# Agile Architecture

Below are a set of architectural principles and values that guide solution architectures and allow teams to slice technical development efforts in manner that match backlog story slicing and can be completed within a sprint.

## Agile Architectural Principles

* Economic view balancing lead time and quality for the delivery of value
  + A commitment to delivering valuable working solutions with a minimized amount of lead time while maintaining a baseline of quality.
  + Avoiding scope creep and excessive anticipation of future enhancement.
* Balance of Emergent and Intentional Architecture
  + Emergent Architecture: *An extensible design resulting from collaboration and experimentation.*
    - Optimal Extensible design example: adding functionality through hooking business logic layers or inheritance
    - Less optimal modular design example: Updating existing logic modules to add new logic to existing expectations.
  + Intentional Architecture: *Architectural design intended to solve for a particular scenario through modeling*
    - Intentional Strategy: initiative-based foundational frameworks developed to promote performance, usability, extensible enhancement, and incremental implementation.
* Fast learning Cycles
  + Quick research is possible within backlog refinement sessions (find existing answers)
  + Spike research is possible in a reasonable timeframe (learning new things)
  + Prototyping is possible with solutions containing larger degrees of uncertainty (experiments on which story solutions can be based)
* Preservation of options
  + Identify and maintain variability and flexibility in both solutions and framework to keep solutions more loosely coupled and extensible
* Objective evaluation of existing systems and solutions
  + Reality ease in inspecting existing framework and solution elements

## Agile Architectural Values

* Emergent Design: *Architecture is discoverable and the result of collaboration*
  + Instead of: big up-front blueprinting
* Extensible Domain Design: *Collaboration across Business Architecture, Information Architecture, and Technical Architecture.*
  + Instead of: siloed blueprinting that works in gates or phases
* Less is More: *Design simple uncoupled solution with focused user experiences*
  + Instead of: tightly coupled layers with deep shared logic
* Decentralized decision-making: *Emergent design and development plans implement the intentional architecture at the team level*
  + Instead of: Team designs requiring solution approval
* When in Doubt, Model it out: Proactive *research spikes, and prototypes, with collaborative (sprint-oriented) domain and use-case models drawn from experiments*
  + Instead of: Big up-front design determining technical design requirements
* Test what you build: *Work of testing and validate belongs with the team that builds*
  + Instead of: Separating incremental development from incremental test
* Teams lead innovation: *The work of innovation comes from team-level discoveries in collaboration and experimentation*
  + Instead of: Innovation from prescriptive technical up-front design
* Architectural flow is followed: *Solution intent leads at the Solution Train level and then trickle down to the Features and stories in the backlog as teams collaborate and experiment in spikes and stories.*
  + Instead of: *Big up-front design handed to Agile Dev Team members as proactive technical specifications.*