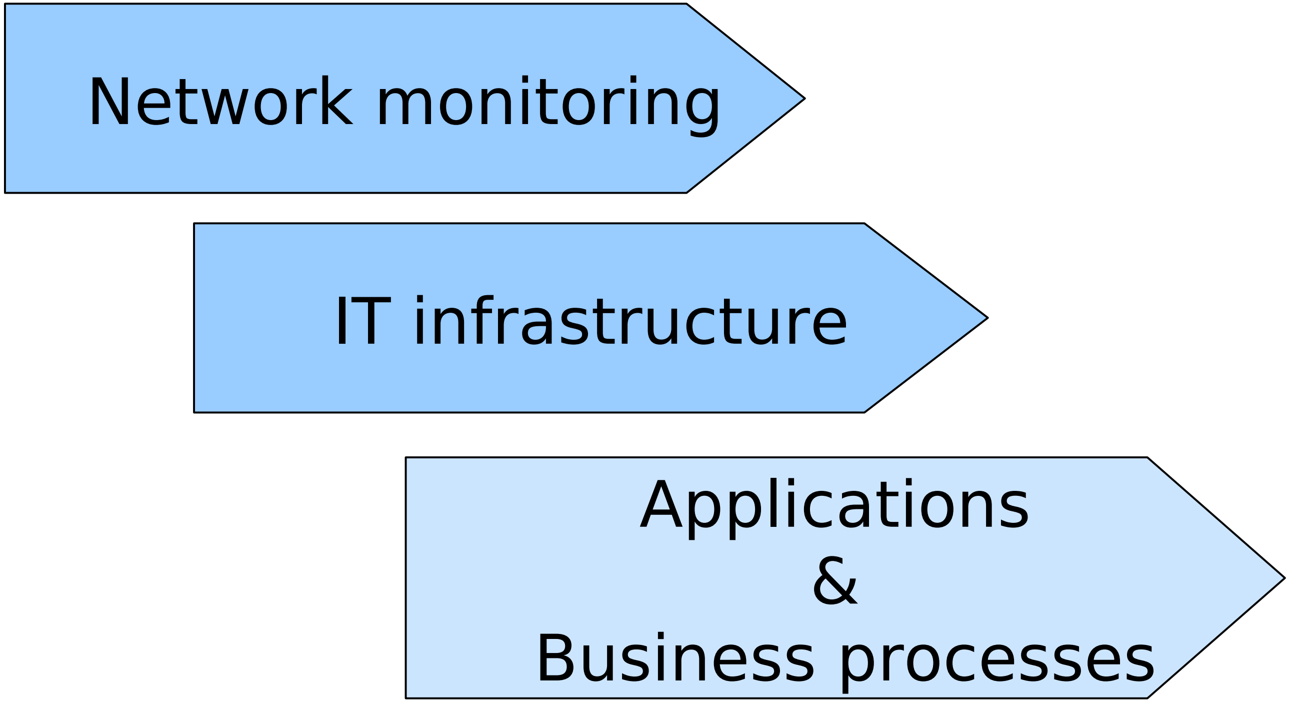
Business application monitoring - BAM

*The goals of business activity monitoring are to provide **real time information** about the status and results of various business related operations, processes, and transactions.*

*The main benefits of BAM are to enable an enterprise to make better **informed business decisions**, **quickly address problem** areas, and **re-position organizations** to take full advantage of emerging opportunities.*

Wikipedia

Next step in IT surveillance



Network monitoring

IT infrastructure

Applications
&
Business processes

Has nothing to do with IT

If the number of orders drop below my daily/weekly estimate

Warning if the delivery of goods is lower then 80% of ready to ship.

`If the ratio between web and phone orders are higher then ...`

If the number of errors in incoming EDI messages are higher then 5% of total

My route planning must be at least **90%** of my received orders

If the number of international shipment is above 10000 at 17:00
I need to give gateway a warning

What can BAM mean for you?

- Get real measurement of what is going on inside our applications from a business perspective
- Get out of the "IT-blindness"
- Understand application to application dependencies and effects of a process disturbance
- Define thresholds of business related events
- SLA by meaningful numbers
- Business trends in real time
- Notification to the business - not just IT
- IT resource utilization in business numbers

How can this be achieved?

- What to monitor and measure
  - Business processes of importance
  - Applications involved
  - Measurable entities
- Thresholds levels - alarms
- People/groups to notify
- Monitoring infrastructure
- Connecting to applications
  - "None intrusive" probing of the application events

What is different from normal monitoring?

- Business events are naturally dynamic
  - *"The number of orders are **different** depending of time of day and day of week"*
- Business events are dependent of other events
  - *"The number of invoices that should be created are **depending** on the number of received orders"*
- Capability relates to multiple events
  - *"The percentage between "web orders" and "all orders" should not be lower then 80%"*

Example 1 – Monitor the number of orders received during the day

The order management application receives order 24 hours a day during Monday to Friday.

The total aggregated number of orders is different depending on the time of the day. The business expects to have a total number of orders of 1500 at 13:00, at 14:00 it should be 2300, at 15:00 it should be 3400, etc. Between every hour we interpolate that the order rate is according to a linear equation. This means that the threshold at 13:20 is (2300-1500)*20/60+1500 = 1767.

The warning alarm level should be triggered between 90% and 70% of the threshold and a critical alarm should be triggered if the measured value is below 70% of the threshold.

Example 2 – Monitor the number of created invoices in relation to the number of received orders

The invoice system should invoice at least 80% of the daily incoming orders in the same day with one hour delay.

This means that the measured value of orders with one hour delay must be used as a threshold for the number of created invoices.

Example 3 – Monitor the current number of orders and if the inflow is zero, we need an alarm.

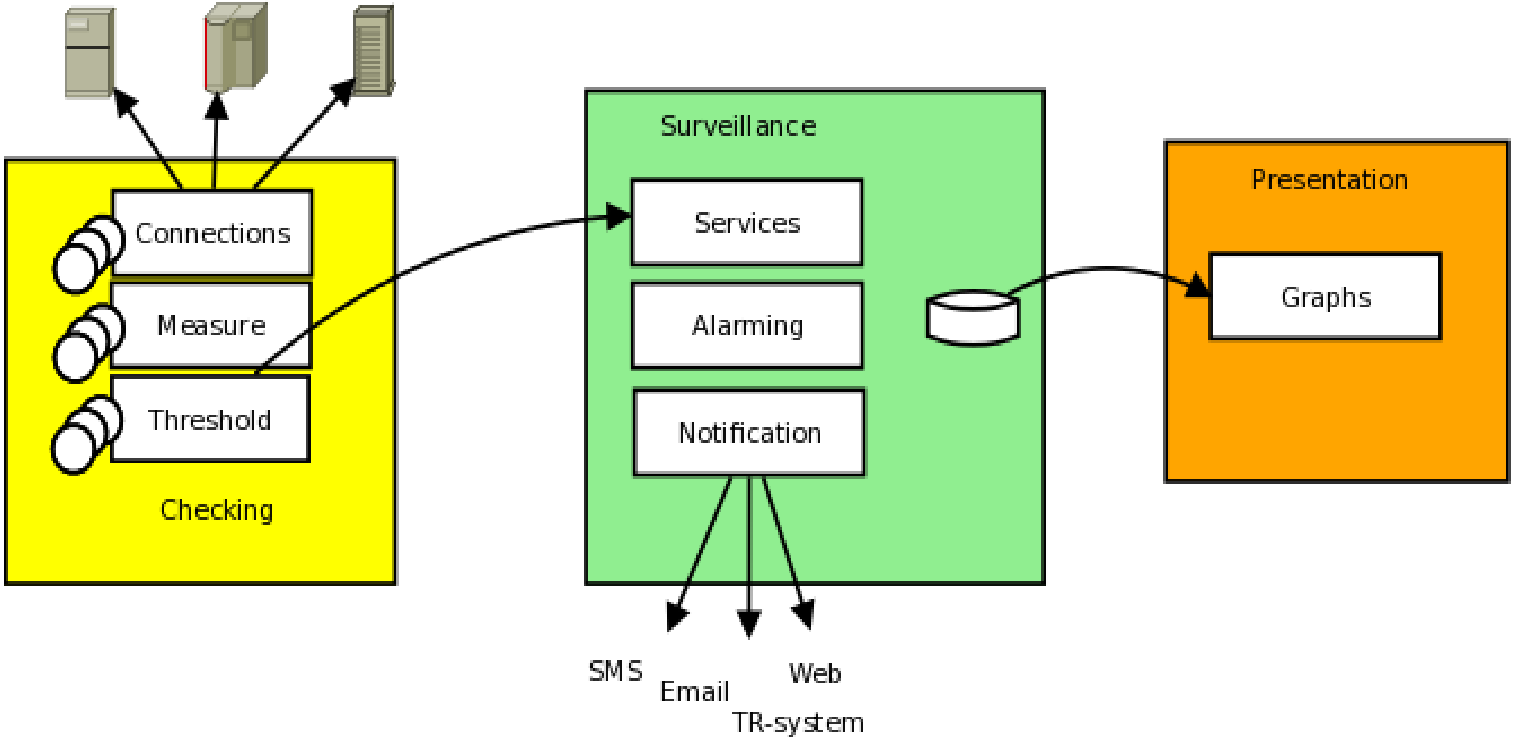
The order system has a table containing all received orders, but the requirement is that we need to monitor how many orders that has been received during the last 10 minutes.

If this value is zero an alarm must be generated since it is an indication that the sales system is not generating orders. To achieve this monitoring we use the last and the previous sample of the total number of orders from example 1 and create the difference between the two to get a new virtual entity to measure, with a threshold level of 0.

What is the same?

- Connectivity to applications and systems
- Definition of responsible groups
- Notification
- Escalation process
- Presentation of status and graphing
- Historical data
- .... all the features you are used to with NOC systems

BAM architecture



bischeck – a Nagios plugin

- An application surveillance and monitoring tool
- Focus on *application* and *business* related measuring and threshold management.
- Integrate with the Nagios plattform with passive checks over the Nagios integration protocol NSCA
- bischeck is licensed under GPL2
  - All other software used are open source licensed like GPL, Apache, etc
- Run as a linux service/deamon

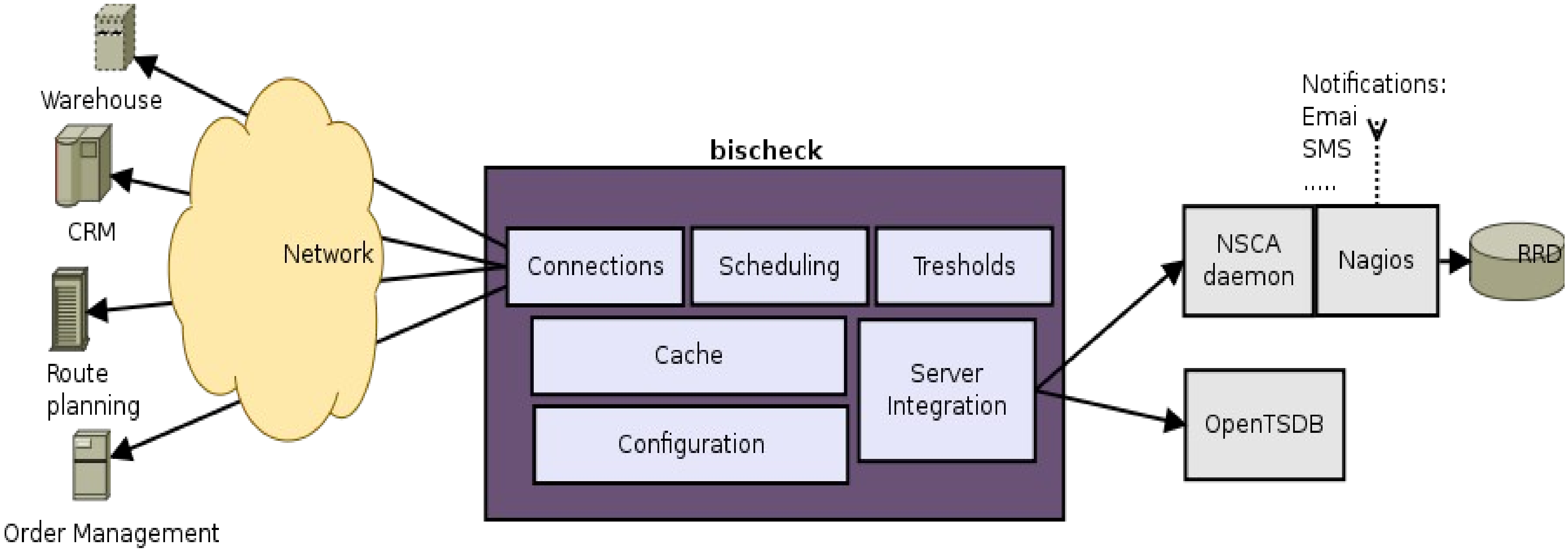
Who is using bischeck – as far as we know?

- DHL Freight Sweden
  - BAM for the whole forwarding and logistic process

Features

- **Different threshold** values depending of time of the day and day of month or week and holiday calender.
- **Dynamic threshold** based on measured values from other monitored services.
- **Multiple scheduling schemas** per service. This enables a fine grained control of when a service should be monitored.
- Configure services based on multiple measured entities, described as **"virtual" services**.
- **Caching** of historical data to provide the creation of virtual services and advanced thresholds.
- Date **macros** in execution statements of measured services, typically used in a “where clause” when selecting from databases.
- Multiple ways to connect to services to measure by allowing **custom service connection** methods.
- Support for **custom** threshold and connection classes.

Overview


Warehouse
CRM
Route planning
Order Management
Network
bischeck
Connections
Scheduling
Tresholds
Cache
Server Integration
Configuration
Notifications:
Emai
SMS
.....
NSCA daemon
Nagios
RRD
OpenTSDB

Host, service and service items

- Primary configuration objects
- Host
  - Name equivalent to Nagios Host configuration
- Service
  - Define a name, same as Nagios Service configuration
  - Connection method, e.g. jdbc
  - Schedule(s) - when to execute the service, cron style
- Service Item
  - Execution specification, e.g. SQL statement
  - Threshold specification, the threshold class to use

What is a threshold?

- Defines some logic that the measured value is compared against to define if an alarm should be triggered
- To the business, thresholds are often dynamic and/or the thresholds source(s) is some other service we measure
- Example
  - Measured value is 112 at time 13:45
  - Threshold configuration returns that the threshold value should be 140 at 13:45 for this service.
  - Threshold warning level is defined to be 10% of the threshold and critical level to be 30%
  - A warning state will be set

bischeck threshold

- Threshold is the definition for a service item that triggers an alarm. 3 different configurations are available:
    - Over the threshold
    - Under the threshold
    - Interval of the threshold
- Warning/Critical is set as a percentage of the threshold
- Threshold logic is implemented as a java class and configured for each service item
    - Threshold classes can easily be custom developed

bischeck cache

- Cache for all the historical data in an LRU structure
  - Number in cache is configurable
  - host-service-serviceitem[index] where index 0 is the latest measured
- Cache is used for “virtual” services and as threshold source
  - Virtual services and thresholds can be created based on mathematical expression on “real” measured services.
  - Example – to get an incremental value of a counter
    - host1-service1-serviceitem1[**0**] **–** host1-service1-serviceitem1[**1**]
  - Mathematical expressions are based on JEP open source version 2.3.1

bischeck architecture

- Java 6 based with internal JMX based surveillance
- Integration with nagios over nsca protocol for passive checks
- Parallel service execution through thread pool
- Run as a Linux service
- Can run on same server as nagios or a dedicated server
- Configuration topics are host, services, service items and thresholds
    - XML based configuration
    - Host and service name must be same as in nagios

Graphing

- Through pnp4nagios
- Bischeck has pnp4nagios templates to plot measured and threshold values

System requirements

- CentOS +5.5 or RedHat +5.5 server (Linux)
  - Support for other platforms following community demand
- Java 6 or later

Future road map – not final

- Other integration protocols than NSCA
  - E.g. OpenTSDB
- Cache distribution, cache persistency
- Installation support for other Linux distributions
  - Windows?