I.J. Modern Education and Computer Science, **2019, 3, 1-9**
Published Online March 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2019.03.01

LINE Messenger as a Transport Layer to Distribute Messages to Partner Instant Messaging

Kadek Darmaastawan
Student, Department of Information Technology, Engineering Faculty, Udayana University, Bali, Indonesia
Email: k.darmaastawan@gmail.com

I Made Sukarsa and Putu Wira Buana
Lecturer, Department of Information Technology, Engineering Faculty, Udayana University, Bali, Indonesia
Email: {sukarsa@unud.ac.id, wbhuana@it.unud.ac.id}

Received: 19 January 2019; Accepted: 13 February 2019; Published: 08 March 2019

***Abstract*—**The rapid development of technology generated a lot of instant messaging applications, as an example is LINE Messenger, Telegram, WhatsApp, Facebook Messenger, BlackBerry Messenger, etc. Many instant messaging applications cause some problems, one of which is the possibility of owning more than one instant messaging applications to meet messaging needs. This is very ineffective because messages will be received from various instant messaging applications and have to open all instant messaging applications one by one to reply to those messages. Application Programming Interface technology of instant messaging can be used to build a transport layer system that allows message exchange with several instant messaging applications using only one instant messaging application. The system built in this research requires a master instant messaging to be able to distribute messages to partner instant messaging, which in this case uses LINE Messenger as the master instant messaging to distribute messages to several partner instant messaging, which is Telegram and Extensible Messaging Presence Protocol.

***Index Terms*—**Instant messaging, application programming interface, transport layer, LINE Messenger, Telegram, Extensible Messaging Presence Protocol.

I. Introduction

Instant messaging is an application that makes people easily communicate in real-time. Instant messaging can connect one user to another user in other cities, also in a different country. The users of instant messaging are able to send messages, make calls, make video calls, share files to a group or person, and also make the user keep in contact with the other people easily [1]. At this moment, there are many kinds of instant messaging applications that have been developed for the desktop application, website application or smartphone application. Facebook Messenger, Telegram, LINE messenger, and WhatsApp are several examples of instant messaging application that can be used these days.

Ferdinand Zebua held a survey in Daily Social ID about the usage of instant messaging in 2017 with 1,022 Indonesian people as the respondents. The result of the survey shows that 97.34% of respondents have used WhatsApp, 85.82% of respondents have used LINE Messenger, 77.26% of respondents have used Facebook Messenger, 35.59% of respondents have used Telegram, 20.55% respondents have used other instant messenger application, and 0.10% of respondents have never used instant messaging applications mentioned in the survey [2]. The result of the survey showed that there are Indonesia citizens who use more than one instant messaging applications for messaging needs. The usage of more than one instant messaging application is not really effective because the message will be received from several instant messaging applications and the user has to open all the instant messaging applications one by one to reply those messages.

The solution that can be given based on the problem that had been explained is by using the Application Programming Interface (API) of instant messaging to build a transport layer system. Transport layer system in this research is a system which uses Line Messenger as a transport layer to distribute messages to several partner instant messaging, that was Telegram and Extensible Messaging Presence Protocol (XMPP). It is possible for the system to send a message to Telegram and XMPP with only using LINE Messenger. LINE Messenger, Telegram, and XMPP are used because API of those instant messaging applications can be used for free and based on the survey [2], those instant messaging applications are used by lots of Indonesia citizen.

II. Related Works

The concept that is used in building the system is by utilizing API as a transport layer. Dmitry Namiot built a prototype called T411 in Twitter as a Transport Layer [3]. T411 is a service that changes Twitter account into an account that can automatically reply the request from a tweet or direct message on Twitter. The concept that is

used in T411 refers to the service-based scheme at Short Message Service (SMS). The scheme works with a certain number which received a message from SMS and then the message can be processed to deliver a response. Diverge with the scheme that used at SMS, T411 uses Twitter API to receive a message. The message that has been received then processed by a particular machine, resulting in a response that matched the keyword in the message and will send a response as a new tweet using Twitter API.

Similar utilization of API has been proposed in [4,5,6,7,8,9,10]. Telegram API is used to build a system that can communicate with the Internet of Things (IoT) component like Arduino or Raspberry Pi. The main idea is using Telegram API as a transport layer to receive a request in the form of a message with a keyword that was sent by the user using Telegram and send back the response based on the request. As an example, receiving the particular value of censor which attached to Arduino or Raspberry Pi. The particular machine is needed to translate the keyword and send the request to Arduino or Raspberry Pi to receive the value of censor. The value of censor then will be sent back to the user as a message using Telegram.

The request of the user is not limited only to receive the value of censor but also changing it, therefore the IoT component can take an action such as locking or opening the door using the message from Telegram [5], [6], [8,9,10].

Dewa Agung Khrisna Arimbawa P in Library System Using Radio Frequency Identification (RFID) and Telegram Bot API proposes a library system by utilizing RFID technology and also Telegram API to optimize the conventional library system. RFID is used to increase the library service such as book inventory, book loan, and security aspects. Telegram API is used to access information such as notification, history of book loan, history of book returning, book catalogs and making a book order using Telegram Bot. A message with a certain keyword is used to receive information using Telegram. The utilization of Telegram API in the library system is very effective because the library information can be accessed from the telegram without opening the library system [11].

The other utilization of Telegram API is discussed in Design of Telegram Bots for Campus Information Sharing. The research proposed a system that is able to share campus information using Telegram Bot. The communication between the system and Telegram Bot is done with webhook. Webhook can supply zero latency in the communication process with Telegram Bot. Telegram Bot that was developed in that research can also receive keyword to give particular information [12].

Similar but not same with research [12], research [13] utilize Facebook Messenger API to make a bot for sharing information around the campus. Facebook Messenger Bot make it possible for the university student to acquire information such as the information of the alumnus also the task given by the lecturer. Keyword also needed to collect the information using Facebook Messenger Bot, where the keyword will be processed to show the information appropriate with the request.

Telegram API also utilized at research [14] to build a bot that can be used to share information about train, such as train status, the availability of seats and others. That information can be obtained using Telegram Bot by sending a request in the form of the train code. The main purpose of using the bot is to increase the comfort of the user. Telegram Bot usage has some benefit for train passengers because they can save time, storage and internet data usage, because the information can be obtained from the bot without opening the train website.

Other API utilization that has been discussed in [15]. The system works when there is new data in the first database, then the new data will be sent to the second database. New data in the first database is forwarded to the second database using Dropbox API. Data exchange between databases includes a particular machine named Scheduler. The Scheduler has a task to receive new data in the first database, synchronize with the Dropbox using API and send it to the second database.

Instant messaging API also used in [16]. The research built a consultation robot about Balinese Calendar using Telegram Bot API.

A system built in [17] also utilize several instant messaging API such as Telegram, LINE Messenger, and XMPP to transform existing information system into chat services with natural interaction using Bahasa conversations, so the user can access information system through chatting with Telegram, LINE Messenger, or XMPP.

All the systems that have been proposed in related works use the same concept, which is using API as a transport layer. The system built in this research utilizes API of LINE Messenger, Telegram, and XMPP as a transport layer that allowed cross-application message exchange.

Here are presented some literature to support the research process which contains the materials that will be referenced in this study. References contained in this part are API, instant messaging and transport layer.

A. API

API provides a set of certain functionalities to the application developer. The main goal of API is to give access in using a certain function that available in the existing system without writing the code from zero [18].

One of the instant messaging that has public API is LINE Messenger, which is named Messaging API. Messaging API from LINE Messenger makes it possible to the application developer to make a service that can do a two-way communication between the services (LINE Messenger Bot) with LINE Messenger user [19].

B. Instant Messaging

Instant messaging is one of the communication services that support two people or more to send a message through the internet in real-time. Instant messaging system has become an important tool in

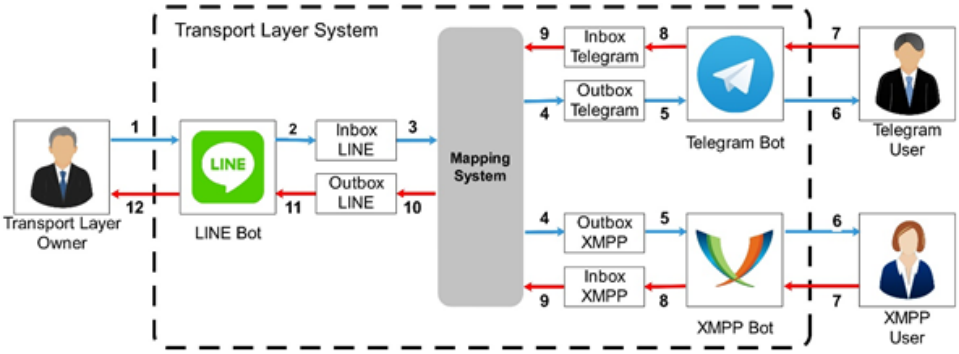

Fig.1. System overview.

everyday life [20]. The rapid technology development induces the availability of the instant messaging application, for example, Facebook Messenger, Telegram, LINE Messenger, WhatsApp, and the others. Those instant messaging applications have some similar features. One of those is the communication features between two or more user through text messages, voice call, and video call, also to share files [21].

1) LINE Messenger

LINE Messenger is an instant messaging that allows a user to send a message in a group or personal chat. LINE Messenger is available on many platforms, such as PC and smartphone [22].

2) Telegram

Telegram is an instant messaging that can be used at several platforms such as smartphone, PC and can be accessed through a website. Telegram message is highly encrypted and has a self-destruct feature. Telegram allowed to access message from multiple devices. [23].

3) XMPP

XMPP is one set of open technology for instant messaging, presence (indication to choose whether the users are ready to accept the message or not), multi-party chat, also video and voice call. XMPP offer several main benefits such as it was open, means the XMPP protocol is free to use, open to the public, and also easy to understand. [24].

C. Transport Layer

Layer in a connection or computer network referred to what is known as the OSI layer. OSI layer is a logical system consists of seven layer that has an important role in the data transmission process and exists in a network connection. Every connection or data transmission has to go through seven layers, therefore it can be transmitted well and perfectly. One of the most important OSI layers is the transport layer, which is the fourth layer in OSI Layer. Transport layer in a computer network has the same meaning as the name, a layer for transportation. Transport layer responsible to send the data to the corresponding process in the computer host [25].

III. System Overview

The process flow of LINE Messenger as a transport layer to distribute messages to partner instant messaging generally can be seen in Fig. 1.

A. Component of The System

According to Fig. 1, LINE Messenger as a transport layer to distribute messages to partner instant messaging has several components.

1) Transport Layer Owner

Transport Layer Owner is the main user of the system that has the Line Messenger Bot as the base instant messaging and also the Telegram or XMPP Bot account as the partner instant messaging. Transport Layer Owner must register their bots to the system in order to exchange message with Telegram and XMPP User using the Line Messenger Bot.

2) Telegram User

Telegram User is the Transport Layer Owner's partner who used Telegram as the main instant messaging to exchange messages. Telegram User can exchange message with the Transport Layer Owner through Telegram Bot that has been registered by the Transport Layer Owner.

3) XMPP User

XMPP User is the Transport Layer Owner's partner who used XMPP as the main instant messaging to exchange messages. XMPP User can exchange message with the Transport Layer Owner through XMPP Bot that has been registered by the Transport Layer Owner.

4) Instant Messaging Bot

Instant messaging bot is a LINE Messenger, Telegram and XMPP Bot that has to be registered by Transport Layer Owner. Transport Layer Owner exchange message with Telegram and XMPP User through LINE Messenger Bot, while Telegram and XMPP User exchange message with Transport Layer Owner through Telegram and XMPP Bot. Every instant messaging bot that has been

registered to the system has its own webhook. Instant messaging bot and webhook work together to receive the messages sent by the user. Webhook is a script that is stored in the cloud. The system will automatically make webhook for every bot after the bot is registered.

5) Messages

The message is a main component of the system. Messages are shown with Line 1 until Line 12 in Fig. 1. Messages in the system sent through each instant messaging bot. The message is divided into two, which is the message sent by the Transport Layer Owner and the message sent by Telegram or XMPP User.

Messages sent by the Transport Layer Owner are shown with Line 1 until Line 6 in Fig. 1. These messages are a message that is sent through LINE Messenger and forwarded to Telegram or XMPP. Message sent by the Transport Layer Owner must have a structure or message format that contain the information of destination instant messaging, whom the message sent to and also the content of the message that will be sent. Example of a message that must be sent by the Transport Layer Owner is "Telegram user1 hello, how are you?" The example has several parts, 'Telegram' is the destination instant messaging. "User1" is the username of the destination of instant messaging, and "Hello, how are you?" is the content of the message that will be sent. The format is used because the Transport Layer Owner has many Telegram or XMPP User, therefore the message format is needed so the message is able to be sent properly.

Messages sent by Telegram or XMPP User are shown with Line 7 until Line 12 in Fig. 1. These messages are a message that is sent through Telegram or XMPP and forwarded to LINE Messenger. These messages do not have any format, which was because every Transport Layer Owner has their own instant messaging bot, therefore the message sent by Telegram or XMPP User through a registered bot must be sent to the appropriate Transport Layer Owner.

6) Inbox and Outbox

Inbox and outbox are a database table to store the message. Inbox is used to store the message sent by the user to the system, while outbox is used to store the message sent by the system to the user. Each instant messaging has their own inbox and outbox.

7) Mapping System

The mapping system is a sub-system used to map every message received by the system, therefore, it can be sent to the destination user. The overview of the mapping system generally can be seen in Fig. 2.

Mapping system consists of several mapping engines. Mapping engine was divided into three kinds, which are L/TX Engine use to map the message from LINE Messenger and the result will be stored in Telegram or XMPP outbox, T/L Engine which is used to map the message from Telegram and the message will be stored in the Line Messenger outbox, also X/L Engine which is used to map the message from XMPP and the result will be stored in the LINE Messenger outbox.

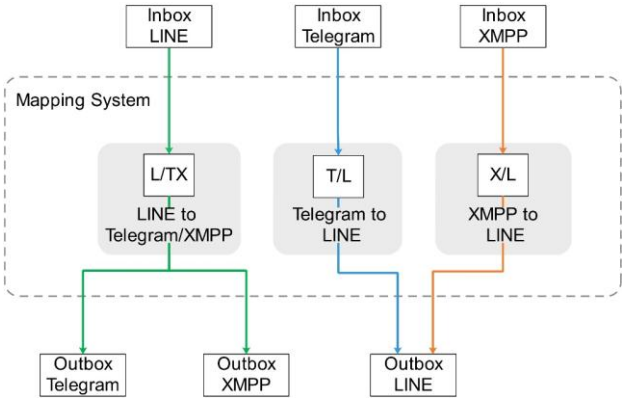

Fig.2. Mapping System.

Message mapping utilized the user chat identifier in each instant messaging. Every instant messaging user has a unique identifier. The identifier is used to map the message by mapping engine, therefore, the message can be sent to the appropriate user.

B. Cross-Application Message Exchange

The message exchange process that happens in the system was divided into two, first is sending a message by Telegram or XMPP user and second is replying the message by the Transport Layer Owner.

Sending message was done by Telegram or XMPP user through Telegram and XMPP Bot and will be received by Transport Layer Owner through LINE Messenger Bot. This process must be done first to get the chat identifier of Telegram and XMPP user, therefore the message can be replied by the Transport Layer Owner using the chat identifier. The steps of the sending messages process are shown with Line 7 until Line 12 in Fig. 1. The following is a more detailed explanation of sending a message by Telegram or XMPP User.

- Line 7 in Fig. 1 is the step where Telegram or XMPP User sends a message through Telegram or XMPP Bot.
- Line 8 in Fig. 1 is the step when the messages sent by telegram or XMPP user received by the system using Telegram or XMPP API and store the message in a text file in the cloud as a backup. Then continue with taking the file text in the cloud to save it on the database, which is at telegram or XMPP inbox.
- Line 9 in Fig. 1 is a step where the stored message in the database is taken by the T/L or X/L Mapping Engine. T/L or X/L Mapping Engine will map the message to determine the Transport Layer Owner.
- Line 10 in Fig. 1 is the steps when the message has been mapped and stored in the LINE Messenger outbox.
- Line 11 in Fig. 1 is the steps when the mapped messages that have been stored in LINE Messenger

outbox is sent to LINE Messenger using LINE Messenger API.
- Line 12 in Fig. 1 is the steps when the message is successfully sent to the Transport Layer Owner through LINE Messenger Bot.

After the message was successfully received by Transport Layer Owner, Transport Layer Owner can reply the message with the steps that can be shown with Line 1 until Line 6 in Fig. 1. The following is a more detailed explanation of replying a message by Transport Layer Owner.

- Line 1 in Fig. 1 is a step when the Transport Layer Owner sends reply message through LINE Messenger Bot.
- Line 2 in Fig. 1 is a step when the reply message sent by Transport Layer Owner is received by the system using LINE Messenger API and store the reply message in text files in the cloud as a backup. Then continue with taking the text files from the cloud to be stored in the database, which is at the LINE Messenger inbox.
- Line 3 in Fig. 1 is a step when the reply message that is stored in the database was taken by L/TX Mapping Engine. L/TX Mapping Engine will map the reply message to determine the Telegram and XMPP user.
- Line 4 in Fig. 1 is the steps when the reply message has been mapped and stored in the Telegram or XMPP outbox.
- Line 5 in Fig. 1 is the steps when the reply messages that have been stored in Telegram or XMPP outbox is sent to Telegram or XMPP using the corresponding API.
- Line 6 in Fig. 1 is the steps when the reply message is successfully sent to the Telegram or XMPP User through the corresponding bot.

IV. Result and Discussion

This section contains the result of the system testing, the data growth analysis, and the discussion.

A. System Testing

Transport layer system testing was done by doing several trials, which is sending a message from Telegram and XMPP to LINE Messenger, and also sending a reply message from LINE Messenger to Telegram and XMPP.

1) Sending a Message from Telegram and XMPP

The process started with the Telegram and XMPP User send a message through Telegram and XMPP Bot that has been registered by the Transport Layer Owner to the system as shown in Fig. 3 and Fig. 4.

Fig. 3 shows the display when the Telegram User sends a message, which is "Halo", through Telegram Bot. Telegram User send a message without any format as if doing the usual chatting. The message then will be received by the system using Telegram Bot API and forwarded to Transport Layer Owner.

Fig.3. Sending message by Telegram User.

Fig. 4 shows the display when the XMPP User sends a message, which is "hai, apa kabar?", through XMPP Bot. XMPP User sends a message without any format as if doing the usual chatting. The message then will be received by the system using XMPP Bot API and forwarded to Transport Layer Owner.

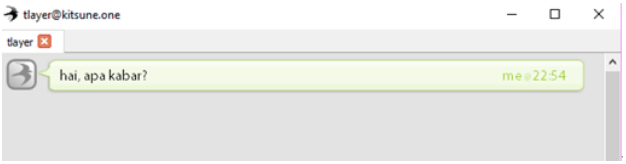

Fig.4. Sending message by XMPP User.

Both messages sent by Telegram and XMPP User will be mapped, therefore it can be sent and received by the appropriate Transport Layer Owner as shown in Fig. 5.

Fig. 5 shows the message sent by Telegram and XMPP User in Fig. 3 and Fig. 4 is received by the Transport Layer Owner through LINE Messenger Bot. In addition to the original message, the message received by Transport Layer Owner contains additional information, namely the origin of instant messaging and the message sender chat identifier.

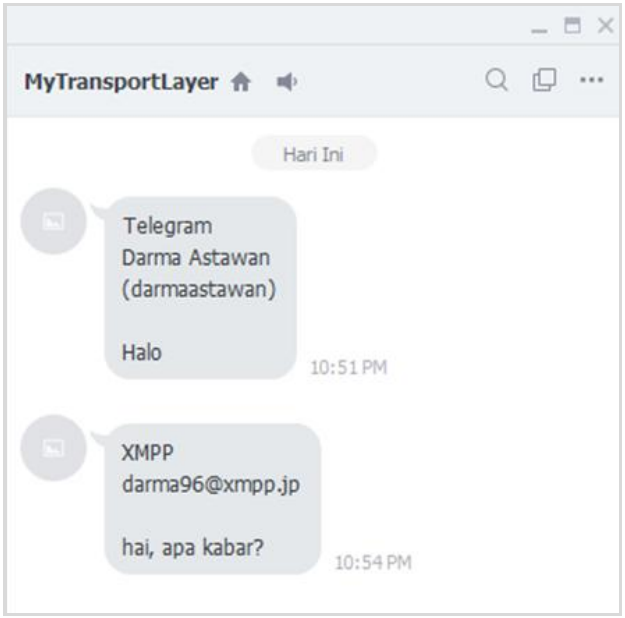

Fig.5. Message received by Transport Layer Owner.

2) Sending a Reply Message from LINE Messenger

Transport Layer Owner can reply the message received from Telegram and XMPP User by using LINE Messenger Bot. The reply message must have the format as shown in Fig. 6.

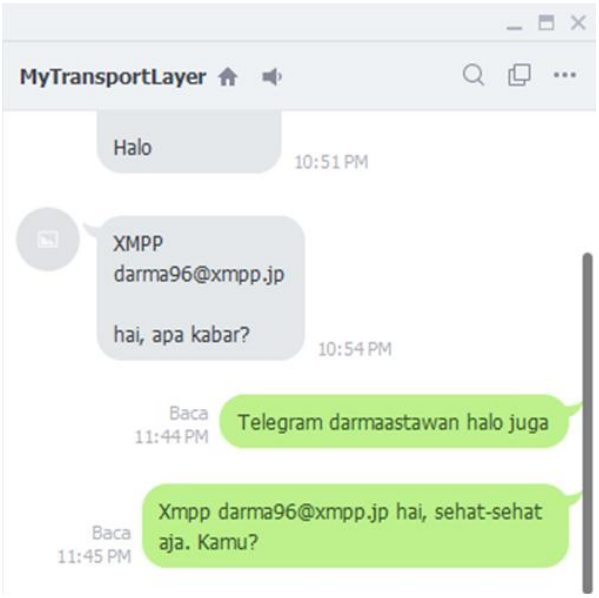

Fig.6. Sending reply message by Transport Layer Owner.

Both reply messages sent by Transport Layer Owner will be mapped, therefore it can be sent and received by the appropriate Telegram and XMPP User as shown in Fig. 7 and Fig. 8.

Fig. 7 shows the reply message, which is "halo juga", was successfully received by Telegram User. The message received by Telegram User did not have any additional information.

Fig. 7. Reply message received by Telegram User.

Fig. 8 shows the reply message, which is "hai, sehat-sehat saja. Kamu?" was successfully received by XMPP user. Same like message received by Telegram User, the message received by XMPP User did not have any additional information. The cross-application message exchange process was completed when the reply message was successfully received by Telegram and XMPP User.

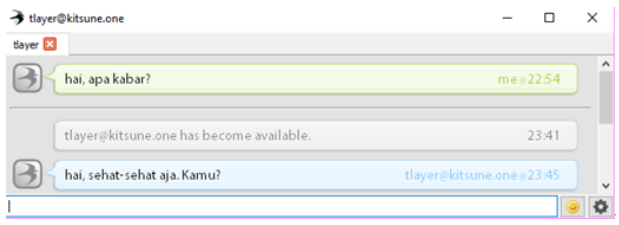

Fig.8. Reply message received by XMPP User.

B. *Data Growth Analysis*

Data growth analysis is the estimated calculation of the table storage space in the database that is needed to store the system data. The system uses several tables in the database. The tables in the database can be seen in Table 1.

Table 1. Master Tables

Table Name	Field	Data Type	Field's Size (Bytes)	Row's Size (Bytes)
tb_user	id (PK)	int (11)	4	251
	username	varchar (12)	13	
	password	varchar (32)	33	
	chatid_user	varchar (200)	201	
det_user	id (PK)	int (11)	4	210
	id_user (FK)	int (11)	4	
	bot	varchar (200)	201	
	messenger	enum	1	
tb_contact	id (PK)	int (11)	4	243
	iduser_mess	int (12)	4	
	nama_mess	varchar (32)	33	
	chatid	varchar (200)	201	
	messenger	enum	1	

Database tables used in the system consist of master tables and transaction tables. Master tables consists of tb_user, det_user, and tb_contact.

Those master tables used to store the Transport Layer Owner data as well as Telegram and XMPP User data. The details about master tables can be seen in Table 1.

Transaction tables consists of in_line, in_tele, in_xmpp, out_line, out_tele, and out_xmpp. in_line, in_tele, in_xmpp, out_line, out_tele, and out_xmpp is used to store the message data in the message exchange process. The details about transaction tables can be seen in Table 2.

Field's size in Table 1 and Table 2 is the size required by a field in one table based on the type and length of the data, where every data size referred to [26]. Row's size in Table 1 and Table 2 is the size of one row of data saved in the table, where the size was received by summing up every field's size per database table.

Data growth analysis is done by assuming that the system has 100 Transport Layer Owners. Every Transport Layer Owner has 2 friends, 1 friend is Telegram User and another is XMPP User that differ between one and another, therefore overall there are 100 Telegram Users and 100 XMPP Users. Each and every Transport Layer Owner is assumed to send 10 messages to each of their friends every day, therefore it becomes 2.000 messages sent in one day. All the Telegram and XMPP Users are also assumed to reply to every message sent by the Transport Layer Owner, which make overall 2.000 replied message sent in one day.

Table 2. Transaction Tables

Table Name	Field	Data Type	Field's Size (Bytes)	Row's Size (Bytes)
in_line	id (PK)	bigint (20)	8	268
	id_user (FK)	int (11)	4	
	chatid_partner	varchar (200)	201	
	in_msg	text	50	
	date	timestamp	4	
	flag	enum	1	
in_tele	id (PK)	bigint (20)	8	268
	id_user (FK)	int (11)	4	
	chatid	varchar (200)	201	
	in_msg	text	50	
	date	timestamp	4	
	flag	enum	1	
in_xmpp	id (PK)	bigint (20)	8	268
	id_user (FK)	int (11)	4	
	user_xmpp	varchar (200)	201	
	in_msg	text	50	
	date	timestamp	4	
	flag	enum	1	
out_line	id (PK)	bigint (20)	8	302
	id_user (FK)	int (11)	4	
	chatid_partner	varchar (200)	201	
	username	varchar (32)	33	
	out_msg	text	50	
	messenger	enum	1	
	date	timestamp	4	
	flag	enum	1	
out_tele	id (PK)	bigint (20)	8	268
	id_user (FK)	int (11)	4	
	chatid	varchar (200)	201	
	out_msg	text	50	
	date	timestamp	4	
	flag	enum	1	
out_xmpp	id (PK)	bigint (20)	8	268
	id_user (FK)	int (11)	4	
	user_xmpp	varchar (200)	201	
	out_msg	text	50	
	date	timestamp	4	
	flag	enum	1	

The master tables did not have the significant data growth, therefore, based on the assumption and data seen in Table 3, the storage space needed by all of the master tables is 50.810 bytes.

Table 3. Storage Space Requirement of Master Tables

Table Name	Row's Size (Bytes)	Number of Data	Storage Requirement (Bytes)
tb_user	251	10	2.510
tb_det_user	210	30	6.300
tb_contact	210	200	42.000

Database table det_user has 30 data because each row of the data stored in tb_user will result in three rows of data stored in det_user.

Contrary to master tables, transaction tables data were significantly growing. It is caused by the data of the message exchange process was stored in the transaction table. The detail about data growth and storage space requirement can be seen in Table 4.

Table 4. Storage Space Requirement of Transaction Tables

Table Name	Row's Size (Bytes)	Number of Data	Storage Requirement (Bytes)
in_line	268	2000	536.000
in_tele	268	1000	268.000
in_xmpp	268	1000	268.000
out_line	302	2000	604.000
out_tele	268	1000	268.000
out_xmpp	268	1000	268.000

The storage space needed by all of the transaction tables based on the assumption and data seen in Table 4 is 2.212.000.

C. *Discussion*

The system has several problems or weaknesses. The following is an explanation of the problems or weaknesses found in the system.

- The system has a chance of failure to send cross-application messages, which is caused by a large number of messages sent and mapped by the system at the same time.
- Based on the data growth analysis, system database can run out of storage, therefore, the new data input in the system cannot be stored in the database and the system will fail.

Future works that can solve system problems or weaknesses are as follows.

- To solve system failure in sending and mapping a large number of messages at the same time is to develop a load balancing system. The load balancing system which is duplicates the mapping engine so the system can send and map several messages at the same time by dividing the workload evenly across several engine mappings.
- It is necessary to develop a model of emptying data regularly from active databases to passive databases to overcoming large transaction data growth so that the size of the database will remain lean and the execution process will be faster.

V. Conclusion

The conclusion obtained from the study of LINE Messenger as a transport layer, to be distributed to an instant messaging partner is API technology of instant messaging application can be utilized to build a system that has functioned as a transport layer to send a cross-platform message, so the users can send and receive messages from several instant messaging using only one instant messaging. To solve the system problems or weakness, further research is needed to develop a load balancing system to divide the workload evenly, so the system failure on sending a cross-application message can be minimalized. In addition, it is necessary to develop a model of emptying database data regularly, so the system will not run out of storage.

References

[1] T. Sutikno, L. Handayani, D. Stiawan, M. A. Riyadi, and I. M. I. Subroto, "WhatsApp, viber and telegram: Which is the best for instant messaging?," *Int. J. Electr. Comput. Eng.*, vol. 6, no. 3, pp. 909–914, 2016.

[2] F. Zebua, "Laporan DailySocial: Survey Instant Messaging 2017," 2017. [Online]. Available: https://dailysocial.id/post/laporan-dailysocial-survey-instant-messaging-2017. [Accessed: 22-Jan-2018].

[3] D. Namiot, "Twitter as a transport layer platform," *Proc. Artif. Intell. Nat. Lang. Inf. Extr. Soc. Media Web Search Fruct Conf. AINL-ISMW Fruct 2015*, pp. 46–51, 2015.

[4] J. C. De Oliveira, D. H. Santos, and M. P. Neto, "Chatting with Arduino platform through Telegram Bot," *Proc. Int. Symp. Consum. Electron. ISCE*, pp. 131–132, 2016.

[5] V. R. Vatsa and G. Singh, "Raspberry Pi based Implementation of Internet of Things using Mobile Messaging Application - ' Telegram ,'" *Int. J. Comput. Appl.*, vol. 145, no. 14, pp. 17–21, 2016.

[6] R. G. Anvekar, "IoT Application Development : Home Security System," *IEEE Int. Conf. Technol. Innov. ICT Agric. Rural Dev.*, pp. 68–72, 2017.

[7] R. Eko, N. Sisyanto, and N. B. Kurniawan, "Hydroponic Smart Farming Using Cyber Physical Social System with Telegram Messenger," *Int. Conf. Inf. Technol. Syst. Innov.*, pp. 239–245, 2017.

[8] G. D. Kumar, "Realization Of A Low Cost Smart Home System Using Telegram Messenger And Voice," *Int. J. Pure Appl. Math.*, vol. 116, no. 5, pp. 85–90, 2017.

[9] P. N. V. S. N. Murthy, S. T. Rao, and G. M. Rao, "Home Automation using Telegram," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 6, no. 6, pp. 64–69, 2017.

[10] V. Jain and S. Chawla, "IMPLEMENTATION OF A SMART SAFETY AND SECURITY DEVICE USING RASPBERRY PI , TELEGRAM BOT , PROTA OS," *Int. J. Comput. Eng. Appl.*, vol. XII, no. II, pp. 221–228, 2018.

[11] D. A. K. Arimbawa P, I. K. G. D. Putra, and I. M. Sukarsa, "Library System Using Radio Frequency Identification (RFID) and Telegram Bot API," *Lontar Komput.*, vol. 9, no. 1, pp. 40–51, 2018.

[12] H. Setiaji and I. V. Paputungan, "Design of Telegram Bots for Campus Information Sharing," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 325, no. 1, 2018.

[13] Y. Mehta, "The College Chatbot," *Int. J. Comput. Appl.*, vol. 173, no. 7, pp. 34–37, 2017.

[14] S. Sakila V, A. Shrivastava, A. M. Ansari, L. K. Kar, and M. Kumar, "RAILBOT : A Railway (IRCTC) Chatbot," *Int. J. Eng. Sci. Comput.*, vol. 8, no. 4, pp. 16723–16726, 2018.

[15] A. Hanafi, I. M. Sukarsa, and A. A. K. A. C. Wiranatha, "Pertukaran Data Antar Database dengan Menggunakan Teknologi API," *Lontar Komput.*, vol. 8, no. 1, pp. 22–30, 2017.

[16] I. M. Sukarsa, I. K. Gede Darma Putra, N. Putra Sastra, and L. Jasa, "Modification of ISONER Framework as Enterprise Service Bus to Build Consultation Robot Using External Engine," vol. 3, no. 2, pp. 123–128, 2018.

[17] I. M. Sukarsa, I. K. Gede, D. Putra, N. P. Sastra, and L. Jasa, "A New Framework for Information System Development on Instant Messaging for Low Cost Solution," vol. 16, no. 6, pp. 2799–2808, 2018.

[18] A. A. Sawant and A. Bacchelli, "A dataset for API usage," *IEEE Int. Work. Conf. Min. Softw. Repos.*, vol. 2015–Augus, pp. 506–509, 2015.

[19] LINE Messenger Corporation, "Messaging API." [Online]. Available: https://developers.line.me/en/services/messaging-api/. [Accessed: 10-Mar-2018].

[20] C. Fu, Y. Tang, C. Yuan, and Y. Xu, "Cross-platform instant messaging system," *Proc. - 2015 12th Web Inf. Syst. Appl. Conf. WISA 2015*, no. Mvc, pp. 27–30, 2016.

[21] H. Chang and W. Ian, "Instant Messaging Usage and Interruptions in the Workplace," *Int. J. Knowl. Content Dev. Technol.*, vol. 4, no. 2, pp. 25–47, 2014.

[22] LINE Messenger Corporation, "LINE Messenger." [Online]. Available: https://line.me/en/. [Accessed: 10-Mar-2018].

[23] Telegram Corporation, "Telegram." [Online]. Available: https://telegram.org/. [Accessed: 10-Mar-2018].

[24] XMPP Corporation, "An Overview of XMPP." [Online]. Available: https://xmpp.org/about/technology-overview.html. [Accessed: 10-Mar-2018].

[25] A. H. Wheeb, "Performance Comparison of Transport Layer Protocols International Journal of Advanced Research in Performance Comparison of Transport Layer Protocols," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 5, no. 12, pp. 121–125, 2015.

[26] MySQL, "Data Type Storage Requirements." [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/storage-requirements.html. [Accessed: 20-Mar-2018].

Authors' Profiles

Kadek Darmaastawan, He graduated from high school in SMAN 3 Denpasar in 2014. He is currently in the process of studies at the Department of Information Technology, Udayana University, Bali, Indonesia. He also has the ability as an android developer, web developer and data management.

I Made Sukarsa, he obtained his Master Degree in Dept. Of Electrical Engineering at Gadjah Mada Univ. in 2003 He currently works as a lecturer in the Department of Information Technology University of Udayana. He has conducted almost 12 journal papers in the field of Data Warehouse, Middleware, and IT Governance.

Putu Wira Buana, he obtained his Master Degree in The Science of Applied Electronics at Brawijaya University in 2007. He currently works as a lecturer in the Department of Information Technology University of Udayana. He has conducted eight journal publications in the field of Emerging Technology And Enterprise And Industry Application.

How to cite this paper: Kadek Darmaastawan, I Made Sukarsa, Putu Wira Buana, " LINE Messenger as a Transport Layer to Distribute Messages to Partner Instant Messaging", International Journal of Modern Education and Computer Science(IJMECS), Vol.11, No.3, pp. 1-9, 2019.DOI: 10.5815/ijmecs.2019.03.01