Resenha artigo Big Ball of Mud.

O artigo “Big Ball of Mud” de Brian Foote e Joseph Yoder, traz dois principais questionamentos, sendo eles o porquê de tantos sistemas distintos implementarem esta arquitetura sem padrão, e o'que podemos fazer para melhorar estes sistemas existentes, pois baseado no tamanho e no nível de acoplamento deste sistema, a manutenibilidade e comprometida.

O termo “Big Ball of Mud”, se trata de um sistema muito acoplado, praticamente sem arquitetura, onde todas as camadas se comunicam entre si, além de não existir padrões de projeto e padrões de código, e assim tornando o projeto uma verdadeira selva de código. Além de que o entendimento deste código se torna algo complexo, e baseado nisto temos um projeto que não tem maturidade, ou seja, o custo de manutenção aumenta, pois não é possível que pessoas menos experientes trabalhem nele.

E apresentado seis padrões que não são considerados anti padrões, baseado no senso comum, são eles “Big Ball of Mud”, “Throwaway Code”, “Piecemeal Growth”, “Keep It Working”, “Sweeping It Under the Rug”, “Reconstruction”. Entretanto, um sistema não nasce como uma “Big Ball of Mud”, e sim se transforma em um no decorrer de sua existência.

Um dos casos apresentados é quando uma funcionalidade é desenvolvida de forma rápida e mal feita para ser usada uma vez é descartada, mas devido à falta de documentação neste projeto esta funcionalidade acaba se tornando parte dele. Onde quando houver a necessidade de uma correção o código não será refeito, mas sim adaptado para a nova necessidade, então se torna um “Throwaway Code”. Esta mudança de código provisório para código permanente é algo que vemos muito no mercado, principalmente em correções de bugs feitas por pessoas menos experientes.

A situação anterior pode também estar presente em um cenário onde temos um sistema de sucesso que passou por mudanças de requisitos e para atendê-los rápido, e sem a necessidade de um reestruturação a arquitetura do projeto começa a ser minada. Um exemplo disto seria em um projeto que utiliza a arquitetura MVC, a camada da View começa a se comunicar diretamente com o Model, começando assim a quebrar as regras desta arquitetura.

Logo em uma situação hipotética onde se possui um sistema de cadastro, e uma nova funcionalidade foi solicitada, mas devido ao pouco tempo disponível para o desenvolvimento. A arquitetura será deixada de lado e a persistência dos dados dessa nova funcionalidade será feita no controller e não no Model. E ao fazer isto estaria quebrando assim as regras da arquitetura, tendo assim um exemplo prático de como é fácil de se começar um “Piecemeal Growth”.

Entretanto, o “Piecemeal Growth” não é o único anti padrão que pode minar sua arquitetura e tornar o seu software uma “Big Ball of Mud”. O “Keep it working” é outro exemplo, que se refere a um momento do desenvolvimento, onde uma manutenção é identificada, mas o tempo que será necessário para realizá-la não é proporcional ao tempo que está disponível para a equipe. É novamente devido ao fator tempo essa manutenção ou refatoração será substituída por um código ruim que servirá apenas para manter a aplicação rodando, mesmo que condenando a longo prazo.

Outro termo abordado pelo artigo e “Reconstruction”, este termo se trata do resultado de um código sujo, sem arquitetura e tenha se tornado de fato uma “Big Ball of Mud” de uma proporção tão grande, que já não se trata de um software que a compreensão e a manutenibilidade será difícil de demorada. Para que a solução para um software seja a sua reconstrução é necessário que isso tenha se impossibilitando de se fazer. Além de claro que o preço de se começar do zero é mais barato do que de se tentar salvar o atual sistema. Infelizmente temos que refazer um software devido a problemas arquitetônicos e que tenham se tornado uma “Big Ball of Mud”, é algo muito comum mesmo atualmente, onde temos processos de desenvolvimentos que deveriam garantir uma qualidade e uma longevidade aos softwares.

Mesmo após vinte e sete anos da publicação deste artigo ainda temos os mesmos problemas apontados em relação à banalização da importância da Arquitetura de um software, onde ainda é visto como um luxo, e não como uma fase primordial. Comparando com a Engenharia Civil, a Arquitetura de um software deveria ser vista com mesma importância que a fundação de um edifício, pois ela que irá possibilitar ou não a escalabilidade e manutenibilidade do software, assim como a fundação de um edifício definirá suas características.

Como resultado da visão apresentada acima, o artigo apresenta também a ideia de que os softwares da época possuíam sua arquitetura um pouco mais bem definida que uma “favela”. Esta comparação acontece, pois em muitos casos os sistemas utilizam frameworks que foram descontinuadas, partes do sistema crescem sem o devido controle, além de utilizarem ferramentas que atualmente são consideradas primitivas para a gestão e manutenção deste software.

Na conclusão, o artigo reconhece que mesmo o “Big Ball of Mud” seja um anti padrão ainda possui uma certa eficácia em certos contextos. Entretanto, reforçam que entender o que leva a um projeto se manter ou chegar neste anti padrão é algo essencial para que em projetos futuros os mesmo erros não aconteçam. Fazendo assim que o surgimento de softwares com arquiteturas corretas começassem a aparecer com mais recorrência.