Unochapecó

Mateus Miranda da Conceição

Trabalho Engenharia de Software

CHAPECÓ

2023

Crise de Software

A crise de software pode ser caracterizada pela dificuldade de gerenciar e produzir software ou aplicações escaláveis, dentro de prazos e orçamentos estipulados. A crise de software é causada por diversos fatores, como a falta de padronização de uma arquitetura, falta de documentação, a complexidade do sistema, pressão para entregar resultados rapidamente e a grande curva de aprendizado para que os profissionais possam manter o sistema e incrementar novas funcionalidades.

A crise de software é um problema que se tornou evidente na década de 1960, quando a demanda por software começou a crescer rapidamente. Ela é caracterizada pela dificuldade de produzir sistemas e aplicações escaláveis e de qualidade dentro de prazos e orçamentos limitados. Esse problema é causado por diversos fatores, que incluem:

1. Falta de padronização e documentação: Falta de documentações e padrões de arquitetura ou na escrita das funcionalidades torna difícil a compreensão do software por parte dos desenvolvedores e usuários. Isso resulta em um processo de desenvolvimento do software mais lento e a cada nova funcionalidade adicionada aumenta a curva de aprendizagem para os futuros desenvolvedores.
2. Complexidade do sistema: Os sistemas atuais estão cada vez mais robustos e complexos, com múltiplas camadas de tecnologias e integração com outros softwares. O alto nível de acoplamento entre as partes do software pode aumentar o nível de complexidade do software. Essa complexidade dificulta a manutenção do software e pode levar a erros e falhas.
3. Falta de qualificação dos profissionais: A falta de profissionais qualificados em desenvolvimento de software é um fator que contribui para a crise de software. A dificuldade de encontrar profissionais que possuam o conhecimento necessário para lidar com a complexidade do sistema produzido e as tecnologias utilizadas.
4. Pressão por resultados rápidos: A pressão por entregar o sistema rapidamente pode levar a um processo de desenvolvimento apressado, com pouca atenção à qualidade do software produzido. Isso pode resultar em software mal projetado, com problemas de desempenho e segurança.

Para solucionar a crise de software, diversas metodologias foram desenvolvidas ao longo das últimas décadas. Entre elas, estão:

1. Metodologias ágeis: As metodologias ágeis buscam entregar software rapidamente, por meio de ciclos curtos de desenvolvimento e entregas de valor. Elas priorizam a colaboração entre as equipe de desenvolvimento e os clientes, visando atender as necessidades dos usuários de forma eficiente.
2. Engenharia de software: A engenharia de software se concentra na aplicação de técnicas e processos para a produção de um sistema de qualidade. Isso inclui a utilização de práticas de documentação, testes e revisões de código, bem como a utilização de ferramentas de desenvolvimento e gerenciamento de projetos.
3. Comunicação e colaboração: A comunicação e colaboração entre as equipes de desenvolvimento e as partes interessadas no projeto são essenciais para o sucesso do desenvolvimento de software.

A crise de software é um problema complexo que envolve diversos fatores, incluindo a complexidade dos sistemas, a falta de padronização e documentação, a falta de qualificação dos profissionais e a pressão por resultados rápidos. Para solucioná-la, é necessário adotar técnicas e metodologias que visem à produção de software de qualidade, bem como promover a comunicação e colaboração entre as equipes de desenvolvimento e as partes interessadas no projeto.

Requisitos de software

Requisitos de software são especificações das funcionalidades, desempenho, segurança, qualidade e outras características que um software deve possuir para atender as necessidades dos usuários. Eles descrevem o que o software deve fazer, como deve funcionar e quais restrições e limitações deve obedecer. Os requisitos são identificados durante o processo de levantamento de requisitos, que envolve a identificação das necessidades e expectativas dos usuários, a definição dos objetivos e metas do software e a avaliação das restrições e limitações do ambiente em que ele será utilizado. Esses requisitos são documentados em um documento de requisitos, que serve como um guia para a equipe de desenvolvimento na construção do software.

Os requisitos de software podem ser divididos em duas categorias principais:

1. Requisitos funcionais: Requisitos funcionais de software são as especificações que o sistema deve possuir, ou seja, são as descrições das tarefas que o

software deve ser capaz de executar para atender as necessidades e expectativas do usuário ou do cliente. Esses requisitos descrevem como o software deve se comportar em determinadas situações, quais ações deve executar em resposta a uma entrada de dados específica. Eles podem ser exemplificados através de diagramas, modelos, casos de uso ou linguagem natural. É importante que os requisitos funcionais sejam claros, concisos e precisos, para que possam ser compreendidos pela a equipe que irá desenvolver o software. Os requisitos devem ser priorizados, de forma que as funcionalidades mais importantes sejam desenvolvidas primeiro.

2. Requisitos não funcionais: Requisitos não funcionais de software são os requisitos que não estão diretamente relacionados às funcionalidades do software, mas sim as suas características de qualidade, desempenho, segurança, usabilidade, experiência do usuário, requisitos de manutenção, como a capacidade de ser facilmente mantido e atualizado, entre outros. Esses requisitos são importantes para o processo de desenvolvimento de software, pois contribuem para a qualidade e eficácia do software, eles também devem ser testados para garantir que o software atenda a esses requisitos.

A definição adequada desses requisitos é essencial para o sucesso do projeto a ser desenvolvido, pois ajudará a garantir que o software atenda às expectativas e necessidades do cliente. Além disso, uma boa documentação dos requisitos pode ajudar a evitar problemas e atrasos durante o processo de produção do sistema, tornando o processo mais eficiente e econômico.