Prof. Philip Koopman

Carnegie Mellon University

Electrical & Computer ENGINEERING

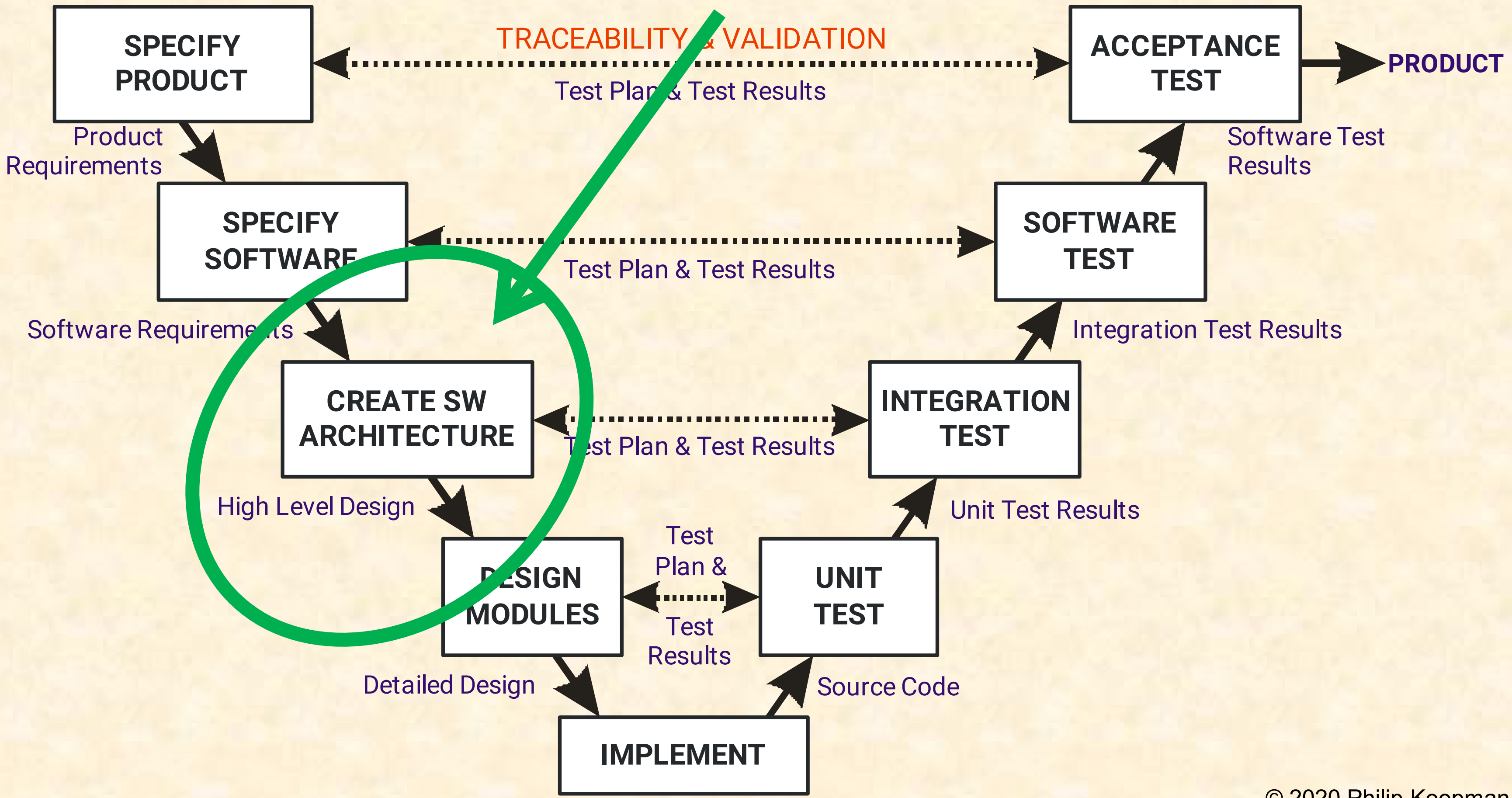
Software Architecture & High Level Design

All the really important mistakes are made the first day.

– Eberhardt Rechtin, System Architecting

YOU ARE HERE

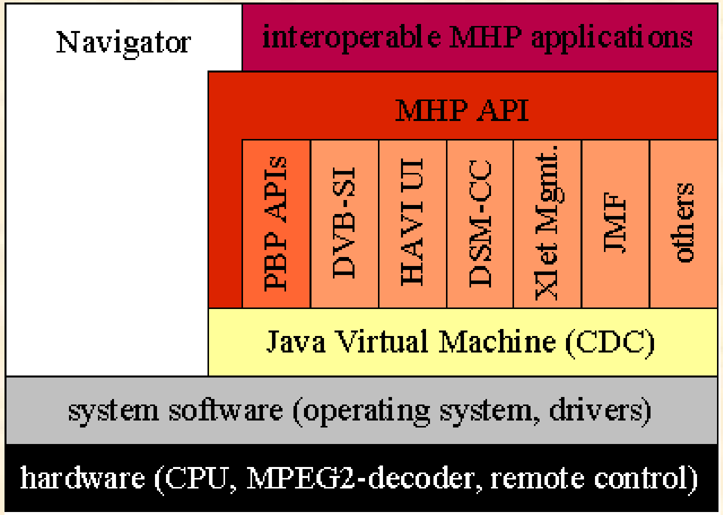
Carnegie Mellon University



Architecture & High Level Design (HLD)

Carnegie Mellon University

- **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



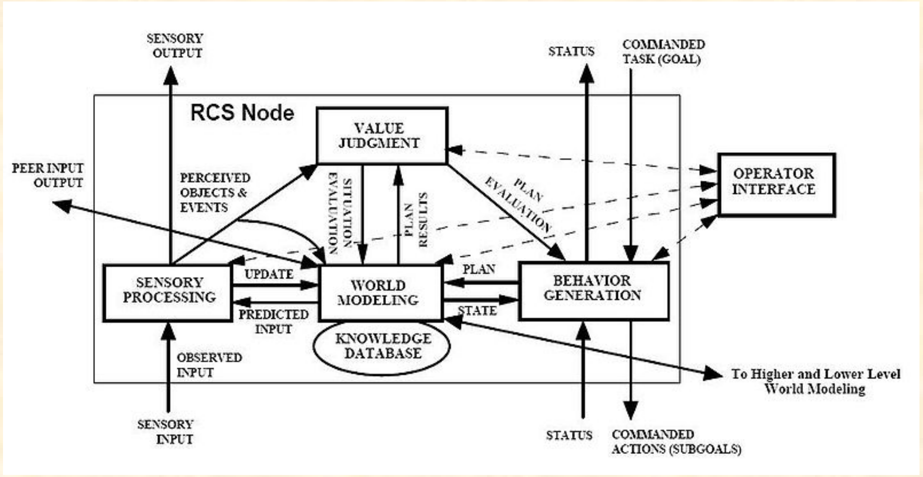
Navigator
interoperable MHP applications
MHP API
PBP APIs
DVB-SI
HAVI UI
DSM-CC
Xlet Mgmt.
JMF
others
Java Virtual Machine (CDC)
system software (operating system, drivers)
hardware (CPU, MPEG2-decoder, remote control)

https://goo.gl/J8MAuK

- **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

Architecture: Boxes and Arrows

- 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

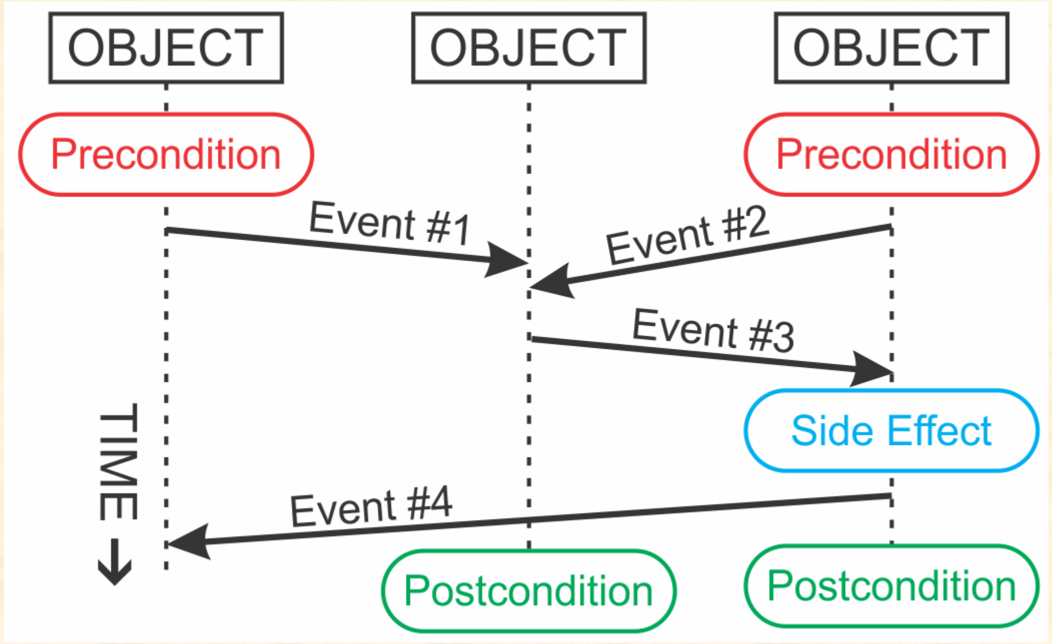


https://goo.gl/WnciF3

- 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

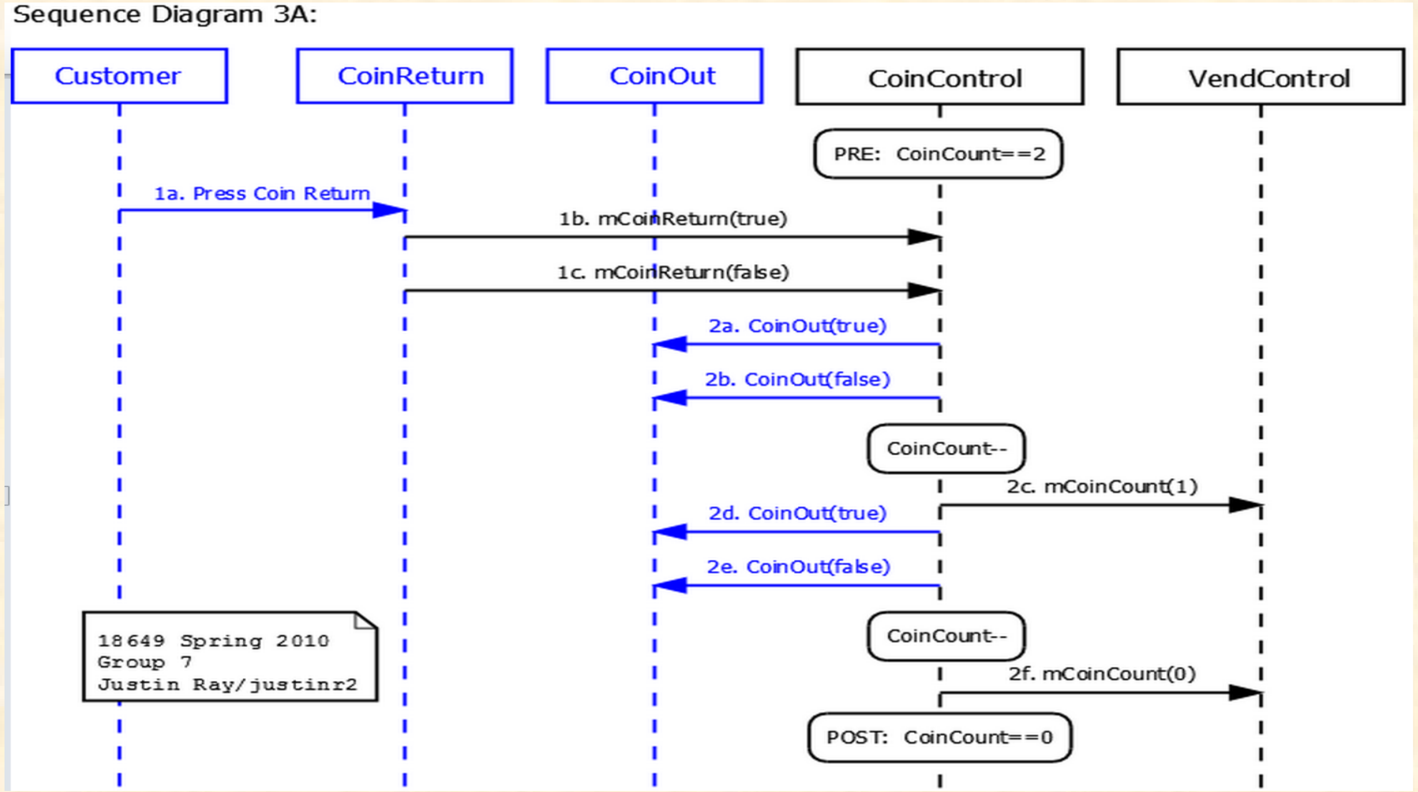
- SD construction:
  - Each object has a time column extending downward
  - Arcs are interactions between objects
- Each SD shows a scenario
  - Top ovals are preconditions
  - Middle ovals are side effects
  - Bottom ovals are postconditions
- SD is a partial behavioral description for objects
  - Generally, 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



Example Sequence Diagram

Legend: Blue = physical objects / Black = microcontrollers with software

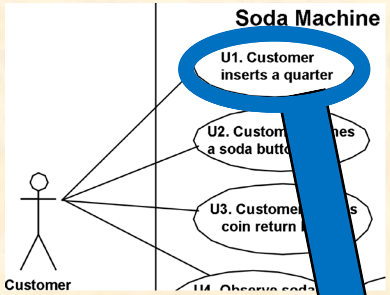
PRE = precondition / POST = postcondition / other ovals are side effects



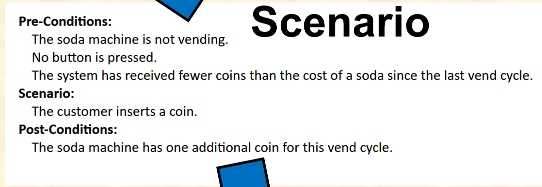
Use Cases to Sequence Diagrams

- Use Case diagram – types of interactions
  - System has multiple use cases
  - Example: Use Case #1: Insert a coin
- Scenario – a specific variant of a use case
  - Each use case has one or more scenarios
    - Scenario 1.1: insert coin to add money
    - Scenario 1.2: insert excess coin (too many inserted)
    - Scenario 1.3: … some other situation…
  - Interactions between objects are different for each scenario
- Sequence Diagram – a specific scenario design
  - For our purposes each scenario has one sequence diagram
    - Sequence diagrams 1.1, 1.2, 1.3 show specific interactions
- Statechart – design that incorporates all scenarios
  - One StateChart per object, addressing all scenarios

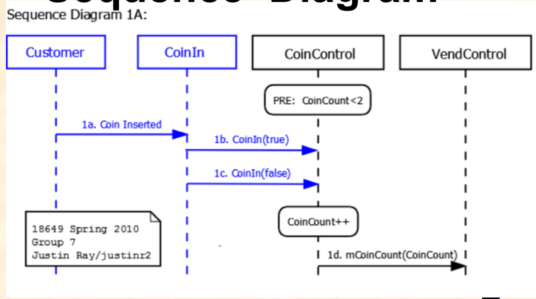
Use Cases



Scenario

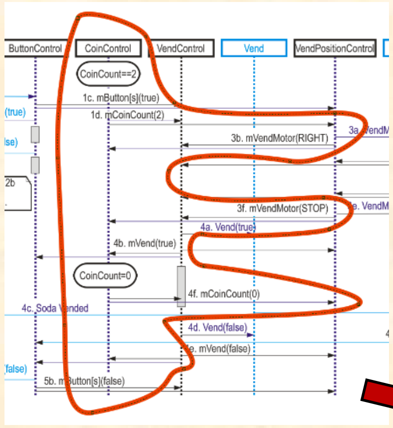

Pre-Conditions:
The soda machine is not vending.
No button is pressed.
The system has received fewer coins than the cost of a soda since the last vend cycle.
Scenario:
The customer inserts a coin.
Post-Conditions:
The soda machine has one additional coin for this vend cycle.

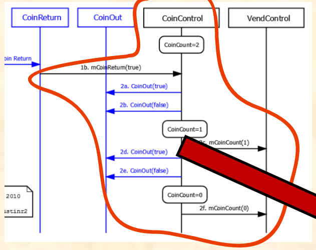
Sequence Diagram



Combining SDs To Make Statecharts

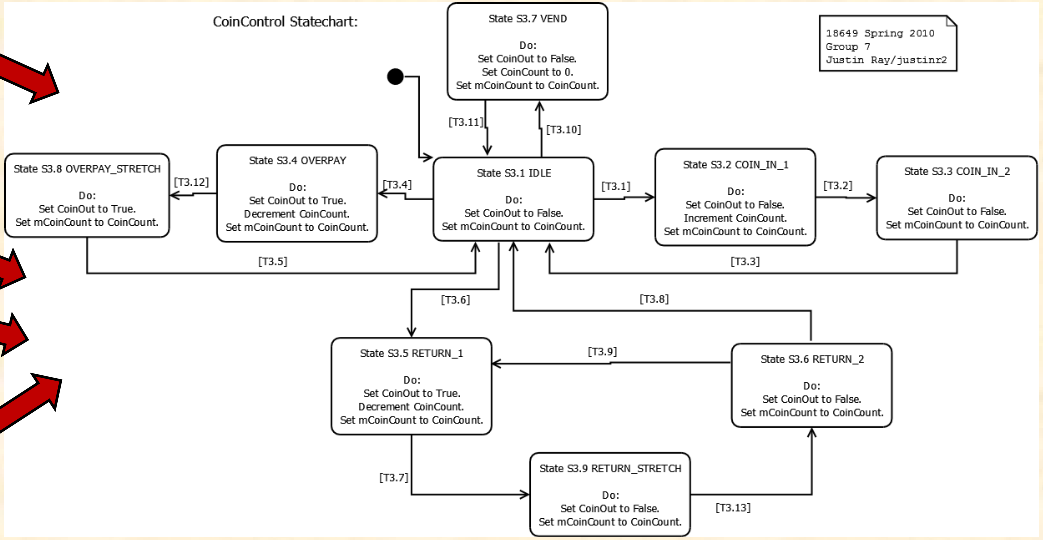
- For each object in each SD: identify input & output arcs
  - Detailed Design: design statechart that accounts for all SD behaviors



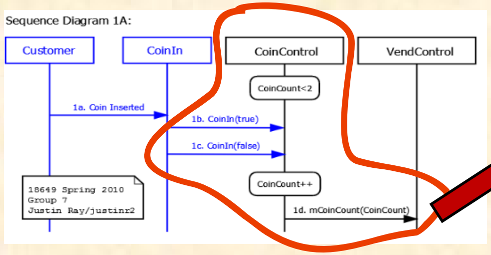


. . .

Statechart Must Exhibit All Those Behaviors

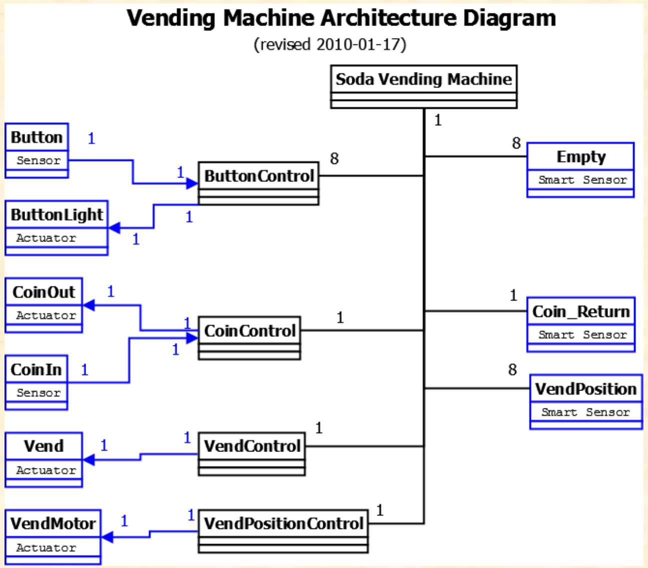


SD set specifies behaviors

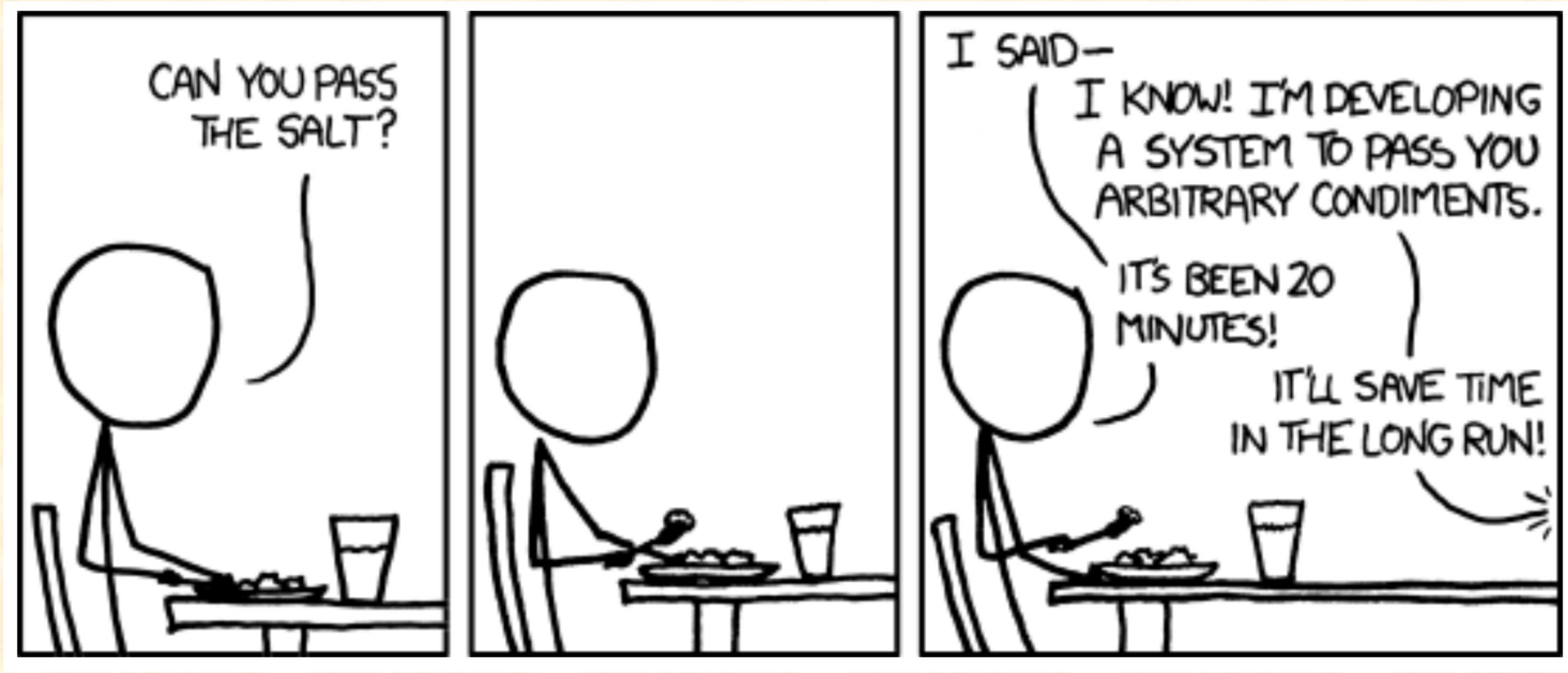


High Level Design Best Practices

- 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



https://users.ece.cmu.edu/~koopman/ece649/project/sodamachine/index.html


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/