International Journal of Engineering & Technology, 7 (2.26) (2018) 95-101

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper



Active Database System Approach and Rule Based in the Development of Academic Information System

M. Miftakul Amin[1], Andino Maseleno[2*], Shankar K.[3], Eswaran Perumal[4], R.M. Vidhyavathi[5], Lakshmanaprabu SK[6]

[1]Politeknik Negeri Sriwijaya Palembang, Sumatera Selatan, Indonesia
[2]STMIK Pringsewu, Lampung, Indonesia
[3]Assistant Professor, School of Computing, Kalasalingam Academy of Research and Education, Tamilnadu, India.
[4]Assistant Professor, Department of Computer Applications, Alagappa University, Tamilnadu, India
[5]Assistant Professor, Department of Bioinformatics, Alagappa University, Tamilnadu, India.
[6]Research Scholar, Department of Electronics and Instrumentation Engineering, BS Abdur Rahman Crescent Institute of science and Technology, Chennai, India.
**Corresponding author E-mail:andimaseleno@gmail.com*

Abstract

Active database system is a database system which is capable of generating a certain action automatically if it detects an event that meets certain conditions. The existence of Event Condition Action (ECA) rules and functional components such as triggers, stored procedures, and stored functions that are owned by active database system make the database system has the ability to automatically monitor input and output data. Separation of ECA rules components in the database with the application program will also facilitate the development of information systems. This research applies active database system in academic information system, so that academic business rules can be planted in database software and be able to produce the right solution automatically. The addition of active database system components to the database software makes procedures such as subject distribution, study plans, academic leave, values, thesis defense and other processes can be monitored by the system automatically. This can be done because between applications and databases use the model driven approach parameters to communicate with each other. The results of this research prove that a database is not only has function as a container of data, but can control the information system actively, this is caused by the logic of programs that are generally planted in the application layer that can be moved and planted in the database software.

Keywords: *active database system, ECA rules, trigger*

1. Introduction

Along with the increase in needs and requests of users in the speed of data processing and ease in data management at an institution, then the current technological developments must also be able to meet those needs. This is not regardless to the ability of the database as a container and data processor which is expected to store valid data. Even today the need for data and information is not only stored and accessed within the scope of an organization, but can be exchanged between organizations despite using heterogeneous platforms with Service Oriented Architecture (SOA) technology [1].

In an information system, user's needs and demands determine business rules that will be managed its data, in this study the user needs to be focused on business rules to process academic data in Higher Education.

The current academic information system is not equipped with business rules that explicitly can be traced by the system developer (programmer), because all business rules are fully embedded in the program code in the application layer, so it needs to be made an information system development model by first modeling business rules in the representation of knowledge. This is related to the condition where if at any time the academic information system is experiencing changes in business rules. Academic data processing can be processed using Active Database System which is expected to provide accurate information processing solution of the changes that occur.

Database system that has been used is a passive database because the data and information is very dependent with the program code in the application layer, in the sense that the database only serves as a container of data without being able to do anything. Therefore it can use an alternative Active Database System which is a database system capable of generating a specific action based on the conditions that occur in accordance with the constraint created and not completely dependent on the program code in the application layer in executing the command.

So most of the processing logic is in the database compared to the application program layer. In Active Database known term ECA (Event Condition Action) Rule, Stored Procedure, trigger which is a component of Active Database system builder.

2. Related Works

Research that was conducted by Kurds [2] uses a client/ server computing model so that the system can run on several client computers. On the server side using MS database. SQL Server 2000 by using the ActiveX Data Object (ADO) data access method. The built system is capable of converting data from the old

Copyright © 2018 M. Miftakul Amin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

system (DBASE) to the new system (MS SQL Server 2000). Thus the data can be integrated on one database system so as to facilitate the maintenance and management of data.

Efendi [3] conducted the research about Academic Information System in terms of Human and Computer Interaction aspects. This research build the analysis on the questionnaire data collected from the respondents which consist of lecturers, administrators, colleges and students leaders. This research gives conclusion that need to be pursued an Academic Information System which user friendly and interesting so user do not feel saturated. This research also gives recommendation to develop an application of web-based academic information system. Fachrurrozi [4] examined the improvement of the functionality of an academic information system by restructuring the data in the database design as well as the DBMS environment used. Data restructuring is done with several stages, ie analysis on the old system, data reengineering, implementation of the results obtained by moving data existing in the old system into the new system, as well as testing the data that has been moved to the new DBMS environment.

Paton [5] had implemented a case of using Active Database Management System in processing a portfolio database. The application has been able to implement ADBMS in monitoring transactions and generating reports periodically. In this research are given many examples of rules and how the script creation rule is built, described also about designing the rule and analyze it. Lanova [6] used Active Database as a constraint in conducting scheduling. Lanova research utilizes many triggers and stored procedures as implementations of constraints implanted in the Database Management System. The implemented Active Database has resulted in a correct and accurate schedule solution.

One important feature of Active Database System is the rule scheduler. Evaluation of the rule scheduler can be done with 5 criteria such as average response time, response time variance, throughput, time overhead per transaction, and CPU utilization [7]. To measure the performance of active database system can also be done by using buffer management [8]. The research by Guo [9] has undertaken an analysis of the shortcomings of the traditional administrative education management system, which further proposes an educational administration management system using an active database embedded in the program. Research shows that in real-time active database service system is efficient and feasible to be implemented.

The application of rule-based that is also used in the field of information system and software development is research conducted by Ephzibah [10]. This research used rule-based and some algorithms in machine learning to build a disease diagnosis framework combined with Big Data management process. This framework can serve as a reference for medical doctors to assist in diagnosing illness.

3. Active Database System

Active Database Management System (ADBMS) is essentially a passive, conventional database, with the possibility to behave reactively. The addition of the reactive functionality is characterized by an ECA-rules (event-condition-action rules) defined by an event, a check of conditions, and if true, an action is executed [11]. Once a group of rules is defined, an Active Database Management System will monitor the event.

The frequently used Active Database Architecture is included in a layered architecture system where all Active Database components are located on a conventional database. In this system architecture, the conventional database can be converted into Active Database without the need to modify the Passive Database as a whole. The shape of the Passive Database architecture is shown in Figure 1.

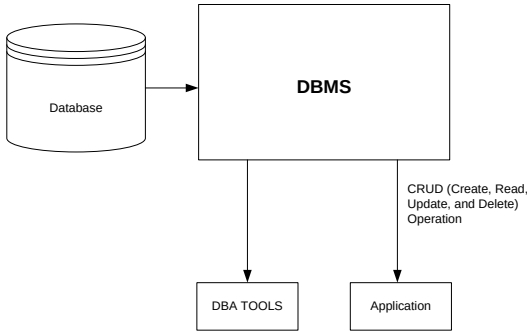


Fig. 1: Passive Database Architecture

In the Passive Database architecture in Figure 1 it is shown that the database serves only as a data container. Program logic is fully embedded in the program or application layer. While the Active Database not only serves as a container of data, but also has additional functionality that can be used to replace the program logic into the database software. The shape of the Active Database architecture is shown in Figure 2.

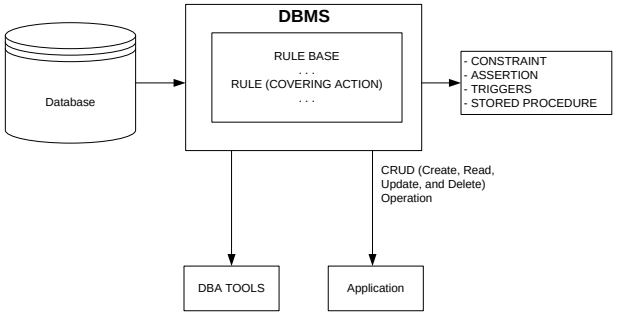


Fig. 2: Active Database Architecture

Figure 2 shows that to implement an Active Database in building the information system required additional functionality of the database software in the form of constraints, assertions, triggers, stored procedures to adjust the features provided by the database software. The functionality is used to replace the logic and program algorithms generally grown in the application layer moved into the database software.

An information system usually consists of data management and presentation (end user). When the information system implements business rules in the development of information systems, there are three layers of data management, presentation (end user) and the layer of business rules (rule layer). These business rules are stored in a repository in the database software [12].

When developing an information system on the basis of business rules, physically the business rules are stored in a repository or data storage that can be managed and changed at any time using the available software. There are 2 approaches that can be used in developing information systems [12]. First, the rules are stored in a database software. These rules are identified by a number of attributes or values in the form of parameters that are able to communicate with front-end applications that use rules. The application will send the values of parameters that have been defined in the rules stored in the database. So through these parameters the application and database communicate with each other to exchange data and information. Second, Independent Process Driven Approach. In this approach, the rules follow the traditional system of information systems development. Rules are translated into the program code in the app directly. These rules are not stored in the database, but in a separate layer, so they can be accessed any time when the system requires the execution of those rules.

An active database management system (ADBMS) is a conventional database system which reacting to multiple events occurring on both, database and beyond it. This can be done with the ECA (Event-Condition-Action) Rules. ECA Rules can be implemented

using either trigger or active rule [13]. After a series of rules has been determined, an ADBMS monitors the relevant event and then notifies the component responsible for executing a rule. A condition is evaluated, if met then ADBMS will execute an action [14]. An Active Database Management System (ADBMS) is an event driven system, where operations such as data changes produce events that can be monitored by active rules [15]. Based on the event, an ADBMS is categorized into 2 groups: simple and complex events [16]. Examples of simple events are basic database operations such as INSERT, UPDATE, DELETE, and time events. Complex events consist of one or more events associated with logical operators. Examples of complex events like REPEAT, SEQUENCE or NEGATION.

4. Rule Based

Methods and algorithms in the field of artificial intelligence (AI) is currently widely used in the field of software engineering and development of information systems. Rule-based systems (RBSs) made from research in the field of AI which in its development can be implemented in information system specially in the database field information system [17]. Another AI method that also used in the field of information systems is a study conducted by Sharm [18] that used a neuro fuzzy system to measure the estimation and reliability of component-based software systems (CBSSs). In the field of software testing, AI methods are also used such as the Fault Detection Technique using Learn To Rank (LTR) [19]. While in the field of project management, research conducted by Kermani [20] has implemented fuzzy logic to control the cost and time in project risk management.

There are two approaches in applying rule-based systems in the database management system, ie deductive database systems and active database systems. Rules in the active database management system can be used as complement to the programming language source code used in the application, and packaged in modules and functions [21]. The rules play a dynamic role to handle various events and conditions that occur during program execution.

Rules provide a formal way to represent recommendations, directions, or strategies. Production rules are written in if-then (if-then) form. The if-then rule connects the antecedents with the consequences it causes. A variety of if-then rule structures that connect objects or attributes as follows:

```
IF premise THEN conclusion
IF input THEN output
IF condition THEN action
IF antecedent THEN consequent
IF data THEN results
IF action THEN purpose
IF action THEN reaction
```

The premise refers to facts that must be true before certain conclusions are obtained. Input refers to data that must be available before output can be obtained. Conditions that refer to the circumstances that must apply before the action can be taken. Antecedents refer to situations that occur before the consequences can be observed. Data refers to the information that should be available so that a result can be obtained. Action refers to activities to be performed before results can be expected. Action refers to activities that cause the effects of such actions to occur. Because it refers to certain circumstances that cause a certain effect. Symptoms refer to circumstances that cause certain damage or circumstances that lead to the examination.

5. Implementation

From the results of research that had been done, obtained an academic information system in which there was database software, where the business rules were planted using the concept of active database system. Academic information system created in this study was a case example of the application of active database system. The software used in making this program using MS. Visual Basic 6.0 and MS database management system. SQL Server 2000. Some rules were embedded in the database layer of management systems that were implemented in the form of stored procedures, stored functions and triggers that pass on databases and tables. In this section we will present some implementation results from the rule-based academic information system using the active database system, since the developed information system was quite complex and involves the input/output form.

5.1 Main Menu

The main menu is the integration of all services within the rules-based academic information system. The concept built in the main menu used the Multiple Document Interface (MDI) model with the goal of more than one form that can be run simultaneously in the main menu. At the top of the menu there is a pool down menu that can be used to select a particular operation to be run. In the middle there is a section used to display the form or report that is being run. On the left side there is a sidebar menu for user information, while at the bottom there is a toolbar menu to tell the user date and time information. The main system menu view is shown in Figure 3.

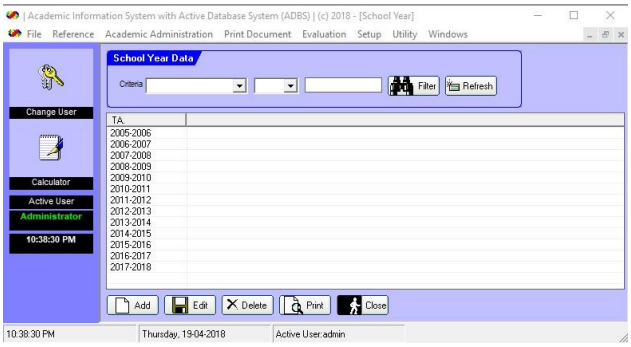


Fig. 3: System Main Menu

5.2 Course Distribution

At the beginning of each semester, the courses offered are distributed with class, faculty, day, hour, and classroom. Subjects that have been distributed as a reference by students to take the study plan card. This lecture distribution menu has rules that are planted in the table of subject distribution in the form of triggers. Scripts that are planted in triggers follow rules that have been represented using the rules of production (rule) as follows:

```
R1: IF course is distributed
THEN set the course
AND specify academic year
AND specify academic semester
AND specify the class
AND specify the course
AND specify lecturer
AND specify class quota
AND define the day
AND specify lecture sessions
```

The display of the course distribution form can be seen as in Figure 4.

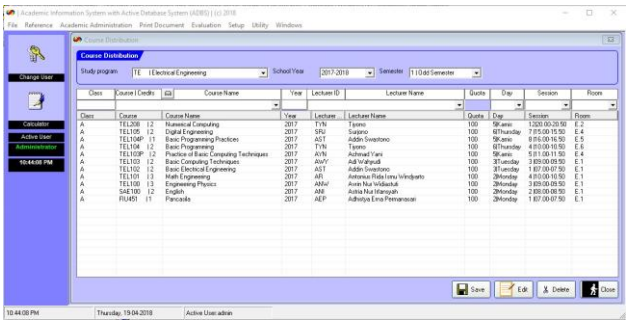


Fig. 4: Course Distribution Menu

If the user does not fill in the class while saving the course distribution data, it will show confirmation from the system like Figure 5.

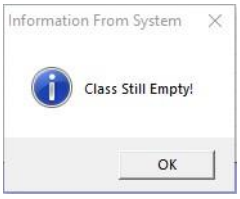


Fig. 5: Error message from Trigger Active Database System

The system confirmation in Figure 5 is not triggered by the script planted in the application, but the result of a database trigger that has been planted in a table conditioned before the data entry process is executed. Confirmation of the system will cancel the process of storing data into the table. The system confirmation as in Figure 5 will appear if any rules specified in R1 are not met. For example there are unallocated day data, unfilled semester data, and other data that have been specified in R1 rules.

5.3 Student Mark Entry

After the lecture in a semester is completed, the score given by the lecturer can be entered into the system. The process of incorporating this score is based on the course data taken by the students in accordance with the planned study. Each mark is filled, then the weight of the mark will change automatically. This weight change is triggered by a trigger that is planted in the table that will be activated when there is a column of data-modified marks. For example, this mark-weighting process matches the rules defined in R5 rules as follows:

```
R5: IF there is a mark-weighted calculation
command
THEN
IF mark = 'A' THEN weighted = 4
IF mark = 'B' THEN weighted = 3
IF mark = 'C' THEN weighted = 2
IF mark = 'D' THEN weighted = 1
IF mark = 'E' THEN weighted = 0
IF mark = 'K' THEN weighted are not calculated
```

Rule that is planted active database system, implemented in the form of trigger and activated if the mark column is changed. The contents of the trigger are:

```
ALTER TRIGGER au_tkrs
ON dbo.tkrs
AFTER UPDATE
US
DECLARE @vid numeric (30.0), @ mark char (2)
SELECT
@ vid = INS.id,
@ mark = INS.mark
FROM INSERTED INS
IF UPDATE (mark)
UPDATE tkrs
SET weights = dbo.getWeighted (@values)
                WHERE id = @ vid
```

Furthermore, the process of weighting involves calling the stored function dbo.getWeighted (@value) with the parameter of the string value.

```
ALTER FUNCTION getWeighted
(
@ arg1 AS char (1)
)
RETURNS decimal (18,0)
US
BEGIN
RETURN CASE
WHEN @ arg1 = 'A' THEN @weightedA
WHEN @ arg1 = 'B' THEN @weightedB
WHEN @ arg1 = 'C' THEN @weightedC
WHEN @ arg1 = 'D' THEN @weightedD
WHEN @ arg1 = 'E' THEN @weightedE
WHEN @ arg1 = 'K' THEN 0.00
ELSE NULL
END
END
```

If the mark column in the table is changed, then the trigger that is planted in the table will be activated according to the rule on R5 as shown in Figure 6.

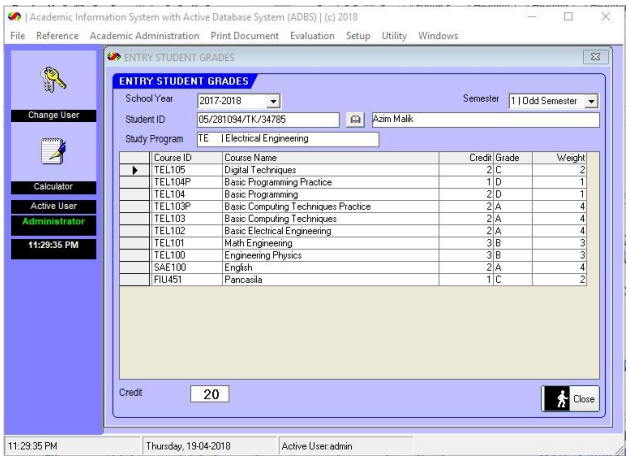


Fig. 6: Student Mark Entry Process

6. Results and Discussion

From the results of research that has been done, obtained an academic information system in which there is software database where the business rules are planted using the concept of active database system. Academic information system created in this study is a sample case of the active database system application. The software used in making this program is Visual Basic 6.0 and MS. SQL Server 2000.

Comparison of systems that use passive database with system using active database can be seen in Table 1.

Table 1: Comparison of passive database system and active database system

Comparison	Passive Database	Active Database
Application layer	Consist of 2 layers, namely data management and presentation.	Consist of 3 layers, namely data management, rules layer and presentation layer.

Business process change	Rediscover the program code in the application, change the corresponding rules and reinstall the application.	Simply change the appropriate rules in the database software.
Script Program	The entire program code is stored in the application.	Some of the program code is stored in the database software, so inside the application the program code is simpler.
Service	Complex and complete	Limited to rules embedded in database software.
Programming language migration	All business rules are rewritten using a new programming language.	Some of the rules embedded in database software can still be used using a new programming language.
Data processing	Data processing depends entirely on the application.	Database software can perform certain action.

The criteria used in this test is the testing of business rules that have been planted in the academic information system using triggers, stored procedures and stored functions that become Active Database System components, then tested using test data (academic data) related to the business rules. In Table 2 we can see some of the tests that have been done on the built system.

Table 2: Result of Active Database testing and analysis

No.	Business Process	Testing and Analysis
1.	Course distribution	Including some data distribution of the course into the system, if there is incomplete data at the time of filling, then the active database gives information to the application that the information is filled incomplete and the data entry process is canceled by active database. At the time the data is entered into the database, then there is an active trigger and check whether the data must be entered incomplete. If the data entered is complete and valid, then the data will be recorded in the database, otherwise the error message is raised and the data recording process is canceled. The following figure provides information about data validation triggered by the trigger on the subject distribution process if at the time of data class filling does not exist.
2.	Credit filling	Credit filling process is made in two models, namely charging credit for new students and charging credit for old students. Charging credit for new students using a stored procedure *sp_kelaskan_mhs_baru*. By using stored procedure that has been successfully done the process of placement of new students in a particular class, and making credit for the new student automatically. While the credit process of old students has been able to check the amount of credit taken in the relevant semester based on the achievement of the previous semester evaluation. When credit data is recorded into the database, a trigger will be activated, then check the number of credits entered and compared to credis that can be loaded, so credits entered into credit (*KRS* system) do not exceed the specified credits. In the trigger there is also a functionality to check the subject prerequisite data, if the course is entered into credit while the course has not been taken yet it will be given a warning by the active database that the course has a prerequisite
3.	Assessment Evaluation	This evaluation is planted in the trigger table credit and will be activated when a value of quality letter is inserted into the system. Active database automatically generates quality weights for each quality letter grade entered into the system according to the weights specified in the rules.
4.	Study results evaluation	The process of evaluating the results of this study is carried out by stored functions that are planted in an active database. Stored function has been able to generate semester achievement index (*IPS*) and cumulative grade point (GPA) of each student. The calling of this function is sufficient to include the specified parameters.
5.	Repetition of Educational Activities	To display the list of values by taking the repeat course taken only the best value, then used the view feature to display the list of courses. With this view then the system can display the list of courses with the best value.
6.	Repetition of Educational Activities	Filling data into educational leave can be handled using triggers that are planted in the leave table. If the terms of the leave requirements such as having passed the first two years of evaluation, the GPA is more than 2, and has accumulated at least 30 credits which are not fulfilled by the student data entered, the active database is able to monitor the data to be entered into the database. A message is displayed in accordance with data conditions that do not fulfil the specified rules.
7.	Rule parameters change	Some rules embedded in the database software have parameters that can be changed. The value changes of the parameter can be performed from the application program and simultaneously affect the actions performed by the rules within the database software.

From some test results it can be seen that the academic information system by using Active Database is able to monitor out and the entry of data in academic information systems, if the input of the user who is initializing the system has not violated the rules.

6.1 Driven Approach Parameter in Trigger

Trigger is a program that is stored in the table and will be executed when the process of manipulating data in the table, such as addition, conversion and deletion of data. When a data set is entered into a record in a table, the active database system has a temporary table named INSERTED that will hold the data entered into the table. All data entered into the table can be identified by <inserted> table name. <Column name>.

6.2 Driven Approach Parameter in Stored Procedure

Stored procedure is another feature provided by active database system. Stored procedure can be directly called from the front-end application, so to run a stored procedure, only applications that support the use of stored procedures that can run it. In the academ-

ic information system there is a stored procedure to conduct student evaluation of the achievement of the study during the first 2 years. The stored procedure script is stored in an active database with the name *sp_evaluasi_2tahun*. With parameters in the form of study program code and force.

6.3 Driven Approach Parameter in Stored Function

Stored function serves to generate return values that can later be used to process important information in the system. As with stored procedures, stored functions can also have parameters that must be filled when calling them, otherwise the stored function can also return a value that can not be found in a stored procedure. With this stored function, the front-end application can call the functions stored in the active database system without the need to re-create it. In the rules that are planted in the active database system, there is a stored function to get the credit quota that can be taken at the time of registering CREDIT based on the semester achievement index (*IPS*).

```
R8: IF the student takes the study plan
        THEN
   IF IP ≥ 3,00 THEN takes credit as many as 21-24
   IF IP ≥ 2,50 AND IP<2,99 THEN takes the maximum
credit as many as 18-21
   IF IP ≥ 2,00 AND IP<2,49 THEN takes maximum
credit  as many as 15-18
   IF IP ≥ 1,50 AND IP<1,99 THEN takes maximum
credit as many as 12-115
   IF IP <1,50 THEN takes maximum credit as many
as 12
```

At the time of filling the credits there is a calling stored function based on R8 rules. The stored function is shown in the following script:

```
ALTER  FUNCTION getJatahCREDIT
        (
        @vips AS decimal(9,2)
        )
        RETURNS int
        AS
BEGIN
        RETURN CASE
        WHEN @vips>=3.00 AND @vips<=4.00 THEN 24 -
maximum credit
        WHEN @vips>=2.50 AND @vips<=2.99 THEN 21 -
maximum credit
        WHEN @vips>=2.00 AND @vips<=2.49 THEN 18 -
maximum credit
        WHEN @vips>=1.50 AND @vips<=1.99 THEN 15 -
maximum credit
        ELSE 12
        END
END
```

The stored function will then be used to validate the number of credits taken based on the previous semester achievement index. If at any time the rule the maximum number of credits that the system will use change due to policy changes in its business rules then there is no need to re-open the front-end application script, simply change the value stored in the stored function.

6.4 Migration to Different Development Tools

The rules that have been planted in an active database system can be used by other development tools, noting that the development tools can connect to the database and execute stored procedures. This is happened if one day the front-end application that will be developed with different programming languages. To use the active database system in the form of trigger, stored procedure, and stored function features, it is necessary to know what triggers are attached to a table, what procedure name is in the database and its input parameters, then what function name and what parameters need to be input to use the function and what return value to know with the data type.

For example, an application built using Borland Delphi to access the active database has been created. Applications can use stored functions that are embedded in the academic information system database. For example to get the number of GPA from a student can use stored function that has been planted in active database. The Delphi programmer simply knows the name of the function and parameters passed to the function and the return value generated by the stored function, which can be modeled as follows:

```
        Dbo.getIPK(@npm varchar(20)) RETURNS deci-
mal(9,2)
```

From the information structure above, the Delphi programmer simply create a connection to the database, then declare a recordset to perform a query to the database eg:

```
        strSQL = "SELECT dbo.getIPK('" + npm.Text
+ "')"
```

Further query results can be used in applications created using Borland Delphi. The important thing to keep in mind is that developing a system requires a thorough understanding of the system to be developed.

6.5 Parameter Change Rules

Some rules embedded in the database software have value parameters used by the system. These parameters can be changed by the app users if there are any rule changes during system trips that require changing the parameter values in the rule. With the facility of parameter changes this rule can also provide an understanding to the user associated with the rules that have been planted in the database software.

6.6 Advantages and Disadvantages of the System

The system developed using this active database system has several advantages among others :

1. Some rules that are embedded in the active database system can be changed and adjusted to the business rules applied without dissecting the program code in the application. This will also make writing the program code on the application side/front-end to be simpler.
2. If at any time the application on the front-end side will be developed with other programming languages, then the rules that have been planted in the database software can be reused by the application.
3. In the use of active database system, the database not only acts as a container of data but also to apply reactive changes in data on the database.

In addition to the advantages possessed by the system developed, this system also has disadvantages, namely:

1. Implementation of active database system in this case take the case of academic information system based on this rule that have consequences to make programming in database side become more complex.
2. The development of the system is limited by following the business rules embedded in the database software.

7. Conclusion

From the results of research that has been done, researchers can be concluded some conclusions. The existence of ECA Rules can be used to translate the business rules of an information system and produce an active database system which is capable to monitor the application which in this study take the case of academic information systems. The existence of an active database system can help in developing an information system, because business rules implemented in logic and program algorithms are stored in the database, not fully stored in each application. So when a develop-

ment tool can access the database, then the development tools can directly use the rules that are planted in the active database system. The application environment serves as an input and output control mechanism of data going in and out of the database, whereas program logic and business rules are stored in the database software to monitor the entire system and ensure that data processing runs correctly.

References

[1] Vijayalakshmi, R., Duraipandian, N. 2014. Service Oriented Architecture and Privacy Preserving Mashup of Healthcare Data. *International Journal of Engineering & Technology*, vol.3, no.3, pp. 337-342.

[2] Kurdi, N.,K. 2005. *Rancang Bangun Sistem Administrasi Akademik (SIAKAD-2004) Berbasis Intranet Untuk Universitas Lampung.* Tesis Magister Ilmu Komputer. Yogyakarta: Universitas Gadjah Mada

[3] Efendi, R.M.M.H, 2007. *Perancangan Sistem Informasi Akademik di Fakultas Adab UIN Sunan Kalijaga Yogyakarta dengan Konsep Human Computer Interaction.* Tesis Magister Teknologi Informasi UGM. Yogyakarta

[4] Fachrurrozi, M. 2008. *Peningkatan Fungsionalitas Perangkat Lunak Melalui Restrukturisasi Data (Studi Kasus : Sistem Informasi Akademik Fakultas Ilmu Komputer Universitas Sriwijaya)*. Tesis Magister Informatika ITB. Bandung

[5] PATON, N. W.; DÍAZ, O.; 1999. Active Database Systems. *ACM Computing Surveys*, vol.31, no.1, pp.63-103.

[6] Lanova, W. 2009. *Sistem Penjadwalan dengan Constraint Menggunakan Active Database.* Skripsi Departemen Teknik Informatika IT. Telkom.

[7] Meybodi, M.R.; Rasoolzadegan, A. 2010. Rule Scheduling in Active Database Using Learning Automata. *International Journal of Computer, Mathematical Sciences and Applications*, vol.4, no.1-2, pp.239-262.

[8] Gupta, S.R.; Ali, M.S. 2010. Performance Evaluation in Active Real-Time Database System through Buffer Management. *International Journal of Computer Science and Application, pp.151-156.*

[9] Guo, YZ. 2013. University Teaching Administration System based on Active Database. Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013), published by Atlantis Press, Paris, France.

[10] Ephzibah, E.P.; Sujatha, R. 2017. Big Data Management with Machine Learning Inscribed by Domain Knowledge for Health Care. *International Journal of Engineering & Technology*, vol.6, no.4, pp. 98-102.

[11] ACT-NET Consortium. 1996. The Active Database Management System Manifesto: A Rulebase of ADBMS Features. *ACM SIGMOD Record*, vol.25, no.3, p.40-49.

[12] Butleris, R; Kapocius, K. 2002. *The Business Rules Repository For Information Systems Design.* ADBIS. Department of Information Systems, Kaunas University of Technology, Lithuania.

[13] Rabuzin K, Malekovi M, Lovreni A. 2007. The Theory of Active Databases vs. The SQL Standard. *The Proceedings of 18 International Conference on Information and Intelligent Systems*; Varaždin, Croatia, pp. 49-54.

[14] Klaus R. Dittrich; Stella Gatziu; Andreas Geppert. 1995. The active database management system manifesto: A rulebase of ADBMS features. *RIDS '95 Proceedings of the Second International Workshop on Rules in Database Systems*, Pages 3-20

[15] F. Casati, S. Castano, and M. Fugini. 2001. Managing Workflow Authorization Constraints through Active Database Technology. *Information Systems Frontiers,* vol. 3, no. 3, pp 319–338.

[16] Rabuzin, K.; Maleković, M.; Bača, M. Active databases, business rules and reactive agents - what is the connection? *Journal of Information and Organizational Sciences*, vol.29, no.1, pp. 63-73.

[17] Kozak, J. 2011. Rules in Database Systems. *WDS'11 Proceedings of Contributed Papers*, Part I, pp.131-136

[18] Sharm, R.K.; Ghandi, P. 2017. Estimate Reliability of Component Based Software System Using Modified Neuro Fuzzy Model. *International Journal of Engineering & Technology*, v.6, n.2, pp. 45-49.

[19] Singh, J.; Sharma, S. 2018. Fault Detection Technique for Test Cases in Software Engineering. International Journal of Engineering & Technology, vol.7, no.1, pp. 53-56.

[20] Kermani, E.R.; Momeni, M. 2016. Simultaneous Management and Control on Cost and Time in The Project Development Through Fuzzy Logic to Risk Management. *International Journal of Engineering & Technology*, vol.5, no.4, pp. 135-138.

[21] Yadav, M.; Kumari, S. 2016. A Paradigm for Active Database Management Systems. *International Journal of Engineering and Computer Science*, vol.5, no.1, pp.15572-15577.