Premissas:

Empresa tem um histórico de vida e trabalha com desenvolvimento de software têm anos, já implementou RUP e agora com a proposta de desenvolvimento novo software, vai adaptar seus processos para se adequar e garantir a qualidade do produto em todas as suas fases. Devido a esse histórico, alguns pontos relacionados aos procedimentos para gestão de riscos, mudanças, e padrões de desenvolvimento, já são pré-estabelecidos.

São 4 fases, concepção, elaboração, construção e transição, pertencentes ao Rational Unified Process e antes de iniciar o processo é preciso ressaltar o objetivo de cada uma delas, sendo eles:

**Concepção -** Abrange as tarefas de comunicação com o cliente e planejamento. É feito um plano de projeto avaliando os possíveis riscos, as estimativas de custo e prazos, estabelecendo as prioridades, levantamento dos requisitos do sistema e preliminarmente analisá-lo. Assim, haverá um assentimento dos stakeholders na definição do escopo do projeto, onde são examinados os objetivos para se decidir sobre a continuidade do desenvolvimento.

**Elaboração -** Abrange a modelagem genérica do processo. O objetivo desta fase é analisar de forma mais detalhada o domínio do problema, revisando os riscos que o projeto pode sofrer e construindo sua arquitetura.

**Construção -** Desenvolve os componentes de software. O principal objetivo desta fase é a construção do sistema de software, com foco no desenvolvimento de componentes e outros recursos do sistema. É na fase de construção que cerca de 80% de codificação ocorre.

**Transição** - Engloba a entrega do software ao usuário e a fase de testes. O objetivo desta fase é disponibilizar o sistema, tornando-o disponível e compreendido pelo cliente. As atividades desta fase incluem o treinamento dos usuários finais e também a realização de testes da versão beta do sistema visando garantir que o mesmo possua o nível adequado de qualidade.

**Plano de iteração**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Disciplinas | Concepção | Elaboração | Construção | Transição |
| Modelagem de negócios | Definir e ajustar objetivos; Visão de negócio  Detalhar uma entidade de negócio; Entidade de negócio e organização  Estruturar o modelo de caso de uso de negócio; Modelo de caso de uso de negócio  Definir arquitetura do negócio; Documento de arquitetura de negócio  Definir requisitos de automação;  Modelo de análise e caso de uso | - | Revisar o modelo de caso de uso de negócio; Documento de revisão | - |
| Requisitos | Capturar um vocabulário comum; Glossário  Detalhar o requisito de software; Atributos e especificações dos requisitos  Estruturar o modelo de caso de uso;  Modelo de caso de uso e caso de uso  Elencar solicitações dos clientes; Requisições do cliente | Detalhar o caso de uso;  Caso de uso e atributos  Gerenciar dependências; Plano de gerenciamento de requisitos e visão do negócio  Revisar requisitos; documento de revisão  Prototipar interface com usuário;  Protótipo de interface | - | - |
| Análise e design | Projeto de caso de uso; Caso de uso e modelo de design | Projetar banco de dados; modelo de dados  Projeto de classes;  Design de classes  Analisar viabilidade de prova de conceito da arquitetura; lista de riscos, visão do projeto  Descrever arquitetura em tempo de execução;  Documento de arquitetura de software  Revisar arquitetura; mudanças e documento de revisão | Construir prova de conceito da arquitetura; Prova de conceito da arquitetura | - |
| Implementação | - | Planejar integração do sistema;  Plano de construção da integração  Estruturar o modelo de implementação;  Documento de arquitetura de software e modelo de implementação  Implementar componente;  Componente e teste de componente  Integrar sistema;  Código executável  Revisar código;  Documento de revisão | Implementar componente;  Componente e teste de componente  Integrar subsistemas;  Subsistema executável  Integrar sistema;  Código executável  Revisar código;  Documento de revisão  Corrigir defeitos;  Componente  Executar teste unitário;  Resultado de teste e componente | Implementar componente;  Componente e teste de componente  Executar teste unitário;  Resultado de teste e componente  Revisar código;  Documento de revisão  Integrar sistema;  Código executável  Corrigir defeitos;  Componente |
| Teste | Definir necessidades de rastreabilidade;  Plano de teste | Identificar alvos do teste;  Plano de teste  Definir elementos de testabilidade;  Mudanças, script de teste, componente de teste, mudanças  Definir detalhes dos testes;  Caso de teste e informações de teste | Implementar teste;  Log de teste e script de teste  Analisar falha no teste;  Mudanças, script de teste  Definir detalhe do teste;  Caso de teste e informações de teste | Implementar teste beta;  Resultado do teste  Implementar teste;  Log de teste e script de teste  Analisar falha no teste;  Mudanças, script de teste  Definir detalhe do teste;  Caso de teste e informações de teste |
| Implantação | - | Criar arte do produto;  Arte do produto  Escrever notas de release;  Notas do release | Desenvolver materiais de suporte;  Material de suporte ao usuário  Escrever notas de release;  Notas de release  Desenvolver plano de implantação;  Plano de implantação | Gerenciar o teste de aceitação;  Mudanças e infraestrutura de implantação  Verificar produto manufaturado;  Produto pronto  Escrever notas de release;  Notas de release |
| Gerência de configuração e mudança | Escrever plano de gerência de configuração;  Plano de configuração | Setup ambiente de gerência de configuração;  Repositório do projeto  Fazer mudanças;  Alterações do ambiente do projeto  Criar baselines;  baseline | Fazer mudanças;  Alterações do ambiente do projeto  Criar baselines;  baseline | Fazer mudanças;  Alterações do ambiente do projeto  Criar baselines;  baseline |
| Gerência de projeto | Planejar fases e iterações;  Plano de desenvolvimento do software  Definir a organização do projeto e da equipe;  Plano de custo e de desenvolvimento  Desenvolver plano de qualidade assegurada;  Plano de qualidade assegurada  Desenvolver o plano de iteração;  Plano de iteração  Monitorar status do projeto;  Medições do projeto, lista de riscos e não conformidades  Definir o caso de negócio;  Caso de negócio  Desenvolver o plano de gerenciamento de riscos;  Plano de gerenciamento de risco  Identificar e analisar riscos;  Lista de riscos  Analisar iteração;  Mudanças  Status report;  Análise do status | Elaborar plano de desenvolvimento do software;  Plano de desenvolvimento de software  Desenvolver plano de indicadores do projeto;  Indicadores  Revisão dos marcos do ciclo de vida;  Documento de revisão  Monitorar status do projeto;  Medições do projeto, lista de riscos e não conformidades  Analisar iteração;  Mudanças  Status report;  Análise do status  Identificar e analisar riscos;  Lista de riscos  Tratar exceções e problemas;  Mudanças  Desenvolver o plano de iteração;  Plano de iteração | Monitorar status do projeto;  Medições do projeto, lista de riscos e não conformidades  Analisar iteração;  Mudanças  Status report;  Análise do status  Identificar e analisar riscos;  Lista de riscos  Tratar exceções e problemas;  Mudanças  Desenvolver o plano de iteração;  Plano de iteração | Revisão da aprovação do projeto;  Documento de aprovação  Preparar para fechamento do projeto;  Plano de implantação de verificação final de qualidade  Revisão da aceitação do projeto;  Documento de aceitação  Monitorar status do projeto;  Medições do projeto, lista de riscos e não conformidades  Analisar iteração;  Mudanças  Status report;  Análise do status  Identificar e analisar riscos;  Lista de riscos  Tratar exceções e problemas;  Mudanças  Desenvolver o plano de iteração;  Plano de iteração |
| Ambiente | Selecionar e adquirir (caso necessário) ferramentas;  Ferramentas | Desenvolver manual de estilo;  Manual de estilo | Suporte ao desenvolvimento;  Infraestrutura de desenvolvimento  Verificar configuração e instalação das ferramentas  Mudanças | - |

**Entregáveis de cada iteração (amarelos acima)**

**Critérios de avaliação ao término de uma iteração**

**Ao final de uma interação deve-se ser analisado o plano de qualidade definido: prazo, custo e documentos predefinidos. Além desses fatores deve ser entregue um programa executável, exceto na fase de concepção, que compreenda uma série de requisitos elencados pelo cliente. Uma iteração pode ser considerada boa quando o plano de qualidade é atendido e seguem as especificações funcionais e não funcionais solicitadas.**

**Atividades e papéis**

|  |  |
| --- | --- |
| Papéis | Atividades |
| Analista de processo de negócio | Definir e ajustar objetivos;  Detalhar uma entidade de negócio;  Estruturar o modelo de caso de uso de negócio;  Definir arquitetura do negócio;  Definir requisitos de automação; |
| Revisor do modelo de negócio | Revisar o modelo de caso de uso de negócio; |
| Analista do sistema | Capturar um vocabulário comum;  Estruturar o modelo de caso de uso;  Elencar solicitações dos clientes;  Gerenciar dependências;  Revisar requisitos; |
| Especificador de requisitos | Detalhar o requisito de software;  Detalhar o caso de uso; |
| Projetista de interface do usuário | Prototipar interface com usuário; |
| Projetista | Projeto de caso de uso;  Projeto de classes; |
| Projetista de banco de dados | Projetar banco de dados; |
| Arquiteto de software | Analisar viabilidade de prova de conceito da arquitetura;  Descrever arquitetura em tempo de execução;  Construir prova de conceito da arquitetura;  Estruturar o modelo de implementação; |
| Revisor de arquitetura; | Revisar arquitetura; |
| Integrador | Planejar integração do sistema;  Integrar subsistemas;  Integrar sistema; |
| Implementador | Implementar componente;  Corrigir defeitos;  Executar teste unitário; |
| Revisor de código | Revisar código; |
| Analista de testes | Definir necessidades de rastreabilidade;  Identificar alvos do teste; |
| Projetista de testes | Definir elementos de testabilidade;  Definir detalhes dos testes; |
| Testador | Implementar teste;  Analisar falha no teste;  Implementar teste beta; |
| Artista gráfico | Criar arte do produto; |
| Gerente de implantação | Escrever notas de release;  Desenvolver plano de implantação;  Gerenciar o teste de aceitação;  Verificar produto manufaturado; |
| Escritor técnico | Desenvolver materiais de suporte;  Desenvolver manual de estilo; |
| Gerente de configuração | Escrever plano de gerência de configuração;  Setup ambiente de gerência de configuração; |
| Qualquer funcionário | Fazer mudanças; |
| Gerente de projeto | Planejar fases e iterações;  Definir a organização do projeto e da equipe;  Desenvolver plano de qualidade assegurada;  Desenvolver o plano de iteração;  Monitorar status do projeto;  Definir o caso de negócio;  Desenvolver o plano de gerenciamento de riscos;  Identificar e analisar riscos;  Analisar iteração;  Status report;  Desenvolver plano de indicadores do projeto;  Preparar para fechamento do projeto; |
| Revisor de projeto | Elaborar plano de desenvolvimento do software;  Revisão dos marcos do ciclo de vida;  Tratar exceções e problemas;  Revisão da aprovação do projeto;  Revisão da aceitação do projeto; |
| Especialista em ferramentas | Selecionar e adquirir ferramentas;  Verificar configuração e instalação das ferramentas; |
| Administrador do sistema | Suporte ao desenvolvimento; |

**Práticas do XP**

São muitas as práticas do Extreme Programming, algumas delas já são utilizadas no RUP, como “versões pequenas” que se trata dos entregáveis ou releases, “jogo do planejamento” que atualiza e revisa o plano em todas as iterações, mantendo o controle do projeto, “teste” em que muitos quesitos são respeitados como teste em cada fase, testes unitários e teste de aceitação com a única exceção do TDD (Test-Driven Design), “integração contínua”, “cliente presente”.

Então analisando as que já são implementadas, foi decidido englobar as que não são, dessa forma temos:

Prática do projeto simples - o projeto deve ser organizado e flexível para que mudanças sejam realizadas de forma barata, simples e rápida. Portanto ele não pode ter duplicação de funcionalidade, deve passar nos testes e ter o menor número possível de classes e métodos.

Prática de padrão de codificação – todos tem acesso ao código e todos o refatoram. Isso implica em definir um padrão de codificação logo no início do projeto. Trazendo não somente um entendedor e responsável por parte de algoritmo.

Metáfora – é uma linguagem verbal que todos os internos são familiarizados, então ao invés de descrever detalhe a detalhe como uma arquitetura funciona, por exemplo, apenas ao comunicar o seu nome todos os integrantes compreendem do que se trata.

**Manifesto Ágil**

A junção entre os métodos acaba se cruzando em muitos pontos como mostrado anteriormente, o manifesto ágil se baseia em valores que são claros, mas põe em prioridade outros quesitos, indivíduos e interações mais que processos e ferramentas, software em funcionamento mais que documentação abrangente, colaboração com o cliente mais que negociação de contratos e responder a mudanças mais que seguir um plano.

Indivíduos e interações evidencia que o desenvolvimento é primordialmente desenvolvido por humanos, então deve-se resolver os problemas crônicos de comunicação. Enquanto processos e ferramentas são importantes, mas também simples e úteis.

Software em funcionamento coloca em cheque que o objetivo central é ter um produto que seja funcional. A documentação é necessária, porém somente quando agrega valor, e não quando o é.

Colaboração com o cliente foca no fato em que o cliente e a empresa devem ter comunicação clara e amigável, gerando interação e a possibilidade de tomada de decisão em conjunto e trabalho em equipe.

Responder a mudanças engloba que é necessário saber receber feedbacks e atualizar o plano no caso de alterações, aprendendo com erros e adaptando processos.

Fere o manifesto?

O manifesto ágil possui os seus 12 princípios que são muito parecidos com a junção dos objetivos do RUP e também do XP, esse último tem o mesmo criador do Manifesto Ágil, o Kent Beck, entre outros membros.

* 1. 4. Elaborar uma justificativa de negócio para o seu processo de desenvolvimento de um novo software: