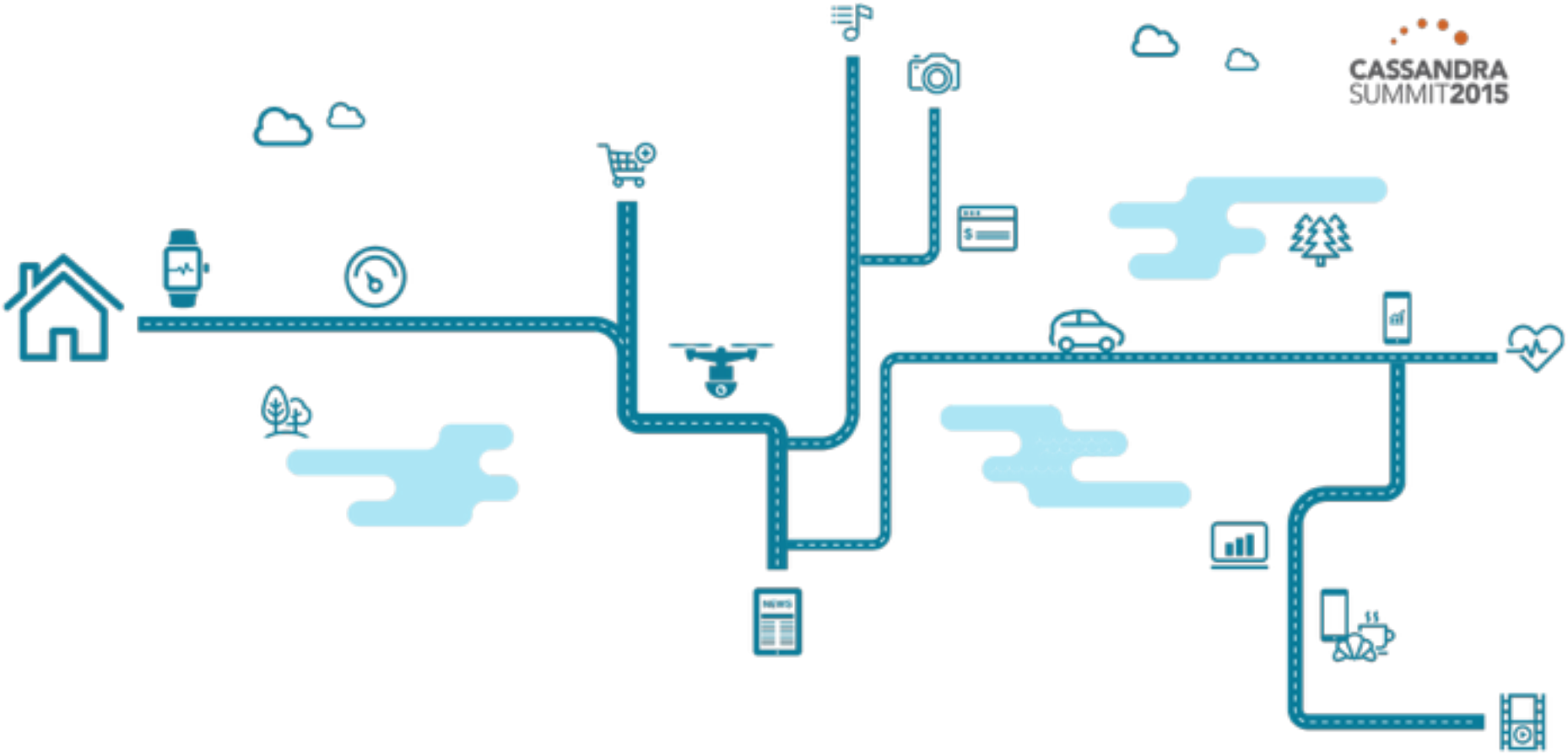

When and how to migrate from a relational database to Cassandra

Introduction

CASSANDRA SUMMIT 2015

- Ben Slater, Chief Product Officer, Instaclustr
- Cassandra as a managed service on AWS, Azure & IBM SoftLayer
- 20 years experience as a developer, architecture and dev team lead

1. Introduction
2. When to consider migration
3. Preparing your application
4. Migration approaches
5. Conclusion

When to consider migration

CASSANDRA SUMMIT 2015

- Reaching physical scalability limits
- Licensing costs becoming prohibitive
- Need 100% availability
- Increasing DBA time to maintain performance / availability
- Active/active multi-DC / disaster recovery requirements

- Weigh against costs:
  - Initial migration
  - Additional logic maintained in app (eg maintaining denormalised duplicate data)

Preparing your application

CASSANDRA SUMMIT 2015

Some approaches while still using relational can help reduce migration costs:

- Abstract data access layer (service oriented architecture)
- Denormalise within relational DB
- Minimise logic implemented in DB
- Build data validation checks & data profiles

Migrations Approaches

CASSANDRA SUMMIT2015

- Big bang cutover
- Parallel run
- Table by table

Big bang cutover

CASSANDRA SUMMIT 2015

- Build & test version of app using C* and convert data from relational to C*
- Shutdown relational, convert data, start-up on Cassandra

- Requires downtime, high risk but likely lowest effort option

Parallel run

CASSANDRA SUMMIT 2015

- Build C* tables
- Modify application to write to both C* and relational
- Develop & execute tool to perform initial sync and reconciliation of dbs
- Run and regularly reconcile
- Migrate reads to C*

- More complex to build and manage
- Lower risk and can be done with no downtime

Table by table / function by function

CASSANDRA SUMMIT 2015

- Either big-bang or parallel run approaches can be done on a table-by-table basis
- Need to be able isolate subject areas with minimal joins in relational DB (likely to correspond to denormalised C* tables)

- Allows staged implementation, gradually moving load from relational to C* - useful if relational environment is under immediate capacity pressure
- Incrementally reduce pressure on relational

Estimating Guide

CASSANDRA SUMMIT 2015

Work Items

- Revise & test operational procedures
- Performance test & soak test
- Trial conversions
- Execute production migration
- Application changes & regression test
- Build migration tool
- Build reconciliation tool
- Build C* schema

Effort Drivers

- # of source tables
- # of access paths
- migration approach
- Level of “preparedness” (slide 5)

Considerations

CASSANDRA SUMMIT 2015

- Don’t forget analytics/ad-hoc querying requirements
- Denormalise – it should feel wrong
- Keep in mind common C* data modelling traps:
  - Partition keys
  - Tombstones
  - Secondary indexes
- Make sure your reads work before migrating / writing
- Upserts make migration easier

Conclusion

CASSANDRA SUMMIT 2015

- It has been done!
- Putting it off won’t make it any easier!

Thank you