(chapter title)Applications and Implications for Developers:

Apache Hadoop has many ways for developers to create applications for users. It can be a knowledge base for AI applications such as a heuristic weather predictor or make use of financial forecasting using mathematical models. It can be a storehouse of data used for production facilities or a framework for retail and commercial software, the applications are endless. The major key to Hadoop’s success is the wide variety of ways to create applications for a wide variety of skilled developers to access, manipulate and maintain the system. We will look at the ways developers can implement applications as well as more abstractly designing software that uses the Hadoop framework. Next we will talk about the implications for developers and interested parties.

(subtitle)Access:

Apache Hadoop provides many ways to access the files stored on the system for developers to create applications. Firstly Apache Hadoop is built on the TCP/IP protocols which allows wide spread use of the system where parallel computations can be done close to where the data exists on the system. Users do not have to worry about performance of these computations as NameNode provides multithreading which avoids the bottleneck of saving to disk so that the application can run parallel and the developers only need to check that the transactions being done have been saved. The developers can continuously maintain the software without having to worry about users having access simultaneously as HDFS implements a single-writer, multiple-reader model such that the users will continue to access and the updates occur only when the users are inactive.(4) Developers can create many ways for software administrators to maintain their data on the system. They can create a web-platform so that the administrators could browse HDFS instances through an HTTP browser.(8)The developers can also create a way to access the HDFS instances easier for administrators by using an NFS gateway so that the HDFS can be mounted as part of the client’s local file system.(9) Hadoop also provides the FS shell command line interfaces for skilled administrators to access the HDFS directly through command line.(14) This also allows developers to create bath programs for their applications using scripting languages such as Python.

(subtitle)Application Development:

The Hadoop architecture makes it easy for developers to create software in various programming languages through detailed libraries and APIs. The HDFS provides a FileSystem Java API so that developers can use the Java programming languages to implement programs to access and manipulate the HDFS file system.(6,13) MapReduce maintains the jobs and tasks that run on the system and Hadoop gives the application Master REST API to allow developers to check the status of the running MapReduce application master, this includes the Jobs API and Tasks API for the Java programming language.(10) With Hadoop 2 providing YARN on top of the MapReduce developers can use the YARN web services REST APIs which are a set of URI resources that give access to the cluster nodes, applications and application historical information.(12) With these developers can create software to maintain all applications that are running and finished running on the system to optimize the software for their users.

For security developers can use the Hadoop Auth API for Java, HTTP and SPNEGO, a library consisting of client and server components to enable Kerberos SPNEGO authentication for HTTP browsers securing the web services created for administrators.(6, 15) Also the CredentialProvider API can be used to maintain passwords securely as it is a SPI framework for plugging in extensible credential providers, as it abstracts the use of sensitive tokens, secrets and passwords from the details of their storage and management.(5) The Hadoop system also gives developers the option of using other languages such as C with the language wrapper for the Java API and REST API, as well as SQL querying.(7)

With Hadoop using Java as the basis for their framework and APIs provided, developers streamline the software development lifecycle. As Java is an object-oriented language developers have all the advantages of abstraction of implementation that comes with such a language such as modularity, maintainability, the use of OODP (Object-Oriented Design Patterns), creating new frameworks and systems for the software that uses Apache Hadoop. The Layers of Hadoop also highlights this with the separate APIs provided for each of the core systems, HDFS, MapReduce, YARN and others in the Hadoop ecosystem allowing for the distribution of work amongst developers making developing and maintaining software easier.

(subtitle)Project Bylaws and Committership – Contributing to Apache Hadoop:

Apache Hadoop defines bylaws under which the project operates and has criteria for committers responsible for reviewing and integrating code changes.

There are a set of roles defined by Apache projects that have certain rights and responsibilities. There are the Users of the Hadoop system which provide feedback to developers via bug reports and suggestions, these users sometime get so involved that they themselves become contributing developers. The Contributors are volunteers that contribute their time, code and resources to Hadoop and continue to help until they may be invited to become committers to the project. Committers are responsible for the technical management of the project and have access to subproject subversion repositories so that they can maintain and build on top of the current development of the project. As noted above they are given access through invitation approved by active PMC (Project Management Committee) members and also may cast binding votes on technical discussion regarding subprojects. These committers can eventually become a member of the PMC themselves after contributing to the project after a long time. There are also committers that are Release Managers they volunteer to produce a Release Candidate according to certain restrictions and are responsible for building consensus around it to achieve a successful Product Release vote. Finally the PMC is responsible for the management and oversight of the Hadoop codebase, they decide what is distributed as products, maintain the project’s shared resources, speak on behalf of the project, resolve license disputes, nominate new members and committers and maintain the bylaws.(1,2)

The decision making concerning the project is done through a voting system. As noted above the PMC members and committers have a vote. The votes are placed as so, a +1 for a “Yes” and available to help, a +0 for agreement but will not be able to help, a -0 for someone who does not agree but is not concerned if the action goes ahead, and a -1 for a “No” and is counts as a veto. Approvals are done differently for different actions. A consensus approval requires 3 binding +1 votes, a lazy consensus requires no -1 votes, a lazy majority requires 3 +1 votes and more +1 votes than -1 votes, and a lazy two thirds majority which requires at least 3 votes and twice as many +1 votes and -1 votes. The vetoes that are cast must be followed with a valid reasonable explanation otherwise it is not considered. The various actions that can be voted on are code changes, a product release, adoption of a new codebase a new branch committer, new committers, new PMC members, branch committer removals, committer removals, PMC member removals, and modifying bylaws. Voting remains open for seven days for all actions with the exception of product releases which run for a period of five days.(1) The guidelines of which additions are made to the Apache Hadoop project allows for general developers using the system to contribute and become a part of the project to better their software as well as the system it runs off.

References:

1. <https://hadoop.apache.org/bylaws.html>
2. <https://hadoop.apache.org/committer_criteria.html>
3. <http://www.aosabook.org/en/hdfs.html>
4. <http://hadoop.apache.org/docs/current/>
5. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/CredentialProviderAPI.html>
6. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>
7. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/LibHdfs.html>
8. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/WebHDFS.html>
9. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsNfsGateway.html>
10. <http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapredAppMasterRest.html>
11. <http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-hs/HistoryServerRest.html>
12. <http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/WebServicesIntro.html>
13. <http://hadoop.apache.org/docs/current/api/index.html>
14. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/UnixShellAPI.html>
15. <http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/HttpAuthentication.html>