AVENTINE SOLUTIONS

#awsvsgcp Episode 1

So you think this is a Democracy?
... Architecture Decisions

So you think this is a Democracy?

- I’ve heard this exact same question twice (to my surprise) discussing two completely different development projects with two different people having two completely different Roles
- After letting it sink in, both times were about how Architecture Decisions are made and I, in fact, had hit a wall each time
- After some research, it turns out that this is not an uncommon experience when trying to talk with Teams about Architecture Stacks and just trying to offer some personal expertise and opinions.
- If feels like emotions get involved or at least strong opinions and “black & white thinking”; it’s a bit jarring and unexpected

Wanting to Contribute

- I believe it's natural for any Software Developer who has any kind of passion for their work to want to contribute to Architecture Decisions.
- But many disagree.
- Some people feel that Software Developers and Coders do not belong in the decision making process even though it may profoundly affect their work and their future experience on a project.

AVENTINE
SOLUTIONS

The System Architect

- One approach to Architecture Design is to hire the System Architect
- They may have an advanced degree from MIT or TU Delft and they often sit on a lonely pedestal, handing their design documents down to the Developers.
- They can also come under other titles such as Database Administrator, Data Architect or Cloud Architect.
- I have run into people that believe this is part of the DevOps Role. (That is to say that DevOps dictates to Developers what the Architecture Stack is.)
- Some believe that this approach is doomed to failure because, among other things, it stifles innovation and ignores the current complexities of modern systems.

Agile Architecture

- Another approach, *Agile Architecture*, emerged around 2015, gaining traction as a distinct methodology. It's a thing, yes.
- Like other other collaborations on Agile Teams, Agile Architecture is about trusting each others' experience equally and valuing each individual perspective.
- Perhaps, as happened to me these last two times, you hit the wall where the Agile Team is not trusted to make Architecture Decisions and is therefore not trusted to define its own future.

AVENTINE
SOLUTIONS

Tools & Techniques

AVENTINE SOLUTIONS

I do not have all the answers but I can, in this episode, suggest a few tools and techniques for introducing Agile Architecture into your Agile processes and rituals.

Types of Architecture

- First, it would help if you identity which of the 4 types of Architecture your project needs work on. I will not take the time to define each of these … because, well, I think you know already:
  1. *Enterprise Architecture*
  2. *Software Architecture*
  3. *Domain Architecture*
  4. *Platform Architecture*
- This will affect the types of tools you use … for example building abstract models versus working prototypes.

AVENTINE SOLUTIONS

Technical Debt

- That uncomfortable feeling we have when the Architecture is not quite right is often caused by the accumulation of Technical Debt.
- Technical Debt causes us to code workarounds or costs extra complexity points when building new features or fixing bugs.
- This can be sometimes be solved by levelling up our Architecture.
- For instance, if the Domain Model does not fully support the current system, we might write uncomfortable workarounds to force a square peg into a round hol This also may cause strange looking unit tests that are hard to understand and justify. In turn, the Senior Developers need to take extra time to explain these to the Junior. As you can see, the cause of Technical Debt builds quickly.
- Recognising the exact Technical Debt you are trying to solve with an Architecture change is important so that the Stakeholders can understand the impact this upcoming change will have on the Team’s Velocity.

The Simplest Possible Solution

- Follow the Agile Principle of frequent and small iterations of change.
- Avoid complicated solutions where a prototype might take more that one Sprint. This approach is more likely to stall or fail.
- Think about the Architecture runway: How will existing code be affected? How will reusable components be leveraged? How much effort will be put into testing? How will the entire Tech Stack look?
- Instead, make simple, small changes that can be quickly evaluated and measured for achieving the project's goals
- Plan for the next change and repeat.

Architecture Spikes

- Once an approach is decided upon, plan for an Architecture Spike in the next Sprint.
- If Sprint Planning and communication is working, the Project Owner and Stakeholders will understand the necessity of this Spike and the trade-off of taking time and resource away from other new features.
- Code a working prototype or proof of concept if possible. Include tests if possible to evaluate how testing code will be worked into the project.
- It should be time-boxed. For example, in a 2-Week Sprint, 1 to 3 days is enough.
- When the time box is complete, report the result immediately in the next Stand-up
- If time ran out, re-evaluate and try another Spike in the next Sprint. Perhaps narrowing the scope will help.

AVENTINE SOLUTIONS

Juniors and Interns

- Consider assigning Spikes to Juniors and Interns.
- Besides being a learning experience, it’s a fantastic opportunity for knowledge transfer and gives them a feeling of high contribution to the project.
- Naturally, the Seniors will monitor their progress during Stand-ups and will need to be sensitive for misunderstandings and other ways they might get blocked.
- Seniors should always make themselves available for Pair Programming sessions in this case. Juniors and Interns should never be made feel ashamed to ask for help.

Architectural Decision Records

- When the Team has made a significant change to architecture, journal this event with an “Architectural Decision Record”
- This will record the rational for the decision around a specific change.
- Try to describe the trade-offs for making a specific choice so that future versions of your Team can understand why the Stack looks the way it does.
- I can’t tell you how many times having this historical information could have helped me when I took on changes to a running Legacy Service.
- This journal of records should be made available to Stakeholders as well as shared with Newcomer Developers to your Team

AVENTINE SOLUTIONS

Backlog Refinement

- Hone your skills to include architecture improvements and spikes during Backlog Refinement.
- Agile is sharing the knowledge, so help the rest of the Team to learn to exercise this muscle during these sessions as well.
- When working on new Features, Bug Fixes or Technical Debt, is there a smart approach which would mean improving the Tech Stack and perhaps splitting out an Architecture Spike from the ticket or story?
- There will always be trade-offs in Velocity for new Feature development. Communication with the Product Owner and Stakeholders is paramount.

Read the Book

- Doing some research will of course help with your particular situation and Team.
- If we search for “Agile Architecture” in Amazon Kindle books, we’ll find many possibilities.
- I haven’t read any of them. I’m still looking for something that will help me in my current project.
- I’ve provided a few links to get you started however.
- Please share your frustrations as well as successes with Architecture Decisions in the comments below … and I’ll see you in the next episode!

AVENTINE
SOLUTIONS

Links & References

- Start with Agile Architecture on Wikipedia
- How the Agile Mindset is Integral to Architecting Modern Systems

  an essay by Diana Montalion 20.July 2023
- Agile Architects: Classical Conductors or Band Members a presentation at the Berlin 2023 *Agile Meets Architecture* conference by Silvia Schreier
- Love Unrequited: The Story of Architecture, Agile, and How Architecture Decision Records Brought Them Together
  by Michael Keeling 20.June 2024

AVENTINE SOLUTIONS