3
Formulação da Metodologia

3.1.
Considerações Iniciais

O presente capítulo tem como finalidade propor e descrever um modelo de referencia para o gerenciamento de projetos de *software* que possa ser mensurável e repetível, considerando os conceitos gerenciamento de projetos, e as principais metodologias ágeis estudadas no capítulo 2.

De acordo com Martins (2007), uma metodologia tem micro e macro componentes, onde os macro componentes são os encarregados de definir o fluxo geral e a sequência de trabalho, enquanto os micro componentes incluem as regras gerais de projeto técnico (*design*), padrões (*patterns*), e boas práticas. Para o caso específico do desenvolvimento de *software*, as regras gerais de design apresentam propriedades para seguir ou evitar, e abordagens gerais que devem ser adotadas para a construção do sistema. Os padrões são soluções que podem ser aplicadas a certo tipo de atividade de desenvolvimento; são soluções padronizadas para certos tipos de problemas. As boas práticas consistem em orientações e dicas que guiaram o bom desenvolvimento dos projetos.

3.2.
Proposta de Modelo de Gerenciamento de Projetos de *Software*

O modelo de referência proposto está dividido em Macro e Micro componentes (MARTINS, 2007). A Figura 3.1 descreve um mapeamento do modelo, onde os macro componentes estão compostos pelos cinco grupos de processos que PMBOK recomenda, os quais definirão o fluxo geral e a sequência de trabalho, e os microcomponentes estão compostos pelas diferentes práticas de gerenciamento do Scrum, FDD e as boas práticas de desenvolvimento de *software* Extreme Programming, produção enxuta de *software* e CMM, com o objetivo de ter um projeto de *software* gerenciado que possa ser adaptado ou padronizado para diversos projetos, com o qual se terá como resultado um cliente satisfeito, e a criação de um repositório de conhecimentos para a empresa.

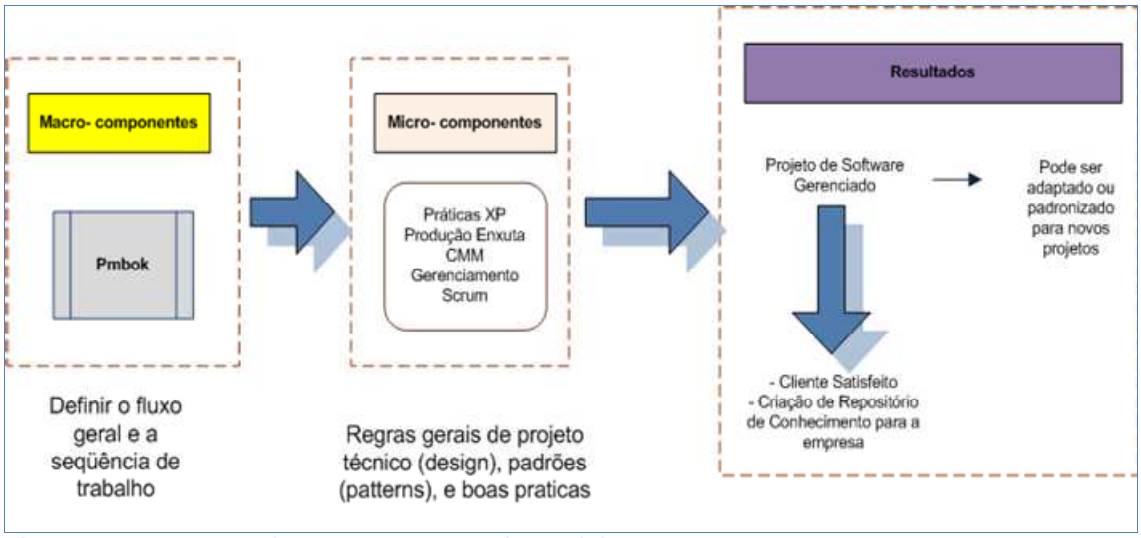
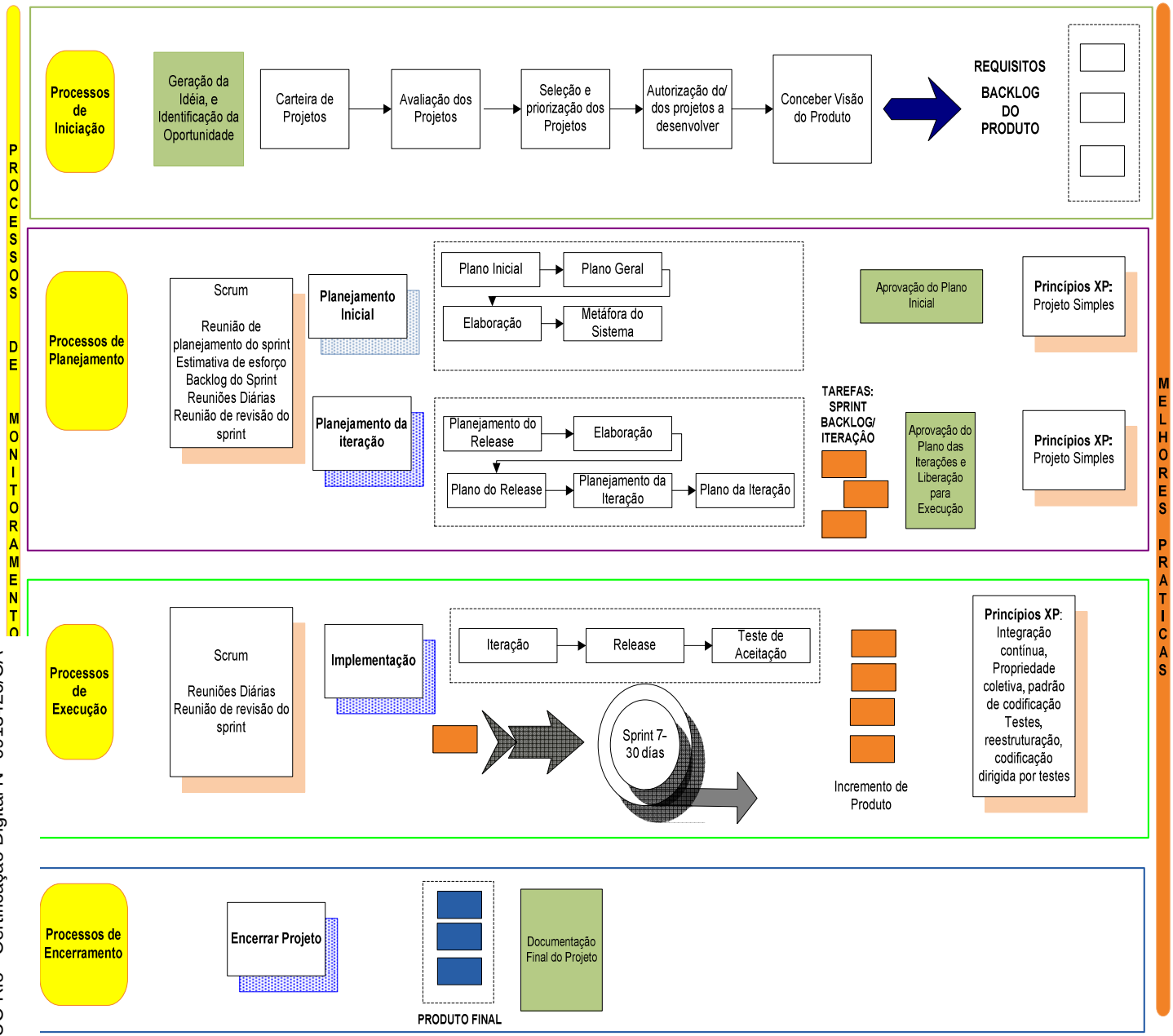


Figura 3.1 –Macro e Micro Componentes do Modelo Proposto

Fonte: Elaboração Própria

A Figura 3.2 descreve o Modelo de Gerenciamento de Projetos de *Software* proposto, que é detalhado a partir do item 3.5.1 ate o item 3.5.6.



,Figura 3.2 – Modelo de Gerenciamento de Projetos de *Software*
Fonte: Elaboração Própria

O Apêndice VI descreve um mapeamento das características das metodologias usadas: XP, Scrum, PMBOK, e as boas práticas aplicadas no modelo proposto: Produção enxuta de *software* e CMM.

3.2.1.
Inicio

Este processo inicia com a geração de uma idéia ou a solicitação de um cliente para desenvolver algum produto de *software* específico. Deverá ser feita uma análise inicial da oportunidade gerada, e categorizar os diversos projetos, com isto a empresa criará uma carteira de projetos (recomendações PMBOK). Posteriormente procede-se á avaliação dos projetos da carteira, se faz a seleção, priorização e finalmente se procede à autorização do projeto a desenvolver, assim como indica a Figura 3.3.

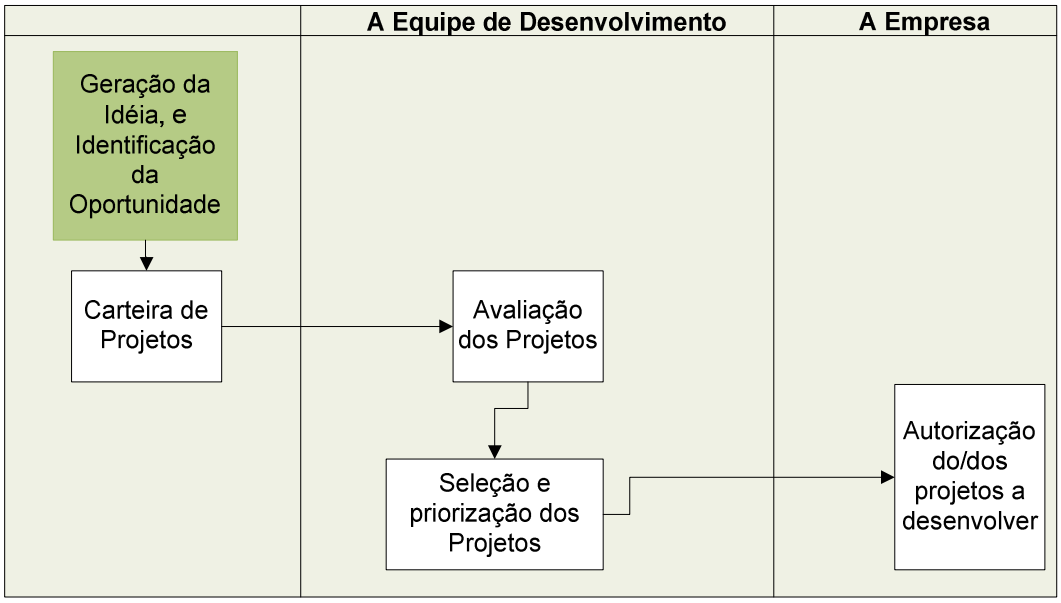


Figura 3.3 – Iniciação: Autorização do projeto a desenvolver

Fonte: Elaboração Própria

Uma vez que o projeto é autorizado, neste processo de inicialização procede-se a declarar a **visão inicial** do produto de *software* para obter a lista de funcionalidades com as premissas e os requisitos principais do projeto (Estas premissas e restrições incluem tudo o que precisa ser realizado, e possa ser associado com valor para o projeto, seja requisitos funcionais ou não, e o *Impediment Backlog* ou restrições são os itens que impedem o progresso do projeto geralmente associado a riscos).

Recomenda fazer as seguintes perguntas para declarar a visão inicial do produto:

- ✓ Qual é a visão que o cliente tem do produto a desenvolver?
- ✓ Que visão a equipe tem do produto a desenvolver?
- ✓ Quais são as limitações e restrições do projeto?
- ✓ A empresa tem a equipe certa para desenvolver o projeto?

Em paralelo procede-se a estimar o orçamento do projeto, assim como os recursos a utilizar para preparar o ambiente de desenvolvimento, (ver a Figura 3.4).

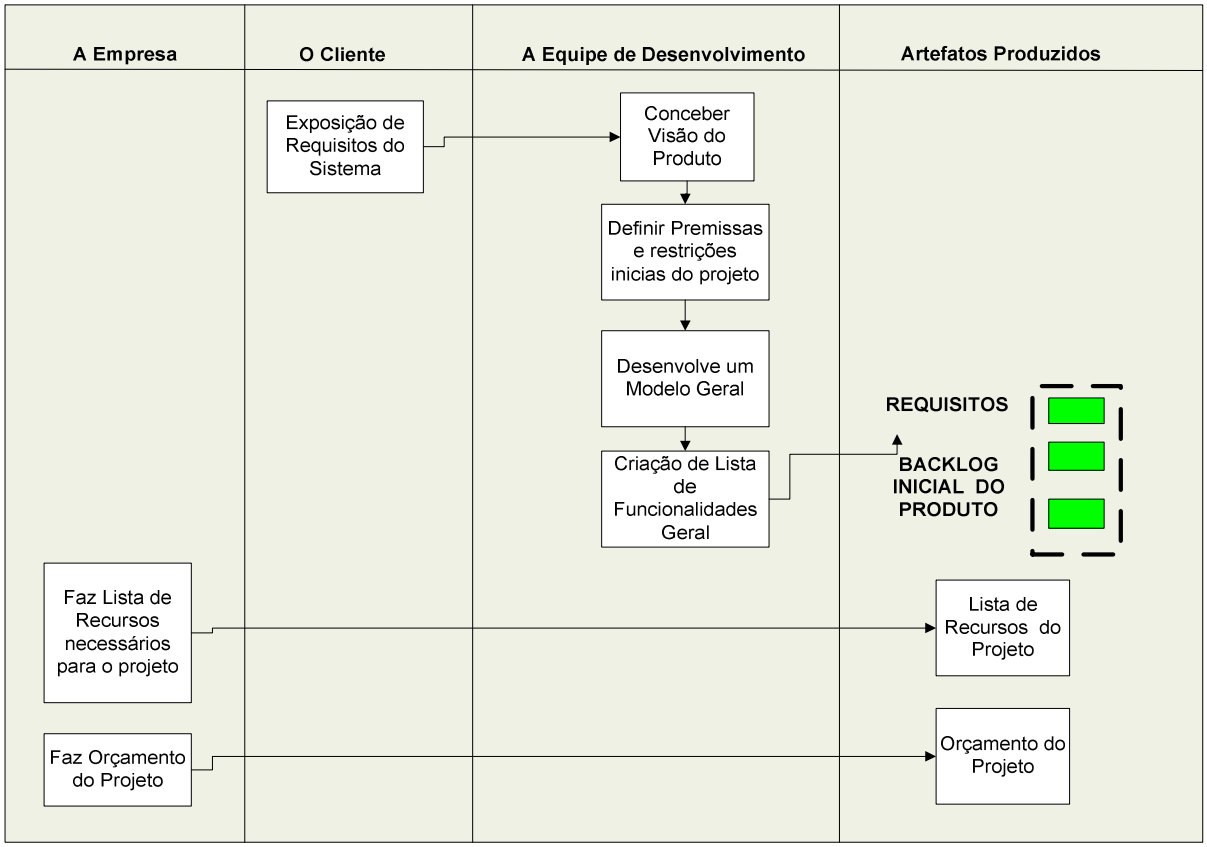


Figura 3.4 – Visão Inicial do Projeto

Fonte: Elaboração Própria

Em relação à análise de recursos se fará análise de:

1. **Análise do ambiente de desenvolvimento:** Nesta etapa se analisará o ambiente atual disponível, e serão listados os diferentes requerimentos necessários para poder desenvolver o projeto.

2. **Análise da necessidade de pessoal:** Nesta etapa, é identificada a equipe necessária para desenvolver o projeto, será analisado se a empresa conta com o pessoal necessário para formar parte da equipe de desenvolvimento, e se procederá a descrever a equipe do projeto, como ajuda da Tabela 3.1, que é baseada nos conceitos da metodologia XP (Revisar Apêndice II), e Scrum.

Tabela 3.1 – Equipe de Desenvolvimento de Projeto de *Software*

SCRUM *Master*	Garante que o time esteja totalmente funcional e produtivo; Facilita a colaboração entre as funções da equipe e elimina os impedimentos do time; Protege o time de interferências externas; Garante que o processo está em andamento; Participam das reuniões diárias, revisão da *Sprint* e planejamento.
SCRUM Team	Equipe de TI Técnico Acompanhador Facilitador Arquiteto Equipe do Cliente Contador de histórias Aceitantes Proprietário de Ouro Planejadores Chefão

Na etapa inicial é importante definir a arquitetura do projeto (do sistema e do *software*), a qual pode ter mudanças e adaptações durante o desenvolvimento do projeto. A Figura 3.5 detalha o os elementos que devem ser definidos nesta etapa.

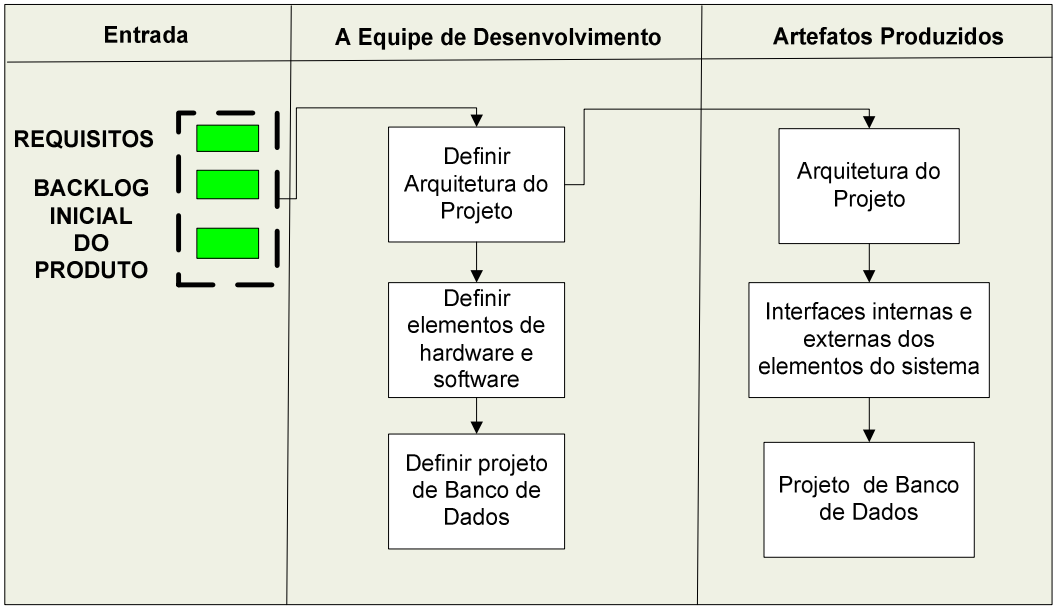


Figura 3.5 – Arquitetura Inicial do Projeto

Fonte: Elaboração Própria

Recomenda-se :

- Considerar que uma boa arquitetura do *software* (definição dos componentes de *software*, propriedades externas, e seus relacionamentos com outros *software*s) ajuda a aumentar a reusabilidade dos componentes, considerando que reusabilidade é um fator de garantia de qualidade num projeto de *software* (Mnkandla e Dwolatzky, 2006).
- A boa especificação das interfaces do sistema contribui notavelmente para a boa manutenção do sistema.

3.2.2. Planejamento Inicial do Projeto

Uma vez aprovado o projeto e gerada a visão inicial, passa-se à fase do planejamento e, considerando a metodologia XP, o planejamento será dividido em duas etapas: Planejamento geral ou inicial e Planejamento detalhado. O plano geral do projeto deverá descrever:

- Os objetivos do projeto,
- Os requisitos do projeto e iterações a desenvolver com as primeiras estimativas em tempo,

- As versões do *software* a desenvolver,
- Descrição geral da equipe de desenvolvimento,
- A freqüência de reuniões da equipe,
- O cronograma do projeto e as estratégias de integração continua do projeto.

Na integração continua dos requisitos do projeto será definidos hardware, *software* e as diversas ferramentas necessárias para integrar o projeto. Na Figura 3.6 se descreve detalhadamente as atividades da equipe de desenvolvimento e os artefatos que deverão ser produzidos nesta etapa.

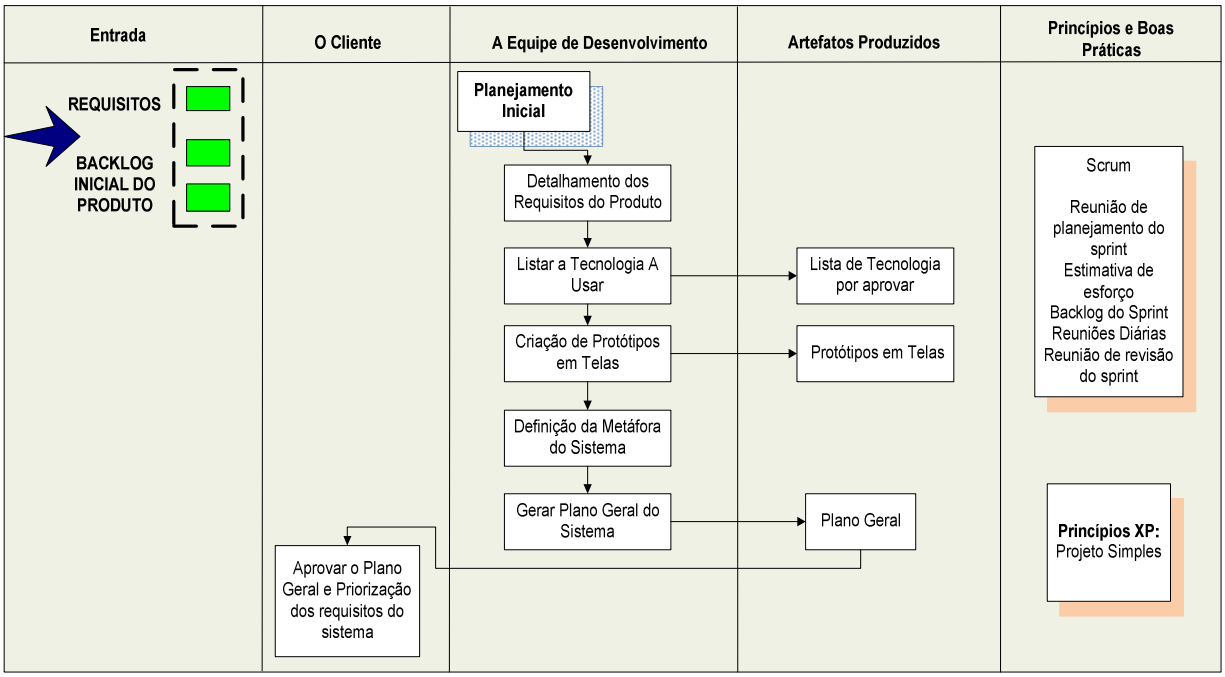


Figura 3.6 – Planejamento Inicial

Fonte: Elaboração Própria

3.2.3. Planejamento Detalhado ou Planejamento das Iterações

São definidas quais requisitos farão parte da primeira, segunda, terceira iteração e assim por diante, considerando requisitos já definidos na etapa inicial.

Nesta etapa de planejamento a equipe de desenvolvimento trabalha com o cliente para priorizar os requisitos definidos inicialmente. Considerando a metodologia Scrum e FDD, criam-se as iterações, fazendo uma seleção dos

requisitos de projeto mais importantes do Product Backlog , com ajuda de um quadro de cartões de funcionalidades como a Figura 3.7.

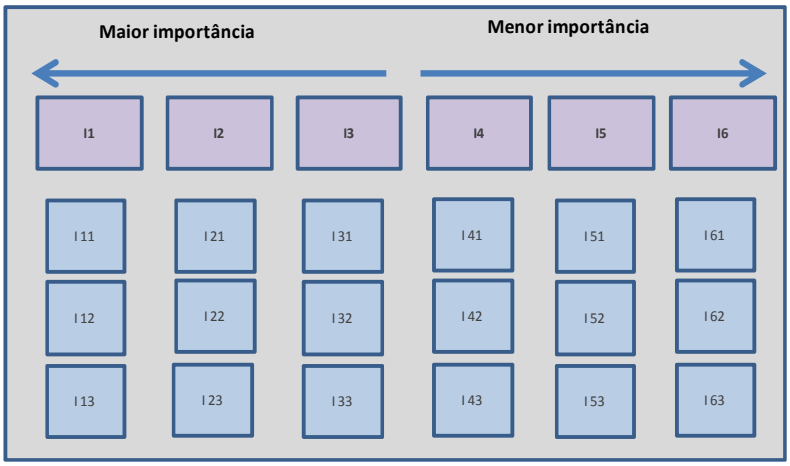


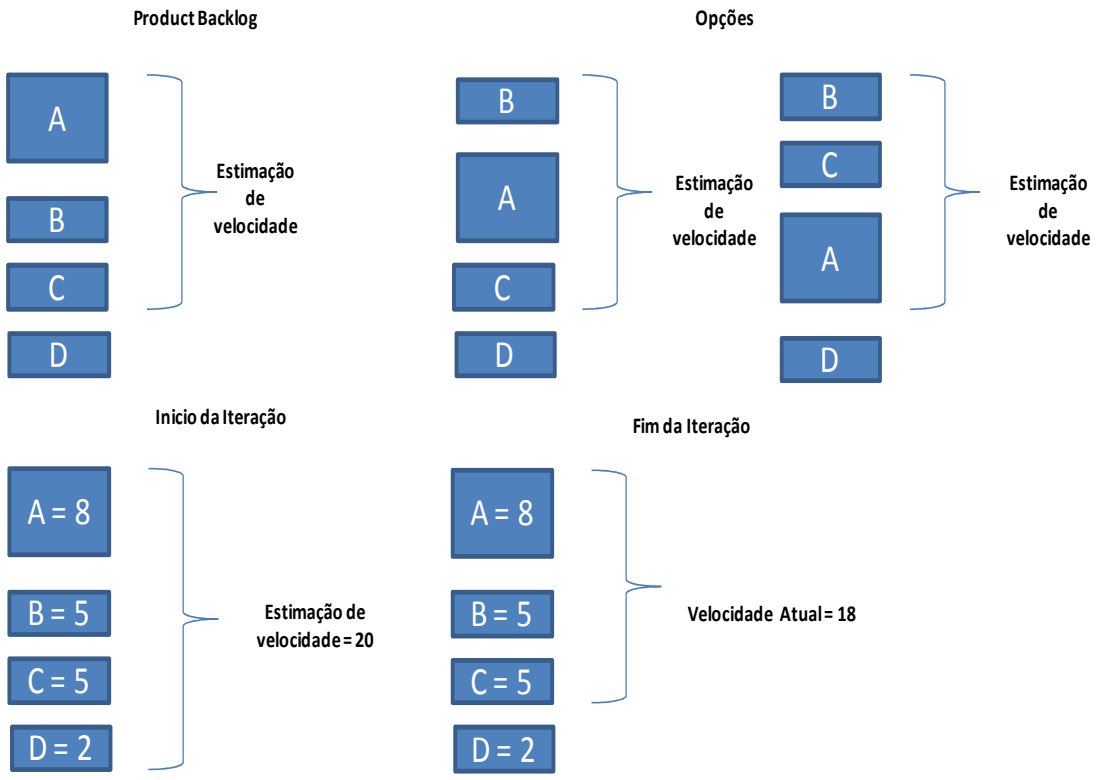
Figura 3.7 – Importância dos cartões de funcionalidades

Fonte: Elaboração Própria

Uma vez priorizados os requisitos pelo cliente a equipe de desenvolvimento deve fazer um estimativa de esforço necessário em tempo para produzir cada um dos requisitos.

Para calcular a velocidade e a ordem em que serão desenvolvidas as iterações com os respectivos requisitos de projeto considera-se os passos da metodologia Scrum, para cálculo de velocidade, ver Figura 3.8.

Para este cálculo de velocidades será necessário considerar um histórico de trabalho da equipe de desenvolvimento.



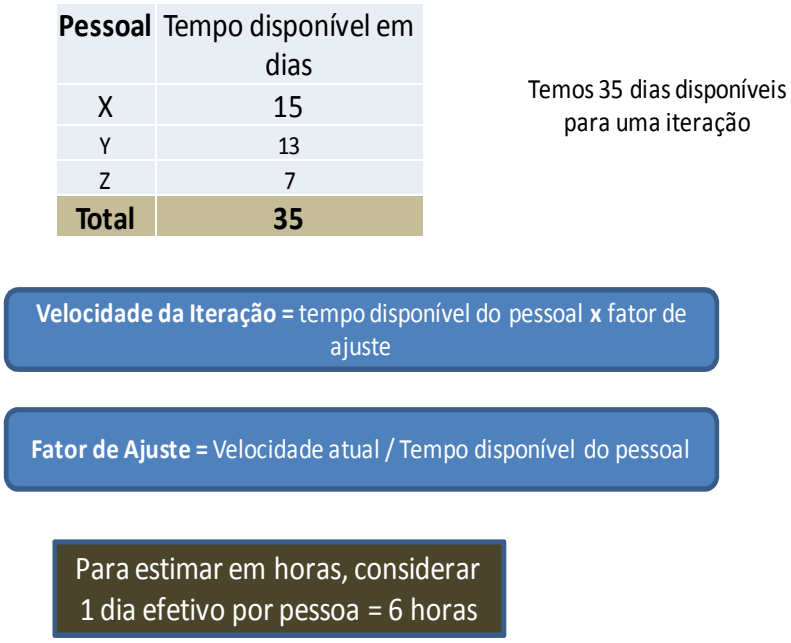


Figura 3.8 – Cálculo de Velocidade das Iterações “SCRUM”

Fonte: Elaboração Própria

- É importante ressaltar a importância do trabalho em conjunto com o cliente já que considerando tanto Scrum, XP, e FDD, é o cliente quem vai exigir quais requisitos serão desenvolvidos em cada iteração de desenvolvimento.
- Finalmente o gerente de projeto deverá gerar um documento com os nomes das iterações, a lista e estimativas dos requisitos de projeto, e os testes de integração para as iterações a desenvolver.

Na Figura 3.9 descreve o processo de planejamento das iterações .

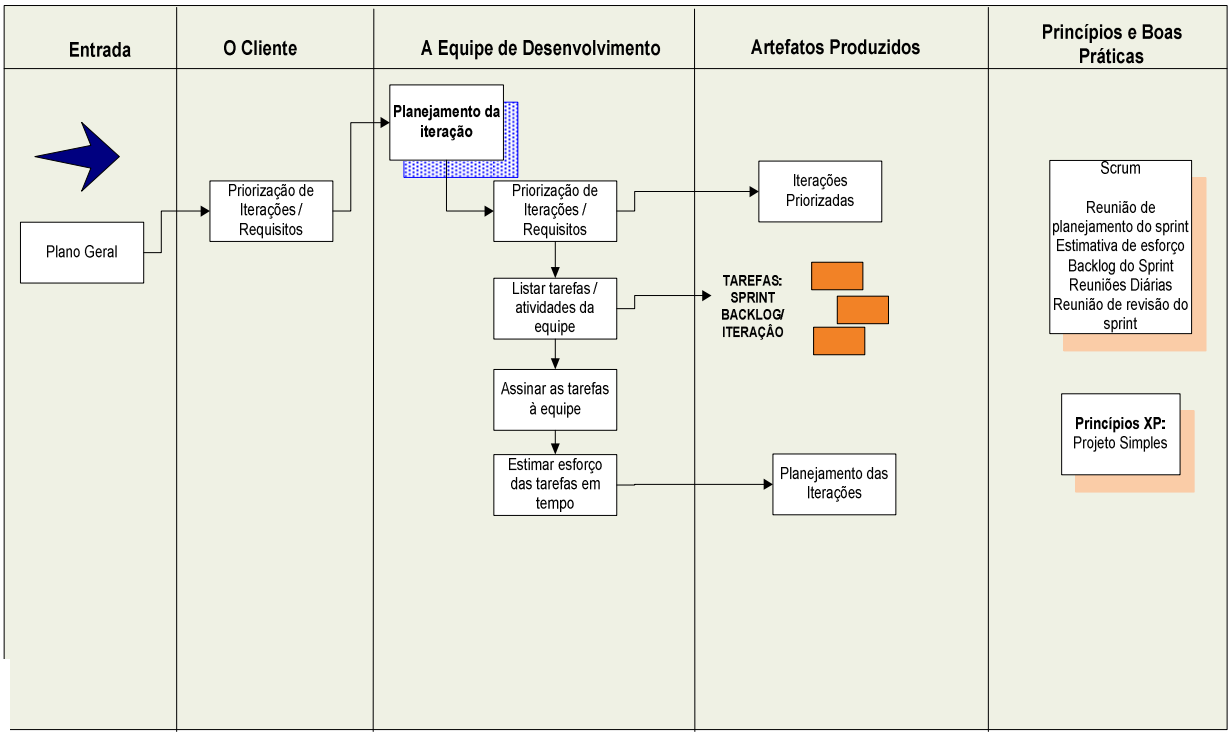


Figura 3.9 – Planejamento Detalhado

Fonte: Elaboração Própria

3.2.4. Execução do Projeto

Procede-se a implantar o listado das iterações definidas na fase do planejamento detalhado. Essa fase é realizada por equipes autodisciplinadas e dirigida por lideres dos projetos que possam garantir um ambiente auto - organizado.

Com ajuda dos princípios XP, Produção enxuta de *software* e gerenciamento Scrum, trabalha-se cada iteração, fazendo os releases (*liberação pública de uma nova versão do projeto*) correspondentes, e os testes de aceitação para ir incrementando o projeto de *software* que se está desenvolvendo, ver Figura 3.10.

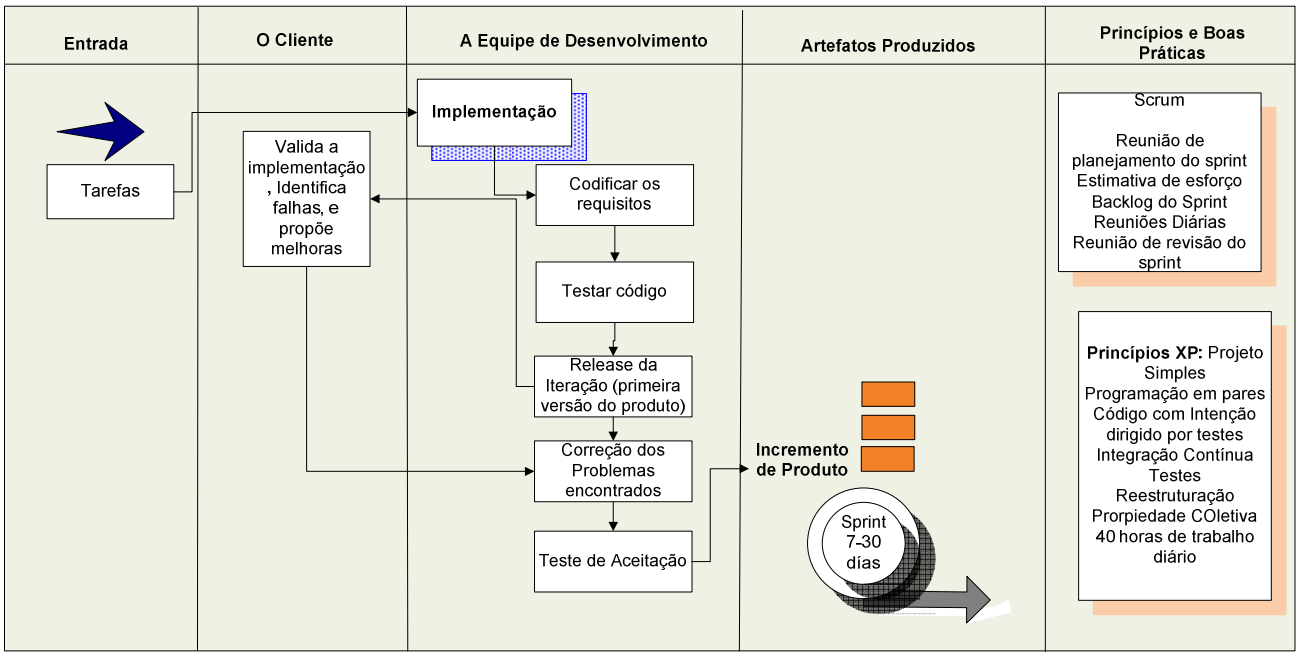


Figura 3.10 – Execução do Projeto

Fonte: Elaboração Própria

Recomenda-se :

- Nesta etapa de projeto, é importante produzir código eficiente. A metodologia XP indica que é importante criar **casos de teste de unidade e resultados esperados** antes de desenvolver cada requisito de projeto com o objetivo de validar a implementação de cada estória na iteração.

- Considerando o fator de qualidade **corretitude**, é necessário **desenvolver código de *software* simples e padronizado**. Este código de acordo com XP, deve **implementar unicamente o que o cliente deseja** e não implementar recursos adicionais, já que isso de acordo com os conceitos de produção enxuta de *software* será considerado como desperdício.

- Como o código é implementado por diferentes pessoas, a utilização de regras de codificação facilita o entendimento e a manutenção do código por todos. Esta prática garante que outras práticas, como o **refactoring e código coletivo** sejam executadas sem maiores dificuldades, e contribui com a adoção

do fator de qualidade **manuntebilidade e reusabilidade** de código.

- A integração das diferentes unidades do sistema deve ser feita freqüentemente, sempre após a execução do seu conjunto de testes. Esta prática garante que os conflitos entre as diferentes partes e o custo de integração sejam mínimos.

- Outro ponto importante na execução é a manutenção. De acordo com XP, deve-se considerar a pratica de programação em pares com o propósito de detectar os defeitos e conseguir corrigi-los a tempo. Assim que quando um programador vai desenvolvendo o projeto, outro devera realizar as revisões e correções necessárias.

- Quando realizadas, modificações nos códigos serão divulgadas e testadas para não comprometer a integridade do projeto.

Outras práticas básicas na organização de trabalho na fase de execução que devem ser consideradas são:

- Reuniões diárias segundo XP e Scrum, como máximo 15 minutos cada dia (SCRUM).
- Reuniões semanais para verificar avanço do trabalho (XP, e SCRUM), 45 min.
- Comunicação com o contador de ***stories*** (XP).
- Programação em pares (XP)
- Propriedade coletiva (XP).
- 40 horas semanais de trabalho (XP e SCRUM).
- Amplificar o aprendizado (Produção Enxuta de *Software*).
- Fazer entregas o mais rápido possível (Produção Enxuta de *Software*): segundo a metodologia Scrum as entregas devem ser feitas cada 7 dias e uma iteração completa aos 30 dias.

3.2.5. Encerramento do Projeto

Uma vez terminado o desenvolvimento das iterações e feitas as integrações e testes do projeto procedesse à etapa do encerramento, ver Figura

3.11, onde o produto final deverá conter todos os itens da lista de backlolg do produto, e se realizará a entrega do produto final ao cliente com a versão completa do sistema.

Existem uma série de documentos que não é necessário criar durante o desenvolvimento do projeto; estes serão criados no final do projeto (etapa do encerramento), com o objetivo de não produzir documentos; antes do necessário, e evitar a atualização desnecessária de documentos, esta é uma prática ágil de desenvolvimento de projetos de *software*.

Teles (2004) afirma que não existe um conjunto de documentos que possam atender as necessidades de todos os projetos em geral, já que cada um dos projetos tem características especificas, mas ele cita um conjunto de documentos coerentes para a realidade dos diferentes projetos que deverão existir antes de encerrar um projeto de desenvolvimento de *software*:

- Estória
- Testes de aceitação
- Testes de Unidade
- Java doc (*padrão de documentação de classes em Java*)
- Modelo de classes
- Modelo de dados
- Processo de Negócio
- Manual do Usuário
- Acompanhamento diário
- Acompanhamento do projeto
- Fotos

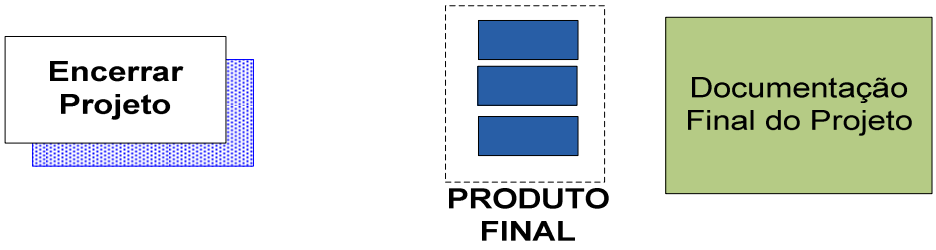


Figura 3.11 – Encerramento do Projeto

Fonte: Elaboração Própria

3.2.6.
Monitoramento e Controle do Projeto

Este processo deverá ser feito em todas as fases do projeto, com o objetivo de assegurar que o produto final de *software* cumpra com as especificações iniciais de qualidade. Nesta etapa é importante identificar os padrões de qualidade que são relevantes para o projeto e determinar como atingi-los. Para definir a qualidade do projeto, o gerente deve:

- Definir os requisitos que satisfaçam as especificações dadas pelo cliente.
- Definir a equipe com as suas responsabilidades.
- Desenvolver os diversos procedimentos e padrões do projeto, e monitorar o desempenho do projeto no longo da sua duração.

Deverão se identificar os fatores de qualidade que se pretendam atender (ver Tabela 2.6). Considerando a metodologia Scrum, se deverá fazer monitoramento e controle do trabalho desenvolvido por cada membro da equipe do trabalho, assim como pela equipe completa.

Kniberg (2007), na metodologia Scrum, explica o uso dos gráficos de controle chamados *Burndown*, onde é analisado o trabalho estimado e as datas estabelecidas pela equipe de trabalho e por cada um dos membros da equipe, que é mostrado nas Figuras 3.12 e 3.13.

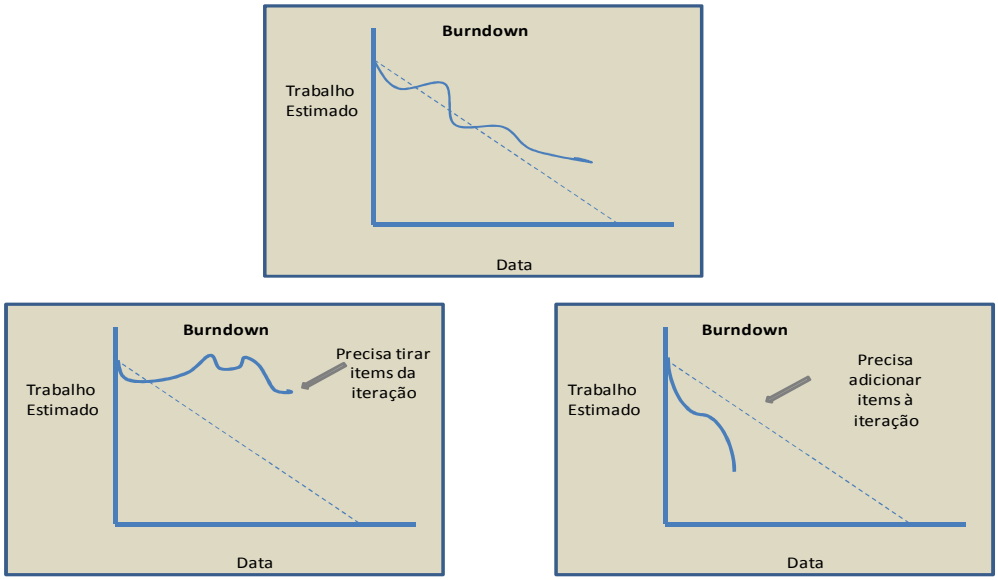


Figura 3.12 – Gráficos de monitoramento e controle

Fonte: Kniberg 2007

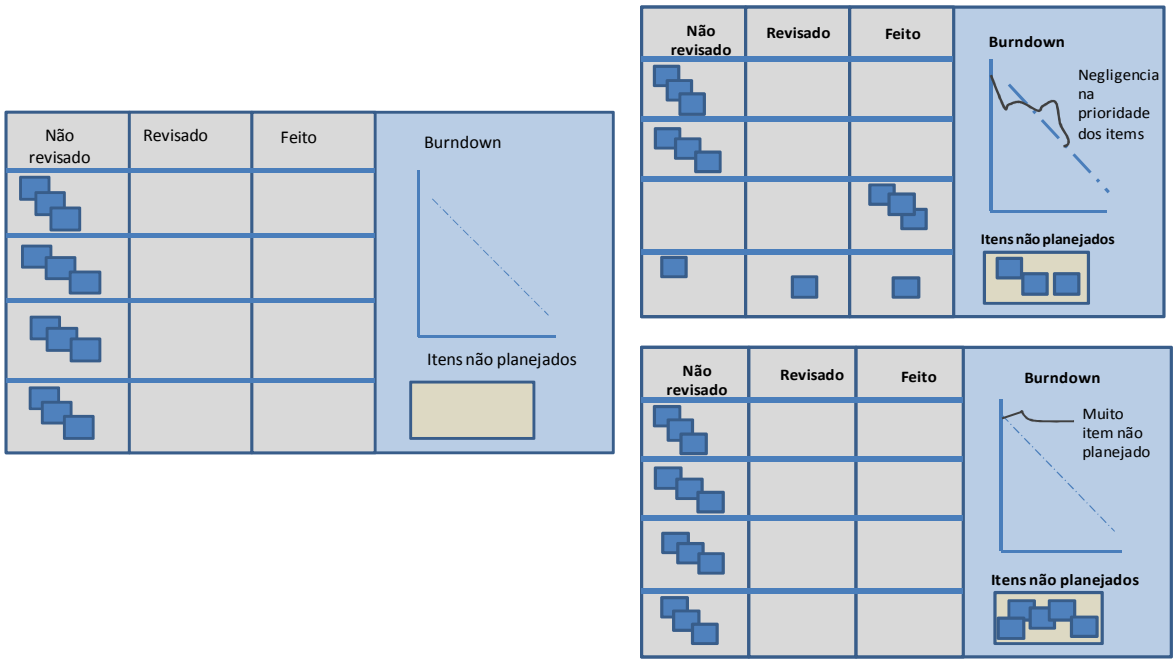


Figura 3.13 – Gráficos de monitoramento e controle

Fonte: Kniberg 2007

Recomenda-se :

- Nesta etapa de monitoramento e controle da qualidade é importante controlar e melhorar os processos e produtos finais de cada uma das iterações em desenvolvimento, considerando os gráficos de monitoramento e controle da Figura 3.13.
- Compete á equipe e ao gerente de projeto identificar os principais problemas e falhas ao longo do desenvolvimento do projeto, por isto é aplicada a prática de reuniões semanais com o objetivo de fazer uma retroalimentação do trabalho desenvolvido ao longo da semana, e conseguir identificar e solucionar os principais problemas encontrados.
- Uma vez definidas falhas ou problemas será importante refinar as iterações, melhorando, adicionando ou mudando requisitos da iteração, já que isto contribui com a melhora na qualidade final do produto.