Research

Closed loop notification software

Driving productivity and higher service levels while mitigating risk

Author: Steve Craggs

August 2008

Version 1.00

Table of Contents

Executive Summary

IT systems have utilized alerts since computers began, with the earliest examples being the lights display on the computer console and the operator screen with all its messages. The concept is simple – the system detects some sort of situation or event, and decides it needs to let someone know in case something needs to be done. Today, this concept is used far more widely, and has become embedded in IT activity.

But as companies struggle to improve the business value delivered by their IT investments, streamlining and optimizing operational process execution as much as possible, this idea of delivering a warning and then standing back awaiting resolution is totally unacceptable. Instead, companies are looking for ways to maintain service delivery targets and achieve efficiencies by embedding the workflow needed to resolve these out-of-line situations into overall business process execution. This reduces or eliminates human latency when handling the problem and ensures a successful resolution.

The challenges in achieving this are many, however. The resolution workflow may well involve having to locate the right person to contact, finding the quickest way to get hold of them and even escalating the situation if that person is unavailable or unable to resolve the problem. On top of this, in order to continually improve the process it will be important to be able to see how the problem was resolved. All of this means that rather than problem resolution being a reaction to a warning light, the whole process is a two-way communication that takes its part in the overall process flow.

1. Locate the right person
2. Identify quickest way to contact them
3. Escalate situation if necessary
4. Track
5. Confim that the situation has been resolved

Fortunately, specialist software suppliers have emerged to handle these needs. Instead of handling situations in terms of alerts, that is fire-and-forget warning lights, these suppliers deal with notifications – that is, true information exchange. What is more, the best of these suppliers have embraced the idea of closed loop notifications, where the focus is on the end-to-end workflow required to move all the way from notifying the appropriate parties to confirming that the situation has been satisfactorily addressed. This paper looks at the whole area of closed loop notification software, and provides a best-of-breed functional checklist designed to aid users in selecting the right software offering for their needs.

Introduction

Alerts / notifications background

The need for alerts has been around for many years in the IT industry. Basically, ever since the first operating system needed to raise something to the attention of the operator, alerts have been generated to warn the operations team of particular technical situations or problems. Then, with the advent of systems monitors, the generation of alerts became much more widespread, with alerts being sent to systems programming teams, operators and technical specialists according to the specific nature of the identified situation or problem.

While systems were essentially mainframe based, with banks of dumb terminals attached, it was relatively easy for these alerts to be sent to the appropriate screen for attention, but once systems became more distributed this started to raise the first real issues. Now, it became necessary for an alert to be dispatched to one or more locations across the network, a much bigger challenge than just taking the action locally. Another problem arose from the fact that many different software components across the infrastructure and application portfolios started to see the value of alerting operational support teams of particular technical occurrences affecting their individual operations, and suddenly alerts were being generated from many different sources and in many different ways, introducing considerable confusion.

Lately, there has been another interesting development. The industry has gradually shifted from talking about alerts to discussing notifications. This change may seem minor, but in fact it represents an important change of perspective. The initial alert concept was to raise a flag to the operational team related to some technical issue that might require some corresponding action. It was the equivalent of putting a yellow or red warning light on an operational display, much as

Alerts
• Raise a flag (fire and forget)
• Communicate technical situation
• No follow-up

Notifications
• Ensure resource is notified
• Infer business context
• Track and monitor resolution

systems monitoring tools do on their systems status displays. By watching these displays, the operational team could monitor the health of the system and take any necessary actions as required. In other words, alerts were associated with a 'fire and forget' mode of operation – that is, the software involved would raise an alert based on a particular technical occurrence, and leave any conclusions to be drawn by the operational staff. In contrast, notification is a stronger word, implying a more directed communication and a corresponding information exchange.
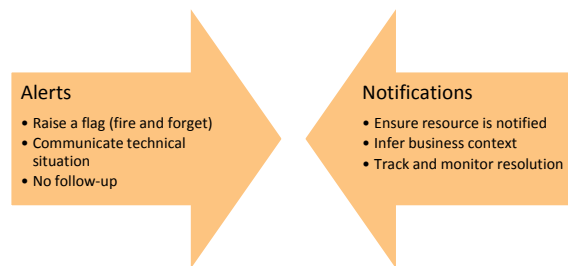
The evolution of the alert / notification software market

Over time, the requirements users have placed on the software handling alerts and notifications have increased dramatically. There are two main aspects to this shift in user needs:

- The need to expand capabilities, particularly based on technology advances
- The strategic imperative to improve the alignment of IT operations with business goals

A clear example of the first category is the requirement to support a wider range of communications channels for distributing the alerts. Whereas in the beginning alerts were usually raised on the operator's screen or through a panel of warning lights, communications today might be through such media as email, mobile phone, pager, instant messaging (IM) or voice. Other requirements might include the need to be able to broadcast some alerts to a group of different users and roles, for instance to inform them that a service is being closed down for a few minutes, or the ability to display some sort of dashboard where alerts can be illustrated in a more visual way such as through the use of red/yellow/green indicators.

The second category of changes is in a different direction entirely. Over the years, companies have been increasingly trying to improve the relationship between their IT investments and the overall business strategy and goals. This involves aligning IT operations more closely with the business processes and activities being executed, so that the IT systems support the goals of the business more effectively. From an IT systems perspective, this represents a major shift. Although IT would always say that the systems have been developed in response to the needs of the business, the fact is that the relationship between the two is often quite a 'hands off' affair. As an illustration of this point, most monitoring tools in the past would show the performance of the IT systems in great detail, showing which programs were executing and the health and load on various resources. But this information is of little value to a business observer, because there is little or no obvious relationship between the IT components being monitored and the business activities being carried out.

As companies strive to improve the linkage between IT operations and the business activities they represent, focus has turned to ways to influence IT operations based on what is happening in business terms. This is one of the main reasons why technologies such as business process management (BPM) and business events handling have become so topical. BPM enables business processes to be much more tightly associated with the underlying IT implementation, making it possible to monitor IT operations in business rather than technical performance terms, while business events support enables users to define particular business conditions that might occur and associate some corresponding action to them. The result is that IT systems start to respond much more closely and dynamically to business demands, increasing business performance and effectiveness.

It is this shift that results in greater pressure on alert software. All of a sudden, the alerts being raised are no longer of a technical, IT-based nature but are instead the result of some business condition. In other words, the raising of alerts is much more clearly part of the overall business process, and the value of the alerts increases. But this introduces major new challenges for the alert software. In the old days, alerts could simply be sent to the operational bridge, where someone would see it and go and find the appropriate team member to decide what action to take and carry it out. This was the 'fire and forget' approach to alerts, as described earlier. However, when the alert becomes part of a business process, it must take its place in the overall business activity workflow. The alert can no longer be sent to a 'catch-all' operational bridge, but must be automatically directed to the right person based on the particular business occurrence being detected. In other words, what is needed is 'closed loop' notification.

Closed loop notification

The dictionary defines an alert as '*warn or raise an alarm*', but notification as '*to inform or give notice to (a person)*'. Informing someone of a situation suggests a two-way exchange, while raising an alarm is more of a one-way action. So the move from 'alert' to 'notification' software accurately reflects the different perspective of dealing with an event as part of a business process rather than just as a warning light. However, the implications on the software handling the notification part of the workflow are extensive. The major challenge is to ensure that whatever event has occurred to trigger the notification, the right people will be informed and the appropriate actions taken.

This can be quite a complex problem to solve. For a start, the software that detects the situation needs to understand the implications and choose the right person or people to be informed. But once someone has been identified as the right person to receive notification of the event occurring, then the notification software has to contact that person in the most appropriate way, whether this be through an email, SMS text or whatever. Then there is the question of what to do when the person is not available. The workflow might well provide some sort of escalation process. Once someone has been located to address the situation, then there will be a requirement for some sort of acknowledgement to the notification software that everything is in hand. There will be other error situations too, such as the need to implement time-outs and have secondary process actions in the event of the first notification activity not succeeding.

The specific features required to address the issues of closed loop notification are discussed in more detail in the next section. However, the reason this area is getting so much attention today is that offering a closed loop notification service can deliver considerable benefit in terms of making the IT infrastructure and applications more responsive to business needs.

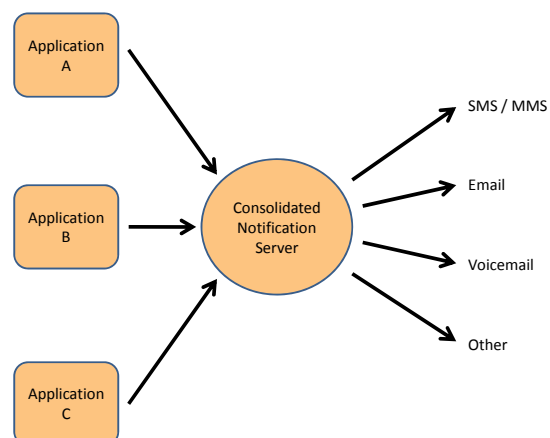# Closed loop notification software requirements

This section will look more closely at the types of functionality that will be required to deliver a closed loop notification capability. But first, it is worth observing that notification software has become a market segment in its own right rather than functionality built in to other infrastructure components.

Notification software

Obviously, the first requirement for any alert / notification solution is that it supports the widest possible range of communications mechanisms. This is important to allow companies the flexibility to use the most appropriate medium, and also to ensure that individuals can opt for the communications method with which they are most comfortable. So, for example, a company may have a general policy of using email for alerts, but might allow individuals the option of choosing to have alerts sent to them as SMS texts or voicemails. There may also be key business policies to take into account here; for instance, if one set of alerts is concerned specifically with disaster situations, it may well be explicitly required that such communications happen through an external network such as GSM to ensure that the alerts can be sent even if local systems and networks are unavailable.

One problem already touched on with alerts and notifications is the issue that many different applications and software infrastructure components need to raise them. While this was initially not a particular problem, as alerts and notifications have proliferated it has become a bigger issue. For a start, it can be extremely confusing to have multiple different pieces of software generating alerts, not least because the recipients may find themselves swamped with messages and be unable to sort them out. Then there is the perennial problem with having multiple different components delivering the same type of service – changes, modifications and maintenance have to be carried out separately for each component. So, for example, if a company decides to adopt SMS as the preferred communications medium, then this needs to be registered in multiple different notification software components.

As a result, software suppliers have emerged that specialize in delivering alert and notification software servers, or services. The idea is that if all components wanting to issue an alert or notification use the common notification service, then it becomes much easier for the user to control and manage these communications and future changes need only be carried out in one place. Also, since these suppliers can focus purely on the area of notifications, they can more easily keep up with technology advances and embrace new communications media quickly and effectively. Supporting closed loop notifications is an example of this; it is easier for a specialist notification software supplier to deliver a consistent solution to the need for closed loop notification services rather than to expect individual infrastructure components and applications to do this.

Closed loop notification requirements

When choosing a notification software supplier to satisfy closed loop notification needs, there are a number of specific functional requirements that should be reviewed prior to any product purchase. The following table identifies the key ones.

Key functional requirements for closed loop notification services	
Delivery / receipt support	Has the notification got through? Has it been acknowledged?
Multiple routing options	What is the preferred contact medium? Are there any alternatives?
Schedule / presence management	Is the desired person in?
Integration with calendar facilities	Where is the desired person?
Escalation process support	What if the notification is not being acted upon? What is the escalation process?
Two-way communications	How can the recipient acknowledge the notification or provide status updates?
Reliability and availability	Can I rely on the notification software delivering the notifications when required?
Reporting facilities	How well is the notification process working?
Integration with issuers of alerts / notifications	Can other software components utilize this notification service easily?

Figure 1: Key functional requirements for a closed loop notification offering

Depending on individual user requirements, these functions may not all be needed initially, but it is important to consider not just the immediate needs but the longer term requirements. These areas will now be discussed in greater detail. However, it should be remembered that it is assumed that any selected notification services offering will already have the basic functionality discussed previously.

Delivery / receipt support

The first point the user needs to know is whether the notification got through to its destination. This is similar to the receipts that can be requested when sending an email to another person; the two acknowledgements in this situation are that the message has been delivered into the inbox of the recipient, and that the recipient has actually read the email. So, the delivery / receipt support confirms that the notification has reached the target system, and that the target individual has picked it up. This functionality not only provides important feedback to the raiser of the notification, but also offers the opportunity to set up time-outs that can warn of out-of-line situations.

Multiple routing options

For important notifications, it may be extremely useful to be able to register a list of different routing options to be used. Then, if the first choice means of communication fails, others can be tried until the notification is acknowledged. Ideally, this functionality should allow the individual to specify preferences based on his or her own working practices. So, an employee who works at home might ask for any notifications to be sent through email, since he is often at his workstation even out of office hours, but he might specify SMS text as a secondary route in case he is out of the house. The key here is to try to give the individual as much flexibility as possible to increase the likelihood of the notification getting through first time.

Schedule / presence management

Scheduling is particularly important in the situation where a company has on-call rotas, for instance. In this situation, if an event occurs then the notification needs to be sent to whoever is now on call. This requires the

software to understand the rota in relation to current timings and circumstances. The user might also want a way to be able to register her presence so that any relevant notifications will be routed directly to her.

Integration with calendar facilities

A number of the required functions relate to trying to automate as much as possible in the activity of finding the appropriate person to notify. A powerful facility here is for the integration software to be able to integrate with office calendar facilities, such as provided in Microsoft Outlook. If a person is unavailable because they are off-site or tied up in meetings, then rather than have to wait until a time-out occurs to work out that the person is unavailable it is a great help if the notification software can make this determination automatically by consulting the appropriate calendar. Depending on the calendar system being used, this could involve bespoke software, but it provides a significant time-saver in the overall notification process.

Escalation process support

This is an essential piece of functionality if notifications are going to be practically integrated into corporate processes and workflows. Situations are bound to occur when the desired contact person is not available, for any one of a variety of reasons. In any closed loop notification process, it has to be possible to specify what to do in this event, particularly for the more important notifications. While this was not so much of an issue with technical alerts, where it could be generally relied upon that there would be someone on duty monitoring the operational bridge, with the more business-related alerts this is a much more likely scenario. The key to resolving this type of situation is to be able to specify some sort of escalation process – an action plan of different routes to try to get someone's attention in order to precipitate the appropriate action. The escalation process may need to go up and down organizational trees, perhaps repeating the same nests of contact sequences, but the result must be the successful identification and contact of someone who can take ownership of the event that has occurred.

Again, the extent of this functionality will affect the degree of automation it is possible to achieve. At a simple level, the escalation process might be to simply notify a 24-hour operations desk and leave whoever is manning the desk to work out who to call. However this is much less efficient than being able to specify a sequence of escalation procedures as part of the automated notification process.

Two-way communications

One of the reasons for talking about notifications as opposed to alerts is that in a process workflow, it is likely that there will be the need for two-way communications. One obvious requirement, particularly for critical notifications, is for the recipient to be able to explicitly acknowledge the notification. While the delivery / receipt support lets the sender know that the notification has been delivered, an acknowledgement confirms that the recipient has accepted the notification and is going to take the necessary action. This acknowledgement is handled as a notification in its own right. The other main need for two-way communications support is to provide the notification recipient with some way to feed status reports back to the sender. Both of these needs are essential when implementing a closed loop process workflow.

Reliability and availability

If notifications are to be a key element of business process operations and workflows, then it will be important to ensure that the closed loop notification software is available and reliable. The availability side can be addressed by providing the normal sort of failover options such as the ability to run a back-up system in parallel to the production one. However, the reliability question has a number of different aspects. The notification software must be able to tolerate the failure of particular communications mechanisms, and indeed this has been addressed in the requirement for multiple routing options. But the software may also need to ensure it does not accidentally send a notification more than once. In some workflows a duplicate notification could have very unpleasant side-effects. In essence, the notification software needs to make sure that any notifications are sent once and only once.

Reporting facilities

The whole point of a closed loop notification approach is that the notification resolution is an integral part of the overall business process. This raises the issue of reporting. Most companies are continually striving to streamline and improve their processes, to optimize efficiency and effectiveness. A vital component of this implementation / analysis / improvement cycle is having accurate records and reports of how the process is performing at the moment. The reports can then be analysed to identify areas for improvement. Therefore, a closed loop notification software offering will need to keep a clear history of any notifications raised, how they were processed and how they were finally addressed. This is one area where the requirement previously discussed for two-way communications is very useful – this enables the final resolver of the notification to send back information on how the situation was addressed, and this can become part of the historical activity report.

Integration with issuers of alerts / notifications

In order to gain maximum benefit from closed loop notification services, it will be necessary for the notification software to integrate with the various different infrastructure components and applications that are issuing the notifications. The major reason for this is that while an alert is a one-way level of communication, a notification is more of a two-way operation and therefore there will be a need for the raiser of the notification to be involved in the resultant process workflow. So, for example, if a notification times out, then the notification software knows it has not been delivered. It may have logic specified to describe alternative actions to take, such as escalations or trying alternate routing options, but if not then it needs to let the sender of the notification know that the message did not get through so that the sender can decide what to do next. The notification software may be unaware of the implications of a particular notification, for example, and therefore it would be up to the sender to precipitate some other action if the detected situation is an important one.

While the closed loop notification supplier might be able to partner with other vendors to ensure integration of their components that issue notifications, some companies may have their own software that they have developed that also needs to send notifications. It may therefore be necessary for the notification software supplier to provide some sort of software developer kit to allow on-site customization of home-grown applications and components.

The benefits of closed loop notifications

The previous section described the types of functions required in any closed loop notification software offering, but is this type of functionality really important? In fact, adopting a closed loop approach to notifications can have some dramatic benefits. The prime benefits are:

- Improved productivity
- Greater process efficiency and reduced latency
- Enhanced availability and service levels
- Reduced operational risk

On the personal productivity side, there are a number of aspects of closed loop notification software that are of benefit. Ensuring that the workflow from the detection of a defined event to the closure of the incident is as automated as possible reduces the time people need to take to get involved. A lot of time can be wasted if an incident requires someone to have to start phoning around to get someone to look at the problem, only to be told 'it is not my area'. Also, if there is no closure to the notification process then the onus will be on an operator or some other support person to monitor the situation to see if the situation is resolved.

This same improvement in process automation obviously contributes greatly to better process efficiency too. However, this is not just an 'in the moment' observation – having clear reports of notifications that have been issued, how the right person was eventually reached and the resulting actions taken provide exactly the information needed to continually analyse and improve the process, increasing efficiency on an iterative basis. A

lot of the process efficiency improvements will stem from the reduced latency, by having decisions taken automatically based on pre-defined corporate policies and procedures rather than having to ask staff to make these sorts of decisions on the fly.

The improvement to availability and levels of service comes mainly from the fact that, in the situations where the notifications raised are of a business critical nature, these notifications are quickly and safely delivered to the right people so that rapid resolution can be carried out. The ability to deal with shift rotas, calendars, multiple different methods of communications and escalations all contribute to getting someone working on the problem with the greatest possible speed, reducing the time to resolution.

All of the above factors contribute to an overall mitigation of operational risk. Using closed loop notification software ensures that situations that could damage operational viability and effectiveness are handled efficiently and reliably, bringing the appropriate resources to bear in the shortest possible time. The ability to support notifications through external networks even helps to mitigate the impact of a major disaster or outage, since the relevant people can be notified even when the internal networks are down.

Summary

The old idea of systems raising alerts has run its course, and the modern business should instead be focusing on generating notifications that inform about defined events rather than alerts that warn. But raising a notification is not as easy as it sounds – the right person has to be identified and contacted through whatever communications means will work best, in order for the cause of the notification to be resolved. Then there is the issue of locating the contact, or finding a replacement if the original person is unavailable.

It is quickly becoming apparent that in order to get the maximum efficiency and reliability from operational processes, the act of resolving any out-of-line situations is critically important, and for that reason many companies are keen to find a way to bring this part of operational workflow into the overall business process. It is not enough for a flag to be raised to indicate there might be a problem; instead, the process should encompass the handling and resolution of the problem too.

For this reason, more and more companies are looking for software that can deliver closed loop notification services, where there is a clear feedback loop between the raising of a notification and the resolution of the issue and where the workflow required to achieve this is as automated as possible. While many software components include some sort of limited alert or notification services, the specialized nature of closed loop notification services means that the best fit for this requirement is likely to be a specialized vendor focused on delivering generic, closed loop notification software that can consolidate notification activities and provide reliable closed loop processing for all.

About Lustratus Research

Lustratus Research Limited, founded in 2006, aims to deliver independent and unbiased analysis of global software technology trends for senior IT and business unit management, shedding light on the latest developments and best practices and interpreting them into business value and impact. Lustratus analysts include some of the top thought leaders worldwide in infrastructure software.

Lustratus offers a unique structure of materials, consisting of three categories—Insights, Reports and Research. The Insight offers concise analysis and opinion, while the Report offers more comprehensive breadth and depth. Research documents provide the results of practical investigations and experiences. Lustratus prides itself on bringing the technical and business aspects of technology and best practices together, in order to clearly address the business impacts. Each Lustratus document is graded based on its technical or business orientation, as a guide to readers.

Terms and Conditions