# QUALITY ASSURANCE:

## *A Requirements Management Perspective*

By: David De Witt, Ph.D.

nuevista group

Quality. Experience. Integrity.

www.nuevista.com

## ABOUT THE AUTHOR

**David De Witt**, Practice Director for NueVista Group's Enterprise Requirements Management Practice, has more than 20 years experience in organizational leadership, management consulting and Information Systems optimization in a number of fields, including supply chain and third party logistics; financial services; and telecommunications.

He holds the prestigious Certified Business Analysis Professional designation from the International Institute of Business Analysts, awarded to candidates who have successfully demonstrated their expertise in the burgeoning specialty of business analysis.

De Witt has served as a member of the Board of Directors, Midwest Facilitators Network, since 1997 and is an Advisor to the College of Business, Indiana State University. A Phi Beta Kappa graduate of Furman University, Greenville, SC, he earned his Master's degree and Doctorate at Tulane University.

CBAP® Certified Business Analysis Professional

IIBA™

nuevista group

**NueVista Group** provides a suite of expert IT consulting, support and staffing services designed to drive maximum value from technology systems, applications and strategies. Our approach is fact-based, implementation-oriented and based on clear, mutually established goals. We combine best practices and a rigorous hiring process to provide clients with the resources and solutions to achieve technology goals.

Our unique combination of background and experience—each member of our core team has more than 20 years of experience—assures our clients of exceptional value. We are committed to facilitating the transfer of knowledge to empower client teams and individuals.

# TOP 10 REASONS You Don't Need a Requirements Document When Upgrading Software

By David De Witt
*Practice Director, NueVista Group*

Everyone knows it's a pain to create a requirements document, especially when all you're doing is upgrading an existing software application. The process is tedious, time consuming and potentially treacherous—what happens if something is forgotten or omitted from the list and neither the business nor the vendor is willing to accept responsibility?

With my tongue firmly in cheek, I offer ten good—or at least very common—reasons, with the wisdom of supporting clichés, why a firm has no need to develop a comprehensive requirements document before updating an existing software application.

With all due deference to another David who, admittedly, is better known for his Top Ten Lists, I offer these in no particular order. ▶

### ❶ Save Time.

Compiling a requirements document takes so much time. You have to talk to the people who collect data, input it and use it. Besides the time required just to talk to them, you'll have to compile the information, analyze it and then decide what you already know – the old system is not as good as the upgrade will be.

### ❷ Save Money.

Well, time is money, so the time you save in point one, above, is money saved. Think short-term efficiency and productivity.

### ❸ An Upgrade is a One-time Expense.

Look at the cost of the upgrade as an investment in the business and you'll have to consider your return on investment. Such a calculation requires you to consider how much more productive the organization will be with the upgrade – and at what cost – compared to the current application. This can open Pandora's box or a can of worms; it's your choice.

### ❹ Eliminates Outside Interference.

More often than not, a good requirements document will involve an objective outsider, who will ask all kinds of questions: What do you use most? Who uses the data? What reports are necessary? What doesn't work as well as you'd like? What do you use in this system that must be in an upgrade? This can lead to too many suggestions for changes, too much anticipation about correcting shortcomings in the current version and too many cross-functional contradictions. And remember: too many cooks spoil the broth.

### ❺ Diverts Your Users' Attention.

Asking users what they want in an updated software application can be counter-productive. It takes their attention away from the real work they're doing and it forces them to think about what could be done better, faster or more efficiently on the updated version. That process takes time, costs money and could postpone installation while your software vendor makes the requested changes to make the program work the way your users want it to. See points 1 and 2 above.

### ❻ Users Are Very Good at Creating Workarounds and Desk Drawer Systems.

If it's not broken, don't fix it, right? Your users know this system so well and they've used it for so long they've developed their own ways to manipulate the data and generate reports the business requires. By learning that your users have developed their own "features" – an inevitable outcome of the requirements development process – you will be forced to ask some serious questions. Why the workaround or desk drawer system? What in the current system made either one a solution? Are these problems or situations addressed – some might say resolved – in the upgrade? How? Why? Or why not?

### ❼ Documentation is Unnecessary.

Let's take the bull by the horns on this one. If you're going to do a costly and time-consuming requirements document, you simply must compare the documentation for the old system with the documentation for the new system. What matches? What doesn't? Why? Why not? Reviewing the documentation also might tell you how some of the old functionalities have been changed, or omitted. Can your business live with those changes or without those functionalities? The answers to these thorny questions can take your eye off that impressive list of features in the upgrade. And once you start asking yourself "how can we use this or that feature?" instead of focusing on what your business requires of the software, you're on a slippery slope. Ignoring the questions, however, does provide a great opportunity for the business and your users to live and learn later.

### ❽ Listen To Your Vendor.

The new version will be easier to maintain, the vendor says. And besides that, the vendor won't support the old version. Accept these promises on good faith, so your IT department won't be inconvenienced by costly and time-consuming maintenance – and your users will be able to use the upgrade (it's just a few tweaks to the old system they know so well, right?) with ease and confidence.

### ❾ Features Are Not Requirements.

Features that come with the update are free, aren't they? It's a no-brainer: even if you don't need them right away, who knows – you might use them later, and you've already paid for them. On the other hand, you'll probably have to pay for specific functionalities you've identified in a requirements document. But remember: If the folks who must use all the "new and improved" features in the upgrade can't do their jobs because the features don't meet their requirements, the new upgrade won't work.

### ❿ Requirements Are Not Features.

Asking your users to identify all the features they like will simplify your task, for sure. But asking them to tell you what their requirements are won't make your job any easier at all – see points 6, 7 and 8. Think of features as all the expensive options on a new car that has no battery – they're nice to have but they're of no use until you can start the car and drive away. Such a situation creates new opportunities, however, for your users to create new workarounds and desk drawer systems, and that will help you justify the absence of a pesky requirements document.

After writing the preceding article, I wanted to provide a realistic counter-point—a list of ten reasons, drawn from my own experience—to clarify why you MUST have a Requirements Document in hand before you invest money in upgrading or purchasing software.

With all due respect to another and far better known David who offers his top ten lists much more frequently, here are my top ten reasons that creating a requirements document is in your best interests—and your organization's. ▶

# TOP 10 REASONS You *REALLY DO* Need a Requirements Document When Upgrading Software

By David De Witt
*Practice Director, NueVista Group*

### ❶ Requirements Are Not Features.

You simply must know what your users must have to do their jobs. Distinguish thoughtfully and carefully between what is needed and what the vendor is offering. At the same time, you must be able to factor into the requirements documentation other considerations, such as budget limitations and regulatory constraints. You will need to train your users how to specify their requirements, which takes tact, training, and patience. A request usually starts with: "All I really need is " and then wanders away. A vague or ambiguous statement – regardless of its intent – will consume user time, IT time and cause some other loss, such as the opportunity to do another project. This issue is methodology neutral: Agile methods will have the same problem if user time is not focused and users are not trained. You cannot fulfill the requirements until they're precise and complete and you understand why they're important to the business.

### ❷ Features Are Not Requirements.

Eliminate decisions made by squeaky wheels – sometimes referred to as "design by whine" – where the loudest department gets the most resources. Your goal is to make certain that business decisions made with regard to requirements (and their associated expenditures) will generate the return you're expecting. Start by working backward, to match the problems solved with existing software to identify the problems that must be and can be solved only with new software. Don't lose anything you have now that is essential to the business; it's easy to forget "hidden" requirements that oftentimes are taken for granted or ignored.

### ❸ Work With Your Resources and Manage With Discipline.

Developing an effective requirements document costs money and managing resources requires discipline. A politically expedient decision, even if it's cost effective in the short run but is not the right decision in the long run, will cost time and money – eventually. From a strategic point of view, ask if the requirements process itself is going to cost more than it's worth. You will find that using the enterprise requirements documentation process will help you avoid useless meetings, which saves time and money, but more about that later. When the appropriate constituents are communicating with each other, they accept the value of their participation because they can track the project's payback to the entire organization.

### ❹ On-going Documentation Is Essential.

It's necessary that you continue to manage your requirements, because sometimes internal departments continue to push for a project, application or feature that was denied in the last upgrade or purchase. Perhaps it wasn't worth the cost, or its payback wasn't acceptable to the enterprise as a whole. By maintaining continuity in your requirements documentation, you can see why a prior request was denied. If the reason hasn't changed, it's likely there is no need to investigate it again— and you can avoid bringing in a "new" solution to solve an equally "new" problem. Referencing prior documentation also can help you eliminate ongoing costs for applications and hardware that are no longer used.

### ❺ Identify and Understand the Workarounds and Desk Drawer Systems.

You need to understand what in your current application made a workaround or desk drawer system an attractive solution. Then, make certain the upgrade eliminates these systems and their causes, with this caveat: You might learn, in the course of such an investigation, that a certain desk drawer system has a unique function or user, and that it's best left alone.

### ❻ Focus Your Users' Attention.

You want to learn from them what can be done better, faster or more efficiently, but realize that users don't always know their current business needs. In fact, users at different levels of an organization have different perceptions of business needs, priority and urgency. They tend to segregate by department or management level dismissing problems because "it's not an IT issue"— but an effective requirements document requires integration and impact analysis for all departments. Know, too, that the needs of other stakeholders (senior management, operations personnel and human resources, for example) must be considered. User frustration comes from asking for one thing and receiving another; help them articulate their needs carefully and fully. When what a user asks for is not possible, feasible or within budget, say so, because unmet expectations foster dissatisfaction. Always ask what's the relative value of a new "requirement" to avoid spending $10 to save a dime. And don't forget: users take what they have for granted, so make sure it's carried forward in the new request.

*Continued on the back page.*

*Continued from previous page...*

### ❼ Benefit From "Outside" Expertise.

An objective outsider, can, will and should ask the questions an insider cannot. What works? What doesn't? What frequent requests create problems because they're difficult to meet with the existing application? That's the outsider's function – to have no vested interest in how the problem is solved. It's a common pitfall in requirements documentation to describe a problem in terms of its supposed solution – which might not be the best, or most cost effective, approach. An outsider also can provide useful guidance on how to handle regulatory requirements that affect information systems, but that most business people are not aware of, such as the prohibition against retail businesses retaining credit card numbers after a transaction.

### ❽ Consider Your Upgrade As An Investment and Apply Metrics To It.

A comprehensive requirements document will help you decide if the benefits of the new application or upgrade justify the cost. How much more productive can the organization be with the upgrade, compared to the current application? Track paybacks for projects and information assets to determine when to re-invest or stop further investment. It's possible that an application was a good investment— at the time. When it was installed, perhaps it saved every one of your 5,000 clerks 30 minutes a day— but today, you have 5 clerks and it saves them three minutes a day. That feature probably isn't worth the cost. And some stand alone features – such as desktop publishing – aren't needed now, thanks to the tools available in word processing applications.

### ❾ Save Time.

Reduce rework by spending time in the requirements gathering and analysis process, where it's much cheaper to eliminate a "need" than realizing it costs too much (or isn't cost effective) downstream in the design, development or application stage. Time invested in the initial steps of the process provides more time for decision making at the right time. More control of the process will help keep it moving; participants will know what to expect as you move from blue sky thinking to brainstorming to understanding to decision making. Tracking the results of your meetings will document what you've agreed to do and what you've agreed not to do and who made the decision.

### ❿ Save Money.

To start, you will eliminate ongoing costs for applications and hardware that your requirements information gathering tells you are used no longer. With the right subject matter experts and decision makers involved at the right time, you will not be able to move forward on ideas that sound promising but contain potentially expensive problems that would have to be solved later in the process— and at a greater cost. You will open many avenues to managing costs, whether it's eliminating unnecessary systems, consolidating a value-added system or evaluating long-term versus short-term value. You might learn, for example, that in the year required to acquire and install a new system the opportunity to use it will disappear. One of the most promising benefits of developing enterprise requirements is the strong possibility you will establish standard solutions to many of your seemingly "unique" problems, obviating the need for custom solutions— and that can be a real time and money saver.

nuevista group

NueVista Group, LLC
1301 West 22nd Street, Suite 802
Oak Brook, IL 60523
630.472.6838

**Quality. Experience. Integrity.**
www.nuevista.com