APPENDIX C-03
Invalidity Claim Chart for U.S. Patent No. 6,785,815 in view of Hanna

WO1999040702A1 to Hanna et al. ("Hanna"), titled "Method and Apparatus for Efficient Authentication and Integrity Checking Using Hierarchical Hashing," was filed February 4, 1999, and was published August 12, 1999. Hanna is therefore prior art at least under § 102(a) and/or (e).

Asserted claims 42 and 43 of U.S. Patent No. 6,785,815 ("the '815 Patent") are invalid as expressly and/or inherently anticipated by Hanna. To the extent that Hanna is found not to anticipate, expressly or inherently, any asserted claim of the '815 Patent, the claim is invalid as obvious in view of Hanna alone or in combination with other prior art references, including but not limited to the prior art identified in the Defendants' Invalidity Contentions and the prior art described in the invalidity claim charts attached thereto as Appendices C.

This invalidity claim chart is based in whole or in part on the Defendants' present understanding of the asserted claims, their current construction of the claims, and/or Intertrust's apparent construction of the claims in its current infringement contentions. The Defendants' are not adopting Intertrust's apparent claim construction, nor admitting to the accuracy of any particular claim construction. To the extent that Intertrust's apparent claim construction or applications thereof are reflected in this invalidity claim chart, nothing herein should be construed as an admission that the Defendants agree with Intertrust's apparent claim construction or Intertrust's application of that claim construction in Intertrust's current infringement contentions.

The use of this reference or combinations of references as invalidating prior art under 35 U.S.C. §§ 102 and/or 103 may be based on Intertrust's allegations of infringement. Defendants do not necessarily agree with the interpretations set forth in Intertrust's infringement contentions and thus this invalidity claim chart is not an admission that the accused products meet any particular claim element or infringe the asserted claim. In addition, nothing in this invalidity claim chart should be interpreted as a position about whether any portion of the asserted claim is limiting or not. Further, by submitting this invalidity claim chart, the Defendants do not waive and hereby expressly reserve their right to raise other invalidity defenses, including but not limited to defenses under 35 U.S.C. §§ 101 and/or 112.

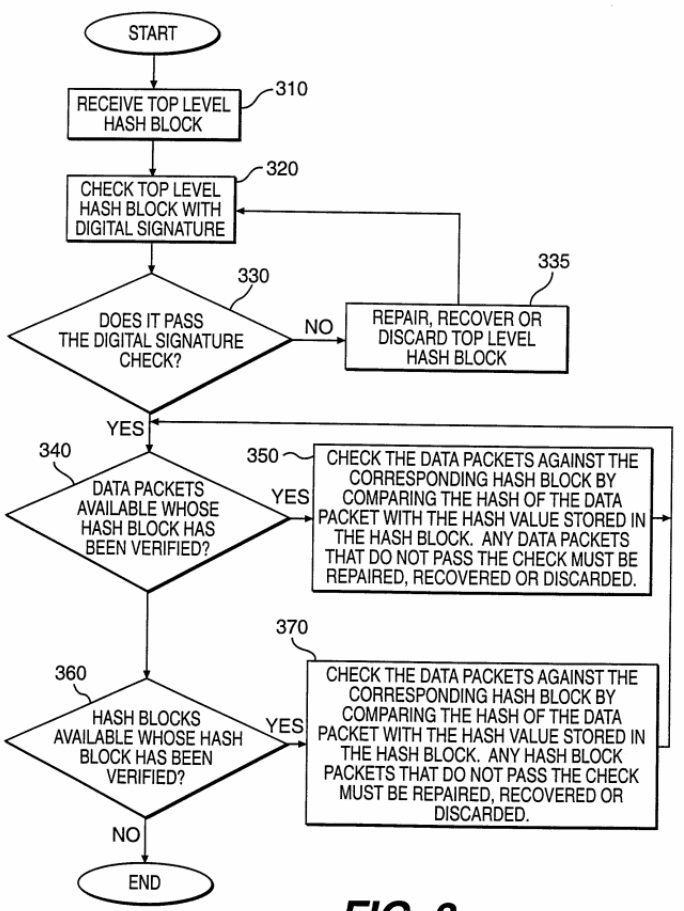
The Defendants reserve all rights to revise, amend, or supplement this invalidity claim chart at a later date as discovery progresses, after the Court issues its claim construction ruling, or if Intertrust amends its infringement contentions.

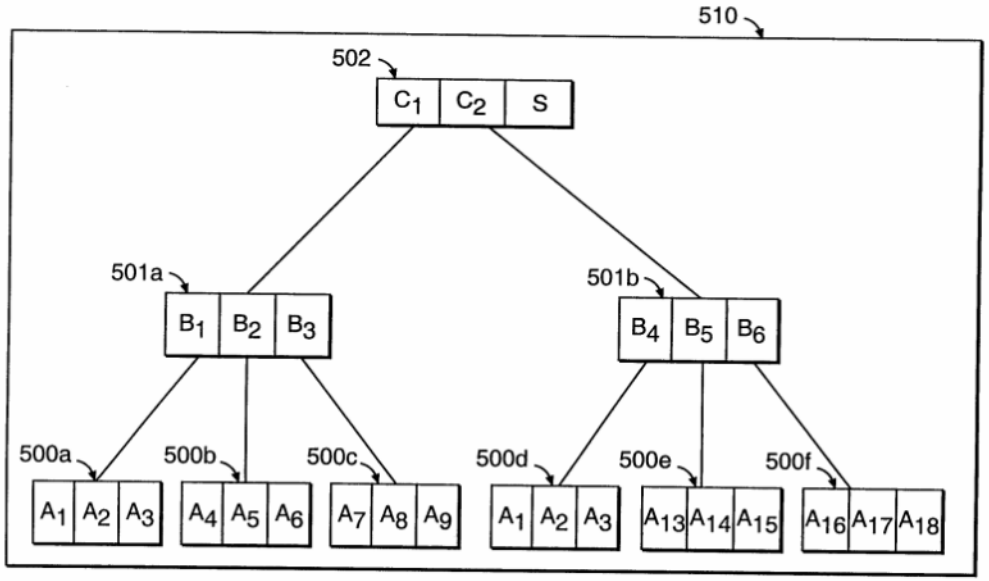
'815 Asserted Claims	Hanna
[815.42p] A method for managing at least one use of a file of electronic data, the method including:	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • Hanna at 1:6-8: This invention generally relates to data corruption detection and, more particularly, to a method and apparatus for efficient authentication and integrity checking in data processing using hierarchical hashing. • Hanna at 3:14-22: In accordance with the present invention, as embodied and broadly described herein, a computer-implemented data processing method comprises the steps of dividing a data set into packets, hashing the data within each of the packets to produce a hash block including hash values and applying a signature to the hash block. In accordance with another aspect of the present invention, as embodied and broadly described herein, a computer-implemented data processing method comprises the steps of receiving a packet including data, a hash block and a digital signature, verifying the digital signature, hashing the data to produce hash values and comparing the hash values to values in the hash block. • Hanna at 1:8-3:4: B. Description of the Related Art Certain types of business activities create the need to transfer information in a secure manner. For instance, banks periodically "backup" their computer files to a remote central database and need to know that these files were successfully copied to the remote database without having been changed or corrupted during the process. Video conferencing is another example of an application that demands the secure transmission of information (video/voice/data). Open networks such as the Internet provide simple and effective means for digital communication. However, such communication can be unintentionally corrupted by network transmission errors or altered by malicious acts. Several conventional techniques guard against these communication problems. These techniques require applying checksums or digital

'815 Asserted Claims	Hanna
	signatures to data before transmission and verifying the checksum or digital signature upon receipt. Checksums, values derived from the data, are typically easy to compute. After computing the checksum, a transmitting machine sends the checksum with the data itself. A receiving machine then recalculates the checksum from the received data and compares the calculated checksum with the received checksum. However, a hacker with the ability to modify data likely also has the ability to replace the original checksum with a recalculated checksum that corresponds to the modified data. Digital signatures protect against malicious acts intended to corrupt data but are more expensive than checksums to compute. The protection provided by digital signatures becomes increasingly important in networked environments where data must pass through unguarded points. Digital signatures are often used in a bulk signature format in which a digital signature is applied to a complete data set. However, the bulk signature format has several drawbacks. The process of confirming the digital signature operates on the whole data set. Bulk signature algorithms cannot identify the portion of the data that was corrupted. This leads to increased cost since the data, if corrupted, must all be sent again. If the data is communicated in discrete packets signed with a bulk signature, detecting corruption or invalid packets inserted by a hacker also carries a high cost. It requires waiting for the receipt of at least one copy of each sequence number of packets. At this point, the signature can be checked. The integrity of the bulk signature cannot be checked until all of the packets are received. This aspect creates a time delay relative to the size of the data set. If several packets with the same sequence number and different contents are received, one is a forgery. Then, to detect which one is the valid copy of any given sequence number if a forgery is inserted, each copy of that sequence number would have to be compared with all the valid packets of the sequence. A method called packet-level signature addresses this problem. The packet-level signature method assigns a digital signature to each individual packet. Although allowing the verification to begin as the individual packets are received making it easy to identify individual corrupted

'815 Asserted Claims	Hanna
	packets, this method requires additional computation and repeated checking of the digital signatures, which is quite time-consuming. A conventional hierarchical hashing technique for neighboring databases on a local area network (LAN) allows a database management system to check if the databases are identical by hashing pieces of the database. The hashes are then hashed, and the final value is broadcast periodically to confirm that all neighbors on the LAN have identical databases. If they do not, the next hash level is compared until the area of the database that differs is located, at which time it can be updated accordingly. However, this system does not deal with the application of a single digital signature to protect an arbitrary amount of data against both malicious errors and unintentional errors. Currently, there is no system that allows a single digital signature to apply to an arbitrary amount of data and allows the data to be verified as it is received. There is, therefore, a need for a system that protects against unintentional data corruption and malicious acts that cause errors in data processing and allows data to be checked as it is received. Additionally, the need exists for a system that provides information regarding which section of data was corrupted while keeping a low computational overhead. • Hanna at Cl. 10: 10. A computer implemented data processing method comprising the steps of: receiving a packet including data, a hash block and a digital signature; verifying the digital signature; hashing the data to produce hash values; and comparing the hash values to values in the hash block. • Hanna at Cl. 11: 11. A computer implemented data processing method comprising the steps of: signing, with a digital signature, a hash block corresponding to a data set; sending the data and the hash block from a source to a destination;

'815 Asserted Claims	Hanna
	upon receipt at the destination, checking the digital signature on the hash block; applying the hash function to the received data set; and comparing the hashes of the received data with the hash values stored in the hash block. *See also* 1:10-13, Abstract, 3:6-17, 4:15, 8:9-11, Figs. 2-5. To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42a] receiving a request to use the file in a predefined manner;	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • Hanna at 5:23-28: The invention is related to the use of computer system 100 for signing data and verifying data. According to one embodiment of the invention, signed or verified data is provided by computer system 100 in response to processor 104 executing one or more sequences of one or more instructions contained in main memory 106. Such instructions may be read into main memory 106 from another computer-readable medium, such as storage device 110. Execution of the sequences of instructions contained in main memory 106 causes processor 104 to perform the process steps described herein. • Hanna at 4:19-21: The top level hash block, the smallest one, is signed with a digital signature. The hash blocks and data are then transmitted to an intended destination such as a network node. • Hanna at 11:12-13: Data verification begins with checking the digital signature on the top level hash block 502. If the top level hash block 502 passes this check, the next level hash block 501 is verified.

’815 Asserted Claims	Hanna
	 FIG. 3

’815 Asserted Claims	Hanna
	 **FIG. 5** • Hanna at Cl. 10: 10. A computer implemented data processing method comprising the steps of: receiving a packet including data, a hash block and a digital signature; verifying the digital signature; hashing the data to produce hash values; and comparing the hash values to values in the hash block. • Hanna at Cl. 11:

'815 Asserted Claims	Hanna
	11. A computer implemented data processing method comprising the steps of: signing, with a digital signature, a hash block corresponding to a data set; sending the data and the hash block from a source to a destination; upon receipt at the destination, checking the digital signature on the hash block; applying the hash function to the received data set; and comparing the hashes of the received data with the hash values stored in the hash block. *See also* 5:2-5, 5:22-28, 6:26-7:7:23. To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42b] retrieving at least one digital signature and at least one check value associated with the file;	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • Hanna at 3:19-22: In accordance with another aspect of the present invention, as embodied and broadly described herein, a computer-implemented data processing method comprises the steps of receiving a packet including data, a hash block and a digital signature, verifying the digital signature, hashing the data to produce hash values and comparing the hash values to values in the hash block. • Hanna at 4:22-25: Systems consistent with the present invention generally verify the data set by checking the digital signature on the top level hash block. Once the top level hash block is verified, the next lower level hash block can be verified by comparing the hash of the packets of that hash block against the hashes stored in the top level block.

'815 Asserted Claims	Hanna
	• Hanna at 11:12-13: Data verification begins with checking the digital signature on the top level hash block 502. If the top level hash block 502 passes this check, the next level hash block 501 is verified. • Hanna at 9:22-28: Fig. 3 is a flowchart of the steps used in the data receiving and verification procedure for systems consistent with the present invention. To verify data, the digital signature on the top level packet must be verified first (step 320). If the signature fails this verification step or it is determined that the packet was corrupted during transmission, the top level packet must be repaired or recovered before checking the other packets (steps 330, 335). If lower level hash blocks were signed for extraordinary redundancy, their signatures may be checked in this case to allow verification to proceed. • Hanna at 8:14-17: Once the data is broken into packets, a collision-proof, one-way hash function is applied to each packet (step 220). Each application of the hash function will produce a single hash value. The application of the hash function to each of the data packets will produce a sequence of hash values. This sequence is called a hash block (step 230). • Hanna at 9:1-4: If, however, the hash block originally created by the hashing of the data packets (step 230) is small enough to fit in a single packet with the digital signature, there is no need to go through the steps of breaking down the hash block and creating another higher level hash block and the top level hash block remains the only hash block. • Hanna at 9:5-8: Systems consistent with the present invention apply a digital signature to the top level hash block packet (step 250). No signature need be applied to any of the other hash blocks or data packets. The single digital signature applied to the top level packet is sufficient to verify all of the data.

'815 Asserted Claims	Hanna
	• Hanna at 9:18-20: In accordance with the data signing procedure, the top level hash block packet with the single digital signature, the hierarchy of lower level hash blocks, and the data are sent to a receiver. • Hanna at Cl. 10: 10. A computer implemented data processing method comprising the steps of: receiving a packet including data, a hash block and a digital signature; verifying the digital signature; hashing the data to produce hash values; and comparing the hash values to values in the hash block. • Hanna at Cl. 11: 11. A computer implemented data processing method comprising the steps of: signing, with a digital signature, a hash block corresponding to a data set; sending the data and the hash block from a source to a destination; upon receipt at the destination, checking the digital signature on the hash block; applying the hash function to the received data set; and comparing the hashes of the received data with the hash values stored in the hash block.

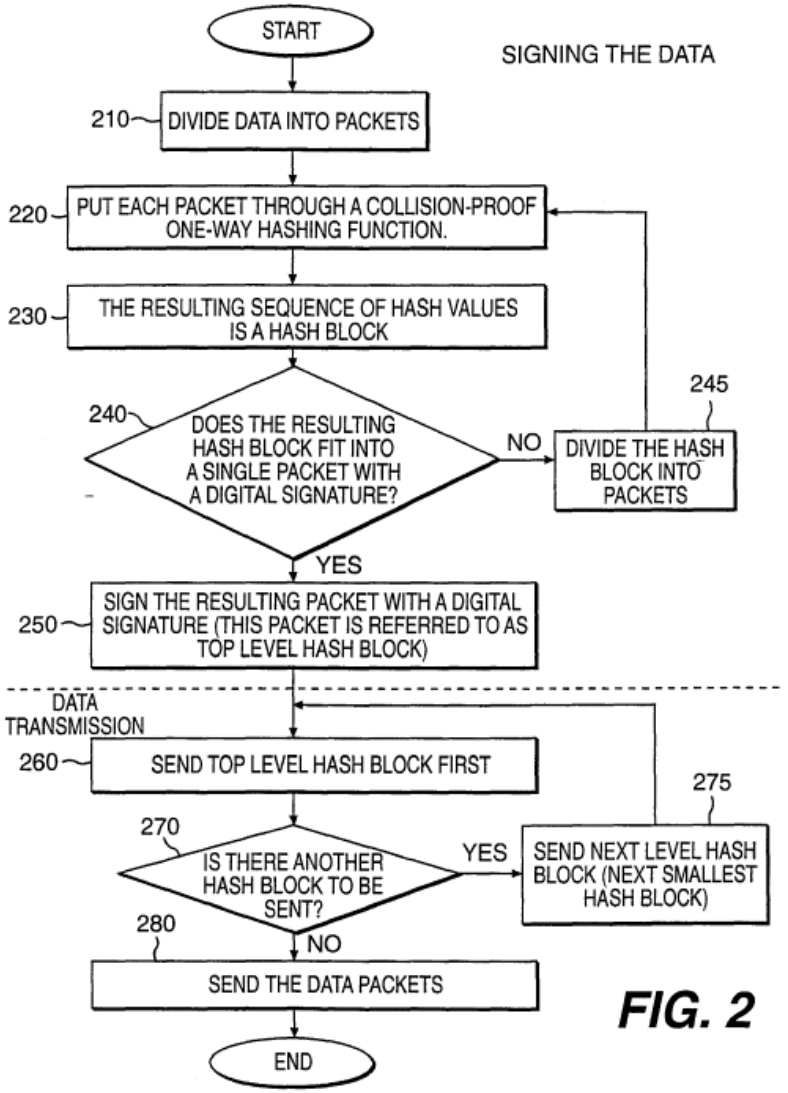
’815 Asserted Claims	Hanna
	

FIG. 2

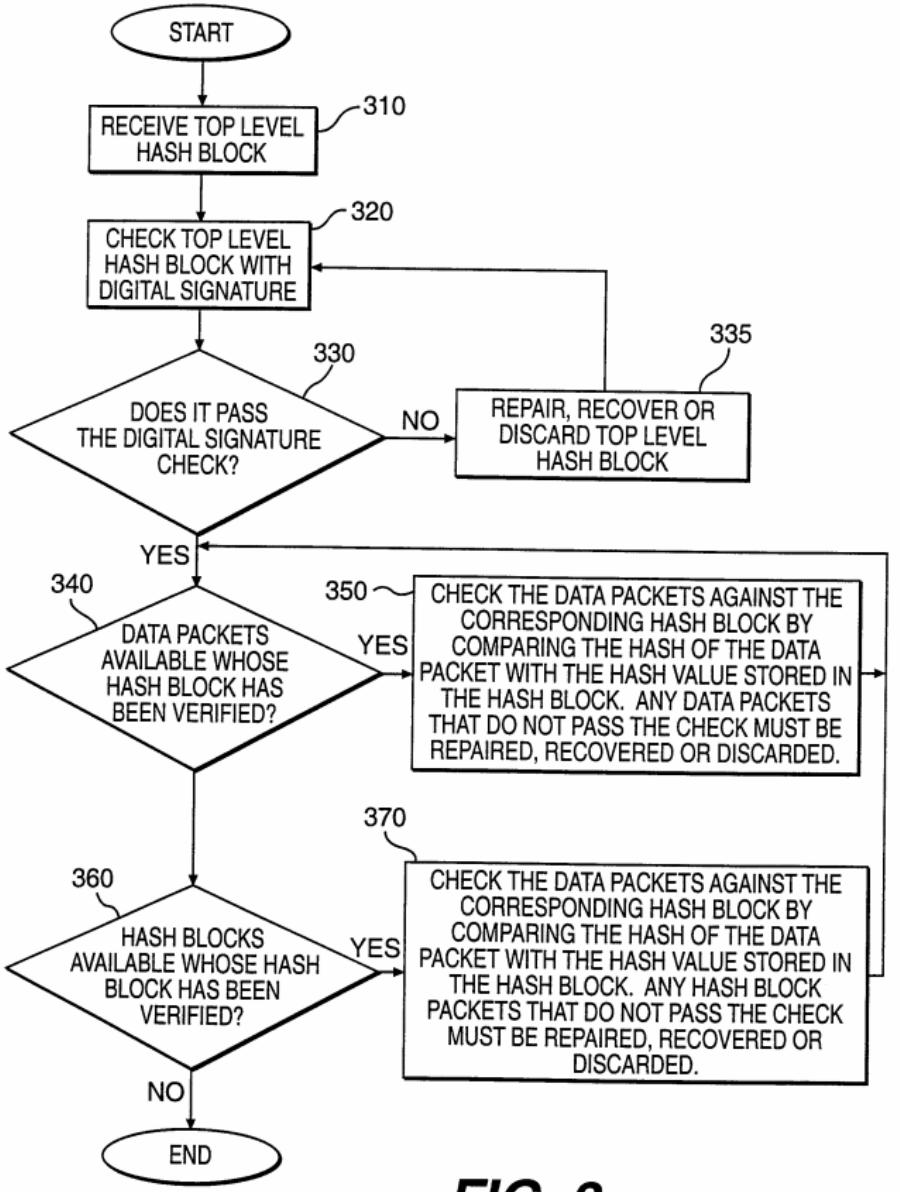
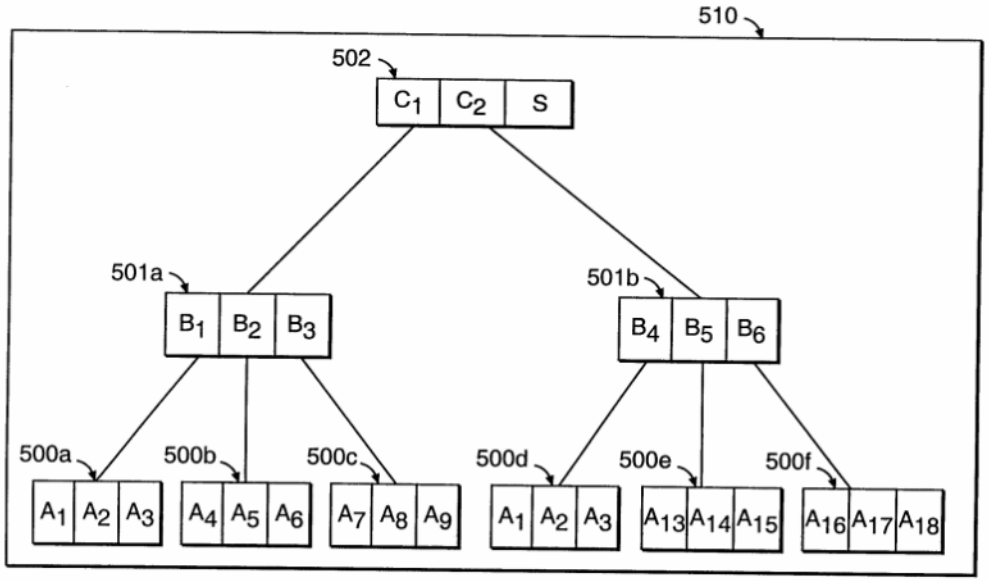

START
RECEIVE TOP LEVEL HASH BLOCK
310
CHECK TOP LEVEL HASH BLOCK WITH DIGITAL SIGNATURE
320
DOES IT PASS THE DIGITAL SIGNATURE CHECK?
330
NO
REPAIR, RECOVER OR DISCARD TOP LEVEL HASH BLOCK
335
YES
DATA PACKETS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
340
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY DATA PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
350
HASH BLOCKS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
360
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY HASH BLOCK PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
370
NO
END

FIG. 3

’815 Asserted Claims	Hanna
	 FIG. 5 To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.
[815.42c] verifying the authenticity of the at least one check value using the digital signature;	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • Hanna at 4:22-28:

'815 Asserted Claims	Hanna
	Systems consistent with the present invention generally verify the data set by checking the digital signature on the top level hash block. Once the top level hash block is verified, the next lower level hash block can be verified by comparing the hash of the packets of that hash block against the hashes stored in the top level block. All lower level hash blocks are checked against the next higher level hash blocks in the same manner. Finally, the data is checked against the hash block containing the hashes of the data set. Any given data packet can be checked once all hash blocks above it are received. • Hanna at 9:22-28: Fig. 3 is a flowchart of the steps used in the data receiving and verification procedure for systems consistent with the present invention. To verify data, the digital signature on the top level packet must be verified first (step 320). If the signature fails this verification step or it is determined that the packet was corrupted during transmission, the top level packet must be repaired or recovered before checking the other packets (steps 330, 335). If lower level hash blocks were signed for extraordinary redundancy, their signatures may be checked in this case to allow verification to proceed. • Hanna at 11:12-19: Data verification begins with checking the digital signature on the top level hash block 502. If the top level hash block 502 passes this check, the next level hash block 501 is verified. A hash function (not shown) is applied to each packet and compared with the corresponding hash value in the hash block in the level above it. For example, the packets 501a and b of the hash block 501, B_1 through B_6, are checked with the values stored in the other hash block 502, C_1 and C_2. In the sample in Fig. 5, the hash of the packet B_1, B_2, B_3 would be compared with the value C_1. If the check fails, the packet B_1, B_2, B_3 is corrupt and must be repaired or replaced before any packets that depend on it can be verified, in this case, data values A_1 through A_9, 500a-c. • Hanna at Cl. 10: 10. A computer implemented data processing method comprising the steps of: receiving a packet including data, a hash block and a digital signature; verifying the digital signature;

’815 Asserted Claims	Hanna
	hashing the data to produce hash values; and comparing the hash values to values in the hash block. • Hanna at Cl. 11: 11. A computer implemented data processing method comprising the steps of: signing, with a digital signature, a hash block corresponding to a data set; sending the data and the hash block from a source to a destination; upon receipt at the destination, checking the digital signature on the hash block; applying the hash function to the received data set; and comparing the hashes of the received data with the hash values stored in the hash block.

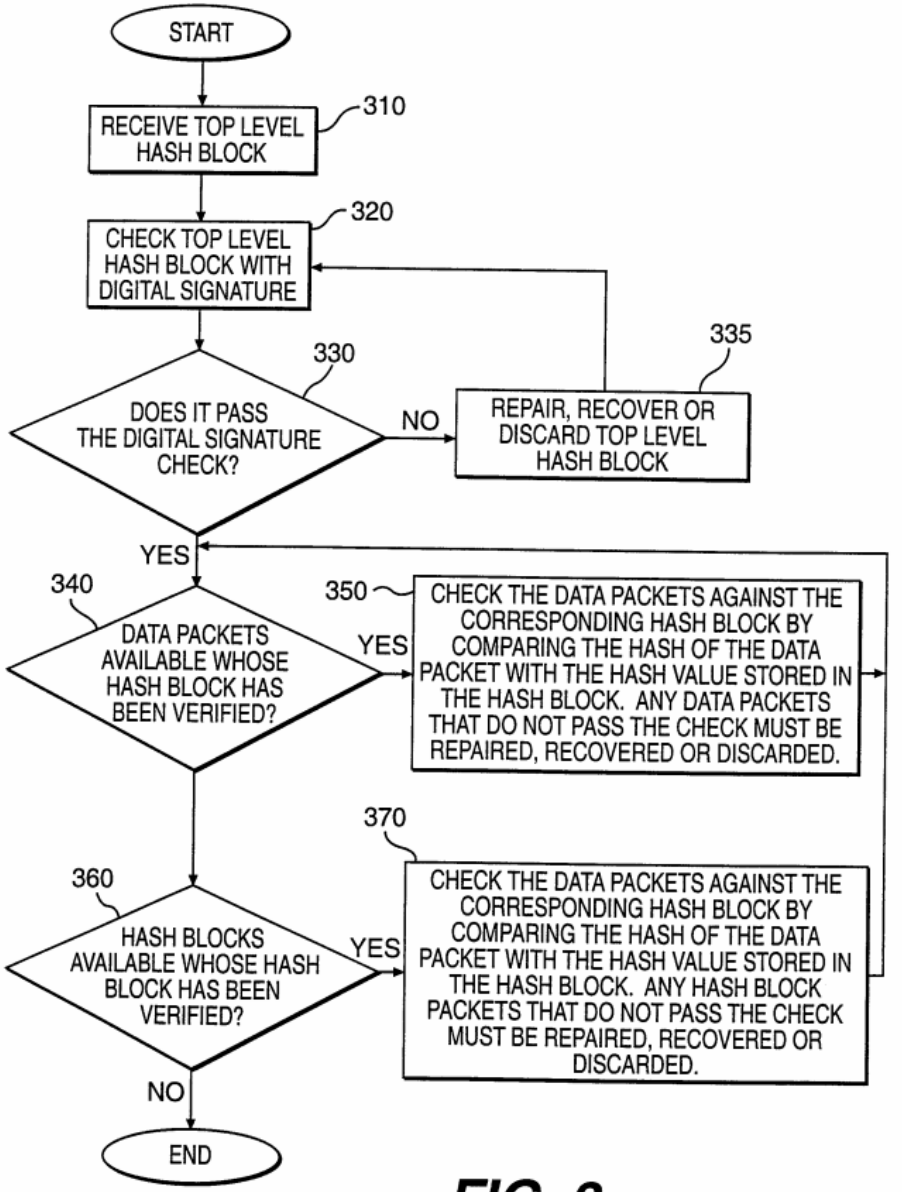
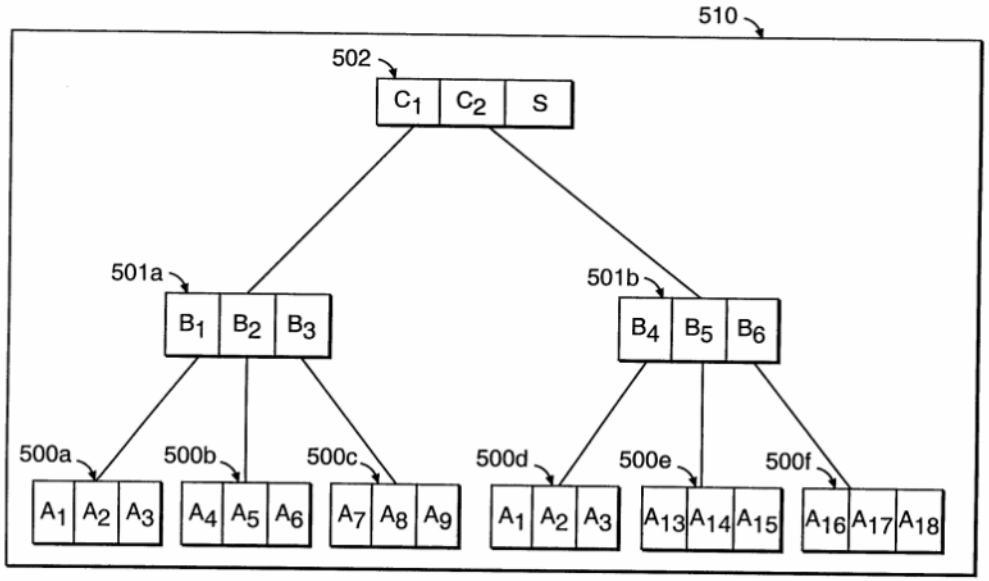

START
RECEIVE TOP LEVEL HASH BLOCK
310
CHECK TOP LEVEL HASH BLOCK WITH DIGITAL SIGNATURE
320
DOES IT PASS THE DIGITAL SIGNATURE CHECK?
330
NO
REPAIR, RECOVER OR DISCARD TOP LEVEL HASH BLOCK
335
YES
DATA PACKETS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
340
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY DATA PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
350
HASH BLOCKS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
360
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY HASH BLOCK PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
370
NO
END

FIG. 3

’815 Asserted Claims	Hanna
	 **FIG. 5** To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.
[815.42d] verifying the authenticity of at least a portion of the file using the at least one check value; and	Hanna discloses this limitation based on at least the following citations. *See, e.g.*, • Hanna at 4:22-28: Systems consistent with the present invention generally verify the data set by checking the digital signature on the top level hash block. Once the top level hash block is verified, the next

'815 Asserted Claims	Hanna
	lower level hash block can be verified by comparing the hash of the packets of that hash block against the hashes stored in the top level block. All lower level hash blocks are checked against the next higher level hash blocks in the same manner. Finally, the data is checked against the hash block containing the hashes of the data set. Any given data packet can be checked once all hash blocks above it are received. • Hanna at 3:10-11: The hierarchical structure used in the method cryptographically protects individual portions of the data and makes it easier to recognize corruption. • Hanna at 3:19-22: In accordance with another aspect of the present invention, as embodied and broadly described herein, a computer-implemented data processing method comprises the steps of receiving a packet including data, a hash block and a digital signature, verifying the digital signature, hashing the data to produce hash values and comparing the hash values to values in the hash block. • Hanna at 4:26-28: Finally, the data is checked against the hash block containing the hashes of the data set. Any given data packet can be checked once all hash blocks above it are received. • Hanna at 10:1-5: Once a hash block is received and verified, data packets that depend on it may be verified (step 340) by computing the hash of the packet and comparing it with the hash value stored in the hash block (step 350). If this check fails, the data packet is corrupt and should be repaired or recovered (step 350). Of course the verification process must use the same hash function used to sign the data. • Hanna at 10:6-11: If it is determined that additional hash blocks are expected (step 360), the packets of these hash blocks are then received and verified (step 370). The hash function is applied to each packet to derive its hash value. The hash value derived from the received hash block packet is compared

'815 Asserted Claims	Hanna
	with the one stored in the received top level hash block. If the comparison fails, the packet is corrupt and must be repaired or replaced before any packets that depend on this hash block can be verified (step 370). • Hanna at 11:12-19: Data verification begins with checking the digital signature on the top level hash block 502. If the top level hash block 502 passes this check, the next level hash block 501 is verified. A hash function (not shown) is applied to each packet and compared with the corresponding hash value in the hash block in the level above it. For example, the packets 501a and b of the hash block 501, B_1 through B_6, are checked with the values stored in the other hash block 502, C_1 and C_2. In the sample in Fig. 5, the hash of the packet B_1, B_2, B_3 would be compared with the value C_1. If the check fails, the packet B_1, B_2, B_3 is corrupt and must be repaired or replaced before any packets that depend on it can be verified, in this case, data values A_1 through A_9, 500a-c. • Hanna at 11:24-12:2: The data packets are checked in the same manner. For example, the data packet A_{10}, A_{11}, A_{12} would be checked by comparing the hash of the data packet 500d with the value B_4. If this check fails, the data packet A_{10}, A_{11}, A_{12}, is corrupt. However, the other data packets can still be verified. For an arbitrary example, in Fig. 5, even if the fourth packet 500d in the data, A_{10}, A_{11}, A_{12}, was corrupt, the fifth packet 500e in the data A_{13}, A_{14}, A_{15} could still be checked by comparison of the hash of the data packet with the value B_5. In this manner, the data receiving 12 and verification procedure (Fig. 3) is applied to the top level hash block 502 to verify data set 500. According to the present invention, data packets may be verified even if the data is sent, received or retrieved out of order. Systems consistent with the present invention need not wait for all of the packets to be delivered to verify a data packet. Any packet can be verified as long as the hash block for that packet has been received and verified. This effectively reduces the delay time to the amount of time required to compute the hash and compare it to the value stored in its corresponding hash block. This delay time is much less than bulk signature's delay time in which all packets must be received before any can be verified. The computational overhead is less than packet-level signature.

'815 Asserted Claims	Hanna
	• Hanna at 12:3-13:6: The choice of when to send a hash block can save time. For instance, the second packet in a hash block need not be sent until after the packets that depend on the first packet in the hash block. This implementation can save the delay time of sending all of the hash blocks first. Conclusion Hierarchical hashing consistent with the present invention protects against malicious modification of data, while checksums do not. It also allows a single digital signature to apply to an arbitrary amount of data, which packet-level signature does not. The structure permits verification of data as it is received, thus eliminating the delay time of waiting for all of the data to be received as in bulk signature methods. Although the data can be verified as it is received, systems consistent with the present invention do not have the high computational overhead associated with individual packet signing. Additionally, the data need not be received or sent in any particular order for verification to begin. The hierarchical structure allows for recognition of corrupt individual packets or sections of data. The other packets that are not corrupt can be used and are unaffected by the corrupt packets. Additionally, other packets that are unverified can be verified even though some packets may be corrupt. Corrupt packets can be repaired or recovered while other packets are being checked. Total replacement of the data is not needed as in bulk signature methods, thus creating greater efficiency and reducing time expended. In summary, systems consistent with the present invention thus allow a single digital signature to protect an arbitrary amount of data, while cryptographically protecting individual portions of the data. To accomplish this, a hierarchy of hash values representing the data is built, and the top level of hashes are signed and sent. After receipt, the digital signature on the hash values is checked. The data, upon receipt, is checked against the hash values that corresponded to the data. • Hanna at Cl. 10: 10. A computer implemented data processing method comprising the steps of: receiving a packet including data, a hash block and a digital signature; verifying the digital signature;

'815 Asserted Claims	Hanna
	hashing the data to produce hash values; and comparing the hash values to values in the hash block. • Hanna at Cl. 11: 11. A computer implemented data processing method comprising the steps of: signing, with a digital signature, a hash block corresponding to a data set; sending the data and the hash block from a source to a destination; upon receipt at the destination, checking the digital signature on the hash block; applying the hash function to the received data set; and comparing the hashes of the received data with the hash values stored in the hash block.

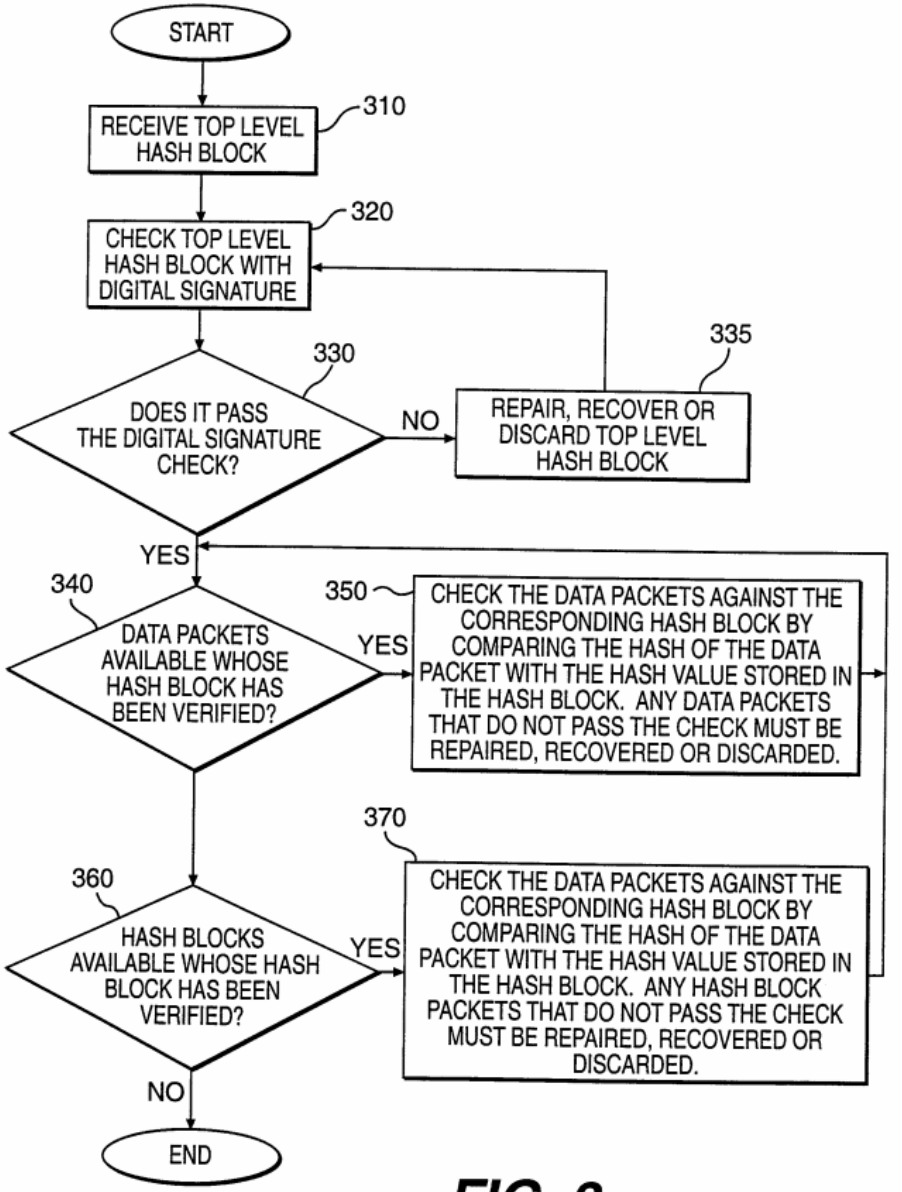
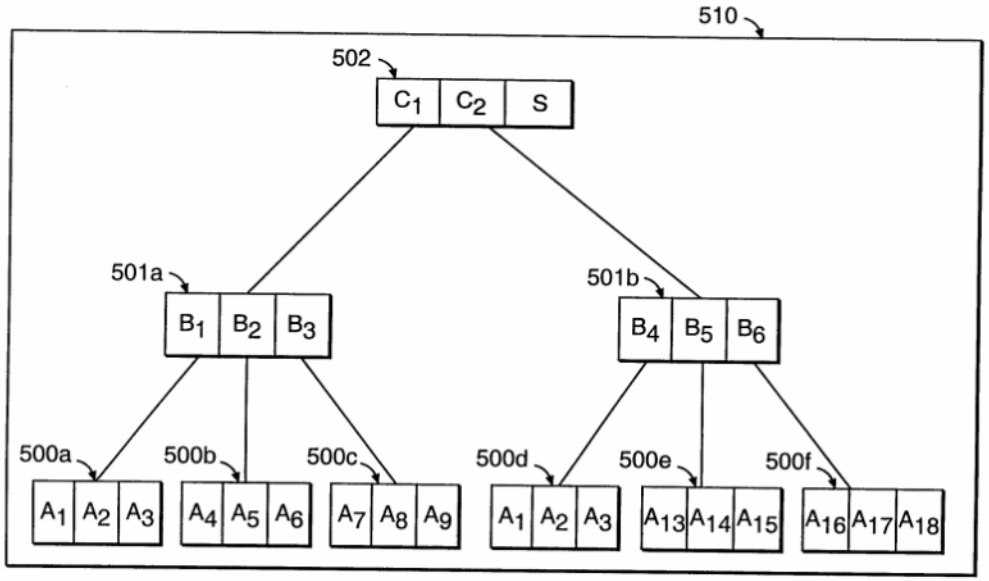

START
RECEIVE TOP LEVEL HASH BLOCK
310
CHECK TOP LEVEL HASH BLOCK WITH DIGITAL SIGNATURE
320
DOES IT PASS THE DIGITAL SIGNATURE CHECK?
330
NO
REPAIR, RECOVER OR DISCARD TOP LEVEL HASH BLOCK
335
YES
DATA PACKETS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
340
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY DATA PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
350
HASH BLOCKS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
360
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY HASH BLOCK PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
370
NO
END

FIG. 3

’815 Asserted Claims	Hanna
	 **FIG. 5** *See also* Abstract, 3:6-17, 4:15, 8:9-11, 9:1-4, 11:27-12:2. To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.
[815.42e] granting the request to use the file in the predefined manner.	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • Hanna at 10:1-5:

’815 Asserted Claims	Hanna
	Once a hash block is received and verified, data packets that depend on it may be verified (step 340) by computing the hash of the packet and comparing it with the hash value stored in the hash block (step 350). If this check fails, the data packet is corrupt and should be repaired or recovered (step 350). Of course the verification process must use the same hash function used to sign the data. • Hanna at 10:12-16: If other hash blocks are expected, the hashes of the packets of each lower level hash block are compared with the values stored in the hash block in the level above. If this verification step fails, the packet is corrupt and must be repaired or recovered before any packets that depend on this packet can be verified. However, other packets in the hash block can be verified. • Hanna at 11:24-26: The data packets are checked in the same manner. For example, the data packet A_{10}, A_{11}, A_{12} would be checked by comparing the hash of the data packet 500d with the value B_4. If this check fails, the data packet A_{10}, A_{11}, A_{12}, is corrupt. • Hanna at 12:23-27: The hierarchical structure allows for recognition of corrupt individual packets or sections of data. The other packets that are not corrupt can be used and are unaffected by the corrupt packets. Additionally, other packets that are unverified can be verified even though some packets may be corrupt. Corrupt packets can be repaired or recovered while other packets are being checked.

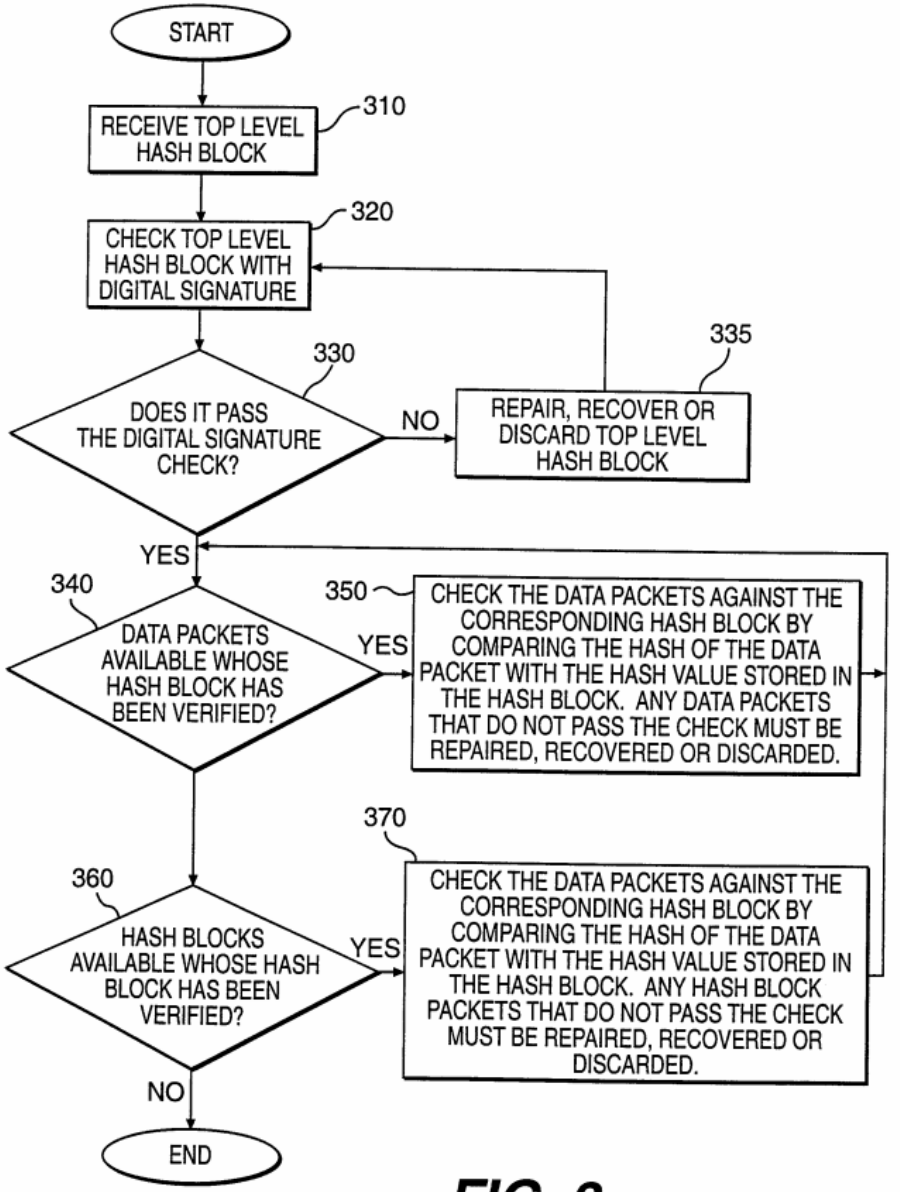

START
RECEIVE TOP LEVEL HASH BLOCK
310
CHECK TOP LEVEL HASH BLOCK WITH DIGITAL SIGNATURE
320
DOES IT PASS THE DIGITAL SIGNATURE CHECK?
330
NO
REPAIR, RECOVER OR DISCARD TOP LEVEL HASH BLOCK
335
YES
DATA PACKETS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
340
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY DATA PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
350
HASH BLOCKS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
360
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY HASH BLOCK PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
370
NO
END

FIG. 3

'815 Asserted Claims	Hanna
	See also 1:6-8, 3:6-13. To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.43] A method as in claim 42, in which the at least one check value comprises one or more hash values, and in which verifying the authenticity of at least a portion of the file using the at least one check value includes: hashing at least a portion of the file to obtain a first hash value; and comparing the first hash value to at least one of the one or more hash values.	Hanna discloses this limitation based on at least the following citations. *See, e.g.,* • *See* Claim [815.42d] above. • Hanna at 3:10-11: The hierarchical structure used in the method cryptographically protects individual portions of the data and makes it easier to recognize corruption. • Hanna at 8:14-17: Once the data is broken into packets, a collision-proof, one-way hash function is applied to each packet (step 220). Each application of the hash function will produce a single hash value. The application of the hash function to each of the data packets will produce a sequence of hash values. This sequence is called a hash block (step 230). • Hanna at 10:1-4: Once a hash block is received and verified, data packets that depend on it may be verified (step 340) by computing the hash of the packet and comparing it with the hash value stored in the hash block (step 350). If this check fails, the data packet is corrupt and should be repaired or recovered (step 350). • Hanna at 10:26-30: Given an initial data sequence of data values 401, the data values are divided into data packets 402a-f. The present example shows 18 data values, A_1 though A_{18}, divided into six packets with three data values in each packet. A one-way, collision-proof hash function 403 is applied to the

'815 Asserted Claims	Hanna
	data packets 403. The hashing of each data packet results in a single hash value. For instance, the hashing of the data packet A_1, A_2, A_3 in this case yields value B_1. • Hanna at 4:26-28: Finally, the data is checked against the hash block containing the hashes of the data set. Any given data packet can be checked once all hash blocks above it are received. • Hanna at 10:1-5: Once a hash block is received and verified, data packets that depend on it may be verified (step 340) by computing the hash of the packet and comparing it with the hash value stored in the hash block (step 350). If this check fails, the data packet is corrupt and should be repaired or recovered (step 350). Of course the verification process must use the same hash function used to sign the data. • Hanna at 10:6-11: If it is determined that additional hash blocks are expected (step 360), the packets of these hash blocks are then received and verified (step 370). The hash function is applied to each packet to derive its hash value. The hash value derived from the received hash block packet is compared with the one stored in the received top level hash block. If the comparison fails, the packet is corrupt and must be repaired or replaced before any packets that depend on this hash block can be verified (step 370). • Hanna at 11:14-15: A hash function (not shown) is applied to each packet and compared with the corresponding hash value in the hash block in the level above it. • Hanna at 11:24-26: The data packets are checked in the same manner. For example, the data packet A_{10}, A_{11}, A_{12} would be checked by comparing the hash of the data packet 500d with the value B_4. If this check fails, the data packet A_{10}, A_{11}, A_{12}, is corrupt.

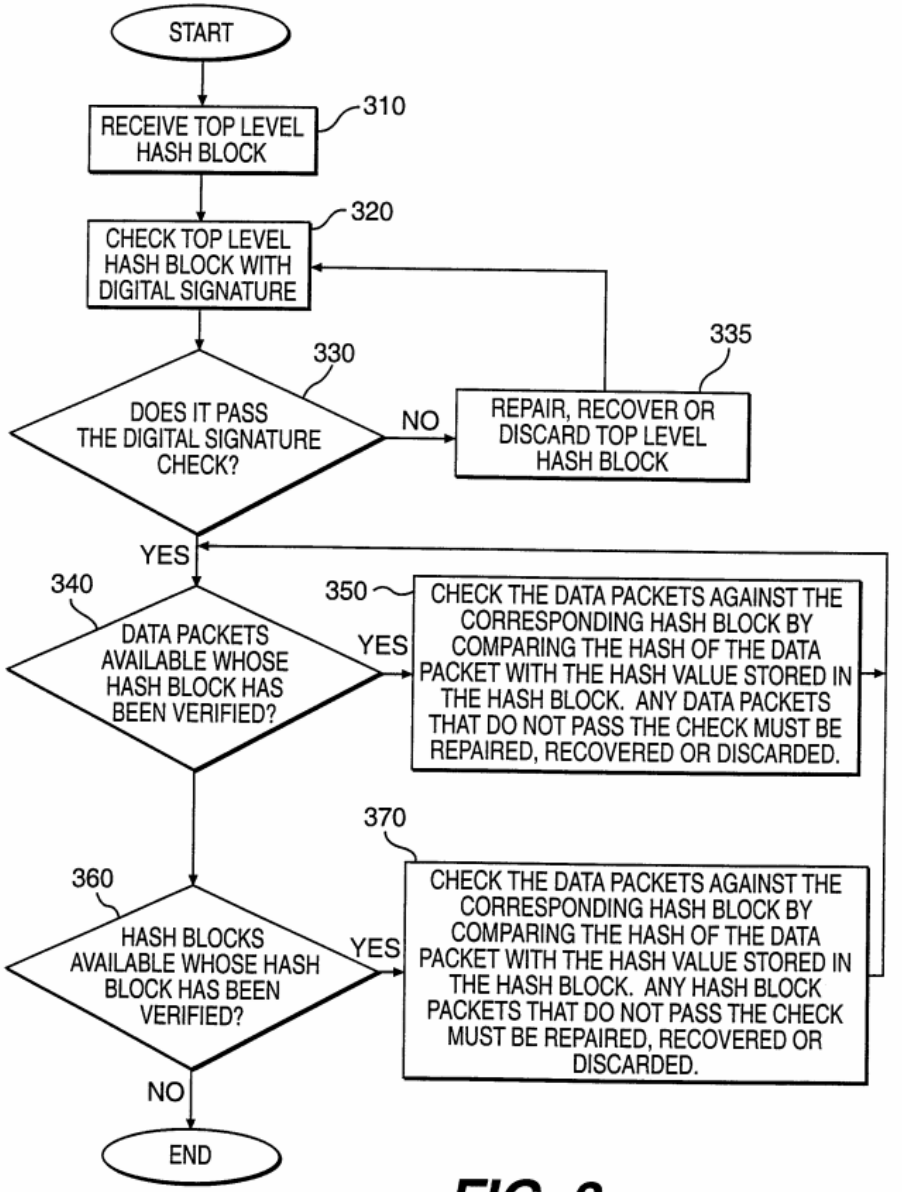
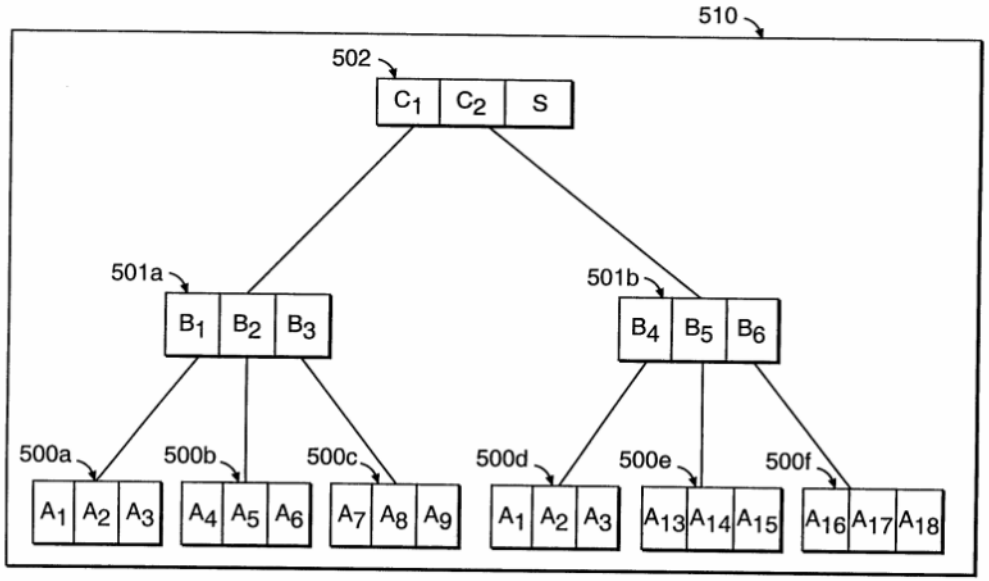

START
RECEIVE TOP LEVEL HASH BLOCK
310
CHECK TOP LEVEL HASH BLOCK WITH DIGITAL SIGNATURE
320
DOES IT PASS THE DIGITAL SIGNATURE CHECK?
330
NO
REPAIR, RECOVER OR DISCARD TOP LEVEL HASH BLOCK
335
YES
DATA PACKETS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
340
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY DATA PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
350
HASH BLOCKS AVAILABLE WHOSE HASH BLOCK HAS BEEN VERIFIED?
360
YES
CHECK THE DATA PACKETS AGAINST THE CORRESPONDING HASH BLOCK BY COMPARING THE HASH OF THE DATA PACKET WITH THE HASH VALUE STORED IN THE HASH BLOCK. ANY HASH BLOCK PACKETS THAT DO NOT PASS THE CHECK MUST BE REPAIRED, RECOVERED OR DISCARDED.
370
NO
END

FIG. 3

’815 Asserted Claims	Hanna
	 **FIG. 5** *See also* Abstract, 3:6-17, 4:15, 8:9-11, 9:1-4, 11:27-12:2. To the extent that Intertrust contends that Hanna does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Hanna in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.

APPENDIX C-04
Invalidity Claim Chart for U.S. Patent No. 6,785,815 in view of Gennaro

U.S. Patent No. 6,009,176 to Gennaro et al. ("Gennaro"), titled "How to sign digital streams," was filed February 13, 1997, and issued December 28, 1999. Gennaro is therefore prior art at least under § 102(a) and/or (e).

Asserted claims 42 and 43 of U.S. Patent No. 6,785,815 ("the '815 Patent") are invalid as expressly and/or inherently anticipated by Gennaro. To the extent that Gennaro is found not to anticipate, expressly or inherently, any asserted claim of the '815 Patent, the claim is invalid as obvious in view of Gennaro alone or in combination with other prior art references, including but not limited to the prior art identified in the Defendants' Invalidity Contentions and the prior art described in the invalidity claim charts attached thereto as Appendices C.

This invalidity claim chart is based in whole or in part on the Defendants' present understanding of the asserted claims, their current construction of the claims, and/or Intertrust's apparent construction of the claims in its current infringement contentions. The Defendants' are not adopting Intertrust's apparent claim construction, nor admitting to the accuracy of any particular claim construction. To the extent that Intertrust's apparent claim construction or applications thereof are reflected in this invalidity claim chart, nothing herein should be construed as an admission that the Defendants agree with Intertrust's apparent claim construction or Intertrust's application of that claim construction in Intertrust's current infringement contentions.

The use of this reference or combinations of references as invalidating prior art under 35 U.S.C. §§ 102 and/or 103 may be based on Intertrust's allegations of infringement. Defendants do not necessarily agree with the interpretations set forth in Intertrust's infringement contentions and thus this invalidity claim chart is not an admission that the accused products meet any particular claim element or infringe the asserted claim. In addition, nothing in this invalidity claim chart should be interpreted as a position about whether any portion of the asserted claim is limiting or not. Further, by submitting this invalidity claim chart, the Defendants do not waive and hereby expressly reserve their right to raise other invalidity defenses, including but not limited to defenses under 35 U.S.C. §§ 101 and/or 112.

The Defendants reserve all rights to revise, amend, or supplement this invalidity claim chart at a later date as discovery progresses, after the Court issues its claim construction ruling, or if Intertrust amends its infringement contentions.

'815 Asserted Claims	Gennaro
[815.42p] A method for managing at least one use of a file of electronic data, the method including:	Gennaro discloses this limitation based on at least the following citations. *See, e.g.*, • Gennaro at Abstract: A method of signing digital streams so that a receiver of the stream can authenticate and consume the stream at the same rate which the stream is being sent to the receiver. More specifically, this invention involves computing and verifying a single digital signature on at least a portion of the stream. The properties of this single signature will propagate to the rest of the stream through ancillary information embedded in the rest of the stream. • Gennaro at 2:30-33: It is therefore an object of this invention to reduce computation time necessary to sign and authenticate a stream of data by reducing the number of digital signatures required for one to authenticate the data stream. • Gennaro at 2:50-54: Finally we assume that the receiver has processing power or hardware that can compute a small number of fast cryptographic c[h]ecksums faster than the incoming stream rate while still being able to play the stream in real-time. • Gennaro at 2:64-67: In the first scenario the stream is finite and is known in its entirety to the signer in advance. This is not a very limiting requirement since it covers most of the internet applications (digital movies, digital sounds, applets). • Gennaro at 3:66-4:1: The preferred embodiment for authenticating streams known in advance to the sender has been designed to work for MPEG video streams. • Gennaro at 5:54-63: The MD5 hash of the recently arrived block is compared against what it should be according to the authentication chain emanating from the Digital Signature from (1.20). If it is different, then tampering has been detected and processing halted. Otherwise, the MPEG

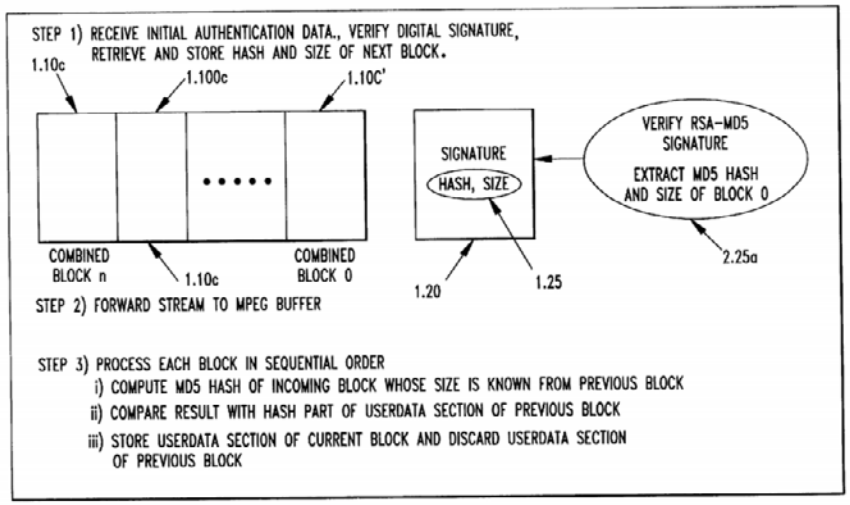
'815 Asserted Claims	Gennaro
	software/hardware is informed that a block of bits of a certain size representing a frame has arrived and it can proceed to process the incoming original block that was authenticated. • Gennaro at 5:3-14: FIG. 2 graphically illustrates how the combined stream previously created is authenticated. The invention requires additional authentication software to be added to any existing MPEG software (2.200) or hardware both of which currently do not do any authentication. The function of this authentication software is to authenticate blocks from the incoming combined stream before passing them on to the MPEG processing software (2.200) to be converted back to video using the MPEG standard. In what follows, the functioning of the additional authentication software is described. • Gennaro at 5:15-20: The authentication software shares with the MPEG processing hardware/software 2.200, a bit buffer which is at least 1.8 M-bits in size. In addition this authentication software now controls how the MPEG processing software/hardware is informed of incoming data. • Gennaro at 12:44-51: As described earlier, the main applications of the present invention are for audio, video and data streams as well as applets in an internet/interactive TV environment. We briefly describe the specifics of how these techniques are applicable for MPEG audio/video and java applets. Later we also describe how our construction carries over to a broadcast environment and how in could be useful in non-stream settings. To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42a] receiving a request to use the file in a predefined manner;	Gennaro discloses this limitation based on at least the following citations. *See, e.g.,*

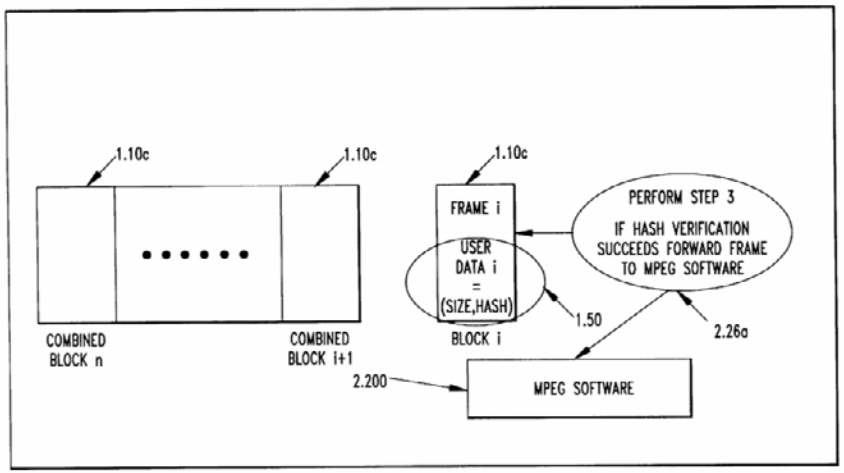
'815 Asserted Claims	Gennaro
	• Gennaro at 1:28-29: When the receiver asks for the authenticated stream, the sender first sends the signed table followed by the stream. • Gennaro at 3:64-4:10: Preferred Embodiment for Signing Streams Known in Advance The preferred embodiment for authenticating streams known in advance to the sender has been designed to work for MPEG video streams. The following is with reference to FIG. 1. In the MPEG video encoding format, each picture frame allows for a USER-DATA section which can act as a placeholder to hold ancillary information. Moreover MPEG standards define that no frame can be more than 1.8 M-bits in size. The original MPEG video stream (1.100) is generated such that there is a 20-byte USER DATA 1.5 section in each picture frame which will act as a placeholder for ancillary information. All these bytes are initialized to the value of 0. • Gennaro at 2:43-54: The present solution makes some reasonable/practical assumptions about the nature of the streams being authenticated. First of all we assume that it is possible for the sender to embed authentication information in the stream. This is usually the case (e.g., USER DATA section in MPEG Video elementary Stream etc.) We also assume that the receiver has a "small" buffer in which it can first authenticate the received bits before consuming them. Finally we assume that the receiver has processing power or hardware that can compute a small number of fast cryptographic ckecksums [sic] faster than the incoming stream rate while still being able to play the stream in real-time. • Gennaro at 5:3-14: FIG. 2 graphically illustrates how the combined stream previously created is authenticated. The invention requires additional authentication software to be added to any existing MPEG software (2.200) or hardware both of which currently do not do any authentication. The function of this authentication software is to authenticate blocks from the incoming combined

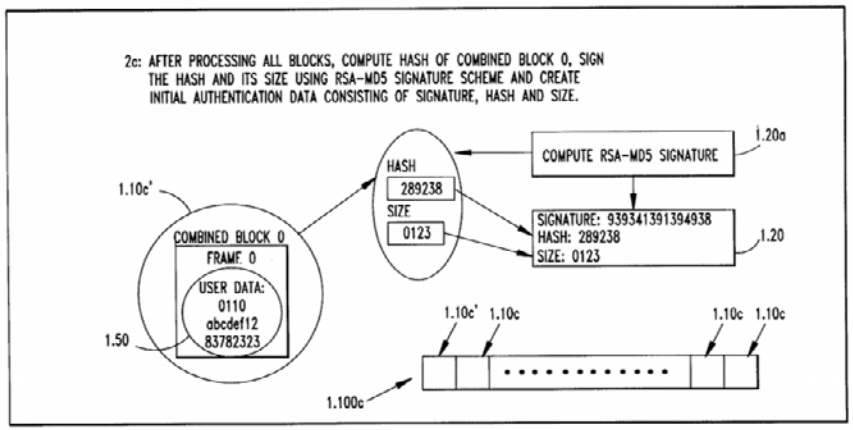
'815 Asserted Claims	Gennaro
	stream before passing them on to the MPEG processing software (2.200) to be converted back to video using the MPEG standard. In what follows, the functioning of the additional authentication software is described. • Gennaro at 5:15-20: The authentication software shares with the MPEG processing hardware/software 2.200, a bit buffer which is at least 1.8 M-bits in size. In addition this authentication software now controls how the MPEG processing software/hardware is informed of incoming data. • Gennaro at 9:63-67: The preferred embodiment for authenticating streams known in advance to the sender has been designed to work for MPEG video streams. • Gennaro at 13:44-46: Accommodating classes or data resources which are generated on the fly by the application server based on a client request. • Gennaro at 13:53-67: Our schemes (both the off-line and the on-line one) can be easily modified to fit in a broadcast scenario. Assume that the stream is being sent to a broadcast channel with multiple receivers who dynamically join or leave the channel. In this case a receiver who joins when the transmission is already started will not be able to authenticate the stream since she missed the first block that contained the signature. Both schemes however can be modified so that every once in a while apart from the regular chaining information, there will also be a regular digital signature on a block embedded in the stream. Receivers who are already verifying the stream via the chaining mechanism can ignore this signature whereas receivers tuned in at various time will rely on the first such signature they encounter to start their authentication chain. • Gennaro at 12:52-13:20: MPEG VIDEO AND AUDIO. In the case of MPEG video and audio, there are several methods for embedding authentication data. Firstly the Video Elementary stream has a USER-DATA section for putting arbitrary user defined information and this section could be embedded in

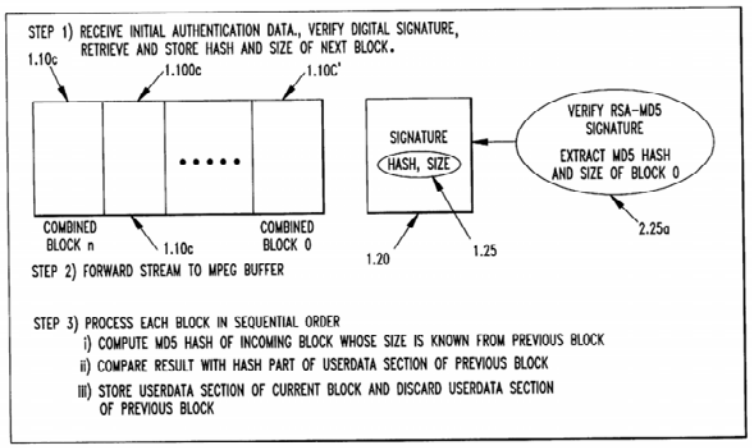
'815 Asserted Claims	Gennaro
	each frame (or field). Secondly, the MPEG system layer allows for an elementary data stream to be multiplexed synchronously with the packetized audio and video streams. One such elementary stream could carry the authentication information. Thirdly, techniques borrowed from digital watermarking can be used to embed information in the audio and video itself at the cost of slight quality degradation. In the case of MPEG video since each frame is fairly large, (hundreds of kilobits) and the receiver is required to have a buffer of at least 1.8 Mbits, both the off-line as well as the on-line solutions can be deployed without compromising picture quality. In the case of audio however, in the extreme case the bit rate could be fairly small (e.g., 32 Kbits/s) and each audio frame can also be small (a few hundred kilobytes) and therefore the receiver's audio buffer could also be very small (3-4 Kbytes). In such extreme cases a direct application of the on-line method, which requires around 1000 bytes of authentication information per block, where a block is limited by the size of the receiver's audio buffer would seriously cut into the audio quality. For these extreme cases, the best on-line strategy would be either to send the authentication information via a separate but multiplexed MPEG data stream. If the receiver has a good sized buffer (say 32 K) then by having a large audio block (say 20 K) our method yields a scheme which has a server-introduced delay of approximately 5-6 seconds and a 5% quality degradation. If the receiver buffer is really tiny (say 2-3 K) a hybrid scheme is required: as an example of a scheme that can be built one could use groups of 33 hash-chained blocks of length 1000 bytes each; this has a 5% degradation and a server initial delay in the 20second range. • Gennaro at 13:20-51: JAVA. In the original version of java (JDK 1.0), for a applet coming from the network, first the startup class was loaded and then additional classes were (down) loaded by the class loader in a lazy fashion as and when the running applet first attempted to access them. Since our ideas apply not only to streams which are a linear sequence of blocks but in general to trees as well (where one block can invoke any of its children), based on our model, one way to sign java applets would be to sign the startup class and each downloaded class would have embedded in it the hashes of the additional classes that it downloads directly. However for code signing, Javasoft has adopted the multiple signature and hash table based approach in JDK1.1, where each applet is composed of one or several Java archives, each of which contains a signed table

'815 Asserted Claims	Gennaro
	of hashes (the manifest) of its components. It is our belief that once java applets become really large and complex the shortcomings of this approach will become apparent: Large size of the hash table in relation to the classes actually invoked during a run. This table has to fully extracted and authenticated before any class gets authenticated. The computational cost of signing each of the manifests if an applet is composed of several archives. Accommodating classes or data resources which are generated on the fly by the application server based on a client request. These could be addressed by using some of our techniques. Also the problem of how to sign audio/video streams will have to be considered in the future evolution of Java, since putting the hash of a large audio/video file is not an acceptable solution. • Gennaro at 13:52-14:2: Broadcast Applications Our schemes (both the off-line and the on-line one) can be easily modified to fit in a broadcast scenario. Assume that the stream is being sent to a broadcast channel with multiple receivers who dynamically join or leave the channel. In this case a receiver who joins when the transmission is already started will not be able to authenticate the stream since she missed the first block that contained the signature. Both schemes however can be modified so that every once in a while apart from the regular chaining information, there will also be a regular digital signature on a block embedded in the stream. Receivers who are already verifying the stream via the chaining mechanism can ignore this signature whereas receivers tuned in at various time will rely on the first such signature they encounter to start their authentication chain. A different method to authenticate broadcasted streams, with weaker non-repudiation properties than ours, was proposed in [8]. • Gennaro at 14:3-15: Long Files When Communication is at Cost

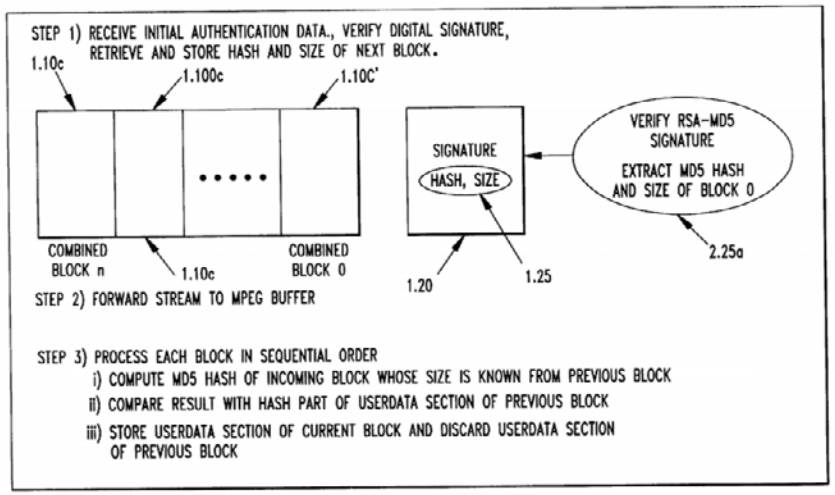
’815 Asserted Claims	Gennaro
	Our solution can be used also to authenticate long files in a way to reduce communication cost in case of tampering. Suppose that a receiver is downloading a long file from the Web. There is no "stream requirement" to consume the file as it is downloaded, so the receiver could easily receive the whole file and then check a signature at the end. However if the file has been tampered with, the user will be able to detect this fact only at the end. Since communication is at a cost (time spent online, bandwidth wasted etc) this is not a satisfactory solution. Using our solution the receiver can interrupt the transmission as soon as tampering is detected thus saving precious communication resources. FIG.2A 

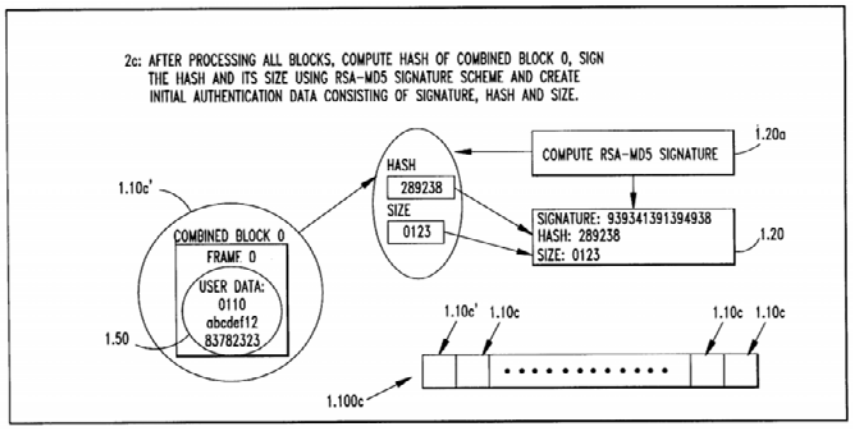
’815 Asserted Claims	Gennaro
	 FIG.2B FIG.2 FIG.2A FIG.2B To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.
[815.42b] retrieving at least one digital signature and at least one check value associated with the file;	Gennaro discloses this limitation based on at least the following citations. *See, e.g.,* • Gennaro at 4:59-5:2: The hash 1.25 and the size of the combined distinguished block, which is the first of the combined blocks, in the stream (1.10c') is calculated, and the said hash value together with the size of said block is signed using a RSA-MD5 signature scheme (1.20a). The signature, the hash and the size are combined to form initial authentication data (1.20). The combined blocks (1.10c) in sequence now constitute the combined stream (1.100c). The initial authentication data (1.20) will be sent to the receiver before the combined stream (1.100c) is sent.

'815 Asserted Claims	Gennaro
	• Gennaro at 5:21-34: The following constitute the steps that the authentication software follows. Step 1) Before receiving the combined stream (1.100c) the receiver receives the initial authentication data (1.20) consisting of a digital signature on the hash of the first combined block (1.10c) and the size of first combined block, together with the purported values of the hash and size. The authentication software at this stage does the following (2.25a): It verifies the signature. If the provided signature verifies, then the purported hash and size values of the first combined block (1.10c) are authentic, and the hash and size values 1.25 are extracted out for use in authenticating the combined stream (1.100c).  FIG.1B FIG.1 FIG.1A FIG.1B

'815 Asserted Claims	Gennaro
	FIG.2A  To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42c] verifying the authenticity of the at least one check value using the digital signature;	Gennaro discloses this limitation based on at least the following citations. *See, e.g.*, • Gennaro at 3:5-7: In this first scenario, verification of the find [sic] stream is performed by verifying the signature of the first block and subsequently verifying hashes of the following blocks. • Gennaro at 5:21-34: The following constitute the steps that the authentication software follows. Step 1) Before receiving the combined stream (1.100c) the receiver receives the initial authentication data (1.20) consisting of a digital signature on the hash of the first combined block (1.10c) and the size of first combined block, together with the purported values of the hash and size. The

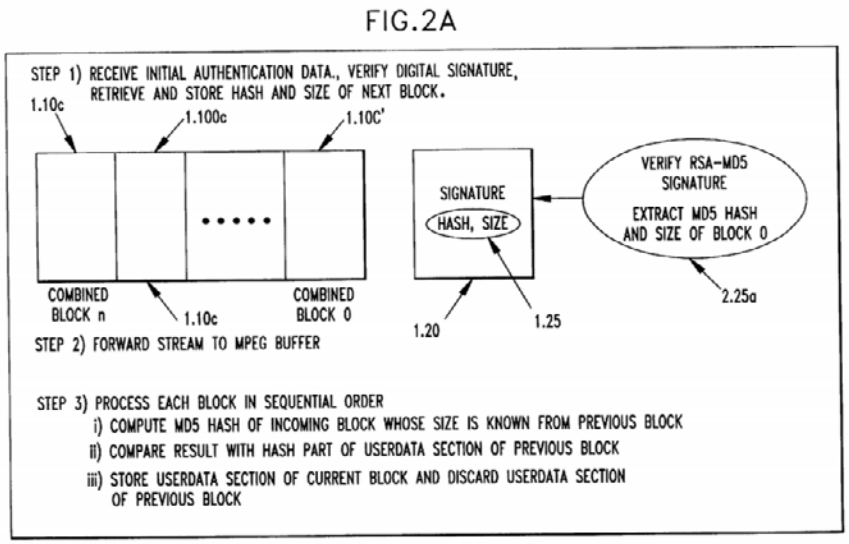
'815 Asserted Claims	Gennaro
	authentication software at this stage does the following (2.25a): It verifies the signature. If the provided signature verifies, then the purported hash and size values of the first combined block (1.10c) are authentic, and the hash and size values 1.25 are extracted out for use in authenticating the combined stream (1.100c). • Gennaro at 5:40-67: Step 3) The incoming combined stream is split into blocks by the authentication software shown in FIG. 2. This splitting is facilitated by the software always knowing in advance the authenticated size of the next incoming combined block long before the incoming block is fully received. The following algorithm or step (2.26a), which is repeated for all the blocks in the stream, is now applied: i) Compute the MD5 hash of the incoming bytes as they are transferred into the buffer of the receiver. As soon as a block comes in, its MD5 hash gets computed, and computation of the MD5 hash of the next incoming block is started (although its length is not immediately known). ii) The MD5 hash of the recently arrived block is compared against what it should be according to the authentication chain emanating from the Digital Signature from (1.20). If it is different, then tampering has been detected and processing halted. Otherwise, the MPEG software/hardware is informed that a block of bits of a certain size representing a frame has arrived and it can proceed to process the incoming original block that was authenticated. iii) Concurrently with the above, the now authenticated size and hash of the next block stored in the ancillary part of the processed block (1.50) is extracted for use to authenticate the next block. • Gennaro at Cl. 8 : 8. A method of authenticating a combined stream of data, comprising: decomposing said stream into a plurality of combined blocks, each of said combined blocks comprising consumable information and ancillary information; establishing non-repudiably the sender of said stream by verifying a digital signature on one of said combined blocks, and authenticating some of said combined blocks by using ancillary information extracted from said authenticated combined blocks.

’815 Asserted Claims	Gennaro
	FIG.2A 

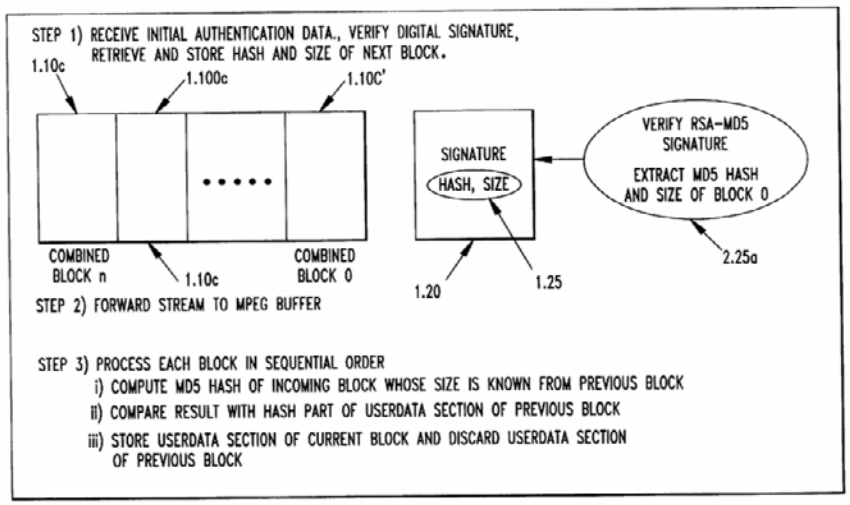
'815 Asserted Claims	Gennaro
	 FIG.1B FIG.1 FIG.1A FIG.1B To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42d] verifying the authenticity of at least a portion of the file using the at least one check value; and	Gennaro discloses this limitation based on at least the following citations. *See, e.g.*, • Gennaro at 5:8-12: The function of this authentication software is to authenticate blocks from the incoming combined stream before passing them on to the MPEG processing software (2.200) to be converted back to video using the MPEG standard. • Gennaro at 2:67-3:4:

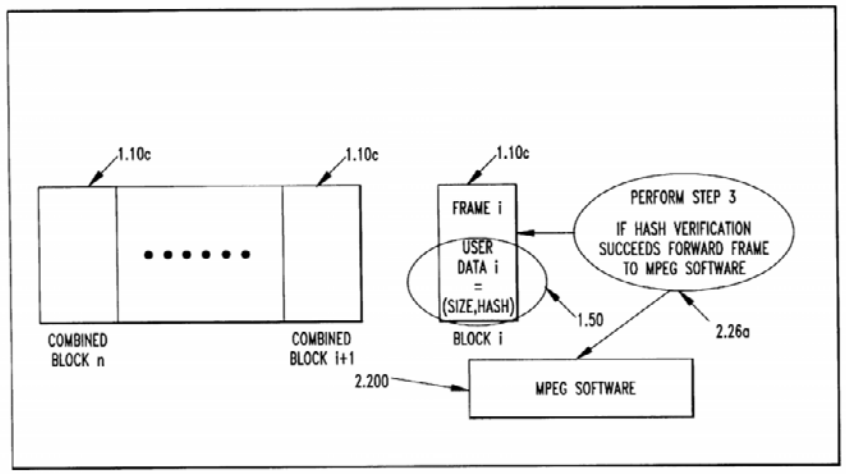
'815 Asserted Claims	Gennaro
	In this case we will show that a single hash computation will suffice to authenticate the internal blocks. The idea is to embed in the current block the hash of the following block (which in turns includes the hash of the following one and so on . . .). • Gennaro at 4:44-48: The next sixteen bytes of the 20 byte ancillary information placeholder is updated to hold the MD5 cryptographic hash of the successor block. This information can be used to authenticate the the successor block which already has its ancillary information in place. • Gennaro at 5:29-34: The authentication software at this stage does the following (2.25a): It verifies the signature. If the provided signature verifies, then the purported hash and size values of the first combined block (1.10c') are authentic, and the hash and size values 1.25 are extracted out for use in authenticating the combined stream (1.100c). • Gennaro at 5:40-67: Step 3) The incoming combined stream is split into blocks by the authentication software shown in FIG. 2. This splitting is facilitated by the software always knowing in advance the authenticated size of the next incoming combined block long before the incoming block is fully received. The following algorithm or step (2.26a), which is repeated for all the blocks in the stream, is now applied: i) Compute the MD5 hash of the incoming bytes as they are transferred into the buffer of the receiver. As soon as a block comes in, its MD5 hash gets computed, and computation of the MD5 hash of the next incoming block is started (although its length is not immediately known). ii) The MD5 hash of the recently arrived block is compared against what it should be according to the authentication chain emanating from the Digital Signature from (1.20). If it is different, then tampering has been detected and processing halted. Otherwise, the MPEG software/hardware is informed that a block of bits of a certain size representing a frame has arrived and it can proceed to process the incoming original block that was authenticated.

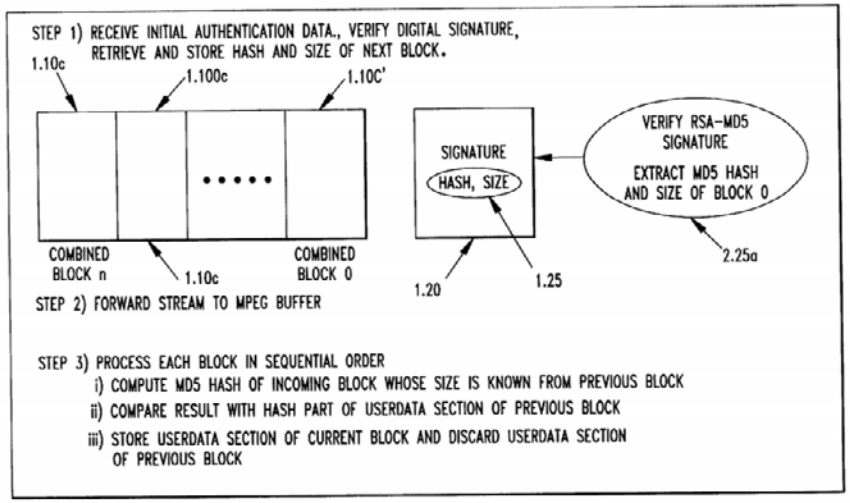
'815 Asserted Claims	Gennaro
	iii) Concurrently with the above, the now authenticated size and hash of the next block stored in the ancillary part of the processed block (1.50) is extracted for use to authenticate the next block. • Gennaro at Cl. 8: 8. A method of authenticating a combined stream of data, comprising: decomposing said stream into a plurality of combined blocks, each of said combined blocks comprising consumable information and ancillary information; establishing non-repudiably the sender of said stream by verifying a digital signature on one of said combined blocks, and authenticating some of said combined blocks by using ancillary information extracted from said authenticated combined blocks. • Gennaro at Cl. 9: 9. A method as recited in claim 8, wherein said ancillary information is a hash of some of said combined blocks and said combined blocks are authenticated by verifying the value of said hash.

'815 Asserted Claims	Gennaro
	 To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42e] granting the request to use the file in the predefined manner.	Gennaro discloses this limitation based on at least the following citations. *See, e.g.*, • Gennaro at 2:39-42: As a consequence, the receiver can consume authenticated data from the stream at the same rate he receives such data from the stream. • Gennaro at 2:43-54: The present solution makes some reasonable/practical assumptions about the nature of the streams being authenticated. First of all we assume that it is possible for the sender to embed

'815 Asserted Claims	Gennaro
	authentication information in the stream. This is usually the case (e.g., USER DATA section in MPEG Video elementary Stream etc.) We also assume that the receiver has a "small" buffer in which it can first authenticate the received bits before consuming them. Finally we assume that the receiver has processing power or hardware that can compute a small number of fast cryptographic ckecksums [sic] faster than the incoming stream rate while still being able to play the stream in real-time. • Gennaro at 5:8-12: The function of this authentication software is to authenticate blocks from the incoming combined stream before passing them on to the MPEG processing software (2.200) to be converted back to video using the MPEG standard. • Gennaro at 5:35-39: Step 2) The receiver then receives the combined stream which is forwarded into the MPEG bit-buffer. However the MPEG software is not informed of incoming data before it is authenticated. This prevents the MPEG software to consuming unauthenticated data. • Gennaro at 5:54-63: The MD5 hash of the recently arrived block is compared against what it should be according to the authentication chain emanating from the Digital Signature from (1.20). If it is different, then tampering has been detected and processing halted. Otherwise, the MPEG software/hardware is informed that a block of bits of a certain size representing a frame has arrived and it can proceed to process the incoming original block that was authenticated. • Gennaro at 12:44-51: As described earlier, the main applications of the present invention are for audio, video and data streams as well as applets in an internet/interactive TV environment. We briefly describe the specifics of how these techniques are applicable for MPEG audio/video and java applets. Later we also describe how our construction carries over to a broadcast environment and how in could be useful in non-stream settings.

’815 Asserted Claims	Gennaro
	FIG.2A 

’815 Asserted Claims	Gennaro
	 To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.
[815.43] A method as in claim 42, in which the at least one check value comprises one or more hash values, and in which verifying the authenticity of at least a portion of the file using the at least one check value includes: hashing at least a portion of the file to obtain	Gennaro discloses this limitation based on at least the following citations. *See, e.g.*, • *See* Claim [815.42d] above. • Gennaro at 5:40-67: Step 3) The incoming combined stream is split into blocks by the authentication software shown in FIG. 2. This splitting is facilitated by the software always knowing in advance the authenticated size of the next incoming combined block long before the incoming block is fully

'815 Asserted Claims	Gennaro
a first hash value; and comparing the first hash value to at least one of the one or more hash values.	received. The following algorithm or step (2.26a), which is repeated for all the blocks in the stream, is now applied: i) Compute the MD5 hash of the incoming bytes as they are transferred into the buffer of the receiver. As soon as a block comes in, its MD5 hash gets computed, and computation of the MD5 hash of the next incoming block is started (although its length is not immediately known). ii) The MD5 hash of the recently arrived block is compared against what it should be according to the authentication chain emanating from the Digital Signature from (1.20). If it is different, then tampering has been detected and processing halted. Otherwise, the MPEG software/hardware is informed that a block of bits of a certain size representing a frame has arrived and it can proceed to process the incoming original block that was authenticated. iii) Concurrently with the above, the now authenticated size and hash of the next block stored in the ancillary part of the processed block (1.50) is extracted for use to authenticate the next block. FIG.2A 

’815 Asserted Claims	Gennaro
	To the extent that Intertrust contends that Gennaro does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Gennaro in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants’ Invalidity Contentions.

APPENDIX C-07
Invalidity Claim Chart for U.S. Patent No. 6,785,815 in view of Stefik

U.S. Patent No. 5,629,980 to Stefik et al. ("Stefik"), titled "System for controlling the distribution and use of digital works," was filed November 23, 1994, and issued May 13, 1997. Stefik is therefore prior art at least under § 102(a), (b), and/or (e).

Asserted claims 42 and 43 of U.S. Patent No. 6,785,815 ("the '815 Patent") are invalid as expressly and/or inherently anticipated by Stefik. To the extent that Stefik is found not to anticipate, expressly or inherently, any asserted claim of the '815 Patent, the claim is invalid as obvious in view of Stefik alone or in combination with other prior art references, including but not limited to the prior art identified in the Defendants' Invalidity Contentions and the prior art described in the invalidity claim charts attached thereto as Appendices C.

This invalidity claim chart is based in whole or in part on the Defendants' present understanding of the asserted claims, their current construction of the claims, and/or Intertrust's apparent construction of the claims in its current infringement contentions. The Defendants' are not adopting Intertrust's apparent claim construction, nor admitting to the accuracy of any particular claim construction. To the extent that Intertrust's apparent claim construction or applications thereof are reflected in this invalidity claim chart, nothing herein should be construed as an admission that the Defendants agree with Intertrust's apparent claim construction or Intertrust's application of that claim construction in Intertrust's current infringement contentions.

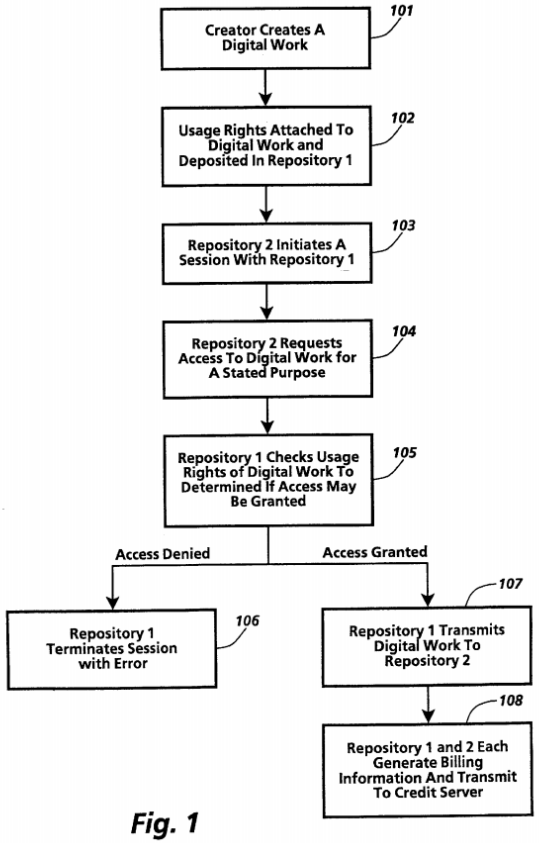
The use of this reference or combinations of references as invalidating prior art under 35 U.S.C. §§ 102 and/or 103 may be based on Intertrust's allegations of infringement. Defendants do not necessarily agree with the interpretations set forth in Intertrust's infringement contentions and thus this invalidity claim chart is not an admission that the accused products meet any particular claim element or infringe the asserted claim. In addition, nothing in this invalidity claim chart should be interpreted as a position about whether any portion of the asserted claim is limiting or not. Further, by submitting this invalidity claim chart, the Defendants do not waive and hereby expressly reserve their right to raise other invalidity defenses, including but not limited to defenses under 35 U.S.C. §§ 101 and/or 112.

The Defendants reserve all rights to revise, amend, or supplement this invalidity claim chart at a later date as discovery progresses, after the Court issues its claim construction ruling, or if Intertrust amends its infringement contentions.

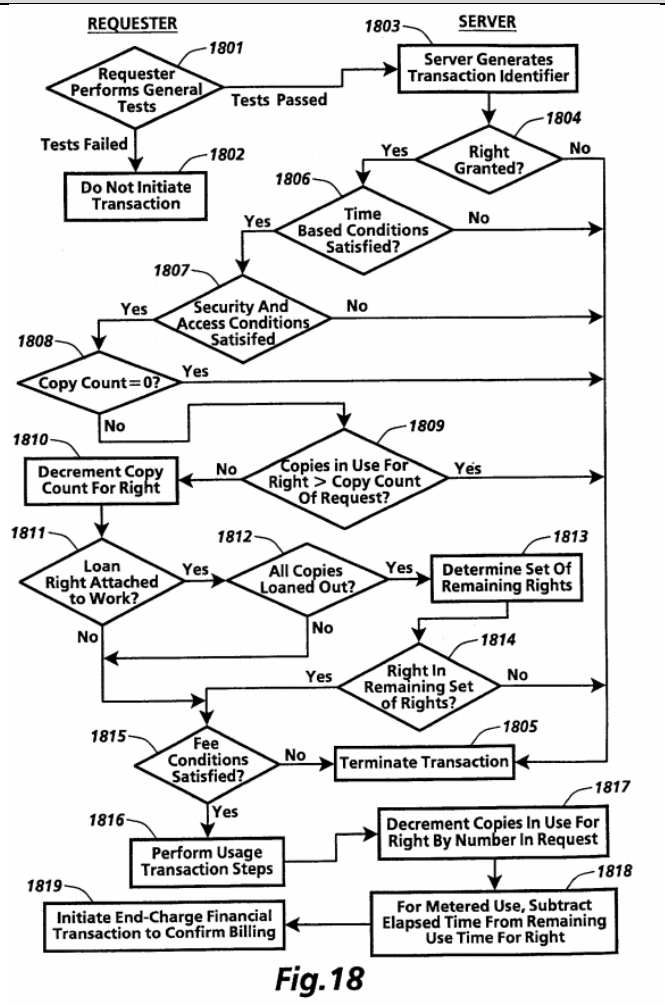
'815 Asserted Claims	Stefik
[815.42p] A method for managing at least one use of a file of electronic data, the method including:	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 3:51-4:5: A system for controlling use and distribution of digital works is disclosed. A digital work is any written, aural, graphical or video based work including computer programs that has been translated to or created in a digital form, and which can be recreated using suitable rendering means such as software programs. The present invention allows the owner of a digital work to attach usage rights to the work. The usage rights for the work define how it may be used and distributed. Digital works and their usage rights are stored in a secure repository. Digital works may only be accessed by other secure repositories. Usage rights for a digital work are embodied in a flexible and extensible usage rights grammar. Conceptually, a right in the usage rights grammar is a label attached to a predetermined behavior and conditions to exercising the right. For example, a COPY right denotes that a copy of the digital work may be made. A condition to exercising the right is the requester must pass certain security criteria. Conditions may also be attached to limit the right itself. For example, a LOAN right may be defined so as to limit the duration of which a work may be LOANed. Conditions may also include requirements that fees be paid • Stefik at 6:18-7:5: Overview A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works. The transition to digital works profoundly and fundamentally changes how creativity and commerce can work. It changes the cost of transporting or storing works because digital property is almost "massless." Digital property can be transported at electronic speeds and requires almost no warehousing. Keeping an unlimited supply of virtual copies on hand requires essentially no more space than keeping one copy on hand. The digital medium also lowers the costs of alteration, reuse and billing. There is a market for digital works because creators are strongly motivated to reuse portions of digital works from others rather than creating their own completely. This is because it is usually

'815 Asserted Claims	Stefik
	so much easier to use an existing stock photo or music clip than to create a new one from scratch. Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised. A Glossary of the terms used herein is provided at the end of the specification. A key feature of the present invention is that usage rights are permanently "attached" to the digital work. Copies made of a digital work will also have usage rights attached. Thus, the usage rights and any associated fees assigned by a creator and subsequent distributor will always remain with a digital work. The enforcement elements of the present invention are embodied in repositories. Among other things, repositories are used to store digital works, control access to digital works, bill for access to digital works and maintain the security and integrity of the system. The combination of attached usage rights and repositories enable distinct advantages over prior systems. As noted in the prior art, payment of fees are primarily for the initial access. In such approaches, once a work has been read, computational control over that copy is gone. Metaphorically, "the content genie is out of the bottle and no more fees can be billed." In contrast, the present invention never separates the fee descriptions from the work. Thus, the digital work genie only moves from one trusted bottle (repository) to another, and all uses of copies are potentially controlled and billable. • Stefik at 7:6-37: FIG. 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to FIG. 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the

'815 Asserted Claims	Stefik
	digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. Assuming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

’815 Asserted Claims	Stefik
	 *Fig. 1* • Stefik at 42:25-43:4: The Install Transaction An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

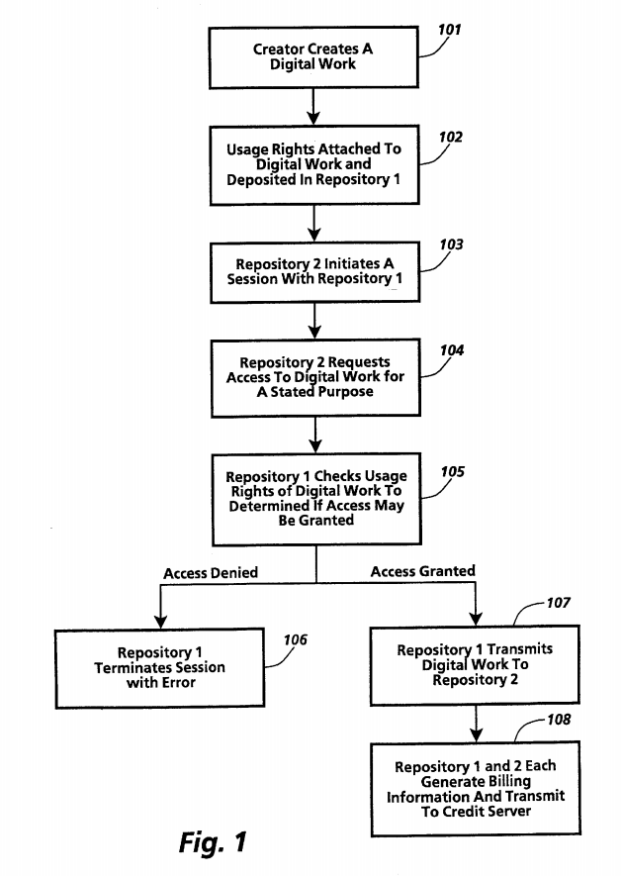
'815 Asserted Claims	Stefik
	The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.) The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps.

’815 Asserted Claims	Stefik
	 *Fig.18* • Stefik at 5:56-6:14: Examples of Sets of Usage Rights REPOSITORY TRANSACTIONS

'815 Asserted Claims	Stefik
	Message Transmission Session Initiation Transactions Billing Transactions Usage Transactions Transmission Protocol The Copy Transaction The Transfer Transaction The Loan Transaction The Play Transaction The Print Transaction The Backup Transaction The Restore Transaction The Delete Transaction The Folder Transaction The Extract Transaction The Edit Transaction The Authorization Transaction The Install Transaction The Uninstall Transaction *See also* Abstract, 6:36-56, 51:1-5, 51:64-67.

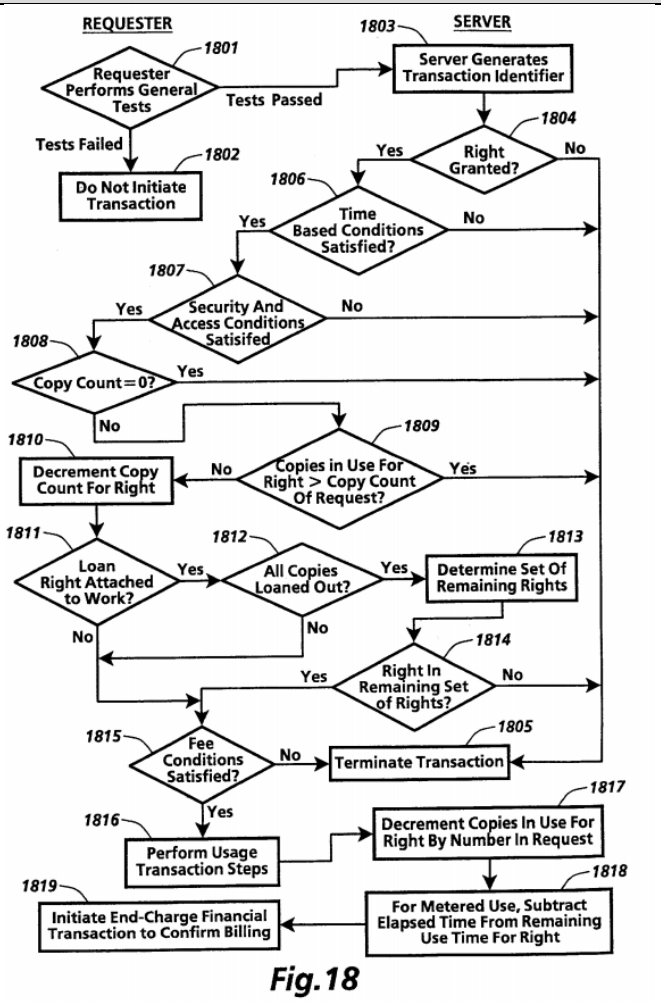
'815 Asserted Claims	Stefik
	To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42a] receiving a request to use the file in a predefined manner;	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 6:18-7:5: Overview A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works. The transition to digital works profoundly and fundamentally changes how creativity and commerce can work. It changes the cost of transporting or storing works because digital property is almost "massless." Digital property can be transported at electronic speeds and requires almost no warehousing. Keeping an unlimited supply of virtual copies on hand requires essentially no more space than keeping one copy on hand. The digital medium also lowers the costs of alteration, reuse and billing. There is a market for digital works because creators are strongly motivated to reuse portions of digital works from others rather than creating their own completely. This is because it is usually so much easier to use an existing stock photo or music clip than to create a new one from scratch. Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised. A Glossary of the terms used herein is provided at the end of the specification.

'815 Asserted Claims	Stefik
	A key feature of the present invention is that usage rights are permanently "attached" to the digital work. Copies made of a digital work will also have usage rights attached. Thus, the usage rights and any associated fees assigned by a creator and subsequent distributor will always remain with a digital work. The enforcement elements of the present invention are embodied in repositories. Among other things, repositories are used to store digital works, control access to digital works, bill for access to digital works and maintain the security and integrity of the system. The combination of attached usage rights and repositories enable distinct advantages over prior systems. As noted in the prior art, payment of fees are primarily for the initial access. In such approaches, once a work has been read, computational control over that copy is gone. Metaphorically, "the content genie is out of the bottle and no more fees can be billed." In contrast, the present invention never separates the fee descriptions from the work. Thus, the digital work genie only moves from one trusted bottle (repository) to another, and all uses of copies are potentially controlled and billable. • Stefik at 7:6-37: FIG. 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to FIG. 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. Assuming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the

'815 Asserted Claims	Stefik
	digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.  Fig. 1

'815 Asserted Claims	Stefik
	• Stefik at 36:28-64: The Play Transaction A play transaction is a request to use the contents of a work. Typically, to "play" a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository. This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent. The requester sends the server a message to initiate the play transaction. This message indicates the work to be played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work. The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory. The repositories perform the common opening transaction steps. The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player. When the player is finished, the player and the requester remove the contents from their memory. The repositories perform the common closing transaction steps. • Stefik at 42:25-43:4: The Install Transaction

'815 Asserted Claims	Stefik
	An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed. The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.) The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps.

’815 Asserted Claims	Stefik
	 *Fig.18* • Stefik at 30:56-40:18: The Extract Transaction

'815 Asserted Claims	Stefik
	A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work. The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved. The repositories perform the common opening transaction steps. The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested. The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work. The repositories perform the common closing transaction steps. • Stefik at 5:56-6:14: Examples of Sets of Usage Rights REPOSITORY TRANSACTIONS Message Transmission Session Initiation Transactions Billing Transactions Usage Transactions Transmission Protocol

’815 Asserted Claims	Stefik
	The Copy Transaction The Transfer Transaction The Loan Transaction The Play Transaction The Print Transaction The Backup Transaction The Restore Transaction The Delete Transaction The Folder Transaction The Extract Transaction The Edit Transaction The Authorization Transaction The Install Transaction The Uninstall Transaction • Stefik at 34:37-67: The Copy Transaction A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work. The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.

'815 Asserted Claims	Stefik
	The repositories perform the common opening transaction steps. The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested. The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work. The repositories perform the common closing transaction steps. *See also* Abstract, 3:51-61, 4:13-23, 6:36-56, 51:1-5, 51:64-67. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42b] retrieving at least one digital signature and at least one check value associated with the file;	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 13:11-41: Communications integrity refers to the integrity of the communications channels between repositories. Roughly speaking, communications integrity means that repositories cannot be easily fooled by "telling them lies." Integrity in this case refers to the property that repositories will only communicate with other devices that are able to present proof that they are certified repositories, and furthermore, that the repositories monitor the communications to detect "impostors" and malicious or accidental interference. Thus the security measures involving encryption, exchange of digital certificates, and nonces described below are all security measures aimed at reliable communication in a world known to contain active adversaries.

'815 Asserted Claims	Stefik
	Behavioral integrity refers to the integrity in what repositories do. What repositories do is determined by the software that they execute. The integrity of the software is generally assured only by knowledge of its source. Restated, a user will trust software purchased at a reputable computer store but not trust software obtained off a random (insecure) server on a network. Behavioral integrity is maintained by requiring that repository software be certified and be distributed with proof of such certification, i.e. a digital certificate. The purpose of the certificate is to authenticate that the software has been tested by an authorized organization, which attests that the software does what it is supposed to do and that it does not compromise the behavioral integrity of a repository. If the digital certificate cannot be found in the digital work or the master repository which generated the certificate is not known to the repository receiving the software, then the software cannot be installed. • Stefik at 42:25-43:4: The Install Transaction An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed. The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)

'815 Asserted Claims	Stefik
	The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42c] verifying the authenticity of the at least one check value using the digital signature;	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 13:11-41: Communications integrity refers to the integrity of the communications channels between repositories. Roughly speaking, communications integrity means that repositories cannot be easily fooled by "telling them lies." Integrity in this case refers to the property that repositories will only communicate with other devices that are able to present proof that they are certified repositories, and furthermore, that the repositories monitor the communications to detect "impostors" and malicious or accidental interference. Thus the security measures involving encryption, exchange of digital certificates, and nonces described below are all security measures aimed at reliable communication in a world known to contain active adversaries.

'815 Asserted Claims	Stefik
	Behavioral integrity refers to the integrity in what repositories do. What repositories do is determined by the software that they execute. The integrity of the software is generally assured only by knowledge of its source. Restated, a user will trust software purchased at a reputable computer store but not trust software obtained off a random (insecure) server on a network. Behavioral integrity is maintained by requiring that repository software be certified and be distributed with proof of such certification, i.e. a digital certificate. The purpose of the certificate is to authenticate that the software has been tested by an authorized organization, which attests that the software does what it is supposed to do and that it does not compromise the behavioral integrity of a repository. If the digital certificate cannot be found in the digital work or the master repository which generated the certificate is not known to the repository receiving the software, then the software cannot be installed. • Stefik at 42:25-43:4: The Install Transaction An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed. The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)

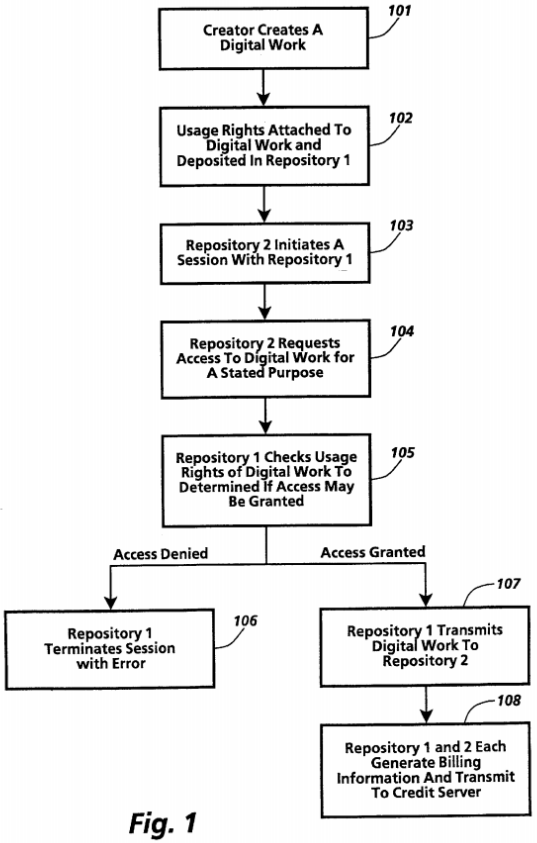
'815 Asserted Claims	Stefik
	The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42d] verifying the authenticity of at least a portion of the file using the at least one check value; and	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 13:11-41: Communications integrity refers to the integrity of the communications channels between repositories. Roughly speaking, communications integrity means that repositories cannot be easily fooled by "telling them lies." Integrity in this case refers to the property that repositories will only communicate with other devices that are able to present proof that they are certified repositories, and furthermore, that the repositories monitor the communications to detect "impostors" and malicious or accidental interference. Thus the security measures involving encryption, exchange of digital certificates, and nonces described below are all security measures aimed at reliable communication in a world known to contain active adversaries.

'815 Asserted Claims	Stefik
	Behavioral integrity refers to the integrity in what repositories do. What repositories do is determined by the software that they execute. The integrity of the software is generally assured only by knowledge of its source. Restated, a user will trust software purchased at a reputable computer store but not trust software obtained off a random (insecure) server on a network. Behavioral integrity is maintained by requiring that repository software be certified and be distributed with proof of such certification, i.e. a digital certificate. The purpose of the certificate is to authenticate that the software has been tested by an authorized organization, which attests that the software does what it is supposed to do and that it does not compromise the behavioral integrity of a repository. If the digital certificate cannot be found in the digital work or the master repository which generated the certificate is not known to the repository receiving the software, then the software cannot be installed. • Stefik at 42:25-43:4: The Install Transaction An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed. The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)

'815 Asserted Claims	Stefik
	The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.42e] granting the request to use the file in the predefined manner.	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • Stefik at 6:18-7:5: Overview A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works. The transition to digital works profoundly and fundamentally changes how creativity and commerce can work. It changes the cost of transporting or storing works because digital property is almost "massless." Digital property can be transported at electronic speeds and requires almost no warehousing. Keeping an unlimited supply of virtual copies on hand requires essentially no

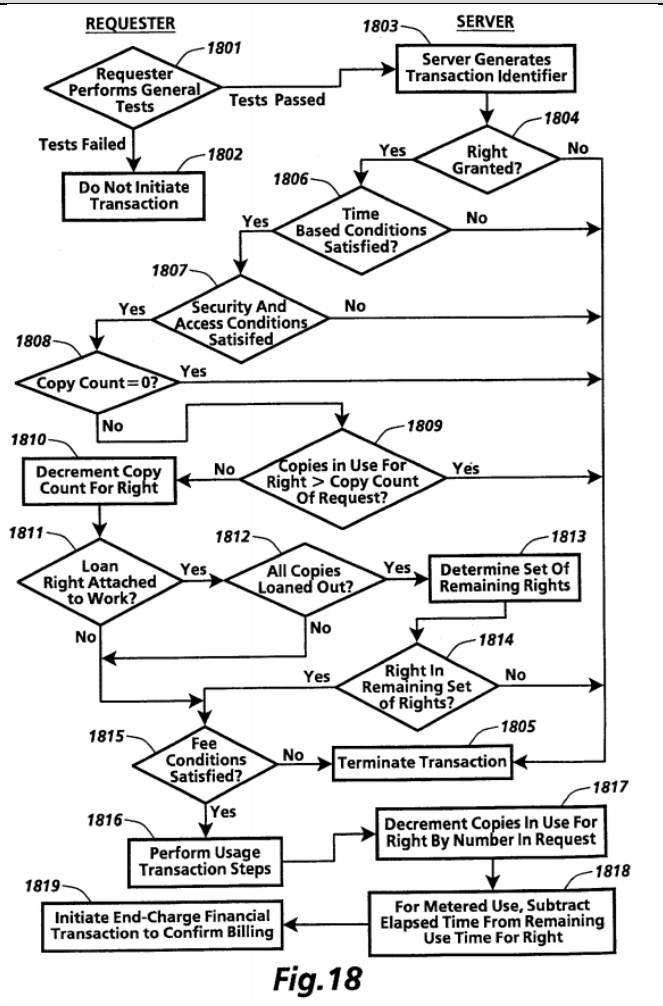
'815 Asserted Claims	Stefik
	more space than keeping one copy on hand. The digital medium also lowers the costs of alteration, reuse and billing. There is a market for digital works because creators are strongly motivated to reuse portions of digital works from others rather than creating their own completely. This is because it is usually so much easier to use an existing stock photo or music clip than to create a new one from scratch. Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised. A Glossary of the terms used herein is provided at the end of the specification. A key feature of the present invention is that usage rights are permanently "attached" to the digital work. Copies made of a digital work will also have usage rights attached. Thus, the usage rights and any associated fees assigned by a creator and subsequent distributor will always remain with a digital work. The enforcement elements of the present invention are embodied in repositories. Among other things, repositories are used to store digital works, control access to digital works, bill for access to digital works and maintain the security and integrity of the system. The combination of attached usage rights and repositories enable distinct advantages over prior systems. As noted in the prior art, payment of fees are primarily for the initial access. In such approaches, once a work has been read, computational control over that copy is gone. Metaphorically, "the content genie is out of the bottle and no more fees can be billed." In contrast, the present invention never separates the fee descriptions from the work. Thus, the digital work genie only moves from one trusted bottle (repository) to another, and all uses of copies are potentially controlled and billable.

'815 Asserted Claims	Stefik
	• Stefik at 7:6-37: FIG. 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to FIG. 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. Assuming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

’815 Asserted Claims	Stefik
	 *Fig. 1* • Stefik at 36:28-64: The Play Transaction A play transaction is a request to use the contents of a work. Typically, to “play” a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any

'815 Asserted Claims	Stefik
	other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository. This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent. The requester sends the server a message to initiate the play transaction. This message indicates the work to be played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work. The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory. The repositories perform the common opening transaction steps. The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player. When the player is finished, the player and the requester remove the contents from their memory. The repositories perform the common closing transaction steps. • Stefik at 42:25-43:4: The Install Transaction An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

'815 Asserted Claims	Stefik
	The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size). The repositories perform the common opening transaction steps. The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error. The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.) The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.) The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.) The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions. The repositories perform the common closing transaction steps.

’815 Asserted Claims	Stefik
	 *Fig.18* • Stefik at 30:56-40:18: The Extract Transaction

'815 Asserted Claims	Stefik
	A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work. The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved. The repositories perform the common opening transaction steps. The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested. The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work. The repositories perform the common closing transaction steps. • Stefik at 5:56-6:14: Examples of Sets of Usage Rights REPOSITORY TRANSACTIONS Message Transmission Session Initiation Transactions Billing Transactions Usage Transactions Transmission Protocol

'815 Asserted Claims	Stefik
	The Copy Transaction The Transfer Transaction The Loan Transaction The Play Transaction The Print Transaction The Backup Transaction The Restore Transaction The Delete Transaction The Folder Transaction The Extract Transaction The Edit Transaction The Authorization Transaction The Install Transaction The Uninstall Transaction • Stefik at 34:37-67: The Copy Transaction A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work. The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.

'815 Asserted Claims	Stefik
	The repositories perform the common opening transaction steps. The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested. The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work. The repositories perform the common closing transaction steps. • Stefik at 4:13-23: Generally, a repository will process each request to access a digital work by examining the work's usage rights. For example, in a request to make a copy of a digital work, the digital work is examined to see if rights have been granted which would allow copies to be given out. If such a right has been granted, then conditions to exercise of the right are checked (e.g. a right to make 2 copies). If conditions associated with the right are satisfied, the copy can be made. • Stefik at 7:57-63: Communication with an authorization repository 202 may occur when a digital work being accessed has a condition requiring an authorization. Conceptually, an authorization is a digital certificate such that possession of the certificate is required to gain access to the digital work. An authorization is itself a digital work that can be moved between repositories and subjected to fees and usage rights conditions. • Stefik at Cl. 1: determining if a request for access to a digital work stored in said storage means may be granted • Stefik at Cl. 19:

'815 Asserted Claims	Stefik
	if said particular usage right is one of said usage rights associated with said digital work, granting access to said digital work and performing usage transaction steps associated with said particular usage right. *See also* 6:36-56, 7:24-37, 51:1-5, 51:64-67, Fig. 1. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.
[815.43] A method as in claim 42, in which the at least one check value comprises one or more hash values, and in which verifying the authenticity of at least a portion of the file using the at least one check value includes: hashing at least a portion of the file to obtain a first hash value; and comparing the first hash value to at least one of the one or more hash values.	Stefik discloses this limitation based on at least the following citations. *See, e.g.,* • *See* Claim [815.42d] above. To the extent that Intertrust contends that Stefik does not disclose this limitation, a person of ordinary skill in the art would have readily arrived at this limitation by combining or modifying Stefik in view of their general knowledge and/or the prior art references disclosed in the corresponding section of Defendants' Invalidity Contentions.

APPENDIX C-Combination
Invalidity Claim Chart for U.S. Patent No. 6,785,815

Asserted claims 42 and 43 of U.S. Patent No. 6,785,815 ("the '815 Patent") are invalid as expressly and/or inherently anticipated by the prior art described in the invalidity claim charts in Appendices C.

To the extent any of the prior art described in the invalidity claim charts in Appendices C is found not to anticipate, expressly or inherently, any asserted claim of the '815 Patent, the claim is invalid as obvious in view of that prior art, alone or in combination with other prior art references, including but not limited to the prior art identified in the Defendants' Invalidity Contentions and the prior art with any other prior art identified in Defendants' Invalidity Contentions and/or listed in this claim chart.

This invalidity claim chart is based in whole or in part on the Defendants' present understanding of the asserted claims, their current construction of the claims, and/or Intertrust's apparent construction of the claims in its current infringement contentions. The Defendants' are not adopting Intertrust's apparent claim construction, nor admitting to the accuracy of any particular claim construction. To the extent that Intertrust's apparent claim construction or applications thereof are reflected in this invalidity claim chart, nothing herein should be construed as an admission that the Defendants agree with Intertrust's apparent claim construction or Intertrust's application of that claim construction in Intertrust's current infringement contentions.

The use of this reference or combinations of references as invalidating prior art under 35 U.S.C. §§ 102 and/or 103 may be based on Intertrust's allegations of infringement. Defendants do not necessarily agree with the interpretations set forth in Intertrust's infringement contentions and thus this invalidity claim chart is not an admission that the accused products meet any particular claim element or infringe the asserted claim. In addition, nothing in this invalidity claim chart should be interpreted as a position about whether any portion of the asserted claim is limiting or not. Further, by submitting this invalidity claim chart, the Defendants do not waive and hereby expressly reserve their right to raise other invalidity defenses, including but not limited to defenses under 35 U.S.C. §§ 101 and/or 112.

The descriptions of prior art systems identified in Defendants' Invalidity Contentions and/or claim charts are stated on information and belief and are supported by the information and documents that will be produced by Defendants and/or third parties. Pending further discovery, any of the prior art systems identified by Defendants, including Intertrust DigiBox and IBM Cryptolope teach and/or render obvious each asserted claim, alone or in combination with prior art identified in Defendants' Invalidity Contentions and/or listed in this claim chart.

The Defendants reserve all rights to revise, amend, or supplement this invalidity claim chart at a later date as discovery progresses, after the Court issues its claim construction ruling, or if Intertrust amends its infringement contentions.

'815 Asserted Claims	Prior Art
[815.42p] A method for managing at least one use of a file of electronic data, the method including:	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik Friedman Sprunk Ginter Yeung Morishita Miwa DigiBox Cryptolope HotJava
[815.42a] receiving a request to use the file in a predefined manner;	Davis Rabin Hanna Gennaro Rohatgi Renaud

'815 Asserted Claims	Prior Art
	Stefik Ginter Yeung Morishita Miwa DigiBox Cryptolope HotJava
[815.42b] retrieving at least one digital signature and at least one check value associated with the file;	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik Friedman Sprunk Ginter Schneier FIPS PUB Miwa DigiBox

’815 Asserted Claims	Prior Art
	Cryptolope HotJava
[815.42c] verifying the authenticity of the at least one check value using the digital signature;	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik Friedman Sprunk Ginter Schneier FIPS PUB HotJava
[815.42d] verifying the authenticity of at least a portion of the file using the at least one check value; and	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik

’815 Asserted Claims	Prior Art
	Friedman Sprunk Ginter Schneier FIPS PUB Miwa DigiBox Cryptolope HotJava
[815.42e] granting the request to use the file in the predefined manner.	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik Ginter Yeung Morishita Miwa DigiBox Cryptolope

'815 Asserted Claims	Prior Art
	HotJava
[815.43] A method as in claim 42, in which the at least one check value comprises one or more hash values, and in which verifying the authenticity of at least a portion of the file using the at least one check value includes: hashing at least a portion of the file to obtain a first hash value; and comparing the first hash value to at least one of the one or more hash values.	Davis Rabin Hanna Gennaro Rohatgi Renaud Stefik Ginter Friedman Sprunk Schneier FIPS PUB Miwa DigiBox Cryptolope HotJava