Applying STPA-based Hazard Analysis to support HBSE for Systems built using MAPs

ISPCE 2015 – Chicago, IL, USA

Sam Procter, John Hatcliff, Kim Fowler
SAnToS Lab
Kansas State University

Anura Fernando
Underwriters Laboratories

Sandy Weininger
Food and Drug Admin.

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

Clinical Protocols
Sensors
Actuators
Sensor Data Displays
Displays
Clinicians
Information Systems

Patient!
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
  - AADL
- Vision
- Language
- Tool
- Hazard Analysis
- Future
Status Quo: MDDS

Medical Device Data Systems – Data only flows from producers to consumers; data must be faithfully re-presented
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
Clinically Supported

Motivating Clinical Problem: PCA Overdose

“A particularly attractive feature may be the ability to automatically terminate or reduce PCA (or PCEA) infusions when monitoring technology suggests the presence of opioid-induced respiratory depression. To facilitate such capabilities, we strongly endorse the efforts to develop international standards for device interoperability and device-device communication.

It is critical that any monitoring system be linked to a reliable process to summon a competent health care professional to the patient's bedside in a timely manner. “
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

**Combined PCA Vitals Monitoring**

- **Device Task controller**
  - Enable bolus dose only when ticket present

- **PCA Bolus “Enable” Ticket**

- **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
Background

PCA Pump Interlock Architecture

Medical Application Platform

App

SUI App Display

View Display

Data for Display

Data should arrive once per second

Start / Stop Commands

Sensor + Alarm Data

Pulse Oximeter, Capnograph, and Patient Controlled Analgesia Pump

PCA, PR, SPO₂, ETCO₂, RR

Patient

Clinician (App Administrator)

Configuration, Alarm Clear

Attach Sensors

View Display
Background

Architecture Analysis and Description Language (AADL)

- SAE Standard, used in e.g., Avionics
- Enables model-driven, component-based development of
  - Software
  - Hardware
  - And the bindings between the two
- Previously applied to a single medical device, what about a system of multiple medical devices?
- How well can it work on a managed platform?
- Can we do anything beyond describing an app’s architecture with it?
Outline

- Background
- Vision
  - Analyses
  - Code generation
- Language
- Tool
- STPA
- Hazard Analysis
- Future
Vision

Analyses and Regulatory Artifacts

Medical Device Coordination Framework

App Developer

Clinical Use Case / Workflow Description

Requirements

Hazard Analysis

Risk Assessment

3rd Party Certifiers

FDA Evaluators

Assurance Case

3rd Party ICE Conformance & Safety Certification Submission Package

FDA 510K Submission Package

App Deployment

FDA 510K Submission Package
Vision

Code Generation

A. The app’s architecture is specified in AADL
   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

C. The app is launched on a compatible MAP
Outline

- Background
- Vision
- Language
  - Why MDD?
  - Why (a subset of) AADL?
  - Constructs
- Tool
- Hazard Analysis
- Future
MAP Characteristics

*MAP constituted device* instances are variable – the constituents that form the MAP constituted device may different on different invocations of the device.

- Same app, and thus same conceptual “system”
- Just one architecture and development framework
- But, different component instances.
**MAP Characteristics**

*MAP constituted device* instances are variable – the constituents that form the MAP constituted device may different on different invocations of the device.

- **Same app, and thus same conceptual “system”**
- **Just one architecture and development framework**
- **But, different component instances.**
Why use AADL?

- History of successful safety-critical projects
  - Avionics / Boeing (SAVI): “integrate-then-build” approach
- Previously found that MAPs lend themselves to pub-sub
  - Device as publisher, apps as subscriber
  - Natural to model with AADL’s port connections
- Annexes support a number of regulatory and verification artifacts
  - Hazard Analysis (EMV2), Interface contracts (BLESS), etc.
AADL is targeted at co-design, i.e., complete systems
  - MAPs are managed platforms

- Semantic mismatches
  - Processes
- Insufficiency of pre-declared properties
- Unrealizable communication patterns
  - No shared-memory access in pub/sub middleware
AADL System

AADL Process: Logic
- Thread1
- Thread3

AADL Process: Display
- Thread1
- Thread2

Device1

Device2

Output 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

```java
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;
```
Language

Device Interface Specification

Device API Only -- Presents the app’s view of the required device capabilities, not the full device capabilities
Language

Device Interface Specification

Device API Only --

Presents the app’s view of the required device capabilities, not the full device capabilities

```java
package Capnograph_Interface
public
with PCA_Shutoff_Types;
    device ICEcapnographInterface
    features
        ETCO2 : out event data port PCA_Shutoff_Types::ETCO2;
        RespiratoryRate : out event data port PCA_Shutoff_Types::RespiratoryRate;
    end ICEcapnographInterface;
end ICEcapnographInterface;

device implementation ICEcapnographInterface.imp
end ICEcapnographInterface.imp;
end Capnograph_Interface;
```

```java
package PCAPump_Interface
public
with PCA_Shutoff_Types;
    device ICEpcaInterface
    features
        PumpNormally : in event data port PCA_Shutoff_Types::PumpNormalCommand
            {MAP_PROPERTIES::Output_Rate => 50 ms .. 75 ms};
    end ICEpcaInterface;
end ICEpcaInterface;

device implementation ICEpcaInterface.imp
end ICEpcaInterface.imp;
end PCAPump_Interface;
```
Language

Process Specification

External ports

Tasks (Threads)

Connections between external ports and threads
Language

Process Specification

```plaintext
import PCA_ShutoffLogic, PCA_ShutoffProperties, MAPProperties;

process ICEpcaShutoffProcess
features
  - No ETCO2 thread / connection because it's a data port
  ETCO2 : in data port PCA_ShutoffTypes::ETCO2;
  RespiratoryRate : in event data port PCA_ShutoffTypes::RespiratoryRate;
  CommandPumpNormal : out event data port PCA_ShutoffTypes::PumpNormalCommand;
properties
  MAPProperties::Component_Type => logic;
end ICEpcaShutoffProcess;

process implementation ICEpcaShutoffProcess.imp
subcomponents
  UpdateRespiratoryRateThread : thread UpdateRespiratoryRateThread.imp;
  PumpControlThread : thread PumpControlThread.imp;
connections
  incoming_rr : port RespiratoryRate -> UpdateRespiratoryRateThread.RespiratoryRate;
  outgoing_pump_command : port PumpControlThread.PumpNormal -> CommandPumpNormal;
end ICEpcaShutoffProcess.imp;
```
Language

Thread Specification

External ports

Properties
Language

Thread Specification

Any necessary architectural annotations can be created!
Component Development

- Development of component architecture using AADL / OSATE2
- 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
# Language Subset

## AADL Constructs Used

<table>
<thead>
<tr>
<th>AADL Construct</th>
<th>MAP Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Components</strong></td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>Layout</td>
</tr>
<tr>
<td>Device</td>
<td>Medical Device API for App</td>
</tr>
<tr>
<td>Process</td>
<td>Software Component</td>
</tr>
<tr>
<td>Thread</td>
<td>Task</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td></td>
</tr>
<tr>
<td>System-level port connection</td>
<td>Channel</td>
</tr>
<tr>
<td>Process implementation-level port connection</td>
<td>Task-Port Communication</td>
</tr>
</tbody>
</table>
Language

Translation Target

- Logic.compsig.xml
- Display.compsig.xml
- System.cfg.xml
- Logic.java
  - Task1
  - Task2
  - Task3
- Display.java
  - Task1
  - Task2
- LogicSuperType.java
- DisplaySuperType.java
- Dev1.java
- Dev2.java

- QoS/RT
Outline

- Background
- Vision
- Language
- Tool
  - OSATE2
  - Availability
- Hazard Analysis
- Future
Tool

OSATE2

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

OSATE2
Tool

OSATE2
Outline

- Background
- Vision
- Language
- Tool
- Hazard Analysis
  - History
  - Fundamentals
  - Control Actions
- Future
Hazard Analysis

Leveraging Semiformal Architectural Descriptions

Requirements

Clinical Use Case / Workflow Description

Hazard Analysis

Risk Assessment

App Developer

Assurance Case

MDCF

FDA 510K Submission Package

FDA Evaluators

3rd Party Certifiers

3rd Party ICE Conformance & Safety Certification Submission Package

App Deployment
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
      Message Garbled by Network
        G3
        Physiological Data within Max Range
          Internal Diagnostics Fail
    Undetected Error
```
## Hazard Analysis

**History: FMEA**

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

### System: PCA Interlock Scenario

<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 SpO₂</td>
<td>Fails to Provide</td>
<td>N/A</td>
<td>Network or dev. Failure</td>
<td>No SpO₂ 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 SpO₂ 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>SpO₂ 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
Hazard Analysis

**History: STPA**

- **STPA**: Nancy Leveson / MIT, 2005(ish)
- Applies systems theory, focuses on control...
  - Loops
  - Actions
STPA in AADL
The Annotated Control Loop

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

- **Feedback:** PulseOx → App
  - Inadequate Feedback: Sends bad SpO₂

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

**Actuator:** PCA Pump
- Inadequate Operation: Pumps Normally

**Sensor:** Pulse Oximeter
- Inadequate Operation: SpO₂ value incorrect

**Controlled Process:** Patient
- Inadequate Feedback: Sends bad SpO₂
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. 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
PatientHarm : constant MAP_ErrorProperties::Accident => [  
    Number => 1;  
    Description => "Patient is killed or seriously injured.";  
    Level => PulseOx_Forwarding_ErrorProperties::DeathOrInjury;
];
```
Hazard Analysis

STPA: Fundamentals

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

Example

- Process Boundary
- System Boundary
- App Boundary
- Patient
- PCA Pump
- Display
- Clinician

- Pulse Oximeter
- Capnography Device
- App
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]

Regulators: Supports strong traceability both in code and in (hypertext) reports.

Benefits:
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; ];
```
STPA in AADL

Fundamentals

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

Example

- App -> Pump: Pump Normally

Benefits:

Developers: Hazard Analysis artifacts are automatically in-sync with system architecture
Hazard Analysis

STPA: Fundamentals

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

Example

```
Patient

Pulse Oximeter

Capnography Device

App

Pump Normal
Pump KVO

PCA Pump

Display

Clinician
```

★Physiological Data

Device Ok

Device Error

Pump Normal

Pump KVO

Device Ok

Device Error

★Physiological Status

* Provide Rx
* Authorize Override

* View Patient Status
* View Device Status

Request More

Verify Rx

Verify Rx

*Physiological Status

Pump Status

* Verif

* Pump Status

* View Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Physiological Status

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok

* Device Error

* Pump Normal

* Pump KVO

* Physiological Data

* Device Ok
## 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
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 in AADL

Where should we start?

Control Action: App → PCA Pump

Feedback Message: PulseOx → App

Controller: App

Actuator: PCA Pump

Sensor: Pulse Oximeter

Controlled Process: Patient

A control action is provided in an unsafe way

How would the control action be unsafe?
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

package PCA_Interlock_System
public

system PCA_Interlock_System
describe PCA_Interlock_System;

system implementation PCA_Interlock_System.imp
subcomponents
  pulseOx : device PulseOx_Interface::MAP_PulseOx_
pcaPump : device PCAPump_Interface::MAP_PCAPump_
  appLogic : process PCA_Interlock_Logic::PCA_Inter
connections
  spo2_data : port pulseOx.SpO2 -> appLogic.SpO2;
pump_cmd : port appLogic.pumpCmd -> pcaPump.cmd;

annex EMV2 {**
  use types PCA_Interlock_Errors;
  properties
    MAP_ErrorProperties::Occurrence => {
      Guideword => Providing;
      ViolatedConstraint => PCA_Shutoff_ErrorProperties::DontLe
      Title => "High Physio Para"
      ErrorType => reference (inadvertentPumpInNormal);
      Description => "One or more physiological parameters are to
                    high, leading the app logic to incorrectly believe the
                    patient is healthy";
      Compensation => "Physiological values are cross-checked
                    with others"
    }
  applies to pump_cmd;
**};

We’ll come back to this one in a moment

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

- 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
STPA’s Causality Guidewords

Annotated Control Loop

Managers: Constrains developers so style and architectural assumptions are consistent

Developers: Guides analysis so “starting from scratch” isn’t necessary

Benefits:

Nancy Leveson. Figure 4.8, Page 93, Engineering A Safer World. MIT Press, 2011
## AADL EM Fault Types

### 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>InappropriateCtrlAction</td>
<td>InadvertentPumpNormally</td>
</tr>
<tr>
<td>ServiceOmission</td>
<td>MissingCtrlAction</td>
<td>InadvertentPumpMinimally</td>
</tr>
</tbody>
</table>

**AADL Standard Error Types** | **STPA Guidewords** | **App Specific Error Types**
AADL EM Fault Types

App Specific Error Library

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

annex EMV2
/**
 * error types
 InadvertentPumpNormally : type extends MAP_Errors::InappropriateControlAction;

 -- Could also be inadequate feedback
 SpO2ValueHigh : type extends MAP_Errors::InadequateSensorOperation;
 SpO2ValueLow : type extends MAP_Errors::InadequateSensorOperation;
 ETCO2ValueLow : type extends MAP_Errors::InadequateSensorOperation;
 ETCO2ValueHigh : type extends MAP_Errors::InadequateSensorOperation;
 RespiratoryRateLow : type extends MAP_Errors::InadequateSensorOperation;
 RespiratoryRateHigh : type extends MAP_Errors::InadequateSensorOperation;
 DeviceAlarmFailsOn : type extends MAP_Errors::InadequateSensorOperation;
 DeviceAlarmFailsOff : type extends MAP_Errors::InadequateSensorOperation;

end types;
/**
end PCA_Shutoff_Errors;
STPA in AADL

Using our fault type

Control Action: App → PCA Pump

Feedback Message: PulseOx → App

Controller: App Logic

Actuator: PCA Pump

Sensor: Pulse Oximeter

Controlled Process: Patient

Inadvertent Pump Normally
Integrated Hazard Analysis

Using our fault type

package PCA_Interlock_System
public

system PCA_Interlock_System
end PCA_Interlock_System;

system implementation PCA_Interlock_System.imp
subcomponents
  pulseOx : device PulseOx_Interface::MAP_PulseOx_Interface.imp;
  pcaPump : device PCAPump_Interface::MAP_PCAPump_Interface.imp;
  appLogic : process PCA_Interlock_Logic::PCA_Interlock_Logic.imp;
connections
  spo2_data : port pulseOx.SpO2 -> appLogic.SpO2;
  pump_cmd : port appLogic.pumpCmd -> pcaPump.cmd;
annex EMV2 {**
  use types PCA_Interlock_Errors;
  properties
    MAP_Error_Properties::Occurrence => [
      Guideword => Providing;
      ViolatedConstraint => PCA_Shutoff_Error_Properties::DontLetPumpRunWhenUnsafe;
      Title => "High Physio Params";
      ErrorType => reference(InadvertentPumpNormally);
      Description => "One or more physiological parameters are too high, leading the app logic to
                      incorrectly believe the patient is healthy";
      Compensation => "Physiological values are cross-checked with other"
    ] applies to pump_cmd;
  **};
end PCA_Interlock_System.imp;
end PCA_Interlock_System;
Where would the bad control action come from?

Control Action: App -> PCA Pump

Feedback Message: PulseOx -> App

Controller: App Logic

Process Model Incorrect: Wrongly believes patient to be healthy

Propagates error out

Actuator: PCA Pump

Sensor: Pulse Oximeter

Controlled Process: Patient

STPA in AADL
Integrated Hazard Analysis

Specification Step 1: Out Propagation

```java
package PCA_Shutoff_Life
public
with PCA_Shutoff_Types, PCA_Shutoff_Properties, MAP_Properties;

process ICEpcaShutoffProcess
features
  SpO2 : in event data port PCA_Shutoff_Types::SpO2;
  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
    HighSpO2LeadsToDeath : error path SpO2{SpO2ValueHigh} -> CommandPumpNormal{InadvertentPumpNormally};
  **};
end ICEpcaShutoffProcess;
-- Process implementation redacted
end PCA_Shutoff_Life;
```
STPA in AADL

Where would the bad control action come from?

Control Action: App $\rightarrow$ PCA Pump

Feedback Message: PulseOx $\rightarrow$ App

Controller: App Logic

Process Model Incorrect: Wrongly believes patient to be healthy

Bad information in

Actuator: PCA Pump

Sensor: Pulse Oximeter

Controlled Process: Patient
Integrated Hazard Analysis

Specification Step 2: In Propagation

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
    HighSpO2LeadsToOD : error path SpO2{SpO2ValueHigh} -> CommandPumpNormal{InadvertentPumpNormally};
  }
end ICEpcaShutoffProcess;

-- Process implementation redacted
end PCA_Shutoff_Logic;
Integrated Hazard Analysis

Specification Step 3: Relation between incoming and outgoing

**Name of flow**

**Specific Ports**

**Type of flow**

**Specific faults**
STPA in AADL

Where should we go now?

Controller: App Logic

Process Model Incorrect: Wrongly believes patient to be healthy

Option 1: Look for the source

Option 2: Look for the impact

Sensor: Pulse Oximeter

Actuator: PCA Pump

Controlled Process: Patient

Control Action: App -> PCA Pump

Feedback Message: PulseOx -> App

Controller: App Logic
STPA in AADL

Where should we go now?

Option 3:
Look for other sources / impacts
Integrated 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;

--- 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"
```
Here's an empty cell (STPA Keyword + Control Action)... could anything go wrong?

Create occurrence and supporting EM annotations.

Where else could this fault go?

What else could cause this error?
Impacts

- Automation
  - Traditionally, analysts have to mine a system and maintain it – without tool support

- Architectural integration
  - Faults can be “bound” to specific components and ports

- Future:
  - Testing + Fault Injection
    - If a compensation is claimed, we can auto-generate a test
Outline

- Background
- Vision
- Language
- Tool
- STPA
- Future
  - Next Steps
  - Tool Extensions
Next Steps

Compositional Reasoning and Assurance Cases

Clinical Use Case / Workflow Description

Requirements

Hazard Analysis

Risk Assessment

App Developer

Assurance Case

3rd Party ICE Conformance & Safety Certification Submission Package

FDA 510K Submission Package

App Deployment

MDCF

3rd Party Certifiers

FDA Evaluators
Future

Tool extensions

- Abstraction Depth
  - Model methods / functions
- Data Types
  - CORBA IDL
- MAP Device Drivers
- Logging Annotations
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
Applying STPA-based Hazard Analysis to support HBSE for Systems built using MAPs

ISPCE 2015 – Chicago, IL, USA

Sam Procter, John Hatcliff, Kim Fowler
SAnToS Lab
Kansas State University

Anura Fernando
Underwriters Laboratories

Sandy Weininger
Food and Drug Admin.

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.