IMPACT: International Journal of Research in Engineering & Technology (IMPACT: IJRET)
ISSN (P): 2347-4599; ISSN (E): 2321-8843
Vol. 5, Issue 8, Aug 2017, 29-40

TRANSACTION INFORMATION EXTRACTION FROM AUTOMATIC TELLER MACHINE ELECTRONIC JOURNAL USING REGULAR EXPRESSION

Ojulari H. O., Arulogun O. T. & Oke A. O

Research Scholar, Department of Computer Science and Engineering, Ladoke Akintola University of Technology, Oyo State, Nigeria

ABSTRACT

The vast majority of Automatic Teller Machines (ATMs) researches is mostly focused on security and ATM modeling, but no study has considered extracting financial information from the electronic journal (EJs). ATM Customer transactions are recorded in a semi-structured text file called EJ. This makes it difficult to run a direct search query on such format to resolve transaction disputes in banks. This research focuses on how to extract financial information from ATM EJs. An EJ Parser algorithm was developed to establish information extraction (IE) method. The IE applied a divide and conquer concept to decompose the EJ into sub-problems of unit transaction sessions, and named entity recognition (NER) was performed to identify all financial transaction tokens or entities, and the extraction task adopted a regular expression (Regex) as an entity classifier. The algorithm was tested with a collection of live EJ data from a Wincor ATM of a bank, and its performance was evaluated accordingly, using standard performance metrics such as precision, accuracy, f-secure, misclassification and recall. The algorithm indicated 99%, 99.7%, 99.7% of precision, recall and accuracy respectively. However, there were a few exceptions that happened as misclassification of which, were traced to 'comments' and 'avail balance' entities.

KEYWORDS: Automatic Teller Machine (ATM), ATM State chart, EJ Parser, ATM Electronic Journal, EJs, EJ, NDC, CEN XFS, NCR

INTRODUCTION

Khalifa and Saadan (2013) defined automatic teller machine (ATM) as a computerized telecommunications device and real-time system that provides the clients of a financial institution with access to their bank accounts in a public space, without the intervention of the administration of the financial institution. These machines are found at most supermarkets; convenience stores and travel centers (Bowen, 2000). There are various brands of ATMs such as NCR, Wincor, Diebold, King Teller, and Hyosung deployed to Nigerian banking industry. Generally, ATM runs on an operating system (OS) such as Windows and Linux; and device drivers called CEN XFS and ATM client applications (e.g. Process or NCR direct connect). However, all ATMs deployed in Nigeria run on Microsoft Windows OS, XP or Windows 7. Currently, almost all the ATMs in Nigeria have been migrated to Windows 7.

Concepts, Wang, Zhang, Sheu, Li, and Guo (2010) described ATM as a 5-tuple finite state machine (FSM), which assumes a set of states and a set of state transition functions. Wang *et al* modeled ATM using a transition diagram and a transition table. The ATM system comprises subunits like Card Reader, Keypad, Monitor, Bill disburser (a unit that dispenses money), Bill storage (that stores money), and System clock. All these subunits are connected to ATM processor (which can be a personal computer, PC); and Wang, *et al* (2010) illustrated a model to conceptualize an ATM system as shown in Figure 1.

ATM operation depends on events and states. An event could be a stream or a combination of multiple signals or just a single signal which can be either human or system

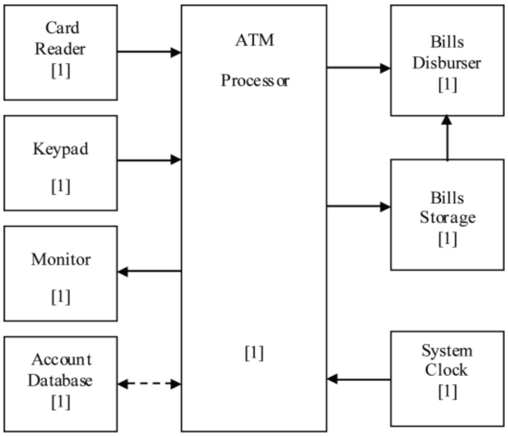


Figure 1: The Conceptual Model of the ATM System

(**Source:** Wang, 2010)

Invoked; hence it is an atomic occurrence and has theoretically zero duration Gomaa (2011). Examples of events are card inserted, pin entered, shutter opened etc. Card inserted, for instance, always precedes PIN entered into the state-flow. The state chart in Figure 2 illustrates how events and states interrelated during ATM operation. The state chart was adapted (Gomaa, 2011). ATM requires a set of input signals (events) before the transition can occur, depending on its current state. The sequence of these states is defined in the state flow received from the ATM host server as part of "ATM download". At every transition, there is usually an entry in the EJ detailing the transaction flow and the interoperability between the ATM and the user. This is what brings about the ATM electronic journal. This EJ contains both ATM message and financial transactions performed on ATM. The area of concern is the customer or financial transactions, which are being recorded in a semi-structured text format on ATM as an electronic journal or EJ, as usually called in the industry. EJ samples adapted from Hyosung and Wincor ATM are shown in Figure 3 a and b, respectively.

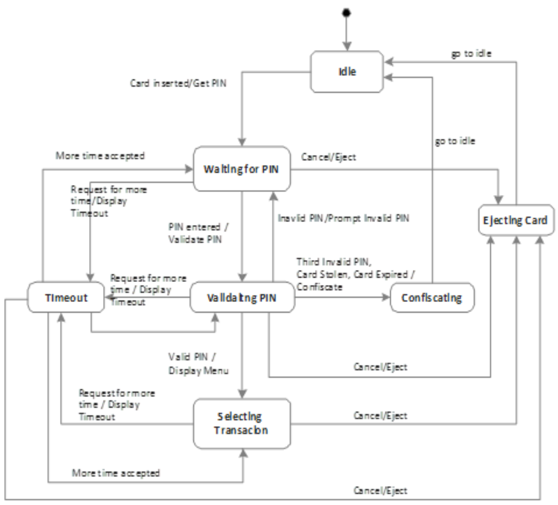


Figure 2: ATM State Chart (Adapted: Gomaa, 2011)

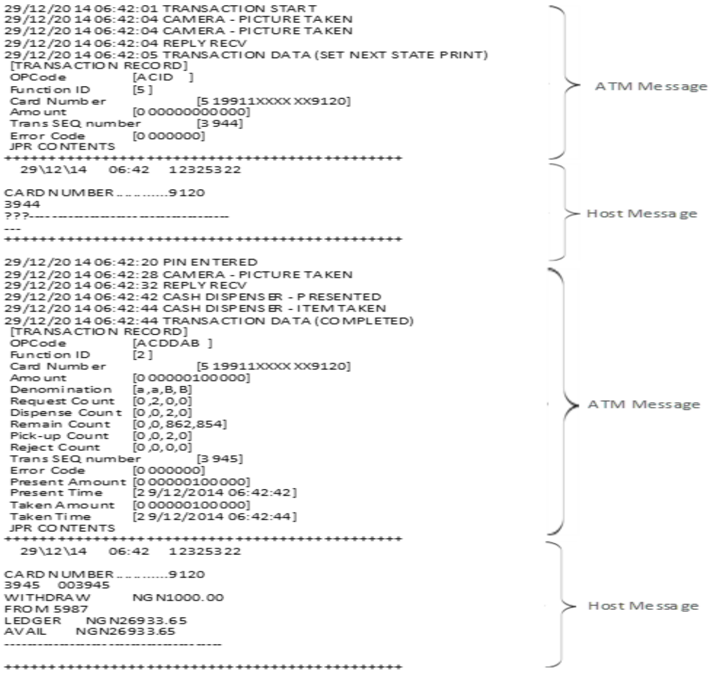


Figure 3 (A) EJ Sample (Adapted From Hyosung ATM)

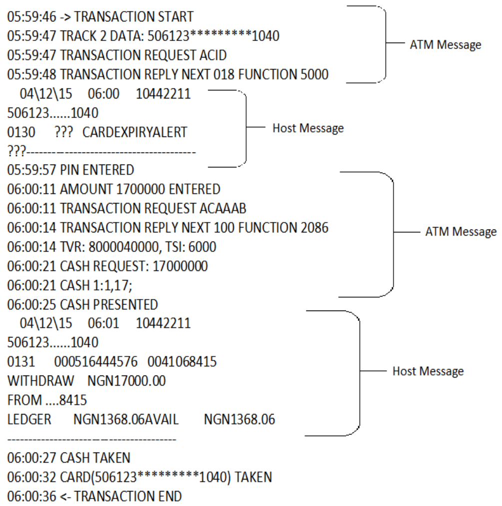


Figure 3 (B) EJ Sample (Adapted From Wincor ATM)

RELATED WORK

There is no relatively specific research work at the time of writing this paper on the ATM electronic journal. Nonetheless, there are few pieces of research on extracting information from both semi-structured and unstructured electronic documents. Information extraction process from a text file is similar to the text mining process. Tan (1999) emphasized that text is the most natural form of storing information, and mining it has a higher commercial potential than data mining. Tan stated that 80% information of an organization is in text documents. However, text mining is much more intricate than data mining; this is because text is naturally unstructured. However, Tan created a framework consisting two phases as shown in Figure 4. These are:

- Text refining that transforms text documents into an intermediate form, IF.
- Knowledge distillation that deduces patterns or knowledge from the IF.

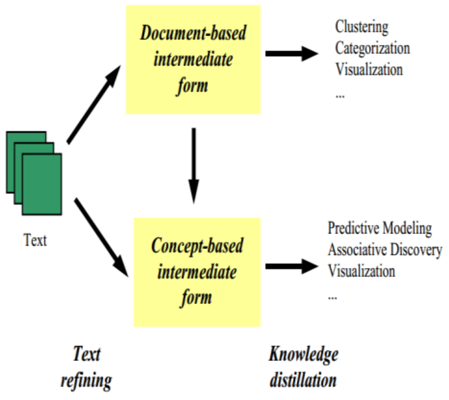


Figure 4 Tan Text Mining Frameworks

(**Source:** Tan, 1999)

Tan further views text mining as a collection of information retrieval, text analysis, information extraction, clustering, categorization, visualization, database technology, machine learning, and data mining.

Garofalakis, Rastogi, and Shim (1999) postulated an algorithm called Sequential Pattern mining with Regular expression constraints (SPIRIT). Conventional mining systems provide users with only a very restricted mechanism (based on minimum support) for specifying patterns of interest. Garofalakis*et al* (1999), proposed the use of regular expressions (REs) as a flexible constraint specification tool that enables user-controlled focus to be incorporated into the pattern mining process. The main peculiar factor among the proposed schemes is the degree to which the RE constraints are enforced to trim the search space of patterns during computation.

Kawtrakul and Yingsaeree (2005) proposed a unified framework to extract metadata automatically from various forms of electronic documents such as such, as pdf, doc, and image, excel, and text files using regular expressions. The use of a regular expression to extract information has been a dominant practical IE method for several years (Li, Krishnamurthy, Raghavan, Vaithyanathan, and Jagadish, 2008), but creating a regular expression for complex information extraction tasks is time-consuming and tedious. Kawtrakul*et al* designed a system comprises an optical character recognition (OCR) that extracts the content and converts it to a standard text format; and discovered knowledge is analyzed.

SYSTEM DESIGN

This paper focuses on how to extract customer transactions from ATMs. The research methods involve the use of information extraction (IE) method to extract specific pieces of data from EJ document. As a part of the optimal means to devise an IE method, context-free grammar (CFG) was implemented to analyze the production rule of ATM host message derivation in the EJ. Regular expression (Regex) was used to extract named entities such as card number or Primary Account Number (PAN), transaction type, transaction serial number (TSN), amount, comment, transaction date and time. It was easier and faster when 'divide and conquer' concept was applied to break EJ content into transaction sessions. The extraction was recursively performed across multiple transaction sessions. All these procedures were implemented in the algorithm design.

Algorithm Design

The algorithm design was segmented into two main parts; the EJ decomposition and information extraction.

EJ Decomposition

The EJ decomposition needs the concept of divide and conquer paradigm to break down an EJ file into transaction sessions. Each session has one or more transaction events such as Withdraw, Inquiry, Transfer, Virtual tops up, Bill payment and so on as defined by a bank. This is because, a customer might decide to perform multiple transactions within a cycle or session (from card insert till card eject). The transaction cycle is decomposed into events and all the information pertaining to customer transaction is captured or extracted into a dataset (or session table) using NER.

The dataset is an accumulation of customer transaction events. Typically, EJ File comprises multiple transaction sessions. However, each transaction is extracted based on regular expression defined in the CFG. The extraction procedure follows the algorithm developed (EJ Parser algorithm). The expensive part of the algorithm is the conversion module, i.e. structuring the extracted entities into a relational dataset. The extracted elements are stored in the database.

Information Extraction

Information regarding customer transactions is embedded in both ATM messages and host messages, which follow a language rule. There exist both regular and non-regular patterns within the journal. A non-regular language must thus include an infinite number of words. If a language includes an infinite number of words, there is no bound on the size of the words in the language. In the language rule, regular expressions help to generate or describe all strings in the language while finite automata recognize a specific string in the language. This helps to create, host message template. Most banks in Nigeria design the host message template themselves obeying a rule based on central bank regulations. The same principle was adopted to develop the production rule needed in the reflex design. Some notations for non-terminal in the production rules to be considered are expressed in Figure: 5.

Non-terminal	Meaning	Sample
ATM_JOUR	ATM journal	NIL
TRANSACTION	ATM Transaction within a session	05:59:46 -> TRANSACTION START 05:59:47 TRACK 2 DATA: 506123*********1040 05:59:57 PIN ENTERED 06:00:11 AMOUNT 1700000 ENTERED 06:00:11 TRANSACTION REQUEST ACAAAB 06:00:14 TRANSACTION REPLY NEXT 100 FUNCTION 2086 06:00:14 TVR: 8000040000, TSI: 6000 06:00:21 CASH REQUEST: 17000000 06:00:21 CASH 1:1,17; 06:00:25 CASH PRESENTED 04\12\15 06:01 10442211 506123......1040 0131 000516444576 0041068415 WITHDRAW NGN17000.00 FROM ....8415 LEDGER NGN1368.06AVAIL NGN1368.06 --- 06:00:27 CASH TAKEN 06:00:32 CARD (506123*********1040) TAKEN 06:00:36 <- TRANSACTION END
ATM_MSG	ATM Message	05:59:46 -> TRANSACTION START 05:59:47 TRACK 2 DATA: 506123*********1040 05:59:47 TRANSACTION REQUEST ACID 05:59:48 TRANSACTION REPLY NEXT 018 FUNCTION 5000
HOST_MSG	Host message	04\12\15 06:01 10442211 506123......1040 0131 000516444576 0041068415 WITHDRAW NGN17000.00 FROM ....8415 LEDGER NGN1368.06AVAIL NGN1368.06 --- 06:00:27 CASH TAKEN
PAN_TSN	PAN (card no) with TSN (transaction serial no)	PAN=506123......1040 and TSN= 0131
CUST_TRANS	Customer transaction detail	WITHDRAW NGN17000.00 FROM ....8415 LEDGER NGN1368.06 AVAIL NGN1368.06
EVENT	Transaction event	WITHDRAW, INQUIRY, INTERBANK TRANSFER, MINISTATEMENT, THIRD PARTY PAYMENT, etc.

Figure 5: Non-terminal notations for designing ATM journal CFG.

The Developed Production Rule is given As Follows:

ATM_JOUR -> **TRANSACTION**
TRANSACTION -> **ATM_MSG HOST_MSG ATM_MSG**
ATM_MSG -> **INFO** | **ERRMSG**
INFO -> pre transaction info+ | post transaction info+
ERRMSG -> errmsg+
HOST_MSG -> ϵ | **HOST_MSG**
HOST_MSG -> date<**SP**>time<**SP**>terminal<**LF**>**PAN_TSN**<**LF**>**CUST_TRANS**<**LF**>**HOST_COMMENT**
PAN_TSN -> **PAN**<**SP**>**TSN**<**SP**>**UNICODE** | [**PAN** |**TSN**<**SP**>**UNICODE**]
PAN -> digit+ char* digit+
TSN -> digit [4]
UNICODE -> ϵ | digit [6]
CUST_TRANS -> **EVENT**<**SP**>**AMOUNT**<**LF**>**ACCT**<**LF**>**LEDGER**<**LF**>**AVAIL**<**LF**>
EVENT -> ϵ | 'Withdraw' | 'Inquiry' | 'Transfer' | 'Virtual top' | 'Third party payment' |
'Cash deposit' | 'Mini statement'| 'Advance Prepaid'
AMOUNT -> ϵ | **CURR MONEY**
CURR -> 'ngn' | 'usd' | 'cfa' | 'gbp'
MONEY -> digit+, digit+.digit [2] | digit+.digit [2]
ACCT - > 'From' [.]*<**SP**>**ACCTNO**
ACCTNO -> digit* | word
LEDGER -> ϵ | 'Ledger'<**SP**>**AMOUNT**
AVAIL -> ϵ | 'Avail'<**SP**>**AMOUNT**
HOST_COMMENT -> ϵ | transaction_comment
<**SP**> -> \t
<**LF**> -> \r | \r\n | \n
digit = [0….9]
word = \w+
errmsg* = \w+

Figure: 6

The Following Named Entities and Regex Chunks Were Considered Based on the Production Rule

- Transaction Date = (? <Date>\d{2}[/|\\]\d{2}[/|\\]\d{2})
- Transaction Time = (? <Time>\d{2}:\d{2})
- ATM used = (? <Terminal>\w+)
- PAN = (? <PAN>\d*[.|*]*\d+)
- Transaction Serial No = (? <TSN>\d{4})
- Withdraw = (? <Transaction> (WITHDRAW)? [ ]+ (?<CurrencyCode>\w{3})? (? <Amount>\d* [.] \d+)?
- Inquiry = (? <Transaction>(INQUIRY)] *)
- Transfer = (? <Transaction>(INTERBANK TRANSFER))] * (? <CurrencyCode>\w{3})? (? <Amount>\d* [.] \d+)?) \r\n (FROM (? <From Account>([.] |\w)+) TO (? <To Account>>([.] |\w)+))
- Mini statement = (? <Transaction>(MINISTATEMENT))
- Bill payment = (? <Transaction>(THIRD PARTY PAYMENT)] * (? <CurrencyCode>\w{3})? (? <Amount>\d* [.] \d+)?) \r\n
- However, placeholders such as (? <Date>), (? <Time>), (? <Transaction>) etc. Are variables holding the matches from the input string.

THE DEVELOPED ALGORITHM

EJParserAlgorithm

```
• Input: Read EJ File as Filestream;
• Declare ejstream as string :
      Let ejstream = FileStream.Read(EJ) as string;
• Apply divide and conquer paradigms

Step1: Split ejstream into array of transaction session using transaction delimiters
      Let session_delimeters = { transaction start, transaction end };
      Let ejsessionArray = Split (ejstream, session_delimeters);
      /*Create object array*/
          Let data = object of data array;
      /*Initialize index*/
          Let i = 0;
Repeat
    Step 2: Let ejsession = ejsessionArray[i];
        - Match all transaction event(s) in ejsession with defined production rule
          using regex;
        - Let matchArray = Match (ejsession, regex);

    Step 3: Tokenize transaction event in matchArray into fields based on regex
          placeholders; and add  them to data object.
          if (matchArray != null)
            Tokenize (matchArray, {pan, tsn, date, time, terminal, transtype, currency,
                      amount, account, ledger, avail, comment, error, status});

           /*save extracted entities in data object*/
            Add (data, {pan, tsn, date, time, terminal, transtype, currency, amount,
                 account, ledger, avail, comment, error, status});

          else goto Step 4:

    Step  4: increment the index by 1
            Let i = i +1;
Until
        i = ejsessionArrayLength - 1;
```

Figure: 7

The algorithm decomposes EJ file into transaction entities, which are broken down mathematically in equations 1 and 2.

$$f(t,j) = \sum_{t=1}^{n} Match(S_t) \qquad (1)$$

$$extract(t,j) = \sum_{i=1}^{n} f(t_i,j) \ + \Phi \qquad (2)$$

Equation 1 says the sum of matched sessions found in a transaction cycle, it is a function of financial transactions *t in* EJ, *j*; while equation (2) is an aggregate of equation 1, which are entities found in the transactions; and n is the total number of transactions in the EJ.

Where S_1, S_2,..., $S_{t=n}$ are transaction sessions of the ejstream, French S_t, obtained by a divide and conquer method, it is transaction containing information, j is the journal and Φ is the error handler. EJParser algorithm embedded with regular expressions involves the use of top-down parsing technique to extract transaction details of ATM customers. For every EJ file per daily transactions, it holds that the time to decompose the entire EJ file according to equation 3 is:

$$\mathrm{T(n)} = \mathrm{T}\big(\mathrm{Decom(ejstream)}\big) = \mathrm{T}\big(\mathrm{Match(ejsession)}\big) + \mathrm{T}\big(\mathrm{Tok(event)}\big) + \mathrm{T}\,(\mathrm{Com\ (tokens)}) \ (3)$$

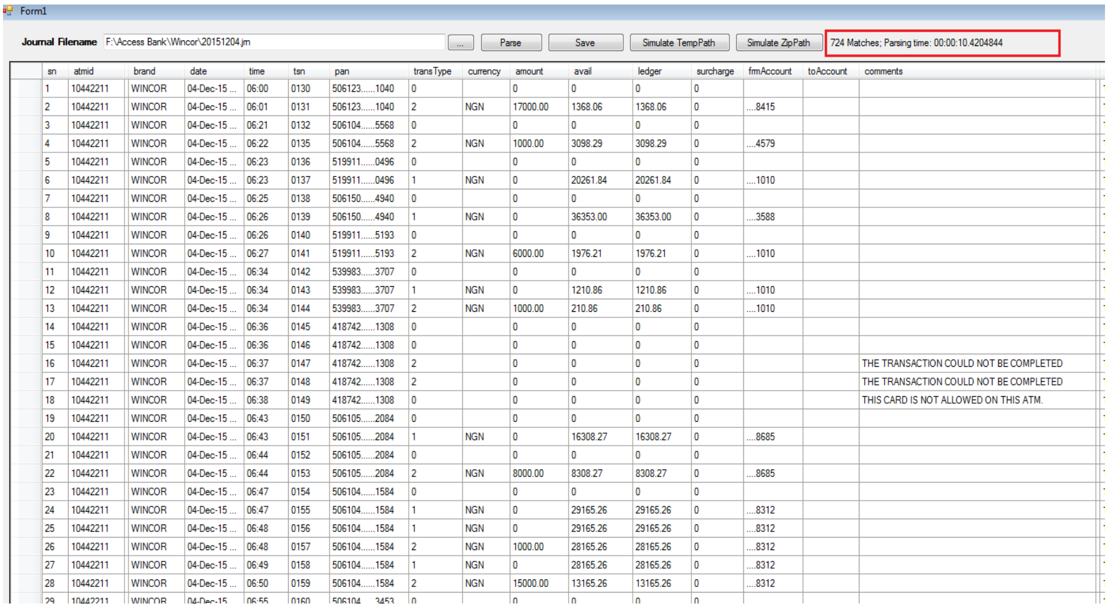
Where, Decom = decompose, Tok = Tokenize, Com = Combine

EJParser algorithm will extract information within O (n) approx. Running time.

RESULT ANALYSIS

The algorithm was implemented as a software application using Microsoft.NET technology and was tested with a live EJ from Wincor Nixdorf ATM. Figure 7 shows the user interface of the application with the named entities extracted.

An EJ file that contains 724 transactions performed on the 4th of December, 2015 was collected from a Nigerian bank. It took the application 10s to extract the named classes from 724 transactions. All the actual transaction events performed that day at the ATM are described in Figure 9, and the extracted are shown in Figure 10.



sn	atmid	brand	date	time	tsn	pan	transType	currency	amount	avail	ledger	surcharge	frmAccount	toAccount	comments
1	10442211	WINCOR	04-Dec-15 ...	06:00	0130	506123......1040	0		0	0	0	0			
2	10442211	WINCOR	04-Dec-15 ...	06:01	0131	506123......1040	2	NGN	17000.00	1368.06	1368.06	0	8415		
3	10442211	WINCOR	04-Dec-15 ...	06:21	0132	506104......5568	0		0	0	0	0			
4	10442211	WINCOR	04-Dec-15 ...	06:22	0135	506104......5568	2	NGN	1000.00	3098.29	3098.29	0	4579		
5	10442211	WINCOR	04-Dec-15 ...	06:23	0136	519911......0496	0		0	0	0	0			
6	10442211	WINCOR	04-Dec-15 ...	06:23	0137	519911......0496	1	NGN	0	20261.84	20261.84	0	1010		
7	10442211	WINCOR	04-Dec-15 ...	06:25	0138	506150......4940	0		0	0	0	0			
8	10442211	WINCOR	04-Dec-15 ...	06:26	0139	506150......4940	1	NGN	0	36353.00	36353.00	0	3588		
9	10442211	WINCOR	04-Dec-15 ...	06:26	0140	519911......5193	0		0	0	0	0			
10	10442211	WINCOR	04-Dec-15 ...	06:27	0141	519911......5193	2	NGN	6000.00	1976.21	1976.21	0	1010		
11	10442211	WINCOR	04-Dec-15 ...	06:34	0142	539983......3707	0		0	0	0	0			
12	10442211	WINCOR	04-Dec-15 ...	06:34	0143	539983......3707	1	NGN	0	1210.86	1210.86	0	1010		
13	10442211	WINCOR	04-Dec-15 ...	06:34	0144	539983......3707	2	NGN	1000.00	210.86	210.86	0	1010		
14	10442211	WINCOR	04-Dec-15 ...	06:36	0145	418742......1308	0		0	0	0	0			
15	10442211	WINCOR	04-Dec-15 ...	06:36	0146	418742......1308	0		0	0	0	0			
16	10442211	WINCOR	04-Dec-15 ...	06:37	0147	418742......1308	2		0	0	0	0			THE TRANSACTION COULD NOT BE COMPLETED
17	10442211	WINCOR	04-Dec-15 ...	06:37	0148	418742......1308	2		0	0	0	0			THE TRANSACTION COULD NOT BE COMPLETED
18	10442211	WINCOR	04-Dec-15 ...	06:38	0149	418742......1308	0		0	0	0	0			THIS CARD IS NOT ALLOWED ON THIS ATM.
19	10442211	WINCOR	04-Dec-15 ...	06:43	0150	506105......2084	0		0	0	0	0			
20	10442211	WINCOR	04-Dec-15 ...	06:43	0151	506105......2084	1	NGN	0	16308.27	16308.27	0	8685		
21	10442211	WINCOR	04-Dec-15 ...	06:44	0152	506105......2084	0		0	0	0	0			
22	10442211	WINCOR	04-Dec-15 ...	06:44	0153	506105......2084	2	NGN	8000.00	8308.27	8308.27	0	8685		
23	10442211	WINCOR	04-Dec-15 ...	06:47	0154	506104......1584	0		0	0	0	0			
24	10442211	WINCOR	04-Dec-15 ...	06:47	0155	506104......1584	1	NGN	0	29165.26	29165.26	0	8312		
25	10442211	WINCOR	04-Dec-15 ...	06:48	0156	506104......1584	1	NGN	0	29165.26	29165.26	0	8312		
26	10442211	WINCOR	04-Dec-15 ...	06:48	0157	506104......1584	2	NGN	1000.00	28165.26	28165.26	0	8312		
27	10442211	WINCOR	04-Dec-15 ...	06:49	0158	506104......1584	1	NGN	0	28165.26	28165.26	0	8312		
28	10442211	WINCOR	04-Dec-15 ...	06:50	0159	506104......1584	2	NGN	15000.00	13165.26	13165.26	0	8312		
29	10442211	WINCOR	04-Dec-15	06:55	0160	506104......3453	0		0	0	0	0			

Figure 8: Application User Interface With Data Extracted.

Transaction Events	Frequency in the EJ
ADVANCE PREPAID	10
CHANGE PIN	4
INQUIRY	103
INTERBANK TRANSFER	2
UNKNOWN*	311
WITHDRAW	294
Total:	724

Figure 9 Actual Transaction Events Performed on the Wincor ATM

Note: There are some are tagged unknown because they were not consummated

Class/Entity	Actual Count	Correctly Extracted
PAN	724	724
TSN	724	724
Terminal	724	724
Date	724	724
Time	724	724
Transaction Type	724	724
Amount	306	306
Avail Balance	326	314
Opcode	724	724
Comment	74	74
	5,774	5,762

Figure 10: Wincor Data Extraction

Performance Evaluation

There are statistical measures of performance that were considered to evaluate the regular expression, and these are accurate, true positive rate (recall or sensitivity), misclassification rate, precision, and F-measures.

Some symbols are defined as follows in order to establish some equations:

Φ – Entities to be identified

R_x – Input regular expression, regex

E_j– Electronic Journal document

Supposing, $M(R_x, E_j)$ represents the set of matches obtained by evaluating regex R_x over an electronic journal (EJ) collection E_j, the outputs are defined over 4 possible outcomes; and these are:

$M_{T+}(R_x, E_j) = \{\, x \in M(R_x, E_j) : x \ instance \ of \ \Phi \,\}$ –The M_{T+} is the true positive match for R_x.

$M_{T-}(R_x, E_j) = \{\, x \in M(R_x, E_j) : x \ instance \ of \ \Phi \,\}$– The M_{T-} is the true negative match for R_x.

$M_{F+}(R_x, E_j) = \{\, x \in M(R_x, E_j) : x \ instance \ of \ \Phi \,\}$ – The M_{F+} is the false positive match for R_x.

$M_{F-}(R_x, E_j) = \{\, x \in M(R_x, E_j) : x \ instance \ of \ \Phi \,\}$ –The M_{F-} is the false negative match for R_x.

The regex R_x is designed to identify instances of Φ. The following metrics were used to evaluate the regex and validate the extraction quality in the search space.

To calculate the accuracy A;

$$A(R_x, E_j) = \frac{M_{T+}(R_x, E_j) + M_{T-}(R_x, E_j)}{actual\ total\ entities} \quad (4)$$

To calculate the misclassification rate M;

$$M(R_x, E_j) = \frac{M_{F+}(R_x, E_j) + M_{F-}(R_x, E_j)}{actual\ total\ entities} \quad (5)$$

To calculate the precision P;

$$P(R_x, E_j) = \frac{M_{T+}(R_x, E_j)}{M_{T+}(R_x, E_j) + M_{F+}(R_x, E_j)} \quad (6)$$

To calculate the true positive rate or recall or sensitivity, R;

$$R(R_x, E_j) = \frac{M_{T+}(R_x, E_j)}{M_{T+}(R_x, E_j) + M_{F-}(R_x, E_j)} \quad (7)$$

To calculate the F-measure or score;

$$F_1\ \text{measure} = \frac{2.P.R}{P + R} \quad (8)$$

Figure 11 shows the confusion matrix of the entire classes extracted from Wincor EJ while Figure 12 shows classification analysis on randomly selected five entities such as ‘comment’, ‘avail balance’, ‘PAN’, ‘Transaction Type’ and ‘Amount’ using true positive (TP), false positive (FP), true negative (TN) and false negative (FN).

		Extracted class									
T= 5,762		A	B	C	D	E	F	G	H	I	J
Actual Class	A	724	0	0	0	0	0	0	0	0	0
	B	0	724	0	0	0	0	0	0	0	0
	C	0	0	724	0	0	0	0	0	0	0
	D	0	0	0	724	0	0	0	0	0	0
	E	0	0	0	0	724	0	0	0	0	0
	F	0	0	0	0	0	724	0	0	0	0
	G	0	0	0	0	0	0	306	0	0	0
	H	0	0	0	0	0	0	0	314	0	8
	I	0	0	0	0	0	0	0	0	724	0
	J	0	0	0	0	0	0	0	0	0	74

Figure 11: Wincor Confusion Matrix for 5,762 Data Extracted

A = PAN, B = TSN, C = Terminal, D = Date, E = Time,F = Transaction Type, G = Amount, H = Avail Balance, I = Opcode, J = Comment

(a) (b)

Confusion Matrix for Comment class	
74 TPs (actual comments that were correctly classified as comments)	8 FPs (others that were incorrectly labeled as comments)
0 FNs (comments that were incorrectly marked as others)	5680 TNs (all the remaining classes correctly classified as non-comments)

Confusion Matrix for Avail bal class	
314 TPs (actual avail balance that were correctly classified as other classes)	0 FPs (other classes that were incorrectly labeled as avail balance)
8 FNs (avail balance that were incorrectly marked as other classes)	5448 TNs (all the remaining classes correctly classified as non-avail balance)

(c) **(d)**

Confusion Matrix for PAN class	
724 TPs (actual PANs that were correctly classified as other classes)	0 FPs (other classes that were incorrectly labeled as PAN)
0 FNs (PANs that were incorrectly marked as other classes)	5038 TNs (all the remaining classes correctly classified as non-PANs)

Confusion Matrix for Trans Type class	
724 TPs (actual transaction types that were correctly classified as transaction type)	0 FPs (other classes that were incorrectly labeled as transaction types)
0 FNs (transaction types that were incorrectly marked as other classes)	5038 TNs (all the remaining classes correctly classified as non-transaction types)

(e)

Confusion Matrix for Amount class	
306 TPs (actual Amounts that were correctly classified as Amounts)	0 FPs (other classes that were incorrectly labeled as Amount)
0 FNs (Amounts that were incorrectly marked as Other classes)	5456 TNs (all the remaining classes correctly classified as non-amount)

Figure 12: Classification Analysis for Selected Classes on Wincor EJ

Class	Pr.	Re.	Acc.	Mis.	F
PAN	100	100	99.7	0.3	100
TSN	100	100	99.7	0.3	100
Terminal	100	100	99.7	0.3	100
Date	100	100	99.7	0.3	100
Time	100	100	99.7	0.3	100
Transaction Type	100	100	99.7	0.3	100
Amount	100	100	99.7	0.3	100
Avail Balance	100	98.7	99.7	0.3	97.9
Opcode	100	100	99.7	0.3	100
Comment	90	100	99.6	0.4	94.7

Figure 12: Performance Measure on Wincor EJ

Note: Pr. = Precision, Re. = Recall, Acc. = Accuracy M is. = Misclassification, F = F-measure

CONCLUSIONS

The study has demonstrated the use of the regular expression to extract financial information from ATM electronic journal. Electronic journal of Wincor ATM was examined as a semi-structured document, which was transformed into structured data. An EJ Parser algorithm was established, implemented, and tested. The EJ was broken down into subunits of transactions using divide and conquer concept, and each unit was recursively extracted using pattern extractor. The pattern extractor implemented a text mining process, considering a linguistic analysis of EJ using context-free grammar, and the information extraction module of the algorithm adopted regular expression as a classifier. There are entities of interest to the banks, which are identified as named entities for recognition task.

The results were presented after the algorithm was tested with a collection of data, and its performance was as well evaluated using standard performance metrics; these are precision, accuracy, f-measure, and recall. The evaluation showed that the IE method has both true positive rate and extraction precision value above 90%. Conversely, the average speed of extraction is approximately 20s. There were few exceptions, which were closely observed on 'comments' and 'avail balance' entities. The overall accuracy of the IE is 99.7% and precision is 99% averagely, but banks' expectation is to achieve 100% accuracy and precision, because of financial implication involvement.

REFERENCES

1. Garofalakis, M. N., Rastogi, R., and Shim, K. (1999). SPIRIT: Sequential pattern mining with regular expression constraints. In *VLDB* 99, 7-10.
2. Gomaa, H. (2011). *Software modeling and design: UML, use cases, patterns, and software architectures*. Cambridge University Press.
3. Kawtrakul, A. (1995). A Computational Model for Writing Production Assistant System. The Proceeding NLPRS'95 Natural Language Processing Pacific Rim Symposium, 4-7.
4. Kawtrakul, A., and Yingsaeree, C. (2005). A unified framework for automatic metadata extraction from electronic document. In*Proceedings of the International Advanced Digital Library Conference. Nagoya, Japan*.
5. Khalifa, S. S., and Saadan, K. (2013). The Formal Design Model of an Automatic Teller Machine (ATM). *world*, *3*(4).
6. Li, Y., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., and Jagadish, H. V. (2008). Regular expression learning for information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 21-30. Association for Computational Linguistics.
7. Tan, A. H. (1999). Text mining: The state of the art and the challenges. In Proceedings of the PAKDD Workshop on Knowledge Discovery from Advanced Databases 8, 65.
8. Wang, Y., Zhang, Y., Sheu, P. C., Li, X., and Guo, H. (2010). The Formal Design Model of an Automatic Teller Machine (ATM). *International Journal of Software Science and Computational Intelligence (IJSSCI)*, *2*(1), 102-131.