**Centro Universitário Ritter dos Reis**

Cintia Cunha, Luana Gonçalves, Milena Alves, Ruanne Godoi

**Artigo Acadêmico sobre a Metodologia Adaptive Software Development**

**Porto Alegre**

**2024**

[**1. Introdução**](#_f5kje2vij1pm) **3**

[**2. Principais Fundamentos do ASD**](#_bp3v3i4aapd3) **3**

[**3. Princípios**](#_m1d0buu060e9) **3**

[**4. Como funciona?**](#_jk4r30z3hmd3) **4**

[**5. Ciclos e Estrutura**](#_pdff3ksox5w2) **5**

[**6. Papéis e responsabilidades em projetos ASD**](#_jcsckkrdwj1u) **5**

[**7. Vantagens**](#_8smrafknbuld) **7**

[**8. Desvantagens**](#_6y1eece3vnoj) **8**

[**9. Conclusão**](#_56ky7slp5e79) **9**

[**10. Referências**](#_t9a4qidrg55) **10**

## **Introdução**

Neste artigo vamos abordar as técnicas da metodologia ágil Adaptive Software Development, mais conhecida pela sua sigla ASD. Não muito conhecida como as metodologias Scrum e Extreme Programming mas igualmente útil, a Adaptive Software Development promete uma estratégia de organização voltada para softwares mais complexos.

A ASD propõe que o sucesso de um software se dá pela aprendizagem contínua ao longo do projeto, proporcionando para toda a equipe planos mais flexíveis. Essa metodologia adota um ciclo iterativo composto pelos 3 principais fundamentos que serão abordados ao longo do artigo, são eles: especulação, planejamento e aprendizado.

O diferencial da ASD está em sua abordagem voltada para a adaptabilidade e resposta rápida às mudanças, em contraste com metodologias tradicionais que podem ser mais rígidas. Através da especulação, as equipes formulam hipóteses sobre o desenvolvimento do projeto, que são constantemente testadas e ajustadas conforme necessário. O planejamento é feito de maneira a ser revisto e atualizado constantemente, permitindo uma adaptação rápida às novas informações e circunstâncias. Já o aprendizado é contínuo, incentivando a reflexão e a melhoria contínua do processo e do produto.

Exploraremos como essas etapas são implementadas na prática, e como a ASD pode ser uma ferramenta poderosa para equipes que enfrentam desafios em projetos de software complexos e dinâmicos.

## **Principais Fundamentos do ASD**

**Especulação**: Nessa fase, o foco é a equipe fazer um brainstorming sobre o que precisa ser feito, ou seja, é onde realizam o resumo das informações coletadas com os stakeholders e fazem um esboço do protótipo.

**Colaboração:** durante essa fase, a comunicação é contínua entre a equipe e os stakeholders do projeto. É onde iniciam as reuniões diárias, o ciclo de desenvolvimento e o feedback constante do cliente.

**Aprendizado**: na fase do aprendizado, a equipe deve refletir sobre todo o trabalho realizado até o momento e identificar quais questões devem ser melhoradas e/ou aprimoradas de acordo com a necessidade dos usuários. Nesse momento são realizadas revisões técnicas, retrospectivas e feedback dos usuários.

## **Princípios**

**Entrega Incremental**

A entrega incremental no método ASD é fundamental para proporcionar valor de forma contínua ao longo do desenvolvimento do software. Em vez de esperar até o final do projeto para entregar o produto completo, o ASD preconiza a divisão do trabalho em ciclos menores e iterativos. Cada incremento representa uma parte funcional do software que pode ser testada e utilizada pelo cliente. Isso não só acelera o feedback do cliente, permitindo ajustes mais rápidos e precisos, mas também reduz o risco ao distribuir a entrega ao longo do tempo. Essa abordagem iterativa não apenas melhora a satisfação do cliente ao fornecer valor mais rapidamente, mas também permite que o software se adapte às mudanças nas necessidades e requisitos ao longo do processo de desenvolvimento.

**Pensado para o cliente**

No método ASD, colocar o cliente no centro do processo de desenvolvimento é essencial para o sucesso do projeto. Isso envolve entender profundamente as necessidades do cliente desde o início e manter uma comunicação constante ao longo do ciclo de vida do projeto. Os requisitos são priorizados com base no valor que trazem ao cliente, garantindo que cada incremento entregue atende diretamente às suas expectativas e necessidades. A colaboração contínua com o cliente não apenas ajuda a validar e refinar os requisitos, mas também fortalece a relação de confiança, essencial para o sucesso a longo prazo do projeto.

**Foco na qualidade**

No ASD, o foco na qualidade permeia todo o processo de desenvolvimento desde o início. Isso inclui a adoção de práticas rigorosas de desenvolvimento, como revisões de código, testes automatizados durante a integração. Cada incremento do software é submetido a testes completos para garantir que funcione conforme o esperado e atenda aos padrões de qualidade estabelecidos. A busca constante pela excelência técnica não se limita apenas à funcionalidade do software, mas também se estende à sua usabilidade, desempenho e segurança. Dessa forma, a qualidade não é apenas um objetivo final, mas sim um princípio que orienta todas as etapas do desenvolvimento, assegurando que o produto entregue não apenas funcione, mas também seja confiável e eficaz para os usuários finais.

**Adaptabilidade**

A adaptabilidade é um dos pilares do ASD, reconhecendo que mudanças são inevitáveis durante o desenvolvimento de software. Isso se manifesta na capacidade de responder de maneira ágil e eficaz a mudanças nos requisitos do projeto, nas tecnologias utilizadas ou no ambiente de negócios. A entrega incremental facilita essa adaptabilidade ao permitir ajustes contínuos com base no feedback do cliente e nas novas informações que surgem ao longo do tempo. Além disso, o método ASD adota uma abordagem adaptativa para o gerenciamento de riscos, identificando e mitigando problemas à medida que surgem, em vez de tentar prever todos os possíveis desafios desde o início. Essa flexibilidade não apenas melhora a capacidade de entrega do software, mas também fortalece a resiliência do projeto diante das mudanças e incertezas que podem surgir ao longo do desenvolvimento do projeto.

## **Como funciona?**

O Adaptive Software Development (ASD) adota uma abordagem baseada em componentes para o desenvolvimento de software. Nesta abordagem, os projetos são divididos em componentes, que são grupos de funcionalidades planejadas, implementadas e entregues juntas. Componentes esses que incluem todos os aspectos relacionados.

Existem três tipos principais de componentes no ASD:

* **Componentes principais:** Refere-se às funcionalidades de negócios essenciais do software.
* **Componentes tecnológicos:** Incluem infraestrutura e tecnologias necessárias para a implementação dos componentes principais.
* **Componentes de suporte:** Engloba outros aspectos do projeto, como documentação externa e materiais de treinamento.

## **Ciclos e Estrutura**

Os ciclos adaptativos são uma característica central do ASD. Eles consistem em períodos de tempo definidos durante os quais o trabalho é realizado e os resultados são entregues. Esses ciclos são:

* **Limitados no tempo:** Cada ciclo tem uma duração específica, garantindo que o trabalho seja concluído dentro de prazos predefinidos.
* **Orientados por riscos:** Os ciclos são impulsionados pelo gerenciamento de riscos, com itens de alto risco sendo priorizados nas iterações iniciais.
* **Tolerantes a mudanças:** Os ciclos são projetados para lidar com mudanças, permitindo que a equipe se adapte rapidamente a novos requisitos ou circunstâncias.

A estrutura típica de um ciclo adaptativo no ASD inclui:

* **Definição do tempo:** Estabelecimento do período de tempo para o ciclo, geralmente de acordo com a complexidade e escopo do projeto.
* **Definição de objetivos:** Identificação dos objetivos específicos a serem alcançados durante o ciclo.
* **Atribuição de componentes:** Alocação de componentes a serem desenvolvidos ou aprimorados durante o ciclo.
* **Implementação e Entrega:** Desenvolvimento e entrega dos componentes definidos dentro do período de tempo especificado.

## **Papéis e responsabilidades em projetos ASD**

**Executivo Patrocinador**

* O executivo patrocinador é uma figura de alto nível na organização que fornece suporte e recursos para o projeto. Suas responsabilidades incluem:
* Visão e Alinhamento Estratégico: Garantir que o projeto esteja alinhado com os objetivos estratégicos da organização.
* Apoio e Recursos: Fornecer os recursos necessários (financeiros, humanos, tecnológicos) para que a equipe do projeto possa executar suas tarefas.
* Remoção de Obstáculos: Ajudar a remover obstáculos que a equipe possa enfrentar, facilitando a comunicação entre diferentes partes da organização.
* Monitoramento e Feedback: Monitorar o progresso do projeto e fornecer feedback, garantindo que o projeto esteja no caminho certo.

**Equipe Principal**

* A equipe principal é composta por desenvolvedores, testadores, designers e outros profissionais técnicos que trabalham diretamente no desenvolvimento do software. Suas responsabilidades incluem:
* Desenvolvimento de Software: Escrever código, testar e garantir a qualidade do software desenvolvido.
* Colaboração e Comunicação: Trabalhar em colaboração com outros membros da equipe, compartilhando conhecimentos e informações relevantes.
* Adaptação e Flexibilidade: Estar preparado para adaptar e ajustar o desenvolvimento conforme surgem novas informações e requisitos.
* Entrega Contínua: Garantir a entrega contínua de incrementos de software funcionais, prontos para serem avaliados e testados.

**Gerente de Projeto**

* O gerente de projeto é responsável por coordenar todas as atividades do projeto, garantindo que os objetivos sejam atingidos dentro dos prazos e orçamentos estabelecidos. Suas responsabilidades incluem:
* Planejamento e Organização: Criar planos de projeto detalhados, incluindo cronogramas, marcos e tarefas.
* Gestão de Riscos: Identificar, analisar e mitigar riscos que possam impactar o projeto.
* Comunicação: Facilitar a comunicação entre a equipe principal, o executivo patrocinador e outras partes interessadas.
* Monitoramento e Controle: Acompanhar o progresso do projeto, ajustando planos conforme necessário e garantindo que o projeto permaneça no caminho certo.
* Facilitação de Reuniões: Organizar e facilitar reuniões regulares, como stand-ups, revisões e retrospectivas, para garantir que a equipe esteja alinhada e focada.

## **Vantagens**

***Entregas rápidas***

A metodologia trabalha com constantes interações rápidas e curtas — as sprints, que consistem em interações, reuniões com as pessoas envolvidas da equipe que acontecem de tempos em tempos para definir as tarefas e necessidades do projeto —, a cada reunião trabalhando com uma implementação e feedbacks, fazendo assim com que o fluxo de trabalho seja dinâmico e com resultados rápidos.

Essa abordagem faz com que os stakeholders — são todas as pessoas, empresas ou instituições que têm algum tipo de interesse na gestão e nos resultados de um projeto ou organização, influenciando ou sendo influenciadas (direta ou indiretamente) por ela — fiquem cientes dos avanços, mudanças e alterações do projeto. As sprints permitem respostas imediatas para a validação e consequentemente o avanço do projeto. Também permite que os clientes tenham noção direta dos avanços e funcionalidades do projeto em um prazo curto.

***Flexibilidade***

Um dos pontos principais a se observar no Desenvolvimento Ágil. Enquanto alguns projetos e metodologias são rígidos, exigindo todos os requisitos e funcionalidades no começo do projeto, a abordagem ágil permite que a equipe se adapte às mudanças ao longo do projeto. Isso é particularmente vantajoso no cenário empresarial e negócios, que está sempre em constante evolução. Na metodologia ágil, novas ideias surgem conforme as mudanças do cenário surgem, fazendo com que o projeto possa evoluir ao longo de seu desenvolvimento, evitando assim que um software fique obsoleto no mercado competitivo mesmo antes de finalizado.

***Qualidade do software***

Em cada sprint temos feedbacks diretos do que precisa ser melhorado, planejamentos futuros, tal qual as funcionalidades que serão definidas e testadas. Os testes são uma parte essencial do projeto, garantindo que cada incremento seja livre de erros e atenda aos critérios estabelecidos. A entrega frequente permite com que os erros e falhas sejam detectados facilmente, e assim, corrigidos na mesma velocidade. A interação com os stakeholders permite a apuração para que o software esteja alinhado às expectativas e necessidades reais dos usuários, aumentando a satisfação e a utilidade do produto final.

***Proximidade com os stakeholders***

Como já dito anteriormente, a interação com os stakeholders é fundamental na metodologia ágil. O contato direto com os interessados faz com que seja fácil detectar erros e falhas, assim os mesmos são solucionados rapidamente, otimizando as entregas. Isso se torna possível devido a uma comunicação constante entre a equipe de desenvolvimento e o cliente. Além disso, a possibilidade de ter essa visão do software final ajuda no processo de tomada de decisões que podem impactar no seu desempenho.

## **Desvantagens**

***Equipe dedicada***

Na metodologia ágil, há necessidade de trabalhar com uma equipe de pessoas dedicadas. A comunicação é um dos pontos principais para que a metodologia funcione. É preciso que a equipe seja dedicada em aprender e otimizar sua performance. Assim como é muito importante que a equipe esteja disposta a se adaptar conforme as necessidades de cada demanda.

***Assiduidade no engajamento dos stakeholders***

Nem todos os stakeholders têm disponibilidade ou interesse de participar ativamente no processo. Caso não tenham tempo o suficiente ou se não estiverem dedicados a participarem ativamente do projeto, tal qual a equipe de desenvolvedores, é possível que haja atrasos nas decisões, o que afeta a agilidade do desenvolvimento e a capacidade de entregar valor em cada sprint

***Possível mudança de mindset***

Como muitas equipes e organizações estão acostumadas com metodologias tradicionais, mudar radicalmente para uma abordagem diferente pode ser complicado. A transição para a metodologia ágil pode exigir um período de aprendizado e adaptação, tanto por parte da equipe quanto dos stakeholders. A implementação inadequada ou a falta de compreensão dos princípios ágeis podem levar a problemas na execução do projeto. A equipe pode enfrentar dificuldades em entender completamente os papéis e responsabilidades de cada membro, a interação entre os diferentes sprints e a definição clara de entregas

## **Conclusão**

Em conclusão, o Adaptive Software Development (ASD) se destaca como uma metodologia ágil que prioriza a adaptação contínua e a entrega de valor incremental ao cliente. Ao contrário de abordagens tradicionais, que seguem planos rígidos, o ASD enfatiza a aprendizagem ao longo do projeto e a flexibilidade para lidar com mudanças nos requisitos e no ambiente de negócios. Através de ciclos adaptativos e entrega incremental, o ASD promove uma abordagem iterativa que não só acelera o feedback do cliente, permitindo ajustes rápidos e precisos, mas também melhora a qualidade geral do software ao longo do tempo. Ao colocar o cliente no centro do desenvolvimento e enfatizar a colaboração contínua, o ASD não apenas fortalece a confiança entre desenvolvedores e clientes, mas também aumenta a satisfação geral com o produto final. Assim, o ASD se apresenta como uma metodologia robusta para projetos de software complexos, focada na melhoria contínua e na entrega de soluções que atendam eficazmente às necessidades do cliente.

## **Referências**

COIMBRA, Rodrigo. Agile Methods 1.2: Adaptive Software Development (ASD). LinkedIn. Disponível em:https://www.linkedin.com/pulse/agile-methods-12-adaptive-software-development-asd-rodrigo-coimbra-wpexf/..

TREINAWEB. O que é ASD (Adaptive Software Development). TreinaWeb. Disponível em:https://www.treinaweb.com.br/blog/o-que-e-asd-adaptative-software-development..

WIKIPÉDIA. Adaptive Software Development. Wikipédia. Disponível em:https://pt.wikipedia.org/wiki/Adaptive\_Software\_Development.

DTI DIGITAL. Métodos Ágeis: Vantagens e Desvantagens. DTI Digital. Disponível em:https://www.dtidigital.com.br/blog/metodos-ageis-vantagens-e-desvantagens.