**TwoTierReview**

Você está escrevendo casos de uso em uma série de SpiralDevelopments, e você acredita que ele é QuittingTime muitas pessoas podem precisar rever os casos de uso. Isso é caro, demorado.

A democracia representativa é um sistema de governo no qual os cidadãos escolhem pessoas para representar eles em um corpo deliberativo. Estes representantes estudam as questões à medida que surgem e votam em eles, com base na compreensão do assunto e dos sentimentos dos seus constituintes. Se alguém tem uma preocupação sobre um problema, eles podem conversar com seu representante sobre isso, potencialmente influenciando sua decisão.

Embora o sistema não seja perfeito, certamente é mais eficiente do que ter milhões de pessoas debatendo cada questão ou votando em cada decisão, grande ou pequena. Enquanto os representantes tomam as decisões, os eleitores têm a palavra final, pois podem votar nessas pessoas fora do cargo que eles sentem são ineficazes ou que não conseguiram representá-los.

Este sistema fornece um bom modelo para revisão de casos de uso, especialmente aqueles destinados a grandes, diversos grupos de partes interessadas. Ele permite que todos os grupos interessados ​​tenham sua opinião sobre o uso Conteúdo do caso, sem abrandar todos, ou exigindo tempo excessivo.

Os comentários são necessários para verificar e validar a escrita e o conteúdo. Os comentários são uma boa maneira de validar a exatidão e exaustividade de um trabalho no início do processo de desenvolvimento.

As pessoas tendem a acreditar que seu trabalho é melhor do que realmente é, muitas vezes acreditando que a qualidade de do trabalho é proporcional ao esforço gasto com isso. Mas o trabalho inadequado e impreciso pode prejudicar a público a partir do propósito de um documento, e suas partes interessadas esperam profissionalismo nos negócios documentos.

É difícil para os escritores pegar seus próprios erros porque eles estão familiarizados com seus

trabalham e tendem a extrair inferências dele que não estão necessariamente lá. Pessoas lendo um documento pela primeira vez é mais provável que note erros porque os erros os distraem

entendendo isso.

As partes interessadas têm interesse em casos de uso. Muitos grupos diferentes têm uma participação interessante em um conjunto de casos de uso e dependem deles para ajudá-los a fazer seu trabalho. Está no seu melhor interesse como escritor para consultar com eles cedo e muitas vezes, para simplificar seu esforço e minimizar o quantidade de retrabalho que você precisa fazer para produzir casos de uso aceitáveis ​​que abordem adequadamente seus problemas.

É caro, cansativo e lento envolver todos no processo de redação. Muitas pessoas tendem

para entrar no caminho do outro, são ineficientes e exigem muito esforço para coordenar.

Um SmallWritingTeam ajuda a manter o processo gerenciável, na pista e tende a reduzir no rastreamento de características.

Se apenas uma pequena equipe de redação estiver fazendo a revisão, nem todos os interesses das partes interessadas são incorporados. Isto é muito difícil, senão impossível para uma pequena equipe de redação de duas ou três pessoas para representar o vistas de uma grande audiência diversificada.

Um SmallWritingTeam sem ParticipatingAudience não tem a experiência ou a base de conhecimento diversificada para entender ou representar todos os

pontos de vista das partes interessadas sobre um grande projeto. É provável que eles vão perder partes chaves do sistema sem essa ajuda.

Os comentários podem ser caros, tediosos e demorados. Para que uma revisão seja efetiva, Os participantes precisam investir muito tempo e energia. O tempo gasto em frequentes ou longas

As avaliações, com muitas pessoas, aumentam rapidamente a mão-de-obra significativa. Portanto, precisamos ser judicioso ao realizar avaliações, agendando-as apenas quando necessário e exigindo o mínimo de as pessoas são absolutamente necessárias. Além disso, enquanto algumas pessoas gostam de revisar material de desenvolvimento,

outros desprezam. Você pode permitir que todas as partes interessadas participem, mas mantêm a presença necessária ao mínimo.

Assim sendo:

Mantenha dois tipos de revisão. O primeiro é feito por uma equipe interna menor, possivelmente muitos vezes. O segundo é feito pelo grupo completo, talvez apenas uma vez.

Primeiro, reveja os casos de uso internamente para verificar sua legibilidade, implementabilidade, precisão. Essas avaliações "internas" podem ser críticas informais, reuniões formais ou uma combinação de ambos. Qualquer tipo de revisão é apropriada, desde que permita que os revisores captem erros e verifique se os casos de uso são suficientes, tanto quanto eles estão em causa. Um dos propósitos de essas avaliações iniciais são para eliminar o "ruído" causado pela ortografia, gramatical e formatação e erros técnicos, que, quando deixados sem correção, são distrativos.

Você pode precisar manter várias dessas revisões internas quando o sistema é grande ou excessivamente complexo.

Porque as pessoas tendem a perder o interesse em discussões detalhadas fora de sua própria área de interesse, considere realizar revisões de grupo separadas para diferentes áreas funcionais ao revisar formalmente use casos destinados a uma base de clientes grande e desigual. Dessa forma, cada grupo de partes interessadas pode

Revise os casos de uso em profundidade do seu ponto de vista particular sem distração.

No final dessas análises internas, as equipes afirmam que é QuittingTime, e que

os casos de uso são completos, corretos e tão implementáveis ​​quanto precisam ser neste momento.

Os casos de uso estão prontos para o grupo maior verificar.

Mantenha pelo menos uma reunião com o grupo completo, uma vez que os casos de uso passem a reunião interna, para rever o sistema como um todo unificado. Confie na primeira camada de revisões para validar o interno funcionamentos do sistema, de modo que o segundo nível pode se concentrar em como as peças se encaixam.

A definição de "grupo completo" varia de acordo com o projeto. Deveriam ser todas as pessoas que analisam os requisitos, antes do desenvolvimento ficam longe demais. Em alguns casos, é apenas o desenvolvimento equipe, às vezes desenvolvedores mais um executivo, às vezes são os analistas de negócios e os programadores principais, às vezes são usuários, executivos e toda a equipe de programação.

O propósito das análises "externas" é determinar:

• isso é realmente o que é apropriado para os desenvolvedores gastar tempo construindo (negócios verificação de valor)?

Isso é correto como uma especificação? (As regras de negócios correm, e ela deixa aberta?

as variações permitidas adequadas na implementação. Bloqueia o importante

decisões - identifica o conjunto apropriado de questões abertas que podem ser tratadas mais tarde?)

• os desenvolvedores podem realmente construí-lo?

Exemplos:

Wings Over the World

Nesta parcela da história de Wings Over the World muitas das partes interessadas estão resmungando sobre as críticas frequentes e algumas estão se recusando a participar. É necessário que cada revisão seja um esforço para todas as mãos? Claro que não!

As avaliações dos casos de uso para verificar os casos de uso são uma representação justa da

requisitos funcionais e que, estruturalmente, os casos de uso exibem os sinais de qualidade defendidos pelos padrões listados neste livro. A maioria das partes interessadas só está interessada em rever os casos de uso para verificar se eles protegem seus interesses. Embora todas as partes interessadas ou suas os representantes querem sua opinião e querem participar do projeto, eles certamente não querem passar longas horas em reuniões formais e tediosas de casos de uso a cada semana para revisar pequenas mudanças.

The Programmer Who Cried Review

Era uma vez um programador que estava projetando um novo sistema para grandes empresas.

Era um sistema importante e afetaria os empregos de muitas pessoas. O programador queria

para garantir que os interesses de todas as pessoas afetadas pelo sistema fossem protegidos, logo depois ele terminou um rascunho da visão do sistema que ele gritou "Reveja!". No momento em que ele gritou todos as pessoas vieram correndo para a sala de conferências porque sabiam que a última pessoa lá iria ter que demorar alguns minutos. Durante as próximas duas horas, eles revisaram o rascunho do sistema, derrubando itens de ação. No final da reunião, todas as pessoas agradeceram ao programador por permitir-lhes para verificar se a visão abordava seus interesses.

O programador tomou todos os itens de ação e os incorporou no segundo rascunho do

Visão do sistema e mais uma vez ele gritou "Rever!". Assim como antes de todas as pessoas

veio correndo para a sala de conferências não querendo ser aquele que teve que levar alguns minutos. Para as próximas duas horas, todos eles revisaram o rascunho atualizado. Algumas pessoas resmungaram desta vez que eles tinha coisas melhores a fazer do que se sentar em uma revisão, mas ainda a maioria das pessoas agradeceu o programador

para permitir que eles verifiquem a visão abordaram seus interesses.

O programador tomou todos os itens de ação da segunda revisão e incorporou-os em uma

terceiro rascunho da visão do sistema e mais uma vez ele gritou "Review!". Tão rápido como antes de todos as pessoas vieram correndo para a sala de conferências não querendo ser a pessoa que teve que levar minutos. Durante as próximas duas horas, todos revisaram o rascunho atualizado. Desta vez, a maioria das pessoas resmungaram que tinham coisas melhores a fazer do que se sentar em uma revisão e que não queriam ser chamado para outra revisão, a menos que existisse algo mais do que apenas documentar a estrutura que tinha mudado.

O programador tomou todos os itens de ação da terceira revisão e os incorporou a um

quarto rascunho da visão do sistema e mais uma vez ele gritou "Review!". Mas desta vez ninguém veio correndo para a sala de conferências, nem mesmo os estagiários. "Reveja, reveja!", Gritou o programador. Ainda não veio ninguém. Então, o programador decidiu basilar a visão do sistema de qualquer forma.

Infelizmente, esta situação permite que uma característica importante caia nas rachaduras, e a empresa.

O CEO estava envergonhado com uma demonstração ao vivo quando não estava lá.

O programador acabou no Gulag do centro de atendimento porque ele chorou rever com muita frequência.