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

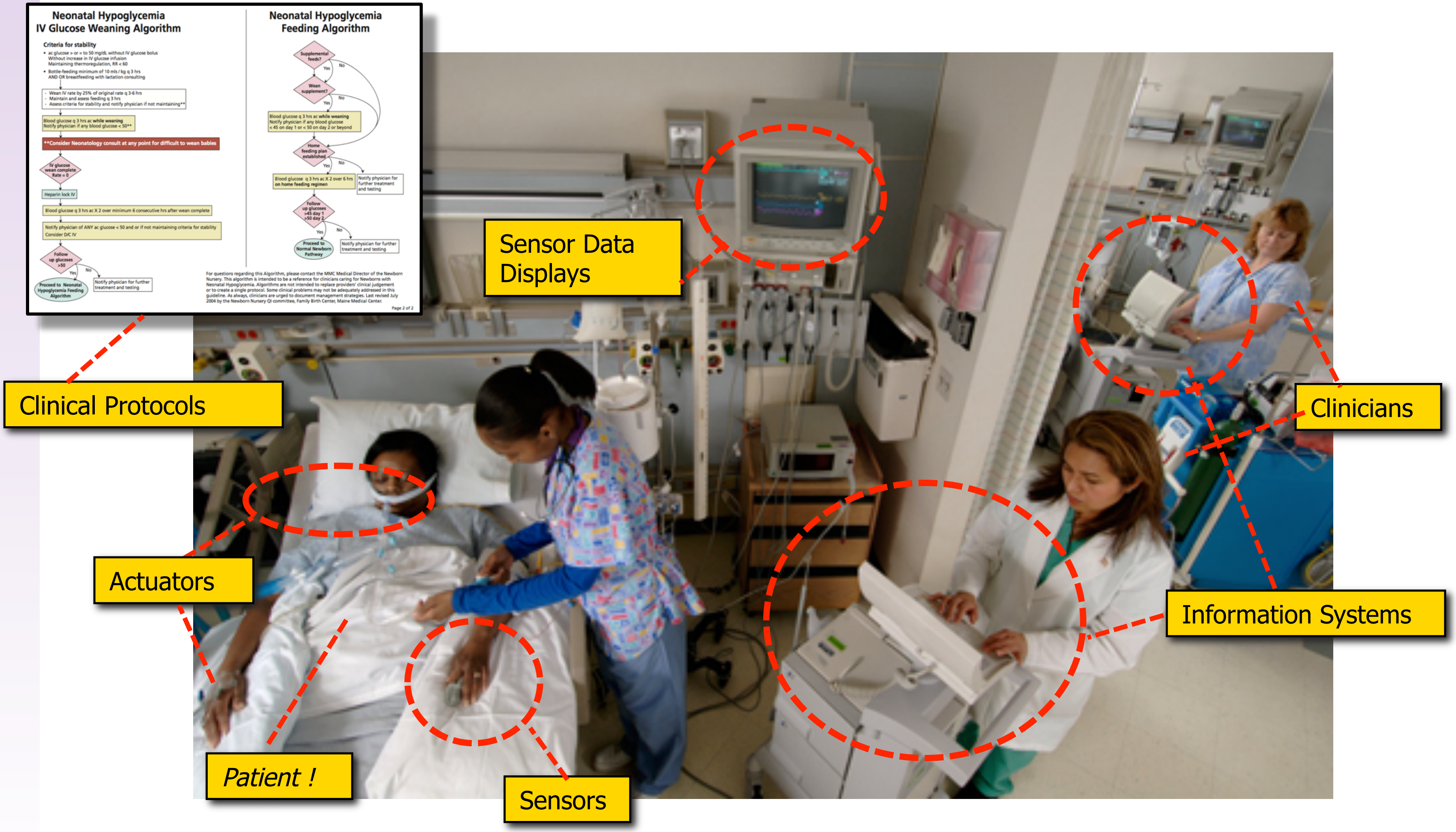
SCC 2015, Rockville Maryland

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



Neonatal Hypoglycemia IV Glucose Weaning Algorithm

Criteria for stability

- ac glucose > or = to 50 mg/dL without IV glucose bolus
  Without increase in IV glucose infusion
  Maintaining thermoregulation, RR < 60
- Bottle-feeding minimum of 10 mls / kg q 3 hrs
  AND OR breastfeeding with lactation consulting

- Wean IV rate by 25% of original rate q 3-6 hrs
- Maintain and assess feeding q 3 hrs
- Assess criteria for stability and notify physician if not maintaining**

Blood glucose q 3 hrs ac **while weaning**
Notify physician if any blood glucose < 50**

****Consider Neonatology consult at any point for difficult to wean babies**

IV glucose wean complete Rate = 0

Heparin lock IV

Blood glucose q 3 hrs ac X 2 over minimum 6 consecutive hrs after wean complete

Notify physician of ANY ac glucose < 50 and or if not maintaining criteria for stability
Consider D/C IV

Follow up glucoses >50

Yes: Proceed to Neonatal Hypoglycemia Feeding Algorithm

No: Notify physician for further treatment and testing

Neonatal Hypoglycemia Feeding Algorithm

Supplemental feeds?

Yes / No

Wean supplement?

Yes / No

Blood glucose q 3 hrs ac **while weaning**
Notify physician if any blood glucose
< 45 on day 1 or < 50 on day 2 or beyond

Home feeding plan established

Yes: Blood glucose q 3 hrs ac X 2 over 6 hrs **on home feeding regimen**

No: Notify physician for further treatment and testing

Follow up glucoses >45 day 1 >50 day 2

Yes: Proceed to Normal Newborn Pathway

No: Notify physician for further treatment and testing

For questions regarding this Algorithm, please contact the MMC Medical Director of the Newborn Nursery. This algorithm is intended to be a reference for clinicians caring for Newborns with Neonatal Hypoglycemia. Algorithms are not intended to replace providers' clinical judgement or to create a single protocol. Some clinical problems may not be adequately addressed in this guideline. As always, clinicians are urged to document management strategies. Last revised July 2004 by the Newborn Nursery QI committee, Family Birth Center, Maine Medical Center.

Page 2 of 2

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

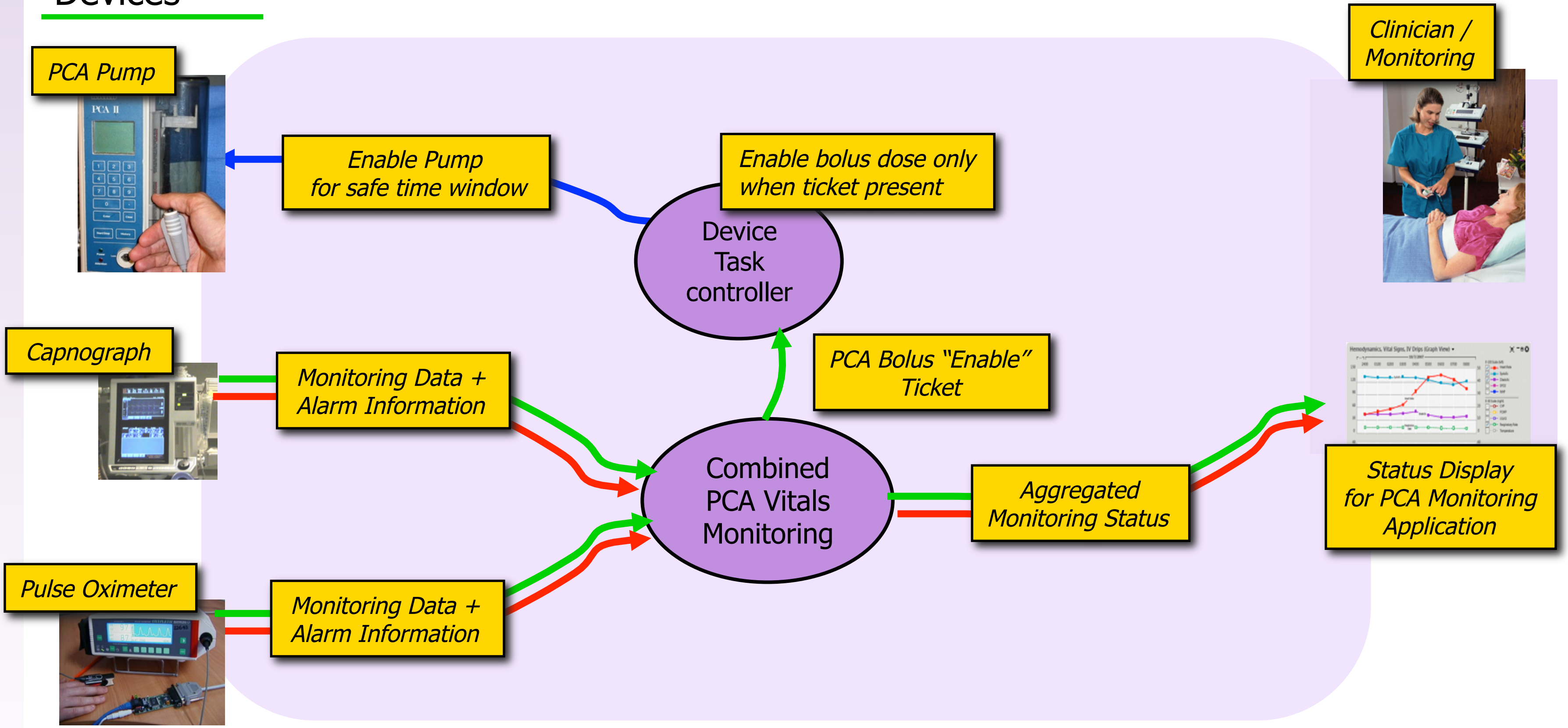
PCA Interlock Scenario

- 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

Enable bolus dose only when ticket present

Device Task controller

Clinician / Monitoring

Capnograph

Monitoring Data + Alarm Information

PCA Bolus "Enable" Ticket

Combined PCA Vitals Monitoring

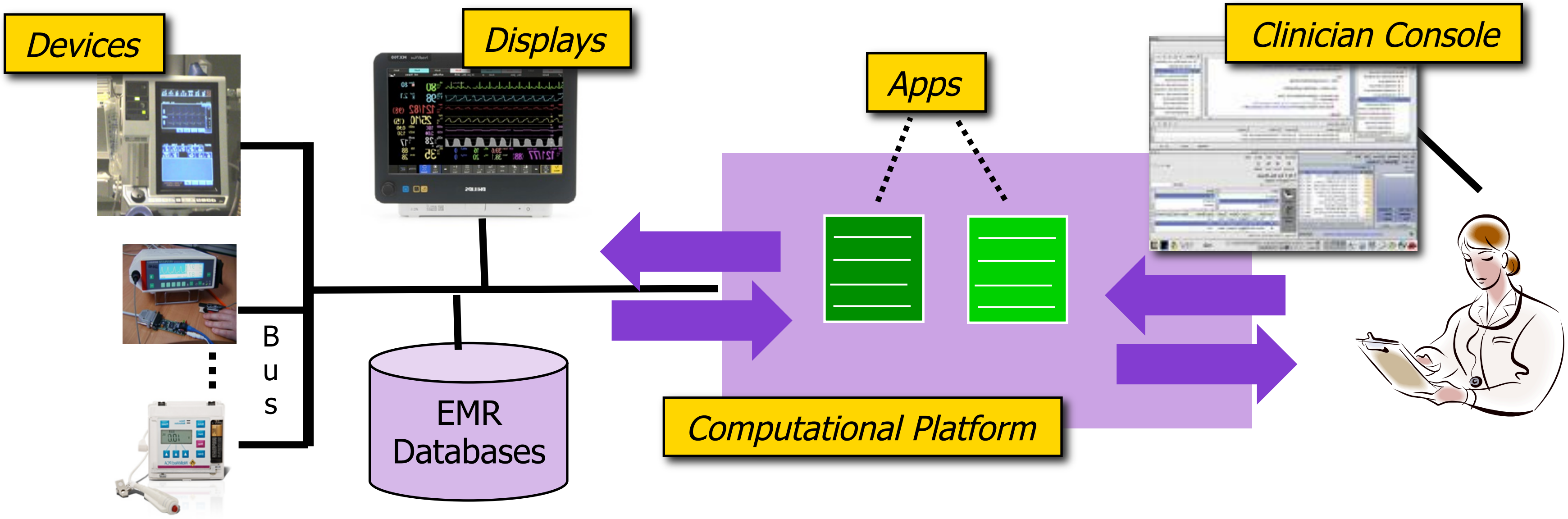
Aggregated Monitoring Status

Status Display for PCA Monitoring Application

Pulse Oximeter

Monitoring Data + Alarm Information

Medical Application Platforms



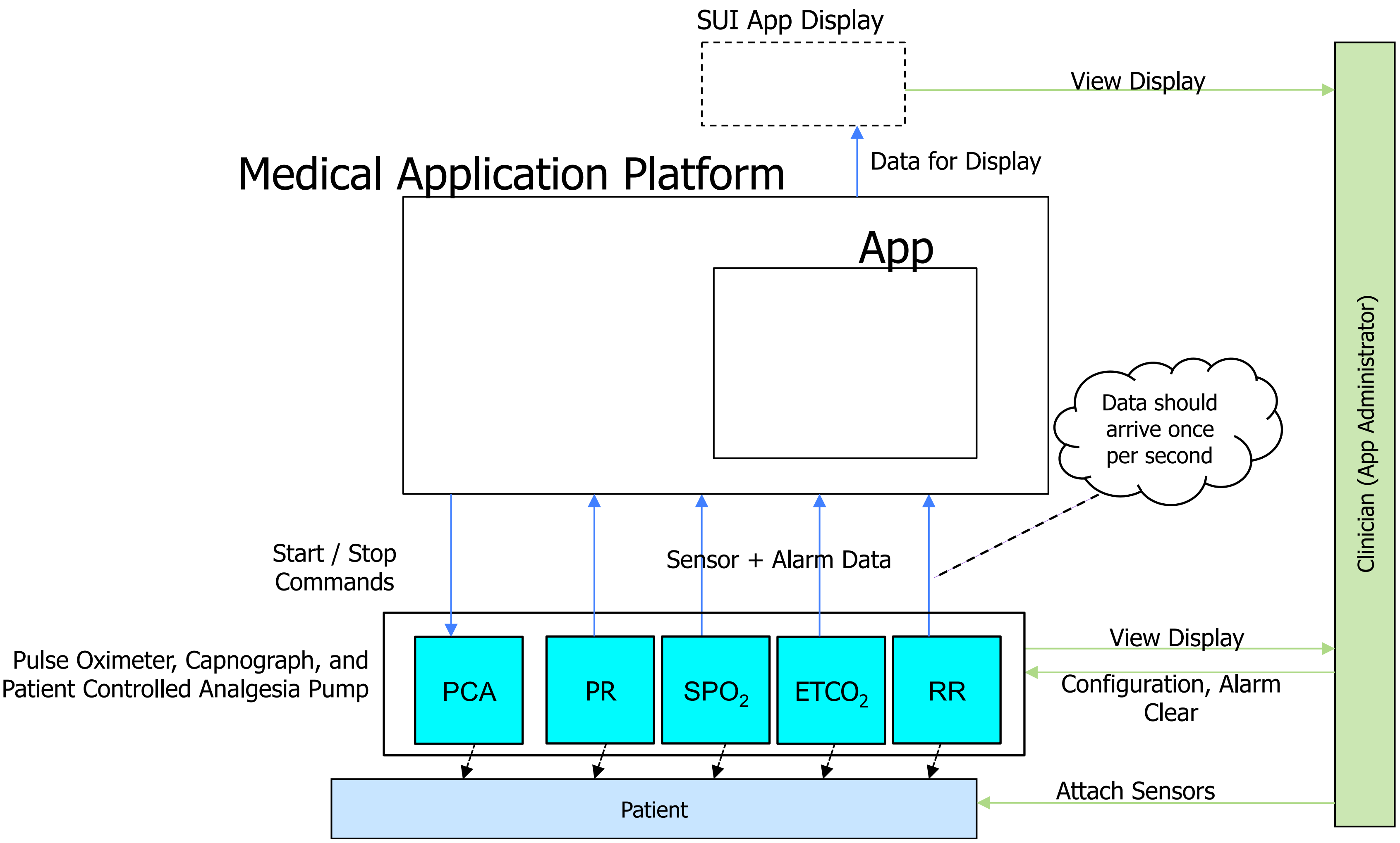
- A *Medical Application Platform* is a safety- and security-critical real-time computing platform for…
  - Integrating heterogeneous devices, medical IT systems, and information displays via communications infrastructure, and
  - Hosting applications ("apps") that provide medical utility via the ability to acquire information from and update/control integrated devices, IT systems, and displays

Extension beyond medicine

- 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

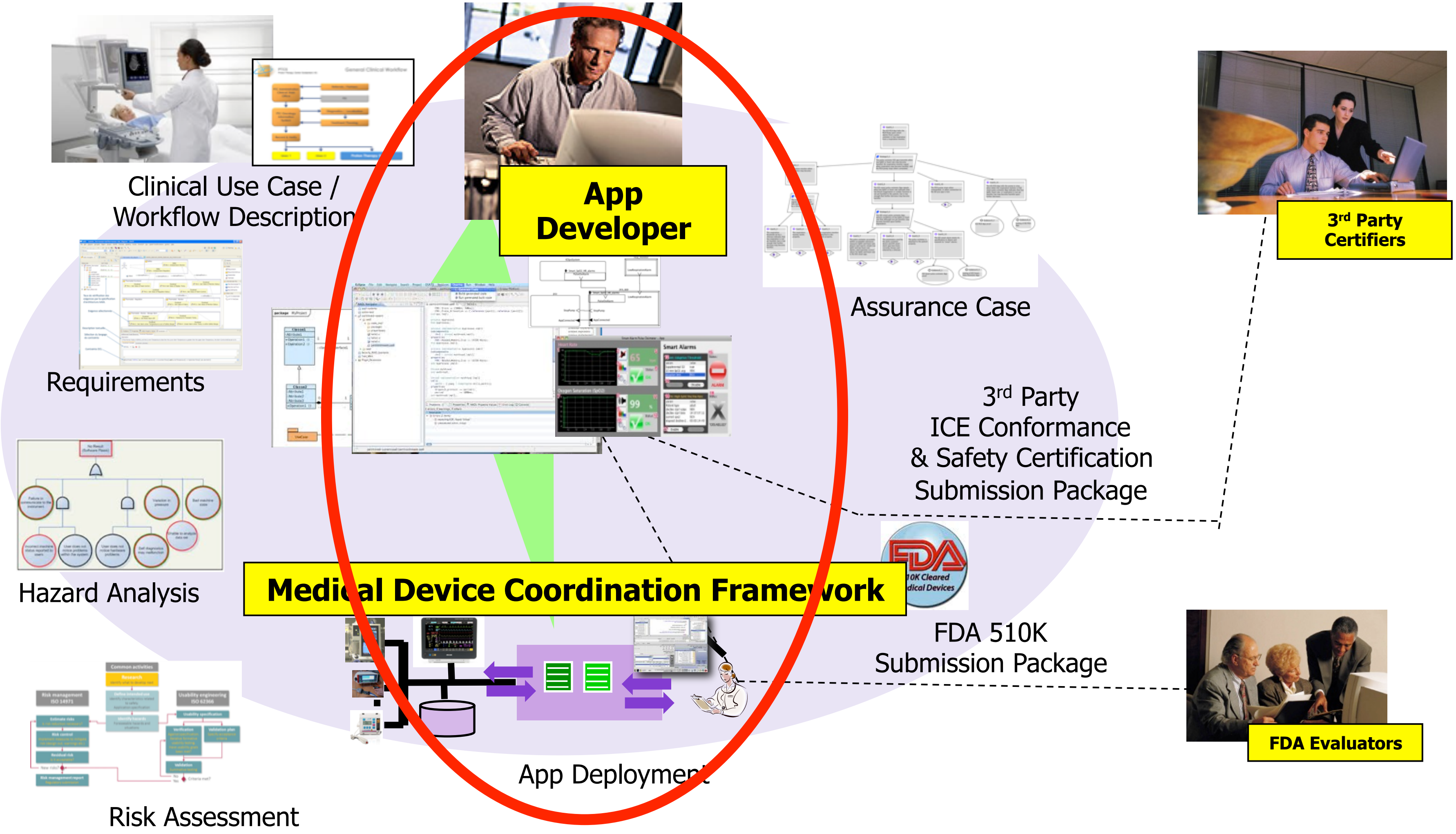
Background

PCA Pump Interlock Architecture


SUI App Display
View Display
Data for Display
Medical Application Platform
App
Data should arrive once per second
Clinician (App Administrator)
Start / Stop Commands
Sensor + Alarm Data
Pulse Oximeter, Capnograph, and Patient Controlled Analgesia Pump
PCA
PR
SPO_2
$ETCO_2$
RR
View Display
Configuration, Alarm Clear
Attach Sensors
Patient

Tooling Vision

Analyses and Regulatory Artifacts



Clinical Use Case / Workflow Description

App Developer

Assurance Case

3rd Party Certifiers

Requirements

3rd Party ICE Conformance & Safety Certification Submission Package

Hazard Analysis

Medical Device Coordination Framework

FDA 510K Submission Package

FDA Evaluators

Risk Assessment

App Deployment

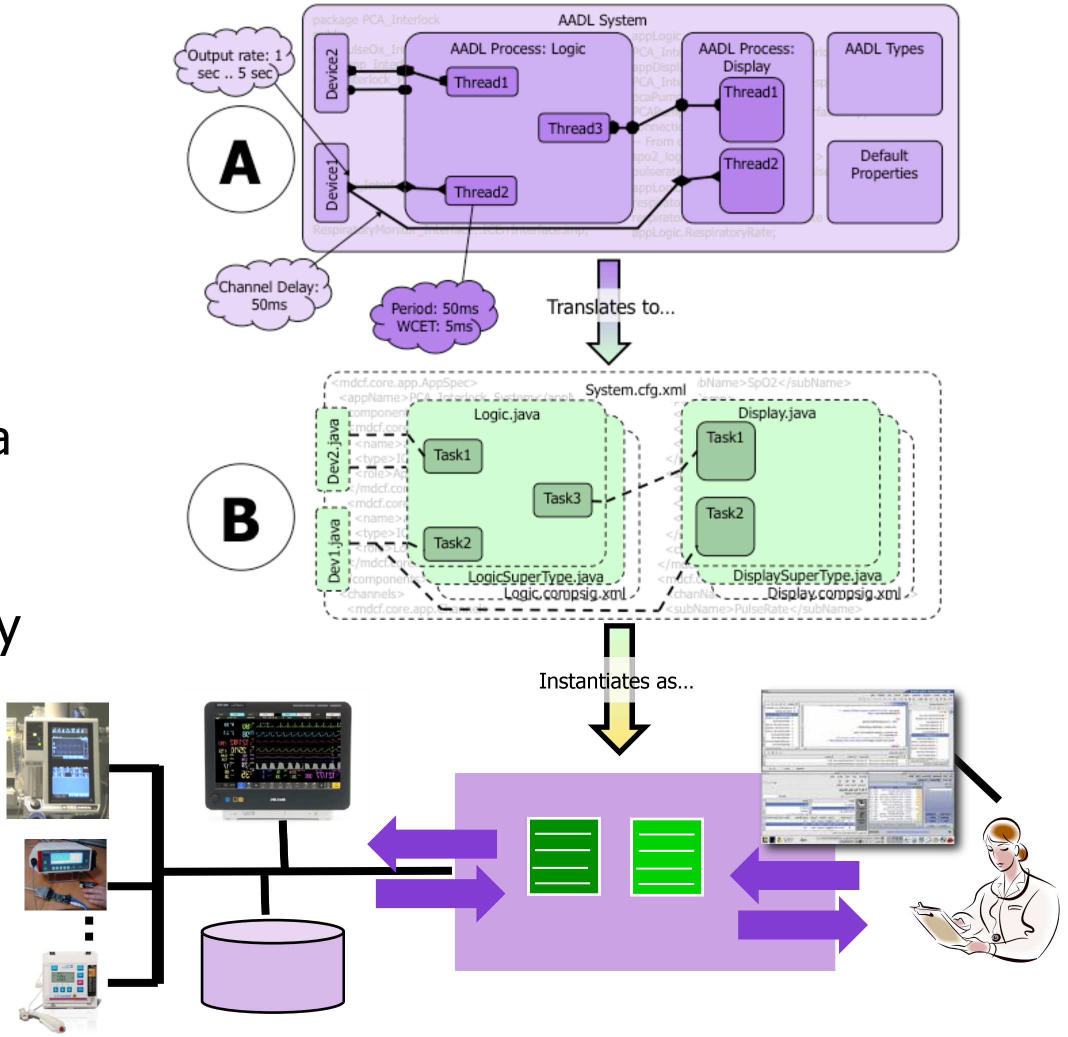
Tooling Vision

Code Generation

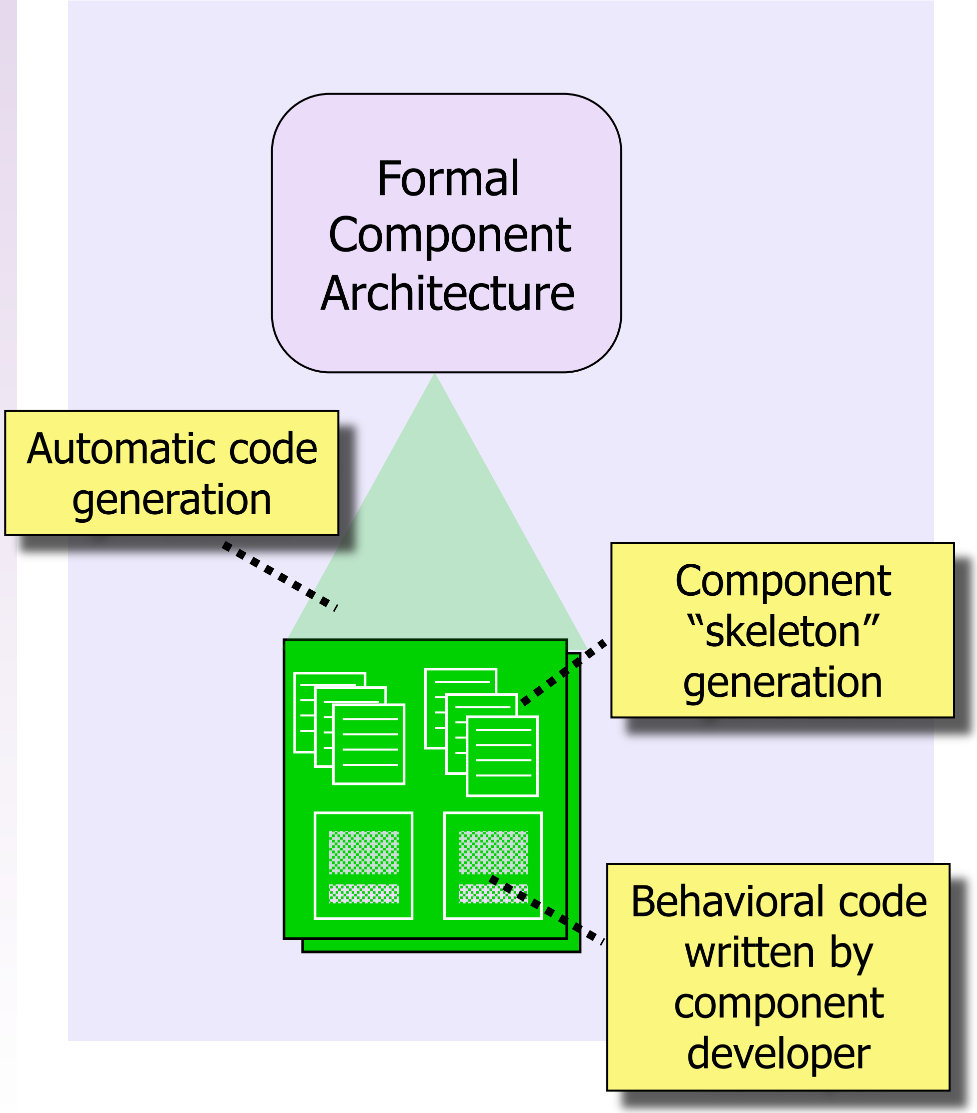
A. The app's architecture is specified in a suitable formalism
   1. Components as AADL Devices / Processes
   2. Connections are specified
   3. RT/QoS Parameters are via AADL's property-specification mechanism

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


Formal Component Architecture
Automatic code generation
Component "skeleton" generation
Behavioral code written by component developer

- 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

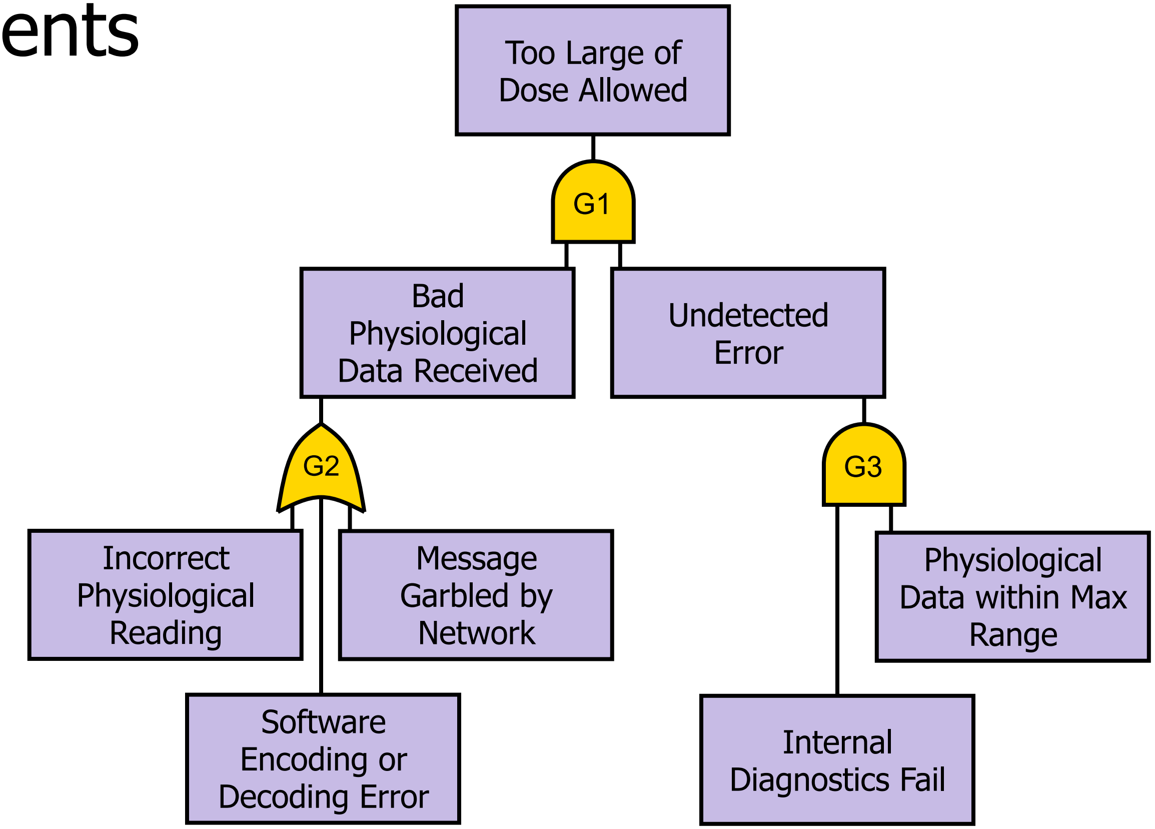
Outline

- 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
Undetected Error
G2
G3
Incorrect Physiological Reading
Message Garbled by Network
Physiological Data within Max Range
Software Encoding or Decoding Error
Internal Diagnostics Fail

Hazard Analysis

History: FMEA

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

System: PCA Interlock Scenario			Subsystem: Pulse Oximeter Device				Mode/Phase: Execution			
Function	**Failure Mode**	**Fail Rate**	**Causal Factors**	**Effect**	**System Effect**	**Detected by**	**Current Control**	**Hazard**	**Risk**	**Rec. Action**
Provide SpO_2	Fails to Provide	N/A	Network or dev. Failure	No SpO_2 data	Unknown patient state	App		Potential OD	3D	Default to KVO
	Provides late	N/A	Network slowness	No SpO_2 data	Unknown patient state	App		Potential OD	3C	Default to KVO
	Provides wrong	N/A	Device error	SpO_2 wrong	Wrong patient state	None		Potential OD	1E	Dev. should report data quality
Analyst: Sam Procter			Date: September 26, 2014				Page 3/14			

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

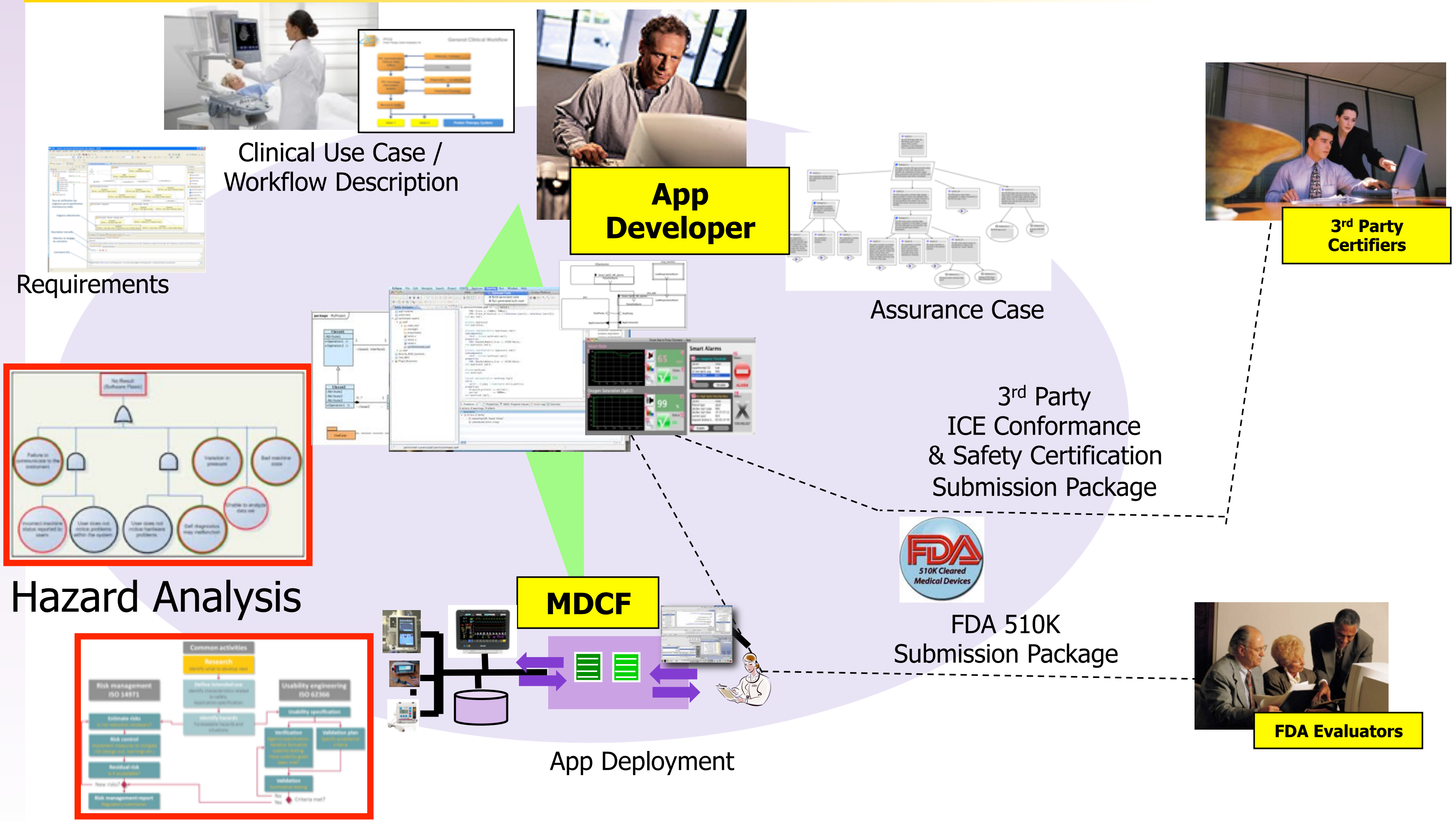
- Formal architecture descriptions become the scaffolding on which:
  - Requirements,
  - Development,
  - Risk management,
  - Deployment, and
  - Ecosphere coordination is organized.

Outline

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

Hazard Analysis

Leveraging Semiformal Architectural Descriptions


No Result (Software Flaws)
Failure to communicate to the instrument
Variation in pressure
Bad machine state
Unable to analyze data set
Incorrect machine status reported to users
User does not notice problems within the system
User does not notice hardware problems
Self diagnostics may malfunction
Common activities
Research
Risk management ISO 14971
Usability engineering ISO 62366
Define intended use
Identify hazards
Estimate risks
Risk control
Residual risk
Risk management report
Usability specification
Verification
Validation plan
Validation
New risks?
No
Yes
Criteria met?
Smart Alarms
65
99
510K Cleared Medical Devices

App Developer

Clinical Use Case / Workflow Description

Requirements

Assurance Case

3rd Party Certifiers

3rd Party ICE Conformance & Safety Certification Submission Package

Hazard Analysis

MDCF

FDA 510K Submission Package

FDA Evaluators

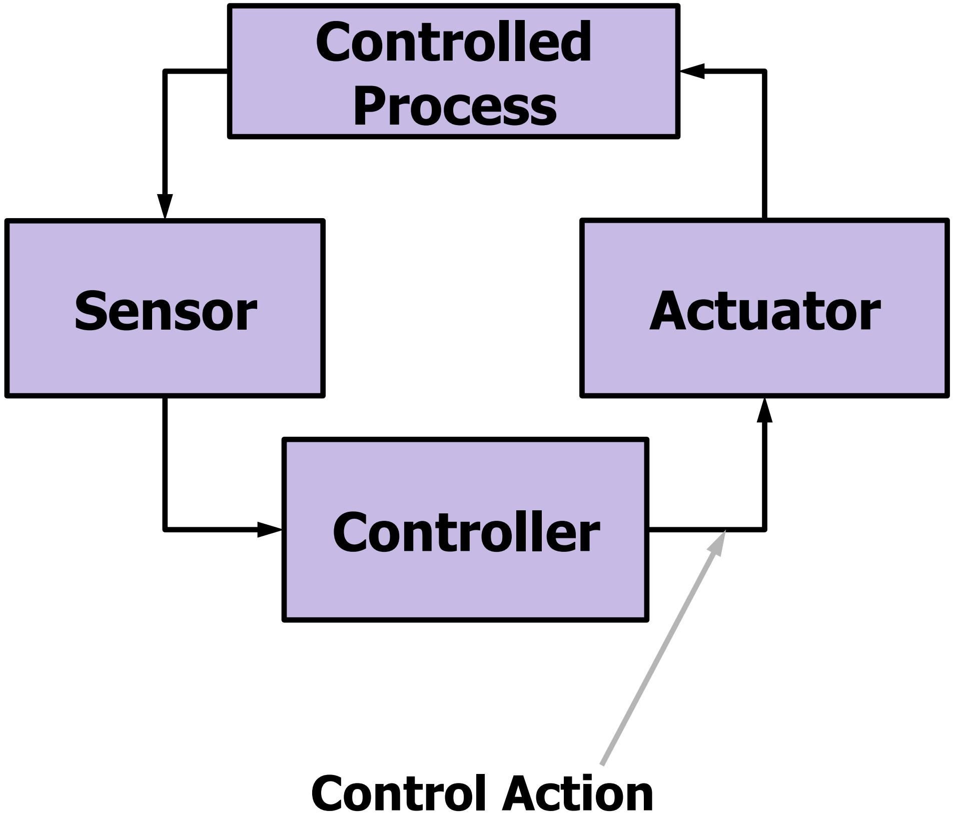
App Deployment

Risk Assessment

Hazard Analysis

History: STPA

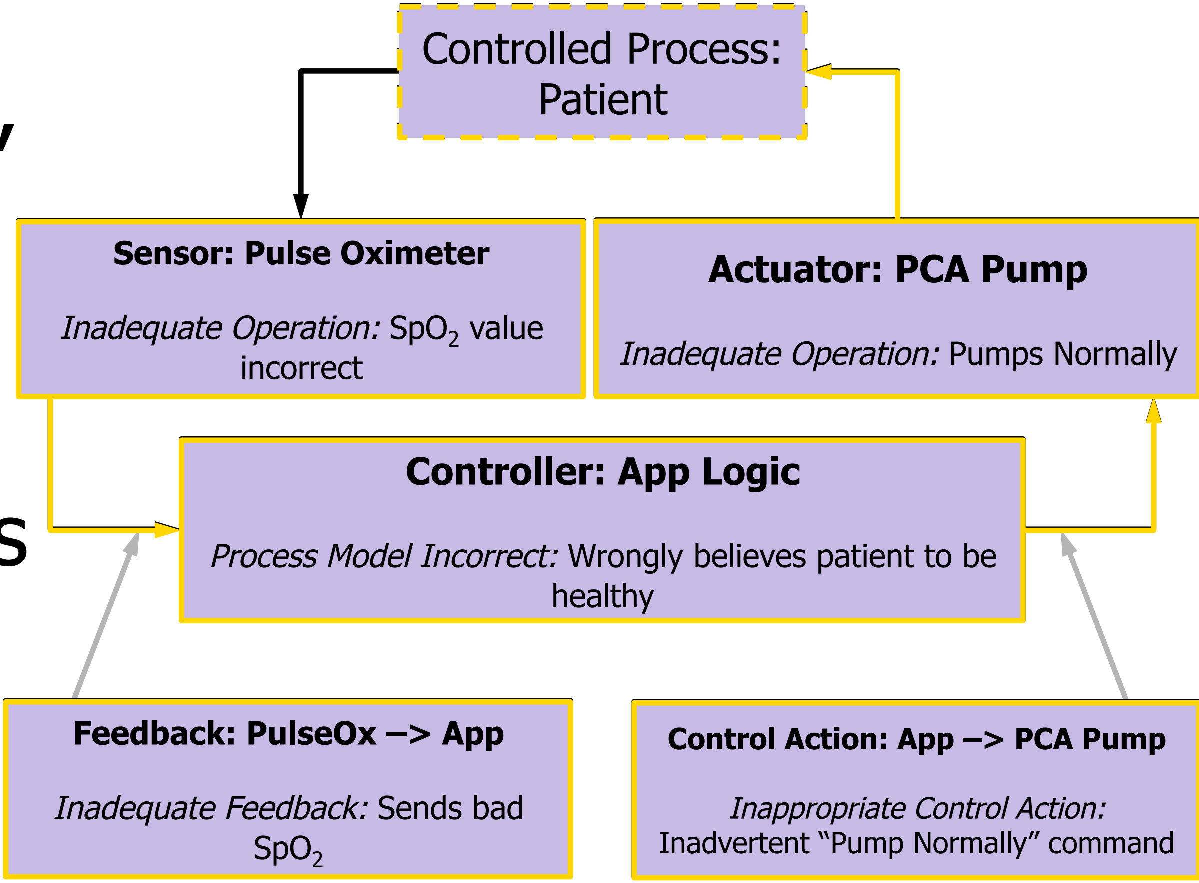
- STPA: Nancy Leveson / MIT, 2005(ish)
- Applies systems theory, focuses on control…
    - Loops
    - Actions



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

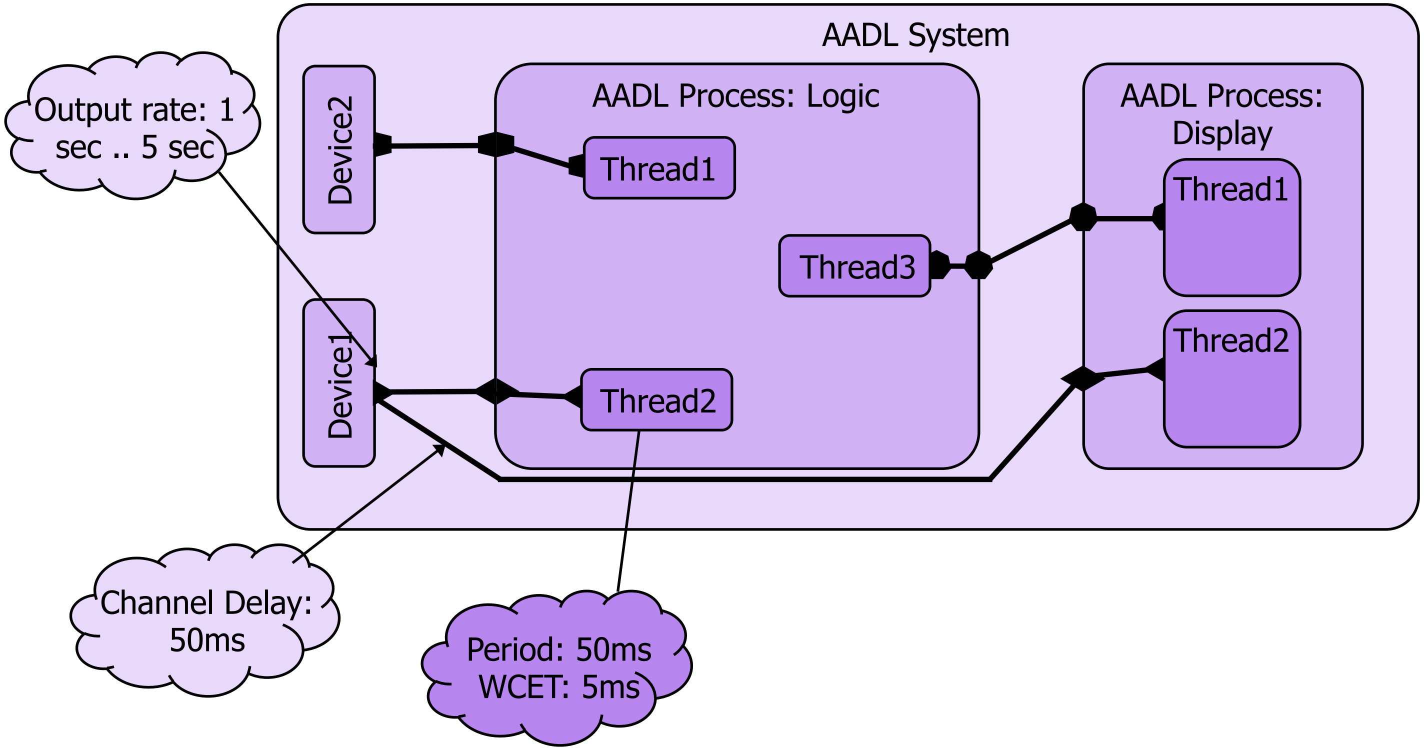
Language

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

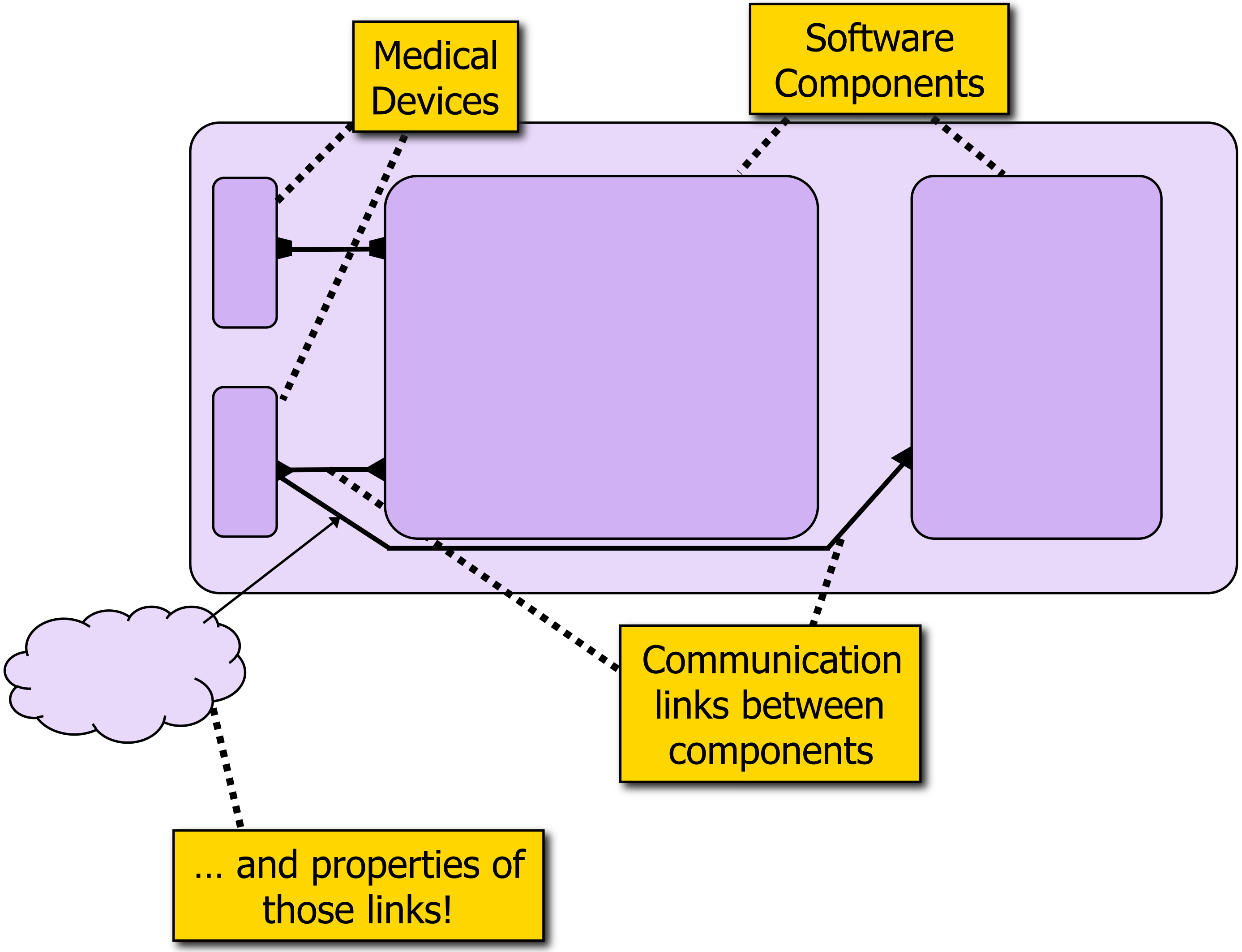
Language

Model



Language

System



Medical Devices

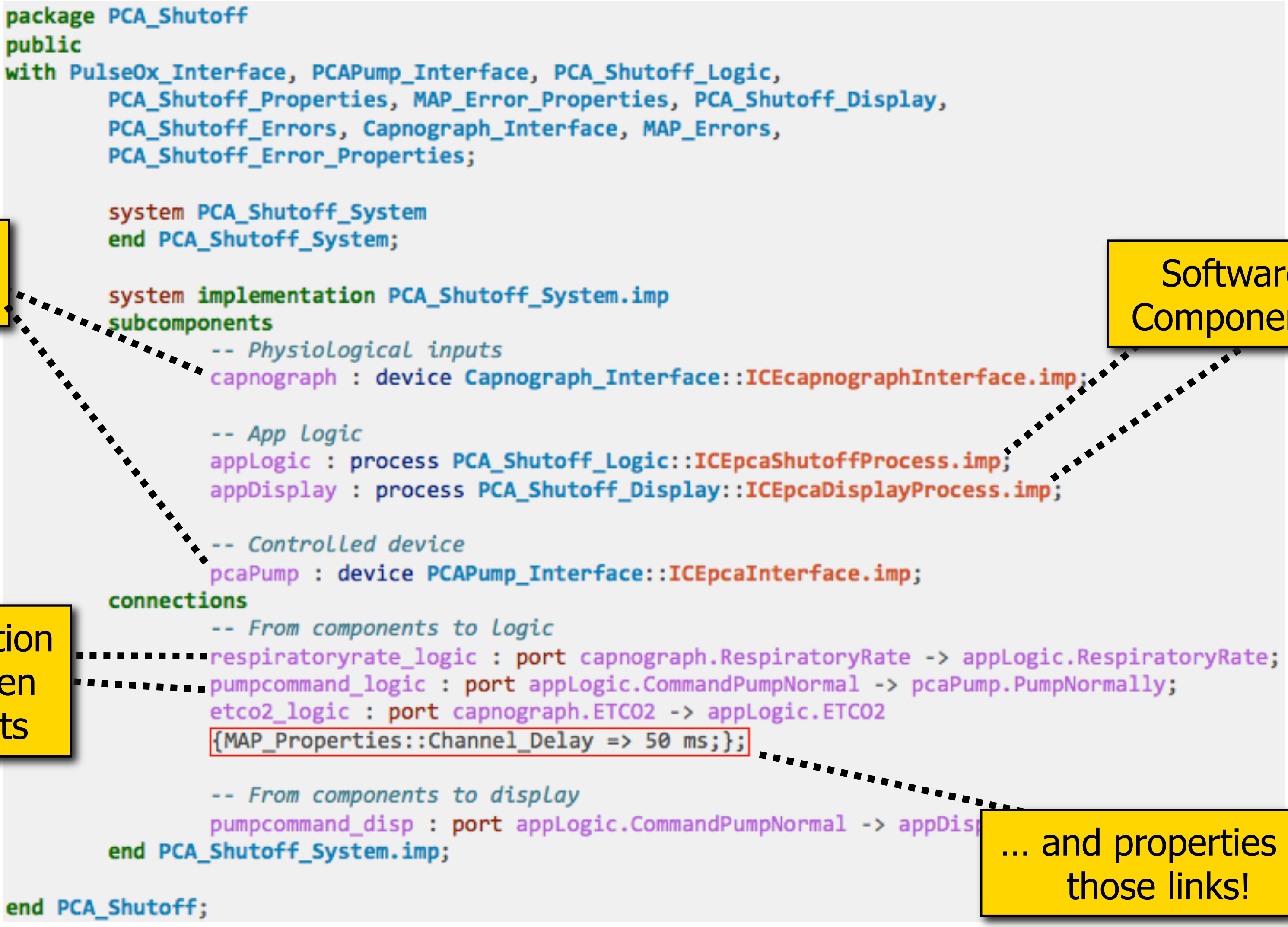
Software Components

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 -> appDis
        end PCA_Shutoff_System.imp;

end PCA_Shutoff;
```

Medical Devices

Software Components

Communication links between Components

… and properties of those links!

STPA: Fundamentals

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

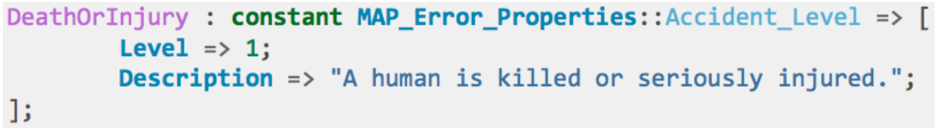
Hazard Analysis

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



```
DeathOrInjury : constant MAP_Error_Properties::Accident_Level => [
        Level => 1;
        Description => "A human is killed or seriously injured.";
];
```

Tie into ISO 14971's notions of criticality?

Hazard Analysis

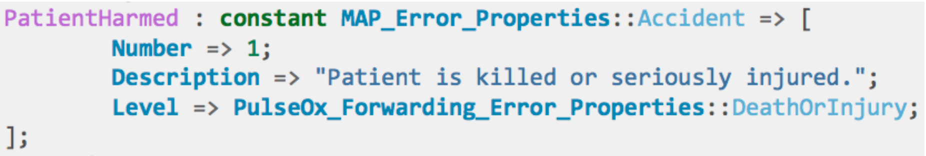
STPA: Fundamentals

- 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]



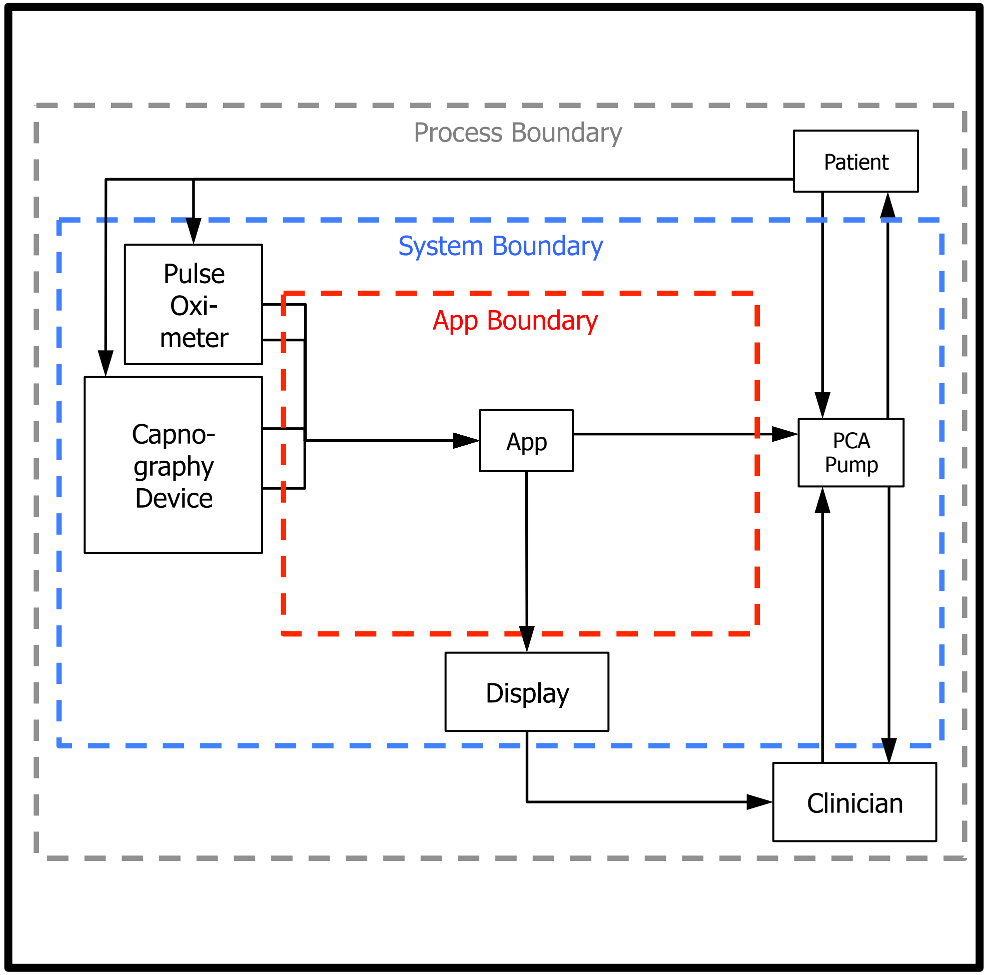
```
PatientHarmed : constant MAP_Error_Properties::Accident => [
        Number => 1;
        Description => "Patient is killed or seriously injured.";
        Level => PulseOx_Forwarding_Error_Properties::DeathOrInjury;
];
```

Hazard Analysis

STPA: Fundamentals

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

Example



Hazard Analysis

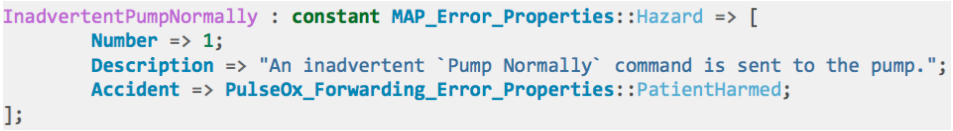
STPA: Fundamentals

- 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]



```
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

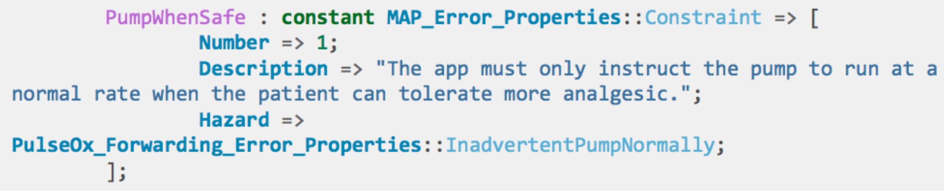
STPA: Fundamentals

- 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]



```
        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;
        ];
```

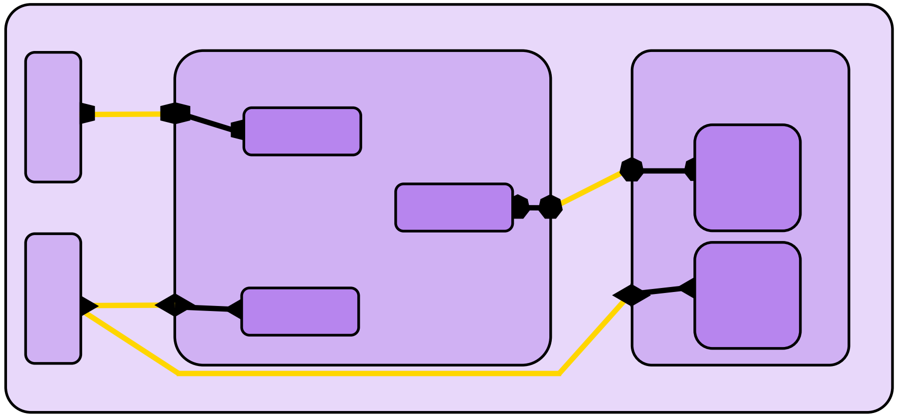
Hazard Analysis

STPA: Fundamentals

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

Example

1. App -> Pump: Pump Normally
2. PulseOx -> App: SpO_2 = 95
3. App -> Display: Patient = Ok

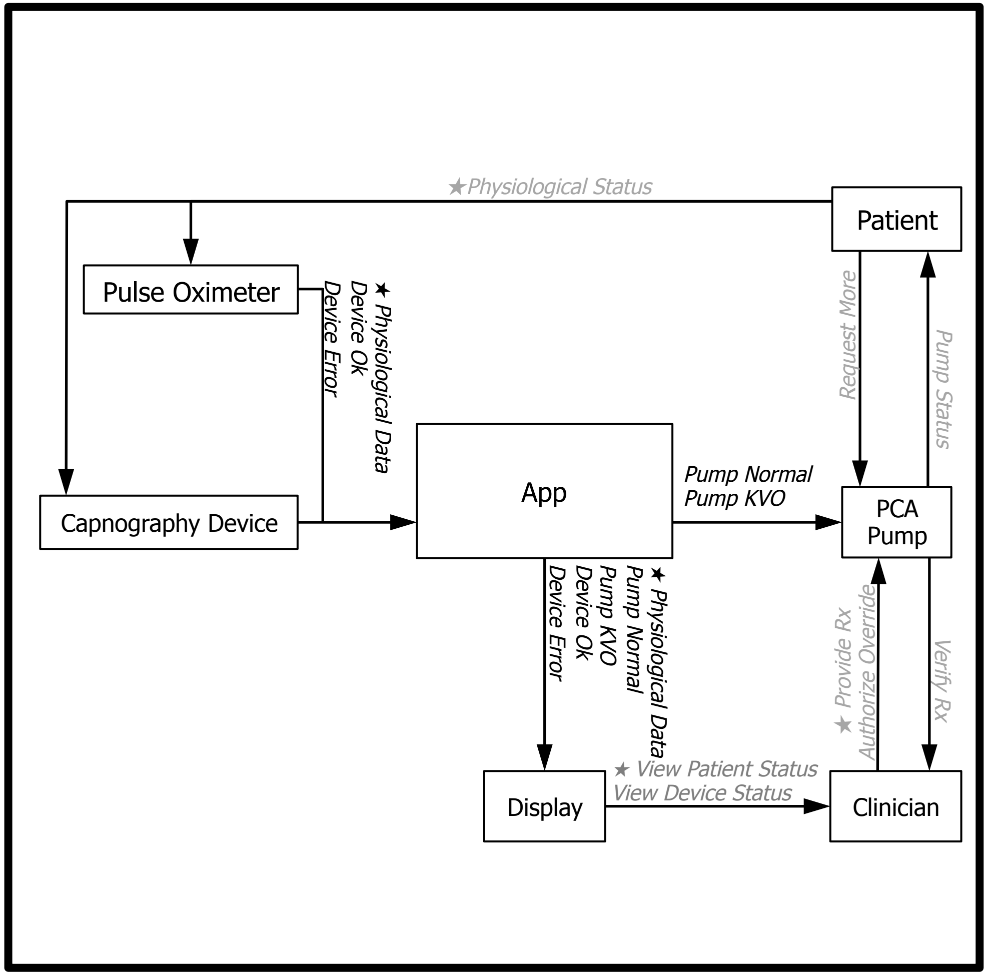


Hazard Analysis

STPA: Fundamentals

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

Example



Hazard Analysis

STPA: Identifying Hazardous Control Actions

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

Control Action	Providing	Not Providing	Applied too Long	Stopped too Soon	Early	Late
App -> Pump: Pump Normally	PH	Not Hazardous	PH	Not Hazardous	PH	Not Hazardous
App -> Disp: Patient Ok	BID	BID	BID	BID	BID	BID
PulseOx->App: Provide SpO_2	Not Hazardous	PH, BID	Not Hazardous	PH, BID	Not Hazardous	PH, BID
PulseOx->App: Provide Pulse Rate	Not Hazardous	PH, BID	Not Hazardous	PH, BID	Not Hazardous	PH, BID

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

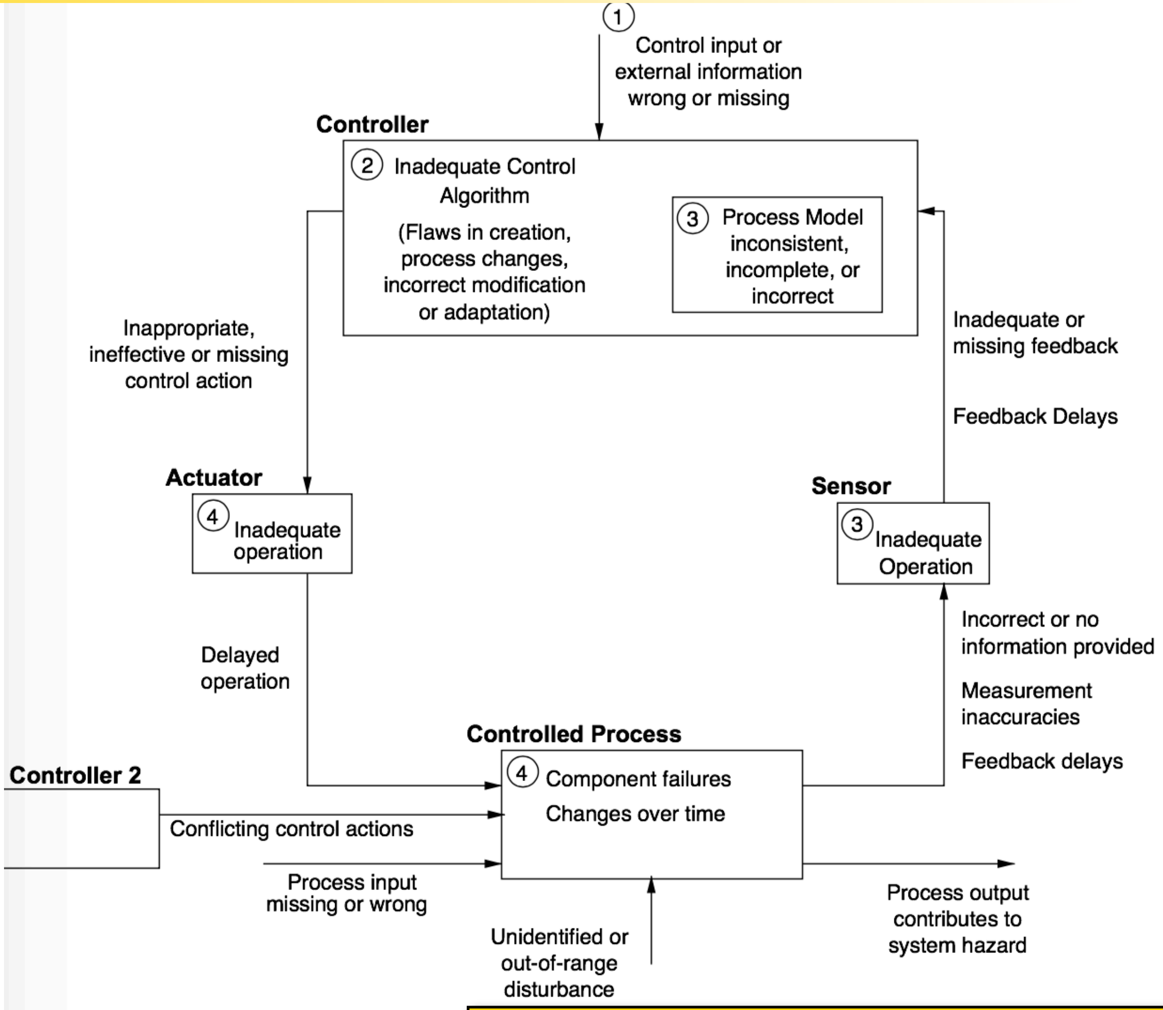
STPA: Hazardous Causes and Compensations

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

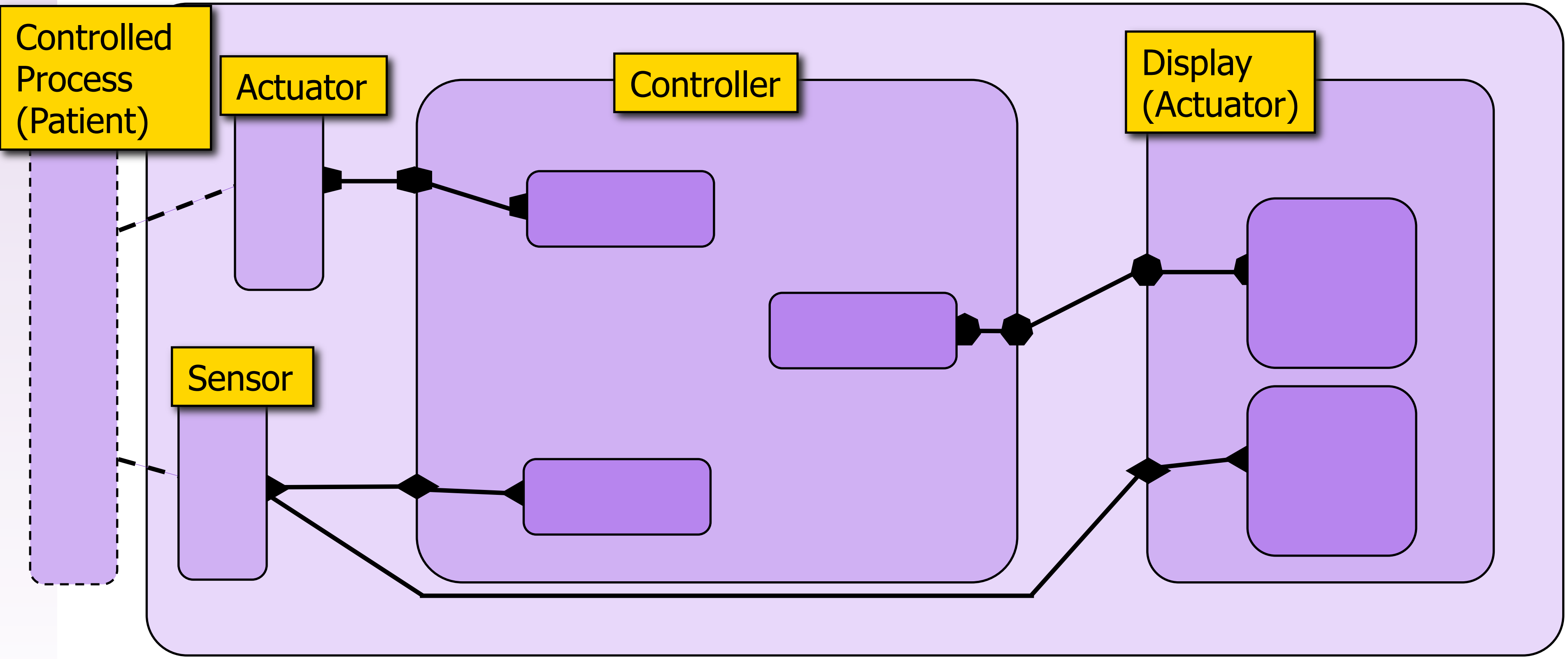
Including Causality Guidewords


1
Control input or external information wrong or missing
Controller
2 Inadequate Control Algorithm
(Flaws in creation, process changes, incorrect modification or adaptation)
3 Process Model inconsistent, incomplete, or incorrect
Inappropriate, ineffective or missing control action
Inadequate or missing feedback
Feedback Delays
Actuator
4 Inadequate operation
Sensor
3 Inadequate Operation
Delayed operation
Incorrect or no information provided
Measurement inaccuracies
Feedback delays
Controlled Process
4 Component failures
Changes over time
Controller 2
Conflicting control actions
Process input missing or wrong
Unidentified or out-of-range disturbance
Process output contributes to system hazard

"Engineering a Safer World" Leveson, 2011

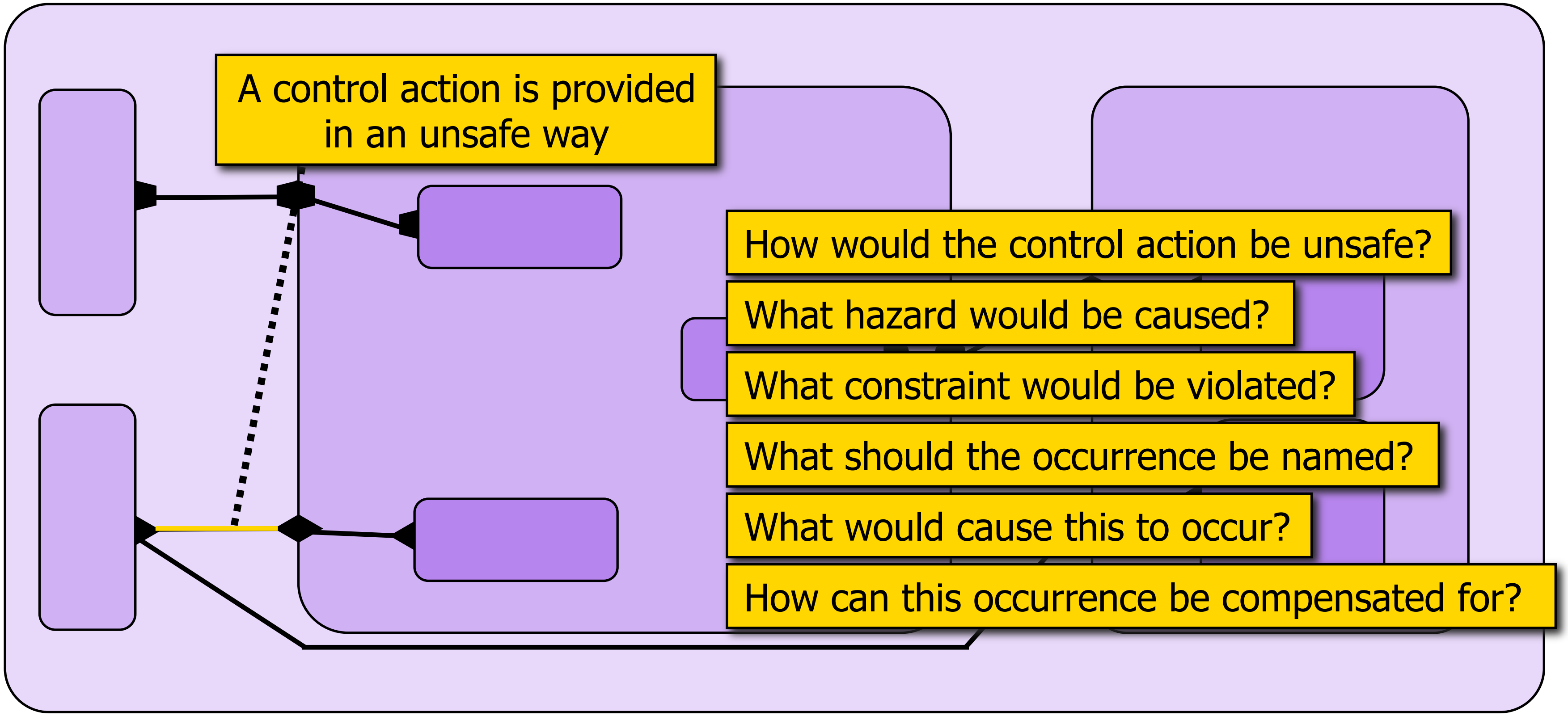
Hazard Analysis

Returning to our Architectural Model



Hazard Analysis

Annotating 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?

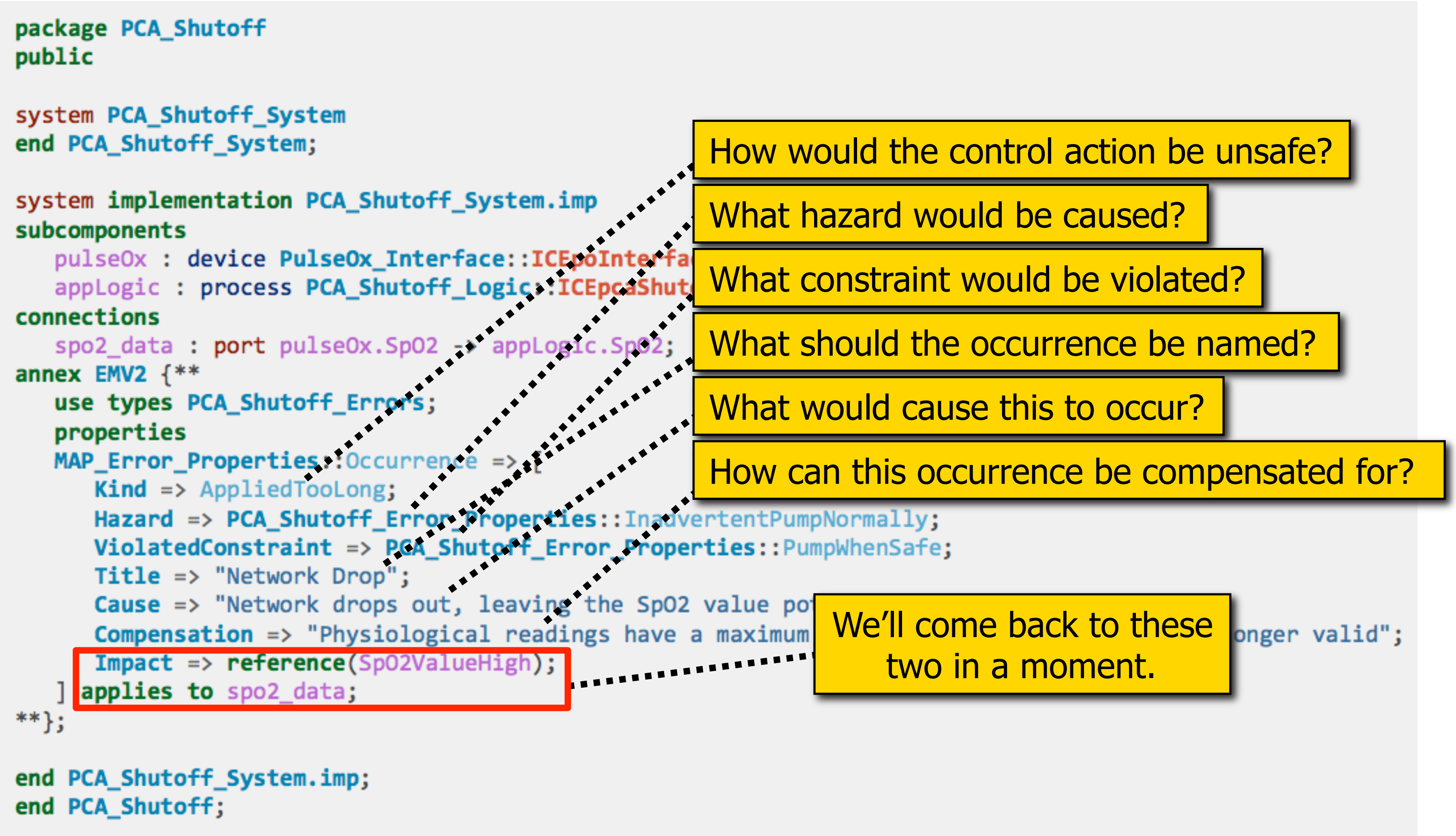
What should the occurrence be named?

What would cause this to occur?

How can this occurrence be compensated for?

Hazard Analysis

Annotating our Architectural Model



```
package PCA_Shutoff
public

system PCA_Shutoff_System
end PCA_Shutoff_System;

system implementation PCA_Shutoff_System.imp
subcomponents
   pulseOx : device PulseOx_Interface::ICEpoInterfa
   appLogic : process PCA_Shutoff_Logic::ICEpcaShut
connections
   spo2_data : port pulseOx.SpO2 -> appLogic.SpO2;
annex EMV2 {**
   use types PCA_Shutoff_Errors;
   properties
   MAP_Error_Properties::Occurrence => [
      Kind => AppliedTooLong;
      Hazard => PCA_Shutoff_Error_Properties::InadvertentPumpNormally;
      ViolatedConstraint => PCA_Shutoff_Error_Properties::PumpWhenSafe;
      Title => "Network Drop";
      Cause => "Network drops out, leaving the SpO2 value po
      Compensation => "Physiological readings have a maximum                          onger valid";
      Impact => reference(SpO2ValueHigh);
   ] applies to spo2_data;
**};

end PCA_Shutoff_System.imp;
end PCA_Shutoff;
```

How would the control action be unsafe?

What hazard would be caused?

What constraint would be violated?

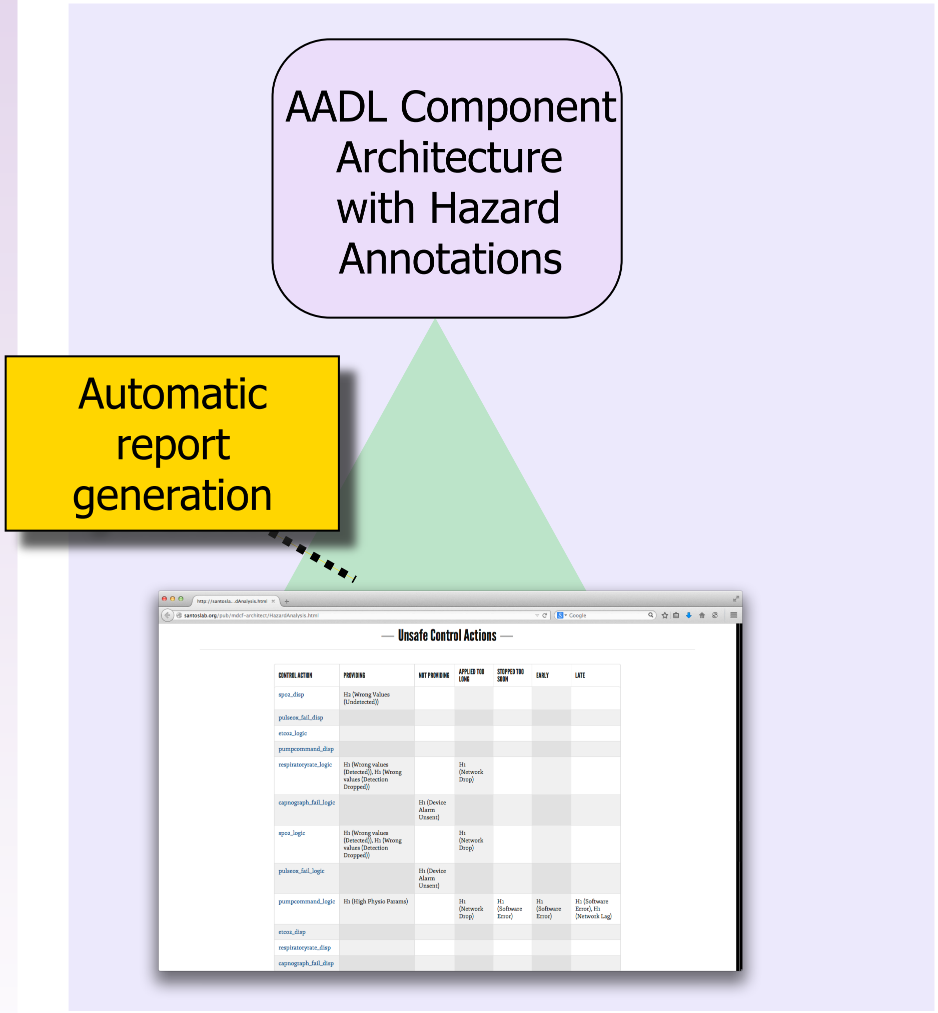
What should the occurrence be named?

What would cause this to occur?

How can this occurrence be compensated for?

We'll come back to these two in a moment.

Report Generation Development


AADL Component Architecture with Hazard Annotations
Automatic report generation
http://santosla...dAnalysis.html
santoslab.org/pub/mdcf-architect/HazardAnalysis.html
Google

— Unsafe Control Actions —

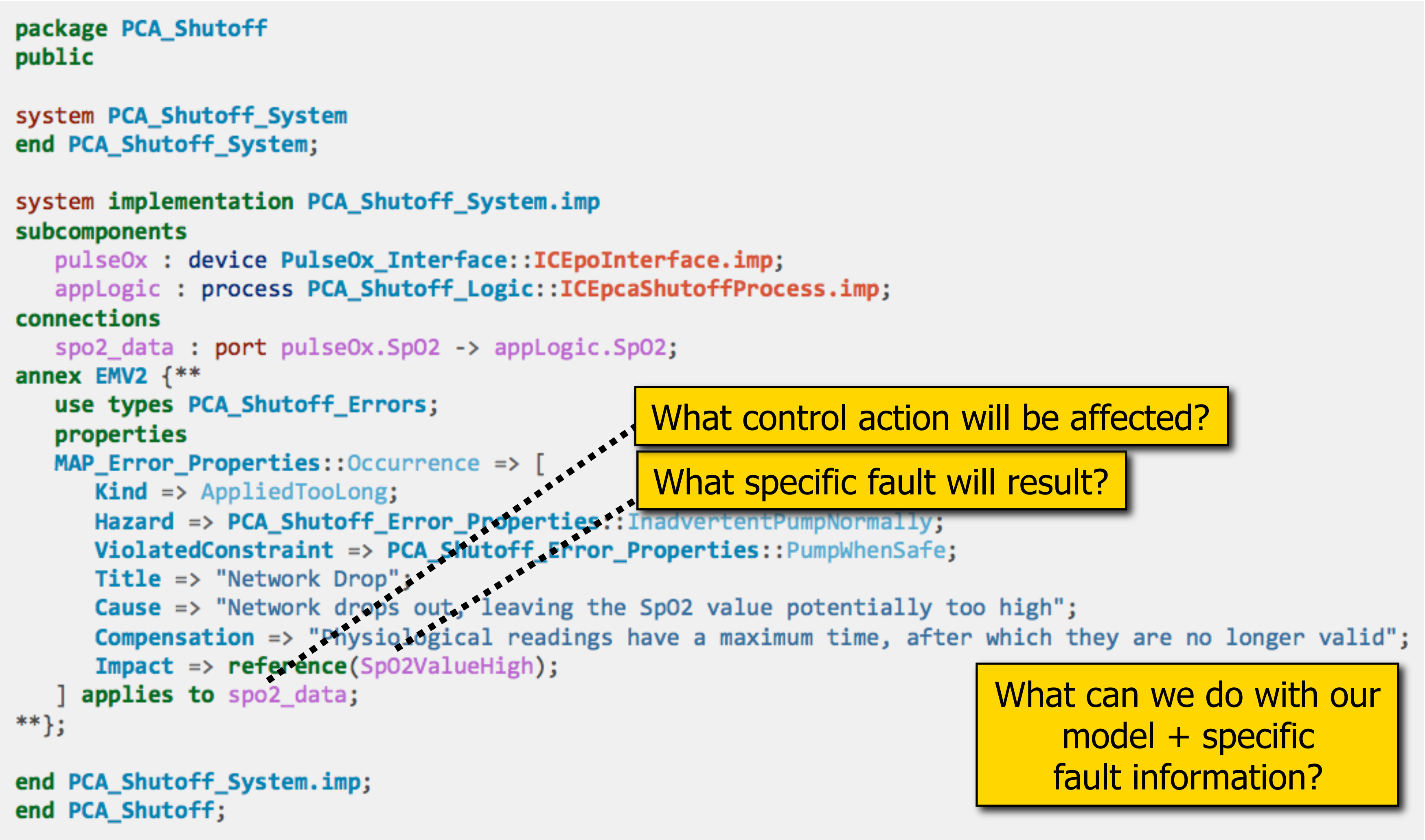
CONTROL ACTION	PROVIDING	NOT PROVIDING	APPLIED TOO LONG	STOPPED TOO SOON	EARLY	LATE
spo2_disp	H2 (Wrong Values (Undetected))					
pulseox_fail_disp						
etco2_logic						
pumpcommand_disp						
respiratoryrate_logic	H1 (Wrong values (Detected)), H1 (Wrong values (Detection Dropped))		H1 (Network Drop)			
capnograph_fail_logic		H1 (Device Alarm Unsent)				
spo2_logic	H1 (Wrong values (Detected)), H1 (Wrong values (Detection Dropped))		H1 (Network Drop)			
pulseox_fail_logic		H1 (Device Alarm Unsent)				
pumpcommand_logic	H1 (High Physio Params)		H1 (Network Drop)	H1 (Software Error)	H1 (Software Error)	H1 (Software Error), H1 (Network Lag)
etco2_disp						
respiratoryrate_disp						
capnograph_fail_disp						

- 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



```
package PCA_Shutoff
public

system PCA_Shutoff_System
end PCA_Shutoff_System;

system implementation PCA_Shutoff_System.imp
subcomponents
   pulseOx : device PulseOx_Interface::ICEpoInterface.imp;
   appLogic : process PCA_Shutoff_Logic::ICEpcaShutoffProcess.imp;
connections
   spo2_data : port pulseOx.SpO2 -> appLogic.SpO2;
annex EMV2 {**
   use types PCA_Shutoff_Errors;
   properties
   MAP_Error_Properties::Occurrence => [
      Kind => AppliedTooLong;
      Hazard => PCA_Shutoff_Error_Properties::InadvertentPumpNormally;
      ViolatedConstraint => PCA_Shutoff_Error_Properties::PumpWhenSafe;
      Title => "Network Drop";
      Cause => "Network drops out, leaving the SpO2 value potentially too high";
      Compensation => "Physiological readings have a maximum time, after which they are no longer valid";
      Impact => reference(SpO2ValueHigh);
   ] applies to spo2_data;
**};

end PCA_Shutoff_System.imp;
end PCA_Shutoff;
```

What control action will be affected?

What specific fault will result?

What can we do with our model + specific fault information?

Fault Types

EMV2 Type Hierarchy

Error Library Type	STPA Error Type	App Error Type
Errors with Physiological Monitors		
LateDelivery	DelayedOperation	SpO2ValueLate
IncorrectValue	IncorrectInformation	SpO2ValueLow
N/A	NoInformation	NoSpO2Data
Errors with App Logic		
ServiceCommission	InnapropriateCtrlAction	InadvertentPumpNormally
ServiceOmission	MissingCtrlAction	InadvertentPumpMinimally

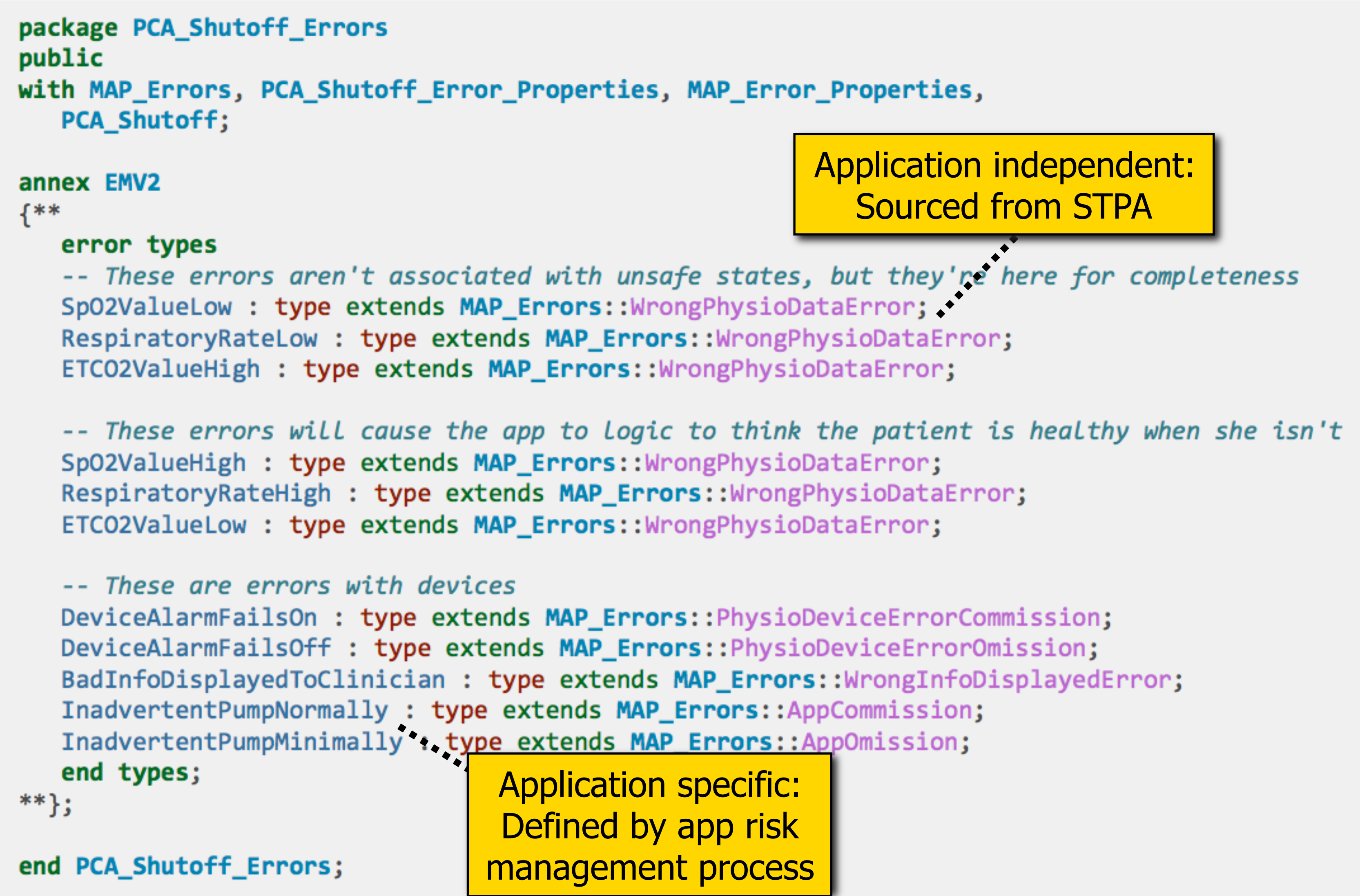
AADL Standard Error Types

STPA Error Types

App Specific Error Types

Fault Types

App Specific Error Library



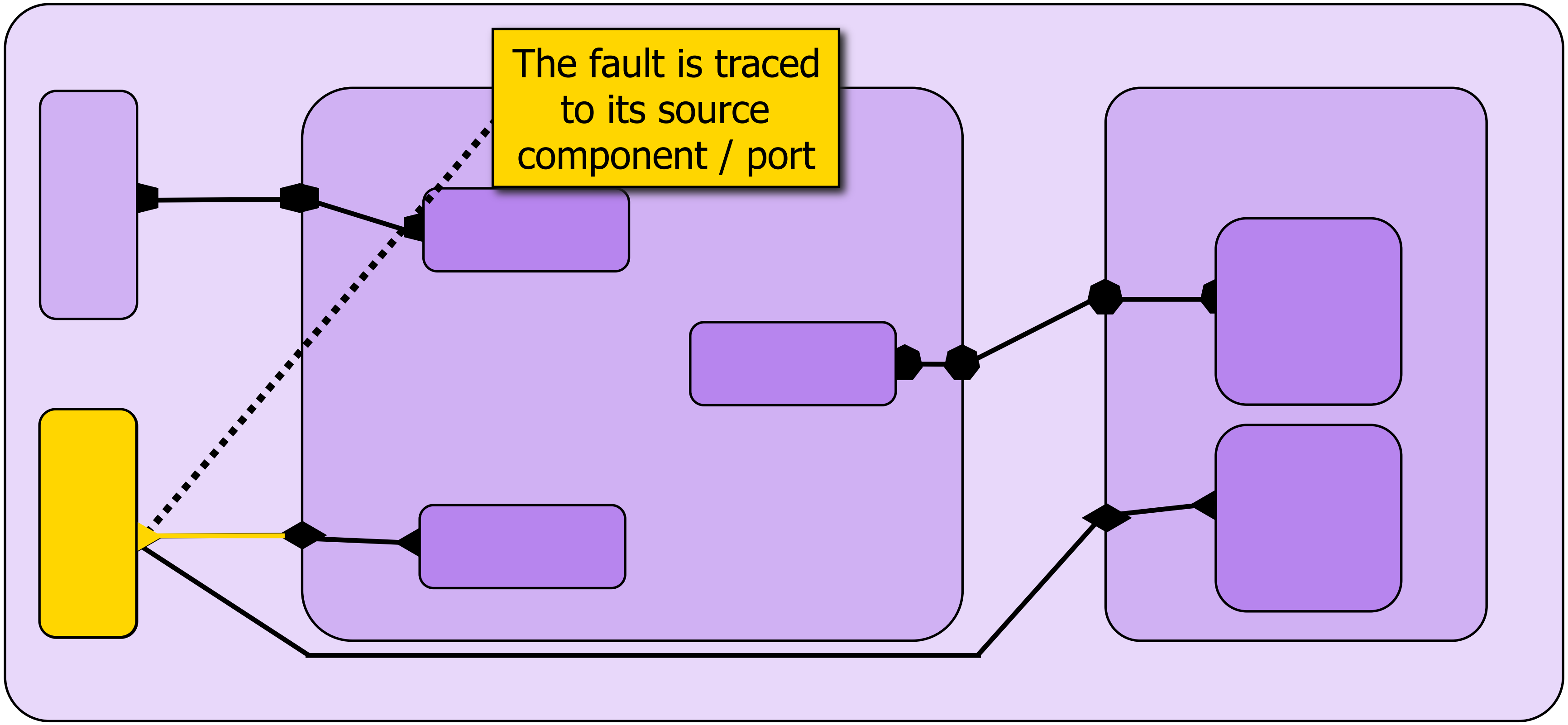
```
package PCA_Shutoff_Errors
public
with MAP_Errors, PCA_Shutoff_Error_Properties, MAP_Error_Properties,
   PCA_Shutoff;

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;
```

Application independent:
Sourced from STPA

Application specific:
Defined by app risk
management process

Hazard Analysis

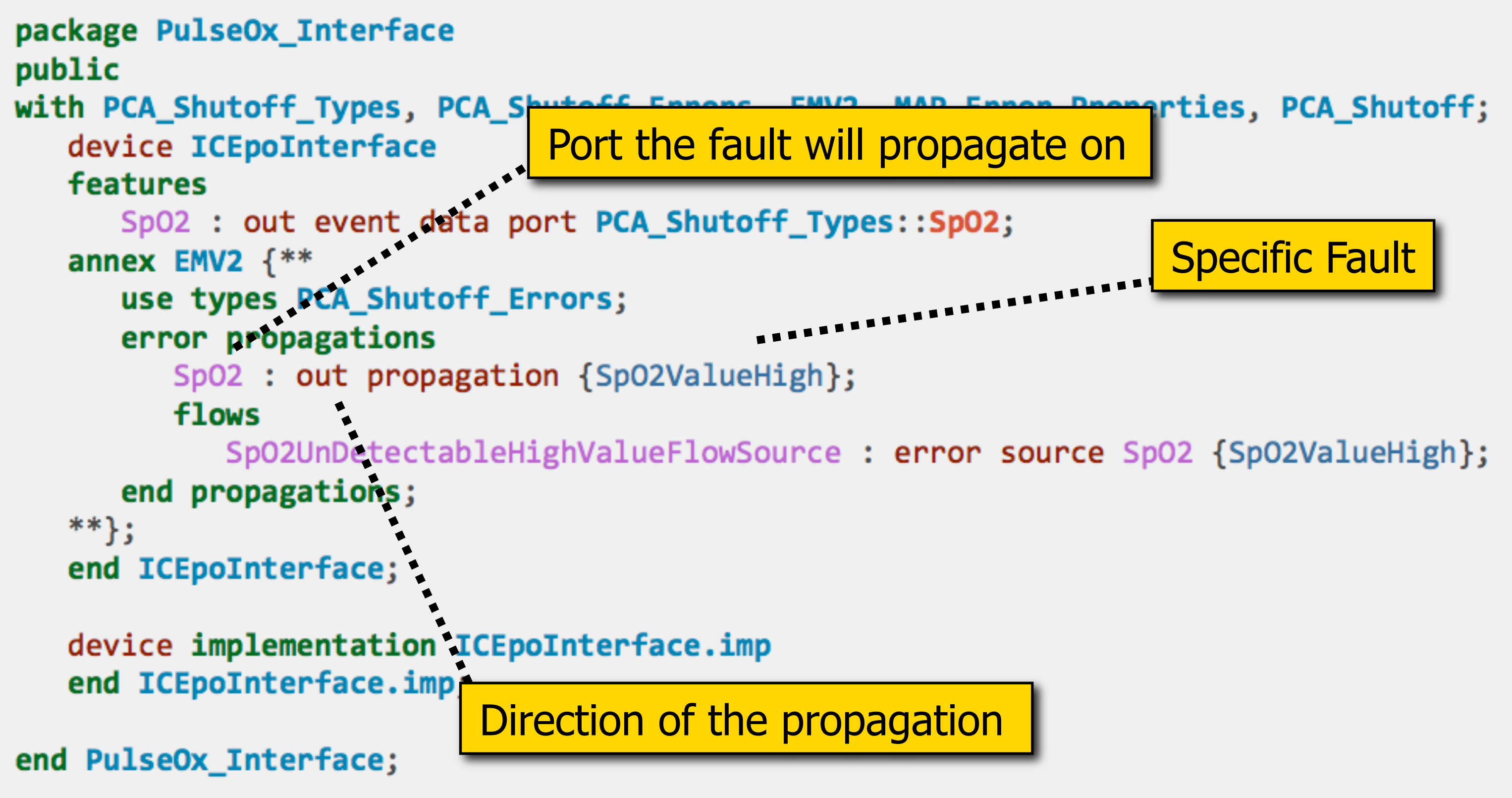
Annotating the Architectural Model



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;
```

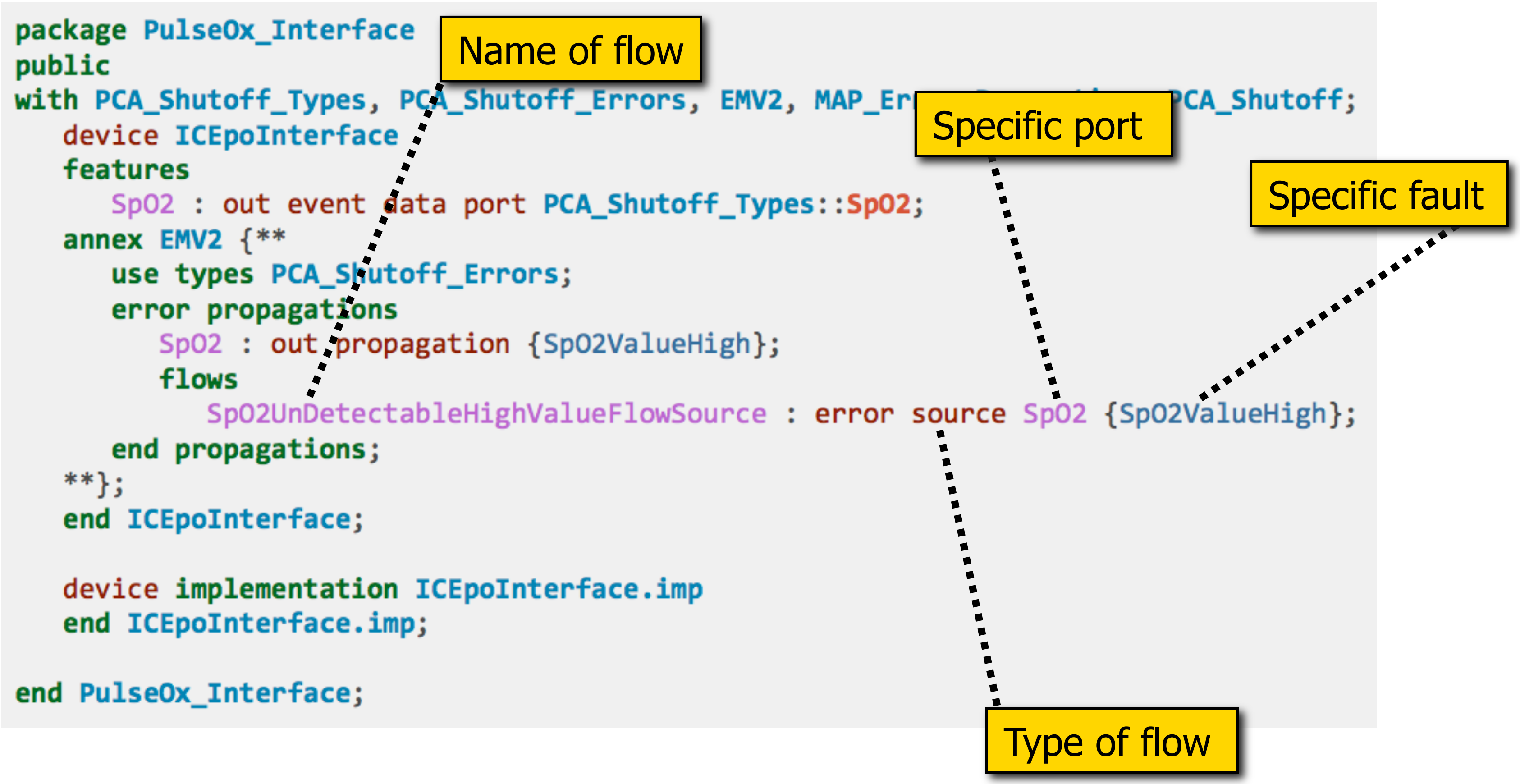
Port the fault will propagate on

Specific Fault

Direction of the propagation

Hazard Analysis

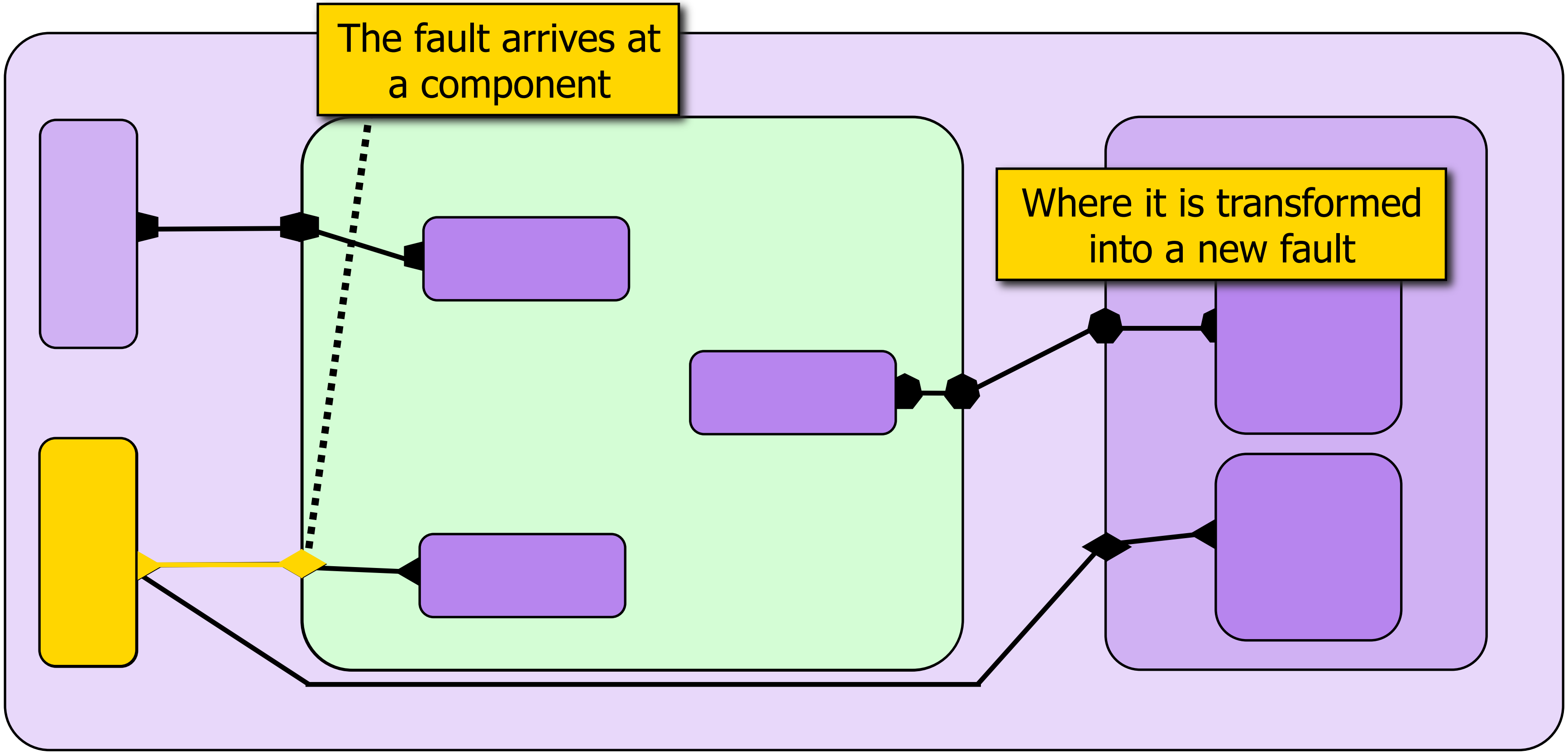
Specification Step 2: Flow



```
package PulseOx_Interface
public
with PCA_Shutoff_Types, PCA_Shutoff_Errors, EMV2, MAP_Er                 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

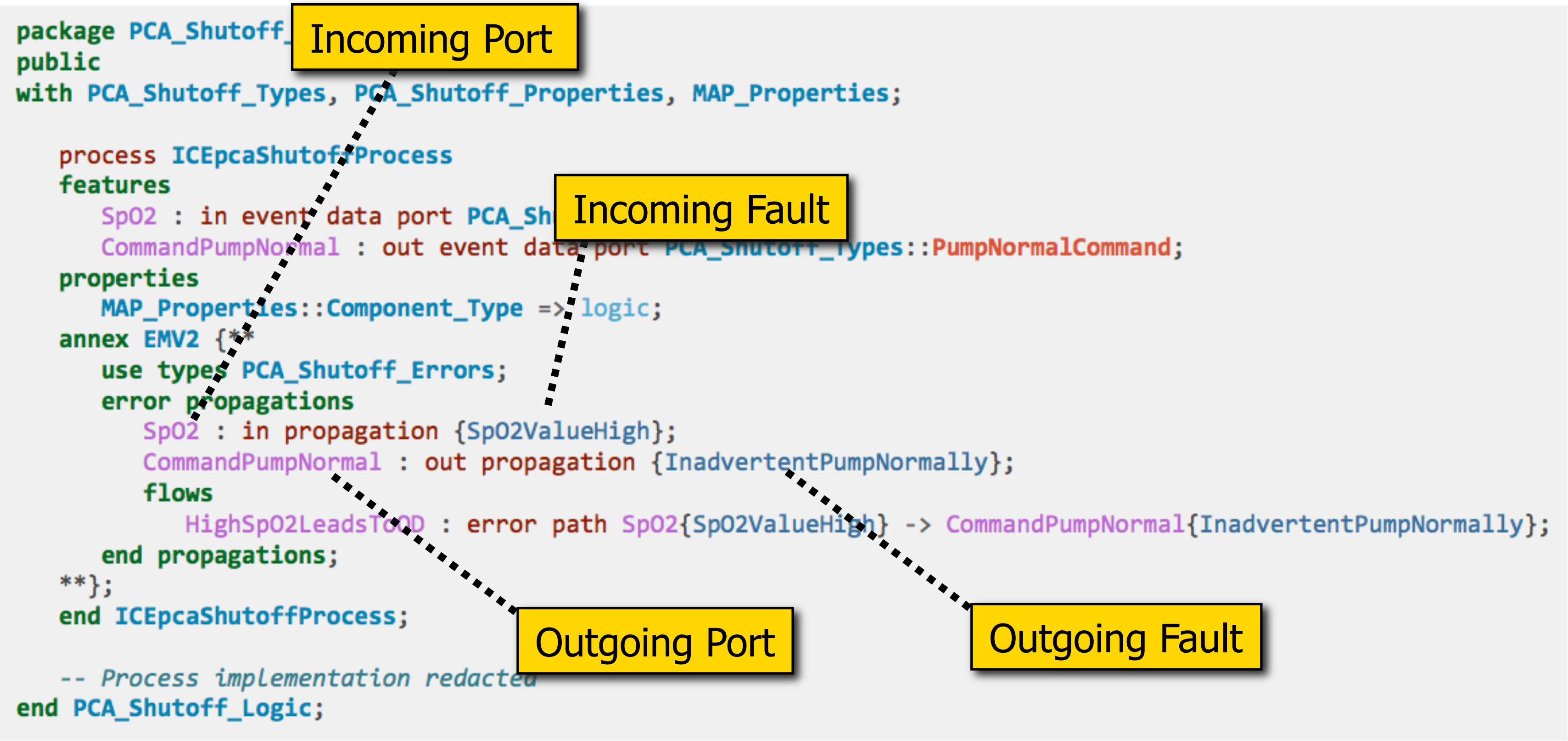


The fault arrives at a component

Where it is transformed into a new fault

Hazard Analysis

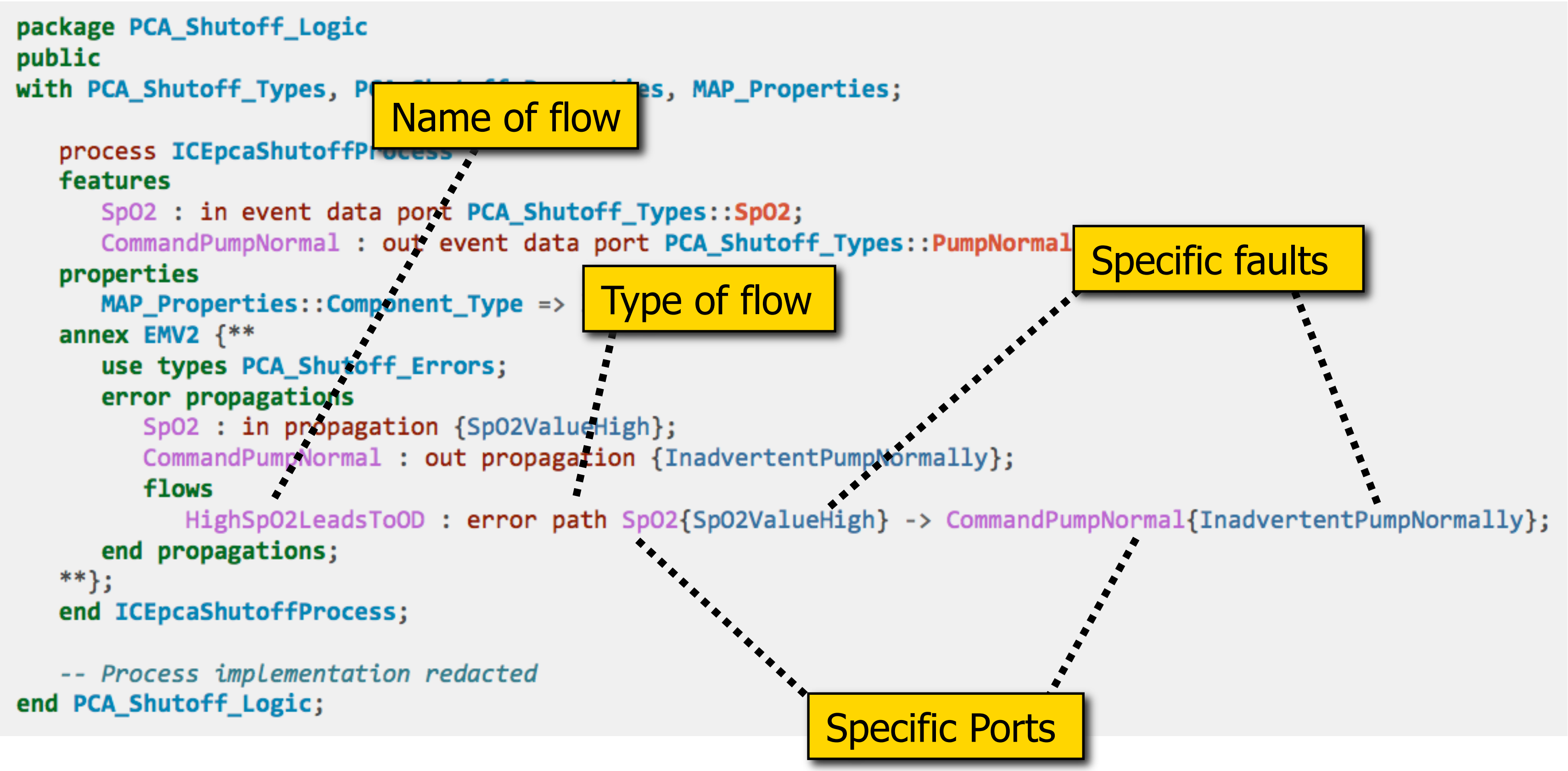
Specification Step 3: Error Transformations



```
package PCA_Shutoff_
public
with PCA_Shutoff_Types, PCA_Shutoff_Properties, MAP_Properties;

   process ICEpcaShutoffProcess
   features
      SpO2 : in event data port PCA_Sh
      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 propagations;
   **};
   end ICEpcaShutoffProcess;

   -- Process implementation redacted
end PCA_Shutoff_Logic;
```

Hazard Analysis

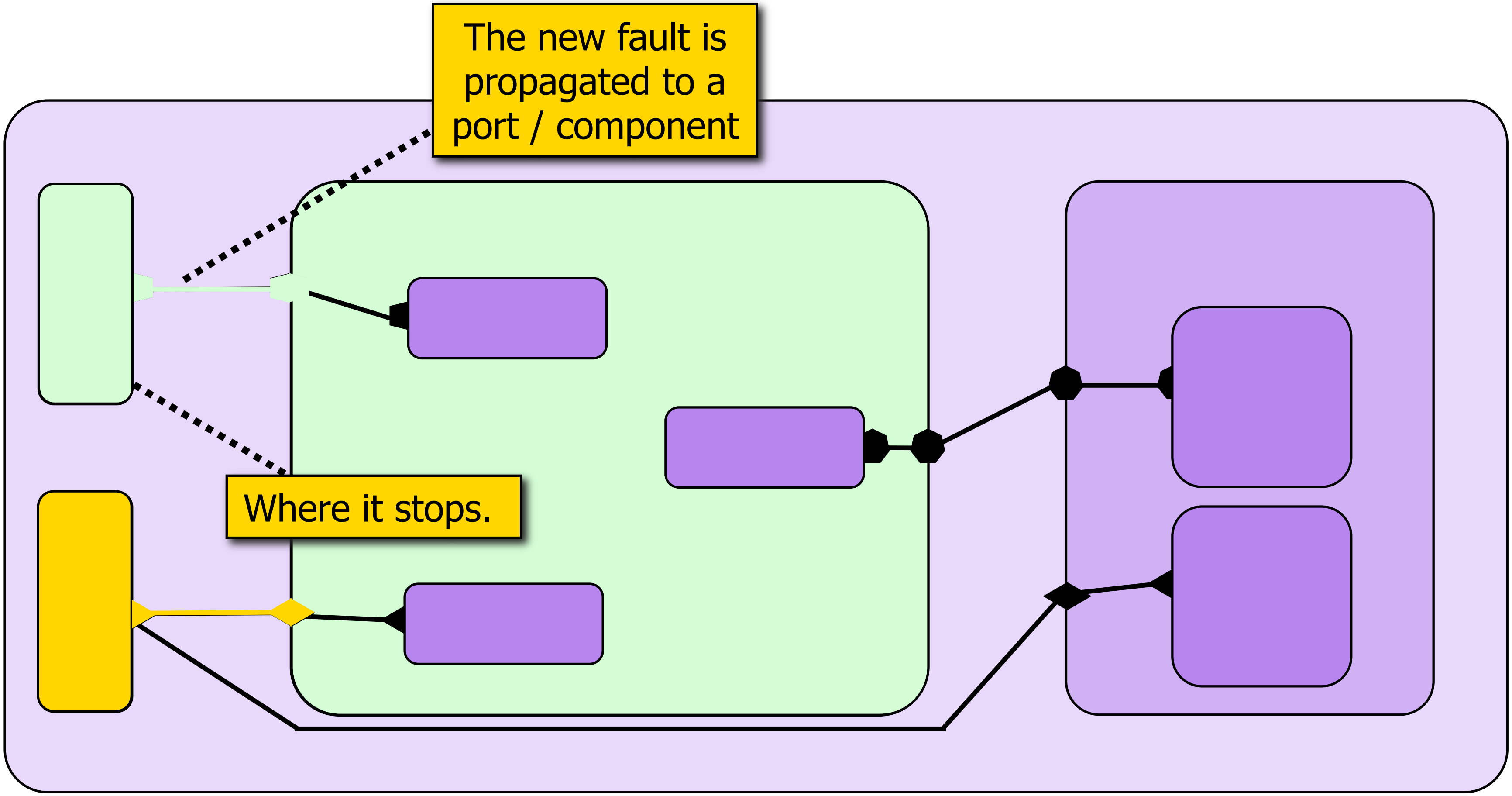
Specification Step 3: Error Transformations



```
package PCA_Shutoff_Logic
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::PumpNormal
   properties
      MAP_Properties::Component_Type =>
   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 propagations;
   **};
   end ICEpcaShutoffProcess;

   -- Process implementation redacted
end PCA_Shutoff_Logic;
```

Hazard Analysis

Error Sinks

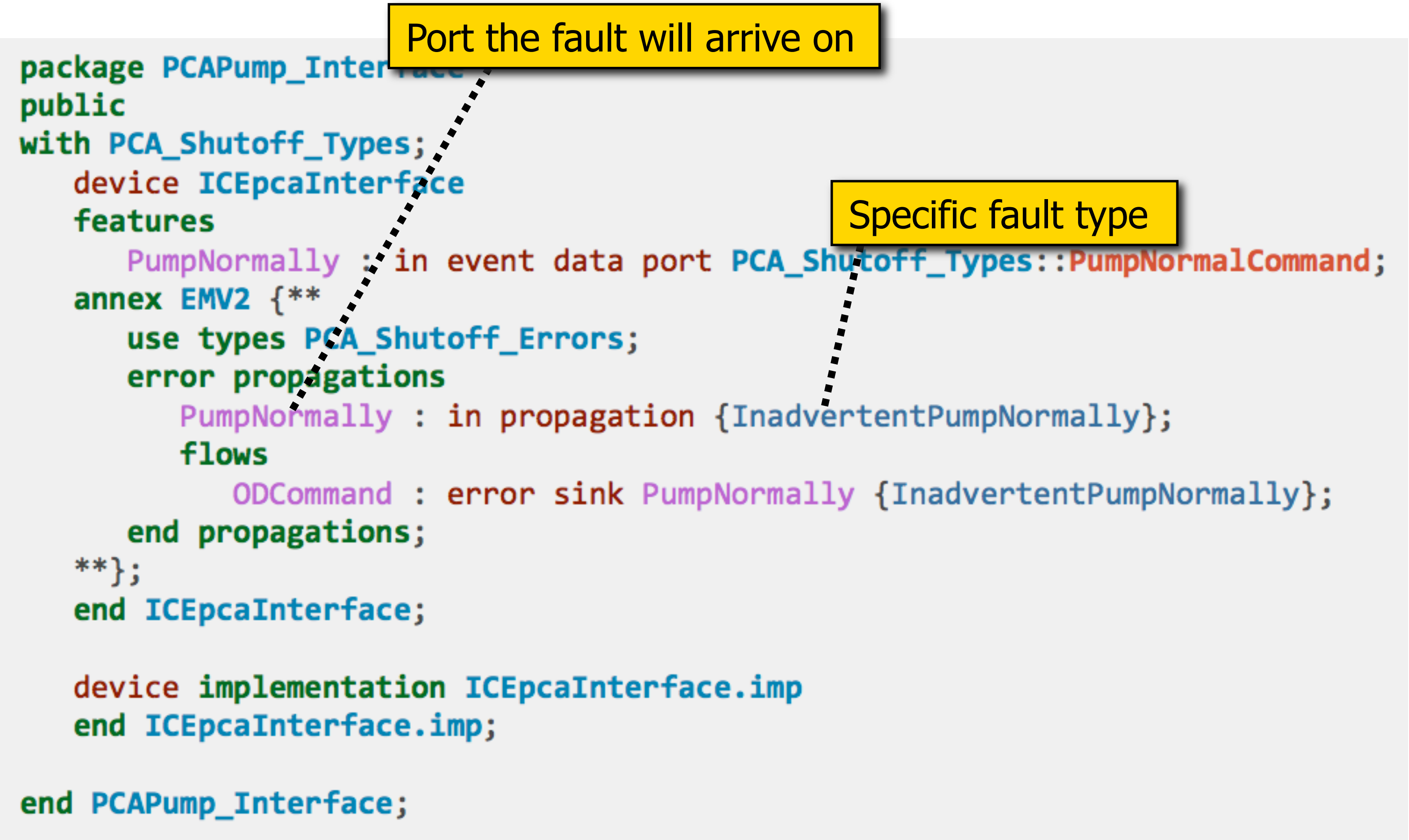


The new fault is propagated to a port / component

Where it stops.

Hazard Analysis

Specification Step 4: Error Sink



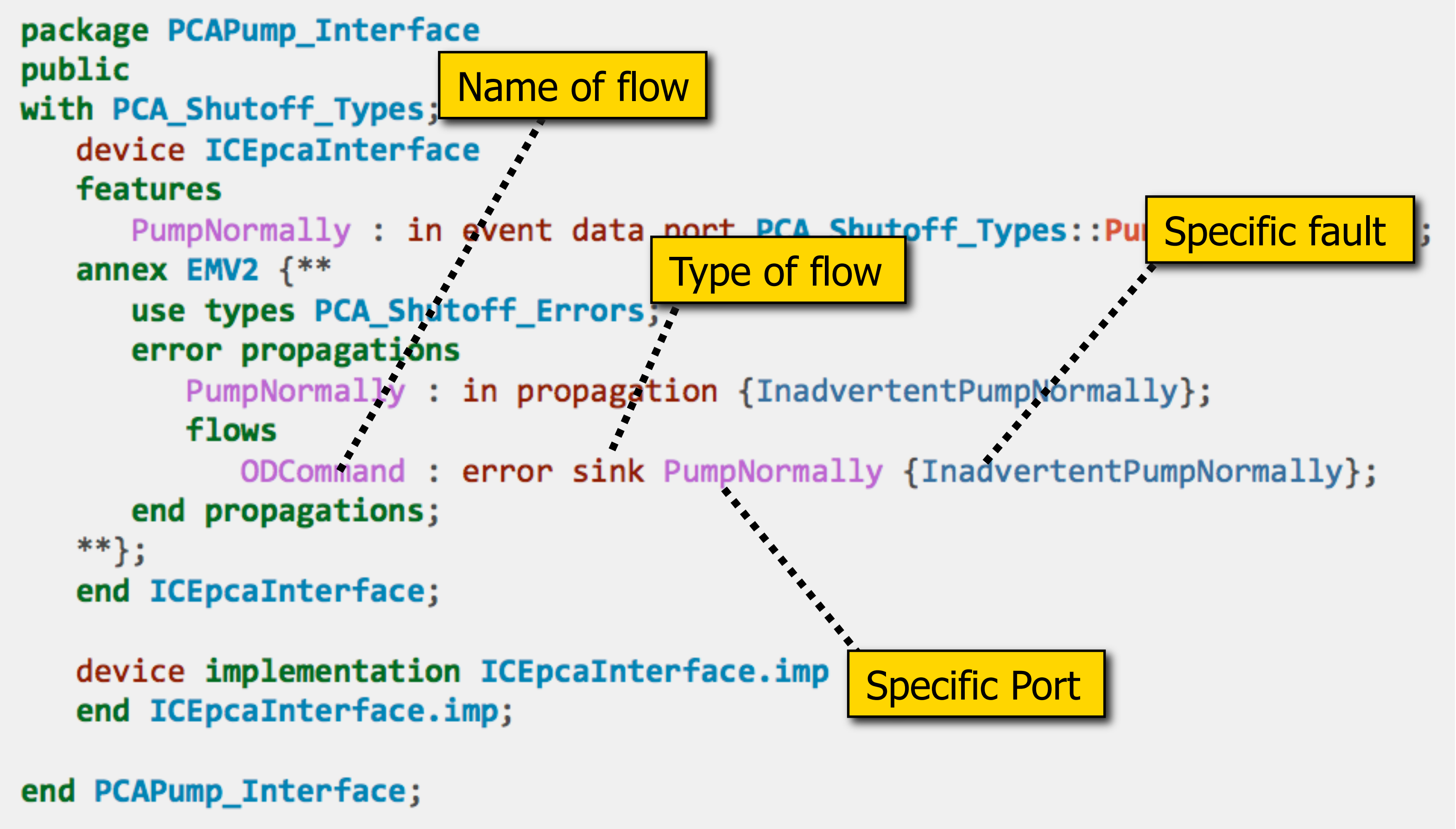
```
package PCAPump_Interface
public
with PCA_Shutoff_Types;
   device ICEpcaInterface
   features
      PumpNormally : in event data port PCA_Shutoff_Types::PumpNormalCommand;
   annex EMV2 {**
      use types PCA_Shutoff_Errors;
      error propagations
         PumpNormally : in propagation {InadvertentPumpNormally};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

   device implementation ICEpcaInterface.imp
   end ICEpcaInterface.imp;

end PCAPump_Interface;
```

Port the fault will arrive on

Specific fault type

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::Pu                 ;
   annex EMV2 {**
      use types PCA_Shutoff_Errors;
      error propagations
         PumpNormally : in propagation {InadvertentPumpNormally};
         flows
            ODCommand : error sink PumpNormally {InadvertentPumpNormally};
      end propagations;
   **};
   end ICEpcaInterface;

   device implementation ICEpcaInterface.imp
   end ICEpcaInterface.imp;

end PCAPump_Interface;
```

Name of flow

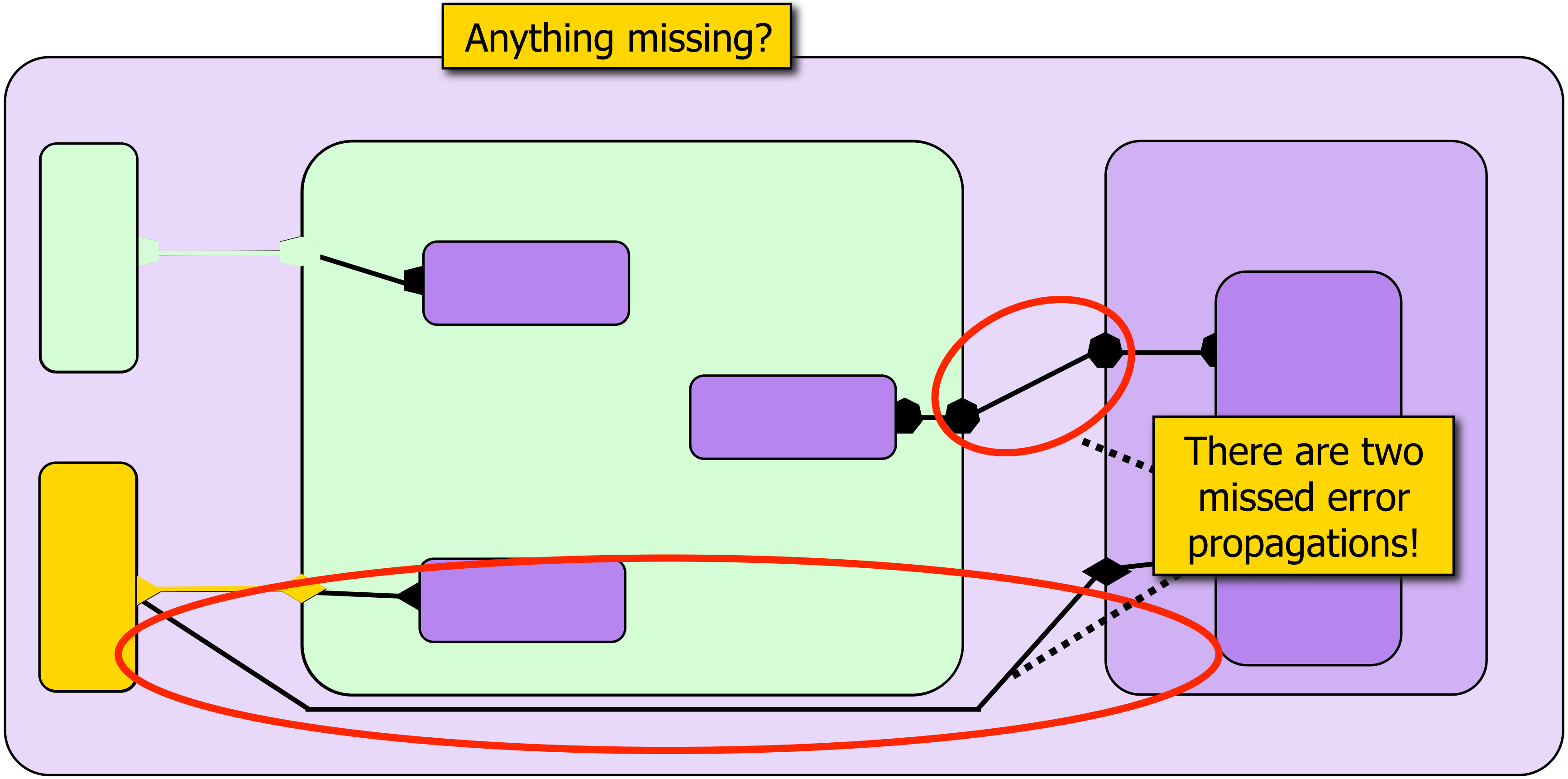
Type of flow

Specific fault

Specific Port

Hazard Analysis

OSATE Remembers a Neglected Connection

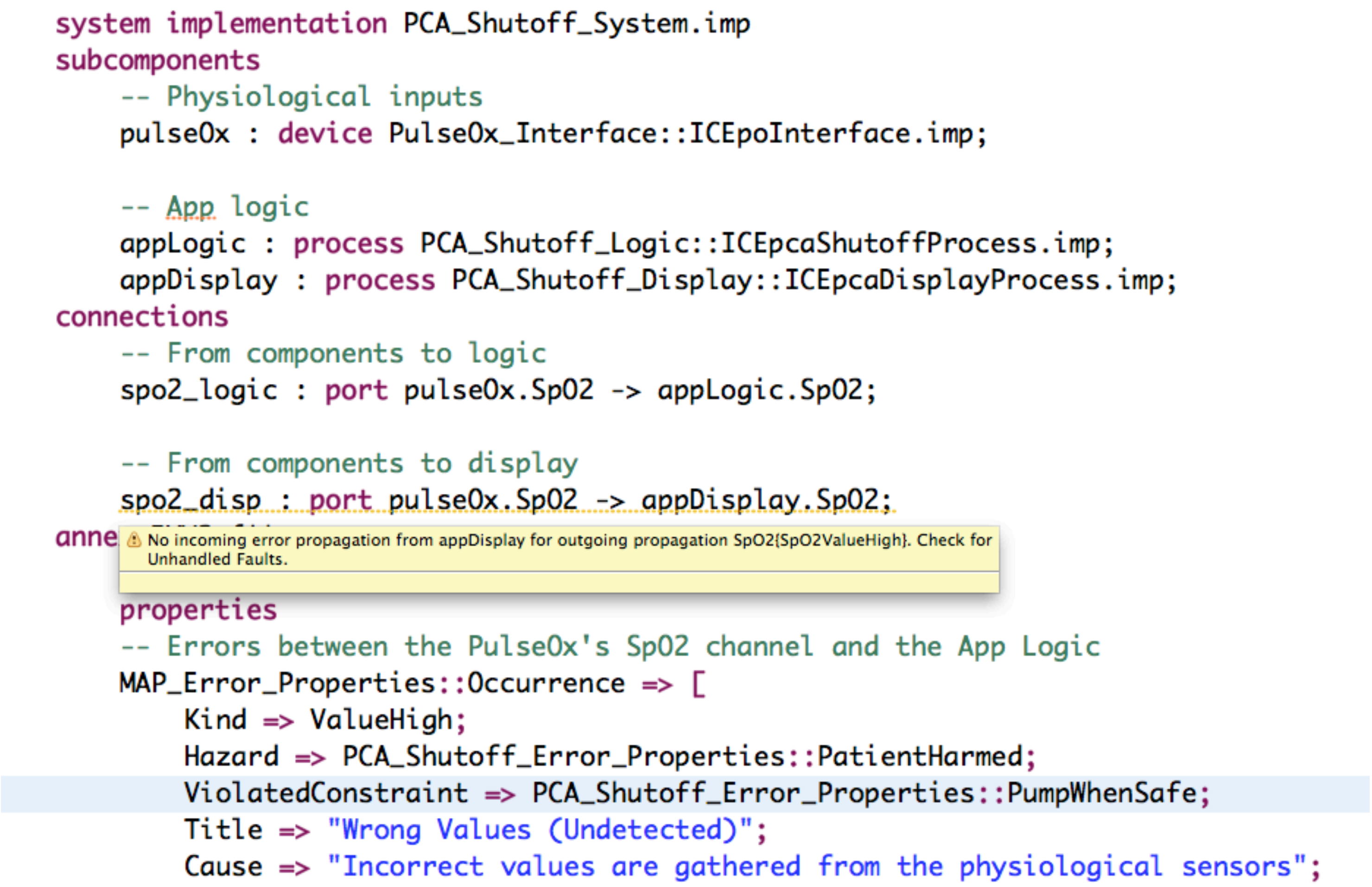


Anything missing?

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;
anne
    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";
```

⚠ No incoming error propagation from appDisplay for outgoing propagation SpO2{SpO2ValueHigh}. Check for Unhandled Faults.

Tool

OSATE2

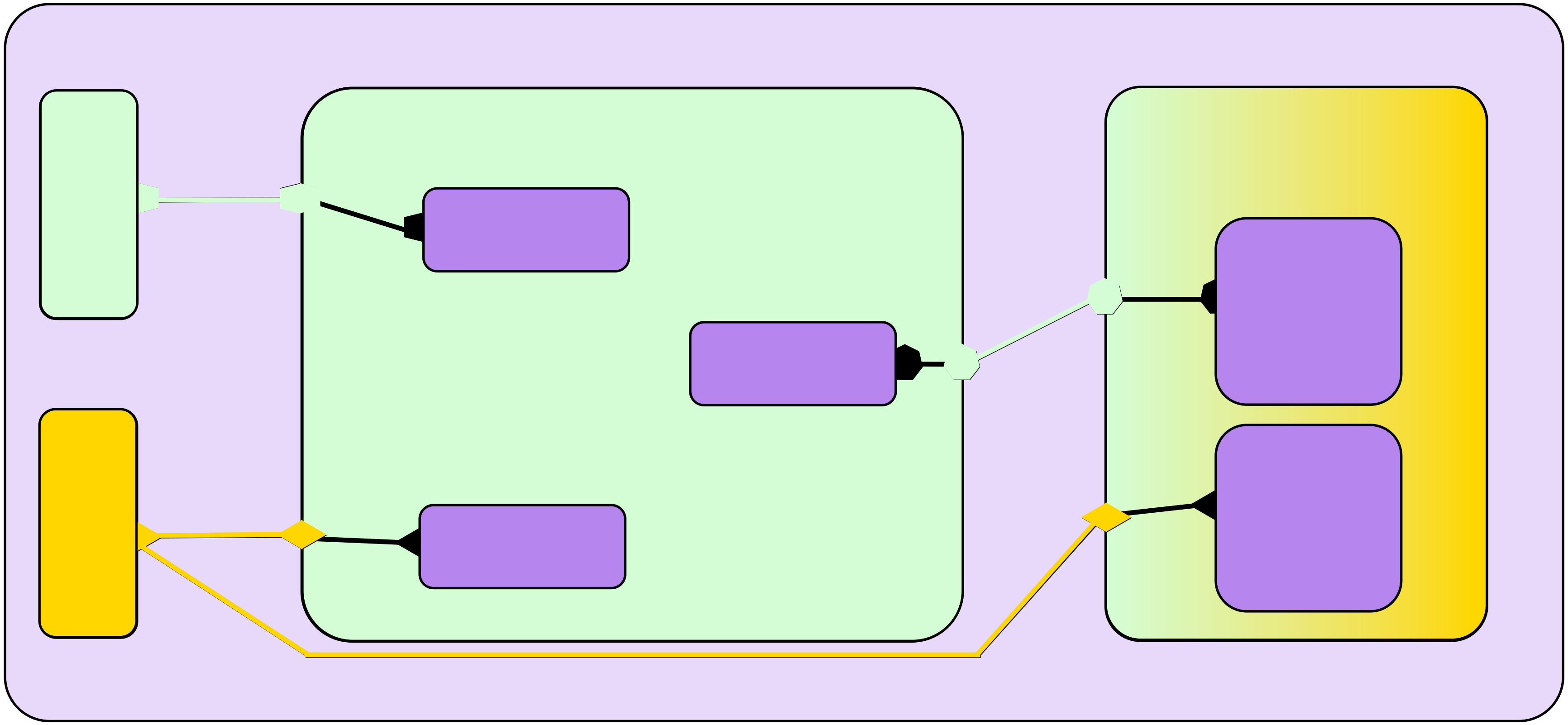
- Open-source, Eclipse-based tool
- Our work is available as a plugin
  - Uses the model-traversal built into OSATE2

Open Source AADL
Tool Environment

http://www.aadl.info
http:/www.aadl.info/wiki

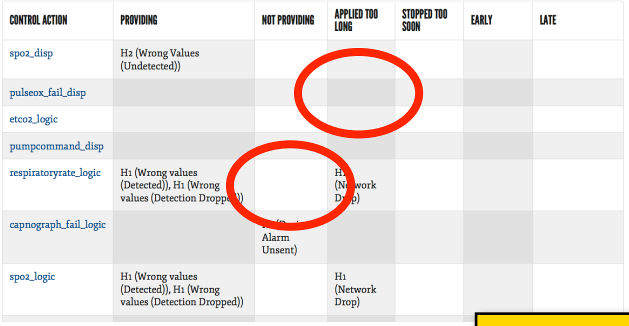
Hazard Analysis

Our model is updated accordingly



Hazard Analysis

Interaction between Report and Model

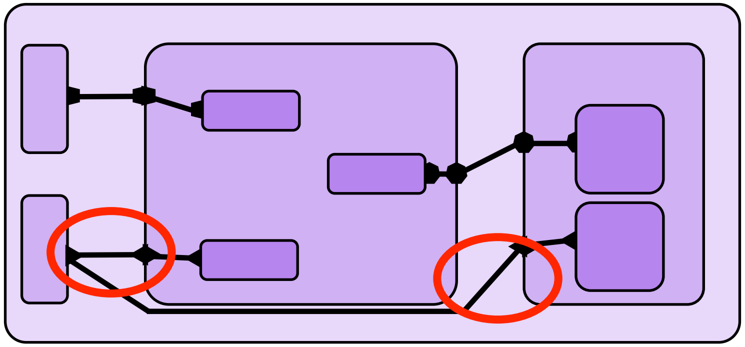


CONTROL ACTION	PROVIDING	NOT PROVIDING	APPLIED TOO LONG	STOPPED TOO SOON	EARLY	LATE
spo2_disp	H2 (Wrong Values (Undetected))					
pulseox_fail_disp						
etco2_logic						
pumpcommand_disp						
respiratoryrate_logic	H1 (Wrong values (Detected)), H1 (Wrong values (Detection Dropped))		H1 (Network Drop)			
capnograph_fail_logic		[illegible] Alarm Unsent)				
spo2_logic	H1 (Wrong values (Detected)), H1 (Wrong values (Detection Dropped))		H1 (Network Drop)			

Top Down

1. Here’s an empty cell (STPA Keyword + Control Action)… could anything go wrong?

2. Create occurrence and supporting EMV2 annotations



Bottom Up

3. Where else could this fault go?

4. What else could cause this error?

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-medical
- 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.