IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

_____

BEFORE THE PATENT TRIAL AND APPEAL BOARD

_____

AMAZON.COM, INC., AMAZON.COM SERVICES LLC,
Petitioner,
v.

NOKIA TECHNOLOGIES OY,
Patent Owner.

_____

IPR2024-01507

U.S. Patent No. 8,996,693

_____

PETITION FOR *INTER PARTES* REVIEW

OF U.S. PATENT NO. 8,996,693

TABLE OF CONTENTS

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 8,995,693 to Boldyrev et al.("the '693 patent")
1002	Prosecution History for the '693 patent
1003	Declaration of M. E. Crovella
1004	Foster et al., *IBM InfoSphere Streams: Assembling Continuous Insight in the Information Revolution* (2011) ("Foster")
1005	Williams et al., *Implementing CICS Web Services* (2007) ("Williams")
1006	Declaration of N. E. Frank-White re Foster (Ex-1004)
1007	Declaration of N. E. Frank-White re Williams (Ex-1005)
1008	Declaration of M. Spicer re Foster (Ex-1004)
1009	Declaration of M. Ebbers re Williams (Ex-1005)
1010	U.S. Patent App. Pub. No. 2008/0127064A1 to Orofino et al. ("Orofino")
1011	Kumar et al., *DEDUCE: At the Intersection of MapReduce and Stream Processing* (2010) ("Kumar")
1012	Verma et al., *ARIA: Automatic Resource Inference and Allocation for MapReduce Environments* (2011) ("Verma")
1013	7 Blackwell Publ'g, The Blackwell Encyclopedia of Management, *Management Information Systems* (Davis ed., 2d ed. 2005) (excerpts)
1014	U.S. Patent. App. Pub. No. 2012/0083715A1 ("Yuen")
1015	O'Reilly & Associates, *JAVA Enterprise in a Nutshell* (2001) ("O'Reilly")
1016	U.S. Patent. App. Pub. No. 2004/0031037A1 to Ikoma et al. ("Ikoma")
1017	Crockford, *Request for Comments 4627* (2006) ("RFC4627")

Data has long been collected and processed to understand the world around us. The invention of computing technologies has led to the automation of such analytical processes. Databases were built to amass information and tools were devised to apply analytical processing on the collected data. This type of data processing is called static processing. Advances in communication bandwidth and processing power has made it possible to collect and process information in virtually real-time. Data processing to data streams is called dynamic processing. Persons of skill in the art knew that both approaches had their respective advantages and devised ways to combine the two. *Infra* §I.A.

The '693 patent does not purport to invent either dynamic or static processing, the apparatuses for carrying out such processing, or the techniques used to interface between two processing mechanisms. Ex-1001, 1:18-36; *infra* §II.A. Instead, it purports to allow for processing data with both dynamic and static processing and combining the results. Ex-1001, 1:32-36. The '693 patent was allowed because the Examiner accepted the Applicant's argument that the art of record did not disclose transferring statically-processed data back to the dynamic processing mechanism from which it came in the first place. *Infra* §II.B. However, the prior art reference Foster discloses this feature. It discloses enriching and bootstrapping ingested streaming data at a back-end server using historic data, and *sending it back to the front-end stream processor* from which the

data had come. This Petition demonstrates that the challenged claims are rendered obvious by Foster alone and in combination with other prior art references, none of which were previously presented to the Office.

I. BACKGROUND

Dynamic data processing refers to processing data that is in motion, such as detecting contextual trends in an incoming live stream of Tweets and social media posts. Static data processing refers to processing data that is at rest, such as analyzing a collection of historic sales and demographic data to predict demand for a product in a new market. Even before the '693 patent was filed in 2012, persons of skill in the art were combining both approaches to provide better data analysis. For example, in the automated stock trading domain, static processing was used to generate models that were applied to the dynamic processing of incoming stock market events to determine whether to carry out trades; these models were updated periodically with newly accumulated data to improve their performance. Ex-1011, Abs.; Crovella (Ex-1003), ¶3.

A. Static and Dynamic Processing

By 2012, various mechanisms were known and used for static processing. Static processing hails back to at least the 1960s when "hierarchical databases" were invented to "capture[] and store[] that data" for post facto processing "to captur[e] and understand[] what has happened." Ex-1004, §1.1.3 at 00034. These

"online transaction processing (OLTP) systems" laid the path for "online analytic processing (OLAP)" that used techniques such as data mining on the stored data to "understand[] why things happened to make more informed recommendations." *Id.*, §1.1.3 at 00035. The commonality of these traditional approaches is that "the data is stored," and therefore the processing occurs on static, accumulated data. *Id.*; Crovella, ¶4.

Over time, technical advances increased the amount and timeliness of data available for processing. Ex-1004, §1.1 at 00022-24, §1.1.2 at 00029-32. Such real-time data was "considered streaming data or data in motion." *Id.*, §1.1 at 00025. The flood of real-time information and increased processing power and techniques gave birth to dynamic processing in order to provide "faster analytics." *Id.*, §1.1.3 at 00036. Such dynamic processing mechanisms and techniques were commonly known as "real-time analytic processing (RTAP)." *Id.*; Crovella, ¶5.

Thus, two computing paradigms were divided based on the temporal nature of the data: the traditional static processing of "data-at-rest (historical data)" and the newer dynamic processing of "data-in-motion (current data)." Ex-1004, §1.1.4 at 00037, §1.1 at 00025-26, Crovella, ¶6.

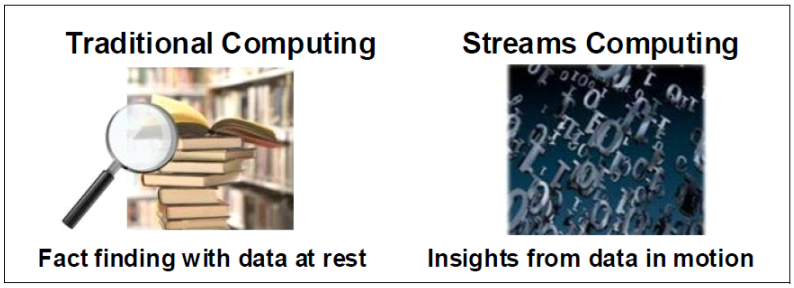


Figure 1-2 Comparing traditional and stream computing

Ex-1004, Fig. 1-2 at 00025. Because the data processed was either real-time data or accumulated slow-moving data, these processing techniques could also be called "real-time" or "slow-moving" processing, respectively. Crovella, ¶6.

Further development in the data processing field led to using both these processing methods together. *E.g.*, Ex-1010, Abs., ¶33 (processing "real-time data, pseudo real-time data and/or archived data."). Various systems and methods were employed to combine the results of static and dynamic processing to provide the "seamless processing of current and historical data." Ex-1004, §7.1 at 00338; *see, e.g.*, *id.*, Fig. 7-1 ("real-time and historical insights"):

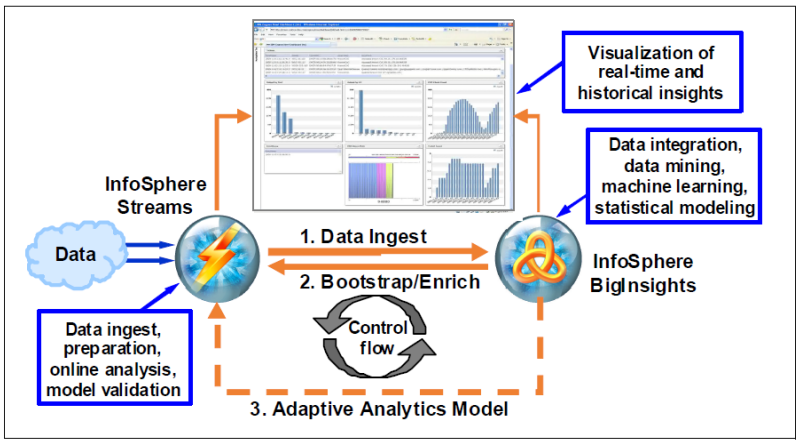


Figure 7-1 Big Data application scenarios

See also, e.g., Ex-1011, §1; Crovella, ¶7.

B. Batch Processing

The claims of the '693 patent refer to a batch mode of static processing, which was well-known and common for processing slowly accumulating data. Data would be processed intermittently in batches, i.e., a number of data items grouped together. Batch processing was used in various algorithms and methods. *E.g.*, Ex-1014, ¶133. Batch processing could be scheduled by frequency or periodicity, such as on daily, weekly, or monthly schedules. MapReduce was a well-known static processing algorithm commonly run on periodic batches. Ex-1012, §1 at 00001, §8 at 00009; Ex-1015, ¶6; Crovella, ¶8.

C. Marshalling and Unmarshalling Data

The claims of the '693 patent refer to marshalling and unmarshalling data, which was well-known. Marshalling was a generic term for gathering data from one process and converting it into a format for storage or for transmission to another process. Ex-1005, §1.4.3 at 00039; *see also, e.g.*, Ex-1015, 000064. It was well-known that unmarshalling was the inverse process to undo the conversion and extract the original data. *See* Ex-1016, ¶2; Crovella, ¶9.

JSON (JavaScript Object Notation) was a well-known format for marshalling data that was first officially described in RFC4627. Ex-1017, 00001. It was commonly used as a data-interchange format. Crovella, ¶10.

II. '693 PATENT

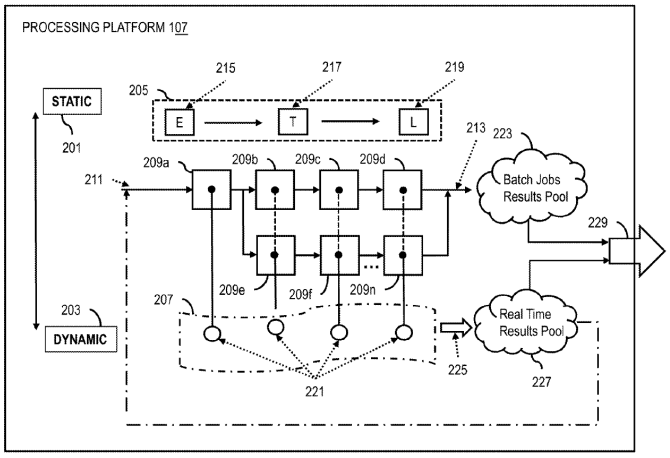
A. Alleged Invention

The '693 patent provides a "method and apparatus for providing dynamic stream processing of data based on static analytics." Ex-1001, Title. The alleged invention is directed to "a method, apparatus, and computer program for smart data processing methods." Ex-1001, 3:35-47. The patent does not purport to invent "dynamic processing mechanisms," "static processing mechanisms," or "processing elements." Ex-1001, Abs. Rather, it acknowledges "data processing architectures and techniques that traditionally have been used" for the processing of both "'slow moving' data (e.g., static or relatively static data)" and "data [] collected as data streams in real-time." Ex-1001, 1:22-36. The '693 patent also notes that "[t]raditionally, data processing and analysis may be implemented via one or more servers (or nodes) and/or clusters of servers (or nodes) that provide, for instance, distributed computing and/or data storage to support the services." Ex-1001, 1:18-21; Crovella, ¶11.

The '693 patent discusses the "need for data processing methods to generate comprehensive results that combine[] the features of a slow moving data systems with real-time data systems." Ex-1001, 1:40-42. The patent purportedly solves the "technical challenges to managing and/or integrating the data processing and analysis methods for slow moving data with the data processing and analysis

methods for real-time data." Ex-1001, 1:32-36. Specifically, the patent proposes "introduc[ing] the capability for utilization of static and dynamic data analytics in various data processing mechanisms." Ex-1001, 4:23-49. Figure 2 illustrates a proposed arrangement to combine the two approaches to processing data. Ex-1001, 11:6-12:60.

FIG. 2



Ex-1001, Fig. 2. According to the patent, this provides the benefit of generating "more accurate and comprehensive results." Ex-1001, Abs., 3:35-52; Crovella, ¶12.

However, processing and analyzing slow-moving data and real-time data was already known in the art, as well as techniques for combining the two processing methods. *Supra* §I.A. The '693 patent merely recites mundane steps using well-known techniques in the field of data processing and analysis, that amount to little more than determining data to process, preparing the data for transfer to or use at a different system, transferring the data, processing the data,

and transferring it back. *See, e.g.*, Ex-1001, 25:31-44 (claim 1), Abs., 1:43-24. In short, the '693 patent "combines the features of a slow moving data systems with real-time data systems" by sending data back and forth to be processed by both types of systems. Ex-1001, 1:40-42; Crovella, ¶13.

Indeed, the single point of novelty the Applicant argued during prosecution was that the prior art transferred the data processed at the static processing application to another dynamic processing application instead of the original dynamic processing application that it came from. *Infra* §II.B. But transferring the data back to the original dynamic processing mechanism was known in the art. *Infra* §VII.A.3.[1e]; Crovella, ¶14.

B. Prosecution History

The '693 patent issued March 31, 2015 from Application No. 13/621,511 ("the '511 application"). It was allowed after a single non-final office action rejecting the claims under 35 U.S.C. §102, and a §112 objection to the preamble. Crovella, ¶15.

In the only office action, the Examiner rejected claims 1-20 as anticipated by the prior art reference Siripurapu. Ex-1002, 00124-127. As-filed independent claim 1 only recited the determining and marshalling limitations of the issued patent and specified that the marshalled data object is processable by a static

processing mechanism, instead of claiming processing by a static processing mechanism. *Id.*, 00039; Crovella, ¶16.

To overcome the rejection, claim 1 was amended to further recite transferring the data object to a static processing mechanism, processing the data object by the static processing mechanism, and transferring the processed data object back to the dynamic processing mechanism. Ex-1002, 00136. In connection with this amendment, Applicant argued that Siripurapu transfers the processed data to a *different* streaming application, rather than "back to" the *original* application that marshalled and transferred the data in the first place. Ex-1002, 00147-148. A notice of allowance issued without stating any specific reason for allowance. Ex-1002, 00169-182. Crovella, ¶¶17-19.

C. Priority Date

This Petition demonstrates that the challenged claims are invalid based on prior art that predates the '693 patent's earliest application filed on September 17, 2012. Crovella, ¶20.

III. IDENTIFICATION OF CHALLENGE

A. Statutory Grounds

Petitioner requests *inter partes* review and cancelation of the challenged claims on the following grounds:

Ground	Claims	Statutory Basis	Prior Art
1	1-5, 7-15, 17-20	§103	Foster
2	1-5, 7-15, 17-20	§103	Foster in view of Williams

Grounds 1-2 render the challenged claims obvious because any differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious at the time the invention was made to a POSITA. *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007); Crovella, ¶35.

B. Relied-Upon Prior Art

1. Foster (Ex-1004)

Foster is prior art under pre-AIA §102(a) because it was published (October 2011) before the alleged invention (earliest filing date September 17, 2012). It was not cited during prosecution of the '693 patent. Foster was publicly accessible through IBM's webpage for its Redbooks series. Crovella, ¶¶37-40.

Foster is directed to big data analysis techniques and tools running on distributed computing systems. Ex-1004, §1.1.4 at 00037. In particular, Foster discusses the implementation and operation of IBM InfoSphere Streams and IBM BigInsights. *Id.* Streams was "architected specifically to help clients continuously analyze massive volumes of streaming data at extreme speeds." *Id.*, §1.2 at 00037-51. Streams was paired with BigInsights, which provides "data-at-rest (historical) analysis." *Id.*, §1.1.4 at 00037. BigInsights integrated industry standard static data

processing technology with IBM's own homegrown technology. *Id.* ("BigInsights delivers an enterprise-ready big data solution by combining Apache Hadoop, including the MapReduce framework and the Hadoop Distributed File Systems (HDFS), with unique technologies and capabilities from across IBM."); Crovella, ¶41.

Foster describes, among other potential applications, "the approach of integrating Streams with…BigInsights." Ex-1004, 00337-345:

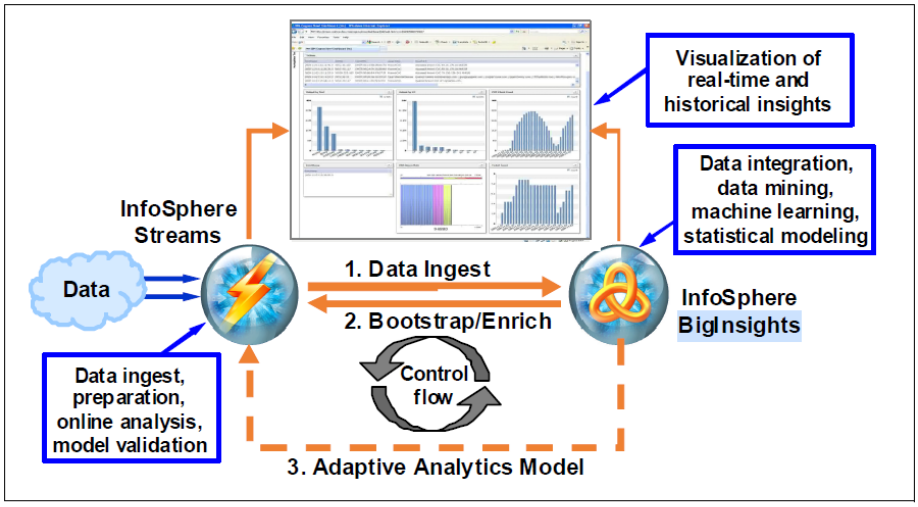


Figure 7-1 Big Data application scenarios

Id., Fig. 7-1 at 00339. In particular, Foster explains how incoming real-time streams of data can be prepared at Streams for ingestion to BigInsights, and further processed at BigInsights to enrich the data and to build adaptive analytics models that are then used for the ingestion and analytics at Streams. *Id.*, §7.1.1-4 at 00338-341. Foster explains the "interactions between Streams and BigInsights" to provide these functions, using a number of examples. *Id.*, §7.1.6 at 00343, §7.1.5-

6 at 00342-344 (discussing use case "tracking positive and negative sentiment being expressed about products"), §7.1.2-4 at 00339-341 (discussing uses cases of "Call Detail Record (CDR) processing" and "social media based lead-generation"); Crovella, ¶42.

Foster is analogous art because it is in the same field of endeavor as the '693 patent, which is data processing and analysis. *Supra* §IV. Foster discusses technologies related to data analytics, including "real-time analytic processing (RTAP)." Ex-1004, §I at 00021, §1.1.3 at 00034-36. Foster presents examples of solutions that combine real-time analysis with static analysis. *Id.*, §1.2.3 at 00047-51. Foster is reasonably pertinent to the problem addressed by the '693 patent's inventor, because it attempts to solve the similar need for "products and tools designed to" "analyze and take action in real time" using an ever-growing volume of real-time data *Id.*, §1.1.1-2 at 00026-33; Crovella, ¶43.

2. Williams (Ex-1005)

Williams is prior art under pre-AIA §§102(a) and 102(b) because it was published (October 2007) before the alleged invention and more than one year before the earliest filing date (September 17, 2012). It was not cited during prosecution of the '693 patent. Williams was publicly accessible through IBM's webpage for its Redbooks series. Crovella, ¶¶44-47.

Williams discloses well-known techniques for "using Internet protocols and standards." Ex-1005, §1 at 00025. In particular, Williams discusses the use of "encodings" in "distributed computing environments." *Id.*, §1.4.3 at 00039. Williams explains that the steps "translat[ing] to and from a protocol format" were known as "*marshalling* and *unmarshalling*."[1] *Id.* (emphasis in original); Crovella, ¶48.

Williams is analogous art because it is in the same field of endeavor as the '693 patent, which is data processing and analysis. *Supra* §IV. Williams discusses technologies related to "technologies that support the connecting or sharing of resources and data in a very flexible and standardized manner." Ex-1005, §1.1 at 00026. Williams is reasonably pertinent to the problem addressed by the '693 patent's inventor, because it teaches various techniques related to "distributed computing" (*id.*, §1.2.2 at 00028), which the '693 patent states were commonly used for implementing "data processing and analysis" (Ex-1001, 1:18-21); Crovella, ¶49.

IV. LEVEL OF ORDINARY SKILL IN THE ART

The field of the '693 patent is data processing and analysis. A hypothetical person of ordinary skill in the art ("POSITA") in the field of the '693 patent would

[1] All emphasis added unless otherwise noted.

have had a (1) bachelor's degree in computer science, electrical engineering, computer engineering, or a comparable field of study, and (2) approximately two years of practical experience with data processing and analysis. Additional experience can substitute for the level of education, and vice-versa. Crovella, ¶¶22-24.

In the 2012 timeframe, a POSITA would have known and had the skills necessary to:

- Design/implement data processing systems, including distributed computing systems;
- Implement techniques for static data analysis and real-time data analysis (including, e.g., known techniques such as MapReduce, sentiment analysis, social network analysis, association rule learning, clustering, machine learning, stream processing);
- Design and use data structures and models (including those for Big Data analysis and distributed computing);
- Use database warehousing, OLTP (On-Line Transactional Processing), OLAP (On-Line Analytical Processing), RTAP (Real-time Analytic Processing) tools and protocols;
- Marshall and unmarshall (e.g.., serialize and deserialize) data;
- Implement known algorithms to process data; and

- Design/implement displays and interfaces.

Crovella, ¶25.

V. CLAIM CONSTRUCTION

A. "processing element"

"Processing element" means "data to be processed" because the '693 patent uses the term in this way in its claims and specification. Independent claim 1 uses processing element (PE) in the sense of data to be processed. Ex-1001, 25:33-44 ("determining…at least one processing element…; marshalling…of the at least one processing element as at least one data object; processing of the at least one data object"). The specification explains that PEs are destined to be processed. *Id.*, 12:32-48 ("[T]he data is loaded into an end target (e.g., data warehouse, service provider, intermediate application, etc.)."); Crovella, ¶¶26-27.

The '693 patent does not limit a PE to any specific type of data; rather, a PE "may include computational *code* and/*or data*, which may be associated with one or more information items from a user and/or a service provider." Ex-1001, 6:63-7:1, 7:1-16 ("[A] PE may include data…."). Examples include "data feeds, users, service providers (e.g., Twitter®), search engines, content providers, etc." *Id.*, 3:67-4:5, 6:58-7:1 (same), 15:54-61 (same); Crovella, ¶28.

B. "dynamic processing mechanism"

"Dynamic processing mechanism" means a "mechanism for processing data in real-time or near real-time" because the '693 patent explains that "data processing mechanisms (e.g., engines, pipelines, etc.) may include a dynamic stream processing (e.g., *real-time or a near real-time*)… where the dynamic stream process may operate on one or more data feeds from various sources (e.g., users, service providers, sensors, etc.)." Ex-1001, 5:4-11. And dynamic processing carried out by such mechanisms is considered a synonym to "stream processing." *Id.*, 3:67-4:5 *("[I]n the dynamic processing (e.g., stream processing), the data may be processed near real-time*, wherein the data may be considered to be fast-moving (e.g., near real-time) and from various sources, for example, data feeds, users, service providers (e.g., Twitter®), search engines, content providers, etc."); Crovella, ¶¶29-30.

The '693 patent's embodiments confirm this understanding, explaining that "the dynamic processing mechanism 601 receives *real-time and/or near real-time data* from one or more data feeds 605 (e.g., one or more data and/or service providers), where the data *is processed by a stream processing engine 607* (e.g., stream processing framework, context mapping algorithms and data-structures, etc.)." Ex-1001, 16:43-56, 17:9-17 ("[T]he dynamic processing mechanism 601 operates in substantially real-time…."), Figs. 6-7 (showing stream processing

engine 607 in dynamic processing mechanism 601 receiving and processing real-time data feed 605); Crovella, ¶31.

C. "static processing mechanism"

"Static processing mechanism" means a "mechanism for processing for processing historical or slow-moving data" because the '693 patent explains that "data processing mechanisms (e.g., engines, pipelines, etc.) may include… a static processing (e.g., map-reduce batch jobs)…where…the static process may operate on data from one or more databases (e.g., historical data)." Ex-1001, 5:4-11. "[H]istorical data may include large volumes of contextual data collected from various sources over time (e.g., hours, days, weeks, etc.)." *Id.*, 5:11-17. Because static processing analyzes "slow-moving data," they operate, for example, periodically in large batches. *Id.*, 3:58-67 ("In the static processing method, as the database sizes may reach multiple gigabytes and terabytes, the data is processed according to one or more predetermined schedules (e.g., daily/weekly batch jobs), wherein the data may be considered to be slow-moving data and available from various sources (e.g., data stores.)"); Crovella, ¶¶32-33.

Dependent claims 9/19 and 10/20 recite limitations consistent with this understanding, which is further confirmed by the description of various embodiments in the specification. *Id.*, 26:25-37 (claiming "batch mode with a predetermined batch frequency" (claim 9) and "performing…slow moving data

analytics" (claim 10)), 28:13-25 (same for claims 19/20), 7:24-35 (same), 7:61-8:2 (static processing mechanisms (e.g., batch jobs)"), 16:45-48 (same), 17:12-24, 16:56-6 ("a map reduce engine 609 to manage and process data"), Figs. 6-7 ("Map Reduce Engine 609"); Crovella, ¶34.

VI. CHALLENGED CLAIMS

A. Grounds 1 and 2

The challenged claims (*supra* §IV.A) are obvious over Foster alone (Ground 1), and the combination of Foster with Williams (Ground 2). Both grounds teach all limitations of the challenged claims. Crovella, ¶36.

1. Independent Claim 1

Claim 1 is rendered obvious by Foster alone. Claim 1 is also rendered obvious by Foster in view of Williams. Crovella, ¶¶50-105.

[1pre] A method comprising:

The preamble is taught by Foster's **method** for processing real-time and slow-moving data on a distributed computing platform, and the combination with Williams teaches the preamble for the same reasons. Crovella, ¶¶51-53. For example, Foster discloses "a Big Data platform, based on two products: IBM InfoSphere Streams and IBM InfoSphere BigInsights" and "describe[s] various scenarios where data analysis can be performed across the two platforms to address the Big Data challenges." Ex-1004, §1.1.4 at 00037. As part of the operations of

these platforms, Foster describes methods for the "[t]he integration of data-in-motion (InfoSphere Streams) and data-at-rest (BigInsights) platforms [that] addresses three main application scenarios." *Id.*, §7.1.1 at 00338-339. The "scalable data ingest," "bootstrap and enrichment," and "adaptive analytics" scenarios encompass a number of functions and operations performed by Streams and BigInsights, such as processing data, marshalling data, updating models, and transferring data discussed below for limitations [1a]-[1e]. Crovella, ¶52.

[1a]	**determining via a processor at least one processing element of at least one dynamic processing mechanism;**

Foster teaches limitation [1a] by disclosing ingesting streaming data at Streams which is determined to be sent to BigInsights, and the combination with Williams teaches this limitation for the same reasons. Ex-1004, §7.1.1-2 at 00338-340; Crovella, ¶¶54-61. Foster teaches **determining…at least one processing element of at least one dynamic processing mechanism**. For example, Foster discloses "scalable data ingest" processing by Streams that determines data from input streams to be further processed. Ex-1004, §7.1.1 at 00338, §7.1.2 at 00339-340 ("Data from various systems arrives continuously, that is as a continuous stream…. Data needs to first be processed for extracting all the required data for consumption by downstream analytics."); Crovella, ¶55:

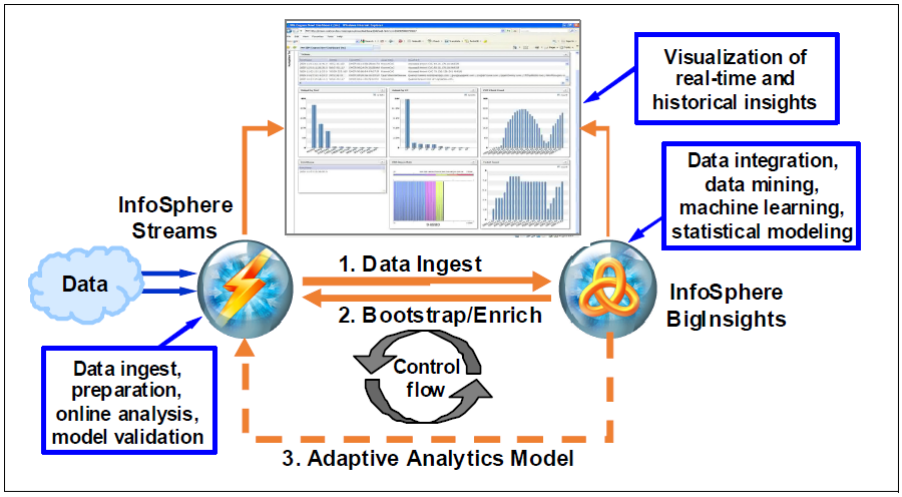


Figure 7-1 Big Data application scenarios

Ex-1004, Fig. 7-1 at 00339 ("Data Ingest").

The data determined to be sent from Streams to BigInsights is a **processing element** in the context of the '693 patent, because it is earmarked to be further processed (e.g., bootstrapped and enriched) at BigInsights. Ex-1004, §7.1.3 at 00340 ("BigInsights can be used to analyze data…. Results from this analysis…are also used to enrich incoming data with additional attributes required for downstream analytics."). As the '693 patent states, a PE "may include computational code and/*or data*." *Id.*, 6:63-7:6; *supra* §V.A; Crovella, ¶56.

Streams, including its data ingest function, is a **dynamic processing mechanism**, because Foster discloses it is a mechanism for processing data in real-time or near real-time. *See supra* §V.B. For example, Foster discloses that Streams' data ingestion and preparation functions "[c]ontinuous[ly] ingest" data and provide "real-time observations" to the downstream functions on BigInsights.

Ex-1004, §7.1.1 at 00338-339. "Streams supports…huge volumes of data…to be *analyzed and acted upon with low latency in close to or in real time*." *Id.*, §2.1 at 00056-57; Crovella, ¶57:

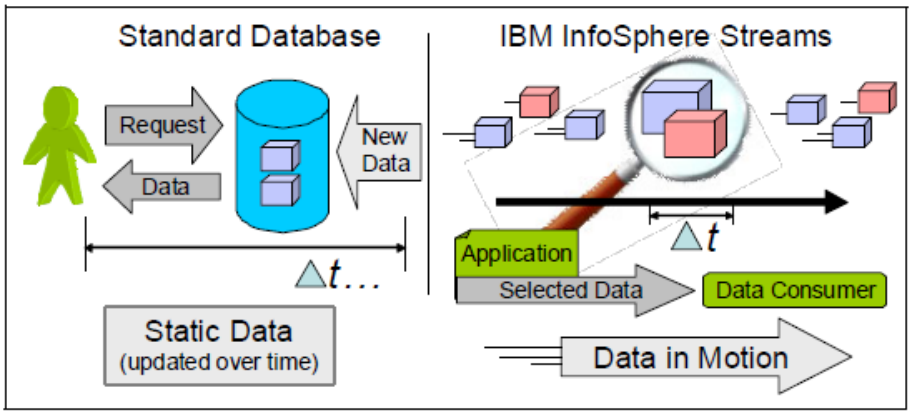


Figure 2-2 A standard relational database compared to IBM InfoSphere Streams

Ex-1004, Fig. 2-2 at 00058 ("Data in Motion").

Indeed, Foster explains "Streams supports a widely *dynamic* data model and data," where "the data of interest within the stream is defined, and then *processed as it flows by* the application." Ex-1004, §2.1 at 00058. Streams' real-time data ingestion contributes to "the seamless functioning of data-in-motion (current data) and data-at-rest (historical data) analysis." *Id.*, §7.1 at 00338, §7.14 at 000341 (same). This is reflected in the reports presented to users, as shown in Figure 7-1. Crovella, ¶58:

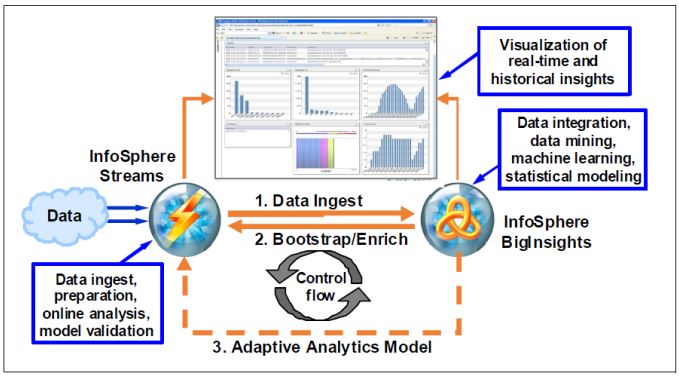


Figure 7-1 Big Data application scenarios

Ex-1004, Fig. 7-1 at 00339 ("Visualization of real-time…insights").

Foster teaches the PE (data) is **determin[ed] via a processor** because it discloses that Streams is an application run on a processor. For example, Foster discloses that Streams instances "execut[e] across one or more host computers." Ex-1004, §2.2.1 at 00060-64 ("An Application host is dedicated to executing the actual Streams applications that were previously described as being the queries that run continuously inside Streams."). Figure 2-3 shows how the Streams instance is comprised of Streams applications running on hosts.

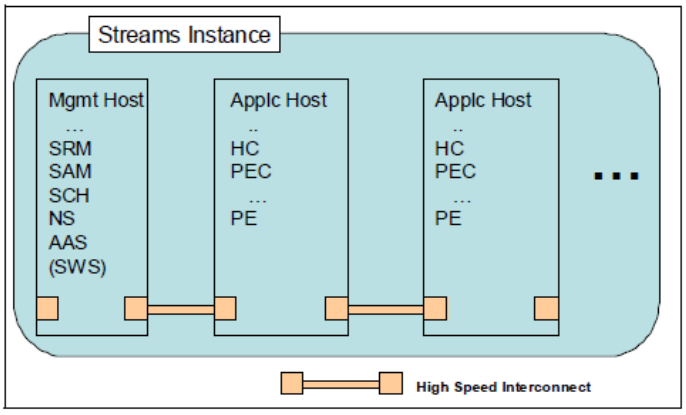


Figure 2-3 Streams instance, hosts, and more

Id., Fig. 2-3 at 00041. It was well-known to carry out processing on such host computers with processors. For example, Foster teaches using an x86 architecture processor. *Id.*, 000050 ("IBM developed an InfoSphere Streams based trading prototype *running on a single 16 core x86 computer*…."); *see also*, *id.*, Fig. 1-8 ("x86 multicore") at 00040:

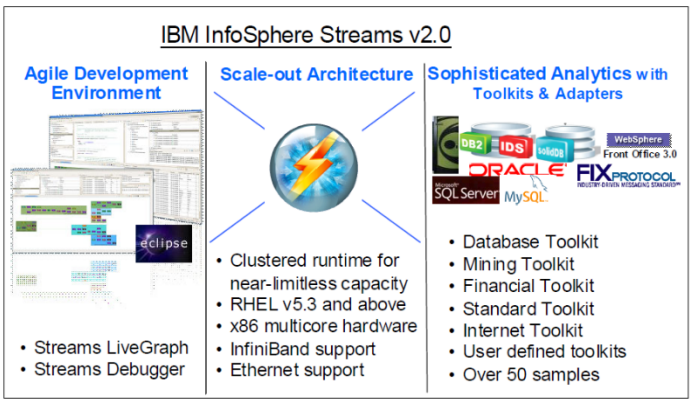


Figure 1-8 The components of InfoSphere Streams V2.0

Crovella, ¶59.

Foster further explains that Streams applications are a "collection of operators" that "defines how the run time should analyze a set of stream data." Ex-1004, §2.2.2 at 00064. This includes the processing for data ingestion and preparation. *Id.*, §7.1.6 at 00343 ("Streams processes data using a flow graph of interconnected operators."), §7.1.4 at 00341 ("operators on Streams continue to observe incoming data"). "An operator is a component of processing functionality that takes one or more input streams as input, processes the tuples and attributes [of the input stream], and produces one or more streams as output." *Id.*, §2.2.2 at 00064. The operators are assigned to processors on host computers to carry out

their processing. *See, e.g.*, *id.*, §3.2.4 at 00117 ("By assigning the operator's threads to different processors or hosts, consecutive tuples may be processed concurrently."). Indeed, Foster discloses the typical environment to execute Streams to be "processor[s] in [] clusters" that can be managed through the monitoring functions of the Streams Runtime. *Id.*, §1.2.1 at 00043; Crovella, ¶60.

[1b]	**causing, at least in part, a marshalling via the processor of the at least one processing element as at least one data object;**

Foster renders obvious limitation [1b]. The combination with Williams further renders obvious limitation [1b]. Crovella, ¶¶62-87.

Foster teaches limitation [1b].

Foster teaches limitation [1b] by ingesting data to BigInsights, including preparing the data and transforming it to a format used by BigInsights (e.g., JSON). Crovella, ¶¶63-70. Foster teaches **causing a marshalling…of the at least one processing element as at least one data object**. For example, Foster discloses that Streams ingests and prepares the incoming data for further processing (e.g., enriching) by BigInsights. Ex-1004, §7.1.2 at 00319 ("The parallel pipeline architecture of Streams is used to batch and buffer data and, in parallel, load it into BigInsights…."). *First*, Streams marshals the input data by processing it in various methods to convert its memory representation into a form that BigInsights can use:

> Data needs to first be processed for extracting all the required data for consumption by downstream analytics. Data-preparation steps include operations such as data-cleansing, filtering, feature extraction, deduplication, and normalization.

Ex-1004, §7.1.2 at 00319; Crovella, ¶63.

Foster illustrates this process with two examples of preparing a call detail record (CDR) and preparing a social media post:

> An example of this function is clear in the Call Detail Record (CDR) processing use case. CDRs come in from the telecommunications network switches periodically as batches of files. Each of these files contains records that pertain to operations such as call initiation, and call termination for telephones. It is most efficient to *remove the duplicate records in this data as it is being ingested*, because duplicate records can be a significant fraction of the data that will needlessly consume resources if post-processed. Additionally, *telephone numbers in the CDRs need to be normalized and data appropriately prepared to be ingested into the back end for analysis*. These functions are most efficiently performed using Streams.
>
> Another example can be seen in a social media based lead-generation application. In this application, *unstructured text data from sources such as Twitter and Facebook is ingested to extract sentiment and leads of various kinds*. In this case, a lot of resource savings can be achieved if the text extraction is done on data as it is being ingested and irrelevant data such as spam is eliminated. With volumes of 140

> million tweets every day and growing, the storage requirements can add up quickly.

Ex-1004, §7.1.2 at 00340; Crovella, ¶64.

Second, Foster discloses details of marshalling data from the memory representation in Streams to the format for BigInsights. Ex-1004, §7.1.7 at 000344 ("Streams formatting operators can transform data between the Streams tuple format and data formats used by BigInsights."). Streams receives streams of data records that are represented in memory as "tuples, which are composed of a fixed set of attributes." *Id.*, §2.2.2 at 00064. These tuples are transformed, for example, to "JSON as the data format for storage in BigInsights." *Id.* "The TupleToJSON operator is used to convert the tuples in Streams to a JSON string for storage in BigInsights." *Id.* Conversely, when data is sent from BigInsights to Streams, it is marshalled again using the inverse operation. *Id.* ("The JSONToTuple operator is used to convert the JSON string read from BigInsights to a tuple for Streams to process."). Foster discloses Stream supports various other "Sink operators" to marshal data into other output formats. *Id.*, §3.1.3 at 00095-97 (e.g., data streams for Microsoft ODBC, or Apache HDFS); *see also id.*, §7.1.6 at 00343 ("The data ingest is achieved using a Streams-BigInsights Sink operator to write to BigInsights (refer to 7.1.7, "Enabling components" on page 324)."), §7.1.7 at

00346 ("Streams source and sink adapters can be used to exchange data with BigInsights."); Crovella, ¶65.

The resulting transformed output is a **data object**. As just discussed, Foster transforms the memory representation of the data into a format suitable for transmission and processing at BigInsights. It is an object representing the instance of data in a particular format (e.g., JSON). This is commensurate with the '693 patent's disclosure that "marshalling may be considered as a process for transforming memory representation of an object *to a data format…suitable for*…transmission of the data to *different components of a process and/or from one process to another process*." Ex-1001 ('693 patent), 4:29-47. A POSITA would have understood that a data object is simply a piece of information packaged in a specific manner, e.g., resulting from a marshalling process and is in a format suitable for use or transmission. Crovella, ¶66.

JSON was a well-known and common data format. In addition to being, and because it was, suitable for storage, many database and cloud computing systems supported JSON both for storage and as a working form for data objects. For interoperability purposes, it was commonplace to marshal data and information into JSON to transfer data between different systems and platforms, such as IBM's Streams/BigInsights platforms. *Supra* §I.C; Crovella, ¶67.

Foster teaches this marshalling occurs **via the processor**. Foster discloses the data-preparation and conversion steps are processed by operators on Streams. Ex-1004, §7.1.6-7 at 00343-344, §3.1.3 at 00095-97. As discussed above for limitation [1a], Streams operators execute on processors in host computers. *Id.*, §2.2.1-2.2.2 at 00060-64, §3.2.4 at 00117, §1.2.1 at 00043; *supra* §VI.A.3 (incorporated by reference); Crovella, ¶68.

A POSITA would have understood the Sink operators (including TupletoJSON) to be executed on the same processor as the data-ingestion and -preparation operators, because it was conventional to use a single processor to carry out related functions on the same data. It would have been inefficient and wasteful to transfer data to another processor in the cluster. Foster aims to "optimize throughput, latency and scalability" of the data analytics carried out by Streams applications. Ex-1004, §4.5 at 00199-201. A POSITA would have realized that unnecessarily transmitting data across different nodes could increase network load and processing times. Thus, a POSITA would have found it obvious for these related operations to be executed on the same processor. Crovella, ¶69.

Additionally, Foster discloses sending information between Streams and downstream systems over a TCP network connection. For example, Foster discloses that, for "interact[ing] with systems that consume the output of a Streams application," a Sink operator "can output a data stream over a TCP network

connection, providing a reliable communication of the stream contents to the recipient." Ex-1004, §3.1.3 at 00095. BigInsights is one such recipient of data transferred over the network from Streams. *See, e.g.*, *id.*, §7.1.1 at 00338 (data ingest), Fig. 7-1 at 00339 (same), §7.1.4 at 00341 ("new information" from Streams), §7.1.5 at 00342-343 (Streams "requests an update from BigInsights"), §7.1.6 ("trigger," "query parameters"), Fig. 7-2 at 00342 (arrows from Streams to BigInsights). A POSITA would have understood that Foster's disclosure of Streams sending information to BigInsights via, e.g., TCP connection teaches marshalling the data before it is sent over the connection because it was well-known that sending data via TCP involves marshalling the data into the TCP packet format. Crovella, ¶70.

Williams further teaches limitation [1b].

The combination with Williams teaches limitation [1b] by disclosing marshalling. Ex-1005, 00039; Crovella, ¶¶71-75. Well before the '693 patent was filed, it was customary to marshal data/information for transmission between nodes in a distributed computing environment; indeed, marshalling was a well-known and pervasive concept. *Supra* §I.C; Crovella, ¶71.

To the extent Patent Owner may argue that Foster does not explicitly disclose limitation [1b], a POSITA would have found this limitation obvious based upon the combined teachings of Foster and Williams. For example, Williams

discloses, "marshalling and unmarshalling" of data values that are "translated to and from a protocol format." *Id.*, 00039 (emphasis removed). A POSITA would have been motivated to combine this teaching with Foster's teachings as discussed further below. Crovella, ¶72.

Williams teaches **marshalling a processing element** (e.g., data, code, or both) **as a data object**. For example, Williams discloses that "[i]n distributed computing environments,[2] encodings define *how data values defined in the application can be translated to and from a protocol format. We refer to these translation steps as* serialization and deserialization, or, synonymously, *marshalling and unmarshalling*." Ex-1005, 00039 (cleaned up and emphasis added). The '693 patent shares this understanding, providing serializing and encoding as examples of marshalling. Ex-1001, 8:3-13 ("For example, in a distributed computing environment, an object is marshalled (e.g., encoded, serialized) for sending the object from a client to a server side or from one process to another, and unmarshalling is a process to decode (e.g., de-serialize) the encoded object at the receiving side."), 7:20-24 ("[M]arshalling is a process of

[2] A POSITA would have understood that Williams' reference to distributed computing environments encompasses Foster's Streams and BigInsights.

encoding an object for sharing, sending, and/or communicating the object….”). Crovella, ¶73.

Moreover, Williams discusses marshalling processing elements (e.g., code, data, or both) as data objects at least because it discusses marshalling/encoding “data values.” Ex-1005, 00039. This is commensurate with the ’693 patent’s description of data objects as being the output of a marshalling operation applied to data. Ex-1001 (’693 patent), 11:33-40, 12:40-52; Crovella, ¶74.

Thus, a POSITA would have found it obvious to take Foster’s data processed by Streams (**processing elements** discussed above for limitation [1a]) and marshal those teachings with the teachings of Williams into a portable form as **at least one data object** to be transferred to and used by other processes or applications (e.g., BigInsights). Crovella, ¶75.

Motivation to Combine

A POSITA would have found the combination obvious and would have been motivated to combine Foster and Williams for a number of reasons. Crovella, ¶¶76-87. *First*, the teachings of the references themselves strongly suggest combining their teachings. For example, Foster discloses transferring data between applications/platforms (e.g., Streams and BigInsights) in a distributed computing environment, marshalling data for transmission between applications/platforms in a distributed computing environment was customary and

well-known, and a POSITA would have understood Foster to teach and Williams to disclose marshalling data for transmission between applications/processes. *Supra* §I.C. Therefore, a POSITA would have been motivated to combine the two references. Crovella, ¶77.

Second, such a combination of prior art elements according to known methods would have yielded predictable results. Foster teaches transferring data between the Streams and BigInsights applications in a distributed computing environment. *See infra* §VI.A.3.[1c]-[1e]. A POSITA would have understood Williams to teach marshalling data for purposes of communicating/transmitting the data between different processes in a distributed computing environment (e.g., by formatting to a protocol). The combination yields the predictable results of applying Williams' teachings regarding marshalling data for transmission between two different applications/processes to Foster's teachings of transferring data between two different applications/processes (e.g., Streams and BigInsights) in order to marshal such data to put it into a format suitable for transmission, which was the known purpose of marshalling as Williams discloses. Crovella, ¶78.

Third, Foster and Williams are analogous art at least because they are applied to similar applications in a similar environment—distributed computing. Ex-1004, §1.1.4 at 00037 ("[A] Big Data platform, based on two products [that] are built to run on large-scale, *distributed* systems…."; Ex-1005, 00039 ("In

distributed computing environments, encodings define how data values defined in the application can be translated to and from a protocol format. We refer to these translation steps as serialization and deserialization, or, synonymously, marshalling and unmarshalling.") (cleaned up and emphasis added). Moreover, both references were published as part of the IBM Redbook series, which was a very popular source of information for engineers working in the field. Crovella, ¶79.

A POSITA would have looked to well-known marshalling schemes, such as Williams' teachings, to implement data transfer between Streams and BigInsights according to Foster's teachings. Both references teach transferring data between applications/processes and Williams teaches that such transferring involves marshalling the data according to well-known and long-established techniques. *Supra* §I.C. Foster teaches transferring data via protocol formats such as JSON or TCP, as explained above, and Williams teaches that marshalling involves translating data "to and from a protocol format," including encoding and serialization Ex-1004, §3.1.3 at 00095, §7.1.7 at 00344; Ex-1005, 00039. Applying Williams' marshalling teachings to Foster's transfer of data between the Streams and BigInsights applications would have yielded extremely predictable results—the data would have been marshalled for transmission. Crovella, ¶80.

Fourth, the combination would have also involved applying a known technique (marshalling) to a known device/method (distributed computing system

with dynamic and static processing mechanisms, *see supra* §I.A) ready for improvement to yield predictable results as previously discussed. Williams' marshalling was a known technique for processing data for transmission between different processes/applications. Crovella, ¶81.

A POSITA would have realized that Foster's distributed computing system was ready for improvement because Foster already contemplated formatting data for transmission to and use by downstream applications, and this formatting would have benefitted from the known implementation details regarding marshalling as taught by, for example, Williams. Implementing marshalling for Foster's transferal of data between applications using known marshalling techniques would have been straight-forward because marshalling data was ubiquitous in distributed computing and data transmission applications at the time, as Williams and other references teach. Ex-1005, §1.4.3 at 00039; *supra* §I.C; Crovella, ¶82.

Combining Foster's and Williams' teachings would not have changed either reference's principle of operation because the combination simply uses teachings from each reference according to their known principles, in the manner taught by each reference. Foster's teachings of formatting data for transmission between applications/processes, when combined with William's related teachings of marshalling in a distributed computing environment, would have still been applied in the manner taught by both references, just with straightforward and well-known

implementation details regarding marshalling disclosed by Williams. Implementing Foster's formatting and transmission teachings using Williams' marshalling teachings would have involved a simple matter of applying known marshalling techniques, which are fundamentally mathematical algorithms that process data, and as such, can be easily implemented as computer software programs running on a computer or processor(s) such as Foster's operators/processors. Crovella, ¶83.

A POSITA would have had a reasonable expectation of success when combining Foster's and Williams' teachings because they are combined in a conventional manner that does not change any principle of operation but instead simply involves tweaks or the addition of implementation details. The combination is based on compatible devices and techniques in data processing/formatting. Foster's and Williams' teachings, when combined, would each have been used for their known purposes in the conventional manner taught by each reference. A POSITA would have been more than capable of combining the two references because the combination would simply marshal data being transferred between Streams and BigInsights according to known techniques. Crovella, ¶84.

A POSITA would have been fully capable of combining Foster's and Williams' teachings, including making necessary modifications—if any—that, as

explained above, would have involved straightforward processing of information using mathematically defined algorithms and implementing processes with straightforward programming. Marshalling and the reasons for doing it were well-known in the prior art. *Supra* §I.C. Developing software and hardware for distributed computing and marshalling data was a common task for artisans in the field. A POSITA would have been able to combine and would have combined the teachings of Foster and Williams. *Supra* §IV; Crovella, ¶¶85-86.

[1c]	**transferring of the at least one data object to at least one static processing mechanism;**

Foster teaches limitation [1c] by ingesting the data prepared at Streams to BigInsights. Ex-1004, §7.1.1 at 00339; Crovella, ¶¶88-93. Foster teaches **transferring of the at least one data object to at least one static processing mechanism**. For example, Foster discloses that the ingested data marshalled into a data object (e.g., in the JSON or TCP formats) is transferred from Streams to BigInsights to further process the data. Ex-1004, §7.1.1 at 00339 ("Historical context generated from BigInsights to bootstrap analytics and enrich incoming data on Streams…Models generated by analytics such as data-mining, machine-learning, or statistical-modeling on BigInsights"). This includes the ingested data from Foster's examples of CDR and social media feed that were marshalled at Streams, as discussed above for limitation [1b]. *Supra* §VI.A.3.[1b]. Figure 7-1

illustrates this "data flow" from Streams to BigInsights. Ex-1004, §7.1.4 at 00341, Fig. 7-1 at 00339 (Data Ingest arrow from Streams to BigInsights):

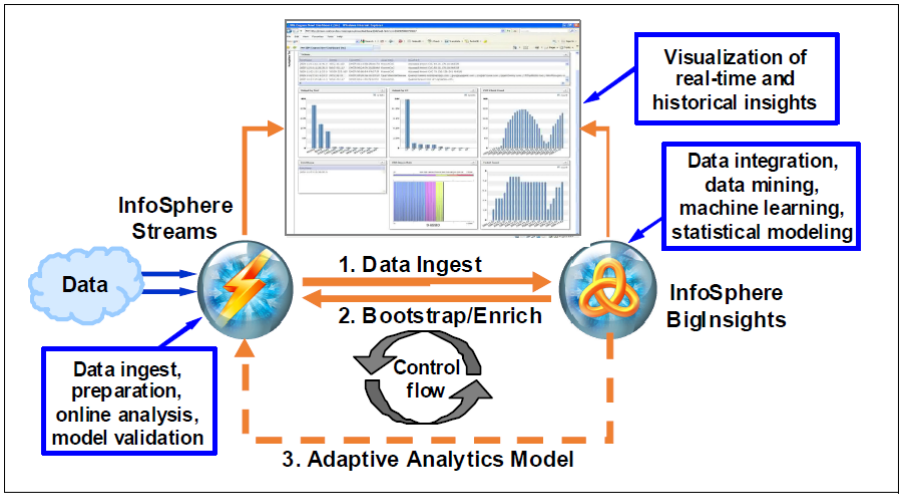


Figure 7-1 Big Data application scenarios

See also id., §7.1.1 at 00338 ("[c]ontinuous ingest of data through Streams into BigInsights."). Thus, a POSITA would have understood these flows to indicate transfer of the ingested data to BigInsights. Crovella, ¶89.

The transferred data object is then processed or stored for later processing at BigInsights. Ex-1004, §7.1.5 at 00342 ("The input streams are also ingested into BigInsights for historical storage and deep analysis"), §7.1.2 at 00339 ("Data is then stored in BigInsights for deep analysis…."). Foster thus confirms the transfer from Streams to BigInsights, because data arriving at Streams would have to be transferred to BigInsights if it were to be stored or processed by BigInsights. Crovella, ¶90.

BigInsights, including its deep analysis functions, is a **static processing mechanism**, because Foster teaches it is a mechanism for processing historical or slow-moving data. *See supra* §V.C. For example, Foster discloses that BigInsights is a "data-at-rest [] platform[]." Ex-1004, §7.1.1 at 00338. BigInsights generates "[h]istorical context" and carries out analytics such as "data-mining, machine-learning, or statistical-modeling." *Id.*, §7.1.1 at 00339, Fig. 7-1 at 00339 ("Data integration, data mining, machine learning, statistical modeling"). "BigInsights can be used to analyze data over a large time window, which it has assimilated and integrated from various continuous and static data sources." *Id.*, §7.1.3 at 00340; Crovella, ¶91.

Data-mining, machine-learning, and statistical-modeling are examples of data processing techniques that are typically carried out on large sets of historical or slow-moving data. Indeed, Foster discloses that BigInsights does the same:

> Deep analysis is performed using BigInsights to detect patterns on data collected over a long period of time. Statistical analysis algorithms or machine-learning algorithms are compute-intensive and run on the entire historical data set, in many cases making multiple passes over the data set, to generate models to represent the observations.

Ex-1004, §7.1.4 at 00341. Among other techniques, BigInsights uses the "MapReduce framework." *Id.*, §7.1 at 00338. MapReduce was a well-known

process carried out in batches—that is, slow-moving data that is collected until it is "periodically" processed. *Supra* §I.B. The '693 patent acknowledges that "map-reduce batch jobs" are an example of "static processing." Ex-1001, 5:4-11; Crovella, ¶92.

[1d]	**processing of the at least one data object by the at least one static processing mechanism; and**

Foster alone, and combined with Williams, teaches PEs (data) from input streams are determined and marshalled as data objects (ingested, prepared, and transformed) and transferred to BigInsights. *Supra* §VI.A.3.[1a]-[1c]. Foster teaches limitation [1d] by bootstrapping/enriching data at BigInsights, and the combination with Williams teaches this limitation for the same reasons. Ex-1004, §7.1.3 at 00340; Crovella, ¶¶94-99.

Foster teaches **processing of the at least one data object by the at least one static processing mechanism**. For example, BigInsights bootstraps and enriches the incoming data. Ex-1004, §7.1.3 at 00340. Foster explains how BigInsights uses historical data it has accumulated and analyzed to "bootstrap [data] to a well-known state." *Id.* The results from analyzing historical data "are also used to enrich incoming data with additional attributes required for downstream analytics." *Id.* As discussed above for limitations [1b]-[1c], Foster provided examples of marshalling CDR and social media feed into data objects

that were transferred to BigInsights. *Supra* §VI.A.3.[1b]-[1c]. Following up on these examples, Foster explains that BigInsights further processes these data objects:

> As an example from the CDR processing use case, an incoming CDR may only list the phone number to which that record pertains. However, a downstream analytic may want access to all phone numbers a person has ever used. *At this point, attributes from historical data are used to enrich the incoming data to provide all the phone numbers. Similarly, deep analysis results in information about the likelihood that this person may cancel their service.* Having this information enables an analytic to offer a promotion online to keep the customer from leaving the network.
>
> In the example of the social media based application, an incoming Twitter message only has the ID of the person posting the message. However, *historical data can augment that information with attributes such as influencer*, giving an opportunity for a downstream analytic to treat the sentiment expressed by this user appropriately.

Ex-1004, §7.1.3 at 00340; Crovella, ¶95.

Additionally, as an alternative and independent basis, Foster teaches this limitation with its disclosure of BigInsights' processing the transferred PE data object to generate and/or update an "adaptive analytics model" that provides the control flow for Streams' ingestion of data. Ex-1004, §7.1.1 at 00339 ("Adaptive analytics model: Models generated by analytics such as data-mining, machine-

learning, or statistical-modeling on BigInsights used as basis for analytics on incoming data on Streams and updated based on real-time observations.");

Crovella, ¶96:

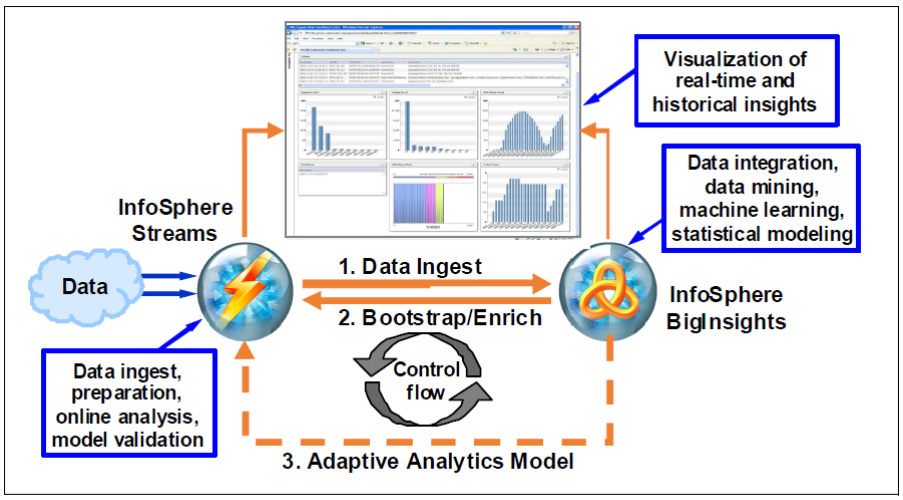


Figure 7-1 Big Data application scenarios

Ex-1004, Fig. 7-1 ("3. Adaptive Analytics Model").

Foster further explains how BigInsights updates the model with the transferred data object, e.g., "new information of interest [] identified" by and transferred from Streams. Ex-1004, §7.1.4 at 00341 ("Where entirely new information of interest is identified [by Streams], the deep analysis [at BigInsights] may be targeted to just update the model in relation to that new information[.]"). Specifically, "BigInsights does deep analysis using the same sentiment extraction analytic as used in Streams and creates a results file to update the Streams model." *Id.*, 7.1.6 at 343. Foster illustrates parts of this process in relation to a limited example that monitors for positive/negative sentiment of social media feed.

BigInsights "queries all of its data using the same sentiment analytics used in Streams and recalculates the list of known causes" that Stream uses to monitor the incoming social media feed. *Id.*, §7.1.5; Crovella, ¶97:

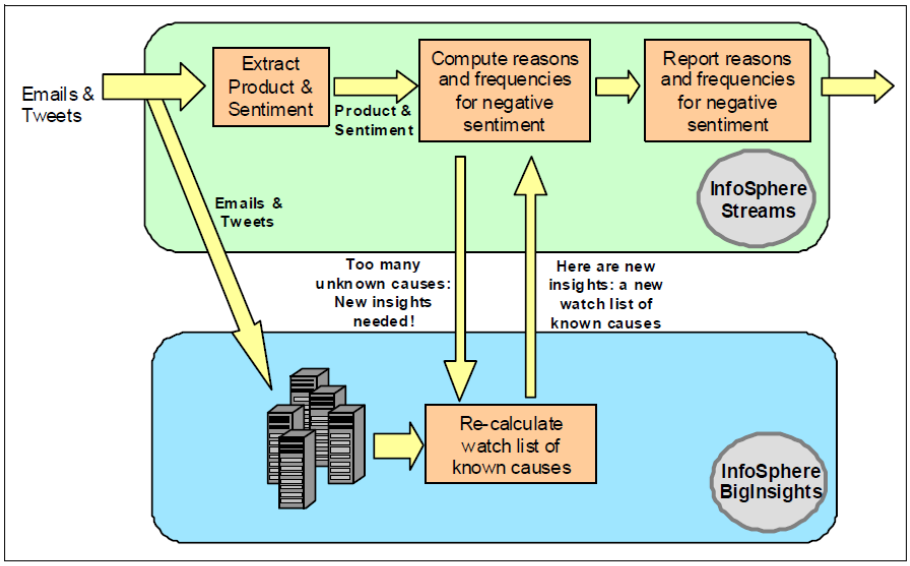


Figure 7-2 Streams and BigInsights application example

Ex-1004, Fig. 7-2 ("Re-calculate watch list of known causes").

BigInsights is a **static processing mechanism**. *Supra* §VI.A.3.[1c] (analysis incorporated here by reference); Crovella, ¶98.

[1e]	**after processing, transferring the at least one data object back to the at least one dynamic processing mechanism.**

Foster alone, and combined with Williams, teaches PEs (data) from input streams are determined, marshalled as data objects (ingested, prepared, and transformed), transferred to BigInsights and is then processed by BigInsights. *Supra* §VI.A.3.[1a]-[1d]. Foster teaches limitation [1e] by transferring data bootstrapped/enriched at BigInsights back to Streams, and the combination with

Williams teaches this limitation for the same reasons. Ex-1004, §7.1.2-3 at 000339-340; Crovella, ¶100-105.

Foster teaches **after processing** at BigInsights (the static processing mechanism), **transferring the at least one data object back to the at least one dynamic processing mechanism**. For example, Foster discloses the bootstrapped or enriched data is used in "downstream analytics." Ex-1004, §7.1.3 at 000340. Such downstream analytics include "downstream analytics on Streams." *Id.*, §7.1.2 at 00339. Figure 7-1 confirms that the bootstrapped/enriched data is transferred back to Streams with its arrow from BigInsights to Streams showing the data flow for "2. Bootstrap/Enrich." Crovella, ¶102:

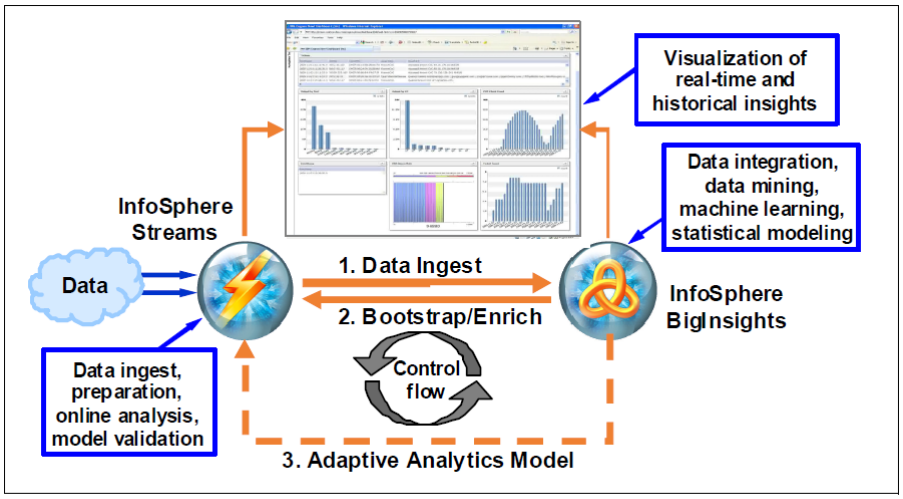


Figure 7-1 Big Data application scenarios

Ex-1004, Fig 7-1 at 00339.

For the CDR processing example, Foster explains that "a downstream analytic may want access to all phone numbers a person has ever used" or that

BigInsights generated "information about the likelihood that this person may cancel their service… enables an analytic to offer a promotion…." Ex-1004, §7.1.3 at 00340. For the social media example, BigInsight can "can augment that information with attributes such as influencer, giving an opportunity for a downstream analytic to treat the sentiment expressed by this user appropriately." *Id.*; Crovella, ¶103.

Streams is a **dynamic processing mechanism**. *Supra* §VI.A.3.[1a] (analysis incorporated here by reference); Crovella, ¶104.

2. Independent Claim 11

Claim 1 is directed to a method with certain steps performed by processors and processing mechanisms. Claim 11 is directed to an apparatus that includes a processor and a memory that contains computer program code configured to perform the steps of claim 1's method. Crovella, ¶106. Dr. Crovella provides a side-by-side claim comparison demonstrating the substance of the independent claims are substantially similar. Crovella, ¶107. Foster discloses a distributed computing system operating host computers, that a POSITA would have understood to include a processor and memory storing computer program code, which is consistent with well-known use of such components. Crovella, ¶108.

[11pre] An apparatus comprising:

The preamble is taught by Foster's disclosures of distributed computing systems executing Streams and BigInsights, and the combination with Williams teaches the preamble for the same reasons. Crovella, ¶¶110-112. Crovella, ¶¶110-112. For example, Foster discloses that Streams and BigInsights are part of IBM's BigData platform and operate on apparatuses (e.g., distributed computing systems). Ex-1004, §1.1.4 at 00037 ("Both products are built to run on large-scale distributed systems, designed to scale from small to very large data volumes, handling both structured and unstructured data analysis."), §7.1 at 00338 (same). Foster provides an example architecture with distributed hardware elements represented as cubes. Crovella, ¶111:

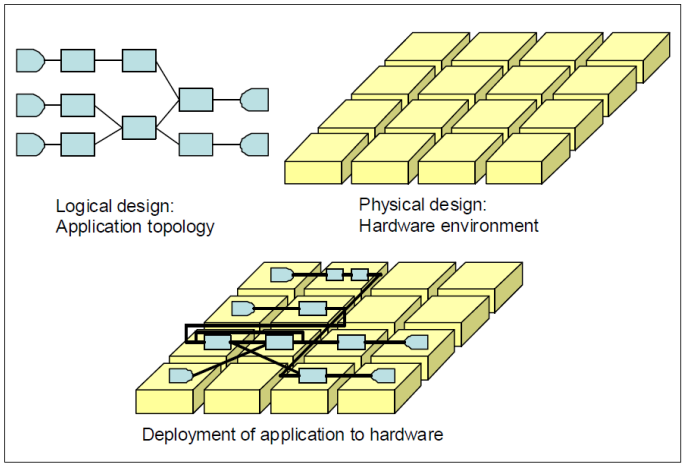


Figure 3-1 Application design, physical design, and deployment

Ex-1004, Fig. 3-1 at 00090 ("Hardware environment"). The apparatus at least comprises the elements of limitation [11a] and performs the steps discussed for limitations [11b]-[11f]. Crovella, ¶112.

[11a]	**at least one processor;** **and at least one memory including computer program code for one or more programs,** **the at least one memory and the computer program code configured to, with the at least one processor, cause the apparatus to perform at least the following,**

Foster teaches this limitation by its disclosures of host computers for operating Streams and BigInsights. Crovella, ¶¶113-118. *First*, Foster teaches **at least one processor**. For example, Foster discloses that Streams executes on host computers in data processing clusters (Ex-1004, §2.2.1 at 00060-64, §1.2.1 at 00043), and that such computers included processors (*id.*, Fig. 1-8 at 00040), as discussed above for limitation [1a]. *Supra* §VI.A.3.[1a]; Crovella, ¶114.

Second, Foster teaches **at least one memory including computer program code for one or more programs**. For example, Foster discloses that Stream is a "software product." Ex-1004, §2.2 at 00059, 00390 ("Streams software distribution file"). And "[a]ll software environments are composed of three major components: process, *memory*, and disk." *Id.*, §2.1 at 00059, §5.6.2 at 00287 ("A Streams Instance refers to the collection of *memory*, process, and disk that is the Streams server proper."). The Streams runtime software can, for example, be installed on a "32-bit RHEL 5.5 VMWare image configured with *2 GB of RAM* and hosted by a PC running Microsoft Windows XP." *Id.*, 00391; Crovella, ¶115.

The memory of Foster's host computers and the programs in them, such as the Streams runtime and applications, include **computer program code**. Foster discloses that Streams applications are written as "[s]ource files [that] are compiled into binary code files." Ex-1004, §5.1.1 at 00226, §5.1.3 at 00231 ("the Streams application code"), §5.3 at 00268 ("Streams code"), §6.4.2 at 00270 (Streams source file includes *code*). For example, Foster provides the source code and details of an example streams program that calls for "compil[ing] and run[ning] the software." *Id.*, §5.5 at 00279. A POSITA would have understood that the compiled computer program code would have been stored in the host computer's memory **for one or more programs** constituting Streams application, as code is stored in memory in order to be executed and programs consist of code. *See also, e.g.*, *id.*, §6 at 00297-336 ("Advanced Streams programming"); Crovella, ¶116.

These teachings equally apply to Foster's teachings of BigInsights, which Foster explains is designed to run on similar "large-scale distributed systems." Ex-1004, §1.1.4 at 00037. A POSITA would have understood these disclosures to teach that the host computers hosting BigInsights also include a **processor** and **memory including computer program code** for the BigInsights programs. Using such memory and code was a conventional and common approach to implement data processing in general, and distributed computing functionality in particular,

and a POSITA would have understood Foster to teach these features. Crovella, ¶117.

Third, Foster teaches **the at least one memory and the computer program code [is] configured to, with the at least one processor, cause the apparatus to perform at least the following** steps claimed in limitations [11b]-[11f], because Foster, including its teachings of Streams and Insights causing the execution of processing on distributed computing systems, teaches those limitations, as discussed below and the corresponding limitations [1a]-[1e], as discussed above. *Infra* §VI.A.4.[11b]-[11f], §VI.A.3.[1a]-[1e]; Crovella, ¶118.

[11b]	**determine at least one processing element of at least one dynamic processing mechanism;**
[11c]	**cause a marshalling via the processor of the at least one processing element as at least one data object;**
[11d]	**transfer of the at least one data object to at least one static processing mechanism;**
[11e]	**process the at least one data object by the at least one static processing mechanism; and**
[11f]	**after the process, transfer the at least one data object back to the at least one dynamic processing mechanism.**

Foster alone, and Foster combined with Williams teaches these limitations. As discussed above, Foster alone, and Foster combined with Williams teach the shared substantive limitation of [1a]-[1e] (*supra* §VI.A.4, incorporated by reference) that correspond to limitations [11b]-[11f]. A POSITA would have been

motivated to combine Foster and Williams for the same reasons discussed above for limitation [1b]. *Supra* §VI.A.1.[1b] (motivation to combine). Crovella, ¶119.

3. Dependent Claims 2 and 12

Claims 1/11 were discussed above. The additional limitations of claims 2/12 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶120-138.

2[a].	**The method of claim 1, further comprising: a processing of at least substantially real-time data via the at least one processing element to determine one or more triggering events,**
12[a].	**The apparatus of claim 11, wherein the apparatus is further caused to: process and/or facilitate a processing of at least substantially real-time…,**

Foster teaches the additional limitation of 2[a]/12[a] in two ways, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶121-127. *First,* Foster teaches this limitation by triggering data to be further processed or not to be processed at BigInsights. Ex-1004, §7.1.4 at 00341; Crovella, ¶¶121-124. Foster combined with Williams teaches processing substantially real-time data (e.g, the incoming data stream) at Streams, including determining processing elements (e.g., selecting such data) for further processing (e.g., bootstrapping/enrichment) at BigInsights. *Supra* §VI.A.3.[1a]-[1b]; Crovella, ¶122.

During ingest, "Streams appl[ies] the [adaptive analytics] model on the incoming data" to identify relevant or remarkable input data, among other things. Ex-1004, §7.1.4 at 00341. Discussing its social media feed example, Foster discloses that during ingest, "irrelevant data such as spam is eliminated." *Id.*, §7.1.2 at 00340. One way Streams monitors the incoming data is whether it has been posted by a "key influencer." *Id.*, §7.1.4 at 00341; Crovella, ¶123.

Foster teaches these are **triggering events** because these events (e.g., detection of an influencer's post, absence of relevant information, and a request for further processing) trigger further processing at BigInsights, rather than discarding the data as irrelevant. Foster's triggering applies to both the bootstrapping/enrichment during data ingest and model updating teachings, because Foster explains that Streams "triggered deep analysis in BigInsights using the same Streams-BigInsights Sink operator as mentioned in 'Data ingest into BigInsights from Streams.'" Ex-1004, §7.1.6 at 00343; Crovella, ¶124.

Second, as an alternative and independent basis, in Foster's positive/negative sentiment example discussed above for limitation [1d], Foster teaches limitation 12[a] by disclosing another model that Stream uses to monitor for "negative sentiment" about a product. Ex-1004, §7.1.5 at 00342-343 (by using "the list of causes to be monitored in real time"); Crovella, ¶¶125-126:

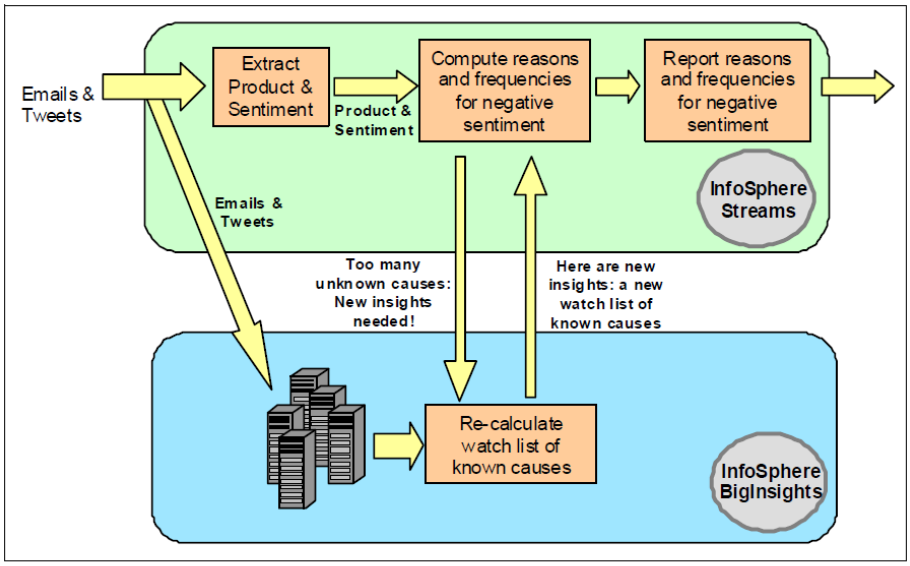


Figure 7-2 Streams and BigInsights application example

Ex-1004, Fig. 7-2 at 000342.

Among the non-discarded, relevant data selected as processing elements, Foster further processes the real-time data to determine the emergence of new trends as a **triggering event** to update the adaptive analytics model. *Id.*, §7.1.4 at 00341 ("If the new observations deviate significantly from the expected behavior, the online analytic on *Streams may determine that it is time to trigger a new model-building process* on BigInsights" and "[w]here entirely new information of interest is identified, the deep analysis may be targeted to just update the model in relation to that new information"). Foster discloses an example where there is a change in influencer relationships. *Id.*, §7.1.4 at 00341 (a "key influencer identified in the model may no longer be influencing others or an entirely new influencer or relationship can be identified"). Another example is if there are "[t]oo

many unknown causes" of negative sentiment. *Id.*, §7.1.5 at 00342; Crovella, ¶126.

[2b]	**wherein, the transferring of the at least one data object to the at least one static processing mechanism is based, at least in part, on the one or more triggering events, and**
[12b]	**wherein the transfer of the at least one data object to the at least one static processing mechanism based, at least in part, on the one or more triggering events, and**

Foster teaches this limitation in two ways, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶128-132. *First*, Foster teaches **the transferring of the data object to the static processing mechanism is based, in part, on the triggering events**. For example, Foster discloses that incoming data that is not identified by a triggering event at Streams for further processing at ingest is discarded and thus, not transferred to BigInsights. Ex-1004, §7.1.2 at 00340 ("[I]rrelevant data such as spam is eliminated."). The data that is relevant is ingested, marshalled, and transferred to BigInsights for bootstrap/enrichment, as discussed above for limitations [1b]-[1d]. *Id.*, §7.1.1 at 000339-340; *supra* §VI.A.3.[1b]-[1d]; Crovella, ¶129.

Foster's explanation of BigInsights' model updating based on the triggering event of new information of interest (e.g., a new influencer) further illustrates how transferring is based on the triggering events. Foster explains that "the deep analysis may be targeted to just update the model in relation to that *new*

information." Ex-1004, §7.1.4 at 00341. In order for BigInsights "to look for all historical context for this new influencer, where the raw data had [already] been stored in BigInsights" (*id.*), Streams would have had to transfer the new information to BigInsights, marshalled into a data object of the right format. A POSITA would have understood the transfer was based on the triggering event, because it occurred in response to the triggering event and it was intended "to trigger a new model-building process on BigInsights" (the resulting processing). *Id.*; Crovella, ¶130.

Second, as an alternative and independent basis, Foster teaches this limitation with its positive/negative sentiment monitoring example where the transferring is based on the triggering event of identifying a significant "percentage of negative sentiment that have an unknown cause." Ex-1004, §7.1.5 at 00342-343. The teachings that were just discussed above also apply to this alternative teaching. As Foster explains, the triggering of the deep analysis is transferred to BigInsights in the same way that data is ingested into BigInsights by Stream. *Id.*, §7.1.6 at 00343 ("BigInsights use[s] the same Streams-BigInsights Sink operator as mentioned in 'Data ingest into BigInsights from Streams'"). It would have been obvious for Streams to transfer the data object in this way (at least including the most recent negative item that tripped the limit and the trigger to commence deep analysis), because 1) it would have been more efficient to send the two as a single

data object instead of separately; and 2) the negative item would need to be included in BigInsight's deep analysis to update the watch list of known causes. Crovella, ¶131.

[2c]	**wherein the one or more triggering events include, at least in part, one or more contextual trends occurring above at least one threshold level.**
[12c]	**wherein the one or more triggering events include, at least in part, one or more contextual trends occurring above at least one threshold level.**

Foster teaches this limitation in two ways, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶133-138. As discussed for limitations [2a] and [2b], Foster teaches determining triggering events and transferring data to BigInsights (a static processing mechanism) based on triggering events such as including relevant information, identifying a deviation in expected behavior, or identifying new information of interest. Alternatively Foster teaches an example with a triggering event is an accumulation of unknown causes, as discussed above for limitations [2a]-[2b]. Crovella, ¶133.

First, Foster teaches that **the one or more triggering events include, in part, one or more contextual trends occurring above at least one threshold level**. For example, Foster discloses that BigInsight's model generating/updating is triggered by trends in the context of an e-mail's or tweet's relation to influencers. Ex-1004, §7.1.4 at 00341 ("a key influencer identified in the model

may no longer be influencing others or an entirely new influencer or relationship may be identified"). Crovella, ¶134.

This is similar to the '693 patent's example of "a contextual trend associated with, for example, *one or more users*, locations, events, topics, and the like." Ex-1001, 8:14-34. Key influencers are users, and they are also often the topic of social media. Their presence, absence, or relation to incoming data entries (e.g., e-mail or tweets) are part of the context for the social media feed. Crovella, ¶135.

The fact that an influencer is no longer influencing others, or the emergence of a new influencer are trends (e.g., movements in data) of the context. As the '693 patent states, "a trend may indicate a movement." Ex-1001, 8:14-34. Foster's triggering reflects the trend **occurring above a threshold level** of social media influence. For example, "an entirely new influencer or relationship" identifies a contextual trend occurring above a threshold level of influence (e.g., accruing a certain number of followers). Similarly, an influencer no longer influencing others identifies a trend occurring above a threshold level of losing influence. These examples demonstrate Foster's disclosure of triggering events including contextual trends occurring above a threshold level (e.g., "new observations deviate significantly from the expected behavior" or when "the real-world has deviated sufficiently from the model's prediction."). Ex-1004, §7.1.4 at 00341; Crovella, ¶136.

Second, as an alternative and independent basis, for Foster's positive/negative sentiment monitoring example, Foster teaches this limitation with a triggering event that is based on the contextual trend of the number of negative sentiments with unknown cause occurring above "a significant percentage." Ex-1004, §7.1.5 at 00342. This example likewise discloses a threshold level, which is set at a numeric percentage. Crovella, ¶137.

4. Dependent Claims 3 and 13

Claims 2/12 were discussed above. The additional limitations of claims 3/13 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶139-144.

3.	**The method of claim 2, further comprising: unmarshalling of the at least one processing element as the at least one data object.**
13.	**The apparatus of claim 12, wherein the apparatus is further caused to: cause, at least in part, an unmarshalling….**

Foster teaches the additional limitation, and the combination with Williams further teaches this limitation. Crovella, ¶¶140-144. As explained above for limitation [1b], Foster teaches marshalling data (e.g., into the TCP packet format) at Streams in order to transfer the data for use at BigInsights. Because the data has been transformed into a certain protocol format, a POSITA would have understood Foster to teach BigInsights unmarshalling the data object (undoing the transform) in order to access the data inside for processing. *Supra* §I.C. A POSITA would

also have understood from Foster that the corresponding marshalling and unmarshalling processes are applied at both Streams and BigInsights when data is transferred between the two in either direction, in order to prepare the data for transmission or for processing after transmission. *Supra* §I.C. A POSITA would have found this limitation obvious in light of Foster's teachings, because it was customary to marshal and unmarshall data before the '693 patent was filed. *See supra* §VI.A.3.[1b]; *supra* §I.C; Crovella, ¶141.

A POSITA would have found this limitation further obvious based upon the teachings of Williams. For example, Williams discloses marshalling a processing element (e.g., data) as a data object (as discussed above for limitation [1b], *supra* §VI.A.3.[1b]), and Williams further discloses unmarshalling processing elements from the data objects.[3] In particular, Williams discloses "*serialization* and *deserialization*, or, synonymously, *marshalling* and *unmarshalling*." Ex-1005,

[3] A POSITA would have understood the claim to require unmarshalling a processing element from a data object, because unmarshalling is the inverse process of marshalling. The '693 patent's disclosures related to unmarshalling are consistent with this understanding. Ex-1001, 8:3-13 ("unmarshalling process is to decode the object that was marshalled," "an unmarshalling is a process to decode (e.g., de-serialize) the encoded object at the receiving side"), 14:12-22 (same).

00039 (emphasis original). A POSITA would have understood that a data object resulting from marshalling a processing element is unmarshalled, as taught by Williams, and the result is a processing element. *Supra* §VI.A.3.[1b]; Crovella, ¶142.

A POSITA would have been motivated to combine these related teachings of Foster and Williams. *See supra* §VI.A.3.[1b] (motivation to combine, incorporated here by reference). The same reasons as for limitation [1b] apply here because Foster's and William's teachings of unmarshalling are the flip side of the marshalling teachings. Crovella, ¶143.

5. Dependent Claims 4 and 14

Claims 2/12 were discussed above. The additional limitations of claims 4 and 14 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶145-152.

4.	**The method of claim 2, wherein the processing of the at least one data object via the at least one static processing mechanism causes, at least in part, an updating of the at least one data object based, at least in part, on the one or more triggering events, the one or more contextual trends, or a combination thereof.**
14.	**The apparatus of claim 12, wherein the process of the at least one data object….**

Foster teaches the additional limitation in two ways, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶146-152. *First*, Foster teaches **the processing of the at least one data object via the at**

least one static processing mechanism, by Foster's disclosures of BigInsight's bootstrapping/enrichment data received from Streams, as discussed above for limitation [1d]. *Supra* §VI.A.3.[1d]; Crovella, ¶147.

Foster further teaches **the processing…causes an updating of the data object based, at least in part, on the triggering event**. For example, as discussed above for claim 2, Foster discloses that BigInsights bootstraps/enriches the data, updating it with relevant historical information, received from Streams in response to the triggering events of 1) the data being found to be relevant (e.g., not discarded); and 2) Streams requesting enrichment or bootstrapping to Streams. Ex-1004, §7.1.4 at 00341, §7.1.2 at 00340; *supra* §§VI.A.5.2[a]-[2b]; Crovella, ¶148.

Second, as an alternative and independent basis, Foster additionally teaches **the processing of the at least one data object via the at least one static processing mechanism** by processing the ingested data object to generate/update an adaptive analytics model that Streams uses as part of its ingestion process, discussing a limited example to monitor positive/negative sentiment in incoming social media feed, as discussed above for limitation [1d]. *Supra* §VI.A.3.[1d]; Crovella, ¶149.

This alternatively and additionally teaches **the processing…causes an updating of the data object based, at least in part, on the triggering event**. For example, Foster teaches building/updating a model at BigInsights in relation to

information processed at and transferred to BigInsights from Streams, when such information triggers building/updating an adaptive analytics model in response to triggering events. Ex-1004, §7.1.4-6 at 00341-343; *supra* §§VI.A.5.2[a]-[2b]. BigInsights' triggering events are based on contextual trends. *Supra* §VI.A.5.[2c]. Therefore, the updating is also **based in part on contextual trends**. Crovella, ¶150.

Foster discloses that the information (e.g., "new information" or "new observations") is transferred to BigInsights after being marshalled and then incorporated into an existing or new model at BigInsights, so the model will be updated "in relation to that new information." Ex-1004, §7.1.4 at 00341, Fig. 7-1 at 00339; *supra* §VI.A.3.[1d], §VI.A.5.2[a]-[2b]. The data object is **updated**, because it is incorporated into an adaptive analytics model that combines the data object with other historic information. Crovella, ¶151.

6. Dependent Claims 5 and 15

Claims 4 and 14 were discussed above. The additional limitations of claims 5 and 15 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶153-161.

5[a].	**The method of claim 4, further comprising: at least one determining of one or more computational chains, one or more parameters of the one or more computational chains, one or more data elements of the one or more computational chains, or a combination thereof associated with the at least one processing element,**
15[a].	**The apparatus of claim 14, wherein the apparatus is further caused to: determine one or more computational…,**

Foster teaches the additional limitation of 5[a]/15[a] in two ways, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶154-158. Foster discloses that Streams uses models and triggers BigInsights to build and update models based on data processed at and received from Streams. Ex-1004, §7.1.4 at 00341; *Supra* §VI.A.3.[1d], §VI.A.5.2[a]-[2c], §VI.A.7; Crovella, ¶154.

First, Foster teaches **at least one determining of one or more computational chains** and **at least one determining of…one or more parameters of the computational chains**. For example, Foster discloses that Streams includes "query parameters to tailor the deep analysis in BigInsights" while triggering model building/updating. Ex-1004, §7.1.6 at 00343. The trigger specifies the steps of the computational chain to be carried out. For the negative sentiment monitoring example, Foster discloses that Streams triggers BigInsights to carry out deep analysis with a "sentiment extraction analytic" function. *Id.* The "query parameters" specify the parameters of the computational steps to be used by BigInsight for deep analysis. *Id.* ("For more advanced scenarios, the

trigger…contain[s] query parameters to tailor the deep analysis"). Thus, the parameters determine (e.g., tailor) how the computation is to be carried out. Crovella, ¶155.

Second, as an alternative and independent basis, Foster teaches **determining of one or more computational chains** by building/updating adaptive analytics models. The models are computational chains, at least because using the models involves several sequential steps of computing that are used by Streams to process incoming data. For example, a model makes a prediction by applying an algorithm to a set of stored data or stored parameters. Multiple parameters and data items/elements are accessed by the model, leading to a series of steps that constitutes a chain of computations. Crovella, ¶156.

In both cases, Foster teaches the determining, the computational chains, and the parameters are **associated with the at least one processing element**. As discussed above for claim 2, the building/updating of models is triggered by Streams' analysis or processing of the processing element (incoming data). *Supra* §VI.A.5.2[a]-[2c]. Therefore, both the trigger and the model are associated with the processing element. Crovella, ¶157.

[5b]	**wherein at least one data object includes, at least in part, the one or more computational chains, the one or more parameters, the one or more data elements, or a combination thereof.**
[15b]	**wherein at least one data object includes, at least in part, the one or more computational chains….**

Foster teaches the additional limitation of [5b]/[15b], and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶159-161. As discussed above for limitations 5[a] and 15[a], Foster discloses its triggers include computational chains and/or parameters (e.g., parameters to tailor the deep analysis). *Supra* §VI.A.8.5[a]/15[a]. As discussed above for limitations [2b] and [12b], Foster teaches transferring the data object and including the trigger for model building/updating deep analysis. Ex-1004, §7.1.6 at 00342-343; *supra* §VI.A.5.[2b]/[12b]. Similar reasons of efficiency would apply here and are incorporated by reference, because the trigger, the query parameters, and the data elements are all related information that are used together by BigInsights. A POSITA would have found it obvious to marshal these together so that the **data object includes, at least in part, the computational chains, the parameters, the data elements, or a combination thereof.** Crovella, ¶160.

7. Dependent Claims 7 and 17

Claims 1/11 were discussed above. The additional limitations of claims 7 and 17 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶162-166.

7.	**The method of claim 1, wherein the processing of the at least one data object causes, at least in part, updating of one or more components of the at least one static processing mechanism.**
17.	**The apparatus of claim 11, wherein the process of the at least one data object….**

Foster teaches the additional limitation, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶163-166. As discussed above for limitation [1d], Foster teaches processing new information in data objects received from Streams to update models generated by BigInsights. Ex-1004, §7.1.4 at 00341; *supra* §VI.A.3.[1d]. Crovella, ¶164.

Foster further teaches the processing **causes, at least in part, updating of one or more components of the at least one static processing mechanism**. For example, Foster discloses that BigInsights' "deep analysis may be targeted to just update the model in relation to [the] new information." Ex-1004, §7.1.4 at 00341. The model is a component of BigInsights, maintained in BigInsights, and used to generate and provide a new model for Streams. *See* Ex-1004, Fig. 7-2:

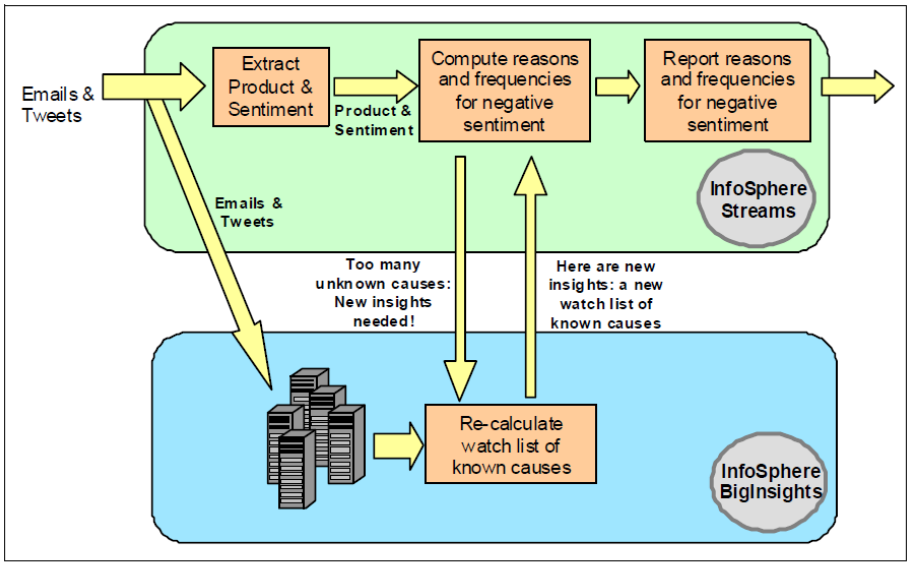


Figure 7-2 Streams and BigInsights application example

Because the updated model is used again the next time a request to update the model is triggered, a POSITA would have understood BigInsights to maintain a local copy, update it as requested, and provide the updated model to Streams. Crovella, ¶165.

8. Dependent Claims 8 and 18

Claims 1/11 were discussed above. The additional limitations of claims 8 and 18 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶167-174.

8[a].	**The method of claim 1, further comprising aggregating of one or more outputs of the at least one dynamic processing mechanism and the one or more other outputs of the at least one static processing mechanism; and**
18[a].	**The apparatus of claim 11, wherein the apparatus is further caused to: cause, at least in part, an aggregation of one or more outputs…; and**

Foster teaches the additional limitation of 8[a]/18[a], and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶168-171. For example, Foster discloses in Figure 7-1 that the outputs of Streams (the dynamic processing mechanism) and BigInsights (the static processing mechanism) are aggregated and displayed together at a user interface in various graphs for "[v]isualization of real-time and historical insights." Ex-1004, Fig. 7-1 at 00339:

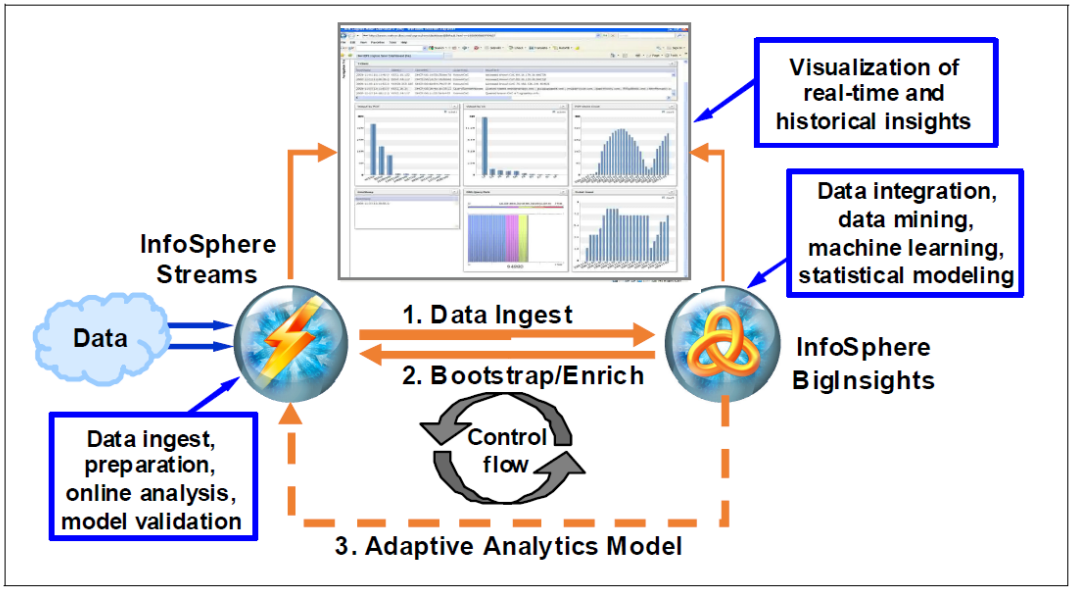


Figure 7-1 Big Data application scenarios

Crovella, ¶169.

For example, the visualized data includes the enriched data discussed above for limitation [1d]. Supra ¶V.A.3.[1d]. The enriched data aggregates the analysis of both Streams and Insights. Ex-1004, §7.1.3 at 00340 ("attributes from historical data are used to enrich the incoming data" and "historical data can augment that information"); Crovella, ¶170.

[8b]	**at least one determination of one or more outputs of at least one processing job based, at least in part, on the aggregating.**
[18b]	**determining one or more outputs….**

Foster teaches the additional limitation, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶172-174. For example, Figure 7-1 shows the result of aggregating the analyzed data from Streams and BigInsights to output visual elements to a user interface or display. Thus, they are the **determination of the outputs of at least one processing job** to "[v]isualize [] real-time and historical insights." Ex-1004, Fig. 7-1 at 00339. The processing for aggregation and visualization are a **processing job**, at least because it involves processors executing functions on data. As the arrows in Figure 7-1 illustrate, the visualization is based on the aggregating that was just discussed above for limitation 8[a]. Foster's teaching is commensurate with the '693 patent's example of "determining one or more outputs of at least one processing job based, at least in part, on the aggregation," where "a processing mechanism element and/or a service

provider may utilize an aggregated output to provide an output to another element and/or another service provider." Ex-1001, ¶15:41-52. Like the '693 patent's example, Foster's platform outputs the aggregated analytics to another element that processes the data for display. Crovella, ¶173.

9. Dependent Claims 9 and 19

Claims 1/11 were discussed above. The additional limitations of claims 9 and 19 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶175-185.

9[a].	**The method of claim 1, wherein the at least one static processing mechanism operates, at least in part, in a batch mode with a predetermined batch frequency; and**
19[a].	**The apparatus of claim 11, wherein the at least one static…; and**

Foster teaches the additional limitation of 9[a]/19[a], and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶176-180. Foster teaches the **static processing mechanism operates…in a batch mode**. For example, Foster discloses that BigInsights processes batched data from Streams. Ex-1004, §7.1.2 at 00339 ("Streams is used to *batch* and buffer data and, in parallel, load it into BigInsights for best performance."). For example, CDR records are received "periodically as batches of files." Ex-1004, §7.1.2 at 00340. Batched data is also stored at BigInsights. Id., §7.1.6 at 00343 ("Sink operator batches the data stream into configurable sized chunks for efficient storage in

BigInsights."). It would have been obvious for BigInsights to operate in a batch mode to process such data in batches; it is a static processing mechanism that is designed to operate in this manner. Crovella, ¶177.

Foster teaches BigInsight **operates…in a batch mode with a predetermined batch frequency**. For example, Foster explains that BigInsights uses a "MapReduce framework" and Hadoop file system. Ex-1004, §7.1 at 00338. It was well-known that MapReduce processing was done according to batches on a predetermined schedule. For example, it was known that "[i]n MapReduce environments, many production jobs are run *periodically* on new data." Ex-1012, §1; *supra* §I.B. This is acknowledged by the '693 patent, which provides "map-reduce batch jobs" as an example of "static processing." Ex-1001, 5:4-11, 16:26-34. If a batch is processed periodically, then it is processed at **a predetermined batch frequency** with intervals of a given period. Crovella, ¶178.

It would have been further obvious because BigInsights continuously accumulates large amounts of Big Data and batch processing at predetermined frequencies was well-known in the art. *Supra* §I.B; Crovella, ¶179.

[9b]	**wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof operate, at least in part, in substantially real-time or at a frequency greater that the predetermined batch frequency.**
[19b]	**wherein the at least one dynamic….**

Foster teaches the additional limitation, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶181-185. Foster discloses the **dynamic processing mechanism**…**operate[s]…in substantially real-time**. For example, Foster discloses that Streams operates on incoming streams for "real-time analysis of data in motion." Ex-1004, §7.2.1 at 00345, §2.1 at 00057. Foster equates data in motion to "current data" and "real-time data." *Id.*, §7.1 at 00338, §2.1 at 00057. Additionally, Foster explains that Streams uses models generated by BigInsights "as a basis for analytics on incoming data" received continuously and updates the models "based on *real-time observations*." *Id.*, §7.1.1 at 00339; Crovella, ¶182.

Foster also teaches that Streams operates **at a frequency greater tha[n] the predetermined batch frequency** of BigInsights. For example, Streams operates on data real-time or near real-time, and therefore operates at a frequency corresponding to the incoming rate of data. Streams operates at a greater frequency because the incoming rate of data is continuous, whereas BigInsights operates on historical or slow moving data according to batch processing (e.g.,

MapReduce), as discussed above. *Supra* §VI.A.11.9[a], §VI.A.3.[1d]; Crovella, ¶183.

Foster confirms this by stating that Streams "batch[es]" incoming data for BigInsights (Ex-1004, §7.1.2 at 00339), which is put in "storage" at BigInsights (*Id.*, §7.1.6 at 00343). Thus, Streams operates at a greater frequency because incoming data items are batched and stored to BigInsights where it is processed later. Crovella, ¶184.

10. Dependent Claims 10 and 20

Claims 1/11 were discussed above. The additional limitations of claims 10 and 20 are rendered obvious by Foster alone, and by Foster combined with Williams. Crovella, ¶¶186-195.

10[a].	**The method of claim 1, wherein the at least one static processing mechanism is associated with performing, at least in part, slow moving data analytics; and**
20[a]	**The apparatus of claim 11, wherein the at least one static…; and**

Foster teaches the additional limitation of 10[a]/20[a], and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶187-190. Foster teaches **the at least one static processing mechanism is associated with performing…slow moving data analytics**. For example, Foster discloses that BigInsights handles "data-at-rest." Ex-1004, §7.1 at 00338; *supra* §VI.A.3.[1d]. Data-at-rest is "data stored in a database." *Id.*, §2.1 at 00057. This data comes

from "input streams…ingested into BigInsights for *historical storage* and deep analysis." Ex-1004, §7.1.5 at 00342. Foster explains that such data stored in databases are "static." *Id.*, §2.1 at 00058. The data is slow moving, in contrast to "streaming data or data in motion" (real-time data). *Id.*, §1.1 at 00025; Crovella, ¶188.

BigInsights performs **slow moving data analytics** on this slow moving data. For example, BigInsights generates/updates models using "analytics such as data-mining, machine-learning, or statistical modeling." Ex-1004, §7.1.1 at 00338-339. Thus, BigInsights is **associated with performing…slow moving data analytics**. Crovella, ¶189.

[10b]	**wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof are associated with performing, at least in part, fast moving data analytics.**
[20b]	**wherein the at least one dynamic….**

Foster teaches the additional limitation, and the combination with Williams teaches this limitation for the same reasons. Crovella, ¶¶191-195. Foster teaches **the at least one dynamic processing mechanism [and] the at least one processing element…are associated with performing…fast moving data analytics**. For example, Foster discloses that Streams operates on incoming streams for "real-time analysis of data in motion." Ex-1004, §7.2.1 at 00345, §2.1 at 00057. Foster equates data in motion to "current data" and "real-time data." *Id.*,

§7.1 at 00338, §2.1 at 00057. The data is fast moving, in contrast to "static data" stored in a database. *Id.*, §2.1 at 00058; Crovella, ¶192.

Streams performs fast moving data analytics on this fast moving data. For example, Streams uses models generated by BigInsights "as a basis for *analytics* on incoming data" and updates the models "based on *real-time observations*." Ex-1004, §7.1.1 at 00339. The continuous stream of data received and prepared at Streams is "also forwarded to downstream analytics on Streams." *Id.*, §7.1.2 at 00339. Thus, Streams is **associated with performing…fast moving data analytics**. Crovella, ¶193.

Additionally, Streams determines a processing element from the incoming stream of real-time data, as discussed above for limitation [1a]. Supra §VI.A.3.[1a]. The determined processing elements are processed at Streams for providing real-time analytics as was just discussed. Thus, **the processing element** (e.g., data) is also **associated with performing…fast moving data analytics.** Crovella, ¶194.

VII. CONCLUSION

For the foregoing reasons, Petitioner requests *inter partes* review and that the challenged claims be canceled as unpatentable.

VIII. DISCRETIONARY ANALYSIS FOR REVIEW

Fintiv. Discretionary denial is not warranted because this Petition's merits are compelling, which "alone demonstrates that the PTAB should not discretionarily deny institution under *Fintiv*." *See* June 21, 2022 Memorandum, 3-5. No case management conference has occurred and no case schedule has issued in the parallel district court case.

Advanced Bionics. Denial under §325(d) is likewise unwarranted because Foster and Williams were not cited or considered during prosecution. This is the only IPR filed against the challenged claims to date. *General Plastic* factors do not weigh against institution.

IX. MANDATORY NOTICES AND FEES

1. Real Party In Interest: The real parties-in-interest here are Amazon.com, Inc. and Amazon.com Services LLC. No other parties directed, controlled, or funded this Inter Partes Review proceeding (IPR).

2. Related Matters:

- *Nokia Corporation and Nokia Technologies Oy v. Amazon.com, Inc. and Amazon.com Services, LLC*, Case No. 1:23-cv-01232 (D. Del.)

3. Lead and Backup Counsel: Pursuant to 37 C.F.R. §42.8(b)(3) and 42.10(a), Petitioner designates Harper Batts (Reg. No. 56,160) as lead counsel and

Chris Ponder (Reg. No. 77,167) and Jeffrey Liang (Reg. No. 69,043) as back-up counsel, each of Sheppard, Mullin, Richter & Hampton LLP.

4. **Service Information:** Petitioner consents to service by electronic mail addressed to:

HBatts@sheppardmullin.com;

CPonder@sheppardmullin.com;

JLiang@sheppardmullin.com; and

LegalTM-Amazon-Nokia-IPRs@sheppardmullin.com

Petitioner's counsel may also be served by mail or hand delivery at Sheppard, Mullin, Richter & Hampton LLP, 1540 El Camino Real Suite 120, Menlo Park, CA 94025. Petitioner's counsel may be reached by telephone at (650) 815-2673 and by facsimile at (650) 815-4668.

5. **Fees:** The Office is authorized to charge fees for this Petition to Deposit Account 50-4561, Ref. 19AG-385608.

Respectfully submitted,

SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/Harper Batts/
Harper Batts (Reg. No. 56,160)
Attorneys for Petitioner

CLAIM APPENDIX

1[pre]. A method comprising:

[1a] determining via a processor at least one processing element of at least one dynamic processing mechanism;

[1b] causing, at least in part, a marshalling via the processor of the at least one processing element as at least one data object;

[1c] transferring of the at least one data object to at least one static processing mechanism;

[1d] processing of the at least one data object by the at least one static processing mechanism; and

[1e] after processing, transferring the at least one data object back to the at least one dynamic processing mechanism.

2[a]. The method of claim 1, further comprising:

a processing of at least substantially real-time data via the at least one processing element to determine one or more triggering events,

[2b] wherein, the transferring of the at least one data object to the at least one static processing mechanism is based, at least in part, on the one or more triggering events, and

[2c] wherein the one or more triggering events include, at least in part, one or more contextual trends occurring above at least one threshold level.

3. The method of claim 2, further comprising: unmarshalling of the at least one processing element as the at least one data object.

4. The method of claim 2, wherein the processing of the at least one data object via the at least one static processing mechanism causes, at least in part, an updating of the at least one data object based, at least in part, on the one or more triggering events, the one or more contextual trends, or a combination thereof.

5[a]. The method of claim 4, further comprising: at least one determining of one or more computational chains, one or more parameters of the one or more computational chains, one or more data elements of the one or more computational chains, or a combination thereof associated with the at least one processing element,

[5b] wherein at least one data object includes, at least in part, the one or more computational chains, the one or more parameters, the one or more data elements, or a combination thereof.

7. The method of claim 1, wherein the processing the processing of the at least one data object causes, at least in part, updating of one or more components of the at least one static processing mechanism.

8[a]. The method of claim 1, aggregating of one or more outputs of the at least one dynamic processing mechanism and the one or more other outputs of the at least one static processing mechanism; and

[8b] at least one determination of one or more outputs of at least one processing job based, at least in part, on the aggregating.

9. The method of claim 1, wherein the at least one static processing mechanism operates, at least in part, in a batch mode with a predetermined batch frequency; and wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof operate, at least in part, in substantially real-time or at a frequency greater that the predetermined batch frequency.

10. The method of claim 1, wherein the at least one static processing mechanism is associated with performing, at least in part, slow moving data analytics; and wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof are associated with performing, at least in part, fast moving data analytics.

11[pre]. An apparatus comprising:

[11a] at least one processor; and

at least one memory including computer program code for one or more programs,

the at least one memory and the computer program code configured to, with the at least one processor, cause the apparatus to perform at least the following,

[11b] determine at least one processing element of at least one dynamic processing mechanism;

[11c] cause a marshalling via the processor of the at least one processing element as at least one data object;

[11d] transfer of the at least one data object to at least one static processing mechanism;

[11e] process the at least one data object by the at least one static processing mechanism; and

[11f] after the process, transfer the at least one data object back to the at least one dynamic processing mechanism.

12[a]. The apparatus of claim 11, wherein the apparatus is further caused to:

process and/or facilitate a processing of at least substantially real-time data via the at least one processing element to determine one or more triggering events,

[12b] wherein the transfer of the at least one data object to the at least one static processing mechanism based, at least in part, on the one or more triggering events, and

[12c] wherein the one or more triggering events include, at least in part, one or more contextual trends occurring above at least one threshold level.

13. The apparatus of claim 12, wherein the apparatus is further caused to: cause, at least in part, an unmarshalling of the at least one processing element as at least one data object.

14. The apparatus of claim 12, wherein the process of the at least one data object via the at least one static processing mechanism causes, at least in part, an updating of the at least one data object based, at least in part, on the one or more triggering events, the one or more contextual trends, or a combination thereof.

15[a]. The apparatus of claim 14, wherein the apparatus is further caused to:

determine one or more computational chains, one or more parameters of the one or more computational chains, one or more data elements of the one or more computational chains, or a combination thereof associated with the at least one processing element,

[15b] wherein at least one data object includes, at least in part, the one or more computational chains, the one or more parameters, the one or more data elements, or a combination thereof.

17. The apparatus of claim 11, wherein process of the at least one data object causes, at least in part, an updating of one or more components of the at least one static processing mechanism.

18[a]. The apparatus of claim 11, wherein the apparatus is further caused to:

cause, at least in part, an aggregation of one or more outputs of the at least one dynamic processing mechanism and the one or more other outputs of the at least one static processing mechanism; and

[18b] determining one or more outputs of at least one processing job based, at least in part, on the aggregation.

19. The apparatus of claim 11, wherein the at least one static processing mechanism operates, at least in part, in a batch mode with a predetermined batch frequency; and wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof operate, at least in part, in substantially real-time or at a frequency greater that the predetermined batch frequency.

20. The apparatus of claim 11, wherein the at least one static processing mechanism is associated with performing, at least in part, slow moving data analytics; and wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof are associated with performing, at least in part, fast moving data analytics.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on October 18, 2024, true and correct copies of the foregoing Petition for *Inter Partes* Review of U.S. Patent No. 8,996,693 and Exhibits 1001-1017 were served in entirety on the Patent Owner at the following address of record as listed on Patent Center, via Priority Mail Express®:

Nokia Corporation and Alston & Bird LLP
Vantage South End
1120 South Tryon Street, Suite 300
Charlotte, NC 28203-6818
Official Correspondence Address

The undersigned certifies that true and correct copies of the Petition for I*nter Partes* Review of U.S. Patent No. 8,996,693 and Exhibits 1001-1017 were also sent via electronic mail to the attorneys of record for Plaintiff in the concurrent litigation matter(s):

Brian E. Farnan
bfarnan@farnanlaw.com
Michael J. Farnan
mfarnan@farnanlaw.com
Warren H. Lipschitz
wlipschitz@mckoolsmith.com
Alexandra F. Easley
aeasley@mckoolsmith.com
R. Mitch Verboncoeur
mverboncoeur@mckoolsmith.com
Joshua Newcomer
jnewcomer@mckoolsmith.com
Kevin Burgess
kburgess@mckoolsmith.com

Bryan Lutz
bryan.lutz@alston.com
Nicholas T. Tsui
nick.tsui@alston.com
Mark A. McCarty
mark.mccarty@alston.com
Shawn P. Gannon
shawn.gannon@alston.com
Stephen R. Lareau
stephen.lareau@alston.com
Theodore Stevenson, III
ted.stevenson@alston.com
John D. Haynes
john.haynes@alston.com

Dated: October 18, 2024

/Harper Batts/
Harper Batts (Reg. No. 56,160)
Attorney for Petitioner

1540 El Camino Real Suite 120
Menlo Park, CA 94025
Tel: (650) 815-2600

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE REQUIREMENTS

1. This Petition complies with the type-volume limitation of 14,000 words, comprising 13,887 words, as counted using the Microsoft Word software that was used to prepare this paper, excluding the parts exempted by 37 C.F.R. § 42.24(a).

2. This Petition complies with the general format requirements of 37 C.F.R. § 42.6(a) and has been prepared using Microsoft® Word in 14-point Times New Roman.

Respectfully submitted,

SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/Harper Batts/
Harper Batts (Reg. No. 56,160)
Attorney for Petitioner

Date: October 18, 2024

1540 El Camino Real Suite 120
Menlo Park, CA 94025
Tel: (650) 815-2673