EXHIBIT E

[illegible] AF

PATENT
Customer No. 22,852
Attorney Docket No. 7643.0042

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	
Russell T. DAVIS et al.	Group Art Unit: 2176
Application No.: 10/052,250	
Filed: January 23, 2002	Examiner: C. Nguyen
For: RDX ENHANCEMENT OF SYSTEM AND METHOD FOR IMPLEMENTING REUSABLE DATA MARKUP LANGUAGE (RDL)	Confirmation No.: 1920

Attention: Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

REPLY BRIEF

Pursuant to 37 CFR § 41.41(a)(1), Appellants present this Reply Brief in response to the Examiner's Answer mailed on November 24, 2008.

I. Response to Examiner's Arguments in the Answer

In addition to the arguments for reversal of the outstanding final rejection provided in Appellants' Appeal Brief filed on August 28, 2008, Appellants provide the following remarks regarding the Examiner's Answer ("Answer") mailed on November 24, 2008.

Regarding the rejection of claims 62-64 under 35 U.S.C. § 103(a), the Examiner continues to assert that the syntax elements in *Krug* correspond to the claimed "software elements" (Answer at pages 18-19). The Examiner states, "the HTML document is transformed into a syntax tree representing the hierarchical relationship of the syntax elements" (Answer at page 19). This is not correct.

In *Krug*, a syntax tree parser 20 "analyses the HTML syntax structure of the search result document by recognizing the HTML tags within the document and constructing a hierarchical HTML syntax tree that represents the hierarchical relationship of the syntax elements (tags)" (col. 8, lines 23-27). *Krug* specifically teaches that the syntax elements are the "tags" within the document (col. 8, line 27). By alleging that the syntax elements in *Krug* could somehow constitute the claimed "software elements," the Examiner is asserting that the tags in *Krug* correspond to <u>both</u> the claimed "tags" <u>and</u> the claimed "software elements." Therefore, according to the Examiner's statements, *Krug* interprets tags included in the document to create tags. This is not correct.

Krug analyzes the HTML syntax structure by recognizing tags and constructs a syntax tree that represents the hierarchical relationship of the tags. Neither the tags, syntax elements, nor any other teaching in *Krug* constitutes the claimed "software elements" at least because *Krug* does not interpret "tags included in the one or more text documents to create software elements," as recited in independent claim 62. Accordingly, *Krug* also cannot teach or suggest determining "the hierarchy of the software elements within a structure representative of the one or more text documents," as further recited in claim 62.

The Examiner also continues to assert that *Hamscher* discloses the claimed "manager" that "provides for the creation of a second hierarchy of the software elements" and "provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document" (Answer at page 19). This is not correct.

According to page 17 of *Hamscher*, an XBRL instance document can be created by concatenating other XBRL instance documents. The Examiner appears to assert that an XBRL document created by concatenating other XBRL instance documents constitutes the claimed "second hierarchy of software elements." Even assuming that this newly created document could correspond to a "hierarchy of software elements," which Applicants do not concede, only <u>one</u> "hierarchy of software elements" would be created (i.e. the created XBRL document).

Both *Krug* and *Hamscher* disclose, at most, information in a <u>single</u> hierarchy (allegedly the hierarchical relationship in *Krug* and the created XBRL document in *Hamscher*). In contrast, claim 62 recites <u>both</u> the determination of a "hierarchy of <u>the software elements</u>" created by interpreting tags included in the one or more text documents" <u>and</u> "the creation of a second hierarchy of <u>the software elements</u>" (emphasis added). The cited references do not provide for <u>both</u> the determination of a "hierarchy of the software elements" <u>and</u> the creation of a "second hierarchy" of the <u>same</u> "software elements," as recited by claim 62.

Therefore, *Hamscher* does not teach or suggest the claimed "creation of a second hierarchy of the software elements." Accordingly, *Hamscher* does not teach or suggest a manager that "provides for the creation of a second hierarchy between the software elements, and provides for the restructuring of the first hierarchy and the second hierarchy into software structures corresponding to a new text document," as recited in claim 62.

As set forth above, and contrary to the assertions of the Examiner, the combination of *Krug* and *Hamscher* does not teach or suggest all elements of claim 62. In view of this mischaracterization of the references, the Office Action has neither properly determined the scope and content of the prior art nor properly ascertained the differences between the prior art and the claimed invention. Therefore, no reason has been clearly articulated as to why the claim would have been obvious to one of ordinary skill in view of the prior art and a *prima facie* case of obviousness has not been established.

Claim 62 is allowable for at least these reasons, and claims 63 and 64 are also allowable at least due to their depending from claim 62.

Regarding the rejection of claims 1-6, 11-21, 24-34, 37-46, 49-57, and 59-61 under 35 U.S.C. § 103(a), the Examiner again relies on *Hamscher* to allegedly disclose "a manager that provides for the creation of a second hierarchical relationship between the software elements and the restructuring of the first hierarchical relationship and the

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase® Smarter legal research.