People and Process Management Antipatterns

Antipatterns in lecture

- Related with people:
  - Micro-Management
  - The Brawl
- Related with processes:
  - The Domino Effect
  - Myopic Delivery

Antipattern

A mechanism to describe a commonly occurring solution to a software development need that generates significantly negative consequences. It examines the causes, symptoms, and consequences of implementing the bad solution and offers a refactored solution that provides a successful method to support the required software development need. It is a standard format that represents recurring software development problems and their refactored solutions. Both the scale of the problem/refactored solution pair and the applicable viewpoints are important factors that aid in identifying the specific scope of the Antipattern.

Anttipattern Scale

On the next figure is illustrated Antipattern scale what is originally software design-level model (SDLM):

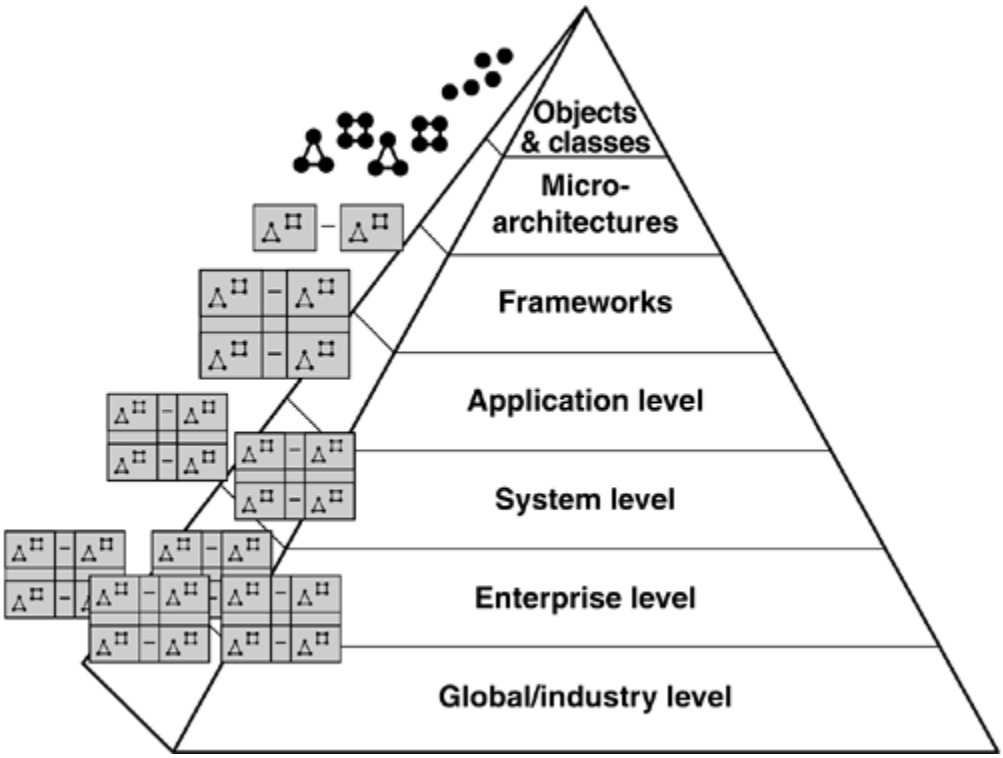

Figure 1. Antipattern Scale

Antipattern Viewpoints

Viewpoints are positions from which a situation can be seen or considered. They affect how a problem is perceived and subsequently dealt with

- Manager viewpoint - people who actually manage teams, projects, and programs; with a specific responsibility for planning and scheduling across the software development lifecycle
- Architect viewpoint - focuses on identifying the technology, specifying its configuration in the form of the architecture, and ensuring its correct implementation
- Developer viewpoint - focused on the implementation of the software development process. Any role in mainstream development implementing the software development tasks

Micro-Management

Also Known As: “Unbalanced People, Technology, and Process, Attrition ‘R’ Us”

Most Applicable Scale: Application

Refactored Solution Name: Macro-Management

Root Causes: Pride and Ignorance

Unbalanced Forces: Management of IT Resources

Anecdotal Evidence

"Management don't have any respect for us"

"Management don't know what we're doing; they live in their own little world"

"If we don't tell the developers what to do each day how can we have any control?"

"The developers should do what they're told and if they don't like it, then they should leave!"

General Form

Project managers' don´t always know how to manage people; understand the technology being implemented, and know how to apply pragmatic processes. They are driven to make decisions in spite of this and any advice to the contrary. They often over manage a particular aspect of people, technology, and process. Either, because they are weak in that particular area, and believe that micro focus will mitigate risks, or because it is the one skill area that they have. The following failure categories identify the common project management weaknesses and the nature of poor practices that lead manager to believe that only apparent solution is micro-management.

People Management Failure

Lack of management focus on people and their issues can cause major disruption to SD. For example, a newly promoted line manager will usually have excellent people skills but no technical skills and often weak process skills, while a recently promoted technical guru or heroic programmer will have excellent technical skills but usually few people skills and marginal developmentwide process skills. Following problems characterize a lack of people management skills and are among most critical

- Unformed development teams – they will not fuction well. Their productivity will be low, and there will always be dissension within the team until it has formed, stormed, normed and performed. A PM is responsible for ensuring that all teams go through this process as early as possible to prevent people problems from getting in the way of developing software

- Lack of clearly defined management roles – leads to uncertainty and indecision among senior staff and team leaders. Managers on a project should understand their roles, their relationships to other management roles, and the decision-making process. It is normally expected that the pm will form the management team and lead the form, storm, norm, and perform process

- Poor motivation and unrealistic expectations – poor motivation causes a lack of productivity. People need a reason to do things. If a developer does not want to perform a software development task there is a tremendous risk of low quality, late delivery, and sometimes non-delivery. Lack of team spirit exhibited by poorly formed

teams also reduces motivation. Unreal expectations can cause frustration and subsequently poor motivation. It is the responsibility of the pm to establish realistic project development vision and set the expectations of all the development staff. Too often this does not happen, assumptions are made, and individual expect-s are unrealistically set

- Uncontrolled Corncobs – cause recurring disruptions and delays to any software development. They are basically disruptive characters who continually vie to have things done their way and ply politics to get their way. PM must manage them strongly or remove them from the development.

- Encouragement of heroes – every development team needs a hero or two, but to rely on them to continually pull a development through its encouraging disaster and the loss of the heroes. Heroes should lead the process by example and act as role models for the others developers.

- Constant delivery pressure – often in the form of unnecessary fire drills causes extreme frustration to developers who need a controlled environment in which to focus on their software development tasks. P managers control the level of urgency imposed on developers and need to understand that constantly high level of urgency benefits no one

- Adding staff to speed up delivery – typical inexperienced approach when managing a software development project. Unfortunately “time*people = work done” is not true. At a certain point adding people to an activity will cause degradation in delivery performance. A p manager needs to know how to balance staffing levels across the project development phases and activities

- Lack of technical respect for developers – many development staff members have successfully delivered a number of systems and their opinions should not only be respected but sought after. Feedback must come from all levels of a software development, not just from management. If a pm does not respect the developers, it is likely that the development team leaders also will be infected by this aberrant thinking

Technology Management Failure

When pm does not understand the technology, he cannot usefully question or validate any information from technical staff and must rely on someone else-s judgment, which introduces risks in technical decision making. Following technical management problems typically cause technical failures of software deliveries.

- Choosing unfamiliar technology – will render a development schedule meaningless because there is no way to accurately account for the learning curve and to predict

the difficulties of successfully implementing a novel technology. A pm must mitigate risk involved in technology adoption and ensure that a project can successfully implement an unfamiliar technology

- Relying on unstable COTS (Commercial of the Shelf) – will cause technical failure in any software development project. Pm must mitigate all technical risks as they are identified. Pm must also be able to decide when a technical approach is unacceptable, rather than relentlessly and blindly pursuing it

- Lack of tools – to properly support development has a critical effect on productivity and quality, particularly in areas such as integrated development environments and testing. Pm must balance budget spending on engineering tools against their effectiveness in helping developers deliver quality code efficiently

- Over engineering – often a symptom of developers adding "cool" technology features that were not required but easy to do with technology. This is a dangerous exercise because it may lead to unanticipated development delays, prompting the pm to micromanage. Usually the mm is applied across the board, not just the single dev team responsible for the delay

Process Management Failure

- Lack of process focus will cause ad hoc practices within a software development project, reducing quality, adding delays, and raising the risk of delivering the expected requirements late. This causes confusion among developers about what is expected in terms of development artifacts and for managers about how to validate the dev artifacts. If a pm cannot state the criteria for the completion of requirements, then the requirements and the nature of the software will endlessly change.
- Lack of planning cycle – many project managers do not see the need to produce a plan, incrementally capture work done against it, and refine the tasking detail and estimates over time. The lack of such a planning cycle reduces the amount of accurate schedule info available to a pm and increases the risks associated with a poor management decisions. This often results in fire drills to meet deliveries on dates that are no longer valid
- Inappropriate lifecycle – many project managers do not think in terms of life cycles and their appropriateness to the nature of development unless there is a guiding corporate standard. For example, using a pure waterfall life cycle for a prototype development will overburden the developers with unnecessary process and documentation. A pm may think then that the developers are not doing their jobs properly and begin to micromanagement
- Optimistic estimating – it is very common to underestimate tasks from a management perspective because it is unusual for a pm to be able to identify every detailed step in developing software and produce an accurate high-level estimate

that has been validated by technical staff. Even when team leaders are producing accurate estimates for their deliveries, a project manager must be able to produce a project plan that removes integration, quality assurance, and documentation bottlenecks, which would cause knock-on delays. These factors can lead to optimistic estimates

- Compressing schedules - because of overrunning the allocated time for earlier development phases, and subsequently compressing the time available to later phases so as not to overrun the delivery deadline, is common among inexperienced pm-s. Of course, since the earlier phases overran, the later phases probably need more time, not less. When the delays become significant, mm is an automatic reaction to get the development deliveries back on track, ignoring the real cause of the problem
- Phase paralysis – occurs when development gets stuck in a phase and cannot complete it. Common phases for this are analysis, design, and coding, and usually incur a creep of scope as subsequent symptom. There are many causes, such as continually changing requirements or architecture. Again mm is an apparent solution for a pm to bring closure to an activity, but it is often perceived with distrust by the developers
- Lack of quality assurance – means that there will be failure of one or more deliverables at some point in the development. This often becomes visible at the release end of the development cycle, although the lack of quality is usually introduced during design or early coding. Once delays occur because of upstream quality errors, the pm again feels that the only solution is micromanagement
- Lack of scope control – scope creep can happen at product requirements, product design, and product coding time. A weak software configuration management process will not be able to monitor scope and check traceability throughout the development artifacts. Each creep will cause a proportional delay. Several delays then trigger mm as an attempt to avoid further delays
- Lack of regular and incremental deliveries – when a development is allowed to deliver in black-box mode, that is, its state of development is not visible, then the risk of technical failure and associated delivery delays increases because the only time that the software can be evaluated is at the end of the process. This software is continually under construction and cannot be tested until very close to delivery, with no contingency built into the development approach. When delivery delays occur because of the lack of regular incremental deliveries, the usual pm response is micromanagement

Symptoms and Consequences

The primary symptom of the mm ap is the effect caused by the lack of pm focus and balanced control of people, technology, and process, which usually yields a lack of

quality and schedule delays. The subsequent is mm as an attempt to erctify the problem(s) exhibited. Original problems usually are increased by a factor, and the nwe problems have a knock-on effect. The final consequence is staff attrition with the side effect of project failure. Shown in the next table:

Table 1. Symptoms and Consequences

Primary Symptom	Primary Consequence	Secondary Symptom	Final Consequence
People Management Failure	• Development of the wrong product • Cost overruns		
Technology Management Failure	• Technical failure • Cost overruns	Micro-management	• Staff attrition • Premature termination of project
Process Management Failure	• Development of the wrong product • Cost overruns		

Typical Causes

Project manager who sees micro management (m-m) as a solution for rectifying people, technology, and process problems. The range of m-m can vary - a fire drill is an occasional form of m-m, while managing daily developer tasks is an extreme form. A project manager often micromanages because of lack of project management skill and experience. In an attempt to make up of lack of skill, the answer for the project manager is to manage very tightly, controlling details that are usually delegated to team leaders and senior developers. The m-m usually has a limited focus, on people, technology, or process. This causes a lack of focus in at least one of the other areas. This then increases m-m for those areas not under control, even though the type of control is incorrect. The greater and longer such micro control is exerted, the larger negative impact on the software development. Known exceptions: when the development team is new and has not formed, stormed, normed, and performed, and to ensure that the technology learning curve is short – making sure that zero time is wasted on dead-end approaches and that appropriate technical support is provided

Refactored Solution

People management success: ultimate software development success depends on everyone working as a team with shared goals. Catalyze formation of management and development teams, and involve everyone in planning

Technology management success: prove the viability of the technology – perform runaheads, and adopt developer-friendly tools.

Process management success: pragmatic project management and controlling the development. Establish project plan and team plan templates for each type of development (prototyping, new technology development, stable/legacy product maintenance)

The Brawl

Also Known As: “Anti-Patton”

Most Applicable Scale: Enterprise

Refactored Solution Name: Leadership 101: Intro to Leadership Concepts

Root Causes: Ignorance

Unbalanced Forces: Management of IT Resources

Anecdotal Evidence

“He’s a great manager/engineer/programmer; he’ll make a great project manager”

“This guy has got degrees from MIT and Harvard, he’ll make a great project manager”

“We don’t believe you have been taking enough risk with your project. We think you should change the architecture to accommodate the latest thinking in technology. This shouldn’t have any impact on the rest of the project”

General Form

There are at least two distinct aspects to being a project: leadership and management. A poor manager or leader typically results in project failure – however, leadership, more than management, can save a project. Often the battle cry of the engineer or the programmer is: “we need a leader not a manager” or “there is no leadership” – when this is the case, development teams muddle about, stagnate, and fail to attain their goals. This festering often results in brawl where each faction of the development team attempts to position themselves into a leadership role. This antipattern explores the result of little, poor, or no leadership by the project manager.

Project manager

The project manager doesn´t understand the difference between management and leadership. He or she focuses on the management of the project and fails to lead. The project manager focuses on the following management processes: planning and budgeting; organizing and problem solving, and process management and process improvement. The project manager doesn´t realize or understands the role of a leader

and these leadership processes: establishing direction; aligning people; motivating and inspiring

Mismatched project leader

Here are two variations: project manager is promoted into the position based on technical or academic capabilities, without training or experience, or a good technologist (engineer or programmer) is promoted into a position or project management and leadership - fails in both the management and the leadership department; ends up meddling in development – often referred to by developers and programmers as micromanagement, or even in the instance where the engineer or programmer is well organized and capable of accomplishing the management skills, he or she fails to be a good leader because

The leadership processes are more instinctive, personality driven, and people oriented. These qualities are more qualitative and less quantitative, which is inconsistent with the traits of most technologists.

Unpowered project manager

Here are also two variations: project manager who is not empowered to be a good leader; He or she cannot effect good leadership skills because the upper-level manager is micromanaging – a good project manager is promoted to the next level of management and is not prepared to handle the responsibilities, or project manager who fails to instill confidence in management and the upper-level managers attempt to involve themselves. This failing typically comes from repeated status briefings when management walks away with an uneasy feeling about the direction of the project. The project manager has failed to inspire his or her own management. The result is the same - the management attempts to run the project; is unable to realize the collateral impact of the decisions made at each of the status meetings when it directs the development team to change directions

Symptoms and Consequences

Lack of leadership results in unsuccessful projects and the symptoms of failure are not altogether different

The project manager finds it difficult to adapt to whatever change is required and fails to realign personnel to accommodate the necessary changes; often this creates a hardship on the staff and morale declines. The development team loses direction. He/she works the solution out on paper but fails to address the development team so that they can comprehend the changes that are required and why. Morale continues to decline and now failures become increasingly more common; the schedule is frequently

impacted at this point, and the project will spiral downward out of control - the death spiral of project is in process

The mismatched project manager as the academic expert. Caution is typically the mode of operation and the project becomes extremely process intensive. He/she will ensure that all documentation is prepared, and that the procedures are followed. They typically do well until some type of challenge is presented to them in which adjustments are difficult. They are unable to adapt, overcome, and succeed just because they haven´t been there before. They are unable to quickly realign personnel assets and motivate and inspire as required. They begin to doubt themselves and quickly look to others for reassurance and direction, including from the project team. It is at this point that the project falls apart and goes through the death spiral.

The mismatched project manager as the technologist. Unless this project manager gets well-experienced, heavy-duty administrative staff, the first symptom will be little or no documentation. Cost overruns and schedule slippage are likely early on as well - because this project manager is spending too much time with the development team trying to figure out a better way. However, even if the technologist project manager is able to evade management, it is short lived when challenge is thrown at the project and the death spiral soon begins. The types of challenges or forces that invoke these problems are typically managerial or administrative - cost, schedule, customer, and market. Rather than technical challenges since the technologist project manager believes any technical problem can be solved given enough time and money

The unpowered project manager. It usually begins with regular project meetings where the management, believing they are more aware and knowledgeable about project effort than the project manager, begins to suggest that specific aspects of the project be evaluated. They may ask the project manager to become more of a risk taker, without fully comprehending the impact. The frequency of review briefings usually increases and, at management direction, they become more detailed. As management continues to issue directives to the project manager and project team, the project manager´s power base is eroded and he or she becomes ineffective. The management directives inflict collateral impact on other parts of the project without the management being aware - this usually causes a domino effect of failure, degrading the already low team morale, and possibly initiating attrition. The final consequence is usually the ousting of the project manager. The new project manager is doomed for failure as well, unless he or she can demonstrate superior technical, managerial, and leadership skills – even then it is an uphill battle.

Another symptom is that management believes that they are experiencing this problem with many of the projects in their purview. Unbelievably (to management) many of the

projects are suffering the same lack of proper project management and morale is bad within their entire organization. Specific symptoms include high attrition among project managers as well as low morale among all of the staff. Additionally, conflict is typical between the project managers and the management.

Typical Causes

There may be primary causes that exist outside of the organization. These external, primary causes drive management to take risks and place young in experienced personnel into project management positions. A good example of this is a tight labor market, especially when individuals with technical skills come at a premium and must be paid a premium. An organization may feel that spending money on technical skills is necessary, but may skimp on paying for good project managers

The project manager - management´s misinterpretation of an individual´s leadership skills, based on the extrapolation of his or her management skills – this results from management´s lack of understanding of the difference between management and leadership

The mismatched project manager - management believes that credentials justify the promotion or hiring. Organizations fail to provide well-delineated paths for promoting and compensating technical personnel. Poor judgment on the part of management – it is all too common that good engineers and programmers are promoted and become bad project managers

The unempowered project manager first variation - upper managers promoted from project management don´t believe that anyone is as good a project manager as they were. They believe that they are no longer the project manager of one project but for several.

The unempowered project manager second variation - lack of leadership, commitment, expertise, and marketing on the part of the project manager. The project manager often presents problems without solutions and without instilling enough confidence for management to believe that resolution are being pursued with an appropriate risk mitigation strategy. Management is not only unimpressed with the project manager but also with the direction of the project and begins to intervene. This is typically the start of the project death spiral

Refactored Solution

To ensure that a project manager also knows how to lead a project, not just manage it. Understanding leadership is the key to the refactored solution. It is critical that an

understanding of the difference between management and leadership be established as well. Applying leadership concepts – expanded situational leadership model for example by Paul Hersey and Kenneth Blanchard. Applying means by which a leader establishes power. Learning to be a leader – leadership courses; mentoring programs; career path for management – separate management and technical tracks. Given that the project is in the beginning of the project death spiral the project manager should be removed immediately. Confront management about their micro-management techniques. Leave the project

The Domino Effect

Also Known As: “Crap rolls Downhill”, “Chaos Theory”
Most Applicable Scale: Enterprise
Refactored Solution Name: Reverse Domino Tilting or Strategic Domino Removal
Root Causes: Multiple
Unbalanced Forces: Management of IT Resources

Anecdotal Evidence

“Hey Rick, your project is doing great! Listen, we’ve decided to pull Ben from your team for just a couple weeks to help out on Jerry’s project. I know he’s your best Java guy, but it’s only for a little while, and Jerry’s project really needs some help. Besides, your guys are doing really great work, and you’re a bit ahead of schedule, so pulling Ben may even help you out by giving some of your junior developers a chance to show some leadership! So talk to Ben about it okay…he’ll have to delay that vacation he’s planned, too, because they want him to start with Jerry this afternoon”

General Form

Antipattern typically occur in organizations that are performing multiple, somewhat isolated tasks simultaneously. These tasks may be closely related, but for the antipattern to be present there must be separate teams for each task. These teams must be largely independent of each other and usually in competition against each other for key resources. Where these conditions are present, a senior manager who has authority over several projects typically tilts the first domino of this antipattern treating the resources for each separate project collectively and interchangeably. The manager believes that he or she can maximize progress (or minimize problems) by temporarily moving resources from one project that´s going well to a faltering project. The manager mistakenly believes that this will bolster the troubled project and that once it is back on track he or she can move the resource back to its original project. The unfortunate an unintended result of this action is the true domino effect. The individual dominoes that are tilted as a result of this first action fall like the proverbial snowball rolls downhill in a

sickening trend of disasters as each project in the organization is eventually touched. The lesson is that to get staff a team must first fail and the reward for success is to have key resources borrowed from the team

Symptoms and Consequences

1. No single project stays on schedule or goes smoothly
2. A general state of crisis management exists for all projects within a particular IT department
3. Cross-project managers frequently move key contributors from project to project to avert crises
4. Retraction of developers into defensive posture that eliminates collaboration and sharing
5. Frequent personnel crises spark expensive personnel searches
6. High personnel turnover (especially among top developers, engineers and project leaders)
7. Strong pressure on project leaders to get deeply involved in development to cover for lost critical staff members
8. A general defeatist malaise sets in because developers no longer identify with their project and no longer really care about success
9. Increasing “blame-storming” and faultfinding as projects tailspin toward failure

Typical Causes

Rapidly expanding or growing organization that has taken on increased workload beyond its present capability with the idea that it can juggle projects and quickly build up the staff needed to support them all. Organizations simultaneously working on multiple isolated tasks with separate, independent teams for each task. Inability to build up an experienced staff to support increased contracted workload, especially exacerbated by the failure to anticipate the high cost of hiring “ready to run” expertise overnight. Key staff members “matrixed” across multiple projects, spreading them so thin that they are rendered effectively useless and moot. Project in competition for key staff members. Senior manager with authority over several projects who treats individual project staff members collectively and interchangeably. Managers and project leaders (those with significant development expertise) being tapped to support the development effort, leaving critical (but not immediately critical) project management to be caught up on later. Dependence on a single person critical to the success of multiple projects where each project requires a large portion of that individual´s concentration and focus.

Refactored Solution

Prevention strategies

First chip is not tilted yet:

- Strategic domino removal - prevent antipattern by entirely putting off or delaying select at-risk projects to ensure the success of those that are doing well
- Strategic domino blocking - Barring cross-project transfers of personnel
- Domino rearrangement - reducing cross-project dependencies on isolated staff members (double up on certain critical skills). Add new, experienced staff to the successful project before reassigning key members from that project to another troubled project

Recovery Strategies

In a situation where the dominoes are already falling

- Spartan domino blocking - Immediately barring all cross-project transfers of personnel without exception, letting the chips fall as they may
- Immediate domino removal - Immediately stopping work, canceling or delaying at-risk projects, and focusing immediate effort on projects that are going well
- Reverse domino tilting (fire block) - If the immediate at-risk project is critical and therefore cannot be cancelled or delayed – push staff up from lower projects to bolster the projects to be robbed of staff. In this way – project manager "backfills" the necessary staff for each project

Myopic Delivery

Also Known As: "Delivery Zone"
Most Applicable Scale: Application
Refactored Solution Name: Get Out of Town
Root Causes: Pride, Narrow-Mindedness
Unbalanced Forces: Management of Resources

Anecdotal Evidence

"What is the difference between management and a terrorist? You can negotiate with the terrorist!"
"Nine women can't make a baby in one month"
"I don't care about the schedule anymore, just do it"

General Form

After months of skillful project management execution, project manager learns that he´s going to miss the delivery date, no fault of his own. Suddenly his management no longer cares about proper project management techniques. His management wants him to deliver to the original date. Slipped schedules generates project compression, what in turn generates a gap in the space time continuum allowing the incomprehensible to accomplished in the minds of management. It causes all logical, pragmatic, rational, reasonable thinking to fly out the window. Management will do anything to meet the delivery date. Everyone will suffer. Nothing is more important than the delivery date

Symptoms

- Management control of the project or micro-management
- Demands by management that additional risk be assumed without compensation
- Management with selective hearing and memory about:
    - Changes they imposed upon the project
    - Identified risks arising from those changes
    - Agreements regarding schedule, cost, and resources
- Incremental crisis project management or little project management and control
- Spending inordinate time on project review meetings and schedules to convince management of project control
- Increased project team frustrations and low morale
- Weak project leadership

Consequences

- Many problems in a compressed period of time
- Low morale and high attrition
- High stress and emotions
- Invalid or unrealistic schedule
- Missed delivery date and project cancellation
- Delivery date is missed and the project is extended with additional funding, then canceled after failing to make delivery the second and third times
- Delivery date is missed and the project is extended with additional funding and then is successful after project leadership is installed
- Delivery is met with the project achieving limited functionality
- Delivery is met with the project achieving limited functionality and poor reliability

- Project is successful in achieving the delivery date, full functionality, and reliability (not likely)

Typical Causes

- Management attempts to control the project, rather than letting the project manager do his or her job
- Delivery becomes the focus of the project, at the expense of all other objectives
- Abandonment of project management principles
- Planning and leadership collapse
- Ignorance of basic project planning and management principles
- Forced management compliance
- Attempting to make management happy without regard for the project

Refactored Solution

The refactored solution must be presented for the various states of the Myopic Delivery Antipattern. For simplicity are states beginning, the middle, and the end considered.

The Beginning

Project manager may consider changing jobs. He/she must consult with project teams and document findings. When meeting management, project manager must be honest, strong, and stand his/her position, but must be absolutely sure that project manager has considered consequences. The project manager must be adamant about not making substantial changes to the project without compensation to the schedule or other resources. Additionally, project manager must get management to become stakeholders in the project. Management is micromanaging. They can´t get enough information. The key is getting them to arrive at the same conclusion project manager has. If project manager can do this often enough, he/she will probably gain their confidence.

The Middle

Leaving project still is not an option. If project manager haven´t implemented the beginning state of the refactored solution, that should be his/her first step. Next, project manager shouldn´t agree to anything that hi/she doesn´t believe is accurate. Don´t acquiesce. If project manager's management seems to have selective hearing and memory, project manager needs to document all concerns, issues, and meeting results and e-mail them to management. If possible, project manager must mail to a larger audience (the next level of management). This will better substantiate project manager's position and also lessens the likelihood of selective hearing and memory of project manager's management. Project managers need to keep management informed and up to date. Too much information is not an option. On the other hand, project manager must attempt to keep the project team isolated from the wrath of management. Project

manager must keep project team morale high and focus on the success of the team. Most important, project manager must stick to fundamental project management and system engineering principles. Finally, project manager must try to keep perspective. There is more to the project than meeting the schedule. This can be hard to do when management has become myopic. Helping management remember that there is more to the project is important too. Often they lose sight of what is really important, which is making the customer happy.

The End

In most cases that is exactly what it is – the end. The end of projects, jobs, and careers. If the Myopic Delivery antipattern has progressed this far, project manager must bail out as quickly as possible. There is one other option. Project manager must determine what he/she can safely and reliably deliver on the promised date, and deliver it. Something is better than nothing. Delivering something will likely save the project and probably more important project managers job. Whatever project manager does, he/she can´t blame his/her management. By now it is too late and he/she will suffer all the consequences.

Used Literature

- William J. Brown, Hays W. McCormick III, Scott W. Thomas: Antipatterns in Project Management, 2000
- Software Design-Level Model, http://users.atw.hu/softarchcamp/ch02lev1sec9.html

Additional reading/watching:

- http://www.youtube.com/watch?v=CZB7XkxR_Fo
- http://en.wikipedia.org/wiki/Anti-pattern
- http://c2.com/cgi-bin/wiki?AntiPatternsCatalog