PNNL-33981

# Blockchain for Fault-Tolerant Grid Operations

March 2023

Fernando Bereta dos Reis
Mark Borkum
Monish Mukherjee
David J Sebastian Cardenas
Hayden M Reeve

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

## DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Battelle Memorial Institute, nor any of their employees, **makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights**. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or Battelle Memorial Institute. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

PACIFIC NORTHWEST NATIONAL LABORATORY
*operated by*
BATTELLE
*for the*
UNITED STATES DEPARTMENT OF ENERGY
*under Contract DE-AC05-76RL01830*

**Printed in the United States of America**

**Available to DOE and DOE contractors from
the Office of Scientific and Technical
Information,
P.O. Box 62, Oak Ridge, TN 37831-0062**
www.osti.gov
**ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov**

**Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov**

PNNL-33981

# Blockchain for Fault-Tolerant Grid Operations

March 2023

Fernando Bereta dos Reis
Mark Borkum
Monish Mukherjee
David J Sebastian Cardenas
Hayden M Reeve

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

# Abstract

Radial topology and vast geographic coverage make distribution systems prone to widespread power outages upon the failure of a single (or multiple) upstream component. Fault-handling algorithms depend heavily on correct estimations of the system's state to effectively isolate the affected area and reduce the number of affected customers while maintaining operational safety. The work described here leverages the core features of distributed, consensus-based decision-making processes and the immutability of blockchain, and demonstrates their value in improving fault-tolerant grid operations.

In this work, blockchain was used to create a trusted data-sharing platform that enables independent actors to reconstruct the system state; this enables distributed resources to make intelligent decisions with limited knowledge. Although the process requires data sharing, its algorithms have been designed to limit the amount of private information that is exchanged, which helps preserve business-sensitive data and maintain customer privacy. In addition, by reducing the information that must be shared, the communication requirements are also reduced; (however, an in-depth analysis of the communication requirements is beyond the scope of this project). The proposed use cases are intended to represent a foundational basis for third parties to develop functional solutions that can eventually be deployed in the field. To further provide guidance, the envisioned use cases have incorporated design requirements that consider the blockchain characteristics and a need to limit information from surrounding resources, which preserve the assumption and the possibility that such resources could belong to different entities. This report presents a detailed design of the three use cases with the tools needed to enable the analysis being tested. The implemented gross error detection method can detect mismatches when the error exceeds 3.8 times the sensor's rated accuracy. Detection of the circuit breaker state successfully identified the correct states across all simulation tests. A distribution-system power-flow solution in the simulator OpenDSS generally possesses a convergency tolerance of 0.01% on the voltage magnitude. The evaluation of possible reconnection using voltage magnitude—preserving the data ownership—has a voltage magnitude difference smaller than 0.001% from the OpenDSS result. The results preserving data ownership have a difference within the expected power flow tolerance with full knowledge of the system, which surpasses expectations.

# Acknowledgments

This project was supported by the Department of Energy (DOE) Office of Electricity's Advanced Grid Research and Development Program. The authors would like to thank Chris Irwin and Ryan Knapp of DOE for their support of this study.

# Acronyms and Abbreviations

| | |
|---|---|
| DFLISR | distributed fault location, isolation, and service restoration |
| DLT | distributed ledger technology |
| DSO | distribution system operator |
| FLISR | fault location, isolation, and service restoration |
| OpenDSS | distribution system simulator provided by the Electric Power Research Institute |
| SCADA | supervisory control and data acquisition |
| SE | state estimation |
| UML | Unified Modeling Language |
| WLS | weighted least squares |
| UCR | utility control resource |

# Contents

## Figures

## Tables

# 1.0 Introduction

The power distribution system is responsible for delivering electric power to consumers. Distribution systems are generally radial, so every segment is crucial in keeping customers connected. Furthermore, distribution systems experience more faults than transmission and generation systems. To reduce the number of customers affected after a permanent fault, a fault location, isolation, and service restoration (FLISR) algorithm is used to temporarily reconnect customers by reconfiguring the topology (usually using a centralized controller). A permanent fault is one that is not cleared upon tripping and reclosing. In the absence of a recloser, all faults are permanent (i.e., given that maintenance personnel must be dispatched). To further increase grid resilience, an autonomous, distributed FLISR solution would be welcomed by the industry (Bhattarai et al. 2021). Such a solution requires a decentralized communication platform, which can also provide a layer of “trust” on the data and the decisions being made. The core features of blockchain[1] technology can add value to maintain fault-tolerant grid operations.

Blockchain consists of a list of ordered records, called blocks, that are connected to each other like a chain (called the ledger). The chain relies on cryptographic hashes “linking” the current block to previous blocks. The addition of blocks (i.e., information) to the chain is approved by multiple participants (via a consensus algorithm). Each participant has a copy of the chain and continually verifies its integrity by validating hashes (and periodically comparing third parties’ ledger states). Approaches that use blockchain inherit its distributed agreement and immutable ledger characteristics. The distributed, immutable ledger and the consensus process support a group-level endorsement of the information being hosted. The trusted ledger information can be used to enable a wide array of distributed participants and resources to cooperate and enact changes to achieve their designed goals. An agent’s trust is a function of its ownership, effective security perimeter, accuracy, and other intrinsic properties determined by a risk analysis. A simplified assessment of the distributed power agent’s trust can be based on the location of the agent. Customer-level agents commonly have low nominal voltage and are located in low-security areas having a naturally higher risk than other agents.

Generally, blockchain adds value by providing data-integrity guarantees to applications. Therefore, a FLISR application built on top of blockchain naturally handles data-driven failures better, which can bring multiple benefits to distribution system operators (DSOs) that embrace this technology. For instance, smart contracts can be used to detect and mitigate the effects of bad data records (e.g., records resulting from inadvertent or malicious injection of erroneous data), reducing the risk of a protection system misoperation (and thereby increasing service availability). Similarly, the distributed decision-making aspects of blockchain technology provide a fair, repeatable, and auditable process that can be used to host mechanisms that validate topologies or minimize the number of disconnected customers after a failure. Moreover, the immutability of the blockchain provides a forensic-level, postmortem digital trail that can be used to identify root causes, kill-chains, or other relevant events that have led to an incorrect state or decision. All these properties enable DSOs to make better-informed decisions, resulting in better operational and financial outcomes.

[1] Even though the use of blockchain is frequently linked with financial applications, the underlying distributed systems and consensus mechanisms are based on computational principles that have been mathematically proven to be efficient, secure, and scalable. Thus, within this work, “blockchain” is used to refer to the more agnostic “distributed ledger technology,” which does not have a financial connotation. Notable features of distributed ledger technologies are the ability to achieve consensus at the logical level. These constructs are commonly known as "smart contracts," which let users execute distributed pieces of code across a number of nodes, reducing the possibility of common points of failure.

Based on this summary, it is clear that blockchain can (1) increase the trust in the data capturing process; (2) improve topology identification, thus enhancing system visibility; and (3) enable a new generation of distributed FLISR (DFLISR) solutions. Furthermore, by relying on a single, common ledger (within a service region) enables trackability, even across multiple operational boundaries (e.g., multiple DSOs).

## 1.1 Study Objective

The main goal of this study is to demonstrate the role blockchain can have in improving fault tolerance of power distribution systems. To achieve this goal, a set of detailed Unified Modeling Language (UML) diagrams were created to guide developers of future implementations. Three specific fault-tolerant use cases were developed, as listed in our previous report (Bhattarai et al. 2021):

- Blockchain for Data Configuration (i.e., Use Case 1)
- Blockchain for Topology Identification (i.e., Use Case 2)
- Blockchain for Distributed FLISR (i.e., Use Case 3).

Use Case 1 evaluates individual sensors and identifies sensors that can be trusted for higher decision-making processes (these are evaluated using mathematically based criteria, i.e., the chi-square test for gross error detection). Using information from sensors that have been deemed trustworthy, Use Case 2 identifies the state of the circuit breakers (i.e., the topology of the distribution network). Again, by leveraging the trust provided by the aforementioned use cases, Use Case 3 can safely execute switching actions that enable to reconnect customers that have been disconnected after a fault event via feeder reconfiguration.

## 1.2 Report Structure

The remainder of the report is structured as follows. Section 2.0 presents an overview of the distribution system operation and blockchain applications. It provides the fundamentals for developing fault-tolerant grid operations by leveraging blockchain. Section 3.0 presents the engineering requirements for each of the proposed use cases, focusing on how blockchain can functionally help a distribution system maintain operations under fault conditions. Section 4.0 describes the implementation of the previously described blockchain application for fault-tolerant distribution system operation. It presents the message passing environment, the power system test case, and system available sensors. Section 5.0 presents the behavior that results during each use case. Finally, Section 6.0 presents conclusions and future work.

# 2.0 Technical Background

This section gives a brief overview of distribution networks and their behavior during outages is presented. It details the vulnerability of FLISR systems to malicious attacks and concludes with a broad overview of blockchain applications that can be leveraged to mitigate the vulnerability of FLISR applications to maintain fault-tolerant grid operations.

## 2.1 Distribution Systems

Power systems are vulnerable to multiple threats, including faults initiated by natural sources such as adverse weather conditions, natural disasters, vegetation growth, equipment failure, and malicious attacks (Bompard et al. 2013). Distribution systems provide power to millions of customers (residential, commercial, and industrial), with equipment spread across vast geographic areas. Every segment of the distribution network is vulnerable to failures, and the larger the network, the more likely a failure is to occur. Since distribution systems are normally radial, the failure of any component would disconnect downstream customers (and equipment). Almost 80% of all customer interruptions are caused by faults and outages in distribution networks (Gonen 2015). Figure 1 presents a breakdown of various causes to distribution system faults based on feeder-level data captured through ~300 outage events recorded by five utilities in the United States.

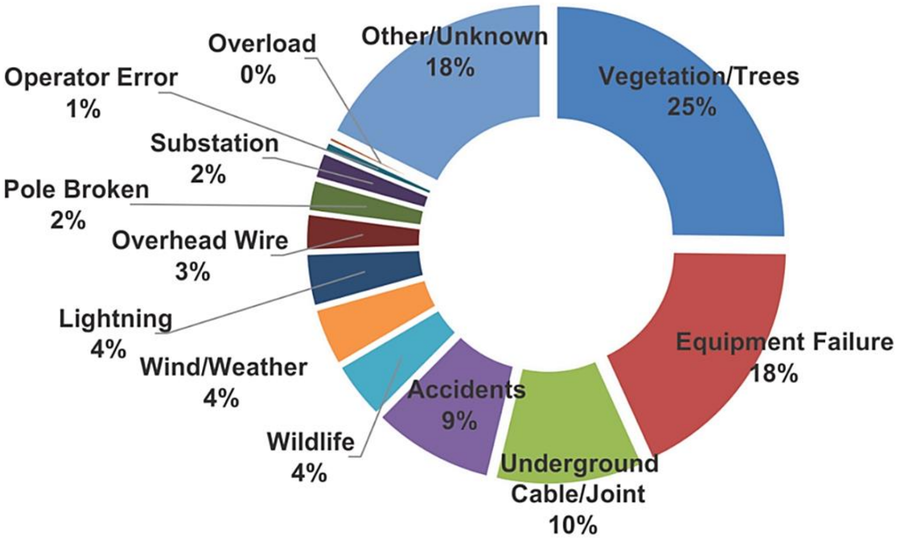


Figure 1. Causes of distribution system faults (Department of Energy 2014)

When a permanent failure occurs, maintenance personnel must find the source of the fault and perform the needed corrections. The restoration time will be subject to the affected component's replacement availability or lead time, available personnel, its location, and availability of tools required to perform the repair. If a permanent fault occurs in the distribution network, all downstream customers will be disconnected until an alternative power delivery method is found (or the fault itself is repaired). To minimize disruptions to customers (and reduce economic repercussions on the utility itself), alternative energy paths are established based on the assumed fault location, the confidence in this assumption, equipment load ratings, and other safety factors needed for operational safety (including repair crew safety). This results in a new topology that can be achieved by selectively operating switching equipment. The FLISR process is illustrated in Figure 2.

The performance of the FLISR system requires coordinated operation of automated feeder switches, reclosers, communication networks, supervisory control and data acquisition (SCADA) systems, historical demand records, and other data to achieve full system-level situational awareness. Figure 2 presents the typical sequence of operations for FLISR systems, starting with locating the fault based on sensor measurements, isolating the faulted sections through appropriate switching operations, and finally re-energizing the non-faulted portion(s) of the feeder to restore customer service. Conventionally the system data is aggregated in a centralizing manner to make sure a complete situational awareness of the system can be attained. With the best knowledge of the state of the system the fault location, and isolation can be performed by engaging the automated feeder switches and reclosers. The service restoration must make sure the resultant topology matches the expected customer demand and the likelihood of introducing operational risks is almost zero.

Clearly, FLISRs are capable of applying automatic restorative actions to the distribution network during degraded operational modes. However, this automated recovery mechanism also makes FLISR systems a desirable target for malicious actors. For example, by cleverly manipulating sensor data a FLISR system could be deceived about the current operational state and result in a misoperation (e.g., disconnect customers under normal operational conditions). Therefore, as part of the engineering requirements, FLISR systems must be able to maintain fail-safe operational conditions when natural or malicious-driven events occur. Another vulnerability of conventional FLISR systems is their centralized architecture, (e.g., applications are hosted at the control center), which can result in a single point of failure while underutilizing the resources hosted in the field or by adjacent DSOs. Due to its architecture, blockchain can increase the trust on the data being produced within the distribution network, enabling applications such as FLISR to take better decisions while reducing the risk of misoperations.

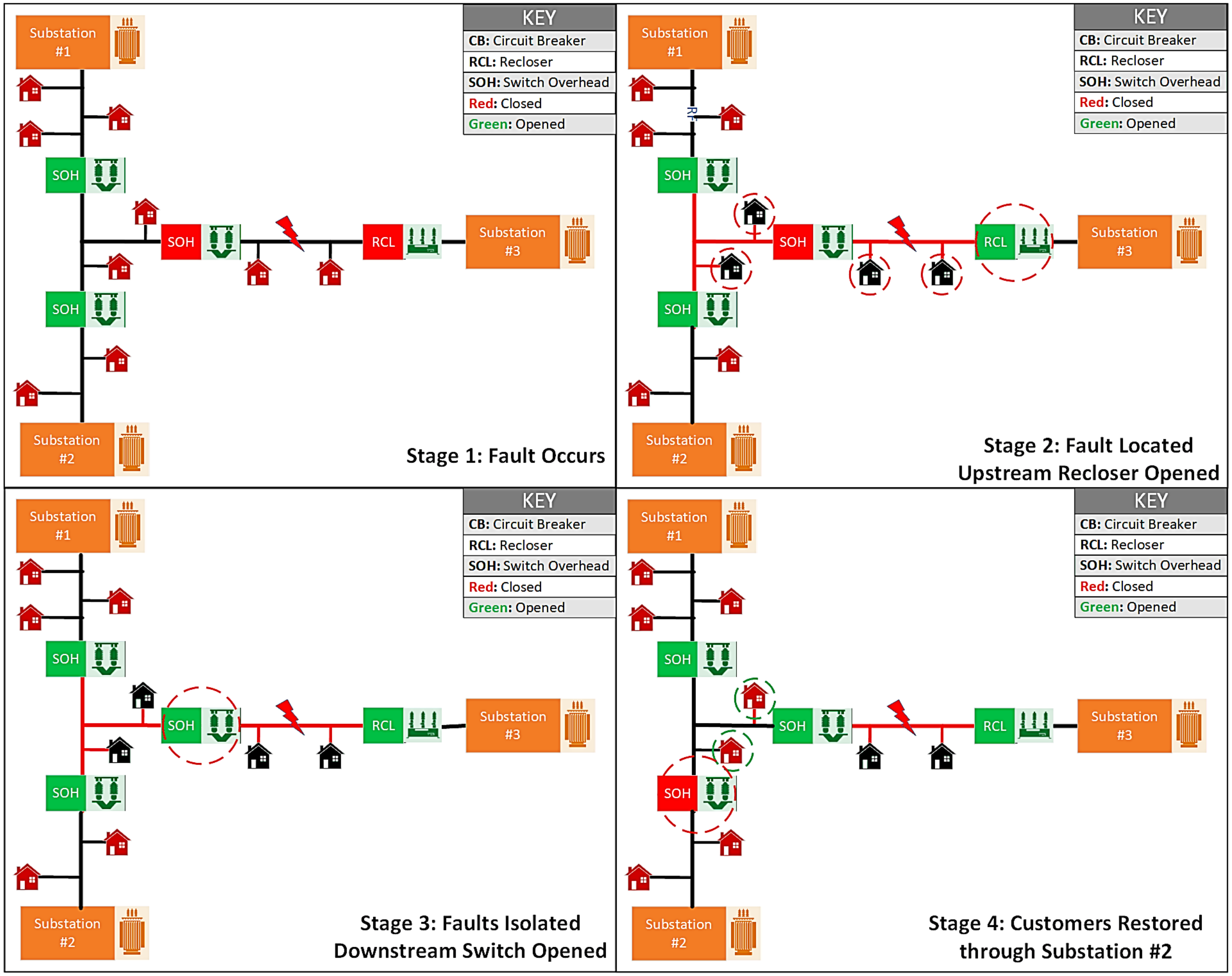


Figure 2. Typical utility fault handling process (Department of Energy 2014)

## 2.2 Blockchain Platforms

At its core, blockchain integrates a wide set of technical constructs that work together to host a distributed global state. These constructs can be briefly summarized as the immutable ledger, consensus mechanisms, credential management, state replication, and distributed system architecture. The constructs can be summarized as follows:

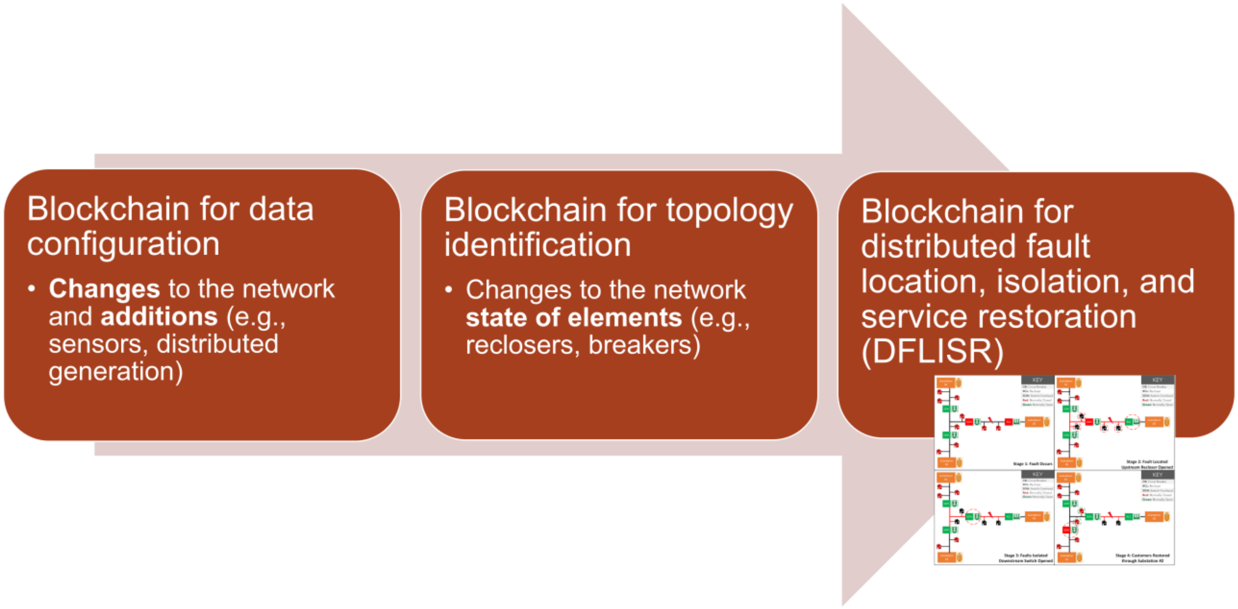
- An immutable ledger is a linked list structure, tied via hashes, that operates over a series of data blocks and supports the immutable ledger. The underlying cryptographic hashing mechanism makes it difficult to edit past records without creating hash mismatches (which are easy to detect). The ledger contains the entire recorded history and is redundantly stored across multiple peers.
- Consensus mechanisms require participants to “agree” on the actions that result in the creation of blocks and their final order. The consensus mechanism is typically tied to a particular blockchain implementation, but it can be chosen based on the requirements of the end user. In general, users must choose between permission-based solutions or fully open, decentralized solutions, each of which has advantages and disadvantages. Because they can link participants’ digital identities to physical service addresses and achieve better computational efficiencies, permissioned blockchain implementations appear to have become the preferred choice in the field of power systems.
- Credential management functions are used to provide additional security properties to blockchain implementations, such as enabling nonrepudiation by appending the digital fingerprint of the entity that requested a new block and enabling tracking during the endorsement process (i.e., which peers approved the request). Supported features are blockchain-dependent; common features include (1) supporting entity registration, renovation, and revocation (either as a self-managed procedure or via a centralized body), (2) enforcing access control via digital identities, and (3) enabling low-level application programming interface access (e.g., to propose a new smart contract).
- Distributed ledger technology depends on well-known distributed-system architecture designs to make sure that agents can communicate with each other across the network and that consensus will finally be established (subject to the CAP theorem[2]). To guarantee data consistency, several voting and data ordering methods are available and may be employed.

[2] According to the Consistency, Availability, and Partition tolerance (CAP) theorem, a distributed system can only ensure two of the three desired characteristics of distributed systems. Specifically, (a) consistency (the systems concur on the data being stored), (b) availability (each request receives a suitable answer), and (c) partition tolerance (the system continues to operate when the set of participating agents breaks).

# 3.0 Blockchain Use Case Description in Fault-Tolerant Grid Operations

## 3.1 PNNL's Identified Use Cases

Use cases envisioned by PNNL are (1) blockchain for grid-data configuration; (2) blockchain for topology identification; and (3) blockchain for DFLISR. These use cases are mutually dependent modules that are stacked to create a complete solution. This codependency is illustrated in Figure 3, where the blockchain-based grid-data configuration module is responsible for collecting measurements—it represents a measurement-based trust anchor that contains the best approximation for the current system state. Blockchain for topology identification leverages the highly trusted measurement data to identify network changes as well as possible outliers (such as out-of-sync circuit breakers). These two systems (blockchain for data configuration and blockchain for topology identification) work together to provide a highly trusted situational awareness platform, which is a requirement for supporting fault-tolerant grid applications such as DFLISR. There are other services that can leverage the situational awareness of the network, but such services are beyond the scope of this report.



Blockchain for data configuration

- **Changes** to the network and **additions** (e.g., sensors, distributed generation)

Blockchain for topology identification

- Changes to the network **state of elements** (e.g., reclosers, breakers)

Blockchain for distributed fault location, isolation, and service restoration (DFLISR)

Figure 3. Blockchain-based use cases' dependency mapping, demonstrating each module's ability to support higher-order functions, such as DFLISR

The abovementioned use cases can enhance the operator's decision-making process by hosting a highly trusted view of the system that has been vetted by multiple agents using consensus mechanisms. This trust layer can be used to complement more traditional state estimation (SE) algorithms that are found in SCADA systems. State estimation algorithms can provide error detection capabilities based on mathematical methods. According to previous research, relevant cyberphysical attacks that can affect SE algorithms include (a) compromising time-series–based measurement data, (b) tampering with state-based measurements, and (c) tampering with the network model (Bretas et al. 2017). Tampering with the network model is especially problematic, because most SE algorithms are built under the assumption that sensors can fail (e.g., report incorrect measurements or go off-line), but networks are static and can only change if a physical change occurs. Relying on an immutable ledger means the provenance of the network model can always be attested (e.g., by capturing the design modifications dictated by the planning department). Furthermore, keeping track of switching device operations allows topology mismatches to be promptly identified and flagged for manual review if needed. Once the topology and measurements can be trusted, logical control actions can take place (e.g., to coordinate distinct distributed resources to agree on the actions to be taken).

## 3.2 Distributed Fault-Tolerant Architecture Leveraging Blockchain Capabilities

Implementation of the use cases illustrated in Figure 3 requires development of a distributed architecture that can provide the infrastructure needed to develop the use cases while ensuring business constraints are met. As part of the initial design phase, a solution architect may assume that all utility agents (which can be sensors, actuators, and services) have access to a common ledger. Such a simple approach enables all utility agents to verify the behavior of the entire system; however, it would also lead to a set of undesirable traits such as the following:

- When all actors are required to validate all the measurements and network information makes the process inefficient, computational resources are wasted and incurring significant communication overheads are incurred.
- By replicating data across all agents, the information effectively becomes public. This may increase the attack surface, enabling malicious actors to acquire vast amounts of system-specific data (which would eliminate the need to devise a collection tactic).
- Consensus networks with a large number of actors become inefficient and do not increase fault tolerance. Although specific numbers are dependent on the consensus algorithm, the practical limit is often set to fewer than 20 actors.
- By relying on a single ledger, data segmentation among competitors (e.g., different DSOs) becomes problematic, requiring the use of client-side encryption to provide confidentiality while losing some of the public auditability benefits.

Given the aforementioned limitations, a segmented-ledger approach was chosen for this project. The proposed approach relies on a multilevel ledger architecture that splits information into smaller regions (effectively creating areas), with a top-level ledger used to host interarea information. A graphical summary of the proposed approach is presented in Figure 4. Within this diagram, the local area ledger contains local measurements and a reduced network model (i.e., the area network model), which enables field devices (even those with limited computational capabilities) to perform a complete assessment of their area. Naturally the state of each area is codependent with the states of other neighboring areas (e.g., from a local perspective, an upstream area provides power, while a downstream area acts as a load). This requires each area to be aware of its boundary conditions (e.g., the amount of power being transferred into other areas via tie points).

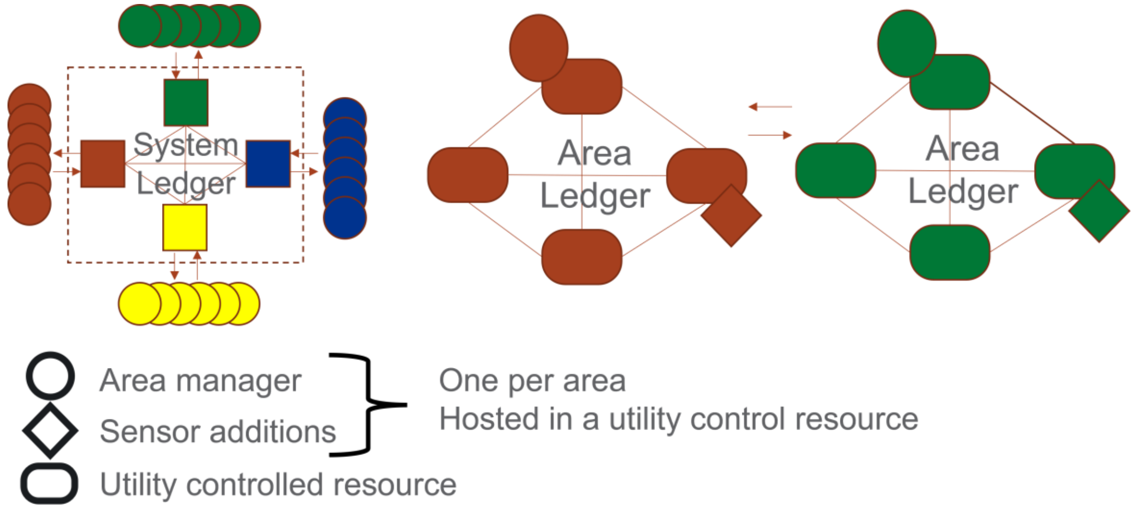


Figure 4. Distributed fault-tolerant architecture leveraging blockchain technology. The area ledger is the focus of this report.

Under the proposed methodology, each area can interact with other areas regardless of their ownership. Areas operated by different utilities can still share data as long as a physical, direct or indirect connection is possible. The system ledger contains the nearly static information, such as the area network model information (i.e., $Y_{\text{bus}}$, the nodal admittance matrix, which comprises the network topology and electrical parameters—here, the physical tie-point location), all the areas of possible physical direct or indirect connection (i.e., list of areas to communicate with, independent of ownership), and the possible physical area connections within its communication list for each area.

Broadly speaking, network information stored within the system ledger is expected to remain relatively constant, with periodic updates to reflect physical changes (e.g., to capture new line segments, tie points, and new service regions). Updates of the area ledger, however, are expected to be continuous (e.g., capturing the latest state of the area, behavior of sensors, and possible reconnections).

It is important to mention that the utility-controlled resources are responsible for evaluating and orchestrating topology changes based on local network area conditions as well as handling external reconfiguration requests. Figure 4 illustrates the computational resources hosted within each local area; these are (1) the off-the-shelf, utility-controlled resources (e.g., switching devices, metering infrastructure, etc.); (2) a dedicated area manager (which provides access to the area ledger, synchronizes data with the system-level ledger, and generally enables actors to access up-to-date information); and (3) a dedicated sensor gateway that manages the possible sensor additions.

### 3.2.1 Data Exchange Requirements

Development of the use cases follows the distributed, fault-tolerant architecture presented in Figure 4. The current phase of this development aims to demonstrate the functionality from an area perspective, thereby focusing on the local area ledger portion. The area ledger starts by verifying that an up-to-date network model (i.e., $Y_{\text{bus}}$) is available; next, it collects nodal information (e.g., nominal voltage, name, and bus type), then, an area communication list, possible area connections from the area communication list, and power flows from tie points.

The list of neighbor areas is based on tie-point information (i.e., identified areas that affect or can be affected by local decisions). The local area manager (which can be implemented via a smart contract[3]), constantly monitors the ledger state, awaits new measurements, and reports faults to initiate a DFLISR event. After a DFLISR event has been triggered, the area manager is responsible for collecting a list of feasible reconnection plans. These plans may restore services within the region itself, or could enable reconnection of an external area (in which case, the local area becomes part of a reconnection path).

By using a common ledger instead of a traditional direct, point-to-point communication architecture, the system gains the fault tolerance attributes typically associated with distributed networks. Furthermore, it eliminates the risk of message-level manipulations, which can result in intentionally altered measurements or injection of ill-intended control signals. It is important to mention that grid measurements are collected via the sensor addition gateway, but they can also be implemented via a smart contract, thereby eliminating single points of failure, and enabling sensors to work toward determining the validity of each collected measurement (via a consensus mechanism).

Actors within the network are issued credentials upon registration. This enables actors to be authenticated against their peers and enables the area manager to effectively track all actors' actions within a region. An actor's credentials are tied to their physical grid interface (e.g., the point of interconnection), which allows actors to participate only in those subregions where their observations (e.g., measurements) and actions (e.g., switching actions) have a valid purpose.

### 3.2.2 Unified Modeling Language (UML)-Based Use Case Description

From a high-level perspective, a grid operational state can be catalogued as a normal, emergency, or restorative state (Liu et al. 2014). Once a fault is detected, the affected area is isolated (e.g., via an overcurrent protection device) and the system enters a restorative state. Requiring operators (or automated mechanisms) to perform restorative actions to return the system back to the normal state. In this case, FLISR is part of the restorative process and should be triggered whenever the system is outside the normal operational scenario. During this work, it has been assumed that high-speed emergency mechanisms can operate autonomously and dependably—removing the need to explicitly handle the emergency state. A state machine representation for the FLISR algorithm based on the aforementioned state transitions is presented in Figure 5. In this case, a normal operation is defined as any circuit configuration that does not unintentionally disconnect customers. Thus, a region with a fault present, or without a power supply (due to an external fault) will enter the FLISR state. Once the area is in a FLISR state, it will only return to a normal operation state once power has been restored.

[3] Smart contracts are programs that are executed when predetermined conditions are meet.

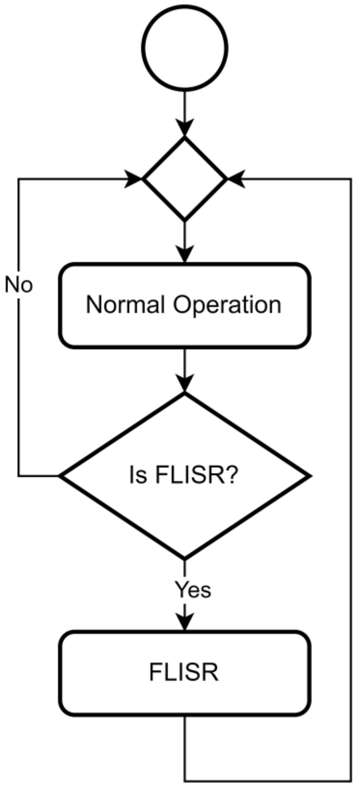


Figure 5. A simplified view of the transition function used by the FLISR algorithm

During normal behavior, the system is in “hot standby,” collecting measurements, validating the topology, and performing background health checks to make sure the system is prepared for a FLISR event. As part of this “normal” operational state, any of seven subfunctions can be executed, depending on the message type and current system conditions. The conditions that trigger each of these subfunctions are illustrated in Figure 6. The subfunctions are attempts to add information to the area ledger. These subfunctions are intended to fulfill use-case–specific tasks:

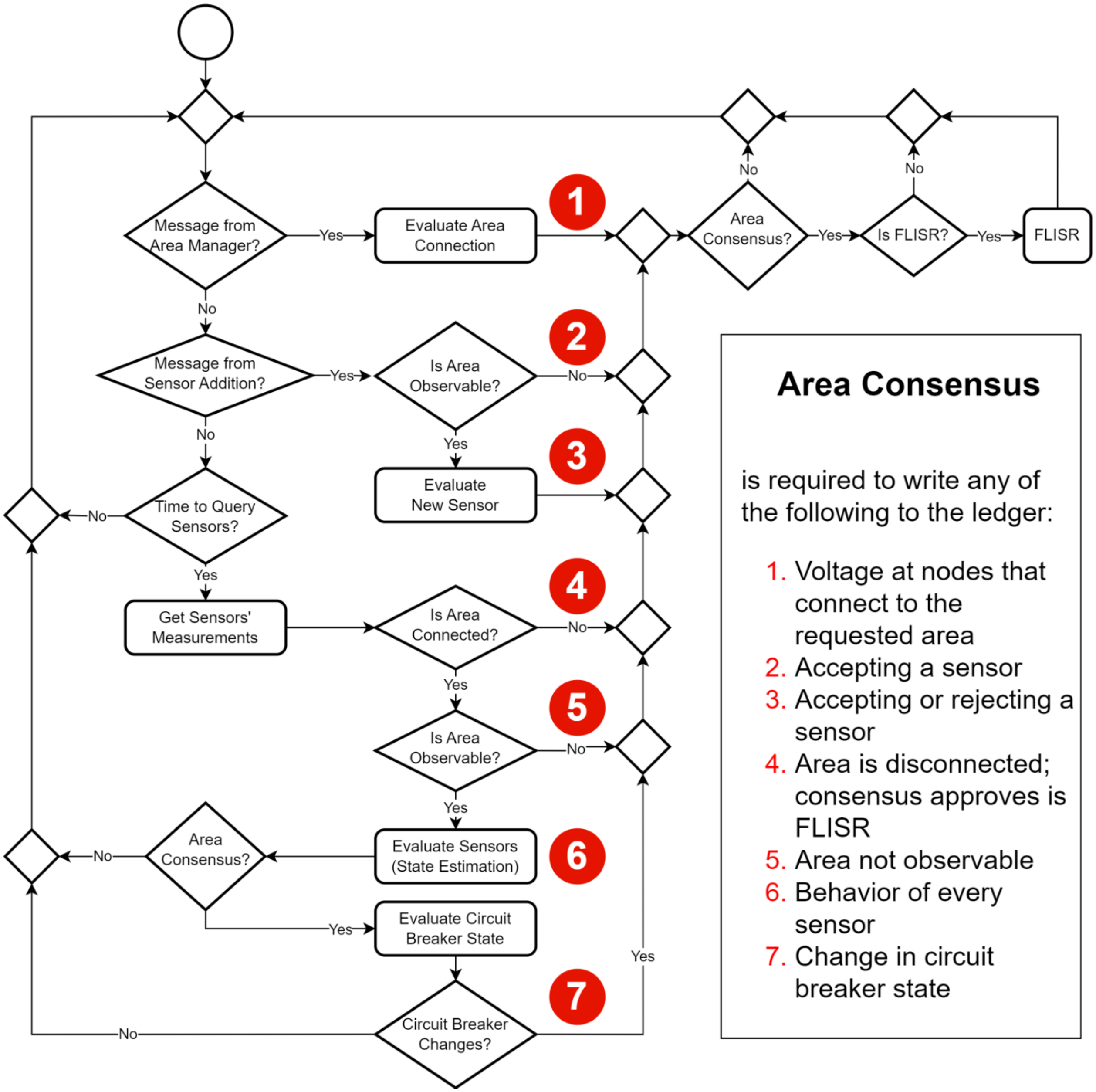
1. Subfunctions 2, 3, 5, and 6 are responsible for handling the blockchain-based data configuration use case.
    a. Subfunction 2 handles the registration of a new sensor, when the system is not observable (e.g., no other nearby sensors can be used to validate the reported measurement). Adding a sensor to an area sensor list requires consensus.
    b. Subfunction 3 handles the registration of a new sensor when its addition can be attested by another sensor (e.g., a measured quantity is also observable by an independent actor). The area consensus requires the area utility control devices to agree whether or not the sensor is behaving properly (i.e., should be added to the list of area sensors).
    c. Subfunction 5 handles the case of a service area that cannot be observed, even if measurements are available. This subfunction can be used to handle an area that is connected but is not observable. Consensus is needed to add a current area state to the ledger.

   d. Subfunction 6 is responsible for computing SE. This only occurs when the area is observable. Consensus is needed to add the trusted field sensors that are available in the area sensor list.

2. Subfunction 7 handles the blockchain for a topology identification use case.
   a. Subfunction 7 is only called when the sensors’ measurements have been validated and the state of a circuit breaker is being audited (e.g., to validate the network topology). This can be used to find mismatches between the physical topology and the topology captured by the ledger.

3. Subfunctions 1 and 4 are used by the FLISR use case.
   a. Subfunction 1 is responsible for calculating the voltage drop (and any other security constraint) when the local area is about to be reconnected or the area is being considered as part of a re-energization path.
   b. Subfunction 4 proactively evaluates the system conditions to determine whether the area is disconnected. If the area is isolated, it will transition into the FLISR handling mode.

All of the above functions are intended to work in a distributed ledger technology environment, because such decisions (ledger writes) require the consensus of qualified actors within the service area (qualified actors represent agents with valid credentials and local ledger access).



**Area Consensus**

is required to write any of the following to the ledger:

1. Voltage at nodes that connect to the requested area
2. Accepting a sensor
3. Accepting or rejecting a sensor
4. Area is disconnected; consensus approves is FLISR
5. Area not observable
6. Behavior of every sensor
7. Change in circuit breaker state

Figure 6. Area behavior during normal operation. The red circles indicate attempts to add information to the area ledger (i.e., requires area consensus).

As shown in Figure 6, the FLISR handling mode is triggered when an area becomes de-energized (i.e., when an upstream connection is lost). Once the algorithm transitions into the FLISR handling mode, the algorithm’s primary goals are redefined and a time-limited restoration process begins. A highly simplified overview of this process is illustrated by Figure 7, which highlights the presence of a timer interrupt, a feature intended to automatically trigger a lockout condition that enables operators and field personnel to manually restore services without the risk of an automated action taking place.

Since the area is initially de-energized (e.g., an open switch condition exists upstream), the initial behavior is to completely isolate the area (by opening all the tie switches) and then initiate a time delay to make sure neighboring areas become aware of the tie switch changes. After the delay, the local ledger queries the system-level ledger to collect the substation's state, and to collect states from adjoining areas. Once this initial data collection process is completed, each local area has all the information needed to independently determine (a) whether the fault is located inside the operational boundaries or outside them (e.g., in another operational area).

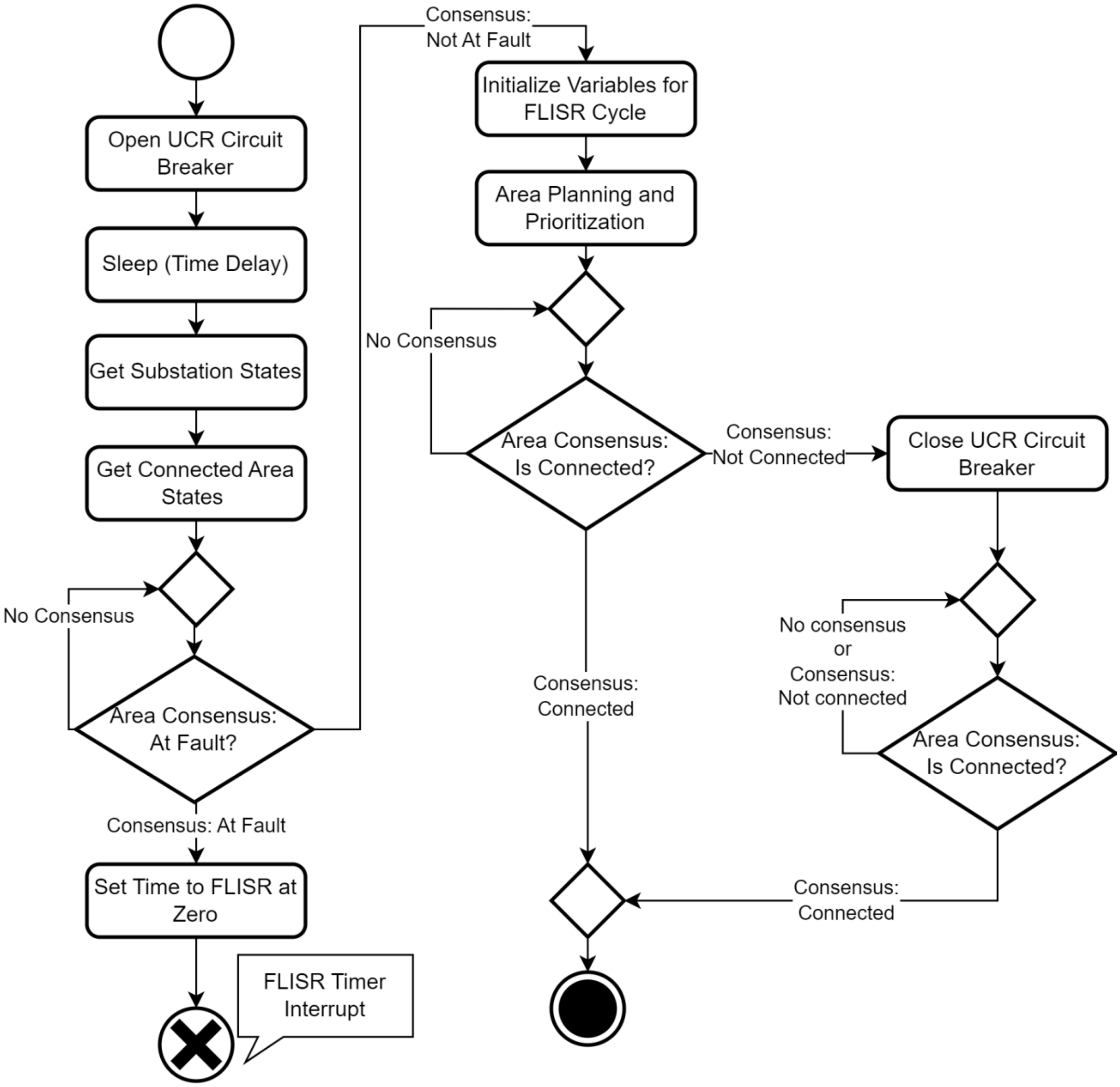

Open UCR Circuit Breaker
Sleep (Time Delay)
Get Substation States
Get Connected Area States
No Consensus
Area Consensus: At Fault?
Consensus: At Fault
Set Time to FLISR at Zero
FLISR Timer Interrupt
Consensus: Not At Fault
Initialize Variables for FLISR Cycle
Area Planning and Prioritization
No Consensus
Area Consensus: Is Connected?
Consensus: Not Connected
Close UCR Circuit Breaker
Consensus: Connected
No consensus or Consensus: Not connected
Area Consensus: Is Connected?
Consensus: Connected

Figure 7. Overall area behavior during the FLISR event

As expected from a blockchain-based solution, this preliminary diagnosis is done by independent agents, and the result is stored within the local ledger (via a consensus mechanism). If a fault is detected within the local area, the FLISR application enters a lockout state until the maintenance crew clears the fault, performs re-energization checks, and eventually restores services (see Figure 8 for details). This lockout condition can also be activated if a consensus cannot be reached within a reasonable time (e.g., if a communication failure occurs), or if the FLISR algorithm times out (e.g., when no feasible topology configuration can be found).

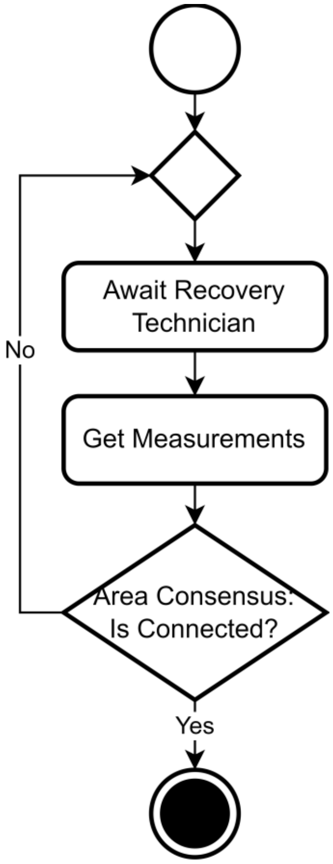


Figure 8. FLISR maintenance standby loop. Time to FLISR is zero.

After confirmation that the area is de-energized and the fault has been determined to be outside the local area's boundaries, the main FLISR algorithm starts to operate. The logical steps for this case are summarized on the right-hand side of Figure 7, with a more detailed view presented in Figure 9. The FLISR algorithm relies on multiple loops to compute a "restoration plan"; this restoration plan is a coordinated effort that has been agreed upon by multiple areas that seeks to maximize the number of reconnected customers while obeying the power transfer limits of each feeder.

The first loop in Figure 9 is responsible for maintaining up-to-date system information (i.e., the grid "state") and creating a list of disconnected areas (i.e., those that need to be re-energized). If the set of disconnected areas has changed from the previous state (i.e., at the beginning of the procedure in Figure 7) and consensus is achieved on the contents of the set, the "Time to FLISR" is increased with Event 1. Accurate knowledge of the states of areas is required for deciding which paths are feasible, given that only areas in the normal state can be included in a reconnection path. Once there is consensus on which areas are available and their reconnection order, the loop is concluded, leading to the increase of "Time to FLISR Event 2." Every area has a list of critical loads that must be served at all times; the list will often include emergency services buildings and health-related infrastructure.

The second loop in Figure 7 occurs for postponing the identification of possible reconnection paths if the area is not the next to be reconnected. In that case, the reconnection order can be adjusted. The area reconnection order is adjusted when it is possible to reconnect the area with the current system state and a given priority delay/pause time has passed. This serves to preserve the priority order and also to change it in case it is not possible or too much time has passed. The third loop makes sure the area agrees with the possible area reconnection paths. Once consensus is achieved on the possible paths of reconnection, the Time to FLISR is increased with Event 3.

The fourth and final loop of the area planning and prioritization process is intended to identify and select viable re-energization paths. The loop is started by collecting data from areas on the possible paths. If the voltage at the connection node is not available, the approach is tested in case any of the paths have improved (i.e., if the assessment of the possible connection is moving forward) and the Time to FLISR is increased with Event 5. In case the voltage at the connection node is available and this is the first time it has been available, the Time to FLISR is increased with Event 4. Using the voltage at the connection node, the possible connection for that area is evaluated. Once consensus is achieved, the connection is considered possible or not possible.

Once a possible path is identified, the area planning and prioritization block is concluded, continuing the overall process according to Figure 7. Before any action begins, the area is again tested for connection. If the area is not connected, the reconnection path previously developed is put into action and the designated circuit breaker is closed to reestablish connection. The behavior of the area goes to normal only when there is consensus that the area is connected.

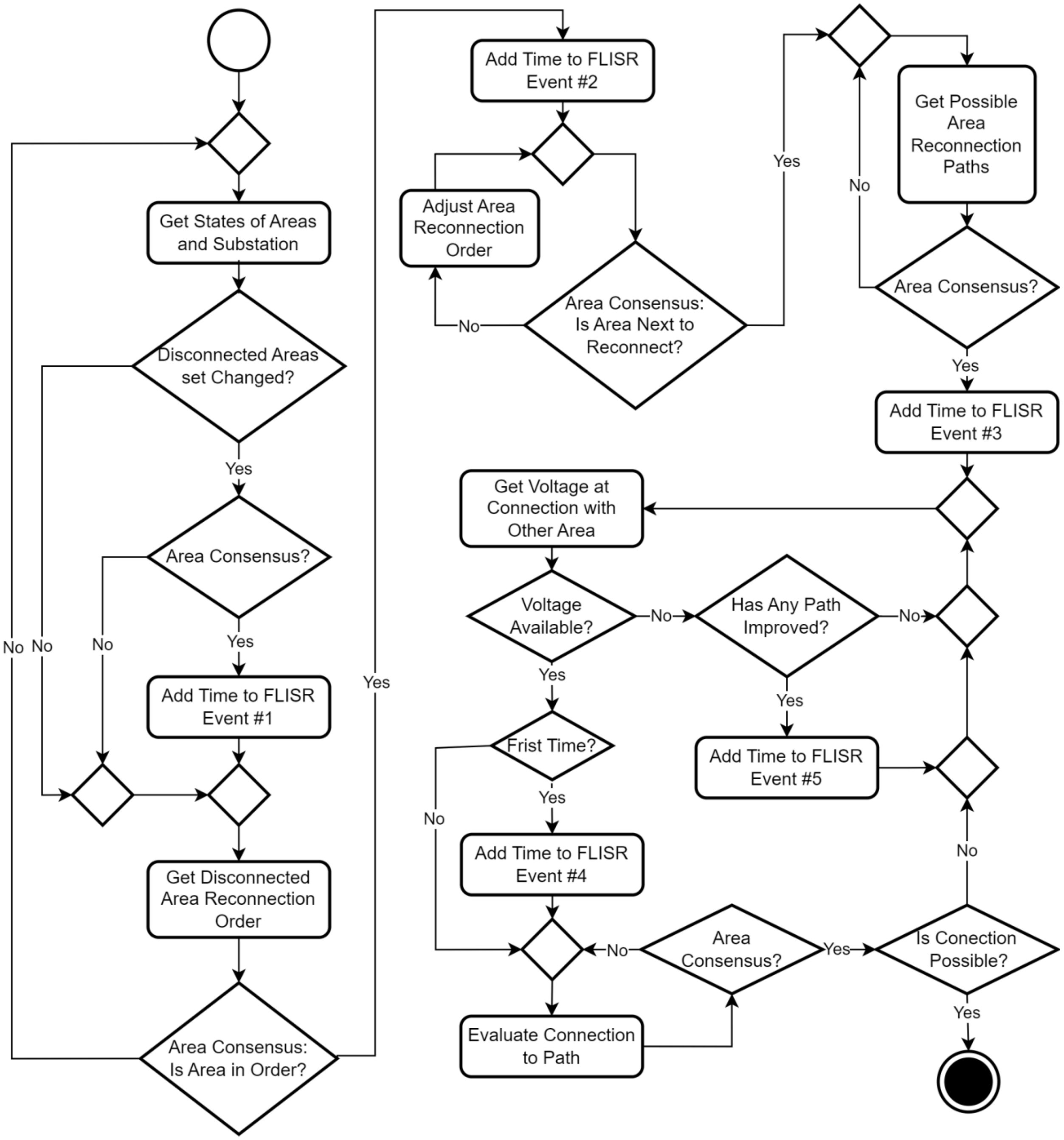

Get States of Areas and Substation
Disconnected Areas set Changed?
Yes
Area Consensus?
Yes
Add Time to FLISR Event #1
No
No
No
Get Disconnected Area Reconnection Order
Area Consensus: Is Area in Order?
Yes
Add Time to FLISR Event #2
Adjust Area Reconnection Order
Area Consensus: Is Area Next to Reconnect?
No
Yes
No
Get Possible Area Reconnection Paths
Area Consensus?
Yes
Add Time to FLISR Event #3
Get Voltage at Connection with Other Area
Voltage Available?
No
Has Any Path Improved?
No
Yes
Frist Time?
Yes
No
Add Time to FLISR Event #4
Yes
Add Time to FLISR Event #5
Area Consensus?
No
Yes
Is Conection Possible?
No
Yes
Evaluate Connection to Path

Figure 9. FLISR area planning and prioritization process

## 3.3 State Estimation by the FLISR Application

### 3.3.1 Introduction to the Weighted Least Squares Method for State Estimation

State estimation enables a digital system to compute the best approximation of the current system state(s) by merging multiple observations. In distribution grids, these are usually in the form of measurements and pseudo-measurements (e.g., measurements derived from indirect observations). The state variables for the distribution networks are the voltage phasors of the nodes, denoted $x$ in Equation 1:

$$x^T = [\theta_2, \quad \theta_3, \quad \dots, \quad \theta_N, \quad V_1, \quad V_2, \quad \dots, \quad V_N] \tag{1}$$

where $\theta_i$ and $V_i$ represent the phase angle and voltage magnitude of node *i,* and each node *i* may consist of multiple phases $\varphi_k$ (such that $\varphi_k \in$ {a, b, c}). *N* is the total number of nodes in the system. The phase angle of the first node is considered the reference; hence, it is not included in the state vector. In this work, the weighted least squares (WLS) method is used to solve the SE problem (Haji and Ardakanian 2019) and the process involves solving over a determined set of nonlinear algebraic equations, as given by Equation (2).

$$z = h(x) + e \tag{2}$$

where

$$z = \begin{bmatrix} z_1 \\ z_2 \\ . \\ . \\ . \\ z_m \end{bmatrix}, \qquad h(x) = \begin{bmatrix} h_1(x_1, x_2, \dots., x_n) \\ h_2(x_1, x_2, \dots., x_n) \\ . \\ . \\ . \\ h_m(x_1, x_2, \dots., x_n) \end{bmatrix}, \qquad e = \begin{bmatrix} e_1 \\ e_2 \\ . \\ . \\ . \\ e_m \end{bmatrix} \tag{3}$$

and where *z* is the vector of measurements, $h_m(x)$ is a continuous function that relates the state vectors ($x$) to the $m^{th}$ measurement, and $e_m$ is the error associated with the accuracy of a particular sensor. In the WLS approach to SE, the errors are assumed to be independent, random variables with zero mean and finite variance that is based on the type of the measuring equipment.

The WLS approach focuses on finding a system state that minimizes the difference between the measurements (z) and the corresponding values from the measurement functions (h). The classical approach of WLS-based SE aims at minimizing the following objective function through an iterative process:

$$J(x) = [z - h(x)]^T R^{-1} [z - h(x)] \tag{4}$$

The inverse of the error covariance matrix (R) is multiplied with the errors ($e$) to provide the sensors and their expected accuracies. Sensors with high precision end up having a higher confidence associated with their measurements. The higher the confidence associated with a measurement, the larger is its effect on the estimated states. The outputs of the SE are the estimated states, the uncertainty of estimated states, and the sensor uncertainties.

$$R = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 \\ 0 & 0 & 0 & .. & 0 \\ 0 & 0 & 0 & 0 & \sigma_m \end{bmatrix} \tag{5}$$

where *R* is a diagonal covariance matrix, in which each diagonal element σ corresponds to the variance associated with a particular measurement. The iterative process minimizes the value of the objective function (Equation (4)), a process that involves determining the Jacobian matrix H, which is the partial derivatives of the measurement functions with respect to the individual system states. The state estimates are then updated iteratively from their starting values, based on Equation (6), until convergence is achieved.

$$
\begin{gathered}
\Delta x^{k+1} = (H^T R^{-1} H)^{-1} H^T R^{-1} e, \\
where\ \ H = \frac{\partial h(x)}{\partial x}\ \ and\ G = (H^T R^{-1} H) \\
x^{k+1} = \left(x^k + \Delta x^{k+1}\right)
\end{gathered} \tag{6}
$$

#### 3.3.1.1 State Estimation for Unbalanced Distribution Networks

For unbalanced three-phase distribution networks, the SE problem is formulated considering voltage magnitudes and active and reactive power injection and flows as measurements. The problem formulation, given in Equation (2), can be modeled using the functions for the chosen measurements, as shown in Equation (7):

$$
z = \begin{bmatrix} z_1 \\ z_2 \\ . \\ . \\ . \\ z_m \end{bmatrix} = h(x) + e;\ S.T.\, h(x) = \begin{bmatrix} P_n^{\varphi_k}(x) \\ .. \\ Q_n^{\varphi_k}(x) \\ .. \\ P_{ij}^{\varphi_k}(x) \\ .. \\ Q_{ij}^{\varphi_k}(x \\ .. \\ V_n^{\varphi_k}(x) \end{bmatrix}, e = \begin{bmatrix} P_n^{\varphi_k}{}^{meas} - P_n^{\varphi_k} \\ .. \\ Q_n^{\varphi_k}{}^{meas} - Q_n^{\varphi_k} \\ .. \\ P_{ij}^{\varphi_k}{}^{meas} - P_{ij}^{\varphi_k} \\ .. \\ Q_{ij}^{\varphi_k}{}^{meas} - Q_{ij}^{\varphi_k} \\ .. \\ V_n^{\varphi_k}{}^{meas} - V_n^{\varphi_k} \end{bmatrix}, \forall\ \varphi^k \epsilon\ \{a, b, b\} \tag{7}
$$

where the superscripts $^{meas}$ represent the measurements, $P_n^{\varphi k}$ and $Q_n^{\varphi k}$ represent the active and reactive power injection at a bus *n* for phase $\varphi_k$, and $P_{ij}^{\varphi k}$ and $Q_{ij}^{\varphi k}$ are the active and reactive power flows through a branch *ij* for phase $\varphi_k$. Considering the inherent characteristics of distribution networks, the model considers all system phases (a, b, c), denoted by $\varphi_k$ and $\varphi_l$. The active and reactive power flows and injection values for a given estimate of states are determined using Equation (8):

$$
\begin{aligned}
P_i^{\varphi_k} &= V_i^{\varphi_k} \sum_{\varphi_l \in \{\mathrm{a,b,c}\}} \sum_{j \epsilon N} V_j^{\varphi_l}\left(G_{ij}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l} + B_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l}\right) \\
Q_i^{\varphi_k} &= V_i^{\varphi_k} \sum_{\varphi_l \in \{\mathrm{a,b,c}\}} \sum_{j \epsilon N} V_j^{\varphi_l}\left(G_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l} - B_{nm}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l}\right) \\
P_{ij}^{\varphi_k} &= V_i^{\varphi_k} \sum_{\varphi_l \in \{\mathrm{a,b,c}\}} \Big[-V_i^{\varphi_l}\big(G_{ij}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l} \\
&\qquad + B_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l}\big) + V_j^{\varphi_l}\left(G_{ij}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l} + B_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l}\right)\Big] \\
Q_{ij}^{\varphi_k} &= V_i^{\varphi_k} \sum_{\varphi_l \in \{\mathrm{a,b,c}\}} \Big[-V_i^{\varphi_l}\big(G_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l} \\
&\qquad - B_{ij}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l}\big) + V_j^{\varphi_l}\left(G_{ij}^{\varphi_k\varphi_l} \sin\theta_{ij}^{\varphi_k\varphi_l} - B_{ij}^{\varphi_k\varphi_l} \cos\theta_{ij}^{\varphi_k\varphi_l}\right)\Big]
\end{aligned} \tag{8}
$$

where $B_{ij}^{\varphi k \varphi l}$ and $G_{ij}^{\varphi k \varphi l}$ are the real and imaginary elements for the bus admittance matrix $Y_{\mathrm{bus}}$. During the iterative process of solving for the objective in Equation (4), the matrix *H* is obtained through partial derivates of the measurement functions presented in Equation (8) with respect to the system states ($V_n^{\varphi k}$ and $\theta_n^{\varphi k}$). For voltage measurements, a particular measurement's partial derivative in relation to the voltage magnitude is equal to unity and all other corresponding entries in that row of the *H* matrix are equal to zero. The expression for partial derivates for the active and reactive power flows and injections, given in Equation (9), can be computed and the entries in the row of the *H* matrix are updated based on values for iterative estimates of the system states. The iteration continues until the values of the estimated states converge.

#### 3.3.1.2 Observability Analysis for State Estimation

Observability analysis is an important part of the SE process objectives, because it involves determining whether the system states can be estimated with reasonable accuracy for a given set of measurements. This work adopts the numerical method, which is based on the analysis of the column rank of the Jacobian matrix (Brinkmann and Negnevitsky 2016). If the number of independent measurements is equal to or greater than the number of state variables, the column rank of the network is full, and for such cases the network is found to be observable. If the column rank is not full, the network is classified as unobservable.

In this work, the number of independent sensors must be greater than the number of estimated states. A secondary condition is that at least one of the sensors must be a three-phase voltage magnitude. This condition is required to permit the SE solution to be independent of a forced assumption of voltage magnitude at the point of common coupling.

#### 3.3.1.3 Gross Error Detection

Once observability is confirmed, the current system state(s) are estimated based on the WLS-based SE algorithm. However, the accuracy of the state estimates depends on the errors associated with the measurement, which can originate from various sources such as equipment malfunction, communications losses, human errors, and even malicious cyberphysical attacks. Therefore, directly using the estimated states without any gross error detection would lead to faulty control decisions for the applications (here FLISR).

In this work, we adopt the chi-squared ($\mathrm{X}^2$) method for gross error estimation. The chi-squared gross error detection is based on the principle that a set of normally distributed random variables can be represented using a single random variable having a $\mathrm{X}^2$ distribution (Angulo-Paniagua and Quirós-Tortós 2020). The sum of squares of the normalized errors associated with the measurements can be represented as shown in Equation 9:

$$f(x) = \sum_{i=1}^{m} R_{ii}^{-1} e_i^2 = \sum_{i=1}^{m} \left( \frac{e_i}{\sqrt{R_{ii}}} \right)^2 \tag{9}$$

where *m* is the number of measurements, $e_i$ corresponds to the error associated with the accuracy of a particular measurement, and $R_{ii}$ is the diagonal entries of the measurement error covariance matrix *R*. For distribution system SE, the measured errors are commonly assumed to be normally distributed, random variables with zero mean and a variance of $R_{ii}$. Therefore *f(x)* will have a $\mathrm{X}^2$ distribution with a maximum of ($m - n$) degrees of freedom, with *n* being the number of states to be estimated. To detect any gross error in the estimated states, *f(x)* is approximated based on the state estimates as shown in Equation (10).

$$f(x) = \sum_{i=1}^{m} \left( \frac{z_i - \; h_i(x_1, x_2, \dots., x_n)}{\sigma_i} \right)^2 \tag{10}$$

The presence of errors will be identified if (a) the calculated value *f(x)* exceeds the threshold value of the chi-square distribution for a given number of degrees of freedom ($m - n$) and (b) the *p*-value (probability of deviations from the expected values); these conditions indicate that the observed deviations are significant. In this work, a *p*-value of 0.05 is chosen. Once gross error is detected, the measurement corresponding to the highest absolute value of normalized error $\left(\frac{\boldsymbol{e_i}}{\sqrt{\boldsymbol{R_{ii}}}}\right)$ is removed, and the system states are computed again using the updated set of measurements. This process is repeated until no gross error is detected for the estimated system states.

### 3.3.2 Leveraging State Estimation to Evaluate the State of Circuit Breakers

As part of this work, it has been assumed that all circuit breakers possess voltage and current sensing capabilities and therefore can measure the flow of power across their terminals. These power flows are used as part of the SE process, and although an incorrect measurement may not be large enough to trigger a gross error warning, the algorithm can still detect errors by comparing the expected error (a nameplate value) to the computed error in the SE process.

### 3.3.3 Leveraging State Estimation to Compute the Voltage on Areas to be Reconnected

The UML activity diagrams presented in Section 3.2.2 document the voltage-drop calculation function, which relies on treating downstream areas as loads, and then computing the terminal voltage (i.e., using a backward/forward sweep algorithm). This approach requires the distribution system to be radial, a topology that most feeders follow, though some distribution systems operate using a ring topology, which would require introduction of a more robust voltage drop calculation method. However, a benefit of the proposed voltage drop calculation method is its relatively simple input data requirements. Each area only needs to know its own demand profile (with node-level granularity) and the net demand of the downstream areas that need to be fed (e.g., the net load of each downstream tie point). Such an approach helps to maintain interarea consumer privacy by only requiring an “aggregated” demand value and the computed tie-point voltage magnitude to be shared among adjacent areas. Furthermore, by assuming that the voltage magnitude at the tie-point connecting the first feeder area to the substation is around 1.0 PU and is relatively constant (e.g., via a voltage regulator or a tap-changing autotransformer), a solution can be achieved without considering the upstream transmission system.

To demonstrate the approach, let us assume that a radial feeder is split into three areas, with each area being connected to others via a tie point (see Figure 10). The procedure for computing the voltage (and thus assessing feasibility) for Area 3 is as follows:

1. Area 1 knows the source bus voltage, the per-node load breakdown of Area 1, and the expected load from Area 2 and Area 3 as seen from the interconnection bus (i.e., the tie point). It is therefore possible to estimate the voltage magnitude at all nodes (including the tie points) within Area 1.
2. Area 1 provides the voltage magnitude at the interconnection bus if the voltage profile satisfies its area specification.
3. Area 2 uses the interconnection voltage provided by Area 1. It also knows the load for each node in Area 2, and the expected load from Area 3 (which appears as a load attached to the tie point); hence, Area 2 can estimate the voltage magnitude of all its nodes via SE.

4. Area 2 provides the voltage magnitude at the interconnection bus if the voltage profile satisfies its area specification.
5. Area 3 has access to the voltage provided by Area 2 and a detailed breakdown of the loads present in Area 3; it therefore can estimate the bus voltage magnitudes within Area 3 via the SE. If the voltage profile satisfies the specification, the reconnection path is viable and can be recorded as a valid option within the ledger.

In the above-described example, there are two areas in the reconnection path to Area 3 requiring multiple steps for assessing the reconnection viability. Naturally, the number of steps will depend on the number of areas involved in a reconnection path (therefore, $O(n)$ operations will be needed). The approach is performed by fully independent actors, (each of which is a member of an area). The system information is obtained from the system ledger at initialization, and eventually when changes occur. The information used is from the areas of possible physical direct and indirect connections, regarding

- possible area connections and substation connections. This enables the area attempting to reconnect to create the possible reconnection paths.
- the net demand by area.
- areas' tie-point connections. This provides a location for placing the net demand.

This results in a FLISR approach that does not need to communicate with the system-level ledger during an event.

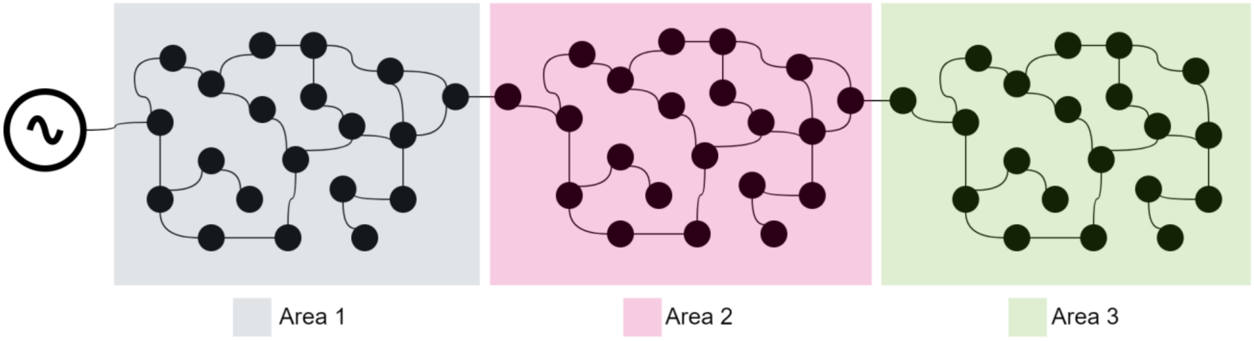


Figure 10. Voltage drop calculation procedure for radial systems. Black dots represent nodes.

# 4.0 Implementation

## 4.1 Message Passing Environment

A prototype implementation of the use cases was developed using the Python programming language. The prototype implementation uses the “actor” model: a mathematical model of concurrent computation that defines the actor as its primitive for concurrent computation (Hewitt et al. 1973). The prototype implementation uses the Pykka implementation (Jodal 2023) of the actor model for the Python programming.

Every actor has its own mailbox to store messages that it receives. In response to a message that is received, an actor can make local decisions, create more actors, send more messages, and/or decide how to respond to the next message that is received. Actors can only modify their own local state, which is private to each actor, but may affect the states of other actors indirectly through message passing. An advantage of the actor model is that it does not require the use of lock-based synchronization mechanisms to enforce a mutual exclusion concurrent control policy (Peyton Jones 2007).

When using the actor model for concurrent computation, the actors themselves are related via a supervision tree, as shown in Figure 11. When an actor creates another actor, the parent becomes the “supervisor” of the child. If a child actor fails, then the parent can decide what action(s) to take, such as creating a new actor, ignoring the failure, and/or propagating the failure to its own parent. In this way, the supervision tree enables fault tolerance within the software system.

When the actor model is applied to blockchain technologies, there are several natural candidates for the actors themselves. For example, both the blockchain ledger and the message broker that manages the transactions that are applied to the blockchain ledger can be modeled as actors. The users of the blockchain ledger are also actors, and the messages that are sent and received between them correspond to the calls to the methods of smart contracts for the blockchain application.

In the context of grid operations, modeled actors can include the grid itself, the nodes, substations, circuit breakers, sensors, etc., with the topology of the grid reflected by the structure of the corresponding supervision tree and the directed connections between the actors (depicted in Figure 12).

The directed connections between actors can themselves be modeled as actors. They are modeled as actors that store the messages they receive and forward them to other actors (depicted in Figure 12). By modifying the local state of the directed connection, faults can be introduced, such as delayed, damaged, reordered, repeated, or lost messages.

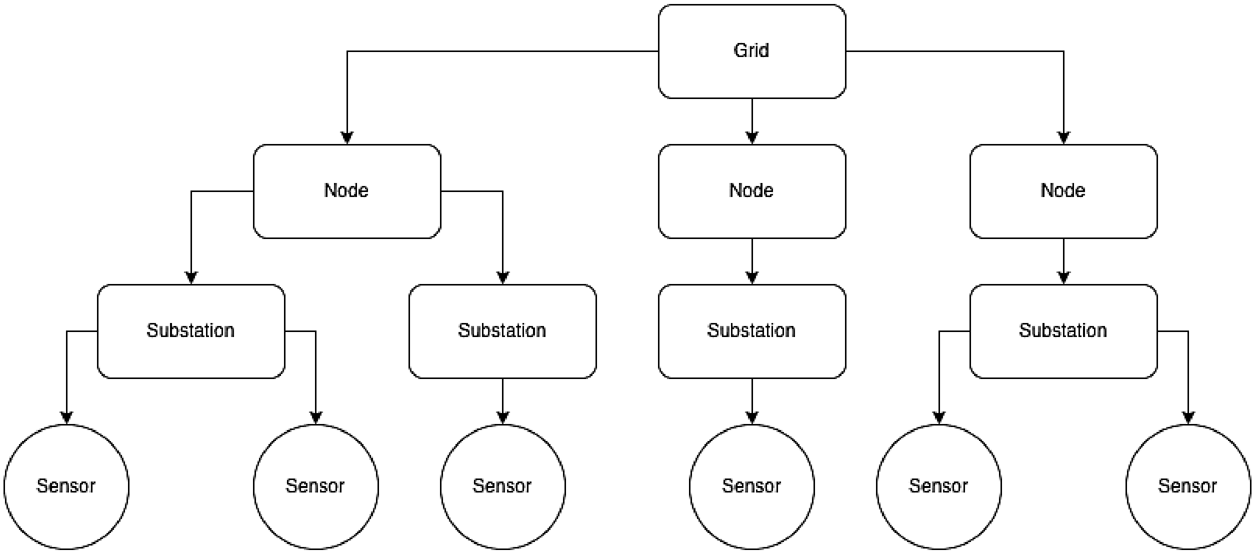


Figure 11. Structure of an example supervision tree. Solid arrows represent parent-child relationships.

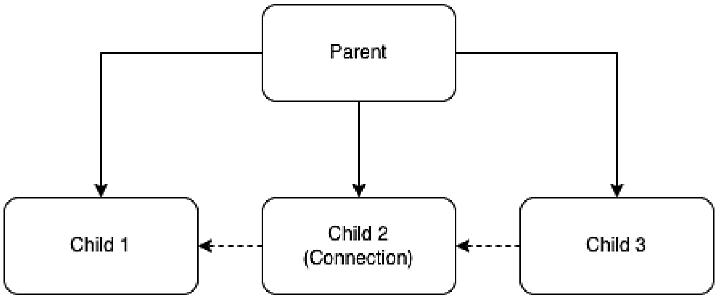


Figure 12. Structure of an example supervision tree where dashed arrows represent directed connections and solid arrows represent parent-child relationships.

Directed connections between actors are created in three steps:

1. The parent creates the first child, which will receive the messages.
2. The parent creates the second child, which will store and forward the messages. The local state of the second child includes a reference to the mailbox for the first child.
3. The parent creates the third child, which will send the messages. The local state of the third child includes a reference to the mailbox for the second child.

Finally, the prototype implementation uses the TCLab package (Kantor and Sandrock 2018) for Python to coordinate simulation time for the actors.

## 4.2 Testbed

To test the validity of the proposed methodology, the accuracy of the methods was assessed on a distribution network. The Midwest 240-Node test distribution system has 240 primary network nodes and 23 miles of primary feeder conductors; it is based on a distribution network from the Midwest U.S. and is publicly available (Wang 2019). In addition to the real network data, a year’s worth of smart meter measurements at the node level are also available. The one-year, minute-resolution, appliance-level load data was generated as described in Bhattarai et al. (2021) and is based on the nodal, hourly smart meter data. Compared to the nodal hourly smart meter data, it has a mean absolute percentage error of 2.58%. The minute-resolution load data and GridLAB-D (Pacific Northwest National Laboratory 2022) version of the Midwest 240-Node test distribution system are also publicly available (Dos Reis et al. 2021a).

The Midwest 240-Node test distribution system is radially configured and consists of three feeders. The feeders are labeled as S, M, and L, referring to the relative sizes of the feeders as small, medium, and large, respectively. Feeders M and L have shunt capacitor banks for voltage regulation. The test system has nine circuit breakers, as illustrated in Figure 13, which are used to partition the distribution system into six areas. Six of the circuit breakers are normally closed, and three are normally open. The Midwest 240-Node test distribution system serves 1,120 homes. There are 193 system load nodes: 15 on Feeder S, 44 on Feeder M, and 134 on Feeder L, each with a unique ID from 0 to 192. Table 1 presents an overview of the number of homes and peak load for every area. The peak area load is identified for the time the area used the largest amount of active power over the year.

Table 1. Midwest 240-Node test distribution system area overview

| | | | Area Peak Load | |
|---|---|---|---|---|
| Feeder | Area | Number of Homes | kW | kvar |
| S | 1 | 76 | 79.41 | 26.87 |
| M | 2 | 85 | 36.39 | 9.63 |
| M | 3 | 23 | 48.06 | 15.91 |
| M | 4 | 262 | 109.61 | 34.25 |
| L | 5 | 292 | 281.39 | 95.75 |
| L | 6 | 382 | 289.87 | 95.18 |

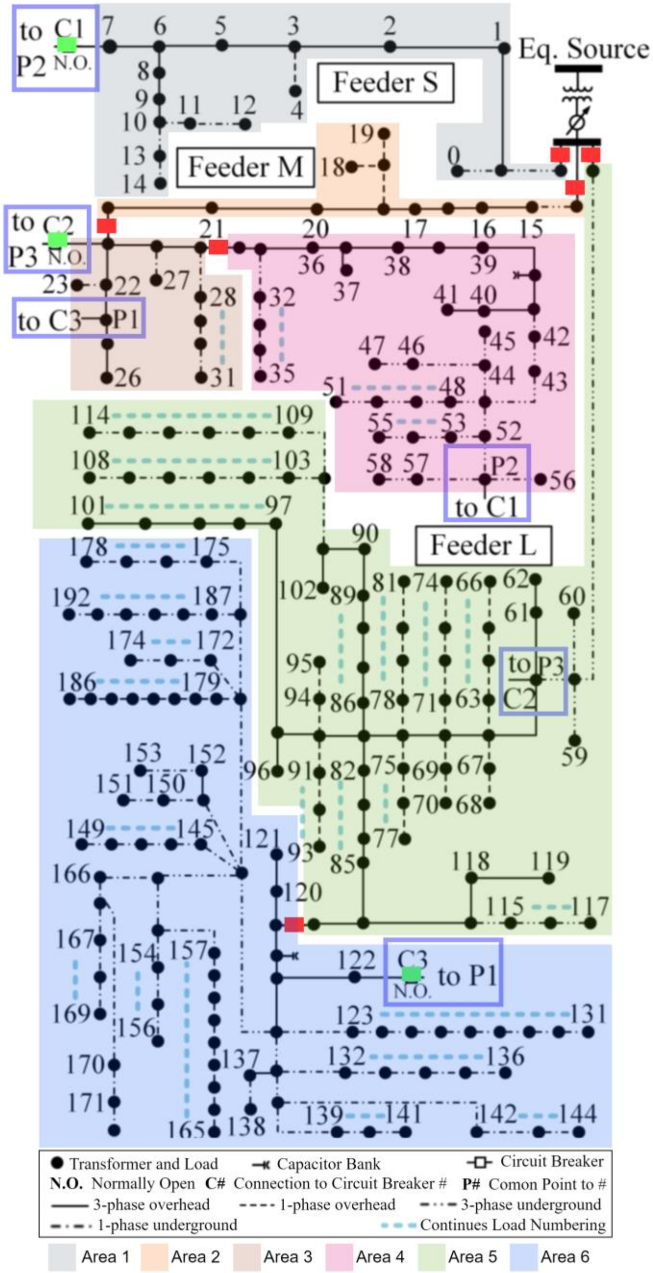


Figure 13. Midwest 240-Node test distribution system. Circuit breakers highlighted in green are normally open and those highlighted in red are normally closed. Image adapted from Dos Reis et al. (2021b).

The main benefits of choosing the Midwest 240-Node test distribution system over similar models are its open access nature, access to load profiles based on real smart meter data, and the existence of multiple feeders and multiple switch-delimited areas, which are ideal for testing the proposed algorithm. In addition, each load bus is broken into the individual houses that are served (i.e., the customers). Therefore, the Midwest 240-Node test distribution system is an ideal candidate for assessing the blockchain-based FLISR capability to reconnect customers.

### 4.2.1 Sensors on the Test System and Uncertainty Determination

The Midwest 240-Node test distribution system contains multiple sensors. Three types of time-series sensors are used; these are voltage magnitude, active power, and reactive power. The voltage magnitude relates the nodal voltage to the common reference, and is exclusive to the node. The active power and reactive power sensors can inform the injection or consumption of power at a node or the flow of power between two nodes. Circuit breakers can measure voltage magnitude and the flow of active and reactive power across its terminals. The amounts of active and reactive power are recorded by the node sensors and they are available to all nodes, whether the sensor is a conventional type or a virtual measurement for nodes with no load or power injection.

The Midwest 240-Node test distribution system comprises 51 voltage magnitude sensors and 453 active and reactive power sensors. The uncertainty or variance of the sensors is assumed to be a fixed percentage error based on the nominal rating of the sensor, and is set equal to three times the standard deviation. With this empirical rule, 99.7% of the sampled measurements are within the nominal error rating of the sensor. The data presented in Table 2 is on a per-unit basis, with a power base of 63,484,243.29 MW. The error of the voltage sensors is 1% of the nominal value. The selected nodal load is set equal to 11 kW with an expected 5% error on the nominal 11 kW rating. The selection of 11 kW nodal load is to consider the overall level of the nodes the absolute maximum nodal load is 28.15 kW and the averaged maximum load node is 9.6 kW. The error for all nodal injection/load active and reactive power nodes is the same. The only exception is for the nodes connected to capacitors where the variance is calculated to be 17 kvar with 5% error for the nodal injection/load active and reactive power. The maximum observed flow of power at three circuit breaker nodes is 280 kW, with an expected 5% error on the nominal 280 kW rating. The errors for all nodal circuit breaker flows for active and reactive power nodes are assumed to be equal.

Table 2. The sensors' nominal values and their respective uncertainty for Midwest 240-Node test distribution system

| Sensor Type | Nominal Value (pu) | Percentage Error (%) | Variance |
|---|---|---|---|
| Voltage magnitude | 1.0 | 1 | $1.11 \times 10^{-5}$ |
| Active and reactive power flow for not capacitor or circuit breaker nodes | 0.00017 | 5 | $8.33 \times 10^{-12}$ |
| Active and reactive power flow for capacitor nodes | 0.00441 | 5 | $1.99 \times 10^{-11}$ |
| Active and reactive power flow for circuit breaker | 0.00026 | 5 | $5.4 \times 10^{-9}$ |

# 5.0 Results

## 5.1 Gross Error Detection in the State Estimation Process

The method for detecting gross error using the chi-square test detailed in Section 3.3 is presented in this section. The analyses consider the gross error to be dependent on the type of sensor and its service location. The expected sensor error is equal to three times its error standard deviation. Table 3 presents the error level needed to trigger the chi-square-test detection of gross error in relation to the expected sensor error. The values presented are the average error by sensor, broken out by area. The gross error is limited to a single sensor of the selection set at a time. The normal error of the sensors is equal to a normal random sample with variance equal to the specific sensor's uncertainty. The simulation time used is "2017-07-17 16:24:00."

Table 3. Error, as a multiple of the expected sensor error, required to trigger the chi-square gross error detection test for the various sensors for all the areas of the system described in Section 4.0

| | | Required error as the number of times greater than the expected sensor error by areas | | | | | |
|---|---|---|---|---|---|---|---|
| Selection | Type | Area 1 | Area 2 | Area 3 | Area 4 | Area 5 | Area 6 |
| All available nodal sensors | Voltage magnitude | 2.10 | 2.20 | 2.10 | 2.20 | 2.10 | 2.20 |
| Circuit breaker flow nodes | Active power | 2.50 | 2.50 | 3.58 | 2.50 | 3.08 | 2.50 |
| | Reactive power | 2.50 | 2.50 | 3.58 | 2.50 | 3.00 | 2.50 |
| Non-circuit-breaker nodes | Active power | 55.65 | 56.44 | 85.00 | 56.43 | 76.50 | 56.89 |
| | Reactive power | 55.40 | 56.50 | 84.00 | 54.78 | 76.50 | 55.91 |

The gross error detection test is significantly more sensitive for sensors attached to the circuit breaker nodes. The circuit breaker nodes are weighted more heavily for the SE, as demonstrated by Table 3. Their relative importance is allocated because their readings can be validated against those of other sensors: the breaker nodes have voltage magnitude sensors and active and reactive flow sensors. The errors on the active and reactive power sensors from non-circuit-breaker nodes (i.e., area load sensors) affect only the circuit breaker sensors; the demand at one node does not affect the demand at another. This makes the gross error detection with the evaluated sensor set significantly more sensitive to the circuit breaker nodes. For example, the gross error detection could be made more sensitive to non-circuit-breaker nodes by including voltage magnitude sensors among the non-circuit-breaker nodes, which would make a node's demand affect a larger set of sensors.

## 5.2 Detection of the State of the Circuit Breaker

Accurate knowledge of the circuit breaker states is essential. The state of each circuit breaker is needed in multiple activities in the power system, especially when evaluating the current system state, which is necessary for assessing possible reconnections. Power system topology identification is explored in multiple papers (Gandluru et al. 2020). As mentioned in Section 3.3.2, identification of the system circuit breaker states uses the expected sum of the errors evaluated during the SE. The circuit breaker state is tested by evaluating the sum of expected errors assuming a different power flow on the circuit breakers. Open circuit breakers should have near-zero flow and closed breakers are expected to have nonzero flow.

The sum of the expected sensor errors of the known state of the circuit breaker is compared with the sum that results with a different amount of flow. A known “closed-switch” state is compared against an “open-switch” state (i.e., power flow is made equal to zero) and a known “open-switch” state is compared against a “closed-switch” state. This last case requires the injection of current to simulate power delivery across its terminals. If the assumed flow is too small, the topology detection will fail. The chosen assumed flow is set at 1/25 (0.04×) the expected sensor flow error.

The circuit breaker test has been performed for all the areas of the test system for the simulation time from “2017-07-17 16:23:00” to “2017-07-17 18:00:00,” with one-minute resolution. The test successfully identified the correct state of the circuit breaker for all areas across all the simulated reference times.

## 5.3 Sequential Calculation of Voltage Droop for the Assessment of Reconnection

The sequential calculation of the voltages on the nodes of the system for reconnection (presented in Section 3.3.3) will be demonstrated to be able to compute the voltage in all the nodes. For the purpose of defining whether an area can be reconnected, the nodal loads are assumed to be equal to the maximum recorded load; however, for the validation of the approach the load is made equal to the temporal system load being experienced. The same period was used, from “2017-07-17 16:23:00” to “2017-07-17 18:00:00,” at one-minute resolution. The voltage magnitude is computed for all system nodes under normal operation conditions and with the circuit breaker states shown in Figure 13. Calculating the expected flow of power to the adjacent areas uses the power flow on the circuit breaker. The absolute voltage magnitude difference between the power flow simulation in the distribution system simulator OpenDSS and that from the proposed sequential approach leveraging the SE is presented in Figure 14. The largest observed absolute difference is 0.000097 PU, which implies a smaller difference for all areas of the system of 0.001% of the voltage magnitude. The average difference for all areas is less than 0.0004%. The maximum observed difference is negligible, demonstrating the accuracy and validity of the proposed approach. The observed differences are expected, given the numerical process used by both the power flow simulation in OpenDSS and the SE. The proposed approach can evaluate the system voltage in multiple steps without requiring any of the area SEs being performed to know the complete model of the system. The approach only needs to know its own model, internal loads, the upstream bus voltage magnitude, and downstream areas’ load demands.

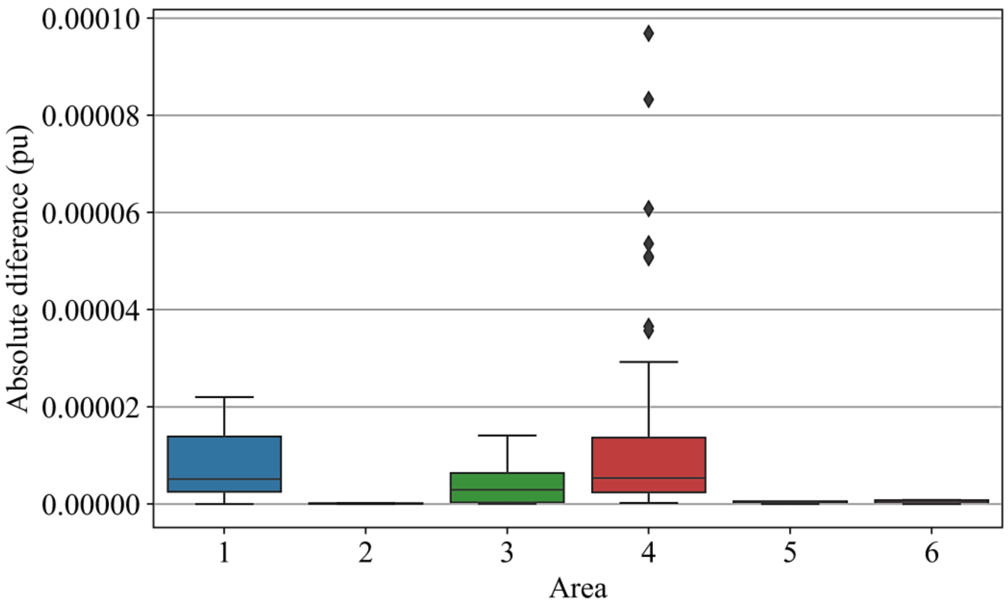


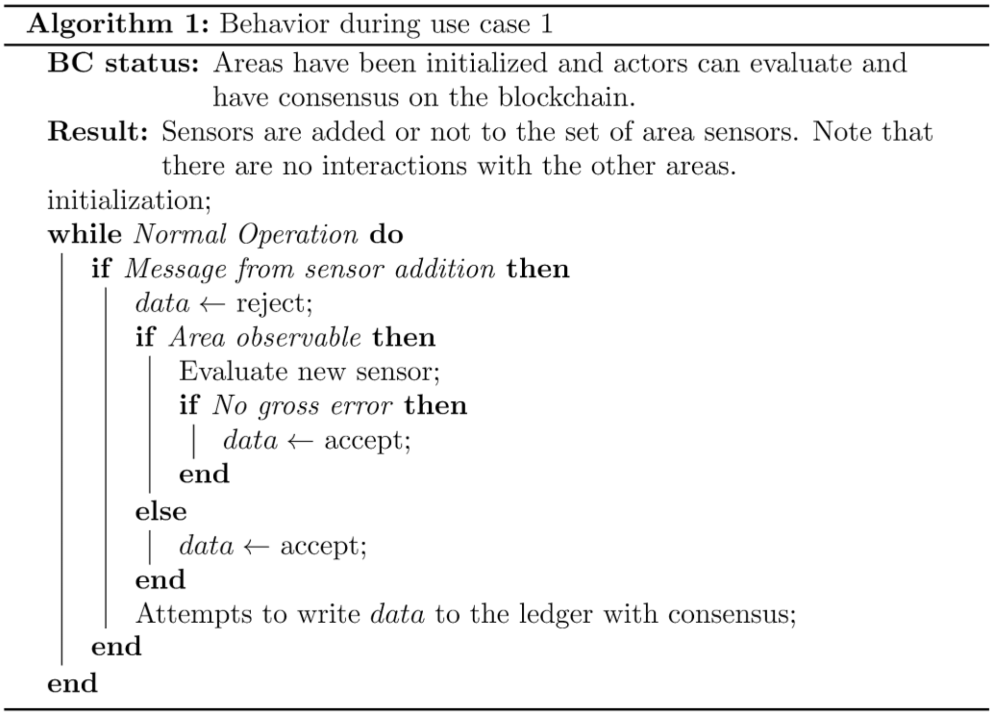
Figure 14. Absolute voltage magnitude difference in each area in Figure 13 between the power flow performed in OpenDSS and the proposed sequential calculation that leverages the SE

## 5.4 Use Case Behavior

Behavior during the use cases has not yet been evaluated with simulations, nor have the actors and message exchange with delays and loss of packages been implemented. However, the expected behavior is explored in this section by looking more closely into the independent behavior in the use cases.

### 5.4.1 Behavior during Use Case 1 – Blockchain for Grid Data Configuration

The blockchain for data configuration is explored by approving or rejecting the participation of an area sensor attempting to join the set of area sensors. The approach is presented in Section 3.0; this section disregards the behaviors of the other processes to focus on Use Case 1 independently. Figure 15 presents the behavior during an attempt to include a new sensor. Notice that “Evaluate new sensor” will perform a SE analysis evaluating whether there is gross error on the new sensor being evaluated. To avoid the possibility of multiple sensors triggering the gross error detection, only the area sensors that behaved appropriately on the previous evaluation (i.e., not triggered the gross error detection) and the sensor attempting to join are in the set of sensors for the SE in “Evaluate new sensor.” The attempt is considered successful if the decision to accept/reject the sensor addition can be added to the ledger. For information to be included in the ledger, the consensus mechanism must be satisfied. The challenges of message package delay and loss can impede the consensus. However, during normal operation, this is only a delay, given that the sensor will attempt to join again once a set time has passed. Since the previous attempt at analysis was unsuccessful, evaluation of the sensor continues without penalty. A sensor that is evaluated and rejected multiple times will no longer be eligible to request a new evaluation; this requires manually resetting the sensor and assigning new credentials. By performing this action, the system can protect itself against the effects of a configuration error or a possible malicious attack. It will be the responsibility of the cybersecurity response team to make a determination after the fact, to assess whether the registration failure was part of a deliberate cyberattack or a human error.



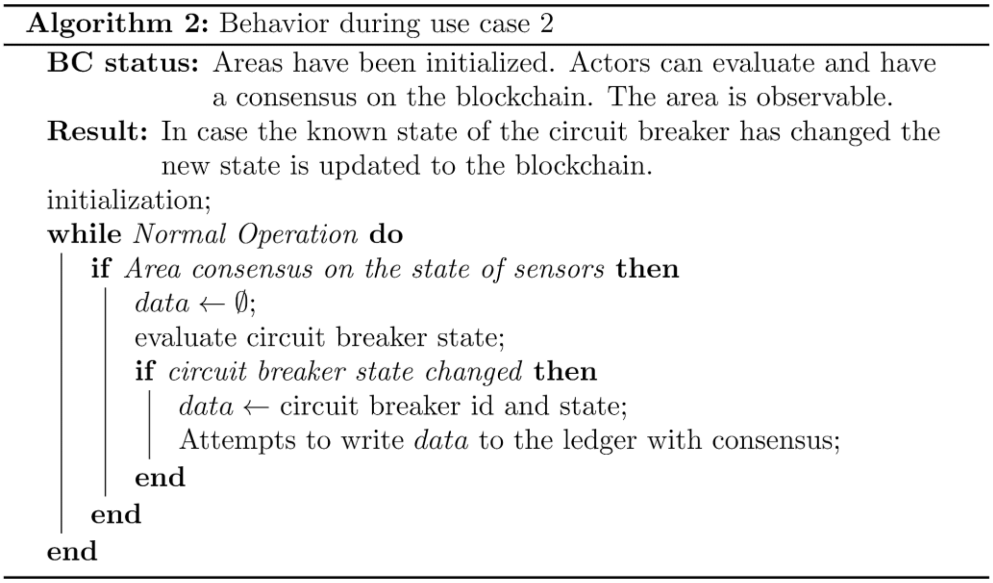
```
Algorithm 1: Behavior during use case 1
  BC status: Areas have been initialized and actors can evaluate and
             have consensus on the blockchain.
  Result: Sensors are added or not to the set of area sensors. Note that
          there are no interactions with the other areas.
  initialization;
  while Normal Operation do
      if Message from sensor addition then
          data ← reject;
          if Area observable then
              Evaluate new sensor;
              if No gross error then
                  data ← accept;
              end
          else
              data ← accept;
          end
          Attempts to write data to the ledger with consensus;
      end
  end
```

Figure 15. Pseudocode of the blockchain for data configuration for sensor additions in Use Case 1

Notice that the addition of distributed generation units could be handled by a similar approach (e.g., by having the sensors attest the distributed generation unit’s capabilities). However, additional validations (e.g., qualification processes) may be in place to make sure the claimed technical capabilities are indeed possible. This may include evaluating the device’s ability to operate in an island mode or to provide energy for critical loads.

### 5.4.2 Behavior during Use Case 2 – Blockchain for Topology Identification

The blockchain for topology identification is explored by assessing the system’s ability to find topology discrepancies. Under normal circumstances, the state of the circuit breaker is updated in the ledger as soon as a switch changes state. The approach is presented in Section 3.0; however, this section disregards the behaviors of the other processes to focus on Use Case 2 independently. Figure 16 presents the behavior of the system shown in Figure 13 during the system state evaluation. The set of area sensors that have been included in the area set are evaluated periodically to determine and monitor the gross error values. This is a repetitive process that is performed for all the sensors within the area; as such, it is only possible once the area becomes observable. The criteria for observability was presented in detail in Section 3.3.1.2. Once the observability criteria are met, Use Case 2 is possible. After a round of gross error detection is completed, a topology check is executed. The topology check is performed by auditing the state of the circuit breakers within a given area. The verification uses only the area sensors that have been deemed healthy by the gross error detection algorithm. Evaluation of the states of the circuit breakers follows the logic presented in Section 5.2. As long as the behavior is within the parameters, the approach can identify the states of the circuit breakers. The combination of the states of the circuit breakers in all the areas of the system yields the topology of the system.



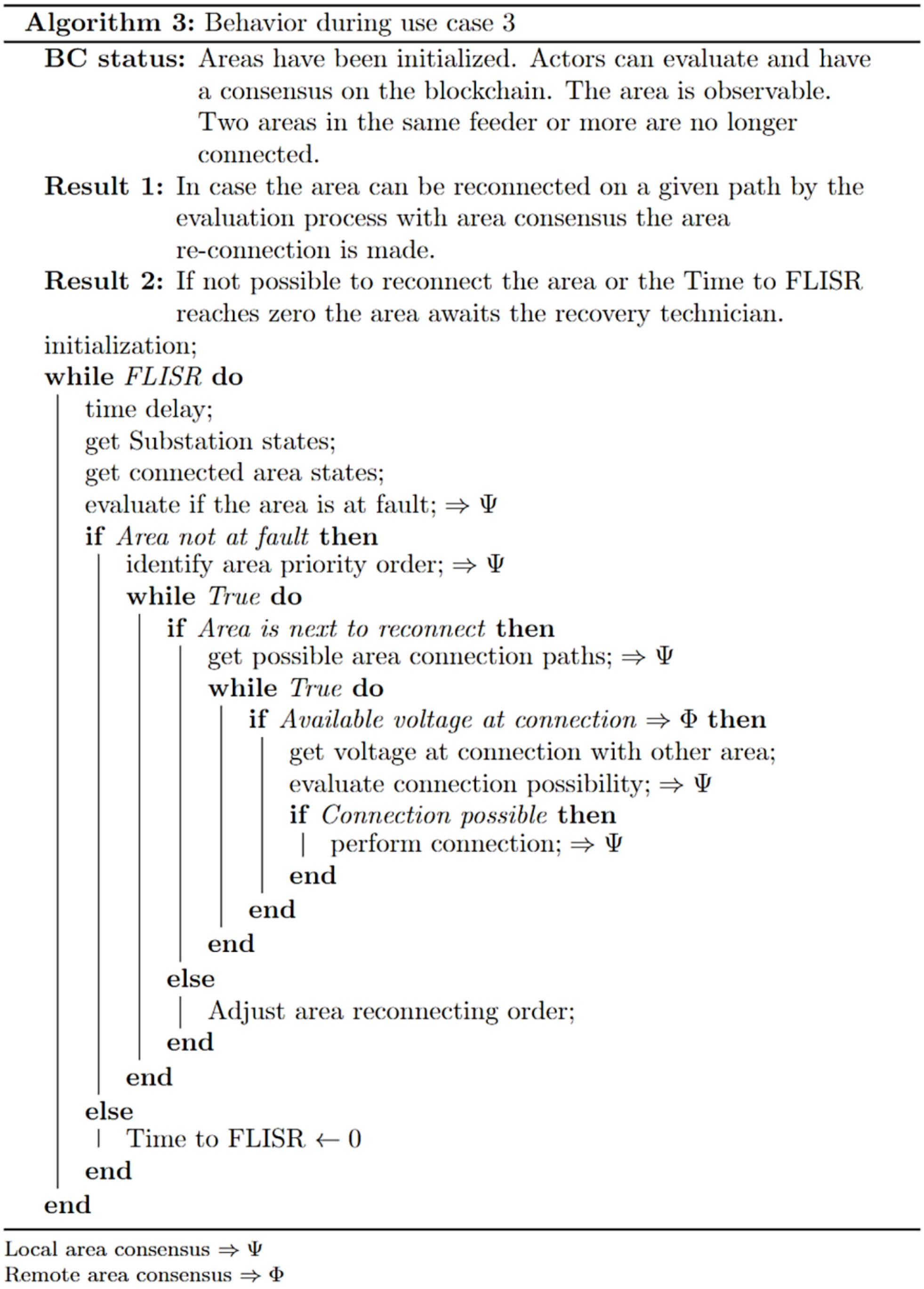
```
Algorithm 2: Behavior during use case 2
  BC status: Areas have been initialized. Actors can evaluate and have
             a consensus on the blockchain. The area is observable.
  Result: In case the known state of the circuit breaker has changed the
          new state is updated to the blockchain.
  initialization;
  while Normal Operation do
      if Area consensus on the state of sensors then
          $data \leftarrow \emptyset$;
          evaluate circuit breaker state;
          if circuit breaker state changed then
              $data \leftarrow$ circuit breaker id and state;
              Attempts to write data to the ledger with consensus;
          end
      end
  end
```

Figure 16. Pseudocode of the blockchain for topology identification for Use Case 2

Because real networks experience message delay and occasional package losses, occasional lack of consensus is expected. However, during normal operation, this will only result in a delayed topology assessment, given that the circuit breaker check will run inside a “while” loop (and thus will be updated in a subsequent iteration). The delay could reduce visibility on a system scale if the area topology is out of date.

### 5.4.3 Behavior during Use Case 3 – Blockchain for Distributed Fault Location, Isolation, and Service Restoration

The blockchain for DFLISR permits reconfiguration of the system to maintain service to customers during a fault event, as long as the faulted area can be isolated from the rest of the system. As presented in the Section 3.3.3, there are multiple steps, and they are distributed among the system areas. The approach is presented in Section 3.0; however, this section disregards the behaviors of the other processes to focus on Use Case 3 independently. Figure 17 presents the behavior during the process of DFLISR. Once the area is identified as disconnected, a time delay will be introduced to enable other nearby areas to update their states. The area managers continue monitoring their areas of interest for disconnected areas. If they receive such information, the cyclical evaluation of the connection is accelerated, in an attempt to reduce the time required for updates during an event. This can also occur when protection has been operated, which means the areas of the system should evaluate their states of connection. Knowing which system areas are not connected and assuming that the protection system is operating accordingly, the area that contains the fault can be identified as the one farthest upstream that is disconnected. The area with the fault cannot be reconnected, given that it requires maintenance (i.e., reclosers have failed to reconnect it, so utility personnel must assess the damage or whether the fault is still on the system). However, the downstream areas can attempt to be reconnected because they are less likely to contain a fault. To evaluate whether the connection is possible, the areas assess the possible voltage drop at the area to be connected. Assessing the voltage requires information on the demand of the areas. To be conservative, the area's historical maximum is used as the demand. Once the areas can evaluate the voltage at the area to be connected as presented in Section 3.3.3 and validated in Section 5.3, the knowledge of the area to be connected is known. If the connection is possible, the circuit breakers are changed and the area is reconnected to the system.



```
Algorithm 3: Behavior during use case 3
  BC status: Areas have been initialized. Actors can evaluate and have
             a consensus on the blockchain. The area is observable.
             Two areas in the same feeder or more are no longer
             connected.
  Result 1: In case the area can be reconnected on a given path by the
            evaluation process with area consensus the area
            re-connection is made.
  Result 2: If not possible to reconnect the area or the Time to FLISR
            reaches zero the area awaits the recovery technician.
  initialization;
  while FLISR do
      time delay;
      get Substation states;
      get connected area states;
      evaluate if the area is at fault; $\Rightarrow \Psi$
      if Area not at fault then
          identify area priority order; $\Rightarrow \Psi$
          while True do
              if Area is next to reconnect then
                  get possible area connection paths; $\Rightarrow \Psi$
                  while True do
                      if Available voltage at connection $\Rightarrow \Phi$ then
                          get voltage at connection with other area;
                          evaluate connection possibility; $\Rightarrow \Psi$
                          if Connection possible then
                              perform connection; $\Rightarrow \Psi$
                          end
                      end
                  end
              else
                  Adjust area reconnecting order;
              end
          end
      else
          Time to FLISR $\leftarrow 0$
      end
  end
Local area consensus $\Rightarrow \Psi$
Remote area consensus $\Rightarrow \Phi$
```

Figure 17. Pseudocode of the blockchain for fault location, isolation, and service restoration

In Use Cases 1 and 2, the effects of message package delay and loss are mostly minor, having the system some level of delay to be up to date with the desired behavior. Unfortunately, that is not the case for Use Case 3. Figure 17 illustrates the increasing extent of consensus required for reconnecting areas. Since the goal is to use the ledger to present information to other areas, the other areas must have consensus on their assessment. Communication challenges among the area attempting to rejoin and the areas in its possible reconnection paths could cause significant delays using the proposed DFLISR approach. The delay will increase the outage time of the disconnected customers and may completely block the reconnection, given that the DFLISR operational time is limited to protect utility personnel who may be performing maintenance.

# 6.0 Conclusion and Future Work

A blockchain-based technology core can provide three features to an application: (1) an immutable, decentralized ledger; (2) a distributed, consensus-based agreement process; and (3) a distributed state-replication engine. These core blockchain characteristics can be leveraged by grid applications to attain fault-tolerant characteristics, and as such have been leveraged to support three use cases here. An important feature of the work is its reliance on the consensus-driven ledger that enables actors (e.g., distributed resources) to have access to a trusted data store, and also enables actors to agree on a joint decision by publishing the result back to the ledger. The first use case (blockchain for data configuration) and the second use case (blockchain for topology identification) work together to provide a trusted, decentralized platform that enhances visibility into the distribution system. Use Case 1 is responsible for evaluating individual sensors and identifying sensors that can be trusted for higher decision-making processes (these are evaluated using mathematically based criteria, i.e., the chi-square test for gross error detection). Using information from sensors that have been deemed trustworthy, Use Case 2 identifies the states of the circuit breakers (i.e., the topology of the distribution network). Again, by leveraging the trust provided by the first two use cases, Use Case 3 can safely execute switching actions that can reconfigure feeders to reconnect customers that have been disconnected after a fault event. The importance of trust cannot be understated, and is a significant contribution to traditional FLISR approaches; furthermore, thanks to its blockchain-based architecture, the FLISR application becomes decentralized, enabling independent areas to make decisions even when communication with a central control center is lost. The main contribution of this report is therefore a detailed design of the three use cases, which can be implemented by third parties. This fosters technology adoption and hopefully, reduces the time to market of similar solutions. From an engineering perspective, the test results confirm the algorithm’s ability to compute accurate results. For example, the gross error detection process can detect malfunctioning circuit breaker sensors when the error exceeds 3.8 times the expected sensor error in the chi-squared gross error detection test. This enables creation of a trusted set of sensors to be used for higher decision-making. The sensitivity to gross error detection depends on the system, the set of sensors, and the detection method. With the deployment of more sensors in the distribution system and other/new detection methods, the detection sensitivity can improve. Detection of the circuit breaker state can identify its proper state during all the simulation scenarios and provide an accurate distribution system network topology for higher decision-making. The procedure for sequential voltage calculation (i.e., to preserve interarea privacy) conforms with the results obtained using OpenDSS. The OpenDSS distribution-system power-flow solution generally has a convergency tolerance of 0.01% on the voltage magnitude. The evaluation of possible reconnection using voltage magnitude and preserving the data ownership has a voltage magnitude difference smaller than 0.001% from the OpenDSS result; this is below the expected power flow tolerance with full knowledge of the system, which surpasses expectations.

## 7.0 References

Angulo-Paniagua, J., and J. Quirós-Tortós. 2020. “Comparing Chi-square-Based Bad Data Detection Algorithms for Distribution System State Estimation.” 2020 IEEE PES Transmission & Distribution Conference and Exhibition - Latin America (T&D LA), 28 Sept.-2 Oct. https://doi.org/10.1109/TDLA47668.2020.9326241.

Bhattarai, B., D. J. Sebastian Cardenas, F. Bereta dos Reis, M. Mukherjee, and S. Gourisetti. 2021. *Blockchain for Fault-Tolerant Grid Operations.* PNNL-32289. Richland, WA: Pacific Northwest National Laboratory. https://www.pnnl.gov/publications/blockchain-fault-tolerant-grid-operations.

Bompard, E., T. Huang, Y. Wu, and M. Cremenescu. 2013. “Classification and trend analysis of threats origins to the security of power systems.” *International Journal of Electrical Power & Energy Systems* 50: 50-64. https://doi.org/10.1016/j.ijepes.2013.02.008.

Bretas, A. S., N. G. Bretas, B. Carvalho, E. Baeyens, and P. P. Khargonekar. 2017. “Smart grids cyber-physical security as a malicious data attack: An innovation approach.” *Electric Power Systems Research* 149: 210-219. https://doi.org/10.1016/j.epsr.2017.04.018.

Brinkmann, B., and M. Negnevitsky. 2016. “A practical approach to observability analysis and state estimation in distribution networks.” 2016 IEEE Power and Energy Society General Meeting (PESGM), Boston, MA, 17-21 July 2016. https://doi.org/10.1109/PESGM.2016.7741271.

Department of Energy. 2014. *Fault Location, Isolation, and Service Restoration Technologies Reduce Outage Impact and Duration.* Washington, DC: U.S. Department of Energy Office of Electricity Delivery and Energy Reliability. https://www.smartgrid.gov/document/fault_location_isolation_and_service_restoration_technologies_reduce_outage_impact_and.

Dos Reis, F. B., R. Tonkoski, B. Bhattarai, and T. M. Hansen. 2021a. A Real-World Test Distribution System with Appliance-Level Load Data for Demand Response and Transactive Energy Studies. https://doi.org/10.17632/d57wxwgt4w.1.

Dos Reis, F. B., R. Tonkoski, B. P. Bhattarai, and T. M. Hansen. 2021b. “A Real-World Test Distribution System With Appliance-Level Load Data for Demand Response and Transactive Energy Studies.” *IEEE Access* 9: 149506-149519. https://doi.org/10.1109/ACCESS.2021.3120923.

Gandluru, A., S. Poudel, and A. Dubey. 2020. “Joint Estimation of Operational Topology and Outages for Unbalanced Power Distribution Systems.” *IEEE Transactions on Power Systems* 35 (1): 605-617. https://doi.org/10.1109/tpwrs.2019.2935401.

Gonen, T. 2015. *Electric Power Distribution Engineering*. Boca Raton, FL: CRC Press. https://books.google.com/books/about/Electric_Power_Distribution_Engineering.html?id=JIDSBQAAQBAJ.

Haji, M. M., and O. Ardakanian. 2019. “Practical Considerations in the Design of Distribution State Estimation Techniques.” 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), Beijing, China, 21-23 Oct. 2019. https://doi.org/10.1109/SmartGridComm.2019.8909805.

Hewitt, C., P. B. Bishop, and R. Steiger. 1973. “A Universal Modular ACTOR Formalism for Artificial Intelligence.” 3rd International Joint Conference on Artificial Intelligence, Stanford, CA, 20-23 August. https://www.ijcai.org/Proceedings/73/Papers/027B.pdf.

Jodal, S. M. Pykka. https://pykka.readthedocs.io/en/stable/.

Kantor, J., and C. Sandrock. TCLab: Temperature Control Laboratory. https://tclab.readthedocs.io/en/latest/.

Liu, S., Y. Hou, C. C. Liu, and R. Podmore. 2014. “The Healing Touch: Tools and Challenges for Smart Grid Restoration.” *IEEE Power and Energy Magazine* 12 (1): 54-63. https://doi.org/10.1109/MPE.2013.2285609.

Pacific Northwest National Laboratory. GridLAB-D 5.0, Richland, WA. https://www.gridlabd.org/.

Peyton Jones, S. 2007. “Beautiful Concurrency.” In *Beautiful Code: Leading Programmers Explain How They Think*, edited by G. Wilson and A. Oram. O'Reilly Media, Inc. https://www.oreilly.com/library/view/beautiful-code/9780596510046/.

Wang, Z. Iowa Distribution Test Systems. Iowa State University. http://wzy.ece.iastate.edu/Testsystem.html.

**Pacific Northwest National Laboratory**

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354
1-888-375-PNNL (7665)

***www.pnnl.gov***