Validation Strategies for
Mobile Devices and Applications

• The contents of this presentation represent the opinion of the
  speaker; and not necessarily that of his present or past
  employers.
About the Author

- **20+** years of experience in the **medical devices, pharmaceutical, biotechnology, and consumer electronics** industries
  - MS Biotechnology, emphasis in **Biomedical Engineering**
  - BS Mechanical Engineering
  - ASQ **Certified Quality Engineer** (CQE)

- I have led validation / qualification efforts in multiple scenarios:
  - High-speed, high-volume automated manufacturing and packaging equipment
  - Enterprise resource planning applications (i.e. SAP)
  - Manufacturing Execution Systems (MES)
  - IT network infrastructure
  - Quality Systems Software
  - Laboratory : information systems and equipment / instruments
  - Product improvements, material changes, vendor changes
  - Mobile applications

- Contact information:
  - **Raul Soto** rsoto21@gmail.com

Contents

1. **Introduction**
   - Types of Mobile Apps
   - Why are Mobile Apps different?
2. **SDLC Approach**
3. **Validation Deliverables**
   - Assessments
   - Requirements
   - Design
   - Architecture
   - Traceability Matrix
4. **Development**
   - Prototypes
   - Iterative testing
5. **Testing**
   - Environments
   - Development testing
   - Testing protocols and strategies
6. **Going live**
   - Governance
7. **Examples of GXP mobile apps**
Assumptions

- Audience has general knowledge of computerized systems validation and associated regulations.
Various Types of Mobile Apps

• **1. Non – Regulated Apps**

• **2. GXP-Regulated Apps, Internal to the company**
  – Support Manufacturing or Clinical Operations, Quality Systems, etc.

• **3. Mobile Medical Apps, Stand-alone**

• **4. Mobile Medical Apps, Connected / Interfaced to a device**

  • **GXP**: Underlying international pharmaceutical, medical devices, biologics, or other life sciences requirements such as the US FD&C Act, FDA regulations, EU Directives, Japanese regulations, other applicable national legislation or regulations.
  – **X**: Manufacturing, Laboratories (pre-clinical), Clinical, Pharmacovigilance, etc.
Mobile Apps

• FDA Definition:
  
  • Software application that can be **executed (run) on a mobile platform**
    – (i.e. a handheld COTS computing platform, with or without wireless connectivity),
  
  • or a **web-based** software application that is **tailored to a mobile platform** but is executed on a **server**.
  
  • If it is used to **support** GXP operations, or the quality system, the app needs **validation**

• Source: FDA Mobile Medical Applications, Guidance for Industry and FDA Staff

Mobile Medical Apps

• FDA Definition:

  • Mobile app that meets the definition of Medical Device in Section 201(h) of the US Food, Drug & Cosmetics Act:
    
    – intended for use in the **diagnosis** of disease or other conditions
    – or in the **cure, mitigation, treatment**, or **prevention** of disease
    – affect the **structure** or any **function** of the body

• Source: FDA Mobile Medical Applications, Guidance for Industry and FDA Staff
Intended Use

• The **Intended Use** of a mobile app determines if it’s considered a Medical Device

• “Objective intent of the persons legally responsible for the labeling of devices”

• 21 CFR 801.4: Intended use is shown by:
  – labeling claims (i.e. 510k, PMA for devices)
  – advertising materials
  – oral / written statements by manufacturers or their representatives

• FDA oversight is based on **functionality, not** on platform

• Source: FDA Mobile Medical Applications, Guidance for Industry and FDA Staff

Our Main Focus

**Mobile applications** that are used at regulated companies:

• as a **component** of a GXP regulated computerized system, supporting
  – manufacturing operations
  – clinical operations
  – distribution
  – the quality system

• as an **interface** to an instrument or control system
Why are Mobile Apps different?

Figure 2.1: Example User Interface of a Mobile Device

Source: GAMP 5

Risk-Based Approach to Regulated Mobile Applications
Mobile Apps vs Typical GXP Software in Industry

- **Less control over OS platform**
  - Apple, Google / phone carrier (for Android) control timing of OS upgrade releases
  - Users control *when* to allow OS upgrades
  - You can't prevent users from upgrading OS until you've tested your app on the new OS release.

- **Less control over other applications running on device**
  - Especially when GXP apps run on personal devices, not company-issued
  - Validated app may have conflict with user-installed apps

- **Less control over app distribution**
  - People can install version in Apple Store / Google Play Store, before you've validated it (i.e. Citrix Receiver)

- **Portability**
  - Mobile devices are at greater risk of theft, damage from impacts.

- **Less hardware standardization**
  - Happens mostly when GXP app runs on personal devices, not company-issued
  - Particularly on Android, wider variety of hardware platforms

- **Less control over device hardware configuration**
  - Especially when GXP apps run on personal devices, not company-issued

- **Use of mobile hardware capabilities such as:**
  - Accelerometer / motion sensors
  - GPS
  - access to cellular voice and data networks
  - SMS capability
  - touch-screen
  - built-in camera and microphone
  - Bluetooth
  - NFC
  - IR blaster
  - stylus
Mobile Apps vs Typical GXP Software in Industry

• Battery capacity
  – Battery may drain fast when transmitting and recording data frequently
  – Quad-core / Octo-core:
    • ½ CPU cores are faster, higher-performance, high power drain
    • ⅝ CPU cores are slower, lower-performance, low power drain

• User-related risks
  – Users may unintentionally allow an unvalidated update to an app
  – Users may turn ON automatic updates, and allow unvalidated updates to load
  – Jailbreaking
  – Internal storage may get filled with user files (i.e. music, photos), no space available for RAM-intensive functions
  – Users may adjust configuration settings in device that impact the performance of the validated app
  – Users may dismiss / ignore error messages, warnings, etc. from app

These differences between mobile apps / mobile devices vs typical PC-based or server-based apps, introduce new risks that need to be managed.
Life cycle approach to Mobile Apps validation

What is an SDLC?

- **SDLC**: Systems Development Life Cycle
- Series of *steps / phases* that provide a model for *development* and *lifecycle management* of an application or piece of software.
- Software validation is NOT a one-shot deal.
- Validated software and its documentation require *continuous* updates and improvements for the *life of the system*
- End-of-life of the app has to be **managed**
  – Data migration or archival
  – Hardware decommissioning
  – App remote uninstall?
Typical List of Validation Deliverables

- **Assessments**
- **Validation Plan**
- **Requirements**
  (User, Functional, Compliance, Performance, etc.)
- **Design**
  (DS, FDS, TDS, DDS)
- **Traceability Matrix**
- **Testing protocols and scripts**
  – Development *
  – Integration *
  – Installation
  – OQ / PQ or System Testing / UAT
- **Testing Reports**
- **Training**
- **Governance - Procedures**
  – Operational
  – Administration
  – Change Control
- **Validation Report**
Typical SDLC Validation Deliverables

- Assessments
- Validation Plan
- Requirements
- Design

- Development
- SAT / FAT
- IQ
- OQ / System Testing
- Traceability Matrix
- Procedures
- Validation Report
- PQ / UAT

---

IVT San Diego

(c) 2017 Raul Soto
Quality Risk Assessment  
(GAMP 5 model)

1. Perform initial risk assessment, determine system impact  
   – FMEA: Severity, Occurrence, Detection

2. Identify which functions impact Patient Safety, Product Quality, Data Integrity

3. Identify controls (see next slide)

4. Implement and verify controls

5. Monitor effectiveness of controls, identify residual risks

Possible Controls

• Restriction of intended use
• Restrict intended users
• Select or reject specific OS / hardware platforms
• Modify app functionality and features
• Modify system architecture or design
• Handshake protocols to ensure a user’s app / hardware / OS is supported and validated
• Improve user instructions
• Increase rigor of testing
Assessments

- **Vendor** Assessment
- **Quality Risk** Assessment / FMEA
- **Compliance** Assessment (i.e. GxP, SOX, etc.)
  - 21 CFR Part 11 (or Annex 11) **ER/ES**
- **GAMP** category -> helps you determine which deliverables you really need
- Information **Security**
- **Safety** / EHS
- others

Compliance Assessment

- Documented assessment, *approved* by subject matter experts (SMEs) in all relevant areas, stakeholders, and management.

- You should KNOW what types of data your system will create / transmit / manage, and what are the legal / regulatory **requirements** for each type of data.

- Results of this assessment should be used to design your validation and testing strategy.
Compliance Assessment

• Some regulations your app may be subject to:

  – **Medical Devices (21 CFR 807, 812, 814, 801, 803, 50, 54, 56), ISO and ICH equivalents:**
    • Is your app used to **diagnose** a disease or other conditions?
    • Is it used to **cure, mitigate, treat, or prevent** a disease or injury?
    • Does it affect the structure or any function of the human body?
    • Does it control, or is integrated with, a medical device?

  – **GxP (21 CFR 820, 210, 211, 606), ISO and ICH equivalents:**
    • Does your app handle manufacturing / batch records, clinical and preclinical research, quality trends, NCs, CAPAs, device design history files, post-market vigilance data, etc.

• Some regulations your app may be subject to:

    • Is your app creating / modifying / storing / transmitting electronic records?
    • Is your app implementing electronic signatures?
    • Is your app used to enter data into a validated computerized system (i.e. LIMS)
Compliance Assessment

• Some regulations your app may be subject to:
  
  – Financial Data (US Sarbanes-Oxley / Securities and Exchange Commission, EU equivalents):
    • [https://www.sec.gov/rules/final/33-8180.htm](https://www.sec.gov/rules/final/33-8180.htm)
  
  – European Protection of Personal Data, US HIPAA / Personally Identifiable Information (PII):
    • information that can be used on its own, or with other information, to identify, contact, or locate a single person, or to identify an individual in context.
    • Clinical trials and post-market surveillance / pharmacovigilance apps

Compliance Assessment

• Other regulations your data may be subject to:
  
  – US Drug Enforcement Agency (DEA) / EU Equivalents
    • i.e. Inventories and use of DEA-regulated substances
  
  – Occupational Safety and Health Administration (OSHA) / EU Equivalents
    • i.e. records of safety incidents, trends
  
  – US Department of Agriculture (USDA) / EU Equivalents
    • i.e. records of any materials derived from animals used in vaccine production, pre-clinical testing, etc.
  
  – US Environmental Protection Agency (EPA) / EU Equivalents
    • i.e. all records related to compliance with EPA regulations
  
  – U.S. Patient Protection and Affordable Care Act (PPACA) / EU Equivalents
    • If the app is used to capture, track, store, approve, monitor, report or transmit payments or other transfers of value to healthcare professionals, healthcare institutions, and members of the scientific community, payors, purchasers and patient advocacy groups.
GxP Assessment

If your app performs / supports any of the following activities, it falls under the scope of FDA’s GxP regulations:

- Manage / control access to a regulated computerized system
- Maintain training records
- Host or support a regulated computerized system (i.e. infrastructure, access control, backup)
- Control, measure, monitor, support manufacturing systems, process parameters, or data that directly influences product quality
- Store and preserve status of interim and final product and/or product samples
- Generate or maintain product master records (i.e. Device History Records)
- Create or trace GMP batch records
- Impact product or human safety
- Impact product efficacy

This list is NOT exhaustive!

GxP Assessment (cont.)

- Maintain or track nonconformances, product complaints, product disposition, adverse event records, change control records, or commercial product recalls
- Transmit or submit records to a regulatory agency
- Archive regulatory data
- Maintain master schedules
- Contain audit records required by a predicate rule
- Support claims of product efficacy
- Support claims of product safety
- Maintain, analyze, report, or transmit GLP non-clinical study data
- Maintain, analyze, report, or transmit GCP clinical data
- Maintain, analyze, report, or transmit safety surveillance data, safety cases, or REMS (Risk Evaluation and Mitigation Strategies) data

This list is NOT exhaustive!
GxP Assessment (cont.)

- Quality Systems support for medical devices, such as
  - CAPA management
  - Change control management
  - Non-conformances management
  - Training management
  - Purchasing Controls data management
- Product registration and Licensing management
- Data Analysis or reporting that leads to Quality decisions
- Provide support to installation or servicing of medical devices, and their components
- Drug sample distribution and management
- Management of instructions for use (IFU), electronic labeling, product inserts

This list is NOT exhaustive!
Validation Plan

- System Overview
- Scope of Validation
- Roles and Responsibilities
- Assessments Required
- Validation Deliverables, author and approvers
- Validation and Testing Strategy
- Acceptance Criteria
- How to deal with discrepancies, defects, changes in scope, changes in strategy

• This is where you establish WHAT needs to be validated, and HOW it will be validated.
• Get everyone’s buy-in and commitment to the Validation Plan from the beginning
• Update as needed, following your organization’s formal change control process for documentation
• Define clear expectations for all roles
  – Definitions of deliverables, author and approvers for each
Validation Report

– Final scope
– List of Actual Deliverables
  • Document Numbers
  • Date of Completion
  • any additions or deviations from planned
– Summary of testing results per protocol; any additions or deviations
– Summary of defects and discrepancies, and resolution status
– Statement declaring the validated state of the system
– Release to Production

Resources

• You will need resources from **multiple** disciplines (i.e. QA, IT, Engineering, Operations, technical writers, etc.)

• Plan **which** resources you need, **when** will you need them, and for **how long**

• This will help you budget for consultants, and borrow internal resources. “*I need two validation resources, 25-30% of their time, during August and September*” is a different conversation than “*I need to borrow some of your people for the Mobile App ABC project*”

• PM tools -such as MS Project- help
Purchasing Requirements

- **NOT YET** the same as your validation requirements
- Develop a high level list of requirements to be used as part of the purchasing decision
- **Evaluate alternatives / choices** based on compliance with these requirements
- Make sure you have a good idea of how much **configuration** and/or **customization** is needed to meet your main requirements
  - **Recommendation:** try to **MINIMIZE** the amount of customization / custom code
- These requirements will also be the basis for **FAT / SAT**
- **SLA : Service Level Agreement**
Requirements Specifications

• Define what you want your Mobile App to do

• **Types:**
  – User Requirements
  – Functional Requirements
  – Compliance / Regulatory Requirements
  – Performance Requirements
  – Security Requirements

• **Classify** by *type* and *criticality*
  – *Low* criticality: nice to have
  – *Medium* criticality: important but not critical
  – *High* criticality: **must** have
  – All regulatory requirements should be classified as **HIGH criticality**

• Results of *Assessments* should be used to develop *Requirements*

Requirements Specifications

**Requirements should be**

• Clear
• Complete
• Correct
• Consistent
• Concise
• Prioritized

• Relevant
• Feasible
• Verifiable
• Modifiable
• Traceable
• Unique
Requirements Specifications

• Expected system inputs and outputs

• Functions that the App will perform

• Proposed *Workflows* and *Use Cases*

• Data Management

• Reports the system is expected to generate, contents

• Definition of all external and user interfaces

• How users will interact with the app

• Required alerts, alarms and checks

• Performance: data throughput, reliability, timing, required response time

• Coding standards – could be a separate document
Requirements Specifications

• Connectivity
  – What is needed (Wi-fi, cellular, RFID radio beacon, etc.)
  – What app should do when connectivity is not available

• Ergonomic and human factors
  – Is all screen content readable on various screen sizes and resolutions?
  – Are links and soft buttons sufficiently far apart?
  – Is selecting the desired option straightforward?

• ISO / IEC 62366:2015 Application of usability engineering to medical devices
  – Provides guidance for human factors engineering in medical device
  – Even if your app is not a medical device, guidance is good

Requirements Specifications

• **Intended operating environment** (hardware platform, software operating system, middleware, database software, etc.)
  – Single mobile OS and hardware platform (i.e. iOS iPad only)
  – Single OS, multiple hardware platforms (i.e. iOS iPad and iPhone)
  – Multiple OS and hardware platforms (i.e. iOS and Android devices)
  – Multiple flavors of a single OS (i.e. **Android** is tweaked by hardware manufacturers and phone carriers), multiple hardware platforms

• Backup, restore, disaster recovery

• User Roles that need to be defined in the system

• Any safety-related specifications

• Timestamps: local time vs UTC

• Part 11: Electronic Records & Signatures requirements
Requirements Specifications

• Display requirements
  – Minimum screen size and resolution
  – Minimum font size for legibility in mobile device screen
  – Portrait / landscape / both?
• Use of mobile device – specific capabilities
  – Cellular
  – NFC, Bluetooth, IR blaster
  – GPS
  – Motion sensors / accelerometer
  – Stylus
  – Camera / microphone

Requirements Specifications

• Error handling during data entry:
  – In the event of:
    • Wi-fi / cell signal loss
    • Device runs out of battery power
    • User presses HOME button by mistake
    • User exits out of app
    • App closes
    • Device is turned OFF
  – What happens to your data?
    • Do you keep the data entered up to that point?
    • Do you lose all the data entered in that session?
    • Does any data corruption occur?
Requirements Specifications

• For apps that interface with associated device / hardware
  – Encrypted handshake with device?
  – Need for calibration?
  – Limiting the mobile devices that drive the hardware
    • Ensure other mobile apps or devices can’t work with the equipment
    • Safety considerations
• Interfaces with other mobile apps
  – eMail
  – SMS
  – Calendar
  – Etc.

Requirements for Data Management

• Assessment should consider:
  – What data will be collected?
    • Data that needs to be pre-populated prior to installation and/or operation
    • Data collected as part of daily use
  – How / Where will it be stored?
    • On the device? On a server? Cloud storage?
    • Storage site under company’s direct control, or outside?
  – Who will have access to the data, and why?
    • National / local privacy laws compliance
    • Check with company’s Privacy, IT Compliance, Legal departments
Requirements for Data Management

– Retention period for the data
  • Data retention should comply with company’s data / document retention procedures
  • If data resides on user’s mobile device, and needs to be deleted, app should include mechanisms to enable and enforce data removal.

– Disaster recovery / business continuity
  • If it’s critical that data remains available, ensure it’s backed up outside the device
  • Should be able to restore data to device in a timely manner

– Define your RTO / RPO
  • Recovery Time Objective: how quickly app needs to be recovered
  • Recovery Point Objective: how much data loss can be tolerated

App Deployment Strategy

• How will users get your mobile app?
  • Commercial App store?
    – iOS: Apple App Store
    – Android: Google Play Store
    – Windows: MS Store
  • Your Company’s own App store?
    – iOS: need Enterprise Developer Account (contact Apple for details)
    – Android: no permission needed
    – Windows: no permission needed
  • Website / network folder
    – iOS: not possible
    – Android: possible
    – Windows: possible
Design and Architecture

Design Specification

- **Design Specifications** describe the actual software solution, as coded and configured
- **Address all requirements**, individually or in groups
- **Design documentation** may include one or more of the following:

<table>
<thead>
<tr>
<th>Design Specification</th>
<th>Software Design Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detailed Design Specification</td>
<td>Hardware Design Specification</td>
</tr>
<tr>
<td>Technical Design Specification</td>
<td>Architecture Design</td>
</tr>
<tr>
<td>Database Design Specification</td>
<td>Security Design</td>
</tr>
<tr>
<td></td>
<td>Interfaces</td>
</tr>
</tbody>
</table>
Design Specification

**Design documentation** should include, at least:

- System architecture
- Modules
- Screens
- Formulas, algorithms, and logic used
- Data structures, data flow diagrams
- Supporting software that is required for App operation
- Hardware required
- Parameters that are measured or recorded
- Reports generated
- Definition of control and data variables, where used

- Messages: Errors, alerts, alarms, warnings
- Actual workflows and use cases, as configured (or coded)
- User roles, as configured
- Physical security
- Information security
- Actual reports as configured
Architecture

• Key components:
  
  – App / Device Connectivity
    • How app will use connectivity architecture
  
  – Device Components
    • Which device components are required by app
  
  – Mobile client approach
    • Client architecture required by app

Architecture: App Connectivity

• Design Spec should *document how* app will use connectivity architecture

• Long-range:
  – 2.5G (GSM/GPRS/EDGE), 3G (HSPA), 4G (LTE cat 4, cat 6) *
  – WLAN, Wi-Max
  – Typically used to connect to mobile networks

• Short-range:
  – Bluetooth, NFC, Wi-fi
  – Typically used to connect to local networks or other devices

* Definitions:
  http://www.arcelect.com/2g-3g_cellular_wireless.htm
Architecture: Device Components

- Design Spec should document the actual hardware specifications or components, should be equal or better than what is required by URS
- Screen size, memory, processor power, sensors such as GPS or motion sensors
- LR / SR connectivity required (see previous slide)
- Operating system(s): Apple iOS, Android, Windows, etc.
- Browser:
  - Does app require a standard browser?
  - If yes, does app work through a specific mobile browser, or is any browser ok?

Architecture: Client Approach

- Design Spec should document the client architecture, and focus on areas of higher impact
- **Thin client**
  - Essentially a browser front-end, configured specifically for required functionality
  - All data processing and storage occurs on a web server, NOT on the mobile device
  - Back-end complexity is higher
- **Thick client / Rich client**
  - Mobile app residing on mobile device carries out functionality
  - All or some data stored locally, there may be some network or cloud data storage
  - Data processing may be distributed with a server
  - Back-end complexity is lowest
Architecture: Client Approach

• **Hybrid Approach**
  – Mobile client can be designed as a native app with embedded browser
  – Browser interacts with remote server
  – Data processing and/or storage distributed between app and server
  – Back-end complexity between thin and thick

Handshaking

• When an App requires a ‘handshake’ with a centralized server in order to run
• Can be used to:
  – force users to install upgraded version
  – disable the app in OS and hardware platforms no longer supported
  – prevent use in unapproved platforms

• If used, should be explicitly stated in user terms and conditions, licenses, contracts.
• Risks:
  – App will not work when cell/wifi signal is weak, intermittent, or not present
Design Review

- Evaluate deliverables vs standards and requirements
- Identify issues early
- Propose corrective actions
- Planned and systematic review of
  - Specifications
  - Design
  - Development
- Planned for suitable stages during the project life cycle

- Guidance: ISO13485, 21 CFR 820 for Medical Devices

Requirements Traceability Matrix

- The RTM should enable us to trace:
  - From individual Requirements to their specific Design Element(s) where the requirement is addressed
  - From Design Elements to the specific Test Script(s) where they are being challenged.

- We should be able to trace back and forward

  Requirement ↔ Design elements ↔ Testing
Requirements Traceability Matrix

- Requirements => start RTM draft
- Design documents => update RTM draft
- Test scripts => finalize RTM

- Final RTM should be approved before the system is released for production
- RTM should be updated as part of system change control every time requirements, design, or test scripts are changed, added, removed.

Living Documents

- The following documents should be treated as “living documents” and maintained up to date throughout the life of a validated computerized system

- Requirements
- Design
- Traceability Matrix
- Risk Assessments
Development / Configuration

• Development testing occurs before validation
  – COTS functionality typically needs configuration
  – Vendor or hired consultants
  – Unit testing, test configuration, integration testing
  – If custom coding is required, development testing should be more comprehensive
  – Use bug tracking software (i.e. FogBugz, Redmine, De PPPP) to keep track of bug resolution
  – Does not need to be formally documented, but IT HELPS
  – Have Coding standards and follow them!
Development Testing

- **Unit Testing:**
  - Test individual software objects / modules / units of code, by themselves
  - Known inputs vs expected outputs

- **Integration Testing:**
  - Individual software objects / modules / units of code are combined and tested as a group
  - Outputs from some objects are inputs for others

Prototypes

- Used to evaluate the following:
  - Acceptability of user interface
  - Performance of critical algorithms
  - Suitability of overall software solution
  - System performance, speed, capacity, etc.

- Prototype evaluation
  - Go through the Use Cases for the mobile app
  - Users can perform key tasks, understand how to
  - Users are clear on what actions are available
  - Users can read information as presented by screen, interface
  - Users can interact correctly with interface items (buttons, pick lists, gestures)
Agile

Testing and Validation
Testing

• Protocols, Test Scripts, Reports
  – Pre- and post- approved
  – Consistency in roles pre- and post- approving is important
  – Avoid conflicts of interest: Approvers cannot execute, testers can’t approve
  – Dependencies and order of testing must be clear, evident, obvious
  – Enforce Good Documentation Practices (GDP)
  – Enforce the use of standard templates
  – Keep evidence (printouts, screenshots, labels, etc.)

Testing Environments
for the server component of thick or hybrid clients

• Development / Sandbox
  – For DEV testing
  – Qualification not required

• Test / Staging
  – For your formal validation (OQ, System Testing, UAT)
  – Qualified
  – Functionally equivalent to your PROD environment

• Production Environment
  – Live system
  – Performance testing
  – Qualified
Testing: IQ

• **Installation Qualification**
  
  – Document / verify the correct *installation* and *configuration* of all software and hardware components, as per the Design Specification
  
  – List **actual software components** and objects installed: name, version, location
  
  – List **actual hardware installed**: name, model, quantity, S/N, location
  
  – **Turn-key test** to ensure the system is ready for OQ / System testing

Testing: System Testing / UAT Model

• **System Testing**
  
  – Test and verify that the system as integrated is functionally complete
  
  – Challenge compliance with *functional* and *non-functional* specifications
  
  – Integration of App with interfaces
  
  – You can test individual functionalities, then end-end process workflows
System Testing

- Tests functional requirements, configuration, security, ER/ES, compliance
- May include positive testing, negative testing, boundary testing, interface testing

- Positive Testing:
  - Ensures that system performs as intended, using normally expected inputs

- Negative Testing:
  - Ensures that system doesn’t accept invalid inputs

- Boundary Testing:
  - Challenges that performance is as expected when specific variables are set to their max / min values.

- Interface Testing:
  - Tests that system components can pass data correctly to one another

Testing: System Testing / UAT Model

- User Acceptance Testing
  - Testing focused on user-related requirements
  - Challenge that the mobile App is capable of supporting your normal business process
  - Test cases should address ease of use from the standpoint of operators, QC technicians, etc.
User Acceptance Testing (UAT)

- Tests functional requirements, configuration, security, ER/ES, compliance
- May include positive & negative testing, business process testing, end-to-end stress testing, performance testing

- Business Process Testing:
  - Verifies that the system works as intended following business process flows

- End – to – End Testing:
  - Verifies that the system is capable of supporting the intended process flows, from beginning to end; ensure data integrity and that the correct data passes between components and interfaces.

- Performance Testing:
  - Verifies system stability, resource usage, and responsiveness under specific workloads.

- Stress Testing:
  - Tests system’s performance beyond the limits established in the specified requirements

Testing: OQ / PQ Model

- Operational Qualification
  - Ensure system can perform under controlled conditions, non-saleable product
  - Document that system is installed and configured as per Design Specifications
  - AND complies with requirements as per Requirements Specifications
  - Testing will challenge all requirements, interfaces, reports, etc.
  - Can test each subsystem individually
  - Positive vs Negative testing
  - Formal change control
Testing: OQ / PQ Model

- **Performance Qualification / Production Qualification**
  - Test that system is able to function under *normal production* conditions
  - End-to-End challenge of the complete workflows
  - **Performance** testing / load testing
  - Can use saleable product

Testing Execution

- Ensure test sets and scripts are executed in correct order
- **GDP (Good Documentation Practices)** are IMPORTANT
- Include evidence: screenshots, printouts, labels, etc.
- Changes to test scripts must follow formal documentation management (version up, reapprove, etc)
- Coordinate execution dates / times with owners of systems that your app interfaces with
- Follow process for handling testing defects / deviations
- If you execute PQ / UAT with real product, QA must provide disposition of such product
Testing Tools

- There are electronic testing tools that can be used in substitution of paper test scripts
  - Example: HP Quality Center/ HP ALM, Valgenesys
- **MUST validate** these tools BEFORE you use them
- **Pros**: reduce GDP errors, standardize testing process, enforce use of correct templates, enforce approvers rules, manage test defects, can search for documents electronically
- **Cons**: administration, training, cost, less flexibility, maintenance, fixes & upgrades
- Tools become your **official repository** of validation documentation
- 21 CFR Part 11 full compliance may require **additional** wrap-around software solution (HPQC)

Positive vs Negative Testing

- **Positive Testing**
  - Test cases that verify your **app functions work as expected**
  - Enter valid values / commands and ensuring the app produces correct results
  - If the app performs calculations, verify results against manual calculations
  - Include borderline cases
  - If app includes steps where there is branching logic, **test all paths**
Positive vs Negative Testing

• Negative Testing
  
  — Test cases that use **deliberately incorrect** inputs and actions
  — Ensures the app is robust. App should handle incorrect inputs:
    • without crashing
    • Without producing false/ incorrect outputs
    • without compromising data integrity

  — Test that it is not possible to bypass security

Negative Testing Examples

• **Required data entry**: ensure mandatory input fields cannot be left blank

• **Field Type test**: ensure data entered is of the correct type (i.e. dates, alphanumeric, etc.)
  
  • Enter alpha or special (@%^#) characters on a numeric field
  • Enter a decimal number in an integer field
  • Enter negative numbers in a positive numbers field
  • Enter nonsensical inputs (i.e. ASCII CRTL codes)
  
  • [http://academic.evergreen.edu/projects/biophysics/technotes/program/ascii_ctrl.htm](http://academic.evergreen.edu/projects/biophysics/technotes/program/ascii_ctrl.htm)

  — **Field Size test**: ensure user can only enter the specified number of characters in a field
Negative Testing Examples

• **Date bounds test**: test upper and lower bounds for dates.

• **Numeric bounds test**: for example, if the number in a field should be between 10 and 50, the app should not accept numbers outside that range.

• **Numeric limits test**: make sure the app doesn’t crash or produce incorrect output when entering a very large number.
Project Change Management

• You must have a **formal change control** process **while** in the project, not just after go-live.

• **Design freeze** date – enforce it!
  – Any design changes requested **after** this date => after Go Live
  – Except for critical (regulatory, business) changes

Going Live - Governance
Governance

• Procedures
  – App / System Administration and Operation
  – Change Control and Configuration Management
  – Maintenance
  – Training
  – Security and Access Control
  – Data Backup and Recovery
  – Data Retention, Migration, or Destruction
  – Management of Distribution Channels
    • App distribution process must be in control
    • others

Governance

• “Hypercare”

• SLA with vendors

• Change Control

• Support structure

• Periodic Review (and revalidation)
Governance

• **EULA / Legal Notice / Terms of Use**
  – EULA: End User License Agreement
  – Users must agree to these rules, terms, and guidelines to get access to the mobile app
  – Disclaimers
  – Must be accessible from the App’s main screen
  – Must be readable from the device’s screen
  – Your LEGAL department must review this
  – Users should accept it (i.e. checkbox) before first use

Governance

• **Privacy Policy**
  – Describes what data is collected, if any; what will be done with it
  – Must be accessible from the App’s main screen
  – Must be readable from the device’s screen
  – Must comply with legal requirements of all countries where app will be used.
  – Your LEGAL department must review this
  – Users should accept it (i.e. checkbox) before first use
Change Control

- **All changes** should be assessed, documented, and if necessary: pre-approved, tested, post-approved
- MINOR and ROUTINE changes can be proceduralized

**Assessment**
- Nature and scope of the change
- Risk assessment
- Validation assessment – does my system retain its validated state after this change?
- Documentation that needs to be updated (URS, DS, RTM, FMEA, etc.)

**Testing:**
- Functional tests
- Regression tests

**Coding / Configuration Changes to App**

**Mobile OS upgrades**
- Regression testing, to ensure app is compatible with updated OS
Regression Testing

- Testing of elements / modules of a software application that were NOT part of the intended change.
- Checks that areas of the software app not involved in the change were not affected adversely
- Purpose is to ensure that your intended change did not cause any unintended issues, problems, bugs, etc. elsewhere in the computerized system.
- Often a subset of the initial validation test scripts is used as a regression test.
- The scope of regression testing should be proportional to the magnitude of the change.
- Can be automated.

Mobile Devices Internal Use Policy

- **User Responsibilities**
  - Use only approved devices
  - Don’t modify the company’s security configuration
  - Don’t circumvent a remote wipe process initiated by the company
  - Report device loss
  - Apply OS and security updates and patches only when indicated by the company
  - Enable device encryption (including microSD cards for Android)

- **Consequences of non-compliance**
  - Disciplinary action, loss of privilege to use mobile device, etc.
Examples

Example 1

• Client wants to use a Commercial Off-the-Shelf (COTS) mobile app to enter data into validated LIMS from iPads

• App can be downloaded from Apple’s App Store by anyone

• Issues:
  – 21 CFR Pt 11.10(h) : this is an input device to a validated system

  – Testing should address data integrity risks (i.e. what happens to data if device loses wi-fi, power, etc. in the middle of a data entry session)

  – Software version control: vendor can push a new version of the app through the App Store, users can install it before it’s tested / validated.
Example 2

- Custom-coded mobile app to replace a paper form used in engineering
- Web app, device-agnostic (accessible through any browser)

- Issues:
  - Does it require validation? Compliance Assessment to determine if intended use falls under GxP.
  - Custom coded app requires full validation
  - SLA with company that does the coding
  - Code Review / Coding standards: Ensure code is maintainable, well documented
  - Write all use-cases, ensure the client concurs with them, test all during development
  - Very detailed URS, don’t assume anything!
  - Where do the forms go? Database
  - Management of eSignatures?
  - Test in ALL platforms that will be allowed (i.e. iPad, Windows, etc.)
  - Screen controls should work on both touch-screen and mouse-driven interfaces

Example 3

- App to view SOPs and other controlled documents on iPad
- User scans 2D barcode that corresponds to specific documents, or types in document number
- Connects to eDMS using the employee’s log-in credentials, shows PDF

- Issues:
  - Validation is required: employees make decisions based on SOPs
  - Ensure system is able to retrieve the correct revision of the correct document
  - If file is saved to local device RAM, clean cache after a specified amount of time
  - Can documents be read properly from an iPad screen?
Example 4

• Mobile app to view and control SCADA system in a manufacturing line from an iPad
• Read-only if the iPad is not close enough to the line

• Issues:
  – *Safety*  How do you ensure an employee can’t activate line remotely while another is servicing it?
  – How does the system know an iPad is in front of / away from the line?
  – SCADA screen is designed for a full-sized monitor, is it readable from a much smaller iPad screen?
  – Is the iPad touchscreen able to replicate the mouse-driven SCADA interface?
  – Is validation required?

Mobile Apps that are NOT devices

• FDA Guidance, Appendix A
• These applications are not regulated

  – Access to electronic copies of medical literature or reference materials
  – Educational tools for medical training
  – General patient education
  – Automate general office operations in health care setting, not intended for use in diagnosis, cure, mitigation, treatment, prevention of disease.
  – Generic aids or general purpose products
**Mobile Apps for which FDA intends to exercise “enforcement discretion”**

- FDA Guidance, Appendix B
- These applications *may be* regulated
- *May* meet the definition of medical device, but are considered a *lower risk* to the public
- Approx 30 examples, including:
  - Apps to allow patients to send alerts or notifications to first responders
  - Apps that give patients health reminders
  - Checklists / questionnaires of common signs and symptoms
  - Collect blood pressure data, trend it, share it

**Mobile Apps where FDA oversight will focus**

- FDA Guidance, Appendix C
- These apps are considered medical devices and regulated as such
- Appendix provides 25+ examples, including:
  - Apps that use sensors to measure and display electrical signals from heart (ECG electrocardiographs)
  - Apps that use sensors to electronically amplify and project sounds associated with heart, arteries, veins, other internal organs (electronic stethoscope)
  - Measure physiological parameters during cardiopulmonary resuscitation
  - Record, analyze, view eye movements for use in diagnosis
  - Measure blood oxygen saturation for diagnosis
If your Mobile App is a device:

- FDA Mobile Devices Guidance, Appendix E
  - Brief summary of the requirements for Medical Devices

- Additional information is available at
  [http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/default.htm](http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/default.htm)
  - “Overview of Medical Device Regulation”
  - “How to Market Your Device”

- Contact the Division of Industry and Consumer Education:
  - Email: DICE@fda.hhs.gov
  - Phone: 301-796-7100 or 800-638-2041.

---

Apple Watch

- Apple initially intended to include functionality (i.e. glucometer) that would have made the Apple Watch a Medical Device
- More than 1 year of discussions with FDA
- Rumors: Apple decided to scale back intended use to stay away from Medical Device territory and associated FDA / international regulations

- FDA Apple Rumors: Apple Seeking FDA Approval for iWatch Sensors, Jan 2014
  [http://investorplace.com/2014/06/friday-apple-rumors-apple-seeking-fda-approval-iwatch-sensors/#.Ve3f7BHbGc](http://investorplace.com/2014/06/friday-apple-rumors-apple-seeking-fda-approval-iwatch-sensors/#.Ve3f7BHbGc)
- Apple and FDA have discussed FDA regulations regarding possible new mobile products, sensors and a glucometer, June 2014
- FDA’s Apple memo points to company’s aim to stay unregulated by the agency, June 2014
- FDA Approval Would Make Apple’s iWatch a Revolutionary Device, Sept 2014
- FDA Won’t be Regulating your iWatch, Jan 2015
AppleWatch: FDA / Apple meetings memo highlights

• Apple thanked FDA for the Mobile Medical Apps guidance. From their perspective, they said it was fair and they were pleased with the balance. Apple has received inquiries from Congress regarding whether the guidance is too heavy-handed, but Apple has responded praising FDA and the guidance document.

• Apple noted that while the guidance is a step in the right direction, industry is always going to be pushing the boundaries. Apple sees mobile technology platforms as an opportunity for people to learn more about themselves. With the potential for more sensors on mobile devices, Apple believes there is the opportunity to do more with devices, and that there may be a moral obligation to do more.

• Sensors already exist on medical devices. For instance, Apple’s devices have cameras and accelerometers. There is still an opportunity to innovate, but Apple wants to make sure they are on the side of the FDA.

• Under the current regulatory scheme, FDA will review a device based on the manufacturer’s intended use for the device. With regards to sensors, the presence of a particular sensor will not necessarily lend the device to FDA regulation. Instead, FDA would be more likely to regulate the software that puts the sensor to use, if use of the software alters the device’s use to be a medical device.

• The current mobile medical app guidance indicates that FDA does not view apps that are purely educational or informational as medical devices. Apps that actively measure something are considered diagnostic. For instance, a glucometer would be considered diagnostic because it measures blood sugar; it would not be considered merely information although it “informs” the user of the blood sugar level. The display screen of the glucometer would not be regulated, as it only receives the data and shows it. The software that does the measuring is the part that is regulated.

• FDA will regulate based on the intended use of a device. Using the glucometer example, the glucometer may be unregulated if the intent is for a user to follow their blood sugar for the purposes of better nutrition. If the glucometer is marketed for diabetics, however, it would more likely be regulated as a medical device. FDA looks at how devices are actually used. If the manufacturer advertises the device for an unapproved use, or FDA sees a lot of off-label use that is potentially dangerous, FDA may regulate after the fact.

• Apple will work closely with FDA as they develop future products. The earlier FDA is involved and advising, the less likely that Apple would be caught by surprise later when they wish to release a new product, if that product must be regulated.
References - USA

- 21 CFR Parts 211, 820, 11
  http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/cfrsearch.cfm

- FDA Mobile Medical Applications, Guidance for Industry and FDA Staff
  http://www.fda.gov/downloads/MedicalDevices/.../UCM263366.pdf

- FDA General Principles of Software Validation, Final Guidance for Industry and FDA Staff (Medical Devices)
  http://www.fda.gov/RegulatoryInformation/Guidances/ucm085281.htm

- FDA Guidance for Industry: Computerized Systems Used in Clinical Trials

- FDA Guidance for Industry: Electronic Source Data in Clinical Investigations

- FDA Medical Device Data Systems, Medical Image Storage Devices, and Medical Image Communications Devices: Guidance for Industry and FDA Staff
References - International


- International Committee for Harmonization ICH Q7 [http://www.ich.org/home.html]


Questions ?