A novel class of frameworks for scalable web services in cloud environments

Thomas Fankhauser, Qi Wang, Ansgar Gerlicher, Christos Grecos, Xinheng Wang

University of the West of Scotland
Stuttgart Media University

fankhauser@hdm-stuttgart.de
background

traffic

\[ t \]
challenges
challenges

+ modularized and distributed web applications
  who manages the distribution components?

+ application logic vs. hosting logic
  how much does the app need to know?

+ scaling considerations
  when to implement scaling?

+ performance prediction
  how much of what components are / would be needed?
There is a lot of relevant research for each component

+ but, we propose to combine those complementary components to a predictable, composed system

+ general concept
  
  web scaling frameworks

+ prototype
  
  mathematical model and empirical data
web scaling frameworks
web scaling frameworks

- caching
- queueing
- sharing
- events
- error handling
- scaling
- replication
- validation
- data model
- logic
- user interface
- navigation
web scaling frameworks

web scaling framework

web application framework

http
web scaling frameworks

+ take over scaling
  separate application logic from hosting logic

+ predict and manage performance
  monitor and control

+ connect to existing web application frameworks
  http as interface, not a replacement
prototype
prototype
processing sub-system

load balancer → server → cache → queue → worker → app

+ modifying requests
POST, PUT, DELETE, …
prototype
read sub-system

load balancer

server

cache

worker

app

+ read-only requests

GET, HEAD

events
Caching everything is impossible

+ but, for most applications it isn’t…

+ application design matters
  design for cacheability

+ fast cloud storage is available
  storage is cheaper than compute units

+ post-processing
  mechanism that keeps resource dependencies updated
+ worker and app are on the same host
  connect web scaling framework and web application framework

+ worker offers interface to app
  register dependency, push content, …

+ application declares resource dependencies
  synchronous and asynchronous dependencies

+ worker ensures updates of dependencies
  optimises and resolves update tree
prototype
post-processing example

+ app: create blog post dependencies
  synchronous: /index
  asynchronous: /sitemap

+ worker: POST /posts
  1. sends request to app
  2. receives and stores sync. and async. dependencies
  3. pushes updates to the cache
  4. recursively resolves sync. dependencies
  5. forwards response to event system … client
  6. recursively resolves async. dependencies
evaluation
evaluation

+ mathematical model
  component delays and sub-systems

+ cache / processing ratio (CPR)
  traffic distribution ratio between 1 and 0

+ scaled version vs. normal version
  web scaling framework + web application framework vs. web application framework

+ empirical data collection
  single machine scope and multi-machine scope
evaluation
+ mathematical model: analytical prediction

normal version does not consider CPR

on a single machine

all components on same host

worst case

best case

on multiple machines

44% fewer machines
**evaluation**

+ **empirical data collection**
  
  *normal version vs. scaled version - single machine scope*

### $V_n$: normal version vs. $V_s$: scaled version

+ 81 parameter tuples
  
  $cpr, da, s, u$

+ expected the $cpr$ to be highly influential
  
  $V_s$ expected to be better for tuples where $CPR = 1.0$
  $V_n$ expected to be better for tuples where $CPR = (0.5, 0.0)$

+ hypothesis: In 33% $V_s$ performs better than $V_n$
  
  accepted with a result of 37%

### model vs. data

97.6% prediction fit

CPR = (1.0, 0.5, 0.0)
$da = (0.0, 0.5, 1.0)$
$s = (25, 50, 100)$
$u = (0, 5, 10)$

$RMSE = 232$

(Predicted - Measured) RPS
in progress
in progress

+ empirical data collection
  multi-machine cloud scope
  raspberry pi cluster of 42 machines

+ further implementations
  web scaling frameworks
thank you!
web scaling frameworks

A novel class of frameworks for scalable web services in cloud environments

Thomas Fankhauser, Qi Wang, Ansgar Gerlicher, Christos Grecos, Xinheng Wang

University of the West of Scotland
Stuttgart Media University

fankhauser@hdm-stuttgart.de