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 ➔ design ➔ implementation
  - Requirement ➔ test

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

- **Safety analysis traceability**
  - Confirm all hazards successfully mitigated
  - Hazard ➔ requirement ➔ mitigation ➔ validation

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

**Project Name:** Online Flight Booking Application

<table>
<thead>
<tr>
<th>Business Requirements Document BRD</th>
<th>Functional Requirements Document FSD</th>
<th>Test Case Document</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business Requirement ID#</strong></td>
<td><strong>Functional Requirement ID#</strong></td>
<td><strong>Priority</strong></td>
</tr>
<tr>
<td><strong>Business Requirement / Business Use case</strong></td>
<td><strong>Functional Requirement / Use Case</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Test Case ID#</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BR_1 Reservation Module</td>
<td>FR_1 One Way Ticket booking</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>FR_2 Round Way Ticket</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FR_3 Multicity Ticket booking</td>
<td>High</td>
</tr>
<tr>
<td>BR_2 Payment Module</td>
<td>FR_4 By Credit Card</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>FR_5 By Debit Card</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>FR_6 By Reward Points</td>
<td>Medium</td>
</tr>
</tbody>
</table>

https://goo.gl/yuFby © 2017 Philip Koopman
Traceability for Hazard Mitigation

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 mps
- Test case T3
- Req 9: Current velocity constraint is displayed on the monitor
- Test case T4
- Req 10: Current velocity constraint must match patient’s personal record
- Test case T5
- Req 11: Current velocity constraint must fall under maximum allowed velocity
- Test case T6

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