Architecting for the ilities

Nathaniel T. Schutta
@ntschutta

Logistics.

Form a group!

Ideally people you don't work with.

You won’t need a laptop.

But pen and paper are very useful!

Bit of lecture.

Do one example together.

Then you'll pick a kata.

http://nealford.com/katas/list.html

Work with your group!

Present your findings…

Questions?

Software projects are hard.

Mix of many things.

Customers focus on functionality. Rightly so.

And as developers that's where we lived!

As architects though…

We need to look beyond the functionality.

Look at the “ilities”.

Otherwise known as the non functional requirements.

Or quality attributes.

Critical part of our job!

How do we find them?

How do we document them?

How do we balance competing requirements?

Quality Attributes.

Sometimes called non functional requirements.

Or quality goals, constraints, quality of service goals…

Cross-functional requirements.

The “ilities”!

Architecturally significant requirements.

Customers usually focus on functionality.

It's what they "see" after all.

Obviously important we meet their needs!

But we have to look beyond that.

Vital that we focus on quality attributes.

Service level objectives if you will.

What are some quality attributes?

Maintainability.

Scalability.

Reliability.

Security.

Deployability.

Simplicity.

Usability.

Compatibility.

Fault tolerance.

Modularity.

The list goes on and on!

http://en.wikipedia.org/wiki/List_of_system_quality_attributes

What quality attributes do you focus on?

Depends a lot on the type of software you build!

We don't get to turn every knob to 11 do we?

Inverse relationships.

Maximizing one may minimize another.

Security and usability

for instance.

It’s a balance.

Some are obvious to our customers.

If the systems won't support the user load…

Or we get hacked say.

Fairly easy to convince people of the importance.

Others - are “invisible”.

Or at least harder to see.

Maintainability and simplicity for instance.

How do we get the decision makers to buy in?

What techniques can we use to influence them?

Outline the benefits.

Find common ground.

Avoid aggression.

Listen.

Have a conversation!

Can be hard to convince people!

Two approaches...

Find the influencers.

Influence the influencers.

Approach as equals.

Rely on the strength of your ideas and your reputation.

Your reputation speaks for you when you aren't there.

Not sure what your rep is?

Ask.

Find common ground.

Reciprocity rules…

Be helpful.

Be respectful.

Research your ideas.

Use trusted sources.

Recruit credible allies.

Nothing wrong with bringing help!

Speak their language.

Avoid techno babble.

You may be impressed by the jargon…

Most customers aren’t.

What resonates in your organization?

Cost savings?

Developer productivity?

Speed to market?

Shape your approach accordingly.

Architectural katas.

What is a kata?

Concept comes from the martial arts.

Forms are an important part of martial arts study.

A form is just a series of steps: block, strike, kick.

Each belt typically has a specific form you master.

Students repeat a form many, many times.

The moves become second nature.

What does that have to do with software?

PragDave, Steve Yegge and others popularized.

Effortful study.

"How do we get great designers? Great designers design, of course."

— Fred Brooks

Idea extends to software architecture too.

Neal Ford, Mark Richards, Ted Neward.

Architectural Katas.

http://nealford.com/katas/index.html

Documenting non functional requirements.

Doesn't have to be complicated!

http://thinkrelevance.com/blog/2011/11/15/documenting-architecture-decisions

Simple vs. Easy…

https://www.infoq.com/presentations/Simple-Made-Easy

Consider a mind map.

Or even just a simple list.

Some are more equal than others…rank them.

Ranking is often very subjective!

Give it a shot, solicit feedback, iterate.

Rinse and repeat.

Everyone happy?

We want to connect quality attributes…

And the architectural decisions that result.

Architectural Decisions.

They happen!

How do we document them?

Title/ID.

What is the problem?

List assumptions and constraints.

What are the options?

List the pros and cons of each alternative.

Which one did you chose?

Why?

Consider a “time capsule”.

Screen cast or podcast of what you did and why.

Prevent Monday morning quarterbacking…

Or just "why did we do this?"

Quality attributes impact architectural decisions.

Questions?

The iDon’tDrive System.

Self driving cars are all the rage these days.

Let's pretend our company has the answer!

VCs can't give us money fast enough.

Customers are flocking to our stylish website.

Our team has been tasked to build a backend system.

The cars generate a lot of data.

Battery level, health of the engine, maintenance.

Basics

- Car "phones home" on a regular basis.
- Sends a standard data payload including VIN.
- Demand is extremely high.
- Expect millions of cars on the road in the next 3 years.
- Marketing is full of great ideas…
- System must be available 24x7.

Customer Facing

- Web interface as well as mobile apps.
- Allow an owner to check the stats of their car.
  - When does it need maintenance? How's the battery? Etc.
  - Must be secure - only access *your* car.
  - Allow owner to summon the car to their present location.
  - Push notifications for maintenance, low battery etc.

Company Facing

- Fleet generates a lot of information that can be mined.
- Data must be anonymized.
- Ensure only authorized users have access.
- Audit access to customer data.
- Push software updates to the car.
- Push recall/update information to customers.

What quality attributes matter most here?

What words/phrases stand out to you?

Basics

- Car "phones home" on a regular basis.
- Sends a standard data payload including VIN.
- Demand is extremely high
- Expect millions of cars on the road in the next 3 years.
- Marketing is full of great ideas…
- System must be available 24x7.

Customer Facing

- Web interface as well as mobile apps.
- Allow an owner to check the stats of their car.
  - When does it need maintenance? How's the battery? Etc.
  - Must be secure - only access *your* car.
  - Allow owner to summon the car to their present location.
- Push notifications for maintenance, low battery etc.

Company Facing

- Fleet generates a lot of information that can be mined.
- Data must be anonymized.
- Ensure only authorized users have access.
- Audit access to customer data.
- Push software updates to the car.
- Push recall/update information to customers.

Auditability! Availability.
Security. Usability.

How would you rank them?

Depends on the perspective of the system!

From a driver/owner…

Rank	Quality Attribute	Comments
1	Usability	Customers will not read a manual on how to use the iDon’tDrive system.
2	Availability	Owner must be able to summon a car 24x7.
3	Security	Owner must be confident that only then can access their car.
4	Reliability	System must work as expected when called upon to maintain confidence in the system.

From a service center...

Rank	Quality Attribute	Comments
1	Security	Only authorized users should have access to the system.
2	Auditability	System should audit access to determine appropriate usage of the system.
3	Efficiency	System should be minimize the time service representatives need to find information.
4	Usability	System should require minimal training and allow new service reps to be productive quickly.

What architectural decisions result from that?

Rank	Quality Attribute	Decision(s)
1	Usability	UX designers will be engaged and ensure the design requires no training to use.
2	Availability	System will be geographically dispersed across multiple data centers. Zero downtime deploys will be utilized.
3	Security	Standard three zone security will be employed. System will encrypt personally identified information and follow all security standards.
4	Reliability	System will be geographically dispersed across multiple data centers.

Questions?

Your turn!

Architectural Katas.

http://nealford.com/katas/index.html

Pick one!

Or I can give you one.

Take time to discuss with your team.

What quality attributes matter most?

How would you rank them?

What architectural decisions might result?

Jot them down!

Use a list, a mind map…
whatever works!

Present your ideas to the rest of the group.

We'll discuss.

Questions?

As architects, we need to look beyond functionality.

We need to focus on the “ilities”.

Important that we take the time to document them.

Doesn't have to be high ceremony!

Validate them. Rank them.

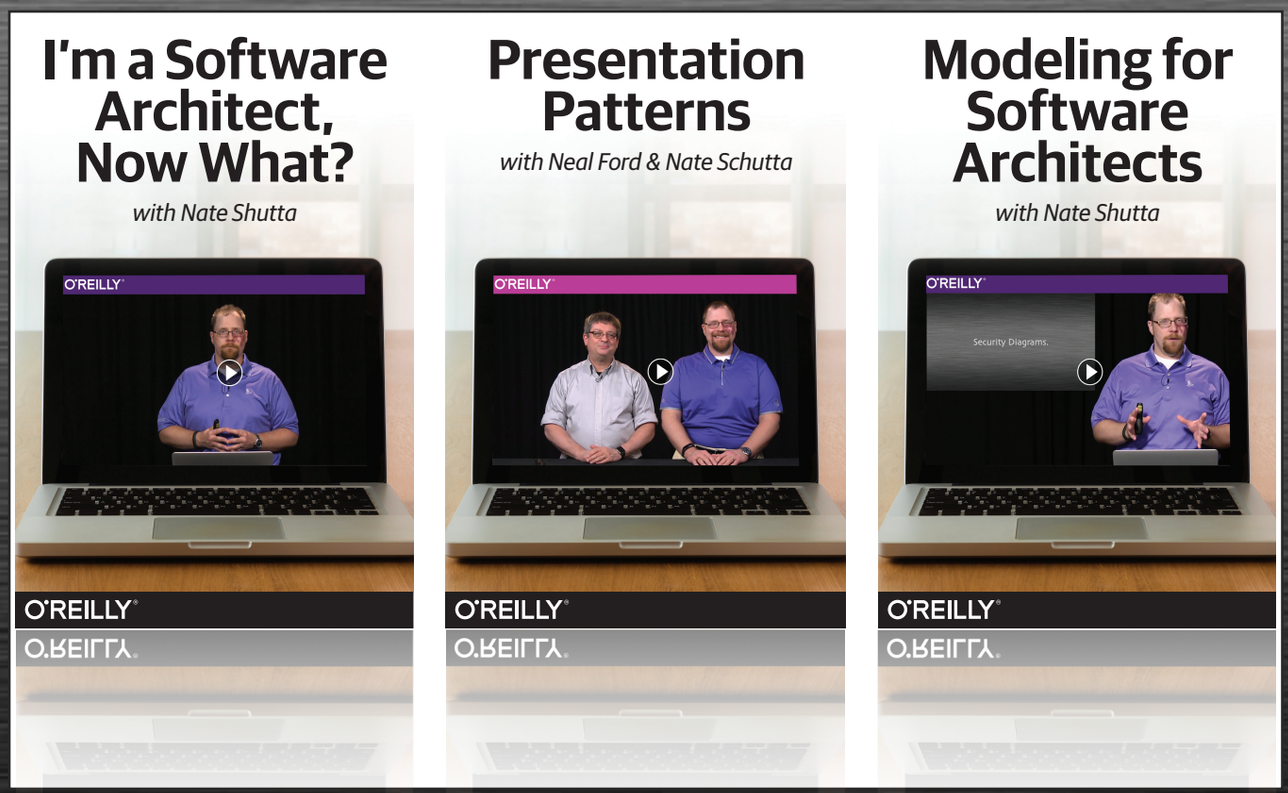
Consider the implications.

Doesn't have to be complicated!

But it is vital to project success.

Good luck!

thanks!



Nathaniel T. Schutta
@ntschutta