Why dynamic & adaptive thresholds matters

anders hååål, ingenjörsbyn ab
anders.haal@ingby.com
@thenodon

Bischeck -
dynamic & adaptive thresholds
for Nagios

www.bischeck.org

Threshold

What is the limitation with static threshold?

- ✗ Not static
- ✗ Load varies throughout the day, week
- ✗ To many or to few alarms
- ✗ Collecting and thresholding in the same context
- ✗ Based on the current measurement
- ✗ Do not consider dependency to other services

How to make thresholds dynamic & adaptive?

{example 1}

“Database table size should not be bigger then 5 % of yesterdays max size “

{example 1}

“Database table size should not be bigger then 5 % of max size yesterday“

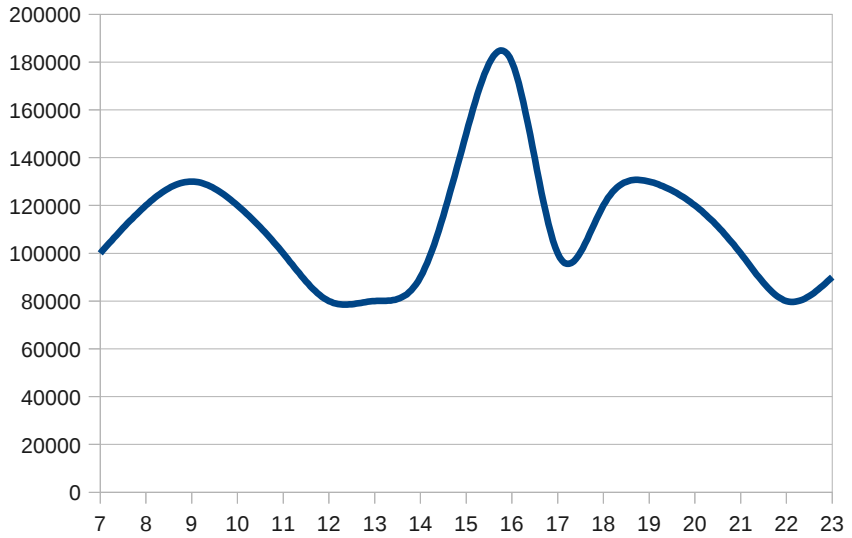


table size < max(yesterday)*1.05

Yesterday

Today

{example 2}

“Number of on-line users should not be more then 10 % higher then the average number of on-line users for the last 10 data points”

{example 2}

“Number of on-line users should not be more then 10 % higher then the average number of on-line users for the last 10 data points”

$$users < avg(X_0+X_1+.....+X_9)*1.1$$

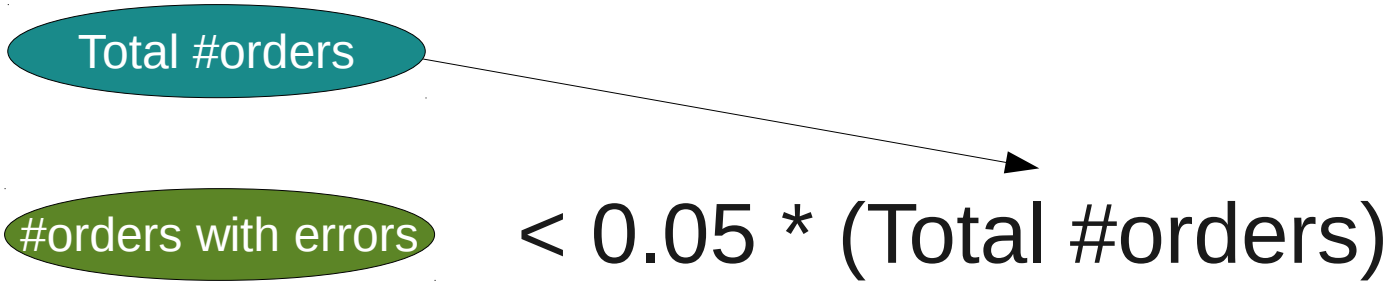
Where X is the historical on-line users data points

{example 3}

“The number of orders with errors should be lower then 5% of the total number of registered orders”

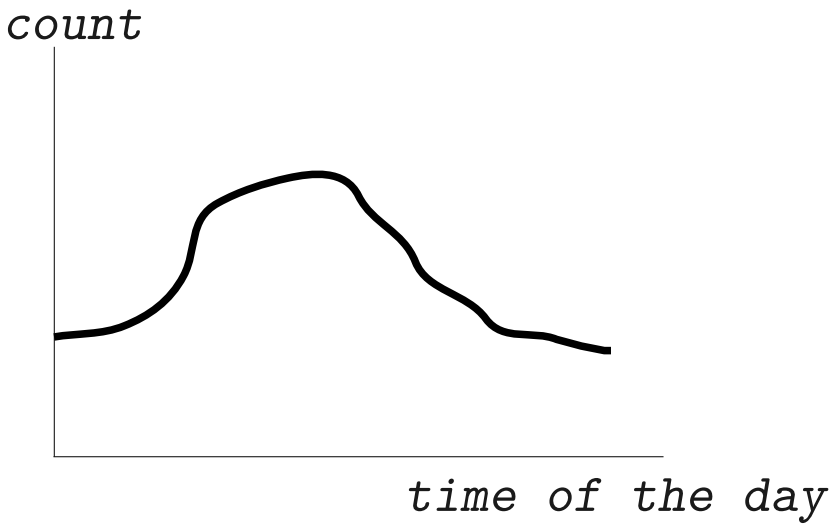
{example 3}

“The number of orders with errors should be lower then 5% of the total number of registered orders”



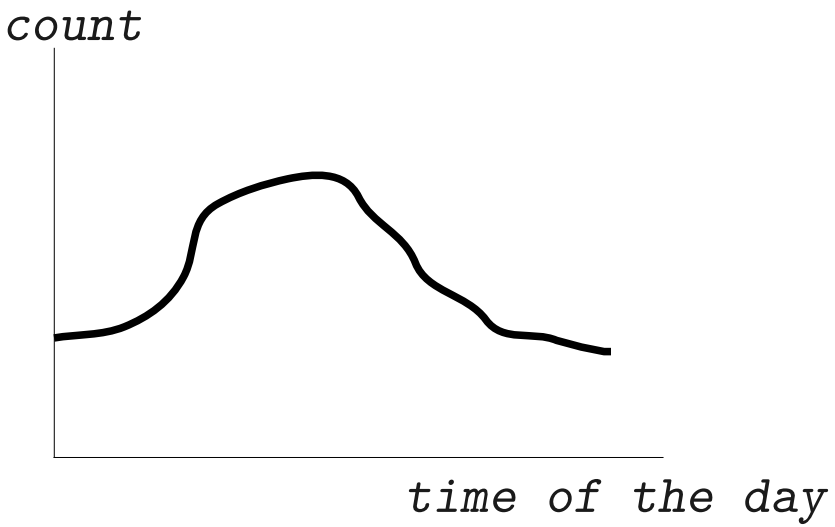
{example 4}

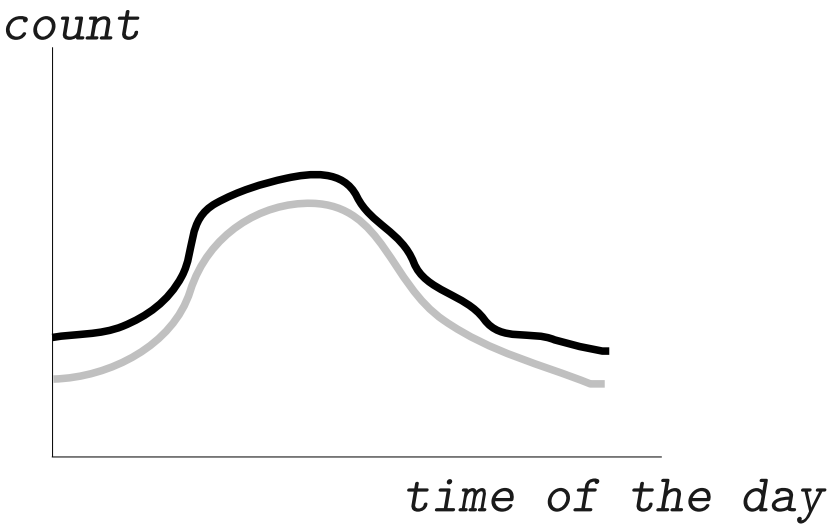
“Message queue size should be above the defined Friday threshold profile”



{example 4}

“Message queue size should be above the defined Friday threshold profile”



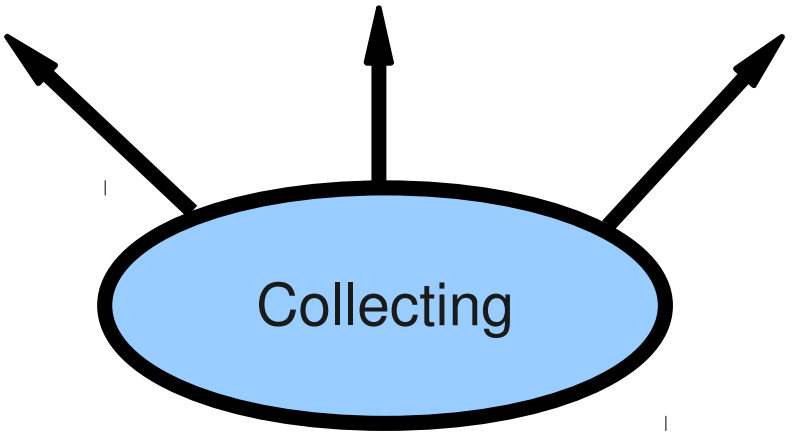


How to make thresholds dynamic & adaptive?

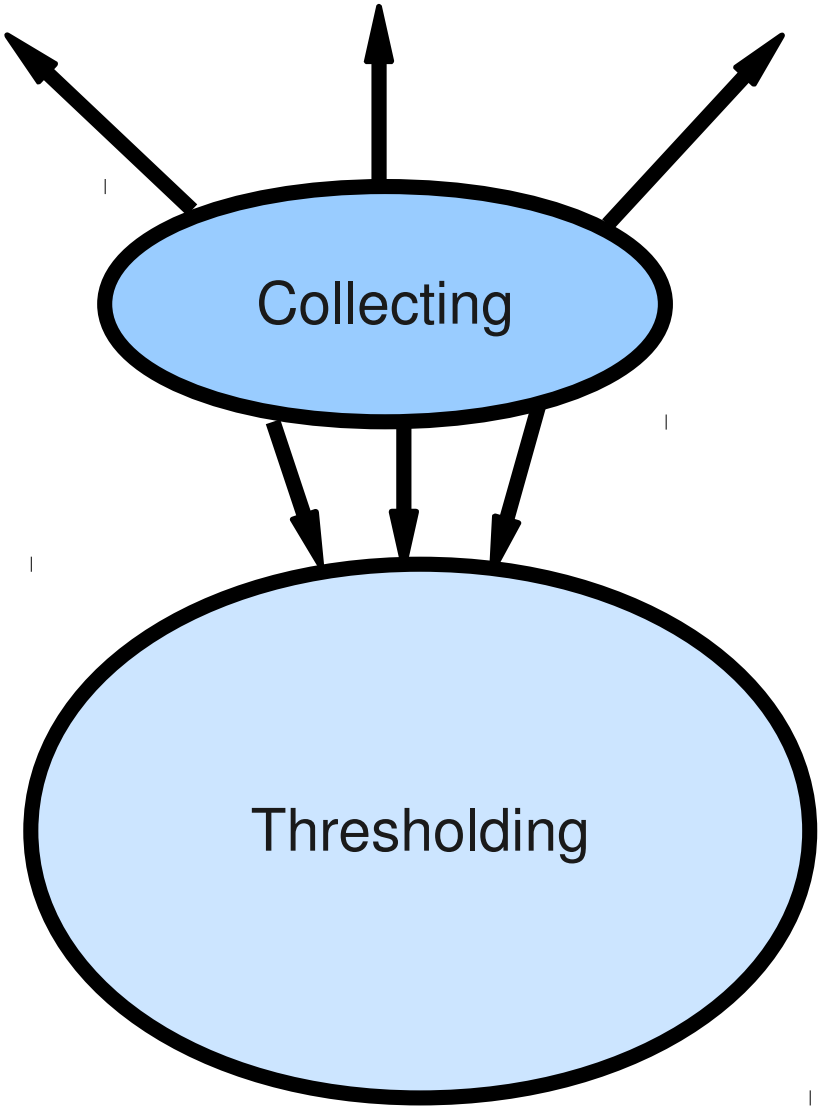
- Time profiles
- Historical data points
- Math and statistical operations

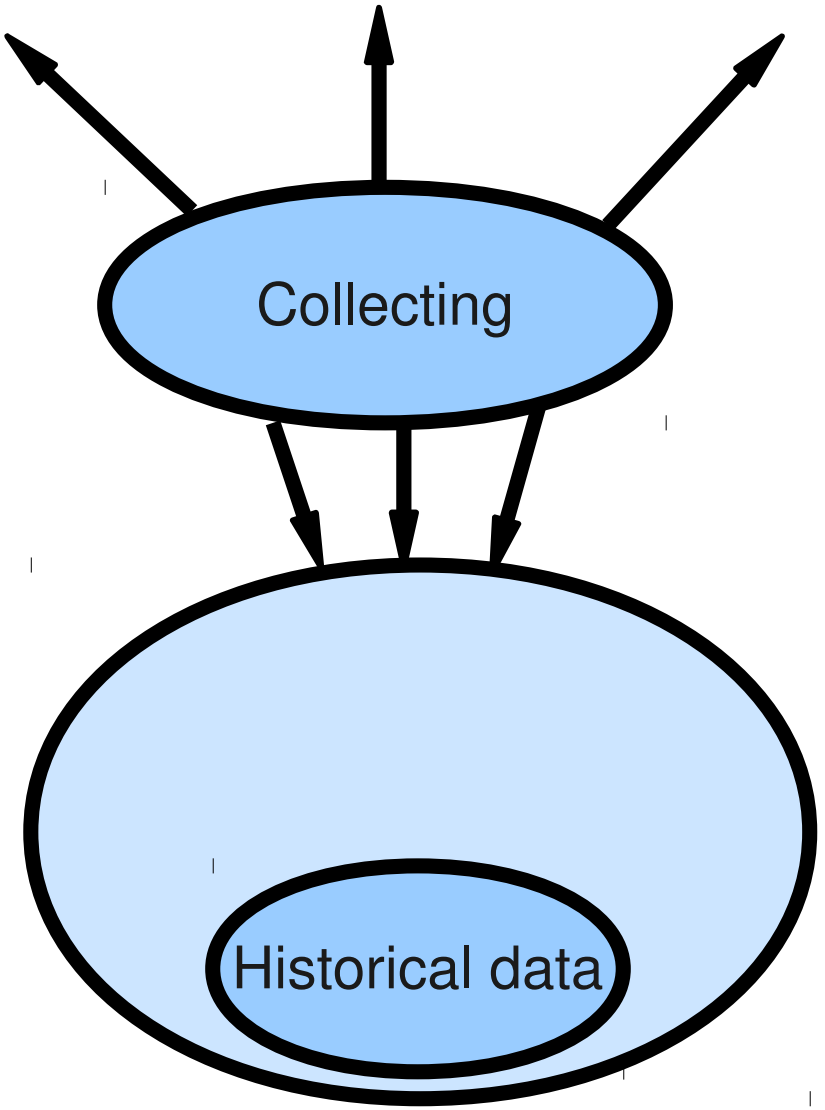
We did not want a check_XYZ hack

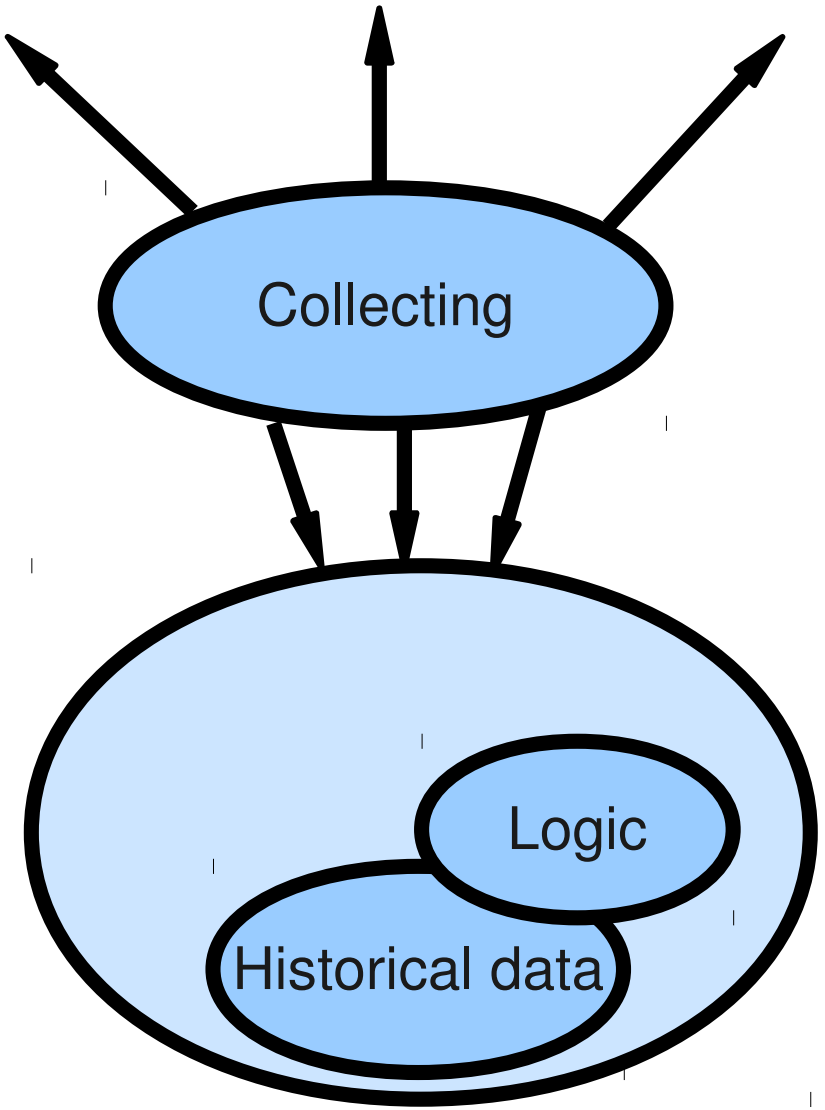
We wanted a tool

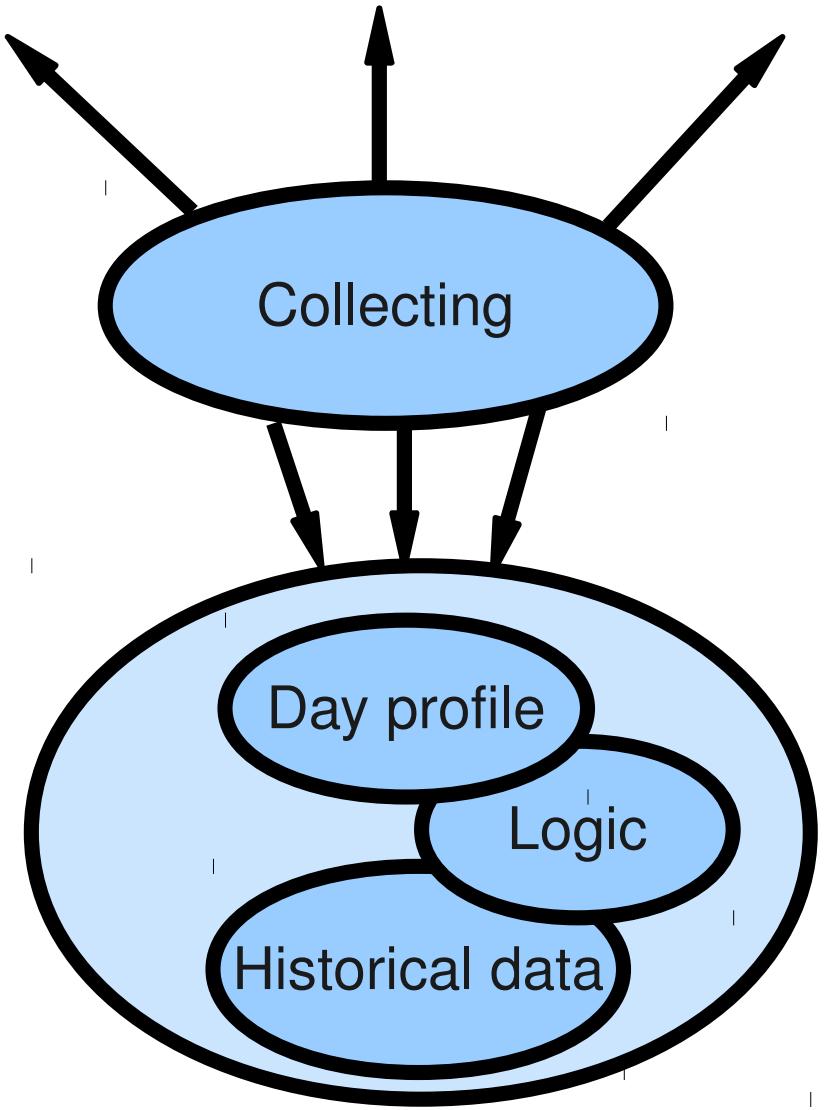

Collecting

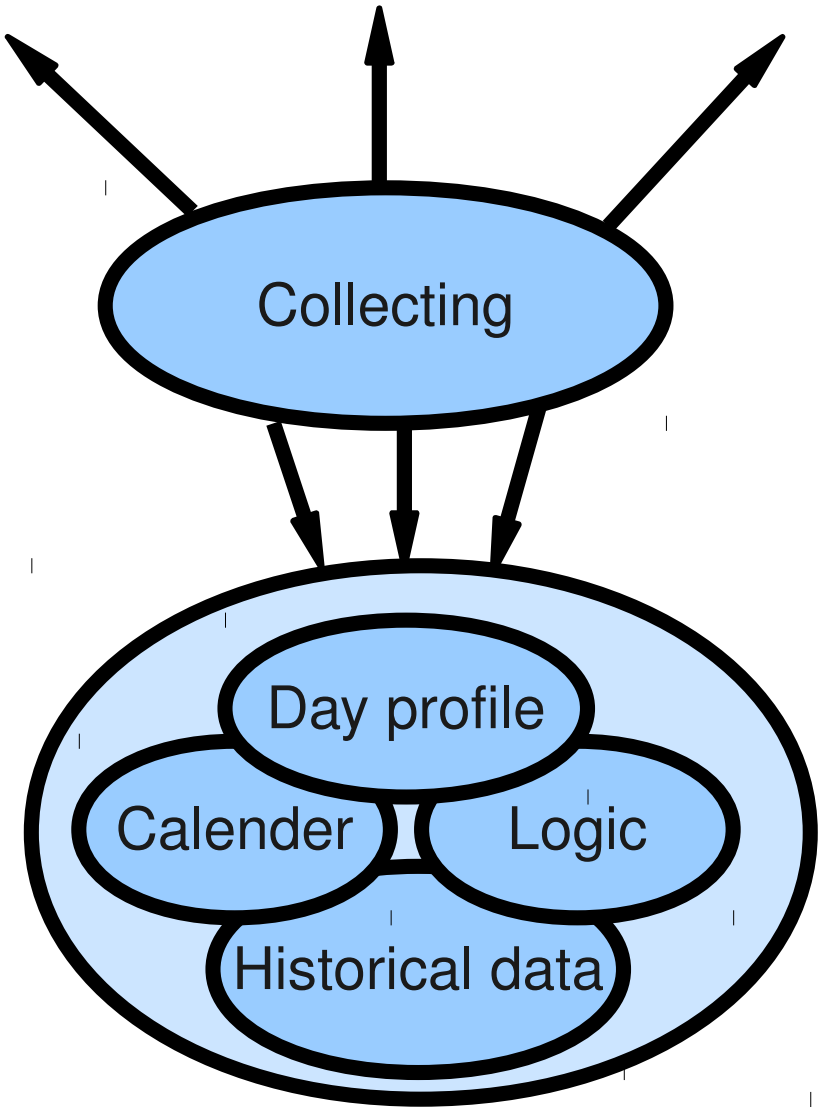
Separation

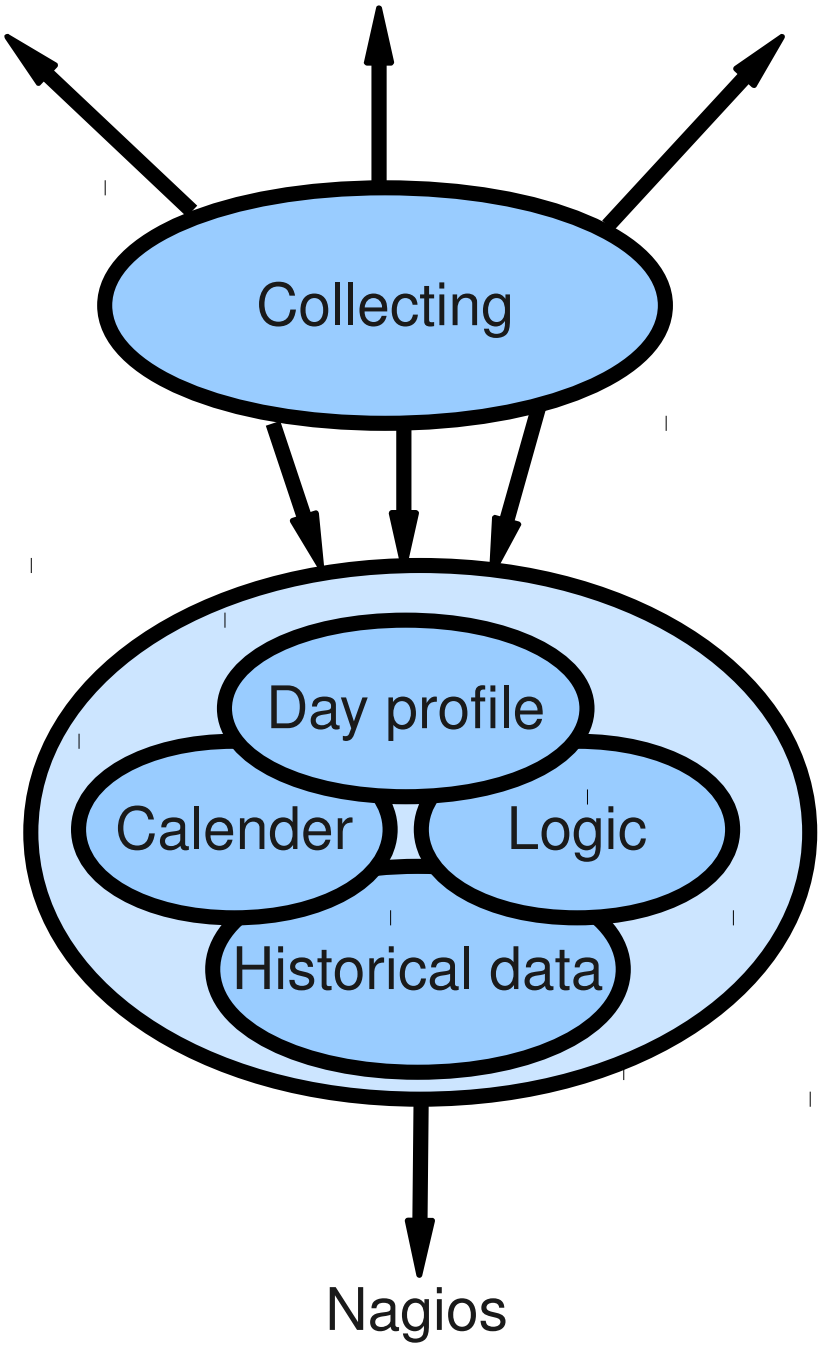


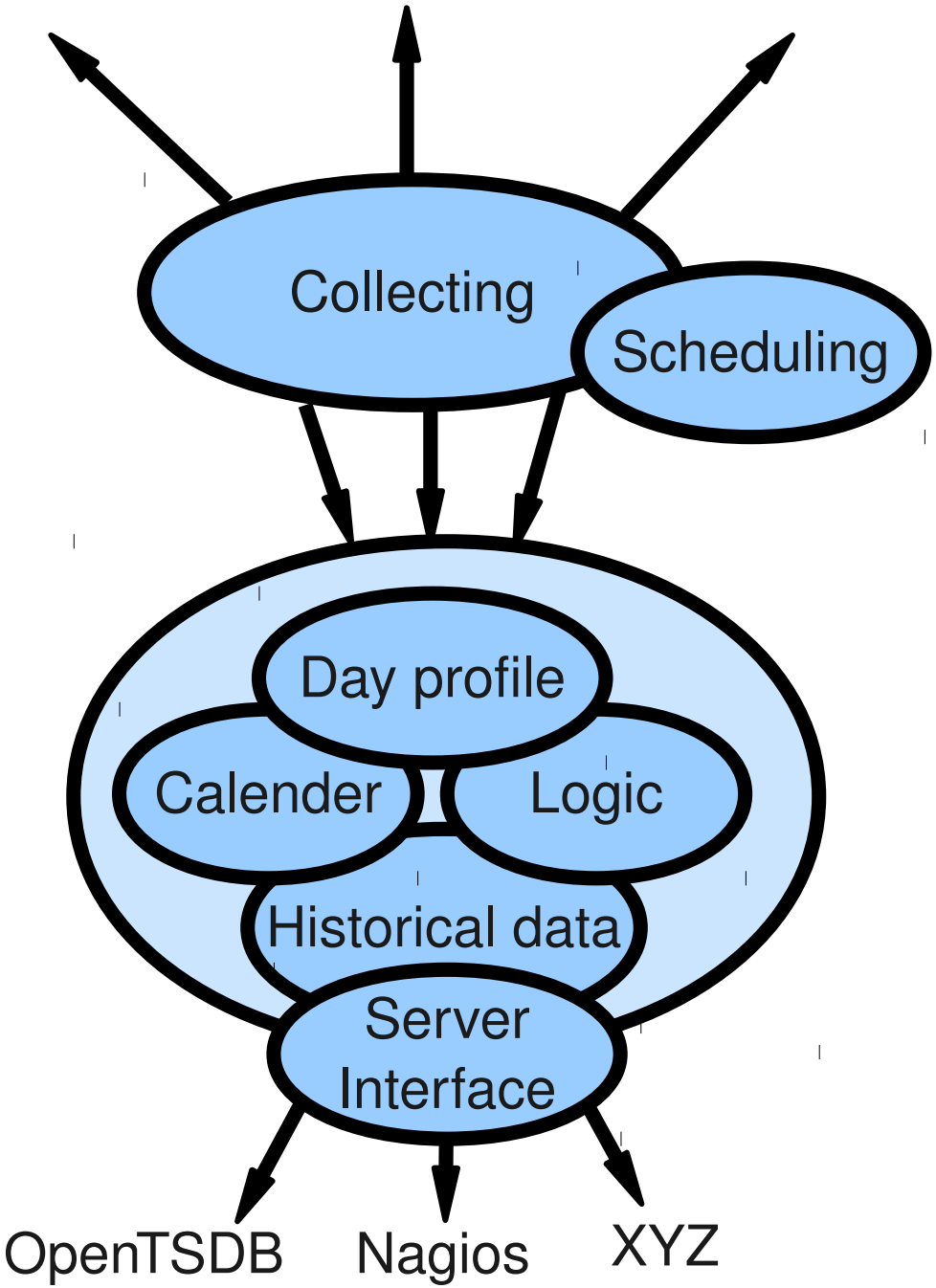

Collecting
Historical data

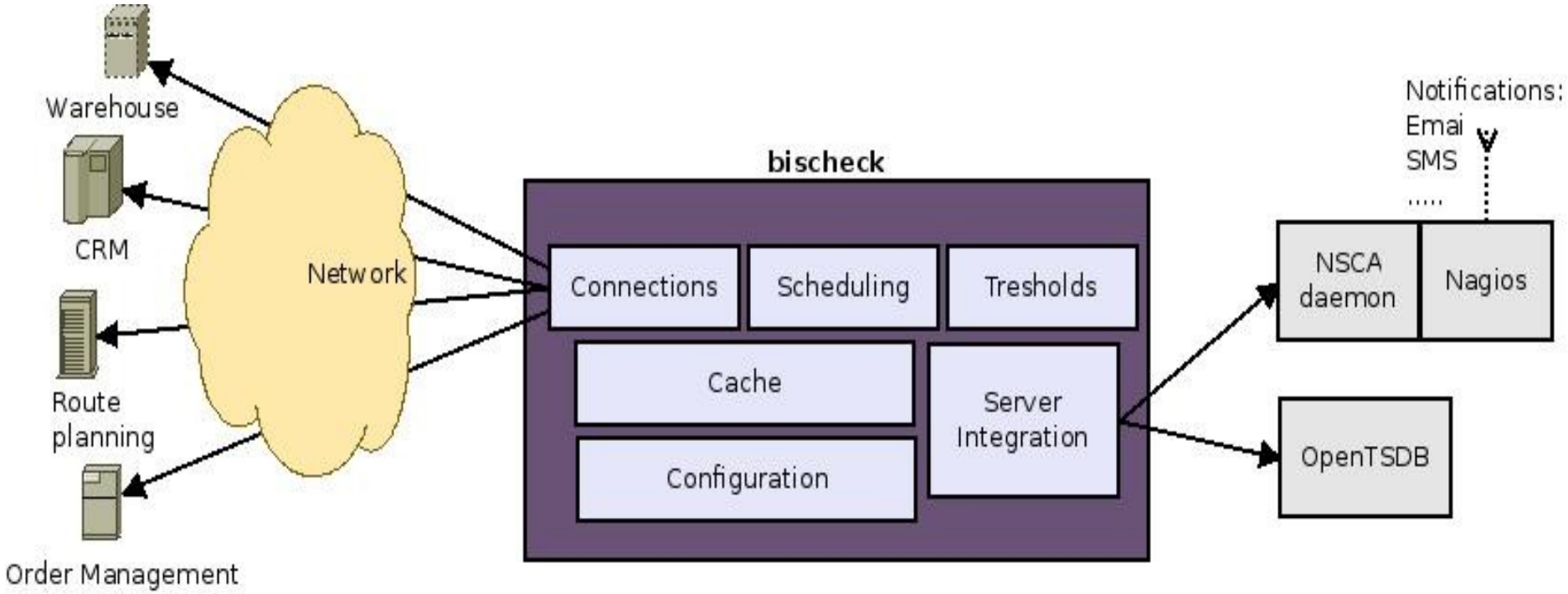

Collecting
Logic
Historical data

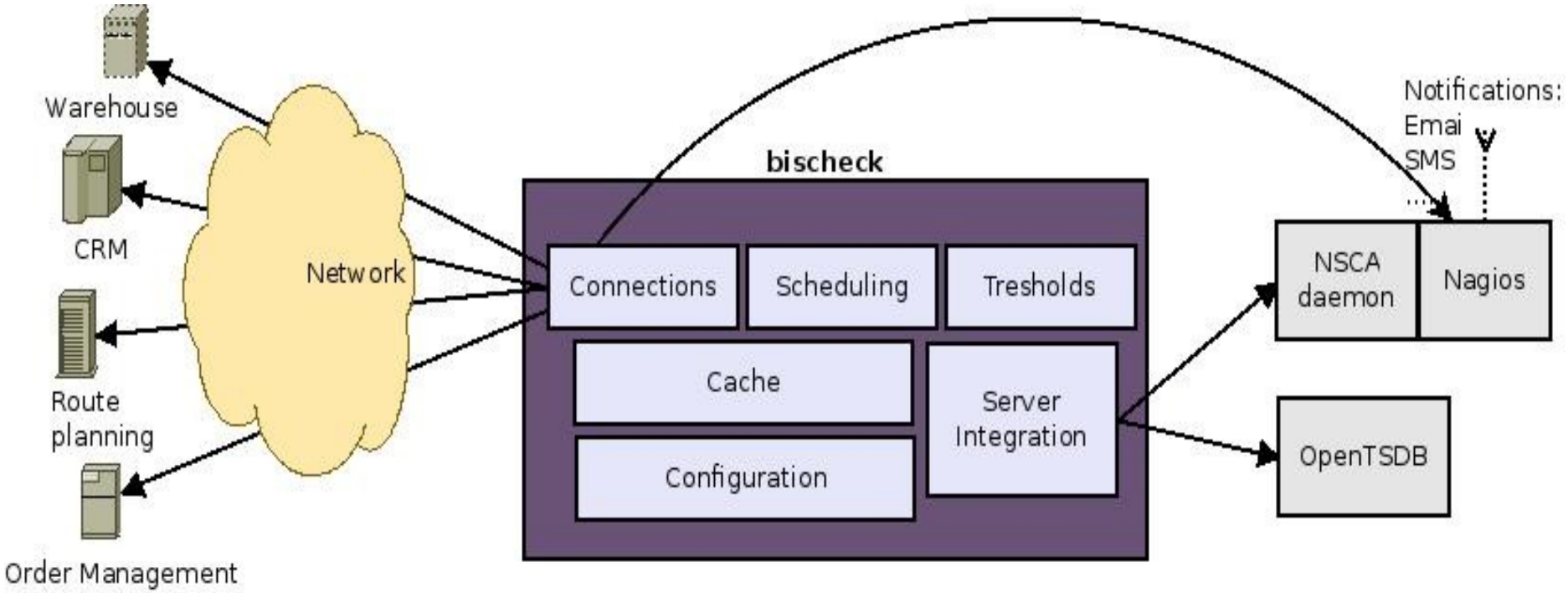

Collecting
Day profile
Logic
Historical data


Collecting
Day profile
Calender
Logic
Historical data


Collecting
Day profile
Calender
Logic
Historical data
Nagios

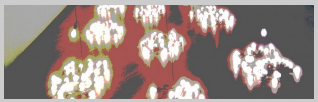

Collecting
Scheduling
Day profile
Calender
Logic
Historical data
Server
Interface
OpenTSDB
Nagios
XYZ


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


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

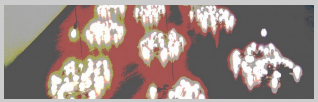
bischeck basics

- Configuration like Nagios – host, service but also service item
  - Host is just a container of the rest
  - Service specify the connection and scheduling
  - Service item specify the “query” and the threshold class to use
- Host and service name must be the same as in the Nagios configuration



Threshold – 24 hour day profile

- Divide the day in 24 hour points, where every point can be:
  - Static value
  - Dynamic value
    - Math expression on single value or range of data from the cache
    - Based on cached data points retrieved by
      - Index – single value or index range
      - Time – single value (closest) or time range (between)



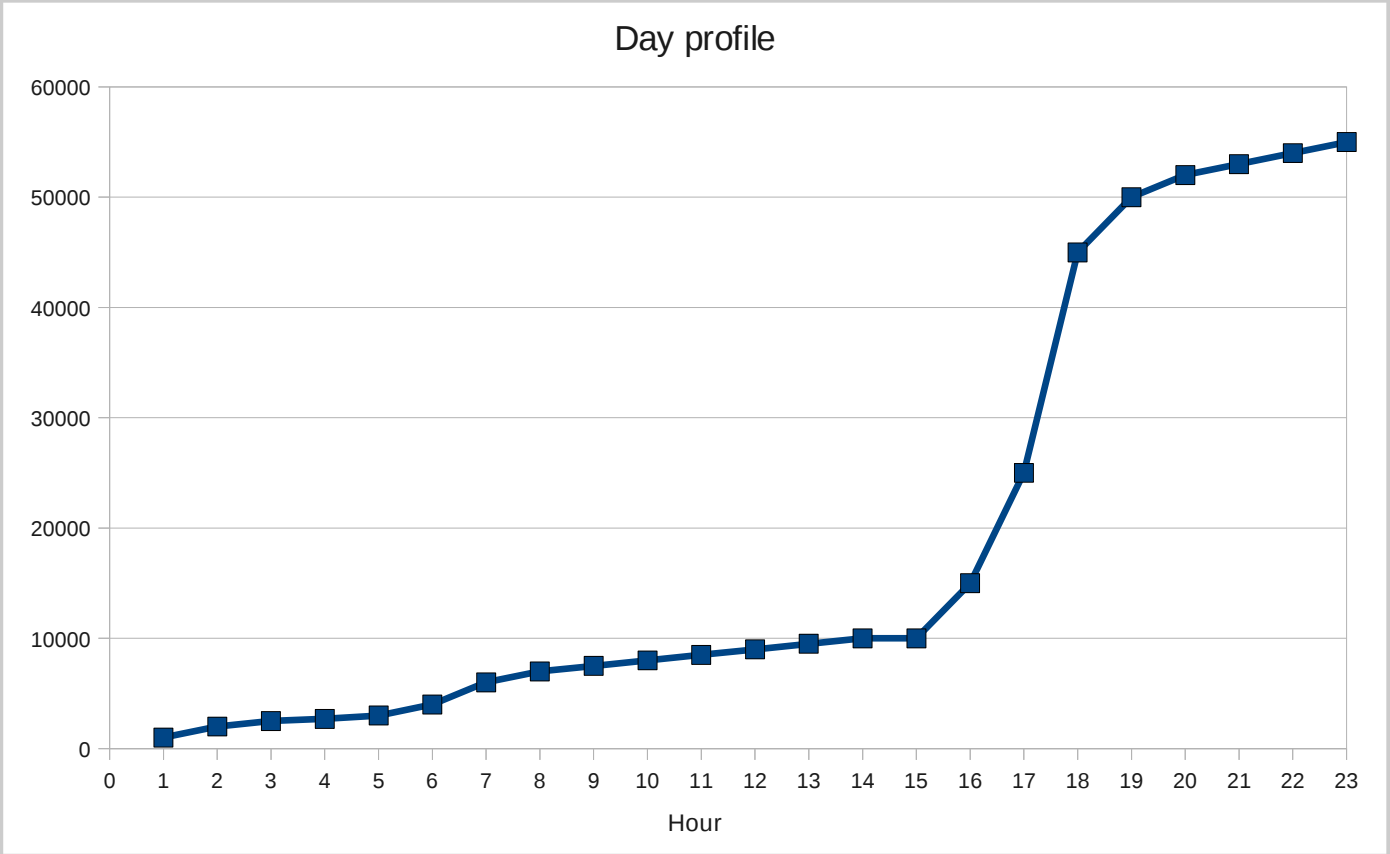
```
....
<!-- 12:00 Static -->
<hour>7000</hour>
....
```

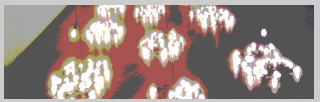
```
....
<!-- 12:00 Static -->
<hour>7000</hour>

<!-- 13:00 Adaptive -->
<hour>erpserver-orders-ediOrders[0] / 3</hour>
....
```

```
....
<!-- 12:00 Static -->
<hour>7000</hour>

<!-- 13:00 Adaptive -->
<hour>erpserver-orders-ediOrders[0] / 3</hour>

<!-- 14:00 Adaptive with math function -->
<hour>avg(erpserver-orders-ediOrders[-30M:-60M]) / 2</hour>
....
```

Threshold – 24 hour day profile

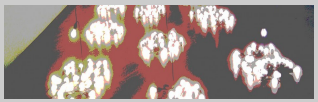
Between every “full” hour a linear equation is calculated





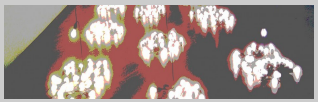
Threshold – 24 hour day profile

- Connect calender to the day profile and evaluate according to the following order:
  1. Month and day of month
  2. Week and day of week
  3. Day in month
  4. Day in the week
  5. Month
  6. Week
- Holiday – exception days



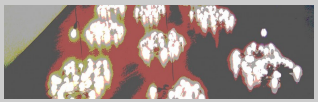
And more....

- Multi-threaded and multi-scheduling schema per service
  - interval
  - cron
- Data collection – jdbc, livestatus, internal cache
- Virtual services
- Date macros in execution statements
- Customize
  - connection (service classes)
  - execution (service item classes)
  - thresholds (threshold classes)
  - server integration (server classes)
- XML configuration supported with WEBui (beta)
- GPL 2 license



Future

- Improved time series database
- Patterns/baselines
- More statistic functions
- “Sensors” - alarms on multiple/aggregated data points
- Any ideas?



Infrastructure monitoring

Application performance monitoring [APM]

Business activity monitoring [BAM]

Operational Business intelligence [OBI]

Questions & Feedback

Pictures – Creative Commons

www.flickr.com/photos/loneprimate/4017405677
www.flickr.com/photos/catatronic/2397319483
www.flickr.com/photos/dtrimarchi/6815004766
www.flickr.com/photos/bikeracer/6740232