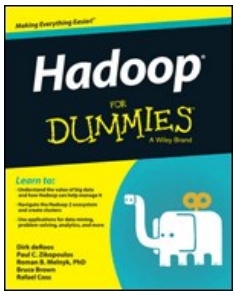


# Hadoop for Dummies

by Dirk deRoos et al.

# Chapter 1: Introducing Hadoop and Seeing What It's Good for

## In This Chapter

- Seeing how Hadoop fills a need
- Digging (a bit) into Hadoop's history
- Getting Hadoop for yourself
- Looking at Hadoop application offerings

Organizations are flooded with data. Not only that, but in an era of incredibly cheap storage where everyone and everything are interconnected, the nature of the data we're collecting is also changing. For many businesses, their critical data used to be limited to their transactional databases and data warehouses. In these kinds of systems, data was organized into orderly rows and columns, where every byte of information was well understood in terms of its nature and its business value. These databases and warehouses are still extremely important, but businesses are now differentiating themselves by how they're finding value in the large volumes of data that are *not* stored in a tidy database.

The variety of data that's available now to organizations is incredible: Internally, you have website clickstream data, typed notes from call center operators, e-mail and instant messaging repositories; externally, open data initiatives from public and private entities have made massive troves of raw data available for analysis. The challenge here is that traditional tools are poorly equipped to deal with the scale and complexity of much of this data. That's where Hadoop comes in. It's tailor-made to deal with all sorts of messiness. CIOs everywhere have taken notice, and Hadoop is rapidly becoming an established platform in any serious IT department.

This chapter is a newcomer's welcome to the wonderful world of Hadoop — its design, capabilities, and uses. If you're new to big data, you'll also find important background information that applies to Hadoop and other solutions.

## Big Data and the Need for Hadoop

Like many buzzwords, what people mean when they say "big data" is not always clear. This lack of clarity is made worse by IT people trying to attract attention to their own projects by labeling them as "big data," even though there's nothing big about them.

At its core, big data is simply a way of describing data problems that are unsolvable using traditional tools. To help understand the nature of big data problems, we like the "the three Vs of big data," which are a widely accepted characterization for the factors behind what makes a data challenge "big":

- **Volume**: High volumes of data ranging from dozens of terabytes, and even petabytes.
- **Variety**: Data that's organized in multiple structures, ranging from raw text (which, from a computer's perspective, has little or no discernible structure — many people call this *unstructured data*) to log files (commonly referred to as being *semistructured*) to data ordered in strongly typed rows and columns (*structured* data). To make things even more confusing, some data sets even include portions of all three kinds of data. (This is known as *multistructured* data.)
- **Velocity**: Data that enters your organization and has some kind of value for a limited window of time — a window that usually shuts well before the data has been transformed and loaded into a data warehouse for deeper analysis (for example, financial securities ticker data, which may reveal a buying opportunity, but only for a short while). The higher the volumes of data entering your organization per second, the bigger your velocity challenge.

Each of these criteria clearly poses its own, distinct challenge to someone wanting to analyze the information. As such, these three criteria are an easy way to assess big data problems and provide clarity to what has become a vague buzzword. The commonly held rule of thumb is that if your data storage and analysis work exhibits any of these three characteristics, chances are that you've got yourself a big data challenge.

**Failed attempts at coolness: Naming technologies**

The co-opting of the big data label reminds us when Java was first becoming popular in the early 1990s and every IT project had to have Java support or something to do with Java. At the same time, web site application development was becoming popular and Netscape named their scripting language "JavaScript," even though it had nothing to do with Java. To this day, people are confused by this shallow naming choice.

**Origin of the "3 Vs"**

In 2001, years before marketing people got ahold of the term "big data," the analyst firm META Group published a report titled *3-D Data Management: Controlling Data Volume, Velocity and Variety*. This paper was all about data warehousing challenges, and ways to use relational technologies to overcome them. So while the definitions of the 3Vs in this paper are quite different from the big data 3Vs, this paper does deserve a footnote in the history of big data, since it originated a catchy way to describe a problem.

As you'll see in this book, Hadoop is anything but a traditional information technology tool, and it is well suited to meet many big data challenges, especially (as you'll soon see) with high volumes of data and data with a variety of structures. But there are also big data challenges where Hadoop isn't well suited — in particular, analyzing high-velocity data the instant it enters an organization. Data velocity challenges involve the analysis of data while it's in motion, whereas Hadoop is tailored to analyze data when it's at rest. The lesson to draw from this is that although Hadoop is an important tool for big data analysis, it will by no means solve all your big data problems. Unlike some of the buzz and hype, the entire big data domain isn't synonymous with Hadoop.

### Exploding data volumes

It is by now obvious that we live in an advanced state of the information age. Data is being generated and captured electronically by networked sensors at tremendous volumes, in ever-increasing velocities and in mind-boggling varieties. Devices such as mobile telephones, cameras, automobiles, televisions, and machines in industry and health care all contribute to the exploding data volumes that we see today. This data can be browsed, stored, and shared, but its greatest value remains largely untapped. That value lies in its potential to provide insight that can solve vexing business problems, open new markets, reduce costs, and improve the overall health of our societies.

In the early 2000s (we like to say "the oughties"), companies such as Yahoo! and Google were looking for a new approach to analyzing the huge amounts of data that their search engines were collecting. Hadoop is the result of that effort, representing an efficient and cost-effective way of reducing huge analytical challenges to small, manageable tasks.

### Varying Data Structures

*Structured* data is characterized by a high degree of organization and is typically the kind of data you see in relational databases or spreadsheets. Because of its defined structure, it maps easily to one of the standard data types (or user-defined types that are based on those standard types). It can be searched using standard search algorithms and manipulated in well-defined ways.

*Semistructured* data (such as what you might see in log files) is a bit more difficult to understand than structured data. Normally, this kind of data is stored in the form of text files, where there is some degree of order — for example, tab-delimited files, where columns are separated by a tab character. So instead of being able to issue a database query for a certain column and knowing exactly what you're getting back, users typically need to explicitly assign data types to any data elements extracted from semistructured data sets.

*Unstructured* data has none of the advantages of having structure coded into a data set. (To be fair, the unstructured label is a bit strong — all data stored in a computer has some degree of structure. When it comes to so-called unstructured data, there's simply too little structure in order to make much sense of it.) Its analysis by way of more traditional approaches is difficult and costly at best, and logistically impossible at worst. Just imagine having many years' worth of notes typed by call center operators that describe customer observations. Without a robust set of text analytics tools, it would be extremely tedious to determine any interesting behavior patterns. Moreover, the sheer volume of data in many cases poses virtually insurmountable challenges to traditional data mining techniques, which, even when conditions are good, can handle only a fraction of the valuable data that's available.

### A Playground for Data Scientists

A *data scientist* is a computer scientist who loves data (lots of data) and the sublime challenge of figuring out ways to squeeze every drop of value out of that abundant data. A *data playground* is an enterprise store of many terabytes (or even petabytes) of data that data scientists can use to develop, test, and enhance their analytical "toys."

Now that you know what big data is all about, what it is, and why it's important, it's time to introduce Hadoop, the granddaddy of these nontraditional analytical toys. Understanding how this amazing platform for the analysis of big data came to be, and acquiring some basic principles about how it works, will help you to master the details we provide in the remainder of this book.

## The Origin and Design of Hadoop

So what exactly is this thing with the funny name — Hadoop? At its core, Hadoop is a framework for storing data on large clusters of *commodity* hardware — everyday computer hardware that is affordable and easily available — and running applications against that data. A *cluster* is a group of interconnected computers (known as *nodes*) that can work together on the same problem. Using networks of affordable compute resources to acquire business insight is the key value proposition of Hadoop.

As for that name, Hadoop, don't look for any major significance there; it's simply the name that Doug Cutting's son gave to his stuffed elephant. (Doug Cutting is, of course, the co-creator of Hadoop.) The name is unique and easy to remember — characteristics that made it a great choice.

Hadoop consists of two main components: a distributed processing framework named MapReduce (which is now supported by a component called YARN, which we describe a little later) and a distributed file system known as the Hadoop distributed file system, or HDFS.

An application that is running on Hadoop gets its work divided among the nodes (machines) in the cluster, and HDFS stores the data that will be processed. A Hadoop cluster can span thousands of machines, where HDFS stores data, and MapReduce jobs do their processing near the data, which keeps I/O costs low. MapReduce is extremely flexible, and enables the development of a wide variety of applications.

**TECHNICAL STUFF** As you might have surmised, a Hadoop cluster is a form of *compute cluster*, a type of cluster that's used mainly for computational purposes. In a compute cluster, many computers (*compute nodes*) can share computational workloads and take advantage of a very large aggregate bandwidth across the cluster. Hadoop clusters typically consist of a few *master nodes*, which control the storage and

processing systems in Hadoop, and many *slave nodes*, which store all the cluster's data and is also where the data gets processed.

## Distributed Processing with MapReduce

MapReduce involves the processing of a sequence of operations on distributed data sets. The data consists of key-value pairs, and the computations have only two phases: a map phase and a reduce phase. User-defined MapReduce jobs run on the compute nodes in the cluster.

### A look at the history books

Hadoop was originally intended to serve as the infrastructure for the Apache Nutch project, which started in 2002. Nutch, an open source web search engine, is a part of the Lucene project. What are these projects? Apache projects are created to develop open source software and are supported by the Apache Software Foundation (ASF), a nonprofit corporation made up of a decentralized community of developers. *Open source software*, which is usually developed in a public and collaborative way, is software whose source code is freely available to anyone for study, modification, and distribution.

Nutch needed an architecture that could scale to billions of web pages, and the needed architecture was inspired by the Google file system (GFS), and would ultimately become HDFS. In 2004, Google published a paper that introduced MapReduce, and by the middle of 2005 Nutch was using both MapReduce and HDFS.

In early 2006, MapReduce and HDFS became part of the Lucene subproject named Hadoop, and by February 2008, the Yahoo! search index was being generated by a Hadoop cluster. By the beginning of 2008, Hadoop was a top-level project at Apache and was being used by many companies. In April 2008, Hadoop broke a world record by sorting a terabyte of data in 209 seconds, running on a 910-node cluster. By May 2009, Yahoo! was able to use Hadoop to sort 1 terabyte in 62 seconds!

Generally speaking, a MapReduce job runs as follows:

1. During the Map phase, input data is split into a large number of fragments, each of which is assigned to a map task.
2. These map tasks are distributed across the cluster.
3. Each map task processes the key-value pairs from its assigned fragment and produces a set of intermediate key-value pairs.
4. The intermediate data set is sorted by key, and the sorted data is partitioned into a number of fragments that matches the number of reduce tasks.
5. During the Reduce phase, each reduce task processes the data fragment that was assigned to it and produces an output key-value pair.
6. These reduce tasks are also distributed across the cluster and write their output to HDFS when finished.

The Hadoop MapReduce framework in earlier (pre-version 2) Hadoop releases has a single master service called a JobTracker and several slave services called TaskTrackers, one per node in the cluster. When you submit a MapReduce job to the JobTracker, the job is placed into a queue and then runs according to the scheduling rules defined by an administrator. As you might expect, the JobTracker manages the assignment of map-and-reduce tasks to the TaskTrackers.

With Hadoop 2, a new resource management system is in place called YARN (short for *Y*et *A*nother *R*esource *M*anager). YARN provides generic scheduling and resource management services so that you can run more than just MapReduce applications on your Hadoop cluster. The JobTracker/TaskTracker architecture could only run MapReduce.

We describe YARN and the JobTracker/TaskTracker architectures in Chapter 7.

HDFS also has a master/slave architecture:

- **Master service**: Called a *NameNode*, it controls access to data files.
- **Slave services**: Called *DataNodes*, they're distributed one per node in the cluster. DataNodes manage the storage that's associated with the nodes on which they run, serving client read and write requests, among other tasks.

For more information on HDFS, see Chapter 4.

## Apache Hadoop Ecosystem

This section introduces other open source components that are typically seen in a Hadoop deployment. Hadoop is more than MapReduce and HDFS: It's also a family of related projects (an ecosystem, really) for distributed computing and large-scale data processing. Most (but not all) of these projects are hosted by the Apache Software Foundation. Table 1-1 lists some of these projects.

**Table 1-1: Related Hadoop Projects**

| *Project Name* | *Description* |
|---|---|
| Ambari | An integrated set of Hadoop administration tools for installing, monitoring, and maintaining a Hadoop cluster. Also included are tools to add or |

| | |
|---|---|
| | remove slave nodes. |
| Avro | A framework for the efficient *serialization* (a kind of transformation) of data into a compact binary format |
| Flume | A data flow service for the movement of large volumes of log data into Hadoop |
| HBase | A distributed columnar database that uses HDFS for its underlying storage. With HBase, you can store data in extremely large tables with variable column structures |
| HCatalog | A service for providing a relational view of data stored in Hadoop, including a standard approach for tabular data |
| Hive | A distributed data warehouse for data that is stored in HDFS; also provides a query language that's based on SQL (HiveQL) |
| Hue | A Hadoop administration interface with handy GUI tools for browsing files, issuing Hive and Pig queries, and developing Oozie workflows |
| Mahout | A library of machine learning statistical algorithms that were implemented in MapReduce and can run natively on Hadoop |
| Oozie | A workflow management tool that can handle the scheduling and chaining together of Hadoop applications |
| Pig | A platform for the analysis of very large data sets that runs on HDFS and with an infrastructure layer consisting of a compiler that produces sequences of MapReduce programs and a language layer consisting of the query language named Pig Latin |
| Sqoop | A tool for efficiently moving large amounts of data between relational databases and HDFS |
| ZooKeeper | A simple interface to the centralized coordination of services (such as naming, configuration, and synchronization) used by distributed applications |

The Hadoop ecosystem and its commercial distributions (see the "Comparing distributions" section, later in this chapter) continue to evolve, with new or improved technologies and tools emerging all the time.

Figure 1-1 shows the various Hadoop ecosystem projects and how they relate to one-another:

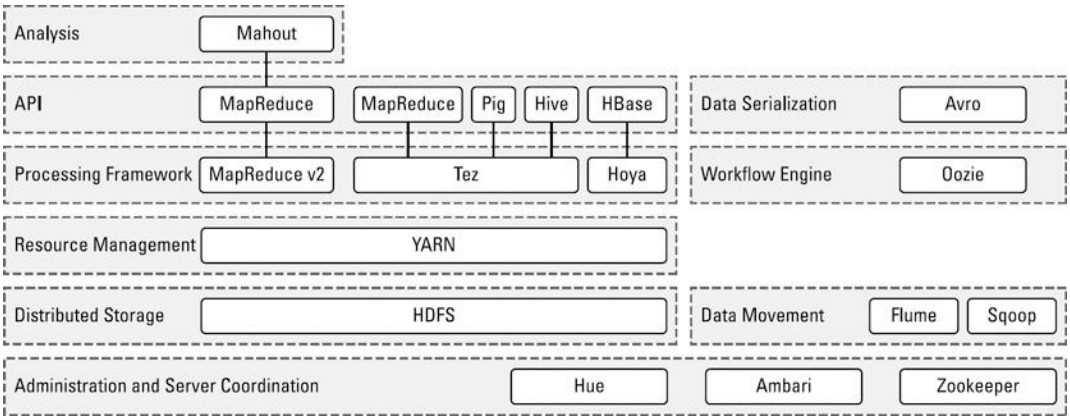


**Figure 1-1:** Hadoop ecosystem components

## Examining the Various Hadoop Offerings

Hadoop is available from either the Apache Software Foundation or from companies that offer their own Hadoop distributions.

**REMEMBER** Only products that are available directly from the Apache Software Foundation can be called Hadoop releases. Products from other companies can include the official Apache Hadoop release files, but products that are "forked" from (and represent modified or extended versions of) the Apache Hadoop source tree are not supported by the Apache Software Foundation.

Apache Hadoop has two important release series:

- **1.x**: At the time of writing, this release is the most stable version of Hadoop available (1.2.1).

  Even after the 2.x release branch became available, this is still commonly found in production systems. All major Hadoop distributions include solutions for providing high availability for the NameNode service, which first appears in the 2.x release branch of Hadoop.

- **2.x**: At the time of writing, this is the current version of Apache Hadoop (2.2.0), including these features:

  - *A MapReduce architecture, named MapReduce 2 or YARN (Yet Another Resource Negotiator)*: It divides the two major functions of the JobTracker (resource management and job life-cycle management) into separate components.
  - *HDFS availability and scalability*: The major limitation in Hadoop 1 was that the NameNode was a single point of failure. Hadoop 2 provides the ability for the NameNode service to fail over to an active standby NameNode. The NameNode is also enhanced to scale out to support clusters with very large numbers of files. In Hadoop 1, clusters could typically not expand beyond roughly 5000 nodes. By adding multiple active NameNode services, with each one responsible for managing specific partitions of data, you can scale out to a much greater degree.

**TECHNICAL STUFF** Some descriptions around the versioning of Hadoop are confusing because both Hadoop 1.x and 2.x are at times referenced using different version numbers: Hadoop 1.0 is occasionally known as Hadoop 0.20.205, while Hadoop 2.x is sometimes referred to as Hadoop 0.23. As of December 2011, the Apache Hadoop project was deemed to be production-ready by the open source community,

and the Hadoop 0.20.205 version number was officially changed to 1.0.0. Since then, legacy version numbering (below version 1.0) has persisted, partially because work on Hadoop 2.x was started well before the version numbering jump to 1.0 was made, and the Hadoop 0.23 branch was already created. Now that Hadoop 2.2.0 is production-ready, we're seeing the old numbering less and less, but it still surfaces every now and then.

## Comparing Distributions

You'll find that the Hadoop ecosystem has many component parts, all of which exist as their own Apache projects. (See the previous section for more about them.) Because Hadoop has grown considerably, and faces some significant further changes, different versions of these open source community components might not be fully compatible with other components. This poses considerable difficulties for people looking to get an independent start with Hadoop by downloading and compiling projects directly from Apache.

Red Hat is, for many people, the model of how to successfully make money in the open source software market. What Red Hat has done is to take Linux (an open source operating system), bundle all its required components, build a simple installer, and provide paid support to any customers. In the same way that Red Hat has provided a handy packaging for Linux, a number of companies have bundled Hadoop and some related technologies into their own Hadoop distributions. This list describes the more prominent ones:

- **Cloudera (www.cloudera.com/)**: Perhaps the best-known player in the field, Cloudera is able to claim Doug Cutting, Hadoop's co-founder, as its chief architect. Cloudera is seen by many people as the market leader in the Hadoop space because it released the first commercial Hadoop distribution and it is a highly active contributor of code to the Hadoop ecosystem.

  Cloudera Enterprise, a product positioned by Cloudera at the center of what it calls the "Enterprise Data Hub," includes the Cloudera Distribution for Hadoop (CDH), an open-source-based distribution of Hadoop and its related projects as well as its proprietary Cloudera Manager. Also included is a technical support subscription for the core components of CDH.

  Cloudera's primary business model has long been based on its ability to leverage its popular CDH distribution and provide paid services and support. In the fall of 2013, Cloudera formally announced that it is focusing on adding proprietary value-added components on top of open source Hadoop to act as a differentiator. Also, Cloudera has made it a common practice to accelerate the adoption of alpha- and beta-level open source code for the newer Hadoop releases. Its approach is to take components it deems to be mature and retrofit them into the existing production-ready open source libraries that are included in its distribution.

- **EMC (www.gopivotal.com)**: Pivotal HD, the Apache Hadoop distribution from EMC, natively integrates EMC's massively parallel processing (MPP) database technology (formerly known as Greenplum, and now known as HAWQ) with Apache Hadoop. The result is a high-performance Hadoop distribution with true SQL processing for Hadoop. SQL-based queries and other business intelligence tools can be used to analyze data that is stored in HDFS.

- **Hortonworks (www.hortonworks.com)**: Another major player in the Hadoop market, Hortonworks has the largest number of committers and code contributors for the Hadoop ecosystem components. (Committers are the gatekeepers of Apache projects and have the power to approve code changes.) Hortonworks is a spin-off from Yahoo!, which was the original corporate driver of the Hadoop project because it needed a large-scale platform to support its search engine business. Of all the Hadoop distribution vendors, Hortonworks is the most committed to the open source movement, based on the sheer volume of the development work it contributes to the community, and because all its development efforts are (eventually) folded into the open source codebase.

  The Hortonworks business model is based on its ability to leverage its popular HDP distribution and provide paid services and support. However, it does not sell proprietary software. Rather, the company enthusiastically supports the idea of working within the open source community to develop solutions that address enterprise feature requirements (for example, faster query processing with Hive).

  Hortonworks has forged a number of relationships with established companies in the data management industry: Teradata, Microsoft, Informatica, and SAS, for example. Though these companies don't have their own, in-house Hadoop offerings, they collaborate with Hortonworks to provide integrated Hadoop solutions with their own product sets.

  The Hortonworks Hadoop offering is the Hortonworks Data Platform (HDP), which includes Hadoop as well as related tooling and projects. Also unlike Cloudera, Hortonworks releases only HDP versions with production-level code from the open source community.

- **IBM (www.ibm.com/software/data/infosphere/biginsights)**: Big Blue offers a range of Hadoop offerings, with the focus around value added on top of the open source Hadoop stack:

  *InfoSphere BigInsights*: This software-based offering includes a number of Apache Hadoop ecosystem projects, along with additional software to provide additional capability. The focus of InfoSphere BigInsights is on making Hadoop more readily consumable for businesses. As such, the proprietary enhancements are focused on standards-based SQL support, data security and governance, spreadsheet-style analysis for business users, text analytics, workload management, and the application development life cycle.

  *PureData System for Hadoop*: This hardware- and software-based appliance is designed to reduce complexity, the time it takes to start analyzing data, as well as IT costs. It integrates InfoSphere BigInsights (Hadoop-based software), hardware, and storage into a single, easy-to-manage system.

- **Intel (hadoop.intel.com)**: The Intel Distribution for Apache Hadoop (Intel Distribution) provides distributed processing and data management for enterprise applications that analyze big data. Key features include excellent performance with optimizations for Intel Xeon processors, Intel SSD storage, and Intel 10GbE networking; data security via encryption and decryption in HDFS, and role-based access control with cell-level granularity in HBase (you can control who's allowed to see what data down to the cell level, in other words); improved Hive query performance; support for statistical analysis with a connector for R, the popular open source statistical package; and analytical

graphics through Intel Graph Builder.

It may come as a surprise to see Intel here among a list of software companies that have Hadoop distributions. The motivations for Intel are simple, though: Hadoop is a strategic platform, and it will require significant hardware investment, especially for larger deployments. Though much of the initial discussion around hardware reference architectures for Hadoop — the recommended patterns for deploying hardware for Hadoop clusters — have focused on commodity hardware, increasingly we are seeing use cases where more expensive hardware can provide significantly better value. It's with this situation in mind that Intel is keenly interested in Hadoop. It's in Intel's best interest to ensure that Hadoop is optimized for Intel hardware, on both the higher end and commodity lines.

The Intel Distribution comes with a management console designed to simplify the configuration, monitoring, tuning, and security of Hadoop deployments. This console includes automated configuration with Intel Active Tuner; simplified cluster management; comprehensive system monitoring and logging; and systematic health checking across clusters.

- **MapR (www.mapr.com)**: For a complete distribution for Apache Hadoop and related projects that's independent of the Apache Software Foundation, look no further than MapR. Boasting no Java dependencies or reliance on the Linux file system, MapR is being promoted as the only Hadoop distribution that provides full data protection, no single points of failure, and significant ease-of-use advantages. Three MapR editions are available: M3, M5, and M7. The M3 Edition is free and available for unlimited production use; MapR M5 is an intermediate-level subscription software offering; and MapR M7 is a complete distribution for Apache Hadoop and HBase that includes Pig, Hive, Sqoop, and much more.

  The MapR distribution for Hadoop is most well-known for its file system, which has a number of enhancements not included in HDFS, such as NFS access and POSIX compliance (long story short, this means you can mount the MapR file system like it's any other storage device in your Linux instance and interact with data stored in it with any standard file applications or commands), storage volumes for specialized management of data policies, and advanced data replication tools. MapR also ships a specialized version of HBase, which claims higher reliability, security, and performance than Apache HBase.

## Working with in-Database MapReduce

When MapReduce processing occurs on structured data in a relational database, the process is referred to as *in-database* MapReduce. One implementation of a hybrid technology that combines MapReduce and relational databases for the analysis of analytical workloads is HadoopDB, a research project that originated a few years ago at Yale University. HadoopDB was designed to be a free, highly scalable, open source, parallel database management system. Tests at Yale showed that HadoopDB could achieve the performance of parallel databases, but with the scalability, fault tolerance, and flexibility of Hadoop-based systems.

More recently, Oracle has developed an in-database Hadoop prototype that makes it possible to run Hadoop programs written in Java naturally from SQL. Users with an existing database infrastructure can avoid setting up a Hadoop cluster and can execute Hadoop jobs within their relational databases.

## Looking at the Hadoop Toolbox

A number of companies offer tools designed to help you get the most out of your Hadoop implementation. Here's a sampling:

- **Amazon (aws.amazon.com/ec2)**: The Amazon Elastic MapReduce (Amazon EMR) web service enables you to easily process vast amounts of data by provisioning as much capacity as you need. Amazon EMR uses a hosted Hadoop framework running on the web-scale infrastructure of Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3). Amazon EMR lets you analyze data without having to worry about setting up, managing, or tuning Hadoop clusters.

  **REMEMBER** Cloud-based deployments of Hadoop applications like those offered by Amazon EMR are somewhat different from on-premise deployments. You would follow these steps to deploy an application on Amazon EMR:

  1. Script a job flow in your language of choice, including a SQL-like language such as Hive or Pig.
  2. Upload your data and application to Amazon S3, which provides reliable storage for your data.
  3. Log in to the AWS Management Console to start an Amazon EMR job flow by specifying the number and type of Amazon EC2 instances that you want, as well as the location of the data on Amazon S3.
  4. Monitor the progress of your job flow, and then retrieve the output from Amazon S3 using the AWS management console, paying only for the resources that you consume.

  **REMEMBER** Though Hadoop is an attractive platform for many kinds of workloads, it needs a significant hardware footprint, especially when your data approaches scales of hundreds of terabytes and beyond. This is where Amazon EMR is most practical: as a platform for short term, Hadoop-based analysis or for testing the viability of a Hadoop-based solution before committing to an investment in on-premise hardware.

- **Hadapt (www.hadapt.com)**: Look for the product Adaptive Analytical Platform, which delivers an ANSI SQL compliant query engine to Hadoop. Hadapt enables interactive query processing on huge data sets (Hadapt Interactive Query), and the Hadapt Development Kit (HDK) lets you create advanced SQL analytic functions for marketing campaign analysis, full text search, customer sentiment analysis (seeing whether comments are happy or sad, for example), pattern matching, and predictive modeling. Hadapt uses Hadoop as the parallelization layer for query processing. Structured data is stored in relational databases, and unstructured data is stored in HDFS. Consolidating multistructured data into a single platform facilitates more efficient, richer analytics.

- **Karmasphere (www.karmasphere.com)**: Karmasphere provides a collaborative work environment for the analysis of big data that includes an easy-to-use interface with self-service access. The environment enables you to create projects that other authorized users can access. You can use a personalized home page to manage projects, monitor activities, schedule queries, view results, and create visualizations. Karmasphere has self-service wizards that help you to quickly transform and analyze data. You can take advantage of SQL syntax highlighting and code completion features to ensure that only valid queries are submitted to the Hadoop cluster. And you can write SQL scripts that call ready-to-use analytic models, algorithms, and functions developed in MapReduce, SPSS, SAS, and other analytic languages. Karmasphere also provides an administrative console for system-wide management and configuration, user management, Hadoop connection management, database connection management, and analytics asset management.
- **WANdisco (www.wandisco.com)**: The WANdisco Non-Stop NameNode solution enables multiple active NameNode servers to act as synchronized peers that simultaneously support client access for batch applications (using MapReduce) and real-time applications (using HBase). If one NameNode server fails, another server takes over automatically with no downtime. Also, WANdisco Hadoop Console is a comprehensive, easy-to-use management dashboard that lets you deploy, monitor, manage, and scale a Hadoop implementation
- **Zettaset (www.zettaset.com)**: Its Orchestrator platform automates, accelerates, and simplifies Hadoop installation and cluster management. It is an independent management layer that sits on top of an Apache Hadoop distribution. As well as simplifying Hadoop deployment and cluster management, Orchestrator is designed to meet enterprise security, high availability, and performance requirements.