##### How to Run a Small Software Development Organization in Internet Time and Not Go Crazy

## Table of Contents

[Table of Contents 2](#_Toc151203352)

[Document History: 4](#_Toc151203353)

[Overview 5](#_Toc151203354)

[Intended Audience 5](#_Toc151203355)

[Description of the problem 5](#_Toc151203356)

[The Solution 5](#_Toc151203357)

[Goals 6](#_Toc151203358)

[Success Criteria 6](#_Toc151203359)

[Glossary 6](#_Toc151203360)

[Discovery: Identify what we're building 8](#_Toc151203361)

[Goals: 8](#_Toc151203362)

[A word on the problem 8](#_Toc151203363)

[Execution 8](#_Toc151203364)

[Discovery 8](#_Toc151203365)

[Refinement 8](#_Toc151203366)

[Articulation, Validation and Prioritization 8](#_Toc151203367)

[Description 8](#_Toc151203368)

[Estimation 8](#_Toc151203369)

[Authorization 8](#_Toc151203370)

[Definition: Translate what we're building into a definition 8](#_Toc151203371)

[Goals 8](#_Toc151203372)

[Execution 8](#_Toc151203373)

[Architecture 8](#_Toc151203374)

[Estimation 8](#_Toc151203375)

[Schedule 8](#_Toc151203376)

[Treatment of "side-projects" 8](#_Toc151203377)

[Risks and schedule impact 8](#_Toc151203378)

[How the plan is kept 8](#_Toc151203379)

[Deliver against the description 8](#_Toc151203380)

[Goals 8](#_Toc151203381)

[Execution 8](#_Toc151203382)

[Detailed design 8](#_Toc151203383)

[Iterative development 8](#_Toc151203384)

[Schedule tracking 8](#_Toc151203385)

[QA 8](#_Toc151203386)

[End game 8](#_Toc151203387)

[Document control 8](#_Toc151203388)

[General Process Issues 8](#_Toc151203389)

[Decision process for research-oriented efforts 8](#_Toc151203390)

[Decisions and Meeting Documentation 8](#_Toc151203391)

[Longer-Term Development Issues 8](#_Toc151203392)

[Change Requests 8](#_Toc151203393)

[Appendix A: Mind Set (or, “The Five Habits of Highly Effective Company X Marketing Professionals”) 8](#_Toc151203394)

[Appendix B: Feature Idea Form 8](#_Toc151203395)

[Title: 8](#_Toc151203396)

[Your name: 8](#_Toc151203397)

[Date: 8](#_Toc151203398)

[A description of the "Business Problem" 8](#_Toc151203399)

[A description of the necessary characteristics of a solution. 8](#_Toc151203400)

[A description of a possible solution or solutions. 8](#_Toc151203401)

[A necessary time-frame. 8](#_Toc151203402)

[Appendix C: Needs/Solution Specification Package 8](#_Toc151203403)

[What exactly is the pain? 8](#_Toc151203404)

[What technologies/solutions currently exist to ease the pain? 8](#_Toc151203405)

[What is the exact required solution? 8](#_Toc151203406)

[What would be the likely bottom-line impact if we delivered the solution? 8](#_Toc151203407)

[Does this line up with our mission? With our initiatives? 8](#_Toc151203408)

[What is its priority? 8](#_Toc151203409)

[Is this a big deal, development-wise, or a small deal? 8](#_Toc151203410)

[Appendix D: Development Authorization Form 8](#_Toc151203411)

[Appendix E: Emergency Management Change Order 8](#_Toc151203412)

[Title 8](#_Toc151203413)

[Signoffs 8](#_Toc151203414)

[Crucial Driving Business Need: 8](#_Toc151203415)

[Time Driver (when must this feature be delivered): 8](#_Toc151203416)

[Downside (what do we lose by doing this): 8](#_Toc151203417)

[Requirements: 8](#_Toc151203418)

[Specifications: 8](#_Toc151203419)

[Risk Analysis: 8](#_Toc151203420)

[Estimate: 8](#_Toc151203421)

# Document History:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 1 | 5/26/06 | Steve |  |
|  |  |  |  |

# Overview

## Intended Audience

Every employee of Company X.

## Description of the problem

We make the following observations about Company X:

There is no clear, articulated roadmap for a) synthesizing a coherent vision b) turning that vision into a product definition c) crystallizing the definition into an implementation and d) delivering the solution.

There is no clear understanding of responsibilities and therefore the schedule slips on the fuzzy front-end. The vision is fuzzy, and solutions are fuzzy vs. this fuzzy vision. It is unclear how a vision takes form. Who decides the requirements and how these relate to the vision? Who decides the specifications? Who signs off? Who is critical path when? Who makes the cost/benefit calls, both in terms of real and opportunity costs? How do we drive progress on research-oriented efforts required to support our long-term vision? How are releases timed and managed along the way towards full releases? How are resources leveled against development efforts? And how are finer-grained questions answered during development of the product?

Furthermore, there is a legitimate tension between the needs of Marketing to react to market demands and the needs of the Development to have stable requirements against which to deliver products. Without the former, a company can deliver a very high quality product which may, after 6 months, be woefully off-target. Without the latter, *nothing ever gets delivered*.

## The Solution

The solution is to view product development as a pipelined process and to fully specify the process. In this way, requirements for one version are being refined while the previous version is being developed. Release cycles are short (we aim for two-month cycles) allowing for stable requirements during the development phase, but allowing for relatively quick reaction times on features: if a feature is needed, it is placed into the pipeline queue for release during the next development phase.

###### Release 1

###### Release 2

Definition

Discovery

Discovery

Delivery

Definition

Delivery

This document outlines the process, called the Product Pipeline, including responsibilities, timelines, documents and deliverables.

## Goals

The overriding goal of this document is to articulate the means by which Company X will efficiently create wildly successful products.

For each member of Company X, from President to Janitor, the goal is for everyone to understand at any moment in time the following

1) What is the driving vision of the current efforts?

2) Where we are in the process?

3) Who is critical path now?

4) What's happening next, and especially what is my next responsibility is?

5) What do I have authority to decide and what do I not?

6) What is my role?

7) How I can get my great idea into the pipeline for consideration?

8) How are feature additions considered and prioritized--that is, how will ideas be judged and compared vs. other great ideas?

9) How, if I do my job well, is my contribution likely to result in the success of the project, the team, and the company?

10) How do I decide which of my current tasks have highest priority?

For the company, goals include the following:

1. Maximize efficiency within the organization (reduction of meetings, eliminate wasted time on the front-end)
2. Get the finished product defined and delivered effectively.
3. Streamline and clarify the decision-making process
4. Allow the company to operate in “Internet Time”

## Success Criteria

Short term, this document will be a success if progress occurs even in the absence of direct and continual managerial attention.

Long term, this document will be considered a success if the following conditions are substantially true during the Leavenworth and EastSound releases:

1) There is no wasted time on the fuzzy front end.

2) These releases are released on time.

3) The releases are considered by management to have been successfully released against a coherent vision.

## Glossary

**PM** Product Manager. Until a Product Manager is hired, the responsibilities of the PM will fall to a designated individual except where noted.

**DD** Director of Development

**NSSP** Need/Solution Specification Package. A document describing everything about a set of functionality, from the business need, through existing solutions, the proposed solution, business requirements, specification and estimate.

**Architecture** Description of the major subsystems within a software package and how they relate, including data flow diagrams, coarse API descriptions, error handling mechanisms.

**Detailed Design** Description of the software down to the subroutine/class, including parameters, descriptions, API's, return values.

**Specification** How the software works from a UI perspective. This includes all possible permutations of stat within a UI group, as well as limits on input and behavior.

**Risk Officer**: A person designated to worry. This can be any member of the team, but their function in this capacity is to identify risks to the schedule, estimate their potential impact, assign a likelihood to this, gather and manage the plan for mitigating or eliminating the risk, and to maintain all this in the Risk Management Spreadsheet.

**CRC** Change Review Committee. A designated group of representatives from various parts within the organization whose job it is to categorize, and approve or deny change requests. Typically, this includes the DD, a developer and the PM. The CRC controls Bug categorization (which get fixed and which don’t, who fixes it) and reviews Feature Idea forms if they are submitted as urgent (that is, if they are to be considered for the current release).

**Development Authorization Document** A document serving as the main authorization for a development effort. It contains general information about the project, hyperlinks to specific documents that make up the project (project schedule, risk management spreadsheet, resource needs, NSSP's) plus signature lines for management, QA, marketing and sales.

# Discovery: Identify what we're building

## Goals:

Efficiently gather great ideas

Distill them to the winning few

Check against initiatives or change the initiatives

Maximize cost/benefit

Articulate the characteristics of the solutions as specifically and clearly as possible

## A word on the problem

We are a small company, with very limited resources, living in Internet time. Our window of opportunity is exceedingly short, and demands that we maintain laser-precision focus on delivering exactly what the market needs in the shortest possible time. Understanding our market and articulating its needs is a prerequisite to success in any industry, but for us, the prerequisite is immediate and urgent.

The problem of figuring out where we should spend our precious resources and time is, however, difficult. It is a mixture of efficient process, of state-of-mind, of prescience and openness and circumspection; it is part art and part science and no small amount of luck. Likewise, delivery against this definition is also part art, and part science (although not as much luck). However, by far the larger determining factor is creating the definition. (See Appendix A, The Five Habits of Highly Effective Company X Marketing Professionals.)

## Execution

The PM[[1]](#footnote-1) will enact the following process:

Discovery

Refinement

Validation, Articulation, and Prioritization

Description

Estimation

Authorization

### Discovery

Discovery is the process of researching the customer needs an capturing them for use in the process of defining a solution.

Process-wise, ideas need to be captured and tracked. This also is the job of the PM[[2]](#footnote-2) and will be recorded on the Feature Spreadsheet. A process for collecting these will be initiated as follows:

Any idea coming from anywhere will be sent via email to the PM, using a form to be defined, which contains the following information:

* A short title
* Name of the submitter
* A description of the "Business Problem": that is, what is the pain that the feature in question will solve? Also, whose pain it is (ours, our customer's, our content providers', the end user's)
* A description of the necessary characteristics of a solution. What must the feature do in order to solve the problem? In fact, if there is no solution, this may be the extent of the form.
* A description of a possible solution or solutions. This includes the technologies (if known),the specific characteristics of the proposed solution(s), and how the solution(s) solve the problem.
* A necessary time-frame (that is, the window in which this must be accomplished in order to be worthwhile)

This document will be entered into the Feature Spreadsheet and reviewed at least during the definition phase of the development cycle, and more quickly if there is an especially urgent need. The PM will work with the submitter to gather as much information as needed to get to the next step.

### Refinement

The PM, with feedback from the submitter, will answer and record the following information:

* What exactly is the pain?
* What technologies/solutions currently exist to ease the pain?
* What is the exact required solution? What are crucial elements, nice-to-have elements, fluff?
* What would be the likely bottom-line impact if we delivered the solution? (More sales, market positioning, resource requirements reduced on our side, etc.)
* Does this line up with our mission? With our initiatives?
* What is its priority?
* Is this a big deal, development wise or a small deal? This requires quick feedback from Development

### Articulation, Validation and Prioritization

Once the idea is refined, it needs to be validated. This would be done by the management group, including representatives from Marketing Management, Admin, Development, and Sales. The PM would present the arguments and documentation described above and the goal of these meetings will be to arrange a prioritization across all solutions on the list. This will be accomplished in the manner of version 1.0. More specifically, the PM will present the whole list and do the following:

1) Mark items they deem of very low priority. In the interest of efficiency, these will be only lightly touched on.

2) Describe all others.

3) Send the parties off to rank order the remaining items on their own.

4) Call another meeting go over everyone's top 3 and debate

5) Score each item, creating a ranked list.

6) Publish the results.

Every effort should be made to come to consensus on this list. If this proves impossible, the PM will make the final call. The resulting list is the definition of the next release/releases of the product.

### Description

At this point, the PM creates a specification of the features/additions described. This portion of the document should completely specify the UI, and should go no further than the UI. This is the responsibility of the PM, but will involve feedback from Development under the direction of the DD in terms of completeness and in terms of efficiency. The more development can understand the specific need, as well as the proposed solution, the more development can aid the process by suggesting efficient ways to accomplish the goal. This and all preceding steps are articulated in a document called a Need/Solution Specification Package (NSSP).

### Estimation

Development will create a time estimate based on the specification. The DD will assign leads for each NSSP, whose responsibility will be to estimate to +/-50% the amount of work required to create the solution in question. This becomes part of the NSSP.

As part of the estimation process, development will begin architectural design when requirements are approximately 80% stable, as defined by the PM and the DD. This will become part of the NSSP.

### Authorization

The PM will host a final authorization meeting in which all the stake-holders (management, PM) are presented with the Development Authorization document. This document will contain the following information (note that responsible parties are shown at the end in brackets):

Release details (internal name, proposed version, release theme) [PM]

Date or Feature Driven [PM, with feedback from MGT]

Context within a multi-release plan (what's been done, what's coming). This section will contain any forward-looking statements about where feature-sets might go in order for the developers to plan ahead in their designs [PM]

Overview of feature additions, with rank ordering (essentially, a copy of the whole Feature Spreadsheet, listing both those features being added as well as those that aren't) [PM]

NSSPs [PM, Development Leads]

+/- 50% schedule [Director of Development]

Resource estimates [DL, DD]

Risk items and Risk Management Spreadsheet [Risk Officer]

Signoffs

# Definition: Translate what we're building into a definition

## Goals

Translate the specifications into implementation plans (architecture, detailed design).

Articulate the details of the solution within the product so that development can accurately refine the estimate

Generate a +/- 10% estimate of time to development

## Execution

### Architecture

The Architect, as designated by the DD, will design and articulate the architecture for the release in accordance with the Architect’s overall vision for the product. The architecture should attempt to anticipate additional changes and additions according to both the feature list as it exists at the time of architecture design, and according to the context description presented in the development Authorization document.

The Architect will hold a review meeting when the architecture is ready for review by development.

### Estimation

#### A word on background and theory:

Software development, and therefore estimation thereof, is inherently uncertain. It is a process of gradual refinement of definition and clarity. Decisions are made all along the way, all of which affect the amount of time it takes to develop the functionality. Events occur that cannot be predicted a priori. Therefore, an estimate presented as a certain date is a lie: statistically, and assuming perfect efficiency, the probability that a software project will be delivered on the exact hour specified is zero. The only meaningful estimate is a range, the range being described as a confidence interval, or a probability that the actual delivery time will occur within the range specified. A 95% confidence interval is therefore wider than a 10% confidence interval (the former means that we are 95% certain the actual date will occur within the range of dates, the latter that we are only 10% certain).

#### Execution

The trick then is to a) make the range as accurate as possible, b) anticipate and quantify uncertainty and unplanned events and c) revise the estimate as the range narrows and d) not substitute wishful thinking for objective and rational estimation techniques.

To accomplish a) and d), we will utilize three different techniques to estimate the process. One of these will be based on historical data. The DD, at various stages in the project, will track and record estimates vs. actuals per developer, and statistics will be kept which will be applied at the next estimation stages. One other technique will be a software estimation tool (TBD). The other will vary, at the discretion of the DD.

To accomplish b), historical record will be kept of side-projects, and applied to available time. For this first go-around, this will be estimated post-facto. This will be published and displayed on the development area.

c) will be accomplished at various planned checkpoints in the process. Uncertainty inherent in fuzzy deliverables will be reduced by the use of clear and granular milestones.

Finally, the stochastic nature of the estimate will be communicated to management during status reviews and at the beginning (authorization) of the project.

### Schedule

The DD will own the schedule, and will publish updates weekly. The schedule will be shown as a Pert chart in the Development Area, as well as a Gantt Chart.

The schedule will utilize milestones at the end of each NSSP, in which QA signs off on the functionality, meaning it is bug-free as defined by the CRC, up to the limitation of system-testing. The goal is for QA to find bugs as soon as they are written (this is discussed in more detail below, in the section on Delivering Against the Definition)

### Treatment of "side-projects"

As mentioned above, side-projects (projects which are not part of a defined development effort, but which may nonetheless be necessary, such as: IT, TS, sales demo's, administrative overhead, etc) will be tracked by adding tasks in the schedule as a number of hours per week and publicized. Side-projects will be tracked during the schedule, and the schedule will be modified as side projects materialize or don't.

### Risks and schedule impact

Part of the weekly effort involved in running the project, and in defining the schedule, will be the risk analysis and management plan. The DD will assign a Risk Officer who will articulate and estimate the impact of schedule risks. These will be multiplied by the likelihood of the risks and the aggregate of this time will be added to the schedule.

### How the plan is kept

All elements of the plan, including the schedule, the Development Authorization document, the Risk Management plan, the Who's In The Hot Seat and Bug Glide Paths (see below) will be kept in SouceSafe under version control. Source control will be initiated at Authorization or earlier, for appropriate documents.

# Deliver against the description

## Goals

Deliver what was specified in the above step according to the estimate

Enforce stability

Discover any bug within 3 days of it being written

Accurate progress tracking and management

Shortest possible delivery date

High-quality output

## Execution

### Detailed design

Detailed design will be done by the Lead for each NSSP and will be reviewed at completion. Any changes to the estimate and the schedule will be updated in the NSSP and communicated via email to the DD. Any changes to the UI as a result of Detailed Design (in order to reduce schedule time) will be approved by the CRC and the PM.

### Iterative development

Iterative development refers to a philosophy guiding software development which emphasizes stability during the process. At Company X, we will implement iterative development by developing in the following ways:

#### Feature-by-feature

Developers will have ownership of specific features as defined in the NSSPs. Schedules will be broken out in terms of these feature sets.

#### High priority items first

As much as possible, the highest priority items will be accomplished first. Thus, lower priority items can go into the next release if the project needs to be delivered early because of crucial business needs. This also means that higher priority items receive the most exposure.

#### QA as it's being done

QA will be performed as the features are implemented. This is to support the goal of catching a bug within three days of it being written. To accomplish this, QA will begin the process of automating the builds, then will automate the installation and finally the "smoke test".

Developers will alert the QA lead or QA member assigned to their area when code is checked in and ready for test. (This, of course assumes unit testing is complete). Any test code to be included in the automation suite will also be flagged to the QA lead.

QA will have the final say (subject to the review of the CRC) of when a feature is complete: it will be flagged as done when no High or Critical bugs exist for the feature.

#### Milestone per feature

Progress on the project will be tracked on the basis of feature completeness as defined above: that is, the milestone for each task will be QA's signoff that the feature is bug-free. Until it is bug-free, it will not be considered done, and the milestone, as well as the project, if it's critical path, will slip.

### Schedule tracking

#### Milestone driven

Milestones will be defined as part of the project plan. These will be, besides QA signoff on features as described above, at other crucial points. Milestone granularity will be about 1 per week per team-member, and will be based on quantifiable attributes. The critical path milestones will drive the project, so that if a milestone date is approaching, the team will do whatever is necessary to make the milestone date, or the project will be slipped. This is an explicit attempt to avoid the Death March at the end, and to spread the pain of the project across the length of the project (hopefully, less of it).

#### Visibility to management

The DD will report weekly to the President and department heads on the status of the project. The Project Plan will also be visible in the development area, as will the Glide Path and Who's in the Hot Seat Now.

#### When does a ship date slip?

The ship date slips when a critical path milestone date is passed without completion. The DD will issue an email to all department heads alerting the company to the slip, along with an explanation.

#### Defect tracking and triage

The CRC will meet at least every other day and daily if necessary to triage bugs. See the document entitled The Life of a Bug (see Appendix G) for a description of the process. Bug tracking will continue to be done using the web-based tool currently in use.

### QA

#### Test Plan

QA will write a test plan for the various features once the specification portion of the NSSP is complete. The test plan should cover features which can be tested as each major feature is being developed, as well as system-wide tests that should be performed once each feature and all features are complete for the release. Test plans should be referenced in the NSSP.

#### Signoff

“QA Signoff” will be accomplished in the NSSP for each specific feature. Signoff means that QA has performed that tests outlined in the test plan and that all bugs are either closed or of low priority (see the definitions of bug priority in the Appendix).

### End game

#### The Death March and "Raising the bar"

The schedule will include time at the end for system stabilization in which no new features are implemented (or at least checked in). As the code freeze date approaches, the CRC will get more and more selective about the types of bugs relegated to the fix category on the theory that a known bug is better than an unknown bug. Also, this is typically a time of great risk to the schedule since the regression cycle can in essence be of indeterminate time.

#### Who decides when it's good enough?

The CRC in conjunction with QA will determine when a release is done.

#### Bake time

Any testing from this point on will result in release notes as opposed to bug fixes, unless a bug of such egregious effect is found (involving data corruption or security breach) that a restart on the bake time is necessary.

### Document control

The following documents will be controlled by the corresponding people. Note that the documents marked with a \* are kept in SourceSafe. Generally, only the owners of the document will change their document. Note that once entered into SourceSafe, a change to any of these documents must be accompanied by a comment describing the change and the reason for the change.

Schedule\* DD

NSSP\* PM, Lead Developer

Development Authorization\* DD

Risk Management Spreadsheet\* Risk Officer

Who's in the Hot Seat DD

Glide Path\* DD

Feature Spreadsheet PM

# General Process Issues

Here is a miscellany of issues which didn’t cleanly fall into the three parts of the Pipeline.

## Decision process for research-oriented efforts

This is the subject of another document, but needs to be mentioned here. There will be a number of research-oriented efforts under way at any point in time. These, since they are not initially connected with a particular release proceed in parallel, with only the resource dependencies as connections between them. When a research effort comes out of Phase 0 and finds definition in Phase 1, it is then brought into the pipeline as a NSSP.

## Decisions and Meeting Documentation

A word on the problem: There have been a couple instances in which decisions were made, but were not well documented and the decision, or the reasoning behind it, couldn’t be recalled or reconstructed. Since we hardly have time to make the decisions once, we certainly can’t be making them twice. On the other hand, the process can’t be onerous or difficult. Therefore, we’ll implement the following with respect to meetings:

1. One person will be designated as the keeper of the notes for each meeting at the discretion of the meeting host (the meeting host is the one who calls and schedules the meeting). This person will write on standard paper (as opposed to DayTimer or other paper) any notes on reasoning and arguments. The goal here is to be able to reconstruct why a decision was made 1 year hence.
2. These notes will be kept by the owner of the project (typically the PM or perhaps the DD).

The meeting host will send (and keep a copy of) an email to meeting participants after the meeting outlining decisions made. These can be very short, and are meant to leave a paper trail on decisions.

## Longer-Term Development Issues

The Pipeline is organized as around a 2-month cycle. There are times when development efforts may take longer than this 2-month cycle, sometimes much longer. However, it is still the goal to release at approximately this cycle: besides allowing for feature additions, it forces the product to stability at regular intervals.

Therefore, efforts that are larger in scope will be handled by defining milestones for these features which fall into these cycles. Functionality may be implemented but not exposed, or partial functionality exposed, but will be releasable within the 2-month window and at the same time. Simply put, tasks for these items may span multiple releases.

In situations involving major changes to the underlying architecture which therefore cannot be released partially, either a strategy will be defined in which parallel underpinnings are employed or a release will be delayed past the normal 2 months. However, this is to be avoided if possible.

## Change Requests

The Pipeline is engineered to affect a balance between the need to have stable requirements in order to deliver a quality product as rapidly as possible and the need to be able to react quickly to market demands. The 2 month release cycle means that any change can essentially be implemented in about 3 months from requirements to delivery. Also, changes during the development process are inherently much more costly than during the design phase: when the affects on coding, documentation, product stability, testing, personnel assignments and training are taken into account, a change done later than design is typically between 50 and 200 times more costly than if it’s done during design[[3]](#footnote-3).

Clearly, it is in the best interest of the company to control change during the delivery phase of the pipeline and to enforce, as much as possible, the early and complete specification of the product.

However, there may be times when change is necessary.

To allow for and control this, the following process will be observed:

1. A Feature Idea form will be completely filled out by the originator of the request (this could be Marketing or Sales) and submitted via email to the PM. On the FI will be an indication of the time-frame of the need: that is, if the FI needs to be considered for the current release or should be placed in the normal pipeline review process.
2. The PM, upon reviewing the FI, will forward it to the DD if it is an urgent and immediate FI. Once the DD accepts the form as complete, it will be reviewed with the CRC. The CRC will hazard a guess as to the likely schedule impact, including design, coding, testing, documentation and configuration management. Schedule impact will include the current and all future releases.
3. Based on the schedule impact, the FI will be classified as one of three categories: Category 1 is minimal, with schedule impact of less than one day. Category 2 is moderate, with an estimated schedule impact of 1 to 5 days. Category 3 is significant, meaning the schedule impact is more than a week, and is significant enough to warrant the EMCO procedure outlined below.
4. Category 1 FI’s will be implemented and the schedule will shift appropriately. Category 2 FI’s will be studied further by Development to refine the estimate and a decision made based on the refined estimate.

If the need is urgent enough and the FI is Category 3, the following process will be observed:

1. An Emergency Management Change Order will be signed by all senior company executives. This is shown in Appendix E. It essentially orders Development to stop development on the current release, tie up any loose ends, and begin estimation based on the specifications contained in the EMCO. The EMCO must therefore contain all business requirements, business justifications, analysis of alternatives, complete UI specification and risk analysis. Work on the current release will not stop until this document is approved.
2. Depending upon where in the Pipeline we are, the product will be brought to stability, and QA will sign off on it as stable, and work will start on the EMCO. The whole Pipeline will be shifted by the amount required to implement the EMCO and once the EMCO is completed, work will progress as normal (just slipped by the corresponding time plus any other time required to test the new functionality along with the originally slated release functionality).

# Appendix A: Mind Set (or, “The Five Habits of Highly Effective Company X Marketing Professionals”)

The first Habit of Highly Effective Company X Product Management is summed up by Clint Eastwood: "A company's got to know its limitations." Ours are time and resources, as discussed above. Therefore the market need must be understood to the degree that it can a) be articulated and defined so that the need is met and b) pared down to its essence. Every minute spent specifying, coding, debugging, testing and documenting even a marginally important feature is a minute lost to competition, a minute we don't have.

The second habit is openness. Closing off the problem set because we can't currently envision a solution is a cardinal sin and misses the point. This second habit needs to be codified in a process that allows great ideas to take root.

The third habit is circumspection. An effective PM will constantly be gathering input from many sources, including sales, competition, industry news, seminars, history, development, fictional literature, military history, CNN, the arts, the sciences, the guy next door, and the self. He or she will be combining them in new ways, keeping things stirring, prodding Development, Management and Sales to see things in other ways. They will always be questioning.

The fourth is persistence. The PM will always be hammering at the problem, always articulating it, describing it as the festering sore that it is in the market. "Ease their pain" and "*this* is their pain."

The fifth habit is meta-thought: the ability to think about how one is thinking; to understand that a solution is not a solution if it is done poorly; that added features are as much of a risk as they are a benefit; that compressing more stuff into a certain amount of time can sacrifice quality in disastrous ways. They should be saying at crucial points, "Is this worth the risk in time? Is this worth the risk in quality? Is this my pet feature or is this really going to make us successful?"

# Appendix B: Feature Idea Form

### Title:

### Your name:

### Date:

### A description of the "Business Problem"

That is, what is the pain that the feature in question will solve? Also, whose pain it is (ours, our customer's, our content providers', the end user's)

### A description of the necessary characteristics of a solution.

What must the feature do in order to solve the problem? In fact, if there is no solution, this may be the extent of the form.

### A description of a possible solution or solutions.

This includes the technologies (if known),the specific characteristics of the proposed solution(s), and how the solution(s) solve the problem.

### A necessary time-frame.

That is, the window in which this must be accomplished in order to be worthwhile)

# Appendix C: Needs/Solution Specification Package

This document starts at the Business Requirement Initiation Form and expands upon it.

### What exactly is the pain?

### What technologies/solutions currently exist to ease the pain?

### What is the exact required solution?

What are crucial elements, nice-to-have elements, fluff?

### What would be the likely bottom-line impact if we delivered the solution?

More sales, market positioning, resource requirements reduced on our side, etc.

### Does this line up with our mission? With our initiatives?

### What is its priority?

### Is this a big deal, development-wise, or a small deal?

This requires quick feedback from Development

# Appendix D: Development Authorization Form

# Appendix E: Emergency Management Change Order

## Title

## Signoffs

## Crucial Driving Business Need:

## Time Driver (when must this feature be delivered):

## Downside (what do we lose by doing this):

Schedule Slip (current and all future releases):

Cost:

Instability:

Initiatives:

## Requirements:

## Specifications:

## Risk Analysis:

## Estimate:

1. Currently, of course, we have no PM, and have therefore tried to fashion one out of a group of managers. This is the best that can be done under the circumstances, but probably sacrifices the coherence and vision (an therefore message) that a single PM will bring. However, the process is essentially the same. And it still demands that one person take on the responsibility of shepherding this process through. This person would be the "acting PM" until one is hired. [↑](#footnote-ref-1)
2. In the absence of this being one person's job, this becomes all of our job. This will be helped by long-range brain-storming sessions as done recently by MJ and JL, by addressing in the bi-weekly company meetings, and by addressing in select development meetings. [↑](#footnote-ref-2)
3. Boehm, Barry W., and Philip N. Papaccio. 1998. “Understanding and Controlling Software Costs.” IEEE Transactions on Software Engineering, vol. 14, no. 10 (October): 1462-1477 [↑](#footnote-ref-3)