## Abschlussklausur

### Cluster-, Grid- und Cloud-Computing (CGC)

15. Juli 2010

Name:
Vorname:
Matrikelnummer:
Studiengang:

#### Hinweise:

- Tragen Sie zuerst auf allen Blättern (einschließlich des Deckblattes) Ihren Namen, Ihren Vornamen und Ihre Matrikelnummer ein. Lösungen ohne diese Angaben können nicht gewertet werden.
- Schreiben Sie die Lösungen jeder *Teil*aufgabe auf das jeweils vorbereitete Blatt. Sie können auch die leeren Blätter am Ende der Heftung nutzen. In diesem Fall ist ein Verweis notwendig. Eigenes Papier darf nicht verwendet werden.
- Legen Sie bitte Ihren *Lichtbildausweis* und Ihren *Studentenausweis* bereit.
- Als *Hilfsmittel* sind ein selbstständig, einseitig beschriebenes DIN-A4-Blatt und Taschenrechner zugelassen.
- Mit Bleistift oder Rotstift geschriebene Ergebnisse werden nicht gewertet.
- Die Bearbeitungszeit dieses Teils der Abschlussklausur beträgt 60 Minuten.
- Stellen Sie sicher, dass Ihr Mobiltelefon ausgeschaltet ist. Klingelnde Mobiltelefone werden als Täuschungsversuch angesehen und der/die entsprechende Student/in wird von der weiteren Teilnahme an der Klausur ausgeschlossen!

### Bewertung:

1)	2)	3)	4)	5)	6)	7)	$oldsymbol{\Sigma}$	Note

#### Abschlussklausur

## Cluster-, Grid- und Cloud-Computing (CGC)

15.7.2010 M.Sc. Christian Baun

#### Aufgabe 1 (4 Punkte)

In der Vorlesung wurde behauptet, dass Cloud-Computing und Grid-Computing nicht das gleiche sind. Nennen Sie vier offensichtliche Unterschiede zwischen einer Cloud und einem Grid.

#### Aufgabe 2 (5+5 Punkte)

Im LHC Computing Grid fallen jährlich 15 Petabyte Daten an, die gespeichert werden müssen. Wie hoch wäre ein Stapel, wenn zur Speicherung

- a) CDs verwendet würden (Kapazität: 625 Megabyte, Dicke: 1,1 mm)?
- b) Festplatten verwendet werden (Kapazität: 2 Terabyte, Dicke: 2 cm)?

#### Aufgabe 3 (4+4 Punkte)

- a) Ordnen Sie die Cloud-Dienste-Kategorien SaaS, IaaS, PaaS und HuaaS den Ebenen zu.
- b) Ordnen Sie die folgenden kommerziellen Cloud-Angebote den Ebenen zu:
  - Google App Engine
  - Amazon Elastic Compute Cloud
  - Google Docs
  - Amazon Mechanical Turk

#### Aufgabe 4 (4+2 Punkte)

- a) Amazon Web Services (AWS)
  - Erklären Sie die beiden Konzepte Availability Zone und Region.
  - Erklären Sie die beiden Konzepte AMI und Instanz.
- b) Google App Engine (GAE)
  - Erklären Sie die Unterschiede zwischen Datastore und Memcache.

#### Aufgabe 5 (8 Punkte)

a) Sie sollen in der Google App Engine ein Gästebuch in der Programmiersprache Python erzeugen. Glücklicherweie haben Sie eine solche Applikation bereits. Diese ist nicht nur in der Lage, neue Gästebucheinträge aufzunehmen, sondern auch die bereits vorhanden Einträge zu löschen. Leider sind wegen eines Dateisystemfehlers ein paar Zeilen verloren gegangen. Glücklicherweise konnte die Zeilen wieder hergestellt werden. Ihre Postition aber nicht. Fügen Sie in den Listings die Nummern der fehlenden Zeilen ein.

#### Aufgabe 6 (5+5 Punkte)

Ein wissenschaftliches Experiment erzeugt 20 Petabyte Daten pro Jahr.

- a) Wie lange dauert die Übertragung in einem Ethernet (LAN) mit 1000 Mbit pro Sekunde?
- b) Wie lange dauert die Übertragung über ADSL mit 16.000 kbit pro Sekunde?

#### Aufgabe 7 (2+2 Punkte)

Berechnen Sie Stromkosten. Was kostete der jährliche Dauerbetrieb (24/7) bei  $0.23 \in /kWh$  für ein Unternehmen XY mit 500 Computerarbeitsplätzen?:

- a) Szenario 1: Fat Clients (PC)
  - Elektrische Anschlussleistung pro Desktopsystem: 300 Watt
  - Elektrische Anschlussleistung pro Bildschirm: 100 Watt
- b) Szenario 2: Thin Clients mit Blade-Servern
  - Elektrische Anschlussleistung pro Thin Client: 25 Watt
  - Elektrische Anschlussleistung pro Bildschirm: 100 Watt
  - $\bullet\,$  Elektrische Anschlussleistung pro Server-Blade: 150 Watt
  - $\bullet\,$  Auf ein Server-Blade passen 25 virtuelle Desktopsysteme

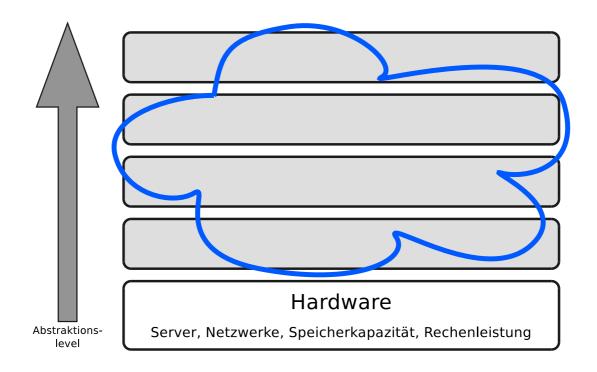
Bedenken Sie, dass es auch Schaltjahre gibt!

Name:	Vorname:	Matr.Nr.:
Aufgabe	1)	Punkte:

Name:	Vorname:	Matr.Nr.:
Aufgabe	2)	Punkte:

Name: Vorname:	Matr.Nr.:
----------------	-----------

# Aufgabe 3)



Name:	Vorname:	Matr.Nr.:
Aufgabe 4	)	Punkte:

Name: Vorname: Matr.Nr.:

## Aufgabe 5)

### Punkte: .....

### Datei app.yaml

```
application: einfaches_gaestebuch
version: 1
<hier fehlt eine Zeile>
<hier fehlt eine Zeile>
handlers:
- url: .*
  script: main.py
```

#### Datei main.py

```
#!/usr/bin/env python
# -*- coding: iso-8859-15 -*-
import os, sys
<hier fehlt eine Zeile>
from google.appengine.ext import db
from google.appengine.ext import webapp
from google.appengine.ext.webapp import template
<hier fehlt eine Zeile>
  message = db.StringProperty(required=True)
  when = db.DateTimeProperty(auto_now_add=True)
  <hier fehlt eine Zeile>
class MyHandler(webapp.RequestHandler):
  def get(self):
   shouts = db.GqlQuery('SELECT * FROM gaestebuch ORDER BY when DESC')
   values = {'shouts': shouts}
    self.response.out.write(template.render('main.html', values))
  <hier fehlt eine Zeile>
   shout = gaestebuch(message=self.request.get('message'), who=self.request.get('who'))
    shout.put()
    <hier fehlt eine Zeile>
class Loeschen(webapp.RequestHandler):
  <hier fehlt eine Zeile>
   alles_loeschen_query = gaestebuch.all(keys_only=True)
   alles_loeschen = alles_loeschen_query.fetch(300)
    <hier fehlt eine Zeile>
    self.redirect(',')
app = webapp.WSGIApplication([(',', MyHandler),
                              <hier fehlt eine Zeile>
                              debug=True)
def main():
  wsgiref.handlers.CGIHandler().run(app)
<hier fehlt eine Zeile>
  main()
```

Name: Vorname: Matr.Nr.:

## Aufgabe 5 – Fortsetzung) Punkte:

#### Datei main.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"</pre>
     "http://www.w3.org/TR/html4/loose.dtd">
<title>Einfaches G&auml;stebuch mit Google App Engine</title>
</head>
<body>
<hier fehlt eine Zeile>
  {% if shout.who %}
     <b>{{shout.who}}</b>
   {% else %}
     <b>Anonymous</b>
   <hier fehlt eine Zeile>
  sagt:
   <hier fehlt eine Zeile>
</div>
{% endfor %}
 
<hier fehlt eine Zeile>
Name:
  <input type="text" size="40"</pre>
                        name="who" value="" if="who">
 Nachricht:
  <input type="text" size="40"</pre>
                        name="message" value="" if="message">
 <input type="submit" value="Absenden">
  <input type="reset" value="L&ouml;schen">
 </form>
 
<form action="loeschen" method="post" accept-charset="utf-8">
<hier fehlt eine Zeile>
 </form>
</body>
</html>
```

Name: Vorname: Matr.Nr.:

## Aufgabe 5 - Fortsetzung) Punkte:

#### Fehlende Zeilen

```
1. api_version: 1
2. def post(self):
3. self.redirect('/')
4. class gaestebuch(db.Model):
5. def post(self):
6. ('/loeschen', Loeschen)],
7. {% endif %}
8. <b>{\{shout.message\}}</b>
9. <input type="submit" value="DB L&ouml;schen">
10. <form action="" method="post" accept-charset="utf-8">
11. import wsgiref.handlers
12. who = db.StringProperty()
13. wsgiref.handlers.CGIHandler().run(app)
14. db.delete(alles_loeschen)
15. runtime: python
16. {% for shout in shouts %}
```

Fügen Sie in den Listings die Nummern der fehlenden Zeilen ein.

Name:	Vorname:	Matr.Nr.:
Aufgabe	6)	Punkte:

Name:	Vorname:	Matr.Nr.:
Aufgabe 7)		Punkte:

Name:	Vorname:	Matr.Nr.:
vanic.	vorname.	WIGUI.IVI

# Zusatzblatt zu Aufgabe.....

Verwenden Sie dieses Blatt nur für eine Teilaufgabe! Verweisen Sie bei der zugehörigen Aufgabe gut sichtbar auf dieses Blatt!