NETTIES 2005, OCTOBER 2005

NETTIES 2005

# ESCALATION MANAGEMENT WITH THE HELP OF OPEN SOURCE TOOLS IN A LOCAL AREA NETWORK AND SERVER ENVIRONMENT

Mag. Walter Sedlacek, MSc MBA

*Keywords: Escalation Management, Open Source, Nagios, Business Processes*

## Abstract

In today's IT departments, escalation management is one of the most critical processes. It must be very fast and stable at the same time. Changing technologies and heterogeneous architectures complicate the implementation of state of the art tools like HP Openview[1] or Tivoli[2]. In addition, due to financial constraints, the high licence costs of those tools make it very difficult to implement them area-wide.

This paper will show that the measurement of important IT services and the comparison with an agreed service level can be done at a low cost, while at the same time achieving a very high quality. Open Source tools trigger the designed escalation process and enable in-time action and information. Both the management and the technical operations team receive the right information early enough to set further actions.

[1] http://www.openview.hp.com/

[2] http://www-3.ibm.com/software/tivoli/

# Content

## Figures

# 1 Open Source: History and Definition

*"No traditional developer can match the pool of talent the Linux community can bring to bear on a problem. Very few could afford even to hire the e.g. 800 people who have contributed to Open Source projects."*[3]

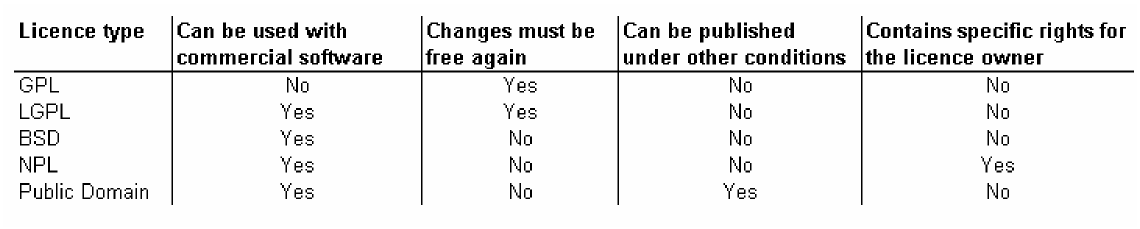
*"Hewlett-Packard is hosting a number of Open Source software projects that run on various Hewlett-Packard systems."*[4]

## 1.1 Definition

One of the main problems in traditional software development is that the one who finds an error in an application communicates this to another person – the software engineer – who fixes the problem. One can easily see that this process is time-consuming and not very effective due to various sources of misinterpretation.

The idea behind Open Source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. It is often the same person who finds the error, fixes it and tests the fix.

There are many Open Source licence types. The main difference is the possibility of re-using the software code for commercial or non-commercial applications:



| Licence type | Can be used with commercial software | Changes must be free again | Can be published under other conditions | Contains specific rights for the licence owner |
|---|---|---|---|---|
| GPL | No | Yes | No | No |
| LGPL | Yes | Yes | No | No |
| BSD | Yes | No | No | No |
| NPL | Yes | No | No | Yes |
| Public Domain | Yes | No | Yes | No |

**Figure 1: Open Source licence types**

[3] http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s11.html

[4] http://www.opensource.hp.com/

Open source does not just mean access to the source code. The distribution terms of Open Source software must comply with the following main criteria:

- The license shall not require a fee for selling the product.
- The program must include source code, and must allow distribution in source code as well as compiled form.
- The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

## 1.2 History

In 1956 AT&T[5] was forced to limit its activities to its core business tele-communication and sell its patents on the free market. One of those patents was an operation system called UNIX, developed 1969 by Ken Thomson and Dennies Ritchie in the Bell Labs[6]. AT&T did not want other companies to have UNIX and therefore the price was incredibly high and there was no support at all. Universities got UNIX for small fees but still no support was available. That is why the exchange of knowledge and source code between universities was a must in order to enhance UNIX and get errors fixed. The coordination was done by the University of Berkeley[7], which later published UNIX by using the in-house software distribution BSD[8]. One of the first versions was announced in 1978 by Bill Joy, who later founded SUN[9] . In the same year Arpanet – the former Internet – was established, based on UNIX computers. 1982 IBM[10], HP[11] and DEC[12] published their versions of UNIX and sold it on the free market, now including support. This was the end of more or less free UNIX versions.

[5] http://www.att.com/

[6] http://www.bell-labs.com/

[7] http://www.berkeley.edu/

[8] http://www.bsd.org/

[9] http://www.sun.com/

[10] http://www.ibm.com/

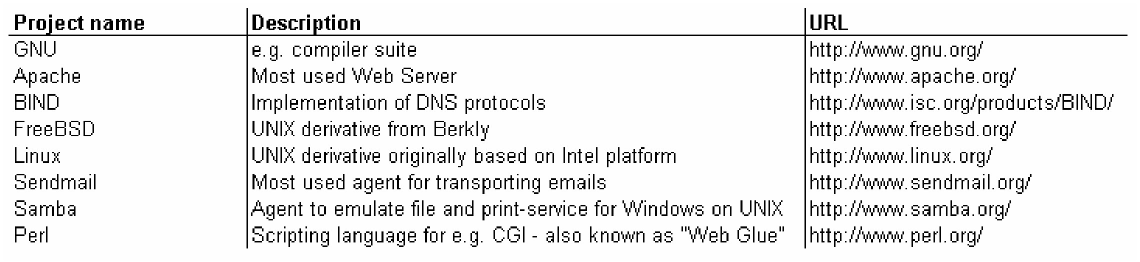
[11] http://www.hp.com/

[12] http://www.digital.com/

In order to keep free UNIX versions available Richard Stallmann founded the company Free Software Foundation[13] and the GNU project[14]. With GPL[15] Stallmann defined a new license-model, which basically ensures that free software stays free after being modified by others. At this time no Kernel for a free UNIX system was available. This problem was first solved by Linus Torvalds[16] by publishing the first Linux[17] Kernel in 1991. This date can be seen as the birthday of Open Source, because now a free operating system including major utilities and applications was available.

## 1.3 Projects

Most users are not aware that many applications they use are from the Open Source world. Most of the UNIX utilities are developed under GPL license. For example the user-friendly Domain Names for Internet web-sites would be not possible without BIND[18], another Open Source product. The most frequently used web-server is Apache[19], which is also free. The following table gives an overview of the most common Open Source applications.



| Project name | Description | URL |
|---|---|---|
| GNU | e.g. compiler suite | http://www.gnu.org/ |
| Apache | Most used Web Server | http://www.apache.org/ |
| BIND | Implementation of DNS protocols | http://www.isc.org/products/BIND/ |
| FreeBSD | UNIX derivative from Berkly | http://www.freebsd.org/ |
| Linux | UNIX derivative originally based on Intel platform | http://www.linux.org/ |
| Sendmail | Most used agent for transporting emails | http://www.sendmail.org/ |
| Samba | Agent to emulate file and print-service for Windows on UNIX | http://www.samba.org/ |
| Perl | Scripting language for e.g. CGI - also known as "Web Glue" | http://www.perl.org/ |

**Figure 2: Most common Open Source projects**

[13] http://www.gnu.org/fsf/fsf.html

[14] http://www.gnu.org/

[15] http://www.gnu.org/copyleft/gpl.html

[16] http://www.cs.helsinki.fi/u/torvalds/

[17] http://www.linux.org/

[18] http://www.isc.org/products/BIND/

[19] http://www.apache.org/

## 1.4 Traditional Software Management versus Open Source

One of the most common arguments against Open Source is that those projects are lacking management in a traditional way. Software managers often argue that each software project needs the following qualities, executed by a manager[20]:

1. To define goals and keep everybody pointed in the same direction
2. To monitor and make sure crucial details don't get skipped
3. To motivate people
4. To organize the deployment of people for best productivity
5. To marshal resources needed to sustain the project

ad 1: One of the best-known theorems of software engineering is that 60% to 75% of conventional software projects either are never completed or are rejected by their intended users. If that range is anywhere near true then more projects than not are being aimed at goals that are either not realistically attainable, or simply wrong. In summary many goals fail in *normal* software project management and many have great success in the Open Source world (e.g., the longevity of Emacs[21], or Linus Torvald's ability to mobilize hordes of developers). Thus the project leaders and tribal elders who fill the manager's role in the Open Source world deliver equal or better qualities in defining goals than traditional software managers do.

ad 2 and 3: The strongest argument of the Open Source community is that decentralized peer review trumps all the conventional methods for trying to ensure that details don't get out of control. Motivation is one of the key elements why tracking and monitoring works in such an environment.

ad 4: Open Source has been successful partly because its culture only accepts the most talented 5% of the programming population. It is often cheaper and more effective to

[20] http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s12.html

[21] http://www.gnu.org/software/emacs/emacs.html

recruit self-selected volunteers from the Internet than it is to manage buildings full of people who would rather be doing something else.

<u>ad 5:</u> Resource marshalling is basically defensive; once you have your people and machines and office space, you have to defend them from other managers competing for the same resources. But Open Source developers are volunteers, self-selected for both interest and ability to contribute to the projects they work on. The volunteer ethos tends to take care of the “attack” side of resource-marshalling automatically - people bring their own resources to the table.

# 2 Problems of Today’s Escalation Management

*“If you do not measure IT, you can not manage it.”*

Besides others, there is one scenario where Escalation Management with the help of Open Source tools can definitely increase the efficiency of IT operations: Outsourcing in large enterprises.

In multi-national enterprises IT operations are mostly run by suppliers based on Service Level Agreements (SLAs). To measure the IT services tools are necessary. In today’s outsourcing environments those tools are delivered by the same vendor who provides the service. Of course those tools are state-of-the-art, complex, expensive and very important – at least for the supplier. In most cases the IT-department of the company which purchased the service can view specific results of the measurement tool on a dedicated screen (*view*). This specific screen can be viewed by the IT-department; all the others and the possibility to create new ones is reserved for the supplier. The following scenario describes this situation:

An IT supplier provides a VPN WAN service, based on MPLS[22] technology, for two remote offices (Point A and Point B). In many cases the IT department of the company which pays the service gets only a *view* from the monitoring system the supplier uses. The disadvantages of this model are:

- The *view* is only a subset of all the information provided by the monitoring system.
- The form and behaviour of the *view* is dictated by the supplier. If the company wants to change the *view* or wants to have some new ones the supplier must always be involved.
- The measurement of the supplier’s service level is done by the supplier. This can be compared with Intel[23] writing it’s own benchmark software to prove that their newest CPU is faster than the competition’s.

[22] Multiprotocol Label Switching;
http://searchnetworking.techtarget.com/sDefinition/0,,sid7_gci214350,00.html

[23] http://www.intel.com/

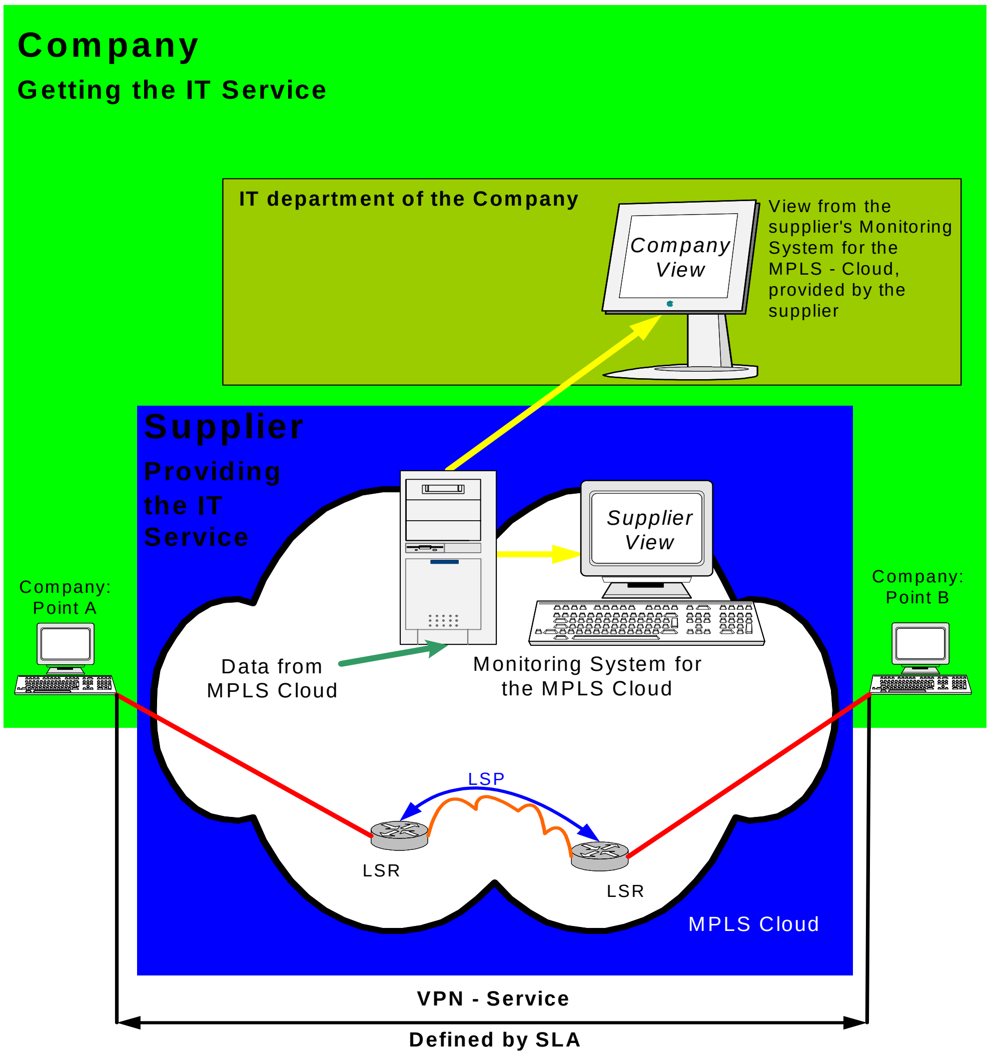


**Figure 3: Traditional monitoring in an outsourcing environment**

Although the contract between the company and the supplier should define the monitoring, reality shows that this definition always leads to discussion, where the issues mentioned above are topics. And if the IT department asks the supplier how to create it’s own measurement system very often the answer is to copy the existing one – which in many cases is an overkill.

This thesis will show that there is an inexpensive  way to build a measurement system, which can verify if the measured values delivered by the supplier are correct and which is also flexible enough to create additional *views*. The implementation of such a measurement system leads to the following adaptation of the last figure: In addition to the existing measurement system – provided by the supplier– there will be a second measurement system fully owned by the IT department.

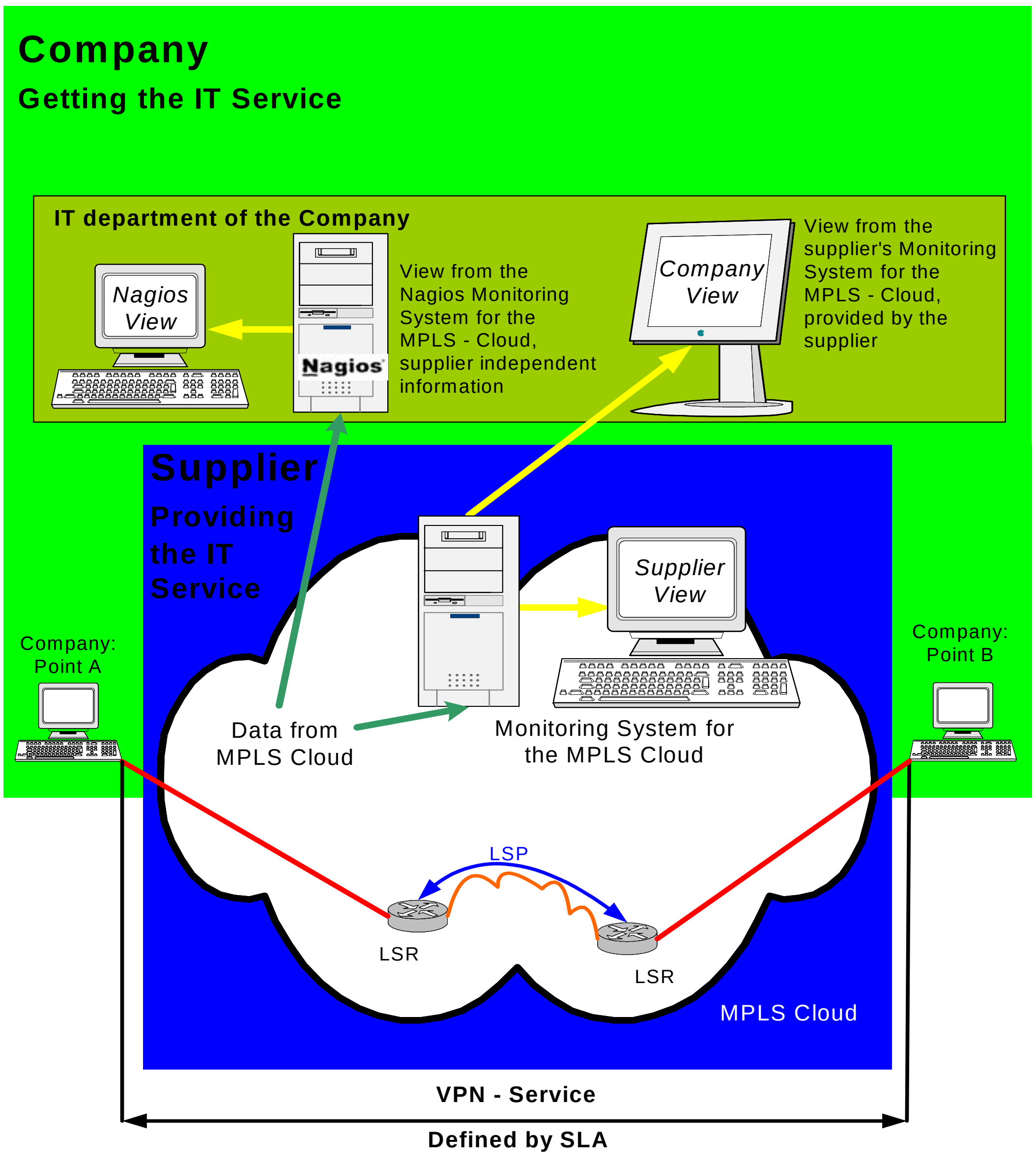

Company
Getting the IT Service
IT department of the Company
Nagios View
Nagios
View from the Nagios Monitoring System for the MPLS - Cloud, supplier independent information
Company View
View from the supplier's Monitoring System for the MPLS - Cloud, provided by the supplier
Supplier
Providing the IT Service
Supplier View
Company: Point A
Company: Point B
Data from MPLS Cloud
Monitoring System for the MPLS Cloud
LSP
LSR
LSR
MPLS Cloud
VPN - Service
Defined by SLA

**Figure 4: Monitoring with an additional measurement system.**

The example described above is located in the Wide Area Network environment, but it illustrates the generic idea of this thesis: Measurement of service levels by the IT department. The further chapters describe scenarios for Local Area Networks and servers, but can easily be extended to other areas of IT, e.g. Wide Area Networks or Wireless Networks.

The following figure shows at which points a measurement in Local Area Networks and Servers is useful. How these measurements are realized with Open Source tools and how this enables the Escalation Management is described in the following chapters.

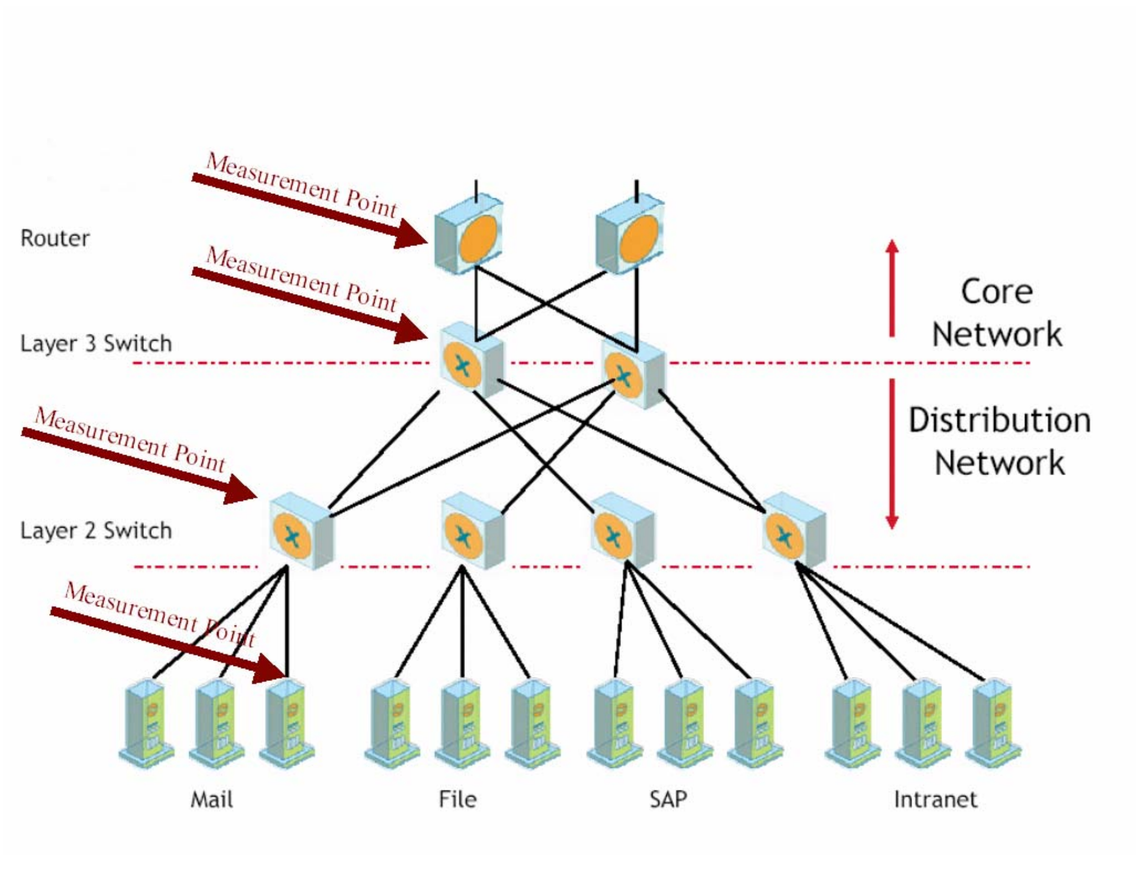


**Figure 5: Measurement points in LANs and for servers**

# 3 Design of the Escalation Management Process

## 3.1 Generic considerations

The following scenario describes the generic IT problem-handling process:

A problem occurs which has a certain impact on the business (1). Then somehow the error is noticed (2) and communicated to one or more persons who fix the problem. At the same time this error is communicated to the employees affected by the outage (3) (4), so that they start the process for business contingency (5). After a certain time the problem is fixed and again the business employees must be informed (6) to stop the process for business contingency. If the problem is not fixed, supervisors must be informed to set further actions (7).

The following example illustrates IT operations without automated escalation processes:

> On a Monday morning the Microsoft Exchange server has a CPU utilization of 95% due to a virus infection. The business impact is that 80% of all users can not read their emails (1). Because there is no monitoring tool, a user is the first to notice the outage, when he can not connect his Outlook client to the Exchange server (2). This user calls a friend in another department and asks him if he has the same problem. When the friend confirms, both call the Exchange server administrator. The administrator now starts to fix the Exchange server and both users tell their colleagues about the problem (3) (4) and that they should utilize the telephone instead of email for communication (5). If the problem is fixed (the administrator downloads the newest Anti-Virus patterns from the Internet and cleans the virus from the Exchange server) the users will be informed by the administrator that they can use Outlook again via group-voice-mail functionality of the PBX (6).
>
> If after a couple of hours the Exchange server administrator fails to clean the server of the virus and the problem still exists the users will ask their

supervisors and managers to call the CIO, who is the supervisor of the Exchange server administrator (7), and tell him very angrily that the problem must be fixed immediately. An embarrassing problem would be if the CIO was not informed about the Exchange server problem because he was in meetings all morning.

To automate the escalation process, the scenario should be analysed as followed and certain definitions must be made:

1. Problems must be grouped in *severity levels*, dependent on their business impact.
2. A system must *determine the outage* by comparing parameters with defined thresholds.
3. Groups of employees – *stakeholders* –, who need the same information at the same time  must be defined.
4. A *communication method* including the *format of the information* must be defined.
5. A *business contingency* process must be defined.
6. It must be defined who is to be informed at what time – *Information Flow*.
7. *Escalation* in the hierarchy must be defined.

To describe severity levels, outages, stakeholders, information method, format and business contingency process and to feed all those definitions into an information flow chart, the following very simplified scenario is given:
The local area network consists of 3 physical networks (production network, office network and server network), attached to a layer 3 switch, which does the routing between those networks. This LAN supports both the IT in the production area and the IT in the office area.

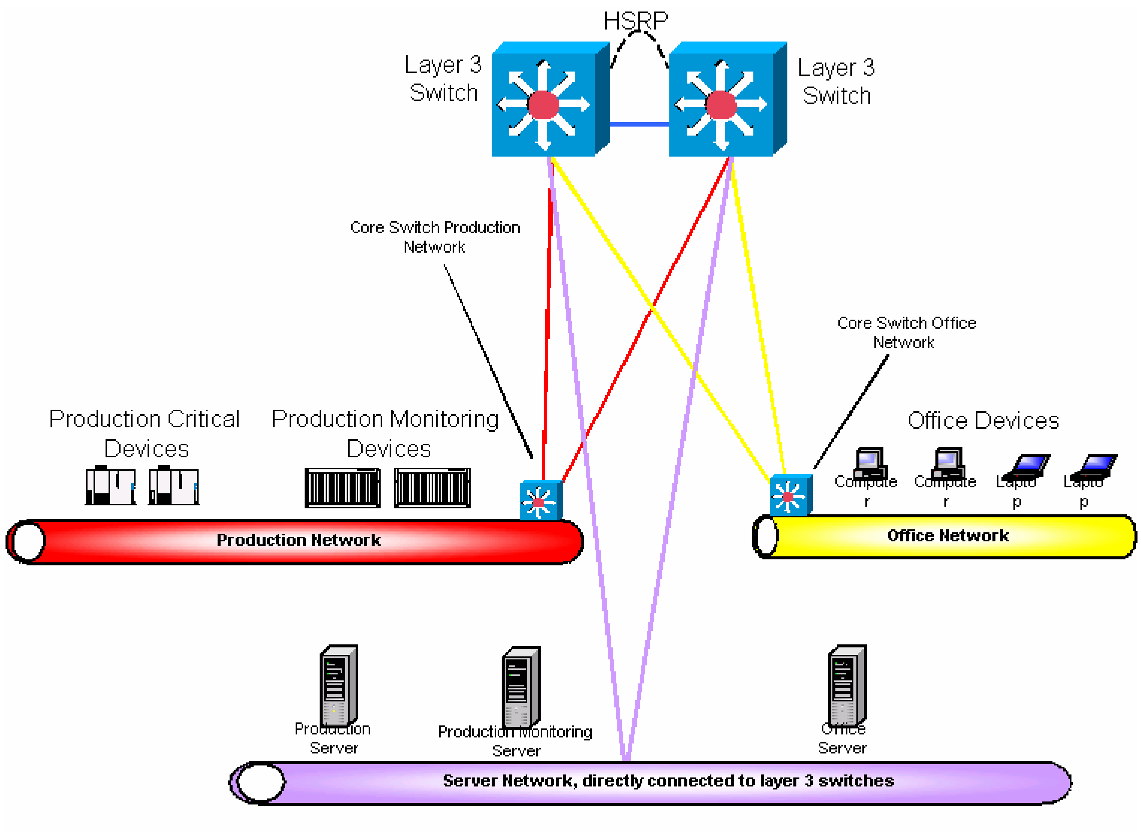

HSRP
Layer 3 Switch
Layer 3 Switch
Core Switch Production Network
Core Switch Office Network
Production Critical Devices
Production Monitoring Devices
Office Devices
Computer
Computer
Laptop
Laptop
Production Network
Office Network
Production Server
Production Monitoring Server
Office Server
Server Network, directly connected to layer 3 switches

**Figure 6: Generic simplified LAN and server scenario**

## 3.2 Define Severities Levels

The following severity levels define the impact an outage can have on the business:

Severity 1 (RED)

The production system is down. One of the following must be true:

- The server of the production system can not be reached via *ping* or a critical service on this server is not running.
- The layer 3 switches of the whole network can not be be reached via *ping* or certain parameters on those switches (e.g. CPU load) exceed a certain prior defined threshold.
- The core switches of the production network or the server network can not be reached via *ping* or certain parameters on these switches (e.g. CPU load) exceed a certain prior defined threshold.

- The access switches, where production critical devices are connected can not be reached via *ping* or certain parameters on those switches (e.g. CPU load) exceed a certain prior defined threshold.

Severity 2 (ORANGE)

The production monitoring system is down. One of the following must be true:

- The server of the production monitoring system can not be reached via *ping* or a critical service on the production monitoring server is not running.
- The access switches where production monitoring devices are connected can not be reached via *ping* or certain parameters on those switches (e.g. CPU load) exceed a certain prior defined threshold.

Note that the outages of the layer 3 switches and the core switches of the production network are covered by severity 1 (RED).

Severity 3 (YELLOW)

A business system (e.g. the  Exchange Server) is down. One of the following must be true:

- The server of the office system can not be reached via *ping* or a critical service on the business server is not running.
- The core switches of the office network or the server network can not be reached via *ping* or certain parameters on those switches (e.g. CPU load) exceed a certain prior defined threshold.
- The access switches where office devices (PCs or Laptops) are connected can not be reached via *ping* or certain parameters on those switches (e.g. CPU load) exceed a certain prior defined threshold.

Note that the outages of the layer 3 switches are covered by severity 1 (RED).

## 3.3 Determine Outages

Outages are automatically detected by using an electronic monitoring system. This system compares parameters with predefined thresholds and indicates when the value of the parameters exceeds the threshold. The chapter *Enabling the Escalation Management Process* describes this system in detail.

## 3.4 Define Stakeholders

The following groups of employees are stakeholders:

- Supervisors of departments in the production area: <u>Production Department Heads</u>
- Supervisors of departments in the office area: <u>Office Department Heads</u>
- Responsible person to run IT operations for Local Area Network and servers (this may be outsourced to an IT service provider): <u>Operations Manager</u>
- Supervisor of the Operations Manager (this may be outsourced to an IT service provider): <u>Service Manager</u>
- Responsible System Owner of the Local Area Network and servers in the IT department: <u>Infrastructure Manager</u>
- Supervisors of the Infrastructure Manager: <u>Chief Information Officer</u> (CIO)
- Supervisors of the CIO: <u>Chief Execution Officer</u> (CEO)

## 3.5 Define Method and Format of Information

Information must be delivered very fast and location independent. Both email and SMS on mobile phones should be utilized synchronously to enable fast information. The information flow described in chapter *Design Information Flow Including Escalation* is always done via email and SMS.
Part of the email and SMS is an URL, which links to a diagram in the Intranet. There the detailed impact on production or office is shown. However the description of all possible impacts overstrains this thesis.
All information delivered to the various stakeholders is logged together with a time stamp in the electronic monitoring system.

## 3.6 Define Business Contingency Process

The business contingency process depends heavily on the outage, e.g. for an outage of the production monitoring system a manual monitoring with additional headcounts might be possible. However the description of all possible contingency processes would overstrain this thesis.

## 3.7 Design Information Flow Including Escalation

The following charts show the information flow depending on the severity levels:

Severity 1 (RED)

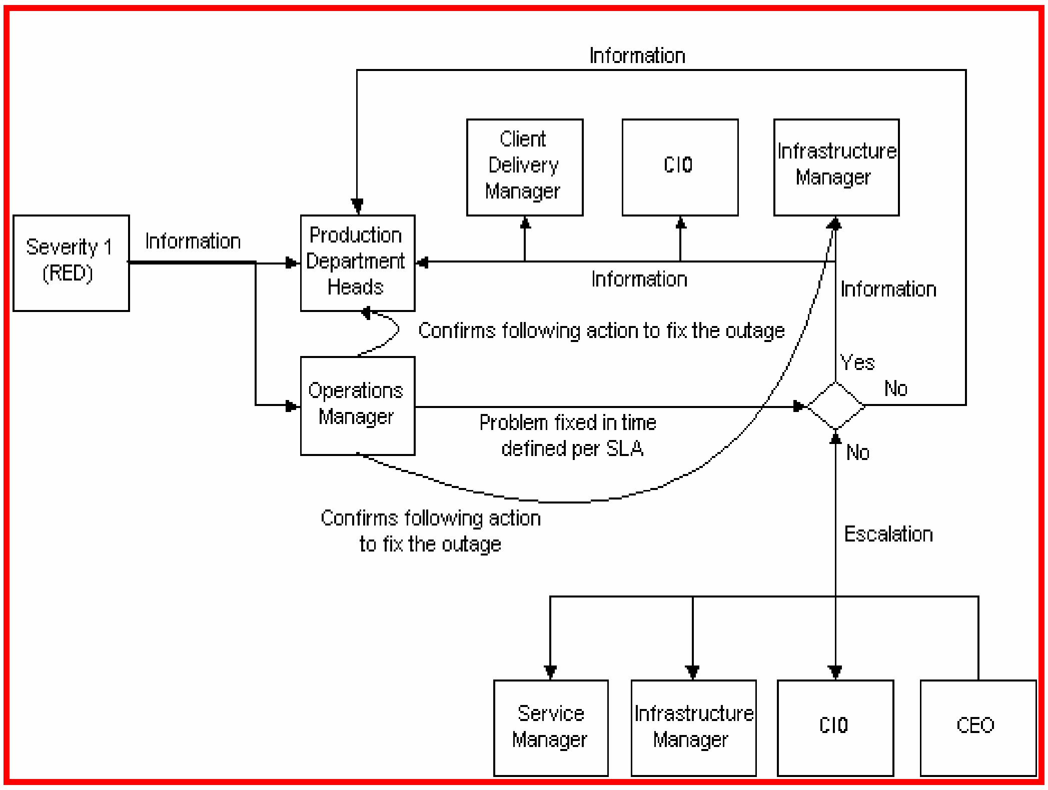


**Figure 7: Flowchart for Severity 1 (RED)**

## Severity 2 (ORANGE)

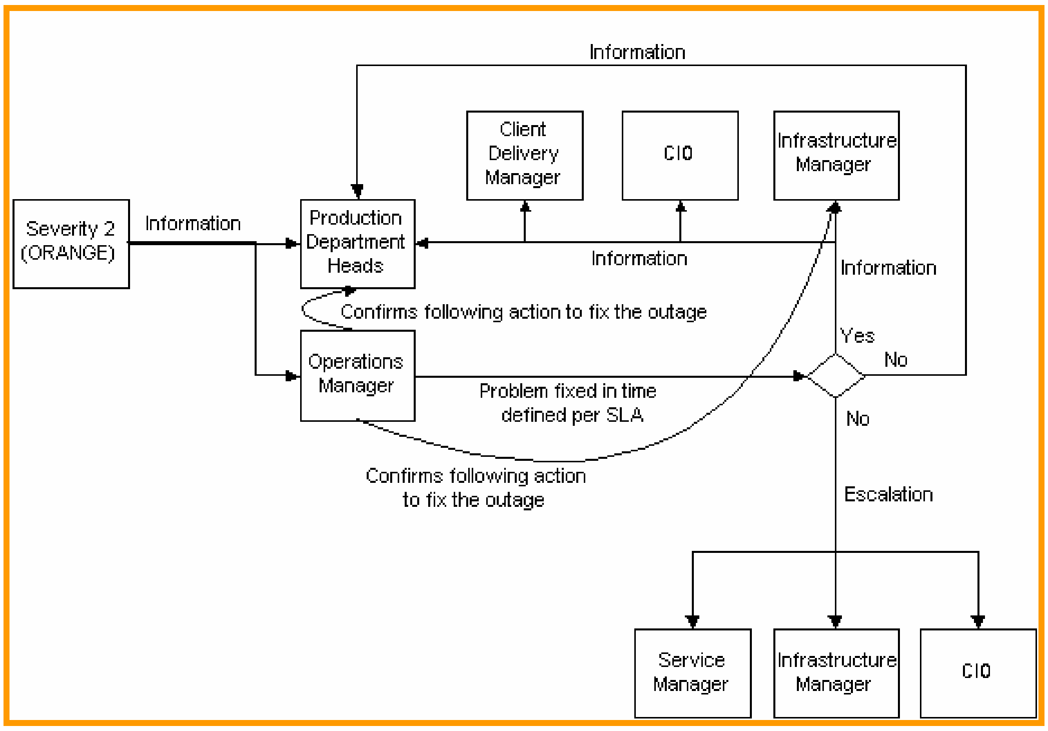


**Figure 8: Flowchart for Severity 2 (ORANGE)**

## Severity 3 (YELLOW)

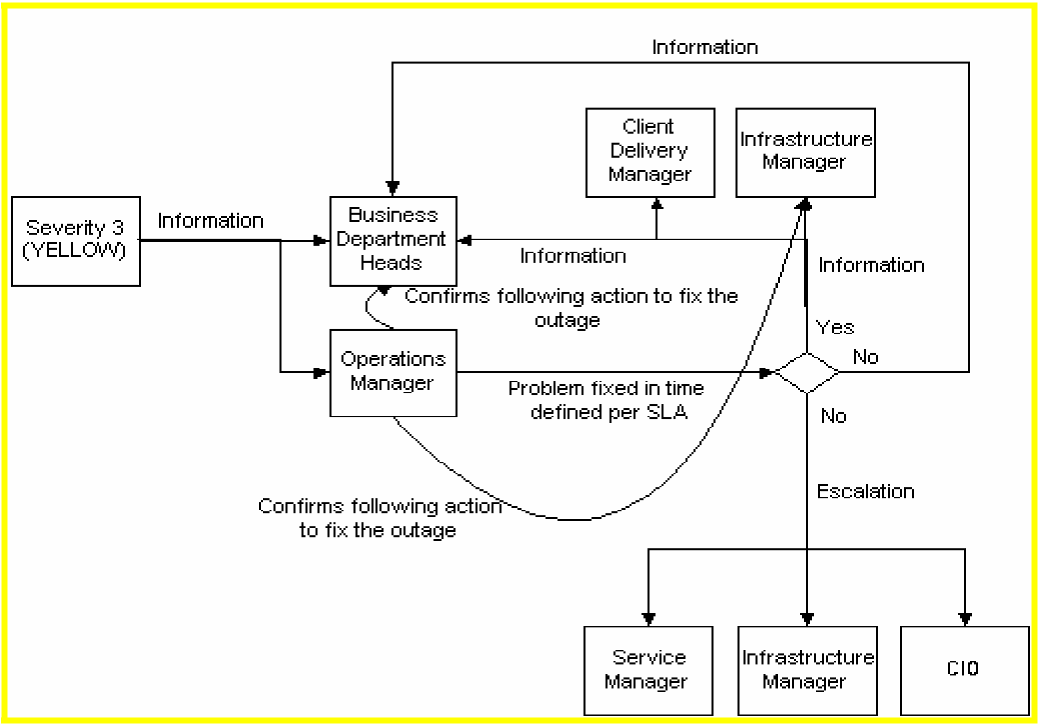


**Figure 9: Flowchart for Severity 3 (YELLOW)**

# 4 Requirements for a supporting IT system

To implement an IT system, which supports the above described escalation process first requirements need to be defined. Below those (high level) requirements are shown including the relationship between Business Requirements and User Requirements.

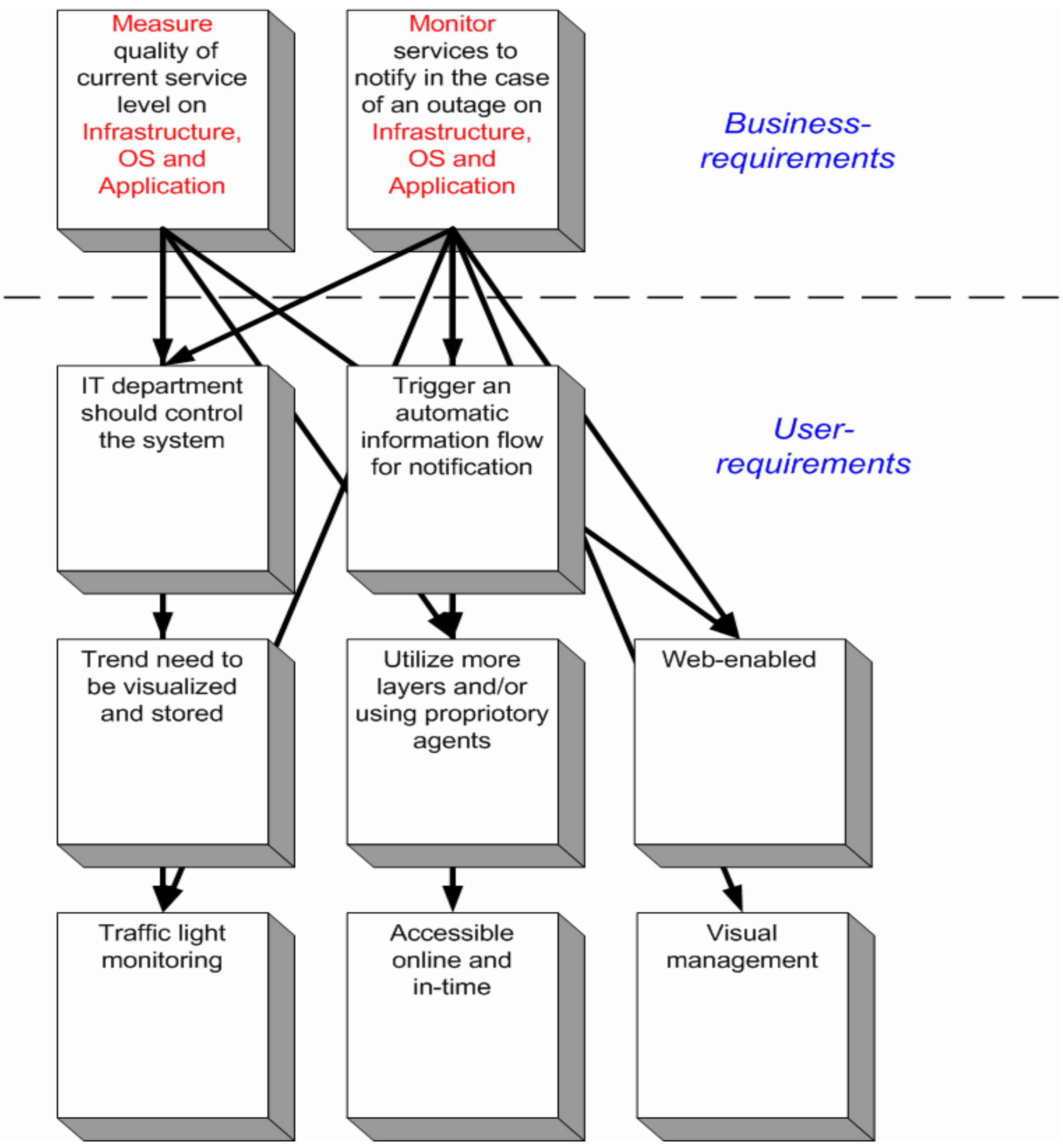

Measure quality of current service level on Infrastructure, OS and Application
Monitor services to notify in the case of an outage on Infrastructure, OS and Application
Business-requirements
IT department should control the system
Trigger an automatic information flow for notification
User-requirements
Trend need to be visualized and stored
Utilize more layers and/or using propriotory agents
Web-enabled
Traffic light monitoring
Accessible online and in-time
Visual management

**Figure 10: System Requirements**

Many tools on the market fulfil the above described requirements, however they all follow a three-tier standard architecture, outlined below. Some components (see figure below) have to be added to an e.g. standard architecture of a company, which operates from a headquarter two remote sites.

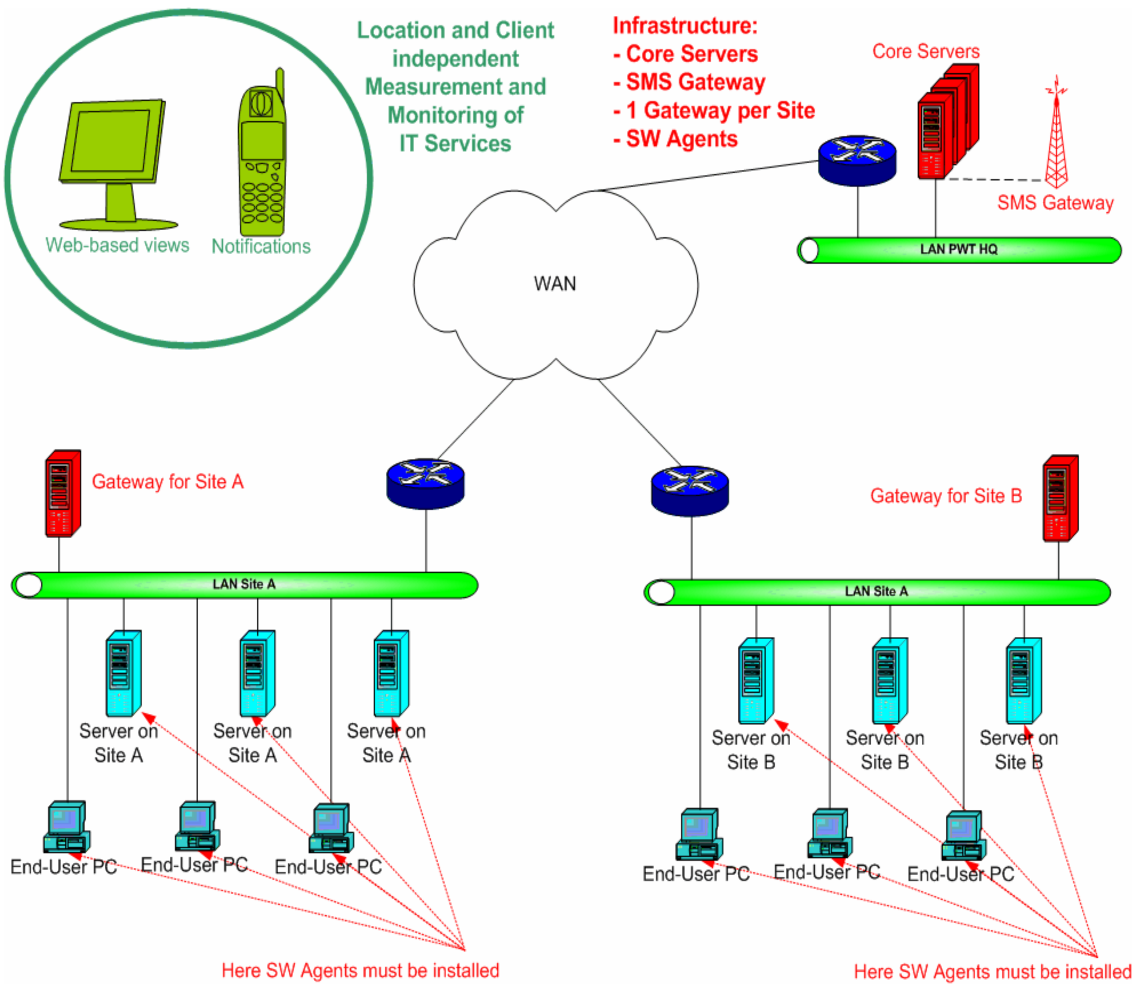


**Figure 11: Standard Architecture**

# 5 Enabling the Escalation Management Process

## 5.1 Generic Considerations for Local Area Networks

For enabling the business process described in chapter *Design of the Escalation Management Process* a short description on which levels network management is possible is given. Network management can be described by applying the layer model by ITU-T (International Telecommunications Union[24]). Here the business management layer, which is the interface to the applications which need the network, is based on the following sub-layers:

- Service Management Layer
- Network Management Layer
- Element Management Layer.

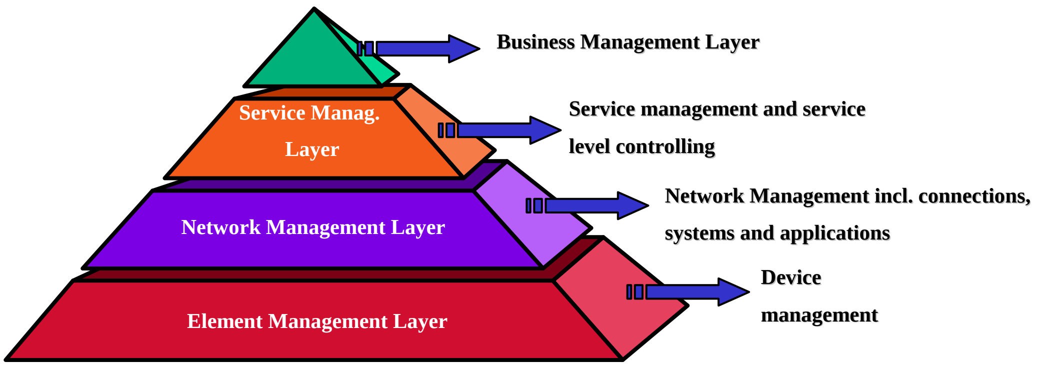

Business Management Layer
Service Manag. Layer
Service management and service level controlling
Network Management Layer
Network Management incl. connections, systems and applications
Element Management Layer
Device management

**Figure 12: Management model for local area networks by ITU-T**

In the basic network layer (*Element Management Layer*) the configuration and administration for the active components of the network is done. For controlling and monitoring the connections inside the network the *Network Management Layer* is used and for a good and comprehensive overview of the network status the *Service Management Layer* is utilized.

For all those different layers it is necessary to install IT tools. For detailed monitoring of the devices (*Element Management Layer*) the special monitoring tool from the vendor

[24] http://www.itu.int/home/

of the components must be installed, e.g. Cisco Works[25] for Cisco network components[26].

For the *Network Management Layer* tools which describe trends in time-series are used. Those tools help network managers visualize and understand the traffic on their networks (e.g. Cricket[27]).

To enable the escalation management the *Service Management Layer* can be utilized. The Open Source tool Nagios is able to notify both the management and the technical operations team early enough to set further actions. A service level agreement (SLA) can be defined because the two vital elements can be realized with Nagios[28]:

- Measurement of the current service level
- Defining thresholds

This thesis describes how to utilize Nagios for enabling the escalation management described in chapter *Design of the Escalation Management Process* regarding Local Area Networks (LAN) and servers.

## 5.2 Generic Considerations for Servers

Besides the LAN, servers are also critical for enabling electronic business processes. For monitoring servers let’s assume that the network protocol is TCP/IP. The TCP/IP suite of protocols provides a set of standards for how computers communicate and how networks are interconnected. The TCP/IP suite of protocols maps to a four-layer conceptual model: network interface, Internet, transport, and application.

<u>Network Interface Layer</u>

At the base of the model is the network interface layer. This layer puts frames on the wire and pulls frames off the wire.

[25] http://www.cisco.com/en/US/products/sw/cscowork/index.html

[26] http://www.cisco.com/en/US/netsol/index.html

[27] http://cricket.sourceforge.net/

[28] http://www.nagios.org/

Internet Layer

Internet layer protocols encapsulate packets in Internet datagrams and run all the necessary routing algorithms.

Transport Layer

Transport layer protocols provide communication sessions between computers. The desired method of data delivery determines the transport protocol.

Application Layer

At the top of the model is the application layer, in which applications gain access to the network. There are many standard TCP/IP utilities and services in the application layer, such as FTP, Telnet, Simple Network Management Protocol (SNMP), DNS, and so on.

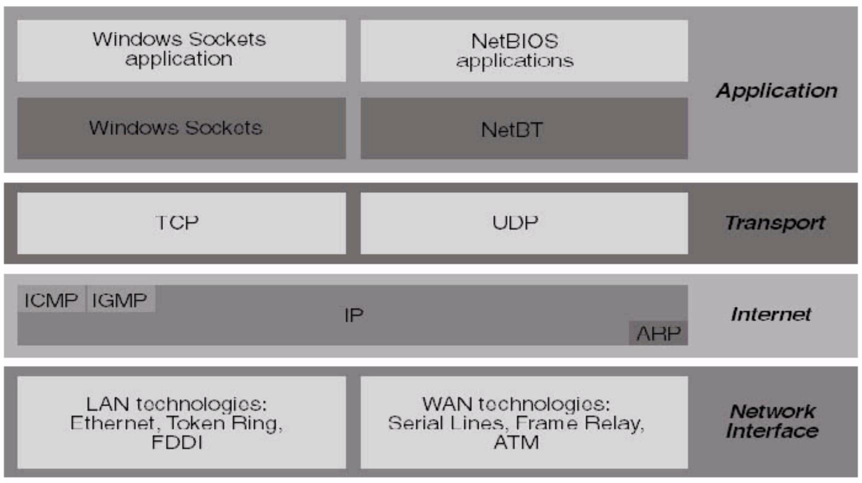

Windows Sockets application
NetBIOS applications
Application
Windows Sockets
NetBT
TCP
UDP
Transport
ICMP
IGMP
IP
ARP
Internet
LAN technologies: Ethernet, Token Ring, FDDI
WAN technologies: Serial Lines, Frame Relay, ATM
Network Interface

**Figure 13: TCP/IP suite of protocols**

For monitoring a server one needs to know on which layer the server should be “tested”. Only to check if the server is alive can be realized by a simple *ping* command, which is located in the Internet layer. But to really know if the server provides the service it should be the service itself that is investigated, e.g. a *NetBIOS* service on a Windows file server. The following table shows the main services in a typical business environment, running Lotus Notes as groupware.

| Application | Service |
|---|---|
| Windows, Domain Contr. | Netlogon |
| DHCP | DHCPServer |
| DNS | DNSServer |
| Windows various | NetBIOS |
| Windows Terminal Server | TermService |
| Windows Cluster Node | ClusSvc |
| Windows Print Server | Spooler |
| Microsoft SQL | MSSQLServer |
| Lotus Notes | notesdata |
| Oracle Database | OracleServiceOW |
| Generic | cpuload |
| Generic | useddiskspace |
| Generic | memoryusage |
| WebSite | W3SVC |
| FTP | MSFTPSVC |

**Figure 14: Main services in a typical business environment**

If one of those services exceeds a defined threshold Nagios can send notifications. Together with the LAN notifications the set of parameters is complete to give the technical operations team the right information early enough to set further actions.

## 5.3 Introduction to Nagios

Nagios is a host and service monitor designed to inform of IT problems (network and/or servers) before end-users or managers do. The monitoring services runs periodical checks on hosts and specified services and return status information to Nagios. When problems are encountered, Nagios can send notifications out to administrative contacts in a variety of different ways (email, instant message, SMS, etc.). Current status information, historical logs, and reports can all be accessed via a web browser.

The most important features of Nagios are:

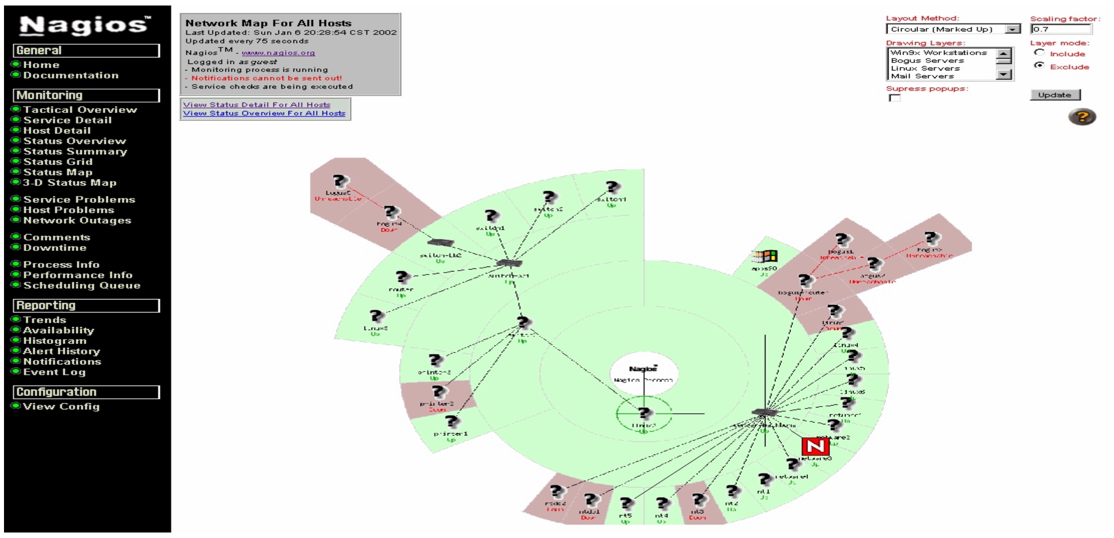
- Monitoring of network services e.g.,
  - SMTP
  - POP3
  - HTTP
- Monitoring of host resources e.g.,
  - Processor load
  - Disk and memory usage
- Monitoring of environmental factors such as temperature
- Ability to define network host hierarchy, allowing detection of and distinction between hosts that are down and those that are unreachable
- Conduct notifications when service or host problems occur and get resolved via
  - Email
  - SMS
  - Other notification methodology
- Ability to define event handlers to be run during service or host events for proactive problem resolution
- Support for implementing redundant and distributed monitoring servers
- Scheduled downtime for suppressing host and service notifications during periods of planned outages
- Web interface with simple authorization scheme

Nagios is licensed under the terms of the GNU General Public License[29] Version 2 as published by the Free Software Foundation[30]

[29] http://www.gnu.org/copyleft/gpl.html

[30] http://www.fsf.org/

The following screenshots give a feel for the performance:

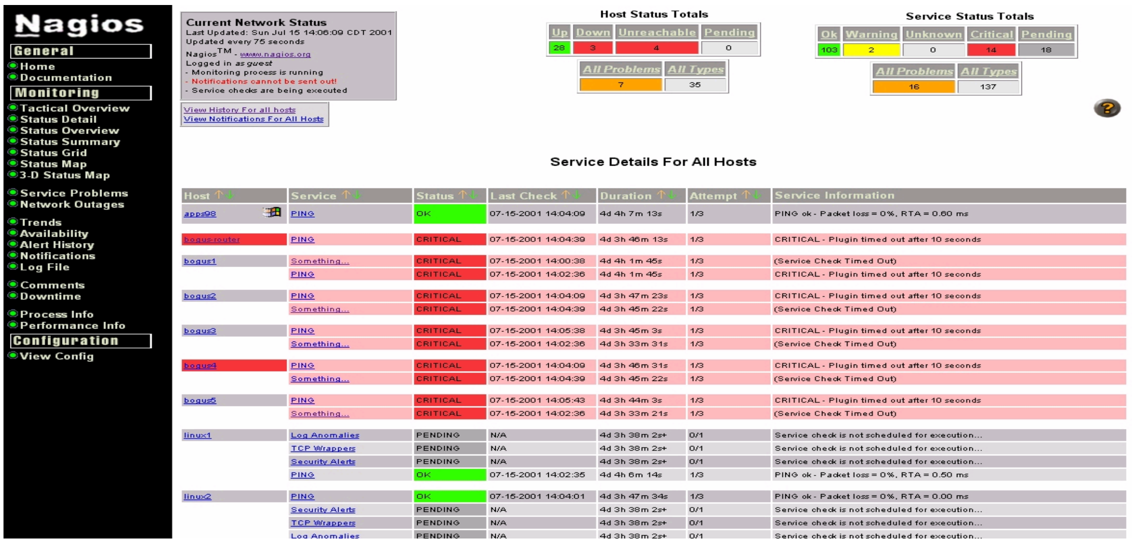


**Network Map For All Hosts**
Last Updated: Sun Jan 6 20:28:54 CST 2002
Updated every 75 seconds
Nagios™ - www.nagios.org
Logged in as *guest*
- Monitoring process is running
- Notifications cannot be sent out!
- Service checks are being executed

Layout Method: Circular (Marked Up)
Scaling factor: 0.7
Drawing Layers: Win9x Workstations, Bogus Servers, Linux Servers, Mail Servers
Layer mode: Include [✓ Exclude]
Supress popups:

**Figure 15: Architectural overview of devices including their status**

The figure above shows an overview of all devices and how they are connected to the Nagios server and to each other. Besides preconfigured views there is the possibility to customize the layout, so that one can get the same network-architecture picture one is used to. Besides this overview many other views exist. One gives you detailed information regarding each monitored host:



**Current Network Status**
Last Updated: Sun Jul 15 14:06:09 CDT 2001
Updated every 75 seconds
Nagios™ - www.nagios.org
Logged in as *guest*
- Monitoring process is running
- Notifications cannot be sent out!
- Service checks are being executed

Host Status Totals

| Up | Down | Unreachable | Pending |
|---|---|---|---|
| 28 | 3 | 4 | 0 |

| All Problems | All Types |
|---|---|
| 7 | 35 |

Service Status Totals

| Ok | Warning | Unknown | Critical | Pending |
|---|---|---|---|---|
| 103 | 2 | 0 | 14 | 18 |

| All Problems | All Types |
|---|---|
| 16 | 137 |

Service Details For All Hosts

| Host | Service | Status | Last Check | Duration | Attempt | Service Information |
|---|---|---|---|---|---|---|
| apps98 | PING | OK | 07-15-2001 14:04:09 | 4d 4h 7m 13s | 1/3 | PING ok - Packet loss = 0%, RTA = 0.60 ms |
| bogus-router | PING | CRITICAL | 07-15-2001 14:04:39 | 4d 3h 46m 13s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| bogus1 | Something... | CRITICAL | 07-15-2001 14:00:38 | 4d 4h 1m 45s | 1/3 | (Service Check Timed Out) |
| | PING | CRITICAL | 07-15-2001 14:02:36 | 4d 4h 1m 45s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| bogus2 | PING | CRITICAL | 07-15-2001 14:04:09 | 4d 3h 47m 23s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| | Something... | CRITICAL | 07-15-2001 14:04:39 | 4d 3h 45m 22s | 1/3 | (Service Check Timed Out) |
| bogus3 | PING | CRITICAL | 07-15-2001 14:05:38 | 4d 3h 45m 3s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| | Something... | CRITICAL | 07-15-2001 14:02:36 | 4d 3h 33m 31s | 1/3 | (Service Check Timed Out) |
| bogus4 | PING | CRITICAL | 07-15-2001 14:04:09 | 4d 3h 46m 31s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| | Something... | CRITICAL | 07-15-2001 14:04:39 | 4d 3h 45m 22s | 1/3 | (Service Check Timed Out) |
| bogus5 | PING | CRITICAL | 07-15-2001 14:05:43 | 4d 3h 44m 3s | 1/3 | CRITICAL - Plugin timed out after 10 seconds |
| | Something... | CRITICAL | 07-15-2001 14:02:36 | 4d 3h 33m 21s | 1/3 | (Service Check Timed Out) |
| linux1 | Log Anomalies | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |
| | TCP Wrappers | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |
| | Security Alerts | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |
| | PING | OK | 07-15-2001 14:02:35 | 4d 4h 6m 14s | 1/3 | PING ok - Packet loss = 0%, RTA = 0.50 ms |
| linux2 | PING | OK | 07-15-2001 14:04:01 | 4d 3h 47m 34s | 1/3 | PING ok - Packet loss = 0%, RTA = 0.00 ms |
| | Security Alerts | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |
| | TCP Wrappers | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |
| | Log Anomalies | PENDING | N/A | 4d 3h 38m 2s+ | 0/1 | Service check is not scheduled for execution... |

**Figure 16: Detailed view of all monitored hosts**

To get a feel how Nagios is working one can access the online demo[31] or download and install the software[32]. Articles regarding Nagios can be found in many magazines[33].

## 5.4 Definition and Quantifying the Parameters

For the implementation of Nagios for LAN and servers, parameters have to be defined, measured and compared with certain thresholds. A set of parameters always describes a Monitoring Quality, e.g. the status (up or down) of a certain switch. The specific parameters and thresholds for LAN and servers are discussed in the following chapters. Generic consideration for all parameters are described here:

For a certain Monitoring Quality Nagios has two threshold types:

- Warning
- Critical

If a certain parameter exceeds one of those threshold types a

- Soft State Alarm

occurs and is logged by Nagios. After *n* Soft State Alarms in sequence a

- Hard Alarm

is set, which leads to a notification via email and/or SMS. Of course also Hard Alarms are logged by Nagios.

For a certain Monitoring Quality the following table describes all information required by Nagios:

| Monitoring Quality (n) | Measured Param. 1 [unit] | Measured Param. 2 [unit] |
|---|---|---|
| Warning | Value 1 | Value 3 |
| Critical | Value 2 | Value 4 |

**Figure 17: Monitoring Quality in Nagios**

[31] http://www.nagios.org/demo.php

[32] http://www.nagios.org/download/

[33] http://www.nagios.org/propaganda.php

### 5.4.1 Generic: The Ping Command

There is one Monitoring Quality which is useful for both LAN components and servers. As already mentioned in chapter *Generic Considerations for Servers* the *ping* command can be used to check on the Internet layer if a server is alive. The same mechanism can be utilized to check if active components of the LAN are alive.

Of course switches do not need an IP address to act as switches, but in today's networks nearly all switches have an IP address for managing reasons. Routers - by definition - have IP addresses.

In summary the *ping* command can be used to check all active components (switches, routers) and servers.

For the Monitoring Quality *PING* the following parameters and thresholds are useful:

| Ping (n=3) | Round Trip Time [msec] | Packet Loss [%] |
|---|---|---|
| Warning | 100 | 20 |
| Critical | 500 | 60 |

**Figure 18: Monitoring Quality Ping**

The parameters are described below:

Round Trip Time:

The time interval between the moment a probe is sent and the moment a response is received.

Packet Loss:

Lost IP packets, where the TTL (Time to Life) counter is zero or the number of hopcounts is bigger than 30. This parameter is useful only in large networks.

### 5.4.2 Local Area Network

To get information from the active components (switches, routers) Nagios uses values read via SNMP messages from the built-in MIB[34]-tree. The MIB is a database which defines the information monitored by Nagios via SNMP.

The following Monitoring Qualities are useful for LANs and should be measured per each port on a switch:

| Collisions (n=3) | Value/time [n/sec in %] |
|---|---|
| Critical | 5 |
| Oversize (n=3) | Value/time [n/sec in %] |
| Critical | 2 |
| Undersize (n=3) | Value/time [n/sec in %] |
| Critical | 2 |
| FCS (n=3) | Value/time [n/sec in %] |
| Critical | 5 |
| CRC (n=3) | Value/time [n/sec in %] |
| Critical | 2 |
| Drops (n=3) | Value/time [n/sec in %] |
| Critical | 1 |

**Figure 19: Monitoring Qualities for Switches**

The parameters are described below:

<u>Collisions</u>

Ethernet networking uses collisions as one of the connections access methods. When the network carrier is not active, any station can send information. If two stations attempt to send information at the same time, the signals overlap with each other, creating a collision. Then nobody can send and after a random time the stations send again.

[34] Management Information Base

<u>Oversize (Giants)</u>

An Ethernet frame can not be bigger than 1514 bytes. Oversize frames are dropped.

<u>Undersize (Runts)</u>

An Ethernet frame can not be smaller than 64 byte. A smaller size indicates missing information like source address or protocol type. Undersized frames are also drops.

<u>Drops</u>

Number of rejected packets on the interface. A reason for a drop can be Giants or Runts.

<u>FCS (Frame check sequence)</u>

Indicates the completion of an Ethernet dialog.

<u>CRC (Cyclic redundancy checks)</u>

When a station sends a frame, it appends a Cyclical Redundancy Check to the end of the frame. The CRC is generated by an algorithm and is based on the data in the frame. If the frame is altered between the source and destination, the receiving station will recognize that the CRC does not match the actual contents of the packet.

Additionally the utilization of the CPU in the switches should be measured:

| CPU Load<br>Core switch (n=3) | [%] |
|---|---|
| Warning | 50 |
| Critical | 60 |
| CPU Load<br>Access switch (n=3) | [%] |
| Warning | 40 |
| Critical | 50 |

**Figure 20: Monitoring Qualities  for CPUs in switches**

### 5.4.3 Servers

As already mentioned in chapter *Generic Considerations for Servers* the most important thing to measure  on servers are services on the application layer. Of course Nagios also pings all servers but detailed information can only be gathered by investigating a defined service.

The following basic Monitoring Qualities are useful for servers and should be measured when applicable on a specific server (here some Monitoring Qualities for WAN are also included).

| HTTP / WAN (n=5) | Connection Time [sec] | Time to Load Site [sec] |
|---|---|---|
| Critical | 2 | 15 |
| HTTP / LAN (n=3) | Connection Time [sec] | Time to Load Site [sec] |
| Warning | - | 5 |
| Critical | 2 | 10 |
| Telnet / WAN (n=5) | Connection Time [sec] | |
| Warning | 4 | |
| Critical | 7 | |
| DHCP / LAN (n=3) | Time for receiving DHCPOFFER [sec] | |
| Critical | 10 | |
| NetBIOS / LAN (n=3) | Connection Time [sec] | |
| Critical | 10 | |
| CPU Load (n=3) | Utilization [%] | |
| Warning | 80 | |
| Critical | 95 | |
| Used Diskspace (n=3) | Utilization [%] | |
| Warning | 80 | |
| Critical | 90 | |
| Memory Usage (n=3) | Utilization [%] | |
| Warning | 80 | |
| Critical | 90 | |

**Figure 21: Basic Monitoring Qualities for servers**

Note that the following services need an agent (NSClient), which must be installed on the server:

- CPU Load
- Used Disk space
- Memory Usage

In addition to the basic Monitoring Qualities, specific services related to the type of the server should be monitored. Again the agent NSClient must be installed on those servers. It gives the following feedback to Nagios:

| Service | Running [Yes / No] |
|---|---|
| Critical | Value |

**Figure 22: Feedback to Nagios for specific services running on servers**

In chapter *Generic Considerations for Servers* the most important specific services are already mentioned, other services are 100% related to certain non off-the-shelf products and are not be mentioned here, because this would overstrain this thesis.

# 6 Bibliography

**Andreas Zink: Lokale Netze**
7. Auflage Addison Wesely, 2003

**Cisco: Ethernet**
http://www.cisco.com/en/US/products/hw/voiceapp/ps967/products_administration_guide_chapter09186a0080080bb0.html
October 20, 2003

**Eric Steven Raymond: The Cathedral and the Bazaar**
http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/
September 13, 2003

**Free Software Foundation**
http://www.fsf.org/
September 10, 2003

**GNU's Not Unix**
http://www.gnu.org
September 16, 2003

**GNU General Public Licenses**
http://www.gnu.org/copyleft/gpl.html
September 10, 2003

**HP Openview: Enterprise Management System**
http://www.openview.hp.com/
September 8, 2003

**HP: Open Source**
http://www.opensource.hp.com/
September 22, 2003

**IBM: Tivoli Enterprise Management System**
http://www-3.ibm.com/software/tivoli/
September 8, 2003

**John Deer: Network Management with Nagios**
Linux Journal, July 2003

**Martin Müller: Open Source**
O'Reilly Verlag, 1999

**Microsoft: Windows 2000 Core Requirements**
2nd edition, Microsoft Press, 2003

**Networkuptime: Ethernet**
http://www.networkuptime.com/faqs/ethernet/
October 20, 2003

**Open Source**
http://www.opensource.org/
September 16, 2003

# 7 Conclusio and Outlook

Due to the fact, that IT outsourcing increases more and more an IT supported Escalation Management process will become vital for corporations. Attendance and time are high valuable in today's and even more in tomorrows world and the fully automation of a very important business process can help companies to gain the competitive advantage they want and have to achieve.

In future Open Source will be established in corporate IT systems and help to deliver the required savings and stability; to start with Open Source today can help IT departments to be prepared for coming business challenges.

*Mag. Walter Sedlacek, MSc MBA*
*Opel Austria – General Motors Powertrain Europe*
*Information Systems & Services*
*Grossenzersdorferstrasse 59*
*A-1220 Vienna / Austria*
*Phone: + 43 1 28899 4605*
*Mobile: + 43 664 3423011*
*E-mail: walter.sedlacek@at.gm.com*