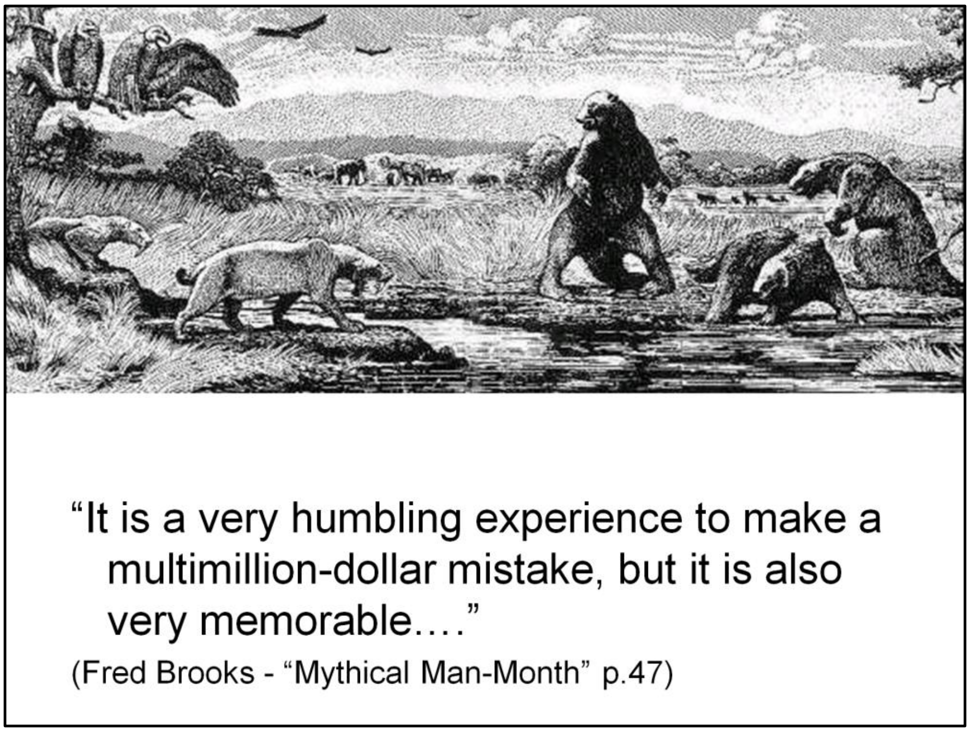
Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD

Getting SPAMMED for architecture

Arnon Rotem-Gal-Oz
Haifa, Israel
arnon@rgoarchitects.com



“It is a very humbling experience to make a multimillion-dollar mistake, but it is also very memorable….”

(Fred Brooks - “Mythical Man-Month” p.47)

More than 30 years ago (1975)

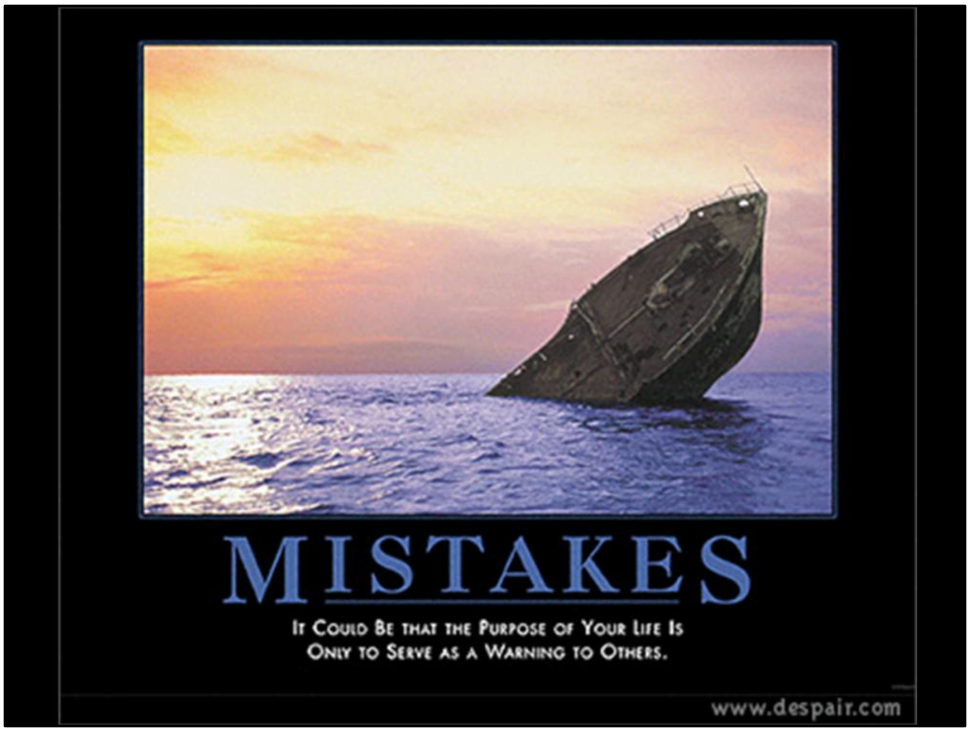
Manager of the OS/360 software project

10 people in the architecture group – Architecture manager thought he would have the spec ready in 10 month (waterfall was still en-vouge back then)

150 people in the control program group– said that working with the architect they will make it the spec in 7 months (on schedule) and not have hi men twiddle their thumbs for 10 months

Architecture manager said that this way it would not be on time (it would take the same 10 months) and would e of lower quality

The architecture manager was right on both counts.Also Brooks estimates the lack of conceptual integrity added a year to the debugging time…


MISTAKES
It Could Be that the Purpose of Your Life Is Only to Serve as a Warning to Others.
www.despair.com

We don’t want to get there- right?

What is architecture

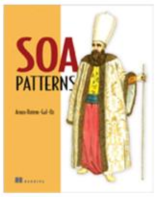
What’s the architect role

How are we going to get from nothing to a working, breathing architecture

Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD



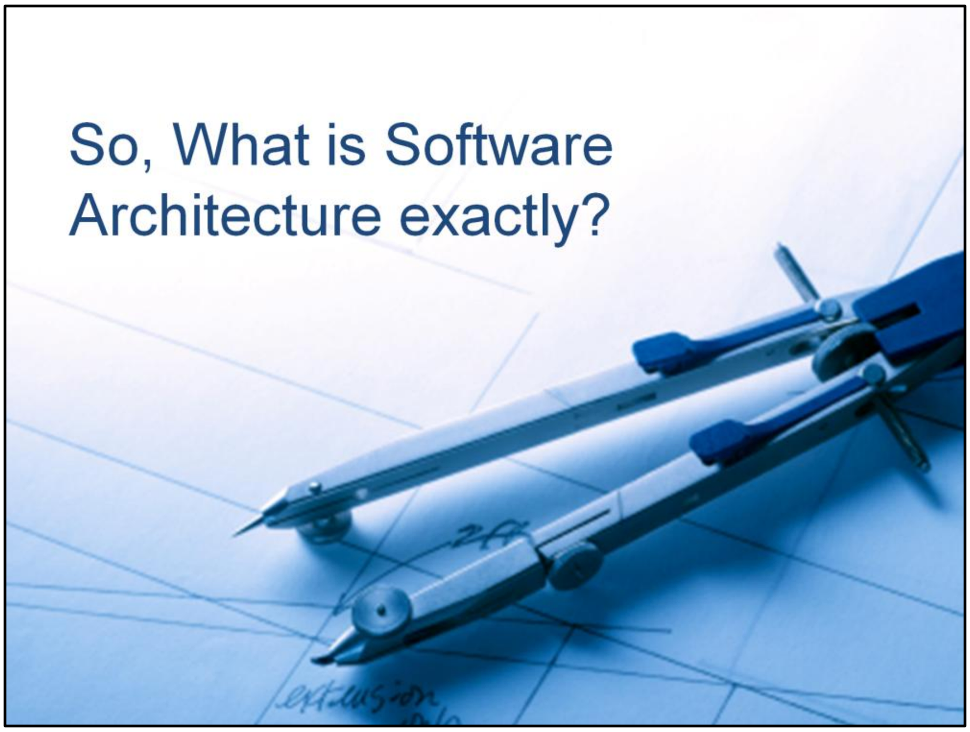
Arnon Rotem-Gal-Oz

Dr.Dobb's Portal
The World of Software Development

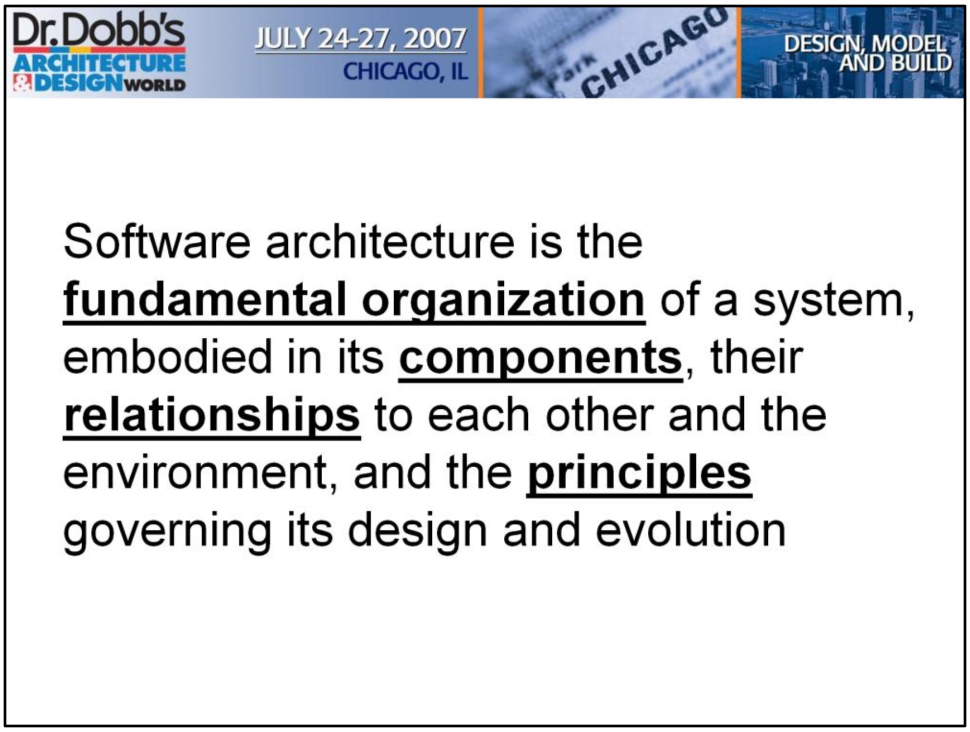
InfoQ

IASA ARCHITECT

MVP
Microsoft®
Most Valuable
Professional



So, What is Software Architecture exactly?


Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

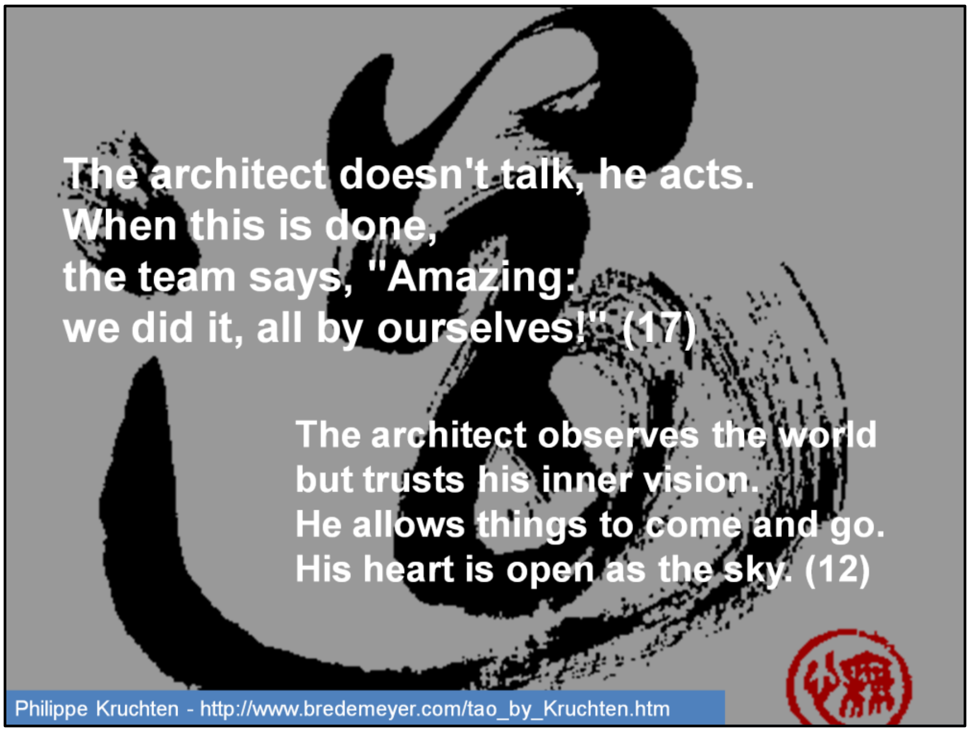
Software architecture is the **fundamental organization** of a system, embodied in its **components**, their **relationships** to each other and the environment, and the **principles** governing its design and evolution

IEEE 1471 – recommended practice for architecture description of software intensive system

Software architecture is the collection of the fundamental decisions about a software product/solution designed to meet the project's quality attributes (i.e. requirements). The architecture includes the main components, their main attributes, and their collaboration (i.e. interactions and behavior) to meet the quality attributes. Architecture can and usually should be expressed in several levels of abstraction (depending on the project's size).

If an architecture is to be intentional (rather than accidental), it should be communicated. Architecture is communicated from multiple viewpoints to cater the needs of the different stakeholders.

Architectural decisions are global tied to quality attributes
Designs decisions are local –tied to functionality

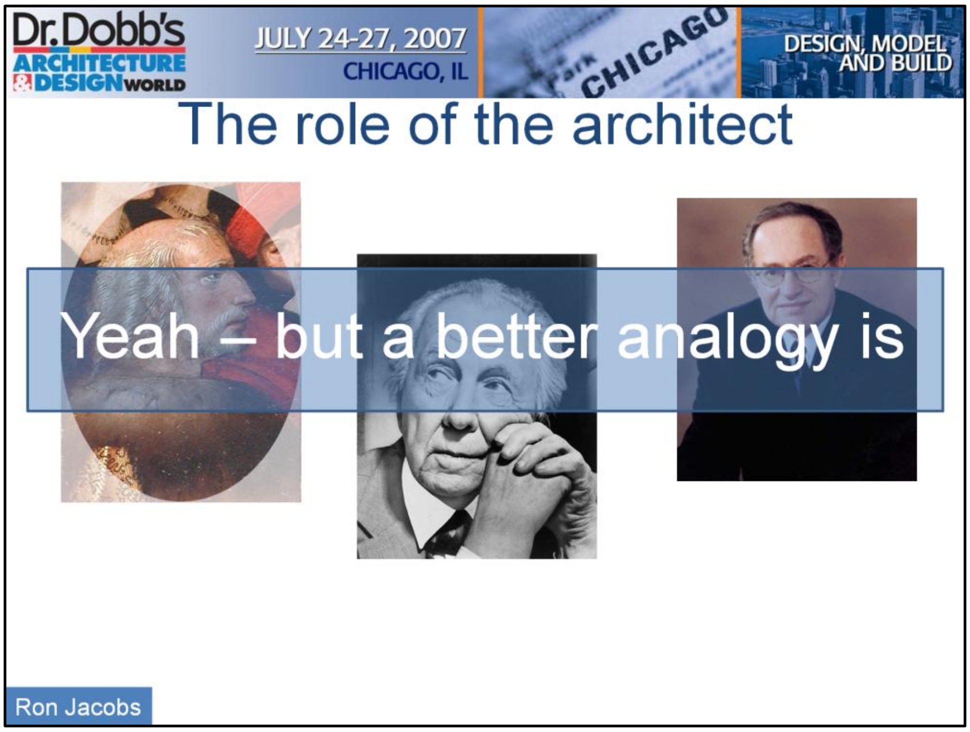


The architect doesn't talk, he acts.
When this is done,
the team says, "Amazing:
we did it, all by ourselves!" (17)

The architect observes the world
but trusts his inner vision.
He allows things to come and go.
His heart is open as the sky. (12)

Philippe Kruchten - http://www.bredemeyer.com/tao_by_Kruchten.htm

The Tao of Software Architect



The role of the architect

Yeah – but a better analogy is

Ron Jacobs

Dr.Dobb's ARCHITECTURE & DESIGN WORLD
JULY 24-27, 2007
CHICAGO, IL
CHICAGO
DESIGN, MODEL AND BUILD

Columbos - **Explorer**

Alan Dershowitz - **Advocate**

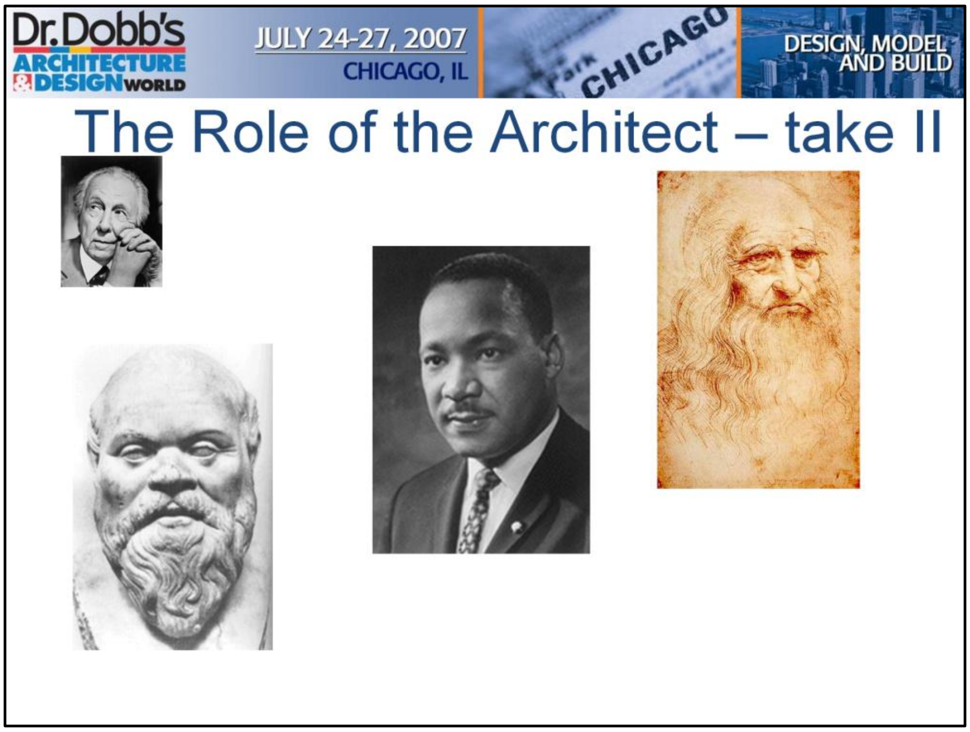
- At the age of 28 he became the youngest full professor in Harvard law school history

Successfully defended high profile clients

- O.J. Simpson
- Claus von Bülow

Frank Lloyd Wright - **Designer**

Frank Lloyd Wright (June 8, 1867 – April 9, 1959) was one of the most prominent and influential architects of the first half of 20th century. He not only developed a series of highly individual styles over his extraordinarily long architectural career (spanning the years 1887-1959), he influenced the whole course of American architecture and building. To this day he remains probably America's most famous architect. (wikipedia)



Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD

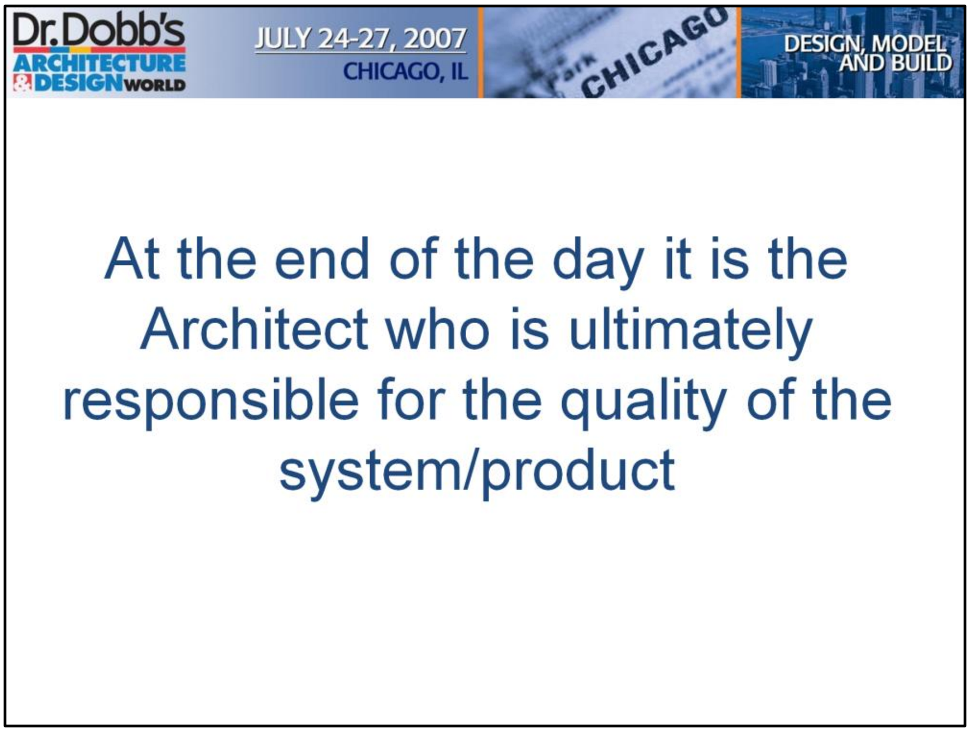
The Role of the Architect – take II

A teacher- a mentor

A visionary -

A renaissance man

An architect is someone who has an holistic view of something

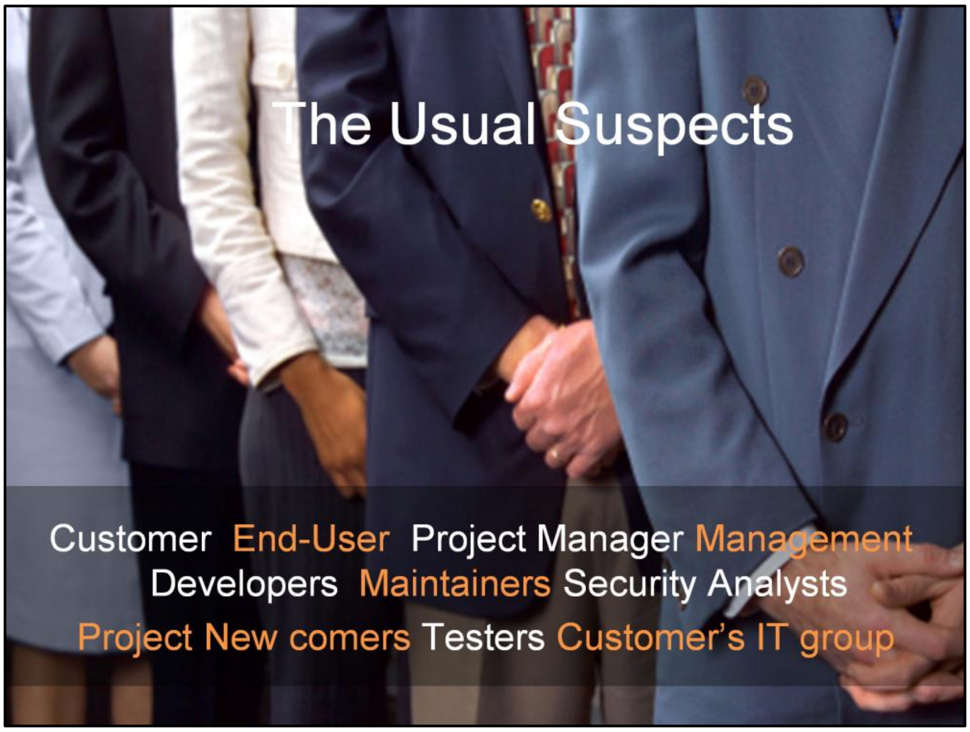


Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

At the end of the day it is the Architect who is ultimately responsible for the quality of the system/product

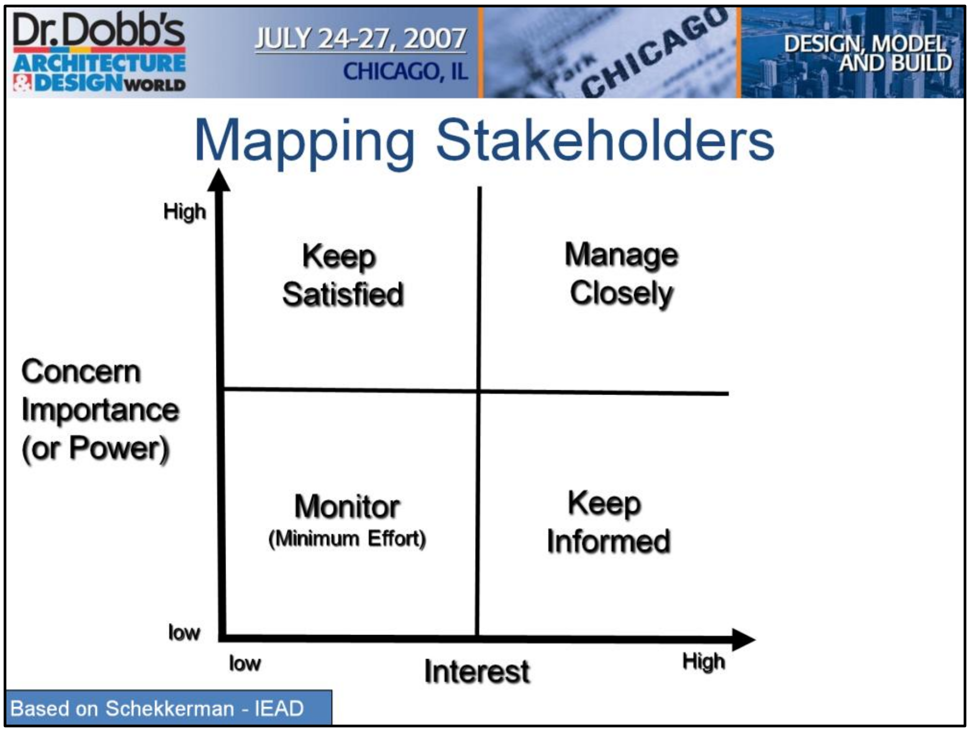
SPAMMED

First, who are we working for?



The Usual Suspects

Customer End-User Project Manager Management
Developers Maintainers Security Analysts
Project New comers Testers Customer's IT group



Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

CHICAGO

DESIGN, MODEL AND BUILD

Mapping Stakeholders

Concern
Importance
(or Power)

High

Keep
Satisfied

Manage
Closely

Monitor
(Minimum Effort)

Keep
Informed

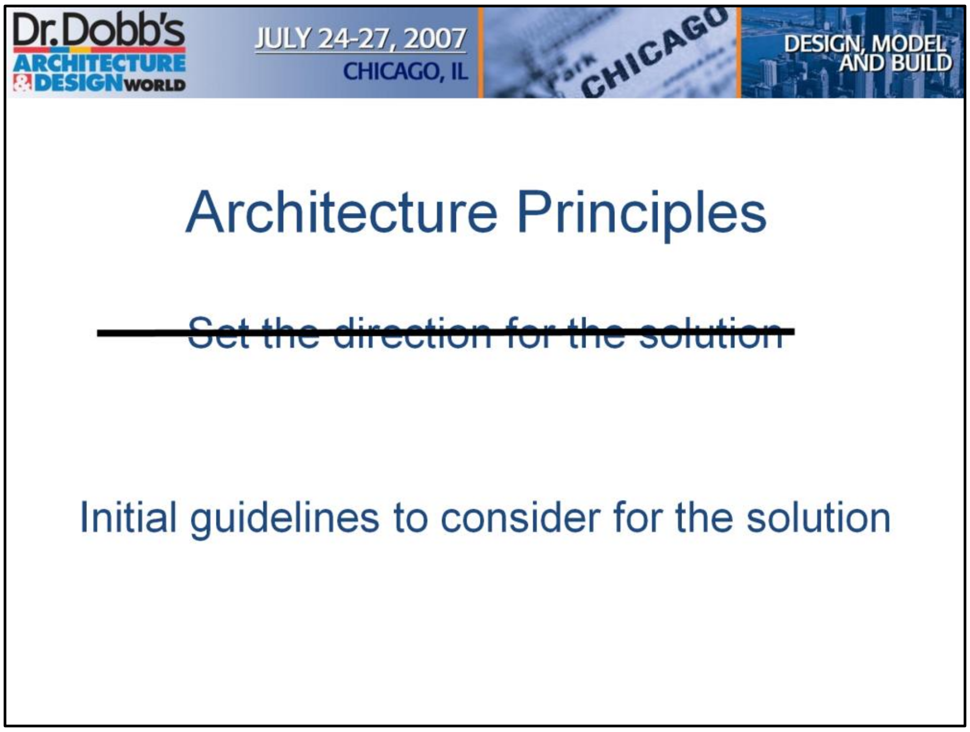
low

low

Interest

High

Based on Schekkerman - IEAD


Dr.Dobb's ARCHITECTURE & DESIGN WORLD — JULY 24-27, 2007 CHICAGO, IL — DESIGN, MODEL AND BUILD

Architecture Principles

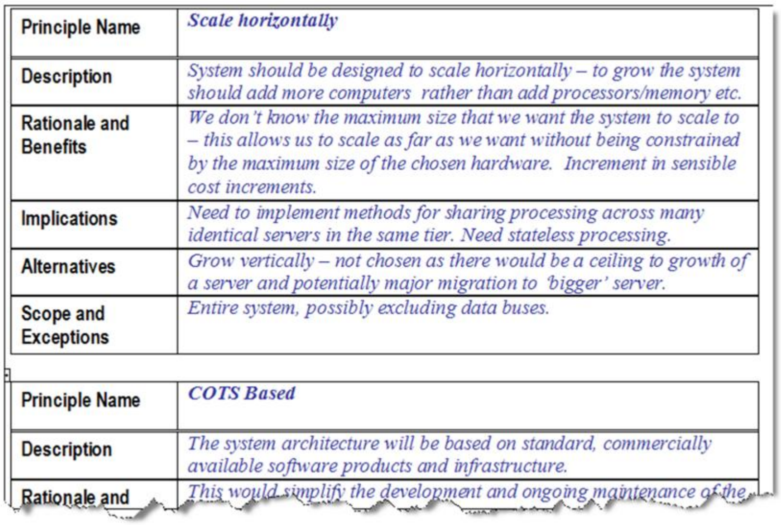
~~Set the direction for the solution~~

Initial guidelines to consider for the solution

Set the direction for the solution….

No, no, that’s actually not true.
it is just an initial guideline

YAGNI vs. Former knowledge



Principle Name	*Scale horizontally*
Description	*System should be designed to scale horizontally – to grow the system should add more computers rather than add processors/memory etc.*
Rationale and Benefits	*We don't know the maximum size that we want the system to scale to – this allows us to scale as far as we want without being constrained by the maximum size of the chosen hardware. Increment in sensible cost increments.*
Implications	*Need to implement methods for sharing processing across many identical servers in the same tier. Need stateless processing.*
Alternatives	*Grow vertically – not chosen as there would be a ceiling to growth of a server and potentially major migration to 'bigger' server.*
Scope and Exceptions	*Entire system, possibly excluding data buses.*

Principle Name	*COTS Based*
Description	*The system architecture will be based on standard, commercially available software products and infrastructure.*
Rationale and	*This would simplify the development and ongoing maintenance of the*

constraints limit the (architectural) solution space

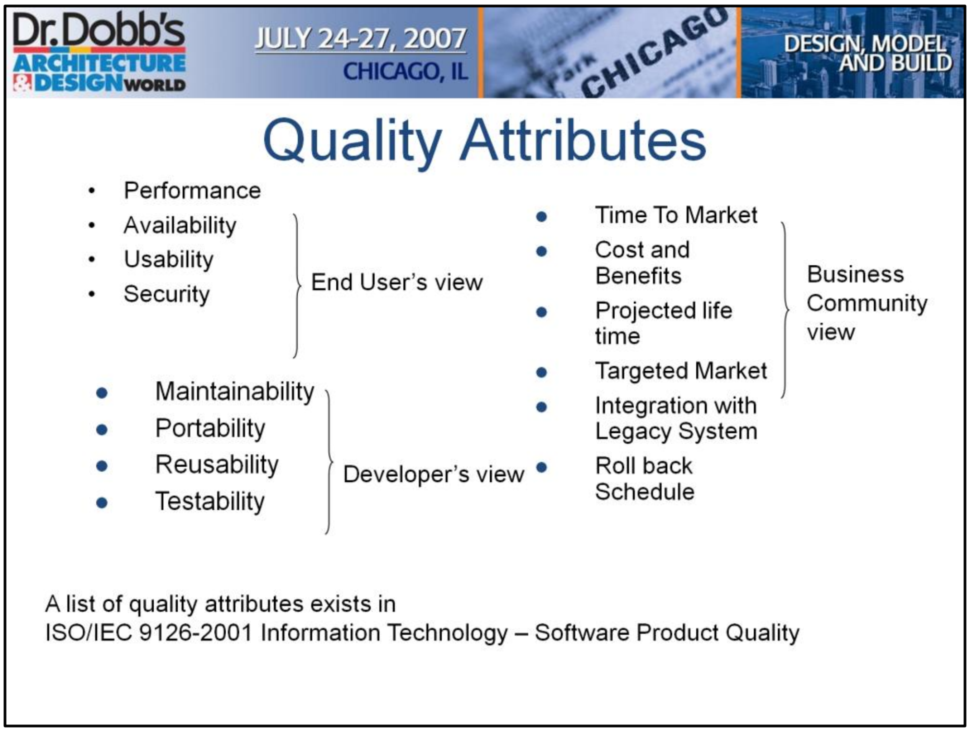
Vs. requirements that set goals for the system

Stakeholders should therefore not only specify requirements, but also constraints!

Technical – Platform/technology (e.g. use .NET)

Financial – Budget (don't event think about that fancy Rule Engine)

Next, we have to think about the quality Attributes


Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

Quality Attributes

- Performance
- Availability
- Usability
- Security

End User's view

- Maintainability
- Portability
- Reusability
- Testability

Developer's view

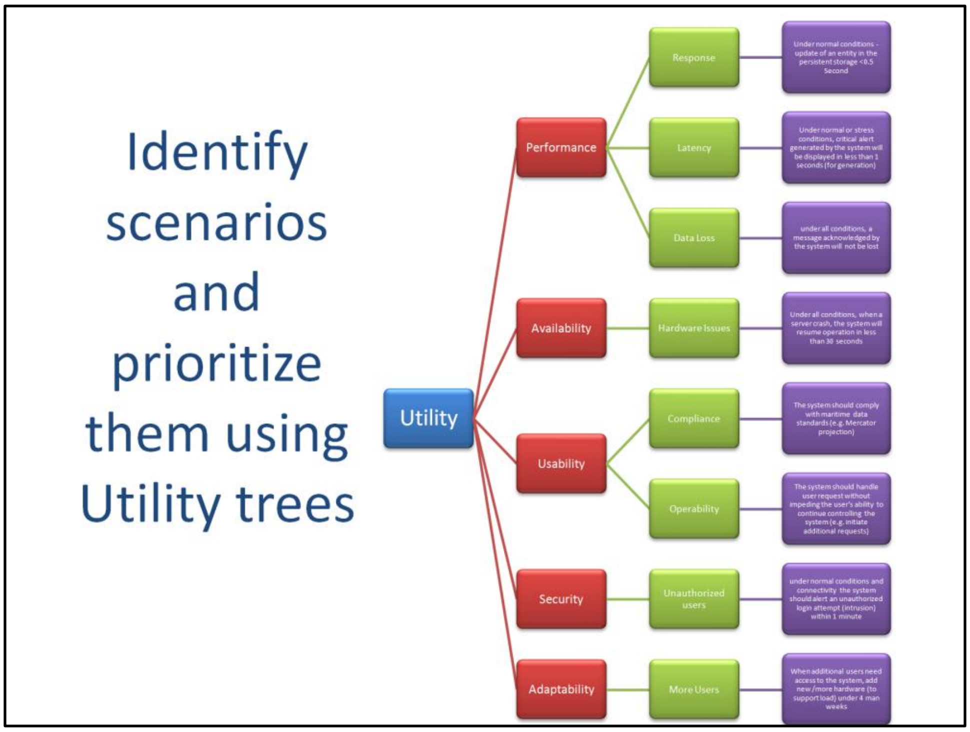
- Time To Market
- Cost and Benefits
- Projected life time
- Targeted Market

Business Community view

- Integration with Legacy System
- Roll back Schedule

A list of quality attributes exists in
ISO/IEC 9126-2001 Information Technology – Software Product Quality

We will return to this when we'll speak about Evaluating Architectures (ATAM, LAAAM)



Identify scenarios and prioritize them using Utility trees

Under normal conditions - update of an entity in the persistent storage <0.5 Second

Under normal or stress conditions, critical alert generated by the system will be displayed in less than 1 seconds (for generation)

under all conditions, a message acknowledged by the system will not be lost

Under all conditions, when a server crash, the system will resume operation in less than 30 seconds

The system should comply with maritime data standards (e.g. Mercator projection)

The system should handle user request without impeding the user's ability to continue controlling the system (e.g. initiate additional requests)

under normal conditions and connectivity the system should alert an unauthorized login attempt (intrusion) within 1 minute

When additional users need access to the system, add new /more hardware (to support load) under 4 man weeks

decompose and refines the business goals and quality attributes

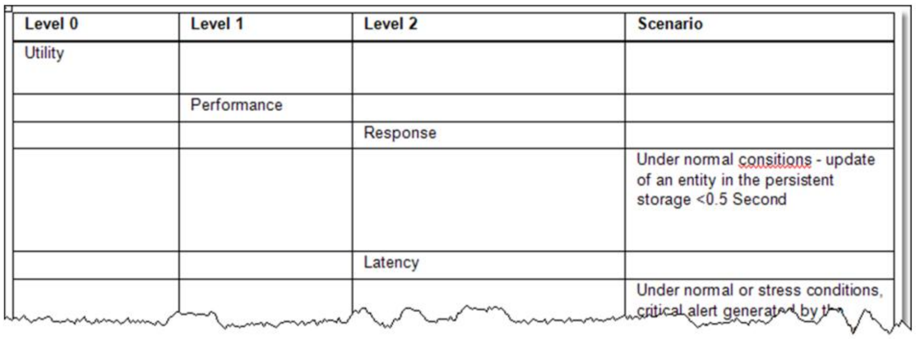
The root of the tree is “utility” – the overall “goodness” of the system

Select the most important quality goals to be the high-level nodes

E.g. performance, modifiability, security, and availability

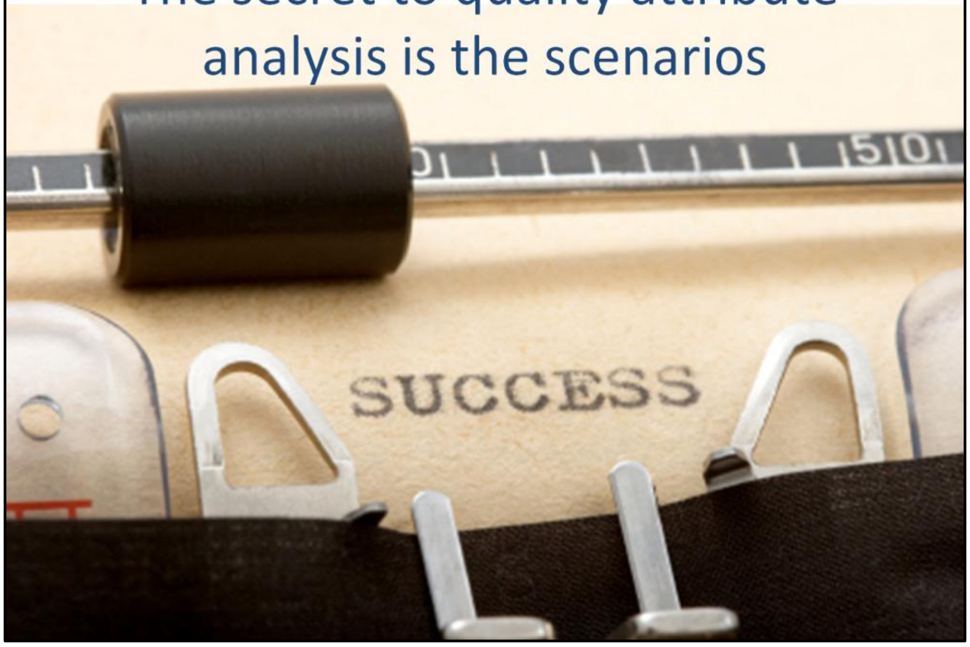
The tree reflects the hierarchical nature of quality attributes and provides the basis for prioritization

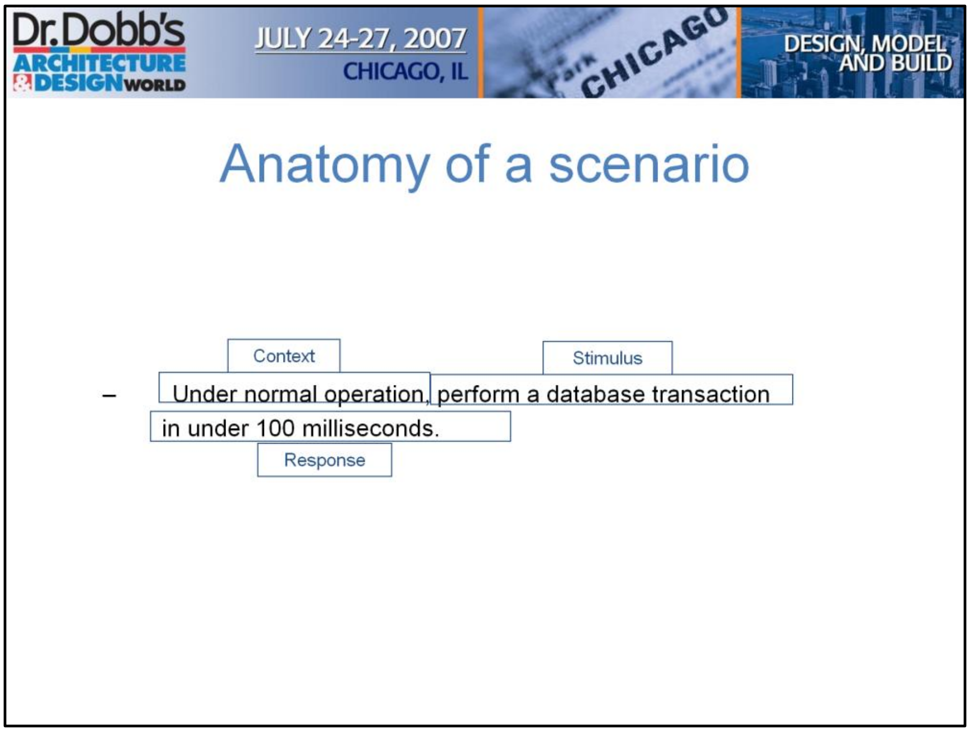
Just remember that they actually look like this…



Level 0	Level 1	Level 2	Scenario
Utility			
	Performance		
		Response	
			Under normal consitions - update of an entity in the persistent storage <0.5 Second
		Latency	
			Under normal or stress conditions, critical alert generat by th

The secret to quality attribute analysis is the scenarios





Anatomy of a scenario

– Under normal operation, perform a database transaction in under 100 milliseconds.

Context
Stimulus
Response
DR.Dobb's ARCHITECTURE & DESIGN WORLD
JULY 24-27, 2007
CHICAGO, IL
DESIGN, MODEL AND BUILD

Remote user requests a database report via the Web during peak period and receives it within 5 seconds.

Growth scenario

Add a new data server to reduce latency in scenario 1 to 2.5 seconds within 1 person-week.

For a new release, integrate a new component implementation in three weeks.

Exploratory scenario

Half of the servers go down during normal operation without affecting overall system availability.

Response

Under normal conditions update 100 moving objects on the map < 200 milisecons

Latency

Under normal or stress conditions, a critical alert generated by the system will be displayed to the user in less than 1 second

Data loss

Under all conditions a message acknowledged by the system shall not be lost (10^5 probability)

Availability

Hardware failure

When a mission is in progress, upon a server mal-function, the system will be fully operable within 30 seconds or less

Changeability

Add Feature

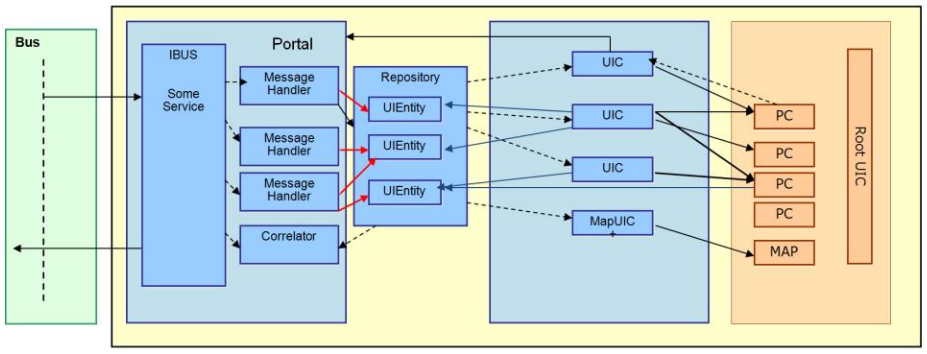
Add a new sensor-type to the system in 2 man-months or less

Oh yeah – there’s also modeling and design…

Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD

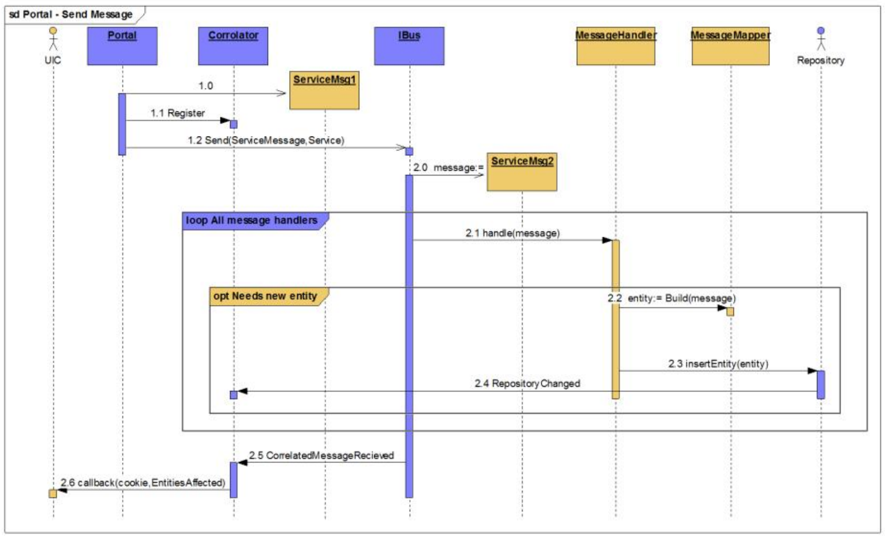


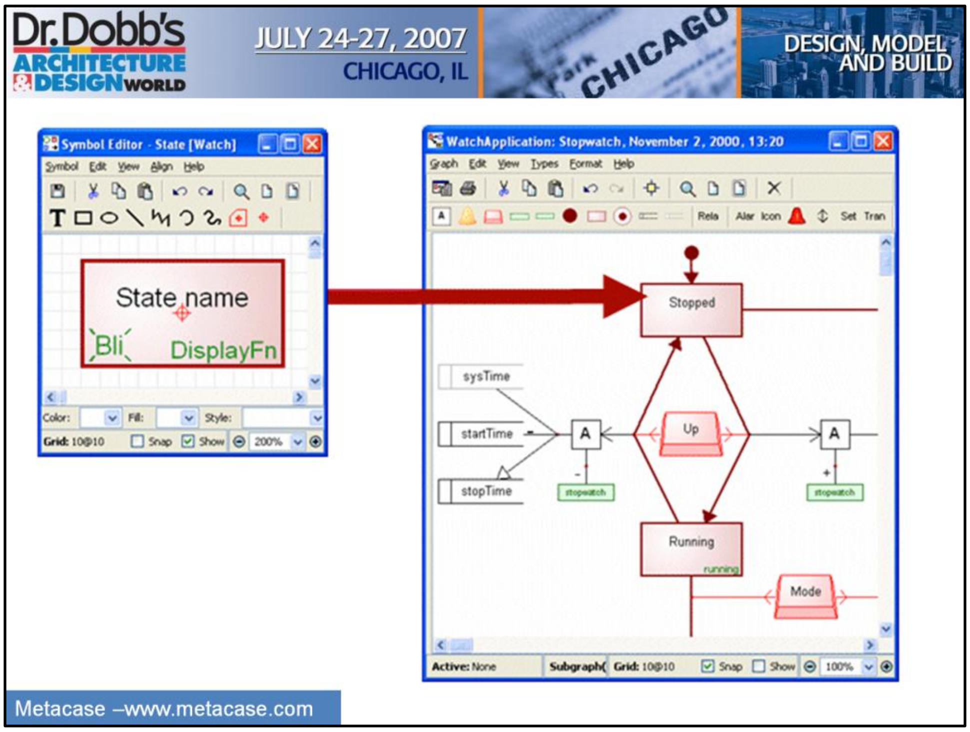
Block diagram, UMLs DSL

Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD




Dr.Dobb's ARCHITECTURE & DESIGN WORLD
JULY 24-27, 2007
CHICAGO, IL
CHICAGO
DESIGN, MODEL AND BUILD
Symbol Editor - State [Watch]
State name
Bli
DisplayFn
WatchApplication: Stopwatch, November 2, 2000, 13:20
Stopped
sysTime
startTime
stopTime
Up
A
stopwatch
Running
running
Mode
Metacase –www.metacase.com

DSL

I can't show you an example from a tool we've made to – simulate and integrate systems.

Software Factories, MDA

Once we had "Model" -> "code" (CASE tools) – didn't work because of "The Generation Gap"

Model + framework -> code +framework

Model -> Model -> Model -> model + framework -> code + framework

Small – code DSLs are better than small model DSLs

Large model DSLs are very hard to achieve

Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007
CHICAGO, IL

DESIGN, MODEL AND BUILD

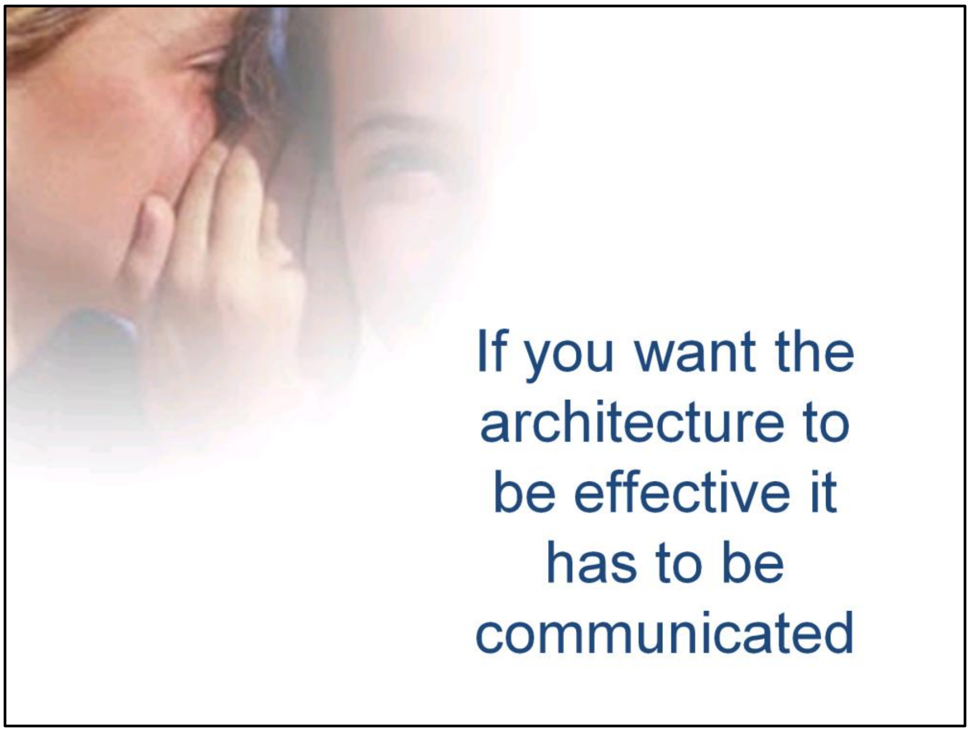
Pattern can save you the trouble of reinventing the wheel

Patterns- package an experience

Context and solutions (not “best practices”)

Encapsulate forces and challenges
Considerations

Remember that patterns are not a silver bullet either..

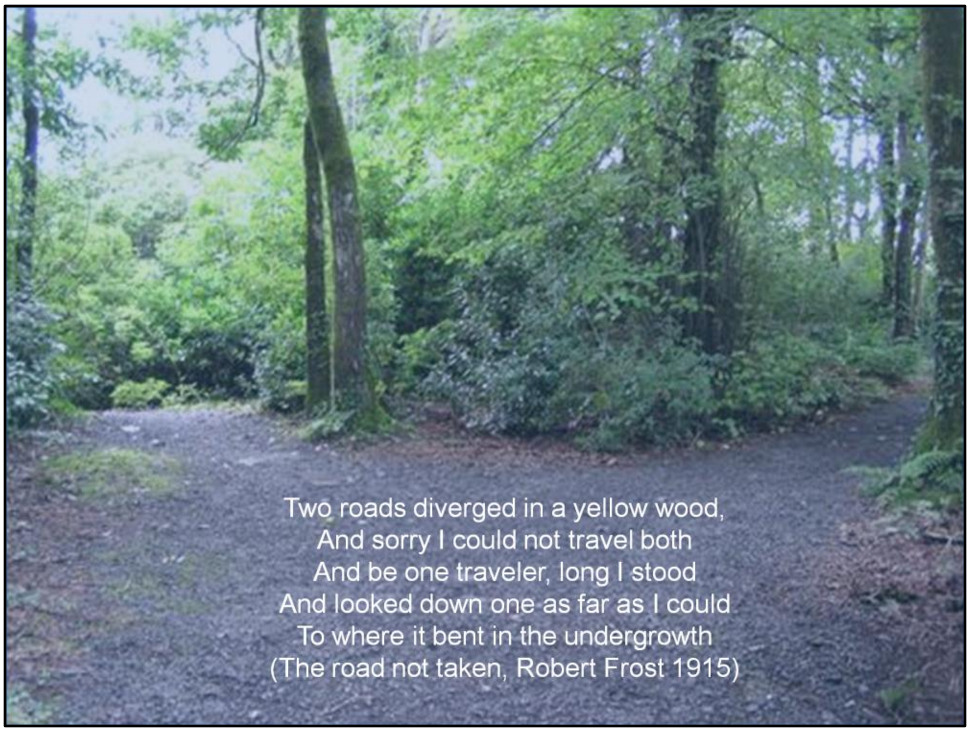


If you want the architecture to be effective it has to be communicated

Communication != elaborate documentation

Viewpoints,

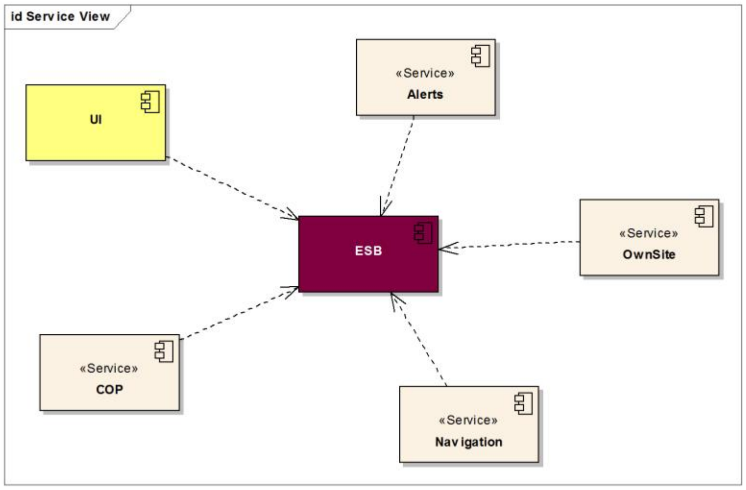
Document architecture at the last responsible moment

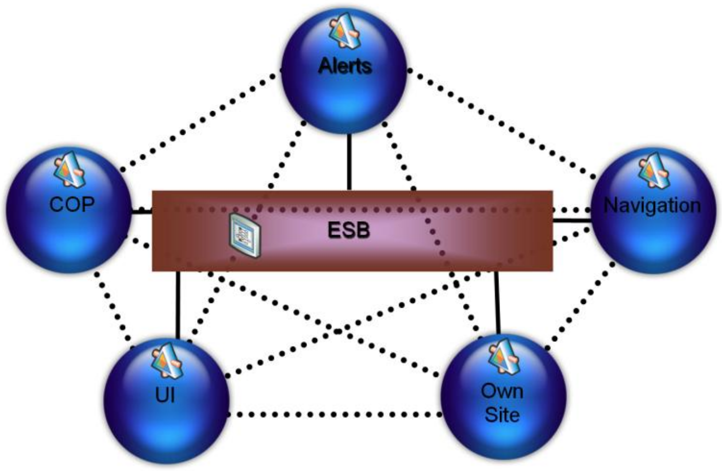


Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth
(The road not taken, Robert Frost 1915)

When you present an architecture you should consider your target audience

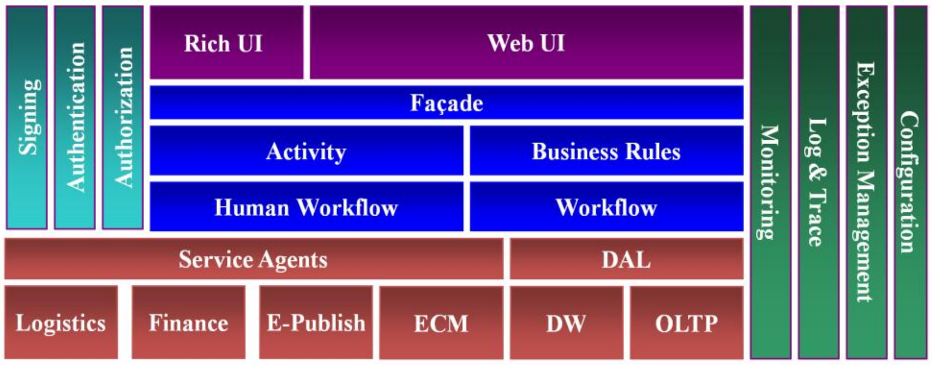
Compare this:

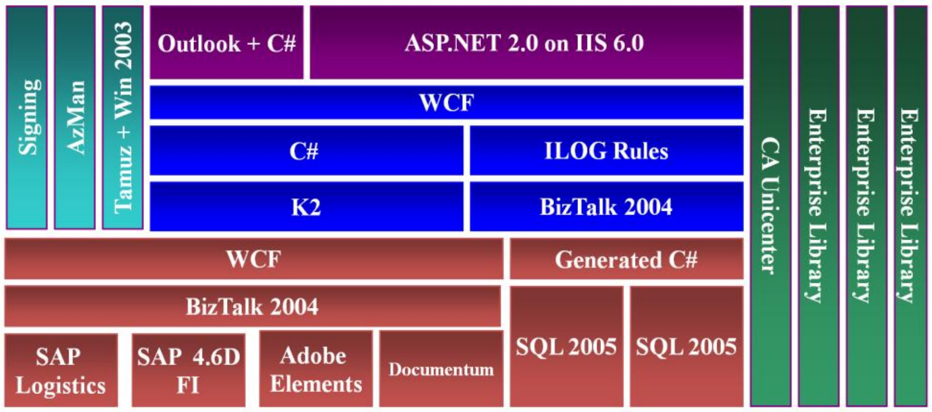




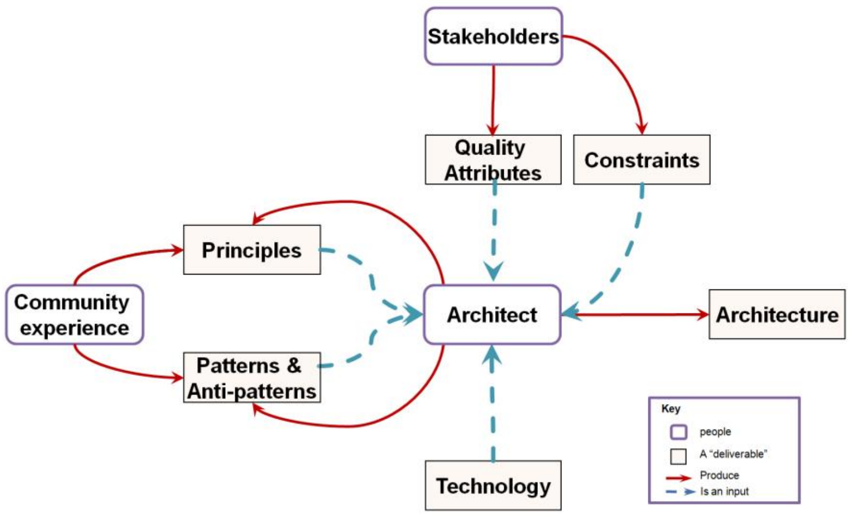
With this

Technology Mapping

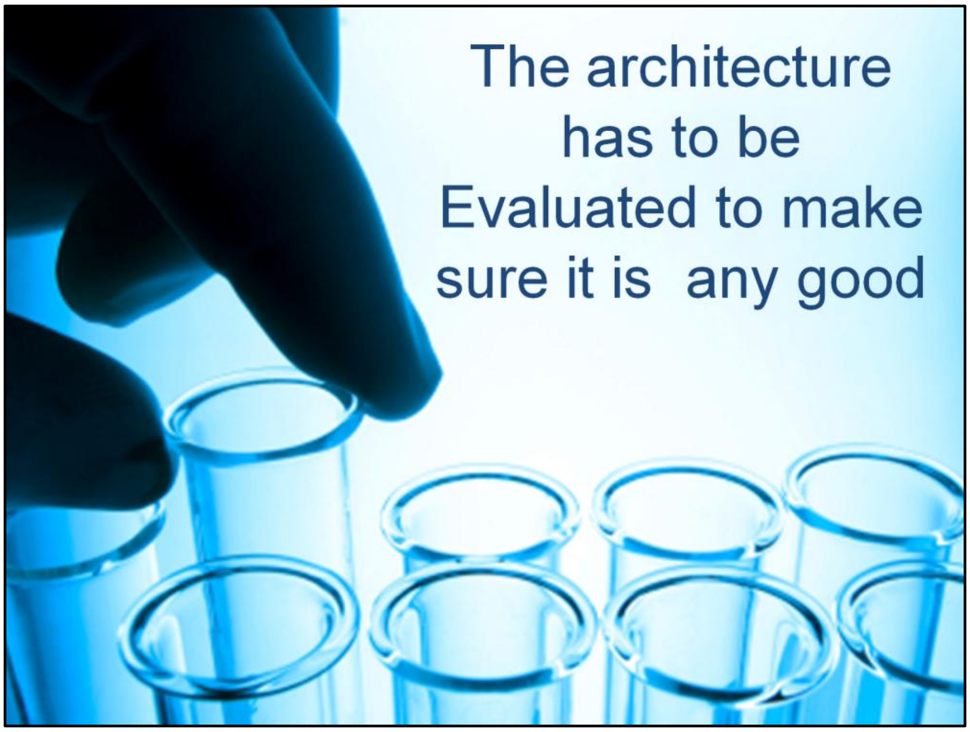

Signing
Authentication
Authorization
Rich UI
Web UI
Façade
Activity
Business Rules
Human Workflow
Workflow
Service Agents
DAL
Logistics
Finance
E-Publish
ECM
DW
OLTP
Monitoring
Log & Trace
Exception Management
Configuration


Signing
AzMan
Tamuz + Win 2003
Outlook + C#
ASP.NET 2.0 on IIS 6.0
WCF
C#
ILOG Rules
K2
BizTalk 2004
WCF
Generated C#
BizTalk 2004
SAP Logistics
SAP 4.6D FI
Adobe Elements
Documentum
SQL 2005
SQL 2005
CA Unicenter
Enterprise Library
Enterprise Library
Enterprise Library

Where are we at?



OK then, so we are all set

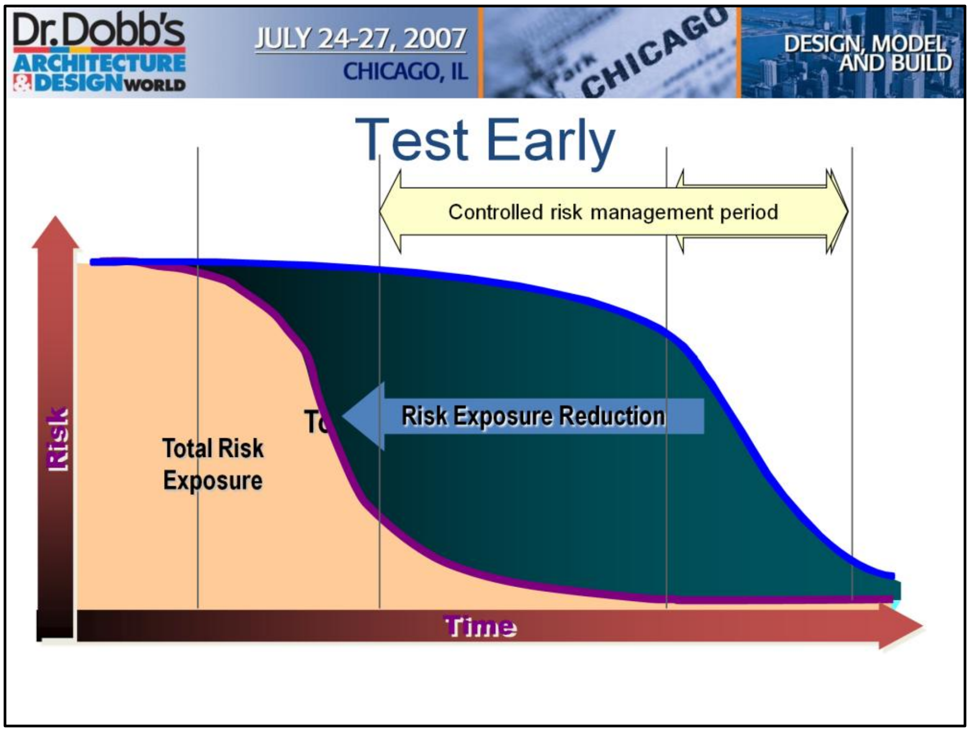

The architecture has to be Evaluated to make sure it is any good

On Paper

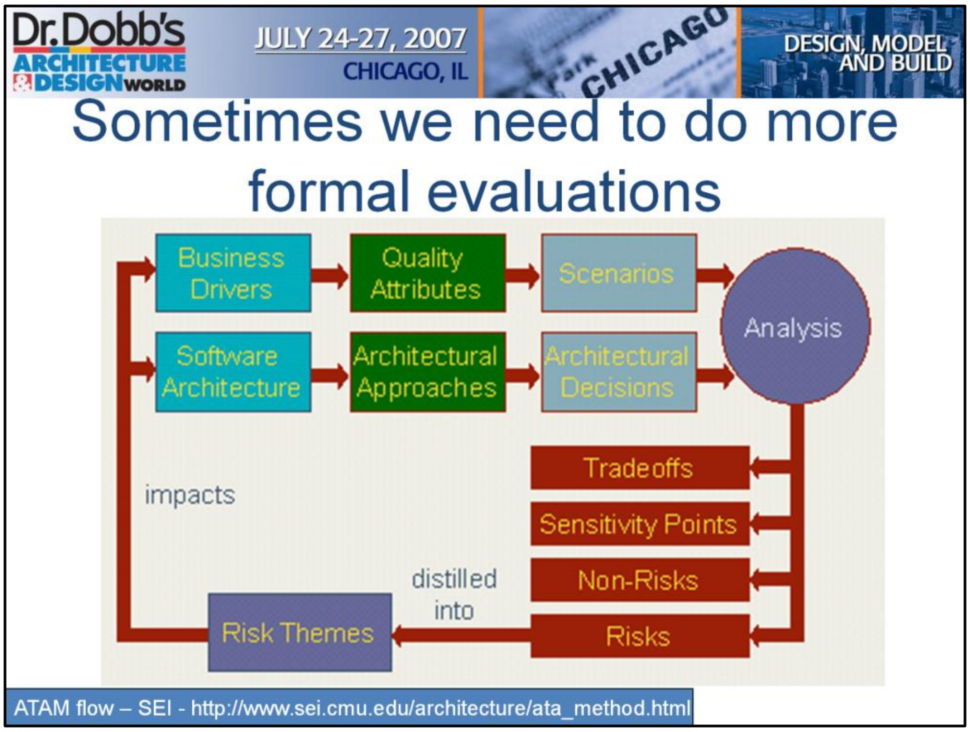
- SEI
  - ATAM; SAAM; ARID
- LAAAM
- Active Design Reviews

In Code

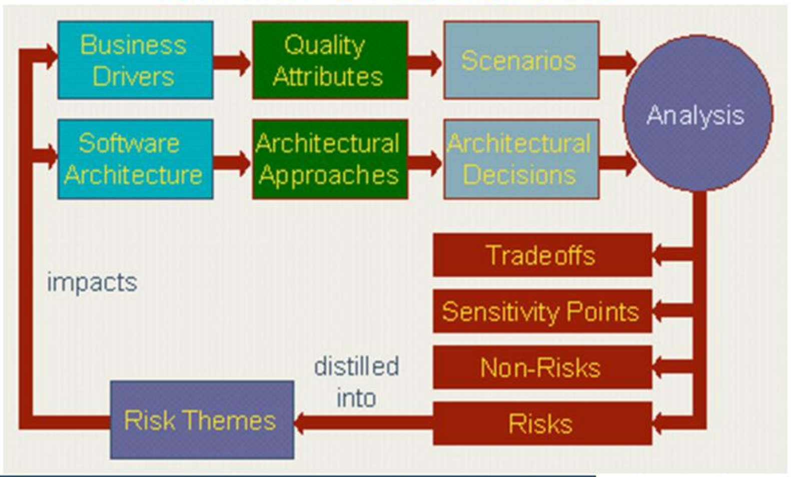
- POCs
- prototype
- Skeleton



Lets try to think about architectural risks in our projects…



Sometimes we need to do more formal evaluations



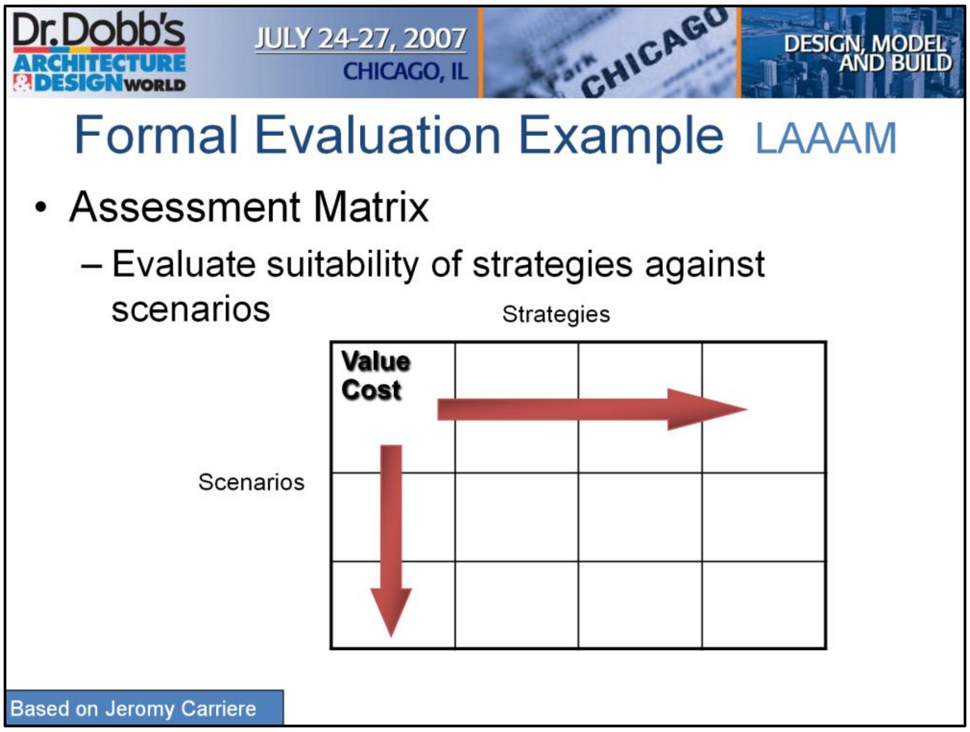
ATAM flow – SEI - http://www.sei.cmu.edu/architecture/ata_method.html

SEI

ATAM; SAAM; ARID

LAAAM

Active Design Reviews


Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

Formal Evaluation Example LAAAM

- Assessment Matrix
  - Evaluate suitability of strategies against scenarios

Strategies
Scenarios
Value
Cost

Based on Jeromy Carriere

Each dimension is rated on a five point scale, from High to Low

Value

Operational cost

Development cost

Each dimension is given a weight, to express its importance relative to the other dimensions

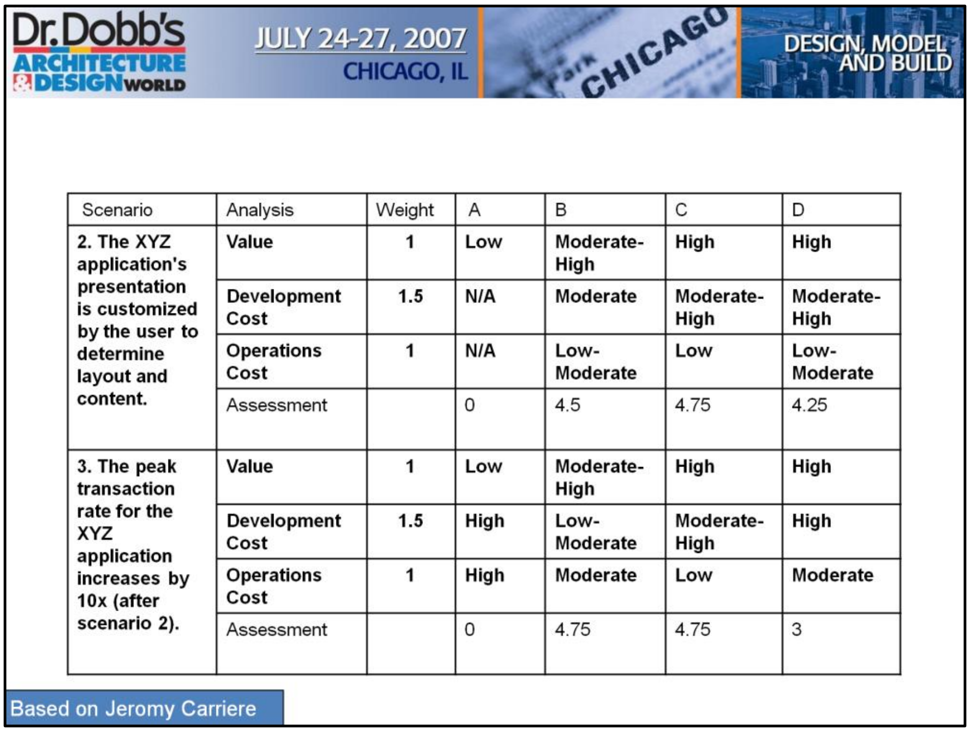
Assessment is performed in two passes:

1. Treat each cell as independent
2. Normalize across each row

Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

			Strategy			
Scenario	Analysis	Weight	**A. Perform no rearchitecting. Maintain with minimal effort the existing ABC application architecture. Introduce no new dependencies on ABC components.**	**B. Incrementally wrap existing ABC application components in the model provided with .NET.**	**C. Completely port existing ABC applications to .NET.**	**D. Completely port existing ABC applications to J2EE, using existing enterprise frameworks.**
1. A new application leverages the XYZ data store.	**Value**	**1**	**Moderate**	**Moderate-High**	**Moderate**	**Moderate**
	Development Cost	**1.5**	**High**	**Low**	**High**	**High**
	Operations Cost	**1**	**Low**	**Low-Moderate**	**Low**	**Low-Moderate**
	Assessment		3	6	3	2.5

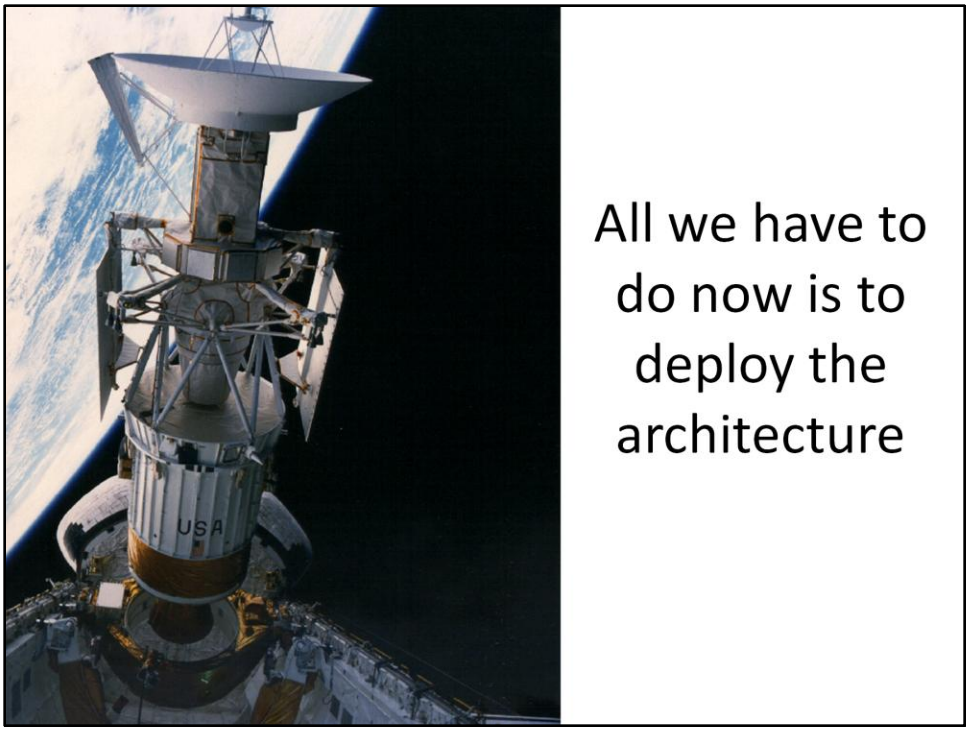
Based on Jeromy Carriere



Dr.Dobb's ARCHITECTURE & DESIGN WORLD JULY 24-27, 2007 CHICAGO, IL DESIGN, MODEL AND BUILD

Scenario	Analysis	Weight	A	B	C	D
2. The XYZ application's presentation is customized by the user to determine layout and content.	**Value**	**1**	**Low**	**Moderate-High**	**High**	**High**
	Development Cost	**1.5**	**N/A**	**Moderate**	**Moderate-High**	**Moderate-High**
	Operations Cost	**1**	**N/A**	**Low-Moderate**	**Low**	**Low-Moderate**
	Assessment		0	4.5	4.75	4.25
3. The peak transaction rate for the XYZ application increases by 10x (after scenario 2).	**Value**	**1**	**Low**	**Moderate-High**	**High**	**High**
	Development Cost	**1.5**	**High**	**Low-Moderate**	**Moderate-High**	**High**
	Operations Cost	**1**	**High**	**Moderate**	**Low**	**Moderate**
	Assessment		0	4.75	4.75	3

Based on Jeromy Carriere



All we have to do now is to deploy the architecture

Making sure the architecture really fits the problem

Making sure the architecture is followed

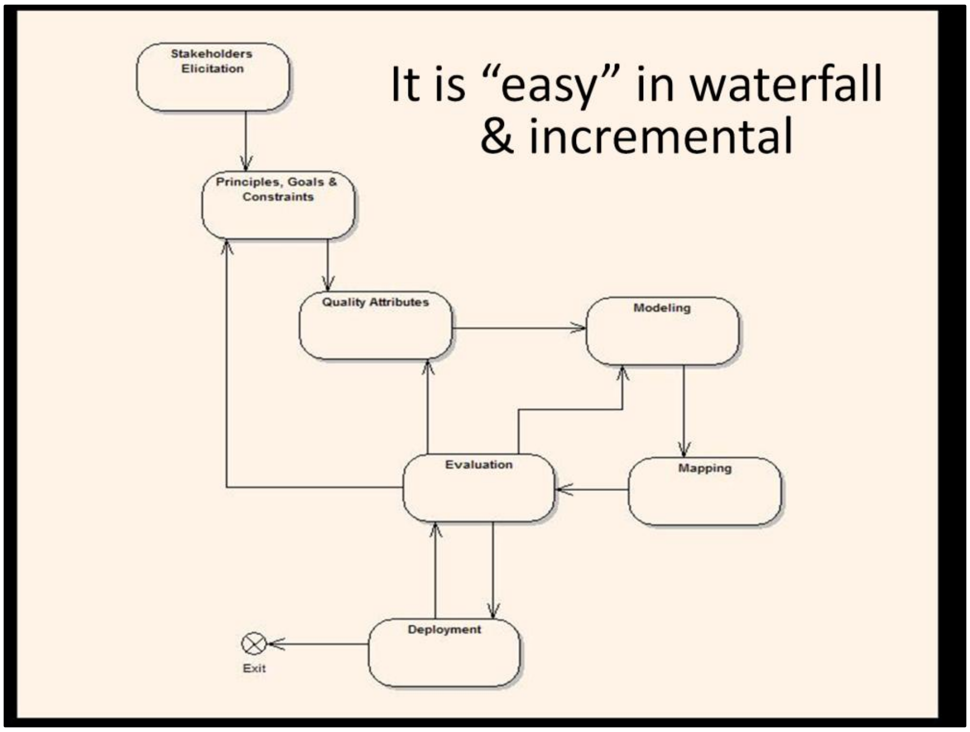
Tip: Short iterations allow for better feedback loop

Consider SCRUM's 30 day sprints or less

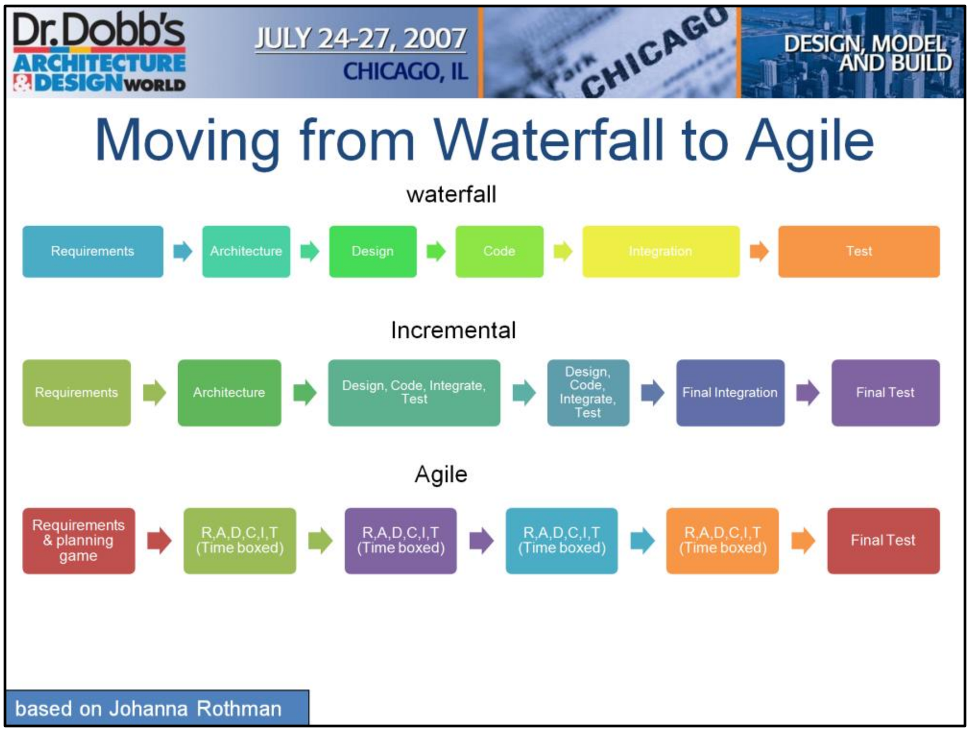
Stakeholders
Principles
Attributes
Modeling
Mapping
Evaluation
Deployment

Not a process guidance

Just a framework of activities that can be used in a variety of ways


It is "easy" in waterfall & incremental

But we've learned that Waterfall is problematic



Dr.Dobb's ARCHITECTURE & DESIGN WORLD

JULY 24-27, 2007 CHICAGO, IL

DESIGN, MODEL AND BUILD

Moving from Waterfall to Agile

based on Johanna Rothman

Iterative is better – but essentially we are doing smaller waterfalls…

Incremental we are doing “mini-waterfalls”

In Agile we don’t

We can’t fix

Time boxing gives us rhythm

Potentially shippable software

Manage requirements changes

Increase trust (demonstration)

Is located in San Jose california

In 1884, a wealthy widow named Sarah L. Winchester began a construction project of such magnitude that it was to occupy the lives of carpenters and craftsmen until her death thirty-eight years later.

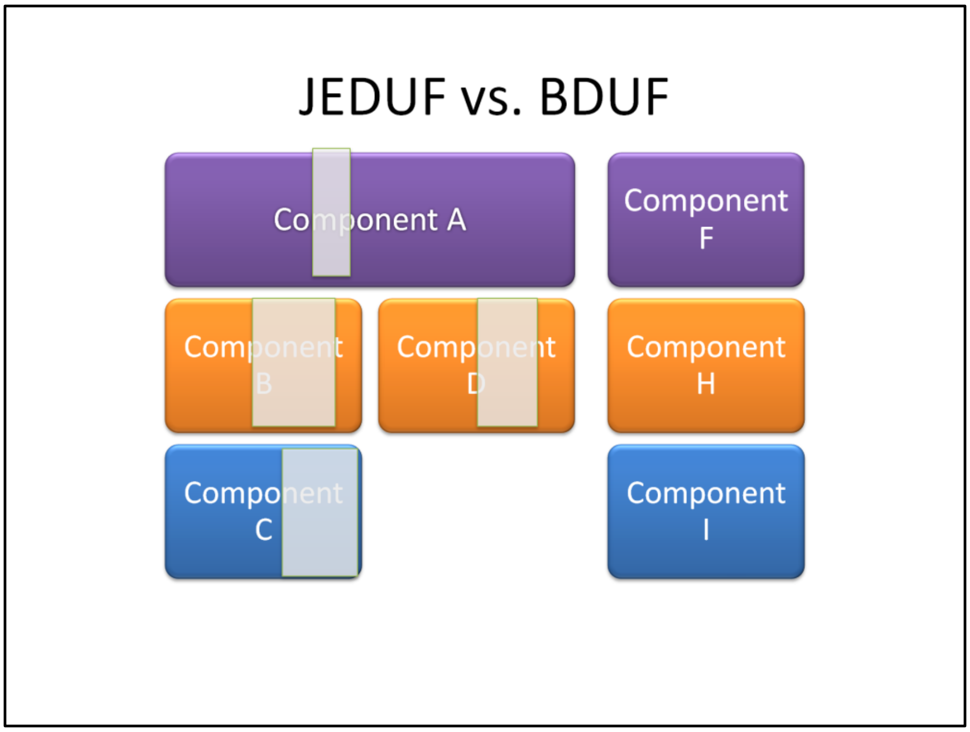
The Victorian mansion, designed and built by the Winchester Rifle heiress,

Winchester Mystery House

This is what hacks look like

- 38 years of construction – 147 builders 0 architects
- 160 rooms – 40 bedrooms, 6 kitchens, 2 basements, 950 doors
- 65 doors to blank walls, 13 staircases abandoned, 24 skylights in floors
- No architectural blueprint exists

Can you do Agile Architecture?

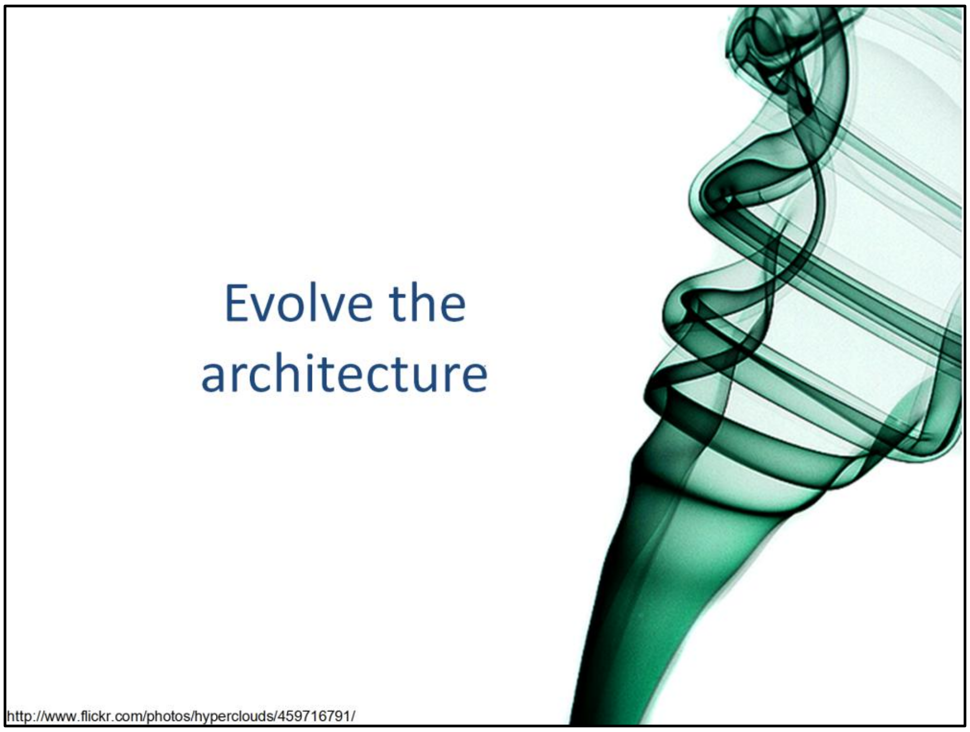


JEDUF vs. BDUF

Just Enough Design Up Front

instead of Big Design Up Front

Lean Architecture



Evolve the architecture

http://www.flickr.com/photos/hyperclouds/459716791/

Architect product owner

Emphasize Flexibility

Postpone decisions

Evolving an architecture sounds very compelling but it is not a simple feat. Architectural decisions tend to have system wide implications which means that changing one too late in the game you'd get a lot of rewrite and/or refactoring to do
.
My strategy to solve that conflict is to:
Set the first one or two iterations as architectural ones. Some of the work in these iterations is to spike technological and architectural risk. Nevertheless most of architectural iterations are still about delivering business value and user stories. The difference is that the prioritization of the requirements is also done based on technical risks and not just business ones. By the way, when you write quality attribute requirements as scenarios makes them usable as user stories helps customers understand their business value.

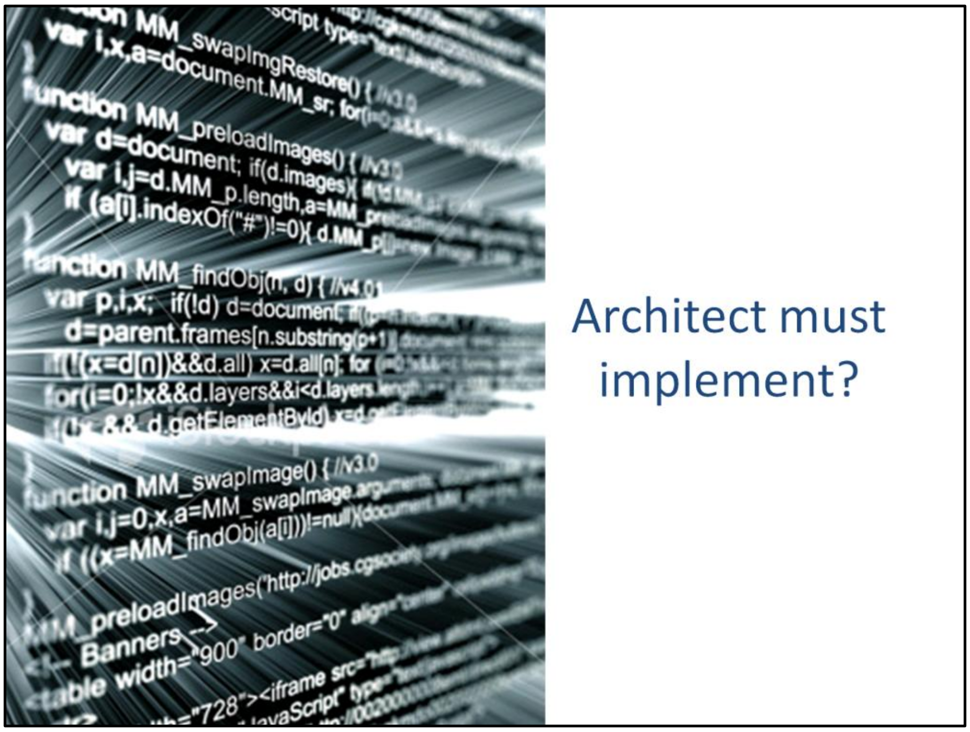
Try to think about prior experience to produce the baseline architecture

One of the quality attributes that you should bring into the table is flexibility - but be weary of putting too much effort into building this flexibility in

Don't try to implement architectural components thoroughly - it is enough to run a thin thread through them and expand then when the need arise. Sometimes it is even enough just to identify them as possible future extensions.

Try to postpone architectural decisions to the last responsible moment. However, when that moment comes - make the decision. try to validate the architectural decisions by spiking them out before you introduce them into the project

These steps don't promise that the initial architecture sticks, but in my experience it makes it possible to minimize the number of architectural decisions but still have a relatively solid foundation to base your project on



Architect must implement?

Scott Ambler told me that “agile ones do”, Jim Coplien “Architect Also Implements” pattern

Reports that they’ve seen this time and time again in successful projects.

For instance, In one presentation I heared Jim mentioned one stellar team- the dev. Team of Quatro pro where the architects had a daily standup (that was 93 mind-you)

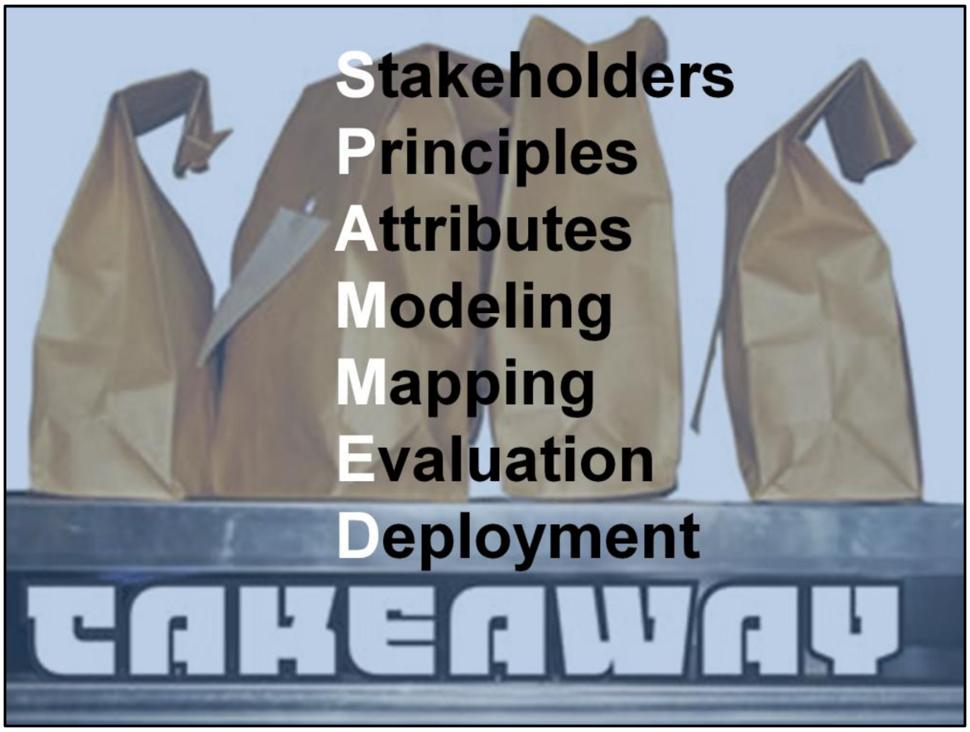
In my experience Architect should almost never own features

I don’t find a lot of value in architects implementing production code unless there are enough architects to go around

Architect must know how to implement

Architect must be able to prove his design in code

Architect can pair program to mentor/validate/solve problem and provide guidance -> this solves the getting recognition by developers part and better


Stakeholders
Principles
Attributes
Modeling
Mapping
Evaluation
Deployment

Services interactions are message driven
Services should be Loosely coupled
Edges should provide location transparency
Business logic and edge are separate layers
Scale inside the service
You can use workflows for long-running interactions
again - inside the service