“big data” & the future of devops

ram c singh
@ramcsingh
+00 1 202 695 2259
ram@10io.co

In 2008, the average revenue costs of an unplanned application outage was estimated to be $2.8M per hour.

IBM Global Services

59% of Fortune 500 companies experience a minimum of 1.6 hours of downtime per week.

Dunn & Bradstreet

yearly cost metrics	best-in-class	median	laggards
# of business interruption events	0.9	3	3.5
duration of each event (hours)	1.3	4.7	8.4
total disruption (hours)	1.2	14.1	29.4
average cost per hour	$60,000	$110,000	$98,000
total cost	**$72,000**	**$1.55M**	**$2.88M**

Aberdeen Group, 2010

business application	estimated outage cost-per-minute
supply chain management	$11,000
e-commerce	$10,000
customer service	$3,700
atm/pos/eft	$3,500
financial management	$1,500
human capital management	$1,000
messaging	$1,000
infrastructure	$700

types of unplanned outages

- acts of nature
- acts of man
- threshold breach/system overload
- hardware/physical failure
- software/logical failure
- configuration issue

80% of unplanned outages dues to ill-planned changes by ops or dev

"Visible Ops"

"Through 2015, 80% of outages impacting mission-critical services will be caused by people and process issues, and more than 50% of those outages will be caused by change/configuration/release integration and hand-off issues."

Gartner

goals

- provision “right-sized” capacity
- minimize - or prevent - downtime

be fast
don’t make mistakes
stay out of trouble

how?

- institute changes quickly & appropriately
- maintain consistency of software & hardware
- manage changes from one place
- *never* log into an individual server
- monitor infrastructure operation

dev's job: to keep...
- the enterprise's software aligned with the business needs
- and keep the business in business

ops' job: to keep...
- keep the enterprise's infrastructure deployed and running
- and the business in business

so, devops seeks: to keep...
- the enterprise's software aligned with business requirements
- the enterprise's infrastructure running
- and the business in business

by keeping risks due to changes at minimum

infrastructure as code

- decompose “infrastructure” into modular, composable *components*
- define configuration manageable *processes* to build those components
- develop *automations* that reliably repeat those processes
- compose automations to configure and deploy *infrastructures*
- configuration manage compositions and automations

devops tools

- operating system installation
  pxe (pre-execution environment)
- configuration management
  cfengine, puppet, chef
- composition
  vagrant
- monitoring
  nagios, zenoss, ganglia, collectd, boundary, splunk, accelops + log files

is this sufficient to...?

- institute changes quickly & appropriately
- maintain consistency of software & *hardware*
- manage changes from one place
- *never* log into an individual server
- monitor infrastructure operation metrics
- use metrics to drive changes

nope. ‘cause ops is more than apps!

Hardware specifications

Virtualizations

Operating systems

Application

APIs/Services

Algorithms

Databases

Caching

Composition

Logical security/ access

Configuration management

Networks

Routing

Switching

Firewalls

Servers

Storage infrastructure

Load balancing

Transmission protocols

Firmware

Scripts

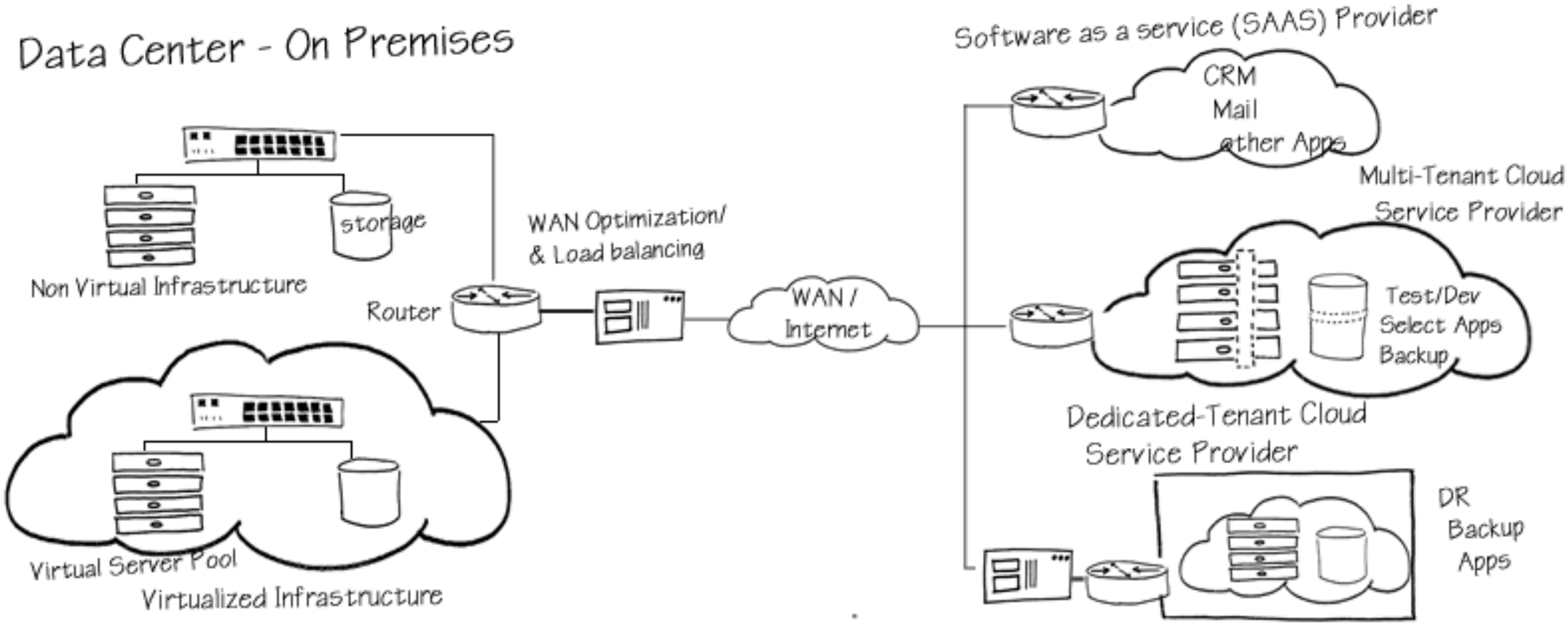
Trend analysis

Capacity planning

NOC management

Disaster recovery

conceptual hybrid data center



http://www.nuspective.com/solutions/data-center-hybrid/

infrastructures are complex

- difficult to predict when they are going to fail
- difficult to identify what has failed
- difficult to determine why it failed

which makes it difficult to determine how to recover from an unplanned outage, and prevent it from re-occurring

for infrastructure to really be code...

Hardware specifications
Virtualizations
Operating systems
Application
APIs/Services
Algorithms
Databases
Caching
Composition
Logical security/ access
Configuration management

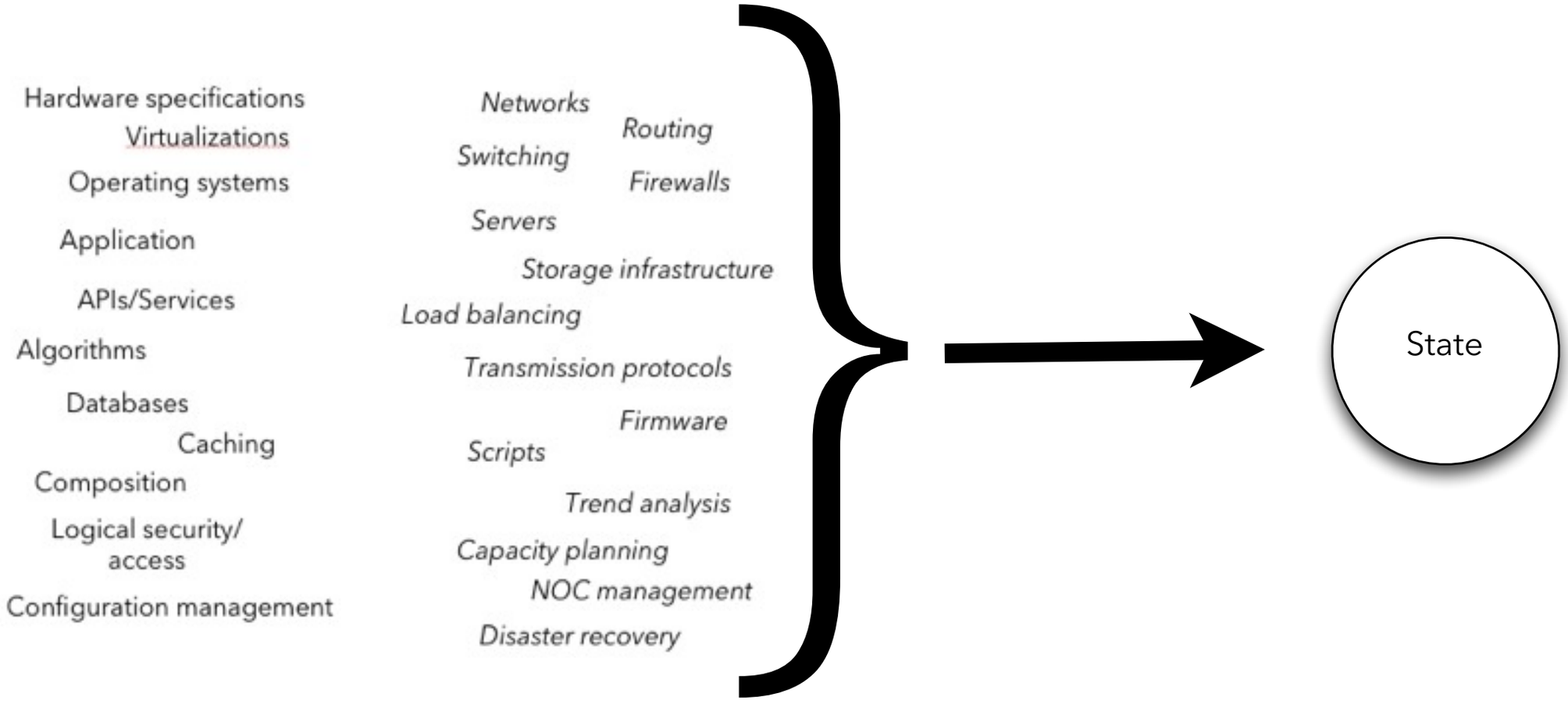
Networks
Routing
Switching
Firewalls
Servers
Storage infrastructure
Load balancing
Transmission protocols
Firmware
Scripts
Trend analysis
Capacity planning
NOC management
Disaster recovery

- decompose infrastructure into modular, composable *components*
- define configuration manageable *processes* to build those components
- develop *automations* that reliably repeat those processes
- compose automations to configure and deploy *infrastructures*
- configuration manage compositions and automations

some other tools in the ops arsenal

- network device configuration
  snmp, netconf, yang
- network device configuration management
  scripts, tail-f
- monitoring
  cisco prime, zyrion, clearaccess,
  solarwinds + log files

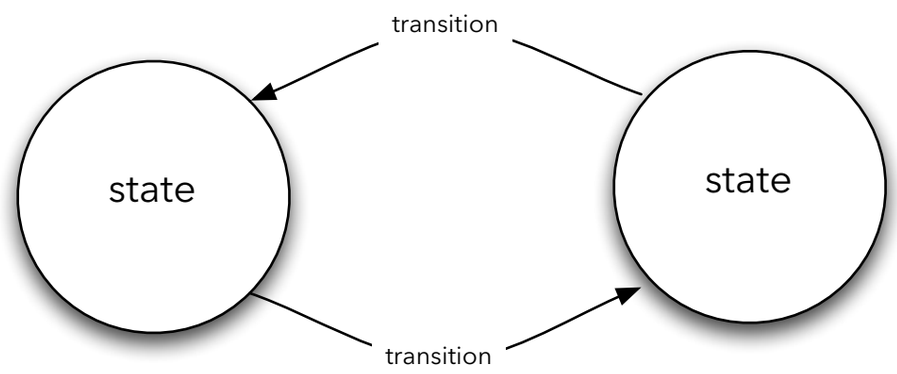
infrastructure as managed state



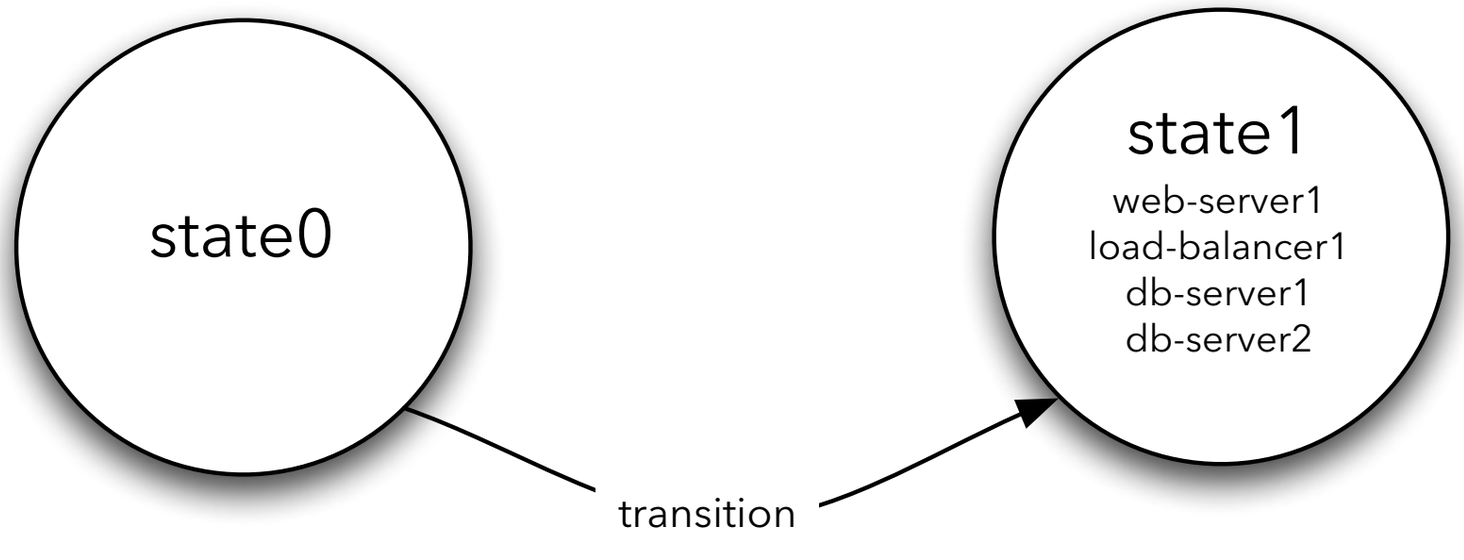
infrastructure management as state transitions

\+

performance characteristics

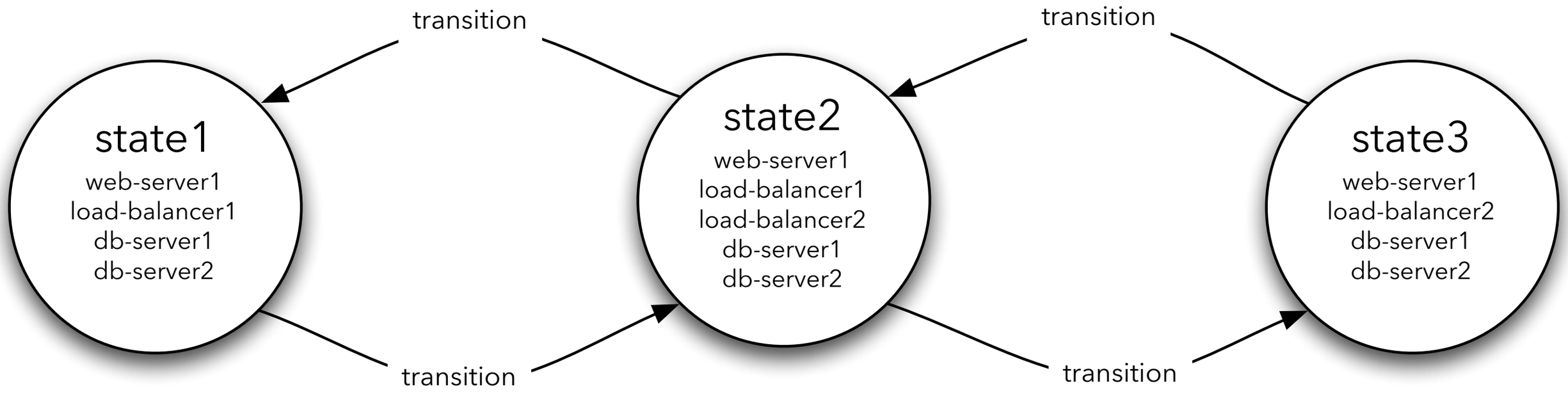


simple example - deploy web app



react to requirement for web app
- identify response
- configure & deploy web app infrastructure by
- running automations
- composed of processes that
- configure reusable components

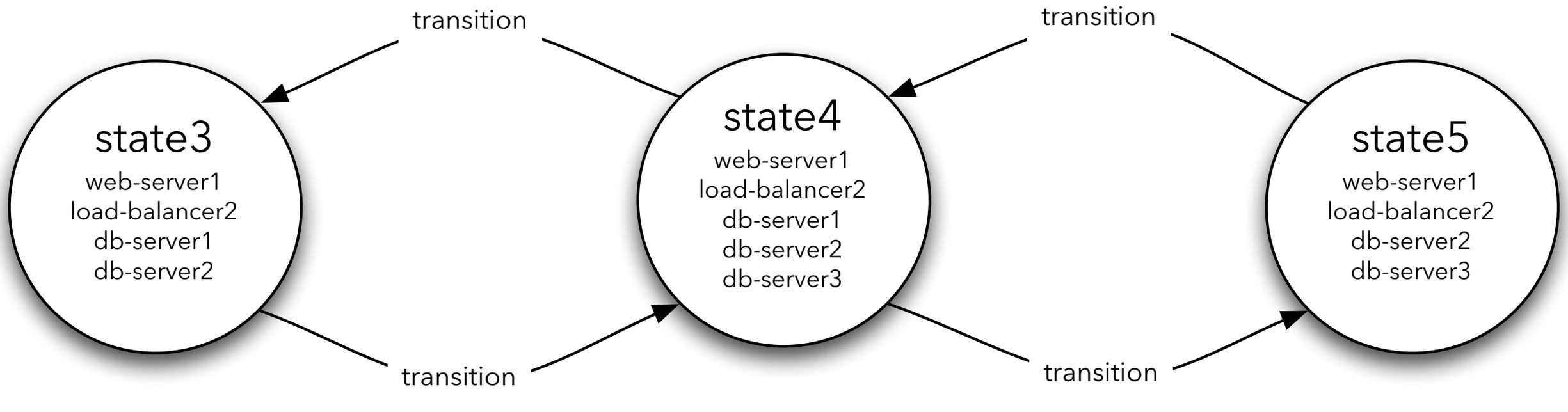
example - db performance issue



monitoring shows db performance issue

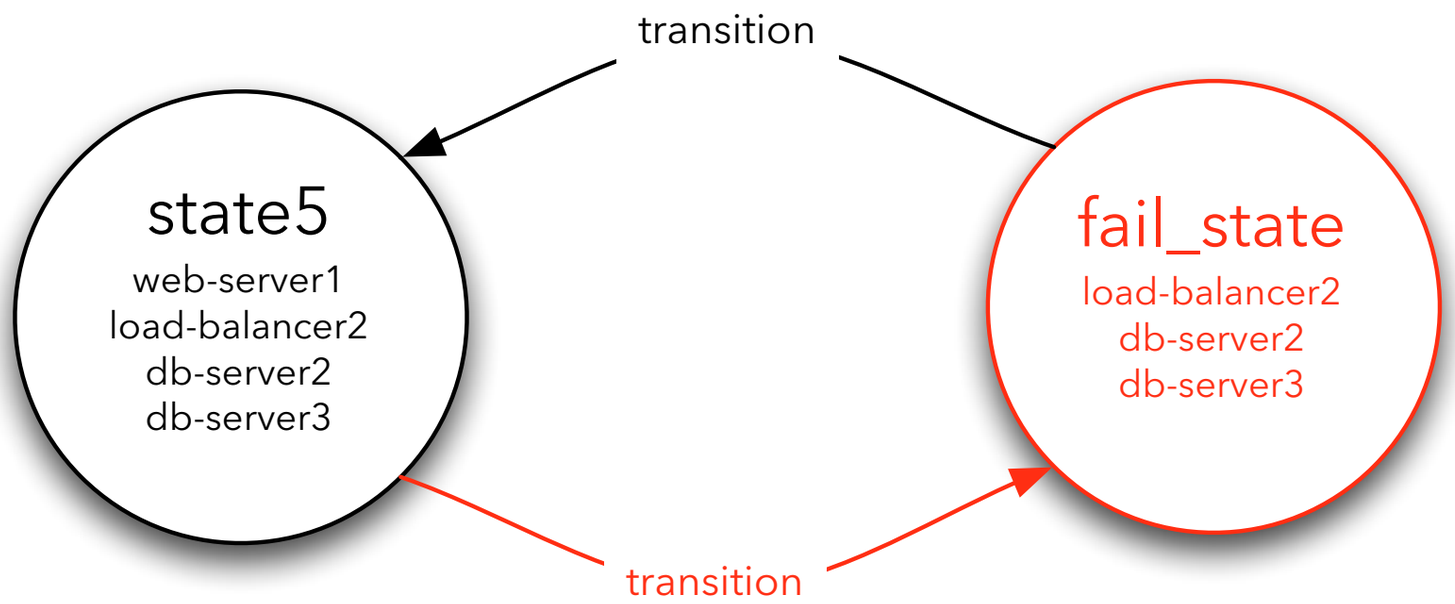
- identify response
- configure & deploy web app infrastructure by
- running automations
- composed of processes that
- configure reusable components

example - db performance issue (cont)



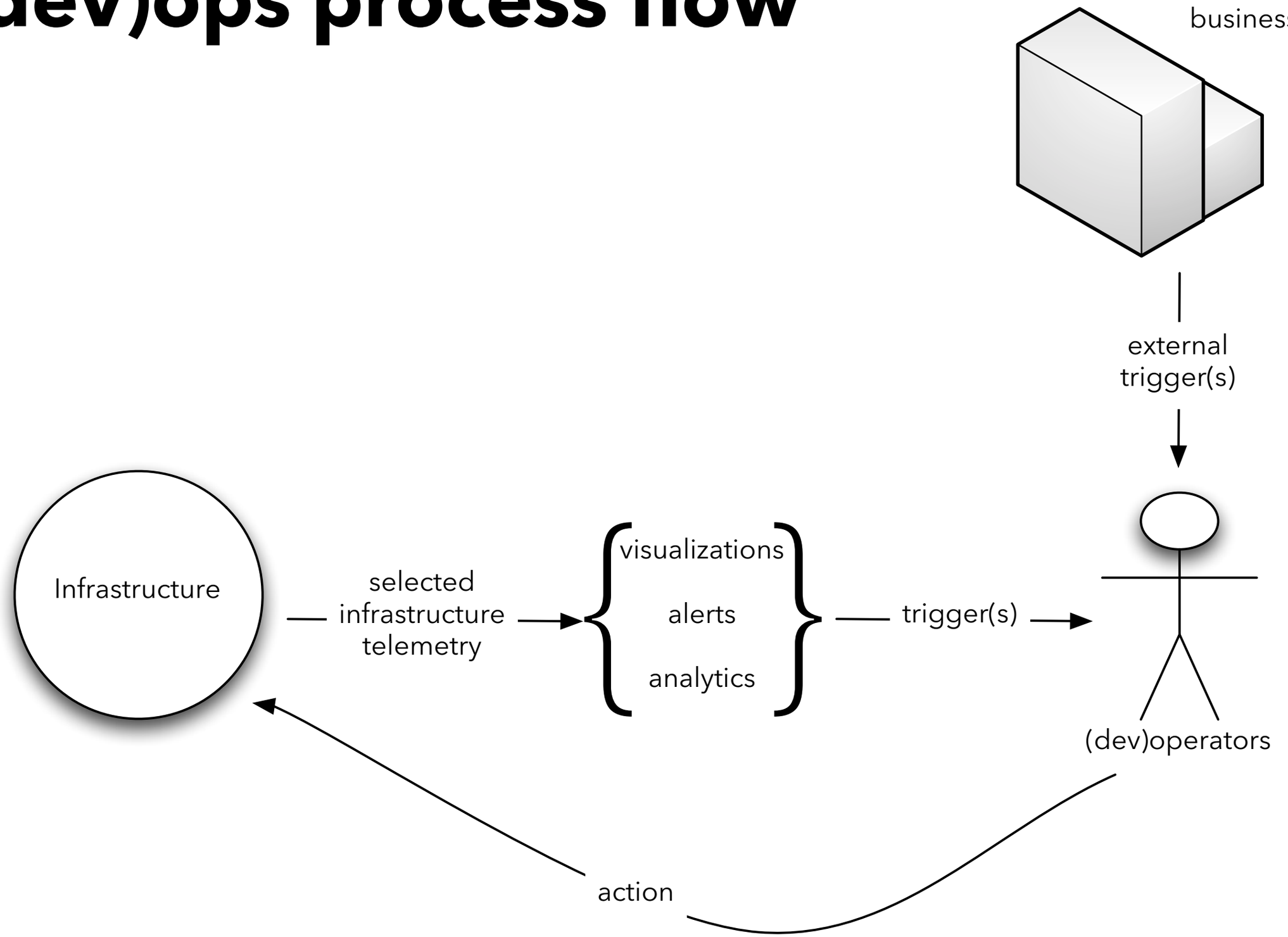
monitoring show db performance issue persists
- identify response
- configure & deploy web app infrastructure by
- running automations
- composed of processes that
- configure reusable components

what does an outage look like?



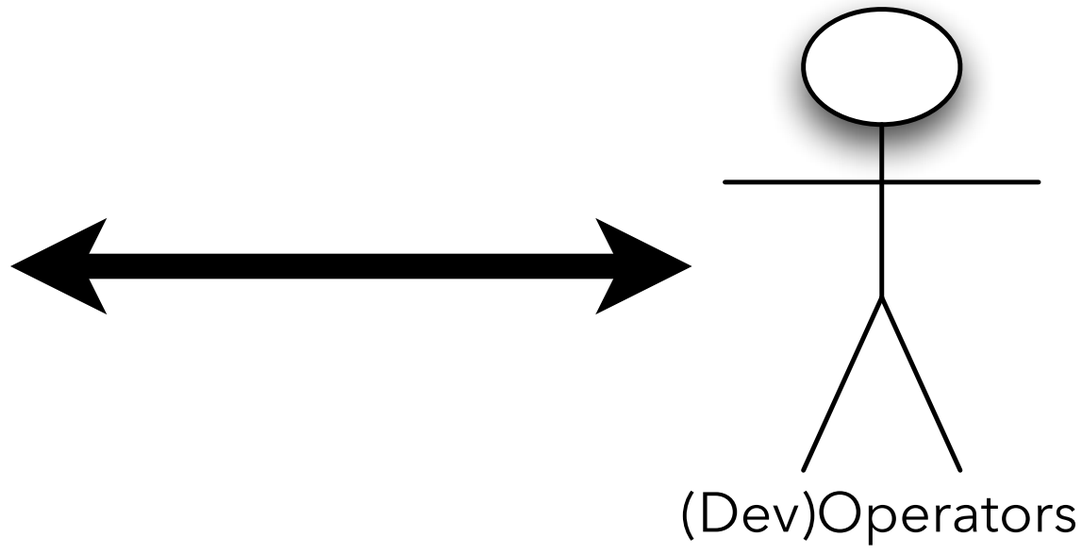
monitoring software fires alert
- identify response
- configure & deploy web app infrastructure by
- running automations
- composed of processes that
- the configure reusable components

(dev)ops process flow



devops tools still require human(s) to

- react
- identify
- configure
- deploy
- running automations
- composing
- configuring



remember this statistic?

"Through 2015, 80% of outages impacting mission-critical services will be caused by people and process issues, and more than 50% of those outages will be caused by change/configuration/release integration and hand-off issues."

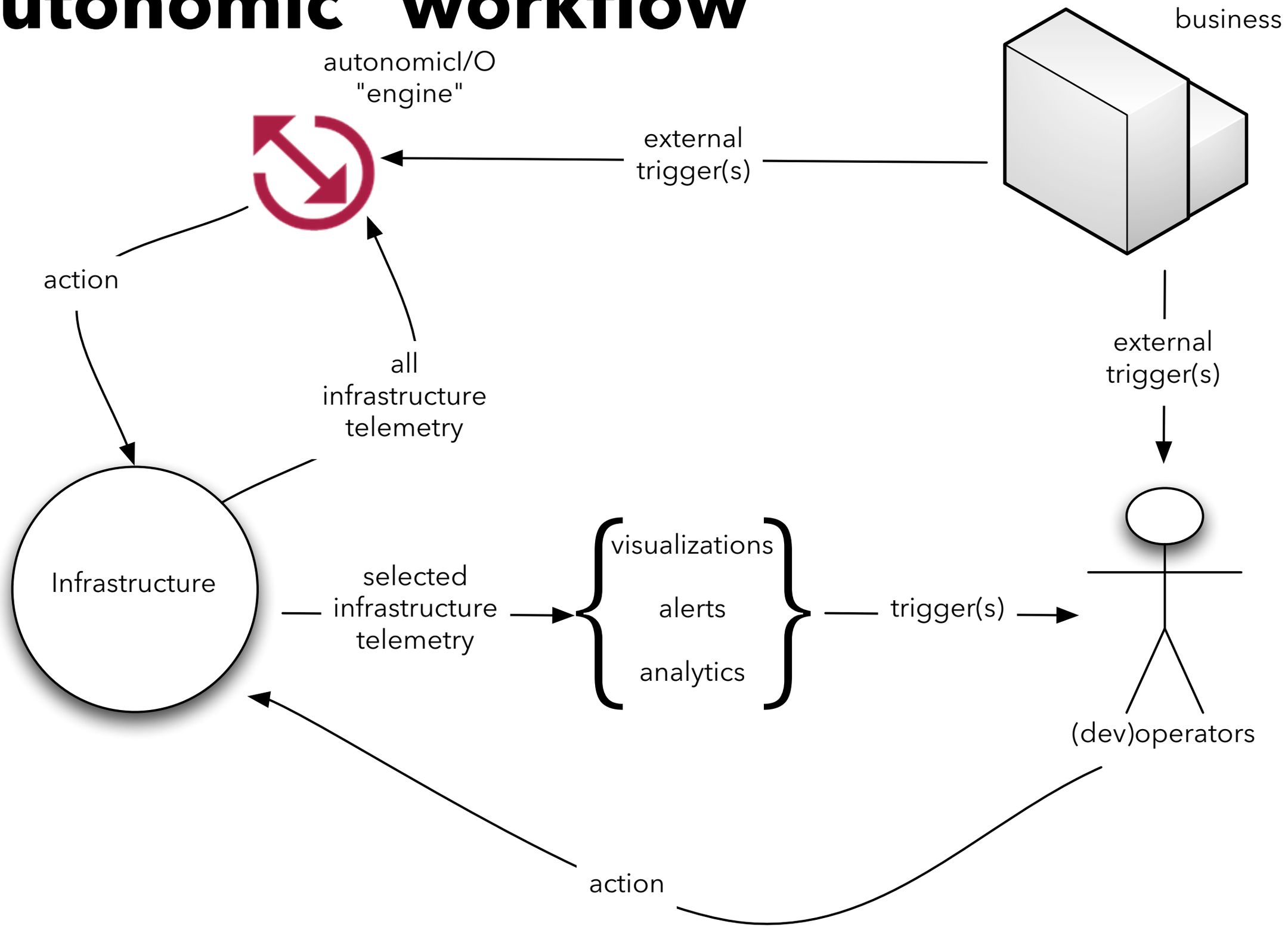
Gartner

how to mitigate human-induced unplanned outages?

get *autonomic*.

autonomic computing infrastructures can run diagnostics and checks, and compensate for any irregularities or glitches that may appear

autonomic “workflow”

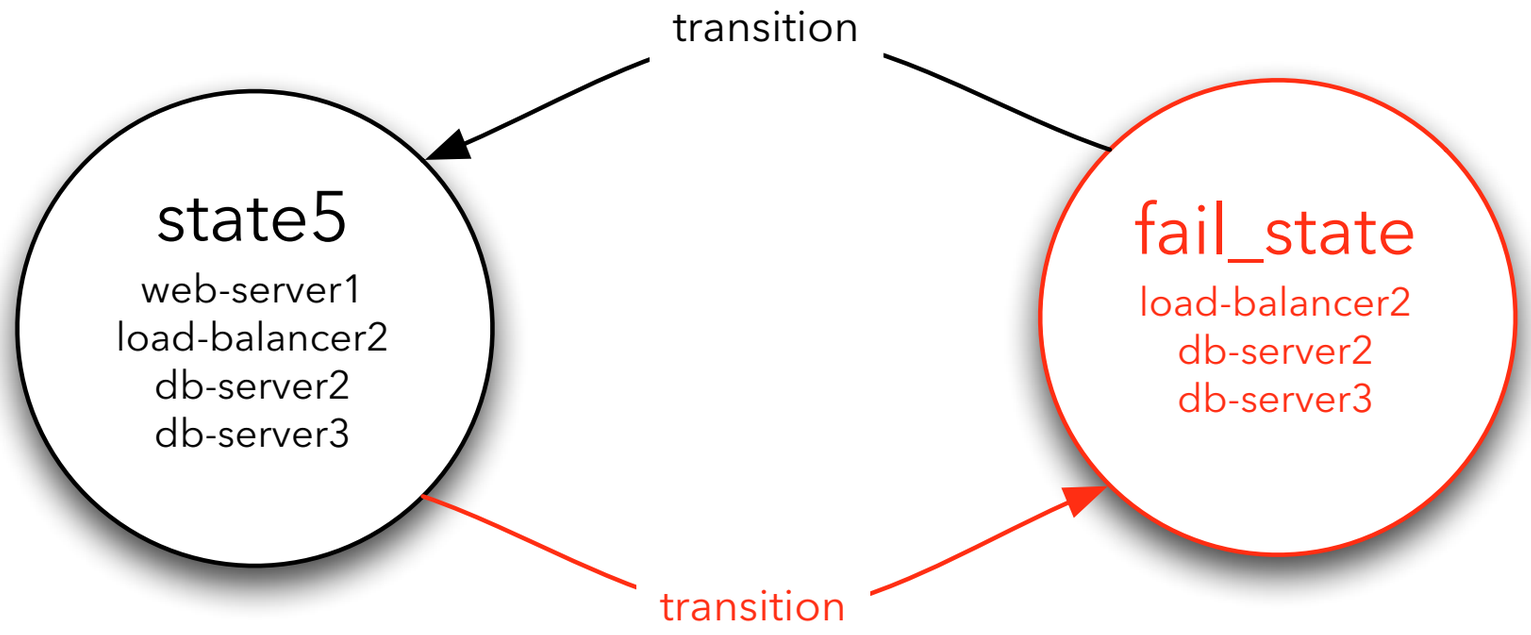


autonomicI/O

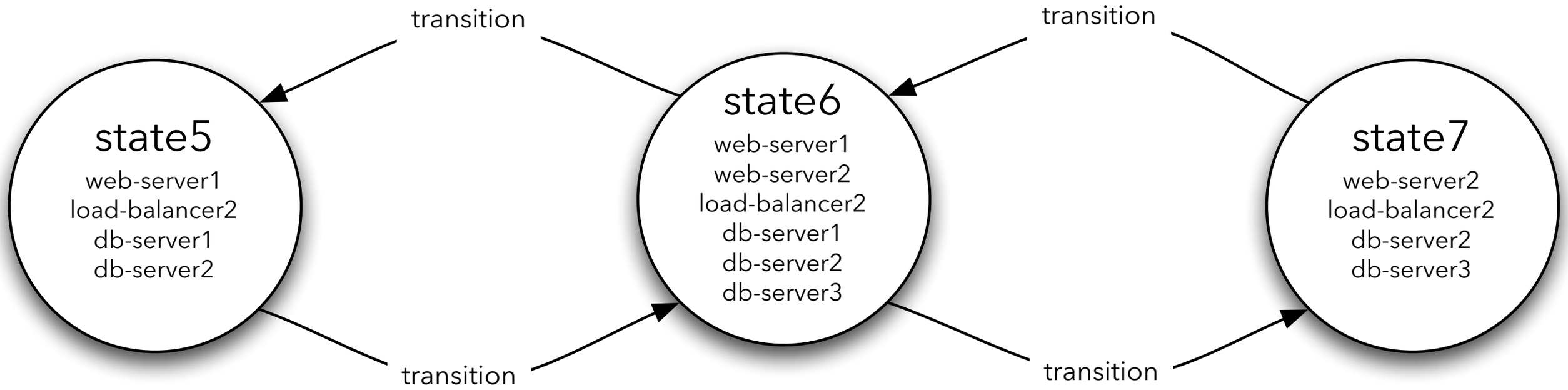
autonomically manages infrastructure by

- ingesting *infrastructure telemetry*
- identifying anomalous patterns
- triggering a transition towards another “desired state”

no more “fail_states”...



...just autonomic transitions between desired states



10io infrastructure maturity model

- basic: each element managed individually
- monitored: consolidated metrics
- reactive: faster response via automation *(devops)*
- proactive: system responds autonomicly to known patterns *(autonomicl/O v1.0)*
- predictive: system devises autonomic responses to unknown patterns *(autonomicl/O v?.?)*
- fully autonomic: system governed by business policies & performance objectives *(autonomicl/O v?.?)*

so...
the future is one where devops ***best practices*** + "big data" ***infrastructure telemetry*** + an ***autonomic engine*** are employed to create **autonomic computing infrastructures**

autonomic computing infrastructures

thanks!

Ram C Singh
@RamCSingh
+00 1 202 695 2259
ram@10io.co