

# Processo de Equipe de Software - TSP

**UNIP - Araraquara**

**Curso: Análise e Desenvolvimento de Sistemas**

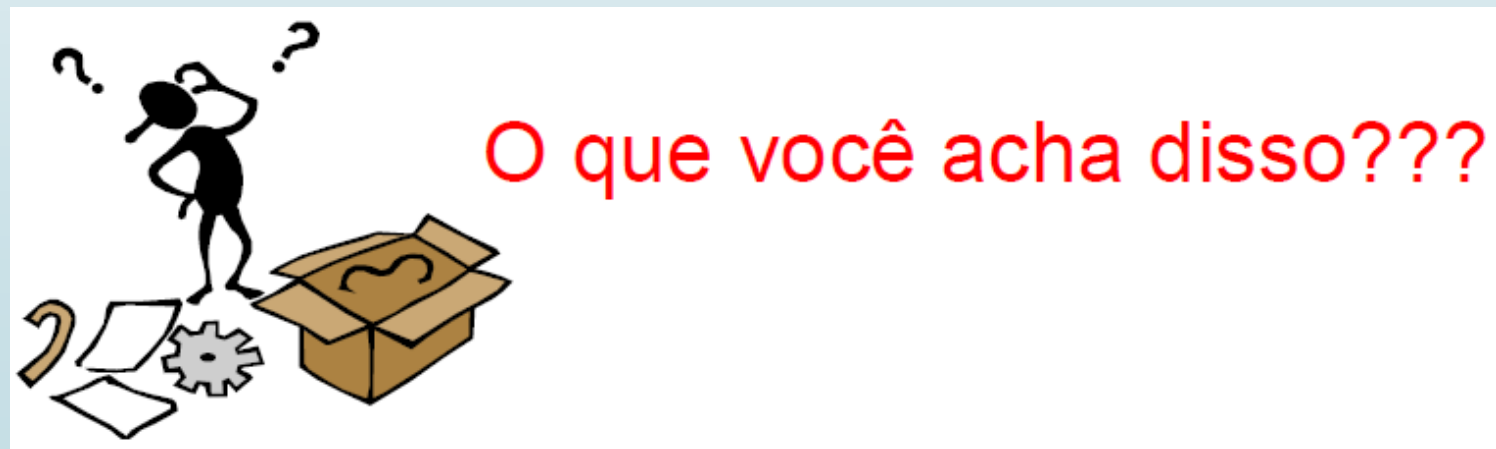
**Disciplina: Engenharia de Software**

**Profº: João Paulo Moreira dos Santos**

# QUALIDADE DE SOFTWARE

- “À medida que os profissionais de desenvolvimento de software aprendem a medir os seus trabalhos, a analisar essas medidas e definir e atingir metas de melhoria, eles passam a enxergar os benefícios de usar o processo definido e são motivados constantemente a utilizá-lo”.

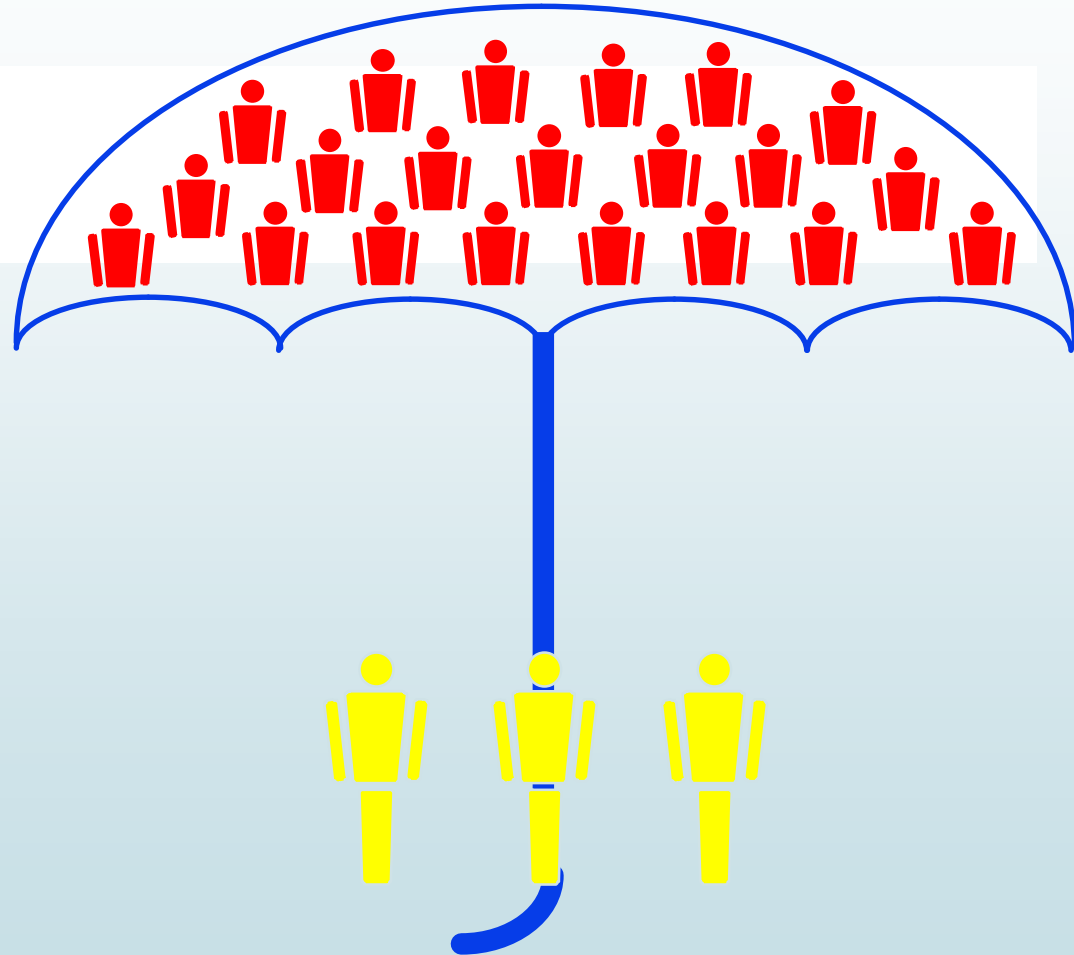
Humphrey



# QUALIDADE DE SOFTWARE

CMM – Foco na  
capacitação  
**Organizacional**

PSP – Foco na  
habilidade  
**Individual** do  
desenvolvedor





# RELACIONAMENTO

- O CMM diz “o que” deve ser feito.
  - Não entra em detalhes de técnicas específicas.
- O PSP diz também “como” deve ser feito.
  - Sugere técnicas e alternativas.



# PSP

- O que o PSP auxilia?
  - Auxilia o desenvolvedor a estimar e planejar suas tarefas, acompanhar sua performance em relação ao planejado e melhorar a qualidade dos produtos produzidos.
  - Foco na melhoria de processos do indivíduo, tornando sua forma de trabalho mais disciplinada.



# PSP

- A qualidade de um software é governada pela qualidade de seus piores componentes.
- A qualidade de um componente de software é governada pelo indivíduo que o desenvolveu.
  - Conhecimento
  - Disciplina
  - Comprometimento



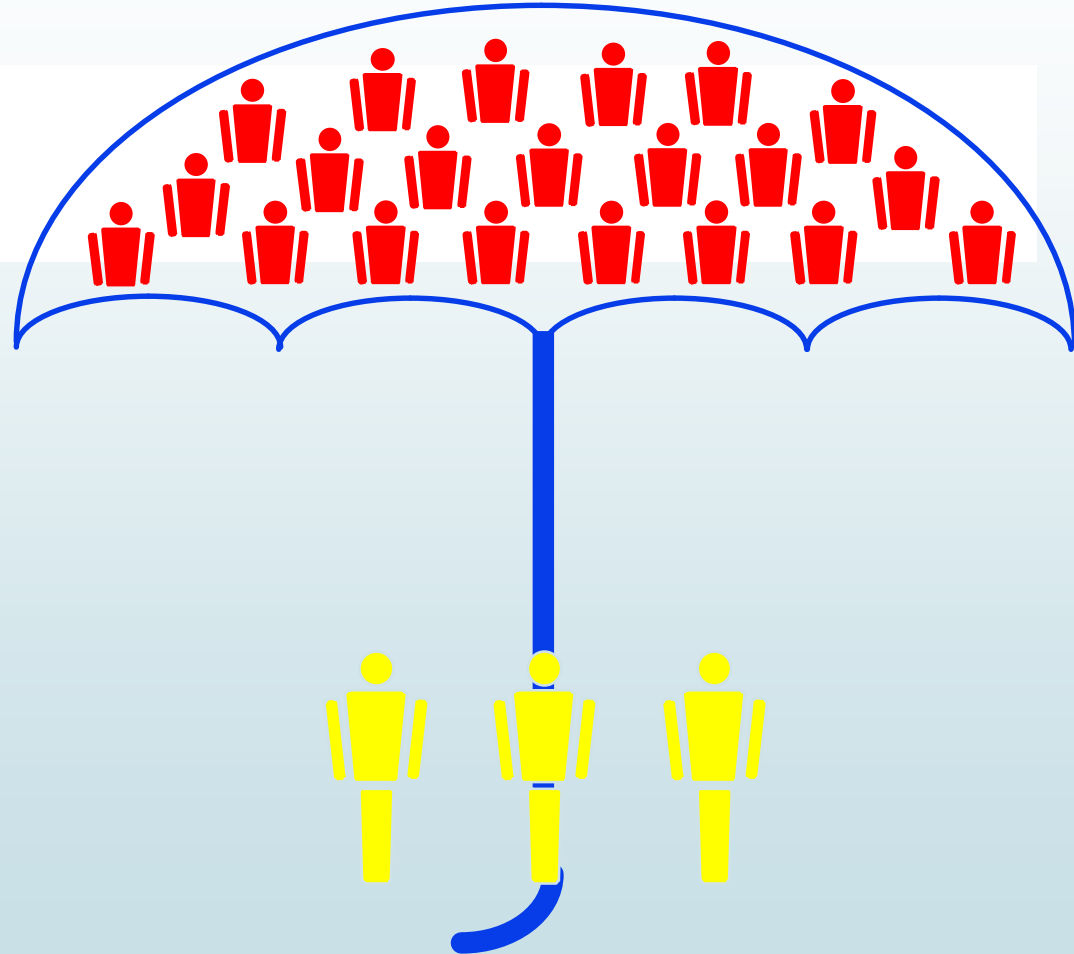
# PSP

- O profissional de software deve conhecer sua própria performance.
  - Medir, acompanhar e analisar seu trabalho.
  - Aprender das variações na performance.
  - Incorporar estas lições em suas práticas pessoais.

# QUALIDADE DE SOFTWARE

CMM – Foco na  
capacitação  
**Organizacional**

PSP – Foco na  
habilidade  
**Individual** do  
desenvolvedor



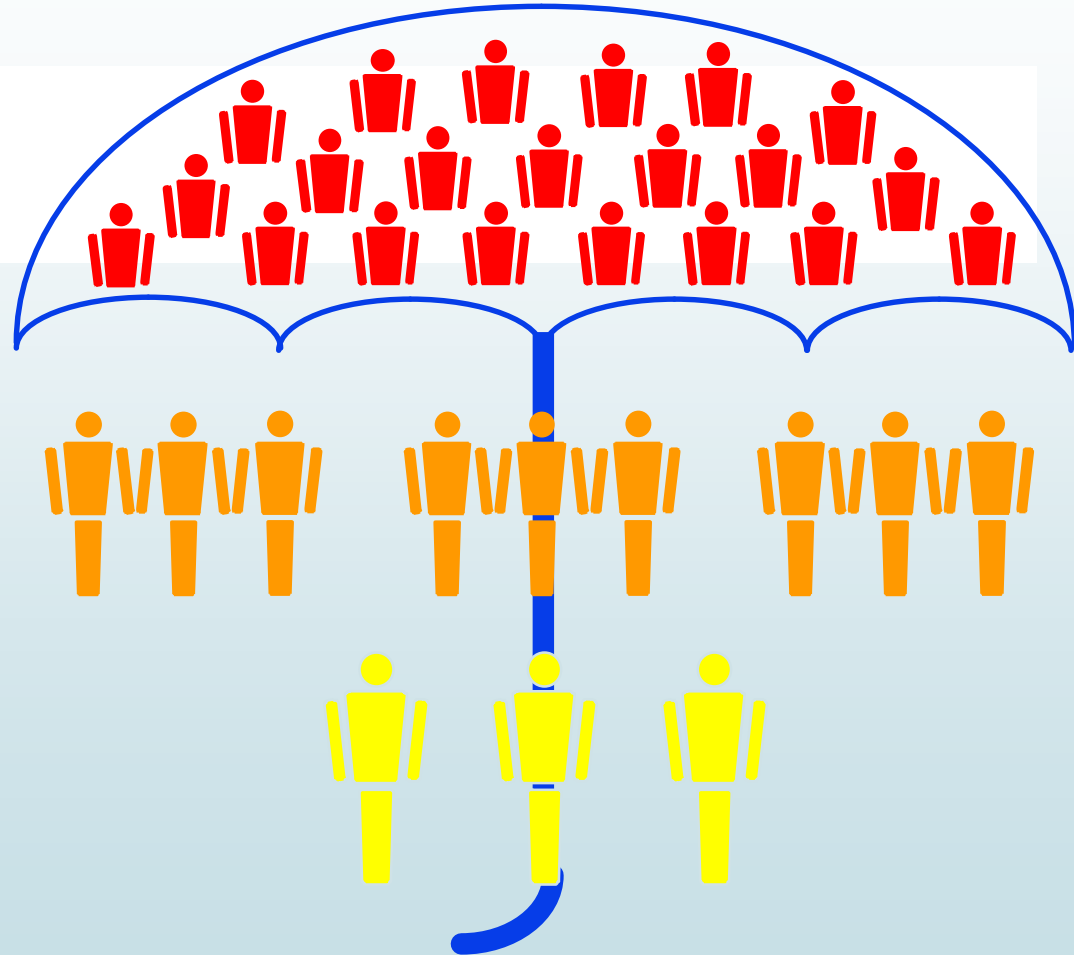


# QUALIDADE DE SOFTWARE

CMM – Foco na  
capacitação  
**Organizacional**

TSP – Foco na  
formação da  
**Equipe** e no seu  
gerenciamento

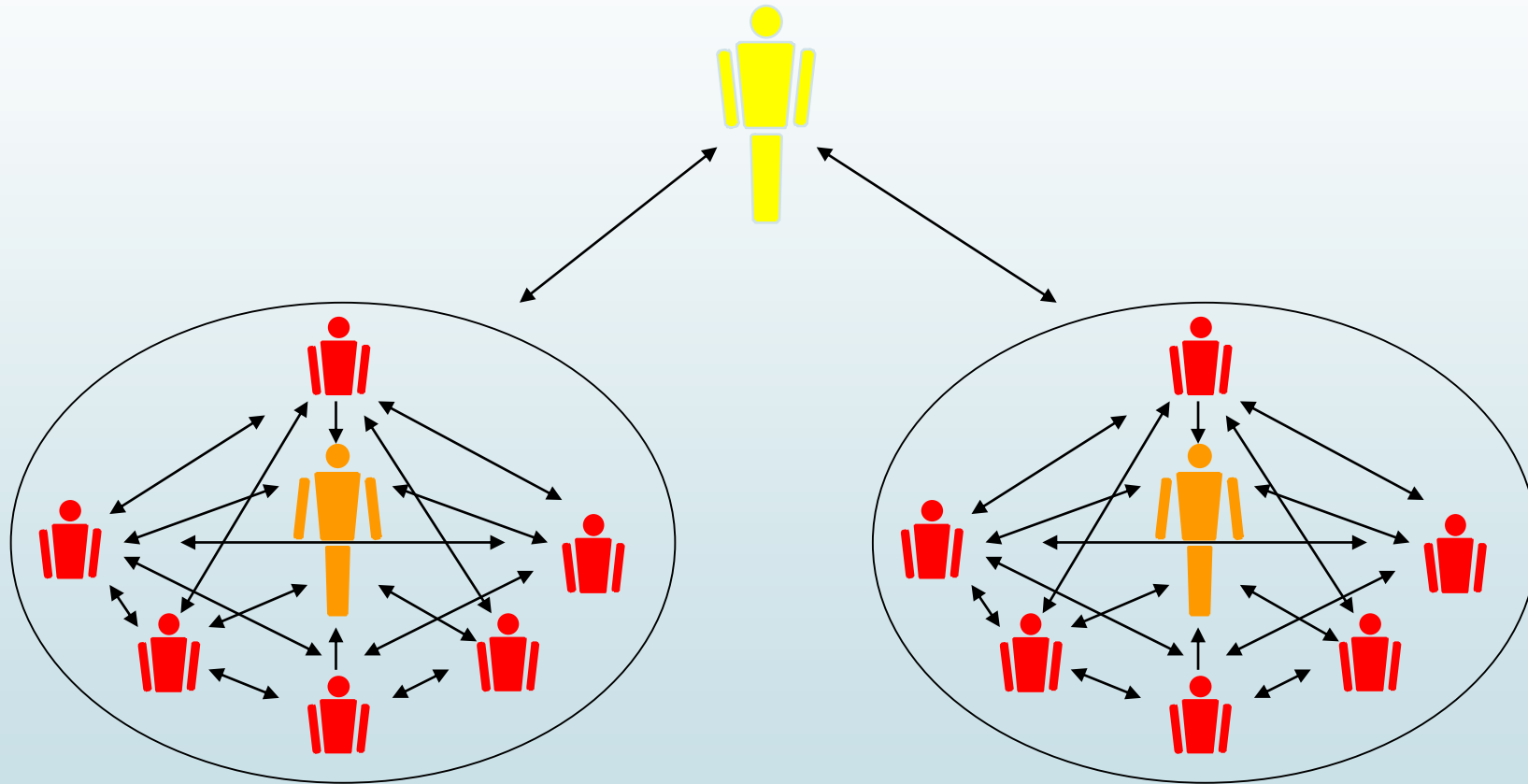
PSP – Foco na  
habilidade  
**Individual** do  
desenvolvedor



# MÉTODOS ORGANIZACIONAIS



# MÉTODOS ORGANIZACIONAIS



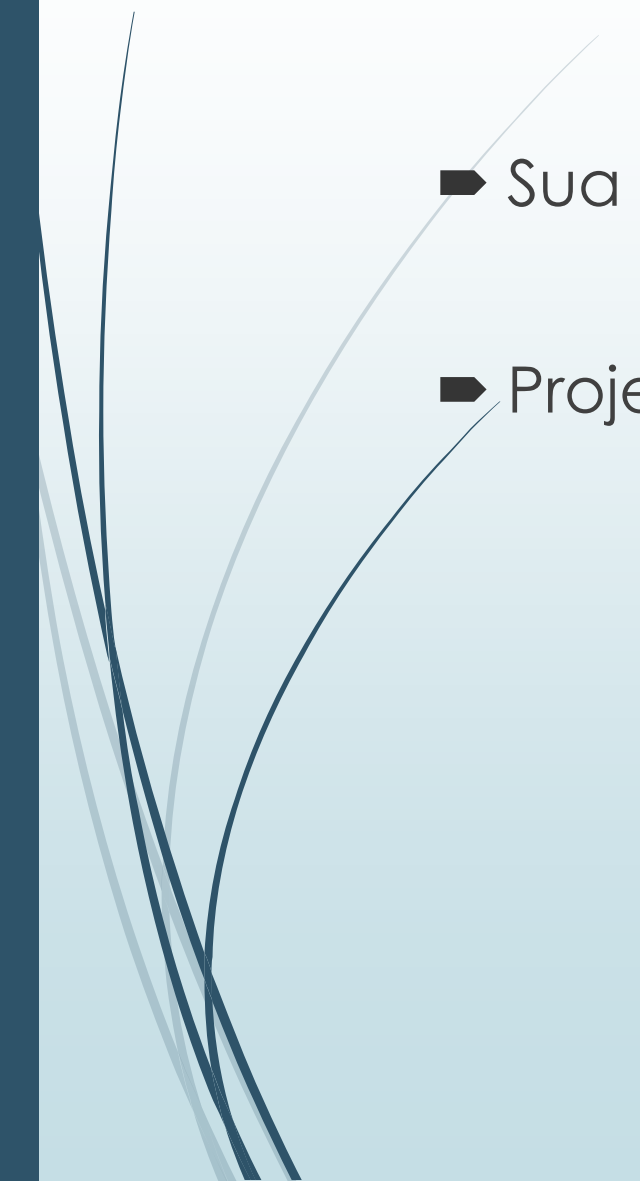


# TEAM SOFTWARE PROCESS

- O TSP (*Team Software Process*) é uma estrutura para a melhoria quantitativa de processo de software que ajuda equipes a desenvolver produtos de software de modo eficaz.
- Baseia-se nos conceitos do CMM.
- Supõe que os membros da equipe tenham sido treinados no PSP.



# TEAM SOFTWARE PROCESS

- Sua primeira versão surgiu em 1996.
  - Projetado por Watts Humphrey.
- 

# TEAM SOFTWARE PROCESS

- O TSP objetiva construir e orientar equipes com a finalidade de fazê-los atingir melhores resultados.





# OBJETIVO

- Maximizar a qualidade de software e minimizar custos.
- Integrar independentes equipes de alta performance que planejam e registram seus objetivos definidos.
- Mostrar aos gerentes como monitorar e motivar suas equipes e ajuda-los a alcançar a produtividade máxima.
- Acelerar a melhoria contínua dos processos.
- Fornecer um guia para melhoria em organizações maduras.



# CONCEITOS E ESTRUTURA

- Equipes auto gerenciadas.
  - A gerência provê orientação e suporte.
  - A equipe planeja o próprio trabalho, acompanha o progresso e gerencia as tarefas do dia-a-dia.
- Cada membro da equipe tem papéis e responsabilidades definidos.
- Todos os membros participam do planejamento do projeto e da tomada de decisões.





# CONCEITOS E ESTRUTURA

- A equipe é proprietária dos seus processos e pode mudá-los sempre que necessário.
- Os processos da equipe são baseados em sua:
  - Experiência
  - Conhecimento
  - Maturidade
- As equipes aplicam práticas do Nível 5 do CMM.

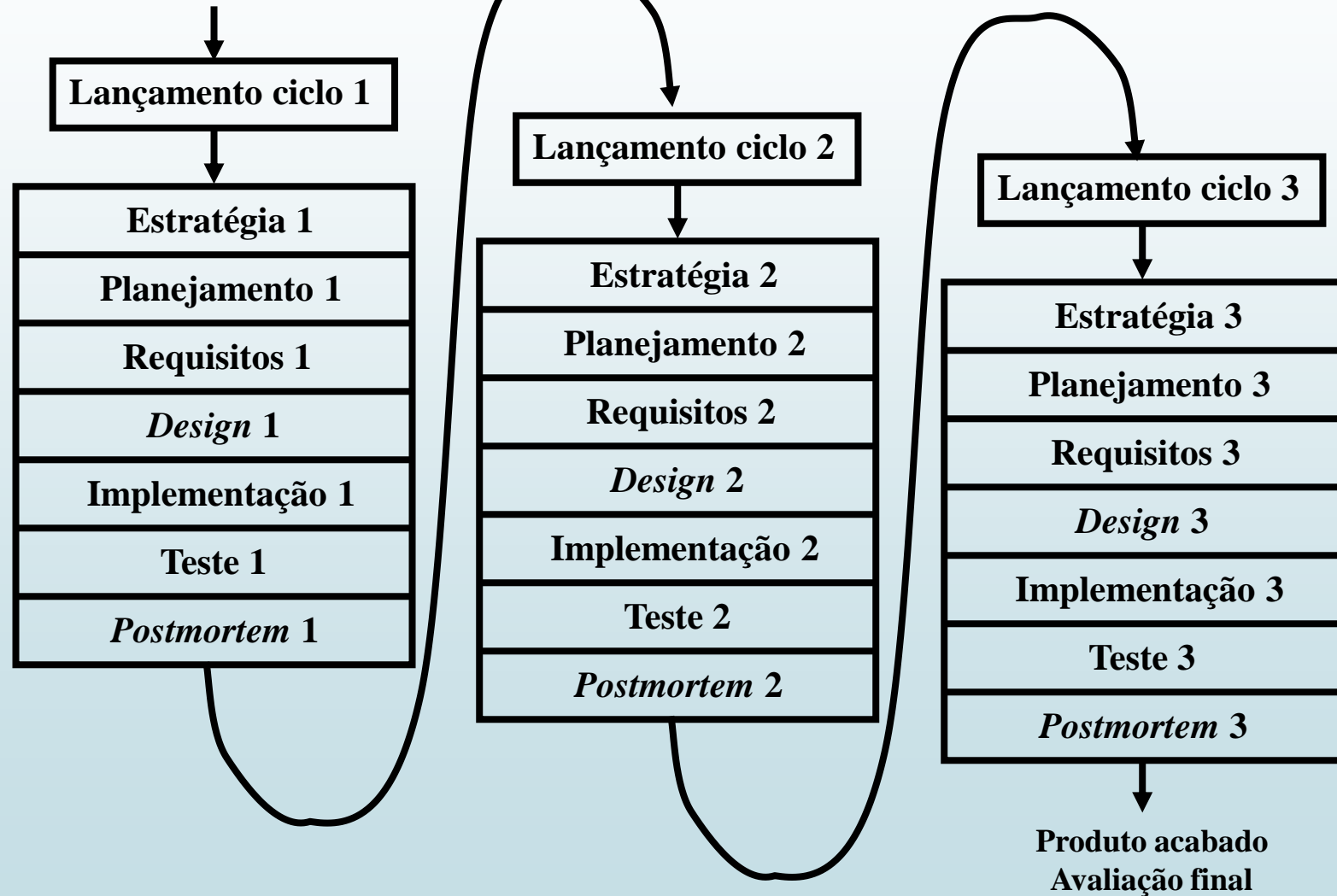


# CONCEITOS E ESTRUTURA

- O TSP provê um conjunto de:
  - *scripts* de processos
  - formulários
  - métodos
  - métricas
- Estes elementos guiam os desenvolvedores em:
  - criar equipes eficazes
  - estabelecer metas e planos para a equipe
  - acompanhar e reportar o trabalho
- TSPi
  - Versão simplificada do TSP para equipes e projetos menores

# ESTRUTURA

Declaração de necessidades do produto





# O PROCESSO DO TSP

- Cada ciclo inclui as seguintes fases:
  - Lançamento (*launch*)
  - Estratégia
  - Planejamento
  - Requisitos
  - *Design*
  - Implementação
  - Teste
  - *Postmortem*
- O processo inclui (Por meio do PSP)
  - *Scripts*
  - Formulários
  - Padrões



# LANÇAMENTO (LAUNCH)

- Por quê o Launch?
- A construção de equipes não ocorre por acaso.
- Um lançamento inicial permite.
  - Estabelecer as relações de trabalho.
  - Definir e distribuir os papéis pelos membros da equipe.
  - Chegar a um acordo sobre as metas da equipe.



# ESTRATÉGIA

- A Estratégia define a ordem na qual as funções do produto serão definidas, desenhadas, implementadas e testadas.
- O processo de desenvolvimento do TSP é cíclico.
  - Cada ciclo produz uma versão operacional do produto.
  - Ciclos subsequentes incrementam a funcionalidade do produto.
  - Este processo é também conhecido como “ciclo de vida incremental”.
  - A equipe decide o conteúdo de cada ciclo ou negocia este conteúdo com o usuário/cliente, com base no prazo e recursos disponíveis.



# PLANEJAMENTO

- Porquê planejar antes de conhecer o produto em detalhes?
  - Ao desenvolver o plano, a equipe adquire uma melhor compreensão comum do trabalho a ser feito.
  - Um plano é a base para acompanhar o trabalho.
  - Sem um plano, a equipe acabará se comprometendo com o prazo imposto pela gerência ou o cliente.
- Por isso a necessidade de iniciar pela Estratégia.



# PLANEJAMENTO

## ■ Com um plano:

- Você trabalha com mais eficiência;
- Você sabe o que fazer e quando;
- Você faz as coisas numa ordem produtiva;
- Você não esquece passos importantes;
- Você tem maior chance de cumprir seus compromissos;
- Você pode assumir compromissos realistas com seus colegas de equipe e com seu cliente;
- Você faz um trabalho melhor, ao não pular, por exemplo, revisões e inspeções, o que levaria a mais tempo gasto em teste e a um produto de baixa qualidade; e
- Você sabe onde está ao longo do desenvolvimento.





# PLANEJAMENTO

## ► Passos do planejamento:

- Liste os produtos a serem desenvolvidos no ciclo e estime seus tamanhos;
- Produza a lista de tarefas;
- Produza o cronograma;
- Produza o plano de qualidade;
- Produza os planos individuais dos desenvolvedores;
- Realize o balanceamento da carga; e
- Produza e distribua o planos.



# PLANEJAMENTO

- Por que balancear os planos?
- Com planos balanceados:
  - Todos os membros da equipe contribuem com esforço igual;
  - Não é necessário esperar pelos outros;
  - Os recursos são usados mais eficientemente; e
  - Consegue-se o menor prazo possível.
- O balanceamento deve ser feito pelos desenvolvedores.
  - São os únicos que podem planejar em detalhes



# FASES DO DESENVOLVIMENTO

- Fases:

- Requisitos.
- *Design*.
- Implementação.
- Teste.

- Estratégias gerais:

- É feito o gerenciamento de configuração de todos os artefatos.
- Todos os artefatos são inspecionados.
- Todos os planos de teste são feitos na fase de desenvolvimento correspondente.



# POSTMORTEM

- Sem uma fase específica para analisar o trabalho feito, pouco se aprende e não se pode fazer melhoria contínua.
- O *postmortem* é uma forma estruturada de aprender e melhorar.
- Compara-se o planejado com o que realmente aconteceu.
- Procura-se oportunidades de melhoria.
  - Mudanças no processo para o próximo projeto ou ciclo.



# PAPÉIS DO TSP

- Por que dividir os papéis?
- Para distribuir a carga de trabalho associada ao desenvolvimento que vão além da construção do produto.
- Para permitir o desenvolvimento de diferentes habilidades pelos desenvolvedores.
- Para explicitar as responsabilidades pelas tarefas.
- Para explicitar a necessidade de tarefas associadas ao desenvolvimento que normalmente são ignoradas pelas equipes.



# PAPÉIS DO TSP

- Escolha dos papéis:
- Cada membro da equipe atua ao mesmo tempo como desenvolvedor e assume um dos papéis do TSP.
- Os papéis devem ser escolhidos/distribuídos
  - Conforme o interesse dos membros da equipe;
  - De acordo com suas habilidades;
- Convém haver rodízio de papéis a cada novo ciclo/projeto.
- Cada pessoa deveria especializar-se em dois ou três papéis.

# FALHAS NOS PROJETOS

- Por que os projetos falham?
- Os projetos falham, geralmente, por causa de problemas no trabalho em equipe (*teamwork*), e não por razões técnicas
- Um dos principais problemas é a dificuldade em lidar com a pressão
  - Tomam-se “atalhos”
  - Usam-se métodos ruins (ou nenhum)
  - Aposta-se em novas linguagens, ferramentas ou técnicas
- O TSP ajuda a lidar com a pressão através da definição da estratégia e do planejamento
  - Permite saber o que é para fazer
  - Permite resistir a cronogramas irrealistas



# EQUIPE

➡ O que é uma equipe?

- (1) Ao menos duas pessoas.
- (2) Trabalhando por um objetivo comum.
- (3) Com cada pessoa assumindo papéis específicos ou funções a desempenhar.
- (4) O atingimento do objetivo requer alguma forma de dependência entre os membros do grupo.



# PROBLEMAS COMUNS NAS EQUIPES

- Liderança ineficaz
  - Abandono dos planos
  - Abandono da disciplina pessoal
- Falta de compromisso ou cooperação
  - Um ou mais membros não querem cooperar trabalhando em equipe
- Falta de participação
  - Um ou mais membros podem não estar dando a contribuição necessária



# PROBLEMAS COMUNS NAS EQUIPES

- Procrastinação e falta de confiança própria
  - Falha em definir objetivos e prazos
  - Resultado de liderança inexperiente, falta de objetivos claros, ou falta de processo e planejamento
- Qualidade pobre
  - Falta de documentação, revisões e inspeções, práticas de implementação pouco rigorosas
- Injeção de requisitos
  - Usuários ou desenvolvedores acrescentando funcionalidade no meio do projeto



# CONDIÇÕES PARA TRABALHO EM EQUIPE

- O trabalho a ser feito é claro e distinto
  - Definido explicitamente
  - Faz sentido para a equipe
  - A equipe sabe o que deve fazer
- A equipe está claramente definida
  - Sabe-se quem está dentro e quem está fora
  - Os membros se conhecem
  - O trabalho de todos é visível
  - Todos sabem os papéis de cada um
- A equipe tem controle sobre a tarefa
  - A equipe controla o processo
  - A equipe é capaz de fazer o trabalho



# COMO O TSP CONSTRÓI EQUIPES

- Propondo um conjunto de objetivos iniciais
  - A cada ciclo, devem ser revistos e ajustados pela equipe
- Identificação antecipada de papéis pré-definidos
  - Distribuição de responsabilidades
- Processo definido para o planejamento
- Comunicação interna
  - Reuniões periódicas
  - Informação disponível (processos, planos, métricas) facilitam a comunicação precisa

# DEVERES NO TRABALHO EM EQUIPE

- Comunicação entre os membros
  - Visibilidade
  - Saber ouvir
  - Negociação
- Estabelecimento e cumprimento de compromissos
  - O compromisso tem que ser livremente assumido
  - O compromisso é público
  - Para assumir responsavelmente um compromisso, é preciso preparação (planejamento)
- Participação nas atividades da equipe
  - Obter a atenção da equipe
  - Pedir e aceitar ajuda



# CONSTRUÇÃO DA EQUIPE

- Para a construção de equipes efetivas, é necessário:
  - Aceitar a responsabilidade por um papel e desempenhá-lo o melhor possível.
  - Participar no estabelecimento de metas e planos da equipe e esforçar-se por cumprir essas metas e seguir o plano.
  - Construir e manter uma equipe efetiva e cooperativa.

# CONCLUSÃO

## ► São quatro as lições do TSP

1. A maior parte do desenvolvimento de software é e será feita por equipes
2. Equipes com as habilidades apropriadas e em que todos os membros trabalham juntos cooperativamente e efetivamente podem produzir resultados extraordinários
3. Um trabalho em equipe efetivo requer oito coisas
  1. Metas da equipe com que todos concordam
  2. Papéis estabelecidos
  3. Um ambiente de trabalho adequado
  4. Um processo de trabalho comum
  5. Um plano para o trabalho
  6. O compromisso mútuo com as metas, papéis e o plano
  7. Comunicação aberta entre todos os membros do time
  8. Respeito mútuo e suporte de todos os membros do grupo
4. Quando times encontram essas condições, produzem um trabalho superior, são mais produtivos e apreciam o seu trabalho