MIT ESD

Massachusetts Institute of Technology
Engineering Systems Division

ESD Working Paper Series

Characteristics and Enablers of Transparency in Product Development Organizations

Ryan M. Shaffer
Research Assistant
System Design and Management
Massachusetts Institute of Technology
Email: rshaffer@mit.edu

Alison Olechowski
Ph.D. Candidate
Mechanical Engineering
Massachusetts Institute of Technology
Email: alisono@mit.edu

Warren P. Seering
Weber-Shaughness Professor of Mechanical Engineering and Engineering Systems
Engineering Systems Division
Massachusetts Institute of Technology
Email: seering@mit.edu

Mohammad Ben-Daya
Professor of Industrial Engineering and Operations Research
Systems Engineering Department
King Fahd University of Petroleum and Minerals
Email: bendaya@kfupm.edu.sa

ESD-WP-2015-06 March 2015 esd.mit.edu/wps

Characteristics and Enablers of Transparency in Product Development Organizations

Ryan M. Shaffer

Alison L. Olechowski

Warren P. Seering

Mohammad Ben-Daya

January 4, 2015

Table of Contents

Characteristics and Enablers of Transparency in Product Development Organizations

Abstract

Risks in product development lead to schedule and cost overruns and poor product quality. While many risk management frameworks have been published and research on specific practices has been conducted, little is understood of key characteristics of successful risk management in product development and how they manifest in real development projects.

This research consists of two phases. The first phase is a survey on 171 best practices in risk management. Analysis of over 200 responses from industry practitioners identified transparency as a key characteristic of successful risk management in product development. Due to the limited exploration of the concept of transparency in the literature, the second phase of this work consisted of a qualitative investigation of transparency through interviews with 15 industry practitioners. Analysis of the interview results suggests a hierarchical structure which decomposes transparency into several characteristics and identifies enablers for each of these characteristics.

We propose that transparency can be a valuable lever for product developers and managers. Future work is needed to validate the generalizability of the observations provided.

1 Introduction

Developing a commercial product is a process wrought with uncertainty in suppliers, requirements, market, information quality, and more. This uncertainty and the negative effect of its associated risk can lead to schedule and cost overruns and poor product quality. Risk management, in a version modified from its original application in the financial markets, has been an increasingly common approach to identify and mitigate product development risks on projects. Alongside industry's growing interest and uptake in risk management, we see a growing collection of risk management guidelines and academic literature, some that are aimed at specific industries, and others that assert universal applicability. The guidelines are typically made up of commonly accepted heuristics for conducting risk management at a large commercial firm; we imagine that these heuristics stem from the thoughtful reflection of experienced practitioners and built-up wisdom. Yet little research has been done to empirically explore and evaluate real risk management in product development organizations, or to understand in detail the mechanisms by which risk management works.

This work seeks to identify and carefully examine a previously unexplored characteristic of successful risk management in the product development context. First, we analyze data from a detailed survey of risk management in product development, which tests the association between successful outcomes and risk management characteristics. The evaluation of the survey results highlights a previously unexplored concept in product development risk management, that of "transparency". We then examine this concept in greater detail through interviews with product development practitioners and propose a hierarchical structure which decomposes transparency into several *characteristics* and identifies *enablers* for each of these characteristics.

2 Literature

2.1 Empirical studies of risk management

With the growing number of risk management guidelines available to practitioners, more work is needed to empirically study the use and effect of risk management practices in product development. Those studies that do exist tend to focus on describing risk management use; those that draw conclusions between risk management practices and project outcomes are highlighted below.

This study aims to enhance this literature by providing an in-depth empirical description of a risk management practice associated with positive product development outcomes.

A study of 700 project managers explores the effectiveness of risk management practices that reduce risks in project management and lead to project success (Zwikael & Ahn 2011). The analysis found that risk was negatively correlated with project success, but that effective risk management planning could moderate the effect of those risks. Actionable findings towards effective risk management included the suggestion that risks should be discussed openly with relevant stakeholders. This final point pertains to the idea of transparency, which is further discussed in the next section. A survey of 175 risk management professionals, focusing on project management risk, found that project success occurred more frequently with greater senior management support of risk management, actual practice of risk management practices, and regular risk monitoring (Voetsch 2004). In another study, the authors show that risk management strategies aimed at technological, organizational, and marketing risk factors contribute both individually and interactively to the performance of new product development (Mu et al. 2009).

2.2 Transparency in risk management

Transparency is a popular topic in the academic worlds of monetary policy, international business and corporate governance, but has been explored less in the product development community.

The findings of Chapman and Ward (2004) are particularly relevant to this discussion of transparency in product development risk management. Best practice in project risk management is said to require the elimination of "dysfunctional 'corporate culture conditions' like 'a blame culture' which fosters inappropriate blame." The authors argue that best practice cannot be

achieved without understanding of – their term – risk efficiency and use of cumulative probability distributions to pursue it.

A large body of literature exists on the topic of information processing with management applications, as presented in the review of Moorman (1995). This literature is informative and thought-provoking on the subject of transparency, but is not immediately applicable to transparency as it relates to risk management in product development. Another study explores the effect of information, or lack thereof, on project managers' decision-making (Bendoly & Swink 2007). The study concludes that greater visibility of situational information impacts project outcomes by affecting the decision maker's actions and perceptions regarding the behavior of others and the priority of the decision maker's task. These findings can be interpreted to suggest that transparency would work via a similar mechanism to impact product development outcomes. Unreported information is explored in a study of how choices are made on the relevance of risk information (Kutsch 2010). This work specifically explores deliberate ignorance of risks through a qualitative study of IT project managers. A review of previous work on ignorance and certainty, as well as a taxonomy of ignorance, is presented in this paper. The author concludes that traditional project risk management assumes "hyper rationality" of stakeholders and thus ignores aspects of managerial behavior, such as deliberate ignorance and judgment of relevance.

In one instance of a study on transparency in the context of product development, Wirthlin identifies five key characteristics the acquisition system values in an investigation of US defense acquisition program performance: cost, schedule, performance, transparency, and flexibility (2009). This expands on the typical view of product development as an endeavor driven purely by cost, schedule, and performance. The author identifies consensus building and desire for openness as the desirable effects of transparency. However, it is pointed out that within the Department of Defense, this transparency comes with burdensome approval and accountability functions.

In their book on project risk management, Cooper et al. suggest that transparency and traceability of risk management decisions is often a requirement established by effective senior management (Cooper et al. 2005). A case study is presented where increased transparency of risk management was achieved through online sharing of reports and studies to all stakeholders.

Transparency is featured as one of the eleven principles of managing "any form of risk within any scope and context" in the ISO 31000 risk management standard (ISO 2009). The ISO standard claims that compliance with these principles will lead to effective risk management. These principles are as follows:

(1) Risk management creates value
(2) Risk management is an integral part of organizational processes
(3) Risk management is part of decision making
(4) Risk management explicitly addresses uncertainty
(5) Risk management is systematic, structured and timely
(6) Risk management is based on the best available information
(7) Risk management is tailored
(8) Risk management takes human and cultural factors into account
(9) Risk management is transparent and inclusive
(10) Risk management is dynamic, iterative and responsive to change
(11) Risk management facilitates continual improvement.

The standard elaborates on principle number 9, on the topic of transparency, as follows:

> *Appropriate and timely involvement of stakeholders and, in particular, decision makers at all levels of the organization, ensures that risk management remains relevant and up-to-date. Involvement also allows stakeholders to be properly represented and to have their views taken into account in determining risk criteria.*

The inclusion of transparency in the broadly applicable ISO 31000 standard reinforces the findings of the previously discussed literature regarding risk management in product development and points to a potentially fertile area for further investigation.

3 Risk management survey

Given the number of risk management guidelines available to practitioners, and the lack of empirical studies of actual risk management use in the product development context, we sought to first test the correlations of a number of best practices with positive project outcomes with a survey of industry practitioners.

3.1 Survey development and dissemination

This section summarizes the work of Oehmen and Bassler in designing the survey (2011). They developed and tested a survey on the topic of risk management in product development over a period of six months. They worked closely with a focus group consisting of twelve individuals from three academic institutions, one risk management consultancy from the aerospace sector, and six companies from the aerospace and defense sector, all based in the United States.

The development of the survey focused on pre-filtering the questions as much as possible to only include risk management characteristics and practices, as well as risks and mitigation actions, which were agreed on as being best practice or of significant impact on the risk management process by general expert and practitioner consensus. Pertinent published guidelines were reviewed and consolidated for inclusion in the survey (NASA 2008; INCOSE 2004; Project Management Institute 2008; ISO 2009; DoD 2006). Notably for the findings of this paper, the ISO 31000 principles of risk management, as listed in Section 2.2, were each asked as questions in the survey.

The total time needed to complete the survey was approximately 45-60 minutes. The survey was administered online and distributed in two ways: First, the survey was sent to the risk management organization of a number of large aerospace and defense companies as part of a benchmarking process. Through this distribution, 90 complete datasets of the survey were collected. Second, the survey was distributed to practitioners through professional organizations and mailing lists. To encourage participation by shortening the response time required for survey completion, the survey was broken down into smaller parts according to respondent function: Part 1 with questions relevant for general program managers (i.e. respondents not working in a dedicated risk management role), and parts 2 and 3 with questions relevant for respondents directly involved in

risk management. The respondents that were binned into the 'risk manager' category were randomly assigned to one of the two risk management parts, with a 50/50 distribution.

In total, 375 responses of various degrees of completion were recorded over a period of seven months between March and September of 2011. Exact response rates are difficult to ascertain, as recipients were encouraged to forward the invitations to colleagues within their organization. Detailed information about the sample population and survey structure are provided in a related study (Oehmen et al. 2014).

3.2 Survey analysis and results

The survey collected extensive information (171 questions) about past projects, specifically regarding project outcomes and risk management process. The survey addressed methods and practices in the areas of risk analysis, risk evaluation, decision-making, and risk monitoring. Questions on use of practices and outcomes were asked on a five-point symmetric Likert scale, which consists of five discrete options ranging from "never" to "always" or from "strongly disagree" to "strongly agree".

The survey included high-level outcome questions which covered the traditional PD goals (success with regards to cost, schedule, technical performance, and customer satisfaction targets). Questions on outcomes were also asked on a five-point symmetric Likert scale, with options ranging from "complete failure to meet target" to "strongly exceeded our target". To simplify the description for this paper, we will focus our evaluation simply on the eleven ISO 31000 risk management principles. In related work, all 171 questions were analyzed and compared in similar fashion.

We then analyzed the data to see if there are any variables which correlate significantly with the outcome variables.

We tested the effect of each risk management principle on the outcome with the Effect Likelihood Ratio test. The effect likelihood ratio was calculated using the chi-square statistic, which is a test for the difference when a model includes all variables versus when the variable of interest is removed. The test measures how much more likely the outcome data is to be from one model or the other. If the p-value is non-significant (in this case, greater than $\alpha = 0.05$), then there is no

statistical significance for the variable of interest to be included in the model. Table 1 lists the calculated p-values for each outcome model. Significant p-values are bolded.

Table 2 below presents a breakdown of the responses to principle 9, in the form of the question "Our Risk Management is transparent and inclusive towards all stakeholders" with the corresponding average PD Target scores (the mean of the answers, asked on a 1-5 scale). The increasing Mean PD Target Scores presented in Table 2 indicates that indeed there is a trend in the degree of transparency/inclusivity and the product development target achievement.

The results in Table 1 show that there were significant associations with project outcomes for a number of risk management principles; this finding suggests to the authors that each of these principles warrants further investigation. In this case, we chose to continue our investigation with a focus on transparency (principle 9), given that it is non-obvious and has been little addressed in the product development literature.

Table 1. Explanation power of each of the eleven ISO 31000 risk management principles (see description of principles with numbered list in Section 2.2) on each of the outcomes, as represented by the ordinal logistic chi-square effect likelihood ratio p-values. Each p-value less than or equal to $\alpha = 0.05$ is significant and is bolded.

Product Development Outcome	*ISO Risk Management Principle*										
	1	2	3	4	5	6	7	8	9	10	11
Cost target	0.86	**0.02**	0.87	0.12	0.22	**0.03**	0.70	0.32	0.20	1.00	0.11
Schedule target	0.63	0.18	**0.03**	**0.04**	0.24	0.16	**0.04**	0.44	0.15	0.39	0.57
Technical performance target	0.71	**0.01**	**0.03**	0.80	0.32	0.07	**0.01**	0.12	**0.03**	0.21	0.11
Customer satisfaction target	**0.02**	**0.00**	**0.00**	0.19	0.06	**0.05**	0.34	0.38	**0.02**	**0.00**	0.15

Table 2. Breakdown of 195 responses to the question: "Our risk management is transparent and inclusive towards all stakeholders." For each group of responses to this question, the mean of the four product development (PD) outcome dimensions is also presented.

	Our RM is transparent and inclusive towards all stakeholders.				
	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
Number of responses	7	33	55	83	17
Mean of four PD outcomes	2.46	2.59	2.82	2.94	3.55

4 Interviews

The statistical results suggest a correlation between transparency and project success but do not show causation or provide explanation. A more in-depth and exploratory study was needed. The qualitative research interview method has been found to be an ideal way to examine topics where different levels of meaning need to be explored (Bouwen et al. 1994), as was the case in this study of transparency.

The interview work done here builds upon research previously done by Alison Olechowski as part of her S.M. thesis at MIT (Olechowski 2012). This thesis broadly analyzed best practices for risk management in product development processes based on a large quantitative survey that was carefully constructed and was completed by 375 professionals over a six-month period. The quantitative analysis of these results revealed a strong correlation between success of product development efforts and the level transparency in an organization. Also as part of the thesis, Olechowski performed some qualitative research into the role of transparency within various product development teams.

The interviews and qualitative analysis performed as part of this research are an effort to build on top of Olechowski's work in order to better understand exactly what is meant by transparency and how it is manifested in product development organizations in various industries.

The interviews provide several results whose goal is to improve understanding of the key factors that affect transparency in the product development process of an organization. Because the results are based on a limited number of interviews, the results are not prescriptive or necessarily generalizable to all product development teams. However, several key themes and results clearly emerge from the data, which implies that these themes are likely applicable in some way across similar organizations.

4.1 Method

Qualitative data was collected by means of 15 personal interviews of industry practitioners conducted both over the telephone and in person. Most of the interviews were recorded for convenience and accuracy of transcription. The interviews were performed between March and October of 2014. Each interview subject had several years of industry experience working in

product development teams, and typical titles included Product Manager, Project Manager, Software Engineer, and System Engineer.

Appendix A contains the interview templates and list of questions used as a guide for the various interviews. Appendix B contains the consent form that subjects were asked to complete. Appendix C contains full transcripts for each of the recorded interviews.

The qualitative data obtained from these interviews was used to produce a high-level definition of transparency. The data suggest a useful distinction between two concepts within the broader topic of transparency: *surface transparency*, which leads to a shared understanding of information and assumptions, and *deep transparency*, which leads to alignment and trust among team members.

This was then decomposed further into a framework of *characteristics* and *enablers* for transparency in product development teams. *Enablers* of transparency are specific actions that an organization can take to encourage transparency within their teams. *Characteristics* of transparency are desired states that interview subjects directly associated with transparency and that have been observed to occur when one or more *enablers* has been successfully applied within a team.

4.2 Interview subjects

The following is the list of subjects interviewed as part of this study, numbered from S1 through S15, along with their titles and industries:

S1. System Engineer, Aerospace and Defense
S2. Software Architect, Enterprise Software
S3. Chief Solutions Architect, Aerospace and Defense
S4. Product Manager, Civil Engineering
S5. Product Manager, Consumer Software
S6. Product Development Engineer, Automotive
S7. Director of Engineering, Internet Software
S8. Senior Software Developer, Enterprise Software
S9. Program Manager, Consumer Electronics
S10. Software Engineer, Internet Software

S11. Project Manager, Transportation

S12. Product Manager, Commercial Electronics

S13. Program Manager, Automotive

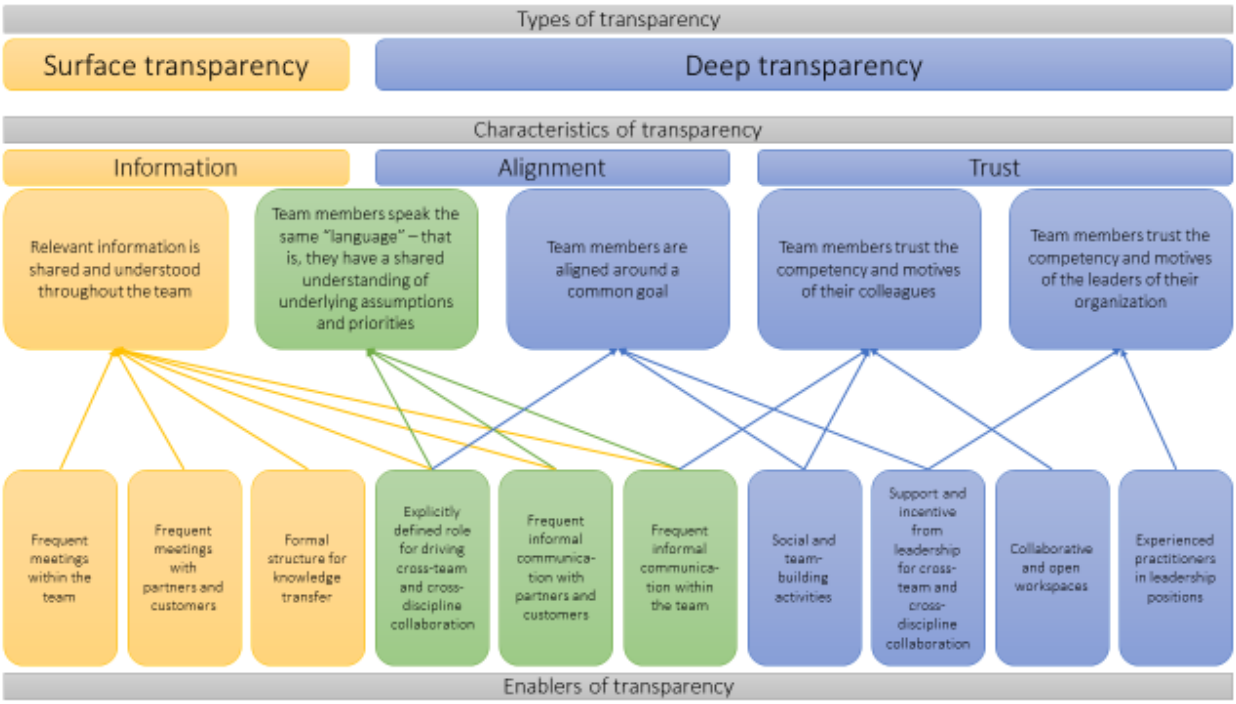
S14. Principal Software Engineer, Enterprise Software

S15. Product Manager, Aerospace and Defense

Throughout this paper, quotes will be attributed using the subject's assigned number from S1-S15.

4.3 Interview analysis and results

The diagram in the figure below represents a possible decomposition of the topic of transparency, first into the high-level concepts of *surface transparency* and *deep transparency*, and further into sets of *characteristics* and *enablers* that were identified over the course of the interview analysis. The figure also contains arrows that link *enablers* to *characteristics*. These arrows denote that one or more interview subjects indicated a possible cause-and-effect relationship between the *enabler* and the corresponding *characteristic*. The remainder of this section will explore each of these *enablers* in more detail.



5 Surface transparency: Information and assumptions

The concept of *surface transparency* can be defined as follows:

> *The sharing of relevant information among members of a team or organization so that the full set of relevant information is being communicated, and so that all parties understand the information as communicated in both a literal and technical sense. This information may include items such as project status, product designs, technical data, and so forth.*

A key theme that emerged repeatedly over the course of this study was the idea that transparency in the product development process is often hindered by a lack of real understanding when two groups of people are attempting to communicate. Of course, there are many reasons for this in practice, but the interviews revealed two main types of understanding that are often lacking.

The first and most readily apparent type of communication problem is a lack of literal or technical understanding between two parties. This can happen in the obvious cross-team cases, for example, where engineers are attempting to communicate with salespeople, or when executives are interacting with customers. But it can also occur in communication within teams. Engineers with different technical backgrounds may have trouble interpreting each other, and cross-discipline teams that include project managers or product managers may also run into this issue due to lack of shared expertise.

The second and more subtle communication issue is caused by differing underlying assumptions or priorities with regard to the product, the process, or the business. These problems can be much more difficult to solve. Often two groups of people will fully understand each other's literal and technical messages, but will reach entirely different conclusions because they are interpreting the data based on their own worldview of what is important or what is a priority for the business.

This section will address both of these types of communication issues, and it will also describe some mitigation strategies for each of these issues that interview subjects have observed in their experience.

5.1 Frequent meetings within the team

Fundamentally, when two groups of people want to communicate, they must express themselves by forming words into logical structures to represent the idea they wish to convey. But clearly, the communication will only be successful if both sides have the same understanding of what those words and logical structures represent. In other words, the semantics of the conversation must be understood by both sides. Depending on the identities and backgrounds of the two parties – that is, whether they are engineers, salespeople, managers, or customers – the problems encountered here, as well as the possible mitigation for these problems, may be very different.

5.1.1 Communication between engineers in different disciplines

When two engineers with different backgrounds want to communicate about a technical topic, there is often a fundamental disconnect based on the many years of technical expertise about the topic that is available to one person but not the other. As a result, issues that are being raised by one side may not actually be understood by the other side:

> *"You've got cross-functional teams. You've got hardware engineers talking to software engineers talking to chemists talking to managers. You're not really saying things that people understand. If I started talking about moment arms, a chemist isn't going to understand that. A software engineer starts talking about classes and scripts – that's great, wonderful. You aren't actually able to pick out if there's a problem." S1*

Not only do the technical details vary for engineers from different disciplines, but there are also vast differences in the processes used in the various professions due to the differing nature of the work. Engineers in some disciplines are used to fast-paced work with frequent requirements changes, while others are used to a much more formal process:

> *"Let's compare, say, software and electronics, which used to always be a clash at [my company]. For software it was always possible to make changes, and it was hard for software people to see why the electronics guys would not make changes, even though it seemed like a pretty straightforward change. And vice versa as well. Electronics guys saw it as really, really unthinkable to have changing requirements, whereas for the*

software team, it was a lot more acceptable that things changed further down the line." S9

5.1.2 Communication between engineers and management

In many companies, engineering managers do not have the technical expertise or experience required to effectively manage risks or interpret the concerns of their engineers. As a result, knowledge does not flow freely up and down the management chain:

> *"Directors are supposed to be the intermediaries between the executives and the senior engineers. But they don't have enough technical expertise to be an effective filter or to transfer knowledge between engineers and executives." S8*

In larger or more mature organizations, management often becomes a separate function that is removed from engineering, which exacerbates this problem. As more intellectual or technical distance is introduced between engineers and their managers, communication up the chain becomes harder:

> *"[Managers] don't understand details. You'll see this kind of thing happens a lot more in older organizations where you have management as a function by itself, and not so much in younger organizations where you have mostly young managers who have just moved up. A startup I worked with didn't have those kinds of issues, whereas at [my company] it was a regular thing. It could be that in big companies, managers don't have an understanding of the product itself." S9*

Most interview subjects also reported that their engineering teams struggle with communicating the right amount of relevant information to their directors or executives – that is, the management that is several layers above them. Often this results in lack of effective understanding due to a combination of lack of expertise on the executives' side, and oversimplification on the engineers' side:

> *"The engineering team tends to package information so that it is digestible to the directors. Because of this, it is often oversimplified, especially since executives don't understand the nuances of everything that is said. This problem is compounded when the*

executives then relay the information on to someone else – information can be completely lost in the process." S10

5.1.3 Shared representations as a vehicle for communication

Another way to mitigate the risk of literal or technical misunderstanding is to establish a shared representation as a communication vehicle – that is, an agreed-upon format which can accurately represent the information that one side is conveying, and which can also be easily understood by the intended target.

In status updates with executives, this can take the form of a particular presentation format or a certain type of visualization:

> *"There are some executives who are very happy with their monthly, PowerPoint-based status report where there's like a traffic light and an explanation. They really don't want to know the details of what the risks are unless they are expected to act. More recently we've had some executives who are quite a bit more interested in making sure they understand the day-to-day operations, and they have devised a bunch of their own methods for communicating status. One of our leaders now likes to have a specific visualization format that he calls the 'heat map' where the core topics that are at risk or of concern are raised through this format, so that he can always see at a quick glance what are the most important topics to pay attention to." S2*

Because typically engineers and executives are both familiar with the product that the team is building, a demonstration of the product itself can also be used as a shared representation which can improve understanding in communication:

> *"One more interesting way we've been communicating our status lately is doing weekly status demos for the executives, which is actually quite a bit of work for the development team. But one thing it does do is ground the conversation on what is actually being done, so it allows the executive to have some level of understanding of what people are doing, and therefore be able to ask probing questions interactively. It allows them to be a lot more engaged and claw for information rather than having it prepared and passed on to*

them. We've definitely identified interesting issues that we probably wouldn't have identified otherwise." S2

Within the product development team itself, it is important that team members from various disciplines can have an accurate sense of project status – and more importantly, that they agree on this status. Many organizations use ticketing systems to help solve this problem:

"One of the things that was important was the visibility of the request from end-to-end. It's always nice to give the developer actual context to know what they're solving. So instead of having a document that they never read in a system that they never access, they develop against a single ticket that originated from the customer. This actually puts things in the context of how the information is used, so you can do a traceability all the way up so the QA guy that finds the problem, he's doing it all in the context of that single ticket." S3

"We use Jira for ticket management. This is very effective and is visible to everyone within the software team. It allows everyone to keep track of issues and promotes excellent visibility of project status throughout the team." S8

Demos or prototypes are also an effective tool for promoting understanding between the product development team and its customers:

"The Technology and Innovation team would only approach the cities and the consulting engineers if they wanted to test or pilot something, when they needed a site test or a third-party innovation from these consulting engineers. This only happened on projects that reached the point of prototyping. Prototyping on-site worked out well. That's the best way to prove the product in the market." S4

"For communication between senior engineers and customers, demos and presentations at the customer site are helpful. It promotes effective two-way communication and helps to keep the line of communication open." S8

It is worth noting that using prototypes as a shared representation is not without its risks and drawbacks. Not only does it create additional overhead for the engineering team, but it can also lead to severe miscommunication if it is not managed properly:

> *"Showing unfinished prototypes or demo code can be problematic and lead to miscommunication. Directors see this and think it is the real product. This can lead to deadline pressures being applied to the engineering team. Also, when the engineering team faces deadline pressures, they tend to report inaccurate status in terms of percent done – they will report that the project is farther along than it actually is." S10*

5.2 Frequent meetings with partners and customers

In addition to communication within the team, the interview subjects indicated that communication and transparency with partners and customers was often a critical aspect to the efficiency and success of the project.

5.2.1 Communication between engineers and customers

In communication between engineers and customers, there is often the classic problem of customers understanding their own desires without knowing the engineering details or effort required to implement a solution to those desires. So engineers always have to add their own layer of interpretation to what the customer says, which is often based on incorrect assumptions or missing information:

> *"The customer rarely gives you a business requirement that includes terms of throughput and latency. They'll say, 'This data format needs to be transformed in a certain way to this data format.' But they don't say it needs to be done within 300 milliseconds at 2000 transactions per minute. And very few people understood how the customer used the system, how they used the product simulation and the ordering." S3*

Many interview subjects observed that the fundamental problem here is that customers and engineers often speak different languages. They have vastly different technical backgrounds, to the point where what a customer says may be of little use to the engineer:

"The language spoken [between customers and engineers] is different. Let's take an example of something very simple, like a website or a server. The language that's used for specifying requirements in engineering – what's important for an engineer to know – is different from what the customer thinks is the key part of it. How they describe things are different. Maybe that's changing over time. But in the cases where there are problems, I think it's just because the language is different." S9

5.2.2 Boundaries between engineers and customers

Another theme that appeared repeatedly over the course of this study was the idea that it is often beneficial for the engineering team to be separated from the marketing team and their customers. One reason for this is that it is beneficial for the company if only one group of people is regularly communicating with the customers, so as to avoid sending conflicting messages:

"Here's one specific example from [my company]. They had a very stringent policy on open engineering contact with customers. And the reason was that people would inadvertently say stuff which would result in the customer getting something which becomes a negotiation point. For example, they'd say, 'Yes, that was a bug, we just sat on it and made it work somehow.' Meanwhile, the sales guy had pitched the same thing as a feature created just for the customer. So, language is different, and there are some things that need to be hidden, and some things that don't." S9

In many companies, engineers simply don't need to know what happens in marketing or have any contact with customers, and adding communication with those groups would only be an irrelevant distraction:

"[The business side of the organization] is irrelevant to engineers. It makes zero difference to anything on a day-to-day basis or a month-to-month basis. And most engineers, when they transfer from one place to another, it's not so much the business part, but the functional skills. So going to another place and saying that you know how they make money – that has very little value in engineering." S9

In other cases, meetings with customers may result in so much overhead for the engineering team that it significantly hinders their progress on the project itself:

> *"I think another piece was that there were a lot of top-down requirements to have a lot of meetings at a high level to present to our DARPA customer. So that led to not a lot of work getting done because we were preparing for those meetings and trying to communicate that we had done a lot of things. But really what we did was prepare reports and prepare for meetings." S15*

Of course, this is not true in all cases. Often, the engineering team has a legitimate need to communicate with customers, particularly in smaller companies or firms doing contract-based work:

> *"Having engineering removed from customers tends to work well for R&D-based work. But having engineering removed from customers would not work well for a more contract-oriented organization because it would greatly increase the turnaround time for customer requests and would limit direct contact with the customer." S7*

5.3 Formal structure for knowledge transfer

The second key theme that emerged from this study is the role that organizational structure plays in promoting effective transparency and communication between teams. The way an organization is architected has impact far beyond the simple management and reporting chain that it sets up. When done correctly, the structure can serve to reinforce desired communication patterns and discourage unnecessary or destructive ones in a way that improves efficiency and transparency across the company.

The most prominent transparency-impacting feature of organizational structure, based on the interviews done for this study, is the set of boundaries that are constructed to limit undesired communication. Because transparency is maximized by increasing *effective* and *useful* communication, it also necessarily implies that an overload of communication, or incorrect communication, can actually be detrimental to overall transparency.

Just as too little communication can lead to lack of effective information transfer, too much communication can also lead to lack of effective information transfer. In other words, the effectiveness of communication can actually be hindered by communicating too much, or with too many people.

Organizations have several issues to address if they wish to deal with this problem. First, information must be filtered in some way before it travels between teams or up the management chain. Second, teams must be given clear guidance and directives to prevent churn. Third, boundaries should be put in place to prevent communication or information sharing in cases where it is not relevant or could be detrimental to the product development process as a whole.

5.3.1 Effective processes for bubbling up risks

Managers play a key role in filtering information as it travels up the management chain. In a well-run organization with effective managers, this means that individual engineers rarely need to communicate with anyone higher up the chain than their immediate manager:

> *"Each engineer works within a Scrum team, and that Scrum team has a Scrum master. He's responsible, along with the product owner, for helping to resolve blocking issues and addressing concerns of the engineer. And then we have a weekly Scrum of Scrums, which the engineering leads manage, and they find out [potential risks] through that mechanism. And it gets bubbled upwards that way. It's pretty rare in our operating environment that there are urgent issues that get escalated without running through those channels over the normal course of time." S2*

Because of this boundary that has been established, managers in a given division of the company also must play a key role in sharing necessary information with each other:

> *"Each director in the division takes part in regular sprints, whose primary goals are to identify and plan for dependencies across the various teams, and to reduce the need for inter-team communication as much as possible." S7*

5.3.2 Established communication channels to reduce noise

Establishing a formal communication process can also be helpful, especially when dealing with cascading communications where one team is typically consuming the output of another team:

> *"The communication between product development and design is good because we have to deal with them a lot, but it's typically only one-way. And it's formal; you have to have different design review meetings and set up agreements. They have to cascade your*

official requirements for aesthetics – parts that are very noticeable on the exterior – and there are certain assumptions." S6

Another way to limit churn and promote more efficient communication is to create an organizational structure where teams communicate with only a limited set of people within their own team:

> *"Each individual team has its own sprints, all of which are currently done in parallel. They try to be independent and communicate only at the director level. Limiting communication between the various engineering teams improves throughput by preventing unnecessary interruptions." S7*

5.3.3 Elimination of inappropriate communication boundaries

The risk of placing boundaries in the organizational structure is that, like anything, it can be taken too far. There are many cases where communication between two groups could increase the efficiency of the product development process and add value to the firm. Organizations should think carefully before instituting such boundaries.

The engineering team, while they may not need direct contact with customers, must still have a way to receive information and feedback from customers. This means that the information must somehow flow from the customers into the organization and eventually to the engineering team. If that information flow is not there, customer problems will get lost:

> *"I would have added more input from field services into Technology and Innovation team. That communication was basically non-existent. They should have communication because field services is where you have all the problems. So I think they should have more visibility there. The executive team should filter those field problems better – either get it directly or get it through product management." S4*

A formal, cascading process for communication can also lead to problems if the development process itself is actually iterative:

> *"The early assumptions of marketing and communicating to product development should be improved. A lot of uncertainty is laid out in the beginning of the product. By forcing you to accept something at the very early stage of product development, you're prevented from exploring your design tradespace. The requirements should be flexible enough that further down the product development process, we are able to adjust as required by the market." S6*

A side effect of putting boundaries between various engineering teams is that their managers and directors become the communication bottleneck. As a result, if the right communication doesn't happen at the management level, it can lead to inefficiencies:

> *"The directors often have commitment conflicts between their own teams' sprints and the all-up director-level sprints. Individual teams try to be independent and communicate only at the director level, but this often results in deadlocks. For example, the analytics and R&D teams may each be blocked, waiting on each other." S7*

Sometimes customers may actually have much more of a technical background than the marketing or sales team. In these cases, it is probably a mistake to completely firewall the engineering team from the customers, since the engineering team likely contains the only people who can correctly answer the questions and concerns coming from the customer:

> *"The people doing the communication may not understand the underlying facts or products well enough. Like, if you take something, say a router, and there is a marketing team and a client who wants to buy the router, and there are a bunch of things that the client says he wants – unless the marketing person is somebody who has a decent understanding of routers, chances are he gets something altogether wrong. It's a complex enough thing that, unless you understand it well, you can get it wrong." S9*

Establishing communication boundaries can also hinder the development of personal relationships between members of various disciplines, which as we saw previously, can have a negative impact on overall understanding and effective transparency:

"Product managers at [my company] sit separately from the engineering team, which decreases contact and discourages development of close relationships. They have technical program managers who sit with the engineering team, but these are really just project managers who coordinate multiple engineering teams, and don't actually work with the product management team." S10

6 Deep transparency: Alignment and trust

The concept of *deep transparency* can be defined as follows:

A common understanding of goals and priorities among all members of a team or organization, in addition to a shared trust among team members of each other's competency and motives.

The literal and technical understanding that comes with surface transparency is a necessary precondition for the existence of deep transparency in a team or organization.

Evidence from the interviews shows that deep transparency can be broadly broken into two characteristics. First, alignment among members of the organization is critical. People must be using the same language and working toward the same goals in order to maximize the chance of success for the project. Second, trust among team members is important for encouraging an open and collaborative culture and to improve transparency overall.

6.1 Alignment: Shared language and shared goals

A key mitigation strategy for these types of literal and technical understanding problems is to establish a process that ensures that the two sides of the communication link are using common language and therefore have the same end goals.

One way to do this is for engineers to package their status and report risks in a way that is fully digestible and accessible to management:

> *"The managers typically didn't know what was going on, which I would think is probably commonplace for most organizations, at least on the day-to-day. Anyone could look at the schedule. The people who mainly looked at the schedule were the managers. That was the biggest tool for them – that's how they knew where the [project] was. They said, 'Look, we're 50% complete here – great.' That was probably the only channel to get red flags out to management – through the schedule and them seeing something was behind." S1*

If there is a manager or executive who actually has technical background and thus can understand the language used by the product development team, this can be a huge benefit – especially if that person is then able to interpret the information for the benefit of the other executives:

> *"Product managers had better communication with the process director than the rest of the executives because he understood our problem. He had the most experience in the industry for the chemistry and the process behind the product. He was a big advisor. Everything related to process would go through him. [The rest of the executives] only cared about sales from product managers. They didn't care much about technical because they had no idea about technical. If I had a technical question and I went there, they would send me to the process director." S4*

Some companies actually enforce policies to ensure that those working in their executive departments must have had recent, relevant experience in the product so that they can understand and speak the same language as the product teams:

> *"There is a department where they drive initiatives coming from the CTO across the breadth of the product line. Product managers must have worked on some other product before coming to this department. That's how they can think across the breadth, because in the last few years they must have worked on each of the products or touched some aspect." S5*

In addition to speaking the same language, having a single, clear source of direction is critical to ensure cohesion within a team and to reduce the possibility for confusion and churn:

> *"The executives would tell the Technology and Innovation group which markets or which areas they should look at based on the advice of the process director. So we would pick the key products based on the tendencies that they felt the industry was going toward. The advantage is that they can have a single direction. There's no confusion as long as they have a clear message." S4*

6.1.1 Explicit role for driving cross-team and cross-discipline collaboration

Interview subjects observed that having an explicit role in the company for people who drive cross-team and cross-discipline communication often leads to much more effective transparency. This role is often referred to in various companies as product management or program management, and having competent people in this role can be critical for enabling communication across the various teams in the organization.

The interviews conducted for this study strongly suggested that having an explicit product management or similar role in a company can provide significant benefits with regard to collaboration and communication across teams and disciplines. Many companies include this role as part of the product development team itself, which improves communication both within the team and across teams. Product managers with experience in the industry can also serve the role of technical expert and can act as an effective filter or firewall between the engineering team and the outside world.

In some cases, product managers can facilitate communication between two disciplines that otherwise may not understand each other or have incentive to collaborate:

> *"Engineering does not care about design. Information from design goes through product managers. But in the absence of product managers, engineers don't care what design does. The organization is driving a design focus, so that perspective is there, at least with the engineering manager. But down the line, I don't think there is anyone driving it so deliberately. In the absence of product managers, I think that design and engineering have a real disconnect." S5*

The product management or project management role can also serve as a translator between customers and the engineering team, since the ideal product manager has some relevant experience on both sides and is appropriately incentivized to improve the efficiency of the communication:

> *"Project management helps to understand and resolve disparities between engineering goals and customer requirements. They used to be part of sales and marketing, but having project management be part of the CTO organization benefits the engineering team because their accountability is for engineering results." S7*

It is worth noting that the product management role tends to be more prominent in younger, more nimble companies. One possible explanation for this is that product management is crucial for enabling efficient and fast communication across the company:

> *"The role is called different things in different places. Sometimes it is called product manager, sometimes it is called program manager, in some places it's called project manager. It's basically enabling this communication between different groups of people, and then at the end of the day having ownership of the overall thing. It's only in the last five-to-ten years that this has become a structured function by itself. At old companies – for example, HP or Microsoft – it follows the traditional engineering structure. You have marketing and you have program management and you have engineering teams. But a lot of the younger companies – like Google, Facebook, Amazon, or Yahoo – have this separate function that does that. And the fact that more and more young companies seem to be having a dedicated role for it shows that it's an important thing." S9*

While most often we think of product management as a role in the software industry, there is nothing about it that inherently limits it to the software industry. Rather, it could be a useful role for any company in a fast-changing market:

> *"I definitely think this concept applies outside of software as well. I think it's applicable in any fast-changing environment. That's maybe why the older companies didn't have it, because the rate of change of technology wasn't as fast as it is now. So if things don't change as fast, you don't need that level of product management. But when things do change, then you need to keep your fingers on what needs to change. Because if things change on a year-to-year basis, it's more important than if things change on a five-to-ten year basis – like cars, for example. The more volatile the industry or the product feature set – the further away from dominant design – the more value product management brings." S9*

Several interview subjects mentioned that personality and passion play an important role in the effectiveness of a product manager and their ability to catalyze information flow:

> *"The best product managers are passionate about technology and have an understanding of what users want. They develop strong relationships with all disciplines – test, development, and design. They use the product themselves and are able to learn from it and extract useful information. Product managers should serve as a conduit for good ideas to flow from all employees in the organization no matter where they originate." S10*

Whether a team has a product manager or not, it still appears to be important for that team to have a designated point-of-contact when communicating with other teams. It ensures consistency in communication and provides a predictable flow:

> *"The product managers recognize themselves as a product management community. So if there is a meeting and the engineering manager comes to push something forward, the product manager will ask to talk to his product manager. And the two product managers will talk to each other. So they emphasize the product management community to facilitate the conversation." S5*

Having someone whose role is explicitly focused around enabling communication can also help to solve problems like working on projects with cross-geographical dependencies:

> *"In cross-company projects, project management is the main conduit between divisions, along with the other directors. Because these are usually cross-geo projects, embedding project managers across teams – long-term assignments at other sites – speeds things up considerably." S7*

These designated points-of-contact do not need to be product managers. Often they are simply engineering managers or even individual engineers who take on a role that is very similar to product management for a particular communication link or project dependency:

> *"Having a main contact point on a team is critical. This is usually the manager at first, though the actual communication usually ends up getting delegated to an individual engineer. In my experience, teams with lots of cross-team organizational dependencies need to be very structured in terms of how they store and share their information. One*

practice at [my company] which turned out to be effective was having 'ambassadors' who work closely with both sides of a communication link, for example, design and engineering, or marketing and engineering. They are sort of like specialized product managers whose main goal is to ensure that communication is happening smoothly." S10

6.1.2 Frequent informal communication with partners and customers

Having frequent informal communication can improve the two-way relationship between the engineering team and customers:

> *"The senior engineers have weekly Scrum calls with the customer, which leads to effective two-way communication. The engineering team itself is agile. They sit together and have daily standups, which leads to excellent communication within the team." S8*

Developing relationships through informal methods can improve the company's overall relationship with the customer by improving information flow:

> *"Product managers communicated most with the consulting engineers. They were the ones who had to sell my product, so I had to give them all the information. They think that you're only selling them a product and that you don't really care about their job. So they will just tell you what they want. You have to become their friend for them to tell you everything." S4*

6.1.3 Frequent informal communication within the team

Frequent communication via informal methods like scrum-style meetings or hallway conversations can lead to an improvement in transparency.

> *"The scrum meetings were useful, especially when we were in a team of 7-12, it was really helpful to know who was working on what and to make sure that we got a sense of what we were going to have at any given time, since we had to communicate that to the larger program to manage expectations and integration and other things." S15*

Another benefit of frequent informal communication is that it can reduce the need for formal meetings and the overhead associated with them:

"The informal style is less structured and tends to lead to more open dialogue and more outside-of-the-box discussions, whereas [formal] reporting is usually shaped in the form of a template and you tend to get more structured answers. If a team is co-located and is always communicating, you don't necessarily have to go to the more formal, written report. You can have the status instantaneously." S11

6.2 Trust: Competency and motives

Competency of colleagues was repeatedly mentioned as a factor that impacted trust among team members. One key reason for this is that competency improves efficiency for the whole team:

"It's much easier to be transparent with people who are easy to work with, and I don't really like working with people who are hard to work with. Being easy to work with means that you, first of all, know about the technical side of things, so you can understand. And second, it means that you don't waste my time with unnecessary information, and I don't have to explain things to you multiple times. You understand what I'm saying and you know what action needs to be taken as a result. It's really about efficiency and knowing that the person will do their job." S14

Trust among team members also improves transparency and people's willingness to be open and talk through issues:

"I think that from a communication or relationship standpoint, I think it's important that people trust the other people on the team, and more importantly, the project manager, if they're going to take the time to be more open and talk through things. Trust means competency and being able to do something about it. Trust has to be built over time. You work with someone and you see that when you bring up an issue or are blocked on something, or if one engineering discipline is causing the holdup, then they can go and get that smoothed out. If they see that it's effective, if it helps them to bring things up, then they'll be more likely to do that." S12

Several techniques were identified that can help to build trust and openness within the team, including social and team-building activities, support from leadership for cross-discipline

collaboration, open workspaces that encourage collaboration, and the placement of experienced practitioners in key leadership positions.

6.2.1 Social and team-building activities

Getting people to an environment outside the office may allow people to be more open and help to improve trust among team members:

> *"It's useful to go away from the normal environment and go on an outing or something, maybe a more social conversation. People may be more honest with you if they're not at work and feel more relaxed. They're more likely to speak up if they have a problem or an issue, when they may not normally do that." S11*

Developing relationships at social events may also improve working relationships by encouraging more direct, informal communication rather than relying on formal methods:

> *"There were lots of social events where people are doing happy hours to celebrate personal achievements or project launches, which brings people together a little bit. Personally, I've always tried to get to know certain people that I work really closely with. I'd much rather people feel comfortable enough to come directly to me and talk about something, or if I need something from them I can go straight to them, instead of having to have something happen some other way." S12*

6.2.2 Support and incentive from leadership for cross-discipline collaboration

Having the appropriate organizational structure and incentives for collaboration and trust within teams is very important for product development projects:

> *"This type of communication is critical to the success of projects. Companies don't always recognize its importance or reward it properly, since it's more about collaborating and helping others rather than directly producing results yourself. It's very difficult to measure, and if a company is looking at the long list of things they need to improve, this is not likely to be at the top of the list. But I think people need to be properly motivated to work in a collaborative way." S14*

6.2.3 Collaborative and open workspaces

Many interview subjects reported a link between open workspaces and improved collaboration. For example, having an open workspace leads directly to more informal communication, which can improve transparency and reduce unwanted surprises:

> *"I sat a few cubes away from the director of product marketing. There was more informal communication in the elevator about issues, or something that is going well. That informal stuff keeps them in the loop. I almost never emailed the director, he gets so many emails, has an admin filtering his emails – I'll just talk to him. Those informal updates, even just really quick, help to avoid major surprises." S12*

Having an environment for easy collaboration is also helpful for reducing communication barriers:

> *"You can communicate easier when you are sitting together. You have more tools available to use. Even on WebEx you can draw on a whiteboard, but unless you do that day in and day out, you're not familiar with it. If you only do this once every couple months, you don't have the skills to make it work. But a whiteboard, doing it in front of people, is easier. So it's less effort." S14*

6.2.4 Experienced practitioners in leadership positions

A method that has been observed to improve this type of transparency is the placement of technical experts in key communication and leadership roles. People with extensive industry and technical backgrounds are much more likely to interpret information correctly and filter it so that the appropriate amount of information is passed along to those who need to know:

> *"Status update meetings where we present the risk – a topic and an issue – give the executive sponsors a good appreciation of the problem. Obviously the challenge with that is that we're not able to do that for everything, either because of the project team's schedule, or more usually because of the executive's schedule. Our project leads are experienced enough to figure out when a risk requires executive attention – when they can actually do something about it or need to know about it. So there's a lot of filtering that happens, for sure." S2*

There is also no substitute for experience. Involving experienced individual contributors in key communications can help to prevent technical misunderstandings or miscommunications:

> *"Experience is an important factor. There is one team member who had 15 years of experience, but still a designer, since that's what she liked to do. So she was an experienced person. More experienced than a product manager. So they always kept her on the emails. She was there for every email from the director." S5*

Designating a single person as the communication hub for the team can also have other benefits, including an improvement in efficiency and cross-team relationships:

> *"The obvious advantage of having a specific person doing the communication is that it saves time for the entire team. Most engineering teams hate sitting in meetings, so it saves time for all the other people if you have somebody who can understand and extract relevant stuff out. Not necessarily a manager. If you have somebody who is an expert on the topic, who acts as the communication face, it makes a lot of sense. And you project a good image of the whole team when you send somebody who is a subject matter expert. A second huge advantage of having one person is that when there are groups of people having a discussion, sometimes tangential discussions kick off and there are arguments. Just having one person makes it a lot smoother to run through a list of things. So that's the advantage of having just one person as opposed to having the whole team communicate with each other." S9*

Leaders with experience as individual practitioners can also collaborate more effectively with each other, which can improve the overall efficiency of the organization:

> *"These two directors, if there was a major issue or conflict between marketing and R&D, sometimes there is a tradeoff, and sometimes R&D says it's nine months late, and that's not ok and we'll need more resources. We'd escalate that to the director. Our directors worked really well together. Things would get escalated if they needed to, and the two of them had a lot of meetings behind closed doors. They were always able to come up with a plan together and roll it back down. They're looking at all the projects and are*

ultimately accountable for the business. Both of the directors were engineers at one point, they've been at the company for a long time, so even on the R&D side, [they] understand the architecture and overall technologies." S12

7 Future work

This section outlines several additional themes which were revealed by the interviews for this study, along with relevant quotes from the interview subjects. Each of these themes was addressed by multiple interview subjects and likely contains valuable and interesting topics for future research in this area.

7.1 Limitations

Some limitations are important to consider when interpreting the survey results. The survey is taken post-project, so accurate recollection of program details may be difficult. The analysis relies on self-reported outcomes which could be biased by the experience of the respondent. The survey was self-administered online; to address potential misinterpretation of the questions, clear descriptions and examples were included throughout the survey and opportunities were given to comment on ambiguity of individual questions. Although the sample included a diverse mix of product development projects, the empirical findings from this data set are not necessarily generalizable beyond this sample.

The potential for self-selection bias exists in our survey and interviews. Those with an already strong opinion about risk management may have been more likely to respond or participate.

7.2 Impact of communication frequency on transparency

Several interview subjects indicated that they have noticed a correlation between the frequency of communication (daily, weekly, monthly, etc.) and the level of transparency that is reached as a result of that communication:

> *"There were a couple different types of check-in meetings. There were small-team meetings, which were just engineering staff and the engineering lead. These happened daily or every other day. Then there were big-team meetings, which were everyone involved with the team. These happened every two or three weeks. They were mainly status updates. Every time I went there it was a waste of time." S1*

> *"There was very little communication between the product managers and the Technology and Innovation team. They were terrible. They would just tell us what they*

were doing when they needed something from us. They did not talk to us. There were no weekly meetings or biweekly meetings, [only] monthly meetings." S4

7.3 Impact of knowledge sharing methods on transparency

Organizations often have specific processes or methods for sharing knowledge. If implemented appropriately, these techniques can serve to improve transparency both within a given team and across teams:

> *"The way we share information has changed quite a bit depending on who's the current leader of the organization. Within the team, the level of information sharing has always been fairly high. We have standardized reporting, and we also have wiki pages where people are pretty good at keeping each other up-to-date on where they're at." S2*

> *"Different people like access to different levels of information. [My company] had this policy of open information, which I thought was a great thing. Everybody had access to all the information. I think the policy of open access to information plus the internal culture develops something positive for the company." S9*

> *"For teams with lots of cross-team organizational dependencies, many teams need information from these teams, and it can become a real bottleneck. One manager I knew created a wiki, and every time someone asked him a question, he added a page to the wiki with the answer and sent them a link to that page. That way he developed an FAQ for everyone." S10*

7.4 Impact of geographical separation on transparency

Larger organizations often must deal with product development teams that are spread across multiple geographical locations. Several interview subjects reported that this geographical separation generally has a negative impact on transparency:

> *"The examples of communication problems tend to be with cross-site development, which we do quite a bit of. Project teams span multiple sites, and there are very different understandings of what status is, even though we've formalized the definition. So one site has a consistent tendency to report things more conservatively than the others, and*

another site would be quite hesitant to raise any issues until very, very late. Over time, this results in mistrust between sites, and even though fundamentally it's not that one site is less accountable than the other, it's just that they have a different convention for explaining and evaluating risks. So it's quite problematic." S2

"In one team I was working with, the product manager was sitting on a different campus, so it was only possible for him to attend meetings once per week. The engineering team and the designer were sitting on a campus together, so they worked more closely. So the product manager used to come once a week, and they would brief him on the status. And through email he would address all other questions. Engineering took care of actual execution." S5

"The biggest issue is whether people are sitting near each other or are sitting in different sites. For example, when Scrum-style meetings are used, being present in person is essential. Having teams across multiple sites also results in micro-cultures developing within different teams, and they end up using different timelines, processes, and tools. You also end up with different levels of disclosure among teams. Not to mention the different acronyms and jargon." S10

"You have less room for miscommunication [in person]. If you say something and people don't quite get it, they come back again to clarify. Remotely, across time zones – you have issues with commute hours and have only short time windows to talk to people. So it really comes down to convenience for people, probably. The cost is higher. I feel bad if I make them talk to me at night if they have family. So you figure, if it's not something we definitely have to talk about, let's just do it on email independently." S14

7.5 Impact of company culture on transparency

The interviews also indicated that aspects of the company culture itself can have a great deal of impact on transparency throughout the organization. For example, if the senior engineers or executives do not react appropriately to receiving negative information, this can lead to a culture where negative results are hidden or not properly reported:

"There were a lot of intelligent people at [my company] and a lot of experienced staff. Because of that, whenever they were wrong – they weren't wrong, because they were the experienced staff. So it was interesting being younger staff, because you would raise a red flag, but you would be ignored, because you didn't have experience. So even people who did try to raise issues were often shut down because they were new." S1

"If the Technology and Innovation group doesn't get the results the executives wanted, they might not tell them the whole truth until they figure out how to fix it. I knew people in this team that had no idea what they were doing, but they would communicate things to the executives saying that everything is going fine. So I don't think they were telling the whole truth. They weren't really transparent." S4

8 Conclusion

This research consists of two phases. The first phase is a survey on 171 best practices in risk management. Analysis of over 200 responses from industry practitioners identified transparency as a key characteristic of successful risk management in product development. Due to the limited exploration of the concept of transparency in the literature, the second phase of this work consisted of a qualitative investigation of transparency through interviews with 15 industry practitioners.

The survey results provide concrete empirical evidence that there are a number of risk management principles which correlate with positive project outcomes in product development practice. This correlation is purely statistical, however, and does not reveal causation or whether the principles are independent factors. We interpret the survey statistical results as clues to potentially worthwhile concepts to further explore. In this case, given that it has been less frequently explored in the product development literature, we elected to focus on transparency.

The interview results offer a possible decomposition of transparency at several levels. First, we can see a high-level distinction between "surface transparency", which is focused on sharing of information and assumptions, and "deep transparency", which consists of mutual alignment and trust among the members of a product development team. Digging deeper into the results, we identify several *characteristics* of transparency relating to information sharing, underlying assumptions and priorities, alignment around a common goal, and trust of the competency and motives of colleagues and leaders. Finally, we analyze the interview data to identify several possible *enablers* that have been observed to increase the likelihood for the existence each of these characteristics, including frequent communication within and between teams, organizational support for cross-team and cross-discipline collaboration, and opportunities for informal communication and social interactions.

We present this work as a preliminary exploration of transparency, which could be a valuable lever for product developers and managers. The possible benefits of increased transparency seem clear. Although the beginnings of a prescriptive conclusion are present, future work is needed to further validate the generalizability, constraints, and trade-offs of transparency. Nevertheless, it is our hope this paper has provided an increased awareness to the enablers and characteristics of transparency and how they manifest in product development organizations.

References

Bassler, D., 2011. *Risk-Driven Design - Integrating Risk Management as an Intrinsic Part of Product Development*. Diploma Thesis, Technische Universitat Munchen.

Bendoly, E. & Swink, M., 2007. Moderating effects of information access on project management behavior, performance and perceptions. *Journal of Operations Management*, 25(3), pp.604–622.

Bouwen, R. et al., 1994. *Qualitative Methods in Organizational Research* 1st ed. C. Cassell & G. Symon, eds., London: Sage Publications.

Chapman, C. & Ward, S., 2004. Why risk efficiency is a key aspect of best practice projects. *International Journal of Project Management*, 22, pp.619–632.

Cooper, D.F. et al., 2005. *Project Risk Management Guidelines*, West Sussex, England: John Wiley & Sons, Ltd.

DoD, 2006. *Risk Management Guide for DoD Acquisition* 6th ed., Washington, D.C.: United States Department of Defense, Office of the Secretary of Defense.

INCOSE, 2004. *Systems engineering handbook* 3.2.2 ed. C. Haskins, ed., Seattle.

ISO, 2009. *ISO 31000:2009(E) - Risk management - Principles and guidelines*, Geneva: International Organization for Standardization.

Kutsch, E., 2010. Deliberate ignorance in project risk management. *International Journal of Project Management*, 28(3), pp.245–255.

Moorman, C., 1995. Organizational Market Information Processes: Cultural Antecedents and New Product Outcomes. *Journal of Marketing Research*, 32(3), pp.318–335.

Mu, J., Peng, G. & Maclachlan, D.L., 2009. Effect of risk management strategy on NPD performance. *Technovation*, 29, pp.170–180.

NASA, 2008. *Agency Risk Management Procedural Requirements*,

Oehmen, J. et al., 2014. Analysis of the effect of risk management practices on the performance of new product development programs. *Technovation*. Available at: http://www.sciencedirect.com/science/article/pii/S0166497213001612 [Accessed February 14, 2014].

Olechowski, Alison. 2012. Product Development Risk Management and the Role of Transparency. S.M. Thesis, Massachusetts Institute of Technology. Available at: http://dspace.mit.edu/bitstream/handle/1721.1/74936/815958097.pdf

Project Management Institute, 2008. Project Risk Management. In *A Guide to the Project Management Body of Knowledge (PMBOK GUIDE)*. Newtown Square, PA: Project Management Institute.

Voetsch, R.J., 2004. *The Current State of Project Risk Management Practices Among Risk Sensitive Project Management Professionals*, George Washington University.

Wirthlin, J.R., 2009. *Identifying Enterprise Leverage Points in Defense Acquisition Program Performance*. PhD Thesis, MIT.

Zwikael, O. & Ahn, M., 2011. The effectiveness of risk management: an analysis of project risk planning across industries and countries. *Risk Analysis*, 31(1), pp.25–37.

Acknowledgments

Funding for this work was provided by The King Fahd University of Petroleum and Minerals in Dhahran, Saudi Arabia, through the Center for Clean Water and Clean Energy at MIT and KFUPM under project R11-DMN-09.

Ryan would like to express a great deal of gratitude to Professor Warren Seering, whose advice and encouragement shaped this work and allowed it to become a useful investigation and, hopefully, a reasonable starting point for more in-depth research in this area.

Ryan would also like to thank each of the industry practitioners and friends who agreed to be interviewed for this study, despite receiving no compensation for their time and effort.

Appendix A: Interview templates

Template #1 (used for S1, S2, S3)

Question #1

What does risk management look like in your organization? What are your processes?

Follow up: At what point in the project does your team analyze potential risks? Who is involved in identifying and analyzing these risks?

Question #2

What does information sharing mean in your organization? Do you have processes around information sharing? What are the goals of information sharing? What happens / is it worth it?

Follow up: In your organization, what happens when team members discover information that is potentially damaging to the success of the project?

Follow up: How is project risk typically reported up the management chain in your organization?

Question #3

In your organization, what types of communication (or lack thereof) lead to failure in projects?

Template #2 (used for S4, S5, S6, S7, S8)

Part 1

Set up interview on whiteboard/paper by writing two headings:

- Organization
- Customer

Work with subject to identify the following subgroups within each heading and give them appropriate names based on the subject's relevant experience:

- Organization
    - Engineering team
    - Product/marketing team
    - Executives
- Customer
    - End users
    - Purchasers/requirements-givers
- Any others? Either within one of these headings or outside?

Part 2

Starting within the organization, and then moving outside it, draw bidirectional communication links between each group of stakeholders (one direction at a time). Ask subject to describe the following information about each link:

- Mechanisms for communication on this link: activities, tools, processes, formal, informal, etc.
- Strengths and weaknesses of these mechanisms.
- Rate the amount/volume of communication on a scale of 0-10, where 0 is complete absence of communication and 10 is the maximum possible communication (i.e., every single detail is communicated).
    - In your opinion, what would the ideal amount/volume of communication be for your organization, on this same 0-10 scale?
- Any best practices for this communication link that you have observed throughout your career?
- Any other thoughts on this communication link?

Template #3 (used for S9, S10)

Part 1: Shared Language

Ask subject to answer each question to the best of their understanding, based on their own experience. (Conjectured response in parentheses.)

1. Engineers in different disciplines (e.g., software vs. chemical vs. mechanical) sometimes have trouble understanding each other. Why? (Because they don't have shared technical expertise?)
2. Communication between customers and the engineering team is sometimes problematic. Why? (Because they have a different understanding of the goals of the project?)
3. Executives often don't understand information that is communicated from the engineering team. Why? (Because they only understand information expressed in their language, i.e., business results like timeline or sales figures?)
4. Managers of peer departments or teams in an organization can have trouble communicating effectively. Why? (Because they have fundamentally different priorities?)
5. Engineers often don't understand how the business side of the organization – e.g., sales and marketing – works. Why? (Because of lack of fundamental business knowledge?)
6. Managers are often unable to serve as effective intermediaries between the engineers and the executives. Why? (Because of lack of technical experience or expertise?)

Part 2: Organizational Structure

Freeform questions.

1. Describe the typical flow of information in your product development organization, from requirements all the way through development and release of a product. For each step:
    a. Is this a formal organizational link, or an informal communication?
    b. What are the pros and cons?
2. Thinking about the communication boundaries in your organization:
    a. Do they help to encourage the right kinds of communication between groups?
    b. Do they effectively reduce unnecessary noise?
3. Some organizations have a person or group of people whose primary responsibility is to serve as a communication hub between top management, engineers, and customers. This is often a group called something like “marketing” or “product management”.
    a. Does your organization have such a group?
    b. If so, are they effective?
    c. If not, how does the communication occur? In your opinion, is this more or less effective than having a dedicated person or group?

Appendix B: Interview consent form

CONSENT TO PARTICIPATE IN INTERVIEW

Study Title: Risk Management in Product Development Processes

You have been asked to participate in a research study conducted by Ryan Shaffer from Engineering Systems Division at the Massachusetts Institute of Technology (M.I.T.). The purpose of the study is to explore the role of information sharing in risk management processes within product development teams. You were selected as a possible participant in this study because you are an acquaintance of the investigator and have relevant industry experience. You should read the information below, and ask questions about anything you do not understand, before deciding whether or not to participate.

• This interview is voluntary. You have the right not to answer any question, and to stop the interview at any time or for any reason. I expect that the interview will take about 30-60 minutes.

• You will not be compensated for this interview.

• Unless you give us permission to use your name, title, and/or quote you in any publications that may result from this research, the information you tell us will be confidential.

• I would like to record this interview so that I can use it for reference while proceeding with this study. I will not record this interview without your permission. If you do grant permission for this conversation to be recorded, you have the right to revoke recording permission and/or end the interview at any time.

This project will be completed by May 31, 2015. All interview recordings will be stored in a secure electronic storage location until May 31, 2016. The recordings will then be destroyed.

I understand the procedures described above. My questions have been answered to my satisfaction, and I agree to participate in this study. I have been given a copy of this form.

(*Please check all that apply*)

[] I give permission for this interview to be recorded.

[] I give permission for the following information to be included in publications resulting from this study:

[] my name [] my title [] direct quotes from this interview

Name of Subject (print) _____

Signature of Subject _____ Date _____

Signature of Investigator _____ Date _____

Please contact Ryan Shaffer at rshaffer@mit.edu with any questions or concerns.

If you feel you have been treated unfairly, or you have questions regarding your rights as a research subject, you may contact the Chairman of the Committee on the Use of Humans as Experimental Subjects, M.I.T., Room E25-143b, 77 Massachusetts Ave, Cambridge, MA 02139, phone 1-617-253-6787.

Appendix C: Interview transcripts

This section contains the transcripts, where available, of each of the interviews performed for this study. All information that could be used to identify specific people or companies has been removed to protect the anonymity of the interview subjects.

C.1 Interview subject S1: System Engineer, Aerospace and Defense

Risk Management

A lot of it started with the stakeholder, or the sponsor in this case, not knowing exactly what he wanted. He didn't come to our organization with requirements. He said, "I've got a cool concept. Why don't you make it work?"

Any sort of requirements definition and verification with the stakeholder – that did not exist.

The biggest thing our group was focused on was probably schedule risk. By definition, what we were trying to do, everything was supposed to be faster. Money typically was not an object, and neither was labor. We usually had enough people. So the biggest thing is that [Microsoft] Project became the invaluable tool that everyone went to – critical path, saying, "Alright, is this object slipping, is it not?" – focusing on that. Other things that were used for risk management were just general check-in meetings, if you will, with whoever the lead system engineer was. But pretty much everything revolved around the schedule in some way.

There were a couple different types [of check-in meetings]. The teams spanned multiple disciplines, so there'd be engineers and chemists and radar guys, whatever. And so you'd have what I'll call "small team" check-in meetings – engineering staff only. So it's you, all the team members who are in engineering, and the engineering lead. And then you'd have the big check-ins, which are everyone involved with the team. So you'd do the small check-ins almost daily or every other day, and then you'd do the big team check-ins like every two or three weeks.

They were mainly status updates. Everyone would kind of go around and say, "Hey, is this going here?" If you had any problems or things that were worrisome, you would raise them at that time. Most people at [the company], because they tend to be highly educated, don't like to admit that things [are] going wrong in their area until the last minute. So a lot of these times, you wouldn't

get anything in terms of problems until it was pretty much unsolvable or already a ginormous problem.

In terms of usefulness – no, they weren't useful. Every time I went there it was a waste of time.

There are a lot of intelligent people at [the company] and a lot of experienced staff. Because of that, whenever they were wrong, they weren't wrong, because they were the experienced staff. So it was interesting being younger staff, because you would raise a red flag, but you would be ignored, because you didn't have experience. So that was something that I noticed was that even people who did try to raise it were often shut down because they were new.

The managers typically didn't know what was going on, which I would think is probably commonplace for most organizations, at least on the day-to-day. They would get the highlight of, "Yep, we're on time." That's all they would know. They wouldn't know that these ten issues are coming up. That's mainly just within the team.

[Updating the schedule] was more ad hoc. It was based on the team's decision to update it as things went along. For the products that I worked on, I was typically responsible for keeping on a schedule, and so I updated it pretty frequently with where we were. Anyone could look at the schedule; the people who mainly looked at the schedule were the managers – that was the biggest tool for them – that's how they knew where the thing was. They said, "Look, we're 50% complete here – great." That was probably the only channel to get red flags out to management – through the schedule and them seeing something was behind.

The problem was that we had a really weak foundation that we did a lot of our projects on. Because people were focused on time, they didn't want to take the time up front to focus on requirements development, making things matter with the stakeholders, things like that. Because of that, you could see problems later on where there was miscommunication between the sponsor and us, for example, as to what was being delivered.

Information Sharing

In terms of sharing information, there were a couple of different avenues. There was typically always a project drive or folder that was set up in which you'd have your working information available. The next piece was the status updates that I spoke about. The next one, and probably the most valuable, was through PowerPoint. You'd put together a deck saying – for example, for me, I was doing design work – here's a design that I'm doing, here are the tradeoffs, here are the problems. That would be your way of conveying what you'd done over the past several weeks. You'd probably build two or three of these decks over the course of a project. That was probably the most helpful for everybody. Obviously, those meetings were a half-day long, or a day. The engineering staff, or whoever was responsible for the area, would put it together. On smaller projects, the regular staff would be doing it, and on the higher projects, you'd have the lead staff gathering all the information and putting it together. The audience was the managers, essentially all the staff, and any sponsors that were involved with the program. Pretty much anyone who touched it in some way was involved. That would be the only time that we actually were able to get feedback to the sponsor and tell them what we were doing. How it worked there was that there was only one person who was the contact with the sponsor, and so whatever they knew about the project was what the sponsor knew, and you didn't get a two-way flow of information as to what you were doing, whether it was right or not – except at these reviews, where he would see it and ask, "Why are you doing that?", "I thought it was this", "That's a good idea", etc. So those were helpful.

One example: We were working on seal design – like a gasket seal. We were trying to close around a vent and we were having a little bit of trouble determining the material that was appropriate. So we had gone around [the company] and couldn't find anybody who had figured it out. What happened is that the sponsor actually had previous experience with different materials, and was like, "Oh, why don't you use [this]?" And so he said a material, and we hadn't thought of that, so we tried it, and it worked out. He was able to connect us, and had that happened earlier, it would have been useful. It saved us a lot of money, a lot of time by having that connection.

Raising Red Flags

I found a couple pieces of information that I tried to derail the project a little bit into a different area. I went first to my engineering lead – one level up – and said, "I found this." They either take

it or they leave it. They said, “Probably, maybe, but I’m not going to worry about it.” So I kept going, I still felt it was a problem. I jumped above him to my engineering manager – “Hey, this is probably a problem, we should take a look at it.” He said, “Ok, we'll bring it up in the next status update” – with the entire group, with like the chemists and everyone else. That did not include the sponsor – it's just [company-internal]. So we went to the [company-internal] meeting, and I brought it up. Experts from different divisions – for example, this was a chemist – I brought this up, and the chemist said, “Oh, that can’t be right. Based on my knowledge, that’s impossible to happen.” And so, obviously, I was overruled and shut down.

As we continued going through the process, it continued being a problem. In this case, it was outgassing of a particular material, and we were doing sampling of chemicals. And so this outgassing was masking the materials that we were looking for. And so we got down to the end of the project, and we couldn’t figure out where this signature was coming from – but it was coming from the valves, and we didn’t have enough time to select new valves, and figure out how to eliminate it. So we ended up putting it into the field with a giant dead area in what we were looking for. So it just happened to be that there was no way as a young engineer to gain the credibility. It was definitely not that they didn’t hear me. I was vocal enough to multiple levels of people. I believe I articulated my case well. But it was a thing where you had someone who was an expert in the field say, “That’s not going to be a problem.” Just like anywhere else, if an expert tells you it’s not going to be a problem, why would you listen to someone else? No research was done to counterprove the point – it was just a gut instinct reaction.

Communication Leading to Success or Failure

My feeling has always been that any time you can do face-to-face communication – actually talk to somebody one-on-one, so that you're not in front of other people – you have a much higher likelihood to succeed and get them to understand your point and agree with you. (When there are other people in the room, there’s an ego and there’s something to lose.) And then, you have kind of like a battle friend, and when you go into the big status meeting, you have someone who’s on your side. So that’s probably the thing I found that was most successful.

Least successful is 100% email. Email is the worst tool to try and get any point across. There are so many times where you'd write an email and someone would misunderstand the point, and then it would just get carried away from there. Email was the worst.

Anytime you had to try to sway a group was probably the second-worst. As far as I'm concerned, anytime there is a group, and someone has something to lose by being proven wrong on something or not knowing, then it's a no-win.

What I learned to do was to figure out who was going to be the voice in the room that was going to be looked to for the answer, and try to speak with them about it if I could. Sometimes that was more difficult than others. If you can prove to them first, they'll agree with you and maybe they'll raise it for you.

I tried to do this a couple of times with the engineering lead. There were a couple decisions that kind of broke the laws of physics, and so I said, "Hey, this isn't going to work." I told the engineering lead and convinced him. Then he went into the big group meeting – and this was with the sponsor in this case – so unfortunately, the sponsor didn't believe in physics – so we ended up getting overruled by the sponsor. So it was successful in me convincing someone to fight the battle for me, but not successful in terms of getting the person who needed to change their mind to change their mind. If there would have been a way for us to reasonably perform some test or some undeniable proof that it wouldn't have worked – that's probably what would have been necessary. But because the focus was on the schedule, any testing that was considered extra was thrown out. And this was considered an extra test. So those little steps and validations were gone.

That system is now sitting in [storage] collecting rust. It was used twice by us to prove that the system worked, and then no one else used the system after that. They didn't use it because of the first problem that I mentioned about the valve signature – that actually ended up being extremely disruptive. That pretty much killed it right there. What also happened was the arms were supposed to locate the vent, and because they didn't allow us to use the correct hardware for that, it ended up not being precise enough because of the laws of physics. It all kind of trickled down from there into a large failure. But we delivered a concept prototype that functioned and that met the stated

goals that were there in the requirements. So that's why I say it was a "success", but in the broad scheme of things it was a failure because no one used it.

Conclusion

The other thing is learning how to communicate – speaking the same language. You've got cross-functional teams, you've got hardware engineers talking to software engineers talking to chemists talking to managers. You're not really saying things that people understand. If I started talking about moment arms, a chemist isn't going to understand that. A software engineer starts talking about classes and scripts, that's great, wonderful. You aren't actually able to pick out if there's a problem. So that is probably something that is very difficult to address, and I don't know if it can be addressed. If there was some way of getting everybody on the same page to start – like a cheat sheet, so people all understand – maybe that helps, but I don't know. It all to me comes down to the same thing – you have to have a strong foundation and a strong beginning. Everyone has to be on the same page and know what requirements they're trying to meet.

C.2 Interview subject S2: Software Architect, Enterprise Software

Risk Management

As we move from one process to another, we have development gates where we make sure that the key questions are answered. And that's also where we disclose any risks that we've identified so far. There are set templates for explaining the risk that you have and the mitigation that you have for a given project. And for each gate, there are also prompters – set questions that most projects will identify or have to address, and you're expected to answer those.

We have various tools for reporting the status. So you're supposed to be maintaining these up-to-date snapshots of where you're at with regards to the risks you identified. You can always introduce new risks – after you've progressed past the gate you're expected to keep your status updated. We have within the project team someone called a delivery manager, one of whose primary objectives is to give everyone, especially the executive sponsors, the visibility they have on the progress and also on the risks of the project. Formally, that's how it's done. Obviously, there is great variation with regards to how much this process is respected.

They depend on the length of the project, but these gates are really to cover phases in between projects. As you shift, for example, from having an idea to staffing a team, that's one gate. Within the development process, for example – let's say the development process lasts a few years – you would actually not go through any gates for a couple of years, because you're not shifting phases, you're just in development. So you're not going through any formal checks, but in your regular status updates, you are expected to communicate your current status, which includes your risks and mitigations.

[The effectiveness of the status updates] depends a lot on the project team. To be honest, also the context of the project. For net-new products that we're bringing to market, I would say they're probably a lot less effective – the formal communication vehicles that we have are less effective. One, the standard templates we have don't tend to cover the risk – they don't suggest to the person filling it out the risks that can occur. And the other part is, with the newer products that we have, there's a lot more context required to explain the risks. And it's really hard to communicate it with these set mechanisms.

We have quite a few informal channels that are probably fairly effective. Status update meetings where we present the risk – a topic and an issue. That would give the executive sponsors a good appreciation of the problem. Obviously the challenge with that is that we're not able to do that for everything, either because of the project team's schedule, or more usually because of the executive's schedule. So I don't know if we have great ways of doing it. Our project leads are experienced enough to figure out when a risk requires executive attention – when they can actually do something about it or need to know about it. So there's a lot of filtering that happens, for sure.

In the last few years we've attempted to shift to an agile software development approach. So the core project team involves a few people. One is the chief development architect for the project. He would be involved in identifying technical risks and evaluating them. Similarly, there is also an engineering lead who has the project delivery responsibility from the developer and QA perspective, and he would identify our schedule risk with regards to execution. There's the chief product owner, who's the product management lead, and he'll identify whether a key piece of scope is in or out, trying to gauge what our mitigations could be. And then there are a number of

key people – like, more recently now, our UX is quite important, so we would have a chief user experience person who tend to be quite involved in our scoping decisions as well.

Right now we have daily top-end meetings, where the project leads meet for 30 minutes every single day to make sure that we're on track for key topics.

Miscommunications

The examples [of communication problems] I can think of tend to be with cross-site development, which we do quite a bit of. Project teams span multiple sites, and there are very different understandings of what status is, even though we've formalized their definition. So one site has a consistent tendency to report things more conservatively than the others, and another site would be quite hesitant to raise any issues until very, very late. So these patterns happen quite a bit. And part of the experience that our project leads accumulate is also just accounting for that. We've had quite a few examples where dependencies, like schedule dependencies, were not met because one team consistently reports their status as more green and the risk lower than what it probably should have been objectively.

The short-term result is just delays. But when the delays are hitting the critical path of the project, there are very significant downstream consequences. The entire scope of the project can be jeopardized. We've had to react to quite a few of these types of things. Over time, this results in mistrust between sites, and even though fundamentally it's not that one site is less accountable than the other, it's just that they have a different convention for explaining and evaluating risks. So it's quite problematic.

Information Sharing

Honestly, the way we share information has changed quite a bit depending on who's the current leader of the organization. Within the team, the level of information sharing has always been fairly high. We have standardized reporting, and we also have wiki pages where people are pretty good at keeping each other up-to-date on where they're at. But with regards to risk reporting upwards, it has varied a lot based on the executive. There are some executives we've had in the past who are very happy with their monthly, PowerPoint-based status report where there's like a traffic light

and an explanation, and they really don't want to know the details of what the risks are unless they are expected to act. More recently we've had some executives who are quite a bit more interested in making sure they understand the day-to-day operations, and they have devised a bunch of their own methods for communicating status. One of our leaders now likes to have a specific visualization format that he calls the "heat map" where the core topics that are at risk or of concern are raised through this format, so that he can always see at a quick glance what are the most important topics to pay attention to. So there's a big variety.

One more interesting way we've been communicating our status lately is we actually have started doing weekly status demos for the executives, which is actually quite a bit of work for the development team. But one thing it does do is ground the conversation on what is actually being done, so it allows the executive to have some level of understanding of what people are doing, and therefore be able to ask probing questions interactively. It allows them to be a lot more engaged and claw for information rather than having it prepared and passed on to them. I think this has been effective, but it is a lot of work, so I don't know if it's worth it. But we've definitely identified interesting issues that we probably wouldn't have identified otherwise. But the team invests a lot of effort into preparing these demos as well.

Raising Red Flags

Each engineer works within a scrum team, and that scrum team has a scrum master. He's responsible, along with the product owner, for helping resolve blocking issues and addressing concerns of the engineer. So he would find out during the daily stand-ups. And then we have a weekly scrum of scrums, which the engineering leads manage, and they find out through that mechanism. And it gets bubbled upwards that way. I think in general it works. It's pretty rare in our operating environment that there are urgent issues that get escalated without running through those channels over the normal course of time.

C.3 Interview subject S3: Chief Solutions Architect, Aerospace and Defense

Risk Management

User adoption was definitely a big risk. Very few customers would do a big-bang migration approach, so users could continue to choose which system to use.

Another measure of risk would be the day or the week after the system does go live. Is it going to meet its SLA? Is it going to be able to stay up? The amount of tickets that came in on that new system that went into production would also be a measure of success. Reducing the amount of sev-1 or sev-2 tickets that came in after go-live, while the system is actually being adopted.

Back in the physical hardware days, the old-school days, we'd have to literally go through a yearly cap-ex process to purchase required hardware and eventually it would show up at the door. So first of all, you work with the software vendor to ensure that they give you some sort of benchmarks. Most of the time these are useless, but it gives you a starting point. As an architect, I would try to simulate that usage. So before we go too far with the implementation, I would have an idea of how that company would use the software. This multi-user simulation was key. But even more important was the amount of time invested in the model. I could pass any benchmark that I designed. But having one that reflects what the actual usage is going to be is really important.

Raising Red Flags

Long story short, we had an architecture that didn't work. We had an asynchronous architecture that would take this bill, and whenever it was possible, would push it over to the other manufacturing system. They wanted this thing immediately, as if it were part of the same exact system. And there are different cultures. There's an expectation from Asian customers that the software just works out-of-the-box. It was quite a challenge to fill that gap. I'm not sure we solved the problem, but we were able to throw enough horsepower at the asynchronous system so that it got pretty close to what they needed.

For this customer, it was not really known by the software team until there was an escalation. The customer rarely gives you a requirement that's precise enough in terms of throughput and latency. They'll say, "This data format needs to be transformed in a certain way to this data format." Yeah, we can do that. But they don't say it needs to be done within 300 milliseconds at 2000 transactions per minute. So the architecture was never really designed to support that level of throughput and latency. And then of course there was the Western vendor trying to meet the demands of a different customer. Ultimately, if we had the time, a simulation would have ruled out that architecture easily. But this wasn't done on the pre-sales side at all. Very few people understood how the customer

used the system, how they used the product simulation and the ordering. No one really expected the orders to come from the manufacturing system. It took us by surprise. It required Western resources to be flown there and solve the issues. But ultimately the good news is that the product itself, the asynchronous product, became pretty scalable and pretty fast as a result of this. It's still asynchronous, but we found ways to increase the bandwidth considerably.

Understanding how engineering data is used in the entire workflow. I don't think we really understand how various facets of the organization tap into that data. Because they have these homegrown systems, they have various direct ways to get to that. But when you start standardizing that, you become the bottleneck for the whole company.

It's important to have all the stakeholders at the table. It's really hard, especially on multi-year developments. There's a different way to achieve the same solution – if you're going to replace an interface, finding a way to make that replacement really scalable. There are ways to actually prepare for that. But I don't think it's possible.

Information Sharing

One of the things that was important was the visibility of the request from end-to-end. It's always nice to give the developer actual context to know what they're solving. So instead of having a document that they never read in a system that they never access, actually put things in the context of how the information is used. So you have the initial request or the need that was defined, and that need results in some sort of either functional or non-functional requirement, and that high-level user story was broken down into specific development tasks that actually ended up in a sprint that the developer is working on. And to check that code base in actually requires that ticket. So you can do a traceability all the way up so the QA guy that finds the problem, he's doing it all in the context of that single ticket. So I became a big believer, kind of an evangelist, of getting whatever it is that we do for a customer, to be based on an actual system of record, which would be a ticket.

Now let's give users access to that and let them vote on that ticket. That was very powerful, to understand that you actually get votes on features that you're working on. And you can actually

use those votes as a way of prioritizing. So users can vote on the full list of features, and I can scrape the top ten out of it. I would do my best to follow those votes, but typically you have to apply architecture as well. Sometimes you have to pull in less popular items because it makes sense to develop them together in the same team. But exposing the tickets that developers are working on, and allowing the end users to actually vote on the tickets was very powerful. And, this is very important, tracing the developer's activity to that ticket throughout the whole lifecycle, so that we know these 200 lines of code support this task that came from this story that came from this user requirement that was highly voted. That's the ultimate in visibility. It's possible to do that. There are products to do that.

Also, it's so cool. Imagine being the developer and having a vote on your ticket that comes from a really cool customer. It gives them some, "Hey, I'm part of something bigger here." At the end of the day, it's about people, and you have to empower your developers. Otherwise, it's hard to get good staffing. So I've found that to be great – connecting the developers to the reporters – for internal projects. But it's really challenging for a contractual setup, because customers may want to get more out of that resource.

Software projects could definitely benefit from this. Our code is as good as the requirements. So the more context we can get into the systems they actually use, the less retooling you have to do at the end. Would a ticketing system work for something that is physical or continuous in nature? Does it really make sense to track everything? And maybe non-deliverable projects don't need that level of traceability. But when you're delivering code, and you want to set expectations that certain features do exist there and they do work, you need to tie those things together.

C.4 Interview subject S4: Product Manager, Civil Engineering

Organizational structure

Organization produced 50 products – equipment manufacturing for water treatment, city and industrial (not residential). We designed the products, and we sent drawings to a third-party manufacturer. They would manufacture what we sent, and they would send it to the site. And we would help with the startup. The installation was done by a third-party contractor.

We had different customers depending on the stage. We would work with engineering consulting firms for them to put our equipment in the specs. The specs would come out, and the building contractors would submit bids to the city according to those specs.

They created a new group called Technology and Innovation, and in that group they would create products across all the lines of products that we had.

We had a process director. He had the most experience in the industry for the chemistry and the process behind water treatment. He was a big advisor.

We had field services spread all over the world. They would have some of the best feedback because they were in the field. But nobody paid much attention to them, except for some product managers.

Communication Within Organization

The executives would basically tell the Technology and Innovation group which markets or which areas they should look at based on the advice of the process director. So we would pick the key products – the ones that have the most revenue – and based on their tendencies that they felt the industry was going toward. Pro is that they can have a single direction. There's no confusion as long as they have a clear message. They only have one direction. But they have no idea what's actually going on in the field without contact.

If the Technology and Innovation group doesn't get the results the executives wanted, they might not tell them the whole truth until they figure out how to fix it. I knew people in this team that had no idea what they were doing, but they would communicate things to the executives saying that everything is going fine. So I don't think they were telling the whole truth. They weren't really transparent.

Field services would communicate to product management the needs that the customer has. Based on guidance of our product, we were not forced to talk to field services on any regular basis. It was up to you – whatever you felt like doing. Typically product managers had a good sense of what was happening in field services. Most of us did.

Very little communication between the product managers and the Technology and Innovation team. They were terrible. They would just tell us what they were doing when they needed something from us. They should have asked us earlier, because we know things that we could have integrated. Because the executives have more push than what a product manager says. If I say, "Why are you doing that? We don't have a problem with that." They say, "That's what the executives are saying that we have to do." They did not talk to us. There were no weekly meetings or biweekly meetings. Monthly meetings.

Executives asked product managers, "How much did you sell?", and "How much are you going to sell?" That's it. "What's your forecast? What are your margins? Give me a strategy of what you're planning to do." Mostly one-way communication. We would have to provide everything. Executives mainly just followed up in response to what we told them. They only cared about sales from product managers. They didn't care much about technical because they had no idea about technical. If I had a technical question and I went there, they would send me to the VP of process [process director].

We [i.e., product managers] had better communication with the process director than the rest of the executives because he understood my problem. Everything related to process would go through him. If I'm using this chemical or changing the volume of my tank or looking at this application and the chemistry requires a special material – "What's your experience? What's your suggestion? I have this problem in the field services; what's your experience with that problem?" So he would be a huge advisor for us.

From field services, executives only wanted to know how effective they were – how many days they were actually in the field and servicing. Mainly just numbers. Here's a very good quote from the president of the company. We had a field service guy that worked with us for like 45 years. And he retired with the company. And at the end, the president told him, "I just want to thank you for all your service." And he was like, "It was my pleasure to provide you with my physical capability and help you with all your problems. It's too bad you never asked me for any ideas or suggestions." They only care about numbers and how much it cost if they have a huge problem in the field.

Communication Between Organization and Customers

Product managers communicated most with the consulting engineers. They were the ones who had to sell my product, so I had to give them all the information. They think that you're only selling them a product and that you don't really care about their job. So they will just tell you what they want. You have to become their friend for them to tell you everything.

Product managers had interaction with the contractors after the product was sold. After we got the order, then we talked to the contractors. It was mainly training and explaining how the product worked. But it was very little, because after the order was done, field services takes over. We gave them what they needed from us.

We talked to the city only when we were required to. We couldn't approach them directly because the consulting engineers would get really upset because we were bypassing them. The consulting engineers were hired by the city.

Field services were huge with contractors. That was their job. Field service technicians live in their homes, spread all over the US. They would have a region that they would cover. And we would tell them we need them somewhere, and they would go there. So they would stay on that site Monday through Friday, and would help the contractor with the installation of our product. So they would work side-by-side with the contractor. And the communication – sometimes it was good, sometimes it was bad – but usually good. Depending on the people and the contract [and the incentives in the contract]. If the incentives are appropriate, it will be a friendly communication.

Field services didn't like to talk to [the consulting engineers and the city]. Every time I asked one of these field service technicians to help me with a presentation, because they have much more experience, they were like, "No, no, no. I don't want to do it. You do it. I don't want to present or be in front of people."

The Technology and Innovation team would only approach the cities and the consulting engineers if they wanted to test or pilot something, when they needed a site test or a third-party innovation from these consulting engineers. This only happened on projects that reached the point of prototyping. It worked out well. That's the best way to prove the product in the market, through

those two entities. So it went well. That was the right way to do it. They have to be very communicative for them to go along with it.

Communication Between Customers

The consulting engineers would have frequent meetings to update the city. But I'm not sure how much was communicated there. The contractors interacted directly with the consulting engineers. Consulting engineers would be on site for auditing to make sure that the contractor is actually doing what they had in mind. If there are any questions on drawings or anything, they'd ask a consulting engineer. Always available.

Conclusion

I would have added more input from field services into Technology and Innovation. That communication was basically non-existent. They should have communication. Because in the field services is where you have all the problems. So I think they should have more visibility there. The executive team should filter those field problems better – either get it directly or get it through product management. It required them to be part of the field services and update more on that end.

Communication between product managers and Technology and Innovation should be higher. This would allow information from field services to flow into the Technology and Innovation team. They could come up with new designs where the product managers have no knowledge of how to do it. Because Technology and Innovation is supposed to be up-to-date on everything. So they can have a perspective that product managers and field services might not. So I think that would really help.

Between customers and product managers, it was hard. We couldn't bypass them if we got upset. I think it would help if everyone in the organization talked more with cities, contractors, and consulting engineers. Executives, mainly. The process guy would talk to customers on a regular basis. Everyone knew he had all the experience. So sometimes he would get direct calls. The rest of the executives had maybe one contact per month with customers. Improving this would have resulted in better product development to meet the actual needs of the customers, and to prioritize

the needs and push them from top-down, instead of all coming from bottom-up and getting diluted because there is no good communication.

The Technology and Innovation group really didn't get any input from customers. I really don't know how they made all those designs. Maybe the process director had some better input to Technology and Innovation. But can you imagine, it's just based on one person out of, how many, 150 people? Leadership team was like 10 people. So one out of 10 knew the technology and the industry and made the communication work. The rest were all management-oriented.

Once a year we would have a conference, and they would bring a contractor, a consulting engineer, and the city together as a panel to talk about what they thought about the company. We had a meeting where we bring people from all the teams – about 25 people from different groups. And they would have a panel for like an hour-and-a-half talking to these guys. Sometimes it was good. But you're inviting them as a guest – are they going to really say something bad? No, right? So it's partially feedback, but it can be biased because you're inviting them, paying for their airfare and hotel in Florida. How much true feedback can you get?

C.5 Interview subject S5: Product Manager, Consumer Software

Organizational Structure

CTO Dev is the office where they drive initiatives coming from the CTO. You will see product managers driving initiatives spanning across product offerings, rather than just one product. For example, they will say, "We want to create this widget which will not only work with [Product A] but will also work with [Product B]." So people inside this department will ensure that they force other product managers from different products to think on that. And they will drive that across the breadth.

Each product will have multiple product managers. Because inside [Product A], they have categorized based on the market they are addressing. For example, there are the [customers] who are new and not so sophisticated and just want to [do the basics]. Then there is another group who are more into a traditional approach and need someone to help them. And a third group might be [online users]. So each of them will have a different product manager. It will be vertically divided. Underneath each product manager they will have an engineering manager, and each engineering

manager will have an architect and a development team. That's inside [Product A]. It might be different in [Product B].

CTO Dev also has product managers. These product managers must have worked on some other product before coming to this department. And that's how they know something about [Product A], [Product B], or usability tests, for example. That's how they can think across the breadth, because in the last few years they must have worked on each of the products or touched some aspect. And this department cares about the breadth rather than just one single product.

The product managers recognize themselves as a product management community. So if there is a meeting and engineering manager comes, and he has something to push forward, the product manager will ask for him to talk to his product manager. And the two product managers will talk to each other. But sometimes CTO Dev creates some initiative which spans across products, and then engineering manager who is taking a lead there because he is technical, he wants to talk to let's say [Product C] – he might be from [Product B], and he wants to talk to [Product C], because they have something to share. And he approaches product manager and takes the meeting. And in the meeting he clearly mentions that he needs to talk with some product manager. So they emphasize the product management community to facilitate the conversation. Even though they take the meetings, they prefer that.

They do have a separate marketing group. But it depends. For example, for [Product D], there is a full-fledged marketing team, because that is more marketing-driven than technically-driven. [Product A] is more feature-driven, so for them, they do not have much sophisticated marketing need. So they have a marketing department, but it varies by product.

The usability group – user experience – is a big thing there. It is a design and user experience group, and usability is one function inside that. They ensure that they stay on top of design things and drive that culture. So they have their own vertical, in terms of design functionality. But these members can work across different products. So they will be pulled depending on their expertise. So they are part of a product team. Reporting-wise, he will have two managers: one, the product on which he is working, so he will report to that product manager; and at the same time, he will have a senior designer or head of design who he might be reporting to as well.

Information Flows

For example, one team I was working with had a meeting structure. Every week they had one meeting, and otherwise everything was going through email. Product manager was sitting on a different campus, so it was not possible for him to travel on an hourly basis – he would have to drive. Engineering team was sitting on another campus, and designer was there, so they worked more closely. So the interaction was like: the product manager used to come once a week, and they would brief what was happening, what are you doing, what are we supposed to do, etc. And through email he used to address all kinds of questions. After that, engineering drives everything basically. Engineering manager will take care of actual execution. Product manager does not interfere in that basically. At least that is what I saw.

Another structure was like, for example, they were doing usability tests. So they would bring some people, actual end users of a product like [Product D]. And they would make them use that and watch them from behind the window. So there, the product manager was there, and the engineering manager was not there. The developers were there and the design person was there. The product manager was more hands-on; he was directing the inputs.

In [Product A], it was very proper. Product manager was there, so they had weekly meetings, depending on urgency, maybe two meetings per week. Product manager used to be in that meeting. But engineering manager would drive that meeting. And the architect was there, and the development team was there. Product manager didn't speak much, only when it really mattered.

Problems

They are more concerned about their product and forget what is happening apart from that. CTO Dev would have a hard time. Even though there are product managers driving that, they still will have a hard time because they need to convince the product managers of different products to force something which CTO Dev has asked of him. And product managers are very busy (I don't know why, but I see meetings on their calendar all day). So they just roam around from one meeting to another meeting. And they really do not have much visibility into engineering. They just understand what, but they don't care about how we do it. So was there any difficulty? Yes. Because

every product manager just wanted to focus on his product because they are pressured to finish something, and they don't have time to look into something other than their product affecting them directly, rather than indirectly or some broader initiative across the company. They don't care.

If the engineering manager tries to drive that, they will say, "Yes, we'll try to," or, "No, that's not our priority. We'll talk to your product manager." So product managers tried to drive. But they will only drive if they have direction like that, and that direction will come from their boss. So that boss will push both of the product managers from different teams and say that this is a directive that we need to complete, so that both product managers know. And that way he doesn't have to do that much convincing. But many times, the initiatives are such that they are influence-based rather than just sending an email. It's not structured, but some initiatives are influence-based. They do not have any formal authority over anybody, but they just need to make it happen because that's going to bring value across the products. So in that sense, it's a very tricky situation because you do not have any formal directive from anyone, but they can still drive a change.

Communication with End Users

Usability testing works well. They will include three things. First, they will call a focus group or some other people who are using the products to infer how this feature will be received. Second, they will go for a broader study where they will send surveys or ask people to show something or videoconference or WebEx – show me how you use [a competitor's product] – to understand some kind of behavior. Third, they will go for data – they will analyze consumer data, usage data, what he is doing – that is mostly done by product managers and the data services department, which focuses more on data analytics. Not sure if information from data services ever gets back to product managers. I think there is a disconnect, so far. That is a new department, recently formed, so they are in the process of making it mature.

Some product managers go into the field and talk to people. In that case the design person is not there. So it's more direct. They will just drive up to any house and talk to people.

Overall Thoughts

[The company's] design focus is a best practice, I would say. They care a lot about design and user experience. They clearly demarcate all those roles. User researcher, interaction designer, and actual designers. The design group is pretty much a close-knit community. It's very happening, cool dude, you will see people around in tattoos and flip-flops and you forget about them anyway. They are in their own world. If you go to their cubicle you will see drawings, some kind of design artifacts, all that stuff. So they live in their own world.

The bad practice about engineering is – engineering managers are so much into technical stuff and new technology that, even though they are managers, sometimes they feel that they are still engineers. Engineering managers listen to product managers, but I believe they are not getting the essence of that talk. Engineers do what engineering managers tell them to do, right? So they are not at fault; they will do what has been asked of them – what the engineering manager interprets out of that meeting. So it takes some iterations to really get into the mind of the engineering manager. Product managers need to be very careful in this case to really ensure that he gets what he's telling him. Many times, the message is lost in that meeting because people are meeting for so many days, they have so many things to tell, that sometimes the product manager and engineering manager may not be in sync. Engineering does not care about design. Information from design goes through product managers. But in the absence of product managers, engineers don't care what design does. But since the organization is driving a design focus, that perspective is there, at least with the engineering manager. But down the line, I don't think there is anyone driving it so deliberately. In the absence of product managers, I think that design and engineering have a real disconnect.

CTO Dev has a director, who will email the product managers, but he will not keep engineering managers on the email. That is too much of a load on the engineering manager; he doesn't need to know things which are not his business. But when it comes to director, he will keep product manager and, if he knows engineering manager, he will put him too. So in that sense, yes, but strictly speaking, it will be product managers that they should be talking to.

One more factor: experience. There is one team member who had 15 years of experience, but still a designer, since that's what she liked to do. So she was an experienced person. More experienced than a product manager. So they always kept her in the emails. She was there for every email from the director. There's no one thing.

C.6 Interview subject S6: Product Development Engineer, Automotive

Overview of Organization

Usually we work with a product specifically, whereas there are different designers that may design parts for different programs. My team is the link between the designer and the program manager. We have to deal mostly with program management. They are part of product development, but in the end there are assumptions. These are driven by an interaction between marketing and product management. Product management cascades these to us, and we investigate whether that feature is feasible to be implemented, and what are the tradeoffs within the design – cost and scope and timing. We have a very hard timing constraint; we cannot go past certain dates for production.

The main customer will be the manufacturing plant at the end. We all work together to deliver something that the plant can manufacture. The manufacturing plant also reports to the executives. But we deliver to the manufacturing plant. From the product development perspective, the manufacturing plant is relatively independent.

Product development has different clusters. We have an early stage product determination called "studio". They don't deal a lot with the technical side, but they deal with attributes. Drafters, and so on. They work directly with marketing but also with product development. But the thing is that product development doesn't have that much authority over studio. They are like a very early stage of product development. They set attributes in the very beginning.

We're very dependent on our relationship with our suppliers. The suppliers are in between purchasing and product development. We deal with a lot of technical data with them, but purchasing doesn't know the detailed requirements. There are certain suppliers that you absolutely need. There are external suppliers and also internal suppliers. For example, there are manufacturing plants that provide engines or transmissions who are core internal suppliers.

Communication Between Product Development and Studio

In the early stages of the project, we deal a lot with studio to set up the attributes. This does not go well at all. The problem is that studio, in terms of getting the best attributes as possible for the vehicle, they have a very long range of time where they can set up a lot of stuff. And we end up picking suppliers late because the studio hasn't defined or frozen certain assumptions that they need to. And that actually affects a lot. They are driven by marketing and by the market, but they are not driven by engineering constraints or time. The communication is good because we have to deal a lot with them, but it's only one way, from product development to studio. That's a good communication. It is formal; you have different design review meetings and set up agreements. They have to cascade your official requirements for aesthetics – parts that are very noticeable on the exterior – and there are certain assumptions. These guys are not providing good assumptions, or late. It puts the development time constraint in danger. They don't freeze their assumption early enough, and this ends up causing problems with launching and timing constraints, which you don't want. So that communication should be more effective in the direction from studio to product development. It has to be formal, but there are a lot of informal assumptions that are not cascaded.

Communication Between Product Development and Marketing

There is huge communication up front between product development and marketing. When you are detailing the amount of content you're going to put into the next vehicle – once the core assumption for the content of the vehicles are pages and pages long – there has to be a lot of interaction. Most of that communication is also one-way, from marketing to product development. Of course, marketing wants everything in the vehicle. But once there is an assumption detailed early in the process, subsequently there is less communication with marketing because you are supposed to have your assumptions of content frozen. But later on, if there is a contradiction between requirements and content – for example, due to regulatory conditions, we have to set a certain weight for the vehicle – we would have to take out stuff. You might want panoramic roof and high-end wheels, but if they end up above the threshold for vehicle weight, you have to eliminate content. These are processes that take 3-5 years from the early stage of the product to when you actually launch them. Powertrains and that stuff take longer. But in those 5 years, the basic regulatory conditions that you were facing at the time may change. So as the product

development process continues, you do checks with marketing later on. You just touch point with all the stakeholders later on, and marketing plays a key role in this.

One thing that goes poorly is the misunderstanding of requirements or content. At the beginning of the project you start with certain assumptions about, let's say, weight, capabilities of the parts you are designing. But of course, it's a whole new process. So you might be working with historical data, but in the new architecture you start with certain assumptions that as the process matures, you really end up looking at the content, and when you do those revisions, you end up saying, hey, this wasn't as feasible as we thought, or there is a high contradiction between what marketing is requiring and what we can do. Only when there is a conflict between requirements and feasibility do we go back to marketing. So it's mostly one-way unless there are problems.

Other Communications

Communication between product development and executives is much bigger when you're about to launch. If the system metrics are not acceptable, it's a no-go. At the beginning, the executives decide which type of product to build because that's what the market expects or we are moving forward. The communication is scarce but it's very important. If you look at the holistic view, it's very valuable because they need to procure a certain quality of a vehicle. It's less frequent but it's very valuable.

Executives and marketing are talking a lot. They need to know how the market is moving.

Product development and finance interact a lot up until a certain point when you're ready to launch and you've frozen your financials. Before that it's very frequent. You want to ensure that all your design decisions are financially feasible and don't drive up cost.

Purchasing is kind of the same thing. They work with both finance and product development. They should work less with product development, but we do a lot of stuff for them. It's a very unhealthy relationship. In certain stages, this is high frequency, but when you're done, it's less.

Purchasing and finance also talk to each other. Usually, when we are setting up the supplier relationships and the key points that we're going to develop with them, we have a very strong unity

of team cell between product development, finance, and purchasing that deals with technical and financial data and communicates with the supplier. We, as product development, cannot negotiate deals with the supplier in terms of prices and so on. Purchasing are the ones that have to do it. But we provide all the support to purchasing to do that. It's a very strong communication and very formal.

Marketing also talks to finance. Suppliers work mainly with purchasing. Product development also talks to the external suppliers, on the technical side, on a day-to-day basis. You keep talking to them, in theory, before you launch – for example, quality-related. So you keep that interaction and communication with the external suppliers strong, even after launching. It happens both formal and informal.

Problems or Improvements

One problem would be the early assumptions of marketing and communicating to product development. That could be improved. A lot of uncertainty is laid out in the beginning of the product. They have a lot of different attributes that they tell you must be in the product, and you must decide what's feasible and what's not – what conflicts with other things that you're requesting. And by forcing you to accept something at the very early stage of the product development, you're left to explore your design tradespace. Marketing is supposed to bring the voice of the customer. The requirements should be flexible enough that further down the product development process, we would be able to adjust as required.

The communication between purchasing and product development can also be improved. Purchasing, in our organization, should have certain assumptions about the suppliers. And they have cost targets they have to hit in order to be profitable. But sometimes the product development organization has more bargaining power than the actual purchasing team, or do a better job bargaining or finding optimal piece prices. I think there could be a better interaction and exchange of information there. Healthy interaction between product development and purchasing could lead to a better initial price for the parts when the vehicle starts. Right now it's very hectic and purchasing doesn't know about the key driving requirements for product development – they are more focused on piece price negotiations.

The product development and studio interaction should be improved tremendously. That would help in the same way of marketing having better communication with product development, in terms of what are your key assumptions and what should you work in the beginning. It should provide some flexibility that studio needs and wants, but it shouldn't be a drastic change later on in the process.

C.7 Interview subject S7: Director of Engineering, Internet Software

[Note: This interview was not recorded, so the following are notes taken by the interviewer.]

Overall structure

The division CTO reports to CTO of the company. Under this division CTO are 5 directors:

- Director/VP of Product Management
- Director of Project Management
- Director of Engineering, R&D team
- Director of Engineering, Analytics team
- Director of Engineering, UX team

These directors form an "E&I team" with sprints, whose primary goals are:

- Identify and plan for dependencies across the various teams
- Reduce the need for inter-team communication as much as possible

Each individual team (Product Management, Project Management, and Engineering) has its own sprints as well, all of which are currently done in parallel.

Product Management works closely with VP Client/Marketing/Sales (C/M/S) to define requirements and to communicate engineering dependencies and scheduling information. VP C/M/S has the most direct communication with customers.

Project management is often involved in customer meetings as well. One of project management's main roles is to "firewall" the engineering team from VP C/M/S and from customers.

Benefits of this structure

Project management helps to understand/resolve disparities between engineering goals and customer requirements. They used to be part of VP C/M/S, but having project management be part of the CTO org in this way benefits the engineering team because their accountability is for engineering results.

In cross-company projects, project management is the main conduit between divisions, along with the other directors. Because these are usually cross-geo projects, embedding project managers across teams (i.e., long-term assignments at other sites) speeds things up considerably.

Limiting communication between the various engineering teams improves throughput by preventing unnecessary interruptions. Having engineering removed from customers tends to work well for R&D-based work.

Problems with this structure

E&I team (directors) often have commitment conflicts between their own teams' sprints and the all-up director-level sprints.

Individual teams try to be independent and communicate only at director level, but this often results in deadlocks (e.g., Analytics and R&D teams are each blocked waiting on each other).

Engineering has difficulty understanding the sales pipeline, because (a) they have limited communication with VP C/M/S, and (b) VP C/M/S often does not understand the engineering implications and what information is important to share.

Having engineering removed from customers would not work well for a more contract-oriented organization because it would greatly increase the turnaround time for customer requests and would limit direct contact with the customer.

Potential changes to this structure

Staggering the individual engineering teams' sprints would reduce deadlocks by allowing teams to be at different phases of the project at any given time. Projects tend to flow from R&D team, then to analytics team, then to UX team.

QE (quality engineering) allocation is often a problem because they are embedded in the engineering teams. A more flexible allocation might help. However, it did not work well when it was a completely separate function, since this led to engineering simply throwing things over the wall to QE.

C.8 Interview subject S8: Senior Software Developer, Enterprise Software

[Note: This interview was not recorded, so the following are notes taken by the interviewer.]

Positive Factors

- For communication between senior engineers and customers, demos and presentations at the customer site are helpful. It promotes effective two-way communication and helps to keep the line of communication open.
- The senior engineers have weekly Scrum calls with the customer, which leads to effective two-way communication. The engineering team itself is agile. They sit together and have daily standups, which leads to excellent communication within the team.
- We use Jira for ticket management. This is very effective and is visible to everyone within the software team. It allows everyone to keep track of issues and promotes excellent visibility of project status throughout the team.
- The CTO has a strong software background and trusts the engineering team and the organizational structure.
- The CTO can effectively translate customer requirements to architecture, and can also effectively communicate project status from engineers to the CEO.

Negative Factors

- Directors are supposed to be the intermediaries between the executives and the senior engineers. But they don't have enough technical expertise to be an effective filter or to transfer knowledge between engineers and executives.

- Some executives meddle and micromanage, resulting in false promises to customers, low morale, and high employee turnover.
- Some executives have poor people skills and are unable to effectively manage relationships with customers.
- Senior engineers are overloaded as people managers, product managers, and customer contacts.

C.9 Interview subject S9: Program Manager, Consumer Electronics

Engineers in different disciplines (e.g., software vs. chemical vs. mechanical) sometimes have trouble understanding each other. Why?

One thing is that the fundamental nature of the disciplines are different. Let's compare, say, software and electronics, which used to always be a clash at [my company]. For software it was always possible to make changes, and it was hard for software people to see why the electronics guys would not make changes even though it seemed like a pretty straightforward change. And vice versa as well. Electronics guys saw it as really, really unthinkable to have changing requirements, whereas for the software team, it was a lot more acceptable that things changed further down the line, and you just make the changes you need to make it work.

I think the root of it lies with the underlying fundamental stuff. Some things change, some things don't. I don't know how it works in mechanical engineering, but I assume there is also something fundamentally different about that. It is something that, at least as I remember, is a big point of conflict that we used to have on a regular basis between software and electronics teams.

It's not necessarily their background. It's got more to do with the thing that they work with – the constraints and the aspects of the pieces they are working with that are different. Like for software, the constraint is usually not locked down – you can always make changes, it's not impossible, usually, to go back and make a change to what you've done. Whereas for applications like PCB design, for example, once you make a change, the amount of testing and cost it takes to redo something is a lot. So the cost of a change is quite large. So that kind of drives certain process things. It's where the conflict comes from. Some processes that work for software don't work for hardware.

Communication between customers and the engineering team is sometimes problematic. Why?

My first guess would be that the language spoken is different. Because, a person who is a customer – let's take an example of something very simple, like, say a website or a server. The language that's used for specifying requirements in engineering – what's important for an engineer to know – is different from what the customer thinks is the key part of it. How they describe things are different, I think. Maybe that's changing over time. Many organizations actually now have better communication with customers. And maybe, again, that's a discipline-specific thing that might not apply as much to one as much as another. But in the cases where there are problems, I think it's just because the language is different.

I'll give you one specific example from [my company]. My work had a very stringent policy on open engineering contact with customers. And the reason was that people would inadvertently say stuff which would result in the customer getting something which becomes a negotiation point. For example, they'd say, yes, that was a bug, we just sat on it and made it work somehow. Meanwhile, the sales guy has pitched the same thing as a feature created just for the customer. It's minor, but I guess that's kind of the old-school hardware business.

So, language is different, and there are some things that need to be hidden, and some things that don't.

Executives often don't understand information that is communicated from the engineering team. Why?

Well, the short answer is that most executives are idiots. Actually, I don't have any intuition around that. It is something that happens and it's hard to understand why. I think it's just, again, for me – two personal reasons. One is that they don't understand details, and you'll see this kind of thing happens a lot more in older organizations where you have sort of management as a function by itself, and not so much in younger organizations where you have mostly young managers who have just moved up. A startup I worked with didn't have those kinds of issues, whereas at [my company] it was a regular thing. It could be that in big companies, managers don't have an

understanding of the product itself and focus on the management aspect. They don't understand over time what it means for an engineering team to build something up. That could be one thing.

Managers of peer departments or teams in an organization can have trouble communicating effectively. Why?

One, which ties back to the previous question, is that the people doing the communication may not understand the underlying facts or products well enough. Like, if you take something, say a router, and there is a marketing team and a client who wants to buy the router, and there are a bunch of things that the client says he wants. Unless the marketing person is somebody who has a decent understanding of routers, chances are he gets something altogether wrong. It's a complex enough thing where there are so many things that, unless you understand it well, you can get it wrong. There are a whole bunch of things. Security, WPA, WPA2, AES, all these things have specific associations. If you just remember the WPA part and forget the WPA2 part, or if you forget PSK as the specific thing that they want – each of these have specific implications which sometimes get missed if you have a person doing the front-end who doesn't understand the stuff. That could be the biggest risk in something like that.

The obvious advantage is that it saves time for the entire team. Most engineering teams hate sitting in meetings, so it saves time for all the other people if you have somebody who can understand and extract relevant stuff out. Not necessarily a manager. If you have somebody who is an expert on the topic, who acts as the communication face, it makes a lot of sense. The chance of missing out on stuff is low. And also, you project a good image of the whole team when you send somebody who is a subject matter expert, rather than something else. And it saves time for the entire team.

A second huge advantage of having one person is that – and maybe this doesn't happen as often as I think it does – but one thing that I've noticed is that when there are groups of people having a discussion, sometimes tangential discussions kick off and there are arguments, and the team says, I thought this, but I thought that. Just having one person makes it a lot more smooth to run through a list of things. So that's the advantage of having just one person as opposed to having the whole team communicate with each other or share information.

Engineers often don't understand how the business side of the organization – e.g., sales and marketing – works. Why?

I am kind of against that thinking. I don't agree with that. I think there are two reasons that this assumption exists. One is that people on the business side like to think that what they're doing is really complex and not something engineers can understand. So that's one reason. And I think the second reason is just from lack of interest. I've seen tons of people who just don't care. Take [my company] for example. There are tons of people who work in the labs who do not understand the whole razor and blade model or printer and ink model. But not so much because of the fact that they don't understand, but just because they don't care. It's irrelevant. It makes zero difference to anything on a day-to-day basis or a month-to-month basis. And the second part is that most engineers, when they transfer from one place to another, it's not so much the business part, but the functional skills. So going to another place and saying that, you know, I know how they make money – that has very little value in engineering. Whereas, when you say, I know what this process is, I know this framework for development, I know this tool – that actually has a lot of value.

So it's a combination of each. One is that it's not relevant, so they don't pay attention – and the second is that business people like to think that what they're doing is complex enough that it's not easy to understand. I don't think that's true – I worked as an engineer for 7 or 8 years. It's just one of those perception things. There are tons of people who are interested and who understand most of these things pretty well if they want to.

Managers are often unable to serve as effective intermediaries between the engineers and the executives. Why?

That's how most places work. There are always first and second level managers. I think it's definitely important. There are two aspects – the strategic aspect, how a company gets where it wants to be, market analysis, competition, etc. – and there's a second part of it, building an organization and talent, which is good enough to actually address these capabilities. So the big strength that this level should bring in is the ability to develop people. You don't need to necessarily develop middle managers that much. Who you do need to develop is your individual

contributors. Most of these people are going generate the bread and butter products for your company at some point in time. So I think it's really important.

The cons are that there aren't a lot of people who take that role as seriously as that, because a lot of people at that level just focus on moving on to the next layer, which is unfortunate. That's one of the big challenges about that layer – how to make sure that layer takes on its job of developing individual as a serious part, and not just looking upward and trying to move on to the next level.

Often that's what happens – it becomes an unproductive layer. And that's what frustrates a lot of people who will never understand why – it doesn't seem that this is purposeful. In theory, that's a problem that can be solved. I've had some good managers, I've had some great managers. But in other cases, it becomes a layer of fat which frustrates people and doesn't do anything.

Organizational Structure

Different people like access to different levels of information. I keep going back to [my company] because it's a great example of a lot of stuff. [My company] had this policy of open information, which I thought was a great thing. Everybody had access to all the information. I thought it was a good thing. Typically you have this thing where managers spend x% of their time in meetings, and individual contributors are not supposed to. So at [my company], as an engineer, I used to spend half my week in calls and conferences and discussions. I enjoyed it, because the more you knew about what was happening all around, the more informed you were when you had to make a call.

So I think there's no one answer to that. It might be an individual thing. I knew a lot of people who used to hate that because they felt it was just being called into a whole bunch of meetings without any real purpose. Some people like it, some people don't like it. A good way for an organization to deal with that is to allow people to opt out of meetings if they want to. But then it's hard to manage, because what if I opt out of a meeting, and it's a meeting where someone wants to assign me work.

Product Management

It’s like you said. The terms are called different things in different places. Sometimes it’s called product manager, sometimes it’s called program manager, in some places it’s called project manager. It’s basically enabling this communication between different groups of people. And then at the end of the day having ownership of the overall thing.

Two things: One – it’s only in the last 5-10 years that this has become a structured function by itself. In the past, at [my company] for example – it’s an example of an old company, maybe it’s very similar at [other old companies] as well – it follows the traditional engineering structure. You have marketing and you have program management and you have engineering teams. The interesting thing is that a lot of the younger companies – Google, Facebook, Amazon, Yahoo – a lot of the younger companies have this separate function that does that. And maybe it just shows the growing importance of that specific function which is having people whose job is pulling these things together. So I think it’s an important function, and just the fact that more and more younger and newer companies seem to be having a dedicated role for it shows that it’s an important thing. The main job is communication in that role.

To be effective, I think you should be interested in what you’re doing. I think that at some level, you need to do the whole horizontal inspiring of everyone that you communicate with. And at a basic level, that gets driven by how excited you are by the topic. If you have somebody who hates lollipops talking about lollipops, it affects it in a tangible way. Same works the other way around – if you’re excited about what you’re talking about, a bit of that rubs off on everybody that you speak to. So I think that one key thing is that it should be something that you have some level of an interest in. How that's driven is different – it could be that you’re interested in something else.

It’s a big difference between that and a project manager. A project manager’s outcome is fixed – it’s just about tracking and things. But for this, more than the tracking part, you need to have an interest in the product so that you are able to think slightly beyond just in terms of schedule and things like that.

I definitely think this concept applies outside of software as well. I think it’s applicable in any fast-changing environment. That’s maybe why the older companies didn't have it, because the rate of change of technology wasn’t as fast as it is now. So if things don’t change as fast, you don’t need

that level of product management. But when things do change, then you need to keep your fingers on what needs to change. Because if things change on a year-to-year basis, it's more important than if things change on a 5-to-10-year basis, like cars for example. The more volatile the industry or the product feature set – the further away from dominant design – the more value product management bring.

C.10 Interview subject S10: Software Engineer, Internet Software

Engineers in different disciplines (e.g., software vs. chemical vs. mechanical) sometimes have trouble understanding each other. Why?

The biggest issue is whether people are sitting near each other or are sitting in different sites. For example, when scrum-style meetings are used, being present in person is essential. Having teams across multiple sites also results in micro-cultures developing within different teams, and they end up using different timelines, processes, tools, etc. You also end up with different levels of disclosure among teams. Not to mention the different acronyms, etc.

Communication between customers and the engineering team is sometimes problematic. Why?

There are always problems setting requirements, agreeing on a timetable, etc. And then customers always change their mind about what they want. The engineering team must set expectations appropriately. Customers don't understand the details or cost of what they are asking for. What seems like a small change to them might actually be a huge change.

Executives often don't understand information that is communicated from the engineering team. Why?

The engineering team tends to package information so that it is digestible to the directors. Because of this, it is often oversimplified, especially since executives don't understand the nuances of everything that is said. This problem is compounded when the executives then relay the information on to someone else – information can be completely lost in this process.

Another thing is that showing unfinished prototypes or demo code can be problematic and lead to miscommunication. Directors see this and think it is the real product. This can lead to deadline pressures being applied to the engineering team. Also, when the engineering team faces deadline pressures, they tend to report inaccurate status in terms of percent done. They will report that the project is farther along than it actually is.

To improve status reporting, first, the reporting must be frequent. The company should encourage direct relationships between individual engineers and directors, so that informal communication can occur. This is better than formal processes. The key is not process, but people – the relationships are more important. Team-building activities and the like can also improve this type of communication.

Managers of peer departments or teams in an organization can have trouble communicating effectively. Why?

Having a main contact point on a team is critical. This is usually the manager at first, though the actual communication usually ends up getting delegated to an individual engineer. In my experience, teams with lots of cross-team organizational dependencies need to be very structured in terms of how they store and share their information. Many teams need information from these teams, and it can become a real bottleneck. One manager I knew created a wiki, and every time someone asked him a question, he added a page to the wiki with the answer and sent them a link to that page. That way he developed an FAQ for everyone.

Engineers often don't understand how the business side of the organization – e.g., sales and marketing – works. Why?

At [my company], we had very frequent meetings between the marketing manager and the engineering team. These meetings were optional, but lots of information was communicated. One practice at [my company] which turned out to be effective was having "ambassadors" who work closely with both sides of a communication link, for example, design and engineering, or marketing and engineering. They are sort of like specialized product managers whose main goal is to ensure that communication is happening smoothly.

Product managers at Amazon sit separately from the engineering team, which decreases contact and discourages development of close relationships. They have "technical program managers" who sit with the engineering team, but these are really just project managers who coordinate multiple engineering teams, and don't actually work with the product management team.

Thoughts on product management

The best product managers are passionate about technology and have an understanding of what users want. They develop a strong relationship with all disciplines (test, development, design). They use the product themselves and are able to learn from it and extract useful information. PM should serve as a conduit for good ideas to flow from all employees in the organization no matter where they originate.

C.11 Interview subject S11: Project Manager, Transportation

[Note: This interview was not recorded, so the following are notes taken by the interviewer.]

Methods of status reporting

Sometimes I asked for a written report from my team. Sometimes you don't always get the true story, since what they write down might not be the full truth. They shape the answer or their status based on what you expect.

If you have a face-to-face meeting or informal discussion, you get both the verbal and non-verbal cues. It is more difficult to hide it in face-to-face as opposed to status reports.

The informal style is less structured, and it tends to lead to more open dialogue and more outside-of-the-box discussions. Reporting is usually shaped in the form of a template and you tend to get more structured answers.

The formation of the team also makes a difference. You might not have an opportunity to meet on a regular basis, and so you are forced to do a written report. The way you have to present it could be easier in a structured form, delineated between the groups and the activities to get a clear level of status.

If a team is co-located and is always communicating, you don't necessarily have to go to the more formal written report. You can have the status instantaneously. Maybe it's shaped around how teams are structured, their location and their roles, and whether they are dedicated to the project or across multiple projects – whether you can bring them all together at one time. If you don't have everyone present, or if there is important information and people can't be part of it, then the more structured report is good because others can see and follow up on it.

To report status higher-up, it's nice to have something written down. For my company, we tend to go more formal as we go higher-up. For senior management, we would have a very formal written report – short, concise, and consolidated for distribution across all the groups. Spoken status or over the phone is more than enough for our small team.

The guys higher-up are less involved in the day-to-day projects and don't have a good sense of what's really happening. They need to be able to quickly go back and check the older written reports. If it's just informal, and you're trying to handle like 20 projects, you may not be able to recall all the details of a brief you got two months ago. At least for us, based on the structure and how the reporting flows, since you're not involved, you don't have the first-hand day-to-day knowledge of the project, so informal reports are hard.

Teams can also use that – you can be stuck to how you've communicated in the past – and informal update can be better because it's essentially off-the-record. It's not written in stone in a document archived away. People tend to be more honest in informal settings.

Structure of status reports

Start with background information, summarize the subject, and keep details to a minimum so that it is easily understood and not too technical. Highlight the key points. At times we will use graphs or graphics, tables. Less commonly some pictures. Just based on speed, a lot of it tends to be text. It's mostly text-based.

If it's a case where it will take way too many words to explain the situation, say a concern of a labor load or something that could be more easily understood with a graphic than text, I'd tend to

go with the graphic. It depends on the type of situation. Use a graphic when it helps with understanding and it's an important enough issue.

It's always tough if you don't know what they actually expect. You might struggle with what they expect or if it's being understood properly on the other side. It helps if you know who you're communicating with, based on feedback from a previous report, I'd structure my report in a way they like it, if they like lots of numbers vs. more highlights or financial figures. It really depends on the perspective and how they like to grasp the information that's being presented. Err on the side of providing more information rather than less.

It's easiest to have relationships with people with a similar background, so if you're in engineering, people who grew from an engineering background. Marketing and so forth might not speak the same type of language as an engineer. Someone who knows a lot about the subject can call your bluff if you're not giving all the information. But it tends to be easier because they speak the same language.

At our company, product managers define the overall product strategy. They will communicate with the different engineering groups (hardware, software) and maybe marketing. They don't tend to be as open and as in dialogue as you would expect. They tend to operate in isolation, which maybe doesn't make sense, but it's the way it's done. We are structured in project teams. PM should facilitate communication between product and project, but it doesn't always happen that way.

Tips/Tricks

It's useful to go away from normal environment and go an outing or something, maybe a more social conversation. People may be more honest with you if they're not at work and feel more relaxed, and they are more likely to speak up if they have a problem or an issue. They may not normally do that. It may be the way the information is handled – how you react at that time. If you give them negative feedback, they're less likely to present you the negative information. They may conceal it because they fear your reaction to that information. If you take their side and become more friends with them in a friendly setting, they'll be more comfortable with you and more likely

to share even if it's not positive because there is a sense of trust and relationship that you've developed.

C.12 Interview subject S12: Program Manager, Commercial Electronics

My most recent role that was more of a product role. I was working on data acquisition hardware. I was a product manager. It's one of these roles that's defined differently in different companies. There, I was technically in product marketing. Marketing communication was a completely separate organization. Product marketing is all engineers. Marketing communication was marketing/PR specialists. So within product management, my role was to work really closely with R&D project managers and leadership within both product marketing and R&D. My role was to bring the customer insight and competitive analysis to define future products.

On a specific project that we were developing, there was a project manager that set up the weekly meetings and split up the work among all the different types of engineers. In the hardware world there are a lot more different types of engineers – analog engineers, digital engineers, software engineers, mechanical engineers, so a lot more different disciplines. So I was a member of that team. Organizationally, we were on different teams, but within a project, we were a project team.

We typically had weekly meetings on the project – I honestly didn't go every week, it wasn't always the best use of time – the project manager would go around to every person and ask how we're doing on analog design, are the test engineers building out the tests. It was pretty much a status update. Someone might bring up a dependency on another team, or something might take a lot more resources than we thought, so we might have to revisit requirements. That's where a lot of those discussions would happen. But any major issues that would change the definition of the project would have to be rolled up and taken outside of that room, because the decision makers weren't in the room. Most of the decision making took place among the product marketing manager, the R&D manager, and some of the tech leads.

Sometimes the tech leads were in the weekly meetings. They were engineers that are really experienced. They would be involved in a lot of projects at a higher level, and they would be more involved with all of the architectures and technologies involved. They help things to go a lot more efficiently. In an ideal world, everyone on the team is operating at the same level that tech leads

are able to. But you have to train people and bring people in, so it's not always possible. Those are people that are going up more like on a technical IC path and are pretty far along.

More recently, we did start for some of the projects – depending on the project manager – to do more of an Agile-type method where we did scrums every day. Those were supposed to be quick standup meetings. They always seemed to take longer than they supposed to. They were supposed to be 15-minute meetings, but everyone was in a chair, and it always took longer. The idea was, from the project manager's standpoint, it was a good way to stay on top of things and bring things up as needed. But they didn't seem to be as efficient as they were supposed to be. I think it's really hard for people to actually give quick updates on things.

The standard was to have weekly meetings, originally. People started trying to adopt the more Agile approach with the main goal of being able to have if somebody's blocked, for that to come up sooner and be able to move faster. For people not to wait a week to talk about something they needed to talk about. Some people would do that anyway, but sometimes people would be like, well, the meeting is on Monday, it's Thursday, I'll just wait. It can be almost like an over-communication to have it every day, but at the same time – one of the things that is tough about those is that – for example, I don't know what half the people are working on. Is this the best use of my time? It doesn't actually intersect with anything I'm working on. It's good for the leads, they get to hear what everyone is working on. So that's one of the things that is tricky. You're in there for however long and you only get a couple of nuggets. So how do you filter that more efficiently?

The majority of the team was on the same floor, but we did have more open workspaces/cubes, so there was naturally more collaboration. In general, everyone was on the big floor. Not everyone sat right next to each other.

I think that from a communication or relationship standpoint, I think it's important that people trust the other people on the team, and more importantly, the project manager, if they're going to take the time to be more open and talk through things. Trust means competency and being able to do something about it. Trust has to be built over time. You work with someone and you see that when you bring up an issue or are blocked on something, or if one engineering discipline is causing

the holdup, then they can go and get that smoothed out. If they see that it's effective, if it helps them to bring things up, then they'll be more likely to do that.

People want me to succeed, but I have to do things for them to see that I'm adding value. People don't assume that they can just give me some big thing and let me run with it. It doesn't matter what's on your resume, people have to see what you do.

Getting to know people personally and also just being open yourself. Trying to get to know, focusing the most on other PMs and those I work with in my immediate role. Those are the ones I try to proactively have lunches with people, get people to leave their offices and have coffee with me, just trying to learn from them, being willing to have them teach me.

Senior management / executives

Within R&D and within product marketing, there were separate reporting structures. I rolled up to the director of product marketing, and the other side rolled up to the director of R&D. The directors were very aware of every project that was going on. Ultimately, they were responsible for our business unit. They were ultimately the decision makers and the approvers, so once a strategy was set in place for the year, they decided everything. It didn't typically have to go higher than that. Directors had accountability. These two directors, if there was a major issue or conflict between marketing and R&D, sometimes there is a tradeoff, and sometimes R&D says it's nine months late, and that's not ok and we'll need more resources. We'd escalate that to the director. Our directors worked really well together. Things would get escalated if they needed to, and the two of them had a lot of meetings behind closed doors. They were always able to come up with a plan together and roll it back down. They're looking at all the projects and are ultimately accountable for the business.

Both of the directors were engineers at one point, they've been at the company for a long time, so even on the R&D side, she understands the architecture and overall technologies. In general, they call them tech leads, because people can be different levels and be tech leads on a project. So if it was like a product manager and a tech lead, the directors really helped with that. They also have a shared goal, which is another thing that changes people's behavior.

Relationships

I sat a few cubes away from the director of product marketing. There was more informal communication in the elevator about issues, or something that is going well. That informal stuff keeps them in the loop. I almost never emailed the director, he gets so many emails, has an admin filtering his emails – I'll just talk to him. Those informal updates, even just really quick, help to avoid major surprises.

When you have a base or a personal relationship, not even hanging out outside of work, but just small talk or coffee, it's a lot easier to go to talk to them about a project if you're friendly and nice. Whereas, if it's someone you don't talk to very often, and every time you talk to them you're asking them to do something, it's harder.

We used visuals for schedules. Software would use bug charts. Nothing really creative or out of the ordinary.

There were lots of social events where people are doing happy hours to celebrate personal achievements or project launches, which brings people together a little bit. Personally, I've always tried to get to know certain people that I work really closely with. I'd much rather people feel comfortable enough to come directly to me and talk about something, or if I need something from them I can go straight to them, instead of having to have something happen some other way. Especially when you have people who are very opinionated or knowledgeable or who can block projects, it's really good to get to know them and know what their concerns are before it gets blocked. I have lunch with people to understand what are people's concerns, what are they worried about, what's hard, what's on their plate, what are their goals, what's keeping them up at night. When you understand what someone's concerns are, it's a lot easier to work with them. One of the things I've learned is that when working with anyone who knows things you don't know, especially the tech leads, because engineers are a breed of their own – the people who have been designing something or working on a certain technology for a really long time – they don't need a product manager or a program manager to come to them and tell them how it is. It works a lot better when you have someone who is really technical, to try to learn from them instead of trying to go the

other way. They appreciate that, they see themselves as the expert and they know you see them as the expert, and they love to be seen as the expert, and then everybody's happy.

I do think there's a lot to the relationship piece. It's tricky. You have to build the credibility. It's not the kind of job you can do on your own. You have partners and other teams to work with, especially being a PM or being remote.

C.13 Interview subject S13: Program Manager, Automotive

The whole office [at my location] is a product development office. It's divided into projects, and from there, I'm in the program management team. I have a supervisor and two program managers above him, and then there's a Chief Engineer. Below the managers there are three supervisors, each one with one or two analysts like me. So a team of six or seven people. For bigger projects, like [Product A] or [Product B], there are two or three or four program managers with two or three supervisors each.

We have more meetings than we need. We have the meetings within the program management team, which are weekly meetings where we catch up on highlights from the week, hot items or milestones coming up. That's within the team. And then you have the engineering community, like real engineering doing product development, suppliers, and the design team, which is a different entity. Each one of the supervisors on the program team works in a stage of the program. So my boss and I are working with the design studio on the design phase. And then at some point, one of the other supervisors will take some part of the job and do another phase. And the last one, which is the launch supervisor, will take the project at that point and keep going.

We have one informal meeting and then a formal program team meeting, which is communication to the rest of the community, like engineers, design, purchasing. Both are weekly. The informal one is just supervisors and analysts. It's just grab coffee and talk about what are you working on, do you need help from any of the other supervisors, and what is the next big milestone for you. It's more like a roundtable, there's no specific agenda, just an email saying these are the things I'm gonna talk about. The supervisor is doing the job, and the analyst is just supporting what the supervisor is doing.

The other meeting, which is the communication for the project, is specifically for the project. It's a more formal meeting, two or two-and-a-half hours normally. In that meeting you have the engineering community, you have purchasing, you may have suppliers invited, you have design, finance, marketing – all the entities around the program team. Weekly, two-and-a-half hours. It's too much. So the program team and I stay there for the whole meeting, even though we don't have anything to do with purchasing or any other part. You can leave, but as it is the program meeting, you want to stay there to catch any issue. The way it works is the program managers try to drive the meeting with the chief, when the chief shows, which is not weekly. They try to drive the meeting, but in my case, everything is urgent at [my company], always. So if you come to the meeting, it's more like you are explaining why you were late for something, or that you're not doing. So people don't really show up to the meeting as they should be. You have to show up to the meeting to get the general status of the project. But that idea has been transformed into, "Come to the meeting because you owe me something or you're late to something, so I need you to explain why you are late." Because the program team tracks the whole project. So instead of the program team telling you when the next milestone is, you're always late, so you come to the meeting to explain to me why you are late. So people started not to show up to the meetings. Maybe they don't know, maybe it's a supplier thing, maybe it's something external and they don't have an explanation. And the program team is just asking, asking.

Several reasons for this. The main reason is the design phase was supposed to be like a year and a half. There are some specific milestones. But when we get to the point where the project went out for market research, to confirm all the work and the design up to that point, we failed market research. People said the [product] is awful, we don't like it, it's not luxury enough, a series of things. So the whole project switched back to the concept phase and the design started again. From the design perspective, that's kind of an easy thing to do – they just get rid of the sketches and start all over again. For engineering team and suppliers, it's a big deal. Maybe you have started some development on key commodities. The engineering team will have to redo all the work. You give me a design, I tell you if it's feasible or not – that's like a back-and-forth – but when you've done that two or three times already, it gets a bit frustrating.

And the thing is that, the design team normally doesn't incorporate the feedback that they get from the engineering community. They are like artists and they don't care. So if you as an engineer provide feedback, and in the next iteration of the design you notice that your feedback was not there, you just stop working. You say, I'm fine, I don't care anymore. But you need to do it. So that's when things start getting delayed, and the program team starts to ask you what you are doing. And it gets messed up. So engineers stop pushing back, and they just let things move until a point where they have to make a decision or they can't move anymore, and they have to redo the designs again.

The biggest communication issue in the product development team is between the engineers and the designers. The design studio is like a unique community of people that is isolated from the rest. You can't get access to the studio, basically – you need authorization. The designers need all the rest of the people to provide feedback, whether it's marketing or engineering or suppliers. But they don't ask for help. They're expecting people to be aware of what they are doing and to provide feedback. But then again, you can't get in. There are some engineers working on an exterior component – those are the guys that can actually see the sketch design and concepts. But there are people like in the electrical team that have no idea what the [product] looks like. Maybe they don't have to, but at some point you need to get a perspective on where are you putting that module you are releasing. So that lack of communication between design and engineering is still causing a lot of trouble. That's my perspective.

Part of it is confidentiality. But the way we are organized makes it difficult. You may have an engineer working on two or three projects at the same time instead of having a dedicated engineer with access to the studio. When they notice that an engineer is working with multiple studios, they try to keep their studio locked. I think it's because they don't want the design to leak out. You can get access, but you always need an escort to take you inside the studio. It's me asking you if I can do my work on your design. People are very different from being an engineer to a designer. The mindset is very different. They don't realize that they need each other. It's ridiculous. And then here's the program team as the third guy at this party. We talk to the studio, and we ask why this is not done. They say, I don't have any engineering feedback. So I go talk to the engineer and ask what's going on. They say, I don't have access to the studio, I haven't seen the surface, I haven't

seen the design. So I schedule a meeting for tomorrow, we walk to the studio, we talk to the designer, we get in, we see the stuff and you provide feedback in a couple of days. So it's us escorting the engineers all the way to the studio to have a meeting inside. That has been the common practice all over. At some point, I stop being the analyst and become just another boss. I'm fine doing that, but it's not the right way to do it.

Design is supposed to attend the big weekly meetings, but they never show up. They don't care about where the project is going. They have their own idea of the project milestones. They know they have to deliver something by that day, but they just don't care about anything else. That's the reality. The studio used to be different, the mindset. But when the [company reorganization] happened, I think they empowered the studio to become the most important attribute. So we're gonna win, we're gonna sell more [products] because they are pretty, because they look nice. And then the rest of the performance are just other attributes. So when you empower an organization like that, but you don't tell them that they need help from the rest of the team, they just feel like they are doing their job and all the rest of the people are just supporting them. That's the message. You can tell – if we end up in a revision with chiefs or maybe vice presidents, and it's a decision between an engineering attribute (like weight or fuel economy) versus a design element, the design element will always win over the engineering attribute. Even if that means you have to reduce your fuel economy number, maybe a label – even in those cases, the design element will win. So the message is very clear – I'm empowering the design team, I'm supporting them, and you guys need to accommodate them. So that's where the engineers are like, ok, do your job, I don't care, I'm not engineering the [product] anymore. So it's difficult. That's what I've seen.

One thing that can help – as the program management team for the studio, we set up weekly meetings specifically so that if there is an issue between an engineering attribute, a design attribute, and maybe marketing or something else, we have those meetings for that kind of discussion. That's the meeting intended for design to show up – it's their meeting, that's the message – and they show up. At some point they were showing up, they agreed to talk about the issues – but then again, when it came to a point where it was a tradeoff between a line on the design or aerodynamic performance, they kind of stopped, hands off, I'm not doing that, we need approval from the design vice president. It's not even a tradeoff, we have to do it. So it all depends on how much weight

you put on the attributes. How you measure the weight of a design element is very subjective, and it's always one person making the call of whether it looks nice or not. And going back to what I said about market research, after all that work and you empowering the design team, to go to market research and just fail – I think everyone was expecting to win because we have been trading off all these things for that design, and people don't like it? It doesn't make sense. So the engineering team and design can blame each other for the failure.

The weekly meetings are accomplishing the purpose of bringing up the issues, but we're not making any decisions because we always need approval from someone else in the design community to make a change. Even if the engineering team came to the meeting with all the data saying why we can or cannot do it, there's no one from the design team who can make a decision. It's a very centralized structure in design – it's the VP of design. Sometimes engineers don't even know who the VP of design is. And you have so many layers to get an issue escalated, that it might take from September to November to get an issue to the top. And then you have to start all over again. In the engineering community, it's more parallel – you have an issue, you need help – but for a design change it always goes to the top. So you realize at some point that your meeting at the first level doesn't matter. You can bring up the issues, you can talk about them, but you're not empowered to make the decisions. Engineers can get their chief to the meeting, but you will never get a chief designer in the meeting, because they know that they are being attacked – that's how they feel – and they don't feel that they need to listen to anyone – and they are outside their studio and they feel vulnerable.

One good example is the engineering offices of [a competitor], when they were building the new facility, they designed the building according to the best distribution of people. And they came up with the idea to put the design studio in the center of the building. And every floor has every team of that project on one floor. And the design of the property will be happening in the center, and all the engineering groups will be sitting around, so that you have some interaction. Then between floors it will be locked, you can't go to another project, so that confidentiality was kept. But then communication could be just walking to someone's desk. I think that's a good example. It really improved the communication within the team a lot. Maybe it would help [our company] if it were organized in that way.

The main tool in my job as an engineer is Outlook. I hate that my main tool is Outlook. I do everything through Outlook. People won't take direction or a meeting outcome if it's not in an email. And we will use emails to let people know what happened. Which to me is an issue. If you are doing an engineering job, and you have some other deliverables or ways to draft your project, add so many emails from so many people. If you can improve the communication by just going to someone's desk, I trust that you have all the data, and you trust me as a designer, trust the program team that we need to do this now, let's just get together and do it. You don't need to send a ton of emails to do that. That's a very good example of lack of communication if you are doing everything through Outlook.

C.14 Interview subject S14: Principal Software Engineer, Enterprise Software

Our teams at [my company] typically followed the waterfall model, where developers write the code and throw it over the wall to QA.

Teams are all sitting together in an open environment, so collaboration is really easy. One very important aspect to transparency is the convenience of communication. When it is easy for people to talk and collaborate, they will do it much more often. When we work with teams that are geographically remote, for example, India, it's much more likely to be formal or over email. It takes much longer and is much less effective.

It's much easier to be transparent with people who are easy to work with, and I don't really like working with people who are hard to work with. Being easy to work with means that you, first of all, know about the technical side of things, so you can understand. And second, it means that you don't waste my time with unnecessary information, and I don't have to explain things to you multiple times. You understand what I'm saying and you know what action needs to be taken as a result. It's really about efficiency and knowing that the person will do their job.

I'd choose working with someone who is easy to work with over someone who is local. All things being equal, of course, I'd also choose the local person, but it's much easier to get things done with a remote person who is easy to communicate with, than it is with a local person who is hard to work with or doesn't communicate well.

This type of communication is critical to the success of projects. Companies don't always recognize its importance or reward it properly, since it's more about collaborating and helping others rather than directly producing results yourself. It's very difficult to measure, and if a company is looking at the long list of things they need to improve, this is not likely to be at the top of the list. But I think people need to be properly motivated to work in a collaborative way. The way knowledge is shared within a team is very important.

One of the most important things is that people need to be aligned around a common goal. If I am in the room with a customer, a sales engineer, and an engineer from a partner company, we need to have a common goal of solving the problem and making the customer happy. Too often our goals are misaligned, and we are each trying to blame the other company rather than making our mutual customer happy. But if our goals are aligned, it motivates us to have much more effective collaboration and solve the problem in the best way for the customer.

I am a big believer in using whiteboards as a means of communication. It helps me to organize my thoughts and think through things as I am presenting them. And if I make a mistake, it's easy to correct. It's much more efficient than creating a PowerPoint slide. I use whiteboards no matter who I am talking to – other engineers, sales, and management.

I think that it comes down to communication. You can communicate easier when you are sitting together. You have more tools available to use. Even on WebEx you can draw on a whiteboard, but unless you do that day in and day out, you're not familiar with it. If you only do this once every couple months, you don't have the skills to make it work. But a whiteboard, doing it in front of people, is easier. So it's less effort. With WebEx, it's hard, people don't really know how to use it, so you end up just writing emails.

So I think that's definitely one reason. Second, you have less room for miscommunication. If you say something and people don't quite get it, they come back again to clarify. Remotely, across time zones – you have issues with commute hours and have only short time windows to talk to people. So it really comes down to convenience for people, probably. The cost is higher. I feel bad if I make them talk to me at night if they have family. So you figure, if it's not something we

definitely have to talk about, let's just like do it on email independently. So that's definitely a tradeoff.

In my opinion, if I could choose – the person that is easy to work with, I would value that more than whether the person is local or not. But if that's equal, then definitely local. It's much easier to solve problems. Because a lot of times you actually find out, if you have someone who is easy to work with remotely, you actually get things done a lot easier. And if you have someone who is very difficult to work with, even sitting right next to you, it's very difficult to communicate and get stuff done.

As an engineer, "easy to work with" means not arrogant, open-minded, willing to discuss things. If you have two components, and there's an interface, and you really want people to do it your way – that's not discussing. That's one thing. Also, when there's a problem, if you just start out blaming people and don't think hard about what you can do better on your side, that's not easy to work with. You basically pre-conclude discussions before people discuss. You can tell when people are difficult to work with. Arrogance is one. Maybe they have certain things that they really want, which means they are not open-minded. Secondly is your ability to explain things.

The type of person I want to work with is one who is easy to work with. You go to them with a question, they know what you are asking for, and you don't have to explain everything, and they give you the answer and really help you out. The ability to communicate well on the technical side, and also the awareness to communicate.

There are two things I value. One is you have to have the awareness to communicate. You will use feedback to learn or find out how to really explain something. If you look at the other guy and they just don't get it, you figure that something is wrong in how you explain this thing. You're explaining it, maybe it's too technical for them that they don't necessarily need to know. So I think awareness is really important. If you have awareness, you can improve your skills over time. It is a balance.

You have to be aware of time management. As an architect, I have people that will come to me twice or three times to ask the same thing. And different people will come ask me the same

question. So that's when I'll push back. Maybe you should be taking notes on this information and then spread it out on your side, because that's better. Also, it's better for the organization, rather than having five people asking the same question. When we communicate, we should really think about what is the value of this conversation.

To be a good communicator, first you have to know the domain. Hopefully that's already a given if people are asking you questions. And then you have to know the audience. Are you talking to a junior engineer or are you talking to a senior engineer? And once you get to know someone well, you know that he doesn't like a particular style, so you avoid it.

Projects are absolutely more successful if they have these people who are good at communication. That's one area where I don't think organizations give enough credit. For example, I work with a lot of people, QA and other engineers. A lot of people say they get a lot of help from me. Great, I get my nice feedback. But what else? Every time I explain to somebody, that's helping – but that's taking away from me doing my own work, because I already know the stuff. That's one reason I left – why do I keep sharing the stuff? I don't mind sharing the stuff, but eight months later I'm still explaining the same thing over and over again.

My view of Agile is to maximize the output of the people you have. You don't always have a choice of the people you pick. But it's important that people are open-minded, and you want to push them to simplify their explanations. People will do better, even if they have a hard time explaining, you can send them to training or help them practice. They will learn. Sometimes organizations might just value that engineers get their job done and not mess with them, and leave them on their own. I'm not a big fan of that. You're ignoring some of the risks that happen.

C.15 Interview subject S15: Product Manager, Aerospace and Defense

We were all on one team, all working for the same company. At the largest point we were twelve developers, nine of which were full-time, plus myself and our project manager. At times he was heavily involved when we had a meeting with the highest stakeholder, and at times not involved because he was chasing down a bunch of other projects. Over the lifecycle, at some points we were maybe four or five engineers plus myself. Sometimes we would pull in a graphic designer to help us out. We also had consultants who would help us out, sometimes from academia who were doing

more foundational research. We had a company in Canada who had a software tool that we integrated into our system. Then we had some subject matter experts from a consulting company who would make sure that we were doing useful things for the user and connecting us to potential observation opportunities like classes or sources of knowledge. That was [our company's] team.

Then the whole system: [Company A] was in charge of the OS, and another company was in charge of sensors and sensor processing, and another team was in charge of data processing. They were giant teams of 50 or so.

With our consultants, we would check in with them monthly. Every three or four months, they would come in person and we would brainstorm or something. Our subject matter experts, I communicated with them a lot, since I was on the product manager side and they were my main source of user feedback. So I was emailing and calling them relatively consistently. At some point in the project we switched to more of an Agile or scrum model, so we were having a standup meeting every afternoon, we had planning meetings every two weeks. We tried to do it pretty by-the-book. As far as the project, the technical side would talk on the phone every week, weekly calls. Same thing with the user side. I was on both of those calls. Every quarter we would have a big review, and then we would have a field test. Leading up to those were a bunch of integration events, so our team was constantly flying to an integration lab where we would meet everyone in person and put the pieces together.

The scrum meetings were useful, especially when we were in a team of 7-to-12, it was really helpful to know who was working on what and to make sure that we got a sense of what we were going to have at any given time, since we had to communicate that to the larger program to manage expectations and integration and other things. So keeping track of all those pieces, it was helpful when we got together to sync up on it. I think the developers also found it helpful because they could see when there was overlap, or who had a problem that someone else could solve. We really kept them to 15 minutes. We stood up, no chairs, it was really 15 minutes.

I think it was helpful for us. We only started doing it toward the end. Before, we sort of had a stand up meeting, but people wouldn't come. We would have planning meetings, but people wouldn't come. One of the reasons it was really successful was that this large of a team didn't really exist

at the company before. And with this much money, and this much visibility to external players, and all that stuff – from a management perspective or a top-down perspective, I don't think they knew how to really deal with it. So there was a lot of disorganization at first. That was a real challenge. There was not a lot of transparency, not a lot of understanding of who was supposed to do what, and when does it need to get done.

I think there was a lack of – we were responsive as opposed to proactive, and it's really hard to be proactive because you get a lot of feedback on the user interface side. Users can't tell when the algorithm isn't working, but they can tell when the icons look bad. So we had to find a balance of being proactive in terms of defining what we should do, and being responsive in terms of showing off the back end pieces that we really needed to show off. We were trying to push for good research, but [partners on the OS and system processing side] wanted product. They would say, oh, we have this really cool tracking algorithm, we need you to show it off on this map. And the other team would have something else. So we weren't super effective in actively managing expectations and doing something useful for the user at the same time.

I think another piece was that there were a lot of top-down requirements to have a lot of meetings at a high level to present to [our primary customer]. So that led to not a lot of work getting done because we were preparing for those meetings and trying to communicate that we had done a lot of things. But really what we did was prepare reports and prepare for meetings. The program manager at the highest level on the [customer] side came from a software industry background. So he was used to moving at a certain pace and having a certain amount of insight on what was going on. He was working with a lot of R&D type organizations, especially [our company] – we did a lot of prototype, doesn't really have to work or integrate with anything, proof-of-concept stuff – so as that pressure flowed down to make a thing and get it out there and sell it, there were some disconnects on how to do that. It's not that it wasn't communicated – those things are often said, we want these deliverables at this time – but then everyone is like, well it's only research, it will be fine if we're only here. And there were inconsistent understandings of what was supposed to be delivered at what time.