DEMYSTIFYING IT™

Common Sense Approaches for IT Excellence

Melissa Liu

Version Descriptor:	
File:	DMIT_v0.1_04Sep07.doc
Last Updated:	3/26/2014 10:35 AM

Table of Contents

ACKNOWLEDGEMENTS

Note to reviewers:

You may use the following as a guide regarding what I hope to accomplish from your review. Feel free, however, to go way beyond the goals I list for you!

I am deeply in debt to the following individuals for their assistance and support in the creation of this book:

Linda Liu, for helping me navigate the intellectual property aspect of creating and publishing a book – as well as gathering tips from respected, published authors as to how a first-time author might get published

Jane Phillips, for bringing her many years of technical publications experience to bear and letting me know in no uncertain terms where I have been unclear, disorganized, or verbose

Andy Hewett, for sharing his vast wealth of IT experience and letting me know where I have left off possibilities, drawn too few or too many conclusions, over-simplified or over-complicated, or just plain screwed up

Todd Ackerman for providing guidance, giving me the benefit of his experience, and connecting me with people who might add useful examples, additional approaches, and wisdom

Dave Breda for general support as well as specific feedback on and ideas for the financial management of IT

PRTM as an organization, for giving me the initial opportunity to become a management consultant, grooming me to be a highly competent one, and continuing to offer opportunities as they arise

<whoever the editor ends up being> for taking this newbie author into the new frontier of book publishing and helping bring this book to the world

<others TBD…>

Thanks to my children for inspiring me to want to be the best that I can be, so that they can learn to aim high, take risks, but above all live your *whole* life well, balancing the many roles that we all have to juggle, and that divide all of our time

And, finally, thanks most of all to my husband, for supporting me throughout the crazy twists and turns of my professional life, and having (or faking!) confidence that I will be wildly successful in all of it

PREFACE

I have spent the last eight years in the management consulting field, primarily focused on helping companies implement and improve information technology (IT). It is obvious to me that U.S. companies are nowhere near the realm of possibility when it comes to IT effectiveness and efficiency. We are wasting money in our IT spending, and we are under-delivering to IT's promise of improving business efficiency and bottom line results. My experience in IT-focused management consulting has shown me a number of areas where we can improve the ways we manage IT.

The world of IT has long underwhelmed the business community. Business folks complain that IT projects take too long, cost too much, and do not achieve the intended business goals[1]. The literature is full of examples and statistics showing that IT projects frequently fail in some way: they are late, over budget, or cancelled. When implemented, they do not achieve the initial objectives that justified their expenditures.

Companies spend millions of dollars on IT – but companies frequently take on too many projects or the wrong kinds of projects. Management teams may not even be aware of all the projects and technologies that exist in the company, resulting in significant waste of IT dollars. IT spending may not be well correlated with business strategy or even technical strategy. Or there may not even be a well-thought out IT strategy – and *one cannot implement a strategy that does not exist*. Alternately, or in addition, the IT project pipeline may be cluttered with non-strategic projects that distract attention and resources from the projects that actually are strategic and are business-critical. [even even even]

Sometimes, the management team has chosen the right projects, for all the right reasons, only to have them fall down in implementation.

Sometimes IT projects fail because, once management decision makers decide to fund a project, their attention goes elsewhere while they assume that the worker bees will properly implement the system – whereas continued management involvement, strong sponsorship, formal decision processes, and adequate resourcing are among the key success factors for IT projects. When management teams turn their eyes away from the implementation work after the teams are off and running [rearrange], business goals and success metrics initially defined by management decision-makers may not be fully translated into the implementation details that are worked out at lower levels of the

[1] Source to be dug up.

organization. [they don't see the picture you have. Natural for people down below to think it's no longer that important. Keep the visibility]

Communication failures are common between business folks and technical folks. Business folks complain that IT does not understand or respond adequately to their needs. Indeed, the technical community *does* frequently lack a deep understanding of the business environment, processes, and constraints that are relevant to the systems they implement or maintain. IT folks may not have any real sense of what a "day in the life" of their users looks like. Likewise, business folks may not be conversant enough with technical concepts to be able to understand or communicate in the technical language their IT counterparts speak. They may not grasp the business implications of possible technical decisions and courses of action, if they are expressed in jargon and techno-babble. Sometimes a "translator" type of analyst role is used; however, there are few good translators that are fully able to ensure that business needs get fully instantiated into technical solutions. Even translators frequently end up more conversant on either the technical or business side of the equation.

Efficiency shortfalls also result when IT is used to "bake" poor processes into a system, rather than using implementation of IT as an opportunity to streamline operations and processes. The processes used to operate the business drive business efficiency. IT can help by making streamlined processes very efficient and by enabling very rapid decision making on high-quality information. Or, IT can be used to hard code a bad, inefficient process into a system, which makes it then even harder to make improvements to those operations. IT implementations that focus on the technology implementation to the exclusion of the underlying business processes are destined to under-perform their potential.

Effectiveness in IT is the ability to choose the right goals and implement them, such as ensuring that IT work supports overall business strategy and goals. *Efficiency* is the ability to do so with minimal waste, such as cancelled projects, wasted resources, time, or money, and so forth. Effectiveness can be thought of as "do the right things" versus efficiency as "do the things right."

Effectiveness and efficiency are sorely lacking in business IT today. This book was written to provide a framework to understand IT within its full business context in a holistic way, in a style intended to be both comprehensive and readable. All corporate personnel, from the line staff in business functions, to IT technical staff, to the upper echelons of the management team, benefit from a deep understanding of IT in the business context. Best possible IT effectiveness and efficiency requires comprehensive collaboration across all functions and all levels.

Effective and efficient use of IT requires: choosing the right projects to fit business and technical strategies; ensuring projects are set up for success with the right sponsorship, financial support, management and decision processes, and team; and ensuring early and deep collaboration between business and technical folks.

I wrote this book in an attempt to raise the level of IT expertise and knowledge across the entire high-tech business community – from IT and business line staff to executive level IT and business managers. While this may be a lofty goal, it is also a critical one to enable us to collectively improve the competitiveness of our businesses in the global marketplace. I believe this book presents content for all of these audience types, and I have attempted to organize it so that it can be easily navigated for each's specific needs. I have also attempted to write this book so that it can be easily read by all parties who could most benefit from its contents. I have attempted to use plain English wherever possible – using jargon only when needed, and then defining it in each instance. I hope that I have succeeded in this attempt, and I am also hopeful that this book can achieve the lofty goal I have set out for it.

Enjoy the read,
Melissa Phillips Liu

FOCUS OF THE BOOK

First off, this book is focused on *business* use of IT – it will not tell you how to navigate social networking sites, configure your email, or upload video to YouTube. It can inform you, however, as to the ways that companies like LinkedIn, Microsoft, or YouTube might use information technology to run their businesses – clearly a critical need for them, as they do much of their business in the electronic world.

Secondly, there is no one *right* way to do any particular thing in the IT world[2]. Anyone who pretends to have the single answer that always works is probably just trying to sell you something. Accordingly, this book will not hand you the recipe for perfect IT outcomes, every time. The intent of the book is to convey the *concepts* that, when applied appropriately for a given set of circumstances, will lead to optimal results. With a deep understanding of the key concepts you are best equipped to figure out the right way to tackle any given problem. This approach is analogous to the way that you might approach college math classes. If you focus on memorizing the formulas, you may have trouble when a problem is set up in a different way than the formula-driven mechanics you practiced in studying for the test. If you go deeper to memorize the *concepts* – and forget the formulas – not only can you solve challenging, differently-presented problems, you can always re-derive the formulas on the fly, based on your deep understanding of the underlying concepts.[3]

While explaining the concepts, I will explain different approaches that are commonly used to solve any particular problem, and their benefits and pitfalls, as well as when each type of approach might be most valuable. Whereas there is no single right method for any

[2] This is the reason that "It depends" is a common phrase to hear coming out of a consultant's mouth.
[3] On the other hand, if you were a B student – or worse – in math, a formulaic approach might explain the shortfall, rather than any lack of capability. I'm pretty sure anyone can do math – most of the people I have seen that are "bad at math" are more just intimidated by it than incapable of it, and tend to either sabotage their own thought processes with insecurity, or jump right to guesswork rather than working through the thought process.

given problem, there are in some cases common *wrong* methods – in that case, they will be mentioned, along with the problems inherent in those approaches.

Finally, I have attempted to make the presentation of the concepts as straightforward as possible. For this reason, the language is fairly informal – and most likely is downright "un-consultant-y". While I recognize that bleeding-edge jargon sounds considerably more impressive, I have chosen to emphasize readability over formality. So, perhaps these concepts will end up sounding like nothing more than common sense – because, often enough, *they are*. The trick is the implementation of the concepts, not the concepts themselves.

The next section provides some context to understand the overall environment. If it is too basic for you – skim it and move on. It may be useful for others, and I would rather not assume knowledge that may or may not be present.

SOME CONTEXT TO SET THE STAGE…

IT can be intimidating, because of the extensive technology lingo used to describe it. A complete appendix is dedicated to helping less technically savvy people navigate some key topics in IT business systems. Fundamentally, IT is used to make business operations efficient and effective, so if you want to understand IT, you need to understand the structure of the business you are examining.

Before you skim over that previous statement without thinking too much about it, consider this: One might infer from the statement above that a technical person that *only* understands the technology environment and concepts, rather than having a thorough grasp of their constituent business user's daily lives, *does not really understand IT*. This may be earth-shattering, as some of those very technically-focused IT people can be quite patronizing toward business staff, and may throw around technology jargon as an intimidation technique.

But, in my experience, the most technically-focused and most-likely-to-be-patronizing IT folks are the very same ones that are least likely to truly understand the business drivers and constraints. Technology for its own sake, isolated from the business needs driving its use, is meaningless – the *only point of IT is to drive tangible business results*.

So, by my definition, techno-snobs that try to out-jargon their constituent business users do not understand IT. (If they give you any trouble, you can tell them I said so.) Fortunately, the really good folks in IT do understand that it is all about business results. You want make sure you have lots of these types, and considerably fewer of the former.

The fundamental business structure (or operations framework, as it is sometimes called) is very different from industry to industry, as the nature of the product or service and the business environment (social, political, regulatory, competitive, and so forth) vary widely. A law firm is very different from a computer hardware manufacturer which is very different from a pharmaceutical manufacturer or a software company. An in-depth understanding of IT in each of those environments requires an understanding of each

one's business model as well as the information flows required to run the business. The industry case studies later will drill down into a small set of industries to demonstrate the nature of each business and the resulting impact on IT in that environment. However, some general information is provided in this section.

Some elements are common no matter what kind of company you are talking about. Companies provide goods or services, from which they derive income, and for which they charge fees.[4] Other organizations, individuals, or governments buy the goods and services and must pay for them. Providing goods or services costs money – floor space for manufacturing and inventory, wages for workers, manufacturing source materials, equipment (manufacturing, office, and other), office supplies, essential services, infrastructure, and so forth.

Regardless of the industry category, a significant amount of information is required to run the business. If the company is public, it must submit financial accounting information to the SEC. Even if it is not, the company must account for its operations in order to calculate and pay taxes. Companies must keep track of employees, their wages, and personnel history. They must keep track of who owes them money and how much, and track payment status. They must be able to purchase office supplies, equipment, and manufacturing source material, track to whom they owe money, and track payments. And, beyond that, they need to track all of the information needed for their industry- and company-specific requirements and processes.

The diagram below shows an overly simplistic, industry-independent business framework, highlighting some aspects of information that may need to be managed. The "Industry-specific Processes" will vary widely depending on company and industry, and in almost all cases would be sufficiently complex to be broken down into multiple process categories.

[4] There are other ways to derive primary income rather than direct fees, for example portals which provide information as a good, but derive income from selling advertising space on the portal.

	Financial / Accounting Processes	Human Resources Processes	Industry-specific Processes	Information Systems Processes
Objective(s):	Track and report money in/out and overall financial health	Manage employee information	Manage the industry- and company-specific operations	Manage systems used by the business
Example Processes:	• Maintain accounts payable and receivable • Provide management reports • Manage department budgets and costs • Support quarterly and annual reports and filings	• Maintain salary histories • Add employees • Change employee statuses (for example, termination) • Allocate wages and bonuses from pools of dollars • Manage employee ratings • Manage benefits	*Goods companies*[5]*:* • Create goods • Manage inventory • Find customers • Sell goods • Deliver goods *Services companies:* • Find customers • Sell services • Perform services • Bill for work	• Create accounts / logins • Manage access to devices and information • Remove accounts • Maintain systems • Implement new systems • Provide systems support

In any business, some information flows from one process area to another. As an example, consider just a few of the information flows needed for a manufacturing company that pays salespeople on a commission basis when a sale is made:

- The product must be manufactured or retrieved from inventory
- The product must be delivered to the customer
- In many situations, the shipping progress must be trackable while the product is in transit
- The customer must be billed
- The accounting system must reflect an amount owed in Accounts Payable
- The revenue earned must be accounted for, and
- The salesperson must be credited for the sale so that it is included in commission

In this example, this single event triggers events related accounting, billing, manufacturing, inventory management, logistics, and human resources process events.

The example above talks about information flowing between *processes* – it does not necessarily mean that information is flowing between information *systems*.

The fact that information must be managed does not mean that information technology is required to do so – or that all areas that *could* be automated through use of IT will be or should be. Implementation and maintenance of IT business systems can be costly. Some small companies can use mostly paper, manual processes: paper forms, with carbon copies if needed, filed in traditional manila folders in metal cabinets, faxed, or mailed as needed. For some small companies, email and word processing may be the bulk of the computer technology needed. However, a large Fortune 1000 company may have many complex, highly interfaced systems. *[Incl statistic here, e.g., # of systems in use at average F500]*

[5] Some "goods" companies may have limited to no inventory needs, for example a software company that sells downloadable software. And, some services companies may have substantial need for tangible assets (although not classified as inventory), for example a company which hosts business applications for other firms on their equipment

Regardless of size, each company must decide where and how the business should be supported by IT[6]. Companies may have a single core system (such as an ERP system), surrounded by a set of smaller, integrated point solutions. They may have disparate systems whereby a person must extract information from one system and enter it into another. A very small company may do all of above manually, using spreadsheets and paper processes. And, larger companies frequently have multiple "core" systems, integrated or cobbled together in a variety of ways[7].

Determining which process areas and specific business process events should be automated is one of the critical decisions an organization can make. But, before we get into the specifics that drive those decisions, we need to first discuss what IT can offer. Why do we use IT?

We use IT to improve the bottom line results – or profits – of the business. Period.

Calculating profits can be rather complex from an accounting perspective, but they boil down to revenues minus expenses, or your income minus your outgo. IT can improve your bottom line by increasing revenue, decreasing costs, or both. The list below demonstrates some possible improvements:

- IT can be used to manage large volumes of information that would have to be manually managed otherwise, which could reduce costs
- IT might enable new capabilities that are projected to provide value in new target markets, which could increase revenue
- IT can be used to decrease cycle times for business processes in an organization. For example, inventory management capabilities can reduce the amount of time it takes to find, manage, and account for inventory in stock
- IT can be used to increase information visibility within the organization, which can improve responsiveness and flexibility. This type of improvement could reduce costs or increase revenue. For example, identification of an upward trend in demand may drive you to increase your manufacturing and therefore your revenue – without that visibility, you may miss revenue opportunities and potentially lose market share to competitors. Likewise, a downward trend could drive a reduction in production levels, which can save on inventory costs for inventory that would have accumulated otherwise

Some of the factors used to determine which processes should be enabled and which should not include:

Driver	Rationale
Volume or complexity of	*Manual processes scale poorly to large*

[6] When a process is determined to benefit from IT support, we sometimes say that it is an *automated* process (or semi-automated, depending on degree), or we may say that it is *IT enabled*.

[7] A common problem in larger companies results from decentralized IT management, whereby there may be multiple systems for the same function across the company, implemented by different divisions or business units, and which may or may not be integrated. These companies may then have to coalesce information across all the systems into a single set of knowledge, for example for financial reporting.

information to be managed	*numbers of complex transactions*
Degree to which the information is time-critical or mission-critical	*If a process used information that must be kept current and acted upon quickly, manual data management will not support it well*
Degree to which key information must flow across process areas	*Areas of information flow between processes are good candidates for integrated systems, because they may require manual data entry or data movement otherwise*
Resources (time, money, people) that could be saved by efficiency gains available through automation	*Automation should be used where it can reduce costs in a meaningful way*
Degree to which automation may allow the company to do things it cannot currently do	*IT may enable a company to offer new capabilities, address new markets, or launch new businesses, thereby increasing revenue*

We call this subject matter "*information* technology" for a reason – the *information* is the key, and it comes from nature of the business and information flows inherent in that business. Technology is merely the enabler by which we improve the flow of business information. If technology were the sole critical part, we would just call it "technology!"

RELEVANCE AND PREVALENCE OF IT

The computerization of the business world over the last twenty-five years has made significant changes to the conduct of business. Work that was previously performed by people and supported by vast quantities of paper documentation can now be automated and centralized on computer systems. Companies in the US and around the world are increasingly dependent on information technology (IT) for basic business operations, and in some cases IT provides the company's competitive advantage.

Pervasive use of information technology has so transformed the business environment that it has had significant and measurable impacts on the overall macro economy. Former Federal Reserve Chairman Alan Greenspan credited information technology for increased productivity. Greenspan noted in a speech in 2002 that, "Over the past seven years, output per hour has been growing at an annual rate of more than 2-1/2 percent, on average, compared with a rate of roughly 1-1/2 percent during the preceding two decades." He went on to say that, "the pickup in productivity growth since 1995 largely reflects the ongoing incorporation of innovations in computing and communications technologies into the capital stock and business practices."

Greenspan also posited that increased use of technology has made changes in macroeconomic conditions ripple through the economy much more rapidly than was possible prior, because companies now have more visibility into up-to-date information driving their business, such as trends in sales, inventories, production costs, and so forth, and can adjust dynamically to changing conditions. He remarked at a 2001 presentation that:

> *The same forces that have been boosting growth in structural productivity seem also to have accelerated the process of cyclical adjustment. …New technologies for supply-chain management and flexible manufacturing imply that businesses can perceive imbalances in inventories at a very early stage--virtually in real time--and can cut production promptly in response to the developing signs of unintended inventory building. … In addition, the foreshortening of lead times on delivery of capital equipment, a result of information, and other newer technologies, has engendered a more rapid adjustment of capital goods production to shifts in demand that result from changes in firms' expectations of sales and profitability. … The result is not only a faster adjustment, but one that is potentially more synchronized, compressing changes into an even shorter time frame.*[8]

Information technology also drives a good portion of the U.S. labor market. Information technology comprised roughly 3 million jobs in the U.S. as of the May 2006 published Bureau of Labor Statistics report[9]. This number includes only the direct technical workforce – it excludes the large number of jobs in our information-based economy that use the information managed in IT systems as a core part of their job functions.

Now, discussions abound in the news and around the office water coolers regarding globalization, outsourcing, and off-shoring. We talk about a dark future for technology workers as jobs are increasingly moved abroad. We talk about having to raise H-1B visa caps to prevent jobs from leaving the country – even though that means those jobs would not be given to U.S. workers in any event. Technology workers tell me that they will discourage their children from pursuing technology careers, because there will not be any jobs for them – and they ask, what *will* our children do?

Regardless of one's place on the political spectrum or the degree to which one is personally affected by outsourcing, it is a reality. Companies outsourcing portions of their work has both benefits and pitfalls to the U.S. economy as a whole and direct impact to the US workforce. Our long-term competitiveness as a nation in a world market with shrinking borders is at risk. We need to consider what our role is in technology in the future, what it should be, and what actions we might need to take to make that future happen.

But, we have an opportunity to improve our competitiveness in the world market *now*, through improved effectiveness and efficiency in our use of information technology.

---

[8] Testimony of former Federal Reserve Chairman Alan Greenspan at the Federal Reserve Board's semiannual monetary policy report to Congress. This testimony was before the Committee on Financial Services at the U.S. House of Representatives on Feb. 28, 2001.

[9] This number includes BLS categories: "Computer and Information Systems Managers"; "Engineering Managers", "Computer and Information Scientists, Research"; "Computer programmers"; "Computer software engineers, applications"; "Computer software engineers, systems software"; "Computer support specialists"; "Computer systems analysts"; "Database administrators"; "Network and computer systems administrators"; "Network systems and data communications analysts"; and "Computer specialists, all other"

IT is relevant to members across the spectrum of our society. IT is relevant to government leaders and policy makers as we consider globalization and off-shoring and and as we weigh factors such as protection of U.S. jobs and workers versus free market economics and the ability of U.S. corporations to compete in the world market.

IT is relevant to the ways we educate our children. Should we – as some high technology leaders suggest – improve education in the math and sciences to increase and maintain our competitiveness in the world economy? Or do we – as some high tech workers suggest – drive our children toward any direction *except* technology, since all those jobs are going overseas or to lower cost U.S. immigrant workers on visas?

IT is relevant across levels and job functions within the business enterprise, as we realize that participation *across* the enterprise – up, down, and horizontally – is necessary to implement, manage, and use IT both effectively and efficiently.

Globalization is a reality regardless of the way we feel about it and the ways it may impact individual lives. The competitiveness of our country and the companies that employ our workers – indeed our very future – is at risk. Our use of IT can be vastly improved to increase the effectiveness and efficiency of our corporations, and their competitiveness in the world market.

HOW TO USE THIS BOOK

The book is organized into three main sections:

- Managing IT
- Industry Case Studies
- Appendices

The first section – "Managing IT" – describes the concepts required to effectively manage IT within a corporate setting. It covers the lifecycle of IT management activities from setting IT strategy to selecting and implementing business systems to managing IT operations. For example, it talks about deriving IT strategy from your business strategy, your technology strategy, and technology trends, and then using that strategy to drive where you invest your IT dollars – evaluating and selecting projects in support of your strategy. Determining where to best spend IT dollars – or even deciding how much money should be spent in IT – is complex. These decisions require careful planning and effective management of the IT spending portfolio. It talks about how to set up a business system implementation project for success: choosing the right team, selecting the right system, ensuring that the business processes are adequately considered, and putting the right implementation processes in place to ensure that the project meets stated goals. The section also covers management of IT operations, from assessing current state to defining needed improvements to best practices for support and maintenance.

The second section – "Industry Case Studies" – provides case studies for a small set of industries to deeply illustrate what the IT environment may look like for companies in those industries. For each industry, the discussion leads with the key concerns for that

industry driven by the business environment: regulatory requirements, financial reporting requirements, competitive landscape, and so forth. The specific environmental situation drives unique requirements for information in that environment – which correlates very strongly with systems required or frequently used. The operational framework for each industry is used to show the ways that systems may support its business operations – including which specific types of systems may be frequently in use.

The third section – "Appendices" – provides more details in order to fill knowledge gaps for specific audiences, or simply to supply more information. For example, there is a section that discusses fundamental IT concepts that may be helpful to less technology savvy business users. There is a section on finance fundamentals for IT managers, which talks about some key finance concepts that may be used within IT, such as ROI, NPV, payback period, and other valuation techniques that may be applied to value potential IT projects. There is also a section on broadly applicable analytical techniques, with information on the fitness for use of each one, and caveats. Finally, generally available appendix material can be found there as well, such as the index and sources for more information.

Those who are less familiar with IT may find it helpful to read the "IT Fundamentals for Business Users" appendix prior to starting the "Managing IT" section.

The "Managing IT" section is targeted for all audiences. Readers may then want to go directly to whichever industry case studies are of interest.

IT project managers and key business stakeholders for IT projects would both do well to read the "Business System Implementation" sub-section very closely.

I would recommend the "Finance Fundamentals for IT Managers" for any IT member (staff or management) who feels weak on that topic.

Business and IT managers responsible for IT funding and spending – or concerned about where and how IT dollars are currently being spent – might find the "IT Strategy" section helpful.

IT STRATEGY

Before we wander into the details of IT strategy, it is worthwhile to provide a definition of strategy. In my dictionary[10], the appropriate strategy definition includes:

> *A careful plan or method; the art of defining or employing plans or stratagems toward a goal*

Implicit in the definition of strategy is the specification of a goal or goals, as well as the plans to achieve them. Not all organizations have explicit IT strategies – some have an implicit strategy, and some have no strategy.

In some cases, the organization is small enough, and has capable enough leadership to be able to effect a strategy without explicitly defining it and sharing it with others in the organization. An implicit strategy is not the same as no strategy. But, implicit strategies do not scale to large, decentralized organizations – no coordinated plans can be effected if the distributed decision makers do not know what goals and plans are intended to be achieved.

Organizations without any coordinated IT strategy tend to have uncoordinated IT spending (duh!). The result can include missed opportunities to support business goals, project pipelines full of low-value-add projects, looming critical issues that do not get addressed, missed opportunities to exploit new technologies for competitive advantage, and late responses to changing technology and market conditions.

A strategy *sounds* like a good thing, but why is it really important?

WHY BOTHER WITH STRATEGY?

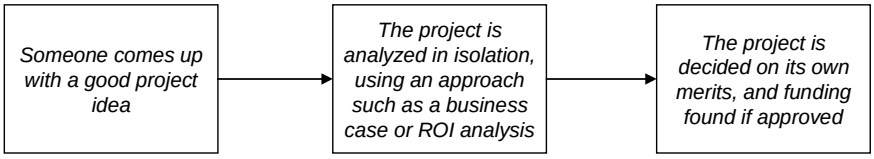
If you don't know what you are trying to accomplish in IT, than you cannot be assured that you are choosing best places to spend your IT dollars. Ensuring that you are spending IT dollars in the most effective and efficient ways requires that you have an explicit or implicit IT strategy – just as executing to a business strategy requires that you actually have a business strategy.

Ensuring best allocation of IT dollars requires asserting strategy and goals to be accomplished, ensuring strategy and goals are used to drive which projects are

[10] Webster's Universal Encyclopedic Dictionary, 2002

undertaken and which proportion of the IT dollars should go to each, and managing the portfolio of IT projects like the investment portfolio it is.

Chew on that a minute. I just made a bunch of statements about the decision process for IT spending without mentioning one word about ROI, business cases, payback periods, or NPV[11]. This view is fundamentally different than some companies' practices. I have seen a number of new project proposals that follow the following process:



So, what's wrong with that? Nothing really – it's a much better approach than *not* developing a business case for a project. But it ignores other places that you could possibly use that funding – the *opportunity cost* of the project, in common financial terms. A better way is to decide what you are trying to accomplish first, and then use the strategy and goals for ideation of possible projects.

If you do not have an IT strategy, then, upon what bases are you using to make project funding approval decisions? They "seem like a good ideas"? I have seen many nice-looking business case presentations to justify expenditures for a project. But, if a good business case is produced and reviewed in isolation, you do not know what other opportunity costs you may incur.

This is not to say that you can never have good ideas crop up after periodic business planning is completed; this approach merely states that these new, good ideas should be measured in the context of the IT strategy, rather than in isolation. I have suggested a periodic, flexible planning process toward the end of the strategy section.

We will continue the discussion of IT strategy in the next section. But we will first look at the key IT spending areas to set the context of the ways strategy influences spending.

CONTEXT SETTING – WHERE'S THE MONEY?

Strategy defines what you plan to accomplish, and funding should follow strategy. Before we go into this further, it is worthwhile to talk about the types of work that the IT dollars may be spent on. In addition, where the money *comes from* drives possible decision biases as to where the money will *go to*, so we will talk about some IT funding approaches as well.

[11] ROI: Return on Investment – A method for estimating an investment rate of return based on cost, value derived, and time period. Payback period: How long it would take for the value derived to equal the cost. Business case: A financial and qualitative analysis of a project's value. NPV: Net Present Value – a method to evaluate an investment that takes cost, value derived, and a given target return rate, and returns a result that, if positive, indicates that the investment meets the targeted return rate and may be accepted.

Where Do IT Dollars Go?

The main buckets of IT spending are shown below[12].

	Projects	Maintenance and Support	Management
Goal	Make changes to operating environment to add capabilities or increase efficiencies	Keep systems and infrastructure operational	Ensure optimal results from IT organization
Example Activities	• Upgrade to a new system version • Implement a new business system • Add functionality to an operational system • Decommission or replace a legacy system • Upgrade infrastructure (network, hardware, operation system, etc.)	• Perform planned maintenance functions (backup, patch application, etc. • Provide system support as needed based on user and system problem notifications • Provide help desk assistance to users	• Determine strategy, goals, key activities • Drive changes (projects, process changes, etc.) • Measure results • Staff organization to meet needs • Guide, manage, incent, reward IT staff

The determined IT strategy should have significant influence on the selection of Projects. We will talk about project selection and pipeline management in detail later.

The Maintenance and Support category is more likely to *influence* strategy than *be influenced by* it. For example, the following might drive IT strategy, and therefore planned projects:

- Noted instability in a system or set of systems may drive an organization to plan an upgrade or replacement
- A vendor's plan to obsolete a particular version of a system may lead to an upgrade project
- Shortcomings in responsiveness or time-to-resolution in the system Support processes may lead to projects intended to address the shortcomings[13]
- Problems supporting too wide a variety of in-place technologies may lead to clearer definition of which technologies are strategic and which have been "accidentally" proliferated
- Noted mis-alignments between the current capabilities of the support staff and either the installed technologies or the stated technology direction may lead to activities to address the issues

As the above indicates, an examination of current and projected issues (plus other factors) may lead to steps to address them in the IT strategy. (Derivation of IT strategy from current issues plus forecasted changes is discussed in an upcoming section.) And, the IT strategy is used – or should be used – as a decision context for IT spending decisions.

[12] Some organizations might split Maintenance and Support into two categories.
[13] Note that these projects could be process-only, if the shortcomings are in the support structure or processes, or could involve system implementation if capability gaps are causing issues

How Are IT Initiatives Funded?

Companies have different ways of funding IT work. A few of the different approaches are summarized below. There is no right way to fund IT – regardless of the overall philosophy chosen, organizations must be careful to avoid the potential pitfalls.

I typically like business-funded IT, because it seems more likely to ensure project work supports business needs in the way they need to be supported. IT-funded projects can be too detached from business users' experiences to achieve their potential in terms of functionality and usability. However, I found out at a recent client that business-funded IT, in certain circumstances, can result in some particularly thorny issues.

At my recent client, the business had *significant* control over the IT funding. However, "the business" was comprised of a number of disparate business stakeholders, each with their own "buckets of money" to spend. There was very little centralized decision making regarding IT projects, and as far as I could tell, very little coordinated strategy or effort outside of my own project. I observed the following results from this funding approach: 1) Too many funding sources, with too coordination or centralization of funding decision-making resulted in too many systems, overlapping systems, and too much effort (time, money, people) on non-strategic system work; 2) IT-driven changes – such as needed infrastructure improvements and fixes for ongoing problems – were getting neglected; and 3) IT was funded (and oriented) only to *react* to problems, rather than plan ahead to prevent future problems. So, as much as business-driven IT seems like it would lead more clearly to better business outcomes – in practice, it doesn't always. It depends.

Some benefits and pitfalls of a few approaches are shown below.

IT Funding Philosophy

Implications	**IT should be funded directly, and should work with business stakeholders to ensure needs are met**	**Business organizations should fund all IT, to ensure decisions make business sense**	**Business organizations should fund IT projects to ensure business value, but maintenance can be directly funded**
Benefits	• Single source of funding makes project decisions less complex, especially for initiatives that benefit multiple organizations • Direct IT funding may help ensure that infrastructure problems get addressed	• Ensures that IT spending aligns with direct business needs	• May result in more appropriate spending levels on infrastructure and maintenance, preventing longer-term issues resulting from neglect • Provides business more control over project selection and implementation
Pitfalls	• May result in overemphasis on technology versus business needs • May result in less business control over IT spending decisions • Does not guarantee that strategic projects are chosen over tactical or unimportant projects	• May shortchange necessary maintenance and infrastructure spending, which may result in longer-term stability issues – which do affect the business • Decentralized decision making by disparate business groups can lead to overlapping systems and too many technologies • Does not guarantee that strategic projects are chosen over tactical or unimportant projects	• Decentralized decision making by disparate business groups can lead to overlapping systems and too many technologies • Does not guarantee that strategic projects are chosen over tactical or unimportant projects

Note that this section has been somewhat simplified so far. For example, the business may decide a project is needed, and may grant the budget to IT to complete it. In this case, IT "holds the money" – but the funding decision was made by the business, which means that the funding power remains in the business. The key detail is where the funding *power* comes from, not necessarily where any particular dollar is currently held.

In addition, the funding situation can be fairly complex, unlike our simple one-business group, one-IT organization scenario so far. Large, complex companies with multiple business units (BUs) may have IT organizations within each BU, as well as a centralized IT organization providing infrastructure support. And, each IT group may have multiple groups of business user constituencies. The key is still to look at how the funding decisions are made, and by whom. How many wallets are involved, where are they, and who is putting money in the wallets?

One approach to consider, regardless of the IT funding approach used, is to form a cross-functional decision body for project decisions. A decision body with IT and key business stakeholders represented may be able to better ensure that business goals and needs are properly addressed, that infrastructure and maintenance needs are not neglected, and that IT dollars are applied to the most critical areas. It can address the pitfalls of all of the above approaches. Likewise, a consistent valuation approach, potentially with a role to support it, can ensure projects are valued in comparative ways, and that only the highest-value projects are funded.

One way to approach a discussion about whether the current organization, funding model, and decision processes are working well is to perform the assessment process discussed in the "Assessing the Effectiveness of Your IT organization" section. If not, an adjustment to one or more of the above might be needed.

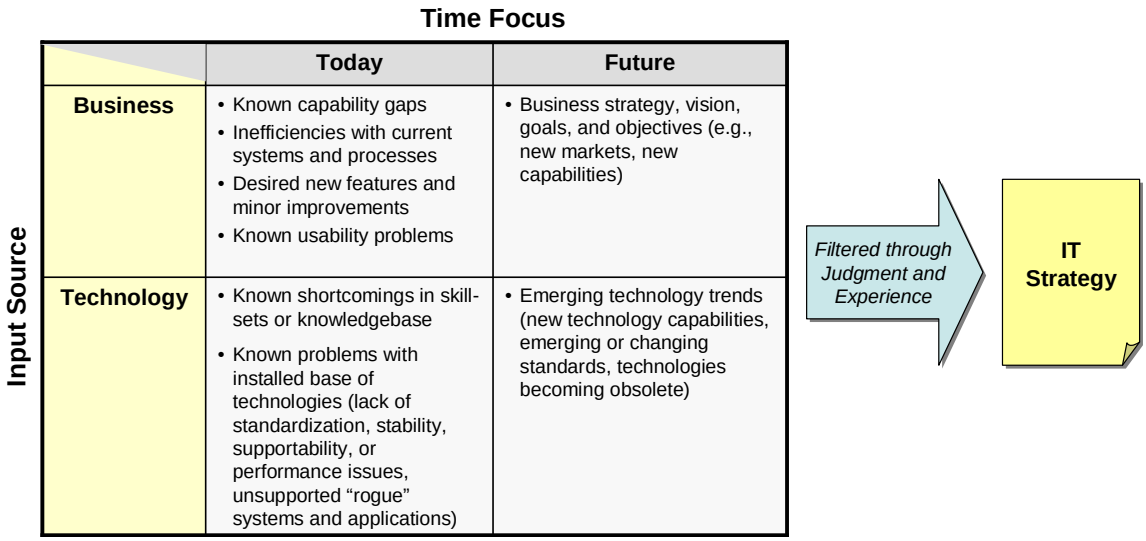
Next we turn our attention to the inputs to and derivation of IT strategy.

Deriving IT Strategy

As we have already stated, effective use of IT dollars requires an IT strategy that guides the possible uses of those dollars. Inputs to IT strategy include both business and technology drivers.

Business drivers include current issues, as well as forward-looking goals, objectives, and trends. Technology drivers include current technologies, skill-sets, and technical issues, as well as emerging technology capabilities and trends.

The inputs to IT strategy are shown in the diagram below.



In the short-term (some might say "tactical") focus, business users and stakeholders may identify shortcomings with existing systems. These could be in the form of lists of new feature requests for existing systems, new capabilities not addressed by any system, or other shortcomings in the current system environment provided to them. IT management may identify current issues related to, for example, too diverse a set of technologies for them to support effectively, misalignments between competencies and installed base, inability to meet support or maintenance requirements with current resources and processes, and so forth.

Longer-term planning identifies other desired changes. The business planning process may identify new system capabilities needed to address business goals. IT management should review the outputs of the business planning process to identify which business goals and objectives may have system implications and what those implications might be.

For example, consider the major shift a few years back to move increasing amounts of business online. As the business organization determines that they must deliver products or services in some new electronically-enabled way, the technology to do so must then be identified and brought to bear. This example is a very direct technical implication of business strategy. There are less obvious ones – for example, business leadership deciding that the management team must respond more quickly to changing conditions. Perhaps the organization is facing issues with timeliness and flexibility because they have systems issues that prevent them from getting information when they need it, where they need it, how they need it.

In addition, IT management must define or revise the technology strategy, which may be driven by the installed base and current competencies as well as technical trends, and their implications for future installed base and needed competencies. In this view, technology strategy is not the same as IT strategy. Technology strategy is an input into the overall IT strategy; the IT strategy is derived from both technology and business inputs. I suspect that some IT managers may stop at technology strategy and not get quite

to the overall business-focused IT strategy. In fact, if you look at journals aimed at IT managers, they tend to be quite technology focused and fairly ignorant about business perspectives. In my experience, CIOs are pretty business conversant, the IT Director level is variable, but the good Directors are fairly business oriented, and the IT Manager level tends to be more technology focused. These are very broad statements – at any level, there are of course exceptions – and, I have not done a wide, statistically-significant survey. Those in IT looking to move up in the organization should consider addressing any gaps they may have in complete understanding of their business environment.

But, back to technology strategy… In the business world, one type of strategic analysis is called a "SWOT analysis", for "Strengths, Weaknesses, Opportunities, Threats". The first two require looking inward at the organization to identify strengths and weaknesses. The last two require looking outward at emerging trends in the world to identify opportunities and threats.

Looking inward, an IT manager[14] may recognize that certain key technologies are prevalent and standard (e.g., their architecture may be predominantly Unix, IBM, or NT; their clients may be predominantly PCs, Linux, or Macintosh), and may decide that gradual migration toward the stated direction is needed.

Looking outward, the IT manager may identify that there is a new key technology emerging that is likely to become important to the business at a later time – potentially because it will offer new capabilities, lower maintenance costs, become a new prevailing standard – there are myriad reasons why a new technology might become important to an organization later. This information may result in a desire to experiment with the new technology or gradually migrate toward it. In addition, the IT manager may determine that some vendors in the installed base will be ending their support of particular system versions at defined end-dates, which will require the organization to define a project to update system versions.

In-place technologies and competencies and technical trends drive the technical strategy. The technical strategy can then be combined with the business strategy into a meaningful IT strategy – a unified vision of where IT in the organization needs to go and how it might get there.

We will examine the translation from vision to concrete plans in the next section.

Key thought before we move on to translating IT strategy into IT portfolio management:

> *Effective IT management requires an IT strategy, and it should be based on business and technology drivers – both strategic and tactical*

[14] "IT manager" is used loosely in this case; the role responsible for this work may vary, e.g., the CIO, a staff member under the supervision of the CIO, Director of IT, portfolio management role, etc.

DEFINING A PLAN OF ACTION TO ADDRESS THE STRATEGY

Strategy is generally defined as part of regular planning cycles, for example as part of the annual planning and budgeting process. The time horizon for the strategy could be the same as the frequency of the planning cycle, or could be longer. For example, a five-year strategic plan could be revised every year, as part of the annual planning activities. With the defined strategy – or as a response to the strategy – should be the actionable steps that will be taken in order to implement the strategy. The strategy and plans could be set at multiple levels: corporate, business unit, or group. In our case, we are talking specifically about the IT strategy, which as we said is derived from business strategy (and plans), plus technology drivers. The types of actions included on the plans could include things such as:

- Implement or upgrade a particular type of system to meet a business need or gap
- Add functionality to one or more business systems to add capabilities or address issues
- Improve IT operations to meet a business need or address a known gap between current performance and performance needed, such as improved support response and quicker time-to-resolution
- Add redundancy – such as an entire replicated operations site – to ensure continuity of operations in adverse situations (power failure, natural disasters, and so forth) to deliver a non-stop capability desired by customers or to prevent loss of revenue due to unplanned events

A plan to address identified actions would define *what* needs to be done and *when*. The when might be an absolute time, or might be based on a trigger (number of customers, business event, etc.). This plan (called an "IT Roadmap" at my former consulting firm) defines a time- and event-based view of what would need to be done and when. Not all companies create such a roadmap – just as not all companies formally define strategy. However, knowing *where you want to go*, and *how you plan to get there* are key to actually getting to a specified destination.

The actions required to be taken are most typically a *project* of some type. Some projects may involve implementing, upgrading, or enhancing a system or infrastructure (e.g., upgrading an ERP system). Some may be purely process-based (e.g., defining and improving the processes by which the IT organization delivers support to its business user constituency).

Projects always have costs. Even a purely process-based improvement, conducted entirely with people already on staff, requires a team of people who would otherwise be doing other work. The cost incurred is the opportunity cost of their time – the salary and benefits expense of the portion of their time that is dedicated to the project. As such, projects are *investments*, akin to any other investment in a portfolio. The next section talks about viewing the IT project portfolio as an investment portfolio, and implications of taking that view.

Before we get there, however, I want to raise a few more points. We talked about translating goals to plans, where the plans are frequently projects of some type. However,

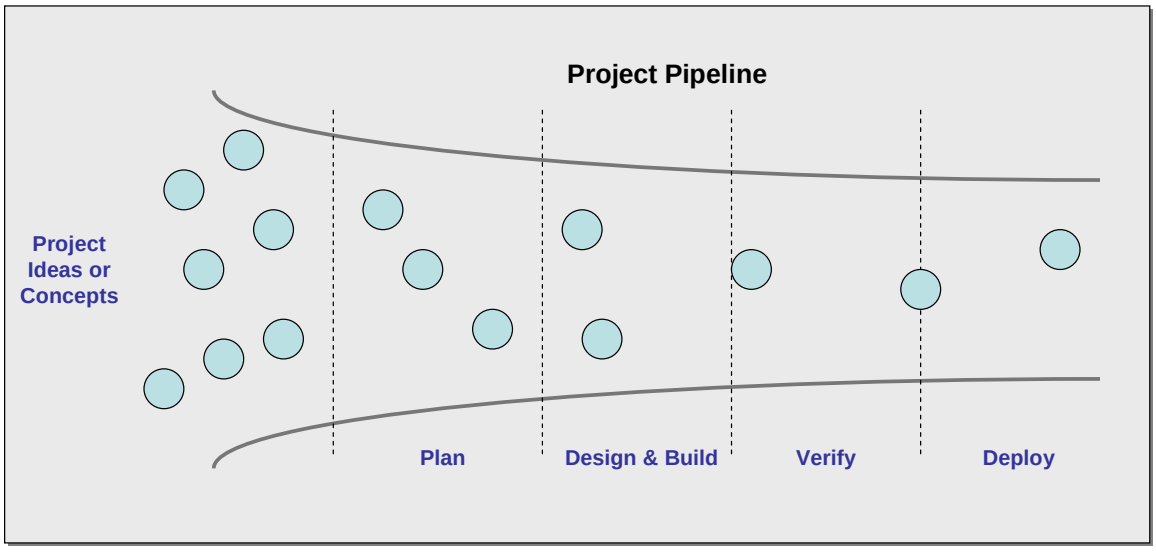
not every IT project in the portfolio is going to be strategic. There will be some – probably the majority as a matter of fact – that qualify more as tactical responses to operational issues than strategic initiatives. Adding small sets of features and addressing known problems with existing systems are typical examples. An organization can only do a few "big" things, but can potentially do a larger number of small, focused improvements. Hence, the actual pipeline by number (not size) of projects may be largely tactical in nature. The trick is to make sure tactical improvements do not get in the way of strategic initiatives. We will return to this topic later.

Finally, just because strategic goals and plans are defined in a periodic planning cycle does not mean that projects can never be conceived outside of that planning cycle. Such a limitation would be very constraining to the flexibility of the organization. Opportunistic project ideation and acceptance are common and desirable. As opportunities arise, they should be evaluated within the context of the IT strategy – as well as against the results they claim to be able to deliver. Acceptance of new projects may delay other previously-planned efforts or require funding not already arranged, and so may require an update to the IT strategic plans. This topic, too, will be revisited later when we talk about IT project valuation models.

MANAGING THE IT PROJECT PIPELINE

Ensuring that the IT project pipeline contains the "right" project investments requires careful consideration of each project prior to its introduction into the project pipeline.

The IT project pipeline is the collection of projects currently in process, where "in process" can mean anything from an idea that has been approved for further investigation to a project that is nearly complete. An example project pipeline is shown below. Ideas are generated at the left, vetted, approved or rejected, and then the project is started, implemented, and completed.[15]



[15] The specific number of phases and their names varies by organization, as will be discussed further in the Business System Implementation section.

Unfortunately, the project pipeline is frequently sub-optimal. Typical issues include the following:

- Too many projects
- Excessive numbers of tactical projects interfering with strategic initiatives
- People assigned to multiple, concurrent projects without recognition that they are therefore available less than 100 percent to each – leading to schedule variance and slippage
- Projects in the pipeline that run counter to strategic direction[16]
- Small, unimportant projects consuming resources that could be used to meet more critical needs

The end result of all these issues is a failure to meet business goals or needs (not to mention the inherent inefficiencies). The cause? Frequently this result comes about because project approval decisions are decentralized and made at lower levels in the organization. This is particularly true for small projects. The dollar cost outlay for a small project may be little enough to not garner any particular attention from the management team – but when you add up all those small buckets of effort and the resources they require (time, money, people), you end up with a lot of sub-optimal spending. You can end up with a whole lot of organizational resources dedicated to work that is not critical – and that interferes with work that is.

The solution? Well, as stated above, the trick is to evaluate the projects fully before they are allowed in the pipeline. As we mentioned in the prior section, some projects may be approved as concepts as part of the periodic planning process, while others some may come up opportunistically.

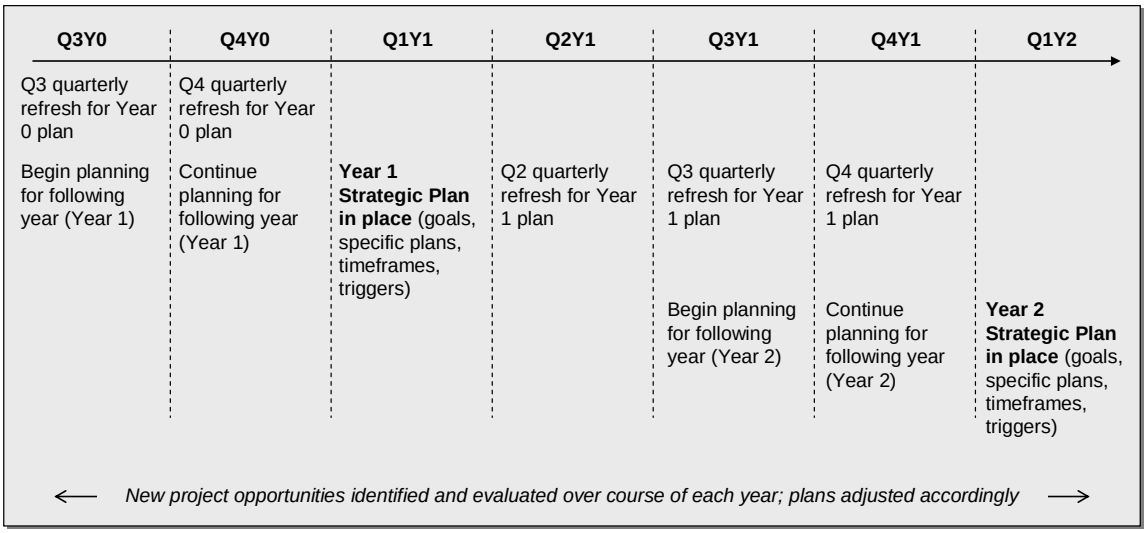
Some consideration should be paid to whether the project contributes to strategic goals – or distracts from them. An organization could centralize the approval decision-making, or could allocate a set of dollars for small, incremental projects to be managed at lower levels, while ensuring that the bulk of the money and resources goes to critical projects. A better solution – if implemented properly – might be to determine a formal method to value the projects and use that value to approve or reject them. In that approach, it matters less whether the approval decisions are centralized or decentralized – or decentralized for projects underneath some specified project size (which could be defined by cost, effort, time). The next section talks about some approaches for performing this valuation.

PERIODIC IT STRATEGIC PLANNING PROCESS

We have already talked about periodic IT strategic planning. This section outlines one possible approach. This approach is similar to one I saw at a prior client for strategic planning related to product development. This approach is very simple in concept but very powerful in practice.

[16] E.g., a planned business system implementation on an NT/Windows platform when the organization is migrating toward a Unix/Java platform.

A diagram is provided below. Strategic planning for the following year starts during the prior year, with the intent of having the plan in place before the new year starts. (An organization can decide which specific quarter to start the next year's plan, based on their environment and needs.)



Over the course of the year, new information is accumulated. This information can include, for example, progress-to-plan on current projects, new information on evolving technologies and capabilities, changes in business climate or direction, changes in the organization, such as a new key business stakeholder, changes in the organizations financial picture, and so forth. As the information evolves, the plan is revised. Quarters (or other periods, based on preference) can be used as triggers to define a plan update.

The periodic refresh process does not need to mean that someone needs to spend a month on a 100-page document – it can simply mean that the original plan is reviewed, an addendum provided, and the new plan (e.g., a one or two page summary) for projects, timelines, and triggers documented.

New information and new project ideas are not guaranteed to show up on quarter boundaries. The periodic refresh is intended solely to make sure the organization is not letting the information get stale. As opportunities arise, they may be considered in light of existing strategy and plans – and those plans may be adjusted, if the new opportunity's attractiveness warrants it. An up-to-date perspective on the strategy and plans, in fact, is quite powerful for evaluating the impact of a new opportunity on the organization's existing plans. Without one, the impact of a new opportunity on other projects may be underestimated.

As mentioned above, this type of approach is an example of the ways an organization can approach IT strategic planning. Formal IT strategy is frequently not all that visible – if existent – at an organization. My personal perspective is that many IT organizations lack good, business-focused staff that are particularly competent at or assigned to either

strategic planning or the financial fundamentals inherent in valuing project opportunities – which happens to be our next topic.

Evaluating Project Value – How Do You Choose the Best Projects?

If you were paying attention, and your grasp of English is pretty good, you probably recognized that the title of this section is redundant. The meaning of the word "evaluate" is to identify the value or worth of something. And, everyone knows that "value" refers to the worth of something.

Why did I bother being redundant? Well, because identifying whether you should do a particular project, in light of the other things you could be doing, involves two things: *identifying the comparative value* of the project, and *making a decision* based on that valuation.

A decision based on no valuation whatsoever may well be flawed. And, performing a valuation if you do not have the willingness to make hard decisions based on the outcome just wastes time and effort. Unfortunately, in many cases projects are undertaken because some person or people have already made up their mind(s) that they are going to do it. Executive pet projects are a prime example: just because an executive proposed a project does not make it the most beneficial use of the investment dollars.

There will always be choices about where to spend IT dollars. There is frequently a longer list of desirable projects than are feasible, due to cost, resource, or timing constraints, to accomplish within a given planning horizon.

There are myriad possibilities for project valuation decision frameworks, from simple to extraordinarily comprehensive. And, it is not uncommon for organizations to not have *any* formal decision framework to evaluate projects – as each project is conceived, its proponents may produce materials to support the validity of the project, and these supporting materials may be produced in no particularly standard way.

However, ensuring that IT dollars go to their most effective use requires that the candidate uses be reviewed and evaluated using some kind of framework to gauge the value of the potential investments.

Before we talk about specific framework approaches, let's look at an example. Let's say that a particular company wants to upgrade their ERP system, implement a PLM system, and improve the system integration infrastructure, but are unable to do all three due to organizational constraints. That is, the sponsors from the management team for the three efforts are partially overlapping, and the IT organization cannot support all three projects. (Lack of money available to fund all three is another possible – and fairly common – constraint.) Operations and Finance claim the ERP upgrade is critical to addressing a long list of known issues, Development and Product Marketing claim they are hampered by the lack of a PLM system, and IT says that they need to upgrade the integration

architecture to be able to scale well with the large number of systems coming on line. How does the organization decide how to evaluate these against each other?

First, a quick observation. The first two projects are driven by business organizations citing business needs. The last is driven by the IT organization based on architectural needs they see.

In an organization with a fairly powerful IT group – absent a formal decision framework – the latter would likely get funded. In an organization where IT is relatively less powerful and perhaps considered subservient to its constituent business groups – again absent a formal framework – the business-driven projects would likely get funded, perhaps at the cost of addressing very real problems with architecture and infrastructure. (The fact that the projects are driven by IT does not necessarily mean that they do not offer real *business* value – they could address current or potential stability problems, or have other business-impacting results. Organizations where IT is overwhelmingly directed and funded by constituent business groups may fail to address architectural and maintenance issues over time – with significant, long-term impact to their business community.)

Regardless, without a formal, unbiased decision framework, the decisions are based on power relationships – with sub-optimal decisions as a possible result, and biased decisions fairly likely.

There is no one framework that is perfect for all organizations. The “right” framework is the one that results in the best decision being made. But, determining which one that is can be challenging. Project valuation frameworks could range from simple *qualitative* tools to organize available information and facilitate discussion between intelligent, experienced decision makers to complex *quantitative* tools that produce a numbered result of some type (e.g., a specific dollar value for a project, or a score calculated using multiple, weighted criteria). We will come back to this thought in a little bit.

Context for IT Project Valuation Frameworks

Before we move on to evaluating specific approaches, it is time to step back and think about what we are trying to do and what theories or approaches might therefore be applicable.

An IT project is a *financial investment* a company makes in the hope that the effort provides sufficient value to the company to make the investment worthwhile. The investment in a project may require some direct cash outlays, such as for software, licenses, and consulting fees. In addition, internal resources will be required to participate in the effort, including management oversight, business users to guide the choices made throughout the implementation, and IT staff to participate in the technical side of the implementation.

Internal resources are a direct project cost because, although you may pay their salary and benefits expenses whether or not you proceed with the project, the portion of their time

dedicated to the project would be available to do other things if they did not participate. So, there is an *opportunity cost* for their contribution to any project. In addition, releasing team members to a project may require someone else to temporarily fill their daily job – we typically call this a "backfill" cost.

As IT projects are investments of money into something you hope has value – that is, an IT project is a financial investment – traditional financial investment concepts apply. In other words, *the importance of a project is its impact to the bottom line* (the company's profits) – if there is no expected impact to the company's results, then the project is not worthwhile. Period. And, traditional risk/reward concepts apply: a risky project should be expected to have commensurately higher return, while a lower risk project may be worthwhile at a lower return.

HOWEVER, no one has quite figured out how to estimate risk and reward for IT projects to any kind of granular or accurate degree. Periodically, valuation methods surface as a hot topic for awhile and different approaches are discussed, but the problem has never been particularly satisfactorily resolved. Organizations have typically been fairly lousy at applying standard financial concepts to IT projects.

WHY? Well, because it is hard. Attributing expected impact to profits by an IT project is very difficult – to estimate bottom line impacts at all is difficult, to estimate them in a meaningful, *accurate* way even tougher. A thorough financial analysis would estimate the impact of the project to the organization's cash flows. For example, putting in place technology *X* will decrease costs by *$Y* million per year over the next *Z* years, which results in a current value of *$T* for the expected incremental future cash flows. (This example uses a financial valuation technique called the Net Present Value (NPV) – the positive impact to the company's cash flows in today's dollars. It can be extraordinarily difficult to calculate for IT projects.)

In some cases, there is a known qualitative benefit, but there is an estimation problem identifying the actual cash flow improvement from the project. For example, a new decision support tool will provide more information visibility and enable management to make faster decisions. How might that project improve profits (through either increased revenues, decreased costs, or both)? An organization might expect the tool to allow it to respond to changes more quickly, for example, by scaling production up or down, or by changing the mix of what is being produced, based on dynamically observed patterns and trends. While that flexibility and responsiveness is inarguably a good thing, particularly for volatile environments, translating that capability to an accurate estimate of cash impact may be difficult and error prone.

In some cases, a new technology takes hold and becomes a cost of doing business. For example, once one online retailer started offering package delivery tracking, customers loved it and began to expect it. That put pressure on other retailers to add the capability – which also meant all the logistics providers had to offer it. In this type of case, if you do not proceed, your competitors may gain on you, which then, absent other factors, may

impact your future profits – but again, the actual cash impact may be very difficult to quantify or estimate.

Some projects may have more easily estimable cash flows. For example, a pharmaceutical manufacturer may have an FDA compliance problem. That is, they may be out of compliance with one or more of the FDA regulations and guidelines that govern pharmaceutical manufacture in a particular manufacturing plant, and may have received a notice of the issue from the FDA following a plant inspection. (For the purposes of this example, we will assume that the problem is related to the IT systems in use at that plant, and so is one that an IT project would solve.) The company may well know the throughput of its plant. The company may also have fairly comprehensively forecasted revenue for the product it produces, and may be able to attribute the portion of the product's revenue that would be manufactured by that specific plant. They might even be able to estimate a probability that FDA will shut down the plant due to its lack of compliance. Therefore, the company would be able to estimate the cost of the shut down at a daily, weekly, or monthly rate, based on the revenue that the plant would be generating if it were not shut down. However, the timing and duration of the shutdown may be highly uncertain. And, at some amount of money, the specific cash flows may not matter anyway – the company executives might be almost equally unhappy losing $2 million a day as $1 million a day (executives tend to not like that level of impact to expected revenues and profits – nor do their shareholders). So, a detailed cash flow analysis – even if it could be performed accurately – may not result in any different decision than other approaches.

In summary, cash flows from IT projects tend to be too uncertain apply cash flow analysis in the typical methods. Beyond the difficulty of estimating cash flow impacts from IT projects is the fact that IT managers are not always that well grounded in financial concepts – and the Finance people who support them are frequently not all that well versed in technology – which makes an *effective* partnership between IT and Finance for the purpose of valuing project choices difficult and rare.

In this section, we will cover various ways to make the decisions, and the concepts behind those approaches. We will suggest some approaches that may be more effective than others – but will also talk about some of the hokey things organizations actually do. My theory here is that if everyone – Business, IT, Finance – has a better grasp of the whole picture, better decisions will result – hokey or non-hokey method notwithstanding.

Quantitative and Qualitative Approaches

We talked above about quantitative and qualitative approaches, where a qualitative approach provides analysis to facilitate discussion and application of subjective judgment, and a quantitative one results in some kind of number to guide the decision. We also said that the "right" framework is the one that results in the best decision being made.

One problem with a qualitative approach is that you need exactly the right kind of people reviewing the projects and making the decisions – they need very good experience and

judgment. Another potential problem is the potential for very superficial analysis and hand-waving type of decision-making – unless the method forces deep analysis, you may end up with no more than gut instinct as a guiding force[17]. For this type of approach to be successful, you need to have people who will question the assumptions, the options presented, and the recommendations proposed to ensure that the decision is based on a full, in-depth analysis.

Various problems can occur with a quantitative approach. Sometimes, companies institute a method that requires better quality information than is actually attainable. In this case, you may end up with a very impressive-looking calculated value out to 3 digits of precision that is, in fact, just a wild guess (e.g., the score may be accurate *plus or minus 10, 40, 80%!*), rather than a score with sufficient accuracy to support rational decision-making. In addition, it can be difficult to ensure that very detailed scoring mechanisms produce good decisions. The type of people who are prone to produce a very detailed model are not always as well equipped to take a big picture view or to explain the model to key stakeholders in an easily comprehensible way. (If a model cannot be explained or understood at a conceptual level, there is a pretty fair likelihood the model is wrong. Among other reasons, on overly complex model is not particularly amenable to garnering meaningful feedback and direction from the key stakeholders and decision-makers, if it is mired in details that cannot be abstracted to the level that would enable them to comment and suggest improvements.)

Characteristics of Good Valuation Frameworks

Feasibility and accuracy of any particular method is paramount. Given the degree of uncertainty in IT projects, both overly complex and overly quantitative approaches are likely to: a) give wrong answers, and/or b) not be used, due to the level of complexity and effort involved.

Regardless, there is a set of information about a project that should be considered, in some way, as part of the go/no-go analysis. A good framework should meet the following criteria:

Characteristics of a "good" framework

- Considers whether a project is aligned with business and technical strategy (a project could be strongly aligned, neutral with respect to strategy, or counter to strategy)
- Considers the benefits that the project is expected to deliver (costs reduced, revenue increased, ROI, or other methods)
- Considers the costs of doing the project (such as people, time, money, and other opportunities forgone)

[17] It should be mentioned that some people have very good gut instincts. Again, with the right person or people in this kind of decision making role, you could end up with very good decisions. Once organizations grow to a certain scale, this type of approach becomes less effective, however.

- Considers the impact of doing the project on other efforts (risk to or impact on other projects planned or in progress, risk to or impact on day-to-day operations, impact to management team overseeing the project)
- Considers any other relevant, critical risks of the project
- *CRITICAL:* Ensures that assumptions driving the result or decision are documented and reviewed
- *CRITICAL:* Ensures that the *quality* of all the information listed above is used to qualify the decision or result, i.e., if the information is largely wild guesses, they are viewed with in the context of the decision-making *as wild guesses*

The last point may be a little complicated to summarize in a bullet, so an example is probably in order. I had a client that was proposing to use a very complicated scoring mechanism to judge the value of possible projects[18]. They had three major dimensions of the score. Each dimension had multiple criteria – there were between five and ten different criteria per dimension, and each criterion had an associated weight. There were two scores for each criterion, one for the benefit along that criterion and the other for risk along that criterion. It was a very comprehensive model. The problem? The quality of the information, especially early in the conception of a project, is not particularly high[19]. If your information is high-level and not particularly granular or accurate, you multiply the error in any calculated score the more criteria and weights you add (since every criteria has a plus-or-minus-x range of precision). So, you can end up, as I suggested before, with a calculated score to multiple digits of precision that is not even remotely accurate.

Precision refers to the number of decimal points in an answer. Accuracy refers to whether the number is right or not. If you have a 1-10 score shown out to three decimal points, but the accuracy of your result is only plus/minus 1, you will show very precise – and largely wrong – calculated values. For example, you might end up with two projects with values of 3.625 and 4.125, respectively, as calculated scores. You might be tempted to say that Project 2 is a better choice than Project 1. However, if your quality of information is such that those scores are effectively plus/minus a whole digit (e.g., 2.625 to 4.625 for Project 1, and 3.125 to 5.125 for Project 2) than you cannot say that Project 2 is actually better than Project 1.

A former boss of mine had an expression: “Measure with a micrometer, mark with a pencil, chop with an axe.” The point was that, if you know you are going to end up chopping with an axe, because that is the tool you have, then there is no point in preparing as if you will cut with a high-precision instrument. Or, as I told my former client, “Don’t make your precision greater than your accuracy.”

What we ended up doing with my client was recognizing that information quality improves as the project progresses, which means that the project valuation model can get more organized and quantitative at the same time. We defined project gates at which we would perform the evaluation. At the first gate, the valuation was largely qualitative, with

[18] This was in the context of product development projects, not IT projects, but the situation is analogous.
[19] This is true both for product development and IT projects – probably to a greater degree for development.

a high-level non-calculated score with which to "eyeball" the value relative to other projects. Later gates used quantitative criteria and weighted scores.[20]

Probabilistic Models

One approach to dealing with imperfect information is to use a probabilistic model, with different outcomes and estimated probability of each outcome. In that case, you would calculate an *expected value* rather than a guessed value, where the expected value for any particular criteria's value score is the sum of the (value) times (probability) for each outcome.

You could take this approach in a few different ways. One approach grounded in financial theory would be to look at the present value of incremental cash flows from the project.[21] If you look at an example of a technology that becomes a new "cost of doing business", like the package tracking example mentioned above, you might estimate the following impacts[22]:

30%	Market share drops 1% this year, and 5% next year, resulting in a decrease in profit of $1.5 million dollars of profit in today's dollars
40%	Market share drops 5% this year, and 10% next year, resulting in a decrease in profit of $2.7 million dollars of profit in today's dollars
30%	Market share drops 6% this year, and keeps dropping at a steady rate over the next five years, resulting in a decrease in profit of $10 million dollars in today's dollars

Using these numbers, the expected value of the project is $4.5 million, since that is the amount of profit (net of project costs) that will otherwise be forsaken if the project is not taken on. Probabilistic approaches can potentially use any number of scenarios, although a best-case, expected-case, and worst-case approach is not uncommon. Too many scenarios could introduce additional estimating error, as the probability values are each estimate with an inherent degree of error.

Scoring Frameworks

As mentioned above, a scoring type of method is one where a number of criteria that drive value for a project are identified, and an importance weighting is assigned to each. For example, each criterion for particular project may be assigned a value from 1-3, with one being low and three being high. The end result, or score, is the sum of the products of the value assigned and its weighting, as in the example below:

---

[20] I would have preferred to make the tool less complex, and qualitative at the second gate as well.

[21] This approach is a Net Present Value (NPV) approach, which is based on discounted cash flow analysis (discounted based on the time value of money). See the Appendix, "Finance for IT Managers" for further details on exact use.

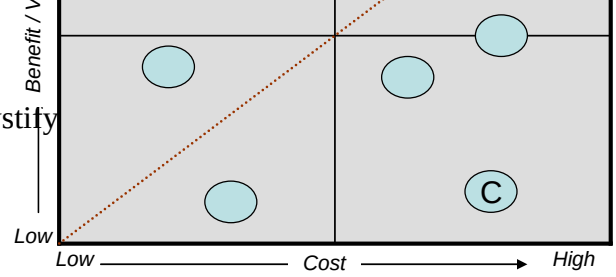
[22] Getting from estimates of revenue or cost impact to cash flow value in today's dollars requires some financial machinations. Among other things, amount and timing of cash outlays and inflows for the project need to be identified, as does a discount rate for the cost of capital. This particular example is made up for illustrative purposes. Refer to the appendix for more details on calculation of NPVs.

Criteria	***Weight***
Impact of project on stated strategy	40%
Degree by which project increases revenues	15%
Degree by which project decreases costs	15%
Degree of competitive advantage conferred	30%

One problem with this type of approach is that the weights are typically arbitrary. All of those impacts are good things: achieving strategy, getting competitive advantage, improving profits through higher revenues or lower costs. And, the 1-3 values assigned are subjective and arbitrary. Again, the relevant measure of a project is its impact to the bottom line – no more, and no less. Does this method measure that? Given two projects measured using this method, will the resulting score be an accurate measure of which project has more value to the company? It depends on the specificity of the criteria, the appropriateness of the weights, *and* the good or bad judgment of those individuals who assign scores to each weighted criteria.
Scoring frameworks are not always good at measuring risk of a project, or interdependencies between projects – although a scoring mechanism could be designed to have those things in mind. It also fails to capture the concept of the time value of money – money earned sooner is more valuable than money earned later – although a detailed translation table to get from present value of cash flows to a score value could be devised as well. However, if you can estimate those cash flows, you are better off just using them and forgetting multiple criteria and weights!

One way to use a calculated score type of model in a more meaningful way may be to use it within a very simple comparative model, like a bubble chart. A bubble chart can show different IT project choices in a comparative way without relying outright on the *exact* calculated value, which may be highly uncertain in any event. That is, the results may have more meaning at a summary level (high value/low cost project versus low value/high cost project) than at a detailed level (comparing a score of 2.1 to a score of 2.2).

For example, in the bubble chart at the right, you can see that Project A has a relatively high value score, but a low cost score – it would provide a lot of benefit for not too much cost. The benefit provided by Project B is about equivalent to its cost. And, Project C provides little benefit, but at a high cost. If you had to select among projects, you would prefer A



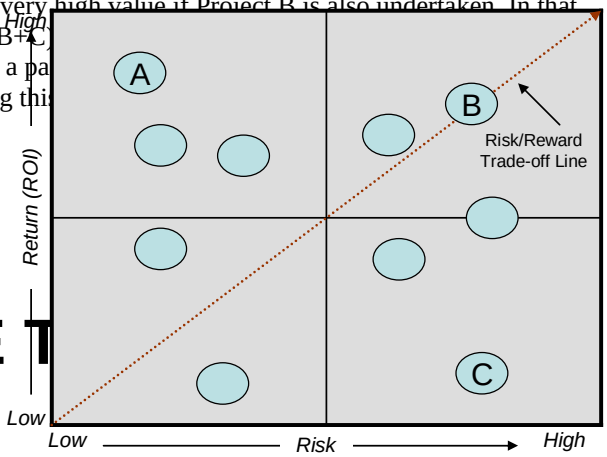
over B and B over C – absent other factors. [23]

The example above compares the cost of a project to the value it provides. The value of a project could be measured a number of ways, as mentioned above (increase in revenue, decrease in costs, ability to meet strategic goal, ability to avoid a negative outcome, and so forth). And, as mentioned above, the costs may include resources (time, people, money) as well as opportunity costs (what you cannot do if you do the project).

An alternative approach is a risk-based approach. This approach is again like an investment portfolio management approach. The purpose of a portfolio approach to investments is to maximize your returns while minimizing your risk. If you think of the risk of adding a particular stock to your portfolio, the relevant risk is its c*ontribution to the riskiness of the overall portfolio*[24]. As in stock investments, one can sometimes reduce risks through diversification (though it may be somewhat more difficult with IT project investments – this topic is discussed further in the next section).

Likewise, the relevant *return* is its contribution to the returns of the portfolio. This concept is not typically applied in the investment world; if two stock investments have correlated results (e.g., two companies in the same industry subject to the same environmental and market conditions), then the result of adding both to the same portfolio is that they are less able to reduce the portfolio risk than two investments whose returns are significantly less correlated. However, in the IT world, there may be network effects between projects – one project may become more valuable if another is conducted as well. For example, your investment in a particular technology may become more

[23] Other factors include such things as relationships or dependencies between projects. For example, Project C might have very little value in isolation, but return very high value if Project B is also undertaken. In that case, you might find on a bubble chart that Projects (B+C) Project A. Another factor to consider is the impact of a pa – the risk-based approach may be better for estimating thi



[24] SEE T

APPENDIX B. FINANCE FUNDAMENTALS FOR IT MANAGERS" AND "

" sections for more information on this and other financial concepts.

valuable as you are able to achieve greater and greater benefits from it.[25] So, you may want to consider the value of any given project in light of other candidate or in-process projects.

As in the cost/benefit approach, a bubble chart may be a valuable way to make comparisons between candidate projects. In this example at right, the axes are calculated scores based on levels of determined risk and reward. In this case, Project A is low risk, but high return, and Project C is high risk, but low return. Project B is about equivalent based on the calculated risk and return scores[26].

As in the prior case, the key problem with using a risk-based approach is the lack of perfect information from which to make a decision. A probabilistic approach may produce better results. One might combine a scoring model with a probabilistic approach to come up with *expected* risk and return scores to improve the accuracy of the output.

Business Case Approach

The probabilistic and scoring models are both quantitative models. The scoring version could be used in a semi-qualitative way using a bubble chart type of comparison, supported by detailed information regarding the assumptions behind each projects calculated values. An alternative, common approach is a business case. A business case is simply a presentation of the estimated value of a project. It is likely to include both quantitative and qualitative information, and should include all key assumptions in order to produce a meaningful decision. A business case is a very good way to sum up the value of a single project. It is less equipped to allow comparisons between projects. A good business case should include the following:

Costs of the project	Software and licensing; consulting fees; costs of allocated resources; other relevant opportunity costs; ongoing maintenance and support costs. Timing of the costs is also relevant.
Benefits to be achieved from the project	Qualitative benefits; *measurable* business improvements; impact to bottom line if estimable; and so forth. Timing of the benefits is also

[25] This is analogous to the market situation for an Operating System (OS) – an OS becomes more valuable as more and more application developers adopt it and create applications for it. The OS can then deliver greater value to its customers than it could with fewer applications that are provided for it. When this occurs, we say that a market exhibits "network effects".

[26] Again, how you produce scores from known information and qualitative evaluation is critical to coming up with valid comparisons and decisions.

	relevant.
Key milestones and timing	Time to completion; *key* milestones along the way (a business case is merely a proposal, not a project plan or schedule)
Resource requirements and team needed	Team structure; proposed oversight structure; proposed team members; specific requirements for team dedication (e.g., role *X* at *Y*%). Timing of resource needs is also relevant.
Risks	Risks of failure; risks to other projects or day-to-day operations
Options	Other options for this project (doing a different project instead, or doing the project in a different way)
Recommendation	Conclusion, based on all of the information above, regarding the best course of action
Assumptions	Assumptions behind the quantitative estimates and qualitative judgments should be documented, reviewed, and confirmed if necessary

Business cases may be documented in presentation format, or possibly a MS-Word type document, frequently supported by spreadsheet calculations for budgets, value, etc, and any other relevant quantitative information. My personal preference is for presentation style formats (e.g., MS PowerPoint) with one caveat… Occasionally, you can get "bullet-point-level" analysis if you expect a presentation style document; that is, the level of thinking may go no further than that needed to type in a bullet. If you want to guarantee a fully thought out proposal, you will need to ensure that detailed supporting information is provided in some way or another.

Conclusions

Alternate methods are available for calculating IT project value. Other financially-oriented methods, such as internal rate of return (IRR) are provided in the Finance for IT Managers appendix section.

The problem with quantitative methods is that bottom line cash flow impacts from IT projects are uncertain and therefore extraordinarily difficult to estimate. You can devise a very complex model, but if the estimates are bad, you face a "Garbage In, Garbage Out" problem.

My personal opinion (which you may take merely as such, or alternatively as the "voice of experience" of one who has seen companies try various methods, and has seen the inherent issues) is that:

- Organizations are usually better off making *qualitative* judgments, but which are supported by *quantitative* information that is estimated as well as it can be (i.e.,

using valid assumptions, real data when possible, considering all relevant and critical factors, etc.)

- The outcome is better if uncertainty can be baked into the method, such as by using the simplest-possible probabilistic approach (e.g., expected-, best-, worst-case scenarios for key quantitative estimates to produce an expected value)
- Summary tools like bubble chart comparisons are good for ensuring projects *actually are compared* in a meaningful way without forcing decisions directly from inaccurate calculations
- That simple approaches are better (more reliable, easier to use) than complex approaches; that is, the KISS rule (Keep It Simple, Stupid) is very much in force

Finally, before we leave this section, whatever approach you use, here is our Key Thought:

Don't make your precision greater than your accuracy

APPROACHING THE IT PROJECT PORTFOLIO AS AN INVESTMENT PORTFOLIO

Some companies view IT as a cost center. IT is a place you are obligated to spend some money, and you try not to spend too much of it. It is true – you must spend money on IT to run your business. However, the portion of your IT spending on *projects*, from incremental fixes to major initiatives, may be better viewed as an investment portfolio.

People often talk about a "portfolio" of projects. It is a very commonly used term in the business world. But, frequently when people talk about their project "portfolio", they really mean their project *pipeline*, which as we discussed above, is the set of projects currently in process. This is just one more example of where people abuse and misuse terms because they sound more impressive than the strictly precise term would sound.

A portfolio, however, includes the concept of risk. The fact that it is a portfolio of IT projects, rather than financial assets, does not much matter – it is a collection of financial investments that happen to all be investments in IT-related work. And, much like our personal investment portfolios, the concepts of risk minimization through diversification and asset allocation are applicable. A significant benefit of a risk-based portfolio approach to IT investments is the ability to diversify risk due to uncertain technology evolution.

New technologies emerge all them time. However, it is frequently unclear which ones will "take" and in what form. There are frequently competing standards during times of technical change, with the likely winner to be determined. Sometimes, new technologies emerge that the consensus of opinion says will take hold – however, some of these are technologies in search of a solution, and they may not take hold until effectively baked into a solution for a real world problem. The pace of technology change in IT is so rapid that corporate IT shops may either spend too much time chasing the next fad, or may be caught unprepared by a new technology that catches on, such as many traditional bricks-and-mortar firms were done during the dot-com expansion.

So, what is an IT manager to do? Part of the job is keeping up with all these emerging and evolving technologies and trying to get a read on where the trends are going. A conservative approach might be to sit back and let the future sort it out. The alternate approach on the complete opposite end of the spectrum is to try to chase every new technology buzzword that looks like it might take hold. Both are risky, however. The risk of the former is that you might end up caught way behind when there actually is a sea change, such as many bricks-and-mortar companies were when a significant amount of business moved online during the dot-com boom (where it has stayed online, despite the later dot-com bust[27]). Chasing every new technology, however, will be expensive, will likely focus your organization on technology over business results, and will likely be wasteful, as many new technologies take much longer to fruition than anyone expects, and may end up in significantly different form from where they started.

A balanced approach would leave some amount of IT dollars to high-risk, potentially high-return projects using new technologies, while ensuring stability of the overall environment.

The IT project portfolio could therefore be viewed much like a traditional investment portfolio, with goals, a stated risk tolerance level, risk/reward trade-offs, allocation strategies, and portfolio diversification to consider prior to each investment. Small companies may be more comfortable with risky new technologies. Large companies are more likely to be expected by the market to have financial stability and may require a more conservative overall investment portfolio – however, they may well have the dollars available to make a few risky investments in some bleeding edge technologies.

An organization could budget and plan to spend a particular portion of the IT budget on risky new technologies that have the potential for significant improvements to the company's capabilities or results. In other words, they would plan to spend a percentage on high-risk/high-reward projects. This could be a higher or lower percentage depending on your and your company's level of risk tolerance. This approach is analogous to management of a stock portfolio, where you allocate fractions of your portfolio to different levels of risk, from cash equivalent, to fixed income, to small cap, international, and large cap stocks, with the percentages of each depend on your own personal risk tolerance.

If you question your ability to "read the tea leaves", want to avoid missing key technology shifts, and do not want to risk failing to deliver to needs or wasting IT dollars by making large bets on unknown technologies, a risk-based portfolio management approach may allow you to strike the right balance.

[27] The dot-com bust did not return the world to the pre-dot-com state. The so-called "bust" was nothing more than a reaffirmation that traditional business concepts – such as the fact that, in the long-run, revenues must be at least as great as expenses, or you do not have a business – were still valid. The so-called "New Economy" did not mean that profits no longer mattered – a fact that somehow escaped even the investment and venture capital communities that should have been expected to understand otherwise.

It is time to present a few examples of applying investment concepts to risk management for IT investments.

One example is competing standards or technologies – which one will win? You might think of wireless infrastructure, or perhaps integration infrastructure, as possibilities where you need a solution and you are not sure which direction you might want to go. Pick any example where you have an uncertain technology choice. At any rate, say you want to enable a particular capability within the company's four walls, but you want to make sure that you do not end up beholden to one standard when another may either win outright, or may evolve into a solution more fitting to your environment – or simply, that you do not yet have the experience with either to judge which one is better in your environment. You could make smaller bets with two standards, if there are two predominant competitors from which the eventual de facto standard will likely be chosen. In that case, you will gain experience with both, and will be able to expand the initial implementation as the situation becomes clearer. You may end up replacing one half of the installed base when you have determined which technical direction you will go, but that is cheaper than replacing the entire base, and if the bets are small enough, may not be particularly painful.

The second example is a simple asset allocation example – what does the content of your project pipeline say about your tacit risk tolerance, and does it match your actual risk tolerance? If you were to compile a list of all the IT projects currently in process, would they be largely high-risk/high-reward, low-risk/low-reward, or somewhere in between? Examples of those levels might be something like the below:

Risk/Reward Level	Example
Low	Add enhancements and fixes to an existing system
Medium	Upgrade CRM system to a new version with improved feature set; add some new capabilities
High	Implement a new PLM system

If you looked at the resources you are contributing to projects – which you could examine by people, money, or time – you might notice that you have more resources committed at a particular risk level than you might expect. For example, you may notice that you have a lot of high-risk/high-reward projects in the pipeline, whereas your organization considers itself to be fairly conservative when it comes to risk. Or, you may notice that your organization is doing mostly enhancement work, whereas you would like the organization to be fairly cutting edge with a tolerance for risk-taking in the name of high growth. If you never took a look at your pipeline, you might not realize the disparity. In general, higher-growth companies seem to have more risk tolerance, while slower-growth companies may have less.

WRAPPING UP: IT STRATEGY IN SUMMARY

We talked about developing IT strategies, and from them, plans to meet the strategies. The choices we discussed were:

No Strategy and Plan	If you do not know where you want to go, or how you plan to get there, you are unlikely to get there – unless you are very lucky
Implicit Strategy and Plans	An implicit strategy (e.g., a vision kept in the head of a visionary, charismatic leader) can be very effective if that leader is involved in all the necessary decision making. If key decisions are delegated and the vision is not shared, the strategy is unlikely to be achieved, except – again – through luck
Explicit Strategy and Plans	For larger organizations with delegated or distributed decision making, an explicit strategy translated to a shared plan to achieve it, is more likely to be realized

We talked about managing the project pipeline carefully to ensure that IT dollars are spent in the most critical places. To do so, we talked about formal valuation methods to approve or reject projects before they may enter the pipeline. We talked about the following choices:

Decentralized Project Approval	May result in large numbers of small, relatively unimportant projects consuming larger-than-predicted number of resources. May result in projects that do not support strategy, or that run counter to strategy
Review and Approval using a Standard Valuation Framework	Can help ensure that most valuable projects consume the scarce IT resources (people, money, time). Can help ensure selected projects support strategic goals and plans. However, potential to achieve these outcomes may be impeded by flawed frameworks or assumptions of perfect information when actual information is quite probabilistic
Pipeline Review and Assessment	Regardless of whether an organization plans to change its approach to project approvals, it may benefit from a review of projects in the pipeline, as the results may be surprising and may trigger some desire for adjustments

As for specific valuation methods for driving project approval or rejection decisions, we talked about the following:

Business Case Presentations	
Cost / Benefit Frameworks	
Risk/Reward Frameworks	

We talked about risk-based portfolio management, with the following choices:

Risk Reduction through Diversification	
Allocation Strategies	
No Risk-based Portfolio Management	

We talked about IT funding models:

Business Funded	
Separate IT Funding	
Hybrid Approach	

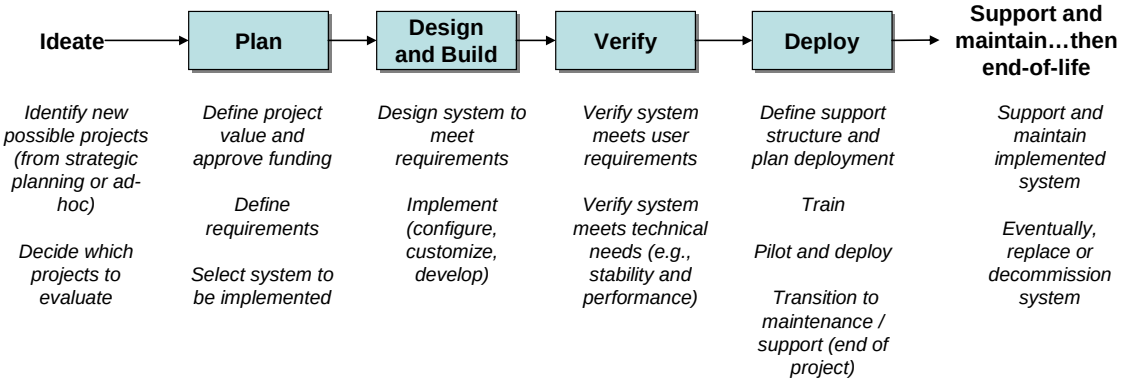
BUSINESS SYSTEM IMPLEMENTATION

This chapter presents the key concepts required to implement business systems successfully.

IT IMPLEMENTATION LIFECYCLE METHODOLOGIES

While IT projects may be run differently from company to company, all go through a natural evolution that starts with the basic concept for the project and results in (if brought to fruition) the system go-live and the transition from project to support. This methodology is called an IT project lifecycle. The nomenclature used to describe the lifecycle methodology can vary widely, although the basic concepts are the same. In addition, there may be industry- or company-specific tasks in the different methodologies – for example, pharmaceutical companies tend to have a whole set of activities in the lifecycle that are intended to ensure compliance with the FDA regulatory requirements under which they operate.

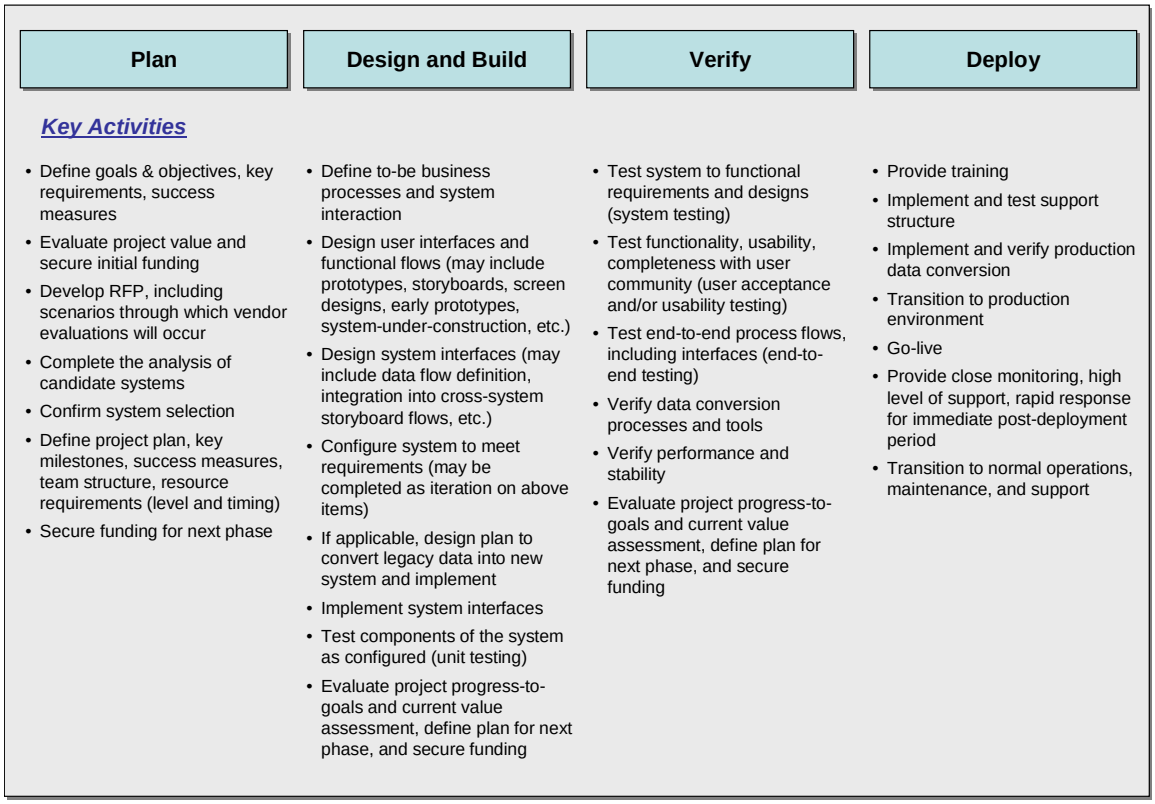
The IT lifecycle tasks are often grouped into phases of activities that occur in the same time buckets[28]. Each company tends to use a slightly different nomenclature for tasks and phases, and may also divide the tasks in different ways. However, conceptually the following progression occurs, by whatever name:



The diagram below shows the next level of detail to highlight some key milestones along the way. Note that the framework shown assumes that the project plans, progress, and funding are revisited at key decision points throughout the process. These decision points

[28] The taxonomy can get quite sophisticated, with a multi-level hierarchy of phases, stages, steps, activities, and tasks. The intent here is to describe the essential evolution of a project without overcomplicating it or describing any one particular model.

could be at the end of phases, or when key milestones have been met – the project plan should specify decision points for the project, if they are not specifically called out in the specific lifecycle methodology the organization uses. In the high-level diagram above, I have putted Design and Build in the same phase – some organizations may choose to split them, and to have decision points in between. Regardless, decision points for the project should be defined up front, so that deviations from plan – delays in schedule, resource availability problems, new resource requirements, discovery of limitations in ability to meet intended goals, and so forth – are more likely to be raised and addressed in a timely manner. In addition, we talked in the IT Strategy section about risk management. Pre-planned decision points at defined points in the project provide a degree of risk management. As new information arises, the organization can revisit and revise earlier decisions, for example: to cancel a project that has proved out to not be worth the investment; to invest more in a project; or to modify the approach, resources, team structure, or scope boundaries.



Also note that, while the above two lifecycle diagrams should activities proceeding over time from left to right, the diagrams should not be interpreted to mean that this framework shows a strictly sequential, linear process. The phases actually will overlap for some activities – that is, some activities that naturally occur as part of one phase may actually *start* in the prior phase.

I talked above about whether Design and Build are naturally two phases or one. I have lumped them together as a natural way to build in an iterative approach, so that some elements of a Design, Build, and lightweight Verify can be performed in an evolutionary, iterative, rapid-feedback-and-response type of approach. The most efficient methodologies typically *evolve* in a more iterative manner, whereas *strictly* sequential activities (such as: Define Requirements ➲ Design➲ Build➲ Test➲ Deploy) tend to result in later discovery of communication errors and also respond less well to changes in circumstances. Iteration is a more efficient approach to evolve toward a complete system; we will revisit this topic further when we talk about user centered approaches for IT.

One final note on IT lifecycle methodologies before we leave the topic. I have seen a very wide spectrum of approaches across firms, some fairly unstructured, some very structured. A degree of formalized approach to system implementation is required to ensure that business goals of the system are identified, requirements are specified, the appropriate system is selected, and then configured appropriately, and that the system is deployed effectively, with users fully trained, maintenance staff and procedures ready, and follow-on work planned. The right level of formality depends on the size of the organization as well as the size and complexity of the implementation. It is very possible to specify and use an overly heavyweight process to implement systems – too many deliverables, too many decision points, too many meetings. Using a process that is "heavier" than you need it will only slow you down. I have also seen organizations that do not spend nearly enough effort on up-front planning and definition, and end up with – surprise! – a system that does not meet users' needs, or worse, no system at all, because the project never gets completed.

I am somewhat of a minimalist. That is to say that I would recommend producing only the deliverables you really need, and that you apply the 80/20 rule – which in the context of an IT implementation means that 80% of the benefit to be derived from an implementation will likely come from 20% of the actual changes to be made. From an efficiency stand point, this means that you should focus a significant part of your energy on the 20% that really matters – and you should make sure you get that part absolutely right. You can mess up in other areas and still have a successful implementation.

If you are unsure which parts are the critical items, then you need to revisit why you are bothering with the implementation at all. The most critical items for the project are defined in the stated goals for the project, and those goals should have also been specified with intended measurable outcomes.

Put those goals and measures in 48-point font somewhere, look at them every day, and make sure all decisions are made with those goals fully in mind as the context. Because, once an implementation starts, people sometimes forget what the basic, stated purpose was. This may be hard to believe, but it is absolutely true. All sorts of project participants, stakeholders, and other "interested parties" will come up with additional ideas for improvement that have varying degrees of benefit.

Each potential change should be weighed carefully. It is very easy to get sucked up into details and apply equal effort to all potential areas of improvement, it is not only wasteful but risky, since it can distract you from the critical items. From a deliverables perspective, you *cannot* skip defining the goals for the system (measurable outcomes), requirements for the system, the process to be implemented, testing approach and results, and plan to ensure smooth deployment (training, support, physical configuration, and so forth).

Having a good IT implementation lifecycle is a good start. However, good project ideas can fall down during implementation anyways for reasons such as the following:

- Inadequate definition of the project's goals, scope, and milestones
- Inadequate processes to manage the project, including sponsorship, project management, team structure, role definition, and decision processes
- Inadequate stakeholder and business user involvement early, often, and throughout implementation
- Poor choice of system to be implemented
- Inadequate planning for the period after the system goes live
- Poor planning for the eventual replacement or decommissioning of the system

Each of these is described in the following sub-sections. In addition, since consultants and contractors are frequently assist in IT business system implementation, a section on effective management of consultants and contractors is included. Finally, the chapter closes with a summary of critical success factors for IT projects.

DEFINING THE PROJECT

You may be noticing an emerging theme in this book (there are a few different themes – we will review them at the end). In this case, the theme I am referring to is the need to identify *what you are trying to accomplish* if you are to have any hope of actually accomplishing it. This concept applies to the need for an IT strategy, the processes for managing a portfolio of IT projects – and the implementation of an IT project.

In the case of an IT project implementation, the objectives of the project should be very clearly spelled out up front. The beginnings of a project plan should start with a concise statement that summarizes why the organization is bothering to go through the time, expense, and effort to implement a system.

Prior to requirements definition, and prior to any detailed project planning, the following should be defined for the project:

- Concise objective statement: *Why is this project being undertaken?*
- Measurable goals or outcomes: *How will success be measured, and what are the target values for that measurement?*
- Key requirements driving the project: *What are the core qualities, expressed at a high level and from a business perspective?*

For example, one of my past projects implemented a system for a biotechnology company expecting to soon have their first product approved by the FDA, and needing a

system to support the increased volume of manufacturing quality test data that commercial operations entails. While I do not recall the specific verbiage we used for the project objective, we could have stated something like the following:

> *Implement a Laboratory Information Management System to manage test definition, result, and approval data for quality control operations that supports commercial volumes*

For the measurable outcomes for this example, we could specify the number of transactions that must be supported in a particular time period, the cycle time from the initiation of sampling to completion of all testing that the system must enable, and so forth. (Cycle time consumed by inefficiencies in test information management was one of the problems that needed to be solved.)

Key requirements at this particular step are only those that are core to the definition of the project. Additional requirements analysis is needed prior to system selection, but at this step needs to include all of those things that "make" the project. And, at this point, they should not assume any particular type of implementation – they should be expressed in terms the management team and business users would use. A bullet point list of 10 to 50 items may be sufficient as a starting point upon which to build.

Remember, these items – objectives, measurable goals, key requirements – are the things you will use throughout the project to set the context for *all* of your project decisions. You would be surprised how often people forget why they began a project. Once all the approvals are gained and the project is funded, people can start digging right down into the details and forgetting the why. You build a big team, everyone is working – and the project starts developing organically. After that point, you are at risk of the project veering off from the starting point. Two things can happen: you can get a whole bunch of "feature creep", as people request things they would like, that may or may not be core or strategic. And/or, you can end up with a system that does not meet intended goals, because you allowed the team to forget what they are.

For this reason, it is critical to keep the goals and objectives at the forefront of *everyone's* mind, from the management team overseeing the project, to the very lowest levels of the organization that doing the real work. Remember, everyone on the team, at whatever organization level, is responsible for *some* kind of decision. I guarantee they will make better decisions if they remember the project's overriding purpose.

So, print the summary of project drivers in 48 point (or larger!) font, display them prominently where they will be seen *every day*, and include them, at least as backup material, in everything you create along the way.

You will need to plan the project further than this, but we will return to this topic after we talk about selecting the right system – at which point, you will have more information from which to proceed.

SELECTING THE RIGHT SYSTEM

First off, we normally think these days about selecting a so-called "off-the-shelf" system[29]. It used to be much more common to build functionality you needed internally. But, with building system functionality from scratch comes the need to maintain it – fix bugs, add enhancements, add new categories of functionality, and so forth – all of which can be expensive, may be outside a company's core competence, and distract focus from the company's core business areas.

A company may still choose to build a tool if it needs functionality that does not exist in the market, particularly if the capabilities can grant it competitive advantage. However, much of the time these days, we do not think too much – in the IT space – about buy-versus-build decisions. For that reason, this section is focused on selecting among systems that already exist – which turns the problem from "build exactly what you need exactly how you need it" to "figure out how to make an existing system fit what you need as closely as possible".[30]

Requirements Analysis

If you are to have any hope of selecting the right system, you need to know what you want it to do. If you are an IT person, generally speaking, this means "what your business users want it to do", although technical requirements will be included as well. It will take a lot of luck for you to select a system that meets requirements if you are not careful to understand what they are.

Your requirements will guide a few things:

- Selecting the best-fit system for your use
- Developing (configuring, customizing) the system to meet those needs
- Verifying that the system meets those needs

As such, the requirements process should be oriented around the needs for its use. That is, they should be written to be *testable*. When you get to the part of the project where you verify that the system meets needs, you should be able to use those statements to help you define how to test the system.

Requirements analysis is, to some extent, a discipline unto itself. As such, there are some guidelines and best practices for it. It is far too detailed to describe fully here, but here are some qualities that a good requirements statement should meet:

Incidentally, there are different types of requirements as well. And, unfortunately, the jargon and taxonomy[31] used to describe the different requirement types can vary. The

---

[29] These are sometimes called "COTS" systems ("C" is for "Common").

[30] I have seen a variant of the "buy" approach where companies take an off-the-shelf package and then customize it so heavily that it cannot easily be patched, upgraded, or maintained. An organization can be caught by surprise later when they find that making changes will be very difficult. Essentially, they have come close to taking a "build" approach, but without consciously doing so.

[31] Taxonomy is just a fancy way of talking about the terminology and hierarchy applicable to a particular area of study. The most widely recognized taxonomic structure may be the biological classification scheme

following are a few of commonly used terms (just a few are included, in keeping with my philosophy of "keeping it simple"):

- User (or Business) requirement — Requirement stated in terms that are meaningful to the intended business users. That is, they express *what* is needed to be accomplished, without any implication of *how* it should be accomplished
- Functional requirement — A requirement statement that goes beyond *what* is needed to include some level of *how*, but without going to the level of a specification. For example, you could have a business requirement that "the system shall support configurable reports", with a functional requirement of a "management report that highlights hours worked per employee, per project, including allocated costs")
- Technical requirement — A requirement, frequently identified by IT team members, that identifies technical qualities the system must have, for example, expected transaction volume to be supported, architectural requirements, performance requirements, scalability requirements, etc.

At the time of system evaluation and selection, the requirements statements are more likely to be business requirements. Those get devolved into functional requirements as part of the definition work at the beginning of the project. For example when the vendor begins work with the organization, they will need to more specifically translate the business requirements into the functionality that will actually be built.

Going back to the point about writing requirements to be testable… Sometimes – not always – companies use the User Requirements as a guide for how business users perform User Acceptance Testing, while Functional Requirements may be used by Quality Assurance (QA) staff to do much more detailed, rigorous testing. This type of approach is particularly common with FDA-regulated pharmaceutical companies, but it is applicable generally.

One final comment before we move on to system selection approaches. There is a later section on user centered approaches to system implementation. There are two key variants on requirements approaches. One is a requirements specification approach; that is, the wording of the requirement drives the design, coding, and verification (where you hope the wording of each is simple, clear, and unambiguous). The user-centered approach includes usage definition within the requirements analysis process: the roles that will use the system, the scenarios – or use cases – under which those roles will use it, etc. The specification versus usage orientation will be discussed further in that section. Again, requirements analysis can be quite a rigorous process. It is worthwhile to have at least one person on the team who is competent in this area and pragmatic about approach.

(Species, Genus, Family, Order, Class, Phylum,…). If you want to know more, wikipedia does have information on taxonomic structures, both what they are as well as examples.

(Make sure your designated someone is not likely to go overboard creating every possible deliverable, for every imaginable future use, because you can go as deep as you want, and you can take a whole lot of time doing it, with diminishing returns.)

Mapping Unmet Needs to Candidate Systems

Going from a set of unmet business needs to a list of applications to be evaluated is sometimes straightforward. However, in certain cases, even identifying the *class* of application you should evaluate can be tricky. And, sometimes the hardest part is getting to a clear statement of needs, especially when the organization has no idea what capabilities are possible, and only knows that "there is a problem" with the way they do things today.

The clearer sense the organization has of what they actually need, the more likely that the possible solution classes will be obvious. And, the more that the unmet needs map to capabilities that are commonly automated or are automated in standard ways, the more likely the solution classes will identify themselves.

For example, consider a young manufacturing business that has managed to operate at their current manufacturing volume with disaggregated point solutions and informal systems (a financial system, an ordering system, spreadsheets to predict demand and plan and track manufacturing and inventory, etc.). If sales are up and they predict continued high growth they may recognize that they are outgrowing these systems and need to do too much manual effort to keep information synchronized across them (for example, valuing inventory, calculating COGS, etc). In this case, they may naturally consider implementing an ERP system.

As another example, I had a client that wanted to automate some of what we call the "fuzzy front end" of product development. That is, they wanted to be able to derive some knowledge and conclusions from a vast trove of information that they encounter or develop that might shed some light on new product opportunities. Product companies more typically start with a basic idea of what they want to build and for whom, and then develop product concepts and details from there. This client wanted to be able to capture the day-to-day problems their customers and potential customers we having, as learned by sales staff, support staff, executives, conferences, etc., in order to be able to identify themes and trends, and from those, potential new opportunities, new directions, and new markets for the company. At that time, there were tools that could be brought to bear in various ways, but no standard way in which that type of knowledge capture and analysis was done, so their problem in identifying candidates was a much more complex one.

If the class of candidate systems to be evaluated is not obvious, it may become more obvious if analyzed within the context of what my PRTM friends call an "Operations Framework".[32] Some examples are provided below.

[32] An operations framework created as a deliverable for a client project would likely be more detailed and rigorous than the examples used here. Since this is a conceptual approach, the simplest possible rendering to make a given point will be provided.

Example 1. Managing financial information

Example 2. Managing the product definition and development process

Example 3. Managing information in a pharmaceutical company

Example 4. Managing orders for a product company

Selection Approaches

The spectrum of approaches for selecting a best-fit system can range from:

- A particular vendor's system is selected by default due to the preferences of key decision makers (e.g., they have past experience with it and liked it, heard/read that it is better, etc.)
- A basic RFP (request for proposal) is created that highlights key requirements, and a decision is based on the response, where the response may include a vendor demonstration with business user participation and grading
- A rigorous requirements analysis is performed, and the candidate systems identified then evaluated against each requirement, with multiple scores and weights for each (for example, a separate score to measure whether the functionality is there versus whether it is easy-to-use) adding up to a vendor with a "best" value result, which then becomes the recommended system

And then, of course, there are the hybrids. For example, a rigorous, unbiased process is used to identify the best-fit system, and then the recommendation is overruled by an executive or powerful stakeholder who prefers a system that did not win the "bake-off". And, it really gets fun when you have multiple, powerful stakeholders or decision makers that *each* have already made the decision in their own mind.

But, if we go back to a point I made in an earlier section that the "right decision is the one that gets the right result", then any of these three approaches may be successful, depending on circumstances (luck being one!).

The first thing to understand is that the only way a system could *perfectly* meet the needs of your organization is if you build it internally just for your use. End even then, since all stakeholders and to-be users have their own preferences and approaches, it would still only conform to the needs and wants of the loudest, most stubborn, or most powerful constituents. Not only that, if you were to build every system you want from scratch – well, then you would be responsible for the maintenance of all that software code: bug fixes, enhancements, new feature requests, etc. It can get very expensive and resource-consuming. However, for some situations, such as when particular functionality does not exist in the market and could give you considerable competitive advantage, building the capability yourself – and keeping it to yourself – is the right thing to do.

One thing to understand… No matter what you do – your users will not be perfectly happy with the result. Even if you implement every function everyone wants, someone will have preferred that something have gotten implemented in a different way than the team chose. You are not going to implement the perfect system; someone, somewhere is going to be unhappy with something – plan on it, get over it. After all is said and done, even if you exceed all of the goals set out for the implementation, users will find fault and complain – as they should, since it will not be perfect out the gate: it will need fixes, enhancements, user interface changes, etc.

You want to select the best-fit system for your needs. This does not mean that the system that meets the most of your requirements wins. For example, if System A meets 45 requirements out of 57 and System B meets 48, this does not mean that System B "wins".

The reason is that, generally speaking, you will not install business system software *as-is*. You will take the software as it exists, configure some aspects to your needs, customize some functionality where needed, add interfaces to other systems as needed, etc. Configuration and customization require effort – typically more for the latter than the former. *Effort* translates to people needed, time needed, and cost.

Since we have not defined the difference between "configuration" and "customization", we will do so now. "Configuration" uses the native capabilities of the system to fine-tune it for use in a particular environment, for example by specifying values that should be presented in a drop-down list. "Customization" is the creation of new code to supply functionality that does not exist in the native application, or that replaces functionality in the application with code that better meets the business needs. Some systems allow fairly

sophisticated changes under the umbrella of configuration, but generally speaking, all those configurable capabilities have been tested by the vendor prior to their release, so less effort is generally involved than making a customization of the same magnitude.

For each requirement that a system must meet, it is important to know not just *whether* a system could meet it out-of-the-box, but *how* it could meet it. For example, consider the following:

	Vendor A	**Vendor B**	**Vendor C**
Requirement 1	Met	Configuration required	Met
Requirement 2	Met	Configuration required	Met
Requirement 3	Significant customization required	Customization required	Met
Requirement 4	Met	Customization required	Cannot be met
Requirement 5	Cannot be met	Configuration required	Cannot be met

Requirements logically have varying degree of importance, from must-have to nice-to-have. If Requirement 5 is a nice-to-have, then Vendor A might be the closest fit, depending on the effort of Requirement 3 on Vendor A compared to the effort to implement all requirements on Vendor B. If Requirement 5 is a must-have, then Vendor B is the only possible choice among these three candidates (in which case, you might consider whether you have the right list of candidates to start with).

Requirement 3 has the highest effort implementation on Vendor A. If Vendor A seems like the closest fit otherwise, then the organization might consider how critical that requirement really is, to gauge whether the requirement is of sufficiently critical to justify the additional time/money/resources to implement it on the Vendor A. If it is a must-have, then Vendor A could still end up being the best fit, depending on all the other factors.

Because "effort" translates to resources, time, and cost, key inputs into a selection decision are:

1) The importance of each requirement
2) *Which* requirements are met and
3) *How* each requirement is met for each system, i.e., how much effort is needed for each system to meet those requirements.

One way that you could include both the criticality of the requirement and the effort to meet it with a particular system is something like the model shown below. Three different scenarios are shown to illustrate how these factors work together. The score assignments are as follows:

Criticality	**Fit / Effort**
1=Nice-to-have	0=Requirement cannot be met
2=Important	1=Significant customization
3=Future must-have	2=Some customization/configuration
4=Must-have	3=Light customization/configuration
	4=No effort/out-of-box functionality

By the way, before we examine the three scenarios, I would like to make one point. Some mathematical purists may object to the fact that one scoring element is zero-based (Fit/Effort), while the other is not (Criticality). To those who would object, I say this: the Criticality element *is* zero-based as well. The requirements that have a criticality of zero are those that are not really needed. That is, they are not requirements and they should therefore not be in the selection criteria (but mentally add a "0=Not a requirement" legend item if it pleases you – just do not bother actually rating systems on non-requirements!). On the other hand, if a system *cannot meet a requirement*, then it makes no sense to give the system an incremental score for its fit on that requirement, which is why there is no positive value allocated for that. The scenarios are shown below:

Fit / Effort		
Vendor A	**Vendor B**	**Vendor C**
4	3	4
4	3	4
1	2	4
4	2	0
0	3	0

Legend

0=Requirement cannot be met
1=Significant customization
2=Some customization/configuration
3=Light customization/configuration
4=No effort/out-of-box functionality

Scenario 1: Candidates eliminated because they cannot meet must-have requirements
Scenario 2: System that appears to be more work is actually best fit, due to other factors
Scenario 3: The best fit <u>only</u> fits the must-have and future must-have items

Requirement	**Criticality**		
	Scenario 1	**Scenario 2**	**Scenario 3**
Requirement 1	3	3	4
Requirement 2	2	3	4
Requirement 3	3	4	3
Requirement 4	4	3	2
Requirement 5	4	3	1

Other inputs besides crit and effort? Risk?

Don't just look at number. Look at results. Slice and dice a few ways, e.g., which one comes out best if you just consider must-haves, or must-haves plus future-must-haves?

Identifying the Right Candidate Vendors

Risks of vendors
Closeness of fit

The RFP Process

An RFP, or Request for Proposal, is a document you send to candidate vendors to tell them what problems you are looking to solve, and to guide them in their response to you as to how their system(s) could provide solutions to your problems.

An RFP should specify the following:

- **Business problem.** The RFP should clearly describe what specific business problems are driving the need for a solution.
- **Business requirements.** Requirements for the system should be clearly laid out *in business terms*, i.e., defining what business problem needs solving without defining how the problem should be solved. Each vendor may have very different solutions to each requirement.
- **Expected role or structure.** In some cases, it may make most sense to make use of a particular vendor's consulting organization; in other cases, you may have a preferred vendor that you would rather perform the configuration, customization, and development work. The former is more likely if you do not have a specific consulting organization that you generally work with and trust, if the vendor's consulting organization is particularly reputable, or if the vendor's knowledge of and expertise with their system is irreplaceable. (Alternatively, an organization may have resources internally to perform this work; however, this approach seems quite rare in my experience.) In any case, you should make your expectations clear, so that they can respond appropriately with timeline and cost information.
- **Timeline.** Vendors may have very different capabilities for meeting the specified timeframe, depending on how close their solution would meet your needs, as well as the efficiency of their implementation processes. The RFP should specify the target timeframe; each vendor will respond with what they can do in what time frame. These timeframes could be different. For example, a vendor might not be able to commit to the exact time frame, but might suggest an alternate. Or, a vendor might be able to commit to slightly less within the target timeframe.
- **Cost.** Costs for a particular vendor's package may include up-front costs for the application, per-user or site license fees, consulting fees to complete the implementation, maintenance fees, etc. The vendor will provide this information; you may want to structure the information in the RFP to guide how they provide it to you.

Note that multiple RFPs may need to be generated for a single project. For example, you may have an RFP for the candidate system vendors, an RFP for management consulting

support (to plan and run the project, perform related business process work, etc.), and an RFP for the technical consulting support. Likewise, if you have a single RFP covering all those aspects, vendors and consulting firms may partner to cover the full spectrum of what you require within a single RFP response.

Putting it All Together

Prior to doing the detailed planning work, you must have accomplished the following:

- Defined the business requirements
- Selected the system to be implemented
- Selected any outside resources needed to support the project, and completed negotiations regarding the scope of their work, timeframe, cost, and terms
- Figured out who owns the project, i.e., who is going to make sure the project is set up for success and can clear any roadblocks as it proceeds – the "project champion"
- Identified any critical, irreplaceable team members and secured their commitment

PLANNING THE PROJECT

Based on the Selection Outcome
Effort not known in detail at Selection time. Info gets refined as project evolves / matures. Ergo, review/revision at forced decision gates is a form of risk management

May refresh items from Definition
Intended outcome, with examples
Measurable goals
Key requirements
Key milestones
Decision criteria? How will you manage scope? What things beyond goals can be added?
Scope definition – defining what you will not do
Phasing
Project Boundaries and scope – segue to team
Schedule

GETTING THE MANAGEMENT PROCESSES RIGHT FOR IT PROJECT SUCCESS

One of the most critical factors in IT project success is getting the project management structure right. This includes: strong sponsorship, clear decision-making processes and milestones, clear definition of the team membership and roles – and getting the right team membership, a good project manager, and good communications.

Role of a Project Sponsor

One of the key drivers for both effectiveness and efficiency in business system implementation is the ability to respond to and resolve issues as they arise. Doing so requires regular project involvement by a person who has the authority to make or drive significant decisions – financial, organizational, or directional. Some of the key elements of a sponsor role include the following:

- Respond to updates from the project team with directional feedback and input
- Respond to issues and possible solutions presented by the project team, making calls on decisions requested, possibly escalating when needed
- Escalate issues where needed – help the project team navigate the organizational hierarchy to resolve thorny cross-organization issues such as resourcing and team membership

The project sponsors act as project *champions* and key stakeholders. The project team will use the sponsor(s) as a sounding board over the course of the project. There may be a wider group of business and IT stakeholders that must receive updates and may be called upon for input on decisions. However, the sponsor team can be thought of as the "day-to-day" escalation path for the project team to use. The reason that a day-to-day escalation path is useful is that decisions get delayed if large groups of busy executives are required every time a judgment call is required that the project team is comfortable making. Significant amounts of time in a project can be wasted waiting for time with leadership decision-makers otherwise.
In some cases, they may be providing the funding for the work – at a minimum, at least one sponsor should be a decision agent for the person funding the project. Note that ultimately organizations do not fund projects – people do. Projects inevitably require trade-off decision-making, and many of those decisions have financial implications; these decisions can be significantly slowed down or impeded if the person whose budget funds the project is not directly involved and does not have a designated agent enabled to make the decisions. The right sponsors are those who can best support the project. This "support" role is frequently as a project advocate; however, if the project is not going well or needs redirection, sponsors must also be willing to make tough decisions – changing project leadership or team membership, making changes in focus, or possibly canceling the project.

The simplest projects may have a single level of project sponsorship, with one or two sponsors at that level. The management leadership for a project, however, can get significantly more complex, depending on the effort involved and number of business user stakeholder groups. For example, there could be a small set of stakeholder leaders providing day-to-day management and sponsorship – sometimes a group like this is called a steering team – while another set provides another layer of management and decision-making higher in the organization. When multiple layers of leadership are required, the roles and decision-making authority of each must be clearly defined.

A clear definition of decisions that the project team is enabled to make on their own without the sponsors or other executive management also facilitates effective decision-making, as described in the next section.

Getting the Project Team Right

A strong project team requires more than picking a few good people and letting them run with the ball. The structure of the team is very important, as is clear definition of the roles and decision making authority is also critical.

The Project Team Leader

The first place to start in setting up the team is choosing who will lead the team. Strong leadership capabilities is required; however, what makes for a good leader can be a little fuzzy. The project manager must be able to get the team to work together as a team and must be clearly viewed as the leader by the team – not just because they have that named role. For a project manager to be viewed as a leader, they must be well-respected within the organization. People will not follow someone they do not respect as a leader; while they may give the impression of compliance in order to not get in trouble, they are unlikely to fully cooperate to the degree needed.

Some of the attributes of a good project manager are the following:

- **Good meeting management.** The project manager must be able to run meetings effectively, ensuring that participants are fully engaged, listening to input and feedback, and driving issues to resolution.
- **Expert knowledge in the project subject area.** Having the project run by someone who intimately understands the business problem and what needs to be done to solve it is extraordinarily valuable. If you have someone in your key business stakeholder organization for the project who is a strong leader, they may be a good choice. If they have a strong background in business system implementation, even better. I have seen some projects run by what I call "pure project managers", that is, people who run meetings, take notes, and manage project schedules. This approach is not always effective. If the leader does not have the expertise to drive the team to meaningful decisions, they are unlikely to be either effective or really seen as leaders. If you do not have a single person who meets the criteria for expertise, consider a partnership or shared role with two experts – but make sure the relationship between the two is very clear and agreed upon.
- **Strong organization skills.** Tracking the minutiae of project progress and status can be very detailed. It requires tracking open action items and making sure they are progressing to closure, knowing the status and progress of each team member's work and how it impacts the overall schedule, keeping track of key decisions made along the way, and the context and rationale for each, and tracking progress to the overall schedule. "Big picture" people may make for good leaders, but generally make for poor project managers.
- **Decisiveness.** Start to finish, getting a project completed is all about making decisions. The typical project team meeting or discussion has two functions: reviewing progress (open action items, as well as work in process) and identifying issues. If an issue is identified, two outcomes are possible: the team discusses the issue and comes to an agreement on a solution; or, the team discusses the issue and decides that it requires escalation. Both of these outcomes *are* decisions. So, the leader must be able to drive the team to closure, including making the difficult call if one is required.
- **Ability to deal well with different types of individuals.** Getting the best work out of each team member requires different approaches for each member, personality and work style. This is particularly important when you consider the

> need to deal well with the "difficult" ones. While there may be projects out there, somewhere, that do not have a single difficult team member, I have not yet seen one. Getting even the most difficult team members to cooperate and participate fully is very important, as the difficult ones seem to frequently have skills, knowledge, or organizational leverage important to the team.

One final note on project leadership. In this discussion, we have not distinguished between a project team leader and a project manager. However, in many of my consulting engagements, these roles have sometimes been split between an employee from the key business stakeholder constituency and a consultant. The advantage of this structure is that it can be a way to get the internal business expertise needed and organizational, leadership, and system implementation expertise combined into a shared leadership role.

Project Team Membership

An effective team is one that works well together. I have seen two ineffective team approaches. In one, the people assigned to the project are those that have free time. In another, the management team attempts to assign "the best and the brightest" to the project team. There are potential pitfalls in both of those approaches.

The problem with using "available" resources is that there are typically reasons that those people have free time. The busiest people in the organization are frequently the most competent, knowledgeable, and productive. Productivity breeds a good reputation, which leads to more people then calling upon their expertise, which then makes them busier – and so on… Beware of any project team members assigned solely because they were available! If you have the right team members, you likely have good subject matter experts that would otherwise be very busy with other activities.

On the opposite side of the coin, some people who fall under the category of "the best and the brightest" may not be good team players. Team work is very important to overall project success, so you should also be wary of anyone you suspect to be a prima donna. Very effective teams have been built from sets of people deemed competent but not stars. Competence is a mandatory attribute, but the team must be able to work together. Never underestimate the value of a highly motivated team. The best team is one that has the most competent people who can effectively work together.

The team composition should reflect the organization stakeholders for the project, so that each key stakeholder group is represented in some way. This does not mean a one-to-one mapping necessarily. Depending on the size and complexity of the project and organization, one person may represent more than one organizational group – as long as they know that they must act as agent for the others, that they do in fact seek input and feedback from their constituency, and that they are truly enabled by those groups to represent them. The ideal team size is around 5-9 people[33]. However, each person on the

[33] One of my former PRTM colleagues calls this the "minivan" rule – the complete team should be able to fit in a minivan. If they cannot, the team is too big.

project team may in fact work with their own team of folks on the area for which they are responsible.[34]

Getting all the right people assigned to the project is not the end of the story. You also need to make sure that they are committed at the level needed for the project's success. That means that you need to estimate how much of their time is needed (hours per week or percent of time), and then you need to make sure that you have agreement from them and their managers that that actually will be committed to the project at that level.

Many projects have been doomed to failure – or at least, very slow success – because the bulk of the project work is performed by "part-timers". The problem with staffing a project only with part-time team members is that their regular day jobs frequently take precedence, especially when issues arise in their daily functions. Dedicated project resources is one of the critical project success factors. Not every project team member needs to be 100% committed to the project – but some do. The greater the commitment to the project, the more likely its success. One way to achieve the needed level of commitment is by determining the key participants, and then back-filling their usual job. Back-filling is hiring a temporary worker, or temporarily re-assigning another employee, to do the work typically performed by your intended team member, so that they can be freed up to focus on the project.

Project Team Decision-making

We talked above about decision-making at the sponsor level. It is also important at the project team level – if every decision has to go to a sponsor, impediments to decision-making will once again slow the project down.

[include a decision-making diag for team structure?]

ROLE OF PROCESS ANALYSIS IN SYSTEM IMPLEMENTATIONS

One thing to make sure of when you are implementing a system is that you pay close attention to the processes that will be implemented in, or will be affected by, the to-be implemented system. As a matter of fact, the business process work may be some of the most important work that you do in the project – more important even than the system selection.

Business process work is critical to IT implementations for a few key reasons. First, you cannot configure a system to be suitable for a process you do not understand. Second, system implementation is a wonderful opportunity to examine your processes, find inefficiencies, and improve them – and implement those improved processes within the system. Finally, if you do not take the opportunity, you risk embedding an inefficient process into a system, and it may then be much more difficult to improve it later.

[34] See the appendix for a list of useful information sources. The PRTM PACE methodology has very comprehensive information regarding team structure, leadership, and decision-making. Although written for product development processes, they are equally applicable to IT projects.

So let's talk a little bit about what you need to do to examine and improve business processes within the context of an IT implementation.

It is very common in larger (or even small-medium) organizations for each individual and group to understand their processes well, and maybe even some upstream and downstream touch points, but for *no one* in the entire organization to understand the overall process flow for any particular process area. This may sound like a remarkable statement, *but it is absolutely true*. Needless to say, it is very difficult to improve something you do not understand. Sometimes, a high level process diagram can be drawn by some knowledgeable person with in the organization; but, if you ask them to show the process in a form that clearly depicts not only process steps, but also the specific organizational role owning each step, and the hand-offs between roles, they are unable to specify the process in that level of detail.

However, a business process does have an inherent work flow – the flow of the process between the roles, including the hand-offs. And, system implementations frequently include workflows implementation. So, if you cannot depict the workflow, you cannot implement it correctly or effectively. An example role-oriented process diagram is shown below; this format is typically called a "swimlane diagram", as each role occupies what looks like a swim lane in the figure.

[insert swim lane diag here]

Sometimes, there may be inefficiencies in the performance of a particular step in a business process. However, the more difficult-to-identify inefficiencies may be in the overall flow and hand-offs (more difficult to identify because few people understand the big picture flow). So, doing a role-based workflow diagram will help you identify areas for improvement in your processes.

There are multiple points in the system implementation lifecycle where you will do process-oriented work. First, you will need to get enough of an understanding of what your processes-to-be-automated are to be able to provide the information for a request for proposal (RFP). An RFP is given to would-be system vendors to enable them to respond with a proposal for how their system would meet the stated requirements[35]. The RFP could state high level business requirements, or could describe the functionality needed by using a scenario-focused approach (e.g., by describing high-level flows of how the system would be used)[36].

After system selection, additional process definition work is performed to understand and document the current process at a detailed level. In the typical nomenclature, this is called "as-is process documentation". The as-is process work supports identification of inefficiencies to be addressed, as well as configuration of the system to meet process needs. Once the as-is process work is complete, attention can be turned to identifying

[35] The response would typically include cost information as well, including basic software costs, licensing, configuration, etc.
[36] More information on the RFP process is included in a later section.

how the processes will be improved, and how they are to be implemented – or "configured" – into the new system. The refined processes are typically called the "to-be" processes.[37]

Finally, it is worth a brief note to comment on the attributes that make for a good process analyst. The process analyst is the person (by whatever role / job function they typically fill) will define and document your current and to-be processes. This person will work with a (typically) broad community of business users to gather the necessary information. Whoever you designate for this function will need to be able to communicate well, take initiative at seeking people out for information, understand the information they are given, and be able to follow up with questions to gather the next level of detail. In short, they must be good communicators, detail oriented, and preferably analytically focused. They must also understand the basic nature of your business, specifically in the targeted process areas. They do not need to be stars, and they also do not need to have decades of experience with process diagrams.

One final thought to sum up this section:

> *If you are not going to bother examining your business processes, don't bother wasting your money automating them.*

Applying User-centered Concepts to IT

A basic framework for an IT system implementation lifecycle was introduced in the earlier section, "IT implementation lifecycle methodologies". In that section, we mentioned that the steps shown in the diagrams were not meant to indicate a strictly linear process. In this section, we will explore that topic further, and talk about evolving toward appropriately configured systems through iterative techniques.

The two major categories approach for system implementations – or for that matter, software development – are "waterfall" or "iterative"[38]. A waterfall model would operate something like the following:

Requirements are gathered from users and documented
- ➲ The design is established
  - ➲ The configuration / customization is performed
    - ➲ The system is tested to verified that it meets user needs and is stable
      - ➲ The system is deployed and transitioned to support / maintenance

This is to say that, once the up-front requirements information is gathered, the technical staff proceeds to design and implement the system per the requirements. Requirements specification strictly precedes design, which strictly precedes configuration, which strictly precedes testing, and the steps along the way are not revisited once completed.

---

[37] This series of process work does not have to be *strictly* linear. Work can start on one while revisions are occurring to the prior.

[38] There are a wide variety of iterative development models; however, the details are not particularly of interest here.

Work proceeds under the assumptions that the users needs will not evolve in any way, and that the documentation of the requirements is sufficient to ensure that the resulting system will meet user needs in the way they need to be met. That is, the project proceeds without close collaboration with users, and may not involve the users again until either verification or deployment.

However, waterfall approaches fail for the following reasons:

- At the beginning of the project, it is common for the users and stakeholders to have a somewhat amorphous sense of what they need. They may be able to specify high-level requirements, but may be lacking details that will emerge later
- Even when the requirements are clearly understood by those who have identified them, the wording of requirements may be read differently by different people, which can lead to incorrect configuration of the system, resulting in much later "but you said…" / "but I meant…" type of conversations between your technical staff and your user community
- It is entirely possible – and not uncommon! – to develop a system that meets requirements in an *entirely unusable way*. Just because you have met the stated requirements does not mean that what you have produced has any correlation to how users might actually want to do their work
- Sometimes, needs evolve over the course of the project, as new stakeholder organizations are uncovered, re-organizations take place, key stakeholders are added or removed – or as business conditions change
- Available technologies can also change, leading to new possibilities for implementation design or configuration
- Relying on documentation as the key communication tool can also cause miscommunication between technical groups configuring the system, such as misinterpretation of design documentation for interfacing components. A waterfall approach would typically have a later attempted integration point, which would delay the discovery of design and coding errors

Key to providing the flexibility to allow the configuration to evolve and to able to respond to a changing environment is an iterative approach. This means that you must involve your business users and key stakeholders *early*, *often*, and *throughout* the project from start to finish. A close partnership between IT and business throughout the project works much more efficiently. The sooner you discover errors in communication, the easier and cheaper they are to fix.

For example, in an iterative approach, the following series of events could occur:

- The initial Requirements are defined in order to identify a candidate list of systems to be selected
- A lightweight form of Design activity occurs in order to evaluate how each candidate system would implement a particular requirement
- A prototype Configuration of each candidate application is demonstrated to key stakeholders, and the demonstration (with other information) is used to judge each system for fit

- The system to be implemented is selected
- Requirements may be flushed out at a more detailed level
- Additional Design activity is performed to get the system configuration closer to requirements
- The prototype system goes through a second iteration and is put before users and stakeholders once more; feedback is provided, which will then go back into further design and configuration
- Additional rounds of iteration are performed until the system is fully configured

[add a picture]

There are different approaches for partnering with the business on implementation – some collaborative but requirements focused, some usage focused. Like other aspects of lifecycle methodology decisions, either type of approach can be done only superficially or to excess, while a happy medium would accomplish the essential goals at least cost. The former occurs sometimes when members of the organization do not really buy into the approach and merely give "lip service" to it. The latter occurs sometimes when the organization initially decides to fundamentally change how they involve business users and overdoes it a little, making the initial process overly complex, inflexible, and burdensome.

The diagram below shows some example deliverables that might be seen under the various approaches. The specific collection of deliverables and the names used for each vary by organization. As indicated on the diagram, the red arrows indicate where in the implementation process you are likely to discover communication issues and usability problems.

Example documents

Traditional Waterfall	**Iterative, Specification-oriented**	**Iterative, Usage-oriented**
High-level Requirements	High-level Requirements	High-level Requirements
Detailed Requirements	Detailed Requirements	Scenarios / Use Cases
		Storyboards
Design Specifications	Design Specifications	Design Specifications
	Prototypes	Prototypes
(Code)	(Code)	(Code)
Test Plans / Approach	Test Plans / Approach	Test Plans / Approach
Test Cases	Test Cases	Usage-focused Test Cases
Test Results	Test Results	Test Results
(Fixes)	(Fixes)	(Fixes)
(Final Code)	(Final Code)	(Final Code)

Red arrows show typical project step where one might find out that current approach is not useful or implements functionality differently than how the business process flows

The diagram illustrates the point that iterative methods will uncover problems earlier, with usage-focused methods likely to expose the issues sooner than non-usage-based iterative methods.

There are a variety of lifecycle methodologies that are iterative in nature, each with their own name and a slightly different take on terminology and approach. I do not think the differences between them are really all that interesting, and you could get lost in the jargon talking about them. Like a lot of IT jargon, not all of it is that valuable. The best fit methodology is the one that fits into the specific organization. Period.

Whatever implementation methodology is chosen, it must enable collaboration between technical staff and the business stakeholder community, and it must be flexible.

Two key thoughts before we leave this section:

> *Waterfalls make for nice scenery, but lousy system implementation methodologies*

And:

> *If you want to implement a usable system, consider how it will be <u>used</u> – not just what it will <u>do</u> – from the very start*

GOOD IT PROJECT COMMUNICATIONS

Who needs to receive what kind of communications
What kinds of communications are there? E.g., status, progress, issues, decisions, training

EFFECTIVE USE OF CONSULTANTS AND CONTRACTORS

Making effective use of consultants requires work
Don't let them work alone – use them as your "right hand"
Make sure internal resources are deeply involved / time-committed
Contracts are important – make sure they reflect what you need – adjust when needed
Watch the money
Contract-for-hire versus trusted adviser roles
When can you let them run with the ball
Types of consultants and what you might expect
What to look for in a consultant
How to find one?
When do you need a consultant?
Summarizing common mistakes

GETTING THE MOST OUT OF YOUR VENDORS

Iterative approach – get it worked into the upfront contract
They want your money
Avoid waterfall
Anticipate surprises / additional expenditures

POST-DEPLOYMENT ACTIVITIES – THE PROJECT IS NOT OVER AT GO-LIVE

Post-deployment issues
System goes live, poor the champagne, everybody celebrate. Poor planning for what happens next can cause a lot of problems. Adoption – inadequate training, cultivation of expert users in the ranks, support models, planning for maintenance functions, planning for enhancements, including planning for a series of post-deployment fixes, staffing to keep enhancements going, problems with dis-banding the team too quickly – transition from implementation staff to enhancement staff – who owns enhancements?
Ability to respond to requests for changes
Maintenance effort, staffing, expense sometimes underestimated, understaffed, esp in orgas that operatie more organically – new sys created w/o adequate spt planning. Pot problem with entrepreneurial innovative culture
Adoption – people used to doin things the old way – verifying and improving adioption after the fact
Adoption of support/maint processes

"IMMORTAL" SYSTEMS AND OTHER MAINTENANCE NIGHTMARES

End-of-lifing
Multiple systems for same purpose
Large sets of systems for single business/process area/function

WRAPPING UP: CRITICAL SUCCESS FACTORS FOR IT PROJECTS

MANAGING IT OPERATIONS

ASPECTS OF IT OPERATIONS

Managing infrastructure upon which biz apps and biz fcns depend:
Asset mgmt, software mgmt, network mgmt
WAN, LAN, WLAN, phone
Hardware mgmt: upgrades, support
Services mgmt: App servers, web servers, db servers
App mgmt: support, backup, maint for each system

Execution
Projects, spending, pipeline

ASSESSING THE EFFECTIVENESS OF YOUR IT ORGANIZATION

Alignment with biz/tech strategy and goals
Alignment of tech infra with tech trends and strategy
Alignment of resources with tech strategy
What's in your pipeline

Benchmarks
Level of spending for IT compared to industry peers
Availability metrics
Percent of projects completed, cancelled
Percent of projects in stages of implementation
Number of systems total
Number of systems supporting each process area
Number of projects that are strategic versus tactical, new dev vs incremental

MEASURING IT RESULTS

Eff/eff Defined – do the right thing vs do the thing right

BEST PRACTICES IN IT SUPPORT AND SYSTEM MAINTENANCE

OTHER TOPICS?

Incentives, support models, relationships with business stakeholders?
Actions to take based on measurements, assessments?
Maintenance and support processes?
Best practices?

WRAPPING UP: MEASURING AND MONITORING OPERATIONS EFFECTIVENESS

INDUSTRY CASE STUDIES

FORMAT OF THIS SECTION

- Defined
- Examples (list) of some companies that fall into this industry, with one-line description of the nature of their product(s)
- Context / nature / environment of the business/industry
- Operations framework and key information flows
- Potential areas for enablement
- Potential application categories and coverage
- Case examples from the industry: high-level approach, key decisions, etc.

MANUFACTURING INDUSTRY EXAMPLES

Defined

General observations

Plants, inventory – and space, COGS, equipment
Accounting: valuing inventory, calc'ing COGS,
Concerns:

Pharmaceutical / Biotechnology

Defined
Additional concerns: Complying with FDA regulations for product quality

Network Equipment

Defined

Semiconductor

Defined

NON-MANUFACTURING INDUSTRY EXAMPLES

Defined

General observations

Limited tangibles, little to no COGS or inventory
High R&D
High gross margin, freq higher margins overall
(Note exception: supply chain svcs)

Packaged Software

Defined
Concerns

Internet and Software Services

Defined
Concerns

Telecommunications Services

Defined
Concerns

IT TRENDS AND HOT TOPICS

POST-MORTEM ON THE DOT-COM BOOM AND BUST

What did we think would change and what did? E.g., world *is* fund different AND the old business models still apply. Still need to make more that you spend or you don't have a viable business
What's Web 2.0 / Enterprise 2.0? Same technology, but now we're smarter and wiser? Hopefully means useful IT, not just fancy stuff with no biz model

INFORMATION UBIQUITY

Applications and considerations
Increased integration and collation of existing data
    Note on uses of integrated data – census data
Integration – intra- and inter-company
More and more information is online – I hope it's safe! Privacy considerations
Usage considerations
Security

A HISTORY OF ABSTRACTION AND VIRTUALIZATION

APIs, Obj oriented programming, XML/Integ, virtualization (load bal/failover)
Less you need to know about details, more flexibility built in

SERVICE-DELIVERED IT

Hosted services
Outsourced IT services
Applications delivered as services
Utility / grid computing

TECHNOLOGY CONVERSION

Internet, phone, cable, satellite – phone, internet, television
Mobility and mobile devices – cell, GPS, music, video
Goes along with info ubiquity
See more and more

IT'S HIP TO BE GREEN

New. Is it a fad? Trend in VC, consumer buying, etc. IT? What would green IT mean?

OUTSOURCING, OFF-SHORING, AND GLOBALIZATION

Globalization, outsourcing, competitiveness, long-term trends?

Where companies are outsourcing to
Long-term impacts (e.g., India – econ boom, salaries rising, eventually will look less attractive)
Benefits and pitfalls
Strategies and CSFs

SARBANES-OXLEY AND OTHER REGULATORY TRENDS

Financial: worldcom, Adelphia, Enron scandals resulting in rqmts for execs to sign off on results, SOX requirements, options scandal, SOX feedback re: impact to smaller cos, continuing restatements for options backdating
FDA: increased scrutiny on FDA decisions => increased scrutiny on LSC cos => longer time to approval, more post-approval monitoring. Impact on LSC IT?

INCREASED RECOGNITION OF POTENTIAL THREATS AND DISRUPTION

Terrorism, mother nature, security risks
DRP, security threats

INNOVATION ISN'T INNOVATION IF EVERYBODY DOES IT

FINALLY, TRENDS I'D LIKE TO SEE

Better solutions for smaller companies – hosted, easily configured, cheaper
Implementations where business needs don't get lost in the details – implemented poorly, without partnership with business, early/often. More iterative implementation
More attention to business processes to be enabled – use as opportunity to simplify, don't bake bad process into a system where it will then be even harder to change.
Better IT leadership – biz oriented, strategic, financially astute (need for port mgr position?)
Better partnerships, understanding/common language, working relationships between IT and biz
Better consulting leadership – keep it simple, drive big ideas down into the detailed results

APPENDIX A. IT FUNDAMENTALS FOR BUSINESS USERS

The intent of this section is to help non-IT professionals to understand some fundamental information technology concepts and terms that are particularly relevant for business systems. The content of this section may also be useful for beginning IT practitioners. All others should skip over any sections that are too basic – or skip right to the next section if needed.

EVERYBODY LOVES "BUZZWORD BINGO"

Dealing with technology people – power games based on technology knowledge
Value of IT is in application of the technology – no technology for technology's sake.
Lots of technologies in search of a solution
Expect your IT people to speak intelligibly to you – don't let them lingo you. Seek out the folks that can communicate with business folks, weed out folks that don't (or keep them in back rooms)
Power games – confuse 'em with techno speak
Orig source (acc to wikipedia) – Feb 22, 1994

IT ARCHITECTURE BASICS

This section describes the key components of an IT system architecture.

User Types
There are two main user types for business systems: the business users, who use the system(s) to do their jobs; and administrative users, who manage the systems in various ways to keep it operational and configured correctly for their business users. While business users are frequently company employees in some business function or another, some systems may also have external users (e.g., customers, partners, or suppliers) as business users.

Client Software
Business users access the business system software via a *client*. Client software can range from web access, in which no software actually resides on the business user's computer, to "thin clients", where a small bit of code runs on users' computers to regular, or "fat clients", with a full set of operational software on users' computers. The type of client access has implications for upgrades and maintenance – no client-side maintenance (other

than web browsers) is required for web-only access, while there may be significant maintenance requirements for fat client access.

Servers
The core business system software usually runs on one or more *servers*. Some companies choose to run most business system software on large mainframes, which are known for their reliability and scalability. Others use application and database servers of various types. An architecture with a single, combined application and database server accessed by the clients is called a *client-server* architecture. An architecture with application servers separate from database servers is called a *three-tiered architecture*.

Networks
The users are connected to the servers via a network. This network consists of one or more local area networks (LANs), which are fairly high bandwidth co-located networks (e.g., for a single company site), connected with wide area network (WAN) links if the company's users are across multiple sites.

A very simple architecture diagram is shown below.

[diagram here]

Determining the appropriate IT architecture depends on factors such as locations of business users, expected transaction volumes, performance needs, system availability requirements, and other qualities as described in the next section.

Quality Considerations for an IT architecture

When evaluating a particular business system's fit for a particular company's needs, one aspect that is typically evaluated is the system architecture. The IT folks might talk about evaluating a given vendor's architecture with respect to qualities like the scalability, performance, and availability.

In business terms, these qualities ask the following:

- Can the system handle the number of users required, and the expected volume of business transactions?
- Can the system provide the needed response time for the required business events, particularly for time- or mission-critical events?
- Can the system provide the stability needed to be available when users need it?

Of course, answering these questions requires: that you have estimated the number of users and transactions; that you have asked the business what their response time requirements are, and which areas are mission- or time-critical; that the business has specified the target availability for the system (typically expressed as a percentage of time that the system is up). Without that information, the IT organization will only be making a guess that the system architecture *seems* sufficient.

IT TRENDS AND HOT TOPICS

POST-MORTEM ON THE DOT-COM BOOM AND BUST

What did we think would change and what did? E.g., world *is* fund different AND the old business models still apply. Still need to make more that you spend or you don't have a viable business
What's Web 2.0 / Enterprise 2.0? Same technology, but now we're smarter and wiser? Hopefully means useful IT, not just fancy stuff with no biz model

INFORMATION UBIQUITY

Applications and considerations
Increased integration and collation of existing data
    Note on uses of integrated data – census data
Integration – intra- and inter-company
More and more information is online – I hope it's safe! Privacy considerations
Usage considerations
Security

A HISTORY OF ABSTRACTION AND VIRTUALIZATION

To evaluate performance, for example, you need to think about not only how many users you expect, but where those users are located, and what kind of network links are between their locations and any servers they need to complete their business functions. For example, if your users are across many geographic sites, and some critical users are within sites that have slower network links, you will need to make sure that you choose a business system that can achieve the required performance even when operating over slow network links. This could imply a web- or thin-client approach, or some server-side optimizations to make sure time-consuming processing is not conducted over the network between the client and the server(s). Regardless of approach, you must make sure that you consider those users, and not just the ones nearest you when you are evaluating and configuring the system or you will end up with disappointed, inefficient users – and wasted time for the users, which means wasted money for the company.

Availability considerations drive not only evaluation of stability of the business system core software, but also the stability of the environment it runs on, and the capability of the system to be run redundantly. Mainframe environments are commonly viewed as very stable environments, while some IT practitioners and companies distrust some "newer", smaller server approaches such as servers running Microsoft Windows NT.

Redundancy can be used both to ensure availability as well as to deliver performance. Redundancy is also frequently used to meet the needs of a company's disaster recovery plan, i.e., the plan providing business continuity or restoring operations after a disaster, such as a natural disaster, takes down systems at a particular business site. Redundancy means that more than one system provides the same functionality. For example, there may be multiple application servers for a particular business system, as well as multiple database servers.

[define, talk about load balancing, failover]

The IT organization first talks about the architecture for a system when it is being evaluated for appropriateness to the company's needs. However, the topic is revisited when it is time to implement the system in the company's infrastructure. At this time, it is the responsibility of the IT organization to determine how to set up the system architecture – where servers will be located, type of client access to be provided, redundancy to be provided, backup needed, maintenance processes to be put in place, etc. The business community should expect to see diagrams from the IT organization that describe these decisions. A sample diagram is provided below.

[insert diagram here]

The specific format of the diagram can vary. Expect to see server locations, client access, relevance of WAN links to the architecture, any server clustering, etc. Also, later in the project, the IT organization should also provide information regarding backup, and maintenance, and support processes.

A FEW KEY DISTINCTIONS IN IT TERMINOLOGY

If you have participated in many IT projects, are in the IT organization, or are generally well-versed in the basics of IT project work, you will most likely want to skip this section. This section presents some key concepts that may help you understand some basic categories within IT: types of IT systems, the nature of different types of IT work, the most basic levels of choice within the implementation of a new system, and some key categories of personnel within IT organizations. This information may help you understand the nature of IT work at a very basic level.

Infrastructure versus Business Systems

IT infrastructure refers to the hardware and core, non-process specific services that support the company. It includes the network components and services, the phone system, and the majority of the hardware, such as computers, printers, and servers. Other core services, such as email and authentication (login) are also typically included.

Business systems refer to applications that are used to implement various business functions, such as order processing, billing, financial reporting, inventory management, manufacturing support, planning, decision support, and many, many other areas.

These basic divisions often correlate to the structure of the IT organization.

IT TRENDS AND HOT TOPICS

POST-MORTEM ON THE DOT-COM BOOM AND BUST

What did we think would change and what did? E.g., world *is* fund different AND the old business models still apply. Still need to make more that you spend or you don't have a viable business
What's Web 2.0 / Enterprise 2.0? Same technology, but now we're smarter and wiser? Hopefully means useful IT, not just fancy stuff with no biz model

INFORMATION UBIQUITY

Applications and considerations
Increased integration and collation of existing data
    Note on uses of integrated data – census data
Integration – intra- and inter-company
More and more information is online – I hope it's safe! Privacy considerations
Usage considerations
Security

A HISTORY OF ABSTRACTION AND VIRTUALIZATION

Projects versus Maintenance

A project is a team effort to implement a new business system, to upgrade an existing business system, or to add major new functionality to an existing business system. An effort of this type typically requires a cross-functional team (which would include IT and at least one business user group), as well as formal project management and decision processes.

Maintenance refers to ongoing activity after a business system is implemented. It includes maintenance activity performed for the business by the IT staff, such as backup, patch application, reboots, and other maintenance needs. However, it also includes small enhancements which may be made by either the business or IT, such as new or modified reports, small UI changes, and small functional changes. The line between project and maintenance work can be somewhat blurry in the case of enhancements; however, small enhancements that can be successfully handled informally probably are more efficiently implemented as maintenance work, rather than requiring a formal team. Another way to look at this is to say that maintenance work uses very small, focused, somewhat informal project teams and project processes.

If a company is finding that its maintenance processes are too informal, leading to mistakes and system instability, it may benefit from formalizing how maintenance work is performed and adding additional checkpoints or decision processes.

Buy versus Build

One decision an organization must make once it has decided that new functionality is required is whether that functionality will be built internally, or purchased in the form of a third-party vendor's application that is generally available for purchase. A third-party application is sometimes called a COTS (common, off-the-shelf) system.

The buy versus build choice is not as clearly black and white as you might think. Even if a third party application is purchased (e.g., a customer relationship management system), it will most likely not meet all of the company's needs in exactly the way they need to be met. If not, the system may need to be configured or customized, as described in the following section.

If a company's needs are particularly unique, the company may do better to build a special-purpose application themselves. The downside to this approach is that the company must then maintain the developed code base, so the support requirements *may* be greater and more expensive than a COTS package.

Configuration, Customization, and Development

If an off-the-shelf package is selected for implementation, it must be configured for use for the specific company's needs and processes. Depending on the fit between the company's processes and the application, that may require configuration, customization, or potentially new development.

Configuration refers to using the built-in capabilities of the system to set it up for the company's use. For example, the values shown in drop-down menus in a user interface are commonly configured items.

Customization refers to writing software programming code to implement specific functionality customized to the company's needs. Customization is used when the application does not provide the ability to configure the system as needed – for example, a custom screen may be created in the user interface for a company-specific process step that does not exist in the standard process enabled by the application.

Development is like customization at a larger scale, such as creation of a large section of code, a sub-system, or interface, which does not exist in the standard application. Some people may use customization or development interchangeably, as both entail new programming code.

The relevance of these three methods of tailoring an application for a company's use is that it both reflects the closeness of fit of the application to the business needs, as well as the expense needed to configure the system. For example, a package that requires mostly configuration with a little customization will likely be a closer fit, with a smaller

IT TRENDS AND HOT TOPICS

POST-MORTEM ON THE DOT-COM BOOM AND BUST

What did we think would change and what did? E.g., world *is* fund different AND the old business models still apply. Still need to make more that you spend or you don't have a viable business
What's Web 2.0 / Enterprise 2.0? Same technology, but now we're smarter and wiser? Hopefully means useful IT, not just fancy stuff with no biz model

INFORMATION UBIQUITY

Applications and considerations
Increased integration and collation of existing data
    Note on uses of integrated data – census data
Integration – intra- and inter-company
More and more information is online – I hope it's safe! Privacy considerations
Usage considerations
Security

A HISTORY OF ABSTRACTION AND VIRTUALIZATION

configuration cost, than an application that requires a large amount of custom code. The implementation may require more time as well.

However, some systems are so configurable that they can do almost anything, and as a matter of fact are more of a template from which to start than a true off-the-shelf package – in this case, there may be a significant amount of work to form the system to meet the needs. In this case, configuration is not necessarily less complex or expensive than customization. Choosing which portions to implement and how to configure them to meet the needs of business processes can be complex.

Before leaving this particular topic it is worthwhile to note that companies generally should not just implement processes as-is into a new system, for two reasons. First, making simple process changes, if possible, can be much cheaper and faster to implement than completely reworking an application to do something it is not meant for, particularly if there is no particular reason the process has to be done exactly that way. Second, a business system implementation is a great time to examine and improve business processes, not only because it can make them more efficient, but also because the worst thing to do is to "bake" bad processes into a system and make them that much harder to improve later. To implement a system without looking to improve business processes is a

recipe for wasteful IT spending. So, when you are trying to find the best-fit application for your business, don't assume it will always be the application changing to make the fit.

BUSINESS SYSTEMS SUPPORT STRUCTURES

The purpose of this section is to help business users understand better how a business system is supported by the IT organization. Business users do not always realize the complexity behind the scenes inherent in supporting a system.

Helpdesk, network admin, telecom, dba, system admin, system support, analyst, developer, test/QA. Levels of support. Expert users.

Business users do not always realize that IT organizations are frequently organized – sometimes in a very fractured way – along these divisions. For example, network support, server and database support, and business support may be provided by different groups, potentially with different reporting structure, goals, and incentives as well. Business users may pick up the phone to report a business system problem of unknown origin, expecting "IT" to provide seamless support regardless of the source of the problem – as they should! – and not realizing that there is not a single IT group, and IT organizations sometimes have trouble coordinating among themselves. This issue can lead to poor support of the business by IT.

KEEPING YOUR IT FOLKS HONEST

Top Ten list of what to ask, e.g., How will you make sure that users in other sites (than where the servers are) will have adequate performance? How will it be architected for that? How will you verify it?

Tips and tricks for dealing with 'em?
Arch, Support (struct/SLA), Deployment

THE MANY FACES OF INTEGRATION AND ABSTRACTION

Exs: government initiatives, CAPPS2, civil liberties issues. Past history – census data DW, EAI, manual interfaces, XML, EDI, web applets (what's the word I'm looking for here)

Common definitions of key data: customer, etc. How typically applied.

COMMON IT TERMS TO HELP YOU CONVERSE WITH YOUR IT FRIENDS

Term	Definition

IT TRENDS AND HOT TOPICS

POST-MORTEM ON THE DOT-COM BOOM AND BUST

What did we think would change and what did? E.g., world *is* fund different AND the old business models still apply. Still need to make more that you spend or you don't have a viable business
What's Web 2.0 / Enterprise 2.0? Same technology, but now we're smarter and wiser? Hopefully means useful IT, not just fancy stuff with no biz model

INFORMATION UBIQUITY

Applications and considerations
Increased integration and collation of existing data
    Note on uses of integrated data – census data
Integration – intra- and inter-company
More and more information is online – I hope it's safe! Privacy considerations
Usage considerations
Security

A HISTORY OF ABSTRACTION AND VIRTUALIZATION

APPENDIX B. FINANCE FUNDAMENTALS FOR IT MANAGERS

Source for this section is Brigham / Gapensky
Conceptual approach – no formulas

Expected value
Risk/Reward
Time value of money
Risk premium
ROI
Benefit/cost ratio (profitability index – PI). PV(benefits)/PV(costs)
Cost of capital = hurdle rate. More information on what goes into it.
NPV – Present value of future cash flows, discounted by cost of capital. If NPV negative, reject. If two projects mut excl, pick one with higher NPV. problem with using for project valuation – do not assume constant cost of capital for all projects. Should evaluate riskier projects with a higher cost of capital. NPV of 0 means that project cash flows sufficient to provide required rate of return and repay invested capital. (Lot goes into determination of the cost of capital). NPV better approach than IRR or PI. IRR assumes reinv at IRR, NPV at cost of capital. Bus execs (according to source, dated 1997) prefer IRR over NPV 3-1.
IRR – in effect an expected rate of return. Rate of return for which NPV is zero. If it takes a higher rate of return to get NPV=0, than IRR is greater than the cost of capital / hurdle rate. If greater than hurdle rate, accept. Link to shareholder wealth: if IRR greater than cost of capital for the firm, increases shareholder wealth; negative – decreases
Modified IRR (MIRR): rate of return where present value of all outflows equals present value of inflows. Present values discounted by cost of capital, and inflows assumed to be reinvested at cost of capital.
Payback period – number of years to recoup a particular investment. Could do discounted payback using each's cost of capital, using higher ones for riskier projects.
Efficient Market Theory
Opportunity Cost
Portfolio Asset Management
Other project valuation methods

APPENDIX C. ANALYTICAL TECHNIQUES AND FITNESS FOR USE

Scorecards
Used for: system selection, project selection, technology selection – identifying the "best" out of a set of candidates
Weights, scores, output
Guidelines for use

Generally managed as spreadsheets with criteria, weights, and values such that each observed value, with weight (relative importance) are multiplied and then added up to a total score

Process Analysis

Bubble charts

FITNESS FOR USE SUMMARY

Tool	Applications
Scorecard	Measuring performance results of an IT organization along a number of dimensions Making *selection* decisions, such as project selection, system selection, technology selection
Simple Metrics	
Bubble Chart	
Process Diagram	
Pie Chart	
Bar Chart	

CAVEATS AND CAUTIONS

Devil in the details
Don't make precision > accuracy
Make your analysts explain the methodology – don't assume it's right. Should be explainable / summarizable at a biz level. If not, probably too complex or wrong. If analyst is unable to explain a methodology at a level that the key stakeholders cannot understand, provide feedback, nod approval – could be wrong, or is too complex, or you have the wrong analyst. Need someone both comfortable in the details as well as well-grounded in the business drivers for the methodology

GLOSSARY

User-centered Design	
Storyboard	
Scenario	
Use case	
Lifecycle	
Vendor	
Consultant	
Usage models	
Outsourcing	
Off-shoring	
Globalization	
Requirements	
Design	
Test Case	
Configuration	
Customization	
Development	
Implementation	
Project	
Maintenance	
Suppport	

FOR MORE INFORMATION…

PRTM

References:
Financial Management Theory and Practice, Brigham & Gapensky
PACE books
Others?

User centered design
Requirements analysis
PACE
IT strategy
Portfolio mgmt
Etc.

INDEX