OpenUP: Um processo prático e otimizado para desenvolvimento de pequenos projetos de software.

Palestrante: Antonio Miguel Batista Dourado
antoniodourado@gmail.com
@antoniodourado

Metodologias de Desenvolvimento

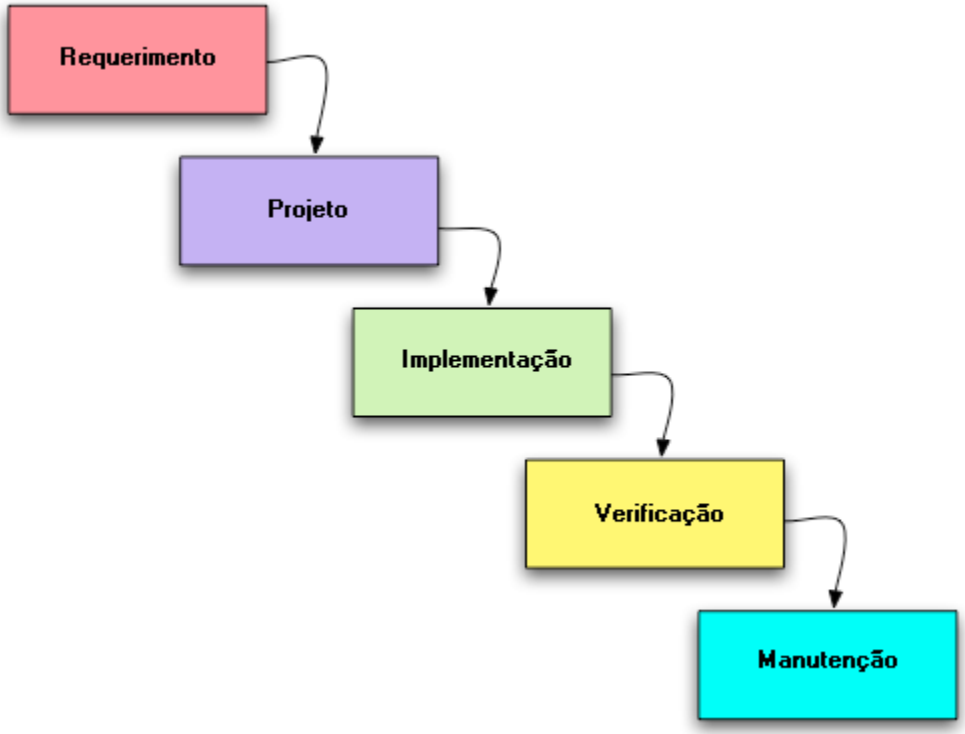
- Representação simplificada do processo.
- Conjunto de práticas recomendadas para o desenvolvimento de software.
- Auxiliar no desenvolvimento de software.
- Metodologias Tradicionais e Metodologias Ágeis.

Metodologias Tradicionais

- Também conhecidas como “orientadas à documentação”.
- Dividem o processo em etapas bem definidas e fechadas.
- Muito utilizada no passado para desenvolvimento em mainframes devido ao alto custo de alterações, resultantes do limitado acesso aos computadores e da falta de tecnologia para depuração e analise de código. (SOARES, 2004)

Metodologias Tradicionais

Um exemplo de metodologia tradicional é o modelo em Cascata:



Metodologias Tradicionais

Problemas:

- Divisão distinta de fases no projeto gera inflexibilidade uma vez que raramente os projetos seguem um fluxo sequencial.
- Requisitos totalmente especificados e “congelados” na primeira fase do projeto dificultam futuras mudanças.
- Arquitetura especificada e “congelada” na segunda fase do projeto torna a arquitetura pouco confiável diante de possíveis mudanças de requisitos.
- Grande dificuldade de alterações no projeto depois de decisões já tomadas.

Metodologias Tradicionais

Metodologias Tradicionais são aconselháveis quando há uma previsibilidade dos requisitos do sistema fazendo com que o projeto seja totalmente planejado e sua gerência seja facilitada. Porém estas metodologias são repletas de burocracias e tem sua eficiência em projetos grandes que não sofram muitas alterações sobre seus requisitos (FAGUNDES, 2005).

Metodologias Ágeis

- Surgiram na década de 90 propondo nova abordagem de desenvolvimento com adaptação à mudanças e apoio à equipe de desenvolvimento.

- Reação às metodologias tradicionais com o intuito de criação de alternativa ao modelo Cascata.

Metodologias Ágeis

- Em 2001, 17 especialistas criaram a Aliança Ágil e através do Manifesto Ágil, popularizou-se o termo “Metodologia Ágil”.

- O Manifesto Ágil busca constantemente melhorias no desenvolvimento ágil e valorizam quatro principais principios.

Metodologias Ágeis

- Indivíduos e interações acima de procedimentos e ferramentas
- Software funcionando acima de documentação abrangente
- Colaboração dos clientes acima de negociação de contratos
- Responder à mudanças acima de um plano pré-estabelecido

Metodologias Ágeis

O Manifesto Ágil não é contra os modelos utilizados pela metodologia tradicional porém não segue os padrões propostos pela mesma.

Não rejeita ferramentas, processos, documentação, contratos ou planejamentos mas prioriza indivíduos e iterações, a executabilidade do software, colaboração e feedbacks.

Metodologias Ágeis

Em resumo, metodologias ágeis são comumente aplicados a pequenos projetos e/ou projetos com baixa complexidade utilizando ciclos iterativos, tolerância à mudança, proximidade da equipe, entre outros.

Exemplos de metodologias ágeis:

-XP

-Scrum

-OpenUP/Basic

Metodologias Ágeis

- Pequenas equipes (entre três e dez pessoas) não comportam a utilização de processos “pesados” como o RUP e/ou não utilizam todos os recursos oferecidos por processos de desenvolvimento como o XP.
- Um mesmo individuo pode ter vários papéis diferentes (RUP).

OpenUP

- Processo Unificado Aberto (Open Unified Process).
- Originalmente chamado de BUP (Basic Unified Process) pela IBM que em 2005 foi liberado para a Fundação Eclipse e renomeado para OpenUP em 2006.
- É parte do EPF (Eclipse Process Framework).

OpenUP

- Aplica uma abordagem iterativa e incremental em um ciclo de vida estruturado, focando na natureza colaborativa do processo de desenvolvimento de software.

- Conjunto compacto de atores, tarefas e artefatos em relação ao RUP.

OpenUP

O OpenUP pode ser definido como:

- *Compacto* – Utiliza apenas conteúdos fundamentais e definidos.
- *Completo* – Abrange todas as fases do ciclo de vida do desenvolvimento de um software.
- *Extensível* – Pode-se utilizá-lo da forma que foi definido mas também é possível adicionar novos conteúdos para atender novas características do projeto.

OpenUP – Princípios

O OpenUP é regido por quatro princípios:

- *Equilibrar as prioridades concorrentes para maximizar o benefício aos Stakeholders*
- *Colaborar para alinhar os interesses e compartilhar o entendimento*
- *Focar na arquitetura, o mais cedo possível, para reduzir o risco e organizar o desenvolvimento*
- *Evoluir para continuamente obter feedback e promover melhorias*

OpenUP – Equilibrar as prioridades concorrentes para maximizar o benefício aos Stakeholders

É preciso ter um conhecimento como um todo sobre as necessidades dos stakeholders e dessa forma, alinhar as prioridades que devem ser de total acordo entre as partes. Além disso, projetar os cenários, casos de uso e escopo do projeto são itens importantes para a definição das prioridades.

OpenUP – Colaborar para alinhar os interesses e compartilhar o entendimento

Em uma equipe cada membro tem seus próprios conhecimentos, habilidades e maneiras de fazer as coisas. Este principio tem a finalidade de alinhar essas diferenças de forma que o projeto seja beneficiado bem como fazer com que todos os membros da equipe tenham um entendimento sobre o projeto. O continuo aprendizado também é estimulado por este principio fazendo com que cada membro da equipe desenvolva mais habilidades e incremente seus conhecimentos.

OpenUP – Focar na arquitetura, o mais cedo possível, para reduzir o risco e organizar o desenvolvimento

Uma arquitetura mal planejada muitas vezes é responsável por grande parte dos problemas de um sistema e até menos por complicações tanto de manutenção quanto do entendimento do mesmo que podem até levar ao fracasso do sistema e consequentemente do projeto.

OpenUP – Focar na arquitetura, o mais cedo possível, para reduzir o risco e organizar o desenvolvimento

O foco na arquitetura tem sua importância dada uma vez que os membros do projeto podem visualizá-la de formas diferentes. Modelos de abstração devem ser usados para evitar este problema.

Além disso, flexibilidade e reuso de recursos são pontos importantes deste principio. O ideal é projetar uma arquitetura onde o acoplamento entre componentes seja baixo, facilitando o reuso dos recursos fornecidos pelos componentes.

OpenUP – Evoluir para continuamente obter feedback e promover melhorias

O objetivo deste principio é fazer com que através de feedbacks, obtenha-se modos de melhorar o produto e também o processo da equipe envolvida. Através de feedbacks, pode-se identificar potenciais riscos e tratá-los mais cedo durante o projeto.

É importante ter o objetivo do projeto de maneira clara ao próprio entendimento para que seja possível fazer uma medição do progresso e identificar possíveis melhorias no processo.

OpenUP – Papéis

Em um projeto de software diferentes pessoas são envolvidas no processo e cada pessoa tem um papel definido no projeto de acordo com seus interesses e habilidades.

Os papéis não tem significado de individualidade sobre alguma tarefa ou artefato, mas sim uma identificação de cada envolvido no projeto quando existe o trabalho em conjunto. Além disso, um papel não limita uma tarefa a apenas um indivíduo quando na verdade uma tarefa pode conter vários indivíduos adicionais e inclusive com outros papéis.

OpenUP – Papéis

OpenUP – Stakeholder

Os Stakeholders são os interessados no resultado do projeto, ou seja, serão suas necessidades que deverão ser satisfeitas.

Geralmente os Stakeholders são pessoas designadas pelo cliente para interagir com a equipe do projeto.

OpenUP – Project Manager

O Gerente de Projeto (Project Manager) é o líder da equipe do projeto. É este papel que tem a responsabilidade de instruir a equipe para conduzí-la a um resultado esperado.

OpenUP – Architect

O Architect (Arquiteto) é o responsável pela definição da arquitetura de software do projeto tomando decisões que orientam tanto equipe de design quanto equipe de implementação.

OpenUP – Analyst

O Analista (Analyst) é o responsável por colher informações dos stakeholders e usuários finais e compreender o problema proposto capturando os requisitos e definindo prioridades para o mesmo.

OpenUP – Developer

O Desenvolvedor (Developer) é o responsável por desenvolver o projeto com base na sua adaptação à arquitetura proposta. Além disso, o desenvolvedor também é responsável pelos testes de desenvolvedor.

OpenUP – Tester

O Tester (Testador) é responsável por toda e qualquer atividade que envolvam testes de software. O Tester deve criar casos de testes, implementá-los e executá-los. Algum conhecimento de codificação também pode ser detido por este papel.

OpenUP – Any Role

O papel Any Role (Qualquer Papel) representa qualquer membro da equipe que possa realizar tarefas gerais como por exemplo criar um caso de teste, implementar códigos no software, etc.

OpenUP – Camadas de Conteúdo

O OpenUP possui duas camadas internas aos papéis chamadas “Camadas de Conteúdo” que tem a finalidade de indicar quais papéis devem interagir entre si a fim de realizar algo.

- Management (Gerência)
- Intent (Inteção)
- Solution (Solução)
- Communication & Collaboration (Comunicação e Colaboração)

OpenUP – Management/Gestão

A camada de Gestão trata do gerenciamento do projeto, incluindo o planejamento do projeto, planejamento da iteração, gerenciamento diário do trabalho durante a iteração e a avaliação da iteração.

Papéis envolvidos: Stakeholder, Gerentes de Projeto e Arquiteto.

OpenUP – Intent/Intenção

A camada de intenção trata como canalizar a intenção dos Stakeholders para o resto da equipe de desenvolvimento, para garantir que construções válidas com capacidades incrementais reflitam as intenções dos Stakeholders.

Papéis envolvidos: Stakeholder, Analista e Testador.

OpenUP – Solution/Solução

A camada de Solução descreve todos os aspectos sobre a criação da arquitetura, o design, a implementação e o teste da aplicação.

Papéis envolvidos: Arquiteto, Desenvolvedor e Testador.

OpenUP – Communication & Collaboration / Comunicação e Colaboração

A camada de comunicação e colaboração é a parte fundamental do OpenUP, refletindo a natureza colaborativa do processo. Ela contém todos os papéis do OpenUP: Stakeholder, analista, desenvolvedor, arquiteto, testador, gerente de projeto e qualquer papel. Os membros da equipe que se prontificam para estes papéis necessitam colaborar para, em conjunto, capturar e definir a intenção dos Stakeholders, desenvolver a solução e gerenciar o projeto.

OpenUP – Tarefas

Uma tarefa corresponde a qualquer unidade de trabalho que um papel possa ser solicitado à executar.

Exemplos:

-Definir um caso de uso.

-Construir o escopo do projeto.

OpenUP – Artefatos

Artefatos são todos e quaisquer resultados de uma tarefa executada por alguém.

Exemplos:

-Documento de caso de uso.

-Documento de escopo.

OpenUP – Disciplinas

Uma disciplina é uma coleção de tarefas que se relacionam a uma "área de interesse" maior em todo o projeto. O agrupamento de tarefas em disciplinas serve principalmente para ajudar a compreender o projeto dentro de uma visão tradicional em cascata. Embora seja comum executar simultaneamente tarefas que pertençam a várias disciplinas (por exemplo, determinadas tarefas de requisitos são executadas sob a mesma coordenação de tarefas de análise e design), separar estas tarefas em disciplinas distintas é uma forma eficaz de organizar o conteúdo, tornando mais fácil a compreensão.

OpenUP – Disciplinas

O OpenUP contém seis disciplinas:

- Requisitos
- Gestão de Projetos
- Arquitetura
- Desenvolvimento
- Teste
- Gestão de Configuração e Mudança

OpenUP – Requisitos

A disciplina de requisitos explica como elicitar, analisar, especificar, validar e gerenciar os requisitos para o sistema a ser desenvolvido.

Tarefas da disciplina:

- Definir visão
- Encontrar e Descrever os Requisitos
- Detalhar os Requisitos

OpenUP – Gestão de Projetos

A disciplina de gestão de projetos explica como instruir, ajudar e suportar a equipe, ajudando-a a lidar com os riscos e obstáculos encontrados quando da construção de software.

Tarefas da disciplina:

- Planejar o projeto
- Planejar a Iteração
- Gerenciar a Iteração
- Avaliar os resultados

OpenUP – Arquitetura

A disciplina de arquitetura explica como criar uma arquitetura, a partir dos requisitos arquiteturalmente significantes. A arquitetura é construída na disciplina de Desenvolvimento.

Tarefas da disciplina:

- Descrever a arquitetura
- Refinar a arquitetura

OpenUP – Desenvolvimento

A disciplina de desenvolvimento explica como projetar e implementar uma solução técnica que seja aderente à arquitetura e atenda aos requisitos.

Tarefas da discplina:

- Projetar a solução
- Implementar a solução
- Implementar os testes de desenvolvedor
- Executar os testes de desenvolvedor

OpenUP – Teste

A disciplina de teste explica como fornecer feedback sobre a maturidade do sistema através do design, implementação, execução e avaliação dos testes.

Tarefas da disciplina:

- Criar os casos de teste
- Implementar os scripts de teste
- Executar os testes

OpenUP – Gestão de Configuração e Mudança

A disciplina de gestão de configuração e mudança explica como controlar as mudanças nos artefatos, assegurando uma evolução sincronizada do conjunto de Produtos de Trabalho que compõem um sistema de software.

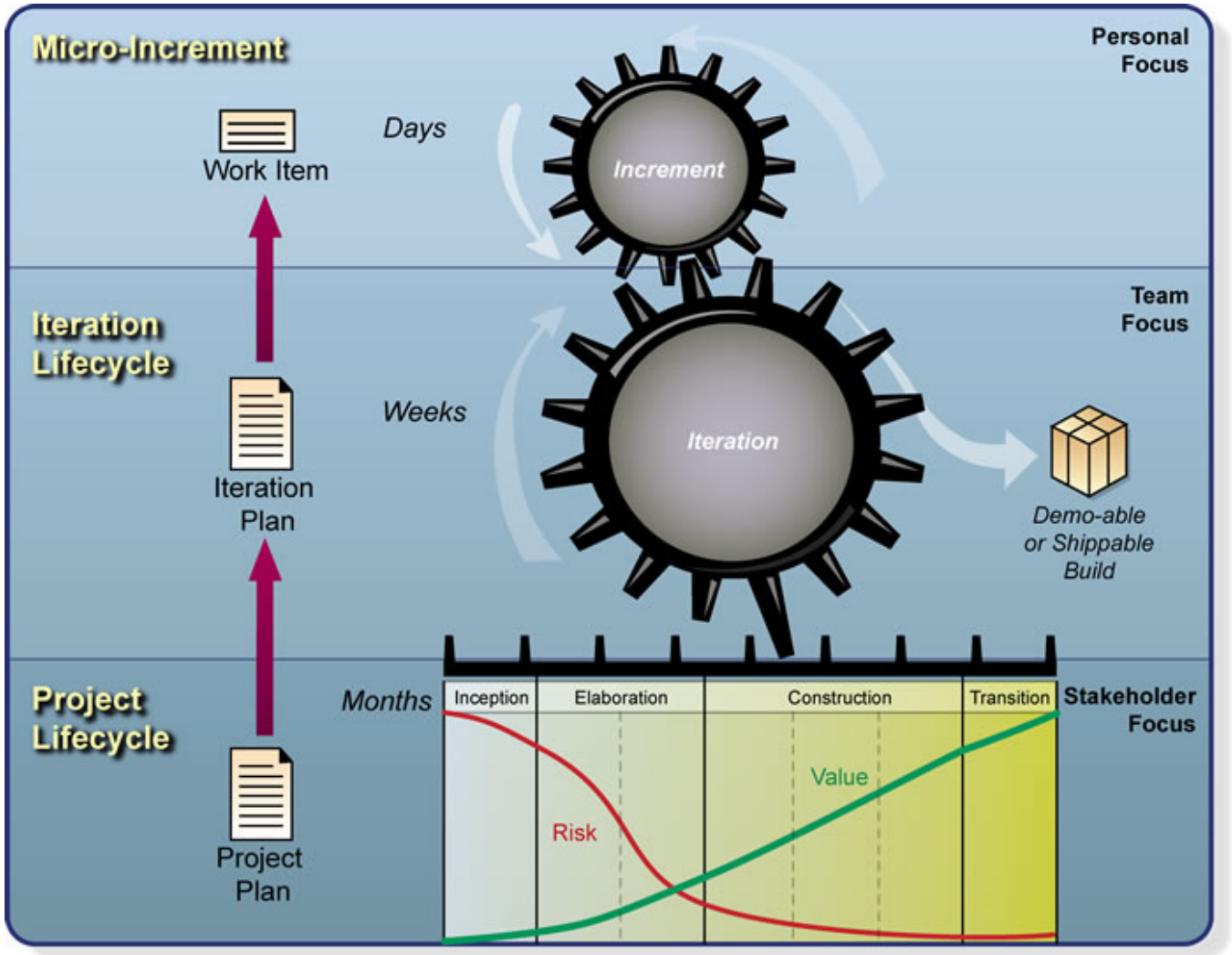
Tarefas da disciplina:

- Solicitar mudança
- Integrar e criar a construção

OpenUP – Camadas

O OpenUP de modo geral conta com três camadas que interagem entre si: Micro-Incrementos, Ciclo de Vida de Iteração e Ciclo de Vida do Projeto.

OpenUP – Camadas



OpenUP – Micro-Incrementos

Os micro-incrementos são considerados pequenos trabalhos que consomem pouco tempo (horas ou dias) e que contribuem para o crescimento da aplicação como um todo.

Exemplos:

- Projetar um caso de uso
- Planejar uma iteração

OpenUP – Micro-Incrementos

Lista de Itens de Trabalho

Lista de todo trabalho agendado para ser feito dentro do projeto, bem como o trabalho proposto que pode afetar o produto neste ou em projetos futuros. Cada Item de trabalho pode conter referências a informações que sejam relevantes para realizar o trabalho descrito no item. Cada item de trabalho corresponde à um micro-incremento.

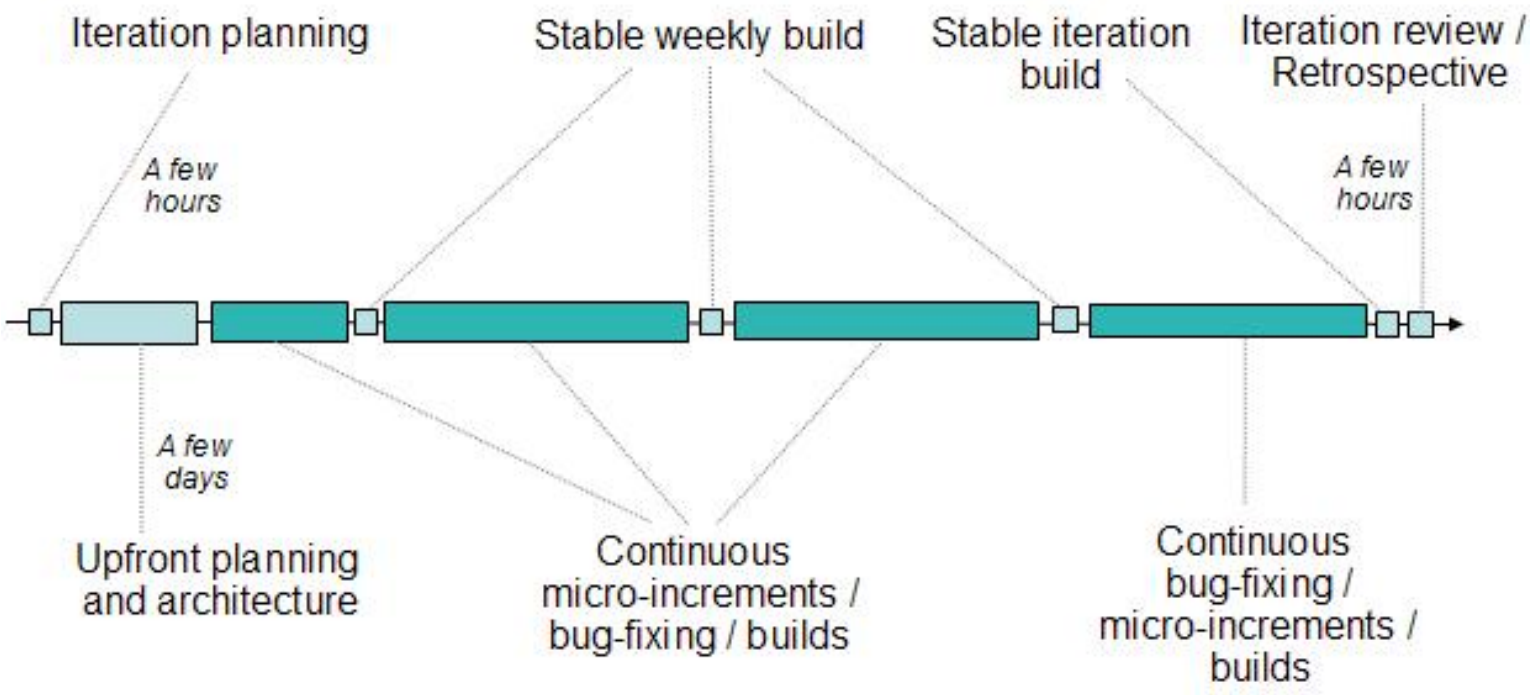
OpenUP – Ciclo de Vida de Iteração

Uma iteração é um período de tempo definido dentro de um projeto em que você produz uma versão estável e executável do produto, junto com toda a documentação de apoio, scripts de instalação, artefatos e similares, necessários para usar a versão.

OpenUP – Ciclo de Vida de Iteração

- O ciclo começa em uma breve reunião de planejamento de iteração, prazos e atividades.
- As iterações em sua maioria são compostas de micro-incrementos mas também correções e ajustes.
- O ciclo termina em uma segunda reunião com os stakeholders onde resultados e melhorias são avaliados.

OpenUP – Ciclo de Vida de Iteração



OpenUP – Ciclo de Vida de Iteração

Plano de Iteração

Os principais objetivos do plano de iteração são fornecer à equipe um lugar central para informações a respeito dos objetivos da iteração, do plano detalhado com as atribuições das tarefas e dos resultados das avaliações. Também ajuda a equipe a monitorar o progresso da iteração e mantém os resultados da avaliação da iteração, que podem ser úteis para melhorar a próxima iteração.

OpenUP – Ciclo de Vida de Projeto

- Dá uma visão geral do projeto a ser desenvolvido bem como o que será realizado, como será realizado e por quem será realizado.

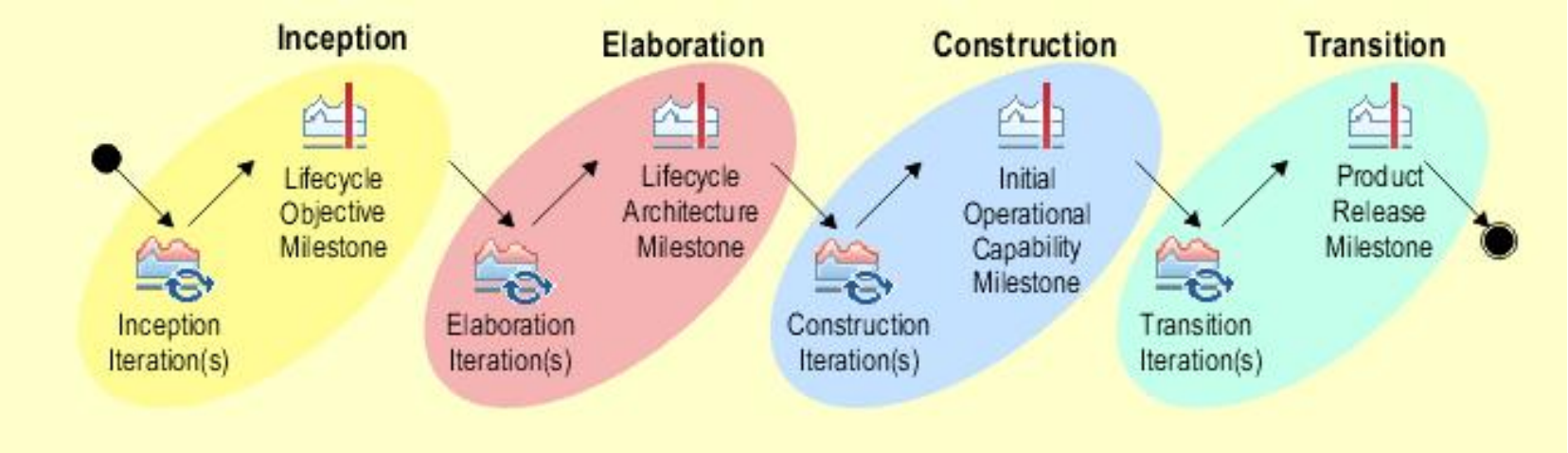
- Pode-se visualizar todas as partes do desenvolvimento de um projeto através das fases (Concepção, Elaboração, Construção e Transição) do ciclo de vida de projeto.

OpenUP – Ciclo de Vida de Projeto / Fases e Marcos

- Uma fase é o tempo entre dois grandes marcos de projeto, durante o qual um conjunto bem definido de objetivos é cumprido, e as decisões são tomadas para entrar ou não na próxima fase.

- Cada fase tem um marco(milestone) ao seu final e através deste, é possível entender o que foi realizado na fase. Os marcos tem como finalidade ser utilizado pelos stakeholders na forma de supervisão do projeto bem como tomadas de decisões sobre o mesmo.

OpenUP – Ciclo de Vida de Projeto



OpenUP – Fase: Concepção

Nesta fase, os stakeholders e os membros da equipe colaboram para determinar o escopo e os objetivos do projeto, e determinar se o projeto deve continuar.

O marco da fase é chamado “Marco dos Objetivos no Ciclo de Vida”. É analisado o custo versus os benefícios, e decide-se entre avançar ou cancelar o projeto.

OpenUP – Fase: Elaboração

Nesta fase, há o entendimento dos requisitos através de um detalhamento, é projetada e implementada uma arquitetura executavel (cenários críticos, interfaces, etc.), riscos essenciais são atenuados e é produzido um cronograma e uma estimativa de custo preciso.

Este marco é chamado de “Marco da Arquitetura no Ciclo de Vida” e é alcançado uma vez que a arquitetura for validada.

OpenUP – Fase: Construção

Esta fase foca o detalhamento dos requisitos, no design, na implementação e no teste da maior parte do software.

O marco desta fase é chamado “Marco da Capacidade Operacional Inicial” e é alcançado quando toda a funcionalidade foi desenvolvida e testes alfa concluída.

OpenUP – Fase: Transição

Esta fase foca a transição do software para o ambiente do cliente e na obtenção da concordância dos Stakeholders de que o desenvolvimento do produto está completo.

O marco desta fase é chamado “Marco de Liberação do Produto” e é alcançado quando os objetivos do projeto foram alcançados e aceito pelos usuários/stakeholders

OpenUP – Ciclo de Vida de Projeto

Plano de Projeto

Define os parâmetros para o acompanhamento do progresso do projeto e especifica os objetivos de alto nível das iterações e seus marcos. Descreve também como o projeto está organizado e quais papéis são executados por quem.

Referências

- FAGUNDES, Priscila Bastos. Framework para Comparação e Análise de Métodos Ágeis. Dissertação de Mestrado. Universidade Federal de Santa Catarina. Florianópolis: 2005.

- SOARES, Michel dos Santos. Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software. 2004. Disponível em: <http://www.dcc.ufla.br/infocomp/artigos/v3.2/art02.pdf> . Acessado em: 09 out 2009.

- FELIPPE, Cecília Macha. Aplicação de Metodologia Ágil no Desenvolvimento de um Componente de Software para Prefeituras. Trabalho de conclusão de curso. Universidade Federal de Santa Catarina. Florianópolis: 2007.

- OPENUP. Apresenta a arquitetura do OpenUP. Disponível em <http://epf.eclipse.org/wikis/openup>. Acessado em: 09 out 2009.

- EPF. Projeto EPF. Disponível em <http://www.eclipse.org/epf/> . Acessado em 10 out 2009.

Obrigado!

Antonio Miguel Batista Dourado
antoniodourado@gmail.com
@antoniodourado