UNIVERSIDADE COMUNITÁRIA DA REGIÃO DE CHAPECÓ

CURSO DE CIÊNCIA DA COMPUTAÇÃO

RAFAEL PEREIRA LUIZ

DISCIPLINA DE ENGENHARIA DE SOFTWARE I

CRISE DE SOFTWARE

PROFESSOR RADAMES PEREIRA

CHAPECÓ
2023

A Crise de Software é um problema complexo que afeta a indústria de software há décadas e é causada pela crescente complexidade dos sistemas de software e pela falta de metodologias adequadas para gerenciá-la. Com o aumento da complexidade, torna-se cada vez mais difícil manter a qualidade do software, cumprir prazos de entrega e manter os custos sob controle.

Para lidar com a Crise de Software, foram desenvolvidas diversas abordagens e metodologias, incluindo o desenvolvimento ágil de software, a engenharia de software orientada a objetos e a engenharia de software baseada em componentes. Essas abordagens buscam reduzir a complexidade dos sistemas de software, dividindo-os em componentes menores e mais gerenciáveis e permitindo um desenvolvimento mais interativo e colaborativo.

Além disso, novas ferramentas e tecnologias estão sendo desenvolvidas para ajudar a enfrentar a Crise de Software. A automação de testes, por exemplo, permite que os desenvolvedores testem o software de forma mais rápida e precisa, enquanto a inteligência artificial pode ser usada para detectar e corrigir problemas de software em tempo real.

Técnicas de modelagem e simulação também podem ser usadas para avaliar a qualidade do software, permitindo que os desenvolvedores identifiquem problemas antes que eles se tornem mais caros e difíceis de corrigir. Além disso, a utilização de boas práticas de gerenciamento de projetos, como a definição clara de requisitos e a comunicação efetiva entre as equipes de desenvolvimento, pode ajudar a garantir que o software seja entregue no prazo e dentro do orçamento.

Apesar dos esforços para enfrentar a Crise de Software, ainda há desafios a serem superados. Por exemplo, a segurança do software é uma preocupação crescente, especialmente em sistemas críticos, como em aplicações médicas e financeiras. A indústria de software também precisa lidar com a falta de mão de obra qualificada em algumas áreas, o que pode dificultar o desenvolvimento de sistemas complexos.

Em resumo, a Crise de Software é um desafio contínuo para a indústria de software, mas com as abordagens e tecnologias adequadas, é possível superá-la. Os desenvolvedores podem reduzir a complexidade dos sistemas de software e melhorar a qualidade do software com o uso de metodologias adequadas e ferramentas inovadoras. Ainda assim, a indústria de software deve permanecer vigilante e adaptar-se às novas tendências e desafios que surgem constantemente.

Requisitos funcionais são as funcionalidades e comportamentos esperados que o software deve apresentar para atender às necessidades dos usuários. Eles devem ser claros, precisos e mensuráveis, além de serem documentados e validados junto aos usuários e stakeholders do projeto. Esses requisitos são essenciais para garantir que o software atenda às necessidades e expectativas dos usuários e stakeholders e devem ser verificados durante todo o processo de desenvolvimento.

- Um e-Commerce deve permitir que os usuários pesquisem e visualizem produtos, adicione itens ao carrinho de compras, realizem pagamentos e receba confirmações de pedidos.
- Um software de gerenciamento de projetos deve permitir que os usuários criem tarefas, atribuam responsáveis, definam prazos e monitorem o progresso do projeto.
- Um aplicativo de música deve permitir que os usuários busquem e reproduzam músicas, criem playlists e recebam recomendações de músicas com base em seus gostos.

Os requisitos não funcionais são as características ou propriedades que o software deve ter, além das suas funcionalidades, para atender às necessidades dos usuários e às expectativas dos stakeholders. Esses requisitos se concentram em como o software deve funcionar, em vez do que ele deve fazer. Eles descrevem as características de qualidade que o software deve ter, como desempenho, segurança, confiabilidade, usabilidade, escalabilidade, entre outros.

Alguns exemplos de requisitos não funcionais são:

- Desempenho: o software deve ser capaz de realizar suas funções dentro de um tempo razoável, sem sobrecarregar o hardware ou a rede.
- Segurança: o software deve ser capaz de proteger os dados dos usuários e do sistema, prevenindo o acesso não autorizado, o roubo de informações ou a perda de dados.
- Usabilidade: o software deve ser fácil de usar, com interface intuitiva, organizada e amigável para o usuário.
- Confiabilidade: o software deve ser capaz de executar suas funções de forma consistente e sem falhas.
- Escalabilidade: o software deve ser capaz de lidar com um grande volume de dados e usuários, sem comprometer seu desempenho ou funcionalidade.