July 2020

System And Method For Validating A Multi-Digit Number

KIM WAGNER
Visa

CHACKAN LAI
Visa

SIMON HURRY
Visa

TITLE: “System And Method For Validating A Multi-Digit Number”

VISA

KIM WAGNER

CHACKAN LAI

SIMON HURRY

TECHNICAL FIELD

The present subject matter relates to field of data processing, more particularly, but not exclusively to a system and method for validating a multi-digit number.

BACKGROUND

For a financial transaction, it is required to validate an account number used for the financial transaction. The identification number may be Primary Account Number (PAN) associated with a payment card for example a credit card or a debit card used for the financial transaction. PAN is a multi-digit number and is standardized to be 16 or 19 digits consisting of the Bank Identification Number (BIN) and the account number.

Conventional systems disclose to validate such multi-digit numbers with Luhn check. The Luhn check uses an algorithm, also known as modulus 10 or mod 10 algorithm. The algorithm may be used to validate a variety of identification numbers, such as credit card numbers, debit card numbers and so on. Using the Luhn check, token PANs are introduced for validating the PANs. In general, the PANs are of really short supply. More importantly, the existing Luhn check limits the number of available PANs as the last digit may only contain 1 digit out of a possible 10 digits. Also, increasing the length of the PAN may majorly impact payment network and payment infrastructure.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this disclosure, illustrate exemplary embodiments and, together with the description, serve to explain the disclosed principles. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The same numbers are used throughout the figures to reference like features and components. Some embodiments of device or system and/or methods in accordance with embodiments of the present subject matter are now described, by way of example only, and with reference to the accompanying figures, in which:

Figure 1 illustrates an exemplary environment of a validation system for validating a multi-digit number in accordance with some embodiments of the present disclosure;

Figure 2a illustrates a flow diagram showing a method for determining a check digit for a multi-digit number in accordance with some embodiments of the present disclosure;

Figure 2b illustrates a flow diagram showing a method for validating a multi-digit number using a check digit in accordance with some embodiments of the present disclosure; and

Figures 3 illustrates a block diagram of an exemplary computer system for implementing embodiments consistent with the present disclosure.

The figures depict embodiments of the disclosure for purposes of illustration only. One skilled in the art will readily recognize from the following description that alternative embodiments of the structures and methods illustrated herein may be employed without departing from the principles of the disclosure described herein.

DESCRIPTION OF THE DISCLOSURE

In the present document, the word "exemplary" is used herein to mean "serving as an example, instance, or illustration." Any embodiment or implementation of the present subject matter described herein as "exemplary" is not necessarily to be construed as preferred or advantageous over other embodiments.

While the disclosure is susceptible to various modifications and alternative forms, specific embodiment thereof has been shown by way of example in the drawings and will be described in detail below. It should be understood, however that it is not intended to limit the disclosure to the particular forms disclosed, but on the contrary, the disclosure is to cover all modifications, equivalents, and alternative falling within the spirit and the scope of the disclosure.

The terms “comprises”, “comprising”, or any other variations thereof, are intended to cover a non-exclusive inclusion, such that a setup, device or method that comprises a list of components or steps does not include only those components or steps but may include other components or steps not expressly listed or inherent to such setup or device or method. In other words, one or more elements in a device or system or apparatus proceeded by “comprises… a” does not, without more constraints, preclude the existence of other elements or additional elements in the device or system or apparatus.

The terms "an embodiment", "embodiment", "embodiments", "the embodiment", "the embodiments", "one or more embodiments", "some embodiments", and "one embodiment"

mean "one or more (but not all) embodiments of the invention(s)" unless expressly specified otherwise.

The terms "including", "comprising", “having” and variations thereof mean "including but not limited to", unless expressly specified otherwise.

The present disclosure proposes a modified type of MOD 10 algorithm for validating a multi-digit number. Such MOD 10 algorithm may also be referred to as modified Luhn check or a Wagner check. The proposed modified Luhn check extends range of the Luhn check from 1 to 10 while preserving integrity of the multi-digit number when the multi-digit number is manually captured or typed.

Figure 1 illustrates an exemplary environment 100 of a validation system 101 which is configured to validate a multi-digit number. The multi-digit number may be a PAN of a payment card, for example a credit card, EMI card, Gift card, prepaid card, or a debit card used for a financial transaction. The multi-digit number is validated to check the validity of corresponding payment card. Every newly issued payment card or token PAN from all payment brands is associated with a multi-digit number. The environment 100 for validating the multi-digit number includes a multi-digit number retrieving unit 103, a communication network 104 and the validation system 101. Consider a payment card 102 with PAN including predefined multi-digit number. Standard format of the multi-digit number is of 16 digits. In an embodiment, number of digits in the PAN may be defined by payment systems. The PAN may include BIN followed by the account number. The PAN, hereafter referred to as the multi-digit number, may be predefined using one or more methods known to a person skilled in the art. In an embodiment, least significant number of the multi-digit number is determined using the proposed algorithm and is further used for the validation. In an embodiment, the least significant number may also be referred to as check digit or Wagner check digit. In an embodiment, along with the validation, the validation system 101 may be configured to determine the check digit for a PAN. Using the proposed method, number of accounts per BIN may be increased by a factor of 10. More importantly the proposed method provides a further function of strong typing of the PAN into up to 10 different categories. In an embodiment, the categories may include, but limited to, standard financial PAN, Card Present (CP) token card form factor, CP token mobile form factor, CP token wearable, Card Not Present (CNP) token, CNP token source, CNP token merchant proprietary and so on.

The proposed method discloses 9 distinct ways to modify calculation of the check digit. Each calculation is distinguished by how much to add i.e., modulo 10 to the check digit. The added number may be referred to as “n” or a predefined number. The values of “n” ranges from 1 to 9.

For example, consider a PAN is to be defined for the payment card 102 for a CP token card form factor. Consider “n” is equal to “1” for category CP token card form factor. Let the PAN be as shown below

4717 1799 9000 007X

In an embodiment, first four digits of the PAN may define the BIN and next eleven digits of the PAN may define the account number. In another embodiment, first six digits of the PAN may define the BIN and next nine digits of the PAN may define the account number. The least significant digit i.e., the sixteenth digit is the check digit, for example “X” in the above-mentioned PAN. For determining the check digit, the product of every second digit, starting from rightmost digit and number “2” is calculated as shown below table:

4	7	1	7	1	7	9	9	9	0	0	0	0	0	7	X
*2		*2		*2		*2		*2		*2		*2		*2	
8	7	2	7	2	7	18	9	18	0	0	0	0	0	14	

Table 1

In case any product is identified to be greater than “9”, digits of such product are summed to obtain a number lesser than “9” and replaced with the product. In the above table, product of fifteenth digit with “2” is “14” which is greater than “9”. Hence “1” and “4” are summed to obtain “5” which then replaces “14”. Likewise, 18 in the above table is replaced with 9. Below table illustrates the calculation of product of digits with “2”.

4	7	1	7	1	7	9	9	9	0	0	0	0	0	7	X
*2		*2		*2		*2		*2		*2		*2		*2	
8	7	2	7	2	7	9	9	9	0	0	0	0	0	5	

Table 2

Further, sum of the products is determined. From the above table, the sum of the products (8+7+2+7+2+7+9+9+9+0+0+0+0+0+5) is calculated to be “65’. Further, the least significant

digit of the sum is considered i.e., “5”. For the selected category, i.e., CP token card form factor, “n” is equal to “1”. Hence the “5” is added to “1” to obtain “6”, which is the check digit for the PAN.

The determined check digit may be included as the sixteenth digit in the multi-digit number and defined to a PAN of a payment card. Hence, for the given example, the PAN may be as given below:

4717 1799 9000 0076

Using similar method of calculating check digit, the check digit may be calculated by varying value of “n”. By proposed method of calculating check digit, number of accounts associated with a BIN increases by factor of 10.

In real-time, for performing a financial transaction, validation of the PAN is to be performed. For the validation, the exemplary environment 100 in **Figure 1** may provision the validation of the PAN. The multi-digit number retrieving unit 103 may be configured to retrieve the multi-digit number from the payment card 102 used for the financial transaction. In an embodiment, the multi-digit number retrieving unit 103 may be configured to image the payment card 102 to extract the multi-digit number. One or more techniques, known to a person skilled in the art, may be implemented to retrieve the multi-digit number associated with the card. The multi-digit number may be provided to the validation system 101 via the communication network 104, for the validation. The communication network 104 may include, without limitation, a direct interconnection, Local Area Network (LAN), Wide Area Network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, and the like. In an embodiment, the validation system 101 may be implemented in a server configured to receive the multi-digit number and validate the multi-digit number. In an embodiment, such server may be a dedicated server or a cloud-based server.

For validating the multi-digit number, the validation system 101 may be configured to calculate a validating digit for the multi-digit number. Using the validation system 101, the validating digit may be calculated. The proposed method provisions nine distinct ways for calculating the validating digit. The distinct ways may be distinguished by how much to add i.e., modulo 10 to output of the algorithm. From previous example, consider the received multi-digit number is as given below:

4717 1799 9000 0076

For determining the validating digit, product of every second digit, starting from rightmost digit and number “2” is calculated as shown below table:

4	7	1	7	1	7	9	9	9	0	0	0	0	0	7	6
*2		*2		*2		*2		*2		*2		*2		*2	
8	7	2	7	2	7	18	9	18	0	0	0	0	0	14	

Table 3

In case any product is identified to be greater than “9”, digits of such product is summed to obtain a number lesser than “9” and replaced with the product. Below table illustrates the calculation of product of digits with “2”.

4	7	1	7	1	7	9	9	9	0	0	0	0	0	7	6
*2		*2		*2		*2		*2		*2		*2		*2	
8	7	2	7	2	7	9	9	9	0	0	0	0	0	5	

Table 4

Further, sum of the products is determined. From the above table, the sum of the products is calculated to be “65’. Further, least significant digit of the sum is considered i.e., “5”. Consider the selected category is CP token card form factor for which “n” is equal to “1”. Hence the “5” is added to “1” to obtain “6”, which is the validating digit for the PAN.

For the given example, the validating digit is equal to the check digit. Hence, the credit card may be identified to be valid. In an embodiment, upon identification of the payment card 102 to be valid, the financial transaction using the payment card 102 may be approved. In case the validating digit is not equal to the check digit, the credit may be identified to be invalid. In an embodiment, upon identification of the payment card 102 to be invalid, the financial transaction using the payment card 102 may be declined.

Figure 2a illustrates a flow diagram showing a method for determining the check digit for the multi-digit number in accordance with some embodiments of the present disclosure.

In an embodiment, a check digit for a PAN may be determined when defining the PAN to a payment card. By the proposed method, the number of PANs per BIN may be increased. This may be achieved by selecting different values of the predefined values or “n”. In an

embodiment, the validation system 101 may be configured to determine the check digit for the multi-digit number. In another embodiment, a server associated with the validation system 101 may be used to determine the check digit and generate plurality of PANs.

For determining the check digit, at block 201, the multi-digit number is received. The received multi-digit number may include the BIN and the account number.

At block 202, the number "2" is multiplied with every second digit from right most digit of the multi-digit number. By the multiplication, plurality of products are determined.

At block 203, each of the plurality of products are checked to be lesser than "9".

At block 204, if any of the products from the plurality of products is greater than "9", step in block 205 is performing for such products. If a product is lesser than "9", step in block 207 is performed.

At block 205, upon determining a product to be greater than "9", sum of the digits in the product is determined. For example, if the product is "12", the sum is "2" +"1" ="3".

At block 206, the calculated sum is replaced with the product.

At block 207, upon determining the product to be lesser than "9" and replacing the product with sum, sum of the plurality of products is determined.

At block 208, the predefined number is added to the least significant digit in the sum of the plurality of products. The predefined number may range from "1" to "9". The number obtained upon the addition is the check digit for the multi-digit number. The determined check digit may be included as the sixteenth digit in the multi-digit number and defined to be a PAN of a payment card.

Figure 2b illustrates a flow diagram showing a method for validating the multi-digit number using the check digit in accordance with some embodiments of the present disclosure. Validation of the multi-digit number may be performed in real-time. The validation system 101 may be configured to retrieve the multi-digit number. For performing the validation, the validation system 101 may be configured to determine a validation digit for the multi-digit number.

For determining the validating number, at block 209, the multi-digit number is received by the validation system 101. The received multi-digit number may include the BIN, the account number, and the check digit.

At block 210, the number “2” is multiplied with every second digit from right most digit of the multi-digit number. By the multiplication, plurality of products are determined.

At block 211, each of the plurality of products are checked to be lesser than “9”.

At block 212, if any of the products from the plurality of products is greater than “9”, step in block 213 is performing for such products. If a product is lesser than “9”, step in block 215 is performed.

At block 213, upon determining a product to be greater than “9”, sum of the digits in the product is determined.

At block 214, the calculated sum is replaced with the product.

At block 215, upon determining the product to be lesser than “9” and replacing the product with sum, sum of the plurality of products is determined.

At block 216, the predefined number is added to the least significant digit in the sum of the plurality of products. The predefined number may range from “1” to “9”. The number obtained upon the addition is the validation number for the multi-digit number.

At block 217, the validating number is checked to be equal to the check digit. If the validating number is not equal to the check digit, step in block 218 is performed. If the validating number is equal to the check digit, step in block 219 is performed.

At block 218, upon determining the validating number to be not equal to the check digit, the multi-digit number may be identified to be invalid.
At block 219, upon determining the validating number to be equal to the check digit, the multi-digit number may be identified to be invalid.

In an embodiment, based on the validation of the multi-digit number, a financial transaction associated with the multi-digit number may be approved or declined.

Advantages of the present disclosure

Embodiments of the present disclosure discloses a modified Luhn check by which provisions validate that a PAN was captured correctly. The problem of "fat fingering" when capturing large numbers of payment card numbers manually is eliminated.

Embodiments of the present disclosure checks on the problem of running short on PANs. By the proposed method, 9 times more PANs may be introduced without increasing the length of the PAN. This provisions major benefits to payment industries.

Computing System

Figure 3 illustrates a block diagram of an exemplary computer system 300 for implementing embodiments consistent with the present disclosure. In an embodiment, the computer system 300 is used to implement the validation system 101 for validating multi-digit number. The computer system 300 may include a central processing unit (“CPU” or “processor”) 302. The processor 302 may include at least one data processor for executing processes in Virtual Storage Area Network. The processor 302 may include specialized processing units such as, integrated system (bus) controllers, memory management control units, floating point units, graphics processing units, digital signal processing units, etc.

The processor 302 may be disposed in communication with one or more input/output (I/O) devices 309 and 310 via I/O interface 301. The I/O interface 301 may employ communication protocols/methods such as, without limitation, audio, analog, digital, monaural, RCA, stereo, IEEE-1394, serial bus, universal serial bus (USB), infrared, PS/2, BNC, coaxial, component, composite, digital visual interface (DVI), high-definition multimedia interface (HDMI), radio frequency (RF) antennas, S-Video, VGA, IEEE 802.n /b/g/n/x, Bluetooth, cellular (e.g., code-division multiple access (CDMA), high-speed packet access (HSPA+), global system for mobile communications (GSM), long-term evolution (LTE), WiMax, or the like), etc.

Using the I/O interface 301, the computer system 300 may communicate with one or more I/O devices 309 and 310. For example, the input devices 309 may be an antenna, keyboard, mouse, joystick, (infrared) remote control, camera, card reader, fax machine, dongle, biometric reader, microphone, touch screen, touchpad, trackball, stylus, scanner, storage device, transceiver, video device/source, etc. The output devices 310 may be a printer, fax machine, video display (e.g., cathode ray tube (CRT), liquid crystal display (LCD), light-emitting diode (LED),

plasma, Plasma Display Panel (PDP), Organic light-emitting diode display (OLED) or the like), audio speaker, etc.

In some embodiments, the computer system 300 may consist of the validation system 101. The processor 302 may be disposed in communication with a communication network 311 via a network interface 303. The network interface 303 may communicate with the communication network 311. The network interface 303 may employ connection protocols including, without limitation, direct connect, Ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, etc. The communication network 311 may include, without limitation, a direct interconnection, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, etc. Using the network interface 303 and the communication network 311, the computer system 300 may communicate with a multi-digit number retrieving unit 512 to retrieve a multi-digit number from a payment card for financial transaction. The network interface 303 may employ connection protocols include, but not limited to, direct connect, Ethernet (e.g., twisted pair 10/100/1000 Base T), transmission control protocol/internet protocol (TCP/IP), token ring, IEEE 802.11a/b/g/n/x, etc.

The communication network 311 includes, but is not limited to, a direct interconnection, an e-commerce network, a peer to peer (P2P) network, local area network (LAN), wide area network (WAN), wireless network (e.g., using Wireless Application Protocol), the Internet, Wi-Fi, and such. The first network and the second network may either be a dedicated network or a shared network, which represents an association of the different types of networks that use a variety of protocols, for example, Hypertext Transfer Protocol (HTTP), Transmission Control Protocol/Internet Protocol (TCP/IP), Wireless Application Protocol (WAP), etc., to communicate with each other. Further, the first network and the second network may include a variety of network devices, including routers, bridges, servers, computing devices, storage devices, etc.

In some embodiments, the processor 302 may be disposed in communication with a memory 305 (e.g., RAM, ROM, etc. not shown in **Figure 3**) via a storage interface 304. The storage interface 304 may connect to memory 305 including, without limitation, memory drives, removable disc drives, etc., employing connection protocols such as, serial advanced technology attachment (SATA), Integrated Drive Electronics (IDE), IEEE-1394, Universal

Serial Bus (USB), fibre channel, Small Computer Systems Interface (SCSI), etc. The memory drives may further include a drum, magnetic disc drive, magneto-optical drive, optical drive, Redundant Array of Independent Discs (RAID), solid-state memory devices, solid-state drives, etc.

The memory 305 may store a collection of program or database components, including, without limitation, user interface 306, an operating system 307, web browser 308 etc. In some embodiments, computer system 300 may store user/application data, such as, the data, variables, records, etc., as described in this disclosure. Such databases may be implemented as fault-tolerant, relational, scalable, secure databases such as Oracle ® or Sybase®.

The operating system 307 may facilitate resource management and operation of the computer system 300. Examples of operating systems include, without limitation, APPLE MACINTOSH® OS X, UNIX®, UNIX-like system distributions (E.G., BERKELEY SOFTWARE DISTRIBUTION™ (BSD), FREEBSD™, NETBSD™, OPENBSD™, etc.), LINUX DISTRIBUTIONS™ (E.G., RED HAT™, UBUNTU™, KUBUNTU™, etc.), IBM™ OS/2, MICROSOFT™ WINDOWS™ (XP™, VISTA™/7/8, 10 etc.), APPLE® IOS™, GOOGLE® ANDROID™, BLACKBERRY® OS, or the like.

In some embodiments, the computer system 300 may implement a web browser 308 stored program component. The web browser 308 may be a hypertext viewing application, such as Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, Apple Safari, etc. Secure web browsing may be provided using Hypertext Transport Protocol Secure (HTTPS), Secure Sockets Layer (SSL), Transport Layer Security (TLS), etc. Web browsers 308 may utilize facilities such as AJAX, DHTML, Adobe Flash, JavaScript, Java, Application Programming Interfaces (APIs), etc. In some embodiments, the computer system 300 may implement a mail server stored program component. The mail server may be an Internet mail server such as Microsoft Exchange, or the like. The mail server may utilize facilities such as ASP, ActiveX, ANSI C++/C#, Microsoft .NET, Common Gateway Interface (CGI) scripts, Java, JavaScript, PERL, PHP, Python, WebObjects, etc. The mail server may utilize communication protocols such as Internet Message Access Protocol (IMAP), Messaging Application Programming Interface (MAPI), Microsoft Exchange, Post Office Protocol (POP), Simple Mail Transfer Protocol (SMTP), or the like. In some embodiments, the computer system 300 may implement

a mail client stored program component. The mail client may be a mail viewing application, such as Apple Mail, Microsoft Entourage, Microsoft Outlook, Mozilla Thunderbird, etc.

Furthermore, one or more computer-readable storage media may be utilized in implementing embodiments consistent with the present disclosure. A computer-readable storage medium refers to any type of physical memory on which information or data readable by a processor may be stored. Thus, a computer-readable storage medium may store instructions for execution by one or more processors, including instructions for causing the processor(s) to perform steps or stages consistent with the embodiments described herein. The term "computer-readable medium" should be understood to include tangible items and exclude carrier waves and transient signals, i.e., be non-transitory. Examples include Random Access Memory (RAM), Read-Only Memory (ROM), volatile memory, non-volatile memory, hard drives, Compact Disc (CD) ROMs, DVDs, flash drives, disks, and any other known physical storage media. media.

The described operations may be implemented as a method, system or article of manufacture using standard programming and/or engineering techniques to produce software, firmware, hardware, or any combination thereof. The described operations may be implemented as code maintained in a "non-transitory computer readable medium", where a processor may read and execute the code from the computer readable medium. The processor is at least one of a microprocessor and a processor capable of processing and executing the queries. A non-transitory computer readable medium may include media such as magnetic storage medium (e.g., hard disk drives, floppy disks, tape, etc.), optical storage (CD-ROMs, DVDs, optical disks, etc.), volatile and non-volatile memory devices (e.g., EEPROMs, ROMs, PROMs, RAMs, DRAMs, SRAMs, Flash Memory, firmware, programmable logic, etc.), etc. Further, non-transitory computer-readable media may include all computer-readable media except for a transitory. The code implementing the described operations may further be implemented in hardware logic (e.g., an integrated circuit chip, Programmable Gate Array (PGA), Application Specific Integrated Circuit (ASIC), etc.).

The illustrated steps are set out to explain the exemplary embodiments shown, and it should be anticipated that ongoing technological development will change the manner in which particular functions are performed. These examples are presented herein for purposes of illustration, and not limitation. Further, the boundaries of the functional building blocks have been arbitrarily

defined herein for the convenience of the description. Alternative boundaries can be defined so long as the specified functions and relationships thereof are appropriately performed. Alternatives (including equivalents, extensions, variations, deviations, etc., of those described herein) will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein. Such alternatives fall within the scope and spirit of the disclosed embodiments. Also, the words "comprising," "having," "containing," and "including," and other similar forms are intended to be equivalent in meaning and be open ended in that an item or items following any one of these words is not meant to be an exhaustive listing of such item or items, or meant to be limited to only the listed item or items. It must also be noted that as used herein and in the appended claims, the singular forms “a,” “an,” and “the” include plural references unless the context clearly dictates otherwise.

Furthermore, one or more computer-readable storage media may be utilized in implementing embodiments consistent with the present disclosure. A computer readable storage medium refers to any type of physical memory on which information or data readable by a processor may be stored. Thus, a computer readable storage medium may store instructions for execution by one or more processors, including instructions for causing the processor(s) to perform steps or stages consistent with the embodiments described herein. The term “computer readable medium” should be understood to include tangible items and exclude carrier waves and transient signals, i.e., are non-transitory. Examples include random access memory (RAM), read-only memory (ROM), volatile memory, non-volatile memory, hard drives, CD ROMs, DVDs, flash drives, disks, and any other known physical storage media.

Finally, the language used in the specification has been principally selected for readability and instructional purposes, and it may not have been selected to delineate or circumscribe the inventive subject matter. Accordingly, the disclosure of the embodiments of the disclosure is intended to be illustrative, but not limiting, of the scope of the disclosure.

With respect to the use of substantially any plural and/or singular terms herein, those having skill in the art can translate from the plural to the singular and/or from the singular to the plural as is appropriate to the context and/or application. The various singular/plural permutations may be expressly set forth herein for sake of clarity.

SYSTEM AND METHOD FOR VALIDATING A MULTI-DIGIT NUMBER

ABSTRACT

The present disclosure provides a modified type of MOD 10 algorithm for validating a multi-digit number. Such MOD 10 algorithm may also be referred to as modified Luhn check or modifies Wagner check. The proposed modified Luhn check extends range of the Luhn check from 1 to 10 while preserving integrity of the multi-digit number when the multi-digit number is manually captured or typed. Also, by the proposed method, 9 times more PANs may be introduced without increasing the length of the PAN.

1/4

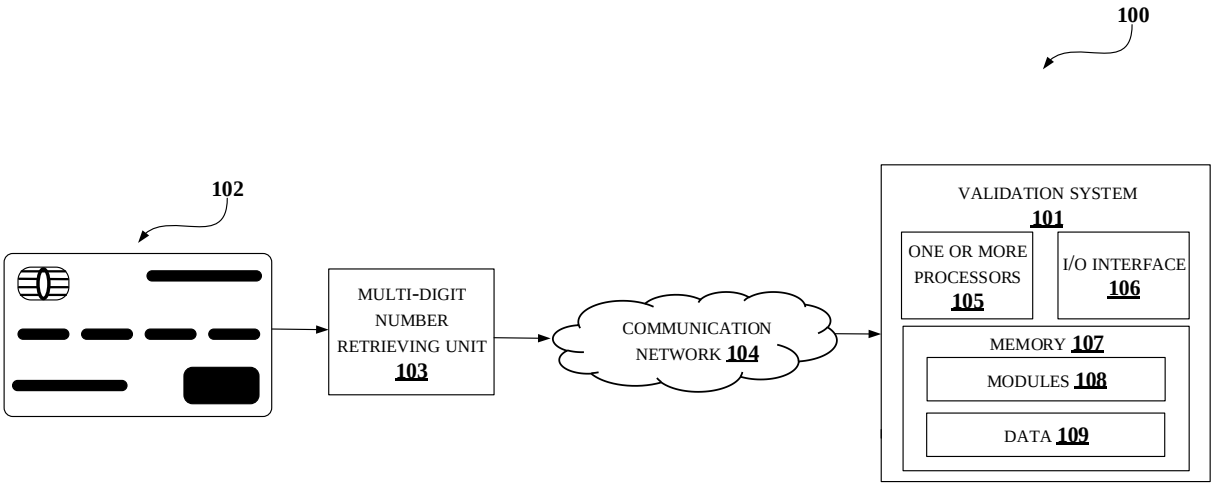
100
102
MULTI-DIGIT NUMBER RETRIEVING UNIT 103
COMMUNICATION NETWORK 104
VALIDATION SYSTEM 101
ONE OR MORE PROCESSORS 105
I/O INTERFACE 106
MEMORY 107
MODULES 108
DATA 109

Figure 1

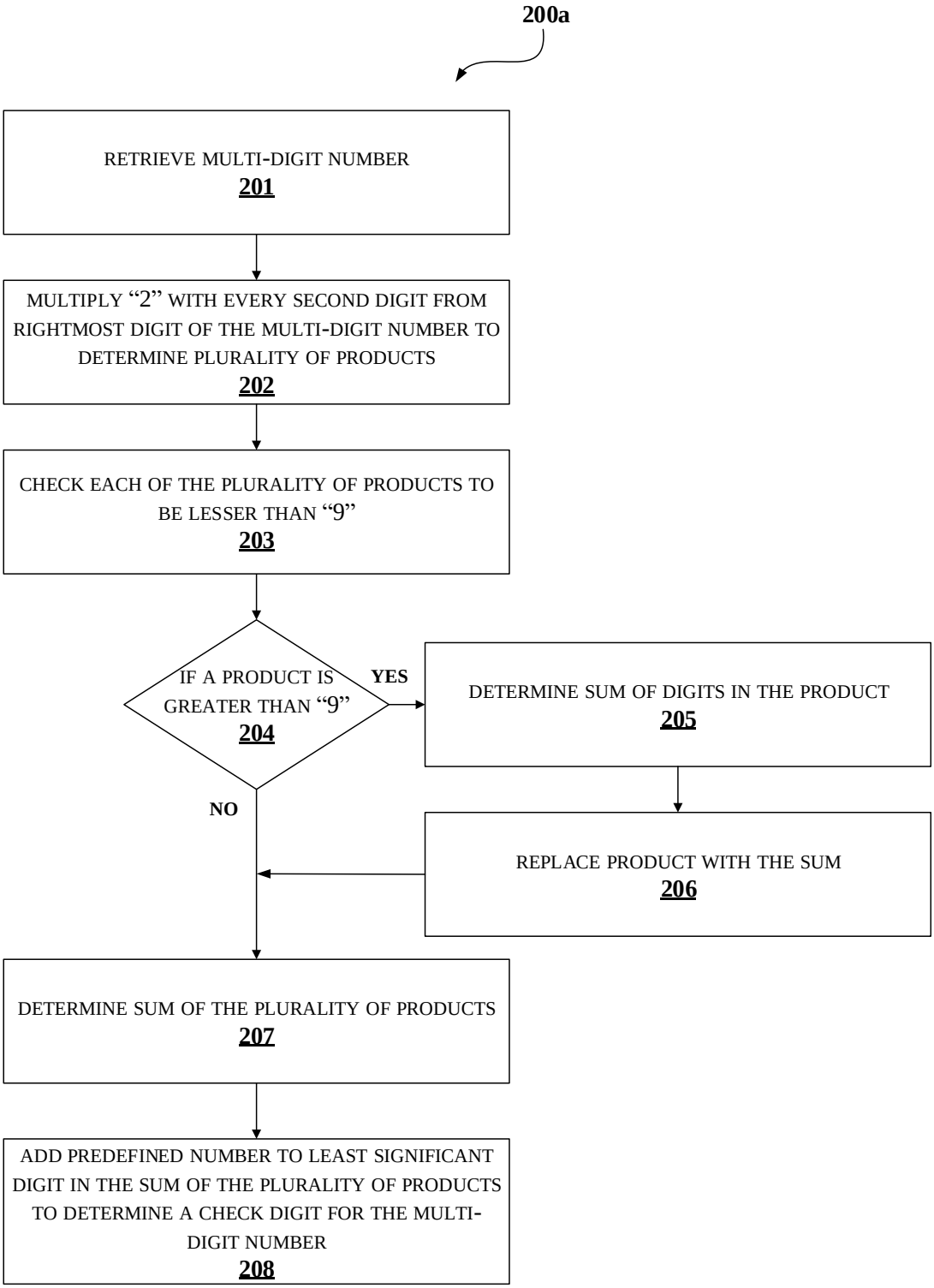
200a
RETRIEVE MULTI-DIGIT NUMBER
201
MULTIPLY "2" WITH EVERY SECOND DIGIT FROM RIGHTMOST DIGIT OF THE MULTI-DIGIT NUMBER TO DETERMINE PLURALITY OF PRODUCTS
202
CHECK EACH OF THE PLURALITY OF PRODUCTS TO BE LESSER THAN "9"
203
IF A PRODUCT IS GREATER THAN "9"
204
YES
DETERMINE SUM OF DIGITS IN THE PRODUCT
205
NO
REPLACE PRODUCT WITH THE SUM
206
DETERMINE SUM OF THE PLURALITY OF PRODUCTS
207
ADD PREDEFINED NUMBER TO LEAST SIGNIFICANT DIGIT IN THE SUM OF THE PLURALITY OF PRODUCTS TO DETERMINE A CHECK DIGIT FOR THE MULTI-DIGIT NUMBER
208

Figure 2a

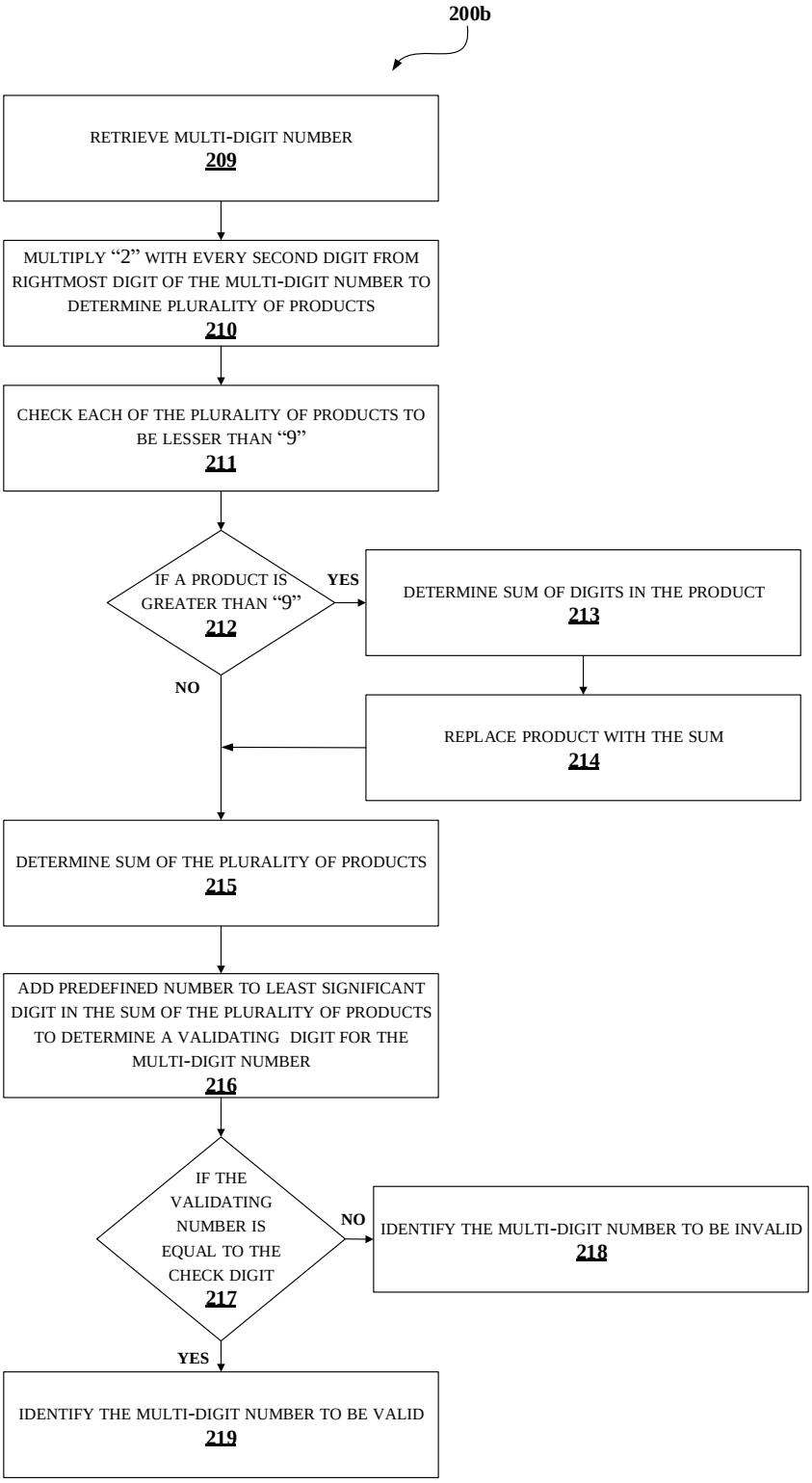
200b
RETRIEVE MULTI-DIGIT NUMBER
209
MULTIPLY "2" WITH EVERY SECOND DIGIT FROM RIGHTMOST DIGIT OF THE MULTI-DIGIT NUMBER TO DETERMINE PLURALITY OF PRODUCTS
210
CHECK EACH OF THE PLURALITY OF PRODUCTS TO BE LESSER THAN "9"
211
IF A PRODUCT IS GREATER THAN "9"
212
YES
DETERMINE SUM OF DIGITS IN THE PRODUCT
213
NO
REPLACE PRODUCT WITH THE SUM
214
DETERMINE SUM OF THE PLURALITY OF PRODUCTS
215
ADD PREDEFINED NUMBER TO LEAST SIGNIFICANT DIGIT IN THE SUM OF THE PLURALITY OF PRODUCTS TO DETERMINE A VALIDATING DIGIT FOR THE MULTI-DIGIT NUMBER
216
IF THE VALIDATING NUMBER IS EQUAL TO THE CHECK DIGIT
217
NO
IDENTIFY THE MULTI-DIGIT NUMBER TO BE INVALID
218
YES
IDENTIFY THE MULTI-DIGIT NUMBER TO BE VALID
219

Figure 2b

4/4

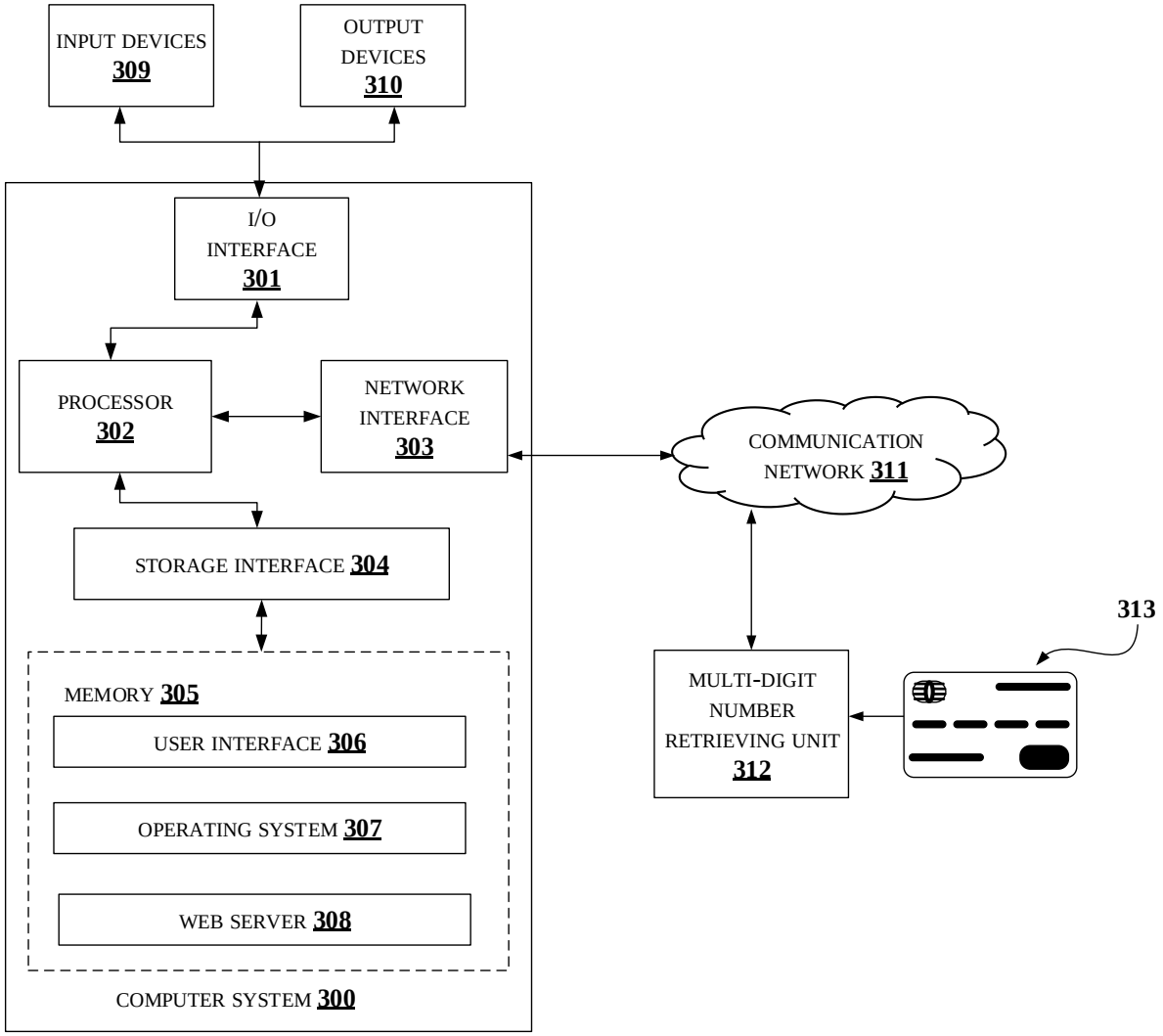
INPUT DEVICES 309
OUTPUT DEVICES 310
I/O INTERFACE 301
PROCESSOR 302
NETWORK INTERFACE 303
COMMUNICATION NETWORK 311
STORAGE INTERFACE 304
313
MEMORY 305
USER INTERFACE 306
OPERATING SYSTEM 307
WEB SERVER 308
MULTI-DIGIT NUMBER RETRIEVING UNIT 312
COMPUTER SYSTEM 300

Figure 3