ASHBEC	Preventing Alert Fatigue Author: Vaughn Frick	Analysis Report Date: Mar 22, 2020 Version: 1.6
Preventing Alert Fatigue in a Patient Treatment Management System		

Management Summary

This paper clears up some misconceptions about alerts in a patient treatment management system. In particular it debunks the notion that you can build a patient treatment management system that does not use alerts. It then describes a method and system (patent pending) that will allow the effective use of alerts in a patient treatment management system. The described approach will eliminate alert fatigue and facilitate dramatic quality increases in treatment delivery.

CONTENTS

1.0 What is an alert?

1.1 The common understanding

Particularly when discussing alert fatigue, alerts in a patient treatment management system are generally thought of as often unwelcome pop-up boxes on a healthcare worker's screen that disrupts their workflow and brings to their attention events or information that they don't necessarily consider important. In some cases they become so onerous that non-technical users of the system request the designers to build the system without any alerts. This is an understandable view of alerts. Having such a narrow definition prevents the insight necessary to solve the problem of alert fatigue. To effectively manage alerts in a patient treatment management system requires a broader, information technology based definition.

1.2 Basic definition

We need to start with some basic definitions and a brief discussion of systems theory. Systems are designed to react to events in the outside world. In fact, system behavior can be defined by describing how the system will react to events. Of course there will be many events about which the system does not care or react. The ones the system cares about are the ones that are said to "cross the system boundary". That simply means that the system has been defined to detect certain events and provide a response to them. For example, a sick or injured patient walking through the emergency room door is an event to which a hospital system will react.

Any system has components or "parts". (These components can include both people and machines.) Many of these system components are designed/trained specifically to react to particular events. When an event crosses the system boundary, the system component that detects the event must decide what system component will react to or "process" the event. Think of it as triage. If one patient walking through the door is complaining of a headache and fever and another is bleeding profusely, the system will react differently based on the type of event. If it detects event A it will display one behavior while event B will elicit another response. To implement this, the detecting component must send a message to the system component that will process that particular type of event. That message is called an alert. The recipient of the alert will be an information processor. The processor may be computer software or a person. In either case, the message is still an alert. This event/alert/response is effectively the definition of the system. ***It is not possible to have a system without alerts.***

In almost all cases, there will be a time delay between when an event is detected and when the appropriate processor can react to the event. The length of the time delay is a system design parameter chosen based on the priority of the event being handled. When the detecting component sends the alert, it will typically be held in a message queue until the processing component has time to process it. Think of it as a waiting room for alerts. Every processor will have a queue in front of it. The system designer is responsible for balancing the queue size and processing capacity so the events are processed in a timely fashion. For example, the system designer must ensure that the queue for a profusely bleeding patient is much shorter than the queue for the headache/fever.

Not all events that are detected will be external to the system. During the processing of an event, other events may be detected that change the processing of the original event. For example, an abnormal test result may cause a change to the treatment plan. These events would also result in alerts that should be directed to a processor for that particular event.

1.3 Alert overload

Alert overload is exactly what it sounds like. Events are arriving faster than the processor can process them. (The message queues holding the alerts are growing rapidly and approaching the queue capacity.) If the arriving events are crossing the system boundary (i.e., they are not internal events), there are only two possible responses. The system processing power must be increased or a way must be found to reduce the number of incoming events. For example, consider a disaster situation. Massive casualties are pouring into a hospital that quickly approaches capacity. The first reaction is to call in all available staff to increase processing capabilities. Since the hospital has no ability to stop the disaster, a standalone hospital system could do nothing else. The good news is that in most instances, the hospital system is actually a subsystem of a larger healthcare system. At that point, the hospital notifies first responders that they have reached capacity and that all additional casualties ("events") should be transported to another hospital. Obviously this is not a desirable steady-state. At this point, one can only hope that the casualties that can't be treated at the first hospital will arrive in another facility in time to be treated.

One of the keys in this example is that the hospital system recognized the overload event and reacted to it. Without this key design feature, casualties would've piled up in the parking lot. The same design feature should be in place anytime there is a reasonable prospect of alert overload.

1.4 Alert fatigue

"Alert fatigue" isn't really a systems term. The phrase is an emotional reaction to alert overload when the message queue that is overflowing is being processed by a human processor. In the vast majority of cases, this occurs not because of understaffing (insufficient processing power) but rather because the internal alert has been misdirected. Alerts should only be sent to "processors" (human or machine) that should provide the system's reaction to the event.

Consider the following example. Assume the hospital has a standard practice of not leaving an IV in the same location for more than seven days. Further assume that the system is capable of detecting an event when the IV has been in the same location for more than seven days. At most, it should issue an alert to the healthcare worker responsible for moving the IV as a reminder. No one else needs to receive the alert. Sending it to someone else that would not act on it themselves simply clogs their message queue obscuring higher priority alerts. If the IV still has not been moved after eight days (a different event), a second alert might be generated to a supervisor that would be expected to take action to protect the patient. The key design feature is to send alerts only to the processor that should process the event. Don't send alerts for informational purposes or simply to suggest what may or may not be a better way for a healthcare worker to perform a task.

If alert fatigue is occurring, it is clearly a sign of system failure. Either alerts are being sent to the wrong location or the system was designed with insufficient processing power/staffing. Like the disaster example, this case of alert overload should also be detected and managed. A method for doing this is described below. Like the disaster example, it does not create a desirable steady-state condition but it will buy time to correct the underlying problem.

1.5 Systems that don't use alerts

All systems use alerts by definition. If a software vendor tells you that their product does not use alerts, they are either giving you a sales fabrication or they don't understand systems theory. If they go so far as to create a work plan for healthcare worker, they are using alerts because the work plan in and of itself is an alert. In any case, proceed with caution. What they're really saying is there are one or more classes of events to which their system has been designed not to react. The key to understanding what they are selling is often to understand what events that you as the system owner consider important that their software will ignore. Quite often, the events that they have chosen to ignore are exactly those required for continuous process improvement of the treatment delivery process.

For any quality improvement program to be successful, the process to be improved must be monitored. This means that alerts must be issued when the process is not being followed. In a patient treatment management system, this implies that alerts must be issued when the treatment plan is not followed. In other words, it is possible for every step in a treatment plan to trigger an alert. At a minimum, an alert would be issued if the step is not completed in a timely fashion. This type of alert would probably be issued only to the healthcare worker responsible for completing the task. The form might be as simple as changing the color of the task in the work plan from blue to yellow. If the treatment regimen used as a template for the treatment plans is not designed carefully, alerts can be issued for the wrong reasons and/or to the wrong people (e.g., suggesting an alternative treatment that may have already been considered) or in enormous numbers. Poor design can also result in alerts being sent to healthcare workers that are not responsible for fixing the underlying problem. For example, assume that an alert is triggered because a significant amount of time has passed after a task in the treatment plan was supposed to be completed but has not been. In such a situation, the first level alert should go only to the healthcare worker responsible for completing the task. Sending an alert to that worker's supervisor would be counterproductive since it is highly likely that the healthcare worker will correct the problem themselves without supervisory intervention. (In some situations it may be appropriate to send a second level alert to the supervisor if there is no response to the initial alert.) Without proper design, it is quite possible to send too many alerts to the wrong people. When this happens, these "wrong" people can become inundated with so many alerts that they are forced to ignore them all. This phenomenon is generally referred to as the aforementioned "alert fatigue".

2.0 Combining people and machines to manage alerts

The method and system described here (patent pending) can be applied to managing alerts for any patient treatment management system that issues alerts for continuous process improvement.

2.1 Creating or modifying an alert

The heart of any patient treatment management system is the treatment regimen. It is a standardized, predictable and repeatable process used as a template to create a patient specific treatment plan. Each step in the treatment regimen represents a work task performed by a healthcare worker. Associated with each step will be a set of conditions that would cause the system to trigger an alert. The alert might be issued if the step is not completed in a timely fashion or if the step resulted in an adverse outcome (e.g., test results or vital signs out of range).

The healthcare worker responsible for the design of the alerts associated with these steps must take into account what healthcare workers (or types of healthcare worker) should respond to the alert. The alert should be defined in such a way that only those workers receive the alert. The designer should also have the ability to suspend the issuing of some alerts. The information associated with an alert that has been suspended is not lost. It will be captured by the automated system for subsequent analysis. Not all alerts are appropriate candidates for suspension. For example, while it may be appropriate to initially suspend alerts related to tasks not completed on time until the cause is determined and a solution is put in place, alerts related to "vital signs out of range" should never be suspended. In the preferred implementation, the designer will have the ability to flag an alert as one that should never be suspended as well as flagging alerts as suspended in the current treatment regimen.

The definition of the alert should also include the maximum number of outstanding alerts issued to a particular healthcare worker before the next alert issued to that worker results in triggering an information overload alert. When that maximum is exceeded, and alert overload alert will be issued only to the analyst in charge of CPI.

In the preferred implementation, the designer should also be able to specify a frequency which, if exceeded, would cause the system to automatically suspend an alert which has been defined as one which can be suspended. The actual frequency may be dependent on the size of the organization in which the treatment regimen is being used and should be controlled by the treatment regimen designer.

The designer must also anticipate how groups of alerts can be categorized for batch analysis. The purpose of the analysis will be to identify a group of alerts that are being issued too frequently, perform a root cause analysis to determine why they are being issued so frequently and lastly, what steps need to be taken to improve the quality of the treatment delivery and reduce the frequency of the alerts. To facilitate this, the system should allow the designer to identify the type of alert being defined. While there will be many common types, the designer must have complete flexibility to define whatever type they feel will be useful in the analysis.

2.2 Alert triggers

2.2.1 Manual Alerts

In most cases the triggering event and how it will be detected will be defined for a particular alert definition. However, healthcare workers reviewing a treatment plan may notice a problem in the plan that will require one or more other people on the healthcare team to adjust the plan and their activities. In a preferred implementation, the alert management system should provide healthcare workers with the ability to manually trigger an alert in such a circumstance. In such an instance, the "healthcare worker assigned to the task" would be the healthcare worker that triggered the alert.

2.2.2 Alert overload triggers

Alert overload triggers will normally be the result of the number of outstanding alerts for particular healthcare worker exceeding the defined limits. In a preferred implementation, alert overloads could also be triggered based on the frequency of alerts for a particular department or the hospital as a whole.

2.3 Processing an alert

Under this approach, the processing of a triggered alert will change as follows and as shown in figure 2:

If the alert frequency for this alert type has been exceeded or the maximum number of outstanding alerts for this healthcare worker has been exceeded:

Notify the system administrator of the alert overload

If the alert can be suspended:

Suspend the alert type in this treatment regimen for the worker, department or hospital.

If the alert type is not suspended:

Notify the alert recipients

Record the alert including:

Alert type

Date and time of the alert

Treatment plan

Treatment step

Healthcare worker assigned (at the time the alert was issued)

It may also be desirable to allow this process to generate its own alert under one particular circumstance. If the alert frequency for this type of alert has been exceeded and the treatment regimen designer has indicated that this alert can NOT be suspended, there is an indication of a serious problem. In such a circumstance it may be desirable to alert the treatment regimen designer themselves.

2.4 Batch analysis of the alerts

The healthcare worker responsible for maintaining the treatment regimen design as well as others involved in the continuous process improvement effort will need to review the recorded alerts on a regular basis. At a minimum, they will need automated tools capable of sorting the recorded data. This will allow them to cross tabulate the alerts by treatment regimen and alert type. They may also wish to examine other characteristics of the alert such as time of day, type of healthcare worker assigned, particular healthcare worker assigned, etc. The intent is to identify patterns in the data that indicate that there may be a common cause of all of the alerts of a particular type. If no pattern is discernible, it may be necessary to redefine the alert types to test theories regarding possible patterns.

In addition to tracking alerts specific to a particular healthcare worker, the continuous process improvement team will also need to track alert frequency at the department and hospital level. While this information can be collected by the alert management system, much of it will already be available via the billing system. For example, each person admitted to the hospital through the emergency room will constitute an event. Each patient treated by a particular department will constitute an event. When the frequency of these events begins to reach the processing capacity of the department involved, the CPI team should investigate to determine if

a) additional resources are required, or
b) there is a seasonal impact indicating that the spike in patients is temporary or
c) some other reason is apparent that is causing the system to reach capacity.

Once a pattern has been identified, root cause analysis proceeds as it would with a single alert. The most straightforward method will usually be the "five whys" approach taken from lean manufacturing. This should result in the identification of one or more actions to be taken to prevent these alerts from occurring again and improve the quality of the treatment being delivered. Once these steps (including changes to the treatment regimen, if any) are put in place, the designer will decide whether to suspend or activate the particular alert type moving forward. If the designer is confident that the number of alerts being issued will be manageable by the recipients, the alerts should be activated.

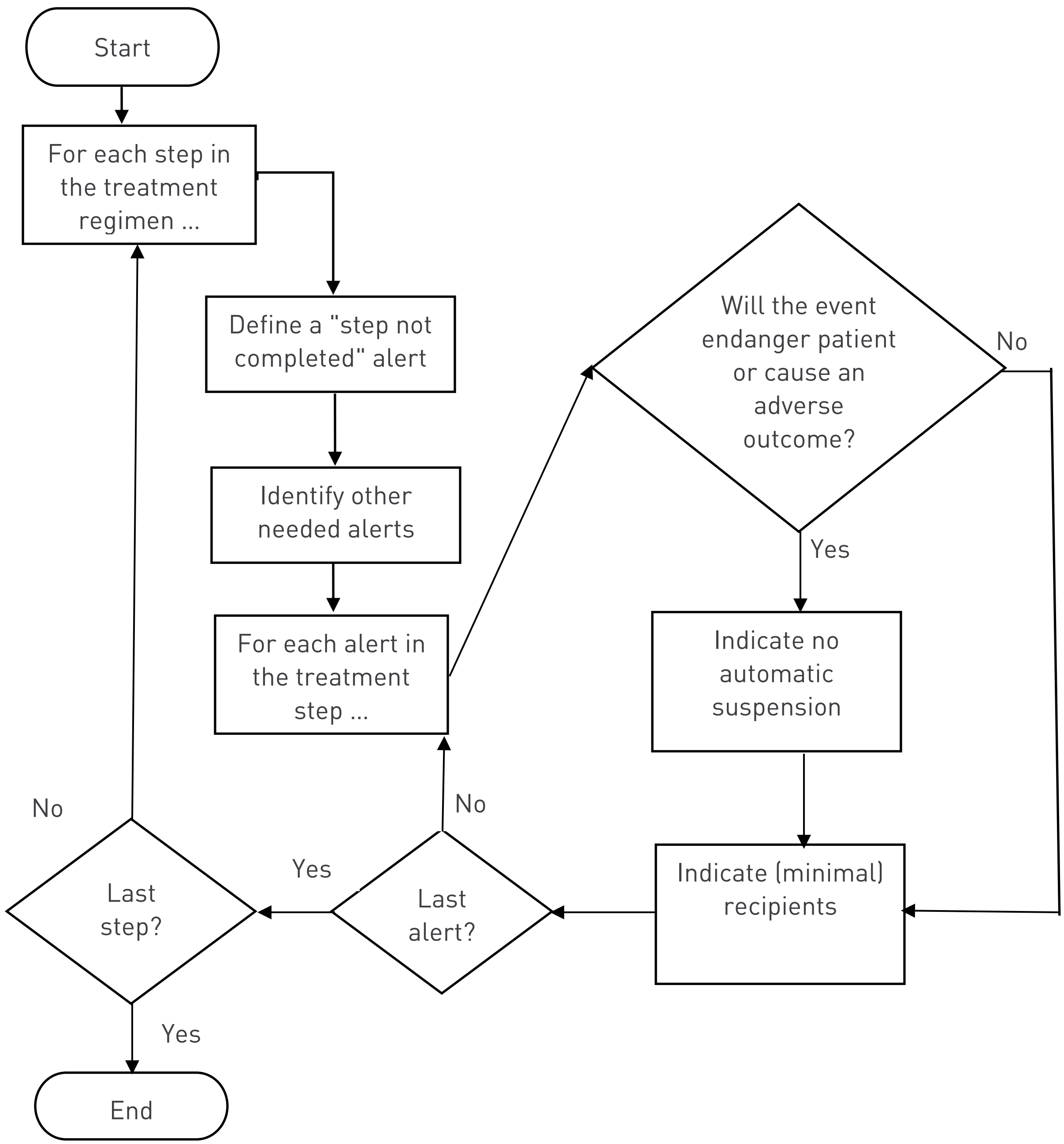
3.0 Expected results

In many EMR implementations, there is no monitoring of treatment plan versus actual. Those implementations can't actually be considered patient treatment management systems since nothing is being managed. In other implementations, alerts have been initially attempted but poor design and management have resulted in alert fatigue. The resulting backlash from users of the system has often caused alerts to be abandoned. In such implementations, analysis of adverse events after the fact will often result in process improvement in the treatment planning process but no improvement in the treatment delivery process. If the system described above is used effectively, the results should be a dramatic increase in the quality of treatment delivery and a dramatic reduction in medical error, mortality rates and readmission rates.

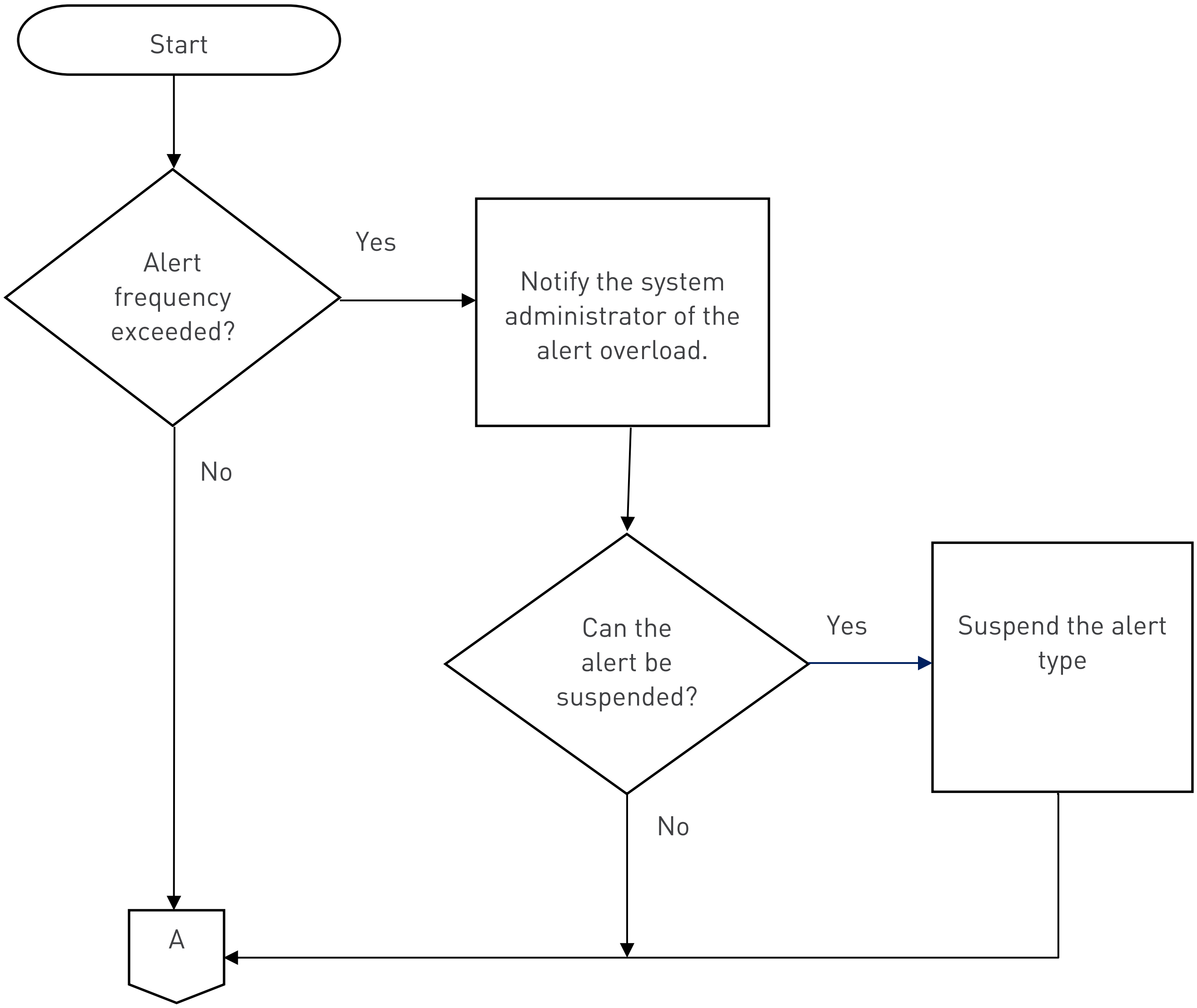
In hospitals where reimbursement is per capita based or where the hospital is affiliated with an insurance provider whose profit is based on cost per capita, profit margins for both organizations should be dramatically improved. Hospitals still on a fee-for-service-based model may see a reduction in demand for their services initially until the improved quality of patient care increases their market share within their community. Even those hospitals will see an improved negotiating position vis-à-vis the insurance providers pressuring them to reduce costs.

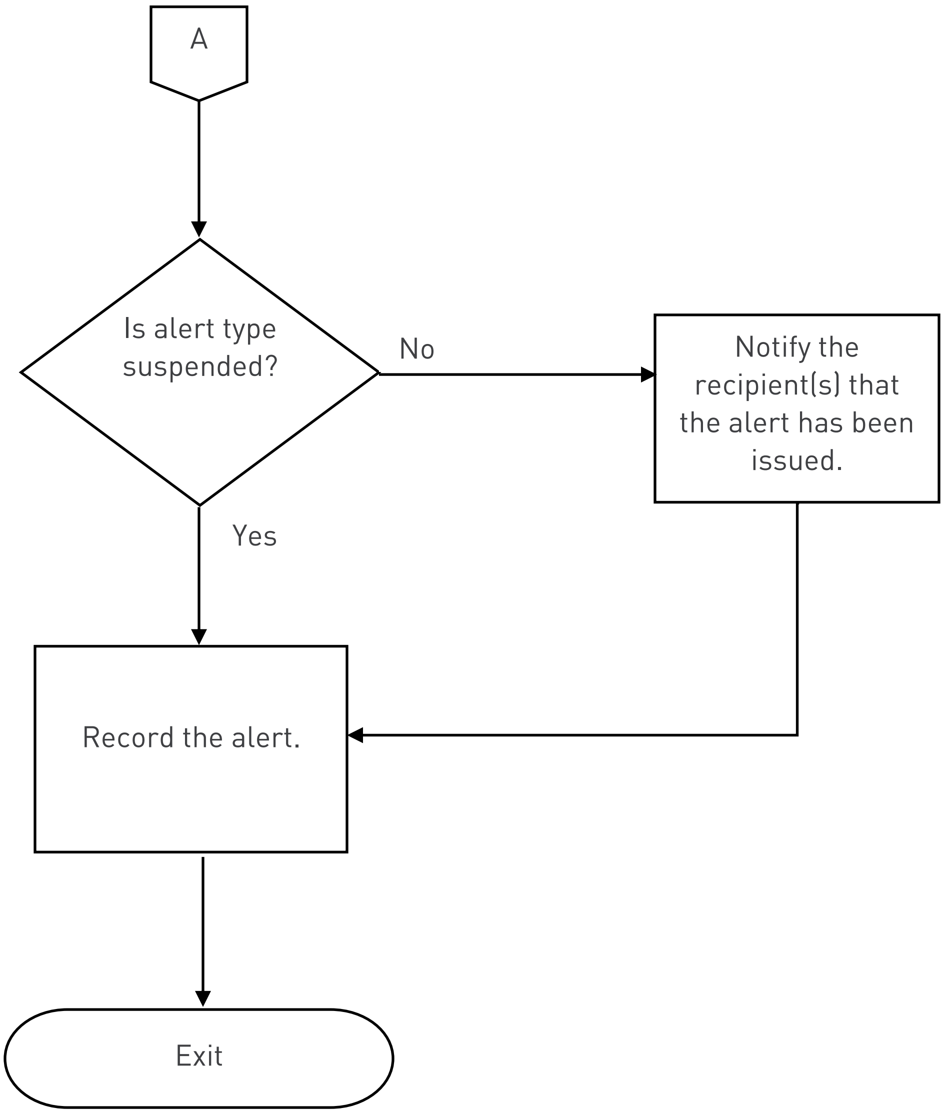
4.0 Figures and Flowcharts

4.1 Figure 1. Creating/modifying an alert



4.2 Figure 2. Issuing an alert




A
Is alert type suspended?
No
Notify the recipient(s) that the alert has been issued.
Yes
Record the alert.
Exit