



ConPaaS: A Cloud Platform for Hosting Elastic Applications

Héctor Fernández and G. Pierre
Vrije Universiteit Amsterdam

Cloud Computing Day, November 20th 2012

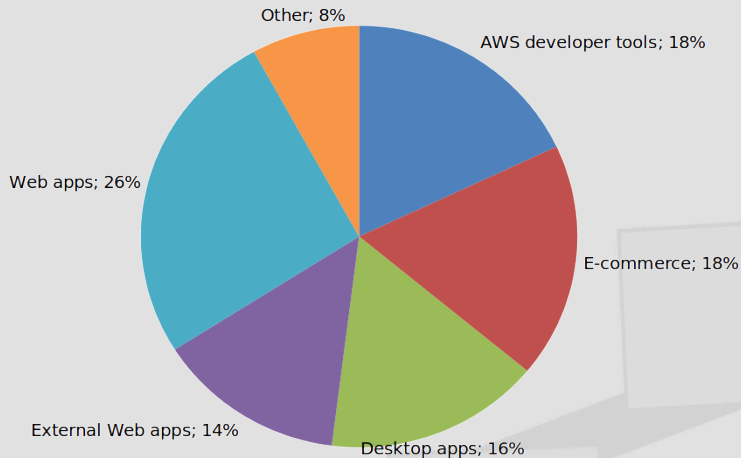


contrail is co-funded by the EC
7th Framework Programme
under Grant Agreement nr.
257438

Typical Cloud Applications (according to AWS)

- ▶ Application Hosting
- ▶ Backup and Storage
- ▶ Content Delivery
- ▶ E-Commerce
- ▶ High Performance Computing
- ▶ Media Hosting
- ▶ On-Demand Workforce
- ▶ Search Engines
- ▶ Web Hosting

Applications running at Amazon Web Services



Sample: 50 applications from the
AWS Customer App Catalog.

Many Cloud applications are alike

- ▶ Web servers
- ▶ Application servers
- ▶ Database servers
- ▶ High-performance frameworks (MapReduce, MPI, Workflows)
- ▶ ... and a few percents of miscellaneous programs

Cloud application developers often rebuild
the same types of frameworks again and again and again...

Can the Cloud help support common types of applications?

- ▶ **Infrastructure-as-a-Service** provides basic computing resources
 - ▶ Absolute flexibility: you can build anything you want
 - ▶ But it can be very complex and time consuming
 - ▶ Deployment
 - ▶ Software upgrades
 - ▶ Fault-tolerance
 - ▶ Performance monitoring
 - ▶ Resource provisioning
 - ▶ Dynamic reconfiguration orchestration
 - ▶ etc.
- ▶ **Platform-as-a-Service** provides high-level services
 - ▶ Each PaaS service targets a specific family of applications
 - ▶ Provide a simple deployment environment for applications
 - ▶ Provide high-level guarantees for applications using these services

What is ConPaaS ?



ConPaaS is an open-source runtime environment for hosting applications in Cloud environments.

- ▶ **Broad range of functionalities**
 - ▶ Application servers, databases, high-performance computing, miscellaneous
- ▶ **Fully integrated**
 - ▶ Applications can compose any set of services together
- ▶ **Easy to use but also very powerful**
 - ▶ Simple Web GUI + powerful command-line tool
 - ▶ Services are highly customizable
- ▶ **Cutting-edge SLA enforcement technologies**
 - ▶ Elasticity and resource provisioning techniques to guarantee performance at the lowest possible cost
- ▶ **Support for deployment on multiple-clouds**
 - ▶ OpenNebula, AmazonEC2, OpenStack (soon)

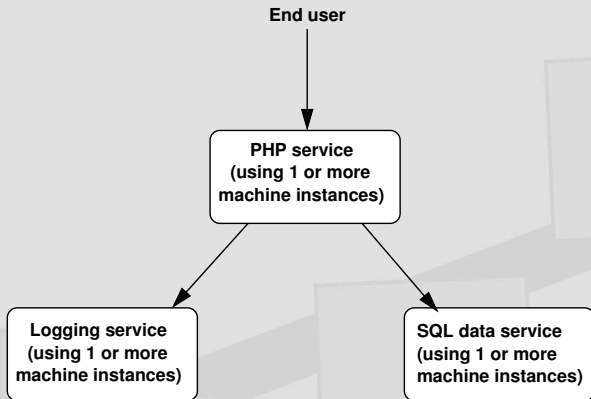
ConPaaS hosting Cloud Applications

- ▶ Fully support for the following services:
 - ▶ Web servers — static content and dynamic web applications (PHP, JSP)
 - ▶ MapReduce — for data-intensive computing
 - ▶ TaskFarming — for scientific applications
 - ▶ Databases (SQL and NoSQL) — for everybody
 - ▶ More services coming: CDN, functional testing, XtremFS, etc.
- ▶ **BUT:** You can easily build your own ConPaaS service.

ConPaaS Applications

A ConPaaS application is defined as a **composition of multiple service instances**

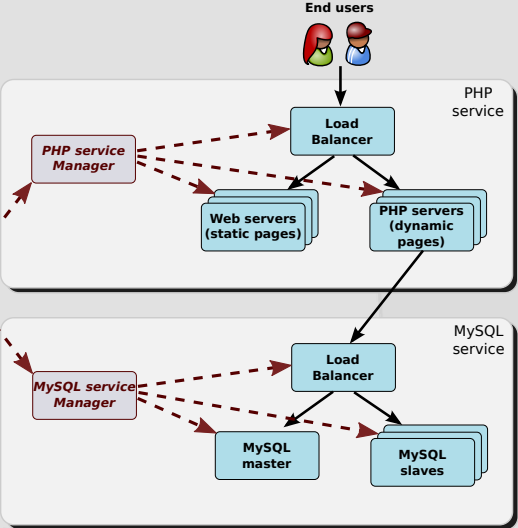
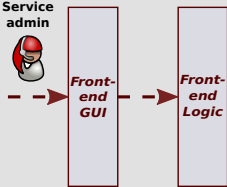
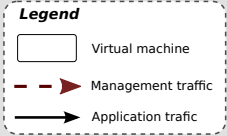
- ▶ For example: web hosting service + MySQL database + logging service (to store access logs)



Architecture of a ConPaaS service

- ▶ A ConPaaS service is implemented as **one or more virtual machine instances** dedicated to a single user
 - ▶ Single-tenant: each VM belongs to a single user
 - ▶ No VM sharing between services (even for the same user)
- ▶ ConPaaS services are **elastic**: we can grow/shrink their capacity at runtime with no service disruption
 - ▶ Horizontal provisioning: add/remove virtual machines
- ▶ ConPaaS services will support **dynamic resource provisioning**: automatic capacity adjustment to support performance guarantees at minimum cost

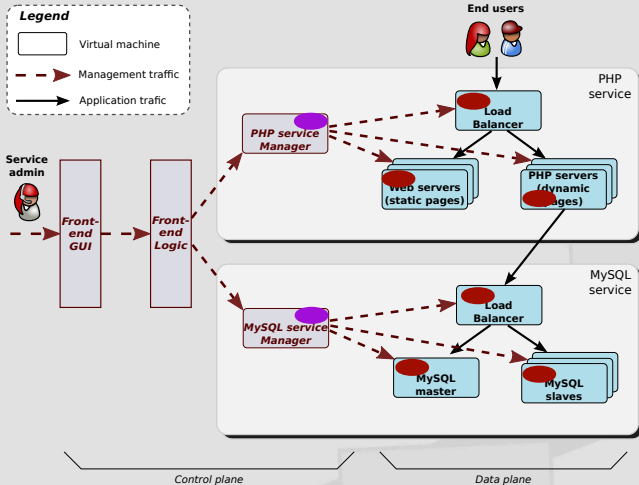
ConPaaS Organization



Control plane

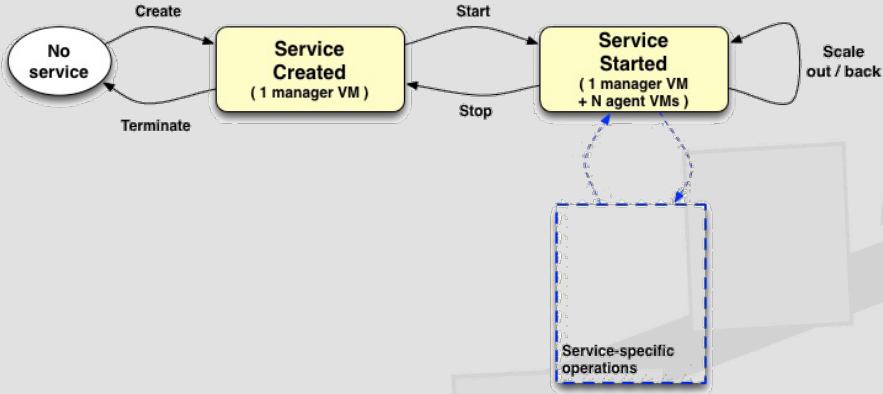
Data plane

ConPaaS Organization



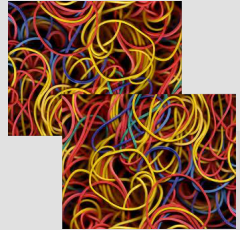
- ▶ Each manager VM contains a **manager process**
- ▶ Each agent VM contains an **agent process**

Lifecycle of a ConPaaS service

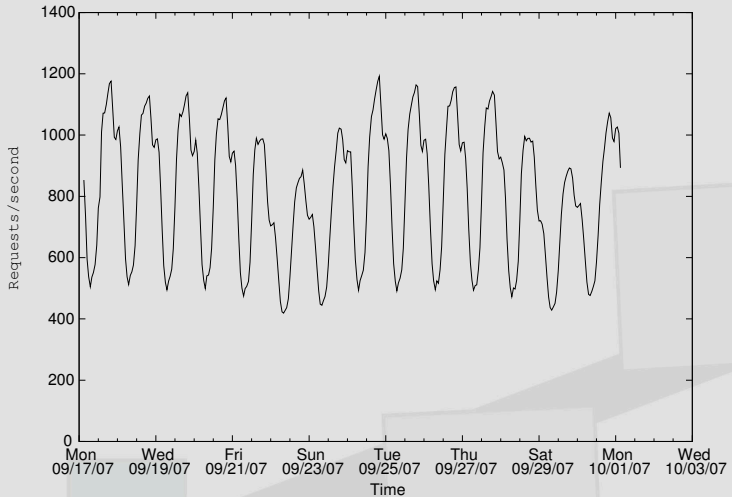


Elastic Cloud Applications

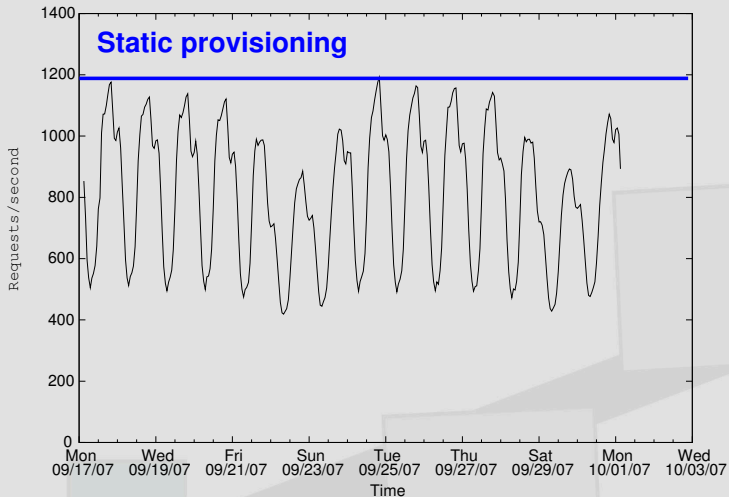
– Web applications



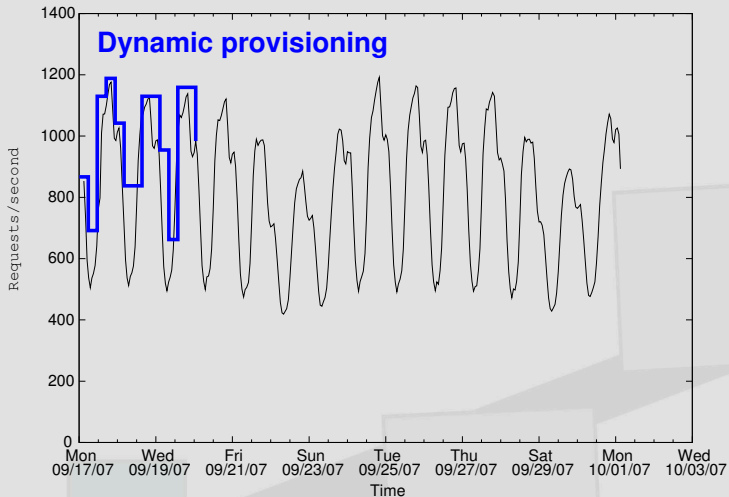
The varying capacity problem



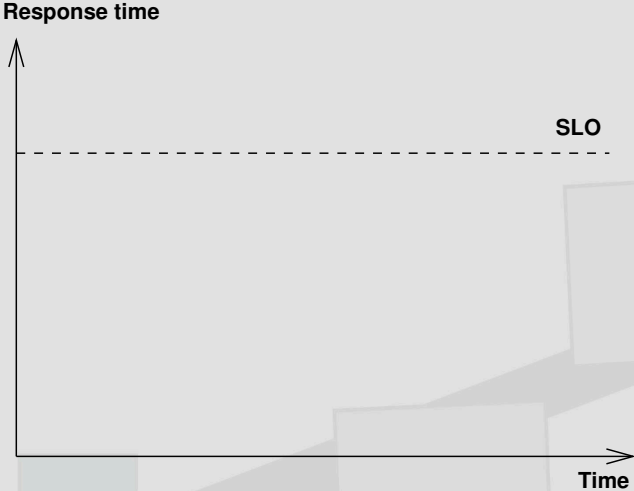
The varying capacity problem



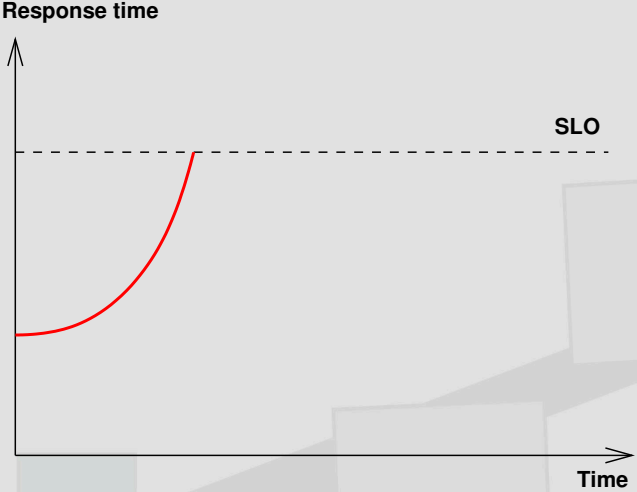
The varying capacity problem



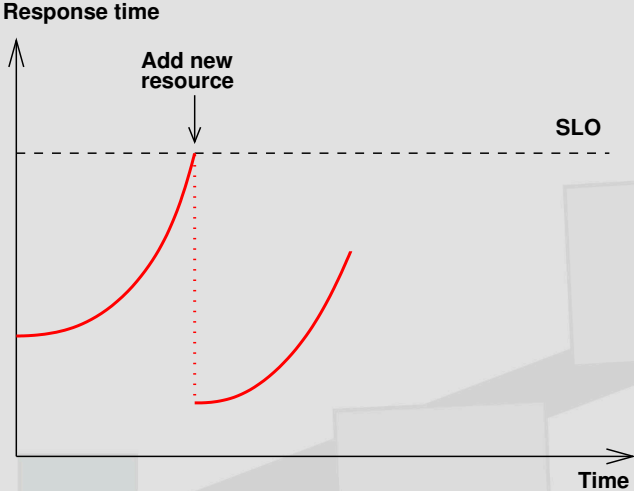
Dynamic resource provisioning



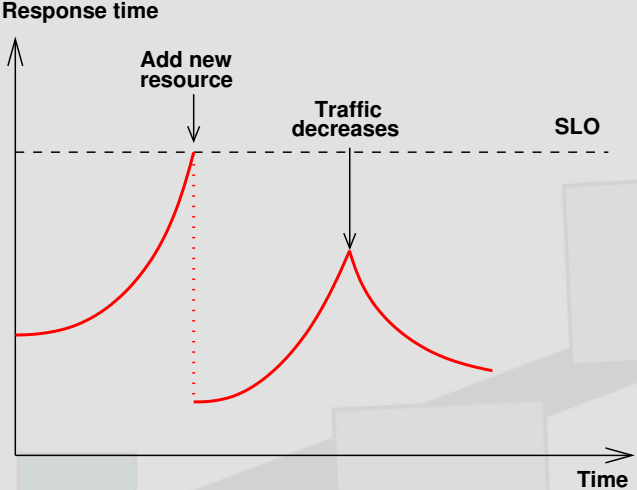
Dynamic resource provisioning



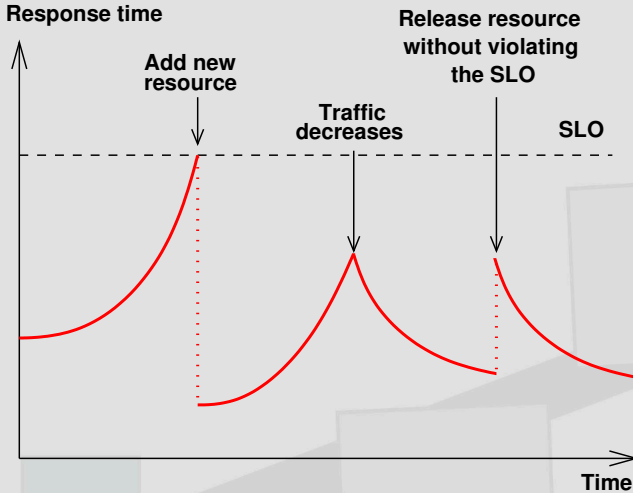
Dynamic resource provisioning



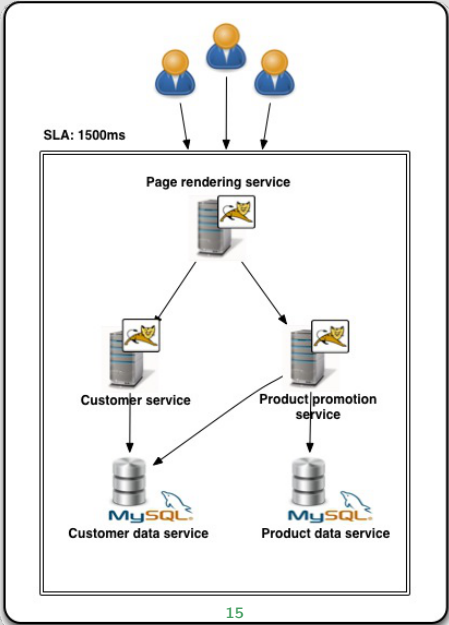
Dynamic resource provisioning



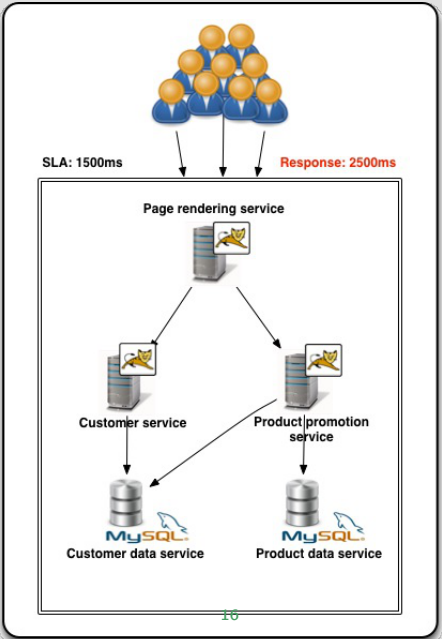
Dynamic resource provisioning



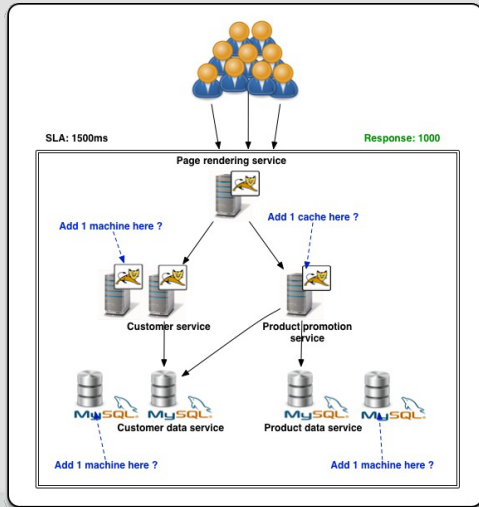
Provisioning in a multi-service Web application



Provisioning in a multi-service Web application



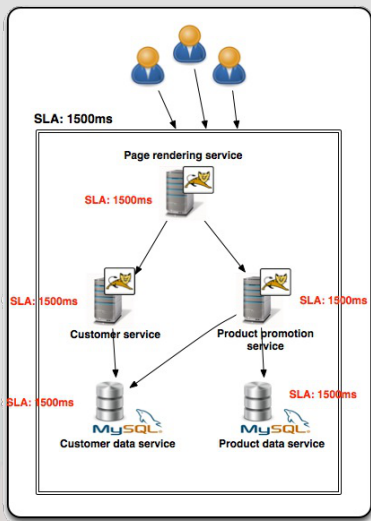
Provisioning in a multi-service Web application



When the application needs more capacity,
where should I place extra resources?

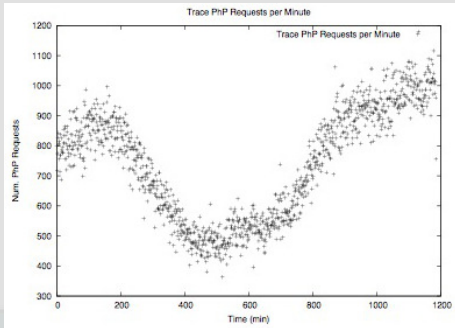
Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually



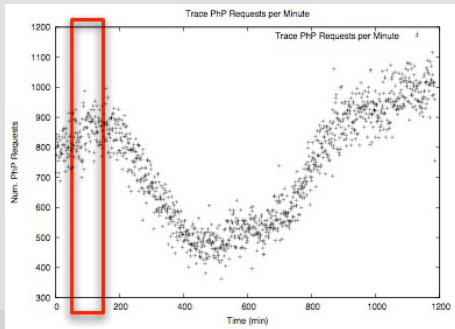
Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually
 - ▶ Scaling decisions are based on the actual workload burst



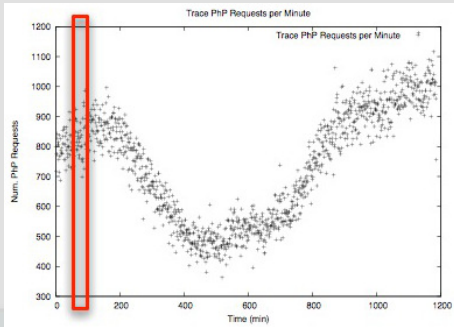
Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually
 - ▶ Scaling decisions are based on the actual workload burst



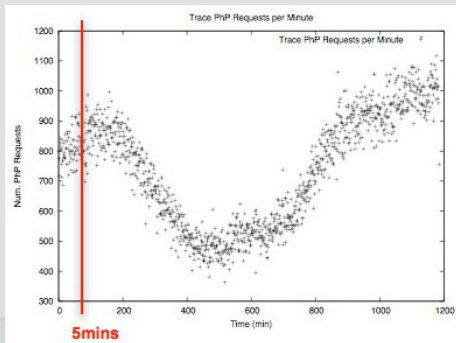
Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually
 - ▶ Scaling decisions are based on the actual workload burst



Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually
 - ▶ Scaling decisions are based on the actual workload burst



Classical solutions

- ▶ Threshold rules-based resource provisioning systems
 - ▶ Impose an SLO to each service individually
 - ▶ Scaling decisions are based on the actual workload burst
 - ▶ Examples: RightScale, OpenShift, Auto Scaling Amazon, etc ...



- ▶ **Advantages**

- 😊 Each service can be scaled in isolation
- 😊 Easily interpreted by non specialists

- ▶ **Disadvantages**

- 😞 Only system-level performance constraints (CPU usage and Resp. time)
- 😞 Easy target to temporal bursty variations in the workload
- 😞 VM performance heterogeneity
- 😞 **Overall performance is suboptimal**: SLO for backend services

ConPaaS: Auto-scaling System

- ▶ Profiling-based resource provisioning system
 - ▶ Extend the threshold rules-based system using profiling techniques
 - ▶ VM performance profiling
 - ▶ Impose an SLO to the **front-end service only**
 - ▶ Scaling decisions are based on the most recent workload history
- ▶ **Advantages**
 - 😊 Services collaborate to maintain the SLO at min. cost
 - 😊 VM performance heterogeneity
 - 😊 Application-specific performance constraints
 - 😊 Avoid flash crowds and slashdot effects
- ▶ **Disadvantages**
 - 😞 Best-effort in terms of SLA fulfillment

VM performance profiling

Purpose: To estimate the application response time that a certain VM will provide under a given workload.

Online profiling:

- ▶ **Idea:** while the application is in use, we direct a number of specified request workloads to the tested VMs and measure the response time
- ▶ **Usage:**
 - ▶ to dynamically adjust the load balancing weights of the provisioned VM's
 - ▶ to refine the conditions for scaling out/back
- ▶ **Implementation:** through a customized web load balancer

VM performance profiling

Purpose: To estimate the application response time that a certain VM will provide under a given workload.

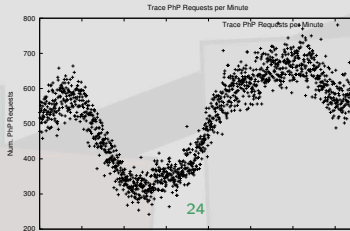
Offline profiling:

- ▶ **Idea:** to gather information about VM instances to have an initial assessment of their throughput.
- ▶ **Usage:**
 - ▶ to select the suitable set of VMs for an initial configuration
 - ▶ to define the flexible threshold ranges for each VM's instance
- ▶ **Implementation:** through training

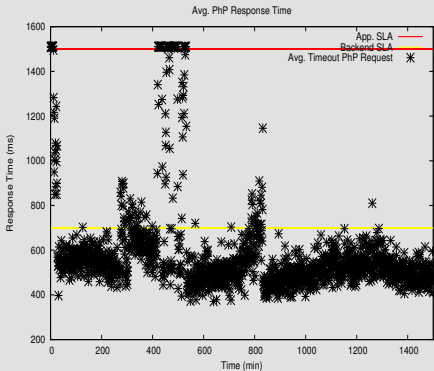
Preliminary Experimental Results

Purpose: Compare the behavioral pattern between the threshold-based and profiling-based (using the PhP web hosting service)

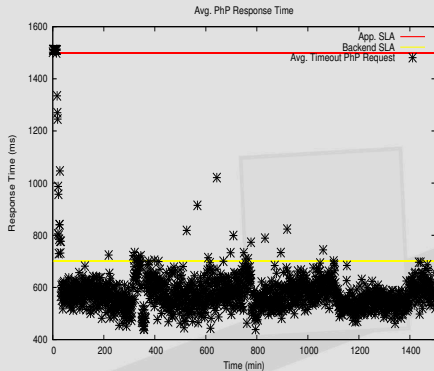
- ▶ **Application:** Wikipedia services - Mediawiki
 - ▶ English Wikipedia articles data (approx. 20Gb)
- ▶ **Monitoring:**
 - ▶ Ganglia (<http://ganglia.sourceforge.net/>)
 - ▶ Modules to monitor **web-specific metrics**
- ▶ **Testing:** Wikibench, a Wikipedia-based benchmark
 - ▶ Real access traces
- ▶ **Enviroment:** Amazon Elastic Cloud Compute



SLA Fulfillment (only PHP requests)

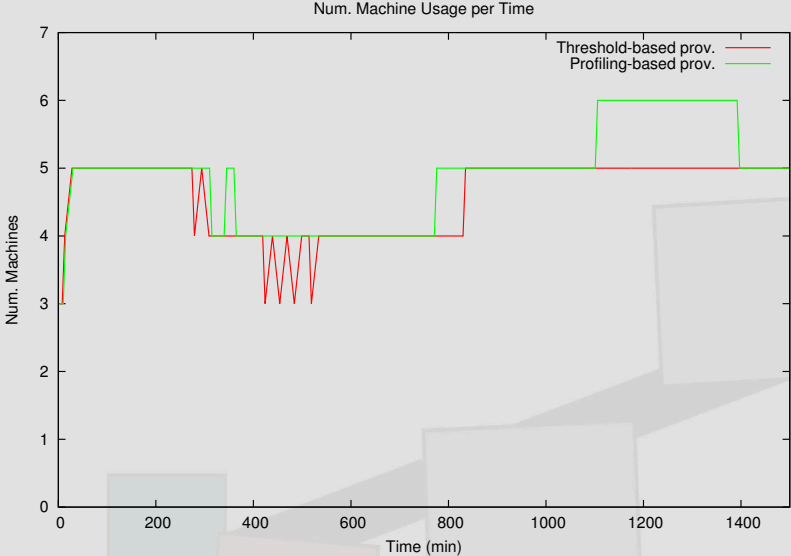


Threshold rules-based



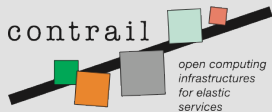
Profiling-based

Resource Consumption



Conclusion

- ▶ ConPaaS is a **platform-as-a-service environment**
 - ▶ Designed to **facilitate elastic application hosting** in the cloud
 - ▶ Auto scaling system: Trade-off between SLA fulfillment and resource consumption
 - ▶ Designed to be easily extensible
- ▶ ConPaaS addresses **two major classes of applications**:
 - ▶ Web applications
 - ▶ Scientific applications
 - ▶ Combinations of both
- ▶ **ONGOING WORK**:
 - ▶ Online profiling when adding/removing VMs
 - ▶ Offline profiling to establish flexible threshold ranges
 - ▶ Vertical scaling (up/down)
 - ▶ Cost-aware resource provisioning



contrail is co-funded by the
EC 7th Framework Programme

Funded under: FP7 (Seventh Framework Programme)
Area: Internet of Services, Software & Virtualization
(ICT-2009.1.2)

Project reference: FP7-IST-257438

Total cost: 11.29 million euro

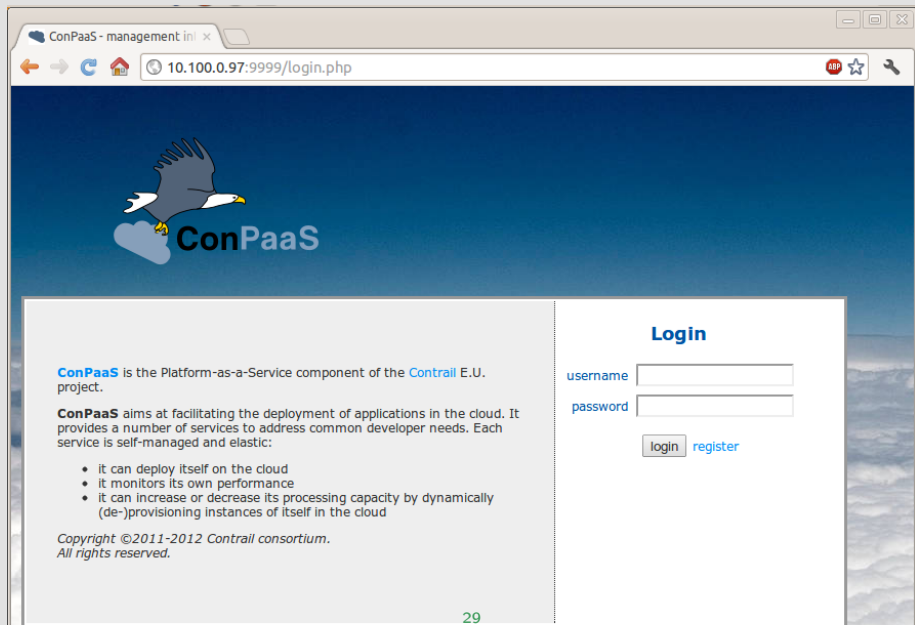
EU contribution: 8.3 million euro

Execution: From 2010-10-01 till 2013-09-30

Duration: 36 months

Contract type: Collaborative project (generic)

The ConPaaS Front-End



The screenshot shows a web browser window with the title "ConPaaS - management inl x". The address bar contains "10.100.0.97:9999/login.php". The page features the ConPaaS logo, which includes a stylized eagle and the text "ConPaaS".

ConPaaS is the Platform-as-a-Service component of the [Contrail](#) E.U. project.

ConPaaS aims at facilitating the deployment of applications in the cloud. It provides a number of services to address common developer needs. Each service is self-managed and elastic:

- it can deploy itself on the cloud
- it monitors its own performance
- it can increase or decrease its processing capacity by dynamically (de-)provisioning instances of itself in the cloud

*Copyright ©2011-2012 Contrail consortium.
All rights reserved.*

Login

username


password

[register](#)

The ConPaaS Front-End

ConPaaS - management inl x

10.100.0.97:9999/login.php



Register

ConPaaS is the Platform-as-a-Service component of the **Contrail E.U.** project.

ConPaaS aims at facilitating the deployment of applications in the cloud. It provides a number of services to address common developer needs. Each service is self-managed and elastic:

- it can deploy itself on the cloud
- it monitors its own performance
- it can increase or decrease its processing capacity by dynamically (de-)provisioning instances of itself in the cloud

*Copyright ©2011-2012 Contrail consortium.
All rights reserved.*

username

email

password

retype password

first name

last name

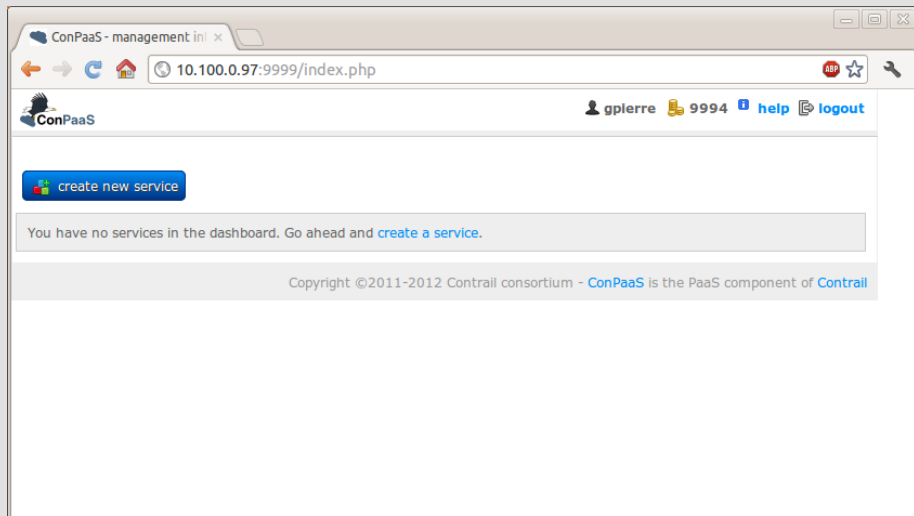
affiliation

e|o|t|e **must**

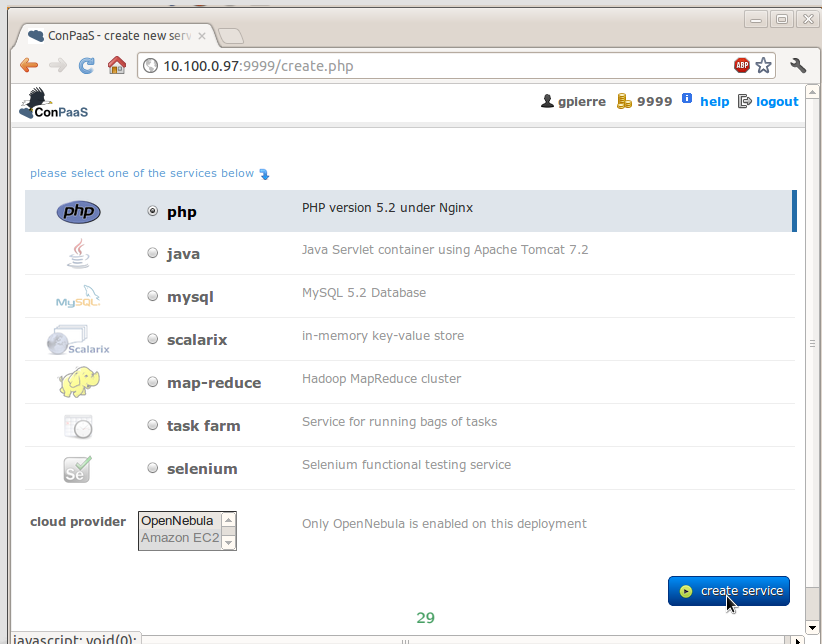
Type the two words:

reCAPTCHA™ stop spam. read books.

The ConPaaS Front-End



The ConPaaS Front-End

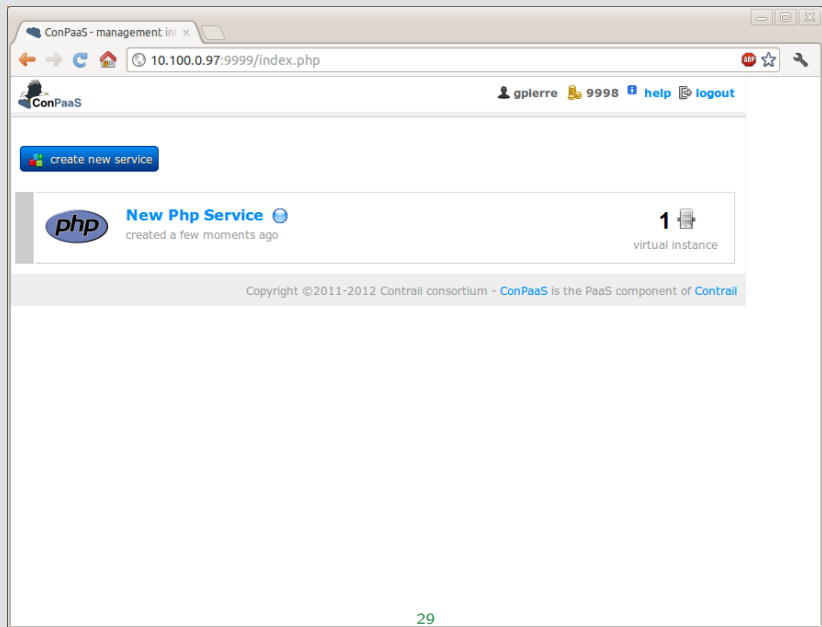


The screenshot shows a web browser window with the address bar at `10.100.0.97:9999/create.php`. The page title is "ConPaaS - create new serv...". The user is logged in as "gpierre" with ID "9999". The page content includes a prompt "please select one of the services below" and a list of services:

- php** PHP version 5.2 under Nginx
- java** Java Servlet container using Apache Tomcat 7.2
- mysql** MySQL 5.2 Database
- scalarix** in-memory key-value store
- map-reduce** Hadoop MapReduce cluster
- task farm** Service for running bags of tasks
- selenium** Selenium functional testing service

Under the heading "cloud provider", there is a dropdown menu with "OpenNebula" selected and "Amazon EC2" as an option. The text below the dropdown states: "Only OpenNebula is enabled on this deployment". A blue "create service" button is located at the bottom right of the page.

The ConPaaS Front-End



The screenshot shows a web browser window with the address bar displaying "10.100.0.97:9999/index.php". The page header includes the "ConPaaS" logo, a user profile for "gpierre" with a balance of "9998", and links for "help" and "logout". A blue button labeled "create new service" is visible. Below it, a card displays a "New Php Service" with the PHP logo, the text "created a few moments ago", and a count of "1" virtual instance. The footer contains the copyright notice: "Copyright ©2011-2012 Contrail consortium - ConPaaS is the PaaS component of Contrail".

The ConPaaS Front-End

The screenshot displays the ConPaaS management web interface. At the top, the browser address bar shows the URL `10.100.0.97:9999/service.php?sid=6`. The page header includes the ConPaaS logo, a user profile for 'gpierre', a notification count of '9998', and links for 'help' and 'logout'. A navigation link 'back to Dashboard' is visible.

The main content area features a 'New Php Service' card. It includes a 'php' logo, 'start' and 'terminate' buttons, and a 'manager log' link. The service status is 'initialized · init a few moments ago'.

Below this, a table lists running instances:

Instance ID	State	IP Address
Instance 15751	running	10.100.0.20

The 'Code management' section allows users to update the service stage. It offers two options: 'uploading archive' (selected) and 'checking out repository'. A 'Choose File' button is present, with a note 'No file chosen' and an example: 'example: .zip, .tar of your source tree'.

Under 'available code versions', a list shows 'code-default' with a 'code-default.tar' file, an 'active' status, and a 'download' link, updated 'a few moments ago'.

The 'Settings' section includes three configuration items:

- Software Version: 5.3
- Maximum script execution time: 30 seconds
- Memory limit: 128M

The ConPaaS Front-End

ConPaaS - management ini x

10.100.0.97:9999/service.php?sid=6

ConPaaS gpierrre 9997 help logout

[← back to Dashboard](#)

php **New Php Service** stop [access application →](#) · [manager log →](#)

running · started a few moments ago

2 Instances running on OpenNebula

Instance 15751 manager 10.100.0.20 running
Instance 15752 proxy web php 10.100.0.21 running

add or remove instances to your deployment

proxy web php submit

Code management [access application →](#)

you may update the stage by

- uploading archive Choose File No file chosen
example: .zip, .tar of your source tree
- checking out repository

available code versions

code-default → [code-default.tar](#) · **active** · [download](#) a few moments ago

The ConPaaS Front-End

ConPaaS - management in x

10.100.0.97:9999/service.php?sid=6

ConPaaS

gpierre 9997 help logout

← back to Dashboard

New Php Service stop

access application → · manager log →

running · started a few moments ago

2 instances running on OpenNebula

Instance 15751	manager	10.100.0.20
Instance 15752	proxy web php	10.100.0.21

add or remove instances to your deployment

+ proxy +2 web 0 php submit

Code management

you may update the stage by

- uploading archive
- checking out repository

Choose File No file chosen

example: .zip, .tar of your source tree

available code versions

code-default → code-default.tar · active · download

a few moments ago

Settings

Software Version 5.3

Maximum script execution time 30 seconds

Memory limit 128M

save

The page at 10.100.0.97:9999

no. of instances (e.g. +1, -2)

Cancel OK

The ConPaaS Front-End

The screenshot displays the ConPaaS management interface in a web browser. The browser's address bar shows the URL `10.100.0.97:9999/service.php?sid=6`. The page header includes the ConPaaS logo, a user profile for 'gpierre' with ID '9994', and links for 'help' and 'logout'. A navigation bar contains a 'back to Dashboard' link.

The main content area features a 'New Php Service' section with a 'php' logo and a 'stop' button. Below this, it indicates the service is 'running' and started 'a few moments ago'. There are links for 'access application' and 'manager log'.

A table lists five instances running on OpenNebula:

Instance ID	Status	IP Address
Instance 15751	running (manager)	10.100.0.20
Instance 15752	running (proxy)	10.100.0.21
Instance 15753	running (web)	10.100.0.22
Instance 15754	running	10.100.0.23
Instance 15755	running (php)	10.100.0.24

Below the table, there is a section for adding or removing instances to the deployment, with buttons for 'proxy', 'web', 'php', and 'submit'.

The 'Code management' section includes a link to 'access application' and options to update the stage by 'uploading archive' (with a 'Choose File' button) or 'checking out repository'. It also shows 'available code versions' with 'code-default' as the active version.

At the bottom of the page, the number '29' is displayed, and a 'Settings' link is visible in the footer.

The ConPaaS Front-End

ConPaaS - management [in] x

10.100.0.97:9999/service.php?sid=6

ConPaaS gplerre 9994 [help](#) [logout](#)

[back to Dashboard](#)

php **New Php Service** [stop](#) [access application](#) [manager log](#)

running - started a few moments ago

5 instances running on OpenNebula

Instance 15751 manager	10.100.0.20
running	
Instance 15752 proxy	10.100.0.21
running	
web Instance 15753 running	10.100.0.22
Instance 15754 running	10.100.0.23
Instance 15755 php	10.100.0.24
running	

add or remove instances to your deployment

[proxy](#) [web](#) [php](#) [submit](#)

Code management [access application](#)

you may update the stage by

[uploading archive](#) [Choose File](#) No file chosen
example: .zip, .tar of your source tree

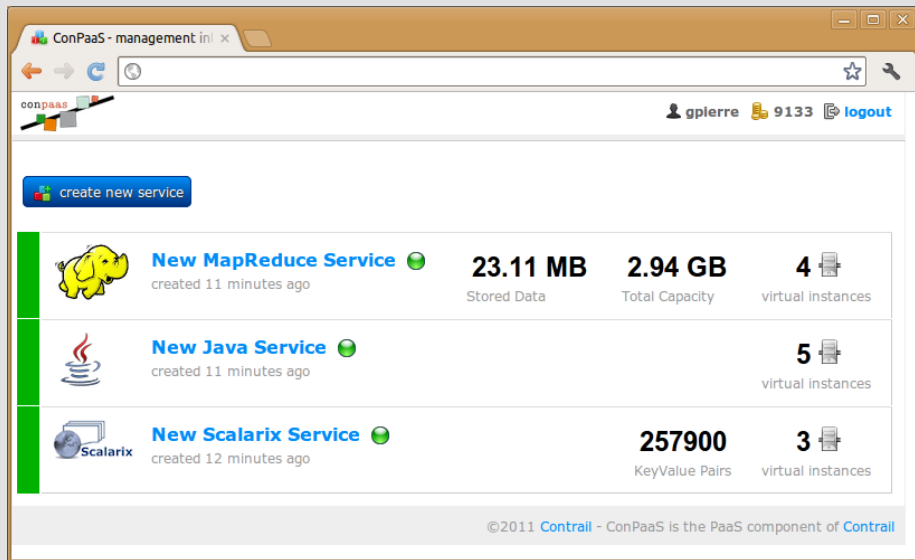
or by

[checking out repository](#)

available code versions

code-y3gOfI wbas.tar set active download	a few moments ago
code-default code-default.tar active download	a few moments ago

The ConPaaS Front-End



The screenshot shows a web browser window titled "ConPaaS - management inl x". The browser's address bar is empty. The page header includes the "conpaaS" logo on the left and the user "gpierre" with a balance of "9133" and a "logout" link on the right. A blue button labeled "create new service" is positioned above a table of services. The table lists three services, each with a status indicator (a green circle) and a printer icon. The services are: "New MapReduce Service" (created 11 minutes ago, 23.11 MB stored data, 2.94 GB total capacity, 4 virtual instances), "New Java Service" (created 11 minutes ago, 5 virtual instances), and "New Scalarix Service" (created 12 minutes ago, 257900 KeyValue Pairs, 3 virtual instances). The footer contains the copyright notice: "©2011 Contrail - ConPaaS is the PaaS component of Contrail".

Service Name	Created	Stored Data	Total Capacity	Virtual Instances
New MapReduce Service	created 11 minutes ago	23.11 MB	2.94 GB	4
New Java Service	created 11 minutes ago			5
New Scalarix Service	created 12 minutes ago		257900	3