RESENHA DO ARTIGO **BIG BALL OF MUD**

Aluno: Alfredo Luis Vieira

Este artigo trata sobre um assunto arquitetural no desenvolvimento de software, o autor fala sobre o padrão “arquitetural” chamado BIG BALL OF MUD (Grande bola de lama), que de acordo com o mesmo, é muito utilizado hoje em dia. Inicialmente, o autor fala sobre o contexto das equipes e organizações de desenvolvimento, cita as questões de prazo e dinheiro investidos no projeto. É de se concordar que atualmente, prazos e custos são variáveis muito consideráveis no desenvolvimento, sendo estas, capazes de ditar o rumo do projeto, podemos afirmar que estas variáveis são definidas por fatores externos a organização (relacionados ao cliente), onde a sua necessidade e orçamento, podem fazer com que decisões difíceis sejam tomadas no projeto.

O autor foca muito na questão da arquitetura no projeto, o mesmo diz que esta é uma questão que na maioria das vezes é deixada em segundo plano durante a etapa de planejamento do projeto, esta afirmação pode ser considerada verdadeira, visto que os benefícios da implementação de uma boa arquitetura no sistema começa a trazer resultados aparentes a longo prazo, e isso faz com que os muitos times optem por decisões que trazem benefícios a curto prazo, mas que a longo prazo, vão causar erosões no projeto, aumentando os custos de manutenção e escalabilidade. Podemos dizer que isto é uma faca de dois gumes, e que a decisão mais óbvia em termos de custo benefício a curto prazo, não é a mais saudável para o projeto como um todo, ou seja, optar por possuir uma boa arquitetura implementada na aplicação é uma tarefa difícil que demanda diversos pré-requisitos como custo, experiencia e tempo, levando em conta que os benefícios de uma boa estruturação no projeto, só vão ser realmente vistos a longo prazo, essa prática é pouco comum e acaba fazendo com que o código vire uma grande bola de lama.

Nestes processos de desenvolvimento, o macarrão formado pelo código é exponencial, à medida que novas funcionalidades e linhas de código são “jogadas” sem documentação para a implementação, o problema só vai aumentando, e os responsáveis pelas refatorações e manutenções futuras ficam com uma grande bomba plantada e pronta pra explodir.
O artigo também fala sobre a etapa de expansão, que na maioria das vezes, pode acontecer sem a devida documentação, como um crescimento desordenado ou sem planejamento, apenas focando e deixar tudo funcionando, obviamente, isso pode gerar muita dor de cabeça no futuro, nas etapas de expansão e refatoração (podemos falar que estas práticas são nocivas até no presente, como, por exemplo, quando um funcionário novo é designado para trabalhar no projeto, o mesmo teria grandes dificuldades para entender o sistema, pois, o mesmo carece

de documentação, e seu código este totalmente desorganizado e misturado com um grande espaguete).

Uma comparação interessante realizada pelo autor, foi o clico de vida dos softwares com o ciclo de vida das cidades, o mesmo comparou o nascimento e o crescimento de cidades planejadas e não planejadas. Foi utilizado como um dos exemplos, a cidade de Brasília, que foi planejada no papel previamente a sua construção, e também outras cidades, como a maioria das existentes hoje em dia, foram crescendo com o tempo e surgiram a partir de pequenas construções. Essa comparação é totalmente válida, pois, podemos notar estes comportamentos em aplicações, existindo casos onde uma aplicação tem sua primeira versão, um sistema pequeno, mas que com o tempo e seu crescimento “desordenado”, vai virando um amontoado de código. Uma exemplificação que poderia ser citada neste artigo, e que é totalmente válida nesta comparação, são os sistemas internos das cidades, com os de segurança, trânsito e coleta de lixo, estes sistemas podem apresentar dificuldades com o crescimento da cidade e da sua população, tal como os componentes de um software, que podem apresentar mau funcionamento a medida que o mesmo vai escalando.

Após nos introduzir o problema, o autor nos dá algumas alternativas para dissolver esta grande bola de lama, o mesmo nos diz que as partes mais acopladas e mais problemáticas devem ser isoladas, e trabalhadas, sendo assim, um planejamento de arquitetura deve ser pensado para as mesmas, ou seja, o autor defende uma fragmentação e resolução de problemas menores, um de cada vez. Sobre esta afirmação, eu não concordo 100%, acredito que o grande problema deve ser fragmentado, mas, ao mesmo tempo, penso que uma arquitetura em alto nível geral sobre o sistema deve ser pensado, para que o problema seja resolvido “de uma vez”, e que na parte da implementação, deve ser trabalhada em fragmentos, o famoso dividir para conquistar. Acredito que esta estratégia de projetar uma arquitetura única para o sistema como um todo seja a mais ideal, pois, pode evitar problemas futuros de integração e de manutenção.

E por último, o autor fala sobre a alternativa de começar do zero, onde o mesmo especifica condições para que esta alternativa seja escolhida, acredito que esta opção é importante de se ter na mesa, mas deve-se realizar uma análise completa sobre o contexto do sistema, o contexto do projeto (orçamentos e prazos) e a decisão de começar ou não do zero seja com certeza a mais sábia possível (visto que esta decisão é difícil, e possui diversas variáveis que podem influenciar, sendo elas, o time de desenvolvimento, as necessidades do cliente, manutenção do software, etc.).