



Rothman Consulting Group, Inc.

Setting Expectations Between Engineering and the Three PMs

Johanna Rothman

Coauthor of *Behind Closed Doors: Secrets
of Great Management* and

Author of *Hiring the Best Knowledge
Workers, Techies & Nerds*

www.jrothman.com

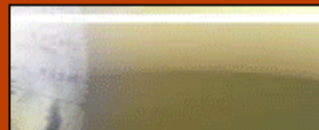
781-641-4046



EXPERT SERIES

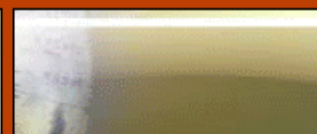
Roles with Respect to Product Development

- ❖ Engineers: solve/implement problems for a given project
- ❖ Project managers: manage a given project
- ❖ Program managers: manage multiple projects, including multiple releases of the same product
- ❖ Product managers: in the context of managing and organizing multiple programs, manage which problems to solve in which release



“PM” Can be a Generic Term

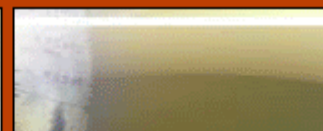
- ❖ Names are interchangeable
- ❖ People perform multiple roles
- ❖ Result: unclear project, program requirements, unclear lifecycle requirements



Product/Program/Project Management Intersection

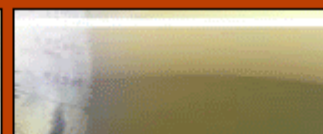
Product Birth	Capture Market Share			Mainstream
Release 1	Release 2	Release 3	Release 4	Release 5
One program	One program			One program

- ❖ Releases 1, 2, 3, 4, 5 are all projects. Each has an Engineering (and other) project manager(s)
- ❖ Each project has a different duration
 - ❖ Project manager selects a development lifecycle for each project based on risk (which includes schedule)
- ❖ Release 1 is a program. Releases 2, 3, 4 are planned multi-project releases with a program manager
 - ❖ Program manager works with the project manager(s) to manage the staged delivery aspect of the program
- ❖ Product Manager plans release durations and which requirements to satisfy in each release



The Problem

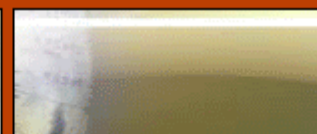
- ❖ Engineering can only *estimate* which features they can complete in a given project
- ❖ Project management can manage how the work is completed to reduce risk of incomplete or defect-laden product
 - ❖ Selecting a development lifecycle and practices
- ❖ Program management manages group- or series-of-project-interactions
 - ❖ With multiple projects and from multiple groups
- ❖ Product management sets the product strategy
 - ❖ And unless everyone works in an adaptive way, they have no more “control”
- ❖ Estimates are guesses. How does everyone manage the ambiguity?



Setting Expectations

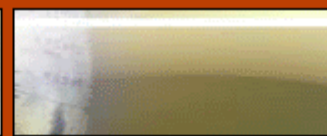
“The significant problems we have cannot be solved at the same level of thinking with which we created them.” ~ Einstein

- ❖ Normally, you'd expect to hear me discuss negotiation, influence, maybe even service level agreements
- ❖ Instead, I'm talking about practices that help to implicitly and explicitly set expectations



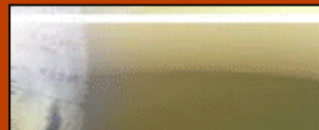
Some Solutions to Build in Adaptability and Risk Management

- ❖ Engineering
 - ❖ Implement by feature
- ❖ Project management
 - ❖ Track what's completed and what's left to do
- ❖ Program management
 - ❖ Continuously adapt the plan as the projects unfold
- ❖ Product management
 - ❖ Release log and backlog (in SCRUM: Sprint Backlog and Product Backlog)
- ❖ For all
 - ❖ Spend as little up-front project time defining requirements if you expect any changes at all
 - ❖ Spend as much time as necessary refining the requirements during the project, as the project unfolds



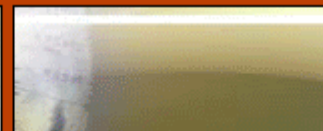
Just the Facts, Ma'am

- ❖ I have never worked on a project where we had any confidence we knew all the requirements at the beginning
 - ❖ I have worked on many projects where we had illusions that we knew the requirements
- ❖ Software is the art of refining requirements
 - ❖ Every project I've worked on has had evolving requirements
- ❖ Estimates are just guesses



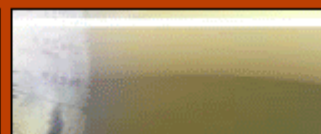
Product Managers Organize Requirements

- ❖ Gather the wish list: “requirements”, featurettes, user stories, defects,...
- ❖ Rank order everything (1, 2, 3, 4. Not H, M, L)
 - ❖ I like pairwise ranking
- ❖ Ask the developers to guess/estimate at how much they can do in a release
- ❖ Draw a line
- ❖ Above the line is the release log (Sprint Backlog)
- ❖ Below the line is the backlog (Product Backlog)
- ❖ Agree with the project managers and Engineering how often you will re-rank (Rothman’s Re-Ranking Riddle)
 - ❖ The more often you re-rank, the more you waste Engineering’s time unless Engineering creates deliverable product between re-ranking
 - ❖ The less often you re-rank, the less responsive you are

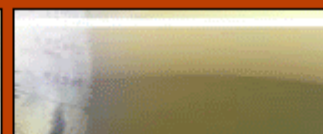


Defining and Refining Requirements

- ❖ Timebox requirements gathering and definition
 - ❖ Requires project manager to plan for timeboxing and explain how, monitor work, and adapt the plan once the timebox is over
 - ❖ Program manager may have to replan the program (what's ready when)
- ❖ Refining the requirements with Engineering
 - ❖ Requires product manager be available at many times during development

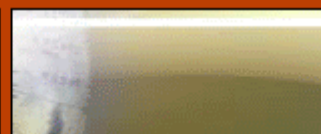
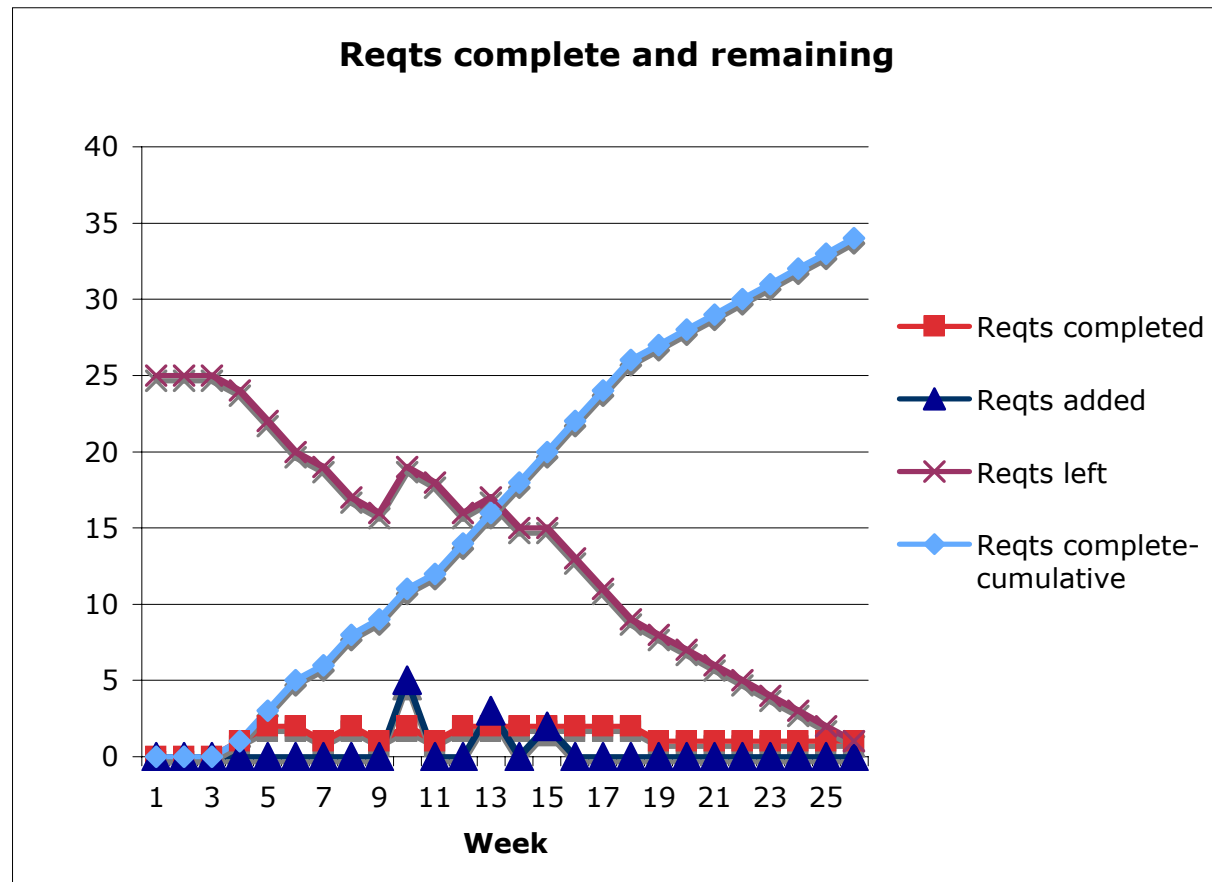


- ❖ Program managers juggle series of projects and/or multiple groups' deliverables
 - ❖ If one group has a problem delivering, manage the issues so that the release is affected as little as possible
 - ❖ Requires proactive risk management



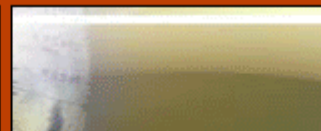
Project Managers Organize and Track the Work

- ❖ Project managers tend to only track what's been done
- ❖ Adding requirements causes what's left to grow
- ❖ Tracking both gives you more data
 - ❖ Feedback on estimates for Engineering
 - ❖ Feedback on project and program planning

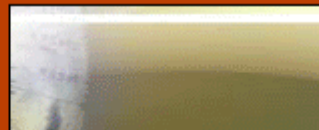


Implement by Feature

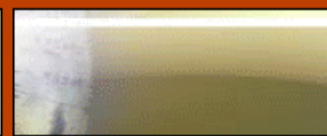
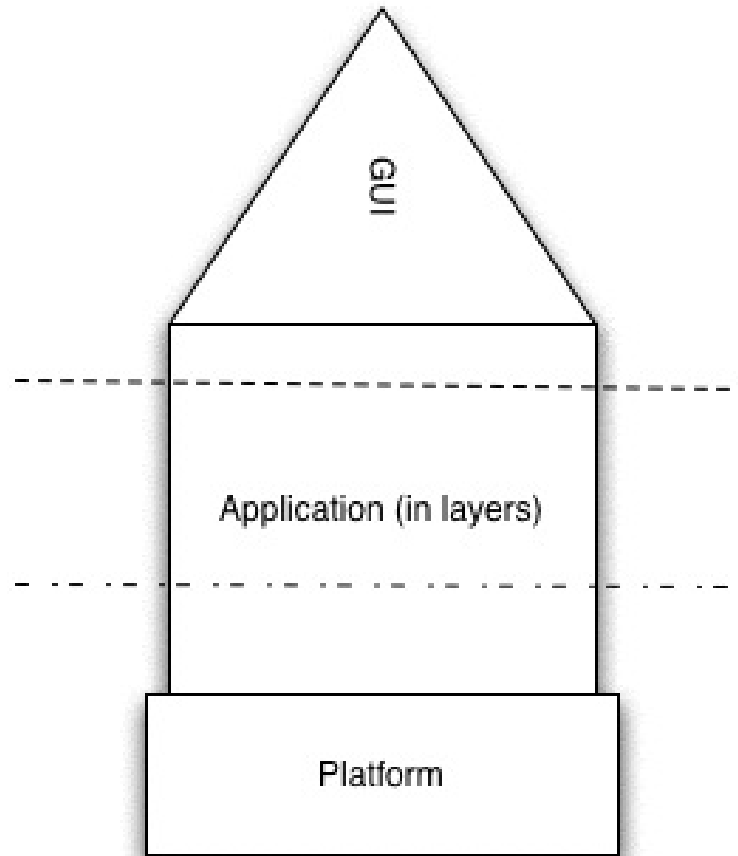
- ❖ Seems to be efficient to implement by architectural piece
- ❖ Turns out, that's least efficient
 - ❖ Leads to 80/20 problem: 80% of your customers use 20% of the product
 - ❖ Rothman's Rule: at least 20% of your product is unused by anyone
- ❖ Develop by feature, not by architectural component



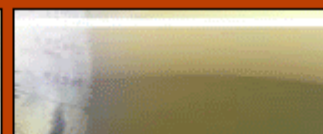
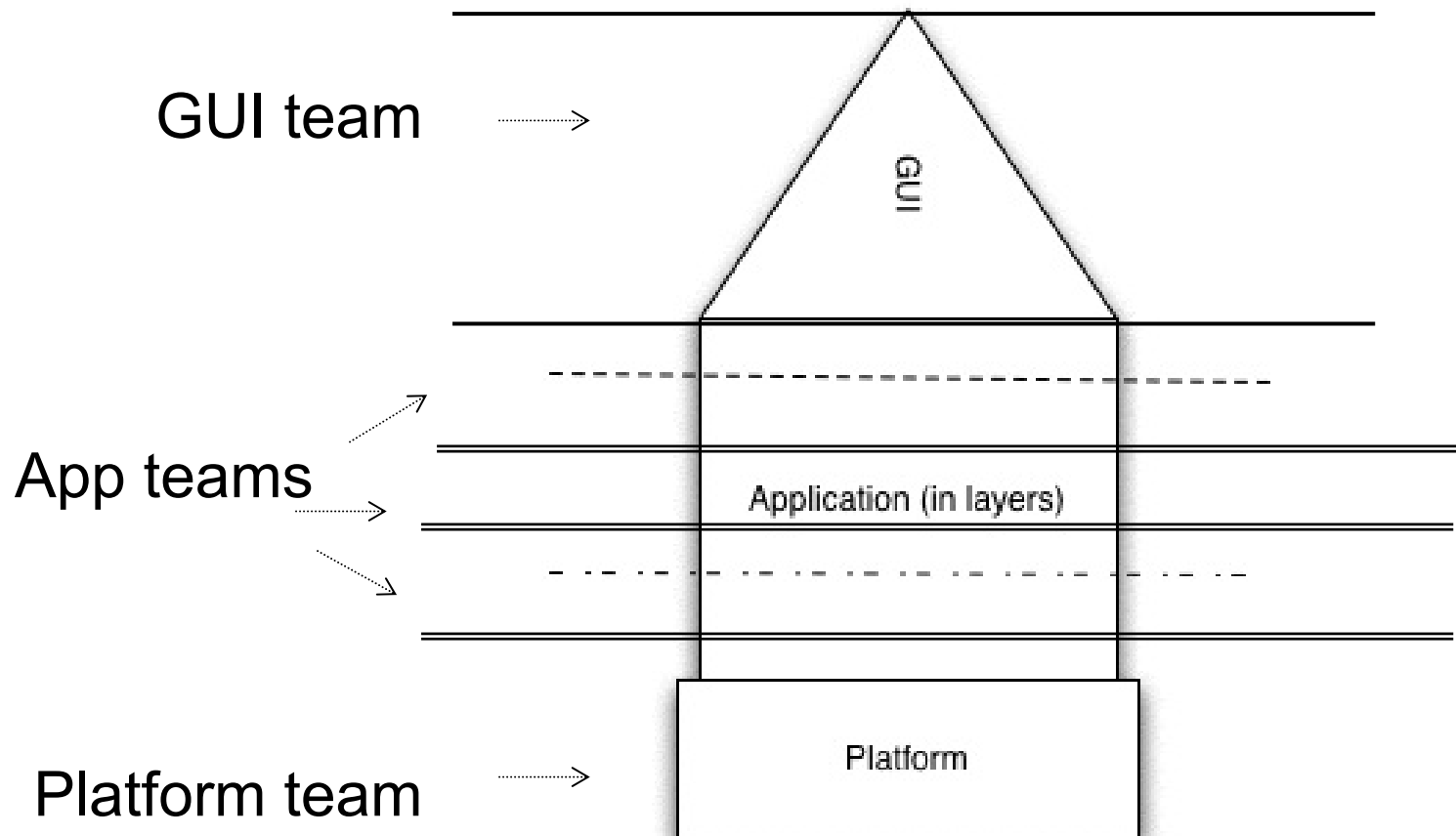
- ❖ Traditional development: Develop each architectural piece, possibly in parallel
- ❖ Implement by feature: Develop each feature entirely through the architecture before you start another feature
- ❖ Your organizational structure can facilitate this (or not)



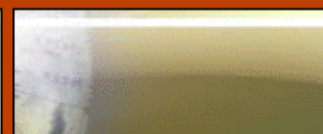
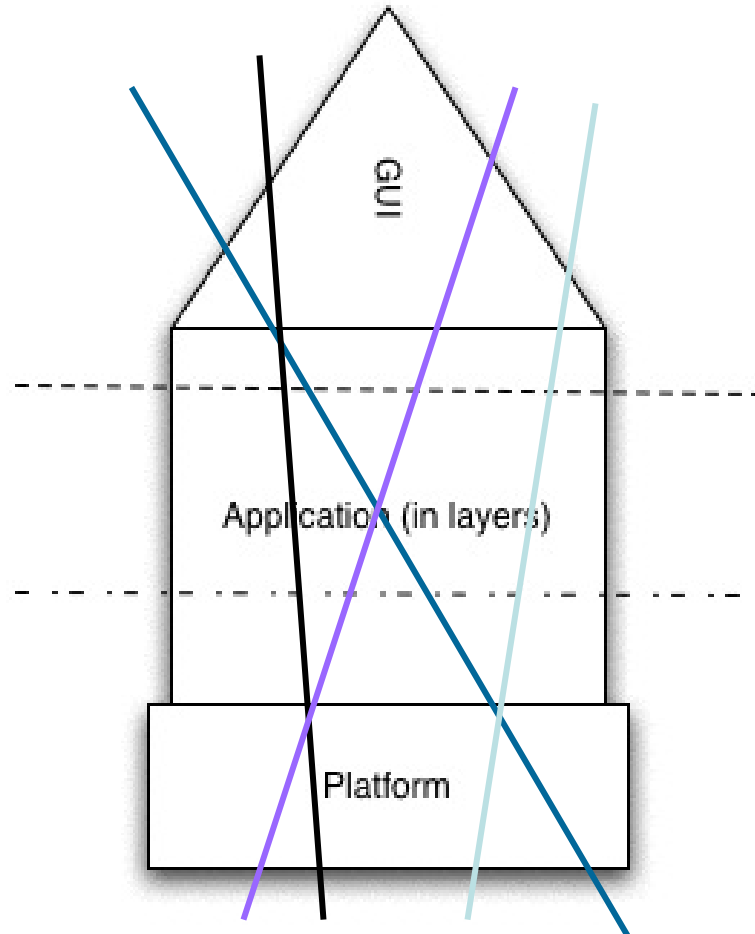
Architecture-Based Development: Prototypical Application



Architecture-Based Development: Implementing by Architecture

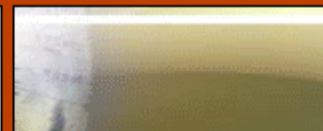


Pictorial View of Implement by Feature



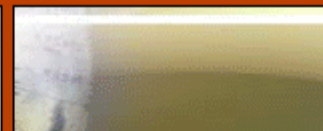
Organizational Structure Can Impede Implementation by Feature

- ❖ Any organization, as long as it is organized by architecture, impedes implementation by feature
 - ❖ Functional organizations
 - ❖ Matrix organizations
- ❖ Project organizations *tend* to be more adaptable to implementation by feature (unless they also have individual code ownership)
- ❖ Implementation by feature seems to require code stewardship, not code ownership



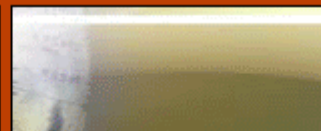
Practices Help

- ❖ Practices that help you
 - ❖ See product pieces early
 - ❖ Organize multiple groups' deliverables and multiple releases
 - ❖ Decide which requirement goes where
 - ❖ Track what's left to do
 - ❖ Improves estimates as you proceed, to refine what you can/cannot accomplish
 - ❖ Help you adapt to reality
- ❖ Are more useful than influence and negotiation alone
 - ❖ Don't forget to build alliances and relationships across the organization
 - ❖ Remember to negotiate on principle, not position



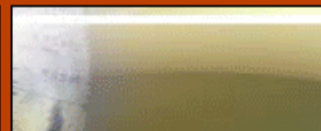
References and Resources

- ❖ AYE articles, www.ayeconference.com/articles.html, especially:
 - ❖ Focus Your Project
 - ❖ Lullaby Language
- ❖ Managing Product Development Blog, www.jrothman.com/weblog/blogger.html. Specifically these entries:
 - ❖ Producing Software is the Art of Requirements Refinement
 - ❖ Art of Timeboxing
- ❖ Top10.pdf (from my newsletter page)
- ❖ Other blogs:
 - ❖ Productmarketing.com
 - ❖ Focusedperformance.com
 - ❖ reformingprojectmanagement.com
 - ❖ bradapp.blogspot.com (see the entry about code stewardship)



Acknowledgements

- I appreciate Cara Moroze, Chris White and all the Orasi folks I don't know about for organizing this series of webinars. Knowing I had an audience helped me write down and think about how to explain things publicly I've been only explaining privately to clients.
- I appreciate each of you for joining the calls, and for asking me questions on the call and later in email. Your questions helped me hone my message.





Orasi Software Event Update



EXPERT SERIES

Orasi's Public Training Workshops

❖ In Search of Excellent Requirements

In 2 full-days, this workshop combines lecture, practice sessions and class discussions on requirements problems and solutions. Practice sessions will give attendees experience in working with use cases, drawing a dialog map, reviewing a requirements specification, writing good requirements, and developing an action plan to improve their group's requirements practices. \$995

October 27-28 in Dallas

❖ How to Create a Product Strategy

Develop and communicate a clear and concise product strategy the whole organization buys into. \$795

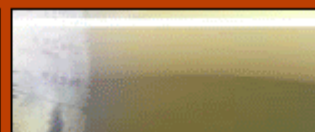
November 9 in Dallas

❖ How to Create & Execute a Product Release Plan

Focuses on simple techniques for defining, prioritizing and validating problems and solutions to create and execute a product release plan that will significantly improve your product delivery process. \$795

November 10 in Dallas

Go to Orasi.com, click on Events, select Seminars & Workshops

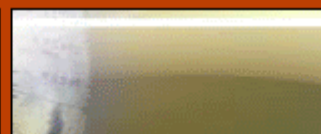


EXPERT SERIES

Attention: Product managers, project managers, QA managers, business analysts, or anyone else who is responsible for ensuring a product or project is properly tested based on the requirements will benefit from attending this webinar.

Thursday, October 27, 2005	2:00pm Eastern
Monday, November 7, 2005	2:00pm Eastern
Tuesday, December 6, 2005	2:00pm Eastern

[Go to Orasi.com](http://Orasi.com), click on Events, select Online Events



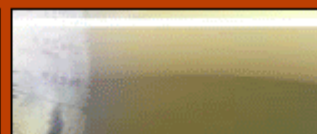
IdeaScope Technology Briefings



In 45 minutes, let us demonstrate how IdeaScope enables organizations to proactively collect insight from customers to achieve buy-in, balance requested features, make objective decisions, and develop products and services that translate into growing sustainable revenue when taken to market. Join us...

October 25, November 8, December 13
9am and 3pm Eastern

www.orasi.com/briefing.php



EXPERT SERIES