GOVERNO DO ESTADO DO CEARÁ

Secretaria da Ciência, Tecnologia e Educação Superior

UNIVERSIDADE ESTADUAL VALE DO ACARAÚ

Engenharia de Software

NEaD UVA

Núcleo de Educação a Distância

Universidade Estadual Vale do Acaraú

ENGENHARIA DE SOFTWARE

INTRODUÇÃO

A partir de fragilidades percebidas na engenharia de software convencional, buscando melhorias, desenvolvedores conceituados uniram-se para desenvolver melhores métodos. Eles buscaram valorizar mais as pessoas envolvidas no projeto, as interações, o funcionamento do programa, participação do cliente, alteração durante o projeto. Estes métodos baseiam-se fundamentalmente na satisfação do cliente, trabalhos em pequenos grupos, motivação da equipe, informalidade, simplicidade no desenvolvimento. Tudo isso provem do dinamismo que a modernidade exige.

Cada vez mais o ambiente de negócios demonstrou-se mutável e a capacidade de responder as mudanças é o que define, de forma geral, a agilidade. Mas outras características fazem parte também: satisfação do cliente em todo o processo através da entrega do software funcionando após etapas, que devem ser no máximo em 02 meses – quanto menor tempo melhor; trabalho conjunto entre desenvolvedores e analistas do projeto; indivíduos motivados participando ativamente do projeto; relação interpessoal presencial; atenção na qualidade técnica; regularmente raciocinar formas para ser tornar mais efetivo para assim ajustar seu comportamento;

O comportamento humano é um dos pontos importantes no desenvolvimento ágil. É ele quem conduz o processo antes, durante e depois, seja como cliente ou equipe de desenvolvimento. Existem algumas características-chaves a serem observadas.

Competência. Abrange tanto a conhecimento específico quanto habilidade e noção do processo que está sendo aplicado no desenvolvimento.

Foco. Mesmo executando diferentes tarefas, os membros devem ter o mesmo objetivo: entregar o programa funcionando dentro do prazo.

Colaboração. Avaliar e analisar, constantemente, as etapas que estão sendo executadas, para gerar informações úteis tanto a equipe de desenvolvimento (entender o trabalho da equipe) quanto para o cliente (decidir por modificações).

Tomada de decisão. Autonomia de tomar decisão sobre questões técnicas e de projeto.

Resolução de problemas. Diante de problemas, gerar as alterações necessárias para prevenir problemas futuros.

Respeito e confiança. Trabalhar conjuntamente. Unir forças, cada um a sua maneira, para que o todo seja beneficiado.

Auto-organização. A equipe gerencia a si própria. Os membros se organizam para o trabalho a ser feito da melhor forma.

O desenvolvimento ágil gera benefícios, mas não é em toda situação que podemos aplica-lo. Diante da variabilidade de situações, no que diz respeito à agilidade, desenvolveu-se modelos cada um com seu comportamento, mas com a mesma fundamentação da agilidade.

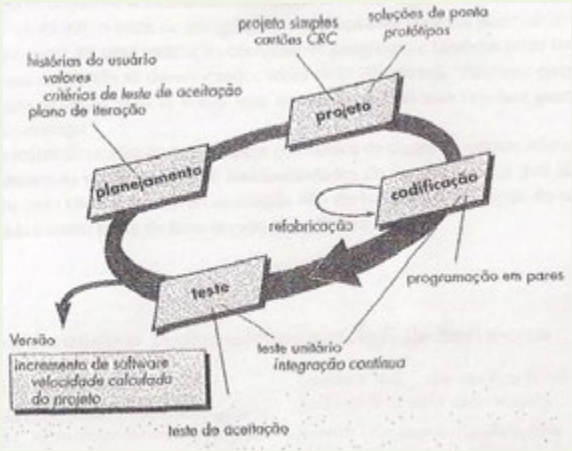
MODELOS DE DESENVOLVIMENTO

Extreme Programming – EP (Extreme Programming - XP). Neste modelo utiliza-se a orientação a objeto no seu desenvolvimento. É baseada também em três pontos.

- **Planejamento.** São criadas histórias pelo cliente onde, em cada história, o cliente cria um nível de prioridade. A equipe avalia as histórias e estipula um prazo para cada. Se este prazo for maior que 03 semanas, solicita-se ao cliente que "quebre" a histórias em partes menores. Estas histórias podem ser escritas a qualquer momento do projeto. Depois que a equipe reavalia e estipula os prazos, dividi-se em um dos três modos para serem desenvolvidas: 1 – todas as histórias são desenvolvidas de imediato; 2 – histórias de alto custo (prioridade) são antecipadas, implementadas primeiro; 3 – histórias de alto risco (complexidade) são antecipadas, implementadas primeiro. Elaborado o projeto, determina-se quanto tempo ficará pronto. Este tempo serve de base para futuras alterações no projeto. Dá poder a equipe de estipular um tempo para executar uma nova versão do projeto.
- **Projeto.** É baseado na simplicidade. Utiliza-se de cartões para organizar e identificar as classes de orientação a objeto relevantes a implementação que está sendo feita. Caso ocorra algum problema, cria-se um protótipo da parte do projeto que se encontrou o problema. Existe ainda a "refabricação" que é alterar, a fim de aperfeiçoar o código, sem modificar o comportamento externo do programa.
- **Codificação.** Com o planejamento (histórias) e o projeto (organização) feitos, não se inicia de imediato a implementação. Primeiro é criado testes unitários para avaliar o que fora elaborado. Assim, os desenvolvedores poderão ser mais eficientes no desenvolvimento do código. Quando iniciada a codificação, ela é feita em pares. Dois programadores trabalham em um mesmo ponto, pois

dois indivíduos, teoricamente, raciocinam melhor que apenas um. De acordo com o prosseguimento, a parte deles é integrada com outros trabalhos. Esta mesma dupla pode ter essa função de integração, ou uma equipe específica pode fazer isso.

- **Teste.** Apesar dos testes unitários antes da codificação, constantemente tem-se que fazer testes de integração e validação. Isso se faz necessário a cada etapa, pois descobrir as pequenas fraquezas no decorrer do projeto é melhor que enfrentar grandes problemas no final do mesmo.

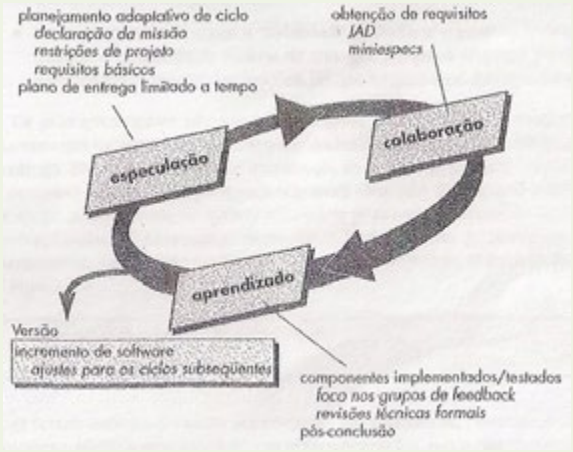


Fonte: PRESSMAN, Roger S. Engenharia da Software, Sexta Edição.

Desenvolvimento Adaptativo de Software – DAS (Adaptative Software development - ASD). Utilizado para construção de sistemas de softwares complexos, necessita da colaboração humana e auto-organização. Baseia-se em três pontos.

- **Especulação.** É o inicio do projeto. O planejamento do ciclo adaptativo é feito neste momento. Ele é baseado nas informações dada pelo cliente, restrições e os requisitos.
- **Colaboração.** Trabalhar em conjunto de modo a multiplicar os resultados. Desta forma, todas as pessoas passam a confiar umas nas outras, assim, quando forem criticar não serem ásperas, ajudam sempre que possível, trabalham tão quando ou mais do que costumam, tem sinergia e expressam-se construtivamente para ações efetivas serem tomadas.
- **Aprendizado.** É feito de três modos:
    - **Foco nos grupos.** Cliente/ou usuário dão retorno indicando se o que foi implementado está dentro do esperado.

- **Revisões técnicas formais.** Aperfeiçoamento do que fora implementado baseado em avaliações.
- **Pós-conclusão.** A equipe faz sua auto avaliação. Tira suas conclusões para aprender e aperfeiçoar a sua abordagem.



Fonte: PRESSMAN, Roger S. Engenharia da Software, Sexta Edição.

Método de Desenvolvimento Dinâmico de Sistemas – MDDS (Dynamic Systems Development Method - DSDM). Este método utiliza-se de protótipos implementados em um ambiente controlado devido o curto prazo. Chega-se a utilizar 20% do tempo total para entregar 80% feito. Um determinado trabalho é suficiente para facilitar o trabalho seguinte. Podem ser complementados depois os detalhes restantes. Este método possui preceitos: 02 atividades de ciclo de vida; 03 ciclos iterativos.

Ciclo de vida:

- **Estudo de viabilidade.** Além de determinar requisitos e restrições do negócio da aplicação, determina se na aplicação pode ser utilizado o método.
- **Estudo de negócio.** Determina os requisitos funcionais e de informação para dar valor de negócio, além disso, determina arquitetura e parâmetros para manutenção.

Ciclos iterativos:

- **Modelo funcional.** Produz protótipos que demonstram a funcionalidade deixando aberto a possibilidade de gerar novos requisitos.

- **Projeto e construção.** Rever os protótipos feitos no modelo funcional para gerar valor de negócio operacional ao usuário. Esta etapa às vezes ocorre juntamente com a do modelo funcional.
- **Implementação.** É feito a última implementação já no protótipo que pode ser operado. Neste momento o programa pode não estar completo ou deva sofrer modificações no decorrer da utilização na prática.

Scrum. Seus princípios são:

- Pequenas equipes visando aumentar a comunicação e o compartilhamento de informações e reduzir a supervisão.
- Adaptável.
- A implementação pode sofrer modificações no decorrer do processo.
- Teste e documentação são produzidos de acordo com o desenvolvimento do projeto.

A metodologia scrum possui algumas etapas que são aplicadas nas sprints (intervalos de tempo - geralmente 30 dias - onde os membros das equipes realizam as tarefas). As etapas são:

- **Pendência de produtos**. São as características dadas pelo cliente.
- **Registro de Sprint**. As características são colocadas na sprint de acordo com a prioridade.
- **Pendência**. Os itens são divididos para as equipes.
- **Reunião.** Após o período determinado, ocorrem reuniões para avaliar o andamento da equipe. Isso é feito por três perguntas básicas: o que você fez desde a última reunião? Teve algum obstáculo? O que você vai fazer?

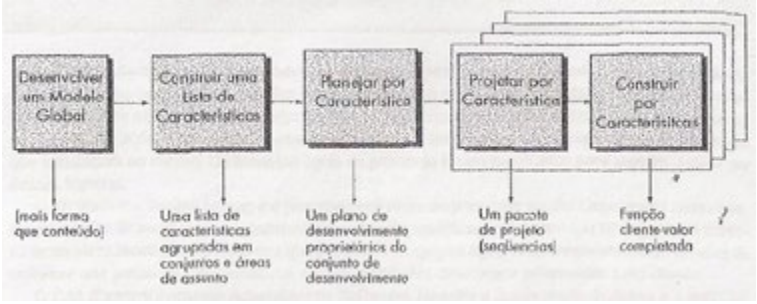
As reuniões são feitas com o intermédio do Scrum Marter. Ele lidera e avalia o comportamento dos membros da equipe. A cada Sprint, se necessário, as funcionalidades são mostradas ao cliente para avaliação. Após esta etapa o projeto pode sofrer modificações e pode passar novamente pelas etapas.

Fonte: images.google.com

Crystal. Baseia-se num conjunto de processos ágeis que, em diferentes projetos, demonstrou-se eficiente. A equipe é desmembrada em "família crystal". Cada família participa de uma "competição" onde a "manobrabilidade", capacidade de atingir os objetivos de forma ágil e eficiente, é o principal meio do desenvolvimento. Este método segue parâmetros onde um deste é que cada equipe escolhe o melhor membro que se encaixa no seu projeto e ambiente.

Desenvolvimento Guiado por Características – DGC (Feature Driven Development - FDD). Modelo de processo fácil aplicável à orientação a objeto. Ele tem como mais importante os quesitos que o cliente passar, tudo sendo guiado pelas diretrizes e técnicas de gestão de projeto. Assim, é possível trabalhar o detalhamento das ambiguidades com o cliente, organizar melhor a ordem de prioridade, desenvolver as características operacionais em duas semanas, avaliar os requisitos mais rigorosamente por serem menores.

Na definição das características utiliza-se o seguinte padrão: <ação> o <resultado> <por | para | de | a> um <objeto>. <*objeto*> é tudo que possa desenvolver papel de pessoa, lugar ou coisa. As características são agrupadas em categorias relacionadas ao negócio da seguinte forma:<verbo no gerúndio (ação)> um <objeto>.



Fonte: PRESSMAN, Roger S. Engenharia da Software, Sexta Edição.

O método tem por padrão as seguintes etapas: travessia do projeto, projeto, inspeção do projeto, código, inspeção do código, promoção para construção.

Modelagem Ágil – MA (Agile Modeling - AM). Os outros tipos de modelagem possuíam problemas como grande volume de notação, grau de formalismo, tamanho dos modelos para grandes projetos e não conseguir manter os modelos após as modificações. Baseavam-se em: os participantes devem entender da melhor forma o que deve ser realizado; divisão dos problemas; avaliação a cada passo para garantir a qualidade. Mas a modelagem ágil tornou-se diferenciada devido:

- Modelagem objetiva: quando identificado à finalidade, os detalhes e a notação serão mais simples desenvolver.
- Modelos múltiplos: para fornecer uma melhor visão, cada modelo seja diferenciado na representação do sistema e apenas modelos relevantes sejam usados.
- Leveza: descartas os modelos que não agreguem valor em longo prazo.
- Mais conteúdo do que representação: dar ênfase ao conteúdo do que sintaxe do modelo pois ele é o mais valioso.
- Conhecimento técnico: conhecer os modelos e as ferramentas que usa para criá-lo. Analisar os prós e contras dos modelos e ferramentas.
- Adaptatividade: capacidade do modelo de se adaptar as necessidades da equipe.

REFERÊNCIA

PRESSMAN, Roger S. **ENGENHARIA DE SOFTWARE**, Sexta Edição.