Por que utilizar agilidade em desenvolvimento de software?

A partir da criação dos PCs (Personal Computers) nos anos de 1980, a demanda por sistemas e software começou a crescer exponencialmente. As empresas e indústrias começaram a ser informatizadas, inicialmente para resolver problemas com as folhas de pagamento e posteriormente, para tornar os controles de estoque mais eficientes. Essas necessidades procuravam atender às novas tendências econômicas que passavam a ser difundidas pelo mundo como o conceito de Just in Time.

Nos anos de 1990, os governos passaram a ver na informatização uma solução simplificada para diversos problemas fiscais, potencializando ainda mais o aumento da demanda de sistemas e softwares.

Atender a essa grande e crescente demanda, gerou iniciativas por todo o mundo que resultaram no surgimento das Consultorias ou Software Houses, empresas especializadas no desenvolvimento de softwares. Mas essa “especialização” não significava que eram eficientes no processo de desenvolvimento.

Nesses anos, a carreira de programador lhe garantia 3 coisas:

1. Alta remuneração
2. Excesso de trabalho
3. O surgimento da palavra Stress

Como o mundo passou a almejar cada vez mais a informatização, a quantidade de dados e processamento de informação era um desafio que tentaram atender através da evolução e criação de Hardwares e, em paralelo, do surgimento de novas linguagens.

Todas essas evoluções causam diversos impactos. Passaram a ser necessários o versionamento, critérios de segurança, backup, entre outros. Essa burocratização do desenvolvimento de software deu origem ao termo HSD - Heavy Software Development (Desenvolvimento de Software Pesado). De uma forma bastante minimalista, este era o cenário:

- Necessidades crescentes
- Novas linguagens
- Novos hardwares
- Pouco entendimento
- Muita burocracia
- Documentação abrangente
- Rápida obsolescência
- Excesso de trabalho
- Falta de profissionais

A partir de 1994, com o surgimento da internet e com os primeiros fóruns para dividir e difundir conhecimento, as primeiras comunidades foram criadas e programadores do mundo todo começaram a se unir. O entendimento de que os problemas eram comuns fez com que alguns entusiastas pensassem em como solucioná-los.

Kent Beck criou o Extreme Programming, ou XP, que se baseava no conceito PDCA de Planejamento, Design, Codificação e Testes. Ele defendia o Pair Programming, ou Programação em Dupla, para que o conhecimento fosse compartilhado e a evolução dos times ocorresse naturalmente.

Alistair Cockburn e Jim Highsmith criaram o Crystal, talvez a primeira ferramenta de desenvolvimento a considerar a complexidade dos itens para elaboração dos times. Baseava-se nos valores: Comunicação Constante, Segurança, Foco, Acesso para Usuários, Testes automatizados e Integração.

Ken Schwaber e Jeff Sutherland criaram o framework Scrum, que possui 3 pilares (Transparência, Inspeção e Adaptação) e um conjunto de eventos (Sprint, Planning, Daily, Review e Retrospective).

Assim como estas 3 ferramentas, diversas outras surgiram e possuíam seus adeptos. Em fevereiro de 2001, nos EUA, precisamente em Wasatch no estado de Utah, dezessete entusiastas da programação se reuniram:

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland e Dave Thomas.

Esse grupo de pensadores independentes sobre desenvolvimento de software e, às vezes, concorrentes entre si, tinham entendimentos em comum e muitos já vinham se aproximando como Kent Beck e Jeff Sutherland. Eles pontuaram diversos ideais e formularam um conjunto de valores e princípios intitulado de Manifesto Ágil.

Vivemos em um mundo conhecido como VUCA:

- Volátil
- Uncertain (Incerto)
- Complexo
- Ambíguo