# Monitoraggio per programmatori Monitoratene tutti

Michele Finelli m@biodec.com BioDec

#### **Indice**

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



#### Indice

#### Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



#### Cloud + Virtualizzazione = Infrastruttura distribuita

RAZIE AI SISTEMI CLOUD E ALLA VIRTUALIZZAZIONE, nei prossimi anni sarà necessario dotarsi di strumenti efficaci per gestire la complessità di un'**infrastruttura distribuita**.

O rinunciare alla gestione della medesima.

#### Cloud + Virtualizzazione = Infrastruttura distribuita

RAZIE AI SISTEMI CLOUD E ALLA VIRTUALIZZAZIONE, nei prossimi anni sarà necessario dotarsi di strumenti efficaci per gestire la complessità di un'infrastruttura distribuita.

O rinunciare alla gestione della medesima.

#### Non solo distribuita, ma anche numerosa

Un secondo effetto è che l'infrastruttura sarà **molto maggiore** di quella a cui è abituato l'IT tradizionale, e certi problemi di gestione si presentano solo su grande scala.

O per meglio dire: sapere quello che accade nel **piccolo** non è un buon metro per capire cosa accade nel **grande**.

#### Non solo distribuita, ma anche numerosa

Un secondo effetto è che l'infrastruttura sarà molto maggiore di quella a cui è abituato l'IT tradizionale, e certi problemi di gestione si presentano solo su grande scala.

O per meglio dire: sapere quello che accade nel piccolo non è un buon metro per capire cosa accade nel **grande**.

#### Indice

Prima premessa: una cosa chiamata cloud

#### Seconda premessa: una cosa chiamata DevOps

**CAMS** 

M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



- Tutto incomincia con ... Patrick Debois che nel 2007 si trova a svolgere un lavoro "ibrido" di sviluppo e di sistemi, e non è contento di come sta procedendo.
- Agile 2008 Andrew Shafer parla di "Agile Infrastructure", o per meglio dire "monologa".
- 23 giugno 2009 John Allspaw presenta il talk "10+ deploys per day: Dev & Ops cooperation at Flickr".
- 30-31 ottobre 2009 Il primo *DevOps Days* a Ghent, in Belgio. Grande successo di pubblico e di critica.
- \*da una presentazione di Damon Edwards pubblicata su IT Revolutions.



- Tutto incomincia con ... Patrick Debois che nel 2007 si trova a svolgere un lavoro "ibrido" di sviluppo e di sistemi, e non è contento di come sta procedendo.
  - Agile 2008 Andrew Shafer parla di "Agile Infrastructure", o per meglio dire "monologa".
- 23 giugno 2009 John Allspaw presenta il talk "10+ deploys per day: Dev & Ops cooperation at Flickr".
- 30-31 ottobre 2009 Il primo *DevOps Days* a Ghent, in Belgio. Grande successo di pubblico e di critica.
- \*da una presentazione di Damon Edwards pubblicata su IT Revolutions.



- Tutto incomincia con ... Patrick Debois che nel 2007 si trova a svolgere un lavoro "ibrido" di sviluppo e di sistemi, e non è contento di come sta procedendo.
  - Agile 2008 Andrew Shafer parla di "Agile Infrastructure", o per meglio dire "monologa".
- 23 giugno 2009 John Allspaw presenta il talk "10+ deploys per day: Dev & Ops cooperation at Flickr".
- 30-31 ottobre 2009 Il primo *DevOps Days* a Ghent, in Belgio. Grande successo di pubblico e di critica.
- \*da una presentazione di Damon Edwards pubblicata su IT Revolutions.



- Tutto incomincia con ... Patrick Debois che nel 2007 si trova a svolgere un lavoro "ibrido" di sviluppo e di sistemi, e non è contento di come sta procedendo.
  - Agile 2008 Andrew Shafer parla di "Agile Infrastructure", o per meglio dire "monologa".
- 23 giugno 2009 John Allspaw presenta il talk "10+ deploys per day: Dev & Ops cooperation at Flickr".
- 30-31 ottobre 2009 Il primo *DevOps Days* a Ghent, in Belgio. Grande successo di pubblico e di critica.
- \*da una presentazione di Damon Edwards pubblicata su IT Revolutions.



#DEVOPS diventa un tema caldo in numerose conferenze: viene creato un formato, i DevOps Days, che in pochi anni si replicano per decine di volte in tutto il mondo.

Si enfatizza il tema di come funziona l'IT e di come dovrebbe invece funzionare. Il discorso verte sugli strumenti, su quali funzionano e su quali no, sulle *best practices* e sulle tecniche.

## Il movimento oggi

Narzo Del 2011 anche Gartner si accorge del movimento e pubblica il suo oroscopo: "The Rise of a New IT Operations Support Model" che prevede che per il 2015 il movimento sarà passato, da una nicchia nell'ambito cloud, all'adozione nel 20% delle imprese Global 2000.

Si afferma a livello globale l'esistenza di un movimento from practitioners, to practitioners.

## Il movimento oggi

NEL MARZO DEL 2011 anche Gartner si accorge del movimento e pubblica il suo oroscopo: "The Rise of a New IT Operations Support Model" che prevede che per il 2015 il movimento sarà passato, da una nicchia nell'ambito cloud, all'adozione nel 20% delle imprese Global 2000. Si afferma a livello globale l'esistenza di un movimento from practitioners, to practitioners.

#### Il movimento in Italia

- ottobre 2012 La prima edizione dei DevOps Days italiana, a Roma, con quasi duecento partecipanti da tutto il mondo.
- febbraio 2013 A Firenze si è tenuto il primo "Incontro DevOps Italia", con 80+ persone presenti.
- febbraio 2014 Bologna: il secondo "Incontro DevOps Italia", con 120+ persone presenti.
- aprile 2015 Bologna: il terzo "Incontro DevOps Italia", con 200 persone presenti.



#### Cosa c'è dentro ...

A MIA PERSONALE impressione è che sia un *pot-pourri* di teorie, tecniche e pratiche proveniente da ambiti differenti:

- 1. il movimento agile,
- 2. le lean methodologies,
- 3. le caratteristiche delle comunità *free software* (apertura, condivisione, codice aperto, standard).

#### Cosa c'è dentro ...

A MIA PERSONALE impressione è che sia un *pot-pourri* di teorie, tecniche e pratiche proveniente da ambiti differenti:

- 1. il movimento agile,
- 2. le lean methodologies,
- 3. le caratteristiche delle comunità *free software* (apertura, condivisione, codice aperto, standard).

#### Cosa c'è dentro ...

A MIA PERSONALE impressione è che sia un *pot-pourri* di teorie, tecniche e pratiche proveniente da ambiti differenti:

- 1. il movimento agile,
- 2. le lean methodologies,
- 3. le caratteristiche delle comunità *free software* (apertura, condivisione, codice aperto, standard).

#### ...e cosa rimane fuori

Si fa prima a dire che cosa non sia DevOps:

- ► non è una certificazione,
- ▶ non è un titolo,
- ▶ non è strumento specifico o un software particolare.

└\_CAMS

#### Indice

Prima premessa: una cosa chiamata cloud

## Seconda premessa: una cosa chiamata DevOps CAMS

M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



CAMS

## Uno slogan: CAMS

C culture

A automate

M measure

Seconda premessa: una cosa chiamata DevOps

└\_CAMS

## Uno slogan: CAMS

C culture

A automate

M measure



Seconda premessa: una cosa chiamata DevOps

CAMS

## Uno slogan: CAMS

C culture

A automate

M measure



CAMS

## Uno slogan: CAMS

C culture

A automate

M measure



## Uno slogan: CAMS

C culture

A automate

M measure



└\_CAMS

#### Culture

CREARE UNA CULTURA della collaborazione. È il primo dettame, ma è sovente il più negletto — anche perché è il più difficile da mettere in pratica.

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

└\_CAMS

#### **Automate**

AUTOMATIZZARE ogni azione. Se un'azione manuale può essere svolta da un programma, che lo si scriva. E lo si scriva secondo i crismi con cui si scrivono i programmi: il fatto che sia un programma per i sistemi (o per i *server*) non è un'offesa.

#### Automate

AUTOMATIZZARE ogni azione. Se un'azione manuale può essere svolta da un programma, che lo si scriva. E lo si scriva secondo i crismi con cui si scrivono i programmi: il fatto che sia un programma per i sistemi (o per i *server*) non è un'offesa

"Sistemista" non è un'offesa.

#### Automate

AUTOMATIZZARE ogni azione. Se un'azione manuale può essere svolta da un programma, che lo si scriva. E lo si scriva secondo i crismi con cui si scrivono i programmi: il fatto che sia un programma per i sistemi (o per i *server*) non è un'offesa.

"Sistemista" non è un'offesa.

#### Corollario: Infrastructure as code

- 1. configurazioni manuali,
- 2. cose che si *cliccano* di qua e di là,
- 3. persone (a.k.a. consulenti) che arrivano e fanno cose

#### Corollario: Infrastructure as code

- 1. configurazioni manuali,
- 2. cose che si *cliccano* di qua e di là,
- 3. persone (a.k.a. consulenti) che arrivano e fanno cose

#### Corollario: Infrastructure as code

- 1. configurazioni manuali,
- 2. cose che si cliccano di qua e di là,
- 3. persone (a.k.a. consulenti) che arrivano e fanno cose

#### Corollario: Infrastructure as code

- 1. configurazioni manuali,
- 2. cose che si cliccano di qua e di là,
- 3. persone (a.k.a. consulenti) che arrivano e fanno cose.

└\_CAMS

## Measure everything

3 MISURARE ogni componente dell'infrastruttura. Il concetto di *monitoring* non è affatto nuovo, l'innovazione è nell'avere degli strumenti che permettano di controllare **tutte le parti**.

Nell'approccio tradizionale si controlla **solo la parte sistemistica** mentre la parte applicativa ha — nella migliore delle ipotesi — al più una soluzione *ad hoc*.

- Seconda premessa: una cosa chiamata DevOps
  - └-CAMS

### Measure everything

MISURARE ogni componente dell'infrastruttura. Il concetto di *monitoring* non è affatto nuovo, l'innovazione è nell'avere degli strumenti che permettano di controllare **tutte le parti**. Nell'approccio tradizionale si controlla **solo la parte sistemistica** mentre la parte applicativa ha — nella migliore delle ipotesi — al più una soluzione *ad hoc*.

- Seconda premessa: una cosa chiamata DevOps
  - └-CAMS

#### Share

4 CONDIVIDERE un progetto comune, un obiettivo, delle pratiche, delle tecniche, degli strumenti, fra gruppi eterogenei, e che hanno obiettivi differenti (complementari).

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

# Forse non si è capito ma ...

... dire che solo il codice definisce l'infrastruttura, e che ogni azione deve essere automatizzata ... ovvero trasformata in software ... implica che chiunque adotti queste pratiche, e indipendentemente dal nome con cui si fa chiamare, è un ...

# PROGRAMMATORE!



└\_CAMS

# Forse non si è capito ma ...

... dire che solo il codice definisce l'infrastruttura, e che ogni azione deve essere automatizzata ... ovvero trasformata in *software* ... implica che chiunque adotti queste pratiche, e indipendentemente dal nome con cui si fa chiamare, è un ...

# Programmatore!



└-CAMS

# Forse non si è capito ma ...

...dire che solo il codice definisce l'infrastruttura, e che ogni azione deve essere automatizzata ... ovvero trasformata in software ... implica che chiunque adotti queste pratiche, e indipendentemente dal nome con cui si fa chiamare, è un ...

# PROGRAMMATORE!



└-CAMS

## Forse non si è capito ma ...

... dire che solo il codice definisce l'infrastruttura, e che ogni azione deve essere automatizzata ... ovvero trasformata in *software* ... implica che chiunque adotti queste pratiche, e indipendentemente dal nome con cui si fa chiamare, è un ...

# PROGRAMMATORE!



#### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



#### Le misure

culture automate **M measure** 

share

- In un test si effettua un'operazione e si verifica che una quantità (tipicamente il risultato) sia consistente con una determinata proprietà (la cui validità è lo scopo del test).
- Nei sistemi di misura si disaccoppia la raccolta delle quantità dalla verifica delle proprietà.
- La misura continua del comportamento del codice puó essere vista come la naturale evoluzione dell'approccio basato sui test.

- In un test si effettua un'operazione e si verifica che una quantità (tipicamente il risultato) sia consistente con una determinata proprietà (la cui validità è lo scopo del test).
- Nei sistemi di misura si disaccoppia la raccolta delle quantità dalla verifica delle proprietà.
- La misura continua del comportamento del codice puó essere vista come la naturale evoluzione dell'approccio basato sui test.

- In un test si effettua un'operazione e si verifica che una quantità (tipicamente il risultato) sia consistente con una determinata proprietà (la cui validità è lo scopo del test).
- Nei sistemi di misura si disaccoppia la raccolta delle quantità dalla verifica delle proprietà.
- La misura continua del comportamento del codice puó essere vista come la naturale evoluzione dell'approccio basato sui test.

- In un test si effettua un'operazione e si verifica che una quantità (tipicamente il risultato) sia consistente con una determinata proprietà (la cui validità è lo scopo del test).
- Nei sistemi di misura si disaccoppia la raccolta delle quantità dalla verifica delle proprietà.
- La misura continua del comportamento del codice puó essere vista come la naturale evoluzione dell'approccio basato sui test.

## Tesi provocatoria

UN SISTEMA DI MISURA BEN STRUTTURATO puó efficamente sostituire un insieme di test, col vantaggio che le misure permangono anche col sistema **in produzione**, mentre i test vengono eseguiti solo nelle fasi antecedenti al *deployment*.

## Measure! Measure! Measure everywhere!

PER DEFINIRE l'atto del controllare, dobbiamo definire cosa intendiamo controllare, ovvero cosa intendiamo misurare.

Una misura è un valore numerico con un nome e il momento in cui essa è stata effettuata.

Una successione di misure è pertanto una serie temporale di valore numerici associati ad un'etichetta (o nome).

## Measure! Measure! Measure everywhere!

PER DEFINIRE l'atto del controllare, dobbiamo definire cosa intendiamo controllare, ovvero cosa intendiamo misurare. Una misura è un valore numerico con un nome e il momento in cui essa è stata effettuata.

Una successione di misure è pertanto una serie temporale di valore numerici associati ad un'etichetta (o nome).

## Measure! Measure! Measure everywhere!

PER DEFINIRE l'atto del controllare, dobbiamo definire cosa intendiamo controllare, ovvero cosa intendiamo misurare. Una misura è un valore numerico con un nome e il momento in cui essa è stata effettuata.

Una successione di misure è pertanto una serie temporale di valore numerici associati ad un'etichetta (o nome).

#### Visualizzare i dati

NA CARATTERISTICA CHIAVE di un sistema di monitoring è la visualizzazione dei dati, ovvero rendere immediatamente esplicite le informazioni.

# I dati applicativi

N GENERALE I SISTEMI DI MONITORAGGIO usati in ambito Ops mostrano dei dati di funzionamento dei server: ovvero quantità come l'uso della CPU, della RAM eccetera.

È fondamentale integrare questi dati con i dati di funzionamento delle applicazioni, ovvero i dati applicativi.

# I dati applicativi

N GENERALE I SISTEMI DI MONITORAGGIO usati in ambito Ops mostrano dei dati di funzionamento dei server: ovvero quantità come l'uso della CPU, della RAM eccetera. È fondamentale integrare questi dati con i dati di funzionamento delle applicazioni, ovvero i dati applicativi.

## I componenti in uso in BioDec

```
Route collectd, statsd;
Store graphite (whisper);
Aggregate graphite (carbon);
Visualize graphite-web, grafana, graph-explorer.
```

Graphite

#### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

# M come monitoraggio Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



Graphite

# Cos'è Graphite

GRAPHITE È UN software free per gestire serie temporali numeriche (metriche) e per visualizzarle. Graphite non raccoglie i dati direttamente: esistono numerosi strumenti che "parlano graphite" e che possono persistere i loro dati su un Graphite server.

## I pezzi che compongono Graphite

Ci sono tre componenti fondamentali, tutte scritte in Python:

carbon È il componente che eroga il servizio di collettore delle serie numeriche.

whisper È un'implementazione di un *round-robin database*, sul quale vengono rese persistenti le serie.

graphite webapp È il *frontend* del sistema, ovvero il sistema di visualizzazione delle serie, tramite un'applicazione web.

## Qualche parola in più per carbon

- ► Esistono tre modi di usare carbon: quello base prevede di lanciare il demone carbon-cache.py è il sistema di default.
- Esistono però altre due modalità: carbon-relay.py e carbon-aggregator.py che soddisfano la necessità, rispettivamente, di fare replicazione (e sharding) e di fare buffering.

#### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

#### Collectd

Statsd

Crofono

All together now

Conclusioni



#### I dati dei server

PER RACCOGLIERE I DATI DEI SERVER usiamo Collectd: esistono altre varianti, come Diamond, o uno recentissimo che si chiama Fullerite.

Il punto chiave è raccogliere le metriche dei dati operativi dei server e mandarli ad un server Graphite.

#### I dati dei server

PER RACCOGLIERE I DATI DEI SERVER usiamo Collectd: esistono altre varianti, come Diamond, o uno recentissimo che si chiama Fullerite.

Il punto chiave è raccogliere le metriche dei dati operativi dei server e mandarli ad un server Graphite.

#### Dati di sistema

N ELLA LORO CONFIGURAZIONE MINIMA questi strumenti permettono praticamente con la configurazione di default di raccogliere le metriche di sistema.

Questo implica che con uno sforzo minimo è possibile consolidare in un unico posto tutte le metriche della parte server della vostra infrastruttura.

#### Dati di sistema

NELLA LORO CONFIGURAZIONE MINIMA questi strumenti permettono praticamente con la configurazione di default di raccogliere le metriche di sistema.

Questo implica che con uno sforzo minimo è possibile consolidare in un unico posto tutte le metriche della parte server della vostra infrastruttura.

#### Dati di servizi o di middleware

A SECONDA DEI SERVIZI — ovvero dei vari middleware che utilizzate — puó essere più o meno complesso aggiungere le metriche relative: nel senso che puó essere necessario configurare dei *plugin*, come doverseli scrivere.

Dipende dal software in oggetto: tutti i servizi più diffusi sotto Linux sono di solito **ben supportati** (postgresql, mysql, apache, postfix, eccetera).

#### Dati di servizi o di middleware

A SECONDA DEI SERVIZI — ovvero dei vari middleware che utilizzate — puó essere più o meno complesso aggiungere le metriche relative: nel senso che puó essere necessario configurare dei *plugin*, come doverseli scrivere. Dipende dal software in oggetto: tutti i servizi più diffusi sotto Linux sono di solito **ben supportati** (postgresql, mysql, apache, postfix, eccetera).

#### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



#### Enter statsd

POICHÉ GRAPHITE, PER SCELTA PROGETTUALE non si occupa della raccolta dei dati, è necessario avere un modo consistente di svolgere quel compito, ad esempio:

A network daemon that (...) listens for statistics, like counters and timers, sent over UDP and sends aggregates to one or more pluggable backend services (e.g., Graphite).

Statsd

#### Enter statsd

POICHÉ GRAPHITE, PER SCELTA PROGETTUALE non si occupa della raccolta dei dati, è necessario avere un modo consistente di svolgere quel compito, ad esempio:

A network daemon that (...) listens for statistics, like counters and timers, sent over UDP and sends aggregates to one or more pluggable backend services (e.g., Graphite).

#### I concetti chiave di statsd

buckets È in pratica il nome, o etichetta, della misura. La convenzione è che il punto '.' separi i campi in modo da organizzarli ad albero, ad esempio:

- collectd.server1.load
- ▶ collectd.server2.load

sono due misure distinte, ma gli strumenti di navigazione mostreranno due children chiamati server1 e server2 sotto la radice collectd.

#### I concetti chiave di statsd

valori Ogni misura ha un valore, che per convenzione è un intero: si noti che non sono presenti le unità di misura.

Ogni valore ha un modificatore.

modificatori Determinano il tipo del valore e possono essere dei seguenti tipo: Counting, Timing, Gauges, Sets.

flush È l'intervallo di tempo dopo il quale le statistiche sono aggregate e mandate *upstream*.

#### I concetti chiave di statsd

valori Ogni misura ha un valore, che per convenzione è un intero: si noti che non sono presenti le unità di misura.

Ogni valore ha un modificatore.

modificatori Determinano il tipo del valore e possono essere dei seguenti tipo: Counting, Timing, Gauges, Sets.

flush È l'intervallo di tempo dopo il quale le statistiche sono aggregate e mandate *upstream*.



#### I concetti chiave di statsd

valori Ogni misura ha un valore, che per convenzione è un intero: si noti che non sono presenti le unità di misura.

Ogni valore ha un modificatore.

modificatori Determinano il tipo del valore e possono essere dei seguenti tipo: Counting, Timing, Gauges, Sets.

flush È l'intervallo di tempo dopo il quale le statistiche sono aggregate e mandate *upstream*.



Counting libri:3 c indica di aggiungere 3 al contatore 'libri'. Al flush il valore corrente è spedito, e 'libri' viene posto a 0.

Counting <a href="libri:3">libri:3</a> c indica di **aggiungere 3** al contatore 'libri'. Al *flush* il valore corrente è spedito, e 'libri' viene posto a 0.

È uno dei contatori più semplici: conta quante volte accade un determinato evento nel periodo di campionamento.

E quindi l'unica operazione consentita è una **add** di un valore intero al valore precedente.

Counting libri:3 c indica di aggiungere 3 al contatore 'libri'. Al *flush* il valore corrente è spedito, e 'libri' viene posto a 0.

È uno dei contatori più semplici: conta quante volte accade un determinato evento nel periodo di campionamento.

E quindi l'unica operazione consentita è una **add** di un valore intero al valore precedente.



Timing page: 511 | ms indica che una certa azione è stata svolta in 511 (millisecondi); statsd in maniera automatica calcola, in base ai valori ricevuti nell'intervallo di campionamento:

- 1. i percentili,
- 2. la media,
- 3. la deviazione standard,
- 4. la somma,
- 5. gli estremi.



Gauges accelerazione: 3 | g è una misura del parametro indicato da 'accelerazione': se durante un intervallo di campionamento il valore non cambia, il *flush* spedisce il valore precedente (si noti la differenza dal counter).

I gauge consentono anche modifiche incrementali, come accelerazione:+1|g oppure accelerazione:-2|g.



Gauges accelerazione:3|g è una misura del parametro indicato da 'accelerazione': se durante un intervallo di campionamento il valore non cambia, il flush spedisce il valore precedente (si noti la differenza dal counter).

I gauge consentono anche modifiche incrementali, come accelerazione:+1|g oppure accelerazione:-2|g.

Sets bag: 34 | s è un po' controintuitivo, in quando si aggiunge il valore '34' ad un insieme chiamato 'bag', e si ritorna la cardinalità di 'bag'.

Ergo, se si inserisce di nuovo il valore '34', il valore di 'bag' non cambia.

Un tipico caso d'uso è inserire le UID di oggetti come le login, per sapere in un determinato momento quante persone sono presenti sul serve

Sets bag: 34 | s è un po' controintuitivo, in quando si aggiunge il valore '34' ad un insieme chiamato 'bag', e si ritorna la cardinalità di 'bag'.

Ergo, se si inserisce di nuovo il valore '34', il valore di 'bag' non cambia.

Un tipico caso d'uso è inserire le UID di oggetti come le login, per sapere in un determinato momento quante persone sono presenti sul server.

Sets bag:34|s è un po' controintuitivo, in quando si aggiunge il valore '34' ad un insieme chiamato 'bag', e si ritorna la cardinalità di 'bag'.

Ergo, se si inserisce di nuovo il valore '34', il valore di 'bag' non cambia.

Un tipico caso d'uso è inserire le UID di oggetti come le login, per sapere in un determinato momento quante persone sono presenti sul server.

Statsd

## Un esempio d'uso di statsd

```
@stats.DebugTimer('export_riak_img')
def export_img(img_binary, img_file_name):
    """ Export image from riak and save as filename."""
    with file(img_file_name, 'wb') as img_file_obj:
        img_file_obj.write(img_binary)
```

### Un semplice decoratore

```
class DebugTimer():
    """ Decorator for statsd.timing().
    If stated is disabled the function will not be altered."""
    def init (self, stat name):
        self.stat_name = stat_name
    def __call__(self, original_func):
        @wraps(original_func)
        def wrapped f(*args, **kw):
            if STATSD ENABLED:
                # MEASURE THE EXECUTION TIME
                with STATSD HANDLER.timer(self.stat name):
                    ret = original func(*args, **kw)
            else:
                # CALL ONLY THE ORIGINAL FUNCTION
                ret = original_func(*args, **kw)
            return ret
        return wrapped f
```



### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

Collectd

Stats

#### Grafana

All together now

Conclusioni



#### Un frontend moderno

#### Dal sito:

The leading graph and dashboard builder for visualizing time series metrics.

È un'alternativa (consigliata) all'uso del frontend che è in *bundle* con Graphite.

- M come monitoraggio
  - Grafana

## Un frontend agnostico

N PARTICOLARE GRAFANA puó mostrare serie temporali che siano già presenti in graphite (e quindi permette, in un certo senso di usare graphite solo come *storage*), così come da altri backend.

#### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

Collectd

Statsd

Grafana

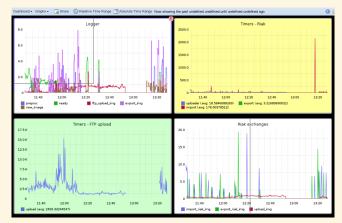
All together now

Conclusioni

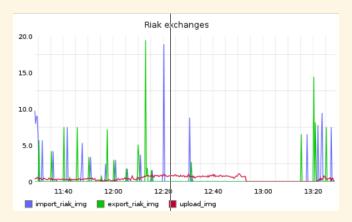


- └M come monitoraggio
  - ☐ All together now

# Graphite

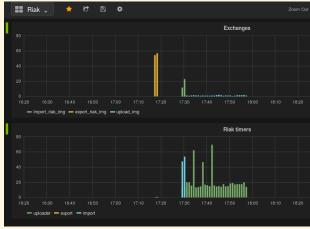


# Graphite



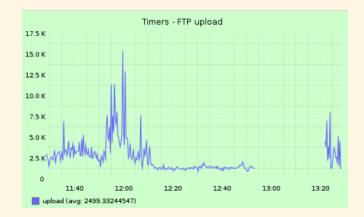
- └M come monitoraggio
  - L All together now

#### Grafana



- └M come monitoraggio
  - L All together now

## Graphite

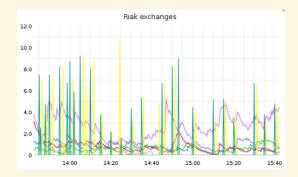


#### Grafana

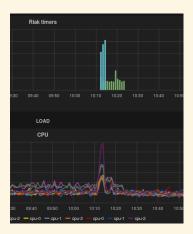


## Integrazione di dati applicativi e di sistema

È lo stesso grafico di prima un paio di ore dopo, per la precisione — ma col load di ogni Riak server del cluster (5 server virtuali diversi).



# Integrazione in Grafana



Le stesse informazioni (in un altro periodo) ma ancora più leggibili e navigabili, nonostante l'aumento del numero di metriche mostrate.

### Indice

Prima premessa: una cosa chiamata cloud

Seconda premessa: una cosa chiamata DevOps CAMS

#### M come monitoraggio

Graphite

Collectd

Statsd

Grafana

All together now

Conclusioni



- ► I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- ▶ Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- ▶ Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



- I sistemi di monitoraggio (e logging, e alerting), servono e serviranno sempre di più.
- Sono sistemi trasversali, soprattutto per quanto riguarda la possibilità di integrare dati applicativi (*Dev*) e di sistema (*Ops*). Integrazione senza la quale certe attività di debug possono essere impossibili.
- Esistono soluzioni moderne per costruire sistemi modulari ed evolvibili.
- Rifuggete la tentazione di adottare un sistema monolitico: usate soluzioni basate su software libero e standard aperti.



Conclusioni

#### Domande



### Thanks & see you soon ...

```
Grazie dell'attenzione!

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a

"Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com@gaunilone
```

\*licenza della presentazione:



### Thanks & see you soon ...

#### Grazie dell'attenzione!

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a. "Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com @gaunilone

#### \*licenza della presentazione:



Conclusioni

### Thanks & see you soon ...

```
Grazie dell'attenzione!
```

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a. "Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com @gaunilone

\*licenza della presentazione:



### Thanks & see you soon . . .

\*licenza della presentazione:

```
Grazie dell'attenzione!

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a.

"Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com@gaunilone
```

http://creativecommons.org/licenses/by-sa/4.0/

SIODEC Subdeat I'll Infrastructure

### Thanks & see you soon . . .

\*licenza della presentazione:

```
Grazie dell'attenzione!

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a.

"Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com@gaunilone
```



### Thanks & see you soon ...

```
Grazie dell'attenzione!

IDI2016 Incontro DevOps Italia 2016, 1 Aprile 2016 a.k.a.

"Non è uno scherzo", a Bologna.

WWW http://www.incontrodevops.it/

Twitter @incontrodevops

Me m@biodec.com@gaunilone

*licenza della presentazione:
http://creativecommons.org/licenses/by-sa/4.0/
```