








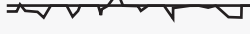







팬마음, 150만 사용자에게 안정적 서비스를 제공하게 되다

제니퍼 도입사례

팬마음

팬마음 개발팀 박운용 작성
제니퍼소프트 위음

Active service 		
TPS 	Today's TPS 	X-view 
Concurrent users 	Today's concurrent users 	
Active DB Connection 	Hourly visitors 	Event 
Hourly call count 	Response time 	

팬마음 기업정보


산업분야


엔터테인먼트, 팬마음은 연예인 팬들을 대상으로 클라우드 펀딩 서비스를 제공하는 서비스다.


주요 성능 지표


 2 개 서비스


 150 만 사용자

 4~5 개 서버운용

 262 TPS 일 최대

 월 1회 이벤트 진행시

 8 천 ~ 1 만 동시 접속

 1338 TPS 호출발생

팬마음은 두 개의 서비스를 제공하고 있으며 150 만 사용자가 사용하고 있다. 올해 초 두번째 서비스가 공개되기 이전에는 일 최대 TPS 가 60 정도 발생했으며, 신규 서비스 런칭 이후 2017년 10 월 현재 일 최대 262 TPS 까지 발생하고 있다. 매월 이벤트를 진행하고 있으며, 이벤트 진행 시 최대 1338 TPS 이상의 호출이 발생하고 있다. 웹서버는 4~5 개가 운용되고 있으며 이벤트 기간에는 평균 동시 접속자수가 최대 8 천 명에서 1 만 명이 넘게 사용하고 있는 서비스다.

제니퍼 사용 시작일

2017 년 6 월 12 일부터

무엇을 모니터링했는가?

WAS 와 DB 를 모니터링했다.

왜 제니퍼인가? :

다른 PHP 모니터링 솔루션을 몇 개 찾아보고 검토했으나, GUI 지원의 불편함, 서버 자체가 국내에 있는데 해외 솔루션을 쓰기엔 다소 느려지는 부분 등 여러 가지 문제가 발생했다. 하여 국내에서 서비스하는 제니퍼를 사용하게 되었다. 지속적으로 사용하다 보니, 정말 여러가지로 마음에 드는 솔루션이었다.

1

제니퍼 장점을 정리하자면 다음과 같다.

① 서버가 일하는지 쉽게 볼 수 있다.

솔루션을 도입하면서 가장 처음 든 생각은 “아 서버가 열심히 일하는구나”였다. 솔루션 도입전에는 CPU 사용량과 메모리 사용량, 네트워크 사용량만 가지고 서버의 운용 현황을 판단했다면 솔루션 도입 후 액티브 서비스가 몇 개가 도는지, TPS 는 얼마나 되는지 등의 정보를 즉각적으로 알 수 있었다. 또한, 서버의 자원이 남는데도 불구하고 응답 속도가 늦는다면 액티브 서비스가 풀(Full)로 차버리는 현상이 발생하면 원인을 알 수가 없어 답답했는데, 제니퍼를 사용하고 난 후 응답시간이 느린 부분을 바로 확인(Analysis -> 애플리케이션 현황으로 이동)할 수 있어 매우 유용했다.

② 시간대별 호출량을 실시간으로 볼 수 있어 부하 시간을 체크해 대응할 수 있게 되었다.

제니퍼를 통해 모니터링 하여 보니, 팬마음 서비스는 오전 06 시부터 08 시까지 그리고 점차 증가해 21 시에서 22 시 사이에 최고치를 찍는다는 것을 알 수 있었다. 우리는 해당 시간에 서버를 증설하는 방식으로 서버의 대응력을 올릴 수 있었다.

주요 개선된 점은 무엇인가 :

2

제니퍼 도입 사례 : 팬마음

JENNIFER

도입 이전에는 서버 장애 발생 시 평균 대응 시간이 2-3 시간이었다면, 제니퍼를 도입한 이후 대응 시간은 10 분 이내로 줄어들었다. 팬마음은 서버 장애를 제니퍼 화면을 통해 바로 확인하여 수정할 수 있게 되었고 개발자의 직관에 의지해서 처리하던 부하 지점을 찾는 것 또한 실제 문제 있는 지점을 찾아 수치적으로 수정하게 되어 훨씬 더 실질적인 서비스를 제공할 수 있게 되었다.

① 모니터링과 장애대응에 대한 생산성의 향상

팬마음 개발팀은 소규모 개발팀이 할 수 있는 린 개발 방법론대로 개발을 시도하고 있어 대부분 개발만 해도 시간이 부족하다. 그러다 보니 업무가 바빠 몇 번 서버 장애가 오기도 했었고 신규 기능 업데이트 이후 모니터링이 어려운 문제가 종종 발생했다. 제니퍼를 도입하고 나서 이런 문제가 해결되었는데, 모니터링에 실시간 모니터링 창을 띄워놓고 보면서 장애 발생이나 부하 상황을 실시간으로 알 수 있게 되면서 대응 속도가 눈에 띄게 향상되었다.

② 성능 튜닝 도움

주로 짧은 기간에 개발을 하다 보니 서비스 성능에 대한 부분이 다소 부족하게 개발되는 경우가 있었다. 팬마음은 제니퍼 기능 중 분석 기능인 애플리케이션 현황에서 주로 성능 튜닝에 도움을 받았다. 서비스가 운영되는 중간에도 튜닝이 필요한 곳을 바로 파악해 성능에 대응할 수 있게 되었다.

③ 서버 자원 관리에 대한 데이터 수집

스타트업의 숙명 중 하나는 저렴한 비용으로 시스템을 운용해야 하는 것이다. 팬마음은 주로 10 만원 이내로 전체 서비스의 트래픽(1 개월 2 억 건 이상 호출, 15TB Bandwidth)을 감당하다 보니 앞으로 진행하게 될 이벤트나 업데이트에 맞춰서 서버 자원을 어떻게 업그레이드 해야 하는지 고민이 많았다. 제니퍼를 사용하며 서버의 사용량이나 부하의 척도를 볼 수 있게 되면서 앞으로 어느 정도 수치가 되면 서버 자원을 업그레이드를 해야 하는지 파악할 수 있게 되었다. 여러가지로 경영진과 이야기하기가 매우 쉽고 편해졌다.

팬마음의 모니터링 대상 서비스 :

3

제니퍼 도입 사례 : 팬마음

Application Performance Management

팬마음의 모니터링 대상 서비스는 연예인들을 좋아하는 팬들을 위한 연예인의 사진을 볼 수 있는 애플리케이션의 API 서비스이다. 이 서비스에는 현재 특별한 수익구조가 존재하지 않기에 광고를 통한 수익 창출이 매우 중요하다.

현재 개발 인력이 부족으로 현재 MSA 를 도입하지 못하고 있기에 웹서버 내에 모든 API 가 작동되고 있다. 그렇기 때문에 만일 API 하나가 문제가 생겨 멈추기 시작하면 금방 스택이 쌓이고, 스택이 쌓이면 웹서버가 증가하며 동시에 DB 서버에 부하를 주면서 순식간에 웹 서버 장애 + DB 서버 장애 + 캐시 서버 장애 까지 발생하는 연쇄 반응을 일으킬 수 있는 위험한 상황이 발생할 수 있는 환경이다.

서버 장애가 발생하여 사용자들이 광고를 볼 수 있는 페이지로 연결되지 못한다면 결국 서버 장애 (비용 상승)가 곧 수익 저하로 이어질 수 있다.

예를 들어 API 서비스 하나가 문제가 생겨 웹서버가 5대까지 늘어난다면 장애 시간 동안 사용되는 데이터베이스 비용 + 웹서버 5 대가 늘어난 비용 + 사용자들의 접속불가로 광고 수익 미발생이라는 큰 손실이 발생할 수 있다.

단순 가정을 해본다면 다음과 같다.



1) 장애 발생 5시간 : 예상 수익금 25% 손실



2) 사용량이 높은 피크시간 장애 5시간 : 예상 수익금 40% 손실

메인 서비스보다 수익은 적지만 지속해서 서비스를 운영할 수 있는 서버비와 약간의 이윤 추구가 가능한 이 서비스에서 장애가 5 시간 동안 발생한다면 대략 예상 수익금의 25% 가량 손실이 예상된다. 이걸 24 시간동안 사용량이 같다는 전제하에 작성된 것이지만, 주로 발생하는 장애 시간은 피크시간일 가능성이 높으므로 5 시간 장애 발생 시 최소 손실금액은 40% 를 넘길 것으로 예상하고 있다.

또한, API 서버는 앞서 언급한 것과 같이 기본적으로 vhost 환경으로 나누어진 한 개의 웹서버 머신에서 작동하고 있으므로, 이 API 서비스가 장애가 발생하면 기존 운영 중이던 팬마음 사이트까지 장애가 발생하는 큰 문제가 발생한다. 본 서비스까지 발생한 장애에 따른 손실 비용은 단순 몇 십만원에서 몇 백만 원까지 규모가 증가해 빠른 대응이 필요한 상황이다.

팬마음에서 사용하는 모니터링 툴 종류와 이를 통해서도 부족한 점 :

기본적으로 별도의 모니터링 툴을 도입하지는 않고 CloudWatch 와 StatusCake 를 사용해 모니터링하고 있었다. 아쉬웠던 것은 CloudWatch 는 시스템 자체의 장애가 발생하는 경우가 아니면 알람을 만들기 위해 별도의 작업을 더 해야 하는 불편함이 있었다. StatusCake 란 툴의 경우 단순 웹페이지 체크 툴이기 때문에 15 분에 한 번 호출하는 것이 전부여서 장애 발생 이후 바로 알기가 어려웠다. 어쨌든 Notification 이 중요한 것이 아니라 최대한 빠른 장애 대응이 첫 번째 우선순위가기 때문에 이를 추적해야 하는 상황에서 두 개의 서비스 모두 전혀 도움이 되지 않았다.

4

제니퍼 도입 사례
: 팬마음

JENNIFER

제니퍼의 구체적인 사용 방법과 제니퍼를 통해 해결된 사항 :

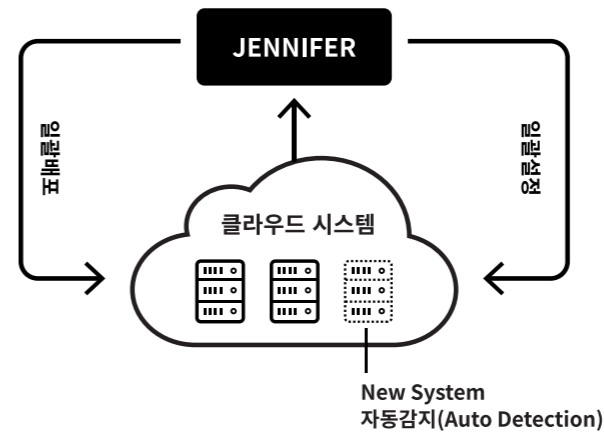
제니퍼를 사용하는 방법은 아주 간단했다. 원격 에이전트를 설치하고 AMI 를 생성한 후 Auto Scale 그룹에 추가했으며, 이후 모니터링은 머신이 런칭되면 자동으로 연결되어 서비스 이용이 가능했으니 매우 심플하고 편리했다. 여러 가지 로그 분석툴을 사용해 봤으나 제니퍼의 간편한 설치와 사용은 최고라고 말할 수 있다. 또한, 실시간으로 발생하는 장애 지점의 확인, 자주 호출하는 파일의 확인, 트랜잭션당 소요 시간을 기준으로 정렬해서 볼 수 있는 방법은 우리에게 정말 많은 도움이 되었다.

5

제니퍼 도입 사례
: 팬마음

Application Performance
Management

[클라우드 환경 안에서의 제니퍼 사용]



엑스 뷰(X-view)는 생각보다 많이 사용하지는 않았지만, 애플리케이션 현황 페이지에서 정말 많은 도움을 받았다. 좋았던 부분에 관해서 설명하고자 한다.

우리가 보길 원했던 부분은 다음과 같다.
첫 번째 : API 가 얼마나 호출되며, 평균적으로 얼마나 많은 지연시간이 발생하는가.
두 번째 : 어느 시간에 API 콜이 가장 많이 발생하며, 이 콜에 대한 지연 지점이 어디인가.
세 번째 : 일일 시스템이 어떻게 일을 했으며, 발생한 호출량과 피크 타임

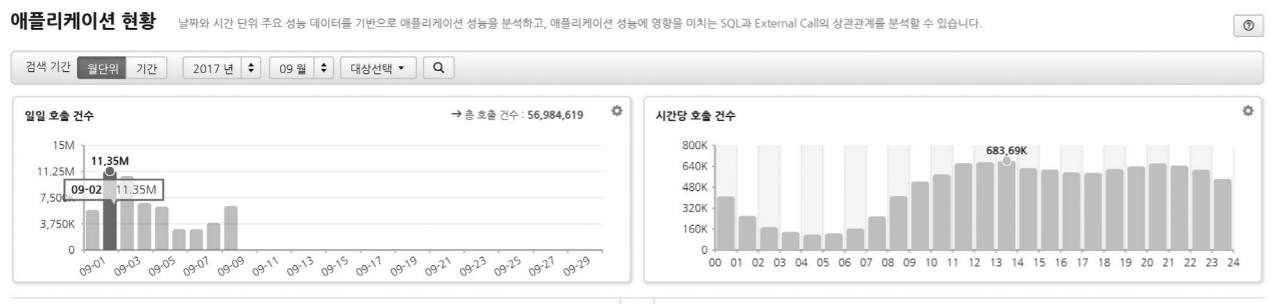
① 제니퍼를 통해 사용할 수 있었던 기능인 ‘애플리케이션 현황’ 페이지 : 먼저 애플리케이션 현황 페이지를 통해 사용량 추이를 확인할 수 있었다. [그림 1] 해당 페이지에서 사용량 추이 자체를 시간당으로 보는 것을 제공하는 것에 그치지 않고 이어서 API 별 호출량을 보여준다. [그림 2] 이어서 해당 API 이름을 클릭하면 이 호출에서 발생한 SQL 쿼리와 사용시간 등을 확인할 수 있다.

② ‘일일 시스템 성능’ 페이지 : 매일 발생하는 사용량에 대한 평가(TPS, 일일 호출 건수, 피크타임, 이전 날짜와 비교 등)가 필요한 시점에서는 언제라도 일일 시스템 성능 페이지가 최고의 역할을 해줬다.

③ 팬마음은 시간적 여유가 부족해 제니퍼를 사용하면서 거의 3 개의 페이지만 사용했지만 (대시보드 - 시스템 관리자 템플릿, 애플리케이션 현황, 일일 시스템 성능 페이지), 대시보드를 통해서 실시간으로 느껴지는 호출에 대한 기능은 팬마음의 서비스를 모니터링을 하는 부분에 있어 정말 큰 도움이 되었다.

④ 팬마음은 제니퍼를 도입하고 난 후 평균적인 장애 대응시간(2~3 시간)에서 대부분 자체 사유인 장애 원인 확인 과정을 극단적으로 단축함으로써 실제 대응 시간을 10 분 내외로 줄일 수 있었다. 또한 평소 서비스가 운영 중인 상황에서도 지속해서 API 응답 시간을 확인해 병목 지점이나 성능 저하 지점을 미리 파악해 장애를 예측하고 발생하지 않도록 선처리가 가능했다.

[그림 1] 9 월 2 일 사용량 추이



[그림 2] API 호출량

애플리케이션 현황 날짜와 시간 단위 주요 성능 데이터를 기반으로 애플리케이션 성능을 분석하고, 애플리케이션 성능에 영향을 미치는 SQL과 External Call의 상관관계를 분석할 수 있습니다.

번호	애플리케이션	호출 건수	실패 건수	Bad Respons...	최대 응답시간 ...	응답시간 표준...	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



팬마음 개발팀 박운용

제니퍼 사용 후 변화된 업무 처리 방법

제니퍼를 도입하기 이전에는 ‘도대체 서버를 얼마나 쓰는가?’와 ‘이 비용을 내고 서버를 운영하는 것이 맞는가?’에 대한 질문을 많이 받고 있었다. 제니퍼를 도입하며 가장 먼저 한 일은 경영진에게 제니퍼 대시보드를 공유하고, 일일 시스템 성능 페이지를 매일 출력해 전달함으로써 서비스의 비용적인 부분과 효율적인 사용에 대해서 커뮤니케이션을 할 수 있게 되었다는 것이다.

또한, 직관적인 문제 확인이 가능하게 되어 장애가 발생하면 운영팀에서 바로 대시보드 확인 후 개발팀에 연락하면서 지연되는 API 를 말해주도록 커뮤니케이션 방침을 세우면서 개발팀은 보다 빨리 문제를 찾아(기존에는 전화/문자로 ‘서버가 안 돼요’라고 전달했다면 ‘○○○ API 가 빨갭게 떠 있고, 없어지지 않으며 몇 초 동안 켜져있고, 다른 호출이 작동하지 않아요’라고만 말했다.) 디테일하게 전달 할 수 있게 되었다.

제니퍼를 통한 향후 활용 계획이 있으신가요?

지속적으로 그리고 오래도록 사용하고 싶다. 현재 사용 중인 신규 서비스 외에도 기존 서비스에 대한 적용, 그리고 MSA 환경에서의 모니터링도 준비하고 있다.



2017 ©JENNIFERSOFT,INC.
All rights reserved.

All trademarks, trade names, service mark and logos referenced herein belong to their respective companies. This document is for your informational purposes only. To the extent permitted by applicable law.

이 문서는 제니퍼를 사용하고 있는 기업을 대상으로 설문 하였던 내용을 제니퍼소프트에서 편집하여 발행한 리포트입니다.