Information System Hazard Analysis

by

Fieran Mason-Blakley
B.Sc., University of Victoria, 2003
M.Sc., University of Victoria, 2011

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Information System Hazard Analysis

by

Fieran Mason-Blakley
B.Sc., University of Victoria, 2003
M.Sc., University of Victoria, 2011

Supervisory Committee

Dr. Jens Weber, Supervisor
(Department of Computer Science)

Dr. Morgan Price, Co-Supervisor
(Department of Computer Science)

Dr. Abdul Roudsari, Outside Member
(School of Health Information Science)

Supervisory Committee

Dr. Jens Weber, Supervisor
(Department of Computer Science)

Dr. Morgan Price, Co-Supervisor
(Department of Computer Science)

Dr. Abdul Roudsari, Outside Member
(School of Health Information Science)

ABSTRACT

We present Information System Hazard Analysis (ISHA), a novel systemic hazard analysis technique focused on Clinical Information System (CIS)s. The method is a synthesis of ideas from United States Department of Defense Standard Practice System Safety (MIL-STD-882E), System Theoretic Accidents Models and Processes (STAMP) and Functional Resonance Analysis Method (FRAM). The method was constructed to fill gaps in extant methods for hazard analysis and the specific needs of CIS. The requirements for the method were sourced from existing literature and from our experience in analysis of CIS related accidents and near misses, as well as prospective analysis of these systems. The method provides a series of iterative steps which are followed to complete the analysis. These steps include modelling phases that are based on a combination of STAMP and FRAM concepts. The method also prescribes the use of triangulation of hazard identification techniques which identify the effects of component and process failures, as well as failures of the System Under Investigation (SUI) to satisfy its safety requirements. Further to this new method, we also contribute a novel hazard analysis model for CIS as well as a safety factor taxonomy. These two artifacts can be used to support execution of the ISHA method. We verified the method composition against the identified requirements by inspection. We validated the method's feasibility through a number of case studies. Our

experience with the method, informed by extant safety literature, indicates that the method should be generalizable to information systems outside of the clinical domain with modification of the team selection phase.

Contents

> Accidents and the threat of accidents are the primary motivators for work on safety. Take accidents away and concern about safety diminishes and attention shifts towards production. Virtually all safety work takes place in the shadow of accidents and experience with accidents - both our direct experience and that which we acquire by hearing about the accidents that happen to others - shapes our general and specific approaches to safety.

–Richard Cook

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

My wife Jennifer for her support through our personal and my professional troubles. I couldn't have done this without your help. I am so proud of your for your had work in taking care of Kai, and the energy you have put into caring for the girls, and your family on top of it all.

My parents, Buffy Blakley, Neil Mason and Liz Mason who have provided so much emotional and financial support throughout this process.

Jens Weber, Morgan Price and Abdul Roudsari for mentoring, support, encouragement, and patience.

Mark Sudul and Toni Foster of Osler systems for their mentoring and the opportunities they have and continue to provide me.

Cathy McGuiness, Francis Lau, and Kevin Kotorynski for your mentoring and guidance through our various professional projects.

DEDICATION

To my mother Buffy Blakley who helped me so much through this phase of my life. To my late father John Neilson Mason whose guidance and love I miss every day. To my late Grandparents Peggy and Herb Blakley whose love and support helped me through my difficult teenage years. Finally, to my loving wife Jennifer who continues to help me through each new challenge life throws at our family.

Chapter 1

Introduction

1.1 Motivation

In 1999, the Institute of Medicine (IOM) reported that as many as 98,000 Americans were dying each year as a consequence of medical error [70]. In tandem with the communication of this finding, the IOM recommended the implementation of a range of safety critical information systems called Clinical Information Systems (CIS) to mitigate the problem.

In spite of the proliferation of this technology over the following years, the integration of these tools into health care systems did not reduce the rate of patient injury - in fact, the Agency for Healthcare Research and Quality (AHRQ) reported that "measures of patient safety ... indicate[d] not only a lack of improvement but also, in fact, a decline" [1]. The failure of this mitigation to reduce the accident rate in these safety critical information systems motivates investigation to understand why. One approach to this investigation is to apply hazard analysis to systems which use these tools.

Unfortunately, traditional hazard analysis methods are proving ineffective in complex sociotechnical systems [76, 52], one category of which are healthcare systems. It is broadly argued that this is because many traditional methods of hazard analysis have a basis in Domino Theory [45], a relatively simplistic theory of accident causation. The theory proposes that a singular event initiates a chain reaction of "falling dominoes" which leads to an accident. This theory gave rise to a series of hazard analysis techniques including Root Cause Analysis (RCA), Fault Tree Analysis (FTA) and Event Tree Analysis (ETA) [131, 28]. It is being observed however, that accidents in which there is a singular initiating event are rare, and that in fact

most accidents are more complicated.

Fortunately, there are a range of new accident frameworks which are collectively known as *systems thinking* or the *systemic approach* which specifically attempt to address accidents in complex sociotechnical systems. The works of leading [135] authors in this field including Leveson [76], Hollnagel [52] and Rasmussen [108] are being applied in a growing number of industries including military, nuclear power, transportation, chemical processes, and to a lesser extent in healthcare [104, 96, 67, 133, 66, 77, 97, 121, 147]. To date however, limited work has been done to explicitly adapt the systemic approach to CIS. CIS depend heavily on medical record software, a specialized type of computerized information system. The focus of application for the traditional hazard analysis methods has been on mechanical, electrical, and chemical systems while the systemic techniques have focused more on organizational failures.

1.2 Terminology

Our research focuses on hazard analysis for CIS, a subset of safety critical information systems. To bound this domain we specify the semantics of our terminology.

We choose the following definition for **Clinical Information System**:

> An "array or collection of applications and functionality; an amalgamation of systems, medical equipment, and technologies working together that are committed or dedicated to collecting, storing, and manipulating healthcare data and information and providing secure access to interdisciplinary clinicians navigating the continuum of client care. Designed to collect patient data in real time and to enhance care by putting data at the clinician's fingertips and enabling decision making where it needs to occur - at the bedside." [88]

We combine definitions of *information system* [27] and *safety critical* [26] with definitions for *catastrophic* [26] and *critical* [26] outcomes to arrive at the following definition of **safety critical information system**:

> "any combination of information technology and people's activities using that technology to support operations, management, and decision-making" [27] "whose mishap severity consequence could result in" [26]

1. "death, permanent total disability, irreversible significant environmental impact, or monetary loss equal to or exceeding $10M or,
2. permanent partial disability, injuries or occupational illness that may result in hospitalization of at least three [people], reversible significant environmental impact, or monetary loss equal to or exceeding $1M but less than $10M." [26]

The key functions of safety critical information systems are supported by **safety-critical software** which we choose to define as software

1. "whose inadvertent response to stimuli, failure to respond when required, response out-of-sequence, or response in combination with other responses can result in an accident." [56]
2. "that is intended to mitigate the result of an accident." [56] or
3. "that is intended to recover from the result of an accident." [56]

We define **accident** and its synonym **mishap** as:

> "[a]n unplanned event or series of events that results in death, injury, illness, environmental damage, or damage to or loss of equipment or property." [56]

We define a **hazard** to be

"[a] real or potential condition [in the system] that could [contribute] to an unplanned event or series of events (i.e. mishap) resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment." [26]

This differs from Leveson's definition of hazard which is described in terms of interactions between the system and its environment: "A hazard is a state or set of conditions of a system (or an object) that, together with other conditions in the environment of the system (or object), will lead inevitably to an accident (loss event)" [74]. The consequence of our choice is that the system boundaries in Information System Hazard Analysis (ISHA) analyses are inclusive of significant environmental concerns - that is, we model systems as closed as opposed open as would a method which used Leveson's definition.

The literature often refers to hazards and contributing factors and intimates that hazards on their own have the potential to *lead* to an accident while contributing factors can increase the likelihood that a hazard will lead to an accident, or may act in concert in the absence of a core hazard to lead to an accident. Due to this vagueness in language used in literature, we instead avoid the distinction.

1.3 Existing Techniques and Methods

There are a wide range of existing traditional hazard analysis techniques and methods. Both Ericson [28] and Stamatis [131] provide comprehensive overviews of some of the most popular of these. We provide our own brief summary of traditional approaches covering FTA, ETA, Failure Mode and Effects Analysis (FMEA), and HAZard OPerability (HAZOP). Beyond these traditional methods, we will also summarize the two most popular [135] systemic methods: the Functional Resonance Analysis Method (FRAM) [52] and System Theoretic Accidents Models and Processes (STAMP) [76]. We choose this subset of traditional and systemic methods based on their popularity and their influence on our primary contributions which we discuss later in this chapter.

1.3.1 Traditional Methods

FTA [28, 131] and ETA [28, 131] are decision tree based techniques which are grounded in Baysian statistics. These methods deduce from a sequence of events what the probability of a top level event might be. FMEA [130] is a reliability analysis method which considers the modes by which failure might occur, and the effects that failure might have. HAZOP [69] is an interaction analysis technique which identifies potential deviations of material, energy or information flow in a system from design intent and then tracks the consequences of those deviations.

1.3.2 Systemic Methods

Functional Resonance Analysis Method

FRAM [52] is a systemic method that considers accidents as events which arise from the resonance of the variability in the behaviour of a system. Variability in performance in FRAM is viewed not only as a potential source of accidents, but also as a

source of system resiliency. Though we agree that accidents can arise in this fashion, we also observe that an analysis which only considered variance in "normal operation" as described by Hollnagel, would not identify hazards which arose from exceptional events like major system component failures.

Analysts using FRAM, model systems as a set of primary system functions. Each function is characterized by five inputs - the input which starts/is transformed or processed, the preconditions, the controls, the resources which are consumed/are needed, and timing elements. Each function is also characterized by an output. Each input is fed by either the output of a previous function, or in the case of an open system - from an outside source. An atomic FRAM function is modelled in Fig. 1.1.

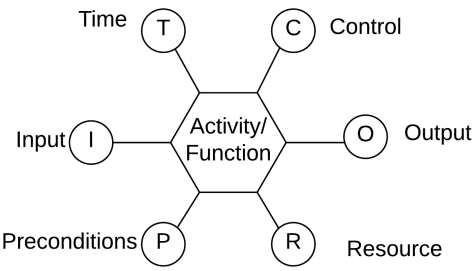

Figure 1.1: A FRAM function - replicated from [52].

Once a model of the system has been constructed, the variability in the system is assessed. The variability of the system arises from the variability of its component functions. Each component function has two types of variability, internal and external. Internal variability is the variability within the function which is independent of the rest of the system. External variability is the variability for the function which is induced by its dependencies on upstream functions. The variability of the system emerges from the aggregation of the variability of its component functions. Analysts use the results of the variability investigation to either redesign functions to reduce their internal variance, or design controls which monitor, dampen or magnify potential variability resonance to optimize the systems' performance against both production and safety goals.

System Theoretic Accidents Models and Processes

In STAMP [76], analysts take the position that accidents arise from the failure of system controllers to enforce constraints necessary for safety. These safety constraints can revolve around a wide variety of system attributes and behaviours including integrity degradation, and communication delays.

STAMP analysts model systems as webs of control loops consisting of components

which are stereotyped as *controllers*, *sensors*, *actuators*, or *processes*. In a STAMP model, a *controller* observes a *process* via a *sensor*. It deliberates on a course of action, and then guides the trajectory of the *process* it controls by means of an *actuator*. In these models, only *controllers* are considered rational. They are thus imbued with a "brain" which is referred to as a *process model*. In order for a STAMP system to be safe, it must observe a series of *safety constraints*. When *safety constraints* are not enforced, the system is subject to the risk of an accident. The safety constraints may restrict the attributes of the components, or the communications between the components. An atomic STAMP control loop which excludes safety constraints and the controller's process model is provided in Fig. 1.2.

In the analysis phase of STAMP a combination of a HAZOP based guide word strategy and consideration of safety constraints is used to identify hazards in the System Under Investigation (SUI).

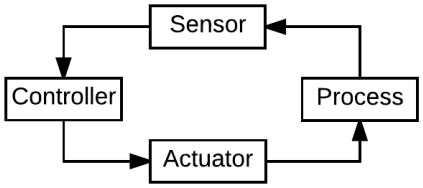

Figure 1.2: An atomic STAMP control loop adapted from [76].

1.4 Limitations of Existing Approaches

We discuss two types of limitations for the existing methods. Internal limitations are those that either compromise analysis results, or which demonstrate constraints on the extent to which they can accomplish the goal of hazard analysis. External limitations highlight mismatches between the structure of the outputs of these methods and the demands of the processes which consume those outputs.

1.4.1 Limitations of Traditional Techniques

Only a single traditional technique was identified that specifically addresses the needs of CIS. This method [24] is a healthcare specialization of FMEA - Health Care Failure Mode and Effects Analysis (HFMEA), but is compromised by the same shortcomings as its parent method which will be discussed below. Traditional hazard analysis techniques including the tree based methods, FMEA, and HAZOP, are too simplistic and limited as each is based on many of the following, and most importantly the first, invalid assumptions [52, 76]:

- System components and functions will either work or fail.
- System events occur in a predictable and sequential fashion.
- System outputs can be described by logical operators and are predictably proportional to system inputs.
- Reliability and safety are equivalent (Section 1.4.1 - A Word on Safety and Reliability).
- Accidents arise from an initiating event followed by a series of tightly coupled "Domino" effects.
- Operators are the primary holders of fault for accidents.
- Assignment of blame is a necessary outcome of accident analysis.

Further, though these methods have been, and continue to be used to demonstrate the safety of software intensive products including medical devices, their outputs are not structured in such a way as to effectively argue system safety without substantial packaging efforts (e.g., into an assurance case [Section 2.1]).

A Word on Safety and Reliability

One of the primary flaws of many traditional, rather than systemic, hazard analysis techniques is that they are grounded in the assumption that reliability and safety are either synonymous or at least closely related. The misconception may have arisen from the expectation that the lack of a failure in individual components would result in a system which did not suffer accidents. This linear and causal perspective has its foundations in Domino Theory [45]. Safety and reliability are in fact *different* system properties. We define **safety** as

> "[f]reedom from [system-induced] conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment." [26]

We adapt the original definition to better suit our domain of application because otherwise, healthcare would be unsafe by definition. We define **reliability** as

> "[t]he ability of a system or component to perform its required functions under stated conditions for a specified period of time." [107]

1.4.2 Limitations of Systemic Methods

Though systemic methods are being applied in healthcare - and occasionally in CIS, a number of limitations constrain their utility.

- They can be time consuming relative to their traditional predecessors [48].
- They are strictly qualitative in their current form [52, 76, 108].
- They require imagination and expertise to execute effectively [52, 76, 108].
- Their support of abstraction is less explicit than may be desirable. This can lead to model diagrams which are large and visually complex with many crossing lines representing function interactions [52, 76, 108].
- None of the systemic methods nor their specializations focus specifically on CIS.

1.4.3 Need for Clinical Information System Hazard Analysis

Unintended consequences of CIS implementation have been widely reported to be of significant concern with respect to safety. Shifting responsibilities between members of the care team, and changes in the distributed cognition of the team between its members and the technology they use to provide care is a fundamental attribute of CIS related hazards [43, 6, 71]. This feature of CIS demands that a systemic approach to hazard analysis be taken which considers the breadth of physical computing resources, documented patient and decision support information, Human Computer Interaction (HCI), people and their training, workflow and communication, organizational structure, regulatory and market environment and finally system measurement, monitoring and control [125].

1.5 Problem Definition

- Existing hazard analysis techniques are not specialized to address the unique needs of CIS including its sociotechnical nature.
- Existing systemic hazard analysis methods provide sparse guidance on abstraction in their modelling processes.

- No single systemic safety analysis method centrally addresses both functional resonance and the transition of responsibilities between system actors over time. Consideration of functional resonance provides a perspective on the emergent nature of system safety which is not achieved with traditional root cause based analysis methods including FTA and ETA.
- Existing methods do not provide guidance on how to structure deliverables to meet the demands of the processes which consume them (e.g., certification).

1.6 Research Goals

The goal of this research is to develop a hazard analysis method for CIS which can be used for prospective incident mitigation. In order to satisfy this goal, the developed method must:

1. provide guidance on how to construct consumable system models which are a prerequisite of hazard analysis.
2. support analysts in systematic evaluation of the breadth of latent technology interaction hazards which are present in CIS.
3. support analysts in the systematic evaluation of the role of human error in the creation of hazards, and in accident causation in CIS.
4. support analysts in systematic evaluation of the hazards posed by the transitions of function responsibility which occur in CIS - between human actors, between machine actors and between human and machine actors.
5. output a compelling argument about the systems' safety or recommendations to mitigate identified and prioritized hazards.

1.7 Research Methods

Research Goal 1 is achieved by developing a relational framework for the generation of models which are grounded in the synthesis of the fundamental concepts developed in the STAMP [76] and FRAM [52] frameworks. Research Goals 2, 3, and 4 are achieved through a combination of method synthesis between the STAMP and FRAM

safety analysis methods, including the modelling framework constructed for Research Goal 1. Finally, Research Goal 5 is achieved by linking the relevant hazard analysis stages of our new method with an assurance case generation process [63, 62] and supplementing the overall process with additional necessary packaging activities to complete the assurance case output.

1.8 Contributions

We provide a number of primary contributions:

1. ISHA, a new method for hazards analysis in CIS
2. A series of artifacts to support its application including
   (a) A formal information model
   (b) A series of design patterns which can be used in the context of the information model
   (c) A taxonomy of hazard factors
   (d) A medical information system model constructed to support the application of the ISHA method on CIS.
3. A series of case studies which demonstrate the feasibility of our new method.
4. A systematic review of literature and incident reports which classify identified hazards against an a priori hazard model which is based in the Leveson's systemic STAMP framework.

1.8.1 Information System Hazard Analysis

ISHA is a systemic hazard analysis method for safety critical healthcare information systems referred to as CIS. The method synthesizes and adapts a combination of traditional hazard analysis approaches including ETA, and FMEA with two systemic methods - STAMP and FRAM. Additionally, it provides guidance on how to transform the data generated in the analysis into an assurance case which argues the relative safety of the SUI and provides prioritized mitigation recommendations for hazards which are relegated to residual risk.

1.9 Evaluation

1.9.1 Information System Hazard Analysis

The need for and validity of the ISHA method are argued based on the research gap expressed in general and CIS safety literature. ISHA is verified by addressing this gap with a series of requirements, and then demonstrating that the method meets those requirements. The further requirement that the output of the process be packaged as an argument about the safety of the SUI or recommendations on how to mitigate prioritized hazards is verified by inspection.
The expressiveness of ISHA's modelling language is validated using a grounded theory approach [132] by way of the review of literature and incident reports in:

1. A pair of systematic reviews of CIS literature
2. An analysis of incident reports related to Electronic Medical Record (EMR)s
3. A pair of running examples of the method's application in diabetes management in both inpatient and outpatient settings, once against a Computerized Provider Order Entry (CPOE) system and once against a Generalized Insulin Infusion Pump (GIIP).

The feasibility of ISHA is validated with

1. The synthesis of a Preliminary Hazard List (PHL) for the software of a GIIP
2. An analysis of the safety of an electronic document exchange standard
3. An analysis of the safety of two prescribing interfaces in a commercial EMR
4. The generation of two assurance cases

The recommendations phase of ISHA is evaluated in a mitigation study

- Evaluating search strategies to prevent misidentification errors in CIS

1.10 Organization of Dissertation

In Chapter 2 we will discuss hazard analysis methods and assurance cases. In Chapter 3 we begin introducing the ISHA method with a running example which demonstrates the early steps of its processes using CPOE functionality in an EMR. In Chapter 4, we continue our introduction of ISHA with elaboration of the later steps of the method in a second running example which describes the application of the method on a GIIP which we model as a CIS. In Chapter 5 we provide a formal information model for representing the ISHA concepts. Alongside this information model we also provide a series of design patterns to assist in its application. In Chapter 6 we validate requirements for ISHA through review of the relevant literature and by argumentation. In Chapter 7, we qualify our contribution through a discussion of limitations and generalizability. Finally we conclude in Chapter 8 with a summary of our contributions, conclusions and future work.

Chapter 2

Background

In Chapter 2, we introduce assurance cases and also provide a detailed introduction to the hazard analysis methods which were most influential in the construction of Information System Hazard Analysis (ISHA). Assurance cases are central to ISHA as the purpose of the method is to develop an argument about the safety of a System Under Investigation (SUI) or at least an argument for recommendations on how to improve the safety of an SUI. This requires the development safety claims, the development of an argument structure, and finally the identification of evidence to support the safety claims in the context of the argument structure. In discussing the evidence generation stage of the assurance case development, we turn our attention to hazard analysis as that is the mode by which evidence is identified when ISHA is applied. In this section of the chapter we introduce traditional hazard analysis methods including Fault Tree Analysis (FTA), Event Chain Analysis (ECA) and Failure Mode and Effects Analysis (FMEA). We also introduce HAZard OPerability (HAZOP) which is an intermediate method for hazard analysis that sits between the traditional methods and the systemic methods. Finally, we introduce two systemic frameworks, System Theoretic Accidents Models and Processes (STAMP) and the Functional Resonance Analysis Method (FRAM), and their related hazard analysis methods.

2.1 What is an Assurance Case?

The FDA describes an assurance case by writing that it "consists of a structured argument, supported by a body of valid scientific evidence that provides an organized case that the [SUI] adequately addresses hazards associated with its intended use within its environment of use. The argument should be commensurate with the

potential risk posed by the [SUI], the complexity of the [SUI], and the familiarity with the identified risks and mitigation measures." [35] The more broad consensus is that this definition describes a safety case. The term assurance case is more broad and can use the same principles of goals, arguments and evidence to demonstrate any choice of system property.

An assurance case consists of

1. **The Claims**: "Statement[s] about a property of the system or some subsystem.
2. **The Argument**: Links the evidence to the claim. Arguments can be deterministic, probabilistic, or qualitative. The argument describes what is being proved or established, identif[ies] the items of evidence you are appealing to, and the reasoning (inference, rationale) that the evidence is adequate to satisfy the claim. Arguments may also introduce sub-claims or assumptions which require further exposition...
3. **The Evidence**: Information that demonstrates the validity of the argument. This can include facts (e.g., based on observations or established scientific principles), analysis, research conclusions, test data, and expert opinions." [35]

2.2 Why do We Need Assurance Cases?

Traditionally, in manufacturing industries, certification has been granted based on data about the product to be certified. This product focus has provided some degree of certainty that the product in question would be safe in its intended operational environment(s). Software is different in that, so far, instead of relying on evidence about the product, legislators have instead relied on certifications of the processes used to generate the products in question. This series of decisions has been made based on the argument that the correctness of software is too difficult to guarantee. This position is fallacious.

Firstly, there are methods of mathematical proof that can, in a subset of cases, guarantee correctness. This set of tools is referred to as formal methods. Secondly, it does not follow that if good process is followed, unit testing for example, then the output will be of high quality - for example, complete path testing is infeasible, and so there are always edge cases which are not covered by testing [94]. The problem with certifying processes is that it provides no guarantees about the products those

processes produce. It only provides circumstantial evidence indicating that it is less likely that those products are of low quality relative to products which were produced outside of such a process [139].

Currently in the US medical devices industry, the Food and Drug Administration (FDA) demands conformance to the Common Good Manufacturing Guidelines (CGMP) [30]. This is relevant for Clinical Information System (CIS) software as the FDA classifies these tools as medical devices [122, 148, 10, 123]. This high level guidance however, is interpreted to demand conformance to a number of International Standards Organization (ISO) certifications and other standards. The FDA's approach does not clearly specify which artifacts will be verified or validated, much less which specific attributes of those artifacts.

2.3 What do Assurance Cases Look Like?

Confusion on the topic of safety assurance cases has been encountered since the FDA suggested their use in guidance for pre-market submissions for infusion pumps [36]. "Questions range from 'What kind of argumentation structures should we use?' and 'What constitutes acceptable evidence?' to 'Where should we start?' and 'How deep should we go in our decomposition of claims to sub-claims?'" [109] However, assurance case generation need not be difficult. Many of the arguments necessary to demonstrate product safety are already made implicitly in the documentation currently submitted for product certification. The structure of these arguments is typically hierarchical. A top level safety goal is asserted be met. The top level claim is then decomposed into sub claims. A SysML Block Definition Diagram (BDD) modelling this structure is synthesized from Ray's work [109] in Fig. 2.1. Each of the higher level claims is, arguably, strictly composed of its sub claims, or is defended based on evidence. The sub claims are a minimum and spanning set of claims necessary to support the parent claim.

2.3.1 How Can We Express Assurance Cases?

Goal Structuring Notation (GSN) is used by a number of assurance case authors [8, 109, 18, 23]; however, alternative graphical notations including Wigmore Charts, Toulmin Diagrams and Claims Argument Evidence Trees [129] can also be used. Further, a text based notation is offered by Holloway [53]. As our preference is for

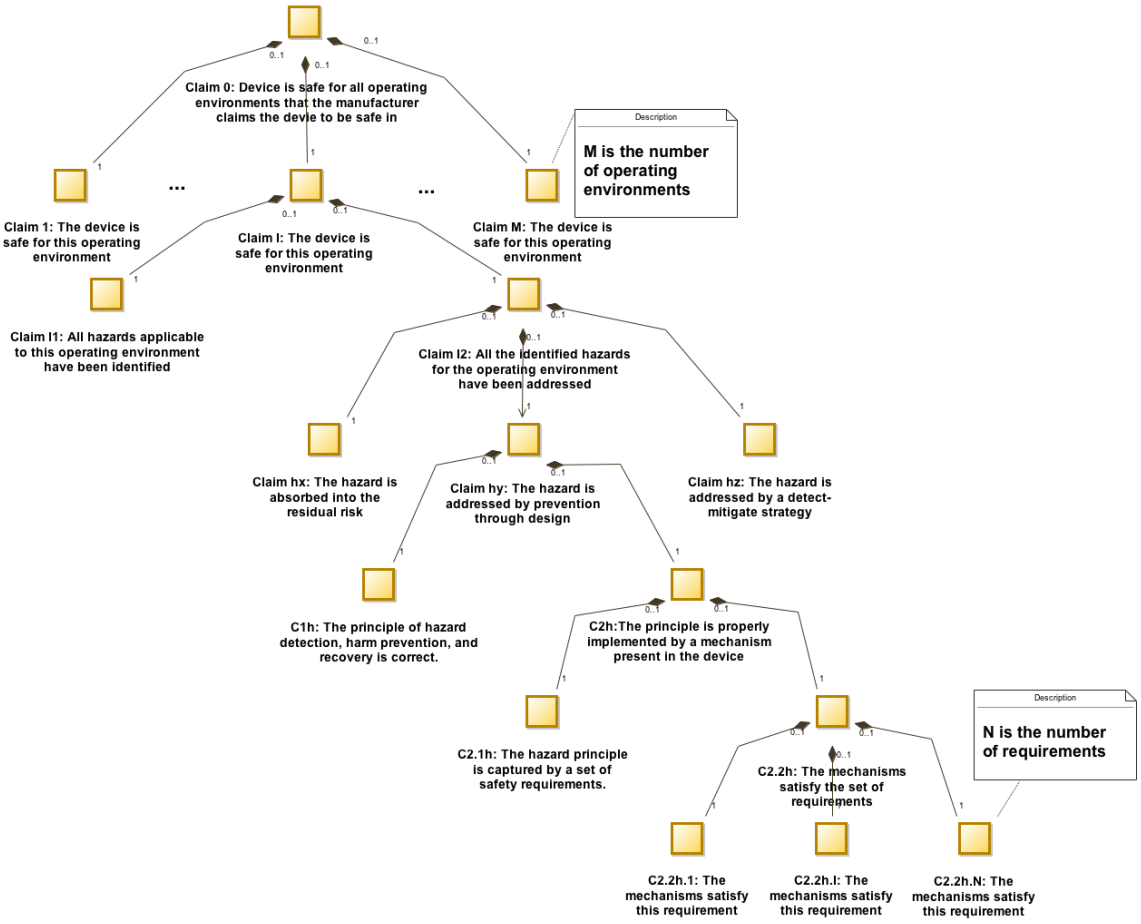

Figure 2.1: Ray's proposed assurance case structure. Adapted from [109].

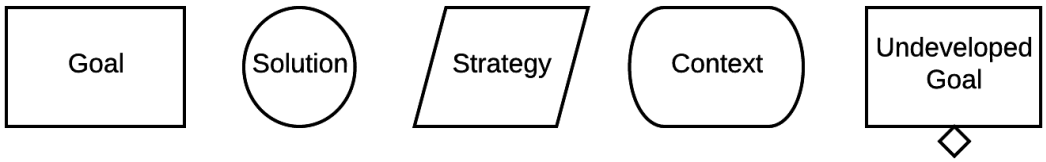

Figure 2.2: Elements GSN. Adapted from [62]

graphical representation, we will use GSN.

Goal Structuring Notation

GSN is a graphical notation that explicitly represents the claims, evidence and context of a safety argument as well as the relationships between them [62]. The notation uses four primary symbols - goals to represent claims, solutions to represent evidence, strategies which decompose goals, and context. Other elements are also used including justifications which can be used to explain why it is that the application of a given strategy is sufficient to demonstrate the satisfaction of the parent goal. Justifications can also be used to make explicit the rationale behind other aspects of the case presented in a given "goal structure". The term "goal structure" is used to describe the GSN graph which is assembled by a modeller to represent the elements of their argument and the relationships between them. Goal structures have goals as root nodes. These root nodes are composed of subgoals which provide decompositions the higher level goals, the decomposition method may be expressed using a strategy, and arguably it should be to make the approach to goal decomposition explicit. Sub-goals may also be subdivided in the same way. Eventually, the satisfaction of the most granular sub-goals is supported with evidence. As shown in Fig. 2.2: goals are represented with rectangular boxes; solutions are represented with circles; context is represented using a rounded box or oval; assumptions are considered as context; and justifications.

We provide an example goal structure in Fig. 2.3. The example is an extension of a goal structure presented by Ray [109]. Ray includes each the three most refined goals, and we extend this with possible solutions for those goals.

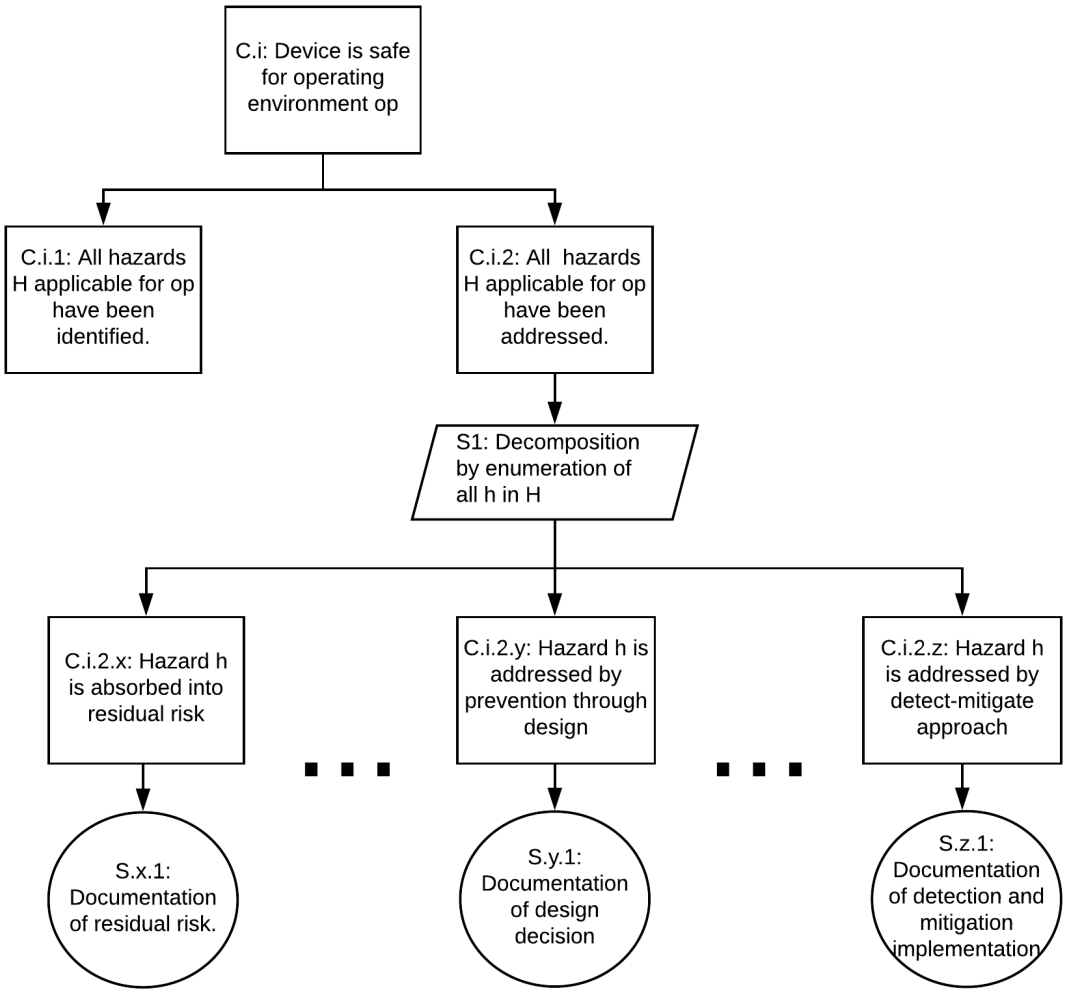

Figure 2.3: An exemplar goal structure expressed in GSN extended from one presented by Ray in [109].

2.4 Identifying Claims

Goals for assurance cases are developed by considering top level safety goals and decomposing them into sub goals. This decomposition should be methodical. It should identify any goals necessary to demonstrate the assured system property. The identification will result from a combination of approaches including review of regulatory requirements and product/service hazard analysis. One potential place to start a safety assurance case is to seek a Preliminary Hazard List (PHL) (Section 3.5) for the SUI.

2.5 Generating Evidence

The necessary methods of evidence generation for an assurance case are in part dependent on the process used to create the product/service under investigation. If a formal methods approach to software creation is taken, then a proof of correctness approach might be paired with the requirements as evidence of correctness in support of the product safety goal. With complex sociotechnical systems however, correctness alone does not address the hazard of systems errors. These systems' errors may in no way require a software error to occur. Software correctness may mitigate the risk of some loss events, but usage factors within the intended environment of use and other aspects of validation must also be argued. To address both of these issues in a broader process of evidence generation we turn to hazard analysis techniques.

2.5.1 Hazard Analysis

In [131], Stamatis provides an overview of a wide range of hazard analysis methods. Many of these are also discussed by Ericson in [28]. Stamatis provides four primary categories of approach: traditional methodologies, tree-based techniques, methodologies for dynamic systems and qualitative methodologies. We introduce a subset of these here. We then provide a theoretical comparison of these approaches to ISHA in Chapter 6.

2.5.2 Traditional Methods

Traditional methods include the What-If Method, Checklists and Interface Analysis. In a What If analysis, experts rely on their knowledge, thinking process, experiences and attitudes to thoroughly inspect a system for hazards. The SUI is decomposed into functional nodes - by static components, or dynamic behaviours. Something which is explicitly considered in the What If method which is excluded from many other analysis methods is layout. The What If method explicitly considers things like noise zones and escape paths for physical process layouts [131]. With Hazard Checklists, analysts consider a list of terms in a given checklist. The list is intended to spur conversation and imagination of the potential accident risks posed by the SUI. These lists are typically constructed of terms which focus on a specific safety concern - e.g., acceleration, chemical contamination, contingencies, control systems, or human factors. Hazard checklists for software are less commonly published, possibly as a consequence of the relative nascence of the domain of software safety. The technique is described by Ericson who also provides sample checklists in his appendix [28]. Interface Analysis is a scoping technique in which the interaction of a system with external stimuli is examined. This method addresses interaction issues which are commonly observed to lead to failures [131].

2.5.3 Tree Base Techniques

FTA and Event Tree Analysis (ETA) are two tree based techniques described by Stamatis [131]. FTA is a top down method in that it assesses the many failures which can lead to an unexpected top level accident. ETA is a bottom up technique in that it considers the many possible outcomes which might arise from a root unexpected event.

Fault Tree Analysis

In FTA, analysts build trees of failures using logical diagrams much like in digital circuit design. The method is deductive in that it begins with the identification of a top level unexpected event, and from there, analysts identify the mechanisms by which that event could realize. The analyses can be either qualitative or quantitative. The principle difference between these two approaches is the mathematical rigour required for the latter. When a quantitative approach is taken, Baysian theory can be used to

compute failure probabilities.

Event Tree Analysis

In ETA, the analysts consider a set of initiating events which perturb the system changing its operating state or configuration. These initiating events are considered as one event in a series that could lead to accidents. The additional events which are necessary to progress from the initiating event to the accident are called pivotal events [28]. Pivotal events may be mitigating or aggravating. Mitigating events are events which divert the behaviour of the system away from failure, while aggravating events either passively allow the progression of the failure or may actively promote it. This play in the nature of pivotal events allows analysts to consider a range or interaction hazards; however, the binary nature of the relationship leads to limitations in the distinguishablility and nature of expressible outcomes. Outcomes at each pivotal event are defined as successful or unsuccessful limiting the capacity of analysts to express partial failures and successes. The structure of the analysis is also sequential making investigation of timing issues via this method challenging relative to methods expressly designed for dynamic analysis.

2.5.4 Methodologies for Dynamic Systems

A range of methodologies for managing dynamic systems including GO, Digraph/Fault Graph, Markov Analysis, DYLAM, and DETAM are available for the assessment of dynamic systems [131]. The methods face a common challenge of being heavily biased towards quantitative evaluation. It is challenging therefore, to use these methods in design/redesign phases of analysis where there is an absence of the necessary data. Functional Hazard Analysis addresses these issues by taking a less structured approach. Analysts instead consider the mechanism by which a system function might not be fulfilled and what might result from the failure [28].

2.5.5 Qualitative Methodologies

Failure Mode and Effects Analysis

In FMEA [130], analysts identify failure modes and determine what their effects are. The severity (S), occurrence (O), and detectability (D) of failure modes are codified on ordinal scales. The three failure mode attributes are multiplied to produce a

Risk Probability Number (RPN) which is used to prioritize the failure modes for mitigation. FMEA is a reliability analysis tool which has been applied in the safety domain. Reliability and safety however, are not equivalent. Though the reliability of some system components can be necessary for safety, using FMEA as a sole tool for safety analysis neglects consideration of the true complexity of safety. While FMEA is most often applied using a structural approach (component failure), it can also be applied with either a functional or hybrid approach [28].

Hazard Operability

HAZOP "assesses the hazard potential arising from deviations in design specifications and the consequences faced by [an] operation or organization" [131]. The analysis involves using a series of guide words to predict the outcome of unexpected deviations in the flow of matter or information through a system [28, 69, 131]. As HAZOP is a qualitative technique, it does not suffer from the same demand for data that many dynamic analysis processes do; however, the inductive approach to following flow deviations through documented system designs depend on accurate high quality system design documentation for high quality analysis.

2.5.6 Systemic Hazard Analysis

Functional Resonance Analysis Method

In FRAM [52], analysts consider the impact of additive variance in system operation. The consequence of this resonance may be constructive in that it improves system production, but it may also be destructive if it results in accidents. Analysts using FRAM focus on typical system performance and attempt to identify sources of variance in primary system functions. They take a breadth first approach to model design by identifying and modelling critical system functions in the context of the purpose of the analysis. The relationships between these functions and the potential outcomes of variance in these parameters are assessed to determine potential safety consequences. This focus on functional dependencies, FRAM has the capacity to identify multi-source hazards.

In FRAM, systems are modelled as a series of functions in which the outputs of early functions in the process are linked to the inputs of the functions which are executed later in the process. These linkages are expressed using a six port function

model. FRAM functions have five input ports and one output port. The output is simply called the Output, while the inputs are called Input, Control, Timing, Precondition, and Resource. While no specific semantic is attached to the Output beyond those inherent in its title, the inputs have more nuanced semantics. We discuss the semantics of each of the FRAM ports in the following paragraphs. We illustrate a model of a FRAM function in Fig. 1.1.

Functional Resonance Analysis Method Port Semantics

Output The Output port of a FRAM function is the source to which the output of the function is pushed once the function has completed execution. This output can be matter, energy or information. Simultaneous with the production of this output, an Output also provides a control token which enables a connected function to begin execution, thus maintaining the same petri-net style semantics as those used in UML/SysML activity diagrams [52].

Input The Input accepts the necessary matter, energy, or information, required to produce the output that is generated on the Output of the function. The Input accepts the input necessary to start the function. It also receives the petri-net style control token. Tokens are not needed on the other four inputs for the function to begin execution [52].

Precondition The Precondition accepts inputs which are required to be true before a function can begin execution. If a precondition is not true after the function begins execution however, continued execution of the function is not precluded. Precondition inputs are not those that are used to produce the output, nor are they the ones that activate the function. They are the other conditions which are required before the function is allowed to begin execution. Hollnagel [52] discriminates between Precondition and Input ports by example with the takeoff sequence for an aircraft. He describes the Input in this situation to be the permission of air traffic control to takeoff. The plane cannot begin takeoff, if the pilots are following the rules, before this permission is given. Hollnagel describes the pre-flight checklist in the same situation as a precondition. The pilot should go through the pre-flight checklist before commencing his journey. On a private flight, there is no strict enforcement of this

rule. We suggest that the distinction made by Hollnagel is not entirely clear, but we also do not discriminate further [52].

Resource The Resource input to a FRAM function consumes an input which is required during the functions execution - "matter, energy, information, competence, software, tools, man-power and so on" [52]. Even if a resource is required for completion of a function, it is not required for the function to begin execution. For example, a baker can begin making a loaf of bread when he has no flour. Hollnagel suggests that there is value in distinguishing between *execution conditions* and *resources*. While *resources* are consumed in the execution of a function, *execution conditions* are simply conditions which must be true - e.g., the presence of a user account for privilege operations in a typical relational database management system. While a Precondition is required to be true before a function starts, an *execution condition* must hold throughout the execution of the function [49].

Control The Control input to a FRAM function accepts regulating input for that function's execution. Controls can be social or technical, manual or automated. They constrain the execution of the function. These constraints can be binary - the function executes or does not - analogue - the function executes within a rate range - or even qualitative - the function can only be performed by a specific combination of operating resources [52].

Time The Time input to a FRAM function accepts the timing regulation input for that function's execution. The timing of a function can be constrained by sequence, duration, or by temporal gates (start and stop times). The timing of a function is a subset of the Control inputs. Timing is provided as a distinguished port in recognition of the significance of its relevance to resonance [52].

System Theoretic Accidents Models and Processes

The STAMP framework [76] addresses safety as an emergent property of complex control structures. Designers construct systems with the expectation that the safety constraints necessary for their accident-limited operation will be enforced. In STAMP analysis, the control structures inherent in these designed systems are modelled as webs of control loops constructed of components who play one of four roles: *Controller*, *Actuator*, *Process* or *Sensor*.

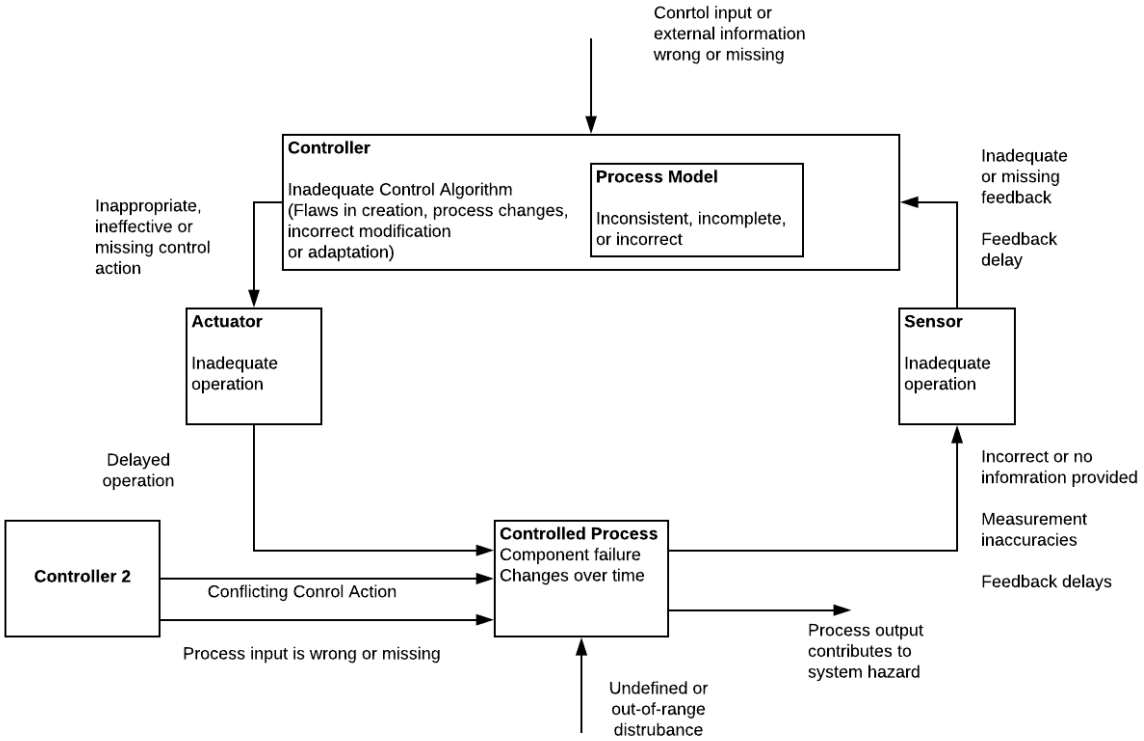

Figure 2.4: A diagram illustrating an atomic STAMP control loop, along with a variety of hazards related to specific system components and interactions. Adapted from [76].[a]

[a]By the strict semantics used in this dissertation, Controller 2 should be attached to the Controlled Process via an actuator and a sensor, but to minimize deviation from the original diagram we have not made these changes from Leveson's original representation.

An adaptation of such a control web provided by Leveson [76] is shown in Fig. 2.4. The control web includes a selection of abstract hazards which may be present in a given system. The control web also demonstrates by way of the use of a second controller how complex systems can be constructed[a]. One of the strengths of the STAMP framework is that its models are sociotechnical by design; components in a STAMP model are neither implied to be human nor machine - they may be either.

System Theoretic Accidents Models and Processes Stereotypes We now introduce each of the STAMP stereotypes and provide a brief description of both the *Process Model*, and of *Constraints*, two other concepts that are central to the STAMP framework.

Controller The *Controller* in a STAMP control loop is the operating entity. The *Controller* is the only STAMP stereotype which is modelled as being capable of rationaliza-

tion. *Controllers* are thus imbued with a *Process Model*. The controller may be a machine or machine component, or it may be a person, role, or even an organization.

Process Model *Process Models* are perceptions of the reality of the state of a controlled *Process*. They are only held by *Controllers* as these are the only stereotypes which are rational by definition. *Process Models* are used both to assess informational input, and to determine a course of action which a *Controller* will take in an attempt to guide the *Process* under control to a more desirable state.

Sensor The *Sensor* in a STAMP control loop measures the state of the *Process* and reports this state to the *Controller*. The *Sensor* is simple in that it does not rationalize, it only senses the state of the controlled *Process* and relays the uncovered state, verbatim, to the *Controller*. The sensor may be a machine or machine component, or it may be a person, role, or even an organization.

Actuator The *Actuator* in a STAMP control loop, like the *Sensor*, is an intermediary between the *Controller* and the controlled *Process*. It is the mechanism used by the *Controller* to guide the trajectory of the controlled *Process*. Like the *Sensor*, the *Actuator* is not rational. It simply responds to the *Controller*'s instructions and performs the requested action, verbatim, on the controlled *Process*. The actuator may be a machine or machine component, or it may be a person, role, or even an organization.

Process The *Process* in a STAMP control loop is controlled by the *Controller*. Its state can be continuously evolving. The trajectory of that state is guided by the *Controller* via the *Actuator*. Concurrently, its state is observed by the *Controller* by way of the *Sensor*. Like the *Sensor* and *Actuator*, the *Process* is not rational. It simply responds to actuation with potential changes in state and/or trajectory, and has its state monitored by the *Sensor*.

Constraints The *Constraints* in STAMP are used to express desired limitations on the operational behaviour and state of the system. *Constraints* are used to model desired limitations on both the attributes of the participating entities and also on their interactions. *Constraints* by themselves do not however ensure that the limitations are satisfied. *Constraints* are declarations about the desired state of the system. In order to ensure their enforcement, applicable controls must be incorporated into the system.

Interaction Stereotypes In addition to the role stereotypes, STAMP also implicitly induces constraints on the interactions between stereotypes: a *Controller* guides a *Process*

by means of an *Actuator*. The process is observed by a *Sensor* whose measures are reported to the *Controller*. As a means of embedding the directionality of associations, we define terminology to refer to each of these interactions here, so as to simplify the expression of model structures in subsequent references. These stereotypes are not explicitly declared nor applied by Leveson in her work [76].

Control The *Controller* is said to *Control* the *Actuator* when it attempts to guide the trajectory of the controlled *Process*.

Actuate The *Actuator* is said to *Actuate* the controlled *Process* when it is used by the *Controller* to guide the trajectory of the controlled *Process*.

Sense The *Sensor* is said to *Sense* the controlled *Process* continuously.

Perceive The *Controller* is said to *Perceive* the state reported by the *Sensor*.

Causal Analysis Using System Theoretic Accidents Models and Processes Causal Analysis Using STAMP (CAST) [76] is a retrospective hazard analysis technique (accident analysis technique) which is grounded in the STAMP framework. The method involves nine steps:

1. "Identify the system(s) and hazard(s) involved in the loss.
2. Identify the system safety constraints and system requirements associated with that hazard.
3. Document the safety control structure in place to control the hazard and enforce the safety constraint. This structure includes the roles and responsibilities of each component in the structure as well as the controls provided or created to execute their responsibilities and the relevant feedback provided to them to help them do this. This structure may be completed in parallel with the later steps.
4. Determine the proximate events leading to the loss.
5. Analyze the loss at the physical system level. Identify the contributions of each of the following to the events: physical and operational controls, physical failures, dysfunctional interactions, communication and coordination flaws, and unhandled disturbances. Determine why the physical controls in place were ineffective in preventing the hazard.

6. Moving up the levels of the safety control structure, determine how and why each successive higher level allowed or contributed to the inadequate control at the current level. For each system safety constraint, either the responsibility for enforcing it was never assigned to a component in the safety control structure or the component or components did not exercise adequate control to components below them. Any human decisions or flawed control actions need to be understood in terms of (at least): the information available to the decision maker as well as any required information that was not available, the behaviour-shaping mechanisms (the context and influences on the decision making process), the value structures underlying the decision, and any flaws in the process models of those making the decisions and why those flaws existed.

7. Examine overall coordination and communication contributors to the loss.

8. Determine the dynamics and changes in the system and the safety control structure relating to the loss and any weakening of the safety control structure over time.

9. Generate recommendations." [76]

System Theoretic Process Analysis System Theoretic Process Analysis (STPA) is a prospective hazard analysis technique which is grounded in the STAMP framework. The STPA method consists of two primary steps:

1. "Identify the potential for inadequate control of the system that could lead to a hazardous state. Hazardous states result from inadequate control or enforcement of the safety constraints, which can occur because:

   (a) A control action required for safety is not provided or not followed.

   (b) An unsafe control action is provided.

   (c) A potentially safe control action is provided too early or too late, that is, at a wrong time or in the wrong sequence.

   (d) A control action required for safety is stopped too soon or applied too long.

2. Determine how each potentially hazardous control action identified in step 1 could occur.

   (a) For each unsafe control action, examine the parts of the control loop to see if they could cause it. Design controls and mitigation measures if they do not already exist or evaluate existing measures if the analysis is being performed on an existing design. For multiple controllers of the same component or safety constraint, identify conflicts and potential coordination problems.

(b) Consider how the designed controls could degrade over time and build in protection, including

i. Management of change procedures to ensure safety constraints are enforced in planned changes.

ii. Performance audits where the assumptions underlying the hazard analysis are the preconditions of the operational audits and controls so that unplanned changes that violate the safety constraints can be detected.

iii. Accident and incident analysis to trace anomalies to the hazards and to the system design." [76]

2.6 Creating an Argument

The six-steps of a proposed assurance case construction method are enumerated below:

1. "Identify the goals to be supported;
2. Define the basis on which the goals are stated;
3. Identify the strategy used to decompose the goals;
4. Define the basis on which the strategy is stated;
5. Elaborate the strategy (and proceed to identify new goals back to step 1), or step 6;
6. Identify the basic solution" [40, 63]

Though this systematic process may yield predictably structured arguments, it is broadly recognized that following this recipe for each new assurance case may result in substantial rework. One of the primary concerns with the construction of assurance cases is that "the rationale connecting the recommended assurance processes, and the artifacts produced, to system safety is largely implicit." [23]. This concern has motivated the development of a lightweight methodology [23] and a range of assurance case patterns including those presented in [44, 8].

2.6.1 Lightweight Assurance Case Assembly

In [23], Denney describes his lightweight methodology for automatically assembling assurance cases from tabular requirements specifications. The proposed method promises *round trip engineering* between assurance cases and tabulated requirements. Denney argues that there are two parallel approaches to the development of assurance cases. Firstly, there is

the traditional standards based approach in which a variety of artifacts are generated to document the process by which a product was developed. Secondly, there is the development of assurance cases which support claims via argument structures which are supported by evidence. Denney goes on to suggest that many of the artifacts which are generated in the current standards based approach to safety argumentation can be incorporated into assurance cases.

Denney's methodology can be summarized as using graph transformations on a set of structured requirements artifacts:

- A hazard table with the following columns
  - hazard
  - cause
  - mitigation
  - safety requirement
- System and Functional Requirements tables with the following columns
  - requirement
  - source
  - allocation
  - verification method
  - verification allocation
- A set of requirements specifications is then defined as the combination of the hazard table, system requirements table and functional requirements table

Given this defined foundation, Denney describes a partial assurance case as a tuple of a goal G, a strategy S to decompose that goal, evidence E to support sub-goals into which G has been decomposed, assumptions A which are made, contexts K in which the strategy is executed, and justifications J which argue the applicability of the evidence to their associated goals or the applicability of strategies in the decomposition of goals.

The system documentation artifacts are assigned to elements of the safety model as follows:

- "hazard, requirement, causes → goal, sub-goal
- allocated requirements → sub-goal

- mitigation, verification method → strategy
- verification allocation → evidence
- requirements source, allocated artifact → goal context" [23]

2.6.2 Assurance Case Patterns

In addition to developing the GSN notation for assurance cases, Kelly and McDermid also developed the idea of pattern based reuse of safety argument components [64]. These authors first adapt an extant language to describe assurance cases. They identify a series of terms necessary to discuss assurance case patterns which includes:

- intent: describes in a brief statement the safety issue addressed by the pattern
- motivation: a scenario demonstrating the necessity for the pattern
- applicability: a statement describing the context in which the assurance case is valid
- structure: a graphical representation of the argument (a goal structure)
- participants: an enumeration of the elements in the goal structure
- collaborations: a description of how participants carry out the function of the pattern
- implementation: guidance on pitfalls in the use of the pattern

Having established some terminology with which to discuss assurance case patterns, Kelly and McDermid provide two such patterns: Hazard Directed Argument and Functional Decomposition Argument.

Hazard Directed Argument

In a hazard directed argument, analysts make two claims. First, they have identified the credible hazards posed by the SUI, and second, that each of those hazards has been sufficiently mitigated. This argument then rests on the fact that all unidentified hazards and the remaining risk of the identified hazards after mitigation is absorbed into the residual risk of the system.

Functional Decomposition Argument

In a functional decomposition argument, analysts begin with two similar claims. First, they have identified all the safety relevant functions of the SUI, and second, that the risk posed by each of these functions has been sufficiently mitigated. This argument then rests on the fact that no safety relevant function of the SUI has been missed, that the functions of the system are sufficient to achieve the goals of its users and will not require substantial workarounds, and that the risk remaining after mitigation of the risk of the identified functions is absorbed into the residual risk of the system. Analysts must also argue that functions are either independent or that the risk of their interactions is either mitigated or is acceptably absorbed into the residual risk of the system. Further analysts must also argue that workarounds which will be foreseeably developed will not compromise the safety features which has been built into the system.

Chapter 3

Information System Hazard Analysis

A Running Example Using a Computerized Provider Order Entry System

In this chapter we introduce Information System Hazard Analysis (ISHA) using a running example of diabetes management in a long term care residential setting that adopts an Electronic Medical Record (EMR) which is enabled with Computerized Provider Order Entry (CPOE) functionality. "Long-term residential care services provide 24-hour professional supervision and care in a protective, supportive environment for people who have complex care needs and can no longer be cared for in their own homes or in an assisted living residence" [98].

In this initial introduction we focus on the early phases of the ISHA method while in Chapter 4 we elaborate on its later stages. ISHA is a novel hazard analysis method that has been tailored to Clinical Information Systems (CIS) which has been developed for prospective hazard analysis. It is designed to be used prior to the occurrence of accidents in order to identify and hazards and qualify their detectibility, probable occurrence and potential severity. ISHA is based on the *analysis* tasks of the United States Department of Defense Standard Practice System Safety (MIL-STD-882E) [26]. This standard sets out a number of tasks which must be completed to assure the safety of products sold to the US military. Though the outputs of these tasks are specified in the standard, little guidance is provided on how to complete the tasks themselves. ISHA synthesizes five hazard analysis methods to overcome this lack of guidance.

The first two methods are traditional: Failure Mode and Effects Analysis (FMEA) (Section 2.5.5) and Event Tree Analysis (ETA) (Section 2.5.3). FMEA is used for component reliability analysis, and ETA for requirements analysis. Next, a simple systemic analysis method called HAZard OPerability (HAZOP) (Section 2.5.5) is used for interaction analysis. The outputs of these three methods are integrated and cross validated using a Universal Triangulation Model (UTM). The UTM is based on a composite of concepts drawn from

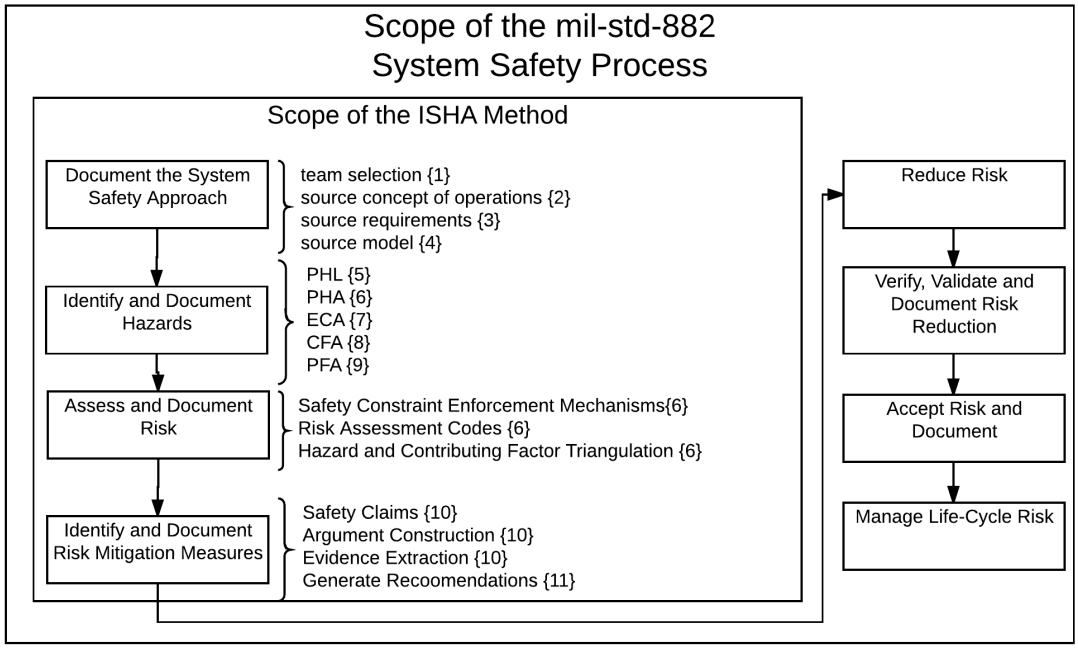

Figure 3.1: A flow chart adapted from MIL-STD-882E [26] modelling the scope of the ISHA method. The diagram illustrates the association of the ISHA activities with the relevant elements of the MIL-STD-882E process.

two systemic methods: System Theoretic Accidents Models and Processes (STAMP) (Section 2.5.6) and the Functional Resonance Analysis Method (FRAM) (Section 2.5.6). ISHA is executed on an System Under Investigation (SUI) which is prescribed by a *study initiator*. ISHA considers only the analysis and design phases of system safety; MIL-STD-882E further considers task phases spanning from risk reduction to life-cycle management (Fig. 3.1).

ISHA proceeds iteratively through six of the tasks listed in MIL-STD-882E. Analysts begin by executing the Preliminary Hazard List (PHL) task. A Preliminary Hazard Analysis (PHA) is then performed before the execution of an in-depth assessment of the SUI through analysis of its requirements, systems, subsystems and operations. An ETA variant we call Event Chain Analysis (ECA) is used to assess the requirements; FMEA, in conjunction with HAZOP is used for static and dynamic analysis of systems and subsystems. HAZOP is used to analyze the SUI's operations. Finally, a synthesis of the results of these analyses is generated using a composite of the STAMP and FRAM methods.

The ISHA method is comprised of the list of *activities* below. The observed noun/verb confusion is largely inherited from the source standard. These *activities* will be discussed in sequence in the subsequent sections. The relationships between the MIL-STD-882E elements and the ISHA activities are illustrated in Fig. 3.1.

1. Select team
2. Source concept of operation
3. Source requirements
4. Source model
5. PHL
6. PHA/UTM
7. Requirements Analysis (ECA)
8. Component Fault Analysis (CFA)
9. Process Fault Analysis (PFA)
10. Assurance case development
11. Generate recommendations
12. Repeat

Through the remainder of this chapter, we will describe each of the phases of the ISHA process. Alongside these explanations, we will demonstrate the method using our running CPOE enabled secondary care diabetes management exemplar process to demonstrate the application of each phase of the method. The CIS in question is asserted to leverage an EMR in its care process. EMRs are CIS software suites that are composed of modules spanning the continuum of care. An EMR for example, will include scheduling, lab, and prescription modules. It may also support Clinical Decision Support (CDS) functionality including drug-drug interaction checking as well as a range of other features.

3.1 Select Team

Stamatis [131, 130] provides extensive discussion of team dynamics in the context of FMEA, while Redmill provides the same in the context of HAZOP [111]. The ISHA method prescribes that a team selection process similar to one of these be used. All approaches to this process will have shortcomings which must be explicitly acknowledged by the analysts who execute the method in order to qualify their analyses.

At a minimum, an ISHA team shall ensure that each of the duties below are fulfilled. These are described here as roles, but what is important is that each of the duties be fulfilled

by someone on the team. It is important that teams be larger than a single evaluator to avoid bias and issue blindness; however, budget constraints will often limit team size. It is more important that the team have the requisite skill sets, and that the indicated responsibilities be assigned to at least one member of the analysis team, than that the team be of a specific size. It is also necessary that the responsibilities be distributed to ensure that the quality and timeliness of the execution of responsibilities is not compromised. The duties are itemized below and will be discussed in sequence in the subsequent sections.

- a *study initiator* [111] who has the authority to provide the team with a budget of resources to perform the analysis
- a *study leader* [111] who manages the execution of the analysis
- a *social designer* who knows the structure of the organization that operates the SUI and how that organization can adapt to meet the range of potential human resources demands which are indicated by SUI design variants
- a *technical designer* who knows how the supporting software is structured and how it could be changed
- *users*, including
  - *doctors* and *nurse practitioners* (NP) who are responsible for using the CIS software in the processes of diagnosis and treatment.
  - *nurses* - either licensed practical nurses (LPNs) or registered nurses (RNs) - who are responsible for administering treatment, observing patients, and providing patient education
  - *Medical Office Assistants (MOAs)* (outpatient settings) or *Unit Clerks* (inpatient settings) who are responsible for scheduling and clinical content curation in the CIS software
  - Relevant allied health professionals (e.g. physio therapists, respiratory therapists).
- a *clinical information expert* who has a depth of knowledge of clinical coding terminologies and medical record exchange formats
- a *cognitive ergonomics expert* [49] who is familiar with usability challenges and other Human Computer Interaction (HCI) issues
- a *recorder* to document meeting minutes, and to manage generated findings during the execution of the process

- an *auditor* who has the authority to sign off on the analysis and to demand rework where deficiencies are identified.

The *study initiator* prescribes the SUI for analysis. They also act as a project "champion". Project champions are necessary in Health Information Technology (HIT) deployment projects as those that do not have them have a poor prognosis [22]. Deployment projects bear substantial similarity to prospective hazard analyses, like those that use ISHA, as they have a shared purpose of "doing things right". The study initiator also sets time lines, manages communications with senior management and/or the executive, and helps recruit key team members. The *study initiator* is also the authority to whom the analysis team reports.

The *study leader* is responsible for the day to day planning and execution of the analysis. They should be an expert in the method so that they can facilitate the team when challenges in execution arise. Common problems include scope creep, challenges with the trade off between breadth and depth of analysis, and problems with system boundary specification. It is important that study leaders have skills in conflict resolution since they are acting as project managers. These skills are especially important in hazard analysis as the process invariably becomes at least somewhat adversarial when the work of system contributors is being criticized.

The *social designer* helps the team recognize the relevance and significance of social factors in the SUI including the delegation of responsibilities and authority between professional roles. These system facets are known to be a particularly acute source of conflict in CIS [7]. It is necessary for the team to generate recommendations which reflect the impact of the social change which will be induced. If the social impact is not considered, recommendations will either not be followed, or if they are it will be to the detriment of the performance and/or safety of the SUI.

The *technical designer* is critical as knowledge of the probable cost of software variants is invaluable in determining optimal choices in the SUI design process. The *technical designer* is also uniquely positioned to understand the resilience of the technical components of the SUI and the likelihood that they will expose a care provider to the risk of making a technology induced medical error.

In any organizational engagement effort it is important to solicit input from the gamut of stakeholders. Failure to do so will lead to a lack of cohesion in the work process and thus lead to the corrosion of collaboration. In clinical care, we identify doctors, nurses and MOAs as key stakeholders [106]. In any given SUI, the most important stakeholders may differ and thus must be identified by the analysis team.

Doctors are responsible for "promoting, preventing, maintaining [and] restoring health through diagnosis and treatment of disease or injury." They prescribe medications, and

perform surgery among other duties [134].

Nurses prepare patients and equipment for diagnostic procedures and for treatments. They also provide post-diagnostic and post-treatment care including medication administration, pain management counselling, hygiene management and patient education [134].

MOAs manage the administration of the process of care. This includes scheduling, paper work, and the curation of paper and electronic medical records (the records themselves not the software).

Clinical Information Experts have a specialty in the breadth of formalizations of healthcare information which are used in CIS technologies. These include clinical coding terminologies like the Systematized Nomenclature of Medicine – Clinical Terms (SNOMED-CT), Logical Observation Identifiers Names and Codes (LOINC), International Classification of Disease (ICD) and others. They are also experts in document exchange and messaging structures including those provided by Health Level 7 (HL7). As information exchange is a central function of information management systems, it is critical that the impact of such exchange be well understood during any safety analysis of these systems [126]. The *clinical information expert* fills this gap and supports the team in understanding the strengths and limitations of exchange formats, clinical coding terminologies and data structures for health information.

Cognitive ergonomics experts are familiar with the interrelation between work and the mind [50]. The relationships between the two have been implicated in a range of challenges with CIS technology from problematic work flows [6] to fundamental safety concerns [126]. This role on the team is responsible for highlighting the various challenges that might arise in the SUI due to HCI problems or challenges related to distributed cognition [55].

The *recorder* is responsible for documenting the analysis as it proceeds. They take meeting notes when the team convenes for discussions. They manage the documents which are generated over the course of the analysis. Lastly, they are responsible for assembling the documentation when the analysis is complete and generating final reports which convey findings and recommendations.

The *auditor* is responsible for objectively assessing the quality of the analysis output. They have sign off authority, and also the authority to demand rework if necessary. The auditor, though necessary for the complete process, is not part of the analysis team. In the same way that accounting and audit must be divided, so must analysis and audit. The segregation of these duties prevents abuse of power by restricting the capacity of the team of analysts to qualify their own work.

3.1.1 Running Example

In our running example, we are prescribed the process of diabetes management in the context of an EMR enabled CIS which is a secondary care context. An appropriate team for this analysis might include:

- a Chief Medical Information Officer (CMIO) who takes on the duties of the *Study Initiator*.
- a licensed Professional Engineer (PEng) in software who has expertise in the ISHA method and who takes on the duties of the *study leader*.
- a doctor with experience in EMR adoption evaluation, and a range of experience in other organizational maturity activities in healthcare who takes on the duties of the *social designer*.
- a software development team leader who builds the EMR used by the CIS and who takes on the duties of the *technical designer*.
- a pair of doctors who work in the CIS and are regular *users* of the EMR.
- a pair of nurses who work in the CIS and are regular *users* of the EMR
- a pair of MOAs that are regular *users* of the scheduling and curation feature of the EMR
- a licensed Health Informatician who fulfills the duties of the *clinical information expert*
- an ethnographer who fulfills the duties of the *cognitive ergonomics expert* with experience in observational studies of workflow in healthcare.
- a junior quality of care specialist at the health care practice that fulfills the duties of the *recorder* and reports to the CMIO
- the department head of the health care organization's quality of care initiative who reviews the findings and takes on all of the other duties of the *auditor*

This team, though extensively skilled, is likely to be an expensive team to operate. There are many tasks to be completed in a healthcare organization and the absence of these members of the organization from their regular duties poses a substantial opportunity cost. Further, placing a junior team member in the role of the *recorder* may compromise the quality of the output as substantial value in such studies is derived during the final synthesis of information.

3.2 Source the Concept of Operations

The concept of operations is a simple description of the purpose and behaviour of the SUI. The intent of the description is to quickly situate the analysts by providing a conceptual understanding of the SUI and its boundaries. The concept of operations should be synthesized from the range of available sources including the SUI design, and the input of design, implementation and operations experts. The description should be concise, but sufficient to accomplish the stated purpose.

3.2.1 Running Example

In our running example we are assessing the process of diabetes management in an long-term care residential setting that uses an EMR software product as a core part of the clinic workflow. In context, an EMR is defined to be:

> *A collection of one or more patient-centric software programs which operate on the retrospective, perspective and prospective data related primarily to the breadth of the longitudinal, general medical treatment of a patient by providing capture, manipulation, analysis and search functionality*

3.3 Source Requirements

Four kinds of requirements are considered in ISHA:

- Functional Requirements (FRs)
- Non-Functional Requirements (NFRs)
- Constraints
- Safety Requirements (SRs)

The FRs specify what the SUI must *do*, while the NFRs specify what the SUI must *be*. Many NFRs are referred to as *ilities: maintainability, testability, interoperability, portability, etc. The constraints specify mathematical bounds within which behaviour and properties of the SUI must remain. The SRs are the subset of FRs, NFRs and constraints which specify the acceptable occurrence and detectability of hazards, as well as the acceptable severity of accident outcomes. The initial version of the requirements delivered to the ISHA team is presumed not to be extensively validated and also to be incomplete, especially

with respect to safety. The purpose of hazard analysis is to develop an understanding of the hazardous aspects of the SUI and identify those components, activities or environmental properties which pose the threat of an accident. Hazard analysis one mechanism through which SRs are identified. They can be solicited directly as we suggest be done in this phase of the ISHA method, but it should be expected that the SRs identified in this way will be incomplete, underdeveloped and otherwise largely deficient.

Due to the assumptions about the completeness and validity of the requirements, a process must be established to develop confidence in these attributes of the requirements. Though validation of end products has been central to software engineering for many years - yielding user acceptance testing for example - research on the direct validation of requirements remains sparsely published [4]. For this reason, in ISHA it is only prescribed that a rigorous process be established to perform this validation. The process may involve extensive interviews with users and domain experts. It may also involve logical analysis of the stated requirements to identify conflicts or under-specification of the operational context in which each requirement is valid. The assessment of requirements completeness suffers similar challenges. This is addressed through documentation of the requirements gathering process.

3.3.1 Running Example

The Health Informatics - Requirements for an Electronic Health Record Architecture (ISO 13308) standard provides requirements for EMRs. Sittig complements these with a list of anti-requirements which address patient safety concerns [126]. Sittig calls these characteristics *red-flags*. They are observable EMR states which suggest the presence of a patient safety hazard. From these two sources we derive a small set of requirements with which to demonstrate the ISHA method. Our selections are based on the value they provide in the explanation of the method and are neither intended to be complete, nor the most important requirements of an EMR. We will begin by providing a brief discussion on each of our chosen requirements sources. We will then provide a list of FRs, NFRs and constraints derived from our sources. From these we will identify which of these are SRs before proceeding with our analysis.

ISO 13308

The ISO 13308 standard lays out the purpose of an EMR, the stakeholders for the technology, characteristics of the technology, and details eight dimensions of functionality. Few, if any, EMRs implement all of the functionality, but many will implement a large portion of it. We list the specified functional dimensions below:

1. Structure
2. Process
3. Communication
4. Privacy and Security
5. Medico-Legal
6. Ethical
7. Consumer-Cultural
8. Evolution

Sittig and Singh's List of Electronic Medical Record Safety Red-Flags

Inspired by Menon's findings [89] in a survey of the American Health Lawyers Association (AHLA) and American Society for Healthcare Risk Management (ASHRM)'s patient safety concerns, Sittig and Singh developed lists of *red-flag* events which indicate patient safety hazards in CIS [126]. Each of these categories list *red-flag* states which can be observed in CIS that indicate the presence of patient safety hazards. These categories include

- Incorrect patient identification
- Failure to heed a computer-generated warning or alert
- System-to-system interface errors
- Failure to identify, find, or use the most recent patient data
- Open or incomplete orders

Functional Requirements We list our selection of requirements below. We remind the reader that this is a small subset of the requirements presented in ISO 13308 and in Sittig's Red Flag list [126] chosen for the value they provide in the explication and demonstration of the ISHA method.

FR1 The system must support recording/retrieving of demographics in the patient registry. These demographics consist of age, gender, name (first, last, middle), Health Insurance Number (HIN), contact information and other identifying information. [Derived from *Administrative, Medico-Legal, and Ethical* sections of [5]].

FR2 The system must support the generation and documentation of medication prescriptions. [Derived from *process and structure sections* of [5]].

FR3 The system must warn a user when they attempt to create a new demographic when one with similar attributes is already present in the system. [Derived from *Incorrect Patient Identification - I1* [126]].

FR4 The system must automatically generate structured and coded bills from encounter notes and send them electronically to the paying authority [Derived from *Administrative Data* section of [5]].

FR5 The system must not compromise the accuracy of incoming laboratory results [Derived from *System-to-System Interface Errors - I5* [126]].

Non-Functional Requirements

NFR1 The system must record demographics with high integrity [Derived from the *Privacy and Security* section of [5]].

Constraints

C1 The system must provide no more than an average of [one] drug-drug interaction alert per [five] generated prescriptions. [Derived from *Failure to Heed a Computer Generated Warning or Alert - I2, I5* [126]].

C2 The system must alert physicians about laboratory orders for which no results have been returned within 48hrs of expected completion [Derived from *Incomplete Orders - I2, I3*].

Safety Requirements Each of the requirements and constraints identified has the potential to cause "harm" to patients. Even *FR4* - which deals with billing - can harm the patient. If the patient is the payer then he/she could suffer a financial loss if a bill generated for one of their encounters were erroneous and they were forced to pay it; he/she would then suffer a financial loss. Our focus of harm however is physical injury, and so we exclude *FR4* from analysis. It should be noted however, that these safety requirements are a subset of the safety requirements of the system. These will be supplemented with the requirement that each of the hazards identified in the analysis be sufficiently mitigated.

3.4 Source System Model

The model of the SUI's process provides a concrete design which can be assessed for realized hazards. The model must be synthesized from all available sources. Using a concrete model allows for evidence based assessment of risks posed by a proposed or inferred SUI implementation. For this reason ISHA makes extensive use of the model. System models may come in many forms, textual, diagrammatic or even formal. The structure is irrelevant so long as the model supports systematic, evidence based evaluation.

3.4.1 Running Example

In our running example, we source a models of the static structure and dynamic behaviour of our SUI to as demonstrations of the types of artifacts which might be used in analyses. We do not further address these models in the dissertation. They would typically be used in

We source an initial static model of an EMR from work by McDonald on Electronic Health Record (EHR) systems [87]. This model is consistent with the architecture for such systems proposed in ISO 13308. Further, it is also consistent with Payne's architecture for a clinical computing system [103]. More complete and arguably realistic examples exist including the HL7 Reference Information Model (RIM) and the work done for the Open EHR project [99]; however, these alternatives are more complex and would require lengthy explanations that would obscure our focus on the *Source Process Model* phase of the ISHA method. McDonald's informal static model of an EMR is provided in Fig. 3.2.

This general architecture for an EMR however, is too abstract for our purposes. In this model, the prescription module, *Order Entry*, is modelled as four boxes - the computing module, the persistence module, the central interface, and a patient database. Other modules are also described including the *Data Entry & Results Review* module and the *Event Monitor* modules which may play a significant role in our diabetes management example; but, a more refined view of the interface design and linkages between components would provide greater visibility into the nature of the hazards which are relevant to the SUI. For this reason we provide a second more refined model of an EMR based on an incident investigation performed by Horsky [54] (Fig. 3.3).

For the dynamic model of the diabetes management process we rely on the guidelines proposed by Diabetes Canada [25]. These guidelines are in alignment with a more generic prescription model developed by Bassi and Partridge [9]. We model this workflow in Fig. 3.4. This initial model however is insufficiently refined to address the interactions between the provider and the EMR in the CIS. It also fails to address the importance of other care team

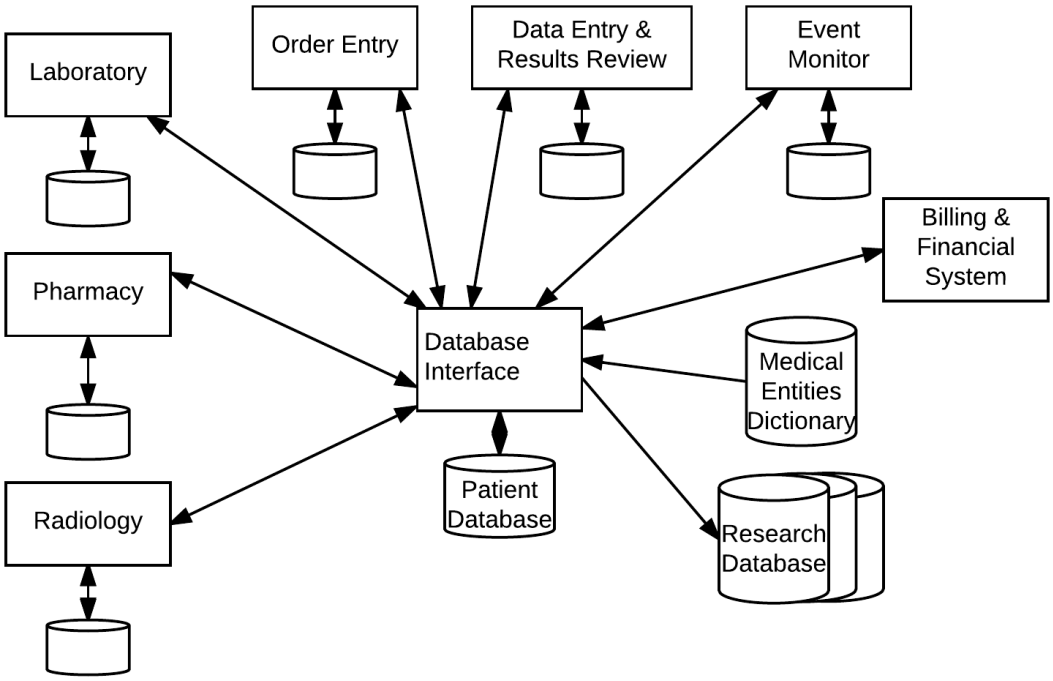

Figure 3.2: McDonald's informal static model of an EMR - replicated from [87]. The arrows represent the bidirectional information flow between a central unifying patient record and the disparate subsystems from which patient information is extracted and to which it is persisted.

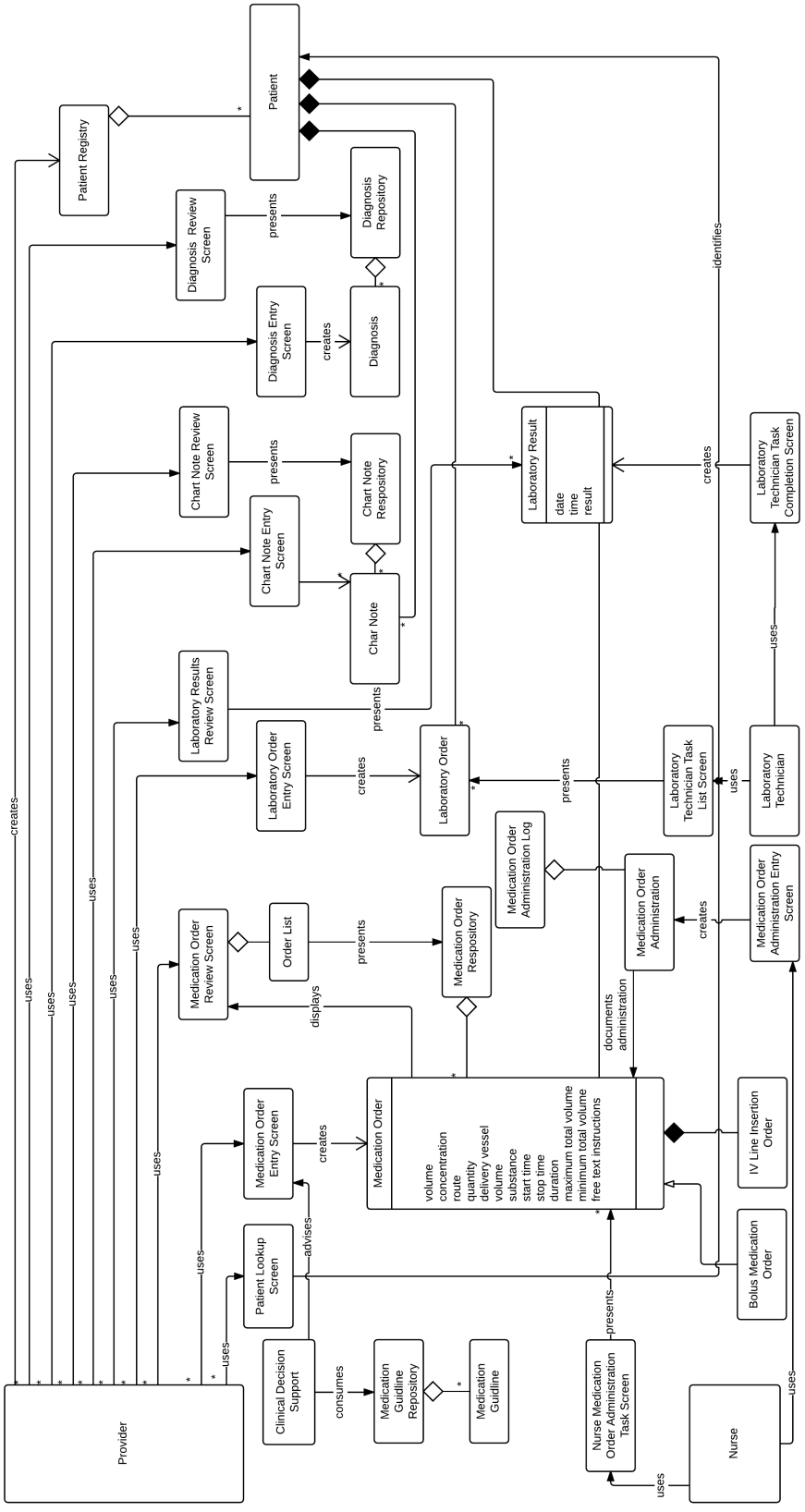

Figure 3.3: A UML class diagram of an EMR extracted from Horsky's investigation of a medication dosing error [54].

members like the nurse who plays a key role in secondary care contexts. For this reason, we extend the model provided in Fig. 3.4 to incorporate these additional interactions by using swim lanes and additional activities. These assignments of responsibility and additional activities are derived from a combination of our knowledge of CIS architecture and the role definitions we established in Section 3.1. This extended model is presented in Fig. 3.5.

3.5 Preliminary Hazard List

The PHL phase of ISHA is comprised of four sub-activities:

1. Identification/construction of a base PHL
2. Description of hazards
3. Hazard checklist
4. Hazard mapping

The outputs of the ISHA PHL phase are:

1. References to external materials including:
    - Root PHL
    - Hazard checklist
    - Other relevant artifacts used in the analysis
2. A brief description of each hazard
3. A mapping of the hazards to the model of the SUI

The base PHL identification portion of the PHL phase is modelled using a SysML activity diagram in Fig. 3.6. The portion of the PHL phase in which the base PHL artifact is consumed is modelled using a SysML activity diagram in Fig. 3.7. Having established the relationships between these outputs and sub-activities, we now address each sub-activity in turn.

3.5.1 Identification/Construction of a Base Preliminary Hazard List

ISHA's PHL artifact takes as its base an existing PHL for a substantially similar system. In medicine as a more broad domain, examples of these lists include the Joint Commission

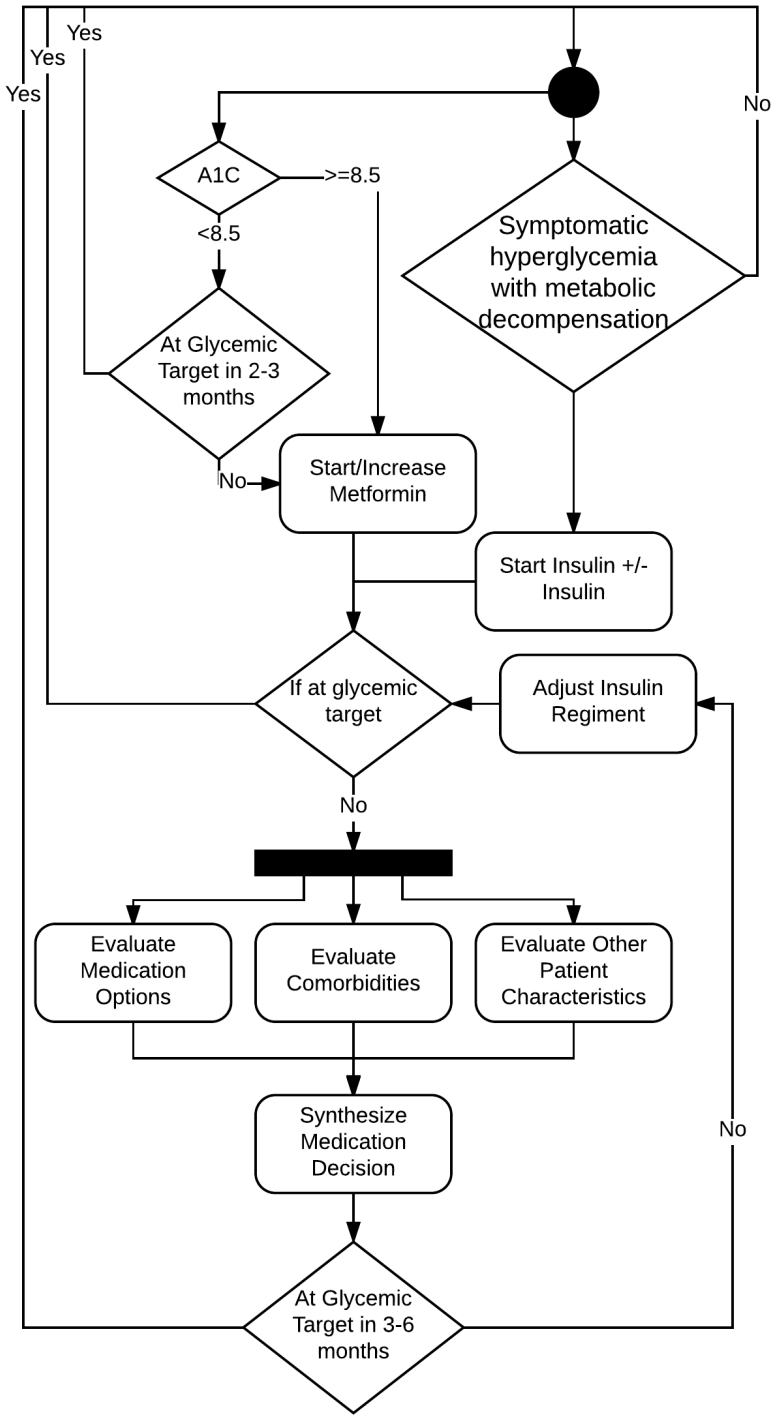

Figure 3.4: The workflow for diabetes management provided in the Diabetes Canada guidelines - adapted from [25]

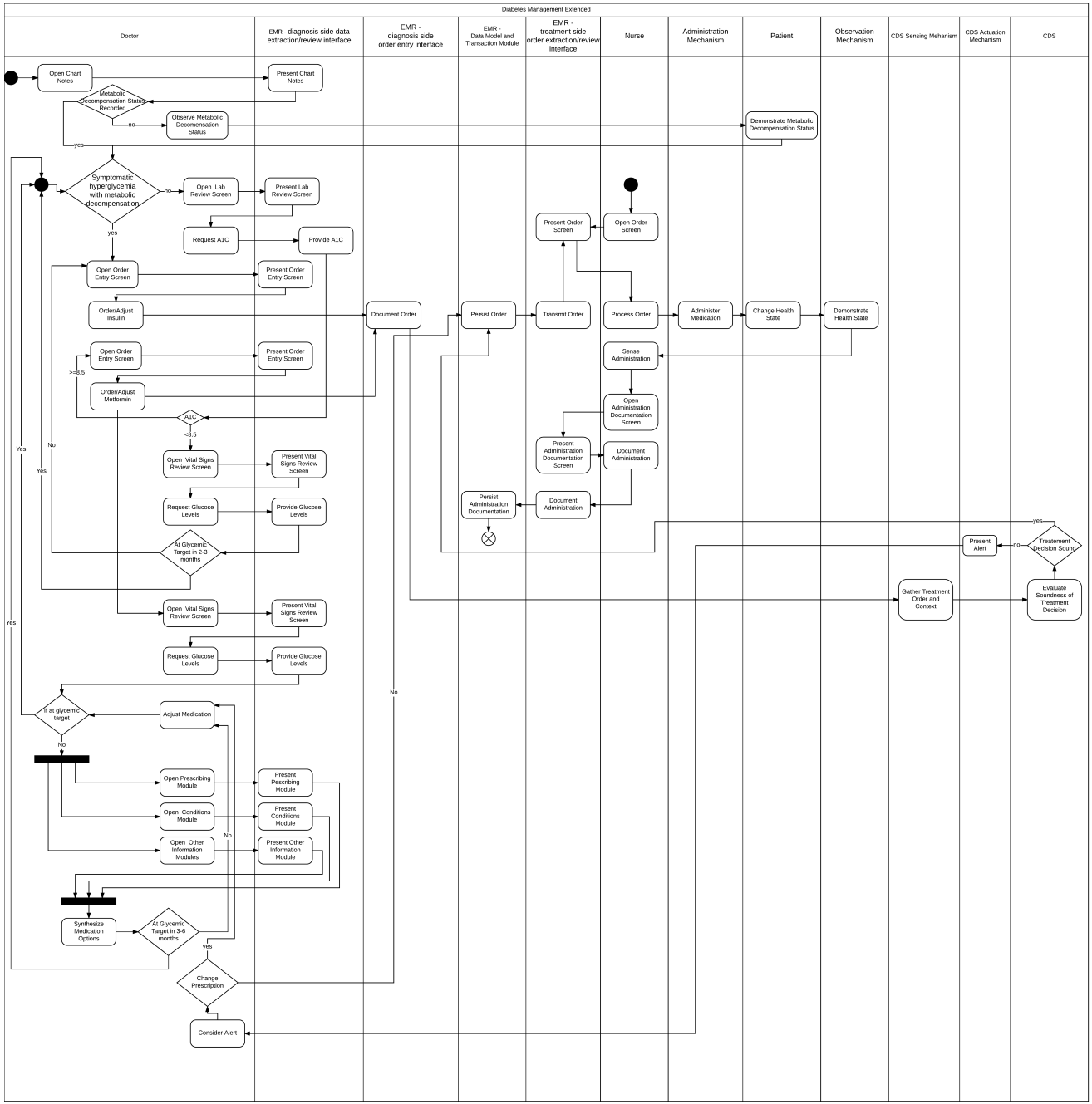

Figure 3.5: We present an extended workflow model for diabetes management in this figure which is synthesized from the Diabetes Canada practice guidelines, the role definitions we established in Section 3.1, and our knowledge of EMR architecture.

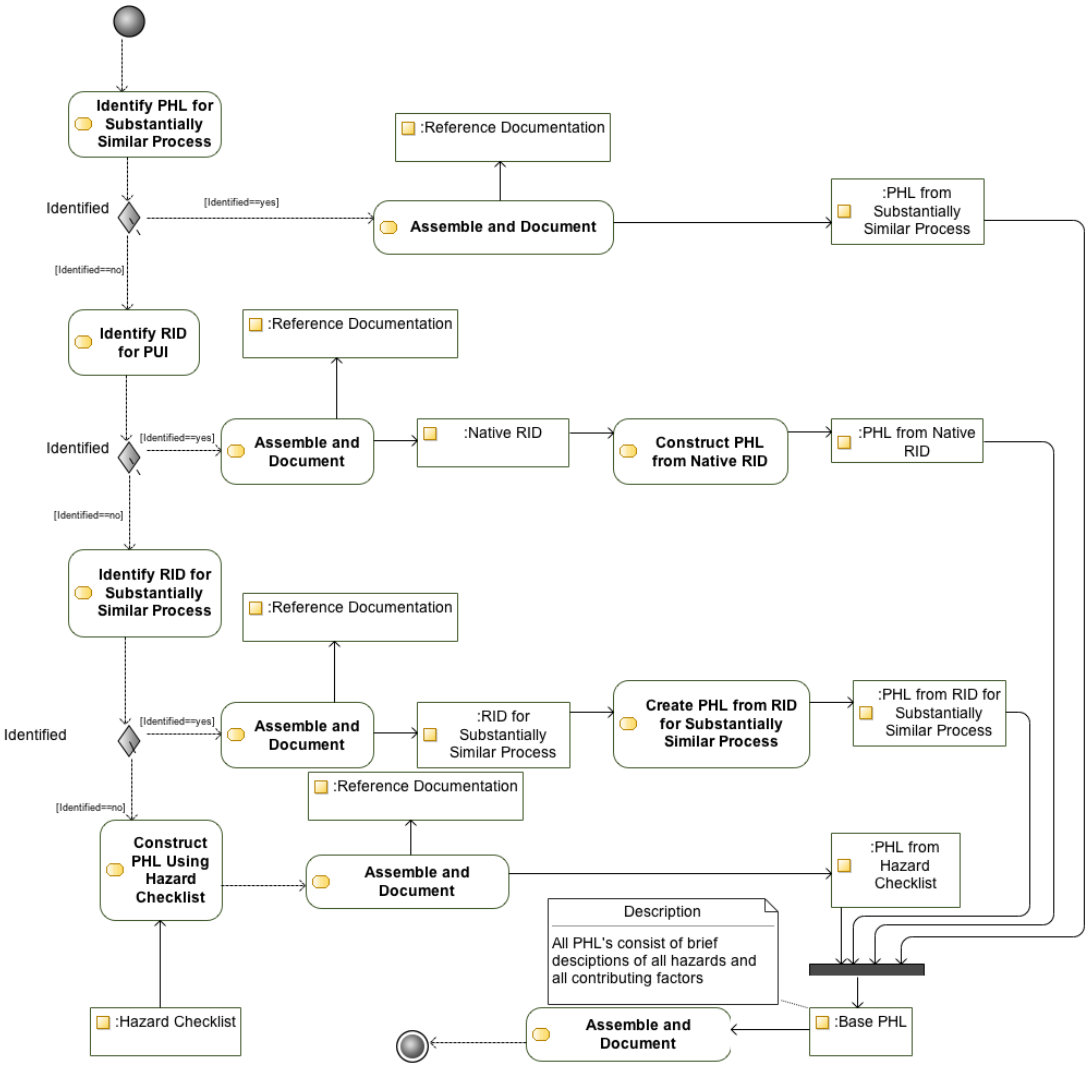

Figure 3.6: A SysML activity diagram modelling the process of constructing the base PHL in the ISHA method.

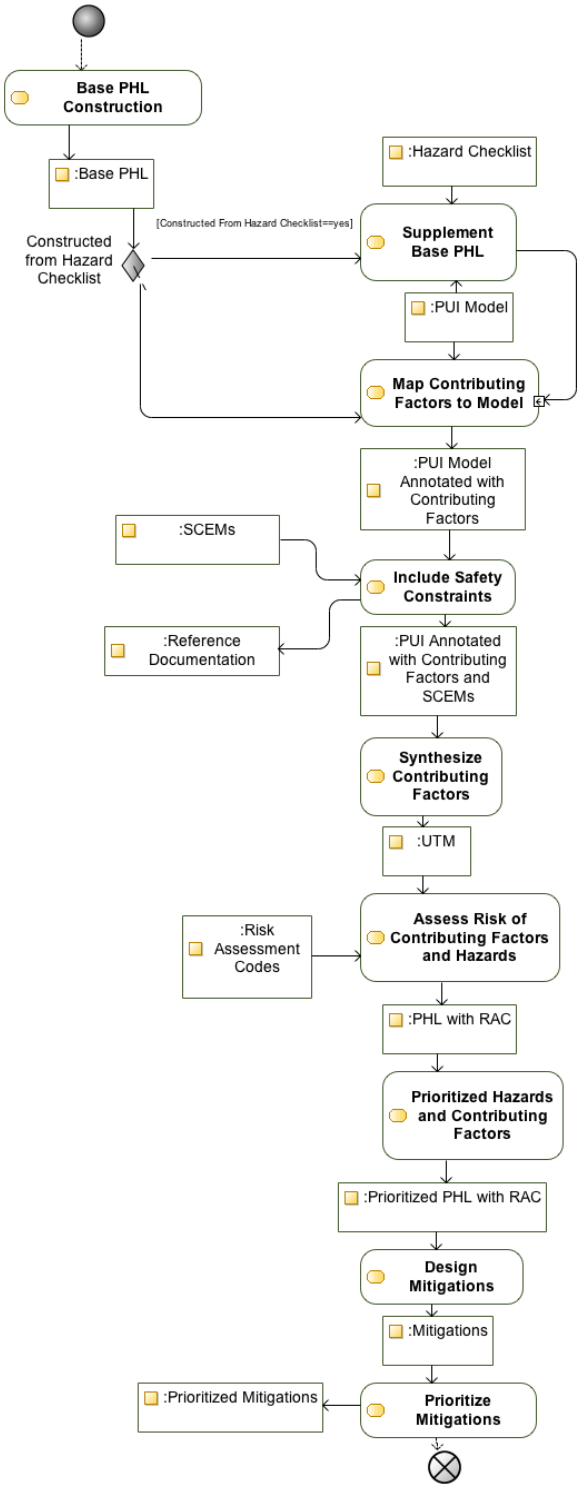

Figure 3.7: A SysML activity diagram modelling the portion of the PHL phase of ISHA in which the base PHL is consumed.

on Accreditation of Healthcare Organizations (JCAHO) taxonomy [17], the National Coordinating Council for Medication Error Reporting and Prevention (NCC MERP) taxonomy [95], and the Agency for Healthcare Research and Quality (AHRQ) Hazard Manager Ontology [137]. CIS specific lists exist as well, though they are less refined. For example, a number of taxonomies of CIS related hazards are available including those of Myers [93], Weber [141], and Magrabi [79]. We have also developed one of our own [29]. The gap between these taxonomies and what might be expected in a PHL is a matter of form and specificity. In addition, Sittig and Singh's red-flag lists [126] bears substantial similarity to a PHL, but possesses the same weakness in form as the listed taxonomies.

In the PHL phase, the root PHL is supplemented using Retrospective Incident Data (RID) for the SUI, or RID for a substantially similar system if native RID is not available. The PHL is further reinforced by applying an appropriate hazard checklist which may be synthesized from a range of sources. If a PHL for a substantially similar system cannot be identified, then one is created from the SUI's RID, or from the RID of a substantially similar system. If there is no RID for the SUI or for a substantially similar system, hazard checklists are used to create a base PHL.

ISHA deviates only marginally on a few domain specific sources from MIL-STD-882E in its prescription of the following RID for PHL construction:

1. Mishap and incident reports
2. Hazard tracking systems
3. Lessons learned
4. Safety analyses and assessments
5. Health hazard information
6. Test documentation
7. Environmental issues at potential locations for system testing, training, fielding/basing, and maintenance
8. Decommissioning plans

Running Example

In our running example, we identify Sittig and Singh's *red-flag* lists [126] as the basis for a CIS PHL. The lists were developed based on lessons learned as reported by expert panels, in addition to the experience of the authors - an informatician and a medical doctor. This

experience arises in part from investigation of hazard tracking systems [93], safety analyses and assessments [124], and incident reports [93]. This demonstrates that the basis for the lists in the range of appropriate RID for an ISHA analysis. We only address a couple of listed hazards in these lists in the interest of brevity for this demonstration:

1. "The 'error-log' of the interface between components of an EHR contains orders or results that were not able to be transmitted automatically between different components of the EHR system." [126]

2. "Reports show widespread non-adherence to computer generated alerts that are based on recommended guidelines." [126]

It is apparent from these listings that hazards are being inferred, but as was mentioned, there is a gap between the structures typically expressed in a PHL and what is found in typical sources of hazard descriptions. Analysts will often find such discrepancies when seeking PHL's and will have to translate the contents of the sources they discover to a more consumable form. This is the purpose of the next phase of the ISHA method.

3.5.2 Hazard Descriptions

All hazards identified in the analysis must be documented using a brief description. A standardized nomenclature consisting of both a grammar and hazard taxonomy must be used to construct these descriptions. For those hazards identified in the base PHL, the brief description can be extracted from the source document, but will need to be translated into the standardized nomenclature. It may be necessary to supplement the descriptions to support the demands of the hazard grammar. This will occur if the grammar demands the incorporation of more detail in the description than is provided in the source document. For example, a PHL might state that poor data quality poses a threat to patient safety. The nomenclature might require that the actor who encounters the hazardous condition be described in the definition of the hazard. For the hazards identified using the hazard checklist or for those identified from RID, analysts create brief descriptions of hazards as part of the execution of this phase of the ISHA method. The descriptions must be sufficiently detailed that a future analyst could understand the problem at a high level, without further information. One potential pairing of nomenclature and taxonomy would be the combination of the Phillips and Gong EMR error nomenclature [105] which addresses hazard description structure, and Magrabi's popular [101, 16, 19, 138] HIT related error taxonomy [79]. The Phillips and Gong nomenclature requires specificity in the description of hazards by necessitating the inclusion of the system element in error, the condition of the error, and the context of this error - we describe this in Tab. 3.1. Though this specificity

can facilitate the mitigation of hazards by making clear what potential causes are, the application of this nomenclature also risks addressing hazards at such a granular level that numerous mitigations might be designed to address these granular hazards when one higher level Safety Constraint Enforcement Mechanism (SCEM) could be sufficient and also more cost effective.

Running Example

In our running example, to address the expression of our chosen safety concerns in a standardized form, we first need to express them as hazards. We use the following re-expressions:

Hazard 1 Medication doses are not delivered, causing harm to patients, because the software fails to transmit the orders through the system from the doctor to the administering nurse.

Hazard 2 Medication overdoses occur, causing patient harm, because physicians suffer alert fatigue and ignore relevant CDS warnings on account of the volume of warnings the doctors regularly dismiss.

Part of the gap between the Sittig and Singh *red-flag* list and traditional PHLs is that it discusses the hazards at a high level without identifying specific and granular conditions which increase the probability of an accident. Therefore, we are required to seek additional sources of information to fill in this gap and provide not only the high level hazards, but also the more detailed hazards which moderate the risk of the more abstract hazards. To identify these more granular hazards we investigate RID - in this case, a combination of academic literature and incident reports. We identify, for the purpose of demonstration, two potential related hazards of greater granularity/specificity for each of our identified abstract hazards:

Hazard 1

Hazard 3 medication orders which are entered near the end of the day and use ambiguous relative temporal terms like *tomorrow* have the potential to be delayed by 24hrs due to misinterpretation [71]

Hazard 4 medication orders entered in one work station may not be persisted in such a way that they propagate to other workstations [32]

Hazard 2

Hazard 5 the ratio of significant alerts to alerts of marginal importance is too low [92, 85]

Hazard 6 the volume of information in alerts is too large making it challenging for users to extract the important details from the less relevant context [92, 85]

For our running example we will use the Phillips and Gong nomenclature [105] and our STAMP-EMR [29] taxonomy to re-express *Hazard 1* and its two identified more specific/granular hazards: *Hazard 3* and *Hazard 4*, as well as *Hazard 2* and its two identified more specific/granular hazards: *Hazard 5* and *Hazard 6*. It is important to note that in the identification of sub-hazards, analysts will often uncover hazards which interact in unexpected ways. A matrix approach to interaction analysis is incorporated into the triangulation phase of the ISHA analysis to address this (Sections 3.10, 4.9). Before constructing these standardized hazard descriptions, we introduce our STAMP-EMR CIS hazard taxonomy and the EMR error nomenclature provided by Phillips and Gong [105].

System Theoretic Accidents Models and Processes for Electronic Medical Records - A Clinical Information System Hazard Taxonomy We developed the STAMP-EMR taxonomy through a combination of literature review and incident report inspection [81, 82, 142, 143, 29]. The model is grounded in Leveson's STAMP framework [76]. It provides a coarse outline of the components involved in CIS systems and their safety relevant characteristics. It is necessary to customize the model to address acute issues in each analysis where it is used. The model is a starting point for analysis rather than a complete representation of all potential hazard sources present in a CIS. It is illustrated in Fig. 3.8.

In the diagram, the squared boxes model CIS components. The rounded boxes represent the safety sensitive attributes of these components. We refer to these attributes as hazard concerns. These hazard concerns were chosen to address the significant portion of hazards that can, in the CIS domain, be attributed to problems with usability [54, 73], availability [59, 60] and integrity [81, 82, 83, 29]. The characteristics are abstracted from a taxonomy of safety factors we developed during an application of the ISHA method on an electronic medical record exchange process [83]. The concerns represented in the STAMP-EMR model are aggregated at a high level and can be decomposed to greater granularity when desired. The taxonomy we developed in [83] that provides a more granular view of the safety factors expressed in STAMP-EMR is illustrated in Fig. 3.9. Definitions for the terms used in both the STAMP-EMR model and the more granular safety factor taxonomy are included in the glossary (Appendix E).

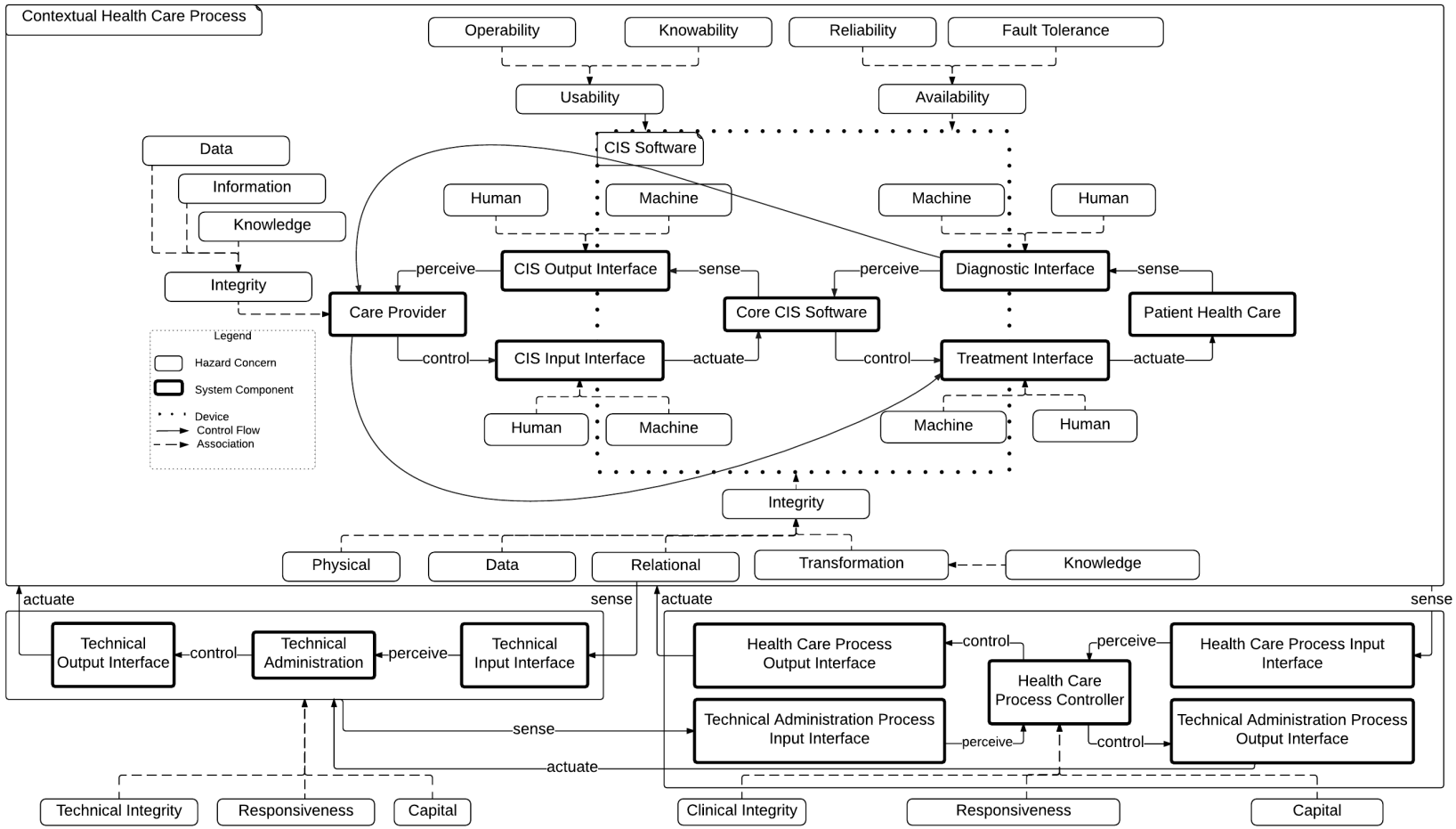

Figure 3.8: The STAMP-EMR model provides a semantically supported terminological basis to describe CIS related hazards.

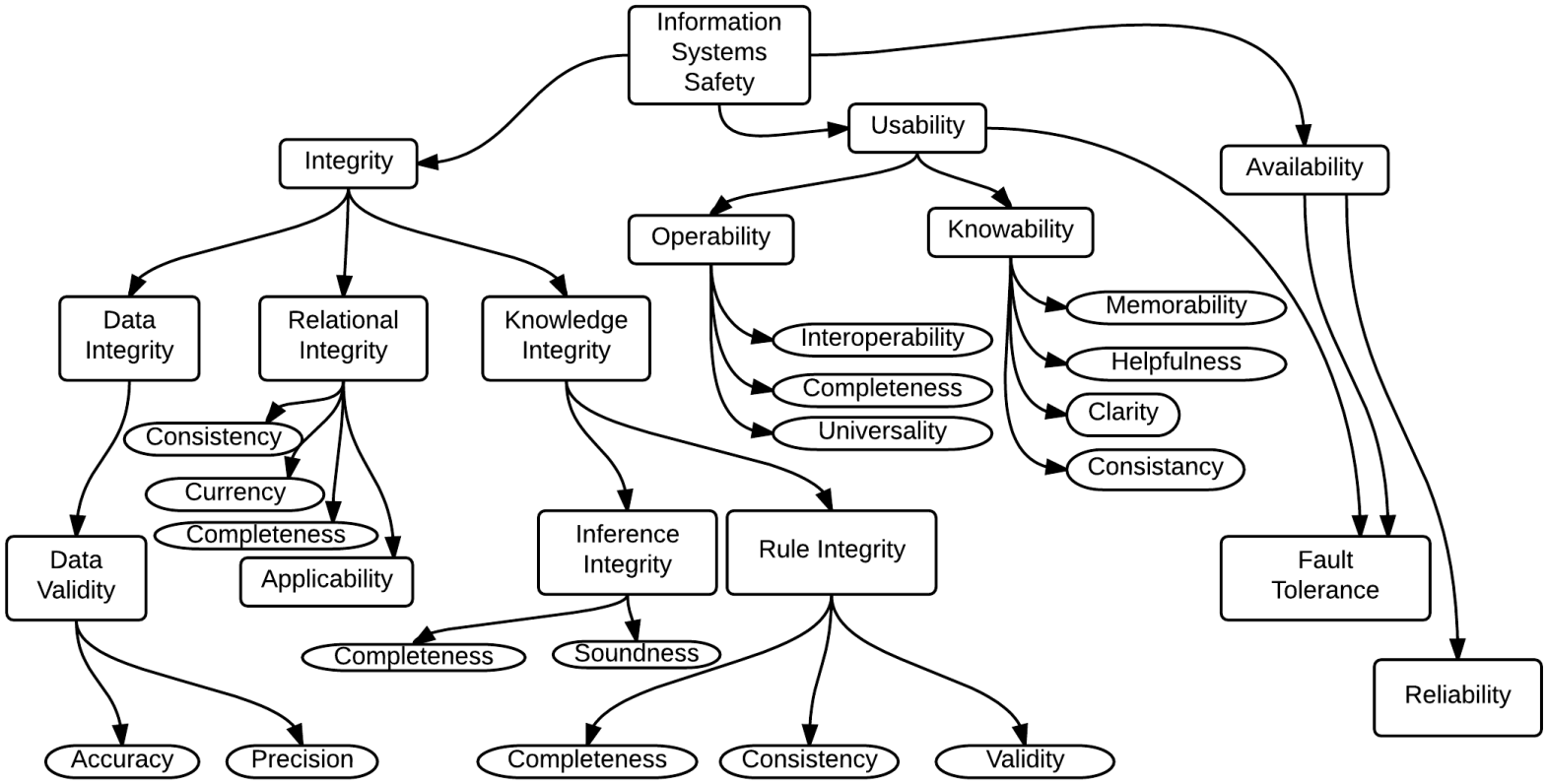

Figure 3.9: The hazard taxonomy we contribute is synthesized from a range of existing taxonomies which provide language to describe integrity [81, 82, 83, 29], usability [54, 73] and availability [59, 60] issues. This triplet of themes is recurrent across much of the CIS safety literature and each is also well represented in a range of incident reporting systems.

The STAMP-EMR model is comprised of a central double control loop that models care providers' care for patients in a CIS. This unit of the model is referred to as the *contextual health care process*. The contextual health care process is subject to the controls imposed by both technical and clinical management. These super-ordinate controllers are modelled in STAMP-EMR with two supplemental and interacting control loops which describe how the clinical administration guides the care process and directs technical management. Because technical management also influences clinical care through the solutions it provides - software, hardware, and technical support for example - it is modelled as an additional controller on the *contextual healthcare process*. It is simultaneously modelled as being controlled by the health care process controller because it is the clinical administration that directs the technical support they employ.

In the STAMP-EMR model's contextual health care process we use human and machine hazard concerns on the peripherals of the CIS software. These concerns are used as classifiers to distinguish between two hazards - human and machine error. These are necessary when using STAMP-EMR to distinguish between the machine and human components of these sensors and actuators as these elements of the model are not more specific. Structural refinement of the model would allow analysts to ingrain such distinctions as different components in the taxonomy rather than relying on qualifying hazard concerns.

Many of the rounded boxes are linked in multi-layered tree structures. For example, the integrity hazard concern is linked to the physical, data, relational, transformational, and knowledge hazard concerns. These deeper elements of the tree structures offer more specificity in the description of the root hazard concern - data integrity for instance is a more specific hazard concern than a generic integrity concern.

Phillips and Gong Nomenclature for Electronic Medical Record Errors

Phillips and Gong proposed a nomenclature to facilitate communication about medical errors which are related to EMRs [105]. Among these authors' concerns was the completeness of the description of errors. These authors proposed a grammar to describe these errors. This nomenclature was composed at the most coarse level of an *error state* and a *precipitating event*. We propose that the singular number of each concept should not be prescribed and that a plurality of both *precipitating events* and *error states* be allowed and encouraged in hazard descriptions. This is more representative of typical accidents and hazards in socio-technical systems [76]. Phillips and Gong's principle concepts are composed of subconcepts. The *error state* is comprised of the *error element*, the *error condition*, and the *error context*. The *precipitating event* is comprised of the *event agent*, the *event task*, and the *event context*. Further, the *event task* is comprised of the *task object*, the *task action*, and the *task parameter*. Again, the singular number of these should and can be

extended to describe the plurality which will be present in described errors. These concepts are summarized in Tab. 3.1.

Hazard Translation Having now established ground terminology and nomenclature for the hazards. We re-express those we have identified using this standardized language. The process of translation proceeds by identifying each element of the nomenclature in the source descriptions of the hazards and restructuring the expression to conform to the nomenclature. When this information is not present, analysts ideally find more complete and/or descriptive sources. Otherwise, they infer these details from a combination of the nuances of the descriptions they have, and their professional experience. The exclusion of components of the nomenclature from hazard descriptions leave *degrees of freedom* in semantics which may produce nebulous descriptions which are difficult to interpret. At the same time, doing so may be a mechanism which can be used to achieve abstraction. Differing re-expressions of less fully specified hazards may also identify new aspects of the system which need to be considered in the context. We provide re-expressions of our identified hazards below.

Hazard 1 Antiglycemic [task parameter] medication doses [task object] that are prescribed [task action] by a doctor [event agent] are not delivered [error condition] to patients. This results in physical harm to patients which could have been prevented and thus constitutes a medical error [Precipitating Event]. This occurs because the CIS software fails to transmit [error condition] the medication orders entered by doctors [event agent] from the medication prescription module [error context] through the system [error context] to the medication administration module [error context] where the order [task object] is observed and acted on by the administering nurse [event agent].

Hazard 3 Antiglycemic [task parameter] medication orders [task object] which are entered [event task] by doctors [event agent] into the medication prescription module [error context] near the end of the day [event context] and use ambiguous relative temporal terms like *tomorrow* [error element] have the potential to be delayed by 24hrs [error element] due to misinterpretation by the administering nurse [event agent] who received [event action] the prescription in the medication administration module [error context] on account of the low *clarity* [error condition] of these expressions [71]

Hazard 4 Antiglycemic [task parameter] medication orders [task object] entered [task action] in one work station [event context] may not be persisted [error condition]

Concept	Subconcepts
Error State (what is wrong)	Error Element (incorrect datum or function)
	Error Condition (what is wrong about the element)
	Error Context (location of element in system)
Precipitating Event (what the user was doing)	Event Agent (user)
	Event Task (activity being performed in system)
	Task Object (involved in task)
	Task Action (what was being done)
	Task Parameters (modifiers)
	Event Context (clinical or administrative context)

Table 3.1: The Phillips and Gong EMR error nomenclature - replicated from [105]

in such a way that they propagate to other workstations [event context] [32] resulting in an instruction set that is lacking in *consistency* across the views [event context] provided of the prescription record [task object] available throughout the system

Hazard 2 Antiglycemic [task parameter] medication over doses occur, causing patient harm, because physicians [event agent] suffer alert fatigue [error condition] and ignore relevant CDS [error context] warnings [error element] on account of the volume [error condition] of unnecessary warnings they regularly dismiss.

Hazard 5 Doctors [event agent] prescribing [task action] antiglycemic [task parameter] medications [task object] are presented by the CDS module [error context] with a ratio of significant alerts to alerts of marginal importance [error element] that is too low [error condition] [92, 85]. This condition highlights compromised *precision* in the displayed data.

Hazard 6 Doctors [event agent] prescribing [task action] anticlycemic [task parameter] medications [task object] are presented by the CDS module [error context] with a volume of information in alerts that is too large [error condition] compromising the *memorability* of the system by making it difficult for the doctors [event agent] to extract the important details from the less relevant context in the alerts [error context] [92, 85]

3.5.3 Hazard Checklist

Hazard checklists are lists of keywords used to spur conversation and objective analysis of the SUI for the purpose of identifying hazards. Hazard checklists address key hazard sources including energy, ergonomics, data integrity, and others. In ISHA, an appropriate hazard checklist is identified or synthesized and then each term listed in the checklist is considered in the context of the SUI to identify possible hazards. Hazard definitions on their own however are of limited utility, and so ISHA analysts rely on RID or any other available source to identify what system characteristics/states may influence the occurrence, detectability or outcome of a hazard. From this information, analysts extract the contextual information including more detailed, granular and specific hazards which influence the probability of the realization of an accident. The hazard checklist execution output supplements the PHL.

Running Example

In appendix A, we provide a hazard checklist synthesized from a number of checklists provided by Ericson [28], as well as from concepts presented in Leveson's [76] System Theoretic

Process Analysis (STPA) method. Using this list, we identify two additional hazards posed by the SUI. The first is inspired by the "maintenance error" item in the checklist. The second is inspired by the "nonexistent/inadequate 'kill switches'" item.

Hazard 7 The system [error element] may be not be *available* due to a ransomware attack [error condition] which infects it during a period when its supporting platform [task object][error context] is not updated [task action] with the most recent [task parameter] software patches [error condition][task object] by the system administrators [event agent].

Hazard 9 The system security modules [event agent] do not sufficiently [error condition] monitor [task action] the process interactions [error element] with the Master File Table [error element] of NTFS based [error condition] platforms [65] (RID).

Hazard 10 Base operating system platforms are not updated with sufficient frequency [112] (STPA - *A control action required for safety is not provided or not followed* [76]).

Hazard 8 A doctor [task agent] does not quickly [task parameter] revert/cancel [task action] an erroneous antiglycemic agent [task parameter] prescription [task object] when using the prescription module [error context].

Hazard 11 A doctor [task agent] can not quickly [task parameter] revert/cancel [task action] an erroneous antiglycemic agent [task parameter] prescription [task object] when using the prescription module [error context] because of the poor *clarity* of the medication management module's [error context] interface [71].

3.5.4 Hazard Mapping

After the base PHL has been identified and supplemented using available RID and the hazard checklist, the hazards in the PHL are mapped to the static and dynamic aspects of the SUI model. Where the model is insufficiently refined, or where elements of the SUI model necessary for the mapping are missing, more refined or complete models are sourced. If they do not exist, the SUI must be further designed before analysis can proceed. Once the design has been extended, the refined models are used in the mapping process.

How refined the models must be is at the discretion of the analysts. At the extreme, any system can be modelled as a single activity in a dynamic model or as a single block in a static model. In these scenarios all hazards are mapped to the single monolithic entities in these two perspectives. This provides little visibility into the nature of the causal factors or what mitigations might be effective; however, additional efforts in design and model refinement take time and involve expenditure. It is therefore to the analysts to determine what degree of granularity is most efficacious for their execution of the method. By mapping the hazards to the static and dynamic SUI models at this stage in the ISHA method, analysts establish a context in which the risk posed by each hazard can be assessed. This understanding is refined during the execution of the UTM phase of the method.

Running Example

For reference in the running example, we aggregate our PHL in Appendix B. These hazards are mapped to the relevant elements of the static (Fig. 3.3) and dynamic (Fig. 3.4) models of the SUI. We present the mapping for our running example in Fig. 3.10 and Fig. 3.11.

3.6 Preliminary Hazard Analysis

The PHA phase of ISHA is comprised of five activities:

1. Preliminary Hazard Prioritization
2. Construction of the UTM
3. SCEM modelling
4. Risk Assessment Codes (RACs)
5. Final Hazard Prioritization

The PHA phase produces four outputs:

1. A subset of the PHL which will be the focus of analysis
2. The UTM for the SUI
3. The RACs for the hazards in the PHL
4. A final prioritization for the hazards in the PHL

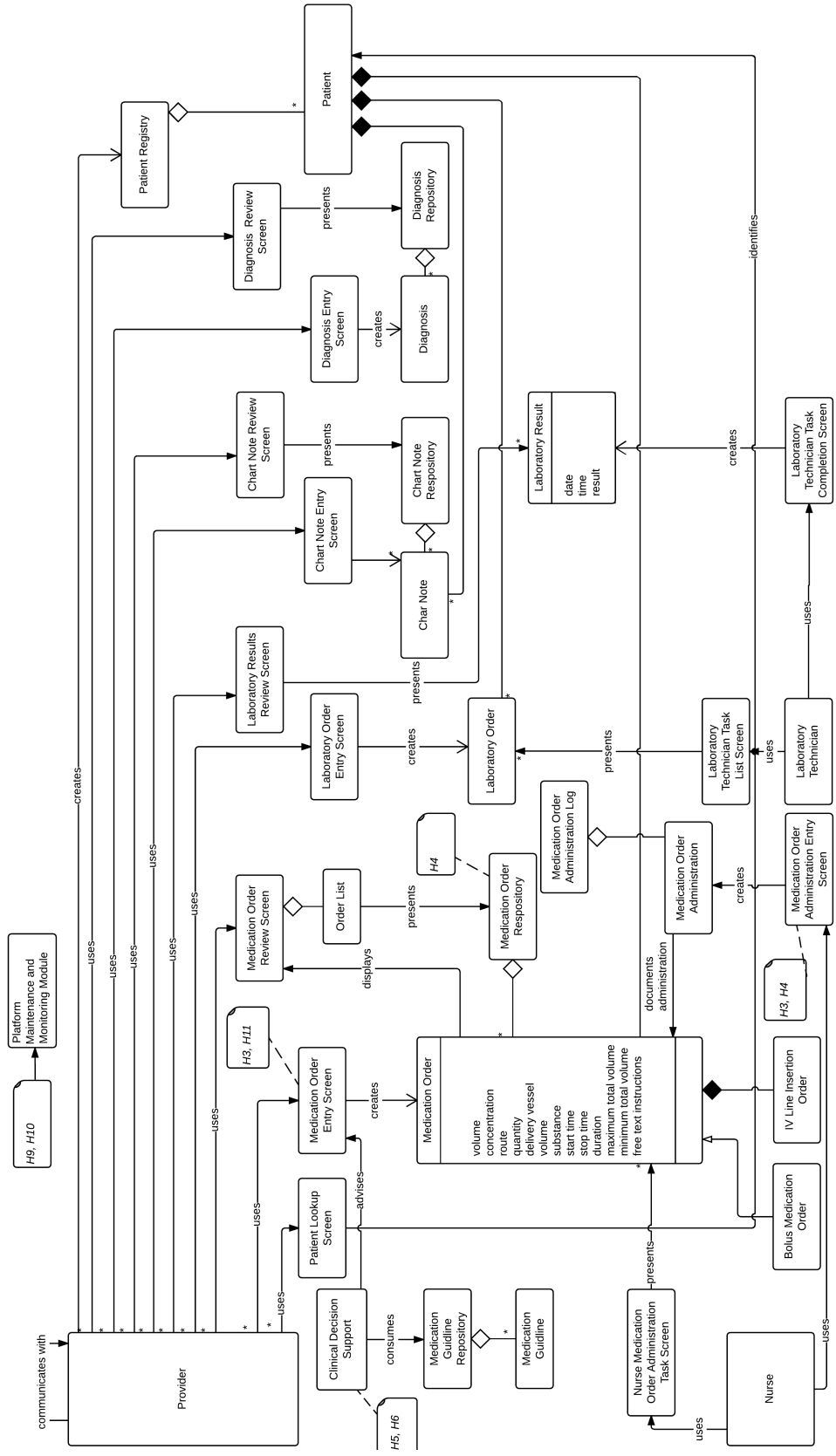

Figure 3.10: A mapping of the hazards identified in the PHL to the static model of the EMR.

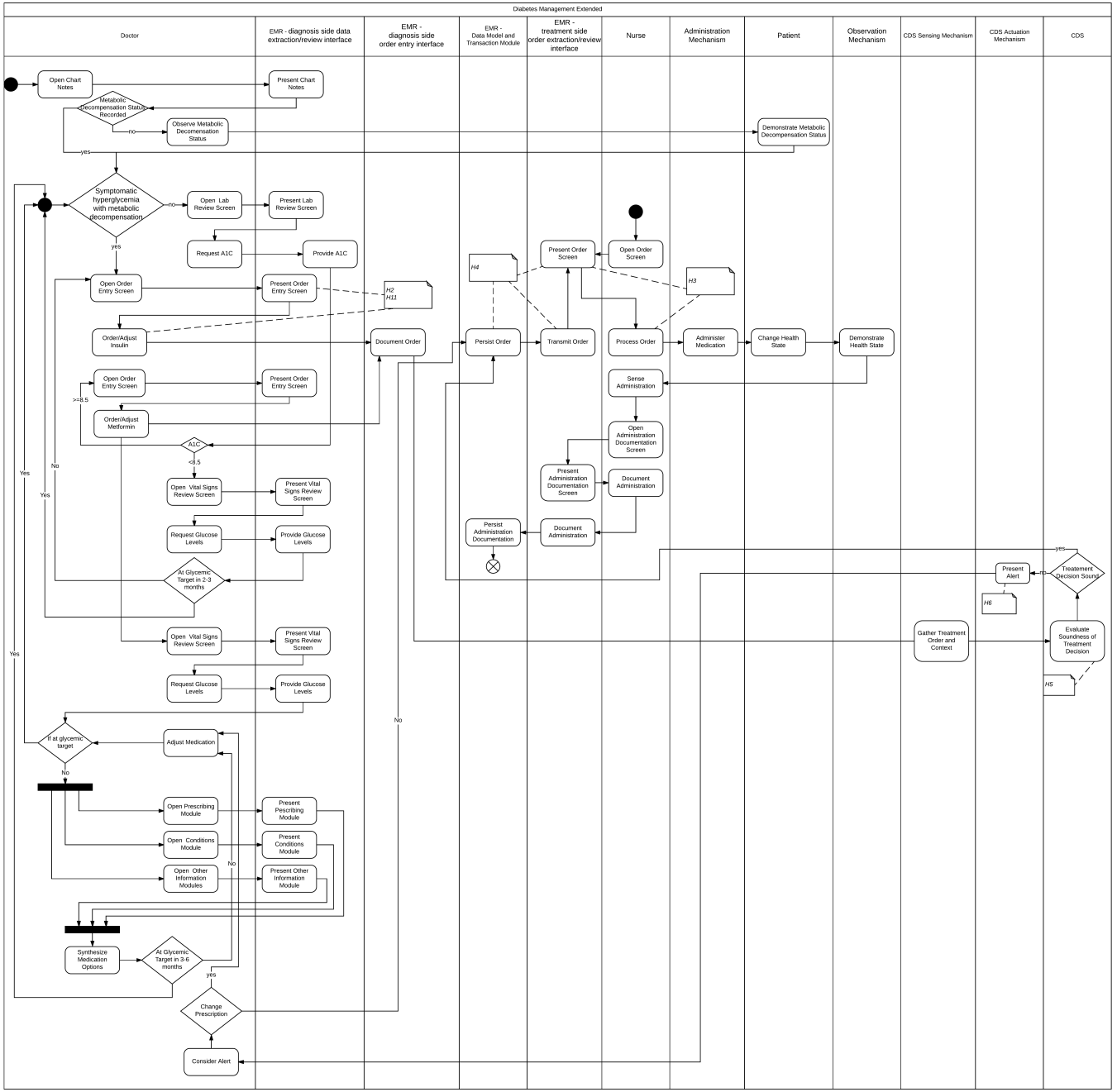

Figure 3.11: A mapping of the hazards identified in the PHL to the dynamic model of the EMR.

3.6.1 Preliminary Prioritization of Hazards

The list of hazards in the PHL artifact generated in the PHL phase of an ISHA may be longer than is manageable with the budget available for the investigation. Analysts will have to trim the PHL to a size commensurate with the budget. At this stage in the ISHA method, there is no mathematical mechanism (e.g., RAC based prioritization) to achieve this reduction in scope; rather, qualitative heuristic methods based on soft skills must be applied to filter the base PHL. This process of prioritization must consider available data about the hazards themselves, the expertise available for the analysis, business needs and other relevant context.

Running Example

In our running example we assert that the study initiator has been asked by his superiors to investigate the safety of the medication order entry and review screens of the EMR in our SUI. A recent story from our local health authority about difficulties in deploying an EMR specifically on account of challenges with prescription interfaces lends credence to this scenario [42]. We assert that responsibility for platform maintenance of the SUI's technology solutions falls out of scope of the study initiator's mandate. Given this assertion, we exclude from further analysis hazard *H7* in the list compiled in Appendix B and its more specific hazards, *H9* and *H10*. To remind the reader, *H7* related to the availability risks posed by randsomware with *H9* representing a lack of Master File Table monitoring, and finally *H10* representing insufficient operating system updating processes.

3.6.2 Construction of the Universal Triangulation Model

The UTM is a control loop based model of the SUI. The base UTM is translated from the sourced static and dynamic models (Section 3.4) of the SUI. The UTM is constructed of units which are expressed as functional components which are associated by the duties in which they exchange information. Further, FRAM [52] framework and STAMP [76] stereotypes are used to characterize these components, associations, and component-association connections. The combination of these UTM units supports the analysts' modelling of the control loop relationships which exist in the SUI. A full introduction to the formalism will be provided in Chapter 5. The process of constructing the UTM consists of five iterative steps:

1. Add Components
2. Add Duties

3. Add Stereotypes
4. Add Constraints
5. Model SCEMs

Add Components

The roles represented in the sourced static view of the model of the SUI provide the components which are represented in the UTM. There is not necessarily a one to one correspondence between the roles in the static model and the UTM; rather, it is at this stage of the UTM construction process that some art is required to recognize the control structures in the static view of the model with the help of the duties expressed in the dynamic view of the model, and of other contextual documentation. Refinement and abstraction are necessary to identify how best to represent the blocks in the static view of the native SUI model in the UTM for the purpose of safety analysis.

Running Example The *Provider*, *Nurse* and *Patient* are three roles in the sourced SUI model. Further to these, there is the *CDS* module which provides a regulating effect on the *Provider*'s prescribing. There are also the *Review* and *Order* screens. The *CDS* module is decomposed into the logical core, the actuating component and the sensing component in anticipation of the control structure which is expected to come to light. Further, the *Diagnostic Side Order Entry Interface* acts as an intermediary between the *CDS* and the *Provider* and so we explicitly represent the components necessary for this interaction. There are also the various repositories which we assert, for our running example, are implemented as a collection of databases. Next, the *Nurse*'s actuation and sensing mechanisms are separated from their cognitive functions allowing analysts to consider errors of intent and action separately. Taking these roles, we can establish a series of components which will be represented in the UTM. We decompose the *CDS* in similar fashion. Our choices of components are listed below:

- Provider
- EMR diagnosis side data extraction/review interface
- EMR diagnosis side order entry interface
- CDS module
- CDS sensing mechanism
- CDS actuating mechanism

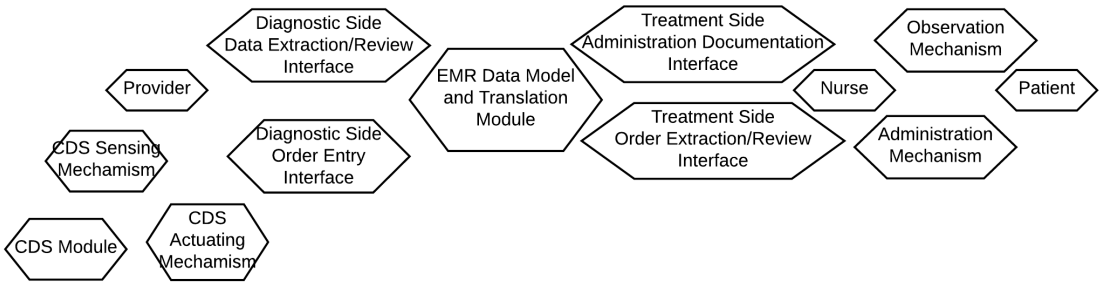

Figure 3.12: A partial ISHA model of the SUI which includes only the components.

- EMR data model and translation module
- EMR treatment side order extraction/review interface
- EMR treatment side documentation interface
- Nurse
- Nurse's observation (sensing) mechanism
- Nurse's administration (actuation) mechanism

Note that we excluded the technical management hazards from our analysis (*H7*) in the *Prioritization of Hazards* phase (Section 3.6.1). Consequently, we do not include the blocks relevant to that hazard or its refinements/extensions (*H9*, *H10*) in the UTM. The resultant partial UTM for our exemplar SUI is provided in Fig. 3.12.

Add Duties

Each activity in the dynamic model of the system is identified as a duty. Additional duties may be extracted from supplemental documentation as well, including the *Concept of Operations* and role definitions established in the *Source System Model* phases of the method. The activity diagram also indicates with swim lanes or an equivalent mechanism the owner of the duties. Once identified, these duties are added to the model and expressed as associations between the components which perform them and the components which consume their output.

Running Example In our SUI, we extract duties from the dynamic model (Fig. 3.4, 3.11) and infer others from the definitions of roles established in Section 3.1. We list the identified duties below suffixing each with the producer and consumer in parentheses. We then add the duties to our model in Fig. 3.13 expressing them as associations.

1. Provide A1C (EMR Diagnosis Side Data Extraction/Review Interface → Provider)
2. Provide metabolic decompensation status (EMR Diagnosis Side Data Extraction/Review Interface → Provider)
3. Provide medication options (EMR Diagnosis Side Data Extraction/Review Interface → Provider)
4. Provide comorbidities (EMR Diagnosis Side Data Extraction/Review Interface → Provider)
5. Provide other patient characteristics (EMR Diagnosis Side Data Extraction/Review Interface → Provider)
6. Synthesize medication decision (Provider)
7. Control Order medication (Provider → EMR Diagnosis Side Order Entry Interface)
8. Control Adjust medication (Provider → EMR Diagnosis Side Order Entry Interface)
9. Persist Treatment Order (Diagnostic Side Order Entry Interface → EMR Data Model and Translation Module)
10. Sense Treatment Administration (EMR Data Model and Translation Module → Diagnostic Side Data Extraction/Review Interface)
11. Control Medication Order (EMR Data Model and Translation Module → Treatment Side Order Extraction/Review Interface)
12. Actuate medication order (EMR treatment side order extraction/review interface → Nurse)
13. Document medication administration (Nurse → EMR Treatment Side Documentation Interface)
14. Persist administration documentation (Treatment Side Administration Documentation Interface → EMR Data Model and Translation Module)
15. Control medication administration (Nurse → Administration Mechanism)
16. Actuate medication administration (Administration Mechanism → Patient)
17. Demonstrate that medication has been administered (Patient → Observation Mechamism)
18. Perceive Administration (Observation Mechanism → Nurse)

19. Document medication administration (Nurse → Treatment Side Documentation Interface)
20. Persist medication administration documentation (Treatment Side Documentation Interface → EMR Data Model and Translation Module)
21. Sense Treatment and Context (Diagnostic Side Order Entry Interface → CDS Sensing Mechanism)
22. Perceive Treatment and Context (CDS Sensing Mechanism → CDS Module)
23. Control treatment guidance (CDS → Diagnostic Side Order Entry Interface
24. Actuate treatment guidance (CDS Actuating Mechanism → Diagnostic Side Order Entry Interface)
25. Sense warning (Diagnostic Side Order Entry Interface → Warning Message)
26. Perceive warning (Warning Message → Provider)
27. Respond to warning (Provider → Response Capture)
28. Actuate Response (Response Capture → Diagnosti Side Order Entry Interface)

Add Stereotypes

In this phase of the UTM construction, stereotypes for the components, duties and association ports are added to the model. The SUI model and other SUI documentation inform which components produce outputs for which other components. The documentation also informs analysts of the various power-over relationships which exist in the model, thus spelling out the STAMP stereotypes played by each of the components. This information is used to stereotype each of the components in the UTM and their associations. Each component is stereotyped once per duty. Further in this phase, we annotate the FRAM semantics of each of the associations by labelling the source and sink ports of the components they connect with the appropriate FRAM port types.

Running Example We provide a visualization of the UTM at the present state of development in Fig. 3.14. The stereotypes are assigned based on the descriptions of the activities in the models of the SUI, the *Concept of Operations*, and the definition of roles established in the *Source Model* phase of this execution of the method. For simplicity, we represent only the first instance of each stereotype in our visual representation of the UTM.

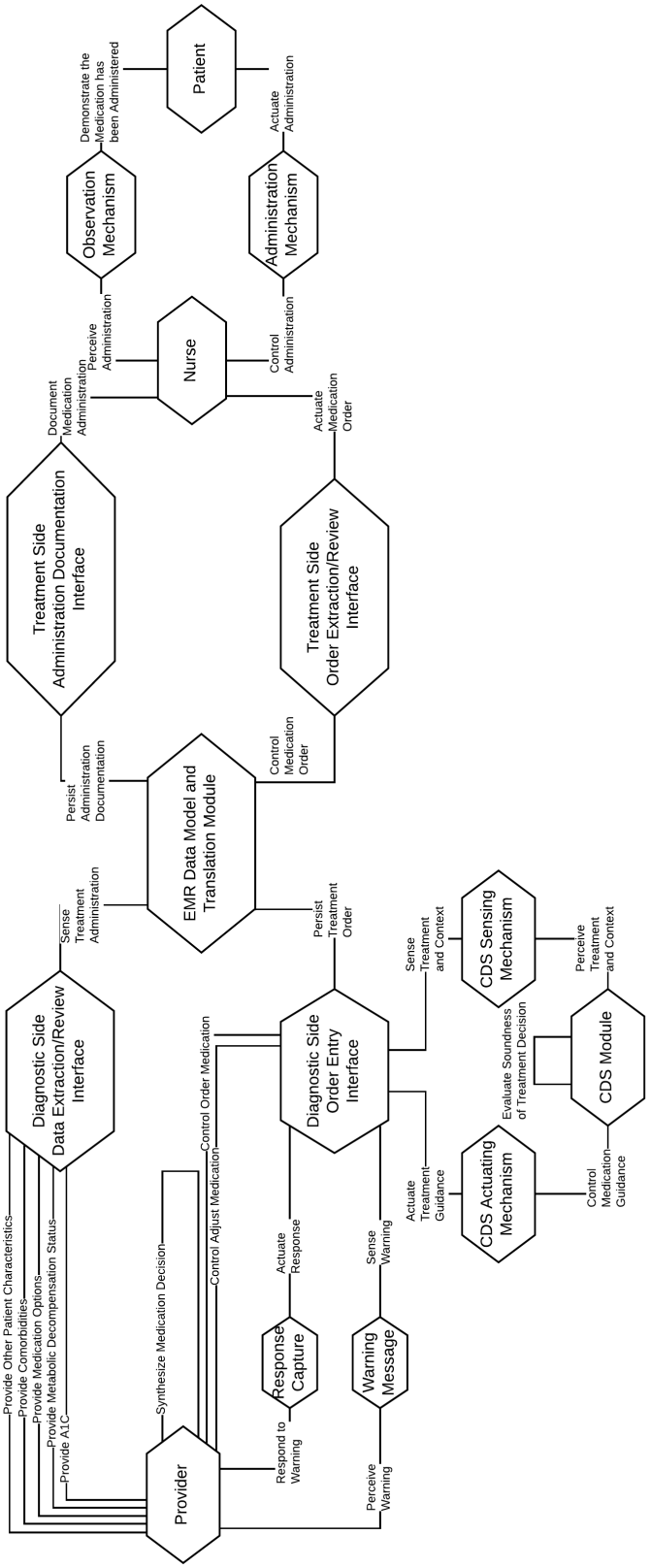

Figure 3.13: The SUI's UTM with duties assigned to components.

Also included in the diagram are the FRAM port pair labels for each association and the association stereotypes. Hollnagel [52] indicates that each component should have a singular port for each of the six inputs and outputs. In our diagram we illustrate components as having multiple copies of the same port. This is for visual simplicity. The underlying model is precisely as Hollnagel expresses; we choose to visualize the model with multiple copies of the same ports to reduce the number of crossing lines in the diagram. In the model, the semantics of the associations are almost exclusively output (O) to input (I) relationships as the inputs are required for each of the central three loops (Provider to EMR to Nurse to Patient). The exceptions lie in the CDS module which is a support module rather than a core module. As the SUI can operate without the support, this input is modelled as control (C). The STAMP stereotypes for both components and for associations were discussed in Section 2.5.6 as were the FRAM port semantics.

Add Constraints

Constraints will be expressed in the requirements as restrictions on the degrees of freedom of components and communications in the SUI. Constraints which are not enforced in the system are only expressed desires, and thus are not structurally represented. For this reason, we incorporate them as annotations on the model.

Running Example In our running example, we identified two constraints in the requirements phase (Section 3.3). We annotate these in our model and illustrate them in Fig. 3.15.

Add Viewpoints

A viewpoint consists of a set of components and duties which form a set of complete *control loops*. Viewpoints can be developed by analysts to help them focus on specific aspects of the SUI and specific scenarios in which the components in the SUI may play different STAMP roles. We will introduce the basic concept of a control loop here, and will then continue with our running example.

Control Loop *Control loops* are a foundational element of systemic thinking. In STAMP, and thus in ISHA, they are comprised of four components, which are related by a set of relations such that each component stereotype (controller, sensor, process, actuator), and each association stereotype (control, actuate, sense, perceive) is included in the loop once.

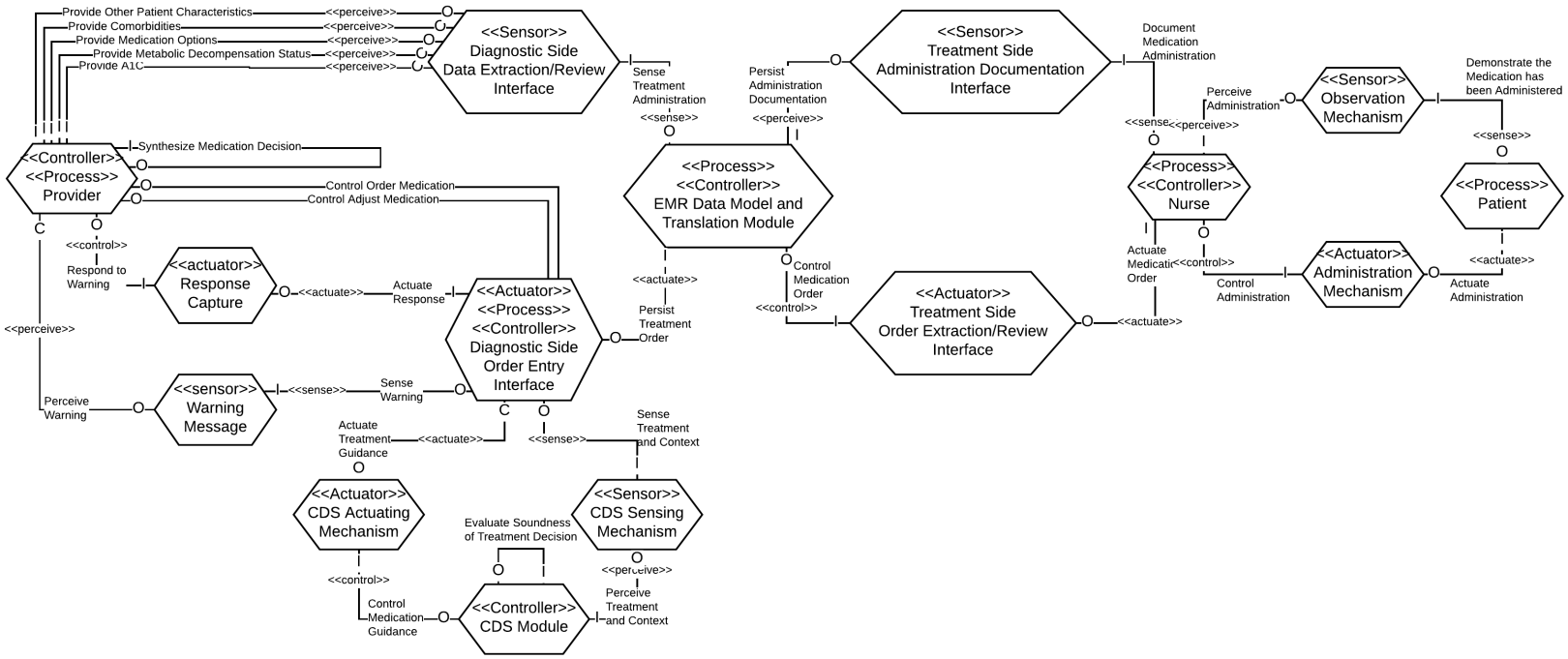

Figure 3.14: The UTM for a diabetes management system in an long-term residential care setting. The UTM includes stereotypes for the components and for the duties between them.

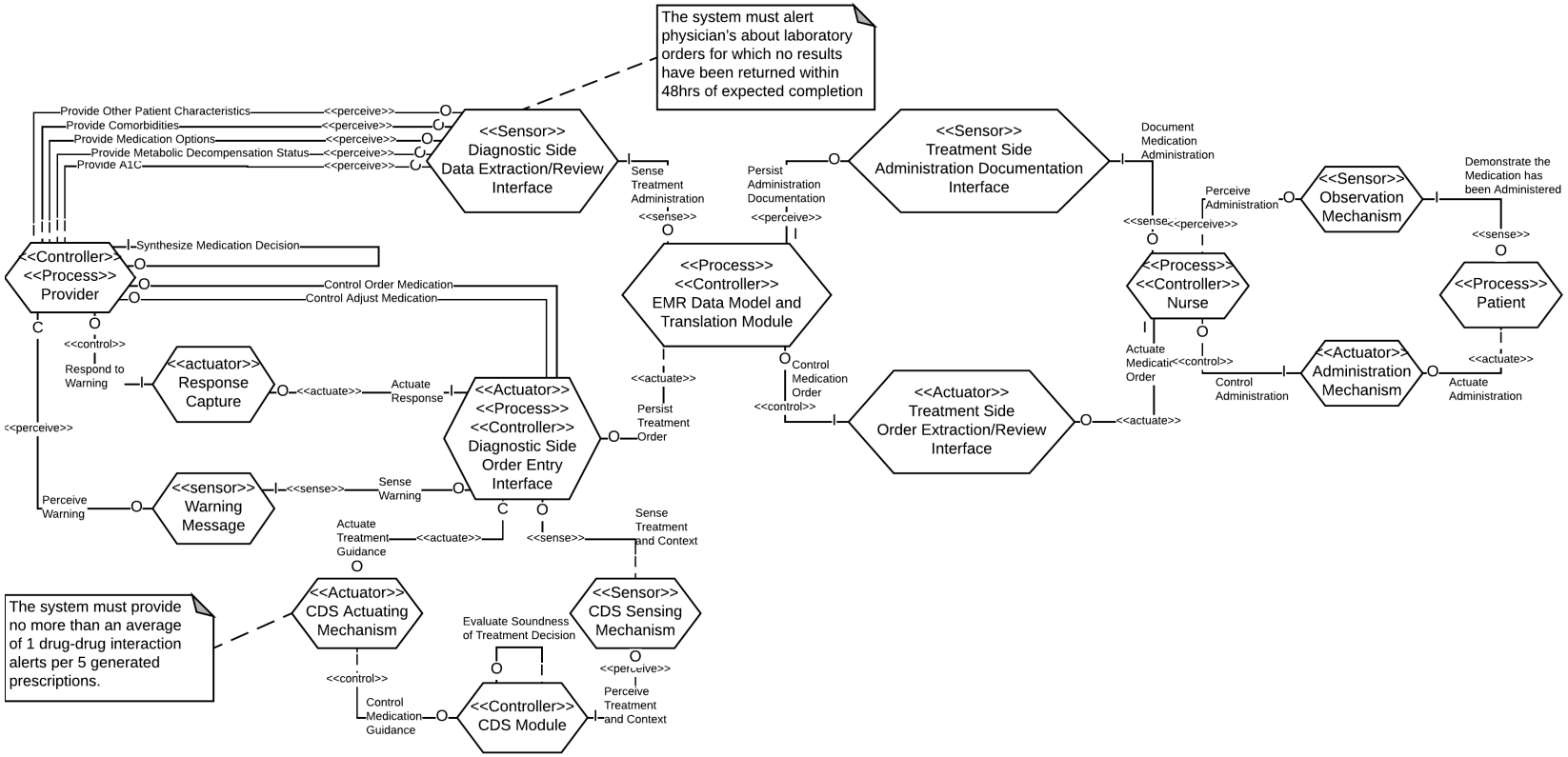

Figure 3.15: The UTM for a diabetes management system in an long-term residential care setting. The UTM includes stereotypes for the components and for the duties between them.

Running Example Taking the UTM in its current state of development as an example, here are three interesting viewpoints:

Viewpoint One:

Viewpoint one collapses the two rightmost core loops leaving only, the Provider, the EMR and the CDS, their peripherals and the related associations. This perspective allows analysts to focus on post visit processes in which the Provider reflects on a patient's case when the provider is detached from the care process. This perspective emphasizes the importance of data quality and appropriate decision support. It does so by reducing consideration of the corrections which might be made to the Provider's process model by downstream components like the nurse and the patient.

Viewpoint Two:

Viewpoint two collapses the Provider and CDS. This allows analysts to focus on scenarios in which the Nurse must act in the absence of the Provider. An example might be when the overnight Nurse identifies an error in a Provider's order - perhaps an off by a factor of ten error in a medication prescription. This viewpoint focuses the attention of analysts on the controls which are in place to manage these situations - none in the given model.

Viewpoint Three:

For viewpoint three we will depart a little from the model as shown in Fig. 3.14. We will assert that analysts have uncovered work flows in which the doctor relies on the EMR to determine a course of action. This could be to leverage a particularly strong CDS feature which might derive answers by mining reliable clinical sources. In such a case, the EMR could be stereotyped as a controller and the Provider as a process. Simultaneously, the data extraction/review interface would be stereotyped as an actuator while the order entry interface would be stereotyped as a sensor. This stereotyping scenario demonstrates how analysts can flip their mental model of which components are "in charge" of the process for situations like the one described where the line between controller and process become blurred. This reversal of stereotypes is a common pattern to consider when analyzing information systems with a control lens.

3.6.3 Safety Constraint Enforcement Mechanism Modelling

The modelling of SCEMs precedes the RAC phase of ISHA as the presence of such mitigations impacts accident risk. SCEMs are the mechanisms which are implemented in the

SUI to enforce the SRs. To identify SCEMs, analysts review the SUI's model and documentation. Analysts first search the documentation for explicitly declared SCEMs and document those that they find. This documentation may involve extension or refinement of the UTM. Next, analysts iterate over the blocks in the static views of the SUI model and the activities in the dynamic views of the SUI model. Each of these entities is assessed in context against the hazards listed in the PHL. If properties of the blocks or the performance of the activities impact the risk of an accident, and the block or activity is also regulated, then the mechanism of regulation is documented as a SCEM in the UTM.

Running Example

In our running example, hazard *H5* indicates that the CDS warns the doctors when they attempt to issue contraindicated prescriptions, sometimes spuriously. This is an example of activity regulation that relates directly to the possibility of the realization of an accident. Consequently, we label the CDS as a SCEM in the UTM. We include this annotation as a note in our model as shown in Fig. 3.16.

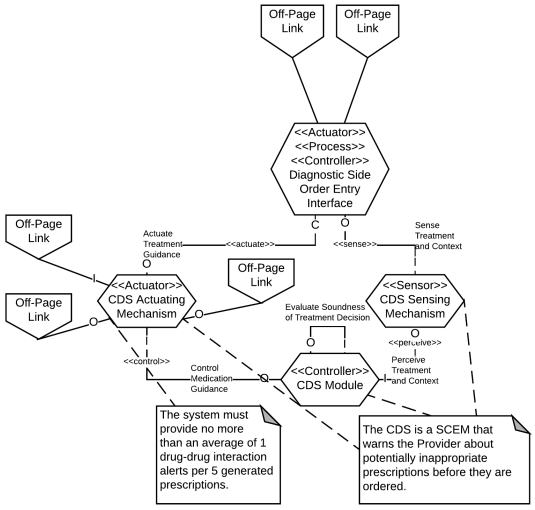

Figure 3.16: The diagram illustrates the annotation of the CDS in the SUI as a SCEM.

3.6.4 Hazard Mapping

Once the base model has been completed, the analysts map the hazards from the scoped PHL into the UTM by relating them directly to the appropriate elements of the UTM if possible, or by refining/extending the UTM as necessary and then mapping them if not.

Running Example

In the running example, we include only the more specific/granular hazards from our filtered hazard list, asserting that the generic hazards are not specific enough to facilitate meaningful mappings. Recall that we excluded hazard *H7* on the basis of the study initiator's mandate. This prompts us to assess the hazards to determine the importance of their inclusion in the UTM. A quick assessment allows us to determine that neither *H9* nor *H10* are appropriate for an assessment of the safety of the prescribing interface from a clinical workflow perspective as they are outside of the scope prescribed by the study

initiator. *H9* identifies the influence of monitoring the Master File table on ransomware vulnerability, while *H10* identifies the role of operating system patches on the same hazard. The annotation of the hazards in the UTM are modelled in Fig. 3.17.

3.6.5 Risk Assessment Codes

ISHA follows MIL-STD-882E's prescription that the significance of assessed risks be expressed using RACs. MIL-STD-882E's RAC is a combination of an ordinal level of severity and an ordinal level of probability [26]. ISHA prescribes that this be supplemented with a detectability score as is done in FMEA [130]. The detectability score indicates the likelihood that a latent error will be observed before it realizes into an accident. In ISHA, each of the combinations of the three metrics are assigned an ordinal risk score similar to those provided in MIL-STD-882E [26]. This explicit assignment circumvents the criticism leveled by Shebl against the validity of the FMEA Risk Probability Number (RPN) [120]. Once the scale has been established and documented, each hazard is assigned an occurrence and detectability score while at the same time the gamut of potential outcome severities is always considered. In the traditional risk literature, including MIL-STD-882E, the position has often been taken that hazards lead to known outcomes. We are skeptical of this position in general, but especially in medicine where timing and dosage can have a significant impact on the effectiveness or deleteriousness of a treatment. It is for this reason that with ISHA, it is prescribed that all outcome severities are considered in the analysis of each hazard. The assignment of the occurrence and detectability scores is based on available data including RID, expert opinion, available statistics, and other available sources. It is also made with consideration of the design of the SUI including identified SCEMs.

The risk scale proposed by MIL-STD-882E includes four levels - high, serious, medium and low. These same risk levels could be used in ISHA. Similarly, MIL-STD-882E provides a four level scale of outcome severity which could also be used in ISHA. The six level scale of priority used in MIL-STD-882E could be used as the occurrence scale in ISHA. Detectability scales are less widely published, but one such scale is provided by Spath [128].

One challenge with the use of RAC scores is that defensibly assigning the risk triplet to hazards can be challenging. This is due to the likelihood that less than optimal data will be available to make the assessments; further, in the absence of data, analysts are forced to rely exclusively on their judgment. This is one of the motivations behind the assignment of the cross functional team with a breadth of skills for the analysis. Further compounding this challenge is the often unknown or unexpected interaction between the various functions or hazards in the system. To mitigate this latter challenge, one process by which RACs for an ISHA might be assessed would be through the use of a hazard matrix. By considering

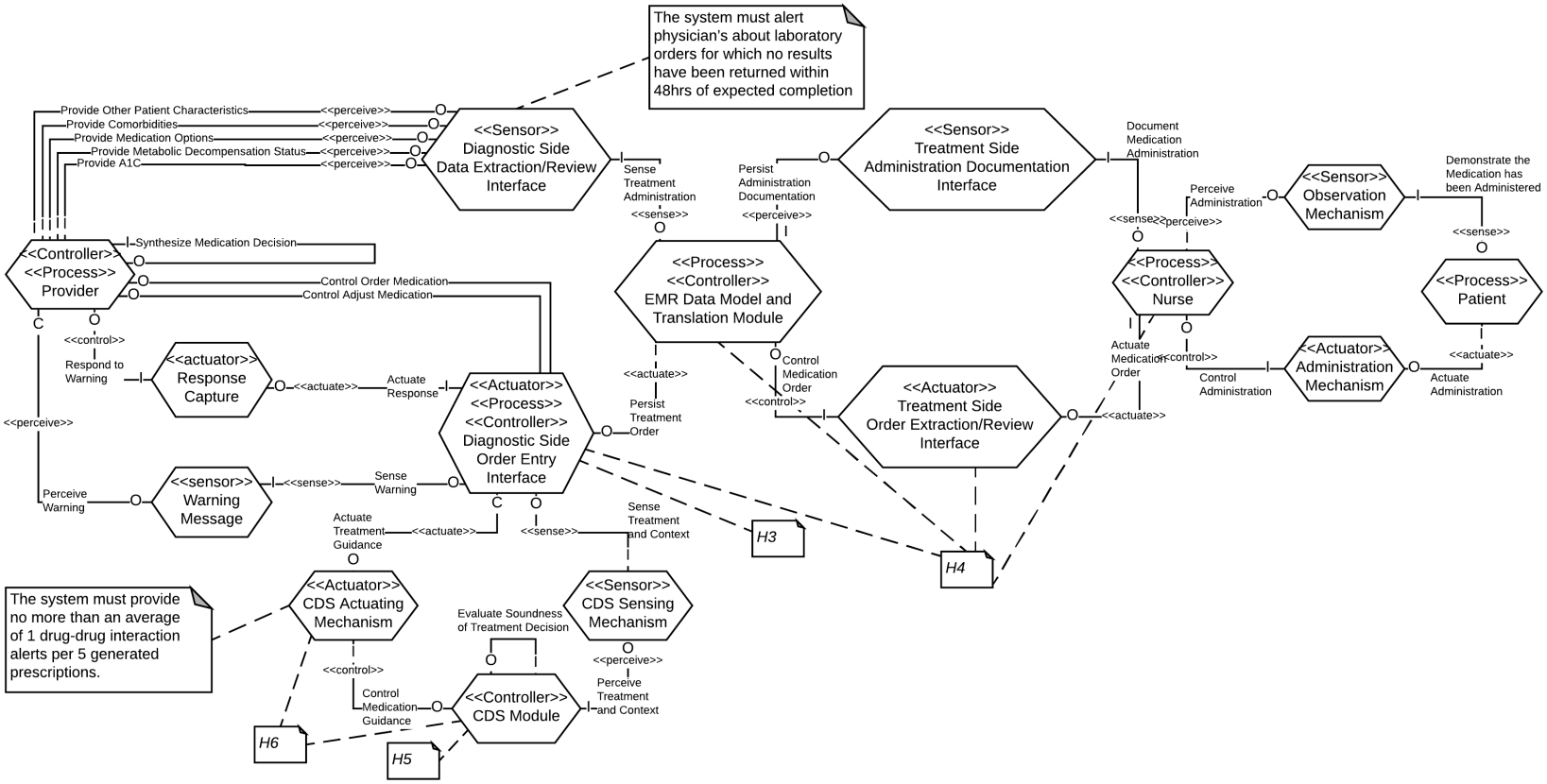

Figure 3.17: The UTM for the running example including the hazard annotations.

the adjacency of hazards and potential correlations and influences analysts may be able to arrive at more realistic estimates for the risk categorizations. This is also formally addressed in ISHA as part of the triangulation phases of analysis (Sections 3.10, 4.9).

Running Example

We tabulate our scales for severity, occurrence, detectability and risk as well as our risk matrix in Appendix C. We use Spath's [128] scale for detectability (Tab. C.1), the MIL-STD-882E probability scale for occurrence (Tab. C.2), the MIL-STD-882E scale for severity (Tab. C.3), and finally the MIL-STD-882E scale of risk (Tab. C.4). We use the aggregate of these scales to develop our risk matrix (Tab. C.5, Tab C.6, Tab C.7, and Tab C.8). The scale definitions used do not directly line up our domain of application. This will be the case when relying on externally defined scales. A more careful analyses would fully adapt the applied scales before use. In our current application, we choose scales that are clearly defined for a loosely related domain - military safety - and rely on some art to negotiate the gap to adapt classifications of risk factors to the long-term residential care context rather than engaging in a full adaptation. An alternative approach would be to take Derosier's Health Care Failure Mode and Effects Analysis (HFMEA) scales [24] and use these as they were developed for the domain of application. These scale however are not as well defined as those in the MIL-STD-882E standard. We choose instead to select the scales with the stronger definitions for our demonstration of the method, but do not concern ourselves with the gaps which arise due to the cross domain application of those scales.

In our running example, we identify three high level hazards included for analysis (*H1, H2, H8*), and five related hazards that were identified based on this initial set (*H3, H4, H5, H6, H11*). We excluded hazard *H7* and its more specific hazards, *H9* and *H10*, based on the mandate of the *study initiator*. To remind the reader, *H7* related to an availability risk to the SUI resulting from a randsomeware attack; *H9* was the monitoring of the Master File Table, and *H10* was the update process for the EMR's platform (operating system). Using the established scales, we first estimate the risk triplets for the derived/specialized SUI hazards - note that each level of severity is always considered. We then infer that the more general hazards have at best the highest risk scores in each of category of the risk triplet of their related hazards. We tabulate the results in Tab. 3.2. To exemplify, if a high level hazard had two related more specific potentially contributing hazards with triplets scoring O:2 D:3 and O:3 D:1 in the Catastrophic category of outcome severity then the "parent" hazard would have a score of O:3 D:3 in the Catastrophic category.

We include the RACs which are identified from the risk matrix which is included in Appendix C in the tabulation. We rely on a combination of the SUI's RID, and design as well as expert opinion - that of the team of analysts - to ascribe the occurrence and detectability

		Catastrophic (S:4)	Critical (S:3)	Marginal (S:2)	Negligible (S:1)
H1	Prescription not transmitted	O:5 D:5 R:High	O:5 D:5 R:High	O:5 D:5 R:Serious	O:5 D:5 R:Serious
H3	Relative temporal language	O:3 D:3 R:High	O:3 D:3 R:High	O:5 D:3 R:Serious	D:3 O:5 R:Medium
H4	Inter-station order propagation	O:2 D:5 R:High	O:2 D:5 R:Serious	O:2 D:5 R:Medium	D:2 O:5 R:Medium
H2	Over or underdose due to alert fatigue	O:5 D:2 R:High	O:5 D:2 R:Serious	O:5 D:2 M:Medium	O:5 D:2 R:Medium
H5	Alert noise to signal ratio	O:5 D:2 R:High	O:5 D:2 R:Serious	O:5 D:2 M:Medium	O:5 D:2 R:Medium
H6	Conciseness of alert messages	O:2 D:2 R:High	O:2 D:2 R:Serious	O:2 D:2 R:Medium	D:2 D:2 R:Medium
H8	Doctor fails to cancel an order for an antiglycemic prescription	O:5 D:5 R:High	O:5 D:5 R:High	O:5 D:5 R:Serious	O:5 D:5 R:Serious
H11	Doctor fails to cancel an order for an antiglycemic prescription on account of poor clarity of order cancellation workflow in the CPOE	O:5 D:5 R:High	O:5 D:5 R:High	O:5 D:5 R:Serious	O:5 D:5 R:Serious

Table 3.2: The table of hazards for the running example of ISHA

for the hazards across their outcome severities. Hazard interactions are explicitly considered in the triangulation phase of the ISHA method (Sections 4.9, 3.10), and the understanding of these interactions are incorporated by analysts into the RACs via their selection of occurrence, and detectability scores.

Hazard 1 *H1* is the generic hazard that as the consequence of a prescription not being transmitted, an under or overdose of a medication will occur. There are a variety of mechanisms which can produce this outcome. Hazards *H3* and *H4* are two. When considering the interaction of this hazard with other hazards, we argue that in a generic system, interactions could exist between alerts and transmission failures. Alerts could be presented for draft orders which have not been executed, or sent, or saved. We assert however that in

our SUI that this functionality is not present. An "interaction" could also be recognized between *H1* and *H8* in that *H8* may in some circumstances be a specific instance of *H1*. For our current analysis we choose to consider the hazards independently, but such similarity in a more extensive analysis might warrant closer attention.

Hazard 3 *H3* identifies the influence of using relative temporal language in prescriptions on the potential for missed doses of medication (Hazard *H1*). We assert that in our SUI the language used in the orders varies. The ways in which the orders are written varies as well. Some orders use free text while others will be fully codified. This variability arises in our SUI from clinical needs. We consider a high base level of detectability for all orders on aggregate as this hazard is easily observable by the nurses who consume the orders (the orders are written instructions which are read). Further, we acknowledge that the use of relative temporal terminology will not always be recognized as hazardous and would thus escape detection. Further still, the hazard could not be easily identified by computational means - computing free text is a challenging problem. From this information we synthesize that there is a "Moderate likelihood of detection and correction" of the hazard before it realizes into an accident. We assert based on reasoned judgment that the detectability of the hazard in this case is independent of both the outcome severity and the frequency of occurrence of the hazard.

We combine two further sources of data to determine occurrence scores - first, we inspect existing medication orders in the SUI to determine that a small fraction (3%) of orders use relative temporal language, and that all of these orders do so in free text fields of the prescriptions. Further, we rely on RID from the SUI - an incident reporting system - for data which indicates that complications from ketoacidosis have only been observed at a rate of 3 in 54750 patient days (2 incidents for a 30 bed facility in 5 years). Of these incidents, only one of the three was related to the issue in question. Further the severity of all three incidents was *marginal* as per the definition in the severity scale (Tab. C.3). Further, we assert tacit knowledge that the realization of *H1* of which *H3* is a specific instance and *H8* to which *H3* is related into accidents with marginal and negligible outcomes is common in this setting. Adverse Drug Event (ADE)s are known to be under reported, and it stands to reason that those with minimal impact on patient health would be frequently dismissed as fluctuations in normal health. From this information we synthesize that accidents with marginal and negligible outcomes are "frequent", but that accidents with more severe outcomes are "occasional".

Inspection of the apparent nature of *H3* reveals no apparent interaction with other hazards beyond those discussed in the paragraph about *H1*.

Hazard 4 *H4* identifies the possibility that a failure of orders to be propagated to all interfaces may contribute to an overdose, or underdose. This may arise from the failure by the provider to transmit an initial order (*H1*), but it is also possible that it could arise from a Provider's inability to cancel an order (*H8/H11*).

In the context of *H1*, the detectability of this hazard is asserted to be low. Nurses working on the ward are unlikely to notice that orders are not available on all stations. Further, computational detection of such inconsistencies is also complex as there are a myriad of potential underlying causes including networking issues, race conditions in the software such as faulty semaphore implementation. Ideally such a hazard would be mitigated by design, as distributed computing theory provides algorithms which solve these issues; however, this is not necessarily practicable. In a realistic investigation like the one we assert here, a healthcare organization may be constrained by politics or budget to continue to use a legacy system with known software faults supported by an unresponsive vendor who will not correct the problem in a timely fashion [29]. From this knowledge we assert that there is only a "remote likelihood of detection and correction" of such a hazard before it realizes into an accident.

The occurrence of this hazard is more difficult to assess. We assert that in this study, we reviewed incident data in the facility's incident reporting system and that no report ever explicitly identified a situation in which this was alleged to have occurred. However, some reports describing situations with the prescription of other medications raised ambiguous concerns about the consistency of the order list available to nurses at different times of day - the times of day did not logically align with the time that rounds were performed by Providers, and so some suspicion remained that this issue may be occurring as a result of an as yet undiagnosed software flaw. From this information we synthesize the expectation that the occurrence of the hazard is "Remote", we choose remote over "Occasional" because the data on which we based our decision never identified this issue arising in the anti-glycemic prescription workflow, but also in knowledge that a common prescription interface is used for these prescriptions and those described in the incident reports. We assert that the relationship between the occurrence and detectability of this hazard is negligible, as is their relationship to severity.

Inspection of the apparent nature of *H4* reveals no apparent interaction with other hazards beyond those discussed in the paragraph about *H1*.

Hazard 2 *H2* is the generic hazard of alert fatigue which arises in, and which may be qualified in, a variety of ways. Two examples of this hazard are provided by *H5* an *H6*. As discussed above we assert that the potential interactions between *H2* and both *H1* and *H8* which might exist, do not in the SUI.

Hazard 5 *H5* identifies the problem of alert fatigue which arises when the ratio of important medication alerts to unimportant ones is too high. We assert that the detectability of this hazard will be relatively high as Providers will quickly become frustrated if they are overwhelmed by alerts. The factor does however have a sliding scale of detectability depending on the severity of the fatigue which is induced. We assert that through Provider interviews, we have roughly estimated the expected detectability of this hazard to fall in a range such that there is a "High likelihood" that it would be detected and corrected before it could contribute to the realization of a hazard into an accident. We assert that the occurrence of fatigue should it occur, will impact roughly 20% of prescriptions yielding a "frequent" occurrence.

We assert that there is a negligible relationship between *H5* and both *H1* and *H8* beyond what has already been discussed.

Hazard 6 *H6* identifies the information overload suffered by Providers who are alerted with an excessive volume of information with presented alerts. We assert that the detectability of this hazard will be high as Providers will immediately recognize if the provided information is consumable during their regular workflow or not, however correction of the problem will be dependent on the clinical information support team. They are known to be overwhelmed with a long TODO list much of the time. We thus temper our detectability score and assert that the hazard has "High likelihood of detection and correction" before it contributes to the realization into an accident. Where occurrence is concerned, there is an expectation that alerts will be triggered for at most 1 in 5 prescriptions. Further, metformin and insulin are by far the most commonly prescribed antiglycemics. The Provider's familiarity with the drugs will thus reduce the likelihood of a verbose warning. This leaves only the more exotic anticlycemics which are used with greatly diminished regularity. Synthesizing these details, we assert a "Remote" likelihood of the realization of this hazard into an accident.

We assert that there is a negligible relationship between *H6* and the other hazards beyond that which have already discussed.

Hazard 8 *H8/H11* identifies the possibility that a doctor fails to cancel an order with the more specific hazard indicating that the mechanism of the failure is that he/she fails to understand the workflow in the CPOE to do so. We assert that detectability of this hazard in general terms is low. This is in opposition to the singular case study we have found by Horsky on the topic and our two follow on studies which suggest in hind-sight that a singular such incident was potentially predictable [54, 142, 143]. The argument we make here is that building a general algorithm to detect user confusion based on interface interaction

across all prescribing behaviour is much more complex that identifying the relatively small number of peculiar choices made by the physicians in Horsky's case study. Provider's may not recognize that they have not cancelled their orders if they are confused by the interface they use to enter them. We assert that there is only a "moderate likelihood" that this difficulty would be recognized by the physician as sufficiently dangerous to warrant attention before it contributed to the realization of *H11* into an accident. We also assert that the our SUI captured only minimal usage metrics which could be used to automatically detect the hazard and did not provide a real time warning system to provide feedback to either the physician or any other members of the care team to address the arising issue.

Where occurrence is concerned, we assert a review of prescription cancellations in the log of the EMR which indicate that cancellations are issued on about 1 in 100 anti-glycemic orders before the orders expire. This yields a low frequency of cancellation in the context of general prescribing when measured in patient days, but a relatively common activity when measured against the lifespan of the EMR itself. Synthesizing these details, we choose to assign the "probable" occurrence category. We assert that the relationship between the occurrence and detectability of this hazard is negligible, as is their relationship to severity.

3.6.6 Final Hazard Prioritization

Once assigned, the RACs may be used to prioritize the analyzed hazards. Any prioritization method deemed reasonable by the analysts may be used in the application of the ISHA method. The traditional approach to prioritization promoted in FMEA and in MIL-STD-882E is the RPN, but as we discussed earlier, the validity of the RPN has been criticized by Shebl [119, 120] on several fronts. Our approach of explicitly assigning risk scores to risk triplets instead of multiplying ordinal scales addresses many of Shebl's concerns. This however, only puts analysts in a less compromised position to perform the final prioritization. The RACs do cluster hazards by importance and thus give analysts a sense of which ones pose the greatest threat the safety of the SUI, but analysts should not confused this with the order in which they should tackle the mitigation of hazards.

Consider for example that a single mitigation technique may impact multiple hazards. Multiple mitigations could be designed largely independently of any certainty of their impact and a secondary ISHA could be performed to prospectively assess the impact of each mitigation to decide which to undertake. This approach to mitigation is independent of hazard prioritization, but nonetheless requires understanding the risk categorization of the hazards for the assessment of mitigation impact. At the same time however, such an approach may seem somewhat "blind" from an engineering approach. Tackling a specific subset of higher risk hazards from the angle of one of the risk triplets may be more desirable and this may

or may not be effective in mitigating risk. The impact may instead be to simply shift the risk to some other part of the SUI.

Running Example

In our running example, rather than promising an optimal approach, we assert a number of realities about our SUI. A reasonable budget is available for mitigation of hazards, but a number of confounding factors constrain the approaches which can be taken to address the problems which have been identified in the SUI. As a consequence, rather than indicating that we will prioritize the hazards discussed so far, all the hazards will be tackled as best as possible within the constraints under which the system exists. This will be further discussed in the recommendations section.

3.7 Event Chain Analysis

In the ECA of ISHA, the safety requirements identified in the Source Requirements phase (Section 3.3) and the mitigation of each of the hazards identified in other phases of the method are iteratively analyzed using an event tree based method. The failure of the SUI to satisfy each requirement is taken as a top level event. All of the events which can be identified by the analysts which might lead to this top level event are then mapped into a hierarchical tree structure with the SUI's failure to satisfy the safety goal in question as the tree's root. This activity is recursively performed at each level of the tree identifying event chains which can lead to the top level failure. The occurrence of each event in the tree is identified as a derived hazard. These are then assembled into a supplementary hazard list which is mapped to the UTM in the hazard triangulation phase. This phase is completed by determining RACs for the identified hazards as was done for the PHL in the PHA phase of the method.

3.7.1 Running Example

In our running example, we demonstrate the process using *FR1*. The requirement demands functionality for recording and retrieving demographics. This *FR* is a safety requirement as it influences the chance of wrong patient treatment.

At a high level, there are in fact two requirements:

FR1a The system must support recording new demographics

FR1b The system must support retrieving existing demographics

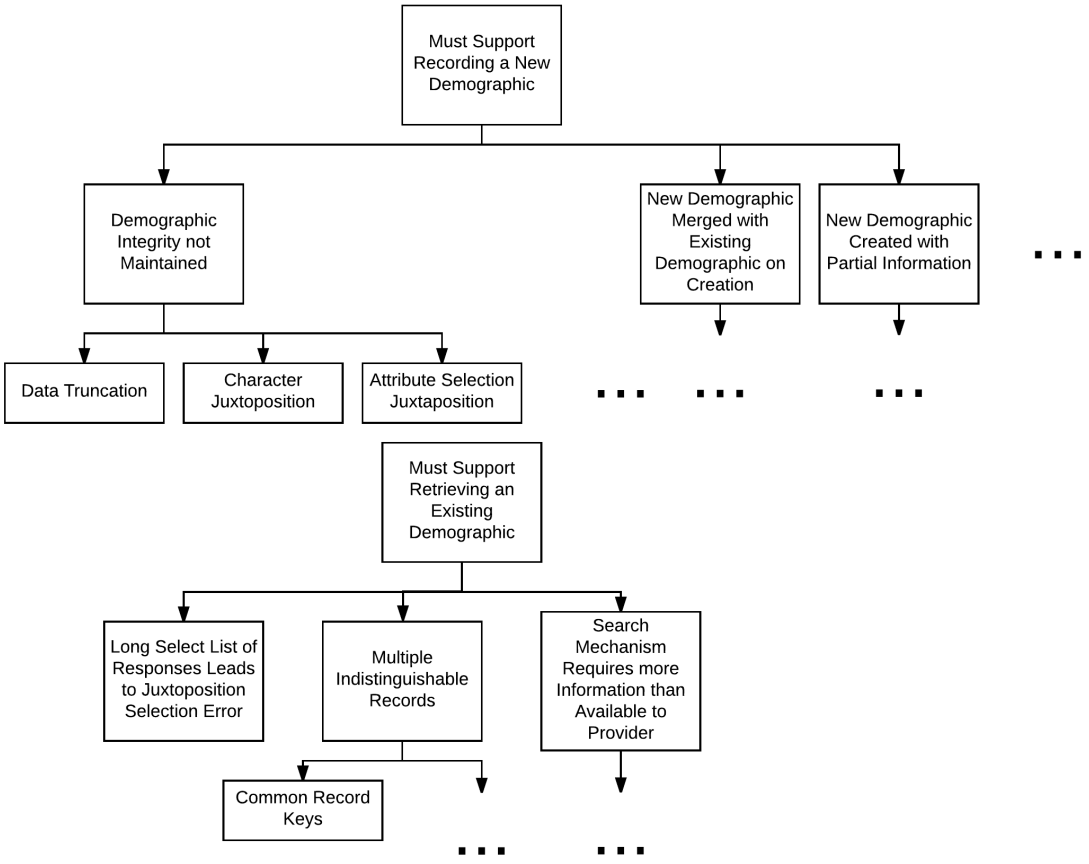

Figure 3.18: An ECA tree for the running example of diabetes treatment in a long-term residential care setting.

A partial ECA tree which proposes a few starting potential hazards is presented in Fig. 3.18. An in depth analysis of the demographics module of the EMR would be undertaken to determine the RACs for the identified hazards.

3.8 Component Fault Analysis

In the CFA of ISHA, analysts use reliability analysis to assess the consequence of component failure in the context of the SUI. The failure of each component in the static model is considered. It is important to consider context and not simply the failure of individual components as many accidents occur as a consequence of interactions. Nonetheless, it is important to assess component reliability as the failure of components in some circumstances can pose a hazard. FMEA is one method which could be used here, but others like Fault

Tree Analysis (FTA) may be effective as well. The output of this phase of the analysis will be a supplementary hazard list whose individual hazards will be mapped to the UTM in the main hazard triangulation phase. As the methods for CFA are well documented by others [28, 130], we do not address the execution of the component fault analysis further here. We only add that the analysis is used to generate RACs for the identified failures.

3.9 Process Fault Analysis

In the PFA of ISHA, analysts use flow analysis to determine the consequence of deviation of the SUI process from design intent. This phase of the ISHA method focuses on the dynamic behaviour of the SUI rather than the nature of its static components. HAZOP may be an effective method of performing a flow analysis, though other techniques for dynamic system analysis may prove equally effective. The output of this phase of the analysis will be a supplementary hazard list whose individual hazards will be mapped to the UTM in the main hazard triangulation phase. This phase is completed by determining RACs for the identified hazards as was done for the PHL in the PHA phase of the method. As the methods for PFA are well documented by others [69, 111, 76, 52], we do not address the execution of the fault process fault analysis further here. We only add that the analysis is used to generate RACs for the identified failures.

3.10 Hazard Triangulation

In the hazard triangulation phase of ISHA, analysts supplement the UTM with the additional hazards identified via the ECA, CFA and PFA. The hazards identified in each are also cross validated. As the ECA is based on the requirements, it should identify all hazards posed by the SUI. The completeness of the ECA will also be reinforced by the incorporation of the requirement that each identified hazard be mitigated. This was mentioned in the description of the Source Requirements phase (Section 3.3). If there are hazards identified in the CFA or PFA which are not found in the ECA, then the ECA must be revisited to understand why and to strengthen the analysis. Further, each of the ECA hazards should be identified in one of the CFA or PFA. If they are not, then these analyses must be revisited to understand what weakness in their execution lead to the exclusion. As there is a clear division between the CFA and PFA across a statics/dynamics axis, the hazards expressed in one may not be present in the other. A cursory review of the interplay across the two analyses is nonetheless important and prescribed as it can provide a further sanity check of the execution of each of these analyses.

Lastly, the results from this investigation is compared against the PHL to identify if any issues identified by the PHL that have not been identified in these analyses (ECA, CFA, PFA). If such hazards are identified, the reason(s) for their absence must be investigated. If the reason(s) identify a weakness in execution, documentation, or other resolvable concern, then this must be revisited in order to explore what other concerns may have been missed. The linkage between the requirements and hazards here is used to argue the completeness of one aspect of the completeness of the safety analysis as will be further described in the next section.

3.11 Assurance Case Construction

As discussed in Chapter 2, an assurance case is composed of claims, arguments and evidence. Claims are made and then arguments are constructed on the basis of evidence to defend them. In this phase of the ISHA method we construct an assurance case about the safety of the SUI. Analysts using ISHA identify the requirements for the SUI in the *Source Requirements* phase. The requirements spell out a series of SRs that the SUI must satisfy. The claims that the SUI satisfies these goals are incorporated into the assurance case as subgoals of the overall system safety goal.
The prerequisites of the assurance case construction phase are:

- the system requirements
- the evidence to support safety claims
- the risks of system hazards

This phase of ISHA is divided into three sub-phases:

- identify safety goals
- construct an argument
- support the argument with evidence

3.11.1 Safety Goals

The highest level safety goal of the assurance case is the safety of the SUI. We use the term safety to refer to the relative benefit of the system when compared against the risk its hazards pose. This goal must be satisfied for each environment of operation [109]. Further, each of the following sub-goals must also be satisfied in each of those environments of operation:

- the safety requirements are valid (validation)
- the design implements the safety requirements (verification)
- the design is faithfully implemented (verification)
- the hazards identified in the PHL are mitigated (verification)
- the hazards identified in the triangulation are mitigated (verification)

Running Example

The failure to satisfy any of the other requirements/constraints can result in physical injury to a patient, and so each of these remaining requirements/constraints is included in the assurance case as a sub-goal.

3.11.2 Construct the Argument

ISHA uses a template based assurance case to demonstrate the completeness of the analysis. By formulating the analysis results in this way, analysts are provided a further opportunity to understand the weaknesses in the analysis they've performed. The triangulation of the three hazard analysis phases - ECA, CFA, PFA - provides a first level of confidence in the completeness of the analysis. This confidence is then be reinforced by evaluating the analysis by demonstrating its completeness through the development of an assurance case which relies on the analysis output. The template is modelled using Goal Structuring Notation (GSN) in Fig. 3.19. The approach to arguing the satisfaction of each goal is to inspect the analysis for the relevant phase of the ISHA process. To demonstrate the validity of the requirements, analysts leverage the documentation which is identified and/or created as part of the *Source Requirements* phase (Section 3.3) of the analysis. To demonstrate the validity of requirements analysts could rely, at least in part, in part on a critical assessment of the requirements gathering process. To argue the satisfaction of the safety requirements by design or by identification and mitigation, analysts would rely on the documentation of the *Source Concept of Operations* phase (Section 3.2), the *Source System Model* phase (Section 3.4) and the UTM construction, SCEM modelling and RAC portions of the PHA phase (Section 3.6). The maintenance of the system which is known be safety relevant is incorporated into the process fault analysis. The process of maintaining the system must be analyzed alongside the processes that the SUI is intended to satisfy.

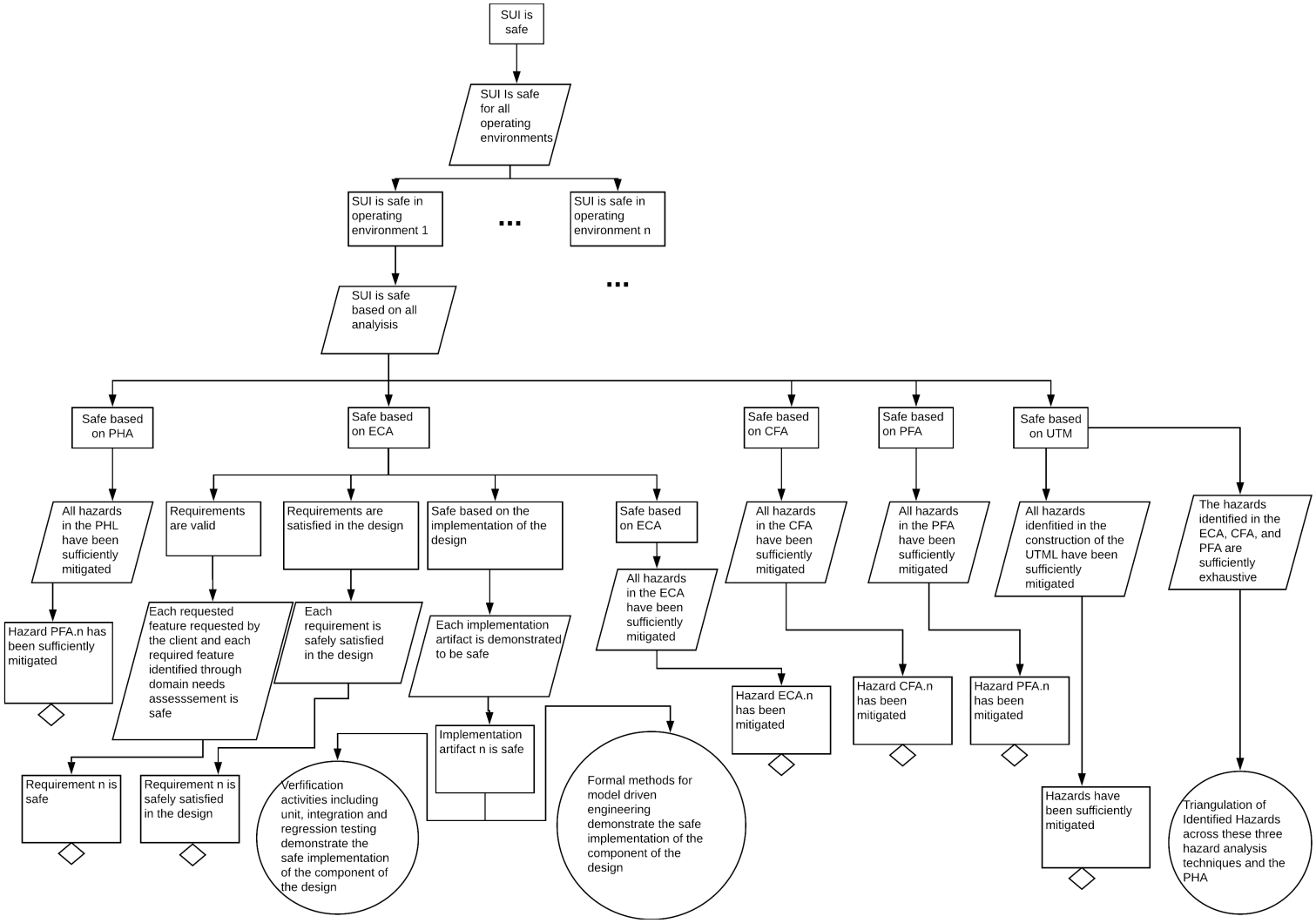

Figure 3.19: The template for ISHA assurance cases modelled using GSN

.

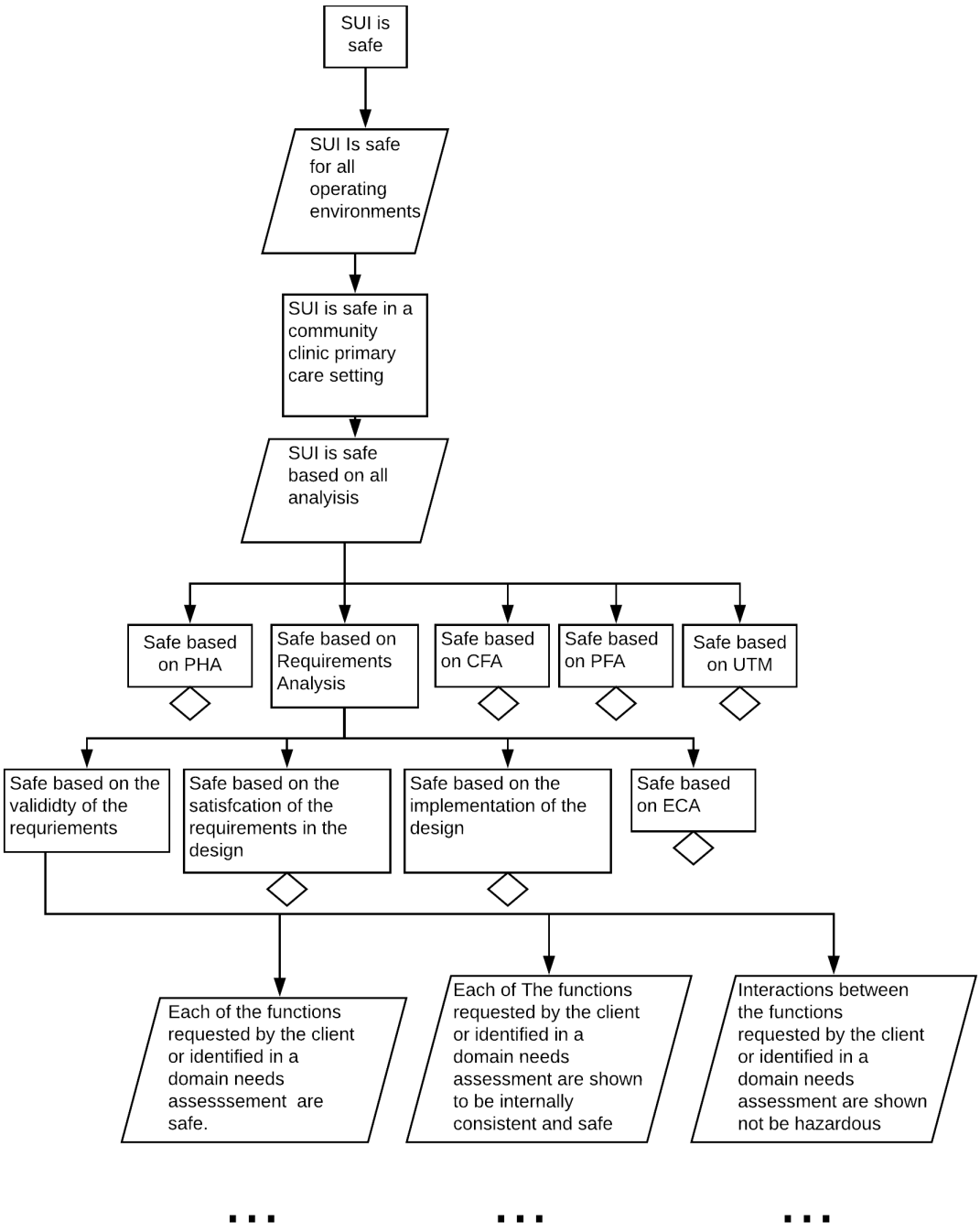

Figure 3.20: A goal structure for our running example of ISHA on a CIS's diabetes management process that is based on the ISHA assurance case template and modelled using GSN

.

Running Example

In Fig. 3.20, we use the template to express the progress in the assurance case for our running example.

3.11.3 Evidence Extraction

The evidence extraction phase may be widely variable depending on the strategies used by analysts in the assurance case. It will also depend on available data on the software and social organization which comprise the SUI. Further, the availability of RID, test documentation including plans and results will further inform the methods used to acquire supporting evidence. Examples of this process may include providing proofs for formal methods of software development or unit, integration and regression testing. It may also involve identifying extensive documentation on the observed outcome severity, occurrence, and detectability of the targeted hazards from available health data sources. The most refined goals will be those which identify the mitigation of the most refined hazards. Evidence specific to these must be extracted from the developed models and the artifacts which were used to generate them, as well as supporting engineering documentation.

Running Example

An example of evidence extraction in our running example would be the presence of an EMR adoption program at the SUI's facility. The regular measure of adoption of the SUI, and solicitation of barriers to use could identify complaints by providers about the frequency (*H5*) and information content (*H6*) of prescribing alerts. The presence of such a program running in tandem with updates to the CDS knowledge repository could have a significant and – we assert for the running example – sufficient mitigating effect.

3.11.4 Defeaters

Seeking only supporting evidence of the safety of the system however, risks confirmation bias. This is a running concern in the assurance case literature [75, 144, 114]. To combat confirmation bias, the ISHA method demands that analysts assess the strength of defeaters of the arguments they have made. The defeaters themselves would ideally be argued by objective parties thus avoiding the creation of weak defeaters and compounding the confirmation bias to which the assurance case is already subject.

Three types of defeaters must be considered: rebutting, undercutting and undermining [144]. Rebutting defeaters counter safety claims with counter examples. Undercutting defeaters argue that preconditions of an argument do not hold, and indicate that a safety

argument is therefore invalid. Finally, undermining defeaters challenge the validity of evidence. A defensible effort to identify defeaters of each type must be documented as part of the assurance case. This can lead to important qualifications of the argument which reveal undocumented assumptions about the SUI. These assumptions must be recorded, supplementing the SUI's assurance case documentation.

3.11.5 Running Example

We provide an example of each kind of defeater for the running example.

Rebutting

A rebutting defeater to the claim that alerts will not be too frequent (*H5*) would be a long running outbreak of a nosocomial infection in the long-term residential care facility which interacted with commonly used medications. This could compromise the CDS's, and thus the SUI's, ability to meet *FR4*.

Undercutting

An undercutting defeater is that no process has been described to ensure that the CDS relies on a current knowledge repository. Without a process in place to ensure that the data in the CDS is current, it cannot be assumed that the warnings it provides are clinically sound.

Undermining

An undermining defeater would be that the instruments or processes used in the EMR adoption measurement program were not validated or are explicitly flawed thus compromising their ability to detect problems with the CDS before they resulted in the failure of the SUI to meet its SRs.

3.12 Generate Recommendations

The output of the recommendations phase of the ISHA method is a set of proposed mitigations. Choosing how to go about mitigation design is done heuristically. There are a range of factors at play including how many hazards can be addressed with the assigned budget, the risk posed by those hazards, how many mitigations can be achieved within budget, and other typical constraints like resource availability.

As an example of a heuristic which might be applied, analysts might try and perform axial coding across hazards based on mitigation techniques which can address multiple problems. This approach has conceptual similarities to seeking Common Cause Failures [28]. Alternatively, seeking high hazard annotation density areas of the UTM may reveal clusters of hazards that are easily mitigated because they revolve around a small number of SUI components or interactions. It may be efficacious to add additional SCEMs in these situations as the location of the problem is clearly identified in the UTM. This approach is in keeping with the STPA process. STPA prescribes that missing constraints be identified and remedied. Another approach still, would be to formalize the language of the hazard expressions and to then use reduction algorithms to find commonalities between the trouble spots in the SUI. This approach might be particularly effective if an easily computable nomenclature (e.g., an ontology) is used to describe the hazards. ISHA does not prescribe how to choose which mitigations to design, it only prescribes that a process be used and documented.

Running Example

In order to make effective recommendations for many analysis projects, it is important to cast the recommendations in the context of other study findings, and we do so here. First, in the SUI a recent procurement contract was signed with the vendor of the health authority governance body reducing the leverage available to affect corrections in the system which might affect *H4*. For this reason, we would propose to raise the issue with the vendor but focus less attention on this hazard than on the internal issue of retraining which can affect the occurrence and detectability of *H3*. Combined, these two efforts can have some mitigating effect on *H1*. Monitoring the impact of this increased vigilance about this issue over time will provide insight into how much impact. This can be achieved by revisiting the issue through a future ISHA at the next evaluation cycle.

For *H2*, the CPOE information provider is about to begin soliciting feedback though a series of workshops in a product improvement campaign. The SUI's healthcare organization can commit staff to the process to ensure the customisability of the information source to ensure that the alert ratio and conciseness can be locally tailored to better suit the organizations needs. Further, a preliminary evaluation of the cost of setting up individualized alert profiles and training systems can be performed to explore the possibility of running different alert profiles for residents and seasoned providers.

Finally *H8* will be addressed by refining recurring usability studies which are being performed by the SUI's owning CIS. These improvements will explore the confusions which may be occurring in the order cancellation processes. As the procurement contract has already been signed with the vendor and as there is no appreciable internal technical resources, the

possibility of implementing more complete dynamic monitoring will be recorded for future consideration but will not be considered further at this time.

3.13 Repeat

The last phase of the ISHA process is to repeat the analysis at some later period in time. In an ideal world, the process of hazard analysis and mitigation would be continuous - depending on real time data collected by probes in the system which collect metrics on the safety margins in the SUI. However, prior to analysis it is unclear what metrics need to be monitored to inform safety decisions; since this data is thus unavailable, the ISHA method promotes migration towards such an ideal state through time series analysis. Within a given analysis, analysts iterate between the phases of the process until they achieve a satisfactory level of safety given the budget assigned to the analysis. They also begin the analysis anew at some time in the future when a new budget is assigned for additional safety analysis. It is important that analysts do so for at least two main reasons. First, the SUI and important environmental factors will evolve over time. Though ISHA uses closed system models, they are based on the premise that all important factors have been incorporated into the SUI model for analysis. In time, it will be discovered that some factors which were initially considered to be largely irrelevant will be found to have significant impact on system safety at a different point in time. Second, hazard mitigation involves changes to the system design. It is possible that by mitigating a hazard in the SUI, that a new hazard will be introduced. Even if caution is taken to consider what new hazard may have been introduced, analysts would be remiss if they did not revisit the SUI after it had operated for some time and new incident data had been collected which might demonstrate the actual versus foreseen impact of the implemented mitigations.

Chapter 4

Information System Hazard Analysis:
Generalized Insulin Infusion Pump

To further demonstrate the application of the Information System Hazard Analysis (ISHA) method we evaluate the safety of a Generalized Insulin Infusion Pump (GIIP). Zhang performed a Preliminary Hazard Analysis (PHA) on the GIIP leveraging his experience working with Food and Drug Administration (FDA) regulators. The quality of his work speaks to the extensive expertise he employed in his analysis. As with Zhang's analysis, our investigation will focus on the software aspects of the System Under Investigation (SUI).

The choice of a GIIP as an exemplar system for application of the ISHA method may seem discordant. At first glance such a system appears to be quite dissimilar to the Clinical Information Systems (CIS) for which the ISHA process was designed. To address this concern we will first provide our motivation in choosing the GIIP for our second case study, and we will then argue the validity of the choice. We were motivated to use the GIIP for our case study because, unlike with CIS, we were able to identify numerous published hazard analyses on these systems [31]. We could have performed such analysis on CIS and had them peer reviewed so as to generate the analyses required by the ISHA method; however, by using reviewed analyses which were executed by others, we are able to demonstrate how the ISHA method is more likely to be used in practice - using preexisting artifacts. Additionally, had we not divorced the execution of the input hazard analysis from the execution of the ISHA method, it could have been argued that our execution of the analyses which are used an inputs to ISHA had been done in such a way as to ensure that weaknesses in ISHA were not exposed. By avoiding this conflict of interest, we provide a more balanced perspective on the strength of the method we present. We argue that the application of the method is valid because the GIIP can be modelled as a control system in the same way as we model CIS. Like with the CIS discussed in Chapter 3, a GIIP system can modelled as a controller which receives information about a controlled process through observation. The control unit

dispenses insulin and then observes that the insulin has been dispensed through monitoring the remaining volume of drug.

Through the rest of this chapter we will apply the steps of the ISHA process in the context of the analysis of GIIP system. For those steps whose application was discussed in depth in Chapter 3 we will provide only a cursory description, while for those which were addressed only summarily Chapter 3, we will provide a more in depth description.

4.1 Select Team

We assert that an appropriate team was chosen for this analysis. For further details on ISHA's team selection process, readers are referred to Section 3.1.

4.2 Source Concept of Operations

In Zhang's article, he provides a succinct concept of operations statement. We adopt his statement but simplify the statement where he expands discussion into requirements statements:

> "The GIIP administers insulin to the user via a delivery path, composed of a drug reservoir, a drug delivery interface, and the infusion set. Along this path, the drug reservoir acts as a built-in storage unit for insulin that will be monitored and administered. The drug delivery interface represents a segment of concealed tubing connecting insulin flow from the reservoir to the infusion set. A pump delivery mechanism provides the force for moving insulin from the pump to the user at a prescribed rate and for a prescribed duration.
>
> The user/patient interacts with the GIIP through the GIIP user interface. The user interface allows the user to receive information from GIIP output devices and input data/commands through GIIP user input devices.
>
> The environment ... is constrained to physical properties such as temperature, pressure, sound, and radiation energies.
>
> The pump controller component represents an abstraction of generic insulin pump software. It provides the operational 'glue' and robustness in the GIIP system. To ensure correct and timely insulin administration...[The pump must provide bolus and basal insulin delivery functions as well as the associated support functions]." [149]

Zhang prescribes the system boundaries of the SUI to include

> "the GIIP (pump), the user, the infusion set (connection), and the environment. [It] exclude[s] device accessories such as glucose meters, the infusion sets themselves, and remote controllers... The environment ... is constrained to physical properties such as temperature, pressure, sound, and radiation energies" [149].

4.3 Source Requirements

Zhang provides a listing of Functional Requirements (FRs) for the GIIP's pump controller. He also expresses limitations of the functionality of the GIIP from which we can infer further SUI requirements. Those FRs specified for the pump controller are listed below:

- FR_1: The pump controller must "[i]nterpret user commands and inputs received from user input devices and act appropriately"
- FR_2: The pump controller must "[m]aintain user-defined insulin delivery profiles"
- FR_3: The pump controller must "[r]ecommend appropriate boluses to correct low [blood glucose] levels or cover future meals based on parameters entered by the user"
- FR_4: The pump controller must "[e]ncode and send instructions to the pump delivery mechanism such that it can precisely administer insulin based on user-defined insulin delivery profiles"
- FR_5: The pump controller must "[s]end information to output devices allowing the user to monitor the status of the pump [and] the current delivery session"
- FR_6: The pump controller must "[i]nstruct output devices to issue user-perceivable alarms and alerts"
- FR_7: The pump controller must "[r]ecord important data and events during pump use to facilitate clinical statistics and problem diagnosis"

Zhang's additional inferred FRs are listed below:

- FR_8: The GIIP must have functionality to provide "[b]asal background insulin replacement, delivered as a low, continuous infusion of insulin, periodically over a 24-hour interval."
- FR_9: The GIIP must have functionality to provide "[t]emporary basal insulin in proportion to physical conditions and activity levels of the user. Once programmed, a temporary basal overrides any ongoing normal basal at user-specified rates for user-indicated durations."

- FR_{10}: The GIIP must have functionality to provide "[n]ormal bolus... a user-defined amount of insulin [infused] immediately for covering food intake or correcting high [blood glucose] levels."
- FR_{11}: The GIIP must have functionality to deliver "[e]xtended bolus ... similar to normal bolus but ... delivered over a period designated by the user."

4.4 Source Model

We provide a high level view of the GIIP features with a use case diagram (Fig. 4.1). Zhang provides only a static model of the GIIP. We regenerate this model in Fig. 4.2 as a SysML Block Definition Diagram (BDD). As only a static model of the SUI is provided, we must infer the dynamic model from a combination of the static model and both the stated functional requirements of the pump controller and the inferred functional requirements of the GIIP. We present this inferred dynamic model using SysML activity diagrams for the "program insulin delivery profile" activity and the "activate insulin infusion profile activity." The profile programming activity is shown in Fig. 4.3; it addresses FR_2, FR_3, FR_8, FR_9, FR_{10}, and FR_{11}. The activation model shown in Fig. 4.4 addresses FR_4, FR_5, FR_6, FR_7. We address FR_1 implicitly through its subsumption by the other FRs.

4.5 Preliminary Hazard List

Identification/construction of Base Preliminary Hazard List

Zhang's PHA offered a Preliminary Hazard List (PHL) for the GIIP as one of its contributions. He indicates that the hazards identified in the analysis were sourced through manufacturer consultation, user consultation, clinician consultation, review of national and international standards, and adverse event reports. These sources are in alignment with the PHL data sources prescribed by the ISHA process for developing a new PHL from scratch. We provide this PHL in Appendix D. The reader may note that Zhang uses the term "contributing factor" in his PHL which we discussed previously in our introduction. This is synonymous with what we describe as a derived or specialized hazard.

As this chapter is about demonstrating the ISHA method, we choose two high level hazards and four derived hazards described in Zhang's PHL for analysis. We choose these high level and derived hazards based on the extent to which we perceived the benefit they provide to our explanation of ISHA. The subset of Zhang's PHL which we address in our analysis is listed below:

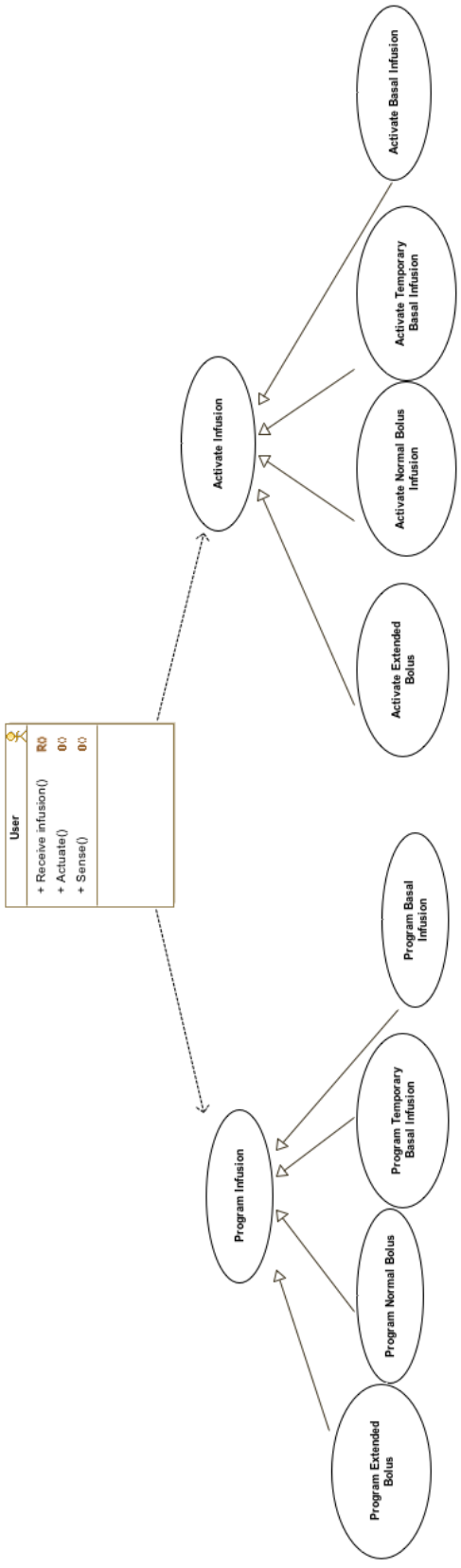

Figure 4.1: A SysML use case diagram for the GIIP

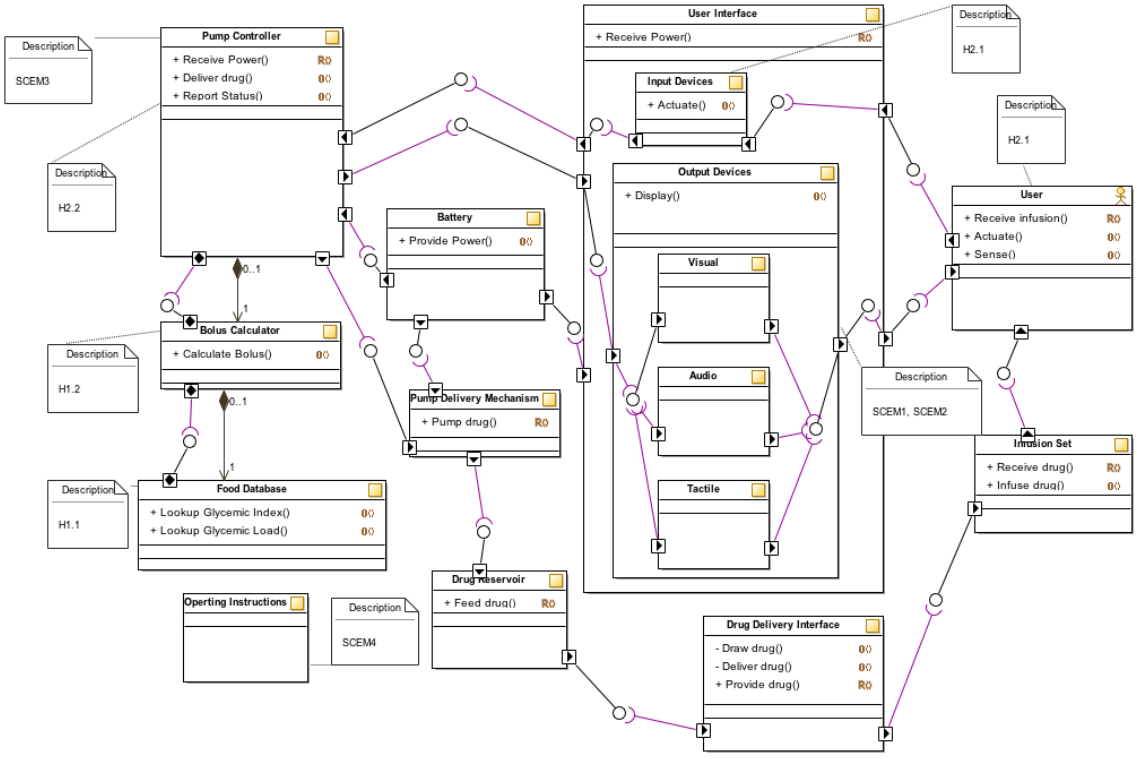

Figure 4.2: Zhang's static model of the GIIP [149] adapted to a SysML BDD. Mapping of the hazards to the GIIP blocks are included.

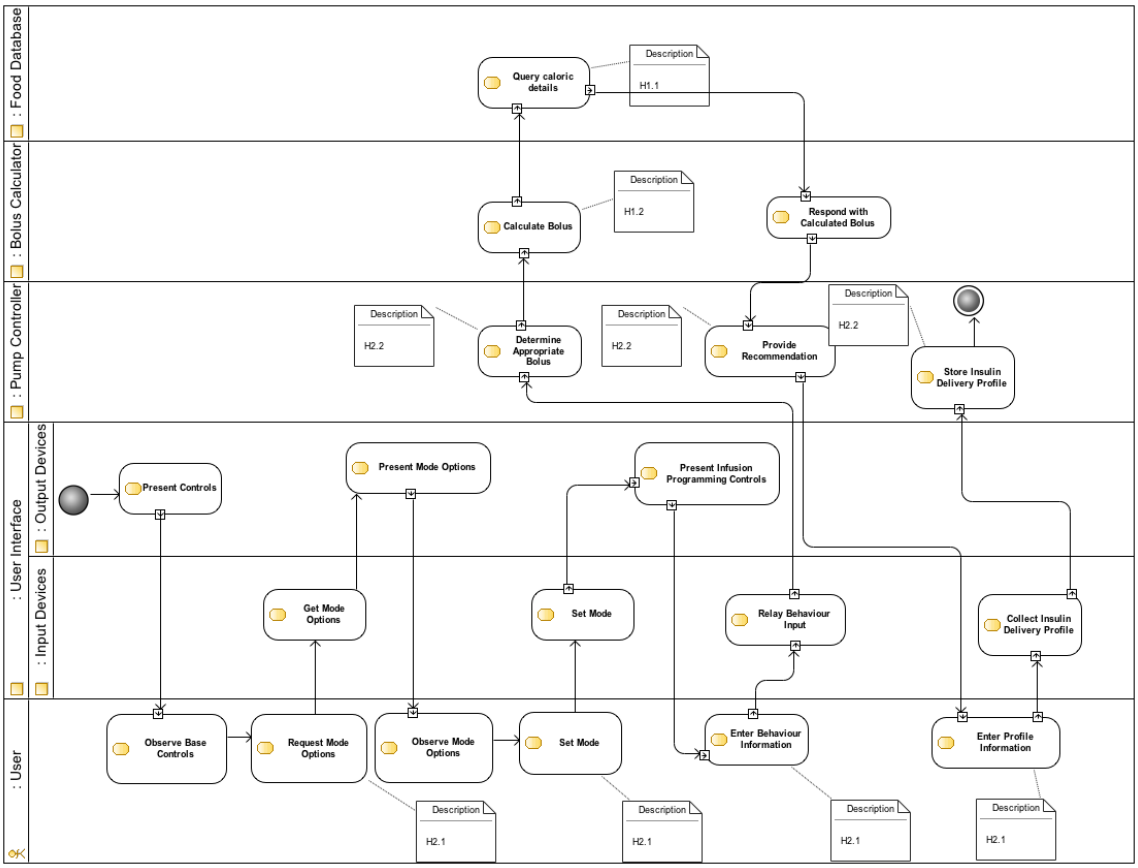

Figure 4.3: An inferred activity diagram modelling the dynamic aspects of the insulin delivery profile entry into the GIIP modelled with a SysML activity diagram. A mapping of the hazards to GIIP infusion programming activities is included.

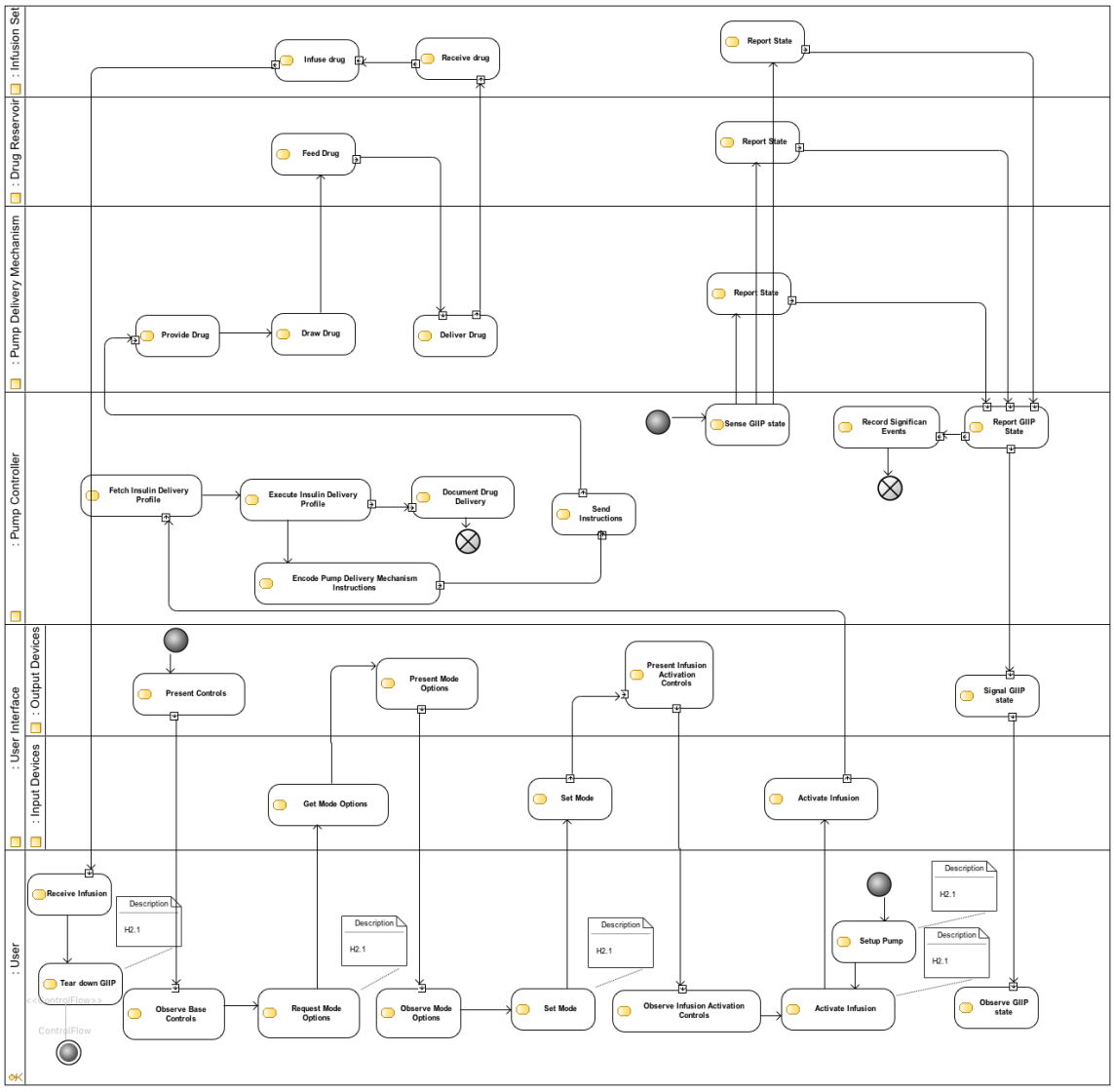

Figure 4.4: An inferred activity diagram modelling the dynamic aspects of the delivery profile activation for the GIIP modelled with a SysML activity diagram. A mapping of the hazards to GIIP infusion activation activities is included.

Hazard *H1* Incorrect meal bolus is recommended by the bolus calculator

- Hazard *H1.1* Food database contains erroneous information, causing incorrect calculation of the number of carbohydrates in a meal
- Hazard *H1.2* Design flaws/implementation defects in the bolus calculator

Hazard *H2* Pump unexpectedly restores to default factory settings without the user's awareness

- Hazard *H2.1* User inadvertently selects a restore of the factory settings
- Hazard *H2.2* Accumulated static electricity during use triggers an unexpected restore of default factory settings

We translate the hazards to a standardized terminology and nomenclature that use our STAMP-EMR model [29], our safety factors taxonomy [83], and the Phillips and Gong nomenclature [105]. We then tabulate the descriptions in Tab. 4.1 along with the risk triplets and Risk Assessment Codes (RACs) which we determine later in the process.

4.5.1 Hazard Checklist

As the purpose of the hazard checklist is to supplement an incomplete PHL, and as we have already needed to scope down the existing PHL for the purpose of this demonstration of the method, and have also explored this phase in depth in Chapter 3, we do not perform this phase of the method in our current demonstration.

4.5.2 Hazard Mapping

The resultant mappings for our case study are provided in Figs. 4.2, 4.3, and 4.4. As a consequence of the alignment of the expression of the hazards and the structure of the GIIP model, no refinement was found to be necessary.

4.5.3 Preliminary Hazard Analysis

Preliminary Hazard Prioritization

We do not perform the PHA's preliminary prioritization of hazards phase as we are considering an appropriate volume of hazards for our analysis budget. Instead we address all of our listed hazards.

		Catastrophic	*Critical*	*Marginal*	*Negligible*
H1	Incorrect [error condition] meal bolus [task object] is recommended [task action] by the bolus calculator [event agent]	D:2 O:2 R:Medium	D:2 O:2 R:Low	D:4 O:5 R:Serious	D:4 O:5 R:Serious
H1.1	Food database [error context] contains erroneous caloric density information [Error Condition], causing incorrect calculation [task action] of the number of carbohydrates in a meal [task object] containing the food whose caloric density was erroneous [task parameter] by the user [event agent]	D:2 O:2 R:Medium	D:3 O:2 R:Medium	D:4 O:5 R:Serious	D:4 O:5 R:Serious
H1.2	Design flaws/implementation defects in the bolus calculator [error context][event agent] produces incorrect [error condition] recommendation [error element]	D:2 O:2 R:Medium	D:2 O:2 R:Low	D:4 O:3 R:Medium	D:4 O:3 R:Medium
H2	Pump [event agent] unexpectedly [error condition] restores [event task] to default factory settings [error condition] without the user's [event agent] awareness [error condition]	D:3 O:2 R:Serious	D:3 O:2 R:Medium	D:3 O:2 R:Medium	D:3 O:2 R:Low
H2.1	User [event agent] inadvertently [error condition] selects a restore [event object] of the factory settings [event task] using the user interface [error context]	D:2 O:2 R:Medium	D:2 O:2 R:Low	D:2 O:2 R:Low	D:2 O:2 R:Low
H2.2	Accumulated [error condition] static electricity [error element] during use [event context] triggers an unexpected [error condition] restore [event task] of default factory settings [error element]	D:3 O:2 R: Serious	D:3 O:2 R:Medium	D:3 O:2 R:Medium	D:3 O:2 R:Low

Table 4.1: The GIIP hazard table including the RAC classifications.

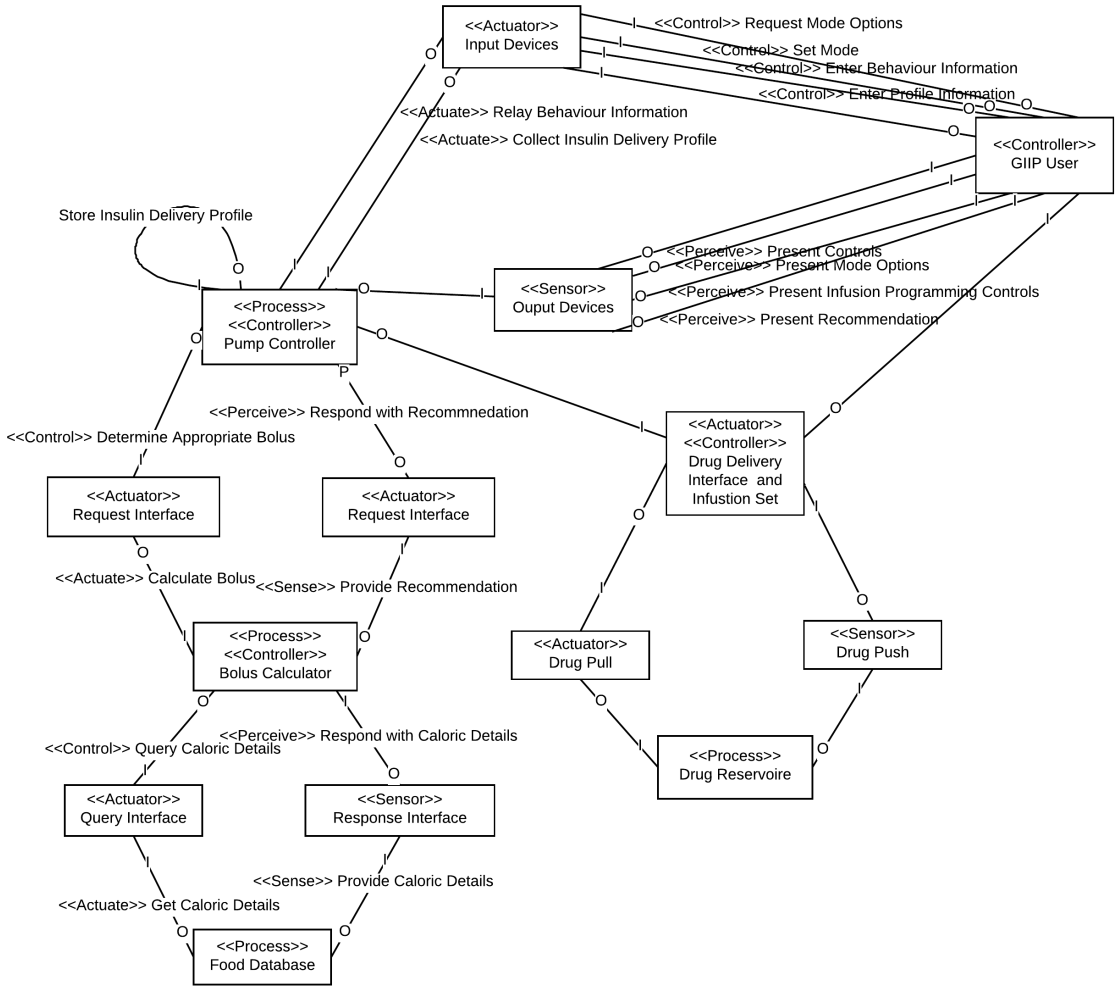

Figure 4.5: The UTM for the insulin delivery profile programming activity.

Construction of the Universal Triangulation Model

A view of the Universal Triangulation Model (UTM) for each of two primary use cases, programming and delivery, are provided in Fig. 4.5 and 4.6. Unlike in the analysis provided in Chapter 3, we use components to cluster the functions in the UTM in this analysis to represent the multiple operations they each perform.

Safety Constraint Enforcement Mechanism Modelling

To find explicitly defined Safety Constraint Enforcement Mechanisms (SCEMs) we reviewed Zhang's full article on the GIIP [149]. In so doing the following statements were found which are interpreted to describe SCEMs. The statements will be discussed in sequence.

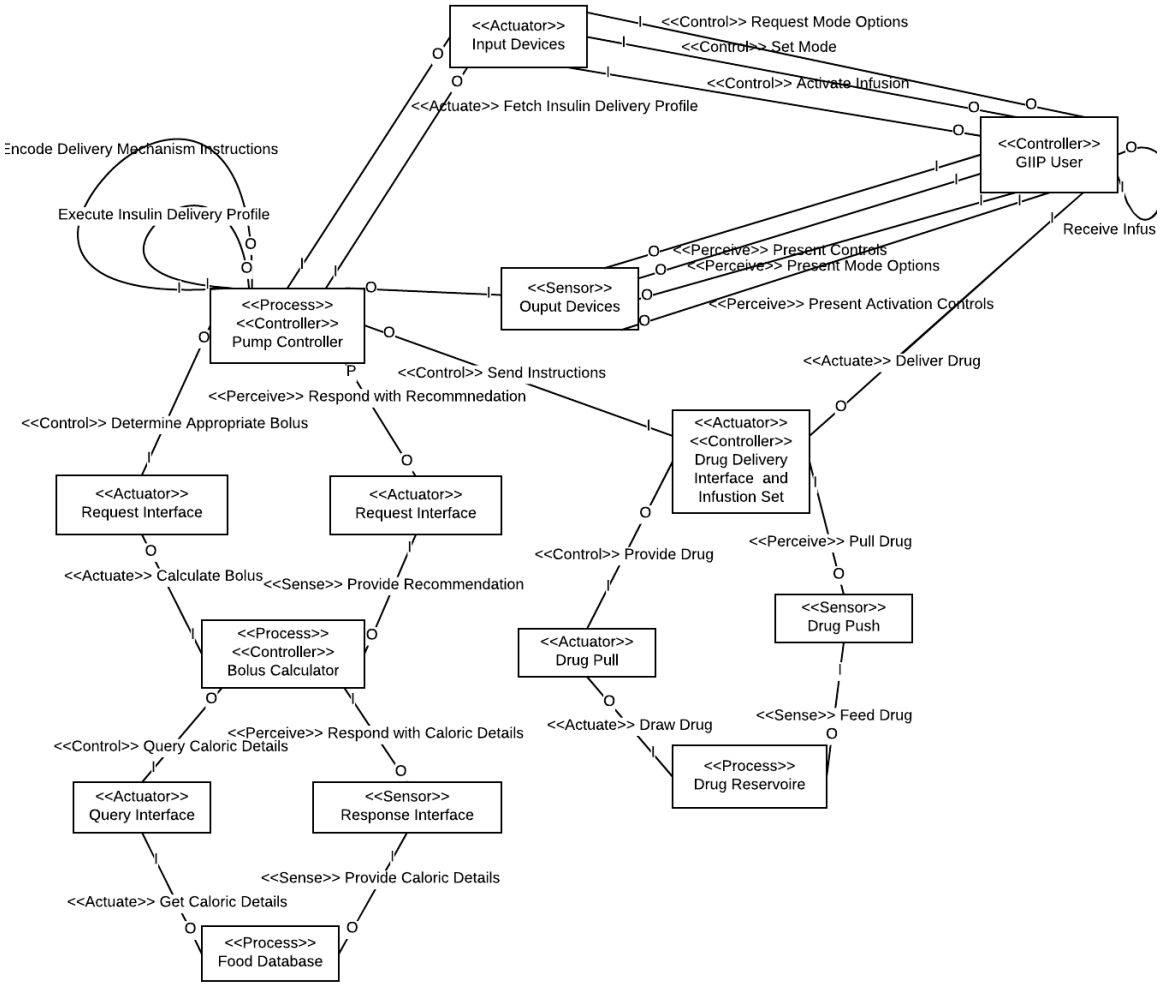

Figure 4.6: The UTM for the insulin delivery profile execution activity.

1. "[Insulin] pumps ... help maintain blood glucose ... levels by delivering rapid-acting insulin through a catheter placed under the skin."

2. "The user/patient interacts with the GIIP through the GIIP user interface. The user interface allows the user to receive information from GIIP output devices and input data/commands through GIIP user input devices."

3. "The pump controller ... provides the operational glue and robustness in the GIIP system. To ensure correct and timely insulin administration, the controller should be able to perform at least the basic [GIIP] functions"

   SCEM 1 "Send information to output devices allowing the user to monitor the status of the pump or of the current delivery session"

   SCEM 2 "Instruct output devices to issue user-perceivable alarms and alerts"

   SCEM 3 "Record important data and events during pump use to facilitate clinical statistics and problem diagnosis"

Statement 1 indicates that the GIIP as a complete unit is a SCEM. The purpose of the GIIP is to mitigate the risk of high or low blood glucose levels thus preventing the associated physical harms of these conditions.
Statement 2 describes how the GIIP's user interface is used to monitor and control the pumps operation thus providing a means by which the user can intervene in the control of their blood glucose process as necessary; however, this statement provides no new information about what SCEMs are present in the GIIP.
Statement 3 indicates that the listed FRs are the mechanisms by which the GIIP mitigates the risk of blood glucose mismanagement; this again provides no new information about what SCEMs are present in the GIIP.

SCEM 1 indicates that the GIIP provides a display which the user can reference to monitor the progress of insulin delivery, thus providing a means to intervene when a profile execution is deemed to be incorrect. *SCEM 2* indicates that the GIIP actively alarms the user if there is a problem with the pump during delivery. These warnings, though multi-modal, cannot address the primary concern of insulin over or underdose in many circumstances on account of the absence of a glucose meter as a component of the system. *SCEM 3* indicates that the GIIP captures and records "important data and events". This feature could be used to identify safety incidents, but Zhang provides no further description of what data, what events, or how the data is stored, nor how and when it is retrieved. Zhang's model is abstract, and so it may not have been feasible to include these details in his description, but this does provide a good place to start looking for potential mitigating

factors in the analysis of a realized insulin infusion pump. Each of these SCEMs is noted in the GIIP model in Figs. 4.2, 4.3, and 4.4.

Following the identification of explicitly declared SCEMs, we seek to identify further implicit SCEMs through manual investigation of the GIIP model. In doing so, we identify only one additional SCEM - provided below. Review of incidents described by Leveson [76] highlighted the role of training materials in controlling hazards and their potential for realization into accidents. In the analysis of more complex systems, or in the assessment of models not derived from safety analyses, results may differ.

SCEM 4 The operating instructions which are identified in the BDD (Fig. 4.2) are a mechanism of control over the user. They support the user in their efficacious use of the GIIP and thus are a SCEM.

Risk Assessment Codes

The ordinal values ascribed to each element of the risk triplets for the GIIP hazards as well as the RACs are presented in Tab. 4.1. We choose to use the same scales as we did in our Chapter 3 running example (Appendix C). As we discussed the assignment of risk factors in some depth in Chapter 3, we do not elaborate here, but instead provide only the summary risk assignments which were determined through analysis of the UTM, other artifacts generated in this analysis and Zhang's article [149].

Final Hazard Prioritization

For our prioritization scheme in this application of the method, we observe that the incorrect bolus calculation (*H1*) and factory reset (*H2*) hazards are largely independent. They will often be unrelated. Consequently, we choose to prioritize the hazards based on their average RACs over the gamut of outcome severities. We note a marginally higher average RAC class for *H1* over *H2*, and thus we recommend prioritizing *H1* over *H2*.

4.6 Event Chain Analysis

In the Event Chain Analysis (ECA) phase of ISHA, we attempt to identify and analyze hazards based on the SUI's failure to satisfy its Safety Requirements (SRs). This is done by developing event trees in which each top level event is a failure of the SUI to satisfy one of its SRs, and in which the branches document the related events that could lead to that top level failure. The SRs are those which must be satisfied to avert an accident. For each event in each of these trees, nominal detectability and occurrence scores must be assigned.

We consider the gamut of potential outcome severities independently of the occurrence and detectability of the hazards as usual.

In determining the appropriate scores to assign, analysts consider each leaf event independently. The parent events are initially assigned the worst of the scores of their children (e.g., a parent event with two children who's scores were D2O3 and D3O2 would have a score of D3O3). Analysts should also consider the potential additive nature of the combination of hazard occurrence. The events in the ECA may be connected by *and* or *or* gates, and thus sibling events may have either a diminishing or additive effect on occurrence or detectability. Further, the analysts must consider the impact of relevant SCEMs through inspection of the UTM. It is important to note that the UTM evolves over the course of ISHA execution, and is notably extended in the triangulation phase which follows the ECA, Component Fault Analysis (CFA) and Process Fault Analysis (PFA). The chosen assignments must be reassessed based on this more complete UTM. SCEMs identified here may mitigate the scores initially assigned to both leaf events and parent events.

The development of the ECA is one of the least structured processes in the ISHA method. This provides strength in that the creativity of the analysts can identify the most unexpected, but potentially problematic hazards. It also presents weakness in that there is never a clear indication of when to stop seeking new hazards. The ECA is one of the most subjective steps in the ISHA method, and consequently the one that is the least repeatable. Different executions of this phase of the method by different analysts or even the same analysts at different times is likely to lead to a different set of identified hazards [119]. Mitigating this variability is the analysts experience in their domain. Analysts will be familiar with the most common, or at least the most infamous failures in their SUI or the failures in systems similar to the SUI, and so they will use this tacit knowledge in addition to available evidence to guide their assignment of the occurrence and detectability scores.

Reviewing the requirements identified in Section 4.3, we find that *FR1* is a generic requirement which is refined in many of the other requirements, and so we do not analyze the requirement further. The remaining FRs are more independent. For demonstrative purposes we focus on *FR2*. We identify the failure of the SUI to satisfy this requirement as *H3*. In Fig. 4.7 we provide a tree structure for the profile maintenance requirement. We tabulate the hazards refined from it, the risk triplets, and RACs in Tab. 4.2.

4.7 Component Fault Analysis

For the CFA, ISHA prescribes that a reliability analysis of the SUI be performed. Though a number of techniques to do so exist, one of the most developed is Failure Mode and Effects Analysis (FMEA). Rather than demonstrate the execution of a full FMEA analysis here,

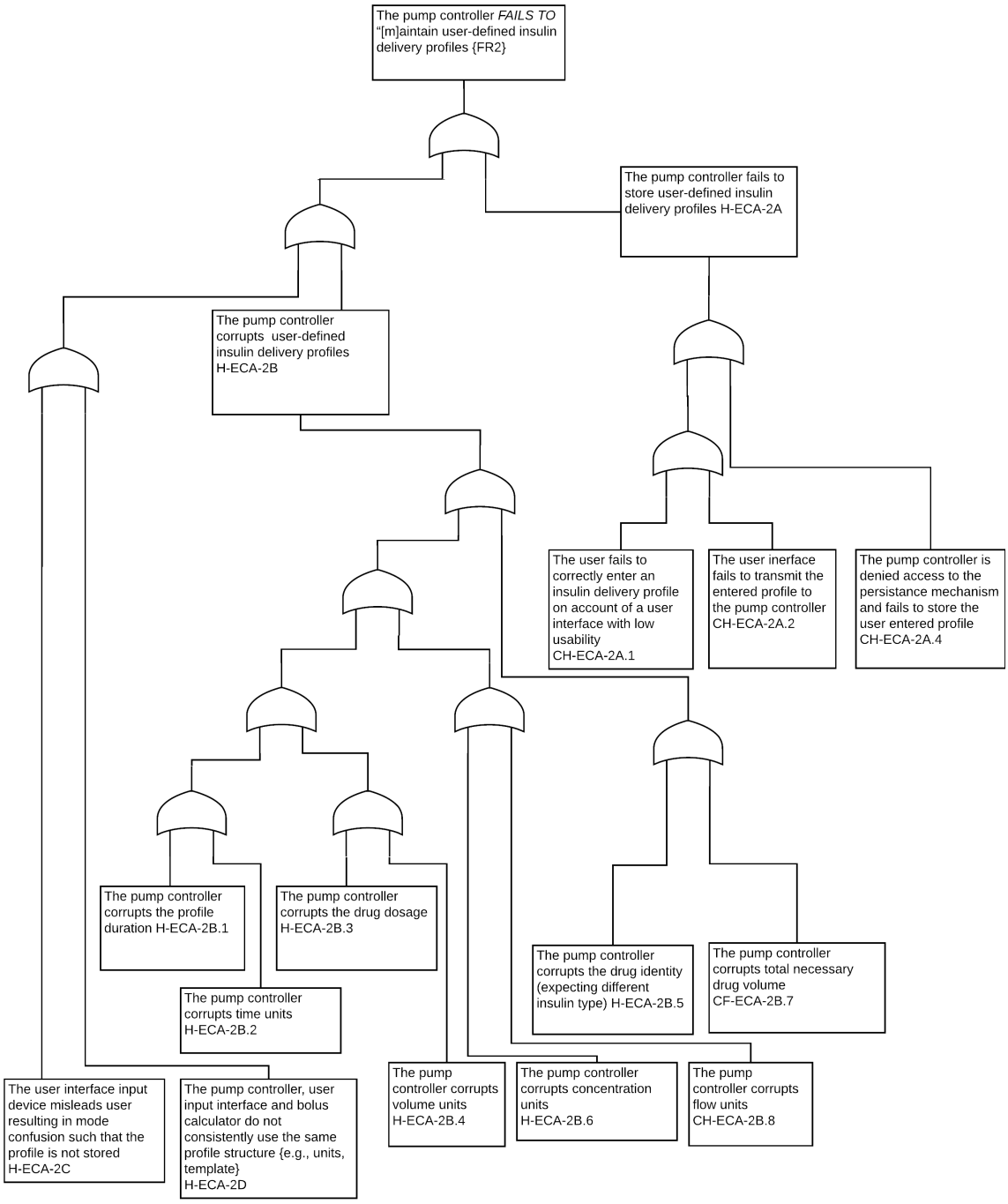

Figure 4.7: An ECA tree for the GIIP.

		Catastrophic	Critical	Marginal	Negligible
FR2	The pump controller FAILS TO maintain user-defined insulin delivery profiles	D:4 O:3 R:High	D:4 O:3 R:Serious	D:4 O:3 R:Medium	D:4 O:3 R:Medium
H-ECA-2A	• The pump controller fails to store user-defined insulin delivery profiles	D:4 O:3 R:High	D:4 O:3 R:Serious	D:4 O:3 R:Medium	D:4 O:3 R:Medium
H-ECA-2A.1	The user fails to correctly enter an insulin delivery profile on account of a user interface with low usability	D:4 O:3 R:High	D:4 O:3 R:Serious	D:4 O:3 R:Medium	D:4 O:3 R:Medium
H-ECA-2A.2	The user interface fails to transmit the user entered profile to the pump controller	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R:Medium
H-ECA-2A.3	The pump controller is denied access to the persistence mechanism and fails to store the user entered profile	D:1 O:2 R:Medium	D:1 O:2 R:Low	D:1 O:2 R:Low	D:1 O:2 R:Low
H-ECA-2B	• The pump controller corrupts user-defined insulin delivery profiles	D:4 O:2 R: High	D:4 O2: R:Serious	D:4 O:2 R: Medium	D:4 O:2 R: Medium
H-ECA-2B.1	The pump controller corrupts the profile duration	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.2	The pump controller corrupts the profile time units	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.3	The pump controller corrupts the drug dosage	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.4	The pump controller corrupts the volume units	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.5	The pump controller corrupts the drug identity (expecting a different type of insulin)	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.6	The pump controller corrupts the concentration units	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.7	The pump controller corrupts the total necessary drug volume	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2B.8	The pump controller corrupts the flow units	D:4 O:2 R:High	D:4 O:2 R:Serious	D:4 O:2 R:Medium	D:4 O:2 R: Medium
H-ECA-2C	• The user interface input device misleads user resulting in mode confusion such that the profile is not stored	D:4 O:2 R: High	D:4 O2: R:Serious	D:4 O:2 R: Medium	D:4 O:2 R: Medium
H-ECA-2D	• The pump controller, user input interface and bolus calculator do not consistently use the same profile structure e.g., units, template	D:3 O:2 R: Serious	D:3 O:2 R:Medium	D:3 O:2 R: Medium	D:3 O:2 R: Low

Table 4.2: A tabulation of the ECA hazards identified for the GIIP's *FR2*.

.

we use results of a Health Care Failure Mode and Effects Analysis (HFMEA) of a generic infusion pump [145] published by Wetterneck. We assert that the findings are largely generalizable to the context of an insulin infusion pump. Though the focus of Wetterneck's article was on execution, and though the full results were not provided, Wetterneck did summarize some of the findings. We tabulate these in Tab. 4.3. Inspection of these results highlights some of the differences between FMEA and HFMEA. These include the use of a two factor Risk Probability Number (RPN), identification of single point weaknesses, the identification of SCEMs, a binary assignment for detectability, a decision about whether or not to proceed with analysis, and finally a differing risk model where severity is independent of occurrence (probability) and detectability. ISHA uses an explicitly assigned RAC instead of an RPN. It incorporates the single point of failure concept and the control measure concept into the influence of SCEMs on the risk posed by hazards. It uses a scale for detectability which can have more than two levels. Finally, it uses prioritization to determine whether to proceed with the pursuit of a hazard; this is a more sophisticated approach to prioritization that HFMEA's binary "Proceed" decision.

In order to use the results provided by Wetterneck in conjunction with those we have derived so far, we need to normalize the results. The severity scale matches closely enough that we will not modify it, though it is necessary to extrapolate scores across the severity scale as our model considers outcome severity as dependent on probability and detectability. We do not do so in our table, and instead leave these entries blank for now. When considering event probability however, we find that the scale we have used is a five point scale, while Wetterneck's is a four point scale. Observing the scale definitions, we choose to normalize these scales by adding one to all probability ratings provided by Wetterneck which are of a magnitude of three or four. Therefore, mapping Wetterneck's scale to ours we would show the following correspondence $\{1 \rightarrow 1, \quad 2 \rightarrow 2, \quad 3 \rightarrow 4, \quad 4 \rightarrow 5\}$. With the mapping of Wetterneck's detectability scores, we have less detail on the native scale, and so are forced to push her binary scores to the extremes of our scale. This results in the following correspondence between her scale and ours - $\{Y \rightarrow 1, \quad N \rightarrow 5\}$. The scoring in Wetterneck's reported results however also includes a number of dashes whose semantics she does not discuss. As the detectability scores are necessary for our assessment of risk, we infer values for them based on their definition and Wetterneck's binary scale. Finally, we replace the hazard score in Wetterneck's data with a RAC based on the risk triplet and the risk tables presented in Appendix C. We tabulate the normalized results in Tab. 4.4.

Failure Modes E:Administration E2:Program Pump E2b:Primary continuous infusion, using manual calculation	Scoring			Decision Tree Analysis			
	Severity[1]	Probability[2]	Hazard Score	Single Point Weakness	Control Measure	Detectable	Proceed
Turn pump on - E2b(1)							
Battery dead	1	4	4	N	—	—	N
No power	4	1	4	Y	Y	—	N
Pump self-checks - E2b(2)							
Electrical mechanical failure	1	4	4	N	—	—	N
Push Options button - E2b(3)							
Perform manual calculations - E2b(4i)							
Inaccurate calculation	3	4	12	—	N	N	Y
Incomplete process - no double check	3	4	12	—	N	N	Y
Incorrect weight	3	4	12	—	N	N	Y
No calculator	1	1	1	N	—	—	N
Enter rate - E2b(5i)							
Push incorrect key pads	4	4	16	—	N	N	Y
Misread order	4	4	16	—	N	N	Y
Enter volume - E2b(6i)							
Adjust volume based on assumptions - too much	2	2	4	N	—	—	N
Adjust volume based on assumptions - too little	3	3	9	—	Y	—	N
Push incorrect key pads	4	4	16	—	N	N	Y
Misread order	4	4	16	—	N	N	Y
Enter rate instead of volume	4	2	8	—	N	N	Y
Push “run” after verifying data entered - E2b(7i)							
No activation - “run” not pushed	1	4	4	N	—	—	N
Misread display	4	4	16	—	N	N	Y

[1]Severity [1-4] - 1:minor, no injury; 4:catastrophic death or permanent loss of function
[2]Probability [1-4] - 1:occurs once in 5 to 30 years; 4:happens several times per year

Table 4.3: A summary of findings for Wetterneck's HFMEA performed on an infusion pump. Adapted from [145]

Id	Failure Modes E:Administration E2:Program Pump E2b:Primary continuous infusion, using manual calculation	Catastrophic	Critical	Marginal	Negligible
H-CFA-1	[Can't] Turn pump on - E2b(1)				
H-CFA-1.1	Battery dead				O:1 D:5 R: Medium
H-CFA-1.2	No power	O:1 D:1 R:Medium			
H-CFA-2	Pump [fails] self-checks - E2b(2)				
H-CFA-2.1	Electrical mechanical failure				O:2 D:1 R:Low
H-CFA-3	Push Options button - E2b(3)				
H-CFA-4	Perform [automated] calculations - E2b(4i)				
H-CFA-4.1	Inaccurate calculation		O:1 D:4 R:Medium		
H-CFA-4.2	Incomplete process - no double check		O:5 D:4 R:High		
H-CFA-4.3	Incorrect weight		O:5 D:4 R:High		
H-CFA-4.4	No calculator				O:1 D:1 R:Low
H-CFA-5	Enter rate - E2b(5i)				
H-CFA-5.1	Push incorrect key pads	O:5 D:4 R:High			
H-CFA-5.2	Misread order	O:5 D:4 R:High			
H-CFA-5.3	Enter volume - E2b(6i)				
H-CFA-5.4	Adjust volume based on assumptions - too much			O:2 D:4 R: Medium	
H-CFA-5.5	Adjust volume based on assumptions - too little		O:4 D:4 R:High		
H-CFA-5.6	Push incorrect key pads	O:5 D:4 R:High			
H-CFA-5.7	Misread order	O:5 D:4 R:High			
H-CFA-5.8	Enter rate instead of volume	O:2 D:4 R:High			
H-CFA-6	Push "run" after verifying data entered - E2b(7i)				
H-CFA-6.1	No activation - "run" not pushed				O:5 D:4 R:Serious
H-CFA-6.2	Misread display	O:5 D:4 R:High			

Table 4.4: The normalized summary of findings for Wetterneck's infusion pump HFMEA [145].

4.8 Process Fault Analysis

For the PFA we perform a HAZard OPerability (HAZOP) analysis of the GIIP's "Program Infusion Profile" activity. To do so, we first tabulate the activities shown in Fig. 4.3 along with an annotation for the owner of the activity, along with the HAZOP keywords suggested by Kletz [69]. We then consider the consequence of the considered deviations in flow described by those keywords in the context of the specified hazards, and thus use it to seek additional derived hazards. Having assessed the consequences, we tabulate a risk triplet and assign a RAC to each considered deviation. We summarize the results of this process in Tabs. 4.5, 4.6. Alternatively, approaching flow analysis using a feedback analysis method like those described by Hollnagel [52] or Leveson [76] would be appropriate.

4.9 Hazard Triangulation

For the hazard triangulation we perform four steps:

- ensure that each hazard identified in the ECA is also identified in either the CFA or the PFA.
- ensure that each hazard in the CFA is identified in the ECA
- ensure that each hazard in the PFA is identified in the ECA
- cross validate the CFA and PFA

For each discrepancy in the first three comparisons listed above, we investigate why the missing hazard is absent. The purpose of triangulation is to cross validate the identification of hazards in the system. Each time one appears to be missing, we need to understand why as there may be other hazards which could have been identified alongside the one which was missing. By performing this cross check we can improve the thoroughness of the hazard identification process.

With the fourth comparison, analysts must maintain awareness of the differing perspectives provided by the CFA and the PFA. One focuses on static analysis while the other focuses on dynamic analysis. While it may be easy to identify hazards arising from simpler flow problems like unintentional signals, failure to send a signal, reversal of fluid flow etc. in a CFA, finding problems with timing gates (minimum start and stop times for a process step), or event sequence would be more difficult. Similarly, identifying the consequence of failures of static components of the system which lie largely beyond the inspected process flows except under unusual circumstances may be difficult to identify in the PFA. Taking

	Catastrophic	Critical	Marginal	Negligible
H1 - Incorrect bolus recommendation	O:2 D:4 R:High	O:2 D:4 R:Serious	O:3 D:4 R:Medium	O:3 D:4 R:Medium
	Present Controls ⟨⟨ Output Device ⟩⟩			
No or Not				
	Display is unavailable			
H-PFA-PresentControls-A-1	✗	✗	✗	✗
More				
✗	✗	✗	✗	✗
Less				
✗	✗	✗	✗	✗
As Well As				
	Superfluous controls/additional features			
H-PFA-PresentControls-D-1	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:2 D:1 R:Low	O:2 D:1 R:Low
Part of				
	Incomplete functionality forcing workarounds			
H-PFA-PresentControls-E-1	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:3 D:1 R:Medium	O:3 D:1 R:Low
Reverse				
	Device expects user to initiate interaction			
✗	✗	✗	✗	✗
Other than/Instead				
	Mode confusion – user is presented with a screen which they misinterpret to be the programming interface, but it is actually the administration interface			
H-PFA-PresentControls-G-1	O:2 D:4 R:High	O:2 D:4 R:Serious	O:2 D:4 R:Medium	O:2 D:4 R:Medium
Early				
✗	✗	✗	✗	✗
Late				
✗	✗	✗	✗	✗
Before				
✗	✗	✗	✗	✗
After				
✗	✗	✗	✗	✗

Table 4.5: A summary of the HAZOP analysis of the incorrect bolus recommendation hazard relative to the *program delivery profile* activity for the GIIP

	Catastrophic	Critical	Marginal	Negligible
H2 - Restore to factory defaults without user's awareness	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:3 D:1 R:Low	O:3 D:1 R:Low
	Present Controls ⟨⟨ Output Device ⟩⟩			
No or Not				
	Display is unavailable			
H-PFA-PresentControls-A-1	✗	✗	✗	✗
✗	✗	✗	✗	✗
More				
✗	✗	✗	✗	✗
Less				
✗	✗	✗	✗	✗
As Well As				
	Superfluous controls/additional features			
H-PFA-PresentControls-B-1	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:2 D:1 R:Low	O:2 D:1 R:Low
Part of				
	Incomplete functionality forcing workarounds			
H-PFA-PresentControls-D-1	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:3 D:1 R:Medium	O:3 D:1 R:Low
Reverse				
	Device expects user to initiate interaction			
H-PFA-PresentControls-E1	✗	✗	✗	✗
Other than/Instead				
	Mode confusion – user is presented with a screen which they misinterpret to be the programming interface, but it is actually the administration interface			
H-PFA-PresentControls-G-1	O:2 D:1 R:Medium	O:2 D:1 R:Low	O:2 D:1 R:Low	O:2 D:1 R:Low
Early				
✗	✗	✗	✗	✗
Late				
✗	✗	✗	✗	✗
Before				
✗	✗	✗	✗	✗
After				
✗	✗	✗	✗	✗

Table 4.6: A summary of the HAZOP analysis of the *program delivery profile* activity for the GIIP

multiple perspectives in the analysis of the SUI provides this strength, and so it is not unexpected that using the same methods to cross validate results would be challenging.

4.9.1 Running Example

In our running example we approximate the equivalence of the results of our ECA and CFA in Tab. 4.7 and the results of our ECA and PFA in Tab. 4.8. Superficially, the initial analysis seems relatively complete with a correlation density greater than half for all three interaction comparisons, however, deeper consideration of each interaction reveals weakness in each of the three analyses. For example, the PFA considered the process of entering weight of the patient, total volume of the drug, band the delivery rate; it did not address entering or calculating the drug dosage, the solution concentration, or explicitly entering the time units for flow rate, or total delivery time of the infusion. The CFA highlights a concern about an unavailable display but the design as modelled in the UTM based on Zhang's description (Figs. 4.5, 4.6) does not clearly indicate the nature of the input devices. The GIIP might use a touch screen or may use standard contact switch buttons. The nature of the the dead battery concern identified in the CFA is nebulous as well on account of the range of possibilities with a dying battery and the consequence this might have on the operation of the GIIP. What kind of health checks are present on the battery? How stringent are they? Could a low power battery cause data corruption on account of poor signal strength in transmission? Further detail on the management of the power source would be necessary to arrive at firm recommendations with respect to this hazard.

The demonstration of the method also, however, highlights the value of triangulating analysis across the three approaches to hazard identification. A very clear example of is the correlation of the void of interactions between the PFA hazards and data integrity requirements (Tab. 4.8) and the girth of interactions between the CFA hazards and the data integrity requirements (Tab. 4.7). With respect to the data integrity requirements identified in the ECA, the specificity with which the hazards were declared proved beneficial in that it forced extensive consideration of the data entry process. It was also detrimental in that such a fine grained evaluation was time consuming. Similarly, consideration of superfluous controls, workarounds and mode confusion in the PFA often yielded similar correlation profiles with the hazards identified in the other two analyses. This is a consequence of the relationship between the two concepts. Additional modes can be considered superfluous. At the same time, an insufficient number of modes may not meet user needs and may result in the development of unsafe workarounds. The inter-connectivity of the hazards identified in these analyses highlights the challenge in hazard analysis of complex systems of disentangling competing concerns in system design and implementation.

		ECA Hazards											
		H-ECA-2A			H-ECA-2B								
		H-ECA-2A.1	H-ECA-2A.2	H-ECA-2A.3	H-ECA-2B.1	H-ECA-2B.2	H-ECA-2B.3	H-ECA-2B.4	H-ECA-2B.5	H-ECA-2B.6	H-ECA-2B.7	H-ECA-2B.8	H-ECA-2C	H-ECA-2D
CFA Hazards	H-CFA-1.1 Battery Dead			✓	✓	✓	✓	✓	✓	✓	✓	✓		
	H-CFA-1.2 No Power			✓	✓	✓	✓	✓	✓	✓	✓	✓		
	H-CFA-2.1 Electrical Mechanical Failure		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
	H-CFA-3 Push Option Button	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	H-CFA-4 Perform Automated Calculations	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	H-CFA-4.1 Inaccurate Calculation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	H-CFA-4.2 Incomplete Process - No Double Check	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	H-CFA-4.3 Incorrect Weight	✓	✓	✓			✓	✓		✓	✓	✓	✓	✓
	H-CFA-4.4 No Calculator													
	H-CFA-5 Enter Rate	✓	✓	✓		✓	✓	✓		✓		✓	✓	✓
	H-CFA-5.1 Incorrect Buttons	✓	✓										✓	
	H-CFA-5.2 Misread Order	✓	✓	✓									✓	✓
	H-CFA-5.3 Enter Volume	✓	✓	✓				✓			✓		✓	✓
	H-CFA-5.4 Adjust Volume Too Much	✓	✓	✓				✓			✓		✓	✓
	H-CFA-5.5 Adjust Volume Too Little	✓	✓	✓				✓			✓		✓	✓
	H-CFA-5.6 Incorrect Buttons	✓											✓	
	H-CFA-5.7 Misread Order	✓	✓	✓									✓	✓
	H-CFA-5.8 Enter Rate Instead of Volume													
	H-CFA-6 Push Run													
	H-CFA-6.1 Run Not Pushed													
	H-CFA-6.2 Misread Display	✓											✓	✓

Table 4.7: A matrix demonstrating a lack of correlation between the results from the ECA for *FR2* and the CFA.

		ECA Hazards											
		H-ECA-2A			H-ECA-2B								
		H-ECA-2A.1	H-ECA-2A.2	H-ECA-2A.3	H-ECA-2B.1	H-ECA-2B.2	H-ECA-2B.3	H-ECA-2B.4	H-ECA-2B.5	H-ECA-2B.6	H-ECA-2B.7	H-ECA-2B.8	H-ECA-2C	H-ECA-2D
PFA Hazards	H-PFA-PresentControls-A-1 Display is Unavailable													
	H-PFA-PresentControls-B-1 Superfluous Controls	✓		✓									✓	✓
	H-PFA-PresentControls-D-1 Incomplete functionality/workarounds	✓		✓									✓	✓
	H-PFA-PresentControls-E-1 Device expects user to initiate interaction	✓	✓	✓									✓	✓
	H-PFA-PresentControls-G-1 Mode confusion admin/program	✓		✓									✓	✓

Table 4.8: A matrix demonstrating the lack of correlation between the results from the ECA and PFA.

		PFA Hazards				
		H-PFA-PresentControls-A-1	H-PFA-PresentControls-B-1	H-PFA-PresentControls-D-1	H-PFA-PresentControls-E-1	H-PFA-PresentControls-G-1
CFA Hazards	H-CFA-1.1 Battery Dead	✓				
	H-CFA-1.2 No Power	✓				
	H-CFA-2.1 Electrical Mechanical Failure	✓		✓		
	H-CFA-3 Push Option Button	✓	✓	✓	✓	✓
	H-CFA-4.1 Inaccurate Calculation		✓	✓		✓
	H-CFA-4.2 Incomplete Process - No Double Check	✓	✓	✓	✓	✓
	H-CFA-4.3 Incorrect Weight		✓	✓		✓
	H-CFA-4.4 No Calculator	✓		✓	✓	✓
	H-CFA-5 Enter Rate	✓	✓	✓	✓	✓
	H-CFA-5.1 Incorrect Buttons	✓	✓	✓		✓
	H-CFA-5.2 Misread Order					
	H-CFA-5.3 Enter Volume	✓	✓	✓	✓	✓
	H-CFA-5.4 Adjust Volume Too Much	✓	✓	✓	✓	✓
	H-CFA-5.5 Adjust Volume Too Little	✓	✓	✓	✓	✓
	H-CFA-5.6 Incorrect Buttons	✓	✓	✓	✓	✓
	H-CFA-5.7 Misread Order					
	H-CFA-5.8 Enter Rate Instead of Volume		✓	✓	✓	✓
	H-CFA-6 Push Run		✓	✓	✓	✓
	H-CFA-6.1 Run Not Pushed		✓	✓	✓	✓
	H-CFA-6.2 Misread Display		✓	✓	✓	✓

Table 4.9: A matrix demonstrating a lack of correlation between the results from the CFA and the PFA.

4.10 Assurance Case Construction

In the construction of the assurance case for the GIIP we will use Kelly's "hazard directed argument" and "functional decomposition argument" [64, 61] patterns and Denney's lightweight assurance case construction process [23] in combination (Chapter 2). Together the patterns and process will extract the safety requirements from the collected artifacts; they will provide an argument structure; finally, they will extract the relevant evidence to support the argument.

Denney's process requires a hazards table and a safety requirements table for execution. We provide the SRs which we identified for the GIIP through analysis in Tab. 4.10. As per Denney's prescription, these are described with an identifier, a source, an allocation, a verification method and a verification allocation. As per ISHA's prescription, these are described using a standardized terminology and nomenclature. We choose a combination of our safety factor taxonomy [83], and the Phillips and Gong nomenclature [105] for this purpose. As the GIIP is an abstraction of an insulin pump rather than a realization, we indicate that the verification method and allocation are to be determined. A consequence of this is that the argument structure will be in a state of incomplete development. The provided goal structure however could be used to implement a complete safety case for a realized pump.

We synthesize the hazard table from our analysis findings in the appendix in Tab. E.1. As per Denney's prescription, we describe hazards using an identifier, contributing factors/derived hazards (causes), and mitigations. As per ISHA's prescription we use a nomenclature and taxonomy to normalize descriptions - our contributing factor taxonomy [83], and the Phillips and Gong nomenclature [105].

To assess the mitigations, we must inspect the UTM. The UTM is extended to included the details learned through the ECA, CFA, and PFA in Fig. 4.8 and Fig. 4.9 for the programming and delivery activities respectively. Further we assert that some processes are in place to assure the quality of the SUI and describe these assumed processes as mitigating factors for some of the entries in the hazard table. With ISHA we prescribe that the risk of hazards be tabulated as this is important for prioritization in the recommendations phase of the method.

Id	Requirement	Source	Allocation	Verification Method	Verification Allocation
FR2	The pump controller [event agent] must maintain [task action] user-defined [event agent] insulin delivery profiles [error element][task object]	Zhang [149]	Pump Controller & User	TBD	TBD

Id	Requirement	Source	Allocation	Verification Method	Verification Allocation
FR3	The pump controller [event agent] must recommend [task action] appropriate boluses [task object] to correct low blood glucose levels [task parameters] or cover future meals based on parameters [task parameters] entered by the user [event agent]	Zhang [149]	Bolus Calculator & User	TBD	TBD
FR4	The pump controller [event agent] must encode [task action] and send [task action] instructions [task object] to the pump delivery mechanism [event agent] such that it can [accurately] administer [task action] insulin [task object] based on user [event agent]-defined [task action] insulin delivery profiles [task object].	Zhang [149]	Pump Controller & Pump Delivery Mechanism & User	TBD	TBD
FR5	The pump controller [event agent] must send [task action] information [task object] to output devices [event agent] allowing the user [event agent] to monitor [task action] the status [error element] of the pump [error object] or of the current delivery session [error context]	Zhang [149]	Pump Controller & Output Devices & User	TBD	TBD
FR6	The pump controller [event agent] must instruct [task action] output devices [event agent] to issue user [event agent]-perceivable [task parameter] alarms [error element] and alerts [error element]	Zhang [149]	Pump Controller & User	TBD	TBD
FR7	The pump controller [event agent] must record [task action] important [task parameter] data [task object] and events [task object] during pump use [event context] to facilitate clinical [task parameter] statistics [task object] and problem [task object] diagnosis [event action]	Zhang [149]	Pump controller	TBD	TBD
FR8	The pump controller [event agent] must have functionality to provide basal background insulin replacement [task action], delivered as a low [task parameter], continuous [task parameter] infusion of insulin [task object], periodically [task parameter] over a 24-hour interval [task parameter].	Zhang [149]	Pump Controller	TBD	TBD

Id	Requirement	Source	Allocation	Verification Method	Verification Allocation
FR9	The pump controller must have functionality to provide temporary [task parameter] basal insulin [task object] in proportion to physical conditions [task parameter] and activity levels [task parameter] of the user [event agent]. Once programmed [task action], a temporary [task parameter] basal overrides [task action] any ongoing normal [task parameter] basal at user [event agent] -specified rates [task parameter] for user [event agent] -indicated [task action] duration [task parameter].	Zhang [149]	Pump Controller & User	TBD	TBD
FR10	The pump controller [event agent] must have functionality to provide normal [task parameter] bolus [task action], a user [event agent]-defined amount of insulin infused immediately [task parameter] for covering food intake [task parameter] or correcting high blood glucose levels [task parameter].	Zhang [149]	Pump Controller & User	TBD	TBD
FR11	The pump controller must have functionality to deliver [task action] extended [task parameter] bolus similar to normal bolus but delivered over a period [task parameter] designated [task action] by the user [event agent].	Zhang [149]	Pump controller & User	TBD	TBD

Table 4.10: The safety requirements for the GIIP.

The primary elements of the goal structure we use for this case study are largely similar to those used in our prior case study (Fig. 3.20); however as we mentioned earlier, we derive our current goal structure from the hazard directed analysis, and functional decomposition patterns proposed by Kelly [64]. The skeleton of this structure is provided in Fig. 4.10.

In further pursuance of the argument structure we continue using Denney's lightweight process. We identify each requirement in the requirements table as one of the system safety related functions in the abstract goal structure presented in Fig. 4.10. Denney's prescribed consumption of the system/functional requirements table is summarized as follows:

- The satisfaction of each safety related requirement in the requirements table is included in the goal structure as a goal. If the requirement is an abstraction or is decomposed, a strategy to address each specialization or component as a goal is added to the goal structure.

- The source or each requirement qualifies the related goal as a context.

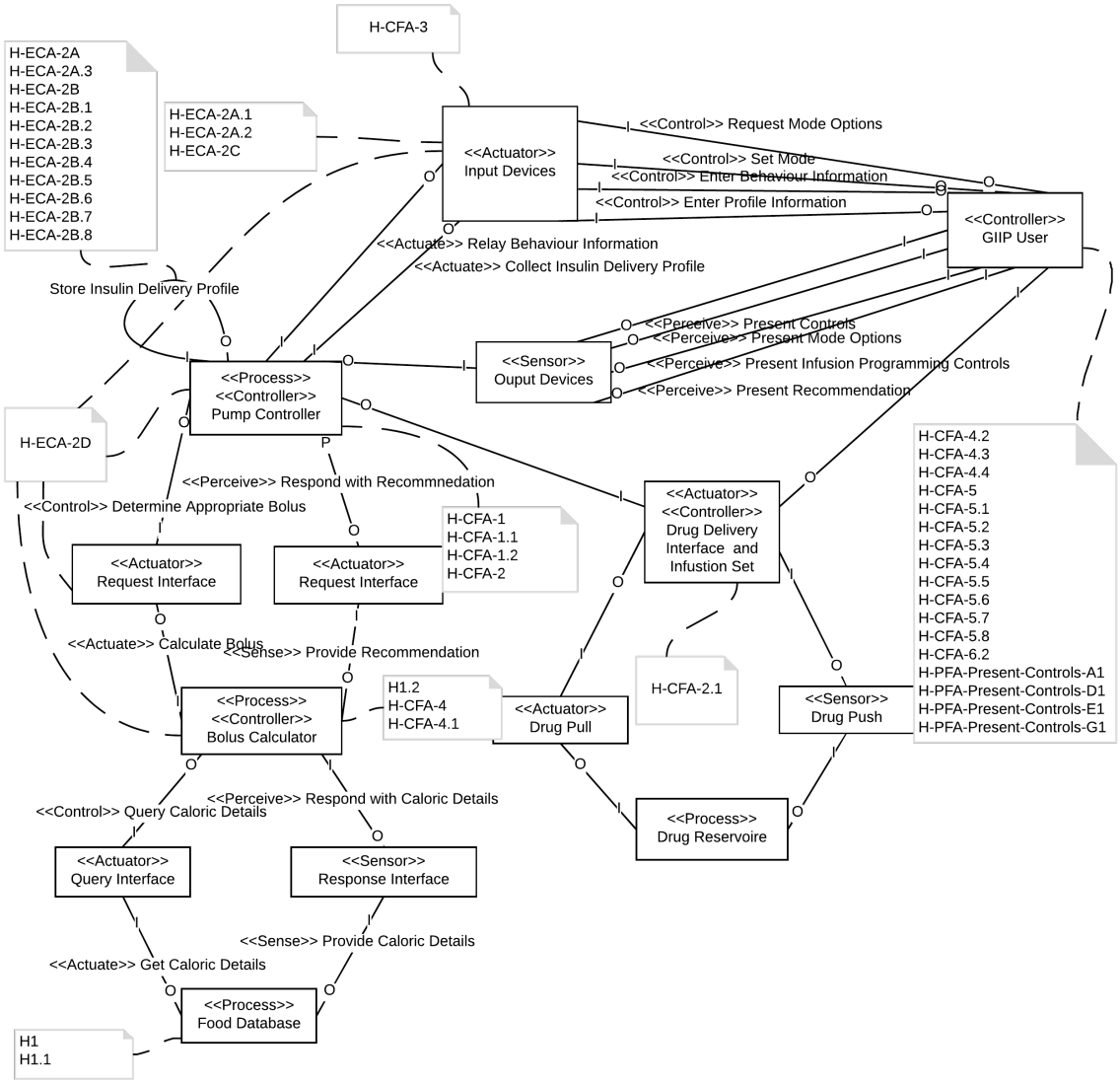

Figure 4.8: A view of the UTM illustrating the jobs and SCEMs for the programming activity as well as the hazard allocations.

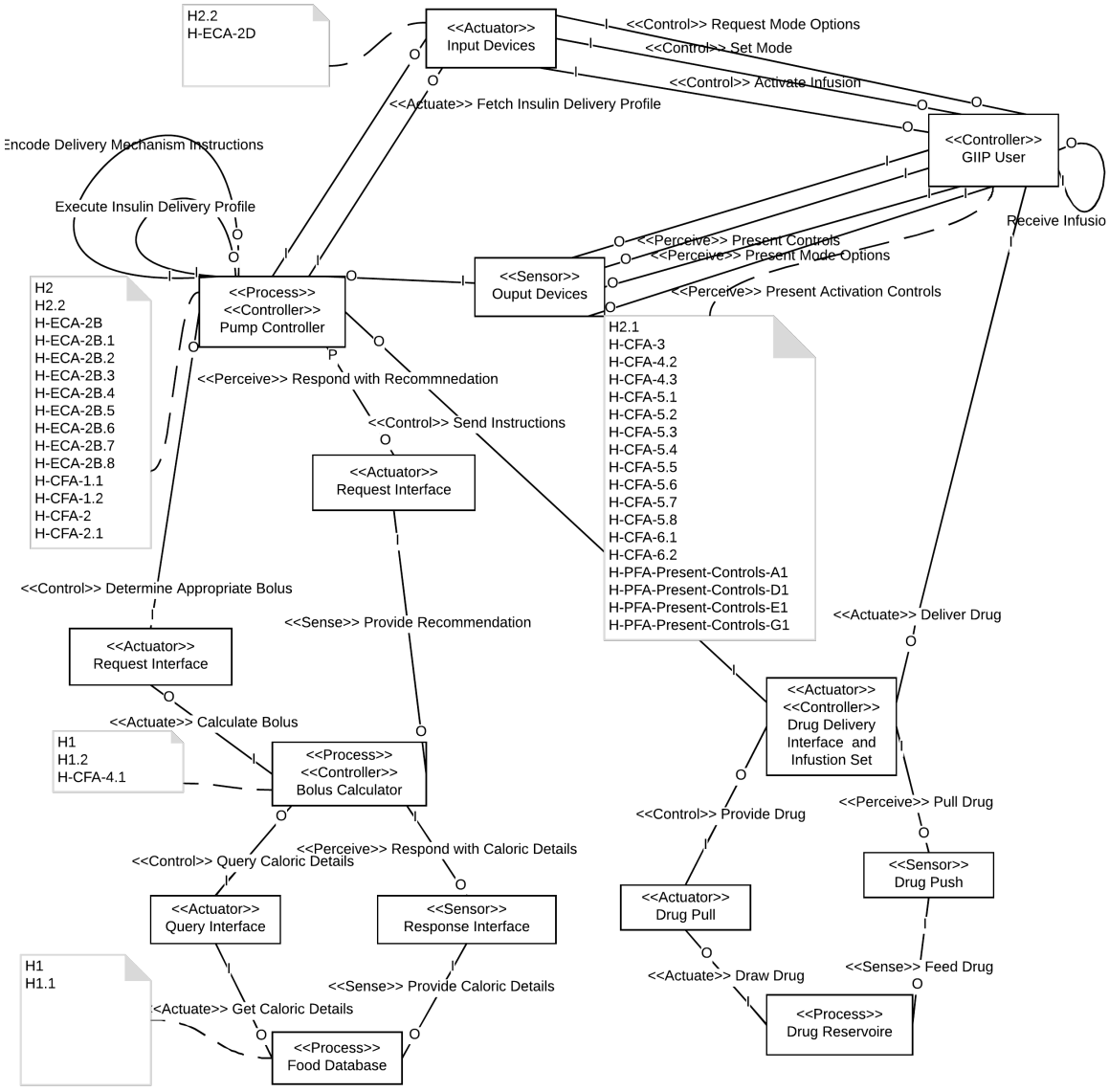

Figure 4.9: A view of the UTM illustrating the jobs and SCEMs for the delivery activity as well as the hazard allocations.

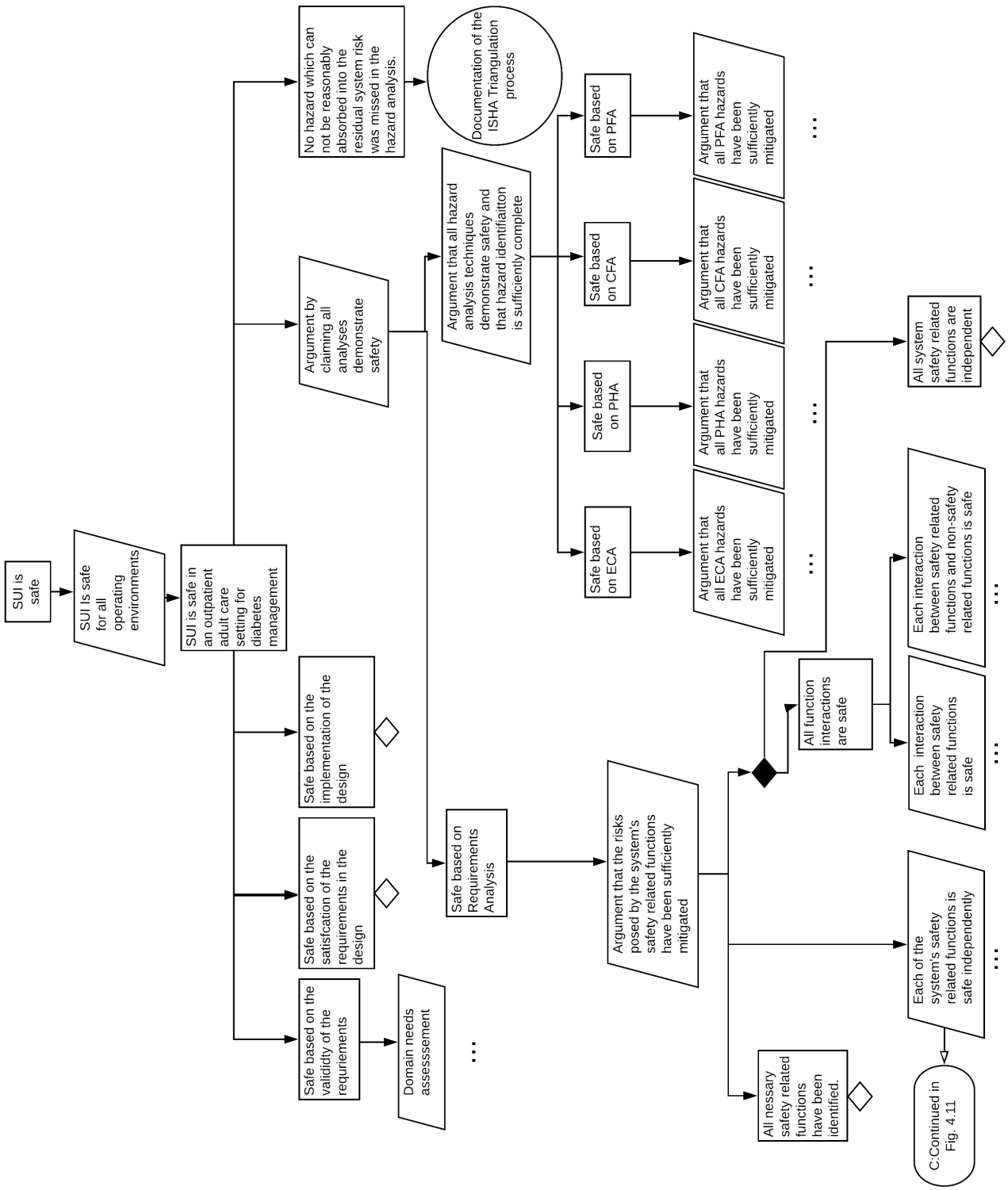

Figure 4.10: The skeleton of the assurance case goal structure developed for the GIIP case study.

- The allocation is included in the goal structure as either a strategy or a context
- The verification method is translated into a strategy to argue the safety of the related requirements with the verification allocation being the solution to that method.

The expansion of the abstract strategy for addressing the safety of the requirements analyzed in the ECA modelled in Fig. 4.10 is presented in Fig. 4.11.

To complete the argument structure for our current iteration of the ISHA process we continue with Denney's lightweight method of assurance case generation by leveraging his instruction on transforming hazard tables into goal structures. This process is summarized as follows with some extrapolation where Denney was unclear in his description of the process:

- For each hazard in the hazard table, a goal of Hazard is sufficiently mitigated is added to the goal structure. If the hazard is an abstraction or composition then appropriate subgoals are added.
  - If a derived hazard is identified in the hazard table, then for each derived hazard, a goal is added to express that the derived hazard has been mitigated.
  - If no derived hazards are identified, but a mitigation is, then a strategy of argument by mitigation is added to the goal structure.
  - If no cause, derived hazard or mitigation is identified for a hazard but a safety requirement exists to mitigate the hazard then a strategy arguing safety by way of the satisfaction of the safety requirement is added to the goal structure.
  - If no cause, derived hazard or mitigation is identified and there is no mitigating safety requirement, then the hazard is described as being absorbed into the residual risk of the system.
- The high level strategy used to argue safety is that assurance is provided over all identified hazards.

Abridged expansions of the abstract strategy for addressing the hazards identified and analyzed in the PHA, CFA, and PFA are modelled in Fig. 4.10 is presented in Figs. 4.12, 4.13, 4.14, 4.15.

4.11 Generate Recommendations

The recommendation generation phase of ISHA is by necessity flexible and non-prescriptive. Many widely varying recommendations can arise from hazard analysis. As the purpose of our

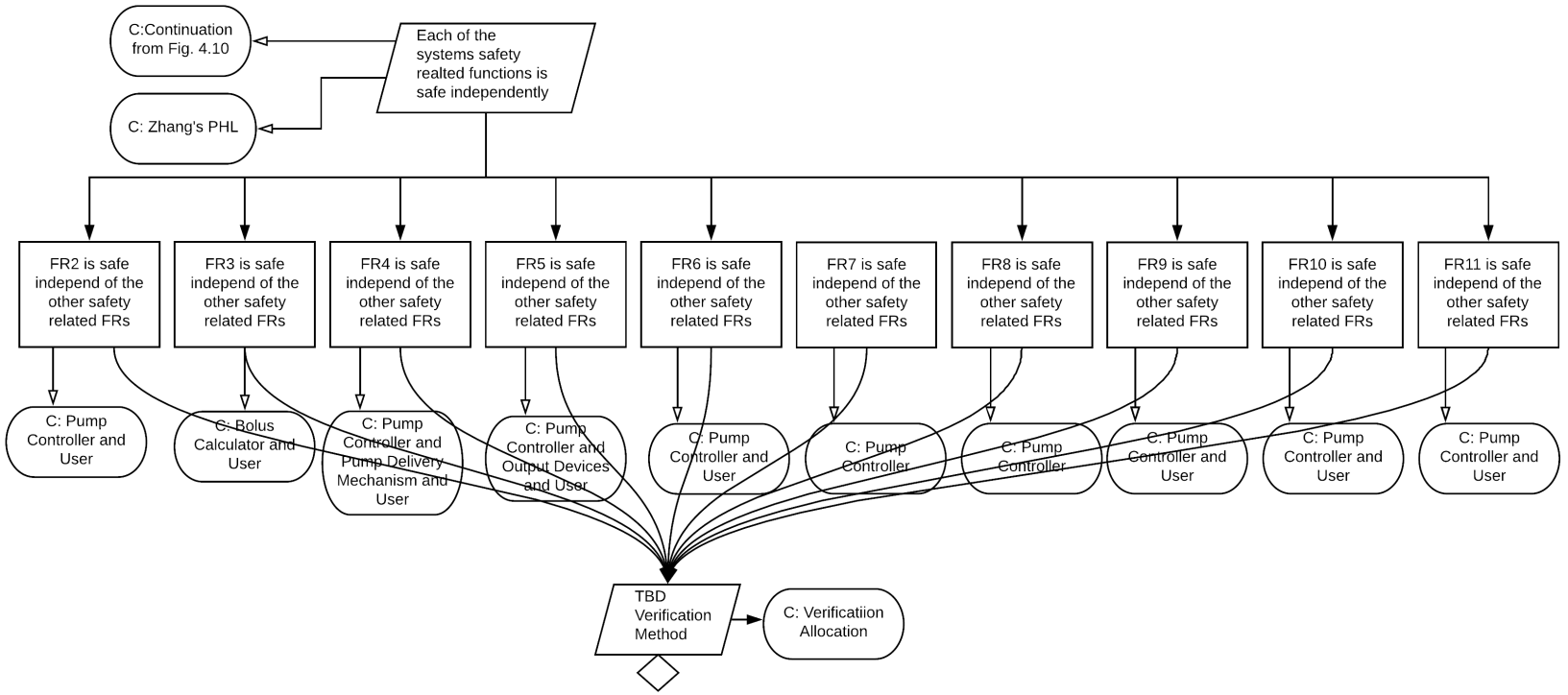

Figure 4.11: A goal structure for the strategy of arguing the safety of the GIIP based on the safety of its independent safety related requirements.

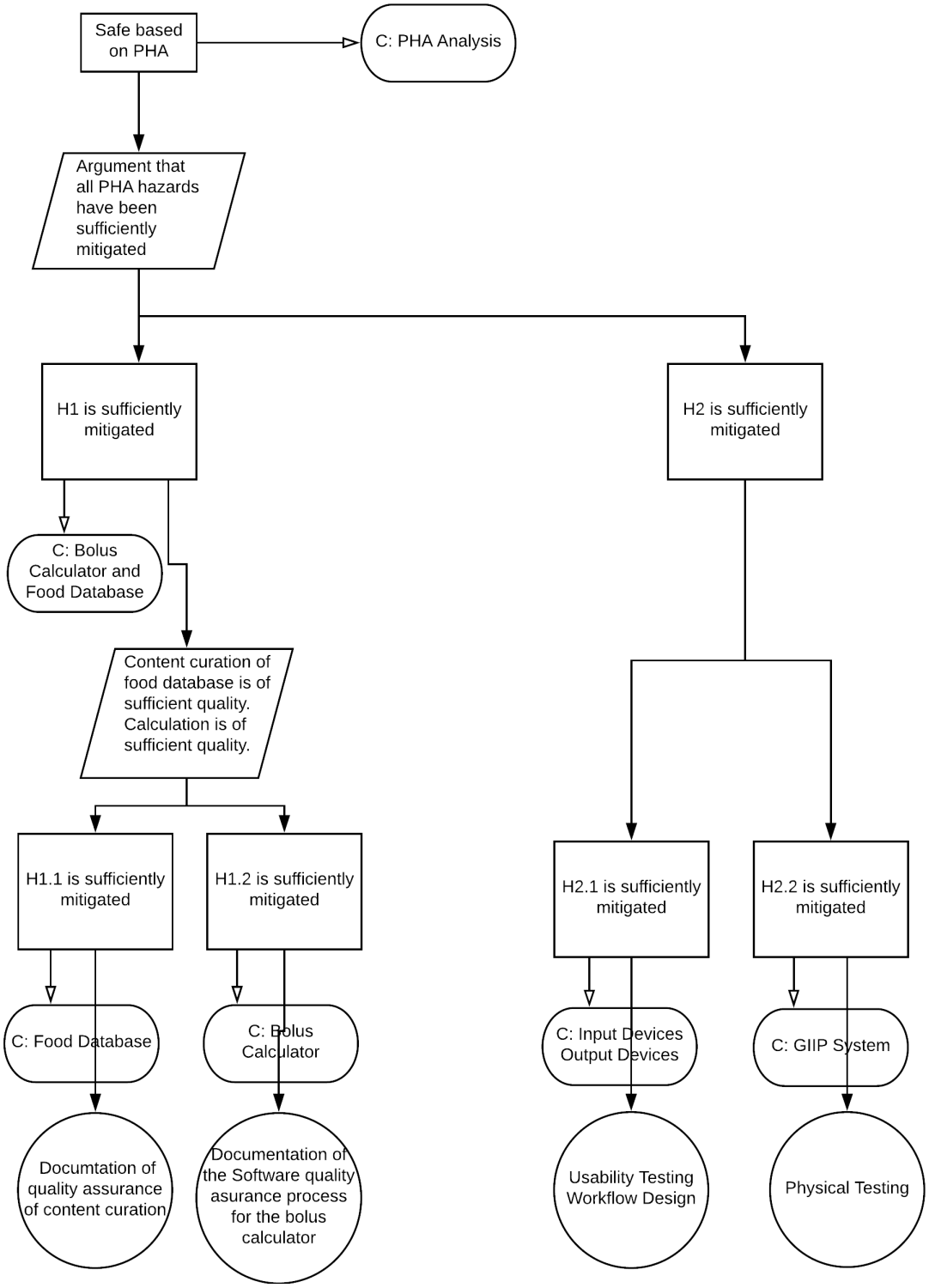

Figure 4.12: The goal structure for the hazard directed assurance case for the hazards identified in the PHA

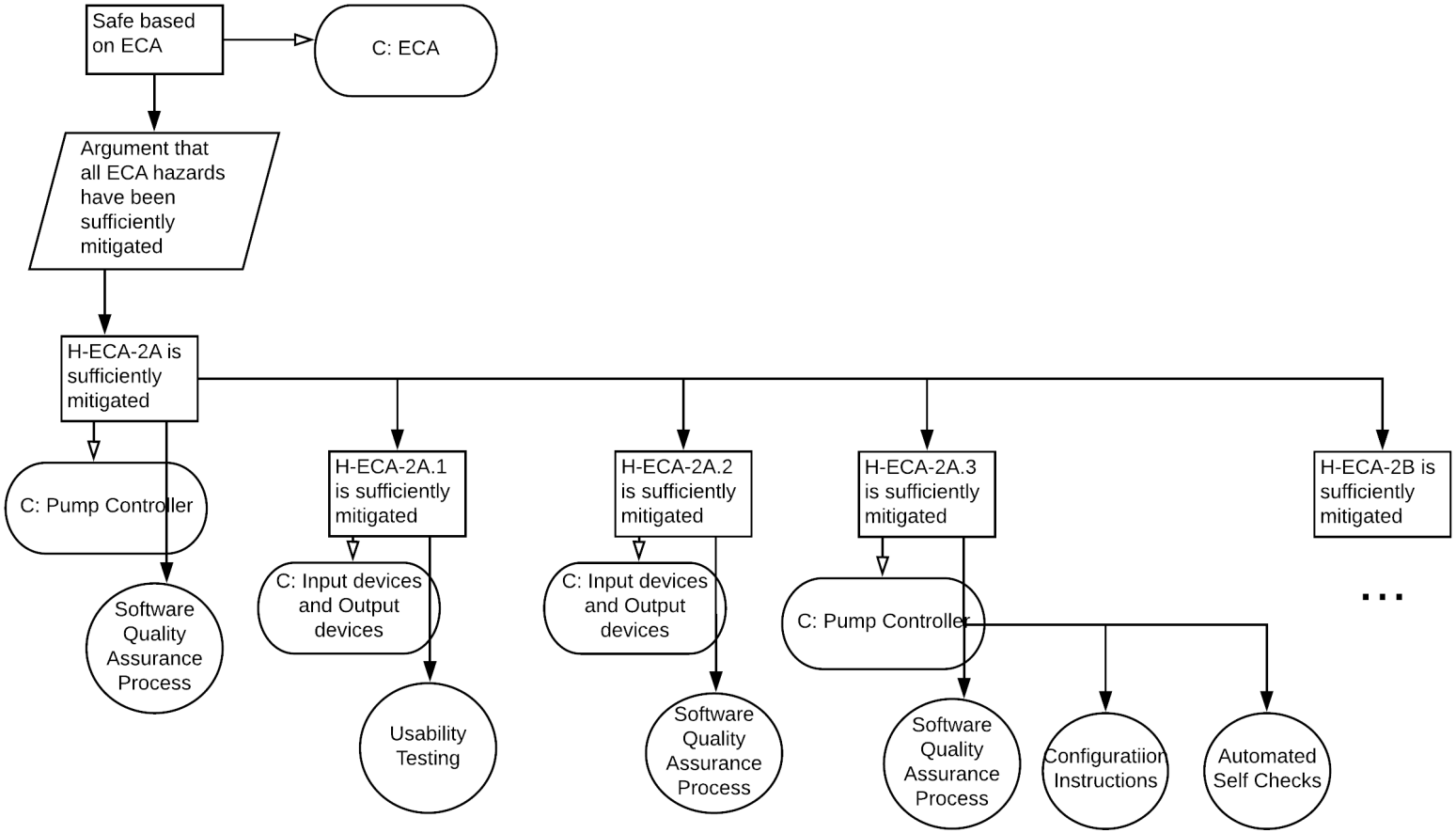

Figure 4.13: The goal structure for the hazard directed assurance case for the hazards identified in the ECA

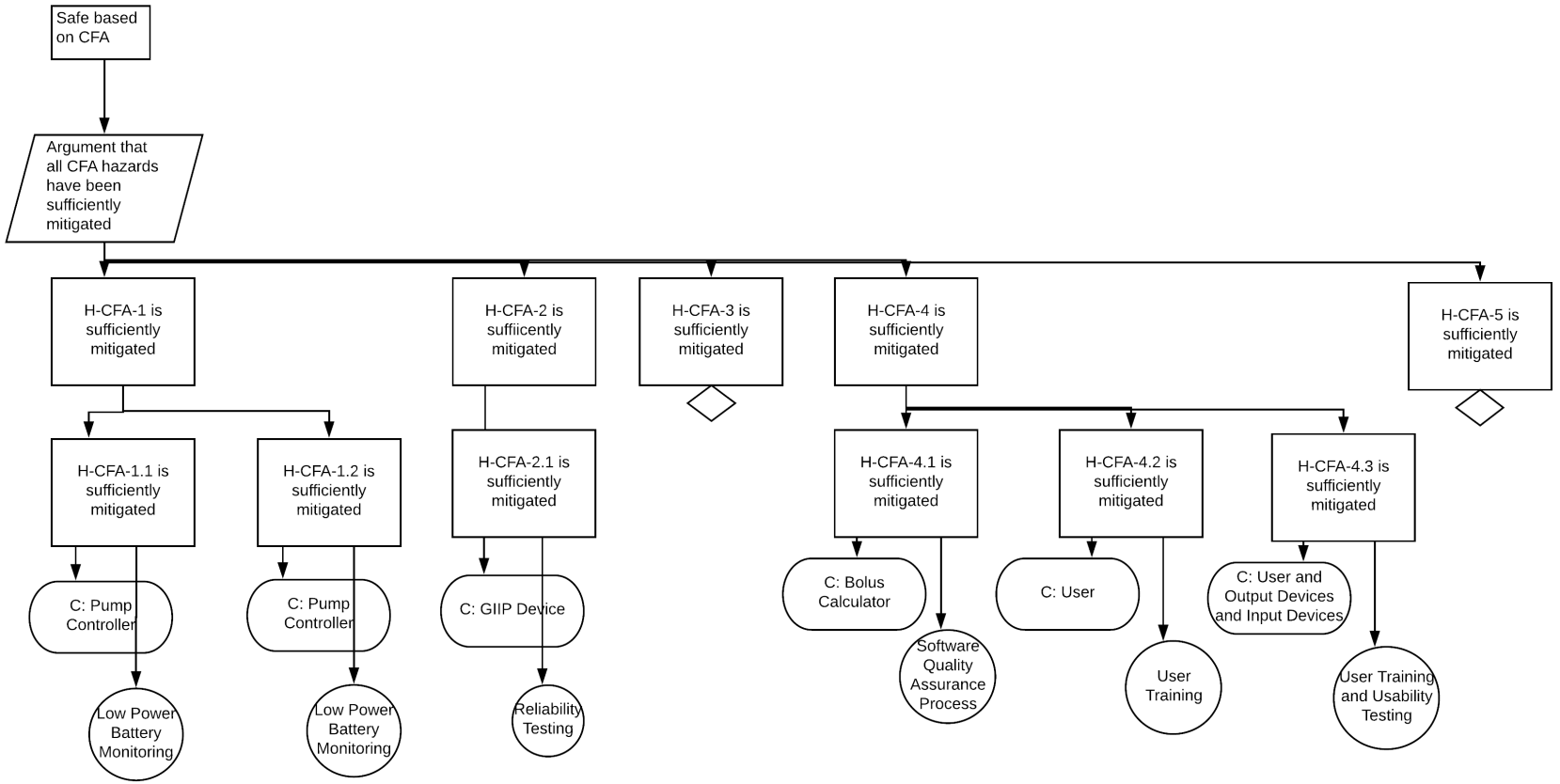

Figure 4.14: The goal structure for the hazard directed assurance case for the hazards identified in the CFA

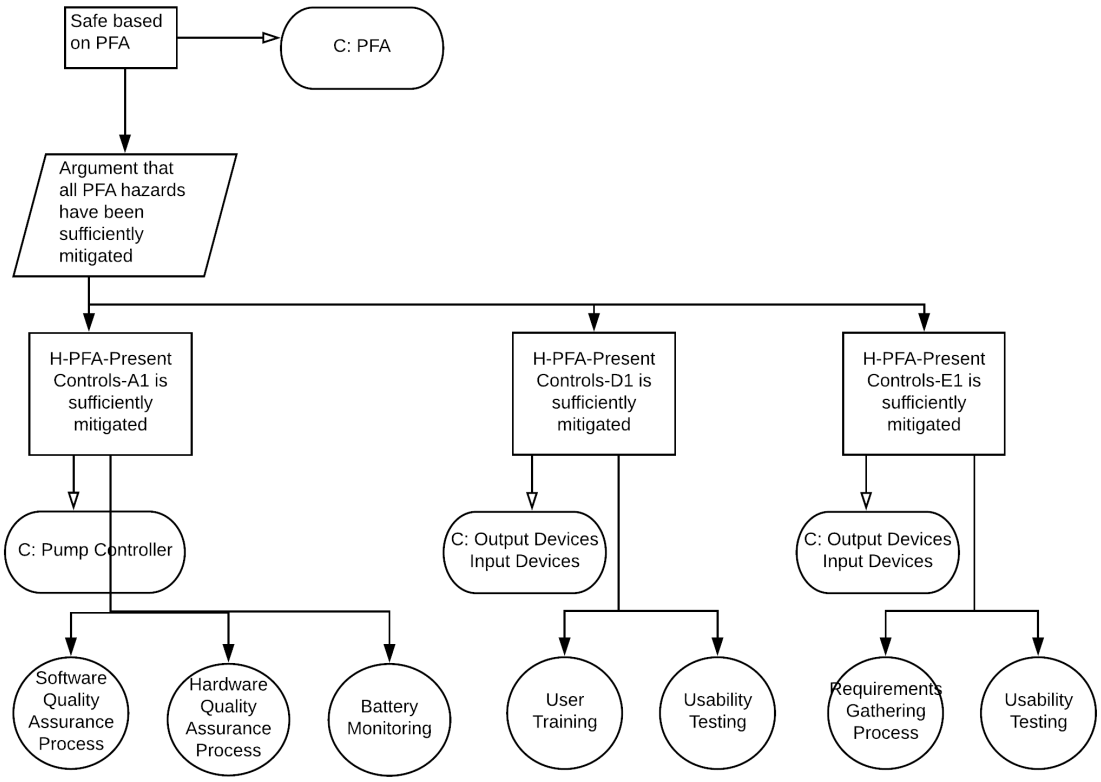

Figure 4.15: The goal structure for the hazard directed assurance case for the hazards identified in the PFA

discussion is to introduce the method and not perform a thorough analysis, we only highlight some of the types of recommendations which might arise when performing an ISHA. In the pragmatic execution of the activity, analysts will find this to be an exercise in synthesis. Some recommendations may be to mitigate hazards. This type of recommendation requires that a prioritization scheme be used to identify which hazards should be addressed, and if many need to be addressed then which ones should be addressed first. We have described a number of techniques which can be used for this purpose in different contexts throughout our description of the method and so we do not pursue these further here. Another type of recommendation which may arise from analysis is process recommendations. For our case study for example, it is apparent from the assurance case goal structure that there is a strong need for comprehensive software quality assurance documentation to demonstrate the safety of the GIIP. Further, usability testing - if considered as a distinct activity from software quality assurance - was another theme that arose frequently. As a last point, the allocation of the functional requirements revealed at least one missing requirement in Zhang's prescribed set. Zhang specified that the pump controller must signal to the user interface output devices what the state of the pump itself and the current delivery session were. He did not indicate in any way that a formal requirement had been expressed that the user interface should consume this output. Though this example is an asymmetry which would likely be mitigated by software developers who would infer the later requirement, the discovery highlights the value of performing multi-pronged analysis to identify the range of system hazards and other requirements and design flaws in the SUI.

Chapter 5

A Formal Information Model for ISHA

In this chapter we provide a formal information model for Information System Hazard Analysis (ISHA). We focus our attention on the expression of hazards, the management of risk, the modelling of the Universal Triangulation Models (UTMs) which are used to represent the System Under Investigation (SUI), and the documentation of assurance cases. The portion of the model which represents the definition of hazards is largely novel and independently developed in this dissertation; it does however leverage an extant terminology and expression model from the Object Management Group (OMG)'s Structured Assurance Case Metamodel (SACM). The SACM is a model which was developed to describe Kelly's [63] assurance case framework. The portion of the model which illustrates the structure of the UTMs which are used in the ISHA method is novel in that we independently synthesized and formalized the concepts put forward by Leveson [76] in her System Theoretic Accidents Models and Processes (STAMP) framework and by Hollnagel [52] in his Functional Resonance Analysis Method (FRAM) framework to create it. Finally, the portion of the model which is used to document assurance cases is drawn from the SACM [115].

We begin by first developing a series of Functional Requirements (FRs) for software tooling to support the ISHA process. We then synthesize the SACM with our *hazard metamodel* into a minimal Universal Modelling Language (UML) class diagram. We then discuss the information model. Finally, using the defined information model, we present a series of information system operations which could be used during ISHA analysis as patterns of common hazard scenarios. "Design patterns, identify, name and abstract common themes." [38] The design pattern idea is attributed in a general context to Alexander [2], but the concept was popularized in software by Gamma [38, 37].

We express these using graph transformations between UTM instances.

5.1 Functional Requirements

The selected FRs of the ISHA support software are derived from the method as documented in Chapters 3 and 4. They span the requirements sourcing, hazard identification, model sourcing, UTM construction, Event Chain Analysis (ECA), Component Fault Analysis (CFA), Process Fault Analysis (PFA), triangulation and assurance case documentation phases. We list the FRs below:

FR1 The system must document the requirements which are gathered in the source requirements phase of the method.

FR2 The system must document the Preliminary Hazard List (PHL) which is constructed in that phase of the method.

FR3 The system must document the multiplicity of models used to represent the SUI.

FR3.1 The system must document the UTM

FR3.1.1 The system must document the SUI component's as recorded in the UTM.

FR3.1.2 The system must document the breadth of ports that are held by each components of the UTM.

A. Output

B. Input

C. Resource

D. Precondition

E. Timing

F. Control

FR3.1.3 The system must document the breadth of component stereotypes that can be held by components of the UTM.

A. Controller

B. Actuator

C. Process

D. Sensor

FR3.1.4 The system must document the breadth of duties recorded in the UTM.

FR3.1.5 The system must document the breadth of duty stereotypes than can be held by duties in the UTM.

A. Control

B. Actuate

C. Sense

D. Perceive

FR3.1.6 The system must document the connectivity of the duties including the source and target components as well as the source and target ports as recorded in the UTM.

FR3.1.7 The system must document the viewpoints recorded in the UTM.

A. The system must document viewpoint component composition.

B. The system must enforce the constraint that viewpoints be composed of complete control loops consisting of connected controllers, sensors, actuators and processes as described in Chapter 3.

FR3.1.8 The system must document the association and annotation of hazards with components, duties, and constraints.

FR3.1.9 The system must document the association and annotation of constraints with components and duties in the UTM.

FR3.1.10 The system must document the association and annotation of Safety Constraint Enforcement Mechanisms (SCEMs) with components and duties as recorded in the UTM.

FR3.1.11 The system must document the model membership of components, duties, connections, constraints and SCEMs.

FR3.1.12 The system must allow analysts to establish the risk scores used in the method's execution.

FR3.1.13 The system must document the artifacts necessary to record the assessed risks posed by the SUI.

FR1.13.1 The system must document the sequence of detectability scores for the risk assessment.

FR1.13.2 The system must document the sequence of occurrence scores for the risk assessment.

FR1.13.3 The system must document the sequence of severity scores for the risk assessment.

FR1.13.4 The system must document the sequence of risk scores for the risk assessment.

FR1.13.5 The system must document the complete set of Risk Assessment Codes (RACs) used in the ISHA execution.

FR3.1.14 The system must document the association of risk scores with hazards.

FR3.2 The system must document the other models, besides the UTM, collected during the *Source Model* phase of the method. It is not *required* to document the components of these models. This function may be achieved to a satisfactory degree through their synthesis into the UTM.

FR4 The system must document the hazards identified in the ECA phase of the method.

FR5 The system must document the hazards identified in the CFA phase of the method.

FR6 The system must document the hazards identified in the PFA phase of the method.

FR7 The system must support the triangulation between the ECA, CFA and PFA hazards.

FR8 The system must document the generated assurance case for the SUI.

(a) The system must document the recorded claims.

(b) The system must document the recorded arguments.

(c) The system must document the recorded evidence.

(d) The system must document the relationships between evidence arguments and claims.

5.2 Hazard Metamodel Model

The information model we developed through the synthesis of the efforts of the OMG on the SACM, and our own work in describing hazards is presented in Fig. 5.1. The model can be decomposed into seven parts:

1. Structured Assurance Case Base Classes
2. Structured Assurance Case Terminology Classes.
3. Argumentation Metamodel
4. Artefact Metamodel
5. Hazard Metamodel
6. System Structure Metamodel
7. Risk Metamodel

The *Structured Assurance Case Base Classes* provide the abstraction foundation for the SACM. The *Structured Assurance Case Terminology Classes* support the terminology needs of the ISHA method and of assurance case construction more broadly. The *Argumentation Metamodel* supports the documentation of assurance case arguments. The *Artefact Metamodel* supports the documentation of assurance case assets. The *Risk Metamodel* supports the parts of the ISHA process where risk is assigned to each of the identified hazards. The *Hazard Metamodel* provides the structure to document the hazards identified through the application of the ISHA method, and the triangulation of hazards between the PHL, ECA, CFA, and PFA. Finally, the *System Structure Metamodel* provides the facility to document the construction of the UTM and the recording of other models of the SUI identified in the *Source Model* phase of the process.

5.2.1 Structured Assurance Case Base Classes

As stated, the SACM base classes form the core conceptual component and relationship model of Kelly's [63] assurance case framework. Summarizing, this part of the model consists of the *ModelElement*, *Description* and *UtilityElement* classes. The *UtilityElements* are abridged in our model (Fig. 5.1) with a more complete set expressed in the SACM standard [115]. The *Description* class is used by the *Hazard* element of the hazards part of the information model as a component that leverages standardized terminology and nomenclature for normalized expressions of the hazards identified in the execution of ISHA.

5.2.2 Structured Assurance Case Terminology Classes

The SACM terminology classes are used to capture the application of standardized terminology and nomenclature in the record of the ISHA analysis. The original OMG standard for the SACM does not sufficiently account for the application of external terminologies or nomenclatures. The original standard does provide for an external citation for terms, but does not go so far as to demonstrate how terms from multiple terminologies could be fully cited. It also does not consider the use of standardized expression templates like the one proposed by Phillips and Gong [105]. We have adapted the terminology elements described in the SACM standard to account for the use of such sources as these are common in medicine - our domain of application. These changes are highlighted in Fig. 5.1 with the use of bold borders on the modified and added classes. In particular, we provide

1. references to terminology and nomenclature identifiers
2. references to unique external identifiers for terms and expressions

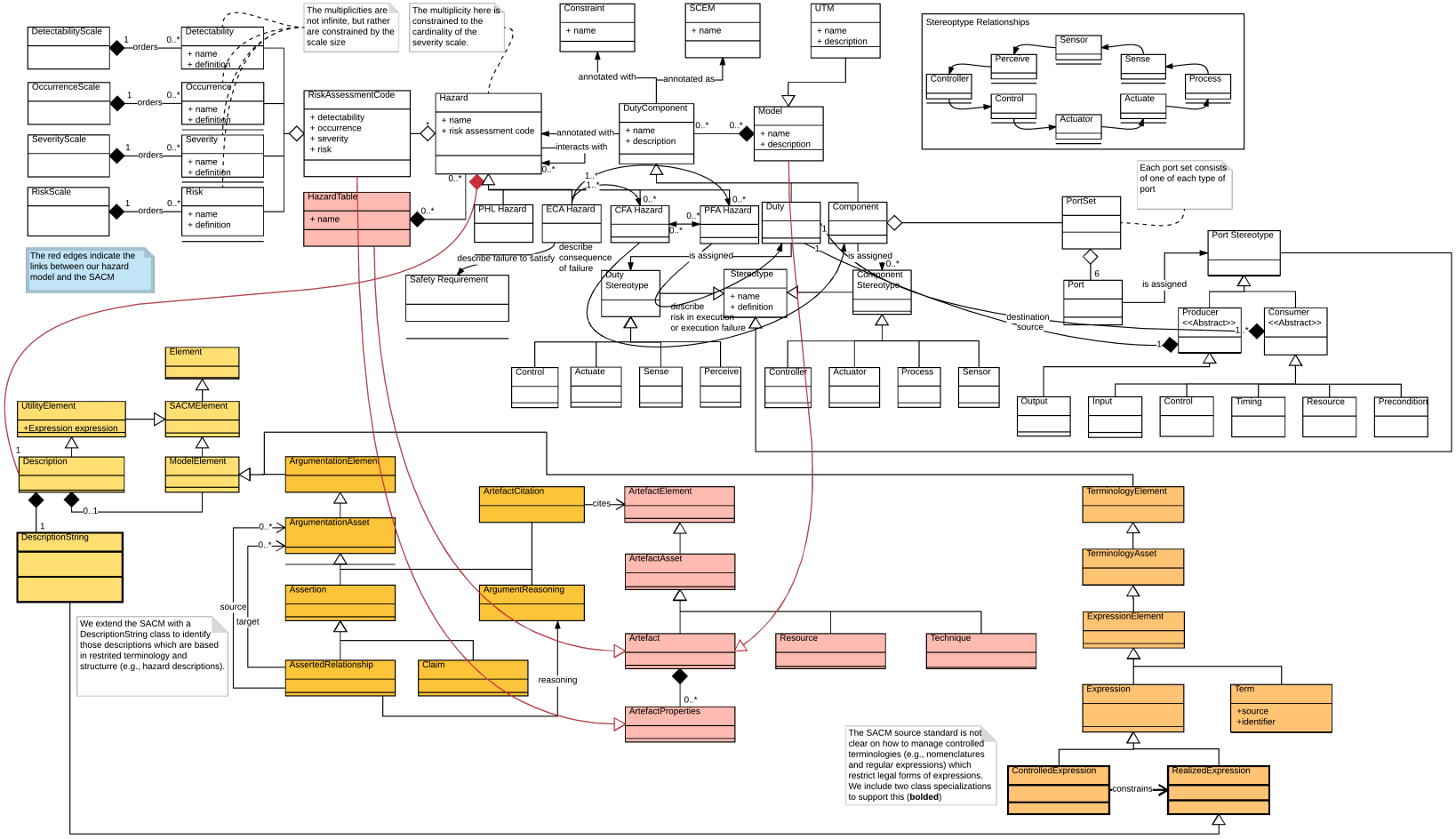

Figure 5.1: An information model to support the execution of the ISHA process. The model is constructed of two parts: the SACM for documenting assurance cases which is colored by package, and our novel *hazard metamodel* which was developed to support the FRs of the ISHA method.

3. a variation in semantics for the expression element of the model

   (a) we use a parent concept of Expression - ControlledExpression - to represent the set of allowable expression structures - much like a regular expression.

   (b) we then use the child Expression concept to represent the concrete instantiations of the ControlledExpression

5.2.3 Argumentation Metamodel

We summarize the *Argumentation Metamodel* as being composed most importantly of the *Claim*, *ArgumentReasoning*, *AssertedRelationship*, and *ArtifactAssetCitation* classes. As described before, an assurance case consists of a series of claims which are argued with the support of evidence. These structures facilitate the documentation of goal structures which are developed to express the documented assurance cases which are developed with the ISHA method.

5.2.4 Artefact Metamodel

We summarize the *Artefact Metamodel* as most importantly including *Resources*, *Techniques*, *Artefacts* and *Artefact Properties*. These structures are used to document the evidence and artifacts (e.g., the hazard table) which are identified or created during the course of an execution of the ISHA method. The remaining classes in this portion of the ISHA information model provide for abstraction and citation of the significant classes we identified as being central. The *Model* table which is novel in our development of the ISHA information model is also appended to the native *Artefact Metamodel* as it is a more fitting assignment that its inclusion elsewhere in the information model.

5.2.5 Hazard Metamodel

The *Hazard Metamodel* is comprised of only the *Hazard* class with the *Hazard Table* class being ascribed to the *Arefact Metamodel* part of the information model. The first documents identified hazards while the second aggregates them in a traceable artefact. Interaction relationships are also included over the hazard class. This formal modelling of hazard interactions allows analysts to record interactions which are subsequently informally used to inform the assignment of risk factors and ultimately RACs to the identified hazards.

5.2.6 System Structure Metamodel

The *System Structure Metamodel* is the component of the ISHA information model which captures the structure of the UTM. It includes classes for the *Components*, *Ports*, *Duties*, *SCEMs*, *Constraints*, *Stereotypes* and relationships necessary for representation of the UTMs and SUI models more generally.

5.2.7 Risk Metamodel

The risk metamodel is composed of the classes required to express the level of risk posed by each of the hazards identified in the ISHA method. The *Hazard* class is composed of one RAC per level on the severity score as described in Chapters 3 and 4. The RACs are in turn composed of one element from each of the risk scales - *Severity*, *Detectability* and *Occurrence*.

5.3 Universal Triangulation Model Patterns

Experience in applying the ISHA method will reveal recurrent UTM patterns to analysts. The concept of a pattern was first introduced by Alexander [2], but was popularized in the domain of software engineering by Gamma [38, 37]. A pattern identifies, names, and abstracts recurrent themes in a domain. The patterns we present here commonly revolve around hazard themes discussed thus far in this dissertation including, integrity, usability and availability. We discuss a series of patterns we have encountered in our applications of ISHA including the cases studies described in Chapters 3 and 4, as well as those described in our prior case studies [83, 140]. The patterns cover both system designs which have the potential to pose hazards as well as those which mitigate hazards. We express these transformations using a variation of the software design pattern structure proposed by Gamma [37]. We will begin by introducing Gamma's structure and our variation of it, and we will follow with descriptions of our patterns including UTM based models of their structures.

5.3.1 Pattern Description Template

Gamma introduced design patterns for Object Oriented Programming (OOP) in his 1995 book on the topic [37]. He motivates software design patterns as an avenue to standardize approaches to common software engineering problems. He establishes a structure for describing these patterns before using the structure to communicate a series of common patterns he and his colleagues had identified through their experience and research. The

structure Gamma uses for the communication of these patterns consists of a thirteen point structure which covers identification, purpose, motivation, applicability, structure, consequences, implementation and examples. As the full structure proposed by Gamma does not provide significant value in work as nascent as ours, we use the simplified structure outlined below:

1. **Pattern Name:** An identifier to facilitate communication about the pattern.
2. **Also Known As:** Aliases of the pattern. [optional]
3. **Motivation:** A description of the problem which needs to be solved.
4. **Applicability:** A description of the context necessary for the pattern to be applicable, as well as an identification of key problem characteristics which designers can identify to recognize when they should adopt the pattern.
5. **Structure:** A formal representation of the structural aspects of the pattern model (e.g., class diagrams, sequence diagrams, or state charts).
6. **Consequences:** A description of how the pattern satisfies its *motivation*, as well as a description of benefits and drawbacks of the pattern's application.
7. **Implementation:** Guidance on how to implement the pattern including warnings about issues that might arise during the pattern's application.
8. **Practical Examples:** A series of descriptions of real scenarios in which the pattern could be applied to address the problem the pattern is intended to solve.

5.3.2 Patterns

In this section we introduce the most important patterns we have identified in our work. We order these patterns based on a rough subjective measure of the patterns' importance. The six patterns we introduce are:

1. Complex Operation
2. Delegation of Responsibility
3. Evolution
4. Peripheral,
5. Inversion of Control,

6. Transition of Responsibility

Pattern Name: *Complex Operation*

Also Known As: *Complex Control*, *Complex Sensing*, *Complex Actuation*, *Replicated Control*, *Duty Monitoring*.

Motivation: The motivation for the *complex operation* pattern is to model SUIs in which multiple components contribute to the regulation/guidance, monitoring or actuation of the controlled *process*.

Applicability: The context of application of the *complex operation* pattern is when there are multiple sources of *control*, *actuate*, or *sense* actions influencing the trajectory of a controlled *process* or, in the case of the *complex sensing* pattern, the state of the *controller*'s *process model*. Key characteristics of these situations are that the behaviour of a controlled *process* appears to be unpredictable suggesting the influence of external environmental factors which are not yet modelled as *controllers*, or situations in which multiple SUI components are involved in the *sensing* or *actuation* of the controlled *process*.

Structure: The structure of the *complex control* pattern - as opposed to the *complex sensing* or *complex actuation* patterns - is modelled in Fig. 5.2 using the same modelling paradigm we applied in the description of the UTM in chapters 3 and 4. The pattern is expressed as a double control loop with a central *process* being guided by a pair of peripheral *controllers* via those *controllers'* respective *actuators* and *sensors*. Standard relationships and connections model the interactions between the *process*, *sensors*, *controllers* and *actuators* with the critical path communications being expressed through connectivity between *output* and *input* ports. The communication between the *actuators* and the *process* however is slightly different. The connectivity here is between the *output* port of the *actuators* and the *control* port of the *process* indicating that the *process* is regulated through the *actuators* by signals which originate in the *controllers*. This indicates that the actuate signals generated by Component B and Component G in this model are not critical path signals - they are not required for the operation of the *process*. Finally, an interaction which is not stereotyped between the controlled *process* and itself has been added from the *output* to *input* ports modelling continuous execution/evolution of that *process* thus representing it as a process which is in continuous execution - e.g., the healthcare process of a patient.

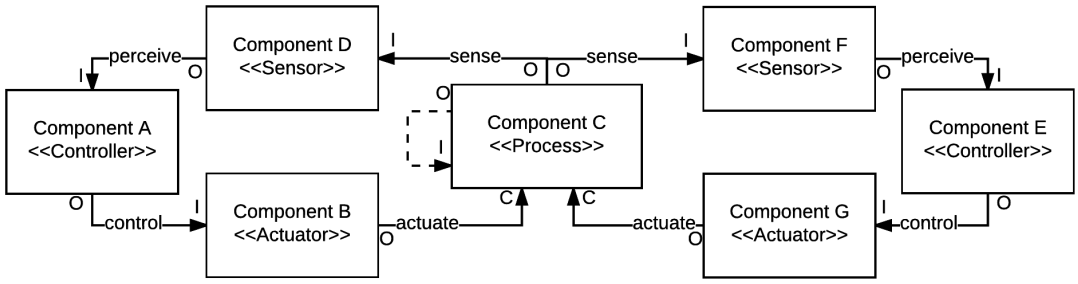

Figure 5.2: This figure illustrates a viewpoint for the *complex control* pattern using the paradigm developed in chapters 3 and 4 for modelling the UTM.

Consequences: The *complex operation* pattern is required when decomposition of the control structure in the SUI must be modelled. The pattern provides analysts with a mechanism to investigate the constructive and destructive interference which arises when *controllers* magnify or mitigate their respective *control* actions, or to investigate the impact of conflicting or reinforcing *sensor* data on the *controller's process model*, or to investigate the impact of conflicting or reinforcing *actuator* signals on the trajectory of the controlled *process*. The pattern also provides a mechanism to investigate the consequence of timing for competing *control* actions on the controlled *process*. Another application of the pattern in the *replicated control* instantiation is that it facilitates the exploration of the replication of control actions for application to disparate *processes*. The *replicated control* instantiation decomposes or duplicates a *process*.

However, the complexity of the instantiation of this pattern yields much larger models than may be necessary. It may be better in some situations to use a *delegation of responsibility* pattern modelling component behaviour as a process which is internal to a given component rather than something which results from a more granulated partition of the larger SUI. Such an approach may allow analysts to constrain the model complexity while still developing the concrete model necessary to identify the most critical SUI hazards. Though hazards are often found as a result of detailed analysis, unnecessarily verbose models complicate investigation.

Implementation: The implementation of the *complex operation* pattern is straightforward and can be applied based on the sample *complex control* application provided in Fig. 5.2 with appropriate modifications for the *complex sensing*, *complex actuation* and *replicated control* variations, and thus we do not discuss this further. There is however value in discussing warning signs that the pattern is being misused or overused. As we will discuss further in section 7.4, scoping granularity can be challenging in the application of

the ISHA process. Analysts may be able to recognize when this challenge is compromising analysis if they observe that a substantially detailed model of the control of a *process* has been developed while the model of the remainder of the SUI remains relatively sparse. Though hazards are often identified in detailed analysis, exhaustive investigation of singular dimensions of the SUI to the neglect of the analysis of the remainder is a sure way to provide an imbalanced and arguably incomplete investigation.

Practical Example: An example of the *complex control* pattern is identified in the modelling of *duty monitoring* in the SUI. *Duty monitoring* is a potential mechanism which can be used to mitigate hazards which arise from the degradation of communication integrity either through content or timing - sequence, duration or timing gates. We model the application of the *complex control* pattern in the context of *duty monitoring* in Fig. 5.3.

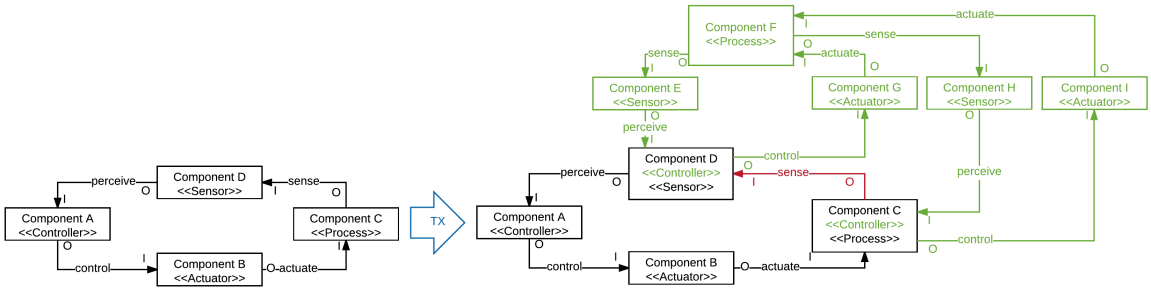

Figure 5.3: A viewpoint pair modelling the application of the *complex control* pattern to transform a simple sensing behaviour into a monitored communication. The red elements have been removed from the left hand side (LHS) model while the green elements have been added to the right hand side (RHS) model.

We choose to demonstrate this use of the *complex control* pattern using a transformation rather than a singular static representation to highlight the difference in the two representations of communication before and after the pattern's application. In the figure we model an atomic control loop on the LHS and indicate the transformation from this base model to the post-pattern-application state on the RHS using a blue transformation arrow (TX).

On the RHS we remove the original *sense* edge between the *process* and the *sensor*. We replace this communication with an instantiation of the *complex control* pattern on the LHS. In the RHS of the transformation a new *process* is created whose *control* is shared by the original two communicators - Component C and D form the LHS. Each of these original components is assigned an additional *controller* stereotype and is connected to the new shared *process* by respective pairs of *actuators* and *sensors* in the RHS. The connectivity of components throughout this pattern on both the RHS and the RHS is expressed here as using the standard *input* and *output* ports, but the *complex control* pattern may be adapted to support the modelling of non-critical path communications by changing the final port on the communication receiver on both the LHS and the RHS to any other appropriate

port (i.e., *time*, *control*, *resource* or *precondition*). Further, this model of the pattern demonstrates only how to apply it to a *sense* communication. Similar transformations can be created for the other communication edge types (i.e., *control*, *actuate*, and *perceive*).

One specific application of the *complex control* pattern is motivated by the need to model portions of the SUI in which the communication between components is monitored. Poor communication integrity either as a consequence of low quality information transmission or as a consequence of poorly timed communication is a known safety issue [76]. An example of the *complex control* pattern is observable in the STAMP EMR model (Section 3.5.2). We have argued that the STAMP EMR model is a representation of Clinical Information Systems (CIS) which is both realistic and pragmatic [142, 143, 81, 82, 83, 29]. In STAMP EMR, the controlled process for the *complex control* pattern is the *Contextual Health Care Process*. The architecture of CIS, as expressed in STAMP EMR, includes a pair of *controllers*, the *Technical Administration* and the *Health Care Process Controller*. These two controllers use their respective *actuators* and *sensors* to share and sometimes compete over control of the *Contextual Health Care Process*. A further example is provided by Leveson in Chapter 8 of her book on STAMP [76]. In this chapter she uses a model of the control of a physical process by a concurrent human controller and and automated controller. She describes such situations as arising any time a human controller is operating a process through both direct control and also through the use of an automated controller. She provides examples of such situations including the piloting of aircraft and the operation of oil refineries. Leveson's representation of these situations is adapted in Fig. 5.4. Her depiction of these systems deviates slightly from the pattern we describe in that it appears that the actuator and sensor used to actuate and sense the process under shared control are the same actuator and sensor used by the two controllers. First, in our pattern it is not necessary for these to be difference actuators and sensors. Second, Leveson's description and original depiction of the architecture indicates that the actuators and sensors used by the controller may in fact be different. Our choice adaptation looses some of this nuance in exchange for simplicity.

A specific consequence of the application of the *complex control* pattern in this context is that the pattern satisfies its motivation by facilitating the modelling of component communication through a shared *process* much like the "shared memory" strategy for facilitating Inter Process Communication (IPC) [146] thus providing a concrete model for evidence based evaluation. A primary benefit of the application of the *complex control* pattern in this fashion is that it facilitates an in depth analysis by the investigators of communications which is a known hazard hot spot. The application of the pattern in this fashion however, requires a decomposition of both sides of the communication simultaneously leading to a significantly more verbose model of the SUI.

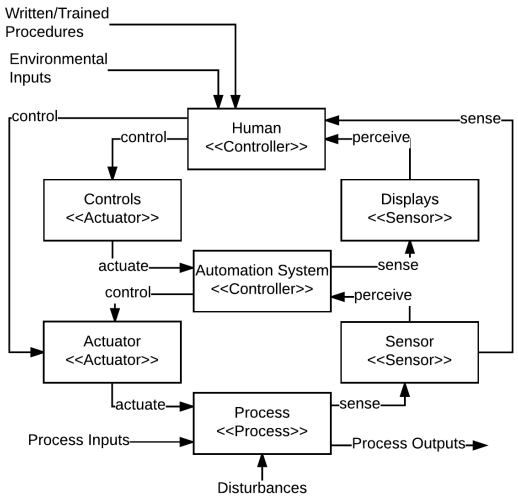

Figure 5.4: An adaptation of Leveson's model of the concurrent control of a process by a human actor who operates the process directly while at the same time also doing so through an automated controller. The model provides an implemented example of the complex control pattern.

Pattern Name: *Delegation of Responsibility*

Also Known As: *To Controller*, *2C*, *Sensor to Controller*, *S2C*, *Actuator to Controller*, *A2C*, *Controller to Controller*, *C2C*, *Process to Controller*, *P2C*, *Abstraction*, *Refinement*.

Motivation: The motivation for the *delegation of responsibility* pattern is facilitate the refinement of UTM components. This is desirable when the more abstract representation obscures the details of the SUI structure which are necessary to understand the hazards presented by the analyzed components.

Applicability: The context of application of the *delegation of responsibility* pattern is when analysts must refine components in their SUI model into more complex representations in which the components are represented as subsystems in which the process of accomplishing the goal of the initial component is achieved by the managing controller of the subsystem. A key characteristic of these situations are that a large number of hazards are being attributed to one UTM component thus suggesting a lack of understanding of the mechanisms of failure on account of the high level of abstraction used to model the SUI.

Structure: The structure of the *delegation of responsibility* pattern is modelled in Fig. 5.5 using the same modelling paradigm we applied in the description of the UTM in chapters 3 and 4. The pattern is expressed as a transformation from the initial model state to the new model state. This expression highlights the changes which are made to the model in an accessible medium. On the LHS of the transformation we represent an atomic control loop as the basis graph. After the transformation, we see in the RHS that we have extended one of the components - the *sensor* in this case - into a full control loop where the original component now plays an additional *controller* role in the new graph. Standard relationships and connections model the interactions between all of the components. This is because the new control loop is asserted to involve critical path interactions which must be executed before the initial component can complete its duties. The application of this pattern to *processes*, *sensors* and *actuators* renders models which deviate from Leveson's [76] initial description of the STAMP framework. Leveson described each of the non-*controller* components as not holding *process models* and therefore being irrational. In our synthesis of her ideas however, we allow for single components to have multiple stereotypes, thus in the transformation expressed in Fig. 5.5, we have Component D which is simultaneously stereotyped as a *controller* and as a *sensor*. This situation allows a component which is stereotyped as a *sensor* to hold a *process model* on account of its dual stereotype as a *controller* as well. Though the transformation represented applies this transformation to a *sensor*, similar transformations can be used to apply the pattern to other component types.

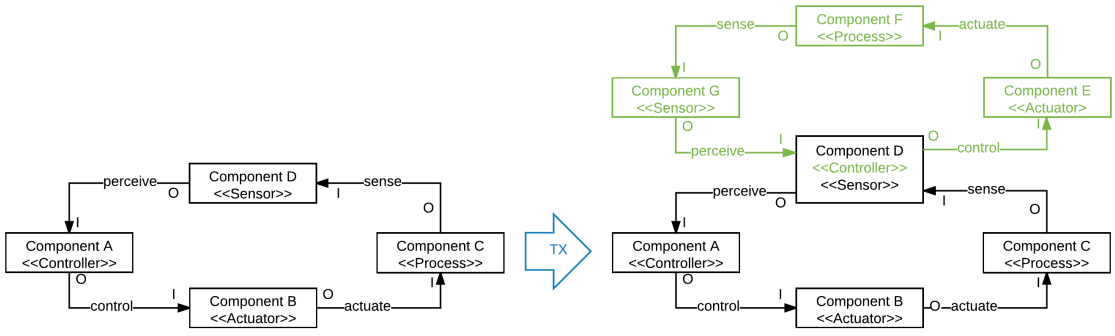

Figure 5.5: A viewpoint pair modelling the application of the *delegation of responsibility* pattern to transform a UTM component into a controlled subsystem. The green elements have been added in the RHS model from the LHS.

Consequences: The *delegation of responsibility* pattern is required when the representation of a UTM component's internal management of responsibilities must be modelled. The pattern provides analysts with a mechanism to investigate the nature of responsibility execution for the components of the SUI. It is necessary for analysts to have such facility as it is in the details of functional execution that many hazards are identified. Hazards are

often defined in abstract terms based on failures of system components, but as these descriptions tend towards the abstract, it is also necessary to be able to model the mechanisms of failure as each may require different mitigation.

Implementation: The implementation of the *delegation of control* pattern is straightforward and can be applied based on the sample *sensor to controller* application provided in Fig. 5.5 with appropriate modifications for the *controller to controller*, *actuator to controller* and *process to controller* variations.

A note on the *process to controller* application of the pattern is appropriate as it is representative of a concept labelled indirect control which Leveson discusses at length [76]. The *process to controller* transformation can be used to represent the control of a system via an intermediary. This intermediary induces a level of indirection between the controlled *process* and its *controller*. Though this is necessary when interpretation of *process* state in real time is challenging or when direct *process* perception and actuation is hazardous, the indirection can also lead to low integrity feedback and thus a compromised process model in the super-ordinate, or initial *controller*.

Though the pattern facilitates the modelling of activity using a control based paradigm, it may not be suitable for all situations. Analysts must be cautious in their application of this pattern so as to ensure that if they choose to represent a component as controlling the execution of their responsibilities that those components are in fact in control and not strictly directed by another controller in the SUI.

Practical Examples: The *delegation of responsibility* pattern is apparent in the STAMP EMR model (Section 3.5.2). The *care provider* in the STAMP EMR model delegates responsibility for the implementation of his treatment to the *core CIS software*. An example of this pattern which we did not author is provided by Leveson [76] in Chapter 9 where she discusses an experimental space shuttle robotic Thermal Tile Processing System (TTPS). This system was designed to inspect and treat the heat shield tiles on NASA's space shuttles before each launch. The architecture of the TTPS consists of a series of higher level controllers which delegate work to lower level processes, at some times partitioning responsibilities for delegation to numerous lower level controllers. This is shown in Fig. 5.6. We highlight that the *Operations Manager* delegates its efforts to two lower level *controllers*, the *Robot Work Planner* and the *Operator*. The *Robot Work Planner* subsequently delegates its work to the *Work Controller*. The *Work Controller* then partitions its duties by delegating different components of the work to each of the *Arm Controller*, the *Injection Controller*, the *Vision System Controller* and the *Movement Controller*. This example demonstrates the prevalence of this pattern in system architecture.

In the figure, we do not illustrate the application of the pattern in the same transformation context as we did in our introduction of the pattern. Our motivation in this expression is two fold. First, we wished to leverage the ambiguity in expression in order to address the use of this pattern as a mechanism for modelling *refinement* or *abstraction* - two of the patterns pseudonyms. Depending on the initial state of the model, an analyst may wish to break down an existing model component using *refinement* to model how that component completes it's task. On the other hand, it may become apparent that the component in question is in fact part of a large whole which is controlled by a higher level more abstract controller. This second situation is the subject of our next pattern. This subtlety highlights the close relationship between these patterns. Secondly, the application of this pattern can be iterative. Determining a LHS for the transformation would require that we choose a given implementation of the pattern out of the series of implementation which would be implemented in the generation of this model. This would require a long series of diagrams illustrating each subsequent pattern implementation. We instead allow the reader to infer the chain of applications based on the instruction we have provided.

Pattern Name: *Evolution*

Also Known As: *To Process*, *2P*, *Controller to Process*, *C2P*, *Sensor to Process*, *S2P*, *Actuator to Process*, *A2P*, *Process to Process*, *P2P*, *Abstraction*.

Motivation: The motivation for the *evolution* pattern is to facilitate modelling of the evolution of SUI components in such a way that their change over time can be analyzed using a concrete representation of this behaviour for evidence based assessment. Leveson spends significant effort expressing the importance of understanding the evolution of systems and their components in here discussion of the STAMP framework. She considered it to be so important that she incorporated a step in the System Theoretic Process Analysis (STPA) process to remind analysts to explicitly consider the influence of such occurrences on the safety of the SUI (Section 2.5.6).

Applicability: The context of application of the *evolution* pattern is when analysts must model the control of existing components in their SUI model over time. A key characteristic of these situations is that their is some concern over the degradation, or at least change, of performance of a component modelled in the UTM.

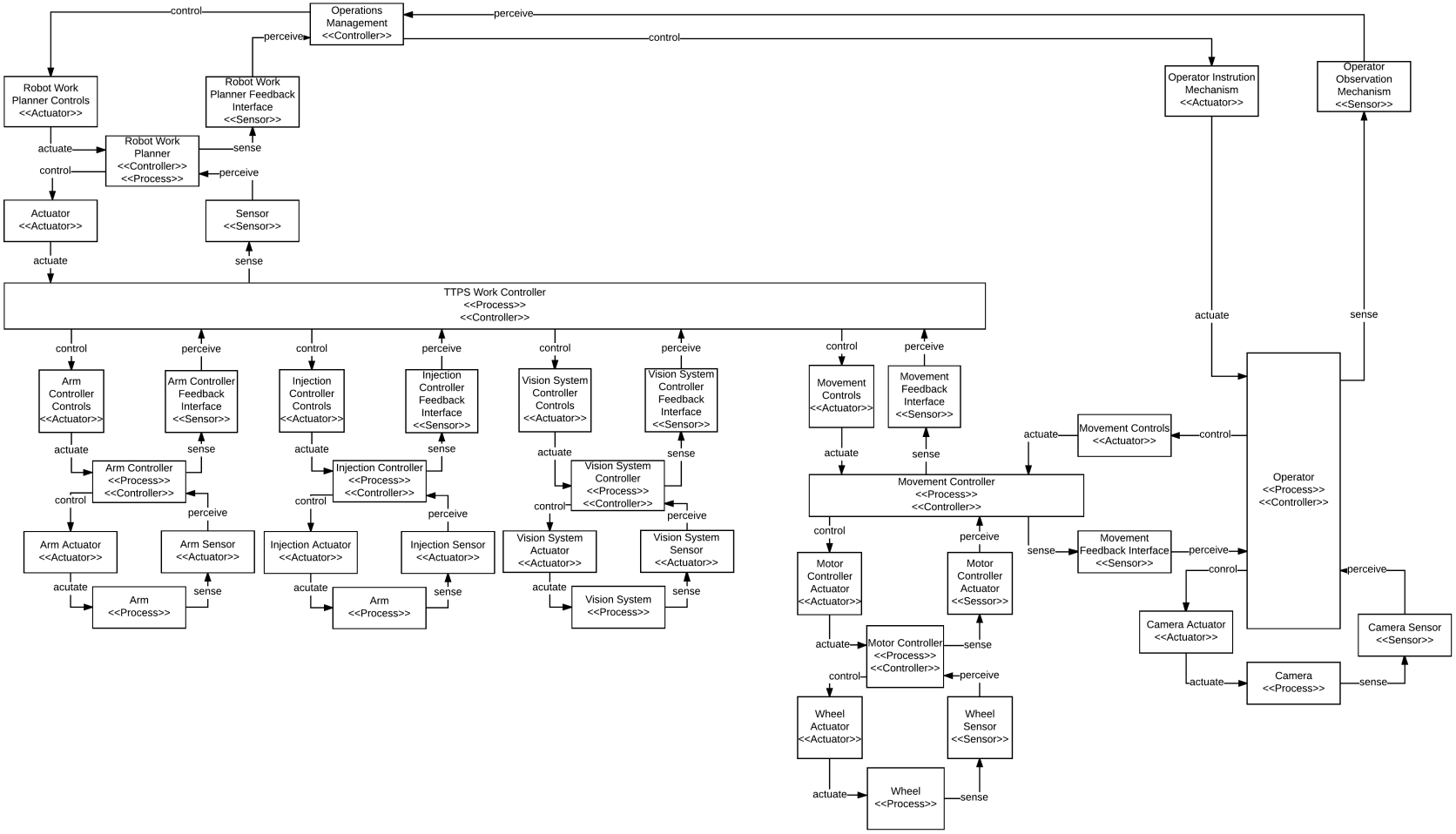

Figure 5.6: An adaptation of Leveson's model of the TTPS system [76]. The figure depicts multiple applications of the *delegation of control* pattern.

Structure: The structure of the *evolution* pattern is modelled in Fig. 5.7 using the same modelling paradigm we applied in the description of the UTM in chapters 3 and 4. The pattern is expressed as a transformation from the initial model state to the new model state. This expression highlights the changes which are made to the model in an accessible medium. On the LHS of the transformation we represent an atomic control loop as the basis graph. After the transformation, we see in the RHS that we have extended one of the components - the *sensor* in this case - into a full control loop where the original component now plays an additional *process* role in the new graph. Standard relationships and connections model the interactions between all of the components except for the component to which the *process* stereotype has been added and its incoming *actuate* edge which is initiated by the new *controller*, Component F. This is because the new control loop is asserted to involve non-critical path interactions which may or may not be executed before the initial component can complete its duties. The pattern may be varied so as to insist on the control of the central component - Component D in the figure - through the use of an *input* port rather than a *control* port in the pattern application. Though the transformation represented applies this transformation to a *sensor*, similar transformations can be used to apply the pattern to other component types as well.

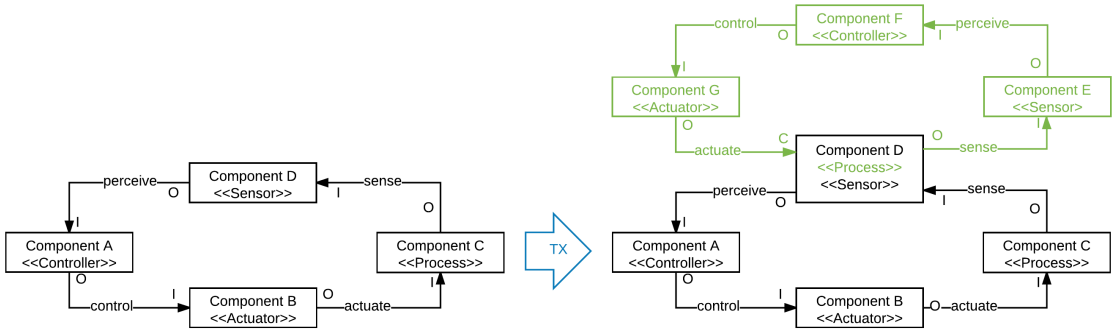

Figure 5.7: A viewpoint pair modelling the application of the *evolution* pattern to transform a UTM component into a controlled process. The green elements have been added in the RHS model from the LHS.

Consequences: The *evolution* pattern is required when the changes of a UTM component over time must be modelled. This is achieved in this pattern by modelling the control of the component in question. The pattern provides analysts with a mechanism to investigate such evolution. It is necessary for analysts to have such facility as the change of the system and of the environment over time is a known source of hazards [76].

Implementation: The implementation of the *evolution* pattern is straightforward and can be applied based on the sample *sensor to process* application provided in Fig. 5.7 with

appropriate modifications for the *actuator to process*, *controller to process* and *process to process* variations. A primary challenge with this pattern is the temptation to use it for all components of the SUI. Each component does evolve over time, but this evolution in many cases will be immaterial to the safety of the SUI in the time span in which analysts must concern themselves. Judicious application is important so as not to delve into details which will not yield the most critical system hazards.

Practical Examples: A common application of the *evolution* pattern is the implementation of "watchdogs". The watchdog concept is common in Real Time Systems (RTS) domain. At a high level the concept is that system components may become non-responsive. In the RTS space, problems arise like infinite loops, kernel panic or stalled behaviour arising from unexpected input. A common solution to this problem is to regularly signal the monitored component to ensure that it remains responsive. If the component does not respond then a restart signal is sent to recover the frozen component. Such a mitigation is easily represented by an *evolution* pattern with the monitor being a *controller* which periodically *actuates* the monitored component and *perceives* its response. If no response is received the *controller actuates* a reboot of the monitored component. An example of the evolution pattern is also provided by Leveson in Appendix C [76]. Leveson describes here, the *E. coli* contamination of the town of Walkerton Ontario's water supply in May 2000. Leveson depicts the control system in place for the town's water management in detail. We provide a simplification of this model in Fig. 5.8 in order to illustrate her application of the evolution pattern. Leveson indicated in her model that the water testing lab was managed by the *Ministry of the Environment*. The *Ministry of Environment* was in turn governed by the *Provincial Government*. The application of the pattern can be understood by observing that the *Provincial Government* was added as a *controller* of the *Ministry of the Environment* to govern the Ministry's behaviour over time.

Pattern Name: *Peripheral*

Also Known As: *Observation*, *To Sensor*, *2S*, *Activation*, *To Actuator*, *2A*

Motivation: The motivation for the *peripheral* pattern is facilitate the modelling of the observation and activation activities of SUI components as internal activities worthy of investigation. With any component of the SUI, their activities will initially be modelled in the abstract and will subsequently be refined as analysts curiosity or suspicion is raised by the Retrospective Incident Data (RID) and other evidence they consume. The *peripheral*

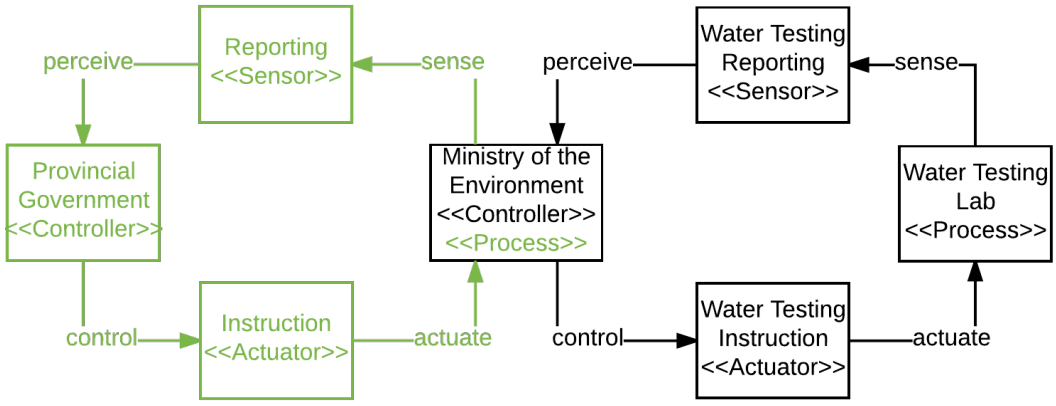

Figure 5.8: An adaptation of Leveson's [76] modelling of the control system in place for the town of Walkerton's water management. The model demonstrates and application of the *evolution* pattern.

pattern facilitates the inspection of the non-control/non-evolution activities undertaken by components in the execution of their duties.

Applicability: The context of application of the *peripheral* pattern is when analysts must model the observation and activation activities of SUI components to discover the hazards they pose. A key characteristic of these situations is that their is some concern over the faithfulness of observation, or competency of execution of duties for a given SUI component.

Structure: The structure of the *peripheral* pattern is modelled in Fig. 5.9 using the same modelling paradigm we applied in the description of the UTM in chapters 3 and 4. The pattern is expressed as a transformation from the initial model state to the new model state. This expression highlights the changes which are made to the model in an accessible medium. On the LHS of the transformation we represent an atomic control loop as the basis graph. After the transformation, we see in the RHS that we have extended one of the components - the *actuator* or Component B in this case - into a full control loop where the original component now plays an additional *sensor* role in the new graph. Standard relationships and connections model the interactions between all of the components. This is because the new control loop is asserted to involve critical path interactions which must be executed before the initial component can complete its duty. The pattern may be varied so as to loosen the constraint on the central component in the pattern application - Component B in the figure - so as to model non-critical path behaviour. Though the

provided example applies this transformation to an *actuator*, and adds a *sensor* role to the component modelling the observations it makes, similar transformations can be used to apply the pattern to other component types as well. These transformations can be applied to add not only *sensor* stereotypes as is done here, but also *actuator* stereotypes.

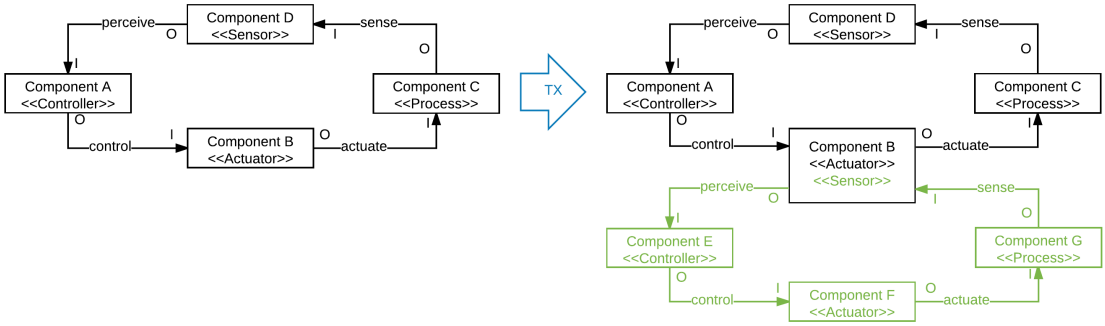

Figure 5.9: A viewpoint pair modelling the application of the *peripheral* pattern to model the observation behaviour of an *actuator* in the execution of its duty. The green elements have been added in the RHS model from the LHS.

Consequences: The *peripheral* pattern is required when the component observation and activation actions of a UTM must be modelled. This is achieved by this pattern by adding *actuator* or *sensor* stereotypes to existing components and building out a control loop in which those component play those roles to fulfill their duty which is modelled as a process which they *sense* or *actuate*.

Implementation: The implementation of the *peripheral* pattern is straightforward and can be applied based on the sample *actuator to sensor* application provided in Fig. 5.7 with appropriate modifications for the *actuator to actuator*, *controller to actuator*, *controller to sensor*, *process to actuator*, *process to sensor*, *sensor to sensor* and *sensor to sensor* variations. None of these variations pose complexity of sufficient depth to motivate further discussion.

Practical Examples: An example of a situation where this pattern could be used is in a secondary care setting in which different systems manage the basic demographic and charting information for patients including treatment plans while a different system handles the ordering and reporting of medical imaging data. Another example would be a remote control car. These toys often have a speed actuator which is used to actuate the rotation of the toys wheels, and also a steering actuator which actuates a servo motor to change the direction the wheels of the toy face allowing the operator to guide trajectory. We model this in Fig. 5.10.

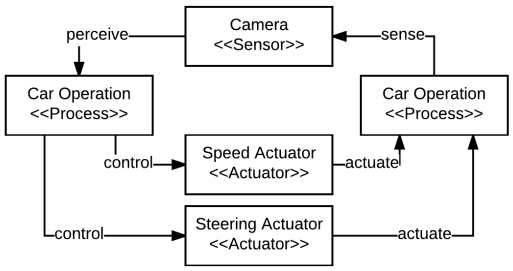

Figure 5.10: A model of a remote control car's control in which we illustrate the *peripheral* pattern by highlighting the used of the multiple actuators used in the control of the toy.

Pattern Name: *Inversion of Control*

Also Known As: *Information Hiding*

Motivation: The motivation for the *inversion of control* pattern is facilitate the investigation of complex control systems by critically considering which components are in fact in control of the progression of the SUI. In complex systems with multiple levels of control indirection, it can be difficult to determine which components of the system are in fact driving the trajectory of the SUI as a whole. In such systems there are also multiple processes which have varying levels of indirect influence over each other. Making sense of such systems is difficult and the *inversion of control* pattern can be applied to assist in this endeavour.

Applicability: The *inversion of control* pattern is most applicable when it is unclear to analysts how best to identify hazards in a system on account of a lack of clarity about "which way" the SUI runs. Does it go from a *controller* on one side of the UTM to a *process* on the other side, or is the *process* in fact a *controller* which is determining the behaviour of the component which is purportedly a *controller*, but is in fact a *process*. This type of situation arises most commonly in what Leveson would describe as indirect control systems where there are levels of indirection between the components on each of the edges of the SUI.

Structure: The structure of the *inversion of control* pattern is modelled in Fig. 5.11 using the same modelling paradigm we applied in the description of the UTM in chapters 3

and 4. The pattern is expressed as a transformation from the initial model state to the new model state. This expression highlights the changes which are made to the model in an accessible medium. On the LHS of the transformation we represent an atomic control loop as the basis graph. After the transformation, we see in the two RHSs that we have filtered inter-component communications and stereotypes. Initially we model full bidirectional control in the LHS. Through the transformation however, we constrain what of the model is visible filtering the perspective to show only one direction of control at a time. We use the light grey color instead of the red used in previous transformation diagrams to differentiate the filtering operation used here which constrains what is visible in the view of the model versus the removal of items from the model which is indicated by the semantics of the red color in the other diagrams. Standard relationships and connections model the interactions between all of the components. The connections used in this pattern are largely immaterial as the most critical aspect of the pattern is the direction of control.

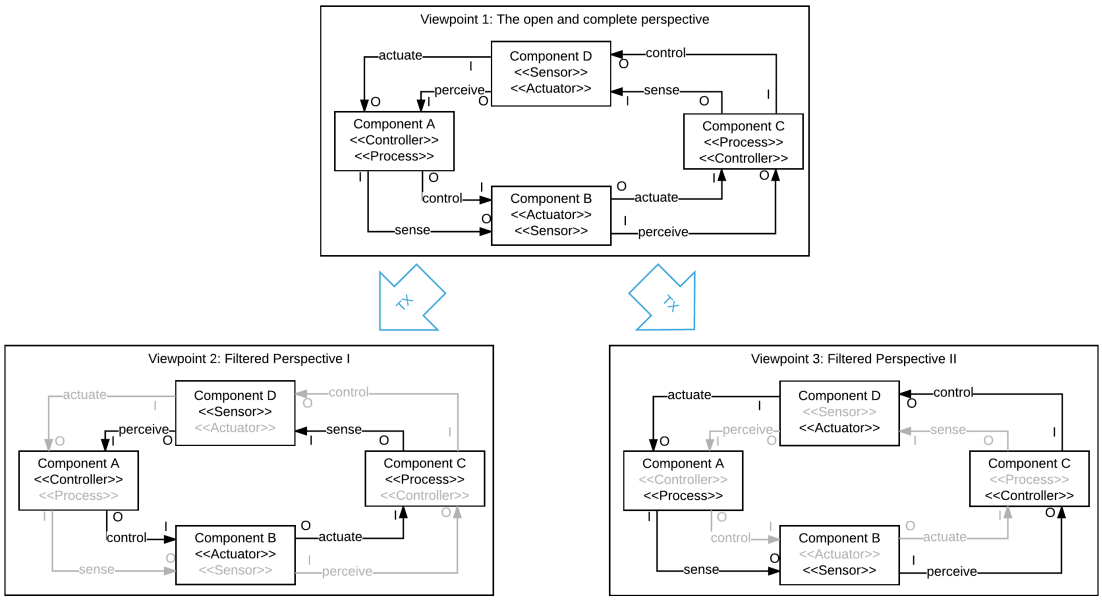

Figure 5.11: The figure models the *inversion of control* pattern with a transformation. In the LHS we begin with a viewpoint modelling full bidirectional control between the components of the SUI. In the two RHSs we use a light grey to indicate which edges and which stereotypes are hidden thus showing only one direction of control in each view.

Consequences: The *inversion of control* pattern is required when analysts need to simultaneously consider the possibility of control in both directions for systems which exhibit bidirectional control behaviour, or where such behaviour is expected or simply a possibility which warrants investigation. This is achieved with this pattern through the application

of complementary stereotyping of all components in a control loop. By complementary, we mean to associate the "primary" - *controller* and *process* - versus "peripheral" - *sensor* and *actuator* stereotypes with their respective pairs. A component that is initially stereotyped as a *controller* then will subsequently be stereotyped both as a *controller* and also as a *process*. Similarly, an *actuator* would subsequently be stereotyped as both an *actuator* and as a *sensor* and so on. By doubly stereotyping control loop components in this fashion, analysts can then develop viewpoints that focus their attention on control in the direction they choose for a specific portion of the investigation.

Implementation: The implementation of the *inversion of control* pattern is straightforward and can be applied based on the sample application provided in Fig. 5.7. Unlike with many of the other patterns discussed here where judicious application is key to maintaining model simplicity, the *inversion of control* pattern is often unexpectedly enlightening. Its application can yield insights about the SUI's operation which are not foreseen by analysts and thus result in a greater appreciation for the complexity o the system they are investigating and the nature of trust and control in it.

Practical Examples: To demonstrate the value of this pattern, let us return out attention to the STAMP EMR model (Section 3.5.2). In the STAMP EMR model, it would initially appear that the *care provider* employs the *core CIS software* in the care of his patient. The *care provider* senses and perceives the patients healthcare process and then determines a course of action based on the available data. The *care provider* then chooses an appropriate *actuation* command to achieve the desired change in the patients healthcare process and applies this command via the *CIS input interface*.

Consider this inversion of perspective however - the patient reports to the doctor that he is feeling blue and would like the doctor to consider what treatments might be available based on a series of experiences the patient has had with a combination of medication and counselling options. The doctor responds to this actuation with a prescription which is sensed by the CIS input device and subsequently perceived by the patient who is able to consume the resulting prescription. This inversion of perspective is the purpose of this pattern.

Pattern Name: *Transition of Responsibility*

Motivation: The *transition of responsibility* pattern supports the modelling of changing duty ownership in an SUI. Ash [7] and others have highlighted the importance of this system

behaviour in the Health Information Technology (HIT) literature as a significant source of hazards. This issue presents itself clearly in situations where work processes are changing - a situation which is often induced by the introduction of technology into an SUI. It is used to support change management.

Applicability: The *transition of responsibility* pattern is applied when duty ownership transitions between components of the SUI.

Structure: The structure of the *transition of responsibility* pattern is modelled in Fig. 5.12 using the same modelling paradigm we applied in the description of the UTM in chapters 3 and 4. The pattern is expressed as a transformation from the initial model state to the new model state. This expression highlights the changes which are made to the model in an accessible medium. On the LHS (on top) of the transformation we represent a complex control system in which we include an additional control duty between the Component A *controller* and the Component D *actuator* as the basis graph. After the transformation, we see in the RHS (bottom) that we have removed the initial "control 2" duty between Component A and Component D, and have added the same duty twice - once between the Component F *controller* and the Component G *actuator* and once between the Component F *controller* and the component D *actuator*.

Consequences: The *transition of responsibility* pattern is used when analysts need to investigate the consequence of duty reassignment. This is achieved by this pattern by removing the initial duty from the UTM and adding it again but between new components.

Implementation: The implementation of the *transition of responsibility* pattern is challenging because of its purpose. The reason analysts will use the pattern is to investigate alternative duty assignments in order to understand the consequence of SUI design decisions. A primary challenge with this pattern is that the UTM provides a singular definitive model of the SUI. It does not support the concurrent consideration of disparate designs. The consequence of this shortcoming is that analysts can only use this pattern by forking the UTM in order to represent the multiple possible duty distributions and then comparing and contrasting these alternatives.

Practical Examples: A relatively commonly described example of the behaviour which is modelled with the *transition of responsibility* pattern is found in the deployment of Computerized Provider Order Entry (CPOE) technology in secondary care settings. Depending on the nature of authority relationships and personalities of the doctors and nurses involved,

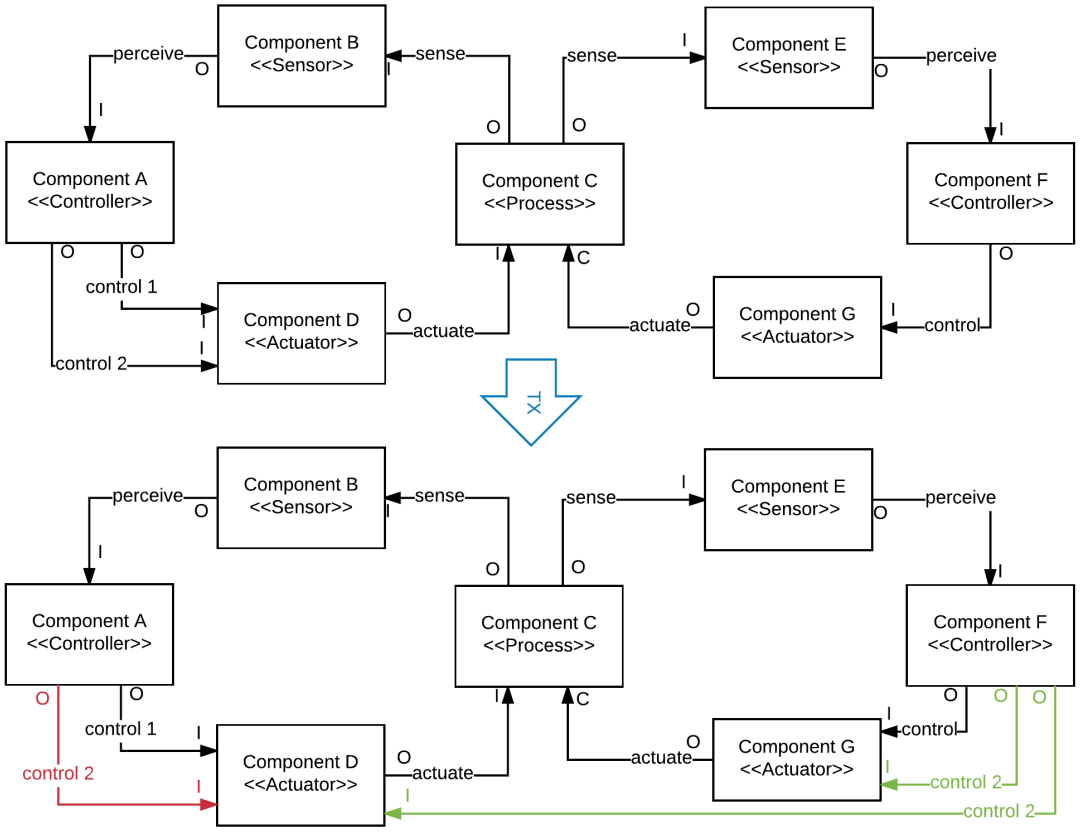

Figure 5.12: The figure models the *transition of responsibility* pattern with a transformation. In the LHS (on top) we begin with a viewpoint modelling the *complex control* of a process. In the RHS (on bottom) we use red to indicate the removal of the "control 2" edge between component A and component D. We use green to denote the addition of the same duty twice - once between component F and component G and then again between component F and component D. In so doing, the "control 2" duty is transitioned from components A and D to components F, G and D.

the duty of prescription authoring can transition from doctors to nurses. The duty of prescribing itself may remain with the doctor, but it has been reported a number of times that in some instances doctors who become frustrated with the data entry necessary to provide a prescription through the CPOE. This can seem a time consuming process and the doctors feel that their time is better spent on other care activities with the prescription entry being a formality. As a consequence they discharge the duty of prescription entry to nursing or Medical Office Assistant (MOA) staff thus demonstrating a transition of responsibility from the state of the SUI prior to the deployment of the technology.

5.4 Summary

In this chapter we have introduced a formal information model for the ISHA method which includes metamodels for hazards, systems, and assurance cases. We have described the components of this model and the models purpose. Further, we have introduced a series of design patterns which can be used in analysis that specifically demonstrate the application of the system metamodel. In Chapter 6 we will provide an evaluation of the ISHA method.

Chapter 6

Evaluation

6.1 Validation of Method Requirements

Prior to Information System Hazard Analysis (ISHA), a gap existed in the published extant hazard analysis techniques. Traditional component focused reliability analysis methods including Fault Tree Analysis (FTA), Event Tree Analysis (ETA), and Failure Mode and Effects Analysis (FMEA) had been criticized [76, 52] for their failure to consider the impact of component interactions. At the same time however, the most popular [136] systemic methods [76, 52, 108] did not provide the same formal modelling rigor as some of the earlier methods including FTA and ETA. While FTA and ETA used formal logical relationships between events that supported the computation of Baysian probabilities, the systemic methods used relatively informal object relations to link system components.

Further, few techniques focused specifically on healthcare, and none which we identified focused on Clinical Information Systems (CIS). There was also a more general need for a hazard analysis method which focused on information systems. This was necessary as, unlike traditional mechanical systems, the emergent safety of information systems is highly dependent on a triplet of non-functional requirements: usability, integrity and availability.

To address this gap, we devised requirements for a new method which we called ISHA and subsequently constructed the method. The requirements are based on a synthesis of strengths of the extant traditional and systemic methodologies, criticism levelled by the authors of the state-of-the-art systemic methods [76, 52] against the traditional methods, and the evidence based positions of leaders in the field of Health Information Technology (HIT) safety including Ash [6, 7], Koppel [72, 71], Sittig [126, 125, 124, 127] and others. These requirements are listed below:

- The method must have a systemic basis

  - The method must support control loop based analysis of the System Under Investigation (SUI)
  - The method must support resonance based analysis of the SUI
- The method must be grounded in a validated systems safety process
  - The method must provide guidance on execution logistics
  - The method must prescribe validated contextual activities
  - The method must support formal modelling of the SUI
  - The method must provide guidance on packaging activities for the delivery of the results of the method execution
- The method must be specialized for CIS
  - The method must be specialized for the analysis of safety concerns which are of central concern in information systems
  - The method must focus on the identification of hazards arising from both technical and human sources - both latent and acute
  - The method must address the transition of responsibilities between roles in the SUI

6.1.1 Systemic Basis

The need for a systemic basis for a defensible hazard analysis technique arises from a series of flawed assumptions which ground traditional techniques like FMEA, Event Chain Analysis (ECA) and FTA. These assumptions include the assertions that [76, 52]:

1. systems can be meaningfully decomposed into their constituent parts for analysis
2. safety and reliability are closely related system properties
3. accidents are predictable based on system design and knowledge of the environment of operation
4. the modelling of complex adaptive systems can provide perfect knowledge of those systems
5. analysis provides a clear understanding of the precipitating causes of accidents
6. operator error is at the root of most accidents

7. accidents arise from the occurrence of simultaneous random events

Experience has demonstrated that the following statements are more accurate [76, 52, 20]:

1. accidents can arise from unexpected interactions of reliable components within a system
2. reliable systems can be unsafe while safe systems can be unreliable
3. complex dynamic socio-technical systems display unexpected emergent behaviour in the context of the unknowable evolution of their environments of operation
4. perfect representation of complex systems is impossible; rather, hazard analysis modelling must focus on providing abstractions which are useful in detecting factors which may contribute to the realization of hazards into accidents
5. hind sight bias is common in hazard analysis, and frequently provides illusory comprehension
6. operators do their best to optimize system performance while maintaining safety, but are strongly influenced by their work environment
7. in the absence of accidents systems migrate towards states of higher productivity with diminishing focus on safety

Reliability, Safety and Emergence

The emergence of a hazard from component interactions is demonstrated in the Mars Polar Lander incident where a craft destined for the red planet crashed due to interactions between components. The landing thruster for the craft was programmed to turn off once the craft reached a threshold altitude. When the craft approached the planet, it reliably deployed its landing gear. This operation generated noise in the craft's circuits resulting in an erroneous altitude reading when the craft was still kilometers from the planets surface. Upon recognizing the altitude reading, the thruster module switched the thruster off leaving the craft to plummet to the planet's surface and resulting in the loss of the mission. Reliability of components is insufficient to guarantee system safety. On the other hand, an airplane that does not fly is safe in that it will not crash, but it is also unreliable - it does not perform its intended duty. Taking a systems based approach forces analysts to consider safety as an emergent property of systems and thus steers them away from the fallacy of the similarity of safety and reliability.

Knowability of the Future

The evolution of the deployment environment of a system is inevitable. Further, system designers cannot perfectly predict this evolution. For this reason, prospective analysis and design is not an activity which is performed once at the inception of a system, but rather is a process which must continue as both the designed system and its environment(s) of operation evolve. Continuous monitoring of a system is essential to safe operation. This perspective of systems being dynamic and evolving is central to the systemic perspective. We will revisit this theme in our coming subsection on the migration of systems to states of higher productivity.

Modelling is Abstraction

Understanding complex systems is difficult. Modelling is a process which has been used to abstract complex systems into simplified renditions that support reasoned analysis of approximate probable outcomes. Modelling is used in all of the hazard analysis techniques we have discussed in depth - FTA, ETA, FMEA, HAZard OPerability (HAZOP), System Theoretic Accidents Models and Processes (STAMP), Functional Resonance Analysis Method (FRAM) as well as other hazard analysis methodologies. Using a model of a system to ground safety analysis supports analysts in arriving at evidence based recommendations. The process provides insight to the analysts into the potential emergent behaviour of the systems they inspect. This is why we prescribe modelling in the ISHA process.

Hind Sight Bias and Illusory Comprehension

Retrospective analysis is plagued by the bias experienced by analysts who review an event after it has occurred. Retrospective analysis, when not considered with substantial suspicion, can provide the illusion of accident comprehension. Analysts will [76]

- Oversimplify the causality model for accidents
- Overestimate the likelihood of the outcomes which occurred
- Overemphasize the role of procedure violations
- Misjudge the significance of the data perceived by operators at the time of an accident
- Color the value/correctness of actions leading to an accident with the outcome even though the outcome was uncertain

This position grounds the systemic approach. The complex feedback loops from which socio-technical systems are constructed can evolve rapidly and unexpectedly. Taking a

careful and considered approach to analysis which is also humble and expectant of the coloring of bias is a guiding principle of systemic analysis.

Operator Error

Operators are often blamed for accidents when analysis is performed. Something about human nature leads to a sense of dissatisfaction when an inanimate machine is blamed for a loss, whereas vengeance against a human operator can seem more fulfilling. The move to systems based thinking [12] has driven safety analysts to consider no fault approaches that support reporting of near misses. The benefit expected is that organizations which take such an approach can learn ideally of hazards, or at least from accidents so that the hazards can be mitigated before they realize into initial or subsequent accidents.

Operators do their best to optimize the productivity of the systems they control while managing safety to the best of their ability with the resources they have available; however, as will be discussed in the next section, safety is often the secondary concern of the operator - especially in the absence of recent accidents.

Migration Towards States of Productivity

"Accidents and the threat of accidents are the primary motivators for [safety work]. Take accidents away and concern about safety diminishes and attention shifts towards production. Virtually all safety work takes place in the shadow of accidents and experience with accidents both our direct experience and that which we acquire by hearing about the accidents that happen to others shapes our general and specific approaches to safety." [20] This mental bias which leads us to forget the risks of our activities also leads to narrowing safety margins. As time goes on, the perceived probability of an accident decreases and thus operators tend to make riskier decisions for the benefit of the productivity of the system they work in. These decisions - not running a regular maintenance schedule to save a few dollars in wages, running a boiler at a marginally increased pressure to improve the output of a chemical process, flying a plane with a little less fuel so as to be able to take on additional passengers - are the decisions that diminish margins of safety in a system, and occasionally result in an accident. If safety margins can be identified and monitored, then greater control can be exercised over the system reducing the risks it poses.

6.1.2 Validated Systems Safety Process Basis

Many of the extant methods provide only a sliver of the overall process necessary to perform hazard analysis. These methods leave out significant components of the more robust pro-

cesses described by Ericson [28], and the US Department of Defense (DOD) [26]. The rigor provided by these established methods promotes a degree of quality in analysis outcomes which may not be achieved in the absence of such a clear process. The rigor we refer to includes the specification of contextual activities including team selection, packaging analysis findings and auditing analysis reports.

Logistics Activities

By providing logistics guidance, a method can support a study initiator in understanding the journey on which they are embarking. Though setting up meeting times, securing venue an resources are common sense activities, team construction is more challenging.

Contextual Activities

Contextual activities including the definition of the concept of operations, the Preliminary Hazard List (PHL), the Preliminary Hazard Analysis (PHA), risk assessment are missing from many of the extant methods including FTA, ETA, FMEA, HAZOP, STAMP, and FRAM. By including these details, a degree of rigor and quality can be promoted in an investigation method. The commonly cited United States Department of Defense Standard Practice System Safety (MIL-STD-882E) process demands that these tasks be completed as does the IEEE 1228 Standard for Software Safety Plans (IEEE1228); similarly, ANSI STD-0010-2009 Standard Best Practices for System Safety Program Development and Execution (ANSI-STD-0010-2009) insists on similar activities. In the absence of these contextual tasks, important analysis outputs are missing. It is therefore important to base analysis methods on the structures in place already.

Prioritization

Prioritization of hazards is the topic of some debate [21, 120]. It is also necessary in order to keep analysis within budget. Safety management is a journey and not a destination. It is a process which is never complete and so it must be bounded by a budget of time and resources. Prioritization techniques which are commonly prescribed like the Risk Probability Number (RPN) strategy for MIL-STD-882E and FMEA are only part of the solution. These methods depend on more foreknowledge of hazards than is available at the PHL phase of an investigation. Techniques beyond those grounded in quantitative risk knowledge are also necessary to make coarse scoping decisions early in the execution of an analysis. ISHA takes the stance that system risk should be reduced such that it is As Low As Reasonably Practicable (ALARP). This means that risks should be reduced based on their significance.

Risks which pose a great threat will require better mitigation. At the same time, mitigation can often transfer risk between elements of a system or from elements in a system to elements in the system's environment. As ISHA uses a closed system modelling approach, this would result in the inclusion of the new elements into the system model to assess the impact of the transferred risk. ISHA does not prescribe a method of prioritizing risk; it only prescribes that a process be chosen and used. This will be further discussed in Section 7.7.

Modelling

Hazard analysis is about details. Without a concrete model - visual, textual, or otherwise - it is difficult to maintain the focus necessary to perform and evidence based evaluation of the safety of an SUI. Many of both the traditional and systemic hazard analysis methods methods prescribe the use of models to ground analysis and to support system coverage. FTA [28, 131], ETA [28, 130], FMEA [130], STAMP [76], FRAM [52], and Accimap [108] are just a few of the methods which rely heavily on modelling to understand the nature of failure and accidents. The guidance provided in the development of traditional models is clear. With FTA and ETA, events are linked together by logic gates thus allowing for the computation of Baysian probabilities. With the FRAM, STAMP and Accimap methods, the model generation techniques are less prescriptive and leave more semantic ambiguity. Neither the traditional, nor the systemic techniques have clear support for abstraction; rather, analysts must infer these mechanisms in each of these methods.

Qualification of Risk

Qualification of risk whether quantitative or qualitative is necessary in hazard analysis. The goal of a clinical information system safety program is not to ensure absolute safety, but rather to achieve the optimally desirable balance between safety and productivity. All patients are in a process of declining health - this is simply a consequence of aging and disease progression. Optimizing a system for absolute safety is also not possible as decisions which make the system safe for one patient can make the system unsafe for another. If the system is not productive then the treatment of all patients ceases and the system provides no benefit. Accidents are acceptable so long as the benefit provided by the system outweighs their consequences. Achieving this goal necessitates that analysts be able to assess residual risk - the risk that a hazard will realize into an accident of a given severity in spite of the mitigations which have been put in place to prevent this. If analyst cannot make this judgment, then they could not know when to stop designing and implementing new mitigations. Qualification of risk is an activity which is closely related to prioritization as it is a common driving factor for the latter. The term Safety Integrity Level (SIL) is used

to describe "the probability (likelihood) of a safety-related system performing the required safety function under all the stated conditions within a stated period of time" [57]. SILs are used to classify systems based on confidence in their safety. Various standards address SILs, and the application of a choice of SIL standard may provide guidance to analysts on how to approach the guarantees they need to provide about their system and thus when analysis and risk mitigation can be halted for a given period.

Packaging Activities

The principle techniques we have addressed in our work provide little guidance in how to package the outputs of analysis. FMEA is one of the strongest of the techniques on this front with substantial guidance on how to format the results provided by Stamatis [130]. Even this guidance however, does not address the fundamental purpose of performing the analysis in the first place - to produce a compelling argument of the safety of the system relative to its benefits. It is necessary for an effective hazard analysis technique to provide a rigorously defined approach to this problem in order to satisfy the goal of producing this compelling safety case.

6.1.3 Clinical Information System Specialization

Information System Needs

In information systems, usability, integrity and availability play central roles in the emergence of safety [81, 82, 29, 78]. Each of these system attributes are recurrent topics in HIT which have garnered specific attention due to the many accidents in which weaknesses in these attributes were blamed. We provide below motivating evidence for our focus on these three system attributes.

Usability The usability of CIS systems is often blamed for compromising patient safety. This was uncovered throughout the literature [81, 82, 91] as well as in an analysis of a large bank of voluntary incident reports [29]. Issues range from poor clarity of order cancellation, information overload, fragmentation of clinically relevant data, frequent irrelevant Clinical Decision Support (CDS) alerts, and a broad range of other issues. Issues with usability can be particularly pernicious as users may not realize the errors they make, for example slips where they fail to consume critical clinical information in a display, or slips where they believe they have submitted critical information like and order, but in fact have not.

Integrity The integrity of the data captured by CIS systems is also often blamed for compromising patient safety. This, again was uncovered throughout the literature [81, 82, 85, 39] and again in an analysis of a large bank of voluntary incident reports [29]. Latent data errors [11], including problems with data integrity and clinical content, have a high potential for patient harm as they are concealed in a wealth of correct information within CIS technology [127]. The consequence of integrity issues can range from orders being transmitted to the wrong provider, treatments being administered to incorrect patients, wrong site surgery, 10 fold over or under medication, and a broad range of additional issues. Issues with integrity are particularly pernicious as a user who trusts the information presented to him/her will not question that data when it is erroneous but plausible.

Availability The availability of critical infrastructure is obviously essential to safety. Dams are protected because of our dependence on power. Water purification plants are protected because of our vulnerability to waterborne pathogens. CIS are also critical infrastructure as was demonstrated by the collapse of the National Health Service (NHS) and Veterans Affairs (VA) systems [59]. This theme was also prevalent in an analysis of the same large bank of incident reports cited above [29]. Issues with availability are particularly pernicious because system users are helpless until the system is recovered by their technical support team. They are forced to rely on paper backup systems. This paper record keeping is difficult to enter into the electronic record after the fact because health care institutions are so often taxed for resources. Consequently, outage events result in data loss - sometimes to the detriment of patient safety.

Sociotechnical Analysis

Some latent errors in CIS relate to the misalignment of organizational functions - for example, not having a position assigned the responsibility of managing the emergency back up key to a medicine cabinet [33]. Latent errors can also be more purely technical - for example, having backup servers for systems running on different microcode thus impeding recovery from system failures [60]. Insufficient organizational readiness can also pose threats to patient safety - for example not testing system failure recovery procedures [60], or failing to understand the impact workflow changes on medication delivery [41]. The need for a means to unearth these hidden problems in complex sociotechnical healthcare systems is a primary motivation of hazard analysts' investigation socio-technical systems.

Identification of the Role of Human Error Though the systemic approach to safety analysis discourages blame on human operators, human error is none the less one of many hazards in accidents [118, 110, 54]. Our intent in identifying human error with

the ISHA method is not to find blame, but to uncover threads that can lead to greater understanding of the context which can put the SUI at risk. The nature of any given error can arise from a range of potential sources including usability issues [73], ineffective changeover communications [54], workarounds for inappropriate technologies [72], simple slips [86], or challenges with distribution of cognition across the sociotechnical platform of a CIS [55, 102].

Identification of the Role of Technical Error Technical error can arise from many sources in a CIS. We have discussed usability, integrity, and availability, but other more basic sources of error exist as well. We distinguish here between data integrity which was our focus in the initial information system discussion, and software correctness. Validation and verification concerns with software were found to be one of the most common hazards to patient safety incidents [29]. This discovery warrants careful consideration of how to approach technical error. It is clear that hazard analysis techniques for information systems, and therefore for CIS, must focus on the identification of hazards related to technical error.

Transition of Responsibilities The shifting of responsibilities across both electronic and human actors in the care team is a known cause of friction and of hazards in CIS [43, 6, 7, 41]. In one case, introduction of a new Computerized Provider Order Entry (CPOE) into a children's hospital lead to changes in workflow which reduced nurse-physician interactions in the care of critically ill patients. In other situations, cumbersome data entry tasks assigned to overburdened nurses have lead to workarounds in which nurses perform batch data entry at the end of their shifts. This unsanctioned modification of the workflow compromises the accuracy of the medication administration record sometimes leading to incorrect dosing, or prescription duplication [43]. Further, sometimes it is either hubris or poor consideration of the economic impact of induced transitions of functional responsibility which cause problems [6] - if the recipient of a duty refuses to take on the assigned tasks then this in itself can pose a threat to patient safety.

6.2 Verification of Method Requirements

The requirements which must be verified are that:

1. the method must have a *systemic basis*
2. the method must be grounded in a *validated systems safety process*
3. the method must have a *CIS specialization*

We will argue that each of these requirements are satisfied by the ISHA process.

6.2.1 Systemic Basis

There are a variety of aspects of systems thinking that have been incorporated into the ISHA process. We address a few here to demonstrate how ISHA has a rich grounding in this paradigm. In particular we address ISHA's treatment of emergent system properties. These properties are not properties of system components but rather are properties which emerge from the aggregate of system components and interactions. We specifically discuss the concepts of safety and reliability and how ISHA incorporates assessment of these properties into analysis. We address how complex systems evolve over time and that the trajectory of their behaviour is unpredictable. We discuss the use of abstraction in the ISHA modelling process to address the challenges of understanding the complex systems which the method is used to analyze. We demonstrate how ISHA mitigates hindsight bias which has plagued accident analysis and has arguably impeded the improvement of the safety of systems which have suffered from the inaccurate analysis which results. Next we discuss how ISHA mitigates the tendency to blame operators. This is a common bias of extant system analyses. Finally, we discuss ISHA's treatment of the migration of systems towards states of productivity. This behaviour is extensively discussed in the systemic thinking literature.

Reliability and Safety as an Emergent Property

In ISHA, analysts consider reliability and its influence on safety outcomes in the PHA and Component Fault Analysis (CFA) method phases. The reliability of components which are critical to the operation of a safety system - one whose purpose is to maintain safety - is likely necessary for the safety of the system whose safety they contribute to. Analysts assess this impact in both the PHA and CFA phases of ISHA. It is not assumed that reliability is necessary for safety; rather, the impact of failure is considered in the light of outcomes to determine the relevance of the reliability of components on safety. Further in ISHA, the impact of interactions is also a central focus of analysis. ISHA prescribes a process that insists that analysts investigate aspects of the system, including component interactions, that go beyond component failure in assessing the safety of the SUI. The combination of consideration of the impact of component reliability on system safety, and the prescription to investigate further demonstrates ISHA's focus on the emergent nature of safety and the fundamental difference between reliability and safety.

Knowability of the Future

ISHA is not intended to be a one time analysis. It is expected that practitioners will repetitively analyze the SUI over the course of time. Within a given analysis, the steps of the ISHA process are performed iteratively in order to achieve a sufficing level of safety given the nature of the system and its functions. This may be governed by regulation, or even by market forces. The method promotes the development of monitoring tooling to identify changing processes and the shrinking safety margins which indicate the potential for escalating risk. One example of this promotion is the recommendation that analysts rely on Retrospective Incident Data (RID) to identify hazards and to assess risk. Though the future is not completely knowable, deviations of socio-technical processes are rarely both instantaneous and unforeseen. This kind of tooling can allow system operators to address such deviations in system performance before they materialize into accidents. ISHA promotes the design of mitigations which provide for a margin of error in system execution, and also continuous improvement. The continuous improvement aspect of issue is highlighted by both the promotion of system monitoring tooling and by the final phase step of the process which indicates that the complete analysis process should be periodically done again. This time series repetition of the analysis will support the identification of changes in the operation of the system (e.g., workflow deviations including workarounds) which were not foreseen earlier in the evolution of the system.

We describe ISHA as a prospective hazard analysis method. By this, we mean that it is used to analyze systems in order to prevent accidents from happening rather than as an accident analysis method. This is different from a method which is intended for use before implementation. One key aspect of this difference in that ISHA does not distinguish between an initial analysis and what is done during system maintenance. It is expected that ISHA will in fact usually be performed on an SUI which is already in operation and thus is currently being maintained, rather than before the system is investigated.

Modelling is Abstraction

ISHA's synthesis of the STAMP stereotypes, the FRAM functional model, and the prescription of the use of a combination of static and dynamic models demonstrates its focus on modelling. To support abstraction, ISHA leverages viewpoints based on STAMP stereotypes and the control loop structures which are created in the modelling phase of the process.

Hind Sight Bias and Illusory Comprehension

Like all methods, ISHA is subject to hind sight bias and the illusion of comprehension; however, the method's focus on continuous observation and repeated analysis provides the

opportunity to correct misunderstandings as data is uncovered which indicates them. By periodically reassessing understanding of the SUI, analysts can mitigate the biases they bring to their investigations. An aspect of the method which exposes it to hindsight bias is it's reliance on RID; at the same time however, ignoring the accidents which have occurred in the past would make the process vulnerable to unnecessary blind spots.

Operator Error

ISHA demands consideration of a socio-technical model of the SUI which considers all components of the system. The method identifies hazards, not errors. In taking an approach which diverges from causal chain analysis, and focuses instead on systemic factors, the method prescribes balanced consideration of human, technical and organizational influences on hazard risk. The method's prescription of such a breadth of analysis mitigates the possibility that analysts will unduly focus on the role of human decision making as a hazard.

Migration Towards States of Productivity

The ISHA process is intended for periodic application over time. By increasing the frequency of analysis, the team can get a more current read on the margins of safety which exist in the system. By focusing the Safety Constraint Enforcement Mechanisms (SCEMs) which attempt to observe hazards, analysts can build a repository of safety metrics which identify margins of error in the system which will evolve over time. As analysts cannot predict the future. There is no guarantee that the metrics they choose will be the ones which matter, but careful consideration and modelling of the SUI can lead to insights into the system which the team would not have in the absence of analysis.

6.2.2 Validated Systems Safety Process

Validated system safety processes have a variety of commonalities. We address a number of these commonalities here including how to handle analysis logistics, preliminary tasks including identifying a PHL and performing a PHA. We also address what aspects of ISHA provide insight into how to prioritize hazards for mitigation. We also talk about how modelling is done in ISHA as it is in many hazard analysis techniques, though processes like MIL-STD-882E are not prescriptive on how this is to be done. We briefly address the qualification of risk; this is closely related to the prioritization of hazards for mitigation. Finally, we discuss how ISHA prescribes that analysis results be packaged. This phase of ISHA sets it apart from many existing hazard analysis methods and processes as many provide little guidance on how this should be done, though FMEA and HAZOP do at least

promote tabular reporting structures.

Logistics Activities

The team selection process in ISHA directs analysts to carefully consider team composition. By identifying specific roles and responsibilities which exist in CIS, and which are important for the successful execution of the analysis process, ISHA provides guidance on this basic aspect of logistics. Further, ISHA identifies work by Stamatis [130] and Redmill [111] which can be leveraged for additional guidance.

Contextual Activities

ISHA follows the prescribed tasks of the MIL-STD-882E standard, thus demanding that contextual activities like the PHL, PHA, modelling, and definition of the concept of operations of the SUI be performed.

Prioritization and Qualification of Risk

ISHA prescribes that prioritization of hazards be performed both at the beginning of the PHA phase of the analysis, as well as during the Risk Assessment Code (RAC) phase. The method does not however prescribe what prioritization scheme to use. We have discussed limitations of some common and simple techniques which can be applied for this purpose. We have also proposed some alternatives and highlighted some of the challenges that these methods as well. We discuss prioritization further in our future works section (Section 8.2.2).

Modelling

ISHA provides a prescriptive modelling language based on a combination of concepts derived from the STAMP and FRAM frameworks. This metalanguage provides an expressive medium with which to express socio-technical systems which are investigated using the method.

Qualification of Risk

ISHA's RAC phase is dedicated to the qualification of risk. It relies on the system models developed earlier in the process, paying special attention to the SCEMs which are expressed in the Universal Triangulation Model (UTM).

Packaging Activities

ISHA prescribes the creation of an assurance case from the models and evidence which are generated during its execution. Assurance cases are formal arguments with prescribed structure. This process satisfies the requirement that ISHA specify how to package the analysis outputs.

6.2.3 Clinical Information System Specialization

The literature on clinical information systems highlights many domain specific hazard analysis needs. Two such needs which have received much attention are the need to address the consequence of the use of information systems in safety critical environments and the necessity to understand the sociotechnical interactions which occur in systems which depend heavily on cognition which is distributed not only between human participants in these systems, but also the technology they use in support of their duties. We demonstrate how each of these two need are addressed in ISHA to verify that the method meets these needs.

Information System Needs

ISHA specifically addressed usability, integrity and availability through its incorporation of hazard taxonomies into its ECA, CFA and Process Fault Analysis (PFA) phases. By using the guide words in the taxonomies which focus on each of these aspects of the SUI for the identification of potential faults in the requirements, components and process of the SUI, ISHA ensures their consideration in the analysis.

Sociotechnical Analysis

ISHA provides for sociotechnical analysis with its human/machine agnostic modelling paradigm. Further, the CFA in the method approaches component fault from an abstract level that is not prescriptive of mechanical/electrical based principles such as mean time to failure. Instead, ISHA prescribes that analysts consider reasonable modes of failure for whichever component is under investigation. Further, the interaction analysis prescribed by ISHA encourages consideration of problems with the communications between all components of the SUI, including those that occur between humans and machines. Finally, the breadth of the model which is encouraged in ISHA considers not only the immediate device, but the social organization which surrounds it. The combination of these features of the ISHA method demonstrate how it prescribes socio-technical analysis rather than focusing strictly on software and devices. Lastly, the structure of the UTM in ISHA separates the functions which perform the duties in the SUI from the duties themselves. With little effort, duties

can be reassigned to different functions in the UTM as the SUI evolves. This association could perhaps be more fluid and satisfy the requirement for transitions of responsibility better, but this mechanism is sufficient to support the modelling of these changes.

Chapter 7

Discussion

In Section 7.1 we address some strengths and limitations of our work. We then discuss a pair of issues relevant to the application of the method including the impact of system requirements on the quality of analysis output (Section 7.2) and the consequence of system evolution on the relevance of executed analyses (Section 7.3). We then discuss a series of challenges analysts face in the execution of the method. In Section 7.4 we address the challenge of managing analysis scope and granularity. In Section 7.5, we address the challenge of assessing risk in the face of incomplete data. In Section 7.7, we address challenges in prioritizing identified hazards for mitigation. Finally, in Section 7.6, we discuss the extent to which the analysis performed using Information System Hazard Analysis (ISHA) might be considered complete.

7.1 Strengths and Weaknesses

Compared with other methods, ISHA demonstrates great strength in identifying hazards. This strength arises from the integration of the many techniques used by both the traditional and systemic methods it aggregates. This strength however is tempered by the fact that only a fraction of hazards can be thoroughly addressed in most analyses. The most significant concerns for a particular product are of course addressed, but as safety management is a journey, and not a destination, the process is constantly uncovering new concerns.

ISHA is less prescriptive than many existing methods as it provides flexibility in the selection of methods to be used for the Component Fault Analysis (CFA) and Process Fault Analysis (PFA). Though on the one hand this feature provides strength through flexibility, it also compromises the reproducibility of results. Further, it impedes the integration of results from disparate safety studies as such integration nearly invariably requires normalization.

ISHA also provides a strong system theoretic modelling framework which can provide

insight into the interactions between system components which have often been found to present hazards. This strength exists in the systemic methods from which ISHA was derived as well - System Theoretic Accidents Models and Processes (STAMP) and Functional Resonance Analysis Method (FRAM), but ISHA provides more structure to the modelling proposed in each of these two source methods thus advancing the specificity and semantic clarify of the produced models.

ISHA is a thorough method which excels at hazard identification and which provides substantial guidance on how to assess risk. It also provides clear mechanisms on how to package analysis results into a compelling argument for safety using an assurance case structure. The cost of this rigor is time. Experience with the method has shown that it is substantially more time intensive than either the traditional or systemic methods from which it was derived. This arises from the fact that ISHA prescribes that many of the activities from each of these methods be executed as part of the process. This aggregation of activities is invariably larger than the activities prescribed by any one of the original methods.

Like with Failure Mode and Effects Analysis (FMEA) [120], the validity of ISHA can be challenged on the basis of the disparity in the results it often produces. The hazards identified by any team of analysts is likely to differ. However, unlike FMEA, ISHA at least mitigates this problem to a degree by prescribing a triangulation step in with the hazards identified in the Event Chain Analysis (ECA), CFA, and PFA are cross validated. Further, the prescription that a safety assurance case be constructed in which it is recommended that hazards be allocated to components and/or processes of the System Under Investigation (SUI) mitigates this issue further since such a process will often lead to the discovery of analysis deficiencies.

7.2 The Strength of Requirements

Like any hazard analysis method which depends on requirements, ISHA is subject to their weakness. In doing prospective hazard analysis it is important for analysts to understand the purpose of the SUI. This purpose is expressed in the requirements. Requirements however are notorious for conflicts, absences and vagueness. When the requirements are of low quality, the analysis which can be performed with ISHA suffers. On the other hand, one benefit of this fact is that the execution of the ISHA method will require that analysts pressure the owners of the SUI into providing better documentation and thus improving the requirements.

7.3 The Evolution of the Environment

As will be discussed further in Chapter 8, the analysis prescribed by ISHA occurs at a point in time. The method is intended to be repeated periodically, but is not structured in a way which supports continuous monitoring of the safety state of the SUI. ISHA does encourage the development and integration of safety monitors throughout the SUI both for their inherent value and for the ease they bring to the method execution, but the method itself is about investigating a static model of the SUI. The SUI itself will change over time, as will the environment in which it resides [76]. The consequence of this truth is that the validity of the assumptions about the nature and purpose of the system will degrade over time, and the environment in which it resides will evolve. This highlights the necessity for constant vigilance when monitoring safety. It also highlights the value which can be achieved by identifying and monitoring metrics which provide insight into the margins of safety in which the SUI operates.

7.4 Scoping Breadth and Granularity

As mentioned in Section 7.1, ISHA can be a time consuming process. Two potential pitfalls in ISHA are the scoping traps of breadth and granularity. These weakness are not unique to ISHA or even to hazard analysis, but they are dangers which must be considered when engaging in modelling based activities.

We describe the breadth trap to be the problem of choosing the scope of the analysis in the abstract. The analysts' choice of how much of a process and how much of the environment to include in the system scope are important decisions which will have a substantial impact on the cost of the analysis and on its success.

The danger however, does not pass once the boundaries of the SUI have been drawn. Within the SUI, there is invariably a large degree of granularity to which the system can be modelled. A sufficient degree of granularity is always required to identify hazards as it is in the details where these are found. However, abstraction is a necessary tool for analysts to keep their investigation within the scope of their budget. Failing to heed this warning will lead to an analysis which is never completed because some aspect of the SUI is investigated in depth, while other aspects of the SUI remain insufficiently inspected.

7.5 Risk and Incomplete Data

Another major challenge in effectively executing a hazard analysis method is accurately determining risk. Leveson warns explicitly against using numeric probabilities for this task as retrospective analysis in the past have cast significant doubt on the capacity of analysts to accurately predict the likelihoods of improbable events [76]. ISHA prescribes the use of nominal scores for risk assessment. Assigning these nominal scores however is a craft that requires domain expertise. No amount of data about a given hazard can be considered complete. There is always something more — however trivial — which could be known before assigning a score to the detectability, occurrence or severity of a hazard.

7.6 Completeness

The combination of the challenges discussed above make guarantees of completeness in hazard analysis difficult to make. ISHA does not make a guarantee of completeness. It does however, provide some evidence of comprehensiveness through its triangulation phase and through its assurance case generation phase. Each of these phases is intended to improve saturation of hazard identification. The triangulation phase does so by ensuring that hazards identified across the three approaches to identification are consistent and complete - at least between the methods used. The assurance case generation phase is used to package analysis results, but it is also used to identify weaknesses in the comprehensiveness of the analysis thus promoting completeness. The complete risk of the SUI is always represented in the assurance case since it incorporates an acknowledgement that the residual risk of the SUI is acceptable. The arguments about the completeness of hazard analysis essentially challenge the analysts' understanding of what is included in the residual risk. A poorer hazard analysis may include larger and more significant unknowns in the residual risk, where stronger analysis will include lesser unknowns of lower significance in the residual risk.

7.7 Prioritization

Our final topic of discussion is the prioritization of hazards for analysis and mitigation. This arises explicitly in two places in the ISHA method. The first time this challenge arises is when the Preliminary Hazard List (PHL) is identified but before the analysis of these hazards is performed. With an SUI which is in a class of system's that has a rich history of hazard analysis, it is likely that the initial PHL may be longer than can be addressed in

the scope of the analysis budget. Even if all of those hazards will eventually be addressed, the analysis budget will likely be divided into phases and so each phase will only consider a subset of the full list. The identification of this subset poses a substantial challenge for analysts because they cannot rely on a prescriptive mathematically based method to make their decisions. Instead, looser heuristic methods, and methods which link available skill sets and experience in the analyst team with the hazards in the list, are what is needed. Consequently, the defense of these decisions requires the analysts to negotiate with the study initiator more than it requires them to engineer. The second time this challenge arises is after the hazards for all of the identification approaches have had their risk assessed, and once they have all been tabulated. Shebl's work on the validity of the FMEA Risk Probability Number (RPN) [119] inspired our initial skepticism in the traditional approach to risk analysis. She highlighted a number of problems with the mathematics of the RPN approach including the sparsity of the risk matrix which results, and the large disparities in calculated risk which are observed when a given hazard has one high value in the risk tuple. Collectively, these facts motivated our prescription of the generation of risk matrices in which Risk Assessment Codes (RACs) are explicitly assigned for all risk tuples.

Even having addressed the mathematics of the RPN, substantial challenges remain in identifying which hazards to address. This difficulty exists because hazards are not independent. They are emergent properties of the SUI which result from a combination of component weaknesses and interaction problems. As a consequence, consideration of hazards in isolation - even when using a RAC lens, does not necessarily lead to a sufficient reduction in system risk. The level of reduction which would be sufficient must be determined by the analysts and may be based on a Safety Integrity Level (SIL) basis or on some other basis prescribed by the domain of the SUI. We provided a number of simple prioritization schemes in Chapter 4, but do not suggest that they will produce the most desirable prioritization. The order in which hazards are mitigated, and the completeness with which they are mitigated must be determined by analysts based on their analysis of the risks those hazards pose. There is a balance between managing hazards which may more likely lead to high severity outcomes but at a lower rate of occurrence, and with hazards which may more likely lead to lower severity outcomes but with greater occurrence. Further, the detectibility of the hazard would influence how long such a condition could persist and thus the number of accidents which could occur before the hazard was mitigated. Further, spending more effort on a given mitigation may improve the probability of getting the mitigation "right" and thus preventing subsequent accidents. This must be balanced so as to not spend excessive effort on one hazard to the neglect of another. Spending enough time on one hazard initial may change the risk profile thus elevating the risk of a second hazard above the first and thus changing the necessary prioritization. We only suggest that negotiating the strat-

egy to be taken for prioritizing hazards will promote sufficing outcomes when considering the evidence generated in analysis including the guidance of the analysts who review the evidence. In taking this course of action the residual risk of the system can be reduced such that it is As Low As Reasonably Practicable (ALARP) given the budget for the safety management of the system during the period of the analysis and mitigation.

Chapter 8

Conclusions and Future Work

8.1 Contributions

We have provided a number of contributions in this thesis which we list below and subsequently discuss in sequence:

1. *Information System Hazard Analysis (ISHA)*, a new method for hazards analysis in Clinical Information System (CIS)
2. A series artifacts to support the execution of the method
   (a) A *Formal Information Model*
   (b) A series of *Design Patterns* which can be applied in the context of the information model
   (c) A *Hazard Factor Taxonomy*
   (d) A CIS model (*STAMP EMR*) constructed to support the application of the ISHA method in the medical context.
3. A series of case studies which demonstrate the feasibility of our new method
4. A systematic review of literature and incident reports which classify identified hazards against an a priori hazard model which is based in the Leveson's systemic System Theoretic Accidents Models and Processes (STAMP) framework.

8.1.1 Information System Hazard Analysis

We have developed ISHA, a novel technique for assessing the safety of CIS. ISHA has been synthesized from extant hazard analysis methods and safety analysis frameworks. The

synthesized hazard analysis methods include Fault Tree Analysis (FTA), Event Tree Analysis (ETA), Failure Mode and Effects Analysis (FMEA), HAZard OPerability (HAZOP), STAMP, and Functional Resonance Analysis Method (FRAM). Influencing this synthesis were the ideas presented in the frameworks proposed by Rogers[113], Huchins [55], Holden [51, 47, 46], Carayon [15] and Sittig [124].

The method is grounded in the safety process prescribed by United States Department of Defense Standard Practice System Safety (MIL-STD-882E). It consists of the following steps:

1. Select team
2. Source concept of operation
3. Source requirements
4. Source model
5. Preliminary Hazard List (PHL)
6. Preliminary Hazard Analysis (PHA) (PHA)/Universal Triangulation Model (UTM))
7. Requirements Analysis [Event Chain Analysis (ECA)]
8. Component Fault Analysis (CFA)
9. Process Fault Analysis (PFA)
10. Assurance case development
11. Generate recommendations
12. Repeat

The requirements for the method were validated through literature review and argument (Chapter 6) while its feasibility was demonstrated through demonstration in running examples (Chapters 3 and 4). We are not aware of any existing safety analysis process that:

- is systemic
- is grounded in a validated safety method (MIL-STD-882E)
- incorporates both control loop structure modelling and explicit formalized consideration of resonance impacts
- is explicitly focused on the needs of CIS

8.1.2 Application Supports

Formal Information Model

We introduced the formal information model for the ISHA method in Chapter 5. The model synthesizes a hazard model which we created based on our exploration of extant analysis methods with the Structured Assurance Case Metamodel (SACM) developed by the Object Management Group (OMG) to formalize Kelly's work on assurance cases [61].

Design Patterns

To support the application of the formal information model we contribute, we have also developed a series of design patterns (Section 5.3.2):

1. Complex Operation
2. Delegation of Responsibility
3. Evolution
4. Peripheral
5. Inversion of Control
6. Transition of Responsibility

The *complex operation* pattern can be used to model situations in which their are multiple sources of control including duty monitoring for example. The *delegation of responsibility* pattern provides analysts with a mechanism to refine UTM components into subsystems which are guided by the original system component. The *evolution* pattern provides a mechanism to model the changes expressed or experienced by an System Under Investigation (SUI) component over the course of time, including for example, the implementation of watchdog functionality. The *peripheral* pattern facilitates the modelling of complex observation or activation of SUI components. The *inversion of control* pattern supports information hiding thus facilitating analysts' consideration of the SUI interactions from multiple points of view. Finally, the *transition of responsibility* pattern demonstrates how analysts can compare and contrast system behaviour based on disparate duty assignments.

Hazard Factor Taxonomy

Through literature review we have developed a taxonomy of terms relevant to safety in CIS [83]. This taxonomy can be married with nomenclatures like the one proposed by Phillips and Gong [105] to evolve and supplement CIS PHLs.

STAMP EMR

We have specialized Leveson's STAMP [76] framework to the domain of CIS. In doing so, we generated the STAMP-EMR model (Section 3.5.2) [142, 143] which we evaluated through an extended literature review [81, 82] and an analysis of incident reports from a voluntary reporting database [29].

8.1.3 Case Studies

We have performed a pair of published case studies [83, 140] and have provided an additional pair of case studies in this dissertation (Chapters 3, 4), demonstrating the application of the ISHA method on real life CIS, rather than on "toy" examples.

8.1.4 Systematic Review

The concluding work of our efforts in developing STAMP EMR provides the Retrospective Incident Data (RID) based evidence necessary to construct a PHL. By combining the STAMP EMR taxonomy with the nomenclature provided in [83] the incident reports in [29] could be reviewed to create a generic PHL for CIS.

8.2 Future Work

8.2.1 Clinical Information System Specific Work

Adoption Measurement and Capability Maturity

One of the critical activities in the ISHA process is the determination of risk performed in the Risk Assessment Code (RAC) phase. Risk is challenging to assess in sociotechnical systems because perfect information is never feasibly attainable. One of the most important factors reiterated in the literature is understanding tooling/workflow mismatch. Health care professionals claim time and again that their CIS tooling is "clunky", overwhelming, slow, and that these tools induce burdensome tasks that slow the process of care. These issues often highlight workflow mismatches. When such mismatches occur, healthcare professionals focus on the task at hand – the care of their patients. They thus devise workarounds [72, 117, 116] which though likely improve patient safety overall just by improving patient throughput, compromise the integrity of the CIS and thus the safety of those patients whose electronic records are not correctly maintained. One key approach to recognizing these hazards before they realize into patient safety incidents is the use of adoption measurement

and capability maturity models.

By measuring adoption over the course of time and regularly checking in with care providers in the CIS, analysts can identify barriers to adoption and workarounds before they result in medical error. A prerequisite to successful execution however, is the organizational maturity to be able to respond to these requests for change fast enough. Further, monitoring the safety of the system over time, and the responsiveness of the technical support team to the clinical requests can allow organizations to allocate resources in such a way as to simultaneously optimize both productivity and safety.

A Preliminary Hazard List for Clinical Information Systems

In the PHL phase of ISHA, analysts are advised to seek a validated PHL on which to base their analysis. This is a challenge for CIS in general as few such lists exist. In our case study of the Generalized Insulin Infusion Pump (GIIP) we were able to identify one, but that was an exceptional discovery which was not entirely accidental. The case study was pursued because of our awareness of the breadth of information available on the device. In general, no such list can be found. Developing a new list for more generic CIS could provide great value to future analysts. One approach to this problem would be to take the results of our studies of incident reports [29], and literature [81, 82], combine it with the works of others [141, 93, 79, 101, 138, 16, 17, 95] and use a foundational nomenclature like the one proposed by Phillips and Gong [105]. With these tools, and a review of a collection of incident reports like those we studied [29], or better still, those studied by Palojoki [101], a robust PHL could be constructed to support the execution of ISHA against a broad range of CIS.

Information System Hazard Analysis Tooling

A principle challenge in the execution of ISHA is the lack of tooling support. A strong tool for this purpose would have to satisfy a number of requirements. The tool would need to support integration with existing requirements management systems. Though ISHA is a prospective method, it will be applied to systems which are already under development. Further, there already exist tools which are well designed to support system development and so building ISHA tooling to support this task as well would be inefficient. ISHA tooling would however need to support PHL management including storing sourced documents, annotating hazard definitions in those documents with the terminology and nomenclature used in analysis, and even translating the language used in the source documents to a more standardized form. The larger part of the tool would have to support the modelling for the UTM, the annotation of components with Safety Constraint Enforcement Mechanisms

(SCEMs), hazards and constraints. One more feature that would be important would be the facility to track risks for the identified hazards.

Strengthen Resonance Focus

As with the work of Leveson [76], ISHA provides mechanisms to analyze the impact of resonance within the SUI; however, the strength of the is focus remains weak. Improving the incorporation of Hollnagel's [52] consideration of duty semantics, ports and feedback loops could significantly improve the impact of the ISHA method.

Method Comparisons

Though we have performed a number of case studies using ISHA, we have not performed comparative case studies using other methods. This would be a challenging experiment to perform. In order to assess the quality of the two methods, independent executions would need to be performed on the same systems. This independence of observation would likely yield divergent results [119, 120]. The great challenge in this type of study would be in the interpretation of results; however, in the absence of comparative studies it is difficult to assess the relative quality of analysis methods.

8.2.2 Generic Hazard Analysis Work

Critical Evaluation of Hazard Analysis Techniques

One of our approaches to evaluation of this new method was to investigate the literature for validated assessment instruments. Having found none, we continued to search for articles describing comparative evaluations of hazard analysis techniques. The findings in these literature searches including a small number of papers which presented widely divergent assessment metrics. This effort revealed the necessity to develop a stronger theoretic foundation for hazard analysis method evaluation.

Generalized Software Hazard Checklist

Another challenge in performing the PHL phase of ISHA is that no widely accepted generalized software hazard checklist could be identified. Ericson [28] provides a series of hazard checklists for a variety of system dimensions, but does not provide a software specific checklist. Zhang [149] includes a subsection of his GIIP PHL which addresses software, but this is not easily identified when an analyst is seeking a software focused hazard checklist. This

void in the available validated checklists makes it challenging to defensibly assess the quality of PHLs which are used in ISHA analyses.

Hazard and Contributing Factor Prioritization

A further challenge in applying the ISHA method, and any other method that requires prioritization of hazards is choosing how to perform the prioritization.

Qualitative Approaches Early on in the analysis process, analysts will at times be required to prioritize the analysis of hazards and contributing factors when they have little data. Mathematical approaches to this prioritization will at times not be feasible. Analysts will have to make decisions at a coarser level to reduce the size of the base PHL to fit the analysis budget. Developing formalized approaches to this process would provide analysts with tools which could then be validated through application to this problem.

Risk Assessment Code Based Approaches When applying hazard analysis techniques which leverage RACs (e.g., FMEA, ISHA, MIL-STD-882E), analysts are still challenged in determining how to prioritize hazards and contributing factors. The traditionally prescribed approach to this problem is to multiply the risk components together to form an Risk Probability Number (RPN), and to then prioritize based on the numeric value of the RPN. This approach has been criticized [120] for its validity based on the distribution of the resultant RPN over the possible result area/volume, equivalence classes of risk triplets, and the variance of the RPN based on small variance in the lower numbered risk components. Determining more sound methods of computing risk based on the risk tuple and differences in scales could provide valuable insights into how to better quantify risk.

Universal Triangulation Model Based Approaches Graph based computation may yield insights into how to manage areas of the UTM which are more heavily annotated with contributing factors and hazards. Identifying potential new SCEMs which could be added may yield highly beneficial mitigations which come at a low price point on account of the simplicity of their implementation. What is needed to begin however, is a method to quantify annotation density in a pragmatic fashion that supports the identification of mitigation needs.

Common Cause Failure Approaches Another potential avenue for prioritizing hazards is to seek common cause failures. A liberal view of what might be considered a "common" cause may be beneficial to the pragmatic analyst. The goal with this approach is to

consider a range of potential mitigations and choose those that address the best set of hazards. One potential approach to this task would be to use language reduction in the PHL to identify commonalities. If more computable language is used to describe the hazards, then analysts will have an easier time of this reduction.

8.3 Summary

We have developed ISHA [80], a novel hazard analysis method for information systems with a focus on CIS which is based on a combination of MIL-STD-882E, and the works of Hollnagel [52] and Leveson [76]. We have evaluated the method through application in a series of case studies - [140, 83, 142, 143] and two running examples in this dissertation. We have validated the method by construction and also by inspection through assessment of the works and recommendations of Hollnagel [52] and Leveson [76]. We have also developed accompanying artifacts which can be employed in the application of the ISHA method: a formal information model, a series of design patterns, a hazard analysis model for CIS (STAMP EMR), and a safety factors taxonomy. We developed STAMP EMR through literature [81, 82] and incident report [29] reviews. The safety factors taxonomy [83] is a synthesis of works by Alonso-Rios [3] and McCall [84], that is influenced by a range of other works as well. Though ISHA has been developed with a specific focus on CIS, we assert that only minor adaptations would be necessary to apply it to non-clinical information systems.

Bibliography

[1] Agency for Healthcare Research and Quality (AHRQ). National Healthcare Quality Report 2008, 2009.

[2] Christopher Alexander, Sara Ishikawa, Murray Silverstein, Joaquim Romaguera i Ramió, Max Jacobson, and Ingrid Fiksdahl-King. *A pattern language*. Gustavo Gili, 1977.

[3] David Alonso-Ríos, Ana Vázquez-García, Eduardo Mosqueira-Rey, and Vicente Moret-Bonillo. Usability: a critical analysis and a taxonomy. *International Journal of Human-Computer Interaction*, 26(1):53–74, 2009.

[4] Talat Ambreen, Naveed Ikram, Muhammad Usman, and Mahmood Niazi. Empirical research in requirements engineering: trends and opportunities. *Requirements Engineering*, pages 1–33, 2016.

[5] ISO ANSI. Ts 18308 health informatics-requirements for an electronic health record architecture. *ISO (Ed.)*, 2003.

[6] Joan S Ash, Marc Berg, and Enrico Coiera. Some unintended consequences of information technology in health care: the nature of patient care information system-related errors. *Journal of the American Medical Informatics Association*, 11(2):104–112, 2004.

[7] Joan S Ash, Dean F Sittig, Richard Dykstra, Emily Campbell, and Kenneth Guappone. The unintended consequences of computerized provider order entry: findings from a mixed methods exploration. *International journal of medical informatics*, 78:S69–S76, 2009.

[8] Anaheed Ayoub, BaekGyu Kim, Insup Lee, and Oleg Sokolsky. A systematic approach to justifying sufficient confidence in software safety arguments. In *International Conference on Computer Safety, Reliability, and Security*, pages 305–316. Springer, 2012.

[9] Jes Bassi and Colin Partridge. Electronic prescribing: Workflow analysis handbook, 2011.

[10] David Bates. FDASIA recommendations. https://www.healthit.gov/facas/sites/faca/files/FDASIARecommendations030913_Final.pptx, 2015. last accessed 2016.

[11] JB Battles and RJ Lilford. Organizing patient safety research to identify risks and hazards. *Quality and Safety in Health Care*, 12(suppl 2):ii2–ii7, 2003.

[12] Marilyn Sue Bogner. There is more to error in healthcare than the care provider. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, number 11 in 49, pages 952–954. SAGE Publications, 2005.

[13] Christopher J Booth and Gediminas P Kurpis. *The new IEEE standard dictionary of electrical and electronics terms*. IEEE New York, USA, 1993.

[14] Canadian Medical Protective Association (CPMA). "Patient safety incident" terminology. https://www.cmpa-acpm.ca/serve/docs/ela/goodpracticesguide/pages/patient_safety/Understanding_harm/patient_safety_incident_terminology-e.html#lc1. last accessed 2016.

[15] P Carayon, A Schoofs Hundt, BT Karsh, AP Gurses, CJ Alvarado, M Smith, and P Flatley Brennan. Work system design for patient safety: the seips model. *Quality and Safety in Health Care*, 15(suppl 1):i50–i58, 2006.

[16] Joanne Castro, Gerard; Buczkowski, Lisa; Hafner. The Contribution of Sociotechnical Factors to Health Information Technology Related Sentinel Events. *Journal on Quality and Patient Safety*, 42(2), 2016.

[17] Andrew Chang, Paul M Schyve, Richard J Croteau, Dennis S Oleary, and Jerod M Loeb. The JCAHO patient safety event taxonomy: a standardized terminology and classification schema for near misses and adverse events. *International Journal for Quality in Health Care*, 17(2):95–105, 2005.

[18] Yihai Chen, Mark Lawford, Hao Wang, and Alan Wassyng. Insulin pump software certification. In *International Symposium on Foundations of Health Informatics Engineering and Systems*, pages 87–106. Springer, 2013.

[19] Ka-Chun Cheung, Willem van der Veen, Marcel L Bouvy, Michel Wensing, Patricia M L A van den Bemt, and Peter A G M de Smet. Classification of medication incidents associated with information technology. *J. Am. Med. Inform. Assoc.*, 21(e1):e63–70, 2014.

[20] Richard I Cook and Michael F O Connor. Thinking about accidents and systems. In *Medicat. Saf. A Guid. Healthc. Facil.*, pages 73–88. American Society of Health Systems Pharmacists, 2005.

[21] Louis Anthony Tony Cox Jr. What's wrong with hazard-ranking systems? an expository note. *Risk Analysis*, 29(7):940–948, 2009.

[22] Kathrin Cresswell and Aziz Sheikh. Organizational issues in the implementation and adoption of health information technology innovations: an interpretative review. *International journal of medical informatics*, 82(5):e73–e86, 2013.

[23] Ewen Denney and Ganesh Pai. A lightweight methodology for safety case assembly. In *International Conference on Computer Safety, Reliability, and Security*, pages 1–12. Springer, 2012.

[24] Joseph DeRosier, Erik Stalhandske, James P Bagian, and Tina Nudell. Using health care failure mode and effect analysis: the va national center for patient safetys prospective risk analysis system. *The Joint Commission journal on quality improvement*, 28(5):248–267, 2002.

[25] Diabetes Canada. `http://guidelines.diabetes.ca/cdacpg_resources/CPG_Quick_Reference_Guide_WEB.pdf`. last accessed 2017.

[26] US DoD. Mil-std-882e, department of defense standard practice system safety. *US Department of Defense*, 2012.

[27] Robert J Ellison and Andrew P Moore. Trustworthy refinement through intrusion-aware design (triad), 2003.

[28] Clifton A Ericson et al. *Hazard analysis techniques for system safety*. John Wiley & Sons, 2015.

[29] J Facelli and C Giraud-Carrier, editors. *Assessing STAMP EMR with Electronic MedicalRecord Related Incident Reports - Case Study: Manufacturer and User Facility Device Experience Database*. International Conference on Healthcare Informatics, 2017. accepted - publication pending.

[30] FDA. Common Good Manufacturing Processes. `https://www.fda.gov/Drugs/DevelopmentApprovalProcess/Manufacturing/ucm169105.htm`. last accessed 2017.

[31] FDA. Generic infusion pump research project. `https://rtg.cis.upenn.edu/gip/`. last accessed 2017.

[32] FDA. Maude report 2904495. MAUDE repoert MDR Key 2904495. last accessed 2013.

[33] FDA. Maude report 975910. MAUDE report MDR Key 975910. last accessed 2013.

[34] FDA. Medical devices; medical device data systems. https://www.gpo.gov/fdsys/pkg/FR-2011-02-15/pdf/2011-3321.pdf. last accessed 2016.

[35] FDA and Center for Devices and radiological Health. Infusion Pumps Total product life cycle - Guidance for industry and FDA Staff, 2014.

[36] US Food, Drug Administration, et al. Infusion pumps total product life cycle: guidance for industry and fda staff. *Food and Drug Administration Std*, pages 0910–0766, 2014.

[37] Erich Gamma. *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995.

[38] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Design patterns: Abstraction and reuse of object-oriented design. In *European Conference on Object-Oriented Programming*, pages 406–431. Springer, 1993.

[39] R Garcia. This isn't my information! the impact of accurate identity management on patient safety. *Health management technology*, 34(3):10, 2013.

[40] GSN Working Group et al. Gsn community standard version 1, 2011.

[41] Yong Y Han, Joseph A Carcillo, Shekhar T Venkataraman, Robert S B Clark, R Scott Watson, Trung C Nguyen, Hülya Bayir, and Richard a Orr. Unexpected Increased Mortality Arter Implementation of a Commercially Sold Computerized Physician Order Entry System. *Pediatrics*, 116(6):1506–1512, 2005.

[42] Cindy Harnett. Nanaimo doctors say electronic health record system unsafe, should be shut down. http://www.timescolonist.com/news/local/nanaimo-doctors-say-electronic-health-record-system-unsafe-should-be-shut-down-1.2264497. last accessed 2017.

[43] M Harrison, R Koppel, and S Bar-Lev. Unintended consequences of information technologies in healthcare - an interactive sociothecnical analysis. *J. Am. Med. Informatics Assoc.*, 14:542–549, 2007.

[44] André Alexandersen Hauge and Ketil Stølen. A pattern-based method for safe control systems exemplified within nuclear power production. In *SAFECOMP*, pages 13–24. Springer, 2012.

[45] Herbert William Heinrich et al. Industrial accident prevention. a scientific approach., 1941.

[46] Richard J Holden. Cognitive performance-altering effects of electronic medical records: an application of the human factors paradigm for patient safety. *Cognition, Technology & Work*, 13(1):11–29, 2011.

[47] Richard J Holden and Ben-Tzion Karsh. A theoretical model of health information technology usage behaviour with implications for patient safety. *Behaviour & Information Technology*, 28(1):21–38, 2009.

[48] Erik Hollnagel. The functional resonance analysis method. `http://www.functionalresonance.com/`. last accessed 2017.

[49] Erik Hollnagel. Cognitive ergonomics: it's all in the mind. *Ergonomics*, 40(10):1170–1182, 1997.

[50] Erik Hollnagel. *Cognitive reliability and error analysis method (CREAM)*. Elsevier, 1998.

[51] Erik Hollnagel. Understanding accidents-from root causes to performance variability. In *Human factors and power plants, 2002. proceedings of the 2002 ieee 7th conference on*, pages 1–1. IEEE, 2002.

[52] Erik Hollnagel. *FRAM, the functional resonance analysis method: modelling complex socio-technical systems*. Ashgate Publishing, Ltd., 2012.

[53] C Michael Holloway. Safety case notations: Alternatives for the non-graphically inclined? In *System Safety, 2008 3rd IET International Conference on*, pages 1–6. IET, 2008.

[54] Jan Horsky, Gilad J Kuperman, and Vimla L Patel. Comprehensive analysis of a medication dosing error related to cpoe. *Journal of the American Medical Informatics Association*, 12(4):377–382, 2005.

[55] Edwin Hutchins. The social organization of distributed cognition., 1991.

[56] IEEE. Ieee standard for software safety plans., 1994.

[57] A Ingrey, P Lereverent, and A Hildebrant. Safety integrity level. *Manual PEPPERL+ FUCHS*, 2007.

[58] Institute of Electrical and Electronics Engineers (IEEE). IEEE standard glossary of software engineering terminology. Standard 610.12-1990(R2002), IEEE, 1990 R 2002.

[59] CW Johnson. Case studies in the failure of healthcare information systems. *available online a t http://www. dcs. gla. ac. uk/˜ johnson/papers/AHRQ/case_study. pd f*, 2010.

[60] CW Johnson. Identifying common problems in the acquisition and deployment of large-scale, safety–critical, software projects in the US and UK healthcare systems. *Safety Science*, 49(5):735–745, 2011.

[61] Tim Kelly and John McDermid. Safety case patterns-reusing successful arguments. *IEE Colloquium on Understanding Patterns and their Application to Systems Engineering*, 1998.

[62] Tim Kelly and Rob Weaver. The Goal Structuring Notation A Safety Argument Notation. *Elements*, 2004.

[63] Timothy Patrick Kelly. Arguing Safety A Systematic Approach to Managing Safety Cases, 1998.

[64] TP Kelly and JA Mcdermid. Safety case construction and reuse using patterns. In *16th International Conference on Computer Safety and Reliability (SAFECOMP'97*. Citeseer, 1997.

[65] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin Kirda. Cutting the gordian knot: A look under the hood of ransomware attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 3–24. Springer, 2015.

[66] Ibrahim Abdullah Khawaji. *Developing system-based leading indicators for proactive risk management in the chemical processing industry*. PhD thesis, Massachusetts Institute of Technology, 2012.

[67] Tae-eun Kim, Salman Nazir, and Kjell Ivar Øvergård. A stamp-based causal analysis of the korean sewol ferry accident. *Safety science*, 83:93–101, 2016.

[68] Anneke G Kleppe, Jos B Warmer, and Wim Bast. *MDA explained: the model driven architecture: practice and promise*. Addison-Wesley Professional, 2003.

[69] Trevor A Kletz. *HAZOP and HAZAN: identifying and assessing process industry hazards*. IChemE, 1999.

[70] Linda T Kohn, Janet M Corrigan, and Molla S Donaldson. *To err is human: Building a Safer Health System*. IOM, 2000.

[71] Ross Koppel, Abigail Cohen, Brian Abaluck, a Russell Localio, Stephen E Kimmel, and Brian L Strom. Role of Computerized. *J. Am. Med. Assoc.*, 293(10):1197–1203, 2013.

[72] Ross Koppel, Tosha Wetterneck, Joel Leon Telles, and Ben-Tzion Karsh. Workarounds to barcode medication administration systems: their occurrences, causes, and threats to patient safety. *Journal of the American Medical Informatics Association*, 15(4):408–423, 2008.

[73] Andre Kushniruk, Helen Monkman, Elizabeth Borycki, and Joseph Kannry. User-centered design and evaluation of clinical information systems: A usability engineering perspective. In *Cognitive Informatics for Biomedicine*, pages 141–161. Springer, 2015.

[74] Nancy G Levenson. *System safety and computers*. Addison Wesley, 1995.

[75] Nancy G Leveson. The use of safety cases in certification and regulation, 2011.

[76] Nancy G Leveson. *Engineering a safer world: Systems thinking applied to safety*. Mit Press, 2012.

[77] Karin Lundblad, Josephine Speziali, Rogier Woltjer, and Jonas Lundberg. Fram as a risk assessment method for nuclear fuel transportation. In *Proceedings of the 4th International Conference Working on Safety*, volume 1, pages 223–1, 2008.

[78] Robyn R Lutz. Software engineering for safety: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 213–226. ACM, 2000.

[79] Farah Magrabi, Maureen Baker, Ipsita Sinha, Mei-Sing Ong, Stuart Harrison, Michael R Kidd, William B Runciman, and Enrico Coiera. Clinical safety of england's national programme for it: A retrospective analysis of all reported safety events 2005 to 2011. *International journal of medical informatics*, 84(3):198–206, 2015.

[80] Fieran Mason-Blakley and Ryan Habibi. Prospective hazard analysis for information system. In *Healthcare Informatics (ICHI), 2014 IEEE International Conference on*, pages 256–265. IEEE, 2014.

[81] Fieran Mason-Blakley and Jens Weber. CIS system hazards derived from literature using systems and human factors perspectives. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 419–428. ACM, 2012.

[82] Fieran Mason-Blakley and Jens Weber. A systems theory classification of emr hazards: Preliminary results. *Enabling Health and Healthcare Through ICT: Available, Tailored and Closer*, 183:214, 2013.

[83] Fieran Mason-Blakley, Jens Weber, Morgan Price, and Abdul Roudsari. Hazard analysis for electronic medical document exchange. In *Proceedings of the 2014 Foundations in Health Informatics Engineering and Systems*, 2014. publication pending.

[84] James A McCall. Quality factors. *encyclopedia of Software Engineering*, 1994.

[85] Allison B McCoy, Lemuel R Waitman, Julia B Lewis, Julie A Wright, David P Choma, Randolph A Miller, and Josh F Peterson. A framework for evaluating the appropriateness of clinical decision support alerts and responses. *Journal of the American Medical Informatics Association*, 19(3):346–352, 2012.

[86] Clement J McDonald. Computerization can create safety hazards: a bar-coding near miss. *Annals of Internal Medicine*, 144(7):510–516, 2006.

[87] Clement J McDonald, Paul C Tang, and George Hripcsak. Electronic health record systems. In *Biomedical Informatics*, pages 391–421. Springer, 2014.

[88] Dee McGonigle and Kathleen Mastrian. *Nursing informatics and the foundation of knowledge*. Jones & Bartlett Publishers, 2014.

[89] Shailaja Menon, Hardeep Singh, Ashley ND Meyer, Elisabeth Belmont, and Dean F Sittig. Electronic health record–related safety concerns: a cross-sectional survey. *Journal of Healthcare Risk Management*, 34(1):14–26, 2014.

[90] Merriam-Webster. Merriam-webster. `https://www.merriam-webster.com/`. last accessed 2017.

[91] Blackford Middleton, Meryl Bloomrosen, Mark A Dente, Bill Hashmat, Ross Koppel, J Marc Overhage, Thomas H Payne, S Trent Rosenbloom, Charlotte Weaver, and Jiajie Zhang. Enhancing patient safety and quality of care by improving the usability of electronic health record systems: recommendations from amia. *Journal of the American Medical Informatics Association*, 20(e1):e2–e8, 2013.

[92] Daniel R Murphy, Brian Reis, Himabindu Kadiyala, Kamal Hirani, Dean F Sittig, Myrna M Khan, and Hardeep Singh. Electronic health record–based messages to primary care providers: valuable information or just noise? *Archives of internal medicine*, 172(3):283–285, 2012.

[93] Risa B Myers, Stephen L Jones, and Dean F Sittig. Review of Reported Clinical Information System Adverse Events in US Food and Drug Administration Databases. *Appl. Clin. Inform.*, 2(1):63–74, 2011.

[94] Kshirasagar Naik and Priyadarshi Tripathy. *Software testing and quality assurance: theory and practice*. John Wiley & Sons, 2011.

[95] NCC MERP. Taxonomy of medication errors, 2004.

[96] Paul S Nelson. A stamp analysis of the lex comair 5191 accident. *Master's thesis, Lund*, 2008.

[97] David Nouvel, Sébastien Travadel, and Erik Hollnagel. Introduction of the concept of functional resonance in the analysis of a near-accident in aviation. In *33rd ESReDA Seminar: Future challenges of accident investigation*, pages 9–pages, 2007.

[98] Government of British Columbia. `https://www2.gov.bc.ca/gov/content/health/accessing-health-care/home-community-care/care-options-and-cost/long-term-residential-care`. last accessed 2017.

[99] Open EHR. The open EHR project. `http://www.openehr.org/`. last accessed 2017.

[100] Oxford English Dictionary. Oxford English Dictionary. `http://www.oed.com/`.

[101] Sari Palojoki, Matti Mäkelä, Lasse Lehtonen, and Kaija Saranto. An analysis of electronic health record–related patient safety incidents. *Health informatics journal*, page 1460458216631072, 2016.

[102] Vimla L Patel, David R Kaufman, and Jose F Arocha. Emerging paradigms of cognition in medical decision-making. *Journal of biomedical informatics*, 35(1):52–75, 2002.

[103] Thomas Payne and Kent A Beckton. Architecture of clinical computing systems. *Practical Guide to Clinical Computing Systems: Design, Operations, and Infrastructure*, page 13, 2008.

[104] Steven J Pereira, Grady Lee, and Jeffrey Howard. A system-theoretic hazard analysis methodology for a non-advocate safety assessment of the ballistic missile defense system. Technical report, DTIC Document, 2006.

[105] Win Phillips and Yang Gong. Developing a nomenclature for emr errors. *Human-Computer Interaction. Interacting in Various Application Domains*, pages 587–596, 2009.

[106] Morgan Price, James Lai, Tyrone Austen, and Jes Bassi. Emr adoption tools. `http://ehealth.uvic.ca/resources/tools/emradoption/EMRAdoption.php`. last accessed 2017.

[107] Jane Radatz, Anne Geraci, and Freny Katki. Ieee standard glossary of software engineering terminology. *IEEE Std*, 610121990(121990):3, 1990.

[108] Jens Rasmussen. Risk Management in a Dynamic Society: a Modelling Problem. *Saf. Sci.*, 273(2):183–213, 1997.

[109] Arnab Ray and Rance Cleaveland. Constructing safety assurance cases for medical devices. *2013 1st Int. Work. Assur. Cases Software-Intensive Syst. Assur. 2013 - Proc.*, pages 40–45, 2013.

[110] James Reason. *Human error*. Cambridge university press, 1990.

[111] Felix Redmill, Morris Chudleigh, and James Catmur. *System safety: HAZOP and software HAZOP*. Wiley Chichester, 1999.

[112] John Ribeiro. Microsoft blames u.s. stockpiled vulnerability after wannacry ransomware attack. `http://www.pcworld.com/article/3196523/security/microsoft-blames-us-stockpiled-vulnerability-for-ransomware-attack.html`. last accessed 2017.

[113] Everett M Rogers. *Diffusion of innovations*. Simon and Schuster, 2010.

[114] John Rushby. The interpretation and evaluation of assurance cases. *SRI International, Menlo Park, CA, USA*, 2015.

[115] OMG SACM. Structured assurance case meta-model 2.0 beta. http://www.omg.org/spec/SACM/2.0/Beta/, 2017. last accessed 2017.

[116] Jason J Saleem, Alissa L Russ, Connie F Justice, Heather Hagg, Patricia R Ebright, Peter A Woodbridge, and Bradley N Doebbeling. Exploring the persistence of paper with the electronic health record. *International journal of medical informatics*, 78(9):618–628, 2009.

[117] Jason J Saleem, Alissa L Russ, Adam Neddo, Paul T Blades, Bradley N Doebbeling, and Brian H Foresman. Paper persistence, workarounds, and communication breakdowns in computerized consultation management. *International journal of medical informatics*, 80(7):466–479, 2011.

[118] Noah Schoenberg, Emily Fondahn, and Michael Lane. Introduction to patient safety, 2016.

[119] Nada Atef Shebl, Bryony Dean Franklin, and Nick Barber. Is failure mode and effect analysis reliable? *Journal of patient safety*, 5(2):86–94, 2009.

[120] Nada Atef Shebl, Bryony Dean Franklin, and Nick Barber. Failure mode and effects analysis outputs: are they valid? *BMC health services research*, 12(1):150, 2012.

[121] GA Shirali, V Ebrahipour, et al. Proactive risk assessment to identify emergent risks using functional resonance analysis method (fram): a case study in an oil process unit. *Iran Occupational Health*, 10(6):33–46, 2013.

[122] Jeffrey Shuren. Testimony of jeffrey shuren, director of fda's center for devices and radiological health to the health information technology policy committee. `https://connex.csc.uvic.ca/access/content/group/Simbioses/public/Shuren2010Testimony.pdf`, 2010. last accessed 2016.

[123] Jeffrey Shuren. Continuing america's leadership: the future of medical innovation for patients. `http://www.fda.gov/NewsEvents/Testimony/ucm445064.htm`, 2015. last accessed 2016.

[124] Dean F Sittig, Kanav Kahol, and Hardeep Singh. Sociotechnical evaluation of the safety and effectiveness of point-of-care mobile computing devices: a case study conducted in india. *Electronic Health Records: Challenges in Design and Implementation*, page 115, 2013.

[125] Dean F Sittig and Hardeep Singh. A new sociotechnical model for studying health information technology in complex adaptive healthcare systems. *Qual. Saf. Health Care*, 19 Suppl 3(Suppl 3):i68–74, 2010.

[126] Dean F Sittig and Hardeep Singh. A red-flag-based approach to risk management of ehr-related safety concerns. *Journal of Healthcare Risk Management*, 33(2):21–26, 2013.

[127] Dean F Sittig, Hardeep Singh, Shailaja Menon, Ashley ND Meyer, and Elisabeth Belmont. The context of ehr safety and the need for risk assessment. In *SAFER Electronic Health Records: Safety Assurance Factors for EHR Resilience*, pages 1–32. Apple Academic Press, 2015.

[128] Patrice L Spath. Using failure mode and effects analysis to improve patient safety. *AORN journal*, 78(1):15–37, 2003.

[129] John Spriggs. *GSN-The Goal Structuring Notation: A Structured Approach to Presenting Arguments*. Springer Science & Business Media, 2012.

[130] Dean H Stamatis. *Failure mode and effect analysis: FMEA from theory to execution*. Asq Press, 2003.

[131] DH Stamatis. *Introduction to risk and failures: Tools and methodologies*. CRC Press, 2014.

[132] Anselm Strauss and Juliet Corbin. Grounded theory methodology. *Handbook of qualitative research*, 17:273–85, 1994.

[133] John Thomas, F Lemos, and Nancy Leveson. Evaluating the safety of digital instrumentation and control systems in nuclear power plants. *NRC Technical Researcy Report2013*, 2012.

[134] UBC Health. Roles of health care professionals. `http://www.health.ubc.ca/roles-and-responsibilities/`. last accessed 2017.

[135] Peter Underwood and Patrick Waterson. A critical review of the stamp, fram and accimap systemic accident analysis models. *Advances in Human Aspects of Road and Rail Transportation. CRC Press, Boca Raton*, pages 385–394, 2012.

[136] Peter Underwood and Patrick Waterson. Systems thinking, the Swiss Cheese Model and accident analysis: A comparative systemic analysis of the Grayrigg train derailment using the ATSB, AcciMap and STAMP models. *Accid. Anal. Prev.*, 68:75–94, 2014.

[137] JM Walker, A Hassol, B Bradshaw, and M Rezaee. Health it hazard manager beta-test: final report, 2012.

[138] D Warm and P Edwards. Classifying health information technology patient safety related incidents - an approach used in Wales. *Appl. Clin. Inform.*, 3(2):248–257, 2012.

[139] Alan Wassyng, Tom Maibaum, and Mark Lawford. On Software Certification : We Need Product-Focused Approaches. *Monterey Work.*, pages 250–274, 2010.

[140] Jens H Weber, Fieran Mason-Blakley, and Morgan Price. Information system hazard analysis: A method for identifying technology-induced latent errors for safety. In *ITCH*, pages 342–346, 2015.

[141] Jens H Weber-Jahnke. A preliminary study of apparent causes and outcomes of reported failures with patient management software. In *Proceedings of the 3rd Workshop on Software Engineering in Health Care*, pages 5–8. ACM, 2011.

[142] Jens H Weber-Jahnke and Fieran Mason-Blakley. The safety of electronic medical record (emr) systems: what does emr safety mean and how can we engineer safer systems? *ACM SIGHIT Record*, 1(2):13–22, 2011.

[143] Jens H Weber-Jahnke and Fieran Mason-Blakley. On the safety of electronic medical records. In *Foundations of Health Informatics Engineering and Systems*, pages 177–194. Springer, 2012.

[144] Charles B Weinstock, John B Goodenough, and Ari Z Klein. Measuring Assurance Case Confidence using Baconian Probabilities, 2013.

[145] Tosha B Wetterneck, Kathleen Skibinski, Mark Schroeder, Tanita L Roberts, and Pascale Carayon. Challenges with the performance of failure mode and effects analysis in healthcare organizations: an iv medication administration hfmea. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 48:15, pages 1708–1712. SAGE Publications Sage CA: Los Angeles, CA, 2004.

[146] Wikipedia. Inter-process communication. `https://en.wikipedia.org/wiki/Inter-process_communication`. last accessed 2017.

[147] Rogier Woltjer, Erik Prytz, and Kip Smith. Functional modeling of agile command and control. In *14th International Command and Control Research and Technology Symposium, Washington DC, USA June 15-17, 2009*. DOD CCRP, 2009.

[148] Janet Woodcock and Jeffrey Shuren. Reviewing the FDA's implementation of FDA-SIA. `http://www.fda.gov/NewsEvents/Testimony/ucm374544.htm`, 2013. last accessed 2016.

[149] Yi Zhang, Paul L Jones, and Raoul Jetley. A hazard analysis for a generic insulin infusion pump. *Journal of diabetes science and technology*, 4(2):263–283, 2010.

Appendix A

Hazard Checklist

This hazard checklist is synthesized from those provided by Ericson [28] and from concepts expressed in the STAMP based STPA prospective hazard analysis method.

- physical issues with real world interfacing
  - maintenance error
  - "hard" shutdowns/failures
  - interference (EMI/RFI)
  - "sneak" software
- Human Factors
  - noise
  - operator error
  - inadvertent operation
  - failure to operate
  - operation early/late
  - operation out of sequence
  - right operation/wrong control
  - operated too long/briefly
  - fatigue
  - inaccessibility
  - nonexistent/inadequate "kill" switches
  - inadequate control/readout differentiation

  - faulty/inadequate control/readout labelling
  - faulty workstation design
  - inadequate/improper illumination
  - glare

- Failure States
  - fails to operate
  - operates incorrectly/erroneously
  - operates inadvertently
  - operates at incorrect time
  - unable to stop operation
  - receives erroneous data
  - sends erroneous data

Appendix B

Preliminary Hazard List

Computerized Provider Order Entry System Running Example

Hazard H1 Antiglycemic [task parameter] medication doses [task object] that are prescribed [task action] by a doctor [event agent] are not delivered [error condition] to patients. This results in physical harm to patients which could have prevented and thus constitutes a medical error [Precipitating Event]. This occurs because the CIS software fails to transmit [error condition] the medication orders entered by doctors [event agent] from the medication prescription module [error context] through the system [error context] to the medication administration module [error context] where the order [task object] is observed and acted on by the administering nurse [event agent].

Hazard H3 Antiglycemic [task parameter] medication orders [task object] which are entered [event task] by doctors [event agent] into the medication prescription module [error context] near the end of the day [event context] and use ambiguous relative temporal terms like *tomorrow* [error element] have the potential to be delayed by 24hrs [error element] due to misinterpretation by the administering nurse [event agent] who received [event action] the prescription in the medication administration module [error context] on account of the low *clarity* [error condition] of these expressions [71]

Hazard H4 Antiglycemic [task parameter] medication orders [task object] entered [task action] in one work station [event context] may not be persisted [error condition] in such a way that they propagate to other workstations [event context] [32] resulting in an instruction set that is lacking in *consistency* across the views [event context] provided of the prescription record [task object] available throughout the system

Hazard H2 Antiglycemic [task parameter] medication over doses occur, causing patient

harm, because physicians [event agent] suffer alert fatigue [error condition] and ignore relevant CDS [error context] warnings [error element] on account of the volume [error condition] of unnecessary warnings they regularly dismiss.

Hazard H5 Doctors [event agent] prescribing [task action] antiglycemic [task parameter] medications [task object] are presented by the CDS module [error context] with a ratio of significant alerts to alerts of marginal importance [error element] that is too low [error condition] [92, 85]. This condition highlights compromised *precision* in the displayed data.

Hazard H6 Doctors [event agent] prescribing [task action] anyticlycemic [task parameter] medications [task object] are presented by the CDS module [error context] with a volume of information in alerts is too large [error condition] compromising the *memorability* of the system by making it difficult for the doctors [event agent] to extract the important details from the less relevant context in the alerts [error context] [92, 85]

Hazard H7 The system [error element] may be not be *available* due to a ransomware attack [error condition] which infects it during a period when its supporting platform [task object][error context] is not updated [task action] with the most recent [task parameter] software patches [error condition][task object] by the system administrators [event agent].

Hazard H9 The system security modules [event agent] do not sufficiently [error condition] monitor [task action] the process interactions [error element] with the Master File Table [error element] of NTFS based [error condition] platforms [65] (RID).

Hazard H10 Base operating system platforms are not updated with sufficient frequency [112] (STPA - *A control action required for safety is not provided or not followed* [76]).

Hazard H8 A doctor [task agent] can not quickly [task parameter] revert/cancel [task action] an erroneous antiglycemic agent [task parameter] prescription [task object] when using the prescription module [error context].

Hazard H11 A doctor [task agent] can not quickly [task parameter] revert/cancel [task action] an erroneous antiglycemic agent [task parameter] prescription [task object] when using the prescription module [error context] because of the poor *clarity* of the medication management module's [error context] interface [71].

Appendix C

Risk Tables

The scales for the detectability (D), severity (S) and occurrence (O) of the RAC are tabulated in Tabs. C.1, C.3, C.2. The associated risk levels are drawn from the 882 standard [26] and are shown in Tab. C.4. Associating the triplet of inputs to the risk, we develop a three dimensional matrix which enumerates the possible risk triplets. Each of these is then assigned risk level. The matrix chosen for this analysis is tabulated in Tabs. C.5, C.6, , C.7, and C.8. Each presents a slice of the risk matrix across the value of the severity score.

D	Description
1	Almost certain to be detected and corrected
2	High likelihood of detection and correction
3	Moderate likelihood of detection and correction
4	Low likelihood of detection and correction
5	Remote likelihood of detection and correction

Table C.1: Ordinal scale of detectability - adopted from Spath [128]

O	Title	Individual Item
1	Improbable	"So unlikely, it can be assumed occurrence may not be experienced in the life of an item."
2	Remote	"Unlikely, but possible to occur in the life of an item."
3	Occasional	"Likely to occur sometime in the life of an item."
4	Probable	"Will occur several times in the life of an item."
5	Frequent	"Likely to occur often in the life of an item."

Table C.2: Ordinal scale of occurrence - adapted from 882 [26].

S	Description	Definition
1	Negligible	"Could result in one or more of the following: injury or occupational illness not resulting in a lost work day, minimal environmental impact, or monetary loss less than $100K."
2	Marginal	"Could result in one or more of the following: injury or occupational illness resulting in one or more lost work day(s), reversible moderate environmental impact, or monetary loss equal to or exceeding $100K but less than $1M."
3	Critical	"Could result in one or more of the following: permanent partial disability,injuries or occupational illness that may result in hospitalization of at least three personnel, reversible significant environmental impact, or monetary loss equal to or exceeding $1M but less than $10M."
4	Catastrophic	"Could result in one or more of the following: death, permanent total disability, irreversible significant environmental impact, or monetary loss equal to or exceeding $10M."

Table C.3: Ordinal scale of severity - replicated from 882 [26].

Risk Level	Description of Risk Criteria
	A software implementation or software design defect that upon occurring during normal or credible off-nominal operations or tests:
High	• Can lead directly to a catastrophic or critical mishap, or • Places the system in a condition where no independent functioning interlocks preclude the potential occurrence of a catastrophic or critical mishap.
Serious	• Can lead directly to a marginal or negligible mishap, or • Places the system in a condition where only one independent functioning interlock or human action remains to preclude the potential occurrence of a catastrophic or critical hazard.
Medium	• Influences a marginal or negligible mishap, reducing the system to a single point of failure, or • Places the system in a condition where two independent functioning interlocks or human actions remain to preclude the potential occurrence of a catastrophic or critical hazard.
Low	• Influences a catastrophic or critical mishap, but where three independent functioning interlocks or human actions remain, or • Would be a causal factor for a marginal or negligible mishap, but two independent functioning interlocks or human actions remain. • A software degradation of a safety critical function that is not categorized as high, serious, or medium safety risk. • A requirement that, if implemented, would negatively impact safety; however code is implemented safely.

Table C.4: Ordinal scale of risk - replicated from 882 [26]

		Occurrence				
		Improbable	Remote	Occasional	Probable	Frequent
Detectability	Almost certain	Low	Low	Low	Low	Medium
	High likelihood	Low	Low	Low	Low	Medium
	Moderate likelihood	Low	Low	Low	Low	Medium
	Remote likelihood	Medium	Medium	Medium	Medium	Serious

Table C.5: The risk table for negligible severity hazards.

		Occurrence				
		Improbable	Remote	Occasional	Probable	Frequent
Detectability	Almost certain	Low	Low	Low	Medium	Medium
	High likelihood	Low	Low	Low	Medium	Medium
	Moderate likelihood	Medium	Medium	Medium	Serious	Serious
	Remote likelihood	Medium	Medium	Medium	Serious	Serious

Table C.6: The risk table for marginal severity hazards.

		Occurrence				
		Improbable	Remote	Occasional	Probable	Frequent
Detectability	Almost certain	Low	Low	Medium	Medium	Serious
	High likelihood	Low	Low	Medium	Serious	Serious
	Moderate likelihood	Medium	Medium	Serious	High	High
	Remote likelihood	Medium	Serious	Serious	High	High

Table C.7: The risk table for critical severity hazards.

		Occurrence				
		Improbable	Remote	Occasional	Probable	Frequent
Detectability	Almost certain	Medium	Medium	Serious	High	High
	High likelihood	Medium	Medium	Serious	High	High
	Moderate likelihood	Serious	Serious	High	High	High
	Remote likelihood	High	High	High	High	High

Table C.8: The risk table for catastrophic severity hazards.

Appendix D

Generalized Insulin Infusion Pump Preliminary Hazard List

Zhang [149] identified a list of contributing factors to hazards posed by a GIIP. Below, we take the additional step of associating Zhang's contributing factors with the hazards that Zhang identified. These hazards are coarsely categorized as unplanned release of energy, failure to release energy, unplanned release of matter and failure to release matter. Further, we identify overdose, underdose, and incorrect treatment as medication related hazards which fall into the categories of unplanned release, and failure to release matter. We present these in Tab. D.1. Futher to Zhang's offerings, we provide a coordination of contributing factors and hazards for to the GIIP in Tab. D.2.

Category	Hazardous Situation
Therapeutic	Overdose: the user receives more insulin than required to maintain desirable BG levels Underdose: the user receives less insulin than required to maintain desirable BG levels Incorrect treatment: the user receives either an incorrect drug or a correct drug with incorrect concentration
Energetic	Excessive thermal energy generation by the pump Electrical shock: the pump transfers electric current to accessible surfaces during operation Excessive electromagnetic emissions by the pump: affects the pump itself, other device(s) worn by the user, or other users and their devices Excessive sound frequencies generated by the pump
Chemical/biological	User infection User allergic reaction/rash to pump materials or insulin[a]
Mechanical	Presence of sharp edges or scissor points Excessive pump vibration, e.g., connectors, components stressed
Environmental	Unsafe disposal of the pump or pump components: user disposes batteries or other pump subassemblies in an unsafe manner
	[a]The user may also be allergic to infusion set adhesives. However, because such adhesives have been excluded from the GIIP system, we do not consider hazardous situations related to infusion set adhesives here.

Table D.1: Zhang's description of hazardous situations [149]

Contributing Factor	H														
	h_1: Unplanned Release of Energy						**h_2: Failed Release of Energy**	**h_3: Unplanned Release of Matter / Unintended Consequence of Matter Release**						**h_4: Failed Release of Matter**	
	Excessive Thermal Energy Generation by the Pump	Excessive Electromagnetic Emissions from the Pump	Electric Shock	Excessive Sound Frequencies Generated by Pump	Collision with Sharp Edges	Excessive Pump Vibration		Overdose	Incorrect Treatment	User Infection	User Allergic Reaction to Pump Materials or Insulin	Unsafe Disposal of Pump or Pump Materials	Contamination of Drug	Underdose	Incorrect Treatment
	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
Operational Sources of Hazardous Situations															
cf_1: Air in Line														✓	
cf_{1a}: Incorrect/incomplete priming process														✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{1b}: User's motions cause the delivery path to be loose or broken														✓	
cf_{1c}: Broken, loose, or unsealed delivery path														✓	
cf_{1d}: Pump or pump components are unable to release gas or air														✓	
cf_{1e}: Cold insulin is loaded and then warms up to form air bubbles														✓	
cf_{1f}: Pump connected with incompatible infusion sets														✓	
cf_2: Free flow								✓							
cf_{2a}: Valve in the delivery path is broken								✓							
cf_{2b}: Air pressure within the pump is much lower/higher than ambient air pressure								✓							
cf_{2c}: Pump is positioned much higher than the infusion site, causing unintentional drug flow								✓							
cf_{2d}: Delivery path is damaged, creating a vent on the path that allows unintentional gravity flow								✓							
cf_{2e}: Large temperature changes causing a mismatch between drug reservoir volume change and insulin density change								✓							
cf_3: Reverse flow														✓	
cf_{3a}: Siphon effect due to the pump being positioned much lower than the infusion site														✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{3b}: Delivery path is damaged, creating a vent on the path that diverts an intentional drug flow from reaching the user Pump														✓	
cf_{3c}: Pump delivery mechanism runs opposite to the expected direction														✓	
cf_{3d}: Air pressure within the pump is much lower/higher than ambient air pressure														✓	
cf_4: Pump is disconnected from the infusion set without the users awareness														✓	
cf_{4a}: Users motions cause the pump or the attached infusion set to be disconnected from the user														✓	
cf_{4b}: Loose connection between parts of the delivery path														✓	
cf_{4c}: Infusion set is not applied to the user correctly or is caught on the infusion set adhesives or tapes, causing it to be disconnected from the user														✓	
cf_5: Excessive bolus administration due to too many bolus requests from the user								✓							
cf_{5a}: Bolus history is corrupted, making the user unable to track previous boluses								✓							
cf_{5b}: User forgets about previously received boluses and requests additional unnecessary boluses without consulting the bolus history records								✓							

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{5c}: User requests a meal bolus but does not eat								✓							
cf_6: Occlusion without the users awareness								✓							
cf_{6a}: Delivery path obstruction, e.g., kinked tubes								✓							
cf_{6b}: Chemical precipitation inside the delivery path								✓							
cf_7: Dosage of bolus is delivered unevenly over its specified duration								✓						✓	
cf_{7a}: Algorithmic errors								✓						✓	
cf_{7b}: Pump delivery mechanism does not operate as instructed								✓						✓	
cf_8: Insulin Leakage										✓	✓			✓	
cf_{8a}: User does not follow instructions to disconnect the pump appropriately										✓	✓			✓	
cf_{8b}: Pump is disconnected without user's awareness										✓	✓			✓	
cf_{8c}: Loose connection between parts of the delivery path										✓	✓			✓	
cf_{8d}:Broken drug reservoir										✓	✓			✓	
cf_{8e}:Occlusion during insulin delivery causes high pressure within the delivery path										✓	✓			✓	
cf_9: Drug reservoir becomes empty during insulin delivery without the users awareness								✓							

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{10}: Insulin level in the drug reservoir becomes low during insulin delivery without the users awareness								✓							
cf_{11}: Actual flow rate does not match the programmed infusion rate								✓						✓	
cf_{11a}:Occlusion without the users awareness								✓						✓	
cf_{11b}:Air pressure within the pump is much lower/higher than ambient air pressure								✓						✓	
cf_{11c}:Outside temperature is out of safe range or fluctuating inadvertently, causing the pump to deliver insulin inaccurately or behave erratically								✓						✓	
cf_{11d}:Outside air pressure is out of safe range or fluctuating inadvertently, causing the pump to deliver insulin inaccurately or behave erratically								✓						✓	
cf_{11e}:Electromagnetic interference due to internal or external electro- magnetic disturbances, causing the pump to deliver insulin inaccurately or behave erratically								✓						✓	
cf_{12}: Excessive flow rate fluctuation								✓						✓	
cf_{13}: A replaceable drug reservoir is detached during normal pump use Unexpected														✓	
cf_{13a}:Drug reservoir compartment is broken or opened														✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{13b}:Users motions cause the reservoir to be disconnected														✓	
cf_{14}: Unexpected delivery of insulin								✓							
cf_{14a}:Software instructs to resume a previous bolus after suspension, or after the battery is replaced, causing an unexpected bolus								✓							
cf_{14b}:Software instructs pump to finish paused basal delivery after a long suspension/interruption, causing a huge bolus to be flushed to the user								✓							
cf_{14c}:User is connected to the pump while it is being refilled or primed								✓							
cf_{14d}:User is connected to the pump while freeing clogged infusion tubes or detaching the reservoir								✓							
cf_{14e}:Releasing occlusion causes unexpected boluses								✓							
cf_{14f}:Large temperature changes causing a mismatch between the drug reservoir volume change and the insulin density change								✓							
cf_{14g}:Pump fails to shut off or stop insulin delivery as commanded, and the user is not aware of this								✓							
cf_{15}: Pump stops delivering insulin without the users awareness														✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{15a}:Pump suspends or stops without the users awareness														✓	
cf_{15b}:Drug reservoir is loaded improperly, causing no insulin to be delivered														✓	
cf_{16}: Pump hardware is not initialized properly								✓						✓	✓
cf_{16a}:Pump platform fails to meet default operational specifications														✓	
Software Sources of Hazardous Situations															
cf_{17}: Incorrect meal bolus is recommended by the bolus calculator								✓						✓	
cf_{17a}:User estimates or enters carbohydrate content of a planned meal incorrectly								✓						✓	
cf_{17b}:Food database contains erroneous information, causing incorrect calculation of the number of carbohydrates in a meal								✓						✓	
cf_{17c}:User determines or enters carbohydrate ratios (food factors) incorrectly								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{17d}:User misunderstands reverse correction or does not use reverse correction upon appropriate conditions Reverse correction refers to an optional feature that automatically adjusts meal bolus recommendations when the user encounters low BG levels. In particular, if this option is chosen, then calculation of a meal bolus dose, when the users current BG level is below the target BG level, should reduce the amount of insulin necessary to bring the BG level back to target. This contributing factor is applicable only if the pump supports reverse correction.								✓						✓	
cf_{17e}:Design flaws/implementation defects in the bolus calculator								✓						✓	
cf_{17f}:Unexpected software execution								✓						✓	
cf_{18}: Incorrect meal bolus is recommended by the bolus calculator								✓						✓	
cf_{18a}:Pump provides the user only limited flexibility, such as coarse increment steps, to input parameters critical to bolus calculation								✓						✓	
cf_{18b}:Inappropriate or incorrect calculation of insulin on board (IOB)								✓						✓	
cf_{18c}:User estimated or entered his/her sensitivity to insulin over time (correction factors) incorrectly								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{18d}:Calculator uses obsolete BG readings as the users current BG level to calculate correction bolus								✓						✓	
cf_{18e}:User measured or entered BG values incorrectly								✓						✓	
cf_{18f}:User estimated or entered target BG levels incorrectly								✓						✓	
cf_{18g}:User estimated or entered target BG levels incorrectly								✓						✓	
cf_{18h}:Unexpected software execution								✓						✓	
cf_{19}: Incorrect correction bolus is recommended by the bolus calculator								✓						✓	
cf_{19a}:Pump provides the user only limited flexibility, such as coarse increment steps, to input parameters critical to bolus calculation								✓						✓	
cf_{20}: Incorrect correction bolus is recommended by the bolus calculator								✓						✓	
cf_{20a}:Pump provides the user only limited flexibility, such as coarse increment steps, to input parameters critical to bolus calculation								✓						✓	
cf_{20b}:Inappropriate or incorrect calculation of insulin on board (IOB)								✓						✓	
cf_{20c}:User estimated or entered his/her sensitivity to insulin over time (correction factors) incorrectly								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{20d}:Calculator uses obsolete BG readings as the users current BG level to calculate correction bolus								✓						✓	
cf_{20e}:User measured or entered BG values incorrectly								✓						✓	
cf_{20f}:User estimated or entered target BG levels incorrectly								✓						✓	
cf_{20g}:Design flaws/implementation defects in the bolus calculator								✓						✓	
cf_{20h}:Unexpected software execution								✓						✓	
cf_{21}: Inappropriate or incorrect calculation of IOB								✓						✓	
cf_{21a}:Pump provides the user limited or no capability to configure the duration of insulin action (DIA), causing input DIA to not be accurate enough for correct IOB calculation User								✓						✓	
cf_{21b}:User injects insulin manually, which is not accounted for by the pump when calculating IOB								✓						✓	
cf_{21c}:Insulin leakage, causing the pump to miscalculate the amount of IOB								✓						✓	
cf_{21d}:Records of previous boluses become corrupt due to memory corruption or become inconsistent due to pump time changes								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{21e}:Design flaws/implementation defects in the bolus calculator								✓						✓	
cf_{21f}:Unexpected software execution								✓						✓	
cf_{22}: Pump unexpectedly restores to default factory settings without the users awareness								✓						✓	
cf_{22a}:User inadvertently selects a restore of the factory settings Accumulated								✓						✓	
cf_{22b}:Accumulated static electricity during use triggers an unexpected restore of default factory settings								✓						✓	
cf_{22c}:Battery is inadvertently disconnected from the pump								✓						✓	
cf_{23}: Pump controller fails to monitor the status of the pump delivery mechanism								✓						✓	
cf_{24}: Pump controller fails to detect mechanism failures of pump delivery								✓						✓	
cf_{25}: Pump annunciates notifications of different importance to the user with similar signals								✓						✓	
cf_{26}: Pump presents inappropriate or inaccurate prompts to the user								✓						✓	✓

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{27}: Incorrect critical data. Data critical to insulin delivery include correction factors, food factor, basal infusion profiles, programmed bolus deliveries, records of previous insulin deliveries, BG logs, and target BG levels, as well as information about loaded insulin and food database (if applicable)								✓						✓	✓
cf_{27a}:Data tampered with by unauthorized personnel								✓						✓	
cf_{27b}:Data corrupted due to memory corruption								✓						✓	
cf_{27c}:User provides the pump with incorrect, inaccurate, or incomplete information								✓						✓	
cf_{27d}:Pump does not record insulin delivered to the user during the period the user chooses to disconnect the pump and actual disconnection								✓						✓	
cf_{27e}:Insulin leakage, resulting in incorrect records of previous insulin deliveries								✓						✓	
cf_{28}: Corrupted infusion commands								✓						✓	
cf_{28a}:Data tampered with by unauthorized personnel								✓						✓	
cf_{28b}:Random-access memory or nonvolatile memory failure, including failing to write to memory, failing to read from memory, and memory corruptions								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{28c}:Watchdog error								✓						✓	
cf_{28d}:Software defects, e.g., stack overflow, pointer corruption, math overflow, race conditions Pump								✓						✓	
cf_{29}: Incorrect or inappropriate basal profiles are programmed/activated								✓						✓	✓
cf_{29a}:Pump only provides limited options for the user to configure correction factors								✓						✓	✓
cf_{29b}:Pump provides limited or no flexibility for the user to program basal delivery profiles to compensate for different behavior patterns								✓						✓	✓
cf_{29c}:Pump does not display necessary details about basal profiles on the user interface, e.g., time of latest modification, causing the user to activate an inappropriate basal profile								✓						✓	✓
cf_{30}: Unexpected software execution								✓						✓	
cf_{30a}:Software update error or failure Software								✓						✓	
cf_{30b}:Software defects, e.g., stack overflow, pointer corruption, math overflow, race conditions								✓						✓	
cf_{30c}:Operating systems and/or runtime supports corrupted, failed, or updated								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{30d}:Hardware failure, e.g., central processing unit (CPU), memory, input/output (I/O), bus, power glitch, radiation/electromagnetic interference (EMI)								✓						✓	
cf_{31}: Data logging/retrieval failure								✓						✓	
cf_{32}: Inappropriate setting of alarm priorities								✓						✓	
cf_{33}: Pump fails to auto-stop upon detecting a critical failing condition that requires it to stop								✓						✓	
cf_{34}: Inadequate or overcomplicated operating instructions								✓						✓	✓
cf_{35}: Software not initialized to appropriate values								✓						✓	✓
cf_{35a}:During pump startup, reset, power-off/power-on sequence software is not initialized to appropriate values								✓						✓	
cf_{36}: Nuisance alarming								✓						✓	✓
cf_{36a}:Inappropriate setting of alarm priorities								✓						✓	
cf_{36b}:Sensor failures								✓						✓	
cf_{37}: Pump unexpectedly resets to default pump settings without the users awareness								✓						✓	
cf_{37a}:User inadvertently selects a device reset								✓						✓	
cf_{37b}:Accumulated static electricity during use triggers an unexpected reset of the device								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{37c}:Battery is inadvertently disconnected from the pump, triggering a reset								✓						✓	✓
cf_{37d}:Hardware failure, e.g., CPU, memory, I/O, bus, power glitch, radiation/EMI								✓						✓	✓
Hardware Sources of Hazardous Situations															
cf_{38}: Central processing unit failure								✓						✓	
cf_{39}: Random-access memory or nonvolatile memory failure, including failing to write to memory, failing to read from memory, and memory data corruptions								✓						✓	
cf_{40}: Read-only memory or external flash memory failure								✓						✓	
cf_{41}: Pump delivery mechanism does not operate as instructed								✓						✓	
cf_{42}: Pump delivery mechanism fails and does not stroke														✓	
cf_{43}: Fail to stop the motor of the pump when a fault condition occurs								✓						✓	
cf_{44}: Fatigued/worn/broken mechanical parts	✓					✓		✓						✓	
cf_{45}: User interface components of the pump, including display units and alarming units, fail or behave abnormally								✓						✓	✓
cf_{46}: Input device, e.g., keypad or touch screen, does not work correctly								✓						✓	✓

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{47}: Key bounce not detected or correctly								✓						✓	✓
cf_{48}: Audio notifications or prompts cannot be heard in a normal use environment								✓						✓	✓
cf_{48a}:Defective audio device(s) Incorrect								✓						✓	✓
cf_{48b}:Incorrect audio volume settings								✓						✓	✓
cf_{49}: Audio notifications or prompts too loud				✓											
cf_{49a}:Abnormal audio device(s)				✓											
cf_{49b}:Incorrect audio volume settings				✓											
cf_{50}: Delayed alarm detection and notifications								✓						✓	✓
cf_{50a}:Abnormal audio device(s)								✓						✓	✓
cf_{50b}:Incorrect audio volume settings								✓						✓	✓
cf_{51}: Nonaudio alarm cannot be seen/interpreted								✓						✓	✓
cf_{51a}:Light-emitting diode failure								✓						✓	✓
cf_{51b}:Color blindness								✓						✓	✓
cf_{51c}:Poor location								✓						✓	✓
cf_{52}: Nonaudio alarm cannot be felt (vibration)								✓						✓	✓
cf_{52a}:Vibration mechanism fails								✓						✓	✓
cf_{52b}:Incorrect vibration setting								✓						✓	✓
cf_{52c}:Pump location								✓						✓	✓
cf_{53}: Inadequate electrical/radiation shielding for the pump		✓	✓												

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{54}: Improper shape design or improper manufacturing process			✓		✓										
cf_{55}: Sensor Failure								✓						✓	✓
cf_{56}: False watchdog interrupt								✓						✓	✓
cf_{57}: Watchdog timer failed; watchdog does not interrupt as expected								✓						✓	✓
cf_{58}: Time base, such as real-time clock (RTC), used by the pump to control insulin delivery speeds up, slows down, or stalls								✓						✓	
cf_{59}: System RTC not synchronized (date/time register not the same as the RTC)								✓						✓	
cf_{60}: Synchronization error between pump components								✓						✓	
cf_{61}: Component communication/bus/channel failure								✓						✓	
cf_{62}: Broken drug reservoir								✓						✓	
Physical Sources of Hazardous Situations															
cf_{63}: Physical damage to the pump or its sub-assemblies		✓			✓			✓						✓	
cf_{63a}:User drops pump accidentally		✓			✓			✓						✓	
cf_{63b}:Pump is sheared due to contact with surrounding surfaces or objects		✓			✓			✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{63c}:Excessive external stress is applied to the pump		✓			✓			✓						✓	
cf_{64}: Fluid/humidity ingress into the pump	✓		✓					✓						✓	
cf_{65}: Air pressure within the pump is much lower/higher than ambient air pressure								✓						✓	
cf_{66}: Pump overheats while running								✓						✓	
cf_{66a}:Mechanical failures								✓						✓	
cf_{66b}:Electrical failures								✓						✓	
Electrical Sources of Hazardous Situations															
cf_{67}: Nonfunctioning/disabled electrical circuits/ components, e.g., shorted electrical circuits.	✓		✓					✓						✓	
cf_{67a}:Electrical circuit/component failures	✓		✓					✓						✓	
cf_{67b}:Fluid/humidity ingress	✓		✓					✓						✓	
cf_{68}: Erratic electric circuit operations		✓						✓						✓	
cf_{68a}:Nonfunctioning/disabled electrical circuits/components, e.g., shorted electrical circuits		✓						✓						✓	
cf_{68b}:Pump develops excessive static charge or experiences electro- static discharge (ESD) that exceeds its ESD immunity		✓						✓						✓	
cf_{68c}:Fluid/humidity ingress into the pump		✓						✓						✓	
cf_{68d}:Voltage level of the battery is too low Voltage		✓						✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{68e}:Voltage level of the battery varies greatly		✓						✓						✓	
cf_{68f}:Battery impedance or contact impedance becomes too high		✓						✓						✓	
cf_{68g}:Electromagnetic interference		✓						✓						✓	
cf_{69}: Pump develops excessive static charge or experiences ESD that exceeds its ESD immunity								✓						✓	
cf_{69a}:Pump is rubbing against surrounding surfaces or articles								✓						✓	
cf_{70}: Leakage current on the surface of the pump			✓												
cf_{71}: Battery depletes without the users awareness														✓	
cf_{72}: Battery depletes rapidly, giving the user insufficient time to respond														✓	
cf_{73}: Voltage level of the battery is too low								✓						✓	
cf_{74}: Voltage level of the battery varies greatly								✓						✓	
cf_{75}: Battery life is unpredictable								✓						✓	
cf_{76}: Battery is inadvertently disconnected from the pump														✓	
cf_{76a}:Broken battery compartment or broken battery compartment cap														✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{76b}:User drops the pump accidentally, disconnecting the battery temporarily														✓	
cf_{77}: Battery impedance or contact impedance becomes too high														✓	
cf_{78}: Depleted batteries are discarded without being recycled												✓			
Biological and Chemical Sources of Hazardous Situations															
cf_{79}: Pump, especially its delivery path, is contaminated with toxic substances											✓				
cf_{79a}:Inadequate pump cleaning/sterilization (e.g., residue after contamination, failure to flush, failure to disinfect)											✓				
cf_{79b}:Battery fluid or other fluid leaks into the delivery path											✓				
cf_{79c}:User uses inappropriate cleaning agents while cleaning the pump routinely											✓				
cf_{79d}:User keeps using the pump for a period longer than recommended											✓				
cf_{80}: Pump is exposed to pathogens, allergens, and other infectious substances										✓					
cf_{80a}:Pump is shared by multiple users										✓					
cf_{80b}:Packaging of the pump is damaged prior to its use, but the user applies the pump regardless										✓					

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{80c}:Inadequate pump cleaning/sterilization, such as residue after contamination, failure to flush, and failure to disinfect, causing the pump to lose its sterility										✓					
cf_{80d}:Pump is connected to non-sterile infusion sets										✓					
cf_{81}: Chemical precipitation inside the delivery path											✓			✓	
cf_{81a}: Incorrect/incomplete pump cleaning procedure											✓			✓	
cf_{82}: Infusion site infection										✓					
cf_{82a}: User fails to clean the infusion site completely before applying the infusion set										✓					
cf_{82b}: User fails to change infusion sites as recommended										✓					
cf_{83}: Insulin, while being delivered to the user, loses its potency															✓
cf_{83a}: Insulin contacts with incompatible pump material															✓
cf_{83b}: Environmental temperature is too high/low															✓
cf_{84}: Pump is made of materials that cause user allergic reactions											✓				
Use Sources of Hazardous Situations															

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{85}: User uses the pump when certain physical/mental conditions such as impaired vision prevent him/her to do so								✓						✓	✓
cf_{86}: User is incapable of using the pump or configuring treatment plans								✓						✓	✓
cf_{86a}: User is not sufficiently trained to operate the pump; user is not sufficiently intelligent to understand the instructions and use the pump correctly								✓						✓	✓
cf_{86b}: User falls asleep or goes into coma due to hypoglycemia								✓						✓	✓
cf_{87}: User injects long-acting insulin shortly before first use of the pump, causing an amount of insulin on board that cannot be accounted for by the pump								✓							
cf_{88}: User is connected to the pump incorrectly								✓						✓	
cf_{89}: User fills the pump with wrong types of insulin, degraded insulin, or drugs other than insulin															✓
cf_{90}: User fails to test his BG levels as frequently as recommended								✓						✓	✓
cf_{91}: User travels to a different time zone and forgets to accommodate so-caused time discrepancy while using the pump								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{92}: User fails to replace consumed pump supplies, including insulin and batteries, in time														✓	
cf_{92a}: User fails to attend to pump notifications														✓	
cf_{92b}: User has no access to backup pump supplies														✓	
cf_{93}: User inputs incorrect drug type and concentration information for currently loaded insulin															✓
cf_{94}: User enters incorrect parameters when configuring basal profiles								✓						✓	
cf_{95}: User enters incorrect parameters when programming temporary basal deliveries								✓						✓	
cf_{96}: User measures or enters BG values incorrectly								✓						✓	
cf_{97}: User provides incorrect parameters to the bolus calculator. These parameters include the users insulin sensitivities and corresponding effective periods; insulin-to-carbohydrate ratios and corresponding effective periods; the users target BG levels and corresponding effective periods; and insulin duration of action								✓						✓	
cf_{98}: User estimates carbohydrate content of planned meals incorrectly								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{98a}: User guesses, instead of consulting with food database, the number of carbohydrates in the meal								✓						✓	
cf_{98b}: User specifies incorrect categories or amounts of ingredients in the meal								✓						✓	
cf_{99}: User touches the input units of the pump accidentally, causing unintentional changes on pump settings, pump states, or insulin delivery programs								✓						✓	
cf_{100}: User interacts improperly with the input mechanisms of the pump, e.g., pressing the keypad for too long or not long enough, causing the pump to misinterpret the users intention								✓						✓	
cf_{101}: User fails to confirm revisions on insulin delivery programs, leaving the pump unchanged without his/her awareness								✓						✓	
cf_{102}: User forgets to confirm his/her action of activating another basal profile, leaving the current basal profile to continue without his/her awareness								✓						✓	
cf_{103}: User forgets to confirm his/her action of starting or stopping a temporary basal, a normal bolus, or an extended bolus								✓						✓	
cf_{104}: User commands a bolus to cover a meal but does not eat								✓						✓	

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{105}: User eats but forgets to bolus														✓	
cf_{106}: User commands boluses without consulting with the bolus calculator or inappropriately overrides boluses recommended by the bolus calculator								✓						✓	
cf_{107}: User inappropriately cancels a bolus in mid-delivery								✓						✓	
cf_{107a}: User misunderstands suggestions from the bolus calculator								✓						✓	
cf_{107b}: User stops a bolus due to false symptoms of hypoglycemia								✓						✓	
cf_{108}: User forgets to resume after suspending the pump														✓	
cf_{109}: User programs a special basal profile targeted at certain occasions, but forgets to activate this profile when targeted occasions occur								✓						✓	
cf_{110}: User fails to attend to pump notifications								✓						✓	✓
cf_{110a}: Human factors issues								✓						✓	✓
cf_{110b}: Excessive background noise								✓						✓	✓
cf_{110c}: Outside lighting condition prevents the user from interacting with the pump correctly								✓						✓	✓

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{110d}: User muffles the speaker of the pump or other audio devices, either intentionally or unintentionally								✓						✓	✓
cf_{110e}: User disregards pump notifications intentionally								✓						✓	✓
cf_{110f}: Nuisance or false notifications occur too often and are subsequently ignored by the user								✓						✓	✓
cf_{110g}: User falls asleep or goes into coma due to hypoglycemia								✓						✓	✓
cf_{111}: Human factors issues								✓						✓	✓
cf_{111a}: Information overload								✓						✓	✓
Environmental Sources of Hazardous Situations															
cf_{112}: Outside temperature is out of safe range or fluctuating inadvertently, causing the pump to deliver insulin inaccurately or to behave erratically								✓						✓	✓
cf_{113}: Outside air pressure is out of safe range or fluctuating inadvertently, causing the pump to deliver insulin inaccurately or to behave erratically								✓						✓	✓
cf_{114}: Electromagnetic interference								✓						✓	✓
cf_{114a}: Inadequate immunity or mitigation								✓						✓	✓
cf_{114b}: Improper manufacturing process								✓						✓	✓

Continued on next page

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

	h_{1a}	h_{1b}	h_{1c}	h_{1d}	h_{1e}	h_{1f}		h_{3a}	h_{3b}	h_{3c}	h_{3d}	h_{3e}	h_{3f}	h_{4a}	h_{4b}
cf_{114c}: Failure to reinstall electromagnetic compatibility (EMC) components after service or reinstalling EMC components incorrectly								✓						✓	✓
cf_{114d}: Physical damage to the pump or its sub-assemblies								✓						✓	✓
cf_{114e}: Pump is used in the presence of electromagnetic disturbances that exceed its design specifications								✓						✓	✓
cf_{115}: Excessive background noise (preventing the user from attending to pump notifications)								✓						✓	✓
cf_{116}: Outside lighting condition prevents the user from interacting with the pump correctly								✓						✓	✓
cf_{117}: Unauthorized personnel tamper with pump configuration settings								✓						✓	✓
cf_{118}: Unauthorized personnel tamper with information critical to insulin delivery								✓						✓	✓
$5cf_{119}$: loose object translation															

Table D.2: The coordination of Zhang's hazards and contributing factors for the GIIP. The contributing factors are extracted from Zhang's appendices [149].

Appendix E

Generalized Insulin Infusion Pump Hazard Table

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H1	Incorrect [low integrity] [task parameter] meal bolus [task object] is recommended [task object] by the bolus calculator [event agent]	PHL	User oversight via user interface. Software quality control	O:2 D:1 R:Medium	O:3 D:1 R:Medium	O:3 D:2 R:Low	O:3 D:2 R:Low	Bolus Calculator & Food Database	TBD	TBD
H1.1	Food database [error context] contains erroneous caloric density information [error condition], causing incorrect calculation [task action] of the number of carbohydrates in a meal [task object] containing the food whose caloric density was erroneous [task parameter] by the user [event agent]	PHL	Quality control of content curation	O:2 D:2 R:Medium	O:3 D:2 R:Medium	O:3 D:2 R:Low	O:3 D:2 R:Low	Food Database	TBD	TBD
H1.2	Design flaws / implementation defects [error condition] in the bolus calculator [error context][event agent] produces incorrect [error condition] recommendation [error element]	PHL	Software quality process	O:2 D:4 R:High	O:2 D:3 R:Serious	O:2 D:4 R:Medium	O:2 D:4 R:Medium	Bolus Calculator	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H2	Pump controller [event agent] unexpectedly [event condition] restores [event action] to default factory settings [task parameter] without the user's [event agent] awareness [task parameter]	PHL	✗	✗	✗	✗	✗	✗	TBD	TBD
H2.1	User [event agent] inadvertently[error condition] selects a restore[event object] of the factory settings [event task] using the user interface [error context]	PHL	Usability Testing & Workflow design	O:1 D:4 R:High	O:1 D:4 R:Medium	O:1 D:4 R:Medium	O:1 D:4 R:Medium	User interface input device & User interface output device	TBD	TBD
H2.2	Accumulated [error condition] static electricity [error element] during use [event context] triggers an unexpected [error condition] restore [event task] of default factory settings [error element]	PHL	Physical testing	O:1 D:2 R:Medium	O:1 D:2 R:Low	O:1 D:2 R:Low	O:1 D:2 R:Low	All	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-ECA-2A	The pump controller [event agent] fails to store [task action] user[event agent]-defined [task action] insulin delivery profile [task object]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2A.1	The user [event agent] fails to correctly enter [task action] an insulin delivery profile [task object] on account of a user interface [event agent] with low usability [task parameter]	ECA	Usability Testing	O:2 D:4 R:High	O:2 D:4 R:Serious	O:2 D:4 R:Medium	O:2 D:4 R:Medium	User interface input device & User interface output device	TBD	TBD
H-ECA-2A.2	The user interface [event agent] fails to transmit [task action] the user [event agent] entered [task action] profile [event object] to the pump controller [event agent]	ECA	Software quality assurance process	O:1 D:3 R:Serious	O:1 D:3 R:Medium	O:1 D:3 R:Medium	O:1 D:3 R:Low	User interface input device	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-ECA-2A.3	The pump controller [event agent] is denied access [error condition] to the persistence mechanism [event agent] and fails to store [task action] the user [event agent] entered [task action] profile [task object]	ECA	Software quality assurance process & Configuration instructions & Self Checks	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump controller	TBD	TBD
H-ECA-2B	The pump controller [event agent] corrupts [error condition] user[event agent]-defined [task action] insulin delivery profile [task object]	ECA	✗	✗	✗	✗	✗	✗	✗	✗
H-ECA-2B.1	The pump controller [event agent] corrupts [error condition] the profile [task object] duration [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2B.2	The pump controller [event agent] corrupts [error condition] the profile [task object] time unit [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-ECA-2B.3	The pump controller [event agent] corrupts [error condition] the profile [task object] drug dosage [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2B.4	The pump controller [event agent] corrupts [error condition] the profile [task object] volume unit [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2B.5	The pump controller [event agent] corrupts [error condition] the profile's [task object] drug identity [error element] (expecting a different type of insulin)	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2B.6	The pump controller [event agent] corrupts [error condition] the profile's [task object] concentration unit [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-ECA-2B.7	The pump controller [event agent] corrupts [error condition] the profile's [task object] total necessary drug volume [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2B.8	The pump controller [event agent] corrupts [error condition] the profile's [task object] flow unit [error element]	ECA	Software quality assurance process	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD
H-ECA-2C	The user interface input device [event agent] misleads [error condition] the user [event agent] resulting in mode confusion [error condition] such that the profile [task object] is not stored [task action]	ECA	Usability testing	O:2 D:2 R:Medium	O:2 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	Pump Controller	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-ECA-2D	The pump controller [event agent], user input interface [event agent] and bolus calculator [event agent] do not consistently use [error condition] the same profile structure [task object] e.g., units, template [error element]	ECA	Software quality assurance process	O:1 D:4 R:Low	O:1 D:4 R:Low	O:1 D:4 R:Low	O:1 D:4 R:Low	Pump Controller	TBD	TBD
H-CFA-1	[Can't] Turn [task action] pump [event agent] on [task parameter] - E2b(1)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-1.1	Battery [event agent] dead [task parameter]	CFA	Low power battery powered self check & Battery monitoring in pump controller	O:1 D:1 R:Medium	O:1 D:1 R:Low	O:3 D:1 R:Low	O:3 D:1 R:Low	Pump controller	TBD	TBD
H-CFA-1.2	No power [error condition]	CFA	Low power battery powered self check & Battery monitoring in pump controller	O:1 D:1 R:Medium	O:1 D:1 R:Low	O:1 D:1 R:Low	O:1 D:1 R:Low	Pump Controller	TBD	TBD
H-CFA-2	Pump controller [event agent] fails [error condition] self-check [task action] E2b(2)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-2.1	Electrical mechanical failure [error condition]	CFA	Reliability testing	O:1 D:1 R:Medium	O:1 D:1 R:Low	O:1 D:1 R:Low	O:1 D:1 R:Low	GIIP device	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-CFA-3	User [event agent] pushes [task action] Options Button [task object] - E2b(3)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-4	Bolus Calculator [event agent] performs calculations [task action] - E2b(4i)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-4.1	Inaccurate calculation [error condition]	CFA	Software quality assurance process	O:1 D:3 R:Serious	O:1 D:3 R:Medium	O:1 D:4 R:Medium	O:1 D:4 R:Low	Bolus calculator	TBD	TBD
H-CFA-4.2	Incomplete process [task action] - no double check [error condition]	CFA	User training	O:3 D:4 R:High	O:3 D:4 R:Serious	O:3 D:4 R:Medium	O:5 D:4 R:Medium	User	TBD	TBD
H-CFA-4.3	Incorrect weight [error element]	CFA	User training & Usability Testing	O:1 D:4 R:High	O:1 D:4 R:Medium	O:2 D:4 R:Medium	O:3 D:4 R:Medium	User & Output devices & Input devices	TBD	TBD
H-CFA-5	Enter rate [task action] - E2b(5i)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-5.1	Push [task action] incorrect [error condition] key pads [task objects]	CFA	User training & Usability testing	O:1 D:3 R:Serious	O:1 D:3 R:Medium	O:2 D:3: R:Medium	O:2 D:3 R:Low	User & User interface input device & User interface output device	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-CFA-5.2	User [event agent] misreads [error condition] order [task action] [task object]	CFA	User training & Physician training	O:3 D:4 R:High	O:3 D:4 R:Serious	O:4 D:4 R:Serious	O:5 D:4 R:Serious	User	TBD	TBD
H-CFA-5.3	User [event agent] enters [task action] volume [error element] - E2b(6i)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-5.4	User [event agent] adjusts [task action] volume [error element] based on assumptions [task parameter] - too much [error condition]	CFA	User training	O:1 D:4 R:High	O:1 D:4 R:Medium	O:2 D:4 R:Medium	O:2 D:4 R:Medium	User	TBD	TBD
H-CFA-5.5	User [event agent] adjusts [task action] volume [error element] based on assumptions [task parameter] - too little [error condition]	CFA	User training	O:1 D:4 R:High	O:1 D:4 R:Medium	O:2 D:4 R:Medium	O:2 D:4 R:Medium	User	TBD	TBD
H-CFA-5.6	User [event agent] pushes [task action] incorrect [error condition] key pads [task object]	CFA	User training & Usability testing	O:1 D:4 R:High	O:1 D:4 R:Medium	O:1 D:4 R:Medium	O:1 D:4 R:Medium	User & User interface input devices & User interface output devices	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-CFA-5.7	User [event agent] misread [error condition] [task action] order [task object]	CFA	User training & Physician training	O:3 D:4 R:High	O:3 D:4 R:Serious	O:4 D:4 R:Serious	O:5 D:4 R:Serious	User	TBD	TBD
H-CFA-5.8	User [event agent] enters [task action] rate [error element] instead of volume [error element]	CFA	User training & Usability testing	O:2 D:4 R:	O:3 D:4 R:	O:3 D:4 R:	O:3 D:4 R:	User & User interface output devices & User interface input devices	TBD	TBD
H-CFA-6	User [event agent] pushes "run" [task action] after verifying [task action] data entered [task object] - E2b(7i)	CFA	✗	✗	✗	✗	✗	✗	✗	✗
H-CFA-6.1	No activation - "run" not pushed [error condition] by user [event agent]	CFA	User training	O:1 D:2 R:Medium	O:1 D:2 R:Low	O:2 D:2 R:Low	O:2 D:2 R:Low	User	TBD	TBD
H-CFA-6.2	User [event agent] misreads [task action][error condition] user interface output device [event agent]	CFA	User training & Usability testing	O:2 D:4 R:High	O:2 D:4 R:Serious	O:3 D:4 R:Medium	O:3 D:4 R:Medium	User & User interface output devices	TBD	TBD
H-PFA-Present-Controls-A1	User interface output device [event agent] unavailable [error condition]	PFA	Software quality assurance process & Hardware quality assurance process & Battery monitoring	O:1 D:1 R:Medium	O:1 D:1 R:Low	O:2 D:1 R:Low	O:2 D:1 R:Low	Pump controller	TBD	TBD

Table E.1: The hazard table for the GIIP used for Denney's lightweight assurance case construction method.

Hazard Id	Hazard	Source	Mitigation	Catastrophic	Critical	Marginal	Negligible	Allocation	Verification Method	Verification Allocation
H-PFA-Present-Controls-D1	User interface input device [event agent] has superfluous [error condition] controls/additional features [task object]	PFA	User training & Usability Testing	O:1 D:3 R:Serious	O:1 D:3 R:Medium	O:1 D:3 R:Medium	O:1 D:3 R:Low	User interface output devices & User interface input devices	TBD	TBD
H-PFA-Present-Controls-E1	User interface [event agent] provides incomplete functionality [task parameter] forcing workarounds [error condition]	PFA	Requirements gathering process & Usability testing	O:1 D:1 R:Medium	O:1 D:1 R:Low	O:1 D:1 R:Low	O:1 D:1 R:Low	User interface input devices & User interface output devices	TBD	TBD
H-PFA-Present-Controls-G1	User interface input device [event agent] expects user [event agent] to initiate [task action] interaction [error condition]	PFA	✗	✗	✗	✗	✗	✗	✗	✗

Glossary of Terms

Accessibility "[T]he extent to which the system can be used by all kinds of users regardless of any physical or psychic characteristic they may have (e.g., disabilities, limitations, age, etc.). This attribute is subdivided into others in accordance with specific characteristics (visual, auditory, speech, motor, and cognitive)." [3].

Accuracy "A qualitative assessment of correctness or freedom from error."[13].

Applicability The fact or state of being pertinent [90].

Association A relationship between entities which is independent of those entities inherent properties..

Availability The degree to which a system or component is operational and accessible when required for use. [58].

Capital The necessary material and human resources to accomplish a stated task.

Care Provider The actor who is monitoring the health care process of the patient through direct or synthesized sensing or controlling the health care process directly or indirectly by making decisions and affecting process changes.

CIS Input Interface The system component used to indirectly control the health care process at the point of diagnosis and planning.

CIS Output Interface The system component used to acquire information recorded through synthesized perception of the health care process at the point of diagnosis and planning.

Clarity "the ease with which the system can be perceived by the mind and the senses. " [3].

Clinical Integrity The soundness of clinical decision-making..

Completeness The quality of "having all necessary parts, elements, or steps" [90].

Consistency The uniformity of design, implementation, and notation [84].

Contextual Health Care Process The health care process being described as opposed to the one being controlled, in other words, the model at the current level of focus.

Core CIS Software synonymous with Medical Data Device System.

Cultural Universality [T]he extent to which users from different cultural backgrounds can use the system. ... [L]anguage and other cultural convention (use of symbols, measurement units, numeric formats, etc.) [independent]" [3].

Currency "The quality or state of being current" [90].

Data A representation of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automatic means... [58].

Data Validity The degree to which data is accurate and precice..

Data Integrity The degree to which a collection of data is complete, consistent, and accurate. [58].

Datum Singular for data. [58].

Diagnostic Interface The system component used in the direct perception of the health care process at the point of observation.

Fault-Tolerance The ability of a system or component to continue normal operation despite the presence of hardware or software faults. [58].

Harmful Incident A patient safety incident that resulted in harm to the patient [14].

Hazard A collection of one or more contextual contributing factors which pose the risk of leading to a harm or loss event.

Health Care Process The controlled health care process (usually the patients care process).

Health Care Process Output Interface The mechanisms and devices that are applied to modify the behavior of the contextual health care process..

Health Care Process Input Interface The mechanisms and devices that are applied to monitor the behavior of the contextual health care process.

Health Care Process Controller The controller that automates the contextual health care process via technical controllers and controls it directly through actuators and sensors.

Helpfulness "[T]he means provided by the system to help users when they cannot infer or remember how to use the system." [3].

Human Hazard A hazard that can be overcome with sufficient cognitive effort.

Information Multiple datum linked by associations.

Integrity Correctness, completeness, fitness for use..

Interoperability The degree to which a system can exchange information with another system which provides input which is necessary to satisfy requirements. Adapted from [107].

Knowability The property by means of which the user can understand, learn, and remember how to use the system.[3].

Knowledge Inference of additional information from information.

Knowledge Integrity The degree to which the premise from which reasoning is done is complete and correct when compared against the expected subset of the domain specific consensus knowledge base, in combination with the degree to which the reasoning itself is sound and complete.

Machine Hazard A hazards can not be overcome with cognitive effort..

Medical Data Device System "[A medical data device system (MDDS)] is a device that is intended to transfer, store, convert from one format to another according to preset specifications, or display medical device data. An MDDS acts only as the mechanism by which medical device data can be transferred, stored, converted, or displayed. An MDDS does not modify the data or modify the display of the data. An MDDS by itself does not control the functions or parameters of any other medical device. An MDDS can only control its own functionality. An MDDS device is not intended to provide or be used in connection with active patient monitoring. Any product that is intended for a use beyond the uses (or functions) identified ... is not an MDDS [34].

Memorability The degree to which the system "enables the user to remember the elements and the functionality of the system" [3].

Near Miss Event A patient safety incident that did not reach the patient and therefore no harm resulted [14].

No Harm Event A patient safety incident that reached a patient but no discernible harm resulted [14].

Operability [T]he capacity of the system to provide users with the necessary functionalities and to permit users with different needs to adapt and use the system..[3].

Patient Security Incident An act of omission or commission that *intentionally* results in potential or actualized patient harm.

Patient Safety Incident An act of omission or acts of commission that *unintentionally* results in potential or actualized patient harm..

Physical Integrity Material soundness.

Precision The degree of exactness or discrimination with which a quantity is stated [107].

Relational Integrity The correctness of associations among data and the completeness of that set when considered against the common ground knowledge base that is shared by the producer and consumer of that information..

Reliability The ability of a system or component to perform its required functions under stated conditions for a specified period of time.[58].

Soundness "Firmness, solidity; freedom from weakness, defect, or damage; goodness of condition or repair" [100] - "Thoroughness, completeness." [100].

Technical Integrity The soundness of technical decision-making.

Technical Output Interface The mechanisms and devices that are applied to modify the software, hardware, and other technical elements of the contextual health care process.

Technical Input Interface The mechanisms and devices that are applied to monitor the software, hardware, and other technical elements of the contextual health care process.

Technical Administration Process Output Interface The tools and processes used by the health care process controller to indirectly control the contextual health care process via a technical controller.

Technical Administration Process Input Interface The tools and processes used by the health care process controller to indirectly observe the contextual health care process via a technical controller.

Technical Administration The controller that directly controls and senses the software and hardware processes and artifacts within the contextual health care process. This controller also mediates the indirect control of the Health Care Process Controller.

Transformation The automatic generation of a target model from a source model, according to a transformation definition. [68].

Transformational Integrity The soundness of computation.

Transformation Rule A description of how one or more constructs in the source language can be transformed into one or more constructs in the target language. [68].

Transformation Definition A set of transformation rules that together describe how a model in the source language can be transformed into a model in the target language. [68].

Treatment Interface The system component that is used in the direct control of the health care process at the point of treatment.

Universality "[T]he extent to which the system can be used by all kinds of users." [3] see Accessibility, Cultural Universality .

Validity The degree to which a solution meets specified requirements - adapted from validation [107].

Glossary of Acronyms

ADE Adverse Drug Event.

AHLA American Health Lawyers Association.

AHRQ Agency for Healthcare Research and Quality.

ALARP As Low As Reasonably Practicable.

ALM Application Lifecycle Management.

ANSI-STD-0010-2009 ANSI STD-0010-2009 Standard Best Practices for System Safety Program Development and Execution.

ASHRM American Society for Healthcare Risk Management.

BDD Block Definition Diagram.

CAST Causal Analysis Using STAMP.

CDS Clinical Decision Support.

CF Contributing Factor.

CFA Component Fault Analysis.

CGMP Common Good Manufacturing Guidelines.

CIS Clinical Information System.

CMIO Chief Medical Information Officer.

CPOE Computerized Provider Order Entry.

DOD Department of Defense.

ECA Event Chain Analysis.

EHR Electronic Health Record.

EIT Engineer in Training.

EMR Electronic Medical Record.

ER Entity Relationship.

ETA Event Tree Analysis.

FDA Food and Drug Administration.

FMEA Failure Mode and Effects Analysis.

FR Functional Requirement.

FRAM Functional Resonance Analysis Method.

FTA Fault Tree Analysis.

GIIP Generalized Insulin Infusion Pump.

GSN Goal Structuring Notation.

HAZOP HAZard OPerability.

HCI Human Computer Interaction.

HFMEA Health Care Failure Mode and Effects Analysis.

HIN Health Insurance Number.

HIT Health Information Technology.

HL7 Health Level 7.

ICD International Classification of Disease.

IEEE1228 IEEE 1228 Standard for Software Safety Plans.

IOM Institute of Medicine.

IPC Inter Process Communication.

ISHA Information System Hazard Analysis.

ISO International Standards Organization.

ISO 13308 Health Informatics - Requirements for an Electronic Health Record Architecture.

JCAHO Joint Commission on Accreditation of Healthcare Organizations.

LHS left hand side.

LOINC Logical Observation Identifiers Names and Codes.

MIL-STD-882E United States Department of Defense Standard Practice System Safety.

MOA Medical Office Assistant.

MRP Most Responsible Provider.

NCC MERP National Coordinating Council for Medication Error Reporting and Prevention.

NFR Non-Functional Requirement.

NHS National Health Service.

NP Nurse Practitioner.

OMG Object Management Group.

OOP Object Oriented Programming.

PEng Professional Engineer.

PFA Process Fault Analysis.

PHA Preliminary Hazard Analysis.

PHL Preliminary Hazard List.

RAC Risk Assessment Code.

RCA Root Cause Analysis.

RHS right hand side.

RID Retrospective Incident Data.

RIM Reference Information Model.

RPN Risk Probability Number.

RTS Real Time Systems.

SACM Structured Assurance Case Metamodel.

SCEM Safety Constraint Enforcement Mechanism.

SIL Safety Integrity Level.

SNOMED-CT Systematized Nomenclature of Medicine – Clinical Terms.

SQL Structured Query Language.

SR Safety Requirement.

STAMP System Theoretic Accidents Models and Processes.

STPA System Theoretic Process Analysis.

SUI System Under Investigation.

TTPS Thermal Tile Processing System.

UML Universal Modelling Language.

UTM Universal Triangulation Model.

VA Veterans Affairs.