ENSF 607
Software Design & Architecture

Lecture 2.1
Topic - Basic Concepts (Terminology)

Tim Reimer

UNIVERSITY OF CALGARY
SCHULICH
School of Engineering

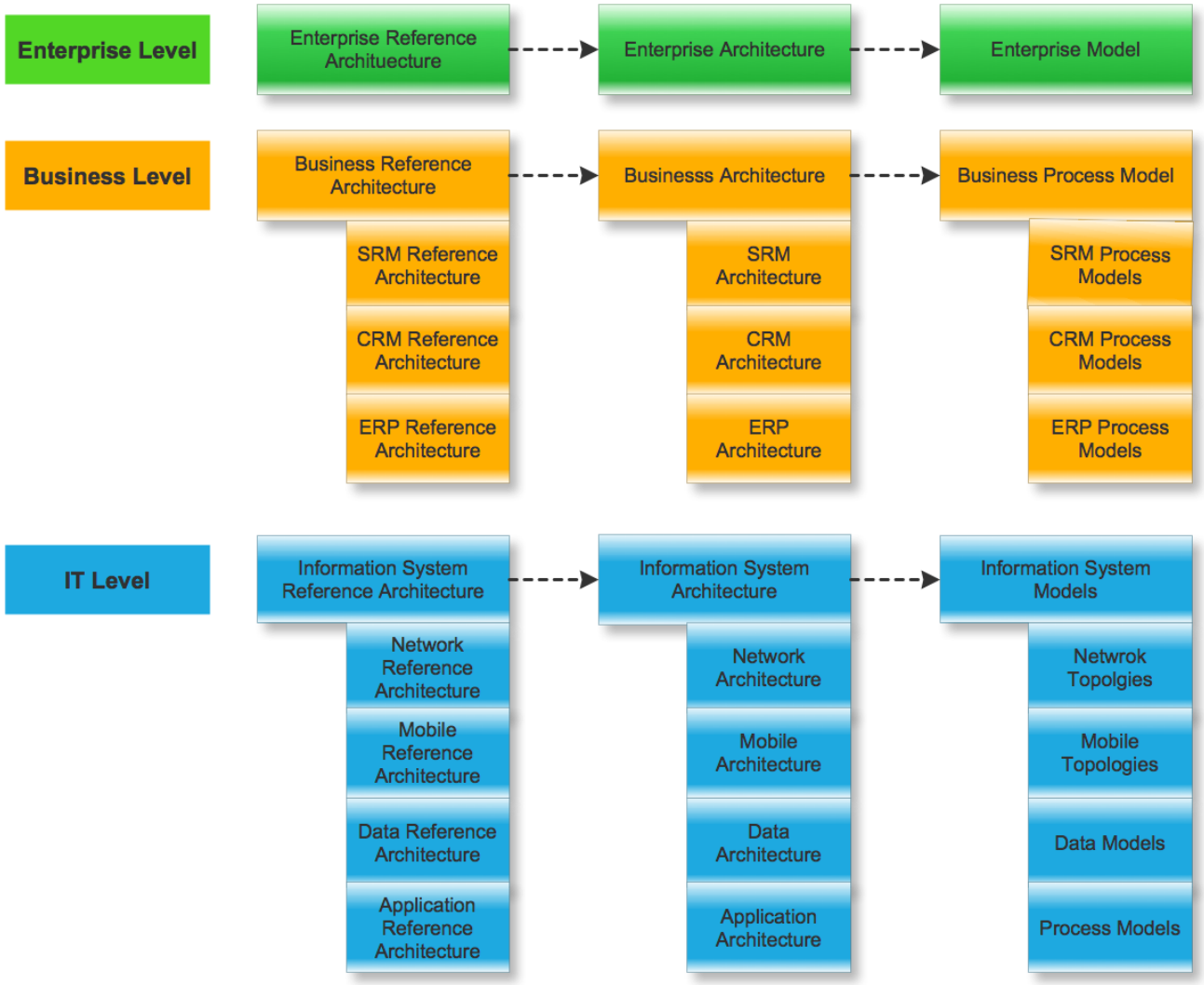
Outline
Four Themes in Basic Concepts

A. Terminology
B. Models
C. Processes
D. Stakeholders

A. Terminology
Key Terms

Term	Definition
Software Architecture	A software system's architecture is the set of principal design decisions made about the system.
Reference Architecture	A reference architecture is the set of principal design decisions that are simultaneously applicable to multiple related systems, typically within an application domain, with explicitly defined points of variation.
Design Decision	Design decisions encompass every facet of the system under development • Structure • Behavior • Interaction • Non-functional properties

Reference Architecture Context



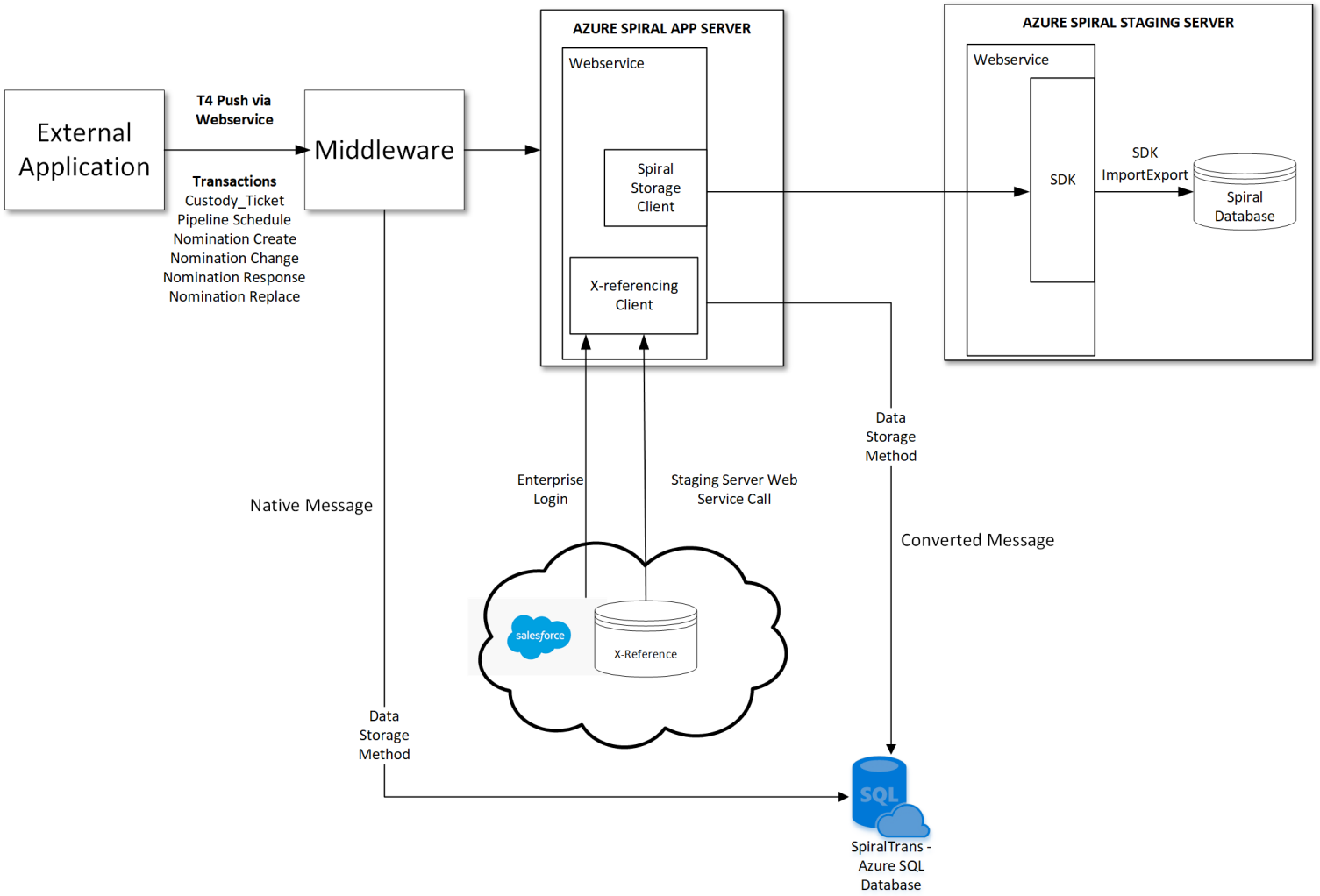
A. Terminology
Principal vs Temporal Design Decisions

Term	Definition
"Principal" Design Decision	• A degree of importance that grants a design decision "architectural status". • Not all design decisions are architectural, i.e., non-principal design decisions do not necessarily impact a system's architecture. • Example, the details of the selected algorithms or data structures • Definition of principal vs non-principal design decisions depends on what the stakeholders define as system goals.
"Temporal" Design Decision	• Design decisions are made/unmade over a system's lifetime • At a given time, a software system has only one overarching architecture • As the architecture of a large, complex, long-lived system is developed and evolved, the corresponding set of architectural design decisions will be changed hundreds of times. The outcome will, in effect, be hundreds of different (though related) architectures, rather than a single architecture. In that sense, architecture has a temporal aspect.

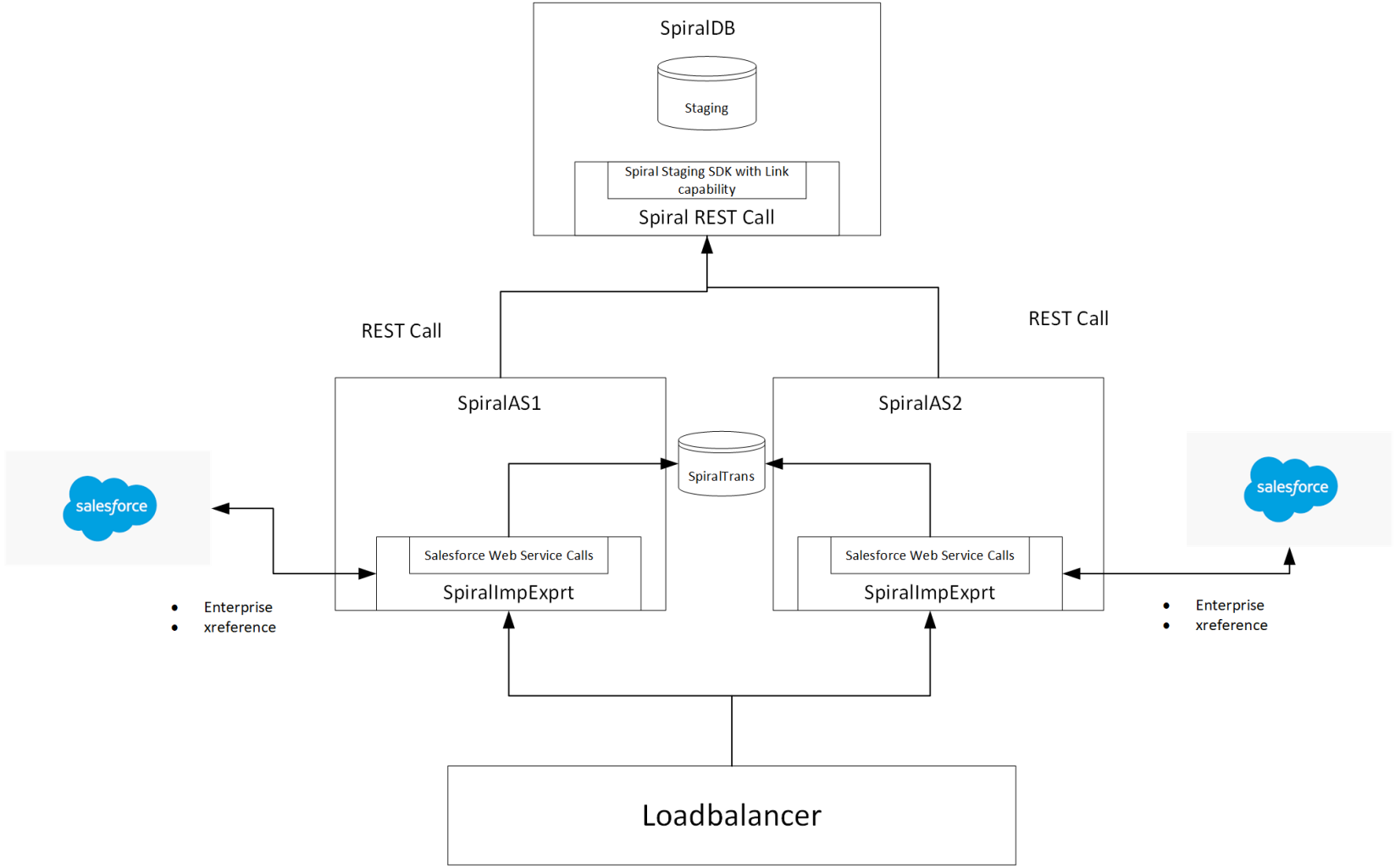
A. Terminology
Prescriptive vs Descriptive Architecture

Architecture	Description
Prescriptive	Captures the design decisions of a software system made prior to the system's construction. • It is as-conceived or as-intended architecture • "At any time, ***t***, during the process of engineering a software system, that system's architects will have made a set of architectural design decisions, ***P***, that reflect their intent. These design decisions comprise the system's prescriptive architecture."
Descriptive	Describes how the software system has been built. • It is as-implemented or as-realized architecture

Architecture Views : Side View



Architecture Views: Top View



A. Terminology
Architectural Evolution vs Degradation

Type	Description
Architectural Evolution	When a system evolves, ideally its prescriptive architecture is modified first. In practice, the system – and thus its descriptive architecture – is often directly modified. This happens because of • Developer sloppiness • Perception of short deadlines which prevent thinking through and documenting • Lack of documented prescriptive architecture • Need or desire for code optimizations • Inadequate techniques or tool support
Architectural Drift	Architectural drift is introduction of principal design decisions into a system's descriptive architecture that (a) are not included in, encompassed by, or implied by the prescriptive architecture, but which (b) do not violate any of the prescriptive architecture's design decisions.
Architectural Erosion	Architectural erosion is the introduction of architectural design decisions into a system's descriptive architecture that violate its prescriptive architecture.

A. Terminology

Component – Unit of Deployment

- Elements that encapsulate processing and data in a system's architecture are referred to as *software components*
- A *software component* is an architectural entity that
    - encapsulates a subset of the system's functionality and/or data
    - restricts access to that subset via an explicitly defined interface
    - has explicitly defined dependencies on its required execution context
- Components typically provide application-specific services

A. Terminology

Connectors

- In complex systems *interaction* may become more important and challenging than the functionality of the individual components
- **Definition:** A *software connector* is an architectural building block tasked with effecting and regulating interactions among components
- In many software systems connectors are usually simple procedure calls or shared data accesses. Much more sophisticated and complex connectors are possible.
- Connectors typically provide application-independent interaction facilities.
- Examples of connectors:
- Procedure call connectors
  - Shared memory connectors
  - Message passing connectors
  - Streaming connectors
  - Distribution connectors
  - Wrapper/adaptor connectors

A. Terminology
Configurations

- Components and connectors are composed in a specific way in each system's architecture to accomplish that system's objective
- **Definition:** An *architectural configuration*, or topology, is a set of specific associations between the components and connectors of a software system's architecture

A. Terminology
Architectural Styles

- Certain design choices regularly result in solutions with superior properties
  - Compared to other possible alternatives, solutions such as this are more elegant, effective, efficient, dependable, evolvable, scalable, and so on
- An *architectural style* is a named collection of architectural design decisions that
  - are applicable in a given development context
  - constrain architectural design decisions that are specific to a particular system within that context
  - elicit beneficial qualities in each resulting system

A. Terminology
Architectural Patterns

- An *architectural pattern* is a set of architectural design decisions that are applicable to a recurring design problem, and parameterized to account for different software development contexts in which that problem appears
- A widely used pattern in modern distributed systems is the *three-tiered system* pattern
  - Science
  - Banking
  - E-commerce
  - Reservation systems

Figure 3-4. Graphical view of the three-tier system architectural pattern.

B. Architecture Models

Term	Definition
Architecture Model	An artifact documenting some or all of the architectural design decisions about a system
Architecture Visualization	A way of depicting some or all of the architectural design decisions about a system to a stakeholder
Architecture View	A subset of related architectural design decisions

C. Architectural Processes

1. Requirement's definition
2. Define Architecture Characteristics
3. Design Architecture Style
4. Document Architecture Decisions
5. Design Technology Stack
6. Define system components

D. Architecture Stakeholders

- Architects
- Developers
- Testers
- Managers
- Customers
- Users
- Vendors