There is never enough time to do it right the first time … but there is always enough time to do it over.
Traceability

- **Anti-Patterns:**
  - Tests don’t map to requirements
  - Requirements aren’t tested
  - Reqts/design elements missing
  - Gold plating (extra functionality)

- **Traceability**
  - Creating something traces to a quality check on the result
    - **Verification**: you did something the way you said you’d do it
    - **Validation**: the thing you created behaves the way it should
  - Ensure nothing left out; nothing added that shouldn’t be there
Traceability Examples

- **Design traceability**
  - Requirement $\Rightarrow$ design $\Rightarrow$ implementation
  - Requirement $\Rightarrow$ test

- **SQA traceability**
  - Confirm process is being carried out
  - Process step $\Rightarrow$ document/artifact $\Rightarrow$ quality metric

- **Safety analysis traceability**
  - Confirm all hazards successfully mitigated
  - Hazard $\Rightarrow$ requirement $\Rightarrow$ mitigation $\Rightarrow$ validation

- **Defect traceability**
  - Ensure that all bugs are fixed
  - Bug report $\Rightarrow$ defect identified $\Rightarrow$ fix task $\Rightarrow$ code check-in $\Rightarrow$ regression test
# REQUIREMENTS TRACEABILITY MATRIX

**Project Name:** Online Flight Booking Application

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business Requirement ID#</strong></td>
<td><strong>Business Requirement / Business Use case</strong></td>
<td><strong>Functional Requirement ID#</strong></td>
</tr>
<tr>
<td>BR_1</td>
<td>Reservation Module</td>
<td>FR_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR_2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR_3</td>
</tr>
<tr>
<td>BR_2</td>
<td>Payment Module</td>
<td>FR_4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR_5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR_6</td>
</tr>
</tbody>
</table>
Traceability for Hazard Mitigation

Mader et al. 2013: http://doi.ieeecomputersociety.org/10.1109/MS.2013.60

Hazard 101: Moving the patient’s arm at an excessive velocity

Fault F1: Velocity sensors fail to sense excessive velocity
  - Req 1: A system test must be run prior to each use to check that sensors are operating correctly
  - Test case T1

Fault F2: Configuration component fails to update correct velocity constraints
  - Req 2: All sensors must be duplicated
    - Test case T2
  - Req 3: Automatic stoppage of the robotic arm if arm velocity sensors disagree on current velocity by more than \( x \) m/s
    - Test case T3
  - Req 4: Current velocity constraint is displayed on the monitor
    - Test case T4
  - Req 5: Current velocity constraint must match patient’s personal record
    - Test case T5
  - Req 6: Current velocity constraint must fall under maximum allowed velocity
    - Test case T6
Traceability Best Practices

- Trace everything in design package
  - Even simple traceability checks can find problems
    - **Gold plating**: design item not traced to a requirement
  - Everything in design has an ID tag for traceability
    - Map left and right sides of V to each other
    - Map each layer of V upward and downward
  - Trace changes to see what else they affect

- Traceability pitfalls
  - Making granularity of trace IDs too big causes problems
  - Re-numbering breaks auto-generated document sections used as trace IDs
  - Don’t use the wrong tool
    - Spreadsheets don’t scale to big projects
    - Big project tools might be overkill where a spreadsheet approach will do
I USED TO THINK CORRELATION IMPLIED CAUSATION.

THEN I TOOK A STATISTICS CLASS. NOW I DON'T.

SOUNDS LIKE THE CLASS HELPED. WELL, MAYBE.

https://xkcd.com/552/