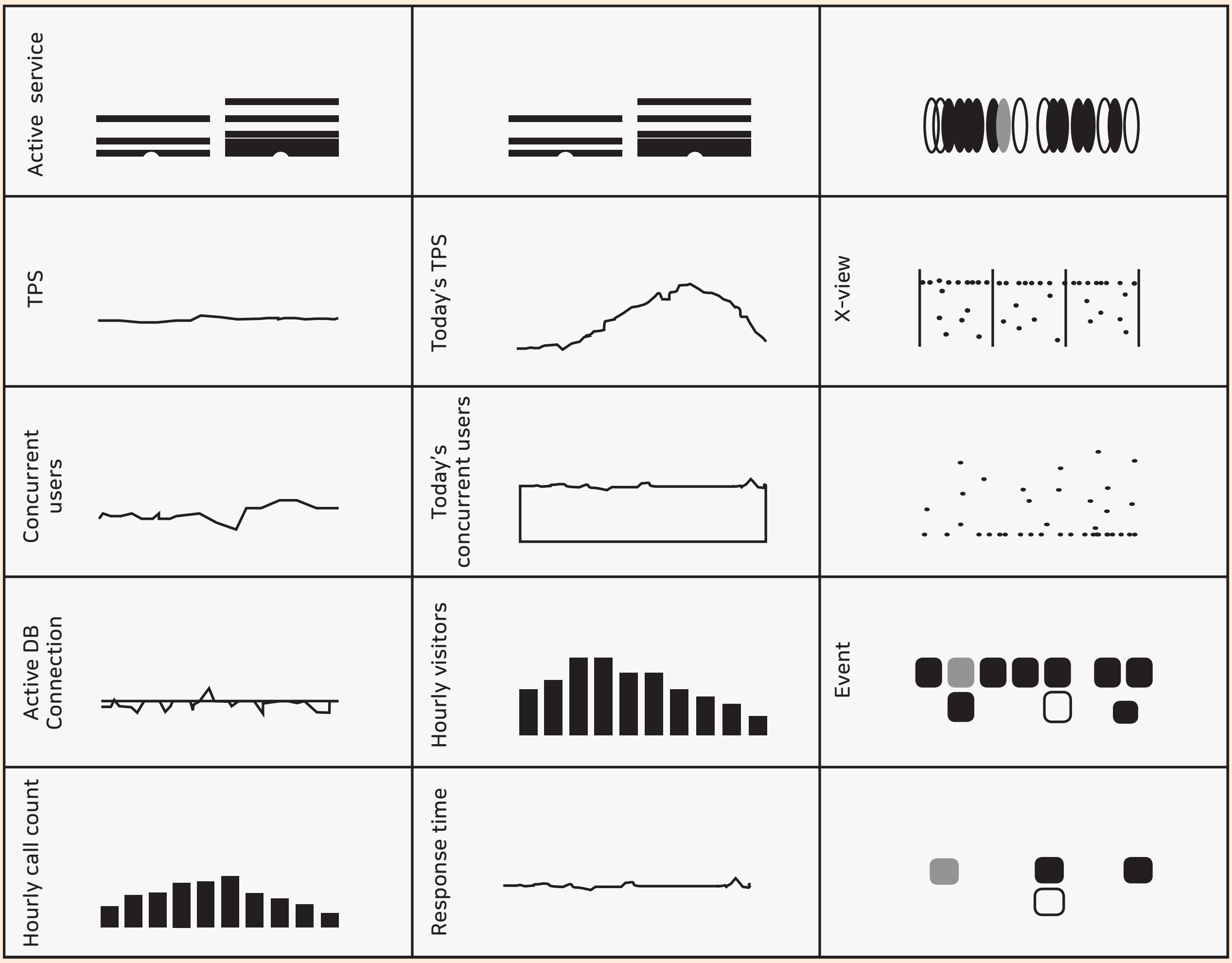
CASE STUDY OF ADOPTING JENNIFER

FANMAUM

FANMAUM, WE PROVIDE A REALIABLE SERVICE TO 1.5 M USERS.

Written by Unyong, Park,
Fanmaun developer

Published by JENNIFERSOFT



JENNIFER®

About FANMAUM

Industry:

Entertainment. FANMAUM provides crowdfunding to celebrity fans.

Key Performance Indicators:

- Two services
- 1.5 m users
- Operates four or five servers
- A maximum 262 TPS a day
- When implementing an event once a month
- 8,000–10,000 users can access it simultaneously
- 1,338 TPS calling occurs

FANMAUM provides two services to 1.5 million users. Before publishing its second service early this year, the daily TPS was recorded at up to 60. After launch, the TPS went up to 262 per day as of Oct. 2017. The company runs an event on a monthly basis, and a maximum of 1,338 TPS occurs during the event. It operates four or five servers that 8,000–10,000 users access simultaneously during the event.

When did you begin using this product?

On June 12, 2017.

What did you monitor?

I have been monitoring WAS and DB.

Name of the interviewee:

Unyong, Park, on the development team.

Why JENNIFER? :

I was looking through a few other PHP monitoring solutions to use; however, they were not convenient to use as they do not have enough support for the GUI. On top of that, given that a server is located locally, using a foreign-based solution would slow down the system. For this reason, I decided to use JENNIFER. I have been using it for a while, and I find myself really loving this solution.

Here is a brief list of its merits:

① Users can easily check whether a server is working.

The first thing that came into my mind when adopting the solution was, "The server is really working hard." Before using it, I noticed the operating status of the server from the CPU usage, memory usage and network usage. However, after using JENNIFER, I can instantly check information such as the number of TPS or how many active services are working.

In addition, even if the resources of the server were sufficient, sometimes, the response time would become sluggish, or the active services would be full. When this happened, I did not know what caused the problem. After using JENNIFER, I could just go to the Analysis and move to the application status to check what was slowing the response time.

② I am able to check the traffic density in hours in real time. As a result, it is possible to check the time load.

While monitoring through JENNIFER, I noticed that FANMAUM service starts slowly increasing from 06:00 to 08:00 in the morning and reaches a high record between 21:00 and 22:00. We were able to respond to the peak time by increasing the number of servers.

What are the major improvements? :

2

JENNIFER Case Study : FANMAUN

JENNIFER

Before using JENNIFER, the average response time to identify the root cause of the problem was about 2 to 3 hours when the server went down. After adopting JENNIFER, however, the average response time decreased to less than 10 minutes. FANMAUM is able to check the server problems on JENNIFER and fix them. The company had to depend on the intuition of its developers to find the location of the load before, but now, it can provide more effective services by identifying the location of the problem and fixing it.

① Improved productivity for monitoring and problem solving.
The FANMAUM developing team has been working according to the lean development method, which only requires a small scale developing team. As such, it lacks time for development alone, and the team experienced a few server errors. For instance, after an update, monitoring became unavailable. After using JENNIFER, all these problems went away because they could instantly recognize any errors or loads using a real-time monitoring window. As a result, the response time was remarkably enhanced.

② Helps performance-tuning.
We usually had to develop under time pressure and we experienced a lack of service performance. For FANMAUM, the performance tuning of the application status, one of JENNIFER'S analysis features, was a great help. In the middle of our operating services, we were able to observe areas that required tuning and respond, enhancing our overall performance.

③ Collect the data necessary to manage our server resources.
An important mission for start-ups is operating a system under a tight budget. FANMAUM handles all (over 200 million calls a month, 15 TB Bandwidth) with less than 100,000 won. One of our biggest concerns was how to upgrade the server resources in line with upcoming events or updates. Using JENNIFER, we can see the usage of the server or the scale of the load and understand when the server resources needed to be upgraded, allowing us to easily communicate with management.

FANMAUM's monitoring service :

3

JENNIFER Case Study : FANMAUN

Application Performance Management

FANMAUM's monitoring service is an API service application where celebrity fans can see the images of celebrities they like. As a profit structure under the current system does not exist, it is important to create profit from advertisement.

Due to the limited number of developers, we could not adopt the MSA, and the entire API is working in the web server. As such, if a single API stops from an error, the number of stacks starts piling up. Increased stacks lead to an increasing number of web servers while loading the DB server at the same time. In the blink of an eye, a risky chain reaction could occur, including errors on the web server, DB server and cash server.

If, due to a server error, a company failed to link users to an ad webpage, profit loss could occur. For instance, if a single API service has an error, increasing the number of web servers up to five, a huge loss could occur, including the DB used during the time of the problem, five added web servers, and advertising revenue loss due to missed user connections.

Here is the simple assumption:

1) If an error occurs for 5 hours
: 25% of the expected profit is lost.

2) If an error occurs for 5 hours
during the peak time,
: 40% of the expected profit is lost.

If an error occurs for 5 hours on a server where profit is relatively lower than that of the main server, but it was running continuously and a small profit could have been earned, about 25 percent of the expected profit could be lost. The assumption above is made under the premise that the usage is identical over a 24-hour period. Given that frequently occurring errors could occur during the peak time, a minimum amount of loss is likely to exceed 40 percent for 5 hours.

In addition, an API server is working on a single web server machine divided into a vhost environment. If an error occurs on the API service, this problem could influence an existing FANMAUM webpage. As a result of influencing the main service, the amount of loss could be a few 100,000 won to over millions of won.

The type of monitoring tool FANMAUM uses and what could be done better :

4

We have used CloudWatch and StatusCake for monitoring instead of adopting a separate monitoring tool. For CloudWatch, except when an error occurs on the system itself, we have had to do additional tasks to create an alert. This is not convenient.

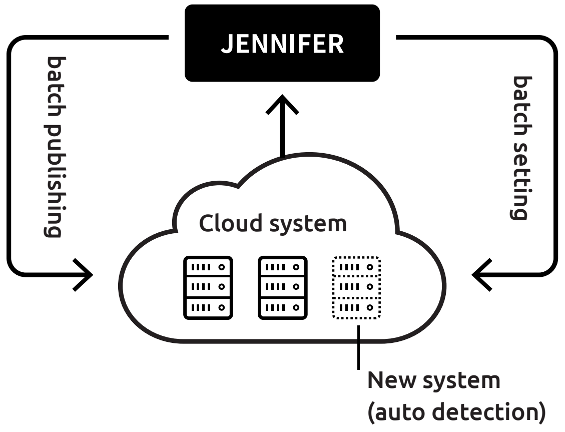
For StatusCake, it is simply a tool for checking a web page. It only calls every 15 minutes. Therefore, it was difficult to notice errors right after they occurred. Responding to errors as fast as we can is much more critical than making a notification. Both services were not of any help at all.

How to use JENNIFER in detail and what it has solved :

5

It was quite simple to use JENNIFER.
I installed the remote agent, AMI, and added the Auto Scale to a group. When a machine is launched, monitoring is automatically connected to the services. It was simple and convenient. Among the various log analysis tools I have used, JENNIFER is the most easy to use and install. In addition, JENNIFER allows me to check the point of error, see the most frequently called files, and display the result by time per transaction.

[Using JENNIFER under the cloud]



I have not used the X-View much. However, I did get a lot of help on the application status page.

Here are the things we wanted to check:
- How frequently is the API called, and how much delay time occurs on average?
- At what time does the API call occur and what is the delay point of the call?
- How did the daily system work and what was the frequent density and peak time?

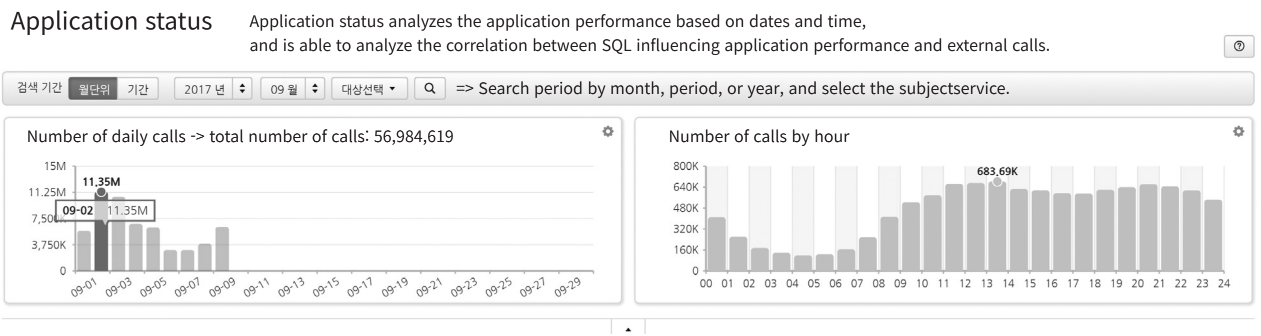
① Application status on JENNIFER
- Application status allows us to see the usage trend (Figure 1).
- Application status not only provides the usage trend by hour for related pages but also shows traffic density by API (Figure 2).
- Once the name of the API had been clicked, I could check the SQL query and usage time for this traffic.

②“Daily system performance” page :
whenever I need to review the daily usage (for TPS, daily traffic amount, peak time, or comparison with previous dates), the daily system performance page was most useful.

③ Only three pages (dashboard system manager template, application status, and daily system performance page) were used on JENNFIER for FANMAUM due to the limited timeline. Identifying the slowed traffic in real-time through the dashboard was a great help for monitoring the FANMAUM service.

④ For FANMAUM, JENNIFER allowed us to considerably shorten the process of Identifying the cause of error and most frequent reason for delay. As a result, the average response time to errors decreased from 2 to 3 hours to less than ten minutes. While the service is running, JENNIER forecasts an error by checking the bottleneck or point where performance slows down in advance.

[Figure 1. Usage trend on Sept. 2.]



Application status

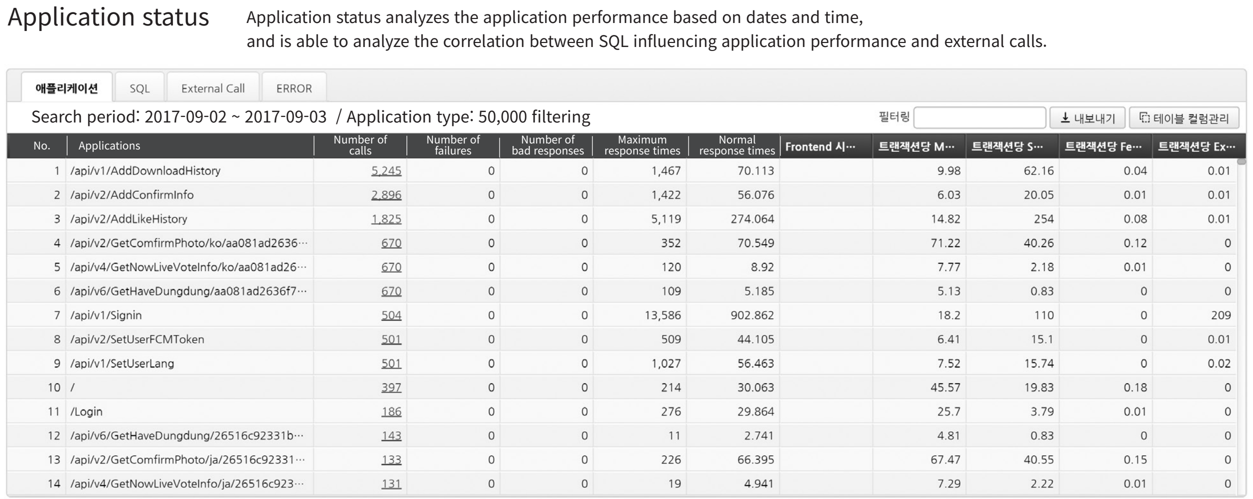
Application status analyzes the application performance based on dates and time, and is able to analyze the correlation between SQL influencing application performance and external calls.

=> Search period by month, period, or year, and select the subjectservice.

Number of daily calls -> total number of calls: 56,984,619

Number of calls by hour

[Figure 2. API traffic density.]



Application status

Application status analyzes the application performance based on dates and time, and is able to analyze the correlation between SQL influencing application performance and external calls.

Search period: 2017-09-02 ~ 2017-09-03 / Application type: 50,000 filtering

No.	Applications	Number of calls	Number of failures	Number of bad responses	Maximum response times	Normal response times	Frontend 시···	트랜잭션당 M···	트랜잭션당 S···	트랜잭션당 Fe···	트랜잭션당 Ex···
1	/api/v1/AddDownloadHistory	5,245	0	0	1,467	70.113		9.98	62.16	0.04	0.01
2	/api/v2/AddConfirmInfo	2,896	0	0	1,422	56.076		6.03	20.05	0.01	0.01
3	/api/v2/AddLikeHistory	1,825	0	0	5,119	274.064		14.82	254	0.08	0.01
4	/api/v2/GetComfirmPhoto/ko/aa081ad2636···	670	0	0	352	70.549		71.22	40.26	0.12	0
5	/api/v4/GetNowLiveVoteInfo/ko/aa081ad26···	670	0	0	120	8.92		7.77	2.18	0.01	0
6	/api/v6/GetHaveDungdung/aa081ad2636f7···	670	0	0	109	5.185		5.13	0.83	0	0
7	/api/v1/Signin	504	0	0	13,586	902.862		18.2	110	0	209
8	/api/v2/SetUserFCMToken	501	0	0	509	44.105		6.41	15.1	0	0.01
9	/api/v1/SetUserLang	501	0	0	1,027	56.463		7.52	15.74	0	0.02
10	/	397	0	0	214	30.063		45.57	19.83	0.18	0
11	/Login	186	0	0	276	29.864		25.7	3.79	0.01	0
12	/api/v6/GetHaveDungdung/26516c92331b···	143	0	0	11	2.741		4.81	0.83	0	0
13	/api/v2/GetComfirmPhoto/ja/26516c92331···	133	0	0	226	66.395		67.47	40.55	0.15	0
14	/api/v4/GetNowLiveVoteInfo/ja/26516c923···	131	0	0	19	4.941		7.29	2.22	0.01	0

Unyong, Park,
Fanmaum development team

How JENNIFER changed your way of working ?

Before using JENNIFER, we were frequently asked about how frequently we used a server or whether this was an appropriate way to operate a server at this cost. The very first thing I did after adopting JENNFIER was to share the dashboard of JENNFIER with the management and print daily system performance pages every day. By doing so, we were able to communicate about the cost side of services and how to efficiently use them.

Furthermore, we are now able to check problems more intuitively. We set up a communication policy. Once an error occurs, an operating team checks the dashboard and contacts the developing team right away to share the delayed API. As a result, the developing team can find the problem in a timely manner and explain it in detail. Previously, we would text or call and say, “The server is not working.”
Now, however, we say, “The API is in red, it does not go away, it just stays for a few seconds. Other traffics are not working.”

How would you like to use JENNIFER down the road?

I would love to use it continuously and for a longer time. On top of new services, we plan to use it on existing services, and we are also preparing to monitor it under an MSA environment.

This document is a report edited and published by JenniferSoft about the questionnaire to companies using JENNIFER.