**NIST Big Data**

**Reference Architecture**

DRAFT

**Version 0.3**

**Reference Architecture Subgroup**

**NIST Big Data Working Group (NBD-WG)**

**September, 2013**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Changes abstract | References | Editor |
| 0.1 | 9/6/13 | Outline and Diagram | M0202v1 | Orit |
| 0.2 | 9/11/13 | Text from input docs | M0142v3, M0189v1, M0015v1, M0126v4, M0231v1 | Orit |
| 0.3 |  | - Additions to the diagram based on a weekly conference call  - Clarification to the RA model based on email conversation  - Adding the Intro text | M-summary, M0239v1, | Orit |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

[Executive Summary 4](#_Toc367306999)

[1 Introduction 4](#_Toc367307000)

[1.1 Background 4](#_Toc367307001)

[1.2 Objectives 5](#_Toc367307002)

[1.3 How This Report Was Produced 6](#_Toc367307003)

[1.4 Structure of This Report 6](#_Toc367307004)

[2 Big Data System Requirements 6](#_Toc367307005)

[3 Conceptual Model 7](#_Toc367307006)

[4 Main Components 9](#_Toc367307007)

[4.1 Data Provider 9](#_Toc367307008)

[4.1.1 Data Service Abstraction 9](#_Toc367307009)

[4.2 Transformation Provider 10](#_Toc367307010)

[4.2.1 System Service Abstraction 11](#_Toc367307011)

[4.2.2 Usage Service Abstraction 11](#_Toc367307012)

[4.3 Capabilities Provider 11](#_Toc367307013)

[4.3.1 Capabilities Service Abstraction 12](#_Toc367307014)

[4.4 Data Consumer 12](#_Toc367307015)

[4.5 Vertical Orchestrator 12](#_Toc367307016)

[5 Security and Privacy 13](#_Toc367307017)

[6 Big Data Taxonomy 13](#_Toc367307018)

[Appendix A: Terms and Definitions 13](#_Toc367307019)

[Appendix B: Acronyms 13](#_Toc367307020)

[Appendix C: References 13](#_Toc367307021)

# Executive Summary

# Introduction

## Background

Big Data is the common term used to describe the deluge of data in our networked, digitized, sensor-laden, information driven world. The availability of vast data resources carries the potential to answer questions previously out of reach. Questions like: How do we reliably detect a potential pandemic early enough to intervene? Can we predict new materials with advanced properties before these materials have ever been synthesized? How can we reverse the current advantage of the attacker over the defender in guarding against cybersecurity threats?

Within this context, on 29 March, 2012 The White House announced the Big Data Research and Development Initiative[[1]](#footnote-1). The initiative’s goals were to help accelerate the pace of discovery in science and engineering, strengthen the national security, and transform teaching and learning by improving our ability to extract knowledge and insights from large and complex collections of digital data.

Six Federal departments and their agencies announced more than $200 million in commitments – spread across 80+ projects – that aimed to significantly improve the tools and techniques needed to access, organize, and glean discoveries from huge volumes of digital data. The initiative also challenged industry, research universities, and non-profits to join with the Federal government to make the most of the opportunities created by Big Data.

Despite the widespread agreement on the opportunities inherent to Big Data, a lack of consensus on some important, fundamental questions continues to confuse potential users and hold back progress. What are the attributes that define Big Data solutions? How is Big Data different from the traditional data environments and related applications we have encountered thus far? What are the essential characteristics of Big Data environments? How do these environments integrate with currently deployed architectures? What are the central scientific, technological, and standardization challenges that need to be addressed to accelerate the deployment of robust Big Data solutions?

The NIST Big Data program was formally launched on 13 June, 2012 to help answer some of the questions surrounding Big Data and to support the federal government effort to incorporate Big Data as a replacement for, or enhancement to, traditional data analysis systems and models where appropriate.

**[Editor’s Note:** Need some transition verbiage here. How did the first conference lead to the BD-PWG?]

On 19 June, 2013 NIST hosted the Big Data Public Working Group (BD-PWG) kickoff meeting to begin addressing those questions. The Group was charged with developing a consensus definition, taxonomy, reference architecture, and technology roadmap for Big Data that can be embraced by all sectors.

These efforts will help define and prioritize requirements for *interoperability*, *portability*, *reusability*, and *extendibility* for Big Data analytic techniques and technology infrastructure in order to support secure and effective adoption of Big Data.

The aim is to create vendor-neutral, technology and infrastructure agnostic deliverables to enable Big Data stakeholders to pick-and-choose best analytics tools for their processing and visualization requirements on the most suitable computing platforms and clusters while allowing value-added from Big Data service providers and flow of data between the stakeholders in a cohesive and secure manner.

Within the BD-PWG, the following working groups were chartered in order to provide a technically-oriented strategy and standards-based guidance for the federal Big Data implementation effort:

* Definitions and Taxonomies
* General Requirements
* Security and Privacy Requirements
* Reference Architectures
* Technology Roadmap

## Objectives

In general terms, a reference architecture provides “an authoritative source of information about a specific subject area that guides and constrains the instantiations of multiple architectures and solutions”[[2]](#footnote-2). Reference architectures generally serve as a reference foundation for solution architectures and may also be used for comparison and alignment purposes.

The broad goal of the Reference Architecture working group is to develop a Big Data open reference architecture that:

1. Provides a common language for the various stakeholders
2. Encourages adherence to common standards, specifications, and patterns
3. Provides consistency of implementation of technology to solve similar problem sets

The reference architecture is intended to facilitate the understanding of the operational intricacies in Big Data. It does not represent the system architecture of a specific Big Data system; instead it is a tool for describing, discussing, and developing system-specific architectures using a common framework of reference.

It provides a generic high-level conceptual model that is an effective tool for discussing the requirements, structures, and operations inherent to Big Data. The model is not tied to any specific vendor products, services or reference implementation, nor does it define prescriptive solutions that inhibit innovation.

The design of the NIST Big Data reference architecture serves the following objectives:

1. To illustrate and understand the various Big Data components, processes, and systems, in the context of an overall Big Data conceptual model;
2. To provide a technical reference for U.S. Government departments, agencies and other consumers to understand, discuss, categorize and compare Big Data solutions; and
3. To facilitate the analysis of candidate standards for interoperability, portability, reusability, and extendibility.

## How This Report Was Produced

The approach for developing this document involved four steps:

1. The first step was to announce a Big Data Reference Architecture Working Group open to the public in order to attract and solicit a wide array of subject matter experts and stakeholders in government, industry, and academia.
2. The second step was to gather available Big Data architectures and materials representing various stakeholders, different data types, and different use cases.
3. The third step was to examine and analyze the Big Data materiel to better understand existing concepts of Big Data, what it is used for, its goals, objectives, characteristics, and key elements, and then document the these using the Big Data taxonomies model.
4. The fourth step was to develop an open reference architecture based on the analysis of Big Data material and the inputs from the other NIST Big Data working groups.

## Structure of This Report

The remainder of this document is organized as follows:

Section 2 contains high level requirements relevant to the design of the Reference Architecture.

Section 3 represents a generic big data system comprised of technology-agnostic function blocks interconnected by interoperability surfaces.

Section 4 describes the main components of the generic system.

Section 5 addresses security and privacy.

Section 6 contains the Big Data taxonomy.

Appendix A lists the terms and definitions appearing in the taxonomy.

Appendix B contains the acronyms used in this document.

Appendix C lists the references used in the document.

# Big Data System Requirements

**[This section contains high level requirements relevant to the design of the Reference Architecture. This section will be further developed by the NIST BDWG Requirements SG.]**

The “big data” ecosystem is an evolution and a superset of a “traditional data” system exhibiting any or all of the following characteristics or requirements:

* Data sources are diverse in their security and privacy considerations and their business relationships with the data system integrators
* Data imported into the system vary in its structure, exhibit large volume, velocity, variety, and other complex properties
* The nature and the order of data transformations varies between vertical systems; it is not prearranged, and is evolving for a given system
* Storage technologies and databases are tailored to specific transformation needs and their scalability properties scale horizontally, vertically, or both
* Innovative analytic functions continuously emerge; proven technologies get enhanced and abstracted, resulting in frequent updates and outsourcing practices
* Data usage vary in structure and format; new use cases can be easily introduced to the system

# Conceptual Model

The NIST Big Data Reference Architecture (RA) shown on Figure 1 represents a generic big data system comprised of technology-agnostic functional blocks interconnected by interoperability surfaces. As such, it is applicable to a variety of business environments including tightly-integrated enterprise systems, as well as loosely-coupled vertical industries that rely on cooperation of independent stakeholders.

Note that because the RA doesn’t represent a specific system type, for the general case its functional blocks cannot be mapped to business stakeholders or authorities. Therefore, there are no RA boundaries or external vs. internal components to the BD system.



Figure 1: Big Data Reference Architecture

The RA is organized around two axes representing the two big data value chains: the information flow (along the vertical axis) and the IT integration (along the horizontal axis). Along the information flow axis, the value is created by data collection, integration, analysis, and applying the results down the value chain. Along the IT axis, the value is created by providing networking, infrastructure, platforms, application tools, and other IT services for hosting and operating of the big data in support of data transformations required for implementation of a specific application or a vertical. Note that the transformation block is at the intersection of both axes indicating that data analytics and its implementation are of special value to big data stakeholders in both value chains.

The five main RA blocks represent different technical roles that exist in every big data system: “Data Provider”, “Data Consumer”, “Transformation Provider”, “Capabilities Provider”, and “Vertical Orchestrator”. The two additional “Security & Privacy” and “Management” blocks are shown as cross-cutting sub-systems that provide services and functionality to the rest of the system components in the areas specific to “big data”.

Note that this RA allows to represent the stacking or chaining of big data systems, in a sense that a Data Consumer of one system could serve as a Data Provider to the next system down the stack or chain.

According to the big data taxonomy, a single actor can play multiple roles, as well as multiple actors can play the same role. This RA doesn’t specify the business boundaries between the participating stakeholders or actors. Therefore, each two roles can reside within the same business entity or can be implemented by different business entities.

The “DATA” arrows show the flow of data between the system’s main blocks. The data flows between the components either physically (a.k.a., by value) or by providing its location and the means to access it (a.k.a., by reference). The “SW” arrows show transfer of software tools for processing of big data *in situ*. The “Service Abstraction” blocks represent software programmable interfaces representing functional abstractions. Manual agreements (e.g., SLAs) and human interactions that may exist throughout the system are not shown in the RA.

# Main Components

## Data Provider

Data Provider is the role of introducing new information feeds into the big data system for discovery, access, and transformation by the big data system.

Note that new feeds are distinct from the data already being in use by the system and residing the various system repositories (including memory, databases, etc.) although similar technologies can be used to access both.

One of the important characteristics of a big data system is the ability to import and use data from a variety of data sources. Data sources can be internal and public records, online or offline applications, tapes, images/audio and videos, sensor data, Web log, HTTP cookies, etc. Data sources can be produced by humans, machines, sensors, Internet technologies, etc.

In its role, Data Provider creates an abstraction of data sources. In case of raw data sources, Data provider can potentially cleanse, correct, and store the data an internal format.

Frequently, the role of Data Provider and Transformation Provider would belong to different authorities, unless the authority implementing the Transformation Provider owns the data sources. Consequently, data from different sources may have different security and privacy considerations.

Data Provider can also provide an abstraction of data being transformed earlier by another system that can be either a legacy system or another big data system. In this case, Data Provider would represent a Data Consumer of that other system. For example, a (big) streaming data source could be generated by another system operation on (big) data at rest.

Data Provider activities include:

* Creating the metadata describing the data source(s), usage policies/access rights, and other relevant attributes
* Publishing the availability of the information and the means to access it
* Making the data accessible by other RA components using suitable programmable interface.

### Data Service Abstraction

The Data Service Abstraction would typically include a registry so that the transformation functions can locate a Data provider, identify what comparable data it contains, understand what types of access is allowed, understand what types of analysis are supported, where the data source is located, how to access the data, security requirements for the data, privacy requirements for the data, etc. As such, the interface would include the means for registering the data source, for querying the registry, and a standard set of data contained by the registry.

Because the data can be too large to economically move across the network, the interface could also allow to submit the analysis request (as a software code implementing a certain algorithm for execution) and the results returned to the requestor.

Subject to data characteristics (such as, volume, velocity, and variety) and system design considerations, interfaces for exposing and accessing data would vary in their complexity and can include both push and pull software mechanisms. These mechanisms can include subscription to events, listening to data feeds, querying for specific data properties or content, and the ability to submit a code for execution to process the data *in situ.*

Note that not all use of data service abstraction would always be automated, but rather might involve a human role logging into the system and providing directions where new data should be transferred (for example, via FTP).

## Transformation Provider

Transformation Provider is the role of executing a generic “vertical system” data life cycle including data collection from various sources, multiple data transformations being implemented using both traditional and new technologies, and diverse data usage.

The Transformation functions would typically be specific to the vertical application and therefore are not candidates for standardization. However, the metadata and the policies defined and exchanged between the transformation blocks could be standardized.

As the data propagates through the ecosystem, it is being processed and transformed in different ways in order to extract the value from the information. Transformation sub-components can be implemented by independent stakeholders and deployed as stand-alone services.

Each transformation function can use different specialized data infrastructure or capabilities best fitted for its requirements, and can have its own privacy and other policy considerations.

In its role, Transformation Provider typically executes the manipulations of the data lifecycle of a specific vertical system to meet the requirements or instructions established by the Vertical Orchestrator.

**[Editor’s Note: Listed activities need to be aligned with the sub-components shown on the diagram.]**

Transformation Provider activities include:

* Data Collecting (connect, transport, stage): obtains connection to Data Provider APIs to collect into local system, or to access dynamically when requested. At the initial collection stage, sets of data (e.g., data records) of similar structure are collected (and combined) resulting in uniform security considerations, policies, etc. Initial metadata is created (e.g., subjects with keys are identified) to facilitate subsequent aggregation or lookup method(s).
* Data Curating: provides cleansing, outlier removal, standardization for the ingestion and storage processes.
* Data Aggregating: Data from different data providers with easily correlated metadata (e.g., identical keys) is aggregated into a single data set. As a result, the information about each object is enriched or the number of objects in the collection grows. Security considerations and policies concerning the resultant data set are typically similar to the original data.
* Data Matching: Data from different data providers with disparate metadata (e.g., keys) are aggregated into a single data set. As a result, the information about each object is enriched. The security considerations and policies concerning the resultant data set may differ from the original policies.
* Data Optimizing (Pre-analytics): determines the appropriate data manipulations and indexes to optimize subsequent transformation processes.
* Data Analysis: implements the techniques to extract knowledge from the data based on the requirements of the data scientist, who has specified the algorithms to process the data to produce new insights that will address the technical goal.
* Data Transferring: facilitates secure transfer of data between different repositories and/or between the Transformation and the Capabilities RA blocks.

Note that many of these tasks have changed, as the algorithms have been re-written to accommodate and be optimized for the horizontally distributed resources.

### System Service Abstraction

TBD

### Usage Service Abstraction

TBD

## Capabilities Provider (or Service Provider or Capacity Provider)

Capabilities Provider is the role of providing a computing fabric (such as system hardware, network, storage, virtualization, and computing platform) in order to execute certain transformation applications, while protecting the privacy and integrity of data. The computing fabric facilitates a mix-and-match of traditional and future computing features from software, platforms, and infrastructures based on application needs.

Capabilities are abstracted functionalities that exist in order to support big data transformation functions. Capabilities include infrastructures (e.g., VM clusters, storage, networking), platforms (e.g., programming languages and runtimes, databases), and applications (e.g., analytic tools).

[Capability Provider | Service Provider | Capacity Provider] have traditionally been implemented by dedicated, custom-made on-premise systems. Recently, Big Data analytics and processing has been flourishing thanks to the adoption and use of cloud computing.

**~~[Editor’s Note: Activities need to be listed and described after the agreement is reached on the sub-components presented on the diagram.~~]**

### Traditional On-Premise Frameworks

Traditional capabilities needed to support data analytics require custom-designed computing resources in order to satisfy the heavy demand on computing, storage and database systems. Such on-premise systems usually use large, high performing custom-design databases designed to address unique demands of the data analytics workloads.

Such traditional solutions offer software, platform and cluster capabilities that can be used separately, or in conjunction with each other. The capabilities could be implemented in a layered fashion, or could be built vertically. Regardless, the implementations of the such capabilities will requires physical resources such as CPUs, storage devices, networking equipment, cooling and electrical facilities, etc.

### Cloud Service Providers

Recent data analytics solutions use algorithms that can utilize and benefit from the capabilities of the cloud computing systems.

Cloud computing has essential characteristics such as rapid elasticity and scalability, multi-tenancy, on-demand self-service and resource pooling which together facilitate the realization of big data implementations.

In Cloud computing systems the **cloud service provider** implements and delivers **cloud services**. Processing of a service invocation is done by means of an instance of the service implementation, which may in turn involve the composition and invocation of other services as determined by the design and configuration of the service implementation.

#### Cloud Service Component

The cloud service component contains the implementation of the cloud services provided by a **cloud service provider**. It contains and controls the software components that implement the services (but not the underlying hypervisors, host operating systems, device drivers, etc.)

The **cloud Capabilities Types** inside the cloud service component are significant. The cloud service component offers its services via a service interface whose capabilities is in turn graphically represented by the inverted L diagrams. Each L represents the service interface(s) associated with the respective **Cloud Capability Type** that is implemented by the **Cloud Service Category** (e.g. IaaS, PaaS and SaaS)**.** The relative positioning of the L diagrams are also significant. The implication of the L shapes is that application capabilities can be implemented using platform capabilities or not (at the choosing of the cloud service provider) and that platform capabilities can be implemented using infrastructure capabilities or not.

Cloud services can be described in terms of the cloud capabilities which they offer, based on the resources provided by the cloud service. There are three cloud capabilities types:

• Application capabilities

• Platform capabilities

• Infrastructure capabilities

Cloud services are also grouped into categories, where each category of services has some qualities that are common between the services. The services in these categories may include capabilities from one or more than one of the capabilities types above. Some common cloud service categories include:

• Infrastructure as a Services (IaaS)

• Platform as a Service (PaaS)

• Software as a Service (SaaS)

• Network as a Service (NaaS)

#### Resource Abstraction & Control

The Resource Abstraction & Control component is used by cloud service providers to provide access to the physical computing resources through software abstraction. Resource abstraction needs to ensure efficient, secure, and reliable usage of the underlying physical resources. The control feature of the component enables the management of the resource abstraction features.

The Resource Abstraction & Control component enables a cloud service provider to offer qualities such as rapid elasticity, resource pooling, on-demand self-service and scale-out. The Resource Abstraction & Control component can include software elements such as hypervisors, virtual machines, virtual data storage, and time-sharing.

The Resource Abstraction & Control component enables control functionality, enabling monitoring and management capabilities implemented in the Operational Support Systems component. For example, there may be a centralized algorithm to control, correlate and connect various processing, storage and networking units in the physical resources so that together they deliver an environment where IaaS, PaaS or SaaS cloud service categories can be offered. The controller might decide which CPUs/racks contain which virtual machines executing which parts of a given cloud workload, and how such processing units are connected to each other, and when to dynamically and transparently reassign parts of the workload to new units as conditions change.

#### Physical Resources

This layer represents physical resources needed for supporting the cloud computing system. Example of such physical resources are CPUs, storage devices, networking equipment, cooling and electrical facilities, etc.

#### Cross cutting aspects

Acting as a foundation for the above activities and components are cross cutting aspects, Security, Privacy, Resilience, Performance, Portability, Interoperability and Governance.

### ~~Capabilities Service Abstraction~~

~~TBD~~

## Data Consumer

Data Consumer is the role performed by end users or other systems in order to use the results of data transformation. Data Consumer uses the Usage Service Abstraction interface to get access to the information of its interest. Usage Services Abstraction can include data reporting, data retrieval, and data rendering.

Data Consumer activities can include:

* Exploring data using data visualization software
* Ingesting data into their own system
* Putting data to work for the business, for example to convert knowledge produced by the transformers into business rule transformation

Data Consumer can play the role of the Data Provider to the same system or to another system. Data Consumer can provide requirements to the Vertical Orchestrator as a user of the output of the system, whether initially or in a feedback loop.

## Vertical Orchestrator

Vertical Orchestrator is the role of defining and integrating the required data transformations components into an operational vertical system. Typically, Vertical Orchestrator would represent a collection of more specific roles performed by one or more actors, which manages and orchestrates the operation of the Big Data System.

The Big Data RA represents a broad range of big data systems: from tightly-coupled enterprise solutions (integrated by standard or proprietary interfaces) to loosely-coupled verticals maintained by a variety of stakeholders or authorities bounded by agreements and standard or standard-de-facto interfaces.

In an enterprise environment, the Vertical Orchestrator role is typically centralized and can be mapped to the traditional role of System Governor that provides the overarching requirements and constraints which the system must fulfill, including policy, architecture, resources, business requirements, etc. System Governor works with a collection of other roles (such as Data Manager, Data Security, and System Manager) to implement the requirements and the system’s functionality.

In a loosely-coupled vertical, the Vertical Orchestrator role is typically decentralized. Each independent stakeholder is responsible for its system management, security, and integration. In this situation, each stakeholder is responsible for integration within the big data distributed system using service abstractions provided by other stakeholders.

In both cases (i.e., tightly and loosely coupled), the role of the Vertical Orchestrator can include the responsibility for

* Translating business goal(s) into technical requirements.
* Supplying and integrating with both external and internal Providers.
* Overseeing evaluation of data available from Data Providers.
* Directing the Transformation Provider by establishing requirements for the collection, curation, analysis of data, etc.
* Overseeing transformation activities for compliance with requirements.

# Security and Privacy

**[This section will be prepared by the NIST BDWG Security and Privacy SG and will contain high level security and privacy considerations relevant to the design of the Reference Architecture.]**

# Big Data Taxonomy

**[This section will be prepared by the NIST BDWG Def&Tax SG and will contain high level taxonomy relevant to the design of the Reference Architecture.]**

# Appendix A: Terms and Definitions

**[This section will contain terms and definitions used in this document.]**

# Appendix B: Acronyms

# Appendix C: References

1. http://www.whitehouse.gov/blog/2012/03/29/big-data-big-deal [↑](#footnote-ref-1)
2. Department of Defense Reference Architecture White Paper, June 2010 [↑](#footnote-ref-2)