Questionamentos

- **Por que tanta demora para entregar?**
- **Por que os prazos se atrasam?**
- **Por que os custos são altos?**
- **Por que não achar todos os erros antes de entregar?**
- **Por que dificuldade em medir o progresso do desenvolvimento de um software?**

Características do Software

- 1- desenvolvido ou projetado por engenharia, não manufaturado no sentido clássico (industrial)
  - sucesso é medido pela qualidade e não quantidade
- 2- não se “desgasta”, mas se deteriora devido as mudanças (Figura)
- 3- a maioria é feita sob medida em vez de ser montada a partir de catálogos de componentes existentes (reusabilidade de software)

Aplicações do software

- SOFTWARE de Sistema: *sistema operacional, compiladores, drivers, etc...*
- SOFTWARE DE TEMPO REAL: sistema de controle de tráfego aéreo, relógio digital, ...
- SOFTWARE Business: folha de pagamentos, contas a pagar e a receber, ...
- SOFTWARE CIENTÍFICO E DE ENGENHARIA : *astronomia, vulcanologia, análise de fadiga da mecânica de automóveis, biologia...*
- SOFTWARE EMBUTIDO*: controle de microondas, combustível, sistemas de freio*
- SOFTWARE DE COMPUTADOR PESSOAL: *envolve processamento de textos, planilhas eletrônicas, computação gráfica, diversões, etc.*
- SOFTWARE Baseado na Web *páginas Web (websites) recuperados por um browser (tecnologia web ASP, HTML, CGI, JAVA, XML, etc...)*
- SOFTWARE DE INTELIGÊNCIA ARTIFICIAL *faz uso de algoritmos não numéricos para resolver problemas que não sejam favoráveis à computação*

Evolução do software

Hardware: maior desempenho, menor tamanho e custo

Os primeiros anos
•Orientação batch
•Distribuição limitada
•Software customizado

A segunda era
•Multiusuário
•Tempo real
•Banco de Dados
•*Software Houses*

A terceira era
•Sistemas Distribuídos
•"Inteligência" embutida
•Hardware de baixo custo (PCs)
•workstation
•Impacto de consumo

A quarta era
•Tecnologia O. O.
•Sistemas Especialistas
•Redes Neurais
•Computação Paralela
•Tecnologia Web
•Segurança

Evolução do software

(1950 - 1965)

- O hardware sofreu contínuas mudanças
- O software era uma arte "secundária" para a qual havia poucos métodos sistemáticos
- O hardware era de propósito geral
- O software era específico para cada aplicação
- Não havia documentação

Evolução do software

(1965 - 1975)

- Multiprogramação e sistemas multiusuários
- Técnicas interativas
- Sistemas de tempo real
- 1a. geração de SGBD’s
- Produto de software - *Software Houses*
- Cresce número de sistemas baseado em computador
- Manutenção quase impossível

..... CRISE DE SOFTWARE

Evolução do software

(1975 - 1990)

- Sistemas distribuídos
- Redes locais e globais
- Uso generalizado de microprocessadores - produtos inteligentes
- Hardware de baixo custo
- Impacto de consumo

Evolução do software

(Quarta era do software de computador)

- ✓ Tecnologias orientadas o objetos
- ✓ Sistemas especialistas e software de inteligência artificial usados na prática
- ✓ Software de rede neural artificial
- ✓ Computação Paralela

Crise de software

Refere-se a um conjunto de problemas encontrados no desenvolvimento de software:
(problemas não se limitam a softwares que não funcionam adequadamente)

1- As estimativas de prazo e de custo freqüentemente são imprecisas

2- A produtividade das pessoas da área de software não tem acompanhado a demanda por seus serviços

3- A qualidade de software às vezes é menos que adequada

4- O software existente é muito difícil de manter

Causas dos problemas associados à crise de software

1- PRÓPRIO CARÁTER DO SOFTWARE

O software é um elemento de sistema lógico e não físico. Conseqüentemente o sucesso é medido pela qualidade de uma única entidade e não pela qualidade de muitas entidades manufaturadas

O software não se desgasta, mas se deteriora

Causas dos problemas associados à crise de software

2- FALHAS DAS PESSOAS RESPONSÁVEIS PELO DESENVOLVIMENTO DE SOFTWARE

Gerentes sem nenhum *background* em software

Os profissionais da área de software têm recebido pouco treinamento formal em novas técnicas para o desenvolvimento de software

Resistência a mudanças.

Mitos do software(Gerenciais)

Mito: **Meu pessoal tem ferramentas de desenvolvimento de software de última geração; afinal compramos para eles os mais novos computadores.**

Realidade:

- *É preciso muito mais do que os mais recentes computadores para se fazer um desenvolvimento de software de alta qualidade.*
- *Ferramentas de engenharia e software auxiliada por computador CASE (Computer-Aided Software Engineering) são mais importantes do que o hardware*

Mitos do software(Gerenciais)

Mito: **Se nós estamos atrasados nos prazos, podemos adicionar mais programadores e tirar o atraso (conceito de hordas de mongóis).**

Realidade:

– *O desenvolvimento de software não é um processo mecânico igual à manufatura.*

– *Acrescentar pessoas em um projeto torna-o ainda mais atrasado.*

– *Pessoas podem ser acrescentadas, mas somente de uma forma planejada e bem coordenada.*

Mitos de software(Cliente)

Mito: **Uma declaração geral dos objetivos é suficiente para se começar a escrever programas - podemos preencher os detalhes mais tarde.**

Realidade:

– *Uma definição inicial ruim é a principal causa de fracassos dos esforços de desenvolvimento de software.*

– *É fundamental uma descrição formal e detalhada do domínio da informação, função, desempenho, interfaces, restrições de projeto e critérios de validação.*

Mitos de software(Cliente)

Mito: **Os requisitos de projeto modificam-se continuamente, mas as mudanças podem ser facilmente acomodadas, porque o software é flexível.**

Realidade:

- ***Requisitos podem ser mudados, mas o impacto varia de acordo com o tempo que é introduzido (projeto e custo)***
- ***Um mudança, quando solicitada tardiamente num projeto, pode ser mais do que a ordem de magnitude mais dispendiosa da mesma mudança solicitada nas fases iniciais (Figura)***

Mitos do software(Profissional)

Mito: **Assim que escrevermos o programa e o colocarmos em funcionamento nosso trabalho estará completo.**

Realidade:

– *Os dados da indústria indicam que entre 50 e 70% de todo esforço gasto num programa serão despendidos depois que ele for entregue pela primeira vez ao cliente*

Mitos do software(Profissional)

Mito: **Enquanto não tiver o programa "funcionando", eu não terei realmente nenhuma maneira de avaliar sua qualidade.**

Realidade:

– *Mecanismo (Revisão Técnica Formal) de garantia de qualidade de software é aplicado desde o começo do projeto*

– *Revisões de software são um "filtro de qualidade" - descobre erros/defeitos*

Mitos do software(Profissional)

Mito: **A única coisa a ser entregue em um projeto bem-sucedido é o programa funcionando.**

Realidade:

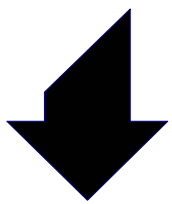
– *Um programa funcionando é somente uma parte de uma Configuração de Software que inclui todos os itens de informação produzidos durante a construção e manutenção do software.*

– *A DOCUMENTAÇÃO é o alicerce*

Qual é a SOLUÇÃO?

- Reconhecer os problemas e suas causas e desmascarar os mitos do software são os primeiros passos

  Métodos e Técnicas para o disciplinar o processo de desenvolvimento do software



Engenharia de Software

Engenharia de Software

ENGENHARIA DE SOFTWARE

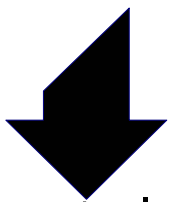
Uma disciplina da Ciência da Computação que oferece Métodos, Técnicas e Ferramentas para desenvolver e manter softwares com alta qualidade para a resolução de problemas

(Anneliese Mayrhauser 1990)

Engenharia de Software

Abrange um conjunto de três elementos fundamentais:

Métodos, Ferramentas e Processos (Procedimentos)



- Possibilitar ao gerente o controle do processo de desenvolvimento
- Oferecer ao profissional uma base para a construção de software de alta qualidade

Engenharia de Software

MÉTODOS: **proporcionam os detalhes de “**como fazer**” para construir o software**

FERRAMENTAS: **fornecem suporte automatizado ou semi aos métodos.**

PROCEDIMENTOS: **constituem o elo de ligação entre os métodos e ferramentas**

Engenharia de Software

ENGENHARIA DE SOFTWARE: **compreende um conjunto de *etapas* que envolve** MÉTODOS, FERRAMENTAS **e** PROCEDIMENTOS.

- **Essas etapas são citadas como** CICLOS DE VIDA ou MODELOS DE PROCESSO DE SOFTWARE
  - Uma estratégia de desenvolvimento que englobe processos, métodos e ferramentas, e as fases de desenvolvimento

Engenharia de Software

- **Engloba um PROCESSO, GERENCIAMENTO E MÉTODOS TÉCNICOS, FERRAMENTAS**
- **Engenharia é Análise, Projeto (design), Implementação, Verificação e Gerenciamento de entidades técnicas**
  - **Definição - O QUÊ (*What*)**
  - **Desenvolvimento - COMO (*How*)**
  - **Suporte - MANUTENÇÃO (*Change*) - Segurança**
  - **corretiva, adaptativa, melhoramento funcional, preventiva (Reengenharia)**

Definição: Modelo

- Modelagem é uma técnica de engenharia aprovada e bem aceita
- Um modelo é uma simplificação da realidade.
  - **Planos de detalhes, podem ser estruturais (organização do sistema) ou comportamentais (dinâmica do sistema)**
- Construímos modelos para compreender melhor o sistema que estamos desenvolvendo.
  - **ajudam a visualizar o sistema como desejamos que seja**
  - **especificar a estrutura e comportamento**
  - **guia para construção do sistema**
  - **documentam as decisões tomadas**

Definição: Modelo

- Construímos modelos de sistemas complexos porque não é possível compreendê-los em sua totalidade.
  - **ajudam a visualizar o sistema como desejamos que seja**
  - **especificar a estrutura e comportamento**
  - **guia para construção do sistema**
  - **documentam as decisões tomadas**
- Os melhores modelos estão relacionados à realidade (modelos simplificam a realidade)
- Nenhum modelo único é suficiente. Conjunto de modelos independentes

Processo de Software

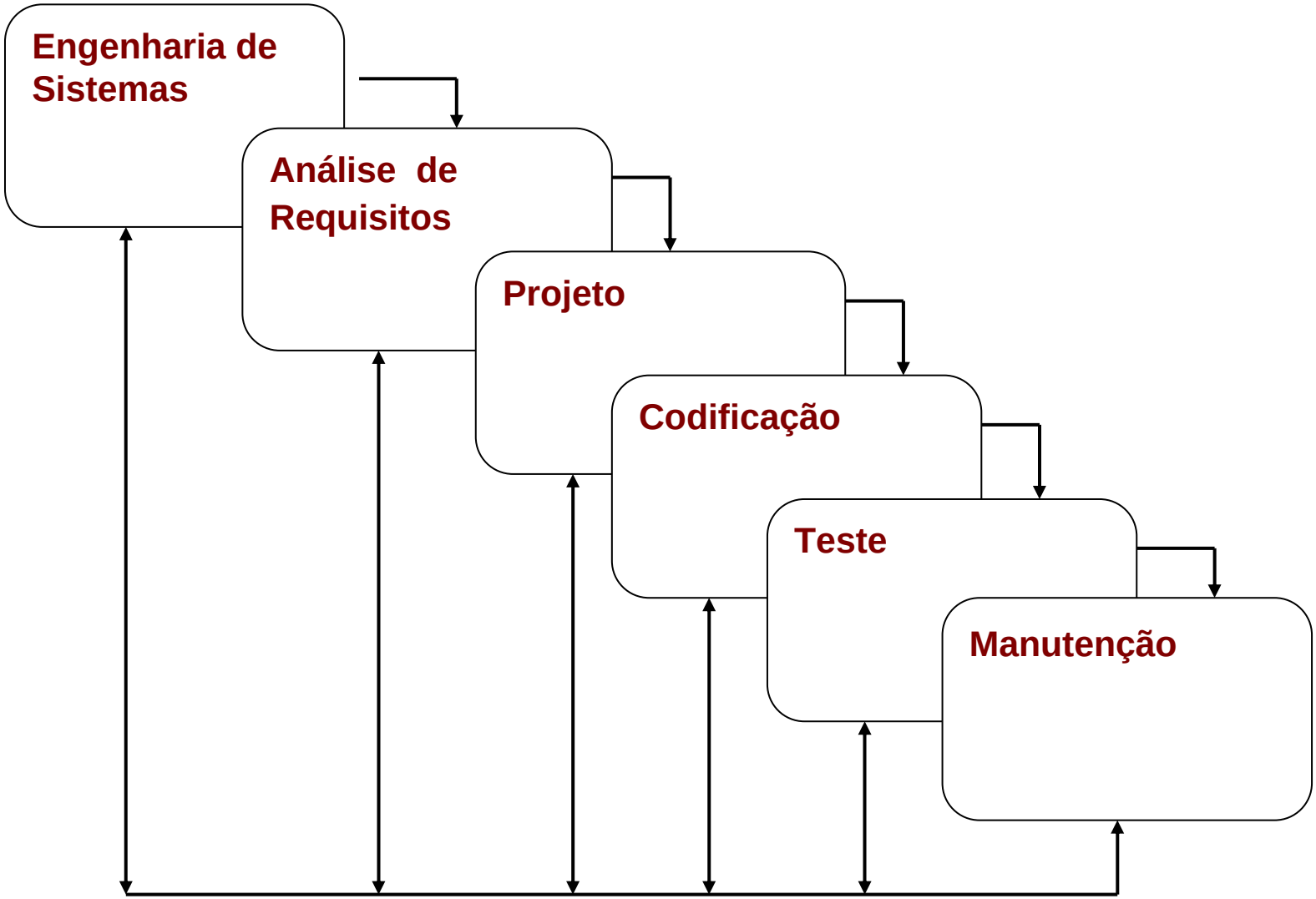
- Estrutura Comum de Processo
  - **definição das atividades de estrutura aplicáveis a todos os projetos de software**
- Conjuntos de Tarefa
  - **adaptação das atividades da estrutura às características do projeto e aos requisitos da equipe**
- Atividades guarda-chuva
  - **apóiam o modelo de processo (garantia de qualidade, gerenciamento de configuração, produção de documentos, etc...)**

Modelos de Processo

1. Modelo Seqüencial (ciclo de vida clássico)
2. Modelo de Prototipação
3. Modelo RAD (Rapid Application Development)
4. Modelos Evolutivos
    1. Modelo Incremental e Espiral
5. Modelo de desenvolvimento Concorrente
6. Modelo de Montagem de Componentes
7. Técnicas de 4a geração
8. Modelo do RUP

Modelo de Ciclo de Vida Clássico (Cascata)

- **modelo mais antigo e o mais amplamente usado da engenharia de software**
- **modelado em função do ciclo da engenharia convencional**
- **requer uma abordagem sistemática e seqüencial para o desenvolvimento de software**
- **cada atividade é uma fase em separado. A passagem entre fases é formal.**



Ciclo de Vida Clássico

Atividades do Ciclo de Vida Clássico

1- ENGENHARIA DE SISTEMAS

- envolve a coleta de requisitos em nível do sistema, com uma pequena quantidade de projeto e análise de alto nível

2- ANÁLISE DE REQUISITOS DE SOFTWARE

- o processo de coleta dos requisitos é intensificado e concentrado especificamente no software
- tarefa de descoberta, refinamento, modelagem e especificação

Atividades do Ciclo de Vida Clássico

3- PROJETO

- tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie
- se concentra em 4 atributos do programa:
  - *Estrutura de Dados*
  - *Arquitetura de Software*
  - *Detalhes Procedimentais*
  - *Caracterização de Interfaces*

Atividades do Ciclo de Vida Clássico

4- CODIFICAÇÃO

- tradução das representações do projeto para uma linguagem artificial resultando em instruções executáveis pelo computador

5- TESTES

- Concentra-se nos aspectos lógicos internos do software, garantindo que todas as instruções tenham sido testadas e nos aspectos funcionais externos, para descobrir erros e garantir que a entrada definida produza resultados que concordem com os esperados.

Atividades do Ciclo de Vida Clássico

6- MANUTENÇÃO

- provavelmente o software deverá sofrer mudanças depois que for entregue ao cliente

- Caracterizada como o "*iceberg*"

Problemas com o Ciclo de Vida Clássico

- projetos reais raramente seguem o fluxo seqüencial que o modelo propõe
- Ilogo no início é difícil estabelecer explicitamente todos os requisitos. No começo dos projetos sempre existe uma incerteza natural
- o cliente deve ter paciência. Uma versão executável do software só fica disponível numa etapa avançada do desenvolvimento
- Iterações e dificuldades para o planejamento e a supervisão
- Congelamento de fases iniciais e comprometimento do atendimento aos requisitos reais do cliente

Vantagens do Ciclo de Vida Clássico

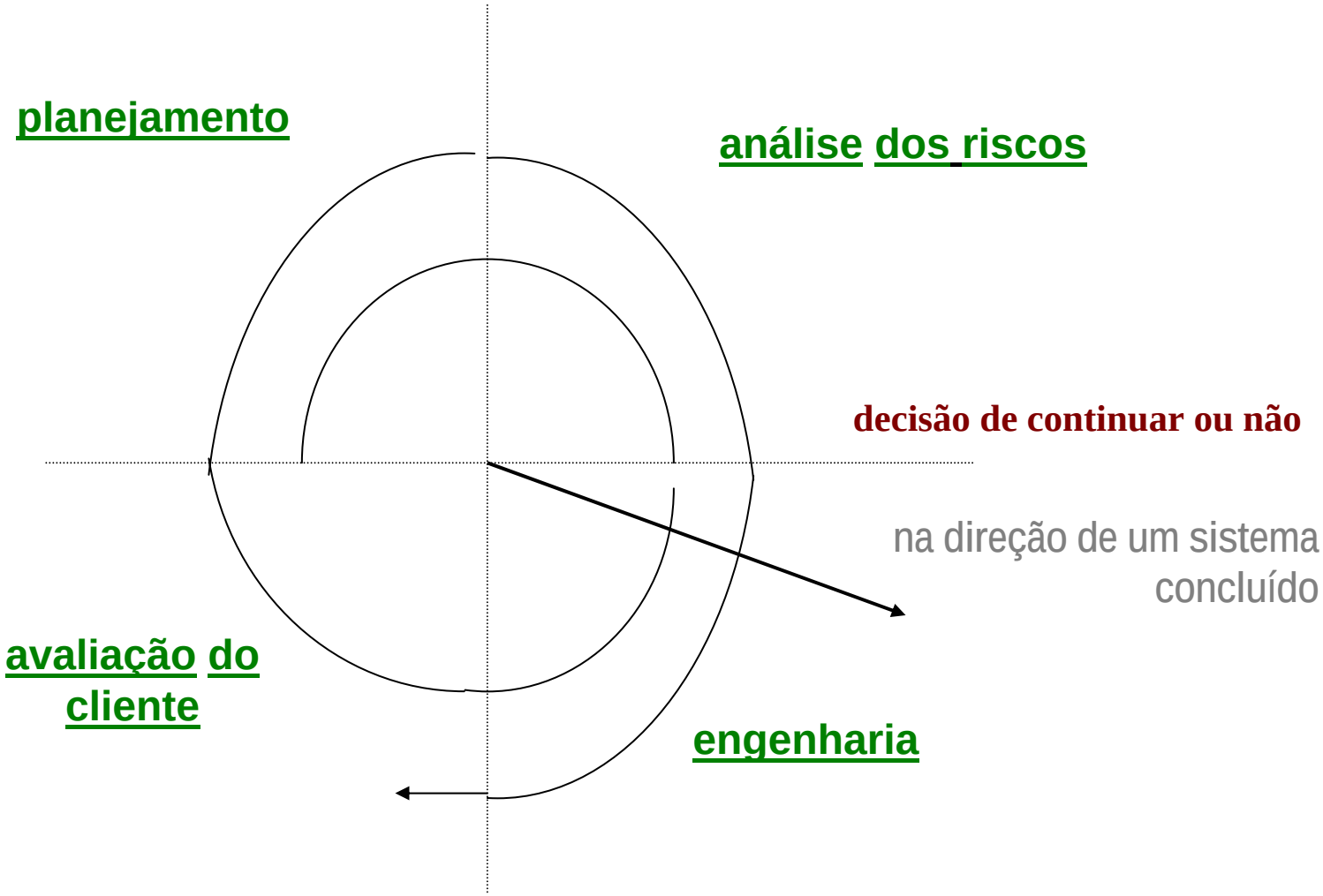
- ✓ **Visibilidade do processo**
- ✓ **Embora o Ciclo de Vida Clássico tenha fragilidades, ele é significativamente melhor do que uma abordagem casual ao desenvolvimento de software**

Ciclo de Vida em Espiral

– Engloba as melhores características do ciclo de vida Clássico como o da Prototipação, adicionando um novo elemento: a *ANÁLISE DOS RISCOS*

– Segue a abordagem de passos sistemáticos do *Ciclo de Vida Clássico* incorporando-os numa estrutura iterativa que reflete mais realisticamente o mundo real

– Usa a *Prototipação,* em qualquer etapa da evolução do produto, como mecanismo de redução de riscos


planejamento
análise dos riscos
decisão de continuar ou não
na direção de um sistema concluído
avaliação do cliente
engenharia

Atividades do Ciclo de Vida em Espiral

1- ***PLANEJAMENTO:*** **determinação dos objetivos, alternativas e restrições**

2- ***ANÁLISE DE RISCO:*** **análise das alternativas e identificação / resolução dos riscos**

3- ***CONSTRUÇÃO:*** **desenvolvimento do produto no nível seguinte**

4- ***AVALIAÇÃO DO CLIENTE:*** **avaliação do produto e planejamento das novas fases**

Comentários sobre o Ciclo de Vida em Espiral

- **É uma abordagem realística para o desenvolvimento de software em grande escala.**
- **usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva.**
- **pode ser difícil convencer os clientes que uma abordagem "evolutiva" é controlável**
- **exige considerável experiência na determinação de riscos e depende dessa experiência para ter sucesso**

Comentários sobre o Ciclo de Vida em Espiral

- A cada iteração ao redor da espiral, versões progressivamente mais completas do software são construídas
- o modelo é relativamente novo e não tem sido amplamente usado