"Há princípios essenciais no âmago de qualquer aplicação bem sucedida dos casos de utilização:
Princípio 1: Manter a simplicidade contando histórias. As histórias abordam como atingir o objectivo e como lidar com problemas que ocorrem no decurso (da interação)." [1]

Explique em que consiste o "princípio" defendido por I. Jacobson e em que medida contribui para a a aplicação bem sucedida dos casos de uso num projeto.

---

[1] Adaptado do texto original: "There are basic principles at the heart of any successful application of use cases:

Principle 1: Keep it simple by telling stories. The stories cover how to achieve the goal and how to handle problems that occur on the way." In: *I. Jacobson et al, "Use Case 2.0: The hub of software development"*.

Tópicos de resposta pretendidos (tendo presente o *White paper* User Cases 2.0):

- Para manter o processo de levantamento de requisitos simples, Jacobson defende que a análise se requisitos se baseie na **exploração de cenários de uso**, em que o foco está em contar como a interação com o sistema se deve desenrolar, para realizar um objetivo.
- O foco nas histórias/narrativas, adotando a perspetiva dos atores, elimina barreiras de comunicação (e.g.: disciplinas técnicas especializadas) e favorece a **colaboração interdisciplinar**.
- As narrativas (histórias de utilização) revelem o comportamento esperado do sistema, ou seja, os **requisitos e o contexto** em que ocorrem.

Em que consiste o princípio?

O princípio defende que **casos de uso devem ser abordados como "histórias"** simples e narrativas, focadas em:

1. **Objetivo do usuário**: Descrever **o que** o usuário quer alcançar (ex: "Realizar um pagamento").
2. **Fluxo natural**: Detalhar **como** o sistema e o usuário interagem para atingir esse objetivo, incluindo:
   - Passos principais (fluxo feliz).
   - Problemas comuns e como lidar com eles (fluxos alternativos/exceções).

Como esse princípio contribui para o sucesso dos casos de uso em um projeto?

1. **Facilita o entendimento compartilhado**
   - Histórias são mais acessíveis do que documentos formais, permitindo que todos (incluindo não técnicos) compreendam os requisitos.
2. **Promove foco no valor do usuário**
   - Ao narrar a interação do usuário com o sistema, mantém-se o foco nas **necessidades reais**, evitando funcionalidades desalinhadas.
3. **Simplifica a identificação de riscos**
   - Problemas ("fluxos alternativos") são antecipados durante a narração, reduzindo surpresas em fases posteriores.
4. **Alinha-se a métodos ágeis**
   - A estrutura de histórias é compatível com **User Stories** e técnicas como **BDD (Behavior-Driven Development)**, facilitando a transição para abordagens iterativas.
5. **Reduz burocracia**
   - Substitui documentação complexa por narrativas objetivas, acelerando o início do desenvolvimento.

"Verdade universal #3: A mudança vai acontecer

É inevitável que os requisitos mudem. As necessidades do negócio evoluem, identificam-se novos utilizadores ou mercados, as regras do negócio ou a regulamentação imposta pelos governos são atualizadas, e os ambientes de operação mudam com o tempo. [Para além disso] os requisitos ficam mais claros para os *stakeholders* à medida que eles vão sendo solicitados a pensar com atenção sobre o que pretendem realmente fazer com o produto." [1]

O autor alerta para uma mudança de atitude por parte da engenharia de software, face às prioridades tradicionais quanto à a fixação dos requisitos de um sistema. Que atitude é essa? Como é que os métodos de desenvolvimento podem dar o repetivo suporte?

---

[1] Adaptado do texto original: "Cosmic Truth #3: Change happens.

It's inevitable that requirements will change. Business needs evolve, new users or markets are identified, business rules and government regulations are updated, and operating environments change over time. Requirements become clearer as the key stakeholders are prompted to think more carefully about what they really are trying to do with the product."

Karl Wiegers, "Ten Cosmic Truths About Software Requirements", available from: https://medium.com/analysts-corner/ten-cosmic-truths-about-software-requirements-edd33292a456

Tópicos de resposta pretendidos:

- Nova atitude: aceitar que é normal os requisitos de um projeto de desenvolvimento sofrerem alterações, quer por alterações do contexto (e.g.: mercado), quer pela alteração da perceção do projeto [pelos *stakeholders*]. O **objetivo não deve ser impedir a mudança, mas assegurar que o projeto dispões de mecanismos para acolher as alterações** necessárias, pelas razões corretas.
- Para se poder adaptar às novas condições, o processo de desenvolvimento deve adotar práticas que facilitem a clarificação dos requisitos e integração de alterações, designadamente:
  - Promover o desenvolvimento evolutivo, com entrega frequente de incrementos a funcionar, obtendo o feedback regular do promotor.
  - Prever a (possibilidade de) revisão do plano e das prioridades, por exemplo, adotando um desenvolvimento iterativo, por ciclos.
  - Em vez de tentar obter todos os requisitos e "congelá-los" no início do projeto, trabalhar num primeiro conjunto de requisitos como base, usando o que é conhecido na altura. Estes requisitos são priorizados e implementados incrementalmente. Os novos requisitos podem ser incluídos, de forma evolutiva.
- A abertura à alteração de requisitos ("*embrance change*") é própria dos métodos ágeis, por oposição às abordagens lineares tradicionais, que procuravam mitigar o risco tentando "congelar" a especificação inicial.