Software in Medical Devices

Module 2:

• Device Software Development and Validation

Ali Nakoulima
Software Architect
Cerner Corporation
Disclosure: Cerner Corporation has applications pending before FDA. Ali Nakoulima is involved in the submission or direct support of those applications.

Ali Nakoulima
Software Architect
Cerner Corporation
Device Software Development and Validation - OBJECTIVES

✓ Understand most common Software Development process models
✓ Identify the key roles involved in developing Software
✓ Understand general Software Development Lifecycle
Waterfall Model - Process

- Project Planning
- Requirements Definition
- Design
- Development
- Integration & Test
- Installation & Acceptance
Waterfall Model – Overview

• Project planning: Project Managers, Lead Engineers and Solution Designers
  • Results in: solid and contracted timelines for delivery

• Requirements definition: Solution owners, Lead Engineers, Test Engineers or Analysts from Quality Assurance group.
  • Results in: Versioned and signed off document of user refined specifications

• Design: Lead Engineers, Engineers, User Experience experts
  • Results in: System, user or architecture flow diagram. Technical Implementation guide for Engineering implementation

• Development: Lead Engineers, Engineers
  • Results in: Tested and testable software that Quality Assurance can execute verification steps

• Integration & Test: Leader Engineers, Engineers, Quality Assurance
  • Results in: documented success or failure with bug tracking

• Installation & Maintenance:
  • Results in: customer driven test execution before production use
“When preparing for battle, I find that plans are useless, but planning is indispensable”

- Dwight D. Eisenhower
Agile - Terminologies

• Iteration Planning: Half-day to, 2 day long session to determine units of work, goals and objectives for a relatively short period of time (2, 3 or 4 weeks).

• User story: Succinct and precise request to the Software that translates into a requirement.

• Story points: Measure of complexity of a user story relative to a baseline.

• Velocity: Average number of story points a team can produce in a given iteration

• Capacity: Projected number of story points to be absorbed by a given team

• Iteration retrospective: Session to reflect on performance – the good, the bad and the ugly

• Scrum: Short (15 minutes), daily team meeting aimed at reporting progress and identifying roadblocks.

• Scrum master: Moderator of the scrum sessions.
Agile - Process

Agile Software Development Life Cycle

- **Concept**: Select and prioritize projects
- **Inception**: Initiate the project
- **Construction/Iteration**: Deliver working product
- **Release**: Test and deploy release into production
- **Production**: Operate and support release
- **Retirement**: Remove system from production
Agile – Overview

• Concept/Inception: Project Managers, Lead Engineers and Solution Designers
  • Results in: Iteration planning and story points generated. Commitment to deliver agreed upon functionality

• Inception: Lead Engineers, Test Engineers, User Experience experts or Analysts from Quality Assurance group.
  • Results in: Generation of use cases, design artifacts and discovery, assessment and documentation of functional and traceable requirements for testing and audit purposes. Risks and Hazards actively identified, documented, traced for testing.

• Construction/Iteration: Lead Engineers, Engineers, Test Engineers
  • Results in: Implementation and testing of acceptance criteria. May repeat as many times as necessary to deliver quality product. Testing performed at the level of traced requirements.
    • Integration & Test *: Leader Engineers, Engineers, Quality Assurance
      • Results in: documented success or failure with bug tracking

• Deploy/Production: Technology specialists, Lead and/or System Engineers
  • Results in: Documented and traceable trail of steps to accomplish effort. Monitoring health of system and newly installed software.
  • Installation & Maintenance *:
    • Results in: joint-customer driven test execution before production use
User Needs

“The most difficult part of requirements gathering is not the act of recording what the user wants, it is the exploratory development activity of helping users figure out what they want.”

Steve McConnell

“Walking on water and developing software from a specification are easy if both are frozen.”

Edward V. Berard

“A primary cause of complexity is that software vendors uncritically adopt almost any feature that users want.”

Niklaus Wirth

“The Software isn’t finished until the last user is dead.”

Sidney Markowitz
## User Needs

### Characteristics

- Often starts as what users think they want
- Foundation to the implementation of the software
- Subject to bugs too!
- Fluid and sometimes hidden.
- Example (Project Cumulus):
  - Determine criticality of reading and sharing of data.
  - What demographics are targeted? Young, older, doesn’t matter? Drives usability requirements.
  - Security restrictions relative to user’s preferences
## System and Software Requirements

### Technology

- A **System Requirements Specification (SRS)** (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application.

- Can be expressed as a network or flow diagram or as a use case diagram.
System and Software Requirements - Example
System and Software Requirements - Cumulus

![Diagram of Patient submits BP reading - simple](image-url)
Architecture and design

Characteristics

- Next to User needs, this is the most crucial step in process
- Must take into account many dimensions
- Leans heavily on best practices
- Patterns differ depending on product
  - Backend, first etc.
Architecture and design

- Security
- Deployment
- Passivity
- Infrastructure
- Networking
- Resilience
- Maintenance
- Operations
- Accessibility
- Reusability
Architecture and design – Service Oriented Architecture

SOA

Application frontend
Service
Service repository
Service bus

Contract
Implementation
Interface

Business logic
Data
Architecture and design – Cloud Cumulus

SaaS - Software as a Service
Providing software application services to end users

PaaS - Platform as a Service
Providing framework services to IT directors

IaaS - Infrastructure as a Service
Providing infrastructure hosting

DaaS - Desktop as a Service
Providing virtual desktops for enterprises
## Implementation

**Code, Code ... code**

- Must be tightly coupled with OS in use for testing and operations purposes.
- Design or team’s best practices will dictate programming language of choice.
Implementation

Testing

- Whitebox testing
- Blackbox testing
- Units:
  - Locale sensitive display
  - Appropriate conversion and decimal interpretation
- Source of Blood Pressure:
  - Systolic, Diastolic?
- Date and time handling
  - Time zone issues with patient’s timezone vs. cloud
  - Date of Birth: born yesterday in New Zealand but tomorrow in Madagascar?
Typical Software Errors

• Specification-based:
  • Considered erroneous

• Operational
  • Unexpected workflow causing unhandled exceptions

• Implementational
  • Missed the addition conditional statements
Software verification and validation

**Unit Testing**
- Testing occurs at or near the actual code for software
- Requires speed but also attention to details
- Several frameworks exist

**Integration Testing**
- Testing occurs at the specification level
- Usually involves QA
- Test closer to real life scenario

Change = Test
✓ Software Development processes vary in their application.
✓ Well defined user requirements are the pillars of a successful software project.
✓ Next to user requirements is a thorough design that takes into account many variables.
✓ Testing is crucial and really never stops.
References

✓ http://wiki.c2.com/?QuotesOnRequirementsAndUsers
✓ Cerner Training