Documentações e o “tempo perdido que não volta mais”

11/06/2012 10:25

Tempo. Na área de TI, quase tudo se resume ao tempo que você, desenvolvedor, vai gastar para fazer X atividades. Existem milhares de técnicas e maneiras de mensurar quanto tempo você vai gastar para fazer uma determinada tarefa, e todas elas são de extrema importância para o mercado. Afinal, como já dizia nosso grande *Super Sam*, interpretado pelo grande *Ramón Valdés, “***time is money**”

Oh Yeah!

Porém, uma das coisas que eu acho mais engraçadas na nossa área, é o tempo que é perdido com documentações falhas e que **nunca** vão ser usadas pra nada além de fazer volume no “pacotão da homologação”. As metodologias dizem que tais documentações são importantes, mas a prática nos mostra o contrário. Então eu pergunto: **o que diabos acontecem com as documentações técnicas em projetos SAP?**

Se você já pensou que eu vou escrever aqui que documentações são descartáveis, **errou feio.** Eu acredito que elas tem muita importância em qualquer projeto de TI. O problema é que o conceito de documentação em vários projetos e empresas que usam o SAP está completamente equivocado. Horas e horas são gastas em documentos que não dizem absolutamente nada sobre o software. Milhares de palavras gastas para descrever um software linha a linha, e nem um mísero parágrafo para explicar o que determinado FORM ou Método fazem dentro do processo da execução. São **desenvolvedores brincando de documentar**, e **aprovadores** se divertindo com a existência do documento e **ignorando completamente seu conteúdo**.

Mas vamos lá: para que serve uma **EF** (Especificação Funcional) e um **ET** (Especificação Técnica)? Sei que boa parte dos que lêem estes site já devem estar carecas de saber, mas lembrem-se que o ABAPZombie abriga muito novatos na área 🙂 Se você já sabe, tenha em mente que vou falar de ETs no resto do post, e pode pular o momento noob aí embaixo.

[começando o momento abap noob]

A Especificação Funcional (**EF**) é um documento criado por Funcionais, que explica como um determinado software vai ter que se comportar no sistema para atender à uma necessidade de um(a) usuário/empresa. Ele contém principalmente explicações de como o software deve ser portar do ponto de vista do processo. Pode ou não ter explicações um pouquinho mais técnicas (nome de tabelas, campos, nome de funções) – isso depende do funcional que as escreve. Podemos dizer, em outras palavras, que ela deve descrever o que o usuário quer, de uma forma que o programador consiga entender. Ou, pelo menos, deveria 😛

A Especificação Técnica (**ET)** é um documento criado pelos programadores, que precisa explicar tecnicamente como o software funciona dentro do sistema. É suportado pela EF, e contém descrições de como o programador criou o software para atender o que a EF pediu. Normalmente contém a lista dos objetos criados, uma explicação de como cada objeto foi usado no desenvolvimento e uma descrição de como cada parte do programa foi tecnicamente implementada. Ou, pelo menos, deveria-ao-quadrado 😛

[fim do momento abap noob]

Para exemplificar o problema das ETs nos projetos, eu tentei separar alguns padrões que encontrei nos projetos em que atuei como desenvolvedor. Será que você já se deparou com algum deles?

———————————————–

ETs de 800 páginas para programas de 500 linhas

Muitas empresas tem templates fechados para a criação de especificações técnicas e a grande maioria pede detalhes DEMAIS sobre o programa. Descrições de tabelas campo-a-campo (colocando o tipo, tamanho, descrição, e outras X bilhões de informações sobre cada mísero campo), listagem de todos os TEXTs criados no programa com as suas respectivas traduções, atributos técnios de cada um dos objetos do desenvolvimento, e mais outras informações que simplesmente não precisariam estar ali.

Esse é o tipo de ET que tem o menor tempo de vida: é só você mudar a descrição de **UM** campo, em qualquer tabela que você criou, e ela já está desatualizada. É também aquela que não explica quase **nada** sobre como o software funciona, ou seja: se o programador que criou o software sair da empresa, o outro que debugue o programa até a morte para descobrir como ele funciona.

Criar a ET antes de codificar

Se um mesmo programador vai criar o código e a ET, é muito comum ele primeiro codificar e só depois documentar – mesmo que o mundo corporativo diga que isto está errado. Porém, existem casos onde um programador mais experiente cria a ET, e um outro menos experiente codifica com base no documento. Esse tipo de ET tende a ficar mais bem criada, porque ela explica tecnicamente como o programa deve funcionar, mas não chega num nível de detalhe absurdo e irrelevante para o documento.

O problema é que, na grande maioria das vezes, o programador júnior acaba fazendo alguma coisa completamente diferente do que estava na ET, simplesmente porque no mundo da programação você só sabe que certas coisas não vão funcionar quando você, de fato, vai lá e tenta codificar. Ok, essas coisas acontecem... mas nossa classe é preguiçosa e ninguém atualiza a ET. Ela já nasce desatualizada, e não descreve totalmente como o software criado funciona.

Fazer da ET uma cópia sem vergonha da SE38

E eu já cansei de ver gente que, no meio da correria do projeto, abre um documento novo no word, dá CTRL+C no código fonte e um CTRL+V na ET, salva e entrega o programa. E isso é mais comum do que parece, acredite. E estas ETs passam por todos os níveis de aprovação, acredite de novo.

Criar a DOMM – Documentação Orientada a Mim Mesmo

Muita gente esquece, mas esse monte de coisas que você escreve nas ETs servem como referência para o futuro. Num mundo perfeito, se você chegasse em um projeto para fazer um modificação funcional em um programa de

829382 linhas, você não teria que debugar tudo, primeiro iria ler a EF para saber que processo o programa cobre, e depois iria ler a ET para entender como aquele processo funcional foi implementado pelo programa.

O problema é quando o cara utiliza a ET como um rascunho, um documento onde ele posso escrever todas aquelas coisas maravilhosas que ele precisa lembrar sobre o desenvolvimento. O mais legal é que ele costuma deixar tudo descrito da maneira mais desorganizada possível, o que ajuda muito a vida de quem for ler aquilo no futuro #NOT.

———————————————

O tempo destinado à criação de documentações precisa passar a ser usado com sabedoria. Não é por acaso que o **manifesto ágil** valoriza *Software em Funcionamento mais que documentação abrangente*. A triste verdade é que boa parte das ETs em projetos SAP não servem pra nada, e só existem porque a metodologia dos projetos obriga o desenvolvedor a criar o documento para que o produto possa entrar em processo de homologação.

Eu gosto da idéia de ter um documento que explique o software, mas odeio a maneira como eles normalmente são gerados. “*Mas Mauricio, então como você acha que podemos gerar documentações que sejam mais úteis?*“, você me pergunta, jovem (ou velho) padawan. E eu explico: temos que ser realistas quando tratamos de melhorias em documentações. Não dá para jogar tudo para o alto e dizer que ETs não prestam, que agile é legal, e que vamos criar nossas próprias documentações, do jeito que a gente quiser… isso simplesmente não vai acontecer em projetos SAP, nem a curto e nem a médio prazo. Temos que melhorar nossas documentações dentro do cenário atual.

Portanto, segue aqui a minha lista de dicas de como criar uma ET que traga valor para o seu desenvolvimento, e que vão fazer o seu tempo ter sido gasto com alguma coisa útil. E sim, eu gosto de listas – abraços à todos aqueles que gostam de listas 😀

1 – Pense no que você gostaria de ler em uma ET alheia

A primeira coisa que eu costumo fazer ao documentar é pensar nas coisas que um outro programador possa achar interessante se ele fosse ler a minha ET. Quais pontos precisam de uma explicação mais detalhada? Porque eu tive que fazer um lógica muito mais enrolada num ponto que, ao se basear na EF, pareceria bem mais simples? Porque optamos pelo paralelismo? Tem algum ponto que é mais fácil dele entender se olhar diretamente o código? Porque as classes foram criadas dessa maneira? Entre outros. Se você responder algumas dessas perguntas, sua ET já vai valer MUITO MAIS do que a maioria das ETs criadas por aí.

2 – Pare de encher linguiça: Guarde particularidades do código para o CÓDIGO

Já vi várias ETs que contém descrições demais sobre partes que só fazem sentido para pontos muitos específicos do código. Aqui entram criações de variáveis auxiliares, loops binários, utilização de rotinas de conversão, métodos e funções para conversões de valores… A não ser que esses pequenos trechos de código façam sentido para o entendimento da solução implementada, pra mim, não vale a pena descrevê-los na ET.

3 – Lembre-se: “nada de inibir a criatividade do programador”

Esse frase foi dita por um grande parceiro meu de projetos, há mais de 5 anos atrás e ela continua válida até hoje. Se você está escrevendo uma ET para um colega menos experiente, cuidado para não cair no mesmo problema anterior: descrever demais o que deve ser criado. Se fosse para você ter que resolver TUDO na ET, seria mais fácil ir lá e escrever o código de uma vez, certo? 😉

Pessoas mais novas precisam de estímulos para desenvolver sua capacidade de programar, portanto, nada de inibir a criatividade dos seus pupilos. Ah, e peça para ele atualizar depois, deixando a ET fácil de ser entendida por algum outro programador no futuro.

4 – Copiar e colar códigos inteiros na ET… just DON’T

Essa atividade é feita levando em conta a “perda de tempo em criar documentações”. Sinceramente, neste caso, até esses míseros 10 minutos que você vai perder dando CTRL+C CTRL+V no código serão perdidos. Não faça isso, em hipótese nenhuma.

Quando eu leio uma ET assim, para mim está claro que o programador não faz a mínima questão de explicar como o seu desenvolvimento funciona para ninguém (talvez porque nem ele mesmo sabe o que ele está fazendo 😛). Criar documentações **faz parte SIM do nosso papel como desenvolvedores de projetos SAP**, e, se você não concorda, mude de profissão.

5 – Faça sugestões de mudança nos templates

Se você não trabalha para um projeto que tenha um template 100% fechado, pode ser de grande valia sugerir mudanças para deixar a ET mais enxuta e de fácil entendimento para outros programadores (ou quem sabe até pra você mesmo no futuro).

Em projetos que utilizam metodologias ágeis, a documentação costuma ser um pouco mais livre, e cada equipe adota somente documentações que eles julguem extremamente necessárias. Se você conhece alguém que trabalha num empresa que adota esse tipo de metodologia (mesmo em outras linguagens), pergunte para ele o que eles adotaram e o que faz sentido para aquela empresa. Você pode trazer algumas idéias para sua equipe, e fazer a ET, enfim, prestar para alguma coisa.

6 – Tente começar a ler ETs de vez em quando

Nós programadores temos o hábito de sair olhando para o código antes de ler qualquer tipo de documentação. É como quando você compra uma TV nova e nem dá bola para o manual: já sai fuçando e aprendendo tudo sozinho. Para pequenas correções e modificações faz sentido ir direto no código e alterar, porém, para modificações maiores e erros malucos, pode ser que o seu colega programador tenha sido solidário e tenha escrito uma ET bacana, daquelas que vão lhe ajudar a entender como o programa foi implementado. Ler essa ET pode fazer com que você gaste muito menos tempo na sua atividade.

Mas confesso: infelizmente, a grande maioria das ETs por aí ainda são descartáveis... e falo isso com certa propriedade, afinal, como você acha que eu consegui mapear aqueles padrões ali de cima? Lendo ETs pelos projetos afora! Mas ler é legal, porque você vê cada coisa bizarra... e algumas coisas bem legais.

Ah, e é aqui que você que escreve ETs zuadas fica com vergonha ao saber que alguém um dia deve ter lido e xingando seu nome ali em cima do documento. 😈

Quem lê os meus posts aqui no site já sabe: **agora é a sua vez**. Quais são seus comentários sobre esse mundo maravilhoso de ETs? Quais tipos de coisas você já encontrou? Você se encaixa em algum daqueles padrões? Identifica mais algum? Tem mais alguma dica?

Post gigante, mas eu precisava falar desse tópico. Estava pensando nele há bastante tempo, e fico feliz por, finalmente, conseguir criá-lo! 🙂 Um grande abraço à todos aqueles que se preocupam em deixar algum valor em suas documentações. A atividade é odiada, mas é necessária e pode sim fazer sentido – mas só quando o tempo é utilizado de forma correta. Espero que este post possa lhe ajudar a re-pensar este assunto! Até o próximo!

ET perfeita, eu quero acreditar!
*insira a música do X-Files aqui - PANANANANANANAN TU RU RU RU TU RUUUUUU *

Comentários

Valdeir Gomes — *20/07/2020 18:39*

Caramba Mauricio, vejo que preciso melhorar minhas ET :'(me encaixei em 3 de 6.
Obrigado pelo Post com certeza irá ajudar nas próximas.

Sucesso.

Andre Manchester — *14/12/2018 09:02*

Algum template disponível de especificação funcional SAP ? Obrigado.

Gabriela Marques de Sá — *29/12/2014 09:26*

Adorei esse post e é raríssimo uma EF ou ET que tem sido bem escrita cujo conteúdo seja realmente útil. Rolei de rir com este trecho: ‘...e o programador que criou o software sair da empresa, o outro que debugue o programa até a morte para descobrir como ele funciona.” pois acabei de passar por isso recentemente.
Sou ABAP desde 1999 e workflow desde 2001.
Adoro este site. Parabéns!

Vanessa Morais — *17/12/2013 10:08*

Maurício,

Não publique meu comentário na pagina por favor!!

Era só um desabafo rs

Vanessa Morais — *17/12/2013 10:07*

Maurício,

Li o post e com certeza irei melhorar minhas ET’s, mas um dos motivos delas não serem muito boas é que quando estou desenvolvendo faço muitas perguntas aos Seniors sobre o que deve ser feito, daí quando termino o DEV, vou começar a entender por que fiz tal coisa e depois outra coisa, enfim, nem eu entendo direito rs então a ET fica do jeito que eu entendi...

hey joo — *07/06/2013 22:28*

Legal o post. Aborda formas erradas de fazer uma coisa que tem que ter.
Um ponto bom da ET ser chata e grande é que na verdade programas não deveriam ser criados. Já tá pagando uma fortuna pelo software e ainda vai desenvolver ... xD
O que as empresas não podem esquecer de cobrar também são os comentários dentro do código.

Fábio Mazetto — *15/07/2012 20:13*

Muito bom este post Maurício. Trabalho como consultor em parceiros Mastersaf e as interfaces já são um espetáculo à parte, as ET então nem se fala. E pra piorar nosso caso, normalmente a alocação é tão curta que nem há tempo hábil para escrever alguma coisa. Estou justamente estudando possibilidades de documentação destes ajustes pois, em muitos casos (grande maioria) o projeto SAP já acabou e/ou a implantação do Mastersaf foi há muito tempo, ou seja, histórico nem pensar!

Abraço

Renan William Alves Oliveira — *22/06/2012 11:38*

hahaha pensei que só eu era revoltado com as ETs alheias. Eu vivo um cenário ainda pior: além das maiorias das ETs serem do pior padrão, elas são literalmente descartáveis!! O inteligente que fez o desenho do repositório da documentação no SOLMAN não amarrou as ETs com os objetos ABAP e sim com o ID da melhoria.
Ou seja: quando vem aquela modificação em um programa de 101039128182 linhas e você PRECISA de pelo menos algum tipo de documento de como aquela coisa funciona, é impossível de se encontrar A MENOS QUE você saiba os ID das melhorias anteriores naquele objeto. É literalmente impossível, é um time de +/- 100 consultores, em um projeto de mais de 5 anos. Imagina minha felicidade em ainda continuar fazendo a santa ET, sabendo que nunca mais ela será usada.

> **Danilo Cardoso** — *22/06/2012 17:45*
>
> Trágico

Danilo Cardoso — *21/06/2012 15:26*

ET? Só se for um ser de outro de planeta mesmo, pois, Espec Técnicas são em grande maioria, sem noção, não-objetivas, enfim, não servem pra nada (eu disse a grande maioria).

Mas esse post me fez pensar melhor no jeito de documentar. A carapuça serviu direitinho em mim em algumas situações citadas acima e creio que vai servir em todos os desenvolvedores, afinal, que nunca fez uma ET de merda que atire o primeiro mouse .. rsrs .. 🙂

Um abraço a todos aqueles que (convivi muito tempo com o Mauricio ..rs), por mais que tenham feito uma ET zuada um dia, após esse post, pensarão em melhorar cada vez mais.

Peace...

> **Mauricio Cruz** — *21/06/2012 15:44*
>
> Graaaaaaaaaaaaaaaaaaaaaaaaaande Danilão!
>
> Obrigado pelo comentário! E eu não posso atacar mouse em ninguém não, pq eu já fiz ETs zuadas haha. Mas o post é para isso mesmo, refletir um pouco sobre a qualidade das documentações.
>
> Abraços a todos aqueles que curtiram esse post. Quem não curtiu, não ganha abraço. FIM. (e sim, eu ainda tenho esse costume hahahaha :D).

Edu — *14/06/2012 15:04*

Agradeçam a uma saudosa empresa auditoria, por que graças ao belo trabalho que fizeram na Enron, os Estados Unidos ficou com medo de evasão de investidores no país. Aí um senador e um deputado americano (claro pensa que só político daqui inventa moda?) inventaram uma série de mecanismos de auditoria e segurança e chamaram de Lei Sarbanes-Oxley, que é na verdade um monte de documento que gente como eu e voce cria e ninguem lê, com o formato que eles querem e nem sabem porque.

> **Mauricio Cruz** — *14/06/2012 15:16*
>
> Foi só eu que senti uma leve alfinetada à uma consultoria atual, que, por sinal, baseia seu processo em quilos e litros de documentos sem sentido?
>
> 😀

Henrique Menon — *13/06/2012 13:28*

Já participei de projetos que na TD citava a linha e colocava o codigo direto e sem explicação.
Isso era normal!
E pior, quando acontecia o Realese as documentações se perdiam.

Como pode????

Daniel Jesus — *12/06/2012 14:18*

quem me dera 10% dos programas e exits tivessem uma documentação decente.

Junior — *11/06/2012 15:39*

> Opa, muito bom. Quando fiz minha primeira ET ela ficou muito zuada, mas foi porque me passaram assim na empresa. Lembro que tive esse pensamento “Que cagada é essa, não foi isso que aprendi na faculdade” kkk tenso demais zombies!