Resenha: Documentando Decisões de Arquitetura por Michael Nygard

O artigo "Documenting Architecture Decisions" de Michael Nygard, publicado em 15 de novembro de 2011, aborda uma questão crucial e recorrente no desenvolvimento de software, especialmente em projetos que seguem metodologias ágeis: a documentação da arquitetura. Nygard propõe uma abordagem pragmática e eficiente para registrar as decisões arquiteturais de forma que agreguem valor contínuo ao projeto e à equipe, em vez de se tornarem um fardo obsoleto.

O Problema: A Ineficácia da Documentação Tradicional

Nygard inicia destacando a inadequação dos métodos tradicionais de documentação em ambientes ágeis. Projetos ágeis são caracterizados pela evolução contínua, onde nem todas as decisões são tomadas no início. A documentação extensa e monolítica, comum em modelos mais rígidos, rapidamente se torna desatualizada, e o esforço para mantê-la é raramente justificado pelo seu valor. O autor ressalta que "ninguém jamais lê documentos grandes", uma afirmação com a qual muitos desenvolvedores podem se identificar.

O cerne do problema, no entanto, reside na dificuldade de rastrear a **motivação** por trás das decisões tomadas. Um novo membro da equipe, ao se deparar com uma decisão de arquitetura, enfrenta um dilema:

1. **Aceitar cegamente:** Uma atitude que pode perpetuar uma decisão que não é mais válida devido a mudanças no contexto do projeto, levando a um acúmulo de "dívida técnica" e ao medo de realizar alterações.
2. **Mudar cegamente:** Uma ação arriscada que, sem a compreensão das forças e consequências originais, pode comprometer requisitos não-funcionais ou outros aspectos importantes do sistema.

Ambas as opções são subótimas e podem levar a um projeto estagnado ou a retrocessos inesperados.

A Solução: Registros de Decisão de Arquitetura (ADRs)

Para resolver esse impasse, Nygard propõe a utilização de **Registros de Decisão de Arquitetura (Architecture Decision Records - ADRs)**. A ideia é manter uma coleção de registros para decisões "arquiteturalmente significativas", ou seja, aquelas que afetam a estrutura, características não-funcionais, dependências, interfaces ou técnicas de construção do software.

Um ADR é um arquivo de texto curto e modular, com um formato simples e direto, inspirado nos "Alexandrian patterns". Cada registro documenta uma única decisão, o contexto que a motivou e as suas consequências. A proposta é que esses registros sejam mantidos junto ao código-fonte do projeto (por exemplo, em `doc/arch/adr-NNN.md`), utilizando uma linguagem de formatação leve como Markdown.

Os ADRs devem ser numerados sequencialmente, e um número nunca deve ser reutilizado. Se uma decisão for revertida, o ADR original é mantido, mas marcado como "superado", preservando o histórico da evolução da arquitetura.

A Estrutura de um ADR

Nygard sugere um formato com seções bem definidas para garantir que cada ADR seja fácil de ler e entender:

- **Título:** Uma frase curta e descritiva. Exemplo: "ADR 9: Uso de LDAP para Integração Multitenant".
- **Contexto:** Descreve as forças em jogo que levaram à necessidade da decisão. Isso inclui fatores tecnológicos, políticos, sociais e específicos do projeto. A linguagem nesta seção deve ser neutra, apenas descrevendo os fatos.
- **Decisão:** A resposta a essas forças, declarada de forma clara e direta, usando voz ativa. Exemplo: "Nós iremos utilizar..."
- **Status:** Indica o estado da decisão: "proposta", "aceita", "depreciada" ou "superada" (com referência ao ADR que a substituiu).

- **Consequências:** Descreve o contexto resultante após a aplicação da decisão. É crucial listar todas as consequências, sejam elas positivas, negativas ou neutras, pois todas afetam o futuro do projeto e da equipe.

O documento inteiro deve ser conciso, idealmente com uma ou duas páginas, escrito como se fosse uma conversa com um futuro desenvolvedor.

Impacto e Experiência Prática

Uma das grandes vantagens dos ADRs é que as consequências de uma decisão frequentemente se tornam o contexto para decisões futuras, criando uma narrativa coesa da evolução da arquitetura. Isso garante que a motivação por trás das escolhas permaneça visível para todos os membros da equipe, presentes e futuros, eliminando a confusão e o questionamento sobre "o que eles estavam pensando?".

O próprio artigo é formatado como um ADR, servindo como um exemplo prático da sua aplicação. Nygard relata que, após a implementação da prática em alguns projetos, o feedback de clientes e desenvolvedores foi extremamente positivo. Novos membros da equipe, em particular, apreciaram o grau de contexto que obtiveram ao ler os ADRs. A prática se mostrou especialmente útil para capturar intenções de longo prazo, evitando que decisões atuais dificultem futuras reestruturações planejadas.

Uma possível objeção seria a acessibilidade desses documentos para stakeholders não-técnicos. No entanto, Nygard argumenta que o uso de repositórios como o GitHub, que renderiza arquivos Markdown de forma amigável, torna os ADRs tão acessíveis quanto uma página de wiki.

Conclusão

"Documenting Architecture Decisions" apresenta uma solução elegante e de baixo custo para um problema complexo e difundido no desenvolvimento de software. Os Architecture Decision Records (ADRs) oferecem um método leve e eficaz para documentar o que realmente

importa: o **porquê** das decisões arquiteturais. Ao focar no contexto, na decisão e nas suas consequências, os ADRs transformam a documentação de uma tarefa burocrática e sem valor em uma ferramenta viva e útil, que capacita as equipes a construir e evoluir sistemas de software de forma mais consciente e sustentável. É uma leitura essencial para arquitetos, líderes técnicos e desenvolvedores que buscam melhorar a comunicação e a longevidade de seus projetos.