US 20090281946A1

(19) United States

(12) Patent Application Publication
Davis et al.

(10) Pub. No.: US 2009/0281946 A1

(43) Pub. Date: Nov. 12, 2009

(54) ACH PAYMENT PROCESSING

(76) Inventors: **Peter A. Davis**, Plymouth, MN (US); **Keith Pierce**, Lauderdale, MN (US); **Stephen P. Hanten**, Minneapolis, MN (US); **Erik Tennant**, Apple Valley, MN (US); **Jennifer Larson**, Elk River, MN (US)

Correspondence Address:
KING & SPALDING
1180 PEACHTREE STREET , NE
ATLANTA, GA 30309-3521 (US)

(21) Appl. No.: **12/119,262**

(22) Filed: **May 12, 2008**

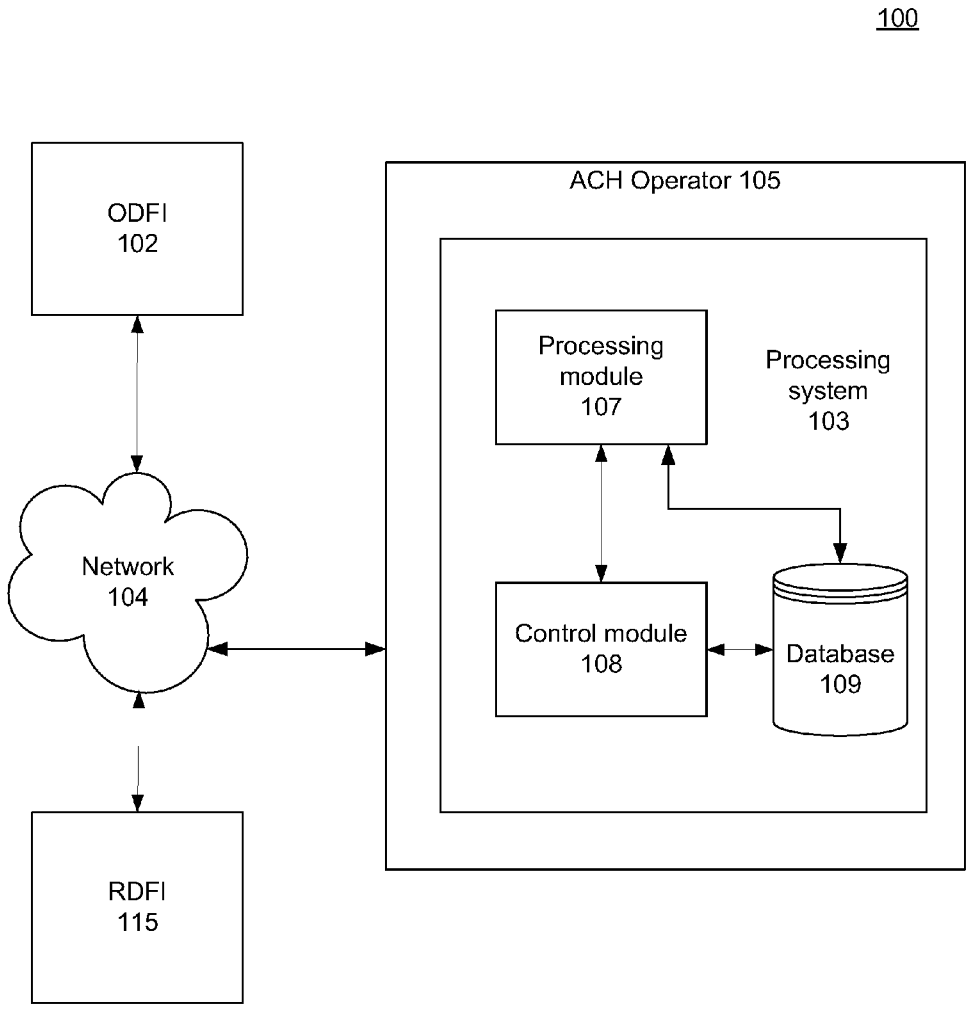
Publication Classification

(51) **Int. Cl.**
G06Q 40/00 (2006.01)

(52) **U.S. Cl.** **705/40**

(57) **ABSTRACT**

Efficiently processing ACH payments by processing batches of ACH payments in parallel. A processing system of an ACH operator receives an ACH file including multiple batches of ACH items. Each batch includes at least one ACH item. A control module of the processing system organizes data in the ACH file into multiple partitions according to a selected strategy. Each partition includes at least one of the batches. A processing module of the processing system separately processes each partition in parallel, validating the batches and ACH items and creating at least one output batch for each partition. If the control module determines that the ACH file is acceptable, based on this parallel processing, then the processing module settles the ACH items in the output batches and creates at least one new ACH file for transmitting the settled ACH items to one or more corresponding receiving depository financial institutions.



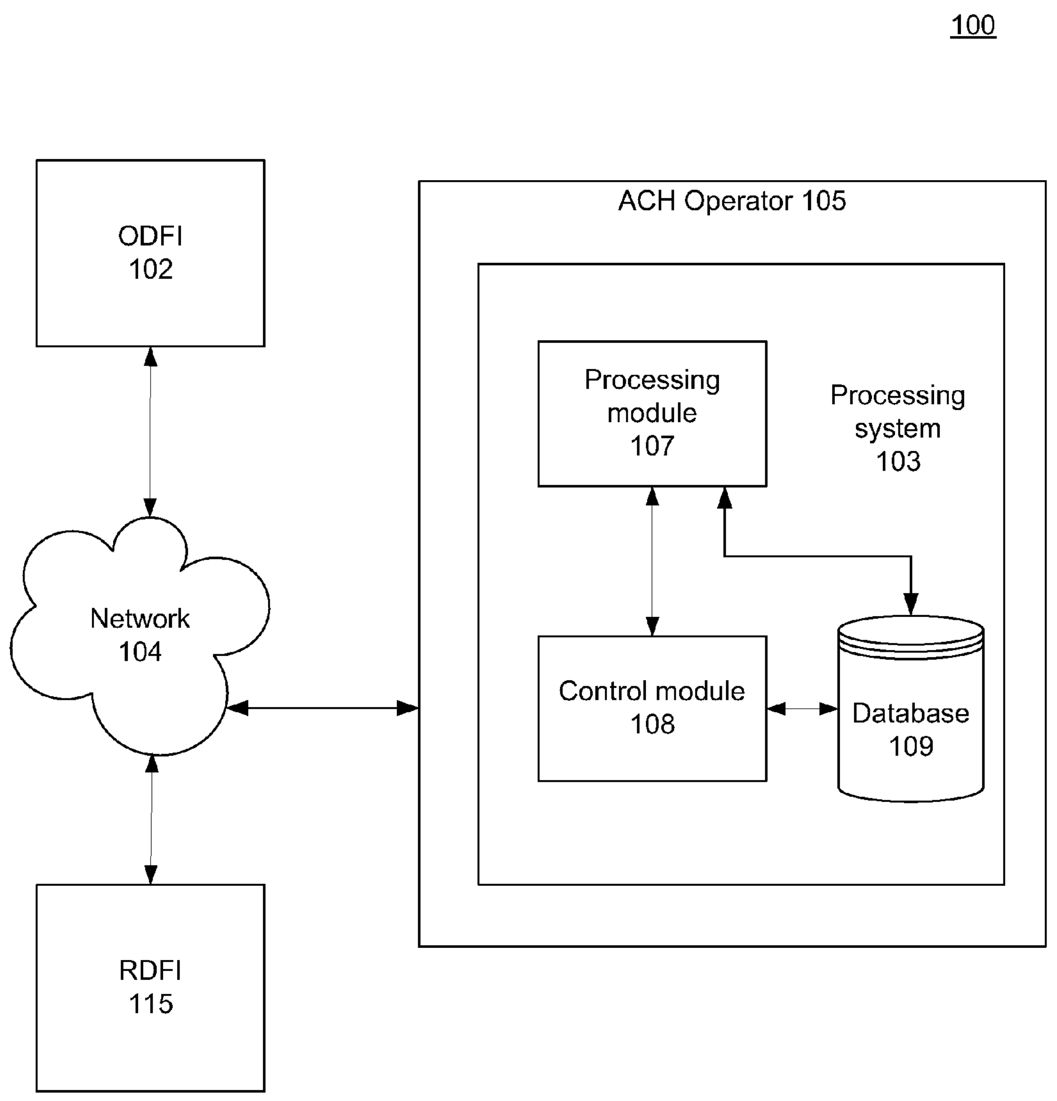

100
ODFI
102
ACH Operator 105
Processing module
107
Processing system
103
Network
104
Control module
108
Database
109
RDFI
115

Fig. 1

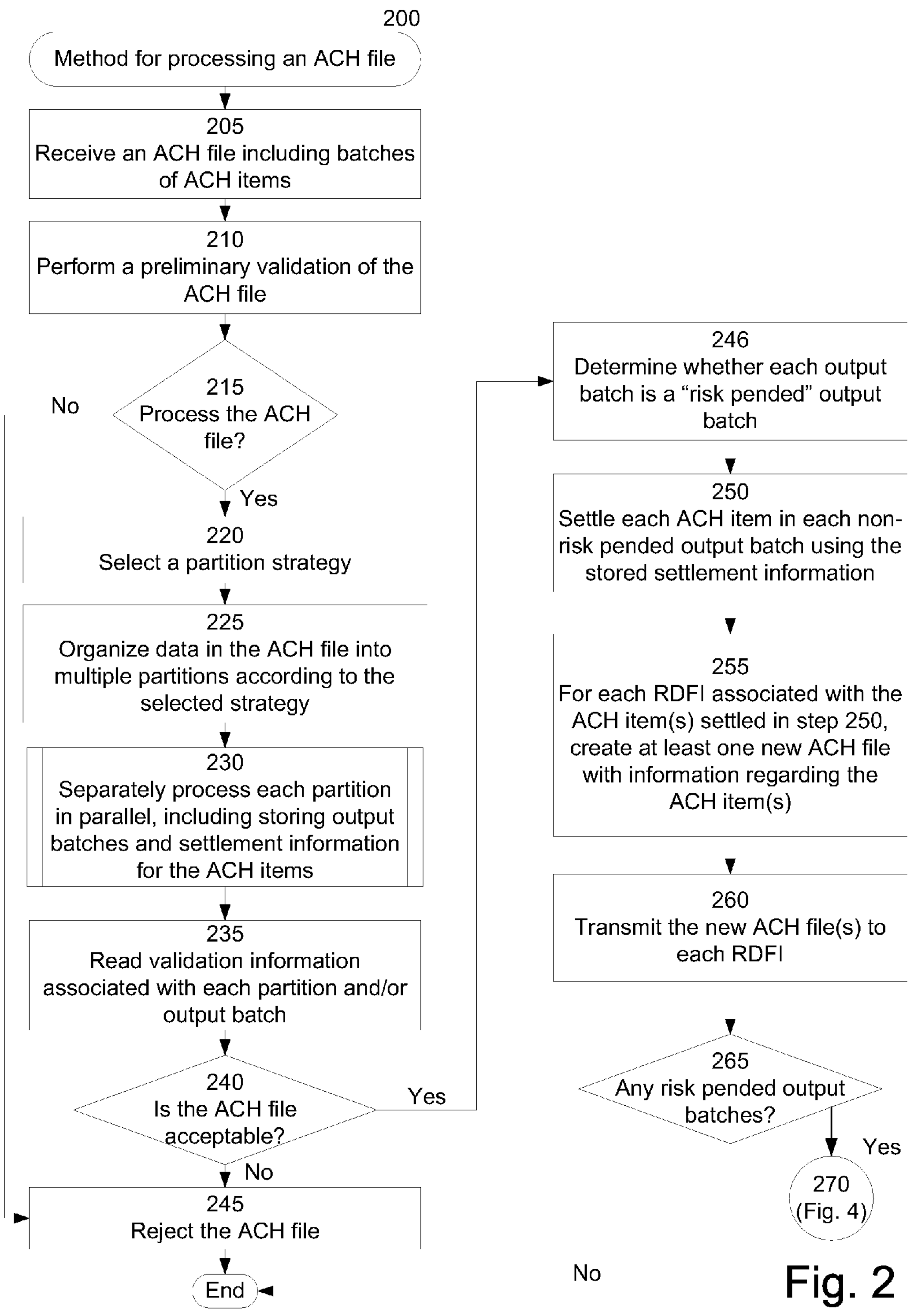

200
Method for processing an ACH file
205
Receive an ACH file including batches of ACH items
210
Perform a preliminary validation of the ACH file
215
Process the ACH file?
No
Yes
220
Select a partition strategy
225
Organize data in the ACH file into multiple partitions according to the selected strategy
230
Separately process each partition in parallel, including storing output batches and settlement information for the ACH items
235
Read validation information associated with each partition and/or output batch
240
Is the ACH file acceptable?
Yes
No
245
Reject the ACH file
End
246
Determine whether each output batch is a "risk pended" output batch
250
Settle each ACH item in each non-risk pended output batch using the stored settlement information
255
For each RDFI associated with the ACH item(s) settled in step 250, create at least one new ACH file with information regarding the ACH item(s)
260
Transmit the new ACH file(s) to each RDFI
265
Any risk pended output batches?
Yes
270
(Fig. 4)
No

Fig. 2

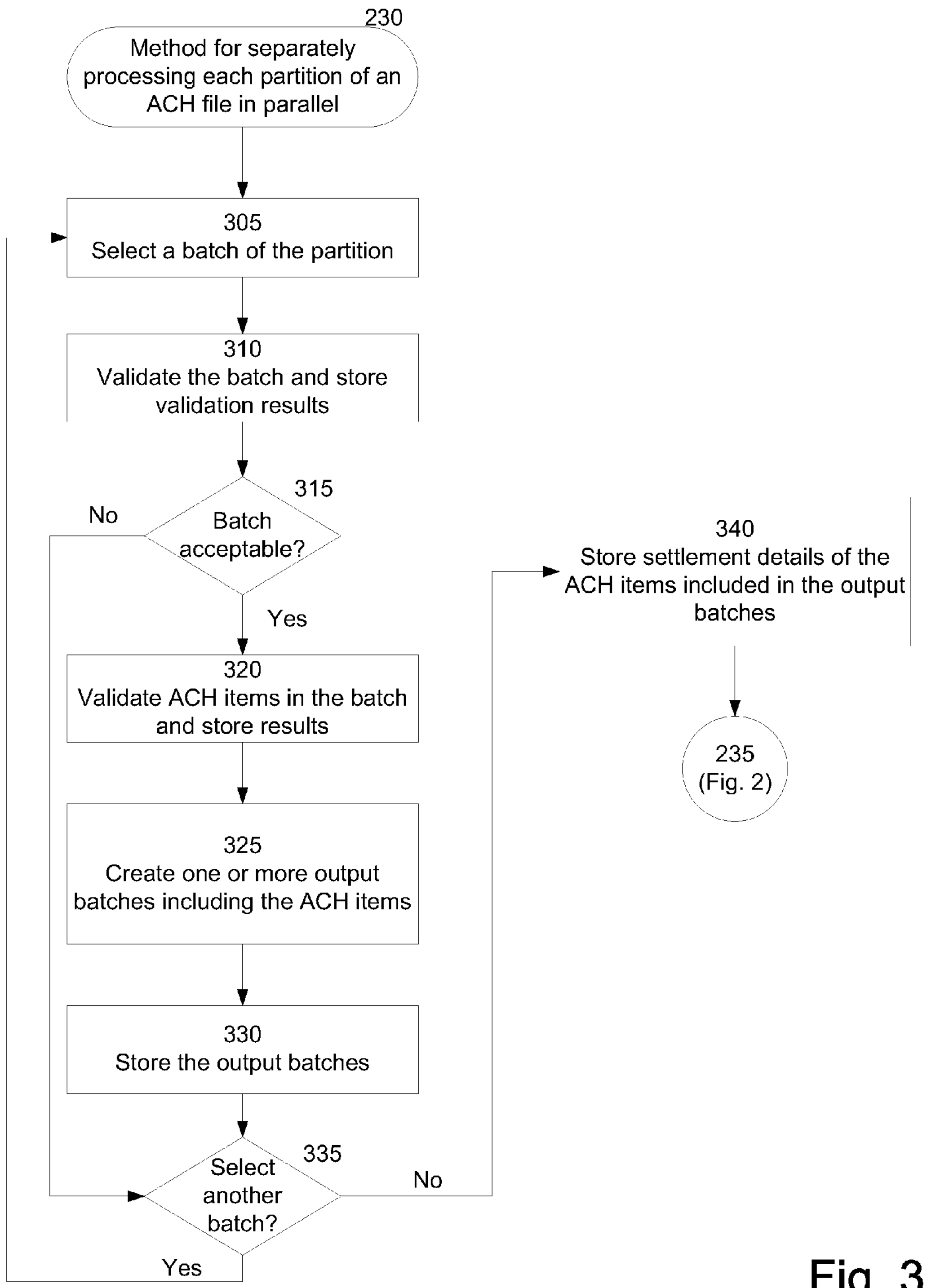

230
Method for separately processing each partition of an ACH file in parallel
305
Select a batch of the partition
310
Validate the batch and store validation results
315
Batch acceptable?
No
Yes
320
Validate ACH items in the batch and store results
325
Create one or more output batches including the ACH items
330
Store the output batches
335
Select another batch?
No
Yes
340
Store settlement details of the ACH items included in the output batches
235
(Fig. 2)

Fig. 3

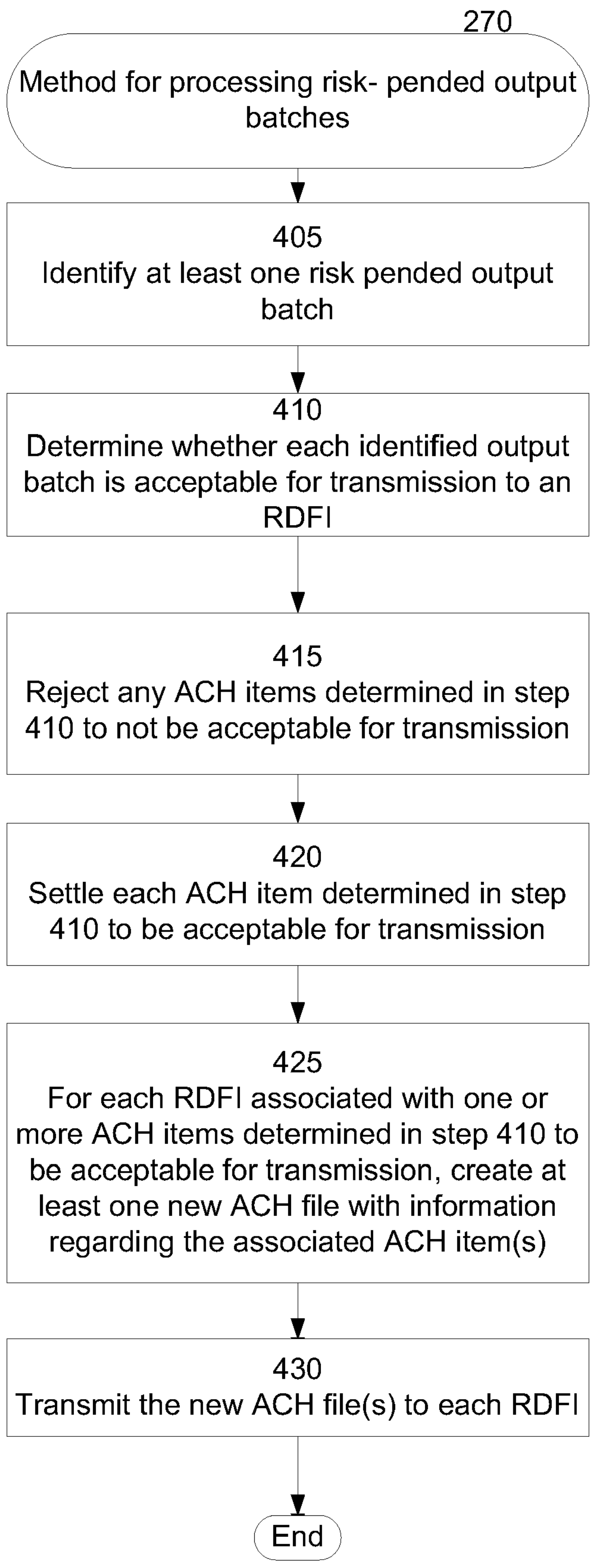

270
Method for processing risk- pended output batches
405
Identify at least one risk pended output batch
410
Determine whether each identified output batch is acceptable for transmission to an RDFI
415
Reject any ACH items determined in step 410 to not be acceptable for transmission
420
Settle each ACH item determined in step 410 to be acceptable for transmission
425
For each RDFI associated with one or more ACH items determined in step 410 to be acceptable for transmission, create at least one new ACH file with information regarding the associated ACH item(s)
430
Transmit the new ACH file(s) to each RDFI
End

Fig. 4

ACH PAYMENT PROCESSING

TECHNICAL FIELD

[0001] The invention relates generally to processing payments in an Automated Clearinghouse ("ACH"), and more particularly to efficiently processing ACH payments by processing batches of ACH payments in parallel.

BACKGROUND

[0002] Financial institutions are increasingly clearing financial transactions using electronic systems such as the Automated Clearinghouse ("ACH") network. The ACH network is a nationwide electronic funds transfer system supported by several operators, including the Federal Reserve Banks and other institutions. The ACH network is governed by a set of rules administered by the National Automated Clearinghouse Association ("NACHA"). ACH offers financial institutions, companies, consumers, and others an efficient alternative to paper based payment methods.

[0003] In ACH, an originator sends electronic transaction items to an originating depository financial institution ("ODFI"). The originator is a person or organization that agrees to initiate ACH payments in the ACH network according to an arrangement with another, receiving person or entity (a "receiver"). For example, the originator can be a company that agrees to originate an ACH payment to an account of a consumer.

[0004] The ODFI packages the transaction items in one or more batched ACH files, according to the NACHA rules. The ODFI transmits the ACH files to an ACH operator. The ODFI may send the ACH files to the ACH operator directly or via a third party or "remote" sending point. The third party or remote sending point may be a depository financial institution or a company providing processing services for a depository financial institution. The term "ACH file" is used herein to refer to any collection of batched and/or unbatched ACH transaction items.

[0005] The ACH operator is a Federal Reserve Bank or other entity that receives ACH files from an ODFI and distributes transaction items within the ACH files to at least one corresponding receiving depository financial institution ("RDFI") associated with a receiver. In some cases, the ACH operator also may perform settlement functions (crediting and debiting of accounts) for the affected financial institutions.

[0006] Each ACH file includes at least one batch of transaction items. Each batch includes one or more transaction items. The terms "transaction item" and "ACH item" are used interchangeably herein to refer to any batched electronic payment or payment instruction, whether international or domestic, and/or information associated with a batched electronic payment or payment instruction. For example, a payment or payment instruction can be a credit, a debit, or a rejected or returned transaction.

[0007] Upon receiving an ACH file, processors of the ACH operator sort, batch, and re-assemble the transaction items in the ACH file in at least one new ACH file for delivery to one or more RDFI's. The RDFI's may receive the ACH files directly or via one or more third party or "remote" receiving points. Each third party or remote receiving point may be a depository financial institution or a company providing processing services for a depository financial institution. The RDFI's forward the transaction items in the ACH files to their corresponding receivers. Each receiver is a person or organization that has authorized an originator to initiate an ACH payment to an account of the receiver, at the receiver's RDFI.

[0008] Traditionally, ACH operators include multiple mainframe processors, which process ACH files, file by file, on a first in, first out basis. Each ACH file is processed by a single one of the mainframe processors. Because each ACH file can include a varying number of batches with a varying number of transaction items, each mainframe processor can bear a different processing load. For example, one mainframe processor may process an ACH file with only a single transaction item, while another mainframe processor may process an ACH file with multiple batches including large amounts of transaction items. Thus, at least some mainframe processors may be over-loaded or under-loaded. This results in many processing inefficiencies, including ineffective use of the under-loaded mainframe processors and delays in processing of transaction items handled by the over-loaded mainframe processors.

[0009] Therefore, a need exists in the art for a system and method for efficiently processing ACH payments.

SUMMARY

[0010] The invention provides systems and methods for processing ACH payments. In particular, the invention provides systems and methods for efficiently processing ACH payments by processing batches of ACH payments in parallel.

[0011] An ACH processing system of an ACH operator receives an ACH file including multiple batches of ACH items. For example, the ACH processing system can receive the ACH file from an ODFI or an operator acting on behalf of an ODFI. Each batch of the ACH file includes at least one ACH item.

[0012] A control module of the ACH processing system organizes data in the ACH file into multiple partitions according to a selected strategy. Each partition includes at least one of the batches. For example, the control module can select a strategy that assigns (a) a fixed number of the batches to each partition, (b) a substantially equal number of batches to each partition, or (c) a substantially equal number of ACH items to each partition. Other strategies for organizing batched data are well known to a person of ordinary skill in the art having the benefit of the present disclosure.

[0013] A processing module of the ACH processing system separately processes each partition in parallel. For example, each of multiple processors associated with the processing module can process at least one of the partitions. Alternatively, multiple of the partitions may be processed on the same processor using application threading. For example, the processing module can utilize one or more java objects, such as IBM WebSphere Application Server's "asynchronous beans" running on a Java 2 Platform Enterprise Edition ("J2EE") application, to perform the application threading. The processing module can modify threading parameters in the J2EE application to ensure that each processor is not over-loaded or under-loaded during processing.

[0014] During processing, the processing module validates the batches and ACH items and creates at least one output batch for each partition. Each output batch includes information regarding at least one of the ACH items. For example, a particular output batch can include multiple ACH items associated with the same ODFI and/or RDFI. The processing module also may store settlement information for each ACH item.

[0015] The output batches can include ACH items that were successfully validated during processing or ACH items that were not successfully validated during processing. For example, "risk pended" output batches can include information regarding ACH items that were not successfully validated during processing, and non-risk pended output batches can include information regarding ACH items that were successfully validated during processing.

[0016] The control module evaluates the validation results for each partition to determine whether the ACH file is acceptable. If the control module determines that the ACH file is not acceptable, then the ACH operator may return the ACH file to the ODFI, suspend processing of the ACH file, and/or output a notification advising the ODFI and/or an operator of the ACH processing system that the ACH file will not be processed. If the control module determines that the ACH file is acceptable, then the processing module determines whether each output batch is a risk pended output batch or a non-risk pended output batch. The processing module settles the ACH items in the non-risk pended output batches and creates at least one new ACH file for transmitting the settled ACH items to at least one corresponding RDFI.

[0017] The processing module determines whether each risk pended output batch is acceptable for transmission to an RDFI. For example, the processing module can make that determination based on information from one or more operators of the ACH processing system and/or the ODFI. If the processing module determines that a particular risk pended output batch is not acceptable for transmission to an RDFI, then the ACH operator may return the output batch and/or the ACH items contained therein to the ODFI, suspend processing of the output batch and/or ACH items, and/or output a notification advising the ODFI and/or an operator of the ACH processing system that the output batch and/or ACH items will not be processed. If the processing module determines that the risk pended output batch is acceptable for transmission to an RDFI, then the processing module settles the ACH items in the risk pended output batch and creates at least one new ACH file for transmitting the settled ACH items to at least one corresponding RDFI.

[0018] Additional aspects, features, and advantages of the invention will become apparent to those skilled in the art upon consideration of the following detailed description of illustrated embodiments exemplifying the best mode of carrying out the invention as presently perceived.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] FIG. **1** is a block diagram depicting a system for efficiently processing an ACH file, in accordance with certain exemplary embodiments.

[0020] FIG. **2** is a flow chart depicting a method for efficiently processing an ACH file, in accordance with certain exemplary embodiments.

[0021] FIG. **3** is a flow chart depicting a method for separately processing partitions of an ACH file in parallel, in accordance with certain exemplary embodiments.

[0022] FIG. **4** is a flow chart depicting a method for processing risk pended output batches, in accordance with certain exemplary embodiments.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0023] The invention is directed to systems and methods for efficiently processing ACH payments. In particular, the invention is directed to systems and methods for efficiently processing ACH files by processing batches of ACH payments in parallel.

[0024] The invention includes a computer program that embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the invention in computer programming, and the invention should not be construed as limited to any one set of computer program instructions. Further, a skilled programmer would be able to write such a computer program to implement an embodiment of the disclosed invention based on the flow charts and associated description in the application text. Therefore, disclosure of a particular set of program code instructions is not considered necessary for an adequate understanding of how to make and use the invention. The inventive functionality of the claimed computer program will be explained in more detail in the following description read in conjunction with the figures illustrating the program flow.

[0025] Turning now to the drawings, in which like numerals indicate like elements throughout the figures, exemplary embodiments of the invention are described in detail.

[0026] FIG. **1** is a block diagram depicting a system **100** for efficiently processing an ACH file, in accordance with certain exemplary embodiments. The system **100** is described below with reference to the methods illustrated in FIGS. **2-4**.

[0027] FIG. **2** is a flow chart depicting a method **200** for efficiently processing an ACH file, in accordance with certain exemplary embodiments. The exemplary method **200** is illustrative and, in alternative embodiments of the invention, certain steps can be performed in a different order, in parallel with one another, or omitted entirely, and/or certain additional steps can be performed without departing from the scope and spirit of the invention. The method **200** is described below with reference to FIGS. **1** and **2**.

[0028] In step **205**, an ACH operator **105** receives an ACH file including multiple batches of ACH items. The ACH operator **105** is a Federal Reserve Bank or other entity that receives an ACH file from an originating depository financial institution **102** ("ODFI") and distributes transaction items within the ACH file to at least one corresponding receiving depository financial institution **115** ("RDFI"). In some cases, the ACH operator **105** also may perform settlement functions (crediting and debiting of accounts) for the affected financial institutions **102**, **115**.

[0029] In certain alternative exemplary embodiments, the ACH operator **105** can receive the ACH file from a third party or "remote" sending point (not shown) operating on behalf of the ODFI **102**. The third party or remote sending point may be a depository financial institution or a company providing processing services for a depository financial institution. The term "ACH file" is used herein to refer to any collection of batched and/or unbatched ACH items.

[0030] In certain exemplary embodiments, the ACH operator **105** can receive the ACH file via a network **104**. The network **104** can include any wired or wireless telecommunication means by which computerized devices can exchange data, including for example, a local area network (LAN), a wide area network (WAN), an intranet, an Internet, or any combination thereof. For example, the network **104** can include the ACH network.

[0031] Each batch of the received ACH file includes one or more transaction items. The terms "transaction item" and "ACH item" are used interchangeably herein to refer to any batched electronic payment or payment instruction, whether international or domestic, and/or information associated with a batched electronic payment or payment instruction. For example, a payment or payment instruction can be a credit, a debit, or a rejected or returned transaction.

[0032] In step **210**, a processing module **107** of a processing system **103** operated by the ACH operator **105** performs a preliminary validation of the ACH file. For example, the processing module **107** can determine whether the file is properly formatted and/or includes suitable information for processing. In certain exemplary embodiments, the processing module **107** can perform this preliminary validation based on one or more rules stored in a database **109** of the processing system **103**.

[0033] In step **215**, the processing module **107** determines whether to process the ACH file. For example, the processing module **107** can determine to process the ACH file if the ACH file is successfully validated in step **210**. Similarly, the processing module **107** can determine to not process the ACH file if the ACH file is not successfully validated in step **210**.

[0034] If the processing module **107** determines in step **215** to not process the ACH file, then the method **200** branches to step **245**, which is discussed below. If the processing module **107** determines in step **215** to process the ACH file, then the method **200** continues to step **220**.

[0035] In step **220**, a control module **108** of the processing system **103** selects a partition strategy. The partition strategy is a schema for dividing information within the ACH file into multiple partitions. Each partition includes one or more of the batches of the ACH file.

[0036] For example, one partition strategy can be to assign a fixed number of batches to each partition. A second partition strategy can be to assign substantially equal numbers of batches to each of the partitions. A third partition strategy can be to assign batches to the partitions so that each partition includes a substantially equal number of ACH items. A person of ordinary skill in the art having the benefit of the present disclosure will recognize that many other strategies exist for partitioning batches of data.

[0037] In step **225**, the control module **108** organizes the data in the ACH file into multiple partitions according to the strategy selected in step **220**. In step **230**, the processing module **107** separately processes each partition in parallel. For example, each of multiple processors (not shown) associated with the processing module **107** can process at least one of the partitions. Alternatively, multiple of the partitions may be processed on the same processor using application threading. For example, in certain exemplary embodiments, the processing module **107** can utilize one or more java objects, such as asynchronous beans running on a Java 2 Platform Enterprise Edition ("J2EE") application, to perform the application threading. In certain exemplary embodiments, the processing module **107** can modify threading parameters in the J2EE application to ensure that each processor is not over-loaded or under-loaded during processing.

[0038] During processing, the processing module **107** validates each batch and ACH item and creates and stores output batches and settlement information for the ACH items. For example, the processing module **107** can store the output batches and settlement information in the database **109**. Each output batch includes one or more ACH items. For example, all the ACH items in a particular output batch can be associated with the same RDFI **115** and/or ODFI **102**. The processing module **107** can use the output batches and settlement information to complete the processing of the ACH items, as described below. In certain exemplary embodiments, the processing module **107** can store validation information, such as validation results, for each batch and ACH item in the database **109** of the processing system **103**. Step **230** is described in more detail below, with reference to FIG. **3**.

[0039] In step **235**, the control module **108** reads validation information associated with each partition. For example, the control module **108** can read the validation information from the database **109** of the processing system **103**. In step **240**, the control module **108** determines whether the ACH file is acceptable, based on the validation information read in step **235**. For example, the control module **108** may determine that an ACH file is not acceptable if a large quantity of validation errors were identified in step **235**. The control module **108** also may determine that an ACH file is not acceptable if one or more significant validation errors were identified in step **235**. In certain exemplary embodiments, the processing module **107** can make the decision of step **240** based on one or more rules stored in the database **109**. In some cases, the control module **108** may determine that an ACH file is not acceptable even if the ACH file successfully completed the preliminary validation of step **215**.

[0040] If the control module **108** determines in step **240** that the ACH file is not acceptable, then the method **200** branches to step **245**. In step **245**, the control module **108** rejects the ACH file for further processing. For example, the control module **108** can return the ACH file to the ODFI **102** via the network **104**, suspend processing of the ACH file, and/or output a notification advising the ODFI **102** and/or an operator of the ACH processing system **103** that the ACH file will not be processed. In certain exemplary embodiments, the ODFI **102** and/or operator can override the rejection decision if it determines that the ACH file actually should be processed.

[0041] If the control module **108** determines in step **240** that the ACH file is acceptable, then the method **200** continues to step **246**. In step **246**, the processing module **107** determines whether each output batch (stored in step **230**) is a "risk pended" output batch or a non-risk pended output batch. A risk pended output batch is an electronic file or record that includes information regarding at least one ACH item that was not successfully validated (in step **230**). Similarly, a non-risk pended output batch is an electronic file or record that includes information regarding at least one ACH item that has been successfully validated (in step **230**). In certain exemplary embodiments, a "flag" or indicator within, or associated with, a particular output batch can indicate whether the output batch is a risk pended output batch or a non-risk pended output batch.

[0042] In step **250**, the processing module **107** settles each ACH item in the non-risk pended output batches using the settlement information stored in step **230**. For example, the processing module **107** can cause accounts associated with the ODFI **102** and RDFI **115** associated with each ACH item to be credited and/or debited in accordance with the settlement information.

[0043] In step **255**, the processing module **107** creates at least one new ACH file. Each ACH file includes information regarding at least one of the ACH items settled in step **250**. For example, each ACH file can include one or more of the non-risk pended output batches.

[0044] In certain exemplary embodiments, the processing module **107** can create at least one new ACH file for each

RDFI **115**. Each ACH file for a particular RDFI **115** can include one or more batches of ACH items associated with the RDFI **115**. In step **260**, the processing module **107** transmits each new ACH file to its corresponding RDFI **115**. For example, the processing module **107** can transmit the new ACH file(s) via the network **104**.

[0045] In step **265**, the processing module **107** determines whether any risk pended output batches were identified in step **246**. If so, then the method **200** branches to step **270**, which is described below with reference to FIG. **4**. If the processing module **107** determines that there were not any risk pended output batches identified in step **246**, then the method **200** ends.

[0046] FIG. **3** is a flow chart depicting a method **230** for separately processing partitions of an ACH file in parallel, in accordance with certain exemplary embodiments, as referred to in step **230** of FIG. **2**. The exemplary method **230** is illustrative and, in alternative embodiments of the invention, certain steps can be performed in a different order, in parallel with one another, or omitted entirely, and/or certain additional steps can be performed without departing from the scope and spirit of the invention. The method **230** is described below with reference to FIGS. **1-3**.

[0047] The processing module **107** typically performs multiple instances of the method **230** in parallel, with each instance being associated with a different one of the partitions of the ACH file. In step **305**, the processing module **107** selects a batch of the partition. In step **310**, the processing module **107** validates the batch and stores validation results in the database **109**. For example, the processing module **107** can validate the batch by determining whether the batch is properly formatted and/or includes suitable information for processing. In certain exemplary embodiments, the processing module **107** can perform this validation based on one or more rules stored in the database **109**. For example, the stored validation results can include information regarding any validation errors identified during validation and/or information indicating that the batch was successfully validated, if appropriate.

[0048] In step **315**, the processing module **107** determines whether the batch is acceptable, based on the validation performed in step **310**. In certain exemplary embodiments, the processing module **107** can make the decision of step **315** based on one or more rules stored in the database **109**. If the processing module **107** determines in step **315** that the batch is not acceptable, then the method **230** branches to step **330**, which is discussed below.

[0049] If the processing module **107** determines in step **315** that the batch is acceptable, then the method **230** continues to step **320**. In step **320**, the processing module **107** validates ACH items in the batch and stores validation results in the database **109**. For example, the processing module **107** can validate each ACH item by determining whether the ACH item is properly formatted and/or includes suitable information for processing. In certain exemplary embodiments, the processing module **107** can perform this validation based on one or more rules stored in the database **109**. For example, the stored validation results can include information regarding any validation errors identified during validation and/or information identifying each successfully validated ACH item.

[0050] In step **325**, the processing module **107** creates one or more output batches. Each output batch is an electronic file or record that includes information regarding at least one ACH item of the partition. For example, all the ACH items in a particular output batch can be associated with the same RDFI **115** and/or ODFI **102**.

[0051] In certain exemplary embodiments, the processing module **107** can create "risk pended" and "non-risk pended" output batches. A risk pended output batch is an electronic file or record that includes information regarding at least one ACH item that was not successfully validated. Similarly, a non-risk pended output batch is an electronic file or record that includes information regarding at least one ACH item that has been successfully validated. In certain exemplary embodiments, a "flag" or indicator within, or associated with, a particular output batch can indicate whether the output batch is a risk pended output batch or a non-risk pended output batch.

[0052] In step **330**, the processing module **107** stores the output batches in the database **109**. In step **335**, the processing module **107** determines whether to select another batch. If so, then the method **230** branches back to step **305** to repeat steps **305-330** for another batch. If the processing module **107** determines in step **335** to not select another batch, then the method **230** continues to step **340**.

[0053] In step **340**, the processing module **107** stores settlement information for each of the output batches in the database **109**. For example, the settlement information may include information useful in settling payment of each ACH item included in the output batches, such as an aggregate credit or debit amount for each ODFI **102** and/or RDFI **115**, a credit or debit amount for each ACH item, an American Bankers Association ("ABA") number associated with each ODFI **102**, and/or an ABA number associated with each RDFI **115**. The method **230** continues to step **235** of FIG. **2**, discussed above.

[0054] FIG. **4** is a flow chart depicting a method **270** for processing risk pended output batches, in accordance with certain exemplary embodiments, as referred to in step **270** of FIG. **2**. The exemplary method **270** is illustrative and, in alternative embodiments of the invention, certain steps can be performed in a different order, in parallel with one another, or omitted entirely, and/or certain additional steps can be performed without departing from the scope and spirit of the invention. The method **270** is described below with reference to FIGS. **1-4**.

[0055] In step **405**, the processing module identifies at least one risk pended output batch. In step **410**, the processing module determines whether each identified risk pended output batch risk is acceptable for transmission to an RDFI **115**. In certain exemplary embodiments, the processing module **107** can make that determination based on information from one or more operators of the ACH processing system **103** and/or the ODFI **102**. For example, the processing module **107** can transmit at least a portion of each risk pended output batch to its corresponding ODFI **102** for review, via the network **104**. The operator(s) of the ODFI **102** and/or ACH processing system **103** can evaluate whether each risk pended output batch is suitable for transmission to the RDFI **115** based on one or more rules. For example, the rules may include certain formatting and/or content requirements for each batch and/or ACH item therein. The rules may be the same or different from rules used in validating the ACH file, partitions, and ACH items in steps **210** and **230** of FIG. **2**. In certain exemplary embodiments, the rules can be stored in a database, such as the database **109**.

[0056] In step **415**, the processing module **107** rejects any risk pended ACH items determined in step **410** to not be acceptable for transmission. For example, the processing module **107** can return each rejected output and/or the ACH items contained therein to its corresponding ODFI **102**, suspend processing of the output batch and/or ACH items, and/or output a notification advising the ODFI **102** and/or an operator of the ACH processing system **103** that the output batch and/or ACH items will not be processed.

[0057] In step **420**, the processing module **107** settles each ACH item in each risk pended output batch, if any, determined in step **410** to be acceptable for transmission. For example, the processing module **107** can cause accounts associated with the ODFI **102** and RDFI **115** of each ACH item to be credited and/or debited in accordance with information in the ACH item. Similar to step **255** of FIG. **2**, in step **425**, the processing module **107** creates at least one new ACH file for each RDFI **115** associated with one or more of the ACH items settled in step **420**. Each new ACH file for a particular RDFI **115** includes one or more batches of acceptable ACH items associated with the RDFI **115**. In step **430**, the processing module **107** transmits each new ACH file to its corresponding RDFI **115**. For example, the processing module **107** can transmit the new ACH file(s) via the network **104**.

[0058] It will be appreciated that the exemplary embodiments of the invention overcome the limitations of the prior art. From the description of the exemplary embodiments, equivalents of the elements shown therein and ways of constructing other embodiments of the invention will be apparent to practitioners of the art. Many other modifications, features and embodiments of the invention will become evident to those of skill in the art. It should be appreciated, therefore, that many aspects of the invention were described above by way of example only and are not intended as required or essential elements of the invention unless explicitly stated otherwise. Accordingly, it should be understood that the foregoing relates only to certain embodiments of the invention and that numerous changes can be made therein without departing from the spirit and scope of the invention.

We claim:

1. A computer-implemented method for processing automated clearinghouse ("ACH") payments, comprising the steps of:

receiving an ACH file comprising a plurality of batches of ACH items, each of the batches comprising at least one of the ACH items;

organizing data in the ACH file into a plurality of partitions, each partition comprising at least one of the batches; and

separately processing each partition in parallel to generate at least one new batch.

2. The computer-implemented method of claim **1**, wherein the step of organizing data in the ACH file comprises the steps of:

selecting a strategy for organizing data in the ACH file; and

organizing data in the ACH file into the plurality of partitions according to the selected strategy.

3. The computer-implemented method of claim **1**, wherein the step of organizing data in the ACH file comprises the step of assigning a fixed number of the batches to each of the partitions.

4. The computer-implemented method of claim **1**, wherein the step of organizing data in the ACH file comprises the step of assigning a substantially equal number of batches to each of the partitions.

5. The computer-implemented method of claim **1**, wherein the step of organizing data in the ACH file comprises the step of assigning a substantially equal number of ACH items to each of the partitions.

6. The computer-implemented method of claim **1**, wherein the step of separately processing each partition in parallel comprises the step of storing at least one new batch for each partition, each new batch comprising information regarding at least one ACH item from the new batch's partition.

7. The computer-implemented method of claim **1**, wherein the step of separately processing each partition in parallel comprises the step of storing settlement information for each partition.

8. The computer-implemented method of claim **1**, wherein the step of separately processing each partition in parallel comprises the steps of:

for each partition,

determining whether each ACH item of the partition is suitable for processing; and

creating at least one new batch in response to determining that any ACH item of the partition is suitable for processing, each new batch comprising at least one ACH item.

9. The computer-implemented method of claim **8**, further comprising the steps of:

determining whether the ACH file is suitable for processing based on whether the ACH items of the partitions were determined to be suitable for processing; and

creating at least one new ACH file in response to determining that the ACH file is suitable for processing, each new ACH file comprising information regarding at least one of the at least one new batch.

10. The computer-implemented method of claim **9**, further comprising the step of transmitting each new ACH file to a corresponding receiving depository financial institution.

11. A computer-implemented method for processing automated clearinghouse ("ACH") payments, comprising the steps of:

receiving an ACH file comprising a plurality of batches of ACH items, each of the batches comprising at least one of the ACH items;

selecting a strategy for organizing data in the ACH file into a plurality of partitions, each partition comprising at least one of the batches, the strategy comprising one of assigning a fixed number of the batches to each of the partitions, assigning a substantially equal number of batches to each of the partitions, and assigning a substantially equal number of ACH items to each of the partitions;

organizing data in the ACH file into the plurality of partitions according to the selected strategy; and

separately processing each partition in parallel to generate at least one new batch.

12. The computer-implemented method of claim **11**, wherein the step of separately processing each partition in parallel comprises the step of storing at least one new batch for each partition, each new batch comprising information regarding at least one ACH item from the new batch's partition.

13. The computer-implemented method of claim **11**, wherein the step of separately processing each partition in parallel comprises the step of storing settlement information for each partition.

14. The computer-implemented method of claim **11**, wherein the step of separately processing each partition in parallel comprises the steps of:

for each partition,

determining whether each ACH item of the partition is suitable for processing; and

creating at least new batch in response to determining that any ACH item of the partition is suitable for processing, each new batch comprising at least one ACH item.

15. The computer-implemented method of claim **14**, further comprising the steps of:

determining whether the ACH file is suitable for processing based on whether the ACH items of the partitions were determined to be suitable for processing; and

creating at least one new ACH file in response to determining that the ACH file is suitable for processing, each new ACH file comprising information regarding at least one of the at least one new batch.

16. The computer-implemented method of claim **15**, further comprising the step of transmitting each new ACH file to a corresponding receiving depository financial institution.

17. A computer-implemented method for processing automated clearinghouse ("ACH") payments, comprising the steps of:

receiving an ACH file comprising a plurality of batches of ACH items, each of the batches comprising at least one of the ACH items;

organizing data in the ACH file into a plurality of partitions, each partition comprising at least one of the batches;

separately processing each partition in parallel by storing at least one new batch for each partition, each new batch comprising information regarding at least one successfully validated ACH item from the new batch's partition; and

determining whether the ACH file is suitable for processing based on validation information associated with each partition; and

creating at least one new ACH file in response to determining that the ACH file is suitable for processing, each new ACH file comprising information regarding at least one of the at least one new batches.

18. The computer-implemented method of claim **17**, wherein the step of organizing data in the ACH file comprises the steps of:

selecting a strategy for organizing data in the ACH file into the plurality of partitions, each partition comprising at least one of the batches, the strategy comprising one of assigning a fixed number of the batches to each of the partitions, assigning a substantially equal number of batches to each of the partitions, and assigning a substantially equal number of ACH items to each of the partitions; and

organizing data in the ACH file into the plurality of partitions according to the selected strategy.

19. The computer-implemented method of claim **17**, wherein the step of separately processing each partition in parallel further comprises the step of storing settlement information for each partition, the stored settlement information for each of the partitions comprising information regarding at least one successfully validated ACH item from the partition.

20. The computer-implemented method of claim **17**, further comprising the step of transmitting each new ACH file to a corresponding receiving depository financial institution.

* * * * *