Self-healing in payment switches with a focus on failure detection using State Machine-based approaches

Hassan Jahedi
Electronic Unit, Islamic Azad University, Faculty of Engineering,

Hamid ShayeghBoroujeni
ShahidRajaee University, Faculty of Computer Science,

Abstract
Composition, change and complexity have attracted everyone's attention towards Self-Adaptive systems. These systems, inspired by the human body, are capable of adapting to changes in the inner and outer environment. The main objective of this study is to achieve a more convenient availability for e-banking services in the payment switch, using self-healing systems and focusing on the failure and the state machine approach. This study was carried out on the 136559039payment switch records of Melal Credit Institute.
Using the concepts of SOA architecture and programming technologies based on the requirements of the self-healing module, the payment switch architecture has been changed and new services are implemented. After re-testing the data, the state machine and self-healing module were able to heal the failures occurred successfully. The results indicate an increase of 2.98% inavailability of the bank's serving after the change of architecture and adding the self-healing module to the payment switch.

Keywords:
Self-Adaptive System, Self-healing system, E-Banking,Availability, Payment Switch

Introduction
Any Time any Place responding to the customer is the top priority for any bank as providing E-Banking services. Given the widespread debates and the emergence of autonomic systems, this topic has come to a variety of areas like Availability, Security, Performance, Inventions, customer Satisfaction et.al. So any financial organization can work on these eras to populate better services.
Self-adoptive systems are kind of context aware systems which sense the environment and analyze the perceptions then react autonomously according the location and existing conditions [1]. In as much as using the knowledge of expertise in respective domain, it is expected that self-adoptive systems cause more satisfying result than a system without this approach.
The purpose of self-* systems is to reduce human intervention and, in fact, making the system autonomous and context aware. Following the way of self-* implementation and related evaluation is compulsory to make a metric for the effectiveness and success of the applied ideas. To reach this aim, techniques such as feature descriptors, OOP, service-oriented architecture and relevant efforts in this direction have been used.
This approach has dedicated view on bank customer's satisfaction in case of doing any type of financial transactions that involve payment switch action and try to degrade the human being involvement in any warning or failure situation by making the payment switch a self-healing module based on state machine concept.
In this study, firstly, the related works are reviewed and next is a brief description about understanding the project vision. In the next step payment switch architecture is exposed. Besides, various types of failures in payment switches are addressed and the respective ways of healing them are explained based on our novel approach. Finally, we will explain the results of the experiments and future work.

Related work
Management automation is not a new issue in the field of computer science. But for many years, systems and system components have faced system control and operational management with increasing complexity in controlling systems. Self-healing, which is one of the features offered by self-adaptive systems, can play an important role in healing and improving system reliability. Therefore, the issue of the production and development of self-healing systems has attracted the attention of many scholars in different fields. A brief description of the number of articles presented is as follows:
Pong et al. [2] presented a self-healing method for healing the quality requirements of software systems. In this study, a priority-based inference algorithm is presented that consists of two parts objective-based deduction and

the feedback control theory.

Stoicescuet al. [3] examined the failure-detection approach and architecture based on the interpretation of object-oriented programming, and, by combining aspect-oriented programmingand artificial intelligence and machine learning methods, and its analysis, they believed that they were detecting and healing a lot of failures.

FazilHanif et al. [4] reviewed the self-healing frameworks in distributed systems in 2013. The logical factors in which the system operates is done in it, are controlled in the first layer. Feeding factors are considered as the second layer. It uses the FLISR module to achieve its own goal, which is itself self-healing. Based on the simulation results, this module has been very effective in reducing the error rate.

In 2014, Schiffer et al. [5] has presented self-healing fuel pumps mapped in memory based on finite state machines. His approach is based on the transformation of the traditional achievements of hardware design into the general framework and search of tables. Instead of relying on custom hardware reduction strategies, such as triple modular redundancy, an approach based on the data error recognition and correction codesis preferred. The logic of memory mapping is one of the self-healing capabilities that can be used in a wide range of finite-state machines and has illustratedits method using FPC and (LUT) fuel pump plan mapping.

Understanding the project site

Transparency of issues, goals and resources is one of the things that should be taken into account in this section and will be identified by knowing the scope of the problem. Self-healing in medical science is a term associated with the process of recycling, which is driven by the patient's own motivation and guidance, and sometimes driven only by instinct.

Self-healing is a mechanism that identifies and heals the issues related to performance and services provided by the electronic system. In somecomponent-oriented systemsand single services, it's usually easy to test and analyze the system, but complex systems are involved in unpredictable issues that lead to problems that are difficult to troubleshoot and heal. Self-healing mechanisms focus on discovering and overcoming these issues. As a result, self- healing is a feature of IT systems that enables such systems to recognize themselves and react to errors so that after an error or failure, the operating state is satisfactorily restored; therefore, the first step is to detect failure points and the way to deal with them.In this study, we have tried to make a good sampling of existing data. For this purpose, from 15th Oct 2015 to 12th Jul 2017, the transactions registered at the Melal Credit Institute, which corresponds to 14737917 card, have been selected. The total number of transactions registered at this time is 136559039. An almost two-year timeframe for testing and testing has been considered to provide more reliable results from experiments. Frequency of transactions of the institution is based on the response codes received in Table 1. According to this table, the response code "00" has the highest percentage with the rate of 91.570. Due to the attention of this study on the systemic errors, most of the codes expressed in Table 2 are emphasized.

The degree of success and failure of transactions based on a variety of business and system failures is stated in Table 3. As can be seen in this table, business failures from system failures have devoted more share to themselves. The amount of successful transactions is equal to 91.85% of total transactions and the amount of unsuccessful transactions is equal to 8.15% of total transactions.

Commercial errors are failures that are created by customers and the bank is not responsible for these failures [6]. For example, customers who do not enter their passwords correctly or those who try to transfer funds above the permissible limits, so healing commercial errors requires the provision of e-banking training to bank customers, which is not in line with the objectives of this study.

Payment switch architecture

It is important to identify systems, standards, modules, and the relationships between them in order to diagnose failures. The Melal Credit Institute Payment Switch to connect with the Interbank Information Exchange Network uses ISO8583 Version 1987 and uses the RMI communication protocol to communicate with other internal services. Overview of the payment switch and its associated services before applying the self-healing module[7] is shown in Fig 1 and the primary and internal architecture of the payment switch is shown in Fig2.

In this architecture, there are four layers that each layer has its own tasks. Accepting and export transaction calls are located in the first layer. This layer is based on ISO8583 standard. The task of this layer is to receive customer requests. After receiving the requests, the accuracy of the request and its security issues are reviewed.

Table 1.The frequency of response codes in the time period from 15th Oct 2015 to 12th Jul 2017

Response Code	Number	Ratio to Total%	Description
00	125, 047 395	91.570	Successful transaction
02	379, 991	0.278	System error
06	177, 129	0.130	System error
09	304, 223	0.223	System error
12	289, 583	0.212	System error
25	425, 760	0.312	System error
30	11, 989	0.009	System error
31	2, 794	0.002	Commercial error
34	166, 319	0.122	Commercial error
68	1 631 402	1.195	System error
77	50, 930	0.037	System error
80	40, 219	0.029	Commercial error
84	742, 227	0.544	System error
90	99, 207	0.073	System error
91	447, 428	0.328	System error
93	479, 370	0.351	System error
94	1, 186	0.001	System error
96	12, 422	0.009	System error
The other	6,249,465	4,576	Commercial error

Table 2. Status of payment switch transactions in the time periodfrom15th Oct 2015 to 12th Jul 2017

Transaction Status		Frequency	Percent
Successful transaction		125 , 427 , 386	91.85
Unsuccessful transaction	System	4 , 672 , 856	3.42
	Commercial	6 , 458 , 797	4.73
Total		136 , 559 , 039	100.00

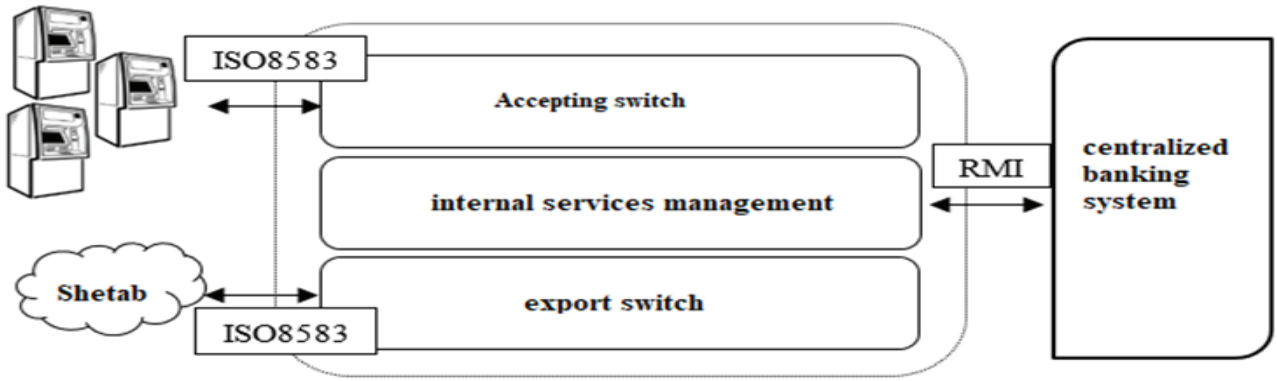
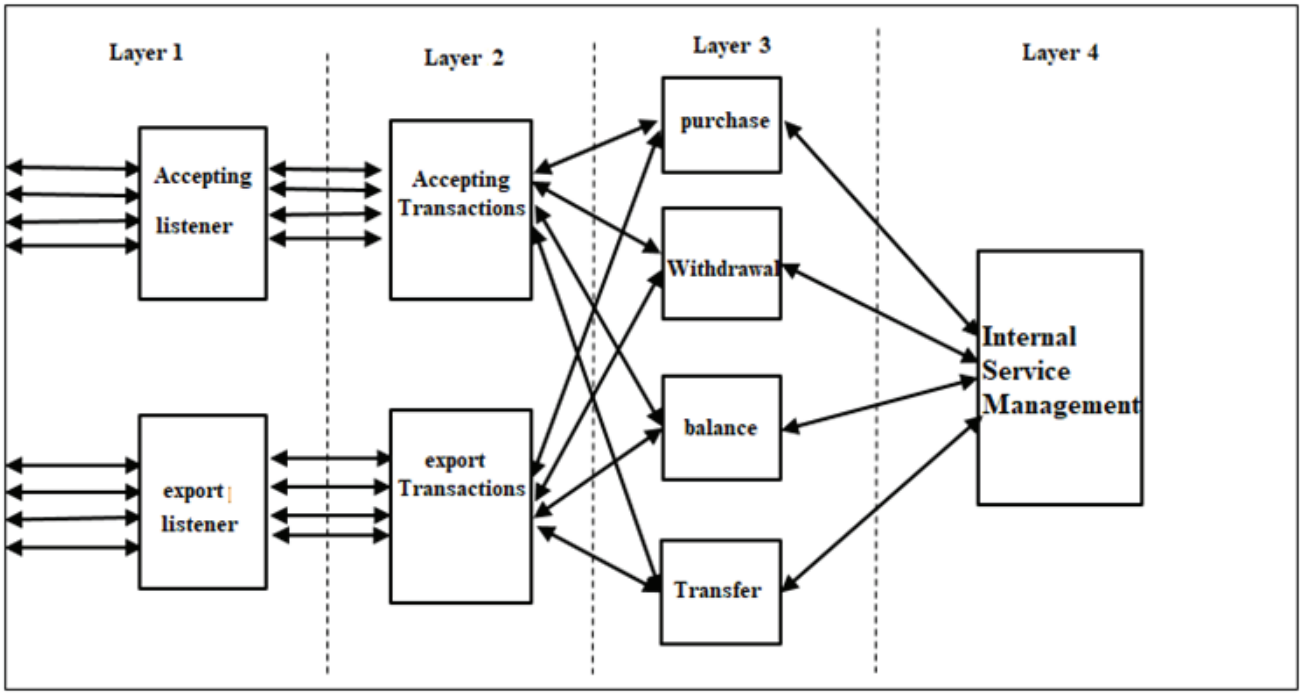


Fig 1. Overview of the payment switch and its associated services before applying the self-healing component

If all of the controls in this section are verified, then the client request is sent as a transaction to the second layer that manages transaction and issuer transactions. This layer will call each transaction based on the desired service transaction type ID.The third layer is responsible for transactions received transactions into the language of the destination services.On the contrary, it is true too. That is, if the transaction response is received from the servers, those responses will be translated into the transaction requester language. After the transaction is translated, the transaction will be requested from the servers' management layer. At this stage, the fourth layer should specify the server depending on the service type, and send the desired request from the server, and communication with the servers is based on the RMI standard. The layers must all wait to receive the response from the layers of the requester, and, after receiving the response, perform their duties to send response to the upstream layers.

Fig 2. Payment Switch Layered Architecture



The first, second, and fourth layers are highly sensitive due to the focus of requests and transactions, and based on the knowledge of the expert person is more likely to occur in these layers, so failure management for these layers should be considered.

Features

Today, one of the most important features of any software system is qualitative features [6]. With the advances made and the development of various tools for software production, the development of software that meets the needs of the customers is easy and fast. At present, the difference between two applications is determined by the ability of software to meet the expected qualitative features [8]. In this research, the goals of "availability" and "Quality of Service" are considered.

Availability: Availability is defined as the ability of users to access information and services from each terminal, which is dependent on many criteria [6]. These criteria include the content, hardware, software format and user settings; adequate and reliable connections; environmental conditions and capabilities and disadvantages for users.

Quality of Service:

There are several quality models for the system. One of the most relevant ones is presented in [9], which categorizes quality in an organized set of main and secondary features as follows.

Efficiency: Suitability (appropriateness), accuracy, interoperability, acceptance, security

Reliability: complete, recyclable, fault tolerance

Usability: learnable, ready-to-use, operational

Productivity (Performance): Time Performance, Source Performance

Ability to heal (maintainability): Sustainability, Analytic, Modifiable, Testable

Transferability: customizable, interchangeable, adaptive

If a system has quality of service conditions, can achieve

maximize customer satisfaction; therefore, the main motivation is to provide more profits to customers satisfied [6]. This action is evident in the customer satisfaction chain reaction in Fig 3.

Fig 3. Customer Satisfaction and Bank Profit Cycle

Relationship with qualitative features: Self-adoptive features are related to the quality features of the software. Relationshipshave been created between these features and qualitative features that help to better define self-adaptive features and to use knowledge of features, metrics and qualitative needs in the development of self-adaptive software [10].
Self-configuration affects several qualities such as maintenance, performance, portability and reuse. Self-optimization has an efficient relationship with performance, as well as the affects performance of the code. Self-protection is also associated with reliability and performance [6].

Failure Types in the Payment Switch

Today, customer judgment in banking affairs is based on the ability of the bank to help solve customer problems and development of its financial affairs. Therefore, in order to reduce the failure and increase the trust of customers, it should identify the bank vulnerabilities. In general, failures can be classified into commercial failures, human failures, and system failures [11, 12]. Various failure patterns are shown in Fig 4. Due to this categorization, the failures that the bank is responsible for and has caused, are divided into two categories: commercial failure and system failure [12].

Commercial failure: failures that occur due to users' errors or trading restrictions on transactionsare termed a commercial failure. These errors are based on ISO-8583 standards [6, 12].

System failure: Some failures that occur in a network infrastructure, switches, centralized connectivity, or connection to other channels, including acceleration (Inter-bank communication network in Iran) and interface switches, are termed a System failure [6, 12].

Note that the knowledge of customers in the field of e-banking in developed countries is more than in developing countries, so the frequency of commercial failures is greater than system failures [6].
Availability is defined as the rate of service time to total time [2]. This is expressed in Equation 1; therefore, availability is a necessary condition for confirmation of failure.

$$A\_svc=T(svc)/(T(svc)+T(un\_svc)) \quad (1)$$

For example, Inter-bank non-servicing is not a matter of being out of service; therefore, if switches availability are more, the given statistics are more valid in terms of switches. Availability is calculated based on export, accepting indicators, and domestic transactions.
The purpose of this study is to increase availability by reducing the failure of services. These failures are identified by the experts and the table of its different states is expressed. The place of failure in system failures is one of the most important principles of coping or healing failures. Some of these points have been introduced in the introduction of the switch and the centralized banking system. After identifying these discovered failures, and based on the states created, a suitable solution for each error will be identified. After the restored components are tested and their operational health is assured, they are returned to the system cycle. Table 4 is the result of the success and failure of the company's transactions during the period of October 7, 2015 toJuly 12, 2017, which includes the frequency and percentage of unsuccessful transactions.

Table 3 shows that only 8.15% of transactions resulted in errors in services, some of which may be a failure. Approximately 3.42% of these transactions have system errors in which specific goals are pursued in these areas because these failures are noticeable and the purpose of this study is first to eliminate errors that may lead to failure

within the bank and sources and transactions management at the time of the crisis of other banks; therefore, using Table 3 and expert knowledge, the table of failure and determining the state of collision and eliminating failure have been prepared. Table 4 shows solutions to deal with the failures provided.

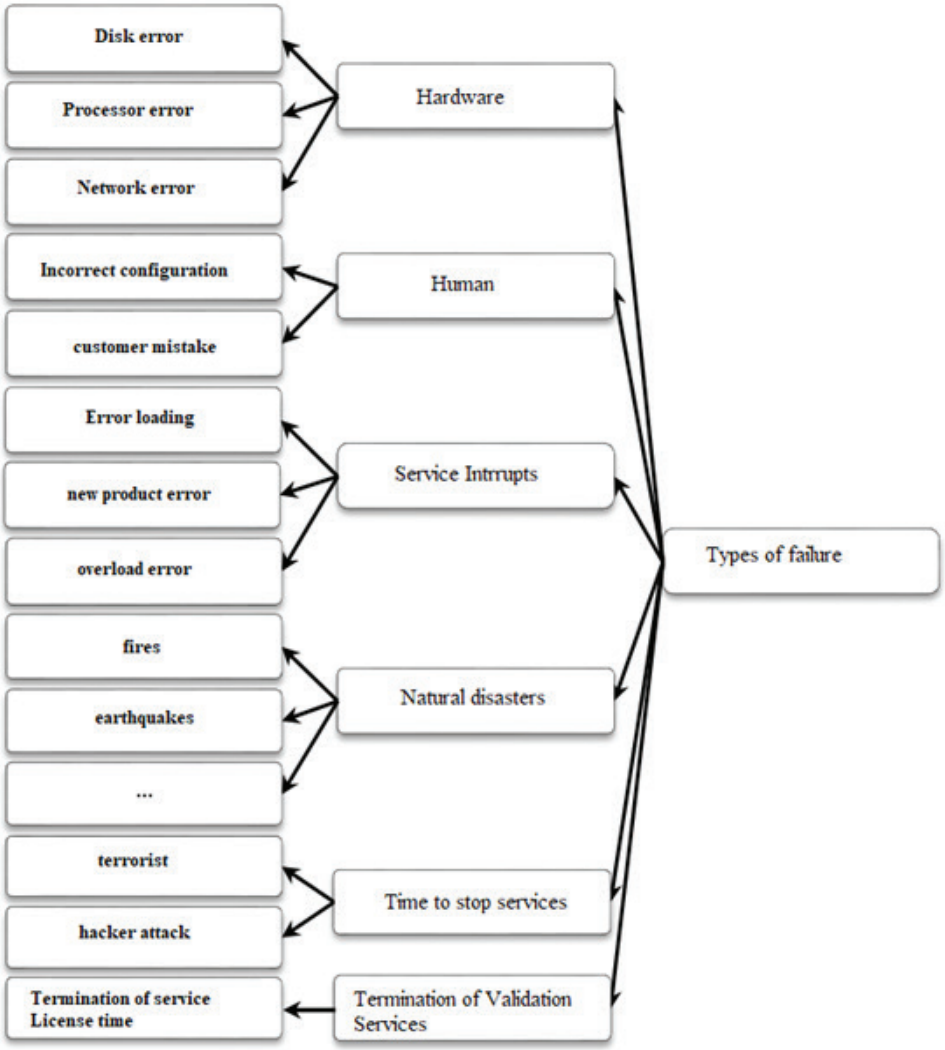


Fig 4. Failure classification in banking systems

Table 3.The success and failure transactions based on failure

Types of transactions	Total percentage	Frequency	The rate of error to total errors %
Successful transactions	91.85	125,427,386	-
Failed transactions	8.15	11,131,653	-
Failed commercial failure-transactions	4.73	6,458,797	58.02
Failed system failuretransactions	3.42	4,672,856	41.98

In the next section, the proposed model and the way to deal with this responses code will be expressed.

The Proposed model (Experimental)

When a system uses a self-healing cycle, the service-oriented architecture has the best feedback between current architectures, Because each system sector is considered as a service, therefore error management and detection are easier than other architectures, such as component-based architecture, or Aspect Oriented Architecture [13], but the electronic banking model is more than half of the Iranian banks based on centralized banking, in which system components are managed centrally according to the importance of security [14]. The Melal credit Institution has used a service oriented architecture because of the use of new technologies and architecture, so in this article, the focus is on self-healing in a serviceoriented architecture.

The other part that should be paid attention to the switch is the accepting switch and the problems that arise in this area [14]. This switch is especially important due to the connection to the acceleration switch. If one of theacceleration members faces a problem in their structure or switch and other banks do not manage their transactions, that bank can hardly escape the problem [7]. Due to frequent sending of the transactions, there is no way to heal or fix the problem. Only if thetransactions will beblocked or reduced by acceleration center, the healing will be possible [7].

Therefore, in this section, two objectives of managing export transactions during an error and managing accepting transactions are acceptable for other banks at the time of the existence of the problem. Fig 5 shows the design pattern for self-healing in the Melal Credit Institution Switch System.

In this model, we have tried to create acoordinator layer to call for a variety of services, manage the services in the internal services sector and have a better management of the self-healing process. This middle layer allows the switch to use the equivalent services and even modify or select other servers. For example, it can be noted that if the bill payment service is not properly accountable, the self-healing component, based on the status table, can receive this service from alternative servers that are already defined for the system.

The model and architecture of the switch must be modified in order to accommodate the self-healing and the mechanisms that are needed in this regard; therefore, using the capabilities of the science of programming, we tried to make and implement these mechanisms. In the first step, in order to change the architecture, equipping switching with more reliable services and decentralizing critical points and using healable solutions can be mentioned. The following section describes each of them. Fig. 6 illustrates the model designed for the switch.

Payment switches have internal processes for managing factors such as settlement, end-of-day operations, and management of reversal queues [6]. When there is a problem with the switch, the healing operations can't be done due to the lack of availability as offline, and must be performed at runtime. Because the healing operations itself also increases the internal processing of the switch. In the first step, internal processing should be reduced to the extent possible. For this, the "Spring" framework has been used as anarchitecture context management framework.

Table 4. An example of failure and healing

Response code	Error descriptions	Solutions for healing
93	Failure to receive customer information from the database	No connection to the database. Restarting the database and connecting with it
96	Failure to receive information from the banking service layer	Restarting banking service and request a reconsideration of the previous request
96	Failure to send information to the bank service	Reducing system load and changing the server
91	Failure to receive information from the bill service	Restarting banking service and request a reconsideration of the previous request
91	Failure to send information to the bill service	Reducing system load and changing the server
06	System error in the service	Reduce system load and change the server
84	Failure to communicate with the service	Changing the server
25	Not receiving the transaction principle	Restarting the wiretappingservices
30	The format of the data sent is incorrect	Rebooting the service and informing
68	No response	Rebooting the wiretappingservices

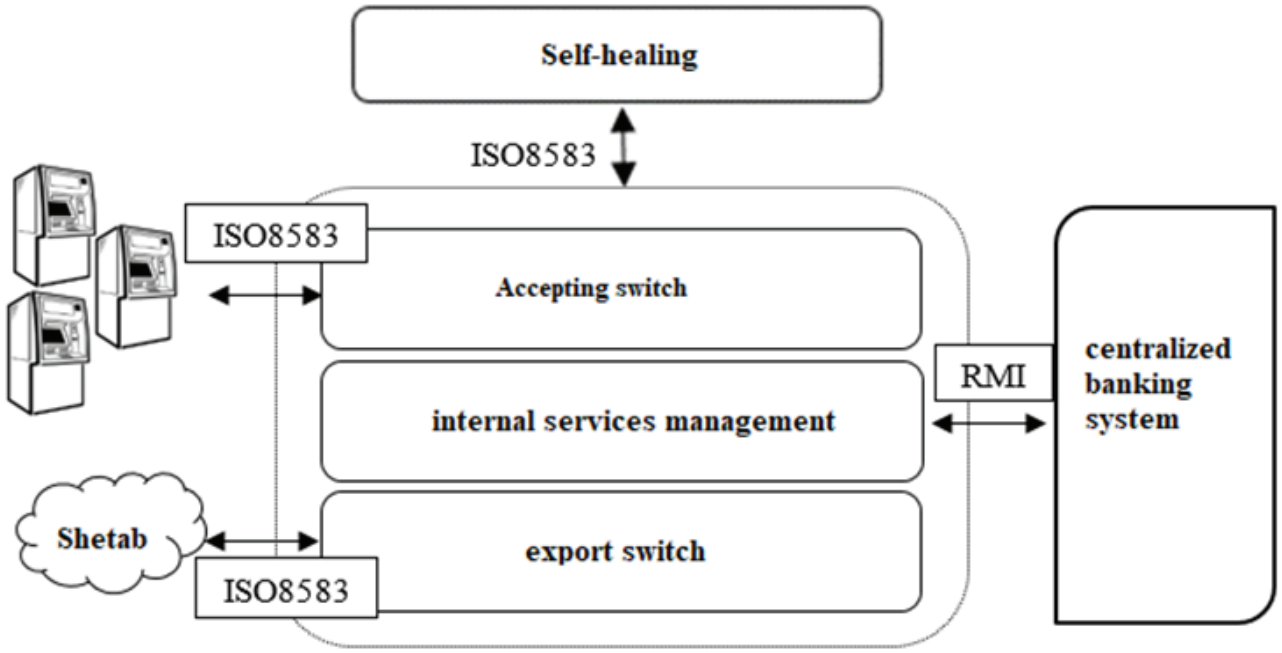


Fig5. Overview of the payment switch after adding the self-healing module

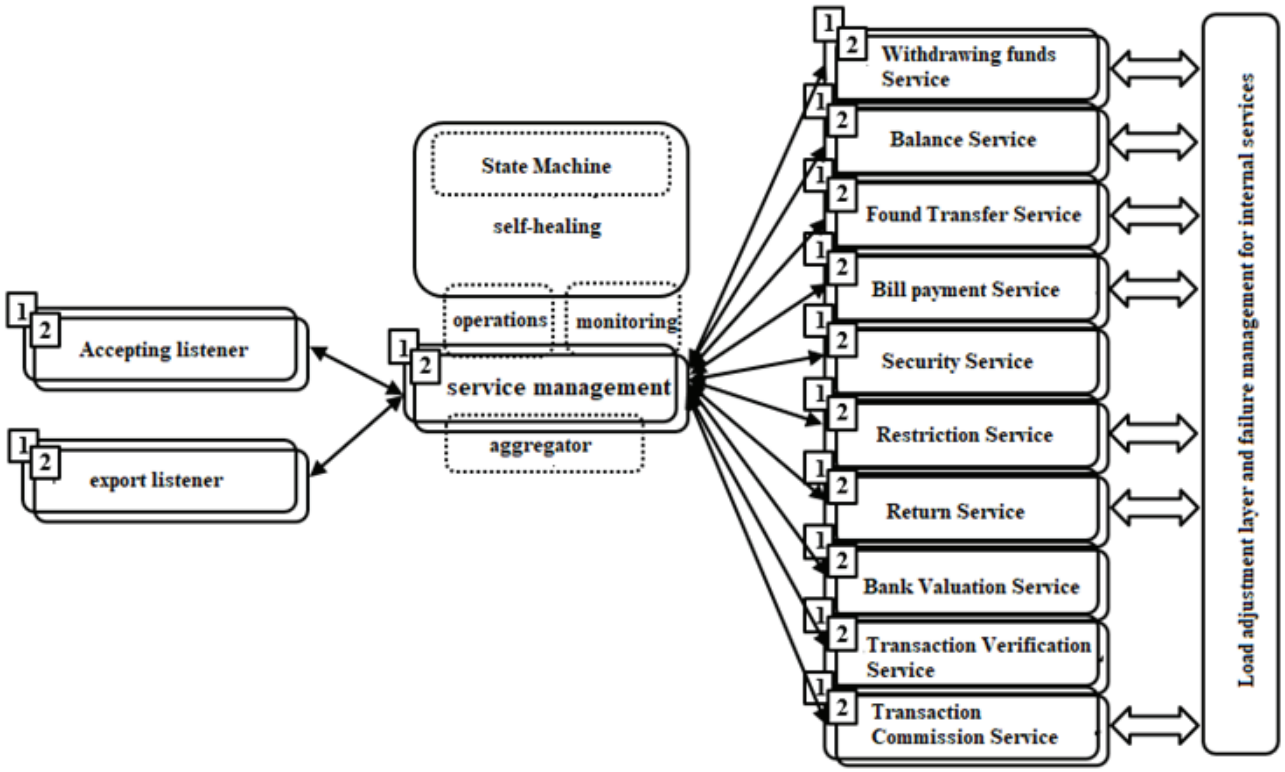


Fig6. Modeling the modules and services available on the payment switch

One of the most important features of this framework is the management of modules and the dependency between them by XML files [15]. This makes it easy for modules to be managed, and it does not allow the programmer to write additional code [15].

The proposed model uses different design patterns, which are used in the component of the "Service Management", state and factory design pattern. Each of the services used in this model has a restart feature and the self-healing component can reboot it if it detects a failure. Another feature of these services is the status feature that shows the current state of each service.

Each service has a backup service to replace the original service according to the decision of self-healing when the original service faces with problem. The reason for the presence of numbers 1 and 2 on the services with this shows this capability. The status of each service will change depending on its current status. The status of each service has "Service" and "out of service" values. In case of outsourcing of any of the services, the backup service will be activated and the original service will be provided in order to heal to the self-healing component.

The process and the accepting transaction cycle is such that when the transaction request is in this mode, the accepting mode, the transaction is received by the relevant listener service and will be redirected to the "service management" central service. If the message passes successfully the security issues, the service will be specified based on the characteristics of the Message type identifier and the process code. Once the service is specified, the turn will be validated by the bank and then the message will be validated. If all these parts are valid, the service will be called up.

In the provided model, the service layer is individually linked to the bank's internal services. If an aggregator component is called from the management service to complete the process, communication or forwarding to acceleration is required. After this, transaction will be sent to the acceleration. The steps for receiving and completing the response will return and send the respond to the requester.

The graphic sample of this process for the balance transaction is shown in Fig.7. Each step is indicated by numbers, respectively. The steps below are explained below.

Getting the requested transaction and checking the transaction template

Request relate totransaction security review requested

Responding to security review request (security verification)

Request for review by the issuing bank

Responding to a request for review by the issuing bank (verified(

Request for transaction type evaluation

Respond to request for transaction type evaluation

Balance request

Sending a transaction to the issuing bank

Receiving a response from the issuing bank

Responding Balance Service

Application for registration of a transaction fee from the commission management service

Application for registration of a fee accounting account

Response to registration fee accounting fee

Service commission response

Responding and sending the balance to the transaction requester.

Self-healing steps on the payment switch

Self-healing has different stages that can be personalized for use in the various concepts of each stage based on the purpose [14]. In this regard, the steps are described in the following statement.

Monitoring: Due to the desired architecture, there are various methods for collecting data for the discovery stage [2, 4, 16]. According to the problem model, the data is gathered from the system and are introduced and then used based on which some criteria; for introducing the criteria, the system properties are determined first and stored in a database called Feature. Then, based on the system requirements and expert knowledge of the individual, the criteria for the detection will be determined and will be considered as system criteria. To use these criteria in the system, it is necessary to define a multi-protocol benchmark. One of these protocols is expressed in Equation 2 and its values are expressed in Equation 3.

{M|M□Metrics AND □ A□Attributes
AND □C□Conditions:(A:C)}
{p|p□Protocols AND □ M□Metrics
AND □ sc□□sys□_svc:(m,{sc})} (2)

Attributes:{a_1,a_2,…a_n }
Conditions:{c_1,c_2,…,c_m }
Metrics:{M_1:{a_1:c_1;a_2:c_2;a_3:c_3 },…,
M_1:{a_2:c_6;a_(n-4):c_4;a_n:c_9}}
□Sys□_svc:{□sc□_1,□sc□_2,…,□sc□_k} (3)

c_1,c_2,…,c_m are conditions that are assigned to some of the features and □Sys□_svcis a list of system services that the criteria should apply to them. Then the protocols are defined as follows:

Protocols: p_1 (m_1,{□sc□_3,□sc□_11,□sc□_(k-1) }), p_2 (m_4,{□sc□_1,□sc□_8,□sc□_k}),p_2 (m_(l-1),{}) (4)

For each protocol, if the system components are set to an empty value, it indicates that simple control conditions should apply to all components of the system. According to the banking system [12], the following elements can be mapped to the following.

Attributes:
Server response time for received transactions
Database response time
Time of response to external services, including purchasing charges and paying bills
The accuracy (authenticity) of the operational efficiency of the banking services
The accuracy of the operation efficiency of the switch services
Calculation and recording transaction commission document
Metrics:
Server response time for sent transactions should not exceed 2 seconds.
The database server response time should not exceed 1 second.
Access to banking services should be possible.
Access to billing and charging services should be possible.
Metrics rules:
(The database response time for packets received should not exceed 1 second: {database})
(The server response time for packetsreceived should not exceed 2 seconds: {Centralized Banking System})
(Access to bill payment services and charge purchase should be possible: {Billing system and charging})
After the definitions are completed, some services will include several methods for using these protocols, called local services [12]. Their task is to collect data from the system and decide which of the data is considered to be a health index for using protocols at a later stage. Local services are implemented based on several plans and organizational procedures at specific courses. For example, some of these services are defined in the Table 5.
Therefore, when the data collection is completed, the data operations recorded are sent to the discovery stage.

Detection (discovery): The data collected from the pre-

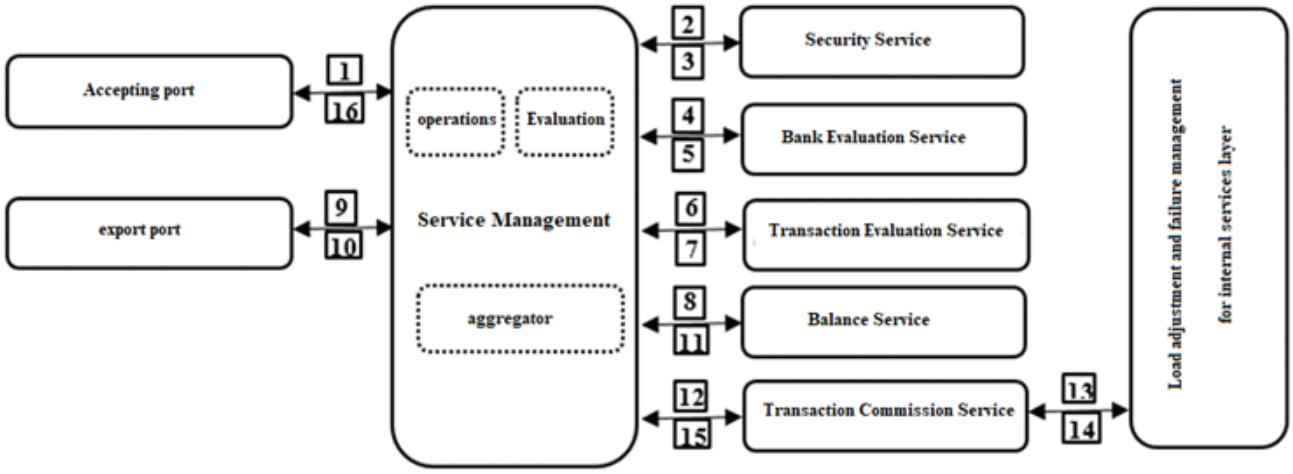


Fig7. The steps to check and respond to a balance transaction in accepting sate

Table 5.Added environmental methods of payment switch

Component name	Method name	Parameter name	Description
Transaction	BalanceInquiry_bkp ()	Request	Receiving account balance
Transaction	fundTransfer _bkp ()	Request	Money Transfer Service
Transaction	cash _bkp ()	Request	Cash withdrawal service
Transaction	GoodAndService_bkp ()	Request	Shopping service
Transaction	BillPayment_bkp ()	Request	Bill payment service
Database	TransactionInquery_bkp ()	Trn_ID	Receiving and reviewing the transaction
Network	Echo ()	Ip port	Checking network connection
Logger	DurationCalculator ()	Transaction	Calculating transaction time
Logger	DurationCalculator ()	Entity	Calculating the time of doing database operations

vious step are compared with certain critical conditions introduced in the state machine. All data is recorded in a database and the results are viewed and their status is checked. In case of occurrence of certain states that lead to failure, the process to be restored will be executed. Considering the scale of the banking system scale, standard protocols must be taken into account; therefore, after discovering system conditions that have conditions like failure, one of the healing methods should be applied to them. The investigation of the failure conditions is based on the conditions of the state machine; however, considering the quantity of discovered states and low importance of some of them, it is necessary to classify them. Received response codes have a switch failure factor. The sum of the various coefficients can be placed in different classes and, based on the specified states or intervals, the healing method will be specified and chosen to the healing operations.

{S|S□States AND □ A,B□Attributes :(C: A, B)} (5)

It should be noted that the weight coefficients of the classes are determined by the exports and the certain failure coefficient is considered for the response codes. Depending on the degrees obtained based on received response codes, the switch failure factor, which is maintained by the general variable called offset, will be measured. In Table 5, the refractive index of each of the response codes is based on the state of exportability and acceptance.
In Table 6, the refractive index of each of the response codes is based on the export or accepting state.
Each of the received response codes causes moving towards failure to a certain amount. If the received response code increases a deviation, it will move towards the more critical states in switch and will be healed based on the healing processes in each case. The response code given in Table 6 will increase the level of health deviation. To reduce the deviation from the health criterion, in addition to the success's responses code, the time parameter is also effective. The time periods elapsed from the last recorded status is controlled by the self-healing component and the five minutes will reduce the refractive index by three units. Reducing deviations is only valid in critical situations. In times of alert, time will not be effective, and only on the basis of the registered response code will be taken at the monitoring stage to reduce the deviation. In this research, the expert has identified different states:

S1: Health state, in the situation where it has not yet reached the failure border. This will continue up to 20. And only notification of the errors will be made.
S2: Alert State 1, the system must manage the healing due to the failure indices. Reducing load and initial healing will begin in this case, so that an index of 40 in this mode will be maintained.
S3: Alert State 2, the system will continue the process of reducing load and healing more seriously, so that an index of 60 this mode will be maintained.
S4: Alert State 1, in this case, the system minimizes internal traffic and the healing will begin with this step, so that an index of 80 in this mode will be maintained.
S5: Alert State 2, in this case, internal traffic will be stopped completely and healing of services with failure will begin.

Healing: After failure detection at the discovery stage, healing turn for failure is detected. According to the previous sections, we can use a state machine to get an acceptable solution and heal the parts. There are various ways to achieve this goal, some of which are listed below. Most self-healing methods are different in using intermediate mechanisms; therefore, different failure recovery techniques are used, which are listed below:

• **Extension Techniques:** Self-healing and Rollback techniques and their categorization, and some agent-based self-assembly mechanisms, which copy components to replace the components with errors and allow us to reconstruct the overall structure [14]. Another method inspired by the biological process allows the system to copy the cells against external penetration. One of the techniques used in Rollback Oriented Calculation, including isolating components with errors and replacing them with extension components [4, 17].

• **Architectural models and procedures (policies):** Some component-based frameworks support interchangeable architectural styles to overcome the deviations of performance [6]. Two examples of this are Rainbow and Madam. Rainbow statically considers a set of rules for each failure state previously specified. Madam uses some useful functions to select an architecture to repair an error [6].

• **Component rebooting:** Components with errors have the ability to reboot independently and automatically prevent propagation error. Wherever the incorrect performance can be observed, it can reboot the component with errors. The effectiveness of this technique actually requires less time to complete it; this means that restarting individual components requires less time than restarting the entire system [5].

There are other Rollbackoperations, including the SOA-based reorganization process, voting methods for the recovery / Byzantine agreement, etc. [12, 18, 19].
But in this study, Rainbow framework and the state machine have been used to recover and repair failures and

errors. Before each operation, an error table, as in Table 7, must be provided, the degree of importance of each one is determined in the state machine, and then the repair solution is determined. After mapping the observed failures to the corresponding states, services with a higher degree of importance will be selected to perform the healing operations. The expert in Table 8 has considered different modes for the payment switch.

Therefore, for online repair, the switch needs to reduce internal load. Reducing the internal traffic load of the switch and changing its operational status is determined by the states stated in 0. These states and conditions are based on expert knowledge and operational experiences associated with the payment switches and in accordance with the degree of failure and based on ISO8583 standard documentation. Table 9 shows how to reduce the load on the switch and repair any of the relevant parts.

Table6. The refractive index of the response codes in the export and accepting states

Response code	Refractive index	
	Export	Accepting
84	10	5
06	10	5
09	5	2
77	10	5
91	5	2
68	5	2
93	3	1
94	3	1
96	3	1
25	2	1

In the monitoring phase of each bank, if the code issues a response that has a refractive index, then it will receive a negative rating corresponding to it and will be recorded in a table called Score. The data stored in the monitoring stage, in addition to determining the degree of accuracy of

Table7.Deviation intervalof health and identifying any cases of it

Deviation interval	State	State name	Description
[0 ... 19]	Normal	S1	Up to the deviation level of 19, only the information charts of the switch authorities will be changed and nothing will happen on the switch itself.
[20 ... 39]	Stage One alert	S2	The first steps to reduce traffic begin to repair.
[40 ... 59]	Stage Two Alert	S3	Reducing traffic increases and healing will begin.
[60 ... 79]	Critical, Stage One	S4	The critical Situation of the first step to minimize internal traffic and initiate relevant repairs
[80 ... 100]	Critical, Stage Two	S5	The critical status of the second stage and the stopping of internal traffic and the start of the corresponding repair

Table 8.States related to the switch and how to reduce the load and repair

state	Interval		Parallel Run	Repair
	Repeat (min)	Run (s)		
S1	1	1	active	
S2	5	5	Inactive	Replacing and restarting specified services at the monitoring and testing stage
S3	5	10	Inactive	Replacing and restarting specified services at the monitoring and testing stage
S4	10	15	Inactive	Withdrawal of financial transactions, and replacement and restarting of specified services at the monitoring and testing stage
S5	15	20	Inactive	Withdrawal of financial transactions and troubled banks, and replacement and restartingof specified services at the monitoring and testing stage

the banks and how it responds to requests, internal switching services will receive a point in the same manner in terms of the failure. This issue can be identified based on the code of responses issued for export transactions. The repair component uses its operators in the service management module and begins the repair operation. The steps are described below.

1. Declaring a Service or Services with Failure to Repair Operating Component
2. Declaring Transactions with Failure to Repair Operating Component
3. Declaring banks with Failure to Repair Operating Component
4. Reducing the internal traffic load of the switch
5. Notification of the repair condition for services with bug
6. Replacing services with bug with their backup
7. Restarting services with bug
8. Transaction Test by Test Transactions Considered

All repair steps will be recorded and, considering that all self-healing steps are managed by the self-healing component, the experience gained in each repair will be considered for future states. This is called the service refractive index. As an example, in the past three repairs, the internal transfer service has been involved; therefore, in addition to informing the system administrator, this service will be prioritized if the service continues to fail with another service or other services.

After receiving each response code from each service, the general offset variable will be changed. In this section, it will be clear to what extent each service has affected this variable. After entering the value of this variable into any of the intervals or states stated in Table 9, the services that have the most impact will be the highest with the least repair priority. Once the service is selected for repair, the amount of service's impact from the variable will be deducted and the service will be sent to the repair stage.If there is no solution to the diagnostic failure, the system can report it to a specialist. Then this error will be added to the error table and if it happens later, the system can manage it.

Test: After performing recovery operations on the components that resulted in failure, a test scenario is used to ensure that there is no negative impact on other components [16]. If the test result is good, the recovered component is returned to the system; otherwise, the system should notify the backup. With this framework and method, the system can perform recovery operations for failures leading to failure; there is no need for expert personnel for failures that can be retrieved using self-healing cycles.

After healing the service, it should be tested and checked for accuracy. At this stage, a test transaction is used to test. For each service, a sample of the transaction test is considered. These transactions are known for the switch and will have no financial impact on bank documents. Test transactions are separate for each service and have service tasks specific to that service, depending on the type of service. The other names of these transactions are echo transactions [6].

If the service test is successful, that service will return to the operating cycle of the switch, otherwise the system administrator will be notified. Internal switch processes, depending on the received responses code, can also be returned to normal state, and return operations to the "S1" state or normal will be transmitted mechanically. By entering the value of the offset variable to the normal state boundary, all switch operations continue to operate with complete operational power.

Results and discussion

The self-healing results based on the response code received in accepting state are described completely in Table 10. Based on the results, it can be concluded that "How does the self-healing process in the switch increase the availability of bank services?"

The highest healing rate for transactions with the response code of 84 was 27.38%, and the lowest was the response code 94, which was unchanged zero. The results of self-healing in the export state are also given in Table 10.

As indicated, the highest healing rate in the export mode is related to the response code 68 with 64.88%, and the lowest amount for the response code 94 and 96 is zero. In general, considering the number of issued errors before and after self-healing, it can be concluded that system er-

Table9.Results of responses code in accepting state before and after self-healing component

Code	After healing (percent)	Before healing (percent)	Healing(percent)
6	0.06	0.29	0.24
9	0.36	0.82	0.47
12	2.75	11.32	8.57
25	2.11	8.53	6.42
30	0.21	0.59	0.38
68	0.63	11.82	11.19
77	0.52	2.50	1.98
84	1.59	28.97	27.38
90	1.16	4.82	3.66
91	4.85	21.44	16.59
93	1.11	8.22	7.11
94	0.06	0.06	0.00
96	0.42	0.61	0.19

Table10. Results of responses code in export state before and after self-healing component

Code	After healing (percent)	Before healing (percent)	Healing(percent)
6	0.63	8.41	7.77
9	1.93	14.11	12.18
12	1.56	2.90	1.34
25	0.26	12.38	12.11
30	0.00	0.00	0.00
68	3.41	68.29	64.88
77	0.00	0.00	0.00
84	0.27	7.47	7.21
90	0.03	0.05	0.02
91	0.23	0.53	0.30
93	5.60	15.32	9.72
94	0.00	0.00	0.00
96	0.00	0.00	0.00

rors ranged from 3.42% to 0.44%. The number of system faults before the self-healing switch is 4672856 transactions, which is reducedto605917 after a self-healing. Fig. 8 shows the number of transactions.

Imagine removing the error used by internal systems is used by external systems, so failures from both sides are reduced, and a banking system with higher availability can be achieved. The comparison of the number of errors in the banking system is based on the error table and based on the new architecture used in the payment switch with the self-healing component. Table 11 shows the degree of system error reduction and shows that availability has increased by 2.98%.

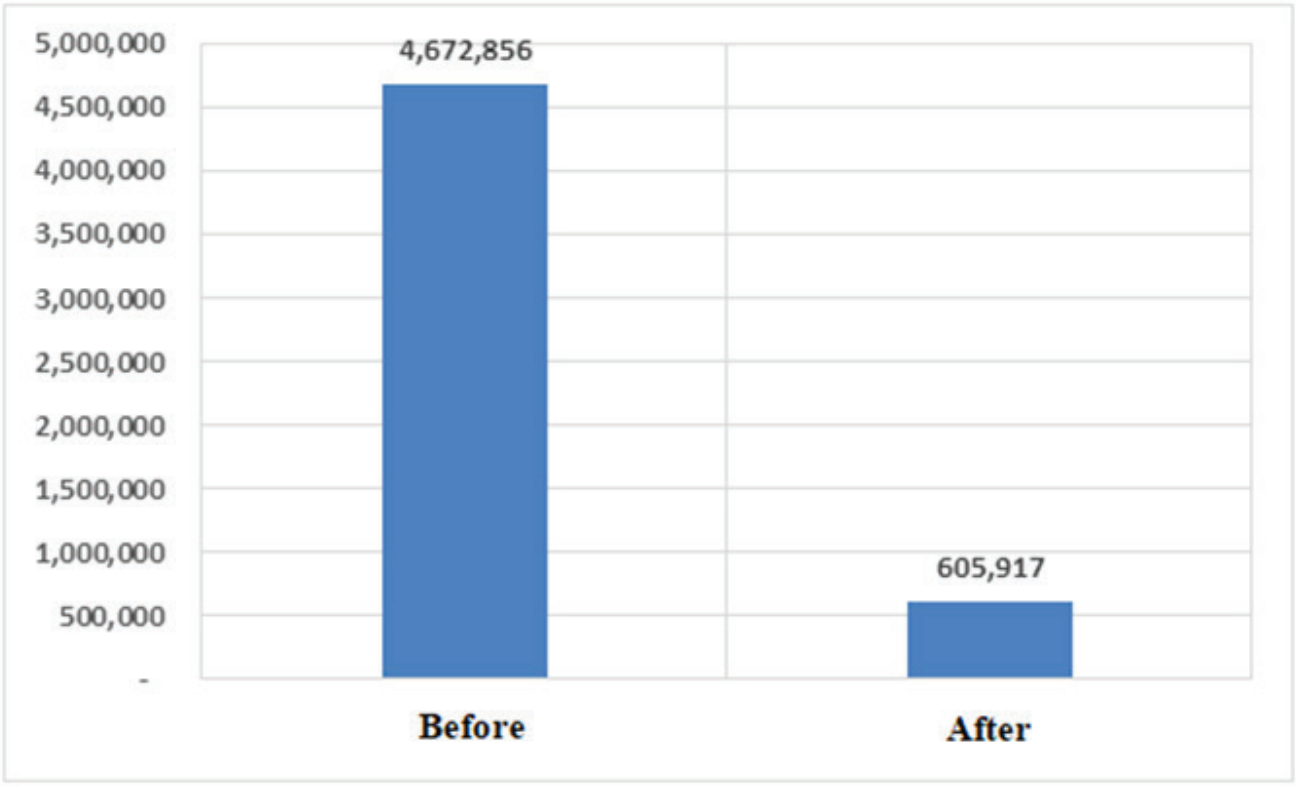


Fig8. The number of issued errors before and after self-healing

Table11.The status of transaction types before and after self-healing

Transaction Status	After the self-healing		Before the self-healing	
	Frequency	Percent	Frequency	Percent
Successful transaction	12 9 4 94, 3 25	9 4.83	125, 427 386	91.85
Unsuccessful system transactions	6 05.917	0 0.44	4, 672,856	3.42
Unsuccessful commercialtransactions	6 , 458 , 797	4.73	6, 458,797	4.73
Total	136 , 559 , 039	100.00	136, 559 039	100

If more factors are taken into account, the number of discovered and repaired failures will continue to change. For example, a different error sequence or different states for a state machine results in failure, and only some of them are discovered and used as a criterion at the discovery stage; therefore, other failures should be discovered by expert individuals and methods of pattern recognition, and added to set of criteria and the states; therefore, in view of these existing states, a coefficient must be defined that takes into account the number of self-healed failures of the variable. Relationship (5.1) indicates how therelationship between number of failures before and after self-healing.

$F_b-\alpha \leq F_h \leq F_b-\beta$ where $\alpha \leq \beta$ (6)

Fb: Number of failures before using self-healing cycle

Fh: Number of failures before using self-healing cycle

As a result, despite the existence of intelligent systems and its good performance in reducing system failures, the need for expert personnel remains tangible.

Conclusion

In this paper novel approach based on self-healing technique using state machine concept is introduced. It is defined for increasing the availability of Payment Switches in core banking systems that is considered as one of the significant issues in large scale payment systems. Also respective types of failures are analyzed and classified.

Using of self-adaptive technique leads to have more reliable and available environment. Besides autonomous agents can do the appropriate react in case of any incon-

sistency or alarm sensed by the system sensors. It is exposed that this approach degrades the failure percentage of payment switch sensibly and in performance view, drastic optimization can be seen,

State machine mechanism also enhances the power of self-healing approach in controlling the conditions and reconfiguration of the parameters according to the knowledge gathered from the relevant expertise.

Another point of view is security and indeed one of the most critical challenges for any Bank customer is this topic. Using this approach improves the quality of security by sensing and doing the respected acceptable reaction to the variety of events in each state including normal, warned, critical so on.

Using voting system in self-healing techniques, more complicated statistical reward and penal functions also expanding this approach in orchestration of ESB in large and ultra large systems is some clues to ameliorate this approach as future work.

References

Herbst, N.R., et al., Self-adaptive workload classification and forecasting for proactive resource provisioning. Concurrency and computation: practice and experience, 2014. 26(12): p. 2053-2078.

Peng, X., et al., Self-tuning of software systems through dynamic quality tradeoff and value-based feedback control loop. Journal of Systems and Software, 2012. 85(12): p. 2707-2719.

Stoicescu, Miruna, Jean-Charles Fabre, and Matthieu Roy. "Architecting resilient computing systems: A component-based approach for adaptive fault tolerance." Journal of Systems Architecture 73 (2017): 6-16.

Haneef, F. and S. Angalaeswari, Self-Healing Framework for Distribution Systems. 2016.

Schiefer, P., R. McWilliam, and A. Purvis, Self-healing Fuel Pump Controller Mapped into Memory Based Finite State Machine. Procedia CIRP, 2014. 22: p. 132-137.

Nazemi, E., T. Talebi, and H. Elyasi, Self-Healing Mechanism for Reliable Architecture with Focus on Failure Detection. International Journal of Information Engineering and Electronic Business, 2015. 7(3): p. 32.

Harker, P.T., The Global Interdependence Center's Payment Systems in the Internet Age Conference. 2017.

HASANZADEH, A., Efficiency and its determinants in the Iranian banking system. 2010.

Sharan, K., Java remote method invocation, in Beginning Java 8 APIs, Extensions and Libraries. 2014, Springer. p. 525-548.

Okon, S. and P. Asagba, Deploying Self-Organizing-Healing Techniques for Software Development of Iterative Linear Solver. 2015.

Peng, X., et al., Self-tuning of software systems through dynamic quality tradeoff and value-based feedback control loop. Journal of Systems and Software, 2012. 85(12): p. 2707-2719.

Begley, T.A., A. Purnanandam, and K. Zheng, The Strategic Underreporting of Bank Risk. The Review of Financial Studies, 2017: p. hhx036.

Raz, O., P. Koopman, and M. Shaw. Enabling automatic adaptation in systems with under-specified elements. in Proceedings of the first workshop on Self-healing systems. 2012. ACM.

Kaur, P., A.K.G. Verma, and R.G. Kumar, Comparison and Analysis of Service Oriented Architecture. 2017.

Guo, Yujie, Ming Chen, and Kanglin Wei. "Research of Recycling Resource Website Based on Spring and MyBatis Framework." Information Technology and Intelligent Transportation Systems. Springer International Publishing, 2017. 307-314.

Chrysoulas, C. and M. Fasli, Towards an adaptive SOA-based QoS & Demand-Response Provisioning Architecture for the Smart Grid. 2017.

Albassam, E., H. Gomaa, and D.A. Menascé. Model-based Recovery Connectors for Self-adaptation and Self-healing. in ICSOFT-EA. 2016.

Provencher, Samuel D., and Ashok R. Subramanian. "Method and system for stateful recovery and self-healing." U.S. Patent No. 9,569,480. 14 Feb. 2017.

Iftikhar, Usman, and Danny Weyns. "ActivForms: A Runtime Environment for Architecture-Based Adaptation with Guarantees." (2017).