CODE FREEZE 2020
TWO-SPEED ARCHITECTURE IMPACTS FOR ML/AI

James Vollmer
Distinguished Engineer
CRHF Software
“There is no AI without IA.”

(The consensus at Strata 2019)
WHY?

ML/AI IS IMPOSSIBLE WITHOUT...
- Knowing where the data is
- Knowing where it came from
- Knowing who the stewards + experts are
- Quality measurements
- Ongoing curation + management

THIS ISN’T “FIRE AND FORGET”
- Iterative, versioned, metrics-driven (“DataOps”)
- Part of a feedback loop to the original entity
- Subsequently used for other AI/data initiatives

DATA DEMOCRATIZATION
- Build trust in our source(s)
- Comprehend where it goes
- Publicize + broadcast results
- Eliminate friction
- Enable access and experimentation
 EXAMPLE PATTERN
(CURRENT)

Observability lacking, as is extensibility

SERVICE A
SERVICE B
SERVICE C
SERVICE D

Transactional DB
Transactional DB
Transactional DB

ODS, DataLake, EDW

FILTER
EXTRACT

Mart 1
Mart 2
Mart ...
Mart N

Cross-domain, 3NF data model creates impedance mismatch with analytics

"Soft" SLA, traditional ETL serves traditional BI Users

Desired state? If so, need to think about integration options.

Downstream marts subset, filter and transform data to meet their needs: provenance and security controls lost

Inherent friction, lack of ontology and disparate formats makes timely access and integration of data difficult if not impossible

January 2020  |  Code Freeze 2020 |   CONFIDENTIAL AND PROPRIETARY
Best-practice reference-data architecture

Data distribution & consumption
- Operational reporting
- Management reporting and business intelligence
- Advanced analytics
- Data-driven apps

Data warehousing
- Operational data store
  - Data warehouse
  - Data lake

Data production & sourcing
- Operational applications
  - Master data management

Data infrastructure
- Data infrastructure

Core Principles

- Limited components with clear objectives and use cases
- Layers and tooling to allow observability of data flow provides a single source to prevent silos and inconsistencies
- Integration of new/legacy capabilities to satisfy emerging requirements based on modern and open technologies
- Build out of a "data mesh" to provide self-serve capabilities to search, acquire, use and monitor

References:
- https://martinfowler.com/articles/data-monolith-to-mesh.html
TRENDS FOR DIGITAL ORGANIZATIONS
ARCHITECTURE @ TWO SPEEDS (LAMBDA)

DEFINITION

- **Normal**: Traditional persistence methods (file, RDBMS, etc.)
- **Fast**: Event-based (messaging, streaming, etc.)

GETTING THERE...

1. **Data**
   - What data provide competitive differentiation?
   - What associated business processes need modification?
   - Data classification/privacy/formats **at ingest** drive downstream efforts (security/consent)

2. **Capabilities**
   - Consumption pattern(s) drive delivery/integration patterns

3. **Technology**
   - Focus on industry standard tooling and outcomes
   - Distinct scalable platforms for distinct use cases (e.g. polyglot persistence, compute/munge tooling)

DATA CLASSIFICATION/DEFINITION
FUNDAMENTAL #1

CURRENT

- At best, data catalog exists (dated), incomplete, lacks security classification(s)
- "What is that thing... again?"
- "Where did it come from?"
- Security abstracted via application
- “Not the job” of data engineers, never baked into effort

DESINTED

- Entities + attributes assigned classification + description
- Protection methods aligned with data classification + associated security policies
- Data classification permeates all data movement + storage
- Classification + Definition deemed 100% necessary for onboarding
- Vetted via DevOps pipeline
DATA STEWARDSHIP  
FUNDAMENTAL #2

CURRENT
- “It’s owned by SAP”
- Everything is confidential
- Everyone wants authority, nobody wants responsibility
- "Where did it come from?"
- Domain bounded context? What’s that?

DESired
- Data owners aligned with domains
- Owners + teams contribute to ontology
- Lineage clearly projected
- Data available in a variety of speeds
  - Analytical
  - Transactional
  - Streaming
- Provisioning: Clearly defined and projected via tooling, regardless of data speed
CURRENT

- Where is it?
- Who did it?
- When did it happen?
- How much does it cost?
- How did it get acquired? And who “owns” it?

DESIRE

- Present all known metadata as soon as it becomes available
- Track all operations on data as it is acquired, transformed and persisted
- Include process metrics (duration, volume, etc.)
- Project and allow consumption based upon use case
- Lineage + provenance NOT an afterthought
- Measure everything (e.g. use, cost)
- Curate KPI’s in line with OKR’s
  - Cost
  - Performance
  - Satisfaction
TWO-SPEED ARCHITECTURE IMPLEMENTATION – ROUND 1

SERVICE A

SERVICE B

SERVICE C

SERVICE D

SERVICE E

Transactional DB

Transactional DB

Transactional DB

Transactional DB

Transactiona

DB

QUEUE

Async listener provides deterministic outcome based on training, forwards result back to relational store

Event-driven (CDC, trigger, etc.) used to detect, build and forward message

Queue used to build data corpus for model training and validation

FILTER

EXTRACT

Data Mart

Model Training

ODS, DataLake, EDW

Model Training

Data Mart used to build data corpus for model training and validation
DATA MESH

Object Store(s)

Service A
Service B
Service C
Service D

Queues

Transaction DB
Transaction DB
Transaction DB
Transaction DB

Model Training

Model Learning/testing is iterative from corpus and observability of results against baseline

Asynchronous event creation/consumption can leverage scalable compute capabilities

Model execution becomes part of the overall transactional flow and is a first-class citizen of the overall solution

Use of alternate persistence methods to provide flexibility/lower cost

All services log begin/end/metrics in a well known format.

Data mesh becomes a method to find, use and create derivative data artifacts
DO WE WANT....

Hunters/gatherers?

- Scrounge and forage for sustenance
- Varying levels of quality
- Unknown or questionable sources

Or modern farmers?

- Data-driven
- Controlled environments
- Sources well known and documented
- Quality measured against internal + industry benchmarks
IMPLEMENTATION
DEPLOYING AN ONTOLOGY AT THE SPEED OF DATA

1. Define + Categorize @ ingest
   - Common schema(s)
   - Well known formats
   - Data Classification

2. Demonstrate
   - Lineage/provenance from origin to consumption
   - Search capabilities
   - Quality measurements

THE BENEFITS
- Promotes consistency from conception
- Minimal effort to implement
- Codified – CI/CD Pipelines
- Develops our “data mesh”
- Project a common definition of our data
- Paramount to building secure solutions
“THE CATALOG”
CORPUS OF EVENTS RELATED TO DATA ACQUISITION

- Receipt/processing/results
- Additive: As we derive more content, we continue to augment
- Searchable
- Analytics
- Enables visibility much sooner in our data pipeline
| THE TECH STACK |

<table>
<thead>
<tr>
<th>OBSERVABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ElasticSearch (operations/catalog)</td>
</tr>
<tr>
<td>Apache Atlas (lineage/provenance)</td>
</tr>
<tr>
<td>AppDynamics (infrastructure)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INGEST + EGRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swagger-defined services (C#/Java)</td>
</tr>
<tr>
<td>Apache NiFi</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>QUEUE MANAGEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Kafka</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Avro</td>
</tr>
<tr>
<td>Schema Registry</td>
</tr>
<tr>
<td>CloudEvents Spec</td>
</tr>
</tbody>
</table>
INGEST CATALOG
SEARCH/PRESENT

Note: Numbers are fictitious
THE DATA REPOSITORY
SEARCH/FIND/USE ...AND REPEAT

1. **Interface** to search for details pertaining to:
   - Entities + attributes
   - Synonyms
   - Classifications

2. **Visualize**
   - Lineage
   - Access
   - Stewardship
   - Provisioning Methods

3. **Derivative data** generated on the platform continue to populate and curate the repo