Engineering human-based services in elastic systems

Hong-Linh Truong
Distributed Systems Group, TU Wien

truong@dsg.tuwien.ac.at
http://dsg.tuwien.ac.at/staff/truong
@linhsolar
What this lecture is about?

- Motivating scenarios
- Human service units
- Provisioning and employing human service units – frameworks
Scenario

Predictive maintenance company

- Offers services for handling IoT Data
- Offers services for big, data analytics
- Offers services for complex problem solving using human experts

IoT Cloud Platform

Data Analytics Platform

Expert Provisioning Platform

Sensors

Chillers

<<send data>>

<<analyze data>>

<<control services>>

<<control algorithms>>

<<notify possible problem>>

<<predict and solve problems>>

<<monitor>>

<<control/configure sensors>>

ASE Summer 2016
Integrated systems of software, things and people services
Human-based services for solving complex problems (2)

But how to program human-based services and software-based services together?
Example: some common tasks in data analytics

- Labeling
- Annotating
- Cleansing
- Evaluating data/content
- Detecting patterns
- Classifying objects
- Steering analytics

Human activities in Data Analytics
Human service units in data analytics -- functions

- Evaluating: is the quality of picture good?
- Classifying: is it a man’s or a woman’s picture?
- Detecting: any unidentified object in a picture?
- Labeling: adding location information of a picture
- Cleansing: remove duplicated pictures
- Steering: the quality of picture is bad, should we continue to merge it with others?

How to model such functions for human units? E.g., with WSDL or REST?
HUMAN SERVICE UNITS
Human service units

Human acting as a „service unit“

- Functions
- Non-functional parameters
- Interaction models
- Provisioning mechanisms
- Service models
- Technical interfaces

Human service unit

ASE Summer 2016
Forms of human service

- **Individual Compute Unit**
  - An individual is treated like „a processor“ or “functional unit“. A service can wrap human capabilities to support the communication and coordination of tasks

- **Social Compute Unit**
  - A set of people and software that are initiated and provisioned as a service for solving tasks

- **Web services interfaces can be built**

- **Different pricing models and different quality models**
Human service units – provisioning mechanisms (1)

- An infrastructure can be introduced for accessing many ICUs in a crowd
  - Allow people to register their service unit capabilities
  - Facilitate communication, task bidding, retrieval and result delivery
  - Act like a marketplace: multiple providers and multiple consumers
Human service units – provisioning mechanisms (2)

- An „infrastructure-as-a-service“ for ICUs
  - Facilitate communication, task retrieval and result delivery
  - Single ICUaaS provider and multiple consumers
MTurk as an ICU provider

Mechanical Turk is a marketplace for work.
We give businesses and developers access to an on-demand, scalable workforce. Workers select from thousands of tasks and work whenever it's convenient.

1,102,549 HITs available. View them now.

Make Money by working on HITs

HITs - Human Intelligence Tasks - are individual tasks that you work on. Find HITs now.

As a Mechanical Turk Worker you:
- Can work from home
- Choose your own work hours
- Get paid for doing good work

Get Results from Mechanical Turk Workers

Ask workers to complete HITs - Human Intelligence Tasks - and get results using Mechanical Turk. Get Started.

As a Mechanical Turk Requester you:
- Have access to a global, on-demand, 24 x 7 workforce
- Get thousands of HITs completed in minutes
- Pay only when you're satisfied with the results

or learn more about being a Worker

ASE Summer 2016 13
An „infrastructure-as-a-service“ for SCUs
- Facilitate communication, task retrieval and result delivery
- Single SCUaaS provider and multiple consumers
Human service units – technical interfaces (1)

- Analytics Activity
- Web Service (REST/SOAP)
- People

- Analytics Activity
- Web page
- People
Human service units – technical interfaces (2)

1. Analytics Activity
   - Web Service (REST/SOAP)
   - Web page
   - People

2. Analytics Activity
   - Communication Services
   - Email/SMS/Instant Messaging
   - People
Human service units – interaction model

Analytics Activity

Task queue

Scheduler

ICU/SCU

Analytics Activity

Task queue

Scheduler

ICU/SCU

Analytics Activity

Task queue

Scheduler

ICU/SCU
Which are important considerations when interpreting non-functional properties for human service units?
Incorporating human units into complex processes

- How to provision and employ human compute units?
- How to select human units?
- Where to place human units in data analytics and why?
- How to monitor and test human units in data analytics?
PROVISIONING AND EMPLOYING HUMAN SERVICE UNITS-- SOME FRAMEWORKS
SELECT c.name
FROM celeb c JOIN photos p
ON samePerson(c.img,p.img)
AND POSSIBLY gender(c.img) = gender(p.img)
AND POSSIBLY hairColor(c.img) = hairColor(p.img)
AND POSSIBLY skinColor(c.img) = skinColor(p.img)

TASK gender(field) TYPE Generative:
Prompt: "<table><tr> 
<td><img src='@s'> 
<td>What is this person’s gender? 
</table>", tuple[field]
Response: Radio("Gender",
["Male","Female",UNKNOWN])
Combiner: MajorityVote
Qurk system architecture (2)

Jabberwocky approach (1)

Figure 1: Overview of Jabberwocky

Source: Salman Ahmad, Alexis Battle, Zahan Malkani, Sepandar D. Kamvar: The jabberwocky programming environment for structured social computing. UIST 2011: 53-64
Jabberwocky approach (2)

```haskell
map :name => :extract_disease_facts do |key, value|
  facts = RiskExtractor.extract (value)
  for fact in facts do
    emit (fact["disease"], fact["risk_factor"])
  end
end

reduce :name => :summarize do |key, values|
  task = SummarizeFacts.prepare
  :task_name => "Summarize disease risks: #{key}"
  task.facts = values
  task.ask do |answer|
    emit (key, answer)
  end
end
```

Source: Salman Ahmad, Alexis Battle, Zahan Malkani, Sepandar D. Kamvar: The jabberwocky programming environment for structured social computing. UIST 2011: 53-64
Automan approach

```java
import edu.umass.cs.automan.adapters.MTurk_

object SimpleProgram extends App {
  val a' = MTurkAdapter { mt =>
    mt.access_key_id = "XXXX"
    mt.secret_access_key = "XXXX"
  }

  def which_one() = a'.RadioButtonQuestion {
    q =>
      q.budget = 8.00
      q.text = "Which one of these does not belong?"
      q.options = List(
        a'.Option("oscar", "Oscar the Grouch"),
        a'.Option("kermit", "Kermit the Frog"),
        a'.Option("spongebob", "Spongebob Squarepants"),
        a'.Option("cookie", "Cookie Monster"),
        a'.Option("count", "The Count")
      )
    }
  }

  println("The answer is " + which_one()
```

Karastoyanova, Dimka; Dentsas, Dimitrios; Schumm, David; Sonntag, Mirko; Sun, Lina; Vukojevic, Karolina: Service-based Integration of Human Users in Workflow-driven Scientific Experiments. In: Proceedings of the 8th IEEE International Conference on eScience (eScience 2012)
SW4H approach (2)

- Similar concepts in collaborative working environments but integrated into workflows
- Do not discuss about where and how to select human units

Karastoyanova, Dimka; Dentsas, Dimitrios; Schumm, David; Sonntag, Mirko; Sun, Lina; Vukojevic, Karolina: Service-based Integration of Human Users in Workflow-driven Scientific Experiments. In: Proceedings of the 8th IEEE International Conference on eScience (eScience 2012)
Hybrid compute unit (HCU): a set of service units includes software-based services, human-based services and things-based services that can be provisioned, deployed and utilized as a collective on-demand based on different quality, pricing and incentive models.
Hybrid compute unit design – fundamental elements

Hong-Linh Truong, Hoa Khanh Dam, Aditya Ghose, Schahram Dustdar "Augmenting Complex Problem Solving with Hybrid Compute Units", 9th International Workshop on Engineering Service-Oriented Application (WESOA's 2013), In conjunction with ICSOC 2013, Dec 2, 2013, Berlin, Germany, (c)Springer-Verlag
## Hybrid compute unit design -- Relationships

<table>
<thead>
<tr>
<th>Relationship Type</th>
<th>HBS</th>
<th>SBS</th>
<th>TBS</th>
<th>HCU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similarity</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Composition</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Dependency</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Control Dependency</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Location Dependency</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Forwarding</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Delegation</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Social Relation</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Elasticity</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Highlights: Virtualizing Communication

- Extensible architecture
- Adapters for: email, Dropbox, REST, Android
- Integrated with WP4,6,8; API access for WP5,2
- Open source and documentation: [https://github.com/tuwiendsg/SmartCom](https://github.com/tuwiendsg/SmartCom)


Specifying and controlling elasticity of human-based services

What if we need to invoke a human?
SYBL extension

Notification description

```
Notification := notificationID:NOTIFY Role WHEN ComplexCondition
  : notify(NotificationType, message)
Role := ROLE(Responsability1, Responsability2), Role |
    ROLE (Responsability1, Responsability2)|
    RoleX, Role | RoleX
NotificationType := NOTIFICATION | ERROR | WARNING
```

Notification directive example

```
No1: NOTIFY OperationsManager WHEN responseTime > 1.2 s :
  notify(WARNING, "Response time exceeds 1.2 s")
```
Selecting human units

- Do not select at all
  - Let human units bid the tasks
    - E.g., in crowdsourcing platforms

- Static/fix mapping
  - E.g., using static information for human-task mapping

- Simple selection techniques
  - Using the requirement of the task to find the suitable human units based on their capabilities

- Complex selection techniques
  - Utilizing complex dependency graphs to find suitable human units
Selecting SCU based on task graphs

Hong Linh Truong, Schahram Dustdar, Kamal Bhattacharya: Programming Hybrid Services in the Cloud. ICSOC 2012: 96-110

ASE Summer 2016
Placement techniques for human units

- Usually at design time the developer/designer decides
  - Where to put human units
  - Where some triggers should be put in order to invoke human units if needed
- At runtime
  - Find suitable human units
  - Invoke human units
- Placement of human units
  - Application-specific
  - Needs automatic algorithms and supporting tools
Exercises

- Read mentioned papers
- Analyze pros and cons of existing frameworks for data analytics
- Survey existing algorithms for matching human units to data analytics tasks
- Examine requirements for locating places for human units and implement some algorithms
- Examine monitoring techniques for cloud of human compute units
Thanks for your attention

Hong-Linh Truong
Distributed Systems Group, TU Wien
truong@dsg.tuwien.ac.at
http://dsg.tuwien.ac.at/staff/truong