Algorithm for Payment with POS Terminals in Real Time

T.Pavlov, J.Ostrev, A.Marinchev, V.Galabov

***Abstract*— In recent years, real-time payments with POS terminals have increased and their application is ubiquitous (for example at 2022 they are about 77.4 million per hour). Paying with POS terminals is also a convenience, which is a preferred way of transferring amounts in electronic form. Bank transaction information is related to the transfer amount, bank card type, card issuer, card type, transaction type and many other parameters. POS terminals are often operated in more than one or two shifts per day. Existing algorithms for POS terminals offer functionality, only for the most important minimum amount of data for preparing accounting documents. Very often POS terminals are used with cash registers (Cash Register). In these cases, automated data processing usually exists and data reporting is facilitated, and when not, addition is relatively easier compared to adding the functionality directly to the POS terminal software. The proposed algorithm does not require the presence of a cash register and provides the opportunity to receive detailed reporting in a pre-selected period of time by the user.**

***Keywords*—payments, POS terminals**

I. Introduction

In this paper, we present an algorithm designed to process data from POS terminal devices, providing comprehensive accounting analysis without the need for additional hardware such as cash registers. The proposed algorithm aims to reduce costs, increase flexibility, improve reporting capabilities and security, offer better customer service and improve overall competitiveness for users who have POS terminals but no cash registers.

By eliminating the need to purchase and maintain a separate cash register, users can save money, adapt their POS terminal settings more efficiently, and receive detailed real-time reports.

The algorithm allows for easy development and processing of both structured and unstructured data, incorporating advanced machine learning techniques and data analysis to identify patterns, anomalies and potential fraud.

Users can easily generate custom reports and export them in various formats.

The algorithm also allows the integration of robust security measures, such as data encryption, two-factor authentication, and regular system auditing. With the ability to scale and continuous maintenance and update, the proposed algorithm provides users who have POS terminals but no cash registers a more cost-effective, flexible and efficient way to manage their financial transactions and perform accounting analysis.

The data that is processed during a transaction represents many parameters affecting the cardholder and the banks between which the transfer takes place, but the focus is on the pro-cessing of the given transaction and ensuring security, which includes not only the transfer of data, but and additional checks for the cardholder, PIN code verifications, using addition-al parameters for information carriers (chip or magnetic stripe), etc. [1], [2], [3], [4], [5]. The most typical parameters are the type of transaction, the amount of the transaction, the time it takes place, parameters for the banks (only one bank is also possible) participating in the transfer [5].

II. Information that POS terminals work with

Most often, transactions at POS terminals are purchases, but in reality, the number of possible transactions is several dozen, as the construction and operation of all of them is subject to numerous standards and specifications [1].

EMV (Europay MasterCard Visa) is a global electronic transaction standard named after the three organizations that established it. The new EMV standard enables EFTPOS terminals worldwide to process chip-based debit and credit cards [1], [2], [3], [4].

A. Transactions

All transactions affect the development of the process started by the cardholder from the moment the amount is entered and confirmed through the POS terminal. All possible transactions when paying with POS terminals in real time contain a similar type of information [2], but what differentiates them significantly is the type of transaction:

- Purchase
- Refund
- Tip
- Reversal (if no response is received in the card system)
- and many others

B. Bank cards

Bank card data is a subset of all information about a given transaction. Although seemingly the same, bank cards differ from each other. They have a chip and a magnetic stripe to work with devices performing bank transactions. The most common are Visa, Master Card, Diners Euro Card, etc. Another characteristic parameter is the type of bank card -

Received: 06.08.2024
Published: 08.09.2024
https://doi.org/10.47978/TUS.2024.74.02.007

Toshko Pavlov is with the Technical University of Sofia, 1000 Sofia, Bulgaria (e-mail: t.pavlov@tu-sofia.bg)

Alexandar Marinchev is with the Technical University of Sofia, 1000 Sofia, Bulgaria (e-mail: amar@tu-sofia.bg)

Vasil Galabov is with the Technical University of Sofia, 1000 Sofia, Bulgaria (e-mail: vtg@tu-sofia.bg)

debit or credit.

Important, basic and typical other parameters are:

- transaction type
- the time of the transaction (year, month, day, hour, minute, second)
- the issuing bank of the card
- the receiving bank
- other parameters

C. Other data

POS terminals are often used in commercial establishments, where work is done in two or even three shifts. Algorithms for terminals produced by different companies are relatively similar in terms of scenarios and in the main scenario the focus is on transaction and communication so that a POS terminal creates and executes transactions and meets standards and requirements, not on processing already implemented transactions, which is often necessary for the user of the device. On the other hand, the algorithms of two different POS terminals of different manufacturing companies or different banks with which they work (and this determines specific requirements for the algorithms) use similar ways of processing the data, but in reality they are different implementations because they are various APIs and libraries to work with the device. Whit other words the main algorithms offered for POS terminals are directly related to those offered by their manufacturer ('Ingenico', 'Gemalto', etc.) and they can be upgraded, but without violating the functionality defined by standards implemented.

III. General Description of the Algorithm

Most often, the different versions of software in POS terminals offer only basic functionalities related to transactions and the need for additional information about the customer, when it needs to be carried out, can be usually implemented using the Cash Register. This algorithm belongs to the group of so-called ‚Shift Report' algorithms, but it does not require a connection to a cash register to work. On the other hand, the presence or absence of the Cash Register does not disrupt the normal operation of the algorithm in the device, because it does not rely on data from other devices external to the terminal. The proposed algorithm affects only a part of the data that the POS terminals work with only for the purpose of reporting information and statistics in any upcoming and pre-selected time interval by the user. It is added to existing software for working POS terminals, can be upgraded and is oriented to the needs of the user of the device, only for information about already completed transactions, as well as the type of presentation. The scenario of the algorithm and its subordination is presented on (Figure 1).

Most often, the main algorithms (Figure 1) implementing the transactions (T1,T2,T3 ...) basically do not care about what the proposed algorithm processes, because they are oriented to the banks and not to the client of the POS terminal. More precisely, these basic algorithms also work with the data of the proposed algorithm, but usually do not keep statistics about them. During each individual transaction (Figure 1), the algorithm reaches and retrieves data about the card issuer (Visa, MasterCard and so on), the type of card (debit, credit ...), the type of transaction (purchase, refund, tip ...), performs the necessary processing for summation and sorting and after completion of the current transaction stores the data in an XML file that is accessible at any time (from the main menu of the POS terminal and stored in it) for reading/writing.

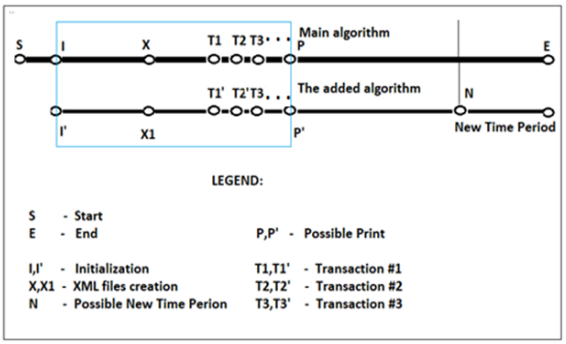

Fig. 1. The scenario of the algorithm and its subordination.

In Figure 1 the duplicate steps in the basic and the added algorithm represent only one example possible scenario of operation of a POS terminal, when a time period is set only once by the user for data processing. There is no restriction on resetting a new period, setting new periods - regardless of whether the already set period has started, or it has started, but the data has not been used yet, etc., with other words setting a new time period has the highest priority. In this way, the last created (if any) time period is worked with. If the time period is not set or the time period has expired, the algorithm does not process data.

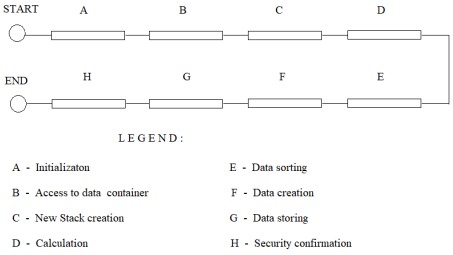

Fig. 2. Sequence diagram.

All actions affecting the algorithm - parallel processing of data in the main algorithm of the POS terminal, presetting a period of time in which it will work, parameters for issuer and card type, as well as any transaction performed, do not in any way violate the requirements and recommendations of EMV, on which the main algorithm works (as well as its autonomy) of a given POS terminal, regardless of whether it is mobile, stationary or part of some system. Operations that implement the algorithm for each transaction (for example, transaction T1 of Figure 1) are:

- Access to data containers of the main algorithm
- Search and find parameters (time, type and type of card, amounts, type of transaction, etc.)
- Summing and sorting
- Backing up data results
- Parallel and independent calculations
- Printing of current results at any time - after the expiration of the time of the set period or during the set period of time in which it works.
- Set or change for a new time period for work.

In essence, the algorithm continuously collects data during the communications of the POS terminal, processes them and stores the current state in an XML file, and in this practical implementation the parameters - start and end of the working time, amounts by type of transaction, time in which it is completed transaction, amounts by card issuer, amounts by card type and total amounts, amounts by issuer and card type and accounting documents by the user of the POS terminal.

A sequence diagram for an authorized transaction type (for example T1) is presented in Figure 2.

The algorithm does not necessarily require a set period to work, but only if there is a set period, it starts automatically, and a constant check function has been added to the main algorithm - presence or absence of a set period for 'Shift Report'.

In every POS terminal there is software that meets the requirements and standards of working with bank cards, and it is usually built on C/C++ - the functionality that is realized through the algorithm is added to it. For reasons of portability, the current data (relating only to the algorithm) are stored and processed in an independent and previously created file for the purpose - for convenience in XML format. In case of force majeure (for example, with simultaneous overlap in the time of power outage and battery failure), the recovery process for the device does not reduce its quality level in any way (before the added algorithm). In these cases, (and for the example of a force majeure situation) when the power is restored, the saved information is used fully automatically to reinitialize and reach the state of the device to the current moment in time, and the recovery process ends only with the loss of missing potential transactions in the crash time period, but this is due to the impossibility of their having been carried out. In fact, for the complete processing of the data, data is written in a separate new XML file for each transaction, which is only a choice of operation of the proposed algorithm, and not compliance with a restrictive requirement. In this way, the main algorithm of the POS terminal when adding this algorithm retains its autonomy, works with its own (usually XML files) and does not require reworking.

A. Implementation of the algorithm

The proposed algorithm is designed and implemented in an existing POS terminal software 'Gemalto', in which a 'Shift Report' functionality is added and does not work with Cash Register, using C++ 11 language version and standard versions of its libraries (STL), as well as 'Gemalto' company libraries and APIs for working with the device. An older version of the C++ language, for example C++98, can also be used for the implementation of the algorithm. A class diagram for a basic data type (an object of the CShift class) is presented in Figure 3: Only the basic data members and member functions for the CShift class are shown. An object of this class is used to implement the algorithm - in an independent parallel process (a thread that is created by default after the typical initialization procedures typical of working in POS terminals) in the main algorithm. The algorithm is realized and implemented with C++/11.

Functions such as GetCurrentTime(), GetData(), Calculation() perform the more essential operations of the algorithm (presented in the class diagram for CShift). They sequentially access the statistics data, and after each transaction, the necessary update is added in variables and XML file, which are current for the given time period, sums by types and types of cards, also by types of transactions. These calculations are performed in a separate thread running in parallel to the main algorithm in the POS terminal. Safe handling of shared resources (of the main process and the additional algorithm thread) is implemented with a CRITICAL_SECTION global object from the C++ standard STL library.

After the initial power-up of the device, a procedure for initializing multiple parameters begins - data related to time, device parameters and other auxiliary information. This dataset as information is used by the main algorithms and a part of it is needed for the proposed algorithm. Objects from classes are used to protect when working with common resources, such as data for working with bank transactions. It is built in such a way that the existing minimum sufficiency of parameters with which the main algorithm works in the POS terminal is automatically accessed. Additional new item is added to the main terminal menu for possible user interaction. As an added process, in the process of the normal operation of the terminal, the algorithm processes the data for reporting and statistics, which in this solution are:

- name of commercial establishment, address, etc.
- the previously selected time period (yyyy-mm-dd-hh-MM-ss) (year-month-day-hour-minute-second)
- total number of transactions and total amount for the time period - total number and total amount for transactions by card issuer (Visa, MasterCard, Euro Card, Diners, etc.)
- total number and total amount for transactions by card type (debit, credit)
- type of available currency (current amount at the end of the given period).

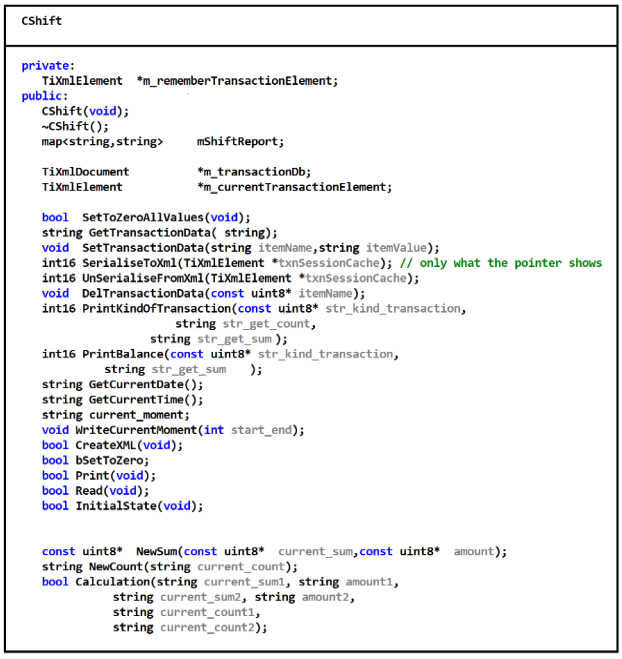

Fig. 3. CShift class diagram.

The algorithm is made up of several steps of calculations (Figure 1), building on current existing ones, which are carried out in the process of the normal operation of the POS terminal, but this process works actually independently and in parallel to the main process and dynamically these data only are processed in it. This data is local to the device, affects reporting and statistics in a summarized and broken down form, and does not participate in the data transfer from the POS terminal.

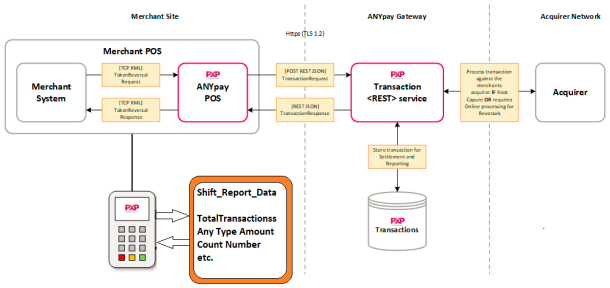

Fig. 4. The subordination of the algorithm in an arbitrary transaction, in the case 'Reversal'.

The user manages the process by requesting a period of time and requesting data. This is done by adding items to the main menu of the existing software. A new time period can be re-claimed and then this triggers a reset if an old claimed time period exists it is destroyed and new start and end data validates a new time period. Information can be requested at any time. The information about the reports is stored in a separate added XML file and through it information is accessed in emergency situations, for example lack of standard power supply - visualization on the display of the POS terminal or printed out at any time (Figure 4). The subordination of the algorithm in an arbitrary transaction, in the case 'Reversal', is given.

To facilitate reporting and statistics, data is processed for a period of time previously selected by the user for the transactions according to the following parameters:

- a period of time for a specific commercial object
- number and amount of 'Visa' card transactions
- number and amount of 'MasterCard' transactions
- number and amount of 'Diners' transactions
- number and total amount of 'Purchases'
- number and amount of 'Refund' transactions
- number and amounts of 'Reversal' transactions
- total

IV. Conclusion

Although the presented algorithm is implemented on Gemalto POS terminals, it does not depend on the manufacturer and type of terminal and can be implemented in terminals manufactured by other companies. The addition of the algorithm does not affect the main operation of the terminal, but adds functionality related to reporting documents and statistics, and the information it processes can be upgraded. A separate thread is used for the algorithm, which runs in parallel to the main process, some existing archive files are used for the data, and the meantime the existing software are not affected or disturbed. With the proposed algorithm, time is saved when reporting information at arbitrary moments of time and it is performed according to precise statistics. Access to the information added for reporting and statistics for a given period of time can be done at any time after its entry until the time of setting a new reporting period. The proposed algorithm also solves issues such as random access to user-required data for already completed transactions from a POS terminal not working with a cash register, as well as the speed with which the presentation of necessary and already processed data is carried out in arbitrary times chosen by the user. For users who have POS terminals but no cash registers, implementing an algorithm that performs comprehensive accounting analysis without the need for additional hardware offers several advantages.

Reduced costs: By eliminating the need to purchase and maintain a separate cash register, consumers can save money.

Improved flexibility: Without the need for a cash register, users can more easily adapt their POS terminal setup to meet their specific needs.

Enhanced reporting: The algorithm can generate detailed reports in real-time, helping to make informed decisions about pricing, marketing and other business strategies.

Increased security: Direct processing of data from the POS terminal reduces the risk of errors and improves financial data protection measures.

Better customer service: Analytics capabilities allow for more personalized service, tracking customer preferences and immediate resolution of payment issues.

Competence: Investing in a modern, innovative algorithm improves reputation and demonstrates users' commitment to providing high-quality services. These benefits can lead to improved efficiency, security, customer service and competitiveness for consumers who choose to use the algorithm instead of a traditional cash register.

Acknowledgment

The authors would like to thank the Research and Development Sector at the Technical University of Sofia for the financial support.

References

[1] EMV2000 Integrated Circuit Card Specification for Payment Systems Book 1 Application Independent ICC to Terminal Interface Requirements Version 4.0 December, 2000

[2] EMV2000 Integrated Circuit Card Specification for Payment Systems Book 2 - Security and Key Management Version 4.0 December, 2000

[3] EMV2000 Integrated Circuit Card Specification for Payment Systems Book 3 Application Specification Version 4.0 December, 2000

[4] EMV2000 Integrated Circuit Card Specifications for Payment Systems Book 4 Cardholder, Attendant, and Acquirer Interface Requirements Version 4.0 December, 2000

[5] SETCo. Secure Electronic Transaction Specification — Books 1–4. SETCo, 1997.