CONTROLLING ALERTS IN YOUR ENSEMBLE OR HEALTHSHARE HL7 ENVIRONMENT

INTRODUCTION

HL7 V2 message routing solutions often have hundreds of business services, processes and operations. Inevitably things will go wrong and you have to monitor the production to be able to react quickly and resolve any problems before they can become serious issues.

Ensemble includes some powerful alerting capabilities that can help, but if you aren't careful you will be inundated with alerts and your inbox more or less becomes a copy of the event log defeating the object of the exercise.

These notes are based on experience of other customers who have set up alerting and will hopefully be a good starting point for anyone setting up alerting for the first time.

ALERTS IN GENERAL

Errors and various conditions can trigger an alert. An alert is just a message of type Ens.AlertRequest that will always be sent to a configuration item in your production called 'Ens.Alert'. If you don't have a configuration item of that name the alert will be written to the event log but not be sent anywhere. Ens.Alert can be a business process or a business operation that you have designed to handle the alerts. Typically the alert request is ultimately sent to a business operation such as EnsLib.EMail.AlertOperation so it can be sent as an email or text message. Some organizations have alert management applications serving many applications and Ensemble alerts can be fed in to this type of application but the exact interfaces vary.

SETTINGS THAT CONTROL ALERTS

Each configuration item can be configured using settings of the item to control the way alerts are generated. The settings include:

ALERT ON ERROR

Check this option on the configuration settings of an item if you want this item to generate alerts. The section below discusses settings that control what is considered an error.

ALERT RETRY GRACE PERIOD

This is a time in seconds. If the configuration item is retrying and eventually succeeds within this time there will be no alert.

Setting this to a value such as 60 seconds will suppress alerts on transient issues such as a dropped network connection that fixes itself, but won't wait too long before alerting you to a real problem.

QUEUE WAIT ALERT

This is a time in seconds. An alert will be generated if a message has been on the queue longer than this time in seconds. Setting this to a value such as 300 seconds would be a good starting point. For critical systems, 5 minutes might be too long, but for other less critical systems you might be happy with a longer interval.

QUEUE COUNT

Setting this to a value such as 10 will cause an alert when a queue starts to build on a critical item indicating some hold-up. But for some items this might not be an issue and if a component receives a nightly batch of work, alerting on queue size might not make sense at all.

INACTIVITY TIMEOUT

This is probably only of interest for a business service. If you set it to a value such as 300 seconds for a busy system you will be alerted fairly quickly if you stop receiving messages. For quieter systems you might want a longer interval.

This value applies all day, so you might get false alarms over night or on a holiday when traffic is much lower than normal, but you can filter these in your alert handler if necessary.

SETTINGS THAT CONTROL ERRORS

There are many configuration settings that control exactly when an error condition is triggered and. You should consider how all these should best be used in your environment but here are a few suggestions that are of particular importance for an HL7 routing solution.

STAY CONNECTED

If this is a positive integer, the connection will be dropped after that number of seconds with no activity.

Setting this = -1 means never disconnect and throw an error on a disconnect from the other end.

A large value such as 99999 means the connection will be dropped after an extremely long period of inactivity, but has the side effect of not throwing an error if the other end drops the connection.

FAILURE TIMEOUT

Setting this to -1 will mean messages will keep trying indefinitely, which is probably the desired behavior for an HL7 connection. If you set it to a positive number, the message will fail after that time and the Business Operation will move onto the next message.

REPLYCODE ACTION

This is a critical setting. This determines what you do when you get a failure. The values are fairly complex but like all settings you can see an explanation by clicking on the 'Replycode Action' label in config settings tab.

Typically you will want to retry indefinitely if there is an error sending the message. To ensure indefinite retries after all NACKs set FailureTimeout to -1 and ReplyCodeAction to :?R=RF and :?E=RF.

If the target application rejects the message (AR) retrying will probably have the same results so you might want to do something different. However applications can't always be relied on to use the ACK codes correctly and each application should be considered individually.

One option is to disable the BO after a rejected message but that stops all traffic. Suspending rejected messages until the problem can be fixed will allow other messages to flow but will break the FIFO rules.

REPORT ERRORS AND IGNORE MISSING SOURCE SEGMENT

These two settings on a DTL should probably be set to true (checked in the UI) to make sure that copying segments that aren't there won't cause a failure but all other errors will be reported.

VALIDATION

This setting on a message router controls whether or not a message is considered valid. If it isn't valid the message is routed to the BadMessageHandler. The default validation dm-z means that the message must a DocType and must have the correct segments structure, except that it allows trailing Z segments even if none are defined for this message structure.

Because HL7 messages so often don't meet the standard you may want to set this to 0 (zero) and route messages even if they are missing required segments. If you know that a target system needs certain things then you should take care of those in the routing rules or data transformations rather than fail to route the message.

ROUTING, FILTERING AND MODIFYING ALERTS

One of the most popular ways to handle alerts is to set up a rules based router, similar to the ones you use for HL7 messages.

In the rules for this router you have access to the alert text and the config item that generated the alert. Using these and possibly the CurrentDateTime() function you can decide where to route the alert request and which data transformation to call.

In the rule, you might want to filter out inactivity messages for some systems overnight and at weekends. Also you might want to filter out some messages that you don't think are important such as a failure to connect to a remote system within a certain period of time.

Data transformations can be used to modify the alert request if wanted. For example it could be used to add email recipients to the request.

Warning: you should not set alert on error to true for any item that is involved in delivering the alerts.