SE 577
Software Architecture

Intro to Software Architecture

What is Software 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.

ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems.

What is Software Architecture? (cont'd)

A software system architecture comprises:

- A collection of software and system components, connections, and constraints.
- A collection of system stakeholders' need statements.
- A rationale which demonstrates that the components, connections, and constraints define a system that, if implemented, would satisfy the collection of system stakeholders' need statements.

Boehm, et al., 1995

What is Software Architecture? (cont'd)

- 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 and behavioral elements into progressively larger subsystems, and the architectural style that guides this organization---these elements and their interfaces, their collaborations, and their composition

P. Kruchten: The Rational Unified Process.

What is Software Architecture? (cont'd)

- As the size and complexity of software systems increases, the design problem goes beyond the algorithms and data structures of the computation: designing and specifying the overall system structure emerges as a new kind of problem.

  Structural issues include gross organization and global control structure; protocols for communication, synchronization, and data access; assignment of functionality to design elements; physical distribution; composition of design elements; scaling and performance; and selection among design alternatives.

  This is the software architecture level of design.

  Garlan and Shaw, 1999

Every system has an architecture, regardless if it was planned or not planned

- Every system comprises elements and relations among them to support some type of reasoning.
- But the architecture may not be known to anyone.
  - Perhaps all of the people who designed the system are long gone
  - Perhaps the documentation has vanished (or was never produced)
  - Perhaps the source code has been lost (or was never delivered)
- An architecture can exist independently of its description or specification.
- Documentation is critical.

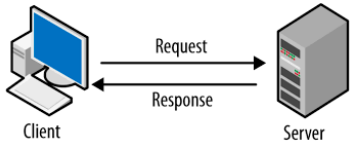
But why do we need architecture?

Purpose Built Computers

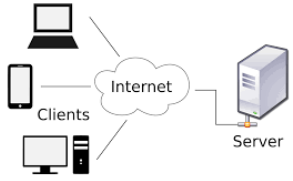
Batch Business Processing on Mainframes

monolythic applications on a desktop computer

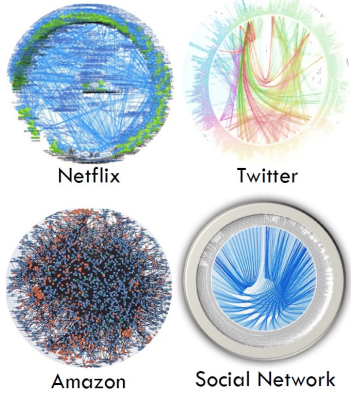
Basic Client/Server



Basic Web Applications

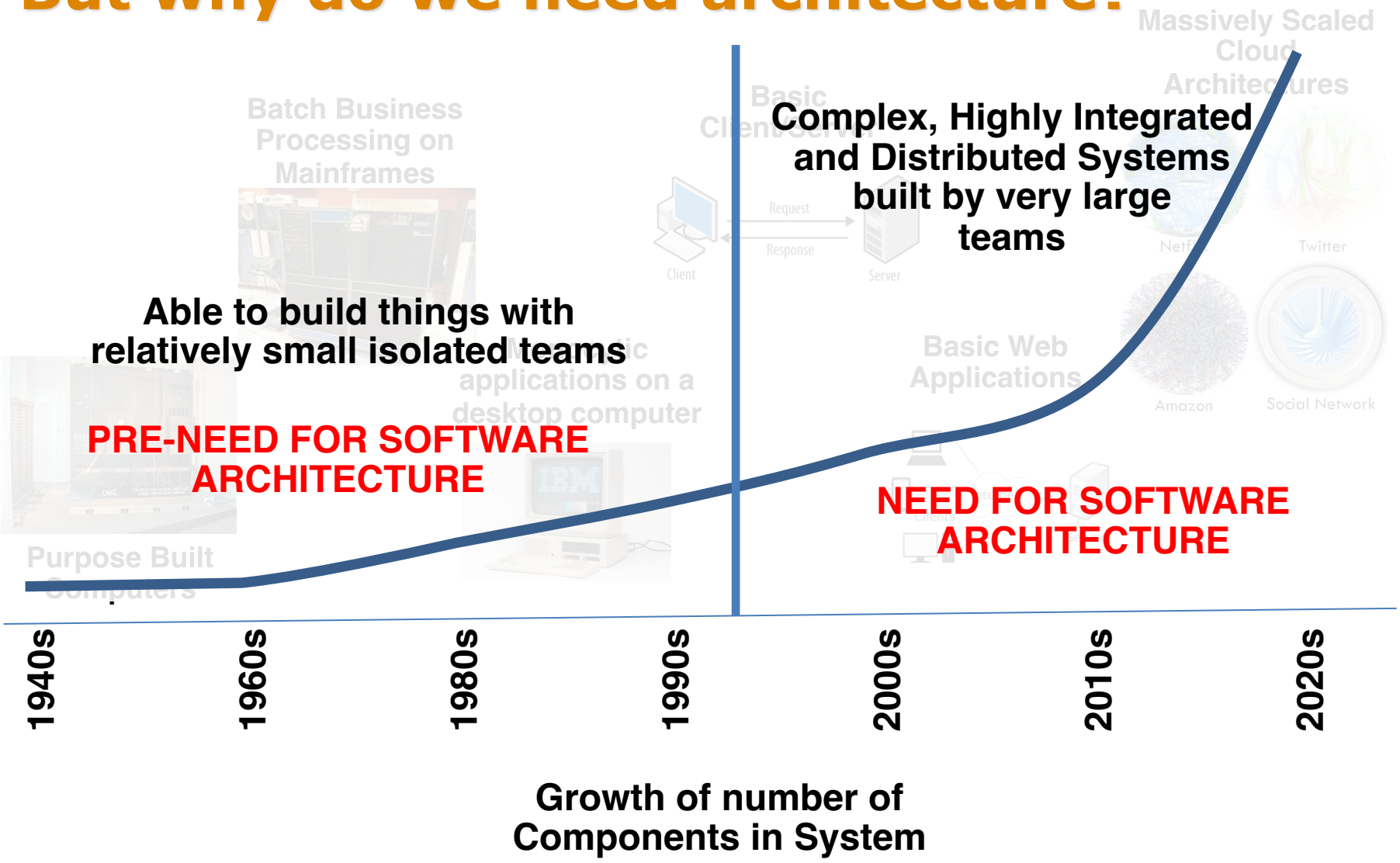


Massively Scaled Cloud Architectures



1940s 1960s 1980s 1990s 2000s 2010s 2020s

But why do we need architecture?



But why do we need architecture?

An Introduction to Software Architecture

David Garlan and Mary Shaw
January 1994

CMU-CS-94-166

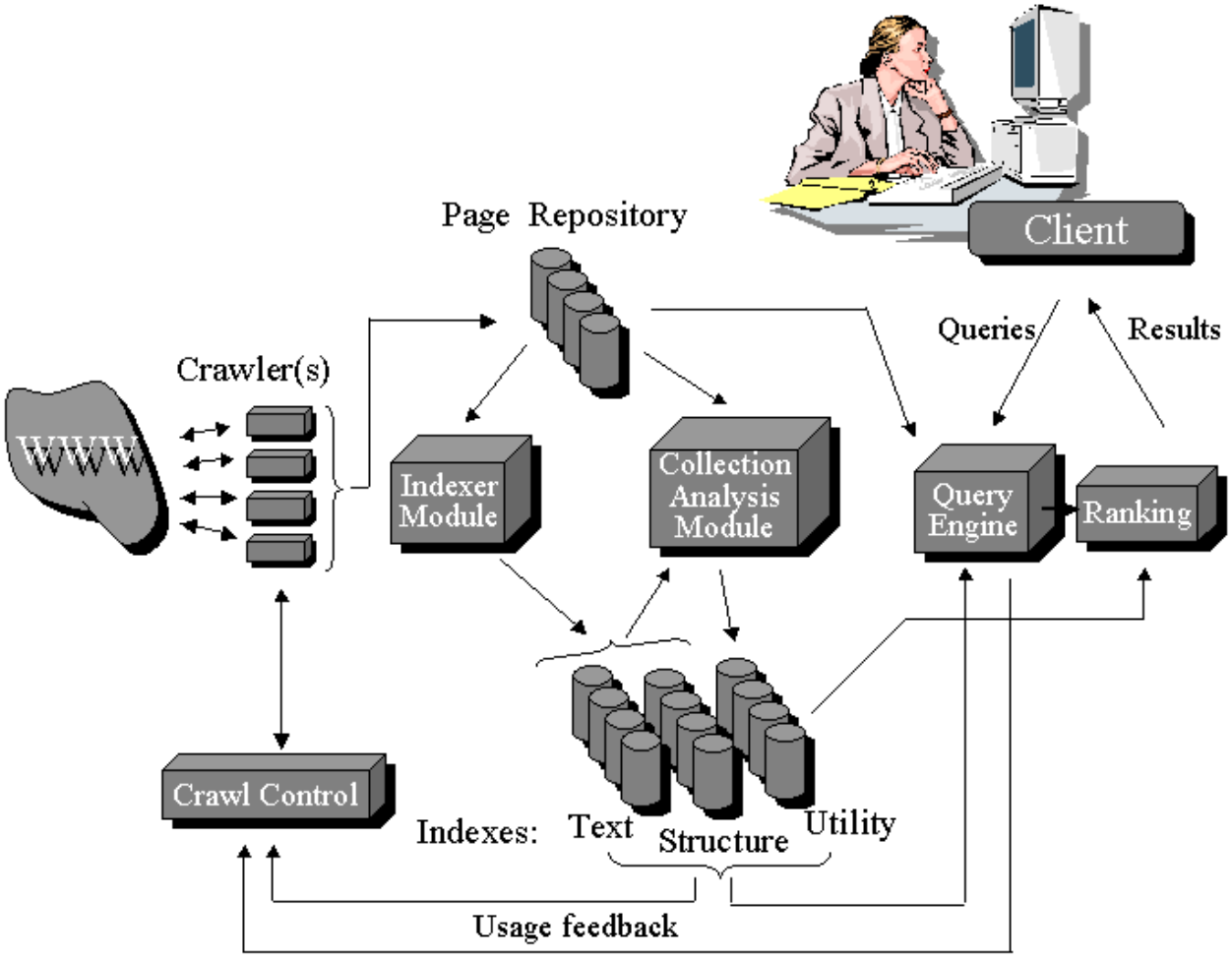
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Also published as "An Introduction to Software Architecture," *Advances in Software Engineering and Knowledge Engineering, Volume I,* edited by V.Ambriola and G.Tortora, World Scientific Publishing Company, New Jersey, 1993.

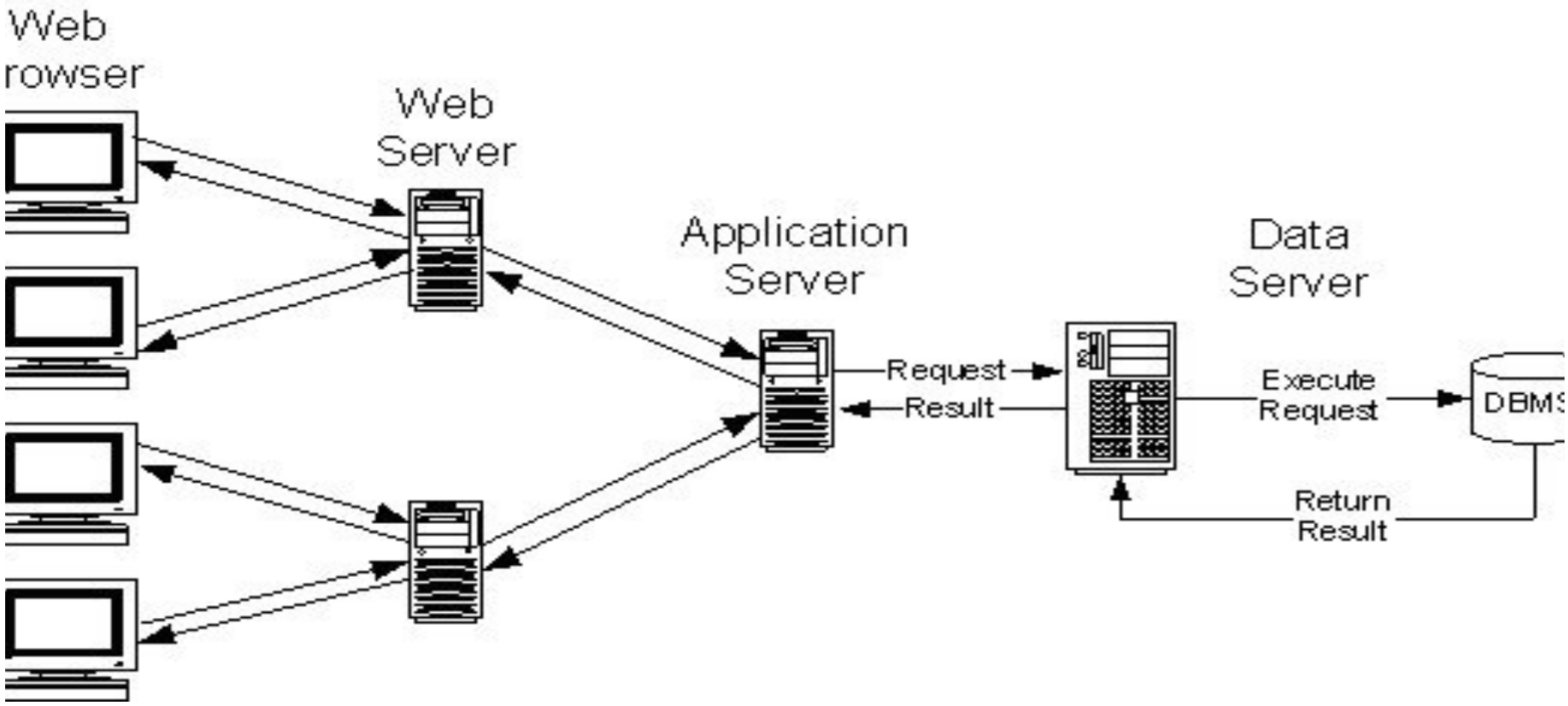
Also appears as CMU Software Engineering Institute Technical Report CMU/SEI-94-TR-21, ESC-TR-94-21.

The Seminal Paper Describing the Need for Software Architecture

General Search Engine Architecture (Arvind)



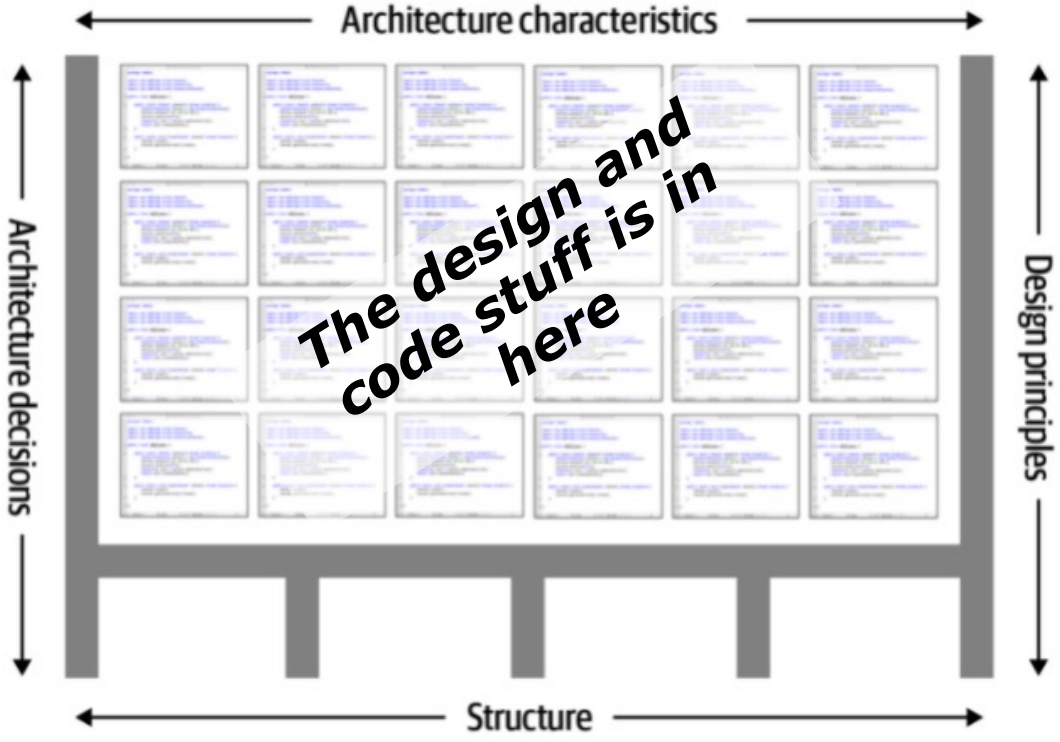
Multi-tier Architecture



Software Architecture as a Design Activity

- It's about software design
  - All architecture is software design, but not all design is software architecture
  - Part of the design process
- Simply, architecture focuses on issues that will be difficult/impossible to change once the system is built
  - Quality attributes like security, performance
  - Non-functional requirements like cost, deployment hardware
  - More on these later in this course
- Generally architecture is about the effective management of constraints on the to be built system

Software Architecture in the Context of Design



Architecture is key to constraining and structuring the design of systems

Ref: Fundamentals of Software Architecture – N. Ford & M. Richards

Architecture is Expressed using Models

- Software architecture emerged when it was becoming obvious that designs are too complicated to develop from scratch (circa early 1990s)
- Good designs tend to be build using models…
  - 1) Abstract different views of the system
  - 2) Build models using precise notations (e.g., UML)
  - 3) Verify that the models satisfy the requirements
  - 4) Gradually add details to transform the models into the design
- And such models can be derived from proven/established architecture patterns – more on this later

Architecture Abstractions

There are different techniques to perform software architecture analysis…

- Reference Architecture – focus is on the broad domain
- Solution Architecture – focus on the solution space
- Software Architecture – focus on the key technical abstractions

As well as a diversity of use cases where this information can be useful…

- New product definition
- Monitoring design quality (trends)
- Redocumenting and/or extending the lifespan of existing systems

The "art" of architecture is to pick the correct abstractions based on the desired objectives or outcomes

Architecture Framing

When thinking about a reference-, solution-, or software architecture focus on:

- Who is the constituent – aka, who am I trying to talk to or influence
- What do I want them to understand – aka, why am I talking about architecture
- What are the benefits, constraints, or tradeoffs associated with the architecture I am describing

If you cant answer any/all of the above questions the impact of your architectural work will likely be sub-optimal

We will look at documentation approaches later, but I'm not necessarily a big fan of standard notations – more on that later.

Architecture Done Poorly …

Get it Right the First Time !

Why Study Architecture?

- It is an integral aspect of structuring software so that it can be built, understood and maintained:
  - Client / Server
  - Layered systems
  - Cloud based systems
  - etc
- Architectural choices are critical in determining SW quality:
  - Performance
  - Scalability
  - Reuse
  - … and other non-functional properties

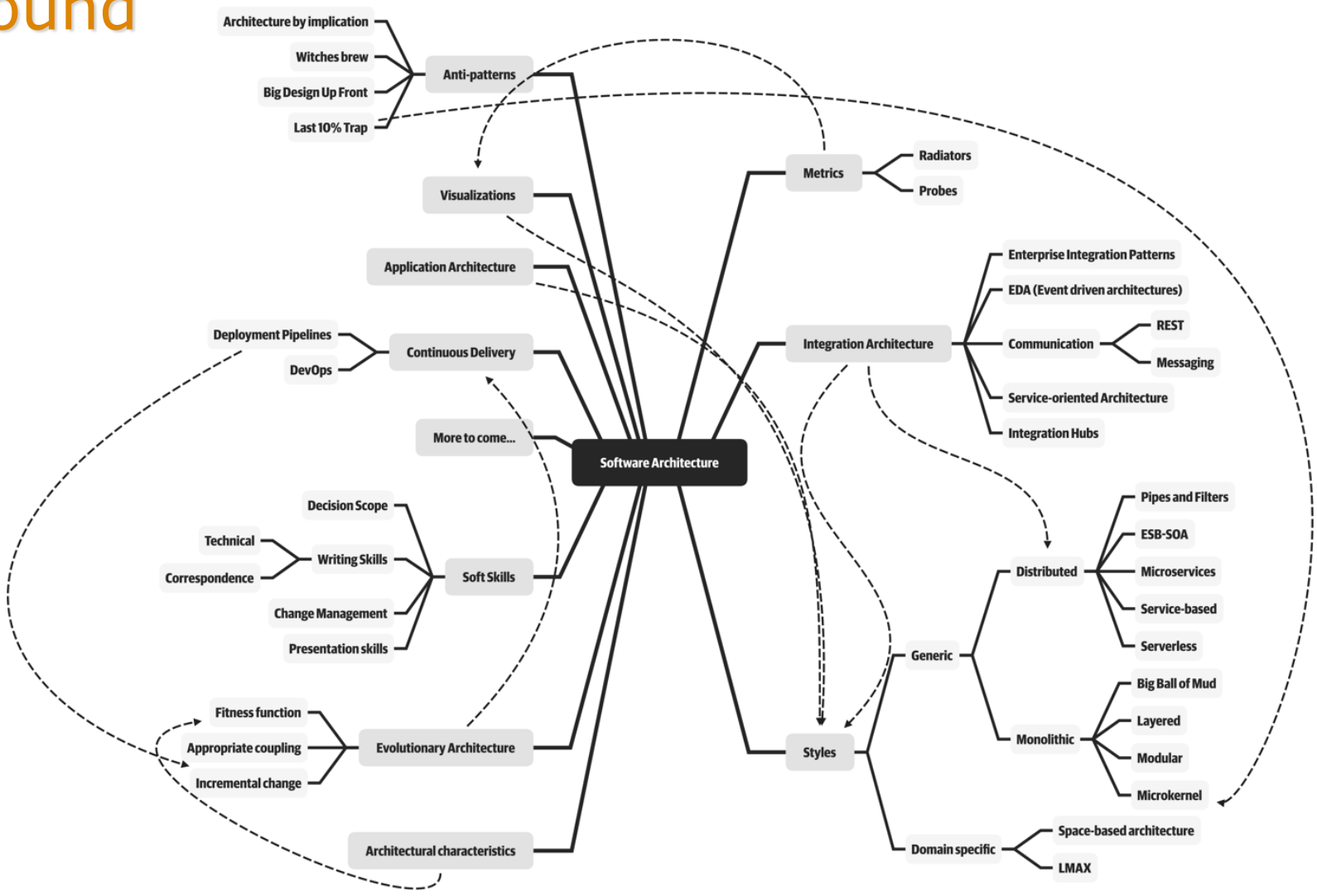
Why Study Architecture? (cont'd)

- As a architect, you need to:
  - Effectively manage complex tradeoffs to meet stakeholder objectives
  - Layout, socialize and influence your decisions.
  - Communicate with your team and convince your customer / boss of their merit.
  - Reason about the properties of the overall system in an abstract and convenient way.

Why Study Architecture? (cont'd)

- Certain "canonical" architectural designs provide valid blueprints for a lot of systems.
  - Architectural styles
- Learn to recognize applicability and characteristics of styles.
- Learn to use technologies that induce and support those styles.

Software Architects need to cover a lot of ground



What does a Software Architect do?

- Interact with stakeholders to make sure their needs are being met.
- Craft the right architecture to solve the problem at hand.
  - Make sure the right modeling is being done, to know that qualities like performance are going to be met.
  - Make sure that the architecture is not only the right one for operations, but also for deployment and sustainment.
- Document, influence, and communicate it.
- Give input as needed to issues like tool and environment selection.
- Manage risk identification and risk mitigation strategies associated with the architecture, and architecture decisions.
- Work closely with engineering teams, ensuring architecture guidance is relevant, is followed, and is adding value.

Expectations of a software architect

- Be able to create abstractions and models that capture key constraints and tradeoff decisions
- Continually analyze and update architecture as new things are learned
- Keep current with the latest trends
- Ensure compliance with decisions
- Have diverse exposure and experience
- Have business domain knowledge
- Possess interpersonal skills (required for influencing)
- Understand and navigate politics
- Effectively demonstrate credibility to diverse audiences