Efficient Opportunistic Sensing using Mobile Collaborative Platform MOSDEN

Prem Prakash Jayaraman, Charith Perera, Dimitrios Georgakopoulos and Arkady Zaslavsky

Presenter: **Dimitrios Georgakopoulos**

20 October 2013

Agenda

- Introduction
- Collaborative Mobile Sensing
- Challenges and Contributions
- Motivating Scenario
- MOSDEN – Mobile Sensor Data ENgine
- Proof-of-concept Crowdsensing Application
- MOSDEN Evaluation
- Conclusion

Introduction

- Mobile phones have become a ubiquitous central computing and communication device in people's lives [1]
- Smartphones have the potential to generate an unprecedented amount of data
- Individual smartphone can be used to infer information about its user
- Fusing data from a multitude of smartphones from a population of users, high level context information can be inferred [3]

Collaborative Mobile Sensing

Mobile sensing defines the capability of a mobile smartphone system to access any-time, anywhere data produced by on-board sensors, sensors carried by entities or embedded in the physical environment.

- Applications performing mobile sensing are called Mobile Sensing Applications
- Mobile phones more specifically smartphones are equipped with a rich set of onboard sensors
- Mobile sensing applications classified into
  - Personal: Focusing on the individual
  - Community: takes advantage of a population of individuals to collaboratively measure large-scale phenomenon (opportunistic crowdsensing)

Challenges – Collaborative Mobile Sensing

The key challenge here is to develop a platform that is autonomous, scalable, interoperable and supports efficient sensor data collection, processing, storage and sharing.

- Autonomous ability to work independently during disconnections
- Collecting all sensor data and transmitting it to a central server is expensive due to bandwidth and power consumption
- Need for local storage, processing and ability to answer to queries
- Scalable and interoperable to support collaborative crowdsensing applications with community of users with heterogeneous device capabilities

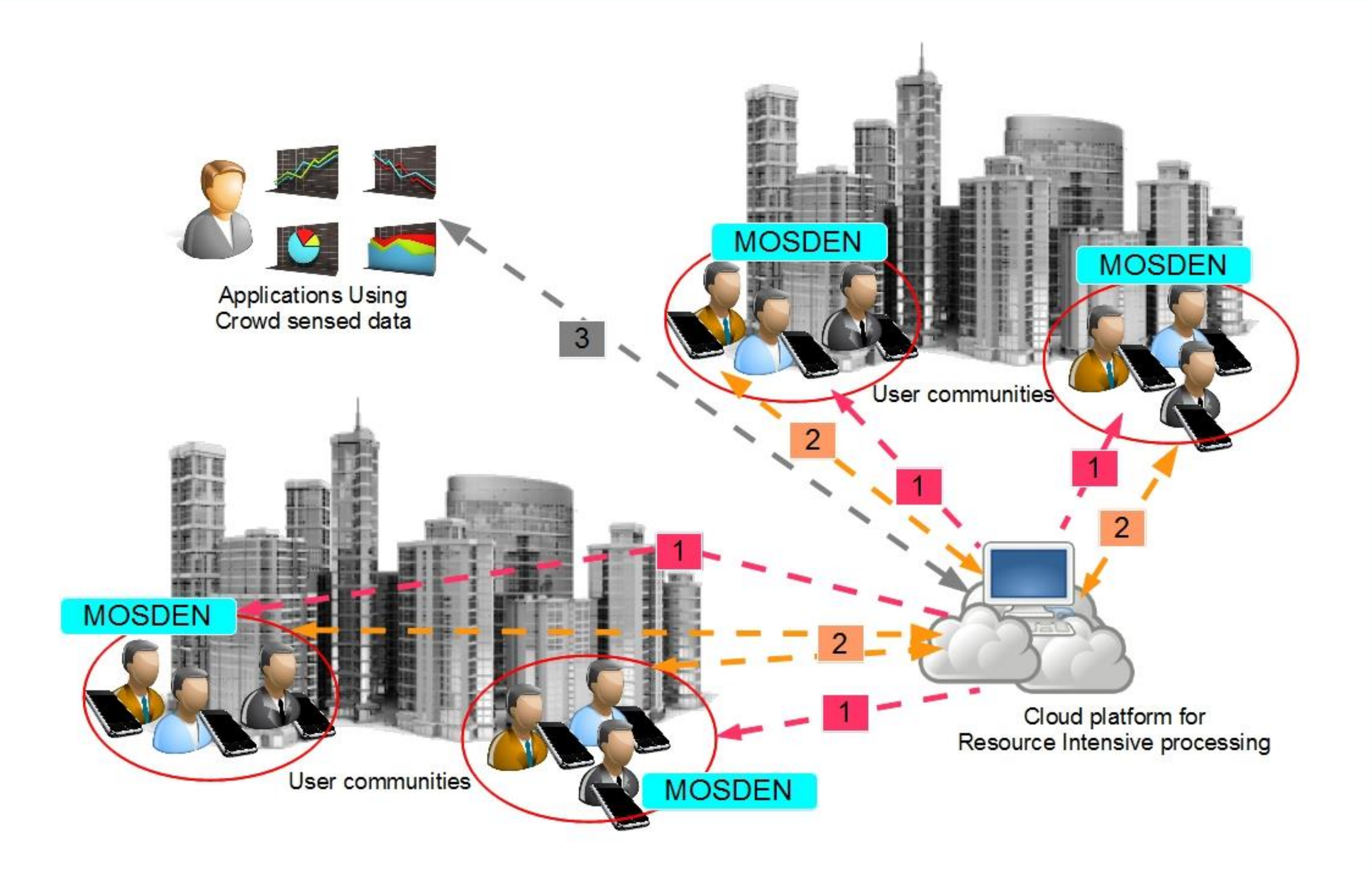
Controbutions

We propose a collaborative mobile sensing framework namely Mobile Sensor Data Engine (MOSDEN)

- We present the design and implementation of MOSDEN, a scalable, easy to use, interoperable platform that facilitates the development of collaborative mobile crowdsensing applications
- We demonstrate a proof-of-concept collaborative mobile crowd-sensing application developed and deployed using MOSDEN platform
- We present experimental evaluation of MOSDEN's ability to respond to user queries under varying workloads to validate the scalability and performance of MOSDEN.

Motivating Scenario

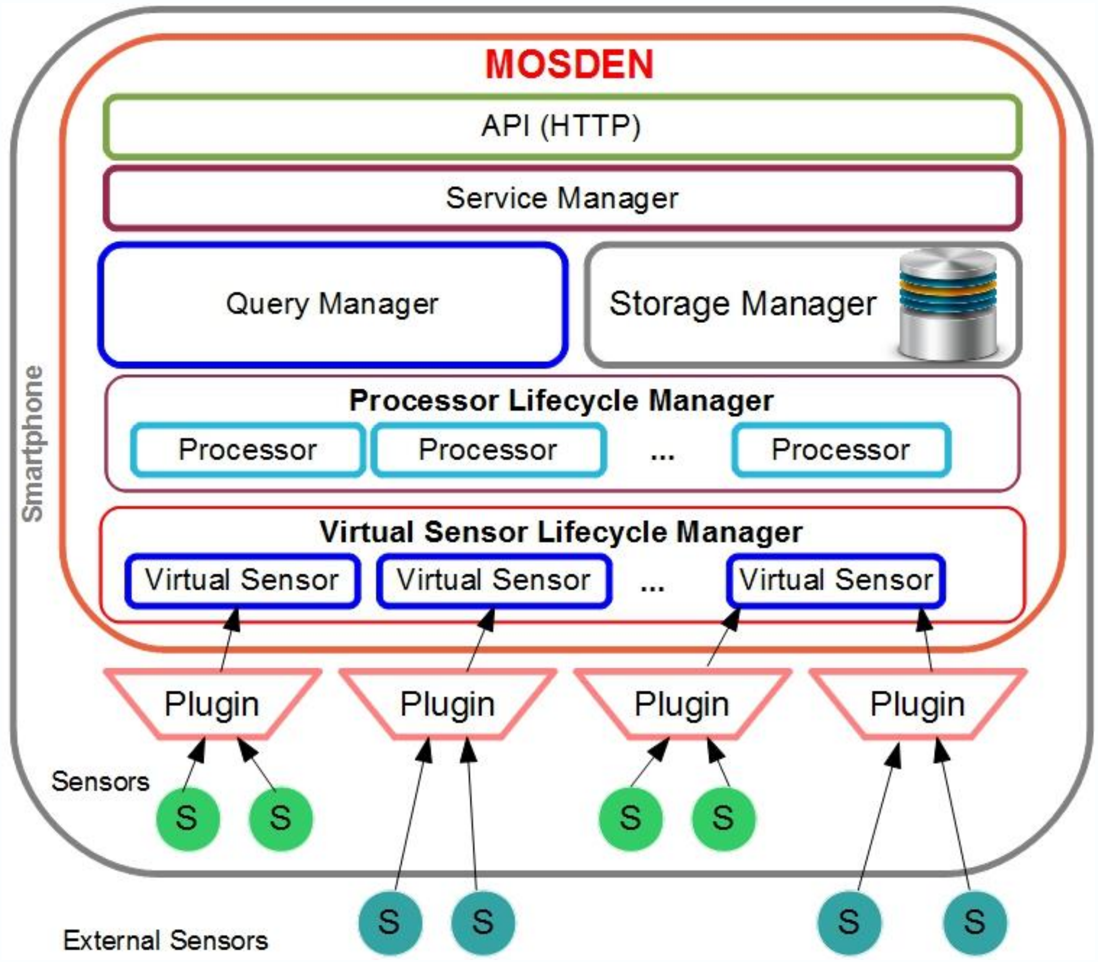
Environmental Monitoring - Mobile Crowdsensing Scenario



MOSDEN - MOBILE SENSOR DATA ENGINE

- MOSDEN, a crowdsensing platform built around the following design principles
  - Separation of data collection, processing and storageto application specific logic
  - A distributed collaborative crowdsensing application deployment with relative ease
  - Support for autonomous functioning i.e. ability to self-manage as a part of the distributed architecture
  - A component-based system that supports access to internal and external sensor and implementation of domain specific models and algorithms

MOSDEN – Platform Architecture



MOSDEN – Platform Architecture

- *Plugins*: The Plugins are independent applications that communicates with MOSDEN. Plugin define how a sensor communicates with MOSDEN
- *Virtual Sensor*: The virtual sensor is an abstraction of the underlying data source from which data is obtained
- *Processors*: The processor classes are used to implement custom models and algorithms
- *Storage and Query Manager:* The raw data acquired from the sensor is processed by the processing classes and stored locally. The query manager is responsible to resolve and answer queries from external source
- *Service Manager:* The service manager is responsible to manage subscriptions to data from external sources

Benefits of MOSDEN Design

- The proposed MOSDEN model is architected to support scalable, efficient data sharing and collaboration between multiple application and users while reducing the burden on application developers and end users
- By separating the data collection, storage and sharing from domain-specific application logic, our platform allows developers to focus on application development rather than understanding the complexities of the underlying mobile platform
- MOSDEN hides the complexities involved in accessing, processing, storing and sharing the sensor data on mobile devices by providing standardised interfaces that makes the platform reusable and easy to develop new application

Benefits of MOSDEN Design

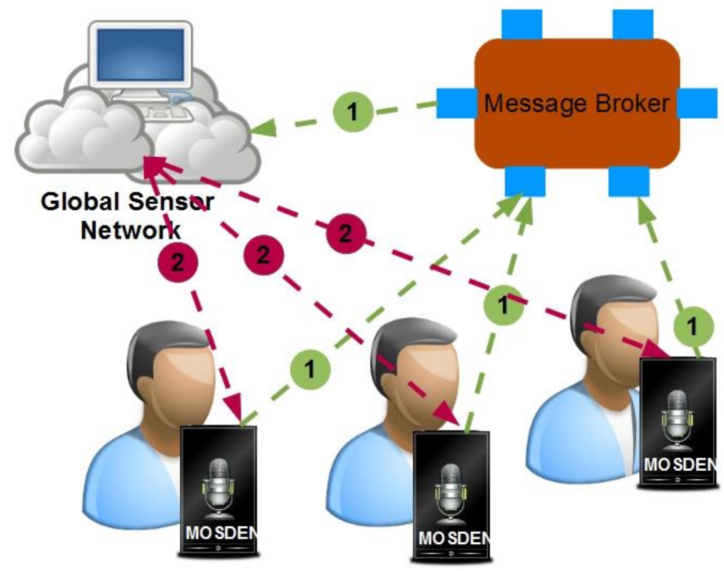
- Minimal user interaction reducing the burden on smartphone users
- Reduce frequent transmission to a centralised server. The potential reduction in data transmission has the following benefits:
  - helps save energy for users' mobile device;
  - reduces network load and avoids long-running data transmissions.

Proof-of-Concept Crowdsensing Application Using MOSDEN

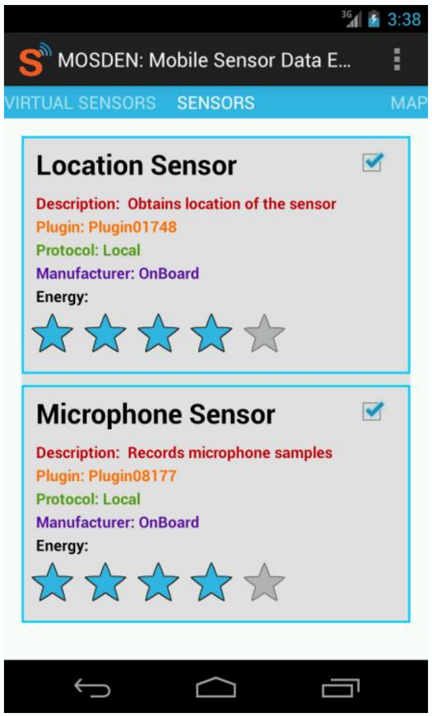
(1) MOSDEN instances running on the smartphone registers with the cloud GSN instance using a Message Broker

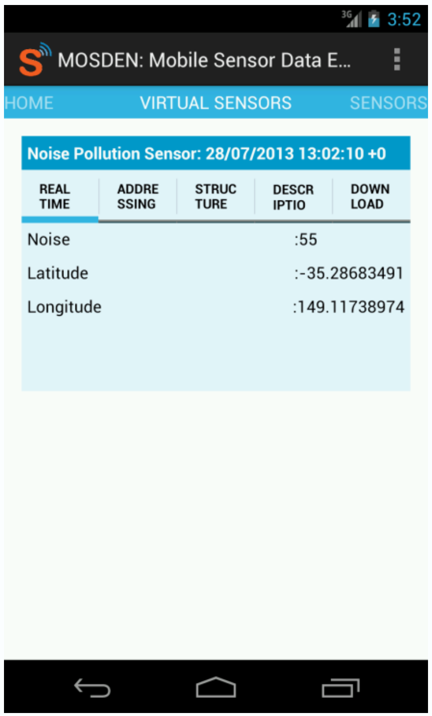
(2) The cloud GSN instance registers its interest to receive noise data from MOSDEN.

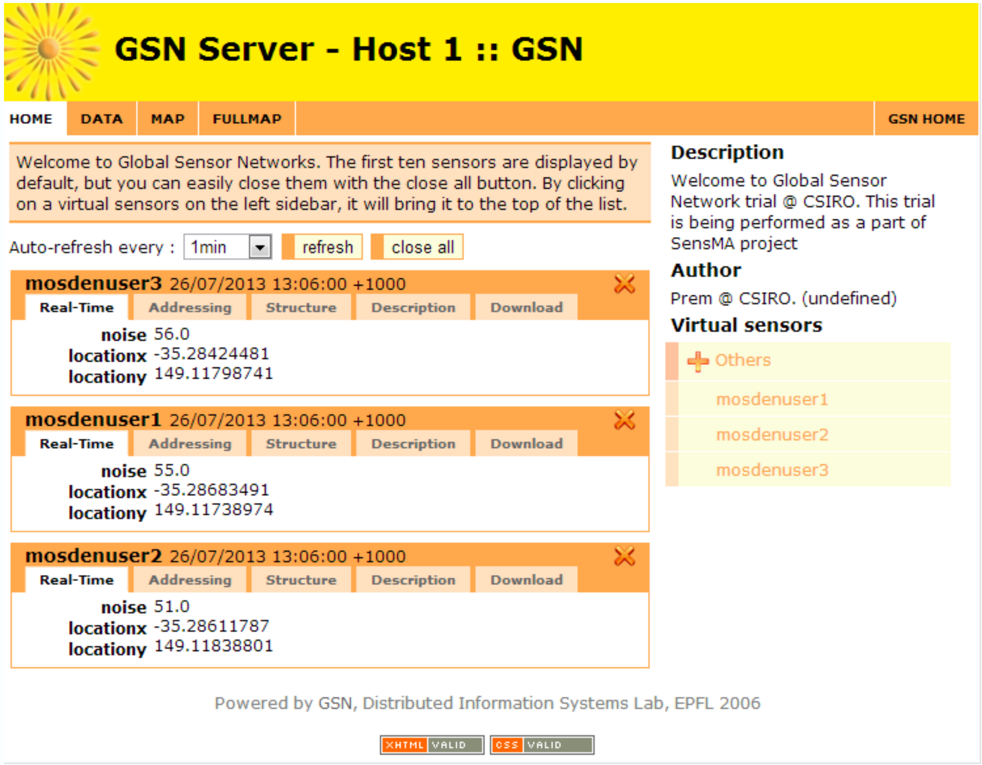
When data is available, MOSDEN streams the data to the cloud GSN. The streaming processes can be push or pull based depending on application requirement.



Proof-of-Concept Crowdsensing Application - Screenshots

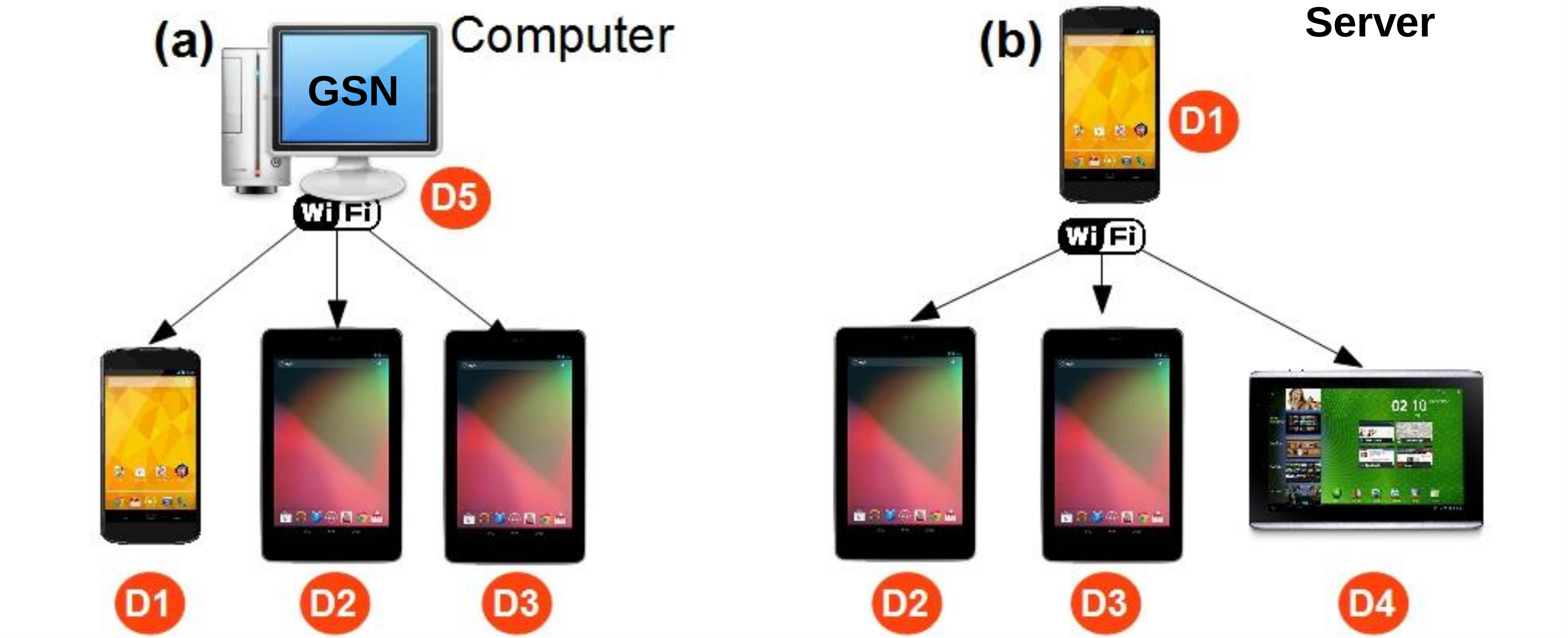






Evaluation of MOSDEN – Experimental Testbed

MOSDEN Server



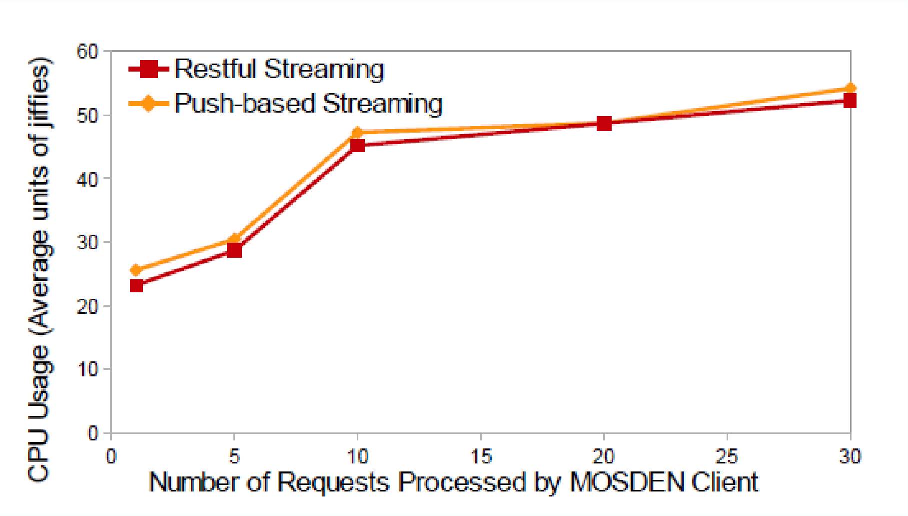
MOSDEN Clients

Evaluation of MOSDEN – Experiments

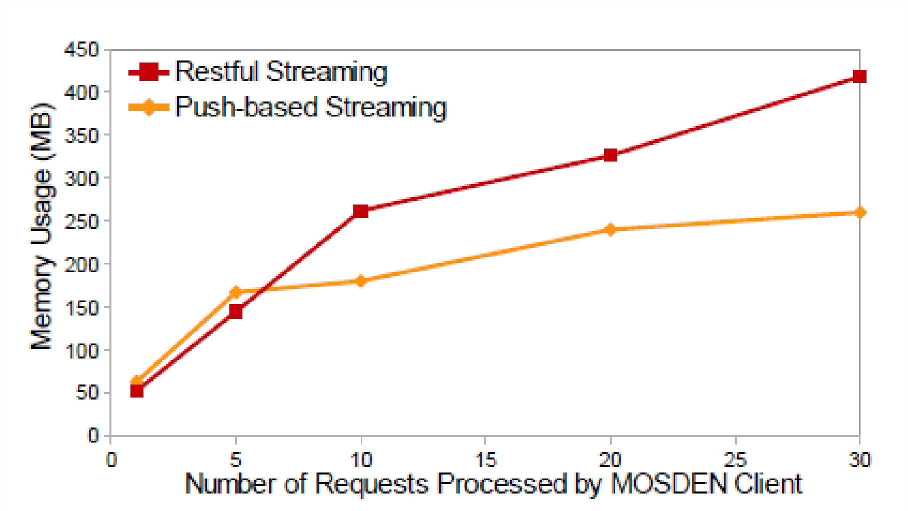
- Performance of restful streaming and push-based streaming methods in terms of CPU usage and memory usage
  - MOSDEN running as Client on mobile device (MOSDEN-Client)
  - MOSDEN running as Server on mobile device (MOSDEN-Server)
- Impact on storage requirements with varying number of sensors
- Query response time

Evaluation of MOSDEN – Results

Comparison of CPU Usage by MOSDEN Client

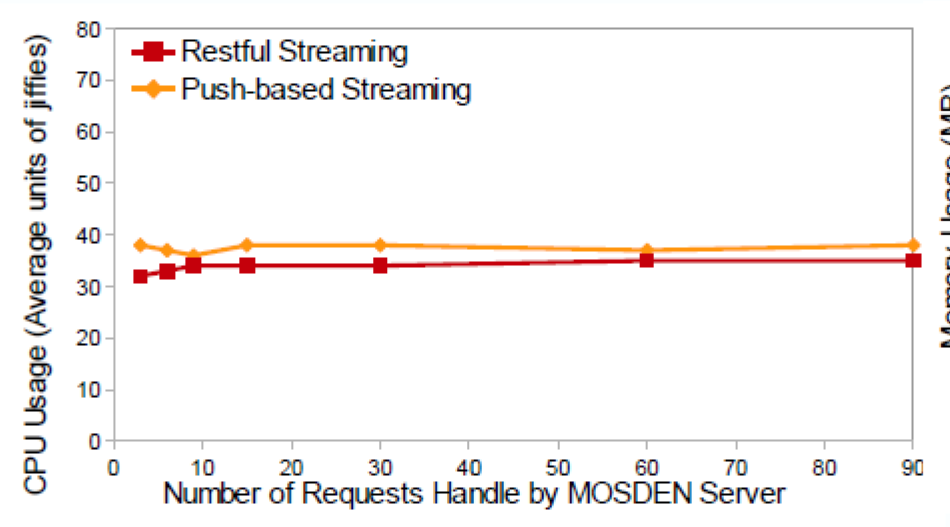


Comparison of Memory Usage by MOSDEN Client

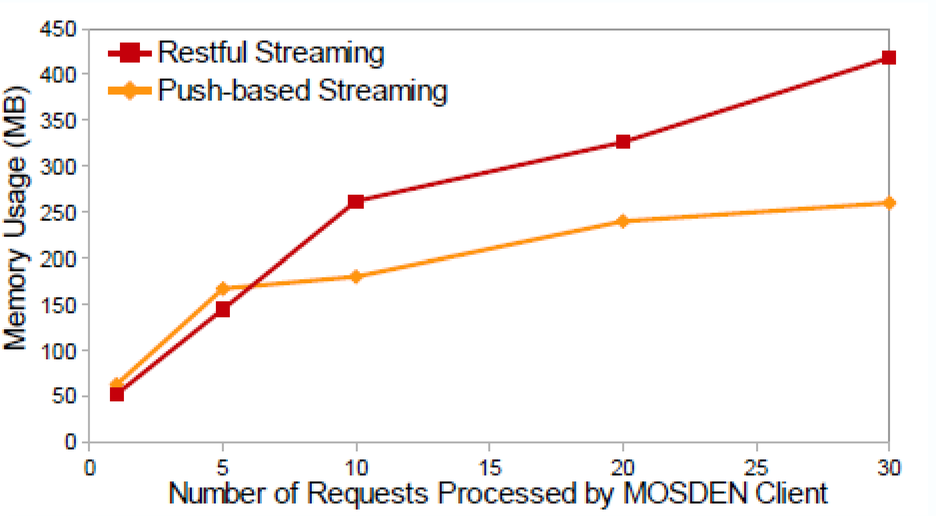


Evaluation of MOSDEN – Results

Comparison of CPU Usage by MOSDEN Server

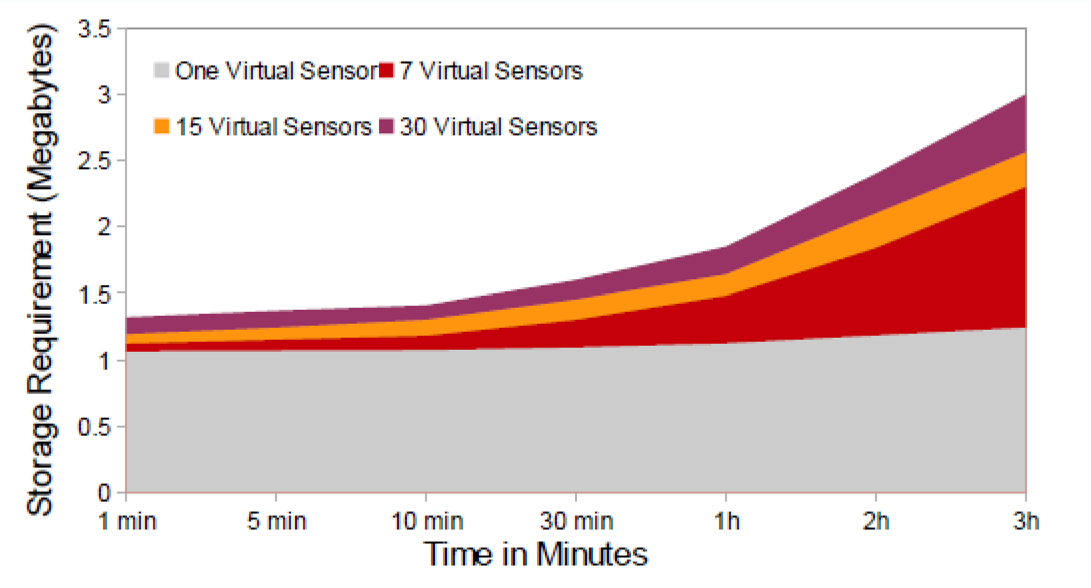


Comparison of Memory Usage by MOSDEN Server



Evaluation of MOSDEN – Results

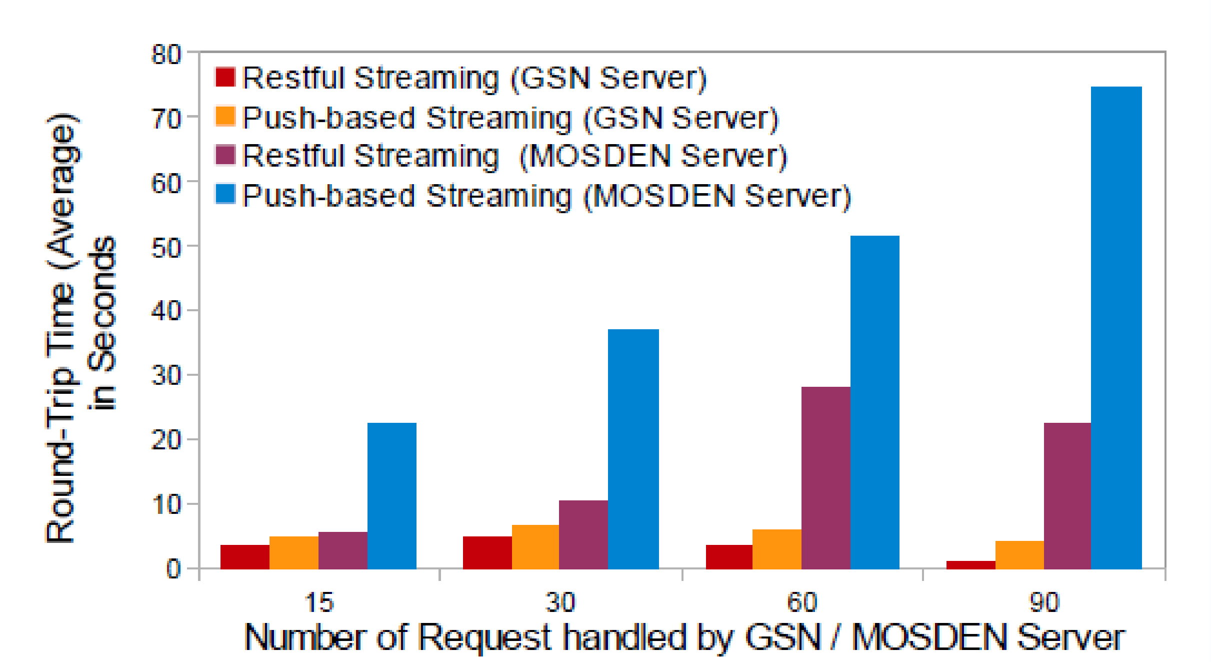
Storage Requirements of MOSDEN Client with Increasing number of Virtual Sensors



Evaluation of MOSDEN – Results

Query Response Time (Sampling rate of 1 second)

Round trip time - the time taken for the server to request a data item from a given virtual sensor on a client. The total time is computed as the interval elapsed between server request and client response



Evaluation of MOSDEN - Discussion

- Overall MOSDEN performed extremely well in both server and client roles in collaborative environments.
  - MOSDEN (as a server) was able to handle 90 requests (i.e. 180 sub requests) where each request has a sampling rate of one second
  - Such intensive processing and extreme load (1 second sample interval resulting in 1800 data points every minute (MOSDEN Client) and 5400 data points (MOSDEN Server with 3 clients) is rare in real-world application
  - If MOSDEN is configured to collect data from 10 different sensors and handle 30 requests (typical of realworld situations), it can perform real-time sensing with delay of 0.4 - 1.5 seconds.
- Experiments validated MOSDEN as a scalable platform for deploying large-scale crowdsensing applications

Related Work

To date most efforts to develop crowdsensing applications have focused on building monolithic mobile applications that are built for specific requirements.

- ❑ Numerous real and successful mobile crowd-sensing applications have emerged in recent times such as WAYZ for real-time traffic/navigation information and Wazer for real-time, location-based citizen journalism, context-aware open-mobile miner (CAROMM) [2]
- ❑ The efforts to build crowdsensing application have focused on building monolithic mobile application frameworks
- ❑ Extending these frameworks to develop new applications is difficult, time consuming and in some cases impossible.

Related Work

- Crowd-sourcing data analytics system (CDAS) [5] is an example of a crowdsensing framework.
- In CDAS, the participants are part of a distributed crowd-sensed system. The CDAS system enables deployment of various crowd-sensing applications that require human involvement
- Mobile edge capture and analysis middleware for social sensing applications (MECA) [6] is another middleware for efficient data collection from mobile devices in a efficient, flexible and scalable manner.
- The MetroSense [7] project at Dartmouth is an example of another crowdsensing system. The project aims in developing classification techniques, privacy approaches and sensing paradigms for mobile phones
- MineFleet [4], another mobile sensing system that use custom built hardware devices on fleet trucks to continuously process data generated by the truck.

Related Work

- Mobile crowdsensing is becoming a vital technique and has the potential to realise many applications
- The aforementioned crowdsensing frameworks and applications are mostly hard wired allowing very little flexibility to develop new applications.
- Frameworks like MECA [6], CDAS [5] use the smartphone as a dumb data generator while all processing is offloaded to the server layer
- Crowdsensing applications like Waze4, MetroSense [7] and MineFleet [4] are built around specific data handling models and application requirements

Conclusion

- We proposed MOSDEN, a scalable collaborative mobile crowdsensing platform to develop and deploy opportunistic sensing applications
- MOSDEN differs from existing crowdsensing platforms by separating the sensing, collection and storage from application specific processing
- A reusable framework for developing novel opportunistic sensing applications
- We validated MOSDEN's performance and scalability when working in distributed collaborative environments by extensive evaluations under extreme loads

Overall MOSDEN performs extremely well under extreme loads in collaborative environments validating its suitability to develop large-scale opportunistic sensing applications.

References

[1] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, “A survey of mobile phone sensing,” Communications Magazine, IEEE, vol. 48, no. 9, pp. 140–150, 2010

[2] W. Sherchan, P. Jayaraman, S. Krishnaswamy, A. Zaslavsky, S. Loke, and A. Sinha, “Using on-the-move mining for mobile crowdsensing,” in Mobile Data Management (MDM), 2012 IEEE 13th InternationalConference on, 2012, pp. 115–124.

[3] A. Zaslavsky, P. P. Jayaraman, and S. Krishnaswamy, Sharelikescrowd: Mobile analytics for participatory sensing and crowd-sourcing applications,” 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), vol. 0, pp. 128–135, 2013

[4] H. Kargupta, K. Sarkar, and M. Gilligan, “Minefleet: an overview of a widely adopted distributed vehicle performance data mining system,” in Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, ser. KDD ’10. New York, NY, USA: ACM, 2010, pp. 37–46.

[5] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, “Cdas: a crowdsourcing data analytics system,” Proc. VLDB Endow., vol. 5, no. 10, pp. 1040–1051, Jun. 2012.

[6] F. Ye, R. Ganti, R. Dimaghani, K. Grueneberg, and S. Calo, “Meca: mobile edge capture and analysis middleware for social sensing applications,” in Proceedings of the 21st international conference companion on World Wide Web, 2012, p. 699702.

[7] “Metrosense.” [Online]. Available: http://metrosense.cs.dartmouth.edu/

Thank you