International Journal of Recent Engineering Research and Development (IJRERD)
ISSN: 2455-8761
www.ijrerd.com || Volume 04 – Issue 05 || May 2019 || PP. 45-55

MS-SQL Server Stored Procedure for ABC Analysis Calculation in Power Plant Inventory Management

Yosua Alvin Adi Soetrisno[1], Budi Setiyono[2], Sukiswo[3], Agung Warsito[4], and Denis[5]

[1,2,3,4,5]*Diponegoro University, Department of Electrical Engineering,*
Jl. Prof. H. Sudarto, Semarang, Indonesia

Abstract: Stored Procedure is one kind of active database system that generates an automatic action on a specific time that needs updating the overall score of the entire database system. There are several kinds of Event Condition Action (ECA) including stored procedure, such as trigger and stored function that has a special capability. The capability is to monitor many inputs and to transmit several actions into many outputs. Separation of ECA rules component and access the process via program can extend the flexibility of development which is not interrupt the business process of application. This research applies the stored procedure in the asset management information system for ABC analysis. ABC analysis is needed to update the inventory turnover ratio through an overall analysis of every item. ABC analysis could be implemented in a database system by the stored procedure, so can become the right solution for the analytic problem. The ABC analysis calculates the lead time of inventory item from purchase requisition until the item can be applied in exchanging the broken item or used in the production environment. A stored procedure is used because there is a model-driven approach to communicating between several processes. Stored procedure process can update the database in the background process so not disrupt the model-driven process. The result of this research shows that the database can also have a role to maintain and control the process of updating the information acts as a program base function.

Keywords: stored procedure, ECA, ABC analysis, active database

1. Introduction

Data processing and management control of the inventory system is needed in the industry to maintain the quality of inventory item delivery and usability. There are several levels of customization in enterprise application with each risk and difficulty level.

Store procedure is one kind of solution which is easy to maintain but have a low risk of modification because not touching the internal class process of the program. Today, there are several stored procedure is used for exchanging information between two information system like an Application Programming Interface (API) [1].

In an asset management system, business need from the user can determine the business rule that would be maintained its data. In this research, stored procedure focus on the quality of item movement in inventory by calculating lead time.

Lead time in information system usually set manually by updating in database. There is no processing technique implemented in the programming layer or application layer to maintain inventory lead time adjustment [2].

The business process can be customized in top layer application by modify flow chart of in job assignment but not to recalculate all inventory item store. A business process can also be customized in the coding layer so the user can extend the functionality of the current class.

Risk of customization on the coding level is can break the functionality in application and broke the natural business process procedure [3]. In asset management system there is some modification to enhance business process management. A business process has many cycles and could change because of the rotation of job position.

A database system that has been used in the asset management system like IBM Maximo is a passive database. IBM Maximo store the function in the application layer on the database. Although the application layer has to validate the function through the database, the database is still acting passively [4].

Therefore for using database actively, there is a need to have some scenario and combination of stored procedure and scheduling. Database system is able to trigger action right on schedule and not limited with programmability constraint to extend the class of application [5]. In the active system database, language and logic that used is like a function in standard programming language and logic.

2. Related Work

Research conducted by Miftakul [6] uses an active database system in an education information system. Miftakul [6] research shows that 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 the application. Development tools can directly use the rule implemented in an active database system.

The disadvantage of that research is programming in database side become more complex and the development of the system is limited by following a business rule. In this research, the application is different because the analysis process is based on inventory business rule.

There is another research done by Kalanat [7] using ECA with triggering active database system using data mining methods. Kalanat [7] research show that ECA could strongly express application semantics. So when there is an association between action and program context, the system is automatically done the ECA process.

The disadvantage of that research is the implementation of an active application requires many complex rules from the data mining model to specify the system's active behavior. Associative classification method was introduced into the process of to design the trigger of active rule process. In this research, because there is a specific analysis that can be done, data mining technique no need to be done.

Morais [8] research using active in database processing to support ambient assisted living system. Database triggers are used to detect and respond to events taking place in a home environment. The event-driven architecture provided by active database makes it possible to implement an in-database service to monitor an individual's presence or absence in bed, as well to discover common room transitions and deviations during the night.

The future development of the application can be developed in a smart environment that doing tracking for the situation. The difference from Morais [8] research, in this research, all stored procedure is to trigger some action on particular time not on every situation change. The update only needs in many time because if done in every time transaction happens, it can disturb transactional processing.

Adi [9] research presents the "situation manager", a tool that includes both a language and an efficient run time execution mechanism, aimed at reducing the complexity of active applications. Situation that need to be handled are when a client wishes to activate an automatic "buy or sell" program, customer relationship manager wishes to receive an alert, groupware user wishes to start a session which there are ten members of the group, and a network manager wishes to receive an alert if the network will be overloaded. The application in Adi [9] research is for active monitoring and difference from this research. This research acquires data at a particular time, not aiming for a real-time alert system but important to update every day.

Contribution in this research is to provide a proper solution for exchanging information and recalculating score based on several rule criteria. The field is specified in inventory management purposes. Real-time processing is no need because there are active transaction happen. If the score is updated in an active transaction, there would be out of sync process, because if the secure stock point is reached there is an automatic process that generates purchase requisition.

3. Active Stored Procedure System

Active stored procedure system is implemented in a traditional database system, with some extra possibility to interact with data without application function. Stored procedure rule defined by an event, a check of condition, a chain of join rule from several business data, have some input generated from another rule and also have output action that can be applied to the whole database. Once store procedure function is defined there are scheduling system that created in the database to automatically execute the function.

The stored procedure is located in the same layer of a conventional database. In this architecture, conventional database can be converted to Active Database without need to modify the Passive Database section. The shape of the passive database shown in Figure 1.

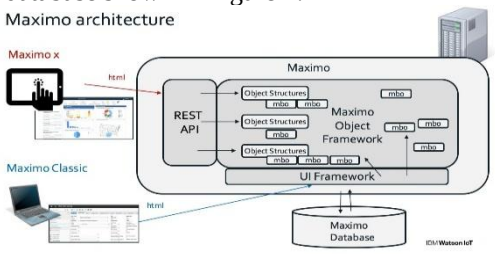

Figure 1: Maximo Passive Database Architecture

In the passive database architecture in Figure 1 it is shown that the database serves only as a data container for object structure and Maximo Object Framework. Maximo database also serves UI component for Maximo interface. Maximo database can be accessed online via REST API. Different from a passive database, active database not only serve a container of data but also has many functionality that can maintain status and score of the data. The shape of active database architecture is shown in Figure 2.

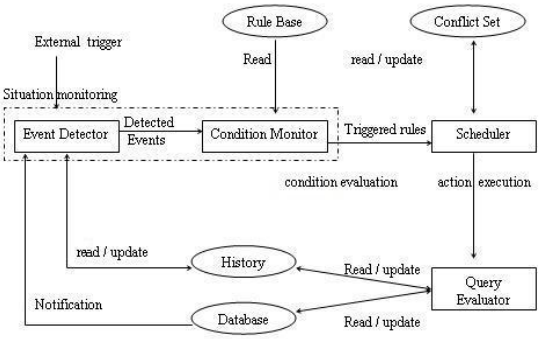

Figure 2: Active Database Architecture

Figure 2 shows the structure of action that can be handled by an active database. An active database can become situation monitoring agent that consists of an event detector and condition monitor component. Situation monitoring executed when there is an external trigger or there is a rule base for the condition of when the action executed. Triggered rule is pointing the scheduler to read/update conflict set and also query evaluator that can read/update history and entire database.

An enterprise information system usually consists of data management, business rule management, and user customization environment. Business development depends on what kind of enterprise information system original business aim. If customization is changing the original way to maintain the connection between organization function that could break the business process. When developing an information system on the basis of business rules, physically business rules are stored in a repository or database. There is 3 approaches to developing an information system. First, enterprise system has a customization tool that has already built inside the system.

Customization tools must add two component, in view and also in the database. Second is an enterprise system there is a class that can be extended to overcome difference process. The class can be extended with different behavior or overwrite the entire function, but could also to be used to call custom function. The third approach is to extend database functionality without changing the application process. This approach does not modify the application function. Modification of application function could not make change an entire database. Change the entire database also need application function that calls database function.

A stored procedure is decided to be used because ABC analysis changes the entire score of an inventory item in the database. ECA work after a series of rules has been determined. Active database monitor relevant event and then notifies the component responsible for executing the rule. Based on events there are two groups of the event for the active database. First, for simple events like INSERT, UPDATE, DELETE, and time events. Second, complex events consist of one or more events associated with some logic. There is some linear algebra condition in the database when selecting criteria that can be associated as a complex event like SUBQUERY, GROUPING condition, REPEAT, and CASE condition.

4. Active Stored Procedure System

There are two approaches in applying a rules-based system in the database management system, which is the deductive and active database. Rules in the active database can be used to accompany programming language that usually used in the programming environment. Rules use dynamic manipulation to handle various input and output that happen in stored procedure execution.

Rules provide a function to present recommendations, directions, or strategies. Stored procedure use if-then form to present many inputs, process, and outputs condition. If then rules connect OLD parameter and NEW parameter that inserted, updated, or deleted from the database. There are some varieties that connect object and attribute. The cause varieties are Premise, Input, Condition, Antecedent, Data, and action. The consequent varieties are Conclusion, Output, Action, Consequent, Results, Purpose, and Reaction.

5. Implementation

Active database in this application implemented in SQL server from Microsoft. System database that must be analyzed is based on Maximo version 7 database. Maximo 7 programmed using Java Environment and using WebSphere as middleware for an interconnecting database that contains function in Maximo business object in Java. Middleware is like web server with special service that encapsulating every server environment based on service that running. One middleware could handle many Maximo instance. Rule of a stored procedure is embedded and created directly in SQL server manager. In this section, we will present some implementation of database programming process in SQL server.

5.1 Stored Procedure Availability

Every item in the inventory system has a lead time from purchase requisition until material received to the warehouse or returned from the warehouse. In stored procedure input parameter set with @ character and name of the variable for input, which is in this case is "itemnum". Output parameter of this stored procedure is "leadtime". Lead time acquire by calculate date difference between purchase requisition and material receive divided by a number of item exist in a purchase requisition, left join with a purchase order on same Purchase Requisition (PR) number with a maximum revision number, left join with material receive base on Purchase Order (PO) number. This is a complete source implemented in SQL server.

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE dbo.sp_leadtime
@itemnumvarchar(30),
@leadtimevarchar(30) output
AS
BEGIN
SET NOCOUNT ON;
SET @leadtime = (select sum (datediff (day,pr.issuedate,matrectrans.transdate)) / count (*) from prline left join pr on pr.prnum = prline.prnum left join matrectrans on prline.ponum = matrectrans.ponum and matrectrans.itemnum = prline.itemnum and prline.polinenum = matrectrans.polinenum where prline.ponum is not null and prline.itemnum = "+@itemnum+" and matrectrans.porevisionnum = (select max(porevisionnum) from matrectrans where itemnum = "+@itemnum+" and matrectrans.matrectransid not in (select receiptref from matrectrans where itemnum = "+@itemnum+" and issuetype = 'RETURN') and matrectrans.issuetype != 'RETURN'))
IF @leadtime is null
BEGIN
SET @leadtime = 0
END
ELSE
SET @leadtime = @leadtime
UPDATE inventory
SET deliverytime = @leadtime
WHERE itemnum = @itemnum
IF @leadtime> 90
BEGIN
UPDATE inventory
SET availability = 'A'
WHERE itemnum = @itemnum
END
IF @leadtime<= 90 and @leadtime>=30
BEGIN
UPDATE inventory
SET availability = 'B'
WHERE itemnum = @itemnum
END
IF @leadtime< 30
BEGIN
UPDATE inventory
```

```
SET availability = 'C'
WHERE itemnum = @itemnum
END
IF @leadtime is null
BEGIN
UPDATE inventory
SET availability = 'C'
WHERE itemnum = @itemnum
END
END
GO
```

After query for select lead time is defined, the condition of each lead time must be defined in if then else terminology. If lead time calculation is missed then the value of lead time is set to be zero. Lead time zero shows that item never reach material receive in the warehouse and needed to be re-inspect. In ABC analysis term if item lead time is greater than 90 days then item availability is graded to A. If lead time greater or equal 30 days and lower or equal 90 days then item is graded to B. If lead time lower than 30 days or null then item is graded to C. After getting all level of the item than database must be updated with current value of ABC analysis calculation.

Stored procedure of dbo.sp_leadtime need input parameter "itemnum" that must be populated from another stored procedure call dbo.sp_updateleadtime. This complete stored procedure listed below:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE dbo.sp_updateleadtime
AS
BEGIN
SET NOCOUNT ON;
DECLARE @var1 varchar(30)
DECLARE @var2 varchar(30)
SET @var1 = 1
SET @var2 = (select max(inventoryid)+1 from inventory)
WHILE @var2>@var1
BEGIN
SELECT @var2 = @var2 – 1
DECLARE @leadtimevarchar(30)
DECLARE @pilihitemnumvarchar(30)
SET @pilihitemnum = (select itemnum from inventory where inventoryid = @var2 and location = 'WH01')
EXEC dbo.sp_leadtime @pilihitemnum,@leadtime output
END
END
GO
```

The stored procedure of dbo.sp_leadtime select the max id of the item in inventory and then doing while loop until the smallest item id. This stored procedure execute stored procedure dbo.sp_leadtime with transferring all item id to that procedure so there is an update to all item in inventory.

5.2 Stored Procedure Usage

Every item in the inventory system also has usage cost. Usage is got from material total usage multiplied by cost in the current currency in one year period. Usage stored procedure can be combined with lead time stored procedure to get service level inversion. Usage can also be combined with availability to get Reorder Point, and also Reorder quantity to get safety stock level. This is a complete source implemented in SQL server.

```
CREATE PROCEDURE dbo.sp_usage
@itemnumvarchar(50),
@usage decimal output
```

```
AS
BEGIN
SET NOCOUNT ON;
DECLARE @now datetime
SET @now = (select GetDate())
DECLARE @oneyearagodatetime
SET @oneyearago = (select DateAdd (dd,-365,GetDate()))
SET @usage = (select sum (currencylinecost) from matusetrans where itemnum = "+@itemnum+" and transdate
between "+@oneyearago+" and "+@now+")
IF @usage > 500000000
BEGIN
UPDATE inventory
SET usage = 'A'
WHERE itemnum = @itemnum
END
IF @usage <= 500000000 and @usage >=100000000
BEGIN
UPDATE inventory
SET usage = 'B'
WHERE itemnum = @itemnum
END
IF @usage < 100000000 and @usage > 0
BEGIN
UPDATE inventory
SET usage = 'C'
WHERE itemnum = @itemnum
END
IF @usage <= 0
BEGIN
UPDATE inventory
SET usage = 'D'
WHERE itemnum = @itemnum
END
IF @usage is null
BEGIN
UPDATE inventory
SET usage = 'D'
WHERE itemnum = @itemnum
END
END
GO
```

There are also several criteria for usage score. If usage is greater than 500 million then usage rank is A. If usage is lower than equal 500 million rupiahs and greater than equal 100 million rupiahs then usage rank is B. If usage is lower than 100 million rupiahs and greater than zero then usage rank is C, other than that, the rank is D. Stored procedure for usage rank also be completed with triggered action like sp_update leadtime but with execution for update usage rank stored procedure.

5.3 Stored Procedure Service Level Inverse

After getting usability and availability, the service level of inventory can be calculated based on several conditions. Service level (inventory) represents the expected probability of not hitting a stock-out. This percentage is required to compute the safety stock. Intuitively, the service level represents a trade-off between the cost of inventory and the cost of stock-outs (which incur missed sales, lost opportunities, and client frustration among others). This is the snippet of the stored procedure for service level inverse.

```
CREATE PROCEDURE dbo.sp_servicelevel
@itemnumvarchar(50)
AS
BEGIN
```

```
SET NOCOUNT ON;
DECLARE @criticality varchar(30)
SET @criticality = (select criticality from inventory where itemnum = "+@itemnum+" and location = 'WH01')
DECLARE @availability varchar(30)
SET @availability = (select availability from inventory where itemnum = "+@itemnum+" and location = 'WH01')
DECLARE @usage varchar(30)
SET @usage = (select usage from inventory where itemnum = "+@itemnum+" and location = 'WH01')
IF @criticality = 'A'
BEGIN
IF @availability = 'A'
BEGIN
IF @usage = 'A'
BEGIN
UPDATE inventory
SET servicelevel = '99.99'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '3.719'
WHERE itemnum = @itemnum
END
IF @usage = 'B'
BEGIN
UPDATE inventory
SET servicelevel = '99.99'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '3.719'
WHERE itemnum = @itemnum
END
IF @usage = 'C'
BEGIN
UPDATE inventory
SET servicelevel = '97'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '1.881'
WHERE itemnum = @itemnum
END
END
IF @availability = 'B'
BEGIN
IF @usage = 'A'
BEGIN
UPDATE inventory
SET servicelevel = '97'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '1.881'
WHERE itemnum = @itemnum
END
IF @usage = 'B'
BEGIN
UPDATE inventory
SET servicelevel = '97'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '1.881'
```

```
WHERE itemnum = @itemnum
END
IF @usage = 'C'
BEGIN
UPDATE inventory
SET servicelevel = '95'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '1.645'
WHERE itemnum = @itemnum
END
END
IF @availability = 'C'
BEGIN
IF @usage = 'A'
BEGIN
UPDATE inventory
SET servicelevel = '90'
WHERE itemnum = @itemnum
UPDATE inventory
SET servicelevelnormsdist = '1.282'
WHERE itemnum = @itemnum
…
END
GO
```

There are several conditions and normal probability distribution for each percentage of service level shown in Table 1. A normal probability distribution is better to be achieved for entire warehouse system if the warehouse system want to be effective for delivery an item service.

Table 1: Normal Probability Distribution of each Service Level

No	Service Level	Normal Probability Distribution
1	99.99%	3.719
2	98.00%	2.054
3	97.00%	1.881
4	96.00%	1.751
5	95.00%	1.645
6	93.00%	1.476
7	92.00%	1.405
8	90.00%	1.282
9	87.00%	1.126

ABC analysis of one item in inventory sampling is shown in Figure 3.

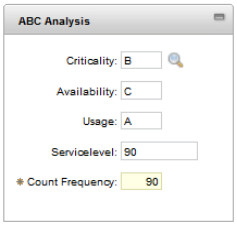

Figure 3: ABC Analysis GUI in Maximo Item

ABC analysis process checking is shown in Table 2. From table 2, we can consider that pre-memory and retrofit item is an item that has not to be used again around the inventory, so inventory ABC analysis

become null for that status.

Table 2: ABC Analysis Process Checking in Power Plant Inventory

Item ID	Inventory Status	Total Item	ABC Analysis Process
1	NORMAL	1492	OK
0	PRE-MEMORY	1492	NULL
4	NORMAL	1596	OK
16	NORMAL	1602	OK
2	NORMAL	1645	OK
25	NORMAL	1708	OK
10	PRE-MEMORY	1708	NULL
15	NORMAL	1710	OK
30	PRE-MEMORY	1710	NULL
20	PRE-MEMORY	1713	NULL
14	PRE-MEMORY	1716	NULL
3	NORMAL	1854	OK
8	NORMAL	1879	OK
7	NORMAL	1940	OK
5	NORMAL	2008	OK
88	PRE-MEMORY	2059	NULL
80	NORMAL	576	OK
99	PRE-MEMORY	576	NULL
40	NORMAL	585	OK
50	NORMAL	592	OK
19	PRE-MEMORY	592	NULL
60	NORMAL	599	OK
6	NORMAL	619	OK
51	NORMAL	626	OK
24	NORMAL	634	OK
32	PRE-MEMORY	634	NULL
96	RETROFIT	7	NULL

6. Conclusion

From the result of ABC analysis testing on a real database running in Maximo Environment, this research can concluded some conclusions. The existence of ECA stored procedure could be use to standardize several item in the database using ABC analysis. ABC analysis is a special case that needs to address at the inventory management system. Inventory management ABC analysis in Maximo Power Plant asset management is ruled by Decision Letter from Director of the Perusahaan Listrik Negara (PLN). Developing a stored procedure that separates from the application layer can make an easy solution for the overall analysis of the system without interfering the entire transaction process and also application interop ability in a model-driven manner. A stored procedure can directly be implemented and become part of the database. A stored procedure could be executed in safely time like after transaction is over in after work hour. The application testing of ABC analysis using usability, lead time, and criticality shows that ABC analysis can be executed successfully in entire system expect the obsolete item status. Obsolete status in the database is for the pre-memory item and also retrofit item.

References

[1]. M. Brantner, "Modern stored procedures using GraalVM: invited talk," in Proceedings of The 16th International Symposium on Database Programming Languages - DBPL '17, Munich, Germany, 2017, pp. 1–1.
[2]. S. Nallusamy, R. Balaji, and S. Sundar, "Proposed Model for Inventory Review Policy through ABC Analysis in an Automotive Manufacturing Industry," Int. J. Eng. Res. Afr., vol. 29, pp. 165–174, Mar. 2017.
[3]. M. Meyer, F. Beck, and S. Lohmann, "Visual monitoring of process runs: An application study for stored procedures," in 2016 IEEE Pacific Visualization Symposium (PacificVis), Taipei, Taiwan, 2016, pp. 160–167.
[4]. H. Schwichtenberg, "Reading and Modifying Data with SQL, Stored Procedures, and Table-Valued Functions," in Modern Data Access with Entity Framework Core, Berkeley, CA: Apress, 2018, pp. 305–315.
[5]. R. Adaikkalavan and S. Chakravarthy, "SnoopIB: Interval-based event specification and detection for active databases," Data Knowl. Eng., vol. 59, no. 1, pp. 139–165, Oct. 2006.
[6]. M. Miftakul Amin, A. Maseleno, S. K, E. Perumal, R. M Vidhyavathi, and L. Sk, "Active Database System Approach and Rule Based in the Development of Academic Information System," Int. J. Eng. Technol., vol. 7, no. 2.26, p. 95, May 2018.
[7]. N. Kalanat and M. R. Kangavari, "Data Mining Methods for Rule Designing and Rule Triggering in Active Database Systems," Int. J. Database Theory Appl., vol. 8, no. 1, pp. 39–44, Feb. 2015.
[8]. W. de Morais, J. Lundström, and N. Wickström, "Active In-Database Processing to Support Ambient Assisted Living Systems," Sensors, vol. 14, no. 8, pp. 14765–14785, Aug. 2014.
[9]. A. Adi and O. Etzion, "Amit - the situation manager," VLDB J. Int. J. Very Large Data Bases, vol. 13, no. 2, pp. 177–203, May 2004.

Author Profile

Yosua Alvin Adi Soetrisno was a lecturer from Diponegoro University. Yosua was born on Semarang, 13th October 1990. Yosua got his first bachelor degree from Diponegoro University majoring computer networking. Yosua got his master degree from GadjahMada University majoring software engineering. Yosua has a brief experience in programming of computer services, especially in IoT web-based environment.

Budi Setiyono was a senior lecturer and lectorfrom Diponegoro University. Budi was born on Purbalingga, 21th May 1970. Budi got his first bachelor degree from GadjahMada University majoring control and automation process. Budi also got his master degree from GadjahMada University also majoring control and instrumentation system. Budi has a brief view in the control system and has a strong experience in building IoT system.

Sukiswo was a senior lecturer and lectorfrom Diponegoro University. Sukiswo was born on Semarang, 14th July 1969. Sukiswogot his first bachelor degree from Diponegoro University majoring satellite communication. Sukiswo got his master degree from Bandung Institute of Technology also majoring telecommunication system. Sukiswo has a brief view in telecommunication and computer networking system.

AgungWarsito was a senior lecturer and associate professor from Diponegoro University. Agung was born on Solo, 17th June 1958. Agung got his first bachelor degree from GadjahMada University majoring electrical power. Agung got his master degree from Institut National Polytechnique de Toulouse also majoring power electronic. Agung has a brief view in electrical power and has a strong experience in power electronic and power quality factor analytic.

Denis was a lecturer from Diponegoro University. Denis was born on France, 17th April 1991. Denis got his first bachelor degree from Diponegoro University majoring power electrical. Denis got his master degree from GadjahMada University majoring renewable energy system. Denis has a brief experience to covering all energy flow through control system.