e-Informatica Software Engineering Journal, Volume 6, Issue 1, 2012, pages: 7–26, DOI 10.5277/e-Inf120101

Time coordination of heterogeneous distance protections using a domain specific language

Marcin Kowalski*, Jan Magott*
*Institute of Computer Engineering, Automatics and Robotics, Wroclaw University of Technology
marcin.kowalski@pwr.wroc.pl, jan.magott@pwr.wroc.pl

Abstract

BACKGROUND: Distance protections are widely used in protection of energy transmission lines, but their time coordination is still an important and difficult problem. Inappropriate configuration leads to a hazard event: remote circuit breaker tripping provided the local circuit breaker can be opened, which severely impairs power system operation.
OBJECTIVE: To describe a method and provide software tools to alleviate the hazard in power systems.
METHODS: A domain specific language (DSL) for representation of a transmission line with its distance protection schema, and a translation algorithm from the DSL to probabilistic fault trees with time dependencies (PFTTDs) are employed.
RESULTS: The paper presents software tools that can support power protection experts in time coordination of distance protections. The tools are based upon abstract and concrete syntax of the DSL designed specifically for the purpose of the distance protection time coordination problem. In order to render creation of power line and its protection schema models easier, a DSL-dedicated editor supporting syntax and semantic aspects of the DSL has been developed. Additionally, a translator from the DSL into PFTTD language has been implemented.
CONCLUSIONS: Power system experts are enabled to perform hazard probability assessment and sensitivity analysis.
LIMITATIONS: Translation supports two types of distance protections, which are: single-system relays with starting elements as well as multi-system relays without starting elements. For the single-system relay, there is one timer per relay. For multi-system relays, there is one timer for each of possibly many protection zones. Other types of protections, e.g. overcurrent are not considered.

1. Introduction

Distance protections are widely used in protection of energy transmission lines. The transmission line is divided into sections, whose boundaries are defined by power stations. Because of important consequences of faults like short circuits in high voltage transmission lines, a schema with primary (local) and backup (remote) distance protections is applied. Section where fault occurs is called the local section. Backup protection should disconnect the transmission line in only when the local protection with its circuit breaker has not done it beforehand. However, if the backup protection reacts, then greater part of transmission network is isolated compared with the part disconnected by the local protection. The hazard is the event: remote circuit breaker tripping provided the local circuit breaker can be opened.

For each distance protection, a set of zones, e.g. Zone 1, Zone 2, Zone 3 is defined to approximately point out fault occurring place. The greater the zone number is, the greater part of the line it covers.

Time coordination of primary and backup protections is a significant and difficult problem. If the local protection has not interrupted the

line, then the remote protection has to do it as soon as possible. However, the remote one has to wait for symptoms of line disconnection made by the local one. The remote protection waiting time (also known as time delay or tripping time) for symptoms of local protection activity is usually selected according to generally accepted rules that are not adapted to particular cases. These rules are based on the worst case analysis with safety margin. According to these rules for distance protections [1], time delays for Zone 2 are selected about 350ms, and about 800ms for Zone 3. Therefore, the selected settings are not optimal as far as the prompt fault clearance is concerned.

Time coordination of distance protections (relays) is a part of extensive research in the field of power systems. The approaches already used are: Petri nets [2], linear programming [3,4], evolutionary algorithms [5] and multi-agent systems [6]. In these papers, overcurrent protection schemes are mainly analyzed. Distance protection cooperation with overcurrent protection is considered in papers [4, 7, 8]. These papers do not concern cooperation of distance protections.

Linear programming [3,4] and soft-computing approaches [5, 7] are used in order to solve the relay coordination problem provided tripping times (coordination intervals) are known. The main difference between the above papers and our approach is as follows. In [3–5, 7] time delays are supposed to be input data selected accordingly to generally accepted rules. Our goal is such selection of time delays for zones which is based on parameters of: transmission line, source, load, protections, and time characteristics.

Papers [9] and [10] show that time coordination of distance protections can be achieved for significantly smaller values than the recommended ones by using respectively: fault trees with time dependencies (FTTDs) and probabilistic fault trees with time dependencies (PFTTDs). In FTTD [9] time parameters are specified in a non-deterministic way by their minimal and maximal values. On the contrary, in PFTTD [10] time parameters are characterized by random variables. Models of the following time parameters have to be found: entrance (exit) times of impedance into (from) characteristics of different zones of protections for different locations of fault, circuit breaker opening time. In the present paper the following distance protection coordination process is proposed.

1. Define scopes of distance protections zones of the power line in question.
2. Specify the power subsystem and its protections in a domain specific language (DSL).
3. Translate the DSL-based model into FTTD (or PFTTD).
4. Determine time parameters from the real system or simulation experiments involving e.g. the EMTP utility [11].
5. Find the time delay for each zone of each protection using analytic bounds for FTTD [9] or by simulation for PFTTD [10].

Structures of both trees obtained in point 3. are the same. Time parameters are different only.

When time parameters are calculated using simulation program, then the following power system features have to be taken into account in the evaluation: resistance and reactance of transmission line, source impedance, types of simulated faults, fault resistance, loads, fault locations over the line, impedance characteristics for protection zones. In the present paper two distance protection types are considered, namely:

– single-system relays with starting elements,
– multi-system relays without starting elements.

For the single-system relay there is one timer. For multi-system relays the format of distance relays is the full distance scheme without starting elements, i.e., delays are timed individually for each zone. Relatively long operation time in Zone 1, the fundamental one, is a severe disadvantage of single-system relays. Therefore, in high voltage networks, and extra high voltage networks in particular, multi-system protections are applied instead.

Time coordination of multi-system distance protections using PFTTD has been considered in [10], whereas using FTTD in [9]. Time coordination of single-system protections with starting elements using FTTD has been studied in [12].

For different protection types, PFTTD of different structure is constructed. The output

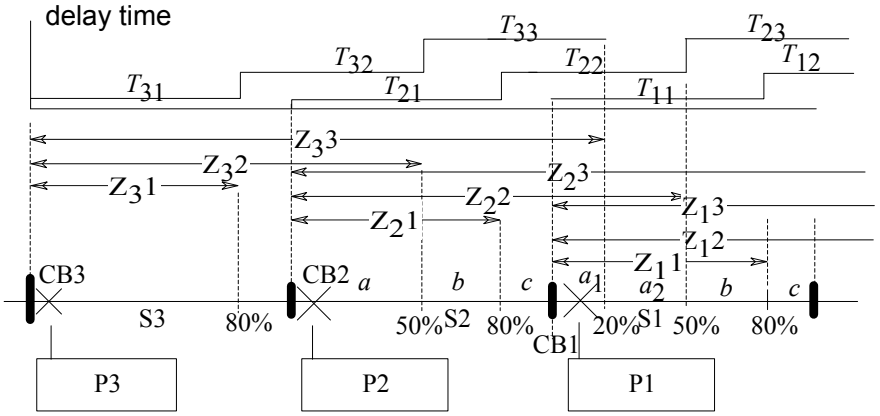

Figure 1. A three-section transmission line with protections, their zones and timing [10]

model differences arise from the aforementioned variety of distance protections supported, each of which finally could possibly disconnect the power line fragment. Since power lines with mixed distance protection types are common, the DSL to PFTTD translation should seamlessly integrate different model fragments to correctly capture the hazard. In the present paper, PFTTD models are considered. Although they have the same structure as FTTD models, they are probabilistic by nature, whereas the latter ones express time parameters in the non-deterministic way.

Although building a new domain language requires substantial development effort, the cost is quickly returned by providing work efficiency unapproachable to generic solutions. In fact, none of general purpose modeling languages may reach level of intuitiveness provided by a domain one, which has been designed specifically to support some particular domain, which in this case is time coordination of power system protections. So, by means of the DSL power system experts may build and optimize models using their own technical vocabulary without familiarizing themselves with computer engineering concepts.

Contrary to a general language, which inherently is a trade-off between requirements of various domains, taking precise semantics offered by a DSL for granted allows to build software engineering tools more effectively accomplishing their tasks, like the PFTTD generation. Had the goal been achieved with some generic solution, it would have involved not only a more verbose transformation, but also implicit restrictions of the generic language used. This is for reasons of intuitiveness and precision that domain languages win acceptance of engineers.

Structure of the paper is as follows. In Section 2, distance protection schema of a power transmission line composed of three sequential sections is outlined. In the next two sections, abstract and concrete syntax models of the DSL are presented. In Section 5, abstract syntax of PFTTD language is given. Next, PFTTD models for power lines with mixed protection types are presented. In Section 7, the transformation from DSL to PFTTD with few PFTTD models is described. The last section recapitulates the research.

2. Distance protection

The ultimate goal of distance protection schema depicted in Fig. 1 is selectivity, i.e. only those Circuit Breakers (CBs) that are required to isolate a fault (short circuit) are opened.

A notation used in Fig. 1 is as follows: Si, where i $\in \{1, 2, 3\}$, is section,
Pi, where i $\in \{1, 2, 3\}$, is the protection placed on the left-hand side of section Si,
Z_ij, where i $\in \{1, 2, 3\}$ and $j \in \{1, 2, 3\}$, is j-th zone of protection Pi.

In the paper faults located by protections in section S1 are considered. According to the zones shown in Fig. 1, the a1, a2, b and c subsections can be distinguished for the S1 section, whereas a,b and c for S2.

Let us suppose that the fault is located by the P1 protection in the $Z_1 1$ zone, i.e. a part of S1 composed of subsections a1, a2, and b. In this case, the P1 protection that is close to the left-hand side of S1 should trip its CB1. The P1 is a primary protection and triggering its breaker turns S1 off. It is possible, however, that a faulty operation of either P1 or CB1 may occur. If the fault occurs, then the remote backup protections P2 and sometimes P3 should trip CB2 and CB3 respectively. In this case, however, the remote section S2 or even S3 are turned off.

The P2 protection operates in three zones. Zone $Z_2 1$ covers 80% of S2. Zone $Z_2 2$ contains S2 and half of S1. Zone $Z_2 3$ covers both sections and 20% of the section which is the right neighbor of S1. The a1 subsection is covered by the $Z_3 3$ zone of P3. This subsection is also covered by zones $Z_1 1$, $Z_1 2$, $Z_1 3$. Hence, this subsection is covered by six zones of three protections altogether. The c subsection, on the other hand, is covered by three zones $Z_1 2$, $Z_1 3$, and $Z_2 3$. These zones are used to roughly recognize a fault location. Finding the zone where fault has occurred is based on measurements of impedance of transmission line from the place of protection mounting to the fault location. Protections trip after T_{ij}, where $i \in \{1, 2, 3\}$ and $j \in \{1, 2, 3\}$, which is a time delay of j-th zone of protection Pi. If P2 recognizes a fault in the $Z_2 j$ zone, where $j \in \{1, 2, 3\}$, then after time delay T_{2j}, P2 sends signal to CB2 in order to open it. Graded times of the tripping delays for zones of Pi, where $i \in \{1, 2, 3\}$, are used ($T_{i1} < T_{i2} < T_{i3}$), i.e. the greater number of the zone, the greater time delay. For the $Z_i 1$ zone, the time delay of the start of the CBi tripping is usually equal to zero.

Two distance protection types are considered, namely:

- single-system relays with starting elements and one timer,
- multi-system relays without starting elements and with one timer for each zone.

Let us explain protection schema assumed for single-system protection with starting element. In analysis performed in the paper, each distance protection with starting element has one timer for each protection. The following protection schema is assumed. There are impedance characteristics: Starting, Zone 1, Zone 2, and Zone 3. The Starting impedance characteristic contains Zone 3, and Zone i characteristic, where $i \in \{2, 3\}$, contains Zone (i-1) one, i.e. there is decreasing order of characteristic areas.

Let τ be time instant when the fault started. An algorithm for protection with starting elements is as follows:

if the starting element of the protection recognized that measured impedance entered Starting impedance characteristic at instant τ
then the timer is set to $\tau + T_1$;
if impedance is in Zone 1 impedance characteristic at instant $\tau + T_1$
then tripping signal is sent to the CB at instant $\tau + T_1$
else at instant $\tau + T_1$ the timer is set to $\tau + T_2$;
if impedance is in Zone 2 characteristic at instant $\tau + T_2$
then at instant $\tau + T_2$ tripping signal is sent to the CB
else at instant $\tau + T_2$ the timer is set to $\tau + T_3$;
if impedance is in Zone 3 characteristic at instant $\tau + T_3$
then at instant $\tau + T_3$ tripping signal is sent to the CB.

The goal is to derive statistical relations between time settings for protection zones of protection and the hazard probability. To carry out investigations, information on operating time of distance relay with respect to fault occurring instant, as well as on dropout time with respect to fault clearance instant are required.

Probability distribution functions of entrance time to and exit time from all concerned zones for protections P1, P2, and P3, under assumption of faults located by relays in subsections a1, a2, b, c of section S1 and subsections a, b, c of section S2 have to be known (Fig. 1). They can be obtained by measurements of real system or simulation experiments using, e.g. EMTP. In the paper, these times are expressed by random variables denoted as $T_{iej|kf}$ entrance time to (exit time from) impedance characteristics of the Zone j for the protection Pi, where:

- $i \in \{1, 2, 3\}$ number of protection Pi,
- $e \in \{en, ex\}$ where en (or: ex) stands for the entrance time to (or: the exit time from) the impedance characteristics

- $j \in \{1, 2, 3, S\}$ number of Zone j or S for starting zone of protection Pi, under assumption: the fault is located by relay (protection) in section k and its subsection f, where
- $k \in \{1, 2\}$ number of section,
- $f \in \{a, b, c, d\}$ name of subsection.

3. Abstract syntax of the domain specific language for heterogeneous power line protection systems

The need to successfully conduct sensitivity analysis of the hazard accounts for development of a language capable of describing distance protections. Then, on the basis of models expressed in that language, PFTTD may be automatically produced and analyzed. To properly support the process, the language should consists of abstract and concrete syntax, which will be analyzed in the current and next section.

As Fig. 2 indicates, these are sections and protections that comprise any power line. Sections are in turn further divided into subsections, and each of them has the *factor* attribute assigned that is determining its length in relation to the containing section. Therefore, the *PowerLine*, *Section* and *Subsection* classes along with their containment hierarchy lay the foundations of designing protections, which is performed using classes for the remaining metamodel.

Consequently, a number of protections and their circuit breakers (represented by the *Protection* and *CircuitBreaker* classes) are found in the metamodel. Since a breaker is triggered by the protection when a fault is found in one of its zones, the *cb* and *zones* associations have been added to specify these objects.

For the protection to trip the breaker, the timing constraints for a particular subsection must are met. This is why every zone works in the range of a few subsections specified by objects of the *SubSectionProtection* class and its *subsection* association. A set of random variables is then necessary to perform hazard analysis. When some protection is local with respect

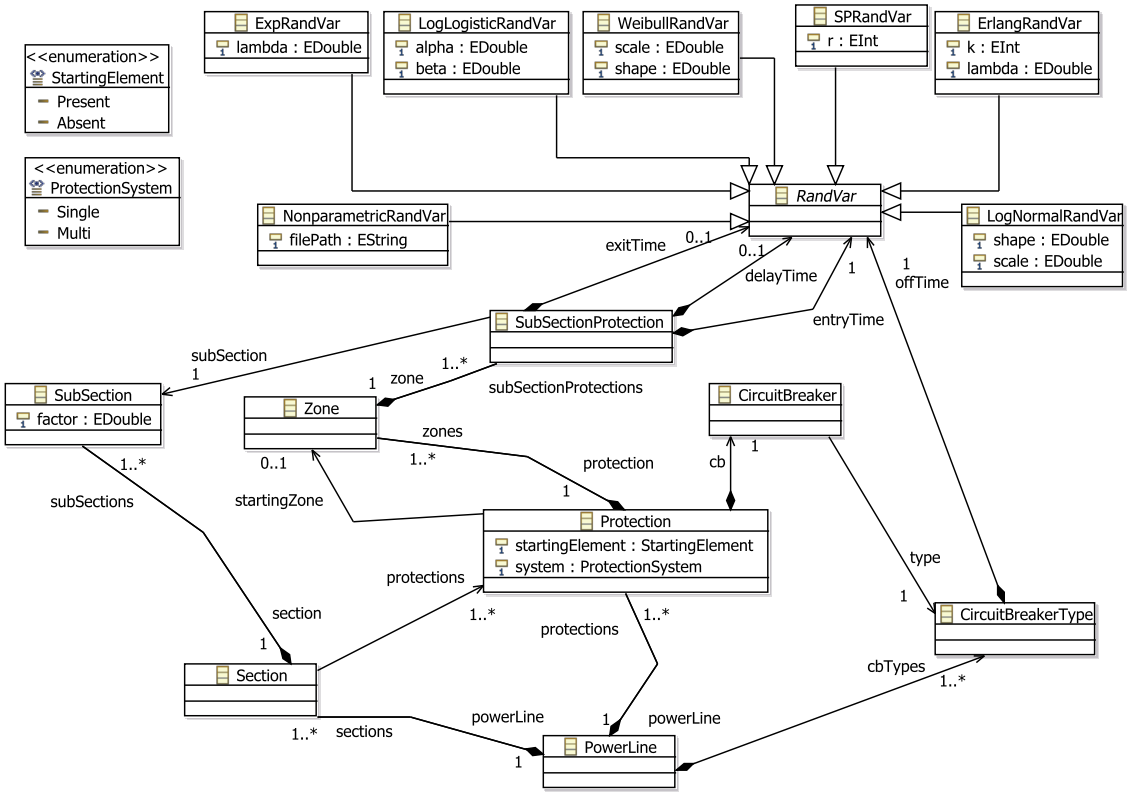

Figure 2. Abstract syntax of the domain specific language for heterogeneous power line protection systems

to the subsection it protects (i.e. the subsection belongs to the same section the protection is located in), only *entryTime* and *delayTime* are needed to run the transformation and simulate the model (see Section 2). Otherwise, if the protection is remote, *exitTime* must be additionally defined to correctly capture hazard dependencies. The *RandVar* class abstracts the variety of probability distributions that random variables may conform to. Logarithmic-normal distribution (*LogNormal*) has been found to fit well when modeling entry times (denoted by the *entryTime* reference in Fig. 2) and circuit breaker opening times (*offTime*) [13].

Compared with [14], the language has been refactored to comply with requirements of the transformation supporting heterogeneous distance protections. As a result, using enumerations such as *StartingElement* or *ProtectionSystem*, power system experts may indicate presence or absence of a starting element, and whether the protection consists of one or many parts.

In order to investigate the hazard probability correctly when a starting element is on the run, a starting zone of a protection must be indicated. This is modeled as a non-containment reference, since all the zones are stored in the *zones* association.

4. Concrete syntax of the domain specific language for heterogeneous power line protection systems

A language may be considered domain specific when domain experts are enabled to conveniently define systems they operate on in this language. Hence, the power system language abstract syntax should be accompanied with concrete syntax, from which an actual editor could be generated. This way, power system experts have to understand neither object oriented paradigms, nor internals of the language they use, which otherwise would limit intuitiveness of the approach.

Generally, there are two ways of combining abstract with concrete syntax [15]. First, the abstract syntax could be derived automatically from concrete syntax by means of fixed metamodel-level translation. However, keeping the distance protection to fault tree model transformation in mind, full control of the abstract syntax is retained in this paper by incorporating the second approach. Having defined the abstract syntax, concrete syntax is built by referencing objects and their relationships in the grammar rules. This way we managed to keep the translation robust and independent from the actual user representation of the model.

The *EMFText* [15] tool from the *Eclipse Platform* was used to design and produce concrete syntax, but feasibility study showed that the *Xtext* [16] tool would have been also useful. By turning the abstract syntax into a focal point of development, both tools could be even used simultaneously.

The grammar definition language is an extension of Backus-Naur Form consisting of rules, each starting with a rule name and ending with a semicolon (for example lines 2–3 from Listing 1). Each rule name refers to some concrete class called alike in the distance protection metamodel (Fig. 2). Moreover, rules are defined using tokens and class features (attributes or references from Fig2) from the metamodel. When a parser enters the rule, it creates a new object conforming to the proper class from the metamodel. It then fill values of attributes and references according to the following tokens.

For example, as line 26 in Listing 1 suggests, when a parser stumbles across the 'Section' token, it creates a new instance of the *Section* class. Next, the parser expects to find the *name* attribute value, followed by an opening curly brace. The *subSection* literal comes from the *Section* class drawn in Fig. 2. At that point, it notifies the parser to invoke (possibly many times due to the '+' character) the rule for *SubSection* and add newly created objects describing subsections to the aforementioned association. Once again for another rule, as line 28 indicates, the parser expects now the 'SubSection' token followed by the *name* attribute value, the 'Factor' token and the *factor* attribute value. There may be many *SubSection* objects, but once a closing curly brace is found, the Section is complete.

When a similar analysis will be started from the 'PowerLine' rule to the last possible rule, the resulting hierarchy will constitute a tree spanning the metamodel. As a result, the parser will create a complete distance protection model designed by a power system expert.

To be more specific about rule definitions, when some feature is an attribute, its name is always followed by square brackets with an optional type attribute written inside the brackets. On the other hand, when some feature is a reference and its name is followed by brackets, the parser will assign to it some already existing object with the *name* attribute equal to the user typed token at that place. When some feature is a reference and is not followed by brackets, a new object will be created. Compare the *subSection* reference in the 'SubSectionProtection' rule (line 36 in Listing 1) with a S1a1 subsection protected by Zone $Z_3$3 (line 28 in Listing 2). The latter listing shows also a definition of the concrete syntax for heterogeneous power line protection systems.

Listing 1. The DSL concrete syntax definition

```
1  RULES{
2  PowerLine::= "PowerLine" name[] "{" "CircuitBreakerTypes" "{" cbTypes+ "}"
3          "Sections" "{" sections+ "}"  "Protections" "{" protections+ "}" "}";
4
5  CircuitBreakerType::= "CBType" name[] "OffTime" offTime;
6
7  WeibullRandVar::= "Weibull" "{" "scale" ":" scale[FLOAT]
8          "Shape" ":" shape[FLOAT] "}";
9
10 ExpRandVar::= "Exp" "{" "Lambda" ":" lambda[FLOAT] "}";
11
12 SPRandVar::= "SP" "{" "R" ":" r[INT] "}";
13
14 ErlangRandVar::= "Erlang" "{" "K" ":" k[INT]
15         "Lambda" ":" lambda[INT] "}";
16
17 LogNormalRandVar::= "LogNormal" "{" "Scale" ":" scale[FLOAT]
18         "Shape" ":" shape[FLOAT] "}";
19
20 LogLogisticRandVar::= "LogLogistic" "{" "Alpha" ":" alpha[FLOAT]
21         "Beta" ":" beta[FLOAT] "}";
22
23 NonparametricRandVar::= "Nonparametric" "{"
24         "Histogram" ":" filePath[] "}";
25
26 Section::= "Section" name[] "{" subSections+ "}";
27
28 SubSection::= "SubSection" name[] "Factor" factor[INT];
29
30 Protection::= "Protection" name[] "{" cb zones+
31         "System" system[] "StartingElement" startingElement[]
32         ("StartingZone" startingZone[])? "}" ;
33
34 Zone::= "Zone" name[] "{" subSectionProtections+ "}";
35
36 SubSectionProtection::= "SubSection" subSection[]
37         "{" "EntryTime" entryTime ("ExitTime" exitTime)?
38         ("DelayTime" delayTime)? "}" ;
39
40 CircuitBreaker::= "CircuitBreaker" name[] "Type" type[];
41 }
```

Let us consider P1, P2 and P3 protections of the a1 subsection contained in the S1 section in Fig. 1. Protections P1, P3 are multi-system, whereas P2 is one-system. A model defined in Listing 2 uses the language from Listing 1, so that the grammar parser can bind rules and eventually create the object model (an instance of Fig. 2) that will become subject of transformation described in the next section.

So, the *PowerLine* block is started first. Then come parts for types of circuit breakers (*CircuitBreakersTypes*), which are further referenced while describing protections. Power line structure (i.e. *Section* and *SubSection* rules) is defined next. Finally, the three protections are described inside the *Protections* block in the following way. For each zone controlled by the protection, a set of subsections is referenced to assign *EntryTime*, *DelayTime* and possibly *ExitTime* values. For the sake of simplicity, only parts of the $a1$ subsection are given in this example. Timing parameters are equal to 0, because EMTP simulator has not been run yet.

Listing 2. A sample model expressed in the DSL for subsection a1 of S1, where protections P1, P3 are multi-system, whereas P2 is one-system

```
1  PowerLine powerLine1 {
2  CircuitBreakerTypes {
3     CircuitBreakerType typeA OffTime LogNormal { Scale: 0 Shape: 0 }
4  }
5  Sections {
6     Section S3 {
7              SubSection S3a Factor 80
8              SubSection S3b Factor 20
9     }
10    Section S2 {
11             SubSection S2a Factor 50
12             SubSection S2b Factor 30
13             SubSection S2c Factor 20
14    }
15    Section S1 {
16             SubSection S1a1 Factor 20
17             SubSection S1a2 Factor 30
18             SubSection S1b Factor 30
19             SubSection S1c Factor 20
20    }
21 }
22 Protections {
23    Protection P3 {
24       CircuitBreaker CB3 Type typeA
25       ...
26       Zone Z_33{
27       ...
28       SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
29          ExitTime Erlang { K: 0 Lambda: 0 } DelayTime SP {R: 0} }
30       }
31       System Multi
32       StartingElement Absent
33    }
34    Protection P2 {
35       CircuitBreaker CB2 Type typeA
36       ...
37       Zone Z_22{
38       ...
39       SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
40          ExitTime Erlang { K: 0 Lambda: 0 } DelayTime SP {R: 0} }
```

```
41        }
42        Zone Z_23{
43        ...
44        SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
45           ExitTime Erlang { K: 0 Lambda: 0 } DelayTime SP {R: 0} }
46        }
47        Zone Z_2S {
48        ...
49        SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 } }
50        }
51        System Single
52        StartingElement Present
53        StartingZone Z_2S
54     }
55     Protection P1 {
56        CircuitBreaker CB1 Type typeA
57        Zone Z11{
58        ...
59        SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
60           DelayTime SP {R: 0}  }
61        }
62        Zone Z_12{
63        ...
64        SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
65           DelayTime SP {R: 0}  }
66        }
67        Zone Z_13{
68           ...
69        SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 }
70           DelayTime SP {R: 0}  }
71        }
72        System Multi
73        StartingElement Absent
74  }}}
```

All in all, the S1a1 subsection is protected in 6 zones: Z_33, Z_22, Z_23, Z_11, Z_12 and Z_13. The Z_22 and Z_23 zones are protected by the P2 protection, so their work is driven by the starting element.

The second example discussed in the following sections differs from the first one in the P3 Protection configuration, so now it is one-system with starting element (Listing 3). First of all, a new zone Z_3S is added (lines 9-13). To indicate that the zone is starting line 16 has been added. Changes made in lines 14-15 notify the transformation to build a PFTTD for a single-system protection with a starting element.

Listing 3. The sample model with the P3 protection being one-system with starting element

```
1  Protection P3 {
2        CircuitBreaker CB3 Type typeA
3        ...
4        Zone Z33{
5        ...
6        SubSection S1a1 { EntryTime Erlang { K: 0 Lambda: 0 }
7           ExitTime Erlang { K: 0 Lambda: 0 } DelayTime SP {R: 0} }
8        }
9        Zone Z3S{
10       ...
11       SubSection S1a1 {EntryTime Erlang { K: 0 Lambda: 0 } }
12       }
```

```
13        System Single
14        StartingElement Present
15        StartingZone Z3S
16  }
```

One possible application of modeling the same line with two different protection schema is to evaluate impact of protection modernization on the hazard. Differences in the two hazard scenarios will be analyzed in the subsequent sections.

5. The language of probabilistic fault trees with time dependencies

The PFTTDs (modeled by the *FaultTree* class in Fig. 3) are bipartite graphs with a single node denoted to be the root, which usually specifies the hazard event. An object of the *FaultTree* class contains objects of classes such as *Node* and *Edge*, which are both further specialized by the notions related to the fault tree language. These are *Event* and *Gate* which constitute the two types of bipartite graph nodes. Objects of those classes are connected through *EventOutput-* and *GateOutput-* edges. The first ones start with events and end with gates (the one drawn between E7 and G1 in Fig. 5 for example), whereas the latter ones start with gates and end with events (the one drawn between G1 and E1).

The second part of the language depicted in Fig. 4 refines the PFTTD gates, which can be causal or generalization. Names of gates consist of two parts. The first letter denotes a kind of gate (C for causal and G for generalization) and the remaining part defines preconditions for a gate to start its output event. The AND, OR and NOT types relate to the classical logic, whereas PAND generates output when both input event occur and the left one occurred as first. Delay gates, which are denoted by an hour-glass symbol, operate like CXOR gates with a single input by introducing random variable-based time delay between input and output events.

6. Discussion of a generated probabilistic fault trees with time dependencies

Although details of the DSL to PFTTD translation will be explained in the next section on the grounds of some specific model cases, in this section discussion of the output model (Fig. 5 from transformation of the a1 subsection of S1 section from Fig. 1) will follow.

Probabilistic Fault Tree with Time Dependencies (PFTTD) analysis starts with identifying hazards, which is the event: remote circuit breaker tripping provided the local circuit breaker can be opened and faults are located by relays in the a1 subsection. For each hazard, a PFTTD is created.

Let us make the following assumption regarding fault occurrence.
Assumption 1: At most one fault can occur during analysis interval, and once it happens, it is permanent.
Types of protections are as follows:
P1, P3 - multi-system without starting elements, and with one timer per each zone,
P2 - single-system with starting elements, and with one timer per one protection.

According to requirements specification, if there is a fault in S1, and additionally P1 and the local breaker CB1 are operational, then only S1 should be disconnected. The hazard is event E1: remote circuit breaker (CB2 or CB3) tripping provided the local CB1 can be opened. Hence, the hazard happens when excessively large part of the power network is isolated. The PFTTD for this hazard and fault located by relays in section S1 and its subsection a1 generated by the translator is illustrated in Fig. 5. The tree contains parts that are similar to the ones for multi-system protection [10] and fragments similar to those for single-system protections [12].

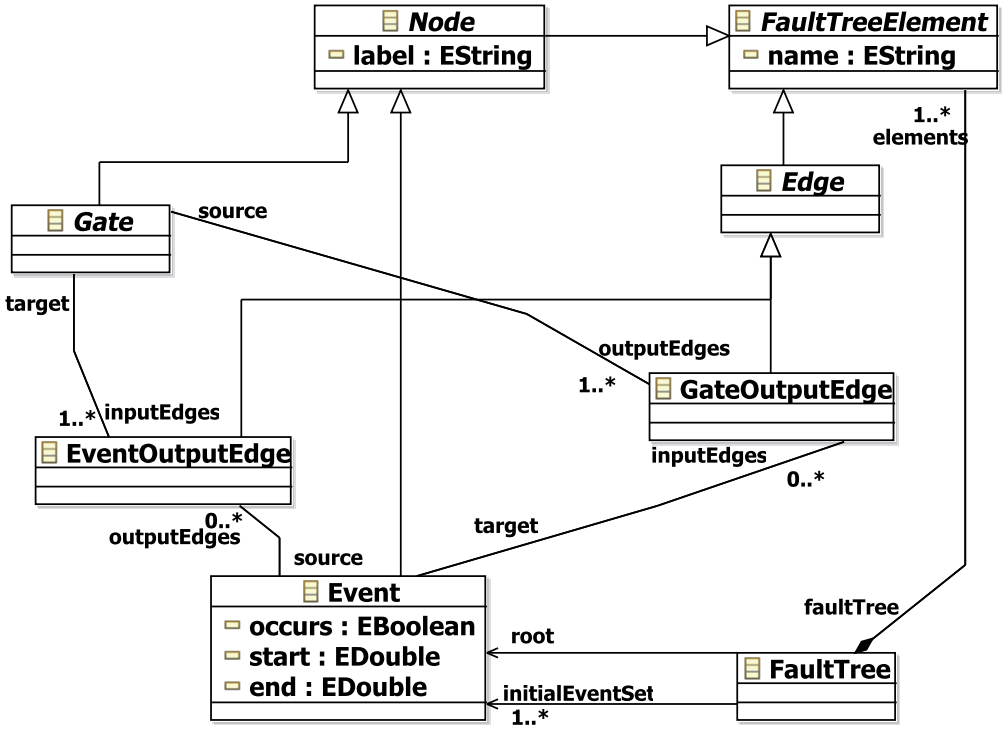

Figure 3. The core part of the PFTTD language

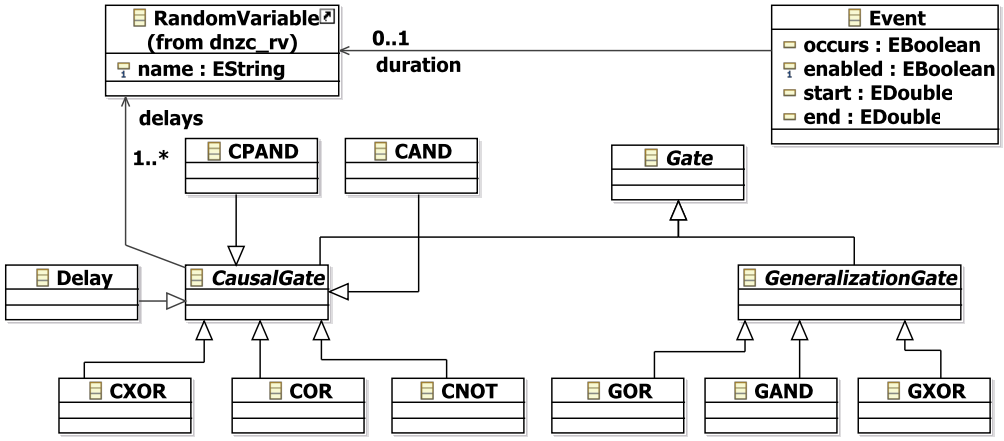

Figure 4. Causal and generalization gates of the PFTTD language

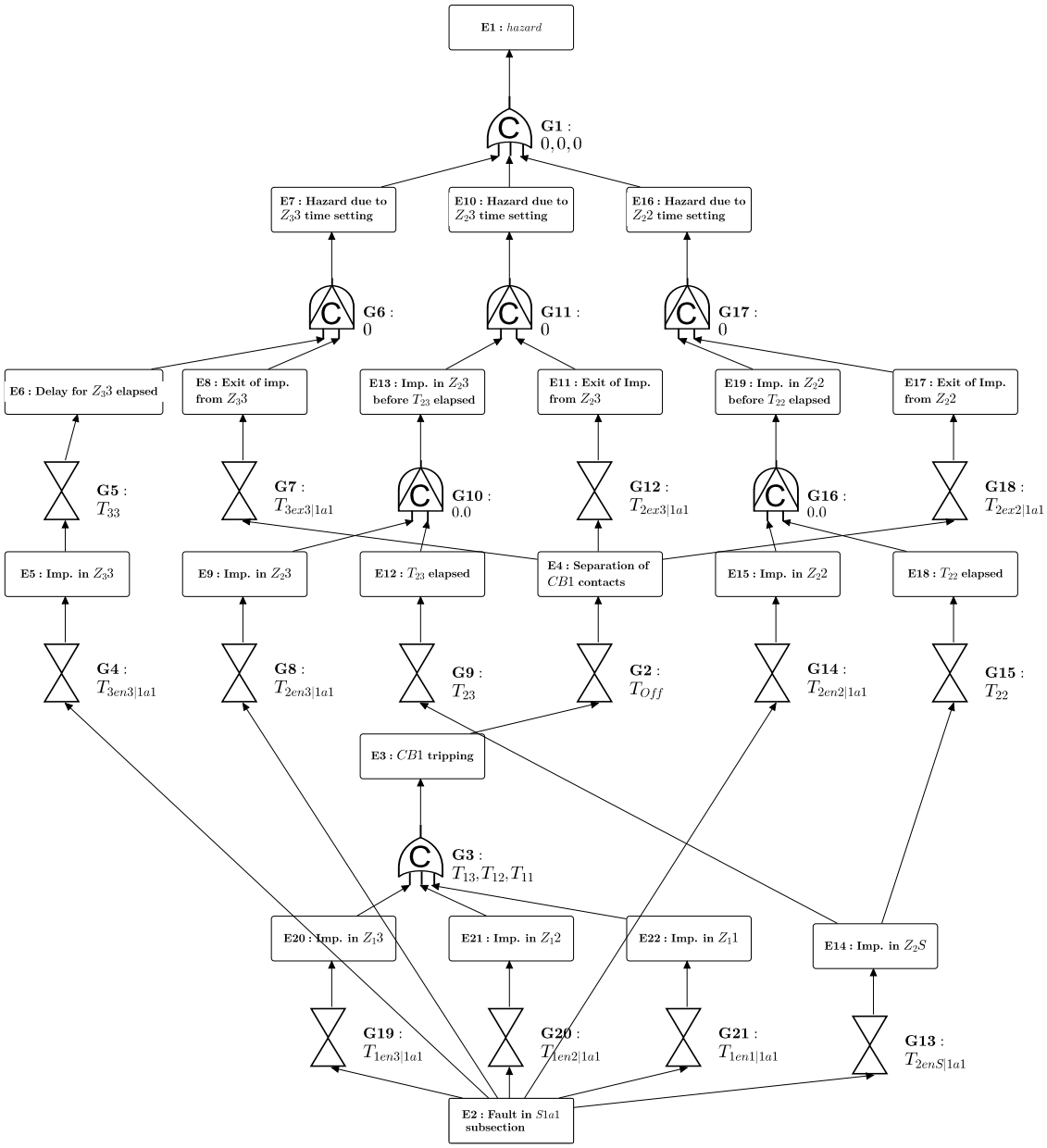

Figure 5. A PFTTD for the first example: fault located by relays in the a1 subsection of S1 section, which is guarded by P1, P3 - multi-system protections without starting elements, and P2 - single-system with starting element.

Event E1 occurs if at least one delay time (T_{33}, T_{23}, T_{22}) is too small, i.e. at least one of the E7, E10 or E16 events has occurred. Hence, the hazard occurs, if at least one remote protection time delay for zones $Z_2$2 or $Z_2$3 of protection P2, or zone $Z_3$3 of P3 has been set incorrectly. Time delay between start instant of event E7, E10 or E16 and start instant of event E1 is equal to 0. Hence, delays for all inputs of gate G7 are equal to 0.

Let τ(Eis) denote instant when event Ei has started. Some events may stop immediately, where others last longer. For G6, if event E7 has occurred then event E6 had occurred not later than the E8 event, i.e. the start of E6 had occurred not later than the start of E8, so τ(E6s)<τ(E8s). In this case, P3 trips its CB3. Turning off process will not be stopped, and CB3 will be opened. In this case, tripping of CB3 will be prior to detection of fault clearance symptoms.

Hence, the P3 relay has reacted too early, what caused that the hazard event E7 has occurred. In order to avoid the hazard, the following condition: τ(E8s)<τ(E6s) has to be satisfied. Additionally, if CB1 of P1 is not opened, then the P3 will not detect that CB1 is opened. Hence, the E8 event does not occur, and consequently neither does the hazard. On the other hand, let us suppose that the P3 had observed that the CB1 is opened (event E8) before time delay T_{33} has passed, so P3 will not turn off its CB3. Hence, event E7 does not occur (the hazard does not occur).

Let us consider a sub-tree with the E7 event as the root. In this sub-tree, the part with the E6 event concerns the P3 protection and its zone $Z_3 3$, while the right sub-tree with E8 concerns P1. In sub-tree with event E16 as the root, the part with the E19 event is related to P2 and its zone $Z_2 2$, while part with E17 is associated with opening activity of CB1 by P1.

If there is a fault in subsection a1 of S1 then impedance seen by P3 can be inside operating characteristics of $Z_3 3$. In this case impedance seen by P2 can be inside characteristics of $Z_2 2$ or $Z_2 3$. Hence, tripping of CB3 can be started after time delay T_{33} from instant the impedance entered characteristic of $Z_3 3$. Time T_{33} is the time from start instant of event E5 till start instant of event E6. The tripping of CB2 can be started after time delays T_{22}, T_{23}, respectively, from instant the impedance entered impedance characteristic of the Starting zone $Z_2 S$ of P2, provided the impedance remains in characteristics of $Z_2 2$, $Z_2 3$. These times are given by real numbers, and are represented by delays of the G9, G15 gates associated with E14 event.

Impedance trajectory measured by P2 enters characteristics of $Z_2 S$ after time $T_{2enS|1a1}$ relatively to instant τ being start of the fault. This time is the parameter of the G13 delay gate. If the fault in subsection a1 of S1 occurred at time instant τ and it still lasts then the impedance seen by P3 enters characteristics of $Z_3 3$ at instant $\tau + T_{3en3|1a1}$. Time $T_{3en3|1a1}$ is the delay of delay gate G4. Times $T_{2en2|1a1}$, $T_{2en3|1a1}$ respectively, associated with P2 are the time delays of the gates G8, G14.

Trajectory of the impedance seen by P3 exits from characteristics of $Z_3 3$ after time $T_{3ex3|1a1}$ since separation of CB1 contacts. This time is the delay of gate G7. CB1 tripping lasts the time given by random variable T_{Off} (Fig. 5). Hence, delay of gate G2 (time between start instant of event E3 and start instant of event E4) is equal to T_{Off}.

If there is a fault in a1 of S1 then impedance seen by P1 can be inside operating characteristics of $Z_1 1$, $Z_1 2$ or $Z_1 3$. Hence, CB1 tripping can be started either immediately, or after time T_{12}, or after T_{13}, relatively to the instant when the impedance seen by P1 entered characteristic for $Z_1 1$, $Z_1 2$ or $Z_1 3$ respectively. Therefore, three times, namely $T_{11} = 0$, T_{12} or T_{13} are delays of the G3 COR gate. They are all equal to time intervals between start instants of input events E20, E21, E22 of this gate and start instant of the E3 output event. Entry times of impedance into characteristics for zones $Z_1 1$, $Z_1 2$ and $Z_1 3$ of P1 are random variables $T_{1en1|1a1}$, $T_{1en2|1a1}$ and $T_{1en3|1a1}$ respectively. These random variables characterize delays of the G19, G20, G21 delay gates.

Detailed explanation of fault trees with time dependencies for single-system protection with starting elements can be found in [12], while explanation of probabilistic fault trees with time dependencies for multi-system protection is given in [10].

7. A power line protection DSL to PFTTD transformation

According to the procedure proposed in the Introduction, the third step of hazard analysis is to translate a domain model into PFTTD. The translation in question is discussed below.

Generation of output models takes place in three phases. The first one produces a fault tree skeleton and is protection independent. In the second phase, the skeleton is supplemented with elements generated from local protections guarding a subsection. Finally, parts related to remote protections are created. The procedures described below (or mappings in the Query View

Transformation parlance [17]) constitute the second and third phase and are run iteratively for each protection guarding the subsection.

The skeleton of every model consists of the E1, E2, E3 and E4 events along with G1, G2 and G3 gates (see Fig. 5). These parts are common among any produced models and describe the fault, hazard and breaker tripping performed by the local protection. Depending upon type and placement of subsection's protections, modifications will be applied to the resulting fault tree.

When a local protection without a starting element is employed, time to trigger the breaker depends on zone entry time as well as on a deliberately introduced delay (possibly 0). Hence, the missing part between the E2 event and G3 gate is composed of: a delay gate, event and delay variable assigned to the G3 gate. For example, G19, E20 and T_{13} in Fig. 5.

Presence of a starting element, however, substitutes that model fragment with the one presented in Fig. 6. The tripping process is altered in such a way that impedance must enter characteristic of a protecting zone (E23) before (E22), so it enters the starting zone (E25) and awaits the delay of the G22 gate. The translation process of this PFTTD model fragment will be discussed on the basis of code snippet presented in Listing 4.

As the name of the $localProtectionWithSe$ mapping suggests (line 10), it produces a set of PFTTD elements related to the operation of a local protection driven by a starting element. They all fit between the line fault event ($E2$) and the COR gate ($G3$), hence the mapping's arguments. The class name after the *query* or *mapping* keywords (e.g. line 1 or 10) indicates the execution context, i.e. a class of the $self$ local variable specific to a particular mapping invocation. Moreover, in this case the mapping can be executed only when the *isLocal* and *hasSe* queries both return the logical truth (lines 10-11), which is when the mapping is applicable. Otherwise, some other mappings (not listed) are executed.

The first query (lines 1-3) traverses the distance protection model to find out whether the starting element has been indicated by a user. The second query checks if the protection that the subsection protection ($self$) belongs to (right-hand operand in line 9) is the same as the local protection of the subsection being analyzed (left-hand side operand).

Three events are created in lines 13-20. For example, the $E22$ event, *delayElapsed* was generated using the *delayElapsed* variable. Similarly, $impInZone$ is $E23$ and $impInZoneBeforeDelay$ is $E24$. Arguments of the event constructor (not shown) are expressions wrapped in some Latex tags using the *stroke* and *latexText* queries (not shown).

Next, the $G21$, $G22$ and $G23$ gates are created, in that order, in lines 22-24. Then, starting from line 26 up to 37 are all the aforementioned elements are bound together by the $GateOutputEdge$ and $EventOutputEdge$ objects. In each case, the first constructor argument is the source element, and the second is target. Finally, the starting element part, whose creation will be discussed shortly, is connected with the gate represented by *delay* variable (lines 36-37).

The final part of the mapping is the assignment of correct random variables. When the starting element is present, all random variables of the COR gate should be equal 0 (line 39). Contrary, entry time of the subsection protection should be transferred from the DSL model, which is performed by the $toEntryTime$ mapping (not shown). When it comes to delay values (lines 41-42), only a new random variable is created without assigning its distribution. It should be filled manually by a domain expert while analyzing the resulting PFTTD.

Special attention should be paid to the $StartingElement$ mapping invoked in lines 36-37, whose code is listed in lines 45-59. It is responsible of creating the $G24$ and $E25$ elements, which specify how a starting element operates. This mapping has returned, for example, the $E25$ event initialized in lines 48-51. Next, the Delay gate is created and initialized. When assigning its timing parameters (lines 53-55), the respective starting element description is searched over the collection of entry times to the starting zone. Newly created elements are finally connected in lines 57 and 58.

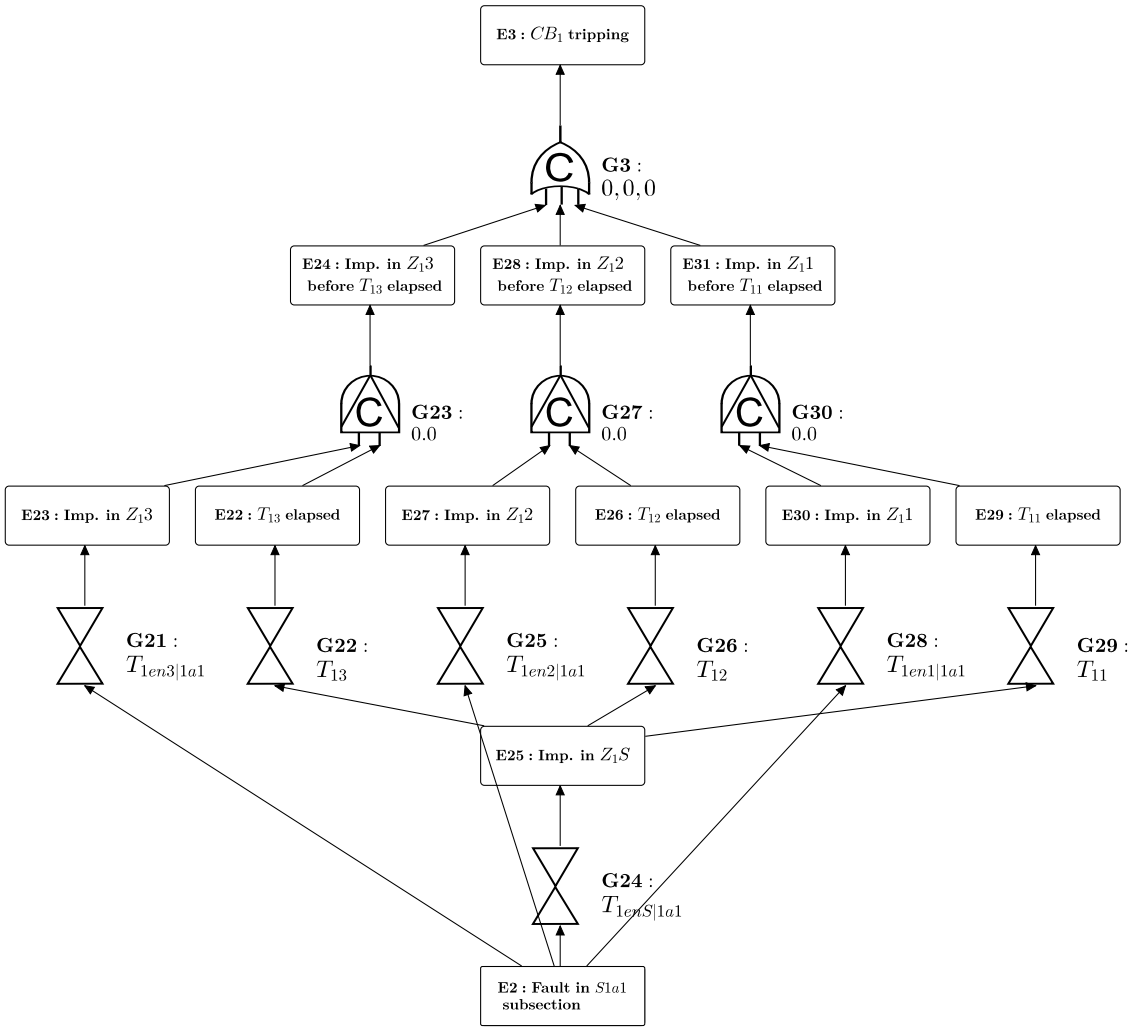

Figure 6. The local circuit breaker tripping by the single-system protection with starting element

Listing 4. A QVT mapping for translation of a single system protection with a starting element

```
1  query SubSectionProtection::hasSe() : Boolean {
2   return self.zone.protection.startingElement = StartingElement::Present;
3  }
4
5  query SubSectionProtection::isLocal() : Boolean {
6   return self.subSection.localProtection() = self.zone.protection;
7  }
8
9
10 mapping SubSectionProtection::localProtectionWithSe(in lineFault: Event,
11     inout cbTripping: COR) when {self.hasSe() and self.isLocal()}{
12
13   var delayElapsed:= new Event(stroke("T_{" +
14     self.zone.protection.protectionNo() +
15     self.zone.zoneNo() + "}") + latexText(" elapsed"));
16   var impdInZone:= new Event(latexText("Imp. in ") + stroke(self.zone.name));
17
18   var impInZoneBeforeDelay:= new Event(latexText("Imp. in ") + stroke(self.zone.name)+
19     newLine() + latexText(" before ") + stroke("T_{" + self.zone.
20     protection.protectionNo() + self.zone.zoneNo() + "}") + latexText(" elapsed"));
```

```
21
22   var entry:= new Delay();
23   var delay:= new Delay();
24   var order:= new CPAND();
25
26   new GateOutputEdge(delay, delayElapsed);
27   new GateOutputEdge(order, impInZoneBeforeDelay);
28   new EventOutputEdge(impInZoneBeforeDelay, cbTripping);
29
30   new EventOutputEdge(lineFault, entry);
31   new GateOutputEdge(entry, impdInZone);
32
33   new EventOutputEdge(impdInZone, order);
34   new EventOutputEdge(delayElapsed, order);
35
36   new EventOutputEdge(self.subSection.map
37     StartingElement(self.zone.protection, lineFault), delay);
38
39   cbTripping.delays += new RandomVariable("0");
40   entry.delays:= Sequence {self.map toEntryTime()};
41   delay.delays += new RandomVariable(stroke("T_{" + self.zone.protection.
42     protectionNo() + self.zone.zoneNo()+"}"));
43 }
44
45 mapping SubSection::StartingElement(in protection: Protection,
46     in lineFault: Event): Event  {
47
48 init {
49   result := new Event(latexText("Imp. in ") + stroke("Z_" +
50     protection.protectionNo() + "S"));
51  }
52  var faultInZoneS := new Delay();
53  faultInZoneS.delays := Sequence{protection.startingZone.
54     subSectionProtections->select(e |e.subSection = self)->
55     first().map toSeEntryTime()};
56
57  new EventOutputEdge(lineFault, faultInZoneS);
58  new GateOutputEdge(faultInZoneS, result);
59 }
```

Independently of the local protection type, the main goal of the second phase is to refine E3, being the circuit breaker tripping event, which is referred to in the last step.

The hazard results from competitions between the local breaker tripping and each of remote protections, which may interrupt too soon some excessively large line area before the local protection will disconnect the local breaker. The third phase binds E4 with E1 in a way dependent on presence of a starting element.

When there is no starting element, the structure such as E5, E6, E7 and E8 is constructed similarly to Fig. 5. However, as described by the end of Section 4, in the second example, protection P3 was turned into a single-system with a starting element. The transformation correctly captured that change and produced a new model fragment with E5, E6, E7, E8 and E9 as shown in Fig. 7.

Furthermore, depending on the real system configuration, protections may be placed either at the beginning of the section (as in Fig. 1) or at its end (as in Fig. 8). Sometimes protections are located at both sides of a section. By analyzing the hazard probability for the model with reverted zones, power system experts may decide on redesigning the real system. Figure 9 depicts the results of transforming the system defined in Fig. 8 for the a2 subsection in the S3 section.

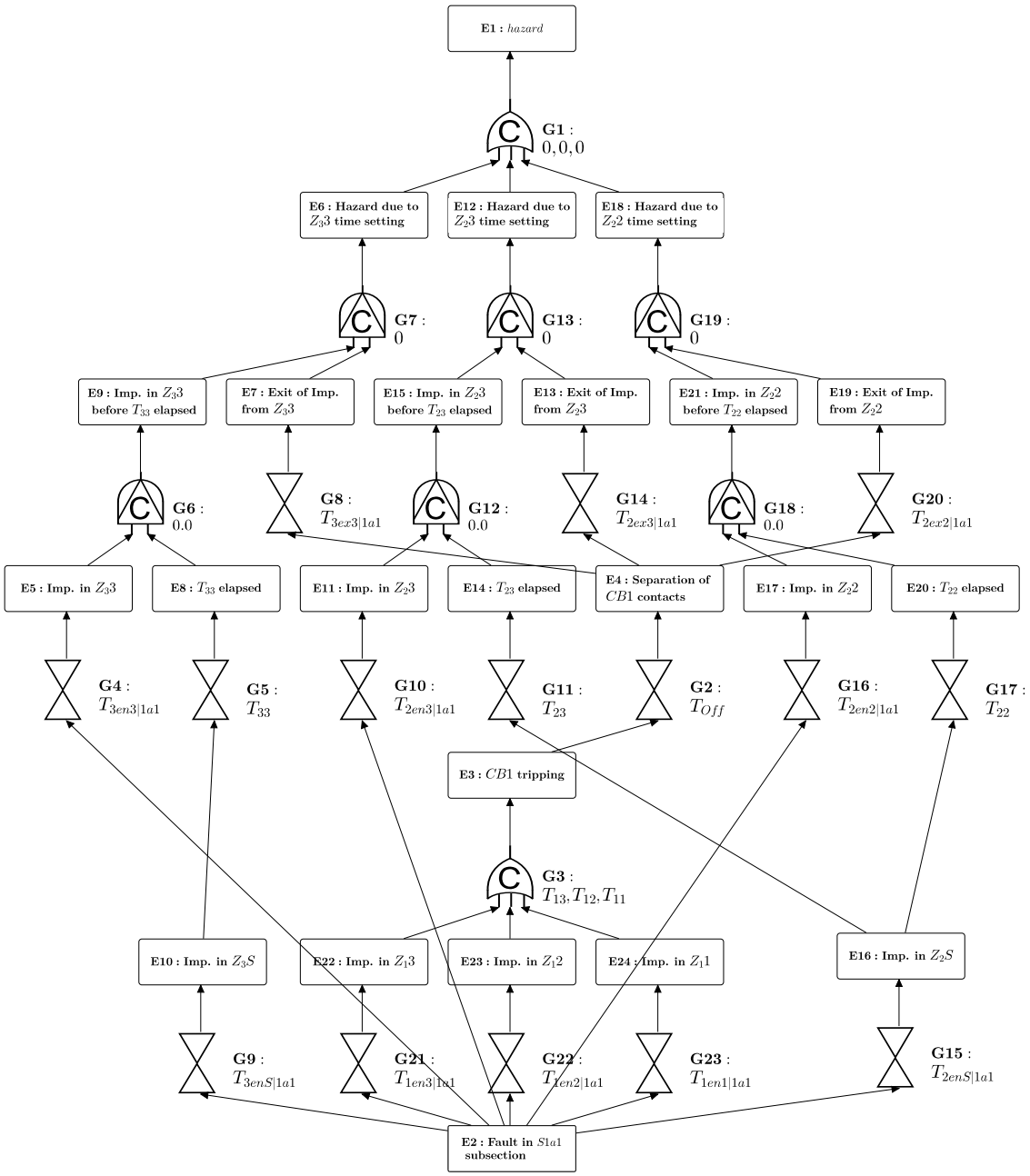

Figure 7. A PFTTD for the a1 subsection in S1 section, which is guarded by: P1 - multi-system protection without starting elements, P2, P3 - single-system with starting element.

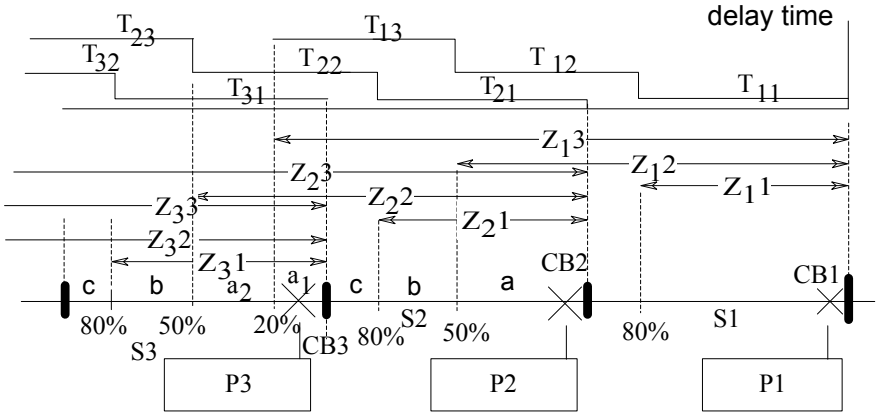

Figure 8. A three-section transmission line with protections placed at the sections' ends

Originally there were no remote protections for this subsection, so no redundant safety measures were taken in the case of fault. According to new protection schema, however, two zones of remote protection P2 additionally guard the area and hazard estimation is possible owing to model drawn in Figure 9.

8. Final remarks

The new domain specific language for representation of a transmission line with its distance protection schema accompanied by the translator to probabilistic fault trees with time dependencies were designed, implemented and verified. The verification was run for different protection schemes. Two types of distance protections: single-system relays with starting elements as well as multi-system relays without starting elements were analyzed. Additionally, protections were located at both ends of the section.

Since structures of FTTD models for distance protection are the same as those conforming to the PFTTD metamodel, the translator might be also employed to the non-deterministic models.

In the DSL, protection schema for line with transformer can be expressed according to paper [9], and the translator can be used for sections with transformer too.

The resulting fault tree proves its usefulness in:

– communication of the risk analysis outcome between power system experts
– simulation
– hazard sensitivity analysis by means of altering protection schema
– hazard sensitivity analysis due to upgrade of protection type.

The last step of the distance protection coordination process proposed in Introduction, i.e. model simulation, requires respective tooling. Software capable of handling PFTTDs generated from the DSL was described in [18].

Acknowledgment

The authors wish to express sincere gratitude to Prof. Leszek Trybus for his in-depth review.

References

[1] *Network Protection and Automation Guide*, Version 1 ed., ALSTOM T and D, 2002.

[2] L. Jenkins and H. P. Khincha, "Deterministic and stochastic petri net models of protection schemes," *IEEE Transaction on Power Delivery*, Vol. Volume 7, No. 1, pp. 84–90, July 1992.

[3] L. G. Perez and A. J. Urdaneta, "Optimal coordination of directional overcurrent relays considering definite time back-up relaying," *IEEE Transaction on Power Delivery*, Vol. Volume 14, No. 4, pp. 1276–1284, October 1999.

[4] ——, "Optimal computations of distance relay second zone timing in a mixed protection scheme with directional overcurrent relays," *IEEE Transaction on Power Delivery*, Vol. Volume 6, No. 3, pp. 385–388, July 2001.

[5] C. W. So and K. K. Li, "Time coordination

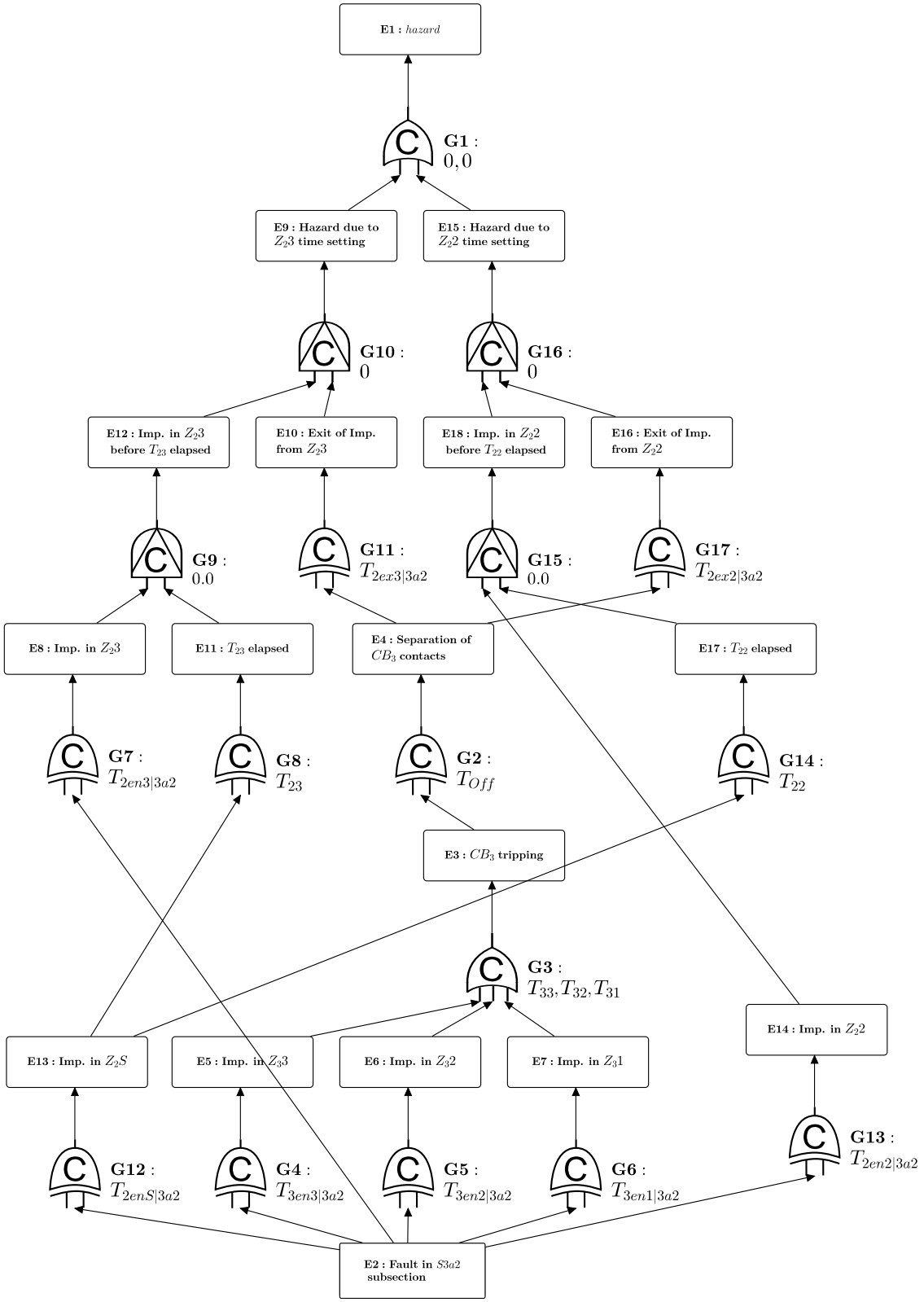

Figure 9. A PFTTD for the a2 subsection in S3 section when protections are placed at the end of sections: P1 - single-system with a starting element, P2 - single-system with a starting element, P3 - multi-system

method for power system protection by evolutionary algorithm," *IEEE Transactions on Industry Applications*, Vol. Volume 36, No. 5, pp. 1235–1240, 2000.

[6] J. H. Chen, S. H. Chen, and Y. M. Yang, "Multi-agent based protection relay system for transmission network," in *Proc. Second International Conference on Machine Learning and Cybernetics*, J. Smith, Ed. IEEE Press, Nov 2003, pp. 2251–2254.

[7] H. A. Abyaneh, S. Kamangar, F. Razavi, and R. M. Chabanloo, "A new genetic algorithm method for optimal coordination of overcurrent relays in a mixed protection scheme with distance relays," in *43rd International Universities Power Engineering Conference UPEC 2008*, 2008, pp. 1–5.

[8] S. Jamali and M. Pourtandorost, "New approach to coordination of distance relay zone-2 with overcurrent protection using linear programming methods," in *39th International Universities Power Engineering Conference*, 2004, pp. 827–831.

[9] M. Lukowicz, J. Magott, and P. Skrobanek, "Selection of minimal tripping times for distance protection using fault trees with time dependencies," *Electric Power Systems Research*, Vol. Volume 81, pp. 1556–1571, 2011.

[10] T. Babczyński, M. Lukowicz, and J. Magott, "Time coordination of distance protections using probabilistic fault trees with time dependencies," *IEEE Transaction on Power Delivery*, Vol. Volume 25, No. 3, pp. 1402–1409, July 2010.

[11] *Electromagnetic Transient Program EMTP*, Leuven Center, 1987.

[12] J. Magott and M. Lukowicz, *Reliability, risk and safety : back to the future.* London: Taylor, Francis, 2010, ch. Time coordination of primary and back-up distance protections with starting elements in electrical power systems, pp. 514–521.

[13] J. Zając, "Short circuit duration time in 110 kv electrical power networks in the light of statistical-probabilistic research (in polish)," Ph.D. dissertation, Poznań University of Technology, Electrical Faculty, 2007.

[14] M. Kowalski and J. Magott, *Methods of development and application of real time systems (in Polish).* Gdańsk: Pomorskie Wydawnictwo Naukowo Techniczne, 2010, ch. A domain specific language for time coordination of distance protections in power systems, pp. 199–208.

[15] F. Heidenreich, J. Johannes, S. Karol, M. Seifert, and C. Wende, "Derivation and refinement of textual syntax for models," in *Model Driven Architecture – Foundations and Applications*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2009, Vol. 5562, pp. 114–129.

[16] *Xtext Reference Documentation*, Itemis, www.eclipse.org/text/documentation.

[17] *MOF Query / Views / Transformations Specification*, Version 1.1 ed., Object Management Group, http://www.omg.org/spec/QVT/, Dec. 2009.

[18] M. Kowalski, *Software engineering in the process of information system integration (in Polish).* Gdańsk: Pomorskie Wydawnictwo Naukowo Techniczne, 2010, ch. A model driven tool for design and simulation of Probabilistic Faults with Time Dependencies, pp. 225–234.