Requirements and Architecture
System Requirement Categories

- **Functional** requirements - what the system must do, how it must react to run-time stimuli
- **Non-functional** - these requirements qualify functional requirements
  - Quality attributes (the “abilities”)
  - Business
  - Architectural
- **Constraints** - design decisions with zero degrees of freedom; i.e., a design decision that has already been made for you.
Quality Attribute(QA) Considerations

“A QA is a **measureable** or **testable** property of a system that is used to indicate **how well the system satisfies the needs** of its stakeholders.” (SAiP p.63)

- If a functional requirement is "when the user decides to change runtime preferences a change preferences dialog shall appear":
  - a **performance** QA might describe **how quickly** the dialog will appear;
  - an **availability** QA might describe **how often this function will fail**, and **how quickly it will be repaired**;
  - a **usability** QA might describe how **easy** it is **to learn** this function.

**Functionality determines software architecture true or false?**
Functionality and Architecture

- **Functionality does not determine architecture**
  - Given a set of required functionality, there is no end to the architectures you could create to satisfy that functionality
  - So functional design doesn’t matter?

- **Functionality and quality attributes are orthogonal**
  - Many ways to implement functionality with varying degrees of quality
Architecturally Significant Requirements

- Architecture designs build systems that satisfy requirements.
- An architecturally significant requirement (ASR) is a requirement that will have a profound effect on the architecture
  - Significant = high cost of change
- How do we find those?
  - In requirements documents, right?

NOTE: ASRs are predominately but not exclusively comprised of quality attributes
Don’t Get Your Hopes Up

- **High-quality requirements** documents may **not exist**
- **Emphasis on functionality** not quality attributes
- **Quality attributes (ASRs)**, when captured at all, are often **captured poorly**.
  - “The system shall be modular”
  - “The system shall be **easy to use**”
  - “The system shall meet users’ performance expectations”
- **Requirements documents do not include everything useful to the architect**
  - ASRs often derive from **business goals**
  - Development environment
- An **architect can’t wait** for “finished” requirements
  - Proactively interview stakeholders but also …. 
  - **Suggest** relevant ASR’s for discussion
“Twin Peaks” Model

- **Interplay** between requirements and architecture
- **Tradeoffs** between system problem and solutions
  - Challenges the tradition of avoiding solution thinking while discovering requirements
- **ASRs** may only be recognized after some architecture design
- **Architecture feedback** may eliminate infeasible requirements or poor cost/value benefits
Characteristics of ASRs

- Hard to define and articulate - general and abstract concepts users don’t understand
- Needed early in the life cycle before needs are fully understood
- Vaguely described, subjective; e.g., 24/7
- Tend to be neglected initially – significance not appreciated
  - Embedded in the expression of other requirements
- Variable – subject to requirements and technology change
- Situational – significance may depend on system context; e.g., scale, legacy systems, technology
Recognizing ASRs

- **Wide system impact**
- May involve resolving **requirements tradeoffs**
- **Strict** (constraining, limiting, nonnegotiable) requirements that **dictate a design**
- May **invalidate** conventional **design** tactic decisions
- **Difficult** to achieve technically
Quality Attributes

- **Operational categories**
  - Availability
  - Interoperability
  - Reliability
  - Usability
  - Performance
  - Deployability
  - Scalability
  - Monitorability
  - Mobility
  - Compatibility
  - Security
  - Safety

- **Developmental categories**
  - Modifiability
  - Variability
  - Supportability
  - Testability
  - Maintainability
  - Portability
  - Localizability
  - Development distributability
  - Buildability
Business Goals May Drive ASRs

- Market share, competition
- Product lines
- Global markets
- Revenue
- Cost to develop, deploy, operate, and maintain
- Personnel objectives
- Liability, safety, reputation
- Standards and regulations
- Intellectual property
- Environmental and sustainability concerns
Business Qualities (System QAs)

- **Time to market** (buy or reuse)
- **Cost and benefit** (tradeoffs)
- Projected **lifetime** (modifiability, scalability, portability)
- **Target market** (product line strategy, reliability, performance, usability)
- **Rollout schedule** (scope and scale, deployment flexibility)
- **Integration** with legacy and external systems (interfaces, constraints, interoperability)
- **Constraints** (safety, security, modifiability, …)
Architectural Qualities

- **Conceptual integrity** – consistency, do similar things in similar ways
- **Correctness and completeness**
Conceptual Integrity