All the really important mistakes are made the first day.

– Eberhardt Rechtin, System Architecting
YOU ARE HERE

SPECIFY PRODUCT

SPECIFY SOFTWARE

CREATE SW ARCHITECTURE

DESIGN MODULES

IMPLEMENT

UNIT TEST

INTEGRATION TEST

SOFTWARE TEST

ACCEPTANCE TEST

PRODUCT

TRACEABILITY & VALIDATION

Test Plan & Test Results

Product Requirements
Software Requirements
High Level Design
Detailed Design
Test Plan
Source Code

Test Results

Test Results

Integration Test Results

Unit Test Results

Software Test Results

Test Plan & Test Results

Test Plan & Test Results

Test Plan & Test Results

Test Plan & Test Results

Test Plan & Test Results

Product Requirements

Test Results

Test Results

Test Results

Test Results

Test Results
Anti-Patterns:

- Skipping from requirements to code
- No picture that shows how all the components fit together
- “Wedding cake” layer diagram that omits interface information

Elements of High Level Design

- Architecture: boxes, arrows, interfaces
  - Arrows/interfaces show communication paths between components
  - Recursive: one designer’s system is another designer’s component
- High Level Design (HLD) = architecture (nouns) + requirements (verbs)
  - Sequence Diagrams (SDs) show interactions
Software architecture shows the big picture
- Boxes: software modules/objects
- Arrows: interfaces
- Box and arrow semantics well-defined
  - Meaning of box/arrow depends on goal
- Components all on a single page
  - Nesting of diagrams is OK

Many different architecture diagrams are possible, such as:
- Software architecture (components and data flow types)
- Hardware architecture with software allocation
- Controls architecture showing hierarchical control
- Call graph showing run-time hierarchy
Sequence Diagram as HLD Notation

- **SD construction:**
  - Each object has a time column extending downward
  - Arcs are interactions between objects

- Each SD shows a scenario
  - Top ovals are preconditions
  - Middle ovals are side effects
  - Bottom ovals are postconditions

- **SD is a *partial* behavioral description for objects**
  - Generally, each object participates in *multiple* SDs; each SD only has *some* objects
  - The set of all SDs forms the HLD for all objects in the system
For each object in each SD: identify input & output arcs

- Detailed Design: design statechart that accounts for all SD behaviors
Modes vs. States

- State: corresponds to internal state machine
  - “When in state S1 the system shall display current time”
- Mode: user-visible change in operations
  - “When in stopwatch mode, pressing Button 1 shall do XYZ”

State-type descriptions in HLD should be modes

- Input & output behaviors can change depending upon mode
  - **GOOD:** Pressing X in Mode Y displays Z
    » Mode Y tells you which sequence diagram applies
  - **NOT:** Pressing X in state S1 changes state to S2
    » There is no point describing the detailed design this way
HLD should include:
- One or more architecture diagrams
  - Defines all components & interfaces
  - HW arch., SW arch., Network arch., ...
- Sequence Diagrams
  - Both nominal and off-nominal interactions
  - See 18-649 soda machine for a fully worked example
- HLD must co-evolve with requirements
  - Need both nouns + verbs to define a system!

High Level Design pitfalls:
- Diagrams that leave out interactions
- Boxes and arrows don’t have well defined meanings
- HLD that bleeds into detailed design information
  - Should have separate Detailed Design per component

High Level Design Best Practices

http://www.ece.cmu.edu/~ece649/project/sodamachine/index.html
2011 Health Plan Flow Chart: What’s wrong with this as an architecture diagram?

CAN YOU PASS THE SALT?

I SAID—
I KNOW! I'M DEVELOPING A SYSTEM TO PASS YOU ARBITRARY CONDIMENTS.
IT'S BEEN 20 MINUTES!
IT'LL SAVE TIME IN THE LONG RUN!

https://xkcd.com/974/