Hortonworks

HCatalog

Table Management For Hadoop

Alan F. Gates
@alanfgates

Who Am I?

HCatalog committer and mentor

Co-founder of Hortonworks

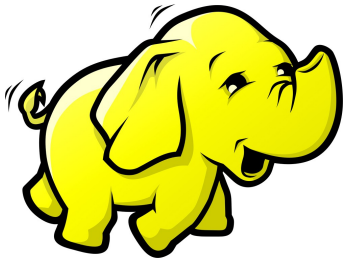
Lead for Pig, Hive, and HCatalog at Hortonworks

Pig committer and PMC Member

Member of Apache Software Foundation and Incubator PMC

Author of *Programming Pig* from O'Reilly

Many Data Tools



MapReduce

- Early adopters
- Non-relational algorithms
- Performance sensitive applications

Pig

- ETL
- Data modeling
- Iterative algorithms

Hive

- Analysis
- Connectors to BI tools

Strength: Pick the right tool for your application

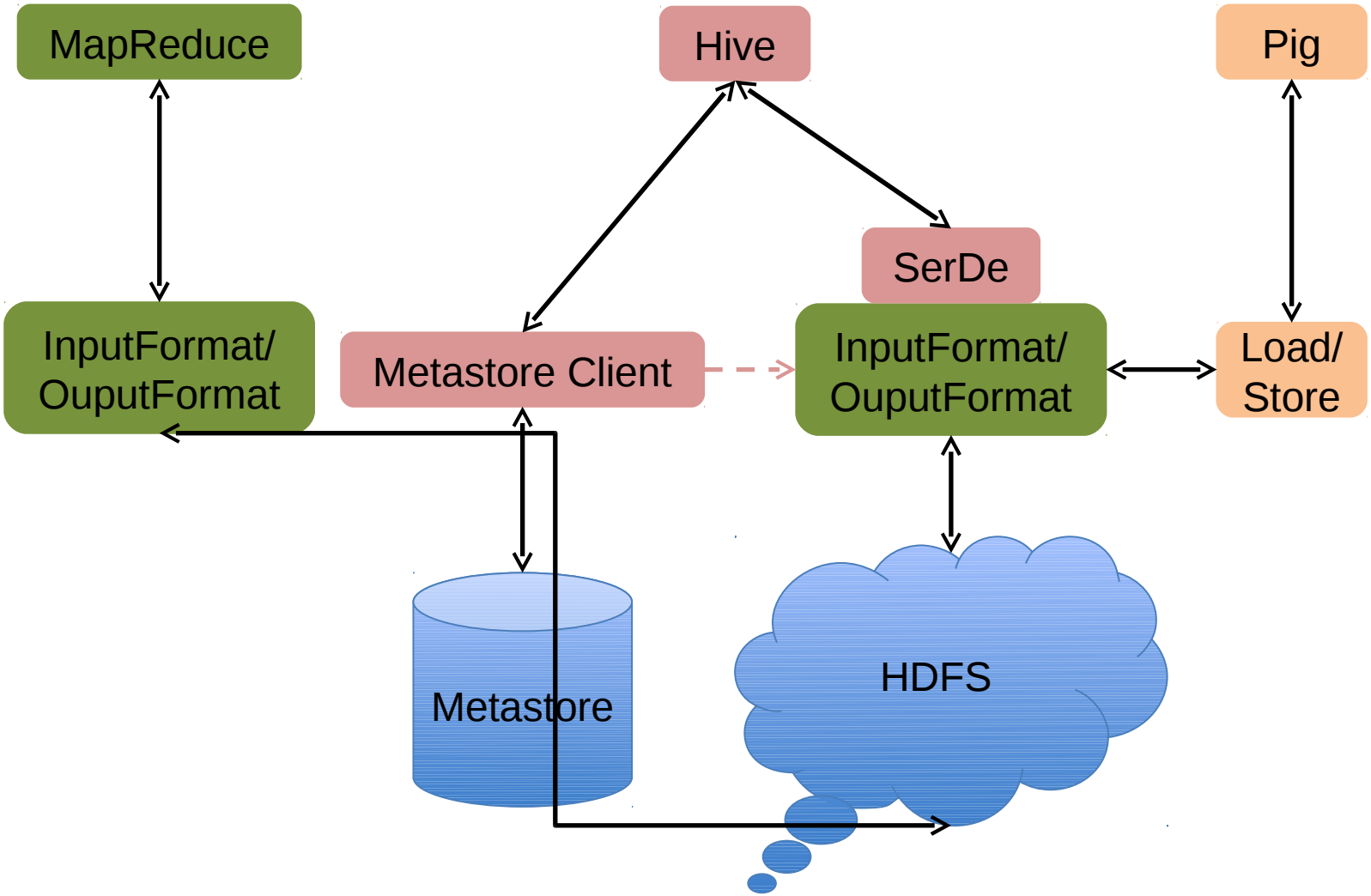
Weakness: Hard for users to share their data

Tool Comparison

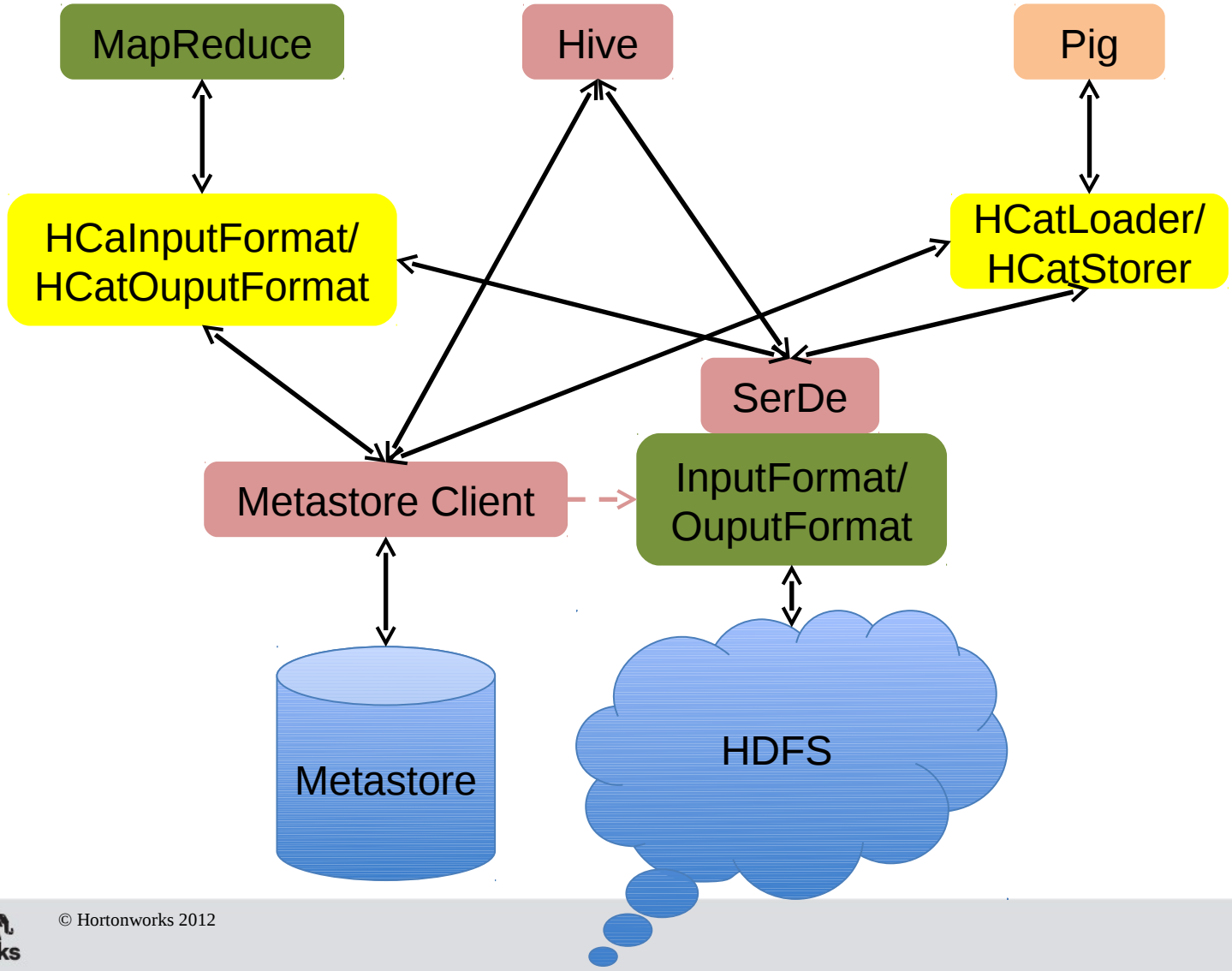
Feature	MapReduce	Pig	Hive
Record format	Key value pairs	Tuple	Record
Data model	User defined	int, float, string, bytes, maps, tuples, bags	int, float, string, maps, structs, lists
Schema	Encoded in app	Declared in script or read by loader	Read from metadata
Data location	Encoded in app	Declared in script	Read from metadata
Data format	Encoded in app	Declared in script	Read from metadata

- Pig and MR users need to know a lot to write their apps
- When data schema, location, or format change Pig and MR apps must be rewritten, retested, and redeployed
- Hive users have to load data from Pig/MR users to have access to it

Hadoop Ecosystem



Opening up Metadata to MR & Pig



Tools With HCatalog

Feature	MapReduce + HCatalog	Pig + HCatalog	Hive
Record format	Record	Tuple	Record
Data model	int, float, string, maps, structs, lists	int, float, string, bytes, maps, tuples, bags	int, float, string, maps, structs, lists
Schema	Read from metadata	Read from metadata	Read from metadata
Data location	Read from metadata	Read from metadata	Read from metadata
Data format	Read from metadata	Read from metadata	Read from metadata

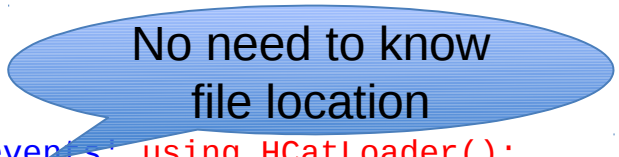
- Pig/MR users can read schema from metadata
- Pig/MR users are insulated from schema, location, and format changes
- All users have access to other users' data as soon as it is committed

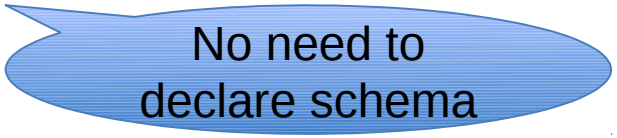
Pig Example

Assume you want to count how many time each of your users went to each of your URLs

```
raw     = load '/data/rawevents/20120530' as (url, user);
botless = filter raw by myudfs.NotABot(user);
grpd    = group botless by (url, user);
cntd    = foreach grpd generate flatten(url, user), COUNT(botless);
store cntd into '/data/counted/20120530';
```

Using HCatalog:






```
raw     = load 'rawevents' using HCatLoader();
botless = filter raw by myudfs.NotABot(user) and ds == '20120530';
grpd    = group botless by (url, user);
cntd    = foreach grpd generate flatten(url, user), COUNT(botless);
store cntd into 'counted' using HCatStorer();
```

Working with HCatalog in MapReduce

Setting input:

```
HCatInputFormat.setInput(job,
  InputJobInfo.create(dbname, tableName, filter));
```

table to read from

database to read from

specify which partitions to read

Setting output:

```
HCatOutputFormat.setOutput(job,
  OutputJobInfo.create(dbname, tableName, partitionSpec));
```

specify which partition to write

Obtaining schema:

```
schema = HCatInputFormat.getOutputSchema();
```

access fields by name

Key is unused, Value is HCatRecord:

```
String url = value.get("url", schema);
output.set("cnt", schema, cnt);
```

Managing Metadata

If you are a Hive user, you can use your Hive metastore with no modifications

If not, you can use the HCatalog command line tool to issue Hive DDL (Data Definition Language) commands:

```
> /usr/bin/hcat -e ”create table rawevents (url string,
user string) partitioned by (ds string);";
```

Starting in Pig 0.11, you will be able to issue DDL commands from Pig

Templeton - REST API

- REST endpoints: databases, tables, partitions, columns, table properties
- PUT to create/update, GET to list or describe, DELETE to drop

Create new table “rawevents”

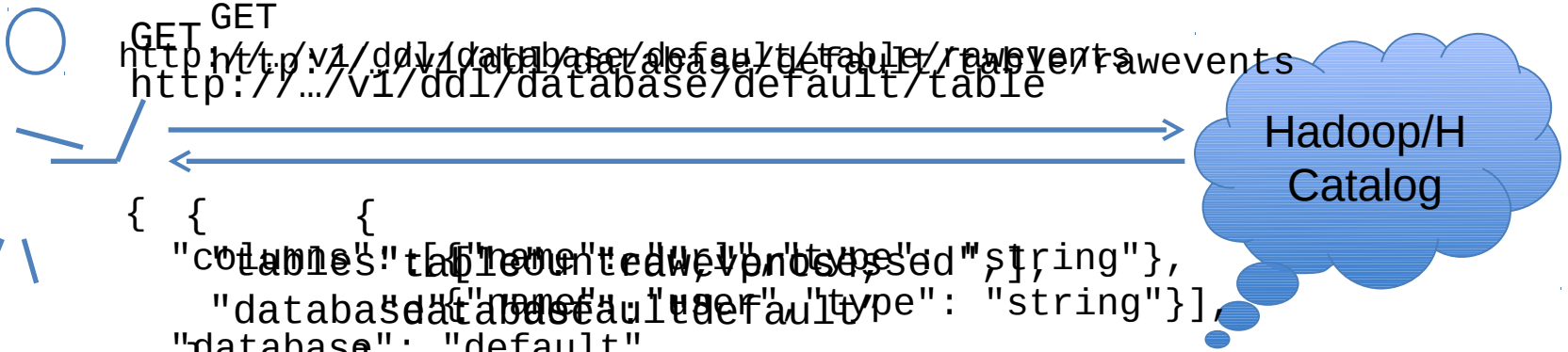
```
PUT
{"columns": [{ "name": "url", "type": "string" },
             { "name": "user", "type": "string"}],
 "partitionedBy": [{ "name": "ds", "type": "string" }]}
```

Describe table “rawevents”

```
GET
http://…/v1/ddl/database/default/table/rawevents
```

Get a list of all tables in the default database:

```
GET
http://…/v1/ddl/database/default/table
```




```
{  {      {
  "columns":[{"name":"url","type":"string"},
   "database":{"name":"user","type":"string"}],
  "database": "default",
  }  }
  "table": "rawevents"
}
```

- Included in HDP
- Not yet checked in, but you can find the code on Apache’s JIRA HCATALOG-182

HCatalog Project

HCatalog is an Apache Incubator project

Version 0.4.0-incubating released May 2012

- Hive/Pig/MapReduce integration
- Support for any data format with a SerDe (Text, Sequence, RCFile, JSON SerDes included)
- Notification via JMS
- Initial HBase integration

Questions