DevOps techniques and tools

. . . with an overview of the cultural aspects

M. Finelli
BioDec

BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

Cloud + Virtualization = Distributed Infrastructure

THANKS TO THE CLOUD AND TO THE VIRTUALIZATION technologies every company will need tools and techniques to deal with the complexities of a distributed infrastructure.

The **timeframe** is the next *few* years, not decades.

The **alternative** is *giving up* controlling or *giving up* the advantages of those technologies.

Cloud + Virtualization = Distributed Infrastructure

THANKS TO THE CLOUD AND TO THE VIRTUALIZATION technologies every company will need tools and techniques to deal with the complexities of a distributed infrastructure.

The **timeframe** is the next *few* years, not decades.

The **alternative** is *giving up* controlling or *giving up* the advantages of those technologies.

BIODEC
Evolving ICT Infrastructures

Cloud + Virtualization = Distributed Infrastructure

THANKS TO THE CLOUD AND TO THE VIRTUALIZATION technologies every company will need tools and techniques to deal with the complexities of a distributed infrastructure.
The **timeframe** is the next *few* years, not decades.
The **alternative** is *giving up* controlling or *giving up* the advantages of those technologies.

A *large* distributed infrastructure

Your infrastructure will be not only *distributed*, but also **large**: a very different scenario from that of traditional (yesterday's, perhaps today's if you are a laggard) infrastructures.

Different rules, different problems and different solutions.

Hint: adapting to the **large** what happens in the **small** is not a good strategy . . .

A *large* distributed infrastructure

Your infrastructure will be not only *distributed*, but also **large**: a very different scenario from that of traditional (yesterday's, perhaps today's if you are a laggard) infrastructures.
Different rules, different problems and different solutions.
Hint: adapting to the **large** what happens in the **small** is not a good strategy . . .

BIODEC
Evolving ICT Infrastructures

A *large* distributed infrastructure

Your infrastructure will be not only *distributed*, but also **large**: a very different scenario from that of traditional (yesterday's, perhaps today's if you are a laggard) infrastructures.
Different rules, different problems and different solutions.
Hint: adapting to the **large** what happens in the **small** is not a good strategy . . .

Index

BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

A timeline*

At the very beginning . . . Patrick Debois, year 2007. He is doing a job that requires *hybrid* skills: both programmer and sysadmin.

Agile 2008 Andrew Shafer proposes a session on *"Agile Infrastructure"*, but exactly zero people show up.

June 2009 John Allspaw gives the talk *"10+ deploys per day: Dev & Ops cooperation at Flickr"*.

30-31 October 2009 The very first *DevOps Days* in Gent, Belgium.

*from a presentation by Damon Edwards on IT Revolutions.

A timeline*

At the very beginning . . . Patrick Debois, year 2007. He is doing a job that requires *hybrid* skills: both programmer and sysadmin.

Agile 2008 Andrew Shafer proposes a session on *"Agile Infrastructure"*, but exactly zero people show up.

June 2009 John Allspaw gives the talk *"10+ deploys per day: Dev & Ops cooperation at Flickr"*.

30-31 October 2009 The very first *DevOps Days* in Gent, Belgium.

*from a presentation by Damon Edwards on IT Revolutions.

A timeline*

At the very beginning . . . Patrick Debois, year 2007. He is doing a job that requires *hybrid* skills: both programmer and sysadmin.

Agile 2008 Andrew Shafer proposes a session on *"Agile Infrastructure"*, but exactly zero people show up.

June 2009 John Allspaw gives the talk *"10+ deploys per day: Dev & Ops cooperation at Flickr"*.

30-31 October 2009 The very first *DevOps Days* in Gent, Belgium.

*from a presentation by Damon Edwards on IT Revolutions.

A timeline*

At the very beginning . . . Patrick Debois, year 2007. He is doing a job that requires *hybrid* skills: both programmer and sysadmin.

Agile 2008 Andrew Shafer proposes a session on *"Agile Infrastructure"*, but exactly zero people show up.

June 2009 John Allspaw gives the talk *"10+ deploys per day: Dev & Ops cooperation at Flickr"*.

30-31 October 2009 The very first *DevOps Days* in Gent, Belgium.

*from a presentation by Damon Edwards on IT Revolutions.

The early days

THE HASHTAG #DEVOPS is adopted. The *devops* topic — even if still undefined — gets debated in conferences and a new brand of community-driven meetings are organized all over the world.

Almost 50 **DevOpsDays** in five years, with thousands of participants.

The early days

THE HASHTAG #DEVOPS is adopted. The *devops* topic — even if still undefined — gets debated in conferences and a new brand of community-driven meetings are organized all over the world. Almost 50 **DevOpsDays** in five years, with thousands of participants.

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

The early days

Attendees discuss about:

1. modern IT management,
2. techniques and tools to manage large infrastructure,
3. providing value to enterprise through faster delivery cycles, and faster deployment,
4. bridging the gap between developers and operations — does it ring a bell ?

All that is old is new again

Uncle Bob Martin says:

> *In 2001 a few of us met in hopes that we could agree on a simple statement that defines lightweight processes. We wrote a simple manifesto, and chose the name Agile. We had no idea how successful this idea would be. At that meeting, Kent Beck stated a prime goal: "**To heal the divide between business and development**".*

Today

MARCH, 2011. Gartner Group publishes the report *"The Rise of a New IT Operations Support Model"*, where it is stated that by year **2015** (*i.e.* exactly in 80 days from tomorrow) the DevOps movement would have grown from *a niche movement for cloud companies* to **adoption in more than a fifth of *Global 2000* enterprises**.

Italy: what's happening ?

October 2012 first Italian DevOpsDays, in Rome. A lot of people (200 participants) from all over the world.

February, 2013 in Florence, first Italian DevOps Meeting (Incontro DevOps Italia), 80+ people, community driven.

February, 2014 in Bologna, second Italian DevOps Meeting, 120+ people: a lot of interesting conversations.

Year 2015 who knows ?

BIODEC
Evolving ICT Infrastructures

Italy: what's happening ?

October 2012 first Italian DevOpsDays, in Rome. A lot of people (200 participants) from all over the world.

February, 2013 in Florence, first Italian DevOps Meeting (Incontro DevOps Italia), 80+ people, community driven.

February, 2014 in Bologna, second Italian DevOps Meeting, 120+ people: a lot of interesting conversations.

Year 2015 who knows ?

BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

So, what is this DevOps thing ?

MY VERY PERSONAL OPINION is that DevOps is a *pot-pourri* of different things, ideas, techniques and practices. I would say that it is a blend of:

1. agile *stuff*,
2. lean methodologies,
3. some characteristics of the free software communities: openness, sharing, open standards,
4. and probably something else.

So, what is this DevOps thing ?

MY VERY PERSONAL OPINION is that DevOps is a *pot-pourri* of different things, ideas, techniques and practices. I would say that it is a blend of:

1. agile *stuff*,
2. lean methodologies,
3. some characteristics of the free software communities: openness, sharing, open standards,
4. and probably something else.

BIODEC
Evolving ICT Infrastructures

So, what is this DevOps thing ?

MY VERY PERSONAL OPINION is that DevOps is a *pot-pourri* of different things, ideas, techniques and practices. I would say that it is a blend of:

1. agile *stuff*,
2. lean methodologies,
3. some characteristics of the free software communities: openness, sharing, open standards,
4. and probably something else.

BIODEC
Evolving ICT Infrastructures

So, what is this DevOps thing ?

MY VERY PERSONAL OPINION is that DevOps is a *pot-pourri* of different things, ideas, techniques and practices. I would say that it is a blend of:

1. agile *stuff*,
2. lean methodologies,
3. some characteristics of the free software communities: openness, sharing, open standards,
4. and probably something else.

BIODEC
Evolving ICT Infrastructures

So, what is this DevOps thing ?

MY VERY PERSONAL OPINION is that DevOps is a *pot-pourri* of different things, ideas, techniques and practices. I would say that it is a blend of:

1. agile *stuff*,
2. lean methodologies,
3. some characteristics of the free software communities: openness, sharing, open standards,
4. and probably something else.

. . . and what it is not

Defining DevOps by *negation*. That part is easier:

1. it is *not* a certification,
2. it is *not* a job title,
3. it is *not* a tool nor a software.

. . . and what it is not

Defining DevOps by *negation*. That part is easier:

1. it is *not* a certification,
2. it is *not* a job title,
3. it is *not* a tool nor a software.

. . . and what it is not

Defining DevOps by *negation*. That part is easier:

1. it is *not* a certification,
2. it is *not* a job title,
3. it is *not* a tool nor a software.

. . . and what it is not

Defining DevOps by *negation*. That part is easier:

1. it is *not* a certification,
2. it is *not* a job title,
3. it is *not* a tool nor a software.

An acronym: CAMS

C *culture*

A *automate*

M *measure*

S *share*

An acronym: CAMS

C *culture*

A *automate*

M *measure*

S *share*

An acronym: CAMS

C *culture*

A *automate*

M *measure*

S *share*

An acronym: CAMS

C *culture*

A *automate*

M *measure*

S *share*

An acronym: CAMS

C *culture*

A *automate*

M *measure*

S *share*

Culture

1. CREATE A CULTURE of collaboration. The first issue is the harder to get in practice, but it is probably the most important.

   People and process first. If you don't have culture, all automation attempts will be fruitless. (John Willis)

Automate

2 AUTOMATE everything. Let any task that can be done with software, be done by a program: write it, deploy it and run it.

All software is born equal under the sun: a *system* program is not an excuse for sloppy practices, lack of quality or misfeasance. Corollary: *sysadmin* is not an insult.

Automate

2 AUTOMATE everything. Let any task that can be done with software, be done by a program: write it, deploy it and run it.
All software is born equal under the sun: a *system* program is not an excuse for sloppy practices, lack of quality or misfeasance.
Corollary: *sysadmin* is not an insult.

Automate

2 AUTOMATE everything. Let any task that can be done with software, be done by a program: write it, deploy it and run it.
All software is born equal under the sun: a *system* program is not an excuse for sloppy practices, lack of quality or misfeasance.
Corollary: *sysadmin* is not an insult.

Infrastructure as code

Since only software determines what can be done, it means that also the *lack of it* defines what *cannot be done*, and in particular it is not acceptable to have:

1. hand-made configurations (*i.e.* snowflakes servers),
2. things that happens clicking on interfaces, with no versioning or change management in place,
3. people (*a.k.a.* consultants) that come, cast a spell and run away with money.

Infrastructure as code

Since only software determines what can be done, it means that also the *lack of it* defines what *cannot be done*, and in particular it is not acceptable to have:

1. hand-made configurations (*i.e.* snowflakes servers),
2. things that happens clicking on interfaces, with no versioning or change management in place,
3. people (*a.k.a.* consultants) that come, cast a spell and run away with money.

Infrastructure as code

Since only software determines what can be done, it means that also the *lack of it* defines what *cannot be done*, and in particular it is not acceptable to have:

1. hand-made configurations (*i.e.* snowflakes servers),
2. things that happens clicking on interfaces, with no versioning or change management in place,
3. people (*a.k.a.* consultants) that come, cast a spell and run away with money.

Infrastructure as code

Since only software determines what can be done, it means that also the *lack of it* defines what *cannot be done*, and in particular it is not acceptable to have:

1. hand-made configurations (*i.e.* snowflakes servers),
2. things that happens clicking on interfaces, with no versioning or change management in place,
3. people (*a.k.a.* consultants) that come, cast a spell and run away with money.

Measure everything

3 MEASURE all the parts of the infrastructure. Monitoring is not a new idea, and it has obviously not been invented by the DevOps community: the novelty of the DevOps approach is in considering monitoring *as a whole*: systems, applications, network. **Everything has to be available for anybody involved**.

Measure everything

The traditional approach to monitoring consists of some system management tool, **usually just for the system administrators**, tracking server resources or hardware performance data. Trouble arise since that tool is usually decoupled from an **ad hoc solution devised for the applications, by the application developer themselves**.

Share

4 SHARE a project outcome, an objective, practices, techniques, tools among different groups that have different roles and responsibilities.

> *Sharing is the loopback in the CAMS cycle. Creating a culture where people share ideas and problems is critical. (John Willis)*

Wrapping up

There is a chain of implications, and if you accept the premises the conclusion is inescapable:

1. if only the code defines the infrastructure,
2. and every action on the infrastructure has to be automated (that means: translated into code),
3. then the only way of determining an effect on the infrastructure is by programming,
4. and this means that **you are programmer**, willing or not, independently of your job title.

BIODEC
Evolving ICT Infrastructures

Wrapping up

There is a chain of implications, and if you accept the premises the conclusion is inescapable:

1. if only the code defines the infrastructure,
2. and every action on the infrastructure has to be automated (that means: translated into code),
3. then the only way of determining an effect on the infrastructure is by programming,
4. and this means that **you are programmer**, willing or not, independently of your job title.

BIODEC
Evolving ICT Infrastructures

Wrapping up

There is a chain of implications, and if you accept the premises the conclusion is inescapable:

1. if only the code defines the infrastructure,
2. and every action on the infrastructure has to be automated (that means: translated into code),
3. then the only way of determining an effect on the infrastructure is by programming,
4. and this means that **you are programmer**, willing or not, independently of your job title.

BIODEC
Evolving ICT Infrastructures

Wrapping up

There is a chain of implications, and if you accept the premises the conclusion is inescapable:

1. if only the code defines the infrastructure,
2. and every action on the infrastructure has to be automated (that means: translated into code),
3. then the only way of determining an effect on the infrastructure is by programming,
4. and this means that **you are programmer**, willing or not, independently of your job title.

BIODEC
Evolving ICT Infrastructures

Wrapping up

There is a chain of implications, and if you accept the premises the conclusion is inescapable:

1. if only the code defines the infrastructure,
2. and every action on the infrastructure has to be automated (that means: translated into code),
3. then the only way of determining an effect on the infrastructure is by programming,
4. and this means that **you are programmer**, willing or not, independently of your job title.

Index

BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

My personal interpretation

THE FOLLOWING PART OF THE TALK is my own personal view of DevOps' ideas. My aim is fostering the collaboration between programmers and ops (and front-end engineers, and QA, and whatever): to achieve that goal there are recent tools and techniques that I suggest to adopt.

DevOps as an evolution of XP ?

Agile / XP movement Good software is tested: you can disagree of course, but it is undoubted that “test first” was a (disrupting) novelty when it was introduced by eXtreme Programming.

DevOps Testing is not enough: good software is also monitored, logged and instrumented.

Should it have been called eXtreme sYstem Administration ? XYA ? XSD ? ESA ?

DevOps as an evolution of XP ?

Agile / XP movement Good software is tested: you can disagree of course, but it is undoubted that “test first” was a (disrupting) novelty when it was introduced by eXtreme Programming.

DevOps Testing is not enough: good software is also monitored, logged and instrumented.

Should it have been called eXtreme sYstem Administration ? XYA ? XSD ? ESA ?

A theory of monitoring, logging and alerting architectures

LET US DEFINE some key concepts: it will help us to properly understand how the components fit together. When we are *observing a system* we are interested in:

Logging that we define mainly as **events** management.

Monitoring that amounts to **measuring the system** and measure management.

Alerting that deals with **signalling the state of the system** and alarms management.

A theory of monitoring, logging and alerting architectures

LET US DEFINE some key concepts: it will help us to properly understand how the components fit together. When we are *observing a system* we are interested in:

Logging that we define mainly as **events** management.

Monitoring that amounts to **measuring the system** and measure management.

Alerting that deals with **signalling the state of the system** and alarms management.

A theory of monitoring, logging and alerting architectures

LET US DEFINE some key concepts: it will help us to properly understand how the components fit together. When we are *observing a system* we are interested in:

Logging that we define mainly as **events** management.

Monitoring that amounts to **measuring the system** and measure management.

Alerting that deals with **signalling the state of the system** and alarms management.

A theory of monitoring, logging and alerting architectures

LET US DEFINE some key concepts: it will help us to properly understand how the components fit together. When we are *observing a system* we are interested in:

Logging that we define mainly as **events** management.

Monitoring that amounts to **measuring the system** and measure management.

Alerting that deals with **signalling the state of the system** and alarms management.

Three components, a single system

An important issue: a lot of tools try to do many things at once, because of some misplaced sense of *semplicity* or *easy of use*. For example a software may store log data *and* check for conditions that trigger an alert.

Better would have been to stick with the Unix principle of doing *just one thing and doing it best*.

Three components, a single system

An important issue: a lot of tools try to do many things at once, because of some misplaced sense of *semplicity* or *easy of use*. For example a software may store log data *and* check for conditions that trigger an alert.
Better would have been to stick with the Unix principle of doing *just one thing and doing it best*.

Index

BIODEC
Evolving ICT Infrastructures

Beyond `tail -f /var/log/syslog`

LOGGING IS ABOUT MANAGING EVENT DATA. Often tools murk the waters putting together log data and measurement data, and sometimes log data do *murk themselves* because event information and measurement information are **mixed together**.

A log is not a *measurement system*, even if one of the characteristics of logs is *time information*.

Beyond `tail -f /var/log/syslog`

LOGGING IS ABOUT MANAGING EVENT DATA. Often tools murk the waters putting together log data and measurement data, and sometimes log data do *murk themselves* because event information and measurement information are **mixed together**.
A log is not a *measurement system*, even if one of the characteristics of logs is *time information*.

Example

Let us have a look at this line from an Apache log:

```
109.234.57.170 - - [07/Jul/2011:09:34:26 +0200] "GET /clienti-e-progetti/biocomp/biocomp-ups
HTTP/1.1" 302 5367 "-" "Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.2.18) Gecko/20110628
Ubuntu/10.10 (maverick) Firefox/3.6.18"
```

The *event* is that a certain URL has been served in a given moment in time. The other data helps in defining the context of that event, but do not change the nature of what happened (*i.e.* literally, the event itself).

Example

Let us have a look at this line from an Apache log:

```
109.234.57.170 - - [07/Jul/2011:09:34:26 +0200] "GET /clienti-e-progetti/biocomp/biocomp-ups
HTTP/1.1" 302 5367 "-" "Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.2.18) Gecko/20110628
Ubuntu/10.10 (maverick) Firefox/3.6.18"
```

The *event* is that a certain URL has been served in a given moment in time. The other data helps in defining the context of that event, but do not change the nature of what happened (*i.e.* literally, the event itself).

Example

5367 is the *size of response in bytes, excluding HTTP headers*

```
109.234.57.170 - - [07/Jul/2011:09:34:26 +0200] "GET /clienti-e-progetti/biocomp/biocomp-ups
HTTP/1.1" 302 5367 "-" "Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.2.18) Gecko/20110628
Ubuntu/10.10 (maverick) Firefox/3.6.18"
```

BIODEC
Evolving ICT Infrastructures

Example

5367 is the *size of response in bytes, excluding HTTP headers* while **302** is the *HTTP status*

```
109.234.57.170 - - [07/Jul/2011:09:34:26 +0200] "GET /clienti-e-progetti/biocomp/biocomp-ups
HTTP/1.1" 302 5367 "-" "Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.2.18) Gecko/20110628
Ubuntu/10.10 (maverick) Firefox/3.6.18"
```

BIODEC
Evolving ICT Infrastructures

Beyond `tail -f /var/log/syslog`

A logging infrastructure has the following components:

Route	syslog-ng, rsyslog, logstash, heka,
Store	elasticsearch (with mongodb),
Aggregate	graylog2,
Visualize	graylog2, kibana3 (soon kibana4),
Analyze	graylog2, kibana3 (soon kibana4),
Alert	an alerting system.

BIODEC
Evolving ICT Infrastructures

Beyond `tail -f /var/log/syslog`

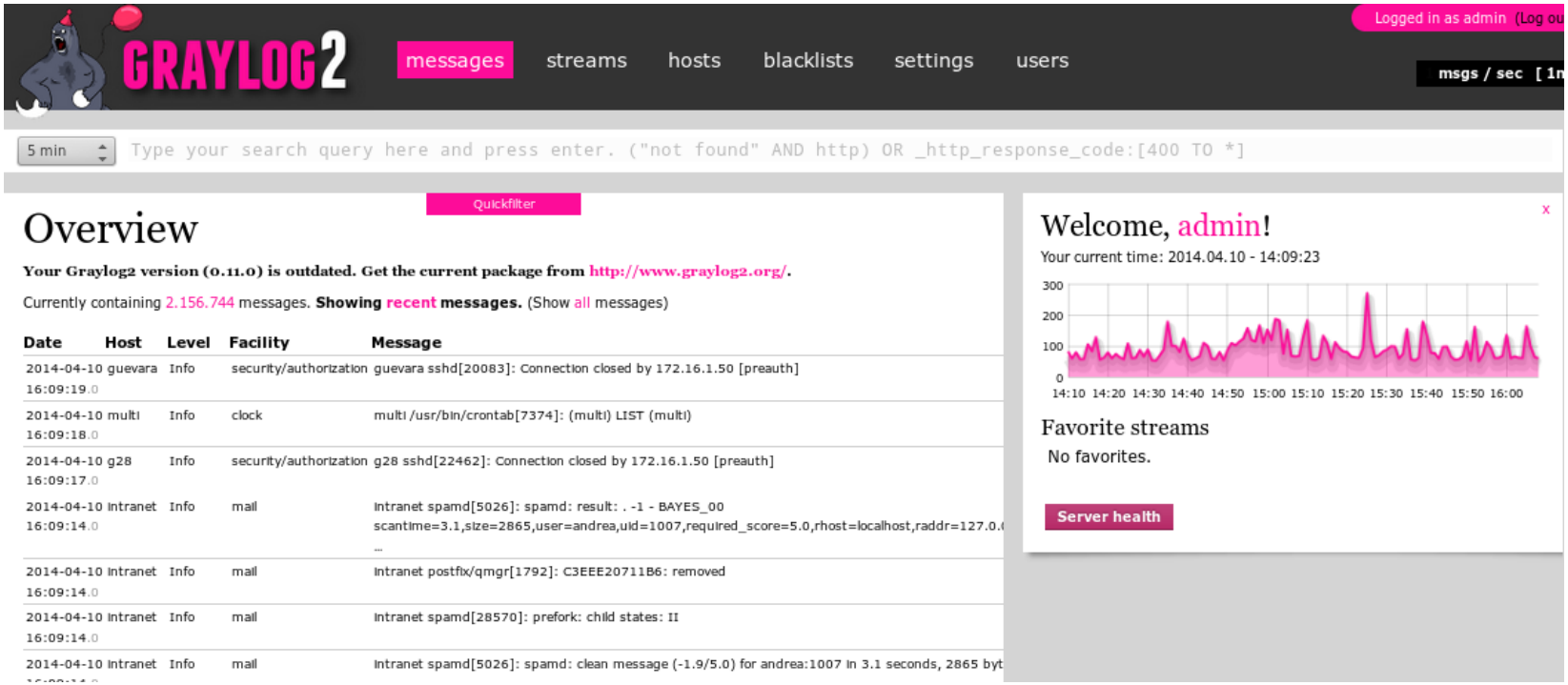
A “classical” system like syslog has a single software program doing almost everything — just alerting can be demanded to logwatch or similar scripts.

A classical system trades simplicity for scalability and extendability and is usually **useful only in the simplest scenarios**.

Beyond `tail -f /var/log/syslog`

A “classical” system like syslog has a single software program doing almost everything — just alerting can be demanded to logwatch or similar scripts.
A classical system trades simplicity for scalability and extendability and is usually **useful only in the simplest scenarios**.

Graylog2



BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

Measure ! Measure ! Measure everywhere !

TO DEFINE MEASUREMENT, we have to define what a **measure** is. A measure is a **numerical value with a name** and the **time** when that measurement was done. A succession of measures is a temporal series of numerical values linked to a tag (or name).

Measure ! Measure ! Measure everywhere !

A measurement infrastructure has the following components:

Route	collectd, statsd, metricsd,
Store	graphite (whisper), blueflood, influxdb,
Aggregate	graphite (carbon), blueflood, influxdb,
Visualize	graphite-web, grafana, graph-explorer,
Analyze	sensu,
Alert	an alerting system.

BIODEC
Evolving ICT Infrastructures

Measure ! Measure ! Measure everywhere !

A very simple “classical” system is a Nagios server with maybe Cacti / Pnp4Nagios or Munin to display the graphics of some metrics.

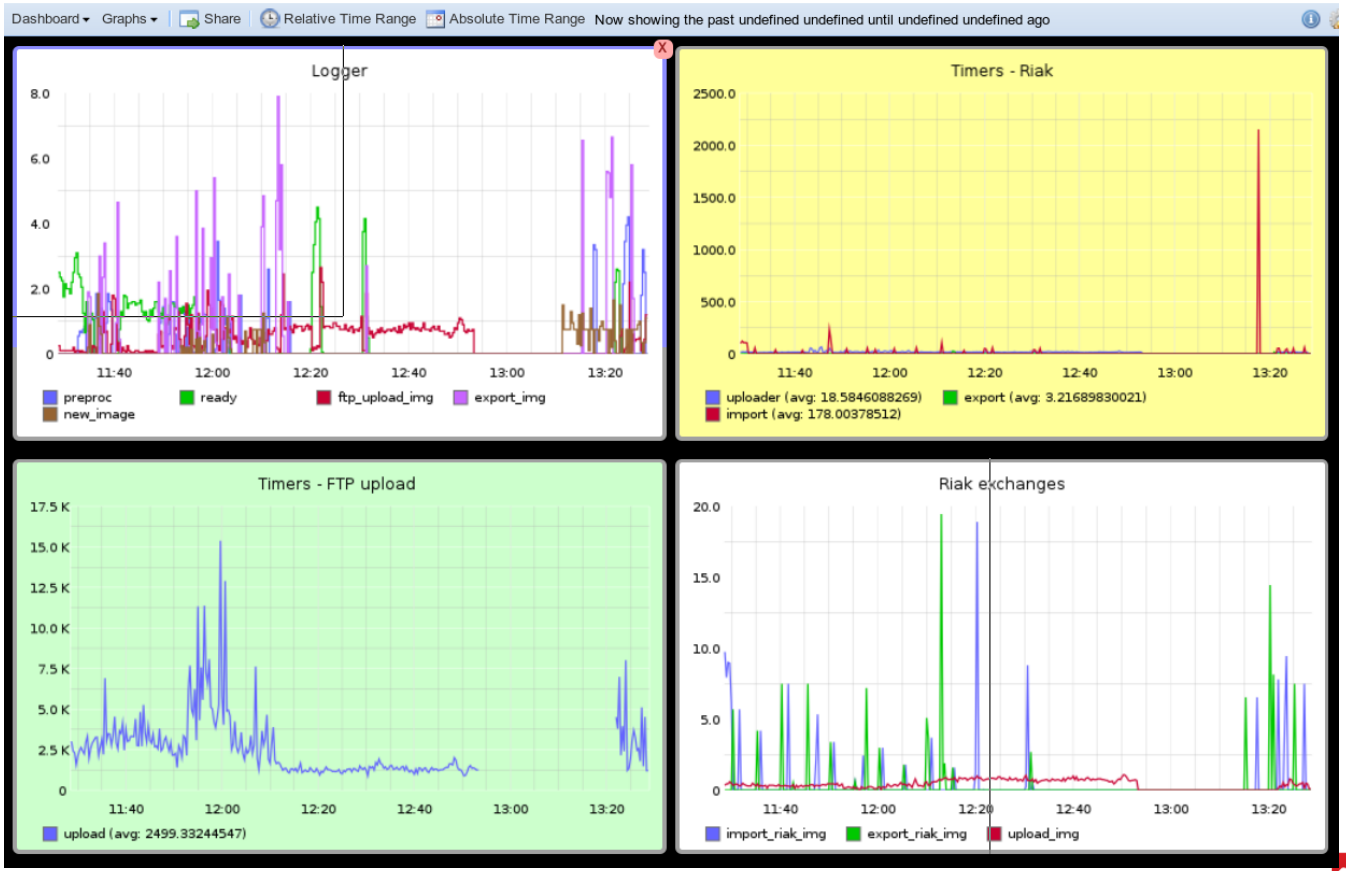
Visualizing information is the key

A ALL THE COMPONENTS are important, but one of them is more critical than the other. **Visualizing information is the key objective** of a monitoring infrastructure. Visualizing means making immediately available and **explicit** all the information gathered about the system.

Visualizing information is the key

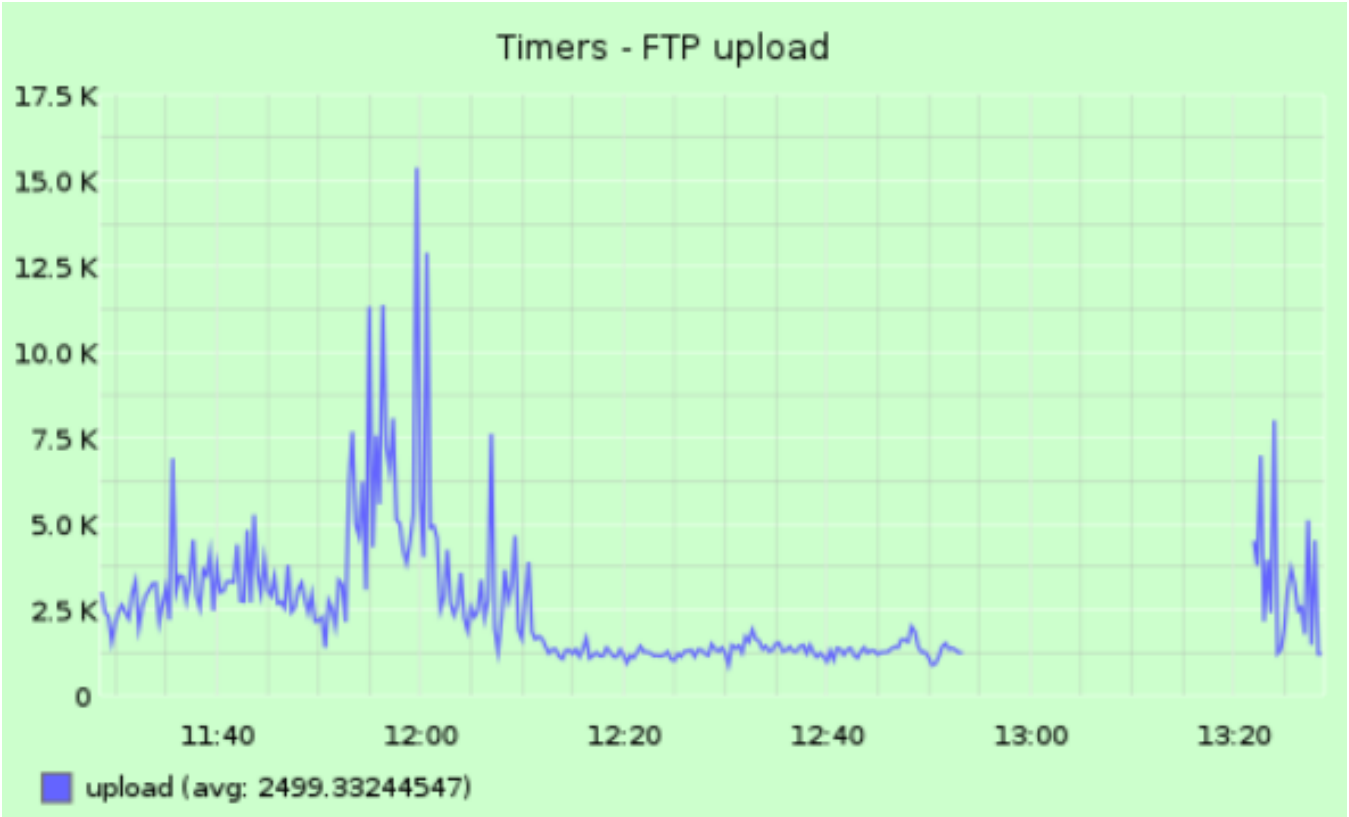
A ALL THE COMPONENTS are important, but one of them is more critical than the other. **Visualizing information is the key objective** of a monitoring infrastructure. Visualizing means making immediately available and **explicit** all the information gathered about the system.

Graphite + Statsd



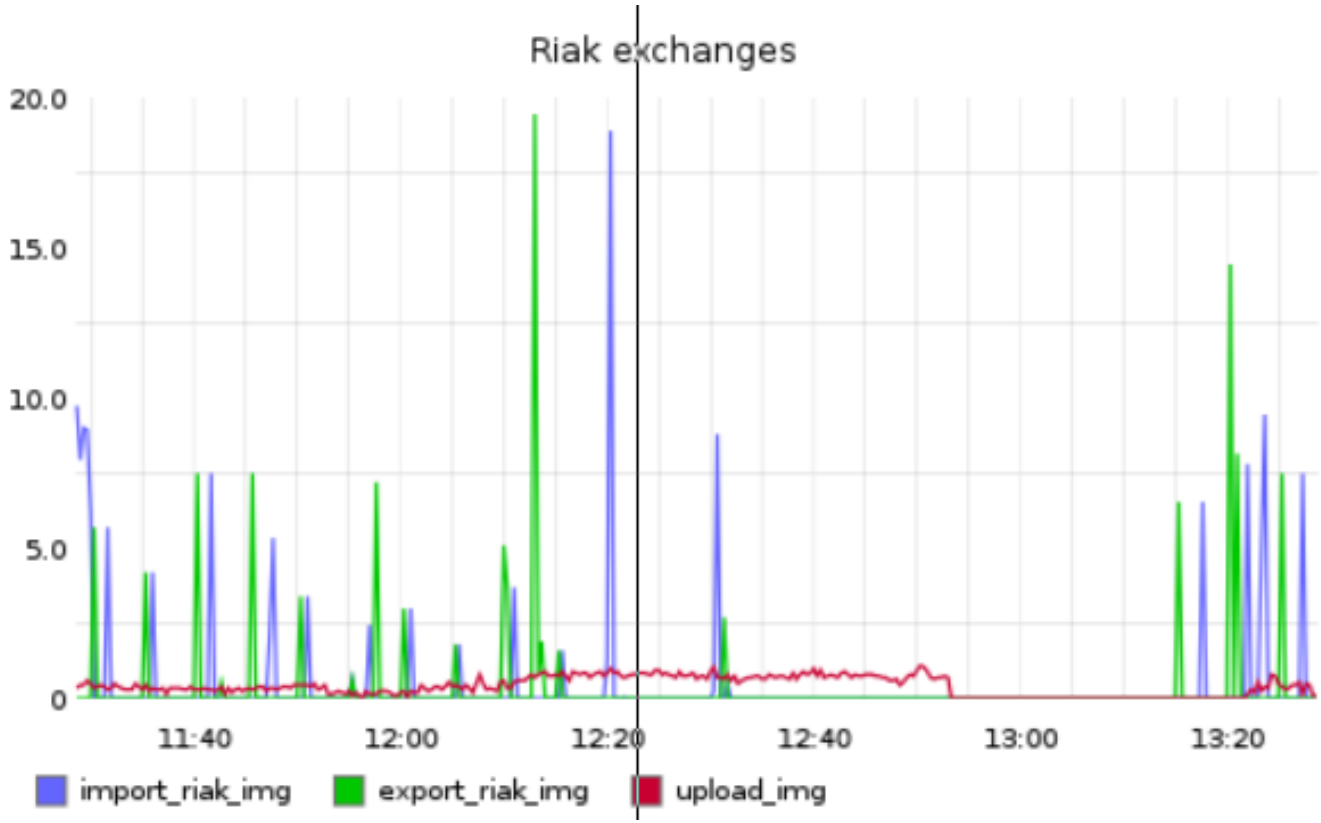
BIODEC
Evolving ICT Infrastructures

Graphite + Statsd



BIODEC
Evolving ICT Infrastructures

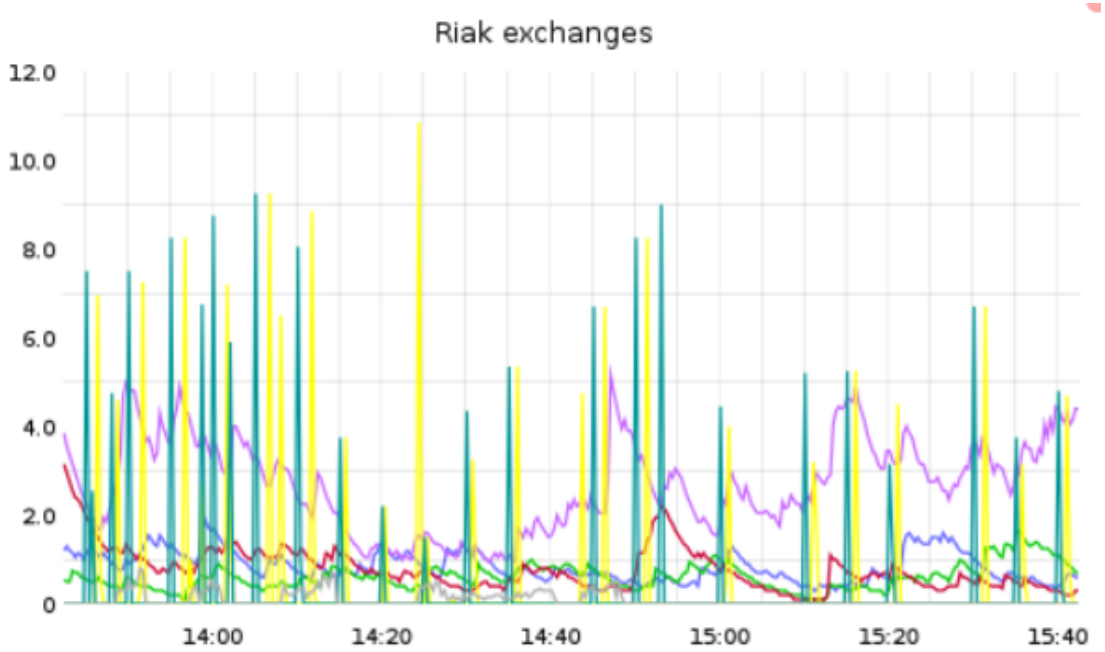
Graphite + Statsd



BIODEC
Evolving ICT Infrastructures

Application and system data together

This is the same graph as before, plotted together with the CPU load of **each Riak server**.



BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

A bell rings in the middle of the night

```
Address: node022.example.com
Service: Memory used
State: WARNING -> OK (RECOVERY)
Command: check_mk-mem.used
Output: OK - 3.07 GB used (2.82 GB RAM + 0.24 GB
SWAP, this is 4.9% of 62.89 GB RAM)
```

An alerting system

THE ALERTING SYSTEM was a common subcomponent of either the monitoring and the alerting system. An alerting system is a tool to **generate messages** related to a specific **state of the system**; the alerting system must also take care of **delivering** the messages to the correct **recipients**.

An alerting system

THE ALERTING SYSTEM was a common subcomponent of either the monitoring and the alerting system. An alerting system is a tool to **generate messages** related to a specific **state of the system**; the alerting system must also take care of **delivering** the messages to the correct **recipients**.

Building an alerting system

At minimum, an alerting system has the following components:

1. an alarm **generator**,
2. the **message** that describes the alarm,
3. the **recipients** of the message,
4. the **sub-system demanded to the delivery** of the message.

Building an alerting system

At minimum, an alerting system has the following components:

1. an alarm **generator**,
2. the **message** that describes the alarm,
3. the **recipients** of the message,
4. the **sub-system demanded to the delivery** of the message.

Building an alerting system

At minimum, an alerting system has the following components:

1. an alarm **generator**,
2. the **message** that describes the alarm,
3. the **recipients** of the message,
4. the **sub-system demanded to the delivery** of the message.

Building an alerting system

At minimum, an alerting system has the following components:

1. an alarm **generator**,
2. the **message** that describes the alarm,
3. the **recipients** of the message,
4. the **sub-system demanded to the delivery** of the message.

Building an alerting system

At minimum, an alerting system has the following components:

1. an alarm **generator**,
2. the **message** that describes the alarm,
3. the **recipients** of the message,
4. the **sub-system demanded to the delivery** of the message.

Building an alerting system

Which software to use for the various components of the alerting system:

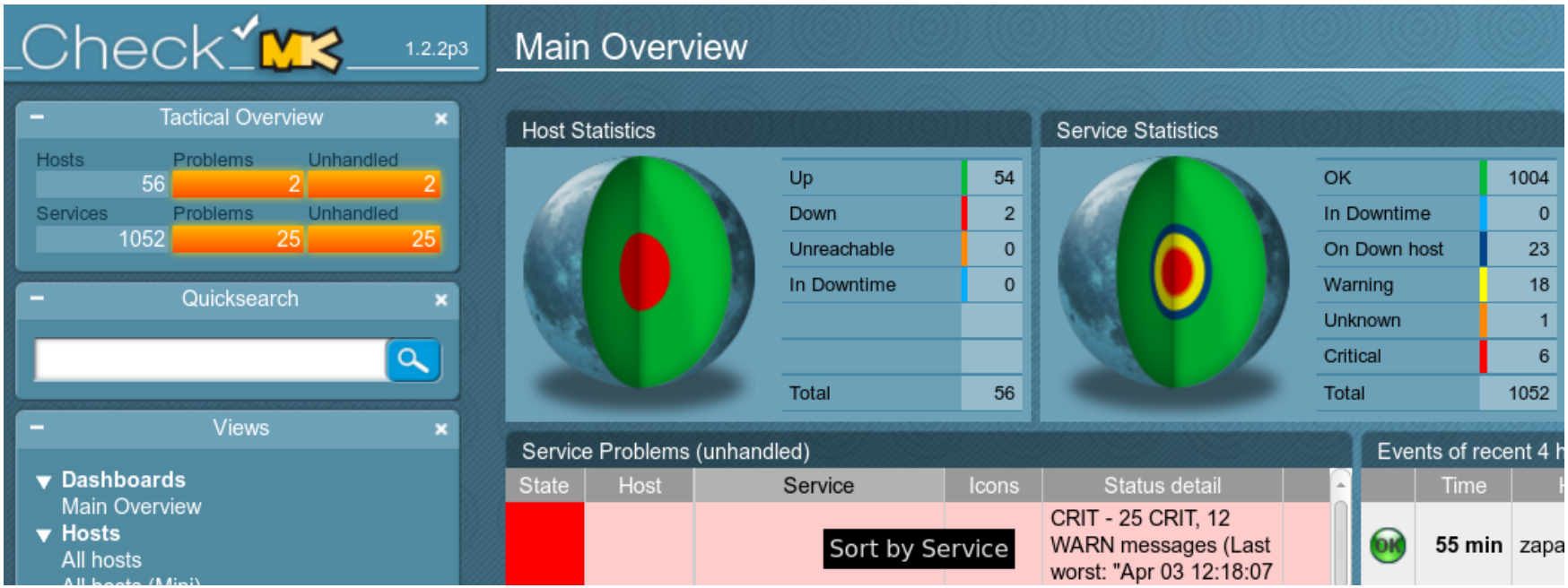
Generator nagios, icinga, flapjack (which calls this component *event processing*), sensu,

Message email, SMS, ...

Router nagios, icinga, flapjack, sensu,

Delivery is determined from the message type: SMTP for email, etcetera.

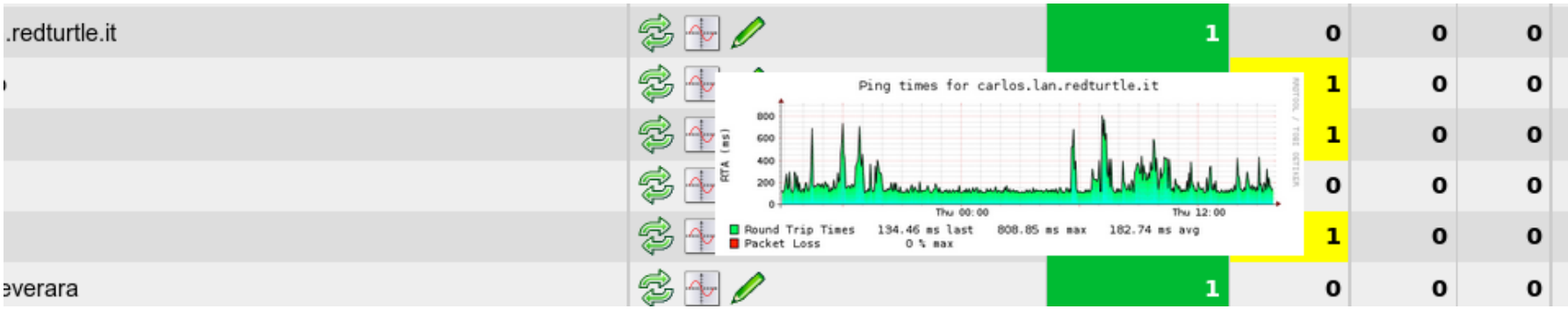
~~Good ol' Nagios~~ Check_MK Multisite



BIODEC
Evolving ICT Infrastructures

~~Good ol' Nagios~~ Check_MK Multisite

Again: "classical" systems are characterized by having many different kind of data mixed together in the same interface: for example monitoring graphs mixed to event status and alerting information.



BIODEC
Evolving ICT Infrastructures

Index

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Logging, monitoring and alerting are **useful** and more and more **mandatory** in the next future.
- They are components that — because of their nature — are **cross-functional**, since they integrate application (*Dev*) and system (*Ops*) information. The data integration is an **enabler** of activities (*i.e.* debug, continuous deployment, etcetera) that would otherwise be **impossible** or too expensive to achieve.
- There are modern tools in the free software world that allow companies to build **evolvable** and **modular** solutions.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Do not build or buy a **monolithic** solution !
- If you just do what everybody else is doing in the world of **high-performance companies**, you will have a robust solution to evolve your IT infrastructure and you will have enabled your teams **to operate as a single team**, and to bridge the divisions among different business areas.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Do not build or buy a **monolithic** solution !
- If you just do what everybody else is doing in the world of **high-performance companies**, you will have a robust solution to evolve your IT infrastructure and you will have enabled your teams **to operate as a single team**, and to bridge the divisions among different business areas.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Do not build or buy a **monolithic** solution !
- If you just do what everybody else is doing in the world of **high-performance companies**, you will have a robust solution to evolve your IT infrastructure and you will have enabled your teams **to operate as a single team**, and to bridge the divisions among different business areas.

BIODEC
Evolving ICT Infrastructures

Fostering collaboration through a common core of tools and techniques

- Do not build or buy a **monolithic** solution !
- If you just do what everybody else is doing in the world of **high-performance companies**, you will have a robust solution to evolve your IT infrastructure and you will have enabled your teams **to operate as a single team**, and to bridge the divisions among different business areas.

Thanks & see you soon . . .

Thanks for participating !

IDI2015 Incontro DevOps Italia 2015 ???

More news on BioDec's blog at `http://blog.biodec.com/`

BIODEC
Evolving ICT Infrastructures

Thanks & see you soon . . .

Thanks for participating !

IDI2015 Incontro DevOps Italia 2015 ???

More news on BioDec's blog at `http://blog.biodec.com/`

BIODEC
Evolving ICT Infrastructures

Thanks & see you soon . . .

Thanks for participating !

IDI2015 Incontro DevOps Italia 2015 ???

More news on BioDec's blog at http://blog.biodec.com/

BIODEC
Evolving ICT Infrastructures

Thanks & see you soon . . .

Thanks for participating !

IDI2015 Incontro DevOps Italia 2015 ???

More news on BioDec's blog at `http://blog.biodec.com/`

BIODEC
Evolving ICT Infrastructures