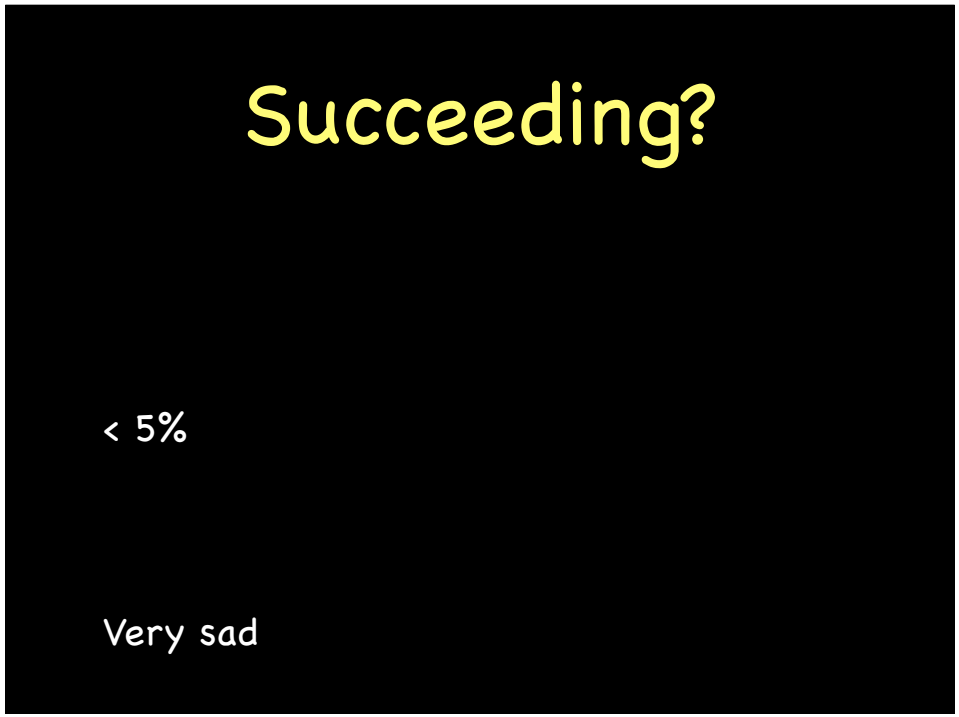
Patterns context is the big E enterprise. Additional context as we go along

Architecture conference workshop -- AKA lions den

Their role in their organization

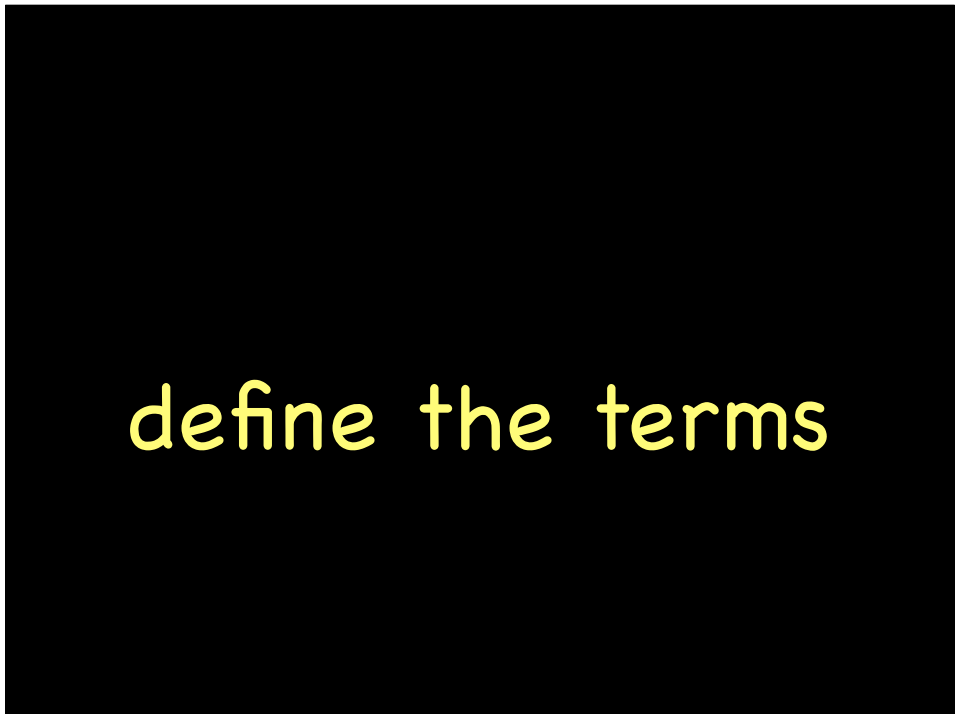
I thought lack of success resulted from unclear expectations



Guess not. Other issues, like the organizational set up, contribute to this result. And some of the behaviors I'll describe.

Agile will not succeed without addressing legitimate concerns of EAs

Hypothesis.
BTW, I don't shoot architects on sight and I am not anti-architecture



Note I am not discussing “succeed” here

Agile will not succeed without addressing legitimate concerns of EAs

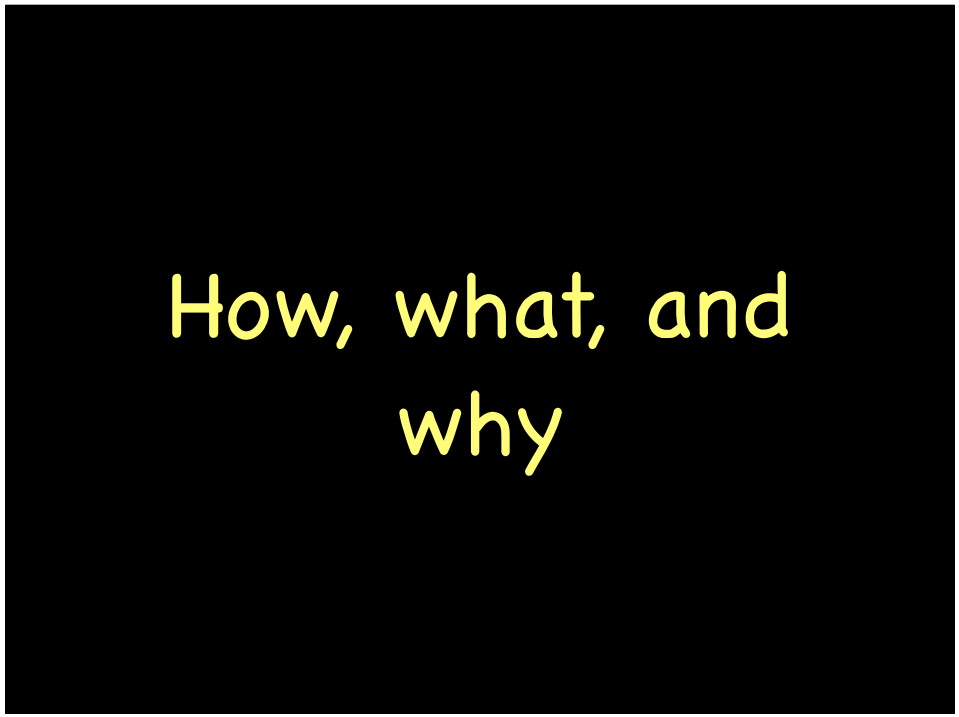
Principles of agile: rapid feedback, visibility into progress, adaptable, frequent peer review, focus on the value/cost equation.
Also, not just about development but building an organization that operates to these principles.

Agile will not succeed without addressing **legitimate concerns** of EAs

They do exist. Summary -- responsible for the value of the business asset (software and dev capability) that delivers the value to the business. It isn't the software but the value delivered that matters.

Agile will not succeed without addressing legitimate concerns of **EAs**

Many definitions. Focus here on enterprise, data, integration and technical aspects of arch rather than app arch.

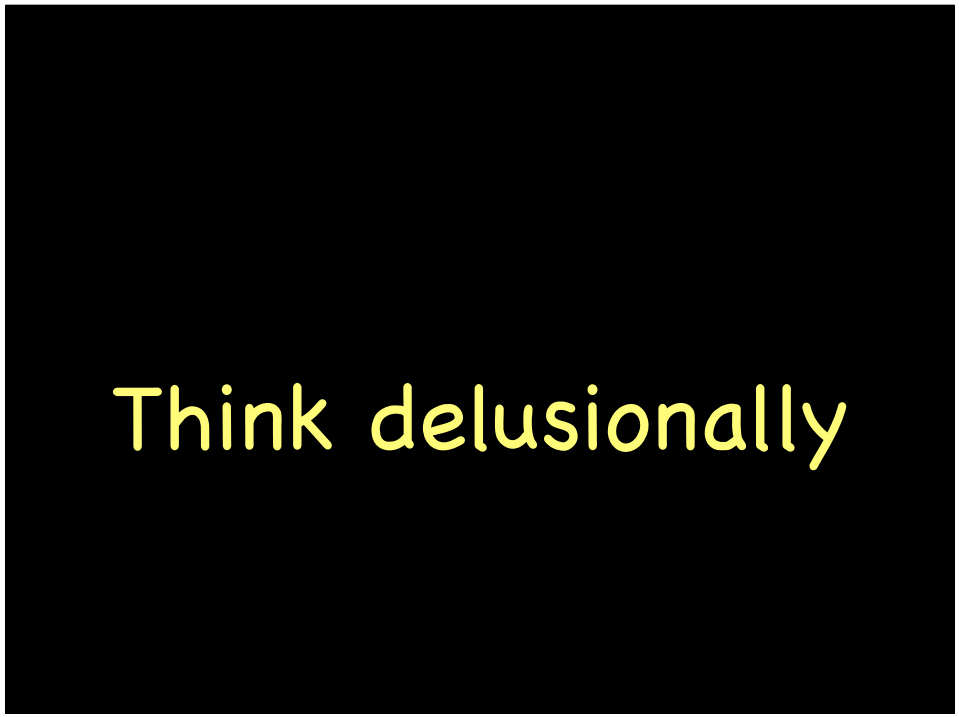


Three aspects of how EAs can do their work within the context of a dev effort following agile principles. For each: problem statement, some observed behavior and a “pattern” for productive behaviors. How to work, what to deliver and why we’re delivering what?

How do I do my job? How do I know they're "doing what they're told"? How do I tell them what to do. How do I tell them what I am worried about?

General office folks swoop in from on high, no concern for what's already there, make a big mess, and leave. Folks clean up the mess and pay no more attention to the contribution.

Gasp! Write code? Read code? Yes.
Org problems and ego problems and possibly skills issue. But yes.



(A over B means B still has to be good -- I don't have that constraint.
0% success is effective. Lack of innovation not related to stifling initiative.

Engage with the individuals. Devs and archs are both human beings. Really. Shared understanding.

Pattern 1: Member of the Team

Be vested in success. Have context. Share concerns. Articulate risks. Articulate short and long terms costs of choices.
Org issue -- we're outnumbered? Virtual arch group

What are the artifacts? How do I communicate what I want and need? How do we define success?

Remember the seagull? Still, this is the SOP for most architecture groups: standards documents. Written out of context. Often ignored.
Can be consequence of being outnumbered as well.

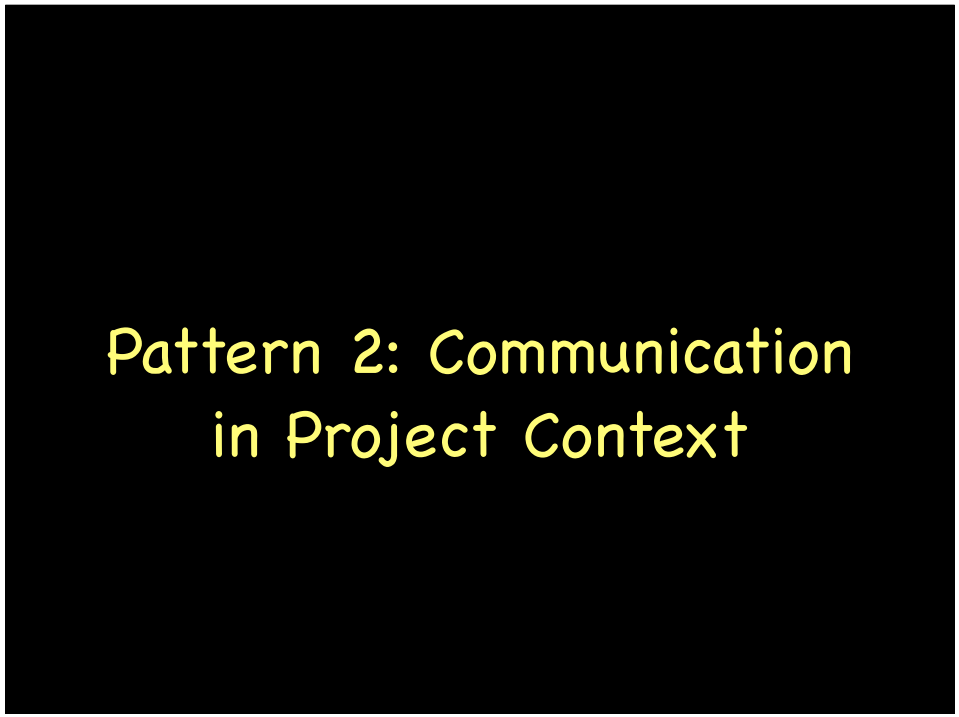
With articulated business value, risks, etc. Allows for prioritization and understanding of effort.

Criteria most important, automation where possible. What does maintainable really mean? How do we know we're done? Arch gets to sign off on story!

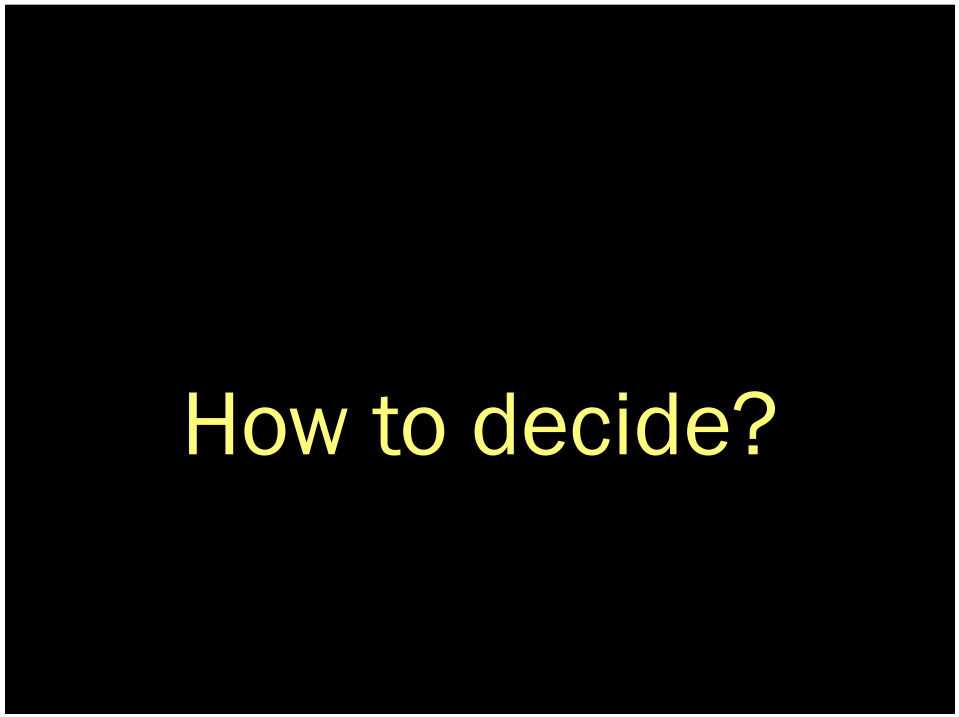
Scary but true. Arch's view of what the reusable components are, how to use them and what the teams' need (not always with input from the teams). Yes Arch has enterprise context. Descriptive vs prescriptive

When you see actual re-use, harvest it. Don't speculate about what it will look like. See it and exploit it. And communicate it. Virtual team again. Guessing right most of the time is hard.

Documents assumptions and allow for parallel effort. Invaluable for integration projects (and most enterprise projects are such animals).



Fit the needs of the architect into the way the team actually works and consistent with principles. Don't speculate. Don't be vague. Project is the driver so their work patterns matter most. So use them to achieve your objectives.



Why do we do what work when?

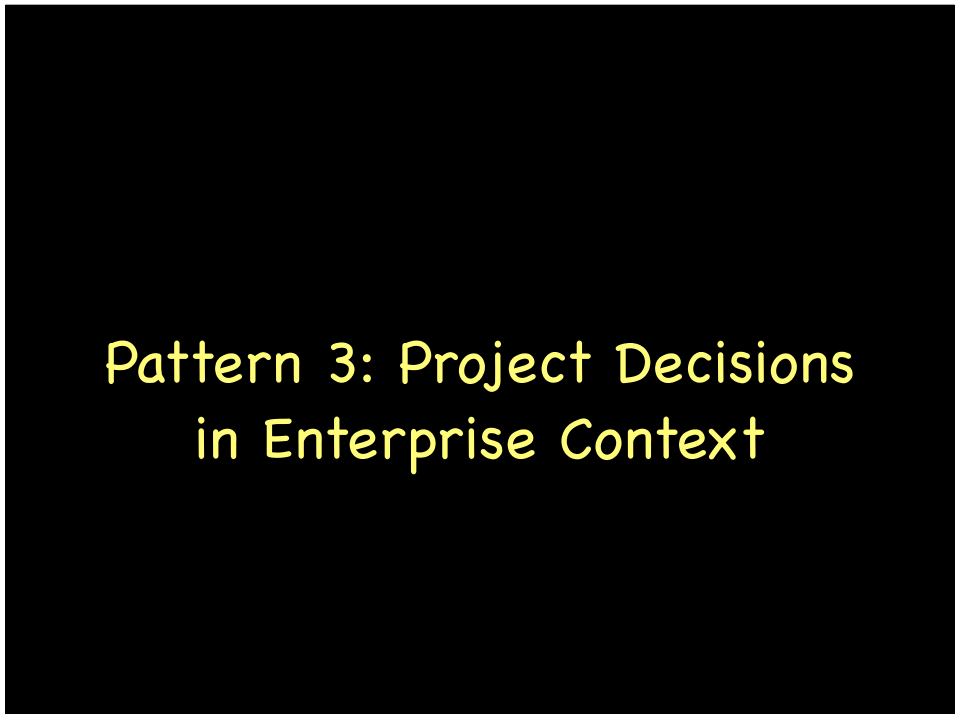
They'll never understand (for some value of they) so we'll need to make them do as we say or we'll just ignore them and do what we want. I need a bigger stick. AKA ignore the problem. Result - the wrong things get done.

During prioritization, explain why the arch stories are important in business terms. Understand time scales and relative value. If you consistently fail, perhaps the initiative needs rethinking? Maybe?

We can't start until we have everything worked out. Data model, reuse framework, etc.
That means we'll never actually start. Does it really have to be the whole OPS manual?

Informed risk taking

Last responsible moment. Plan what needs to be planned.
Prioritize for risk as well as value, but explain it.
Business has the right to take risks – we must explain risk but it is their call. Doesn't always work out well, but still...

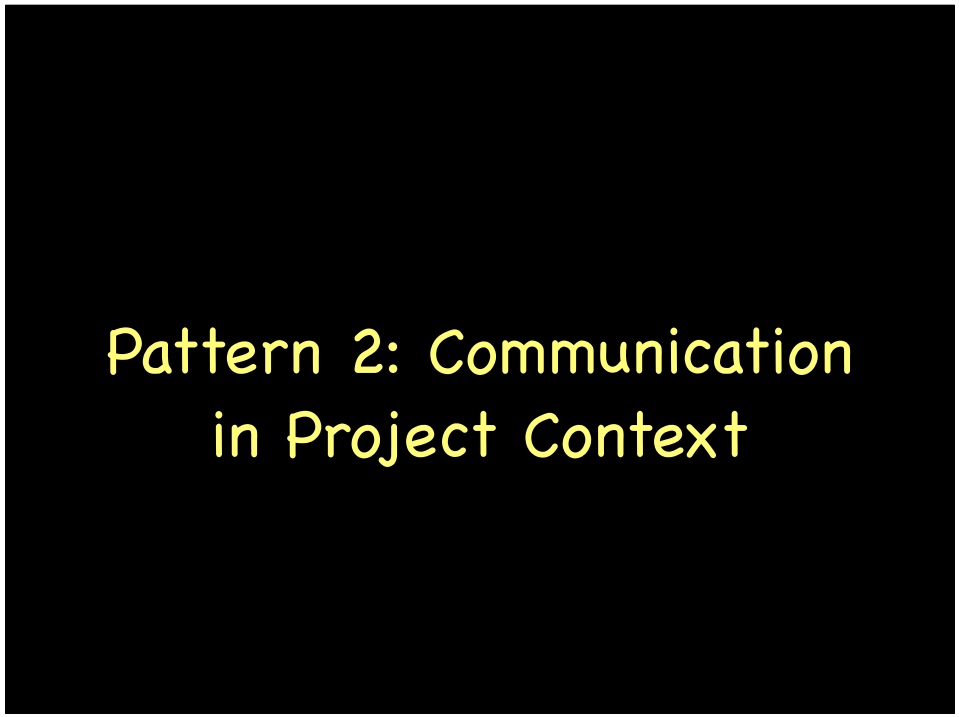


Projects are focused on delivery in the short term. EA must also look out for longer term. Must balance but someone must have the big picture. Project teams are narrowly focused.

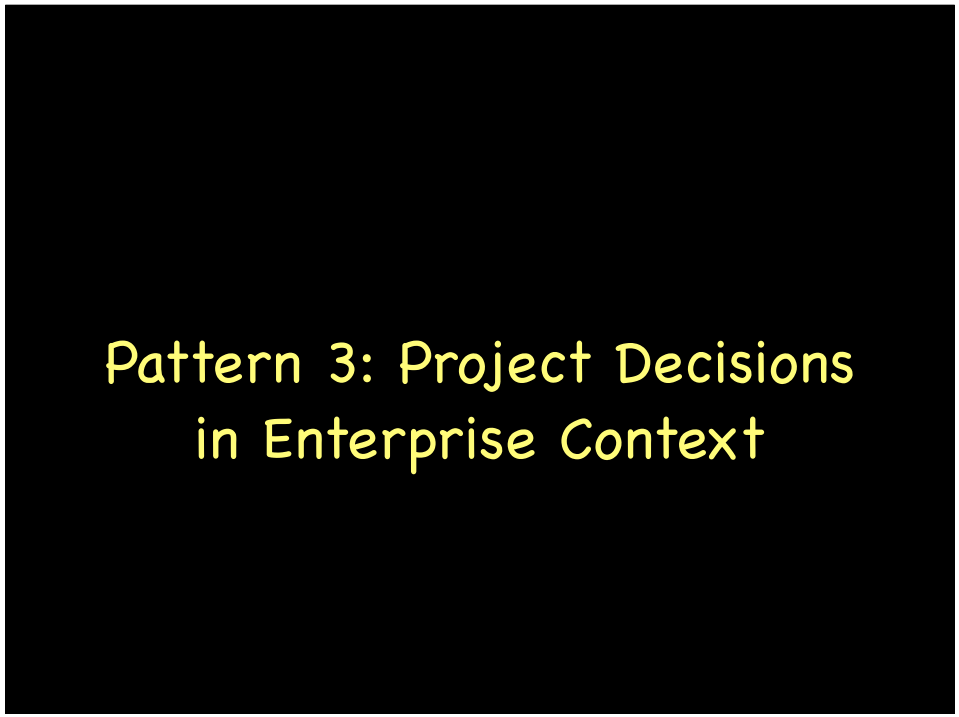
Recap of patterns

Pattern 1: Member of the Team

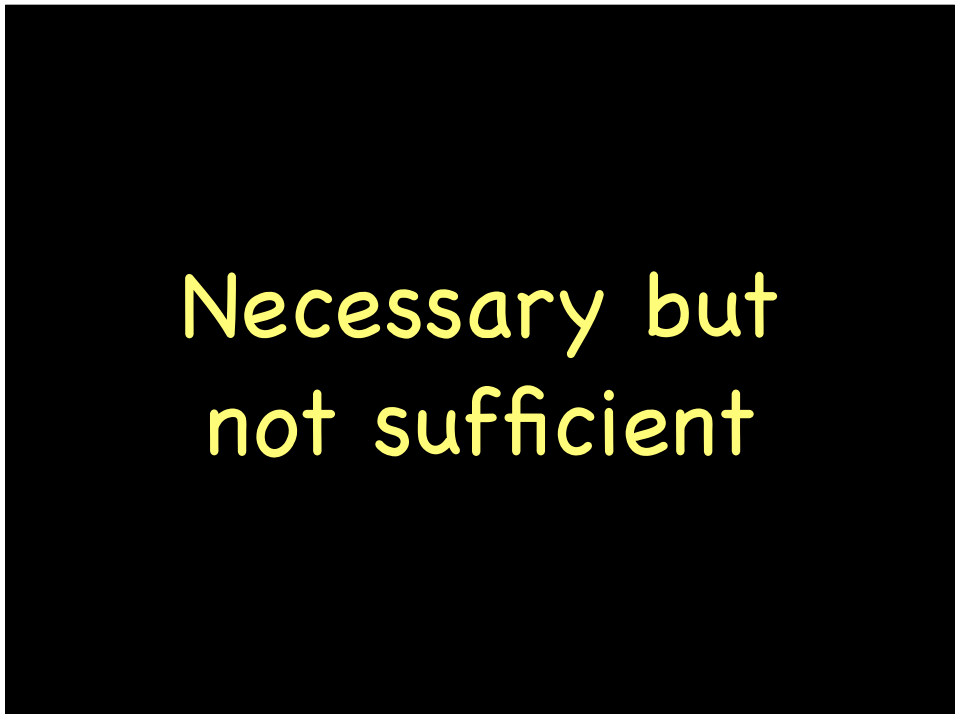
Architects are people. Devs are people. Get work done by collaborating as an equal on the team. Dictates don't work well. Figure out how to get leverage to overcome the numbers imbalance. Virtual team.



Deliver artifacts consistent with the work patterns. Use the artifacts to increase effectiveness. Effectively re-use (ops handover for example).



Balance time frames. Understand and accept risk responsibly. Informed decisions. Give the business their levers. It's their business.



And yes, it can still go horribly wrong. Requires change in roles, in organizational relationships, in staffing, and in relationship building. It really is people over process.

Questions?

http://www.thoughtworks/com
http://rebeccaparsons.com