On Providing Crowdsourcing as a Service
(Position Paper)

Thomas Springer, Tenshi Hara, Alexander Schill

Technische Universität Dresden
Computer Networks Group
MapBiquitous Scenario

- Multiple ways to use crowdsourcing
  - update floor plans and navigation paths
  - updating wifi fingerprint db (e.g. by scanning barcodes distributed inside the building)
  - sharing calibration data of barometer for building level detection
  - Application level crowdsourcing, e.g. is the mensa currently empty or crowded

- Integrated indoor/outdoor location-based system
- Decentralized client/server architecture
  - Building servers manage floorplans and data for positioning and routing
Current State

- Long lists of self-contained crowdsourcing projects
- Cloud-based Crowdsourcing Platforms
  - Mechanical Turk (Amazon)
  - Open Turk
- Major drawbacks
  - Submission processing at the 3rd party platform
  - Crowdfunders don’t have full control over submission processing
  - Limited to types of crowdsourcing tasks and processing capabilities supported by the crowdsourcing platform
  - Crowdsourcers and crowdfunders have to trust the platform provider
  - Users have to maintain multiple logins
Challenges

- **Contributors want to:**
  - contribute using different (mobile) devices independent of their location
  - track, correct and withdraw submissions
  - be motivated
  - protect privacy

- **Requester want to:**
  - address a broad base of contributors
  - use specific mechanisms to process submissions
  - provide rewards
  - ban malicious users

- **System should**
  - be easily adoptable
  - be highly scalable and fault tolerant
  - operate in a global setup
Introducing a crowdsourcing proxy

- Server Access Network Entity (SANE)
- Separation of user/submission management and submission processing
Submission Forwarding

1. Deploy CrowdSourcing Plug-in
2. Exchange Keys
3. Get Server Key
4. Exchange Keys
5. Exchange Keys

Contributor

Requester

SANE

CGCloud 2013 - On Providing Crowdsourcing as a Service
Scalability

- multiple SANE instances cooperate to offer Crowdsourcing as a service
- self-organization of proxies to ensure scalability and fault tolerance by using a Distributed Hash Table (DHT)
Assigning clients to SANE instances

- client identifier -> hash -> managed by particular SANE instance
- neighbouring SANE instances act as backup

Client
- myID: PeterMustermann
- ID-hash: bbe29c00e7661de0fd6ab795f49bda549ffcae5dd63d96d8670b712360528c32

SANE 1
- myArea: [0-3]
- lower: [0-3]
- upper: [4-7]
- handle: [c-7]

SANE 2
- myArea: [4-7]
- lower: [0-3]
- upper: [8-b]
- handle: [8-3]

SANE 3
- myArea: [8-b]
- lower: [8-b]
- upper: [4-7]
- handle: [4-f]

SANE 4
- myArea: [c-f]
- lower: [8-3]
- upper: [0-3]
- handle: [8-3]

CH DHT
- 00-11.ch.sane.example.com
- 20-37.ch.sane.example.com

DE DHT
- 40-57.de.sane.example.com
- 58-7f.de.sane.example.com

DNS

Country-API

Client
Submission types:
- Wifi and GSM fingerprints
- Position correction
- Crowdedness of student mensa
- Road surface state for bicycles
SANE Evaluation

- Performance and scalability comparable to website with DB usage
- Linear correlation between requests, replies and packet-loss (99.8% of packets send are being received) -> saturation @ about 5000 requests
- DB at no time a bottleneck
- SANE communication introduces overhead

500 – 1000 simulated Clients on Lenovo Thinkpad X220 Tablet (Intel Core i7-2640M @2.80 GHz/8GB), Windows 7 Professional SP 1, x64

SANE 1 on VM (Intel Xeon E5620 @2.40 GHz/2 GB), Windows Server 2003 SE SP 2, x32, Apache httpd 2.2.14, PHP 5.2.6, MySQL 5.1.

SANE 2/3 VMs at all-inkl.com
connectivity: Client - DSL-6000 (5880 Mbit/s), SANE - Gigabit-Ethernet Full-Duplex (1 GBit/s), all-inkl.com - Fast-Ethernet Full-Duplex (100 Mbit/s)
average submission size 384 Byte
Conclusion and Outlook

- SANE approach
  - reuse of basic crowdsourcing functionality
  - protect contributors privacy (submitterID, submission encryption)
  - arbitrary mechanisms for submission processing applicable
  - scalability and fault tolerance based on cooperating SANE instances
  - standard web technologies used for implementation
    - Performance and scalability follow characteristics of website with DB usage
    - Proxy delay time inevitably

- Future work
  - test SANE with other scenarios
  - rethinking DHT approach
  - add submission and user rating
  - gamification mechanisms and rewards to motivate contributors
Thank you for your attention!

MapBiquituous project information
mapbiquitous.inf.tu-dresden.de