# Engenharia de Software – Essencial para próximas décadas

**Antonio Mendes da Silva Filho***

*"Nothing in life is to be feared,*
*it is only to be understood.*
*Now is the time to understand more, so that we may fear less."*
*Marie Curie*

Engenharia compreende o uso de princípios e conhecimento empírico e científico obtidos a partir de estudos, investigações e experiências visando o desenvolvimento de um produto, que dependendo da área pode ser um motor, um dispositivo eletrônico, um prédio ou um sistema de software. Para tanto, três aspectos essenciais devem ser considerados: custo (orçamento de desenvolvimento), tempo (cronograma de execução) e qualidade. Dentro deste contexto, a engenharia de software (a exemplo de outras engenharia) também visa o desenvolvimento de um produto (software) que pressupõe satisfazer aos requisitos de qualidade, dentro do prazo e sem estourar o orçamento. Nesse sentido, este artigo destaca que o profissional de engenharia de software é crucial para momento atual e também será durante as próximas décadas, esteja ele atuando como analista, desenvolvedor ou gerente de equipe de projeto [1], [2], [3], [4] e [5].[1]

## Software no cotidiano

Você já percebeu que software está praticamente em todas as coisas de seu cotidiano? Um exemplo simples é a central telefônica que permite duas pessoas conversem ao

* Professor e consultor em área de tecnologia da informação e comunicação com mais de 20 anos de experiência profissional e Doutor em Ciência da Computação pela Univesidade Federal de Pernambuco.

[1] [1] Liderança, compromisso, confiança e plano de projeto: ingredientes essenciais à gestão de projetos, disponível http://periodicos.uem.br/ojs/index.php/EspacoAcademico/article/viewFile/13268/6972
[2] Intelecto Humano: Liderança Requer Compromisso e Compleição, disponível em http://www.periodicos.uem.br/ojs/index.php/EspacoAcademico/article/view/13040/685 9
[3] Gestão de Projetos: Estratégia Essencial às Corporações, disponível em http://www.espacoacademico.com.br/066/66amsf.htm
[4] Componente de Software: componentização no desenvolvimento de software, disponível em http://www.espacoacademico.com.br/087/87amsf.htm
[5] Arquitetura de Software – sobre a importância do reuso, disponível em http://www.espacoacademico.com.br/068/68amsf.htm

telefone. O controle da operação das centrais telefônicas é todo feito por software. Você já foi a alguma casa lotérica para efetuar um pagamento de conta de água ou energia? Ou já arriscou jogar na loteria? Quando você vai à casa lotérica por qualquer um dos motivos acima, você está usando o sistema que tem todo seu controle feito por software e o mesmo acontece quando você vai ao banco. Perceba que quase todos os sistemas hoje em dia têm seu controle operacional sendo feito por software. E com certeza você é usuário de computador que possui diversos tipos de software operando nele. Observe que o software tem se tornado em um *companheiro* e sido uma ferramenta fundamental de nosso dia-a-dia.

Se você 'olhar' para trás, poderá perceber que há, aproximadamente, cinco décadas atrás, software constituía uma pequena senão ínfima parcela dos sistemas computacionais quando comparado ao hardware. Naquela época, os custos de desenvolvimento e manutenção de software eram desprezíveis. Hoje, porém, software é responsável por significativa porção dos sistemas computacionais. Encontramos software nas mais diversas aplicações. No uso doméstico, fazemos uso de processadores de texto (como, por exemplo, Word da Microsoft). Adicionalmente, software tem sido um componente importante e muito utilizado em diversos sistemas. Podemos exemplificar seu uso no controle e supervisão dos sistemas de geração e distribuição de energia bem como em sistemas de telecomunicações, onde ele é encarregado do controle e roteamento de milhares de ligações telefônicas. Observe que empresas e pessoas têm conseguido otimizar suas atividades, geralmente, fazendo uso de software. Mas, o que é software?

Software compreende um conjunto de instruções que quando são executadas em um processador fornecem funcionalidades com desempenho desejado. Software é também entendido como programa de computador, o qual é composto de instruções que fazem o computador prover as funcionalidades desejadas. Entretanto, esse conjunto de instruções não precisa apenas ser executadas num computador. Com a redução de custo dos processadores, podemos encontrar software em outros dispositivos como, por exemplo, o telefone celular.

Agora, se você quiser uma visão mais completa, observe que software compreende não apenas o conjunto de instruções que são executadas num processador para fornecer as funcionalidades de um sistema, mas também todo artefato vinculado ao código como a documentação do sistema (que traz informações do projeto), bem como documentação para usuário (informando-lhe como utilizar o software).

Nesse sentido, um engenheiro de software é o profissional responsável por levantar e analisar o conjunto de requisitos, desenvolver o projeto, implementá-lo, testá-lo e entregar ao cliente. Mas, para ter sucesso, na execução do projeto, o engenheiro de software precisa considerar três aspectos essenciais às engenharias, que costumo denominar dos pilares da engenharia: Custo, Tempo e Qualidade. Esses pilares, juntamente com a compreensibilidade e princípios de engenharia constituem as preocupações do engenheiro de software desde o primeiro dia de qualquer projeto.

Um engenheiro de software faz uso de técnicas de modelagem e precisa se preocupar com a estrutura do sistema de software. A estrutura do software deve ser intrinsecamente mais fácil de compreender, além de ser bem documentada, permitindo ser compreendida em nível global ou em detalhes. Pensar e projetar um software dessa forma é torná-lo mais fácil de ser modificado e, portanto, quando qualquer de suas

características é mudada, ele continua funcionando. Isso significa considerar a necessidade inequívoca de manutenção.

Perceba que seu papel como engenheiro de software vai muito além de apenas conceber o projeto. Ele precisa também antecipar possíveis modificações. Dentro do contexto discutido acima, é importante observar que a manutenção é uma das fases que certamente o software irá lidar. Outras duas fases são definição e desenvolvimento como ilustrado na Figura 1.

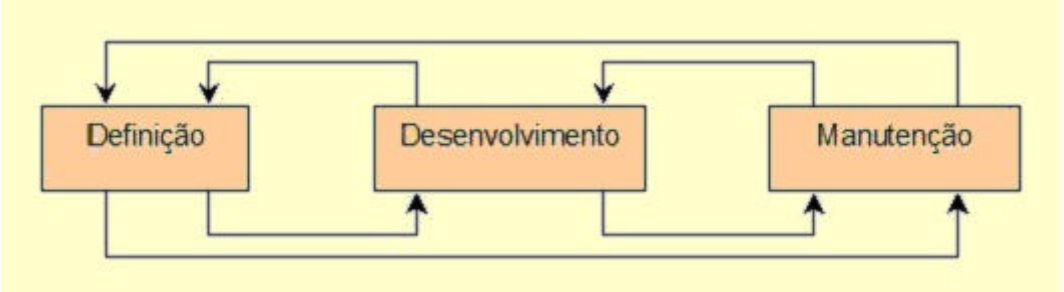


Figura 1 – Fases genéricas no desenvolvimento de software.

Cabe destacar que o desenvolvimento de qualquer produto (como, por exemplo, software) ou artefato requer (do engenheiro de software) saber quais os passos necessários para alcançar o objetivo de ter o produto pronto (desenvolvido). Isso compreende as fases de definição, desenvolvimento e manutenção, como mostradas na Figura 1.

A fase de ***definição*** engloba a identificação de informações que deveriam ser processadas, funções e desempenho desejados, tipo de interface a ser utilizada, tarefas que o sistema deveria prover suporte, perfil de usuários do sistema, dentre outras.

A fase de ***desenvolvimento*** concentra-se no projeto de estruturas de dados e arquitetura de software do sistema (isto é, como ele está organizado), conversão do projeto para uma linguagem de programação específica (ou seja, implementação), realização de testes e avaliação.

Finalmente, a ***manutenção*** considera modificações e/ou correções necessárias no sistema a fim de que este atenda aos requisitos do sistema. Perceba que o processo de desenvolvimento de um sistema de software tem duas grandes atividades de interesse que envolve o desenvolvimento da porção de software que implementa as funcionalidades do sistema e a atividade que a antecede e norteia o desenvolvimento que é o projeto de software. Esta última atividade é resultado do levantamento e análise de requisitos que provê informações para decisões de projeto.

A *análise* visa investigar e resolver um problema. O objetivo da análise é levar o analista ou projetista a investigar e a descobrir como tratar o problema que ele tem em mãos. Perceba que quando você finalizar a análise, você terá uma compreensão completa do problema (comumente chamado de 'descrição do problema'), e, portanto, o projeto será a sua solução.

Note que o profissional de engenharia de software é crucial para momento atual e também o será durante as próximas décadas. Por que?

Há atualmente uma enorme necessidade de formar mais capital humano para esta área. De acordo com pesquisa realizada pela Forrester Research, Inc., os gastos no setor de TI a nível mundial no ano de 2009 totalizaram US$ 1,469 trilhões, sendo US$ 362 bilhões referentes apenas a software, havendo previsão para os anos de 2010 e 2011 de que os gastos com software serão de aproximadamente US$ 400 e 440 bilhões, respectivamente, como mostrado na Figura 2. Desse total, quase metade do total do mercado de software é pertinente aos EUA, enquanto que cerca de 10% do total é pertinente à América Latina.

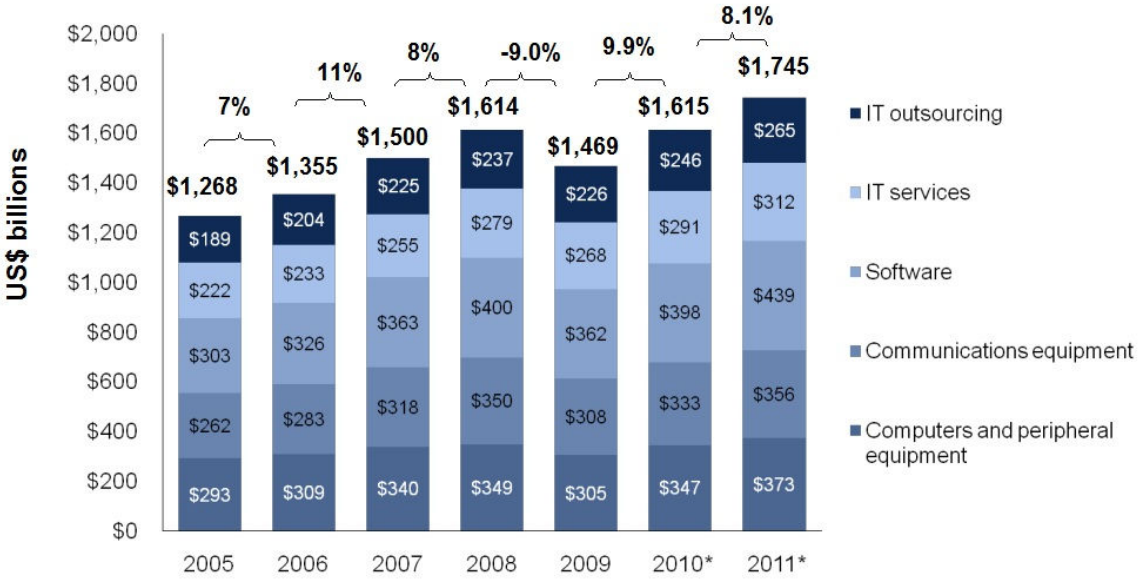


Figura 4 – Gastos em TI a nível mundial.

(Fonte: *Global IT Market Report*, Forrester Research, Inc., 2010)

**Necessidade da Engenharia de Software**

No contexto atual, qual a real necessidade da engenharia de software?

Se uma análise criteriosa for feita, levando em consideração o uso e dependência em larga escala de software, a necessidade de prover confiabilidade (operacional) e segurança, pode-se afirmar que é crítica. E, mais ainda, há consequente demanda por profissionais da área qualificados.

A adoção das práticas de engenharia de software é essencial para assegurar a confiabilidade dos produtos e serviços (de software). As práticas compreendem instituir uma ***'cultura' de engenharia de software***, onde a equipe de projeto de sistemas de software esteja comprometida com a documentação de todos os artefatos de um projeto de modo a prover suporte a:

- Rastreabilidade
- Manutenibilidade

- Reuso.

A fase inicial do desenvolvimento de software na qual ocorre a definição do sistema considera o entendimento dos requisitos desejados além de claramente estabelecer a origem do requisito. Essa informação será fundamental para a rastreabilidade.

Observe que ***rastreabilidade*** está relacionada aos relacionamentos entre os requisitos, as fontes (dos requisitos), bem como o projeto de sistema. A rastreabilidade das fontes dos requisitos visa identificar e relacionar os requisitos às partes interessadas (ou stakeholders) daqueles requisitos. Além disso, há o relacionamento entre os requisitos (isto é, a dependência existente entre eles) e a ligação de requisitos aos componentes do sistema em desenvolvimento.

E, quanto a ***manutenibilidade***?

Neste momento, é importante distinguir entre manutenção que compreende um conjunto de atividades planejadas a fim de realizar mudanças, enquanto que a evolução se refere aquilo que de fato acontece com o software. E, nesse sentido, cabe destacar à medida que software tem se tornado quase ubíquo nos sistemas, ele tem também crescido em tamanho (isto é, na quantidade de linhas de código) e complexidade. Perceba que esse crescimento dos sistemas de software requer mecanismos apropriados para documentação, rastreamento, e manutenção.

Por outro lado, o ***reuso*** de software pode ser entendido como um processo de implementar e atualizar sistemas de software fazendo uso de software existente. Vale lembrar que software compreende componentes, objetos, requisitos, modelos de projeto, arquitetura, documentação do código, cenários de testes, metas de projeto, manuais. Vale ressaltar que reuso de software pode se dar num sistema de software, num conjunto de sistemas similares ou ainda em sistemas completamente diferentes. Note que reuso de software não é desenvolver um sistema de software desde seu início, mas desenvolver um sistema de software a partir dos componentes de software já existentes. Cabe destacar que a meta de reuso de software é utilizar tanto quanto possível o software resultante de esforços de desenvolvimentos anteriores, visando reduzir os custos, tempo e riscos associados com novos desenvolvimentos.**Cultura da Engenharia de Software**

Instituir uma ‘cultura’ de Engenharia de Software significa adotar suas práticas e obter da equipe de projeto um comprometimento de documentar adequadamente todos os artefatos de um projeto. Talvez, você possa até estar imaginando que dedicar esforço em documentar todos os artefatos produzidos seja um esforço fútil e desnecessário. E, concordo com você se estiver considerando desenvolver um pequeno (sistema de) software para uma pequena farmácia ou mesmo para aquele ‘mercadinho da esquina’. Tal tarefa pode ser comparada ao esforço de você construir uma casinha de madeira para seu cachorro de estimação. Note que em tal situação, você, sozinho, pode dar conta do recado.


No entanto, se você tiver a necessidade de informatizar um sistema de uma biblioteca de uma instituição que possui cerca de 10.000 usuários (havendo renovação de quase 2000 usuários por ano) e que tem mais de 50.000 títulos entre livros, revistas e outros itens (com aquisição regular de novos títulos), então você terá a necessidade de trabalhar em equipe a fim de desenvolver esse software. Aqui, torna-se prudente documentar o projeto. Também, aqui, documentação não é um 'luxo', mas sim uma necessidade, pois tal sistema com certeza terá modificações.

Agora, pode-se considerar uma situação ainda mais extrema de documentação de projeto. Você tem noção de quantas linhas de código há num avião Boeing 777?

Um Boeing 777 possui mais de 4 milhões de linhas de código (isto é, software) rodando em cerca de 1.300 processadores. Agora, você imagina desenvolver tudo isso sem qualquer planejamento ou (documentação de) projeto?

Desenvolver um sistema como de uma biblioteca (ou similar) requer atividades de modelagem para apoiar o projeto, um processo de desenvolvimento com atividades (bem) definidas, além de ferramentas que automatizem atividades do processo. E, isso será ainda mais essencial no desenvolvimento de sistemas de grande porte como, por exemplo, no software utilizado em aviões, em sistemas de comércio eletrônico ou sistema de administradoras de cartão de crédito.

Perceba que desenvolvimento de software tem sido e ainda permanece uma atividade difícil, pois ela requer que o engenheiro de software considere os elementos influenciadores sobre o software como, por exemplo, custo, prazo, confiabilidade, manutenibilidade, inovação tecnológica, escalabilidade e desempenho.

Dentre essas 'forças' influenciadoras sobre o software, três delas têm impacto determinante sobre o software que compreendem: custo, prazo e qualidade

(caracterizada, principalmente, pela confiabilidade que constitui atributo essencial da qualidade).

Afinal, ***qual o problema com o software?***

Falibilidade de software (ou possibilidade de existência de falhas) e a consequente falta de confiabilidade, além do fato de que ***software é quase sempre modificado***, resultando num produto quase sem garantia. Observe que a única certeza que se tem é a de que o software será modificado e, portanto, a documentação do projeto é essencial para permitir a rastreabilidade e, mais importante, a manutenibilidade, além de reuso (do projeto).

Para lidar com essa demanda, a engenharia de software provê um conjunto de técnicas que visam a construção de sistemas de software que atendam os requisitos do sistema. Note que a proposta da engenharia de software compreende estabelecer e utilizar princípios de engenharia a fim de obter software de baixo custo que seja confiável e opere de maneira eficiente nas máquinas onde for empregado. Mas, então, há um aspecto que diferencia software de outros produtos. Geladeiras, TVs e outros equipamentos são fabricados (leia-se 'montados'). Software, diferentemente, é construído (por seres humanos). Todavia, isso será explorado num outro artigo.