Res Eng Design (2013) 24:375–394
DOI 10.1007/s00163-013-0156-2

ORIGINAL PAPER

Common cause failure analysis of cyber–physical systems situated in constructed environments

Seppo Sierla · Bryan M. O'Halloran · Tommi Karhela · Nikolaos Papakonstantinou · Irem Y. Tumer

Received: 5 November 2012 / Revised: 4 March 2013 / Accepted: 31 May 2013 / Published online: 22 June 2013

Abstract While cyber–physical system sciences are developing methods for studying reliability that span domains such as mechanics, electronics and control, there remains a lack of methods for investigating the impact of the environment on the system. External conditions such as flooding, fire or toxic gas may damage equipment and failing to foresee such possibilities will result in invalid worst-case estimates of the safety and reliability of the system. Even if single component failures are anticipated, abnormal environmental conditions may result in common cause failures that cripple the system. This paper proposes a framework for modeling interactions between a cyber–physical system and its environment. The framework is limited to environments consisting of spaces with clear physical boundaries, such as power plants, buildings, mines and urban underground infrastructures. The purpose of the framework is to support simulation-based risk analysis of an initiating event such as an equipment failure or flooding. The functional failure identification and propagation (FFIP) framework is extended for this purpose, so that the simulation is able to detect component failures arising from abnormal environmental conditions and vice versa: Flooding could be caused by a failure in a pipe or valve component. As abnormal flow states propagate through the system and its environment, the goal of the simulation is to identify the system-wide cumulative effect of the initiating event and any related common cause failure scenario. FFIP determines this effect in terms of degradation or loss of the functionality of the system. The method is demonstrated with a nuclear reactor's redundant coolant supply system.

Keywords Cyber–physical systems · Common cause failures · Constructed environment

S. Sierla · N. Papakonstantinou
Department of Automation and Systems Technology, Aalto University, P.O. Box 15500, 00076 Espoo, Finland
e-mail: nikolaos.papakonstantinou@aalto.fi

B. M. O'Halloran · I. Y. Tumer (✉)
School of Mechanical, Industrial, and Manufacturing Engineering, Oregon State University, 204 Rogers Hall, Corvallis, OR 97331, USA
e-mail: irem.tumer@oregonstate.edu

T. Karhela
VTT Technical Research Centre of Finland, P.O. Box 1000, 02044 Espoo, Finland
e-mail: Tommi.Karhela@vtt.fi

1 Introduction

As increasingly complex and software-intensive cyber–physical systems are being deployed into constructed environments, a new class of hazards emerges from the interactions of such systems and the environment. For example, if a pipeline leaks, the resulting flooding may damage both cyber and physical components, resulting in degradation or loss of system functions. This might lead to loss of production or harm to humans; the former case involves reliability while the latter involves safety as well as reliability. The damage caused by the flood depends on the layout of the environment: The network of rooms, corridors, tunnels or shafts will govern what components are under risk of being flooded if a pipeline in a specific location leaks. Numerous other examples can be thought of as follows: explosions, impacts from heavy objects or plumes of toxic, radioactive or flammable gases can damage components either in the immediate vicinity of the hazardous event or at a distance. Such events may damage both the cyber–physical system and its environment, for example, by

breaching physical safety barriers or by disabling automatic safety functions; a safety function is defined as a function that can prevent or mitigate harm. There is a lack of methodology for identifying and analyzing hazards arising from the interaction of the cyber–physical system and abnormal conditions in its environment.

Traditional safety analysis methods, such as failure modes and effects analysis (FMEA) (Stamatis 2003), fault tree analysis (FTA) (Vesely 1987) and probabilistic risk assessment (PRA) (Stewart and Melchers 1997), rely on human expertise and historical data to identify hazards, and they are best suited for studying direct consequences of hazardous events. Complex hazard scenarios arising from the interaction of the cyber–physical system and its environment cannot be satisfactorily analyzed with available methods; the lack of methodology becomes even more glaring when dynamic changes to the environment need to be modeled as a part of the hazard scenario (Banerjee et al. 2012b). An example of such a dynamic change is the propagation of flood or gas through rooms and tunnels; another example is the collapse of physical structures due to explosion or impact. Natural disasters, accidents and terrorism are potential initiating events for common cause failures, in which several components of the cyber–physical system as well as physical structures of the environment are damaged. As critical infrastructures and operations of society are relying on increasingly software-intensive, distributed and pervasive cyber–physical systems, questions arise concerning the outcome of initiating events such as floods, airplane crashes or just simple equipment failures.

In order to develop methodologies for answering these questions, modeling the interactions between the cyber–physical system and its environment is a prerequisite (Banerjee et al. 2012b). However, this can lead to unmanageable complexity, if a dynamic high-fidelity 3D model of the environment is interfaced to a model of a large-scale cyber–physical system and studied at real-time. Even if this were to be accomplished, there is an unlimited number of ways in which explosions or impacts may alter physical structures. Therefore, as traditional safety analysis methods have relied on human expertise, it is proposed that new safety analysis methods relying on analyzing or simulating models must incorporate human safety analysis expertise to capture the essential interactions between the cyber–physical system and its environment.

The research goal of this paper is to propose a simulation-based method for identifying and analyzing common cause failures of cyber–physical systems situated in constructed environments. The term common cause has a different meaning in quality engineering and in reliability and safety engineering. In reliability and safety, a common cause failure is defined as the simultaneous failure of several components due to a single event, which might be an environmental disturbance, human error or a component failure that causes other components to fail (Vaurio 1998). Identification of common cause failures is especially important in applications with high safety or reliability requirements, since such failures can result in the failure of several redundant systems (Kancev and Cepin 2012).

In order to keep the modeling complexity feasible while retaining essential information, a second research goal is to define a systematic procedure for incorporating human safety analysis expertise into the simulation models. An assumption is that qualified safety analysts are able to consider hazardous events and identify their local and direct consequences on the cyber–physical system and the structures in the environment. However, they are not expected to identify failure propagation, common cause failures or sequences of interactions between the system and its environment. As the discovery of unanticipated hazard scenarios often implies major changes in the design, it is of considerable financial benefit to obtain this knowledge as early as possible. Accordingly, the proposed methodology targets the early design phase.

In this research, a constructed environment is defined as consisting of spaces with clear physical boundaries. An example of an environment that is not in scope of this definition is an overhead power transmission network, if trees have been cut and ground has been leveled under the power lines. Environments that are in scope include nuclear, conventional and hydraulic power plants, factories handling hazardous materials, buildings and ships. In all of these environments, increasingly complex, ubiquitous and safety critical cyber–physical systems are already being deployed, even though considerable uncertainty persists regarding their safety and reliability in the event of natural disasters, accidents or terrorism. Other constructed environments satisfying our definition are found underground in mines, electric power distribution systems, water supply systems and transportation systems. The safety and reliability of these systems are relying increasingly on sophisticated cyber–physical technology such as smart electric grids, automatic isolation of damaged pipelines and metro automation, so an evaluation of their robustness under abnormal environmental conditions is topical.

2 Related work

2.1 Failure propagation frameworks

The foundation of our research method is based on how failures propagate. A variety of failure propagation methods exist in the literature, and the relevant ones are discussed here. The function failure identification and propagation (FFIP) framework developed by Kurtoglu and

Tumer (2008) relates a fault propagation to the function stage of design. Krus and Grantham (2007) develop the failure propagation analysis method to capture failure propagation using a functional model for system representation. Wang and Jin (2002) take a different approach by using a Function Event Network to analyze failure propagation and assign a statistical reliability measure to each functional failure. Huang and Jin (2008) extend this work to the conceptual stress and conceptual strength interference theory (CSCSIT) where the conceptual stress is related to the energy, material, and signal flow in the functional model and conceptual strength is related to functions. While the results of this method are qualitative, the historical data are recorded for a single component failure and do not capture emergent behavior. Abbas and Vachtsevanos (2009) develop a hierarchical fault propagation method for complex systems. Hiller et al. (2004) analyze software systems and determine a method to identify the propagation and effects of software errors. They also discuss how to weigh the cost-benefit of fixing software errors. Remenyte-Prescott and Andrews (2011) generate a fault propagation technique using Petri Nets to move toward an informed diagnostic analysis. Han et al. (2005) generate a similar analysis for complex systems that uses fuzzy Petri Nets. Ness et al. (1989) develop a knowledge-based approach focused on identifying initiating failure mechanisms in avionics. A large and complete set of data is required for accurate identification and simulation of failure propagation, and often the extent of this data is not available. Augustine et al. (2012) use cognitive maps to find interactions between multiple failures. Mohamed and Zulkernine (2008) develop a component-based failure propagation framework for software systems which uses physical component connections to determine system level affects. The TRELSS (Hardiman et al. 2003) and Manchester (Kirschen et al. 2004) methods model cascading failures in power grids. While these models use a realistic rendering of the power grid to acquire detailed results, they are complex and model fidelity that is not available in early design. Simplified models such as hidden failure (Chen et al. 2005) and PSA (Anghel et al. 2007) also model power grids, but with less fidelity. However, each of these models is specifically developed for power grids. In addition, they do not relate the function of the system to the failure. Other research has focused on isolating component failures (Voas 1997), mitigating faults and errors in software (Voas 1997; Hiller et al. 2001, et al. 2002; Nassar et al. 2004), mitigating faults in both hardware and software (Wallace 2005; Ge et al. 2009) and linking product functionality to fault modes (Tumer and Stone 2003; Stock et al. 2005a, b). One major limitation between each of the previously mentioned methods is that they do not reason about function losses.

In this paper, we use the function failure identification and propagation (FFIP) framework. This is a function-based fault propagation framework used to identify functional health of a component failure mode (Kurtoglu and Tumer 2008; Kurtoglu et al. 2010). Functional health is the functional impact sustained in the event of a failure. The basic FFIP framework consists of three major elements: the system representation, simulator and reasoner. The system representation contains a functional model, capturing designer intent and a configuration flow graph (CFG). In the simulator, individual component models are built and simulated. A range of values is defined for each component to determine whether the health is nominal, degraded or lost. In addition, a lost recoverable state is defined when a component does not receive a signal as a result of an upstream component being lost. The reasoner uses a function failure logic (FFL) to update the simulation. Component health values are increased or decreased by one depending on the input signal. The relationship between functions and components allows FFIP to reason about functional health and thus can be used as an early design tool. FFIP has been further developed since its introduction to address the challenges of assessing the functional reliability of complex systems in early design. Kurtoglu et al. (2010) use a failure impact quantification method to make design decisions using the total impact over a set of critical failure scenarios. Jensen et al. (2009a, b) demonstrate how component behavioral models can be adjusted or a database of related fault causes, and symptoms can be used to account for and capture faults that propagate outside the nominal system architecture. Further, the FFIP framework has been developed to be effective in evaluating both software and hardware domains (Jensen et al. 2008; Tumer and Smidts 2010) and to be applicable to the analysis of large complex systems (Papakonstantinou et al. 2011; Sierla et al. 2012). Application to a large system introduced several modeling and analysis complexities that were detailed in (Papakonstantinou et al. 2011). FFIP has also been updated as a tool to filter out design alternatives (Papakonstantinou et al. 2012). In this paper, the basic FFIP implementation has been extended to interact with a structured environment. This change is accomplished by adding an environmental flow graph (EFG) to the system representation. The EFG captures propagation of abnormal environmental conditions, such as flood, between rooms. The interface from the CFG to the EFG propagates component failures, such as pipe leaks, to the EFG, flooding the room that contains the pipe. In the other direction, abnormal environmental conditions in a room can transition components in that room to a failure mode. This allows FFIP to model fault propagation through rooms and components.

As an alternative to FFIP, complex topology networks offer a scalable way to study the effect of perturbations in

large-scale engineered systems (Braha and Bar-Yam 2007). The FFIP framework has partially similar foundations, in the sense that graph-based representations of the system are simulated to identify the system-wide effects of failure. The difference is that the CFG in FFIP uses the functional basis to type arcs between nodes according to a taxonomy of energy, material and signal (EMS) flows (Stone and Wood 2000). Also, the nodes of the CFG are components with behavioral models, which may be first-principles simulation models of mechanical components, or software components with behavior expressed as code. The behavioral models relate flows on the incoming and outgoing arcs based on an understanding of the types of EMS flows and how the component in question affects them. Behavior is also modeled for the failure modes of each component type, which is crucial for capturing safety-related interactions over the boundary of the system and its environment. For example, if a room containing an automatic valve is flooded, the control electronics of the valve are damaged and the valve fails open or fails closed; however, if there is a pipe in the room, it can still perform its function despite the flooding.

A key result in the study of complex topology networks is the possibility to determine the sensitivity of the network to perturbations in certain nodes; the system properties that can be developed through this approach are sensitivity and robustness. The key property for identifying nodes that are vulnerable to perturbations is connectivity, and this approach has been demonstrated to scale up to very large-scale real systems (Braha and Bar-Yam 2004a, b). This paper targets systems that have high safety and reliability requirements; such systems use special designs to ensure that functions may be performed despite component failures. In order to capture these designs and their possible weaknesses, the behavior of nodes in the CFG and the type of flow in the arcs is modeled. Redundancy is used to cope with failures in single components (Kancev and Cepin 2012). Propagation of failures is prevented by physical separation, electrical isolation and functional independence (Davis and Schultze 1976). Diversity is used to prevent common cause failures from affecting several redundant systems that are not separated, isolated or independent (Kim and Han 1995). Thus, it is of crucial importance for risk analyses to identify interconnections between systems that were not intended by the designer. One type of interconnection is the coupling mechanism, which makes components in a redundant system susceptible to a common cause (Kancev and Cepin 2012). The methodology and system modeling approach that is proposed in this paper is aimed at finding interconnections between the system and its environment that were not intended by the designer and which may compromise safety and reliability.

The intended context for using the method proposed in this paper is risk analysis. Risk is a broad concept and is often used in the context of managing project risks (Fang and Marle 2012). In the context of safety and reliability engineering, risk is defined in terms of the likelihood of an accident and the severity of consequences to humans, systems and the environment (Redmill et al. 1999). Several risk assessment methods have become established in safety and reliability engineering, including fault tree analysis (FTA), probabilistic risk analysis (PRA) and hazard and operability study (HAZOP). In Sect. 7 of this paper, the usefulness of the proposed method is evaluated in the context of FTA and PRA.

2.2 Common cause failures (CCFs)

Common cause failures (CCFs) have become an increasingly important topic for complex cyber–physical systems design. This comes from the basic assumption that, in practice, failures are not statistically independent from one another (Dhillon 1978). Thus, a single failure event can be determined to have catastrophic impact across an entire system. A variety of frameworks and techniques have been developed to model CCFs. In general, these address finding ways for the system to handle the CCF, determining the impact, developing a way to reduce the likelihood (Himanen et al. 1989), the coupling or interdependencies between failures and CCF identification (Mosleh et al. 1989). In this paper, we will cover the identification of CCFs and their impact on the health of safety-related functions of the system.

Early work on fault impact by Wang et al. focuses on describing the nature of faults from the conceptual design perspective (Wang and Jin 2002). This work led to other methods that show how faults may affect the performance of components in the system (Smith and Clarkson 2005; Huang and Jin 2008; Kurtoglu and Tumer 2008). Some of these methods use qualitative descriptions of fault probability which allows them to be applicable in early design (Hata et al. 2000a; Stone et al. 2006; Grantham-Lough et al. 2009). In contrast, quantitative methods identify the likelihood of failures and evaluate failures in terms of the product's or system's ability to perform a desired functionality. O'Halloran et al. (2011) develop a framework to calculate a minimum, maximum and weighted average system reliability during functional design. This research has been further developed to calculate the functional failure rates using a hierarchical Bayesian model (O'Halloran et al. 2012). This allows the likelihood of a failure to be adjusted calculated while capturing uncertainty in the early stages of design. Hata et al. (2000a) identify failure modes by representing functions with streams and a network structure. Historical data and rules are used to determine the impact to the functional model from a defect. This method cannot handle multiple failures, dynamic

behavior and requires a large amount of input on potential failures in the design prior to the evaluation. Within this paper, we address impact by simulating component behavior models to locate fault propagation paths. The fault impact is assessed using the function failure logic (FFL) where each component is updated at each time step in the simulation. Component fault impact is then translated back to functions using the relationship between the CFG and the functional model.

While CCF methods have been developed to reduce the impact of CCFs, their major limitation is the initial identification of the CCF. During the design of a new complex system, historical data are used in most methods for identifying CCFs. Hata et al. (2000b) identify failures by representing functions with streams and a network structure. Historical data and rules are used to determine the impact to the functional model from a defect. Thibaux et al. (2005) group failures for a specific system and structured data using matrices to identify the root cause of a failure. They use historical data to filter out failures that are less relevant, prior to finding the CCF. Fault tree analysis (FTA) has also been used to identify CCFs (Dhillon and Singh 1981). FTA systematically analyzes complex systems to find a set of basic failure events that cause a top-level undesirable event. If the set of basic events cannot be further minimized, this is referred to as a minimal cut set. Summers and Raney (1999) and Summers (2000) use FTA to model CCFs by implementing the CCF as a separate branch. While FTA is useful to identify CCFs, it is tedious and time-consuming to perform and requires significant domain knowledge on how failures in the system can occur, and the consequences of a minimal cut set can only be related to the top-level event and other consequences may need to be considered (Lazzaroni et al. 2011). Common cause failure analysis (CCFA) is used to systematically identify common causes of multiple failure events (Flemming 1975; Dhillon and Proctor 1977; Ericson 2005). The formal methodology includes generating a fault tree, a basic system screening to identify dependent failures, and performing detailed calculations to determine CCFs. CCFA requires an in-depth knowledge of the CCFA analysis and fault tree analysis and requires large amounts of historical data. Databases have been constructed that contain historical data used in CCF methods, one of which is described in Wierman et al. (2007). The observations stored in this database are linked directly to components. Since historical data are configuration-specific, it cannot be used to identify a CCF for a new system during the design process without also accounting for uncertainty in the historical data. This is primarily because complex systems have significant failure interactions, and historical data are configuration-specific. In comparison with the method presented in this paper, we use behavior simulation instead of historical data and create individual component models. This allows the complexity of the cyber–physical system to increase without significantly increasing the complexity of the analysis. Relationships between the CFG and the EFG allow the system to share information with a structured environment. A formal method has been developed to systematically identify potential hazards. The simulation determines whether a hazard can result in a CCF by monitoring the functional impact.

2.3 Modeling environment during failure analysis

The research interest in this paper is to analyze CCF scenarios in which component failures in the cyber–physical system (CPS) may affect the environment and in which abnormal environmental conditions affect components. A variety of hazard simulations have been developed to address concerns about levees breaching, dams breaking, fires or smoke propagating, etc. Chen et al. (2004) develop a semi-distributed physically based hydrological model for flash floods that has parameter dependence on a basin environment. This model is simulated to capture peak discharge floods rates. Similarly, Mani and Chakravorty (2008) simulate a failure event of the Panchet and Maithon dams. They use geometric inputs from the river to determine the flood extent to the downstream environments. Castrillón et al. (2011) develop a fire simulation framework that uses extensive geographical information system data including ground vegetation type, wind, spacial and weather. Ali and Ariffin (2011) create a flood inundation model using hydrodynamics and a digital elevation model. Choi et al. (2005) determine the effect to smoke propagation from its environment. This is been specifically formulated for tunnels where the tunnel size and scalability are investigated. Xiao et al. (2008) propose a sensor network-based framework to locate and follow the propagation path of a fire. While this approach has broad application, it is proposed to study the dynamics of fire in a building and is not practical to model early designs. Hostikka and Keski-Rahkonen (2003) develop the probabilistic fire simulator (PSF) to model the environment's geometry and how a hazard propagates from the source (initial fire location) to the target (cable). While this method models the geometry of the environment, it requires detailed information to build the models. Other relatable examples include the smoothed particle hydrodynamics numerical model (Vacondio et al. 2012), the flood simulation model by Lin et al. that focuses on system architecture, a flood simulation by Liang (2010) used to handle complex ground geometry, and the CCHE2D model used to incorporate topographical data (Hossain et al. 2011). Each of these models has a large amount of location-specific data about the environment (e.g., basin or tunnel size and shape, river bed geometry, average annual flood

flow rates, etc.), resulting in high-fidelity models to facilitate detailed simulations. None of these methods capture the interactions between a CPS and its environment.

Robotics for environmental monitoring is one field in which the interaction of a cyber–physical system with its environment is of central importance. Such robots or fleets of robots are often deployed in hazardous environments such as volcanoes (Muscato et al. 2012), accident sites involving release of hazardous gases (Neumann et al. 2012), or damaged nuclear plants. Safety and reliability is identified as a key limiting factor for the widespread use of this technology (Dunbabin and Marques 2012); for example, the deployment of unmanned aircraft systems for studying the development of storms and tornadoes is constrained by regulation that requires unmanned systems to demonstrate the same level of safety as is required for manned systems (Frew et al. 2012). The research in the field of environmental robotics focuses on technology for observing the environment, avoiding obstacles (Muscato et al. 2012) and in few cases actively mitigating hazards in the environment, as in vacuuming radioactive materials in the damaged Fukushima nuclear reactor. However, safety and reliability require the consideration of all interactions between the system and its environment regardless of whether they are within the scope of the sensors and actuators of the system. This requires simulation-based risk analyses that capture interactions such as collisions of the vehicle to objects in the environment and the exposure of the vehicle to substances or forces that may damage some of its components.

More recently attempts have been made to model the interactions between CPSs and their environment. Zhang et al. (2012) use a simulation-based approach for CPSs to maintain safety standards in air traffic management systems while modeling weather patterns and 4D path planning. This method only models the flow of information between the system and environment. Jiang et al. (2011) develop a real-time virtual heart model to support verification of medical cyber–physical systems. This work captures electrophysiological operations of the heart which is used as the environment interacting with a pacemaker. The method supporting the virtual heart model cannot be used to discover hazards. The BAND-AiDe method uses a region of impact (ROIm) and region of interest (ROIn) to characterize the spacial extent of the effect for each local CPS (Banerjee et al. 2012a). A physics-based model is used to capture the physical dynamics of an initiating event (e.g., explosion). While a unique advantage of the BAND-AiDe method is the monitoring of an unconstructed environment, it requires intricate understanding of the physics governing the initiating event, and it has not been extended for risk assessment. Willems' (2007) uses the Tearing, Zooming and Linking framework as a behavior-based model where the system is represented using a black box. This method can be limited by not being able to determine the input from the environment. While this method models connections between a system and its environment, it has not been applied to risk and reliability analysis. In addition, it requires the formulation of mathematical models that may not be available during early design. While approaches to modeling hazards in various environments exist, and while a few modeling approaches integrate a CPS with its environment, no framework has been proposed for identifying hazards and analyzing risks in a CPS, in which the potentially hazardous interactions between the CPS and the environment are modeled. In this research, the environmental flow graph (EFG) formalizes the structured environment and communicates directly with the configuration flow graph (CFG). A systematic procedure proposed in this paper is used to determine the interface between the CFG and EFG to allow failures to be propagated between the two.

3 Case study

Figure 1 shows rooms of a nuclear reactor building containing components of an emergency supply system. The purpose of the system is to inject water into the reactor vessel if the surface level starts falling. There are two redundant pipelines, which pass through the rooms A and D in Fig. 1. A typical design requirement for redundant systems in nuclear reactors is that the reactor must cope with the loss of one such system, and that there must be no common cause failure that could result in the loss of more than one system. A common cause failure is defined as a failure of more than one component or structure in the cyber–physical system or its constructed environment due to a single event. The layout of the building and the allocation of process components into rooms are done carefully in order to meet the requirement related to common cause failures.

Design requirements for the emergency supply systems in Fig. 1 are as follows:

- Either of them must singly be able to maintain the reactor surface level in case of emergencies.
- There may be no common cause failure that could result in the loss of both emergency systems.

In Fig. 1, the solid wall between rooms B and C is a structure that prevents hazardous conditions in rooms A and B from reaching rooms C and D, and vice versa, so in either case, only one emergency system is in the affected area. There is no obvious common cause failure that could result in the loss of both systems.

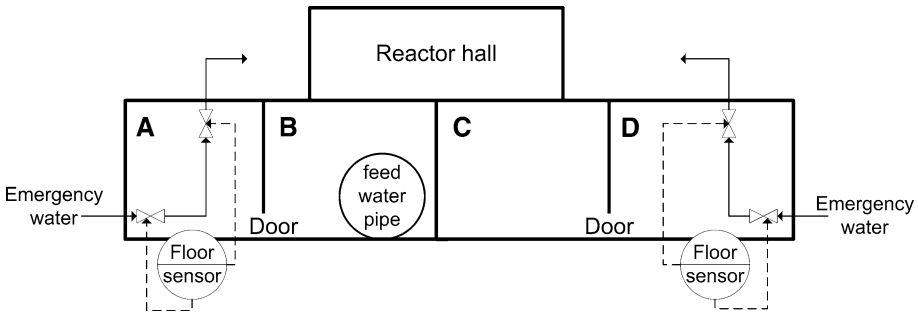

Fig. 1 Redundant emergency water supply systems in a nuclear reactor building

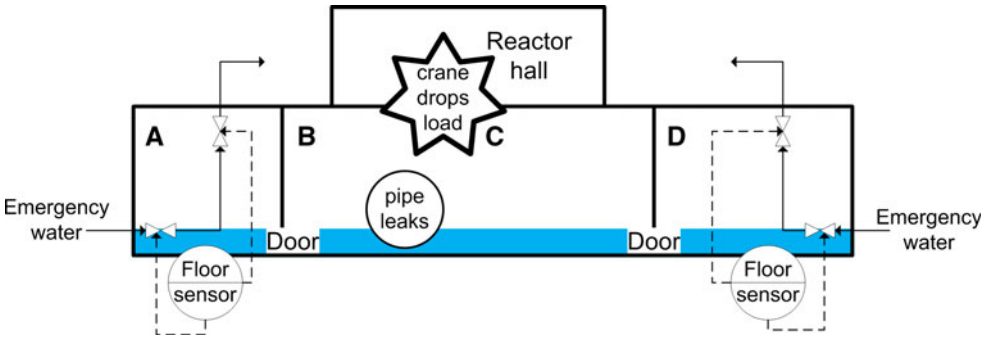

Fig. 2 A common cause failure scenario resulting in dropping surface level in the reactor

A common cause failure situation that may lead to loss of both emergency supply systems and dropping surface level in the reactor is presented in Fig. 2. In the reactor hall, there is a crane used to lift very heavy objects. If such an object is dropped, the ceiling of the rooms below may collapse. In Fig. 2, the ceiling of room B collapses and the wall between rooms B and C is breached. The collapsing ceiling damages components in room B, including the reactor's feedwater pipe, which leaks and floods the room. The flood propagates to room A through the door, to room C through the breached wall and to room D through the door; the flooding is shown as the shaded area at the bottom of Fig. 2. A flow sensor of the feedwater pipe detects the loss of feedwater and activates the emergency supply systems. However, in both rooms A and D, there is a safety function that senses a leak in the emergency supply pipeline and actuates the closing of valves that isolate the section of the pipeline located in that room. The sensing is performed by a floor sensor that detects water on the floor. The designer of the control system has mistakenly assumed that water on the floor will always be from the emergency pipeline. In this case, the flooding that originated from outside the room has spuriously activated the safety functions in both rooms A and D, disabling both emergency supply systems. Since the main feedwater pipe was damaged, the reactor surface level continues to fall, resulting in the eventual exposure and melting of fuel rods.

This example raises several questions. What kind of model can support simulation-based risk assessment for the purpose of identifying the common cause failure scenario and its impact on safety functions such as reactor surface level regulation? What is the minimum level of detail required in the model, so that the scenario can be discovered early in the design? The design contains a great quantity of information not shown in Fig. 1, so how can the model be built with all the relevant details before the scenario is anticipated? A solution to these questions is presented in the Sect. 4.

4 Methodology

In this section, the methodology is presented independently of any simulation package or other tools. The functional failure identification and propagation (FFIP) framework presented in Kurtoglu and Tumer (2008) is extended as shown in Fig. 3. The three boxes on the right belong to the original framework. A functional model capturing design intent is specified using the functional basis taxonomy of functions and their interfaces consisting of material, energy and signal flows (Hirtz et al. 2002). In order to simulate failure propagation, a model consisting of components is built: the configuration flow graph (CFG). Behavioral models of components capture the input–output relationships of components in the nominal mode and one or more failure modes. In order to study the functional effect of a component failure, a critical event is injected, which transitions the behavioral model of the selected component to the chosen failure mode. This failure then propagates throughout the simulation as abnormal flow levels. As the

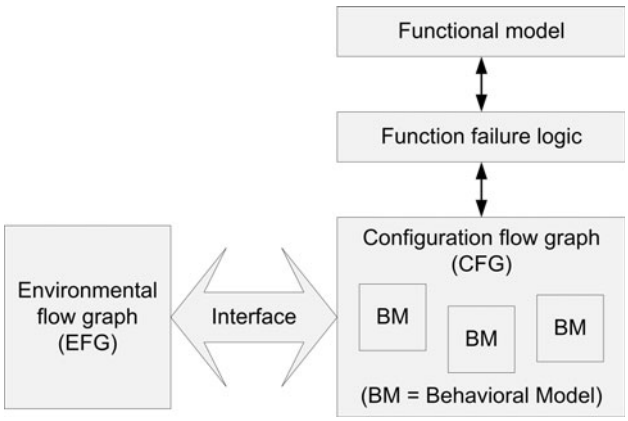

Fig. 3 Framework for failure propagation analysis

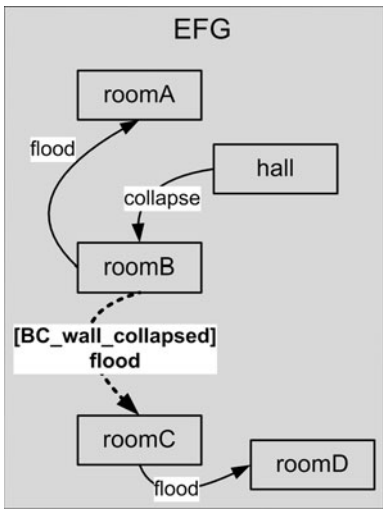

Fig. 4 Environmental flow graph (EFG) for the example in Fig. 2

CFG and functional model use the same flow levels, a function failure logic passively observes the flows in the simulated CFG and uses this information to determine whether specific functions in the functional model are degraded or lost.

The original FFIP framework describes the cyber–physical system without considering its environment. In the introduction of this paper, a constructed environment was defined as consisting of spaces with clear physical boundaries. A room is defined as any space with clear physical boundaries, such as a hall, corridor, tunnel or shaft. The environmental flow graph (EFG) in Fig. 3 is defined as a graph with a node corresponding to each room in the environment. The EFG for the example in Fig. 2 is shown in Fig. 4. The flows of abnormal environmental conditions between rooms are identified by safety engineers and expressed as arcs between rooms in the EFG. A set E is defined, consisting of all the abnormal environmental conditions that should be considered in the application domain. In this paper, $E = \{\text{flood, fire, collapse, explosion, flammableGas, toxicGas}\}$. Each arc is labeled by one element of E. If several elements of E may propagate between two rooms, separate arcs are used for each of them; for readability, Fig. 4 shows only the arcs relevant to the scenario presented in Sect. 3.

In order for a simulator to process the flows, a more detailed definition is needed. In concept figures such as Fig. 4, an arc from room X to room Y is labeled with e, which is an element of set E. In the simulator, the arc is from outport e_out of X to inport e_in of Y. An example is shown in Fig. 5, in which room B is X, room A is Y and flood is e. The concept notation is used to significantly reduce clutter in figures. The simulation constructs are displayed later when the implementation in the Simulink environment is discussed. The semantics of the arc is that the value at the source port (flood_out in Fig. 5) is written to the destination port (flood_in in Fig. 5). The values are boolean: true if the abnormal environmental condition e exists in that room at that time in the simulation and false otherwise.

Consider the barrier between rooms B and C in Fig. 1: It is a wall that prevents flooding, but an explosion or impact may open a path for flooding. At the start of the simulation, there should be no arc between these rooms, but such an arc should appear when an explosion or impact capable of destroying this barrier occurs. For this purpose, Fig. 4 has a dotted arc with the label "[BC_wall_collapsed]flood." As shown in Fig. 2, safety analysts have identified the possibility of the wall between rooms B and C collapsing as a direct consequence of the crane dropping a heavy load. When this event actually occurs during simulation, the dotted arc should appear on the EFG. There is considerable technical difficulty and computational load involved in recompiling a simulation model while the simulator is running, which is avoided by using guard conditions.

A guard is a boolean variable that is visible to all components in the simulation model consisting of the EFG and CFG. When building the simulation models, a guard is

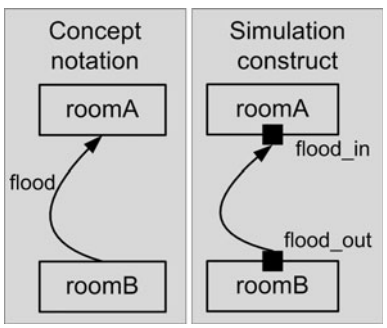

Fig. 5 Concept notation used in EFG figures (*left*). Corresponding constructs in the simulation environment, with ports depicted as *black squares* (*right*)

defined for each hazardous component failure that may alter physical boundaries between rooms. BC_wall_collapsed is defined when the crane component of the CFG is analyzed and the possibility of impact is identified. The default value of the guard is false. When the hazardous failure occurs, the behavioral model of the failed component sets the value of the guard to true. The semantics of the guard is that if its value is true, the value at the source port is written to the destination port, as described above for arcs not involving guards. If the guard is false, nothing is written and the destination port retains its default value of false. Any techniques available in the chosen simulation environment may be used to implement the guard; our implementation in the Simulink environment uses the switch block, as described in Sect. 6.

As the CFG is a graph with a node for each component, and the EFG is a graph with a node for each room, the entire simulation model is a graph consisting of all the nodes and arcs in the EFG and CFG. The arcs between rooms and components have not yet been defined. Figure 3 shows a bi-directional arrow between the CFG and EFG. The arrowhead pointing left implies dynamic changes in the environment produced by component failures. Figure 6 shows this as arcs from components in the CFG to rooms in the EFG (i.e., arcs with numbers 1 and 4). Numbers on arcs indicate the sequence of events for the scenario in Fig. 2. The numbers are not part of the models. The room at the destination of each arc corresponds to the room in which the source component is located; this information must be available before applying this methodology, but no more details on equipment positioning are required.

Formally, the interface from the CFG to the EFG is a set of 4-tuples (c, r, h, g). c is an element of C, the set of components in the CFG. r is an element of R, the set of rooms in the EFG. h is an element of H, the set of hazardous events that are relevant to this domain. In this paper, $H = \{\text{explosion, ignition, impact, waterLeak, flammableGasLeak, toxicGasLeak}\}$. g is a boolean guard in the EFG that can be set to true from the behavioral model of c, as explained earlier. An example in Fig. 6 is the feedwater pipe, which may cause the hazardous failure waterleak in room B. Each tuple corresponds to an arc from outport h_EFG of node c to inport h_CFG of node r (this is the naming convention of ports used in the implementation in this paper; any other naming convention may be used). The semantics of the arc are that the value at the source port will be written to the destination port. Values may be either true or false, and the value at the source is set true by the behavioral model of c when the hazardous failure occurs during the simulation.

The simulation model of the room processes its inputs, which are flows of abnormal environmental conditions from other rooms as well as hazardous failures of components located in that room. The model is stateless and determines the value of the output port through a logic that examines the values of the input ports. The logic that is used for all rooms in our model is shown in Fig. 7. For example, if a room receives flooding from an adjacent room (inport flood_in has value true) or if a component in the CFG located in this room leaks water (inport waterLeak_CFG has value true), then the outport flood_out has value true.

In the bi-directional "Interface" arrow in Fig. 3, the arrowhead pointing right implies component failures caused by abnormal environmental conditions in the room in which the component is located. For example, if a room is flooded, safety analysts might decide that this will not damage a pipe located in that room. However, if an explosion occurs, this would be seen as an event which has potential to damage the pipe, transitioning the component

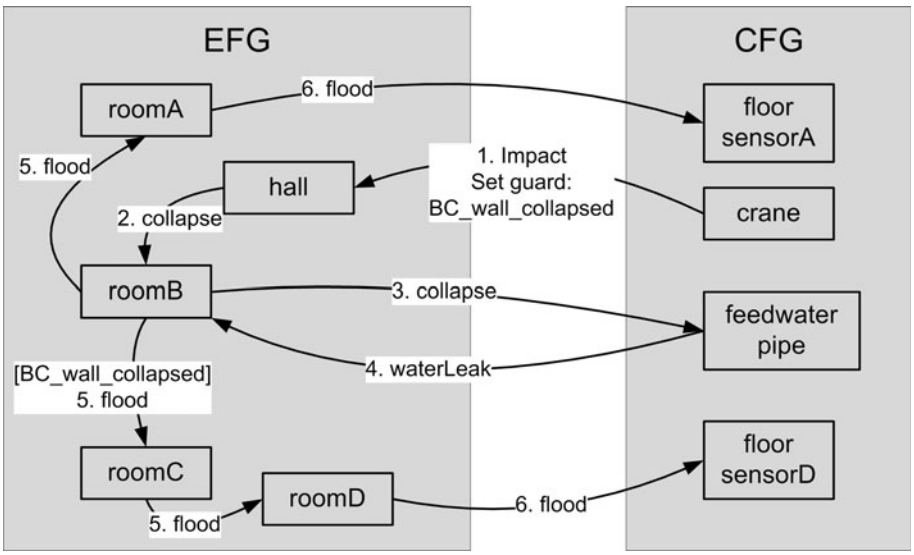

Fig. 6 Interface between the EFG and CFG. *Numbers* on *arcs* indicate the sequence of events for the scenario in Fig. 2. The *numbers* are not part of the models

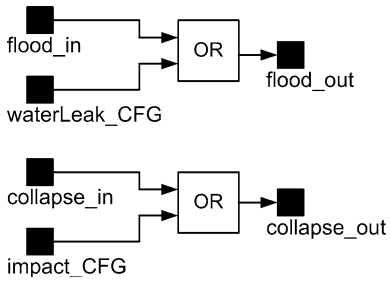

Fig. 7 Logic how a room forms its outport values based on the values of its inports. Ports depicted with *black squares*

to the leaking failure mode. Formally, the interface from the EFG to the CFG is a set of 3-tuples of the form: (r, c, e). r is an element of R, the set of rooms in the EFG. c is an element of C, the set of components in the CFG. e is an element of the set of abnormal environmental states E; in this paper, $E = \{$flood, fire, collapse, explosion, flammableGas, toxicGas$\}$. An example in Fig. 6 is room B: If the ceiling collapses, the feedwater pipe is damaged. Each tuple corresponds to an arc from outport e_out of node r to inport e_in of node c (this is the naming convention of ports used in the implementation in this paper; any other naming convention may be used). The semantics of the arc is that the value at the source port will be written to the destination port. Values may be either true or false, and the value at the source is set by the logic of the room illustrated in Fig. 7.

Equation 1 presents the system state in behavioral simulation, exactly as defined in (Kurtoglu and Tumer 2008). $X(t)$ is the system state and $c(t)$ is a vector that specifies the nominal or failed modes of each component in the simulation. This part of the equation is still sufficient for the framework presented in this paper, since the rooms that were added are stateless and do not have modes. $\mathbf{v(t)}$ is a vector of system state variables with one element for each arc in the CFG. In this paper, arcs in the EFG as well as arcs between the CFG and EFG are also system state variables and thus elements of $\mathbf{v(t)}$. Kurtoglu and Tumer (2008) specify that an element of $\mathbf{v}$ may take values from a set that can be different for each element of v. For the additional elements of $\mathbf{v}$ needed to support the framework presented in this paper, the set of possible values is {true, false}.

$$X(t) = \Theta(c(t), \mathbf{v}(\mathbf{t})) \quad (1)$$

According to the simulation principles specified in Kurtoglu and Tumer (2008), the simulation is started after initial values have been given to all elements in the vectors in Eq. 1. The continuous time system is solved in the intervals between discrete events. An injected critical event scenario is a component failure that can be inserted into the simulation at any step; these are handled by stopping the continuous time simulation and transitioning the component into a failure mode, which defines a new logic of forming output flow values in terms of input flow values. In order to study a common cause failure scenario, it would be necessary to separately specify each component failure at the correct time, so the approach in Kurtoglu and Tumer (2008) is only applicable for investigating common cause failure scenarios that are already known. The framework in this paper expects the user only to define a single initiating event, after which the simulation determines whether a common cause failure scenario occurs. Unlike (Kurtoglu and Tumer 2008), the simulation presented in this paper will independently cause component failures in response to abnormal environmental conditions and vice versa, as defined earlier for the interface between the EFG and CFG. Component mode changes resulting from these are handled in the simulator in the same was as injected critical event scenarios in Kurtoglu and Tumer (2008).

Finally, common cause failures are not interesting for their own sake but due to their potential impact on safety critical functions of the system. In order to detect degradation or loss of such functions, the original FFIP framework in Kurtoglu and Tumer (2008) uses a function failure logic (FFL), which detects changes in function health by observing and analyzing abnormal flow values in the simulator. This part of the framework, described earlier in the discussion on Fig. 3, is not changed in this paper.

The function of interest in our case study is the supply of emergency water. The FFL is coded according the functional requirements of our case study, which are presented below:

- If feedwater flow to tank is nominal or high, function must not activate
- If feedwater flow is less than nominal, both emergency supplies should be activated
- Function is considered operating if one or both emergency supplies provide a nominal or high flow to a tank
- Otherwise, function is considered degraded if one or both emergency supplies provide a low flow to a tank
- Otherwise, both supplies provide zero flow, so the function is lost

A formal specification of the FFL is presented in Fig. 8, which also displays a block diagram view of how the FFL observes flows between components in the CFG. f is the flow of water from the feedwater pipe to the tank. $e1$ and $e2$ are the flow of water to the tank from emergency pipes 1 and 2, respectively. The logic is presented in pseudocode, as this section does not assume the use of any specific programming language or simulation tool.

```
if (f=nominal OR f=high) then
  if (e1=zero AND e2=zero) then
    return operating;
  else return lost; /*spurious activation*/
  endif
endif
if (f=low OR f=zero) then
  if (e1=nominal OR e1=high OR
      e2=nominal OR e2=high) then
    return operating;
  endif;
  if (e1=low OR e2=low) then
    return degraded;
  endif;
  return lost; /*only choice left*/
endif;
```

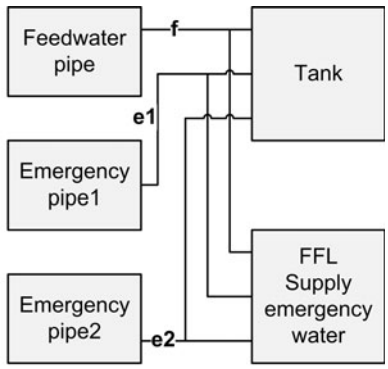

Fig. 8 Function failure logic (FFL) for the function "Supply emergency water"

5 Procedure for building the models

The methodology presented in Sect. 4 is useful for identifying unforeseen complex common cause failure scenarios only if it is possible to build the models without foresight of such scenarios. A research goal stated in the introduction is to define a systematic procedure for incorporating human safety analysis expertise into the simulation models. It is assumed that qualified safety analysts are able to consider hazardous events and identify their local and direct consequences on the cyber–physical system and the structures in the environment. They are not expected to identify failure propagation, common cause failures or sequences of interactions between the system and its environment.

The flowchart in Fig. 9 specifies the procedure that safety analysts follow to create the EFG. This workflow only considers the environment without any information of the cyber–physical system. For each room, all of the abnormal environmental conditions are considered. If the condition, such as a flood, can propagate to adjacent rooms, the corresponding arc is added to the EFG. Figure 4 shows the result of applying this procedure to the case in Fig. 1. However, Fig. 4 omits several arcs irrelevant to our scenario, such as propagation of fire and gas. The dotted arc in Fig. 4 that has a guard condition is not created in the workflow in Fig. 9; this arc is created after considering interactions of the cyber–physical system and its environment in the next workflow.

Figure 10 presents the flowchart for creating arcs representing interactions between the cyber–physical system and its environment. Dashed lines link a flowchart step to the arcs created in that step. The procedure systematically considers each room and each component in the room. For any component, each hazardous event is considered. In step 4, the safety analyst will determine whether this combination may alter physical boundaries. For example, if c is "crane" and h is "waterLeak," it is not possible to find a failure of the crane which would result in the crane leaking water into the room, so steps 4 and 5 are bypassed via the

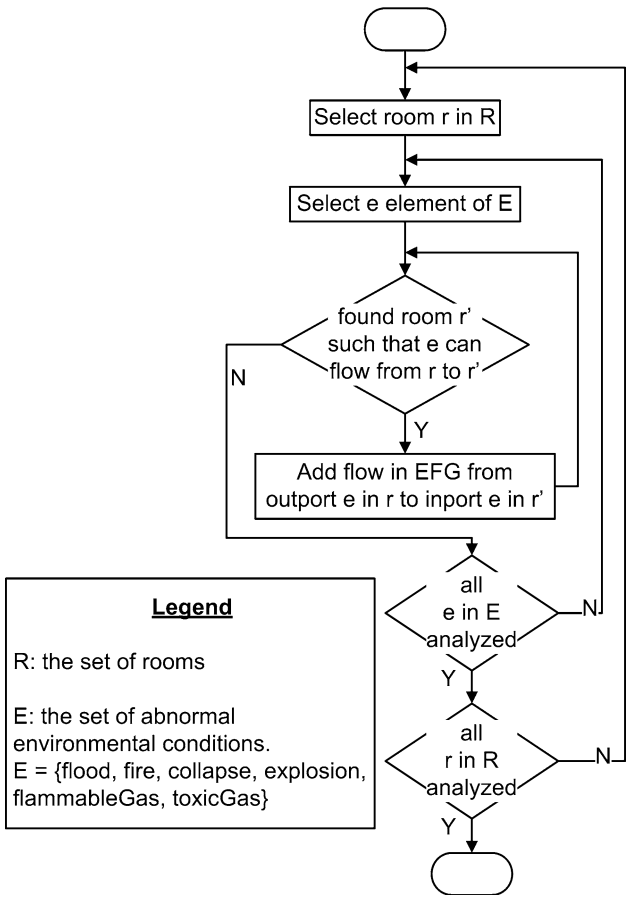

Fig. 9 Flowchart for creating the EFG. *R* is the set of rooms in the model. *E* is the set of abnormal environmental conditions

arrow labeled "*N*." Other elements of the set *H* are then considered for the same component, until it is determined that a failure of the crane may cause "impact." In step 4, safety analysts need to be aware of the kinds of loads that are moved, so that they can judge what physical barriers may be breached. In this example, since the loads are very heavy, the collapse of the wall between rooms B and C on the lower floor is considered possible. The guard condition "BC_wall_collapsed" and the arc from roomB to roomC

are created. In step 5, the arc from the crane to the hall and the arc from the feedwater pipe to roomB are created. After all hazardous events have been analyzed for a component, abnormal environmental changes affecting the behavior of the component are identified in the step 7. The dashed lines from step 7 indicate the arcs that are created in this step.

The procedures in Figs. 9 and 10 expect safety analysts to only identify immediate and local consequences of hazards without considering system-wide effects or common cause failure scenarios. The procedures may seem tedious and work intensive. However, industrially accepted risk analysis methods involve work-intensive session through long lists, and managers commissioning such analyses realize that significant costs are involved (Redmill et al. 1999). The analyst would not need to keep track of the steps and iterations of the flowchart. Based on the specifications in this publication, it is possible to develop a database application with a user interface prompting the analyst with questions such as "What abnormal environmental conditions in roomB could cause the feedwater pipe to transition to a failure mode?" From a drop-down list containing all the elements of set E, the analyst could select "collapse." In further work, the information in the database could then be used to generate the EFG and the arcs between the EFG and CFG.

6 Implementation

In this section, the framework is implemented onto a simulation platform, which nuclear safety authorities accept for the purpose of validating the design. According to Finnish nuclear safety regulations, both deterministic and probabilistic methods (PRA) should be used (STUK 2002). Deterministic methods are used in this paper; combining our approach with fault tree modeling, which serves as the starting point of PRA, is described in the Sect. 7. Deterministic methods require that the simulation is based on natural scientific theory (STUK 2002), which implies first-principles simulation models using differential equations based on the laws of physics. Continuous time simulation is nearly always used for this purpose; discrete-

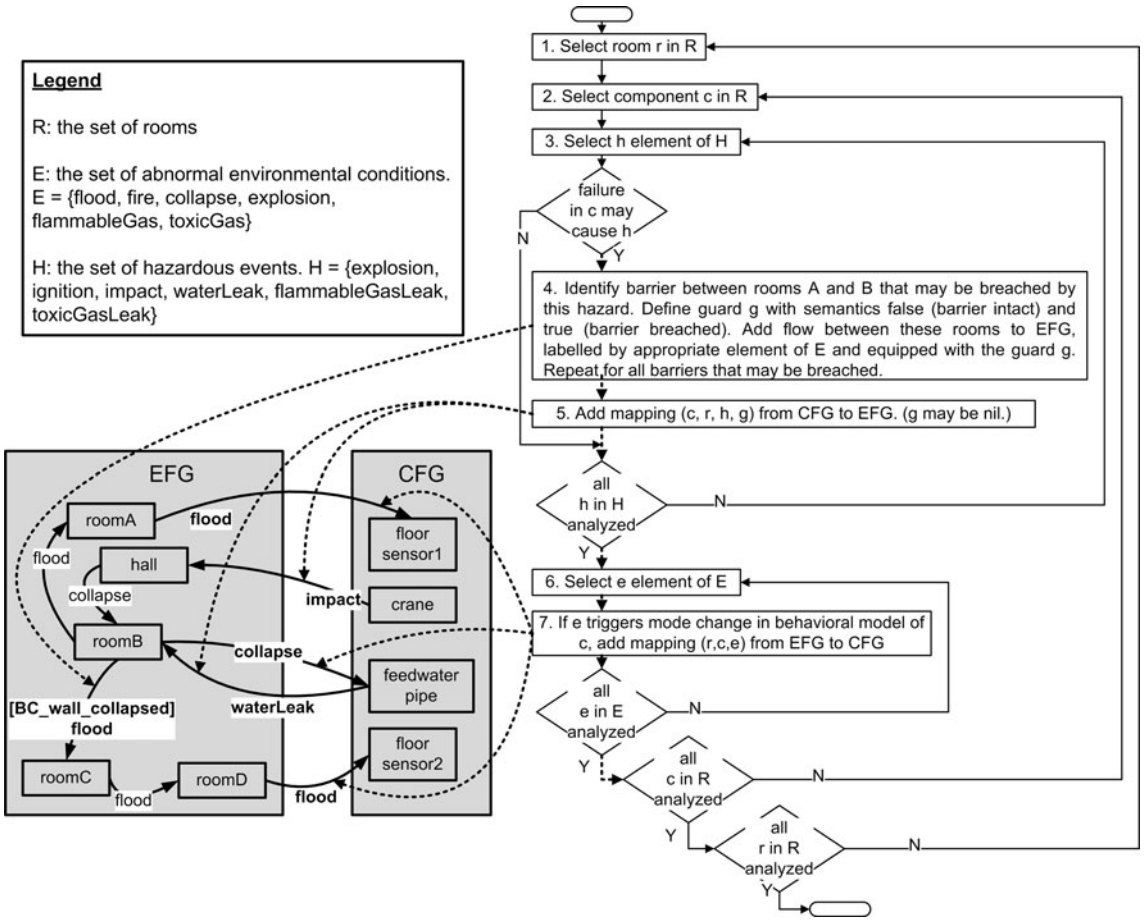

Fig. 10 Flowchart for creating arcs representing interactions between the cyber–physical system and its environment. *Dashed lines* link a flowchart step to the arcs created in that step

event-based simulation options exist (Zeigler et al. 1994), but these have not been accepted by nuclear safety authorities for validation of designs. However, our framework also includes discrete events, so a combination of discrete and continuous simulation is needed, and the issues arising from that are discussed next in this section. In this paper, the implementation is done on a first-principles simulator APROS, which has been accepted by safety authorities for validation of designs for several nuclear plants in Finland, Sweden, Russia and Hungary (VTT 2013).

The framework simulates continuous physical processes, but also includes several kinds of discrete events. Component failures such as valve closure or pipeline leak are discrete events, and the standard approach for handling them in a continuous simulation is to gradually ramp the variable in question from the current value to the desired value; for example, the aperture of the valve or the diameter of the hole in the pipe is changed continuously with a ramp so that the simulator will not fail to converge. In cyber–physical systems, the cyber parts of sensing, control and actuation involve functionality that is naturally discrete. A dynamic process simulation package consists of a thermo-hydraulic solver, material property computation and an automation solver for the control systems. The thermo-hydraulic solver is used for the flow, pressure, temperature and concentration solution, while the automation solver is used to simulate the cyber components. The EFG is a discrete model, so it is also simulated with the automation solver. In a dynamic process simulation environment, the pressure-flow solution is computed first, followed by temperature and material property calculation. The automation solver is a separate entity solving the control system in a sequential modular manner. This solution interacts with the thermo-hydraulic solver through the measurement and actuation signals (Juslin 2005).

The models in Figs. 6, 7 and 8 are implemented in the APROS simulator. The CFG implementation is in Fig. 11. At the center is a tank, which models the surface level behavior of the reactor vessel. Above the tank is the feedwater pipeline with a pump and a check-valve, which prevents reverse flow if the pressure in the tank is greater than the pressure after the pump. A pipeline with a valve leading to an empty point branches away from the feedwater pipeline. This is used to model the leak: In case of a leak, the valve is opened. The logic for controlling the valve that simulates the leak is above the valve, and it is triggered by a signal from the EFG, indicating that the ceiling of the room containing the pipeline has collapsed (the model interface EFG_Configuration/XB_10). Under this valve is a flow measurement connected to a model input EFG_Configuration/XA_02; this is connected to the flood input of the room containing the feedwater pipe in the EFG. Under the tank is a pipe leading to a point. This represents loss of water due to evaporation. The mass flow in pipes that reach these empty points will simply exit the simulator.

The feedwater pump is controlled by an on–off logic, so the pump is turned on when water level drops below a low water mark, and the pump is turned off when the water exceeds a high water mark. In a physical system, these marks would be detected by sensors. In the simulation, they are determined by a limit value checker that receives the level measurement of the reactor vessel.

On the right and left sides of the reactor vessel are the emergency water supply lines. Each contains a pump, a check-valve to prevent reverse flow and a shut-off valve, which should be closed if water is detected on the floor of the room, as described in Sect. 3. The actuation logic of these shut-off valves requires information from the EFG, which tells if the room containing the valve is flooded. This information is received from the model interfaces EFG_Configuration/XB_23 and EFG_Configuration/XB_27. The pumps in the emergency water lines are turned on when their actuator is notified that the reactor level has dropped below a critical threshold, which is determined by a limit value checker component that receives the level measurement.

Figure 12 shows the EFG model containing the rooms. Figure 13 shows the internals of a room; the same model is instantiated for all 5 rooms in the EFG. The model is created with the automation components of the APROS simulator and is thus executed as discrete models by the APROS solver. Room B in Fig. 12 receives onto its uppermost input the model interface signal CFG_Configuration/XA_02, which comes from the CFG (Fig. 11) and indicates a leak from the feedwater pipe. Room B contains this pipe, and the leak signal is processed in Fig. 13 as the Flood_in signal. The internal logic will determine that the room is flooded and that the uppermost output Flood_out is true. This in turn is connected to the Flood_in input of Room A in Fig. 12, which also is flooded, so its uppermost output Flood_out is true. This is sent to the model interface CFG_Configuration/XB_23. As discussed above in the explanation on Fig. 11, this information is used by the logic that actuates the valves that close the emergency pipelines. As described in Sect. 3, the logic mistakenly assumes that the flooding was caused by a leak of the emergency pipeline in Room A, resulting in the disabling of the safety function that was meant to deal with the feedwater pipeline leak.

Figure 14 shows some output from the simulator. The level of the tank varies between the upper and lower limits, since on–off control is used for the feedwater pump. After the ceiling of Room B collapses, the hazard scenario described in the case study occurs, so reactor level drops

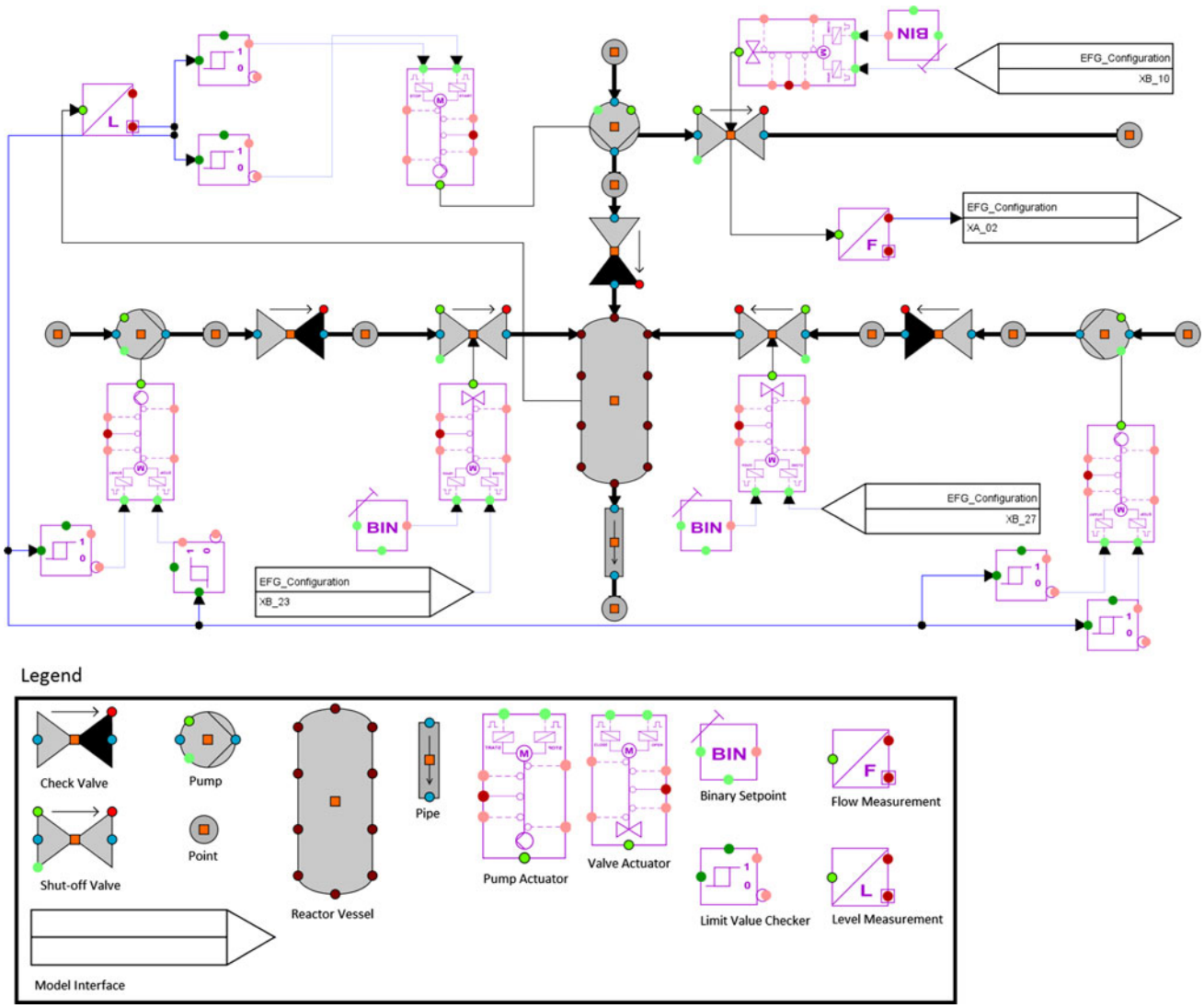

Fig. 11 The implementation of the CFG in the APROS simulator

since the feedwater line is leaking and the emergency lines do not activate.

7 Discussion

7.1 Validation against established risk analysis practice

Fault tree analysis and PRA are considered to be state-of-the-art safety analysis methods in many safety critical industries such as nuclear power. In this section, these methods are compared to the methodology proposed in this paper. A fault tree of the example presented in this paper is in Fig. 15. There are 7 basic events corresponding to the failures of pipes and valves or due to activation of emergency sensors. Additionally, the common cause failure of all of the three pipelines is modeled as a basic event labeled CCF. This approach is the explicit method of modeling common cause failures in fault trees (Vaurio 1998). However, only one combination of basic events has been modeled, while every combination of the 7 basic events should be added if the explicit method is used. One weakness of this approach is the radical growth of the fault tree even for simple applications. The other weakness is that the approach does not support the identification of root causes of a common cause failure involving a specific combination of basic events.

As discussed in the Sect. 2, a key technique for achieving the desired level of safety and reliability is redundancy, such as the two emergency water pipelines in the case study in this paper. Common cause failures can undermine this technique, if a root cause can result in the

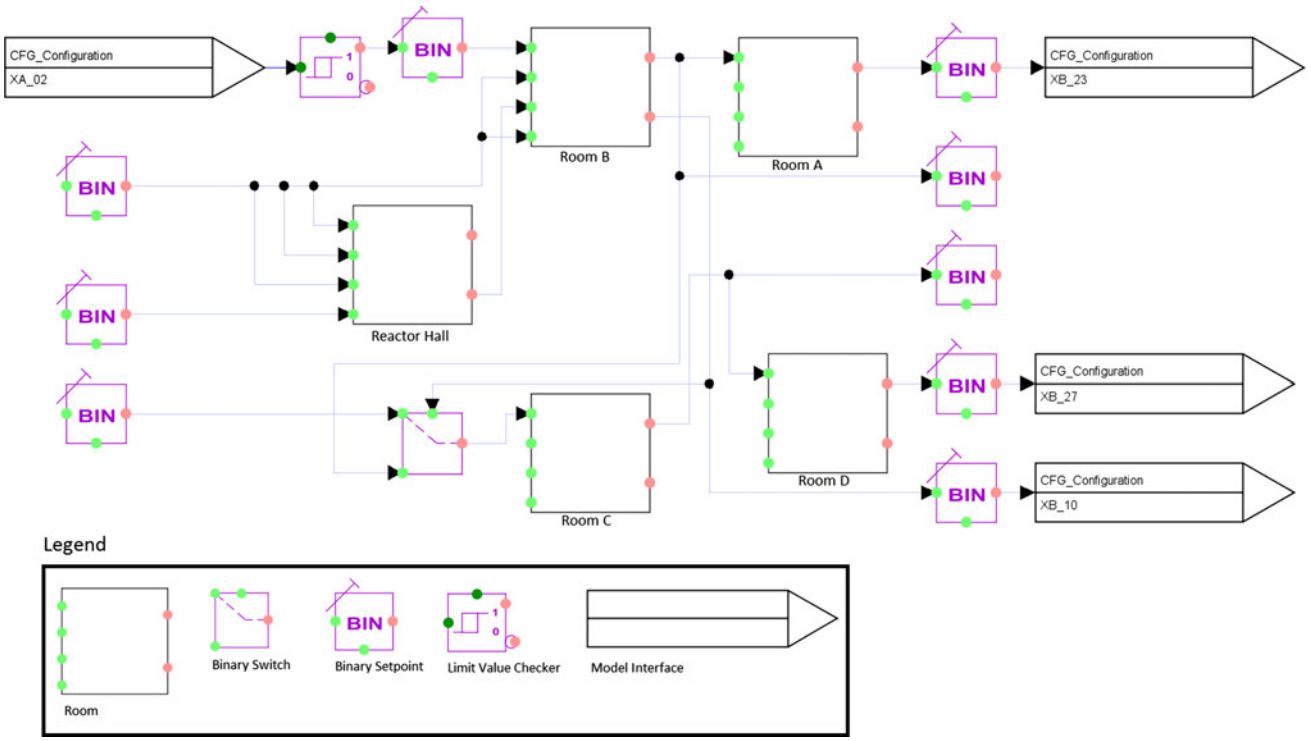

Fig. 12 The EFG model in the APROS simulator

Fig. 13 The model of a room in the EFG

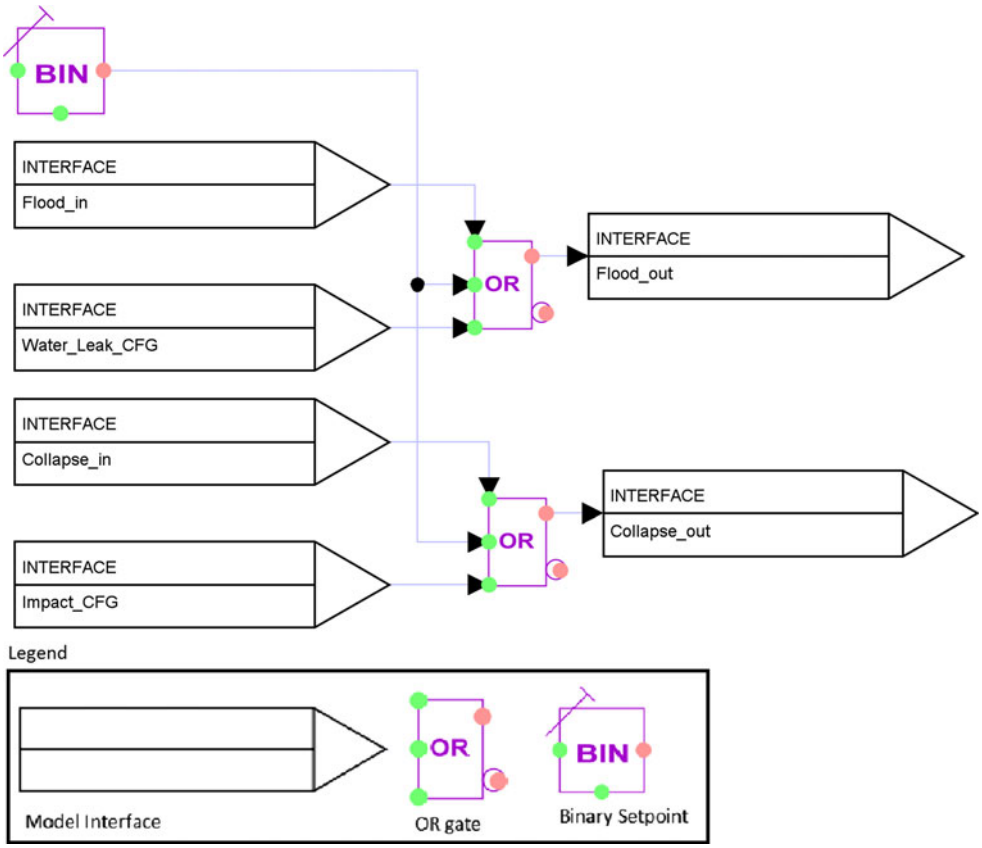

Fig. 14 The surface level measurement of the reactor vessel

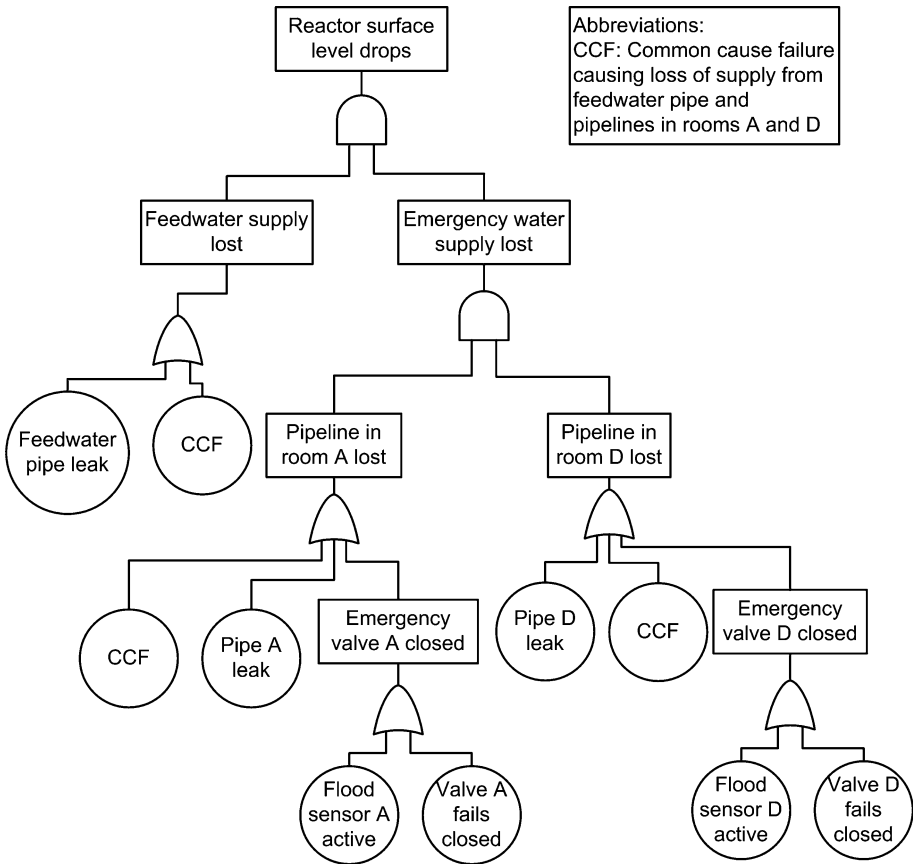

Fig. 15 Fault tree of the case study including the common cause failure

failure of several redundant systems via a coupling mechanism (Kancev and Cepin 2012). Coupling mechanisms can be removed from the design by techniques such as physical separation. In the example in this paper, the emergency pipelines were separated in different sections of the building that were separated by a waterproof wall. Recent research on common cause failure analysis with fault trees and PRA focuses on managing the rapid growth of the fault tree resulting from the explicit method of modeling all combinations of common cause failures (Kancev and Cepin 2012). In contrast, the method in this paper serves to identify coupling mechanisms that make it possible for a root cause to result in common cause failure of redundant systems leading to the top event in the fault

tree. In the example, the coupling mechanism was related to the layout of the building, in which heavy loads were being lifted above structures responsible for physical separation of redundant systems. The method can be used to overcome the weaknesses of studying common cause failures with fault trees or PRA: Only combinations of basic events that occur in response to a root cause are added explicitly to the fault tree as a basic event. By specifying the crane failure as a possible root cause, the simulation results revealed a common cause failure involving 3 basic events that caused the top event of Fig. 15 to occur. In Fig. 15, the identified common cause failure is represented by the basic event CCF that was added to the tree using the explicit method. A PRA analysis based on this fault tree would assign CCF the probability of the crane dropping the load and breaking the structures on the lower floor. After this, established PRA methodology can be followed: Either the crane is designed to reduce the probability of CCF or the physical layout of the building is redesigned to remove the coupling mechanism that makes the common cause failure possible.

7.2 Relevance to cyber–physical systems

In a CPS, the cyber elements of networked software-based computing interact with the physical world through sensing and actuating devices. The physical world in this case consists of the equipment that is intended to be controlled as well as the environment of the system. The case study in this paper is a primitive CPS, as there are three distributed computing modules controlling valves, which together realize the function of maintaining the reactor surface level under normal and exceptional conditions. Poovendran et al. (2012) list numerous examples of more advanced cyber–physical systems, but also points out that that the key issue in cyber–physical research is not the cyber advances but the development of methodologies that are able to model and analyze the interactions over the interface of cyber-space and the physical world. In this paper, these challenges are confronted when the framework in Sect. 4 is implemented onto a simulation platform that has both cyber and physical elements.

In our case study, the interface between cyber and physical elements is the surface level measurement from the reactor and the actuation signals to the valves and pumps. For even a very complex CPS, the interface can still be modeled as measurements and actuation signals (Banerjee et al. 2012b). In order to obtain valid analyses, the dynamic simulation package must be able to simulate both the physical and cyber aspects of the model and to interface them through the measurement and actuation signals. The general procedure for accomplishing this was discussed in the Sect. 6 and is presented in detail in Juslin (2005). The complexity of cyber–physical phenomena that can be modeled depends on what dynamic simulation environment the framework in Sect. 4 is implemented, so the concepts of failure propagation in Sect. 4 are presented in general terms that do not assume the use of a specific simulator product.

The method proposed in this paper is not limited to cyber–physical systems, as the framework in Sect. 4 can be applied to any system that can be described with a CFG. The implementation with APROS also does not require the use of any cyber components in the model, so, for example, an underground electric grid with no software could equally well be studied. The limitation of the method is that the environment should be possible to describe with the proposed EFG. This limits the use of the method mainly to systems that are located underground or in buildings or ships. One area of further research would be to adapt the framework in Sect. 4 to replace the EFG with other environmental simulations. For example, a surface flooding simulation or a storm simulation could be used to study systems which are not located in constructed environments, as defined in this paper. A non-CPS example of such a system is the above-ground parts of an old electric grid, and a CPS example is the above-ground parts of a smart electric grid. This further research would involve defining the principles for simulating the environmental hazard as well as the failure propagation interface between the environment and the system.

A goal of identifying safety-related constraints as early as possible is in conflict with a goal of obtaining more realistic results through the use of more detailed simulation models. This tradeoff is inherent in CPS development, and it should be made by designers rather than be dictated by the framework proposed here. An iterative approach could also be used, so that the analysis proposed in this paper is performed several times for successively more detailed simulation models in order to obtain safety and reliability constraints as early as possible; in the final iteration, the analysis would be performed on the most realistic model. In this section, the application of the proposed approach to modern simulation environments is evaluated.

8 Conclusion

The framework presented in this paper revolves around the idea of using a discrete-event-based model to simulate dynamic changes in the environment. Much more detailed and computationally costly modeling and simulation approaches could have been used. The advantage of the presented approach is early phase applicability, computational lightness and exploitation of human safety analysis expertise to systematically model hazardous changes in the

environment. The results provided by the proposed method do not guarantee that the scenario will actually occur. For example, the leak size, room dimensions, floor elevations and location of equipment within rooms will affect the outcome. The significance of the result is that, according to professional safety engineering judgment, the functional failure scenario is possible, depending on the parameters of the model. Designers, possibly in collaboration with safety engineers, will then decide whether a fundamental design change, such as a new physical barrier, is required. Alternatively, no fundamental changes are made, but a requirement for detailed design is added, stating that the design must cope with the initiating event and be parameterized in such a way that the probability of the identified common cause failure scenario is within acceptable limits.

The approach for modeling the environment will not account for poor layout decisions within rooms. For example, if two components are installed too close to each other, heat from one component may damage the other. A more detailed model to replace the EFG would be needed. However, the rearrangement of components within a room is a considerably less drastic design change than moving components between rooms or modifying the physical barriers between rooms. The latter kinds of changes can be captured by the framework here, and due to the high cost of making such changes in later design phases, a methodology for capturing them in the early phase has been proposed in this paper.

The approach presented here applies to constructed environments consisting of spaces with clear physical boundaries. Examples are nuclear, conventional and hydraulic power plants, factories handling hazardous materials, buildings, ships, mines, underground electric power distribution systems, underground water supply systems and underground transportation systems. CPSs not in scope of our definition of the constructed environment include smart surface electric grids, fleets of autonomous machines on surface worksites, fleets of autonomous aerial vehicles and highly automated next generation surface transportation systems. In these cases, the room-based modeling approach used in this paper would need to be replaced by a region-based approach, such as the regions of interest and impact suggested in Banerjee et al. (2012b). One problem for further research is to upgrade the EFG presented here to a region-based modeling approach in order to obtain a framework for identifying hazardous and potentially catastrophic common cause failures in the CPSs that are not in the scope of this paper. The new environmental model would necessitate redefining the procedure of incorporating human safety expertise into the models.

Acknowledgments This research was partially supported by the Academy of Finland, Grant Number 257459.

References

Abbas M, Vachtsevanos GJ (2009) A hierarchical framework for fault propagation analysis in complex systems. IEEE AUTOTESTCON 2009—systems readiness technology conference: mission assurance through advanced ATE. Anaheim, CA, United states, pp 353–358

Ali ANA, Ariffin J (2011) Model reliability assessment: a hydrodynamic modeling approach for flood simulation in damansara catchment using infoworks rs. 1st international conference on civil engineering, architecture and building materials. Haikou, China, pp 3769–3775

Anghel M, Werley KA, Motter AE (2007) Stochastic model for power grid dynamics. 40th annual Hawaii international conference on system sciences 2007, HICSS'07. Big Island, HI, USA

Augustine M, Yadav OP, Jain R, Rathore A (2012) Cognitive map-based system modeling for identifying interaction failure modes. Res Eng Design 23(2):105–124

Banerjee A, Kandula S, Mukherjee T, Gupta SKS (2012a) Band-aide: a tool for cyber-physical oriented analysis and design of body area networks and devices. ACM Trans Embed Comput Syst 11(2):49–77

Banerjee A, Venkatasubramanian KK, Mukherjee T, Gupta SKS (2012b) Ensuring safety, security and sustainability of mission-critical cyber physical systems. Proc IEEE Special Issue CPS 100(1):283–299

Braha D, Bar-Yam Y (2004a) Information flow structure in large-scale product development organizational networks. J Inf Technol 19(4):234–244

Braha D, Bar-Yam Y (2004b) The topology of large-scale engineering problem-solving networks. Phys Rev E 69(1):1131–1137

Braha D, Bar-Yam Y (2007) The statistical mechanics of complex product development: empirical and analytical results. Manage Sci 57(3):1127–1145

Castrillón M, Jorge PA, López IJ, Macías A, Martín D, Nebot RJ, Sabbagh I, Quintana FM, Sánchez J, Sánchez AJ, Suárez JP, Trujillo A (2011) Forecasting and visualization of wildfires in a 3d geographical information system. Comput Geosci 37(3):390–396

Chen Y, Zhu D, Zhao J (2004) Small basin flash flood simulation with topmodel. GIS and remote sensing in hydrology, water resources and environment. Three Gorges Dam, pp 41–49

Chen J, Thorp JS, Dobson I (2005) Cascading dynamics and mitigation assessment in power system disturbances via a hidden failure model. Int J Electr Power Energy Syst 27(4):318–326

Choi JS, Kim MB, Choi DH (2005) Experimental investigation on smoke propagation in a transversely ventilated tunnel. J Fire Sci 23(6):469–483

Davis JW, Schultze RG (1976) The practical implementation of regulatory guide 1.75 in nuclear plant instrumentation systems. IEEE Trans Nucl Sci. New Orleans, LA, USA, pp 717–721

Dhillon BS (1978) On common-cause failures. Microelectron Reliab 18(1):533–534

Dhillon BS, Proctor CL (1977) Common-mode failure analysis of reliability networks. Reliability and maintainability symposium. Philadelphia, PA, USA, IEEE, pp 404–408

Dhillon BS, Singh C (1981) Fault trees and common cause failures. John Wiley & Sons. Inc., New York City

Dunbabin M, Marques L (2012) Robots for environmental monitoring: significant advancements and applications. IEEE Robot Autom Mag 19(1):24–39

Ericson CA (2005) Common cause failure analysis. John Wiley & Sons Inc., New York City

Fang C, Marle F (2012) A simulation-based risk network model for decision support in project risk management. Decis Support Syst 52(3):635–644

Flemming KN (1975) A redundant model for common mode failures in redundant safety systems. Sixth Pittsburgh annual modeling and simulation conference. Pittsburgh, pp 579–581

Frew EW, Elston J, Argrow B, Houston A, Rasmussen E (2012) Sampling severe local storms and related phenomena: using unmanned aircraft systems. IEEE Robot Autom Mag 19(1):85–95

Ge X, Paige RF, Mcdermid JA (2009) Probabilistic failure propagation and transformation analysis. In: Proceedings of the 28th international conference on computer safety, reliability, and security. Hamburg, Germany, pp 215–228

Grantham-Lough K, Stone RB, Tumer IY (2009) The risk in early design method. J Eng Des 20(2):144–173

Han G-C, Sun S-D, Si S-B, Fu P (2005) Research on model of fault diagnosis and propagation in complex system. CIMS 11:6

Hardiman RC, Kumbale M, Makarov YV (2003) Multiscenario cascading failure analysis using trelss. Quality and security of electric power systems. Birmingham, AL, USA, pp 176–180

Hata T, Kobayashi N, Kimura F, Suzuki H (2000a) Representation of functional relations among parts and its application to product failure reasoning. Int J Manuf Sci Prod 3(2/4):77–84

Hata T, Kobayashi N, Kimura F, Suzuki H (2000b) Representation of functional relations among parts and its applications to product failure reasoning. Int J Manuf Sci Prod 3(2):77–84

Hiller M, Jhumka A, Suri N (2001) An approach for analyzing the propagation of data errors in software. In: Proceedings of the 2001 international conference on dependable systems and networks. Washington, DC, USA, pp 161–172

Hiller M, Jhumka A, Suri N (2002) Propane: an environment for examining the propagation of errors in software. International symposium on software testing and analysis. Roma, Italy, pp 81–85

Hiller M, Jhumka A, Suri N (2004) Epic: profiling the propagation and effect of data errors in software. IEEE Trans Comput 53(5):512–530

Himanen R, Kosonen M, Mankamo T (1989) Defenses against common cause failures: introduction to quantitative approach. Stavanger, Norway: Elsevier Applied Science, London, New York

Hirtz J, Stone R, Mcadams D, Szykman S, Wood K (2002) A functional basis for engineering design: reconciling and evolving previous efforts. Res Eng Design 13(2):65–82

Hossain AKMA, Jia Y, Ying X, Zhang Y, Zhu TT (2011) Visualization of urban area flood simulation in realistic 3d environment. World environmental and water resources congress 2011: bearing knowledge for sustainability. Palm Springs, CA, USA, pp 1973–1980

Hostikka S, Keski-Rahkonen O (2003) Probabilistic simulation of fire scenarios. J Nucl Eng Design 224(3):301–311

Huang Z, Jin Y (2008) Stress and conceptual strength for functional design for reliability. In: ASME international design engineering technical conferences & computers and information in engineering conference, vol 4. Brooklyn, NY, pp 437–447

Jensen D, Tumer IY, Kurtoglu T (2008) Modeling the propagation of failures in software-driven hardware systems to enable risk-informed design. In: Asme ed. International mechanical engineering congress and exposition. Boston, MA, USA, pp 283–293

Jensen D, Tumer I, Kurtoglu T (2009a) Design of an electrical power system using a functional failure and flow state logic reasoning methodology. Prognostics and health management society annual conference. San Diego, CA, pp 1–13

Jensen D, Tumer I, Kurtoglu T (2009b) Flow state logic (fsl) for analysis of failure propagation in early design. ASME IDETC/CIE, design theory and methodology. San Diego, CA, pp 1033–1043

Jiang Z, Pajic M, Mangharam R (2011) Cyber–physical modeling of implantable cardiac medical devices. Proc IEEE Special Issue CPS 100(1):122–137

Juslin K (2005) A companion model approach to modelling and simulation of industrial processes. Aalto University, Espoo

Kancev D, Cepin M (2012) Limitations of explicit modeling of common cause failures within fault trees. Reliability and maintainability symposium (RAMS). Reno, Nevada, pp 1–6

Kim S, Han JB (1995) Plc based desfas in nuclear power plant. In: Proceedings of the 1995 international IEEE/IAS conference on industrial automation and control: emerging technologies. Taipei, Taiwan, pp 686–692

Kirschen DS, Jayaweera D, Nedic DP, Allan RN (2004) A probabilistic indicator of system stress. IEEE Trans Power Syst 19(1):1650–1657

Krus D, Grantham K (2007) Applying function-based failure propagation in conceptual design. ASME international design engineering technical conferences, Las Vegas, NV

Kurtoglu T, Tumer I (2008) A graph-based fault identification and propagation framework for functional design of complex systems. Mech Design 130(5):051401-1–051401-8

Kurtoglu T, Tumer IY, Jensen DC (2010) A functional failure reasoning methodology for evaluation of conceptual system architectures. Res Eng Design 21(4):209–234

Lazzaroni M, Cristaldi L, Peretto L, Rinaldi P, Catelani M (2011) Reliability engineering basic concepts and applications in ict. Springer, New York City

Liang Q (2010) Flood simulation using a well-balanced shallow flow model. J Hydraul Eng 136(9):669–675

Mani P, Chakravorty B (2008) Dam break flood simulation for maithon and panchet dams using nws dambrk model and inundation mapping. J Inst Eng 89:16–19

Mohamed A, Zulkernine M (2008) On failure propagation in component-based software systems. 8th international conference on quality software. Oxford, United kingdom, pp 402–411

Mosleh A, Fleming KN, Parry GW, Paula HM, Rasmuson DM, Worledge DH (1989) Procedures for treating common cause failures in safety and reliability studies. Division of Reactor and Plant Systems, Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory, Washington DC

Muscato G, Bonaccorso F, Cantelli L, Longo D, Melita CD (2012) Volcanic environments: robots for exploration and measurement. IEEE Robot Autom Mag 19(1):40–49

Nassar DM, Shereshevsky M, Gradetsky N, Gunnalan R, Ammar HH, Yu B, Mili A (2004) Error propagation in software architectures. In: Proceedings of the 10th international symposium on software metrics. Chicago, IL, pp 384–393

Ness PS, Bereket D, Hakimi M, Uthus T, Chakravarty A (1989) Knowledge based tool for failure propagation analysis. 1989 American control conference. Pittsburgh, PA, USA, pp 344–348

Neumann PP, Asadi S, Lilienthal AJ, Bartholmai M, Schiller JH (2012) Autonomous gas-sensitive microdrone: wind vector estimation and gas distribution mapping. IEEE Robot Autom Mag 19(1):50–61

O'halloran BM, Stone RB, Tumer IY (2011) Early design stage reliability analysis using function-flow failure rates. International design engineering technical conference—design, theory, and methodology. Washington, DC

O'halloran BM, Hoyle C, Stone RB, Tumer IY (2012) A method to calculate function and component failure distributions using a hierarchical bayesian model and frequency weighting. International design engineering technical conference—design, theory, and methodology. Chicago, IL, USA

Papakonstantinou N, Jensen D, Sierla S, Tumer I (2011) Capturing interactions and emergent failure behavior in complex engineered systems and multiple scales. International design engineering technical conferences and computers and information in engineering conference. Washington, DC, USA, pp 1045–1054

Papakonstantinou N, Sierla S, Tumer IY, Jensen DC (2012) Using fault propagation analyses for early elimination of unreliable design alternatives of complex cyber-physical systems. International design engineering technical conferences and computers and information in engineering conference. Chicago IL

Poovendran R, Sampigethaya K, Gupta SK, Lee I, Prasad KV, Corman D, Paunicka JL (2012) Scanning the issue. Proc IEEE Special Issue Cyber Phys Syst 100(1):6–12

Redmill F, Chudleigh M, Catmur J (1999) System safety: Hazop and software hazop Chichester. John Wiley and Sons Ltd., England

Remenyte-Prescott R, Andrews JD (2011) Modeling fault propagation in phased mission systems using petri nets. Annual reliability and maintainability symposium, RAMS 2011. Lake Buena Vista, FL, USA

Sierla S, Tumer I, Papakonstantinou N, Koskinen K, Jensen D (2012) Early integration of safety to the mechatronic system design process by the functional failure identification and propagation framework. Mechatronics 22(2):137–151

Smith J, Clarkson PJ (2005) Design concept modelling to improve reliability. J Eng Design 16(5):473–492. Available from http://proxy.library.oregonstate.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=aph&AN=18685508&site=ehost-live

Stamatis DH (2003) Failure mode and effect analysis: Fmea from theory to execution. ASQ Quality Press, Milwaukee

Stewart M, Melchers RE (1997) Probabilistic risk assessment of engineering systems. Springer, Berlin

Stock M, Stone RB, Tumer IY (2005a) Comparing two levels of functional detail for mapping historical failures: you are only as good as your knowledge base. Res Eng Design 116(1):425–434

Stock M, Stone RB, Tumer IY (2005b) Linking product functionality to historic failures to improve failure analysis in design. Res Eng Design 16(2):96–108

Stone R, Wood K (2000) Development of a functional basis for design. J Mech Des 122(4):359–370

Stone RB, Tumer IY, Stock ME (2006) Linking product functionality to historical failures to improve failure analysis in design. Res Eng Design 16(2):96–108

Stuk (2002) *Stuklex* [online]. http://www.edilex.fi/stuklex/en/lainsaadanto/saannosto/YVL2-0. Accessed Date 2013

Summers AE (2000) Viewpoint on isa tr84.0.02—simplified methods and fault tree analysis. ISA Trans 39(1):125–131

Summers AE, Raney G (1999) Common cause and common sense, designing failure out of your safety instrumented systems. ISA Trans 38(1):291–299

Thibaux R, Kiciman E, Maltz DA (2005) Grouping failures to infer common causes. Microsoft Corporation, United States

Tumer IY, Smidts CS (2010) Integrated design and analysis of software-driven hardware systems. IEEE Trans Comput 60(8):1072–1084

Tumer IY, Stone RB (2003) Analytical methods for mapping function to failure during high-risk component development. Res Eng Design 14(1):25–33

Vacondio R, Rogers BD, Stansby PK, Mignosa P (2012) Sph modeling of shallow flow with open boundaries for practical flood simulation. J Hydraul Eng 138(6):530–541

Vaurio JK (1998) An implicit method for incorporating common-cause failures in system analysis. IEEE Trans Reliab 47(2):173–180

Vesely WE (1987) Fault tree handbook. illustrated ed.: Government Printing Office

Voas J (1997) Error propagation analysis for cots systems. Comput Control Eng 8(6):269–272

Vtt F (2013) Apros process simulation software [online]. http://www.apros.fi/en/references/nuclear_references. Accessed Date 2013

Wallace M (2005) Modular architectural representation and analysis of fault propagation and transformation. Electron Notes Theor Comput Sci 141(3):53–71

Wang K-L, Jin Y (2002) An analytical approach to functional design. International design engineering technical conferences and computers and information in engineering conference. Montreal, Canada, pp 449–459

Wierman TE, Rasmuson DM, Mosleh A (2007) Common-cause failure database and analysis system: event data collection, classification, and coding. US Nuclear Regulatory Commission, Washington, DC

Willems JC (2007) A behavioral approach to open and interconnected systems: modeling by tearing, zooming, and linking. IEEE Control Syst Mag 27(6):46–99

Xiao S, Xu Wm, Yu Y (2008) A simulative building fire spread tracking system based on fpga and 1-wire bus sensor network. 2008 Asia simulation conference—7th international conference on system simulation and scientific computing. Beijing, China, pp 1482–1486

Zeigler BP, Song HS, Kim TG, Praehofer H (1995) DEVS framework for modeling, simulation, analysis, and design of hybrid systems. In: Proceedings of the 3rd workshop on hybrid systems, Lecture notes in computer science. Springer, Ithaca, pp 529–551

Zhang W, Kamgarpour M, Sun D, Tomlin CJ (2012) A hierarchical flight planning framework for air traffic management. Proc IEEE Special Issue CPS 100(1):179–194