

THIS OPINION WAS NOT WRITTEN FOR PUBLICATION

The opinion in support of the decision being entered today (1) was not written for publication in a law journal and (2) is not binding precedent of the Board.

Paper No. 13

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte RICHARD W. KACOR
and COREY O. SELLERS

Appeal No. 96-3301
Application 08/176,603¹

ON BRIEF

Before THOMAS, HAIRSTON, and BARRETT, Administrative Patent Judges.

BARRETT, Administrative Patent Judge.

DECISION ON APPEAL

¹ Application for patent filed January 3, 1994, entitled "Method And System For Accessing Functions Of A User Interface Environment From Processes Running Outside Of The User Interface Environment."

Appeal No. 96-3301
Application 08/176,603

This is a decision on appeal under 35 U.S.C. § 134 from the final rejection of claims 1-3 and 8-10, all of the claims pending in the application. Claims 4-7 and 11 have been canceled. The amendment received November 20, 1995 (Paper No. 11), has been entered and deemed to overcome the rejection of claims 1-3 and 8-10 under 35 U.S.C. § 112, second paragraph (Supp. Examiner's Answer, Paper No. 12, page 2).

We reverse.

The disclosed invention is directed to a method and system that allows a program to run outside of a user interface shell while accessing component portions of the user interface shell.

Claim 1, as amended in Paper No. 11, is reproduced below.

1. A method of accessing functions of a user interface environment in a computer, said user interface environment operating within an operating system of said computer, said user interface environment being structured and arranged to display a user interface on a display that is connected to said computer, said method being performed on said computer, comprising the steps of:

a) providing an object in said computer, said object operating outside of said user interface environment, but within said operating system;

b) receiving a request from said object to utilize one of said functions in said user interface environment;

c) performing said one function within said user interface environment and on said computer so as to produce a result; and

d) providing said result to said object.

Appeal No. 96-3301
Application 08/176,603

Appeal No. 96-3301
Application 08/176,603

displays a user interface, and which access occurs from outside of the user interface environment." The arguments are supported by limitations in representative claim 1, which recites a user interface environment, "said user interface environment being structured and arranged to display a user interface on a display that is connected to said computer," where an object outside of the user interface, but within the operating system, makes a request to use a function in the user interface environment, the function is performed in the user interface environment, and the result is provided to the object.

The examiner states (Examiner's Answer, page 5):

In response, examiner interprets the "user interface environment" as another object located inside the object oriented environment as described by Davidson, which object provides functions accessible by other objects located outside thereof.

We are not persuaded by the examiner's reasoning which ignores that the "user interface environment" is claimed as follows:

"said user interface environment being structured and arranged to display a user interface on a display that is connected to said computer" (claim 1); or "said user interface environment displaying a user interface on a display that is connected to said computer" (claim 8).

"Objects" are abstract entities in an object-oriented system that encapsulate all the procedures and data related to

Appeal No. 96-3301
Application 08/176,603

something. These can then be treated as a package which can be manipulated in various ways. Objects may represent hardware such as CPUs, memory, printers, disks, tape drives, and other devices; software entities such as programs, files, and semaphores; and various other entities. While a "user interface environment" can be treated as an object, not all objects are a user interface environment. The examiner does not point to where Davidson discloses an object corresponding to the claimed "user interface environment" that displays "a user interface on a display that is connected to said computer." The examiner seems to equate the server object in Davidson with the "user interface environment" and the client object with the object in step a of claim 1. However, the server object for handling a remote procedure call has not been demonstrated by the examiner to be part of a user interface environment. The example in Davidson of a daemon process for handling a shared resource, such as a printer, has no need to access the user interface environment in order to perform the printing function. Davidson deals with remote procedure calls in a distributed computer system and the examiner has not offered any explanation of why a computer would need to access a function in the user interface of another computer. The examiner's statement that Davidson describes an "environment where a (client) object may access (interface) another object

Appeal No. 96-3301
Application 08/176,603

(server) to run a function (for example, print) inside the later [sic]" (Examiner's Answer, page 3) seems to confuse an interface between client and server and a user interface which is an interface between the user and the computer via a display. Since the examiner has not shown that the server process in Davidson is located in a user interface environment the rejection of claims 1-3 and 8-10 must be reversed.

Appellants also argue (Brief, page 6):

Davidson does not teach providing an object that runs or operates within the same operating system as a user interface environment, yet runs independently of the user interface environment, as provided by claim 1. An operating system is typically not a shared resource between computers.

The examiner's response is (Examiner's Answer, page 5): "All the objects of an object oriented environment may be located inside the same operating system and operating [sic] indecently [sic] of each other." Davidson deals with implementing remote procedure calls in a distributed computer system. "A remote procedure call is a mechanism by which a program executing in a process space of one computer causes a procedure to be executed in a process space of another computer ('remote computer')." Col. 1, lines 46-49.

While we agree with the examiner that objects may be located inside the same operating system, Davidson does not appear to show the client/server objects located within the same operating

Appeal No. 96-3301
Application 08/176,603

system. Accordingly, for this additional reason the examiner has failed to show that Davidson anticipates claims 1-3 and 8-10.

Appeal No. 96-3301
Application 08/176,603

CONCLUSION

The rejection of claims 1-3 and 8-10 is reversed.

REVERSED

JAMES D. THOMAS)	
Administrative Patent Judge)	
)	
)	
)	
)	BOARD OF PATENT
KENNETH W. HAIRSTON)	APPEALS
Administrative Patent Judge)	AND
)	INTERFERENCES
)	
)	
)	
LEE E. BARRETT)	
Administrative Patent Judge)	

Appeal No. 96-3301
Application 08/176,603

Geoffrey A. Mantooth
WOFFORD, FAILS, ZOBAL & MANTOOTH
110 West Seventh, Suite 500
Fort Worth, TX 76102