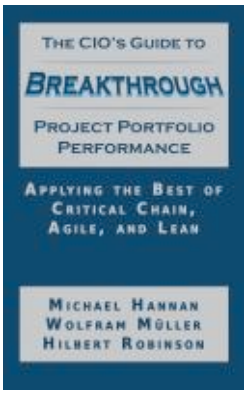
The CIO's Guide to Breakthrough Project Portfolio Performance

Applying the Best of Critical Chain, Agile, and Lean

Michael Hannan
Wolfram Müller
Hilbert Robinson



This book is for senior executives who are on a mission to jack up the performance of their information-technology (IT) project portfolios, and who are no longer satisfied that established "best practices" are sufficient to achieve their organizations' business objectives. The authors cut through the confusion and zealotry of leading improvement approaches, and distill them down to a practical set of specific techniques you can apply for maximum benefit to your IT project portfolio.

The CIO's Guide to Breakthrough Project Portfolio Management

The CIO's Guide to

Breakthrough

Project Portfolio Performance

Applying the Best of Critical Chain, Agile, and Lean

Michael Hannan

Wolfram Müller

Hilbert Robinson

First Printing: 2014

www.FortezzaConsulting.com

This title is available on www.Amazon.com and www.BookLocker.com. Bulk discounts are available for corporations, associations, libraries, educators, and others through www.BookLocker.com.

Special promotions may be offered by the publisher from time to time—consult www.FortezzaConsulting.com for current promotions.

ISBN 978-1-63443-943-5 (hard cover)
ISBN 978-1-63443-942-8 (paperback)

ISBNs listed in Books in Print under Primedia E-launch LLC

Printed in the United States of America

Purpose of this Book

Our driving passion is helping organizations achieve breakthrough improvements in the performance of their project portfolios. For this book, we focus that passion on the uniquely wonderful challenges and opportunities inherent in information technology (IT) project portfolios.

The failure rate of IT projects has remained stubbornly high, in spite of a host of methods offering promise of relief—to the point that very few project stakeholders actually expect IT project portfolios to deliver a high volume of reliable project completions. And even when we do find ways to deliver some IT projects faster and more reliably, that success has proved difficult to replicate across the portfolio. "One-size-fits-all" approaches—such as mandating that all projects adopt Agile—often produce a hit-or-miss track record, leaving chief executives scratching their heads wondering why some projects realize great success, while others do not.

Exacerbating these throughput and reliability problems is the general sense that many projects in our portfolios, even when they come in as planned, fail to deliver anywhere close to the level of organizational

benefit originally envisioned. Sometimes this is because of over-inflated promises made in order to obtain approval, but the more fundamental root cause is simply that most organizations have misguided project selection processes.

When we talk about breakthrough portfolio-wide improvements, we mean selecting much higher-impact projects, at least *doubling* the number of them that your organization can complete, and being able to deliver *over 90 percent* of them within plan—all within existing resource constraints. For profit-seeking enterprises, this translates to an effective return on investment (ROI) that is at least five times current levels, which in turn can multiply project-driven profits by 10X or better. For organizations measuring impact in terms other than profit, the breakthrough potential is commensurate.

As you might expect, achieving such dramatic results requires big changes—in how we work, in how we manage work, in our organizational values, and in how we foster unity of purpose at all levels and build trust with all internal and external stakeholders. But we have demonstrated that it can be done, and this book will show you how to do it.

Chapter 1: The Three Most Important Objectives

The three most important objectives for any project portfolio are

1) Selecting the right projects.
2) Maximizing the portfolio's throughput of project completions.
3) Optimizing the portfolio's reliability of project completions.

To many CIOs and IT project portfolio management (PPM) practitioners, focusing on these three objectives might seem obvious—of course we all want to pick high-impact projects and deliver a healthy volume of them reliably. The problem is that most of us haven't quite figured out how to do it.

Just how bad is this problem? The Project Management Institute (PMI) does an annual survey of over 2,500 project-management leaders and practitioners from all over the world, and asks them to assess their organizations' project-management

performance.[1] Here are a few of the more telling metrics from the most recent survey:

- **Project selection:** Only 42 percent of projects were classified as having "high alignment" to organizational strategy.

- **Portfolio throughput:** Only 9 percent of respondents consider their organizations "excellent" at executing their highest-priority projects.

- **Portfolio reliability:** Only 17 percent of respondents believe that their organizations are able to realize envisioned project benefits with "high maturity."

Compounding this sad state of affairs is that most PPM solution approaches focus on only a part of the problem, or on addressing symptoms. For example, because portfolio reliability is so poor, projects going over budget must either siphon funds from other projects, or be killed or de-scoped to free up funds for

[1] *PMI's Pulse of the Profession: The High Cost of Low Performance*, 2014 (available free of charge at www.pmi.org/pulse).

other over-budget projects. As a result, the focus drifts from the root problem of poor reliability towards coming up with better approaches for deciding which projects to kill.

It's gotten so bad that even some of the largest-scale, most visible attempts to improve the performance of IT project portfolios fail to even mention throughput or reliability, while pointing to "savings" from killing or drastically de-scoping worthwhile projects. For example, the U.S. Federal Government launched an effort in 2010 to help improve the performance of its $60 billion-per-year IT project portfolio. A year later, the Federal CIO proudly cited almost $1 billion in cost reductions achieved, which sounds like a pretty good start. However, it turns out that nearly half of these cost reductions resulted from one-time data-center consolidations, and about a third came from terminating, halting, or de-scoping projects. Just $30M of the savings, a mere 0.5 percent of the Federal IT project portfolio, was attributed to "accelerated delivery."

We've seen similar patterns in dozens of IT project portfolios, across many major industries, all over the U.S., Europe, and Asia. The problem is large and pervasive, so we'll start by assessing each of the three

primary objectives in turn. The chapters that follow will then provide specific guidance, techniques, and approaches for how to improve each of them.

Project Selection

While the PMI survey mentioned previously reveals poor project selection across many project-centric industries, many IT executives we speak with maintain that their project-selection processes are at least adequate. They say things like, "Project selection isn't rocket science—there are always mandatory initiatives to address security or compliance requirements; there is almost always at least one critical modernization initiative that should have been tackled years ago; there is usually a meaningful number of projects directly aligned with the CEO's strategic initiatives; and if there's any funding left over, it's not all that hard for each business unit to identify and advocate for its own top-priority initiatives."

In some ways, these statements resonate with us. After all, IT has become so critical to so many aspects of organizational performance, it seems there's no shortage of "must-do" projects—so maybe it's not so hard to pick the right projects if the only ones that fit within budget constraints are the must-do's. But if this logic were sound, we wouldn't see so many instances

of IT project portfolios that fail to enable or improve our organizations' most critical business processes.

To cite one of the more sobering examples, the U.S. Department of Veterans Affairs' IT project portfolio has now grown to about $4 billion annually; yet 400,000 veterans' disability claims remain stuck in a queue for more than four months, and the IT system that manages the scheduling of patient visits recently showed 57,000 veterans waiting more than three months for a first appointment—so long, that some veterans *died* waiting. Of course, multiple factors contribute to such poor performance, well beyond inadequate IT project selection, but when such generously funded IT project portfolios fail to improve processes that are so central to the organization's mission, we must conclude that poor IT project selection remains a significant part of the problem.

Portfolio Throughput and Reliability

Once the right projects are selected, we then look at how well the portfolio executes them—specifically, how well the portfolio maximizes the throughput of project completions, while optimizing the percentage of projects that are delivered within planning constraints (scope, schedule, and budget). Because the throughput and reliability of project completions are

often closely related, we will address them in tandem in this introductory chapter.

The PMI survey mentioned previously posted some bleak metrics on throughput and reliability; and given that these are measures of planning and execution performance, let's look first at the IT profession's most prevalent efforts to address project planning and execution. Two of these are the CMMI Institute's Capability Maturity Model Integration (CMMI), and PMI's similarly pervasive Project Management Body of Knowledge (PMBOK) and associated standards and certifications. CMMI focuses on improving the underlying processes required for successful IT project delivery, while PMI provides a set of foundational and "generally recognized good practices" that it reinforces via its certifications.

It would be logical to presume that using CMMI to improve IT project processes, and using PMI standards to promote good project practices, would result in improved throughput and reliability for IT project portfolios. As the PMI's own survey indicates, however, this is not often the case—and there are at least two primary reasons why. First, there is no mention of portfolio throughput or reliability in either CMMI or the PMBOK (or even in PMI's Standard for

Portfolio Management), let alone a maturity model to guide their improvement. Second, there is significant emphasis on what we would consider "input metrics"—such as repeatable processes and practices—without corresponding outcome metrics to assess whether this repeatability actually helps improve throughput or reliability.

To be fair, neither the CMMI Institute nor PMI claim to offer a specific set of methods or techniques for achieving a targeted set of results, and both would likely agree that any disciplined approach that helps improve the performance of IT project portfolios is welcome. Our purpose in mentioning them is not to criticize, but to make the point that, if your goal is to improve portfolio throughput and reliability, you will need to do more than achieve a certain CMMI-DEV maturity level or adhere to PMI's PMBOK and associated standards.

Agile represents another increasingly widespread attempt at improving the planning and execution of IT projects, and is arguably the most successful to date. If you have adopted Agile methods with any success in your IT project portfolio, you might reasonably point to improvements in throughput and reliability, and we commend you for achieving results that have eluded so

many of your peers. Too often, however, Agile successes are isolated on just a few projects, or seem to follow a puzzling "hit-or-miss" pattern of impressive results on some projects, and minimal improvements on others. In addition, far too many attempts to scale Agile to every project in the portfolio have failed to deliver the dramatic improvements sometimes experienced at the project level. In some cases, the failure of "scaled" Agile adoption has been so visible and so complete that Agile is abandoned completely. This is unfortunate—we believe that Agile can play an important role in helping to improve the performance of IT project portfolios, but not without some adjustments, and only when placed into a disciplined portfolio management construct designed to maximize throughput and reliability.

To many Agile zealots, this may sound like heresy. "Adjustments? No need to fix something that ain't broke!" "Disciplined management construct? Agile works because it's self-managing!" We understand where this passion comes from, and while we enthusiastically embrace Agile and its contributions to throughput and reliability improvements, we cannot yet declare victory. The IT world remains plagued with failing project portfolios—including those with Agile projects—and too many smart, capable professionals

are set up for failure. What's desperately needed is a practical mix of proven techniques and approaches, blended harmoniously into a simple, coherent, "best-tool-for-the-job" framework.

Chapter 2: How to Select the Right Projects

Let's start by looking at a reasonably mature project selection process that closely resembles many that we've come across:

1) **Project Identification**—Communicate the organizational strategy, and then ask each business unit to submit its top 10 project candidates, along with summary descriptions that include level of alignment with strategy, and some cost/benefit info for each.

2) **Project Validation**—Perform some high-level validation of the full list of project candidates, throwing out those candidates that don't meet minimum criteria, that seem only minimally compelling to the business as a whole, or that lack the strong political backing necessary to be considered seriously.

3) **Project Prioritization**—Convene an investment review board (IRB) comprised of top-echelon executives from across the organization, and have the IRB rack and stack the project candidates and

place them in priority order, using a weighted-criteria ranking approach.

4) **Project Selection**—Draw a line below the lowest-priority project candidate that still fits within the overall PMO budget, fund all projects above that line, and list the remaining below-the-line projects as "future candidates" (or just remove them from consideration altogether).

5) **Portfolio Politicking**—This has been going on the entire time, of course, but often gets particularly intense just before decisions are locked in. Watch the political wheeling and dealing kick into high gear, resulting in a "political peanut-butter spread" that avoids favoring or neglecting any business unit excessively, but makes sure to take special care of the most powerful business units.

There's a lot to like about this project selection process—it shows some discipline and alignment with organizational priorities, it fits within budget constraints, it's inclusive and tries to be fair, and it's flexible enough to bend to political realities. You might even wish your PMO's project selection process was as good as this one. However, it's also almost

guaranteed to give the organization a mish-mash of middling results.

A real-world example might help illustrate why. One of the authors (Mike) had a client whose IT project selection process looked a lot like this—it had taken a lot of leadership initiative and trial-and-error to mature their process to this level, and they were justifiably proud of how far they'd come. Whereas not much of a process at all had been in place previously—they essentially just had Step 5 (the political wheeling and dealing)—now they could claim a mature, functional project selection process.

To their surprise, however, this new process resulted in an IT project portfolio that delivered essentially the same benefits and ROI as the old process. They asked Mike to take a look at their list of newly approved projects, and offer any insights he might have on ways to boost ROI. The list looked reasonable enough, and Mike couldn't take much issue with the merits of any project in particular. So he began by asking a simple question:

Mike: "What projects have the greatest potential to deliver the highest impact to overall organizational performance?"

Client: ”I’d say Projects 1, 2, and 3…the ones that made it to the top of the priority list.”

Mike: ”OK, but Project 1 is for Business Unit 1, Project 2 is for Business Unit 2, and Project 3 is for Business Unit 3. Are there any projects that have the potential to deliver high impact to multiple business units?”

Client: ”Yes, we have a few infrastructure upgrade projects that will benefit all business units, and a few projects that were actually proposed jointly by multiple business units.”

Mike: ”OK, but if you could point to a single business process that, if significantly improved, might deliver enormous benefit to the entire business, what would that business process be?”

Client: ”Well, that would have to be the procurement process—we contract out a large portion of our budget, so any improvement in procurement would have immediate impact everywhere, and in a way that is central to our mission. But no one wants to touch procurement, because of all the complex regulatory requirements and mystifying legal issues—in fact, we’ve tried to improve procurement processes in the past, and gotten nowhere. Plus, our head of

procurement has no real power base—it's the revenue-generating business units that have the power around here, not cost centers like procurement. And on top of all that, the head of procurement isn't a believer in IT, and hasn't even submitted any project proposals for consideration."

Mike: "OK, so you're saying that delivering high-impact results requires dealing with complexity, succeeding where others have failed, overcoming political hurdles, and taking leadership initiative when others won't?"

Client: "Point taken—OK, so how do we improve our project selection process?"

Our purpose in sharing this story is to show that, even with a mature project selection process, organizations often miss high-impact project candidates. In addition, this process lacks the discipline necessary to rank candidates according to their true ability to drive dramatic improvement in overall business performance. In the above story, it was obvious where such improvement was possible, but whether obvious or not, there is a discipline to identifying such high-impact processes.

This discipline comes from the Theory of Constraints (TOC), and its logic is straightforward: In any system, there is one function, resource, process area, or process step that constrains the entire system's ability to deliver on its mission. In the above story, that constraint is procurement. Once an organization has identified its system constraint, it knows that any improvement anywhere other than at the constraint will have little or no impact on overall organizational effectiveness. Putting this concept into practice helps provide much-needed clarity on where to focus improvement efforts.

Let's provide a more concrete example to demonstrate how to apply this TOC logic. Imagine that we have three software-development projects that we're trying to choose between, and that once the projects are complete and the software is put into operation, we've estimated the following expected revenues and costs:

Table 1: Project Selection Considering Profit

	Project X	Project Y	Project Z
Additional revenue per month	$90,000	$90,000	$100,000
Additional cost per month[2]	$45,000	$45,000	$40,000
Additional profit per month	$45,000	$45,000	$60,000

At first glance, it seems clear that Project Z is the best option of the three—it has both the highest revenue expectation, and the lowest expected cost. However, we have yet to factor in our system constraint—what if we only have 30 days' worth of work hours per month at our constraint to devote to whatever mix of projects is selected? In that case, Project Z uses up all available "constraint units," and delivers a monthly profit per constraint unit of $60,000. Projects X and Y are both lower profit per month ($45K each), but each requires only half of the available constraint units, affording us the option to choose both of them in lieu of Project Z. The result is a monthly profit of $90,000, or *50 percent*

[2] This consists of total variable cost, plus the fully amortized total project investment cost.

higher than Project Z. Even if we were to double our available constraint units, we will want to fund as many project candidates like X and Y as we can, before it makes sense for us even to consider Z.

Table 2: Project Selection Considering Profit Per Constraint Unit

	Project X	Project Y	Project Z
Days of constraint time required per month	15 days	15 days	30 days
Additional monthly profit (throughput[3]), per month of constraint time	$90,000	$90,000	$60,000

For organizations that do not measure their mission success in profit dollars, we can substitute whatever metric makes the most sense, and the approach works the same. So, whether we want to measure the throughput of profit dollars, or the throughput of

[3] Note that we are equating "additional monthly profit" with "throughput." This is because we are assuming that any operating expense is incorporated into total variable cost.

veterans receiving quality medical care, we can just use the term 'throughput' to cover all bases.[4]

This gives us a simple formula to start with:

Throughput per Constraint Unit (T/CU).

One way to think of T/CU is in terms of "effective throughput," as it represents what we actually expect to achieve, given what we know about how our system constraint limits throughput.[5] I simply need to get defensible estimates of T/CU for each project candidate, and fund the highest-scoring ones for which I have budget and available CUs to support.

Factoring in Investment

For simplicity, the above example assumes that all project investment funds are fully amortized into total variable cost. While this is becoming more of an option through software-as-a-service (SaaS) offerings, most IT project portfolios require a substantial pool of

[4] Not to be confused with "project throughput," which we use throughout this book to indicate the rate at which projects are completed.

[5] Effective throughput is sometimes referred to as "octane level" in TOC literature.

investment funds. As a result, we must incorporate investment into our project-selection model.

Table 3: Project Selection Considering "Effective ROI"

	Project X	Project Y	Project Z
Additional revenue per month	$90,000	$90,000	$100,000
Additional cost per month (total variable cost)	$30,000	$30,000	$30,000
Additional profit per month (throughput)	$60,000	$60,000	$70,000
Days of constraint time required per month	15 days	15 days	30 days
Additional monthly profit (throughput), per month of constraint time (T/CU)	$120,000	$120,000	$70,000
Total investment required to deliver project into operation	$900,000	$900,000	$600,000
Expected life-span of software system	5 years	5 years	5 years
Return on Investment (ROI) over life-span of software system	$3.6M/$900K, or 4x	$3.6M/$900K, or 4x	$4.2M/$600K, or 7x
T/CU per $1,000 invested	133	133	117

We've kept expected revenue and days of constraint time the same for all three projects, but moved the portion of total variable cost that was amortized investment down to the "total investment" line, in a single lump sum, and deliberately showed Projects X and Y as requiring significantly higher investment levels than Project Z. Keeping the expected life span for all three IT systems the same (five years), we then calculate the new monthly profit (throughput) figures that result, as well as the ROI and "T/CU per $1,000 invested" metrics.

As before, Project Z looks like the best project when considering its profitability, the smaller investment level required to fund the project, and the resulting high ROI. However, when considering Effective Throughput (T/CU) per dollar amount invested, Projects X and Y still score higher. We now have a somewhat improved project-selection metric that we'll call "Effective ROI," as it calculates the actual ROI expected when taking into account the system constraint:[6]

[6] Note that T/CU/I is mathematically equivalent to T/I/CU. For those readers familiar with the notion of "investment turns," T/I is the throughput accounting formula for investment turns, so T/I/CU can also be thought of as "investment turns per constraint unit."

Throughput per Constraint Unit,
per Investment (T/CU/I)

To be even more accurate, we would need to refine the metric further by factoring in how long those investment dollars are tied up in project work, and incorporating the time value of money for the entire model. However, try to resist the temptation to over-engineer these calculations, as you will very quickly see the "law of diminishing returns" come into play.

Incorporating Strategic Value

While Effective ROI provides a focused method for high-impact project selection, strategic considerations should often be factored in as well. For example, some IT projects are designed to help improve brand awareness, test out the effectiveness of a new marketing campaign or new solution offering, gain market share, build customer loyalty, and so on. In order to promote a consistent "apples-to-apples" framework for assessing competing project candidates, we must convert any "strategic value" component into a rough equivalent of revenue. In addition, if there are any "non-dollar costs" such as the risk of harming brand value, losing market share, or alienating customers, these should be factored in as well—either

by deflating the converted revenue figure, or by inflating the total variable cost figure.

Some of our clients have also stressed the importance of showing a variety of "portfolio views" of their candidate projects, to depict the spread of candidate projects across business units, or the mix of strategic vs. profit-making projects, or the mix of candidate projects by product line. These are fine—our only words of caution are to make sure these breakouts don't lead you into stovepipe thinking and practices. In other words, never lose the discipline of examining the entire portfolio of candidate projects according to Effective ROI.

IT Projects That Expand Capacity at the Constraint

Typically, the first few IT projects that go into production operation can take full advantage of available CUs, especially as the organization learns to expose hidden capacity by focusing all efforts on maximizing throughput at the constraint. At some point, however, most or all of this hidden capacity will get used up, such that any further projects delivering new capabilities into operation will only serve to overload the constraint, degrading throughput. As a result, the only projects that make sense at that point

are those that can actually expand capacity at the constraint. Note that our T/CU/I formula still applies, but it is critical that the organization understand which projects are designed to exploit available CUs, and which are designed to expand CUs—and then sequence projects accordingly.

Special Consideration: When IT Is The System Constraint

For many organizations, IT has become so critical to so many facets of the organization, that IT itself has become the system constraint. Sometimes the problem may manifest itself as the entire function of IT constraining overall organizational throughput, but more often it's focused on a select few IT staff resources—typically senior technical architects or developers, expert on multiple critical IT systems and technologies, who are also highly effective troubleshooters. To personify this resource type, let's call her "Susan" (though there may well be more than just one "Susan"). Beyond the management challenges of how to avoid spreading Susan too thin, how to avoid multi-tasking her, and how to focus her on key organizational priorities (addressed in Chapters 3 and 4), this scenario presents some tricky problems for project selection.

As an example, consider the following. We have a candidate software-development project that ranks first in Effective ROI, and once delivered into production operation, will require a substantial level of constraint time from Susan. In addition, it turns out that the project itself also requires considerable time from Susan. If this project is the first one in the queue, these demands may present no issue at all, as Susan first works the project to completion, and then devotes her time as "constraint unit" once the software system goes into operation. But what about all ensuing projects? Their Effective ROI must take into account that the available constraint units (Susan) will be far less available for their operations once Project #1 goes live. Magnify this problem further once Project #2, #3, #4, and so on all enter the picture: All will require more of Susan, both for development and for operation. If the projects are not managed well, the result will be that the organization will nearly grind itself to a halt. Project work and operations work will both find that there just aren't enough Susans to go around. All original estimates of T/CU/I end up way off in the midst of this compounded CU shortage, projects take significantly longer than initially estimated, and operational effectiveness is severely degraded.

The solution to this problem has three parts:

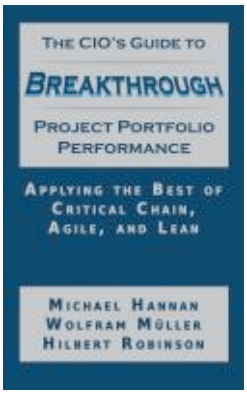
1) **Keep Susan focused.** Protect her from ad-hoc tasking, while maximizing single-task focus that is aligned with organizational priorities. Make clear that her priority is no longer responding to fires, but staying focused on executing the assigned task at hand. Schedule her resource to take on project tasks according to a given number of available hours per day or per week, and operations tasks for the remainder of her time.

2) **Subordinate all other resources to Susan**. In other words, all resources other than Susan (non-CUs) must do whatever they can to help alleviate the pressure on Susan. Even minor assistance can have a big impact—we've even seen examples of organizations asking non-CUs to go bring Susan her lunch so she can maximize her available CU time. Even better is when non-CUs shadow Susan and document some of her more repeatable approaches, such as how best to troubleshoot a particular system; oftentimes, the non-CUs even find ways to automate or simplify these approaches, further freeing up Susan.

3) **Generate more Susans**. While this may well take more time and effort—and will likely require even more of Susan's CUs initially—it simply must be done. For example, there must be deliberate efforts to have non-CUs pick up knowledge or skills that only Susan currently has, such as gaining expertise in a critical operational system that Susan knows really well.

Note that these steps should be taken whether IT is the constraint or not. When IT is the constraint, make sure to maintain project-selection discipline, even in the face of major capacity issues, and even if those capacity issues drive down the volume of approved projects in the near term. Stay as true to Effective ROI as you can, being appropriately conservative on CU estimates when IT (or Susan) represents both the scarce project resource and the operational constraint.

A critically important final point on this: If you can find a way to get more projects done without adding resources, you will have a greater ability both to expand capacity at the constraint, and to use that additional capacity to drive up throughput. The next chapter focuses on how to do exactly that.



This book is for senior executives who are on a mission to jack up the performance of their information-technology (IT) project portfolios, and who are no longer satisfied that established "best practices" are sufficient to achieve their organizations' business objectives. The authors cut through the confusion and zealotry of leading improvement approaches, and distill them down to a practical set of specific techniques you can apply for maximum benefit to your IT project portfolio.

The CIO's Guide to Breakthrough Project Portfolio Management