Untangling Spaghetti - When and How to Split Projects

Owen O'Malley
owen@hortonworks.com
@owen_omalley

June 2018

HORTONWORKS
POWERING THE FUTURE OF DATA

Who Am I?

- Worked on Hadoop since Jan 2006
- Apache Member
- Mentor for a lot of Apache projects
- Hadoop, Hive, and ORC

Goal

- You’ve developed a new Program Bar
  - It works with an open source project Foo.
  - You want to open source it.
- How do you release it?
  - Ask to join project Foo
  - Start as a new Project
- What if you change your mind?

Considerations

HORTONWORKS®

Project Lifecycle

- Open source projects form ecosystems.
  - Environment always change.
- They compete for community.
- Start young and nimble.
- As they age, they slow down.
  - Compatibility over new features
- Eventually releases stop.

Joining an established project

- Advantages
  - –Get instant name recognition.
  - –Can get a large installed base.
  - –Easier integration.
- Disadvantages
  - –Wait to become committers.
  - –Tie your project to their release cycle.

Hadoop RecordIO

- Early serialization library
  - Generated classes using Hadoop's Writable API
  - Put into Hadoop
- Few people realize it is there
- Included in every single Hadoop install
- Can't delete it because it has users

Hive LLAP

- Live, Long & Process
  - Long living daemons that run Hive queries
  - Avoids JDK startup costs
  - Cache hot data in memory
- Integrated development community
- Tightly integrated code base

Together, but Separate

- Apache projects can have multiple releasable sub-projects.
  - Apache Commons is the canonical example
- Allows separate release trains.
  - And bug tracking, source version control, etc.
- Do the two communities overlap significantly?

Hadoop Ozone

- New distributed key/value store
- Introduces a new block storage layer
- Overlapping communities
- Integrates well with HDFS
- Needs faster releases than Hadoop.
  - And to work with older Hadoop versions.

Starting a New Project

- Advantages
  - Release quickly and often!
  - Excitement over a new project
  - Have to address integration immediately.
    - Possible to have version flexibility.
- Disadvantages
  - Will your development community be large enough in the long term?

HORTONWORKS

Avro

- A serialization library started as a project
  - Much better known that RecordIO
- Allowed frequent release cycles
- Used by many projects outside of Hadoop
- Created a complicated dependency tree with Hadoop

Tez

- A execution engine for Hive
  - We needed a replacement for the old MapReduce
  - Needed to execute DAGS instead
  - Lots of performance optimizations
- Started a separate project
  - Almost all use is with Hive
  - Mostly tied to Hive release cycle

Splitting Up is Hard To Do

Where to put ORC?

- While developing ORC, we considered where to put it.
- During the decision, a Hive committer shot down adding bindings for Trevni.
  - Wanted to get buy in, so renamed to ORC
  - Decided to become part of Hive
- Helped the Hive-ORC integration a lot

ORC and Hive

- ORC being in Hive was hindering adoption
  - Other projects didn’t want all of Hive as dependency
  - Viewed as only useful for Hive
  - Other projects’ needs weren’t taken as seriously
  - New C++ code was coming in with new commiters
  - Needed to be more agile
- Now we are repeating with the Metastore

Tooling is Important

- ORC depending on 16,000 classes
  - Outside of Hadoop & Protobuf
- Built custom tool to analyze dependencies
  - Root set of “ORC” classes
  - Ignore “system” classes – Java, Hadoop, Protobuf
  - Sorted by depth from root classes and transitive dependency weight

Splitting up the Code

- Make a module of the code
  - Decide whether it is at the top or bottom of the dependency tree.
  - Make heavy use of interfaces & plugins.
  - Minimize the amount of code duplication.
- If you are in the middle…
  - Make a minimal chunk and release it separately

Splitting up the Code

- After the new module is self contained
  - Including not using the other project as parent POM
  - You can copy the code to new code repository
  - Rename the packages of the code
  - The new project makes a release of the current code
- Now the old project needs to switch
  - New project becomes a dependency of the old

Change is hard

- Part of the untangling meant new APIs
  - –ORC used Hive's ObjectInspectors
    - ObjectInspectors had a huge transitive dependency set
  - –We also had fast vectorized methods
  - –Removed the ObjectInspect methods
  - –Created a compatibility layer in Hive

HORTONWORKS

Avoiding the Cycles

- Got ORC’s dependency on Hive down to ~40 classes
  - Critical for Hive, so couldn’t move to ORC
  - Created a new Hive sub-project called “storage-api”
  - It releases independently of the rest of Hive
    - Current storage-api version is 2.6.1 while Hive is 3.0.0
  - Has its own release branches

Splitting Headache

- We tried to split up Hadoop 10 years ago
  - Common, HDFS, & MapReduce
  - No community for Common!
- Using Ant & Ivy made this hard
- Failure to adequately plan made it worse
- YARN ended up blocking HDFS releases.

HORTONWORKS

Splitting up the community

- You won't make an exclusive community
  - Only people who have worked on this code…
  - Won't work since Apache projects are democracies
  - Half of the ORC committers have no patches. ☺
- You will start forming a new community
  - Build aggressively
  - Be friendly and welcoming

Joys of Small Projects

- Fewer politics!
- Faster builds (minutes vs hours)
- Faster release cycle
- Easier for newcomers to pickup code
- More time on outreach & documentation
  – Good investment anyway

Challenges of Small Projects

- Backwards compatibility is critical
- The projects will be each other's tests
  - Storage-api releases are tested with ORC
  - ORC releases are tested with Hive
- Cross-project changes are extra work
  - To make a change to storage-api, we need to make 3 releases

Conclusions

Conclusions

- First priority should be the community
- Consider how tight the integration will be
  - Tight integration has good points and bad
- Think about the tools available
- Don’t worry, you can always change your mind!

Thank you!

Twitter: @owen_omalley
Email: owen@hortonworks.com

HORTONWORKS®
POWERING THE FUTURE OF DATA™