Life Beyond Distributed Transactions on the Edge

[Draft of ACM paper]

Ali Shoker
HASLab, INESC TEC & Minho University
Braga, Portugal
ali.shoker@inesctec.pt

Paulo Sérgio Almeida
HASLab, INESC TEC & Minho University
Braga, Portugal
psa@di.uminho.pt

Ziad Kassam
HASLab, INESC TEC & Minho University
Braga, Portugal
zakassam@inesctec.pt

Carlos Baquero
HASLab, INESC TEC & Minho University
Braga, Portugal
cbm@di.uminho.pt

ABSTRACT

Edge/Fog Computing is an extension to the Cloud Computing model, primarily proposed to pull some of the load on cloud data center towards the edge of the network, i.e., closer to the clients. Despite being a promising model, the foundations to adopt and fully exploit the edge model are yet to be clear, and thus new ideas are continuously advocated. In his paper on "Life beyond Distributed Transactions: an Apostate's Opinion", Pat Helland proposed his vision to build "almost infinite" scale future applications, demonstrating why Distributed Transactions are not very practical under scale. His approach models the applications data state as independent "entities" with separate serialization scopes, thus allowing efficient local transactions within an entity, but precluding transactions involving different entities. Accessing remote data (which is assumed rare) can be done through separate channels in a more message-oriented manner. In this paper, we recall Helland's vision in the aforementioned paper, explaining how his model fits the Edge Computing Model either regarding scalability, applications, or assumptions, and discussing the potential challenges leveraged.

Keywords

Edge/Fog Computing, Internet of Things, Scalability

1. INTRODUCTION

Cloud Computing [3] is an appealing computing model due to the pay-per-use business model and the elastic resource management that is transparent to the user. The bulk of the storage and computation is mainly done at the cloud data center. The voluminous amounts of data generated by the Internet, especially through the Internet of Things, data analytics, and social networks leveraged new challenges to this model as the data center becomes the bot-

tleneck [9]. The Edge (a.k.a., Fog) Computing [4] model was proposed to mitigate the load on the cloud center by storing and computing the data at the proximity of the user when possible. This brings a lot benefits, among them: (1) A better user experience as the response time will be reduced. (2) Balanced load on the cloud data center and peripherals through using the center only when needed. (3) Better security and privacy as the data is stored closer to the client rather than being exposed in the public cloud. (4) Good use of the network as most of the computation can be done at the edge and often only aggregated data is sent to the central cloud [12].

On the other hand, the challenges at the edge are various, and thus no single storage, communication, or computational model is dominating or adopted as a norm. In search of a data model at the edge, we observed a promising model that was introduced, by Pat Helland in his position paper titled "Life Beyond Distributed Transactions: an Apostate's Opinion" [7], in the area of scalable and distributed applications. In a nutshell, the author presents a model that aims at building "almost infinite" scale applications. The paper builds on the argument that distributed transactions on a (shared) data object cannot work in practice at scale. This suggests having separate data abstractions, called "entities", where each entity is a separate scope of serializability (which allows using transactions efficiently). Consequently, this assumes the presence of applications that often access local (to the entity) data, and rarely require remote access (to other entities). In this latter case, communication is done through separate communication channel abstraction, called "activity", that assumes at-least-once communication medium and implements the desired semantics. This is often implemented by retaining the meta-data that controls the inter-entity communication.

Being proposed one decade ago, Helland's vision in his paper [7] may look an out of the box idea; however, interestingly, we believe that this model fits very well the Edge Computing model. In this paper, we recall the main concepts of Pat Helland's vision, demonstrating how this model fits the edge. We make more emphasis on the properties and features that are related to the edge application patterns as well as pointing out the potential challenges and limitations. We plan to perform a comprehensive theoretical and empirical study on this subject in the future.

In the next section, we summarize the vision of Pat Hel-

land in his paper [7] and then discuss the synergizes or issues that may exist on the edge; we conclude in Section 3.

2. LIFE BEYOND DISTRIBUTED TRANSACTIONS: AN EDGE PERSPECTIVE

In this section, we aim at both recalling Helland's vision in his paper [7], which we refer to as HV to avoid repetition, and demonstrating how it can be applied to the edge computing model. To tackle the subject in a comprehensive way, we organize this section as follow:

- We start by showing the motivations and key aspects that lead to HV.

- We describe the key approach of HV.

- And finally, we discuss the assumptions of HV.

To keep the subject easy to follow, we adopt this strategy on each discussed aspect: we summarize the point as proposed in HV, and then we immediately discuss its application and challenges on the edge. We also try to avoid extensive discussion when we think the addressed point is fairly clear.

2.1 The Motivations

Helland's vision (HV) is driven by two main observations.

Limitations of Distributed Transactions.

First, the increased business demand for greater scalability led to realize that distributed transaction are not very practical at high scale. Some of its symptoms are: bottlenecks due to its increased processing overhead, lack of partition tolerance, and other disadvantages that makes it fragile and disturb availability. Consequently, almost infinite scalability suggests data designs with separate "fine-grained" items having separate scope of transactions. In this way, a transaction is kept efficient as long as it does not span remote items. Dealing with remote data, however, requires a complementary method through messaging. This view is common to edge computing as well. In fact, edge computing is motivated by the fact that many applications may often require local data view of an *edge region* (also called cloudlet and sometimes simply fog) rather than a full view of the cloud. The analogy here is that an edge region can be considered a separate scope of transaction whereas accessing other edge regions or the cloud will require other messaging channels. Note that this aspect faces some challenges and limitations on the edge with highly dynamics nodes and network partitions within the same edge region; whereas, it is easier to deploy on topologies that have fixed membership and good connectivity.

The trend is to scale out.

The second observation is that the trend for higher scalability can be often described by: "individual things do not get significantly larger, we just get more and more of them". In fact, this is one of the main reasons towards adopting the edge model in which some of the load on the cloud data center is pulled to the (plenty of) machines on the edge, especially as the Internet of Things (often composed of maybe millions of "Things") represents the prime application for the edge. Therefore, in the edge model the dominant pattern is to move from scaling up to scaling out.

2.2 The Approach and Abstractions

HV approach divides the data abstractions into independent items, called "entities", and messaging controlling state between them, called "activities". The objective is to hopefully reach linear scalability as items are added.

The "Entity" abstraction.

In HV, an entity lives on a single machine at a time and the application can manipulate just one entity at a time. Instead of global transactional serializability, multiple disjoint scopes of transactional serializability are assumed (discussed later). Only changes in single entities are atomic; atomic transactions cannot span entities, and messaging is used by entities to communicate. In other words, messages are used to manage the workflow between disjointed transactions. From an edge perspective, an entity can live in one edge region or cloudlet. We generalize this since it is unlikely to constrain an edge application (e.g., IoT) to span the local device only. Note that a region can comprise many entities, provided that they are application-independent. Consequently, a transaction can be efficiently applied on an entity being executed close to the user with fairly tightly coupled machine. The atomicity can however be challenging in loosely coupled edge regions which may require dissemination overlays with stronger guarantees.

The "Activity" abstraction.

In HV, an activity is an abstraction that controls the interaction and communication between entities. Each entity will have an activity for each partner entity from which it receives messages. Intuitively, this requires a reliable communication medium. HV assumes only at-least-once messaging provided that the application tolerates duplication or messages are idempotent (discussed later). The knowledge of the received messages creates some meta-data that is wrapped up on a per-partner basis. Activities retain the delivery meta-data to keep track of messages between entities. In the edge model, the activity can be used across entities within a single edge region, since messaging is the norm rather than the exception (in order to tolerate the hostile communication medium and mobility). Another challenge here is reduce the meta-data stored as some edge devices (e.g., gateways, motes, etc.) have a limited storage capacity. On the other hand, an activity can also be used across edge regions or the cloud in the applications which occasionally require remote data.

2.3 The Assumptions

Pat Helland proposed the above approach based on three main assumptions. We present these assumptions discussing how they are merely what is assumed in edge applications, and we elaborate on how to guarantee some of them.

2.3.1 Multi-layer Architecture

HV assumes that an application must be divided into several layers, at least two, which differ in their perception of scaling: a lower layer which is scale-aware and an upper scale-agnostic layer. The former directly deals with adding/removing nodes and understands the underlying scale and guarantees, thus making this scalability transparent to the upper scale-agnostic layer. This architecture will remove the burden of scalability issues to make the applications easy to program. In edge computing, this assumption also holds

as the design will need to separate the dissemination layer from the application layer, otherwise the latter will be very complex to develop and maintain, given the hostile communication nature on the edge. We however don't assume this is the only paradigm to use. If the application is multi-layered we can build one layer to take care of data that is shared across entities. Although this can be constrained in some cases, different models of the recent synchronization-free computing can be used to maintain high availability through eventual consistency [11, 2, 5].

2.3.2 Scopes of Transactional Serializability

The second assumption is to use multiple disjoint scopes of transactional serializability. Each data item resides in a single computer or cluster, and atomic transactions are executed within a single scope of transactional serializability, but not across these disjoint scopes. This can also be assumed for edge applications. When the edge region is almost centralized, atomicity is easy. Even when the entity is distributed in a small cluster, serializability can be achieved by using a famous consensus protocols as Paxos [6] as long as the data nodes are close. Therefore, an edge region can be considered a single scope of serializability, whereas the interaction across entities is performed through the following assumption. However, in edge models where hundreds of nodes are used, it is very hard to achieve atomicity. Alternatives can be using recent tree-based gossip protocols that can achieve high performance when possible, and fall back into a more reliable gossip when the network is unreliable [10].

2.3.3 Message Delivery Semantics

HV assumes at-least-once delivery provided that a message is idempotent. Although most applications may use at-least-once message delivery, which is good with short-lived Unix-style process, sometimes a message may modify some data durably represented on disk, which can lead to the situation where the message will be consumed but not acknowledged directly. Even worse, whenever a failure happens, a message may be dropped or duplicated (upon retransmission). This will require the application to tolerate message duplication, otherwise exactly-once delivery is required. Although the exactly-once issue is common to any system model, it can lead to catastrophic impacts on the edge (i.e., imagine this happening in missile actuators or sensors). Classically, exactly-once can be maintained through keeping track of delivered messages (e.g., using UDP). This, however, imposes a high storage and time overhead when logs get larger. On the edge, this can be worse with the existence of thousands of devices. On the other hand, relying on TCP is not very recommended in hostile environments especially that TCP cannot guarantee exactly-once under failure [8]. A possible alternative can be to experiment the approach presented in [1] which describes how efficient exactly-once delivery can be maintained even under bad network conditions.

3. CONCLUDING REMARKS

In this paper, we recalled Pat Helland's vision in his paper "Life beyond Distributed Transactions: an Apostate's Opinion" in the area of scalable distributed applications; and we showed how it fits the edge computing model and where it skews or becomes challenging. We figured out through this conceptual study that this vision can be very promising as an alternative or complementary edge model. Some of the concepts in the mentioned paper may not be very practical for the edge and thus hybrid solutions are advocated (e.g., the use of Conflict-free Replicated DataTypes to efficiently share mutable data). Finally, we identified several challenges that require more investigation in the future, among them are the problems of storage overhead and exactly-once, which we are currently working on.

4. ACKNOWLEDGMENTS

5. REFERENCES

[1] Ali Shoker, Paulo Sergio Almeida, and Carlos Baquero. Exactly-Once Quantity Transfer. In *In the Proceedings of the SRDS Planetary-Scale Distributed Systems Workshop*, W-PSDS'15. IEEE-CS, September 2015.

[2] P. S. Almeida, A. Shoker, and C. Baquero. Delta state replicated data types. *CoRR*, abs/1603.01529, 2016.

[3] M. Armbrust, I. Stoica, M. Zaharia, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, and A. Rabkin. A view of cloud computing. *Communications of the ACM*, 53(4):50, apr 2010.

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog Computing and Its Role in the Internet of Things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16, 2012.

[5] P. S. A. Carlos Baquero and A. Shoker. Making operation-based crdts operation-based. In *Distributed Applications and Interoperable Systems - 14th IFIP WG 6.1 International Conference, DAIS'14 Proceedings, Berlin, Germany*, pages 126–140, 2014.

[6] T. D. Chandra, R. Griesemer, and J. Redstone. Paxos made live: an engineering perspective. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, pages 398–407. ACM, 2007.

[7] P. Helland. Life beyond distributed transactions: an apostate's opinion. In *CIDR*, pages 132–141, 2007.

[8] P. Helland. Idempotence is not a medical condition. *Communications of the ACM*, 55(5):56–65, 2012.

[9] IBM. Bringing big data to the enterprise, 2016.

[10] J. Leitão, J. Pereira, and L. Rodrigues. Epidemic broadcast trees. pages 301–310, Oct. 2007.

[11] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski. Conflict-free replicated data types. Technical Report RR-7687, INRIA, July 2011.

[12] Steve Wexler. Dell Does IoT. . . Totally! http://it-tna.com/2015/06/01/dell-does-iot-totally/, 2015.