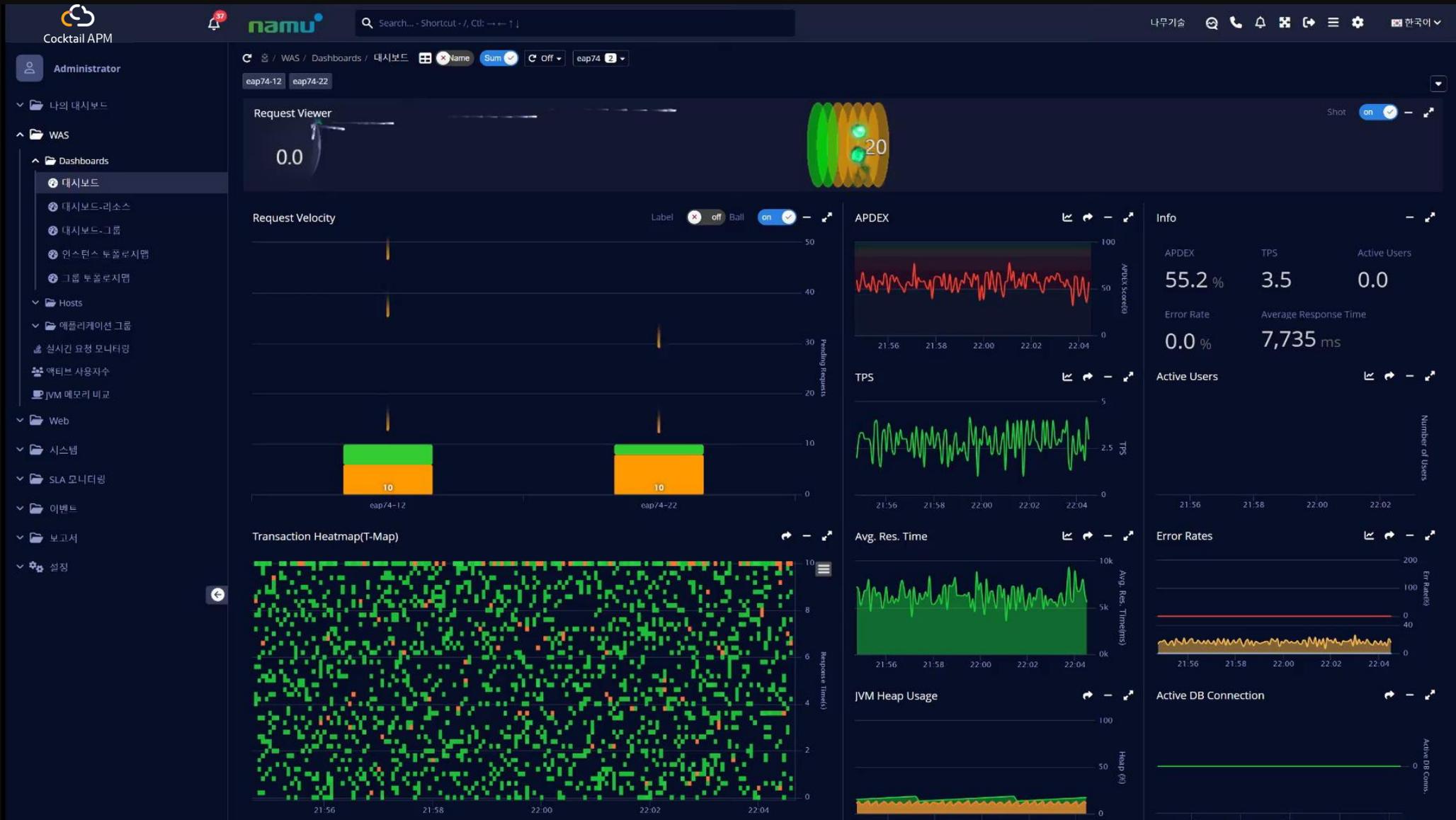


Cocktail APM



Application 중심의 PaaS와 Cocktail APM



주요 사업 영역

Cloud Native 기술을 기반으로 PaaS 구축, Cloud 전환 & APM을 통한 성능/안정성 확보



사업 영역 별 주요 실적

2021년 Red Hat PaaS Business Partner 1위 기업 , AWS Migrations 등 Cloud Native 전문 기술 기업

구분	고객사	프로젝트명	운영 환경	수행 시기
U2L 전환	 건국대학교병원 KONKUK UNIVERSITY MEDICAL CENTER	병원정보시스템 주전산기 성능 모니터링	Red Hat Linux	2017년
클라우드 환경		eCOM / POS 시스템 성능 관리	Amazon AWS	2018년
		MyACUVUE 모바일 및 홈페이지 서비스 성능 관리	Amazon AWS	2018년
	 한국교육학술정보원	디지털 교과서 플랫폼 서비스 클라우드	G-Cloud	2018년
PaaS 환경		온라인 콘서트 동시 접속 100만명 지원	Cocktail Cloud	2022년
		Life Platform 프로젝트 WAS 모니터링 솔루션	OpenShift	2018년
		온-나라 문서 2.0 확산 및 고도화	OpenShift	2018년
		SET 시스템 PaaS 플랫폼	SDS PaaS	2017년
		PaaS IT 모니터링	OpenShift	2017년
오픈소스 환경		머니트레이너 시스템 (계정계, 정보계)	RHEL./JBoss	2017년

Cloud Native

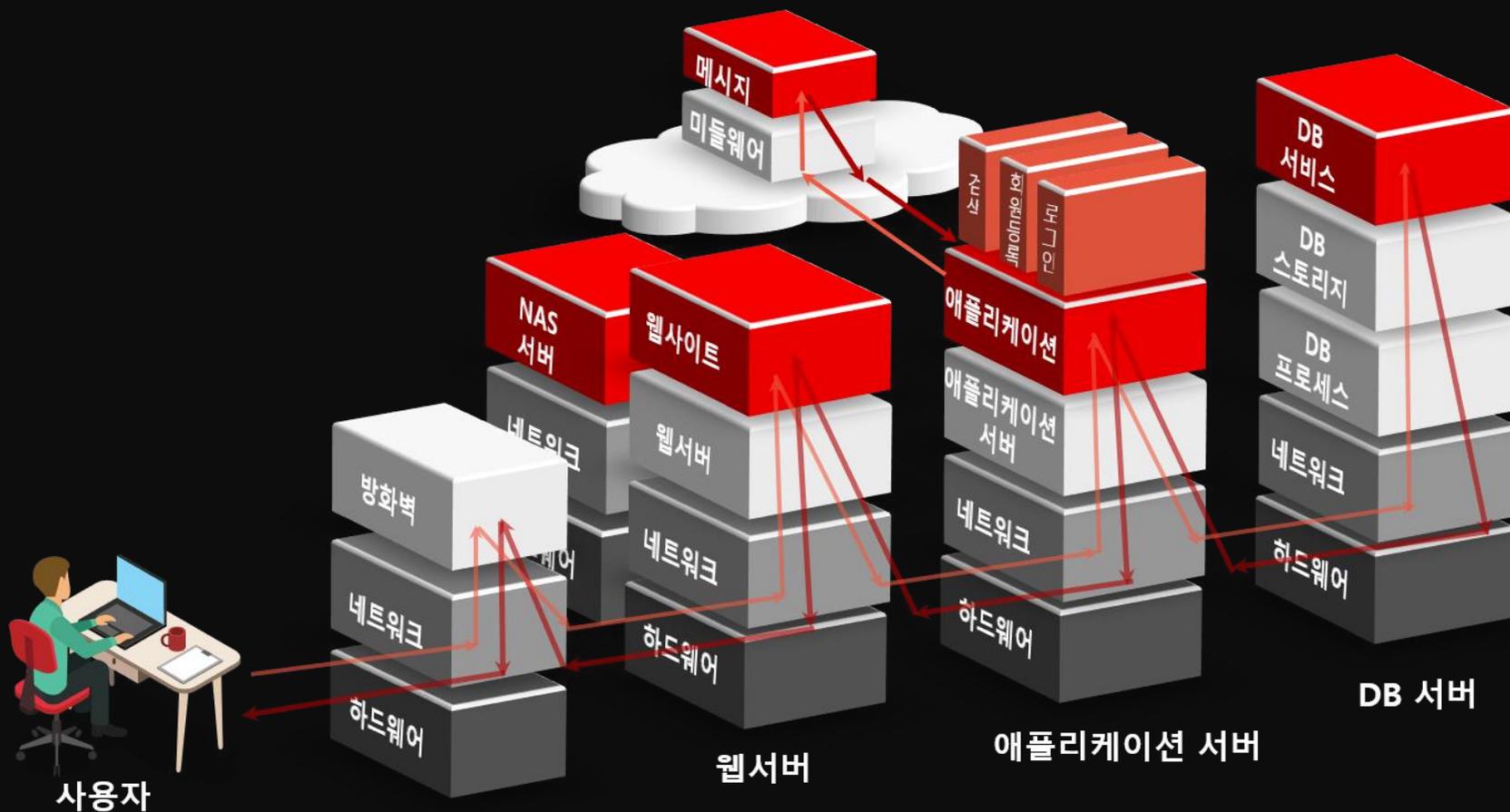


APM
(Application Performance Management)



미들웨어는 시스템 장애의 관문이자 시작점

- 데이터베이스가 50% 느려진다면 사용자 응답시간은 어떻게 될까요?
- DB 테이블 변경으로 SQL 에서 오류로 인하여 페이지가 오류가 난다면?



[긴급] 비정상 상황 발생 - 대시보드



[심각-CRITICAL] 'Worker Usage %' (평균값: 98.9)이 심각(CRITICAL) 임계값 '95'을 넘었습니다.
발생에이전트 : apache@EBS-OC-PROD2-WEB05[172.17.11.35]

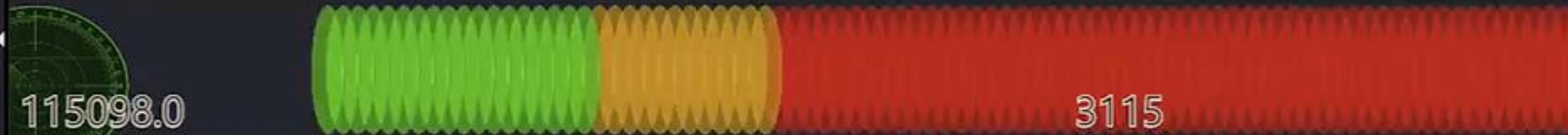
클릭하여 상세한 정보를 확인하세요... 10s



[심각-CRITICAL] 'Worker Usage %' (평균값: 99.95)이 심각(CRITICAL) 임계값 '95'을 넘었습니다.
발생에이전트 : apache@EBS-OC-PROD2-WEB02[172.17.11.5]

클릭하여 상세한 정보를 확인하세요... 10s

Request Viewer



Request Velocity



APDEX



APDEX **93.3 %** **115277.2**

TPS **1071.8** Error Rate **0.1 %**

Average Response Time **2,884ms**

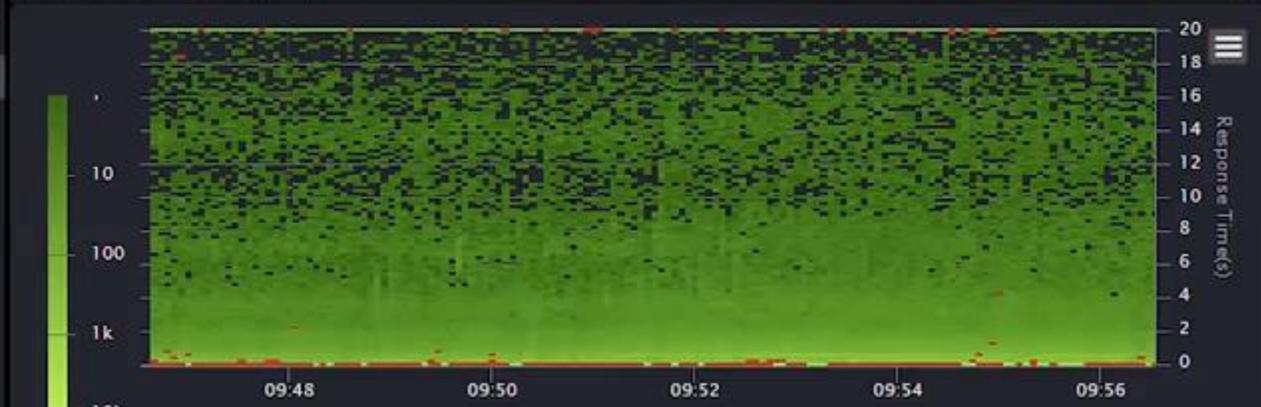
TPS



Active Users



Transaction Heatmap (T-Map)



Avg. Res. Time



Error Rates

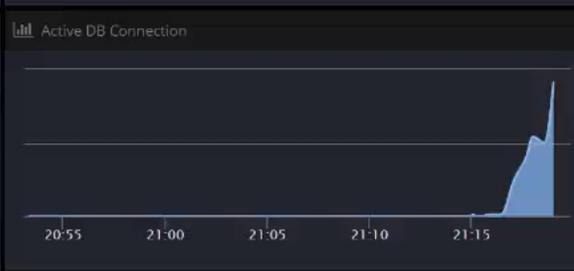
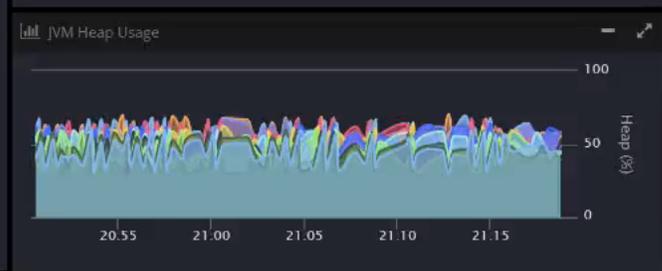
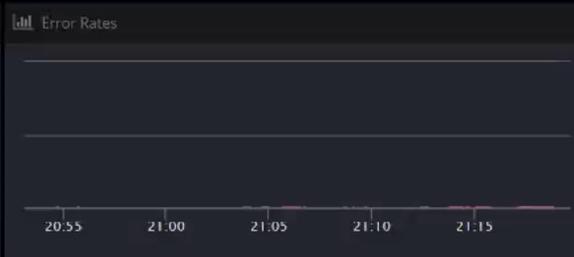
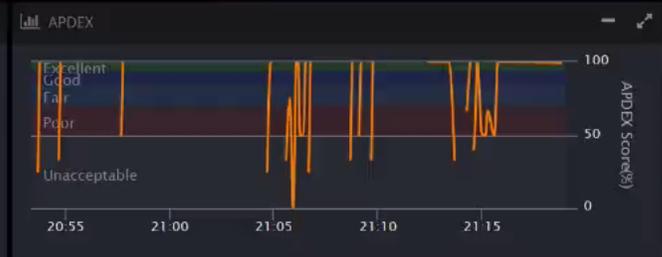


JVM Heap Usage

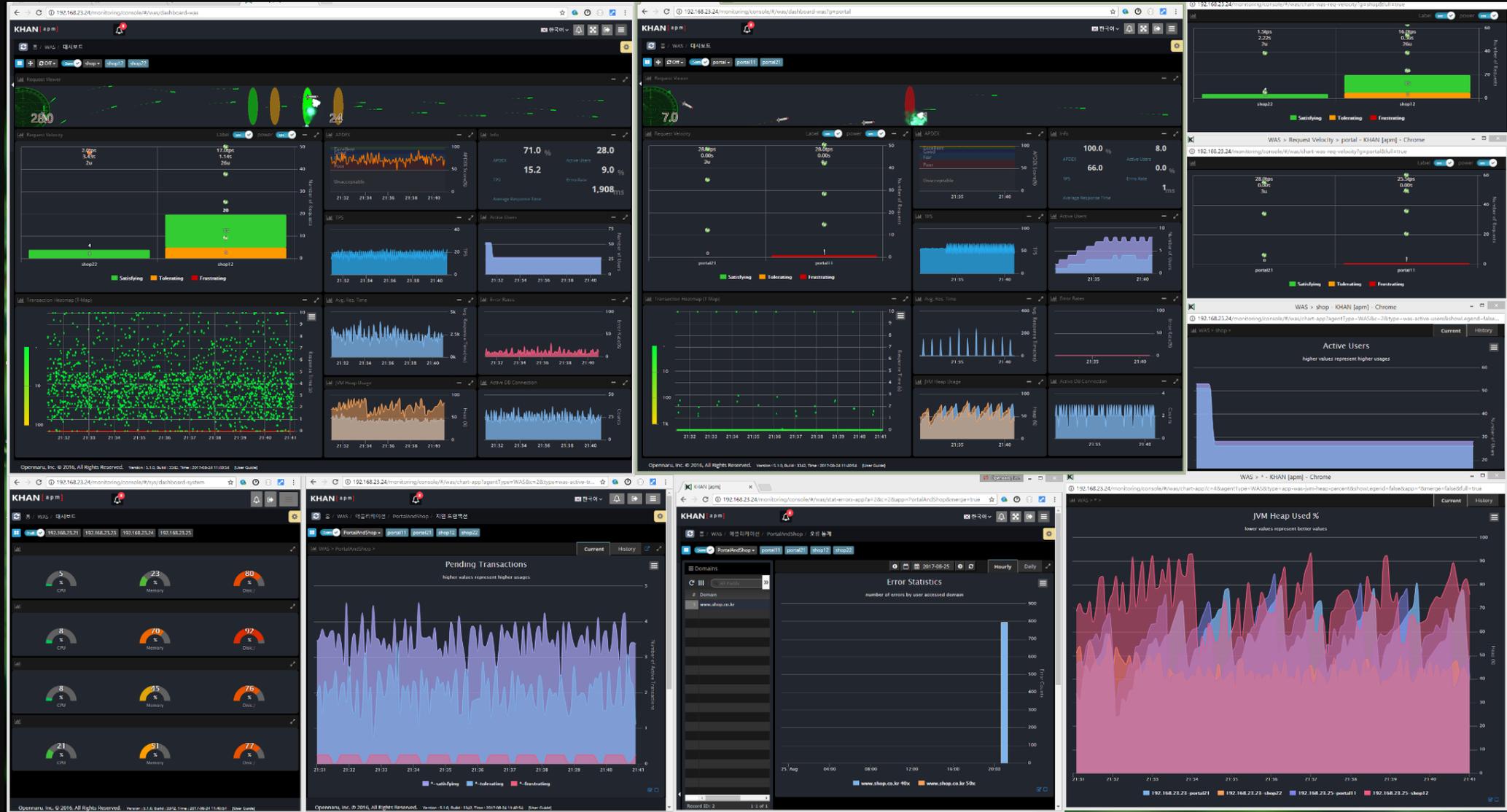


정상 서비스 상황

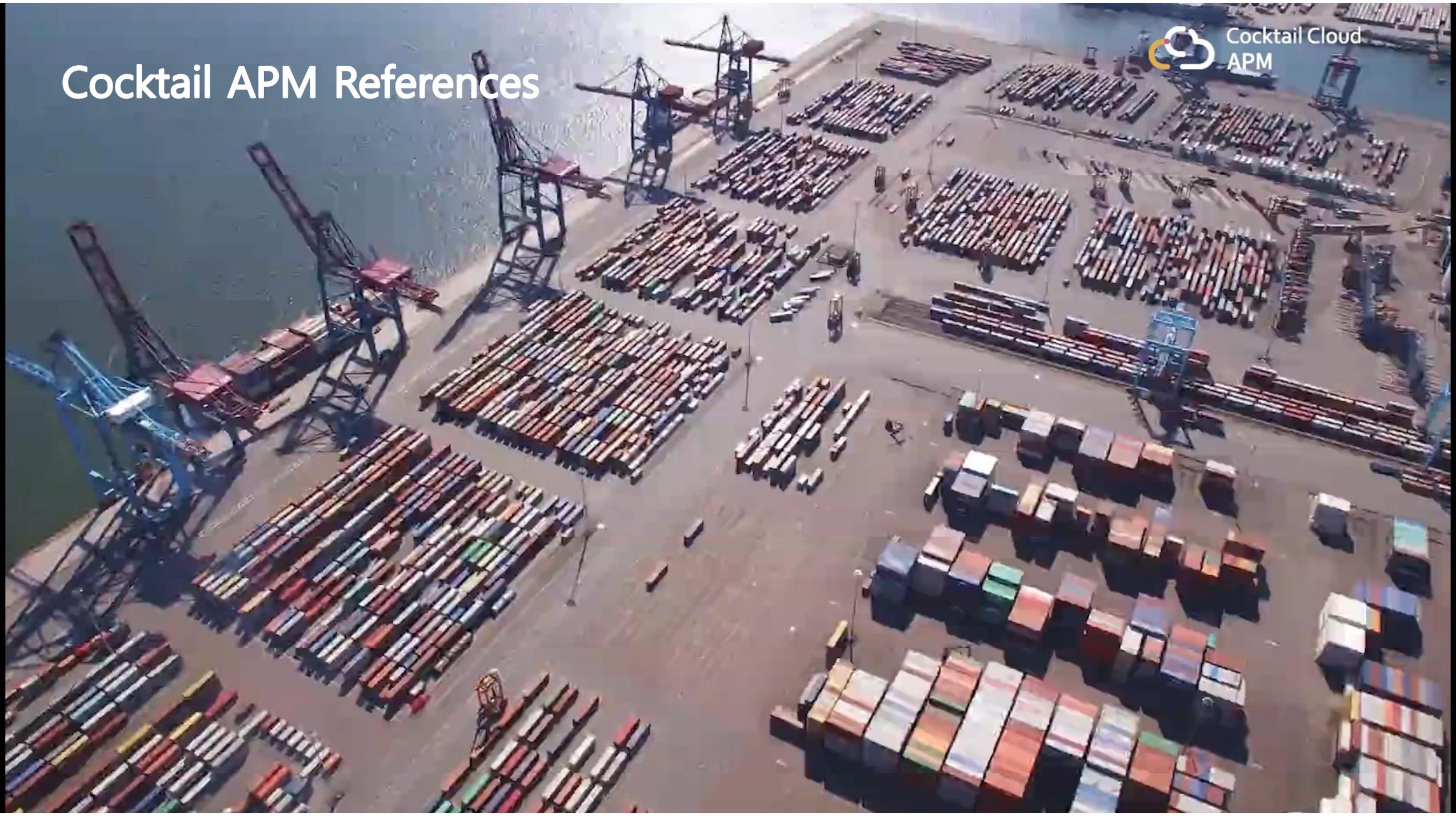
WAS Dashboard | On | Sum | @maru | @maru12 | @maru11 | @maru21 | @maru22 | @maru21 | @maru24 | @maru23 | @maru14 | @maru12 | @maru13 | @maru11 | @maru51 | @maru52



Cocktail APM:: 9 – HTML5



Cocktail APM References



다양한 운영 환경에서 검증된 APM 제품

성과와 장애 관리 도구인 APM은 물리/가상서버, 클라우드 등 다양한 WAS 운영 환경에서 검증되어야 합니다.

- Cocktail APM은 국내 최초로 PaaS와 Kubernetes에서 WAS와 클라우드 인프라 모니터링을 제공하였습니다.
- 중앙부처와 지방자치단체를 비롯한 금융권과 주요 국내 기업에서 Cocktail APM을 사용하고 있습니다.

Cocktail APM & PaaS 9개 이상

주요 지방 자치단체 구축 사례 외 다수

주요 기업 구축 사례 외 다수 10개 이상

주요 공공기관 구축 사례 외 다수 30개 이상

학교 / 병원 외 다수 5개 이상

대규모 PaaS 운영환경에서 검증된 Cocktail APM

국내 공공기관 및 대기업에 도입된 Cocktail APM 주요 구축사례입니다.

<p>발주처</p>				
<p>프로젝트명</p>	<p>온-나라 클라우드 문서 2.0</p>	<p>스마트 플랫폼 구축</p>	<p>차세대 나라통계 시스템 구축</p>	<p>롯데카드 채널계 클라우드 구축</p>
<p>당면 과제</p>	<ul style="list-style-type: none"> 1차 사업 - 성능 및 안정화 이슈 2차 사업 - 나무기술이 성능/장애 해결 	<ul style="list-style-type: none"> 한국항공우주산업의 글로벌 경쟁력 확보에 필요한 스마트플랫폼 구축 	<ul style="list-style-type: none"> 전세계 최초로 클라우드 기반 국가 단위 통계 플랫폼 표준화를 위한 업무 개별서버 255대 규모 	<ul style="list-style-type: none"> 금융권 최초로 채널계 시스템 도입으로 대고객 접점 서비스 국내 최초로 Cocktail APM으로 안정화 지원 및 성능 튜닝
<p>나무기술 역할</p>	<p>PaaS/WAS 성능관리 사업자</p>	<p>PaaS BMT 수행/구축/유지보수</p>	<p>PaaS 구축 및 안정화 지원</p>	<p>APM 공급 및 안정화 지원</p>



- PaaS 문제 발생!! 소방관
- WAS 문제 발생!! 해결사







행정안전부 - 국가 80여 개 중앙부처 그룹웨어(온-나라) 시스템

한국지역정보개발원 온-나라 클라우드 문서 2.0 사업(2018년) - Red Hat PaaS & Cocktail APM



프로젝트명	온-나라 클라우드 문서 2.0
목적	정부부처 20군데와 지방자치단체 약 80군데 등 100만 공무원이 사용하는 문서 결제 시스템
규모	Worker Node 80대 규모
나무기술 역할	PaaS 전문인력 상주와 Cocktail APM으로 성능 관리를 담당하여 안정화 지원

☑ Cocktail APM은 국내 최초, 최다 컨테이너 & 클라우드 환경 지원

법정부 기반 업무관리시스템으로 효율적인 행정 구현

- 행안부, 26개 기관 대상으로 법정부 기반 온-나라시스템 고도화 착수 -

- ☐ 행정안전부(장관 김부겸)는 17일 정부세종컨벤션센터(세종시 소재)에서 중앙부처 및 지자체 온-나라시스템* 담당자 및 관계자 등 70여명이 참석한 가운데 '클라우드(인터넷 기반 정보 통신 자원 통합·공유 서비스) 기반 온-나라시스템 고도화 사업' 착수보고회를 개최하였다.
 - * 온-나라 시스템 : 행정기관의 업무에 대한 문서 작성·검토·결재·등록·공유·공개 등 문서처리의 모든 과정을 기록·관리하는 전자결재시스템
- 이번 고도화 사업은 국무조정실, 금융위원회 등 26개 기관을 대상으로 각 기관별로 보고서 및 문서를 저장·보관하는 기존방식에서 통합저장소(클라우드)에서 공동기간·결제가 가능하도록 하는 사업으로 2019년까지 전 중앙부처에 확산할 계획이다.



행정안전부 보도 자료
법정부 기반 온-나라 시스템 고도화 착수

클라우드 전환 시 노후 장비 교체 대비
2.6배의 비용 효과 발행

클라우드 온-나라 문서2.0
개념도

온나라 클라우드 APM
성능 테스트

[이슈] 10분 이상 문서 작성 후 제출하면 로그인 페이지로 이동

애써 작성한 문서를 다 날리고 처음부터 다시 작성해야 함



사용자 불편 사항	<ul style="list-style-type: none"> • 사용자가 온-나라시스템에서 10분 이상 문서 작업을 한 후 제출 버튼을 눌렀으나 다시 로그인 페이지로 이동 • 다시 로그인 하면 작성했던 문서가 보여져야 하는데 사라져 버린 상태로 사용자가 매우 당황스러운 상황 • 문서 작성 도구처럼 중간 자동 저장 기능이 있을 거라 생각하고 사용함 			
운영자 불편 사항	<ul style="list-style-type: none"> • 어느 정도 시간이 흐른 후 정보가 사라지는 확인하기 어려움 (5분, 10분, 20분..) • 상황 재현 후 SR 진행 			
대상	<table border="1" style="width: 100%;"> <tr> <td style="background-color: #0056b3; color: white;">전체 사용자</td> <td style="background-color: #0056b3; color: white;">조건</td> <td>• 장시간 문서 작업 시 겪는 장애 상황</td> </tr> </table>	전체 사용자	조건	• 장시간 문서 작업 시 겪는 장애 상황
전체 사용자	조건	• 장시간 문서 작업 시 겪는 장애 상황		

장애 처리 과정	담당자	세부 과정	소요시간
	온-나라 사용자	1. 사용자 문서 작성 후 완료 하였으나 다시 로그인 되어 모든 내용이 사라짐. 2. 작성한 내용을 찾을 수 없고, 다시 작성하기 어려워 당황함. 3. 온-나라 담당 주무관에게 이러한 현상을 전달	30분
	온-나라 담당자	1. 접수 받은 내용을 재현 2. 온나라 주무관도 10분 이상 문서작성을 한 후 제출 후 같은 현상 확인 3. SR에 오류 상황을 요청	60분
	유지보수 담당자	1. 주무관에게 전달 받은 내용을 재현 및 확인 2. 시스템 담당 엔지니어에게 요청	120분
	전문 엔지니어	1. SR에서 전달받은 내용을 재현 및 확인 2. 어떤 시스템에서 문제인지 로그분석 3. 장애 처리 조치	400분

구분	세부 항목
발생 시점과 주기	<ul style="list-style-type: none"> • 오픈 후 초기 에 간헐적으로 전체 사용자들이 겪는 상황 • 시스템 작업이 없다면 설정 변경이 없었다면 서비스 초기부터 꾸준히 발생 • 로그아웃을 하거나 웹 브라우저 창을 끄지 않은 사용자들은 일정 시간(timeout)시간이 지나면 같은 현상 발생
관련 이슈	<ul style="list-style-type: none"> • 운영자 입장에서 사용자의 불편 사항을 미리 알 수가 없음 • 업무 진행 에 치명적인 문제가 되는 시스템 오류 • 관련자 들이 동일한 상황에 대해서 계속적으로 재현하고 보고

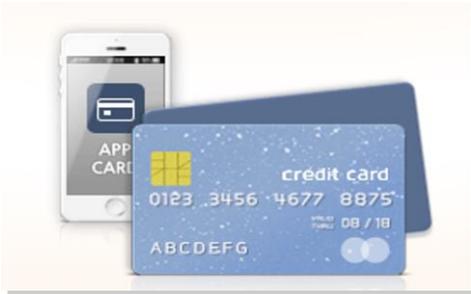
코리아 크레딧 뷰로(KCB) - 클라우드 네이티브 전환

코리아 크레딧 뷰로(KCB) MSA 기반 개인신용정보, 기업신용정보, 마이데이터 서비스 - PaaS & Cocktail APM



프로젝트명	올크레딧 개인신용정보 서비스
목적	신용관리, 명의보호, 금융관리를 통합하여 관리하는 국가대표기관 시스템
규모	개인신용정보서비스와 기업신용정보서비스 같은 핵심적인 서비스 운용 중
나무기술 역할	오픈소스 WAS 마이그레이션 컨설팅, 클라우드 네이티브 환경 도입 컨설팅

☑ 나무기술과 함께 PaaS 환경을 도입/구축한 서비스



카드 본인 확인 서비스
상용 WAS를 오픈 소스로 전환



개인신용정보서비스
PaaS 도입



기업CB 구축사업
MSA 기반으로 재개발



마이데이터 신규 구축 사업
API G/W 도입

한국항공우주산업 (KAI) - 클라우드 인프라 구축

한국항공우주(KAI)의 스마트 플랫폼을 위한 PaaS 도입 사례입니다.



프로젝트명	스마트 플랫폼 구축
목적	한국항공우주 분야의 국가 경쟁력 제고를 위한 업무
규모	국내 최대 클라우드 기반 스마트 팩토리 플랫폼
나무기술 역할	PaaS 구축과 Cocktail APM으로 성능 튜닝, 안정화 지원

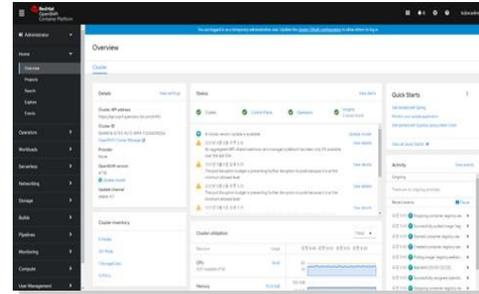
✓ 스마트 플랫폼을 위한 PaaS 환경을 나무기술이 구축



한국항공우주산업(KAI)
항공기개발센터 전경



스마트 플랫폼 구축 관련 기사
이지경제



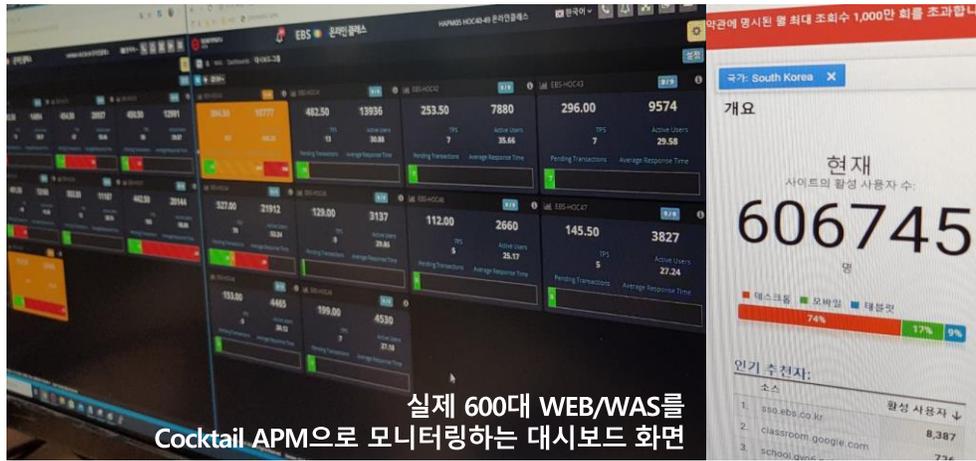
스마트 팩토리 플랫폼
구축을 위한 PaaS 환경



스마트 팩토리 플랫폼으로 구축한
KF-21(보라매)

한국교육방송공사 - 코로나 대응을 위한 온라인 클래스 시스템

EBS 온라인 클래스에 도입되어 **동시 접속자 130만 명**, VM 600대 , **WAS 인스턴스 1200여 개** 모니터링



실제 600대 WEB/WAS를 Cocktail APM으로 모니터링하는 대시보드 화면

프로젝트명	EBS 온라인 클래스
목적	코로나로 인하여 초,중,고 300백만 명의 학생이 접속하는 온라인 클래스 구축 업무
규모	WEB/WAS 600대 / 인스턴스 1,200개 규모
나무기술 역할	1주일 내 600대 머신에 1,200개 인스턴스 구축 및 APM으로 오픈지원, 성능 튜닝 및 안정화 지원

✓ Cocktail APM으로 안정화를 실현한 실제 EBS 방송 화면



교육 차관에게 Cocktail APM으로 보고하는 화면



600여대의 서버의 총 1,228개 WAS 모니터링



실제 인스턴스 1,200개 그룹대시보드 모니터링 화면



실시간 Active User를 확인할 수 있는 모니터링 화면

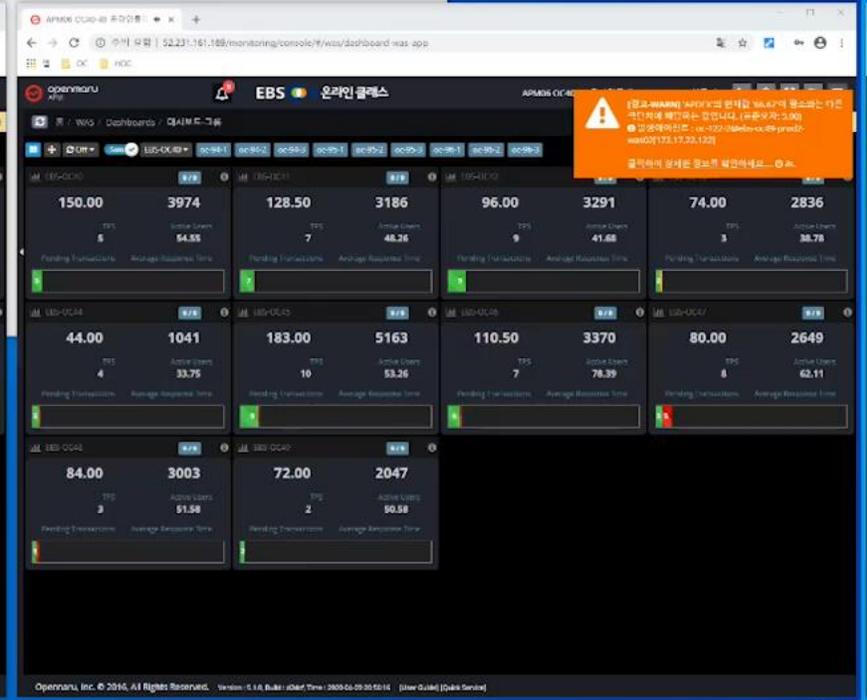
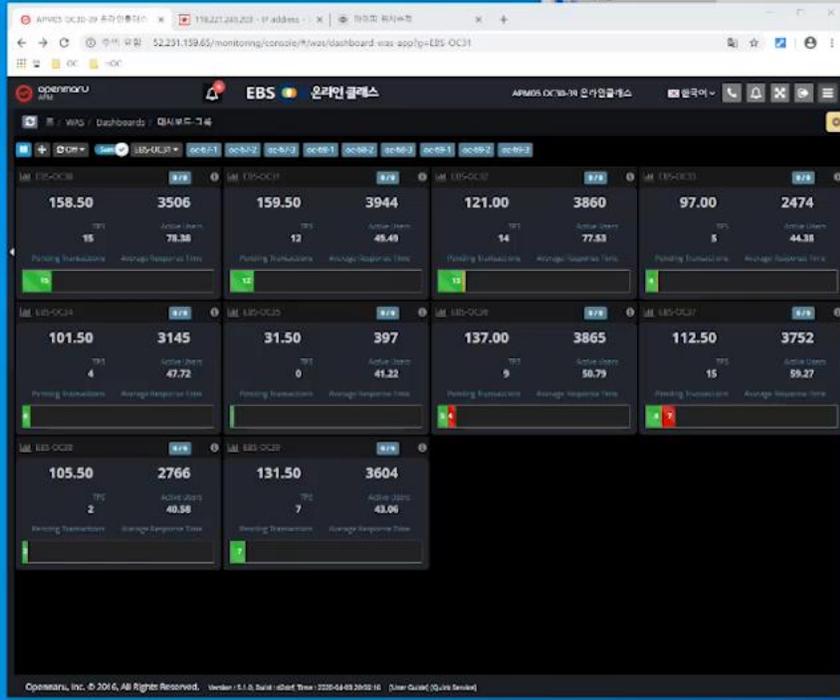
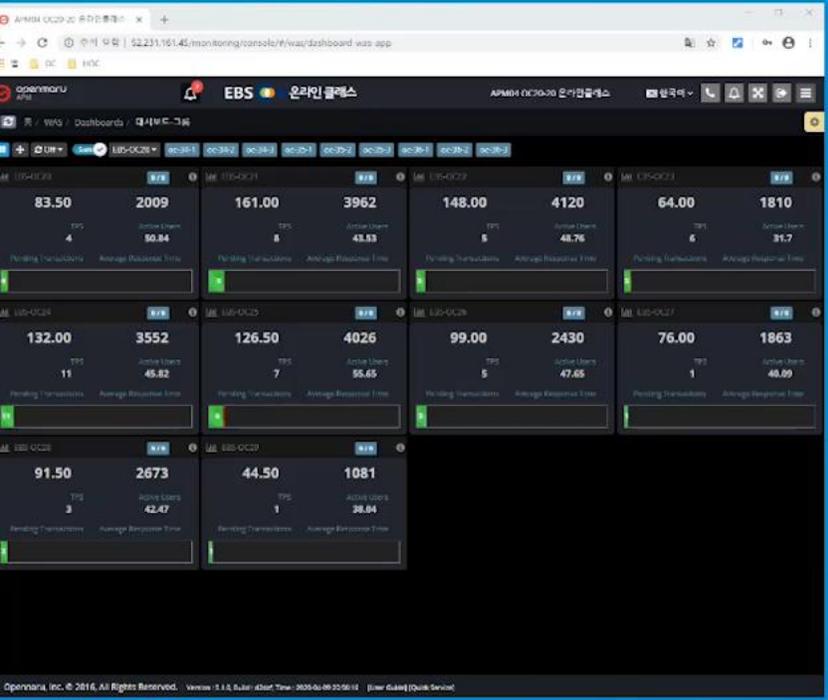
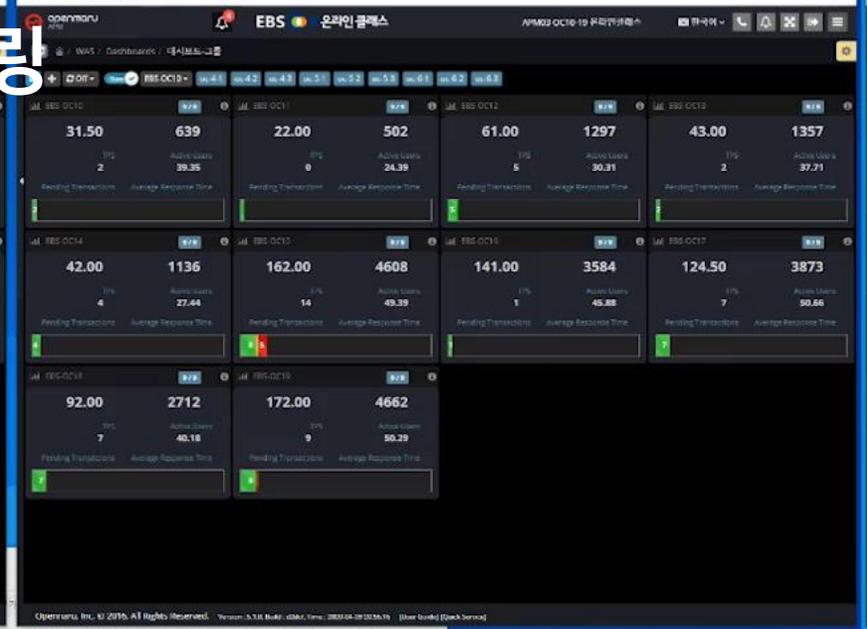
교육 멈출순 없다.

코로나 극복



Cocktail
APM

[장애 발생] = 비정상 상황 발생 그룹 모니터링



[중요-WARN] 서버의 운영을 위해서 필요한 것은 데이터 백업입니다. (중요-우선) 운영을 위한 db-backup을 꼭 하세요. (17.12.13)

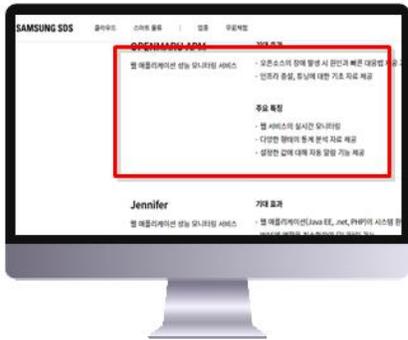
클라우드를 넘어 SaaS 서비스까지 하는 APM 전문기술기업

설치형 APM 제품에서 부터 SaaS 클라우드 서비스까지 구축하고 운영중인 클라우드 전문 기술 기업입니다.

- 국내 최초로 OpenShift와 Kubernetes 지원, 2021년에는 AWS Cloud 에 Cocktail APM SaaS서비스를 오픈
- SaaS 서비스 구축은 클라우드 네이티브 기술과 보안, 멀티테넌시, 미터링, 데이터 관리 등 높은 수준의 기술들을 구현



- 삼성이 검증한 APM 제품
- 컨테이너 환경에 최적화된 APM
- 클라우드 적합한 가격 정책



클라우드 환경에서 다양한 제품과의 호환성 검증

클라우드 환경에서는 다양한 S/W들이 사용될 수 있어 **APM 제품도 폭넓은 제품 지원이 중요합니다.**

- Cloud, OS, WAS, DBMS , 프레임워크 등 오픈소스S/W 와 상용 S/W 넘어 다양하게 지원 할 수 있어야 합니다.
- OpenShift(Kubernetes)와 클라우드 환경에 검증된 APM 제품인지 확인 필요합니다.

오픈소스WAS 와 상용WAS 모두 지원

 JBoss EAP 5~7	 9 버전 이상
 JBoss AS 5, 8이상	 6 버전 이상
 5.5 버전 이상	 8 버전 이상
 1 버전 이상	 17 버전 이상
 1.5 버전 이상	 3 버전 이상

Unix/Linux 모두 지원

지원하는 OS 환경

지원하는 OS	Version
Red Hat Enterprise Linux	5 이상
CentOS	5 이상
Amazon Linux	2016이상
IBM AIX	5 이상
HP-UX	11i 이상
Solaris	10 이상
Windows	2003 이상

지원하는 클라우드 및 컨테이너 환경

Cloud	Service/Version
Public Cloud	AWS, Azure, GCP 지원
Red Hat OpenStack	가상화 지원
OpenShift Container Platform	3.X 이상
Kubernetes	1 이상
Docker	1.6 이상

Public Cloud 와 OpenShift/OpenStack/ Kubernetes 모두 지원

모니터링 가능 플랫폼



Cocktail APM

물리 환경	리눅스/가상화	Private Cloud	Public Cloud	컨테이너
Active-X	HTML5	HTML5	HTML5	HTML5
Java EE	Spring /e-Gov	Spring /e-Gov	Spring /e-Gov	Spring /e-Gov
JEUS/ WebLogic	Apache/ Tomcat/JEUS	Apache/ Tomcat/JEUS	Apache /Tomcat/JEUS	Apache/ Tomcat/JEUS
Java	Java	Java	Java	Java
vPar/PowerVM /OracleVM	Red Hat Virtualization			
HP-UX/ AIX/Solaris	Red Hat Linux			
PA-RISC/ Power/SPARC	X86			

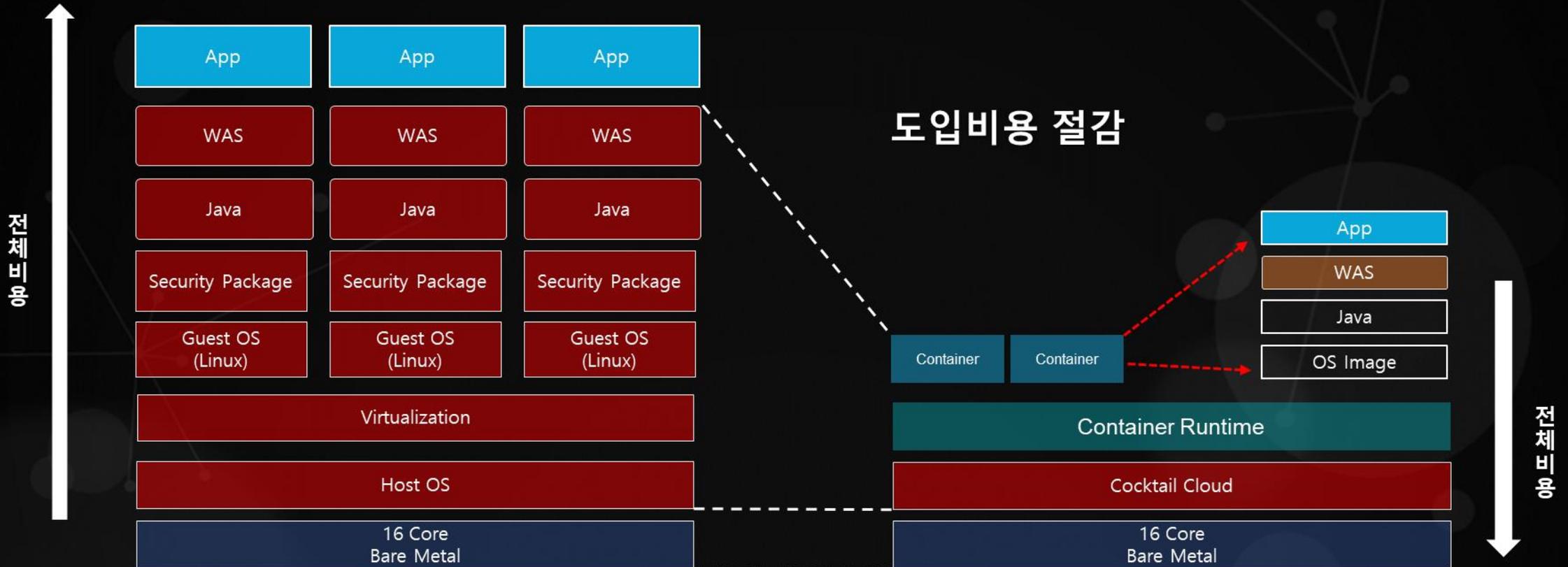
Application Performance Management



핵심은 Application !!!! APM
(Application Performance Management)

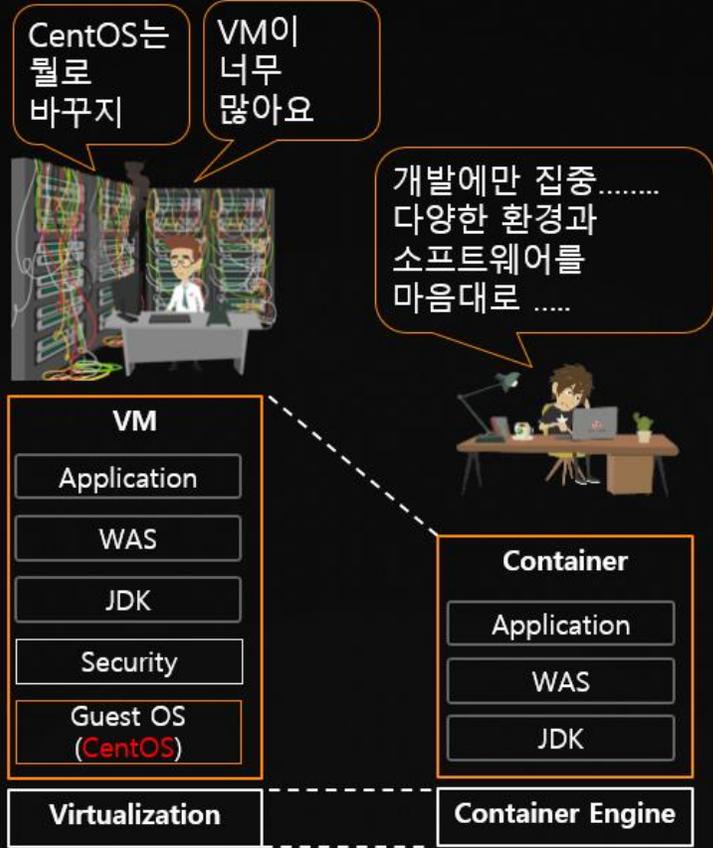
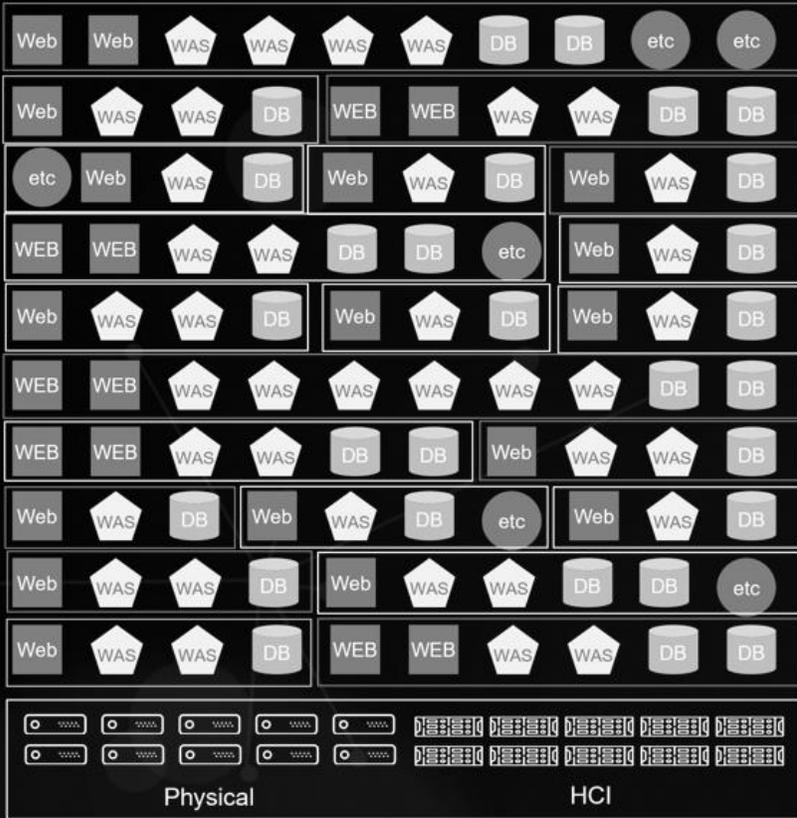
가상화 VS 컨테이너 비교 - 비용적인 측면

- 가상화 대비 Guest OS 유지보수, 라이선스, 관리비용 제거
- 서버 접근제어를 비롯한 보안 솔루션 제거



Legacy 시스템 구축 vs 클라우드 네이티브 구축

시스템마다 Web/WAS/DB 형태로 구축



S/W 를 플랫폼에서 제공 - 개발에만 집중



- 물리서버나 가상화 형태로 WEB/WAS/DB 3 티어로 구축
- Guest OS에 따른 OS 라이선스 이슈와 보안 솔루션 구매 비용 발생

- 애플리케이션에 필요한 S/W만 컨테이너로 배포
- 클라우드 네이티브로 전환, DevOps, CD/CD 기반

머신 중심에서 애플리케이션 중심 인프라로 변화

- 컨테이너화는 데이터 센터를 머신 중심에서 애플리케이션 중심으로 전환
- 개발자와 운영팀에게 서버와 운영 환경에 대한 세부 사항을 추상화
- 운영 중인 애플리케이션과 개발자에 미치는 영향을 최소화하면서 새로운 하드웨어 또는 운영 환경을 업그레이드하여 인프라팀에게 유연성을 제공
- 서버의 CPU와 메모리 정보 뿐만 아니라 애플리케이션과 관련한 매트릭을 연결하여 오토 스케일링

Machine Centric Infrastructure



Application Centric Infrastructure

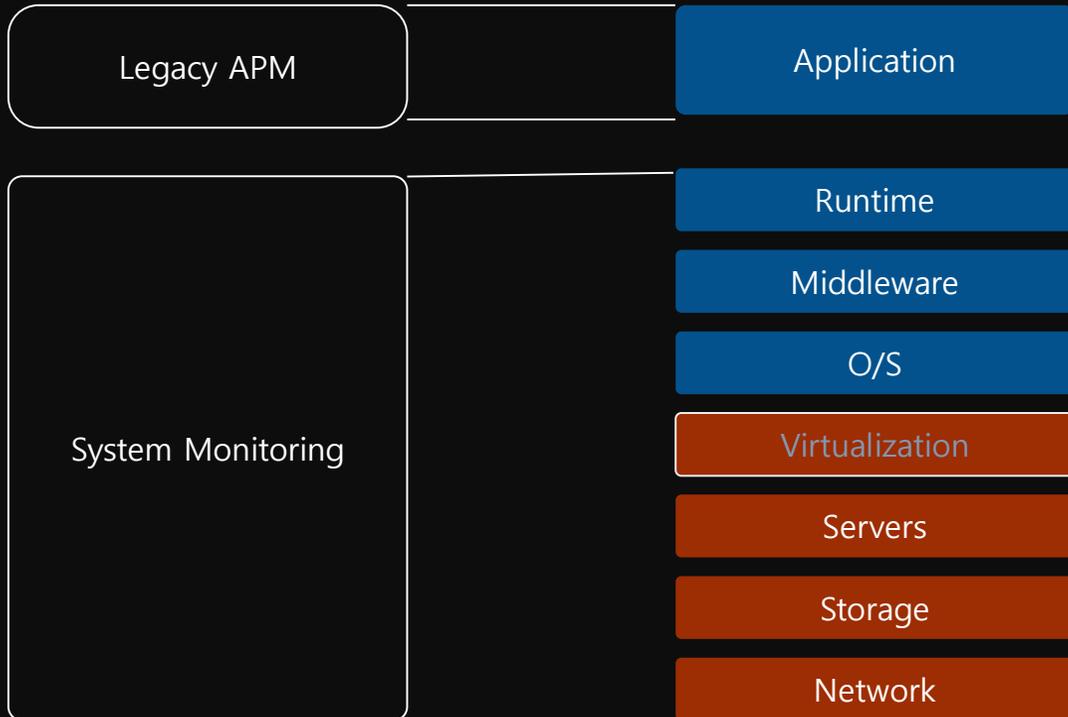


애플리케이션 중심 인프라 스트럭처에서 모니터링

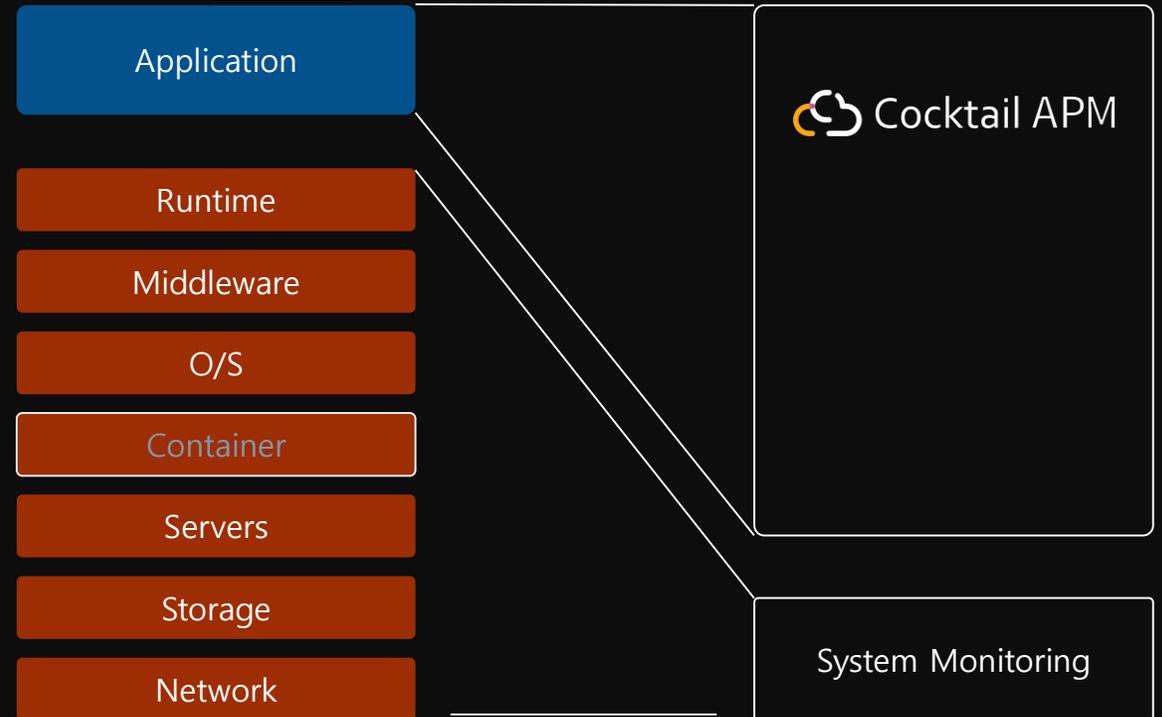
- PaaS는 Application 중심의 모니터링
- IaaS는 인프라 중심의 모니터링



Infrastructure-as-a-Service (IaaS)



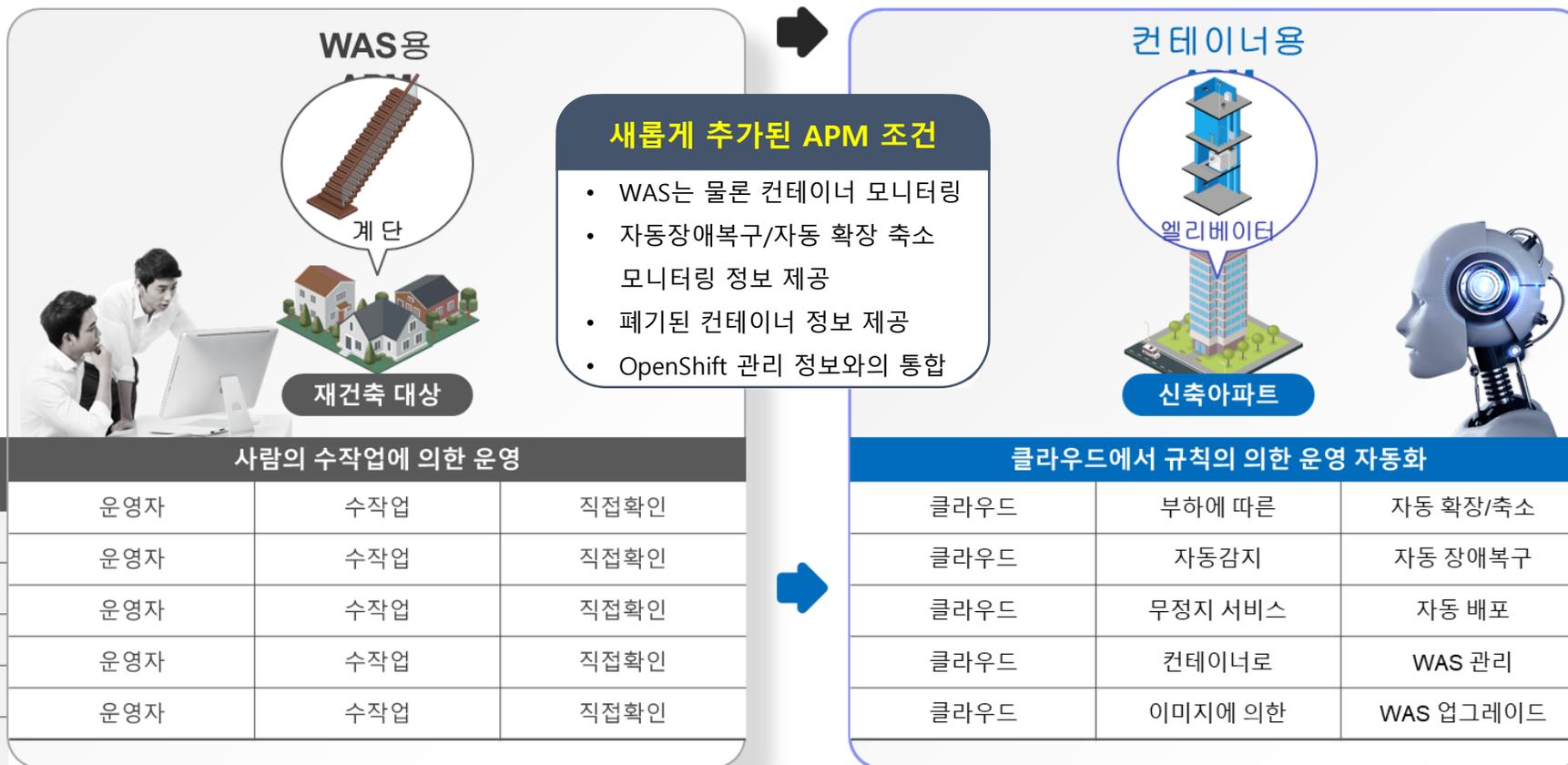
Platform-as-a-Service (PaaS)



클라우드 전환에 따른 APM 요구사항의 변화

클라우드에서는 APM 모니터링 대상이 WAS에서 컨테이너로 변경

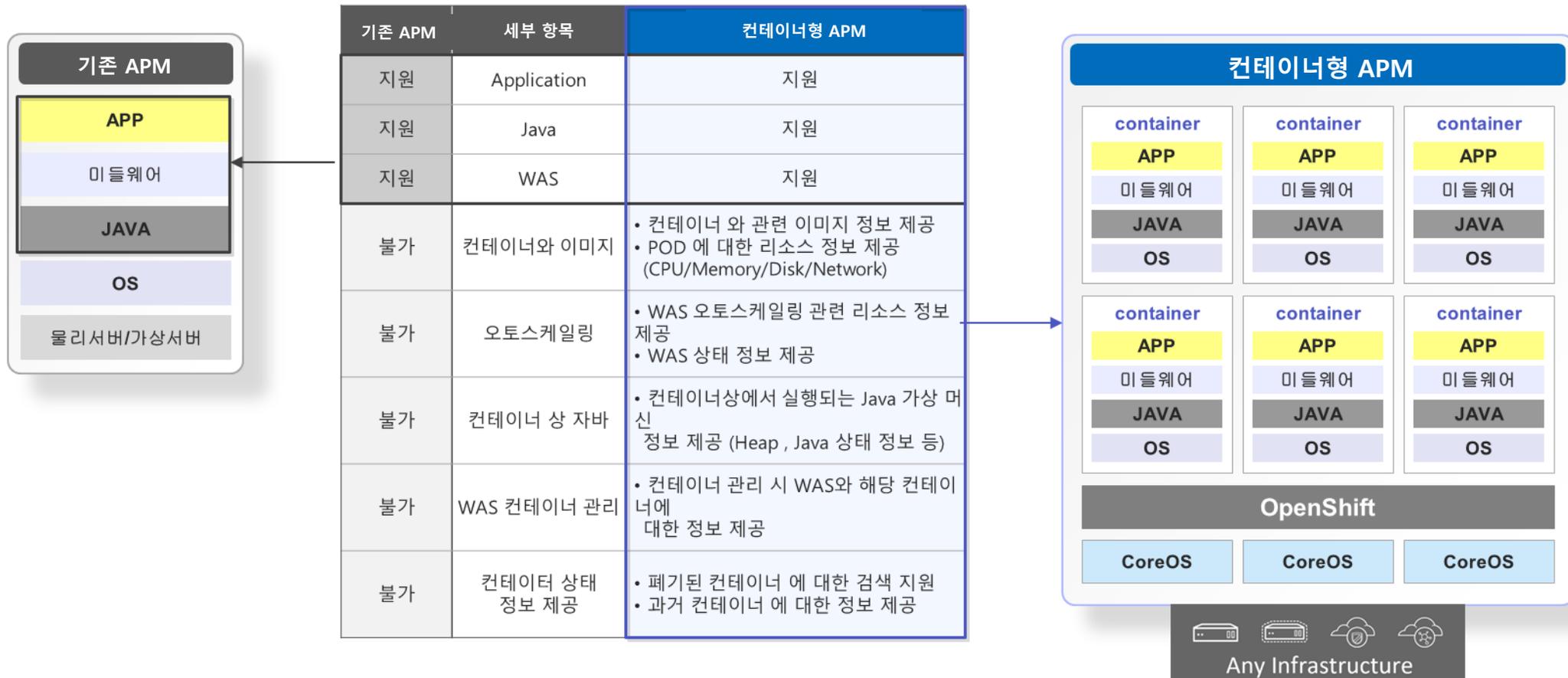
- 컨테이너 단위로 WAS에 대한 확장/축소, 장애 복구, 업그레이드, 패치 작업하여 WAS와 함께 컨테이너를 모니터링해야 합니다.
- 기존 물리서버나 가상서버와는 달리 컨테이너는 휘발성으로 상태를 가지고 있지 않습니다.



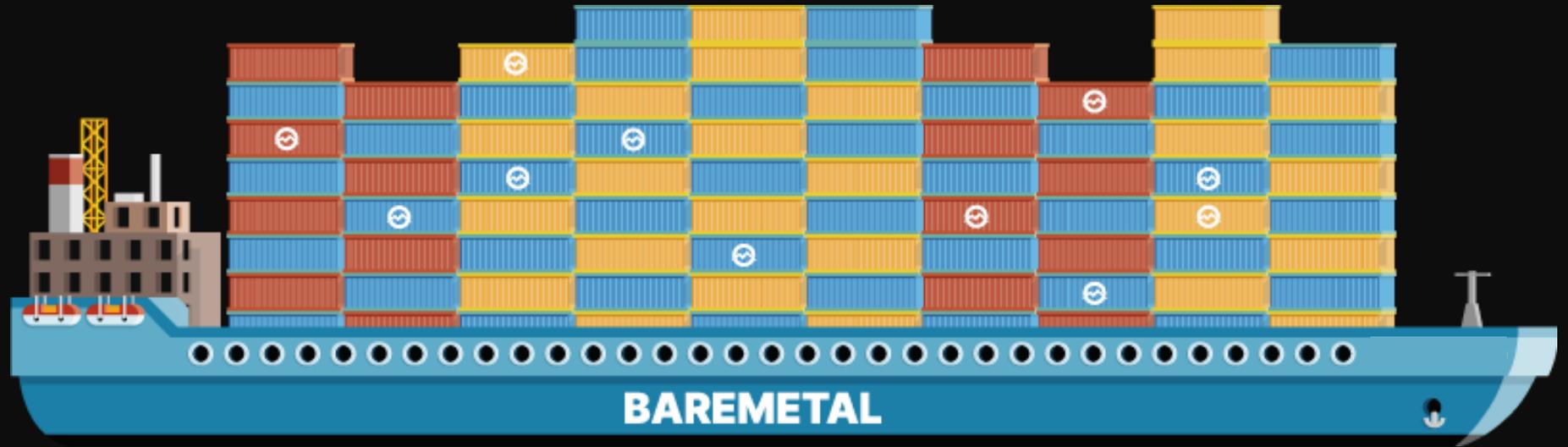
클라우드에서 APM 요구사항과 제품 비교

Cocktail APM은 국내 최초의 OpenShift/Kubernetes 모니터링을 지원하는 APM입니다.

- WAS 모니터링을 넘어 OpenShift의 CoreOS, 컨테이너, WAS 컨테이너 관리 정보, 기존 WAS 모니터링 정보 제공해야 합니다.
- OpenShift 관리도구와 이벤트 정보를 APM과 통합하여 운영 효율성을 확보해야 합니다.
- Cocktail APM은 OpenShift 상에서 가장 많은 구축 사례를 가지고 있습니다.



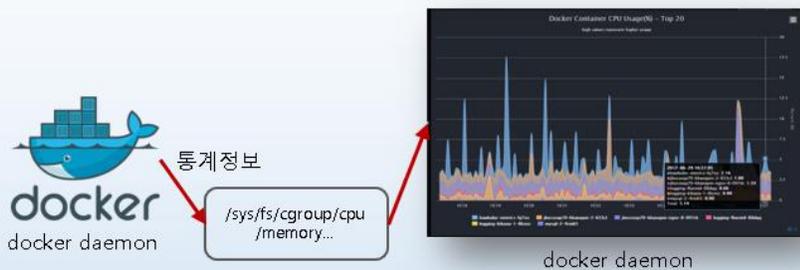
어떤 것이 상식적인 아키텍처로 보이시나요?



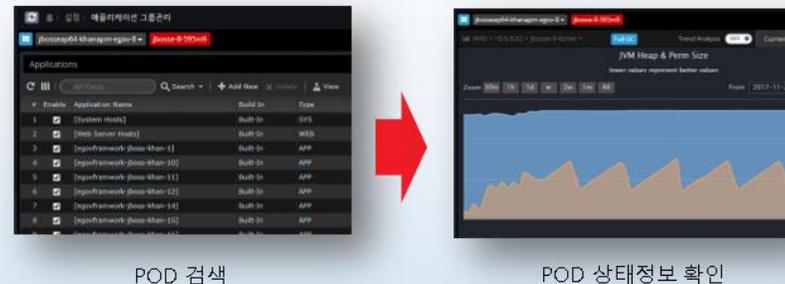
PaaS 환경에서 Cocktail APM이 필요한 이유는?

- Docker 환경은 가상 OS환경이기 때문에 시스템 리소스 관리가 어려움
- 유동 IP이며, 멀티캐스트를 지원하지 않는 환경

▪ 가상 OS 환경에서 CPU/Memory/Disk/Network 정보 실시간 제공



▪ 폐기된 POD 인스턴스에 대한 검색 및 상태 정보 제공



▪ 오토스케일링 시에 부하 분산과 pod 별 처리 현황 파악



▪ WAS 장애 상황에서 필수적으로 필요한 분석 도구 제공



Micro Service Architecture

Cocktail APM 특징점



01



차별화된 기능

- 머신러닝 기반의 장애 경고 알람
- 헬스체크
- 킷서비스
- 프로비저닝
- 웹 서버 모니터링
- 시스템 모니터링

02



전문 장애분석 도구 제공

- 스레드 덤프 분석
- JVM 메모리 객체분석
- 네트워크 상태 분석
- 오픈파일 분석
- 시스템 프로세스 분석
- 데이터 추세 분석

03



오픈소스 S/W에 최적화된 APM

- 오픈소스에 최적화된 APM
- 컨테이너 환경에 최적화된 APM
- AWS 환경에 최적화된 APM

04



미들웨어 전문기술기업

- GS 인증 1등급
- 오픈소스 전문기업이 만든 APM
- WAS 장애 지원 및 튜닝 지원
- 행정안전부 온-나라 BMT 성능 평가 1위

05



최신 IT 기술 지원

- WebSocket을 통한 포트 사용 최소화
- 아마존 클라우드/컨테이너 환경에서 오토스케일링 지원
- PaaS 지원 APM 시장 점유율 1위

머신러닝 기반 장애 경고 알람

Cocktail APM은 머신러닝 기반으로 임계치에 도달하기 전에 이상 징후를 사전에 예측

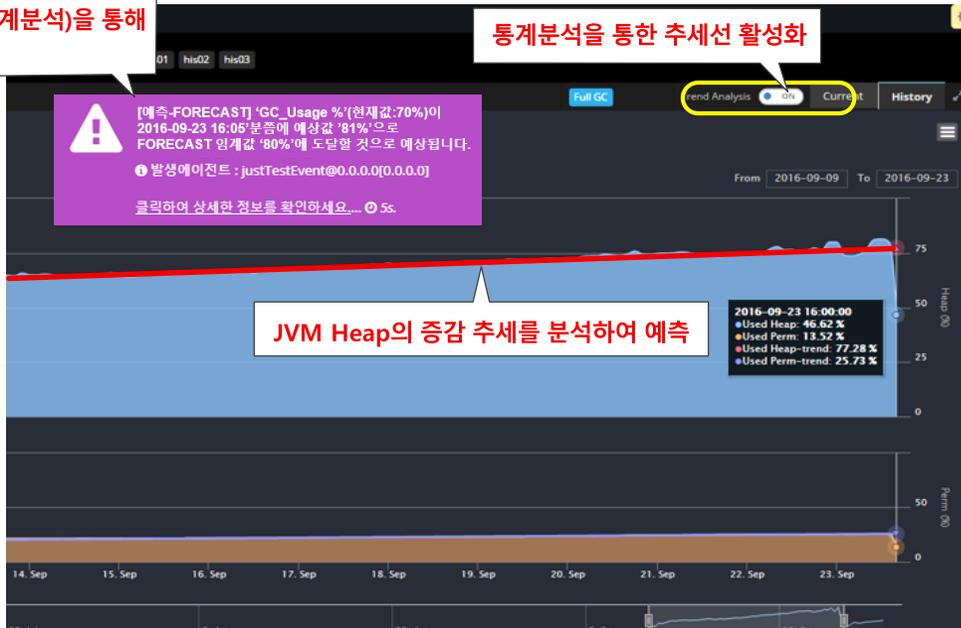


- ⊙ 사용자가 다양한 성능과 장애 정보들의 임계치나 추이 파악이 어려움
- ⊙ APM에서 제공하는 모든 정보를 확인할 수 선제적 대응과 장애와 성능관리가 어려움



- 실시간 머신러닝(통계분석)을 통해 앞으로 몇 분 후에 관리자가 설정한 임계 값에 도달할 것이라는 예측 이벤트를 통보합니다.
- APM은 현재 상태 모니터링 및 추세분석을 통해 임계값 도달 시점을 예측하여 이벤트를 통보 한다.

머신러닝(통계분석)을 통해 예측 알림



통계분석을 통한 추세선 활성화

JVM Heap의 증감 추세를 분석하여 예측

Java VM의 Heap 사용량 증가 추이를 분석하여 GC 임계값 초과 시점에 대한 정보 제공

☑ 장애경고 알람 대상 항목

항목	이벤트 발생 항목
WAS	<ul style="list-style-type: none"> JVM Heap Usage APDEX Error Rate JVM Perm Usage Database Response Time Pending Transactions GC Usage Database Connection Pool Usage
WEB	<ul style="list-style-type: none"> WEB Traffic Worker Usage
System	<ul style="list-style-type: none"> Memory Usage Disk Usage Memory Swap Usage CPU Usage Network Packet Error Rate
Cubrid	<ul style="list-style-type: none"> CAS Usage

온라인 장애분석서비스(퀵서비스) 제공

장애 발생 즉시 온라인을 통해 **전문가에게 장애 원인 분석을 요청할 수 있을까요?**



고려사항



고려사항

- ☑ 장애 발생 시 엔지니어가 도착하기 전에 장애 분석이 필요함
- ☑ APM 사용 시 APM에서 제공하는 정보가 궁금할 때 즉각 문의와 내역 관리



방안

- 퀵서비스는 시스템 장애 발생시 해당 화면과 시스템 정보를 전문가에게 이메일 등으로 전달하여 원인이 무엇인지 분석해 주는 실시간 온라인 서비스 입니다.
- 실시간으로 문의된 퀵서비스 내용은 신속하게 담당자가 할당되고, 처리 과정을 포탈, 메일등을 통해서 보고서 형태로 공유합니다.

퀵서비스

이슈 내용을 간략하게 남겨주세요.

Company	Username
오픈	나루
Phone	Email
000-0000-0000	support@opennaru.com
Summary	
홈 / WAS / Dashboards / 대시보드	
Description	
Description	

Cancel 보내기

화면 캡처와 장애 현상에 대한 설명으로 기술 지원 요청

☑ 퀵서비스 처리 프로세스



사용자 관점의 주기적인 서비스 품질점검 서비스

서비스 접속이 안된다고 불편이 접수되는데, 내부에서는 인지하지 못하는 상황을 겪어보진 않으셨나요?



- ☑ 인터넷망 사용자의 접속이 문제가 있는 경우 메신저, SMS, 이메일 등으로 상황 전파가 되어야 함
- ☑ 사용자 관점의 SLA 를 일간/주간/월간 등의 주기로 보고서를 생성해서 전달해야 합니다.



- 사용자 입장에서 주기적인 서비스 이상유무를 판단합니다. (예를 들어, 서버는 이상이 없는데 네트워크 장비 등의 문제로 서버 접속이 안되는 경우)
- 서비스별로 URL 기준으로 정의하며, 요청 성공률, 가용성, 평균응답시간, 호출 횟수 등의 서비스 정상 유무를 확인 합니다.

브라우저 엔진 활용하여 주기적으로 체크



☑ 헬스체크 기능 활용

헬스체크 기능

- 서비스에 대한 주기적인 생사 확인
- 평균 응답 시간 추이 확인
- 특정 기능/페이지 상태 점검
- 재 배포 후 다수의 페이지에 대한 일괄 점검
- 사용자 관점의 서비스 상태 확인
- 인프라 구간별 서비스 상태 확인
- 복합적인 점검 수행과 오류에 대한 사용자 알림
- 스크립트를 이용하여 프로세스/파일/자원 사용량 등을 주기적으로 모니터링

대규모 클라우드 모니터링 기능 (그룹 대시보드)

대규모 클라우드 운영환경에서 수천 개의 WAS를 어떻게 모니터링 할 수 있을까요?



- ⊙ 대규모 클라우드 환경에서는 수천 대 서버와 수천 개의 WAS를 일목요연하게 모니터링 할 수 있어야 함
- ⊙ 수천 대 이상의 대규모 클라우드 환경에서 대규모 데이터를 수집하고 모니터링한 제품

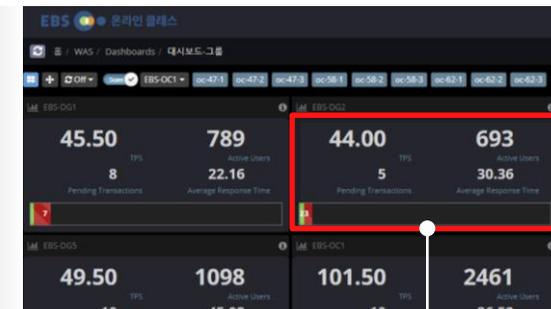


- 그룹별 실시간 서비스 지연 현황 확인 - 그룹별 TPS, 평균 응답시간, 액티브 사용자수 확인 가능
- 수천 대의 서버와 수천 개의 WAS에서 발생하는 성능과 장애 관련 데이터를 수집하고 시각화 할 수 있습니다.

600여 대의 서버의 총 1,228개 WAS 모니터링



30여 대의 서버의 총 120개 WAS 모니터링



4대의 서버의 총 10개 WAS 모니터링



- 1 그룹 총 TPS
- 2 지연
- 3 그룹 총 동시 접속자
- 4 그룹 총 평균 응답시간
- 5 전체 지연 큐 그래프



2020.4.22 민간기업 적극 도움.. 원격 수업 3일째 안정/EBS 뉴스

멀티 애플리케이션 운영 환경에서 클라우드 모니터링

여러 애플리케이션을 동시 운영 시 자원을 효율적으로 활용할 수 있을까요?



고려사항



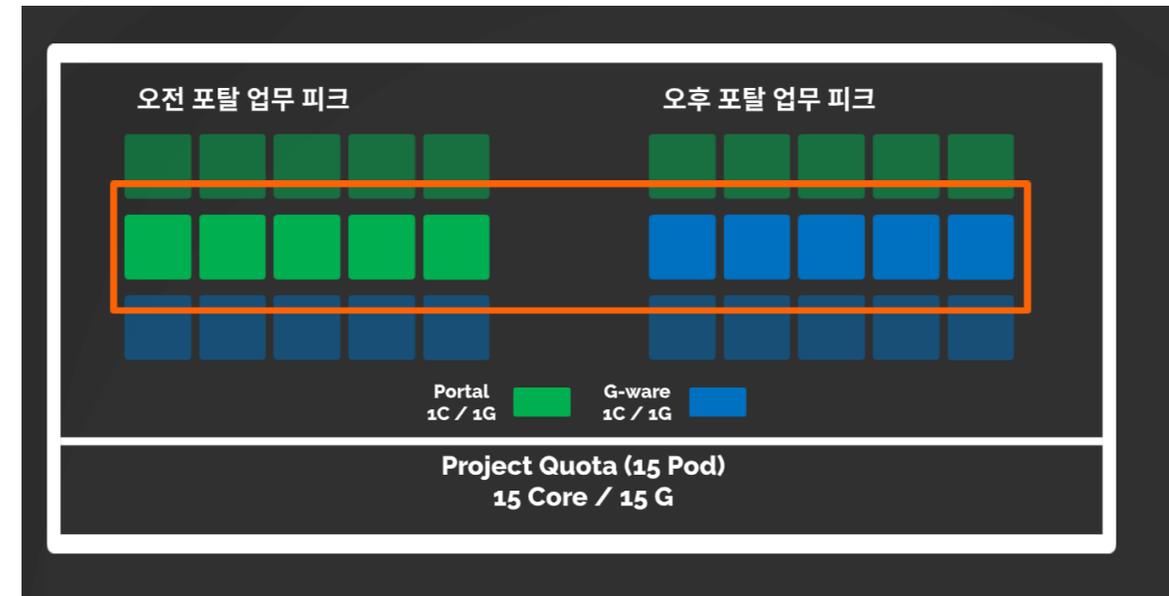
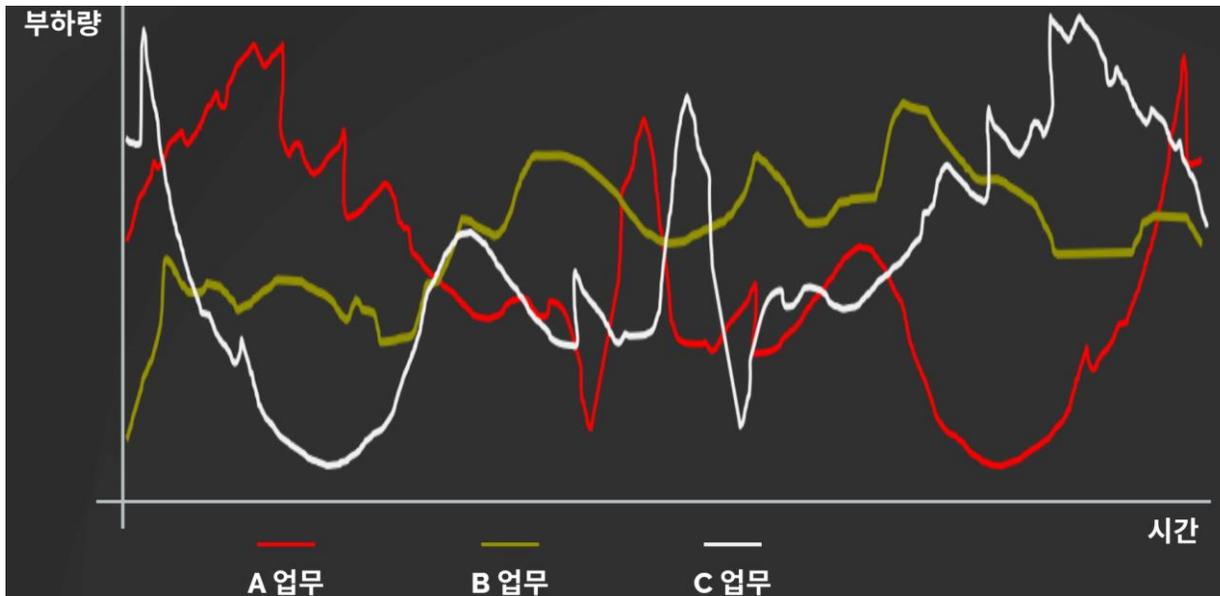
고려사항

- ⊙ Peak 시간 대가 다른 여러 애플리케이션을 운영 시 기존의 시스템 구성과 사이징은 적합하지 않음
- ⊙ 관리자가 매일



방안

- 관리자의 개입 없이도 시스템 부하량에 맞추어 자원일
- 클라우드 서비스 환경에서도 중요한 시스템



클라우드 장애관리와 성능상의 특화기능

APM은 모니터링을 넘어 클라우드 환경에서 장애에 대한 원인규명과 분석을 할 수 있는 기능이 있어야 합니다.



고려사항



고려사항

- ☑ 장애 발생 시 엔지니어가 도착하기 전에 장애 분석이 필요함
- ☑ APM 사용 시 APM에서 제공하는 정보가 궁금할 때 즉각 문의 할 수 있어야 함



방안

- 애플리케이션 성능 모니터링 및 진단, 장애 원인 분석을 통해 서비스를 최적의 상태로 운영할 수 있는 모니터링 솔루션이다.

Java 스레드 덤프 분석기

#	Lin.	Type	Name	State	URL	Durati...	CPU TI.	Class	At
1	696	Active Thread	ap-0.0.0.0-8009-	RUNNABLE		1,309	0.0	jav...	
2	871	Active Thread	ap-0.0.0.0-8009-	RUNNABLE		123,355	0.0	jav...	
3	936	Active Thread	ap-0.0.0.0-8009-	RUNNABLE		156,825	10.0	jav...	
4	1,556	Active Thread	ap-0.0.0.0-8009-	RUNNABLE		183,887	0.0	jav...	

Lock을 추적가능, URL 정보표시

Java 메모리 누수 분석기

#	Num	Classname	Bytes [%]	Bytes	Instances
1	1	char[]	35.25%	35.2 MB	258,535
2	2	byte[]	8.62%	8.6 MB	10,388
3	3	java.lang.String	6.11%	6.1 MB	253,859
4	4	java.util.HashMap\$Node	5.55%	5.5 MB	173,073
5	5	java.util.jar.JarFile\$JarFileEntry	5.96%	5.0 MB	52,595

Java 메모리를 점유한 객체 분석/비교

네트워크 상태 분석기

#	Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
1	tcp	0	0	127.0.0.1:9999	0.0.0.0:*	LISTEN	509/java
2	tcp	0	0	10.6.3.224:5455	0.0.0.0:*	LISTEN	509/java
3	tcp	0	0	10.6.3.224:7600	0.0.0.0:*	LISTEN	509/java
4	tcp	0	0	0.0.0.0:8080	0.0.0.0:*	LISTEN	509/java

Java 프로세스,시스템이 사용중인 네트워크 분석

오픈파일 분석기

#	Cam.	PID	User	FD	Type	Device	Size/Off	Node	Name
1	○	java	509	jboss	cwd	DIR	253,28	85	4199509 /home/jboss
2	○	java	509	jboss	rfd	DIR	253,28	261	1027 /
3	○	java	509	jboss	txt	REG	253,28	7376	6295581 /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.1
4	○	java	509	jboss	mem	REG	253,28	411090	8420366 /opt/esp/standalone/tmp/dfu/deployme
5	○	java	509	jboss	mem	REG	253,28	24956	4372527 /opt/esp/standalone/tmp/dfu/deployme
6	○	java	509	jboss	mem	REG	253,28	2666695	90124 /opt/esp/standalone/tmp/dfu/deployme
7	○	java	509	jboss	mem	REG	253,28	220526	29437985 /opt/esp/standalone/tmp/dfu/deployme

Java 프로세스가 오픈한 파일 분석

시스템 프로세스 분석기

Proceses: 750 Total / 749 Sleeping, 1 Running, 0 Zombie, 0 Stopped

CPU Usage: User: 10.0%, System: 0.0%, Nice: 0.0%, Wait: 0.0%, Idle: 90.0%

#	PID	User	Start Time	Mem	RSS Mem	Shared	State	CPU Time	CPU	ME
1	115,243	(root)	2019-01-14 17:53...	4.2 GB	254.5 MB	10.5 MB	Sleeping	05:37:11.2	23.1%	1
2	78,345	(?)	2019-01-18 18:32...	6.3 GB	404.8 MB	20.1 MB	Sleeping	00:02:20.0	21.5%	2

시스템의 프로세스 CPU, 메모리 사용량 분석/비교

데이터 추세 분석

Datasource Pool Count

higher values represent higher usages

Number of Pools

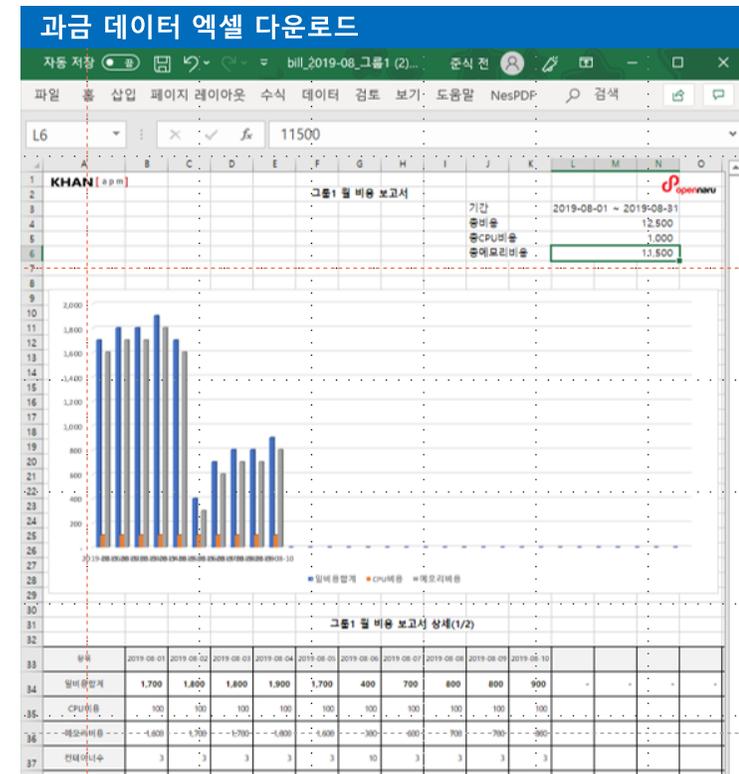
13:42 13:43 13:44 13:45 13:46

과거 데이터의 증감 추세를 분석하는 기능

컨테이너 환경 미터링 및 과금 모듈(tachoMeter)

컨테이너 환경에서 CPU, 메모리 사용량에 따른 과금 기능 제공

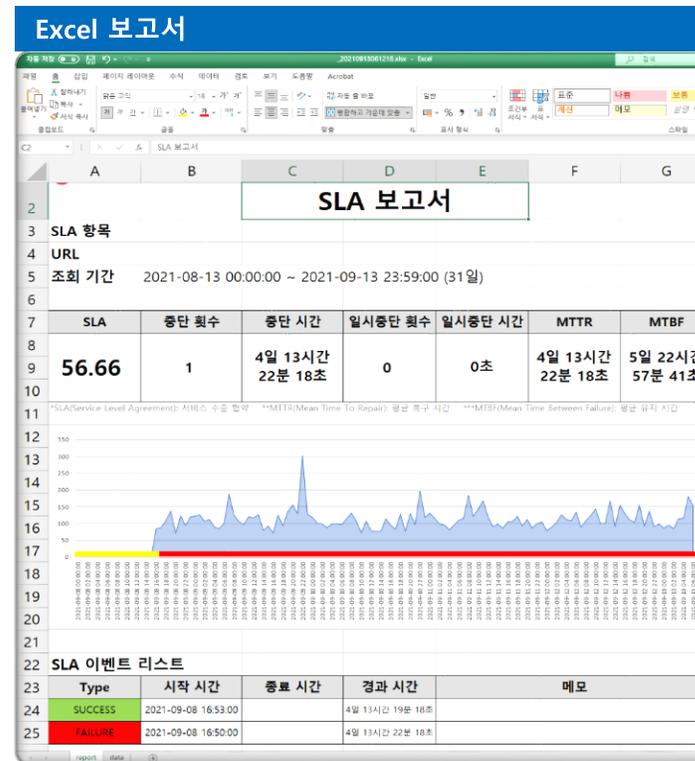
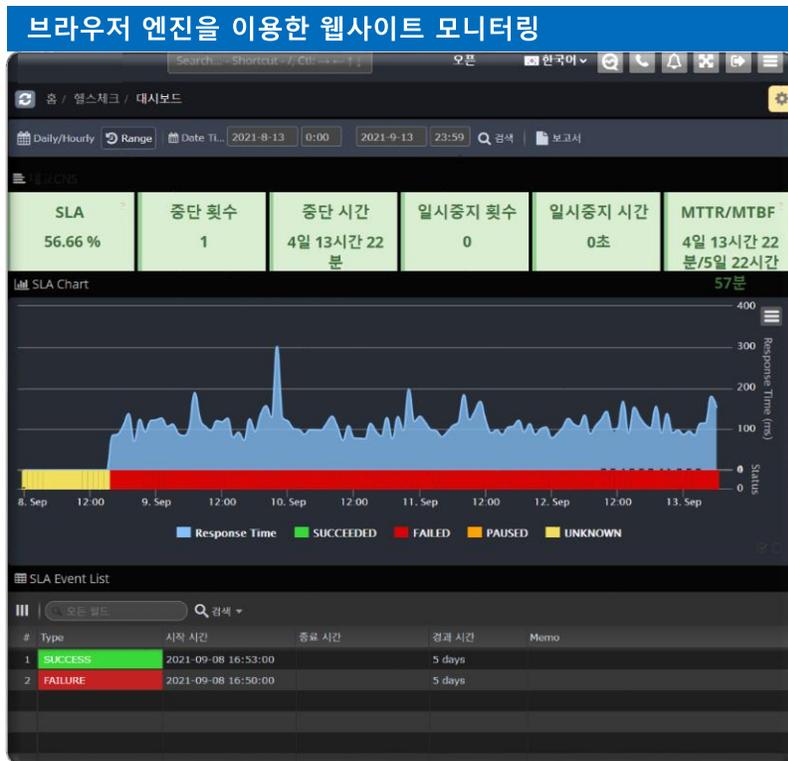
- 컨테이너의 특성상 부하에 따라 개수가 쉽게 늘어나고 줄어듦, 배포될 때마다 기존 컨테이너를 버리고 새로운 컨테이너를 생성하기 때문에 특정 시간에 사용한 컨테이너의 사용량을 측정하기 어렵습니다.
- 컨테이너 환경에서 CPU, 메모리 사용량에 따라 사용료를 계산할 수 있는 기능을 제공합니다.



SLA 모니터링 기능

사용자 측면에서 웹 서비스의 품질을 모니터링하여 서비스 품질 보고서를 제공합니다.

- 웹서비스의 응답 코드 및 콘텐츠를 측정하여 웹사이트의 품질을 측정하는 기능 제공
- 주기적으로 웹사이트가 정상 동작하는지 파악하고 빠르게 장애에 대응할 수 있도록 신속한 알림 제공, 장애 발생 시 에이전트를 통해 오류 복구 명령 실행 가능
- 단위 시간별 SLA(Service Level Agreement) 품질 점수 및 중단 시간, MTTR, MTBF 등 모니터링 지표 제공
- SLA 보고서를 엑셀 파일로 다운로드 할 수 있는 기능 제공
- 오류 발생 이력 기록, 오류 시 APM과 연계 기능 제공



컨테이너 환경에서 필요한 모니터링 방안

APM 모니터링 대상이 WAS에서 컨테이너로 변경됩니다.

- 클라우드 환경에서는 모니터링 대상이 WAS가 아닌 WAS를 담고 있는 컨테이너로 변경
- 컨테이너 단위로 WAS에 대한 확장/축소, 장애 복구, 업그레이드, 패치 작업

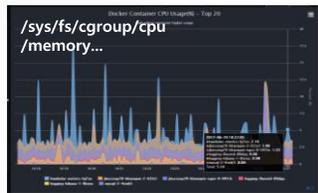
WAS가 아닌 컨테이너 단위로 관리



자동확장 축소 단위	▶ 컨테이너
자동 복구 단위	▶ 컨테이너
애플리케이션 배포 단위	▶ 컨테이너
애플리케이션 실행 환경 단위	▶ 컨테이너

↓

컨테이너 CPU/Memory/Disk/Network 실시간



Container 정보

클라우드에서는 서버 상태가 없음

mutable Infrastructure	Immutable Infrastructure
물리/가상 서버	컨테이너
 하드디스크	 이미지
VS.	
기존의 물리서버나 가상서버에서는 하드디스크에 상태를 저장 관리	컨테이너는 DVD처럼 이미지를 사용하여 읽기 전용으로 상태 저장 불가

↓

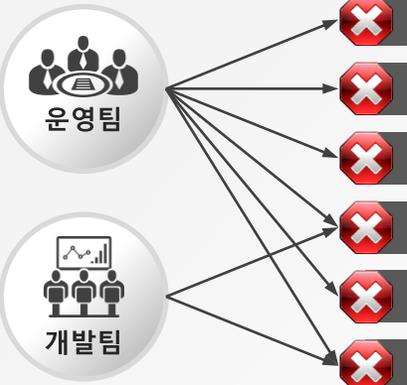
오토스케일링 시에 부하 분산과 Pod별 처리 현황 파악



자동확장/축소

2개 Pod 컨테이너 4개 Pod 컨테이너

관리자가 직접 접속하여 관리 할 수 없음



- OS 직접 접속 어려움
- WAS 관리 콘솔 접속 어려움
- WAS CLI 접속 어려움
- WAS에 직접 애플리케이션 배포 불가
- WAS Log / OS Log / Config 파일 관리 어려움
- 장애 시 직접적인 대응 어려움

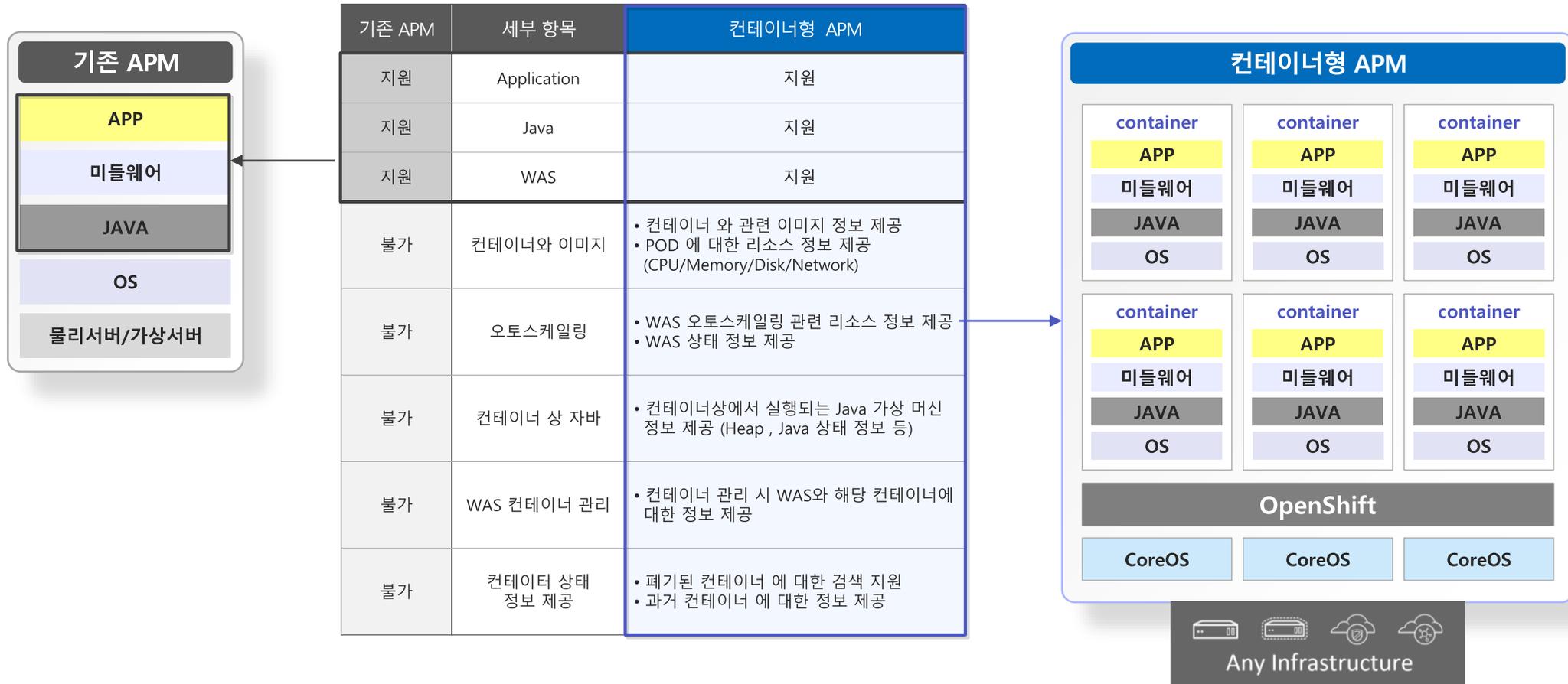
↓

폐기된 POD 인스턴스에 대한 검색 및 상태 정보 제공	WAS 관리 필수도구 제공
 POD 검색	 POD상태정보 확인
 스택드럼프 분석	 자바 메모리 객체 분석

클라우드에서 APM 요구사항과 제품 비교

Cocktail APM은 국내 최초의 OpenShift/Kubernetes 모니터링을 지원하는 APM입니다.

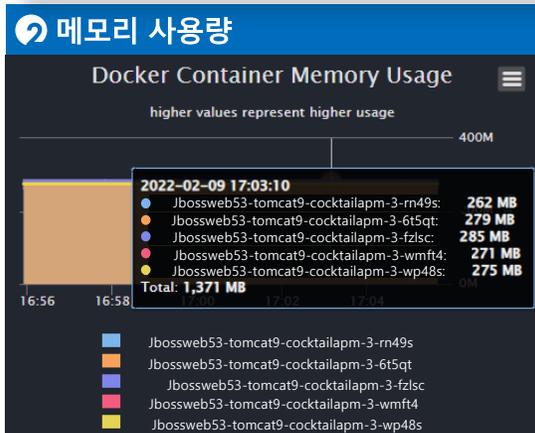
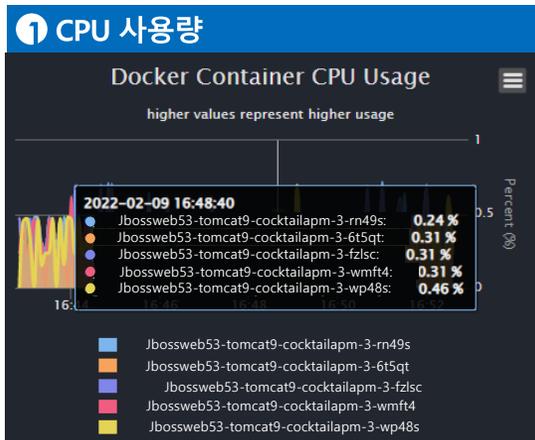
- WAS 모니터링을 넘어 OpenShift의 CoreOS, 컨테이너, WAS 컨테이너 관리 정보, 기존 WAS 모니터링 정보 제공해야 합니다.
- OpenShift 관리도구와 이벤트 정보를 APM과 통합하여 운영 효율성을 확보해야 합니다.
- Cocktail APM은 OpenShift 상에서 가장 많은 구축사례를 가지고 있습니다.



클라우드 관리도구와 통합된 모니터링 지원

Cocktail APM은 **OpenShift와 유기적인 연계**를 통해 효율적인 운영이 가능하도록 지원합니다.

- 클라우드 운영 시에 **운영 현황과 성능과 장애 이슈를 APM에서 통합적으로 모니터링**할 수 있어야 합니다.
- OpenShift 관리 기준인 **프로젝트와 운영 이벤트**를 Cocktail APM을 통해서 손쉽게 확인할 수 있습니다.



애플리케이션 그룹

Jbossweb53-tomcat9-cocktailapm-3

PaaS 웹 콘솔

오픈Shift 이벤트

pod 별 메모리 사용량 (상위 10개)	Memory Usage (MiB)
jbossweb53-tomcat9-openmaruapm-3-fzjsc	272.2 MiB
jbossweb53-tomcat9-openmaruapm-3-6t5qt	266.1 MiB
jbossweb53-tomcat9-openmaruapm-3-wp48s	262.6 MiB
jbossweb53-tomcat9-openmaruapm-3-wmft4	259 MiB
jbossweb53-tomcat9-openmaruapm-3-rn49s	249.8 MiB

Cocktail APM

1. 이벤트 리스트

- [WARN] eap74-12@test-was1 [192.168.23.193] WAS가 종료되어 에이전트와 연결이 끊어졌습니다.
- [CRITICAL] @test-apm-190 [192.168.23.190] 헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...
- [CRITICAL] @test-apm-190 [192.168.23.190] 헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...
- [CRITICAL] @test-apm-190 [192.168.23.190] 헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...
- [CRITICAL] @test-apm-190 [192.168.23.190] 헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...

2. 메시지 창

- [WARN] APDEX의 현재값 66.67이 평소와는 다른 극단적에 해당하는 값입니다. (최근오차: 14.57)
- [CRITICAL] APDEX (평균값: 22.5)이 심각 (CRITICAL) 임계값 50을 넘었습니다.
- [CRITICAL] APDEX (평균값: 43.06)이 심각 (CRITICAL) 임계값 50을 넘었습니다.
- [CRITICAL] Error Rate % (평균값: 30.44)이 심각 (CRITICAL) 임계값 20을 넘었습니다.

Updated: 2022-01-19 13:34:19.757

Cocktail APM은 **Websocket API 통신(양방향)** 방식으로 양방향 지속적으로 연결되어 실시간으로 발생하는 이벤트를 모니터링 가능

클라우드 환경에서 다양한 제품과의 호환성 검증

클라우드 환경에서는 다양한 S/W들이 사용될 수 있어 **APM 제품도 폭넓은 제품 지원이 중요합니다.**

- Cloud, OS, WAS, DBMS , 프레임워크 등 오픈소스 S/W와 상용 S/W를 넘어 다양하게 지원 할 수 있어야 합니다.
- OpenShift(Kubernetes)와 클라우드 환경에 검증된 APM 제품인지 확인 필요합니다.

오픈소스WAS 와 상용WAS 모두 지원

 JBoss EAP 5~7	 9 버전 이상
 JBoss AS 5, 8이상	 6 버전 이상
 5.5 버전 이상	 8 버전 이상
 1 버전 이상	 17 버전 이상
 1.5 버전 이상	 3 버전 이상

Unix/Linux 모두 지원

지원하는 OS 환경

지원하는 OS	Version
Red Hat Enterprise Linux	5 이상
CentOS	5 이상
Amazon Linux	2016이상
IBM AIX	5 이상
HP-UX	11i 이상
Solaris	10 이상
Windows	2003 이상

지원하는 클라우드 및 컨테이너 환경

Cloud	Service/Version
Public Cloud	AWS, Azure, GCP 지원
Red Hat OpenStack	가상화 지원
OpenShift Container Platform	3.X 이상
Kubernetes	1 이상
Docker	1.6 이상

Public Cloud 와 OpenShift/OpenStack/ Kubernetes 모두 지원

모니터링 가능 플랫폼

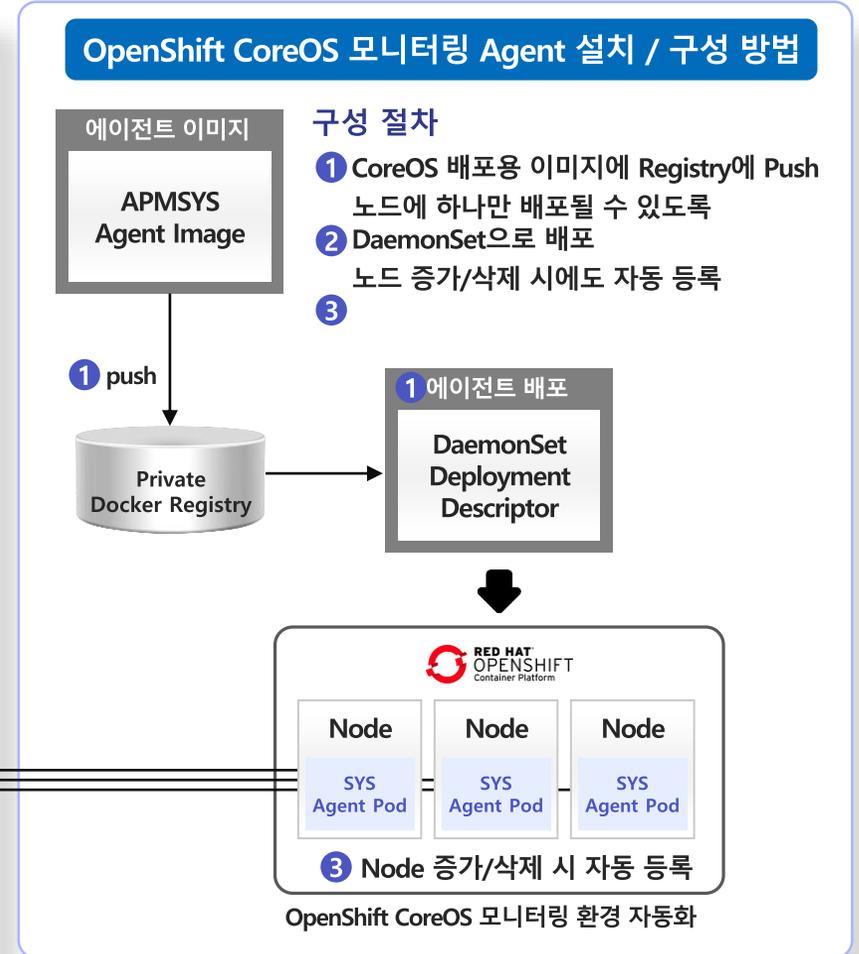
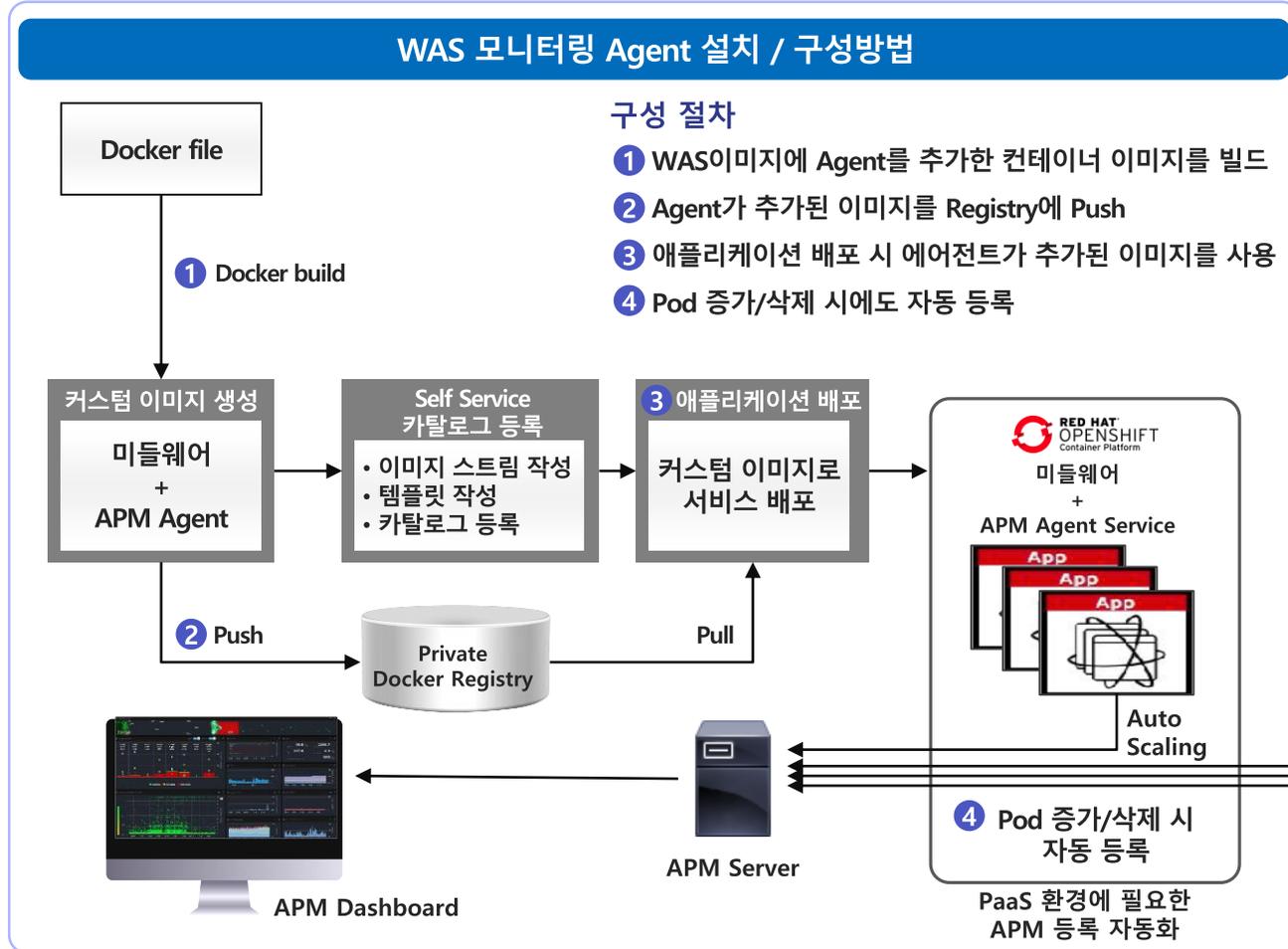


물리 환경	리눅스/가상화	Private Cloud	Public Cloud	컨테이너
Active-X	HTML5	HTML5	HTML5	HTML5
Java EE	Spring /e-Gov	Spring /e-Gov	Spring /e-Gov	Spring /e-Gov
JEUS/ WebLogic	Apache/ Tomcat/JEUS	Apache/ Tomcat/JEUS	Apache /Tomcat/JEUS	Apache/ Tomcat/JEUS
Java	Java	Java	Java	Java
vPar/PowerVM /OracleVM	Red Hat Virtualization			
HP-UX/ AIX/Solaris	Red Hat Linux			
PA-RISC/ Power/SPARC	X86			

컨테이너 환경의 APM 모니터링 구축 방법

컨테이너 기반의 클라우드 환경에 최적화된 모니터링 환경을 구축합니다.

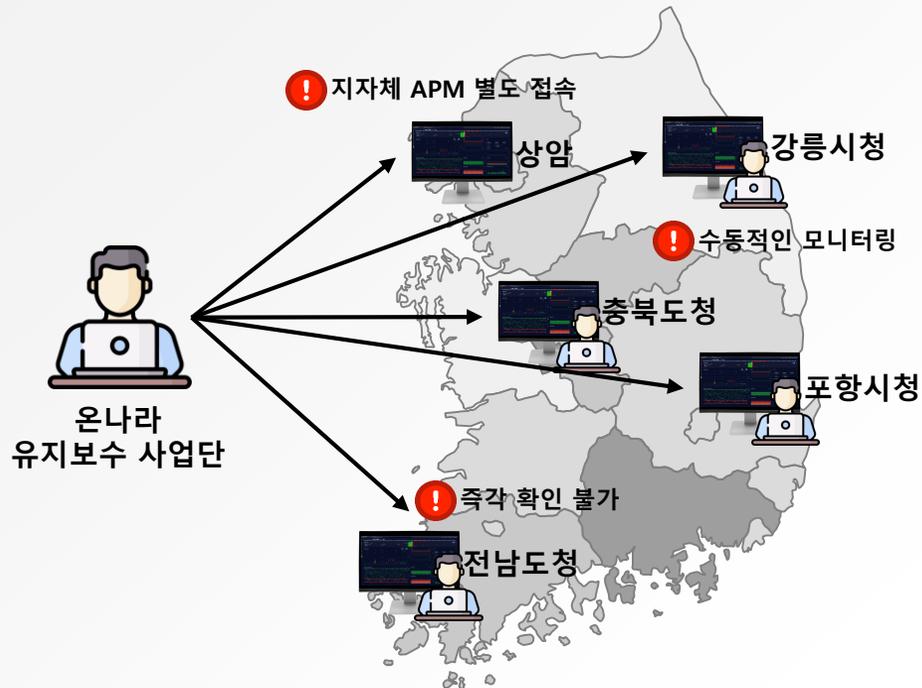
- 컨테이너 이미지에 WAS Agent를 추가하여 Auto-Scaling등으로 Pod 증가/삭제 시에도 에이전트가 자동으로 등록될 수 있도록 구성합니다.
- OpenShift CoreOS 모니터링 및 컨테이너 자원(CPU, Mem, Network등) 모니터링을 위하여 각 노드에 하나씩 모니터링 Agent Pod를 배포합니다.



대규모 통합 모니터링 방안, Cocktail APM 대시보드

대규모 혹은 분산된 모니터링 시스템을 모니터링하기 위한 **통합 모니터링 기능** 제공

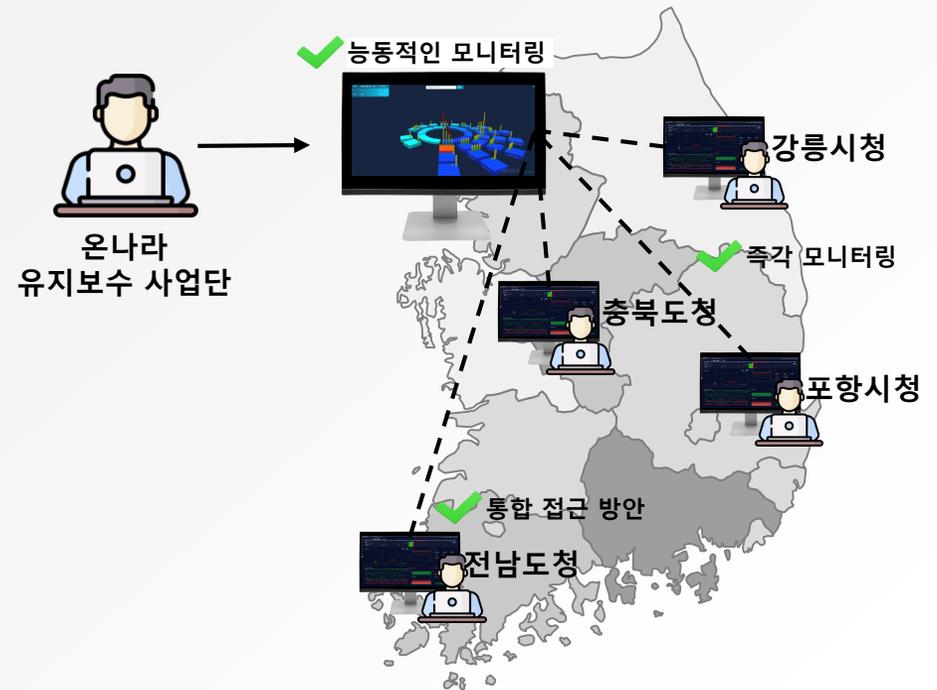
지자체 148개의 모니터링이 분산된 환경



문제점

- 지자체 장애 발생 알림 이후의 수동적인 모니터링
- 각 지자체 APM 확인에 계정, IP 확인 등 불필요한 작업
- 지자체 장애 발생 시 유지보수 사업단에서 즉각 확인 불가
- 정상 운영시에도 지자체별 모니터링 운영자 필요

모든 지자체 모니터링이 통합된 환경



개선점

- 유지보수 사업단의 상시 모니터링으로 능동적인 모니터링
- 통합 대시보드로 지자체 APM과 통합 및 접근 연동
- 장애 발생시 유지보수 사업단의 즉각 지자체 APM 모니터링
- 전 지자체 온나라 업무를 유지보수 사업단의 통합 모니터링

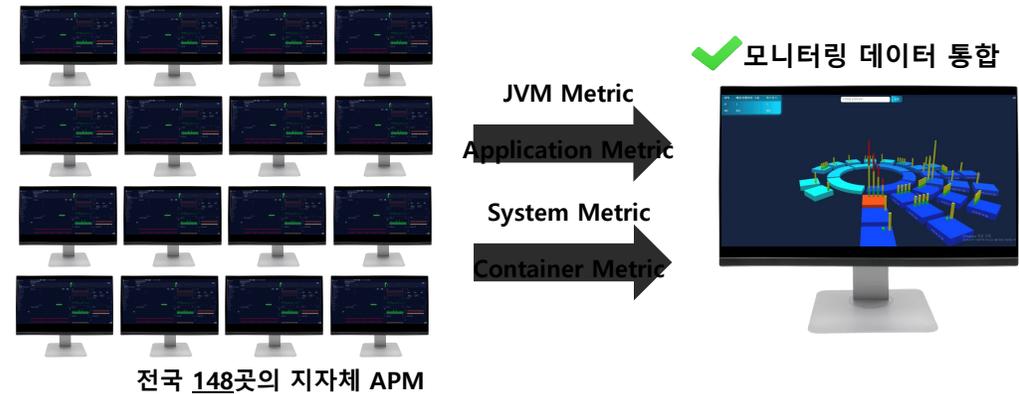
Cocktail APM 대시보드 통합 모니터링 주요 특징

기관별, 부서별 모니터링을 운영하는 **대규모 운영환경에 필요한 통합 모니터링** 특징점 제공

148개 지자체 APM을 한 눈에 볼 수 있는 가시성



148개 지자체 APM의 데이터를 하나로 통합하는 통합성



각 지자체별로 APM 접근에 용이해지는 접근성



지자체, 권역 등 그룹으로 모니터링할 수 있는 편의성



Cocktail APM 대시보드 주요 기능

Cocktail APM들과 통합하여 3가지 형태의 직관적인 대시보드 제공

- Cocktail APM Dashboard는 3가지 형태의 직관적인 대시보드(WAS 인스턴스별, 지자체 별, 전국 단위) 통해 가시성 제공
- 업무 애플리케이션의 그룹 단위에 따라 모니터링을 수행할 수 있는 대시보드
 - ✓ WAS 인스턴스 별로 모니터링을 할 수 있는 Cocktail APM H-MAP 통합 대시보드
 - ✓ 지자체 별로 모니터링할 수 있는 Cocktail APM 일반 통합 대시보드
 - ✓ 전국 단위로 모니터링 할 수 있는 Cocktail APM 3D 통합 대시보드

WAS 인스턴스 별 모니터링 대시보드



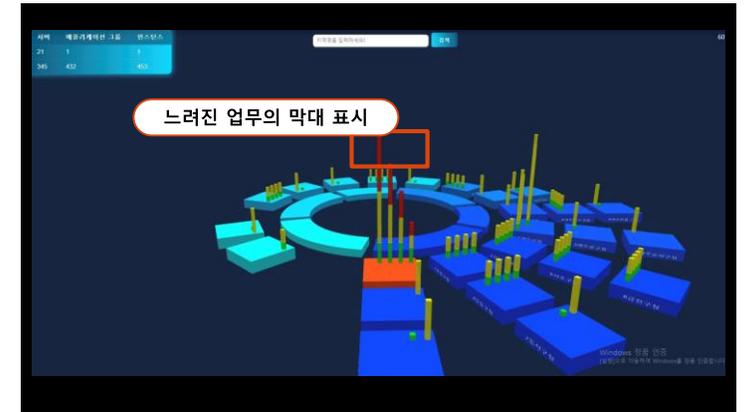
- 인스턴스별 느린 트랜잭션 확인 가능
- 인스턴스 종료 시 알림 기능
- 그룹의 트랜잭션 처리에 따라 배경색 변경
- Cocktail APM H-MAP 통합 대시보드

지자체 별 모니터링 대시보드



- 지자체별 애플리케이션 모니터링 지표 확인 가능
- 그룹의 전체 트랜잭션의 상태 확인 가능
- 그룹의 트랜잭션 처리에 따라 배경색 변경
- Cocktail APM 일반 통합 대시보드

전국 단위 모니터링 대시보드



- 전국 단위의 느린 트랜잭션 현황 파악 가능
- 느린 그룹으로 화면 자동 전환
- 트랜잭션 처리에 따라 직관적으로 파악 가능
- Cocktail APM 3D 대시보드

Cocktail APM 대시보드 도입 효과

분산된 모니터링 시스템을 통합하여 애플리케이션을 **안정적이고 편리하게 모니터링 가능**

왜? Cocktail APM인가?

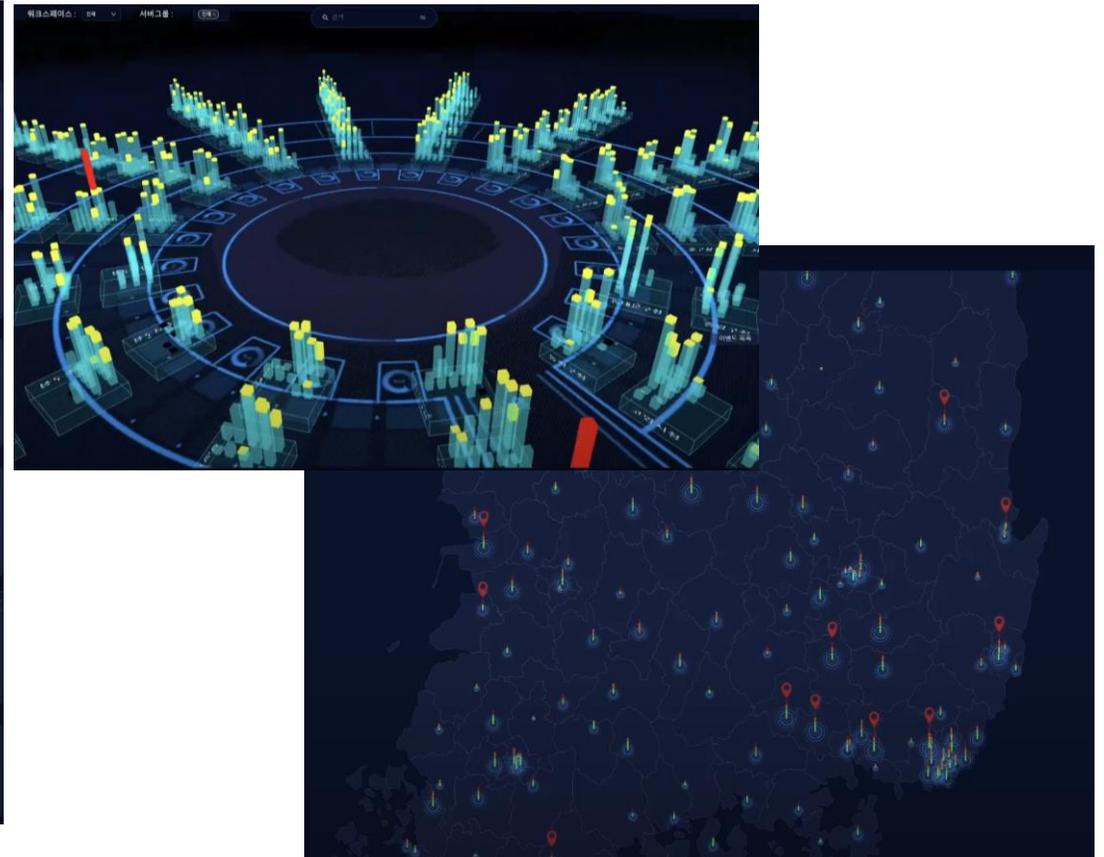


01	정상 상태의 서비스를 지자체 운영자가 모니터링하고 계신가요?	온나라 유지보수 사업단의 통합 모니터링으로 지자체 운영자의 업무 효율성 증대
02	장애 발생을 즉각적인 혹은 미리 인지하지 못하는 수동적인 모니터링을 하고 계신가요?	통합 모니터링으로 장애 인지에 대한 즉각적이고 능동적인 대응
03	지자체 장애 시 모니터링 시스템의 활용에 어려움이 있으신가요?	온나라 유지보수 사업단의 전문적인 모니터링 엔지니어 지원 용이
04	지자체의 업무가 느려지거나 장애가 발생했을 시 증상 설명 등 기술지원 요청에 어려움이 있으신가요?	온나라 유지보수 사업단이 각 지자체 APM에 손쉽게 접근하여 문제 확인 가능

Cocktail APM 주요 Screen Shot

Cocktail APM들과 통합하여 전국 900대 규모 대시보드 제공

- 공공기간에서 운영 중인 서버를 모니터링 하는 것이 아니라 서비스를 모니터링
 - ✓ 서버 등의 인프라는 문제 발생시 즉각적인 현상으로 나타남
 - ✓ 서비스 즉 애플리케이션은 다양한 문제 유형이 있고, 관리자가 인지 하기 어려움
 - ✓ 실제 사용자가 느끼는 불편 사항



Application Performance Management

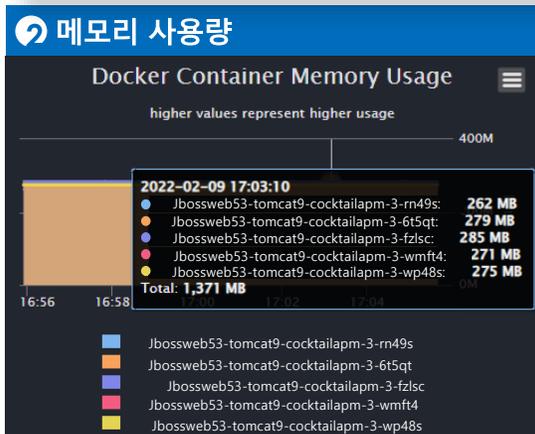
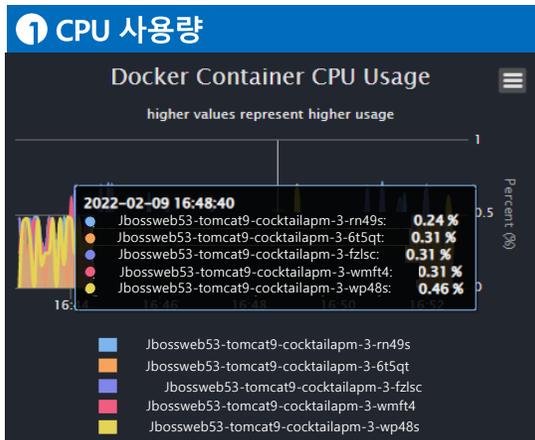
Cocktail APM 제품 기능 소개



컨테이너 지원 - 클라우드 관리도구와 통합된 모니터링 지원

Cocktail APM은 PaaS와 유기적인 연계를 통해 효율적인 운영이 가능하도록 지원합니다.

- 클라우드 운영 시에 **운영 현황과 성능과 장애 이슈를 APM에서 통합적으로 모니터링**할 수 있어야 합니다.
- PaaS 관리 기준인 **프로젝트와 운영 이벤트**를 Cocktail APM을 통해서 손쉽게 확인할 수 있습니다.



애플리케이션 그룹

Jbossweb53-tomcat9-cocktailapm-3

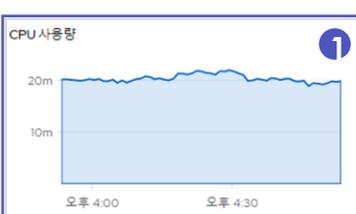
PaaS 웹 콘솔

배로 설정 > 배로 구성 세부 정보

Jbossweb53-tomcat9-cocktailapm PaaS 이벤트

리소스 사용

CPU 사용량



이벤트 스트리밍 중

- Created container, Jbossweb53-tomcat9-cocktailapm
- Started container, Jbossweb53-tomcat9-cocktailapm

pod 별 메모리 사용량 (상위 10개)

Container Name	Memory Usage (MiB)
Jbossweb53-tomcat9-cocktailapm-3-fzjsc	272.2 MiB
Jbossweb53-tomcat9-cocktailapm-3-6t5qt	266.1 MiB
Jbossweb53-tomcat9-cocktailapm-3-wp48s	262.6 MiB
Jbossweb53-tomcat9-cocktailapm-3-wmft4	259 MiB
Jbossweb53-tomcat9-cocktailapm-3-rn49s	249.8 MiB

Cocktail APM

1. 이벤트 리스트

2. 메시지 창

[WARN] eap74-12@test-was1 [192.168.23.193]
WAS가 종료되어 에이전트와 연결이 끊어졌습니다.

[CRITICAL] @test-apm-190 [192.168.23.190]
헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...

[CRITICAL] @test-apm-190 [192.168.23.190]
헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...

[CRITICAL] @test-apm-190 [192.168.23.190]
헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...

[CRITICAL] @test-apm-190 [192.168.23.190]
헬스 체크 오류: 아디다스 > 메인페이지 - HTTP 응답...

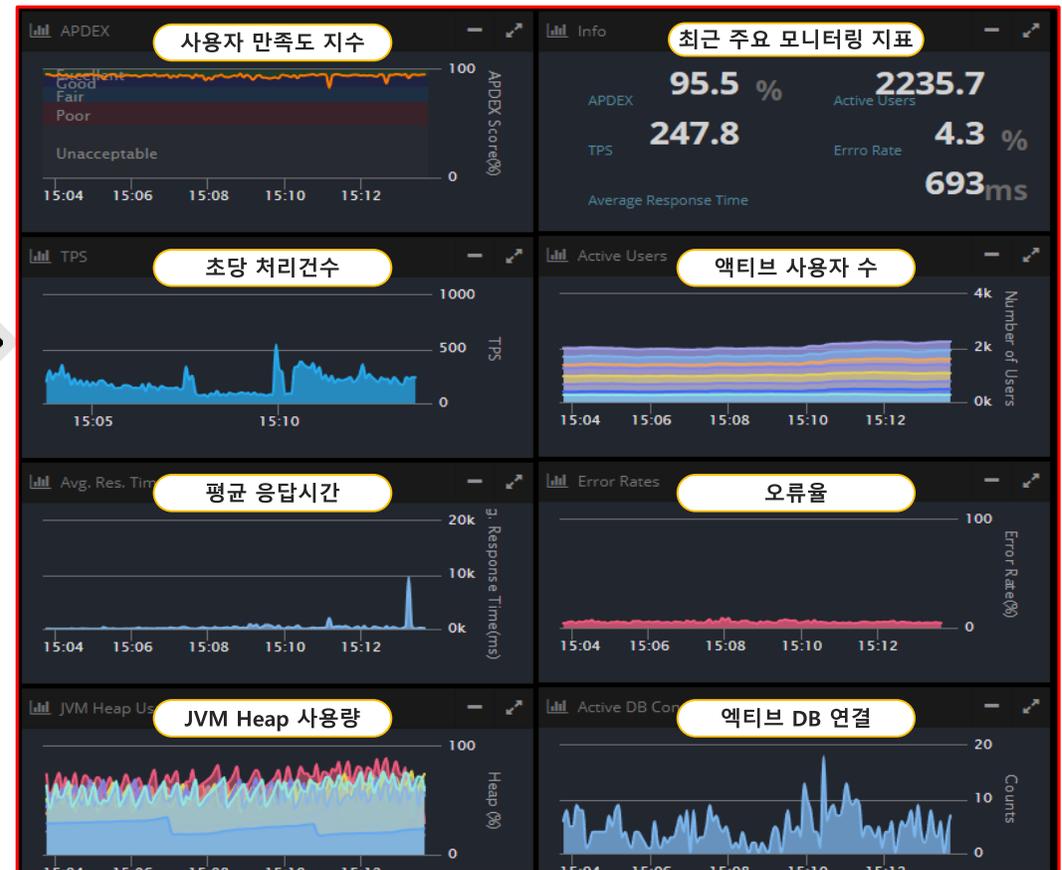
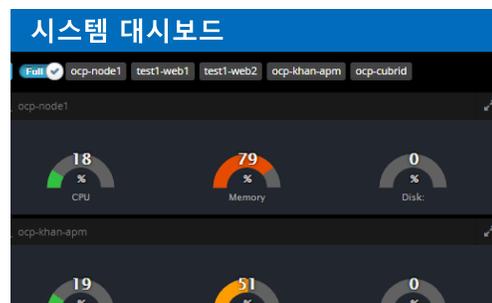
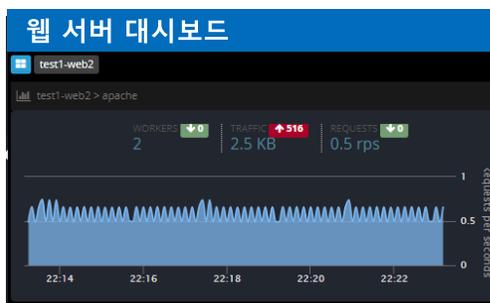
Updated: 2022-01-19 13:34:19.757

Cocktail APM은 **WebSocket API 통신 (양방향) 방식**으로 양방향 지속적으로 연결되어 실시간으로 발생하는 이벤트를 모니터링 가능

1. 실시간 대시보드

대시보드에 다양한(트랜잭션, 리소스 등) 정보를 실시간으로 표시하는 기능 제공

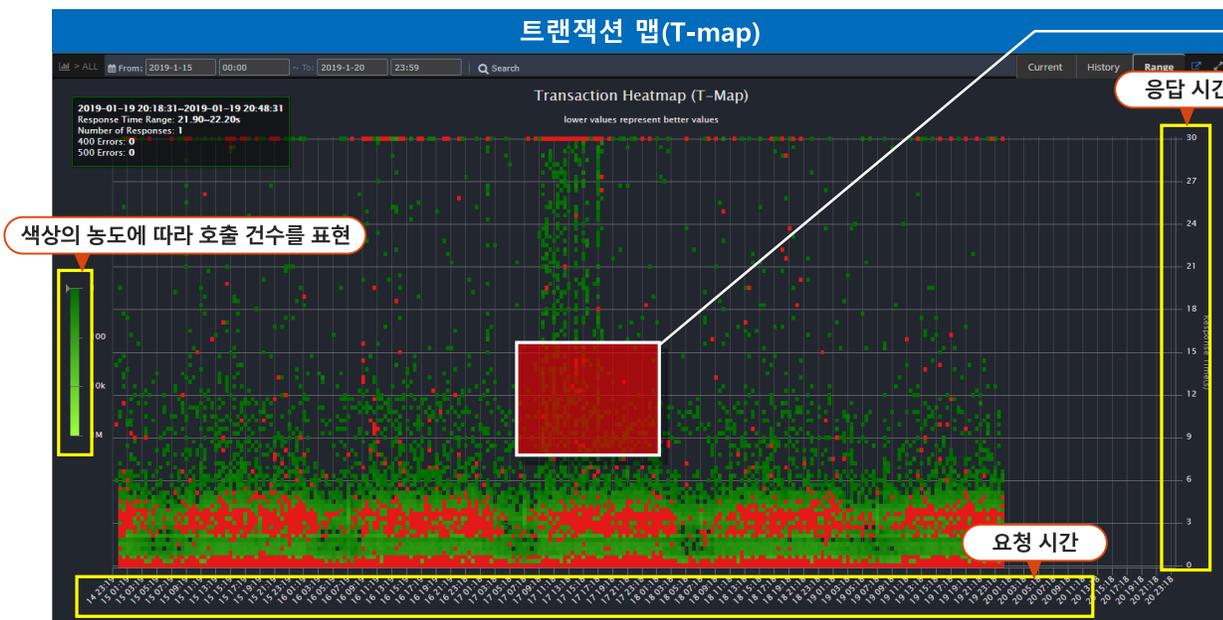
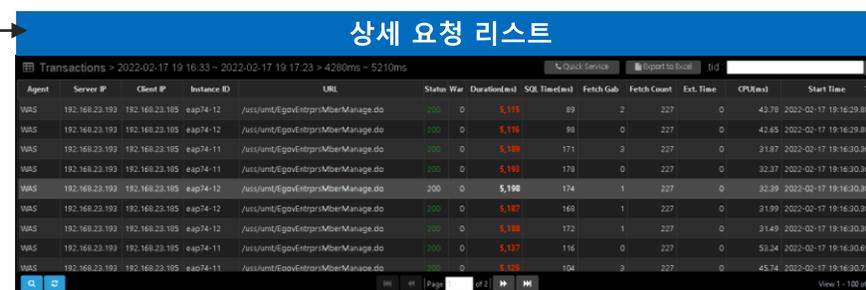
- Cocktail APM 대시보드 및 웹 서버, 시스템 대시보드 기능 제공
- 실시간 요청 모니터링, 액티브 사용자 수, TPS, 지연 트랜잭션, 응답시간, 요청건수 등 다양한 정보를 실시간 표시 기능 제공



2. 개별 트랜잭션 응답 시간 분포도

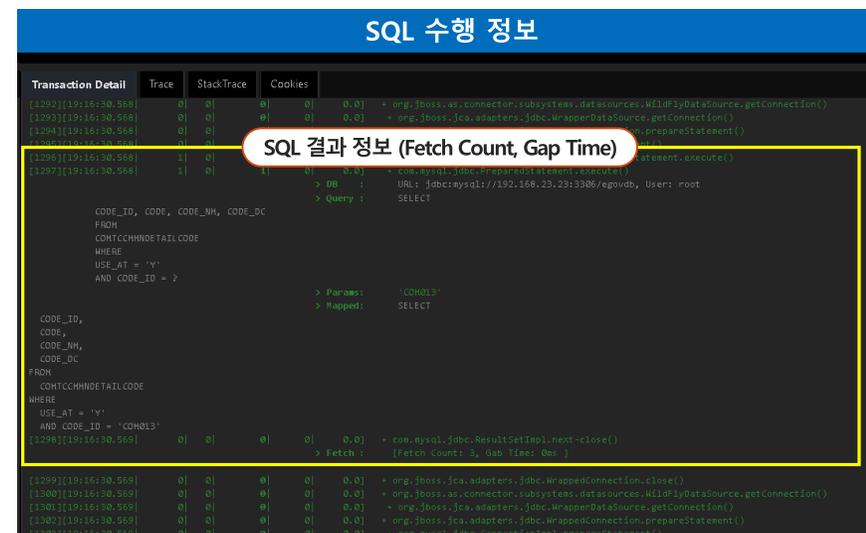
개별 트랜잭션(WAS-DB 구간 등)에 대한 응답 시간 분포도 제공

- Transaction(WAS-DB 구간 등)의 분포도를 확인 할 수 있는 T-Map 기능 제공
- 호출 통계 기능을 통하여 수행 시간이 느린 메소드 및 SQL 쿼리 정보 확인 가능

상세 요청 리스트

Agent	Server IP	Client IP	Instance ID	URL	Status	Duration(ms)	SQL Time(ms)	Fetch Gap	Fetch Count	Est. Time	CPU(ms)	Start Time
WAS	192.168.23.193	192.168.23.195	esp74-12	/ussunt/EgovEntprsMberManage.do	200	0	5,116	89	2	227	0	43.78 2022-02-17 19:16:29.807
WAS	192.168.23.193	192.168.23.195	esp74-12	/ussunt/EgovEntprsMberManage.do	200	0	5,116	98	0	227	0	42.85 2022-02-17 19:16:29.898
WAS	192.168.23.193	192.168.23.195	esp74-11	/ussunt/EgovEntprsMberManage.do	200	0	5,189	171	3	227	0	31.87 2022-02-17 19:16:30.368
WAS	192.168.23.193	192.168.23.195	esp74-11	/ussunt/EgovEntprsMberManage.do	200	0	5,183	179	0	227	0	32.37 2022-02-17 19:16:30.368
WAS	192.168.23.193	192.168.23.195	esp74-12	/ussunt/EgovEntprsMberManage.do	200	0	5,198	174	1	227	0	32.29 2022-02-17 19:16:30.382
WAS	192.168.23.193	192.168.23.195	esp74-12	/ussunt/EgovEntprsMberManage.do	200	0	5,187	169	1	227	0	21.99 2022-02-17 19:16:30.393
WAS	192.168.23.193	192.168.23.195	esp74-12	/ussunt/EgovEntprsMberManage.do	200	0	5,188	172	1	227	0	31.49 2022-02-17 19:16:30.393
WAS	192.168.23.193	192.168.23.195	esp74-11	/ussunt/EgovEntprsMberManage.do	200	0	5,137	116	0	227	0	53.34 2022-02-17 19:16:30.697
WAS	192.168.23.193	192.168.23.195	esp74-11	/ussunt/EgovEntprsMberManage.do	200	0	5,182	104	3	227	0	45.74 2022-02-17 19:16:30.738



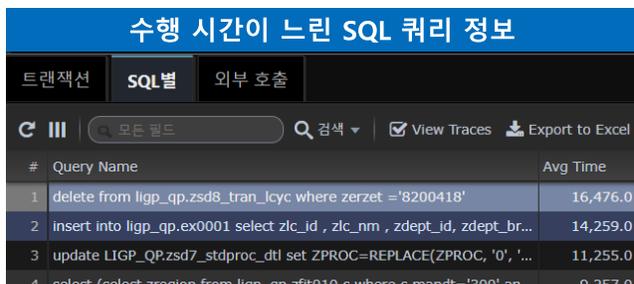
SQL 수행 정보

Transaction Detail	Trace	StackTrace	Cookies
[1292][19:16:30.568]	0	0	0
[1293][19:16:30.568]	0	0	0
[1294][19:16:30.568]	0	0	0
[1295][19:16:30.568]	0	0	0
[1296][19:16:30.568]	1	0	0
[1297][19:16:30.568]	1	0	0
[1298][19:16:30.568]	0	0	0
[1299][19:16:30.568]	0	0	0
[1300][19:16:30.568]	0	0	0
[1301][19:16:30.568]	0	0	0
[1302][19:16:30.568]	0	0	0
[1303][19:16:30.568]	0	0	0
[1304][19:16:30.568]	0	0	0

SQL 결과 정보 (Fetch Count, Gap Time)

```

CODE_ID, CODE, CODE_NH, CODE_DC
COHTCCHHDETAIL CODE
WHERE
USE_AT = 'Y'
AND CODE_ID = 2
> Params: 'COH013'
> Rapped: SELECT
CODE_ID,
CODE,
CODE_NH,
CODE_DC
FROM
COHTCCHHDETAIL CODE
WHERE
USE_AT = 'Y'
AND CODE_ID = 'COH013'
[1298][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.ResultSetImpl.nextClose()
> Fetch: (Fetch count: 2, Gap Time: 0ms)
[1299][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.adapters.jdbc.WrapperConnection.close()
[1300][19:16:30.568] 0 0 0 0 0 0 + org.jboss.as.connector.subsystems.datasources.WildFlyDataSource.getConnection()
[1301][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.adapters.jdbc.WrapperDataSource.getConnection()
[1302][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.adapters.jdbc.WrapperConnection.getConnection()
[1303][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.adapters.jdbc.WrapperConnection.prepareStatement()
[1304][19:16:30.568] 0 0 0 0 0 0 + org.jboss.jdbc.adapters.jdbc.WrapperConnection.prepareStatement()
  
```



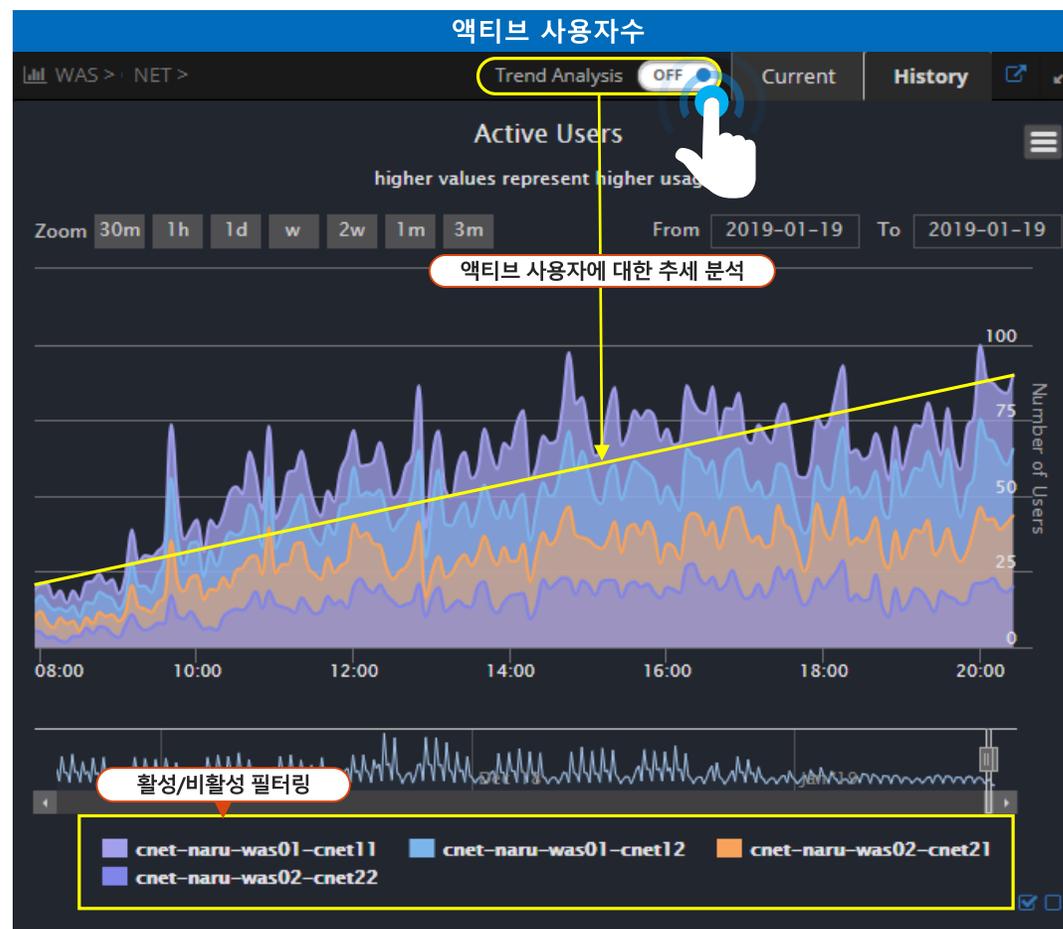
수행 시간이 느린 SQL 쿼리 정보

트랜잭션	SQL별	외부 호출
#	Query Name	Avg Time
1	delete from ligp_qp.zsd8_tran_lcyc where zerset = '8200418'	16,476.0
2	insert into ligp_qp.ex0001 select zlc_id , zlc_nm , zdept_id, zdept_br...	14,259.0
3	update LIGP_QP.zsd7_stdproc_dtl set ZPROC=REPLACE(ZPROC, '0', '...	11,255.0
4	select (select zrcion from ligp_qp.zft010 c where c.mandt='300' an...	9,257.0

3. 동시접속 사용자 실시간 모니터링

동시 접속 사용자, 로그인 사용자 등에 대한 정보를 실시간으로 상세 모니터링 기능 제공

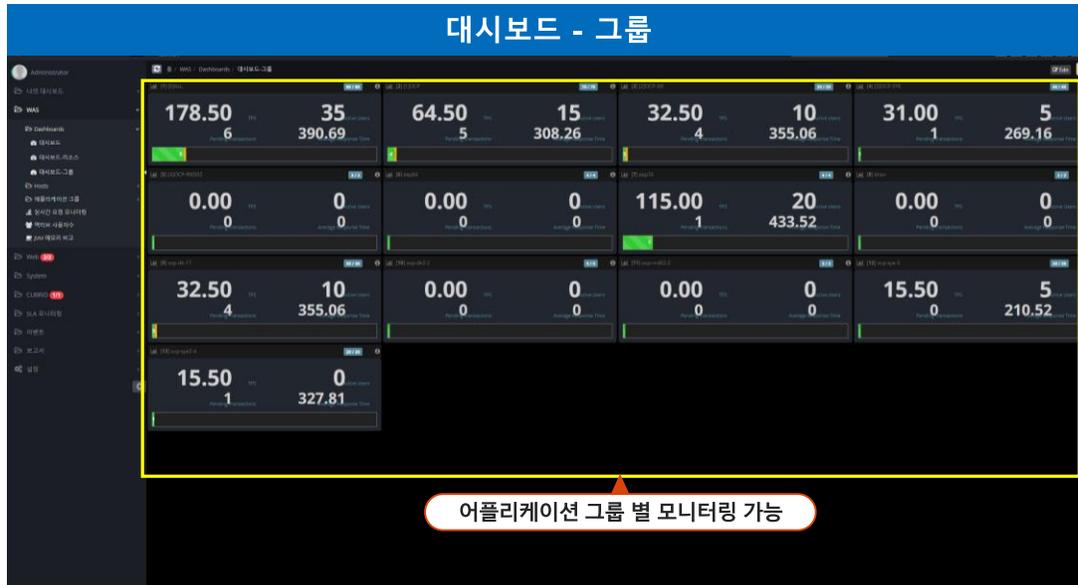
- 애플리케이션을 이용중인 사용자를 실시간으로 표현
- 로그인 아이디를 기반으로 실시간 사용자를 파악 할 수 있는 기능 제공



4. 업무별 애플리케이션 그룹핑 기능

애플리케이션별 업무 설정을 통한 단위 성능 모니터링 기능 제공

- 애플리케이션 그룹관리를 통해서 사용자가 원하는 그룹을 작성
- 개별 인스턴스 뿐만 아니라 사용자 그룹 단위로 모니터링 제공



애플리케이션 업무 설정

애플리케이션 목록

#	Enable	Application Name	Build In	Type	First Date	#	Enable	Ena...	Host Name	IP Address
1	<input checked="" type="checkbox"/>	[System Hosts]	Built-In	SYS		1	<input checked="" type="checkbox"/>	Show	khan-agent-ea...	10.6.3.224
2	<input checked="" type="checkbox"/>	[Web Server Hosts]	Built-In	WEB		2	<input checked="" type="checkbox"/>	Show	khan-agent-ea...	10.6.3.231
3	<input checked="" type="checkbox"/>	[0]ALL				3	<input checked="" type="checkbox"/>	Show	khan-agent-ea...	10.6.5.64
4	<input checked="" type="checkbox"/>	[1]EAP71	User Defined	APP	2018-12-12 11:12:...	4	<input checked="" type="checkbox"/>	Show	khan-agent-ea...	10.6.4.193
5	<input checked="" type="checkbox"/>	[2]EAP72	User Defined	APP	2018-12-14 19:36:...	5	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.5.69
6	<input checked="" type="checkbox"/>	[3]EAP64	User Defined	APP	2018-12-12 11:12:...	6	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.5.63
7	<input checked="" type="checkbox"/>	[e71-khan-agent-eap71...	Built-In	APP	2019-01-19 13:31:...	7	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.5.86
8	<input checked="" type="checkbox"/>	[e712-khan-agent-eap7...	Built-In	APP	2019-01-19 13:32:...	8	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.5.56
9	<input checked="" type="checkbox"/>	[jbosseap64-khanapm3...	Built-In	APP	2019-01-16 23:18:...	9	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.5.96
10	<input checked="" type="checkbox"/>	[jbosseap72-beta-khan...	Built-In	APP	2019-01-19 13:31:...	10	<input type="checkbox"/>	Hide	khan-agent-ea...	10.6.4.195

애플리케이션 그룹관리에서 대상 인스턴스들을 선택



5. 성능 저하 애플리케이션 메소드 레벨 Trace 기능

성능 저하 애플리케이션 목록에 대한 **메소드 레벨의 Trace 분석 기능** 제공

- 일반적인 Trace 뿐만 아니라 사용자가 지정한 클래스/메서드 레벨의 분석 기능을 제공합니다.
- 클래스/메서드 설정 또는 사용자 어노테이션을 사용 할 수 있습니다.

T-Map



트랜잭션 응답시간 분포도 확인

성공/실패를 색상으로 파악 마우스로 드래그

T-Map 상세 Request 리스트

Agent Type	Server IP	Client IP	Instance ID	URL	Status
WAS	10.6.3.187	192.168.23.10	eap64-2929-f	/uss/umt/EgovUserManage.do	302
WAS	10.6.3.183	192.168.23.10	eap64-2929-f	/uat/ua/egovLoginUsr.do	200
WAS	10.6.3.187	192.168.23.10	eap64-2929-f	/uat/ua/egovLoginUsr.do	200
WAS	10.6.3.187	192.168.23.10	eap64-2929-f	/testapp/null.jsp	500
WAS	10.6.3.183	192.168.23.10	eap64-2929-f	/uss/umt/dpt/selectDeptManageLi	302
WAS	10.6.3.183	192.168.23.10	eap64-2929-f	/uss/umt/dpt/selectDeptManageListV	302
WAS	10.6.3.183	192.168.23.10	eap64-2929-f	/testapp/dstest.jsp	300
WAS	10.6.3.183	192.168.23.10	eap64-2929-f	/uss/umt/EgovMberManage.do	302

Request 선택

범용적으로 알려진 메서드 레벨 분석 기능

```

=====
[ Start Time | Elapsed | % | Excludi | A-Gab | CPUtime ]      Method Call
-----
[15:50:10.734| 5,188 | 100 |      0 |      0 | 28.2]+ org.springframework.web.servlet.FrameworkServlet.service()
[15:50:10.734| 5,188 | 100 |      1 | 5,187 | 28.1]+ org.springframework.web.servlet.DispatcherServlet.doService()
[15:50:10.734| 5,184 | 100 | 5,015 | 169 | 24.4]+ @RequestMapping egovframework.com.uss.umt.web.EgovEntrprsManageController
> Parameters: ["Object","Object"]

[15:50:10.734|      0 |      0 |      0 |      0 | 0.0]+ org.jboss.as.connector.subsystems.datasources.WildFlyDataSource.getConnection()
[15:50:10.734|      0 |      0 |      0 |      0 | 0.0]+ org.jboss.jca.adapters.jdbc.WrapperDataSource.getConnection()
[15:50:10.734|      0 |      0 |      0 |      0 | 0.1]+ org.jboss.jca.adapters.jdbc.WrappedConnection.prepareStatement()
[15:50:10.734|      0 |      0 |      0 |      0 | 0.1]+ com.mysql.jdbc.ConnectionImpl.prepareStatement()
[15:50:10.734|      6 |      0 |      6 |      0 | 0.2]+ org.jboss.jca.adapters.jdbc.WrappedPreparedStatement.execute()
[15:50:10.734|      6 |      0 |      6 |      0 | 0.2]+ com.mysql.jdbc.PreparedStatement.execute()

> DB      : URL: jdbc:mysql://192.168.23.23:3306/egovdb, User: root
> Query   : SELECT
            uniqId, userTy, userId , cmpnyNm, userNm, emailAdres, areaNo, middleTelno, endTelno, mobilphonNo, groupId, sttus, sbscrbdE
FROM(
SELECT
    ESNTL_ID          uniqId,
    'USR02'          userTy,
    ENTRPRS_MBER_ID  userId,
    CMPNY_NM         cmpnyNm,
    APPLCNT_NM       userNm,
    APPLCNT_EMAIL_ADRES emailAdres,
    AREA_NO         areaNo,
    ENTRPRS_MIDDLE_TELNO middleTelno,
    ENTRPRS_END_TELNO endTelno,
  
```

사용자 정의 메소드 Tracer

User-interceptor.conf

```

#\uD328\uD0A4\uC9C0 \uC774\uB984\uC744 '/'\uB85C \uCC3E\uC744 \uC218 \uC
#egovframework/let/cop/bbs/service/BoardMaster=/setBb.*//setRe.*

#\uD328\uD0A4\uC9C0 \uC774\uB984\uC744 '/'\uB85C \uCC3E\uC744 \uC218 \uC
#egovframework/let/cop/bbs/service/BoardMaster=/setBb.*//setRe.*

egovframework.com.uss.umt.service.impl.UserManageDAO=selectUserList
  
```

User-interceptor.conf를 통해서 상세 프로파일링 해야 할 클래스 및 메소드명 입력

상세 프로파일링 내역을 출력

회사 내부 규칙의 이름을 가진 메서드 레벨 분석 기능

```

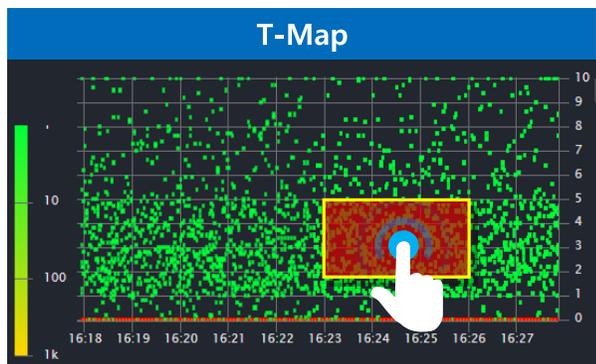
COMMIT
COMMIT
+ org.jboss.jca.adapters.jdbc.WrappedConnection.setAutoCommit(true)
+ org.jboss.jca.adapters.jdbc.WrappedConnection.close()
+ org.jboss.as.connector.subsystems.datasources.WildFlyDataSource.getConnection()
+ org.jboss.jca.adapters.jdbc.WrappedConnection.getConnection()
+ org.jboss.jca.adapters.jdbc.WrappedConnection.prepareStatement()
+ org.jboss.jca.adapters.jdbc.WrappedConnection.setAutoCommit(false)
+ egovframework.com.uss.umt.service.impl.UserManageDAO.selectUserList()
+ org.jboss.jca.adapters.jdbc.WrappedConnection.prepareStatement()
+ org.mariadb.jdbc.MariaDbConnection.prepareStatement()
+ org.jboss.jca.adapters.jdbc.WrappedPreparedStatement.execute()
+ org.jboss.jca.adapters.jdbc.CachedPreparedStatement.execute()
+ org.mariadb.jdbc.MariaDbServerPreparedStatement.execute()
URL: jdbc:mysql://192.168.23.53:3306/gstest
SELECT uniqId, userTy, userId , userNm, emailAdres, areaNo, middleTelno,
EMAIL_ADRES emailAdres, AREA_NO areaNo, HOUSE_MIDDLE_TELNO middleTelno, HOUSE
  
```

상세 프로파일링 내역을 출력

6. DB 및 외부연계 구간 응답시간에 대한 상세 분석

구간별(WAS와 외부연계, WAS와 DB등) 응답시간 기준 상세 분석 기능 제공

- DBMS SQL 질의, 외부 HTTP REST 통신에 대한 분석 기능 제공
- 전체 응답시간 대비 DBMS SQL 시간 및 HTTP REST 통신 시간 모니터링 제공



T-Map 상세

Duration(m)	SQL Time(ms)	Fetch Gab	Fetch Count	Ext. Time	CPU(ms)	Start Time
5,185	166	1	227	0	32.80	2022-02-18 11:0
5,190	172	0	227	0	32.14	2022-02-18 11:0
5,186	167	1	227	0	32.51	2022-02-18 11:0
5,191	169	0	227	0	32.26	2022-02-18 11:0

T-Map 상세 Request 리스트

Transaction Detail | Trace | StackTrace | Cookies

HTTP REST 외부 연계 분석

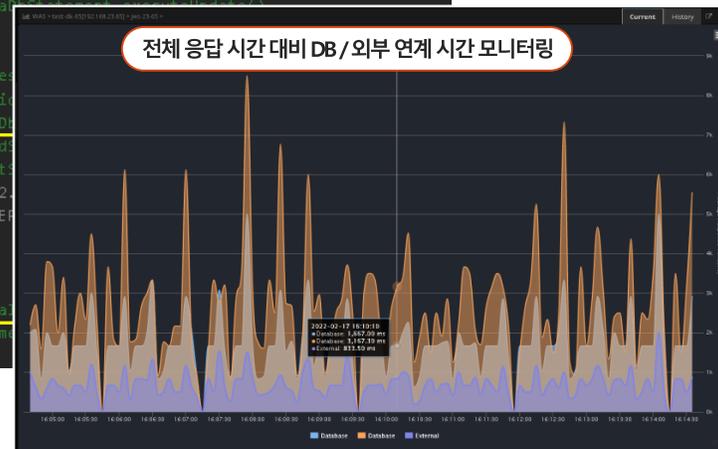
```

[ 5][14:07:25.810] 570| 82| 3| 567| 46.2] + org.apache.http.impl.client.CloseableHttpClient.execute()
[ 6][14:07:25.813] 567| 82| 567| 0| 43.0] + org.apache.http.impl.client.CloseableHttpClient.execute()
  > Infos: GET http://localhost:8080/ 200
  > Child Transaction Detail
[ 7][14:07:26.381] 2| 0| 2| 0| 0.8] + net.sf.log4jdbc.Log4jdbcProxyDataSource.getConnection()
[ 8][14:07:26.383] 0| 0| 0| 0| 0.1] + net.sf.log4jdbc.ConnectionSpy.setAutoCommit(false)
[ 9][14:07:26.383] 0| 0| 0| 0| 0.1] + org.mariadb.jdbc.MariaDbConnection.setAutoCommit(false)
[ 10][14:07:26.383] 0| 0| 0| 0| 0.1] + org.mariadb.jdbc.MariaDbConnection.setAutoCommit(false)
  > Query : set autocommit=0
  > Mapped: set autocommit=0
[ 11][14:07:26.384] 0| 0| 0| 0| 0] + org.springframework.jdbc.datasource.DataSourceUtils.getConnection()
[ 12][14:07:26.406] 10| 1| 6| 0| 0.1] + org.mariadb.jdbc.MariaDbConnection.setAutoCommit(false)
[ 13][14:07:26.406] 4| 1| 4| 0| 4.2] + org.mariadb.jdbc.MariaDbConnection.setAutoCommit(false)
[ 14][14:07:26.416] 3| 0| 1| 2| 2.2] + net.sf.log4jdbc.PreparedStatementWrapper.prepareStatement()
[ 15][14:07:26.417] 2| 0| 2| 0| 1.2] + org.mariadb.jdbc.ClientImpl.prepareStatement()
  > DB : URL: jdbc:mariadb://192.168.1.10:3306
  > Query : /* test 1 */ SELECT USER_ID FROM tn_users
  > Mapped: /* test 1 */
[ 16][14:07:26.425] 0| 0| 0| 0| 0.0] + org.mariadb.jdbc.internal.ResultSetImpl.fetch()
  > Fetch : [Fetch Count: 4, Gab Time: 0.000000]
    
```

DBMS SQL 연계 분석

```

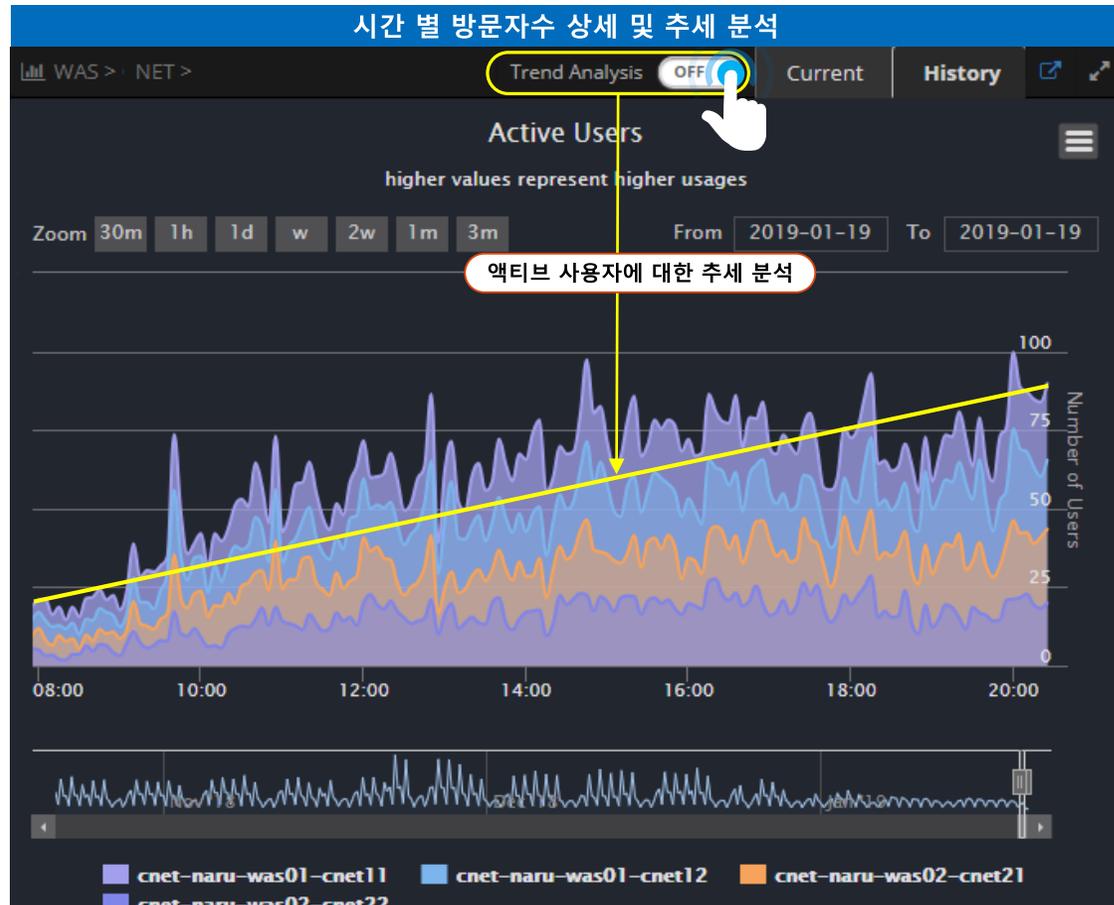
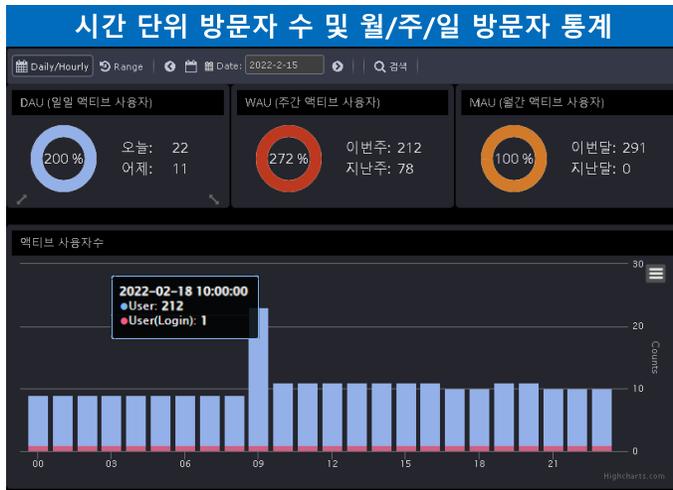
SELECT USER_ID
FROM tn_users
    
```



7. 시간별, 일별 방문자수 측정기능

일일, 시간당 방문자수 및 동시 단말 사용자수에 대한 측정기능 제공 (구글이 사용하는 알고리즘으로 정확한 방문자 수 측정)

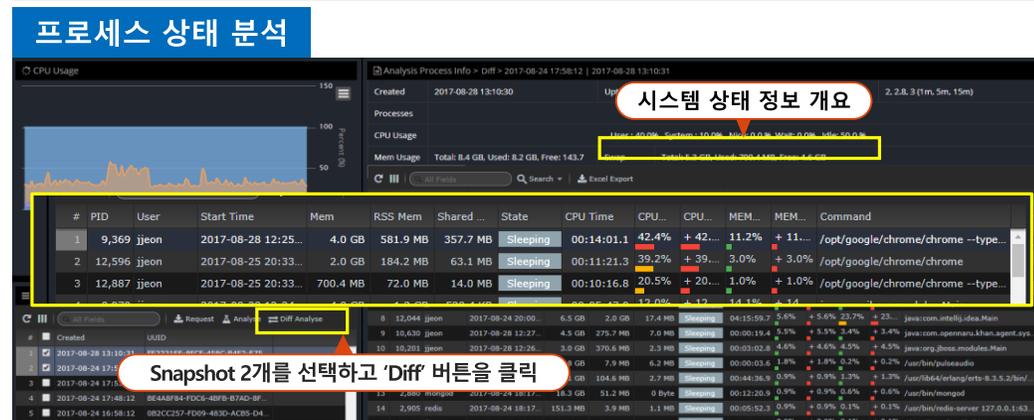
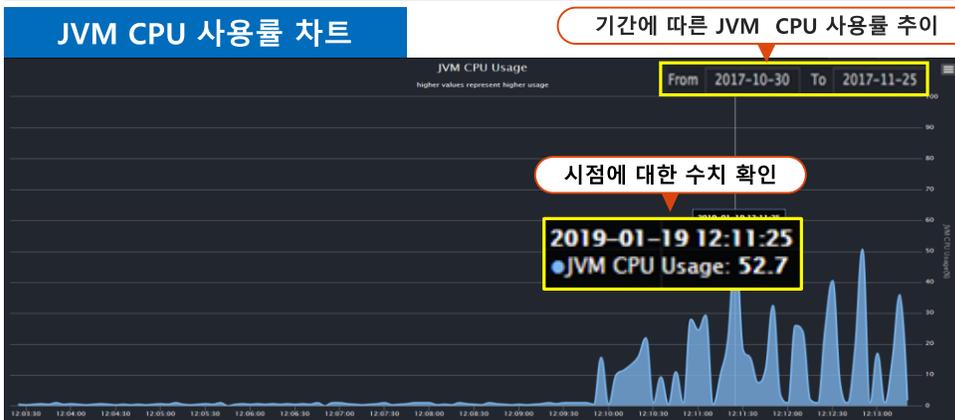
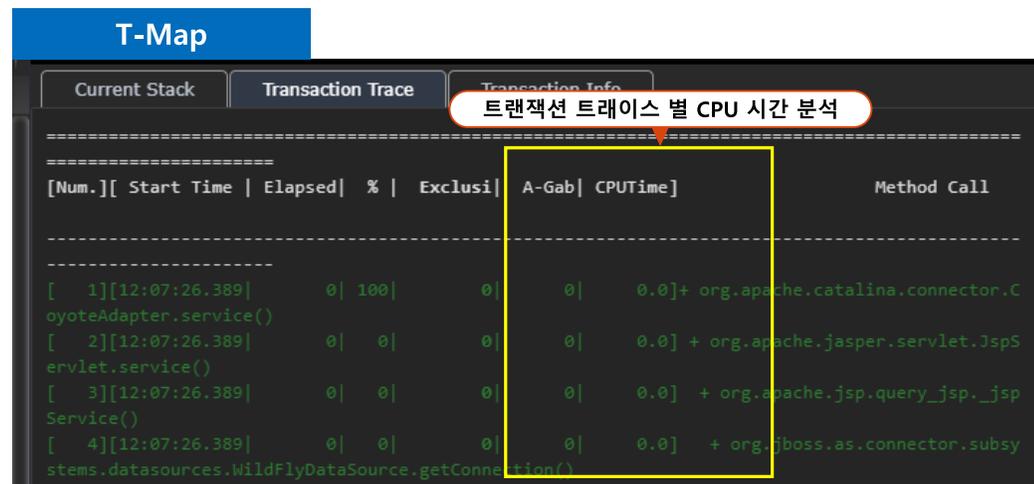
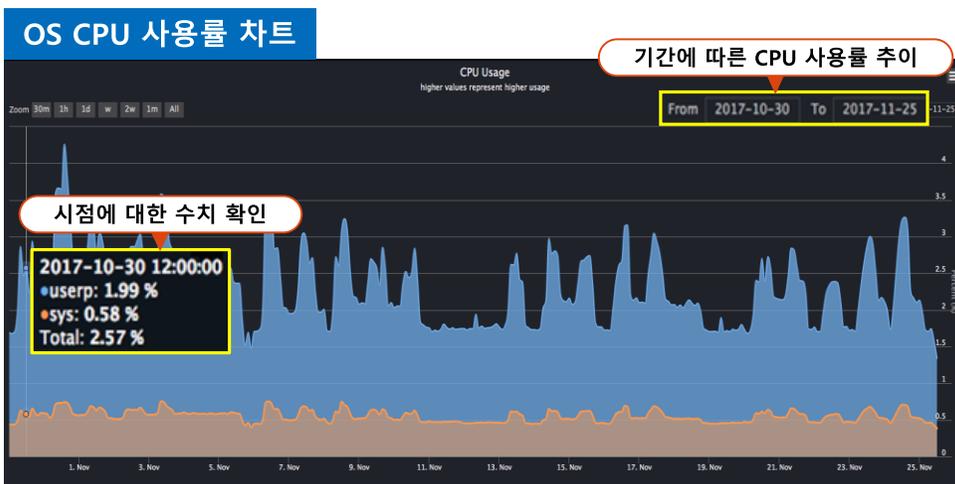
- 최소 초 단위의 방문자 정보를 제공하고, 시간/일/주/월 단위로 추출하여 분석 할 수 있습니다.
- 접속 사용자 뿐만 아니라 로그인 아이디 기반으로 가장 정확한 방문자를 분석 할 수 있습니다.



8. CPU 병목에 대한 추적 기능

실시간 애플리케이션, OS 프로세스별, CPU 사용률 모니터링 기능을 통한 CPU 병목현상 추적기능 제공

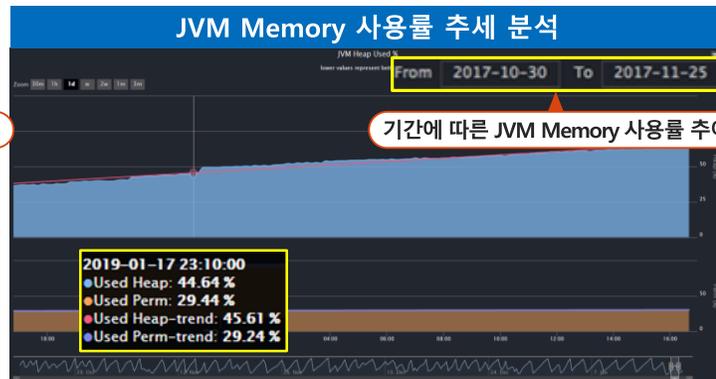
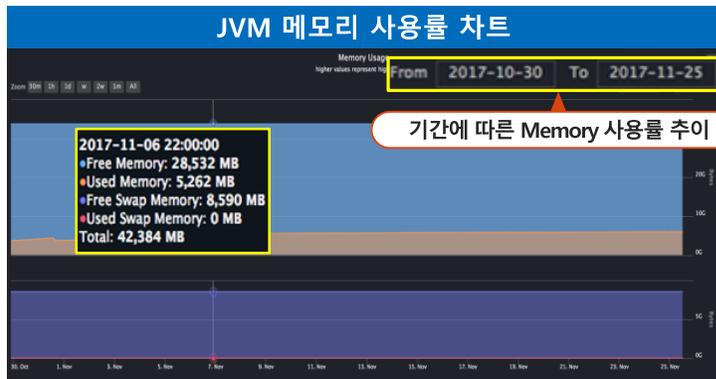
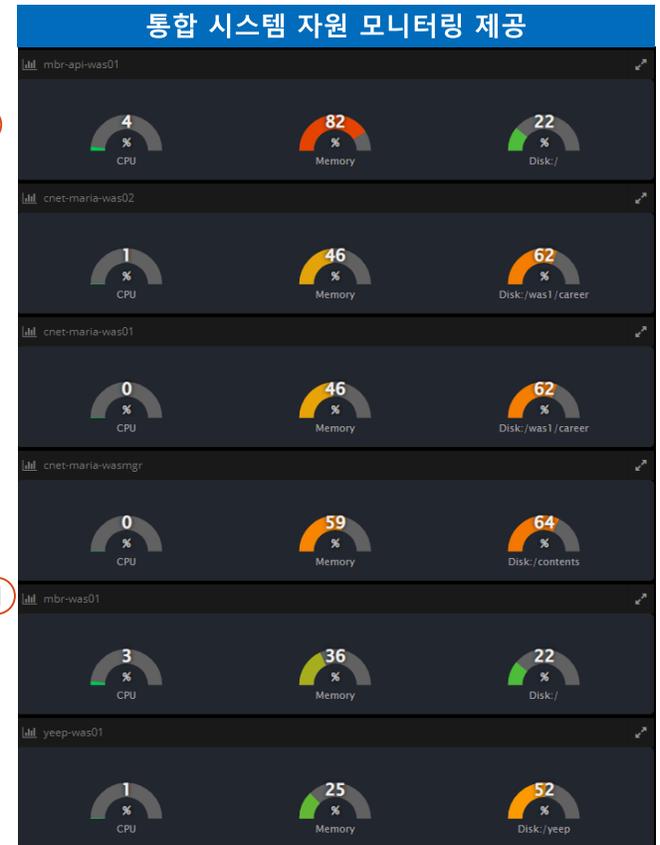
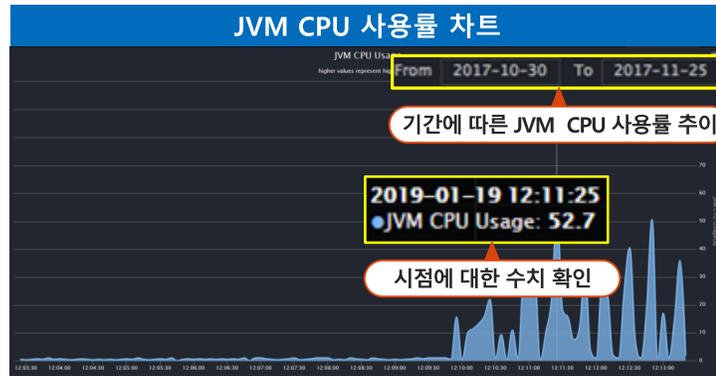
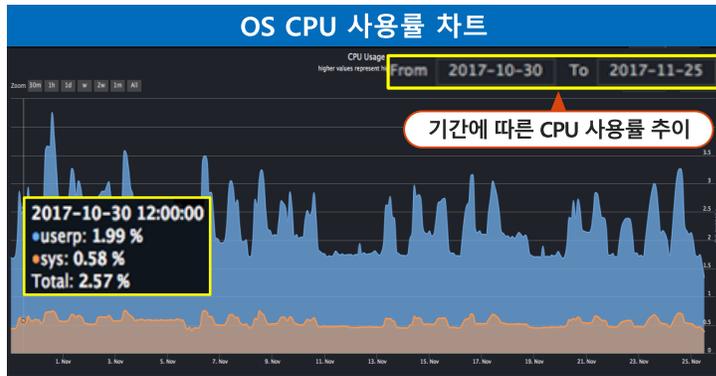
- 실시간 OS CPU 및 JVM CPU 사용률을 제공
- OS Process별 CPU/MEM 정보를 모니터링 하고 분석할 수 있는 기능 제공, 트랜잭션 Trace 레벨까지 CPU 사용 시간 분석 제공



9. WAS의 CPU/메모리 자원 모니터링

WAS 자원(CPU, MEMORY)에 대한 모니터링 기능 제공

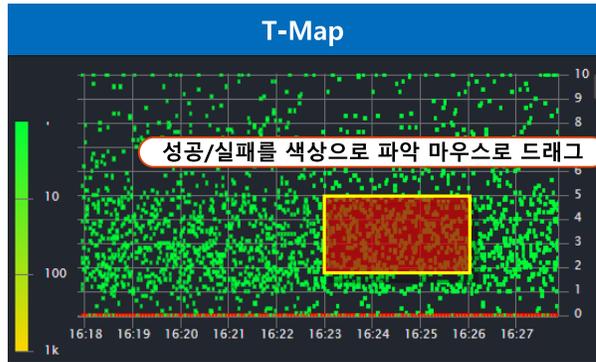
- System Agent를 통해 시스템 자원(CPU, Memory, Network, Disk) 사용량 모니터링 기능을 제공합니다.
- 시스템 OS 영역의 자원 모니터링 뿐만 아니라 WAS 의 JVM CPU 사용량 및 Heap Memory 사용량을 실시간 그래프로 제공합니다.



10. 트랜잭션 및 쿼리별 호출 건 수/응답 시간 등 상세 통계 제공

트랜잭션별 애플리케이션 호출 건 수 및 응답시간, SQL 호출 건 수 및 응답시간 기능 제공

- 호출 통계 및 느린 URL, 느린 SQL 통계 정보 제공



T-Map 상세 Request 리스트

Agent Type	IP address	Instance ID	URL	Status	Duration(m)	SQL Time(ms)	Fetch
WAS	172.31.53.31	ligp-qa-app11	@RequestMapping com.test.ligp.api.batch.web.Batch03C...	200	391,843	10,155	
WAS	172.31.53.31	ligp-qa-app11	@RequestMapping com.test.ligp.api.batch.web.Batch04C...	200	65,853	65,787	
WAS	172.31.53.31	ligp-qa-app11	@RequestMapping com.test.ligp.api.batch.web.Batch02C...	200	2,597	2,581	
WAS	172.31.53.31	ligp-qa-app11	@RequestMapping com.test.ligp.api.batch.web.Batch01C...	200	2,438	2,411	

애플리케이션/SQL 쿼리 튜닝 대상 정보 제공 (홈 > WAS > 트랜잭션 맵(T-Map))

호출 통계

트랜잭션 SQL 별 외부 호출 호출 통계 느린 트랜잭션 리스트

#	URL	Avg Time	Counts
1	+ @RequestMapping com.test.ligp.api.batch.web.Batch03C...	890,305.0	1
2	+ @RequestMapping com.test.ligp.api.batch.web.Batch04C...	51,721.0	1
3	+ @RequestMapping com.test.ligp.api.batch.web.Batch02C...	3,462.0	1
4	+ @RequestMapping com.test.ligp.api.batch.web.Batch01C...	3,210.0	32
5	+ /app/api/jsp/menu.do	220.0	2
6	+ /app/api/jsp/08.do	140.0	1
7	+ /app/api/21.do	121.0	101,305

호출 통계

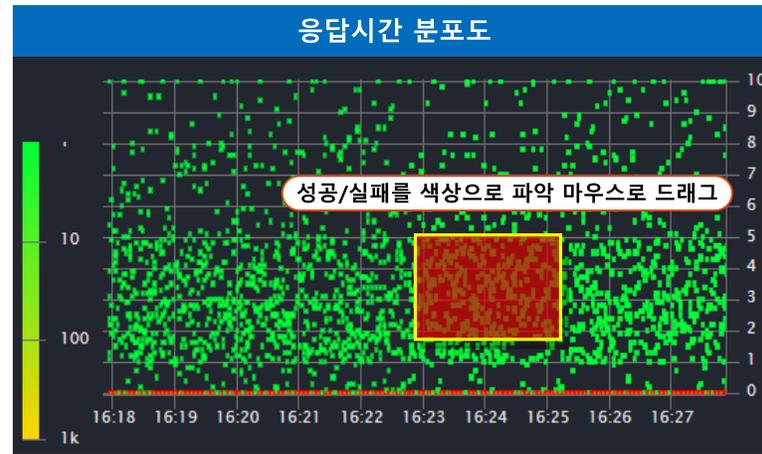
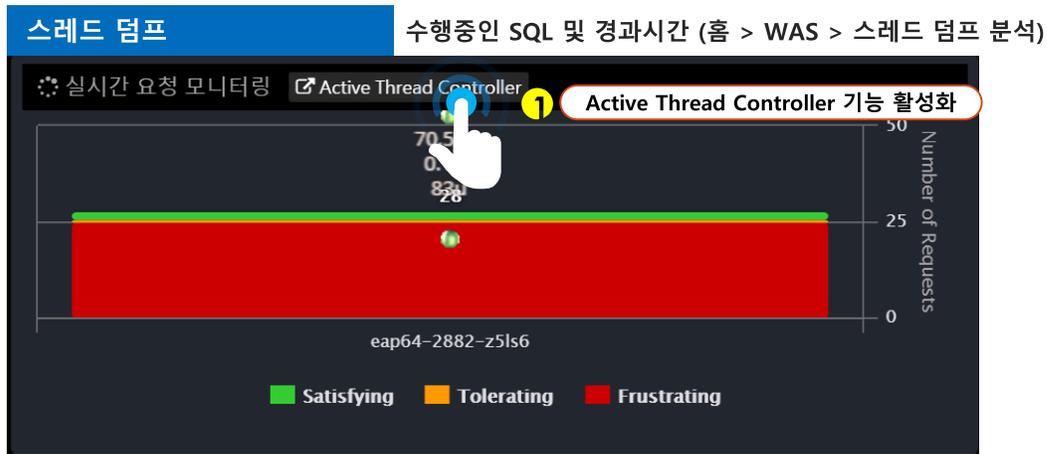
트랜잭션 SQL 별 외부 호출 호출 통계 느린 SQL 리스트

#	Query Name	Avg Time	Counts
1	delete from ligp_qp.zsd8_tran_lcyd where zerset='8200418'	16,476.0	1
2	insert into ligp_qp.ex0001 select '300', a.kunnr, ' ', a.zdept_i...	14,259.0	1
3	update LIGP_QP.zsd7_stdproc_dtl set ZPROC=REPLACE(ZPROC, '0', '...	11,255.0	1
4	select (select zregion from ligp_qp.zfit010 c where c.mandt='300' an...	9,257.0	1
5	insert into ligp_qp.zsd8_tran_lcyd select '300', a.kunnr, ' ', a.zdept_i...	4,833.0	1
6	select a.kunnr as "kunnr", 'AC000' as "stsCd", to_char(now(), 'yyyy...	2,800.0	11
7	insert into ligp_qp.zsd8_lcyd_member select '300', a.zlc_id, a.kunnr, ...	1,753.0	1

11. 애플리케이션별 SQL 처리 상태/응답 시간 모니터링 기능

실시간 애플리케이션 별 SQL 처리 상태 및 응답 시간 모니터링 기능 제공

- 응답 분포도 및 T-Map 상세를 통해 SQL 처리 상태 및 응답 시간 제공
- 지연되는 스레드를 실시간으로 모니터링 하고 분석하는 기능 제공



Active Thread Controller

Active Thread Controller

#	Thread Name	State	URL	Duration(ms)
1	default task-39	BLOCKED	/uat/ua/actionLogin.do	1,365
2	default task-245	BLOCKED	/uat/ua/actionLogin.do	1,210
3	default task-173	BLOCKED	/uat/ua/actionMain.do	1,096
4	default task-197	BLOCKED	/uat/ua/actionMain.do	1,087
5	default task-194	BLOCKED	/uat/ua/actionMain.do	1,075
6	default task-176	BLOCKED	/uat/ua/actionMain.do	1,071
7	default task-117	BLOCKED	/uat/ua/actionLogin.do	1,055
8	default task-174	BLOCKED	/uat/ua/actionMain.do	1,029
9	default task-209	BLOCKED	/uat/ua/actionLogin.do	1,011
10	default task-191	BLOCKED	/uat/ua/actionLogin.do	967
11	default task-203	BLOCKED	/uat/ua/actionMain.do	967
12	default task-5	BLOCKED	/uat/ua/actionMain.do	966
13	default task-60	BLOCKED	/uat/ua/actionMain.do	966
14	default task-211	BLOCKED	/uat/ua/actionMain.do	966
15	default task-160	BLOCKED	/uat/ua/actionMain.do	966
16	default task-81	BLOCKED	/uat/ua/actionMain.do	939

Transaction Trace

```

SELECT emp_mn AS name, password AS password, inidnum AS inidnum, email_adres AS email, 'USR' AS userse, orgntz_id AS orgntzId, esntl_id AS unidId FROM CONTNEMPLYRINFO WHERE emp_mn = ? AND password = ? AND emp_mn_sttus_code = 'P'
    
```

실시간으로 현재 사용중인 Thread 및 SQL 경과 시간 및 상태 출력

T-Map 상세 트레이스

해당 쿼리 수행시간이 느릴 경우 Stacktrace LOG

```

at com.mysql.jdbc.PreparedStatement.execute(PreparedStatement.java:1499)
at org.jboss.jca.adapters.jdbc.WrappedPreparedStatement.execute(WrappedPreparedStatement.java:297)
at org.jboss.jca.adapters.jdbc.WrappedPreparedStatement.execute(WrappedPreparedStatement.java:494)
at com.ibatis.sqlmap.engine.execution.SqlExecutor.executeQuery(SqlExecutor.java:185)
at com.ibatis.sqlmap.engine.mapping.statement.MappedStatement.executeQueryWithCallback(MappedStatement.java:221)
at com.ibatis.sqlmap.engine.mapping.statement.MappedStatement.executeQueryForList(MappedStatement.java:189)
at com.ibatis.sqlmap.engine.impl.SqlMapExecutorDelegate.queryForList(SqlMapExecutorDelegate.java:567)
at com.ibatis.sqlmap.engine.impl.SqlMapExecutorDelegate.queryForList(SqlMapExecutorDelegate.java:541)
at com.ibatis.sqlmap.engine.impl.SqlMapSessionImpl.queryForList(SqlMapSessionImpl.java:118)
at org.springframework.orm.ibatis.SqlMapClientTemplate.doInSqlMapClient(SqlMapClientTemplate.java:295)
at org.springframework.orm.ibatis.SqlMapClientTemplate.doInSqlMapClient(SqlMapClientTemplate.java:1)
    
```

Query 원본 출력 Parameter 자동 맵핑 Query 출력

```

+ com.mysql.jdbc.PreparedStatement.execute
URL: jdbc:mysql://192.168.56.101:3306/
SELECT AUTHOR_CODE AS authorCode FROM
SELECT AUTHOR_CODE AS authorCode FROM
    
```

Fetch Count Fetch Time

```

[Fetch Count: 1, Gab Time: 0ms]
    
```

SCRTY_ESTBS_ID = BIND 변수 자동 맵핑된 모습

```

SCRTY_ESTBS_ID = 'USRCNFRM_00000000001'
    
```

Fetch Count Fetch Time

12. 개별 응답 시간 분포 그래프 기능 제공

서비스 사용현황 분석 및 확인할 수 있도록 개별 응답 시간 분포 그래프 기능 제공

- 응답 시간 분포도를 제공하고 특정 Request 를 필터링할 수 있는 기능 제공합니다.

Duration

응답 시간에 따른 필터링 기능 제공

Client IP

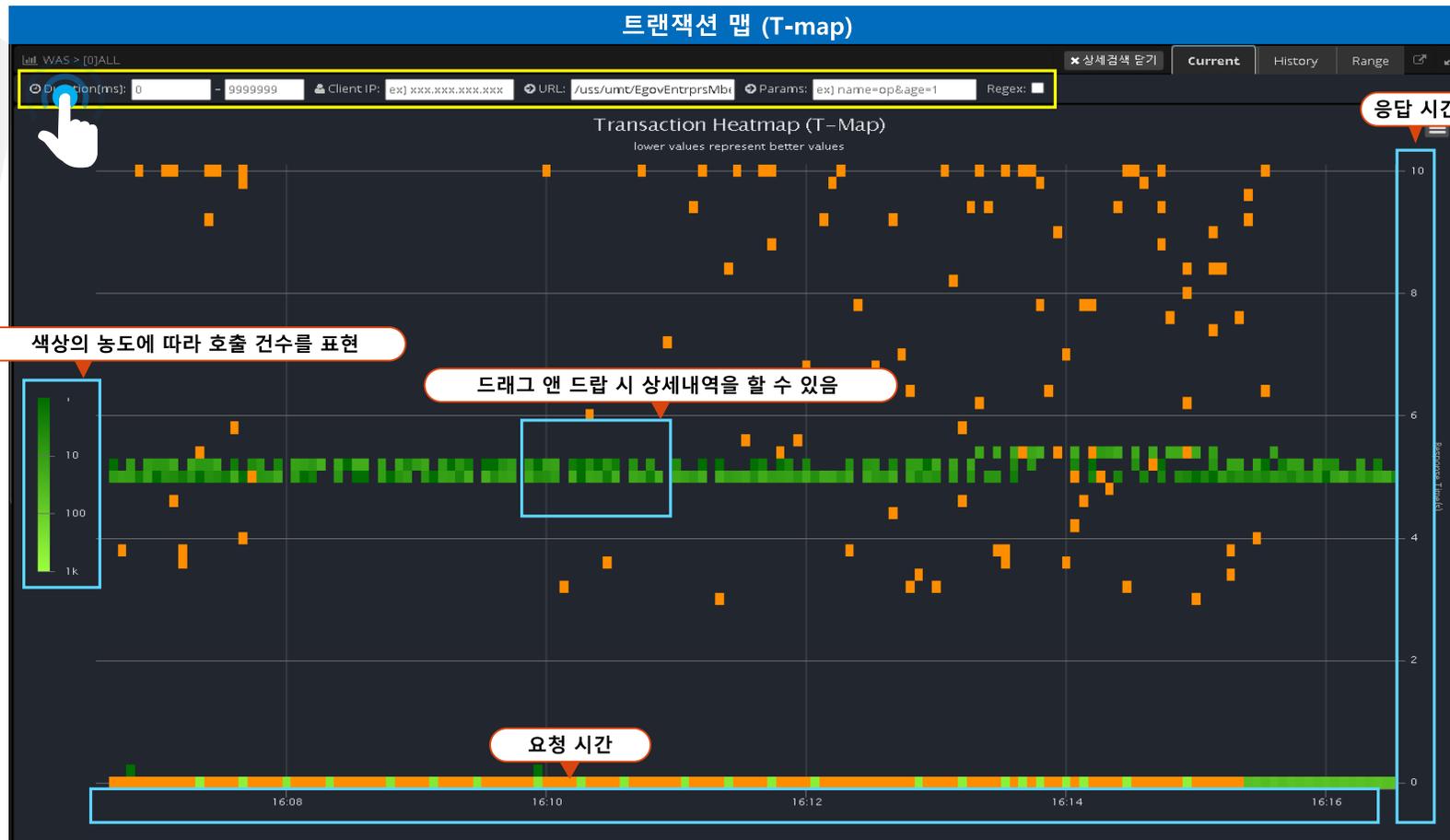
클라이언트 IP 지정을 통한 특정 IP의 Request transaction 제공

URL

URL 필터링을 통한 특정 URL Request만의 Transaction 제공

Parameter

Params 필터링을 통하여 특정 Parameter값의 Transaction 제공



13. 전체 에이전트 목록 및 활성화 상태 모니터링

관리 화면을 통해 전체 Agent 목록 확인 및 활성화 상태 모니터링 기능 제공

- 애플리케이션 그룹 관리를 통해 사용자 그룹핑 및 에이전트 활성화/비활성화 기능을 제공합니다.
- 실시간 모니터링 시 에이전트를 필터링 할 수 있는 기능을 제공합니다.

애플리케이션 그룹 관리

애플리케이션 목록

애플리케이션 그룹 관리 및 에이전트 활성화 여부 및 제어 기능

#	Ena...	Application Nam...	Type	Name	IP Address	Instance ID	Type	Status	First Date				
1	<input checked="" type="checkbox"/>	[System Hosts]	Built-In	SYS		1	<input checked="" type="checkbox"/> Show	ocp-rnd02-jws...	10.128.0.156	ocp-rnd02-2-k...	WAS	RUNNING	2022-02-13 06:45:20
2	<input checked="" type="checkbox"/>	[Web Server Hosts]	Built-In	WEB		2	<input checked="" type="checkbox"/> Show	ocp-rnd02-jws...	10.128.0.158	ocp-rnd02-2-8...	WAS	RUNNING	2022-02-13 07:43:09
3	<input checked="" type="checkbox"/>	[0]ALL	User D...	APP		3	<input type="checkbox"/> Show	ocp-rnd02-jws...	10.128.0.159	ocp-rnd02-2-q...	WAS	RUNNING	2022-02-13 07:40:41
4	<input checked="" type="checkbox"/>	[1]OCP	User D...	APP	2022-02-10 10:15:48	2022-02-17 16:1...	<input type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.200	ocp-dk-17-zm...	WAS	RUNNING	2022-02-10 10:15:51
5	<input checked="" type="checkbox"/>	[2]OCP-DK	User D...	APP			<input type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.201	ocp-dk-17-w7...	WAS	RUNNING	2022-02-12 04:45:05
6	<input checked="" type="checkbox"/>	[3]OCP-SYE	User D...	APP			<input type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.202	ocp-dk-17-kc6...	WAS	RUNNING	2022-02-14 03:30:04
7	<input checked="" type="checkbox"/>	[4]OCP-RND02	User D...	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.203	ocp-dk-17-fzw...	WAS	RUNNING	2022-02-14 09:30:10
8	<input checked="" type="checkbox"/>	[eap64]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.204	ocp-dk-17-qjbb...	WAS	RUNNING	2022-02-14 05:20:00
9	<input checked="" type="checkbox"/>	[eap74]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.205	ocp-dk-17-nt47r...	WAS	STOPPED	2022-02-14 09:56:41
10	<input checked="" type="checkbox"/>	[khan]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.206	ocp-dk-17-9s4...	WAS	RUNNING	2022-02-12 04:45:57
11	<input checked="" type="checkbox"/>	[ocp-dk-17]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.207	ocp-dk-17-b5tdr...	WAS	RUNNING	2022-02-10 10:15:55
12	<input checked="" type="checkbox"/>	[ocp-dk2-2]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.208	ocp-dk-17-hbq...	WAS	RUNNING	2022-02-12 04:44:30
13	<input checked="" type="checkbox"/>	[ocp-rnd02-2]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.209	ocp-dk-17-9q7...	WAS	RUNNING	2022-02-14 03:03:37
14	<input checked="" type="checkbox"/>	[ocp-sye-3]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.210	ocp-dk-17-njz27...	WAS	STOPPED	2022-02-10 10:16:32
15	<input checked="" type="checkbox"/>	[ocp-sye2-4]	Built-In	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.211	ocp-dk-17-c66gf...	WAS	RUNNING	2022-02-14 03:44:16
16	<input checked="" type="checkbox"/>	TEST	User D...	APP			<input checked="" type="checkbox"/> Show	ocp-dk-jws56...	10.128.0.212	ocp-dk-17-dgn...	WAS	STOPPED	2022-02-13 23:46:07

애플리케이션 그룹 관리

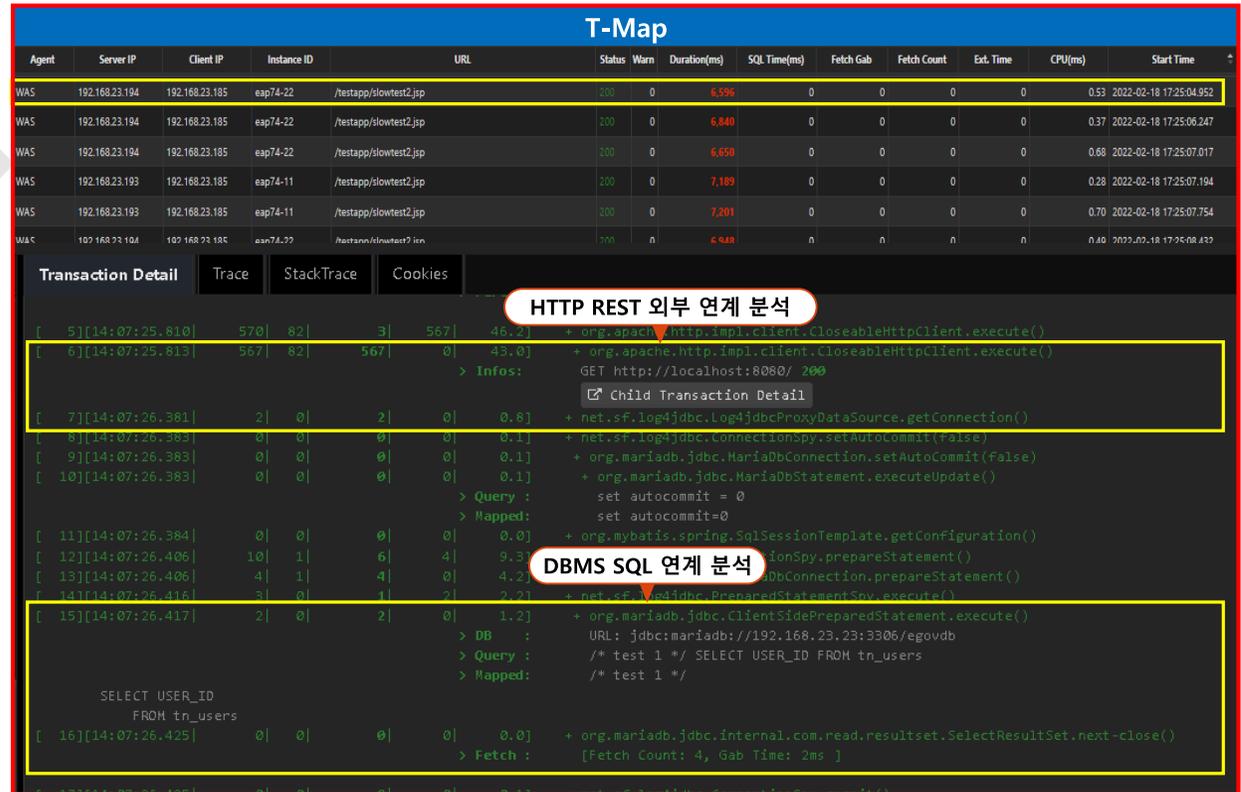
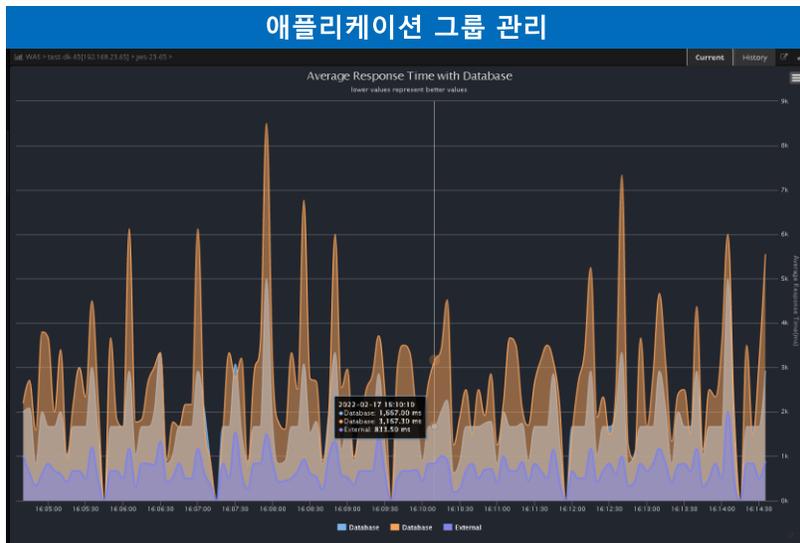
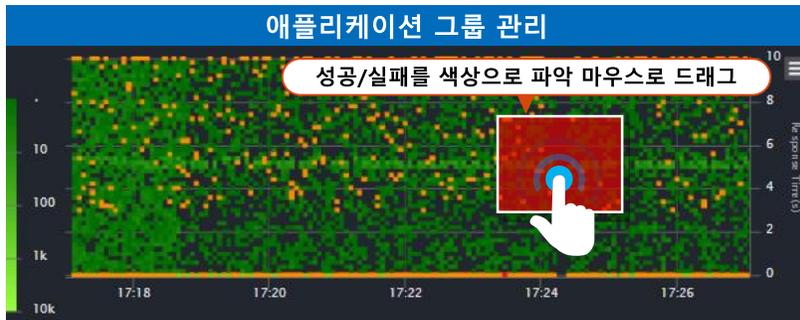
애플리케이션 그룹별 에이전트 활성화 상태 모니터링

api-second-5-2vndg [10.128.2.93]	api-se-5-2vndg (7.9) ▾
api-first-8-wqtz5 [10.131.1.137]	api-fi-8-wqtz5 (7.9) ▾
api-gateway-12-vlzmz [10.131.1.149]	api-ga-12-vlzmz (7.9) ▾
api-gateway-12-8shq6 [10.131.1.41]	api-ga-12-8shq6 (7.9) ▾
'egov' Group [BuiltIn]	
egov-5-8p29q [10.131.1.143]	egov-5-8p29q (7.9) ▾
'groupware' Group [BuiltIn]	
localhost [192.168.23.174]	groupware22 (d89f8) ▾
'infinispan' Group [BuiltIn]	
localhost [100.100.30.200]	ispn11 (7.3) ▾
'jws56-1' Group [BuiltIn]	
jws56-1-xsscs [10.131.0.35]	jws56-1-xsscs (7.8) ▾

14. 응답 시간 분포도 모니터링

실시간 DB, WAS, 외부 연계 서비스의 응답 시간 분포도 모니터링 기능 제공

- DBMS SQL 질의, 외부 HTTP REST 통신에 대한 분석 기능 제공합니다.
- 전체 응답시간 대비 DBMS SQL 수행 시간 및 HTTP REST 통신 수행 시간에 대한 정보를 제공합니다.



T-Map

Agent	Server IP	Client IP	Instance ID	URL	Status	Warn	Duration(ms)	SQL Time(ms)	Fetch Gab	Fetch Count	Ext. Time	CPU(ms)	Start Time
WAS	192.168.23.194	192.168.23.185	eap74-22	/testapp/slowtest2.jsp	200	0	6,996	0	0	0	0	0.53	2022-02-18 17:25:04.952
WAS	192.168.23.194	192.168.23.185	eap74-22	/testapp/slowtest2.jsp	200	0	6,840	0	0	0	0	0.37	2022-02-18 17:25:06.247
WAS	192.168.23.194	192.168.23.185	eap74-22	/testapp/slowtest2.jsp	200	0	6,650	0	0	0	0	0.68	2022-02-18 17:25:07.017
WAS	192.168.23.193	192.168.23.185	eap74-11	/testapp/slowtest2.jsp	200	0	7,189	0	0	0	0	0.28	2022-02-18 17:25:07.194
WAS	192.168.23.193	192.168.23.185	eap74-11	/testapp/slowtest2.jsp	200	0	7,201	0	0	0	0	0.70	2022-02-18 17:25:07.754
WAS	102.168.23.104	102.168.23.185	eap74-22	/testapp/slowtest2.jsp	200	0	6,948	0	0	0	0	0.40	2022-02-18 17:25:08.432

Transaction Detail

HTTP REST 외부 연계 분석

```
[ 5][14:07:25.810] 570| 82| 3| 567| 46.8| + org.apache.http.impl.client.CloseableHttpClient.execute()
[ 6][14:07:25.813] 567| 82| 567| 0| 43.0| + org.apache.http.impl.client.CloseableHttpClient.execute()
  > Infos:
    GET http://localhost:8080/ 200
  > Child Transaction Detail
[ 7][14:07:26.381] 2| 0| 2| 0| 0.8| + net.sf.log4jdbc.Log4jdbcProxyDataSource.getConnection()
[ 8][14:07:26.383] 0| 0| 0| 0| 0.1| + net.sf.log4jdbc.ConnectionSpy.setAutoCommit(false)
[ 9][14:07:26.383] 0| 0| 0| 0| 0.1| + org.mariadb.jdbc.MariaDbConnection.setAutoCommit(false)
[10][14:07:26.383] 0| 0| 0| 0| 0.1| + org.mariadb.jdbc.MariaDbStatement.executeUpdate()
  > Query :
    set autocommit = 0
    set autocommit=0
[11][14:07:26.384] 0| 0| 0| 0| 0.0| + org.mybatis.spring.SqlSessionTemplate.getConfiguration()
[12][14:07:26.406] 10| 1| 6| 4| 9.3| + org.mybatis.spring.SqlSessionTemplate.getConnectionSpy().prepareStatement()
[13][14:07:26.406] 4| 1| 4| 0| 4.2| + org.mariadb.jdbc.MariaDbConnection.prepareStatement()
[14][14:07:26.416] 3| 0| 1| 2| 2.2| + net.sf.log4jdbc.PreparedStatementSpy.execute()
[15][14:07:26.417] 2| 0| 2| 0| 1.3| + org.mariadb.jdbc.ClientSidePreparedStatement.execute()
  > DB :
    URL : jdbc:mariadb://192.168.23.23:3306/egovdb
  > Query :
    /* test 1 */ SELECT USER_ID FROM tn_users
  > Mapped:
    /* test 1 */
[16][14:07:26.425] 0| 0| 0| 0| 0.0| + org.mariadb.jdbc.internal.com.read.resultset.SelectResultSet.next-close()
  > Fetch :
    [Fetch Count: 4, Gab Time: 2ms]
```

DBMS SQL 연계 분석

알람 설정 및 이메일/SMS 연계 기능

임계치 설정을 통한 알람 발생 및 SMS, E-Mail 연계 기능 제공

- 모니터링되는 지표는 임계치 설정을 통해 알림 기능을 제공합니다.
- 브라우저, E-mail, SMS, HTTP REST 등 연계 기능을 제공합니다.

모니터링 항목별 임계치 설정

임계치 설정

WAS Web System

임계치 조절을 통한 운영 환경에 알맞는 알림 설정

지정된 시간의 평균값을 기반으로 경고 정책 설정

Warning Threshold * 3 minutes

Critical Threshold * 1 minute

Alerts to * admin x

Alerts to(Group) *

중복된 이벤트가 계속 발생하지 않도록 설정

Prevent Duplicated 5 minutes

5분 후의 값을 미리 예측

Next X at 5 minutes

극단치에 대한 이벤트 설정

Sigma 2.5 times (98.75%)

임계치 도달시 알람 발생

항목	설명
대상	WAS, WEB, SYS
일반	INFO, WARN, CRITICAL, FORECAST
예측	앞을 예측하여 통지
극단치	평소와 다른 이상 징후 통지

Event Count Chart

Event List

Normal Bell-shaped Curve

정규 분포상 극단에 있는 값이 나올 확률은 낮음 (극단치)

대부분의 데이터는 통계적으로 일정한 범위 내에 존재함 (정규 분포)

Email/SMS/Messenger 알람 연계

[SMS]

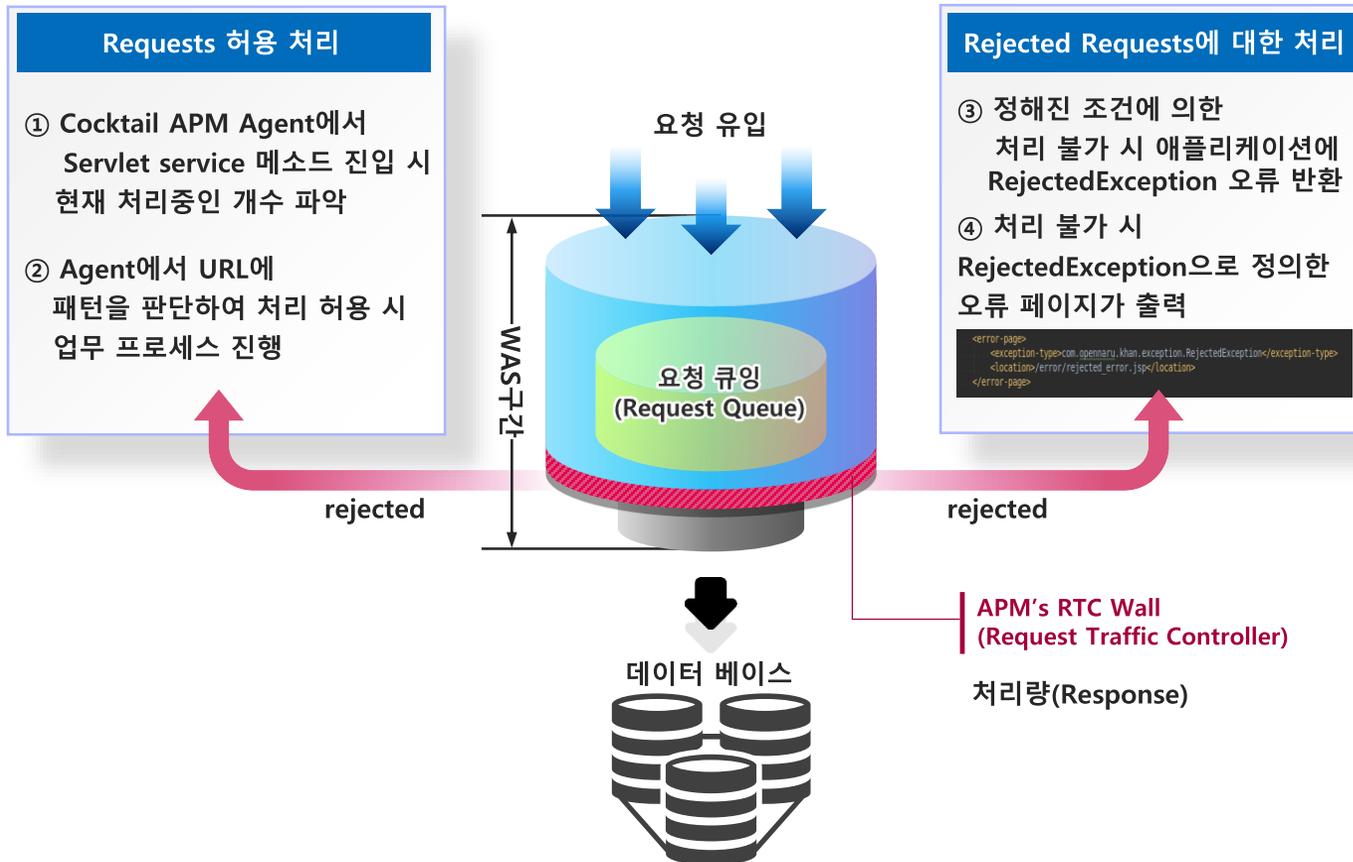
[E-mail]

[Slack 메신저]

애플리케이션 부하량 제어 기능

서비스 폭주시 업무 중요도에 따라 애플리케이션 유입 부하량 제어 기능 제공

- DBMS 등 Backend 자원을 보호하기 위한 유입량 제어 기능 제공
- 데이터베이스 등 주요 Backend 자원을 보호하기 위한 유입량 제어 기능

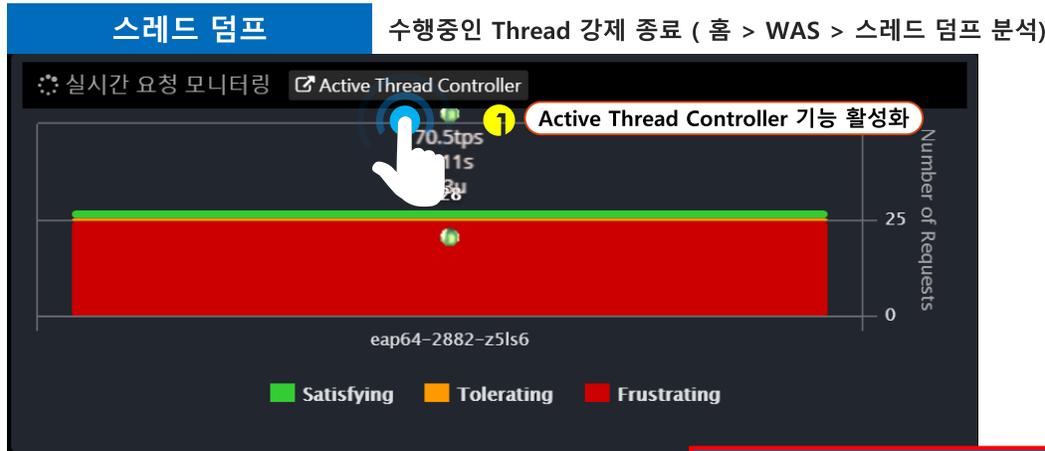


The top screenshot shows a browser window with the title '메시지 출력 후 Count Down' (Message output after Count Down). The page content includes: '523 Rejected Error', 'Requested URL is rejected by OPENMARU APM - RTC(Request Traffic Controller) Wall. Count Down : 09 Seconds', and 'About Exception' with details: Exception: com.opennaru.khan.exception.RejectedException: Request Rejected Exception, errorStatusCode: 500, errorType: class com.opennaru.khan.exception.RejectedException, errorMessage: JBWEB000248: Servlet execution threw an exception, errorUri: /test/slow.jsp. A blue arrow labeled 'Count Down 완료후' (After Count Down complete) points to the bottom screenshot. The bottom screenshot shows the same error page but with the 'Count Down' timer at '00 Seconds' and a 'Retry' button highlighted by a hand icon.

비정상 스레드 강제 종료 기능

비정상적으로 수행중인 애플리케이션 스레드를 강제 종료(KILL)하는 기능 제공

- 지연되는 스레드를 강제로 종료 할 수 있는 기능 제공



Active Thread Controller

Active Thread Controller

 10s 5s

#	Thread Name	State	URL	Duration(ms)
1	default task-39	BLOCKED	/uat/uia/actionLogin.do	1,365
2	default task-245	BLOCKED	/uat/uia/actionLogin.do	1,210
3	default task-173	BLOCKED	/uat/uia/actionMain.do	1,096
4	default task-74	BLOCKED	/uat/uia/actionMain.do	1,087
5	default task-197	BLOCKED	/uat/uia/actionMain.do	1,075
6	default task-176	BLOCKED	/uat/uia/actionMain.do	1,071
7	default task-117	BLOCKED	/uat/uia/actionLogin.do	1,055
8	default task-174	BLOCKED	/uat/uia/actionMain.do	1,039
9	default task-209	BLOCKED	/uat/uia/actionLogin.do	1,029
10	default task-191	BLOCKED	/uat/uia/actionLogin.do	1,011
11	default task-203	BLOCKED	/uat/uia/actionMain.do	967
12	default task-5	BLOCKED	/uat/uia/actionMain.do	966
13	default task-60	BLOCKED	/uat/uia/actionLogin.do	956

실시간으로 현재 사용중인 Thread STOP, Interrupt로 강제 종료 가능

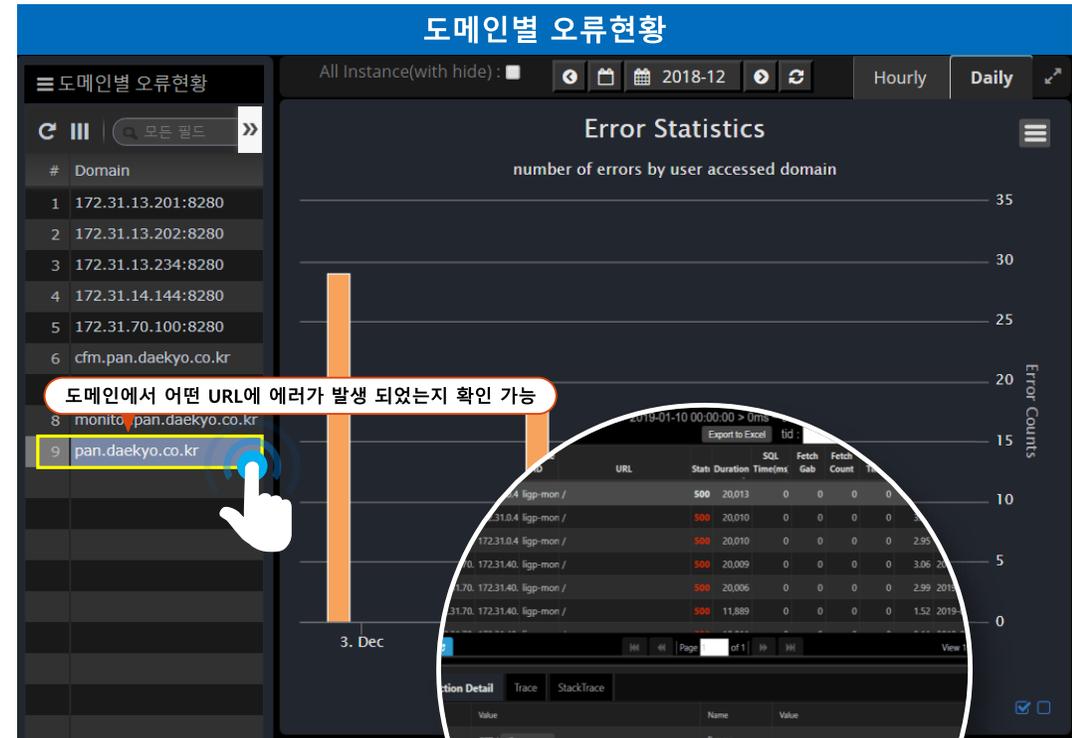
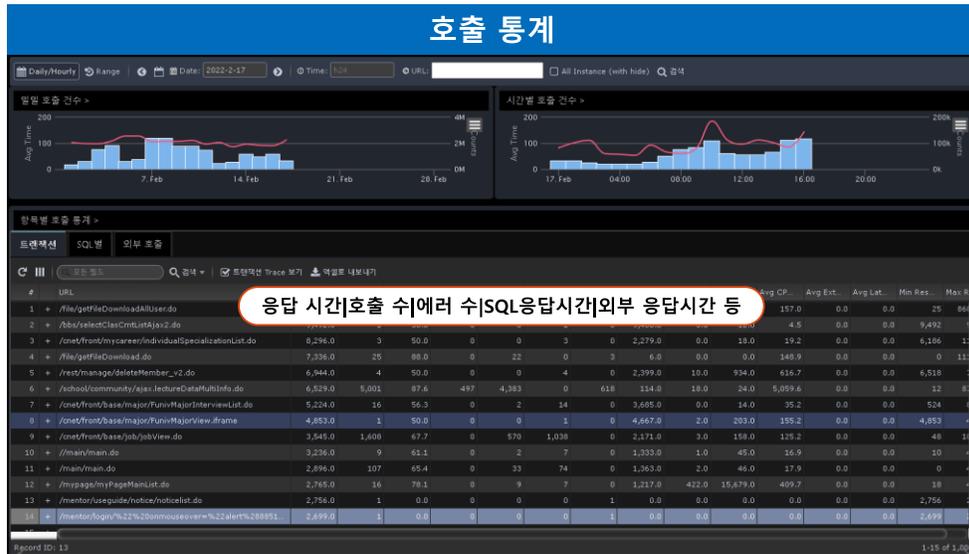
```

Active Thread Text > default task-39 #228 - /uat/uia/actionLogin.do
Current Thread:
[ 15][01:27:51.883] 0| 0| 0| 0| 0.0] + org.jboss.as.connector.s
[ 16][01:27:51.883] 0| 0| 0| 0| 0.0] + org.jboss.jca.adapters.j
jdbc.WrapperDataSource.getConnection()
[ 17][01:27:52.345] 0| 0| 0| 0| 0.0] + org.jboss.jca.adapters.j
dbc.WrapperConnection.setAutoCommit(false)
[ 18][01:27:52.345] 19| 0| 0| 19| 0.4] + org.jboss.jca.adapters.j
dbc.WrapperConnection.prepareStatement()
[ 19][01:27:52.345] 19| 0| 0| 19| 0.4] + org.jboss.jca.adapters.j
dbc.WrapperPreparedStatement.execute()
> DB : URL: jdbc:mysql://192.16
8.23.53:3306/gstest
> Query : SELECT emplr_id AS id ,
user_nm AS name , password AS password , ihidnum AS ihidNum , email_adres AS email , 'USR'
AS userSe , orgnzt_id AS orgnztId , esntl_id AS uniqId FROM CONTNEMPLYRINFO WHERE emplr_id
= ? AND password = ? AND emplr_sttus_code = 'P'
> Params: 'webmaster', '78gXjFDDWz
epOP4QIIFHRFOT/msFgkKHig6OU95..
> Mapped: SELECT emplr_id AS id ,
  
```

개별 트랜잭션 통계 분석 기능

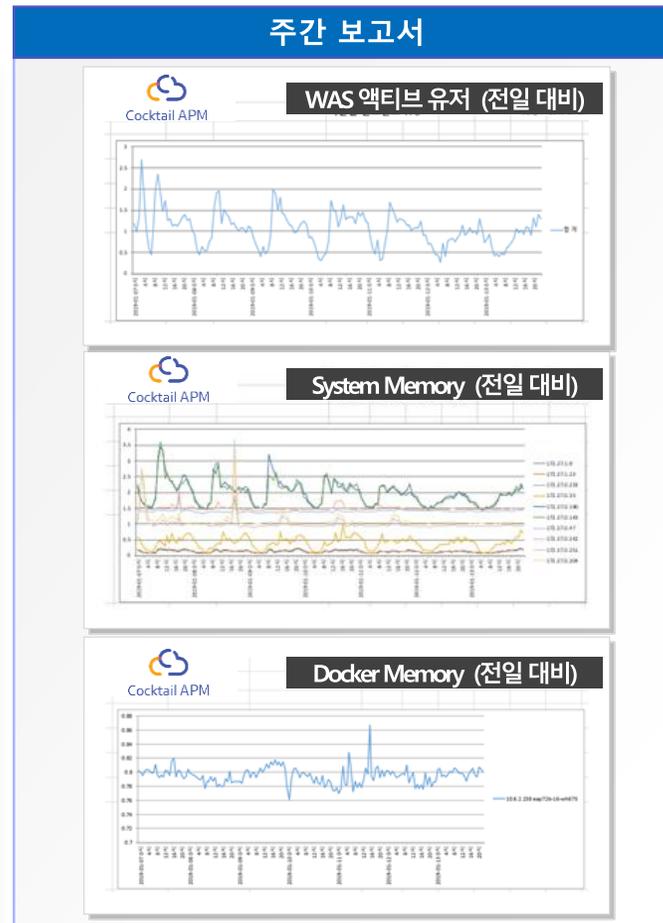
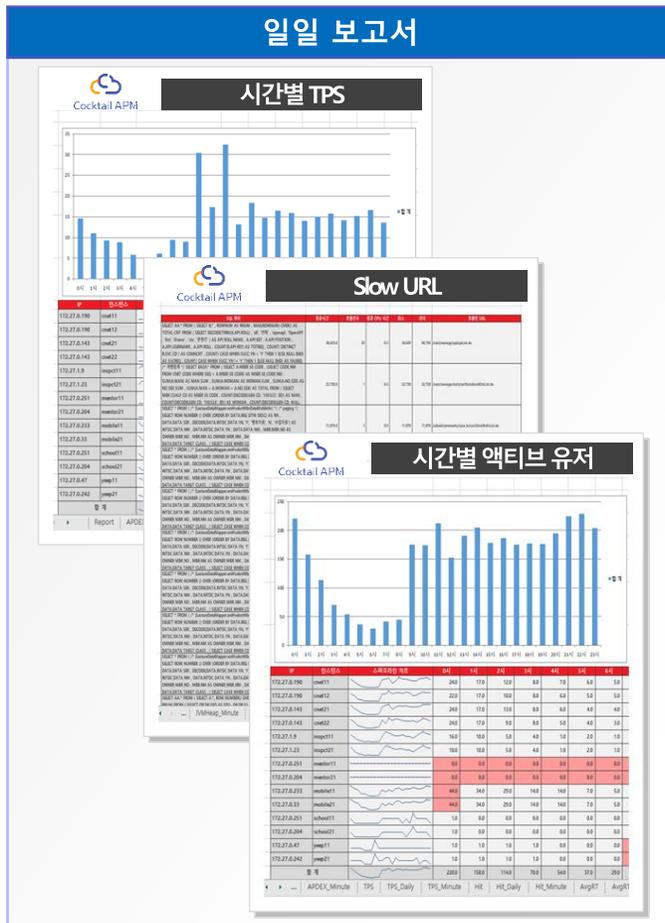
개별 트랜잭션에 대한 통계 분석 기능 제공

- 개별 요청에 대한 통계 정보를 제공하고 전체 응답시간 이외 데이터베이스 요청 시간, HTTP 외부 요청 시간 또한 확인
- 요청 도메인별 에러 통계를 확인하고 에러에 대한 상세 리스트 및 분석



일/주/월 단위로 자동으로 통계 보고서를 생성하는 기능 제공

- 일/주/월 단위 보고서를 자동 또는 수동으로 생성 할 수 있고 엑셀 형식으로 제공
- 보고서는 그룹별 권한이 있는 담당자가 다운로드 또는 E-mail 전송 제공



월간 보고서

[1]OCP 월간 보고서 2022. 01

날짜	요청수	에러수	평균응답 시간		액티브 사용자			TPS		
			최소	최대	최소	최대	평균	최소	최대	평균
2022-01-01 (토)	196,738	1	211	100	23	343	321	56	123	254
2022-01-02 (일)	460,081	0	233	111	12	343	329	48	234	246
2022-01-03 (월)	317,598	0	222	121	45	343	329	39	455	276
2022-01-04 (화)	225,292	0	245	123	32	334	345	64	343	297
2022-01-05 (수)	193,539	10	219	112	44	233	323	38	343	246
2022-01-06 (목)	135,351	0	234	99	22	111	323	34	363	272
2022-01-07 (금)	196,738	0	234	122	23	343	333	37	342	284
2022-01-08 (토)	460,081	0	211	100	12	343	329	64	257	275
2022-01-09 (일)	317,598	0	233	111	45	343	329	56	123	254
2022-01-10 (월)	225,292	8	222	121	32	334	329	48	234	246
2022-01-11 (화)	193,539	0	245	123	44	233	321	39	455	276
2022-01-12 (수)	135,351	0	219	112	22	111	329	64	343	297
2022-01-13 (목)	196,738	0	234	99	23	343	329	38	343	246
2022-01-14 (금)	460,081	0	234	122	12	343	345	34	363	272
2022-01-15 (토)	317,598	7	211	100	45	343	323	37	342	284
2022-01-16 (일)	225,292	0	233	111	32	334	323	64	257	275
2022-01-17 (월)	193,539	0	222	121	44	233	333	56	123	254
2022-01-18 (화)	460,081	1	245	123	32	111	329	48	234	246

구분	기능	비고
WAS	<ul style="list-style-type: none"> •APDEX •TPS •호출건수 •평균 응답시간 •액티브 유저 •에러율 •JVM 힙사용률 	전일/전주 비교 포함
System	<ul style="list-style-type: none"> •OS CPU 사용률 •OS Memory 사용률 	
Docker	<ul style="list-style-type: none"> •Docker CPU 사용률 •Docker Memory 사용률 	
애플리케이션 통계	<ul style="list-style-type: none"> •느린 URL •느린 Query 	

성능 통계 데이터 추출 기능

성능 통계 데이터를 자유롭게 추출/가공할 수 있는 기능 제공

- 모든 성능 데이터는 표준화된 포맷으로 CSV 파일로 추출하여 가공 할 수 있도록 제공
- 데이터베이스 SQL문을 통해 자유롭게 추출 기능 제공

Exporter 컴포넌트

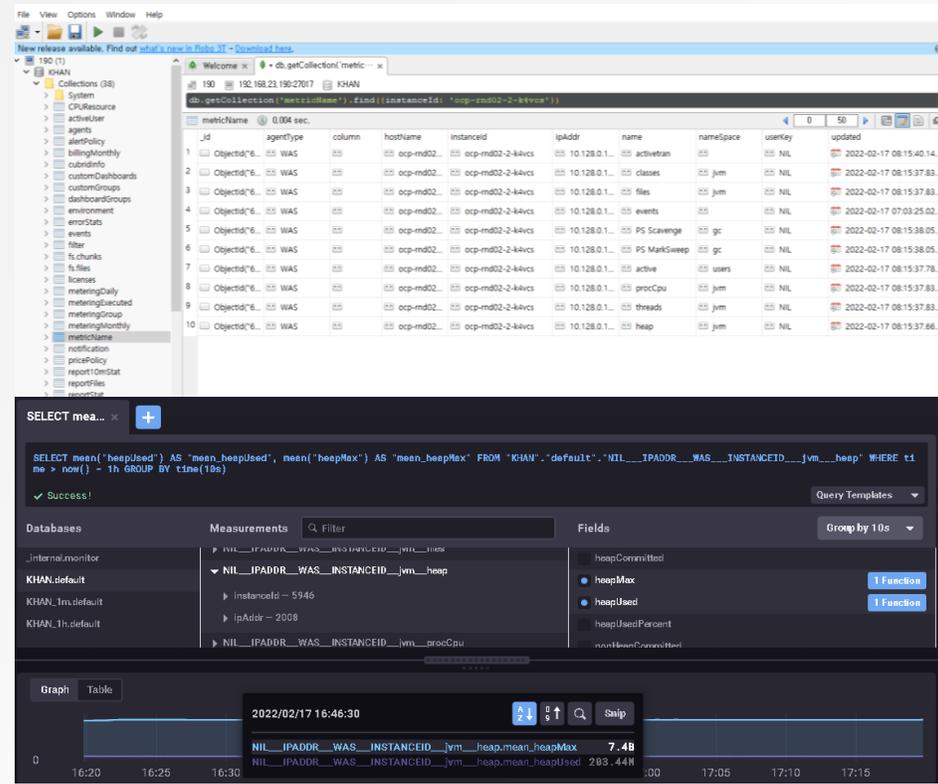
Cocktail APM 성능 테스트 데이터 중 TPS, Average Response Time, Active User, Transaction, CPU/MEM(OS, Docker) 항목에 대해 최소 5초 단위(최소 60초 권장)로 CSV 파일을 생성하는 기능으로 PC나 Linux 머신에서 명령 줄 옵션으로 CSV 파일을 생성하는 기능을 제공합니다.

F32					E29					
A	B	C	D	E	A	B	C	D	E	
1	Milliseconds	Time	TPS	ActiveUser	ResponseTime	1	Millisecon	Time	SystemCpu	SystemMem
2	1.32945E+12	2012-02-17 13:00	254	321	100	2	1.33E+12	2012-02-17 13:00	65	33
3	1.32945E+12	2012-02-17 13:01	246	329	111	3	1.33E+12	2012-02-17 13:01	65	34
4	1.32945E+12	2012-02-17 13:02	276	329	121	4	1.33E+12	2012-02-17 13:02	56	28
5	1.32945E+12	2012-02-17 13:03	297	345	123	5	1.33E+12	2012-02-17 13:03	64	34
6	1.32945E+12	2012-02-17 13:04	246	323	112	6	1.33E+12	2012-02-17 13:04	67	35
7	1.32945E+12	2012-02-17 13:05	272	323	99	7	1.33E+12	2012-02-17 13:05	67	34
8	1.32945E+12	2012-02-17 13:06	284	333	122	8	1.33E+12	2012-02-17 13:06	65	33
9	1.32945E+12	2012-02-17 13:07	275	329	100	9	1.33E+12	2012-02-17 13:07	57	34
10	1.32945E+12	2012-02-17 13:08	254	329	111	10	1.33E+12	2012-02-17 13:08	59	28
11	1.32945E+12	2012-02-17 13:09	246	329	121	11	1.33E+12	2012-02-17 13:09	57	34
12	1.32945E+12	2012-02-17 13:10	276	321	123	12	1.33E+12	2012-02-17 13:10	67	35
13	1.32945E+12	2012-02-17 13:11	297	329	112	13	1.33E+12	2012-02-17 13:11	53	34
14	1.32945E+12	2012-02-17 13:12	246	329	99	14	1.33E+12	2012-02-17 13:12	65	33
15	1.32945E+12	2012-02-17 13:13	272	345	122	15	1.33E+12	2012-02-17 13:13	65	34

A1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R		
1	ipAddress	clientip	instanceId	app_group	agentType	transactionId	html	startTime	duration	cpuTime	jdbcTime	fetchTime	fetchCount	externalTs	status	url	warningCt	RegexFilter	
2	172.31.13.0:0.0	0	172.31.13.0:0.0	llgp-app-1PROD	WAS	1754942f400001,175	2020-10-21 12:45	6280	3.83	0	0	0	0	0	200	org.apache.http.impl.client	0	FALSE	
3	172.31.13.218.51	127	172.31.13.218.51	llgp-mon-1PROD	WAS	17549428c00002,175	2020-10-21 12:44	515	3.1	513	0	15	0	0	200	/rest/home/class/Status	0	FALSE	
4	172.31.13.106.255.51	127	172.31.13.106.255.51	llgp-mon-1PROD	WAS	17549428c00003,175	2020-10-21 12:44	569	2.96	566	1	74	0	0	200	/rest/home/study/Status	0	FALSE	
5	172.31.14.218.147.13	127	172.31.14.218.147.13	llgp-mon-1PROD	WAS	17549429c00004,175	2020-10-21 12:45	518	3.23	511	1	29	0	0	200	/rest/home/class/Status	0	FALSE	
6	172.31.13.122.24.2	127	172.31.13.122.24.2	llgp-app-1PROD	WAS	17549429c00005,175	2020-10-21 12:45	1552	7.78	164	0	7	0	0	200	/app/api/03.do	0	FALSE	
7	172.31.13.106.255.51	127	172.31.13.106.255.51	llgp-mon-1PROD	WAS	17549429c00006,175	2020-10-21 12:45	569	3.02	567	0	73	0	0	200	/rest/home/study/Status	0	FALSE	
8	172.31.14.112.163.13	127	172.31.14.112.163.13	llgp-mon-1PROD	WAS	17549429c00007,175	2020-10-21 12:45	537	3.36	531	0	33	0	0	200	/rest/home/class/Status	0	FALSE	
9	172.31.14.106.253.25	127	172.31.14.106.253.25	llgp-mon-1PROD	WAS	17549429c00008,175	2020-10-21 12:45	602	3.99	596	1	65	0	0	200	/rest/home/class/Status	0	FALSE	
10	172.31.13.106.249.16	127	172.31.13.106.249.16	llgp-app-1PROD	WAS	17549429c00009,175	2020-10-21 12:45	1405	9.7	195	0	7	0	0	200	/app/api/03.do	0	FALSE	
11	172.31.13.115.95.0.7	127	172.31.13.115.95.0.7	llgp-mon-1PROD	WAS	1754942a000010,175	2020-10-21 12:45	633	3.84	629	1	117	0	0	200	/rest/home/class/Status	0	FALSE	
12	172.31.14.218.147.13	127	172.31.14.218.147.13	llgp-mon-1PROD	WAS	1754942a000011,175	2020-10-21 12:45	523	3.52	516	0	14	0	0	200	/rest/home/class/Status	0	FALSE	
13	172.31.45.49.172.214	admin@k1	172.31.45.49.172.214	admin@k1	PROD	WAS	1754942a100012,175	2020-10-21 12:45	2043	13.54	2030	2	25	0	0	200	/dktch/popup/evalInfoPril	0	FALSE
14	172.31.13.106.255.51	127	172.31.13.106.255.51	llgp-mon-1PROD	WAS	1754942a100013,175	2020-10-21 12:45	568	3.09	566	0	71	0	0	200	/rest/home/study/Status	0	FALSE	
15	172.31.14.182.208.44	127	172.31.14.182.208.44	llgp-mon-1PROD	WAS	1754942b000014,175	2020-10-21 12:45	572	2.6	569	0	14	0	0	200	/rest/home/study/Status	0	FALSE	
16	172.31.13.106.253.24	127	172.31.13.106.253.24	llgp-mon-1PROD	WAS	1754942b000015,175	2020-10-21 12:45	634	2.95	632	0	61	0	0	200	/rest/home/class/Status	0	FALSE	

Database SQL

데이터 베이스 Client Tool 을 활용하여 SQL 문을 통해 조회



The screenshot shows a database client interface with a table of metrics and a SQL query editor. The table has columns for metricName, column, hostName, instanceId, ipAddress, name, nameSpace, userKey, and updated. The SQL query editor shows a query to select mean values for heapUsed and heapMax from the KHAN database.

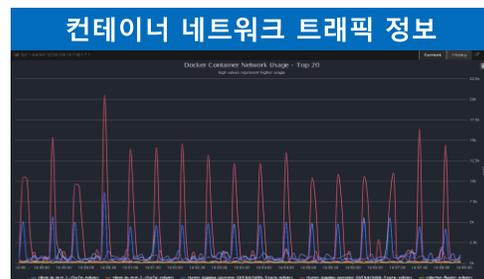
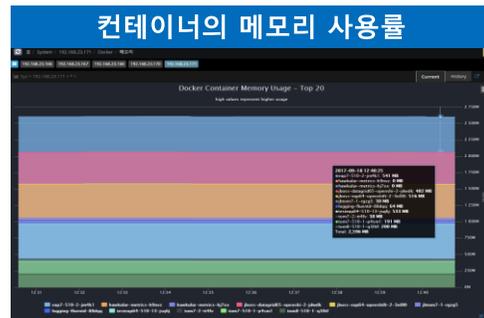
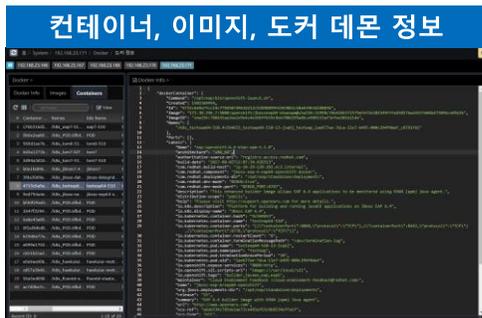
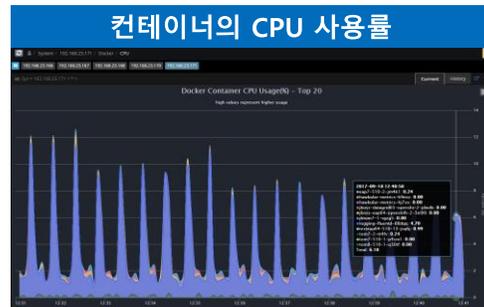
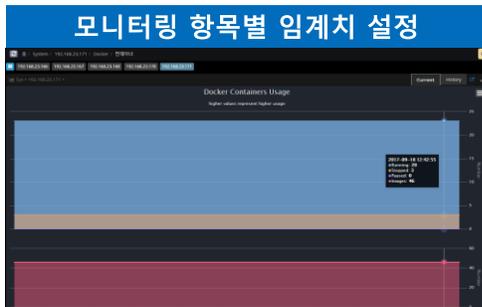
```
SELECT mean("heapUsed") AS "mean_heapUsed", mean("heapMax") AS "mean_heapMax" FROM "KHAN"."default"."NIL__IPADDR__WAS__INSTANCEID__jvm__heap" WHERE t1 me > now() - 1h GROUP BY time(10s)
```

The interface also shows a 'Fields' section with a list of fields including heapCorrupted, heapMax, heapUsed, and heapUsedPercent. A graph at the bottom shows the results of the query over time, with a value of 7.48 displayed.

컨테이너 기반 모니터링 기능

컨테이너(쿠버네티스) 기반 모니터링 기능 제공

- 컨테이너에서 작동하는 WAS의 모니터링 뿐만 아니라 컨테이너의 CPU, 메모리, 네트워크 트래픽/패킷 오류를 확인 할 수 있습니다.
- 동작하는 컨테이너의 모니터링 뿐만 아니라 컨테이너 이미지의 메타 상세 정보도 확인 할 수 있습니다.

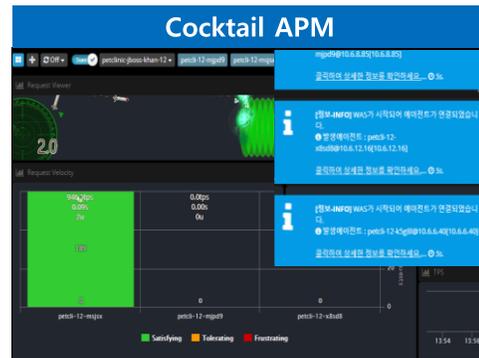
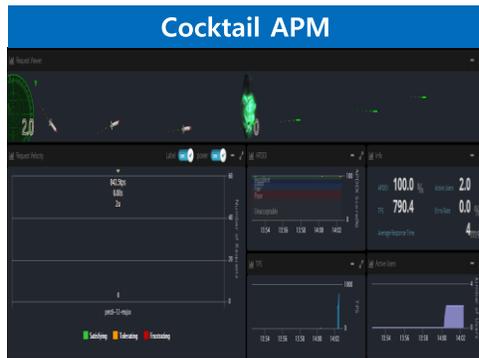
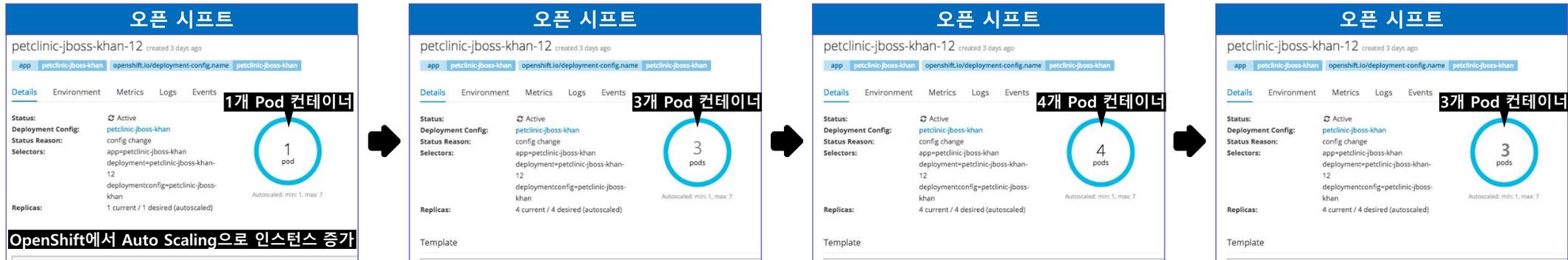


컨테이너(PaaS) 지원 APM과 기존 APM 기능 비교			
구 분	세부 항목	Cocktail APM(컨테이너 지원 APM)	기존 APM제품
기존 시스템	Java	• 모니터링 지원	모니터링 지원
	WAS	• 모니터링 지원	모니터링 지원
클라우드 시스템	PaaS 환경	• Docker Daemon 과 관련 이미지 정보 제공 • POD 에 대한 리소스 정보 제공 (CPU/Memory/Disk/Network)	모니터링 지원 불가
	오토스케일링	• 오토스케일링 관련 리소스 정보 제공 • WAS 상태 정보 제공	모니터링 일부 지원
	PaaS 에서 Java 정보	• POD 상에서 실행되는 Java 가상 머신 정보 제공 (Heap , Java 상태 정보 등)	모니터링 일부 지원
	PaaS 에서 WAS 정보	• POD 상에서 실행된 WAS 에 대한 정보	모니터링 일부 지원
	POD 상태 정보 제공	• 폐기된 POD 에 대한 검색 지원 • 과거 POD 에 대한 정보 제공	모니터링 일부 지원

컨테이너 환경 에이전트 관리 방안

컨테이너 환경에서 WAS 인스턴스 관리 기능 제공

- 오토 스케일 인/아웃 상황에 따라 에이전트가 자동으로 추가되고, 자동으로 숨겨집니다.
- 컨테이너 중지로 인해 삭제된 인스턴스의 모니터링 정보는 삭제되지 않고 숨겨져 있으므로 필요 시 다시 분석 가능합니다.



Auto Scaling으로 증가된 인스턴스 자동 추가 및 모니터링

Auto Scaling으로 제거된 인스턴스 자동 삭제

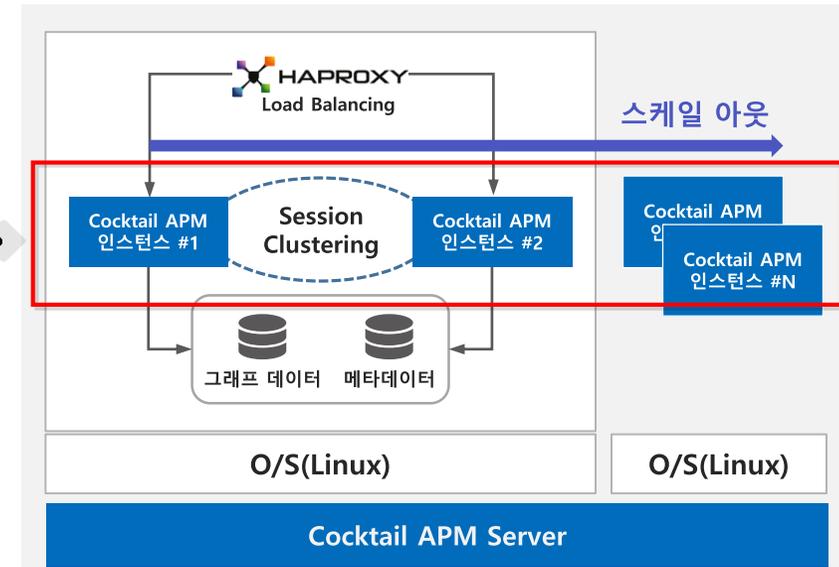
확장 가능한 수집 서버 아키텍처

확장 가능한 아키텍처 지원 기능 제공

- 대용량 에이전트를 지원하기 위해 Cocktail APM은 기본적으로 이중화로 구성되며, 필요한 경우 수집 인스턴스를 확장하여 부하를 분산할 수 있습니다.
- 이중화 된 Cocktail APM인스턴스는 HAProxy와 Cocktail APM CLUSTER 세션 클러스터링을 통해 무 중단 서비스를 지원합니다.



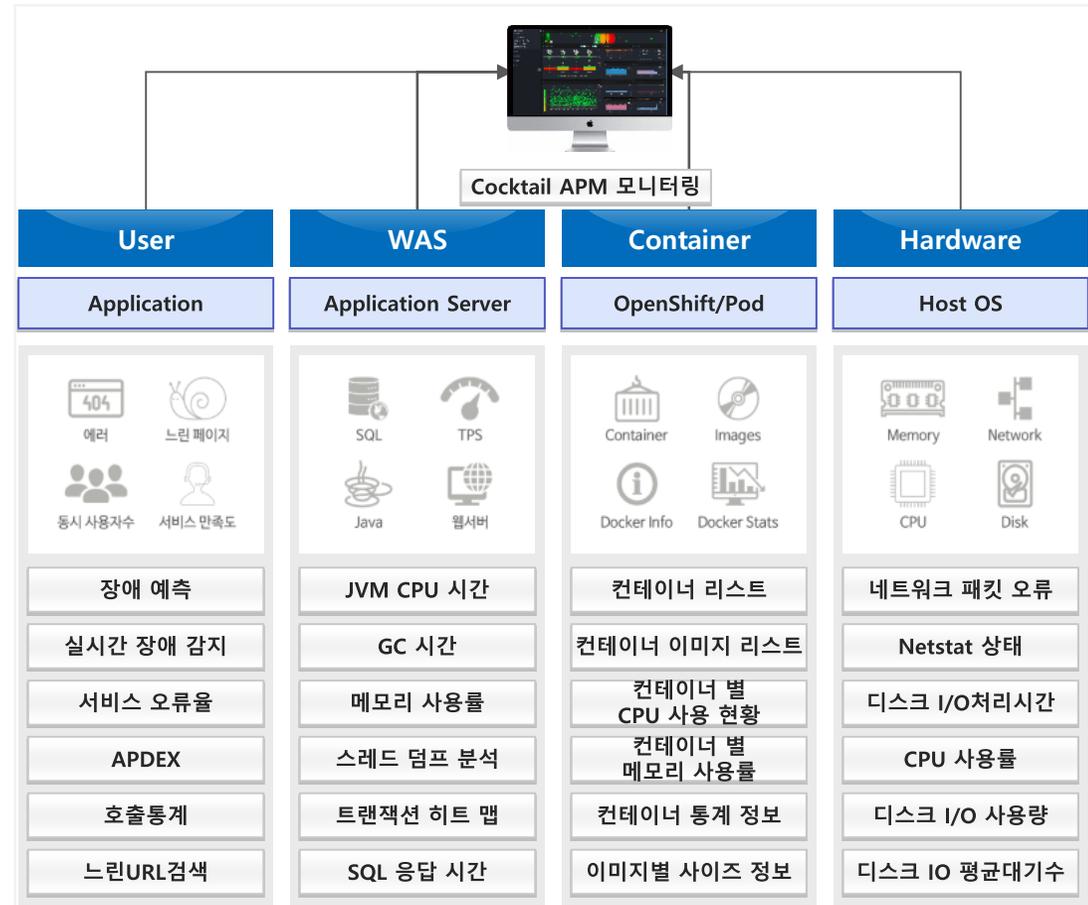
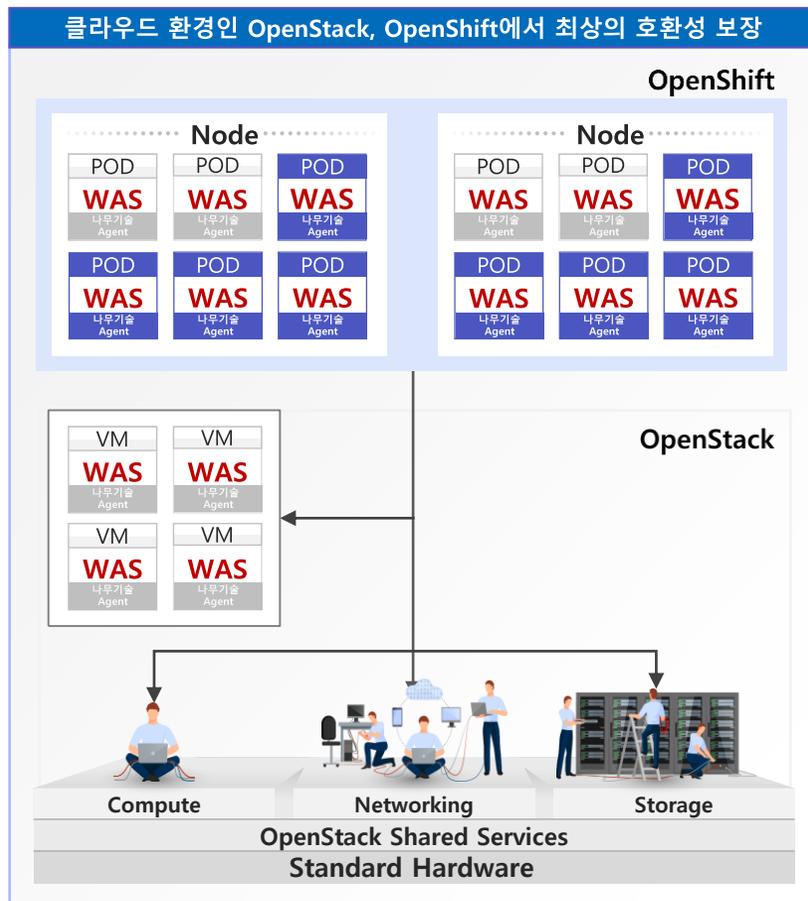
- ⊙ 세션 유지를 위해 Cocktail APM CLUSTER 세션 클러스터링 구성
- ⊙ 자체적인 이중화를 통해서 한쪽 서버가 중단되더라도 서비스뿐만 아니라 모니터링도 정상적으로 유지
- ⊙ 시스템 장애 발생 시 장애 복구가 용이
- ⊙ 패치 및 업그레이드 과정 시에도 365일 24시간 안정적인 무중단 서비스 작업을 위한 이중화 구조



클라우드 환경에 최적화된 APM

클라우드 운영 환경, 개발, 검증 환경에 최적화된 기능 제공

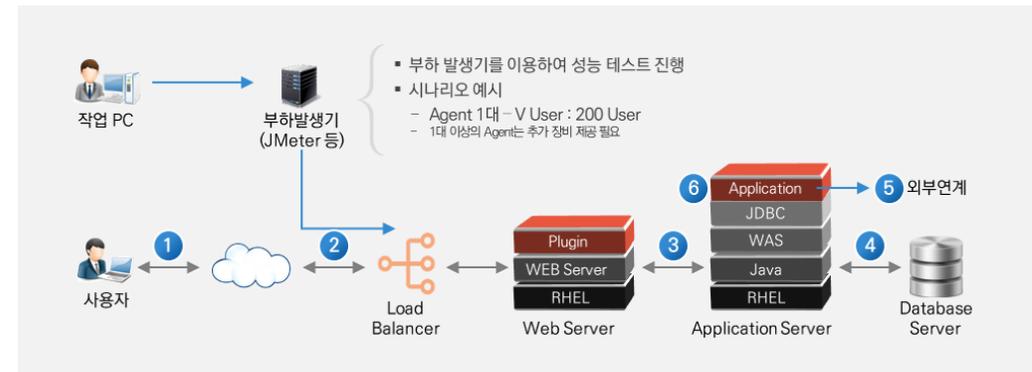
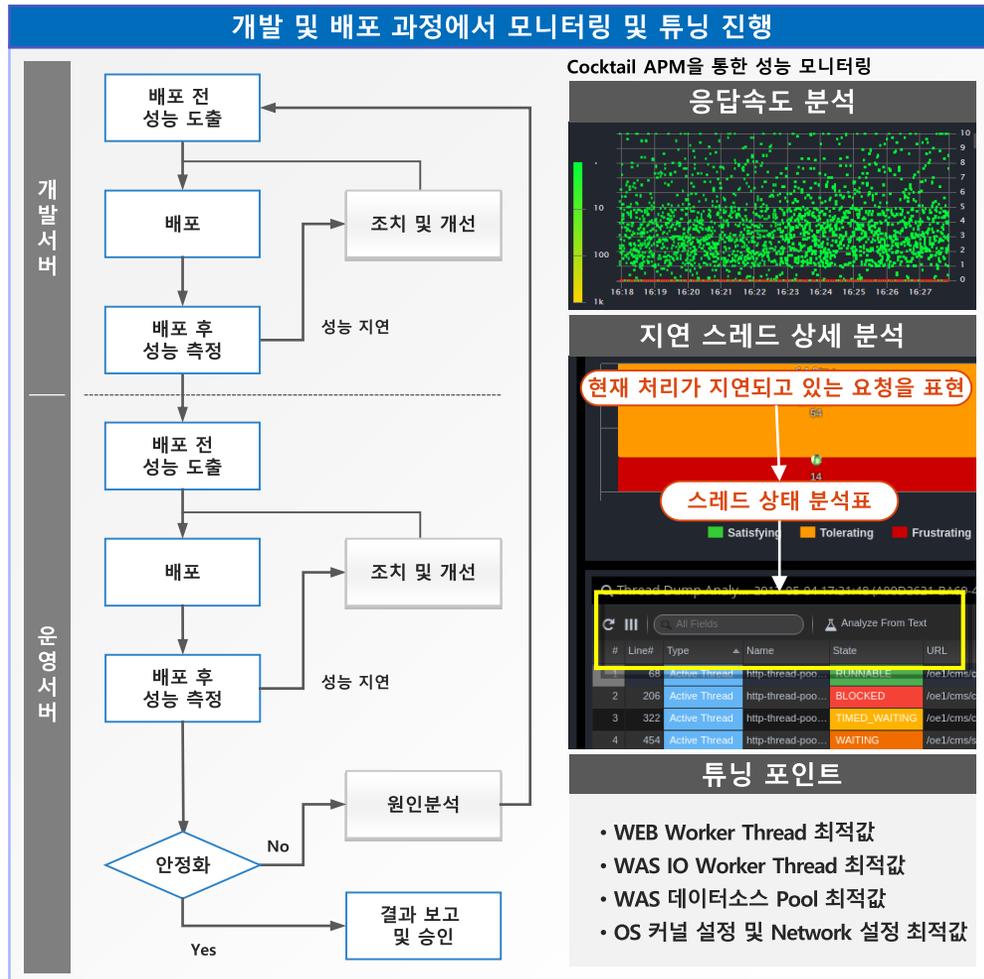
- 가상화(OpenStack 등), Kubernetes(OpenShift)에 최적화된 APM입니다. 다수의 대규모 컨테이너 환경에 적용되어 적합성을 이미 검증 받았습니다.
- Cocktail APM은 클라우드 환경(VM/Container)에 유리한 라이선스 정책을 적용하고 있습니다.



서비스 안정화 및 기술지원

APM은 서비스 개발 및 오픈이후 안정화까지 모니터링과 튜닝이 필요합니다.

- Cocktail APM을 활용하여 개발 시점부터 오픈 시점까지 지속적인 모니터링 및 튜닝을 시행하여 시스템의 안정적 오픈과 운영을 지원합니다.



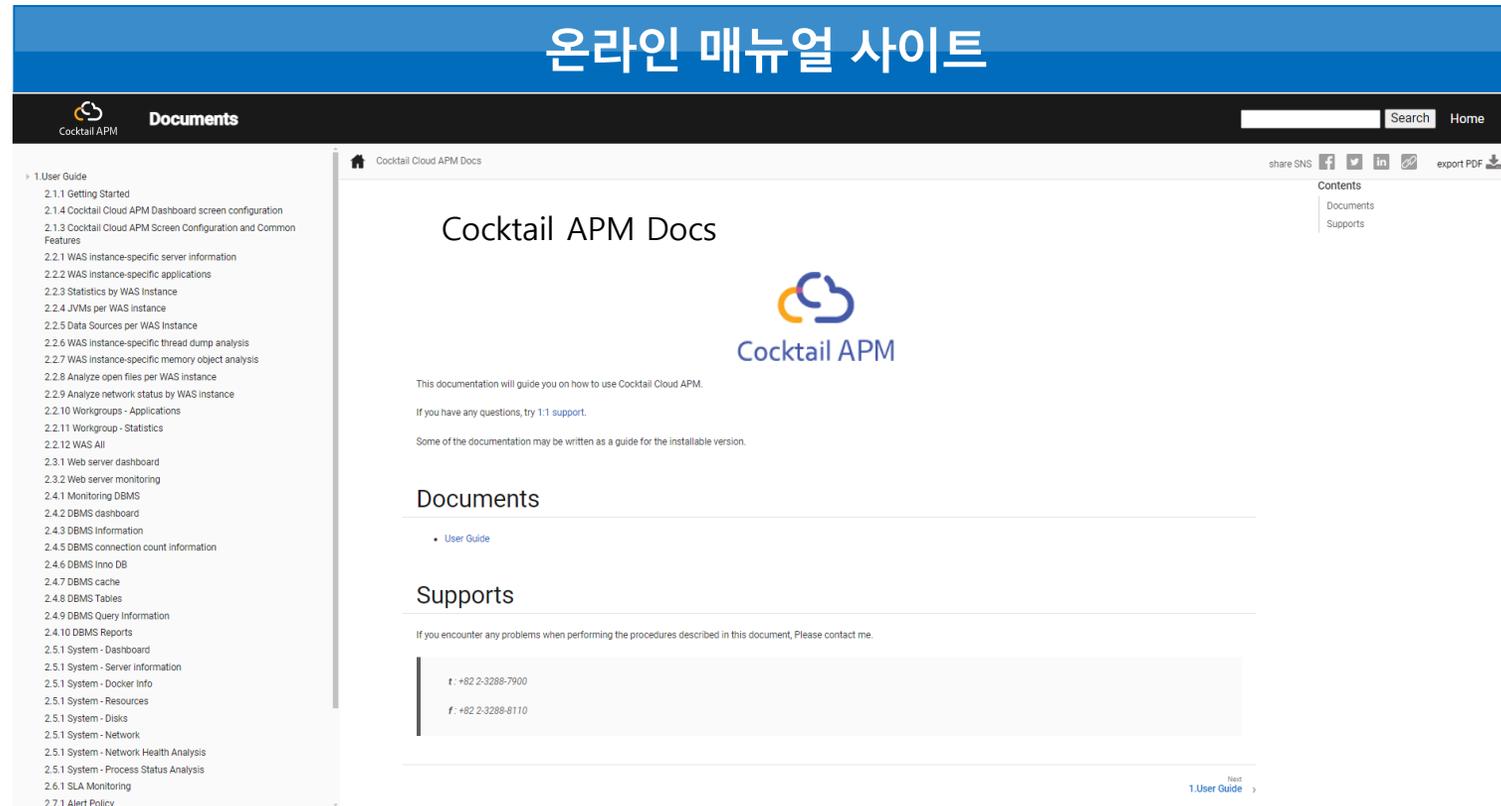
주요 구간별 확인 및 튜닝 포인트

1	사용자 - 방화벽 및 L4(LB)	<ul style="list-style-type: none"> 방화벽 대역폭 확인
2	방화벽 및 L4(LB) - WEB	<ul style="list-style-type: none"> 목적지 연결 설정 로드 밸런싱 옵션 (Client IP Hash)
3	WEB - WAS	<ul style="list-style-type: none"> MEM Worker Thread 튜닝 ModJk 연결 옵션 튜닝 ModJk Sticky 옵션
4	WAS - DBMS	<ul style="list-style-type: none"> IO Worker Thread 튜닝 DB Connection Pool 튜닝 DB Statement Cache 튜닝 JVM Heap Memory 튜닝
5	WEB/WAS - NAS	<ul style="list-style-type: none"> IOPS, 지연 시간 확인
6	WAS - 외부 인터페이스	<ul style="list-style-type: none"> HTTP API Connection Pool 튜닝
7	애플리케이션	<ul style="list-style-type: none"> 오류 확인 비즈니스 로직 성능 확인

APM 사용자 온라인 매뉴얼

Cocktail APM 사용자를 위한 온라인 매뉴얼 제공

- Cocktail APM의 사용자 가이드 등 문서를 온라인 매뉴얼로 제공합니다.
- 또한, “거침없이 배우는 JBoss” 책을 웹 사이트에서 제공하고 있습니다.



The screenshot shows the 'Online Manual Site' for Cocktail APM. The page has a blue header with the text '온라인 매뉴얼 사이트'. Below the header is a navigation bar with the Cocktail APM logo, the word 'Documents', a search bar, and a 'Home' link. The main content area is titled 'Cocktail APM Docs' and features the Cocktail APM logo. It includes a paragraph of introductory text, a 'Documents' section with a link to 'User Guide', and a 'Supports' section with contact information: 't: +82 2-3288-7900' and 'f: +82 2-3288-8110'. A sidebar on the left lists a detailed table of contents for the User Guide, including sections like 'Getting Started', 'Cocktail Cloud APM Dashboard screen configuration', and 'Monitoring DBMS'. The bottom right corner of the page has a 'Next' link pointing to '1.User Guide'.

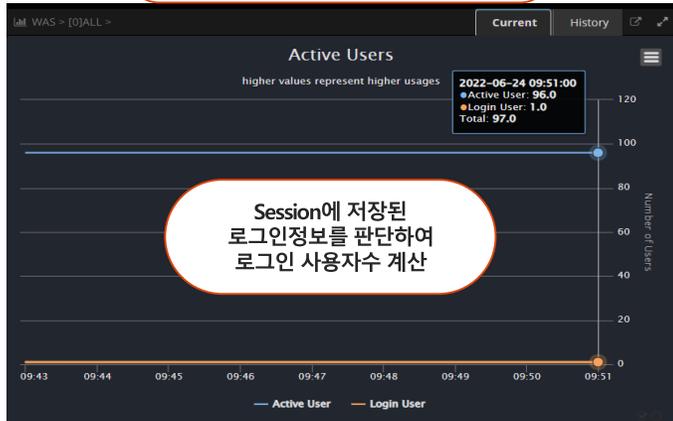
Google Analytics 방식의 사용자 통계 적용

구글 애널리틱스와 동일한 사용자 측정 알고리즘을 이용한 방문자 수 계산 방법을 제공합니다.

- 구글 애널리틱스에서 활용한 알고리즘을 활용한 액티브 사용자 수 기능 제공, 사용자가 여러 WAS 인스턴스에 동시 접속하더라도 동일 사용자로 계산함
- 일일(DAU)/주간(WAU)/월간(MAU) 사용자 분석 기능 제공
- 로그인 세션 아이디를 기반으로 사용자 수를 계산하여 제공



구글 애널리틱스에서 활용한 알고리즘 적용



마이크로 서비스(MSA) 호출 추적 기능 (1/2)

마이크로 서비스(MSA)와 같이 애플리케이션이 분산되어 있는 환경에서 호출된 API를 추적하는 기능 제공

- MSA 환경에서 업무간 호출되는 REST API(HTTP) 호출에 대해 추적하는 기능 제공
- 호출관계를 추적하여 MSA 애플리케이션의 상세 수행을 확인할 수 있습니다.

Child Transaction Detail

분석하고자 하는 MSA 시작점 Transaction

Agent	Server IP	Client IP	Instance ID	URL	Stat	Wa	Duration(m)	SQL Time(ms)	Fetch Gab	Fetch Count
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	9,784	0	0	0
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	8,794	0	0	0
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	9,760	0	0	0

Transaction Detail

Agent Type: WAS
Transaction ID: 181d1ae2cc3-58ac9eff69ccd9d141

Method Traces

```

[ 1][00:04:33.189] 9,783 100 0 0 1.0] org.apache.catalina.connector.CoyoteAdapter.service()
[ 2][00:04:33.189] 9,783 100 0 0 1.7] org.apache.jasper.servlet.JspServlet.service()
[ 3][00:04:33.190] 9,782 100 0 0 1.6] org.apache.jasper.servlet.JspServletWrapper.service()
[ 4][00:04:33.190] 9,782 100 0 0 1.6] org.apache.jasper.jsp._jspServlet()
[ 5][00:04:33.190] 4,983 53 4,983 0 0.5] org.apache.http.impl.client.CloseableHttpClient.execute()
[ 6][00:04:33.190] 4,983 53 4,983 0 0.5] org.apache.http.impl.client.CloseableHttpClient.execute()
    > Infos: GET http://api-status:8080/
  
```

Legend: 0 <= Time < 1,000ms 1,000ms <= Time < 3,000ms 3,000ms <= Time

분석하고자 하는 Child Transaction

Child Transaction Detail

Child Transaction Detail

분석하고자 하는 Child Transaction

Agent	Server IP	Client IP	Instance ID	URL	Stat	Wa	Duration(ms)	SQL Time(ms)	Fetch Gab	Fetch Count
WAS	10.131.1.91	api-st-4-m5	api-st-4-m5	/	200	0	4,979	0	0	0
WAS	10.128.2.104	api-co-3-4r5	api-co-3-4r5	/	200	0	4,797	0	0	0

Transaction Detail

Agent Type: WAS
Transaction ID: 181d1ae2cc3-58ac9eff69ccd9d141

Method Traces

```

[ 1][00:04:33.623] 4,979 100 0 0 1.3] org.apache.catalina.connector.CoyoteAdapter.service()
[ 2][00:04:33.623] 4,979 100 0 0 1.2] org.apache.jasper.servlet.JspServlet.service()
[ 3][00:04:33.623] 4,979 100 0 0 8.878] org.apache.jasper.servlet.JspServletWrapper.service()
[ 4][00:04:33.623] 4,979 100 0 0 1.2] org.apache.jasper.jsp._jspServlet()
[ 5][00:04:38.590] 12 0 1.1] org.apache.http.impl.client.CloseableHttpClient.execute()
[ 6][00:04:38.590] 12 0 1.1] org.apache.http.impl.client.CloseableHttpClient.execute()
    > Infos: GET http://api-status:8080/
  
```

Legend: 0 <= Time < 1,000ms 1,000ms <= Time < 3,000ms 3,000ms <= Time

분석하고자 하는 Child Transaction

Child Transaction Detail

Child Transaction Detail

분석하고자 하는 MSA 끝단의 Transaction

Agent	Server IP	Client IP	Instance ID	URL	Stat	Wa	Duration(ms)	SQL Time(ms)	Fetch Gab	Fetch Count	Ext. Time	CPU(ms)	Start Time	
WAS	10.131.1.65	egov-5-linr	egov-5-linr	/	200	0	10	3	0	0	0	0	2.82	2022-07-06 13:04:38

Transaction Detail

Agent Type: WAS
Transaction ID: 9-b8ecc846b6692b9e513

Method Traces

```

[ 25][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.setAutoCommit(false)
[ 26][00:04:38.000] 0 0 0 0 0.0] org.apache.commons.dbcp.DelegatingConnection.prepareStatement()
[ 27][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 28][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.setAutoCommit(true)
[ 29][00:04:38.000] 11 100 0 11 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 30][00:04:38.000] 11 100 0 11 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 31][00:04:38.000] 11 100 0 11 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 32][00:04:38.000] 11 100 0 11 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 33][00:04:38.000] 21 200 0 21 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 34][00:04:38.000] 21 200 0 21 0.1] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 35][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 36][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 37][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 38][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 39][00:04:38.000] 0 0 0 0 0.0] org.postgresql.jdbc.PgConnection.prepareStatement()
[ 40][00:04:38.000] 0 0 0 0 0.0] org.apache.jasper.servlet.JspServlet.service()
[ 41][00:04:38.000] 0 0 0 0 0.0] org.apache.jasper.servlet.JspServletWrapper.service()
[ 42][00:04:38.000] 0 0 0 0 0.0] org.apache.jsp.WEB_000000_jsp_egovframework.com.com.fgov01Main_jsp._jspServlet()
  
```

URL: jdbc:postgresql://postgre01-s432/egovdb
INSERT INTO CONTNMBLOG (REQQT_ID, URL, RQESTER_IP, OCCRRNC_DE) VALUES (?, ?, ?, NULL);
INSERT INTO CONTNMBLOG (REQQT_ID, URL, RQESTER_IP, RQESTER_IP, OCCRRNC_DE) VALUES ('M...

끝단 Transaction의 쿼리 등 Detail

마이크로 서비스(MSA) 호출 추적 기능 (2/2)

마이크로 서비스(MSA)와 같이 애플리케이션이 분산되어 있는 환경에서 호출된 API를 추적하는 기능 제공

- MSA 환경에서 REST API 호출 관계를 트리 형태로 보여주는 기능 제공
- REST API 호출관계를 트리형태로 표현합니다.

Relationship

분석하고자 하는 MSA 시작점 Transaction

Agent	Server IP	Client IP	Instance ID	URL	Stat	Wa	Duration(m)	SQL Time(ms)	Fetch Gab	Fetch Count	Ext. Time	CPU(ms)	Start Time
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	8,768	0	0	0	8,767	1.76	2022-07-08 10:41:3
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	8,458	0	0	0	8,457	1.71	2022-07-08 10:41:3
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	8,919	0	0	0	8,916	1.57	2022-07-08 10:41:4
WAS	10.128.2.102	192.168.60.11	api-ga-4-8g	/gateway.jsp	200	0	8,430	0	0	0	8,430	1.84	2022-07-08 10:41:4

Transaction Detail

Trace StackTrace

Name Value Name Value

Request URL GET /gateway.jsp 200 View URL Params

Request Host api-gateway-egov.apps.buahn.ocp.com View Page Sessions

Client IP 192.168.60.100,10.131.0.1 User Location User Agent Apache-HttpClient/4.5.12 (Java/1.8.0_302)

Start Time 2022-07-07 21:41:37.480 End Time 2022-07-07 21:41:46.248

Duration (ms) 8,768 CPU Time (ms) 1.76

JDBC Time (ms) 0 Latency (ms)

Thread Name http-nio-8080-exec-22

IP 10.128.2.102 Instance ID api-ga-4-8g4

Agent Type WAS Transaction ID Relationship Q 31d67809e5-3f5c12bccc9f5851a4c

Method Traces

Legend: 0 <= Time < 1,000ms 1,000ms <= Time < 3,000ms 3,000ms <= Time

트랜잭션간 맺어진 Relationship

Relationship

Relationship을 맺은 모든 MSA 트랜잭션이 한번에 출력, Tree 구조로 호출 관계 표시

Agent	IP address	Instance ID	URL	Status	Wa	Duration(ms)	SQL Time(ms)	Fetch Gab	Fetch Count	Ext.
WAS	10.128.2.102	api-ga-4-8g	/gateway.jsp	200	0	8,281	0	0	0	
WAS	10.131.1.91	api-st-4-m5	/	200	0	3,948	0	0	0	
WAS	10.131.1.65	egov-5-lmsr	/	200	0	18	5	0	0	
WAS	10.128.2.102	api-co-3-4r5	/	200	0	4,314	0	0	0	
WAS	10.131.1.65	egov-5-lmsr	/	200	0	12	3	0	0	

Transaction Detail Trace StackTrace

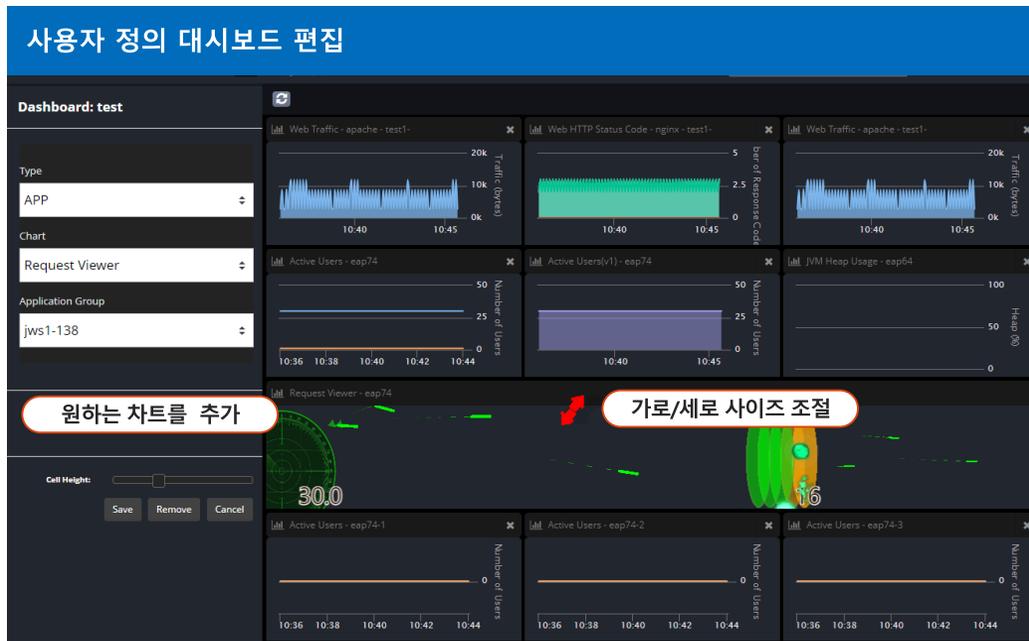
```

[ 27][23:48:53.680] 0| 0| 0| 0| 0.0] + org.postgresql.jdbc.PgConnection.prepareStatement()
[ 28][23:48:53.680] 0| 0| 0| 0| 0.0] + org.postgresql.jdbc.PgConnection.prepareStatement()
[ 29][23:48:53.680] 0| 0| 0| 0| 0.0] + org.postgresql.jdbc.PgConnection.prepareStatement()
[ 30][23:48:53.680] 1| 6| 0| 1| 0.3] + org.apache.commons.dbcp.DelegatingPreparedStatement.execute()
[ 31][23:48:53.680] 1| 6| 0| 1| 0.2] + org.apache.commons.dbcp.DelegatingPreparedStatement.execute()
[ 32][23:48:53.680] 1| 6| 1| 0| 0.1] + org.postgresql.jdbc.PgPreparedStatement.execute()
> DB :
> Query :
INSERT INTO COMTNMBLOG (REQUEST_ID, URL, REQUESTER_ID,
?, ?, ?, NOW())
> Params:
'MEBLOG_0000000376938', '/index.do', '', '127.0.0.6'
> Headers:
INSERT INTO COMTNMBLOG (REQUEST_ID, URL, REQUESTER_ID,
    
```

사용자 정의 대시보드 기능 지원

사용자 목적에 맞게 대시보드를 구성할 수 있는 기능 제공

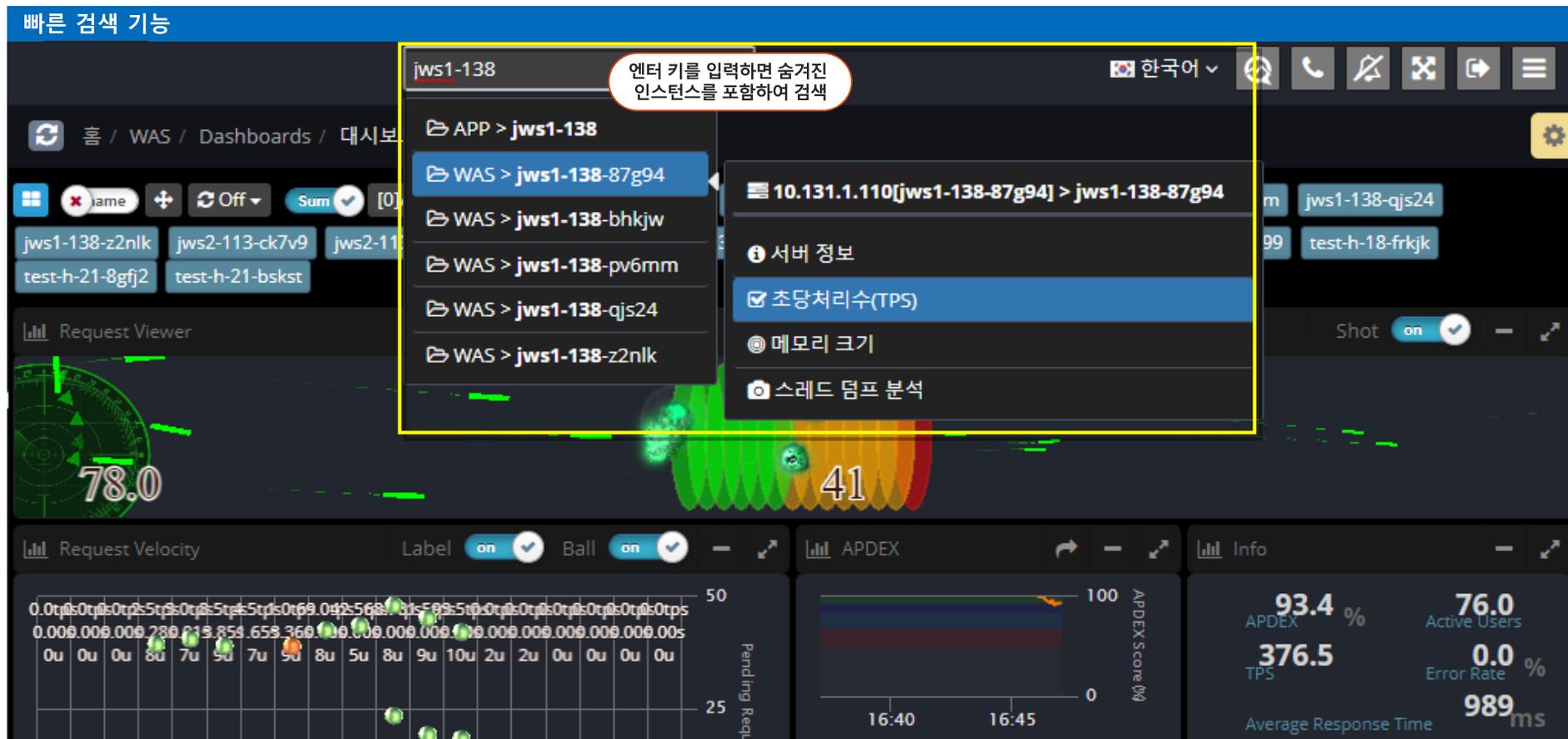
- N개의 애플리케이션 그룹 및 개별 인스턴스 차트를 별도로 구분하여 사용자가 원하는 그래프들을 대시보드로 정의하여 모니터링 할 수 있는 기능 제공
- 대시보드 모니터 환경에 맞게 세부적으로 차트 사이즈를 조절 할 수 있는 기능 제공



빠른 인스턴스 검색 기능 제공

모니터링 대상 인스턴스 이름 및 IP로 빠른 검색 기능 제공

- 인스턴스 이름 검색을 통해 빠르게 메뉴를 이동 할 수 있는 기능을 제공
- 컨테이너 환경에서 삭제되어 숨겨진 인스턴스를 검색하고 곧바로 활성화하여 분석 할 수 있는 기능 제공



The screenshot displays the '빠른 검색 기능' (Quick Search Function) search dropdown menu. The search input field contains 'jws1-138'. A tooltip indicates: '엔터 키를 입력하면 숨겨진 인스턴스를 포함하여 검색' (Search including hidden instances upon pressing the Enter key). The dropdown menu lists several instances:

- APP > jws1-138
- WAS > jws1-138-87g94
- WAS > jws1-138-bhkjw
- WAS > jws1-138-pv6mm
- WAS > jws1-138-qjs24
- WAS > jws1-138-z2nlk

A secondary dropdown menu is open for the selected instance '10.131.1.110[jws1-138-87g94] > jws1-138-87g94', showing the following options:

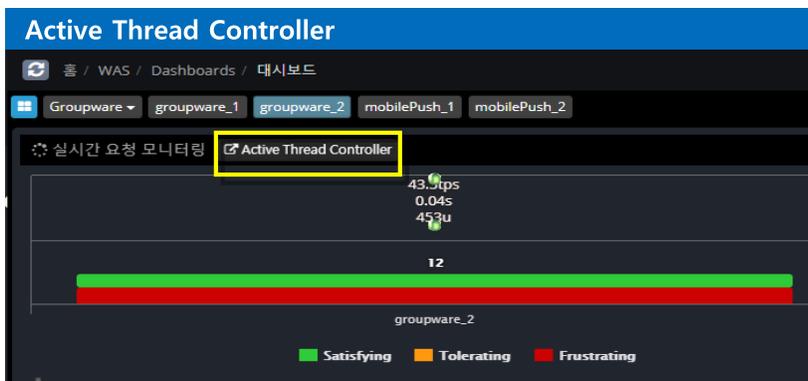
- 서버 정보 (Server Info)
- 초당처리수(TPS) (Requests per second)
- 메모리 크기 (Memory Size)
- 스레드 덤프 분석 (Thread Dump Analysis)

The background interface shows various monitoring charts and metrics, including 'Request Velocity' (78.0), 'APDEX' (93.4%), 'TPS' (376.5), and 'Average Response Time' (989ms).

실행 중인 트랜잭션에 대한 모니터링 및 제어 기능

현재 실행 중인 트랜잭션의 호출 정보, 쿼리 정보를 모니터링하고 스레드를 중지하는 기능 제공

- 스레드 덤프를 생성하지 않고 실시간으로 트랜잭션을 모니터링 하는 기능 제공
- 실행 중인 스레드 리스트, Current Stack(스택 트레이스), Transaction Trace(SQL, HTTP 등) 트랜잭션 기본 정보를 제공



현재 실행 중인 스레드 목록

#	Thread Name	State	URL	Duration(ms)
1	ajp-bio-8011-exec-3507	RUNNABLE	/ekp/service/eml/emlFileDownload	1,834,275,665
2	ajp-bio-8011-exec-4188	RUNNABLE	/ekp/inc/eml/selectMailView	1,834,141,980
3	ajp-bio-8011-exec-4331	RUNNABLE	/ekp/inc/eml/insertSendMail	1,834,290,686
4	ajp-bio-8011-exec-4343	RUNNABLE	/ekp/inc/eml/selectMailWrite	1,834,281,252
5	ajp-bio-8011-exec-4303	RUNNABLE	/ekp/inc/eml/deleteMailForever	1,834,140,747
6	ajp-bio-8011-exec-4336	RUNNABLE	/ekp/inc/eml/selectMailView	1,834,144,007
7	ajp-bio-8011-exec-8563	RUNNABLE	/ekp/inc/eml/selectMailView	1,758,681

실행 중인 스레드에 대한 중지, 인터럽트 기능

현재 실행 중인 스택 정보

Current Stack	Transaction Trace	Transaction Info
1 [ajp-bio-8011-exec-4188] #61856 daemon prio=5 tid=0x0000F1A0 RUNNABLE		
2 java.lang.Thread.State: RUNNABLE (RUNNABLE)		
3 java.net.SocketInputStream.socketRead0(Native Method)		
4 java.net.SocketInputStream.read(SocketInputStream.java:152)		
5 java.net.SocketInputStream.read(SocketInputStream.java:122)		
6 com.sun.mail.util.TraceInputStream.read(TraceInputStream.java:124)		
7 java.io.BufferedInputStream.fill(BufferedInputStream.java:235)		
8 java.io.BufferedInputStream.read(BufferedInputStream.java:254)		
9 com.sun.mail.iap.ResponseInputStream.readResponse(ResponseInputStream.java:95)		
10 com.sun.mail.iap.Response.<init>(Response.java:95)		
11 com.sun.mail.imap.protocol.IMAPResponse.<init>(IMAPResponse.java:60)		
12 com.sun.mail.imap.protocol.IMAPProtocol.readResponse(IMAPProtocol.java:286)		
13 com.sun.mail.imap.Protocol.<init>(Protocol.java:123)		
14 com.sun.mail.imap.protocol.IMAPProtocol.<init>(IMAPProtocol.java:115)		
15 com.sun.mail.imap.IMAPStore.newIMAPProtocol(IMAPStore.java:685)		
16 com.sun.mail.imap.IMAPStore.protocolConnect(IMAPStore.java:636)		
17 javax.mail.Service.connect(Service.java:295)		
18 javax.mail.Service.connect(Service.java:176)		
19 com.naon.biz.gw.email.connector.EmailMapgetStore(EmailMap.java:191)		
20 com.naon.biz.gw.email.connector.EmailMap.getMailMessage(EmailMap.java:1224)		
21 com.naon.biz.gw.email.connector.EmailMap.getMailMessageDisplayThumbnail(EmailMap.java:1224)		
22 com.naon.biz.gw.email.service.EmailService.selectMailView(EmailService.java:16)		
23 com.naon.biz.gw.email.service.EmailService\$\$FastClassByCGLIB\$\$a7680f1e.invoke(<generated>)		
24 org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:204)		
25 org.springframework.aop.framework.cgliblib.CglibAopProxy\$DynamicAdvisedInterceptor.intercept(CglibAopProxy\$DynamicAdvisedInterceptor.java:57)		
26 com.naon.biz.gw.email.service.EmailService\$\$EnhancerByCGLIB\$\$ba8d7eac1.selectMailView(EmailService\$\$EnhancerByCGLIB\$\$ba8d7eac1.java:16)		
27 com.naon.biz.gw.email.controller.EmailController.selectMailView(EmailController.java:16)		
28 sun.reflect.GeneratedMethodAccessor1473.invoke(Unknown Source)		

현재 실행 중인 트랜잭션 정보

Current Stack	Transaction Trace	Transaction Info
serverlet.DispatcherServlet.doService()		
[7][09:57:28.103] 0 0 0 5 0.0 + @RequestMapping com.naon.biz.gw.email.controller.EmailController.selectMailWrite()		
[8][09:57:28.103] 0 0 0 0 0.0 + org.apache.tomcat.dbcp.dbcp.BasicDataSource.getConnection()		
[9][09:57:28.103] 0 0 0 0 0.0 + org.apache.tomcat.dbcp.dbcp.PoolingDataSource.getConnection()		
[10][09:57:28.103] 0 0 0 0 0.1 + org.apache.tomcat.dbcp.dbcp.DelegatingConnection.prepareStatement()		
[11][09:57:28.103] 0 0 0 0 0.1 + oracle.jdbc.driver.T4CDSimpleStatement.allocatePreparedStatement()		
[12][09:57:28.103] 5 0 0 5 0.2 + org.apache.tomcat.dbcp.dbcp.DelegatingPreparedStatement.execute()		
[13][09:57:28.103] 5 0 0 5 0.2 + org.apache.tomcat.dbcp.dbcp.DelegatingPreparedStatement.execute()		
[14][09:57:28.103] 5 0 5 0 0.1 + oracle.jdbc.driver.OraclePreparedStatement.execute()		
> DB : URL: jdbc:oracle:thin:@192.168.2.196:1521:mgdb, User: EKPMOON_MEDIAWILL		
> Query : SELECT BOX.MAILBOX_ID, BOX.EMAIL, BOX.PHYSICAL_NAME, BOX.PHYSICAL_PATH, BOX.UID_VALIDITY, BOX.INNER_MAIL_USE_SIZE, BOX.EXTRL_MAIL_USE_SIZE, BOX.INNER_MAIL_CNT, BOX.EXTRL_MAIL_CNT, BOX.INNER_UNREAD_MAIL_CNT, BOX.EXTRL_UNREAD_MAIL_CNT, BOX.MAILBOX_KIND, BOX.MAILBOX_STATUS, BOX.MAILBOX_OWNER, BOX.BOX_ID, BOX.ID, BOX.PATH, BOX.PATH_ID, BOX.READ_CNT, BOX.READ_UID, BOX.READ...		

트랜잭션 URL 호출 정보

Current Stack	Transaction Trace	Transaction Info	
Name	Value	Name	Value
Request URL	POST /ekp/inc/eml/selectMailWrite	Params	
Request Host	mg.mediawill.com	Status Code	200
Client IP	192.168.13.121	User Agent	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/7.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; .NET4.0E)
Start Time	2018-09-11 09:57:28.096	End Time	2018-09-11 09:57:28.096
Duration (ms)	1,834,281,252	CPU Time (ms)	0.00
JDBC Time (ms)	5	Latency (ms)	0

LDAP 연동 로그인, Java 11, Spring Boot 지원

현재 실행 중인 트랜잭션의 호출정보, 쿼리 정보를 모니터링하고 스레드를 중지하는 기능 제공

- 최신 Java 11 버전을 지원
- LDAP 및 Active Directory를 이용한 APM 로그인 연동을 지원
- Spring Boot Embedded Datasource HikariCP 모니터링 지원

지원 Java 버전 업데이트



Spring Boot 2.x 버전 지원



Spring Boot의 기본 Datasource인 HikariCP 모니터링

LDAP / Active Directory와 APM로그인 연동 기능

LDAP

LDAP 사용여부: TRUE

APM 유저 LDAP 인증 사용여부

LDAP 서버 URL:

Enter LDAP server's URL
ex) ldap://10.10.10.10:389 or ldaps://10.10.10.10:636

Connect 타임아웃:

LDAP 서버에 대한 Connect 타임아웃(ms)

Read 타임아웃:

LDAP 서버에 대한 Read 타임아웃(ms)

SSL 사용여부: FALSE

LDAP 연결에 SSL 사용여부

유저 DN 접두사(Prefix):

유저 DN(Distinguished Names) 접두사(Prefix). ex) uid=

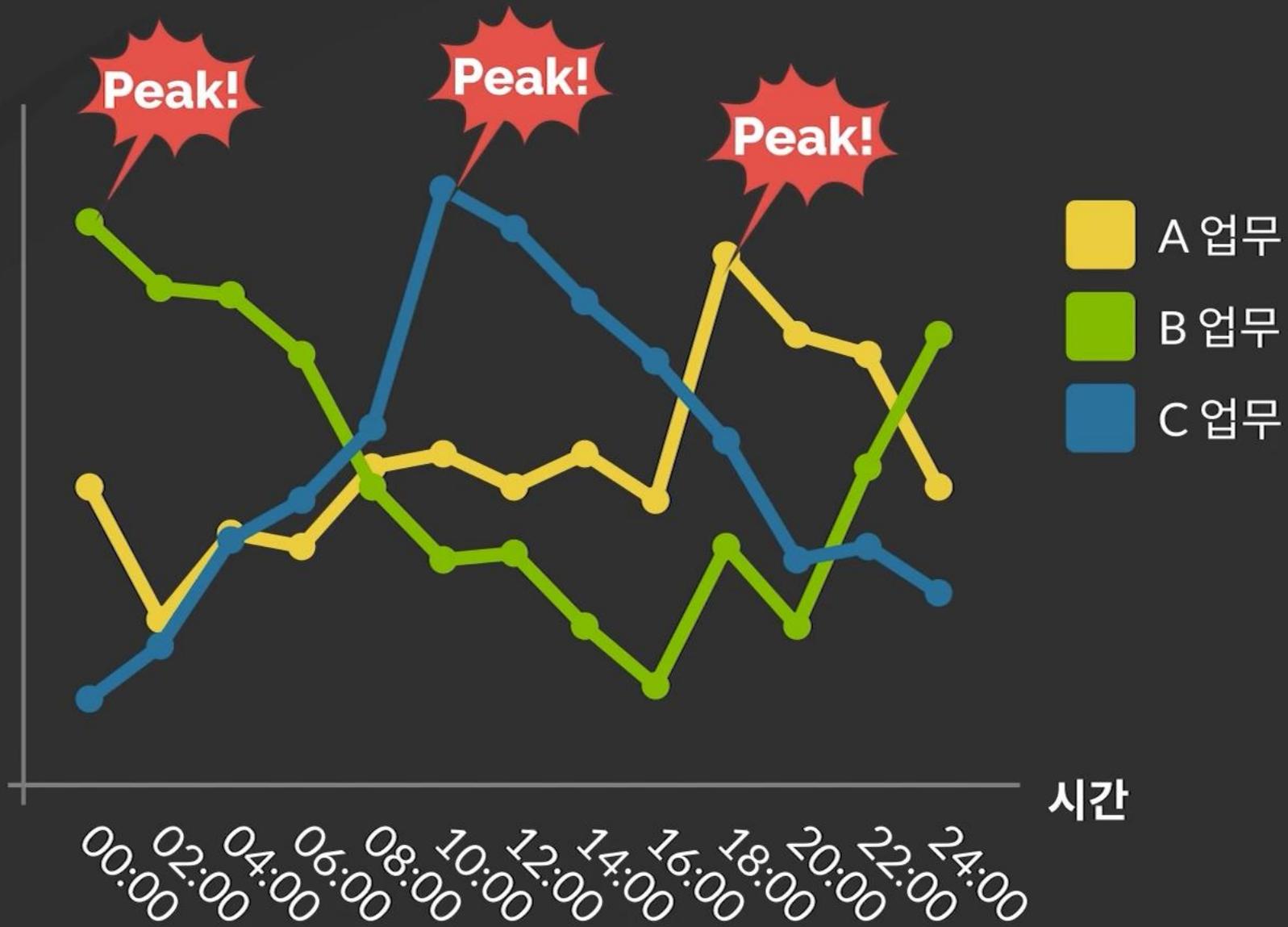
LDAP/Active Directory과 연동한 로그인 기능 제공

Application Performance Management

클라우드 네이티브 구축 사례
- 통계청 차세대 나라 통계



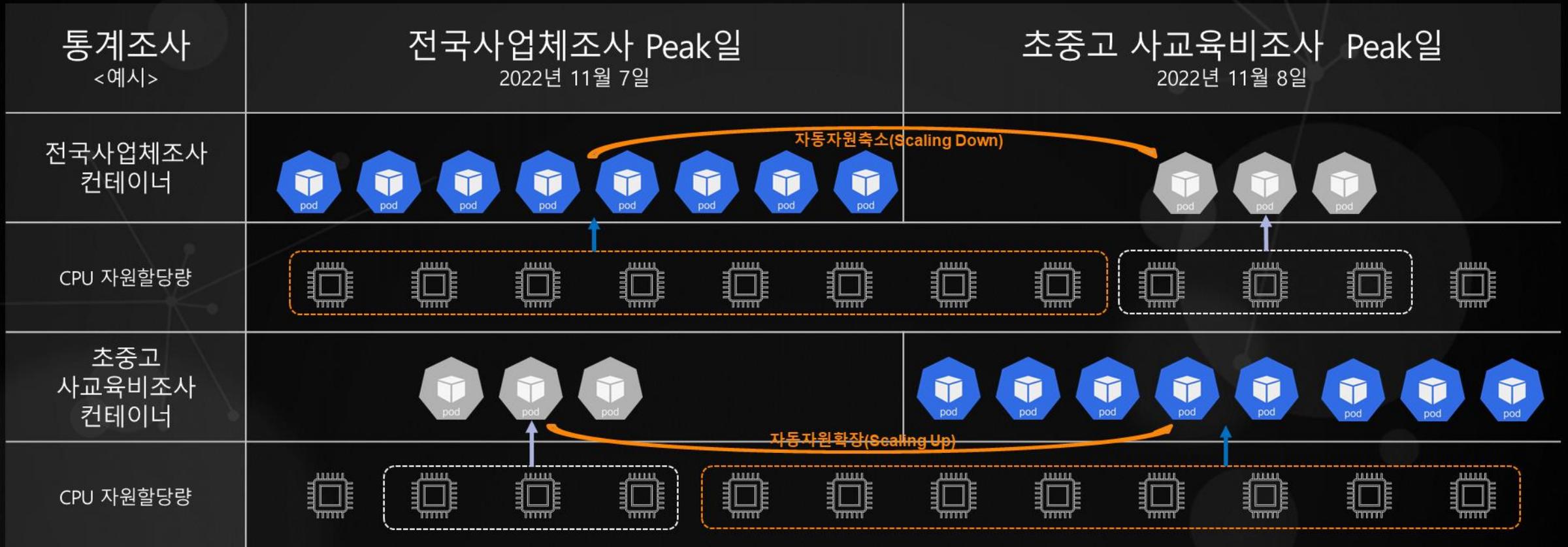
부하량



시간

차세대 나라 통계 - Peak 시점 별 자동 자원 할당 활용 예시

- 관리자의 개입 없이 시스템 사용량에 따라 자동으로 확장/축소
- 한정된 서버 자원을 Peak 시점에 따라 효율적으로 배분하여 사용 가능
- Peak 시점이 다른 통계조사에 대해 사람의 개입 없이 자동자원할당이 가능



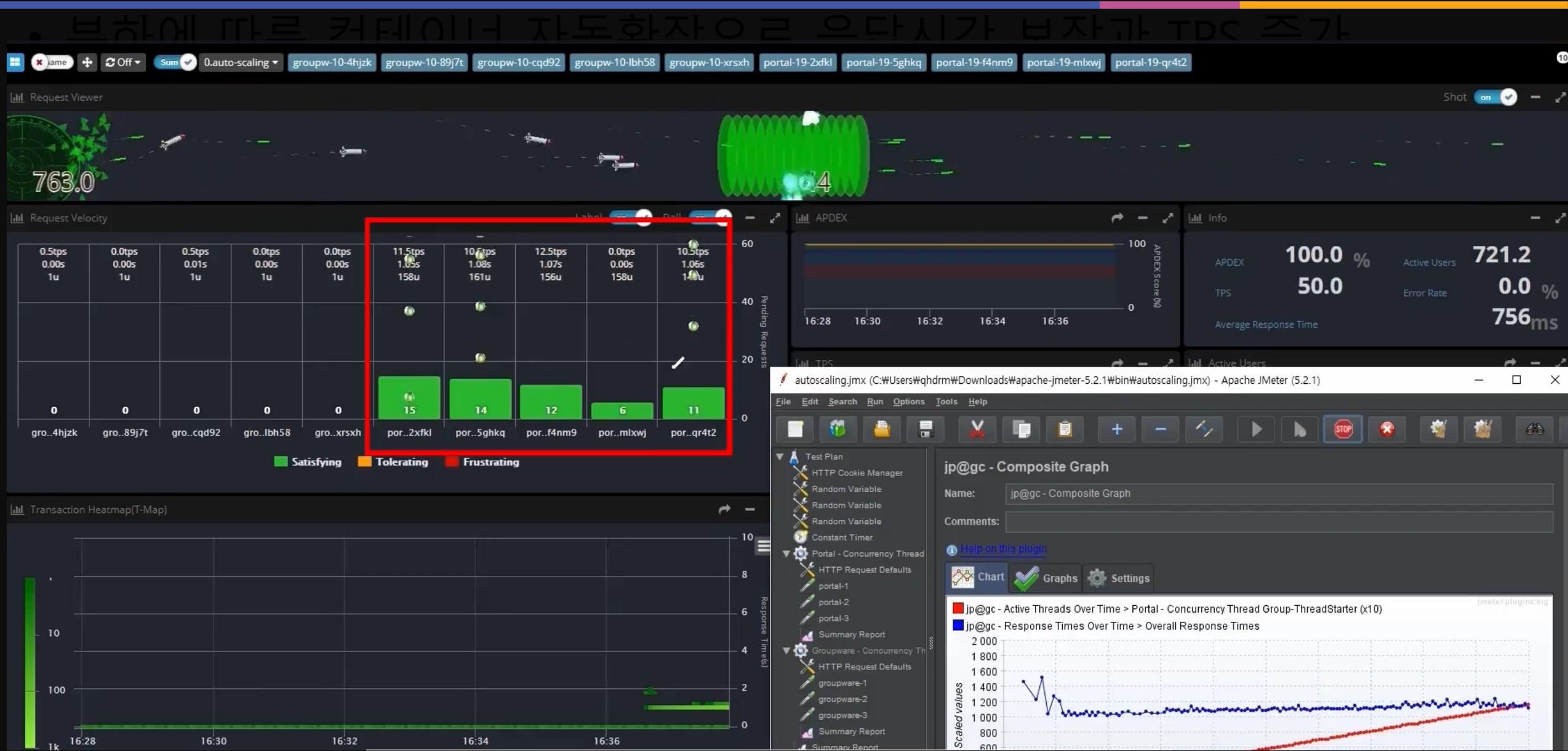
DEMO - 자동 자원 할당 테스트 시나리오

- 자동 자원 할당(Auto Scaling) 지원 환경과 미지원 환경을 Cocktail APM을 사용하여 비교
- Peak 시점이 다른 통계조사에 대해 사람의 개입 없이 자동 자원 할당이 가능한 지 테스트
- 비교를 위해 동일한 샘플 애플리케이션으로 동일한 부하로 테스트

시나리오 번호	테스트 시나리오	테스트 시나리오에서 확인 사항
T-01	기존 나라 통계 환경 인스턴스 개수 한정된 환경	자동 자원 할당이 더 많은 부하를 처리할 수 있는지 비교할 Base 테스트
T-02	전국사업체조사 Peak일 때 자동 자원 할당이 되는 환경에서 부하 테스트	자동 자원 할당 환경에서 더 많은 부하를 처리할 수 있는지 T-01과 비교
T-03	초중고 사교육비조사 Peak일 때 자동 자원 할당이 되는 환경에서 부하 테스트	Peak 시간이 다른 통계조사에 대해 사람의 개입 없이 처리가 가능한 지 확인

- ▶ 부하 테스트 도구 : JMeter v5.2
- ▶ 부하 테스트 방법 : 400 가상 유저, Ramp Up 5분, 5분간 부하 유지
- ▶ 인스턴스 개수 : **T-01** : 5개, **T-02~T-03** : 5 ~ 10개 Auto Scaling 설정

T-01 : 인스턴스 개수 한정된 환경 (기존 나라 통계)



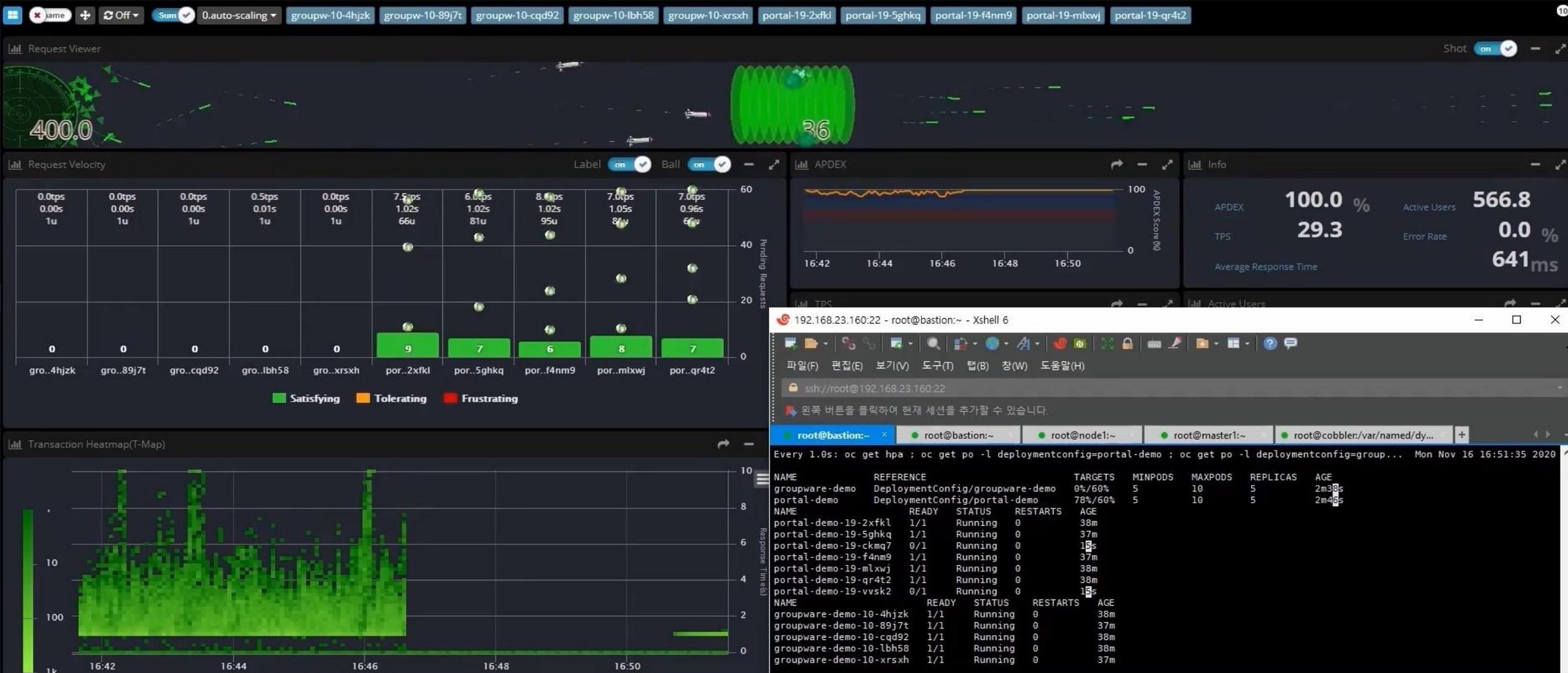
The image displays a performance monitoring dashboard for Cocktail APM and the Apache JMeter interface. The dashboard shows various metrics for different instance groups, including Request Velocity, APDEX, and Transaction Heatmap. A red box highlights a specific data point in the Request Velocity table.

Instance Group	Request Velocity (tps)	Request Velocity (s)	Request Velocity (u)
gro..4hjkz	0.5	0.00s	1u
gro..89j7t	0.0	0.00s	1u
gro..cq92	0.5	0.01s	1u
gro..lbh58	0.0	0.00s	1u
gro..xrsxh	0.0	0.00s	1u
por..2xfkl	11.5	1.03s	158u
por..5ghkq	10.7	1.08s	161u
por..f4nm9	12.5	1.07s	156u
por..mlxwj	0.0	0.00s	158u
por..qr4t2	10.5	1.06s	150u

The Apache JMeter interface shows a test plan with various components like HTTP Cookie Manager, Random Variable, Constant Timer, and Portal - Concurrency Thread. The 'jp@gc - Composite Graph' is visible, showing active threads and response times over time.

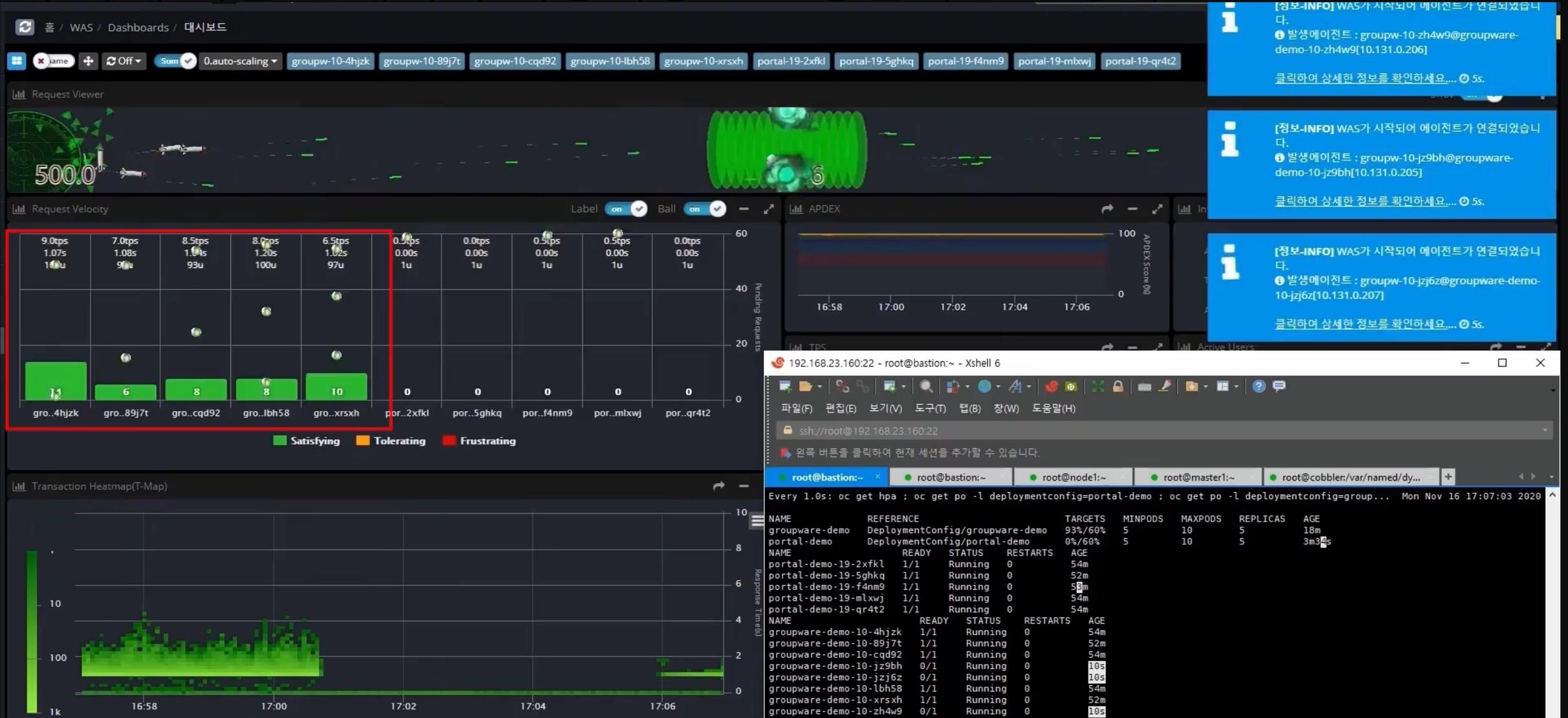
T-02 : 전국사업체조사 Peak일 때 자동부하분산 환경 부하테스트

부하에 따른 컨테이너 자동확장으로 응답시간 보장과 TPS 증가



T-03 : 초중고 사교육비조사 Peak일 때 자동부하분산 환경 부하테스트

부하에 따른 컨테이너 자동확장으로 응답시간 보장과 TPS 증가



Request Velocity

Group	TPS	RT	U
gro..4hjkz	9.0tps	1.07s	14u
gro..89j7t	7.0tps	1.08s	9u
gro..cq92	8.5tps	1.01s	93u
gro..lbh58	8.0tps	1.20s	100u
gro..xrsxh	6.5tps	1.02s	97u
por..2xfkl	0.5tps	0.00s	1u
por..5ghkq	0.0tps	0.00s	1u
por..f4nm9	0.5tps	0.00s	1u
por..mbxwj	0.5tps	0.00s	1u
por..qr4t2	0.0tps	0.00s	1u

Transaction Heatmap(T-Map)

Terminal Output:

```

Every 1.0s: oc get hpa ; oc get po -l deploymentconfig=portal-demo ; oc get po -l deploymentconfig=group... Mon Nov 16 17:07:03 2020
NAME          REFERENCE          TARGETS  MINPODS  MAXPODS  REPLICAS  AGE
portal-demo   DeploymentConfig/portal-demo  93%/60%  5         10        5          18m
portal-demo   DeploymentConfig/portal-demo  0%/60%  5         10        5          3m34s
NAME          READY  STATUS   RESTARTS  AGE
portal-demo-19-2xfkl  1/1    Running  0         54m
portal-demo-19-5ghkq  1/1    Running  0         52m
portal-demo-19-f4nm9  1/1    Running  0         50m
portal-demo-19-mbxwj  1/1    Running  0         54m
portal-demo-19-qr4t2  1/1    Running  0         54m
NAME          READY  STATUS   RESTARTS  AGE
groupware-demo-10-4hjkz  1/1    Running  0         54m
groupware-demo-10-89j7t  1/1    Running  0         52m
groupware-demo-10-cq92   1/1    Running  0         54m
groupware-demo-10-jz9bh  0/1    Running  0         10s
groupware-demo-10-jz6z  0/1    Running  0         10s
groupware-demo-10-lbh58  1/1    Running  0         54m
groupware-demo-10-xrsxh  1/1    Running  0         52m
groupware-demo-10-zh4w9  0/1    Running  0         10s
    
```

[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.
 발생에이전트 : groupw-10-zh4w9@groupware-demo-10-zh4w9[10.131.0.206]
 클릭하여 상세한 정보를 확인하세요... 5s.

[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.
 발생에이전트 : groupw-10-jz9bh@groupware-demo-10-jz9bh[10.131.0.205]
 클릭하여 상세한 정보를 확인하세요... 5s.

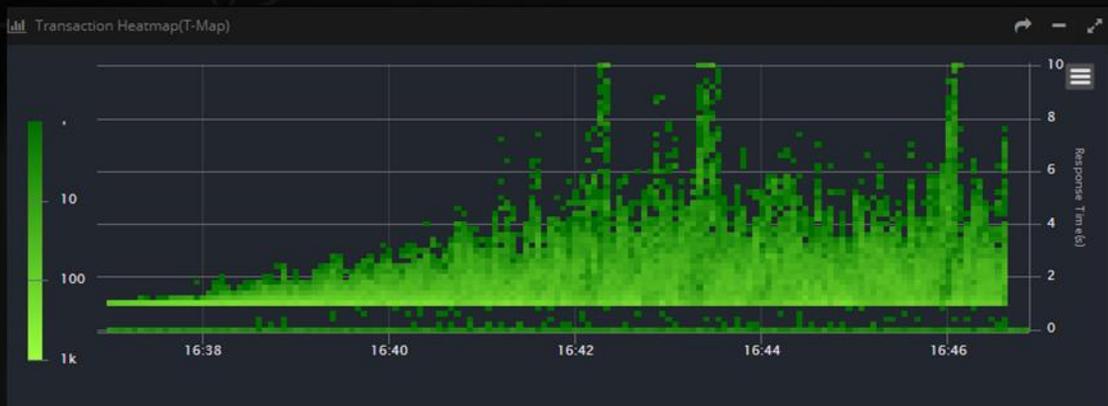
[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.
 발생에이전트 : groupw-10-jz6z@groupware-demo-10-jz6z[10.131.0.207]
 클릭하여 상세한 정보를 확인하세요... 5s.

DEMO - 부하 테스트 결과 비교

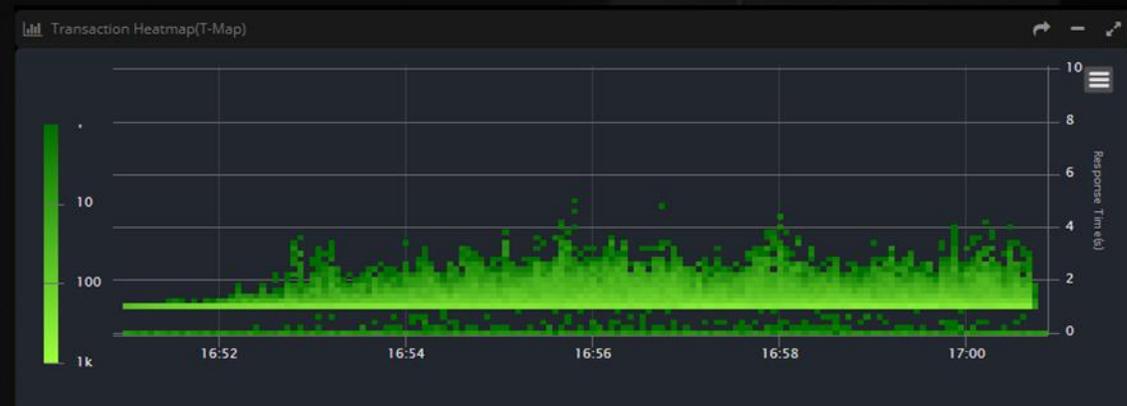
- 기존환경과 비교하여 **자동자원할당이 되는 환경이 1.7 배 많은 양을 처리하며, 평균 응답 시간이 2.5 배 빠름**
- **Peak 시점이 다른 통계조사에 대해 별도 개입없이 자동 자원 할당이 가능함을 확인**

테스트 케이스	테스트 내용	시뮬레이션 테스트 환경	처리량	처리량 비교	평균응답시간	응답시간비교	최소응답시간	최대응답시간	TPS
T-01	자동자원할당이 안 되는 환경에서 부하테스트	기존 나라통계 환경	53,582	100%	2,374	100%	1,010	19,979	88
T-02	Peak 시점이 다른 자동 자원할당이 되는 환경에서 부하테스트	전국사업체 조사 Peak 일 때 환경	93,869	169%	919	258%	10	4,589	155
T-03	Peak 시점이 다른 자동 자원할당이 되는 환경에서 부하테스트	초중고 사교육비 조사 Peak일 때 환경	91,394	164%	971	244%	10	5,134	151

T-01 자동자원할당 안되는 기존 환경 응답시간분포



T-02/03 자동자원할당이 되는 환경 응답시간분포



Application Performance Management

데모로 이해하는 클라우드 네이티브
- 하이브리드 클라우드 데모

클라우드 네이티브 특징

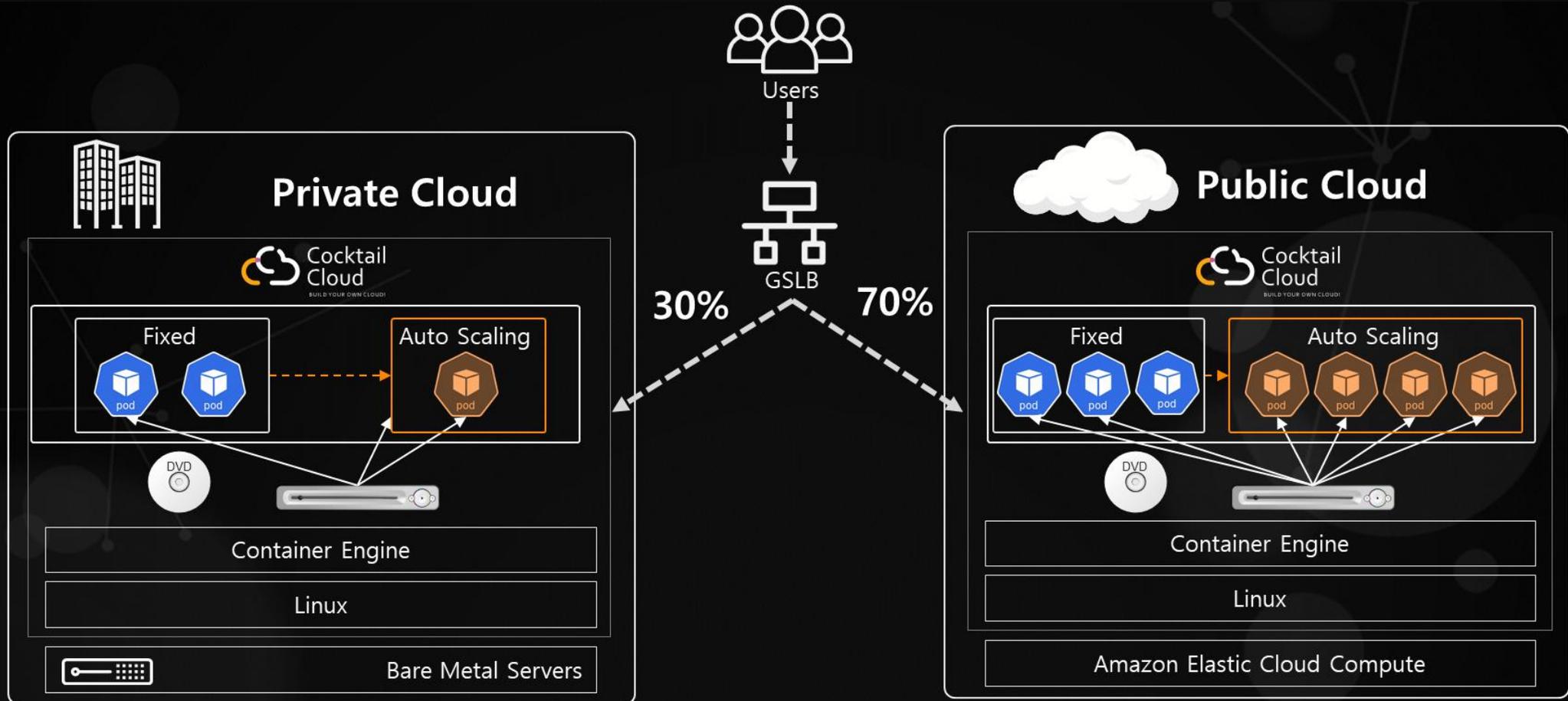
- 클라우드 네이티브는 작고, 가볍고 손쉽게 배포
- 클라우드 환경에서의 →서비스 배포는 전 세계에 한번의 클릭으로 애플리케이션 배포
→ 전 세계를 상대로 서비스 가능



Source: 클라우드 네이티브 추진 시 고려사항 (교육 교재)
클라우드 네이티브 기반 행정·공공 서비스 확산 지원 - 한국지능정보사회진흥원

하이브리드 클라우드 데모 - 사용자 증가 자동 확장

- 하이브리드 클라우드로 운영되는 포털을 접속하는데 자동확장이 필요한 사용자 폭주상태를 가정
- 내부 클라우드와 외부 클라우드에 모두 동일한 홈페이지 서비스를 하고 프라이빗 30% vs 퍼블릭 70%로 운영 중



하이브리드 클라우드 데모 - 전자정부 F/W 포탈에 대한 글로벌 서버 부하 분산

- GSLB(Global Server Load Balancing)을 통한 업무 부하 분산 데모
- 전자정부 F/W 포탈 서비스를 내부와 외부 클라우드에서 동시 운영

<input checked="" type="checkbox"/>	portal.egov.namutech.co.kr	가중치 기반
<input type="checkbox"/>	portal.egov.namutech.co.kr	가중치 기반
<input type="checkbox"/>	private.egov.namutech.co.kr	단순
<input type="checkbox"/>	public.egov.namutech.co.kr	단순

<https://portal.egov.namutech.co.kr>



30%

70%

<https://private.egov.namutech.co.kr>



<https://public.egov.namutech.co.kr>



하이브리드 클라우드 데모 - 자동 확장 이전

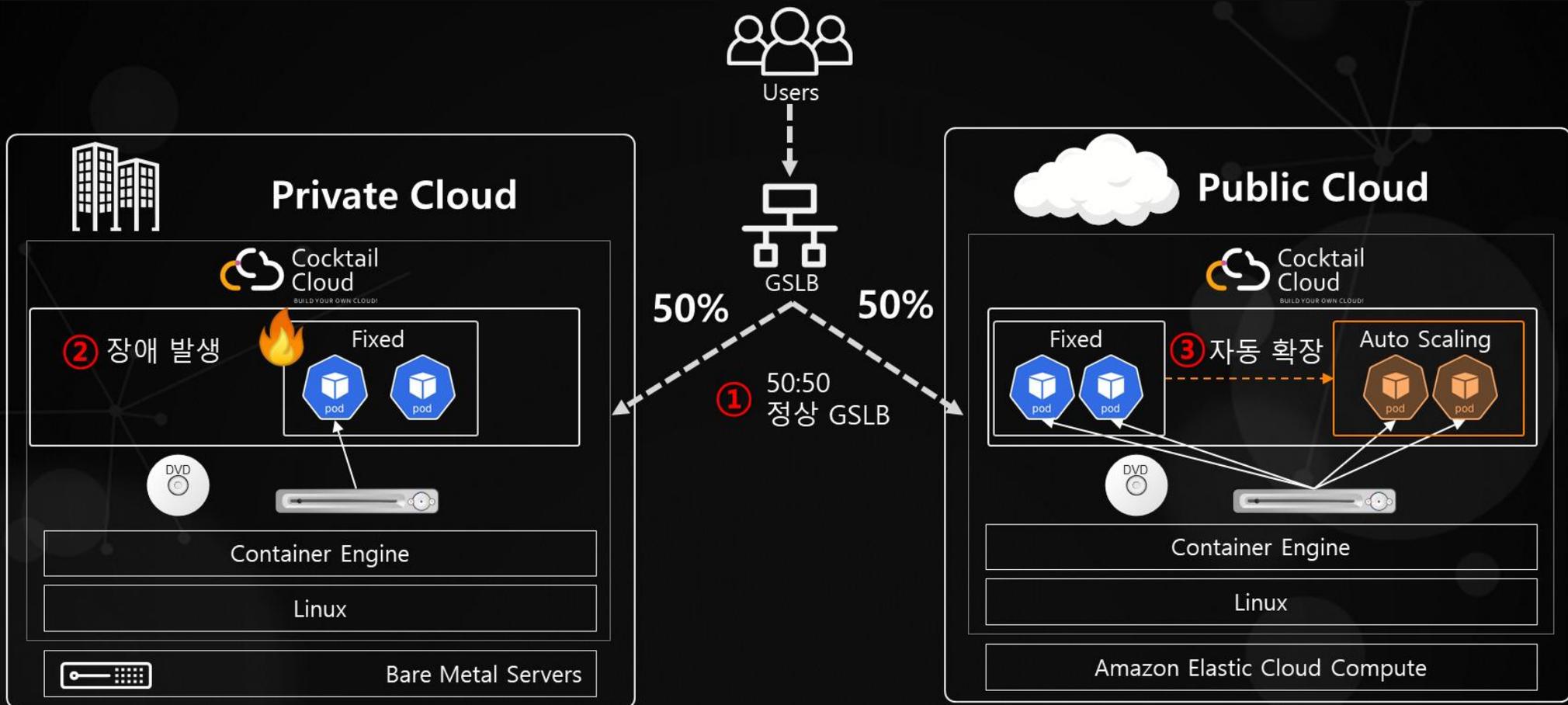


하이브리드 클라우드 데모 - 자동 확장 이후



하이브리드 클라우드 데모 - Active Active DR

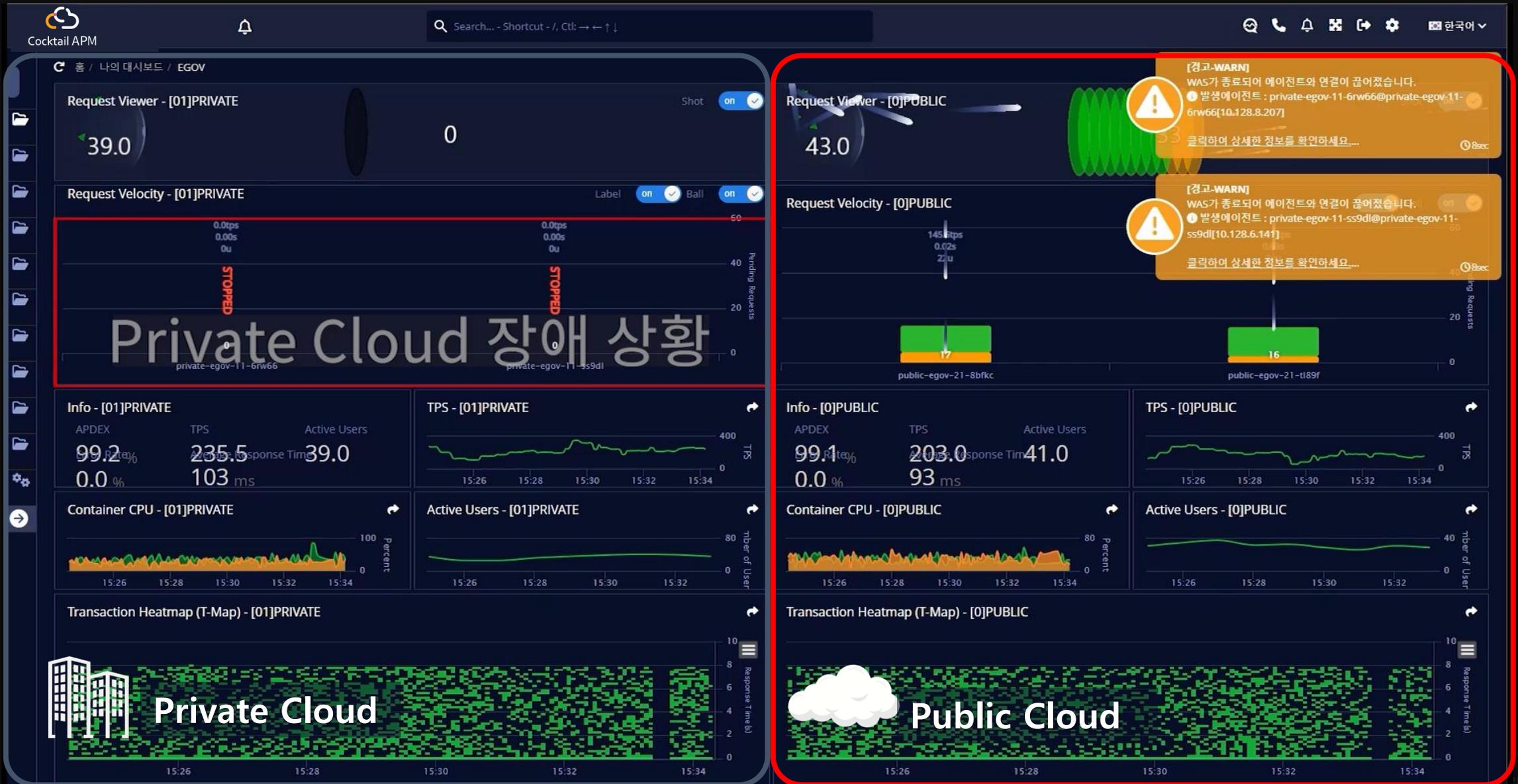
- 하이브리드 클라우드로 운영되는 포털을 접속하는데 자동확장이 필요한 사용자 폭주상태를 가정
- 내부 클라우드와 외부 클라우드에 모두 동일한 홈페이지 서비스를 하고 프라이빗 30% vs 퍼블릭 70% 로 운영 중



하이브리드 클라우드 데모 - 장애 발생 이전 Active-Active GSLB



하이브리드 클라우드 데모 - Public Cloud에서 부하에 따른 자동 확장



감사합니다.

