Resenha: Managing Technical Debt

O white paper "Managing Technical Debt" de Steve McConnell, Chief Software Engineer da Construx Software, publicado em junho de 2008, aborda o conceito de "Dívida Técnica" no desenvolvimento de software. O documento explora o que é a dívida técnica, seus diferentes tipos, como ela é contraída, como gerenciá-la e, crucialmente, como comunicá-la a stakeholders não técnicos.

Introdução à Dívida Técnica

McConnell inicia o artigo definindo Dívida Técnica como "trabalho técnico atrasado que é incorrido quando atalhos técnicos são tomados, geralmente em busca de cronogramas de software orientados pelo calendário" [1]. Ele traça um paralelo direto com a dívida financeira, onde algumas dívidas podem servir a propósitos de negócios valiosos, enquanto outras são contraproducentes. A capacidade de uma organização de contrair, rastrear, gerenciar e pagar sua dívida varia.

Tipos de Dívida Técnica

O autor categoriza a dívida técnica em dois tipos principais:

1. **Dívida Não Intencional (Tipo I):** Incorrida sem querer, geralmente devido a um trabalho de baixa qualidade, como uma abordagem de design que se revela propensa a erros ou código mal escrito por um programador júnior. Esta dívida é um resultado não estratégico de um trabalho deficiente.
2. **Dívida Intencional (Tipo II):** Incorrida conscientemente, quando uma organização decide otimizar o presente em detrimento do futuro. Exemplos incluem adiar a reconciliação de bancos de dados ou a escrita de testes unitários para cumprir prazos de lançamento. O white paper foca principalmente neste tipo de dívida.

Dentro da Dívida Intencional, McConnell distingue entre:

- **Dívida de Curto Prazo (Tipo II.A):** Assumida taticamente e reativamente, geralmente como uma medida de última hora para lançar um produto. Pode ser:
    - **Dívida de Curto Prazo Focada (Tipo II.A.1):** Atalhos identificáveis individualmente, como a decisão de "apenas hackear e consertar depois do lançamento". É comparada a um empréstimo de carro, grande e rastreável.
    - **Dívida de Curto Prazo Não Focada (Tipo II.A.2):** Acumulada a partir de centenas ou milhares de pequenos atalhos, como nomes de variáveis genéricos, comentários esparsos ou não seguir convenções de codificação. É comparada à dívida de cartão de crédito, fácil de incorrer e difícil de rastrear e gerenciar. McConnell enfatiza que este tipo deve ser evitado, pois não compensa nem no curto prazo.
- **Dívida de Longo Prazo (Tipo II.B):** Assumida estrategicamente e proativamente, como investir em novos equipamentos. No contexto técnico, pode ser a decisão de não suportar uma segunda plataforma por alguns anos, adiando o custo de desenvolvimento.

Incorrendo Dívida Técnica

A razão fundamental para incorrer dívida técnica estratégica é que o custo do trabalho de desenvolvimento hoje é visto como mais caro do que será no futuro. Isso pode ser devido a:

- **Tempo de Lançamento (Time to Market):** Em mercados críticos, atrasar o lançamento pode resultar em perdas de receita muito maiores do que o custo futuro de pagar a dívida.
- **Preservação de Capital Inicial:** Startups com capital limitado podem adiar despesas para pagá-las com mais recursos no futuro.
- **Atrasar Despesas de Desenvolvimento:** Quando um sistema é desativado, toda a sua dívida técnica é "aposentada" com ele. Próximo ao fim da vida útil de um sistema, torna-se difícil justificar investimentos em algo que não seja o mais expedito.

Serviço da Dívida e Transparência

Assim como a dívida financeira, a dívida técnica exige "serviço", ou seja, juros. Se a dívida crescer muito, a organização pode gastar mais para mantê-la do que para agregar valor. McConnell compara isso a uma base de código legada onde muito esforço é gasto para manter o sistema funcionando, deixando pouco tempo para novas funcionalidades.

Para tornar a dívida técnica mais transparente, o autor sugere duas abordagens:

1. **Manter uma lista de dívidas no sistema de rastreamento de defeitos:** Cada dívida é registrada com as tarefas necessárias para pagá-la, esforço estimado e cronograma. Dívidas não resolvidas com mais de 90 dias são tratadas como críticas.
2. **Manter a lista de dívidas como parte do backlog de produto Scrum:** Cada dívida é tratada como uma "história" do Scrum, com esforço e cronograma estimados da mesma forma que outras histórias.

Ambas as abordagens aumentam a transparência e servem como salvaguarda contra a acumulação de "dívida de cartão de crédito" (atalhos não focados), pois forçam a equipe a registrar e planejar o pagamento de qualquer atalho significativo.

Pagamento da Dívida

McConnell argumenta que "trabalhar para quitar a dívida" pode ser motivador. Uma boa abordagem para dívidas de curto prazo é dedicar a primeira iteração de desenvolvimento após um lançamento ao pagamento dessa dívida. A capacidade de pagar a dívida também depende do tipo de software; dívidas em firmware de aviônicos, por exemplo, exigem um padrão muito mais alto para serem contraídas devido ao alto custo de correção.

Comunicação sobre Dívida Técnica

Um dos maiores desafios é comunicar a dívida técnica a stakeholders não técnicos. McConnell sugere usar uma linguagem financeira para tornar o conceito mais compreensível:

- **Usar o orçamento de manutenção como proxy:** Diferenciar a manutenção que mantém o sistema funcionando (dívida técnica) da manutenção que estende as capacidades.
- **Discutir a dívida em termos monetários:** Por exemplo, "40% do nosso orçamento de P&D atual está indo para o suporte de lançamentos anteriores" ou "Estamos gastando $2.3 milhões por ano para servir nossa dívida técnica" [1].

Tomada de Decisão sobre Dívida Técnica

Ao considerar a dívida técnica, as equipes geralmente veem duas opções: o caminho "bom, mas caro" e o caminho "rápido e sujo". McConnell argumenta que é crucial considerar uma terceira opção: um caminho "rápido, mas não sujo" que minimize os juros da dívida. Ele apresenta um exemplo detalhado com custos e prazos para ilustrar como a terceira opção pode ser a mais vantajosa a longo prazo, mesmo que tenha um custo inicial um pouco maior que a opção "rápida e suja".

Conclusão

O white paper de Steve McConnell oferece uma estrutura clara e prática para entender e gerenciar a dívida técnica. Ele enfatiza a importância da tomada de decisões explícitas ao contrair dívidas, o rastreamento transparente e a comunicação eficaz com stakeholders. Ao tratar a dívida técnica como uma dívida financeira, as organizações podem tomar decisões mais informadas que equilibram as necessidades de curto prazo com a sustentabilidade de longo prazo do software.

Referências

[1] McConnell, S. (2008). *Managing Technical Debt*. Construx Software. Disponível em: www.construx.com/whitepapers