You have been writing use cases in a series of SpiralDevelopments(66), and you believe that it

is QuittingTime(71).

Many people may need to review the use cases. This is an expensive, time-consuming

proposition.

Representative democracy is a system of government in which citizens elect people to represent

them in a deliberative body. These representatives study the issues as they arise, and vote on

them, based on their understanding of the subject and their constituents’ feelings. If someone

has a concern about an issue, they can talk to their representative about it, potentially influencing

their decision. While the system isn’t perfect, it is certainly more efficient than having millions of

people debating every issue or voting on each and every decision, big or small. While the

representatives make the decisions, the voters have the final say, as they can vote those people

out of office that they feel are ineffective or who have failed to represent them.

This system provides a good model for reviewing use cases, especially those intended for large,

diverse groups of stakeholders. It allows every interested group to have their say as to the use

case’s contents, without overwhelming everyone, or requiring excessive time.

Reviews are necessary to verify and validate writing and content. Reviews are a good way to

validate the correctness and completeness of a piece of work early in the development process.

People tend to believe their work is better than it is really is, often believing that the quality of

work is proportional to the effort spent on it. But inadequate, inaccurate work can detract the

audience from a document's purpose, and your stakeholders expect professionalism in business

documents. It is harder for writers to catch their own mistakes because they are familiar with their

work, and tend to draw inferences from it that aren’t necessarily there. People reading a

document for the first time are more likely to notice errors because the errors distract them from

understanding it.

The stakeholders have a vested interest in the use cases. Many different groups have a vested

interest in a set of use cases, and depend upon them to help them do their work. It is in your best

interest as a writer to consult with them early and often, to simplify your effort and minimize the

amount of rework you need to do to produce acceptable use cases that adequately address their

issues.

It is expensive, tiring and slow to involve everyone in the writing process. Too many people tend

to get in each other’s way, are inefficient, and require a lot of effort to coordinate. A

SmallWritingTeam(XXX) helps keep the process manageable, on track, and tends to cut down

on feature creep.

If only a small writing team is doing the review, not all stakeholders' interests are incorporated. It

is very difficult, if not impossible for a small writing team of two or three people to represent the

views of a large diverse audience. A SmallWritingTeam(47) without ParticipatingAudience(50)

doesn’t have the experience or the diverse knowledge base to understand or represent all of the

stakeholders’ views on a large project. It is likely that they will miss key parts of the system

without this help.

Reviews can be expensive, tedious and time consuming. For a review to be effective, the

participants need to invest a great deal of time and energy. The time spent on frequent or long

reviews, with many people, quickly adds up to significant manpower. Therefore, we need to be

judicious when conducting reviews, by only scheduling them when needed, and requiring as few

people as absolutely necessary. Also, while some people like reviewing development material,

others despise it. You can allow all interested parties to attend, but keep the required attendance

to a minimum.

Therefore:

Hold two types of review. The first is done by a smaller, internal team, possibly many

times. The second is done by the complete group, perhaps just once.

First, review the use cases internally to verify their readability, implementability, precision and

accuracy. These “inner” reviews can be informal desk-reviews, formal meetings, or a combination

of both. Any kind of review is appropriate as long as it allows the reviewers to catch errors and

verify that your use cases are sufficient as far as they are concerned. One of the purposes of

these initial reviews is to eliminate the “noise” caused by spelling, grammatical, and formatting

and technical errors, which when left uncorrected are distracting.

You may need to hold several of these inner reviews when the system is large or overly complex.

Because people tend to lose interest in detailed discussions outside of their own area of interest,

consider holding separate group reviews for different functional areas when formally reviewing

use cases aimed at a large, disparate customer base. That way, each group of stakeholders can

review the use cases in depth from their particular point of view without distraction.

At the end of these inner reviews, the teams are asserting that it is QuittingTime(71), and that

the use cases are complete, correct, and as implementable as they need to be at this point. The

use cases are then ready for the bigger group to check.

Hold at least one meeting with the complete group once the use cases pass internal muster, to

review the system as a unified whole. Trust the first tier of reviews to validate the internal

workings of the system, so that the second tier can focus on how the pieces fit together.

The definition of "complete group" varies by project. It should be all the people who review the

requirements before development gets too far underway. In some cases it is just the development

team, sometimes developers plus an executive, sometimes it is the business analysts and the

lead programmers, sometimes it is users, executives and the entire programming team. The

purpose of the “outer” reviews is to determine:

• is this really the appropriate thing for the developers to spend time building (business

value check)?

is this correct as a specification? (Are the business rules correct, and does it leave open

the proper allowed variations in implementation. Does it lock down the important

decisions - does it identify the appropriate set of open issues that can be handled later?)

• can the developers really build it?

Examples;

Wings Over the World

In this installment of the Wings Over the World story many of the stakeholders are grumbling

about the frequent reviews and some are even refusing to participate. Is it necessary that every

review be an all hands effort? Of course not! SpiralDevelopment(66) requires regular re-

assessments of the use cases to both verify the use cases are a fair representation of the

functional requirements and that structurally the use cases exhibit the signs of quality advocated

by the patterns listed in this book. The majority of stakeholders are only interested in reviewing

the use cases to verify that they protect their interests. While all the stake holders or their

representatives want their say and want to participate in the project, they certainly don’t want to

spend long hours in formal and tedious use case meetings each week to review minor changes.

The Programmer Who Cried Review

Once upon a time there was a programmer who was designing a new system for large company.

It was an important system and would impact the jobs of many people. The programmer wanted

to make sure the interests of all the people affected by the system were protected, so soon after

he finished a draft of the system vision he cried out “Review!”. The moment he cried out all the

people came running to the conference room because they knew that the last person there would

have to take minutes. For the next two hours they reviewed the draft of the system, taking down

action items. At the end of the meeting all the people thanked the programmer for allowing them

to verify that the vision addressed their interests.

The programmer took all the action items and incorporated them into the second draft of the

system vision and once again he cried out “Review!”. Just as quickly as before all the people

came running into the conference room not wanting to be the one who had to take minutes. For

the next two hours all they reviewed the updated draft. Some people grumbled this time that they

had better things to do than sit in a review, but still most of the people thanked the programmer

for allowing them to verify the vision addressed their interests.

The programmer took all the action items from the second review and incorporated them into a

third draft of the system vision and once again he cried out “Review!”. As quickly as before all the

people came running into the conference room not wanting to be the one who had to take

minutes. For the next two hours all they reviewed the updated draft. This time most of the

people grumbled that they had better things to do than sit in a review and that they did not want to

be called into another review unless there was something more than just document structure that

had changed.

The programmer took all the action items from the third review and incorporated them into a

fourth draft of the system vision and once again he cried out “Review!”. But this time no one

came running to the conference room, not even the interns. “Review, review!” cried out the

programmer. Still no one came. So the programmer decided to baseline the system vision

anyway.

Unfortunately, this situation let an important feature fall through the cracks, and the company’s

CEO was embarrassed at a live demo when it was not there. The programmer ended up in the

call center gulag because he cried review too often.