Enterprise System Implementation Best Practices

Paul Deis

Contents

Overview and Background

What makes major software implementation projects so challenging, risky and cost so much? Research has shown consistently that more than 70% of all ERP system implementations result in one of these conditions:

- **Outright failure** – damages the company's ability to perform for some time; sometimes resulting in bankruptcy or closure (rare, but does occur).

- **Troubled operation** – the system "gets up" but the company operations are *less* efficient than before, due to the increase in cumbersome-ness and clumsy or gap-ridden processes. We have seen this especially in well-funded startups where rapid growth was expected, and system conversions feared. Many of these were so hampered by putting in a Fortune 100 level ERP system in a startup with 50 people that the venture never got moving well.

- **Protracted transition** – after the initial Go Live event, there are endless meetings to resolve a myriad of problems that have remained or resulted.

- **Acceptable performance but higher costs** – the new system is OK, but doesn't add anything measurable to performance – same customer response times, product/service quality, etc. Just better looking reports, screens, maybe a web interface, but nothing with major cost reduction or strategic benefit impact. Costs are higher to pay for the new software, hardware, and implementation project (which costs a lot, of course).

So, only about 30% of these projects result in a measurable benefit and significant operational performance improvement to the company. The frustrating thing to many in these projects is that the software itself is almost never the problem – i.e., the functions needed to create the expected benefit, but somehow, the don't get used properly, or otherwise don't generate the benefits that were used to "sell" the project to top management.

It is probably a safe bet to assume that most of the people leading the 70% that didn't go as planned have prior experience, have read the leading books on implementation, attended seminars, training, and other valuable preparation for the task at hand. It is probably a safe assumption that the people running the company are not seriously deficient mentally – after all, they are running an otherwise successful company, a challenging task in an intensely competitive world. So what is going on here? How can otherwise intelligent, informed, well-intended people have such a poor track record?

A number of factors have "conspired" to make implementations challenging, such as:

4   Enterprise System Implementation Best Practices

- **Truly complex software** – any large scale system designed for use by a wide variety of companies, industries and business styles has literally thousands of options built into it. Configuring and selecting these can be a serious challenge.

- **Highly integrated functions** – part of the advantage of bigger, enterprise-wide systems is there high degree of integration. Ironically, this becomes part of the implementation challenge, preventing simpler, localized, department level implementations one at a time, mitigating heavily for a "big bang," all at once Go Live event.

- **Incentives in the wrong place** – those selling enterprise systems and implementation projects frequently have a number of reasons to drive the cost UP, such as:

  - **Sales commissions** – which can be stunning; often including hidden "finder's fees" for referring firms.

  - **Higher billing revenue** – from more consultants and hours charged to the project; these individuals may stress the essentiality of their consultants' expertise and experience, sometimes terrifying the project's leaders with various tales of difficulties at "company X" who did it "on their own" and fell into the "flaming pit" as a result.

  - **Risk aversion / mitigation** – internal project stakeholders may feel safer with larger budgets, project teams and more consultants and experts.

- **Cost-only Estimating** – using a multiple of the software cost, which has only a little to do with the level of implementation difficulty. This can produce a costly project simply because the budget is there – and the work expands to consume the budget, of course. If you get a deep discount on the software fees, does this mean that the implementation will be simpler?

Providing a larger budget and expectations for a protracted, complex, drawn out implementation process has a strong tendency to become a self-fulfilling prophecy.

Machiavelli principle – implementation projects "establish a new order" which has long been identified as carrying high risk to its leader – one where there are few friends, and many enemies. In many of the unfortunate situations project management ends up consistently in a defensive posture, working in a conflicted state where each step forward exposes the manager to more criticism, opposition, and various fears.

To pretend there is no risk is, of course, not a wise move. However, bearing this principle in mind, senior management can, in effect, "shield" the project manager by providing solid, consistent backing. This must go beyond just budget to clear, explicit leadership via words and deeds that clearly show support and an intention to share all risks associated with the changes being made.

If there is even a *hint* by top management that, in the event of a short-fall in the project that "heads will roll," everyone will take cover, and the project will soon make no steps forward without solid CYA material in place, greatly hampering its effectiveness.

Successful – Best Practice Implementations

Having established the risk of significant system implementation projects, we can now detail the Best Practices that under gird the many successful projects that delivered on their potential.

Overriding Principle – Ownership – At every level in a Best Practice implementation project there is the principle of ownership. The project that comprises the implementation is, itself a process – one that results in a slew of other processes. This means that the project manager, the team members, and adjunct participants such as those in functional work groups whose role is to interface with the implementation team – all of these individuals must *feel* the ownership of their tasks, that it is *their job* to see it through to successful completion, and to work as a team to bring this about.

The principle of ownership and its importance can hardly be overemphasized. Here is why:

- Risk mitigation - The team will not just "jump off the cliff" blindly. They'll move when they are ready – because it is *their* project, *their* success.

- Details – ownership helps insure that participants are truly paying attention to relevant details – and ignoring irrelevant ones.

- Education and training – when non-owners attend classes, one is luck if they a) show up, b) stay awake, c) retain much. In contrast, if it is *my* project, *my* success at stake, I *will pay attention* and make *sure* I retain everything I need.

- Preparation for Go-Live – a goal driven team that owns the outcome, the success, will not agree to a go-live until they *know* they are truly ready. At this point, the risk of failure or serious problems is virtually zero.

- Painless transition – if the team owns its success, and is prepared, the Go-Live is almost just another work day. There is little or no re-training of everyone, because they are prepared, looking forward to working with a new, better system and work flows.

Implementation Landmines

Sometimes the quickest way to get focused on the best way to do something is to identify and eliminate the myths, misunderstanding and error prone pathways. Here are a few:

- **Parallel implementations** – still talked about, amazingly enough. Here's why they never work:

  - Only one "real" system – the new system is always "catching up" with the real system. Often ends up sabotaging the credibility of the new system.

  - Disagreement – since the new system is always behind or out of synch with the real system, it is perceived to be the one with the problem.

  - Resources insufficient – if the company has enough extra resources, in all affected work groups to operate two systems in parallel, properly and completely, reconcile their differences, etc. what does this say about the number of extra people?

- **"Big Bang"** – this refers to the method of completing technical testing, some end user training ("show them how it works"), then convert the data over the weekend, and "turn it on." In one company we worked with this approach in an over 30% production shortfall the first month – and the company was poorly organized and inefficient to begin with! It was a disaster, needless to say.

- **Module by module** – also talked about, but almost guaranteed to result in confusion and difficulty. Unless the modules are almost completely independent of each other, the integration issues will kill this approach, as users struggle to enter the missing data in the un-integrated portions of the other new/old system – or both. Most systems, in any industry, now are very tightly integrated. This is how they provide much of the

efficiency and benefit – faster, more accurate communication and data by eliminating the departmental "silos" used in the past. Implementing, say the sales order module only, separate from the rest of the system requires that other, manual or spreadsheet, or whatever, be used to communicate sales information elsewhere in the company.

- **Financials first** – CFO's love this approach, of course, and it CAN go well under some circumstances. However, unless things are really a mess, it is not the GL/AP/AR modules that are providing the benefit to the company for the project. So, why are the functions that ARE providing the benefit being significantly delayed while sub-ledgers are reconciled, and the chart of account issues are resolved? Our view is that capable accounting functions CAN, with extra resources, create the necessary journal entries from operational data to adequately complete the month-end financial reporting.

  Putting the financials first inevitably results in waiting, just waiting, until a year end, or at least a quarter-end to convert things, further delaying operational benefit, and exposing the whole project to the effects of time-delays – changes in key leaders, project champions, loss of focus, etc., none of which improve the time-to-benefit or lower the risk of problems.

- **IT run project** – viewing a new system as primarily a technical problem, to be delegated to technical staff members who have little, if any involvement with operations, and seldom even understand business well, is a cause of major difficulties. In one company we know, the very well-meaning IT team led the selection project, installed the software, but half-way through the implementation process everyone realized that they had purchased the wrong software product for their company the fit was so bad. Valuable resources and time-to-benefit was irretrievably lost for this rapidly growing company when they had to regroup, re-do the whole selection process, hire new leadership and re-do the implementation from scratch.

- **Outsourced implementation** – we'll have more on this later, but overall, the idea that external experts can manage and perform an implementation is a cousin of the idea that IT can perform the task. Experienced IT managers will say that the only thing worse than having the IT manager select the system and run the project is to have consultants do it. And experienced consultants will concur wholeheartedly. Its too organic to a company to have it treated like installing a phone system or office furniture.

Leadership – Key

When a team of change agents, which is what an implementation team is, is organized, it is vital that, at the very start, that top management display real leadership – backing the team's efforts, making sure everyone in the company understands their role, its importance, and that management is "in the boat" with the team – and will succeed or fail *with* the team.

The oldest project joke is that the most important task at the start is to "figure out who to blame when it fails." Unfortunately, this is what happens all too often. Make sure there isn't even a suggestion that this could happen and your project will go well.

Experience – remember that most people on a project team have, at most, participated in one, perhaps two implementation projects in their career. This is OK, normal, but needs to be factored in. They are learning both how to do *this* one, and how to do them in an overall sense.

Top Management Interface – Best Practices

Every major project in a mid to large sized company needs a process to connect it with the CEO, ideally the Board of Directors, key investors, and "C-level" executives. There are several ways to accomplish this, such as:

- **Executive Management Team (EMT)** – in mid-sized or smaller companies, the project leader can just interface directly with the executive top management team. This provides an immediacy and a reality to the endeavor, as the EMT is fairly closely involved, and must understand and concur with all major issues, resolutions and decisions.

- **Steering Committee** – these can be very effective, or not, depending on how they are constituted, their charter, and how they are led and operated. We define a Steering Committee as an appointed group of senior level executives, either a mix of "C" level and below, or managers who are most affected by the project. Typically these are not the EMT, but are appointed by the EMT to function in their behalf. Hazards with Steering Committees:

  - **Disconnected from top management** – since the project SHOULD be tightly tied to the company's business strategy, having an additional reporting layer (read: " insulation layer") allows the EMT the costly luxury of imaging that they don't need to be concerned with it, seldom a good idea.

- **Second-guessing / excessive approvals** – a poorly led steering committee will require the project manager and his / her team to review each detail with the committee at its (monthly) meetings, until they fully understand it, then approve it. This severely hobbles forward progress, needless to say. It occurs when there is weak or no trust of the project manager and the team. The trust issue should be addressed head-on, here as in all other circumstances where it occurs and changes made so trust can function – this is the only way true, effective delegation can occur.

- **"No-shows"** – key managers may miss meetings delaying key decisions, or producing "I'm not involved" attitudes in the missing manager's mind. It can also foster an attitude of "abdication" instead of delegation.

- Single Senior Executive Responsibility – if the single executive is the CEO or President, this can work, unless his/her availability and access is very limited, typically the case. More likely, if the CEO says "I'll manage this myself" it is a case of inability to delegate, which of course severely hampers the project. Alternatively, if another senior executive assumes this role, it CAN be very effective. While there are some great exceptions, generally this should not be the CFO, unless the person in this role is unusually operationally oriented. Otherwise its like having the CFO have reporting responsibility for all of IT. The financial function, in too many of these cases, ends up with everything it needs, while the operational functions wait, or worse, are starved of budget and leadership. When in doubt, remember the objective of the project – to improve (operational) performance of the business.

Best Practice Summary – in the area of project leadership interfacing with the top management of the company, as we've seen, the mechanism or process used is not the major factor – it is how it is run or used that determines success. Some guidelines for a Best Practice executive interface:

- **Keep the goal in mind** – The goal here is a fast-moving, low-cost process that transitions the company from operating with its current systems and processes, to a new one. Everything that aids this process adds value to it, and activities and actions that do not subtract or are just waste – expensive waste. A Best Practice, of course, is to constantly strive to improve this, as with other processes by eliminating waste.

- **Summary level reporting only** – one big time/cost waster is elaborate PowerPoint presentations and reports, which don't add value. One of our favorite report Best Practices is that of a major global corporation. You are limited to one 11 x 17 piece of paper, both sides, but can do anything you want with it., which forces conciseness and delegation of details. This is similar to the "stand-up meeting" concept – everyone remains standing a) to stay awake, and b) so attention remains focused on the vital few points, skipping the trivial many. Technique – "If I only had 10 minutes for this meeting, what would I focus on?"

- **Frequent is better** – more frequent, short, informal, summary level meetings which focus on unresolved issues that need top management involvement, budget, time-line, schedule, or resource issues. This allows the project to move fast, change plans and directions quickly, without having get bogged down in the "why did the plan change?" kind of discussion, a waste of time. It changed because the team learned something important and adapted, rather than force-fit a previous plan to avoid looking "bad." With many short meetings, Internet meeting technology can be used, so everyone can attend, even if off site. However, we've seen hazards with these – people "think" they are paying attention to the meeting, but really they are checking their email, working on a report, spreadsheet, or making coffee. In-person is best for this reason.

- **Confront CYA head-on** – inherent in this kind of reporting structure and project is the desire to look good, look like you knew what was going on from the start, etc. The evidence that CYA forces are operating is when one sees an expansion of presentation materials, reports, minutes of meetings, emails and memos to "document" discussions and decisions, multi-media presentations, and other time-consuming items that do not move the project ahead.

  Remember – once the project is done, no one will care in the least who said what at a meeting, or what the basis for a minor decision as – only that it was successful, and if/where it is not, what is underway *now* to correct it.

  When you find this going on, drag it and whatever "sacred cows" are involved out into the open, shine light on it with candid, honest discussion, then provide leadership and support to re-establish trust. If project leadership has proven weak, consider adjustments and re-organizing of responsibilities as far as possible before replacing the project leader. We have seen fairly mediocre, somewhat weak project

leaders succeed when they are well supported and have a good team. One project manager we worked with, many years ago, even had a serious drinking problem, yet the project succeeded in a major way.

Selecting Team Resources

Making a large change in a company's system and its operations is a very big effort and requires major, dedication of resources. This is perhaps the most difficult aspect of all because, depending on how it is done, the resulting team will have more, or less, ownership of the whole endeavor.

Mistakes – **how NOT** to create an implementation team:

- **Have an external project manager** – assign project management to a person who is an outsider, not in any way a part of the company's success, failures, or culture. He/she will be an "expert" in a mysterious, dangerous process, but if/when it crashes, will be long gone.

- **Depend heavily on external skills and resources** - hire temps, consultants, people hired only for the project. This will make the internal people feel completely incapable of performing on their own, and thus remove ownership from it. Almost all huge implementation failures have this element in common.

- **Reassign key internal people full time to the team** – remove them from their daily jobs and responsibilities. This way, they cannot fully own the resulting success or evaluate risks. They will now be in "their own little world." Meanwhile, life moves on in their former departments, new political alliances are formed, new in/out groups, and new "secret handshakes" created. They must "sell" everything they do to those still in their old departments and work groups. Challenges, high potential for difficulties and failure are virtually assured.

- **Assign expendable people to the team** – when department managers are asked to select people for implementation teams, it is VERY hard for them to select their best people – or even harder, to take the responsibility on themselves; they just feel way too overwhelmed. Further, they depend on their best people to keep things together, working well – vital for their own performance reviews, raises, etc. So, the "weakest link" is often selected. Once again, challenges, high probability for difficulties or failure are virtually assured.

- **Create a large team** - with many people on the team, they'll have to spend a lot of time in meetings, communicating with each other, resolving disagreements, etc. This dramatically increases project overhead, adds confusion, decreases individual ownership. Once again…. You can see where this leads – once again.

- **Make a long schedule** – allowing a long time for the team to prepare, convert and Go-Live greatly adds to the number of meetings, CYA projects, and changes in team members, none of which actually moves the implementation forward. When new people join the team, they have to "get up to speed" – all extra work, with no added value on the actual project itself.   With a long project, the percentage of time devoted to status reporting, meetings, communications, reporting to top management, collaborative sessions with work groups, changes in business processes and strategy – all dramatically increase, thus "once again…. – increasing the probability of difficulties or failure. A long schedule may appear counter-intuitive, but it is a fact. A multi-year implementation project is almost assured of never succeeding fully, simply because of leadership changes, both within the company, and on the team alone.

This depressing "checklist" is included here, in an otherwise positive-oriented set of guidelines specifically because we, and others, have so frequently seen them in actual practice. Although it is widely known that implementation projects are risky, what is NOT so widely discussed are the causes of the risks. We've just covered some of the major ones – where problems or failure were almost built-in from the start.

To take an example – sky-diving – the act of jumping out of a perfectly good airplane couple of miles above the earth's surface, would appear to be highly risky, and it *is, if you aren't prepared.* Just "going for it," in this situation can and has resulted in a greatly shortened life span. Similarly, in a complex business change, i.e., software implementation, rigorous planning, preparation, education and training virtually eliminate risks, just as it does in sky-diving. And high blood levels of testosterone won't bridge the gap.

Best Practice implementation team formation -Now that we've discussed how NOT to create an implementation team, let's flip things around to some clear guidelines on Best Practices for assembling effective, successful, low-cost implementation teams and projects:

- **Strong, internal project leader** – and we mean leaders, not "managers" – a key hands-on executive or relatively senior manager (not the IT

manager) should take this role – he/she will be a powerful force for ownership. Here's how to keep from overwhelming this person:

- **Add project administration** – provide a full-time project administrative assistant to the leader – most of the project management work can be handled by a capable assistant. The most time intensive part is gathering status information, preparing reports, presentations.

- **Add key role deputy** – assign a capable deputy, a fully-capable "stand-in" who can, if/when needed for the functional manager serving in the project leader role. This can and will off-load the leader, so he can have enough time to effectively lead the implementation project, while remaining effective in his / her primary functional role – essential for full ownership.

  We have found there is frequently a lot of confusion over the roles of project leadership, management and administration. Leadership is clearly the most powerful and critical, yet most of the time for getting a project to move forward is devoted to administrative work.

  Keeping the project leadership securely in his/her power base of a key line management role insures that reality is integral to the change/implementation process and keeps ownership solidly in place as well. The Best Practice here is to select a real, effective leader, keep that person in their primary job, while providing supplementary support to back-fill the person in their primary leadership role, while off-loading as much of the administrative work as possible.

  This strategy allows the project leader to truly be physically and emotionally able to continue to provide leadership in the primary business role, yet also effectively lead the change process for the company, including his/her own work area as well at the same time.

- **Strong, functional managers as team members** – everywhere possible, assign a strong manager for team membership, one who exhibits real *leadership* characteristics, more than just someone who really knows the functional area. Follow the same guidelines described above for insuring that these people have enough time to effectively carry the dual responsibilities of their functional management role plus the implementation project role. Off-load and support them in their regular job role to allow quality, effective time for the implementation project.

- **Keep the team small** – a highly focused, tight, small team of intensely motivated people who really know what they want to accomplish, will move mountains, quickly to get it done. Communication lines will, on a small, tight team, be short, concise, and trust-filled.

- **Continuity** – ideally, the implementation is the same team that performed the "as-is" and "to-be" business process analysis, and which thoroughly understands the business strategy and its critical success factors.

Successful implementations, especially those that not only just don't fail, but hit, or exceed their benefit targets and expectations, typically have most of these elements in the implementation project team. And here, we have identified the reason *why* this is so – effective ownership that goes beyond just being motivated, to being a process owner plus a stakeholder with some degree of power in the day-to-day business operations – critical to making successful changes in those operations.

Having a team that has more outside membership seriously dilutes this effective ownership.

Success Example - One company, a $ 50 million/year high tech manufacturing company, we are familiar with was unable to utilize much of its implementation consulting budget that it had planned. The company is highly customer focused, with many short-notice on-site visits by key customers. Consulting resources from the software company had to be scheduled in advance, and frequently were cancelled at the last minute, or went under-utilized while they were on-site.

This forced the management to "do it themselves" – using webinar and conference calls to tap into outside expertise just for educational purposes, so they could learn what was needed. Since they were working nights and weekends, the *really wanted* to get it done soon, yet since the team was entirely composed of key line managers, making sure it went well was critical.

As a result, all of the planned functions in the new system went into live use only a few months after starting, with only a small portion of the external consulting support that had been planned being utilized.

This simple example illustrates the key points involved in Best Practice implementations – all centered around maintaining effective ownership of the before and To-Be processes, and all steps between the two.

Education and Training

The first step in this area is to clearly separate education from training, as we do in other chapters in this book. Briefly, in the context of implementation, the purpose of education includes:

- **New concepts** – these are underlying thought processes, and assumed understanding that is embedded in the Best Practices integral to the new software.

  Example – many problems associated with implementations of material planning (MRP) functions stem from the fact that those using it have not been adequately educated in MRP concepts. Effectively using software delivering MRP capabilities has a poor chance of succeeding if the users are blindly clicking on buttons and following rote procedures. A person who truly understands the concepts involved with a particular software function can almost figure out how the software does it for one's self.

- **Precedes and informs detailed planning** – when project planning is done by people who truly understand the concepts behind the business processes and more effective work flows the new software can bring plan the steps from "here" to "there" there is always a shorter, direct path to the objective than will otherwise occur.

- **Speeds up detailed, hands-on training** – As was just mentioned, the actual amount of detailed, hands-on training needed to become proficient with the software is a small fraction of that required to "teach" rote-style, how a person is to do their job with the new system. We have observed people like this taking notes that say "hit the down arrow 3 times, then press Enter…" and the like. Frightening, from a management point of view! As MRP legend, George Plossl said many years ago "If you think education is expensive, try ignorance!"

Ideally, education of the core project team precedes the business process analysis and software selection process discussed in the preceding chapters. If it has, so much the better. If not, start now. In any case, though, the education process should be expanded to include others in the company who will be using or otherwise involved in the system.

Example – one of the most successful large-scale ERP implementation projects we personally were involved with utilized a 20-course education and training curriculum that provided appropriate concept-level education and hands-on training to over 1,500 of the company's 2,000 employees. Even

though there were a number of very challenging issues, and the software did not provide all of the functions needed at the time, nevertheless the Go-Live event and subsequent experience was the best kind – uneventful and smooth.

Dress Rehearsals – the Conference Room Pilot

In any event or occasion where things have to "go right" on the first day, not only planning, but repeated rehearsals, practice and training are absolute requirements. Smart theater producers know this and make *sure* that, after the cast has been selected, and the production planned in detail, that there is enough time to practice, to rehearse the play, so that on opening night, when *everyone* is watching, things go smoothly. In similar fashion, the wise sky-diver will train, prepare, and practice thoroughly before stepping out of that door in the perfectly good airplane at 10,000 feet.

We have wondered for years why this kind of analogy seems often lost on software implementation planners, who sometimes treat preparations with low-level contempt – as though everyone should just "know how to do their jobs" with the new tools "auto-magically."

The Conference Room Pilot (CRP) has been used for decades by wise implementation leaders to educate, train, plan, rehearse and otherwise make *sure* that the Go-Live event and transition is smooth. To clarify, other discussions of the CRP often include the business process analysis and the as-is and to-be work described in the preceding chapters. Here, we assume these tasks have been properly completed and focus on the rehearsal aspects.

The major phases of a Best Process CRP include:

1. Establish, setup and organize the facility itself – where the team will work and accomplish its tasks.

2. Implementation preparation and planning – validating the system, working through the "to-be" steps with the new software.

3. Data conversion design and validation – extraction of data from the existing system, converting/processing it, and adding it to the new system's data base.

4. Go-live preparation and training – this occurs after the core implementation team completely understands how, in detail, the "to-be" business process and software environment will work. Then, spreading the work to others can be done effectively

Phase 1 - establishing and setting up the CRP facility, the key steps and aspects are:

- **Dedicated, adequate work space** – a Best Practice CRP requires a separate, adequately equipped work space for the team to meet, work, and operate a test version of the system being implemented. Short-changing any of these seriously degrades the effort.. Adequate means:

  - **Computer resources** – either dedicated workstations, or where people can connect their personal laptops to the appropriate network, data base, etc. so that the system can be tested and exercised.

  - **White-board space** – preferably, the walls of the room are covered with write-on-able surfaces, e.g., white boards. These are essential communication and collaboration tools.

  - **Business Process Maps/Documents** – readily available and visible in the CRP work room should be at least high-level charts of the business process mapping and documentation process completed earlier, with emphasis on the "to-be" information, as this will be repeatedly used through the entire CRP process to guide decisions, serve as references and to keep the team on track.

  - **Food, drink** – it is helpful if the team can keep working in the room, to make the most of the too-short (it always seems) time available, something enabled by water, coffee ("business rocket fuel"), and healthful snacks, so people can keep focused. We also recommend bringing in meals for the same reason. Restrooms should be nearby also. Help everyone make the most of the work sessions.

  - **Chairs** – this is not the room for the 30-minute meeting folding chairs. Keep the seating from becoming a distraction or a business source for the local chiropractor.

  - **Room for everyone** – the entire team should be able to sit down and work effectively at the same time, not sharing a workstation.

- **Scheduled times** – the work sessions for the CRP should be regularly scheduled, ideally during times of the day when the participants are not regularly interrupted for operational decisions and actions, other meetings, etc.

- **Dedicated system and data base** – the team will practice, test, alter the configuration, enter data, and other activities during the CRP. The

system used for this should be separate from the live system, perhaps even on its own network, to keep response times short. We do not generally recommend using a vendor's remote training or demonstration system for the CRP – sometimes done.

There are too many differences, data issues, and other factors that can seriously reduce the effectiveness, relevance and efficiency of the CRP. If such a setup can be uniquely configured to work like a live system for the company, it may work well. Planners need to make sure that all configuration settings and data can be modified, updated, changed, restored to a previous setting, and other aspects needed for effective testing, planning can be performed, regardless of where the data base and servers are physically located.

Phase 2 – Implementation planning and preparation - Once the team has been assembled, and the CRP workspace and IT work completed, the actual CRP process itself, the real work, can begin. A Best Practice CRP process includes, at a minimum, these steps:

- **First –software validation** – once the CRP space has been set up and is operational, the first task is to establish a software configuration baseline, so you 'know what you got." Create a test scenario that, using sample data, allows the team, hands-on, themselves, to step through a basic business cycle and process so the team is assured that it is working as expected, and/or any areas where this is not the case, have been identified and documented. A vendor support consultant may be helpful at this stage – conducting, in effect, a preliminary training session. It is vital that the team do the work, not just sit back and watch another demo. Use the "to-be" business process maps and documents as guidelines.

- **Repeat until stable** – if bugs or missing, but expected functions are identified, unless they are minor, repeat the validation process until the software is stable and the team understands how it work, and that it DOES work as intended.

- **Initial data conversion** – we prefer doing an initial data conversion, using partial or subsets of live, real data to create a set of hand-match data that everyone already understands. IT professionals will sometimes claim to be able to convert any data but it just isn't true. Why? The "same" data in two different systems may be handled differently, giving it a different "meaning." By doing it by hand first, the team will completely understand these often subtle differences. We have never found an exception to this experience, often glossed over by non-users of the data, who in all honesty do not grasp these subtle differences, but that

can be critical in some cases. This deeper understanding can allow the team to help design and plan the conversion process – placing the ownership where it is most effective – with the team, not IT "experts."

- **Detailed to-be process validation and planning** – this is THE critical part of the CRP process. For this the team steps through every part of the to-be business processes, identifying every issue, policy level, procedural, work flow, and anything that affects who owns what parts of the process.

  Example: The As-Is business process calls for new customer accounts to have their data gathered by a customer service rep, written up, then forwarded to someone in the credit department who then approves it and enters the new account in the system.

  The To-Be process allows the customers to open a new account by themselves via an Internet accessible web site/page. Who is responsible for credit approvals? The CRP process must identify and resolve this question.

- **Software configuration established and tested** – a Best Practice CRP does not involve setting up new software configurations, then running them, then repeating the process over and over. The software vendor should be able to furnish a pre-configured version of his software that is close to what your team's To-Be vision is. Install and test this one, making minor adjustments if/as needed. We are familiar with too many projects where the entire original implementation budget was consumed and over run by this activity alone – months and months of changing the configuration, then running through the entire business scenario / process again. Exhausting, demoralizing, discouraging, self-defeating, expensive – otherwise just fine…

- **Outputs and results** – the detailed To-Be process validation and planning work must generate the following outputs or results:

  - **Training materials and plans** – how to adequately train everyone so the Go-Live event and subsequent business operations will be smooth and uneventful, free from lingering unresolved problems. This includes who the instructors will be, what training sessions will be required, when, and who the participants will be. The materials, of course, must be company-specific to be effective, although they can be adapted from vendor-furnished material to some extent.

- **To-Be Adjustments -** to the To-Be process maps and documentation where needed. Some of these will be improvements, and some may not. Much is learned during this work, and changes to original visions, plans and understandings are inevitable, even desirable.

- **Validated software configuration** – the team will be know what the software does, how it does it, and be able to say with confidence how the business will succeed using the new software's capabilities. This is a critical aspect to eliminating the risk of the project. It is NOT a technical issue, although often complex settings of parameters and table data that control the software's function. Large ERP systems can have thousands of settings, hidden away, but controlling functions anyway. The team does not have to understand how these work – how could they? Typically the vendor does not have anyone who understands them all either; no human being could – just make sure the configuration you have works in a way you understand and control.

- **Tested, validated data conversion plan** – data conversion, both software performed and manually entered, require multiple iterations to overcome a wide variety of often-subtle challenges. The only way to prevent the wrong kind of surprise after Go-Live day is to prepare, run / perform the process, test and validate the results until it works as well as possible. This is another critical risk-elimination step, often the source of serious problems after Go-Live but fortunately completely avoidable. Key aspects of data conversions to bear in mind:

- **Software performed** – required for all large volumes of data; careful design of the process (there are excellent conversion mapping tools available, but apply thought carefully to their use), then test, re-test, and re-re-test the results by using the data in simulated live use. The key – can the to-be work flows and business process steps be correctly, reliably performed with the converted data? Data conversion software *does not know the answer to this question, nor can it.*

- **Manually performed** – this is data that is printed out, possibly adjusted in some way and then entered by direct data entry process into the new system. We recommend this method everywhere possible, because problems stemming from these data are rare. Even with sizeable volumes it is possible to hire temporary help or otherwise "crunch" the data into the new system in a short time period.

- **Go-Live Cutoffs** - These are always difficult especially for 24/7, global businesses. However, careful, thoughtful planning will usually enable the implementation team to separate truly high-volume, transaction data from more permanent data so absolutely everything does not have to be converted a few hours before the Go-Live event. Some ways that help:

  a) Convert through a date before Go-Live ahead of time. This means only the last few days of transaction data has to be converted at the last moment.

  b) Stratify by degree of activity. Some groups, contracts, products, services, etc., are inevitably less volatile than others. By converting the slower-moving data ahead of time, the last-minute conversion will be shorter and quicker.

- **Volume testing** – the software has been tested under high-volume circumstances, so possible response time problems have been flushed out, fixed and will not cause people in the company to be unable to do their jobs because the system has gotten bogged down. This has been a cause of a number of serious implementation shortfalls and difficulties and must not be neglected, even though it is often hard to perform.

- **Post-Go-Live Transition Plan** – this includes an on-going education and training process, so that both those people already working with the new system can have their training refreshed if/when needed, as well as new people who join the company later. It also includes a quick-reaction issue identification and resolution process that starts the day of the Go-Live event, and continues until it is "out of business" – i.e., life has become "normal" under the new system.

- **Fully revised documentation** – this means that the new work flows, new processes under the new system are reflected in updated, accurate policies, procedures, audits, and other working documentation that is the "DNA" of the company – its institutional memory.   It is this updated documentation that becomes the new process baseline from which the continuous improvements of the future will be started.

Implementation Recap

Of all of the myriad "things that can go wrong" ("Murphy" does live in implementation projects), in our experience, the must important, yet most

subtle, thread that runs through all of the aspects is that of making sure that ownership of the process is supported, led, and maintained throughout is by far the most critical.

It is the loss of the feeling of ownership by key process owners on the implementation team that is, in actuality, the most common cause of difficulties, shortfalls and outright failures. People find themselves going through the motions, but no longer caring a lot about the outcomes.   Lots of bucks get passed, like a game of hot-potato.

If the ownership is strongly felt, the team members will literally be thinking about the project when they are "not working." This energy and clear focus on getting it right will ensure that serious technical errors are not made, and will cause people to recognize when they don't know something important, and to get some help with it.

The key elements of a low or virtually zero-risk, yet major, enterprise software implementation project, once ownership is established and maintained, are:

- Detailed before and after business process understanding. The team has addressed and resolved all important issues and *knows* how the business will operate under the new system and its processes.

- Landmine clearing / elimination – the team candidly identifies potential hazards or barriers to a successful, smooth transition and has eliminated them.

- Top management fully supports and shows effective leadership where needed for the project.

- An effective interface between the team and top management has been in place from the start.

- Education and training – the team has identified and completes education for everyone in every position that could cause difficulties if left uneducated or poorly trained.

- New DNA – new processes are adequately documented so that continuous improvements on them can be effectively done into the future.