Addressing Situational Awareness in Critical Domains of a Smart Grid

Cristina Alcaraz, and Javier Lopez

Computer Science Department, University of Malaga,

Campus de Teatinos s/n, 29071, Malaga, Spain

{alcaraz,jlm}@lcc.uma.es

October 27, 2015

Abstract

Control and situational awareness are two very important aspects within critical control systems, since potential faults or anomalous behaviors could lead to serious consequences by hiding the real status of supervised critical infrastructures. Examples of these infrastructures are energy generation, transmission or distribution systems that belong to Smart Grid systems. Given the importance of these systems for social welfare and its economy, a situational awareness-based model, composed of a set of current technologies, is proposed in this paper. The model focuses on addressing and offering a set of minimum services for protection, such as prevention, detection, response, self-evaluation and maintenance, thereby providing a desirable protection in unplanned situations.

Keyword: Critical Infrastructure Protection, Smart Grid, Supervisory Control and Data Acquisition Systems, Situational Awareness, and Wireless Sensor Networks

1 Introduction

A Smart Grid is a complex infrastructure composed of a set of domains and stakeholders. According to the conceptual model of the National Institute of Standards and Technology (NIST), these domains correspond to customers, markets, providers, energy generation, distribution and transmission networks (e.g., power substations), as well as control systems such as SCADA (Supervisory Control and Data Acquisition) systems [1]. This last domain can be considered as the main core of the entire system that widely interconnects with the other domains/sub-domains. This interconnection enables the SCADA Center to know the performance of the entire Grid and control its functions for delivering essential services, such as electrical energy.

Unfortunately, control substations in charge of supervising in real-time the performance and functionality of energy bulk generation systems (either renewable or non-renewable), or electrical transmission or distribution lines have a tendency to suffer numerous and unforeseen events caused by failures or errors. The origin of these suspicious events may even provoke disturbances or instabilities within a particular substation that could trigger a devastating cascading effect, with a high probability of reaching other domains within the Grid. This is due to the existing interdependency relationships [2,3] that may intensify the spread of the effect, thereby (partially or totally) disrupting functionalities/services of other domains/sub-domains.

We agree with NIST that it is necessary to provide preventive and proactive solutions to face emergency situations [1]. In fact, NIST classifies this need as one of the eight priority areas to be considered for the protection of Critical Infrastructures (CIs), and it is known as Wide-Area Situational Awareness (WASA). Given its importance within a Smart Grid, in this paper we propose a model based on the use of different technologies to ensure control at all times, in addition to offering a support for situational awareness. The proposed approach is specifically composed of:

- The technology of Wireless Sensor Networks (WSNs) for monitoring the actual state of the infrastructure observed and its industrial resources (e.g., turbines);
- The ISA100.11a standard [4] for managing different kinds of SCADA incidents, represented through alarms and classified into five levels of priority;
- Two preventive methods. One of them focusing on anticipating critical situations and the other on controlling anomalies or malfunctions in the control tasks.
- Cloud computing based on Sensitive Data (SD) for data redundancy (i.e., alarms and readings) and safety-critical; i.e., take control of a highly critical situation to avoid the propagation of a cascading effect [5]; and
- A self-validation mechanism to evaluate the real state of the entire system, itself.

Self-validation basically consists of evaluating the level of accuracy of the methods applied for protection of CIs. These methods correspond to the prevention (anomalous situations related to the infrastructure controlled) and/or detection (anomalies or threats within the control network). This detection is mainly based on the use of simple behavior patterns that help to detect unsuitable (hardware and software) functions in sensor nodes, thereby offering a support for maintenance and auditing tasks. Note that some of these solutions try to address new research areas, such as cloud computing for critical contexts, and others try to fill some research gaps such as prevention. Indeed, although there are some action plans and initiatives [6] to provide preventive solutions, there is not so far enough research on this topic for critical contexts; and more particularly in the provision of specialized predictive solutions based on simple forecast models.

The paper is organized as follows. Section 2 introduces the basic components for the construction of the approach, which will be later used for the design in Section 3. In particular, the approach and its components, technologies and methods for prevention, detection, response and self-validation are discussed in detail in Section 3.1 and Section 3.2. Finally, Section 4 concludes the paper and outlines future work.

2 Four Basic Components for the Construction of the Approach

A system based on situational awareness basically comprises advanced monitoring components with integrated techniques that help to analyze and interpret data streams, normally from embedded devices (e.g., sensor nodes), which are distributed close to the controlled infrastructure (e.g., machineries). Likewise, these techniques have the capability for decision-making and alerting. Therefore, four main components should form the foundations of our approach; (i) a *detection component*, (ii) an *information recollection component* to store evidence, (iii) an *alarm management component* to issue alerts and warn the system, and (iv) a *reaction component*. The detection component is based on WSNs since their devices (sensor nodes) are able to monitor physical events (such as high/low levels of voltage); detect and track anomalous behaviors; warn of anomalous situations; and actively interact with the gateway [7]. The gateway is a powerful device that serves as an interface between the acquisition world (i.e., the WSN) and the real world (i.e., the SCADA Center). In addition, these sensor nodes are smart devices with the capability of collaborating with each other and guaranteeing self-configuration to adapt themselves to the conditions of the network, as well as self-healing to address unforeseen situations.

The information recollection component in our model is represented by the SCADA Center itself, the SD cloud and any external storage device in charge of registering and storing SCADA evidence flows. The use of cloud computing for evidence storage enables the system to maintain a register of events occurred in the past. If the control is (temporarily or permanently) lost (e.g., the SCADA Center is out of service), another SCADA system may retake control through the ICCP (Inter-Control Center Communications Protocol) industrial protocol, and know the state of the system by querying the DS cloud [5]. The effectiveness of using this technology and its application for managing incidents in critical contexts are thoroughly analyzed in [5]. In fact, one of its great advantages is the availability of resources, keeping control at all times and recovering sensitive data irrespective of the situation; and in this way ensuring a continued supervision and safety-critical in crisis scenarios. A safety-critical is considered an essential property [3] that should be considered when the underlying infrastructure is critical, as the existence of unplanned events may potentially lead to serious consequences [2, 3]; e.g., overload in generators, high voltage peak in transformers, etc.

The alarm management component is based on specific management systems offered by existing wireless industrial communication standards, such as ISA100.11a. This standard provides a set of services for communication reliability, security (based on symmetric and asymmetric cryptography), coexistence, and priority-based alarm management using up to five criticality levels: *journal, low, medium, high* and *urgent*. Its networks can support sensor nodes working at 26MHz, 96KB RAM, 128KB flash memory and 80KB ROM, and one or several gateways to establish redundant connections with the SCADA Center. The information from sensors is managed through DMAP (Device Management Application Process) objects. DMAP is a class installed inside each device, which includes a set of objects used for configuring, supervising and requesting parameters belonging to sensor nodes. More specifically, DMAP contemplates the ARMO (Alert Reporting Management Object) class for managing alerts and generating reports through an AlertReport service to ARO (Alert Receiving Object). ARO is a class configured in only one device in the network (the gateway in our case). Finally, the reaction component focuses on carrying out decision-making processes that depend on a set of factors, amongst others, the simplicity of the technique applied (which should not increase functional complexities that can compromise the control of the underlying infrastructure and its services) and the autonomous and dynamic capacity of the approach to address threatening situations. In our case, this component is principally based on a set of integrated modules that collaborate with each other to carry out several tasks. Some of them are; to estimate the proximity of a possible anomaly; locate and warn the nearest operator in the area; evaluate the level of accuracy in the detection and prevention tasks; and frequently report the real state of the network.

3 A Dynamic and Automatic Situational Awareness

As ISA100.11a allows configuring diverse types of networks, the architecture of the approach (See Fig. 1) is based on a hierarchical configuration; where nodes are grouped into clusters and all the organizational decisions are carried out by a trustworthy entity known as the *Cluster Head* (CH). Each CH$_i$ is responsible for receiving and checking information (either readings or ISA100.11a alarms) from their sensors in order to detect and warn of anomalous behaviors through patterns, in addition to filtering and aggregating information (main tasks of a CH) to be resent to the gateway later. The selection of this configuration is for two main reasons. First of all, this configuration not only allows the system to efficiently manage its resources in computation and energy terms, but it also helps to locate anomalies by knowing the network deployment in advance. Second, part of the processing is straightforward, since the approach has been designed for very specific situations using simple behavior patterns.

An anomalous behavior can be defined as "*something deviated from what is standard, normal, or expected*". From this definition, taken from the Oxford Dictionary [8], we deduce that if a reading is not inside a prescribed threshold, $[V_{min}, V_{max}]$, then it can be considered anomalous. As our approach measures readings of voltage, denoted as v_i, a deviation from the allowable thresholds is therefore considered as an anomaly. When this situation appears, the system has to deliver an alarm. Taking advantage of ISA100.11a and its alarm management, we can consider three principal situations:

- Valid readings, $v_i \in [V_{min}, V_{max}]$, where V_{min} and V_{max} refer to the acceptable thresholds of readings. To highlight and signal this case, we use the value 0.
- Non-critical alarms, $v_i \notin [V_{min}, V_{max}]$, but they do not compromise the security of the system. These alarms are journal, low and medium, and are signaled with values 1, 2 and 3 respectively.
- Critical alarms, $v_i \notin [V_{min}, V_{max}]$, but they can compromise the security of the system. These correspond to alarms with high and urgent priority, which are signaled with values 4 and 5 respectively.

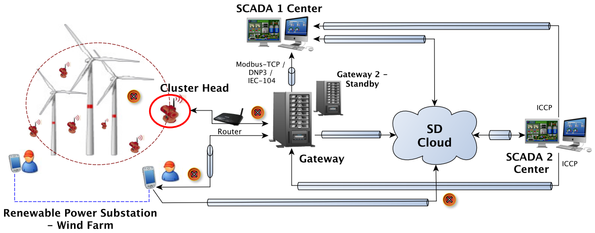

Figure 1: General Architecture of the Model

The gateway is in charge of resending any type of information (valid readings, non-critical alarms and critical alarms) from the WNS to the SCADA Center; interpreting and translating (e.g., Modbus-TCP/IP - ISA100.11a) messages using GSAP (Gateway Service Access Point) points; and storing information copies in the SD cloud for backup. It is also responsible for anticipating future anomalies, managing critical alerts [4-5], and validating the entire approach itself. For dealing with critical alerts, the gateway also has to locate the most suitable operator equipped with a hand-held device within the area, which makes use of different communication systems (e.g., Mobile Ad-Hoc Networks). On the other hand, although security aspects are beyond the scope of this paper, we assume that communication channels 'sensor-sensor' are protected by using security credentials and cryptographic services provided by the ISA100.11a standard [4]; and the rest of the communications will depend on the security services of the TCP/IP standard and on the use of virtual private networks.

3.1 Sensors and The Cluster Head for Dissemination and Detection

Figure 2 depicts the chief modules of the CHs: *Message Normalization*, *Pattern Association*, *Alarm Manager* (AM-CH), *Data Aggregation*, and *Diagnosis Manager*. Each sensor node, s_i, with identification IDs_i sends its messages (either a v_i or an alarm) to its CH$_j$ with IDch_j, which first operates the Message Normalization module. The main task of this module is to combine and represent different data inputs in a generic format. The normalized message is then sent to the Pattern Association module in order to verify the nature of such inputs using simple behavior patterns. For example, verify if readings or critical alarms received from a s_i are outside their acceptable thresholds before being forwarded to the gateway. In this way, we can make good use of the cluster head by supervising the functional instabilities of the nodes included within it. These instabilities may be, for example, caused by software/hardware errors or malfunctions due to a lack of maintenance. Depending on the detected anomaly, the AM-CH module will generate, through the ARMO class, a new alarm signaled with high priority (4) so that a human operator can be made aware of the situation and can review the scenario.

For simplicity, we consider the following network model. The network deployment is based on trustworthy nodes where sensors are distributed close to their cluster heads, and each cluster is based on a small configuration of nodes. Each node has to transmit a message with the value of the reading and priority assigned, the identifier IDs_i and

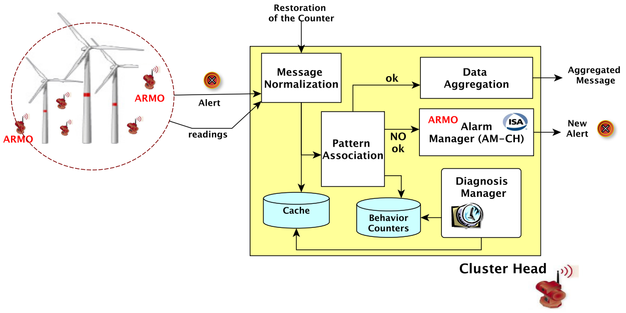

Figure 2: Architecture of the Cluster Head

the time-stamp. To address the software malfunction problems, each CH must verify the payload of each message to check whether its value of reading corresponds to the priority assigned by the sensor; e.g., verify whether $v_i \in$ (or $\notin$) $[V_{Low_{min}}, V_{Low_{max}}]$. These thresholds of criticality must be defined according to security policies established by the SCADA organization, electrical companies and countries. Only in the case where a CH analyzes a discrepancy in the control made by a sensor, the CH then has to penalize its attitude by updating its behavior counter, $counter_{SensorBh}$ by one unit. This counter is unique for each node and when its value is greater than a prescribed behavior threshold (i.e., $counter_{SensorBh} > T_{SensorBh}$), the CH will also have to warn the AM-CH.

On the other hand, hardware problems are managed using the Diagnosis Manager, which periodically queries the last sequence of events received from the sensors using a cache memory. This memory, which is maintained by the Message Normalization, allows the Diagnosis Manager to know when a particular node of the cluster is not sending messages for a short time period. If this occurs, the CH infers that something anomalous is happening with the sensor, and updates its $counter_{SensorBh}$. This problem could be attributed to a significant reduction in battery levels or the lifetime of the sensor is over. It should be noted that the counter used coincides with the behavior counter described above, because when a node is behaving incorrectly, the system increases (without any distinction of the cause) its value until it reaches its threshold, $T_{SensorBh}$. In that moment, the CH will have to warn of the situation so that the sensor can be tested.

For generating a new alarm, both the Pattern Association and the Diagnosis manager will have to send the AM-CH a set of data. For example, the IDch_j; IDs_i; the type of alarm (only if the received message from s_i is an alarm); the priority assigned by the sensor; the priority assigned by the CH; and the type of event detected. The kind of event is an indicator that will help to make the gateway and the human operator aware of the type of problem to check. It should be noted that this type of validation is only effective for critical alarms [4-5], since valid readings and non-critical alarms will be used as input for prevention. In particular, two types of events are used: *event_detectionSensor* and *event_detectionCH*. The former refers to the detection made by a sensor node (i.e., the control of the CI and its services), whereas the latter is at-

Resources	**1 CH - 0 sensors**	**1 CH - 1 sensor**	**1 CH - 2 sensors**	**1 CH - 3 sensors**
CPU	7,36MHz	7,37MHz	7,37MHz	7,36MHz
Memory (r-w)	2,75% - 3,02%	2,74% - 3,01%	2,73% - 2,99%	2,72% - 2,98%
Energy (CPU-Radio)	3,31 J - 8,62 J	3,31 J - 8,61 J	3,31 J - 8,62 J	3,32 J - 8,63 J

Table 1: Resources of one Cluster Head When Sensors Are Being Integrated within the Cluster

tributed to the detection carried out by the CH (i.e., the control of behaviors within the cluster). To show the simplicity of the Pattern Association module, the Pseudo-Code 1 summarizes the order of execution of its actions.

We have validated this part of the approach using the Avrora simulator under the de-facto standard operating system for sensor nodes, TinyOS 2.x [9]. Avrora is able to interpret conventional sensor nodes (e.g., Mica2), which belong to the category II defined in [7]; i.e., 4-8 MHz, 4-10 KB RAM, 48-128 KB ROM with 2-8 mA of energy. The results of the simulation (See Table 1) indicates that a cluster working as a Mica2, requires less than 8MHz to execute the software, consuming around 3,3 Joule for CPU and 8.6 Joule for radio, and approximately reaching a maximum of 2.8% for reading (r) and a 3% for writing (w) in memory. Therefore, if traditional sensors are able to work as CHs, then ISA100.11a sensors belonging to the category III with higher capabilities (13-180 MHz, 256-512 KB RAM, 4-32 MB ROM and 40 mA of energy) are also able to server as CHs.

```
//Obtain normalized message and extract values to analyze
message = NormalizedMessage();
reading = Extract_ReadingData(message);
prioritySensor = Extract_Priority(message);
$IDs_i$ = Extract_IdentifierSensor(message);
//Verify the accuracy of the sensor to assign priority
IF (VerifyData(reading,prioritySensor)) THEN
        IF (Priority(prioritySensor, 0)) THEN
                //Aggregate whether the contain is a reading
                DataAggregation($IDs_i$, reading);
        ELSE
                //Resend the alarm to the Gateway
                ForwardAlarm_AM − CH($IDch_j$, $IDs_i$, reading, prioritySensor, ”event_detectionSensor”);
        END
ELSE
        //Determine the real criticality of the received reading according to behavior patterns; and
        //Update the $counter_{SensorBh}$ of the sensor node $s_i$
        priorityCH = DeterminePriority(reading);
        $counter_{SensorBh_i}$ = UpdateBehaviorCounter($IDs_i$);
        IF ($counter_{SensorBh_i} \leq T_{SensorBh}$) THEN
                //Generate a new alarm to evaluate behaviour in the gateway
                GenerateNewAlarm_AM − CH($IDch_j$, $IDs_i$, high, prioritySensor, priorityCH, ”event_detectionCH”);
        ELSE
                //Generate a new alarm to warn the operator of the replace/discard of the sensor
                GenerateNewAlarm_AM − CH($IDch_j$, $IDs_i$, high, ”event_discardNode”);
        END
END
```

Pseudo-Code 1: Control of Software/Hardware Malfunctions within a Cluster

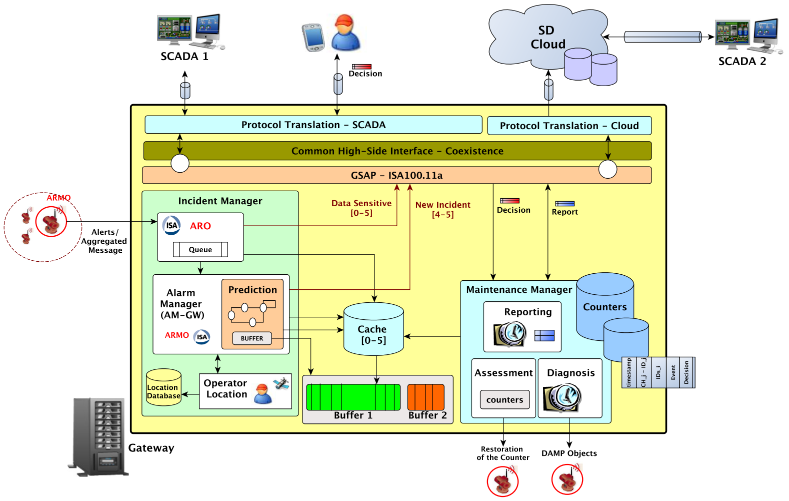

Figure 3: Architecture of the ISA100.11a Gateway

3.2 A Powerful Gateway for Control, Prevention, Response and Maintenance

As part of the approach, a gateway is integrated inside the model (See Fig. 3), which is composed of two chief managers: An *Incident Manager* and a *Maintenance Manager*.

3.2.1 Incident Manager: Prevention, Data Redundancy and Response.

Any type of information received from CHs is taken in through the *ARO* sub-module, which temporarily stores them within a cache memory and send a copy to both the SCADA Center and the SD cloud. For incident management, ARO uses one organized queue, which is sorted by priorities. Depending on the criticality of the message, the *Alarm Manager* (AM-GW) sub-module will carry out two actions; one predictive and other reactive. For the predictive part, the AM-GW must compute the rate of valid readings (0) and non-critical alarms [1-3] received from the network. The idea is to calculate, for each sensor, rates of consecutive values of non-critical alarms with value 3 over the last time period, as it may mean the proximity of a possible incident. Although, there are currently several forecast models that could be used to anticipate such situations [10], we propose below a simple prevention method, which is included inside the *Prediction* sub-module belonging to the AM-GW.

The method basically consists of calculating probabilities of transition between states: st_0 (represents valid readings), st_1, st_2, st_3 (represents different types of criticality [1-3]). These states and their values have to be previously exported from the cache memory to a separate temporal buffer, which is assigned to each network sensor, Bff_i, with a size Δ_{Bff_i}. However, this buffer is not only based on information exported from the cache, but also on past information (a small percentage) in order to keep a sequence

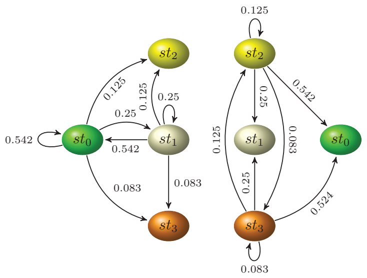

Figure 4: Transitions Between States: From st_α to st_β

of events with respect to the time line. Therefore, the size of Bff_i is based on exported information with a size of Δ_{Bff1_i} and on past information with a size of Δ_{Bff2_i}; i.e., $\Delta_{Bff} = \Delta_{Bff1} + \Delta_{Bff2}$, where $\Delta_{Bff1} \geq \Delta_{Bff2}$. In this way, we can restrict the size of Δ_{Bff} and reduce computational costs by avoiding to computing several times the predictive algorithm and the cache memory.

For each of the states, we also design a particular probability of transition pr_{st_α,st_β}, which corresponds to the probability of going from a state α to a state β; i.e., $pr_{st_\alpha,st_\beta} = Pr(st_{i+1} = \beta | st_i = \alpha)$, where $\sum_{i=0}^{3} pr_{st_\alpha,st_\beta} = 1$. Taking this into account, we assume that the probability of remaining in the st_0 is much greater than transiting to the st_3 or remaining within this; i.e., $pr_{st_0} > pr_{st_1} > pr_{st_2} > pr_{st_3}$. In order to calculate probabilities, we consider the following Equation: $1/(4 \times \alpha)$, where $\alpha >= 1$ and $pr_{st_0} = 1 - (\sum_{\alpha=1}^{3} pr_{st_\alpha})$. Note that we have taken this simple equation as an initial approach. Other approaches could also be equally valid if they are achieved with the restriction of $pr_{st_0} > pr_{st_1} > pr_{st_2} > pr_{st_3}$. The result of computing the probabilities for each state is as follows: pr_{st_0} - 0.542; pr_{st_1} - 0.25; pr_{st_2} - 0.125; and pr_{st_3} - 0.083. Figure 4 graphically depicts the relationships between states together with the cost of their transitions.

Considering the previous assumptions and notions, the occurrence of an event can be computed as follows.

$$\frac{InitialState + \sum_{j=0}^{\Delta_{Bff_i}-1} pr_{Bff_i[j],Bff_i[j+1]}}{\Delta_{Bff_i}} \leq (pr_{st_3} + \sigma_{error}) \qquad (1)$$

where *InitialState* corresponds to $pr_{Bff_i[0]}$ and σ_{error} represents an acceptable margin of error. This means that if the result of computing Equation 1 is lower than $pr_{st_3} + \sigma_{error}$, the system can determine that the next value to be received will be either a non-critical alarm with value 3 or a critical alarm (a stressed situation). To the contrary, when the system determines that the result of computing Equation 1 is higher than $pr_{st_3} + \sigma_{error}$, it may infer that the next entry may be either a valid reading or a non-critical alarm (a normal/acceptable situation). To make this clearer, two examples are shown below, which are based on a $\Delta_{Bff_i} = 10$ ($\Delta_{Bff1_i} = 5$ and $\Delta_{Bff2_i} = 5$) with a $\sigma_{error} = 0$.

- Let the sequence of events stored in a Bff_i as 0 3 2 3 3 3 0 3 3 3, the system then computes the transitions and their probabilities using Equation 1. Resulting in, $0.179 > pr_{st_3}$. Then the system estimates that the next event to be received may be either a valid reading or a non-critical alarm.

- If the sequence of events has 3 3 3 3 3 3 3 3 3 3, the result of calculating would be $0.083 \leq pr_{st_3}$. Therefore, the system determines that the next event to be received may be either a non-critical alarm with value 3 or a critical alarm due to the high rate of alarms received with value 3.

To address this last case, and of course the reactive part, the system has to warn of the proximity of this situation by sending a new alarm with high priority through the AM-GW. Such an alarm must be sent to both the SCADA Centre and the nearest operator within the affected area so as to immediately attend to the situation. Similarly this can also occur when ARO directly receives critical alarms [4-5] from the sensor network (e.g., alarms with the type of event "*event_discardNode*"). For operator location, the AM-GW uses the *Operator Location* sub-module, which considers the operator's availability (according to his/her contract), his/her responsibility/role to carry out a task, and his/her location within the area. To carry out such a search, the Operator Location makes use of both a local database, called *Location Database*, and a location external device, such as a geospatial information device, so as to geographically identify the physical position of the nearest human operator within the affected area. Lastly, and as mentioned in Section 2, the AM-GW not only has to send a copy of new incident generated to the SCADA Center but also to the SD cloud for future governance aspects and recovery purposes.

3.2.2 Maintenance Manager: Self-validation and Maintenance.

In order to know the real state of the entire approach, the *Assessment* sub-module needs to receive certain feedback on how accurate the prevention and detection modules have been. This feedback is dependent on the operator's final decision, who is obliged to verify, validate and notify (through their hand-held interfaces) the system of the reliability of the detection/prevention made by the control network. In fact, four possible situations could occur: (i) The node determines that an anomaly is occurring within the system, and it coincides with the operator's decision (a True Positive (TP)); (ii) the node determines that an anomaly is occurring within the system, and it does not coincide with the operator's decision (a False Positive (FP)); (iii) the node determines that no anomaly is occurring within the system, and it does not coincide with the operator's decision (a False Negative (FN)); and (iv) the node determines that no anomaly is occurring within the system, and it coincides with the operator's decision (a True Negative (TN)). It should be noted that a TN does not make sense within our approach, thus it has not been considered.

Depending on the operator's decision, the Assessment sub-module will have to update the level of accuracy for a node using three kinds of counters (associated with

	Prevention	**Detection and Control of the CI**	**Detection and Control of the Cluster**					
	priority	***prioritySensor***	***priorityCH***			***prioritySensor***		
pirotyOp.	**High**	**High**	**0**	**[1-3]**	**[4-5]**	**0**	**[1-3]**	**[4-5]**
Normal Sit. - 0	FP	FP	TP	FP	FP	TP*	FP	FP
Unstable Sit. - [1-3]	FP	FP	FN	TP	FP	FN	TP*	FP
Critical Sit. - [4-5]	TP	TP	FN	FN	TP	FN	FN	TP*

Table 2: Table for Evaluating the Prevention and Detection Modules, and Updating Counters

each network node); $count_{tp}$ for TPs, $count_{fp}$ for FPs, and $count_{fn}$ for FNs. If said counters reach their respective prescribed thresholds, then the Assessment sub-module will have to issue a new alarm with a high priority through the AM-GW. The new alarm should contain, at the very least, information related to the nodes involved (e.g., IDs_i, IDch_j, IDgw) and the action to be carried out, such as *event_review_detectionModule*, *event_review_predictionModule*, or even *event_discardNode* (discard/replace devices).

For evaluating the prevention, it is enough to take into account the operator's decision and the estimation of the Prevention sub-module. The operators' decision is going to depend on three types of criticality levels: *normal situation* (0), *unstable situation* [1-3], and *critical situation* [4-5]. For example, if the operator's feedback corresponds to a *normal situation*/*unstable situation* (See Table 2), the $count_{fp}$ of the Prevention sub-module should be increased accordingly. This validation method is equivalent to evaluate the reliability of sensors in their control tasks of CIs (with event *event_detectionSensor*, See Section 3.1); and the reliability of CHs in their supervision tasks of malfunctions (with event *event_detectionCH*, See Section 3.1). Nonetheless, it is worth mentioning that this last kind of validation is a little more complex, as the sub-module requires contrasting the version of the CH$_j$ (i.e., *priorityCH*, See Pseudo-Code 1 of Section 3.1) and the version of the sensor involved, IDs_i, (i.e., *prioritySensor*, See Pseudo-Code 1 of Section 3.1) with respect to the criticality provided by the human operator (i.e., *priorityOp*). When contrasting versions, a further two specific situations may take place:

- The *priorityOp* coincides with the *priorityCH*; i.e., TP in CH: The system rewards the CH by increasing its $count_{tp}$, and penalizes the s_i according to the real criticality of the system. Hence, if *priorityOp* $>$ *prioritySensor*, then $count_{fn}$ of the s_i is increased; otherwise, its $count_{fp}$ is updated by one unit.
- The *priorityOp* does not coincide with the *priorityCH*; i.e., FP/FN in CH: The system increases the $count_{fp}$/$count_{fn}$ of the CH, accordingly. However, a further two cases may also occur when the the counters of the sensor have to be updated:
  - The *priorityOp* is equal to the *prioritySensor*; i.e., TP in s_i: The system rewards the s_i by updating its $count_{tp}$, and proceeds to restore the value of the $counter_{SensorBh_i}$ (See Section 3.1). To this end, the Assessment sub-module has to send a notification to its corresponding CH$_j$ to increase its value by one unit. Note that this action, also depicted in Table 2 using the indicator '*', significantly reduces the communication overhead. If this counter was managed by the gateway, this could mean a high communication cost,

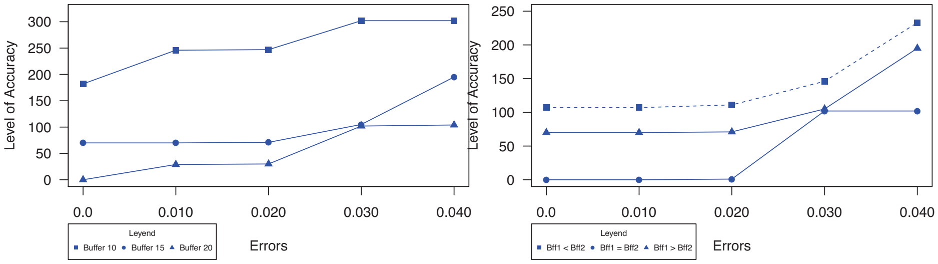

Figure 5: Left Hand Side Figure: The Importance of σ_{error} and Δ_{Bff} for Critical Contexts; Right Hand Side Figure: The Importance of Δ_{Bff1} and Δ_{Bff2}

as the $counter_{SensorBh}$ needs to be continuously updated by the Association Pattern module and Diagnosis Manager of the CH.

- The *priorityOp* is not equal to the *prioritySensor*; i.e., FP/FN in s_i: If the *priorityOp* is less than the *prioritySensor*, the system increases the $count_{fp}$ of the s_i; otherwise the system penalizes the node by increasing its $count_{fn}$.

When a $count_{fp}$ and/or a $count_{fn}$ reach their acceptable thresholds (T_{fp} and T_{fn}, respectively), the SCADA Center should be warned in order to take new protection and security measures, and thereby guarantee continuity of services. Note that the T_{fn} should be much more restrictive than the T_{fp}, such that $T_{fn} \leq T_{fp}$. We cannot accept that anomalies within a CI and its industrial resources are not detected properly, since they could lead errors or faults into cascading [3]. One way to know the situation and reliability of the entire system, would be to (periodically or on-demand) generate a report with accumulative values of the counters ($count_{tp}$, $count_{fp}$, $count_{fn}$) through the *Reporter* sub-module. Finally, and for extending the functionality of the approach, a *Diagnosis Manager* is also used to check the lifetime of the CHs. As the Diagnosis Manager of Section 3.1, it will have to frequently check whether a specific CH$_j$ stopped sending messages during a significant time period by analyzing its sent frequency in the cache memory. If this occurs, the manager will have to diagnose its existence by sending a message based on DMAP objects. If the CH$_j$ does not respond within a maximum time limit, the manager will have to warn of the situation using the type of event *event_CH_discardNode*. These diagnoses allow the system to manage isolated areas caused by malfunctions or denial of service attacks in CHs. Obviously, this action should be carried out for each network node, but this could mean a degradation of performance. For this reason, we supervise the lifetime of sensors using the counter $counter_{SensorBh}$, and thus we avoid a increasing in the communication overhead.

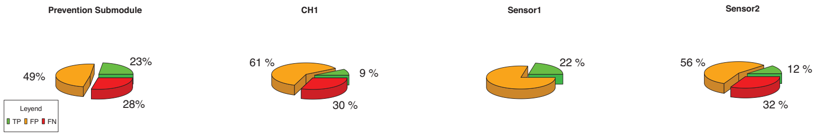

Figure 6: A Report Obtained from a Simulation of an Critical Scenario (Intentionally Unstable)

3.3 Other Major Points of Discussion

It is quite important to define a suitable value for σ_{error} and an appropriate buffer size for Δ_{Bff} (See Section 3.2.1) for prevention. The higher the margin of error and the smaller the buffer are, the greater the probability of obtaining a high false positive rate. Figure 5 (left hand side Figure) shows this aspect and its importance for critical contexts. The values are obtained from a simulation executed under the Java platform, where a critical scenario has been implemented which is composed of three clusters with two or three sensors each, and the control of the network is managed by three (virtual) available operators. Sequences of events (intentionally stressed) have been analyzed according to different sizes Δ_{Bff1} (5, 10, 15), Δ_{Bff2} with value 5, and different values of σ_{error} (0.0, 0.010, 0.020, 0.030 and 0.040). Given this, Δ_{Bff} then takes the following values 10, 15, 20 ($\Delta_{Bff} = \Delta_{Bff1} + \Delta_{Bff2}$). For the generation of such event sequences, we have assumed the following criteria. Each sensor node periodically produces events with values that can range between 0 and 5. Each production maintains a special correlation with events transmitted in the recent past, such as the frequency of a particular type of event and its priority. If a type of event with a specific priority is significantly repeated in a short time period, a new type of event with a higher priority is generated.

As shown in Figure 5 (left hand side Figure), a system configured with a Δ_{Bff} size of 10 is less restrictive and precise than using a buffer with a size of 20. This is also the case when the system is configured with a σ_{error} with value of 0.040. On the other hand, Figure 5 (right hand side Figure) represents the importance of determining the sizes of Δ_{Bff1} and Δ_{Bff2}. The results indicate that a $\Delta_{Bff1} \geq \Delta_{Bff2}$ (continued line - $\Delta_{Bff1} = 10$ and $\Delta_{Bff2} = 5$; and $\Delta_{Bff1} = 10$ and $\Delta_{Bff2} = 10$) is more precise than using a $\Delta_{Bff1} < \Delta_{Bff2}$ (dashed line - $\Delta_{Bff1} = 5$ and $\Delta_{Bff2} = 10$). The reason is that the system is able to contrast more present information with a small portion of past information so as to follow the behavior of the sensors in the time.

Although, all these configurations normally depend on the requirements of the SCADA organization and its security policies, they can change throughout of the life-cycle of the system. This change may occur when the Reporter Manager reports the current situation of the context. An example of a report could be the representation of percentages obtained from the values associated with the counters of TPs, FPs and FNs. Figure 6 shows said representation, which is also based on the results obtained from the simulation. In the extreme case that the counters of FPs and FNs are greater than their prescribed threshold (e.g., $count_{fp} > T_{fp}$), the SCADA Center could reconfigure the parameters to restrict the values associated to σ_{error} and Δ_{Bff}. On the other hand, it is essential to have good software maintenance of sensors, as their outputs are the input of the Prevention. This can be seen as a dependency relationship of 'cause-effect'. If a sensor does not work properly, the prediction can then tend to false positives or false negatives. Therefore, the role of the CH to detect malfunctions in sensors and the role of the Maintenance Manager to control anomalous behaviors in the entire system are fundamental to avoid disturbances in the final prediction.

4 Conclusions

A dynamic situational awareness model for control systems has been proposed here. The approach is based on the composition of different technologies and construction blocks in order to provide a set of benefits for situational awareness, such as *dissemination, prevention, detection, response, control, maintenance* and *safety-critical*. In particular, we have seen that we can obtain information from the infrastructure and its surroundings by using a WSN, and know their real states by managing different kinds of incidents. Through a hierarchical configuration, the system can detect particular malfunctions using simple behavior patterns, in addition to preventing and warning of the proximity of unstable situations, and responding to them in a timely manner. In addition, data redundancy enables the system to be aware of incidents that have occurred in the past, and recover the control when essential parts of the system remain isolated or out of service. Finally, it is worth highlighting that the design proposed in this paper can be extrapolated to other critical contexts such as transport systems.

Unfortunately, it is still necessary to continue further with the topic of situational awareness for protection of CIs to endow the system with autonomous and dynamic capacities. It would be interesting to explore new technologies and techniques and adapt them to the critical context without compromising its security and performance. Our next goal will be to extend the approach to consider all of these aspects, in addition to those topics related to security. In particular, this research will focus on open privacy issues to protect sensitive information within the cloud [5], and on designing simple behavior patterns to detect threats/attacks within a sensor network [11]. Note that parts of these topics are still very dependent on advances in hardware/software resources of sensor nodes. Therefore, investigation in this area is also needed.

Acknowledgments

This work has been partially supported by PISCIS (P10-TIC-06334), ARES (CSD2007-00004) and SPRINT (TIN2009-09237) projects. This last one is co-funded by the FEDER Program.

References

[1] NIST Special Publication 1108R2, *NIST Framework and Roadmap for Smart Grid Interoperability Standards*, Release 2.0, Office of the National Coordinator for

Smart Grid Interoperability, February 2012.

[2] B. Falahati, and Y. Fu, *A Study on Interdependencies of Cyber-Power Networks in Smart Grid Applications*, 2012 IEEE PES Conference on Innovative Smart Grid Technologies, Washington DC (USA), January 2012.

[3] C. Alcaraz, and J. Lopez, *Analysis of Requirements for Critical Control Systems*, Sixth IFIP WG 11.10 International Conference on Critical Infrastructure Protection, National Defense University, Washington DC (USA), March 2012.

[4] ISA100.11a, *ISA-100.11a-2009. Wireless systems for Industrial Automation: Process Control and Related Applications*, The International Society of Automation, 2009-2012.

[5] C. Alcaraz, I. Agudo, D. Nunez, and J. Lopez, *Managing Incidents in Smart Grids à la Cloud*, In IEEE CloudCom 2011, IEEE Computer Society, pp. 527-531, 2011.

[6] ENISA, *Securing European Information Society*, Work Programme 2011.

[7] J. Lopez, R. Roman, and C. Alcaraz, *Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Networks*, In Foundations of Security Analysis and Design 2009, LNCS 5705, pp. 289-338, August, 2009.

[8] Oxford Dictionary, *Anomalous Situation*, `http://oxforddictionaries.com/definition/anomalous`, Retrieved on June 2012.

[9] TinyOS Working Group, `http://www.tinyos.net/`, Retrieved on June 2012.

[10] F. Salfner, *Event-based Failure Prediction An Extended Hidden Markov Model Approach*, PhD Thesis, Humboldt-Universittzu Berlin, 2008.

[11] C. Alcaraz, and J. Lopez, *A Security Analysis for Wireless Sensor Mesh Networks in Highly Critical Systems*, IEEE Transactions on In Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 40, no. 4, pp. 419-428, 2010.