

Em nosso estudo de caso, tem-se um sistema hoteleiro. Nesta fase de exame do domínio do problema, estabeleceu-se que o objetivo é apenas o controle da disponibilidade de quartos do hotel (não envolve qualquer outro aspecto, tais como: controle financeiro, contábil, etc...).

Uma vez claramente estabelecido o domínio do problema depois de um intenso levantamento de dados (*que começou pela pergunta acima: por que este sistema deveria existir ?*), o Analista de Sistema deve se concentrar unicamente no domínio do problema, eventuais interfaces e nos requisitos necessários ao desenvolvimento do sistema, passando para a etapa seguinte.

Rigoroso Levantamento de Dados

Após um rigoroso levantamento de dados, que pode envolver, exame e cópia de documentos originais, acompanhamento do fluxo de trabalho, entrevistas com usuários e até mesmo estágio no local objeto do levantamento de dados. O Analista de Sistemas deve sair desta fase conhecendo todos os detalhes do negócio, suas regras, exceções, linguagem empregada e eventuais particularidades da organização sobre o negócio.

Se o Analista ignorar esta fase ou desenvolve-la superficialmente, assumindo que já possui conhecimentos suficientes sobre o assunto, provavelmente, estabelecerá requisitos confusos ou inválidos, fracassando já no início de seu sistema.

Em nosso estudo de caso, após estes procedimentos iniciais, chegou-se a um conhecimento acerca do que deveria ser feito. As informações colhidas encontram-se no enunciado que segue:

Estudo de Caso - Sistema Hoteleiro - Controle de Disponibilidades de Quartos

Deseja-se desenvolver um sistema para um pequeno hotel que atenda aos seguintes requisitos:

- Quando o cliente telefona ou vem até o hotel e pede para reservar um quarto, o funcionário verifica se existe quarto disponível no período solicitado. Caso afirmativo, é feita a reserva do quarto. Caso negativo, é informado ao cliente a não disponibilidade do quarto.

- Quando o cliente não mais desejar o quarto reservado o funcionário providencia o cancelamento da reserva, disponibilizando novamente o quarto.

- Quando o cliente não comparecer ao hotel para hospedar-se até às 12:00 horas do dia da reserva, deve ser cancelado a sua reserva.

- Quando o cliente ocupar um quarto, reservado previamente, o funcionário faz o registro do cliente. Caso o quarto não esteja reservado uma mensagem de rejeição será emitida. Caso contrário, um pacote com informações úteis e a confirmação serão fornecidos ao cliente.

- Quando o cliente deixar o hotel, notificando sua saída, será fornecido a conta, e o quarto será disponibilizado para limpeza.

- O cliente pode pagar a conta à vista, ou usando cartão de crédito. Caso use cartão de crédito, é verificado sua situação para aceitar ou não o cartão.

- Quando o quarto estiver limpo, após uma ocupação, o gerente torna-o disponível para nova locação.

De posse destas informações, em nosso estudo de caso, segue um descritivo da análise do problema e a descrição da solução escolhida pelo Analista de Sistemas, diante, é claro, além das informações, os desejos manifestados pelo usuário (e possíveis de serem desenvolvidos).

Descritivo da Solução Escolhida

"Depois de estudar o volume de dados processado pelo sistema atual e considerando o tempo de resposta requerido, chego a conclusão que o sistema novo precisará de quatro pessoas e um pequeno computador comercial.

Deverá ser atribuído tanto trabalho quanto for possível ao computador de modo a poder minimizar os custos de processamento. Porém, não ser correrá o risco de perder clientes por força-lo a um contato direto com o computador (considerando a situação atual da interface homem/máquina); consequentemente, optou-se por alocar tudo ao computador, exceto as partes das atividades essenciais que envolvem um contato direto com os clientes do hotel.

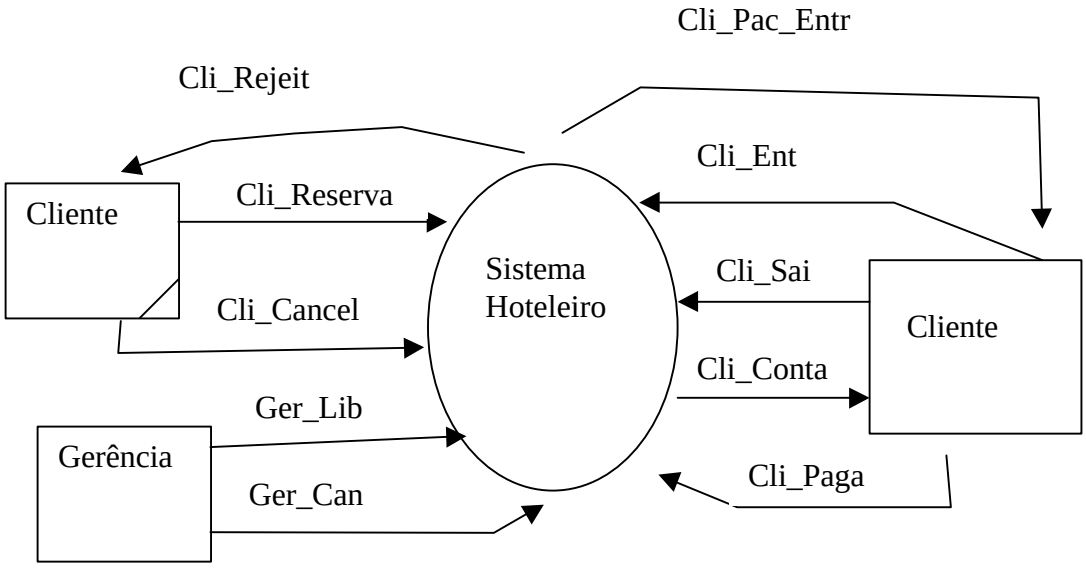
Uma atividade essencial, Cancelar não comparecimento, não requer nenhum contato direto com o cliente e, portanto, pode ser desempenhado diretamente pelo computador."

O que vem a seguir é a especificação técnica do projeto do sistema, começando inicialmente pelo modelo ambiental.

1.1.3.1. Declaração do Objetivo do Sistema

Controlar o serviço de reservas, registros e cobrança de quartos de um sistema hoteleiro.

1.1.3.2. Diagrama de Contexto



1.1.3.3. Lista de Eventos do Estudo de Caso

Nº	Evento	Descrição do Evento	Estímulo	Tipo Estímulo	Ação	Resposta
01	Cliente Reserva Quarto	Quando o cliente telefona ou vem até o hotel e pede para reservar um quarto, o funcionário executa um procedimento padrão	Cli_Reserva	F	Efetuar Reserva	Cli_Reservado ou Cli_Indisp
02	É hora de cancelar reserva	Quando o cliente não comparecer ao hotel para hospedar-se até as 12:00 horas do dia da reserva	-	T	Cancelar Reserva Automaticamente	Ger_Cancel
03	Cliente registra-se no hotel	Cliente faz o registro para a ocupação do quarto, reservado previamente. Caso não reservado uma mensagem de rejeição será emitido, caso contrário um pacote com informações será fornecido	Cli_Ent	F	Registrar Cliente	Cli_Pac_Ent ou Cli_Rej
04	Cliente solicita saída do hotel	Quando o cliente deixar o hotel este solicita que providencie o fechamento de sua conta, havendo a disponibilidade do quarto para limpeza	Cli_Sai	F	Fechar Quarto	Cli_Conta
05	Cliente paga a conta	Cliente paga a quantia correspondente ao aluguel do quarto e às despesas efetuadas durante sua estadia	Cli_Paga	F	Registrar Pagamento	Cli_Recibo
06	Cliente cancela a reserva	Quando o cliente não mais desejar o quarto reservado e comunicar o fato, será cancelado a reserva, disponibilizando o quarto novamente	Cli_Cancel	F	Cancelar Reserva por Solicitação	Cli_Canc
07	Gerência disponi-biliza quarto	Quando o quarto estiver limpo o gerente torna-o disponível	Ger_Lib	F	Liberar Quarto	Msg_01
08	Gerência cadastra quarto	Gerência inclui, exclui ou modifica dados do quarto	Ger_Cad	F	Manipular cadastro de quarto	Msg_02

Como você sabe, deve haver entre a lista de eventos e o diagrama de contexto uma consistência. É importante lembrar também que a Lista de Eventos ou Diagrama de Contexto podem ser desenvolvidos paralelamente.

Exercício:

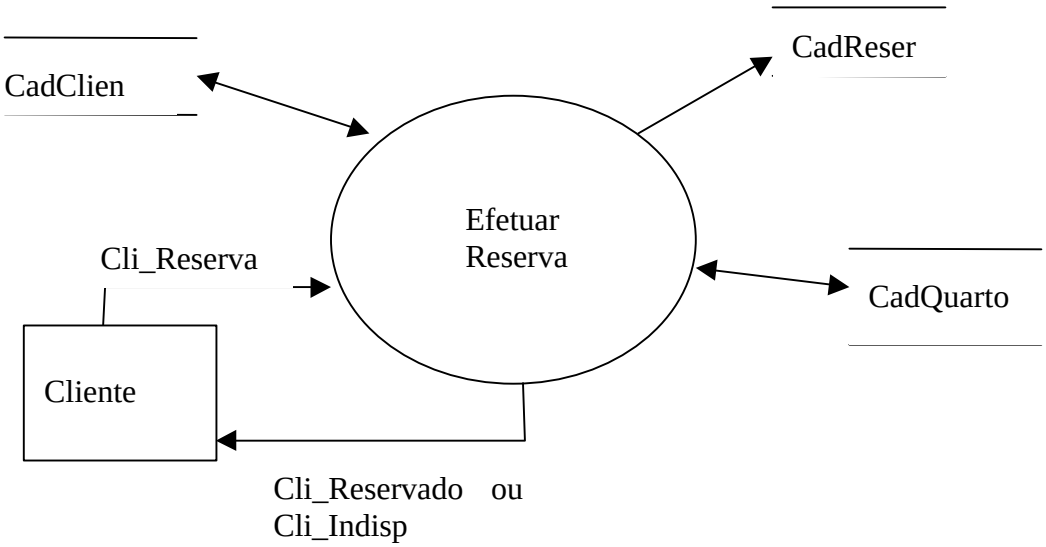
Cheque ambos (Lista de Eventos X Diagrama de Contexto) e acerte as possíveis inconsistências encontradas.
(eventuais alterações faça sobre o próprio manual, no local adequado, a fim de que você tenha o material completo para estudo)

1.1.3.4. DFD Particionado por Evento

Depois que a lista de eventos estiver *concluída* (é claro que as vezes você esquecerá um ou outro evento, que depois podem ser acrescentados, mas, com certeza os eventos essenciais estarão lá), você deverá fazer o Diagrama de Fluxo de Dados particionados por Eventos, também conhecido como Diagrama das Atividade Essenciais.

Em nosso estudo de caso, conforme nossa lista de eventos, deverá haver oito DFDs particionados por evento. Abaixo, como exemplo, segue o Diagrama correspondente ao primeiro evento da lista.

Evento 01 – Cliente Reserva Quarto



Exercício:

Faça o D.F.D. particionado por eventos para os demais itens da lista de eventos.
(antes porém, a seguir, dê uma olhada no modelo de dados completo que é utilizado neste estudo de caso – pág. 21)

1.1.3.5. A Especificação dos Processos

Como você observou no diagrama da página anterior (DFD particionado por evento – referente ao evento 01 da lista de eventos), tudo que se enxerga é que existe um processo que vai tratar determinado evento (Cliente reserva quarto), o qual mexe com determinados arquivos.

Tem-se portanto, claramente, ***o que*** acontece, porém, precisa-se saber, ***como*** tal coisa acontece. Veja que o diagrama não expressa, por exemplo, nenhuma regra do negócio, a qual, se não for documentada, o desenvolvedor fará um programa de reserva de quarto inválido.

Veja, neste hotel do nosso estudo de caso, um cliente só pode reservar um quarto se o mesmo estiver liberado, após ter passado por uma faxina (QuaSit = 0). Isto pode parecer óbvio, mas como expressar essa regra do negócio, a fim de que o programador, ao desenvolver o módulo que vai tratar da reserva, possa saber que isto deve ser checado ?

Outro exemplo de regra do negócio é:
O cliente que efetuar nova reserva no máximo 15 dias após sua última hospedagem, terá um desconto de 10 %.
Isto de alguma forma tem que ser documentado e o programador, ao fazer o processo de fechamento da conta deve aplicar a regra, caso contrário o sistema não será válido.

Portanto tem-se que fazer a *especificação funcional do módulo* (como ele deve funcionar), permitindo que exista uma documentação formal das regras do negócio, bem como alguém possa implementar isto no sistema.

Há apenas alguns anos, esta especificação era extremamente formal, cuidando de mínimos detalhes, basicamente, um programa era escrito para que depois um programador apenas codificasse. Como por exemplo, o pedaço de especificação mostrada abaixo:

Limpe a Tela
Abra o arquivo CadCli
MostreMensagem “Digite o C.G.C.:”
Aguarde a digitação
Consistir o C.G.C. conforme rotina CONF_CGC
.
.
.

Miniespecificações

Atualmente o enfoque é para que haja apenas uma *miniespecificação.* Uma miniespecificação vai tratar apenas das linhas genéricas do processo enfatizando os ***aspectos essenciais*** e aqueles que referem-se as ***regras do negócio***. De tal maneira que se não houver esta especificação um programador não fará satisfatoriamente o programa, ou seja, ele não será um programa válido, de acordo com a realidade que deve atender.

Portanto ninguém sairá ditando como fazer um programa para incluir, alterar, excluir ou consultar cadastros, salvo se houver detalhes de implementação proveniente das regras do negócio. E, neste caso, haverá um miniespecificação acerca daquele aspecto.

"O homem observa a montanha, atrás da esposa, com binóculo".

Quem estava com o binóculo ?

Quem estava atrás da esposa ?

Observe que em uma narrativa comum, pode haver uma série de ambigüidades. Para tentar eliminar interpretações diferenciadas de um texto, através do qual será expresso regras de um negócio, criou-se algumas ferramentas.

Ferramentas para Miniespecificações

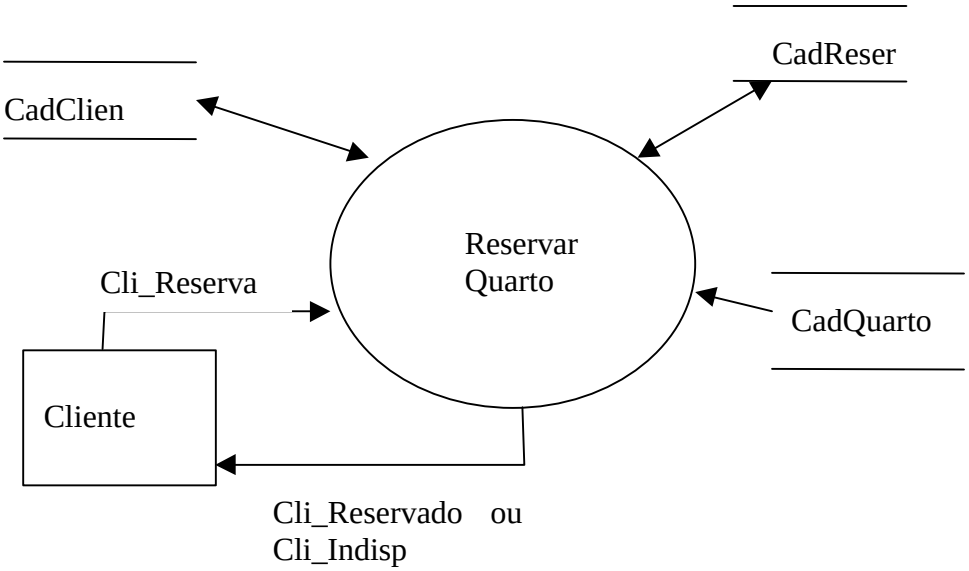
Português Estruturado

É um texto em português, desenvolvido dentro de uma hierarquia. Pode-se empregar palavras chaves para delimitar esta hierarquia, bem como simultaneamente, indentar (margens diferenciadas) o texto.

Admite uma certa liberdade de expressão, com relação a tamanho e conteúdo do texto, não há critérios rigorosamente estabelecidos. É desejável, porém, que se possa transmitir algo sem ambigüidade de uma maneira bem sintética.

Exemplo de português estruturado:

DFD Particionado Por Evento - Evento 01 – Cliente Reserva Quarto



Miniespecificação

PEGAR dados Cli_Reserva
SE cliente inexistente, faça seu cadastro (para campo Cli_Sit jogar 0)
PARA reservar quarto, faça:
    MOSTRAR lista de quartos livres para reserva (todos que tiverem Qua_Sit = 0)
    SE não existir pelo menos um quarto livre, faça:
        Mensagem “Desculpe, Hotel totalmente ocupado.”
        Encerrar
    FIM SE
    QUANDO escolhido um quarto livre, faça:
        MARCAR quarto como reservado (jogar 1 para campo Qua_Sit)
        MARCAR cliente como solicitado reserva (jogar 1 para Cli_Sit)
    FIM QUANDO
FIM PARA

Pseudocódigo

Basicamente, pseudocódigo e português estruturado não trazem grande diferença, contudo é possível distinguir entre ambos (algumas vezes os termos são usados indistintamente como sinônimos).

O pseudocódigo usa notação mais formal, mais orientada ao profissional de processamento de dados, empregando as palavras chaves e tentando seguir uma sintaxe bem mais próxima de uma linguagem de programação.

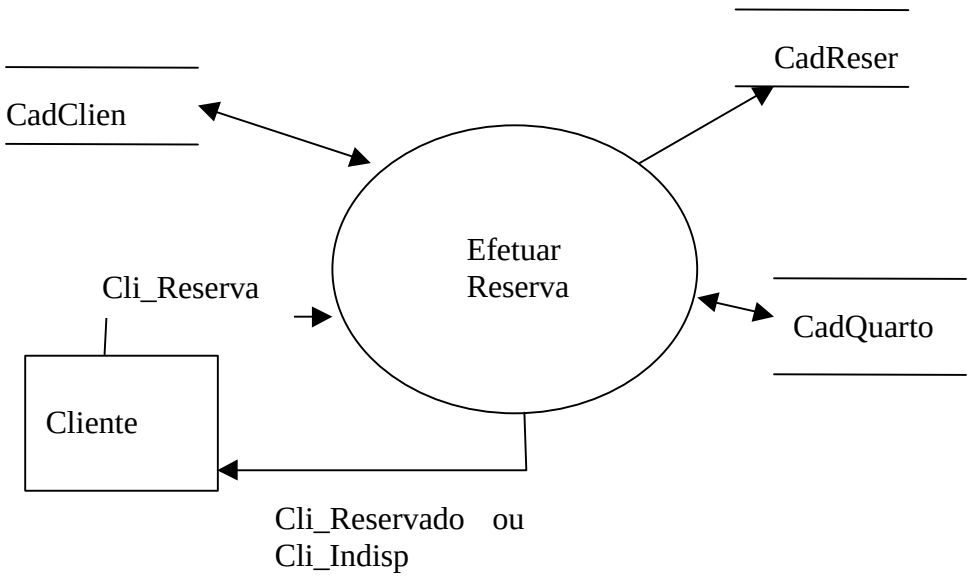
Usa-se palavras-chaves para realçar o que seriam comandos utilizadas nas linguagens de programação; por exemplo: SE, SENÃO, ENTÃO, FIMSE, REPETIR, ENQUANTO, REPETIRATÉ, FIMREPETIR, ENCERRAR.

Com relação as expressões lógicas, emprega-se a mesma gama de signos: >, <, =, <=, >=, E, OU.

A grande desvantagem em se utilizar pseudocódigo, no lugar do português estruturado, é que neste, se possibilita no próprio texto uma explicação mais detalhada acerca do processo que está sendo especificado. No pseudocódigo, também é possível agregar explicação, porém deve seguir como se fosse um comentário em uma linguagem de programação, devendo ficar entre /* */.

```
/*  um comentário qualquer em pseudocódigo  */
```

Vamos examinar a seguir, como fica o exemplo feito em português estruturado referente ao evento essencial 01, agora refeito em pseudocódigo



Miniespecificação

```
PEGAR Cli_Reserva
SE cliente = "inexistente" ENTÃO EXECUTAR Cadastro_Cliente
SE reservar = VERDADEIRO ENTÃO:
        EXECUTAR Lista_Quartos_Livres
        SE quarto = "inexistente" ENTÃO:
                Mensagem "Desculpe, Hotel totalmente ocupado."
                ENCERRAR
        FIM SE
        SE quarto = "escolhido" ENTÃO:
                 Qua_Sit = 1.
                 Cli_Sit   = 1.
        FIM SE
FIM SE
```

Ferramentas Auxiliares de Miniespecificações

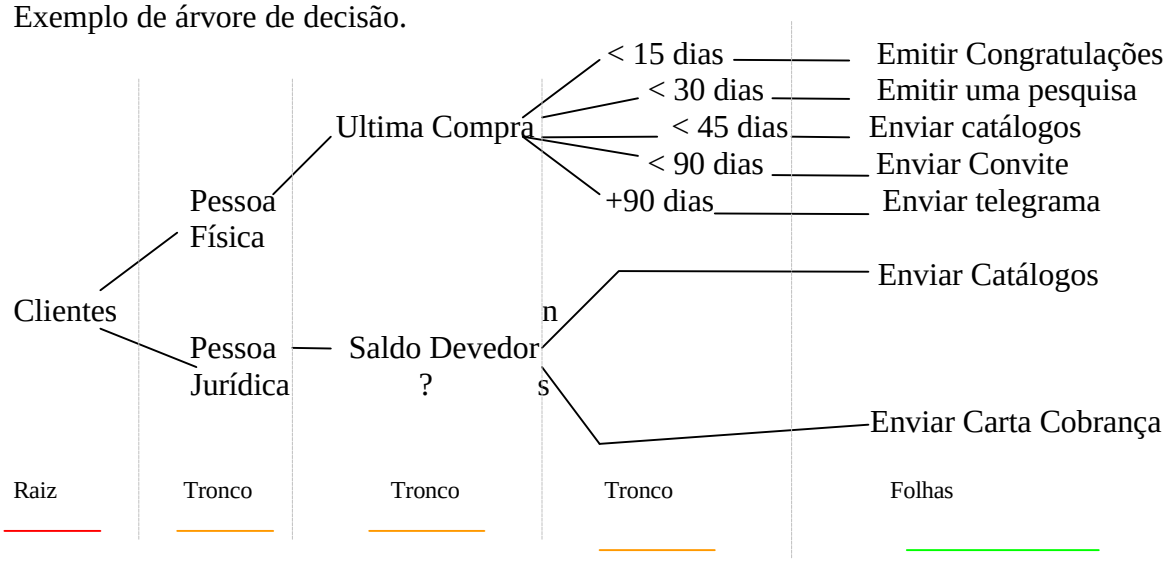
Duas outras técnicas de especificação podem ser utilizadas para expressar situações de lógica complexa ou não.

Árvores e Tabelas de Decisão

Uma árvore de decisão é um modelo de uma função na qual é determinado o valor de uma variável, com base neste valor, alguma ação é executada.

A ação pode ser para a escolha de outra variável a ter seu valor calculado ou a saída da função. Então, cada ação executada depende do valor atual da variável que está sendo testada e de todas as ações anteriores que foram executadas. Em uma árvore de decisão definida formalmente, uma variável só é testada uma vez em qualquer caminho através da árvore. Esta restrição é feita para evitar testes redundantes.

Exemplo de árvore de decisão.



Exercício – Desenvolva uma árvore de decisão para o enunciado que segue.

"Se cliente for maior de 18 anos, pode fazer assinatura da revista. Os menores de 18, apenas com uma declaração dos pais. A assinatura pode ser feita para 6 meses, 9 meses, 12 meses ou 24 meses. Caso seja paga com cartão, terá um desconto de 2% (6 meses), 3% (9meses), 5%(12 meses) e 10% (para 24 meses)."

Árvores de decisão *não são* indicadas para expressar lógica complexa, neste caso, é preferível as tabelas de decisões.
Uma árvore começa sempre em uma raiz, a partir da qual, sempre terá no mínimo dois troncos, que darão origem a outros troncos ou folhas. Em cada tronco, pode surgir no mínimo dois e no máximo n outro troncos. Um tronco de um galho não pode ligar-se a outro tronco em outro galho. Uma árvore sempre terminará em suas folhas.

Tabelas de Decisão

Tem a mesma finalidade das árvores de decisão, porém abrem mão do aspecto gráfico e proporcionam uma forma tabular. Pode-se por elas construir lógicas complexas. As tabelas de decisão possuem os seguintes componentes:

Condições /Ações \ Regras	01	02	03	04	05	06	07	...	n
Condicão 1	S	-							
Condicão 2	N	S							
Condicão n	S	N							
Acão 1	X	X							
Acão n	-	X							

Tem-se três elementos distintos na tabela de decisão: As condições, as regras de decisão e as ações.

A leitura da tabela de decisão deve ocorrer a partir da regra de decisão nº 1. Deve-se ler no sentido vertical.
No exemplo acima, lê-se a condição 1. Se a resposta for S, passar para a condição seguinte. Se a resposta não for S, pula-se para a regra de decisão nº 2 (e inicia-se da mesma forma).

A cada condição existente, verifica-se se a resposta obtida é aquela assinalada, se afirmativo, seguir para a condição seguinte. Se esta regra for totalmente satisfeita, ou seja, as respostas dadas as condições sejam de acordo com o que foi assinalado (s, n), checa-se os cursos de ação. Cada ação que estiver assinalada com X, será executada.

Uma vez satisfeita totalmente uma regra de decisão, a aplicação da tabela se encerra. Se pelo menos uma condição da regra não for satisfeita, pula-se para a próxima regra de decisão e assim sucessivamente.

Veja na próxima página um exemplo de tabela de decisão.

Condições e ações	Regras de Decisão 01	02	03	04	05	06	07	08	09	10	11	12
Cliente é Pessoa Física ?	S	S	S	S	S	N	N					
Ultima Compra foi feita a menos que 15 dias ?	S	N	N	N	N	-	-					
Ultima Compra foi feita a menos que 30 dias ?	-	S	N	N	N	-	-					
Ultima Compra foi feita a menos que 45 dias ?	-	-	S	N	N	-	-					
Ultima Compra foi feita a menos que 90 dias ?	-	-	-	S	N	-	-					
Ultima Compra foi feita a mais de 90 dias ?	-	-	-	-	S	-	-					
Saldo Devedor ?	-	-	-	-	-	N	S					
Emitir Congratulações	X											
Emitir uma pesquisa		X										
Enviar Catálogo			X			X						
Enviar Convite				X								
Enviar Telegrama					X							
Enviar carta cobrança							X					

Exercício

Para o mesmo enunciado do exercício existente na página 17, construa uma tabela de decisão.

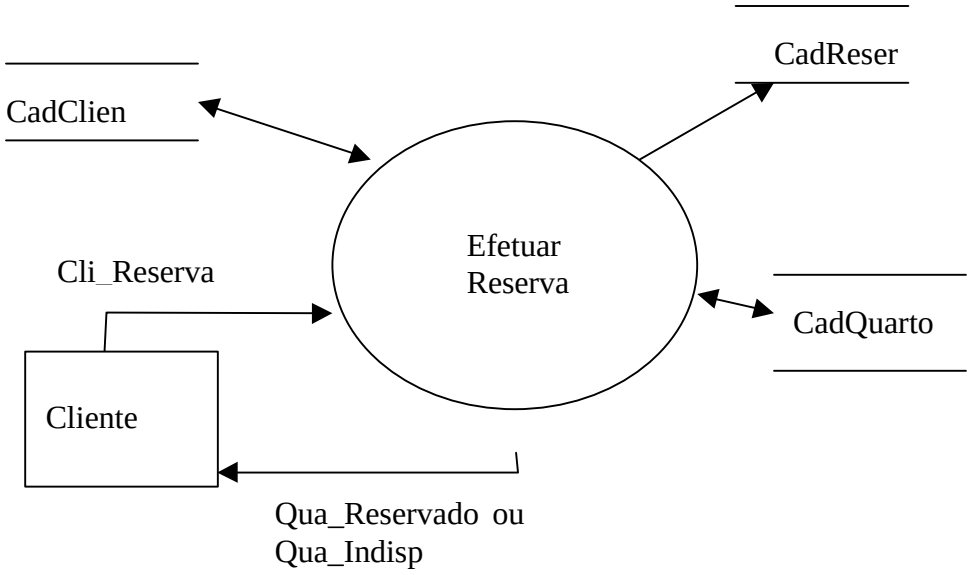
Miniespecificação de Processos
Orientação e Padronização para aplicação no T.C.C.

Uma miniespecificação vai tratar apenas das linhas genéricas do processo, enfatizando os ***aspectos essenciais*** e aqueles que referem-se as ***regras do negócio***, de tal maneira que, se não houver esta especificação, um programador não fará satisfatoriamente o programa, ou seja, ele não será um programa válido, de acordo com a realidade que deve atender.

Portanto ninguém sairá ditando detalhadamente como fazer um programa para incluir, alterar, excluir ou consultar cadastros, deverá ser apresentado apenas a síntese do objetivo global do processo, salvo se houver detalhes de implementação proveniente das regras do negócio, neste caso, haverá um miniespecificação um pouco mais ampla.

Para qualquer miniespecificação sugere-se a utilização de uma ferramenta que tente evitar ao máximo ambigüidades de interpretação, com relação as idéias que estão sendo expressas. Neste sentido emprega-se: Português Estruturado, Pseudocódigo, Tabelas de Decisão ou Árvores de Decisão.

O método da análise essencial indica estas ferramentas para a miniespecificação, porém, deixa em aberto ***o formato*** de apresentação da miniespecificação. Muitos autores, influenciados pela fase estruturalista da análise, normalmente exemplificam uma miniespecificação de forma estruturada, conforme exemplo a seguir:



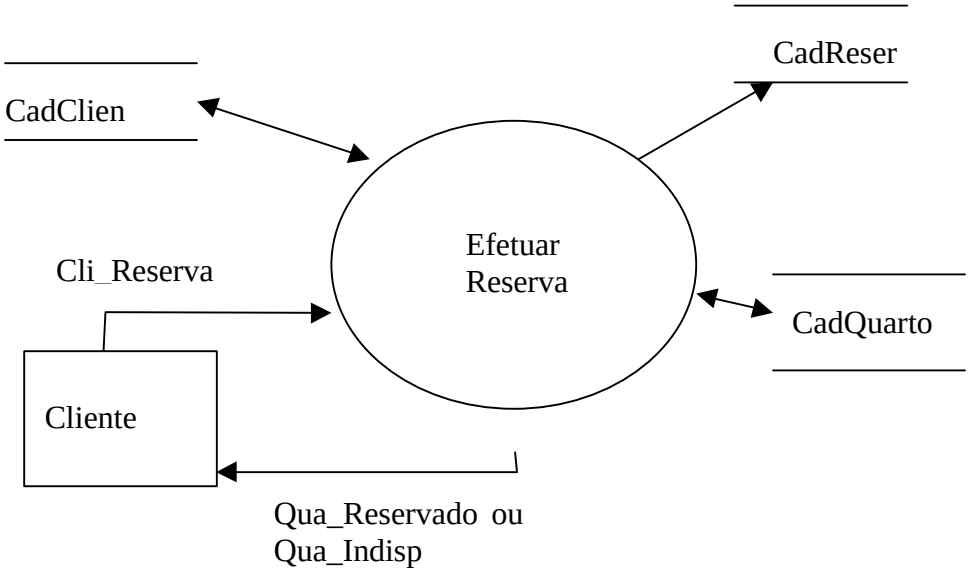

```
PEGAR Cli_Reserva
SE cliente = "inexistente" ENTÃO EXECUTAR Cadastro_Cliente
SE reservar = VERDADEIRO ENTÃO:
        MOSTRAR Lista_Quartos para escolha de um
        SE quartos escolhidos = "ocupados" ENTÃO:
                Mensagem "Desculpe, Hotel totalmente ocupado." (Qua_Indisp)
                ENCERRAR
        FIM SE
        SE quarto = "livre" ENTÃO:
                 Qua_Sit = 1.    /* marcar quarto como ocupado */
                 Cli_Sit   = 1.   /* marcar cliente com reserva feita */
                Mensagem "Reserva Concluída." (Qua_Reservado)
                FIM SE
FIM SE
```

Verifica-se contudo, que *é perfeitamente livre a criação de uma miniespecificação em outros formatos*, propiciando formas mais abstratas, sem a idéia de uma estruturação.

Este fator é de fundamental importância, já que, com um formato mais abstrato, tem-se uma análise imparcial com relação a linguagem de implementação.

Um modelo mais abstrato de miniespecificação, possibilita sem qualquer tipo de influência, que o Analista se decida por implementar utilizando uma linguagem estruturada, linguagem de quarta geração, ou orientação a objetos.

Dentro desta perspectiva, com intuito de estabelecer um único formato para as miniespecificações dos TCCs, será adotado o formato mais abstrato. Segue exemplo e algumas observações a serem consideradas, com relação ao formato proposto. O exemplo a seguir é o mesmo mostrado anteriormente, porém, utilizando a forma que deverá ser empregada nos TCCs.



PEGAR Cli_Reserva
EFETUAR cadastro do Cliente
RESERVAR quarto SE houver disponibilidade
RETORNAR mensagem ao cliente de acordo com resultado da operação

Acima um exemplo com o mesmo conteúdo mostrado anteriormente, porém, utilizando ***a forma*** que deverá ser empregada nos TCCs. Esta forma não induz a uma lógica específica para implementação que poderá vir a ser de forma estruturada, orientada a eventos, orientada a objetos ou ainda um mix destes paradigmas.

Considerações Gerais sobre o exemplo

O fluxo de dados Cli_Reserva, Qua_Reservado e Qua_Indisp, e os depósitos CadClien, CadReserva, CadQuarto devem estar descritos no dicionário de dados. Com estas descrições, mais a modelagem de dados, a miniespecificação acima é de fácil implementação em qualquer linguagem.

Observações para a construção do modelo proposto.

1. Não deve aparecer na miniespecificação qualquer descrição que já exista na lista de eventos.

2. Se uma regra de negócio já foi estabelecida e descrita a nível de dicionário de dados, não deve ser objeto de descrição na miniespecificação. Exemplo: “Não aceitar cadastro de clientes menores que 18 anos”. Se no campo cliente.idade, no dicionário de dados, já foi mencionado que o conteúdo do mesmo deve ser >= 18; então, não há motivos para especificar isto novamente na miniespecificação (evitar qualquer redundância). Detalhes com relação ao conteúdo de campos e seu significado devem necessariamente estar no dicionário de dados.

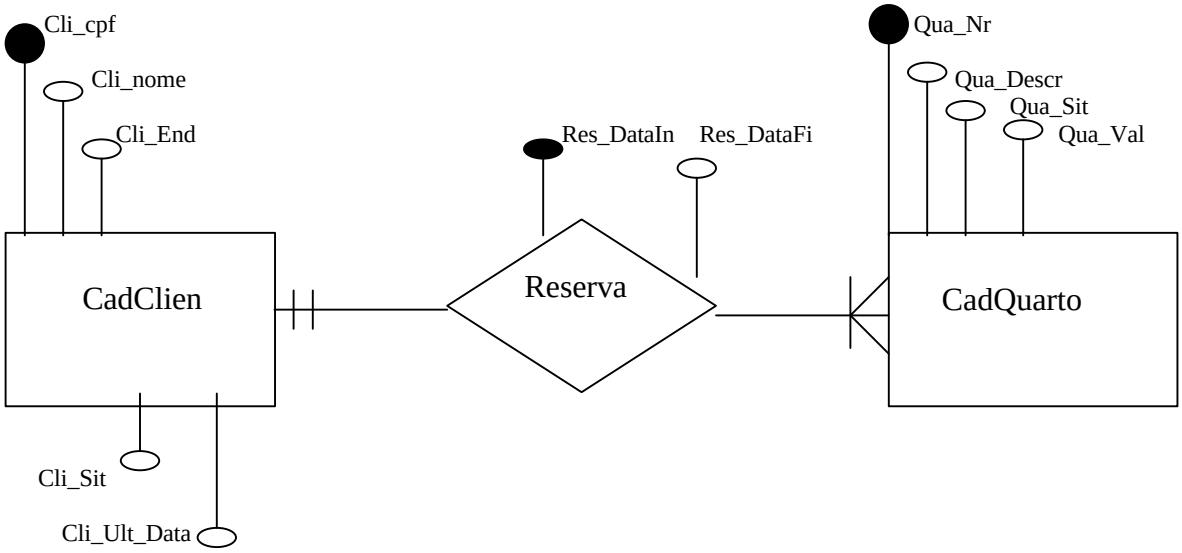
3. Iniciar as sentenças com um verbo sempre no infinitivo. O verbo deve dar a idéia da operação que se deseja realizar. Uma vez utilizado um verbo para determinada operação, ele sempre deve ser empregado quando se tratar daquela operação (padronização específica).

4. O complemento da sentença possui conteúdo livre, desde que: seja claro, sem ambigüidades e o mais sintético possível. Pode ser empregado outros verbos no complemento da sentença para facilitar a expressão da idéia.

1.1.3.6. Modelagem de Dados do Estudo de Caso

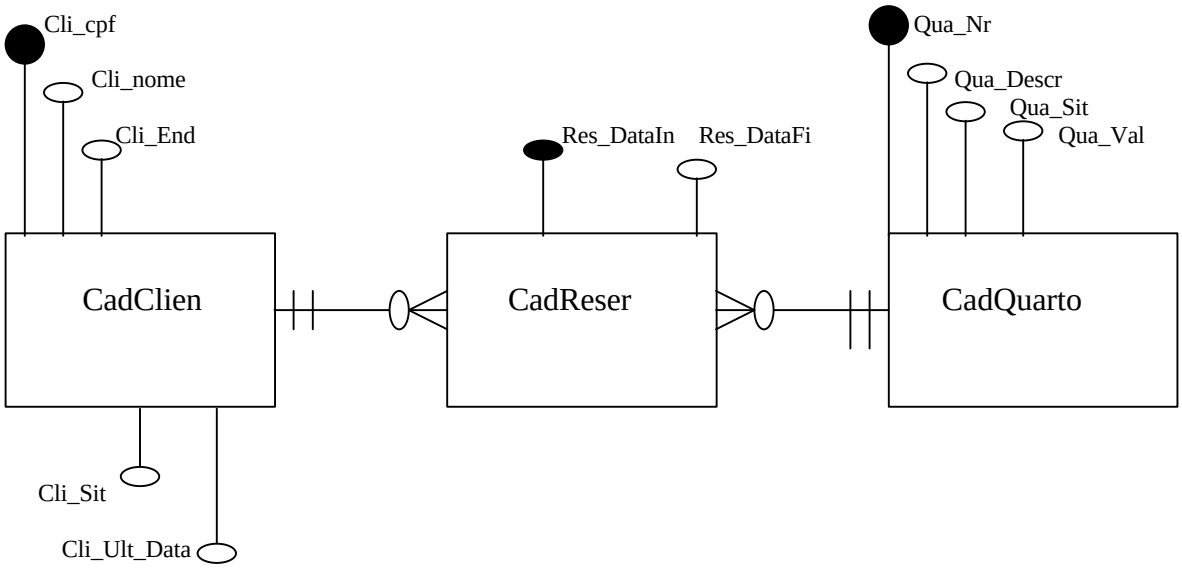
O Analista verificou que havia uma relação entre Clientes e Quartos (Reserva), sobre a qual deveria armazenar informações.

Desta forma, o Analista de Sistemas gerou o Diagrama de Entidade Relacionamento, fazendo uma modelagem lógica de dados, conforme segue:



É claro que antes de se chegar a este modelo de dados (D.E.R.), para cada depósito de dados que existia nos DFDs particionados por evento, foi elaborado uma lista de atributos, e eleito a chave principal. Depois, ao modelo inicialmente obtido, foi aplicado a *teoria da normalização* (1ª, 2ª e 3ª formas normais). O passo seguinte é a partir da modelagem lógica dos dados, gerar sua modelagem física, ou seja, criar um modelo de dados com uma forma a partir da qual os dados serão implementados de fato. Para tanto, gera-se o ***Diagrama de Estrutura de Dados***.

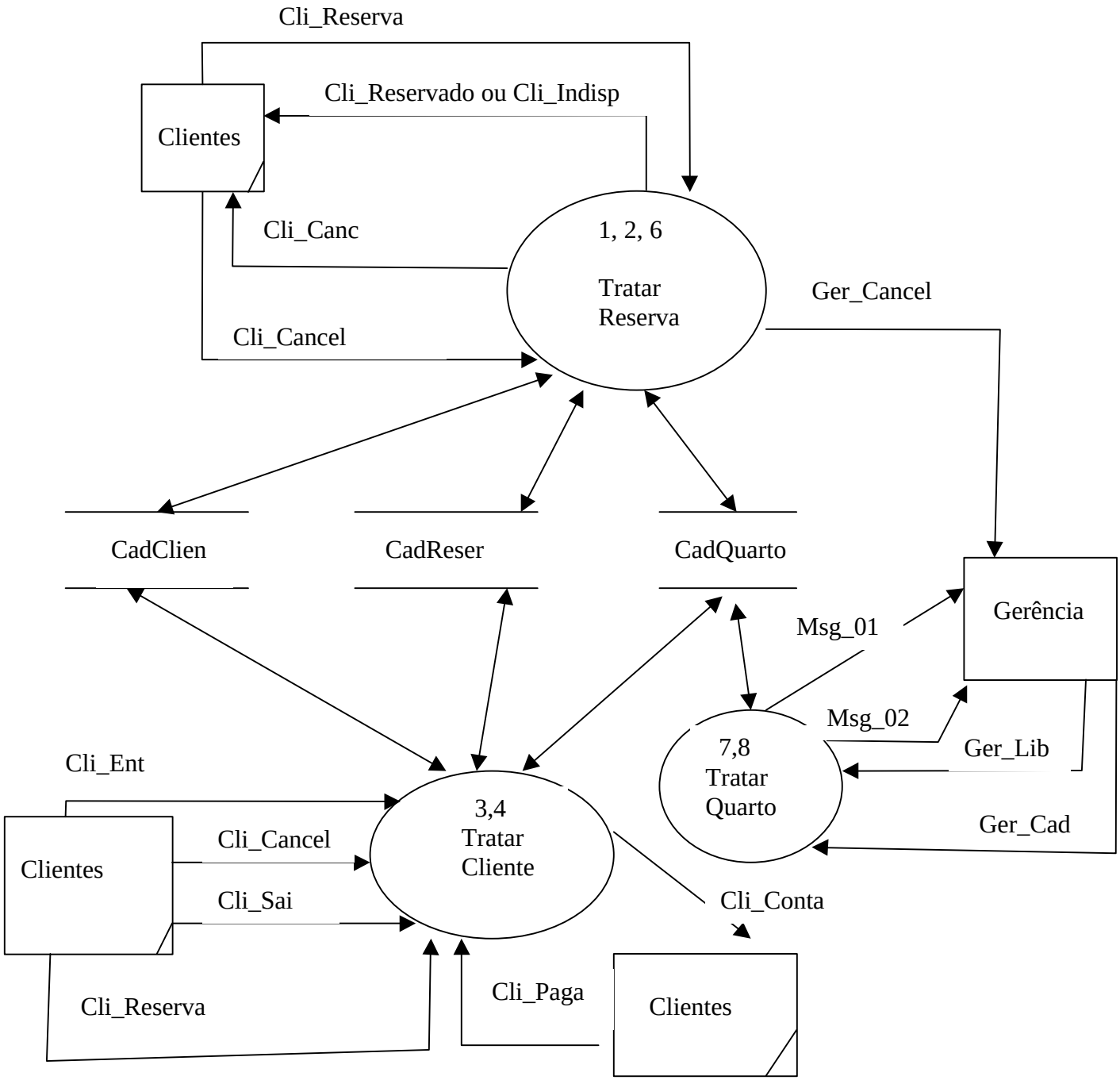
Aqueles relacionamentos existentes do DER, para os quais foi verificado a necessidade do armazenamento de atributos, no DED serão transformados em uma Entidade (depósito de dados), conforme visto a seguir.



Portanto, para modelo físico, o analista designou o nome CadClien para o arquivo ou tabela onde iria ser armazenado os dados do cliente, de igual forma CadQuarto para os dados do quarto e CadReser para os dados da reserva.

1.1.3.7. Diagrama Hierárquico de Macro Atividades

Uma vez concluído os DFDs particionados por evento, pode-se desenvolver este diagrama. Ele consiste em um DFD que agregará eventos relativos a um mesmo assunto, permitindo uma visão simplificada do sistema.



1.1.3.8 Dicionário de Dados (parcial) do Estudo de Caso

**Caracter_Valido = * Conjunto de caracteres que poderão ser utilizados *
Tipo: Alfanumérico
Tamanho: 01
[A-Z | 0 –9 | @ | & | / | a – z | , | . | - | *]**

**Cli_Cpf = * Conterá o Código do Cadastro de Pessoa Física (CIC) *
Formato: 999.999.999-99**

**Cli_Nom = * Nome do Cliente *
Tipo: Alfanumérico
Tamanho: 40
Conteúdo: {Caracter_Valido}**

**Cli_Rua = * Rua, Avenida, Praça, e n.º onde reside o Cliente *
Tipo: Alfanumérico
Tamanho: 40
Conteúdo: {Caracter_Valido}**

**Cli_Bairro = * Nome do Bairro onde reside o cliente *
Tipo: Alfanumérico
Tamanho: 20
Conteúdo: {Caracter_Valido}**

**Cli_Cidade = * Nome da Cidade onde reside o Cliente *
Tipo: Alfanumérico
Tamanho: 20
Conteúdo: {Caracter_Valido}**

**Cli_UF = * Sigla do Estado onde se encontra a cidade do cliente *
Tipo: Alfanumérico
Tamanho: 02**

**Cli_Cep = * Código do endereçamento postal de onde reside o cliente *
Formato: 99999-999**

**Cli_End = * Endereço completo de residência do cliente *
Cli_Rua + Cli_Bairro + Cli_Cid + Cli_UF + Cli_Cep**

**Cli_Sit = * Indicará se o cliente encontra-se hospedado ou não *
Tipo: Inteiro
Tamanho: 01
Conteúdo: 0 * Não está Hospedado *
1 * Encontra-se com Reserva Feita *
2 * Encontra-se Hospedado ***

**Cli_Ult_Data = * Deverá Ter a última data em que o cliente hospedou-se *
Formato: 99/99/9999**

**Cli_Registro = * Tupla do Cliente *
@Cli_cpf + Cli_Nom + Cli_End + Cli_Sit + Cli_Ult_Data**

CadCli = {Cli_Registro}

Observe que o exemplo deste pequeno dicionário de dados acima, é compatível com a entidade CadCli do modelo de dados mostrado anteriormente.

Lembre-se de que no dicionário deve estar todos os dados tratados no modelo de dados, bem como os Fluxos de Dados e Mensagens especificadas na lista de eventos.

Exercício:
Refaça o dicionário de dados (parcial) mostrado como exemplo, tentando diminuir seu tamanho sem perder o seu conteúdo. (Otimize o jeito de escreve-lo utilizando os recursos de sintaxe disponíveis)

Não Esqueça

Até este ponto (desde de o levantamento de Dados, modelo Ambiental e modelo Comportamental) tem-se os processos e resultados da Análise do Sistema.

Na continuação do processo de desenvolvimento do sistema, deverá ser elaborado o chamado ***design"***

1.1.3.9. CASE - (Computer Aided Software Engineering)

Engenharia de Software Auxiliada por Computador

Este termo foi criado no começo dos anos 80, atribuído a qualquer software que busca auxiliar o Analista na construção de sistemas.

Teoricamente, para uma ferramenta ser considerada um software CASE, ela deve incorporar e suportar toda a trajetória e técnicas de especificação e construção de um sistema de acordo com determinado método.

Deve ainda gerar o código fonte de acordo com uma linguagem escolhida, compativel com o método empregado.

No caso de Orientação a Objeto, deverá gerar o código fonte em uma linguagem OO, ou trabalhar integrado a um repositório de metadados de um banco de objetos.

Pelo fato de existir softwares de apoio ao desenvolvimento de sistemas, que não atendem na integra a definição teórica de CASE, foi criado termos para classifica-los, tais como UPPER-CASE, LOW-CASE.

Se um software auxilia o analista apenas na parte inicial da metodologia, onde são tratados os aspectos lógicos (especificação/documentação) é dito que ele é um UPPER-CASE, porém, se o software auxilia na modelagem lógica e física, propiciando até a possibilidade criação da base de dados, pode-se classificá-lo como um LOW-CASE.

2. Projeto (“design”)

Project do Inglês

maior do que aquele representado pela nossa palavra projeto. Project leva em consideração fases anteriores ao nosso projeto, abordando tarefas de planejamento,

O projeto, de que vamos tratar a partir deste ponto, portanto, é o projeto propriamente dito (no inglês encontramos mais adequadamente a palavra).

O Projeto Estruturado Moderno de Sistemas é a atividade de transformação das

implementação através da automação eletrônica.

O objetivo é modelar o sistema determinando implementar, num ambiente em que deixa-se de ter a tecnologia perfeita, passando a levar em consideração o hardware

As restrições de implementação, da tecnologia não ideal e imperfeita serão incorporadas através de atividades de infra-estrutura e administrativas.

Completeza
O projeto não deve perder nada do que foi identificado na fase de análise como requisito

Manutenibilidade
Deve-se projetar um sistema de forma a permitir facilidades de alterações, provenientes

Erros do Sistema
Novas necessidades do usuário

(observa-se que os sistemas mais fáceis de alterar são aqueles construídos de forma *modular desempenhando funções bem definidas e coesas*)

Ela está diretamente relacionada ao uso otimizado dos recursos de hardware, software e pessoal disponíveis.

Com relação aos fatores que afetam o desempenho, temos:

Deficiência do Projeto de Interface

Vários cuidados devem ser tomados quando se projeta uma interface (superfície entre duas faces), mecanismos pelo qual ocorre a comunicação atual homem-máquina.

A interface deve permitir que a comunicação se estabeleça no sentido homem-máquina e vice-versa, devendo ser um mecanismo de mão dupla, permitindo ao homem não apenas ser o agente deflagrador de um processo comunicativo, mas contemplar o desenrolar do mesmo; intervindo, requerendo ou sendo requisitado.

Quando houver entradas de dados referentes a um documento fonte original, a partir do qual as informações estão sendo transcritas, a ordem dos campos na tela (quando for o caso de um vídeo) deve obedecer a mesmo ordem dos campos existentes no documento, principalmente quando se tratar de um grande volume de transação. O analista de sistema deve estar atento e acompanhado o nascimento de novos meios para o input de dados nos computadores (voz, ondas cerebrais e outros), adequando sempre a interface segundo o novo contexto.

Tempo de acesso a disco e periféricos

Observe que para este tópico, deve-se conhecer muito bem o ambiento topológico do hardware em que se está trabalhando e sua periferia (de periféricos), além é claro do hardware que está sobre esta estrutura.

Devido ao tempo gasto em acesso a discos (que atualmente é muito maior do que as operações na CPU), deve-se prover o sistema de mecanismos que minimizem este tempo, empregando-se organizações de arquivos adequadas, ou ajustando o parque de hardware.

Falta de Reorganização de Arquivos

Em arquivos grandes de muita utilização, deve-se verificar a possibilidade de limpezas periódicas, ou particionamento do arquivo, gerando um atual e um outro histórico.

Processos longos em horário de pique

Evite ao máximo a execução de processos batch de longa duração (transferências de grandes arquivos, atualização de grande volume de dados) durante o horário de pique.

Segurança
Criar mecanismos que evitem acessos indevidos (ou não desejados) ao sistema.

-Infiltração não intencional (linhas cruzadas, irradiações)
-Acidentes – erro do usuário, falha de hardware, falha de software

Está relacionada a redução do risco de interrupção no fluxo normal de processamento das informações, que pode ser causada por:

indisponibilidade de acesso – o sistema não oferece o serviço no tempo estipulado
perda da integridade da informação

Deve haver uma admissível distribuição de custos entre os aspectos que seguem:

- custo da tecnologia
- custo operacional
- custo de construção e manutenção

2.2 Modelo de alocação de processadores

define *quem* *quem*, entende-se computador ou pessoas (processadores

Em relação aos modos de processamento de um sistema, há um espectro razoavelmente grande de possibilidades, que vai desde um processamento inteiramente manual até um

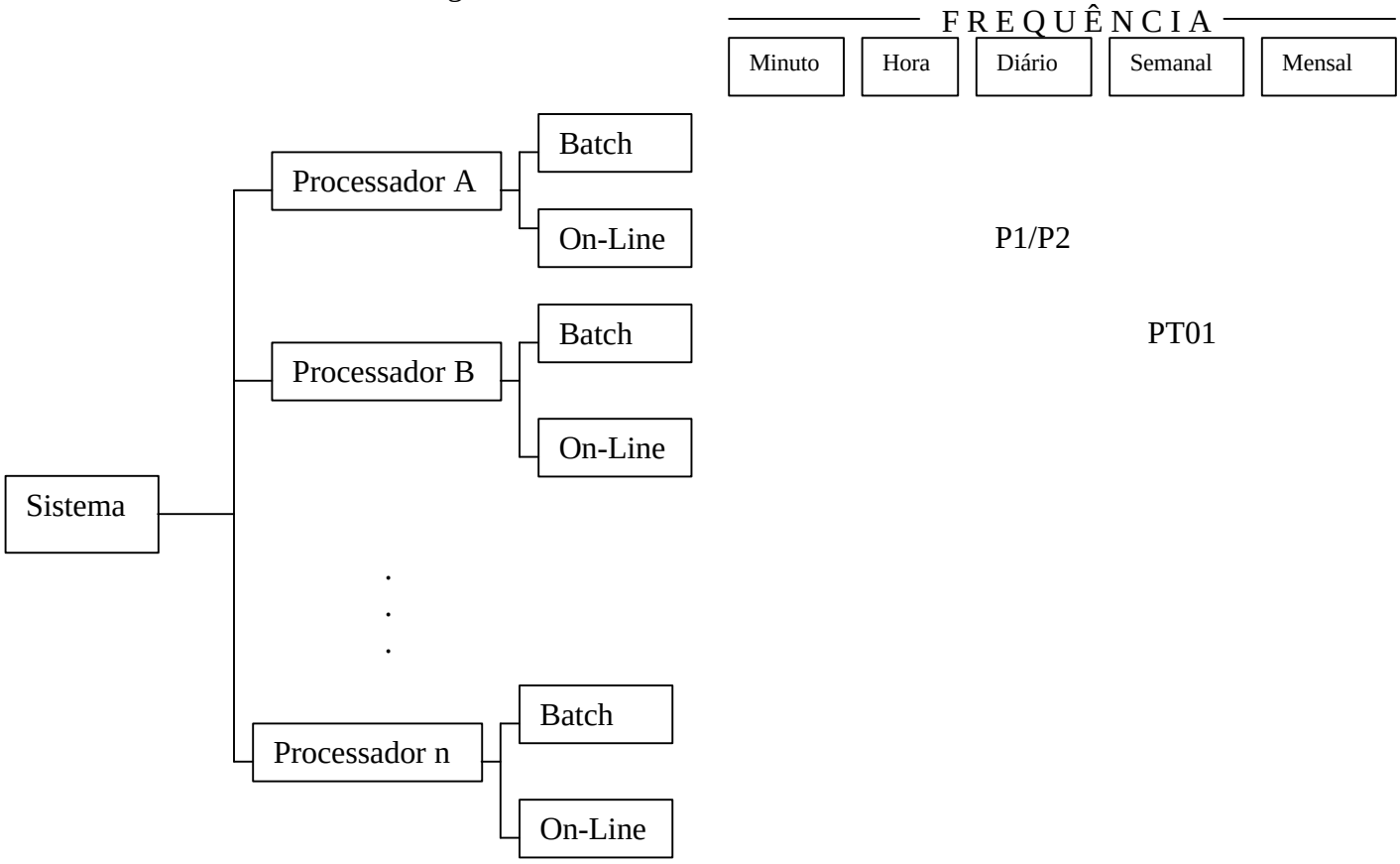
Pode-se ir desde a total centralização em um único processador até uma solução que utilize uma complexa rede de processamento distribuído. Na verdade, todo sistema possui

Tem-se aqui o objetivo de alocar as atividades essenciais (segundo a lista de eventos) aos processadores que as executarão. Deve-se também alocar atividades adicionais

Portanto, é necessário responder a estas questões:

-Quais os processos manuais e quais os automáticos ?
-Qual o processador de cada processo essencial (segundo a lista de eventos) ?
-Qual o processador de cada processo adicional (visto a limitação de tecnologia) ?
-Qual o modo de processamento de cada processo ?
-Qual a freqüência de execução de cada processo ?

Um maneira de mostrar essas características do sistema é através de um quadro de referência como a seguir:



Há várias formas que podem ser apresentadas para satisfazer visualmente a alocação de

Processador	Processo	Evento	Tipo Processamento	Frequência
PC-01	P01	Cliente Reserva Quarto		Diária
	P02	É hora de cancelar reserva	Batch	Diário
	PT01	Executar Antivírus	Batch	Semanal
PC-02 Pentium 133	P07	Quando Quarto estiver limpo gerente disponibiliza-o	On-Line	Diário
	PT01	Executar Antivírus	Batch	Semanal
	PT02	Backup dos Dados	Batch	Diário
	PT03	Reorganização de Arquivos	Batch	Anual
. . .				

Pnn = Atividade Essencial da lista de eventos
PTn = Processo de restrição Tecnológica

O exemplo acima leva vantagens sobre o anterior, notadamente pelo fato de que pode ser aplicado a pequenas ou grandes quantidades da relação processador X processo. No modelo anterior, fica muito difícil a representação de vários processos a serem executados em um mesmo processador num mesmo período.

2.3.1 Verificação e Validação

"Em 22 de Julho de 1962, o foguete espacial Mariner I foi lançado em um primeiro

desviou de sua rota e o controle terrestre deu a ordem para explodi-lo. Posteriormente, uma investigação revelou que o software apresentava erros. A omissão de um simples

fracasso de toda a missão. O custo para o contribuinte fiscal foi de 18,5 milhões de dólares" (MARTIN & MCCLURE, 1991:737).

comparado ao Mariner I, contudo, poderá se enquadrar nas mesmas proporções catastróficas, quanto aos possíveis erros que venha a apresentar.

desenvolvidos, apesar dos esforços especiais que se tem empregado para se atingir a perfeição. A preocupação com tais fatos, se deve ao prejuízo econômico que eles podem

Certamente que a proeza de se conseguir um software 100% correto no momento de sua elaboração, nem sempre é uma tarefa possível, mas isto seria o ideal.

verificação e validação, fazendo uso de técnicas dirigidas.

Um questionamento sempre levantado sobre tais atividades, refere-se ao momento ideal

durante o mesmo ?

Constata-se que a correção deve ser aplicada por todo o ciclo de desenvolvimento,

"Verificação é a demonstração da consistência, completeza e correção do software à medida que ele evolui... Validação é a demonstração de que o sistema de software

& MCCLURE, 1991:734).

Nota-se então, que a verificação busca corrigir o software etapa por etapa, durante o ciclo

finalizado.

Além do software final livre de erros, a economia monetária e de tempo que se pode obter quando se utiliza as atividades de verificação e validação, são consideráveis.

A trajetória básica e *mínima* que se espera que o Analista deva seguir é sugerido abaixo:

Com relação a Verificação
Efetuar testes de unidade a cada módulo do software desenvolvido, verificando se este atinge seus objetos sem apresentar qualquer anomalia na execução e no desempenho lógico.

Lembre-se de que *teste* deve ser a incansável busca de erros, e a *depuração*, a eficiente eliminação dos erros encontrados.

Com relação a Validação
Primeiro efetuar um teste funcional do sistema, onde, após todos os módulos terem passado pelo teste de unidade, verifica-se como se comportam quando executados em conjunto.
Submeter, este processo de teste funcional, a um rigoroso exame pelo usuário, a fim de que se manifeste sobre a validade do produto.

Segurança de Dados

Por incrível que pareça, muitos daqueles diretamente ligados a tecnologia da informação, tais como tecnológos, analistas, programadores, raramente, ao editar um texto, por exemplo, tem a preocupação de efetuar um backup. O que pensar então de um usuário mais leigo ?

É fundamental, em qualquer sistema de informação, sob qualquer aspecto, que haja mecanismos para salvaguardar os dados manipulados no sistema. Tais mecanismos podem ser desde meras rotinas diárias de backup, até sofisticados hardwares com o meio de armazenamento espelhado e componentes redundantes, ou ainda técnicas de replicagens da informação.

Para a transação da informação também há absoluta necessidade de segurança. Sob este aspecto podemos considerar dois fatos:
1) Duas ou mais operações acionadas em um processo que, para manter a integridade da informação, todas as operações devem ser corretamente concluídas. Havendo falha de uma, todo o processo deve ser revertido (Roll-Back).
2) Casos em que uma informação em trânsito poderá exigir sua mais absoluta inviolabilidade deve passar pelo processo de criptografia.

Outra característica de segurança de dados a ser tratada é o acesso aos mesmos. Tanto quanto necessário, checar identificação e senhas de acesso conforme o caso exigir.

Segurança de Hardware

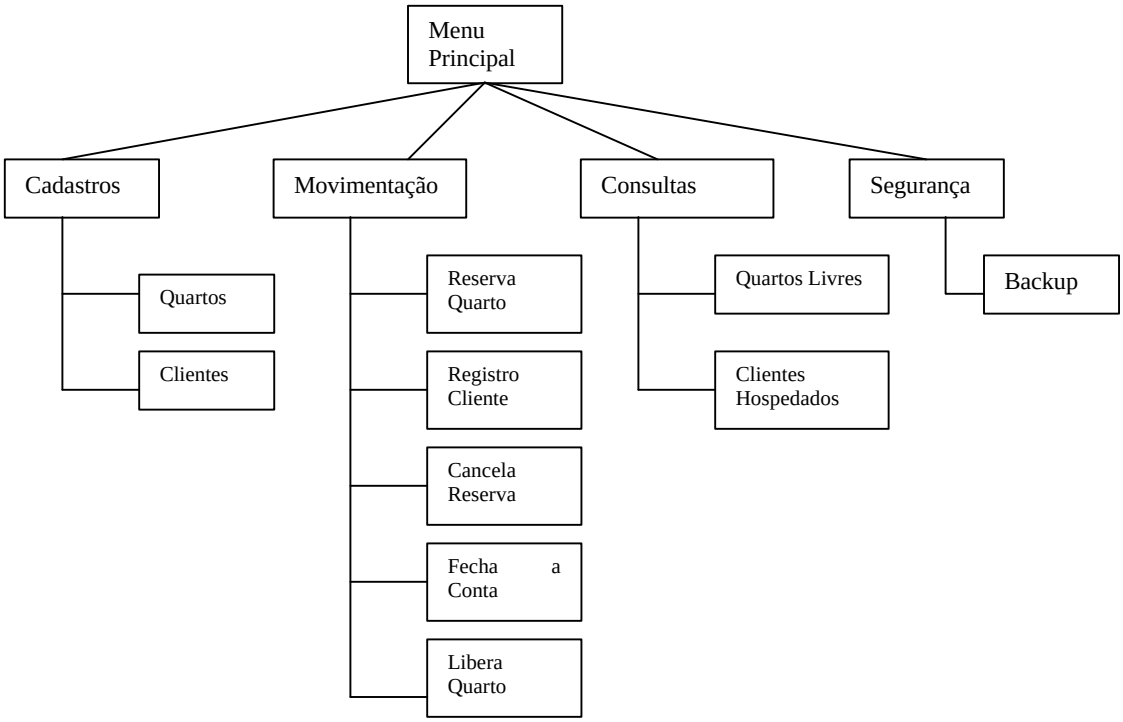
Sabe-se que para um hardware funcionar perfeitamente, existe algumas especificações a serem observadas. Via de regra, ignora-se tal fato, visto que existem muitos lugares onde, por exemplo, o pino terra da alimentação elétrica não é utilizado. Outros fatores importantes como a estabilidade da tensão elétrica e a temperatura ambiental são fatores encarados como totalmente secundários.

As instalações para utilização do hardware deve atender as suas especificações. Caso contrário, o seu sistema já começa com alto risco de erro.

2.4 Diagrama hierárquico do sistema

Mais como um mecanismo de estabelecimento da organização hierárquica do sistema, o diagrama também é um documento de interface, já que é um esboço do que virá a ser o menu do sistema.

Desta forma ele também é parte integrante da documentação do sistema, uma vez que se pode recorrer a ele para identificar posições ou seqüência de caminho até se chegar a uma determinada opção.



Observa-se que podem haver outros programas no sistema, porém, sua execução dá-se em função de outros fatos, que não sejam da escolha de um usuário. Como exemplo pode-se citar: programa de conversão de valor para extenso, programas cujo estímulo seja um fluxo temporal (rodar todo dias as 12h).

Estas chamadas de programas estarão documentadas nas miniespecificações dos programas chamadores. Por exemplo, em um programa que emita um recibo, cujo valor deve ser convertido para extenso, deverá conter em sua miniespecificação a chamada a esta subrotina ou programa.

2.5 Lay-Out de telas / relatórios

Qualquer sistema de informação utiliza interfaces e cuida do meio pelo qual este mecanismo opera.

Na construção dos sistemas de informação, é necessário um cuidado especial com relação aos requisitos exigidos por cada uma das partes (homem-máquina) que se ligam pela interface.

Para a máquina basta que o meio físico esteja em perfeita condição de funcionamento, pois assim, os dados do "input" estando corretos, o resultado certamente estará (sob ponto de vista do hardware).

O homem por sua vez, ao receber uma informação proveniente da máquina, deve visualizá-la de uma forma amigável e de fácil entendimento. Este aspecto pode ter muita influência na negociação política de venda do produto para o usuário.

Em certos momentos a forma mais amigável e de fácil entendimento, será aquela que o usuário deseja ver, independente do que você tenha estudado sobre interfaces e ergonometria (não se esqueça disto).

O fato é que uma vez definido um relatório ou uma tela, tente seguir aquele modelo como padrão para os demais.

Por que padronizar ? Ora, você estará universalizando funções, bem como diminuindo o tempo de aprendizado e uso de seu sistema. Não reinvente a roda. Se já existe um certo padrão de mercado, emprega-o .

Veja por exemplo, se você fizer *hoje* um software para utilização em um microcomputador IBM-PC, qual a tecla mais indicada para ser utilizada como help do seu sistema ? Por quê ?

mencione:

Nome da tela/Relatório

Data / Hora
Página

no caso de um desenvolvimento para o ambiente WINDOWS, você deve respeitar todas as características de interface já definidas.

Atualmente existe um leque muito grande de recursos para tornar amigável a interface homem-máquina.

atualmente melhor estão sendo empregadas, já que é uma forma bastante amigável de enviar e receber informações.

como sons, imagens estáticas, animadas ou filmadas e outros elementos de multimídia.

“Multimídia é a combinação de texto, som e vídeo para apresentar informações por

muda a maneira como as pessoas usam os computadores...” (JAMSA, 1993:XIII).

Por exemplo, em um sistema de controle de biblioteca, poderia-se ter além da opção de

na língua previamente escolhida, e também, a seu critério, poderia “ver” as imagens relativas a estória que estaria sendo narrada.

possibilidade futura da inexistência de um vídeo como hoje o concebemos, mas a projeção de imagens holográficas, propiciando a interação através de uma realidade

circunstâncias normais (como por exemplo, caminhar dentro de um vulcão ou tapar sua chaminé e verificar o resultado, segundo as leis da física).

3. Gerência de Projetos

A partir deste ponto vamos tratar de projeto (“Project”) no sentido do empreendimento global, diferente portanto de projeto (“design”) que é apenas uma fase dentro deste empreendimento.

Exceto pela execução das atividades de desenvolvimento propriamente dito, que envolvem metodologias, técnicas e ferramentas de desenvolvimento, tudo mais em um projeto de sistemas é ação de gerência.

Assim deve-se ter em mente o que segue, como definição do nosso empreendimento global:

Projeto de Sistemas = estabelecimento de OBJETIVOS +
Planejar e executar ATIVIDADES +
Administrar PRAZOS +
Gerenciar RECURSOS +
Conviver com RISCOS/INCERTEZAS

3.1 Problemas em projetos de sistemas

3.1.1 Relacionados com a rápida evolução da tecnologia

- Proliferação desordenada de microcomputadores
- Vida útil do Hardware
- Grande quantidade de software não integrado
- Aumento dos níveis de expectativa devido à divulgação instantânea das inovações
- Ferramentas de desenvolvimento muito recentes, ainda não consolidadas

3.1.2 Relacionados com pessoas

- As pessoas freqüentemente não sabem o que realmente querem ou necessitam
- desenvolvimento de sistemas exige muita comunicação:
  - No. De interações dois a dois = N(N – 1) / 2
  - Muitas pessoas falam bem, pouquíssimas ouvem bem
  - Palavras tem diferentes significados

- É muito comum desenvolvedores não ouvirem os usuários e raramente constróem o que é necessário
- Desenvolvedores não aceitam mudança no seu próprio domínio (paradigma de comportamento)
- Conflitos sempre existirão. Cabe ao gerente administrar adequadamente

3.1.3. Outros problemas Gerenciais

“ Imprevisibilidade “ previsível

Férias de elementos da equipe
Baixa disponibilidade de recursos de hardware
Alocação temporária de pessoal para outro sistema
Mudança de prioridades da organização

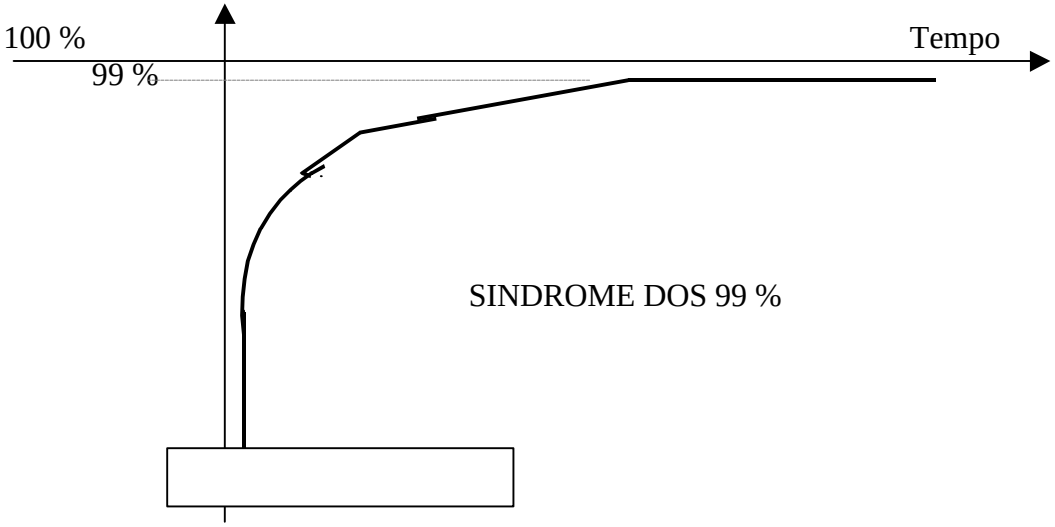
Seleção de pessoal

Quantidade X Qualidade
Nível do profissional (conhecimento X experiência X dinamismo)

A somatória destes dois fatores trás como conseqüência um terceiro:

Dificuldade de estimar prazos e recursos

3.1.4. Dificuldade de aferir progresso



Não há fórmulas mágicas para evitar atrasos em projetos de sistemas (No silver bullet – não tem bala de prata)

A única maneira de monitorar o progresso de um projeto, é através do estabelecimento de marcos ou pontos de controle, ao longo das fases e atividades que o compõem, que se constituem em etapas sobre as quais podemos afirmar com certeza: está 100% concluída.

3.1.5. A Crise do Software

Em 1993 foi feito uma estimativa sobre das metodologias, técnicas e ferramentas de desenvolvimento de sistemas existentes na época (que foram uma somatória evolutiva dos 25 anos anteriores), e pode-se concluir que:

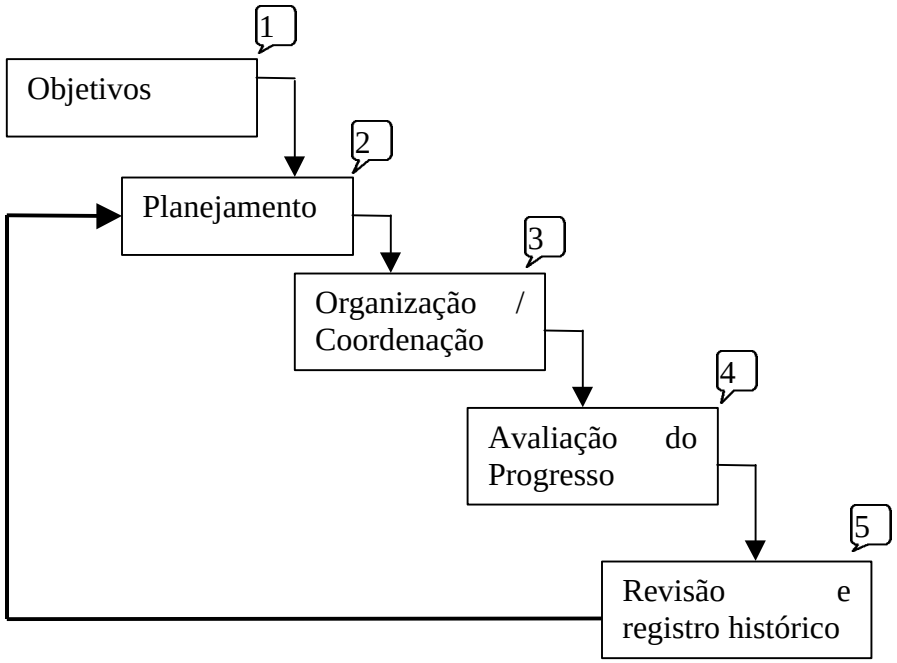
“ Se todos os sistemas necessários tivessem que ser implementados nos próximos dez anos com a tecnologia de software disponível, mais da metade da população do mundo teria que ser de programadores ”.

O recente advento dos gerenciadores de bancos de dados, das ferramentas CASE (Computer Aided Software Engineering) , das linguagens de programação visuais e da orientação a objeto tem contribuído para melhorar a produtividade no desenvolvimento de sistemas. Porém, apenas uma gerência adequada do uso dessas ferramentas tornará possível a superação da crise do software ainda existente.

Muitas pessoas acreditam que desenvolver um sistema é simplesmente programar.

Como se fosse possível aplicar a sistemas complexos do mundo real, as mesmas decisões de projeto, utilizadas na resolução de problemas que estão ao alcance da compreensão de um único programador.

4. Processo de Gerência de Projeto



1. **Defina os objetivos do projeto**

Repetir

2. **Planejar as tarefas para alcançar os objetivos**
3. **Organizar / Coordenar os recursos – colocar as tarefas em execução**
4. **Avaliar o progresso em relação ao plano – até alcançar os objetivos**
5. **Revisar o planejamento, a organização dos objetivos, conforme necessário. Registrar o histórico do projeto**

FimRepetir

4.1.

O primeiro princípio de gerência de projetos é estabelecer um conjunto claro de objetivos e requisitos do projeto, bem como obter informações que permitam decidir sobre a

Objetivos técnicos ou meta do projeto

Especificações do produto, lista de padrões aplicáveis, restrições assumidas

Prazos: sempre um fator limitante

Orçamento: Quase sem um fator limitante

Muitas vezes, estas informações para serem uma base segura sobre a qual seja possível estabelecer um "contrato de desenvolvimento", devem estar reunidas em um documento, *Proposta de Desenvolvimento de Sistema.*

Organização da Proposta de Desenvolvimento de Sistema

antecipar respostas a perguntas que surgirão durante o desenvolvimento.

Conteúdo:

Descrição do sistema a ser desenvolvido, definindo seus objetivos, requisitos, as necessidades e expectativas dos usuários bem como as vantagens do sistema proposto;

proposto

Para que a proposta cumpra seu papel, ela deve ser:

- Macroscópica, de forma a exigir um mínimo de detalhe;
- Fidedigna, sem omitir ou exagerar fatos;
- Imparcial;
- Realista e viável;

Em uma estrutura organizacional mais simples, a proposta de desenvolvimento acaba se resumindo num documento interno do tipo Ordem de Serviço, em outras organizações menores ainda, simples reuniões com o desenvolvedor, de forma verbal, se elabora todo este contexto acima exposto referente a proposta de desenvolvimento.

4.2. Planejamento

Um planejamento bem feito deve primeiro definir quais atividades devem ser realizadas (O QUE), devidamente justificadas (POR QUE), em que ordem (QUANDO) devem ser feitas, com que técnicas (COMO), empregando quais recursos humanos (QUEM), trabalhando em que lugar (ONDE).

Esta matriz O QUE x (POR QUE, QUANDO, COMO, QUEM, ONDE), permitirá estimar QUANTO será despendido em cada atividade, nas fases e no projeto como um todo, em termos de TEMPO e CUSTO.

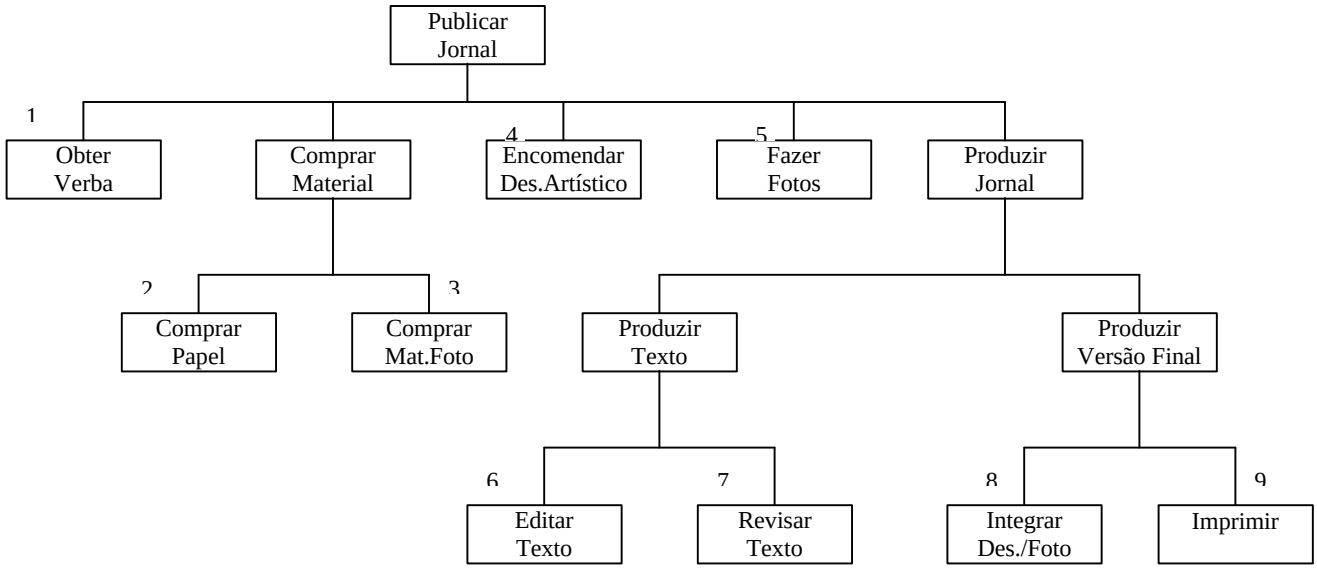
Na prática, o planejamento não é tão cartesiano como aparenta. O grau de complexidade e *precisão* depende do estágio em que se está planejando. Nos estágios iniciais, ele é mais simples e impreciso, nos estágios mais avançados mais complexo e detalhado. Ao longo do desenvolvimento, o objetivo do planejamento é um alvo móvel.

Portanto, segundo dizem, " a única certeza sobre um plano é que as coisas não ocorrerão conforme planejado"; ainda assim, tenha um plano, ruim com ele, muito pior ainda sem.

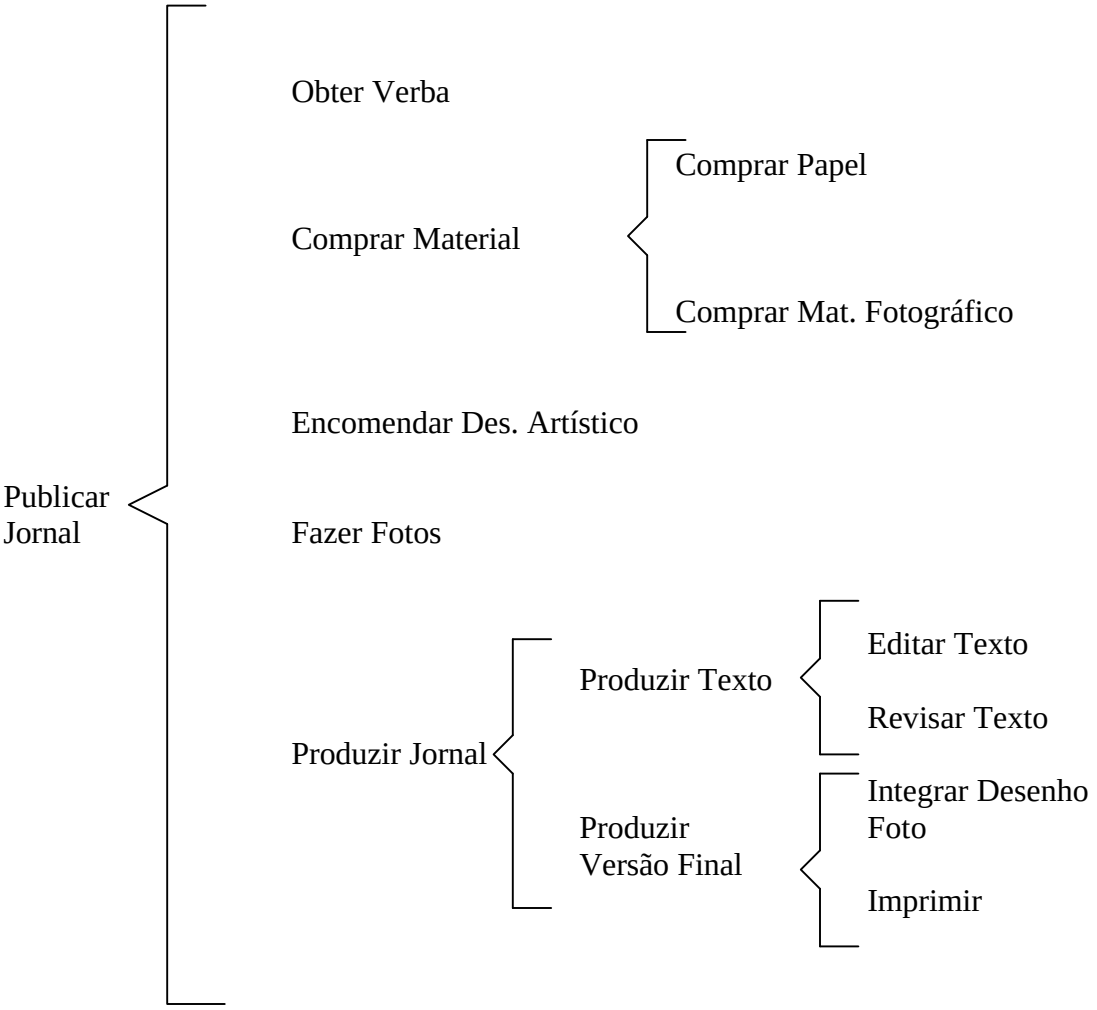
Exemplo de Tabela de Planejamento

O quê ? Atividades	Por Que ?	Quem Fará	Quando	Onde	Como ferramentas e métodos	Quanto Custa ?	Quanto Tempo ?

Outra forma de efetuar um planejamento é a utilização e um organograma, conforme exemplo abaixo:



Exemplo de planejamento, utilizando aplicação de chaves



Publicar Jornal

1. Obter Verba

2. Comprar Material

    2.2. Comprar Mat. Fotográfico

3. Encomendar Des. Artístico

5. Produzir Jornal

    5.1 Produzir Texto

        5.1.2 Revisar Texto

    5.2 Produzir Versão Final

        5.2.2 Imprimir

Exercício: Fazer um DFD para expressar o planejamento de acordo com os exemplos

Exemplo do planejamento com lista de atividades, incluindo duração e seqüência

Atividades

(a) Obter Verba
(b) Comprar Papel
(c) Comprar Material Fotográfico
(d) Encomendar Desenho Artístico
(e) Fazer Fotos
(f) Editar Texto
(g) Revisar Texto
(h) Integrar Foto/Desenho
(i) Imprimir

Anterior	**Atividade**	**Posterior**	**Duração**
-	(a)	(b) (c) (d)	3
(a)	(b)	(i)	2
(a)	(c)	(e)	1
(a)	(d)	(h)	7
(c)	(e)	(h)	3
-	(f)	(g)	5
(f)	(g)	(h)	5
(e) (d) (g)	(h)	(i)	1
(b) (h)	(i)	-	1

Planejamento utilizando o diagrama de GANTT

ATIVIDADES	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**
(a) Obter Verba												
(b) Comprar Papel												
(c) Comprar Mat.Fotog.												
(d) Encom.Des.Artist.												
(e) Fazer fotos												
(f) Editar Texto												
(g) Revisar Texto												
(h) Integrar Desc./Foto												
(i) Imprimir												

Atividades

(a) Obter Verba
(b) Comprar Papel
(d) Encomendar Desenho Artístico
(e) Fazer Fotos
(g) Revisar Texto
(h) Integrar Foto/Desenho
i) Imprimir

HUMANOS	**01**		**03**	**04**		**06**	**07**		**09**	**10**		**12**
EDITOR											(h)	(i)
				(b) (c)								
				(d)	(e)							
	(f)											
REVISOR												

Lembre-se que onde existe a numeração, pode-se trocar por Datas.

4.3. Organização / Coordenação

A tarefa de organizar pode ser vista como montar um time: dado um grupo de pessoas e uma meta comum, qual é o melhor papel para cada indivíduo, e como as responsabilidades devem ser divididas ? Como motivar estas pessoas ? Como gerar uma sinergia ?

Veja como não existe a *priore*, respostas para estas questões, dissociadas de um contexto onde aplica-las (pessoas com seus interesses, visão de mundo, qualificação, experiências diferenciadas). Portanto, a tarefa de coordenação não é fácil (para um técnico, pior ainda, se não tiver um mínimo de formação voltada para ciências humanas/administrativas).

Objetivos da Organização / Coordenação

Combinar conhecimentos técnicos de cada pessoa com as tarefas que ela vai realizar

Reduzir ao mínimo as horas ociosas das pessoas

Designar cada pessoa para apenas uma tarefa de cada vez

Obtenha não apenas o envolvimento, mas o comprometimento das pessoas

Diferença entre envolvimento e comprometimento.

Na floresta, numa linha manhã ensolarada, a coruja, a galinha e o porco se encontram:

CORUJA: Vamos fazer um omelete ?
TODOS: Vamos !!!
CORUJA: Bom, eu posso entrar com a frigideira .

A galinha pensou, pensou, pensou...

GALINHA: Eu posso entrar com os ovos.
CORUJA: Legal !

Por alguns segundos a coruja e a galinha entreolharam-se e simultaneamente, voltaram-se ao porco...

GALINHA e CORUJA: Você entra com o bacon...

Moral da estória: A galinha estava envolvida o porco comprometido.

Fatores Humanos

Tenha sempre em mente que as Pessoas são o mais importante recurso em projetos de sistemas (às vezes o único além do tempo).

Serão também as pessoas que irão julgar a utilidade dos softwares desenvolvidos, portanto, é fundamental que se entenda um pouco de fatores humanos (psicologia da personalidade, motivação, comunicação)

Aspectos chaves:
- Trabalho em grupos pequenos
- Liderança técnica por competência
- Local de trabalho adequado

Benefícios:
- Redução de problemas de comunicação
- Padrão de qualidade
- Aprendizado mútuo
- Sociabilização do trabalho

Avaliação do Progresso

acordo com pontos de controles especificados (*Milestones*

Dois níveis de controle:

Informal

equipe; não gera relatórios.
Um controle informal efetivo, minimiza a freqüência e as sobrecarga ( ) de reuniões e relatórios.

Requer relatórios periódicos gerados por revisões sobre andamento do projeto. Normalmente, são de dois tipos:

_____ – reuniões realizadas em cada ponto de controle com todo o grupo de desenvolvimento; gerando um relatório de progresso (com a narrativa do ponto atual e

Revisões Técnicas – voltadas para aspectos específicos dentro das etapas existentes,

objetivo eliminar problemas de transcurso, mantendo um registro sobre isto – relatório técnico.

Exemplo de um projeto com pontos de controle

FASE	**ATIVIDADE**	**PONTOS DE CONTROLE**
100		
110	Avaliação Preliminar	
	Estudo do Projeto	
199		Fase 100 Concluída
300		
310	Esboço das Funções do Sistema	
	Avaliação de Recursos Tecnológicos	
330		
340	Análise de	
350	Revisão e Aprovação	
	Ponto de Revisão	Fase 300 concluída
	DESENVOLVIMENTO	
510		
520	Projeto Lógico (Comportamtal)	Atividade 520 concluída
	Projeto Físico (Implementação)	
540		Fase 500 concluída
700		
710	Planejamento da Implantação	
	Teste Piloto	
730		
	Instalação e Treinamento	
740		Fase 700 concluída
		(encerramento do projeto)

Um exemplo de pauta para reuniões referente as revisões gerenciais

Que pode ser aplicada em qualquer momento dentro da evolução do projeto:

01 Atividades Desenvolvidas
02 Fases e Atividades em curso
03 Problemas Encontrados
04 Soluções Propostas
05 Atividades a serem desenvolvidas
06 Ocorrência de Contingências
07 Considerações sobre os prazos
08 Providências
09 Conclusão da Reunião

Revisões Técnicas

Com relação as *Revisões Técnicas*, se propõe:

01 Revisão de Requisitos

Motivação principal: Assegurar que, se os desenvolvedores construírem um sistema que satisfaça os requisitos, ele satisfará as necessidades dos usuários.

Avaliação dos aspectos:

Incorreção:	requisitos que demandam algo que os usuários não querem.
Imcompleteza:	a especificação falha em dizer algo que algum usuário necessita.
Ambigüidade:	requisitos que são imprecisos a ponto de admitir muitas possíveis interpretações.
Inconsistência:	requisitos que se contradizem um ao outro

02 Revisão de design

O design satisfaz os requisitos ?

O design é consistente ?

O design é completo ?

Fazendo parte deste elenco, aplicar "_____" (inspeções para validação e verificação)

desenvolvimento. Na fase de análise de requisitos, o objetivo é a validação do sistema, em geral envolvendo usuários.

Nas fases subsequentes, o objetivo é a verificação do desenvolvimento, como pressuposto de que os requisitos estão válidos.

4.5

Revisar um projeto é o processo de fazer mudanças a fim de resolver os desvios do planejamento.

alvo móvel do projeto.

São cinco as boas razões usuais para se fazer revisão:

Perda de prazo para o término de tarefas
Tarefas mal feitas
Tarefas não realizadas
Mudanças imprevistas de pessoal
Corte de recursos inicialmente previstos

5. BIBLIOGRAFIA

Kugler, Aguinaldo Aragon Fernandes & Kugler, José Luiz Carlos.
Gerência de Projeto de Sistemas. Rio de Janeiro, LTC, 1990.

Martin, James & McClur, Carma.
Técnicas Estruturadas e Case. São Paulo. Makron, 1991.

McMenamin, Stephen M. & Palmer, John F.
Análise Essencial de Sistemas. São Paulo, McGraw-Hill, 1991.

Pompilho, S.
Análise Essencial. Rio de Janeiro. Infobook, 1994.