Software Design and Architecture

Software Architectural Design Space

Introduction

- Architectural design space
  - Design alternatives that can support functional and non-functional requirement specifications

- Major challenge: how to produce agile software architectures that can be easily adapted to their changing environments without major re-engineering

Introduction …

- **Software architecture**
  - Software elements: process, object, instance of software component, service
  - Elements could be deployed on different hardware or software platforms, developed in different languages or on different software frameworks
  - Connectors: local method invocation, remote method invocation, service call, messaging

Types of Software Structures

- A software architecture can be described with various software structures, each from a different perspective

  - **Static structure**
  - **Runtime structure**
  - **Management structure**

- Different structures use different elements and connectors, and have different performance attributes

Static Structure

- At software development time, the main software elements are source code modules or files
- Each of these software modules has assigned functional and non-functional attributes
- The connectors at this level are in the form of module dependency
  - Module A is connected to module B if and only if A needs to invoke some methods in module B during execution

Static Structure …

- Static connectors have attributes including
  - *Direction*. If module A invokes a method of module B during execution, then there is a unidirectional connector from module A to module B.
  - *Synchronization*. A method invocation can be synchronous or asynchronous.
  - *Sequence*. Some connectors must be used in a particular sequence.

Managing Static Structural Representations

- Two kinds of static hierarchical relations
  - *A linear client-server relation is formed when a component provides primitive abstractions to another component; in this sense, components can refer to abstractions that once defined can be used throughout the entire design (at all levels).*
  - A tree-like hierarchy of refinement relations defined between an abstraction (i.e., a component) and its implementation, by the recursive division of components into sub-components

Software Runtime Structure

- At runtime elements are threads, processes, functional units, and data units
- These elements may run on the same computer or on multiple computers across a network
- The same element in a code structure can implement or support multiple runtime elements
  - In a client-server application, the same client module may run on many client computers
- Several code structure elements may implement or support a single runtime element
  - Many threads may run multiple methods from different classes that may be packaged in different code units

Software Runtime Structure

- Runtime connectors inherit attributes from their source-code structure counterparts, with a few extra attributes
  - ***Multiplicity***. One element can be connected to multiple other elements if it needs to invoke methods of multiple elements at runtime
  - ***Distance and connection media***
  - ***Universally invokable***. A connector with this attribute set to true allows any external software system, no matter what hardware/ software platforms that they run on and what programming languages or software frameworks that they are developed in, can invoke the method at the connector's target
  - ***Self-descriptive***. A connector with this attribute set to true can allow external software systems invoke its target method without the pre-installation of any software specific for the method

Summary

- Introduce software architecture
- Introduce software static structure
- Introduce software runtime architecture