I524 Lecture Notes

Release Draft

Gregor von Laszewski

Jan 26, 2017

CONTENTS

CHAPTER

ONE

I524 BIG DATA AND OPEN SOURCE SOFTWARE PROJECTS

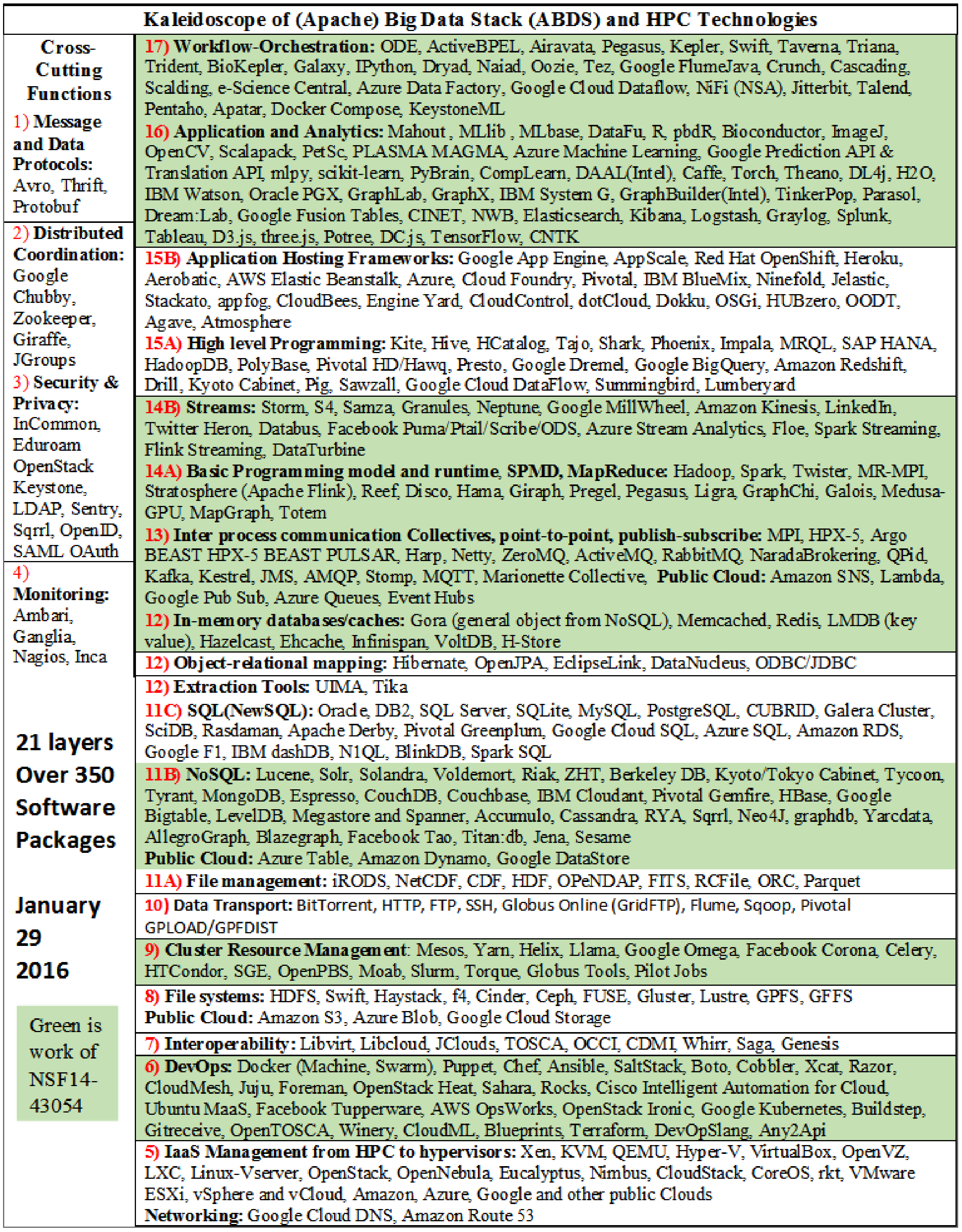
1.1 Overview

This course studies software used in many commercial activities related to Big Data. The backdrop for course contains more than 370 software subsystems illustrated in Figure 1. We will describe the software architecture represented by this collection and work towards identifying best practices to deploy, access and interface with them. Topics of this class will include:

1. The cloud computing architecture underlying open source big data software and frameworks and contrast of them to high performance computing
2. The software architecture with its different layers covering broad functionality and rationale for each layer.
3. We will go through selected big data stack software from the list of more than 370
4. We will be identifying how we can create and replicate software environments based on software deployed and used on clouds while using Containers, OpenStack and Ansible playbooks.
5. Students will chose a number of open source members of the list each and create repeatable deployments as illustrated in class.
6. The main activity of the course will be building a significant project using multiple subsystems combined with user code and data. Projects will be suggested or students can chose their own. A project report will summarize the work conducted.
7. Topics taught in this class will be very relevant for industry as you are not only exposed to big data, but you will also be practically exposed to DevOps and collaborative code development tools as part of your homework and project assignment.

Students of this class will need to conduct their project deployments in python using ansible and enabling a software stack that is useful for a big data analysis. While it is not necessary to know either python or ansible to take the class it is important that you have knowledge of a programming language so you can enhance your knowledge on them throughout the class and succeed. You will be expected to have a computer on which you have python 2.7.x installed. You will be using chameleon and possibly our local cloud. Optionally some projects may use docker.

Figure 2 illustrates that you can follow the components of the class in a variety of ways and in parallel. For example, you do not have to wait to start the project or to find out more about any of the subsystems.



Kaleidoscope of (Apache) Big Data Stack (ABDS) and HPC Technologies	
Cross-Cutting Functions 1) **Message and Data Protocols:** Avro, Thrift, Protobuf	17) **Workflow-Orchestration:** ODE, ActiveBPEL, Airavata, Pegasus, Kepler, Swift, Taverna, Triana, Trident, BioKepler, Galaxy, IPython, Dryad, Naiad, Oozie, Tez, Google FlumeJava, Crunch, Cascading, Scalding, e-Science Central, Azure Data Factory, Google Cloud Dataflow, NiFi (NSA), Jitterbit, Talend, Pentaho, Apatar, Docker Compose, KeystoneML
	16) **Application and Analytics:** Mahout , MLlib , MLbase, DataFu, R, pbdR, Bioconductor, ImageJ, OpenCV, Scalapack, PetSc, PLASMA MAGMA, Azure Machine Learning, Google Prediction API & Translation API, mlpy, scikit-learn, PyBrain, CompLearn, DAAL(Intel), Caffe, Torch, Theano, DL4j, H2O, IBM Watson, Oracle PGX, GraphLab, GraphX, IBM System G, GraphBuilder(Intel), TinkerPop, Parasol, Dream:Lab, Google Fusion Tables, CINET, NWB, Elasticsearch, Kibana, Logstash, Graylog, Splunk, Tableau, D3.js, three.js, Potree, DC.js, TensorFlow, CNTK
2) **Distributed Coordination:** Google Chubby, Zookeeper, Giraffe, JGroups	15B) **Application Hosting Frameworks:** Google App Engine, AppScale, Red Hat OpenShift, Heroku, Aerobatic, AWS Elastic Beanstalk, Azure, Cloud Foundry, Pivotal, IBM BlueMix, Ninefold, Jelastic, Stackato, appfog, CloudBees, Engine Yard, CloudControl, dotCloud, Dokku, OSGi, HUBzero, OODT, Agave, Atmosphere
	15A) **High level Programming:** Kite, Hive, HCatalog, Tajo, Shark, Phoenix, Impala, MRQL, SAP HANA, HadoopDB, PolyBase, Pivotal HD/Hawq, Presto, Google Dremel, Google BigQuery, Amazon Redshift, Drill, Kyoto Cabinet, Pig, Sawzall, Google Cloud DataFlow, Summingbird, Lumberyard
3) **Security & Privacy:** InCommon, Eduroam OpenStack Keystone, LDAP, Sentry, Sqrrl, OpenID, SAML OAuth	14B) **Streams:** Storm, S4, Samza, Granules, Neptune, Google MillWheel, Amazon Kinesis, LinkedIn, Twitter Heron, Databus, Facebook Puma/Ptail/Scribe/ODS, Azure Stream Analytics, Floe, Spark Streaming, Flink Streaming, DataTurbine
	14A) **Basic Programming model and runtime, SPMD, MapReduce:** Hadoop, Spark, Twister, MR-MPI, Stratosphere (Apache Flink), Reef, Disco, Hama, Giraph, Pregel, Pegasus, Ligra, GraphChi, Galois, Medusa-GPU, MapGraph, Totem
4) **Monitoring:** Ambari, Ganglia, Nagios, Inca	13) **Inter process communication Collectives, point-to-point, publish-subscribe:** MPI, HPX-5, Argo BEAST HPX-5 BEAST PULSAR, Harp, Netty, ZeroMQ, ActiveMQ, RabbitMQ, NaradaBrokering, QPid, Kafka, Kestrel, JMS, AMQP, Stomp, MQTT, Marionette Collective, **Public Cloud:** Amazon SNS, Lambda, Google Pub Sub, Azure Queues, Event Hubs
	12) **In-memory databases/caches:** Gora (general object from NoSQL), Memcached, Redis, LMDB (key value), Hazelcast, Ehcache, Infinispan, VoltDB, H-Store
	12) **Object-relational mapping:** Hibernate, OpenJPA, EclipseLink, DataNucleus, ODBC/JDBC
	12) **Extraction Tools:** UIMA, Tika
21 layers Over 350 Software Packages	11C) **SQL(NewSQL):** Oracle, DB2, SQL Server, SQLite, MySQL, PostgreSQL, CUBRID, Galera Cluster, SciDB, Rasdaman, Apache Derby, Pivotal Greenplum, Google Cloud SQL, Azure SQL, Amazon RDS, Google F1, IBM dashDB, N1QL, BlinkDB, Spark SQL
	11B) **NoSQL:** Lucene, Solr, Solandra, Voldemort, Riak, ZHT, Berkeley DB, Kyoto/Tokyo Cabinet, Tycoon, Tyrant, MongoDB, Espresso, CouchDB, Couchbase, IBM Cloudant, Pivotal Gemfire, HBase, Google Bigtable, LevelDB, Megastore and Spanner, Accumulo, Cassandra, RYA, Sqrrl, Neo4J, graphdb, Yarcdata, AllegroGraph, Blazegraph, Facebook Tao, Titan:db, Jena, Sesame **Public Cloud:** Azure Table, Amazon Dynamo, Google DataStore
	11A) **File management:** iRODS, NetCDF, CDF, HDF, OPeNDAP, FITS, RCFile, ORC, Parquet
January 29 2016	10) **Data Transport:** BitTorrent, HTTP, FTP, SSH, Globus Online (GridFTP), Flume, Sqoop, Pivotal GPLOAD/GPFDIST
	9) **Cluster Resource Management:** Mesos, Yarn, Helix, Llama, Google Omega, Facebook Corona, Celery, HTCondor, SGE, OpenPBS, Moab, Slurm, Torque, Globus Tools, Pilot Jobs
	8) **File systems:** HDFS, Swift, Haystack, f4, Cinder, Ceph, FUSE, Gluster, Lustre, GPFS, GFFS **Public Cloud:** Amazon S3, Azure Blob, Google Cloud Storage
Green is work of NSF14-43054	7) **Interoperability:** Libvirt, Libcloud, JClouds, TOSCA, OCCI, CDMI, Whirr, Saga, Genesis
	6) **DevOps:** Docker (Machine, Swarm), Puppet, Chef, Ansible, SaltStack, Boto, Cobbler, Xcat, Razor, CloudMesh, Juju, Foreman, OpenStack Heat, Sahara, Rocks, Cisco Intelligent Automation for Cloud, Ubuntu MaaS, Facebook Tupperware, AWS OpsWorks, OpenStack Ironic, Google Kubernetes, Buildstep, Gitreceive, OpenTOSCA, Winery, CloudML, Blueprints, Terraform, DevOpSlang, Any2Api
	5) **IaaS Management from HPC to hypervisors:** Xen, KVM, QEMU, Hyper-V, VirtualBox, OpenVZ, LXC, Linux-Vserver, OpenStack, OpenNebula, Eucalyptus, Nimbus, CloudStack, CoreOS, rkt, VMware ESXi, vSphere and vCloud, Amazon, Azure, Google and other public Clouds **Networking:** Google Cloud DNS, Amazon Route 53

Fig. 1.1: Big data relevant technology layers

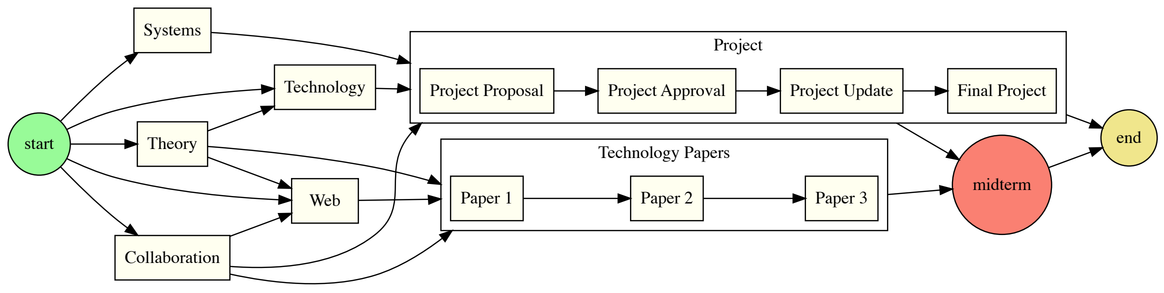


Figure 2: Components of the Class

Note: You do not have to take I523 in order to take I524.

For previous I523 class participants: While I523 is a beginners class I524 is a more advanced class and we expect that you know python which you hopefully have learned as part of I523 while doing a software project. If not, make sure you learn it before you take this class or consider **significant** additional time needed to learn it for the class.

Residential students need to enroll early so we avoid the situation like last year where we had many signing up, but did not even show up to the first lecture. I have asked that students from I523 have preference, but I am not sure if we can enforce this. So enroll ASAP. Those that are on the waiting list are recommended to show up in the first class. It is likely that you can join as others drop.

1.2 Meeting Times

The classes are published online. Residential students at Indiana University will participate in a discussion taking place at the following time:

- Monday 09:30am - 10:45am EST, I2 130

For the 100% online students see the office hours.

1.3 Online Meetings

For the zoom information please go to

https://iu.instructure.com/courses/1603897/assignments/syllabus

A doodle was used and all students that answered the doodle have times that they specified. We covered 100% the time for the students through the following schedule:

All times are in Eastern Standard Time.

Day of Week	Meetings
Monday	8-9am Office Hours 9:30-10:45am Residential Lecture 6-7pm Office Hours
Tuesday	1-2pm Office Hours 4-5pm Office Hours
Wednesday	6-7pm Office Hours
Thursday	6-7pm Office Hours (Gregor)
Friday	4-5pm Office Hours
Saturday	8-9pm Office Hours
Sunday	9-10am Office Hours 8-9pm Office Hours

1.4 Who can take the class?

- Although Undergrads can take this class it will be thaught as graduate class. Make sure you have enough time and fulfill the prerequisites such as knowing a programming language well. You need to have enough time to learn python if you do not know it.
- You can take I524 without taking I523, but you must be proficient in python. Overlap between I523 and I524 only relates to some introduction lectures and naturally lectures from the systems track such as github, report writing, introduction to python.
- Online students
- Residential students

1.5 Homework

Grading policies are listed in Table 1.

Table 1.1: Table 1: Grading

Percent	Description
10%	Class participation and contribution to Web pages.
30%	Three unique technology papers per student of the 370 systems. Each paper as at least 2 pages per technology without references.
60%	Project code and report with at least 6 pages without references. Much shorter reports will be returned without review. Do not artificially inflate contents.

- **Technology papers:** Technology papers must be non-overlapping in the entire class. As we have over 370 such technologies we should have enough for the entire class. If you see technologies missing, let us know and we see how to add them. Technology papers could be a survey of multiple technologies or an indepth analysis of a particular technology.

- **Technology paper groups:** Groups of up to three students can work also on the technology papers. However the workload is not reduced, you will produce 3 times the number of group members technology papers of unique technologies. However, you can have multiple coauthors for each paper (up to thre) that are part of your group. Please do not ask us how many technology papers you need to write if you are in a group. The rule is clearly specified. Example: Your group has 3 members, each of them has to procude 3 unique papers, thus you have to produce 9 unique technology papers for this group. If you have 2 members you have to produce 6, if you work alone you have to produce 3.
- **Technology deployment Homework:** Each student will develop as a preparation for the project a deployment of a technology. Points may depend on completeness, effort of the deployment. Technology deployments should as much as possible be non overlapping. In many cases you chose wisely such deployments may line up with your technology papers as you can add a section reporting on your achievement and experience with such deployments.
- **Project groups:** Groups of up to three students can work on a project but workload increases with each student and a work break down must be provided. More than three students are not allowed. If you work in a group you will be asked to deploy a larger system or demonstrate deployability on multiple clouds or container frameworks while benchmarking and comparing them. A group project containing 2 or 3 team members shoudl not look like a project done by an individual. Please plan careful and make sure all team members contribute.
- **Frequent checkins**: It is **important** to make frequent and often commits to the github repository as the activities will be monitored and will be integrated into the project grade. Note that paper and project will take a considerable amount of time and doing proper time management is a must for this class. Avoid starting your project late. Procrastination does not pay off.
- **No bonus projects:** This class will not have any bonus projects or regrading requests. Instead you need to focus your time on the papers and the project assignments and homework.
- **Voluntary work:** You are welcome to conduct assignments and excerises you find on the class Web page on your own. However they are not graded or considered for extra credit.
- **Late homework**: Any late homework will be receiving a 10% grade reduction. As this is a large class and the assignments are not standard multiple choice questions, grading will take a considerable time. Some homework can not be delivered late as they are related to establish communication with you. Such **deadline specific** homework will receive 0 points in case they are late. See course calendar. It is the student's responsibility to upload submissions well ahead of the deadline to avoid last minute problems with network connectivity, browser crashes, cloud issues, etc.
- **Chance for publishing a paper:** If however you find that the work you do could lead to a publishable paper, you could work together with the course instructor as coauthors to conduct such an activity. However, this is going to be a significant effort and you need to decide if you like to conduct this. In such cases if the work is sufficient for publication submission, an A+ for the class could be considered. It will be a lot of work. The length of such a paper is typically 10-12 high quality pages including figures and references. We may elect for the final submission to use a different LaTeX style

1.6 Prerequisites

We expect you are familiar with:

- Linux and the Operating system on which you will focus your deployment.
- Note that Windows as OS will not be sufficient as Ansible is not supported on it. However you can use virtualbox or log onto one of the clouds to get access to an OS that supports ansible. So you can use your Windows computer if it is powerful enough.

- Python 2.7.x (we will not use python 3 for this class as it is not yet portable with all systems) Although python is considered to be a straight forward language to learn, students that have not done any programming my find it challanging.
- Familaiarity with th Python eco system. The best way to install python on a computer is to use virtualenv, and pip (which we will teach you as part of the class).
- Familiarity with an editor such as emacs, vi, jedit, pyCharm, eclipse, or other that you can use to program in and write your reports.

If you are not familiar with these technologies, we expect that you get to know them before or during class. This may pose additional time commitment.

1.7 Open Source Publication of Homework

As this class is about open source technologies, we will be using such technologies to gather the homework submissions. We will not be using CANVAS so we teach you these technologies that are often mandated in industry. CANVAS is not.

As a consequence all technology papers from all students will be available as a single big technical report. To achieve this all reports must be written in the same format. This wil be LaTeX and all refernces have to be provided a bibtex file while you use jabref. Alternatively lyx.org can be used, if you prefer to edit latex in *what you see is almost what you get* format. The use of sharelatex or overleave or lyx.org is allowed.

1.8 Piazza

All communication will be done via Piazza. We will not read e-mail send to our university or private e-mails. All instructors are following this rule. Any mail that is not send via Piazza will be **not read** and **deleted**. This is also true for any mail send to the inbox system in CANVAS. We found CANVAS a not scalable solution for our class and will not use CANVAS for reaching out to you. If you need a different mechanism to communicate with us, please ask on Piazza how to do that. Please note that private posts in piazza are shared among all instructors and TAs.

To sign up in piazza please follow this link:

- https://piazza.com/iu/spring2017/i524

We have created a number of piazza folder to organize the posts int topics. Thes folders are:

help: Our help folder is just like a ticket system that is monitored by the TA's. Once you file a question here, a TA will attend to it, and work with you. Once the issue you had is resolved, the TA is marking it as resolved. If you need to reopen a help message, please mark it again as unresolved or post a follow up.

project: Questions related to projects are posted here.

logistics: Question regarding the logistics of the class are posted here. This includes questions about communication, meeting times, and other administrative activities.

papers: Questions regarding the paper are posted here.

grading: Questions regarding grades are posted here.

clouds: Questions regarding cloud resources are posted here.

faq: Some questions will be transformed to FAQ's that we post here. Note also that we have an FAQ on the class Web page that you may want to visit. We try to move important FAQ's from Piazza into the Web page, so it is important that you check both.

meetings: Here we will post times for meetings with TA's and instructors that are not yet posted on the Web page as part of the regular meeting times. Class participants are allowed to attend any Zoom meeting that we annonce in this folder. For online students we will also determine a time for regulare meetings. The TAs are required to hold 10 hours of meeting times upon request with you. Please make use of this.

other: In case no folder matches for your question use other.

1.9 Tips on how to achieve your best

While teaching our classes we noticed the following tips to achieve your best:

- Listening to the lectures
- **Set aside enough undisturbed time for the class**. Switch off facebook, twitter, or other distracting social media systems when focussing on the class.
- **Ask for help**. The TAs can schedule custom help office hours on appointment during reasonable times.
- Do not **Procrastinate**.
- Do not **take your other classes more serious**.
- **Start the project in the first 4 weeks of the class**
- Be aware that this class is not based on a text book and what this implies.
- Do not overestimating the technical abilities.
- Do not underestimating the time it takes to do the project.
- Do not forget to include benchmarks in your project.
- Unnecessarily struggling with LaTeX as you do not use an example we provide.
- Trying to do things just on Windows which is typically more difficult than using Linux.
- Not having a computer that is up to date. Update your memory and have a SSD
- Ignoring obvious security rules and not integrating ssh form the start into your projects.
- Not posting passwords into git. For example git does **not** allow to **easily** completely delete files that contain secret information such as passwords. It takes significant effort to do that. Make sure you do add in git on individual files and never just a bulk add.
- Having your coleagues do the work for you
- Underestimating the **time** it takes to do deployments
- Not reading our piazza posts and repeating the same question over and over
- Use Piazza to communicate and not CANVAS or e-mail.
- When you receive an e-mail from piazza, reply to it while clicking on the link instead of replying via e-mail directly. This is more reliable.

1.10 Submissions

Your papers and projects will be developed on GitHub and submitted using Pull Requests. The process is as follows:

1. fork the sp17-i524 repository.
2. clone your fork and commit and push your changes.

3. submit a pull request to the master branch of the origin repository.

See the repository for details on the individual assignments. As it will periodically be updated, make sure you are familiar with the process of Syncing a fork.

Some things to keep in mind:

- space on github is limited, so do not add datasets to the repository. Any datasets you use should be publicly hosted and deployed as part of your project ansible deployment scripts.
- never add ssh private keys to the repository. This results in a security risk, possible point deductions, and lots of time and effort to fix.
- all work will be licensed under the Apache 2 open source license.
- all submissions and discussion will be visible to the world.

1.11 Selected Project Ideas

Students can select from a number of project ideas. We will improve and add additional ideas throughout the semester. Students can also conduct their own projects. We recommend that you identify a project idea by the end of the first month of the class. Example project descriptions that you may want to take a look at include:

- robotswarm
- dockerswarm
- kubernetes
- slurmcluster
- authordisambiguity_b
- NIST Big Data Working group examples: Selected and approved use case from http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf
- Selected examples from Fall I523: Some students may have created an example as part of I523. Not all examples created as part of this class qualify for a I524 project. Please contact Gregor von Laszewski via Piazza to discuss suitability of your previous I523 project. If such a project is selected, approved and used it is expected it is significantly enhanced.
- Cloudmesh Enhancements: A number of projects could center around the enhancements of cloudmesh for the improvement of big data projects using virtual machines and containers. This includes:
  - Development of REST services for cloudmesh while using cloudmesh client
  - Development of benchmarking examples while using cloudmesh client
  - Development of a better Azure interface to additinal services
  - Development of a better AWS interfac to additinal services
  - Development of a Web interface while using django
  - SLURM integration to create virtual clusters on comet
  - Port cloudmesh client to Windows 10
  - Integrate docker into cloudmesh and demonstrate its use
  - Integrate kubernetes into cloudmesh and demonstrate its use
  - Expand the HPC capabilities of cloudmesh

1.12 Software Project

For a software project you have the choice of working indifidualy or working in a team of up to three students. You can use the **search teammate** folder to find and form groups:

- https://piazza.com/class/ix39m27czn5uw?cid=5

The following artifacts are part of the deliverables for a project

Code: You must deliver the code in github. The code must be compilable and a TA may try to replicate to run your code. You MUST avoid lengthy install descriptions and everything must be installable from the command line. We will check submission. All team members must be responsible for one part of the project.

Project Report: A report must be produced while using the format discussed in the Report Format section. The following length is required:

- 4 pages, one student in the project
- 6 pages, two students in the project
- 8 pages, three students in the project

Work Breakdown: The report contains in an appendix a section that is only needed for team projects. Include in the section a short but sufficiently detailed work breakdown documenting what the team has done. Back it up with commit information from github. Such as how many commits and lines of code a team member has contributed. The section does not count towards the overall length of the paper.

In addition the graders will check the history of checkins to verify each team member has used github to checkin their contributions frequently. E.g. if we find that one of the students has not checked in code or documentation in the same way at other teammates, it will be questioned. An oral exam may be scheduled to verify that the student has contributed to the project. In an oral exam the student must be familiar with **all** aspects of the project not just the part you contributed.

License: All projects are developed under an open source license such as Apache 2.0 License. You will be required to add a LICENCE.txt file and if you use other software identify how it can be reused in your project. If your project uses different licenses, please add in a README.md file which packages are used and which license these packages have while adding a licenses file.

Additional links:

- projects

Reproducability: The reproducability of your code will be tested twice. It is tetes by another student or team, it is also tested by a TA. A report of the testing team is provided. Your team will also be responsible for executing as many tests as you have team members on other projects. A reproducability statement should be written with details about functionality, readbility, and report quality. This statement does not have to be written in latex but uses RST.

1.13 Report Format

All reports will be using the format specified in Section reports.

There will be **NO EXCEPTION** to this format. Documents not following this format and are not professionally looking, will be returned without review.

1.14 Github repositories

Class content repository: https://github.com/cloudmesh/classes

Class homework repository: https://github.com/cloudmesh/sp17-i524

1.15 Code Repositories Deliverables

Code repositories are for code, if you have additional libraries or data that are needed you need to develop a script or use a DevOps framework to install such software. They **must** not be checked into github. Thus zip files and .class, .o, precompiled python, .exe, core dumps, and other such files files are not permissible in the project. If we find such files you will get a 20% deduction in your grade. Each project must be reproducible with a simple script. An example is:

```
git clone ....
make install
make run
make view
```

Which would use a simple make file to install, run, and view the results. Naturally you can use ansible or shell scripts. It is not permissible to use GUI based DevOps preinstalled frameworks (such as the one you may have installed in your company or as part of another project). Everything must be installable form the command line.

1.16 Learning Outcomes

Students will

1. gain broad understanding of Big Data applications and open source technologies supporting them.
2. have intense programming experience in Python and ansible and DevOps.
3. use open source technologies to manage code in large groups of individuals.
4. be able to communicate reserach in professional scientific reports.

Outcome 1 is supported by a series of lectures around open source technologies for big data.

Outcome 2 is supported by a significant software project that will take up a considerable amount of time to plan and execute.

Outcome 1 and 4 is supported by writing 3 technology papers and a project report that is shared with all students. Students can gain additional insight from reading and reviewing other students contributions.

Outcome 3 is supported by using piazza and github as well as contributiong to the class Web page with git pull requests.

1.17 Academic Integrity Policy

We take academic integrity very seriously. You are required to abide by the Indiana University policy on academic integrity, as described in the Code of Student Rights, Responsibilities, and Conduct, as well as the Computer Science Statement on Academic Integrity (http://www.soic.indiana.edu/doc/graduate/graduate-forms/Academic-Integrity-Guideline-FINAL-2015.pdf). It is your responsibility to understand these policies. Briefly summarized, the work you submit for course assignments, projects, quizzes, and exams must be your own or that of your group, if group work is permitted. You may use the ideas of others but you must give proper credit. You may discuss assignments with other students but you must acknowledge them in the reference section according to scholarly citation rules. Please also make sure that you know how to not plagiarize text from other sources while reviewing citation rules.

We will respond to acts of plagiarism and academic misconduct according to university policy. Sanctions typically involve a grade of 0 for the assignment in question and/or a grade of F in the course. In addition, University policy requires us to report the incident to the Dean of Students, who may apply additional sanctions, including expulsion from the university.

Students agree that by taking this course, papers and source code submitted to us may be subject to textual similarity review, for example by Turnitin.com. These submissions may be included as source documents in reference databases for the purpose of detecting plagiarism of such papers or codes.

It is not acceptable to use for pay services to conduct your project. Please be aware that we monitor such services and have TAs speaking various languages and know about these services even in different countries. Also do not just translate a report written by someone in a different language and claim it to be your project.

1.18 Links

This page is conveniently managed with git. The location for the changes can be found at

- https://cloudmesh.github.io/classes/

The repository is at

- https://github.com/cloudmesh/classes

Issues can be submitted at

- https://github.com/cloudmesh/classes/issues

Or better use piazza so you notify us in our discussion lists.

- https://piazza.com/iu/i524

If you detect errors, you could also create a merge request at

- https://github.com/cloudmesh/classes

1.19 Course Numbers

This course is offered for Graduate (and Undergraduate students with permission) at Indiana University and as an online course. To Register, for University credit please go to:

- http://registrar.indiana.edu/browser/soc4172/INFO/INFO-I524.shtml
- http://registrar.indiana.edu/browser/soc4172/ENGR/ENGR-E599.shtml

Please, select the course that is most suitable for your program:

```
INFO-I 524  BIG DATA SOFTWARE AND PROJECTS (3 CR)     Von Laszewski G
         Above class open to graduates only
         Above class taught online
         Discussion (DIS)
      30672 RSTR     09:30A-10:45A   M      I2 130    Von Laszewski G
         Above class meets with ENGR-E 599
INFO-I 524  BIG DATA SOFTWARE AND PROJECTS (3 CR)
      30673 RSTR     ARR              ARR    ARR       Von Laszewski G
         Above class open to graduates only
```

```
        This is a 100% online class taught by IU Bloomington. No
        on-campus class meetings are required. A distance education
        fee may apply; check your campus bursar website for more
        information

ENGR-E 599  TOPICS IN INTELL SYS ENGINEER (3 CR)
      VT: BIG DATA SOFTWARE AND PROJECTS
         ***** RSTR     ARR             ARR    ARR       Von Laszewski G
            Discussion (DIS)
      VT: BIG DATA SOFTWARE AND PROJECTS
         33924 RSTR     09:30A-10:45A   M      I2 130    Von Laszewski G
            Above class meets with INFO-I 524
```

1.20 Refernces

http://hpc-abds.org/kaleidoscope/

CHAPTER

TWO

I524 CALENDAR

Warning: This calendar may be updated based on experience from the class. Please check back here.

Residential classes meet Mondays 09:30A-10:45A, I2 130

Description	**Date/Due Dates**	No
Class Begins	Mon, Jan 9	
Assignment of HID	Fri, Jan 13	
Piazza access verified (HD)	Mon, Jan 16, 9am	C1
Surveys (HD)	Mon, Jan 16, 9am	C2
MLK Jr. Day. Good time for additional class work	Mon, Jan 16	
Github access (needed for TechList)	Mon, Jan 30, 9am	C3
TechList.1a - 1.c	Mon, Jan 30, 9am	T0a
Technology Paper 1	Mon, Jan 30, 9am	T1
Acces and use of cloud verified (HD)	Mon, Jan 30, 9am	C4
Acces to python 2.7.x verified (HD)	Mon, Jan 30, 9am	C5
TechList.1d and Techlist 2	Mon, Feb 27, 9am	T0b
Technology Paper 2 due	Mon, Feb 27, 9am	T2
Technology Paper 3 due	Mon, Mar 27, 9am	T3
Auto Witdraw	Sun, Mar 12	
Project Execution plan and draft due (HD)	Mon, Mar 13, 9am	P1
Spring Break. Good time for additional class work	Mar 12 - Mar 19	
Project Updates due (HD)	Mon, Mar 27, 9am	P2
Project due	Mon, Apr 24, 9am	P3
Last day to submit late Homework	May 1	
Ends	Fri, May 5	

- (HD) hard deadlines must be done in order to obtain full points. These deadlines are important to assure you have access to the resources for the class.

2.1 Comments

- Any late homework will have an automatic 10% grade deduction.
- Any late homework may result in substential delay in grading (one month or more).
- Hard deadlines can not receive any points for late submissions as they are essential to the communication and operation of the class. If you can naturally not communicate with we can not review your work or you can not even execute your work.

- Experience shows that those using additional time during the spring break do typically better. We recommend that you use this time wisely.
- You can start earlier if you like to prepare for this class, to for example learn Python and ansible. However, lectures may change.
- It would be a mistke not to start working on your project by February 1st. You will run out of time. In order to accomodate for this we have segnificantly reduced other homework requirements in contrast to previous classes we taught.

2.2 Official University calendar

- http://registrar.indiana.edu/official-calendar/official-calendar-spring.shtml

CHAPTER

THREE

I524 LECTURES

Warning: This page is under construction, but most lectures are already available. All tracks will change considerably. If you want to work ahead, start with the theory track.

Warning: Lectures listed on this page will be ready for review when they are marked as released. If they are not released they will be updated as part of the class. However instead of waiting for the release, we have opted to show you our current draft lectures.

Based on our experience with residential and online classes we will for the first time not require that you have to do the class videos at a prticular time once they are released. This however has the danger that you are not watching them at all and you cheat yourself as you do not allow yourself the educational lessons that this class offers to you. It also requires you to assemble your own schedule for watching the videos that will have to be managed through github as part of a README.md file in your git repository. You will need to do the technology track, the communications track, as well as the theory track.

Theory Track: Some lectures have been designed to introduce you to a number of technologies. These lectures are of more theoretical nature and do not require much hands on activities. THus you can start them any time.

Collaboration Track: These lectures provide the tools for you to collaborate with your peers and with instructors.

Systems Track: These lectures cover topics that are fundamental to executing your project.

Technology Track: These are lectures with strong technology content and introduce you to using a selected number of technologies as part of the class. It is expected that you will use them as part of the project. Instead of slowing you down with graded homework we expect that you learn these technologies and reuse them as part of the project. It would be a big mistake to start the project 2 weeks befor the semester ends, you will not succeed. You must start your project in the first month of the course. Progress is reported on monthly basis while the report is updated and snapshot every month. We will monitor your progress and include them into the discussion grade. For residential students ther should be no reason why you can not provide a monthly update. For online students a valid update would be: "I changed my company and could not work on the project due to moving". This will give you some points if submitted in time. However, if you submit nothing, we will not issue any points.

3.1 Lectures

3.2 Lectures - Theory Track

Table 3.1: Theory Track

Topic	Description	Resources	Length
Overview	Course Overview	Slides	
	Class Overview - Part 1	Video	11:29
	Class Overview - Part 2	Video	04:10
	Class Overview - Part 3	Video	12:41
Web Page	Course Web Page	**Videos_**	
	Class Web Page - Part 1	Video	11:25
	Class Web Page - Part 2	Video	17:31
Techlist.1	TechList.1 Web Page	Web Page	
	TechList.1 Homework	Video	40:08
Introduction	Course Introduction	Slides	
	Introduction	Video	0:13:59
	Introduction - Real World Big Data	Video	0:15:28
	Introduction - Basic Trends and Jobs	Video	0:10:57
Acess Patterns	Data Access Patterns and Introduction to using HPC-ABDS	Slides	
	1. Introduction to HPC-ABDS Software and Access Patterns	Video - Resource 1	0:27:45
	2. Science Examples (Data Access Patterns)	Video - Resource 2	0:18:38
	3. Remaining General Access Patterns	Video	0:11:26
	4. Summary of HPC-ABDS Layers 1 - 6	Video	0:14:32
	5. Summary of HPC-ABDS Layers 7 - 13	Video	0:30:52
	6. Summary of HPC-ABDS Layers 14 - 17	Video	0:28:02
	Final Part Summary of Stack	Video	0:20:20

3.3 Lectures - Collaboration Track

Table 3.2: Collaboration Track

Topic	Description	Resources	Length
Github	Overview and Introduction	Web page	
	Install Instructions	Web page	
	config	Video	2:47
	fork	Video	1:41
	checkout	Video	3:11
	pull	Video	4:26
	branch	Video	2:25
	merge	Video	4:50
	rebase	Video	4:20
	GUI	Video	3:47
	Windows - unsupported	Video	1:25
Paper	How to write a paper by Simon Peyton Jones	Video	34:24

3.4 Unreleased Lectures

A list of unrelease lectures that we are currently working on is available here: ref-unreleased

CHAPTER

FOUR

ASSIGNMENTS OF HIDS TO TECHNOLOGIES

Table 4.1: Mappings of HIDs to Techs

HID	Technologies
S17-ER-1000	Mbase (1); Hyper-V (1) – Kibana (1) – Yarcdata (1) – CDMI (1) – Kepler (1)
S17-ER-1001	Azure Queues (1) – Sentry (1) – Tableau (1) – Berkeley DB (1) – ODE (1) – OpenStack Keystone (1)
S17-ER-1002	Crunch (1) – point-to-point (1) – LinkedIn (1) – Pig (1) – Hive (1) – IPython (1)
•	•
S17-IO-3000	SQL Server (1) – Nimbus (1) – Taverna (1) – Chef (1) – Tyrant (1) – FITS (1)
S17-IO-3001	Medusa-GPU (1) – Jupyter (1) – OpenJPA (1) – Google and other public Clouds (1) – Airavata (1)
S17-IO-3002	TensorFlow (1) – Azure Stream Analytics (1) – Ambari (1) – Galaxy (1) – Bioconductor (1)
S17-IO-3003	QPid (1) – Stomp (1) – Apatar (1) – Google FlumeJava (1) – Sqrrl (1)
S17-IO-3004	appfog (1) – Dream:Lab (1) – MySQL (1) – ZHT (1) – RYA (1)
S17-IO-3005	Amazon Kinesis (1) – Inca (1) – Gora (general object from NoSQL) (1) – RabbitMQ (1) – JClouds (1)
S17-IO-3006	Tez (1) – Ubuntu MaaS (1) – Llama (1) – GraphChi (1) – Lumberyard (1)
S17-IO-3007	Google Cloud SQL (1) – Ganglia (1) – NaradaBrokering (1) – Event Hubs (1) – Kite (1)
S17-IO-3008	CINET (1) – Linux-Vserver (1) – Networking: Google Cloud DNS (1) – Talend (1) – Haystack (1)
S17-IO-3009	Flink Streaming (1) – Solr (1) – JGroups (1) – Azure SQL (1) – HDF (1)
S17-IO-3010	Azure (1) – Couchbase (1) – Public Cloud: Azure Table (1) – Sawzall (1) – Phoenix (1)
S17-IO-3011	ZeroMQ (1) – Blueprints (1) – Trident (1) – e-Science Central (1) – Winery (1)
S17-IO-3012	MRQL (1) – AWS OpsWorks (1) – GPFS (1) – Hazelcast (1) – Google Bigtable (1)

Continued on next page

Table 4.1 – continued from previous page

HID	Technologies
S17-IO-3013	Google Prediction API and Translation API (1) – LMDB (key value) (1) – QEMU (1) – BioKepler (1) – Google Cloud Dataflow (1)
S17-IO-3014	Riak (1) – Ehcache (1) – Xen (1) – Zookeeper (1) – SSH (1)
S17-IO-3015	Lucene (1) – pbdR (1) – Protobuf (1) – Galera Cluster (1) – Cassandra (1)
S17-IO-3016	DC.js (1) – Aerobatic (1) – CoreOS (1) – AMQP (1) – Argo BEAST HPX-5 BEAST PULSAR (1)
S17-IO-3017	Facebook Tao (1) – MongoDB (1) – Amazon (1) – Kafka (1) – Amazon Dynamo (1)
S17-IO-3018	PostgreSQL (1) – Impala (1) – Hadoop (1) – Floe (1) – VirtualBox (1)
S17-IO-3019	Stackato (1) – Parasol (1) – vSphere and vCloud (1) – Totem (1) – Libvirt (1)
S17-IO-3020	AppScale (1) – CloudControl (1) – Google Fusion Tables (1) – Yarn (1) – TinkerPop (1)
S17-IO-3021	CUBRID (1) – MR-MPI (1) – NWB (1) – Cascading (1) – BitTorrent (1)
S17-IO-3022	Juju (1) – Netty (1) – FUSE (1) – Google Chubby (1) – Mesos (1)
S17-IO-3023	H-Store (1) – Kyoto Cabinet (1) – Globus Online (GridFTP) (1) – Sahara (1) – DataFu (1)
S17-IO-3024	NiFi (NSA) (1) – LXC (1) – Helix (1) – IBM dashDB (1) – Puppet (1)
S17-IO-3025	Voldemort (1) – Buildstep (1) – OCCI (1) – SAP HANA (1) – HPX-5 (1)
•	•
S17-IR-2000	OPeNDAP (1) – CloudMesh (1) – Rocks (1) – Swift (1) – Infinispan (1)
S17-IR-2001	Celery (1) – GraphBuilder(Intel) (1) – HTCondor (1) – HUBzero (1) – Gitreceive (1)
S17-IR-2002	Stratosphere (Apache Flink) (1) – ActiveBPEL (1) – Google Dremel (1) – ImageJ (1) – IBM Cloudant (1)
S17-IR-2003	DAAL(Intel) (1) – Hibernate (1) – PyBrain (1) – CouchDB (1) – OpenStack (1)
S17-IR-2004	PLASMA MAGMA (1) – Samza (1) – Azure Blob (1) – OpenVZ (1) – Jelastic (1)
S17-IR-2005	N1QL (1) – Google App Engine (1) – SciDB (1) – Triana (1) – OpenStack Ironic (1)
S17-IR-2006	Cinder (1) – Spark (1) – R (1) – dotCloud (1) – Pivotal Gemfire (1)
S17-IR-2007	DevOpSlang (1) – Google BigQuery (1) – OSGi (1) – OpenPBS (1) – Lambda (1)
S17-IR-2008	Kestrel (1) – Scalapack (1) – HadoopDB (1) – OODT (1) – Thrift (1)

Continued on next page

Table 4.1 – continued from previous page

HID	Technologies
S17-IR-2009	Megastore and Spanner (1) – Pivotal GPLOAD/GPFDIST (1) – Facebook Corona (1) – Apache Derby (1) – SQLite (1)
S17-IR-2010	MapGraph (1) – three.js (1) – Amazon Redshift (1) – (Dryad) (1) – Disco (1)
S17-IR-2011	DL4j (1) – Solandra (1) – CloudStack (1) – Logstash (1) – Ansible (1)
S17-IR-2012	GraphLab (1) – GFFS (1) – Lustre (1) – Reef (1) – Harp (1)
S17-IR-2013	Flume (1) – OpenCV (1) – ORC (1) – VMware ESXi (1) – Hama (1)
S17-IR-2014	Sqoop (1) – Pivotal (1) – Google MillWheel (1) – iRODS (1) – VoltDB (1)
S17-IR-2015	Facebook Tupperware (1) – Neo4J (1) – Databus (1) – KeystoneML (1) – Rasdaman (1)
S17-IR-2016	ActiveMQ (1) – OpenTOSCA (1) – Avro (1) – SaltStack (1) – Whirr (1)
S17-IR-2017	Eduroam (1) – Potree (1) – Pivotal HD/Hawq (1) – Docker Compose (1) – OpenNebula (1)
S17-IR-2018	CompLearn (1) – OpenID (1) – Cisco Intelligent Automation for Cloud (1) – Pentaho (1) – scikit-learn (1)
S17-IR-2019	Oracle (1) – CNTK (1) – Twister (1) – NetCDF (1) – Oozie (1)
S17-IR-2020	Summingbird (1) – Google F1 (1) – Torque (1) – DataTurbine (1) – Scalding (1)
S17-IR-2021	Docker Machine and Swarm (1) – Shark (1) – Ligra (1) – Redis (1) – Facebook Puma/Ptail/Scribe/ODS (1)
S17-IR-2022	Sesame (1) – Pilot Jobs (1) – Red Hat OpenShift (1) – Google Pub Sub (1) – Boto (1)
S17-IR-2023	SGE (1) – Marionette Collective (1) – Neptune (1) – Amazon Route 53 (1) – Splunk (1)
S17-IR-2024	Cobbler (1) – GraphX (1) – Memcached (1) – graphdb (1) – LDAP (1)
S17-IR-2025	Graylog (1) – InCommon (1) – BlinkDB (1) – Moab (1) – Globus Tools (1)
S17-IR-2026	Ceph (1) – CDF (1) – Jitterbit (1) – Naiad (1) – publish-subscribe: MPI (1)
S17-IR-2027	DataNucleus (1) – Razor (1) – Twitter Heron (1) – Amazon RDS (1) – SAML OAuth (1)
S17-IR-2028	Spark Streaming (1) – Libcloud (1) – Google Kubernetes (1) – mlpy (1) – Dokku (1)
S17-IR-2029	Galois (1) – Slurm [Slu] (1) – Giraffe (1) – Azure Machine Learning (1) – Ninefold (1)
S17-IR-2030	UIMA (1) – Jena (1) – Tycoon (1) – Azure Data Factory (1) – Google Cloud DataFlow (1)
S17-IR-2031	Google Cloud Storage (1) – EclipseLink (1) – Torch (1) – Caffe (1) – Parquet (1)
S17-IR-2032	AWS Elastic Beanstalk (1) – PolyBase (1) – Pivotal Greenplum (1) – Blazegraph (1) – S4 (1)

Continued on next page

Table 4.1 – continued from previous page

HID	Technologies
S17-IR-2033	Spark SQL (1) – LevelDB (1) – Google DataStore (1) – IBM System G (1) – CloudML (1)
S17-IR-2034	rkt (1) – Heroku (1) – Pegasus (1) – Drill (1) – Titan:db (1)
S17-IR-2035	ODBC/JDBC (1) – f4 (1) – Oracle PGX (1) – Eucalyptus (1) – D3.js (1)
S17-IR-2036	OpenStack Heat (1) – Saga (1) – Agave (1) – Storm (1) – JMS (1)
S17-IR-2037	Public Cloud: Amazon SNS (1) – FTP (1) – HBase (1) – MQTT (1) – RCFile (1)
S17-IR-2038	Terraform (1) – H2O (1) – KVM (1) – Cloud Foundry (1) – CloudBees (1)
S17-IR-2039	TOSCA (1) – HTTP (1) – IBM BlueMix (1) – Google Omega (1) – Gluster (1)
S17-IR-2040	Xcat (1) – DB2 (1) – Engine Yard (1) – MLlib (1) – Accumulo (1)
S17-IR-2041	IBM Watson (1) – Public Cloud: Amazon S3 (1) – Kyoto/Tokyo Cabinet (1) – Elasticsearch (1) – Tajo (1)
S17-IR-2042	PetSc (1) – Any2Api (1) – Espresso (1) – Giraph (1) – Pregel (1)
S17-IR-2044	AllegroGraph (1) – Theano (1) – Atmosphere (1) – Granules (1) – HDFS (1)
•	•
S17-TS-0001	Mahout (1)
S17-TS-0001	Tika (1)
S17-TS-0003	HCatalog (1)
S17-TS-0004	Foreman (1)
S17-TS-0005	Genesis (1)
S17-TS-0006	Presto (1)
S17-TS-0007	Nagios (1)

CHAPTER

FIVE

TECHNOLOGIES

In this section we find a number of technologies that are related to big data. Certainly a number of these projects are hosted as an Apache project. One important resource for a general list of all apache projects is at

- Apache projects: https://projects.apache.org/projects.html?category

5.1 Workflow-Orchestration

1. ODE
2. ActiveBPEL
3. Airavata
4. Pegasus
5. Kepler
6. Swift
7. Taverna

   Taverna is workflow management system. According to *[Tav]*, Taverna is transitioning to Apache Incubator as of Jan 2017. Taverna suite includes 2 products:

   (1). Taverna Workbench is desktop client where user can define the workflow. (2). Taverna Server is responsible for executing the remote workflows.

   Taverna workflows can also be executed on command-line. Taverna supports wide range of services including WSDL-style and RESTful Web Services, BioMart, SoapLab, R, and Excel. Taverna also support mechanism to monitor the running workflows using its web browser interface. In his *[tav07]* paper, Daniele Turi presented the formal syntax and operational semantics of Taverna.
8. Triana
9. Trident
10. BioKepler
11. Galaxy
12. IPython
13. Jupyter
14. (Dryad)
15. Naiad
16. Oozie

17. Tez
18. Google FlumeJava
19. Crunch
20. Cascading
21. Scalding
22. e-Science Central
23. Azure Data Factory
24. Google Cloud Dataflow
25. NiFi (NSA)
26. Jitterbit
27. Talend
28. Pentaho
29. Apatar
30. Docker Compose
31. KeystoneML

5.2 Application and Analytics

32. Mahout *[Mah]*

    "Apache Mahout software provides three major features: (1) A simple and extensible programming environment and framework for building scalable algorithms (2) A wide variety of premade algorithms for Scala + Apache Spark, H2O, Apache Flink (3) Samsara, a vector math experimentation environment with R-like syntax which works at scale"

33. MLlib
34. Mbase
35. DataFu

    The Apache DataFu project was created out of the need for stable, well-tested libraries for large scale data processing in Hadoop. As detailed in *[Dat]* Apache DatFu consists of two libraries Apache DataFu Pig and Apache DataFu Hourglass. Apache DataFu Pig is a collection of useful user-defined functions for data analysis in Apache Pig. The functions are in areas of Statistics, Bag Operations, Set Operations, Sessions, Sampling, Estimation, Hashing and Link Analysis. Apache DataFu Hourglass is a library for incrementally processing data using Hadoop MapReduce. It is designed to make computations over sliding windows more efficient. For these types of computations, the input data is partitioned in some way, usually according to time, and the range of input data to process is adjusted as new data arrives. Hourglass works with input data that is partitioned by day, as this is a common scheme for partitioning temporal data.

36. R
37. pbdR

    Programming with Big Data in R (pbdR) *[OCSP12]* is an environment having series of R packages for statistical computing with Big Data using high-performance statistical computation. It uses R, a popular language between statisticians and data miners. "pbdR" focuses on distributed memory system, where data is distributed accross several machines and processed in batch mode. It uses MPI for inter process communications. R focuses on single machines for data analysis using a interactive GUI. Currenly there are two implementation of pbdR, one Rmpi and another being pdbMpi. Rmpi uses SPMD parallelism while pbdRMpi uses manager/worker parallelism.

38. Bioconductor
39. ImageJ
40. OpenCV
41. Scalapack
42. PetSc
43. PLASMA MAGMA
44. Azure Machine Learning
45. Google Prediction API & Translation API
46. mlpy
47. scikit-learn
48. PyBrain
49. CompLearn
50. DAAL(Intel)
51. Caffe
52. Torch
53. Theano
54. DL4j
55. H2O
56. IBM Watson
57. Oracle PGX
58. GraphLab
59. GraphX
60. IBM System G
61. GraphBuilder(Intel)
62. TinkerPop
63. Parasol
64. Dream:Lab
65. Google Fusion Tables
66. CINET
67. NWB
68. Elasticsearch
69. Kibana
70. Logstash
71. Graylog

72. Splunk
73. Tableau
74. D3.js
75. three.js
76. Potree
77. DC.js
78. TensorFlow
79. CNTK

5.3 Application Hosting Frameworks

80. Google App Engine *[App]*

    On purpose we put in here a “good” example of a bad entry that woudl receive 10 out of 100 points, e.g. an F:

    “Google App Engine” provides platform as a service. There are major advantages from this framework:

    (a) Scalable Applications

    (b) Easier to maintain

    (c) Publishing services easily

    Reasons: (a) “major advantages is advertisement” if you add word major (b) grammar needs to be improved (c) the three points do not realy say anything about Google App Engine (d) the reader will after reading this have not much information about what it is (e) a refernce is not included. (f) enumeration should be in this page avoided. We like to see a number of paragraphs with text.

    Note: This is an example for a bad entry

81. AppScale
82. Red Hat OpenShift
83. Heroku
84. Aerobatic
85. AWS Elastic Beanstalk
86. Azure

    Microsoft Corporation markets its cloud products under the *Azure* brand name. At its most basic, Azure acts as an *infrastructure-as-a-service* (IaaS) provider. IaaS virtualizes hardware components, a key differentiation from other *-as-a-service* products. The Wikipedia entry on IaaS notes that IaaS “abstract[s] the user from the details of infrasctructure like physical computing resources, location, data partitioning, scaling, security, backup, etc.” :cite:www-wikipedia-cloud

    However, Azure offers a host of closely-related tool and products to enhance and improve the core product, such as raw block storage, load balancers, and IP addresses *[MicrosoftCorp]*. For instance, Azure users can access predictive analytics, Bots and Blockchain-as-a-Service :cite:www-azure-msft as well as more-basic computing, networking, storage, database and management components *[MicrosoftCorp16]*. The Azure website shows twelve major categories under *Products* and twenty *Solution* categories, e.g., e-commerce or Business SaaS apps.

Azure competes against Amazon's *Amazon Web Service*, :cite:www-aws-amzn even though IBM (*SoftLayer* :cite:www-softlayer-ibm and *Bluemix* :cite :*www-bluemix-ibm*) and Google (*Google Cloud Platform*) [www-cloud- google] offer IaaS to the market. As of January 2017, Azure's datacenters span 32 Microsoft-defined *regions*, or 38 *declared regions*, throughout the world. *[MicrosoftCorp]*

87. Cloud Foundry
88. Pivotal
89. IBM BlueMix
90. (Ninefold)

    no longer active

91. Jelastic
92. Stackato
93. appfog
94. CloudBees
95. Engine Yard
96. (CloudControl)

    No Longer active as of Feb. 2016

97. dotCloud
98. Dokku
99. OSGi
100. HUBzero
101. OODT
102. Agave
103. Atmosphere

5.4 High level Programming

104. Kite
105. Hive
106. HCatalog
107. Tajo
108. Shark
109. Phoenix

     In the first quarter of 2013, Salesforce.com released its proprietary SQL-like interface and query engine for HBase, *Phoenix*, to the open source community. The company appears to have been motivated to develop Phoenix as a way to 1) increase accessiblity to HBase by using the industry-standard query language (SQL); 2) save users time by abstracting away the complexities of coding native HBase queries; and, 3) implementing query best practices by implementing them automatically via Phoenix. *[Kes13]* Although Salesforce.com initially *open-sourced* it via Github, by May of 2014 it had become a top-level Apache project. *[Anonb]*

Phoenix, written in Java, “compiles [SQL queries] into a series of HBase scans, and orchestrates the running of those scans to produce regular JDBC result sets.” *[Anona]* In addition, the program directs compute intense portions of the calls to the server. For instance, if a user queried for the top ten records across numerous regions from an HBase database consisting of a billion records, the program would first select the top ten records for each region using server-side compute resources. After that, the client would be tasked with selecting the overall top ten. [www-phoenix- salesforcedev]

Despite adding an abstraction layer, Phoenix can actually speed up queries because it optimizes the query during the translation process. [www- phoenix-cloudera] For example, “Phoenix beats Hive for a simple query spanning 10M-100M rows.” *[Avr13]*

Finally, another program can enhance HBase’s accessibility for those inclined towards graphical interfaces. SQuirell only requires the user to set up the JDBC driver and specify the appropriate connection string. [www-phoenix- bighadoop]

110. Impala
111. MRQL
112. SAP HANA
113. HadoopDB
114. PolyBase
115. Pivotal HD/Hawq
116. Presto

Presto *[wwwa]* is an open-source distributed SQL query engine that supports interactive analytics on large datasets. It allows interfacing with a variety of data sources such as Hive, Cassandra, RDBMSs and proprietary data source. Presto is used at a number of big-data companies such as Facebook, Airbnb and Dropbox. Presto’s performance compares favorably to similar systems such as Hive and Stinger *[CQB+14]*.

117. Google Dremel
118. Google BigQuery
119. Amazon Redshift
120. Drill
121. Kyoto Cabinet

Kyoto Cabinet as specified in *[Kyo]* is a library of routines for managing a database which is a simple data file containing records. Each record in the database is a pair of a key and a value. Every key and value is serial bytes with variable length. Both binary data and character string can be used as a key and a value. Each key must be unique within a database. There is neither concept of data tables nor data types. Records are organized in hash table or B+ tree. Kyoto Cabinet runs very fast. The elapsed time to store one million records is 0.9 seconds for hash database, and 1.1 seconds for B+ tree database. Moreover, the size of database is very small. The, overhead for a record is 16 bytes for hash database, and 4 bytes for B+ tree database. Furthermore, scalability of Kyoto Cabinet is great. The database size can be up to 8EB (9.22e18 bytes).

122. Pig
123. Sawzall
124. Google Cloud DataFlow
125. Summingbird
126. Lumberyard

5.5 Streams

127. Storm
128. S4
129. Samza
130. Granules
131. Neptune
132. Google MillWheel
133. Amazon Kinesis

     Kinesis is Amazon's *[wwwf]* real time data processing engine. It is designed to provide scalable, durable and reliable data processing platform with low latency. The data to Kinesis can be ingested from multiple sources in different format. This data is further made available by Kinesis to multiple applications or consumers interested in the data. Kinesis provides robust and fault tolerant system to handle this high volume of data. Data sharding mechanism is Kinesis makes it horizontally scalable. Each of these shards in Kinesis process a group of records which are partitioned by the shard key. Each record processed by Kinesis is identified by sequence number, partition key and data blob. Sequence number to records is assigned by the stream. Partition keys are used by partitioner(a hash function) to map the records to the shards i.e. which records should go to which shard. Producers like web servers, client applications, logs push the data to Kinesis whereas Kinesis applications act as consumers of the data from Kinesis engine. It also provides data retention for certain time for example 24 hours default. This data retention window is a sliding window. Kinesis collects lot of metrics which can used to understand the amount of data being processed by Kinesis. User can use this metrics to do some analytics and visualize the metrics data. Kinesis is one of the tools part of AWS infrastructure and provides its users a complete software-as-a-service. Kinesis *[SS16]* in the area of real-time processing provides following key benefits: ease of use, parellel processing, scalable, cost effective, fault tolerant and highly available.

134. LinkedIn
135. Twitter Heron
136. Databus
137. Facebook Puma/Ptail/Scribe/ODS
138. Azure Stream Analytics
139. Floe
140. Spark Streaming
141. Flink Streaming
142. DataTurbine

5.6 Basic Programming model and runtime, SPMD, MapReduce

143. Hadoop
144. Spark
145. Twister
146. MR-MPI
147. Stratosphere (Apache Flink)

148. Reef
149. Disco
150. Hama
151. Giraph
152. Pregel
153. Pegasus
154. Ligra
155. GraphChi
156. Galois
157. Medusa-GPU
158. MapGraph
159. Totem

5.7 Inter process communication Collectives

160. point-to-point
161. publish-subscribe: MPI
162. HPX-5
163. Argo BEAST HPX-5 BEAST PULSAR
164. Harp
165. Netty
166. ZeroMQ
167. ActiveMQ
168. RabbitMQ

    RabbitMQ is a message broker *[wwwh]* which allows services to exchange messages in a fault tolerant manner. It provides variety of features which "enables software applications to connect and scale". Features are: reliability, flexible routing, clustering, federation, highly available queues, multi-protocol, many clients, management UI, tracing, plugin system, commercial support, large community and user base. RabbitMQ can work in multiple scenarios:

    (a) Simple messaging: producers write messages to the queue and consumers read messages from the the queue. This is synonymous to a simple message queue.

    (b) Producer-consumer: Producers produce messages and consumers receive messages from the queue. The messages are delivered to multiple consumers in round robin manner.

    (c) Publish-subscribe: Producers publish messages to exchanges and consumers subscribe to these exchanges. Consumers receive those messages when the messages are available in those exchanges.

    (d) Routing: In this mode consumers can subscribe to a subset of messages instead of receiving all messages from the queue.

    (e) Topics: Producers can produce messages to a topic multiple consumers registered to receive messages from those topics get those messages. These topics can be handled by a single exchange or multiple exchanges.

(f) RPC:In this mode the client sends messages as well as registers a callback message queue. The consumers consume the message and post the response message to the callback queue.

RabbitMQ is based on AMPQ *[OHara07]* (Advanced Message Queuing Protocol) messaging model. AMPQ is described as follows "messages are published to exchanges, which are often compared to post offices or mailboxes. Exchanges then distribute message copies to queues using rules called bindings. Then AMQP brokers either deliver messages to consumers subscribed to queues, or consumers fetch/pull messages from queues on demand"

169. NaradaBrokering
170. QPid
171. Kafka

Apache Kafka is a streaming platform, which works based on publish-subscribe messaging system and supports distributed environment. Kafka lets publish and subscribe to the messages.

In a publish-subscribe messaging system, publishers are sender of messages. They publish the messages without the knowledge of who is going to 'subscribe' to them for processing. Subscribers are users of these messages. They subscribe to only those messages which they are interested in, without knowing who the publishers are. Kafka maintains message feeds based on 'topic'. A topic is a category or feed name to which records are published. Applications can use Kafka's Connector APIs to publish the messages to one or more Kafka topics. Similarly, applications can use Consumer API to subscribe to one or more topics. Kafka has the capability to process the stream of data at real time.

Kafka's stream processor takes continual stream of data from input topics, processes the data in real time and produces streams of data to output topics. Kafka's Streams API are used for data transformation. Kafka allows to store the stream of data in distributed clusters.

Kafka acts as a storage system for incoming data stream. Data is categorised into 'topics'. As Kafka is a distributed system, data streams are partitioned and replicated across nodes. Thus, a combination of messaging, storage and processing data stream makes Kafka a 'streaming platform'.

Kafka is a commonly used for building data pipelines where data is transferred between systems or applications. *[Fou]* Kafka can also be used by applications that transform real time incoming data.

172. Kestrel
173. JMS
174. AMQP
175. Stomp
176. MQTT
177. Marionette Collective
178. Public Cloud: Amazon SNS
179. Lambda
180. Google Pub Sub
181. Azure Queues
182. Event Hubs

5.8 In-memory databases/caches

183. Gora (general object from NoSQL)

Gora is a in-memory data model *[wwwc]* which also provides persistence to the big data. Gora provides persistence to different types of data stores. Primary goals of Gora are:

(a) data persistence

(b) indexing

(c) data access

(d) analysis

(e) map reduce support

Unlike ORM models which mostly work with relational databases for example hibernate gora works for most type of data stores like documents, columnar, key value as well as relational. Gora uses beans to maintain the data in-memory and persist it on disk. Beans are defined using apache avro schema. Gora provides modules for each type of data store it supports. The mapping between bean definition and datastore is done in a mapping file which is specific to a data store. Type Gora workflow will be:

(a) define the bean used as model for persistence

(b) use gora compiler to compile the bean

(c) create a mapping file to map bean definition to datastore

(d) update gora.properties to specify the datastore to use

(e) get an instance of corresponding data store using datastore factory.

Gora has a query interface to query the underlying data store. Its configuration is stored in gora.properties which should be present in classpath. In the file you can specify default data store used by Gora engine. Gora also has a CI/CD library call GoraCI which is used to write integration tests.

184. Memcached

185. Redis

186. LMDB (key value)

187. Hazelcast

188. Ehcache

189. Infinispan

190. VoltDB

191. H-Store

H-Store is an in memory and parallel database management system for on-line transaction processing (OLTP). Specifically , *[HSta]* illustrates that H-Store is a highly distributed, row-store-based relational database that runs on a cluster on shared-nothing, main memory executor nodes.As Noted in *[KKN+08]* “the architectural and application shifts have resulted in modern OLTP databases increasingly falling short of optimal performance.In particular, the availability of multiple-cores, the abundance of main memory, the lack of user stalls, and the dominant use of stored procedures are factors that portend a clean-slate redesign of RDBMSs”.The H-store which is a complete redesign has the potential to outperform legacy OLTP databases by a significant factor. As detailed in *[HStb]* H-Store is the first implementation of a new class of parallel DBMS, called NewSQL, that provides the high-throughput and high-availability of NoSQL systems, but without giving up the transactional guarantees of a traditional DBMS. The H-Store system is able to scale out horizontally across multiple machines to improve throughput, as opposed to moving to a more powerful , more expensive machine for a single-node system.

5.9 Object-relational mapping

192. Hibernate
193. OpenJPA
194. EclipseLink
195. DataNucleus
196. ODBC/JDBC

5.10 Extraction Tools

197. UIMA
381. Tika

    "The Apache Tika toolkit detects and extracts metadata and text from over a thousand different file types (such as PPT, XLS, and PDF). All of these file types can be parsed through a single interface, making Tika useful for search engine indexing, content analysis, translation, and much more. *[Tik]*"

5.11 SQL(NewSQL)

198. Oracle
199. DB2
200. SQL Server

    SQL Server *[sql]* is a relational database management system from Microsoft. As of Jan 2017, SQL Server is available in below editions

    (a) Standard - consists of core database engine

    (b) Web - low cost edition for web hosting

    (c) Business Intelligence - includes standard edition and business intelligence tools like PowerPivot, PowerBI, Master Data Services

    (d) Enterprise - consists of core database engine and enterprise services like cluster manager

    (e) SQL Server Azure - *[azu]* core database engine integrated with Microsoft Azure cloud platform and available in platform-as-a-service mode.

    It is explained that technical architecture of SQL Server in OLTP(online transaction processing), hybrid cloud and business intelligence modes *[RM14]*.

201. SQLite
202. MySQL
203. PostgreSQL
204. CUBRID

    CUBRID name is deduced from the combination of word CUBE(security within box) and BRIDGE(data bridge). It is an open source Relational DataBase Management System designed in C programming language with high performance, scalability and availability features. During its development by NCL, korean IT service provider the goal was to optimize database performance for web-applications. *[www17b]* Importantly most of the SQL syntax from MYSQL and ORACLE can work on cubrid.CUBRID also provides manager tool for database administration and migration tool for migrating the data from DBMS to CUBRID bridging the dbs. CUBRID enterprise version and all the tools are free and suitable database candidate for web-application development.

205. Galera Cluster

     Galera cluster *[www17c]* is a type of database clustering which has all multiple masters and works on synchronous replication. At a deeper level, it was created by extending MySql replication API to provide all support for true multi master synchronous replication. This extended api is called as Write-Set Replication API and is the core of the clustering logic. Each transaction of wsrep API not only contains the record but also other meta-info to requires to commit each node separately or asynchronously. So though it seems synchronous logically but works independently on each node. The approach is also called virtually synchronous replication. This helps in directly read-write on a specific node and can lose a node without handling any complex failover scenarios (zero downtime).

206. SciDB
207. Rasdaman
208. Apache Derby
209. Pivotal Greenplum
210. Google Cloud SQL
211. Azure SQL
212. Amazon RDS
213. Google F1
214. IBM dashDB
215. N1QL
216. BlinkDB
217. Spark SQL

5.12 NoSQL

218. Lucene

     Apache Lucene *[www17a]* is a high-performance, full-featured text search engine library. It is originally written in pure Java but also has been ported to few other languages chiefly python. It is suitable for applications that requires full-text search. One of the key implementation of Lucene is Internet search engines and local, single-site searching. Another important implementation usage is its recomendation system. The core idea of Lucene is to extract text from any document that contains text (not image) field, making it format idependent.

219. Solr
220. Solandra
221. Voldemort

     According to *[wwwb]*, project Voldemort, developed by LinkedIN, is a non-relational database of key-value type that supports eventual consistency. The distributed nature of the system allows pluggable data placement and provides horizontal scalability and high consistency. Replication and partitioning of data is automatic and performed on multiple servers. Independent nodes that comprise the server support transparent handling of server failure and ensure absence of a central point of failure. Essentially, Voldemort is a hashtable. It uses APIs for data replication. In memory caching allows for faster operations. It allows cluster expansion with no data rebalancing. When Voldemort performance was benchmarked with the other key-value databases such as Cassandra, Redis and HBase as well as MySQL relational database (*[RSJ12]*), the Voldemart's throughput was twice lower than MySQL and Cassandra and six times higher than HBase. Voldemort was slightly underperforming in comparison with Redis. At the same time, it demonstrated consistent linear performance in maximum throughput that supports high scalability. The read latency for Voldemort was fairly consistent and only slightly underperformed Redis. Similar tendency was observed with the read latency that puts Voldermort in the cluster of databases that require good read-write speed for workload operations. However, the same authors noted that Voldemort required creation of the node specific configuration and optimization in order to successfully run a high throughput tests. The default options were not sufficient and were quickly saturated that stall the database.

222. Riak
223. ZHT
224. Berkeley DB
225. Kyoto/Tokyo Cabinet
226. Tycoon
227. Tyrant

     Tyrant provides network interfaces to the database management system called Tokyo Cabinet. Tyrant is also called as Tokyo Tyrant. Tyrant is implemented in C and it provides APIs for Perl, Ruby and C. Tyrant provides high performance and concurrent access to Tokyo Cabinet. In his blog *[Tyra]* Matt Yonkovit has explained the results of performance experiments he conducted to compare Tyrant against Memcached and MySQL.

     Tyrant was written and maintained by FAL Labs *[Tyrb]*. However, according to FAL Labs, their latest product *[Tyc]* Kyoto Tycoon is more powerful and convenient server than Tokyo Tyrant.

228. MongoDB
229. Espresso
230. CouchDB
231. Couchbase
232. IBM Cloudant
233. Pivotal Gemfire
234. HBase
235. Google Bigtable
236. LevelDB
237. Megastore and Spanner
238. Accumulo
239. Cassandra

     Apache Cassandra *[www16a]* is an open-source distributed database managemment for handling large volume of data accross comodity servers. It works on asynchronous masterless replication technique leading to low latency and high availability. It is a hybrid between a key-value and column oriented database. A table in cassandra can be viewed as a multi dimensional map indexed by a key. It has its own "Cassandra Query language (CQL)" query language for data extraction and mining. One of the demerits of such structure is it does not support joins or subqueries. It is a java based system which can be administered by any JMX compliant tools.

240. RYA
241. Sqrrl
242. Neo4J

243. graphdb
244. Yarcdata
245. AllegroGraph
246. Blazegraph
247. Facebook Tao
248. Titan:db
249. Jena
250. Sesame
251. Public Cloud: Azure Table
252. Amazon Dynamo
253. Google DataStore

5.13 File management

254. iRODS
255. NetCDF
256. CDF
257. HDF
258. OPeNDAP
259. FITS

     FITS stand for 'Flexible Image Trasnport System'. It is a standard data format used in astronomy. FITS data format is endorsed by NASA and International Astronomical Union. According to *[FITa]*, FITS can be used for transport, analysis and archival storage of scientific datasets and support multi-dimensional arrays, tables and headers sections. FITS is actively used and developed - according to *[FITb]* newer version of FITS standard document was released in July 2016. FITS can be used for digitization of contents like books and magzines. *[FITc]* used FITS for long term preservation of their book, manuscripts and other collection. Matlab, a language used for technical computing supports fits *[FITd]*. In his 2011 paper, Keith Wiley *[pap11]* explained how they performed processing of astronomical images on Hadoop. They used FITS format for data storage.

260. RCFile
261. ORC
262. Parquet

5.14 Data Transport

263. BitTorrent
264. HTTP
265. FTP
266. SSH

267. Globus Online (GridFTP)

    GridFTP is a enhancement on the File Tranfer Protocol (FTP) which provides high-performance , secure and reliable data transfer for high-bandwidth wide-area networks. As noted in *[Gri]* the most widely used implementation of GridFTP is Globus Online. GridFTP achieves efficient use of bandwidth by using multiple simultaneous TCP streams. Files can be downloaded in pieces simultaneously from multiple sources; or even in separate parallel streams from the same source. GridFTP allows transfers to be restarted automatically and handles network unavailability with a fault tolerant implementation of FTP.The underlying TCP connection in FTP has numerous settings such as window size and buffer size. GridFTP allows automatic (or manual) negotiation of these settings to provide optimal transfer speeds and reliability .

268. Flume
269. Sqoop
270. Pivotal GPLOAD/GPFDIST

5.15 Cluster Resource Management

271. Mesos
272. Yarn
273. Helix
274. Llama
275. Google Omega
276. Facebook Corona
277. Celery
278. HTCondor
279. SGE
280. OpenPBS
281. Moab
282. Slurm *[Slu]*
283. Torque
284. Globus Tools
285. Pilot Jobs

5.16 File systems

286. HDFS
287. Swift
288. Haystack
289. f4
290. Cinder
291. Ceph

292. FUSE
293. Gluster
294. Lustre
295. GPFS
296. GFFS
297. Public Cloud: Amazon S3
298. Azure Blob
299. Google Cloud Storage

5.17 Interoperability

300. Libvirt
301. Libcloud
302. JClouds

     [BDM15] Primary goals of cross-platform cloud APIs is that application built using these APIs can be seamlessly ported to different cloud providers. The APIs also bring interoperability such that cloud platforms can communicate and exchange information using these common or shared interfaces. Jclouds or apache jclouds *[wwwe]* is a java based library to provide seamless access to cloud platforms. Jclouds library provides interfaces for most of cloud providers like docker, openstack, amazon web services, microsoft azure, google cloud engine etc. It will allow users build applications which can be portable across different cloud environments. Key components of jcloud are:

     (a) Views: abstracts functionality from a specific vendor and allow user to write more generic code. For example odbc abstracts the underlying relational data source. However, odbc driver converts to native format. In this case user can switch databases without rewriting the application. Jcloud provide following views: blob store, compute service, loadBalancer service

     (b) API: APIs are requests to execute a particular functionality. Jcloud provide a single set of APIs for all cloud vendors which is also location aware. If a cloud vendor doesn't support customers from a particular region the API will not work from that region.

     (c) Provider: a particular cloud vendor is a provider. Jcloud uses provider information to initialize its context.

     (d) Context: it can be termed as a handle to a particular provider. Its like a ODBC connection object. Once connection is initialized for a particular database, it can used to make any api call.

         Jclouds provides test library to mock context, APIs etc to different providers so that user can write unit test for his implementation rather than waiting to test with the cloud provider. Jcloud library certifies support after testing the interfaces with live cloud provider. These features make jclouds robust and adoptable, hiding most of the complexity of cloud providers.

303. TOSCA
304. OCCI
305. CDMI
306. Whirr
307. Saga
308. Genesis

5.18 DevOps

309. Docker (Machine, Swarm)
310. Puppet
311. Chef

     Chef is a configuration management tool. It is implemented in Ruby and Erlang. Chef can be used to configure and maintain servers on-premise as well as cloud platforms like Amazon EC2, Google Cloud Platform and Open Stack. In this book *[Mar13]*, it is mentioned how implementation recipes in Chef to manage server applications and utilities such as database servers like MySQL, or HTTP servers like Apache HTPP and systems like Apache Hadoop.

     Chef is available in open source version and it also has commercial products for the companies which need it *[Che]*
312. Ansible
313. SaltStack
314. Boto
315. Cobbler
316. Xcat
317. Razor
318. CloudMesh
319. Juju

     > Juju (formerly Ensemble) *[BlRW14]* is software from Canonical that provides open source service orchestration. It is used to easily and quickly deploy and manage services on cloud and physical servers. Juju charms can be deployed on cloud services such as Amazon Web Services (AWS), Microsoft Azure and OpenStack. It can also be used on bare metal using MAAS. Specifically *[Canonical]* lists around 300 charms available for services available in the Juju store. Charms can be written in any language. It also supports Bundles which are pre-configured collection of Charms that helps in quick deployment of whole infrastructure.
320. Foreman
321. OpenStack Heat
322. Sahara

     The Sahara product provides users with the capability to provision data processing frameworks (such as Hadoop, Spark and Storm) on OpenStack *[Ope]* by specifying several parameters such as the version,cluster topology and hardware node details.As specified in *[Sah]* the solution allows for fast provisioning of data processing clusters on OpenStack for development and quality assurance and utilisation of unused computer power from a general purpose OpenStack Iaas Cloud.Sahara is managed via a REST API with a User Interface available as part of OpenStack Dashboard.
323. Rocks
324. Cisco Intelligent Automation for Cloud
325. Ubuntu MaaS
326. Facebook Tupperware
327. AWS OpsWorks
328. OpenStack Ironic

329. Google Kubernetes
330. Buildstep
331. Gitreceive
332. OpenTOSCA
333. Winery
334. CloudML
335. Blueprints
336. Terraform
337. DevOpSlang
338. Any2Api

5.19 IaaS Management from HPC to hypervisors

339. Xen
340. KVM
341. QEMU
342. Hyper-V
343. VirtualBox
344. OpenVZ
345. LXC
346. Linux-Vserver
347. OpenStack
348. OpenNebula
349. Eucalyptus
350. Nimbus

    Nimbus Infrastructure *[Nimb]* is an open source IaaS implementation. It allows deployment of self-configured virtual clusters and it supports configuration of scheduling, networking leases, and usage metering.

    Nimbus Platform *[Nima]* provides an integrated set of tools which enable users to launch large virtual clusters as well as launch and monitor the cloud apps. It also includes service that provides auto-scaling and high availability of resources deployed over multiple IaaS cloud. The Nimubs Platform tools are cloudinit.d, Phantom and Context Broker. In this paper *[nim13]* it is mentioned how to used Nimbus Phantom to deploy auto-scaling solution across multiple NSF FutureGrid clouds. In this implementation Phantom was responsible for deploying instances across multiple clouds and monitoring those instance. Nimbus platform supports Nimbus, Open Stack, Amazon and several other clouds.

351. CloudStack
352. CoreOS
353. rkt
354. VMware ESXi
355. vSphere and vCloud

356. Amazon
357. Azure
358. Google and other public Clouds
359. Networking: Google Cloud DNS
360. Amazon Route 53

5.20 Cross-Cutting Functions

5.20.1 Monitoring

361. Ambari
362. Ganglia
363. Nagios *[wwwg]*

    Nagios is a platform, which provides a set of software for network infrastructure monitoring. It also offers administrative tools to diagnose when failure events happen, and to notify operators when hardware issues are detected. Specifically, illustrates that Nagios is consist of modules including *[Jos13]*: a core and its dedicated tool for core configuration, extensible plugins and its frontend. Nagios core is designed with scalability in mind. Nagios contains a specification language allowing for building an extensible monitoring systems. Through the Nagios API components can integrate with the Nagios core services. Plugins can be developed via static languages like C or script languages. This mechanism empowers Nagios to monitor a large set of various scenarios yet being very flexible. *[IPMM12]* Besides its open source components, Nagios also has commercial products to serve needing clients.

364. Inca

    Inca is a grid monitoring *[LW09]* software suite. It provides grid monitoring features. These monitoring features provide operators failure trends, debugging support, email notifications, environmental issues etc. *[wwwd]*. It enables users to automate the tests which can be executed on a periodic basis. Tests can be added and configured as and when needed. It helps users with different portfolios like system administrators, grid operators, end users etc Inca provides user-level grid monitoring. For each user it stores results as well as allows users to deploy new tests as well as share the results with other users. The incat web ui allows users to view the status of test, manage test and results. The architectural blocks of inca include report repository, agent, data consumers and depot. Reporter is an executable program which is used to collect the data from grid source. Reporters can be written in perl and python. Inca repository is a collection of pre build reporters. These can be accessed using a web url. Inca repository has 150+ reporters available. Reporters are versioned and allow automatic updates. Inca agent does the configuration management. Agent can be managed using the incat web ui. Inca depot provides storage and archival of reports. Depot uses relational database for this purpose. The database is accessed using hibernate backend. Inca web UI or incat provides real time as well as historical view of inca data. All communication between inca components is secured using SSL certificates. It requires user credentials for any access to the system. Credentials are created at the time of the setup and installation. Inca's performance has been phenomenal in production deployments. Some of the deployments are running for more than a decade and has been very stable. Overall Inca provides a solid monitoring system which not only monitors but also detects problems very early on.

5.20.2 Security & Privacy

365. InCommon
366. Eduroam

367. OpenStack Keystone
368. LDAP
369. Sentry
370. Sqrrl
371. OpenID
372. SAML OAuth

5.20.3 Distributed Coordination

373. Google Chubby
374. Zookeeper
375. Giraffe
376. JGroups

5.20.4 Message and Data Protocols

377. Avro
378. Thrift
379. Protobuf

     Protocol Buffer *[www16b]* is a way to serialize structured data into binary form (stream of bytes) in order to transfer it over wires or for storage. It is used for inter apllication communication or for remote procedure call (RPC). It involves a interface description that describes the structure of some data and a program that can generate source code or parse it back to the binary form. It emphasizes on simplicity and performance over xml. Though xml is more readable but requires more resources in parsing and storing. This is developed by Google and available under open source licensing. The parser program is available in many languages including java and python.

5.21 New Technologies to be integrated

382. TBD

5.22 Excersise

TechList.1: In class you will be given an HID and you will be assigned a number of technologies that you need to research and create a summary as well as one or more relevant refernces to be added to the Web page. An example is given for Nagios. Please create a pull request with your responses. You are responsible for making sure the request shows up and each commit is using gitchangelog "new:usr: added paragraph about <PUTTECHHERE>" For the repository and create a single pull request with your response for all technologies you are responsible to invesitgate. Make sure to add your refernce to refs.bib. Many technologies may have additional refernces than the Web page. Please add the most important once while limiting it to three if you can. Avoid plagearism and use proper quotations or better rewrite the text.

You must look at techlist-tips to sucessfully complete the homework

A video about this hoemwork is posted at https://www.youtube.com/watch?v=roi7vezNmfo showing how to do references in emacs and jabref, it shows you how to configure git, it shows you how to do the forkrequest while asking you to add "new:usr ...." to the commit messages). As this is a homework realated video we put a lot of information in it that is not only useful for beginners. We recommend you watch it.

This homework can be done in steps. First you can collect all the content in an editor. Second you can create a fork. Third you can add the new content to the fork. Fourth you can commit. Fith you can push. SIx if the TAs have commend improve. The commit message must have new:usr: at the beginning.

While the Nagios entry is a good example (make sure grammer is ok the Google app engine is an example for a bad entry.

Do Techlist 1.a 1.b 1.c first. We will assign Techlist 1.d and TechList 2 in February.

TechList.1.a: Complete the pull request with the technologies assigned to you. Details for the assignment are posted in Piazza. Search for TechList.

TechList.1.b: Identify how to cite. We are using "scientific" citation formats such as IEEEtran, and ACM. We are **not** using citation formats such as Chicago, MLA, or ALP. The later are all for non scientific publications and thus of no use to us. Also when writing about a technology do not use the names of the person, simply say something like. In [1] the definition of a turing machine is given as follows, ... and do not use elaborate sentences such as: In his groundbraking work conducted in England, Allan Turing, introduced the turing machine in the years 1936-37 [2]. Its definition is base on ... The difference is clear, while the first focusses on results and technological concepts, the second introduces a colorful description that is more suitable for a magazine or a computer history paper.

TechList 1.c: Learn about Plagearism and how to avoid it. Many Web pages will conduct self advertisement while adding suspicious and subjective adjectives or phrases such as cheaper, superior, best, most important, with no equal, and others that you may not want to copy into your descriptions. Please focus on facts not on what the author of the Web page claims.

TechList 1.d: Identify technologies from the Apache project that ar enot yet listed here and add the name and descriptions as well as references.

TechList.2: As some students may not complete this assignment because they for example dropped the class, identify a number of not submitted descriptions and complete them. Coordinate with your class mates to identify a non overlapping assignment. The TA's will assign you additional technologies.

TechList.3: Identify technologies that are not listed here and add them. Provide a description and a refrence just as you did before. Make sure duplicated entries will be merged. Before you start do a pull to avoid adding technologies that have already been done by others.

5.23 Refernces

CHAPTER

SIX

FAQ

6.1 What are the prerequisites for this class?

We have communicated the prerequisites to the university, but they may have forgotten to add them to the class. The perquisites can be found in the appropriate class overview. Please review them carefully.

See:

- I524 *Prerequisites*

6.2 Why is this class not hosted on EdX?

I523 and I524 were the first classes hosted on EdX. We wanted to continue offering them on EdX. However, the university system administrators mentioned to us that their systems are not ready to support this class in EdX. Unfortunately, this is out of our control and we hope that the university will soon update to an EdX system that can host our classes.

6.3 Why are you not using CANVAS for communicating with students?

We have found that Piazza supports our needs for communication better than Canvas.

6.4 Why are you using github for submitting projects and papers?

This class is about open source technology and we like that you benefit from material others in the class are developing or have developed. All assignments are openly submitted to the class github for everyone to see. As part of the goal of this class is to develop reusable deployments. Such reuse is only possible if the code is publicly available and others can benefit from it.

The technology papers are made accessible so you can read other technology papers and can get an introduction in technologies that you may not yet know about. It also allows others to contribute to your technology papers and improve them.

6.5 I am full time student at IUPUI. Can I take the online version?

Yes you can.

If you are an international student, I suggest you verify this with the office and the registrar. There may be some restrictions for international students. Also some degree programs may have a limit or do not allow to take online classes. It will be up to you to verify the requirements with the appropriate administrators.

6.6 I am a residential student at IU. Can I take the online version only?

We recommend you take the residential class.

If you are an international student or a student of a particular degree program restrictions may be placed in if and how many online courses you can take. It will be up to you to contact the appropriate administrative departments including the international student office to verify what is allowed for you. In general international students have such restrictions. Please find out what they are and which section of the course is appropriate for you.

6.7 The class is full what do I do?

1. Make sure to put yourself on the waiting list.
2. If you are a residential student show up on the first class in the specified lecture room. More likely than not some students will enroll in more classes than they can do and places will free up. We will create a list and discuss with the registrar what to do.

6.8 Do I need to buy a textbook?

No, the resources will be provided for every unit. However, we recommend that you identify useful books for the class that can help you. Examples include

1. “Taming The Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics”, Bill Franks Wiley ISBN: 978-1-118-20878-6
2. “Doing Data Science: Straight Talk from the Frontline”, Cathy O’Neil, Rachel Schutt, O’Reilly Media, ISBN 978-1449358655

If you find good books, we like to add them here.

6.9 Why is there no textbook?

We cover a wide range of topics and their subject-matter is constantly undergoing changes. A textbook would be out of date by the time of publishing.

6.10 Do I need a computer to participate in this class?

If you are an online student you need access to a computer. If you are a residential student the facilities provided by SOIC will be sufficient. However, as you study involves computers, its probably important to evaluate if a computer will make your work easier.

If it comes to what computer to buy we really do not have a good recommendation as this depends on your budget. A computer running Linux or OSX makes programming probably easier. A windows computer has the advantage of also being able to run Word and ppt (so does OSX). A cheap machine with multiple cores and sufficient memory (16GB+) is a good idea. A SSD will make access to data especially if large data snappy.

For this reason I myself use a Mac, but you probably can get much cheaper machines with similar specs elsewhere.

Other students bought themselves a cheap computer and installed Linux on it so they do not interfere with their work machines or with Windows. Given how inexpensive computers these days are this may be a reasonable idea. However, do not go too cheap have enough memory and use an SSD if you can.

6.11 Representative Bibliography

1. Big data: The next frontier for innovation, competition, and productivity
2. Big Data Spring 2015 Class

6.12 Where is the official IU calendar for the Fall?

Please follow this link

6.13 How to write a research article on computer science?

1. A good lecture about this is presented by Simon Peyton Jones, Microsoft Research https://www.youtube.com/watch?v=g3dkRsTqdDA

Other resources may inspire you also:

1. https://globaljournals.org/guidelines-tips/research-paper-publishing
2. http://www.cs.columbia.edu/~hgs/etc/writing-style.html
3. https://www.quora.com/How-do-I-write-a-research-paper-for-a-computer-science-journal

6.14 Which bibliography manager is required for the class?

We require you use jabref:

1. http://www.jabref.org/

6.15 Can I use endnote or other bibliography managers?

No. Jabref is best for us and we do require that you hand in all bibliographies while cleaning and transferring them to jabref. We will not accept any other bibliography tool such as:

1. http://endnote.com/
2. http://libguides.utoledo.edu/c.php?g=284330&p=1895338
3. https://www.mendeley.com/
4. https://community.mendeley.com/guides/using-citation-editor/05-creating-bibliography
5. https://www.zotero.org

6.16 Plagiarism test and resources related to that

1. https://www.grammarly.com/plagiarism-checker
2. http://turnitin.com/
3. http://www.plagscan.com/plagiarism-check/

6.17 How many hours will this course take to work on every week?

This question can not rely be answered precisely. Typically we have 2-3 hours video per week. However starting from that its difficult to put a real number down as things may also depend on your background.

- The programming load is modest, but requires knowledge in python and linux systems which you may have to learn outside of this class.
- Some students have more experience than others, thus it may be possible to put in 6 hours per week overall, but other may have to put in 12 hours, while yet others may enjoy this class so much that they spend a lot more hours.
- We will certainly not stopping you form spending time in the class. It will be up to you to figure out how much time you will spend.
- Please remember that procrastination will not pay of in this class.
- The project or term paper will take a significant amount of time.

6.18 Is all classes material final?

No. Class material can change. Please remember that in a normal class you will be given several hours of lectures a week. They will be released on a weekly basis. What we do here is to release the material as much as possible upfront and **correct** them when we find it necessary to provide improvements or additions. Additionally, we integrate your feedback into the classes. If you find errors on the class Web page or have additions that you want to add, we would like to hear from you. Pull requests can be issued by you so your contributions get acknowledged and rewarded as part of the grade.

6.19 What are the changes to the web page?

The changes we make are typically fixing errata or clarification of content. We do attempt to indicate when major change is made.

6.20 What lectures should I learn when?

The class is structured in lectures that you can listen to at any time. If you have difficulties with organizing your own calendar, we will develop a sample calendar for you. Please contact us. However we have undergraduates, graduates, residential and online students. We even have students that can only work part of the semester while they use their vacation. Hence, it is impossible for us to provide an exact calendar that satisfies all the different types of students. Hence we appeal to your organizational skills to create a “study” plan for you during the first week of the semester that works for you.

We recommend to do the theory lectures as quickly as possible, but also start learning ansible at the same time as this will be part of your project. You will fail if you assume you can do the project in 2 weeks. You will need to work on it all semester long on weekly basis, starting with learning how to use ansible and cloud resources.

6.21 I524: Why are you doing the papers?

Part of doing research is to communicate your thoughts on topics and to be able to analyze and evaluate technologies that may or may not be useful for you. Our goal within this class is for the first time to gather a significant portion of the technologies that you hear about in class and that you get exposed to as part of the technology list into a “proceedings” developed by all students in class. The papers serve also the dual purpose of you learning how to write a paper and use bibliographies.

6.22 I524: Why are there no homework to test me on skills such as ansible or python?

We used to do smaller homework in previous classes to evaluate you on your skills. However we found that they did not reflect real-world use cases. By focusing on the project instead, you will be forced to develop these skills.

However, we can provide you with additional ungraded homework that you can conduct to test your skills if you like. Please let us know if you like to do that and we can assign such homework to you.

6.23 I524: Why not use chef or another DevOps framework?

We used to use chef and other DevOps frameworks. However we found that for a class grading can not be uniformly conducted while using too many frameworks. We also found that the value of learning on how to collaboratively contribute as part of an opensource class was diminished while a small group were choosing other technologies. These groups complained later on that they had too much work and could not benefit from other students. Hence we make is simple. All DevOps must be provided in ansible. All programming must be provided in python if not an explicit reason exist to use another language or technology such as R or technologies such as neo4j. However all deployment must be done in python and ansible.

6.24 I am lost?

Please contact the instructors for your class.

6.25 I do not like Technology/Topic/Project/etc?

Please contact the instructors for your class.

6.26 I am not able to attend the online hours

Typically we provide many different times for meetings via Zoom. We even schedule within reason special sessions. All of them are however during reasonable hours in United States Easter Standard Time.

6.27 Do I need to attend the online sessions?

No. But you can ask any question you want. We found that in previous classes that some students clearly benefitted from online sessions. If you attend them make sure you have a working and tested microphone if possible.

6.28 What are the leaning outcomes?

If you feel that they are not clearly stated as part of the course please contact us so that we can clarify the material.

6.29 There are so many messages on Piazza I can not keep up.

Residential students typically participate in live lectures in which we discuss with each other important aspects of a topic. As an online class may not have such a lecture, the piazza posts are just a replacement of them. It is required that you read the posts and decide which of them are relevant for you. In a lecture room you will find also that one student asks a question, while the professor answers the question to the entire class.

6.30 I find the hosting Web confusing

Once in a while we find that a student finds the hosting of the class material on the class Web page confusing. This confusion can be overcome by doing the following:

1. You may have to take time to explore the Web page and identify what needs to be done for the class. However each class has a clear overview page.
2. You may have to learn to get used to a class that allows you to work ahead.
3. You may have to learn to appreciate the additional material that assist in learning about python, ansible, LaTex, or the many other topics
4. Please do not blame the instructors for things that are out of their control: You may not be aware that it is not the instructors fault that the university is not able to provide us with an EdX server that works for us. Our choice would be to use EdX.

6.31 I524: I do not know python. What do I do?

This class requires python. Please learn it. We will be using ansible for the project. This you can acquire as part of the class through self study.

CHAPTER

SEVEN

INDEX

genindex

BIBLIOGRAPHY

[Mah] Apache mahout. Web Page. URL: http://mahout.apache.org/.

[Tik] Apache tika. Web Page. URL: https://tika.apache.org/.

[Che] Chef commercial support. Web Page. URL: https://www.chef.io/support/.

[Dat] Datafu. Web Page. Accessed:1/16/2017. URL: https://datafu.incubator.apache.org/.

[FITa] Fits nasa. web. URL: https://fits.gsfc.nasa.gov/.

[FITb] Fits news. Web Page. URL: https://fits.gsfc.nasa.gov/fits_standard.html.

[FITc] Fits vatican library. Web Page. URL: https://www.vatlib.it/home.php?pag=digitalizzazione&ling=eng.

[FITd] Fits matlab. Web Page. URL: https://www.mathworks.com/help/matlab/import_export/importing-flexible-image-transport-system-fits-files.html?requestedDomain=www.mathworks.com.

[Gri] Globus online (gridftp). Web Page. Accessed:1/16/2017. URL: https://en.wikipedia.org/wiki/GridFTP.

[App] Google app engine. Web Page. URL: https://cloud.google.com/appengine/.

[HSta] H-store. Web Page. Accessed:1/16/2017. URL: http://hstore.cs.brown.edu/.

[HStb] H-storewiki. Web Page. Accessed:1/16/2017. URL: https://en.wikipedia.org/wiki/H-Store.

[Kyo] Kyoto cabinet. Web Page. Accessed:1/16/2017. URL: http://fallabs.com/kyotocabinet/.

[Tyc] Kyoto tycoon. Web Page. URL: http://fallabs.com/kyototycoon/.

[Nima] Nimbus. Web Page. URL: http://www.nimbusproject.org/doc/nimbus/platform/.

[Nimb] Nimbus wiki. Web Page. URL: https://en.wikipedia.org/wiki/Nimbus_(cloud_computing).

[Ope] Openstack. Web Page. Accessed:1/16/2017. URL: https://www.openstack.org/.

[wwwa] Presto. Web Page. Accessed: 2017-1-24. URL: https://prestodb.io/.

[Slu] Slurm. Web Page. URL: https://slurm.schedmd.com/.

[azu] Sql server azure. Web Page. URL: https://azure.microsoft.com/en-us/services/sql-database/?b=16.50.

[sql] Sql server wiki. Web Page. URL: https://en.wikipedia.org/wiki/Microsoft_SQL_Server.

[Sah] Sahara. Web Page. Accessed:1/16/2017. URL: http://docs.openstack.org/developer/sahara/.

[Tav] Taverna. Web Page. URL: https://taverna.incubator.apache.org/introduction/.

[Tyra] Tyrant blog. Web Page. URL: https://www.percona.com/blog/2009/10/19/mysql_memcached_tyrant_part3/.

[Tyrb] Tyrant fallabs. Web Page. URL: http://fallabs.com/tokyotyrant/.

[wwwb] Voldemort. Web Page. Accessed: 2017-1-17. URL: http://www.project-voldemort.com/voldemort/.

[wwwc] Gora, components. Web Page. Accessed: 2017-01-18. URL: http://gora.apache.org/.

[wwwd] Inca, components. Web Page. Accessed: 2017-01-16. URL: http://inca.sdsc.edu/.

[wwwe] JClouds, components. Web Page. Accessed: 2017-01-20. URL: https://jclouds.apache.org/.

[wwwf] Kinesis, components. Web Page. Accessed: 2017-01-17. URL: http://docs.aws.amazon.com/streams/latest/dev/key-concepts.html/.

[wwwg] Nagios components. Web Page. Accessed: 2017-1-11. URL: https://www.nagios.org/projects/.

[wwwh] RabbitMQ, components. Web Page. Accessed: 2017-01-19. URL: https://www.rabbitmq.com/.

[nim13] *Rebalancing in a multi-cloud environment in Science Cloud '13*, ACM, 2013. URL: http://dl.acm.org/citation.cfm?id=2465854.

[www16a] Apache cassandra. Web page, 2016. URL: http://cassandra.apache.org/.

[www16b] Protocol buffer. Web page, September 2016. URL: https://developers.google.com/protocol-buffers/.

[www17a] Apache lucene. Web page, January 2017. URL: http://lucene.apache.org/.

[www17b] Cubrid. Web Page, 2017. URL: http://www.cubrid.org/.

[www17c] Galera cluster. Web page, 2017. URL: http://galeracluster.com/.

[Anona] Anon. web page. accessed 2017-01-25. URL: http://phoenix.apache.org/.

[Anonb] Anon. Apache Phoenix. web page. accessed 2017-01-25. URL: https://en.m.wikipedia.org/wiki/Apache_Phoenix.

[Avr13] Abel Avram. Phoenix: Running SQL Queries on Apache HBase [Updated]. web page, January 2013. accessed 2017-01-25. URL: https://www.infoq.com/news/2013/01/Phoenix-HBase-SQL.

[BlRW14] Kent Baxley, JD la Rosa, and Mark Wenning. Deploying workloads with juju and maas in ubuntu 14.04 lts. In *Deploying workloads with Juju and MAAS in Ubuntu 14.04 LTS*. Dell Inc, Technical White Paper, may 2014.

[BDM15] Antonio Esposito Beniamino Di Martino, Giuseppina Cretella. *Cloud Portability and Interoperability*. Springer International Publishing, New York City, USA, illustrated edition, 2015. ISBN 331913700X, 9783319137001.

[CQB+14] Yueguo Chen, Xiongpai Qin, Haoqiong Bian, Jun Chen, Zhaoan Dong, Xiaoyong Du, Yanjie Gao, Dehai Liu, Jiaheng Lu, and Huijie Zhang. *A Study of SQL-on-Hadoop Systems*., pages 154–166. Springer International Publishing, 2014.

[Fou] Apache Software Foundation. Kafka-a distributed streaming platform. Web Page. URL: https://kafka.apache.org/.

[IPMM12] C. Issariyapat, P. Pongpaibool, S. Mongkolluksame, and K. Meesublak. Using nagios as a groundwork for developing a better network monitoring system. In *2012 Proceedings of PICMET '12: Technology Management for Emerging Technologies*, 2771–2777. July 2012.

[Jos13] David Josephsen. *Nagios: Building Enterprise-Grade Monitoring Infrastructures for Systems and Networks*. Prentice Hall Press, Upper Saddle River, NJ, USA, 2nd edition, 2013. ISBN 013313573X, 9780133135732.

[pap11] *Astronomical Image Processing with Hadoop*, volume 442, Astronomical Data Analysis Software and Systems XX. ASP Conference Proceedings, July 2011. URL: http://adsabs.harvard.edu/abs/2011ASPC..442...93W.

[KKN+08] Robert Kallman, Hideaki Kimura, Jonathan Natkins, Andrew Pavlo, Alexander Rasin, Stanley Zdonik, Evan P. C. Jones, Samuel Madden, Michael Stonebraker, Yang Zhang, John Hugg, and Daniel J. Abadi. H-Store: a high-performance, distributed main memory transaction processing system. *Proc. VLDB Endow.*, 1(2):1496–1499, 2008. URL: http://hstore.cs.brown.edu/papers/hstore-demo.pdf, doi:http://doi.acm.org/10.1145/1454159.1454211.

[Kes13] Justin Kestelyn. Phoenix in 15 Minutes or Less. web page, March 2013. accessed 2017-01-25. URL: http://blog.cloudera.com/blog/2013/03/phoenix-in-15-minutes-or-less/.

[LW09] Jinjun Chen Lizhe Wang, Wei Jie. *Grid Computing: Infrastructure, Service, and Applications*. Taylor & Francis, 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL 33487, 2009. ISBN 1420067664.

[Mar13] Matthias Marschall. *Chef Infrastructure Automation Cookbook*. Packt Publishing, 2013.

[OHara07] John O'Hara. Toward a commodity enterprise middleware. *Queue*, 5(4):48–55, 2007.

[OCSP12] G. Ostrouchov, W.-C. Chen, D. Schmidt, and P. Patel. Web page, 2012. URL: http://r-pbd.org/.

[RSJ12] Tilmann Rabl, Mohammad Sadoghi, and Hans-Arno Jacobsen. Solving big data challenges for enterprise application performance management. Web Page, Apr 2012. Accessed: 2017-1-17. URL: http://vldb.org/pvldb/vol5/p1724_tilmannrabl_vldb2012.pdf.

[RM14] Stacia Misner Ross Mistry. *Introducing Microsoft SQL Server 2014 Technical Overview*. number ISBN: 978-0-7356-8475-1. Microsoft Press, 2014.

[SS16] Sumit Gupta Shilpi Saxena. *Real-Time Big Data Analytics*. Packt Publishing, 35 Livery Street, Birmingham B3 2PB, UK, 1st edition, 2016. ISBN 9781784391409.

[tav07] *Taverna Workflows: Syntax and Semantics*, IEEE, December 2007.

[Canonical] Canonical. Juju site. Web Page. URL: https://www.ubuntu.com/cloud/juju.

[MicrosoftCorp] Microsoft Corp. web page. accessed 2017-01-21. URL: https://azure.microsoft.com/en-us/.

[MicrosoftCorp16] Microsoft Corp. web page, July 2016. accessed 2017-01-21. URL: https://www.sec.gov/Archives/edgar/data/789019/000119312516662209/d187868d10k.htm.