**IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
TYLER DIVISION**

DATA ENGINE TECHNOLOGIES LLC,	§	
	§	
Plaintiff,	§	**CAUSE NO. 6:13-CV-860-RWS-JDL**
	§	
vs.	§	
	§	
INTERNATIONAL BUSINESS MACHINES CORP.,	§	
	§	
	§	
Defendant.	§	

MEMORANDUM OPINION AND ORDER

This Memorandum Opinion construes the disputed claim terms in U.S. Patent Nos. 6,247,020 ("the ’020 Patent"), 6,237,135 ("the ’135 Patent"), 6,851,107 ("the ’107 Patent"), 6,976,243 ("the ’243 Patent"), and 7,051,316 ("the ’316 Patent") (collectively, "the patents-in-suit"). On April 23, 2015, the parties presented arguments on the disputed claim terms at a *Markman* hearing. For the reasons stated herein, the Court adopts the constructions set forth below.

BACKGROUND

Plaintiff Data Engine Technologies LLC ("Data Engine") alleges that Defendant International Business Machines Corporation ("IBM") infringes the five patents-in-suit owned by Data Engine. The ’020 Patent is directed to providing synchronization between screen panel displays of the user interface of a software development system. The ’135 Patent is directed to automatic source code generation conforming to selected design patterns, where the generated source code is subject to post-generation editing. The ’107 Patent is directed to a software development tool which allows a programmer to simultaneously view a textual display of source code and a graphical display of source code. The ’243 Patent is directed to an ability to search within a Unified Modeling Language to isolate a portion of the software program that is being visualized. The ’316 Patent is directed to a software development tool that generates code for a computing component to be deployed onto a server for client-server interactions.

APPLICABLE LAW

“It is a ‘bedrock principle’ of patent law that ‘the claims of a patent define the invention to which the patentee is entitled the right to exclude.’” *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (en banc) (quoting *Innova/Pure Water Inc. v. Safari Water Filtration Sys., Inc.*, 381 F.3d 1111, 1115 (Fed. Cir. 2004)). In claim construction, courts examine the patent’s intrinsic evidence to define the patented invention’s scope. *See id.*; *C.R. Bard, Inc. v. U.S. Surgical Corp.*, 388 F.3d 858, 861 (Fed. Cir. 2004); *Bell Atl. Network Servs., Inc. v. Covad Commc’ns Group, Inc.*, 262 F.3d 1258, 1267 (Fed. Cir. 2001). This intrinsic evidence includes the claims themselves, the specification, and the prosecution history. *See Phillips*, 415 F.3d at 1314; *C.R. Bard, Inc.*, 388 F.3d at 861. Courts give claim terms their ordinary and accustomed meaning as understood by one of ordinary skill in the art at the time of the invention in the context of the entire patent. *Phillips*, 415 F.3d at 1312–13; *Alloc, Inc. v. Int’l Trade Comm’n*, 342 F.3d 1361, 1368 (Fed. Cir. 2003).

The claims themselves provide substantial guidance in determining the meaning of particular claim terms. *Phillips*, 415 F.3d at 1314. First, a term’s context in the asserted claim can be very instructive. *Id.* Other asserted or unasserted claims can also aid in determining the claim’s meaning because claim terms are typically used consistently throughout the patent. *Id.* Differences among the claim terms can also assist in understanding a term’s meaning. *Id.* For example, when a dependent claim adds a limitation to an independent claim, it is presumed that the independent claim does not include the limitation. *Id.* at 1314–15.

"[C]laims 'must be read in view of the specification, of which they are a part.'" *Id.* (quoting *Markman v. Westview Instruments, Inc.*, 52 F.3d 967, 979 (Fed. Cir. 1995) (en banc)). "[T]he specification 'is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.'" *Id.* (quoting *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)); *see also Teleflex, Inc. v. Ficosa N. Am. Corp.*, 299 F.3d 1313, 1325 (Fed. Cir. 2002). This is true because a patentee may define his own terms, give a claim term a different meaning than the term would otherwise possess, or disclaim or disavow the claim scope. *Phillips*, 415 F.3d at 1316. In these situations, the inventor's lexicography governs. *Id.*

The specification may also resolve ambiguous claim terms "where the ordinary and accustomed meaning of the words used in the claims lack sufficient clarity to permit the scope of the claim to be ascertained from the words alone." *Teleflex, Inc.*, 299 F.3d at 1325. But, "'[a]lthough the specification may aid the court in interpreting the meaning of disputed claim language, particular embodiments and examples appearing in the specification will not generally be read into the claims.'" *Comark Commc'ns, Inc. v. Harris Corp.,* 156 F.3d 1182, 1187 (Fed. Cir. 1998) (quoting *Constant v. Advanced Micro-Devices, Inc.*, 848 F.2d 1560, 1571 (Fed. Cir. 1988)); *see also Phillips*, 415 F.3d at 1323. The prosecution history is another tool to supply the proper context for claim construction because a patent applicant may also define a term in prosecuting the patent. *Home Diagnostics, Inc., v. Lifescan, Inc.*, 381 F.3d 1352, 1356 (Fed. Cir. 2004) ("As in the case of the specification, a patent applicant may define a term in prosecuting a patent.").

Although extrinsic evidence can be useful, it is "'less significant than the intrinsic record in determining the legally operative meaning of claim language.'" *Phillips*, 415 F.3d at 1317 (quoting *C.R. Bard, Inc.*, 388 F.3d at 862). Technical dictionaries and treatises may help a court understand the underlying technology and the manner in which one skilled in the art might use claim terms, but technical dictionaries and treatises may provide definitions that are too broad or may not be indicative of how the term is used in the patent. *Id.* at 1318. Similarly, expert testimony may aid a court in understanding the underlying technology and determining the particular meaning of a term in the pertinent field, but an expert's conclusory, unsupported assertions as to a term's definition are entirely unhelpful to a court. *Id.* Generally, extrinsic evidence is "less reliable than the patent and its prosecution history in determining how to read claim terms." *Id.* In cases where subsidiary facts, such as the background science or the meaning of the term in the relevant art, are in dispute, "courts will need to make subsidiary factual findings about that extrinsic evidence." *Teva Pharm. USA, Inc. v. Sandoz, Inc.*, 135 S. Ct. 831, 841 (2015). The "evidentiary underpinnings" and "subsidiary factfinding" of claim construction are reviewed for clear error on appeal. *Id.*

AGREED CLAIM TERMS

The parties agree to the construction of the following terms:

Claim Term	Agreed Construction
design pattern(s)	certain naming conventions for properties, methods, and events
synchronizing	to update the display of respective information in panes based on the occurrence of a user event in one of the navigation pane, the structure pane, or the content pane such that all panes show corresponding information
suitable for modification	modifiable

instructions	compiled source code
type of link	relationship between elements
data processing system	Plain meaning.
distributed computing component	a software component that runs on a computer and is designed to perform business logic for client application(s) requiring a solution to a business problem
deployment descriptor file	a file for describing an enterprise java bean and any runtime properties of the EJB to the EJB application server where the EJB is to be deployed and run
deployment information	information describing the enterprise java bean and any runtime properties of the EJB to the EJB application server where the EJB is to be deployed and run
implementation class	the class which implements all methods defined in the remote interface

Docket Nos. 72, 74.

DISPUTED CLAIM TERMS

A. "automatically synchronized"

Data Engine's Proposed Construction	IBM's Proposed Construction
Plain and ordinary meaning. No construction necessary.	"synchronized without user intervention"

Claim 1 of the '020 Patent contains the term "automatically synchronized." Data Engine argues that no construction is necessary because "'automatically' does not have any specialized meaning as used in the patent and is easily understood based on its plain and ordinary meaning." Docket No. 67 at 4. Data Engine contends that "automatically synchronized" merely means "that synchronization is not performed manually." *Id.* at 5. Data Engine argues that IBM's proposal is overly broad because the user "plays a role in triggering automatic synchronization." *Id.* at 5–6. IBM responds that the patentee made a disclaimer during prosecution to traverse the *Leshem* reference. Docket No. 69 at 2–3. According to IBM, the *Leshem* reference taught synchronization "through manual intervention—by clicking on individual icons." *Id.* at 3 (emphasis omitted). Thus, IBM argues, the act of synchronization cannot involve user interaction. *Id.*

Figure 4B of the specification illustrates synchronization among the different panes of the browser. '020 Patent fig.4B; *id.* at col.3 ll.66–67. The patentee's statements regarding the *Leshem* reference were not the unequivocal disclaimer of claim scope that IBM proposes. The patentee merely distinguished the system in *Leshem* which relied on a user manually synchronizing content in a separate program based on the patentee's amended claim limitation. The patentee did so not only on the basis of adding "automatically," but also by adding "by the system." A construction demanding synchronization "without user intervention" would therefore run the risk of reading out preferred embodiments disclosed in the specification. *See id.*

At the hearing, the Court proposed a construction clarifying that synchronization "is not performed manually." Tr. Apr. 23, 2015, Docket No. 82 ("Hearing Transcript") at 53:4–6. IBM expressed concern about the meaning of "manually" in the context of computers, and Data Engine re-urged that such a construction would read out a user's interaction to begin synchronization. *Id.* at 53:10–19, 54:9–55:13. In light of the parties' inability to reach an agreement, and having rejected IBM's argument above, the Court finds that the term "automatically synchronized" needs no construction.

B. “structural information about Java code”

Data Engine’s Proposed Construction	IBM’s Proposed Construction
“information regarding the classes interfaces, ancestor classes, variables and methods of the java code” Alternatively, plain and ordinary meaning. No construction necessary.	“information about how designable objects are nested and interrelated in a java file”

Claims 13–15, 17, 30, and 31 of the ’020 Patent contain the term “structural information about Java code.” Both parties rely on a portion of the specification in support of their proposals:

> D. Structure pane
> The Structure pane of the AppBrowser shows a structural analysis of the file that the user has selected in the Navigation pane. When the user has selected a java file and then selects the Design tab at the bottom of the Content pane, the Structure pane displays the designable objects in the file, and how they are nested and interrelated. For example, if one selects a java file, the Structure pane shows structural information about the java code in that file, such as
> Imported packages.
> The classes and/or interfaces in the file.
> Any ancestor classes and/or interfaces.
> Variables and methods.

’020 Patent col.11 ll.13–24. Data Engine argues that its proposal properly accounts for “what the structural information is,” as opposed to “what the structure pane displays.” Docket No. 67 at 7. IBM responds that Data Engine’s proposal improperly recites a subset of examples provided in the specification. Docket No. 69 at 6. IBM contends that its proposal more clearly describes the “structural information” rather than limiting the construction to certain examples. *Id.*

At the hearing, the parties reached an agreement that “structural information about Java code” is not limited to merely the examples recited in the specification. Hearing Transcript at 65:22–66:2, 66:5–7. The parties further agreed that no construction is necessary. *Id.* at 66:5–15.

C. “Java bean component”

Data Engine’s Proposed Construction	IBM’s Proposed Construction
“a collection of one or more Java classes that has a constructor with no parameters, that is often bundled into a single JAR (Java Archive) file, that serves as a self-contained reusable component, and that usually has properties, methods and events which follow certain naming conventions”	“a collection of one or more Java classes, often bundled into a single JAR (Java Archive) file that serves as a self-contained reusable component”

Claims 6, 15, 16, 23, 31, and 38 of the ’135 Patent contain the term “Java bean component.” The parties rely on the same portion of the specification in support of their proposals:

> At the outset, it is helpful to briefly explain what a “Java Bean” (also “JavaBean” or simply, “bean”) is. A Java Bean is a collection of one or more Java classes, often bundled into a single JAR (Java Archive) file, that serves as a self-contained, reusable component. A Java Bean can be a discrete component used in building a user interface, or a non-UI component such as a data module or computation engine. At its simplest, a Java Bean is a public Java class that has a constructor with no parameters. Java Beans usually have properties, methods, and events that follow certain naming conventions (also known as “design patterns”).

’135 Patent col.10 ll.23–33. Data Engine argues that “[t]here is no justification for IBM’s selective omission of two-thirds of the patentee’s definition.” Docket No. 67 at 13. Data Engine also contends that IBM’s definition would allow “‘java bean components’ that lack a constructor with no parameters, even though this is a requirement for such components.” *Id.* IBM responds that Data Engine’s proposal includes optional elements of a Java bean component and “does not include [the] purported requirement” that a Java bean component include a constructor with no parameters. Docket No. 69 at 13.

Both parties' proposals include optional elements of a Java bean component and extend beyond the definition provided in the specification. At the hearing, the Court proposed "a collection of one or more Java classes that has a constructor with no parameters and serves as a self-contained reusable component." Hearing Transcript at 74:20–22, 76:3–5. Data Engine and IBM agreed to this proposal. *Id.* at 76:9–15. Accordingly, the Court construes **"Java bean component"** as **"a collection of one or more Java classes that has a constructor with no parameters and serves as a self-contained reusable component."**

D. "parsing said emitted source code"

Data Engine's Proposed Construction	IBM's Proposed Construction
"to analyze or separate the emitted source code into more easily processed components"	"to analyze the emitted source code in order to uncover properties, events, and methods"

Claim 1 of the '135 Patent contains the term "parsing said emitted source code" and claim 19 of the '135 Patent contains the term "parsing the emitted source code." Data Engine argues that its proposal avoids technical language and will thus be more helpful to a lay jury than IBM's proposal. Docket No. 67 at 16. Data Engine also criticizes IBM's proposal because parsing itself need "not, by definition, yield 'a list of properties, events, and methods.'" *Id.* at 14–15. Instead, Data Engine, argues, yielding such a list is one result of "parsing." *Id.* In response, IBM emphasizes that "the patented system 'provide[s] a list of properties, events, and methods,'" the underlying source of which is "parsed code." Docket No. 69 at 14 (emphasis omitted). IBM also argues that its proposal "directly corresponds to the inventors' description in the patent." *Id.* at 15 (citing Docket No. 69-4, Ex. C ('135 Patent File History) at 14).

The parties' proposed constructions improperly define the term "parsing said emitted source code" using a result that follows the actual act of parsing. The purpose of parsing—to eventually uncover properties, events, and methods—need not be part of a construction that describes what constitutes parsing. Instead, the specification and claim language make clear that the act of parsing source code involves separating its constituent parts so that they can be isolated for use in separate processing operations. ’135 Patent col.20 ll.22–24; *id.* at col.39 ll.56–60. The construction should avoid conflating the act of “parsing” with the functionality of the code generator. *Id.* at col.20 ll.22–24 (“The *code generator* parses the source code such that it can enumerate this information from source code.”) (emphasis added). Accordingly, the Court construes **“parsing said emitted source code”** and **“parsing the emitted source code”** as **“separating the emitted source code.”**

E. “synchronized so that a modification in one is automatically reflected in the other”

Data Engine’s Proposed Construction	**IBM’s Proposed Construction**
“automatically updating the graphical representation of the source code to reflect changes to the textual representation of the source code or updating the textual representation of the source code to reflect changes to the graphical representation of the source code with no repository no batch code generation and no risk of losing code”	“simultaneously, without user intervention, reflecting any modifications to the source code in both the display of the graphical representation as well as the textual display of the source code, with no repository, no batch code generation, and no risk of losing code”

Claims 1, 8, 15, 22, 29, 31, 36, 43, 50, 57, 63, and 69 of the ’107 Patent contain the term “synchronized so that a modification in one is automatically reflected in the other.” The parties first dispute whether the term “automatically” means “simultaneously, without user interaction.” Data Engine argues that “automatic” requires no construction. Docket No. 67 at 17. Data Engine also contends that IBM’s proposal “is invented wholesale by IBM” and conflicts with the patent’s teaching that “the system waits for certain events” before “automatic synchronization is triggered.” *Id.* at 17–19. IBM responds that the synchronization must be “simultaneous” and “without user intervention.” Docket No. 69 at 17. In support, IBM relies on the patentee’s statement during prosecution that certain “amended claims provide for an improved software development tool . . . that allows a developer to *simultaneously view* a graphical and a textual display of source code." *Id.* at 18 (quoting Docket No. 69-5, Ex. D ('107 Patent File History) at 17–18).

The term "automatically" as used here does not equate to "simultaneously." In making this proposal, IBM relies on the fact that a developer may "simultaneously view a graphical and a textual display of source code." However, the word "simultaneously" within that statement does not modify "automatically," but instead merely describes how a developer may view graphical and textual representations of source code *at the same time*. Further, as discussed above with respect to the term "automatically synchronized," synchronization can be automatic despite user interaction in the moments preceding synchronization. Finally, IBM's proposal would place a timing requirement on the system that is not supported by the specification. "Simultaneously" reflecting changes in either the graphical or textual displays of source code would by definition require continuous updating with every keystroke by a user. This regime would be problematic, for example, when a user enters a single character into the textual representation of source code, giving the graphical display no indication of what object or method that character will eventually become. *See* Docket No. 67 at 18 n.6. Thus, the Court rejects IBM's argument that the construction should require modifications to be reflected "simultaneously" and "without user intervention."

Second, the parties dispute whether the synchronization must be "one-way" or "two-way." Data Engine argues that the claim term "so that a modification in one is automatically reflected in the other" does not "mean that 'any modification' must be reflected in 'both.'" *Id.* at 19. According to Data Engine, "[w]hile the patentee's preferred embodiment discusses both of these capabilities, . . . the plain language of the disputed term itself contradicts [IBM's] construction: the term requires only that 'modification in one is automatically reflected in the other'—not that any modification in *either* display is reflected in the other." *Id.* (quoting '107 Patent col.18 ll.52–53). IBM responds that the patent is clear that "if one of either the source code or the graphical representation is modified, it must be reflected in the other." Docket No. 69 at 19–20 (citing '107 Patent col.4 l.42–col.5 l.2).

IBM's argument that synchronization must be "two-way" is without merit. Although the specification describes two-way synchronization between the textual and graphical representations of source code, IBM points to nothing which limits the invention to this kind of synchronization. Instead, a preferred embodiment in the specification merely includes two-way synchronization. '107 Patent col.4 l.42–col.5 l.2. Without more, such a disclosure does not support limiting this term to the two-way synchronization which IBM proposes. *See Netword, LLC v. Centraal Corp.*, 242 F.3d 1347, 1352 (Fed. Cir. 2001). Further, the claim language itself counsels against IBM's proposed limitation. The term "synchronized so that a modification in one is automatically reflected in the other" does not specify that a modification in *either* representation is reflected in *both* displays.

Accordingly, the Court construes **"synchronized so that a modification in one is automatically reflected in the other"** as **"automatically updating the graphical representation of the source code to reflect changes to the textual representation of the source code or updating the textual representation of the source code to reflect changes to the graphical representation of the source code with no repository, no batch code generation, and no risk of losing code."**

F. "computer-readable medium"

Data Engine's Proposed Construction	IBM's Proposed Construction
Plain and ordinary meaning. No construction necessary.	"transitory and non-transitory media upon which a computer can store and retrieve data, including: secondary storage devices, like hard disks, floppy disks or CD-ROM; a carrier wave from a network, such as Internet; or other forms of RAM or ROM"

Claims 76–150 of the '243 Patent contain the term "computer-readable medium" and claims 1, 10, and 19 of the '316 Patent contain the term "computer readable medium." The parties' dispute centers on whether a carrier wave from a network is a computer-readable medium. Data Engine argues that the specification and claim language make clear that the "computer-readable medium" is an article of manufacture which cannot include transitory signals like a carrier wave from a network. Docket No. 67 at 22. Data Engine criticizes IBM's proposal for "inject[ing] language . . . that is not found in the specification" and for being "directed solely to an invalidity defense." *Id.* IBM responds that "the ordinary meaning of 'computer readable medium' encompasses both transitory and non-transitory media." Docket No. 69 at 22. According to IBM, the patentee claimed a "computer readable medium" without restriction and expressly defined the term to include carrier waves from a network. *Id.* (quoting '243 Patent col.7 ll.36–42). In support of its proposal, IBM cites not only from the patent specification and prosecution history, but also from a number of administrative decisions interpreting the term "computer readable medium." *Id.* at 22–27.

As an initial matter, IBM's citation of unrelated patent applications, district court interpretations, and administrative guidance is of limited utility in assessing the instant claims. IBM points to no binding authority which construes the term "computer-readable medium" based on the same set of facts and the same disclosure. There is no "law of the case" with respect to the term, since claims must be "read in view of the specification, of which they are a part." *Phillips*, 415 F.3d at 1314–15.

The '243 Patent specification discusses a computer-readable medium containing instructions specifically in the context of an article of manufacture:

> In accordance with *articles of manufacture* consistent with the present invention, a computer-readable medium is provided. The computer-readable medium contains instructions for controlling a data processing system to perform a method.

'243 Patent col.4 ll.25–29 (emphasis added). Likewise, the claim language makes clear that the computer-readable medium must "contain instructions." Claim 94 is representative:

> A computer-readable medium *containing instructions* for controlling a data processing system to perform a method, the data processing system having source code comprising a plurality of elements, the method comprising the steps of: receiving a selection of one of the plurality of elements; receiving an indication of a distance; receiving an indication of a type of link; and determining which of the plurality of elements is connected to the selected element via a link of the indicated type and within the indicated distance.

'243 Patent col.25 ll.54–62 (emphasis added). The parties agree that "instructions" should be construed as "compiled source code." Docket No. 72-4, App. D at 2. A carrier wave from a network, while it is a medium that is computer-readable, cannot *contain* instructions under the parties' agreed construction.

IBM attempts to distinguish these portions of the specification by citing to a different passage:

> Although aspects of the present invention are described as being stored in memory, one skilled in the art will appreciate that these aspects can also be stored on or read from other types of computer-readable media, such as secondary storage devices, like hard disks, floppy disks or CD-ROM; a carrier wave from a network, such as Internet; or other forms of RAM or ROM either currently known or later developed.

’243 Patent col.7 ll.36–42. However, this passage does not support IBM’s proposal. The specification describes a data processing system having a hardware memory that stores the software development tool and is accessed by a processor. *Id.* at col.7 ll.25–35. This indicates that aspects “can be stored on *or read from* other types of computer-readable media.” One type of media identified is a carrier wave from a network. Therefore, the data processing system processor could read the software development tool instructions as they are being transmitted over a network carrier wave instead of accessing hardware memory. This description merely describes the various ways in which the processor of the data processing system can obtain the instructions for execution of the method.

Thus, the specification only clearly ties a computer-readable medium “*containing instructions for controlling a data processing system to perform a method*” to an article of manufacture. Construing the term as IBM requests would amount to rewriting the claim to require that the data processing system “obtains instructions from” the computer-readable medium, rather than simply requiring that the computer-readable medium “contains instructions” for controlling the data processing system. As written in the claims, the term “computer-readable medium” does not include a carrier wave from a network.

Having rejected IBM’s argument, the Court finds that no construction is necessary.

G. “remote interface”

Data Engine’s Proposed Construction	**IBM’s Proposed Construction**
“interface that enumerates the business methods defined in the implementation class”	“interface used to invoke the business methods defined in the bean class”

Claims 11, 14, 18, and 22 of the ’316 Patent contain the term “remote interface.” Data Engine first argues that its use of “implementation class” in the construction is preferable to “bean class” because the latter does not appear in the claims. Docket No. 67 at 27. Data Engine further argues that IBM's proposal "merely describ[es] [a remote interface] with reference to how it interacts with a browser." *Id.* at 28. IBM responds that its proposal should be adopted because it derives from the specification. Docket No. 69 at 28. IBM criticizes Data Engine's proposed construction for coming "from a book written by a third party" and being circular. *Id.* at 28–29.

IBM's proposal improperly sets forth the purpose for which the remote interface is used, rather than what constitutes a remote interface. The specification statement on which IBM relies states that "[t]he *browser* 2012 also invokes methods through a remote interface that includes signatures for the business methods of the EJB 2002." '316 Patent col.24 ll.40–46 (emphasis added). Thus, it is the browser, and not the remote interface itself, that invokes the business methods. Data Engine's proposal, which stems from a source that is incorporated by reference into the intrinsic record, better defines a remote interface. '243 Patent col.25 ll.26–40 (expressly incorporating by reference four sources); Docket No. 67-16, Ex. P at 82. Finally, contrary to IBM's assertion, Data Engine's construction is not circular because it uses the term "implementation class." Rather, a jury will more readily understand the term "implementation class" than "bean class" because the parties have agreed on a construction of the former.

Accordingly, the Court construes **"remote interface"** as **"interface that enumerates the business methods defined in the implementation class."**

CONCLUSION

For the foregoing reasons, the Court interprets the claim language in this case in the manner set forth above. For ease of reference, the Court's claim interpretations are set forth in a table in Appendix A and the parties' agreed constructions are set forth in a table in Appendix B.

So ORDERED and SIGNED this 27th day of May, 2015.

John D. Love

JOHN D. LOVE
UNITED STATES MAGISTRATE JUDGE

APPENDIX A

Claim Term	Court's Construction
automatically synchronized	No construction
structural information about Java code	No construction
Java bean component	a collection of one or more Java classes that has a constructor with no parameters and serves as a self-contained reusable component
parsing said emitted source code	separating the emitted source code
synchronized so that a modification in one is automatically reflected in the other	automatically updating the graphical representation of the source code to reflect changes to the textual representation of the source code or updating the textual representation of the source code to reflect changes to the graphical representation of the source code with no repository, no batch code generation, and no risk of losing code
computer-readable medium	No construction
remote interface	interface that enumerates the business methods defined in the implementation class

APPENDIX B

Claim Term	Agreed Construction
design pattern(s)	certain naming conventions for properties, methods, and events
synchronizing	to update the display of respective information in panes based on the occurrence of a user event in one of the navigation pane, the structure pane, or the content pane such that all panes show corresponding information
suitable for modification	modifiable
instructions	compiled source code
type of link	relationship between elements
data processing system	No construction
distributed computing component	a software component that runs on a computer and is designed to perform business logic for client application(s) requiring a solution to a business problem
deployment descriptor file	a file for describing an enterprise java bean and any runtime properties of the EJB to the EJB application server where the EJB is to be deployed and run
deployment information	information describing the enterprise java bean and any runtime properties of the EJB to the EJB application server where the EJB is to be deployed and run
implementation class	the class which implements all methods defined in the remote interface
home interface	an interface that defines methods for creating, destroying and finding instances of Enterprise Java Beans