Sistemas CGTI - SAL

1. SISTEMAS EXISTENTES/ EM DESENVOLVIMENTO

Sistema	**Formalização da declaração de escopo preliminar** (início da construção do sistema)	**Situação atual**
Controle Agenda SAL (NP0310)	**16/04/2010**	O projeto foi iniciado em 16/04/2010. O termo de abertura só foi assinado entre 8 e 14/06/2010, dois meses após a iniciação do projeto. O documento de visão e requisitos foi produzido entre 12/05 e 23/09 (4 meses!). O sistema encontra-se em fase de Desenvolvimento (codificação)
SISNORMA	Sistema já desenvolvido /contratado desde 2003, porém fora do ar (sem visualização de imagens) desde o início de 2009. Quando funcionava, apresentava dados apenas dos anos de 2002 e 2003.	As últimas reuniões foram em 01/ 12 e 17/12 e foi relatado pela CGTI que as imagens estavam íntegras e que o banco de dados não estava funcionando. Devido a grande dificuldade para reestabelecer o Content Manager, sugerimos a CGTI em reunião de 04/Fev que buscasse a relação entre os metadados das imagens presentes no sisnorma e os arquivos do filesystem do content manager. Não há previsão para reestabelecimento do serviço (Content Manager)
SISNORMA 2.0 (NP0260)	**23/09/2009** Ressaltamos que esta demanda foi formalizada pela SAL em 26/05/ 2009, por meio do memorando nº 169/ 2009.	Como o sistema seria a atualização do SISNORMA, a demanda foi suspensa até a estabilização do SISNORMA, requisito para identificação e definição dos próximos passos.

SG-SAL (NP0227)	**26/08/2009**	O projeto foi iniciado no final de agosto de 2009 e a primeira reunião foi realizada apenas em janeiro de 2010. Dessa forma, a Declaração de Escopo Preliminar só foi produzida em 03 de Fevereiro. O documento de visão e requisitos foi iniciado em 20/01/2010, com a última alteração datada de 28/06 e sua aprovação ocorrendo somente no início de agosto de 2010. Com isso, sua elaboração durou 5 meses e o período entre a última alteração e a assinatura foi de mais 1 mês. Certamente grande parte dessa demora se deve a dificuldade de elaboração dos próprios documentos. O projeto ainda encontra-se em fase de elaboração (levantamento dos requisitos e elaboração dos documentos).
AJURI (NP0208)	**17/08/2009** Ressaltamos que esta demanda foi formalizada em 30/ 05/2008, por meio do e-mail do Dr. Daniel Arbix (SAL) para o Dr. Jorilson.	O projeto foi formalizado em 17/08 (via Termo de Abertura, embora a demanda havia sido formalizada mais de 2 meses antes) e foi assinado em 02/09 (TA). Diante da demora no desenvolvimento dos demais sistemas, mais simples, optou-se pela suspensão do desenvolvimento deste, mais complexo.
ISEGORIA (NP0259)	**23/09/2009**	Foram solicitadas modificações e correções no sistema. Houve reunião em **30/07/2010** com Sr. Luis Miguel Carneiro da DBA, e no dia **22/09/2010** foi entregue o documento com o impacto da mudança nos outros documentos. Em **11/11/2010** recebemos o plano de desenvolvimento e cronograma e em **03/01/2011** recebemos impressos o Relatório de Impacto de Mudanças e a Solicitação de Mudanças. Nessa ocasião percebemos a quantificação de 30 pf para as alterações,

		que julgamos excessivas. Em reunião com CGTI em 04/02 solicitamos o envio do valor que as mudanças custariam se fossem interrompidas nesse momento, ainda na fase da documentação. Ainda estamos esperando essa manifestação da CGTI.

2. PARADOXOS IDENTIFICADOS:

1. Desperdícios gerados pela metodologia de desenvolvimento de software

Limitantes: O processo tem que ser documentado e previsível por conta do controle interno

Descrição do problema: Há um grande intervalo de tempo entre o início do desenvolvimento - definição de escopo, visão, premissas, requisitos, etc - e o momento em que algo visualizável pelo cliente começa a existir (protótipo, telas, etc). Com isso, a) muitas vezes os requisitos e as necessidades do sistema já evoluíram, e o sistema não atende mais as demandas, e/ou b) o sistema mostra-se insuficiente para atender as necessidades, por existir dificuldade na linguagem/comunicação entre demandante e desenvolvedores. Em quaisquer casos, o responsável é sempre a área demandante, que “aprovou” os documentos, embora não tivesse plena compreensão dos resultados. No documento de Visão da metodologia é dito que o processo proposto é incremental-iterativo. Mas se olharmos os Modelos de Processos do processo, nota-se que na realidade se trata de um processo cascata. A unica parte onde há um ciclo "pseudo-iterativo" está na etapa de Construção, onde após a Publicação de um componente, este pode voltar a tarefa de codificação. Parece tratar-se apenas de um processo de controle de qualidade, onde o componente é testado e se não é aprovado, volta ao desenvolvimento para correções. Não há qualquer menção ou possibilidade de mudança de requisitos, sequer há participação do cliente (conforme o modelo).

Solução proposta: O primeiro passo talvez seja f**azer com que a metodologia seja, na prática incremental-iterativa**. Outro ponto é tentar quantificar os desperdícios: Há provavelmente um grande **desperdício gerado pela idéia de que um sistema pode ser desenhado no papel** antes de ter seu desenvolvimento iniciado. Valeria a pena analisar os dados sobre a mortalidade de projetos (consta informalmente que o índice é de pelo menos 50%) antes do início do desenvolvimento (quando o cliente recebe a informação do custo - PFs) e daqueles que não são utilizados pela área pois quando ficaram prontos o processo já tinha mudado (ou o produto não se adequou a sua necessidade, embora tenha se adequado aos documentos produzidos)

2. Complexidade dos documentos da metodologia de desenvolvimento de software

Limitantes: O processo tem que ser documentado e previsível por conta do controle interno

Descrição do problema: Uma das motivações da metodologia é garantir que todos (cliente, DBA e CGTI) estão de acordo. Mas são tantos documentos e tão cifrados que **as pessoas tem dificuldade em saber o que estão aprovando**. O cliente fica prejudicado e não consegue

saber de antemão se vai ter o que espera. Se for para ter certeza de tudo, gera uma tremenda carga de trabalho para os 3 atores. Se a empresa de software tentar ser eficiente e ultrapassar os gargalos - ou seja, avancar os trabalhos - se prejudica pois pode estar trabalhando sobre premissas que não foram devidamente validadas com o cliente ou CGTI
Solução proposta: A linguagem empregada na metodologia e seus artefatos tem que ser variável de acordo com o alvo do documento. Ex: documentos de requisitos com visão de negócio, onde o principal envolvido e avalisador é o cliente, deve realmente usar de linguagem clara e comum as pessoas das áreas clientes. Documentos técnicos voltados a equipe, deve usar linguagem técnica. O problema neste caso pode não ser o processo em si ou o documento (forma), mas sim a maneira como é redigido (conteudo). Sugerimos que haja de forma mais atuante a figura do Analista de Negócio, a pessoa responsável por entender as necessidades do clientes, traduzir isso para uma linguagem mais técnica, propor uma solução que atenda a estas necessidades e documentar tudo isso de maneira inteligível e acessível ao cliente. Em suma: colocar desenvolvedor para levantamento e especificação de requisitos normalmente dá problema.

3- Gestão de segurança pode ser compartilhada entre as áreas. Exemplos de gestão ineficiente de segurança:
a- Senhas wireless duram apenas 30 dias (sem exceção, independente do visitante permanecer 1 dia ou 1 ano no MJ) e os dados do usuaŕio login+cpf são perdidos, tendo que ser redigitados todo mês. Essa responsabilidade poderia ser dividida com as áreas solicitantes;
b- Sistema de abertura de chamados SICAU não pode ser acessado via internet. Por outro lado chamados podem ser abertos por telefone sem nenhum mecanismo de confirmação da identidade. Além disso o campo assunto do email enviado pelo SICAU já poderia vir com o que está acontecendo com o processo e não há a necessidade de informar cada passo que o chamado toma.
c- Bloqueio a redes sociais: Isola os servidores do MJ de uma grande e crescente quantidade de informações, prejudicando sua produtividade. Além disso, inibe a produção de conteúdo positivo sobre o MJ nas redes. Um código de ética/conduta nas redes sociais seria muito mais efetivo. Questões de banda também poderiam ser resolvidas com gerência: contratação de banda assimétrica (+ barata) com direcionamento dos acessos ao youtube, por exemplo.

8- Questionar se existe análise qualitativa junto aos clientes. Estes devem ser o principal foco, e a visão deles tem que ser considerada. Sugestão para que façam algumas entrevistas e pesquisas sobre como eles enxergam a metodologia, as dificuldades que encontram, sugestões de melhorias etc. É compreensível que a área de TI queira se resguardar formalmente (documentos assinados pelo cliente e posteriormente escaneados na pasta do repositorio) com relação aos projetos, faz parte do jogo. A questão é como fazer isso de maneira a agregar valor para a casa (MJ) e não apenas criar pseudo-facilidades a uma área específica.

9- Para um processo de desenvolvimento de software funcionar efetivamente, todas as partes envolvidas tem que conhecer plenamente este processo e ter uma visão muito clara de seu

papel e responsabilidades dentro do processo. Não me parece o caso de vocês. Qualquer implantação de metodologia nova deve ter ter planos de treinamento de todos os envolvidos, constituição de um comitê para que melhorias no processo possam ser identificadas e implantas, com participação no comitê de todos os envolvidos também, uma estrutura de suporte na metodologia onde os usuários e clientes possam consultar e tirarem suas dúvidas, figuras em toda a organização com o papel de multiplicadores da metodologia, para que esta seja difundida de fato.
Perfeito, vou incluir essa visão no documento. Mas vejo que uma metodologia que leva em conta a participação do cliente, tem que estar baseada em documentos e processos que ele possa entender, sem ter que fazer um curso de metodologia. É uma volta àquele ponto da metodologia orientada ao cliente, ou algo assim.

Aqui há um nó que é inexorável a qualquer implantação de novo processo: este deve se adaptar completamente ao ambiente e aos processos já instituídos, ou o contrário, a implantação de um novo processo faz com que a organização tenha que se adaptar a ele? Na minha opinião, nem um nem outro. A implantação de um novo processo deve ter como meta a melhoria do processo do cliente, não impor algo estranho e que não lhe dá nenhum ganho. Por outro lado, nem sempre a maneira que se trabalha atualmente é a mais adequada, carecendo de uma análise cuidadosa para identificação de pontos de melhoria. Neste sentido, é muito difícil que o processo implantado seja totalmente claro ao cliente num primeiro momento, salvo em casos onde ele tenha participado efetivamente da modelagem do processo. Por isso, acredito que sempre é necessário o treinamento. Note que o treinamento não é DE METODOLOGIA, mas DA METODOLOGIA.

3. BREVE ANÁLISE CRÍTICA SOBRE A METODOLOGIA DE DESENVOLVIMENTO DE SOFTWARE

Em 1970, Winston Royce apresentou em seu artigo[1] um modelo de desenvolvimento de softwares conhecido hoje como "Waterfall", com o intuito de exemplificar um modelo falho (ironicamente, acredita-se que muitas das primeiras instituições que adotaram este modelo citam este artigo como referência primária). Neste modelo, as fases do desenvolvimento de software são divididas em unidades discretas. A primeira fase é conhecida como "especificação de requisitos", onde todas as descrições do software necessárias para a sua construção são captadas dos clientes. As fases restantes pretendem elaborar e construir o software de acordo com estas especificações. Finalmente, a passagem de uma fase para a próxima só acontece quando se tem absoluta certeza de que a última foi integralmente realizada.

Uma das falhas mais apontadas neste modelo se refere a fragilidade da primeira fase: é extremamente difícil obter os requisitos de forma completa assim como, em inúmeros projetos, os requisitos são instáveis, mudando constantemente com o passar do tempo. O resultado frequente são sistemas construídos que não atendem as expectativas dos clientes.

Um modelo alternativo, conhecido como "iterativo e incremental" propõe que as fases sejam cíclicas (ou seja, que as fases sejam aplicadas repetidamente) e que o desenvolvimento seja feito "de pouco em pouco". As iterações são úteis, principalmente, para captar mudanças de requisitos, e o desenvolvimento incremental permite que mudanças custem menos (pois pouco é investido na construção de um requisito que sofreu alteração). Ainda, o tipo e a quantidade de documentação gerada durante o processo varia e, em tese, está em função da natureza dos projetos e das necessidades que a equipe observa. Uma vez que boa parte das documentações está em função dos requisitos, se estes mudam, as documentações devem ser atualizadas também. Finalmente, um fator de sucesso para este modelo é que exista uma forte proximidade com o cliente e que os ciclos sejam curtos (de uma a duas semanas).

Um fenômeno frequentemente observado é a adoção de um modelo "iterativo e incremental" que, na prática, se torna "Waterfall". Sintomas envolvem ciclos longos (ex, contato infrequente entre equipe do projeto e cliente), e a observação de que o custo de alteração dos requisitos (mesmo triviais) aparece irrealisticamente alto, assim como o prazo para sua realização, e se concentra na manutenção das documentações, e não, na adequação do software em si. Em outras palavras, são sinais de que o processo foca-se em manter o processo, e não, em realizar os serviços e produtos requisitados. Ainda, é provável que uma parcela, tão grande quanto for, da documentação gerada acaba tendo pouco ou nenhum valor (de importância; monetariamente, é o inverso, contabilizando para o alto custo do processo) -- a que se disputar/reavaliar, então, sua utilidade.

Um possível colorário deste fenômeno é a alienação do cliente, onde este é submetido a avaliar e assinar documentos que lhes são indecifráveis (que pecam em excesso de detalhe técnico e/ou "burocracia insignificante") sem o apoio necessário para compreender e negociar seus termos -- não assinar significa interromper, pelo menos, parte do projeto. Este é um sinal de que a equipe e o cliente não estão trabalhando em conjunto. Mais notável, significa que a equipe não está trabalhando em prol do cliente. Isso é ainda mais nocivo por dar a ilusão aos dois lados de que progresso está sendo feito: a equipe (especialmente), por receber documento assinado, e o cliente, em ato de fé, por assinar e esperar pelo melhor. A partir daí, é comum que o projeto se desenvolva de maneira indesejável para o cliente sem que ambos vejam claramente o que está ocorrendo. O resultado vem na forma de frustração e protesto por parte do cliente e a equipe respondendo com documentos aprovados/assinados por este -- e nenhum sinal do produto/serviço, propósito único da relação, realizado com sucesso.

[1] - Managing the development of large software systems: concepts and techniques