**Análise de Sistemas Orientada a Objetos**

Durante o desenvolvimento do sistema Clickdesk, adotou-se a análise orientada a objetos como principal abordagem para compreender as necessidades dos usuários e transformá-las em soluções funcionais. Essa metodologia organiza o sistema em torno de objetos, que representam elementos reais do contexto, como chamados, usuários, técnicos, entre outros. Cada objeto possui suas próprias características (atributos) e comportamentos (métodos), facilitando tanto o entendimento quanto a construção da solução (PRESSMAN, 2016; SOMMERVILLE, 2019).

Para representar essas informações de maneira clara e estruturada, foram utilizados os diagramas UML, que são fundamentais para visualizar a arquitetura e o funcionamento do sistema. Eles também desempenham um papel importante na comunicação entre a equipe de desenvolvimento, além de ajudar a minimizar erros durante a implementação (BOOCH; RUMBAUGH; JACOBSON, 2006).

O primeiro diagrama elaborado foi o Diagrama de Casos de Uso, que permite mapear as funcionalidades que o usuário poderá executar no sistema. Nesse contexto, ações como abrir um chamado, acompanhar seu andamento, interagir com o atendimento ou encerrar o chamado foram representadas, proporcionando uma visão clara dos processos envolvidos na utilização do sistema (LARMAN, 2007).

Em seguida, foi criado o Diagrama de Classes, que permitiu estruturar de forma precisa os dados e suas inter-relações. As classes definidas, como “Chamado”, “Usuário”, “Técnico” e “Categoria”, mostraram como os elementos do sistema se conectam. Esse diagrama foi essencial tanto para a definição do modelo de dados quanto para orientar o desenvolvimento do código (LARMAN, 2007; BOOCH; RUMBAUGH; JACOBSON, 2006).

Posteriormente, o Diagrama de Sequência foi elaborado, permitindo entender a dinâmica da comunicação entre os objetos ao longo do tempo. Por exemplo, ao abrir um chamado, o fluxo prevê que uma inteligência artificial (IA) fará a análise inicial; caso consiga solucionar o problema, retorna uma resposta ao usuário, caso contrário, direciona o chamado para um técnico. Esse entendimento sequencial é crucial para definir a lógica de desenvolvimento (PRESSMAN, 2016).

Por fim, elaborou-se o Diagrama de Implantação, que detalha a infraestrutura do sistema, mostrando onde cada componente será hospedado — como servidores, frontend, backend e banco de dados —, além de ilustrar como ocorre a comunicação entre eles. Esse diagrama é particularmente relevante para a equipe responsável pela infraestrutura e pela manutenção do ambiente de produção (SOMMERVILLE, 2019).

Além da modelagem, foi realizado o levantamento completo dos requisitos do sistema, seguindo as boas práticas recomendadas pela literatura (ALENCAR; BARBOSA, 2018; IEEE, 1998). Esse levantamento foi dividido em três categorias principais:

Requisitos dos usuários: representam as expectativas e necessidades práticas dos usuários, como rapidez no atendimento, acompanhamento do histórico de chamados e integração com chatbot.

Requisitos funcionais: descrevem as funções que o sistema deve oferecer, como abertura e gerenciamento de chamados, categorização automática com IA, acesso à base de conhecimento, encaminhamento para técnicos e geração de relatórios.

Requisitos não funcionais: englobam aspectos técnicos que garantem a qualidade do sistema, como desempenho, segurança, capacidade de atender múltiplos usuários simultaneamente, facilidade de uso e disponibilidade.

Portanto, a utilização da análise orientada a objetos, combinada com a UML e um levantamento criterioso dos requisitos, tornou o processo de desenvolvimento do Clickdesk mais organizado, eficiente e com maior qualidade, reduzindo significativamente os riscos de falhas no produto.

1. PRESSMAN, Roger S. Engenharia de Software: uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2016.

2. SOMMERVILLE, Ian. Engenharia de Software. 10. ed. São Paulo: Pearson, 2019.

3. BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML: Guia do Usuário. Rio de Janeiro: Elsevier, 2006.

4. LARMAN, Craig. Utilizando UML e Padrões: uma introdução à análise e ao projeto orientado a objetos. 3. ed. Porto Alegre: Bookman, 2007.

5. ALENCAR, Felipe; BARBOSA, Bruno. Requisitos de Software: uma abordagem prática. São Paulo: Novatec, 2018.

6. IEEE. IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications. IEEE, 1998.