**CASCATA – ESPIRAL – RUP**

**Características**

**CASCATA:** Rígido e menos administrativo. Simples de entender e controlar.

**ESPIRAL:** É uma abordagem cíclica, para aumentar incrementalmente o grau de definição e de implementação de um sistema enquanto diminui seu grau de risco.

É um conjunto de marcos de ancoragem, para garantir o comprometimento dos interessados com soluções exequíveis e mutuamente satisfatórias para o sistema.

**RUP:** Utiliza UML, Gramática bem definida e estruturada, terminologia, baseado em componentes, guiado por casos de uso, centrado na arquitetura, iterativo, incremental.

**Como Funciona**

**Cascata:** O modelo cascata configura-se como um modelo de engenharia projetado para ser usado no desenvolvimento de diferentes tipos de software. O objetivo principal deste sistema é que as diferentes fases de desenvolvimento seguem uma sequência: A primeira etapa se direciona para a segunda e esta se movimenta para a terceira e assim por diante.

As atividades que devem ser executadas são reunidas em tarefas e executadas sequencialmente, onde uma tarefa só inicia quando a anteriormente foi completamente finalizada.

**Espiral:** Esse modelose divide em duas etapas principais: análise de risco e prototipagem. A cada novo ciclo, esse modelo testa constantemente possíveis erros que podem vir a acontecer.

A cada iteração, à volta da espiral pode ser baseada em um modelo diferente e pode ter diferentes atividades. Ou seja, a cada repetição do ciclo é refeita a análise e a prototipação até que não existam grandes riscos no desenvolvimento do software.

No modelo espiral, em cada repetição do ciclo devem ocorrer no projeto:

**1 – Determinação dos objetivos.**

**2 – Avaliação e redução de riscos.**

**3 – Desenvolvimento e validação.**

**4 – Planejamento da próxima iteração.**

**RUP:** O RUP permite que a equipe do projeto escolha as atividades e os artefatos para serem produzidos, reduzindo assim, o excesso de documentação para torná-lo mais ágil.  O desenvolvimento iterativo e incremental permite aos desenvolvedores o aprendizado em relação ao software, possibilitando assim, a localização de futuros problemas em fases iniciais.

O RUP é descrito a partir de três perspectivas. Na perspectiva dinâmica, o **RUP identifica o ciclo de desenvolvimento do projeto em quatro fases sequenciais sendo**, cada fase, finalizada por um marco principal. As fases do [**RUP**](https://www.devmedia.com.br/rup-rational-unified-process/4574) são iniciação, elaboração, construção e transição.

A perspectiva estática do RUP enfoca as atividades que acontecem no processo de desenvolvimento. Elas são denominadas workflows na descrição do framework. O RUP oferece seis workflows de processos principais e três workflows de apoio principais.

Os workflows são definidos da seguinte maneira:

* **Modelagem de negócios:** os processos de negócios são modelados com a utilização de casos de uso de negócio;
* **Requisitos:** os casos de usos são criados para modelar os requisitos do *software*;
* **Análise e projeto:** **cria-se um modelo de projeto com base em modelos de arquitetura, de componente, de objeto e de sequência**;
* **Implementação:** os componentes do software são implementados;
* **Teste:** o teste é um processo iterativo e é efetuado em conjunto com a implementação do sistema;
* **Implantação:** cria-se uma versão do produto e instala-a no local de trabalho dos usuários;
* **Gerenciamento de configuração e mudanças: esse *workflow* serve como apoio à gerência do projeto em relação às mudanças no sistema;**
* **Gerenciamento de projetos:** esse *workflow* de apoio gerencia o processo de desenvolvimento do *software*;
* **Ambiente:** este *workflow* relaciona-se à disponibilização de ferramentas de *software* adequadas para a equipe de desenvolvimento.

**Vantagens e Desvantagens**

**Cascata Vantagens:** Uma das vantagens do modelo cascata é de que uma tarefa só avança para a outra quando há a validação dos produtos financeira da tarefa atual. Este modelo permite que o idealizador participe de forma ativa do desenvolvimento do projeto do software.

**Desvantagens Cascata:** Um ponto que pode ser considerado como desvantagem por alguns clientes é que o modelo em cascata exige que todos os requisitos sejam definidos no início do projeto, já na primeira fase.

Além disso, só é possível visualizar algo quando as fases já estão bem avançadas, quase no final de tudo. **Não existe uma obrigatoriedade de feedback entre as fases** e não é possível simplesmente voltar para corrigir possíveis falhas.

**Vantagens Espiral:** Estimativas (por exemplo: cronograma) tornam-se mais realísticas com o progresso do trabalho, porque problemas importantes são descobertos antecipadamente.

É mais versátil para lidar com mudanças (sempre inevitáveis) que desenvolvimento de software geralmente exigem.

Engenheiros de Software (que sempre estão impacientes com alongamento da fase do projeto) podem começar o trabalho no sistema mais cedo.

Fácil de decidir quando testar

Não faz distinção entre desenvolvimento e manutenção.

**Desvantagens Espiral:** Pode ser difícil convencer os clientes de que a abordagem evolucionária é controlavél.

O modelo não é usado na mesma extensão que o linear e o de prototipação, e, por isso, não foi “testado” o suficiente.

Avaliação dos riscos exige muita experiência e fia-se nessa experiência para o sucesso. Se um risco não for descoberto, inevitavelmente ocorrerão problemas.

O modelo é relativamente novo e não tem sido amplamente utilizado.

Bem aplicado em sistemas de larga escala.

Sistemas devem ser produtos internos da empresa.

**Vantagens RUP:** Processo robusto e bem definido com a geração de artefatos importantes;

Os maiores riscos são atacados primeiro, diminuindo as chances de fracasso do projeto. Desenvolvimento iterativo. Incremental. Orientada a requisitos. Arquitetura baseada em componentes. Modelagem de software. Modelagem Visual. Produtividade no desenvolvimento.

**Desvantagens RUP:** Complexo e trabalhoso para projetos de pequeno porte.

Exige experiência da equipe. Limitações. Mudanças constantes.

**Manifesto Agile**

* *Indivíduos e interações mais que processos e ferramentas*
* *Software em funcionamento mais que documentação abrangente*
* *Colaboração com o cliente mais que negociação de contratos*
* *Responder a mudanças mais que seguir um plano*

**Os princípios do Agile**

A partir do manifesto Agile foram desenvolvidos **12 princípios a serem seguidos**:

1. Nossa maior prioridade é satisfazer o cliente por meio da entrega rápida e contínua de softwares que tragam valor;

2. Mudanças nos requisitos são bem-vindas mesmo nas etapas finais do desenvolvimento. Os processos Agile aproveitam essas mudanças para oferecer vantagens competitivas ao cliente;

3. Entregar softwares prontos regularmente, em algumas semanas ou meses, com foco sempre na redução do tempo;

4. Profissionais da área de negócios e desenvolvedores devem trabalhar juntos diariamente durante todo o projeto;

5. Criar projetos motivando pessoas. Oferecer a eles o ambiente e o suporte que eles necessitam e depositar confiança em seu trabalho;

6. O método mais eficiente e eficaz de transmitir informações para uma equipe de desenvolvimento é por meio do diálogo presencial;

7. A principal medida do progresso é entregar um software funcionando adequadamente;

8. Os processos ágeis promovem um desenvolvimento sustentável. Patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;

9. Ter atenção contínua à excelência técnica e ao bom design para aumentar a agilidade;

10. A simplicidade - ou a arte de reduzir ao máximo o trabalho desnecessário - é o essencial;

11. As melhores arquiteturas, requisitos e projetos surgem de equipes que se auto-organizam;

12. Em intervalos regulares, a equipe reflete sobre como tornar-se mais eficaz e, em seguida, otimiza e ajusta o seu comportamento de acordo.

A satisfação do consumidor é ressaltada como fator número um do processo de desenvolvimento, e isto só é atingido por meio de um diálogo aberto e constante entre as partes. Ao agir dessa forma, é possível alcançar outro princípio básico: a excelência técnica e atenção aos detalhes, que ampliam a agilidade na elaboração de um software.

O próprio trabalho em equipe é abordado entre estes princípios, no qual se declara fundamental a confiança em cada profissional de desenvolvimento e o espaço para que se trabalhe de forma livre, aumentando a motivação de cada um. Além disso, é estabelecida como essencial a reflexão contínua sobre como tornar o processo mais efetivo e produtivo, com o foco sempre na redução de tempo e desperdício e na velocidade das entregas, que deve acontecer de forma parcial e incremental.

Desta forma, o Agile Software Development traz agilidade e dinamicidade para o desenvolvimento de softwares. A simplicidade é essencial, tanto quanto o diálogo entre a equipe e com quem vai, de fato, utilizar o produto final. No fim das contas, o tempo não é economizado apenas pela redução de etapas, mas também pela inteligência e organização aplicada ao processo.

O Agile promove as relações interpessoais, que muitas vezes são desprezadas no mundo da tecnologia. Dentro da metodologia, as trocas são abraçadas, valorizadas e utilizadas ao máximo. Assim, a empresa que aplica as técnicas propostas consegue, por exemplo, obter o máximo desempenho de cada membro da equipe e atingir um nível de proximidade ímpar com seus clientes.

Uma pesquisa realizada pela HP com 601 desenvolvedores e profissionais da área de Tecnologia da Informação, entre 2015 e 2016, demonstra o avanço do Agile em números. 51% dos times consultados alegam utilizar técnicas parecidas ou adaptadas a partir do Agile no seu dia a dia. Enquanto isso, 24% usam tecnologias híbridas, que aproveitam técnicas do Agile na mesma proporção que as técnicas tradicionais e 16% já afirmam utilizar puramente o Agile em seus processos de desenvolvimento. A tendência é que nos próximos anos esse número cresça e a sua empresa precisa estar preparada para isso.