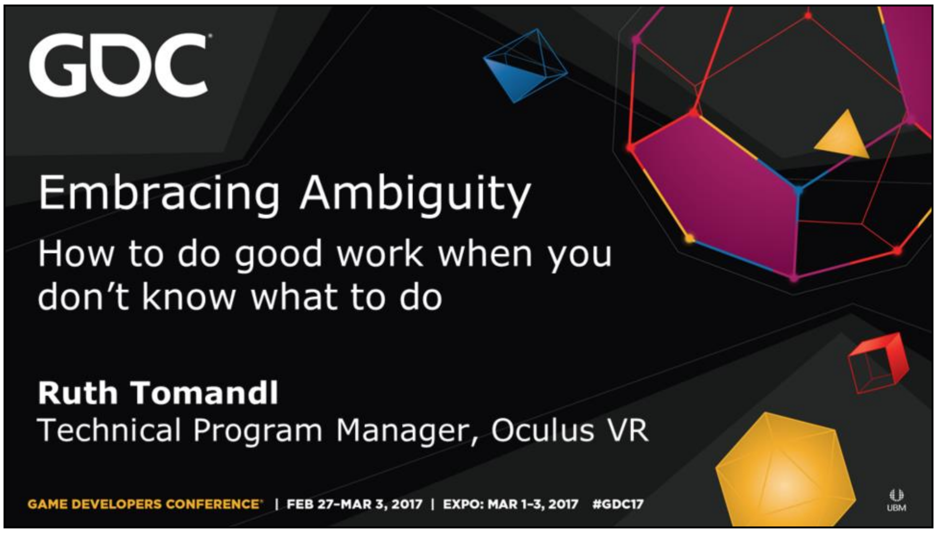


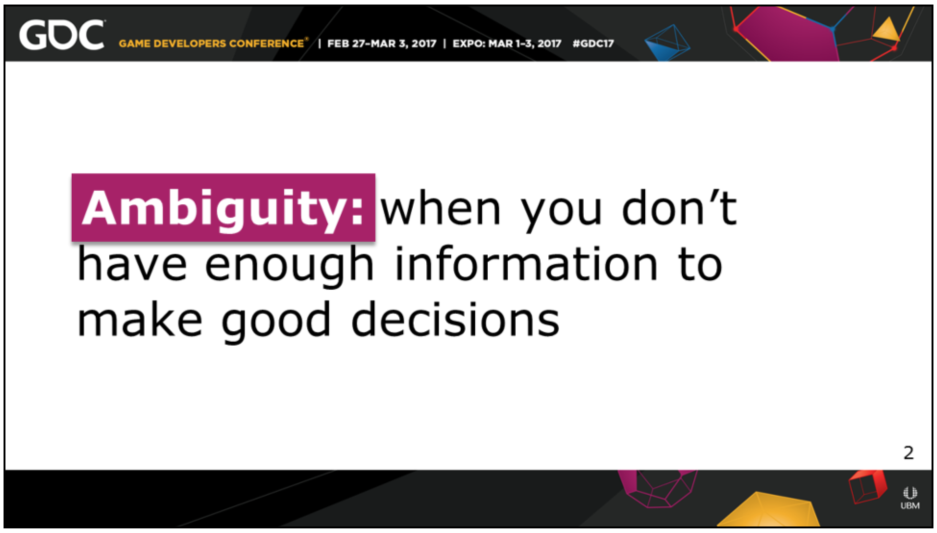
Embracing Ambiguity

How to do good work when you don't know what to do

Ruth Tomandl
Technical Program Manager, Oculus VR

GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

To start with, some definitions: I'm going to use the word "ambiguity" a lot in this talk, so what exactly am I talking about?



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Ambiguity: when you don't have enough information to make good decisions

2

UBM

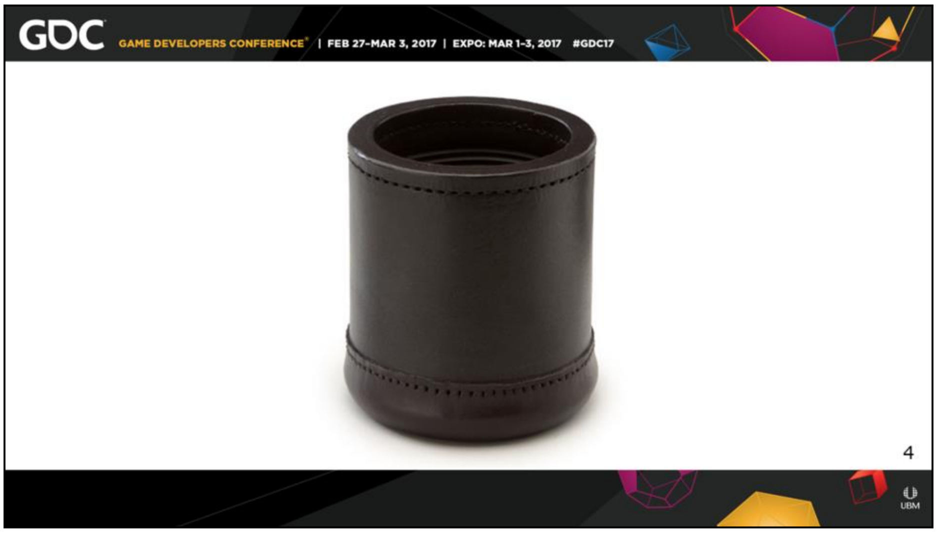
Ambiguity is when you don't have enough information to make good decisions.

Now, ambiguity is different from risk, and here's how:

GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
3

When you bet $100 on a coin flip, you're taking a risk: you know the approximate odds of it coming up heads, you know how much you'll win if it doesn't and how much you'll lose if it does. You can make a very good decision about whether to take that bet based on that information.

Humans are extremely good at risk. We've built an entire casino industry around how much people love to gamble on outcomes they can predict.


GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
4
UBM

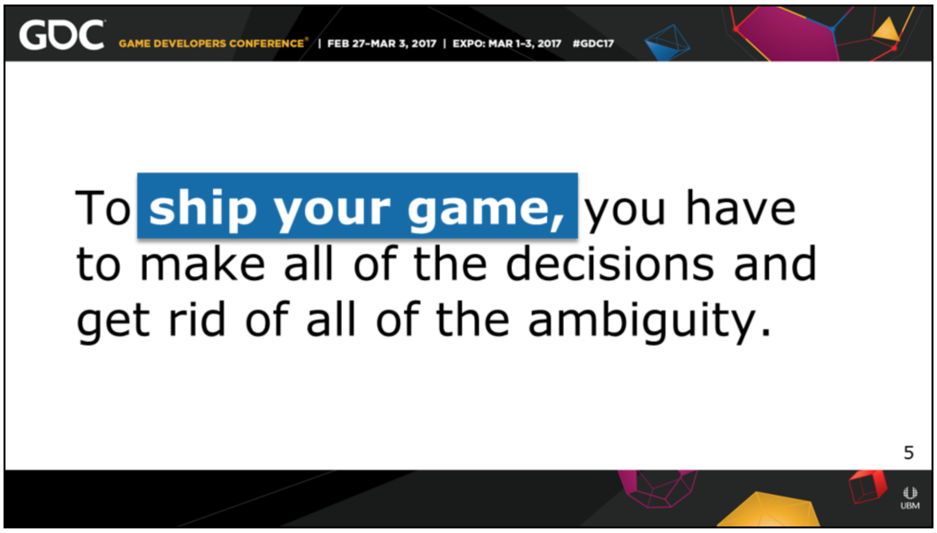
But if you bet $100 on whether when I roll the dice I have in this cup, they'll add up to at least 20, it's much harder to decide whether to take that bet, because you don't have as much information. I might not have enough dice in my cup to make 20, or it might be a tiny chance. I might not even have dice in here; maybe this cup is full of rocks. How can you confidently bet on that?

That's ambiguity. It's something humans are really bad at: we do not like gambling if we don't know the odds or possible outcomes. It makes us uncomfortable and afraid because we can't predict what will happen. As much as we seek out risk, we avoid ambiguity.

And yet, that's what we do when we make games, and when we make almost every decision in development. If you need to, say, decide whether to include multiplayer in your game, you probably have very little information on the possible risks or rewards: how much time and effort will it take to add multiplayer? What bugs will it introduce? Will it help you sell more copies? How many more? How much will your servers cost? Will it be more or less fun than the single-player mode? How will it impact your Metacritic score? You do not have enough information to make a comfortable, predictable decision.

Ambiguity is stressful: we don't like it, but game development is full of it.

[For more information about different levels of uncertainty, see section 3 of this paper: http://mitsloan.mit.edu/media/Lo_PhysicsEnvy.pdf]


GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

To **ship your game,** you have to make all of the decisions and get rid of all of the ambiguity.

5

So not only is ambiguity scary and uncomfortable, it will block you from shipping if you don't get rid of it eventually. A game is only finished when every single decision has been made.

Even for live games, you have to decide what is going in your latest update and what isn't, and if a feature or piece of content is going in, you have to make decisions about every single aspect of it.

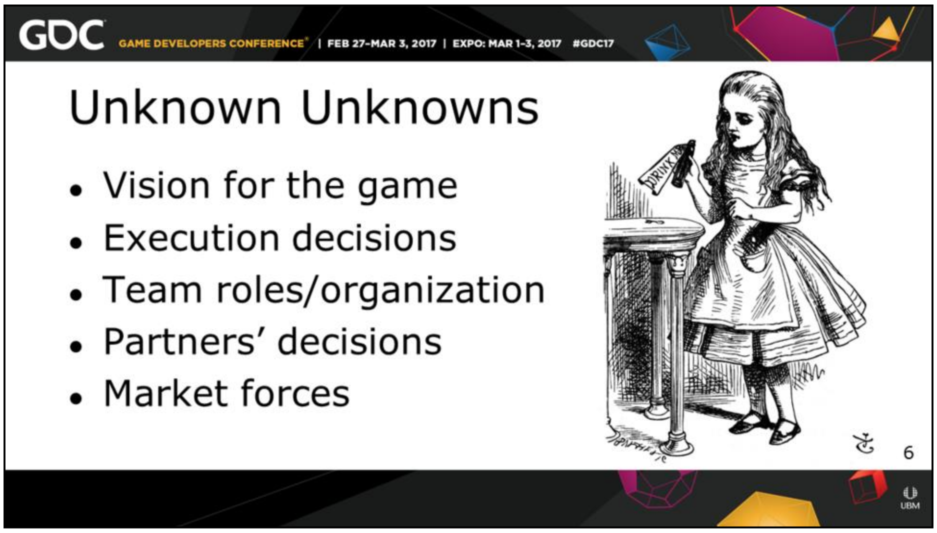
How many people here are working on a project that's super straightforward, well-understood, with really low levels of ambiguity?

Since 2001, I've worked at 7 studios and shipped a dozen games, and I've worked on a project like that exactly once. And that makes sense: most of us got into game development because we want to innovate and make cool things, and when we have the most ambiguity is when we're innovating, or doing something we've personally never done before (which is when we grow the most). And that's what most of us are doing most of the time.

Especially for those of us working on VR: there are tons of unknowns there!

How many of you are working on a really ambiguous game where you still have a bunch of unknowns to figure out?

Yeah. So let's talk about why that's so hard.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Unknown Unknowns

- Vision for the game
- Execution decisions
- Team roles/organization
- Partners' decisions
- Market forces

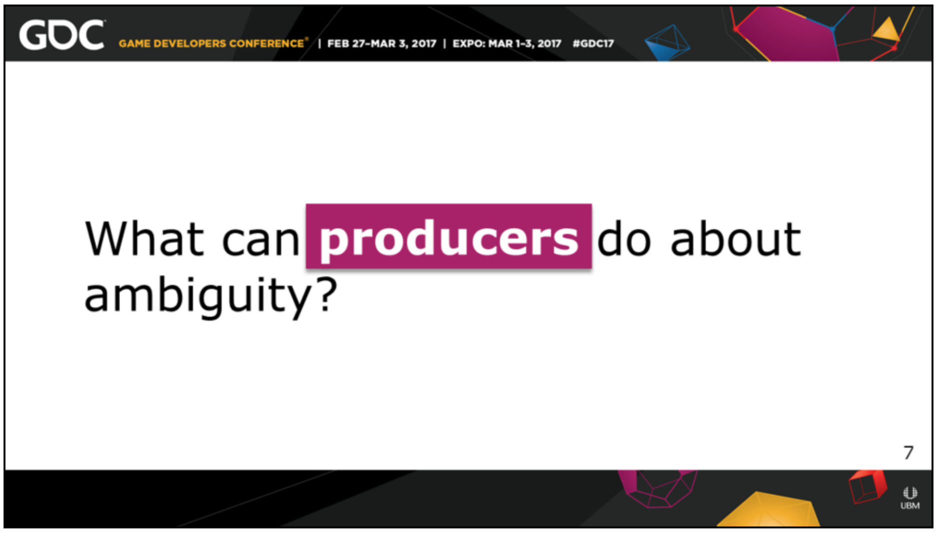
6

It's hard to make good decisions when you just don't know what's going to happen.

And there are a lot of things that might be unknown or unpredictable when you're making a game:

- What is the vision for our game? What are we making? Will it be good?
- How do we make that game? What work needs to be done?
- Who's responsible for what on the team? Who needs to make which decisions? Who's in charge? How do people find out what they're supposed to be doing?
- What is our publisher, outsourcing partner, middleware vendor, etc going to do, and what can we do about it?
- What do players want? Is a game going to come out that will redefine our genre and then we have to reassess our vision?
- What other nasty surprises does the future hold? Is our only multiplayer engineer going to quit? Some of these are more predictable and you can plan for them, but others hit you out of nowhere.

You can never predict everything: there are always unknowns. But you still have to plan for the unexpected.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

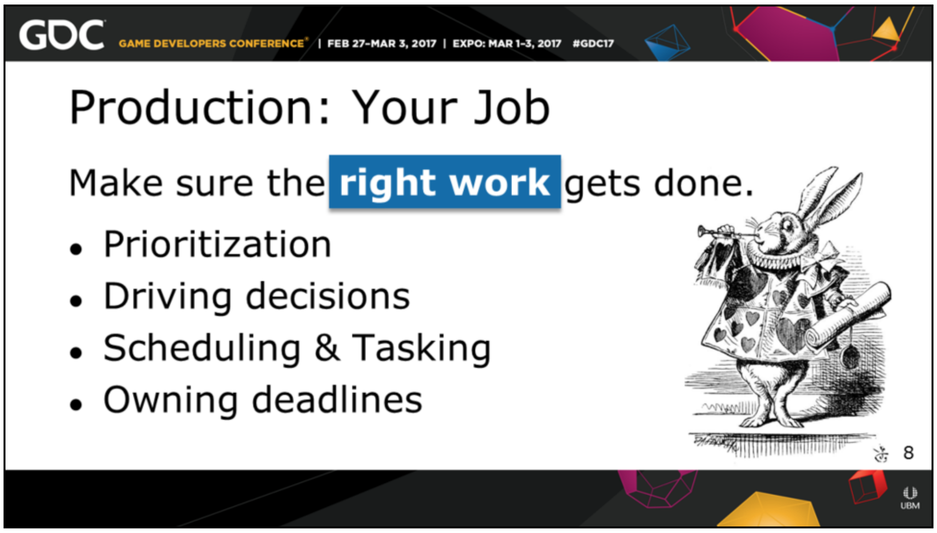
What can **producers** do about ambiguity?

7

UBM

In this talk, I want to talk about the problems that arise from Ambiguity, and what you can do as the producer to solve those problems or avoid them.

Because Ambiguity is something that almost all of us will face on almost every project, so getting really good at dealing gracefully with ambiguity is something that will make you a very valuable producer.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Production: Your Job

Make sure the **right work** gets done.

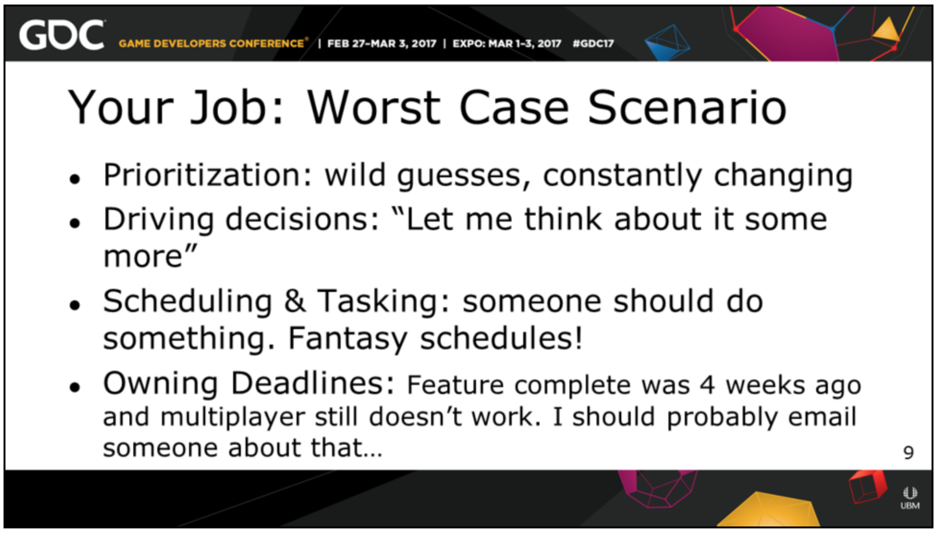
- Prioritization
- Driving decisions
- Scheduling & Tasking
- Owning deadlines

8

Being a successful producer on projects with high levels of ambiguity is really hard, and it's the number one thing that my team looks for when hiring new producers. Because as producers, we can do a lot of things to manage all of this uncertainty. It's a fundamental part of our job.

Your job as a producer is to make sure the right work gets done. Specifically:

- Prioritization: identifying what the right work is
- Driving decisions: making sure the right work is defined when it needs to be, and gathering the information needed to make those decisions
- Scheduling/tasking: communicating what the right work is and who's doing it, making sure the right things are being worked on and that the right people are doing them
- Owning milestones or sprints: making sure the work is getting done (and making sure everyone understands what 'done' means)



GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Your Job: Worst Case Scenario

- Prioritization: wild guesses, constantly changing
- Driving decisions: "Let me think about it some more"
- Scheduling & Tasking: someone should do something. Fantasy schedules!
- Owning Deadlines: Feature complete was 4 weeks ago and multiplayer still doesn't work. I should probably email someone about that…

9

UBM

If you don't have enough information to know what the right work is, or how strongly you should drive things, it becomes very difficult, and sometimes impossible, to do your job well.

- Prioritization: trying to identify what the right work is, but not really getting anywhere.
- Driving decisions: hearing "let me think about it some more" over and over again.
- Scheduling/tasking: communicating that it's very important that somebody should be doing something. You look at the schedule you made 3 months earlier and it has no resemblance to reality.
- Driving towards milestones or sprints: We just missed another milestone unless we just want to call what we have now Alpha, I guess. Sure.

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

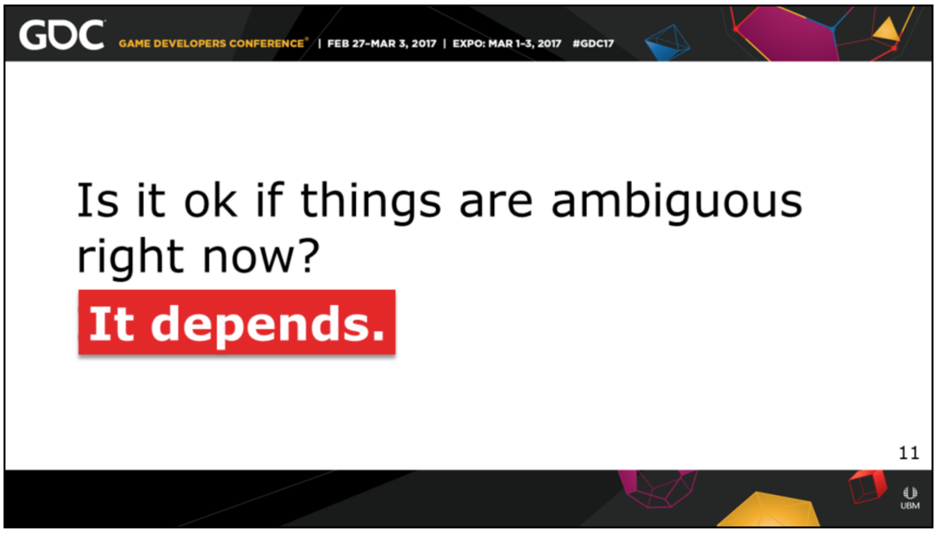
10

UBM

I’ve been in this situation: it feels very bad. And sometimes it feels even worse because you aren’t always sure that it’s actually bad. Everyone else seems to think this is fine. Your lead designer is having a great time thinking up new features to add, and often reminds you that making a great game is more important than hitting some arbitrary milestone.

The engineers keep telling you that this is how other game studios do things: cooler studios, that they’d rather work at, cool studios that don’t even *have* producers, because they don’t need them, because they listen to their engineering team and make cool games.

Oh no, maybe you’re turning into one of those super process-heavy producers that everybody hates? How can you tell? Are you the crazy one, or is everyone else?


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

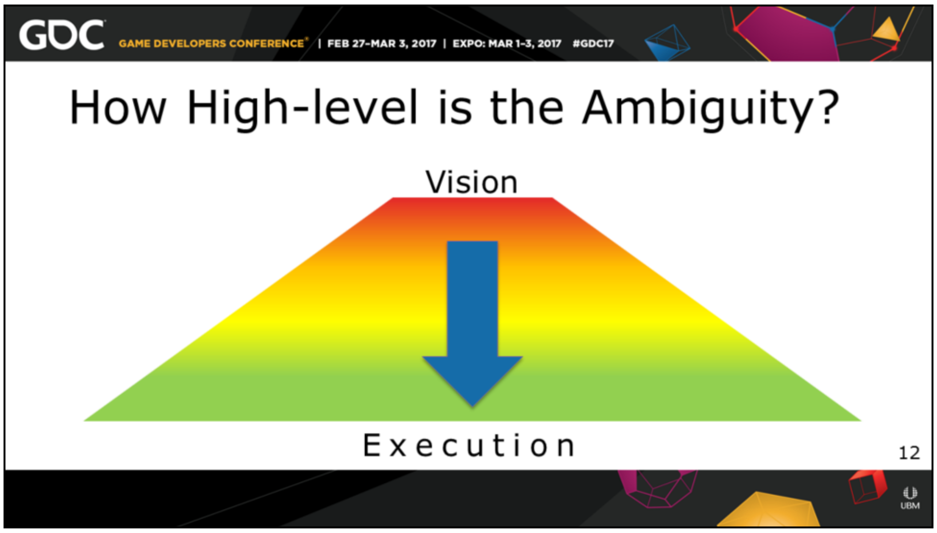
Is it ok if things are ambiguous right now?

It depends.

11

UBM

Whether or not ambiguity is ok right now, on your project, and whether you should fix it or roll with it, depends on a bunch of different factors:


GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

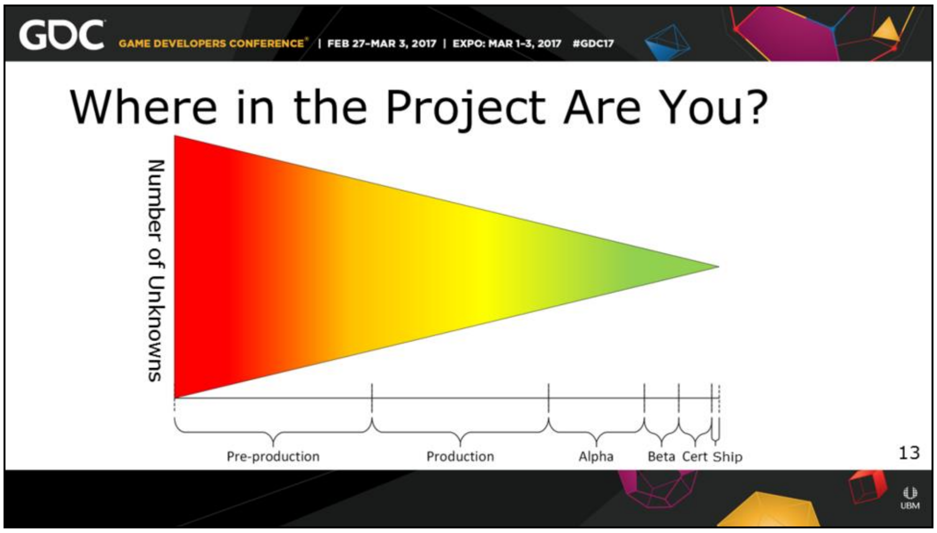
How High-level is the Ambiguity?

The most important factor is: where does the ambiguity lie? To successfully make a game, you create a core vision or design of what the game will be, and then you use that vision to make all of the thousands of lower-level execution decisions throughout development. The higher-level a decision is, the more people it affects, the more of your game it affects, and the earlier it needs to be made.

You need to get rid of any ambiguity in your game's vision as early as possible. I did a whole talk on this last year, and I think it's one of the most critical issues in game development: if you don't have a clear, well-defined, and well-understood vision for your game, your team cannot successfully build it.

If you haven't already, I strongly encourage every producer to read the Game Outcomes Project [http://intelligenceengine.blogspot.com/2014/12/the-game-outcomes-project-part-1-best.html], which was published a couple of years ago. Paul Tozour and his team surveyed hundreds of game developers asking about their development process, teamwork, and culture, and comparing those responses to the success of the teams' finished games. The results are extremely interesting and there's a ton of useful information there, but their strongest result was that teams that had a "viable, compelling, clear, and well-communicated shared vision" were the most likely to make a successful game: critically successful, financially successful, and meeting or exceeding the team's own expectations for what they were making.

If you don't have a clear, strong, and agreed-on vision for your game, then you need to get one, first, before you focus on anything else. And if you do have one, you can rely on it to help you reduce ambiguity down through all the layers of execution.

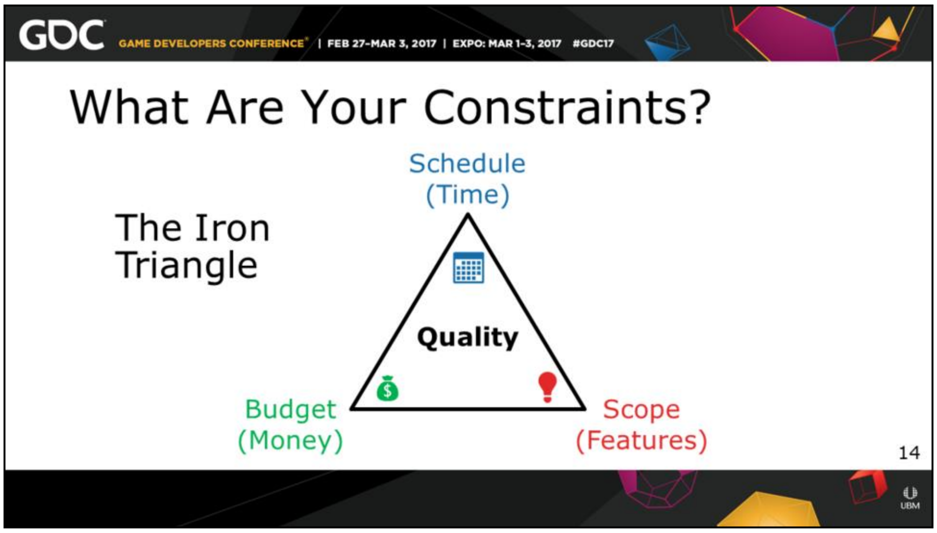

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
Where in the Project Are You?
Number of Unknowns
Pre-production
Production
Alpha
Beta
Cert
Ship
13
UBM

Similarly, ambiguity is less negative, and even can be a positive, the earlier you are in development.

Remember, ambiguity is when you don't have enough information. If you're early on in a project, a lot of what you're doing is gathering or creating that information; trying things to see what works. Almost always, you start with an idea of what your game will be, but you don't know most of the details. You figure out the details as you go and refine your vision.

If you have the time you need, and the trend is from more ambiguity towards more certainty, this level of ambiguity is perfectly healthy. Your job as a producer is to find ways to measure trends and make sure your team is on the right track.

This cone does not get narrower on its own: you have to chip away at it by making decisions day to day, and you have to make sure it's narrowing fast enough to hit that point where every single decision has been made in time for you to ship.


GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What Are Your Constraints?

The Iron Triangle

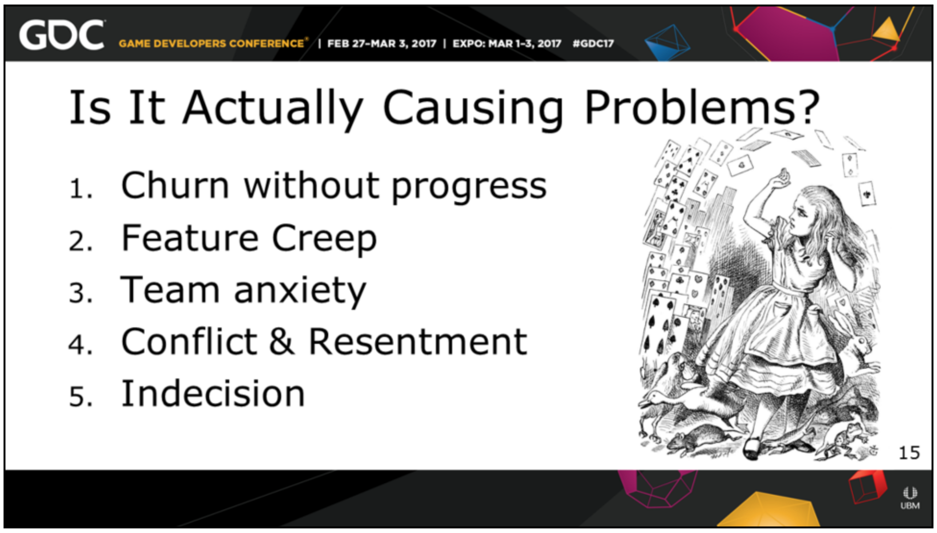
Schedule
(Time)
Quality
Budget
(Money)
Scope
(Features)
14

And third, whether or not ambiguity is a problem depends on your constraints. This is the Iron Triangle. You've probably heard the phrase, "Fast, cheap, or good: Pick two", but in games we usually have to pick all of them and still make something with lots of innovative features. This is a way of thinking about your constraints: the bigger your budget, longer your schedule, and more creative freedom you have, the better your game can be. And the more ambiguous your vision can be at the beginning, because you have the room to refine it and make it great.

If you have all the time and money you need to prototype innovative design ideas for 3 years, then go for it! As the producer, make sure that your team is discovering the things they need to, and that they're working towards a clear vision at the end of that 3 years.

But most of our projects have pretty tight constraints, and we have to make a quality game with a small budget, a short schedule, and required features or design elements. The tighter your constraints, the less ambiguity you're able to have and still ship successfully.

The least ambiguous game I ever worked on was an expansion pack to a sequel. The budget was small, the schedule was tight, and because it was an expansion pack a lot of the design decisions were set and couldn't be changed. We quickly zeroed in on the game design and executed it well, but there was no room for messing around and putting off decisions, and there was very little innovation.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Is It Actually Causing Problems?

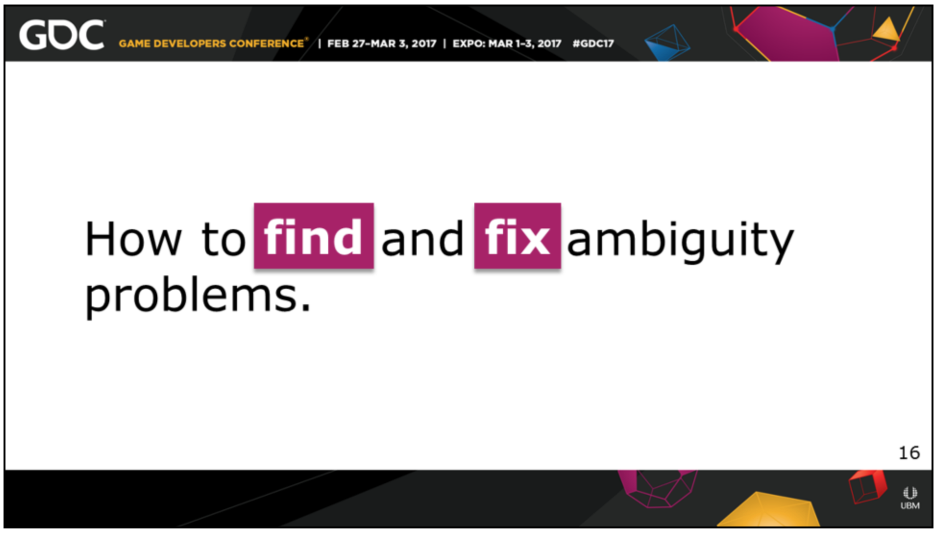
1. Churn without progress
2. Feature Creep
3. Team anxiety
4. Conflict & Resentment
5. Indecision

15

UBM

And finally, is ambiguity causing problems on your project? These are 5 major problems that can result from too much ambiguity too late in a project: I'll talk more about each of these, and give you some tools you can use to fix or avoid them.

Churn without progress: you have too much ambiguity to make real progress on your game, no matter how hard you work.
Feature creep: you don't really know what your game is, so you just add more features to try to make it better.
Team anxiety: your team is afraid of what might happen and they don't have enough information to be confident in your game vision.
Conflict and resentment: You feel like you need to blame someone for the chaos that's keeping you from doing a good job.
Indecision: You don't have enough information to make good decisions, so you just stop making decisions.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

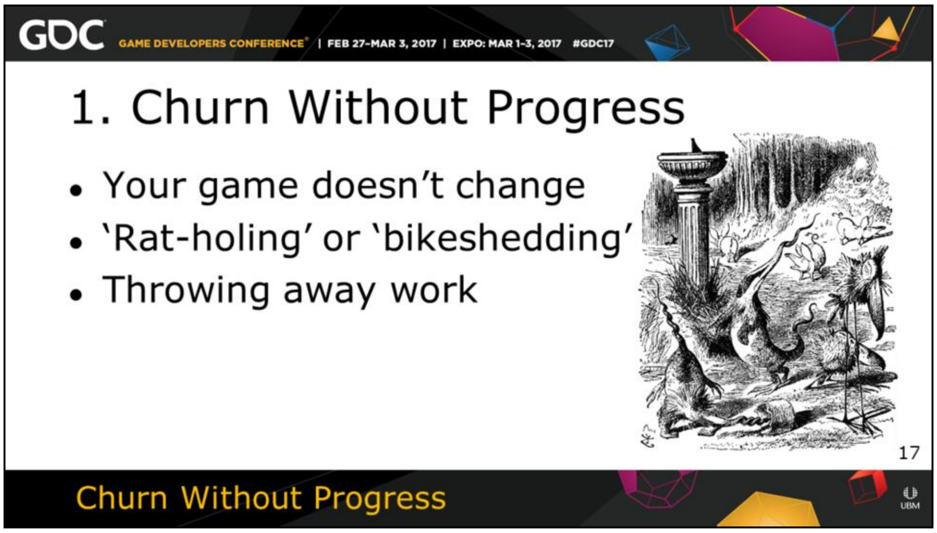
How to **find** and **fix** ambiguity problems.

16

UBM

Obviously most of the time the problems aren't that clear like I'll lay them out here. Even the best, well-managed projects, you'll run into some ambiguity that makes your team start churning and getting anxious about not having clear goals. Often those are isolated, small issues that are fixable, but if not addressed they can grow into big, major problems.

Let's go over some things that you, as the producer, can do to help fix this in each of these situations.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

1. Churn Without Progress

- Your game doesn't change
- 'Rat-holing' or 'bikeshedding'
- Throwing away work

17

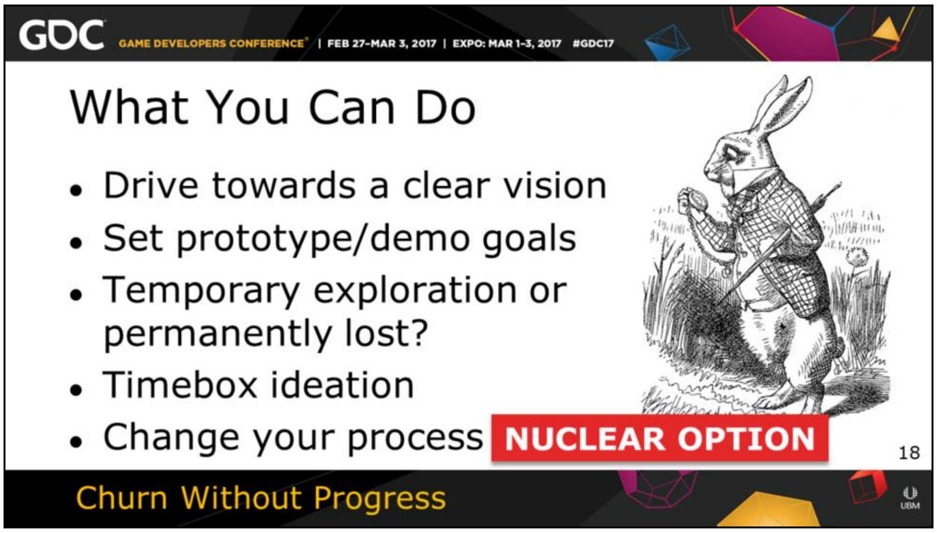
Churn Without Progress

This is probably the most damaging result of ambiguity: you're doing a bunch of *things*, but you aren't really getting anywhere, usually because your vision isn't clear enough to know if you're driving toward it. It can be hard to tell when this is happening; it can feel like you're accomplishing things even if you aren't.

Watch for these problems:
- Your game isn't changing. Maybe in your weekly playthroughs, there's nothing significant to show. People are on their phones during level reviews, and don't ask any questions. Or your milestones just keep slipping because the tasks in them just aren't getting done. Worse, nobody can tell you when they think things will get done, because they are blocked by something they can't really explain and don't know how to fix.
- Rat-holing or bikeshedding: (http://www.urbandictionary.com/define.php?term=bikeshedding) that is, deep, passionate disputes over minor, marginal issues conducted while more serious problems are being overlooked. Having epic reply-all email threads and 3-hour meetings on your font choice makes you feel like you're getting decisions made, but are you really just hiding the fact that you haven't decided which engine to use?

- And last, doing 'throwaway' work: I was a level designer on a game where the level design team was pumping out levels like clockwork: each of us were building 3 levels every 3 weeks. However: the story for the game wasn't close to done, the enemy behaviors weren't designed, and two out of every three of those levels were rejected as 'no, this won't work for what I'm thinking' by the lead designer. It looked like we were getting a lot done, but really we were spinning our wheels. We were doing good work, but it wasn't going into a shippable game, so it was worthless.

People don't get into game development because they're lazy. They get into game development because they want to build cool stuff. They're going to do their best to do that, but if they're not putting that work toward a clear vision, all that energy and creativity will be wasted.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Drive towards a clear vision
- Set prototype/demo goals
- Temporary exploration or permanently lost?
- Timebox ideation
- Change your process **NUCLEAR OPTION**

18

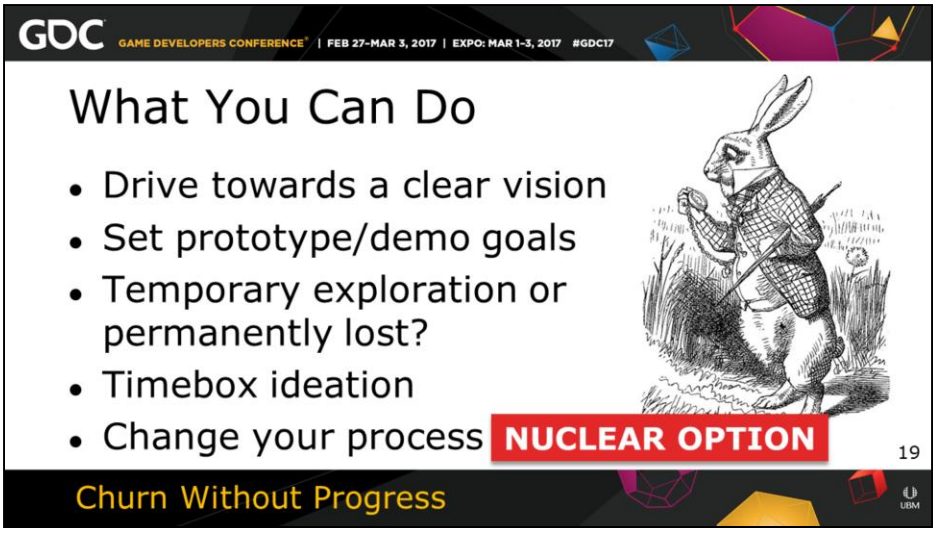
Churn Without Progress

So what can you, the producer, do to fix that situation?

First, remember that the one thing that is most likely to help your team make a successful game is to have a clear, shared vision of what your game is. If you don't have that yet, then every single thing you do should drive toward it. It needs to be your top priority; don't get distracted by the small stuff.

If you do have a clear vision for your game, or are moving towards one, but your team still feels like it's churning, set short-term prototype or demo goals that help you develop that vision. This is what vertical slices are good for: build out a piece of your game vision, play it, and see what you think. How hard was it to build? What unexpected problems did you run into? What is the best part of your demo; what is the thing your team gets excited about when they play it? Focus on developing those aspects of your vision; they're the awesome things that players will fall in love with. This also gets your team in the habit of finishing things, which is helpful.

Is your team exploring cool innovative ideas during pre-production, or are they lost in the woods? The difference is: when you're exploring, you have a goal that you're moving towards, and you know you're moving towards it. When you're lost, you don't. Is this a temporary situation, i.e. trying different things during a preproduction or exploration milestone, or a permanent doldrums that you don't see an end to? Know which of these is happening so you can either: tell your team that everything is ok, this is an exploration phase and we're working towards a specific goal, or: make sure your team leads know that you're stuck and need to get unstuck.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Drive towards a clear vision
- Set prototype/demo goals
- Temporary exploration or permanently lost?
- Timebox ideation
- Change your process **NUCLEAR OPTION**

19

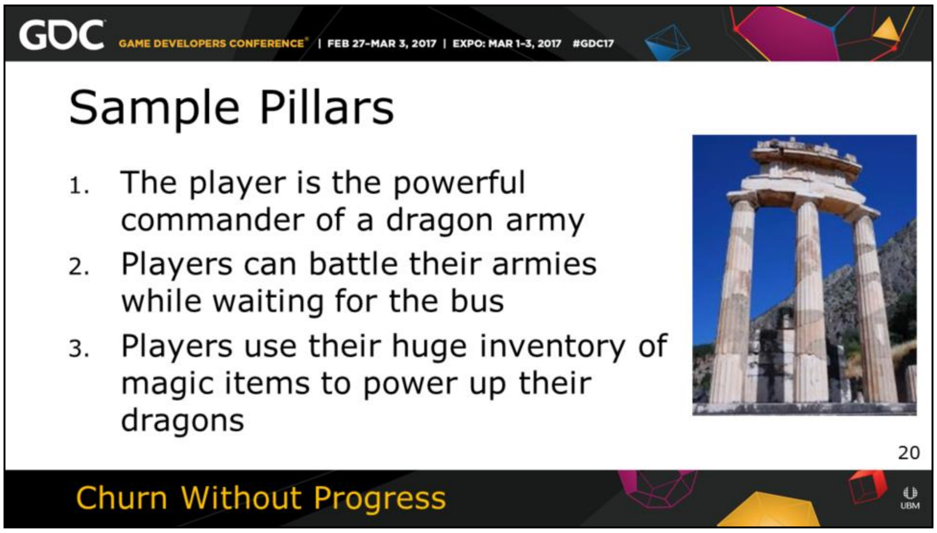
Churn Without Progress

UBM

If you’re churning but it looks like it’s temporary and you might still be making progress, set a time limit: either a hard exit date agreed-on by your team for when you will absolutely have a decision made, or a time limit just for you: if we haven’t decided on whether to add multiplayer by the beginning of May, I’m going to start worrying and talking to the team leads because it’ll start being a problem. Until then, I’ll back off and let them brainstorm. But remember: execution always takes longer than you think, especially if you’re doing something new. So don’t wait too long.

If your team is so stuck that they don’t know how to make progress, you can try changing up how you’re managing the project. Scrum is a set of project management processes that was invented for this exact situation, and while I don’t personally like Scrum on most projects, it can help you get moving again if you’re really deadlocked. Or if you’re using Scrum, try making a waterfall schedule. If you’re having weekly team meetings, try changing to daily design reviews. Maybe you need to step back, reassess, and change your focus, team structure, or something else.

But the most important thing is having a clear vision of what your game is going to be. Here’s a useful tool for developing and protecting that vision.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Sample Pillars

1. The player is the powerful commander of a dragon army
2. Players can battle their armies while waiting for the bus
3. Players use their huge inventory of magic items to power up their dragons

20

Churn Without Progress

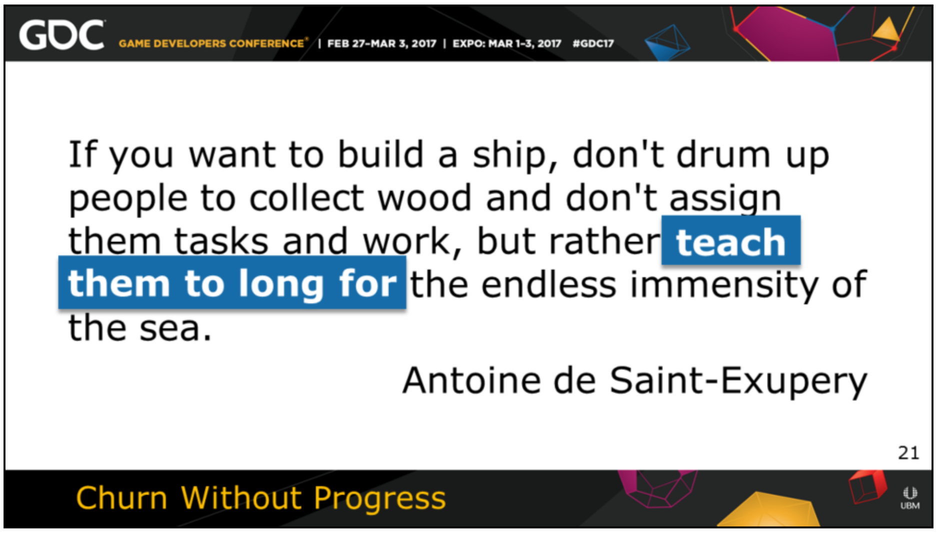
UBM

Pillars are a method for creating the vision for your game, and communicating it to your team. Choose 3-5 core features or aspects of your game that really explain what it is: your design document boiled down to 3-5 bullet points.

These are pillars I made up, so I'm sure your team's will be better, but: these are specific, aspirational, actionable, and positive. Your team members could look at a list like this and know pretty well what they should be working on. Are you working on the UI for giving orders to your dragons? Awesome, that's really important. Are you adding customizable dragon houses? Probably not as important. We should start working on our multiplayer and cloud save system immediately, because that's something only one person on our team has experience with, and he says it's hard. Etc. Use your pillars to drive decisions on more minor features; good pillars will help your team know if they're building the right thing without always having to check with their lead.

Pillars are like a ruler you can use to check that the work your team is doing is the right work. They're also like an advertisement to your team for the game you're making. They should be features you're excited about making and playing, and should make sense to everyone on your team. They're the things that you think players will fall in love with, tell their friends about, and keep coming back for.

I talk about pillars and game vision all the time: I really can't shut up about them. Here's why I think this is so important:


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

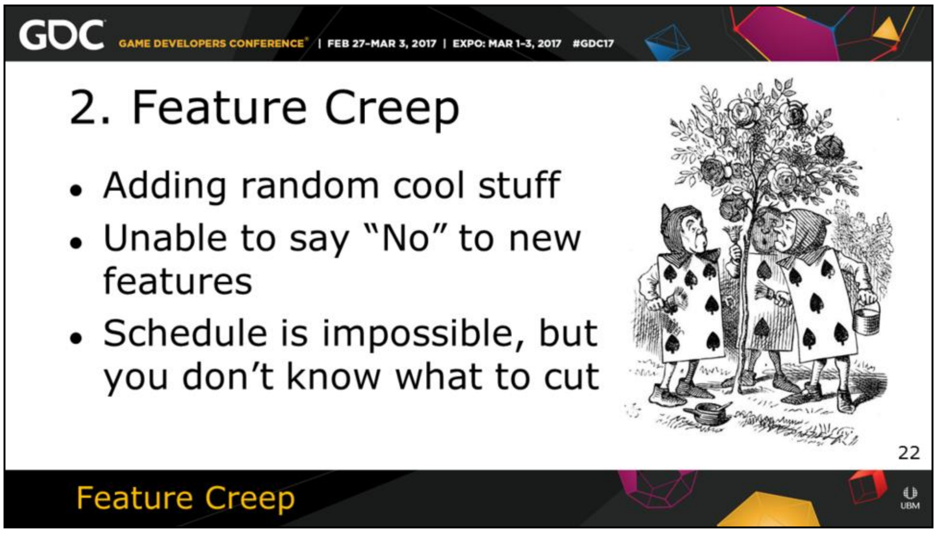
If you want to build a ship, don't drum up people to collect wood and don't assign them tasks and work, but rather **teach them to long for** the endless immensity of the sea.

Antoine de Saint-Exupery

21

Churn Without Progress

I bet that most of the people on your team didn't get into game development because they like being told what to do. They got into game development because they long to make amazing games. Make sure you have a vision for a game that your team will long to make, and that they can believe in and be proud of and pour their creative energy into.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

2. Feature Creep

- Adding random cool stuff
- Unable to say "No" to new features
- Schedule is impossible, but you don't know what to cut

22

Feature Creep

UBM

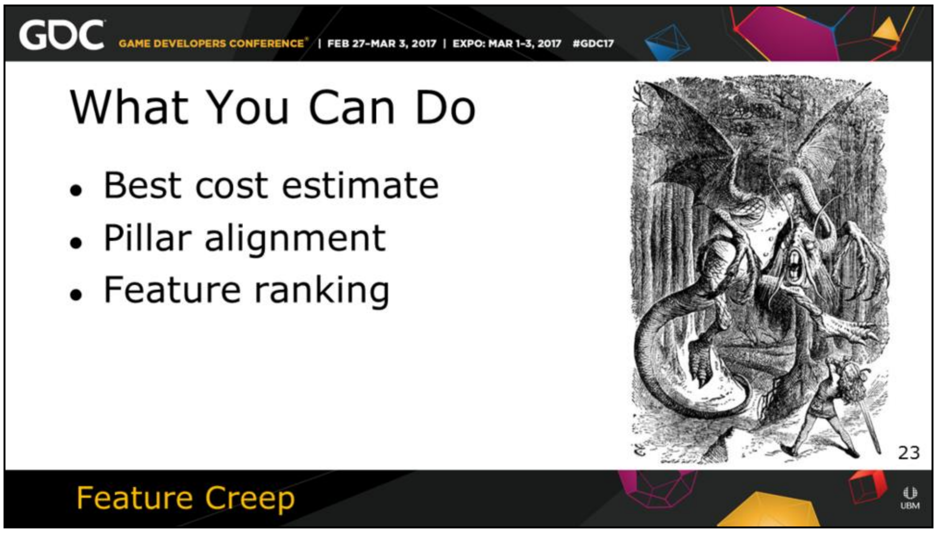
Because if you don't, you'll end up with feature creep.

You aren't sure what your game is, so you add more stuff. You aren't confident in your vision, or you really don't know whether your game will be successful, so your impulse is to add more features in the hope that something awesome will happen (it won't).

Or, your creative director gets super duper excited about every new feature she comes across, and it's not clear who's allowed to tell her no (or when you should start telling her no). Or what you should say no to.

Or, you're midway through production. Your schedule is clearly impossible, so you know you have to cut features from your game, but you don't know which ones will have the biggest payoffs or the biggest risks.

I worked on a really big AAA game that actually had a clear, awesome, and exciting vision, but: it was a big team that hadn't worked together before, so people didn't really know who to listen to. Leadership was fractured, the cost of failure was high, and the publisher was really invested in the game, so there were a lot of conflicting priorities. Tons and tons of expensive features were added to the game that it didn't really need, and that weren't fun or supportive of the vision. Lots of energy and time was wasted making those features ready for demos to the stakeholders who wanted them. And the features core to the vision of the game kept getting shortchanged in favor of all this extra stuff that nobody really wanted. Tension, infighting, and frustration were rampant.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Best cost estimate
- Pillar alignment
- Feature ranking

23

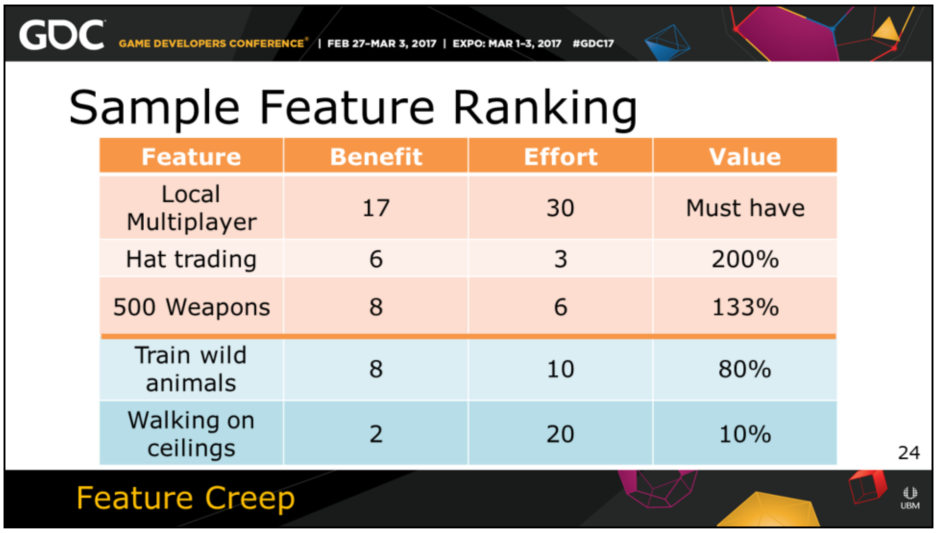
Feature Creep

UBM

Since ambiguity is when you don’t have enough information to make good decisions, you can reduce ambiguity by gathering information. You need data that you can rely on- it’s critical when deciding which features to work on and which to cut. Sit down with your teams and estimate how much of your limited resources each feature will cost. It doesn’t have to be super-precise, and your estimates won’t be perfect, but even if it gives you a little peek inside that dice cup, you’ll be in a much better place to make decisions.

Honestly evaluate each feature against your game’s core vision. If your pillars or design doc do a good job of specifying what your game should be, anything outside them isn’t necessary to make the game you want to make, and should be deprioritized. Rank your features based on how necessary they are to support your game’s design. Remember that if your constraints are tight, you can’t afford to work on anything that doesn’t support your game vision.

Work with your team to rank your features based on these two aspects. Be brutally honest about what you can afford to do and how much it costs.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Sample Feature Ranking

Feature	Benefit	Effort	Value
Local Multiplayer	17	30	Must have
Hat trading	6	3	200%
500 Weapons	8	6	133%
Train wild animals	8	10	80%
Walking on ceilings	2	20	10%

24

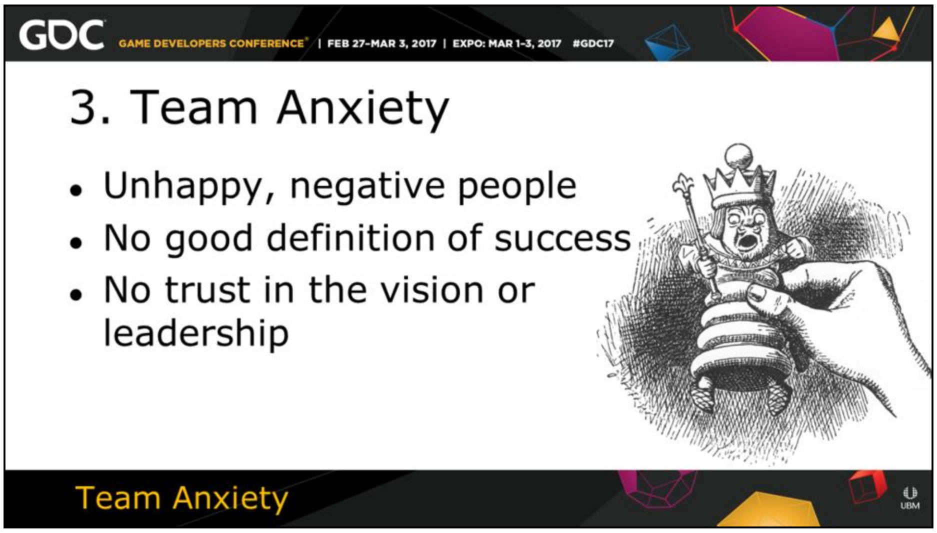
Feature Creep

UBM

Here’s an example of a feature list ranked by benefit to the game, effort, and a calculation of value from those two numbers. Whether you use this method to start discussions of what you should work on first, or if you use it to actually generate your priority list, it’s always a good idea to make sure your team agrees on the value of your planned work.

If you’re having a big problem with ambiguity on your project, it might be because your team doesn’t have a good understanding of feature value, or they don’t agree on how much features add to your game or how much they cost. You need to have some kind of reliable process for figuring that out.

[See also: http://www.gdcvault.com/play/1017938/From-Great-Ideas-to-Game which includes a section on feature ranking for The Witcher 3, and my 2015 talk http://www.gdcvault.com/YKpWZ/play/1022221/Producer-Bootcamp-How-to-Prioritize]



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

3. Team Anxiety

- Unhappy, negative people
- No good definition of success
- No trust in the vision or leadership

Team Anxiety

This is when your team just gets overwhelmed by the amount of ambiguity. They are unhappy and negative and don't see anyway to fix it, so they start grumbling, or they just give up.

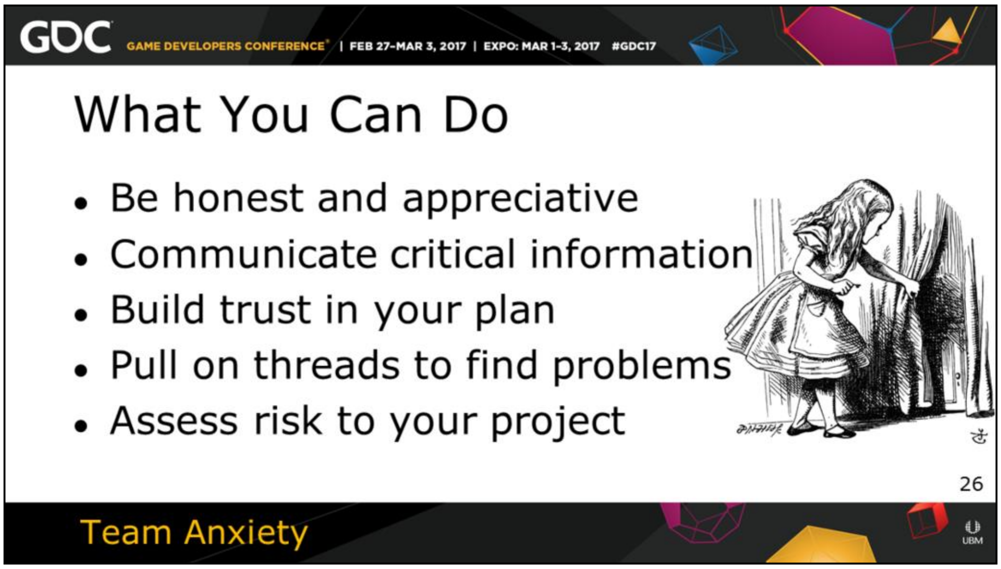
Maybe they don't know what they're expected to do, or what success looks like for them. This can either be because your game vision isn't clear to them, or because they don't see how their work fits into it. Game devs want to do good work, and if they feel like they can't, they won't be happy.

Or maybe they just don't believe in your vision. Maybe you're working on a game in a genre they don't respect or like, or maybe they just don't think the game you're making will be good.

Or maybe you or the leads on your team don't have credibility, so that when you tell them the vision or the plan or the risks, they just don't believe you.

One of the worst instances of this I ever saw was on a project with an enormous amount of pressure on the design team and tons of ambiguity. Major, core changes kept coming down from above, and the designers were working crazy hours implementing those changes. Tempers and stress were high. There were shouting matches over who wasn't doing their job. And in the middle of all that, a team meeting was held during which a leader at the studio said that if teams are working long hours, it's their own fault for not planning better. To be honest, this made all of us lose a *lot* of trust in the studio leadership. Now, not only did we have no idea what was going to happen to us, but we now felt that the leadership didn't know what was happening, which made it much worse.

This happened over a decade ago, and you can probably tell I'm still upset about it. Don't do this to your team.


GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

What You Can Do

- Be honest and appreciative
- Communicate critical information
- Build trust in your plan
- Pull on threads to find problems
- Assess risk to your project

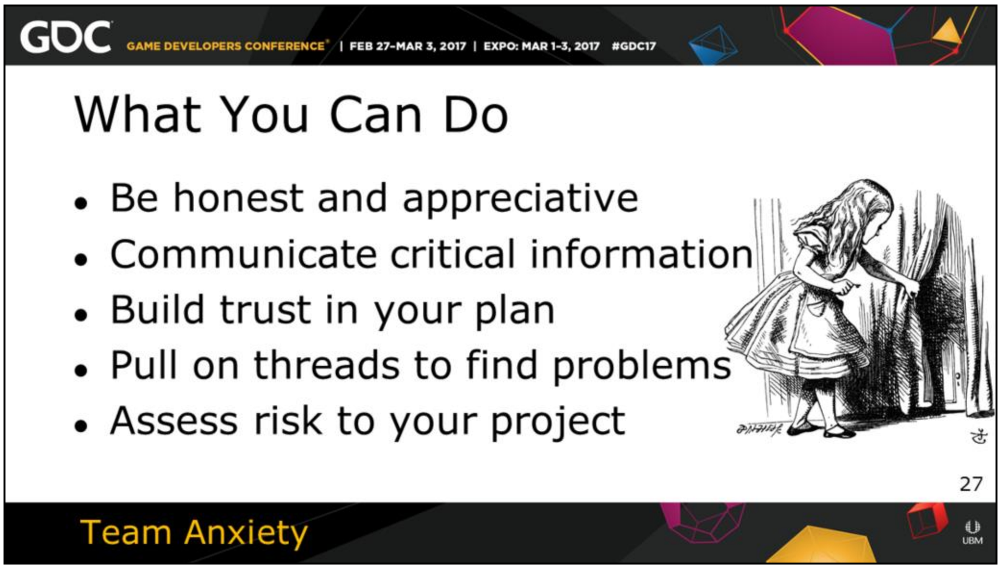
26

Team Anxiety

What can you do to soothe your team, help them be more confident and happy, and ultimately do better work?

First, be honest about what's happening, be honest about the problems, and be appreciative. If someone in charge had stood up at that meeting and said, "look, we know you're working long hours to implement the recent changes, and we know things are changing really fast and it's hard to keep up. We're trying to fix that and trying to have the design of those systems locked by the end of the month," I think the situation would have been a lot better. Also be honest about when your team is going above and beyond: point out their hard work and successes and tell them how much it's appreciated. And make sure it *is* appreciated: if your leadership doesn't respect or acknowledge the hard work of your team, remind them how important that is and point out what your team is doing to make the game great.

Communicate critical information to your team: don't let them be the last to know about a major decision or change that impacts their work. I know that sometimes you'll need to keep sensitive information under wraps or protect your team from politics and frustrations, but if it's something that your team needs to know to do good work, make sure they know it.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Be honest and appreciative
- Communicate critical information
- Build trust in your plan
- Pull on threads to find problems
- Assess risk to your project

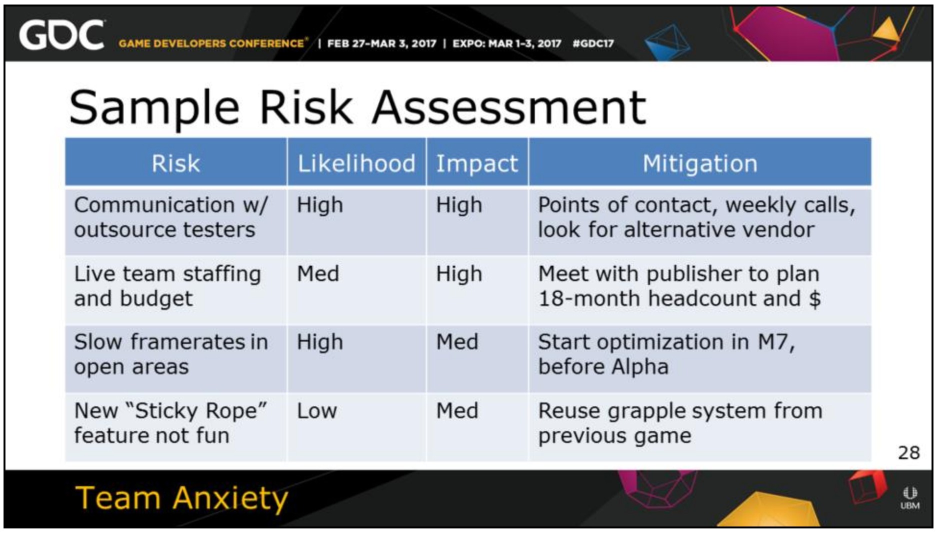
27

Team Anxiety

Build trust in your plan by doing your producer job of making sure the right work gets done. If your plan doesn't let you do that, it's a bad plan and your team is right not to trust it, and you should fix the plan. But if your plan does help you make sure the right work gets done, your team will see that and start trusting it.

Keep your ears open for problems and pull on threads until you find out what's wrong. Most of us are on teams full of decent, helpful people who don't want to complain or get someone else in trouble, and that means that often if someone isn't doing their job, or if a feature really isn't working, they don't want to be the ones to say anything. Nice people can be really good at hiding problems, and if you're working on a high-ambiguity project, often people don't want to add more problems to the pile by speaking out. It's important to talk to your team to ferret out bad news, and if someone is unhappy to figure out if there's a deeper problem that you, the producer, needs to figure out how to fix.

And last, work with your team to really assess the risks to your project. Every project has risks, and sitting down and talking them out, and planning what you would do if they happened, is a really valuable exercise. If you don't have a plan B, you don't really have a plan, and on high-ambiguity projects you might need a plan C, D, and E as well.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Sample Risk Assessment

Risk	Likelihood	Impact	Mitigation
Communication w/ outsource testers	High	High	Points of contact, weekly calls, look for alternative vendor
Live team staffing and budget	Med	High	Meet with publisher to plan 18-month headcount and $
Slow framerates in open areas	High	Med	Start optimization in M7, before Alpha
New "Sticky Rope" feature not fun	Low	Med	Reuse grapple system from previous game

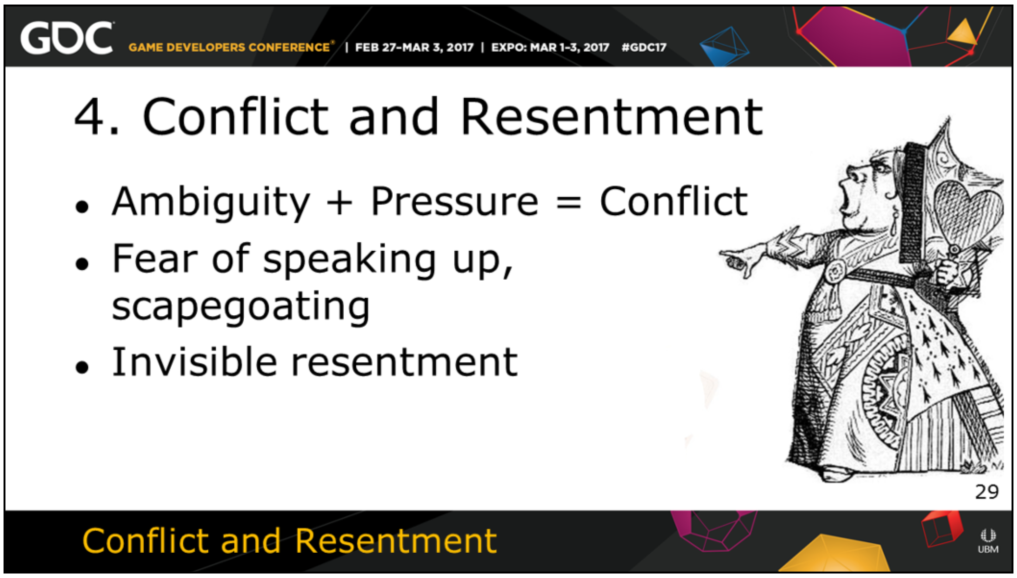
28

Team Anxiety

A good way to do this is to hold a risk assessment. Sit down with your team, or your leads, and ask them what they're afraid will happen; what might happen that could really hurt your game? What keeps them up at night?

For each risk, talk about how likely it is to happen, and how bad it would be if it did. Then come up with a plan to either prevent the problem, or something to fall back to if it happens. Not only does this help your team feel a little bit more in control of their situation and like there's less ambiguity, but it also actually gives you more control over your situation and reduces ambiguity (because doing this exercise will give you more information that can help you make better decisions).

Many publishers require an updated risk assessment delivered at every milestone, which I think is a fantastic idea.


GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

4. Conflict and Resentment

- Ambiguity + Pressure = Conflict
- Fear of speaking up, scapegoating
- Invisible resentment

29

Conflict and Resentment

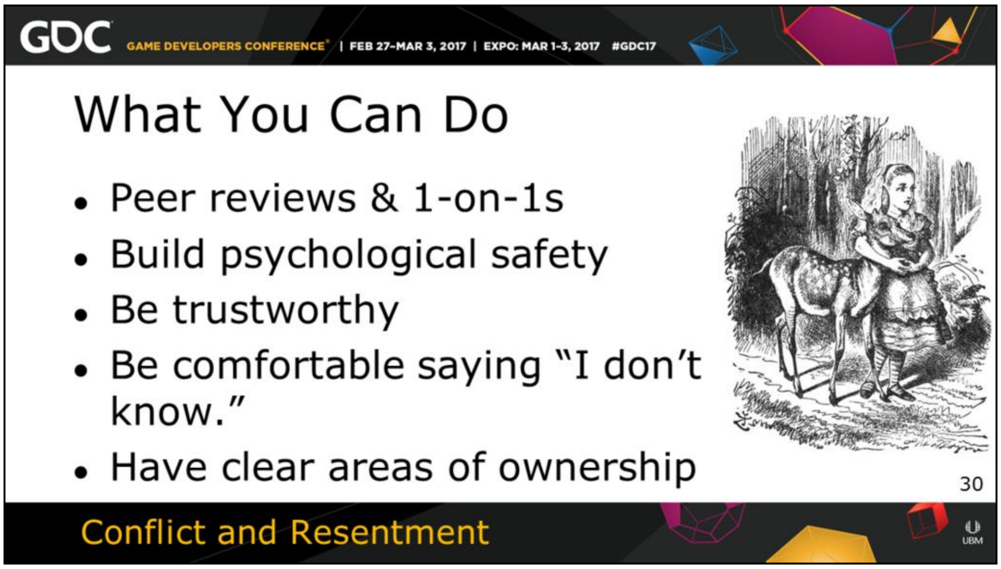
Nice people are *usually* good at hiding problems, but sometimes those problems get so bad that everything just blows up. This can cause everything from actual shouting matches to long-term simmering resentment.

Usually this is because of the fatal combination of high ambiguity and tons of pressure. You know you're doing your best, so all of this ambiguity must be someone else's fault.

People are afraid to bring up ideas or ask questions in meetings, or to raise concerns. They don't feel like they'll be taken seriously, or they're worried that they'll be seen as 'the problem'. Maybe that does happen happening, and someone gets thrown under the bus as the scapegoat because the real problems are too big to address.

These problems aren't always visible: maybe there's resentment, fear, and bad feelings boiling under the surface, ready to erupt at any moment, but none of your team leads really know it's there.

A friend of mine was a producer on a team that was under a ton of pressure to ship a launch title, and the publisher's priorities kept changing wildly. They worked incredibly hard and managed to ship, and their game was well-received. Everything seemed fine. But then the studio held peer reviews, and all these huge interpersonal and inter-team problems showed up. The studio leadership was shocked; they had no idea. They had been hyper-focused on shipping and hadn't had time to notice that their team was falling apart.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Peer reviews & 1-on-1s
- Build psychological safety
- Be trustworthy
- Be comfortable saying "I don't know."
- Have clear areas of ownership

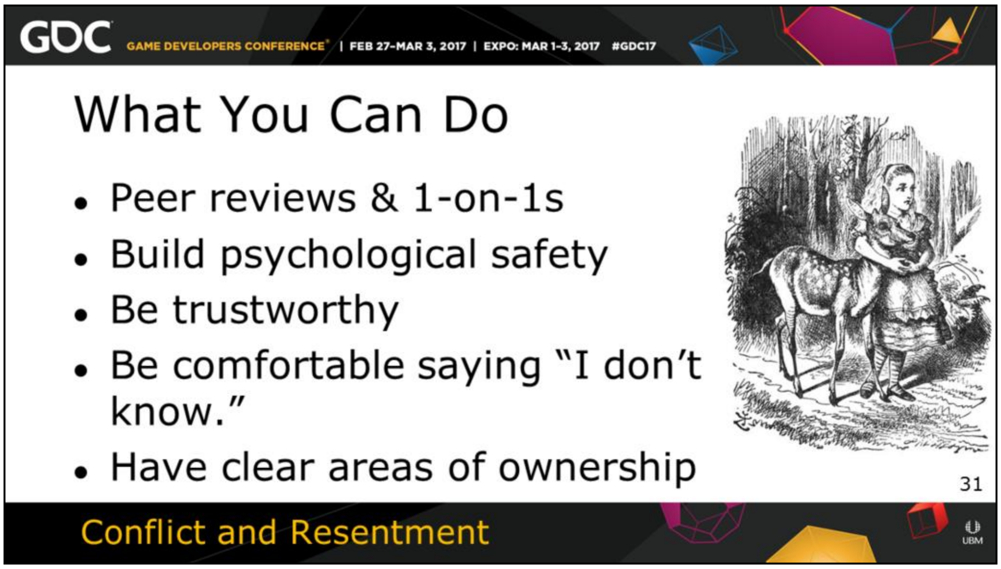
30

Conflict and Resentment

UBM

So what can you do about this? Peer reviews (that is, have everyone on your team pick 3-5 coworkers and write a short assessment of how they're doing, or if there are any issues) are one way to surface problems, and you should do them if you can, but it's not necessarily the best way. They take time, and if there are a lot of interpersonal problems on your team people may not feel comfortable being honest; either because they fear retaliation or because they just don't want to hurt someone's feelings. Make sure there are other channels to get information: anonymous reporting isn't great, but it's better than nothing if nobody will talk. Make sure managers are holding one-on-ones with their team members, and even if you have a small team with light management, make sure that everyone gets a regular one-on-one with someone they trust so they can discuss problems and solutions, and get feedback. This will help surface problems early when they're easier to fix.

If your team members aren't comfortable speaking out, you need to build psychological safety on your team. Look up Project Aristotle (I'll link to it in the notes): Google did a big investigation into what makes a team perform well or badly, and found that the biggest factor was what they called 'psychological safety', or when your team members feel comfortable and safe expressing their feelings and ideas, and where everyone feels heard and respected. Make sure people aren't being dismissed or talked over in meetings, and call it out when it happens. Ask people who don't usually speak up for their input, and really listen to and respect what they have to say.
[https://www.nytimes.com/2016/02/28/magazine/what-google-learned-from-its-quest-to-build-the-perfect-team.html?_r=0]



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Peer reviews & 1-on-1s
- Build psychological safety
- Be trustworthy
- Be comfortable saying "I don't know."
- Have clear areas of ownership

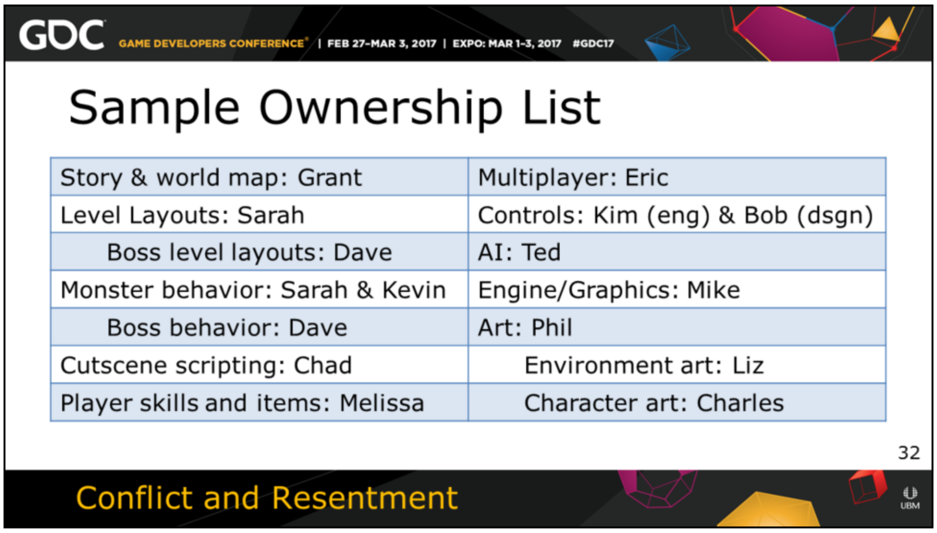
31

Conflict and Resentment

Make sure your team can trust you, personally. Especially if things are really ambiguous, make sure everyone knows what your motives are and what they can expect from you. Do what you say you're going to do, when you say you're going to do it (this is a lot rarer than you might think). [http://www.askamanager.org/2011/03/do-what-you-say-youre-going-to-do.html] And make sure that the information you're giving your team matches what they're hearing from other sources, and if it doesn't, make sure you know why not.

If you don't know something, say "I don't know", and then go find out. Not knowing things is also part of your job: but you have to know what information you're missing and how to get it. Pretending you know something when you don't will eat away at your credibility, and you need credibility to solve these kinds of problems.

Lastly, make sure ownership areas of the project are clear. You can't make sure the right work is getting done if you don't know who's ultimately responsible for doing it. You should know who owns every single feature or level or character or integration on your project, and you should be able to write it down.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Sample Ownership List

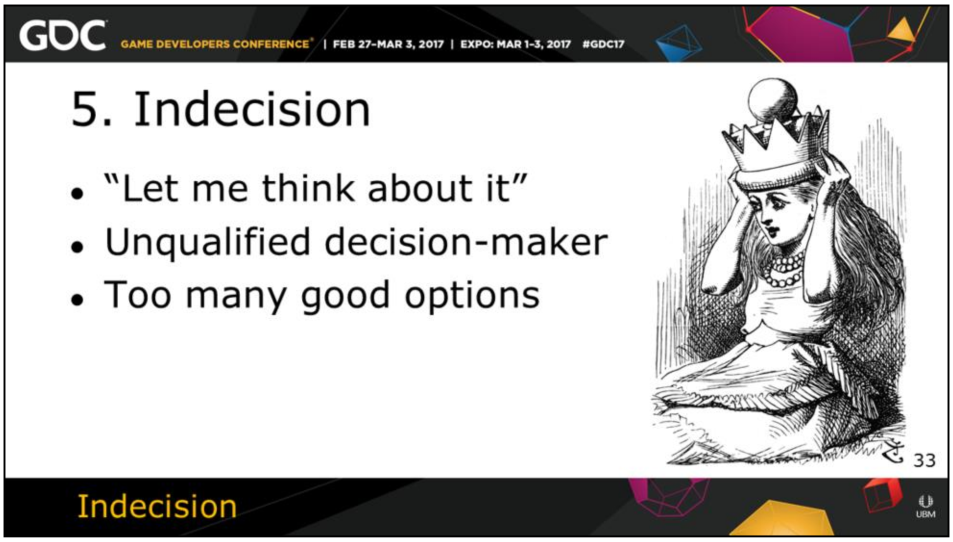
Story & world map: Grant	Multiplayer: Eric
Level Layouts: Sarah	Controls: Kim (eng) & Bob (dsgn)
Boss level layouts: Dave	AI: Ted
Monster behavior: Sarah & Kevin	Engine/Graphics: Mike
Boss behavior: Dave	Art: Phil
Cutscene scripting: Chad	Environment art: Liz
Player skills and items: Melissa	Character art: Charles

32

Conflict and Resentment

Here’s an example. These can be as detailed or high-level as you need. Remember that if someone owns something, they need to have authority over it. They get to make decisions about it, and decide how it’s implemented. If they’re making bad decisions, fix that problem, but don’t say someone owns something and then not actually give them any control over it.

If you make this ownership list and it has one name on it over and over, either you’re the only person on your team, or you have a problem. There’s no reason to hire smart, motivated people if you don’t give them ownership over their work: people who just want to follow directions, punch the clock, and go home are a lot cheaper, so if you aren’t going to let your team members make decisions you might as well save your money.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

5. Indecision

- "Let me think about it"
- Unqualified decision-maker
- Too many good options

33

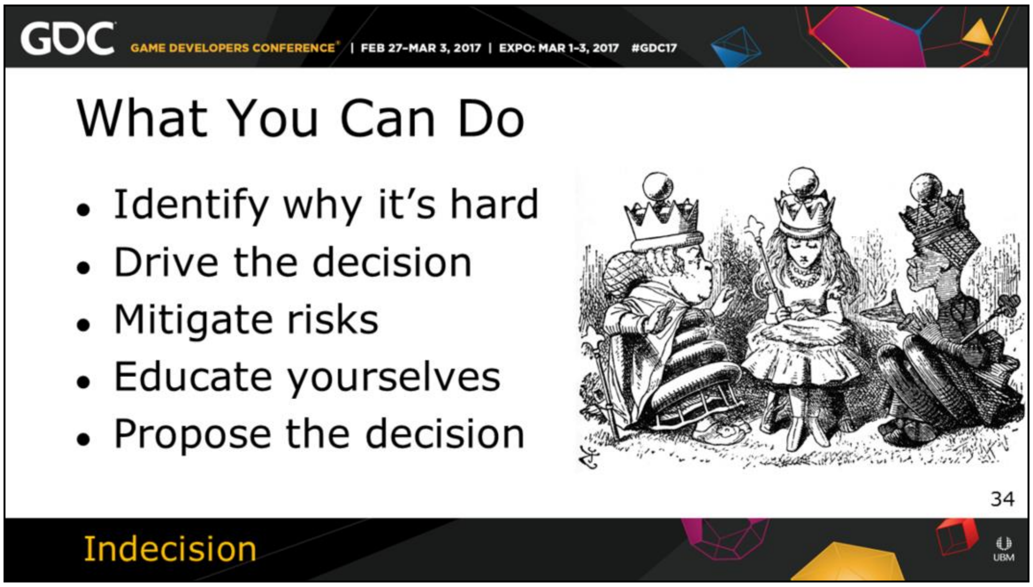
Indecision

Ambiguity is when you don't have enough information to make decisions, so naturally when you have ambiguity people will be reluctant to make decisions. But since you have to make all of your decisions before you can ship your game, indecision will block you from shipping.

The problem I see the most is unspecified delay. They want more time to think before making the decision, but they won't tell you how much time because they don't really know. They might be waiting for inspiration, or for something to happen, but they don't really know what it is, or they might be afraid of the consequences of their decision. If we try to finish this buggy feature, it will add tons of risk, but we promised it in the Kickstarter so if we cut it the backers will feel betrayed. If you're being ambitious at all, you'll face these kinds of decisions: especially if you're working within tight budget, schedule, or feature constraints. Maybe they're waiting until they feel like they have enough information, but they just never get to that point.

Maybe the person making the decision isn't really qualified to do so. Remember your list of areas and owners: Is your art director actually a competent artist? Is your lead designer familiar with the genre you're working in? Is your marketing guy trying to make engineering decisions because he was a programmer 15 years ago?

If you're lucky, it's hard to make decisions because you just have too many good options. Maybe your design vision has 8 pillars and all of those pillars are awesome. Your schedule makes it obvious that you have to cut 4 of them, but they're all so great that you just can't bring yourselves to do it. That's not the worst situation to be in, but ultimately it'll keep you from shipping.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

What You Can Do

- Identify why it's hard
- Drive the decision
- Mitigate risks
- Educate yourselves
- Propose the decision

34

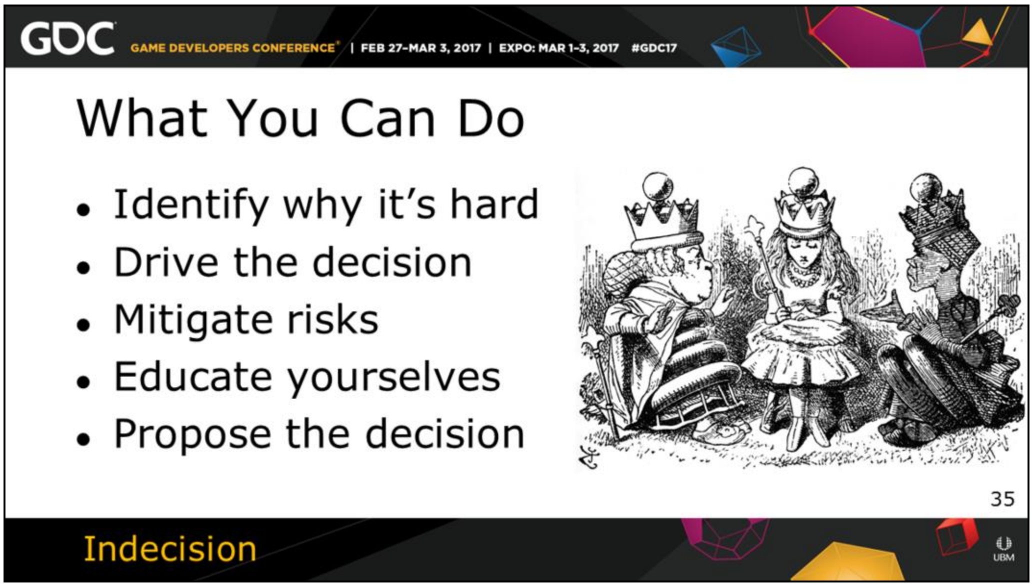
Indecision

UBM

I don't have to deal with this very much right now, because even though I'm on a project with tons of ambiguity, the project director has this philosophy: "All decisions are easy. If it's an easy decision, then you just make it. If it's a hard decision, that means that both options are equally good, so you just pick one." Not only does this make it much easier for me to do my job well, but I also really like it as a yardstick: Why does this decision not fall into one of those two categories? Is it because all options are good? Or because all options are bad? Or because you don't have enough information?

Is it because they really, really want to pick both, and can't let one go? Then you need to clarify why you can't do both, show them the data on why the decision needs to be made, and drive them to make the decision. Be vocal about how much is depending on this decision, and the consequences of delay. The more critical the decision is, the more energy you need to put into getting it made.

Is it because all possible options are going to really hurt the game or your team? Then use your risk mitigation plan from earlier and figure out if there's a way to minimize the negative fallout. Or figure out if there's a way to assign numbers to the outcomes: figure out, as well as you can, what you're risking with each option. Is there a way to find more information before you decide? If not, it's better to decide now. Often it's better to make a decision sooner rather than putting it off, even if it's the wrong decision.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

What You Can Do

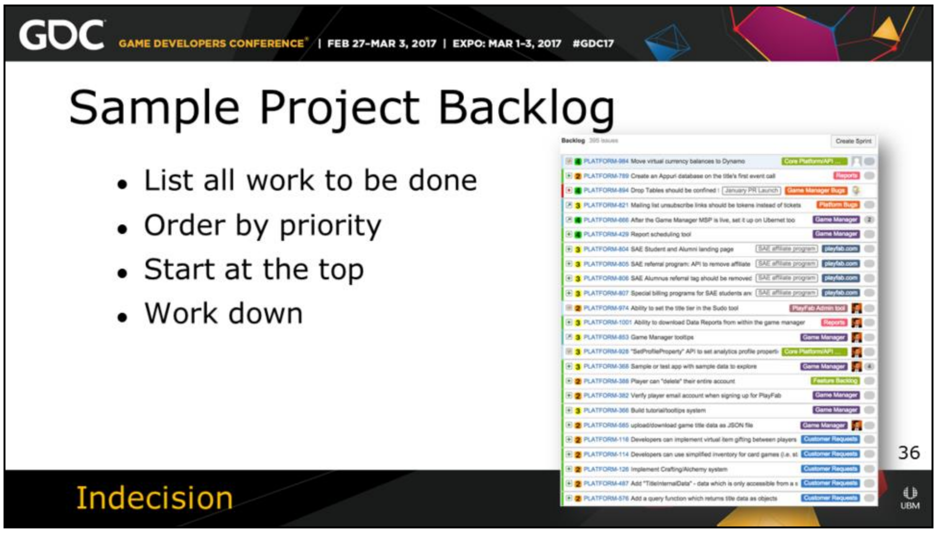
- Identify why it's hard
- Drive the decision
- Mitigate risks
- Educate yourselves
- Propose the decision

35

Indecision

Or is it hard because the person making the decision doesn't have enough expertise or information to make it? In that case, you may need to change who owns this decision, or you might be able to help the current owner become an expert. Tools like playtests, competitive analysis, and customer feedback can be a huge help. If your lead designer has only made hardcore strategy games and your team is making a kid's puzzle game, schedule team playtests of a bunch of kid's puzzle games, and bring in kids to play them with him. I was on a project where we were working in a genre that we were all unfamiliar with, so we scheduled playtests of similar games every week so we could really learn what makes those games good and what makes them bad.

Say out loud what you think the best decision is, and see if you can get the decision-maker to agree or disagree. Sometimes a decision-maker just needs to hear the options. When deciding on features, sometimes it helps to propose a ranking and let them adjust it. Make sure you don't fall into the trap of making these decisions yourself if you're not the feature owner; sometimes producers will jump in and try to make decisions because everyone else is too reluctant to, but that's a temporary fix and I've never seen it turn out well. You as the producer should get decisions made, not make them yourself.



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Sample Project Backlog

- List all work to be done
- Order by priority
- Start at the top
- Work down

Backlog 395 issues

Create Sprint

4	PLATFORM-984	Move virtual currency balances to Dynamo		Core Platform/API ...
2	PLATFORM-789	Create an Appuri database on the title's first event call		Reports
4	PLATFORM-894	Drop Tables should be confined !	January PR Launch	Game Manager Bugs
3	PLATFORM-821	Mailing list unsubscribe links should be tokens instead of tickets		Platform Bugs
4	PLATFORM-666	After the Game Manager MSP is live, set it up on Ubernet too		Game Manager
4	PLATFORM-429	Report scheduling tool		Game Manager
3	PLATFORM-804	SAE Student and Alumni landing page	SAE affiliate program	playfab.com
3	PLATFORM-805	SAE referral program: API to remove affiliate	SAE affiliate program	playfab.com
3	PLATFORM-806	SAE Alumnus referral tag should be removed	SAE affiliate program	playfab.com
3	PLATFORM-807	Special billing programs for SAE students and	SAE affiliate program	playfab.com
2	PLATFORM-974	Ability to set the title tier in the Sudo tool		PlayFab Admin tool
3	PLATFORM-1001	Ability to download Data Reports from within the game manager		Reports
3	PLATFORM-853	Game Manager tooltips		Game Manager
3	PLATFORM-928	"SetProfileProperty" API to set analytics profile properti		Core Platform/API ...
3	PLATFORM-368	Sample or test app with sample data to explore		Game Manager
2	PLATFORM-388	Player can "delete" their entire account		Feature Backlog
2	PLATFORM-382	Verify player email account when signing up for PlayFab		Game Manager
3	PLATFORM-366	Build tutorial/tooltips system		Game Manager
2	PLATFORM-565	upload/download game title data as JSON file		Game Manager
2	PLATFORM-116	Developers can implement virtual item gifting between players		Customer Requests
2	PLATFORM-114	Developers can use simplified inventory for card games (i.e. st		Customer Requests
2	PLATFORM-126	Implement Crafting/Alchemy system		Customer Requests
2	PLATFORM-487	Add "TitleInternalData" - data which is only accessible from a s		Customer Requests
2	PLATFORM-576	Add a query function which returns title data as objects		Customer Requests

36

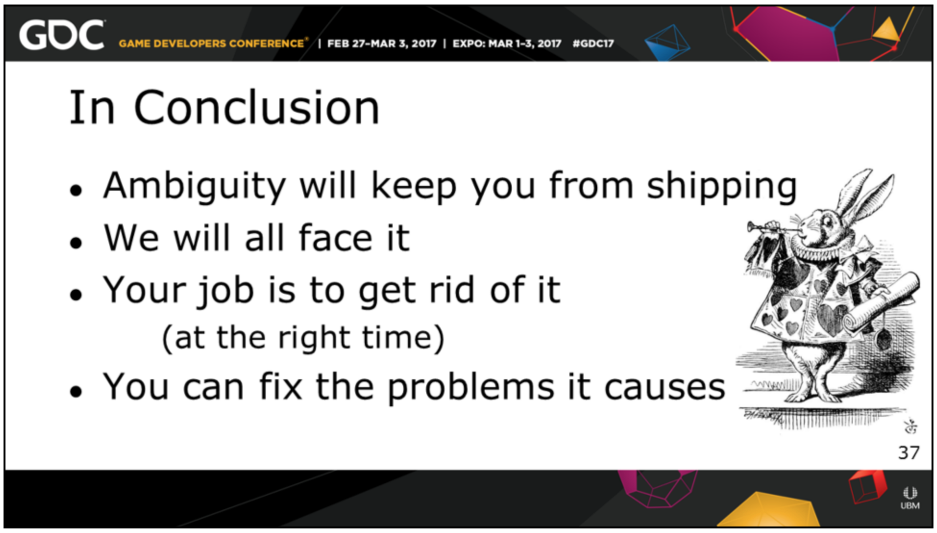
Indecision

UBM

One way to propose a decision is with a Scrum-style Backlog.

The backlog pictured here is from a project I was on that definitely had the "too many good options" problem. It was a live product, and there were soooo many features we wanted to add! And they were all great and they all might make us successful. One of my most valuable contributions to that project was keeping a prioritized backlog in Jira; the feature owners could change the priorities, but the features had to *be* prioritized and we used that prioritization to decide what we'd work on for our next release.

If your team isn't making important decisions, try starting with a backlog: if nothing else, it'll force you to confront the sheer quantity of stuff you want to do, and it'll force you to put some things at the top and others at the bottom.


GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

In Conclusion

- Ambiguity will keep you from shipping
- We will all face it
- Your job is to get rid of it
  (at the right time)
- You can fix the problems it causes

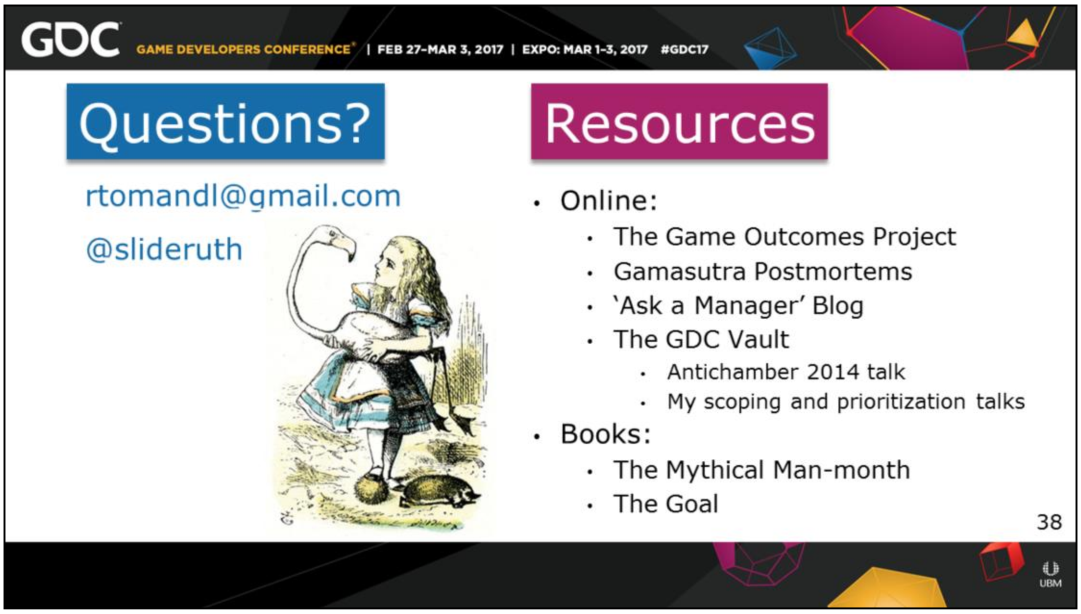
37

In conclusion:
To ship your game, you have to recognize and reduce ambiguity. You can't ship until you've made all of the decisions and reduced ambiguity to zero.

Ambiguity is something that negatively affects all of us, and the producers who can deal with it effectively will be more valuable.

Your job is to make sure you have a plan that drives down ambiguity at the right time, and lets you successfully ship a great game.

Ambiguity can cause a bunch of problems for your game, but you can fix them!


GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

Questions?

rtomandl@gmail.com

@slideruth

Resources

- Online:
  - The Game Outcomes Project
  - Gamasutra Postmortems
  - 'Ask a Manager' Blog
  - The GDC Vault
    - Antichamber 2014 talk
    - My scoping and prioritization talks
- Books:
  - The Mythical Man-month
  - The Goal

38

The Game Outcomes Project:
http://intelligenceengine.blogspot.com/2014/12/the-game-outcomes-project-part-1-best.html

Gamasutra Postmortems:
http://www.gamasutra.com/features/postmortem/

"Ask a Manager":
http://www.askamanager.org/

GDC Vault:
http://www.gdcvault.com/
Especially "Antichamber: An Overnight Success, Seven Years in the Making":
http://www.gdcvault.com/play/1020776/Antichamber-An-Overnight-Success-Seven
And my 2016 talk about scoping your game:
http://www.gdcvault.com/play/1023138/Producer-Bootcamp-How-Saying-No
My 2015 talk about prioritization, while we're at it:
http://www.gdcvault.com/play/1022221/Producer-Bootcamp-How-to-Prioritize

The Mythical Man-Month:
http://www.amazon.com/Mythical-Man-Month-Software-Engineering-Anniversary/dp/0201835959

The Goal:
http://www.amazon.com/Goal-Process-Ongoing-Improvement/dp/0884271951