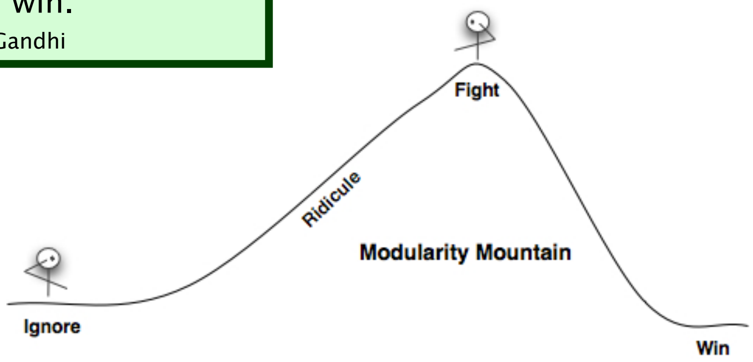
Modularity, Agility, and Architecture’s Paradox

AOSD 2011
March 25th, 2011

Kirk Knoernschild
pragkirk@kirkk.com

Modularity, Agility and Architecture’s Paradox

> “First they ignore you, then they ridicule you, then they fight you, then you win.”
>
> -- Mahatma Gandhi



Modularity is disruptive and will transform how enterprise applications are designed, developed, and managed!

Modularity, Agility and Architecture’s Paradox

1972

(or a bit before)

Modularity, Agility and Architecture’s Paradox

- → Complexity
- → Architectural Agility
- → Paradox
- → Modularity

Modularity, Agility and Architecture’s Paradox

Complexity

Why is software so complex?

How do we tame complexity?

What’s the role of modularity?

Complexity

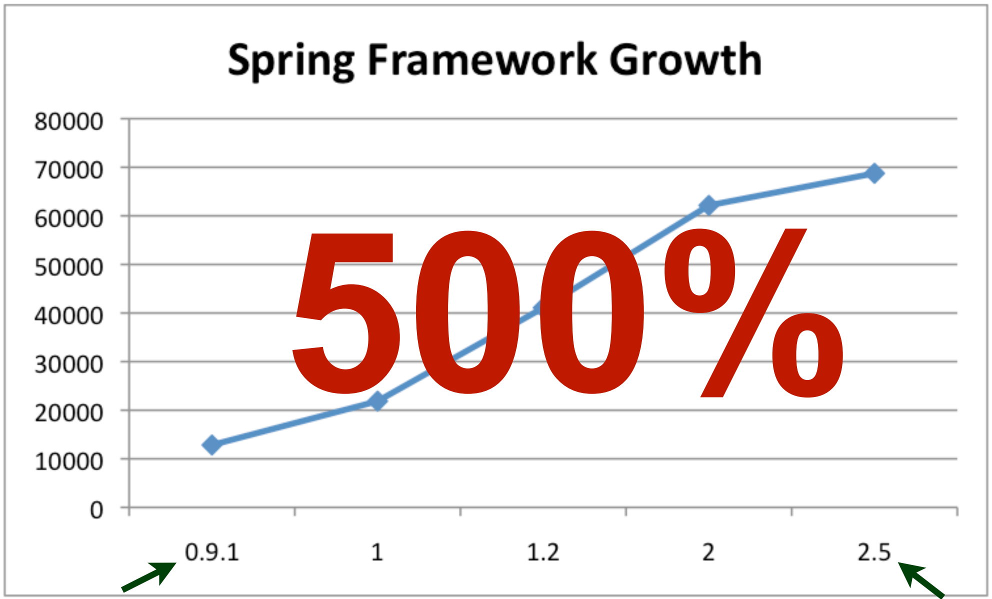
- 120 billion loc in 1990
- 250 billion loc in 2000
- loc doubles every 7 years
- 50% of development time spent understanding code
- 90% of software cost is maintenance & evolution

Source: http://users.jyu.fi/~koskinen/smcosts.htm

2017

Perspective: Not only double the past 7 years, but more than total amount ever written combined!

Complexity



500%

2002

2008

Gartner

Complexity

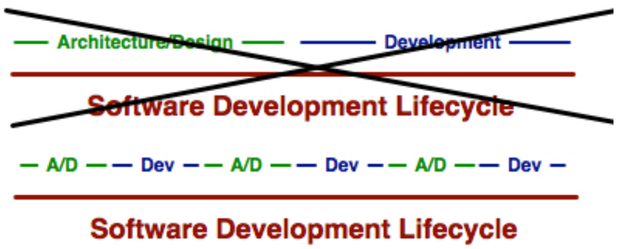
Lehman's Law

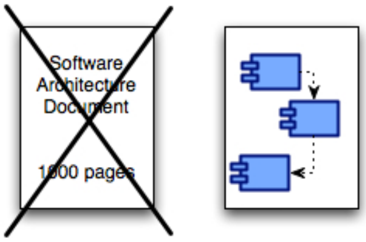
As a system evolves, its complexity increases unless work is done to maintain or reduce it.

www.lumaxart.com/

Gartner

Complexity


~~Architecture/Design~~
~~Development~~
~~Software Development Lifecycle~~
A/D
Dev
A/D
Dev
A/D
Dev
Software Development Lifecycle



- BDUF and BAUF
- Excessive documentation
- Resisting Change

We are often asked to design solutions to problems that require knowledge we currently do not possess.

Complexity

Gall's Law

A complex system that works is invariably found to have evolved from a simple system that worked. A complex system designed from scratch never works and cannot be patched up to make it work. You have to start over, beginning with a working simple system.

Modularity, Agility and Architecture’s Paradox

Architectural Agility

What is architecture?

What is the goal of architecture?

What is architectural agility?

Architectural Agility

- What is Architecture?
  - An architecture is the set of significant decisions about the organization of a software system, the selection of the structural elements and their interfaces by which the system is composed, together with their behavior as specified in the collaborations among those elements, the composition of these structural elements and behavioral elements into progressively larger subsystems, and the architecture style that guides this organization -- these elements and their interfaces, their collaborations, and their composition.

Source: Kruchten: The Rational Unified Process. Also cited in Booch, Rumbaugh, and Jacobson: The Unified Modeling Language User Guide, Addison-Wesley, 1999

Architectural Agility

- What is Architecture?
  - In most successful software projects, the expert developers working on that project have a shared understanding of the system design. This shared understanding is called ‘architecture.’ This understanding includes how the system is divided into components and how the components interact through interfaces. These components are usually composed of smaller components, but the architecture only includes the components and interfaces that are understood by all the developers...Architecture is about the important stuff. Whatever that is.

Source: Fowler, Martin. IEEE Software, 2003. “Who Needs an Architect?” A quote from Ralph Johnson on the XP mailing list.

Architectural Agility

- What is Architecture?
  - 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.

Source: ANSI/IEEE Std 1471-2000

Architectural Agility

- What is Architecture?
  - A formal description of a system, or a detailed plan of the system at component level to guide its implementation
  - The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time.

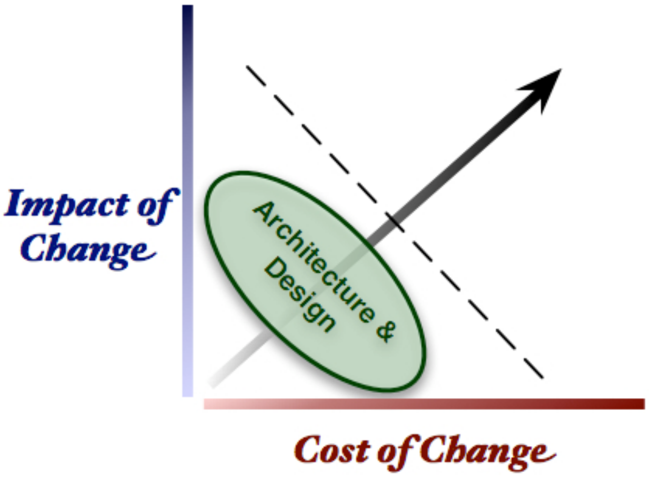
Source: TOGAF - http://www.opengroup.org/architecture/togaf8-doc/arch/chap01.html

Architectural Agility

- What is Architecture?
  - Architecture embodies the critical design decisions that typify a system.
    - Relates to cost of change, organizational structure, structure of code, capabilities of a system, etc.
  - The significance of decisions needs to be understood and assessed
    - A heavy-weight approach is likely to reduce understanding and our ability to assess

Source: QCON London Presentation by James Coplien & Kevlin Henney title Agile Architecture Is not Fragile Architecture - http://www.infoq.com/presentations/Agile-Architecture-Is-Not-Fragile-Architecture-James-Coplien-Kevlin-Henney

Architectural Agility

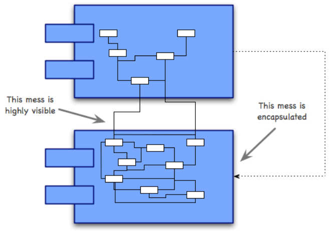


This i
impor

What if we were able to reduce the impact and cost of change?

We need to eliminate architecture!

Architectural Agility



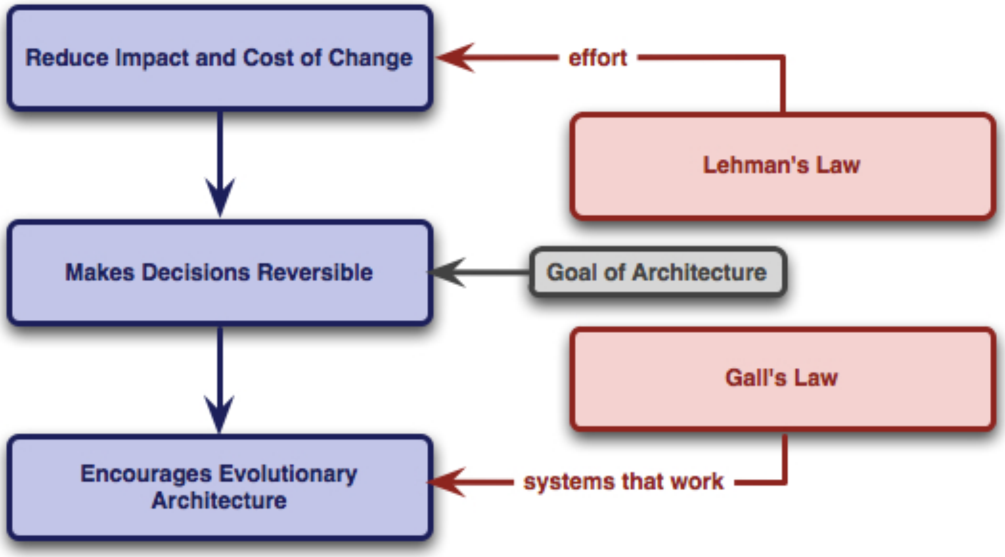
Reversible Decisions can be easily changed because the architecture is able to accommodate change.

Irreversible Decisions are not easily changed because they are too expensive or resource intensive.

First and foremost, we should try to make most decisions reversible, so they can be made and then easily changed.
-- "*Implementing Lean Software Development: From Concept to Cash*"

Irreversible Decisions should be made as late as possible!

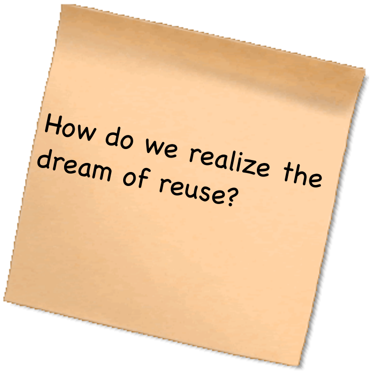
Architectural Agility


Reduce Impact and Cost of Change
effort
Lehman's Law
Makes Decisions Reversible
Goal of Architecture
Gall's Law
Encourages Evolutionary Architecture
systems that work

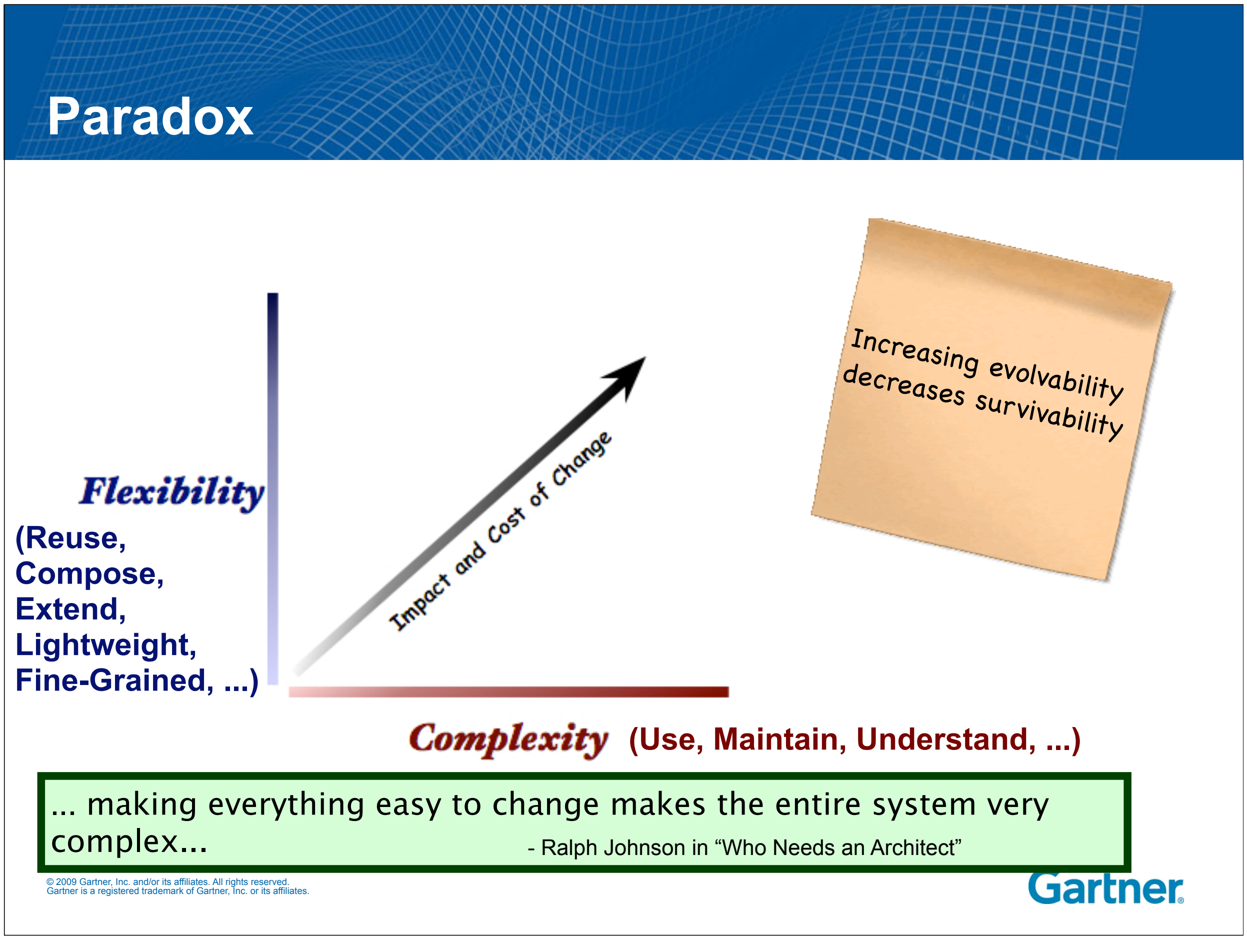
Modularity, Agility and Architecture's Paradox

Paradox

How do we design more flexible software systems?


How do we realize the dream of reuse?

Are we doomed to failure?



Paradox

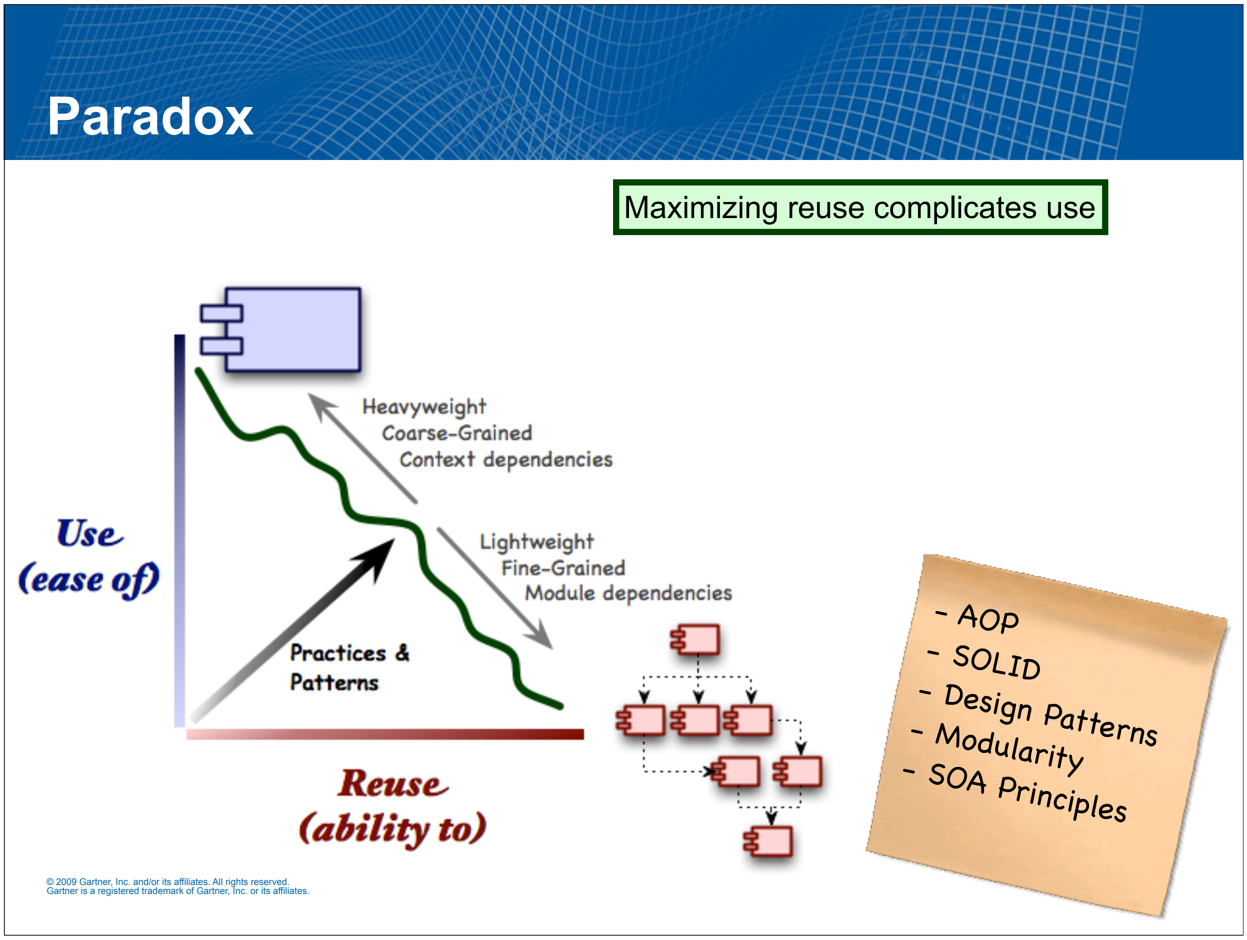
Increasing evolvability decreases survivability

Flexibility (Reuse, Compose, Extend, Lightweight, Fine-Grained, ...)

Impact and Cost of Change

Complexity (Use, Maintain, Understand, ...)

... making everything easy to change makes the entire system very complex...
- Ralph Johnson in “Who Needs an Architect”



Paradox

Maximizing reuse complicates use

Use (ease of)

Heavyweight
Coarse-Grained
Context dependencies

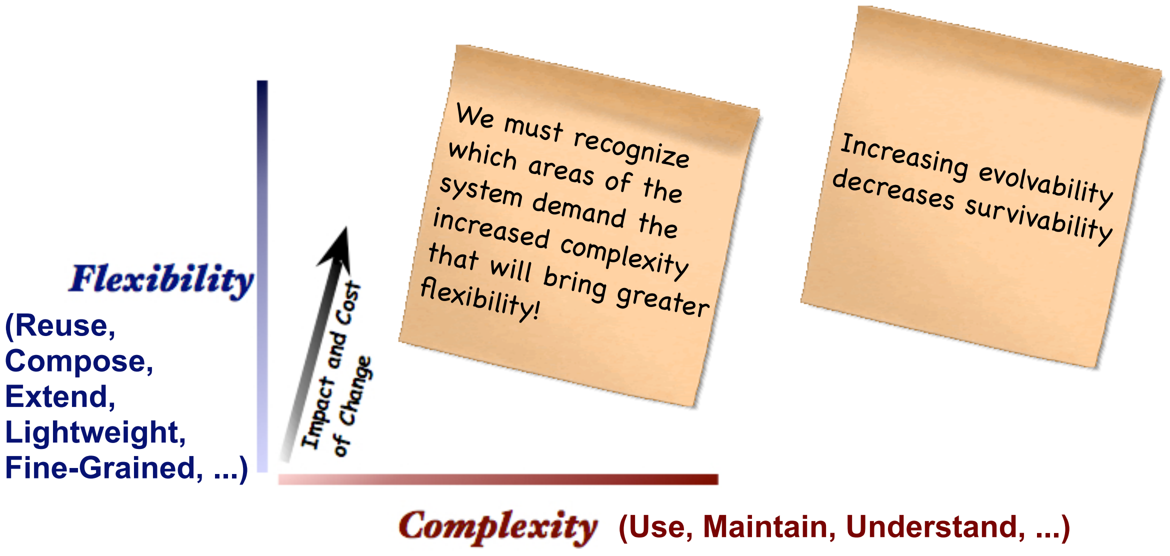
Lightweight
Fine-Grained
Module dependencies

Practices & Patterns

Reuse (ability to)

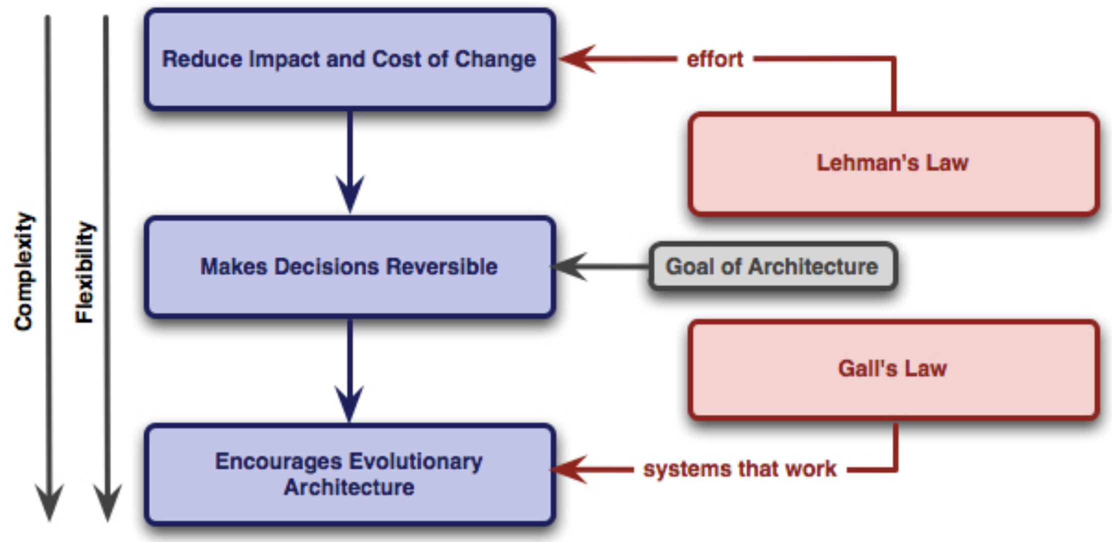
- AOP
- SOLID
- Design Patterns
- Modularity
- SOA Principles

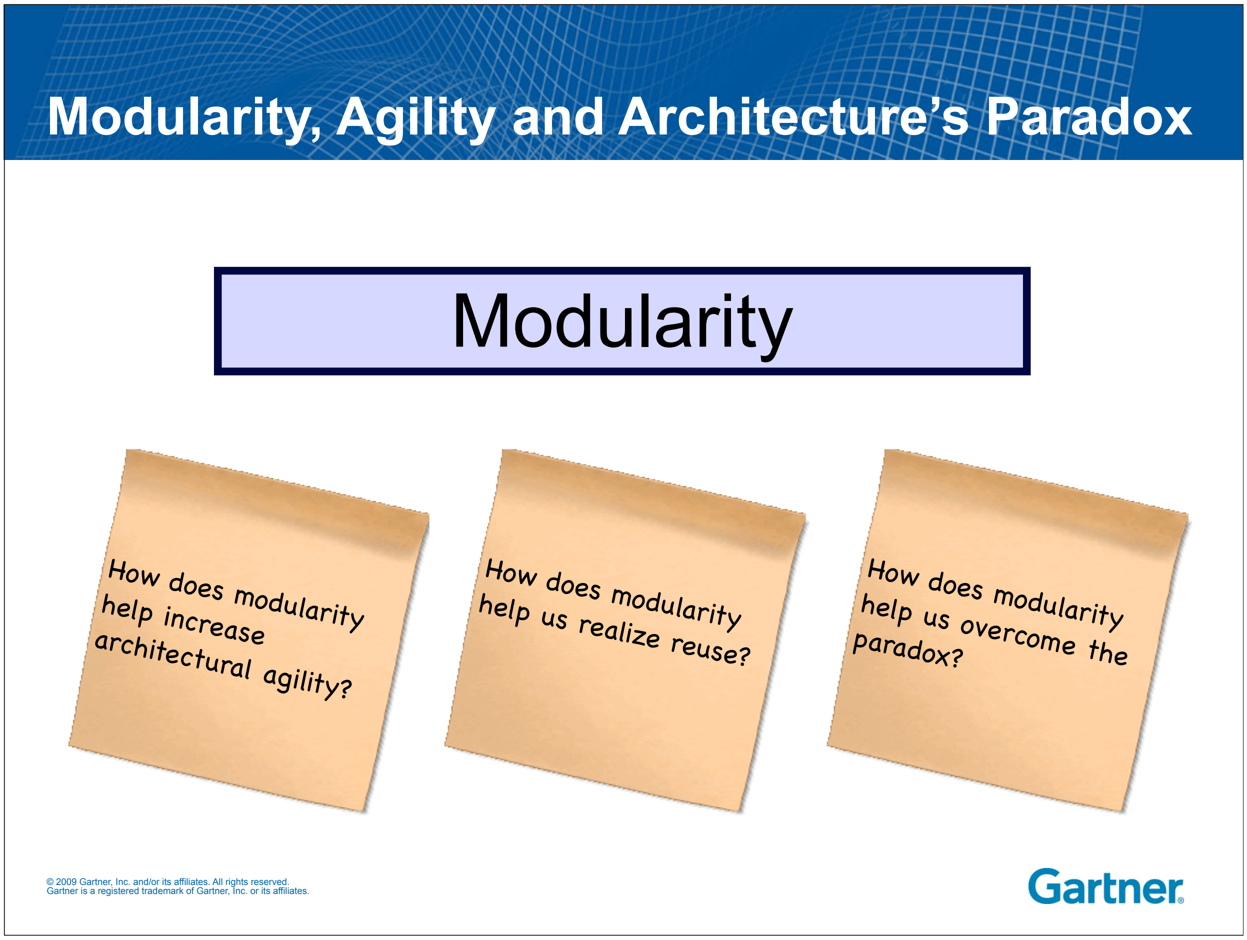
Paradox


Flexibility
(Reuse, Compose, Extend, Lightweight, Fine-Grained, ...)
Impact and Cost of Change
Complexity (Use, Maintain, Understand, ...)
We must recognize which areas of the system demand the increased complexity that will bring greater flexibility!
Increasing evolvability decreases survivability

> ... making everything easy to change makes the entire system very complex...
>
> - Ralph Johnson in "Who Needs an Architect"

Paradox


Complexity
Flexibility
Reduce Impact and Cost of Change
effort
Lehman's Law
Makes Decisions Reversible
Goal of Architecture
Gall's Law
Encourages Evolutionary Architecture
systems that work



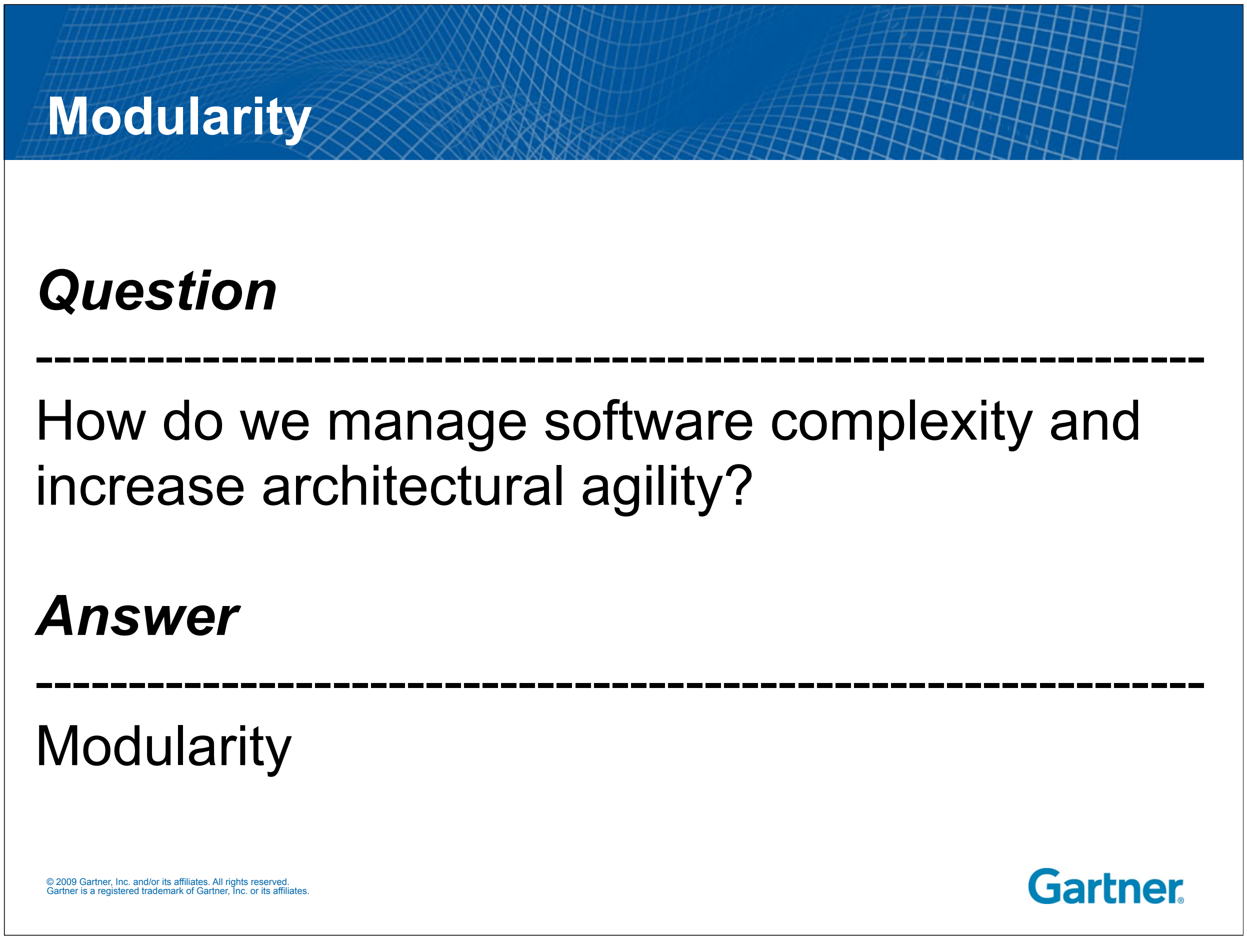
Modularity, Agility and Architecture’s Paradox

Modularity

How does modularity help increase architectural agility?

How does modularity help us realize reuse?

How does modularity help us overcome the paradox?



Modularity

Question

How do we manage software complexity and increase architectural agility?

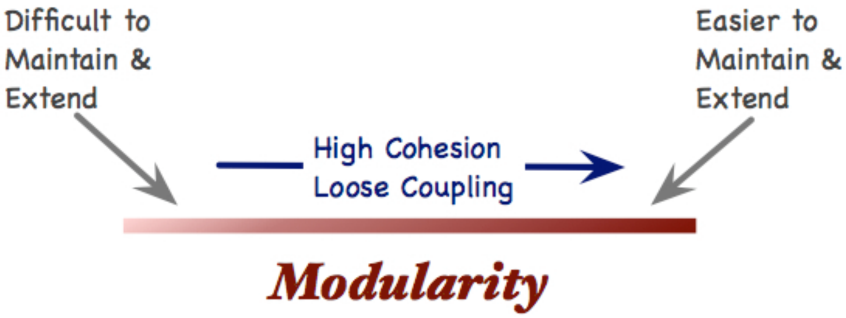
Answer

Modularity

Modularity

- reuse
- reduce complexity
- ease maintenance
- increase extensibility

Increases architectural agility!



Difficult to Maintain & Extend

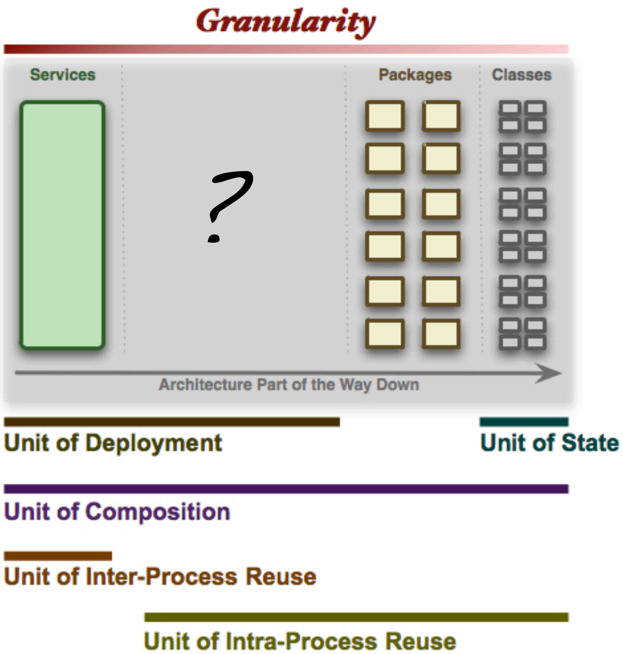
Easier to Maintain & Extend

High Cohesion Loose Coupling

Modularity

Umm...we can already do this with objects, aspects, methods, and services!

Modularity



Granularity

Services

?

Packages

Classes

Architecture Part of the Way Down

Unit of Deployment

Unit of State

Unit of Composition

Unit of Inter-Process Reuse

Unit of Intra-Process Reuse

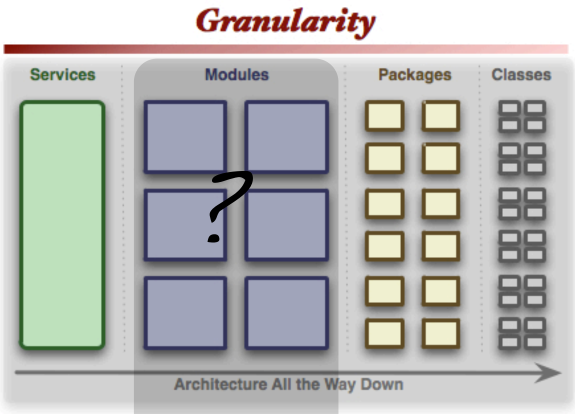
Modularity

Photo courtesy of: http://www.flickr.com/photos/mybloodyself/1108834349/

What does architecture have to do with turtles?

"You're very clever, young man, very clever", said the old lady. "But it's turtles all the way down!"
-- A Brief History of Time

Modularity



Unit of Deployment

Unit of State

Unit of Composition

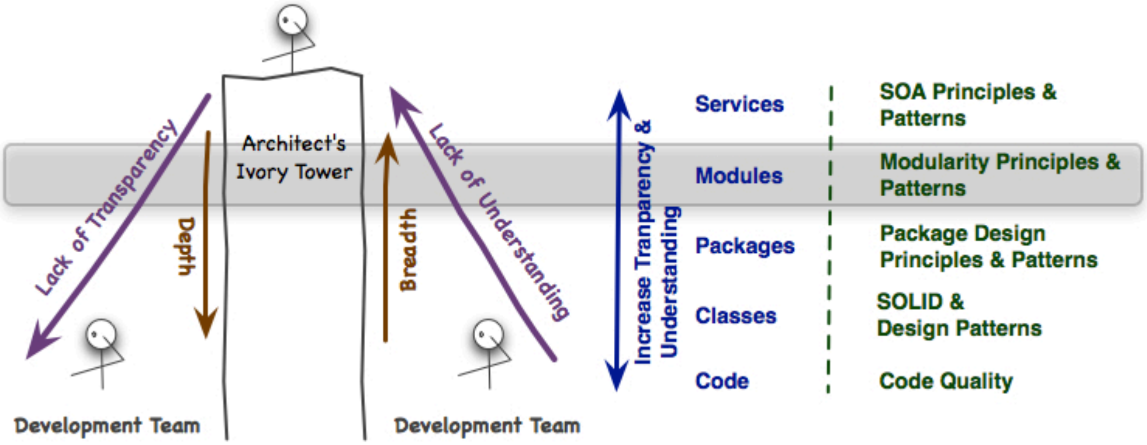
Unit of Inter-Process Reuse

Unit of Intra-Process Reuse

Reuse Release Equivalence:
Unit of reuse is the unit of release!

Gartner

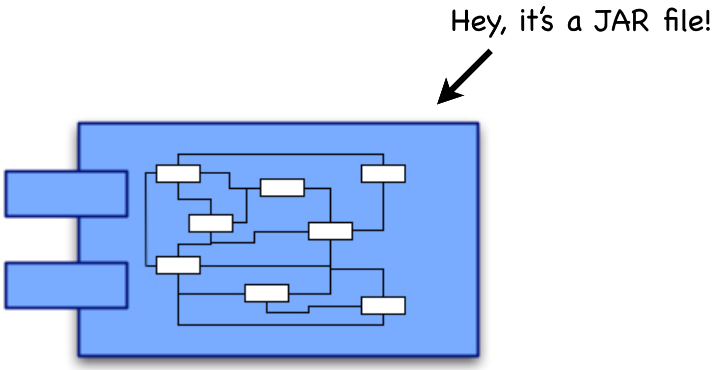
Modularity



Services	SOA Principles & Patterns
Modules	Modularity Principles & Patterns
Packages	Package Design Principles & Patterns
Classes	SOLID & Design Patterns
Code	Code Quality

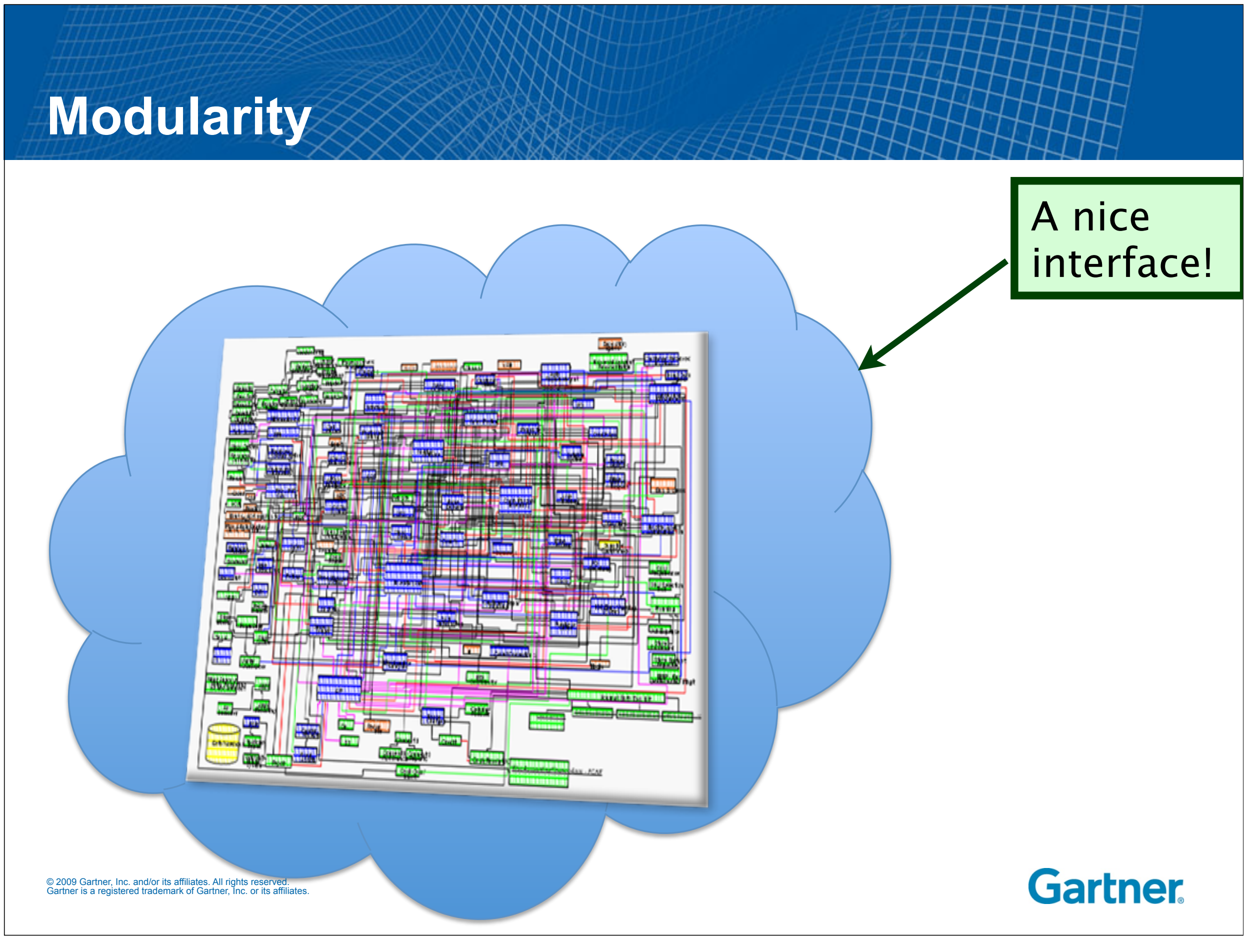
Adapted from http://www.rendell.org/jam/upload/2009/1/tower-12054835.jpg

Modularity



- unit of reuse
- unit of composition
- unit of deployment
- unit of management

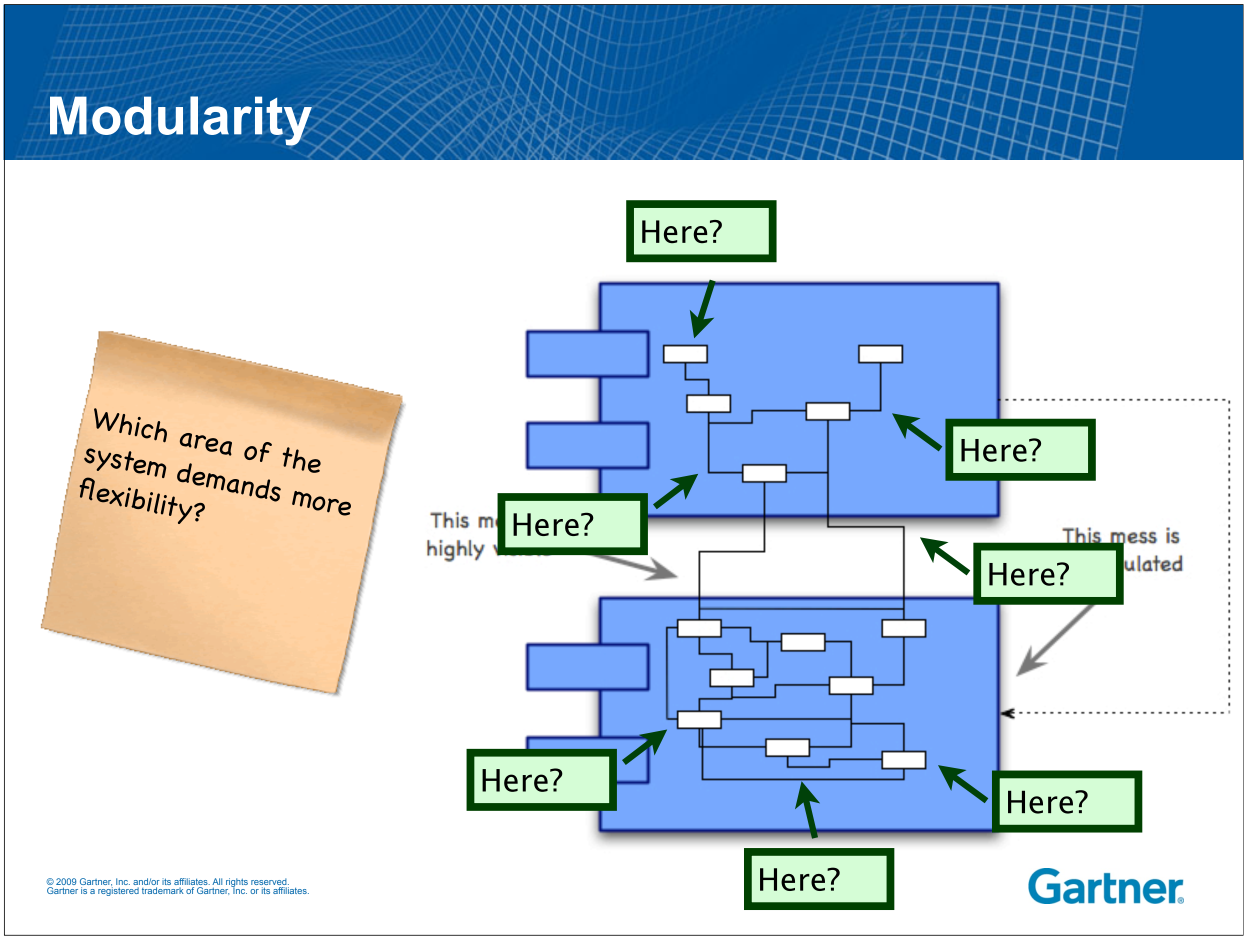
A module system provides a runtime environment for modules



Modularity

A nice interface!

Gartner



Modularity

Which area of the system demands more flexibility?

Here?

Here?

Here?

Here?

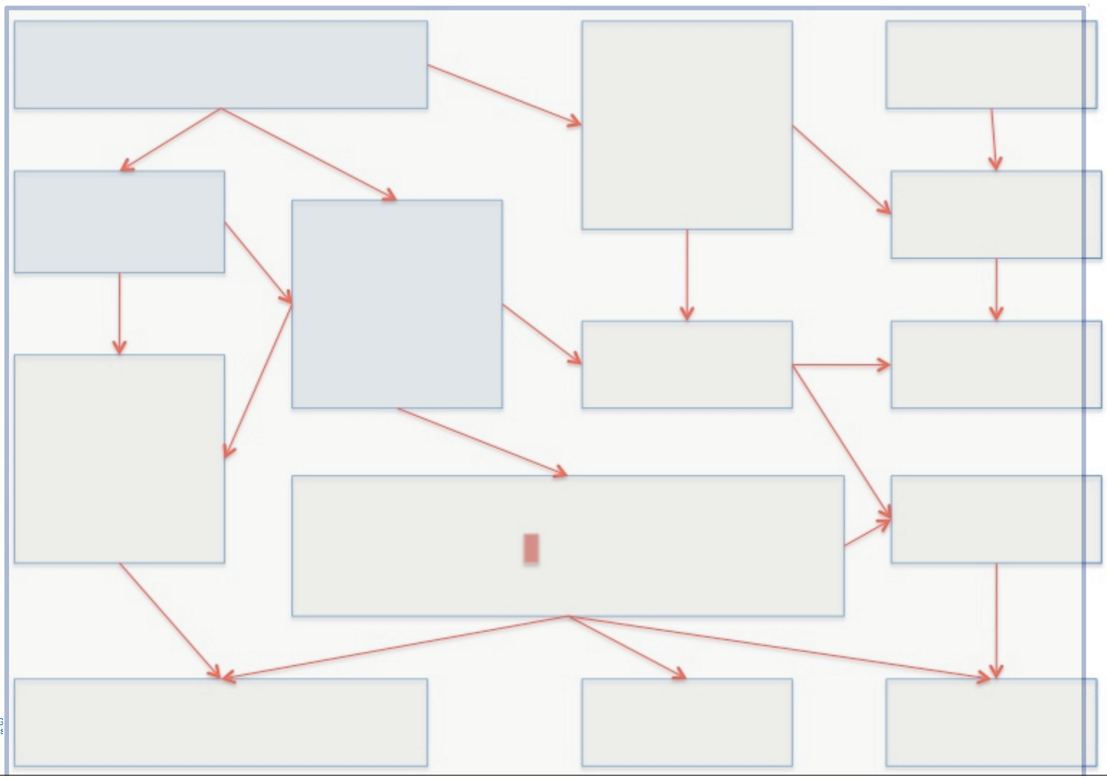
Here?

Here?

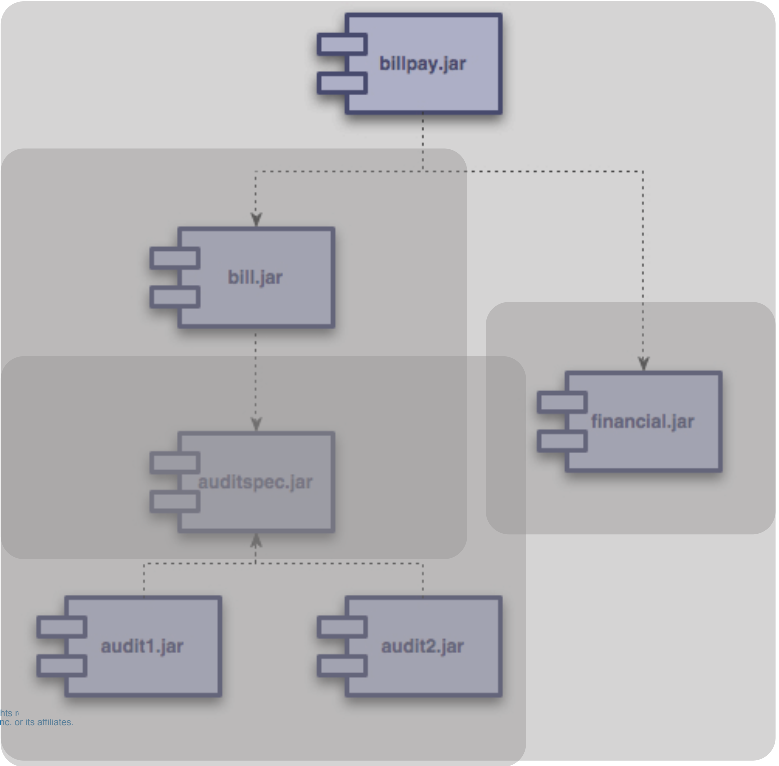
Here?

Gartner

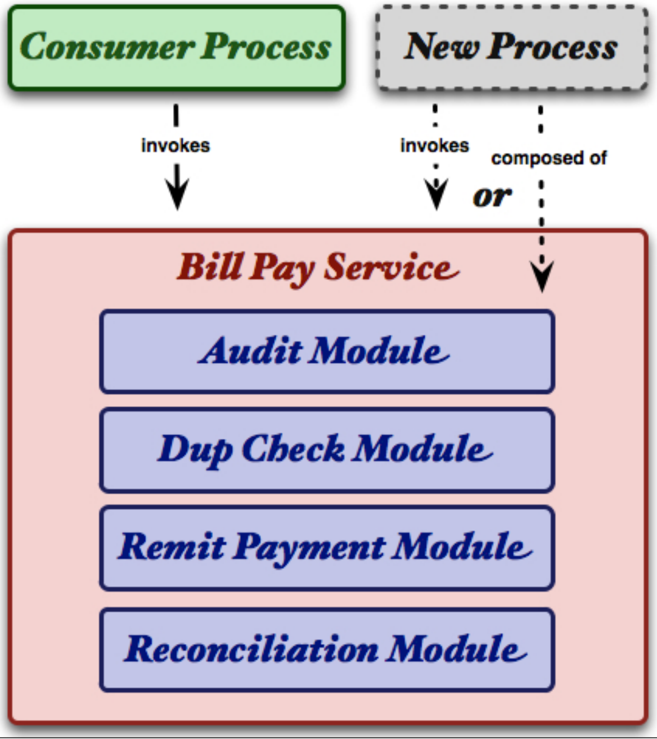
Modularity



Modularity



Modularity



Consumer Process

invokes

New Process

invokes

composed of

or

Bill Pay Service

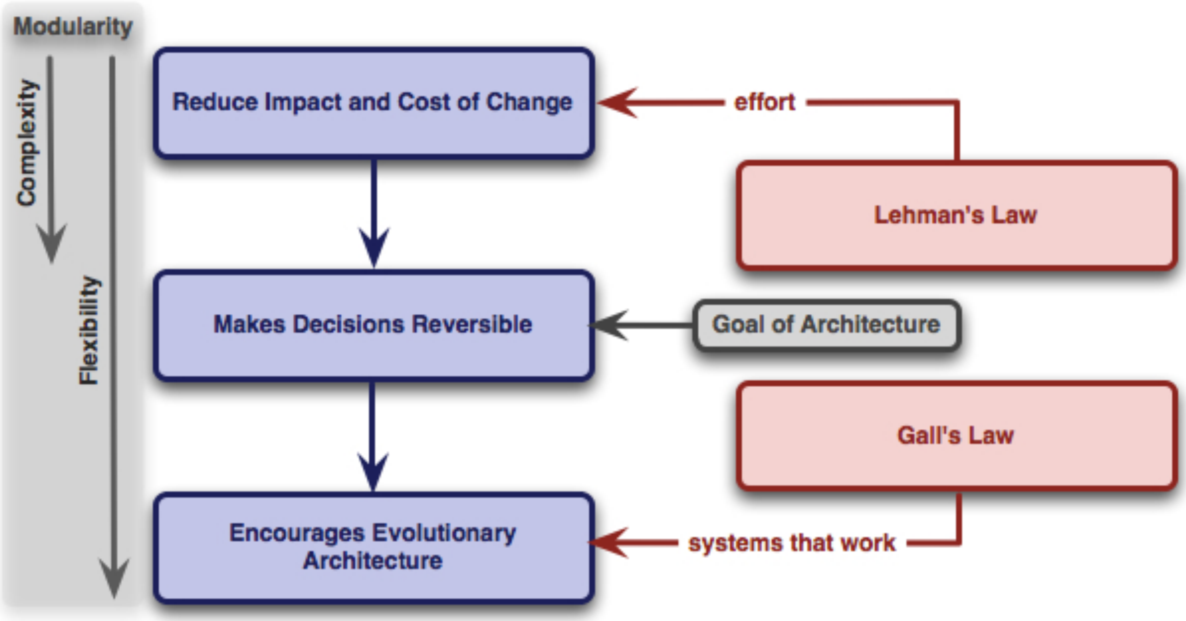
Audit Module

Dup Check Module

Remit Payment Module

Reconciliation Module

Modularity



Modularity

Complexity

Flexibility

Reduce Impact and Cost of Change

effort

Lehman's Law

Makes Decisions Reversible

Goal of Architecture

Gall's Law

Encourages Evolutionary Architecture

systems that work

Modularity

Infrastructure
- Runtime platform support helps enforce modular architecture.

Programming Model
- The frameworks and technologies that allow us to create modular software

Design Paradigm
- The techniques used to identify and create the right set of modules

The Design Paradigm
- What's the right granularity for a module?
- What the right weight for a module?

Modularity

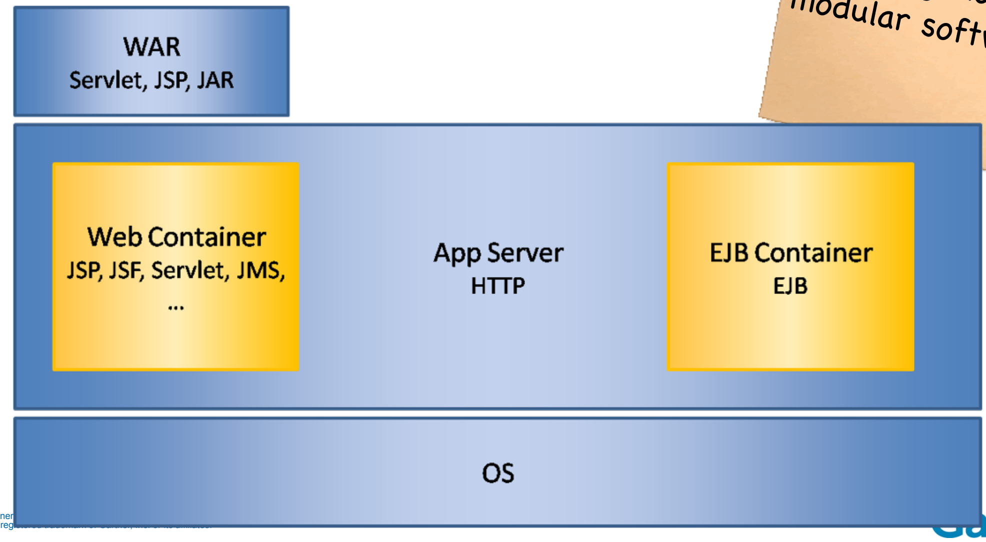
Few teams are designing modular software systems today!

POLL:

Question:	***Response***:
- How many design class relationships?	98%
- How many design package relationships?	25%
- How many design service relationships?	75%
- How many design module (JAR, Assembly) relationships?	< 10%

Modularity

Platforms discourage modularity!



WAR
Servlet, JSP, JAR

Web Container
JSP, JSF, Servlet, JMS,
...

App Server
HTTP

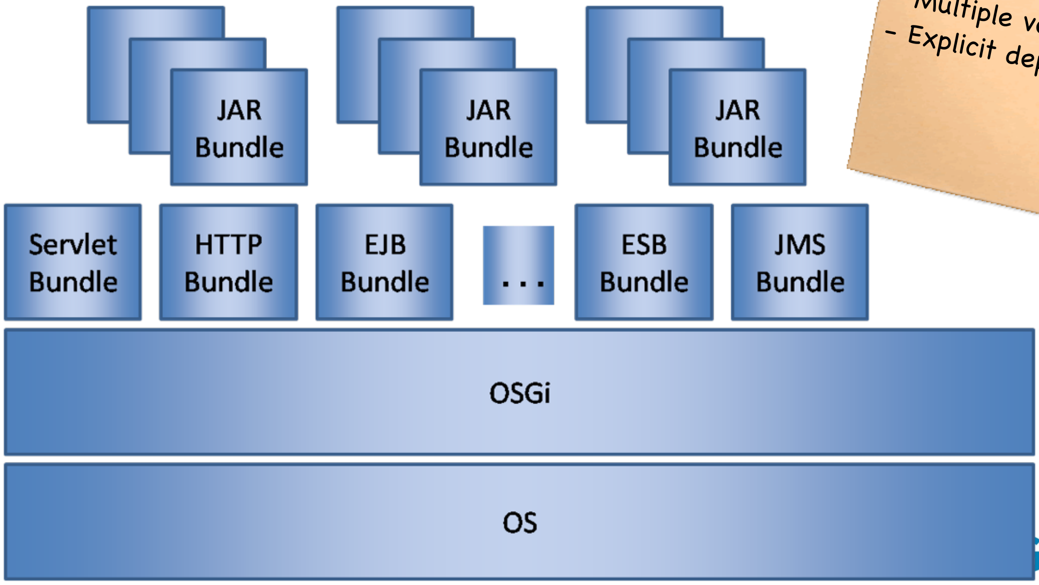
EJB Container
EJB

OS

Why aren't we designing more modular software?

Modularity

This is the next generation application platform!



JAR Bundle

JAR Bundle

JAR Bundle

Servlet Bundle

HTTP Bundle

EJB Bundle

. . .

ESB Bundle

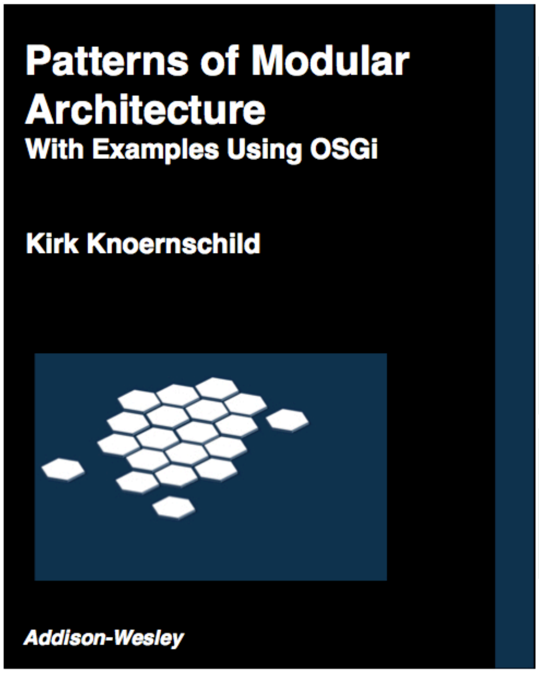
JMS Bundle

OSGi

OS

- Dynamic deployment
- Multiple versions
- Explicit dependencies

Modularity, Agility and Architecture’s Paradox



Patterns of Modular Architecture
With Examples Using OSGi

Kirk Knoernschild

Addison-Wesley

Available September 2011

These, and many other ideas, are discussed in my upcoming book:

“***Patterns of Modular Architecture***”

Draft manuscript available at:

http://modularity.kirkk.com

Code available at:

https://github.com/pragkirk/poma

Modularity, Agility and Architecture’s Paradox

- Additional Resources
  - *http://techdistrict.kirkk.com* - Kirk’s blog with lots of content on modularity.
  - *Patterns of Modular Architecture* - Book in Progress
    - http://modularity.kirkk.com/
  - *http://www.osgi.org* - OSGi HomePage
  - *OSGi in Practice* by Neil Bartlett
    - http://neilbartlett.name/blog/osgibook/
  - *Modular Java* by Craig Walls
  - *OSGi and Equinox: Creating Highly Modular Java Systems* by Jeff McAffer, et. al.
  - Tried to develop a modular architecture without OSGi?
    - JarAnalyzer - http://code.google.com/p/jaranalyzer/