Desenvolvimento Ágil SCRUM

SCRUM integrado com SEMAT

*SCRUM Agile Development*

*SCRUM integrated with SEMAT*

Instituto Politécnico da Guarda

Escola Superior de Tecnologia e Gestão

Guarda, Portugal

Afonso Antunes

Carolina Alexandra Pais Jorge Rei

Vagner Bom Jesus nº - 1701172

Resumo — Scrum é um conjunto de práticas, ou seja, uma abordagem repetível, sistemática e verificável para fazer algo com um propósito específico em mente, concentrado em melhorar a colaboração e o desempenho da equipe para conduzir o desenvolvimento iterativo de maneira colaborativa em duas importantes maneiras: Invocando as partes mais importantes da prática e tornar explícito o que são essas partes importantes.

Palavras Chave – Método Àgil, SCRUM, SEMAT, Sprint, Interação, Espiral.

Abstract — Scrum is a set of practices, that is, a repeatable, systematic and verifiable approach to doing something with a specific purpose in mind, focused on improving collaboration and team performance to drive iterative development in a collaborative way in two important ways: Invoking the most important parts of the practice and making explicit what those important parts are.

Keywords - Agile Method, SCRUM, SEMAT, Sprint, Interaction, Spiral..

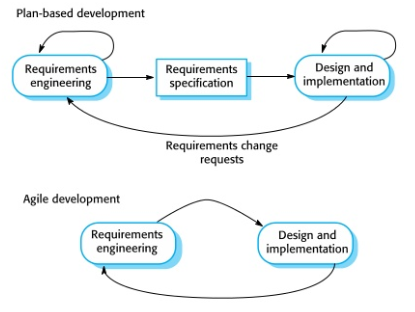
1. Introdução

Scrum é uma abordagem ágil para o desenvolvimento de produtos e serviços inovadores. Com uma abordagem ágil, você começa criando um backlog do produto - um produto priorizado lista dos recursos e outros recursos necessários para desenvolver um produto de sucesso. Guiado pelo backlog do produto, você sempre trabalha primeiro nos itens mais importantes ou de maior prioridade. Quando você fica sem recursos (como tempo), qualquer trabalho que não for concluído terá menor prioridade do que o trabalho concluído (Rubin, 2012). A rica história do Scrum pode ser rastreada até um artigo da Harvard Business Review de 1986, “O Jogo de Desenvolvimento de Novos Produtos” (Takeuchi e Nonaka 1986 citado por Rubin,2012). [1].

Este artigo está organizado da seguinte forma:Iintrodução I, segue-se com os desenvolvimento Ágil na seção II, Modelo Espiral III, Scrum IV, SMANT V

1. desenvolvimento Ágil
   1. Método Ágil

Programadores de software motivados, criam valor comercial ao fazerem entregas de software funcional em curtos espaços de tempo. Este fato gera uma série de ações que proporcionam maior valor aos clientes. No centro destas ações, está a ideia de equipas auto-organizadas onde os membros não são apenas colocados, mas trabalham a um ritmo que sustenta a criatividade e produtividade. Esses princípios incentivam a práticas que acomodam mudanças nos requisitos em qualquer estágio do processo de desenvolvimento dos programas. Além disso, os clientes estão ativamente envolvidos em todo o processo, facilitando *feedback* e reflexões que podem levar a resultados mais satisfatórios. Estes princípios não são uma definição formal de agilidade, mas atuam como guias para se alcançar software de alta qualidade de uma maneira ágil (figura 1) [2]



*Figura 1 – Plan-based development, Agil development* [2]

1. MODELO ESPIRAL

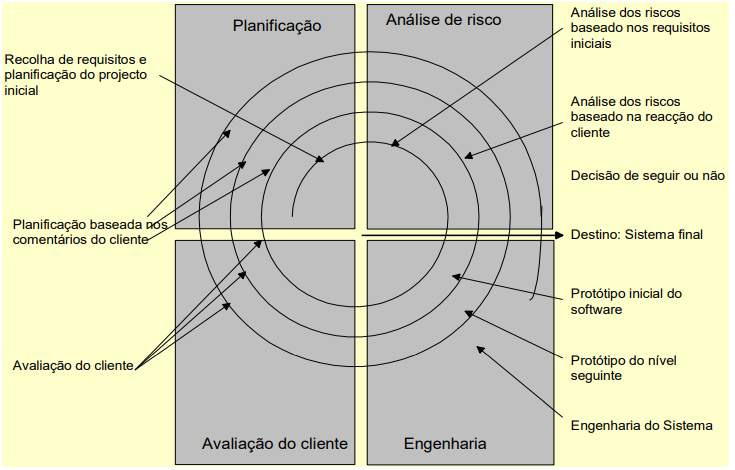
Muitos modelos são compatíveis com o método Ágil, o método espiral e um deles em que a sua característica é de ser modelo orientado a riscos (figura 2). [3]

Figura 2 – Modelo Espiral [3]

Cada ciclo divide-se em 4 fases que são **o planeamento**, onde são definidos os objetivos, **a análise de riscos** que verifica possíveis alternativas, **a engenharia** onde o programa é desenvolvido de acordo com as etapas anteriores e onde, posteriormente, é testado a sua execução e finalmente **a** **atualização** onde é definido o próximo ciclo ou a finalização do projeto e consequentemente do ciclo.

1. SCRUM
   1. Framework

Criado por Jeff Sutherland e Ken Schwaber, este modelo permite implementar as metodologias ágeis, e a sua principal característica é o desenvolvimento interativo. Por consequência o SCRUM é a combinação do *Iterative model* e o *Incremental model*, porque as implementações orientadas a objetos são sucessivas. O SCRUM foi concebido assim para aumentar a velocidade de desenvolvimento [4].

* 1. Terminologia SCRUM

Uma equipa SCRUM tem no máximo sete programadores, que têm uma lista de tarefas (*product backlog)*. Esta lista pode conter requisitos, implementações ou descrições do software. O grupo necessita de fazer entregas de incrementos do programa a cada ciclo não necessitando de testes (*potentially shippable software)*. Na figura o *Scrum* trata-se de uma reunião diária dos elementos para rever o progresso e planear o trabalho, idealmente estas reuniões são curtas e presenciais. O sprint é a fase de desenvolvimento de código geralmente com duas a quatro semanas de duração. Antes de começar cada sprint, existe uma seleção de tarefas e planeamento do mesmo (*select items/ plan sprint)*. No fim de cada ciclo o trabalho é avaliado (*review sprint*) e inicia-se o planeamento do próximo (figura 3) [5].

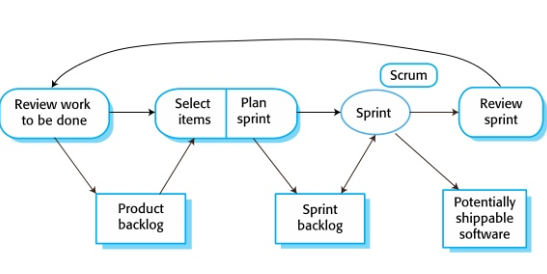


Figura 3- Modelo Scrum [6]

* 1. Exemplo Programação para Internet

**Primeira Iteração**

**Lista de Produção**- Criada uma lista de tarefas de acordo com os requisitos do projeto. **Planeamento do Sprint** – Foi decidido o tema e funcionalidades que iria ser discutido em conjunto, sprint durou (1 semana).

**Sprint -** Discussão e surge o tema “Guarda Eventos”, funcionalidades registadas de acordo com o projeto.

**Avaliação e planeamento do próximo sprint** – professor(a) avaliou e deu *feedback*, e deu novas tarefas para o projeto.

1. SEMAT
   1. Iniciativa Internacional

*Software Engineering Method and Theory,* trata-se de uma iniciativa internacional, dedicada a criar uma engenharia de software baseada na experiência de programadores especialistas para ajudar a próxima geração de profissionais. Ao resultado desta iniciativa, foi dado o nome de essência.

A essência é o núcleo (*kernel*) de uma teoria de engenharia de software, bem como a linguagem utilizada para descrever essa teoria. A sua constituição baseia-se na forma de calcular o progresso do projeto, na categorização das atividades necessárias do software e num conjunto de competências para realizar a referida atividade [7].

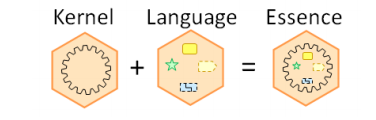


Figura 4- Essência [7]

Em SEMAT, um método é uma descrição de uma maneira de trabalhar na realização de um esforço, o mesmo deriva do funcionamento ou não da experiência que é dividida por regras práticas, diretrizes e padrões [7].

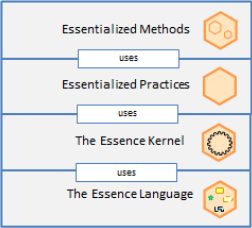
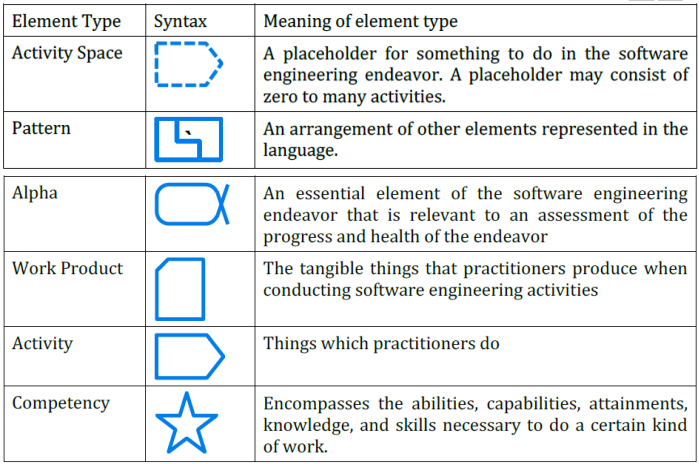
Desenvolvidas com base no *kernel,* as práticas usam métodos ágeis que podem ser expressas pela identificação das áreas em que avança o projeto, pela descrição das atividades que serão usadas para avançar no software e pelas identificações das competências necessárias para a realização das mesmas (figura 5) [7].

Figura 5- Arquitectura SEMAT [7]

* 1. Kernel Essence Language

Existem vários elementos na linguagem da *essence*, (figura 6) dos quais se destacam vários. O “*Alpha”* elemento essencial de engenharia de software, relevante para avaliar o progresso e integridade do esforço. O “*Work Product”* que trata de tudo o que é tangível no decorrer das atividades de engenharia de software. O “*Activity”*, como o próprio nome indica, que é composto pelas atividades executadas pelos elementos do projeto. O “*Competency”* onde são aglomeradas todas as habilidades, capacidades e conhecimentos necessários para fazer uma atividade. O “*Activity Space”* é um placeholder que pode estar vazio ou conter uma ou mais atividades. O “*Pattern*” é composto por várias soluções genéricas para problemas distintos [7].

*Figura* *6- Elementos SEMAT* [8]

/\*\*CAROLINA APAGUEI ALGO SEM QUERER\*\*\*/

O product owner é o ponto central com o poder de liderança do produto. Este é a única autoridade responsável por decidir quais recursos e funcionalidades construir e a ordem em que serão construídas. O product owner mantém e comunica a todos os outros participantes uma visão clara do que a equipa Scrum está a tentar alcançar. Como tal, o product owner é responsável pelo sucesso geral da solução que está a ser desenvolvida ou mantida. Não importa se o foco está num produto externo ou numa aplicação interna; o product owner ainda tem a obrigação de garantir que o trabalho mais valioso possível, que pode incluir trabalho com foco técnico, seja sempre executado. O product owner, o scrum master e a equipa de desenvolvimento colaboram ativamente para desta forma ser garantido um rápido desenvolvimento do produto o product owner também deve estar sempre disponível para desta forma responder o mais rápido possível a questões que sejam colocadas de ambas as partes.

* 1. ScrumMaster

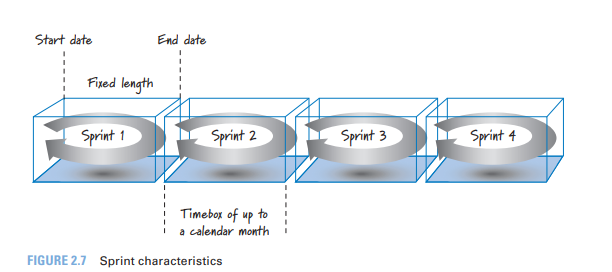
O ScrumMaster ajuda todos os envolvidos a compreender e a abraçar os valores, princípios e práticas do Scrum. Este atua como coach, fornecendo a liderança de processo e ajuda a equipa de Scrum e o resto da organização a desenvolver a sua própria abordagem Scrum de alto desempenho e específica à organização. Ao mesmo tempo, o Scrum-Master ajuda a organização através do complicado processo que pode ocorrer durante a adoção do Scrum. Como um facilitador, o ScrumMaster ajuda a equipa a resolver problemas e a fazer melhorias no uso do Scrum. Este também é responsável por proteger a equipe das interferências externas e assume um papel de liderança na remoção de impedimentos que inibem a produtividade da equipe (quando os próprios indivíduos não podem resolvê-los razoavelmente). A autoridade do Scrum Master é limitada e desta forma este não pode exercer controlo sobre as equipas de desenvolvimento assim este não é como um gerente ou product owner. O Scrum-Master funciona como um líder, não um gerente.

* 1. Equipa de desenvolvimento

Ao desenvolver se um software que ira ser comercializado existem vários elementos de equipa como arquitetos, os administradores de bases de dados entre outros. Scrum define o papel de uma equipa de desenvolvimento, que é simplesmente uma coleção diversa e multifuncional desses tipos de pessoas que são responsáveis por projetar, construir e testar o produto desejado. A equipe de desenvolvimento se auto-organiza para determinar a melhor maneira de cumprir a meta estabelecida pelo product owner. A equipe de desenvolvimento tem normalmente de cinco a nove pessoas; os seus membros devem, coletivamente, ter todas as habilidades necessárias para produzir software funcional de boa qualidade. Claro, Scrum pode ser usado em esforços de desenvolvimento que requerem equipes muito maiores. Com tudo não devemos ter equipas com grandes números de elementos por exemplo 35, estes elementos devem ser divididos em pequenas equipas onde cada por cada equipa Scrum não devem existir mais de nove pessoas na equipa de desenvolvimento.

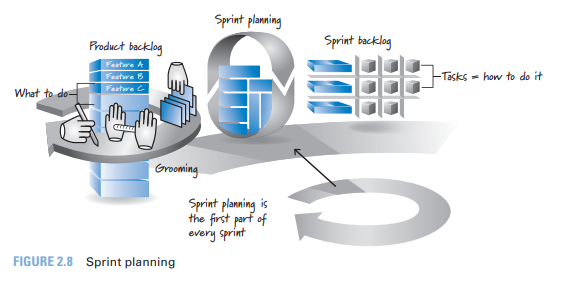
1. ETAPAS DO MODELO SCRUM
   1. Sprint

No Scrum, o trabalho é executado em iterações ou ciclos de até um mês civil, chamados de sprints. O trabalho concluído em cada sprint deve criar algo de valor tangível para o cliente ou utilizador. os sprints são janelas temporais definidas por data de início e de fim onde em geral devem ter sempre o mesmo comprimento, um novo spring começa automaticamente após o término do anterior. Durante este os objetivos são fixos e não podem ser alterados, contudo por vezes os clientes precisam de efetuar alterações aos objetivos.

Figura 3.1 – Características do Sprint

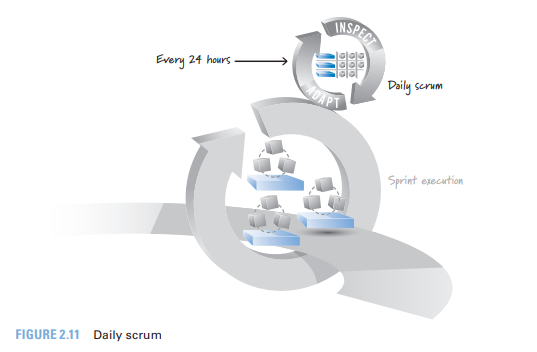
* 1. Spring planning

Um backlog de produto pode representar muitas semanas ou meses de trabalho, que é muito mais do que pode ser concluído em um único sprint curto. Para determinar o subconjunto mais importante de itens do backlog do produto a serem construídos no próximo sprint, o proprietário do produto, a equipe de desenvolvimento e o ScrumMaster realizam o planejamento do sprint. Durante o planejamento do sprint, o proprietário do produto e a equipa de desenvolvimento concordam com uma meta do sprint que define o que o próximo sprint deve atingir. Usando esse objetivo, a equipa de desenvolvimento reavalia o backlog do produto e determina os itens de alta prioridade que a equipa pode realizar de forma realista no próximo sprint enquanto trabalha a um ritmo justificável – este ritmo é confortável para a equipa e assim pode ser mantido por um período de tempo longo e com bons resultados. Para adquirir confiança no que pode ser atingido, muitas equipas de desenvolvimento dividem cada recurso-alvo em a conjunto de tarefas. A coleção dessas tarefas, junto com seus itens associados ao backlog do produto, forma um segundo backlog denominado sprint backlog. A equipa de desenvolvimento então fornece uma estimativa (normalmente em horas) do esforço necessário para concluir cada tarefa. Dividir os itens do backlog do produto em tarefas é uma forma de design e planeamento just-in-time de como fazer os objetivos. A maioria das equipas Scrum que executam sprints de duas semanas a um mês de duração tentam concluir o planeamento do sprint em cerca de quatro a oito horas. Um sprint de uma semana não deve levar mais do que algumas horas para ser planeado (e provavelmente menos). Durante este tempo, existem várias abordagens que podem ser usadas. Uma abordagem simples pode ser: Seleciona se um item do backlog do produto (sempre que possível, o próximo item mais importante conforme definido pelo proprietário do produto), divide se o item em tarefas e determina se se o item selecionado irá caber razoavelmente no sprint (em combinação com outros itens direcionados para o mesmo sprint). Se couber e houver mais capacidade para concluir o trabalho, repetir o ciclo até que a equipe fique sem capacidade para fazer mais trabalho. Uma abordagem alternativa seria o proprietário do produto e a equipa selecionarem todos os itens da lista de pendências do produto de uma vez. A equipe de desenvolvimento sozinha faz a divisão das tarefas para confirmar se esta realmente pode entregar todos os itens de backlog do produto selecionado.

Figura 3.2 – Sprint Planning

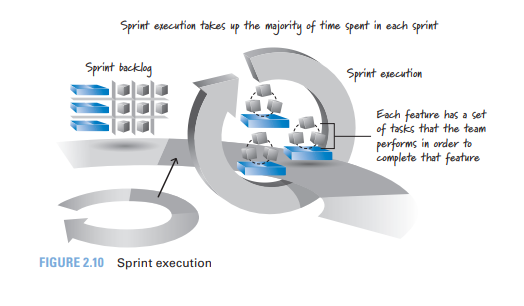
* 1. Daily Scrum

A cada dia do Sprint, os membros da equipa de desenvolvimento realizam uma reunião diária sempre no mesmo horário, de preferência. Esta atividade de inspeção/adaptação pode ser chamada como stand-up diário devido à prática comum de todos se colocarem durante a reunião com intuito de ajudarem a promover a prontidão. O Daily Scrum é primordial para ajudar a equipa de desenvolvimento a gerir o fluxo de trabalho ágil e flexível em Sprint. No entanto, também pode ser muito útil para comunicar o estado do Sprint Backlog entre os membros da equipa de desenvolvimento. Em suma, o Daily Scrum é uma atividade de inspeção, sincronização e planeamento diário adaptável que ajuda uma equipa de auto-organização a fazer melhor o seu trabalho.

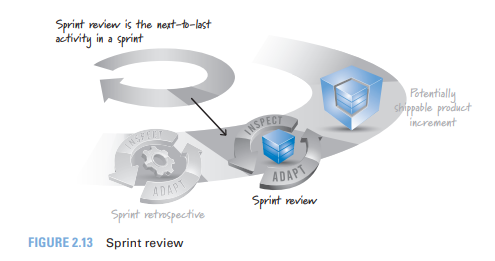
Figura 3.3 – Daily SCRUM

* 1. Sprint Execution

O Sprint Execution é responsável pela maior parte do tempo no decorrer do Sprint. Ele começa após a conclusão do Sprint Planning e termina quando a Sprint Review começa. Durante o Sprint Execution, os membros da equipa de desenvolvimento determinam a melhor maneira de cumprir a meta estabelecida durante o Sprint Planning. O proprietário do produto deve estar disponível, sempre que possível, durante esta etapa para responder às perguntas de esclarecimento, fazer uma revisão ao trabalho intermediário e fornecer feedback para a equipa, discutir os acertos para a meta do Sprint se as condições o justificarem e verificar se os critérios de aceitação dos itens do Product Backlog foram aprovados. Resumindo, o Sprint Execution envolve planeamento, gestão, execução e comunicação do trabalho necessário para criar esses recursos, de forma testada e funcional.

Figura 3.4 -Sprint Execution

* 1. Sprint Review

Nos dois últimos Sprints, observamos que durante o Sprint Planning, planeiam o trabalho e durante o Sprint Execution, fazem o trabalho. No Sprint Review, inspecionam e adaptam o resultado do trabalho. O Sprint Review acontece sempre perto do final de cada ciclo do Sprint, isto é, logo após o Sprint Execution e logo antes do Sprint Retrospective. O Sprint Review dá a todos aos que colaboraram para o esforço do desenvolvimento do produto uma oportunidade de fazerem inspeção/adaptação do que foi construído até agora. Esta etapa proporciona uma visão coerente do estado atual do produto e é o momento exato de discutirem acerca da melhor maneira de seguir para a próxima etapa visto que ajuda a garantir que a equipa está focada na criação de um produto de sucesso. Esta etapa é um dos Sprint’s não só dos mais importantes, mas também dos mais rápidos do modelo SCRUM dado que permite correções consecutivas para manter o desenvolvimento do produto a bom porto. Figura 3.5 – Sprint Review

1. Conclusões

Pretendeu-se com este artigo dar a entender uma visão abrangente do desenvolvimento ágil SCRUM e as suas metodologias. Conclui-se que o método mais popular do método ágil é o SCRUM e que o SEMAT é a iniciativa mais viável que ajuda a equipas de desenvolvimento. Conclui-se também que é necessário seguir cada etapa do método SCRUM para que o desenvolvimento dos projetos sejam realizado com mais eficácia.

# REFERÊNCIAS BIBIOGRAFICA

|  |  |
| --- | --- |
| [1] | K. S. Rubin, “Essential Scrum,” p. 452 , 2012. |
| [2] | ©. S. 2009, “UiOUiO Universitetet i Oslo,” [Online]. Available: https://www.uio.no/studier/emner/matnat/ifi/INF1050/v14/timeplan/sommerville.ch\_03\_agile\_sw\_dev.pdf. [Acedido em 15 Novembro 2020]. |
| [3] | M. C. Silveira, “Processos de software”. |
| [4] | S. B. ,. S. Apoorva Srivastava, “SCRUM Model for Agile Methodology,” *SCRUM Model for Agile Methodology ,* pp. 864-869, 2017. |
| [5] | D. Gurendo, “Software Development Life Cycle (SDLC). Scrum Model Step by Step,” XB Software, 30 06 2015. [Online]. Available: https://xbsoftware.com/blog/software-development-life-cycle-sdlc-scrum-step-step/. [Acedido em 16 03 2020]. |
| [6] | I. Sommerville, “Chapter 3 - Agile Software Development,” em *Software Engineering*, Pearson, 02/01/2015. |
| [7] | D. A. T. (. D. S. Copyright © 2013–2018, Software Engineering Essentialized, 2018. |
| [8] | M. C. Silveira, “Software Engineering Method and Theory,” 2020, p. 36. |