System Stakeholders

- Architectural documents are used by a variety of system stakeholders:
  - Developers
  - Managers
  - Sales
  - Testers
  - Support
  - Maintenance
  - DevOps
  - Customers
Stakeholder Questions

- Management: are we on schedule?
- Developers: who is responsible for what?
- Sales: can we claim it can do this task?
- QA: what teams do we talk to about defects?
- DevOps: where should this component be deployed?
- Support: which QA team signed off on this?
- Maintenance: how can we add this feature?
Stakeholder Conflicts

- System requirements fall into two broad categories:
  - Functional Properties: what the system is supposed to do (‘the system shall do X’).
  - Non-Functional Properties: what the system is supposed to be (‘the system shall be Y’).

- Each stakeholder will have their own opinion about what NFPs matter most:
  - e.g., the development team will care about maintainability more than the customer
  - e.g., QA will be more interested in the testability of the application than sales
NFPs

- NFPs are constraints on the manner in which the system implements and delivers its functionality.

- E.g.,
  - Efficiency
  - Complexity
  - Scalability
  - Heterogeneity
  - Adaptability
  - Security
  - Dependability
FP vs NFP

- Products are sold based on their FPs.
  - e.g., Cell phone, Car, Tent.
- However, NFPs play a critical role in perception.
  - “This program keeps crashing”
  - “It doesn’t work with my [...]”
  - “It’s too slow”
Design guidelines for NFPs

› Provide guidelines that support various NFPs.
› Focus on architectural level:
  › Components
  › Connectors
  › Topologies
NFP: Efficiency

- Efficiency is a quality that reflects a system’s ability to meet its performance requirements.

Components:
- Keep them “small”.
- Simple and compact interfaces.
- Allow multiple interfaces to the same functionality.
- Separate data from processing components.
- Separate data from meta data.

Connectors:
- Carefully select connectors.
- Be careful of broadcast connectors.
- Encourage asynchronous interaction.
- Be wary of location/distribution transparency.

Topology:
- Keep frequent collaborators “close”.
- Consider the efficiency impact of selected styles.

[TAILOR ET AL.]
NFP: Complexity

- Complexity is a property that is proportional to the size of a system, its volume of constituent elements, their internal structure, and their interdependencies.

- Components:
  - Separate concerns.
  - Isolate functionality from interaction.
  - Ensure cohesiveness.
  - Insulate processing from data format changes.

- Connectors:
  - Isolate interaction from functionality.
  - Restrict interactions provided by each connector.

- Topology:
  - Eliminate unnecessary dependencies.
  - Use hierarchical (de)composition.
NFP: Scalability / Heterogeneity

- **Scalability**: The capability of a system to be adapted to meet new size / scope requirements.

- **Heterogeneity**: A system’s ability to be composed of, or execute within, disparate parts.

- **Portability**: The ability of a system to execute on multiple platforms while retaining their functional and non-functional properties.
NFP: Scalability / Heterogeneity

- Components:
  - Keep components focused
  - Simplify interfaces
  - Avoid unnecessary heterogeneity
  - Distribute data sources
  - Replicate data

- Connectors:
  - Use explicit connectors
  - Choose the simplest connectors
  - Direct vs. indirect connectors

- Topology:
  - Avoid bottlenecks
  - Place data close to consumer
  - Location transparency
NFP: Evolvability

- Evolvability: The ability to change to satisfy new requirements and environments.

- Components:
  - Same as for complexity.
    - Goal is to reduce risks by isolating modifications.

- Connectors:
  - Clearly define responsibilities.
  - Make connectors flexible.

- Topology:
  - Avoid implicit connectors.
  - Encourage location transparency.
NFP: Dependability

- Reliability: The probability a system will perform within its design limits without failure over time.
- Availability: The probability the system is available at a particular instant in time.
- Robustness: The ability of a system to respond adequately to unanticipated runtime conditions.
- Fault-tolerance: The ability of a system to respond gracefully to failures at runtime.
  - Faults arise from: environment, components, connectors, component-connector mismatches.
- Survivability: The ability to resist, recover, and adapt to threats.
  - Sources: attacks, failures, and accidents.
  - Steps: resist, recognize, recover, adapt.
- Safety: The ability to avoid failures that will cause loss of life, injury, or loss to property.
NFP: Dependability

- **Components:**
  - Control external component dependencies.
  - Support reflection.
  - Support exception handling.

- **Connectors:**
  - Use explicit connectors.
  - Provide interaction guarantees.

- **Topology:**
  - Avoid single points of failure.
  - Enable back-ups.
  - Support system health monitoring.
  - Support dynamic adaptation.