Implementation Challenges: Another Ten Steps

By Robert Kennedy,
Vice President,
Business Development

Over the years, I've read dozens of articles discussing the "Do's and Don'ts," or the "Ten Reasons (sometimes 12)" revolving around the success of failure of WMS implementations. These are popular topics. I've been to many of these talks. I try to read articles about WMS whenever I see them. I can claim that I usually pick up at least one useful tidbit every time. I will also lament that in many cases they are too broad or too academic or simply too obvious. Nevertheless, they are popular, so much so that a person unfamiliar with the WMS world who might see these pieces could logically conclude that:

1. WMS implementations must be difficult

2. WMS implementations must frequently go awry

3. There seems to be no shortage of people who apparently know how to do it, or at least think they should share their wisdom.

WMS implementations are difficult and often do conclude with less than the optimistic expectations with which they started. This is coming from a guy who has been doing WMS implementations for over 30 years and has seen at least 100 projects. I've learned a thing or two, sometimes by doing things right. I'll admit though, that I've probably learned much from when things didn't go so right. . Those are the ones you remember; the ones you avoid the next time.

What I have come to understand is that, thirty years later, it is surprising how the same mistakes continue to be repeated over and over. It is surprising how WMS implementations are still very difficult, too frequently prone to failure, and how the issues encountered in an implementation ALWAYS seem to be repeated over and over. Isn't anyone reading the "ten Step" articles? So, below, I give you my own version of the Ten Steps, items either to avoid or to definitely not avoid. Some of these you've likely heard before. Some I hope are new and useful.

1. Enhancement Requirements

I could write an entire article covering just this topic. There have always been two types of customers I dreaded working with. The first one stakes out a position that "Your software has to work the way we need it to!" The second, on the other hand will say, "We want the base package – no enhancements!" Opposite ends of the spectrum. Both of them equally problematic.

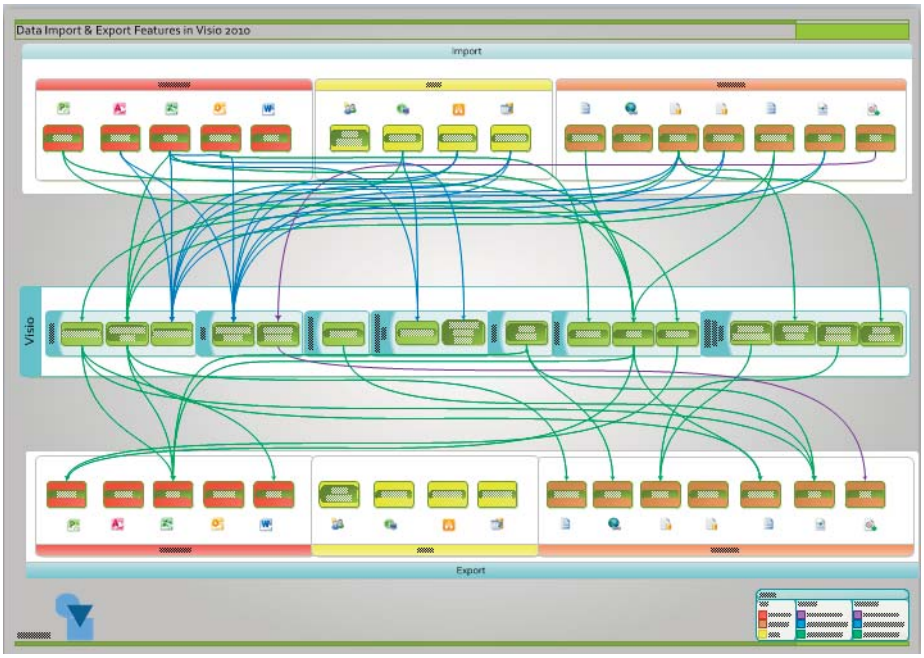
Fact is, even after decades of product evolution, all implementations continue to require software enhancements. The amount and extent will vary, based on the vendor and on the ability of the customer to adapt their processes to the software.

This is a difficult subject; the software has to work to support the requirements of the business, not the other way around. But it a false argument to conclude that it's a zero sum game. The challenge of any Gap Analysis is to strike a balance between the way the system works and the way your business operates. Every gap that pops up in a requirements session needs to be viewed as an enhancement only as a last resort. Measure every enhancement against a quantifiable ROI. In every case, ask these questions with every single enhancement request:

1. What is the negative impact to the operation from not doing the enhancement? In other words, just because we do it today, do we really need to continue doing it?

2. Can we avoid the enhancement by changing the way we do things so that the current functionality can support it? What will that cost us versus what the enhancement will cost us?
3. If we need it, can we design and deliver it as a non-core enhancement so that the vendor is not forced into changing the core.

Consider that each new core enhancement is a new piece of software that has never been used. The core product on the other hand, has been used and is in use by multiple customers. It's been battle-hardened. Despite detailed design sessions and multiple bouts of testing, new core software has inherent risk. It is always highly probable that introducing new core code into an implementation project will impact the budget and schedule, and very possibly will be a cause of trouble at Go-Live. And it's your team that has to ask the questions. Don't rely on the vendor, or your consulting partner to push back against enhancements. Some will. Some won't.



2. Understaffed Project Teams

Most projects are understaffed both by the vendor and customer. By understaffed, I am indeed talking about the number of bodies, but, more importantly I am talking about the very critical resources from that bring the deep knowledge of the operations. Those customer resources who know the operation at the deepest detail are seen as too important to remove them from their "day job" to dedicate them to the project. As a result, you end up with a solution that is being designed by resources who do not know their operations well enough. The formula ensures that the solution will rely on more enhancements and will be result in an operation that is less than it could be.

Sometimes, you simply have to bite the bullet and get you're best and brightest to the table when they are needed and for as long as they are needed – but not a minute longer. Your people are the most valuable on the project and therefore they cost you the most and must be managed most efficiently. But, they HAVE to be there. Try applying a cost to this talent, just as you apply a cost to the vendor talent so that you manage them effectively. In other words, what if your expertise cost you $200 an hour like the vendors? Don't make them attend meetings where they aren't needed.

It's the folks on the floor who know how things work, much more so than the supervisors they work for. There is hidden and therefore lost knowledge when the warehouse workforce is left out of the process. I have always found it the best practice to bring in the gang from the warehouse to review requirements and designs. It never fails that someone will point out, often to the embarrassment to others that an idea won't work. Usually, it's because of something someone forgot that only the workforce really understands. Another reason to involve the workforce: see Item 9 below. You want your users to be involved in the process so they accept the process as theirs, not managements, not the vendors.

3. Scope Creep

Expose a kid to a candy store and he will explain to you why he needs every item he spies along the way. He will convince himself and try to convince you that he absolutely NEEDS those items. This is a runaway situation of scope creep, where the definition of a "Want" becomes confused with a "Need." With the potential for all those bells and whistles in front of them, customers get greedy, lose perspective, and start asking for the world and asking for it NOW. As a result, the fundamental requirements give way to a "new" set of requirements with more cost, complexity and time. This is a profile of a WMS project where unnecessary enhancements are inserted against a schedule and budget that accounted for none of them. What is amazing to me is the customer who decides that they simply cannot live without some feature or function, forgetting the fact that they are buying the WMS because they HAVE been living without that feature or function. You have the rest of the product life cycle to expand its use. You have adaptive software tools to build your own functionality. Plan your implementation that way. What you go live with should be a baseline of what you need to ship orders to your customers. The fact is everything else can come later.

4. Realistic Schedules

Customers often set artificial time lines that may or may not have anything to do with what their business needs. They take on unnecessary risk to satisfy the demands of management, or to achieve a project ROI that is based on forecasts that are almost always overly optimistic. Stop pretending your schedule has life and death consequences (unless it does). Unless you are in a highly seasonal business and the new WMS is necessary to support your next season (a very scary situation to be in), don't sacrifice quality, don't take on more risk, and don't give up functionality just to make the schedule. Every impact to a schedule can be evaluated as an ROI. What is the cost of moving a month versus the cost of sacrificing some part of the solution? When the schedule is cast in concrete, when time creates pressure it's usually testing and training that take the hit. Not a good idea. An under tested system is suicide. An under trained workforce creates huge quality, productivity and acceptance problems. As the saying goes, "You are going to test, the questions is whether

you are going to test in a test environment or in a live production environment.

5. Application Expertise

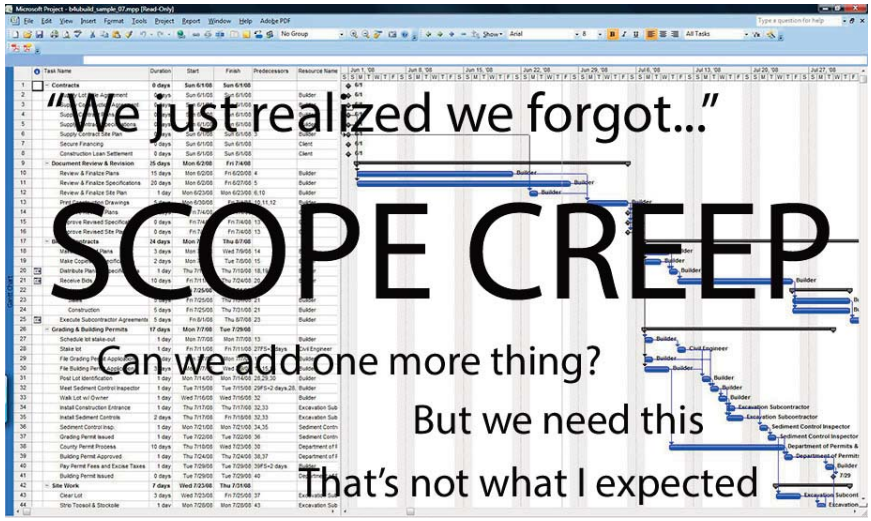
Hedge your bet by investing in third party application expertise. With the large vendors, there is an army of consultants who know the system well, even better in some cases than the vendor resources. Use your third party resources to protect your interest. Use the vendor resources to force them to keep skin in the game. The bonus: these third party folks will cost you less and will advocate for you. I witnessed a recent project where the vendor was preparing to deliver 90 plus enhancements, largely for reports and screen changes, simply because no one took the time to really understand what was already available in the system and no one pushed back on the customer who requested enhancements. It was a third party application consultant who finally raised his hand and said, "Uh, Wait a second." When we finally went through a second review exercise, the 90 became 30.

6. Third Party Tools

Invest in a report-writing tool and invest in your resources to make them autonomous in being able to do reports. Huge investments are made in having vendors change existing reports for operational needs and whims. There is no such thing as a standard report so quit pretending there is. Every user wants to look at data differently, so don't fight them, don't pay the vendor, make it easy for them to do it themselves.

7. Conversion Planning

When the system environment is built, when the facility is prepared, when testing and training are complete, it's time to convert. Manage the Conversion. It starts early in the process by putting together a Conversion Plan. It includes steps you will take to prepare the workforce for what is coming and to prepare your customers. It includes how you manage orders the first few days, when the learning curve is at its steepest, and how you deploy your resources on the floor to quickly react to issues as they appear. Never let your users languish too long with a question or error – if they don't compound the problem, they will lose confidence from the frustration they feel. A conversion plan has to include the number of system



knowledgeable resources deployed to the floor, assigned to a specific group of users, and responsible for a quick response to issues as they arise. The users need to know who these people are and where to go to get help. The steps you take to actually turn on the system need to be synchronized like a Broadway dance routine. Put it on paper; make sure everyone knows what is going to happen when.

8. Market the Project

Market the project to your workforce. Never underestimate the positive effects of communicating the change. People resist change as part of human nature. Everyone wants to live in his or her defined comfort zone. It's not the system they are resisting, it's the change! A new system rips your workforce out of their comfort zone, creating resistance. When you communicate the change and expose your workforce early, you permit an easier transition into a new comfort zone. Think slipping into a swimming pool vs. being tossed in.

Recognizing this, it is a good practice to identify a leader or leaders in the warehouse – that person or persons who the rest of the staff looks up to, and bring them into the fold – put them on the project. You need to get that person into a new comfort zone so that they will go back to the staff with a positive message – making acceptance easier and shortening the learning curve. I also like to get the person who is most negatively verbal about the change. They aren't resisting the change itself, they simply live deeper in their comfort zone. When you can get that person into a new comfort zone, they will become your loudest cheerleader.

When the system goes live, you will be better served if the workforce sees themselves as part of the team. If not, everything that happens during the first days will be "the systems fault." That's what's changed so that will get the blame, regardless of what the real issue is.

9. Testing

Again, I have to put an emphasis on testing. Testing is an investment. Invest well and you will be rewarded with a generous payback at start-up. Failure to test properly and it will come back to bite you during the start-up. Sounds obvious, but when schedule and budget pressures kick-in, testing is one of the first things to get impacted. That's always a mistake. The best investment is to use a structured, formal process, one that forces discipline and standards into the conduct of testing. There are various sources to go to for Life Cycle Testing and Validation procedures. It will take longer but will be worth it.

What isn't always obvious when developing a test plan you design is to also include test scripts these other areas:

a. The data
b. The process
c. Performance

Think of it this way, you are testing the SYSTEM, not just the software. Missing, incorrect data and configuration mistakes are just as deadly as software issues. Individual issues may be easy to correct,

but when your business operates with tens of thousands of SKU's, your start-up can be the victim of a million pin pricks. Process testing and also haunt you. I have been involved in projects where the software worked, the data was clean, but the operation was brought to its knees because the RF messaging was too long and cumbersome. Same thing can happen when the system is improperly sized. Everything works, but things are moving at a snail's pace because the system is undersized.

A comprehensive day in the life testing scheme that uses real data and approximates actual volumes is the truest way to bring it all together and ensure the best position for that opening week.

10. Training

I once participated in a project where the customer agreed to take responsibility for user training on the RF. Our team was responsible for training only the Superuser and Sys Admin. On our scheduled day 1, the customer still had not trained the workforce, so we expected a f delay in order for that training to take place. Not necessary said the customer. He got on the PA system and called an all hands meeting in the warehouse. When everyone arrived, he jumped up on a stack of pallets, held up an RF device and said, "This is your RF terminal, do what it says." Training complete. I looked at my colleague and said, "We're screwed."



Of course we had a difficult start-up. The only training these folks received was literally on the job training. It took us weeks longer to get through the learning curve, and then a couple more weeks to correct the order and inventory issues that certainly arose from improper training. This is an extreme example (but I swear it's true), but the fact is that again, as schedule and budget pressures intensify, training also gets squeezed.

As with testing, training is an investment. Pay me now or pay me more, a lot more, later. Resist the temptation. Structure your training so that it includes a real certification process that documents that all users have been properly trained (and tested?) on every facet of the system they will be required to use. Focus on exceptions. The routine processes are usually a lot easier to understand. Exceptions are not. Too many projects ignore the exceptions because they occur less frequently. Yet when they occur, they can wreak much more havoc and negatively impact the work force's confidence in the system.

About the Author

For more than 30 years, Bob Kennedy has planned, developed and implemented leading-edge supply chain execution systems around the globe. Bob has led more than 100 large-scale implementations of WMS in more than 15 countries and a variety of industries. He has earned an industry reputation for quality and a track record of successes. Bob is a partner at DMLogic, a team of WMS experts who provide services, support and WMS products. He is a member of MHI and CSCMP,and is a frequent presenter at conferences including ProMat, MODEX and WERC and CSCMP. Bob has a BS in Industrial Management from Carnegie-Mellon University.



DMLogic LLC
661 Andersen Drive
Foster Plaza 7
Pittsburgh, PA 15220
412.458.4010
sales@dmlogicllc.com
www.dmlogicllc.com