

# Creating an Architecture Oral History

Minimalist Techniques for Describing Systems

---

Michael Keeling  
Vivísimo  
@michaelkeeling

## Documentation is Important...

Software Architecture Descriptions  
have some major drawbacks.



SAD without Purpose?

**NO!**

5

Architecture Documentation  
is an investment.

6

# LOAN APPLICATION

**Personal Information**

Name (Last)	PUBLIC	(First)	JOHN	(Middle Initial)		Home Telephone	(11)11 - 1111
Address (Mailing Address)	12345 MAIN STREET			(City)	ANYWHERE	Other Telephone	(22)22 2222
E-Mail Address	JQPJQPJQP@JQP.JQP			(State)	22	Zip	999999

APPROVED

APPLICANT IS UNDER REVIEW

SUBJECT	REVIEW

**Services needed**

UNDER REVIEW

**Current Income**

High School Graduate Or General Education (GED) Test Passed? Yes No

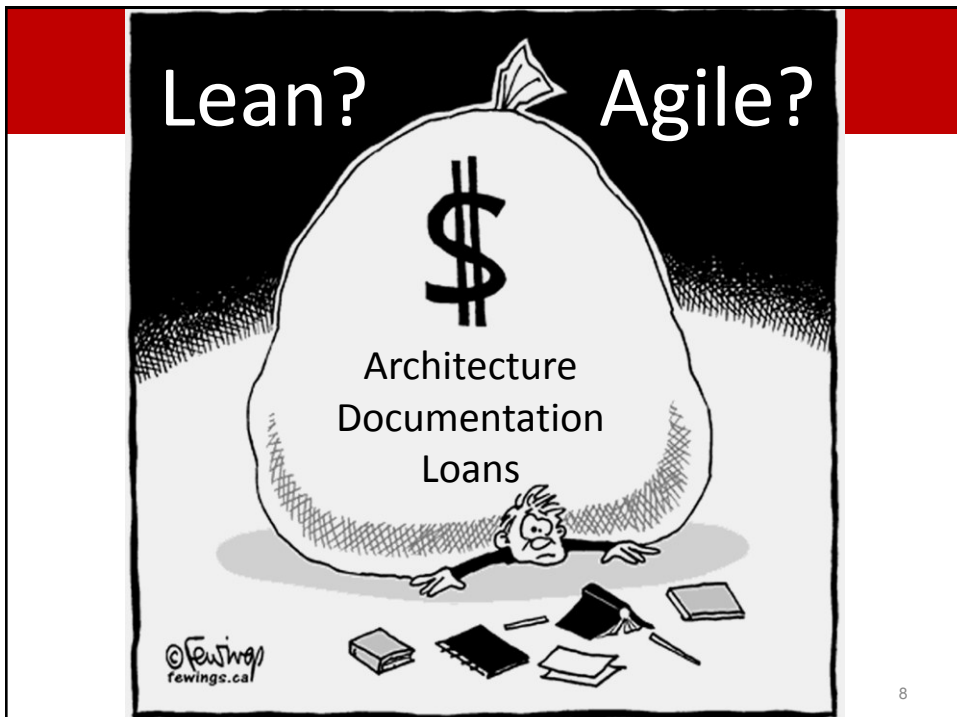
Most grade completed

**Military (Most recent first)**

Credits Earned

Graduate

Major or Subject



## Your Documentation Loan

How much?  
When?  
What do I get?

9

## Documenting Software Architecture

What you document and  
how you document it  
should change as the  
project evolves.

10

# Architecture Oral History

11

**EARLY IN THE SOFTWARE  
LIFECYCLE**

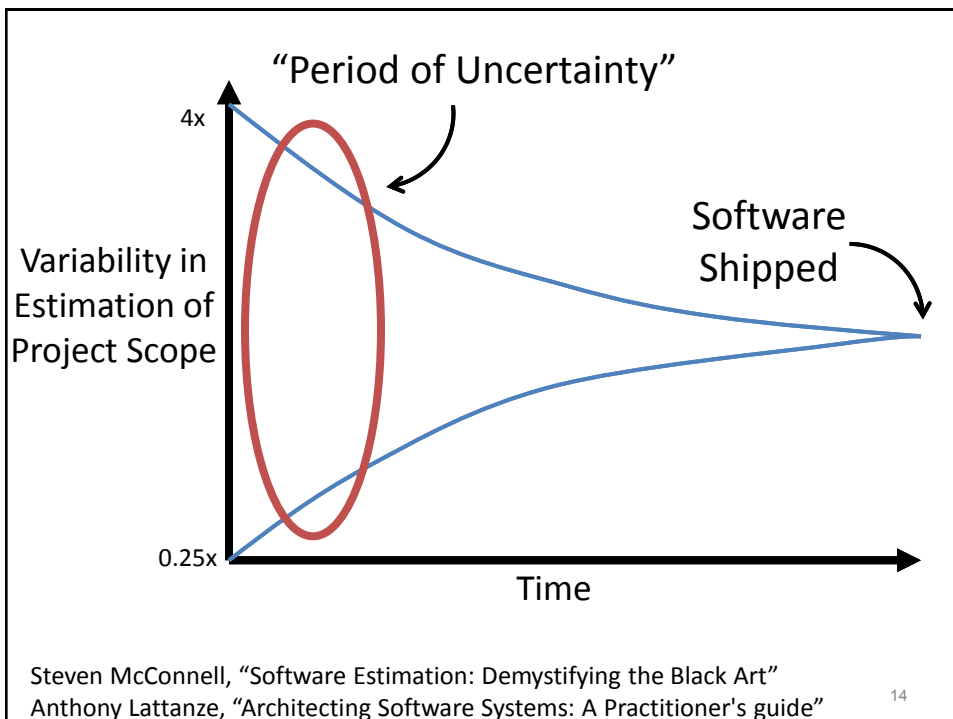
12

*“The easiest time to document the software architecture is at the end of the project...”*

*All the hard decisions have already been made!”*

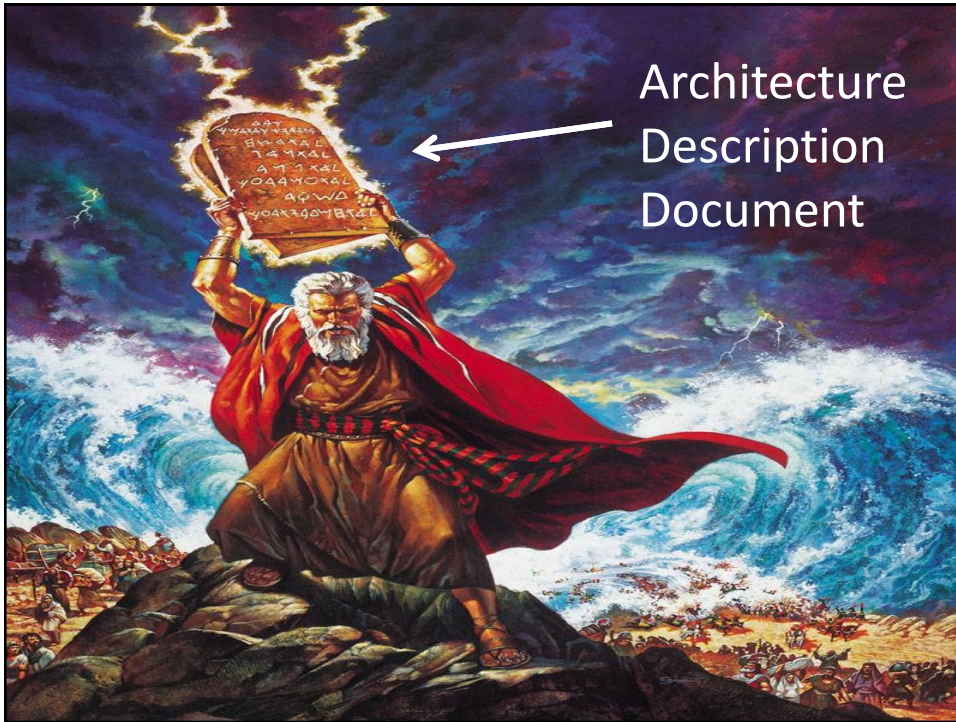
– Me, just now

13



14

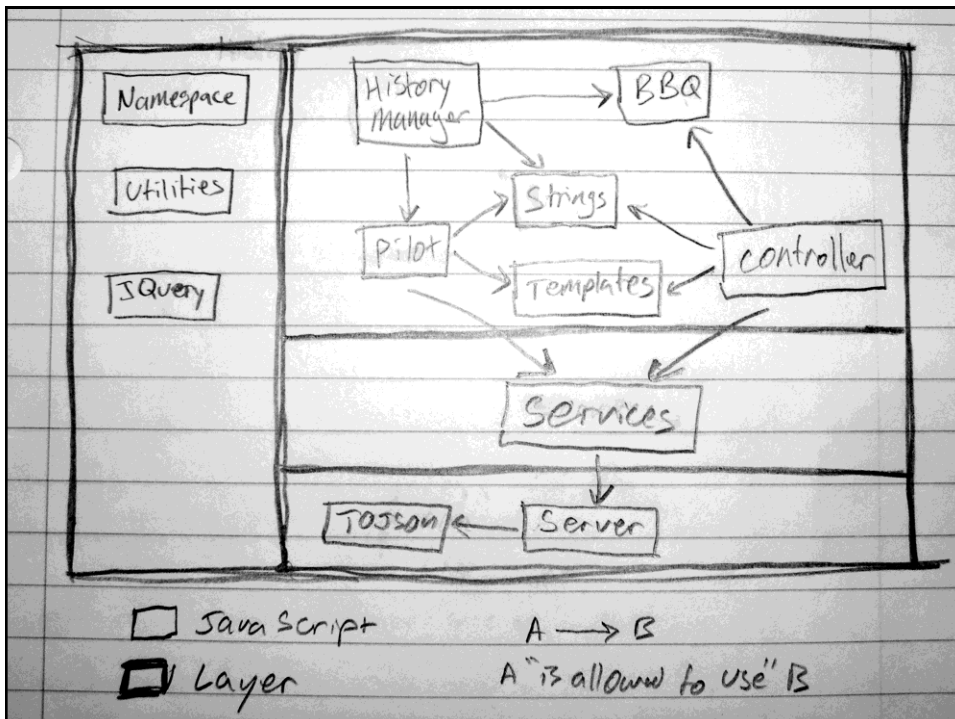


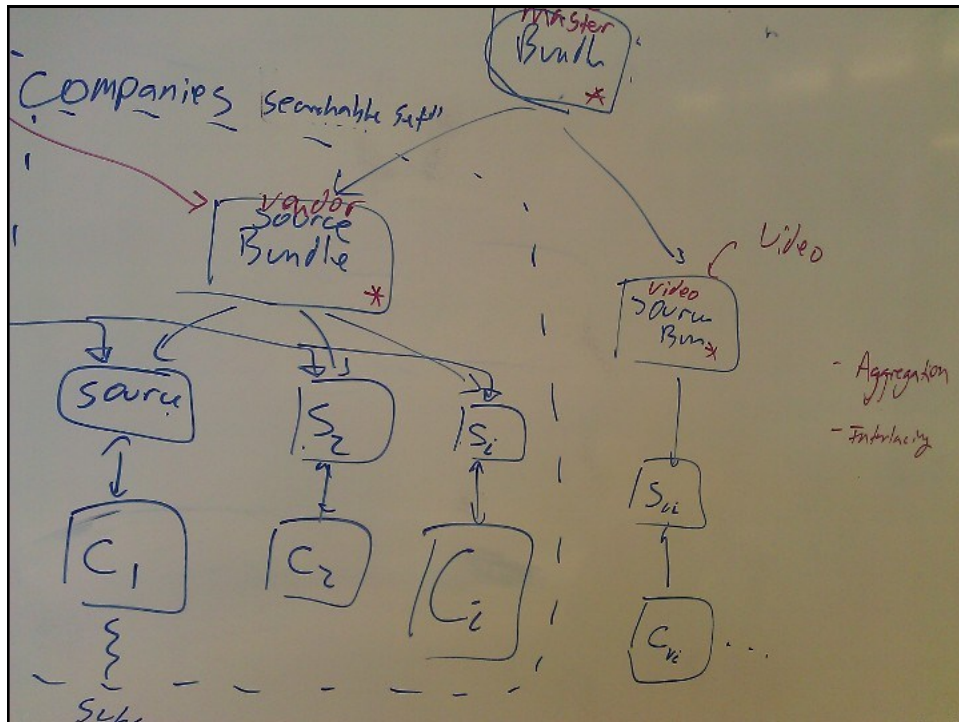


## Architecture Written in Stone...









Promoted in Ideal Representations

Visibility  
Changeability  
Understandability

Talk is cheap.

(That's why it's so useful!)

21

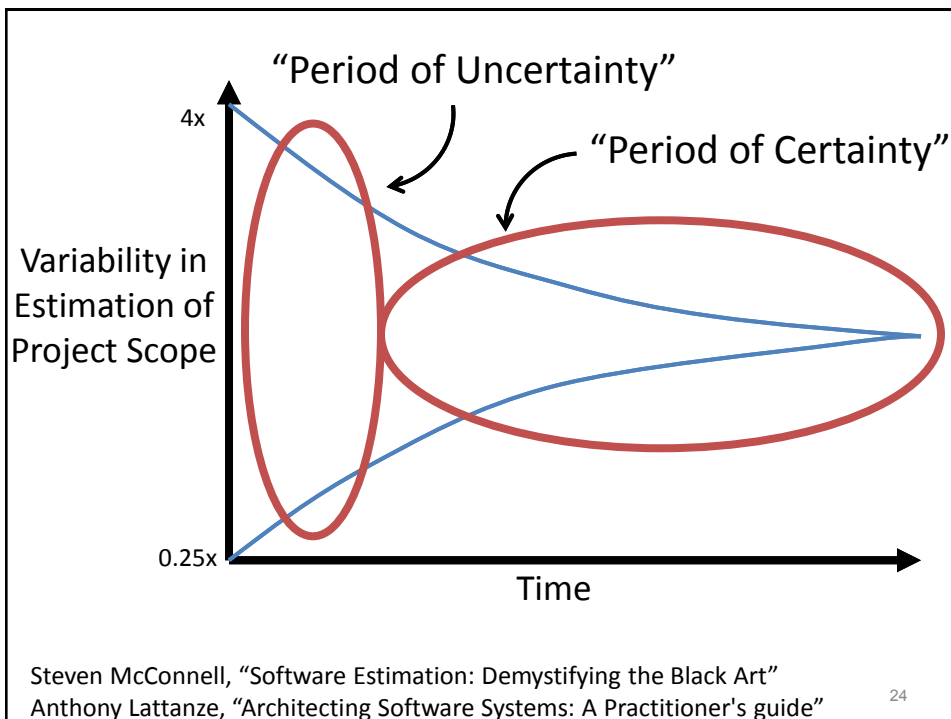
**MIDWAY IN THE SOFTWARE  
LIFECYCLE**

22

*“What did we decide last week  
about this part of the system?”*

– My team,  
a week after our amazing  
whiteboard discussion

23



24

## Cusp of Uncertainty/Certainty

Shift from **exploration**  
to **construction**.

25

The Period of  
Optimistic Certainty

26

## Promoted in Ideal Representations

Visibility  
Changeability  
Understandability  
Guide-ability

27

## Two Example Techniques

System Metaphor  
Architecture Haiku

28

# System Metaphor (Refreshed)

Michael Keeling, "Making Metaphors that Matter," Agile2011 Experience Report  
Michael Keeling, "Guidelines for the System Metaphor"

<http://neverletdown.net/2011/08/guidelines-for-the-system-metaphor/>

29



30



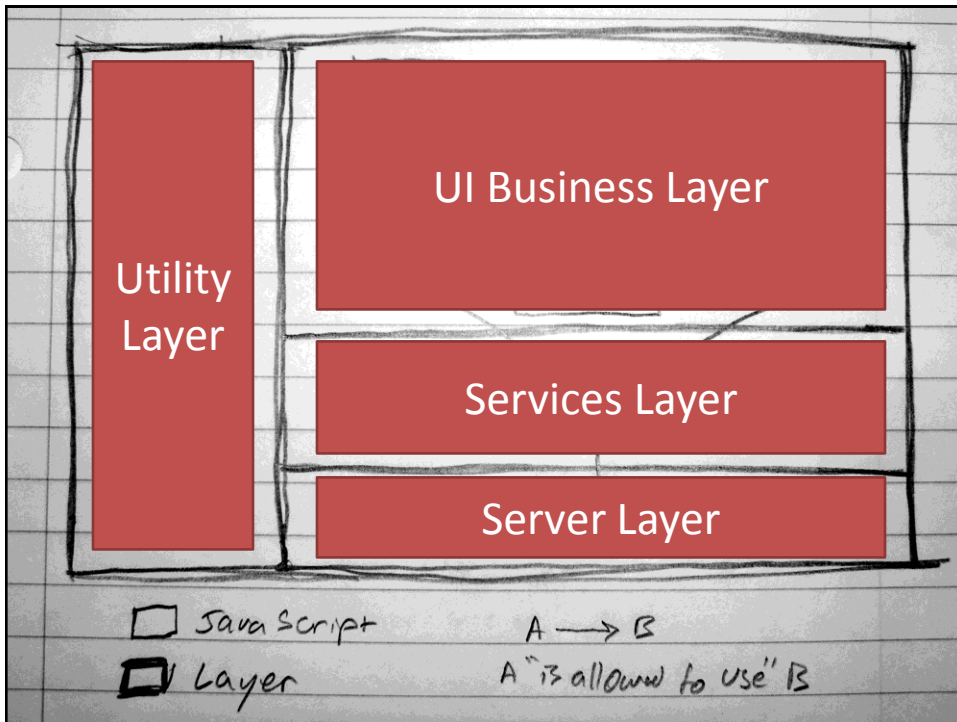
## A Good Metaphor...

1. Represents a single view.
2. Deals with only one type of structure.
3. Gives clear guidance concerning design decisions.
4. Sheds light on system properties.
5. Draws on shared experience.

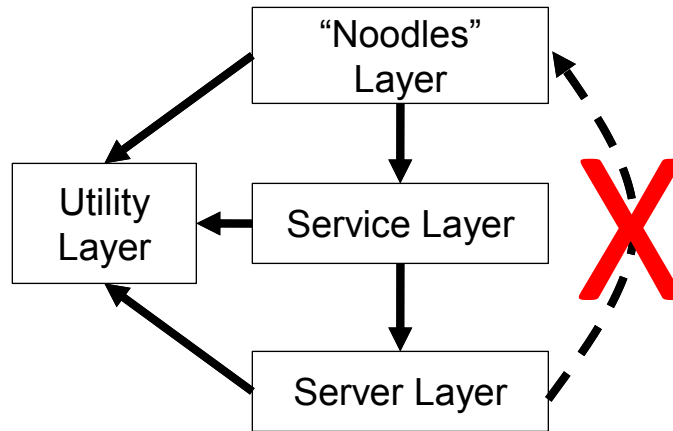
Corollary: Even a good metaphor still requires explanation.

31





# Web Client Overview (Static Perspective) "Bento Box"



## Legend

□ Layer     $A \rightarrow B$     A is allowed to use elements in B

Vocabulary  
Representation  
Reference Point  
+ Information Payload

---

System Metaphor

# Architecture Haiku

George Fairbanks, "Architecture Haiku,"

<http://rhinoresearch.com/content/speaking-boulder-java-user-group-7-sept-2010-architecture-haiku>

37

## Architecture Haiku includes

- Solution description
- Summary of architectural drivers
- Quality attribute priorities
- Design rationale and trade-offs
- Architectural styles/patterns used
- Diagrams

All on a **single** page!

38

# A Recent Haiku at Vivísimo...

## MegaVista Guides Search

A publicly consumed search solution based on the Velocity platform that will help users find companies of interest.

### Business Drivers

- Better search encourages listed companies to purchase advertising, metadata supplements, paid listings.
- Faster results for users and advanced search options (such as refinement, and spelling suggest) brings more users to the site.

### Key Trade-offs

- **Crawl-ability over Maintainability** – Need to split the datasource across multiple search collections, distributed across several servers. Incidentally promotes **Scalability**.
- **Crawl-ability and Query-ability over Configurability** – Highly configurable system reduces speed of crawl and query.
- **Flexibility/Development Speed over Cost** – Stakeholders are not 100% on all features and have a strong desire to go live as soon as possible.
- **Modifiability over Maintainability** – IT will maintain the system and know databases not Velocity so when they make a change it should be detected and reflected.

### Architecture Styles/Patterns

3-tier (data, crawl, query)  
Source-Selector – promotes reliability  
Double Redundancy – promotes availability  
Virtual Documents – all data crawled  
Document Enqueue – for website data  
Collection Farm – promotes crawl-ability  
Geolocation Lookup – promotes maintainability/modifiability

### Top Quality Attributes

#### Crawl-ability > Query-ability > Scalability

- **Crawl-ability** – A batch of up to 900 metadata updates are published to the database and reflected in search within 3 minutes.
- **Query-ability** – An average sized result set can be calculated within 2 seconds of Velocity receiving the query.
- **Scalability** – The size of the datasource increases beyond current capacity and the system can be easily expanded to deal with this.

### Design Decisions with Rationale

- Database must be crawled as multiple views (vice JOINed) to avoid stressing it too much. Currently 1.6 million companies, each with dozens of metadata fields. Metadata joined via Virtual Document (company ID = vse-key).
- Company website must be crawled and website content made searchable under the company (i.e. return a single result with all content searchable as the same document). Website data joined with metadata via Virtual Document (company ID = vse-key).
- Full Recrawl / week, DB Refresh / 30 sec if no crawl running. Full recrawl will pick up changes made to company websites (external data). 30 sec refresh enables catch-up for metadata crawls on DB updated every 60 seconds at most.
- Collections divided by company ID. Dividing allows maximum crawl throughput, using company ID (e.g. collection A has ids 1-10) allows us to control partitioning.
- Paid listings and keyword weighting factors used for relevancy calculation will come from the database (not Velocity configuration). This will allow the business unit to make changes quickly using their existing tools.

39

# Vivísimo's Architecture Haiku Rules

- One page in Visio
- No font smaller than 10pt
- Should follow basic template
- Diagrams optional
  - Velocity Patterns Catalog
- Assume a requirements document exists
  - Customer communication, planning

40

# Haiku + System Metaphor...

## MegaVista Guides Search

A publicly consumed search solution based on the Velocity platform that will help users find companies of interest.

### Business Drivers

- Better search encourages listed companies to purchase advertising, metadata supplements, paid listings.
- Faster results for users and advanced search options (such as refinement, and spelling suggest) brings more users to the site.

### Top Quality Attributes

#### Crawl-ability > Query-ability > Scalability

- Crawl-ability – A batch of up to 900 metadata updates are published to the database and reflected in search within 3 minutes.
- Query-ability – An average sized result set can be calculated within 2 seconds of Velocity receiving the query.
- Scalability – The size of the data source increases beyond current

### Key Trade-offs

## Architecture Styles/Patterns

3-tier (data, crawl, query)

Source-Selector – promotes reliability

Double Redundancy – promotes availability

Virtual Documents – all data crawled

Document Enqueue – for website data

Collection Farm – promotes crawl-ability

Geolocation Lookup – promotes

maintainability/modifiability

### Rationale

Multiple views (vice JOINed) to jointly 1.6 million companies, fields. Metadata joined via vse-key).  
 Crawl and website content made to return a single result with all document). Website data document (company ID = vse-  
 / 30 sec if no crawl running. made to company websites enables catch-up for metadata records at most. ID. Dividing allows maximum ID (e.g. collection A has 1-  
 ing. ing factors used for relevancy database (not Velocity business unit to make ing tools.

41

# LATE IN THE SOFTWARE LIFECYCLE

42

*“The easiest time to document the software architecture is at the end of the project...”*

*All the hard decisions have already been made!”*

– Me, about 11  
minutes ago

43

## Change Becomes Expensive

Shift from **Creating** to  
**Releasing** and  
**Maintaining.**

44



Now is the Time to Document

Most Knowledge  
Least Rework  
Time to Move on

45

Architecture oral history  
collapses without a team to  
keep the culture alive.

46

# CREATING AN ARCHITECTURE ORAL HISTORY

47

## Core Assumption

Architecture oral history requires that the team is both **willing and able** to retell the stories and keep the oral history alive.

48

## Essential Knowledge

- Basics of architectural drivers
- Simple quality attribute scenarios
- Static, dynamic, physical structures
- Basics of architectural styles/patterns
- Identify (and evaluate) trade-offs

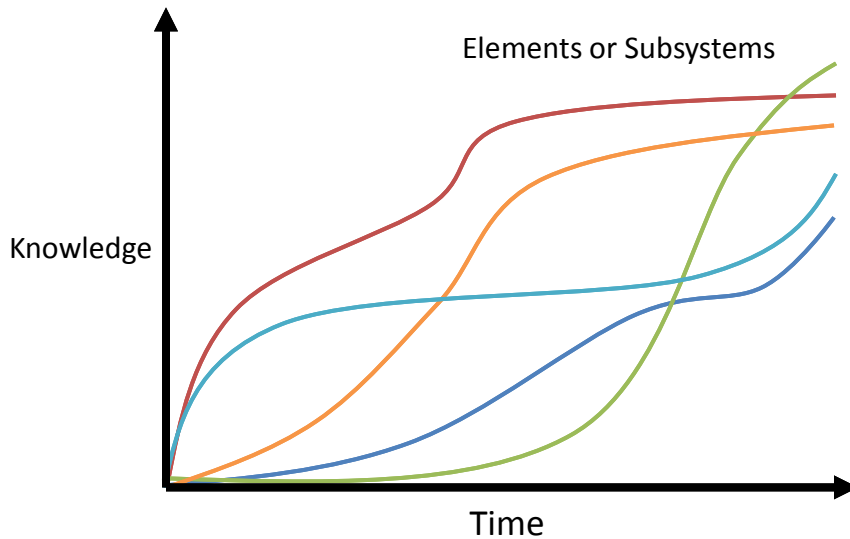
49

## Mileage May Vary - Adjust Accordingly

Size of Project  
Size of Team  
Collocated vs. Distributed  
Team Experience

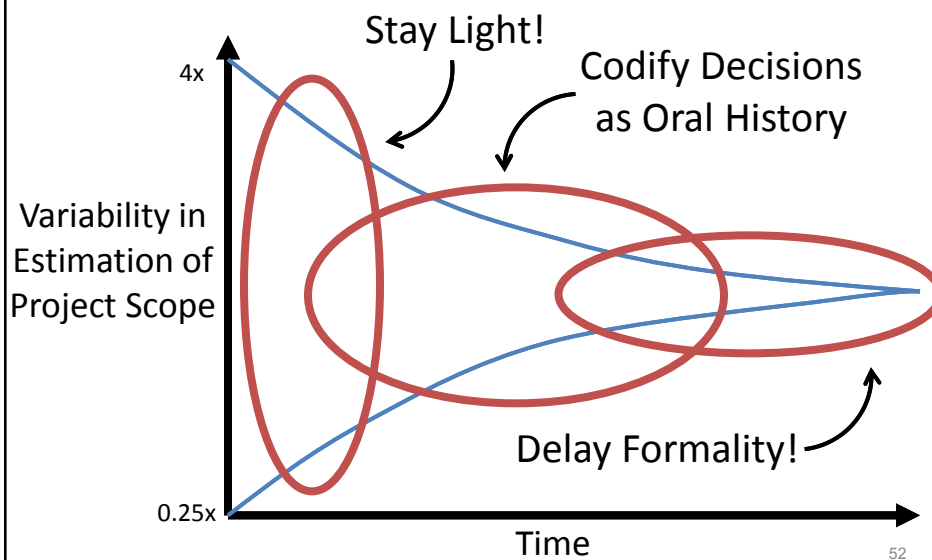
50

## Evolution is not Simple...



51

## To Review...



52

We're Already Doing This...

Discussing  
Whiteboarding  
Sketching  
Talking in Metaphors

53

Thank you!

Michael Keeling  
@michaelkeeling  
[mkeeling@neverletdown.net](mailto:mkeeling@neverletdown.net)  
<http://neverletdown.net>

54