Mobile Device-to-Device Distributed Computing Using Data Sets*

Diogo Remédios, António Teófilo
Área Departamental de Engenharia Electrónica e de Telecomunicações e de Computadores, ISEL - Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa, Portugal
{dremedios, ateofilo}@deetc.isel.ipl.pt

Hervé Paulino, João Lourenço
NOVA Laboratory for Computer Science and Informatics, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa, Portugal
{herve.paulino, joao.lourenco}@fct.unl.pt

ABSTRACT

The rapidly increasing computing power, available storage and communication capabilities of mobile devices makes it possible to start processing and storing data locally, rather than offloading it to remote servers; allowing scenarios of mobile clouds without infrastructure dependency. We can now aim at connecting neighboring mobile devices, creating a local mobile cloud that provides storage and computing services on local generated data. In this paper, we describe an early overview of a distributed mobile system that allows accessing and processing of data distributed across mobile devices without an external communication infrastructure.

Keywords

Mobile Cloud, Mobile Computing, Distributed Computing, Device-to-Device Computing, Android

1. INTRUDUCTION

Smartphones packed with multiple sensors capable of collecting (and hence generating) data, such as cameras and GPS sensors, are becoming ubiquitous [4]. Simultaneously, these devices are also equipped with computing resources that rival the capabilities of servers from a few years ago. Given the ever-growing amount of both the generated data and the computing capabilities of smartphones, it now makes sense to process in the mobile devices the data that is generated by them, instead of offloading the data to external (cloud hosted) services. This mobile cloud computing is even more relevant in case of inexistent or unreliable Internet connectivity that also occur in disaster scenarios or massive crowd gatherings.

In this paper we introduce a preview of a system to process large quantities of locally generated data in a network of mobile devices, which form a local mobile cloud, without network infrastructure support. The idea is to apply the widely used concepts of processing large data in cloud environments [2] but adapted to the mobile computing constraints. Unlike in cloud computing clusters, local mobile clouds suffer mainly from high instability in connectivity, limited communications, and processing restrictions to avoid excessive battery consumption. Additionally, in local mobile clouds, the nodes must handle their own communication, which requires extra processing and resources (e.g., battery).

An example scenario with the aforementioned characteristics occurs when a group of people is on a remote camping trip with no Internet access. In such setting, the construction of an ad hoc network becomes the only possible way to communicate and share data. Ergo, the existence of such a data sharing and computing system would allow each person to provide access to the data he is willing to share, like photos from camping experiences, and to everyone else (in the system) to access and process this data. To tackle the development of such a system, the key insight of this paper is the design and implementation of Distributed Data Sets (DDS) that aggregate locally generated data from the devices and enable access and subsequent processing of that data in no-Internet environments. The main contributions of this paper are the DDS abstraction, its programming model and the system architecture towards allowing data streaming of locally generated data.

2. SYSTEM DESCRIPTION

Each node in the system can provide its own data and ask for operations to be executed on the data available across all the registered nodes. The system organizes each group of data, such as the photos of the motivational scenario, in a distributed structure called DDS. In turn, each DDS is a logical aggregation of structures called Partitions, distributed across the participant nodes. Unlike the normal distributed data sets, where the data is imported and distributed to the nodes by some service, in our DDSs the Partitions of data in each node are built from the local data in the node itself. Moreover, DDSs are immutable; so all the data transforming operations generate new DDSs.

2.1 Architecture

The global architecture, of this system, comprises three layers: network; storage; and computing. The communication layer has to provide communication services between devices, based on a device-to-device (D2D) ha doc backbone, has it should operate in scenario without an exter-

*This work was partially funded by FCT-MEC in the context of the research project CMUP-ERI/FIA/0048/2013

.

MOBIQUITOUS 2015, July 22-24, Coimbra, Portugal

DOI 10.4108/eai.22-7-2015.2260273

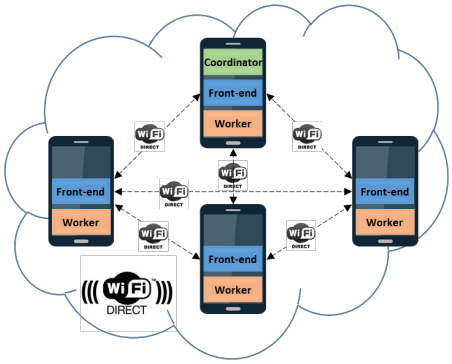

Figure 1: System Architecture

nal network infrastructure. For this layer we use the backbones described in [6] and [1], to offer bi-directional exchange of UDP and TCP messages between devices. The storage layer should offer data persistence and replication, for fault-tolerance, taking in consideration the nodes state and network topology to efficiently deliver data and is an open issue. The computing layer is the focus of this paper, and is composed by three system services: Front-end Service (FES); Worker Service (WS); and Coordinator Service (CS). The FES allows the user to submit processing requests like: register, transform, and filter, or obtain data stored in the system. The WS manages data partitions, e.g. photos, and executes the requested operations on those same partitions. The operations that transform the data generate new partitions belonging to a new DDS. The CS manages DDSs, the nodes running the WS (Worker nodes) and the distribution of the DDS partitions among the Worker nodes. Every node of the network will run the FES and WS and, at least one of them, will have to run the CS, as shown in Figure 1.

A node is uniquely addressable via a unique network-wide identifier, which enables physical mobility and tolerance to eventual changes on IP address.

2.2 Programming Model

The FES offers an interface with an API inspired in cloud programming models like MapReduce [2] or Spark [7], where DDSs can store objects of a given Java type and operate on them with the following operations: open, create, getData, filter, merge, map, reduce and count. All these operations are submitted by the FES to the CS, which forwards them to the Worker nodes that have partitions of the operated DDSs. After the execution of the requested operations on the Worker nodes, they return the results to the CS, which collects them all and returns a final result to the requester FES. For the case of big data objects, the getData operation has been optimized to internally return proxies representing the objects, allowing a direct request from the FES to the Worker nodes minimizing the data traffic passing through the CS.

2.3 Early Prototype

The prototype of the system, currently, has the following limitations: (i) the communication layer uses a local wireless network, but we are currently working on a communication layer that uses Wi-Fi Direct [6, 1]; (ii) the prototype only supports a single Coordinator, but we are currently working to scale up using multiple Coordinators; (iii) the programming model supports all the mentioned operations, but we are working to enrich this set with new operations; and (iv) currently we only support initial imported local data, but we plan to address local data streaming. For now we are working on two prototypes with some shared code: one for the Android platform, and another for the Java SE platform. The former aims to be a proof of concept and the latter to enable to scale the scenarios through simulation.

3. RELATED WORK

Our work focuses on autonomous mobile clouds which means a cloud of mobile devices without any network infrastructure support. In that sense we identify Hyrax [3], a port of Hadoop MapReduce implementation to Android, as the nearest work as it addresses mobile storage and computing, however it's not autonomous in the network infrastructure dependency and doesn't support devices contributing with their own local data. There is also Phoenix [5] which is a mobile autonomous storage system, but doesn't support computing. Nevertheless none of the systems tackle multi-hop networks particularities.

4. FINAL CONSIDERATIONS

This paper presents an early stage of a computational system designed to be deployed on a large number of mobile devices, and to be used in scenarios without an external communication infra-structure (Wi-Fi APs). We are exploring the use of a programming model similar to the ones used in big data processing, adapted to mobile computing and its limitations. The system architecture, that now only supports a single coordinator, is planned to allow for multiple coordinators providing scalability to address the increase of device numbers. We are also working on the communication and storage layers, using Wi-Fi Direct, and planning to explore its characteristics to improve the efficiency of this computation layer.

5. REFERENCES

[1] C. E. Casetti et al. Content-centric routing in wi-fi direct multi-group networks. In *WoWMoM'15, IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks*, Boston, USA, 2015.

[2] J. Dean et al. Mapreduce: Simplified data processing on large clusters. In *6th USENIX Symposium on Operating Systems Design & Implementation*, OSDI'04, San Francisco, USA, 2004.

[3] E. Marinelli. Hyrax: cloud computing on mobile devices using mapreduce. Master's thesis, CMU, USA, 2009.

[4] MobiThinking. Global mobile statistics 2014 part a: Mobile subscribers; handset market share; mobile operators, MobiForge, 2014.

[5] R. Panta et al. Phoenix: Storage using an autonomous mobile infrastructure. *IEEE Transactions on Parallel and Distributed Systems*, 24(9):1863–1873, Sept 2013.

[6] A. Teófilo et al. Group-to-group bidirectional wi-fi direct communication with two relay nodes. In *MobiQuitous'15, 12th Int. Conf. on Mobile and Ubiquitous Systems: Computing, Networking and Services, Coimbra, Portugal*, 2015.

[7] M. Zaharia et al. Spark: Cluster computing with working sets. In *2Nd USENIX Workshop on Hot Topics in Cloud Computing*, HotCloud'10, Boston, USA, 2010.