Towards Assurance of a Patient-Specific Network of Medical Devices.

SCC 2015, Rockville Maryland

Sam Procter, John Hatcliff, and Robby SAnToS Lab
Kansas State University

Support:
This work is supported in part by the US National Science Foundation (NSF) (#1239543), the NSF US Food and Drug Administration Scholar-in-Residence Program (#1355778) and the National Institutes of Health / NIBIB Quantum Program.
Health Care Involves A Variety of System Components

Information Systems

Sensors

Actuators

Sensor Data Displays

Clinical Protocols

Clinicians

Patient

Displays
Motivation

- What are the types of things we could do with device integration?
  - Information forwarding
  - Automation of clinical workflows
  - Closed loop control between devices
- Unlike personal computing, medical devices are not designed to work together
- Integrating medical devices would bring myriad benefits
- ... how can we do so safely?
Outline

- Background
  - PCA Interlock Scenario
  - Medical Application Platforms
  - Tooling
- Status Quo
- STPA + AADL
- Impacts / Future
PCA Interlock Scenario

- Patients are commonly given patient-controlled analgesics after surgery.
- Crucial to care, but numerous issues related to safety.
- Data for disabling the pump exists now (just a system invariant) -- we just need to integrate it.
PCA Pump Safety Interlock

Fully leverage device data streams and the ability to control devices

**Devices**

- **PCA Pump**
  - Enable Pump for safe time window

- **Capnograph**
  - Monitoring Data + Alarm Information

- **Pulse Oximeter**
  - Monitoring Data + Alarm Information

**Device Task controller**

- **Enable bolus dose only when ticket present**
- **PCA Bolus “Enable” Ticket**

**Combined PCA Vitals Monitoring**

- **Aggregated Monitoring Status**

- **Clinician / Monitoring**
  - Status Display for PCA Monitoring Application
A Medical Application Platform is a safety- and security-critical real-time computing platform for...

- Integrating heterogeneous devices, medical IT systems, and information displays via communications infrastructure, and
- Hosting applications ("apps") that provide medical utility via the ability to acquire information from and update/control integrated devices, IT systems, and displays
We use medicine in our examples
  ... but this can extend to other compositional systems

Core idea:
  Integration of heterogeneous
    Sensors,
    Actuators, and
    Complete systems,
  by small chunks of software,
  in a verifiable manner
Background

PCA Pump Interlock Architecture

Medical Application Platform

- PCA
- PR
- SPO2
- ETCO2
- RR

Start / Stop Commands

Sensor + Alarm Data

Data should arrive once per second

View Display

Configuration, Alarm Clear

Attach Sensors

View Display

Pulse Oximeter, Capnograph, and Patient Controlled Analgesia Pump

Clinician (App Administrator)
Tooling Vision

Analyses and Regulatory Artifacts

Clinical Use Case / Workflow Description
Requirements
Hazard Analysis
Risk Assessment

App Developer

Medical Device Coordination Framework

Assurance Case
3rd Party ICE Conformance & Safety Certification Submission Package

FDA 510K Submission Package

3rd Party Certifiers
FDA Evaluators
A. The app’s architecture is specified in a suitable formalism
   1. Components as AADL Devices / Processes
   2. Connections are specified
   3. RT/QoS Parameters are via AADL’s property-specification mechanism

B. The app is programmatically translated to Java and XML
   1. Only “Business Logic” is written by the developer

C. The app is launched on a compatible MAP
Component Development

- Development of component architecture using architecture formalism
- Automatic generation of component architecture (skeletons)
- Automatic generation of component layout and app topology (configuration)
- Development of core behavioral code (business logic) using IDE of choice
- Translator can be retargeted to other languages as desired
Outline

- Background
- Status Quo
  - Existing Hazard Analyses
  - Application to MAP domain
- STPA + AADL
- Impacts / Future
Hazard Analysis

History: FTA

- FTA: Bell Labs, 1962
  - Looks for contributory causes to undesired events

- Too Large of Dose Allowed
  - G1
    - Bad Physiological Data Received
    - G2
      - Incorrect Physiological Reading
      - Software Encoding or Decoding Error
    - G3
      - Message Garbled by Network
      - Physiological Data within Max Range
  - Undetected Error
    - Internal Diagnostics Fail
## Hazard Analysis

### History: FMEA

- **FMEA: US Military, 1949**
  - Analyses impacts of individual components

<table>
<thead>
<tr>
<th>Function</th>
<th>Failure Mode</th>
<th>Fail Rate</th>
<th>Causal Factors</th>
<th>Effect</th>
<th>System Effect</th>
<th>Detected by</th>
<th>Current Control</th>
<th>Hazard</th>
<th>Risk</th>
<th>Rec. Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provide SpO(_2)</td>
<td>Fails to Provide</td>
<td>N/A</td>
<td>Network or dev. Failure</td>
<td>No SpO(_2) data</td>
<td>Unknown patient state</td>
<td>App</td>
<td></td>
<td>Potential OD</td>
<td>3D</td>
<td>Default to KVO</td>
</tr>
<tr>
<td>Provides late</td>
<td>N/A</td>
<td>Network slowness</td>
<td>No SpO(_2) data</td>
<td>Unknown patient state</td>
<td>App</td>
<td></td>
<td>Potential OD</td>
<td>3C</td>
<td>Default to KVO</td>
<td></td>
</tr>
<tr>
<td>Provides wrong</td>
<td>N/A</td>
<td>Device error</td>
<td>SpO(_2) wrong</td>
<td>Wrong patient state</td>
<td>None</td>
<td></td>
<td>Potential OD</td>
<td>1E</td>
<td>Dev. should report data quality</td>
<td></td>
</tr>
</tbody>
</table>

Analyst: Sam Procter

Date: September 26, 2014
Unique aspects of MAP domain

- Software based
  - Hardware is interchangeable
- Component oriented
  - Compositional system needs compositional safety
- Unclear how FTA / FMEA might apply
- Early, firm notion of system architecture
Formalized Notion of Architecture

Formal architecture descriptions become the scaffolding on which:
- Requirements,
- Development,
- Risk management,
- Deployment, and
- Ecosphere coordination is organized.
Outline

- Background
- Status Quo
- STPA + AADL
  - STPA
  - AADL
  - Tool-based Integration: MDCF Architect
- Impacts / Future
Hazard Analysis

Leveraging Semiformal Architectural Descriptions

Clinical Use Case / Workflow Description

Requirements

App Developer

Assurance Case

3rd Party ICE Conformance & Safety Certification Submission Package

MDCF

FDA 510K Submission Package

App Deployment

3rd Party Certifiers

FDA Evaluators

Hazard Analysis

Risk Assessment
Hazard Analysis

History: STPA

- STPA: Nancy Leveson / MIT, 2005(ish)
- Applies systems theory, focuses on control...
  - Loops
  - Actions
Hazard Analysis

History: STPA

- STPA: Nancy Leveson / MIT, 2005(ish)
- Applies “Systems” theory, focuses on control...
  - Loops
  - Actions

Controlled Process: Patient

Sensor: Pulse Oximeter
- Inadequate Operation: SpO$_2$ value incorrect

Actuator: PCA Pump
- Inadequate Operation: Pumps Normally

Controller: App Logic
- Process Model Incorrect: Wrongly believes patient to be healthy

Feedback: PulseOx → App
- Inadequate Feedback: Sends bad SpO$_2$

Control Action: App → PCA Pump
- Inappropriate Control Action: Inadvertent “Pump Normally” command
Hazard Analysis

Can STPA be improved?

- STPA enables reasoning about
  - Hardware,
  - Software, and
  - Socio-technical elements
- And is driven by architecture ("Boundary Crossing")

- No open tooling
  - Tooling isn’t bound to architecture
- Existing work is largely manual
Why use AADL?

- **Architecture** Analysis and Design Language
- History of successful safety-critical projects
  - Avionics / Boeing (SAVI): “integrate-then-build” approach
- Annexes support a number of regulatory and verification artifacts
  - Hazard Analysis (EM) extends notion of interface to include faults
Supply rate: 1 sec .. 5 sec

Channel Delay: 50ms

Period: 50ms
WCET: 5ms
Language

System

Medical Devices

Software Components

Communication links between components

... and properties of those links!
Language

System

package PCA_Shutoff
public
with PulseOx_Interface, PCAPump_Interface, PCA_Shutoff_Logic,
    PCA_Shutoff_Properties, MAP_Error_Properties, PCA_Shutoff_Display,
    PCA_Shutoff_Errors, Capnograph_Interface, MAP_Errors,
    PCA_Shutoff_Error_Properties;

system PCA_Shutoff_System
end PCA_Shutoff_System;

system implementation PCA_Shutoff_System.imp
subcomponents
    -- Physiological inputs
capnograph : device Capnograph_Interface::ICEcapnographInterface.imp;

    -- App Logic
appLogic : process PCA_Shutoff_Logic::ICEpcaShutoffProcess.imp;
appDisplay : process PCA_Shutoff_Display::ICEpcaDisplayProcess.imp;

    -- Controlled device
pcaPump : device PCAPump_Interface::ICEpcaInterface.imp;
connections
    -- From components to logic
respiratoryrate_logic : port capnograph.RespiratoryRate -> appLogic.RespiratoryRate;
pumpcommand_logic : port appLogic.CommandPumpNormal -> pcaPump.PumpNormally;
etco2_logic : port capnograph.ETCO2 -> appLogic.ETCO2
{MAP_Properties::Channel_Delay => 50 ms};

    -- From components to display
pumpcommand_disp : port appLogic.CommandPumpNormal -> appDisplay;
end PCA_Shutoff_System.imp;
end PCA_Shutoff;

Medical Devices

Software Components

Communication links between Components

... and properties of those links!
STPA: Fundamentals

STPA: Background & Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - Control Actions
  - Control Structure
Hazard Analysis

STPA: Fundamentals

- Fundamentals
- Accident Levels
- Accidents
- System Boundaries
- Hazards
- Safety Constraints
- Control Actions
- Control Structure

Example

1. A human is killed or seriously injured.
2. A medical device’s services are unavailable

Tie into ISO 14971’s notions of criticality?
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - Control Actions
  - Control Structure

**Example**

1. The patient is killed or seriously injured [DeathOrInjury]

2. The PCA pump stops responding to commands [DenialOfService]

```plaintext
PatientHarmed : constant MAP_Error_Properties::Accident => [
  Number => 1;
  Description => "Patient is killed or seriously injured."
  Level => PulseOx_Forwarding_Error_Properties::DeathOrInjury;
];
```
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - Control Actions
  - Control Structure
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - Control Actions
  - Control Structure

Example

1. An inadvertent “Pump Normally” command is sent to the pump [PatientHarmed]

2. Commands are sent to the pump too quickly [PCADoS]

```plaintext
InadvertentPumpNormally : constant MAP_Error_Properties::Hazard => {
  Number => 1;
  Description => "An inadvertent `Pump Normally` command is sent to the pump."
  Accident => PulseOx_Forwarding_Error_Properties::PatientHarmed;
};
```
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - **Safety Constraints**
  - Control Actions
  - Control Structure

---

**Example**

1. The app must only instruct the pump to run at a normal rate when the patient can tolerate more analgesic [InadvertentPumpNormally]

2. The app must wait for a designated length of time between sending pump commands [TooManyCommands]

```cpp
PumpWhenSafe : constant MAP_Error_Properties::Constraint => [Number => 1;
  Description => "The app must only instruct the pump to run at a normal rate when the patient can tolerate more analgesic.";]
Hazard =>
PulseOx_Forwarding_Error_Properties::InadvertentPumpNormally;
```
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - **Control Actions**
  - Control Structure

**Example**

1. App -> Pump: Pump Normally
2. PulseOx -> App: SpO₂ = 95
3. App -> Display: Patient = Ok
Hazard Analysis

STPA: Fundamentals

- Fundamentals
  - Accident Levels
  - Accidents
  - System Boundaries
  - Hazards
  - Safety Constraints
  - Control Actions
  - Control Structure

Example

- Pulse Oximeter
  - Device Ok
  - Device Error

- Capnography Device

- App
  - Pump Normal
  - Pump KVO

- PCA Pump
  - Patient

- Clinician
  - Request More
  - Pump Status
  - Verify Rx
  - Authorize Override
  - Provide Rx

- Display
  - View Patient Status
  - View Device Status
Hazard Analysis

STPA: Identifying Hazardous Control Actions

- Hazardous Control Action Table
  - Cross-product of control actions and STPA guidewords

<table>
<thead>
<tr>
<th>Control Action</th>
<th>Providing</th>
<th>Not Providing</th>
<th>Applied too Long</th>
<th>Stopped too Soon</th>
<th>Early</th>
<th>Late</th>
</tr>
</thead>
<tbody>
<tr>
<td>App -&gt; Pump: Pump Normally</td>
<td>PH</td>
<td>Not Hazardous</td>
<td>PH</td>
<td>Not Hazardous</td>
<td>PH</td>
<td>Not Hazardous</td>
</tr>
<tr>
<td>App -&gt; Disp: Patient Ok</td>
<td>BID</td>
<td>BID</td>
<td>BID</td>
<td>BID</td>
<td>BID</td>
<td>BID</td>
</tr>
<tr>
<td>PulseOx-&gt;App: Provide SpO₂</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
</tr>
<tr>
<td>PulseOx-&gt;App: Provide Pulse Rate</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
<td>Not Hazardous</td>
<td>PH, BID</td>
</tr>
</tbody>
</table>

*PH = Patient Harmed
*BID = Bad Info Displayed*
Hazard Analysis

STPA: Hazardous Causes and Compensations

Control Action: App -> Pump: Pump Normally

- Providing:
  - Bad Data:
    - Cause:
      - Incorrect values are gathered from one of the physiological sensors
    - Compensation:
      - Rely on multiple sensed physiological parameters to provide redundancy

- Not Providing:
  - Not hazardous
Hazard Analysis

STPA: Hazardous Causes and Compensations

Control Action: App -> Pump: Pump Normally

- Wrong Timing or Order:
  - Not applicable

- Too Long
  - Network Drop
    - Cause:
      - Network drops out, leaving the pump running normally regardless of the patient’s health
    - Compensation:
      - Commands to pump normally have an associated maximum time, after which the pump returns to KVO
STPA Control Loop
Including Causality Guidewords

“Engineering a Safer World” Leveson, 2011
Hazard Analysis

Returning to our Architectural Model
A control action is provided in an unsafe way

- How would the control action be unsafe?
- What hazard would be caused?
- What constraint would be violated?
- What should the occurrence be named?
- What would cause this to occur?
- How can this occurrence be compensated for?
Hazard Analysis

Annotating our Architectural Model

How would the control action be unsafe?
What hazard would be caused?
What constraint would be violated?
What should the occurrence be named?
What would cause this to occur?
How can this occurrence be compensated for?

We’ll come back to these two in a moment.
AADL Component Architecture with Hazard Annotations

- Development of component architecture using AADL / OSATE2
- Addition of Hazard Analysis Annotations
- Automatic generation of STPA-Styled Hazard Analysis Report

Example “In Progress” Report Online at:
http://santoslab.org/pub/mdcf-architect/HazardAnalysis.html
Annotating our Architectural Model
Inside the AADL System Component

What specific fault will result?
What control action will be affected?
What can we do with our model + specific fault information?
## Fault Types

### EMV2 Type Hierarchy

<table>
<thead>
<tr>
<th>Error Library Type</th>
<th>STPA Error Type</th>
<th>App Error Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Errors with Physiological Monitors</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LateDelivery</td>
<td>DelayedOperation</td>
<td>SpO2ValueLate</td>
</tr>
<tr>
<td>IncorrectValue</td>
<td>IncorrectInformation</td>
<td>SpO2ValueLow</td>
</tr>
<tr>
<td>N/A</td>
<td>NoInformation</td>
<td>NoSpO2Data</td>
</tr>
<tr>
<td><strong>Errors with App Logic</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServiceCommission</td>
<td>InnappropriateCtrlAction</td>
<td>InadvertentPumpNormally</td>
</tr>
<tr>
<td>ServiceOmission</td>
<td>MissingCtrlAction</td>
<td>InadvertentPumpMinimally</td>
</tr>
</tbody>
</table>

**AADL Standard Error Types**  
**STPA Error Types**  
**App Specific Error Types**
Fault Types

App Specific Error Library

package PCA_Shutoff_Errors
public
with MAP_Errors, PCA_Shutoff_Error_Properties, MAP_Error_Properties,
   PCA_Shutoff;

annex EMV2
{**
   error types
   -- These errors aren't associated with unsafe states, but they're here for completeness
   SpO2ValueLow : type extends MAP_Errors::WrongPhysioDataError;
   RespiratoryRateLow : type extends MAP_Errors::WrongPhysioDataError;
   ETCO2ValueHigh : type extends MAP_Errors::WrongPhysioDataError;

   -- These errors will cause the app to logic to think the patient is healthy when she isn't
   SpO2ValueHigh : type extends MAP_Errors::WrongPhysioDataError;
   RespiratoryRateHigh : type extends MAP_Errors::WrongPhysioDataError;
   ETCO2ValueLow : type extends MAP_Errors::WrongPhysioDataError;

   -- These are errors with devices
   DeviceAlarmFailsOn : type extends MAP_Errors::PhysioDeviceErrorCommission;
   DeviceAlarmFailsOff : type extends MAP_Errors::PhysioDeviceErrorOmission;
   BadInfoDisplayedToClinician : type extends MAP_Errors::WrongInfoDisplayedError;
   InadvertentPumpNormally : type extends MAP_Errors::AppCommission;
   InadvertentPumpMinimally : type extends MAP_Errors::AppOmission;
end types;
**};
end PCA_Shutoff_Errors;
Hazard Analysis

Annotating the Architectural Model

The fault is traced to its source component / port
Hazard Analysis

Specification Step 1: Propagation

package PulseOx_Interface
public
with PCA_Shutoff_Types, PCA_Shutoff_Errors, EMV2, MAP, Error_Properties, PCA_Shutoff;
device ICEpoInterface
features
    SpO2 : out event data port PCA_Shutoff_Types::SpO2;
annex EMV2 {**
    use types PCA_Shutoff_Errors;
    error propagations
        SpO2 : out propagation {SpO2ValueHigh};
        flows
        SpO2UndetectableHighValueFlowSource : error source SpO2 {SpO2ValueHigh};
    end propagations;
    **};
end ICEpoInterface;
device implementation ICEpoInterface.imp
end ICEpoInterface.imp
end PulseOx_Interface;
Hazard Analysis

Specification Step 2: Flow

package PulseOx_Interface
public
with PCA_Shutoff_Types, PCA_Shutoff_Errors, EMV2, MAP_Errors, SpO2, SpO2ValueHigh, PCA_Shutoff;
device ICEpoInterface
features
    SpO2 : out event data port PCA_Shutoff_Types::SpO2;
annex EMV2 {**
    use types PCA_Shutoff_Errors;
    error propagations
        SpO2 : out propagation {SpO2ValueHigh};
    flows
        SpO2UndetectableHighValueFlowSource : error source SpO2 {SpO2ValueHigh};
    end propagations;
**};
end ICEpoInterface;

device implementation ICEpoInterface.imp
end ICEpoInterface.imp;
end PulseOx_Interface;
Hazard Analysis

Error transformation

The fault arrives at a component

Where it is transformed into a new fault
Hazard Analysis

Specification Step 3: Error Transformations

```plaintext
package PCA_Shutoff;
public
with PCA_Shutoff_Types, PCA_Shutoff_Properties, MAP_Properties;

process ICEpcaShutoffProcess
features
  SpO2 : in event data port PCA_Shutoff_Types;
  CommandPumpNormal : out event data port PCA_Shutoff_Types::PumpNormalCommand;
properties
  MAP_Properties::Component_Type => logic;
annex EMV2 {
  use types PCA_Shutoff_Errors;
  error propagations
    SpO2 : in propagation {SpO2ValueHigh};
    CommandPumpNormal : out propagation {InadvertentPumpNormally};
  flows
    HighSpO2LeadsToPO2 : error path SpO2{SpO2ValueHigh} -> CommandPumpNormal{InadvertentPumpNormally};
  }
end propagations;
**};
end ICEpcaShutoffProcess;

-- Process implementation redacted
end PCA_ShutoffLogic;
```
Hazard Analysis

Specification Step 3: Error Transformations

```plaintext
package PCA_Shutoff Logic
public
with PCA_Shutoff_Types, PCA_Shutoff_Errors, MAP_Properties;

process ICEpcaShutoffProcess
features
  SpO2 : in event data port PCA_Shutoff_Types::SpO2;
  CommandPumpNormal : out event data port PCA_Shutoff_Types::PumpNormal;
properties
  MAP_Properties::ComponentType => annex EMV2 {**
    use types PCA_Shutoff_Errors;
    error propagations
      SpO2 : in propagation {SpO2ValueHigh};
      CommandPumpNormal : out propagation {InadvertentPumpNormal};
    flows
      HighSpO2LeadsToOD : error path SpO2{SpO2ValueHigh} -> CommandPumpNormal{InadvertentPumpNormal};
      **};
  end propagations;
end ICEpcaShutoffProcess;

-- Process implementation redacted
end PCA_Shutoff_Logic;
```
Hazard Analysis

Error Sinks

The new fault is propagated to a port / component

Where it stops.
Hazard Analysis

Specification Step 4: Error Sink

```plaintext
package PCAPump_Interface;
public
with PCA_Shutoff_Types;
    device ICEpcaInterface
        features
            PumpNormally : in event data port PCA_Shutoff_Types::PumpNormalCommand;
        annex EMV2 {**
            use types PCA_Shutoff_Errors;
            error propagations
                PumpNormally : in propagation {InadvertentPumpNormally};
            flows
                ODCmd : error sink PumpNormally {InadvertentPumpNormally};
            end propagations;
        **};
    end ICEpcaInterface;

device implementation ICEpcaInterface.imp
end ICEpcaInterface.imp;
end PCAPump_Interface;
```
### Hazard Analysis

#### Specification Step 4: Error Sink

```
package PCAPump_Interface
public
with PCA_Shutoff_Types;
    device ICEpcaInterface
features
        PumpNormally : in event data port PCA_Shutoff_Types::PumpNormally;
    annex EMV2 {**
        use types PCA_Shutoff_Errors;
        error propagations
            PumpNormally : in propagation {InadvertentPumpNormally};
        flows
            ODCcommand : error sink PumpNormally {InadvertentPumpNormally};
        end propagations;
    **};
end ICEpcaInterface;

device implementation ICEpcaInterface.imp
end ICEpcaInterface.imp;
end PCAPump_Interface;
```
Hazard Analysis

OSATE Remembers a Neglected Connection

Anything missing?

There are two missed error propagations!
Hazard Analysis

OSATE Remembers A Neglected Connection

```
system implementation PCA_Shutoff_System.imp
subcomponents
  -- Physiological inputs
  pulseOx : device PulseOx_Interface::ICEpoInterface.imp;

  -- App logic
  appLogic : process PCA_Shutoff_Logic::ICEpcaShutoffProcess.imp;
  appDisplay : process PCA_Shutoff_Display::ICEpcaDisplayProcess.imp;
connections
  -- From components to logic
  spo2_logic : port pulseOx.SpO2 -> appLogic.SpO2;

  -- From components to display
  spo2_disp : port pulseOx.SpO2 -> appDisplay.SpO2;

properties
  -- Errors between the PulseOx's SpO2 channel and the App Logic
  MAP_Error_Properties::Occurrence => [
    Kind => ValueHigh;
    Hazard => PCA_Shutoff_Error_Properties::PatientHarmed;
    ViolatedConstraint => PCA_Shutoff_Error_Properties::PumpWhenSafe;
    Title => "Wrong Values (Undetected)";
    Cause => "Incorrect values are gathered from the physiological sensors";
```
Tool

OSATE2

- Open-source, Eclipse-based tool
- Our work is available as a plugin
  - Uses the model-traversal built into OSATE2

Open Source AADL Tool Environment
http://www.aadl.info
http://www.aadl.info/wiki
Hazard Analysis

Our model is updated accordingly
Hazard Analysis

Interaction between Report and Model

1. Here’s an empty cell (STPA Keyword + Control Action)... could anything go wrong?
2. Create occurrence and supporting EMV2 annotations
3. Where else could this fault go?
4. What else could cause this error?

Bottom Up

Top Down
Outline

- Background
- Status Quo
- STPA + AADL
- Impacts / Future
Contributions (1 of 2)

- Showing how STPA methods / artifacts can be integrated with a formal architecture modeling framework
- Demonstrating how AADL EM annotations can aid in supporting STPA
- Demonstrating a methodology for carrying out STPA in AADL-defined architectures
Contributions (2 of 2)

- Tool support to automate parts of the methodology
  - And to aid both analysts and reviewers in analysis and review of the generated STPA artifacts.

- Establishing the basis for very strong traceability between
  - Requirements,
  - Architecture,
  - Hazard analysis,
  - Testing / Verification, and
  - Executable code
Further Reading

- Source available online at https://github.com/santoslab/aadl-translator
- Installable into OSATE2 via update site: http://santoslab.org/pub/mdcf-architect/updatesite
- Full documentation online at http://santoslab.org/pub/mdcf-architect
- Publications online at http://people.cis.ksu.edu/~samprocter
Towards Assurance of a Patient-Specific Network of Medical Devices.

SCC 2015, Rockville Maryland

Sam Procter, John Hatcliff, and Robby SAnToS Lab
Kansas State University

Support:
This work is supported in part by the US National Science Foundation (NSF) (#1239543), the NSF US Food and Drug Administration Scholar-in-Residence Program (#1355778) and the National Institutes of Health / NIBIB Quantum Program.