O artigo *Documenting Architecture Decisions*, de Michael Nygard, propõe um método prático e enxuto para registrar e comunicar decisões arquiteturais em projetos de software, especialmente em contextos ágeis. O autor parte da constatação de que a arquitetura em projetos ágeis deve ser tratada de forma iterativa e evolutiva, já que nem todas as decisões são tomadas no início e muitas delas precisam ser revistas ao longo do tempo.   
  
 Assim, ele critica a ineficiência de grandes documentos de arquitetura difíceis de manter e raramente consultados e defende o uso de registros pequenos, modulares e atualizáveis, que agreguem valor real ao time de desenvolvimento.

Nygard identifica um problema recorrente em projetos de software: a perda do conhecimento sobre o **racional** das decisões arquiteturais. Quando novos membros ingressam no projeto, deparam-se frequentemente com escolhas cujo motivo não está claro, o que leva a dois comportamentos igualmente prejudiciais: aceitar cegamente decisões antigas, mesmo quando o contexto já mudou, ou revertê-las sem compreender suas consequências. Ambos os casos comprometem a evolução e a estabilidade do sistema. Para evitar esse cenário, o autor propõe a criação de registros curtos chamados Architecture Decision Records (ADRs) documentos que explicam, de forma estruturada, o contexto, a decisão tomada e suas implicações.

Cada ADR deve conter cinco seções: título, contexto, decisão, status e consequências. O *contexto* descreve as forças e restrições que motivaram a escolha; a *decisão* apresenta a resposta adotada pela equipe; o *status* indica se a decisão está proposta, aceita ou superada; e as *consequências* registram os efeitos esperados — positivos, negativos ou neutros. Esses documentos devem ser breves, escritos em linguagem clara e organizados como se dialogassem com um futuro desenvolvedor. O autor enfatiza que cada ADR deve tratar de uma única decisão arquitetural significativa, e que as consequências de uma decisão frequentemente formam o contexto para decisões futuras, criando uma espécie de “linguagem de padrões” dentro do projeto.

Os ADRs são armazenados junto ao código-fonte, geralmente em repositórios Git, o que garante fácil acesso e versionamento. Segundo Nygard, essa prática aumenta a transparência e o compartilhamento de conhecimento entre todos os envolvidos no projeto, inclusive quando há rotatividade de equipe. Além disso, o formato em Markdown torna os documentos legíveis tanto para desenvolvedores quanto para gestores, eliminando a necessidade de ferramentas complexas de documentação.

No relato de experiência apresentado, o autor observa resultados positivos após a adoção dos ADRs em vários projetos. Desenvolvedores relataram melhor compreensão do histórico e das intenções arquiteturais, enquanto clientes reconheceram o valor da clareza e rastreabilidade das decisões. Nygard destaca que essa abordagem também auxilia na preservação de planos de longo prazo, especialmente em projetos que estão em transição para novas arquiteturas. Apesar de um possível obstáculo de acesso para não desenvolvedores, a integração com plataformas como o GitHub facilita o compartilhamento e a atualização contínua dos registros.

Concluindo, Michael Nygard argumenta que os Architecture Decision Records representam uma forma ágil, eficiente e sustentável de documentação arquitetural. Eles equilibram a necessidade de registrar conhecimento técnico com a flexibilidade exigida por projetos iterativos, assegurando que as decisões façam parte da memória viva do sistema. Assim, o autor reforça a importância de documentar não apenas “o que foi decidido”, mas também “por que foi decidido”, promovendo a transparência, a aprendizagem contínua e a evolução consciente da arquitetura de software.