Paper No. 35

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte PAUL CLARKE,
PETER JOHNSON, WILLIAM KINGSTON,
ROBIN MILES DREW, GEORGE BLACK,
and ROGER MELI

Appeal No. 2002-1812
Application 08/861,181[1]

ON BRIEF

Before KRASS, BARRETT, and DIXON, Administrative Patent Judges.

BARRETT, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134 from the final rejection of claims 1, 2, 4, 5, and 8-10. Claim 7 has been canceled. Claims 3 and 6 are objected to.

[1] Application for patent filed may 21, 1997, entitled "Method of Transferring Messages Between Computer Programs Across a Network," which is a continuation of Application 08/448,423, filed June 5, 1995, now abandoned.

We reverse.

BACKGROUND

The invention relates to a method of delivering messages between application programs.

Claim 1 is reproduced below.

> 1. A method of transactional control of message transfer across a transaction-oriented data processing network wherein a sender program is responsible for sending messages from a first node of the network and a receiver program is responsible for receiving messages at a second node of the network, the method comprising:
>
> sending messages by the sender program within a first syncpoint-manager-controlled unit of work and receiving messages by the receiver program within a second syncpoint-manager-controlled unit of work, while holding the sending and receiving operations in-doubt, uncommitted, until resolution of the first and second units of work, respectively, wherein the first syncpoint-manager-controlled unit of work and the second syncpoint-manager-controlled unit of work are logically linked so that commit processing at resolution of said units of work comprises the steps of:
>
> in response to successful receipt of the messages by the receiver program, performing the sequence of steps of committing said second unit of work, transmitting to the sender program a positive confirmation of receipt, and in response to the positive confirmation committing the first unit of work; or
>
> in response to unsuccessful receipt of the messages, performing the sequence of steps of rolling back the second unit of work, transmitting to the sender program a negative confirmation of receipt, and in response to said negative confirmation backing out the first unit of work.

The examiner relies on the following references:

Ranade	4,920,484	April 24, 1990

Ferree et al. (Ferree) 5,051,892 September 24, 1991

Britton et al. (Britton) EP 0457112A2 November 11, 1991

Jefferson, David, Virtual Time, ACM Transactions on Programming Languages and Systems, Vol. 7, No. 3, July 1985, pp. 404-425, at pp. 412-417, section 4.2 (Jefferson).

We refer to the final rejection (Paper No. 30) and the examiner's answer (Paper No. 33) (pages referred to as "EA__") for a statement of the examiner's rejection, and to the appeal brief (Paper No. 32) (pages referred to as "Br__") for a statement of appellants' arguments thereagainst.

Claims 1, 4, and 10 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Britton and Jefferson. The examiner finds that Britton teaches the claimed invention except that it does not explicitly teach sending a negative confirmation of receipt (EA5). The examiner finds that Jefferson teaches "rolling back the second unit of work (transmit an antimessage to annihilate positive and negative messages) in response to unsuccessful receipt of the messages by the receiver program (message arrives at virtual clock 162), backing out the first unit of work (rollback/unsend a message) in response to negative confirmation (negative message)" (EA5-6).

Claims 2, 5, 8, and 9 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Britton and Jefferson, further in view of Ranade and Ferree.

OPINION

Appellants argue that Britton does not disclose the specific commit sequence where "the commit of the send operation is only performed in response to a positive confirmation of the message receipt, and the positive confirmation is only transmitted when the message receipt has been committed" (Br4-5). That is, claim 1 calls for a first commit operation at the receiver, transmitting a positive confirmation, followed by a second commit operation at the sender. It is argued that in Britton the commit phase is performed separately by all the resources in response to a single commit instruction which follows a prepare phase, and "[t]here is no disclosure in Britton et al of confirmation of performance of a first commit operation being required before performing a second commit operation" (Br5).

The examiner finds (EA4-5):

> Britton teaches ... [commit processing] (a two phase commit protocol) of two logically linked local units of work including: the messages (update message sent from 56A to 56D in commit phase), commit the second unit of work (56D updates file 78D), transmit a positive confirmation of receipt (reply from 56D to 56A indicating it completed the work/request), commit the first unit of work (56A commits/updates 78A, 78B). See col. 15, line 39 - col. 16, line 34. The sequence of operation is shown in the flow of events in col. 15, line 39 - col. 16, line 34.

Since appellants and the examiner disagree on the teachings of Britton, we make the following findings based on the portions of Britton relied on by the examiner. A syncpoint architecture

including a distributed computer operating system supporting distributed and non-distributed applications is shown in Fig. 2 (col. 9, lines 20-25). A typical application environment 52A has an application 56A which can issue a syncpoint 58A (commit or backout); a single syncpoint manager (SPM) 60A; a plurality of protected resource adapters (RA) 62A and 62B which interface on behalf of application 56A with resource managers 63A, 63B, which manage resource files 78A, 78B; a recovery facility 70A for logging syncpoint managers and providing recovery for failing syncpoints; and a protected conversation adapter (PCA) 64A (col. 10, lines 16-19; col. 10, line 58 to col. 11, line 4; col. 11, lines 39-47). The syncpoint architecture protects both "resources," such as files 78A and 78B, and communication "conversations" (a special type of resource) between two applications. A "protected resource" is a resource that is subject to any form of synchronization point processing or other protective commit or back out procedure (definition at col. 92, lines 41-45). A "protected conversation" is a conversation between two applications that is subject to any form of syncpoint processing or protective commit and backout procedure (col. 10, lines 35-41; definition at col. 92, lines 35-39). Updates performed between syncpoints are called a logical "unit of work" and the updates are identified through a unique name assigned by the syncpoint manager via the recovery facility called a logical

unit of work identifier (LUWID) (col. 10, lines 22-28; col. 12, lines 5-9; col. 22, lines 54-57).

Britton is generally directed to a two-phase commit procedure, which is defined as follows (col. 92, line 56 to col. 93, line 5):

> A procedure for coordinating and/or synchronizing a commit or back out of updates and/or a protected conversation. Usually, the two phase commit procedure is used to atomically commit or back out a plurality of resources or a single resource via a protected conversation. By way of example, the two phase commit procedure can include a polling or prepare phase and a back out or commit phase.

After the phase one prepare-to-commit phase and before the phase two decision to commit or backout, the resources to be changed remain in a state of "in doubt" (col. 13, lines 14-21).

The example of a "protected conversation" between application 56A and application 56D in Britton most closely corresponds to the claimed "message transfer across a transaction-oriented data processing network." The protected conversation is described at column 15, line 1 to column 18, line 50, with respect to Figs. 5A and 5B, and the timing of the commits is discussed at column 27, lines 1-36 with respect to Fig. 9. However, it is difficult to read Britton onto claim 1 because the unit of work in Britton involves an update to file 78D (col. 16, lines 18-21) in system 50D and to files 78A and 78B (col. 16, lines 29-30) in system 50A, rather than

determination of whether a message was successfully received. That is, while Britton sends messages in the protected conversation, the unit of work is not based on the receipt of the messages, as claimed. Britton does not take the specific actions of "committing" a unit of work and "transmitting ... a positive confirmation of receipt" "in response to successful receipt of the messages" or "rolling back" a unit of work and "transmitting ... a negative confirmation of receipt" "in response to unsuccessful receipt of the messages." The rejection does not account for these differences between the subject matter of claim 1 and Britton, which makes it hard to understand how the examiner intends to read Britton onto claim 1. The examiner's reliance on only column 15, line 30 to column 16, line 34 also makes it difficult to understand the rejection since this portion of Britton does not get to the commit procedure. Nevertheless, we try to make the rejection work by looking at Britton and how the examiner maps the claim limitations to Britton.

When application 56A initiates a protected conversation with application 56D in system 50D, a logical unit of work identifier (LUWID) and unique conversation identifier is sent along with a conversation initiate request to the remote system 50D. The LUWID and unique conversation identifier are registered both in the syncpoint manager 60A and syncpoint manager 64D by protected conversation adapters 64A and 64D (step 532 in Fig. 5A) (col. 15,

lines 18-20, 25-29, & 47-50). Protected work done by application 56D will be associated with the logical unit of work originally started by application 56A (Step 532) (col. 15, lines 32-35; col. 23, lines 15-26). Application 56A sends a request to application 56D, which eventually causes application 56D to update file 78D, and application 56D sends a reply to application 56A that it completed its work (step 533; col. 16, lines 18-28). Application 56A then issues update requests for files 78A and 78B (step 533A). No commit processing has been done at this stage. Now application 56A issues a commit 58A (step 534). After this point, a two-phase commit process is carried out.

The examiner considers the claimed "messages" to correspond to the "update message from 56A to 56D in commit phase" (EA4; EA8), the claimed "committing said second unit of work" "in response to successful receipt of the messages" to correspond to "56D updates file 78D" (EA4; EA8), the claimed "transmitting ... a positive confirmation of receipt" to correspond to "reply from 56D to 56A indicating it completed the work/request" (EA4-5; EA8), and the claimed "committing the first unit of work" "in response to the positive confirmation" to correspond to "56A commits/updates 78A, 78B" (EA5; EA8). However, there are several problems with this interpretation. First, updating the file 78D is not "committing said second unit of work" because no commit has been requested at this point. Second, there is no express

teaching that updating file 78A occurs "in response to successful receipt of the messages by the receiver program"; at best, it is indirectly implied that some transport protocol must have verified the data sent from application 56A to application 56D. Third, there is no express teaching of "transmitting ... a positive confirmation of [successful] receipt"; at best, the reply indicating that 56D has completed its work indirectly implies that the message was somehow received successfully. Fourth, updating of files 78A and 78B is not "committing the first unit of work" because no commit has been requested at this point and because claim 1 requires that the first unit of work corresponds to the messages by the sender program, not files on a protected resource. These differences are not addressed or explained away in the rejection. We agree with appellants' argument (Br5) that Britton does not disclose confirmation of performance of a first commit operation being required before performing a second commit operation. Nevertheless, we look at Jefferson to see whether it cures the deficiencies of Britton.

Jefferson discloses rolling back the processing of messages in a queue if the virtual receive time is less than the receiver's virtual time (p. 414). However, we agree with appellants that there is no disclosure in Jefferson of the specific commit or backout processing sequence for logically linked units of work including send and receive operations of a

particular message transfer as recited in claim 1. It is noted that backing out a message occurs in response to the virtual receive time being less than the receiver's virtual time, not in response to an unsuccessful receipt of the messages, as claimed. Jefferson does not cure the deficiencies of Britton. Accordingly, we conclude that the examiner has failed to establish a prima facie case of obviousness. The rejection of claims 1, 4, and 10 is reversed.

It is not clear why the examiner has rejected claim 10 over Britton and Jefferson and not claim 8, since claim 8 is a "data processing system" version of the "computer program product" in claim 10. Nevertheless, we have considered Ranade and Ferree and find that they do not cure the deficiencies in the combination of Britton and Jefferson. Accordingly, we conclude that the examiner has failed to establish a prima facie case of obviousness. The rejection of claims 2, 5, 8, and 9 is reversed.

CONCLUSION

The rejections of claims 1, 2, 4, 5, and 8-10 are reversed.

REVERSED

ERROL A. KRASS Administrative Patent Judge	
LEE E. BARRETT Administrative Patent Judge	BOARD OF PATENT APPEALS AND INTERFERENCES
JOSEPH L. DIXON Administrative Patent Judge	

IBM CORPORATION
3039 CORNWALLIS RD.
DEPT. T81 / B503, PO BOX 12195
RESEARCH TRIANGLE PARK, NC 27709