

The opinion in support of the decision being entered today was not written for publication and is not binding precedent of the Board.

Paper No. 43

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte ALAN SNYDER, RODERICK J. MCCHESENEY, MARK W. HAPNER,
ARTHUR A. VAN HOFF, MAURICE BALICK, and RAPHAEL BRACHO

Appeal No. 2001-0051
Application No. 08/414,240

HEARD: March 21, 2002

Before HAIRSTON, LEVY, and BLANKENSHIP, Administrative Patent Judges.

BLANKENSHIP, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134 from the examiner's final rejection of claims 1-43, which are all the claims in the application.

We reverse.

BACKGROUND

This application returns to our jurisdiction following a remand to the examiner. The application was remanded by this panel for reasons detailed in the paper mailed March 29, 2002 (Paper No. 36). We requested that the examiner reweigh the merits of the standing rejections in light of arguments and evidence submitted by appellants in the Reply Brief. The examiner was authorized to issue a supplemental examiner's answer under 37 CFR § 1.193(b)(1) in the event that the examiner held to the view that the subject matter on appeal was unpatentable in view of the rejections applied against the claims. The examiner issued a supplemental examiner's answer, to which appellants responded in an additional paper.

Appellants' invention is directed to method and apparatus for creating and installing objects on a distributed object system. Representative claim 31 is reproduced below.

31. A distributed object suitable for use on a distributed object system, comprising:
- a) a call to a developer-written function contained in a developer-written servant class of objects; and
 - b) wrapper statements contained in a wrapper class of objects, which wrapper class has an inheritance relationship with the servant class, wherein the wrapper class inherits from the developer-written servant class of developer-written objects, the wrapper statements including services for supporting the operation of the distributed object on the distributed object system.

Appeal No. 2001-0051
Application No. 08/414,240

The examiner relies on the following references:

Moeller et al. (Moeller)	5,473,777	Dec. 5, 1995 (filed Jul. 19, 1993)
Waldo et al. (Waldo)	5,475,817	Dec. 12, 1995 (effectively filed Feb. 25, 1991)
Danforth	5,493,680	Feb. 20, 1996 (effectively filed Jul. 6, 1992)

Jeffrey M. Richter (Richter), Windows 3.1: A Developer's Guide, 2nd Edition, Ch. 2, Subclassing and Superclassing Windows, pp. 63-123, M & T Books (1992).

The Common Object Request Broker: Architecture and Specification (CORBA) Rev. 1.1, Ch. 3, The Common Object Request Broker Architecture, pp. 27-44, Ch. 9, The Basic Object Adapter, pp. 147-170, Object Management Group, Inc. (Dec. 1991).

Claims 1-16 and 19-43 stand rejected under 35 U.S.C. § 103 as being unpatentable over CORBA, Moeller, Danforth, and Richter.

Claims 17 and 18 stand rejected under 35 U.S.C. § 103 as being unpatentable over CORBA, Moeller, Danforth, Richter, and Waldo.

We refer to the Final Rejection (mailed Jul. 6, 1999), the Examiner's Answer (mailed Mar. 28, 2000), and the Supplemental Examiner's Answer (mailed Jun. 18, 2002) for a statement of the examiner's position and to the Brief (filed Dec. 21, 1999), the Reply Brief (filed Jun. 5, 2000), and the Supplemental Reply Brief (filed Aug. 26, 2002) for appellants' position with respect to the claims which stand rejected.

OPINION

The rejection of instant claim 31 is set forth on pages 4 and 5 of the Answer. Appellants contend (Reply Brief at 2) that the rejection errs in equating superclassing, as disclosed by Richter, with an “inverse inheritance” relationship. Appellants argue that an “inverse inheritance” relationship is an implicit requirement of the instant claims. According to appellants, the requirement follows from language in claim 31 regarding the wrapper class having an inheritance relationship with the servant class; in particular, that the wrapper class inherits from the developer-written servant class of developer-written objects.

Appellants rely on the definition of “inheritance” as set forth in the glossary of a text entitled “Object-Oriented Analysis and Design with Applications.” Appellants submitted a copy of the relevant section of the glossary as an attachment to the Reply Brief. The definition indicates that inheritance is a relationship among classes, wherein one class shares the structure or behavior defined in one or more other classes. The definition further indicates that a subclass inherits from one or more generalized superclasses. Appellants argue that, in view of the relevant definition, the superclassing and subclassing as disclosed by Richter have no relation to inverse inheritance. (Reply Brief at 5.)

The examiner responds that the above-noted glossary is not part of the application as filed. (Supp. Answer at 12-13.) The examiner adds that the definition of “inverse” inheritance relationships is that of appellants; only the standard or

Appeal No. 2001-0051
Application No. 08/414,240

conventional class inheritance relationships have been shown in the evidence appellants rely upon. (Id. at 13.)

The examiner further finds that Richter teaches that under window superclassing a message travels first to the superclass window procedure and then to the existing or original window procedure, referring to page 93 and Figure 2-5 of the reference. Richter is deemed to show an “inverse” relationship in superclassing, in comparison to standard subclassing (pp. 64-65), in which a message is first handled by a subclass window procedure and then by the existing or original window procedure. Under superclassing, the examiner asserts that the superclass windows inherit from the existing or original window. (Id. at 13-14.) The examiner reiterates that Richter teaches that the “standard” inheritance relationship (subclassing) and the “inverse” relationship (superclassing) are alternatives to each other, pointing to pages 63 and 98 of the reference. (Id. at 14-15.)

Appellants respond in turn that Richter relates to the WINDOWS 3.1 operating system, and does not relate to distributed object oriented programming. As such, appellants assert that Richter’s use of the terms “subclassing” and “superclassing” do not have the same meaning in the art pertaining to the instant invention. (Supp. Reply Brief at 7.)

The terms used in the claims bear a “heavy presumption” that they mean what they say and have the ordinary meaning that would be attributed to those words by persons skilled in the relevant art. Texas Digital Sys., Inc. v. Telegenix, Inc., 308 F.3d

1193, 1202, 64 USPQ2d 1812, 1817 (Fed. Cir. 2002). Dictionaries, encyclopedias, and treatises are particularly useful resources in determining the ordinary and customary meanings of claim terms. Id. at 1202, 64 USPQ2d at 1818. Indeed, these materials may be the most meaningful sources of information in better understanding both the technology and the terminology used by those skilled in the art to describe the technology. Id. at 1203, 64 USPQ2d at 1818.

We thus consider it of no moment that appellants chose not to include a definition for “inheritance” in their disclosure. Appellants have provided evidence to show the meaning that would be attributed to the term by persons skilled in the art of object-oriented analysis and design. We interpret the instant claims accordingly.

We find that Richter discloses that, under the WINDOWS 3.1 operating system, a window class may be registered whereby a procedure processes messages pertaining to all instances of windows in the class. Whenever a new window is created, the system allocates a block of memory containing information specific to that window. Richter at 63. To subclass a window, a user changes the window procedure address in the window’s memory block to point to a new window procedure. Because the address is changed in one window’s memory block, it does not affect any other windows created from the same class. All messages destined for the original window will be sent to the user’s own window procedure. A message may be stopped from being passed to the original procedure, or it may be altered before sending it. However, most messages are passed to the original procedure. The reason for subclassing is usually to alter the

behavior of a window only slightly; the default behavior for the class of window is normally performed. Id. at 63-65; Fig. 2-1.

Richter teaches that window superclassing is similar to window subclassing in that messages intended for the window procedure of the original class are routed to a different procedure that the user supplies. Superclassing alters the behavior of an existing window class, called the “base class.” When superclassing a window class, the user must register a new window class with the operating system. When a message is dispatched to the superclassed window, the operating system examines the memory block for the window and calls the superclass window procedure. After the superclass window procedure processes the message, it passes the message to the window procedure associated with the base class. Id. at 93; Fig. 2-5.

Richter further teaches that the main difference between subclassing and superclassing is that subclassing alters the behavior of an existing window, while superclassing alters the behavior of all instances of windows created from an existing window class. Id. at 97.

We are persuaded by appellants that Richter fails to teach the “inverse” inheritance relationship that the rejection attributes to the reference. The rejection asserts that a servant class inheriting from a wrapper class, and a wrapper class inheriting from a servant class “resemble” a “standard” inheritance relationship and an “inverse” inheritance relationship, respectively. (Answer at 5.) The rejection further asserts that Richter teaches a “standard” inheritance -- which the rejection equates with

subclassing as taught by Richter -- and “inverse” inheritance relationships -- which the rejection equates to superclassing as taught by Richter.

However, Richter’s description of superclassing appears to be no different in substance from the description in the instant specification of “prior art methods” distinguished by the instant claims. “[A]s illustrated in Figure 5, a developer-written object or class inherits from a base class of system support functions and services that provides the developer with access to the various system functions required for implementation of the servant object.” (Spec. at 10, ll. 15-18.) That is, Richter’s superclassing may be considered as a form of inheritance. However, the user (or developer) creates a new window class that is based on an existing, or “base,” class, making use of existing system resources. We are persuaded by appellants, as argued on pages 14 and 15 of the Supplemental Reply Brief, that any “inheritance” taught by Richter is conventional.

Even if we were to postulate agreement with the examiner’s findings regarding the Richter reference, the rejection does not otherwise appear to set forth a persuasive case for prima facie obviousness of the claimed subject matter as a whole. The rejection asserts that Richter teaches “standard” and “inverse” inheritance relationships, and alleges that these relationships “resemble” a servant class inheriting from a wrapper class and a wrapper class inheriting from a servant class. However, locating a “resemblance” in the prior art falls short of setting out the required factual foundation for

Appeal No. 2001-0051
Application No. 08/414,240

a case of obviousness, but instead is consistent with an improper hindsight reconstruction of the invention.

The rejection (Answer at 5) concludes that “[w]hen the teaching of Richter is applied to the system of COBRA 1.1 as modified [sic; the system of CORBA as modified by the further teachings of Moeller and Danforth?], it would have been obvious for the wrapper class to inherit from the servant class (‘inverse’ inheritance relationship).” The conclusion is not based on any persuasive statement as to why the artisan would have been led to modify the prior art disclosed or suggested by CORBA, Moeller, and Danforth in view of the objective disclosure of Richter. Nor is it clear what conclusion is to be drawn from the allegation in the statement of the rejection of claim 31 that Richter teaches that subclassing and superclassing are “alternative” to each other.

The rejection of independent claim 1 and independent claim 20 (Answer at 7-9 and 11) also relies on Richter for teachings relating to the claimed inheritance relationship with respect to a wrapper class and a servant class. We thus cannot sustain the rejection of claims 1, 20, or 31.

Since not all respective limitations of the independent claims have been shown as disclosed or suggested by the prior art, we do not sustain the rejection of claims 1-16 and 19-43. The rejection of dependent claims 17 and 18, which adds Waldo to the combination of CORBA, Moeller, Danforth, and Richter, fails to remedy the deficiencies

Appeal No. 2001-0051
Application No. 08/414,240

with respect to the rejection of base claim 1. We therefore do not sustain the rejection of claims 17 and 18.

CONCLUSION

The rejections of claims 1-43 under 35 U.S.C. § 103 are reversed.

REVERSED

KENNETH W. HAIRSTON)	
Administrative Patent Judge)	
)	
)	
)	
)	
)	BOARD OF PATENT
STUART S. LEVY)	APPEALS
Administrative Patent Judge)	AND
)	INTERFERENCES
)	
)	
)	
HOWARD B. BLANKENSHIP)	
Administrative Patent Judge)	

Appeal No. 2001-0051
Application No. 08/414,240

BEYER WEAVER & THOMAS, LLP
P O BOX 130
Mountain View , CA 94042-0130