Towards Improved Certification of Complex FinTech Systems – A Requirements-based Approach

Sepehr Sharifi, Daniel Amyot, John Mylopoulos
School of EECS
University of Ottawa
Ottawa, Canada
{sshar190 | damyot | jmylopou}@uottawa.ca

Patrick McLaughlin
Brane Capital
Ottawa, Canada
patrick@brane.capital

Ray Feodoroff
School of CIT
University of Wollongong
Wollongong, Australia
agentorientedassurancecases@gmail.com

Abstract*—*Context: **Financial Technology (FinTech) systems, especially those involving custody of digital assets such as cryptocurrencies, are quickly emerging as a new class of software systems with associated high risks. So far, incidents involving such systems have costed billions of dollars. *Problem:* Providing regulators and insurers with certification cannot simply rely on simple reports generated by auditors. Current practices require a more rigorous and systematic approach for capturing and communicating the design rationale in order to certify such systems. *Method:* The User Requirements Notation (URN) is used to model and analyze the requirements of a FinTech system and capture its design rationale. Then, the Systems Theoretic Process Analysis (STPA) method is applied to the URN model to evaluate system hazards and introduce safety constraints/requirements that aim to avoid bad situations from happening (e.g., loss of assets, private data, or reputation). The results augment the URN model and are conveyed to the stakeholders (especially regulators, auditors, and insurers) in the form of assurance cases. *Results:* Guidelines are now available to model the requirements of FinTech systems and produce assurance cases for certification. The guidelines are illustrated with a real digital asset custodian example. *Conclusion:* This work provides new requirements-based guidelines exploiting URN and STPA that can potentially facilitate the certification process of FinTech systems.**

***Index Terms*—Assurance cases, Certification, Financial Technologies, Goal-oriented Requirements Engineering, Safety, STPA, User Requirements Notation**

I. Introduction

Financial Technology (FinTech) systems aim to automate and improve the delivery and usage of sophisticated financial services. Most such systems are becoming safety critical as they carry larger risks for their owners and users. Take the example of FinTech systems that manipulate *digital assets* (DA) such as Bitcoins. Despite many claims about their security and safety properties, they are prone to various types of breaches and failures that now cost billions of dollars [1]. Furthermore, with the proliferation of DAs, FinTech systems that involve custody or transfer of large amounts of DAs act as hubs in the highly-interconnected financial network, thus becoming a critical safety point that would have systemic effects in case of failure [2].

It is worth mentioning that many clients of such services are not private investors; rather, they are pension funds that hold many people's retirement savings. Loss of a pension fund's assets certainly affects the livelihoods of many people. Therefore, the financial industry has a societal responsibility to move towards more rigorous practices that can ensure client's asset safety, given the added complexity of novel technologies that are increasingly embedded in their processes and products.

Currently, financial institutions must assure regulators (policy makers, authorities, etc.) that their systems minimize the risk of losing investors' assets [3]. Such FinTech systems must also address multi-faceted concerns (finance, business, software, cryptography, distributed ledgers, regulations) and convey compliance and risk assessment results in a clear and understandable manner to certification authorities. Nevertheless, with the added complexity of FinTech systems, financial regulators and auditors face a challenge in assessing the compliance of such systems' requirements and designs to the regulations. Although the regulatory stakeholders should add to their capabilities in assessing more technologically complex systems, FinTech developers should also capture and convey compliance and risk assessment results in a clear and understandable manner to certification authorities.

This paper builds on early work [4], and reports on a project that uses requirements engineering techniques and standards that are foreign to the FinTech industry, but common in other safety-critical engineering domains, in addition to current financial regulations, to rise up to the challenge of building a digital asset custody system that keeps institutional investors' assets safe (e.g., through a set of multi-signature smart contracts and wallets).

The main contribution of this paper is a set of requirements-based guidelines that apply well-known approaches from safety-critical domains to enable an improved and more rigorous certification of FinTech systems. These guidelines exploit goal modeling, systemic safety assessment, and assurance case development. This paper also provides an illustrative example, based on an industrial collaboration project, where these guidelines are applied.

Inspired by Design Science [5], we have identified and scoped our problem by discussing with experts at a FinTech company and by assessing related work (Sect. II). We have iteratively and rigorously developed and validated recommended guidelines towards developing models that would improve the FinTech certification process (Sect. III). These guidelines are the main artefact produced in our research. The relevance of these guidelines comes from a real problem in a real FinTech organization, and they are illustrated in Sect. IV on an example that captures a subset of the problem and of its solution. Finally, Sections V and VI respectively discuss lessons learned and our conclusions.

II. Background

This section provides background information on concepts relevant to this paper. First, an overview of the current state of FinTech certification is presented. Then, *System Theoretic Process Analysis* [6] is introduced, as it forms a major component of the proposed guidelines. STPA allows the developers to design controls on a system's behavior in a top-down manner while ensuring the hierarchy of controls satisfy top-level goals. Finally, assurance cases are briefly discussed.

A. FinTech Certification

Information security has been the focus of much work [7], but many challenges remain in a FinTech context. In North-America, the most notable certification applied to financial entities is the *System and Organization Controls* (SOC) from the American Institute of Chartered Professional Accountants (AICPA) [8], where *controls* are essentially the tools that are imposed on the system to ensure its safety. Different SOC reports approach system controls from different perspectives, namely compliance, operations, and financial reporting. The most relevant report on system-level and entity-level operational controls of a financial service organization is the *SOC2 report*, based on the guidelines provided by the AICPA [9] and CPA Canada, known as Trust Services Principles and Criteria (TSC) for *Security*, *Availability*, *Processing Integrity* and *Confidentiality*. All standards focus on controlling the behavior of the system during its operation [10].

Systems developed in the FinTech industry have major human and software elements, and are hence considered socio-technical systems. Goal modeling has shown its utility when it comes to capturing the needs of stakeholders to be fulfilled by such systems, while providing support for compliance, trade-off and decision-making [11]. The User Requirements Notation (URN) [12] integrates the *Goal-oriented Requirement Language* (GRL) with the *Use Case Maps* (UCM) process notation, together with traceability information and the ability to profile the language to specific domains [13]. URN combines goal and process views used in requirements engineering activities, but also in regulatory compliance [14] and regulatory intelligence [15]. Though much work has been done on goal-based requirements engineering, to our knowledge, none has addressed FinTech certification except for our own early work in this area [4].

B. System Theoretic Process Analysis (STPA)

STPA, proposed by Leveson [6], is a hazard analysis methodology based on *System Theoretic Accident Model and Processes* (STAMP). This methodology evaluates possible ways in which an undesirable systemic event (*loss*) can occur. It also allows for design of a hierarchy of controls on the system and its components. There are parts of STPA that are relevant to our requirements-based certification approach.

STAMP defines *loss states* as system states that are unacceptable to the stakeholders of the system, e.g., loss of human life, of assets, or of reputation. Furthermore, the system states where nothing is yet lost but that can transition to a loss state, given adverse environmental conditions, are called *hazards* [6]. Figure 1 illustrates various states of a system, i.e., safe, hazard, and loss states, together with their transitions. In the case of a passenger transport train, a passenger train is in a loss state if one of its passengers has fallen off the wagon and is harmed (*A1*), whereas a related hazard state can be where the train's doors are open while it is moving (*H1*). Given the state of the system' environment, e.g., the curvature of the railway (*C1*), passengers can fall off the train if it approaches a curve while its doors are open, thus triggering the loss state. Alternatively, the train might be travelling on an almost straight path (*C3*) until it reaches its next stop (stationary train with open doors is considered a safe state, i.e., *S3*).

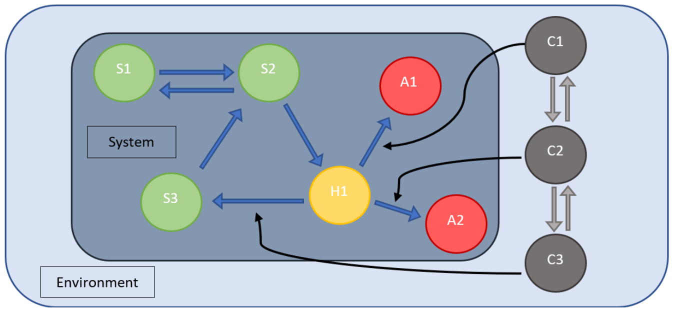

Fig. 1. The relationship among a system's safe states (green), hazard states (yellow), and loss states (red), and its environmental states (grey).

STPA applies these concepts to the hierarchical control structure of a system in order to identify possible ways the system can transition to a hazard state. Such control actions that can lead to hazards are called Unsafe Control Actions (UCAs). These UCAs then can act as a basis for identification of causal scenarios that might cause the UCAs. These hazards are then mitigated with the introduction of traceable safety constraints and requirements based on design recommendations [6].

An important benefit of using STPA as part of a requirements-based certification approach is twofold. First, STPA has a broad definition of loss that is amenable to the types that are commonly defined by the FinTech regulatory stakeholders, i.e., loss of clients' assets (e.g., money or cryptocurrencies), loss of service provider's reputation, and loss of clients' private information. A detailed list of potential system losses is provided in Table I. Second, STPA provides a methodology to systematically define system controls that are traceable to the losses we are trying to prevent.

There also exists examples of applications of STPA within security analysis [16] and combined safety and security analysis [17].

C. Assurance Cases

Assurance cases document evidence that provides a convincing and valid argument that critical claims regarding properties (often related to safety) are justified for a given system in a given environment. Many notations and tools have been proposed in order to create structured assurance arguments [18], including the *Goal Structuring Notation* (GSN) [19].

As an alternative to GSN, Feodoroff [20]–[22] has also proposed that GRL goal models can document arguments and justifications (for system-level qualities) as part of design rationales rather than in other formats, which lack the ontological richness of URN, while also preventing assurance case development activities that may be redundant. GRL supports concepts for actors, their intentional elements (goals, softgoals, tasks, and resources, and their relationships (AND/OR decompositions, dependencies, and weighted contributions). As previous experience with the application of URN in a regulatory compliance context has shown [14], providing regulators with (objective) evidence and their links to qualities in terms of URN models would help attain a "clearer" picture of the system and decide on its acceptability with respect to safety risks. Feodoroff argues that this picture is clearer because weighted dialogical reasoning, as provided by GRL, can account for the weight of assurance claimed versus counter-claims.

This notion of balancing claims with counter-claims is now the new preferred style of reasoning for assurance cases, which is described as *counterfactual* [23]. Counterfactual reasoning has always been native to GRL (and *i**) via the nature of the contribution relations, which can be positive or negative, at different levels of strength.

D. Related Work

Approaches for risk modelling such as CORAS [24] and FTA [25] are also used for *security* risk modelling. FTA offers interesting variants such as *attack trees*, but was developed over 80 years ago when system complexity was far simpler than today. CORAS is based upon models in risk standards but has not kept pace with the gravitation towards assurance cases in the past decade. While risk is a component of any assurance case, the requirement for claim, argument, and evidence now predominates over areas other than risk, especially compliance. Both CORAS and FTA also require a system model to base the structure of a risk tree, but neither approach provides an elicitation mechanism of the level of rigor now applied for STPA [26]. Certainly, STPA is now being enshrined in an SAE standard[1], because it is recognised as superior to risk-based tree approaches for complex systems. The use of a risk calculation approach can always be applied post STPA to provide a risk position post analysis if required [27], especially since the threats and the corresponding mitigations would already have been provided by STPA.

Historically, there have been several approaches that attempted to incorporate GSN models into the design process, but required a separate design notation [28]–[31]. These approaches create a collage of techniques and representations often with informal relationships between the representations. This informality makes it hard to ensure evidentiary chain between the safety analysis, the design, and the assurance argument. This informality can also lead to information lossiness between representations. It is counter-intuitive then how the potential for information lossiness, inherent in any collage of approaches, cannot but affect comprehension and therefore confidence in the assurance arguments. Kelly and McDermid [32] recommend a laborious approach to keeping the design reasoning and the assurance reasoning synchronized, where the synchronization is required because of the separation in reasoning spaces.

Indeed, the reasoning style of GSN is now, circa 2020, deprecated according to Bloomfield and Rushby [33]. The new "style" of assurance reasoning suggested by Bloomfield and Rushby is actually *counterfactual* (see Sect. II-C). The influence of the outdated case-based reasoning style can otherwise be seen in approaches that can opt for a notation that could support counterfactual reasoning, but express the assurance reasoning archaically [34], so as to maintain the status quo. Importantly, circa 2020, the introduction of counterfactual relationships into case-notations is still only a proposal [35, p. 17], so counterfactual reasoning is not available in the more popular case notations. Assurance reasoning, in a design rationale notation with counterfactual relationships, can certainly already be accomplished [22].

It should be noted then that there are a number of approaches that look at using STPA in association with a non-counterfactual style of assurance arguments, the STPA being modelled separately and therefore with an information-lossy gap with the assurance reasoning [30], [34]. This is more error-prone than an approach that models the safety analysis and assurance reasoning within the design notation to keep information lossiness to a minimum [22].

The motivation for this paper is hence to determine means to integrate security analysis, design reasoning, and assurance reasoning within one notation. This work aims to support economical and coherent counterfactual arguments to enhance the veracity of assurance cases by the reduction of information lossiness between representations. We argue that a promiscuous goal-oriented notation facilitates this integration [20], [22].

III. Guidelines for FinTech Certification

The following guidelines are provided as a means to design a FinTech system that satisfies systemic qualities (such as safety, security, privacy, etc.), and communicate the design artifacts (including additional requirements) to the stakeholders in a manner that would improve the effectiveness and efficiency of the certification process.

Step-1 (Identify the stakeholders): The first step in modeling the goals of a FinTech system is to gain a holistic understanding of its stakeholders and their dependencies. Such view is often called a strategic dependency model in goal-

[1]https://www.sae.org/standards/content/j3187_202202/

oriented requirements engineering [36]. Actors can often be categorized into four major expertise domains:

1) *Governance and policy*;
2) *Regulatory and compliance*;
3) *Business and operations*; and
4) *Technical (engineering)*.

The stakeholders, with varied expertise in the above domains, have unequal understanding of the core of the system. In an onion-like stakeholder model, we find them in various layers from the core *operational* layer, to the *system* layer (containing the management and support elements), and then to the *environment* layer.

This view is employed in the early stages of the system development to draft the *Concept of Operations* (ConOps) of the system. Adding a development viewpoint to the ConOps view would require the list of stakeholders to be augmented with, e.g., project managers and system developers. The system development team is present in all three layers of the system and has interactions with virtually all other stakeholders (except negative actors such as hackers).

Various sources of information such as handbooks, standards, and interviews with stakeholders and decision makers are used to uncover valuable dependencies. These dependencies, in turn, become the basis for sketching a strategic dependency model of the system, which starts by outlining the actors and their directional dependencies, hence enabling the elicitation of the actor goals. In GRL, the source and target of a dependency are intentional elements associated with different actors.

Step-2 (Create a strategic dependency model using GRL): Using the list of stakeholders that were identified, a strategic dependency diagram can be created that allows identifying the high-level goals of various actors and their inter-dependencies.

Step-3 (Expand the functional goals using UCMs): After the high-level goals have been refined to system-level functional goals, they can be further refined using Use Case Maps (UCMs). UCMs include process-oriented concepts such as start/end points, responsibilities (i.e., activities), their responsible system components or actors, and various relationships (sequencing, guarded choice, concurrency, process decomposition, timers, and others). UCM and GRL model elements can also be traced to each other to support alignment, consistency, or impact analysis [13]. The operational controls of the system can now be identified, which helps eliciting the control structure of the system.

Step-4 (Perform STPA): According to the STPA handbook of Leveson [37], the STPA process for a specific level of hierarchy is as follows:

- **Step-4.1 (Define the systems boundary):** The loss states and the hazard states that could transition to the loss states under undesirable environmental conditions (see Fig. 1) must be defined. By defining hazards, the boundary of the system, i.e., the domain that the designers and engineers think they can control, is determined.
- **Step-4.2 (Define the hierarchical control structure):** The hierarchical control structure of the system must be developed based on the functional architecture of the system.
- **Step-4.3 (Identify Unsafe Control Actions (UCAs)):** The control actions identified in **Step-4.2** can be hazardous in certain contexts if *provided*, *not provided*, *applied too soon/too late*, or *applied for too long/too short*. Identifying these UCAs paves the way to the identification of paths to hazards.
- **Step-4.4 (Identify loss scenarios):** Loss scenarios are the paths that can lead to the hazards. Causal factors can be identified on the basis of the scenarios and the control model.
- **Step-4.5 (Elicit safety constraints and requirements):** Identification of hazards and UCAs can lead to the introduction of safety constraints, where the identification of the causal factors can lead to making design recommendations and deriving corresponding safety requirements.

Step-5 (Create assurance cases based on STPA-related argumentation): The results of **Step-1** to **Step-3** can iteratively be updated based on the results of **Step-4**. Then, assurance cases can focus on presenting the design rationale rather than trying to prove a quality by omission or trying to prove safety after the system has been fully developed (which is prone to *confirmation bias* [38]).

IV. An Illustrative Example

The selected FinTech example is a commercial digital assets custody system. The integration of Distributed Ledger Technologies (DLTs) into financial systems is becoming increasingly popular as the utilization of digital assets is growing. The parties that are mandated with keeping digital assets safe are called digital assets custodians (DA Custodians) or crypto-custodians. They act similarly to custodian banks but focus on digital assets.

Customers should trust the DA custodians to keep their digital assets safe, other banks should trust them to do business with them, and regulators should trust these DA custodians to allow them to operate. Thus, SOC certifications are used to provide a part of that trust. The guidelines mentioned in Section III are applied to the aforementioned system. The following subsections, provide an excerpt of the results that are shareable (due to confidentiality constraints).

A. Step-1 to Step-3: Goal and Process Models

1) Stakeholder identification: The following stakeholders where identified for the digital assets custodian system:

- **System Operations:** Normal Operators (Finance Manager and Technical Users), Maintenance Ops (Hardware, Software), Operational Support (Transaction Mining, Customer Support), Interfacing System (Cloud, Internal Network).
- **System:** Internal Consultants (Compliance Officer), Client (Investors, Management), Functional Beneficiary (Institutional Funds).

- **System Environment:** Customer (White-label Partners), Interfacing System (Crypto-exchanges, Blockchains Platforms, Insurers), Regulators (Securities, Judiciary, Tax and KYC/AML[2] Authorities, Non-statutory Regulators including standards, consortia, and Self-Regulatory Organizations), Negative Actors (Competitor, Hacker), External Consultants (Business Auditor, Security Specialist, Legal Counsel).

Note the high number of regulatory stakeholders that are present in the ecosystem, which is typical of FinTech systems.

2) Elicit the strategic dependency model: A slice of the strategic dependency model is provided in Figure 2, illustrating how three actors, namely the Underwriter, the Securities Regulator, and the Judiciary, depend on the system's goal Report to produce the Security Report and on the Auditor's ability to perform verification (Verify). Multiple in/out dependency links are a shorthand for a *distributive* dependency relationships, e.g., there are 6 pairwise dependency relationships that have Security Report as their dependum. Here, the Brane actor is the DA custodian and service provider.

The creation of the GRL model has allowed the organization to discover critical resources, i.e., resources associated with a large number of dependencies which, if not realized properly, would result in the simultaneous dissatisfaction of multiple stakeholders. Thus, the requirements elicitation, system development/testing, and certification efforts can be prioritized in accordance with the criticality of the *dependums* and the system goals that provide them.

Note how many stakeholders depend on the Auditor, which shows how auditing companies (e.g., the "Big Four", namely KPMG, Deloitte, EY, and PwC) are playing a major role in the FinTech ecosystem.

3) Elicit the process models: The main concern of financial regulators is the controls on systems operations. To analyze the operational aspect of the system, the GRL model is expanded by defining Use Case Map processes for functional goals of the system. A process model of the Perform Due Diligence system goal is provided in Figure 3. The UCM model allows for analysis of the business process. The operationalization of goal Perform Due Diligence starts by the activity Evaluate Jurisdiction, which is the responsibility of the service provider (Brane), and yields one of two possible outcomes (illustrated by an OR-fork), i.e., being of low or high risk (the latter leading to the rejection of the client).

Performing operational analysis enables the organization to check the controls it has in place, assess alternative ones, provide a means to demonstrate compliance to the regulators, and have a basis for the analysis of systemic safety.

As an indicator of the size of the requirements/design model, the full URN model of the FinTech's digital asset management system, developed with the jUCMNav tool [39], is composed of 15 actors, 100 intentional elements, and 96 intentional links for the GRL view, and of 5 components, 84 responsibilities, and 22 interconnected process maps with up to 7 levels of nested decomposition for the UCM view.

[2]Know Your Client (KYC) and Anti-Money Laundering (AML) efforts are crucial in the financial industry.

TABLE I
LIST OF SYSTEM LOSSES

ID	System Losses
L.1	Loss of clients' digital assets under custody.
L.2	Loss of custodian's infrastructure.
L.3	Loss of custodian's reputation.
L.4	Loss of regulatory compliance of the custodian's operation.
L.5	Loss of clients' private/confidential information.

B. Step-4: Perform STPA

1) Define the system boundary: Based on the top-level goals of the system, the loss states (Table I) and their respective hazards (Table II) are defined to determine the boundary of the system.

For the identified system-level hazards, a set of safety constraints can be expressed using *inverted conditions* and *conditional statements*, in case hazards should be enforced, or how the system must prevent or minimize losses in case the hazard does occur [40]. Inverted conditions are expressed as: <System>&<Condition_to_Enforce>.
On their side, the conditional statements are stated as: if <Hazard> occurs, then <What needs to be done to prevent or minimize loss>.

2) Define the hierarchical control structure: STAMP treats system safety (and other systemic qualities) as a dynamic control problem.

A section of the hierarchical control diagram of the Digital Assets Custody system is provided. Figure 4 illustrates the system during account setup and normal state of operation.

Control diagrams are structured in the following manner: controllers issue commands (directly or through actuators) to their controlled process, and receive feedback from them (directly or through sensors). In Figure 4, **A.1** (*Submit Authentication Transaction*) is an input from the *Human Controller* (which is the operations staff of the company) to the *Distributed Ledger* (which is a controlled process). **A-FB.1** (*Log of Distributed Ledger Transactions with The Recorded Authentication Transaction*) is the feedback that would be provided from the *Distributed Ledger* to the *Human Controller*. **A.1** and **A-FB.1** are instances of the **A** and **A-FB** shown in Figure 4 respectively, there can be many other control actions and feedback between *Human Controller* and *Distributed Ledger*.

3) Identify the UCAs: Regarding control action **A.1**, there are 4 types of UCAs that can occur, as mentioned in Section III. Table IV illustrates when **A.1** can be hazardous if *provided*, *not provided*, or *provided too soon/too late*. Since this action is discrete (either an authentication transaction is submitted or not), not a continuous action, the last category of UCAs (*applied for tool long/too short*) does not apply.

4) Identify the loss scenarios: Table V provides a list of causal scenarios that have been developed based on the unsafe control action (**UCA.101**). These scenarios can be developed based on various factors, e.g., wrong process model [6], [37].

TABLE II
LIST OF SYSTEM HAZARDS

ID	System Hazards	Traceability to Losses
H.1	Lose existing certifications.	L.2, L.3
H.2	Required certification not obtained.	L.2, L.3
H.3	Applied laws and regulations are violated.	L.2, L.3, L.4
H.4	Unauthorized (potential/actual) access to the clients' DAs that are under the custodian's custody.	L.1, L.3, L.5
H.5	Unauthorized (potential/actual) access to the custodian's assets.	L.2, L.3
H.6	The (clients'/custodian's) assets are transferred to an unintended destination.	L.1/L.2, A.3
H.7	Loss of authorized access to the (clients'/custodian's) assets.	L.1/L.2, L.3
H.8	The funds are not transferred to an intended destination within the acceptable time-frame after a valid transaction request has been approved.	L.1, L.3
H.9	Loss of transactions log and the balance of clients' digital assets under custody.	L.1, L.3, L.5
H.10	Client dissatisfied with the time and performance of the custodian's services.	L.3
H.11	Custodian's expenditure exceeds the planned amount.	L.2

TABLE III
LIST OF SAFETY CONSTRAINTS RELATED TO THE HAZARDS

Hazard ID	Safety Constraints
H.1	**SC.1:** System shall not operate outside the bounds set by the certification authority. **SC.2:** If the System loses its existing certifications, then the System shall re-obtain the certifications. **SC.3:** If the System loses its existing certifications, then the System shall disclose and reassure the client of hazards and recovery actions.
H.2	**SC.4:** System shall operate within the bounds set by the certification authority. **SC.5:** If the System has not obtained required certification, then it shall augment/modify its internal controls and processes to become compliant with certification criteria.
H.3	**SC.6:** System shall not operate out of the bounds of laws and regulations of its jurisdiction. **SC.7:** If the System has violated its applied laws and regulations, then it shall modify and correct its internal controls and processes to be regulatory compliant, if it can be done within acceptable time frame; or **SC.8:** If the System has violated its applied laws and regulations, then it shall relegate custody of clients' DA, if regulatory compliance cannot be restored within acceptable time frame.
H.4	**SC.9:** Keys to assets shall only be shared with authorized parties. **SC.10:** Unauthorized parties shall not know who has the means for authorized access (keys) to client assets. **SC.11:** Unauthorized parties shall not (be able to) gain access to the clients' assets. **SC.12:** If there is an unauthorized access to client assets, the clients' DA shall be secured by re-establishing secure access and elimination of unsecure access. **SC.13:** System shall verify authorized access of the transaction requester.
H.5	**SC.14:** System shall prevent unauthorized parties from gaining access to custodian assets. **SC.15:** If unauthorized parties gain access to custodian assets, the custodian shall minimize the impact of loss on client assets. **SC.16:** If unauthorized parties gain access to custodian assets, they should not be able to reverse engineer custodian trade secrets (critical information about the custody service process). **SC.17:** If unauthorized parties gain access to custodian assets, the custodian should retrieve and secure the asset.
H.6	**SC.18:** System shall not allow the custodian to transfer (client/custodian) assets to a destination not intended (by the client/custodian).
H.7	**SC.19:** The means for authorized access to the assets shall not be forgot/lost by the authorized party. **SC.20:** : If the authorized access to the assets is lost, then it shall be recovered by an authorized party.
H.8	**SC.21:** System shall not allow funds to be transferred to an intended destination outside an acceptable time frame after a valid Transaction (Tx) request has been approved.
H.9	**SC.22:** System shall not lose the Tx log and the current balance of client assets under custody. **SC.23:** If the Tx log and the current balance of client digital assets have been lost, the Tx log and current balance of client assets under custody shall be recovered by the custodian.
H.10	**SC.24:** System shall perform in a way that ensures satisfaction of clients from custodian services. **SC.25:** If clients become dissatisfied with the time and performance of custodian services, the custodian shall improve the time and performance of its services to restore satisfaction.
H.11	**SC.26:** System shall not incur excessive costs as a result of re-doing normal operational tasks.

TABLE IV
SAMPLE UNSAFE CONTROL ACTIONS (UCAS) IDENTIFIED FOR THE CONTROL ACTION **A1** (*Submit Authentication Transaction*).

Control Action	Provided	Not Provided	Provided Too Soon/Too Late	Applied for Too Long /Too Short
A1: Submit Authentication Transaction	**UCA.101:** The operator submits the Tx to the DLT with the wrong parameters (wrong source address, destination address, amount, nonce value). **[H.10]**	**UCA.102:** The operator does not submit the Tx to the DLT. **[H.8]**	**UCA.103:** The operator submits the Tx to the DLT before the integrity of the key generator has been verified by the External Auditor. **[H.10, H.11]**	–

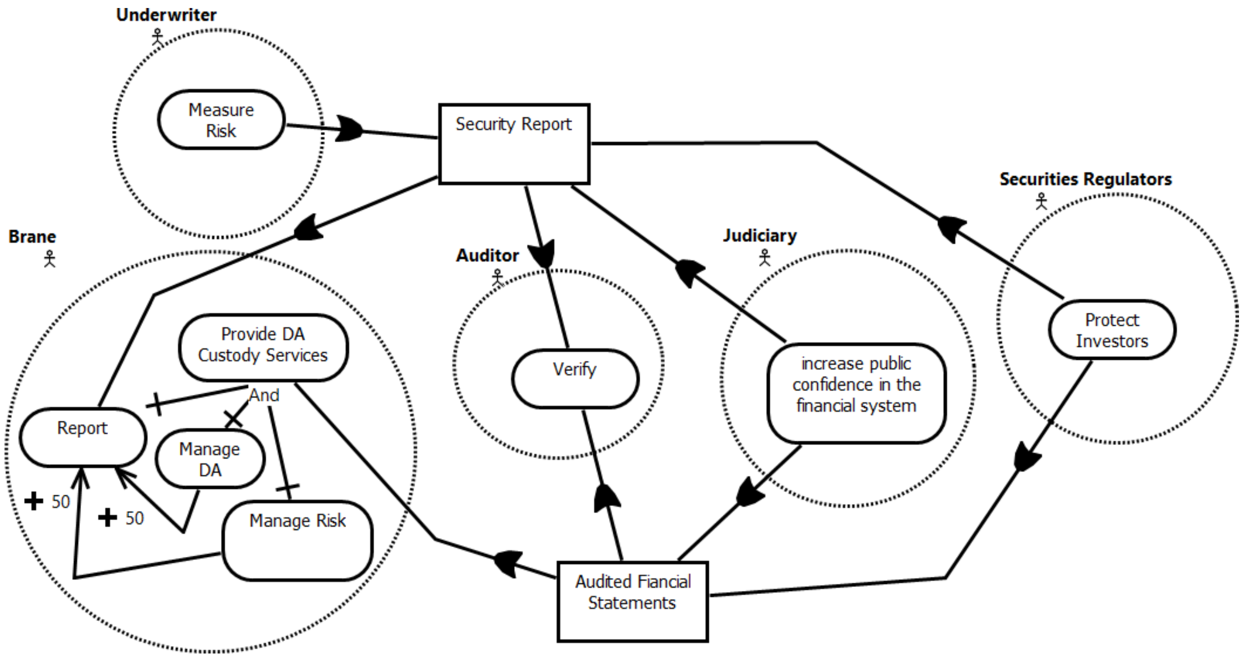

Fig. 2. A slice of the FinTech system's strategic dependency model during its operation, specified in GRL. Note that the criticality of a resource is proportional to its involvement in dependencies as a dependum.

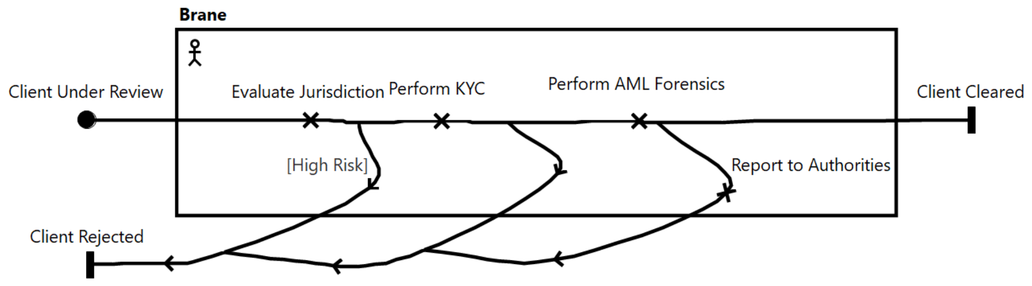

Fig. 3. Extract of the UCM model that describes the Perform Due Diligence goal, which is a subgoal of the Provide DA Custody Service in Figure 2.

TABLE V
LIST OF CAUSAL SCENARIOS FOR **UCA.101** (*The operator submits transaction to the DLT with the wrong parameters.*)

ID	Causal Scenarios
CS.1	Operator believes that she has input the right parameters.
CS.2	Operator has received wrong input parameters.
CS.3	Operator believes she knows how to create, sign and submit Txs to DLT but creates and signs Txs with wrong parameters.

5) Elicit safety constraints and requirements: Using the causal scenarios that were defined in Table V, design recommendations (Table VI) and safety constraints can be made (Table III). These design recommendations then can be further refined into requirements (Table VII).

C. Step-5: Assurance Cases

Based on the previous steps, a URN-based assurance case is presented in Figures 5 and 6. Our approach to assurance cases is that they should focus on the risks and shortcomings rather than trying to make the argumentation that the system is free of hazards. As many system safety engineers would concur, it is not possible to know *all risks* to the system in advance. Therefore, arguing that *all risks* to the system have been mitigated lacks merit. The GRL representations of Figures 5 and 6 show the dialectic form of the safety case pattern "all hazards are mitigated", which affords the reader a means of assessing the graph for the quality of indefeasibility and therefore of confidence.

The elements have different stereotypes that are defined between << and >>. The links are also *make* contribution links, i.e., if hazard **H.9** occurs, loss **L.3** might occur and does not require any contributions from other hazards.

TABLE VI
LIST OF SAFETY CONSTRAINTS AND DESIGN RECOMMENDATIONS FOR **UCA.101**

ID	Safety Constraint/Design Recommendation
SC.101	The operator must not submit Tx to the DLT with the wrong parameters (wrong source address, destination address, amount, nonce value).
DR.101.1	System should require confirmation of parameters before submitting a Tx.
DR.101.2	System/process should minimize the number of times parameters are entered manually.
DR.101.3	Training of operators should include information about what are the mechanics of DLTs and how they can create, sign, submit & verify authentication Tx to the DLT.
DR.101.4	The operators' manual has instructions on how to create, sign, submit, & verify the authentication Tx to the DLT.
DR.101.5	System shall instruct operator of the next step of the procedure after inputting each parameter.

TABLE VII
LIST OF REQUIREMENTS ELICITED FROM TABLE VI

Requirement	Traceability to DRs	Verification Method
R.101.1: System shall display given transaction Tx parameters (source/destination address, amount, currency, nonce value) to user before the final submission of the Tx to the DLT.	DR101.1	Demonstration
R.101.2: System shall prefill the input parameters for Auth Tx creation and signing.	DR.101.2	Demonstrations, Black-box testing
R.101.3: System manual shall contain the steps for creating, signing, submitting and verifying the authentication Tx to the DLT.	DR.101.4	Demonstration
R.101.4: System shall instruct operator of what is the next step of the procedure after inputting each parameter.	DR.101.5	Scenario-testing

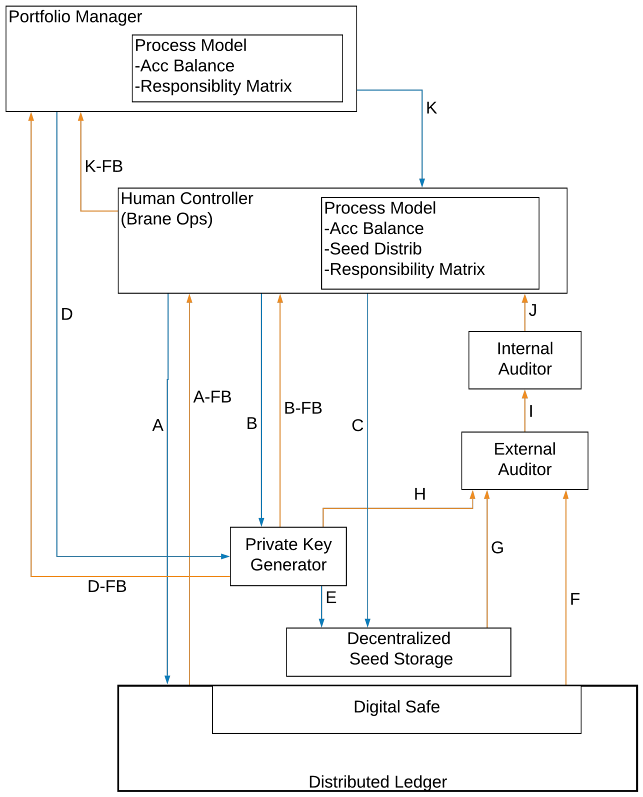

Fig. 4. The control diagram of the Digital Assets Custody system during account setup and normal state of operation

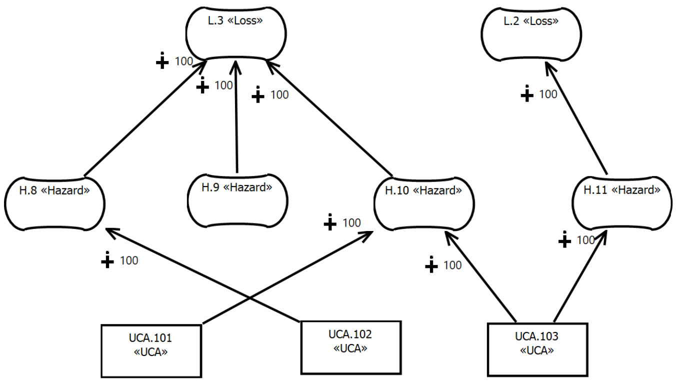

Fig. 5. The assurance case that presents the artifacts from losses until UCAs

V. DISCUSSION

An important challenge in the certification of a FinTech system is to design the system with safety and privacy in mind and communicate the results to the regulators in a way that enables them to efficiently and effectively evaluate the system. The use of STPA provides a systematic approach to design for safety and privacy (or any other systemic property) as well as a clear traceability between hazards and design. An assurance case presents the results of STPA to the regulators more effectively than simple tables by leveraging the traceability provided by STPA.

We learned many lessons by applying our proposed guidelines. First, design and certification of such socio-technical systems require frequent meetings with various stakeholders, especially regulators, to better operationalize their goals and requirements.

Our guidelines have also mainly considered privacy and safety. Security as well as many ethical considerations, including fairness, lack of discrimination, and lack of bias, are also systemic qualities. One could argue that the same approach can be used to design and certify more ethical socio-technical systems.

System developers were able to introduce controls to the systems that were explicitly traceable to the hazards that they

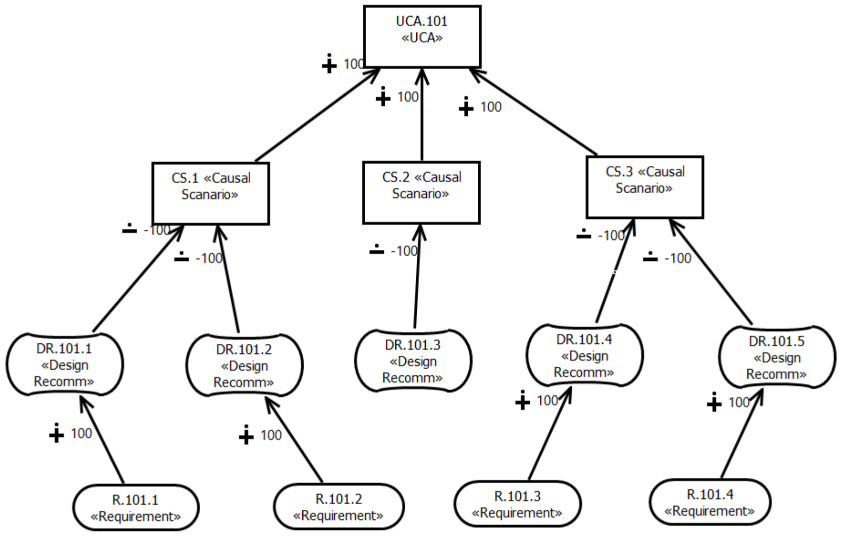

Fig. 6. The rest of the assurance case presented in Figure 5 from UCAs

were trying to address. Furthermore, these controls (stated as requirements, see Table VII) have a verification method associated with them that addresses the need for specifying control verification methods in [8].

Despite the fact that the current work was motivated by a FinTech certification problem, we believe that the proposed guidelines could potentially be applied to socio-technical systems in other domains such as healthcare and public administration while adapting the implementation of the steps to the context of the domain.

At the same time, there are important limitations to our work:

- The guidelines were applied to one system only.
- All steps were done manually by the authors, possibly introducing bias. However, we expect some parts to be automatable in the future.
- Much of the artifacts produced by the research contain confidential information that cannot be publicly shared; this negatively impacts scrutiny in this paper.
- Although many stakeholders involved in the project, such as the management and the auditors, confirmed informally that this approach is clearer than the previous certification audits and reporting methods, additional data and formal assessments are needed to confirm the scale of cost, time, and thoroughness improvement resulting from these guidelines.
- Many references cited in this paper provide detailed advice on how to perform the steps, on the pitfalls of each step, and on how to evaluate models' completeness and correctness. However, there might be further adjustments needed by such generic advice for FinTech systems.

VI. Conclusions and Future Work

This paper partially addressed several challenges related to certifying novel FinTech systems, especially the ones employing complex technologies such as DLTs. A set of guidelines for developing the requirements and design of a FinTech system that addresses the systemic qualities required by certifications is provided. These guidelines are based on goal-based modeling, process modeling, and systemic safety analysis practices. They also help conveying the results to non-technical stakeholders. The guidelines were illustrated using a commercial FinTech system.

For future work, fundamental concepts could be integrated in one tool to provide a streamlined approach from goal modeling, through hazard analysis and assurance case development. A URN profile [41] can be provided that supports the safety concepts as well as the assurance concepts while being enriched by their respective (formal) semantics [42], [43].

Additionally, the guidelines could be applied to more case studies considering other systemic qualities such as availability, privacy, and fairness. Common generalization strategies for multiple case studies could be used along the way [44]. Such case studies would improve the guidelines and would also provide data on the effectiveness of the approach on improving the certification process.

Acknowledgment

This work was supported by a Mitacs Accelerate research project titled "*Goal-oriented Safety-Guided Design and Assurance for FinTech*", and by Brane Capital.

References

[1] M. Rauchs, A. Blandin, K. Klein, G. C. Pieters, M. Recanatini, and B. Z. Zhang, *2nd global cryptoasset benchmarking study*. Cambridge Centre for Alternative Finance (CCAF), 2018.

[2] M. Haentjens, T. de Graaf, and I. Kokorin, "The Failed Hopes of Disintermediation: Crypto-Custodian Insolvency, Legal Risks and How to Avoid Them," *Singapore Journal of Legal Studies*, vol. September, pp. 526–563, 2020.

[3] Office of the Comptroller of the Currency, "Custody services – comptroller's handbook, USA," 2002.

[4] S. Sharifi, P. McLaughlin, D. Amyot, and J. Mylopoulos, "Goal modeling for fintech certification," in *Proceedings of the Thirteenth International iStar Workshop co-located with 28th IEEE International Requirements Engineering Conference (RE 2020)*, ser. CEUR Workshop Proceedings, R. S. S. Guizzardi and G. Mussbacher, Eds., vol. 2641. CEUR-WS.org, 2020, pp. 73–78. [Online]. Available: http://ceur-ws.org/Vol-2641/paper_13.pdf

[5] R. J. Wieringa, *Design science methodology for information systems and software engineering*. Springer, 2014.

[6] N. G. Leveson, *Engineering a safer world: Systems thinking applied to safety*. Boston, USA: The MIT Press, 2016.

[7] M. Hulshof and M. Daneva, "Benefits and challenges in information security certification – a systematic literature review," in *Business Modeling and Software Design*, B. Shishkov, Ed. Cham: Springer International Publishing, 2021, pp. 154–169.

[8] American Institute of CPAs (AICPA), "System and organization controls: SOC suite of services," https://www.aicpa.org/soc, 2021.

[9] ——, "Trust services principles and criteria (TSC) for security, availability, processing integrity and confidentiality," https://bit.ly/TSC-2017-2020, 2020.

[10] S. Psaila, "Perspective: Cryptocurrency Security Standard (CCSS)," https://bit.ly/3eJuYav, 2019.

[11] J. Horkoff *et al.*, "Goal-oriented requirements engineering: an extended systematic mapping study," *Requirements Engineering*, vol. 24, no. 2, pp. 133–160, 2019.

[12] ITU-T, "Recommendation Z.151 (10/18): User Requirements Notation (URN) - Language Definition," https://www.itu.int/rec/T-REC-Z.151/en, 2018.

[13] D. Amyot, O. Akhigbe, M. Baslyman, S. Ghanavati, M. Ghasemi, J. Hassine, L. Lessard, G. Mussbacher, K. Shen, and E. Yu, "Combining goal modelling with business process modelling two decades of experience with the User Requirements Notation standard," *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.*, vol. 17, 2022. [Online]. Available: https://doi.org/10.18417/emisa.17.2

[14] O. Akhigbe, D. Amyot, and G. Richards, "A systematic literature mapping of goal and non-goal modelling methods for legal and regulatory compliance," *Requirements Engineering*, vol. 24, no. 4, pp. 459–481, 2019.

[15] O. Akhigbe, D. Amyot, G. Richards, and L. Lessard, "GoRIM: a model-driven method for enhancing regulatory intelligence," *Software and Systems Modeling*, pp. 1–29, 2021.

[16] J. M. Sayers, B. E. Feighery, and M. T. Span, "A STPA-Sec case study: Eliciting early security requirements for a small unmanned aerial system," in *2020 IEEE Systems Security Symposium (SSS)*, 2020, pp. 1–8.

[17] I. Friedberg, K. McLaughlin, P. Smith, D. Laverty, and S. Sezer, "STPA-SafeSec: Safety and security analysis for cyber-physical systems," *Journal of Information Security and Applications*, vol. 34, pp. 183–196, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212616300850

[18] D. J. Rinehart, J. C. Knight, and J. Rowanhill, "Current practices in constructing and evaluating assurance cases with applications to aviation," NASA, Tech. Rep. CR2015-218678, 2015, https://ntrs.nasa.gov/search.jsp?R=20150002819.

[19] SCSC, "Goal Structuring Notation Community Standard (Version 2)," 2018, https://scsc.uk/scsc-141B.

[20] R. Feodoroff, "URN in place of GSN - Design Rationale versus Assurance Argument," 2016.

[21] ——, "Intentional enterprise architecture," in *IEEE Systems Conference (SysCon)*. USA: IEEE, 2016, pp. 1–8.

[22] ——, "Back to the Future – Pollock versus Toulmin," 2018.

[23] R. Bloomfield and J. Rushby, "Assurance 2.0: A manifesto," 2020.

[24] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis: The CORAS Approach*, 1st ed. Springer, 2010.

[26] J. Thomas, "Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis," Ph.D.

[25] R. Kumar and M. Stoelinga, "Quantitative security and safety analysis with attack-fault trees," in *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, 2017, pp. 25–32.

dissertation, Engineering Systems Division, 2013. [Online]. Available: http://hdl.handle.net/1721.1/81055

[27] C. Bensaci, Y. Zennir, D. Pomorski, F. Innal, Y. Liu, and C. Tolba, "Stpa and bowtie risk analysis study for centralized and hierarchical control architectures comparison," *Alexandria Engineering Journal*, vol. 59, no. 5, pp. 3799–3816, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110016820303045

[28] I. Bate, "Systematic approaches to understanding and evaluating design trade-offs," *J. Syst. Softw.*, vol. 81, no. 8, p. 1253–1271, aug 2008. [Online]. Available: https://doi.org/10.1016/j.jss.2007.10.032

[29] W. Wu and T. Kelly, "Safety tactics for software architecture design," in *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, 2004, pp. 368–375 vol.1.

[30] C. M. Hirata and S. Nadjm-Tehrani, "Combining GSN and STPA for safety arguments," in *Computer Safety, Reliability, and Security - SAFECOMP 2019 Workshops*, ser. Lecture Notes in Computer Science, vol. 11699. Springer, 2019, pp. 5–15. [Online]. Available: https://doi.org/10.1007/978-3-030-26250-1_1

[31] I. Habli, W. Wu, K. Attwood, and T. Kelly, "Extending argumentation to goal-oriented requirements engineering," in *Advances in Conceptual Modeling - Foundations and Applications, ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM,SeCoGIS*, ser. Lecture Notes in Computer Science, vol. 4802. Springer, 2007, pp. 306–316. [Online]. Available: https://doi.org/10.1007/978-3-540-76292-8_36

[32] T. P. Kelly and J. A. McDermid, "A systematic approach to safety case maintenance," *Reliab. Eng. Syst. Saf.*, vol. 71, no. 3, pp. 271–284, 2001. [Online]. Available: https://doi.org/10.1016/S0951-8320(00)00079-X

[33] R. Bloomfield and J. Rushby, "Assurance 2.0," *CoRR*, vol. abs/2004.10474, 2020. [Online]. Available: https://arxiv.org/abs/2004.10474

[34] J. Vilela, C. Silva, J. Castro, L. E. G. Martins, and T. Gorschek, "SARSSi*: a safety requirements specification method based on STAMP/STPA and i* language," in *Anais do I Brazilian Workshop on Large-scale Critical Systems*. Porto Alegre, RS, Brasil: SBC, 2019, pp. 17–24. [Online]. Available: https://sol.sbc.org.br/index.php/bware/article/view/7504

[35] R. Bloomfield, G. Fletcher, H. Khlaaf, L. Hinde, and P. Ryan, "Safety case templates for autonomous systems," 2021. [Online]. Available: https://arxiv.org/abs/2102.02625

[36] X. Franch, G. Grau, E. Mayol, C. Quer, C. Ayala, C. Cares, F. Navarrete, M. Haya, and P. Botella, "Systematic construction of i* strategic dependency models for socio-technical systems," *International Journal of Software Engineering and Knowledge Engineering*, vol. 17, no. 01, pp. 79–106, 2007.

[37] N. G. Leveson and J. P. Thomas, *STPA handbook*. Cambridge, MA, USA: PSASS, 2018. [Online]. Available: https://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf

[38] N. G. Leveson, "The use of safety cases in certification and regulation," MIT, USA, Tech. Rep. ESD Working Paper ESD-WP-2011-13, 2011, https://dspace.mit.edu/handle/1721.1/102833.

[39] G. Mussbacher and D. Amyot, "Goal and scenario modeling, analysis, and transformation with jUCMNav," in *2009 31st ICSE - Companion Volume*. USA: IEEE CS, 2009, pp. 431–432.

[40] J. M. Rising and N. G. Leveson, "Systems-theoretic process analysis of space launch vehicles," *Journal of Space Safety Engineering*, vol. 5, no. 3-4, pp. 153–183, 2018.

[41] D. Amyot, J. Horkoff, D. Gross, and G. Mussbacher, "A lightweight GRL profile for *i** modeling," in *Advances in Conceptual Modeling - Challenging Perspectives*, C. A. Heuser and G. Pernul, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 254–264.

[42] J. P. Thomas IV, "Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis," Ph.D. dissertation, MIT, USA, 2013.

[43] ISO/IEC/IEEE, "15026-1:2019, Systems and software engineering – Systems and software assurance – Part 1: Concepts and vocabulary," 2019.

[44] R. Wieringa and M. Daneva, "Six strategies for generalizing software engineering theories," *Science of Computer Programming*, vol. 101, pp. 136–152, 2015, towards general theories of software engineering. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167642314005450