How to Simplify Connected Medical Device Software Integration and Certification

Oct. 2014
Your Presenters

Chris Ault – Product Manager, QNX Software Systems

Alan McLaren – VP Business Development, MDI Solutions
Why “How to Simplify Connected Medical Device Software Integration and Certification”

Simplify

– this is more complicated than it has been in the past, and needs to get easier, more reliable, less costly

Medical device software integration

– Putting it all together on your hardware platform can take man-years
  • Graphics, security, Wi-Fi, cellular, Open*, databases, encryption, ...

Certification

– Certification bodies are getting more and more stringent with what is included in scope
  – New FDA guidance on Cybersecurity

Connected

– Improved workflows
– Improved equipment monitoring
– Opportunities for new business models
– New threats

*From IHE (Integrating the Healthcare Enterprise)*
NantHealth – connected healthcare delivery

Interconnectivity enables centralized data:

- Information processing (patient data, sensor data, machine data)
- Improved workflows and compliance
- Heuristics and predictive analytics
Why a ‘connected’ Medical device?
FDA locking down medical devices

- Content of Premarket Submissions for Management of Cybersecurity in Medical Devices
- Issued Oct. 2, 2014

- guidance provides recommendations to consider and information to include in FDA medical device premarket submissions for effective cybersecurity management

General Principles:

- Manufacturers should:
  - Develop a set of cybersecurity controls
  - Address cybersecurity during the design and development of the medical device
  - Establish design inputs related to cybersecurity
  - Provide justification for the security chosen functions
FDA locking down medical devices

Things to consider:

• Identify and protect
  – Limit access to trusted users only
    • User ID, smartcard, biometric
  – Ensure trusted content

• Detect, Respond, Recover
  – Implement device features that protect critical functionality

Implementation options:

• **TPM**: hardware solution
  – Increases BOM cost

• **Remote attestation**: networked solution

• Advances in operating systems are needed!
Security threats are real

- Medical devices are NOT carried, roaming across multiple “free” insecure networks. Hackers are.
- ‘root’ permissions give the attacker full control of the system
- **ALERT:** Feb. 22, 2014: Apple Security flaws
  - “Mac computers even more exposed”
  - "It's as bad as you could imagine, that's all I can say," said Johns Hopkins University cryptography professor Matthew Green.
  - “authorities had **100 percent success rate** in breaking into iPhones”

**Embedded devices contain corporate data,** not just:
- Contacts, e-mails
- Calendars, Facebook

**Medical devices:**
- Contain patient data
- Control/affect lives

[http://www.reuters.com/article/2014/02/22/us-apple-flaw-idUSBREA1L01Y20140222](http://www.reuters.com/article/2014/02/22/us-apple-flaw-idUSBREA1L01Y20140222)
The OS needs to increase its support for Security

• The OS needs to provide more protection against hackers in mobile networks
• Permitting a thread to elevate to ‘root’ permission to do an operation is **too coarse**
  – Processes and threads need access to system-level resources – Sure.
  – *We know* to which system resources a process or thread needs access – Yup.
    • User-input needs access to keyboard driver and interrupts
    • File I/O needs access to the filesystem
    • *Neither* of these need access to mmap() or fork() (for example)

• We have a good idea of the system-level privileges to which each process and thread needs access
The OS needs to increase its support for Security

• The OS should provide much more fine-grained control of system privilege levels
  – control settings that govern and protect which operations a process can perform, with granularity down to the system-call level
  – no longer have to give processes ‘root’ access to the entire system

• Breaks ‘root’ into multiple separate capabilities that comprise root authority
• Individual capabilities can be assigned to processes that need access to each specific resource
  – But no other resources
• Compromised processes only have a tiny subset of privileged operations available
  – Even if they become ‘root’

<table>
<thead>
<tr>
<th>Privilege Level</th>
<th>App1</th>
<th>App2</th>
<th>App3</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Resource:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mmap</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>exec</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>fork</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>ioctl</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>sockets</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>shimem</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>
Poll Question #1: Software Pedigree

What is the software pedigree of the underlying operating system for the device that you are currently developing? (or have plans to in the next 2 years)?

1. Open Source – free
2. Open source – commercial
3. Commercial OS
Challenges of integrating middleware technologies

**Connectivity, Graphics, and Middleware Components**

- ECG
- BP
- Medical application
- Infusion
- Pulse Ox
- Wi-Fi
- IAP
- Device Management
- IEEE 1588
- BTLE
- Java
- Video
- Camera
- HTML 5
- HTC 1588

**Core OS**

- Connectivity
- Filesystems
- Networking
- Utilities
- Database
- Drivers
- HA Manager
- microkernel
- scheduler
- Adaptive partitioning
- libc
- multicore

**Hardware platforms**

- TI AM335
- TI AM437
- TI AM572
- FSL i.MX6
- x86

**Compliance Documentation**

- Hazard & Risk Analysis
- Failure Analysis
- Testing records
- High level design
- Safety Case
Implementation – OS

- Safety-Critical App
- Device Housekeeping
- GUI Subsystem
- Communications
- Kernel
  OS Services
  (FS, Net, drivers)
- BSP

Core Intellectual Property

Product-specific device management

Middleware Stacks: Standard

Operating System: Commodity

Not all Created Equal
An OS is **SOUP**?

- IEC 62304:
  - a) assumes that off-the-shelf software (commercial or otherwise) will be used, and
  - b) offers two definitions of SOUP, which can be either (or both of) *software not developed for a medical device*, or software with *unavailable or inadequate records* of its development processes
- Distinction is not between COTS vs. SOUP
- More useful distinction is between *opaque* SOUP and *clear* SOUP
  - Depends on what artifacts are available to support a safety case for the software
  - These artifacts are necessary to support *your* claims of safety
SOUP: Clear? Opaque?

- For example:
  - Microsoft Windows OS is *opaque* SOUP:
    - well-documented development process
    - its vendor adheres to a development processes
    - is in possession of the source code
    - has tracked and documented the software’s failure history
    - But not available for public scrutiny
  - Open source (Apache or Linux) is *clear* SOUP
    - source code and fault histories freely available
    - software’s characteristics are well-known
    - can be scrutinized with code symbolic execution and path coverage analysis
    - the software’s long (and freely available) histories make findings from statistical analysis particularly relevant
- Clear SOUP: Software that we can *examine*
SOUP: *Clear? Opaque?*

- *Clear* SOUP: May not be the best solution for medical devices
- processes for open source development are neither clearly defined nor well documented
  - A precise concern of IEC 62304
- SOUP or COTS software may include more functionality than is needed
  - leaves dead code in the system, a practice that functional safety standards, such as IEC 61508 and IEC 62304, expressly discourage
    - Device drivers for devices that are not in the medical product
    - Support for filesystem types that are not in use
  - Removing dead code form the system can be a significant burden
    - Initial removal
    - Maintenance and patches over the product’s in-service life
Evaluating, Selecting SOUP

- Questions to ask, when evaluating COTS / SOUP Software:
  - Does the vendor make any functional safety claims about their software?
  - Has the COTS software vendor implemented a QMS?
  - Was the COTS software evaluated with Fault-Tree Analysis?
  - Does the Safety Manual state the functional safety claims for the COTS software?
    - Does the COTS software vendor provide a Safety Manual?
  - What design artifacts does the COTS software vendor provide?
  - Does the COTS software vendor use static analysis?
Poll Question #2: IEC 62304 vs. FDA Premarket Notification (510k)

You are developing a life-critical medical device. What level of safety certification are you going to seek prior to market launch?

1. FDA Pre-Market Notification
2. IEC 62304
3. Neither
4. Both
Benefits of a complete medical software ‘platform’

Connectivity, Graphics, and Middleware Components
- Wi-Fi
- Video
- Camera
- Device Management
- IEEE 1588
- BTLE
- Java

Medical application
- ECG
- BP
- Infusion
- Pulse Ox

Core OS
- Connectivity
- Filesystems
- Networking
- Utilities
- Database
- Drivers
- HA Manager
- microkernel
- scheduler
- Adaptive partitioning
- libc
- multicore

HTML 5

Graphics Subsystem

Hardware platforms
- TI AM335
- TI AM437
- TI AM572
- FSL i.MX6
- x86

Compliance Documentation
- Hazard & Risk Analysis
- Failure Analysis
- Testing records
- High level design
- Safety Case

Integration Services
- Patches/Updates
- Version alignment
- Hardware support
The Qt Framework

• Cross platform software development framework
• Implemented in C++
• API in C++ with binding to JavaScript, Python, Ruby, etc.
• Available on all desktop platforms, embedded and mobile
• Excellent documentation and learning resources

User Interface Technologies
  – Native Widgets (push buttons, combo boxes, etc.)
  – QML (JavaScript based declarative UI)
  – HTML5 (integrates WebKit)
  – OpenGL (acceleration on GL or GL/ES2)

Non-graphical, middle-ware components
  – Network, SQL, XML, printing, PDF generation, etc.
  – Easy to interface via SOAP, REST, XMLRPC
  – Regular expressions, containers, powerful string handling, etc.
Qt for medical Devices - example
HMIs in Medical Devices
Multiple UI Rendering - example

Example with 4 UI technologies

- Background wallpaper: HTML5
- Video: From iPhone or iPod
- Navigation: OpenGL ES
- Control application: Qt5

- Transparently combine multiple off-screen buffers onto the display
- Windows can be moved around, zoomed in / out, rotated, or have transparency effects applied without requiring the application to redraw or even be aware
- Hardware and virtual layer support
- Full hardware acceleration
- Multi-modal input device support
Why a ‘connected’ Medical device?
IoT means many connected applications
Why a ‘connected’ Medical device?
Orthogonal planes of device management. Equal planes of opportunity.

Physical device itself: Remote Management and Diagnostics

- Memory errors, alarms
- CPU utilization alarms
- Firmware updates
- Hardware failures
- Security breaches
- Low supply alarms (battery, drug supply, ) [AED !]

Revenue-generating use cases

- Imaging
  - Learn from the Storage Industry

- Infusion pumps
  - Reduce manual workload while maintaining drug library compliance

Technical considerations

- Connectivity (Wi-Fi, cellular, Bluetooth)
- Security
Why a ‘connected’ Medical device?
Orthogonal planes of device management. Equal planes of opportunity.

Sensor Data: IoT and sensor layer
• Typical m2m use case
• Application level data being transmitted to back-end analytics engine

Revenue-generating use cases
• Low-cost hand-held imaging device
  – Rural and remote locations
  – 100x patients, 10x nurses, 1x Doctor
  – Brings the Doctor to the remote towns
  – Scan for chronic diseases
  – Device is distributed for free
  – Imaging analysis pay-per-use model
• RPM

Technical considerations
• Location of imaging data storage
  – In-country, within Enterprise
• Security, patient data
Remote patient monitoring, BANs

• Data wirelessly transmitted to receiver
  – Smartphone with BTLE
• Devices connected to patients, transmitting patient functions
  • ECG, EEG
  • Blood pressure
  • Stress, sweat, temperature
  • Patient movement
• Monitoring app transmits/receives data to/from Clinician’s IT systems
• Alerts/alarms can be received by Clinician as they occur

Benefits:
• Send patients home sooner, while still being continuously monitored for alert situations
Problems can arise ....

- **AlivCor** – ECG on iPhone 5
- **AirStrip** Patient Monitoring

---

**PRESS RELEASE**
May 8, 2013, 8:00 a.m. EDT

**AliveCor Expands Mobile ECG Device Offering to Include iPhone 5**

Heart Monitor Product Line on Display at Heart Rhythm 2013

SAN FRANCISCO, May 08, 2013 (BUSINESS WIRE) -- AliveCor, Inc., a premier digital health company, announced today that its Heart Monitor is now available for the iPhone 5, joining the current models for the iPhone 4 and 4S. The AliveCor Heart Monitor is the first FDA-cleared mobile device-based ECG monitor. The product line will be on display at Heart Rhythm 2013, the Heart Rhythm Society's (HRS) 34th Annual Scientific Sessions, in Denver, Colorado. AliveCor will also be providing Heart Monitors for HRS's Citywide Awareness Campaign for atrial fibrillation (AFib), promoting "Cardiac Rhythm Awareness Month" in Colorado.

---

Problems can arise....

Signal Processing Algorithms:
- Sample Windows
  - Filtering, spatial, frequency domain processing
  - Energy calculation
- Specific sample frequency must be maintained
VI. CONCLUSIONS

The RPM system providing real-time alerts/notifications has hard real-time performance requirements. Without guaranteed real-time performance, the RPM system on mobile devices would not be able to deliver real-time monitoring and can only be viable for monitoring non-real-time critical signals. Our results indicate that the real-time performance of the RPM system may be significantly affected if multiple tasks are scheduled concurrently.

One of the benchmarks used to evaluate system computing is the response time. A low response time is crucial for real-time applications like the RPM system, where quick response is needed to prevent errors or complications. The RPM system needs to be able to provide timely notifications and alerts to healthcare providers, and a low response time is essential to ensure this.

In conclusion, the RPM system is a valuable tool for remote patient monitoring. However, the real-time performance requirements need to be carefully considered to ensure the system is effective and reliable.
Artificial Pancreas

• Fully automated multi-hormone closed-loop system
• Existing Insulin pumps are single-hormone; do not dispense glucagon
• Clinical trials run through Summer 2014
  – Children ages 6-11 with Type 1 diabetes
  – Measurable improvement in glycemic control and quality of life
• The trial system based on an iPhone 4S with an attached glucose monitoring device, two pumps, and reservoirs for **insulin** and **glucagon**

Using a Real-Time Operating System for Multitasking in Remote Patient Monitoring

Michael Rockwood\textsuperscript{1}, Vilas Joshi\textsuperscript{1}, Kevin Sullivan\textsuperscript{1}, Rafik Goubran\textsuperscript{1} Fellow IEEE
\textsuperscript{1}Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada
\{mrockwood, vyoshi, goubran\}@ece.carleton.ca, kevinsullivan@memail.carleton.ca

V. CONCLUSION

Real-time remote monitoring is becoming increasingly popular as mobile processing power becomes cheaper and available more ubiquitously. This will allow the use of multiple sensors to monitor a variety of chronic and acute conditions simultaneously. Using high sampling rate sensors like video, audio, or multi-channel ECG/EEG along with complex signal processing algorithms will significantly increase the processor workload, and may lead to starvation of real-time critical tasks. An RPM system based on an RTOS can prioritize the time-critical tasks to ensure that alarms are generated in a timely manner.
Other Medical Device Connectivity Options, Benefits, and challenges: HL7 interfaces to EMR Systems

www.mdisolutions.com
Alan McLaren
amclaren@mdisolutions.com
416 720 7577
MDI Solutions – challenges with device integration

Multiple dependencies

- what data needs to be collected?
- where does it go?
- who looks at it?
- when do they need to look at it?
- how often?
- how is the data stored?
- how long is data stored?

Flexibility creativity to address work flow needs

Clinical work flow is unique to each environment
Device to EMR

Device exchanges with Server
Server exchanges with EMR

Often busy HL7 interfaces:

a single monitor transmitting 5 vital signs at 15 minute intervals sends 480 vital signs in a 24 hour period
200 hundred monitors transmitting at this rate, send 96,000 vital signs per day.

HL7 Messaging
MDI Link installed on QNX Neutrino – device integration
Device to EMR
MD LINK DEVELOPMENT CYCLE

Interfaces are the customized mapping of data, application and network resources. MD Link’s growing list of Interface adapters provide visual customization and data mapping.

Processes link Event, Task and Resource Interfaces to create active solution Services.

Services are processes that can be hosted on any server. Services monitor Interface status and Transactions activity.

Transactions are Service requests that can generate documents, messages and database updates.
MD LINK DEVELOPMENT CYCLE

Transactions are service requests that can generate documents, messages and database updates (See Server console)
MD LINK Single Screen Design Studio
Developer Challenges

• Evolve continually with different vendors that you interface with

• Flexibility to easily configure for each of their clients without having to always hardcode changes
MD LINK addressing developer challenges

• Example standard data set that they transmit from their devices, use MDLink in the middle to create the HL7 to get Patient Demographics from EMR or Update EMR with device results

• Transmit in other formats for oddball EMRs, some don't use HL7

Presents options

1. For those that have engines, have a "standard" HL7 configuration that is a fairly low cost

2. For those that don't have engines, a "services" approach supporting detailed changes to meet all of the requirements of their EMR
Poll Question #3: Connectivity

Are you currently developing a medical device that will be connected to a back-end management system (or have plans to in the next 2 years)? If so, what is the use case?

1. Yes. Remote Management and Diagnostics
2. Yes, IoT and Sensor data acquisition
3. Yes, both RM&D and IoT
4. No current plans
Summary
Questions?

Chris Ault – Product Manager, QNX Software Systems

Alan McLaren – VP Business Development, MDI Solutions