A Component-Based Approach for Modeling Failure Propagations in Power Systems

Ajay Chhokra, Abhishek Dubey, Nagahbhushan Mahadevan, Gabor Karsai
Institute for Software-Integrated Systems, Vanderbilt University,
Nashville, TN 37235, USA
Email:{chhokraad,dabhishe,nag,gabor}@isis.vanderbilt.edu

***Abstract*—Resiliency and reliability is of paramount importance for energy cyber physical systems. Electrical protection systems including detection elements such as Distance Relays and actuation elements such as Breakers are designed to protect the system from abnormal operations and arrest failure propagation by rapidly isolating the faulty components. However, failure in the protection devices themselves can and do lead to major system events and fault cascades, often leading to blackouts. This paper augments our past work on Temporal Causal Diagrams (TCD), a modeling formalism designed to help reason about the failure progressions by (a) describing a way to generate the TCD model from the system specification, and (b) understand the system failure dynamics for TCD reasoners by configuring simulation models.**

Keywords—Cyber Physical Systems, Modeling Language, Generic Modeling environment, Temporal Causal Diagrams, Cyber Physical Energy Systems

I. Introduction

Resilient and reliable operation of cyber-physical systems (CPS) of societal importance such as Electric Power Systems is one of the several top national priorities. Recent blackouts and hurricane Sandy in 2012 have demonstrated grid vulnerability and gave reasons to look at existing defense mechanisms more closely. Electrical protection systems include detection devices such as fast-acting relays that are designed to detect abnormal changes in physical properties (current, voltage, impedance) and actuation devices such as breakers that can be triggered to open the circuit in electrical networks. However, resilient protection system design and operation is still challenging in both the transmission as well as the distribution systems [1]. Distance relays have been known to incorrectly initiate tripping due to an apparent impedance that fell into the zone settings of line relays caused by heavy load and depressed voltage conditions [2]. Protection malfunction and its correlation with major blackouts require a careful rethinking of its system-wide effects [3], [2]. One way to improve the status quo is to invest in the development of a robust diagnostics and prognostics technique that can timely diagnose and pinpoint the source(s) of failures combined with the potential side-effects of automated protection actions.

Our approach is to use a discrete event model that captures the causal and temporal relationships between failure modes (causes) and discrepancies (effects) in a system, thereby modeling the failure cascades while taking into account propagation constraints imposed by operating modes, protection elements, and timing delays. Temporal Causal Diagrams (TCD) [4] can model the effects of faults and protection mechanisms as well as incorporate fine-grain, physics-based diagnostics into an integrated, system-level diagnostics scheme. The uniqueness of the approach is that it does not involve complex real-time computations involving high-fidelity models, but performs reasoning using efficient graph algorithms based on the observation of various anomalies in the system [4]. This approach differs from existing practice where fault analysis and mitigation is dependent on a logic-based approach that relies on hard thresholds and local information, often ignoring system-level effects introduced by the distributed control algorithms.

Electrical power networks are essentially Cyber-physical systems build out of several components. The failure dynamics of the whole system depends upon the failure dynamics of each component and their interactions. So, if we have access to TCD models of system components then it is possible to derive the system TCD as a composition of the component TCDs for a given topology. Consider a section of transmission line system shown in Fig. 1 as to be the system under study. It is a three bus, two transmission line system. The transmission line TL1 is connected to TL2 via a bus B2 in series. A pair of breaker-relay assembly is added to each end of the transmission line. Creating a system level TCD model is highly tedious and error prone even for the simple system shown in Fig. 1. In this paper we discuss the automatic generation of TCD models of small parts of system. We describe a modeling paradigm which captures all the relevant aspects to generate TCD models, for instance, the behavior of components and their interconnections followed by failure propagation. This paper also discusses the automatic generation of simulation models of the system under study to understand the dynamics of the different failure modes identified while creating TCD models.

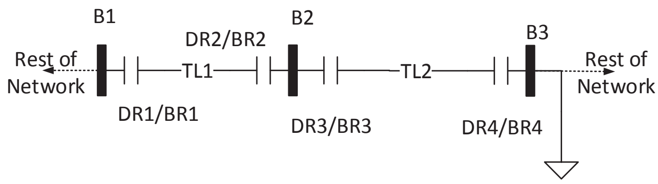

Fig. 1. A section of an electrical network showing two transmission lines, substation buses, and protective elements.

The paper is organized as follows. The next section focuses on related research. Section III describes the modeling language. Sections IV, V discuss the generation of TCD and simulation models respectively; Section VI documents an example scenario to show the generation of component based TCD models and fault analysis of the design using the generated simulation model by inspecting the results. Section VII concludes with a discussion on future work.

978-1-4799-7358-3/15/$31.00 ©2015 IEEE

II. Related Research

In a cyber physical system, physical processes are controlled with various cyber processes running in the controllers. A cyber physical system thus can be viewed as an amalgamation of three components 1) Physical/ Continuous Models 2) Computation/ Discrete Models 3) Interactions between the components.

Modelica [5], [6] is an object oriented, acausal and equation based multi domain modeling language. It offers a large set of standard libraries for modeling physical domains spanning mechanical, electric, hydraulic etc. There are both commercial and free Modelica simulation environments available, such as Dymola [7], MapleSim [8] and OpenModelica [9]. Simscape [10] is another block diagram based (acausal) language for modeling physical processes. It supports a number of domains and is well-integrated into the Simulink environment.

Similar to physical modeling, a number of tools are available that support the modeling of information flows and communication between different components. OMNeT++ [11] and ns-3 [12] are two discrete event simulation frameworks that are widely used to model and analyze computer networks. Other commercially available modeling software include NetSim [13] and OPNET Modeler [14].

There are number of modeling and simulation tools for cyber physical energy systems that cover generation, transmission and distribution. OpenDSS [15], InterPSS [16] are the examples of open source implementations. Some of the free Matlab [10] toolboxes include PSAT [17], VST [18], MATPOWER [19]. Another open source framework associated with modeling, simulation and analysis of energy systems is GridLAB-D [20] and the modeling language is called GLM (GridLAB-D modeling language). A large variety of proprietary solutions exists today, which include PowerFactory [21] and PSCAD [22] etc. [23] provides an excellent reference for studying the challenges involved in modeling and simulation of large power systems.

All of the languages discussed above give more emphasis to the simulation aspect of a system rather than modeling the failure dynamics. Our modeling language is inspired by CyPhyML [24], [25], a domain specific modeling language defined specifically to model different aspects of cyber physical systems. Both these languages were defined using the Generic Modeling Environment (GME) [26]. Since the focus of this work is to aid in fault diagnosis, the modeling paradigm presented in this paper includes objects to define components and their interconnections. As well as the abstract behavior of protection devices, concepts for modeling different failure modes and their propagation, and the timing and mode constraints on the propagation.

III. Modeling Language

The modeling paradigm is created using Generic Modeling Environment (GME) [26], which is a configurable tool set for creating domain specific modeling and program synthesis environments. The modeling paradigm uses generic modeling concepts supported by GME. These include concepts such as hierarchy, aspects, constraints, associations, and generalization.

Some of the GME objects used in defining the modeling paradigm are briefly discussed here. For more information see [26].

1) Atom: These are the elementary parts that cannot contain anything. These atomic objects can be associated to other objects and can have predefined set of attributes whose value can be changed.
2) Model: Models are similar to atoms but the only difference is that the inner structure of a model can be defined. It is a compound object that can contain other types of objects defined by the modeling paradigm.
3) Connections: Connections are primarily used to show relationship/ association among different objects which are contained in the same model.

The following subsections describe the various features of this modeling paradigm.

A. Component Models and their classification

The modeling paradigm divides components into three categories: Plant nodes, Interface nodes and Protection Element nodes. Plant nodes represent physical components like power delivering elements (Transformers and Lines), power conversion elements (Generators and Loads) and buses. Interface nodes include components that interface between the physical and cyber components. Protection Element nodes, as the name suggests, are components that were designed to protect the system by arresting failure-effect propagation.

The component definition hierarchy is as follows: *SystemNode* is an abstract base model for all the components in the system. *PlantNode*, *ProtectElementNodes* and *InterfaceNode* are three abstract models that derive from SystemNode. *Sources*, *TransmissionLine*, *Bus*, *Load* and *Transformer* are children models that extend PlantNode. Similarly, *CT* (current transformer) and *PT* (potential transformer) derive from InterfaceNode. And finally, *Relay* and *Breaker* components, inherit from the ProtectElementNodes class.

All of these components contain ports that serve as interfaces. A port is a model element defined by the GME object atom. There are two types of ports: *Phaseport* (for power and failure propagation) and *DigitalPort* (for information flow).

B. Connections and their classifications

Connections are used in the modeling language to represent flow of (1) power, (2) information (e.g. commands issued by the relays to the breakers or communication between the relays), and (3) failure-effects both within and across component boundaries. Connections include a *label* attribute as well as an *ActivationCondition* attribute. The ActivationCondition attribute imposes constraint on the flow. The use of label attribute helps to simplify the model, as a single Phaseport port can be used to represent 3-phase or single phase connections as well as propagation of one or more failure-effects. This simplification can be extended to digital ports as well.

To provide a layer of abstraction to a modeler, the concept of hierarchy is used to encapsulate the ProtectionElementNodes and InterfaceNode into one component. This component is called the ProtectionAssembly which is of type Model. The

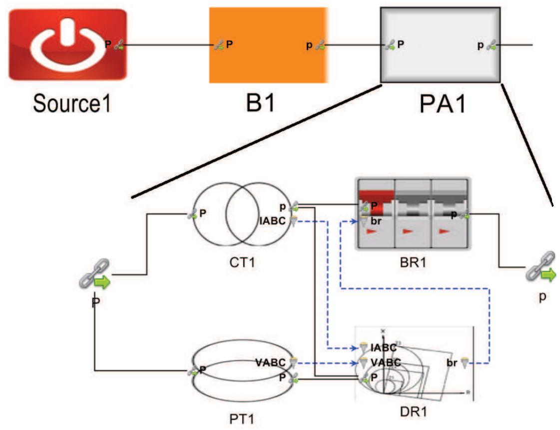

Fig. 2. Different types of components and connections

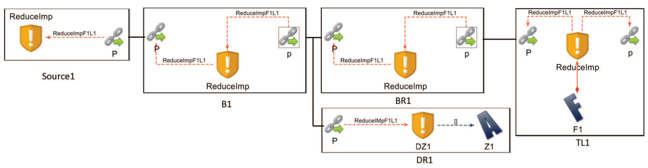

Fig. 3. Fault propagation

ProtectionAssembly model also has PhasePorts and DigitalPorts.

Fig. 2 shows the high-level view of some of the components and their connections. These include components corresponding to source (Source1), bus (B1) and ProtectionAssembly (PA1). It also shows inner structure of PA which contains a current transformer (CT1), a potential transformer (PT1), a breaker (BR1) and a distance relay (DR1). Solid dark line and dotted blue line represent energy, fault, and information propagation across the components.

C. Failure Modes and Propagation

Fault elements in the PlantNode (physical) and the ProtectionElementNode (cyber) components capture the faults in cyber-physical components. The fault effects (or anomalies caused by the fault) are modeled as *discrepancies*. The connection between a fault and a discrepancy represents the cause-effect relationship. The failure effect propagation across component boundaries is captured by connections to and from the Phaseports. Multiple failure-effect propagations through the same ports are distinguished by the labels on the connections.

Fig. 3 exhibits the fault propagation among the components shown in Fig. 2. A three phase to ground fault: F1 is induced in a transmission line TL1. This fault produces a discrepancy/ abnormality, ReduceImp in TL1 and propagates to Breaker BR1, Bus1 and Source1. This is shown by the yellow icons in other components and connections between these discrepancy icons and ports. Inside a component, dotted and solid red line implies the local failure propagation. A label ReduceImpF1L1 is added to these connections in order to disambiguate between different failure effect flows. In case of a protection element like distance relay, these failure effects might be identified as zone$_n$ discrepancies. In the Fig. 3 the ReduceImp discrepancy is mapped to DZ1 failure effect. This failure effect produces alarm A1. The components CT1 and PT1 of Fig. 2 are assumed to be sensors for distance relay DR1 and hence not shown in the Fig. 3

D. Behavioral Semantics

This modeling paradigm provides a rich set of concepts to model the behavior of components. The different objects provided are *State, Transition, Junction State, Parameters, Variables, Events*. In this work, we focus on the behavioral models of protection elements such as relays and breakers. These components are modeled as timed discrete event systems. In the future, we plan to extend the behavioral model to all types of components.

The events associated with protection elements behavior fall into 3 categories: *messages*, *alarms*, and *commands*. An alarm event occurs when an abnormality is detected. A command event is triggered when one component instructs another one to perform an action. A message events captures general information flow, for instance, a trip message being sent from one relay to other.

The purpose of the other objects in the behavioral model is evident from their names; e.g. object of class State represents a composite state, which connects to other states by connections of class Transition. These connection links have two attributes, Guard and Action. Guards are compound Boolean expressions (that evaluate to true or false). The structure of a guard is defined as *Event[Boolean condition]*. A transition will be enabled if Event occurs and the Boolean condition is true.

Only one initial state is allowed in any state machine which can be any state. An action can also be associated to a state. The last element is a junction state that helps to model compound transitions. The connective junction enables representation of different possible transition paths for a single transition [10].

Fig. 4 shows the behavioral model of a breaker. The model of a breaker includes two states, *Open* and *Close* which are identified by icons of letter *S*. Close is the initial state. The breaker reacts to the command sent by the distance relay and changes its state accordingly. In Fig. 4 c_close and c_open are two such commands identified by icons of letter C. The objects with icon of letter F represent failure modes Stuck_open and Stuck_close. The state of the breaker changes the mode of the subsystem, and affects the flow of power and failure-effects. In order to model this aspect the states of the breaker are related to specific modes of the subsystem. Mon is a variable to save the status of the breaker. This model captures the behavior of the breaker in both normal and faulty conditions.

IV. Generation of System Level TCD Model

This section discusses the notion of generation of TCD models which can be used for failure hypothesis as shown in [4]. Any given design/circuit can be perceived as graph $G = (N, E)$ where N is a set of nodes and E is a set of edges. Nodes can be any of the components discussed in the previous section and edges are power flow connections between them. A series of steps are needed to generate failure modes and their effects in the circuit. The steps are listed as follows:

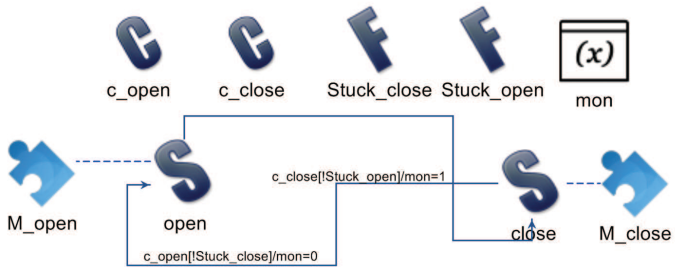

Fig. 4. GME model of breaker

1) Generate a tree for each ProtectionAssembly (Relay) showing association between a relay and the transmission lines it can protect by doing the breadth first search in G starting from every ProtectionAssembly node. The depth of every tree is constrained by the maximum zone reach of a relay contained inside the ProtectionAssembly.
2) Generate another set of trees such that each tree shows the association between a single transmission line and its primary and secondary protection elements.
3) Using the trees produced in step one, for each protection element its zone reaches (z_n) place a marker on the corresponding transmission lines. N marks on a transmission line divides that line into N+1 segments. Each segment acts like a separate failure mode. Since the pair of distance relays at each end of a transmission line are looking in opposite directions, we need to compensate for the distance relay which is set in the opposite direction from the local frame of reference. This is done by subtracting the zone reach mark from the length of the transmission line.
4) Using the trees generated in step one and two, each segment in the transmission lines/failure mode is mapped to a $zone_i$ discrepancy in the relays (secondary and primary). The same compensation is required here for the relays looking in the opposite direction.

V. Generation of Simulation Model

The models developed by this modeling language can be translated to configurable and parametrized simulation models for a variety of simulation frameworks for failure analysis. In this paper, we focus on Simulink simulation framework. The generation is done in a two-step process. In the first step, a Matlab script file is generated from the GME model and from this file a Simulink model is generated.

The steps for the conversion are listed as follows:

- All the PlantNode models are mapped to their respective counterpart library elements with a given set of parameters (except buses).
- For each model of kind ProtectionNode, a custom block is created that encapsulates the behavior and the interface.
- For the InterfaceNode, a V-I interface measurement block is added.
- The local ProtectionNode and InterfaceNode are combined to make a breaker-relay assembly which is referred to as the ProtectionAssembly in the modeling paradigm.
- Then all the connection objects are traversed and corresponding connections are made in the Simulink model.

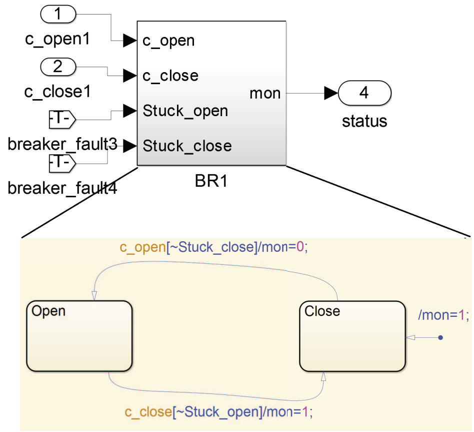

Fig. 5. Stateflow model of breaker

For this paper the behavioral models of distance relay and breaker were exported to the Simulink library and directly referenced.

The Simulink breaker model is also composed of two parts one being a Stateflow chart and second is the in-built three-phase breaker block. The Stateflow model has two kinds of input ports: one for cyber faults and other to receive the breaker commands from the distance relay. Fig. 4 shows a simplified model of a three phase breaker in the current modeling paradigm and Fig. 5 shows the generated Simulink/ Stateflow model.

VI. Results

Fig. 6 is the GME model[1] of the system under study shown in Fig. 1. Using this topology, we will show the outputs of steps 3 and 4 mentioned in section IV to obtain TCD models. There should be 4 trees generated for 4 ProtectionAssembly blocks and 2 trees for 2 transmission lines. Markings for each protection element are briefly discussed as follows:

- DR1 in PA1: Zone1 reach creates a mark in TL1 at a distance $(63.4008 * 0.80/0.3522) = 144$ km, Zone2 reach creates a mark in TL2 at a distance $(63.4008 * (1.25 - 1)/0.3522) = 45$ km, and Zone3 reach creates a mark in TL2 at a distance $(63.4008 * (2 - 1)/0.3522) = 180$ km.
- DR2 in PA2: Zone1 reach should create a mark in TL1 at a distance 144 km. As it is looking in the opposite direction, the actual distance becomes 36 km (180-144).
- DR3 in PA3: Zone1 reach creates a mark in TL2 at a distance $(126.8016 * 0.8/0.3522) = 288$ km.
- DR4 in PA4: DR4 is also looking in the opposite direction and hence Zone1 reach creates a mark in TL2 at 72 km (360-288) and Zone2 reach creates a mark in TL1 at 90 km (180-126.8016*(1.25-1)/0.3522).

Transmission lines TL1 (3 marks) and TL2 (4 marks) have 4, 5 segments respectively. Segments [0,36), [36,90), [90,144), [144,180] in TL1 and [0,45), [45,72), [72,180), [180,288), [288,360] in TL2 are mapped to failure modes F1, F2, F3,

[1] TL1, TL2 lengths are 180, 360 km, and unit impedance per length is 0.3522 ohms. Zone 1,2,3 thresholds are 80%, 125% and 200% of line impedance respectively.

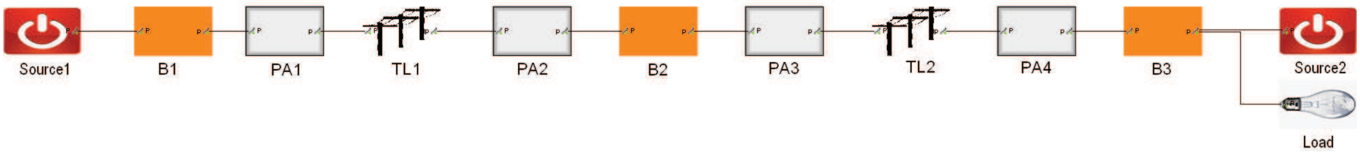

Fig. 6. Two transmission line three bus system GME model

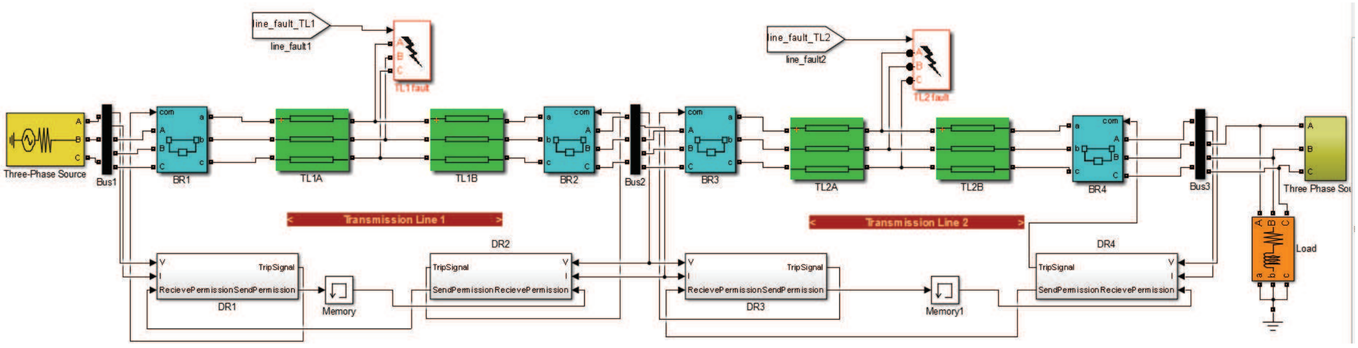

Fig. 7. Two transmission line three bus system SimPower model

TABLE I. FAILURE MODE AND DISCREPANCY MAPPING

Fault	Discrepancy	Fault	Discrepancy	Fault	Discrepancy
F1	DR1.Z1 DR2.Z2 DR4.Z3	F4	DR1.Z2 DR2.Z1 DR4.Z2	F7	DR3.Z1 DR4.Z1 DR1.Z3
F2	DR1.Z1 DR2.Z1 DR4.Z3	F5	DR3.Z1 DR4.Z2 DR1.Z2	F8	DR3.Z1 DR4.Z1
F3	DR1.Z1 DR2.Z1 DR4.Z2	F6	DR3.Z1 DR4.Z2 DR1.Z3	F9	DR3.Z2 DR4.Z1

F4, F5, F6, F7, F8, F9 respectively. The table I shows the mapping between each failure mode and zone discrepancies observed by the protection elements.

Fig. 7 shows the generated SimPower [10] model[2] for the given topology. Faults[3] can be inserted into the system using three phase fault blocks Tl1_fault and TL2_fault blocks. Each transmission line is divided into two blocks in order induce different failure modes generated during TCD model generation. This Simscape model can be used to examine and verify the effects of different faults. The section below discusses the results of two failure mode scenarios.

1) **Persistent Physical fault F1:** A 3 phase to ground fault is introduced in TL1 at time t=6 seconds at 10 % of the length of TL1. This fault forces DR1, DR2, DR4 to fire zone 1, zone 2 and zone 3 alarms respectively. DR3 does not issue any alarm as the fault is behind the DR3. However, due to POTT scheme DR1 sends a message to relay DR2 as soon as it detects zone 1 discrepancy. After receiving this message distance relay DR2 considers this zone 2 discrepancy as zone 1 and both the relays follow fast and delayed reclosure cycles. As the fault is permanent, both relays finally reach blocking state and wait for the operator to manually reset the relay. Fig. 8 shows the status of breakers and commands sent by the distance relays.
   The time stamps *a* and *a1* in Fig. 8 represent the detection of fault by the relays DR1 and DR2. Due to the POTT scheme both relays instructs its respective breakers to open as shown in the Fig. 8. There is a slight drift between these commands due to communication delay. After a certain period (the forward reclosure wait time), both the relays again check the status of the fault by issuing a close command to the breakers. This event takes place at time stamps b and b1 as shown in the Fig. 8. As the fault is of a persistent nature this reclosure is blocked and open command is sent to the breakers. Similarly, after a delayed reclosure wait time the same cycle is repeated again as shown by the marker *c* and *c1*.
2) **Persistent Cyber Fault and Physical Fault F1:** In addition to a 3 phase to ground fault, a cyber fault is induced in the distance relay DR2. Because of this fault the distance relay fails to detect any physical faults. As a result, distance relay DR2 is not able to detect fault F1. The trip message received from the DR1 is also ignored. The distance relay DR4 acts as a secondary relay and trips the breaker BR4 after zone 3 wait time. Fig. 9 shows the status of breakers and commands sent by the distance relays.
   Relay DR1 works normally and follows all the cycles described in the previous scenario. However, due to the cyber fault DR2 is unable to react as shown in the Fig. 9. Since DR4 is acting as a backup relay for DR2, it sends a command to the breaker after a certain period (zone 3 wait time) shown by marker *d* in Fig. 9.

VII. CONCLUSION AND FUTURE WORK

In this paper a new modeling paradigm is presented. It captures the taxonomy of different components used in power

[2]Few modifications are made to the generated model for instance, *goto* and *from* tags are added followed by the removal of container ProtectionAssembly objects PA_n in order to show the encapsulated components and their connections.

[3]Only considering 3 phase to ground fault

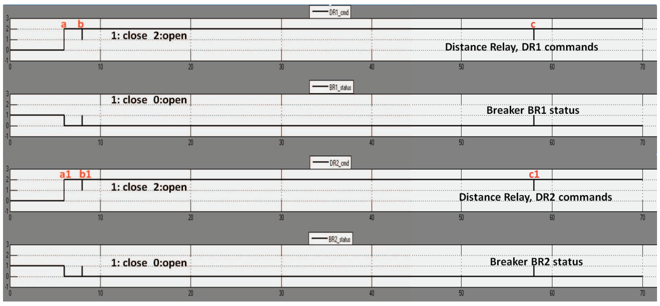

Fig. 8. Distance relay commands and breaker status for scenario 1

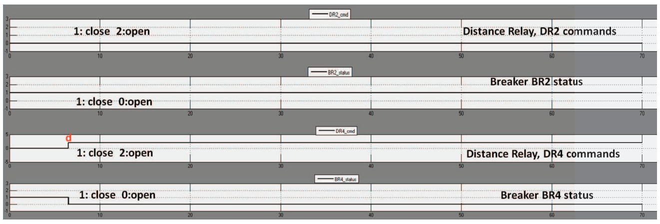

Fig. 9. Distance relay commands and breaker status for scenario 2

systems and their interconnections, behavior of protection elements, failure modes and their propagation. We also discuss the generation of TCD and simulation models from the GME model and show the results of two scenarios for a simple two line, three bus transmission system. As a part of future work, we will try to integrate other open source simulation frameworks for fault analysis and investigate the scalability of the current approach of generating failure modes and their effects against more realistic and complex systems.

ACKNOWLEDGMENT

This work is funded in part by the National Science Foundation under the award number CNS-1329803. The authors are thankful to their colleagues Rishabh Jain, NCSU, Prof. Srdjan Lukic, NCSU, Prof. Anurag Srivastava, WSU and Prof. Chen-Ching Liu, WSU for their suggestions and comments on this work. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

REFERENCES

[1] North American Electric Reliability Corporation, "2012 state of reliability," Tech. Rep., 2012. [Online]. Available: http://www.nerc.com/files/2012_sor.pdf

[2] P. Pourbeik, P. Kundur, and C. Taylor, "The anatomy of a power grid blackout-root causes and dynamics of recent major blackouts," *Power and Energy Magazine, IEEE*, vol. 4, no. 5, pp. 22–29, 2006.

[3] Y. Zhang, M. Ilic, and O. Tonguz, "Mitigating blackouts via smart relays: A machine learning approach," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 94 –118, January 2011.

[4] N. Mahadevan, A. Dubey, G. Karsai, A. Srivastava, and C.-C. Liu, "Temporal causal diagrams for diagnosing failures in cyber-physical systems," *Annual Conference of the Prognostics and Health Management Society*, 2014. [Online]. Available: http://www.phmsociety.org/node/1439

[5] P. Fritzson, *Principles of object-oriented modeling and simulation with Modelica 2.1*. John Wiley & Sons, 2010.

[6] M. Tiller, *Introduction to physical modeling with Modelica*. Springer Science & Business Media, 2001.

[7] http://www.3ds.com/, Dassault Systems, France.

[8] http://www.maplesoft.com/products/maplesim/, MapleSoft, 615 Kumpf Drive Waterloo, ON Canada.

[9] P. Fritzson, P. Aronsson, H. Lundvall, K. Nyström, A. Pop, L. Saldamli, and D. Broman, "The openmodelica modeling, simulation, and software development environment," *In Simulation News Europe*, vol. 44, 2005.

[10] http://www.mathworks.com/, The mathworks, Natick, MA, USA.

[11] A. Varga *et al.*, "The omnet++ discrete event simulation system," in *Proceedings of the European simulation multiconference (ESM2001)*, vol. 9, no. S 185. sn, 2001, p. 65.

[12] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM demonstration*, vol. 15, p. 17, 2008.

[13] http://www.tetcos.com/, Tetcos, Bangalore, India.

[14] http://www.riverbed.com/, OPNET Technologies, San Francisco, CA, USA.

[15] R. C. Dugan, "Reference guide: The open distribution system simulator (opendss)," *Electric Power Research Institute, Inc*, 2012.

[16] M. Zhou and S. Zhou, "Internet, open-source and power system simulation," in *Power Engineering Society General Meeting, 2007. IEEE.* IEEE, 2007, pp. 1–5.

[17] F. Milano, "An open source power system analysis toolbox," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 1199–1206, 2005.

[18] S. Ayasun, C. O. Nwankpa, and H. G. Kwatny, "Voltage stability toolbox for power system education and research," *Education, IEEE Transactions on*, vol. 49, no. 4, pp. 432–442, 2006.

[19] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *Power Systems, IEEE Transactions on*, vol. 26, no. 1, pp. 12–19, 2011.

[20] D. Chassin, K. Schneider, and C. Gerkensmeyer, "Gridlab-d: An open-source power systems modeling and simulation environment," in *Transmission and Distribution Conference and Exposition, 2008. T&# x00026; D. IEEE/PES*. IEEE, 2008, pp. 1–5.

[21] http://www.digsilent.de/, DIgSILENT GmbH, 72810 Gomaringen, Germany.

[22] https://hvdc.ca/pscad/, Manitoba Research Center, 211 Commerce Drive Winnipeg, Manitoba Canada.

[23] P. Palensky, E. Widl, and A. Elsheikh, "Simulating cyber-physical energy systems: challenges, tools and methods," *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, vol. 44, no. 3, pp. 318–326, 2014.

[24] Z. Lattmann, A. Pop, J. De Kleer, P. Fritzson, B. Janssen, S. Neema, T. Bapty, X. Koutsoukos, M. Klenk, D. Bobrow *et al.*, "Verification and design exploration through meta tool integration with openmodelica," in *Proceedings of the 10th International Modelica Conference*, 2014.

[25] G. Simko, D. Lindecker, T. Levendovszky, S. Neema, and J. Sztipanovits, "Specification of cyber-physical components with formal semantics–integration and composition," in *Model-Driven Engineering Languages and Systems*. Springer, 2013, pp. 471–487.

[26] A. Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, and P. Volgyesi, "The generic modeling environment," in *Workshop on Intelligent Signal Processing, Budapest, Hungary*, vol. 17, 2001.