
Rethinking Validation

Spending Smart on Quality

Phil Kirsch
Statistical Center for HIV/AIDS Research and Prevention
Fred Hutchinson Cancer Research Center
Seattle, WA, USA



Validation can be

- Value-Added

- Identify potential problems
 - Staff training

- Essential documentation

- Irrelevant

- The test system bears little resemblance to production
 - System design flaws are overlooked

- Expensive

- Counter-productive



It all depends on understanding

- The goal of validation
- The setting
- The system



The goal of validation is demonstrating that a system

- Functioned as intended
- Produced reliable data
 - Accurate within a specified error rate
 - With a clear history of
 - Problems encountered and
 - Resolutions implemented



The setting includes

- Users
- Software
- Hardware



The System

How the users interface with the hardware and software to verify data



Bad validation practices often

- Involve approaches developed in manufacturing plants and laboratories
- Ignore the benefits and risks of human intervention
- Start with the assumption that we should be able to build an unbreakable system.



What other models are available?

- Airplane pilots use checklists to provide a systematic review of complex systems when human lives are at stake.
- Most computer professionals instinctively watch key systems to assure themselves that a network is functioning normally.
- People drive defensively in the hope of avoiding accidents and carry insurance for times when their best efforts fail.



To have confidence in clinical data, we need

- Detailed hardware and software requirements
- Confidence in the Vendor
 - Audit
 - Reputation
- Installation/configuration instructions that are
 - Accurate
 - Complete
- System history
- Plans for backup and recovery



Software problems frequently occur because of

- Installation errors
- Configuration errors
- Use, e.g.
 - Trying to make the software do things the programmer never created it to do
 - Accidentally typing the wrong key
- Unexpected events



End user testing is limited by

- **Access to the code**
It's usually proprietary
If you had the programming skill to understand it, you might write it yourself.
- **Staff time, experience and computing resources**
Imagine every possibility
Write, conduct and report appropriate test cases.



Bad validation practices can

- **Produce a lot of paper work that proves little**
- **Ignore errors not anticipated by the**
Programmers or
Testers
- **Fail to regularly verify system health**



Quality control should focus on

- Requirements
- Vendor Audit
- Installation
- Configuration
- User training
- System history including regular status reviews, and
- Disaster Recovery / Business Continuity Plans



This reduces the cost of quality

- The manufacturer already knows
how a system should be installed;
the proper uses of any configuration option; and
what skills/information administrators and users need.
- The client organization's job is to translate
that into
Hierarchical work instructions (checklists)
Requirements that explain the purpose of each step



Users then become responsible to show they

- Installed and configured the system according to instructions provided
- Had an appropriate rationale for each configuration choice selected.
- Completed appropriate training, and
- Regularly checked appropriate measures of system health
- Adequately Documented
 - System Changes and
 - Unexpected events



Related SOPs include

- Establishing CDM System Requirements
- Qualifying Software Vendors
- Training
- System Design
 - Translating manufacturer recommendations into specific, documented configurations
 - Translating installation instructions into checklists
- Documenting System Status
- Documenting Unexpected Events



This demonstrates that quality
was present at each stage of
implementation and use, not an
after thought or side activity.

