Pentaho Data Integration
Best Architecture Practices

Matt Casters
Pentaho Chief Architect of Data Integration, Hitachi Vantara
Contents

• Introduction
• General advice
• Specific advice
• Practical examples
• Recap
• Q&A
Introduction:
What is “Data Integration Architecture?”
Introduction

• What is “data integration architecture”?
  – High level view on a (potential) DI solution
  – Describes components and their relationships
  – Taking into account all parts
  – Avoiding details without skipping anything
Introduction

• Why do you need an architecture?
  – Solutions get very complex
  – Teams of engineers get large
  – Conscious decisions on use of solution components
  – Holistic views on security, quality, transparency, performance
  – Allows for validation of high level requirements
  – Allows for the creation and validation of scenarios
  – Clearly defines stakeholders
General Advice:
Some Pointers in Setting up Solid Architectures for Solid Solutions
General Advice – Don’t Forget the Details...

- Learn the basics of the building blocks...
  - PDI Best Practices #PWorld14
    - Standards, naming, ...
  - PDI Best Governance Practices #PWorld15
    - PM, CI, VCS, Testing, ...
  - Get expertise for all software components you use
General Advice – Whiteboarding

• Whiteboarding
  – Is done with interested stakeholders
  – Tries to compromise knowledge from various parties
  – Allows for quick high level design
  – It is just a starting point!
  – Needs to get followed up, validated against scenarios
  – Forget conviction: time to change your mind
General Advice – Scalability

• Parallelize on a high level
  – Aggressive low level parallelization can get you into trouble

• Remember to allow data to flow in swim lanes
  – Parallelization of as much as possible
  – “Sharding” and so on should be architected in

• Identify time window early on, assess HW needs
General Advice – Transparency

• Great complexity requires transparency
  – Something will always go wrong
  – At the worst possible time

• As a rule:
  – always trace data moving between parts of architecture
  – When in doubt: add more logging, tracking and tracing

• Use components in architecture that allow for monitoring
  – Prefer servers that allow you to see what’s going on
General Advice – Predictability

• Enormous workloads, batch jobs, put systems under stress
• Batches tend to grow bigger over time, causing more stress
• As a rule:
  – If you can in any way, use micro-batching
  – Chop up 1 large nightly workload into hundreds of small ones throughout the day
• Advantages:
  – More frequent updates
  – Predictable workload
  – Fail early scenario: problems are detected earlier
Specific Advice:
Advice for IoT and Others
Specific Advice – Hadoop

• Hadoop has itself become an ecosystem of software
• Select the software in the ecosystem to fit your ideal architecture
• Only select properly supported components, avoid bleeding edge
• Combat lack of transparency with extensive logging
• Follow the right sizing for your architecture, balance correctly
• Use it as a scalable part, not just as a “Database”
Specific Advice – IoT

- IoT is Messy
  - Data Quality varying
  - Data Connectivity problems
  - Late arriving data
  - Flash-floods of data (low predictability)
  - High complexity
  - Varying data formats and versions
  - Number of different devices can be high
Hitachi Vantara IoT Offerings

**Dashboard**
- EDGE
  - Asset Integration
  - Data Transformation
  - Data Filtering
  - Edge Analytics

**Alerts / Notifications**
- CORE
  - Data Collection
  - Asset Avatar

**Application Enablement**
- ANALYTICS
  - Artificial Intelligence
  - Batch / Stream / Analytics
  - Data Blending / Orchestration

**Operational Insights**
- Asset Intelligence
- Maintenance Optimization
- Manufacturing Optimization

**FOUNDRY**

**CONNECTED THINGS**
Specific Advice – IoT

• Plan ahead for failure
• Use modern techniques like Metadata Injection
• Make extensive use of queues in any format
• Assume that things will go wrong in every scenario
• Design the architecture to cope with failures
• Design the architecture to report on statistics
Practical Examples:
War Stories from the Field
Examples – Large Services Vendor

- Moving large amounts of small data packets around
- Picked the right tools, didn’t pick an overall architecture
- Different teams “working together” in different countries
- Architecture became secondary to the overall solution
- Technology was selected not architecture
Examples – Large Services Vendor

• Carte servers got hammered thousands of times per second
  – Use of a specific scheduler was mandated
  – Running out of sockets, HTTP server buckling under the load

• Complaints about PDI startup times

• Overall performance too low

• Services called in to solve “critical” issues in our software
Examples – Large Services Vendor

• Don’t allow internal organizational needs drive the architecture
• Don’t allow technology choices to drive architecture
  – And if you too, handle the implications
• To scale, ramp up performance, always queue and intelligently handle queued tasks (not one at a time for example)
• The performance of the whole is determined by the slowest link
  – Consider this up-front in the architecture
Examples – Handling TV Set-top Data

- Periodic in nature, handling clicks
- Reading from MQTT, dumping data into Oracle for analysis
- Reported PDI performance trouble, services called in
- Small scale test, predicted ten-fold increase in size, already in trouble
Examples – Handling TV Set-top Data

• MQTT: great for queuing and IoT
• Not always possible to read in parallel from queues!
• Oracle is an RDBMS, kills parallelism in architecture
Examples – Handling TV Set-top Data

• Consider partitioning large amounts of clients
• Consider data extraction for any data storage mechanism
Examples – Big Bank

• Processed a gazillion records every night
• Had a batch window of 2 hours
• Got a monster computer to do the job with 64 cores
• Ran complex data quality validations in PDI, hundreds of steps
• Got into a performance problem
• Needed extensive performance tuning
Examples – Big Bank

- Good
- Pick 2
- Cheap
- Fast
Examples – Big Bank

Lots of work

Pick 2

On 1 server

In batch window
Examples – Big Bank

• Consider up-front whether HW choices will pin you down later

• Weigh the importance of specific requirements into the architecture
  – time vs complexity vs hardware in this case
Recap:
PDI Best Architecture Practices
Recap

• Make an architecture up-front, not as part of the documentation
• Be critical
• Be detailed
• Run scenarios against it
• Be ready to change your mind
• Get stakeholders involved
• Use PDI : Pessimistic Data Integration
Questions and Discussion