

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

Case: IPR2024-01507
U.S. Patent No. 8,996,693

PATENT OWNER’S PRELIMINARY RESPONSE

Mail Stop “PATENT BOARD”
Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-145

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	BACKGROUND	2
A.	Dynamic and Static Processing Mechanisms.....	2
B.	The '693 Patent	4
1.	Overview	4
2.	Processing element.....	6
3.	Marshalling	10
C.	The Asserted Prior Art	13
1.	Foster (Ex-1004)	13
2.	Williams (Ex-1005).....	16
D.	Person of Ordinary Skill in the Art	17
III.	CLAIM CONSTRUCTION	17
A.	Legal Principles of Claim Construction.....	17
B.	The Plain and Ordinary Meaning of “Processing Element”	19
C.	Petitioner Fails to Show a Plain-Meaning Exception is Met	27
D.	Aside From its Admission Regarding the Well-Known Meaning of “Processing Element,” Petitioner’s Expert’s Declaration Should be Afforded Little to No Weight.....	31
IV.	ARGUMENT	33
A.	Petitioner Has Not Established a Reasonable Likelihood that the Independent Claims are Unpatentable Under Ground 1	33
1.	Foster does not disclose or render obvious claim [1a]’s “determining via a processor at least one processing element of at least one dynamic processing mechanism” (or similar language in claim [11b])	33
2.	Foster does not disclose or render obvious claim [1b]’s “causing, at least in part, a marshalling via the processor of the at least one processing element as at least one data object” (or similar language in claim [11c])	35

3.	Foster does not disclose the “data object” found in claims [1c], [1d], and [1e] (or similar language in claims [11d], [11e], and [11f])	38
B.	Petitioner Has Not Established a Reasonable Likelihood That Any of the Dependent Claims are Unpatentable Under Ground 1	41
C.	Petitioner Has Not Established a Reasonable Likelihood that the Independent Claims are Unpatentable Under Ground 2	41
D.	Petitioner Has Not Established a Reasonable Likelihood That Any of the Dependent Claims are Unpatentable Under Ground 2	42
V.	CONCLUSION.....	43

TABLE OF AUTHORITIES

	Page(s)
CASES	
<i>ACTV, Inc. v. Walt Disney Co.</i> , 346 F.3d 1082 (Fed. Cir. 2003)	19
<i>CCS Fitness, Inc. v. Brunswick Corp.</i> , 288 F.3d 1359 (Fed. Cir. 2002)	18
<i>Graham v. John Deere Co.</i> , 383 U.S. 1 (1966).....	37
<i>Helmsderfer v. Bobrick Washroom Equip., Inc.</i> , 527 F.3d 1379 (Fed. Cir. 2008)	18, 30
<i>In re Am. Acad. of Sci. Tech Ctr.</i> , 367 F.3d 1359 (Fed. Cir. 2004)	31
<i>In re Fine</i> , 837 F.2d 1071 (Fed. Cir. 1988)	41, 43
<i>In re Omeprazole Pat. Litig.</i> , 483 F.3d 1364 (Fed. Cir. 2007)	36
<i>K-fee Sys. GmbH v. Nespresso USA, Inc.</i> , 89 F.4th 915 (Fed. Cir. 2023)	31
<i>KSR Int’l Co. v. Teleflex, Inc.</i> , 550 U.S. 398 (2007).....	37
<i>Phillips v. AWH Corp.</i> , 415 F.3d 1303 (Fed. Cir. 2005) (en banc)	18, 24
<i>Prometheus Lab’ys, Inc. v. Roxane Lab’ys, Inc.</i> , 805 F.3d 1092 (Fed. Cir. 2015)	37
<i>Rohm & Haas Co. v. Brotech Corp.</i> , 127 F.3d 1089 (Fed. Cir. 1997)	32
<i>Samsung Elecs. Co., Ltd. v. RightQuestion, LLC</i> , IPR2022-00251, Paper 17 (PTAB June 9, 2022)	passim

<i>Thorner v. Sony Computer Ent. Am. LLC</i> , 669 F.3d 1362 (Fed. Cir. 2012)	18, 19, 27, 30
<i>Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.</i> , 200 F.3d 795 (Fed. Cir. 1999)	17

STATUTES

35 U.S.C. § 282(b)	17, 18
--------------------------	--------

OTHER AUTHORITIES

37 C.F.R. § 42.65(a).....	32
37 C.F.R. § 42.100(b)	18

TABLE OF EXHIBITS

Exhibit	Description
2001	RFC 2713, “Schema for Representing Java (tm) Objects in an LDAP Directory,” <i>available at</i> https://datatracker.ietf.org/doc/html/rfc2713 (October 1999)

I. INTRODUCTION

Petitioner advances two grounds in its Petition (Paper 3, “Pet.”). Both grounds allege that claims 1-5, 7-15, and 17-20 (the “Challenged Claims”) of U.S. Patent No. 8,996,693 (Ex-1001, the “’693 Patent”) are obvious, and both grounds rely on Foster (Ex-1004) as the primary reference. Pet., 13-14. Petitioner’s first ground relies on Foster alone, and its second ground relies on Foster combined with Williams (Ex-1005). *Id.* The Petition fails on both grounds.

Both grounds are premised on an erroneous construction of the term “processing element,” which appears in all Challenged Claims (including via both independent claims). As Petitioner’s expert admits, the “well-known use of the term [is] as an element that carries out processing (e.g., a processing mechanism).” Ex-1003, ¶27. The ’693 Patent confirms that this can include at least code or parameters. *E.g.*, Ex-1001, 17:17-24. Despite this, Petitioner argues that the Board should give the term a construction at odds with the well-known understanding of the term—mere “data to be processed.” Pet., 19. But Petitioner’s limited claim construction analysis is flawed, including because Petitioner’s construction renders the claims nonsensical while Petitioner provides no showing that the patentee acted as its own lexicographer. Petitioner also attempts to bolster its erroneous claim construction through its expert. However, other than acknowledging the “well-known use of the term,” Petitioner’s expert merely parrots the Petition. *Compare* Pet., 19 *with* Ex-

1003, ¶¶26-28. The Board should disregard that conclusory testimony and decline to deviate from the well-known, plain and ordinary meaning of “processing element” “as an element that carries out processing (e.g., a processing mechanism).” *Samsung Elecs. Co., Ltd. v. RightQuestion, LLC*, IPR2022-00251, Paper 17 at 24 (PTAB June 9, 2022) (“Samsung/RightQuestion”) (“Petitioner’s declarant also makes this same statement verbatim, without more, and with no further explanation or additional citations to evidence. ... We give such conclusory, unsupported assertions by Petitioner’s expert little or no weight.” (internal citations omitted)).

With Petitioner’s improper deviation from the well-known meaning of “processing element” set aside, both of Petitioner’s grounds fail. Accordingly, Patent Owner respectfully requests that the Board deny institution.

II. BACKGROUND

A. Dynamic and Static Processing Mechanisms

Historically, data processing and analysis “have been segregated based on whether the data being processed is ‘slow moving’ data (e.g., static or relatively static data that is collected over long periods of time such as user behavior data, service usage information, etc. processed by analytics systems) or whether the data is collected as data streams in real-time (e.g., social networking feeds, location tracking feeds, etc. processed by stream processing systems).” Ex-1001, 1:22-29. Moreover, “[e]ach different type of data (e.g., slow moving vs. real-time) has had

different data processing architectures and techniques that traditionally have been used in isolation.” *Id.*, 1:29-32.

These different data processing architectures and techniques can be classified as “static and/or dynamic methods.” Ex-1001, 3:58-62. For instance, “[i]n 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.)” *Id.*, 3:62-67. And “in the dynamic processing [method] (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.” *Id.*, 3:67-4:5.

Static and/or dynamic processing can be performed by mechanisms that operate differently based on the data being processed. For example, “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 PE [processing element], or a combination thereof operate, at least in part, in substantially real-time or at a frequency greater than the predetermined batch frequency.” Ex-1001, 13:40-46. Moreover, in some embodiments, “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 PE [processing element], or a combination thereof are associated with performing, at least in part, fast moving data analytics.” *Id.*, 13:46-52.

B. The ’693 Patent

U.S. Patent No. 8,996,693 (the “’693 Patent”) was filed as U.S. Application No. 13/621,511 on September 17, 2012, and issued by the United States Patent and Trademark Office (“USPTO”) on March 31, 2015. Ex-1001, Cover. On December 8, 2015, the USPTO issued a certificate correction for typographical errors in claims 7, 8, and 17. Ex-1002, 000027.

1. Overview

The ’693 Patent introduces techniques “for integrating data processing methods for more accurate and comprehensive results.” Ex-1001, Abstract. The inventors of the ’693 Patent recognized and appreciated how “[e]ach different type of data (e.g., slow moving vs. real-time) has had different data processing architectures and techniques that traditionally have been used in isolation” and that as result, “service providers and device manufacturers face significant 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.” *Id.*, 1:29-36. The inventors therefore sought to provide “data processing

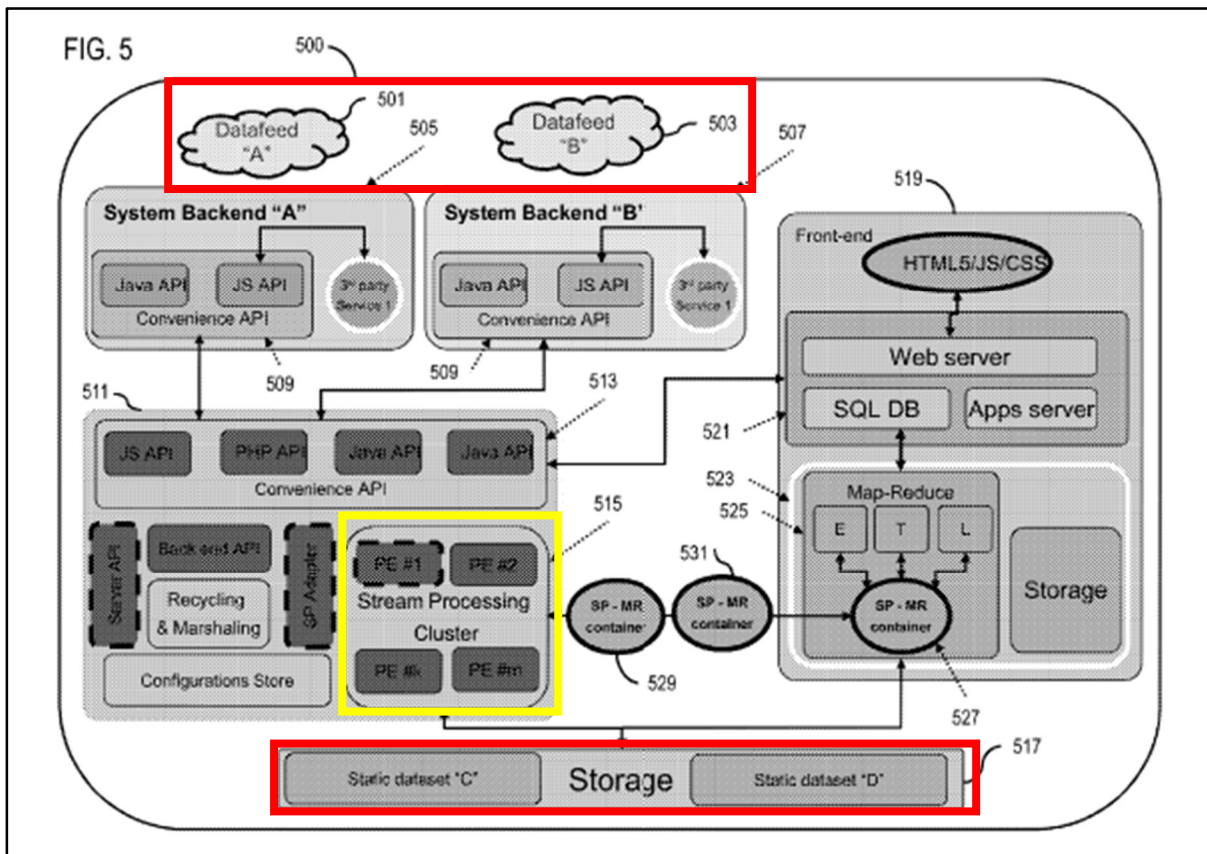
methods to generate comprehensive results that combines the features of a slow moving data systems with real-time data systems.” *Id.*, 3:40-42.

This combination includes, for example, “the processing platform 107 determin[ing] at least one PE [processing element] of at least one dynamic processing mechanism (e.g., engine).” Ex-1001, 13:11-13. Next, “the processing platform 107 causes, at least in part, a marshalling of the at least one PE as at least one data object, wherein the at least one data object is processable by at least one static processing mechanism.” *Id.*, 13:37-40. In some embodiments “the processing platform 107 processes and/or facilitates a processing of at least substantially real-time data via the at least one PE to determine one or more triggering events” and “the processing platform 107 causes, at least in part, a 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.” *Id.*, 13:53-14:4. Next, “the processing platform 107 processes and/or facilitates a processing of the at least one data object via the at least one static processing mechanism.” *Id.*, 14:23-25. The ’693 Patent provides examples where this processing takes the form of a “shaping process” that “may pass the PE as a data object to the static processing where it may update parameters, computations, and/or re-encode (e.g., marshalling) the updated computations, parameters, etc. back into data object for transmission to the dynamic process.” *Id.*, 15:7-12.

Accordingly, the '693 Patent provides for a system with “feedback mechanisms” between static and dynamic processing mechanisms in order to take advantage of both techniques for processing data. Ex-1001, 4:5-22.

2. Processing element

In one embodiment, the '693 Patent describes a “processing element” (or “PE”) as being a part of a dynamic processing mechanism. Ex-1001, 6:58-60 (“In one embodiment, the system 100 determines at least one PE of at least one dynamic processing mechanism (e.g., engine)”); *see also id.*, 25:34-35 (claim 1: “determining via a processor at least one processing element of at least one dynamic processing mechanism”). An example of processing elements is also shown in Figure 5 (below, with yellow box added to emphasize the processing elements) and described as making up a “stream processing cluster.” *Id.*, 16:7-14 (“the processing platform 511 includes an stream processing cluster 515 (e.g., dynamic process) which may further include an S4 engine and various PEs with access to one or more data stores 517, which may store various data (e.g., user data, sensor data, historical data, etc.).” The processing elements (PEs) are not shown as part of the data store or data feeds (below, with red boxes added to emphasize), but are instead described as part of the processing platform and may access the data store. *Id.*



Id., Figure 5 (annotations added).

The '693 Patent provides several examples of processing elements, and in each example the processing elements are used for processing. For instance, “the encapsulated computational chains form the processing elements (PEs) of stream processing mechanisms.” Ex-1001, 4:31-34. The specification describes the processing elements being associated with a processing speed. *Id.*, 7:27-30 (“...wherein the at least one dynamic processing mechanism, the at least one PE, or a combination thereof operate, at least in part, in substantially real-time or at a frequency greater than the predetermined batch frequency”). Moreover, the '693

Patent explains that “the PE may include an executable code (e.g., binary code), which may be executing in the stream process, wherein the data object may include information on the current process state, execution steps (e.g., snapshot of the state of the process), values of variables, and the like where the data object may continue to be processed at a different time and/or in a different process (e.g., static process).” *Id.*, 7:36-43.

The '693 Patent discloses further examples of what may make up a processing element. For instance, processing elements “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 (e.g., a Twitter® feed, location based information, etc.), wherein the PEs may indicate a most common topic, location, hash-tag, and the like.” Ex-1001, 6:63-7:1. Moreover, processing elements “may include data as well as executing code where one or more functions of a process may be defined (e.g., process a counter, determine occurrence of certain data patterns or popularity of a certain topic, traffic analysis, route planning, reverse geocoding (RGC), etc.)” *Id.*, 7:2-6.

The '693 Patent further explains how the processing elements may process data to determine one or more triggering events. Ex-1001, 7:44-51 (“In one embodiment, the system 100 processes and/or facilitates a processing of at least substantially real-time data via the at least one PE to determine one or more

triggering events.”), *see also id.*, 25:46-48 (claim 2: “...a processing of at least substantially real-time data via the at least one processing element to determine one or more triggering events.”).

The '693 Patent also discloses that processing elements can be updated. For example, processing elements can be updated by updating parameters or code. Ex-1001, 17:17-24 (“In one embodiment, processing results from the static processing mechanism 603 are transferred back to the dynamic processing mechanism 601 as updated PEs (e.g., including updated parameters, computational codes, etc.), wherein one or more configurations of the dynamic processing mechanism 601 may be updated according to the updated PEs (e.g., one or more trends, additional data, updated codes, etc.)”).

The '693 Patent additionally describes the processing elements being marshalled into a data object that can be processed by a static processing mechanism. For instance, a data object may include various information associated with the processing element including “information on the current process state, execution steps (e.g., snapshot of the state of the process), values of variables, and the like where the data object may continue to be processed at a different time and/or in a different process.” Ex-1001, 7:38-42.

The '693 Patent does not anywhere state or suggest that a processing element can take the form of mere data. *See generally* Ex-1001.

3. Marshalling

The '693 Patent explains that “marshalling may be considered as a process for transforming memory representation of an object to a data format (e.g., similar to serialization) suitable for storage and/or transmission of the data to different components of a process and/or from one process to another process.” Ex-1001, 4:37-42; *see also id.*, 7:20-24 (“In one embodiment, marshalling is a process of encoding an object for sharing, sending, and/or communicating the object to one or more component, processes, and the like of a computer and/or a network (e.g., from a client to a server).”). Marshalling is useful because different components or processes in a system may be configured to accept inputs in different forms.

Petitioner suggests a view of marshalling unlike how the term is used in the '693 Patent. For instance, the Petition cites Williams and states that “[m]arshalling was a generic term for gathering data from one process and converting it into a format that can be used either for storage or for transmission to another process.” Pet., 9 (citing Ex-1005, 00039 (§1.4.3)). Petitioner’s expert parrots this line in his declaration. Ex-1003, ¶9; *see Samsung/RightQuestion*, IPR2022-00251, Paper 17 at 24 (stating that limited weight should be given to Petitioner’s declarant who makes the “same statement verbatim, without more, and with no further explanation or additional citations to evidence”). But consider the Ex-1015 quote that Petitioner and its expert cite (*i.e.*, “Marshalling involves converting local objects into portable

form so that they can be transmitted to a remote process.”). Pet., 9 (citing Ex-1015, 000064), Ex-1003, ¶9 (quoting Ex-1015, 000064). Neither Ex-1015 nor the ’693 Patent describe marshalling as the process of “gathering data from one process and converting it into a format that can be used either for storage or for transmission to another process.” Pet., 9 (emphasis added). Instead, like the ’693 Patent, Petitioner’s Ex-1015 describes how marshalling “involves converting local objects into portable form so that they can be transmitted to a remote process.” Ex-1015, 000064 (emphasis added)).

Petitioner’s “marshalling” background section is inaccurate to the extent it presents marshalling as a process that can only be applied to data. For example, the ’693 Patent describes generating and storing information in a Resource Description Framework as an example of marshalling computational closures. Ex-1001, 11:33-41. (“In one embodiment, the computation closures are marshalled as one or more data objects to facilitate transfer and processing among processing platforms 107 (e.g., static or slow moving data systems such as analytics engines, and real-time stream processing systems such as S4 systems.”). These computational closures are not mere data, but are computational procedures that process data. *Id.*, 11:23-32 (“The term ‘computational closure’ refers to a particular data retrieval and computation procedure together with relations and communications among various processes including passing arguments, sharing process results, flow of data and

process results, etc. The computation closures (e.g., a granular reflective set of instructions, data, and/or related execution context or state) provide the capability of slicing of computations and data for processes and transmitting the slices between devices, infrastructures, and processes.”).

Accordingly, “marshalling” can apply to data along with other objects such as code or parameters. This understanding is consistent with other pre-’693 Patent usage of the term. Ex-1015, 000064 (“Marshalling involves converting local objects into portable form so that they can be transmitted to a remote process.”); Ex-2001, 4-5 (§2.3) (“To ‘marshal’ an object means to record its state and codebase(s) in such a way that when the marshalled object is ‘unmarshalled,’ a copy of the original object is obtained, possibly by automatically loading the class definitions of the object. You can marshal any object that is serializable or remote (that is, implements the `java.rmi.Remote` interface). Marshalling is like serialization, except marshalling also records codebases.”).

Unmarshalling is the reverse process of marshalling and includes decoding or de-serializing the data object that was re-coded or serialized as part of the marshalling process. Ex-1001, 8:11-13; *see also* Ex-2001, 4-5 (§2.3).

C. The Asserted Prior Art

1. Foster (Ex-1004)

Foster is a 456-page book released as part of IBM's Redbooks series. Ex-1004, 000001. Foster discusses an IBM product called InfoSphere Streams (hereinafter, "Streams"). *Id.*, 000037. Foster describes Streams as "a product architected specifically to help clients continuously analyze massive volumes of streaming data at extreme speeds to improve business insight and decision making." *Id.*

Relevant to the Petition, Streams is able to share and receive data from another IBM product called InfoSphere BigInsights (hereinafter, "BigInsights"). Foster describes Streams as a platform meant to process data-in-motion (current data), and BigInsights as a platform that processes data-at-rest (historical data). Ex-1004, 000338. The relationship between Streams and BigInsights is summarized in Foster's Figure 7-1, replicated below:

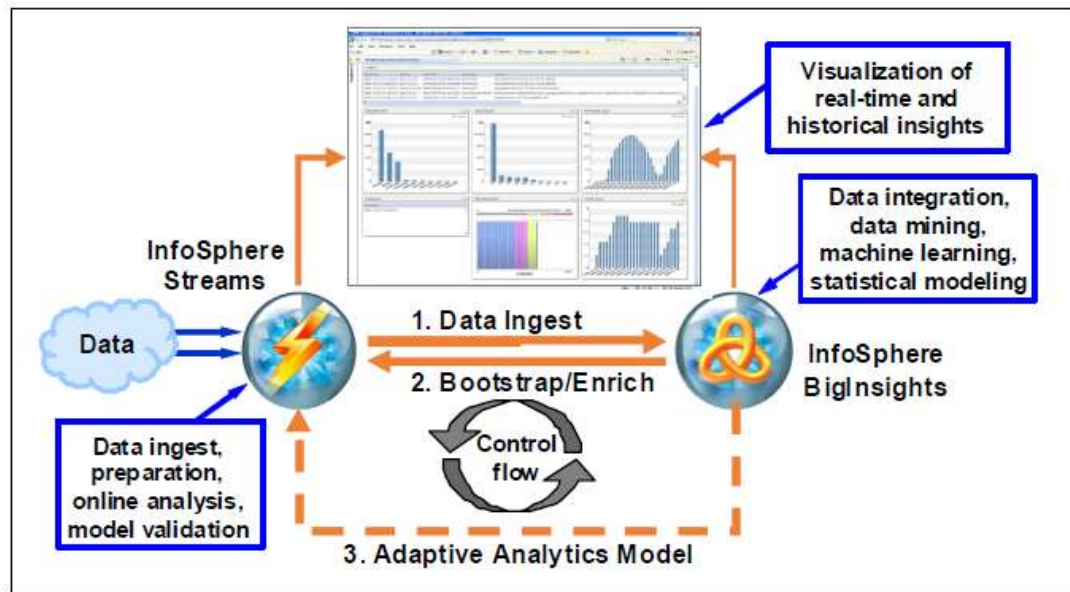


Figure 7-1 Big Data application scenarios

Id., 000339.

Streams is represented by the sphere on the left of Figure 7-1, while BigInsights is represented by the sphere on the right. Data is continuously input into Streams for real-time analysis, but at the same time, data is being “ingested” into BigInsights for “deep analysis.” Ex-1004, 000339-000340. This creates a parallel pipeline architecture where data is processed both by Streams and BigInsight. *Id.* Foster also describes how formatting operators can be used to “transform data between the Streams tuple format and the data formats used by BigInsights.” *Id.*, 000344.

Foster describes three features to illustrate how Streams and BigInsight can be integrated. The first feature is called “data ingest” and is represented by arrow 1 in Figure 7-1 above. Ex-1004, 000339-000340. The data ingest feature involves data

being received by Streams and pre-processed before being passed along to BigInsights. *Id.*, 000339. Foster describes various pre-processing steps that can be performed, including “data-cleansing, filtering, feature extraction, deduplication, and normalization.” *Id.* Relatedly, Streams can be used to identify and remove irrelevant or duplicate data before passing along the data to BigInsights. *Id.*, 000340.

The second feature is bootstrap and enrichment, which is illustrated by arrow 2 in Figure 7-1 above. Ex-1004, 000340. This feature involves BigInsights processing data over a longer period of time to determine ways to enrich the data. *Id.* For example, BigInsights may process data received from social media messages over a long period of time to identify information about the sender of the message. *Id.* BigInsights can enrich this data with the sender information and send the enriched data back to Streams. *Id.*

The third feature is the adaptive analytics model represented by arrow 3 in Figure 7-1 above. Ex-1004, 000341. With this feature, BigInsights uses data received from Streams to construct a data model to represent “observations” from BigInsight’s analysis of the data. *Id.* Streams may incorporate the data model received from BigInsights and apply it to future data streams. *Id.* Foster also describes Streams determining scenarios where the data model’s prediction deviates significantly from the data actually being received. *Id.* This may trigger BigInsights to build a new data model. *Id.*

Although **data** is transferred between Streams and BigInsights, none of these features describe any object used for processing being transferred from Streams to BigInsights. *See generally* Ex-1004, 000339-000341.

2. Williams (Ex-1005)

Williams is a 596-page book from IBM's Redbook's series. Ex-1005, 000001. Williams discusses IBM's Web services project used to help businesses with web-based services. *Id.*, 000026.

Petitioner cites to only a single section from a single page of Williams in the body of its analysis, and Petitioner does so for the narrow purpose of generally disclosing the concepts of marshalling and unmarshalling. Pet., 33 (citing Ex-1005, 000039 (§1.4.3)). The section of Williams that Petitioner relies upon is shown below:

1.4.3 Encodings

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*.

When implementing a Web service, we have to choose one of the tools and programming or scripting languages that support the Web services model. However, the protocol format for Web services is XML, which is independent of the programming language. Thus, SOAP encodings tell the SOAP runtime environment how to translate from data structures constructed in a specific programming language into SOAP XML and vice versa.

The following encodings are defined:

SOAP encoding	The <i>SOAP encoding</i> enables marshalling/unmarshalling of values of data types from the SOAP data model. This encoding is defined in the SOAP 1.1 standard.
Literal	The <i>literal</i> encoding is a simple XML message that does not carry encoding information. Usually, an XML Schema describes the format and data types of the XML message.

Ex-1005, 000039.

D. Person of Ordinary Skill in the Art

Regardless of the definition of a person of ordinary skill in the art (“POSITA”), the Petition fails. Accordingly, Patent Owner has applied Petitioner’s POSITA definition (Pet., 17-19) solely for the sake of this Response. Patent Owner reserves its right to argue in favor of a different level of skill should trial be instituted.

III. CLAIM CONSTRUCTION

Petitioner proposes constructions for three claim terms: (1) “processing element”; (2) “dynamic processing mechanism”; and (3) “static processing mechanism.” Pet., 19-22. Construction of (2) “dynamic processing mechanism” and (3) “static processing mechanism” is not necessary to resolve whether institution should be granted and so the Board need not construe them at this stage. *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999).

Construction of “processing element” is necessary. Petitioner’s construction of the term is inconsistent with its plain and ordinary meaning in the field of data analytics and as it is used in the ’693 Patent. And as discussed in Section IV below, Petitioner’s erroneous construction of this term is dispositive: when “processing element” is afforded its proper construction, the Petition fails.

A. Legal Principles of Claim Construction

In an *inter partes* review, the Board “us[es] the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C.

[§] 282(b).” 37 C.F.R. § 42.100(b). Under the civil action standard, the Board construes the claim “in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.” *Id.*; see also Patent Trial and Appeal Board Consolidated Trial Practice Guide (Nov. 2019), 46, available at <https://www.uspto.gov/TrialPracticeGuideConsolidated> (“Trial Practice Guide”).

“[T]he ordinary and customary meaning of a claim term is the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention” and “after reading the entire patent” and its prosecution history. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1313, 1321 (Fed. Cir. 2005) (en banc). “There are only two exceptions to this general rule: 1) when a patentee sets out a definition and acts as his own lexicographer, or 2) when the patentee disavows the full scope of a claim term either in the specification or during prosecution.” *Thorner v. Sony Computer Ent. Am. LLC*, 669 F.3d 1362, 1365 (Fed. Cir. 2012) (internal citations omitted). To act as lexicographer, “a patentee must ‘clearly set forth a definition of the disputed claim term’ other than its plain and ordinary meaning” and “must ‘clearly express an intent’ to redefine the term”; “[i]t is not enough for a patentee to simply disclose a single embodiment or use a word in the same manner in all embodiments.” *Id.* (quoting *CCS Fitness, Inc. v. Brunswick Corp.*, 288 F.3d 1359, 1366 (Fed. Cir. 2002); *Helmsderfer v. Bobrick Washroom Equip., Inc.*, 527 F.3d

1379, 1381 (Fed. Cir. 2008)). “The standard for disavowal of claim scope is similarly exacting,” and may be met where “the specification makes clear that the invention does not include a particular feature” or where a patentee “include[s] in the specification expressions of manifest exclusion or restriction, representing a clear disavowal of claim scope.” *Id.* at 1366 (internal quotations omitted).

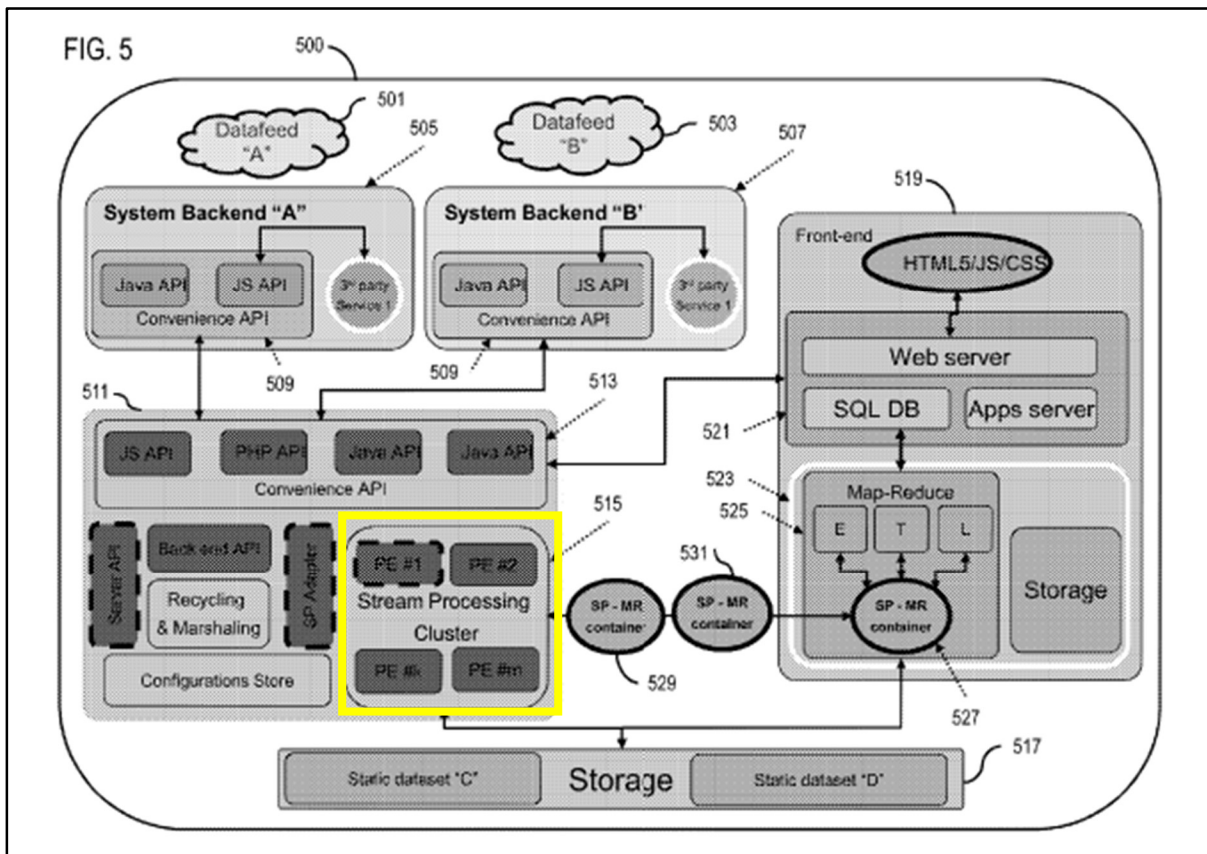
B. The Plain and Ordinary Meaning of “Processing Element”

Petitioner’s expert explains that the “well-known use of the [“processing element”] term [is] as an element that carries out processing (e.g., a processing mechanism).” Ex-1003, ¶27. The ’693 Patent confirms that this can include at least code or parameters. *E.g.*, Ex-1001, 17:17-24. Yet Petitioner argues that the term should be construed as “data to be processed,” which Petitioner’s expert says is the “reverse sense” of the plain meaning. Pet. 19; Ex-1003, ¶27. Petitioner’s construction ignores the term’s well-established meaning and how the ’693 Patent uses the term in accordance with this well-established meaning.

The ’693 Patent’s claims use the “processing element” term consistent with its well-established meaning. *ACTV, Inc. v. Walt Disney Co.*, 346 F.3d 1082, 1088 (Fed. Cir. 2003) (“First and foremost, the analytical focus of claim construction must begin, and remain centered, on the language of the claims themselves.” (internal citations omitted)). The “processing element” term is first introduced in limitation

[1a]:¹ “determining via a processor at least one processing element of at least one dynamic processing mechanism.” *See also* Claim [11b] (including similar claim language). That is, a processing element (PE) is a component or part of a dynamic processing mechanism, which implies the processing element carries out some processing. The ’693 Patent’s specification comports with this language and discloses that processing elements are within a stream processing cluster 515 of a processing platform 511, as shown in Figure 5 (below, with yellow box added to emphasize the processing elements). Ex-1001, 16:7-14 (“the processing platform 511 includes an stream processing cluster 515 (e.g., dynamic process) which may further include an S4 engine and various PEs with access to one or more data stores 517, which may store various data (e.g., user data, sensor data, historical data, etc.).” Thus, a “processing element” must be a component that carries out data processing; it cannot be “data to be processed.” Ex-1003, ¶27.

¹ For ease of reference, Patent Owner uses Petitioner’s claim limitation labels. *See* Pet., 81-86 (Claim Appendix).



Id., Figure 5 (annotation added).

Moreover, utilizing Petitioner’s proposed “processing element” and “dynamic processing mechanism” constructions would render claim [1a] (and, similarly, [11b]) nonsensical. Inserting both of Petitioner’s constructions in claim [1a] reads:

determining via a processor at least one [**data to be processed**] of at least one [**mechanism for processing data in real-time or near real-time**];

It is unclear what “at least one data to be processed” of a “mechanism for processing data” means. Indeed, the ’693 Patent’s specification and the Petition explain that a dynamic stream process operates (*e.g.*, in the form of code or steps of computing)

on data feeds, but does not consist of data itself. Ex-1001, 5:8-10; Pet., 20. The nonsensical nature of claim [1a] with Petitioner’s constructions inserted confirms the impropriety of Petitioner’s deviation from the well-established meaning of “processing element.”

The dependent claims confirm that the patentee used “processing element” in accordance with its well-established meaning. For example, dependent claims 2[a] and 12[a] require “a processing of at least substantially real-time data via the at least one processing element to determine one or more triggering events.” These claims thus require that a “processing element” be able to “process[]” “at least substantially real-time data” “to determine one or more triggering events.” This processing by a “processing element” would be implausible if a “processing element” were “data to be processed.” And again, inserting Petitioner’s proposed construction of “processing element” renders this dependent claim 2[a] nonsensical:

The method of claim 1, further comprising: a processing of at least substantially real-time data via the at least one [*data to be processed*] to determine one or more triggering events;

See also Claim 12[a] (including similar claim language).

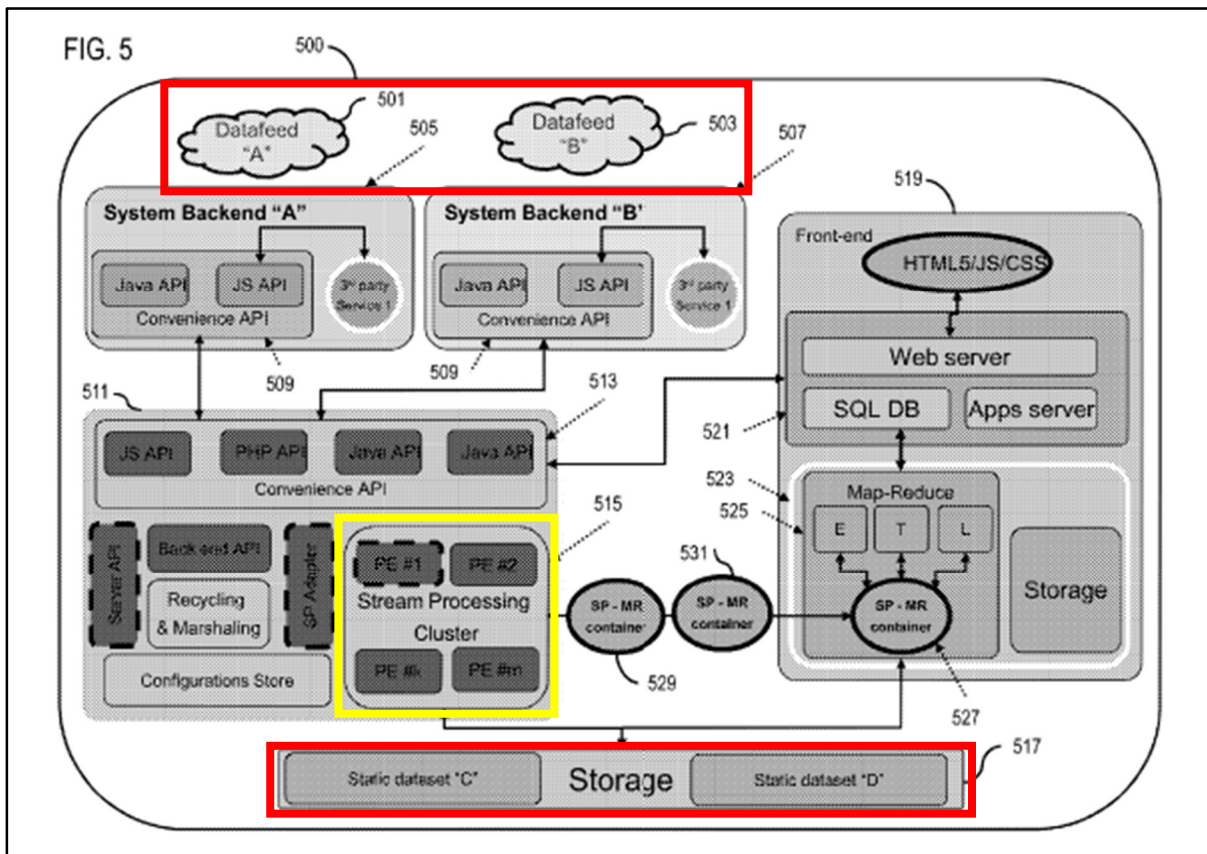
As another example, dependent claims 9 and 19 require that “wherein the at least one dynamic processing mechanism, the at least one processing element, or a combination thereof operate, at least in part, in substantially real-time or at a

frequency greater that the predetermined batch frequency.”² But data alone does not “operate” at any frequency. Instead, processing mechanisms operate at a processing frequency. For example, the ’693 Patent’s specification explains that “the static processing mechanism 603 operates in a batch-mode with a predetermined batch frequency, for example, daily, weekly, etc., and the dynamic processing mechanism 601 operates in substantially real-time or at a frequency greater that the predetermined batch frequency.” Ex-1001, 13:40-46.

As a final example, dependent claims 10 and 20 require, *inter alia*, “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.” This language contemplates that the “at least one processing element” be associated with performing “fast moving data analytics.” Data analytics implies processing. Ex-1001, 4:23-26 (“To address, at least, these problems, a system 100 of FIG. 1 introduces the capability for utilization of static and dynamic data analytics in various data processing mechanisms (e.g., distributed and/or local analytics engines).” Moreover, it is nonsensical to say that data alone can perform analytics of “fast moving data,” which is what Petitioner’s construction suggests.

² There appears to be a typo that reads “greater that” but was intended to read “greater than.”

The '693 Patent specification also uses the “processing element” term consistent with its well-established meaning. *Phillips*, 415 F.3d at 1315 (noting that “the specification is always highly relevant to the claim construction analysis” (internal quotations omitted)). For example, the specification describes “processing elements” as making up a “stream processing cluster,” which stands in contrast to its description of “data stores” as “stor[ing] various data.” Ex-1001, 16:7-14 (“the processing platform 511 includes an stream processing cluster 515 (e.g., dynamic process) which may further include an S4 engine and various PEs with access to one or more data stores 517, which may store various data (e.g., user data, sensor data, historical data, etc.).” Notably, the processing elements (PEs in Figure 5 below, with yellow box added to emphasize) are shown as distinct from the data store and data feeds (in Figure 5 below, with red boxes added to emphasize), and are described as part of the processing platform that may access the data store. *Id.*



Id., Figure 5 (annotations added).

As a further example, the specification describes processing elements as encapsulated computational chains. Ex-1001, 4:31-33 (“In one embodiment, the encapsulated computational chains form the processing elements (PEs) of stream processing mechanisms.”). Computational chains, by virtue of their name, are not mere data; they are elements related to performing computations on data.

In another example, the specification refers to processing elements as examples of processors and as including executable code. Ex-1001, 12:17-18 (“remote processors (e.g., PEs 221, processing platforms 107, etc.)”), 7:36-38 (“In

one embodiment, the PE may include an executable code (e.g., binary code), which may be executing in the stream process ...”). The specification further confirms that processing elements can include at least code or parameters. *E.g.*, Ex-1001, 17:17-24 (“In one embodiment, processing results from the static processing mechanism 603 are transferred back to the dynamic processing mechanism 601 as updated PEs (e.g., including updated parameters, computational codes, etc.), wherein one or more configurations of the dynamic processing mechanism 601 may be updated according to the updated PEs (e.g., one or more trends, additional data, updated codes, etc.)”).

Accordingly, the intrinsic record is consistent with “processing element” being construed according to its well-established meaning (such as Petitioner’s expert’s “element that carries out processing (e.g., a processing mechanism)” (Ex-1003, ¶27)) and not Petitioner’s “data to be processed” construction of the term.

Aside from the intrinsic record, Petitioner’s Foster reference confirms the well-established meaning of “processing element” (such as Petitioner’s expert’s “element that carries out processing (e.g., a processing mechanism)” (Ex-1003, ¶27)). For example, Foster describes a “processing element” as “custom shared libraries” that can be “deploy[ed].” Ex-1004, 000063 (“Processing Elements exist as custom shared libraries, and are loaded by PECs as required by given the Streams application definitions.”); 000089 (“The application should be segmented to produce suitably sized deployable processing elements for the Streams Scheduler to

deploy.”). As yet another example, Foster also describes a processing element as “the process running the database adapter’s code.” Ex-1004, 000375 (“In particular, some of these environment variables need to be set in the process running the database adapter’s code, that is, in the corresponding Processing Element (PE).”). Accordingly, Petitioner’s Foster reference confirms that “processing elements” are not mere data to be processed, but include code or parameters or another element that carries out processing.

C. Petitioner Fails to Show a Plain-Meaning Exception is Met

As noted in Section III.A above, there are only two exceptions to the plain-meaning rule: “1) when a patentee sets out a definition and acts as his own lexicographer, or 2) when the patentee disavows the full scope of a claim term either in the specification or during prosecution.” *Thorner*, 669 F.3d at 1365 (internal citations omitted). Without even mentioning the “exacting” standard it must meet, Petitioner spends just two paragraphs spanning four sentences arguing for a construction of “processing element” that deviates from the term’s well-known meaning *Id.*; Pet., 19. Petitioner’s argument falls short.

Petitioner first argues that “[i]ndependent claim 1 uses processing element (PE) in the sense of data to be processed.” Pet. 19. Not so. Rather, as explained in Section III.B above, independent claim [1a] (and similar independent claim [11b]) reinforce that “processing element” cannot be mere data. The claim describes the

processing element as being a part of the dynamic processing mechanism, which even Petitioner agrees is a mechanism that performs processing, not data that is processed. The dependent claims confirm this reading of the independent claims. *See* Section III.B, *supra*. Moreover, Petitioner seems to suggest that the “marshalling” aspect of independent claim 1 requires a “processing element” to be data, but marshalling is not unique to data, and it was well-known to marshal components used for processing (*e.g.*, models or code). *See* Section II.B.3, *supra*.

Petitioner also points to a handful of citations from the '693 Patent specification to argue that “processing elements” can be data alone. Again, Petitioner is mistaken. For example, Petitioner says that “[t]he specification explains that PEs are destined to be processed” and cites to 12:32-48 of the '693 Patent. Consider that passage in its entirety:

Further, the map-reduce process 205 may utilize a process including extraction 215 (E), transformation 217 (T), and loading 219 (L); wherein the extraction process includes extracting data from various data sources and the transformation stage includes series of rules or functions applied to the extracted data from the various sources in order to derive data for the loading stage, wherein the data is loaded into an end target (*e.g.*, data warehouse, service provider, intermediate application, etc.). In one embodiment, the one or more PEs may be extracted from the dynamic processing engine and sent to the static processing 201 (*e.g.*, extract, transform, and load (ETL) process) as data

objects for processing, shaping (e.g., creation of hierarchical or nested sets of record sets), changing, and/or adjusting, and after the processes, the PEs are sent back to the dynamic process 203, where the real-time PEs may be correlated with the static (e.g., historical) data.

Ex-1001, 12:32-48. Petitioner misinterprets this passage. Nothing in this passage states that a “processing element” is equivalent to “data to be processed.” Instead, this passage first describes the map-reduce process, which may be used to extract, transform, and load (ETL) data. Separately, the passage describes that one or more processing elements from the dynamic processing engine may be extracted, and—after the processing elements are converted into data objects—sent to, *e.g.*, an ETL process and processed by the static processor. This is consistent with the marshalling of objects discussed in Section II.B.3 above and does not require that a “processing element” be “data to be processed.”

Petitioner further argues that “[t]he ’693 patent does not limit a PE to any specific type of data,” and points to specification language stating that “processing elements” “may include computational code and/or data” and that “a PE may include data.” Pet., 19 (quoting Ex-1001, 6:63-7:1, 7:1-16). Again, context matters. For example, the ’693 Patent specification explains:

For example, 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 (e.g., a Twitter® feed, location based

information etc.), wherein the PEs may indicate a most common topic, location, hash-tag, and the like. In another example, PE may include data as well as executing code where one or more functions of a process may be defined (e.g., process a counter, determine occurrence of certain data patterns or popularity of a certain topic, traffic analysis, route planning, reverse geocoding (RGC), etc.)

Ex-1001, 6:63-7:6. Petitioner mistakenly reads this passage as equating a processing element with data to be processed. This passage instead indicates that a processing element “may include” data. Petitioner reads this language as being *exclusive* and stating that the processing element may be data *alone*. However, the plain language of these passages is *inclusive* describing how the processing element “may include” data associated with the processing mechanism. Importantly, this passage would be read within the context of the understanding of a POSITA who would appreciate that the well-known meaning of a processing element implies some form of processing mechanism. Ex-1003, ¶27.

Moreover, even if Petitioner’s reading of these citations was correct—and it is not—none of these *examples* from the specification show that the patentee “‘clearly express[ed] an intent’ to redefine the term” that would be necessary to justify a construction that deviates from the plain and ordinary meaning of “processing element.” *Thorner*, 669 F.3d at 1365 (quoting *Helmsderfer*, 527 F.3d at 1381). Indeed, Petitioner fails to even expressly argue that the patentee acted as a

lexicographer with regards to “processing element.” The broad, inclusive statements that Petitioner points to are not sufficient to establish lexicography. *K-fee Sys. GmbH v. Nespresso USA, Inc.*, 89 F.4th 915, 923 (Fed. Cir. 2023) (“To act as its own lexicographer, a patentee must clearly set forth a definition of the disputed claim term other than its plain and ordinary meaning and must clearly express an intent to redefine the term” (internal quotations and citations omitted)).

D. Aside From its Admission Regarding the Well-Known Meaning of “Processing Element,” Petitioner’s Expert’s Declaration Should be Afforded Little to No Weight

Petitioner attempts to bolster its erroneous claim construction with citations to its expert’s declaration. Pet. 19 (citing Ex-1003, ¶¶26-28). However, the cited portions of the expert’s declaration purportedly supporting Petitioner’s claim construction are mere recitals of Petitioner’s own arguments. *Compare* Pet., 19 *with* Ex-1003, ¶¶26-28. Accordingly, the Board should disregard that conclusory testimony and decline to deviate from the well-known, plain and ordinary meaning of “processing element” “as an element that carries out processing (e.g., a processing mechanism).” *Samsung/RightQuestion*, IPR2022-00251, Paper 17 at 24 (“Petitioner’s declarant also makes this same statement verbatim, without more, and with no further explanation or additional citations to evidence. ... We give such conclusory, unsupported assertions by Petitioner’s expert little or no weight.” (internal citations omitted)); *see also In re Am. Acad. of Sci. Tech Ctr.*, 367 F.3d

1359, 1368 (Fed. Cir. 2004) (“[T]he Board is entitled to weigh the declarations and conclude that the lack of factual corroboration warrants discounting the opinions expressed in the declarations.”); *Rohm & Haas Co. v. Brotech Corp.*, 127 F.3d 1089, 1092 (Fed. Cir. 1997) (“Nothing in the rules or in our jurisprudence requires the fact finder to credit the unsupported assertions of an expert witness.”); 37 C.F.R. § 42.65(a) (“Expert testimony that does not disclose the underlying facts or data on which the opinion is based is entitled to little or no weight.”).

In fact, the only substantive difference between Petitioner’s claim construction argument and the corresponding section of its expert’s declaration is that Petitioner’s expert acknowledges the well-known meaning of the “processing element” term (“an element that carries out processing (e.g., a processing mechanism)”), which the Petition omits. *Compare* Pet., 19 *with* Ex-1003, ¶27. Petitioner’s expert otherwise parrots Petitioner’s arguments for declining to apply this well-established meaning of the term. *Compare* Pet., 19 *with* Ex-1003, ¶¶26-28.

Other sections of the expert’s declaration, such as the analysis of limitations[1a]-[1e], generally parrot Petitioner’s arguments without significant additional explanation or evidence. *Compare* Pet., 23-48 *with* Ex-1003, ¶¶54-105. Accordingly, the Board should afford the expert’s declaration, including as to limitations [1a]-[1e], little to no weight. *Samsung/RightQuestion*, IPR2022-00251, Paper 17 at 24.

IV. ARGUMENT

A. Petitioner Has Not Established a Reasonable Likelihood that the Independent Claims are Unpatentable Under Ground 1

1. Foster does not disclose or render obvious claim [1a]’s “determining via a processor at least one processing element of at least one dynamic processing mechanism” (or similar language in claim [11b])

Petitioner argues that Foster’s Streams discloses claim [1a] (“determining via a processor at least one processing element of at least one dynamic processing mechanism”) “by disclosing ingesting streaming data at Streams which is determined to be sent to BigInsights.” Pet., 23. Specific to “processing element,” Petitioner makes clear that it equates “processing element” with “data” when it asserts that “[t]he 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.” Pet., 24; *see also* Pet., 26 (“Foster teaches the PE (data)”). Petitioner then focuses on the “and/or data” language discussed in Section III.C above (Pet., 24), but data alone does not read on the “processing element” of the ’693 Patent. *See* Section III.C, *supra*. Notably, Petitioner nowhere argues that Foster discloses this limitation under the proper construction of “processing element” (such as Petitioner’s expert’s “element that carries out processing (e.g., a processing mechanism)”).

Petitioner’s analysis for another aspect of claim [1a]—that Foster teaches “determining via a processor”—highlights Petitioner’s errant use of Foster. Pet., 26-27. Recall that Foster confirms that “processing element” should be given its well-known meaning in light of Foster’s clear disclosures that a “processing element” includes code or another computational element. *See* Section III.B, *supra*. For claim [1a]’s “determining via a processor,” Petitioner’s cited Figure 2-3 (captioned “Streams Instance, hosts, and more”) depicts “PE” as an application (such as code) running on application hosts. But Petitioner does not point to this PE application as the claimed “processing element” in claim [1a]. Petitioner’s decision to point to mere data and ignore Foster’s PE application running on application host for the claimed “processing element” makes sense given that there is no evidence that Foster ever causes the marshalling of its PE application (*e.g.*, there is no disclosure of this PE application being transferred to the static processing mechanism (BigInsight)) as claim [1b], discussed next, requires. Petitioner’s Foster-based theory is thus broken.

Accordingly, Petitioner fails to show that Foster discloses the claimed “processing element”—and claim [1a] that the term is a part of.

Independent claim [11b] includes analogous language to claim [1a] (*i.e.*, “determine at least one processing element of at least one dynamic processing mechanism”). Petitioner also relies on its claim [1a] analysis for claim [11b]. Pet., 52-53 (“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].”). Accordingly, Foster fails to teach or render obvious claim [11b] for the same reasons discussed as to claim [1a].

**2. Foster does not disclose or render obvious claim [1b]’s
“causing, at least in part, a marshalling via the processor of the
at least one processing element as at least one data object” (or
similar language in claim [11c])**

Petitioner argues that Foster discloses claim [1b] (“causing, at least in part, a marshalling via the processor of the at least one processing element as at least one data object”) “by ingesting data to BigInsights, including preparing the data and transforming it to a format used by BigInsights (e.g., JSON).” Pet., 28. However, as discussed for claims [1a] and [11b] above, data from Streams does not correspond to the “processing element” of the claim. Accordingly, even if what Petitioner describes from Foster can be considered marshalling, this would not be “marshalling...*of the at least one processing element.*” For example, Petitioner points to data that is represented by tuples in Streams and converted to a JSON format that is input into BigInsights. Pet., 30. However, data stored in tuples is not a “processing element” for all of the reasons discussed in Section III.B & III.C above. As another example, Petitioner points to sink operators used by Streams to allegedly “marshal data into other output formats.” Pet., 30-31. Even assuming that this is an accurate characterization of what the sink operators in Streams are doing, this does

not read on the claim limitation because marshalling data is not the same thing as marshalling “at least one processing element.” For this reason alone, Foster fails to disclose this limitation or otherwise render it obvious.

Furthermore, Petitioner’s Foster-only theory for this limitation is unclear and appears to skip the proper analysis under § 103. On the one hand, Petitioner does not advance any § 102 grounds for rejection. Pet., 14, 22. However, Petitioner does not appear to make any obviousness argument regarding Foster alone. Instead, Petitioner seems to argue that Foster inherently discloses all limitations of the independent claims without any modification. For example, Petitioner argues that a POSITA would have understood the sink operators of Streams “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.” Pet., 32. Petitioner further argues that sending data between Streams and BigInsights over a TCP network connection inherently discloses marshaling the data beforehand. Pet., 33.

Petitioner appears to be conflating analysis of inherency with analysis for obviousness. However, the legal standard for each is different. To show inherency, Petitioner must establish that the limitation is necessarily present in the prior art reference. *In re Omeprazole Pat. Litig.*, 483 F.3d 1364, 1378 (Fed. Cir. 2007) (explaining that “anticipation by inherent disclosure is appropriate only when the

reference discloses prior art that must necessarily include the unstated limitation, [or the reference] cannot inherently anticipate the claims”). To show obviousness, Petitioner must walk through the analysis articulated in *Graham* and *KSR*, including explaining what is missing in the prior art and how it would have been obvious to modify this reference to arrive at the claimed limitation. See *Prometheus Lab'ys, Inc. v. Roxane Lab'ys, Inc.*, 805 F.3d 1092, 1097-98 (Fed. Cir. 2015) (“Obviousness is based on underlying factual findings, including: (1) the level of ordinary skill in the art; (2) the scope and content of the prior art; (3) the differences between the claims and the prior art; and (4) secondary considerations of nonobviousness, such as commercial success, long-felt but unmet needs, failure of others, and unexpected results.” (citing *KSR Int’l Co. v. Teleflex, Inc.*, 550 U.S. 398, 406 (2007); *Graham v. John Deere Co.*, 383 U.S. 1, 17-18 (1966))). Petitioner walks through this analysis in connection with ground 2 (Foster + Williams, see Pet., 35-40), but fails to do so in connection with ground 1 (Foster alone), which is also allegedly being made under § 103. Pet., 14.

Petitioner’s analysis of claim [1b] thus fails to meet the requirements to establish a *prima facie* case of obviousness, and independent of Petitioner’s erroneous understanding of the processing element, does not evince a likelihood of showing that the Challenged Claims are unpatentable.

Independent claim [11c] includes analogous language to claim [1b] (*i.e.*, “cause a marshalling via the processor of the at least one processing element as at least one data object”). Petitioner also relies on its claim [1b] analysis for claim [11a]. Pet., 52-53 (“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].”). Accordingly, Foster fails to teach or render obvious claim [11c] for the same reasons discussed as to claim [1b].

3. Foster does not disclose the “data object” found in claims [1c], [1d], and [1e] (or similar language in claims [11d], [11e], and [11f])

Petitioner further fails to show that Foster discloses claim [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”) because Foster fails to teach or render obvious the “data object” that each of these limitations refers to.

Petitioner argues that data transferred from Streams to BigInsights corresponds to the data object of the claims. Pet., 40. However, the claims specify that the data object is formed by “a marshalling via the processor of the at least one processor element.” Ex-1001, 25:36-38. Petitioner does not point to any data object

that is formed by such a process. Streams is only configured to transfer *data* to BigInsights. *See e.g.*, Ex-1005, 000339. Streams is not configured to transfer any “processing elements” to BigInsights. Petitioner’s analysis of this limitation is thus marred by its erroneous construction of a “processing element.”

For example, Petitioner alleges that “Streams marshals the **input data** by processing it in various methods to convert its memory representation into a form that BigInsights can use.” Pet. 28 (emphasis added). As another example, Petitioner alleges that “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.” Pet., 43 (emphasis added). Without needing to dispute any of Petitioner’s characterizations of how Streams operates, this cannot read on the claim language because any data object resulting from this process would not be the data object of the claims, as it would not be the result of marshalling at least one “processing element.”

That Foster fails to teach any processing element being transferred from Streams to BigInsights in the form of a data object is further demonstrated by the fact that Petitioner does not raise any arguments with regards to claims 6 or 16. These claims require “wherein the updating of the at least one data object comprises, at least in part, causing a shaping of the one or more computational chains based, at least in part, on the one or more triggering events, the one or more contextual trends,

or a combination thereof.” Claims 6 and 16 make clear that a processing element is “an element that carries out processing,” and, at least in this embodiment, these processing elements take the forms of “computational chains” that are shaped by the “static processing mechanism.” Petitioner fails to challenge these claims because Foster fails to disclose or otherwise suggest any computational chain or other “element that carries out processing” being transferred from Streams to BigInsights. Accordingly, when a processing element is properly understood according to its well-known meaning as “an element that carries out processing,” Petitioner also recognizes that Foster does not read on the Challenged Claims.

Independent claims [11d], [11e], and [11f] include analogous language to claims [1c], [1d], and [1e] (*i.e.*, “transfer of the at least one data object to at least one static processing mechanism”; “process the at least one data object by the at least one static processing mechanism”; and “after the process, transfer the at least one data object back to the at least one dynamic processing mechanism”). Petitioner also relies on its claims [1c], [1d], and [1e] analysis for claims [11d], [11e], and [11f]. Pet., 52-53 (“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].”). Accordingly, Foster fails to teach or render obvious claims [11d], [11e], and [11f] for the same reasons discussed as to claims [1c], [1d], and [1e].

B. Petitioner Has Not Established a Reasonable Likelihood That Any of the Dependent Claims are Unpatentable Under Ground 1

As discussed in section IV.A, Petitioner has not established a reasonable likelihood that independent claims 1 or 11 are unpatentable in light of ground 1. Each of the dependent claims depends on either claim 1 or 11. Accordingly, Foster does not render the remaining claims obvious, for at least the same reasons discussed in Section IV.A. *See In re Fine*, 837 F.2d 1071, 1076 (Fed. Cir. 1988) (stating that “[d]ependent claims are nonobvious under section 103 if the independent claims from which they depend are nonobvious”).

C. Petitioner Has Not Established a Reasonable Likelihood that the Independent Claims are Unpatentable Under Ground 2

The only difference between Petitioner’s first and second grounds is that Petitioner relies on Williams (Ex-1005) as allegedly suggesting the limitation of “causing, at least in part, a marshalling via the processor of the at least one processing element as at least one data object” in its second ground. Pet., 33. In particular, Petitioner cites to Williams as generally disclosing the concepts of marshalling and unmarshalling. Pet., 33. Notably, Petitioner’s analysis cites to only a single page from a single section of the 594-page Williams book. Pet., 33 (citing Ex-1005, 00039 (§1.4.3)). This sole section relied upon by Petitioner is shown below:

1.4.3 Encodings

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*.

When implementing a Web service, we have to choose one of the tools and programming or scripting languages that support the Web services model. However, the protocol format for Web services is XML, which is independent of the programming language. Thus, SOAP encodings tell the SOAP runtime environment how to translate from data structures constructed in a specific programming language into SOAP XML and vice versa.

The following encodings are defined:

SOAP encoding The *SOAP encoding* enables marshalling/unmarshalling of values of data types from the SOAP data model. This encoding is defined in the SOAP 1.1 standard.

Literal The *literal* encoding is a simple XML message that does not carry encoding information. Usually, an XML Schema describes the format and data types of the XML message.

Ex-1005, 000039.

This Williams passage merely describes marshalling at a high-level of generality, and does not disclose or otherwise suggest marshalling of an object such as code that is capable of performing processing (as opposed to marshalling of mere data). Importantly, this passage does nothing to address the shortcomings in the Foster reference identified in Section IV.A above. Accordingly, Petitioner's ground 2 fails to establish any reasonable likelihood of the independent claims being unpatentable for the same reasons described in connection with ground 1.

D. Petitioner Has Not Established a Reasonable Likelihood That Any of the Dependent Claims are Unpatentable Under Ground 2

As discussed in section IV.C, Petitioner has not established a reasonable likelihood that independent claims 1 or 11 are unpatentable in light of ground 2. Each of the dependent claims depends on either claim 1 or 11. Accordingly, Foster in combination with Williams does not render the remaining claims obvious for at

least the same reasons discussed in section IV.C. *See In re Fine*, 837 F.2d at 1076 (stating that “[d]ependent claims are nonobvious under section 103 if the independent claims from which they depend are nonobvious”).

V. CONCLUSION

For the reasons detailed above, Petitioner has failed to establish any reasonable likelihood of the Challenged Claims being unpatentable, and Patent Owner respectfully requests that the Board deny institution.

Dated: January 30, 2025

Respectfully submitted,

/s/Scott W. Hejny

Scott W. Hejny

Reg. No. 45,882

shejny@mckoolsmith.com

Eric S. Hansen

Reg. No. 63095

ehansen@mckoolsmith.com

McKool Smith, P.C.

300 Crescent Court, Suite 1200

Dallas, TX 75201

Telephone: (214) 978-4000

Facsimile: (214) 978-4044

*Counsel for Patent Owner, Nokia
Technologies Oy*

CERTIFICATION OF WORD COUNT

The undersigned hereby certifies that the portions of the above-captioned **PATENT OWNER'S PRELIMINARY RESPONSE** specified in 37 C.F.R. §42.24 has 8,789 words, in compliance with the 14,000 word limit set forth in 37 C.F.R. §42.24(b)(1). This word count was prepared using Microsoft Word 2010.

Dated: January 30, 2025

Respectfully submitted,

/s/Scott W. Hejny
Scott W. Hejny
Reg. No. 45,882

CERTIFICATION OF SERVICE

The undersigned hereby certifies that a copy of this **PATENT OWNER'S PRELIMINARY RESPONSE** pursuant to 37 C.F.R. § 42.10(b) has been served on Petitioner via email on the following counsel of record for Petitioner:

Harper Batts
hbatts@sheppardmullin.com

Chris Ponder
cponder@sheppardmullin.com

Jeffrey Liang
jliang@sheppardmullin.com

LegalTM-Amazon-Nokia-PRs@sheppardmullin.com

Dated: January 30, 2025

Respectfully submitted,

/s/Scott W. Hejny
Scott W. Hejny
Reg. No. 45,882