

## 웹 애플리케이션 개발환경 및 관리영역의 새로운 변화 IBM WebSphere Application Server V8.5

발표자 : 이정운 과장

직함 : 웹스피어 기술영업팀, 소프트웨어 사업부



# 무엇이 보이나요



## WAS라고 다 똑같은 WAS가 더 이상 아닙니다

- IBM WAS V8.5는 개발자들을 위한 **경량화된 IBM WAS Liberty Profile**을 제공합니다. 약 50MB 정도의 하드디스크만 차지하며, 구동시간이 5초도 걸리지 않는 아주 가벼운 WAS입니다.
- 애플리케이션 관리 기능을 통해 가동시간 **중단 없이 애플리케이션 업그레이드**가 가능합니다.
- 자가 보호 및 치유 기능을 통하여 **미리 문제를 감지하고 조치**를 취할 수 있습니다.
- WAS의 상태를 체크하여 최고의 성능과 가용성을 낼 수 있도록 **지능적으로 요청을 동적 분배**할 수 있습니다.

# 목차

## ■ 시장의 트렌드

## ■ WAS V8.5

- Liberty Profile을 통한 개발자 경험 향상
- 애플리케이션 복원력
- 작업과 제어 향상

## ■ 요약

# 애플리케이션 인프라 트렌드

빠른 애플리케이션 개발과 전달을 위하여 개발 및 운영 라이프사이클에 대한 통합, 자동화, 단순화를 요구

모바일, 소셜, 클라우드 애플리케이션의 폭발적 증가에 따라 미들웨어 인프라에 대한 추가적인 요구사항 증가

하루에도 엄청나게 생성되는 데이터에 대항하여 대용량 트랜잭션 볼륨 지원 및 대기시간 최소화를 위해 탄력적인 캐싱 기술 요구

다양한 폼 팩터와 접속 방법에 따라서 탄력성과 확장성, 멀티-테넌시, 컨텍스트 등을 제공하기 위하여 클라우드 전달 모델 사용

# WAS에 대한 고객의 지속적인 변화 요구

WAS의 향상과 변화에 대해서 고객과 분석가들은 지속적으로 많은 요구사항 전달

- 개발자의 생산성 향상
- "Low-End" 애플리케이션 환경을 보다 단순화
- "Low-End"와 "High-End" WAS 환경 사이의 플랫폼 정확성(Fidelity)
- 미션 크리티컬 애플리케이션을 위한 보다 높은 레벨의 복원력(Resiliency)
- 작업 제어의 향상
- 사용자 만족도에 대한 지속적인 향상
- 애플리케이션 환경에 대해 비용 절감

# 고객의 요구에 대한 IBM의 답변

## Change (변화)

# 목차

- 시장의 트렌드

- **WAS V8.5**

- Liberty Profile을 통한 개발자 경험 향상
- 애플리케이션 복원력
- 작업과 제어 향상

- 요약

# 새로운 IBM WAS V8.5

기존 WAS와 같지만 또 다른 새로운 차세대 WAS인 IBM WAS V8.5

## 개발자 경험



빠르고, 유연하고,  
단순화된  
애플리케이션 개발

- 아주 가벼운 WAS 서버인 Liberty Profile 제공
- 확장된 툴링과 WAS 툴링 번들
- JDK7 지원
- OSGi 프로그래밍 모델 강화
- OSGi 애플리케이션에서 EJB 지원
- 마이그레이션 툴킷
- 웹 2.0 & 모바일 툴킷: IBM Worklight 통합
- SCA OASIS 프로그래밍 모델

## 애플리케이션 복원력



지능적인 관리  
& 강화된  
복원력

- 애플리케이션 에디션 관리
- 애플리케이션 서버 자가관리(Health Management)
- 새로운 지능적 라우팅 능력
- 메시징 인프라 복원력
- 동적 클러스터링
- WAS에서 메모리 누수(Memory Leak) 감지 및 방지

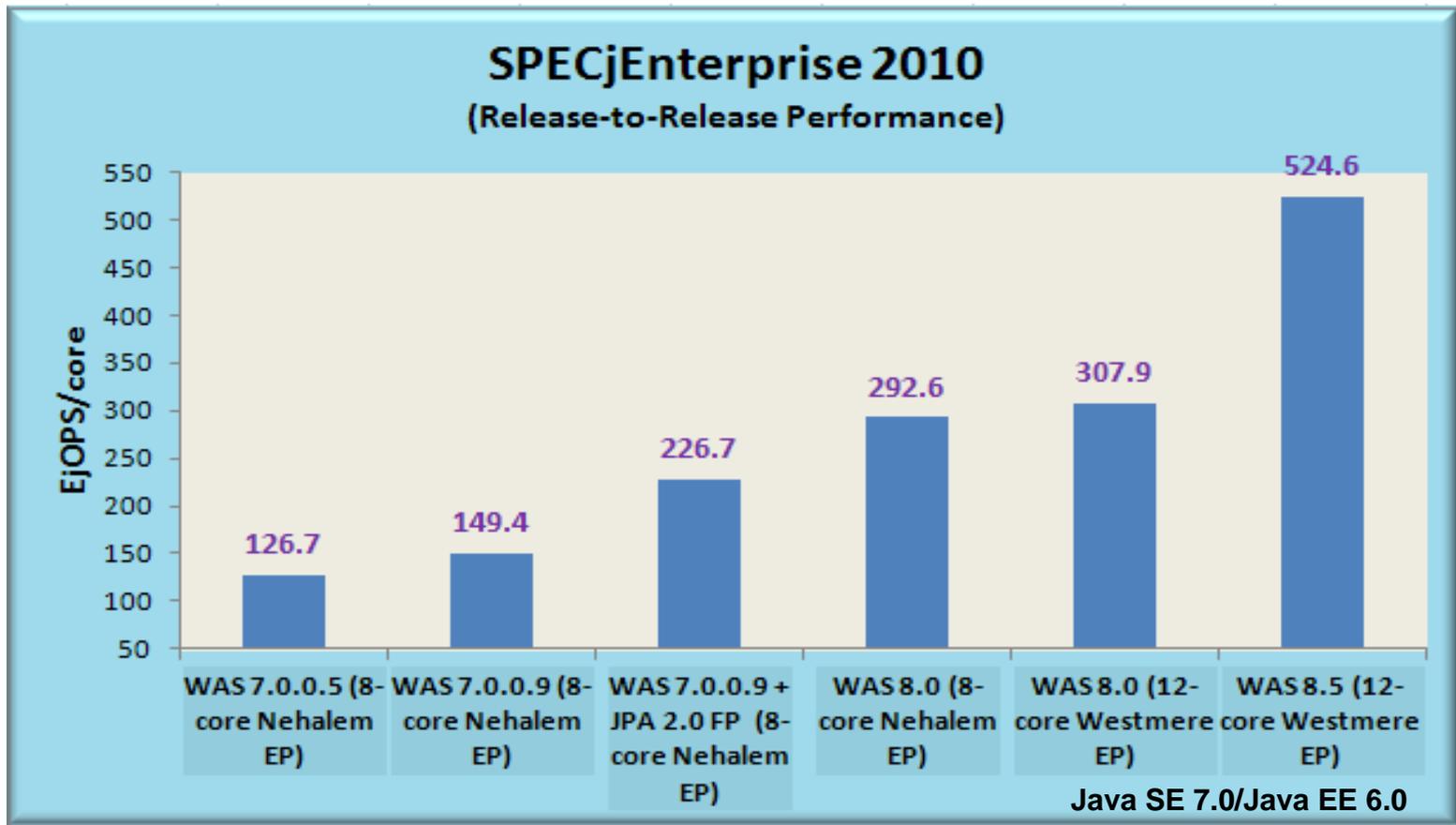
## 작업과 제어



작업과 보안, 제어,  
통합을 향상

- 선택적인 JDK 활용
- WebSphere 배치(Batch) 강화
- 관리자 보안 감사
- OSGi Blueprint 보안 향상
- Cross Component Trace (XCT)
- IBM Support Assistant 강화
- 보다 향상된 로그와 추적 필터링

# WebSphere 버전 별 성능 비교



**Consistent Performance gains across WAS Releases**

As per SPEC Published Data as of 4/26/2012

<http://www.spec.org/jEnterprise2010/results/jEnterprise2010.html>

# 새로운 IBM WAS V8.5

기존 WAS와 같지만 또 다른 새로운 차세대 WAS인 IBM WAS V8.5

## 개발자 경험



빠르고, 유연하고,  
단순화된  
애플리케이션 개발

- 아주 가벼운 WAS 서버인 Liberty Profile 제공
- 확장된 툴링과 WAS 툴링 번들
- JDK7 지원
- OSGi 프로그래밍 모델 강화
- OSGi 애플리케이션에서 EJB 지원
- 마이그레이션 툴킷
- 웹 2.0 & 모바일 툴킷: IBM Worklight 통합
- SCA OASIS 프로그래밍 모델

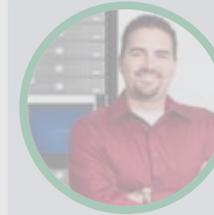
## 애플리케이션 복원력



지능적인 관리  
& 강화된  
복원력

- 애플리케이션 에디션 관리
- 애플리케이션 서버 자가관리(Health Management)
- 새로운 지능적 라우팅 능력
- 메시징 인프라 복원력
- 동적 클러스터링
- WAS에서 메모리 누수(Memory Leak) 감지 및 방지

## 작업과 제어

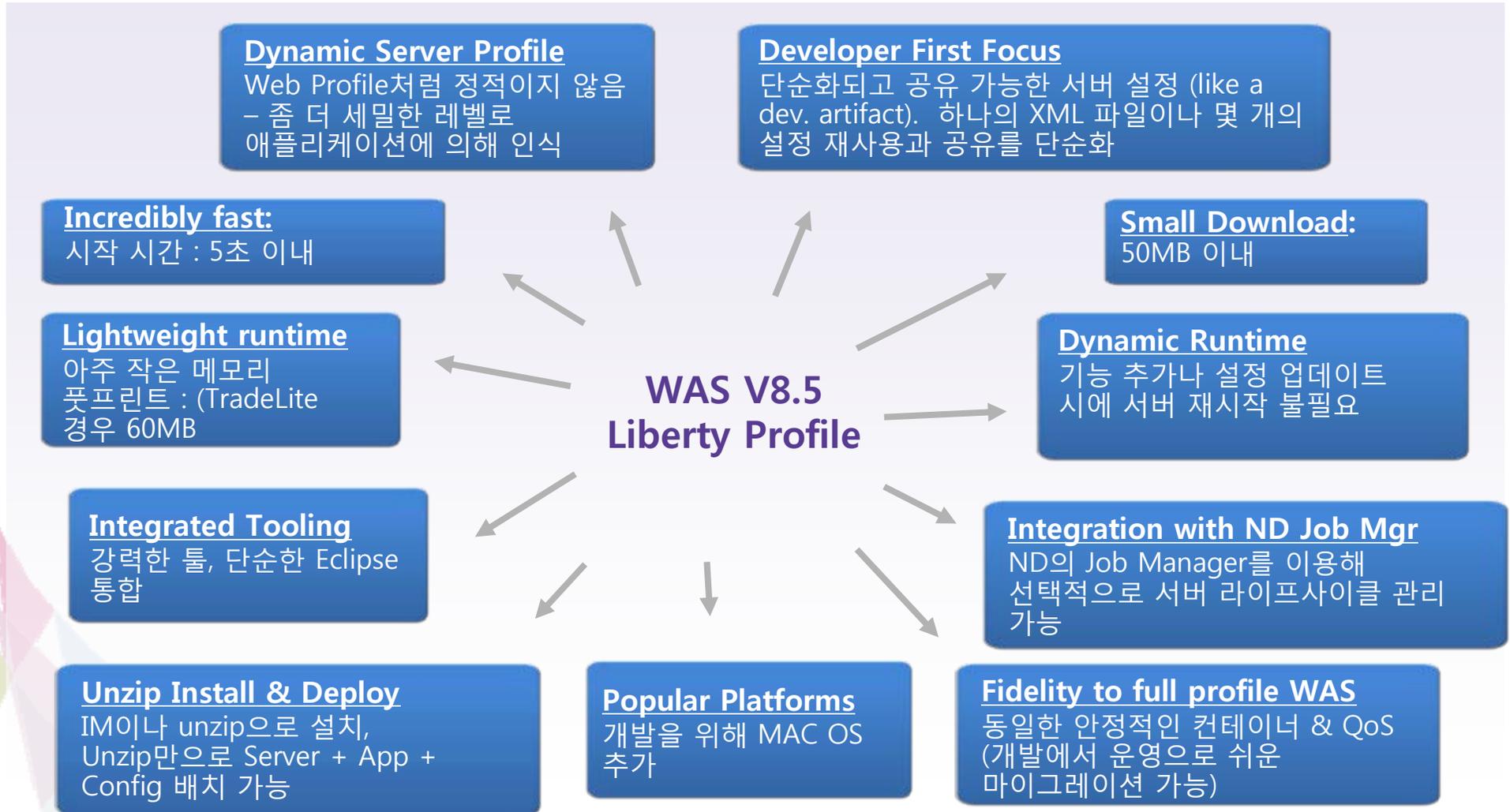


작업과 보안, 제어,  
통합을 향상

- 선택적인 JDK 활용
- WebSphere 배치(Batch) 강화
- 관리자 보안 감사
- OSGi Blueprint 보안 향상
- Cross Component Trace (XCT)
- IBM Support Assistant 강화
- 보다 향상된 로그와 추적 필터링

# IBM WAS V8.5 Liberty Profile

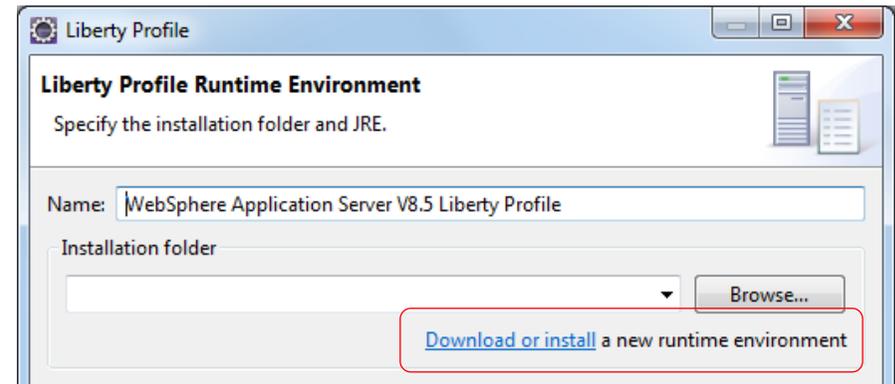
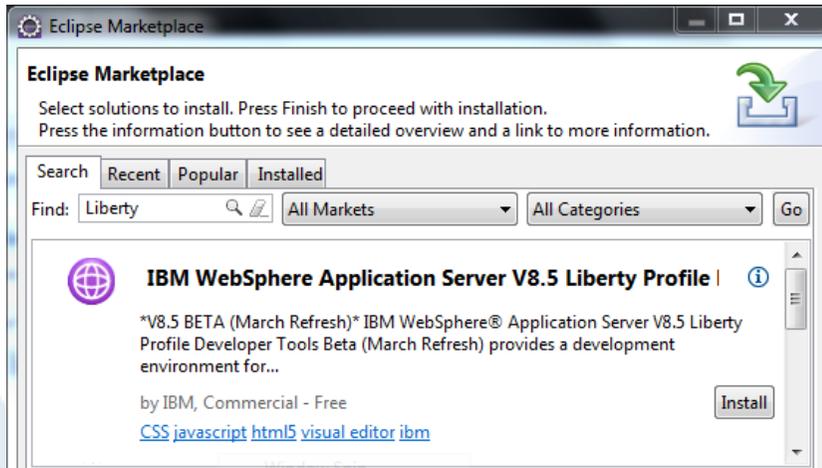
웹, OSGi 및 모바일 애플리케이션의 빠른 개발 및 전달을 위해 더 경량화된 WAS



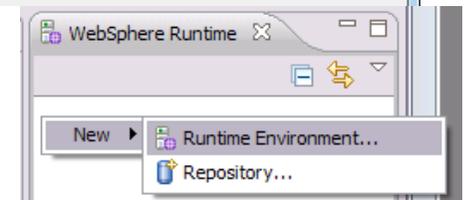
# 손쉬워진 설치

- 개발자를 위한 개발도구와 런타임이 무료
- 해당 개발도구를 위한 Eclipse 기능 설치만 필요: 서버 프로파일을 위해 40MB zip 파일 다운로드로 충분
- Installation Manager를 이용한 설치 지원
- 약 2분 정도면 아무것도 없는 상태에서 준비가 완료된 상태로 전환 가능

## 1. Eclipse 기능 추가를 위한 WAS Developer Tools 설치



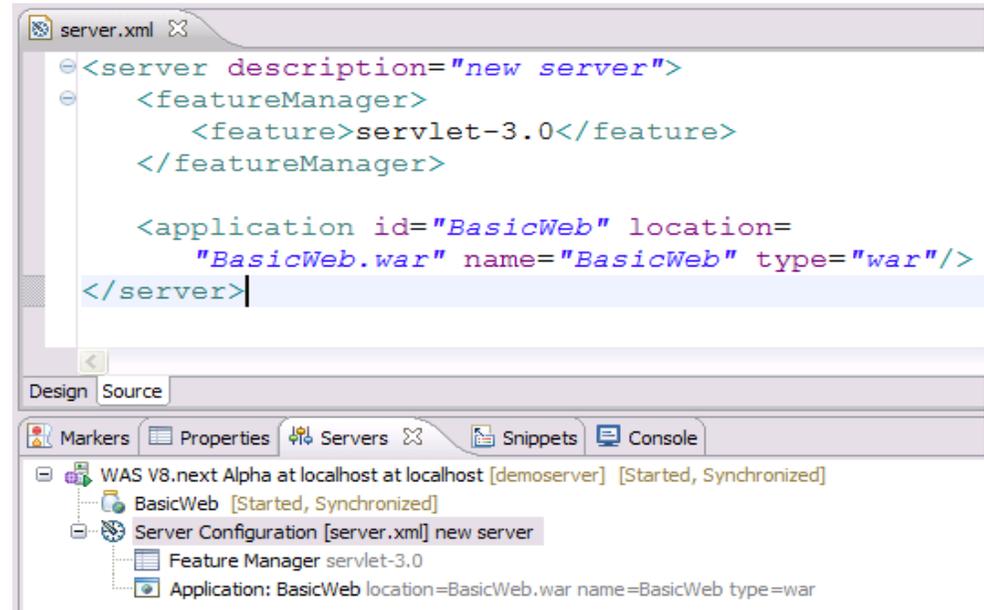
2. 해당 툴을 이용해서 WAS Liberty Profile을 다운로드 하거나 WASdev.net 사이트에서 40MB zip 파일 하나만 다운로드 받으면 충분



# 단순화된 서버 설정

단 하나의 server.xml 파일로 모든 설정 관리 가능

- 가장 단순한 케이스 : 모든 서버의 설정이 하나의 XML 파일로 가능
- 워크스페이스 안에서 수정 가능
- 추출 및 공유, 버전 관리가 가능



```
<server description="new server">
  <featureManager>
    <feature>servlet-3.0</feature>
  </featureManager>

  <application id="BasicWeb" location=
    "BasicWeb.war" name="BasicWeb" type="war"/>
</server>
```

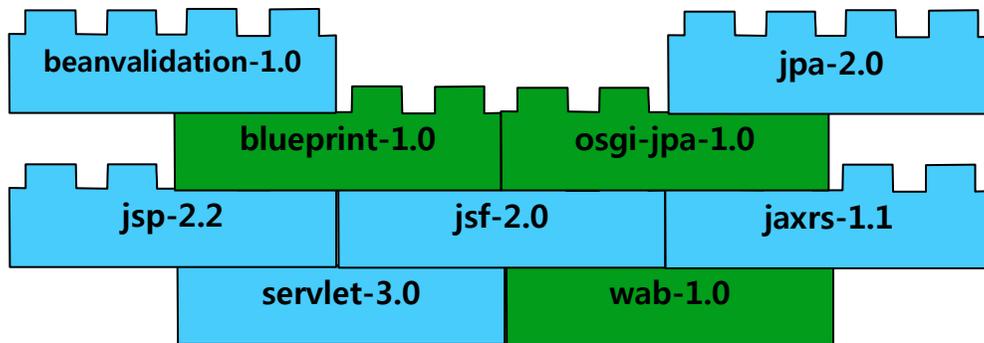
The screenshot shows the XML code in the editor and the project explorer below it. The project explorer shows a hierarchy: WAS V8.next Alpha at localhost at localhost [demoser] [Started, Synchronized] > BasicWeb [Started, Synchronized] > Server Configuration [server.xml] new server > Feature Manager servlet-3.0 > Application: BasicWeb location=BasicWeb.war name=BasicWeb type=war.



관리 콘솔이나 wsadmin, 확장 EAR  
등이 더 이상 필요 없음

# 애플리케이션의 기능 집합을 동적 적용 가능

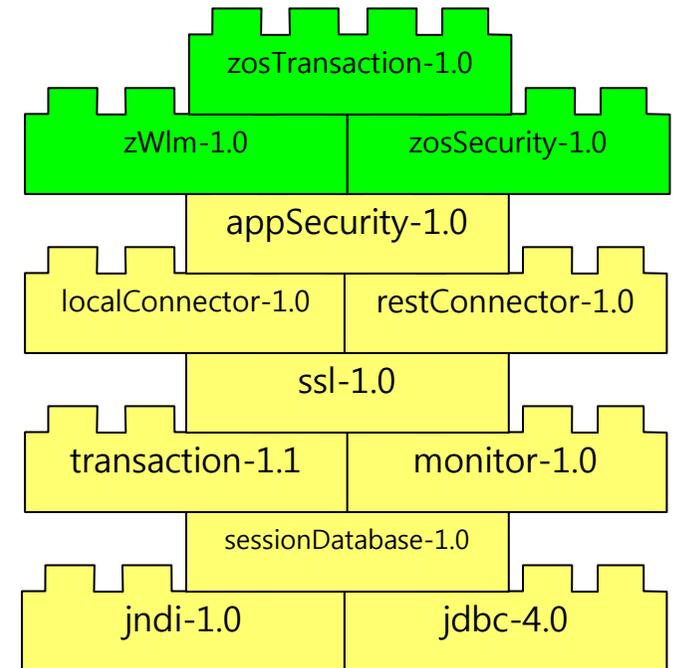
## 애플리케이션



## WAS V8.5 Liberty Profile

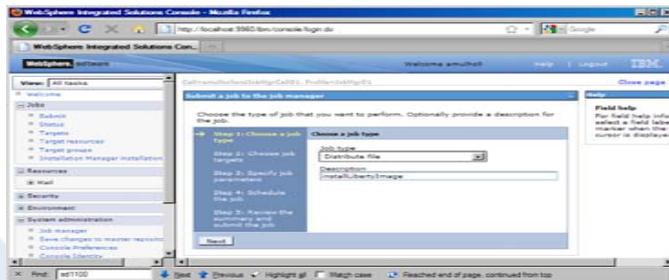
- 웹 및 모바일 애플리케이션
- OSGi 애플리케이션
- JPA(Java Persistence API)
- 트랜잭션, 보안 등의 연관된 서비스

## 운영환경



# Liberty Profile을 실제 운영 환경에 도입

- Liberty 서버 형태로 설정된 운영 인스턴스인 “Embedded Server” 프로파일
  - 테스트했던 애플리케이션, 설정과 서버 형태를 그대로 압축하는 방식을 고려
  - 애플리케이션에 집중 – 특정 애플리케이션에 미리 설정된 서버
- 배치 옵션
  - 관리가 필요 없을 경우 unzip 설치
  - “embedded server” 패키지로 분산 및 unzip을 통해서 관리가 필요할 경우 ND Job Manager로 “Liberty 배치”
    - “Light-touch” ND 관리 : 서버를 시작하거나 중지 가능
  - 서버 설정은 개발 환경과 동일하게 단순화된 XML 설정에 그대로 유지



WebSphere ND Job Manager

리모트 호스트로 임베디드 서버를 unzip

중앙에서 서버를 확인하거나 시작, 중지 가능

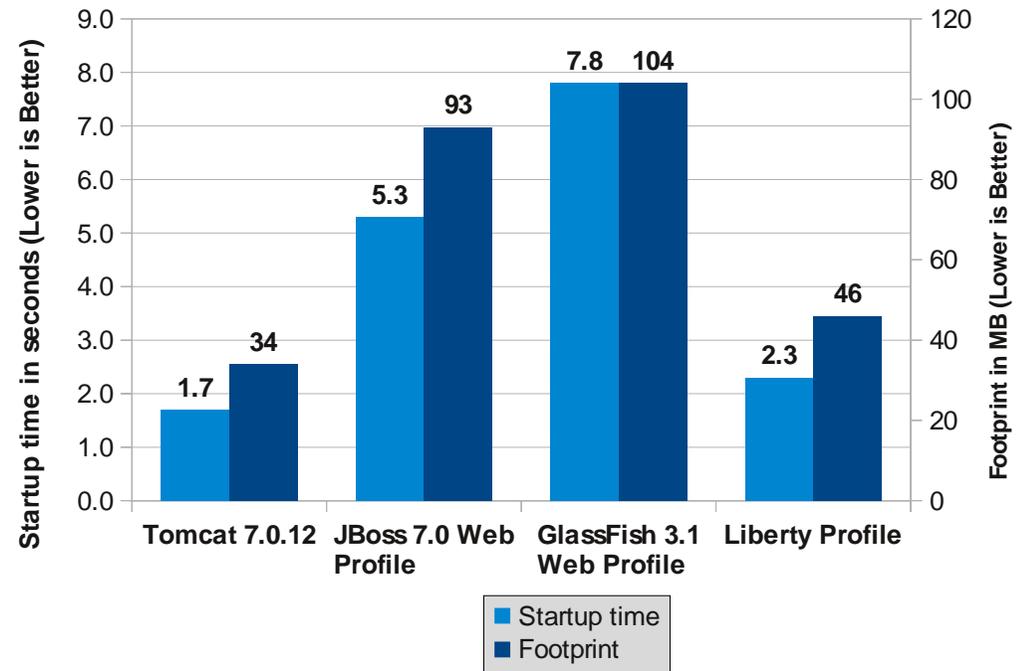


# Liberty Profile – 시작 시간 & 풋프린트

경량화된 개발 환경에 대한 이슈를 WebSphere에서 직접 해결할 수 있음

- Liberty Profile의 시작 시간 & 풋프린트가 Tomcat과 거의 동등함
- Liberty Profile은 JBoss Web Profile 시작 시간의 절반만 소요

Startup & Footprint Comparison of various lightweight servers



Note: Tomcat, JBoss, and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.

## System Info:

Lenovo T60p - 2 x 2.16 GHz Intel Core Duo T2600  
2GB RAM, Windows XP 32-bit

Apache Tomcat 7.0.12

JBoss Community Edition 7.0 Web Profile server

GlassFish Server 3.1 Open Source Edition Web Profile

WAS V8.Next Liberty Profile

(All servers had the TradeLite benchmark application installed)

# Liberty Profile – 처리량(Throughput)

경량화된 서버에서도 실제 운영환경 서버 속도로 서비스 요청 처리 가능

- Liberty Profile은 JBoss에 비해서는 최대 20%, Tomcat에 비해서는 25% 이상의 높은 성능을 보여줌

## System Info:

IBM x3550 – 4 x 1.86 GHz Intel Xeon E5320, 8 GB RAM  
RedHat Linux 5.3 32-bit

Apache Tomcat 7.0.12

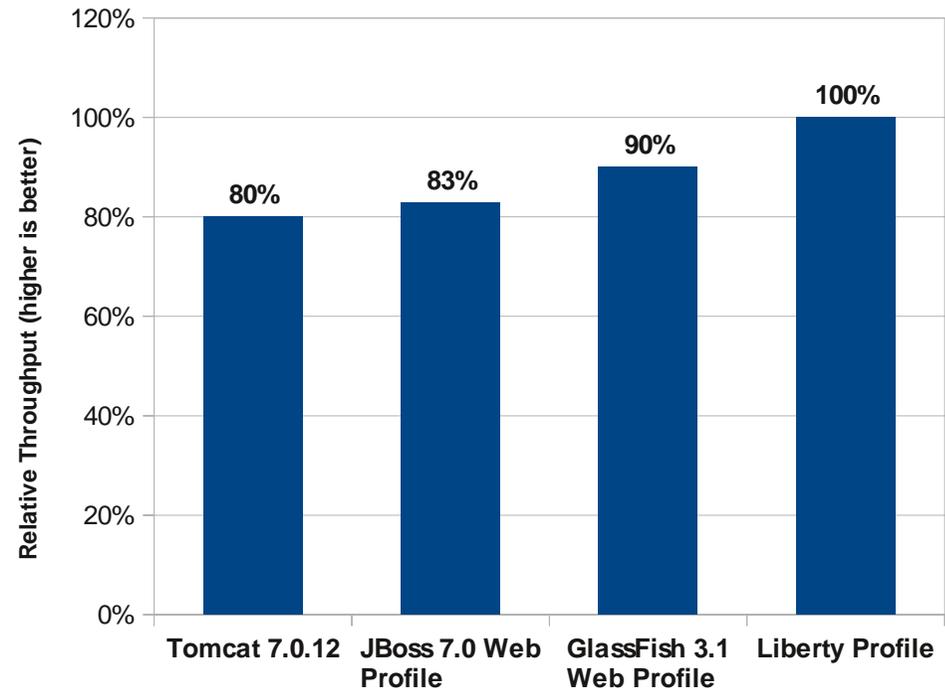
JBoss Community Edition 7.0 Web Profile server

GlassFish Server 3.1 Open Source Edition Web Profile

WAS V8.Next Liberty Profile

(All servers had the TradeLite benchmark application installed)

Throughput Comparison of various lightweight servers



*Note: Tomcat , JBoss, and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.*

# 요약 – 개발자들을 위한 가치

경량화된 WAS Liberty Profile 런타임으로 개발자들은 개발과 작업 경험에만 집중

- 테스트 서버 런타임에 대한 아주 작은 풋프린트
- 단순화되고 공유 가능한 설정
- 가장 빠른 서버 시작 시간
- WAS 프로파일 에디션과 완벽한 호환
- 개발자에게 무료 제공!
- 초기에는 웹과 OSGi, 모바일 애플리케이션에 집중
  - JPA(Java Persistence API), 트랜잭션, 보안을 포함

# 새로운 IBM WAS V8.5

기존 WAS와 같지만 또 다른 새로운 차세대 WAS인 IBM WAS V8.5

## 개발자 경험



빠르고, 유연하고,  
단순화된  
애플리케이션 개발

- 아주 가벼운 WAS 서버인 Liberty Profile 제공
- 확장된 툴링과 WAS 툴링 번들
- JDK7 지원
- OSGi 프로그래밍 모델 강화
- OSGi 애플리케이션에서 EJB 지원
- 마이그레이션 툴킷
- 웹 2.0 & 모바일 툴킷: IBM Worklight 통합
- SCA OASIS 프로그래밍 모델

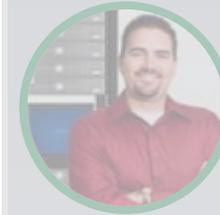
## 애플리케이션 복원력



지능적인 관리  
& 강화된  
복원력

- 애플리케이션 에디션 관리
- 애플리케이션 서버 자가관리(Health Management)
- 새로운 지능적 라우팅 능력
- 메시징 인프라 복원력
- 동적 클러스터링
- WAS에서 메모리 누수(Memory Leak) 감지 및 방지

## 작업과 제어

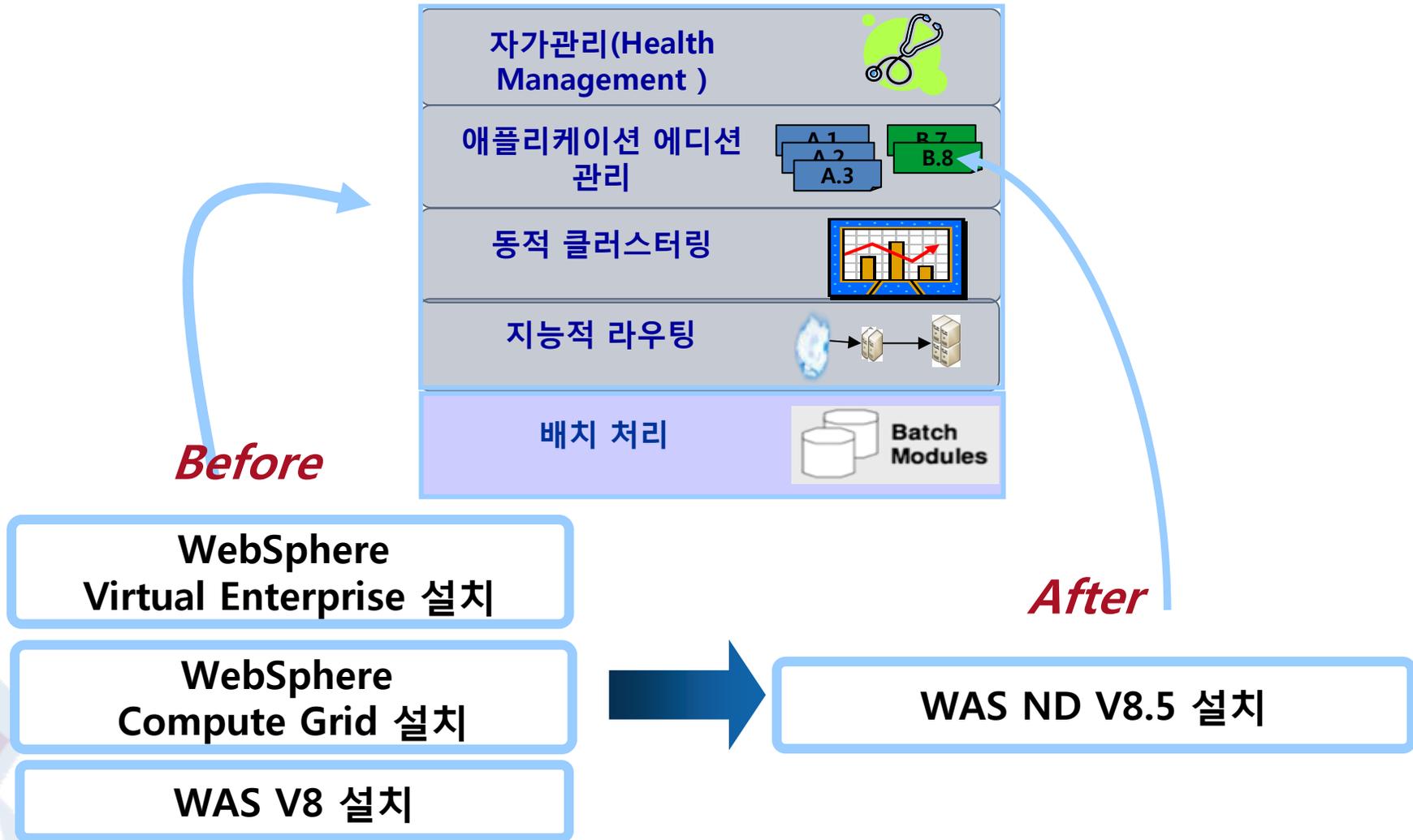


작업과 보안, 제어,  
통합을 향상

- 선택적인 JDK 활용
- WebSphere 배치(Batch) 강화
- 관리자 보안 감사
- OSGi Blueprint 보안 향상
- Cross Component Trace (XCT)
- IBM Support Assistant 강화
- 보다 향상된 로그와 추적 필터링

# WAS ND V8.5 복원력(Resiliency) 강화

자동화 컴퓨팅을 통한 서비스 품질 능력 확장



*Before*

*After*

WebSphere  
Virtual Enterprise 설치

WebSphere  
Compute Grid 설치

WAS V8 설치

WAS ND V8.5 설치

# 애플리케이션 서버 자가관리(Health Management)

운영상황에서 실시간으로 개별 서버들의 서비스 품질 및 자원 사용 상태를 지속적으로 감지하여 장애 상황을 예측하고 이를 자동으로 조치하고 관리자에게 통지

## ▪ 자동 조치 순서 (순서 조정 가능)

- 로그, 이메일 등으로 관리자에게 현재 상황 통지
- 해당 서버로 추가적인 요청 차단
- 향후 문제 원인 파악을 위해 Thread Dump
- 또는 Heap Dump 출력
  - 서버 재시작
  - 사용자 지정

## ▪ 체크 가능 조건

- 과도한 응답 시간
- 과도한 량의 타임아웃
- 서버 시작 후 경과 시간
- 서버가 처리한 요청의 수
- 메모리 사용률
- 메모리 누수(Memory Leak)
- 비정상적인 응답시간 감소(Stom Drain)
- 사용자 지정

\* APM 툴의 경우 모니터링은 가능하지만 WAS를 직접 제어할 수는 없음)



장애를 사전에 방지

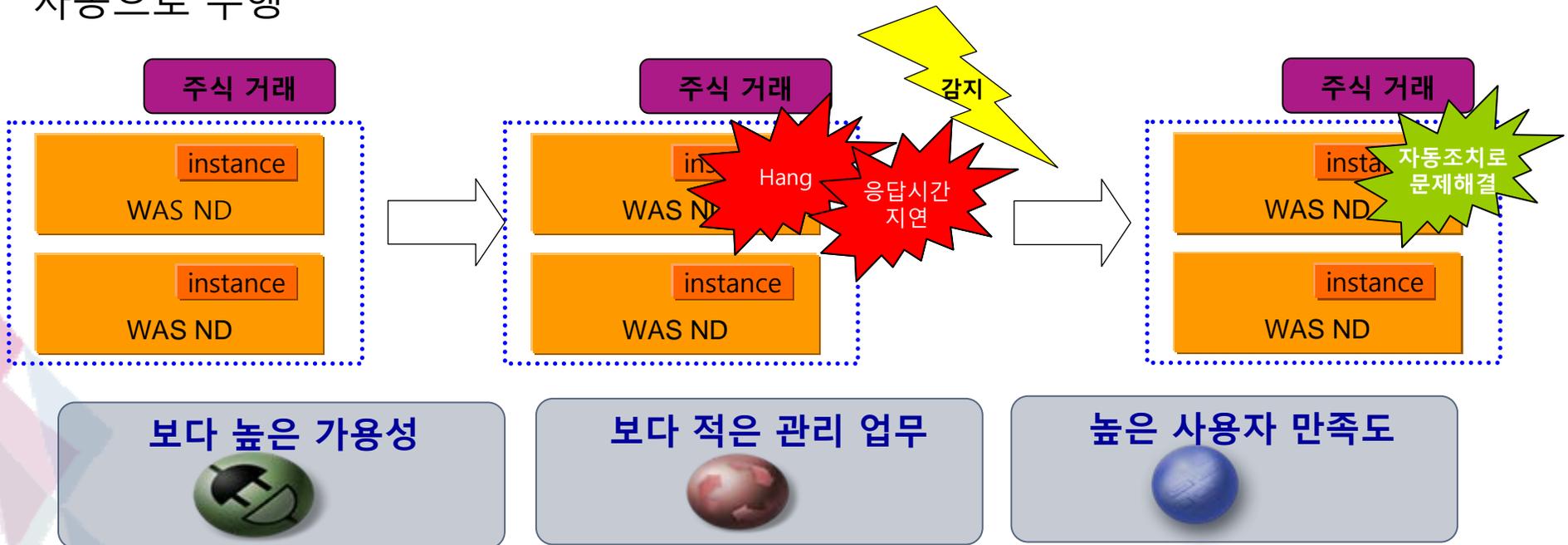
정확한 원인 규명 가능



# 애플리케이션 서버 자가관리(Health Management)

장애가 발생한 후에 조치하는 것이 아니라 실시간으로 개별 서버들을 모니터링한 후 이상 동작을 감지했을 때 지정된 조치를 취하여 사전에 장애를 방지

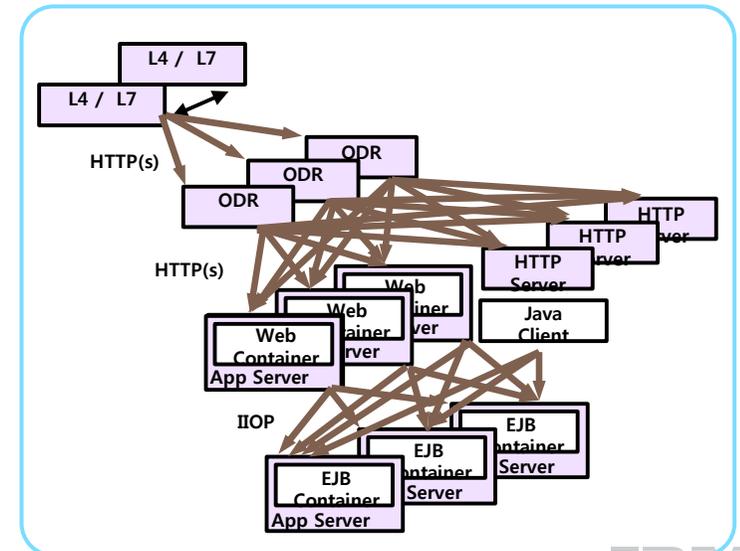
- WAS는 다양한 이유로 인하여 장애가 발생할 수 있음
- 장애를 근본적으로 해결하는 것도 중요하지만 일시적으로 회피하거나 임시적인 자동 복구 기능은 안정적인 운영환경에 필수
- 다양한 조건에 의해 모니터링하여 장애 상황 예측 시 WAS에 미리 지정한 조치들을 자동으로 수행



# 동적 워크로드 관리

클러스터 내의 모든 서버 및 컴포넌트에 대해 가중치 기반의 부하분산을 수행하며, 각 가중치는 수동으로 관리자가 지정하거나 WAS가 자동으로 결정

- 수동 가중치 지정 방식
  - 관리자가 직접 관리 콘솔을 통해 서버 별 부하분산 가중치 지정
  - 서버의 상황을 고려하거나 실시간 반영이 어려움
- 자동 가중치 지정 방식
  - WebSphere의 Workload Manager Module이 다음과 같은 조건을 실시간으로 측정하여 서버 별 부하분산 가중치를 자동으로 최적화
    - 서버의 고유 용량
    - 서버 별 응답 시간
    - 서버 별 메모리 사용량
    - 서버 별 CPU 사용량
    - 서버 별 동시 사용자 수

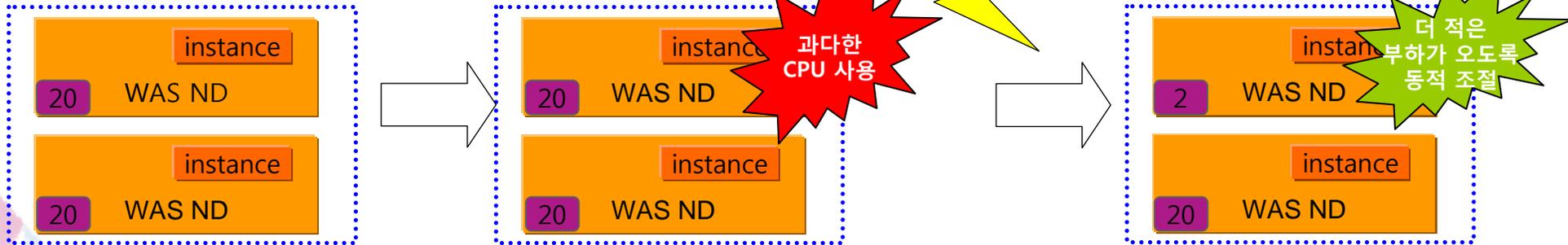


# 동적 워크로드 관리

동적 워크로드 관리를 통한 실시간 자동 가중치 지정 방식은 수동 가중치 지정방식에 비해 운영 환경에서 다양한 장점을 가지고 있음

- 서버의 고유용량, CPU, 메모리, tpmc 등을 고려하여 직접 계산하여 가중치를 줄 필요가 없음 (관리자가 직접 작업해야 할 관리 포인트 최소화)
- 서버의 상태에 따라 적절하게 가중치가 조절되므로 수동 가중치 지정방식에 비해 보다 나은 성능을 보여줌
- 장애 상황이나 예기치 않은 돌발 상황의 경우 자동으로 상황을 고려하여 가중치를 변경하므로 보다 안정적인 서비스를 제공

X : 서버 가중치



보다 높은 가용성



보다 적은 관리 업무



높은 사용자 만족도



# 애플리케이션 에디션 관리

하나의 버전에서 다른 버전으로 서비스 중단 없이 애플리케이션 자동 업데이트

## ■ 애플리케이션 버전 관리 기능

- 애플리케이션 배치 시 버전 번호를 부여
- 보관 가능한 버전의 수에는 제한 없음
- 관리 콘솔에서의 조작만으로 원하는 버전으로 업데이트 및 복구 가능
- 업데이트 및 복구는 운영 도중, 서비스 중단 없이 수행 가능

## ■ 관리 모드를 이용하여 무중단 배포 정책 수립

- 관리 모드를 사용하게 되면 해당 서버로 더 이상 요청을 보내지 않음
- 요청을 받지 않는 상태로 서버가 살아있으므로 장애를 고립시키거나 애플리케이션 재배포, WAS 서버의 재시작이 가능
- 위의 기능을 이용하면 클러스터링 환경에서 운영 중에 장애 없이 무중단 배포 정책 수립 가능

Application=StockTrade  
Preferences

Edition Control

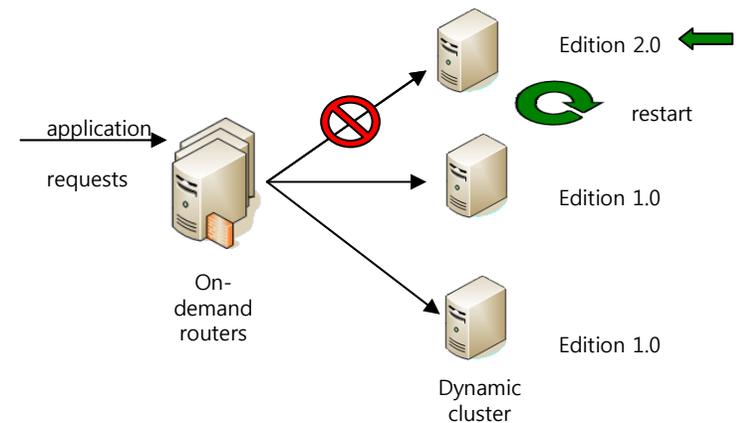
Select	Editions	Description	Target	State
<input type="checkbox"/>	1.0	1.0	WebSphere:cell=localhostCell01,cluster=TestCluster	INACTIVE
<input type="checkbox"/>	2.0	2.0	WebSphere:cell=localhostCell01,cluster=TestCluster	ACTIVE
<input type="checkbox"/>	Base edition	No description available	WebSphere:cell=localhostCell01,cluster=TestCluster	INACTIVE

Total 3

Preferences

Select	Edition	Description	Target	State	Status
<input type="checkbox"/>	Base	Base Edition	ProductionDC1	Inactive	↻
<input type="checkbox"/>	1.0	Generation 2 prototype	StaticTestCluster-Server1	Inactive	↻
<input type="checkbox"/>	2.0	Generation 2	ProductionDC1	Active	➔
<input type="checkbox"/>	3.0	Project "Blue Diamond"	ProductionDC1-Validation	Validation	➔

Total 4

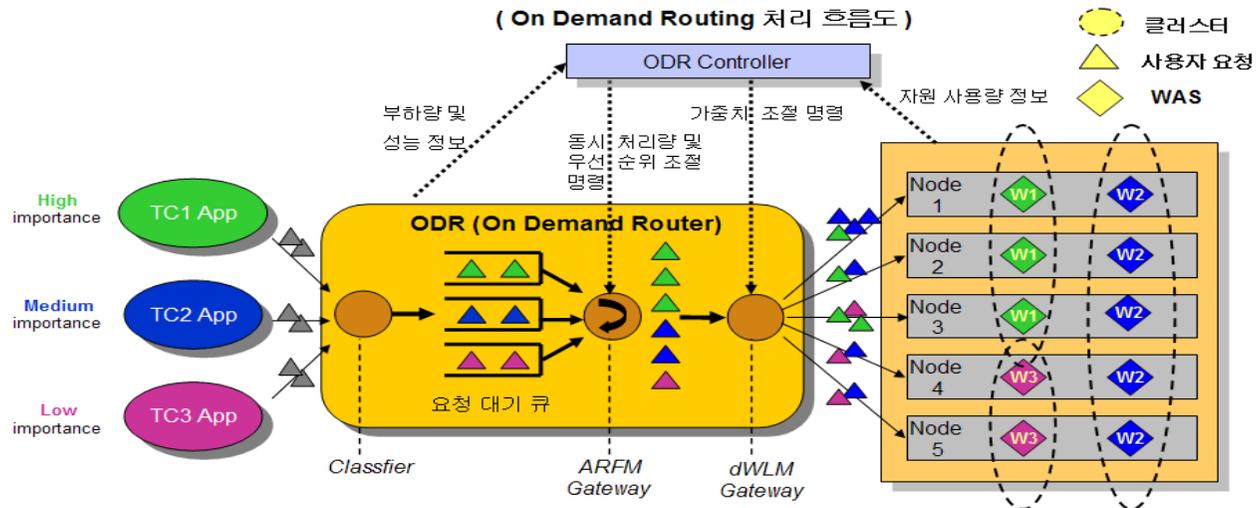


# 지능화된 라우팅

애플리케이션을 인지할 수 있으며 업무의 우선 순위에 따라 서비스 품질을 자동으로 관리하여 비즈니스 결과 향상을 지원 가능

## ▪ 목표치 및 우선 순위 기반의 서비스 품질 관리

- 트랜잭션 클래스 별 목표치 및 중요도를 지정하여 서비스 품질(QoS)을 실시간으로 관리
- 동일한 애플리케이션도 사용자마다 서로 다른 목표치 및 중요도로 서비스 품질 관리
- 해당 기능은 리소스 부족과 경합 시에 효과를 발휘하므로 평상시 서비스에서는 문제 발생을 야기하지 않음
- ODR(On Demand Router)이 지능적인 라우팅의 포컬 포인트
- 부하 변화, 자원 사용량, 성능 정보 등을 실시간으로 추적, 분석하여 사용자 요청을 지능적으로 분배, 조정하여 서비스 품질을 관리



# 메모리 누수(Memory Leak) 감지 및 방지

운영 시에 애플리케이션 코드의 의심되는 패턴 감시를 통해서 메모리 누수 감지나 방지, 조치를 제공할 수 있으며 이를 통해 보다 높은 가용성을 제공

- 애플리케이션을 중지하거나 재배포할 때 메모리 누수를 방지하기 위한 자동 방안 제공
- 애플리케이션이나 개별 모듈이 중지될 때 활동을 모니터링하여 진단 조치 수행
- 메모리 누수를 방지/경고를 제공하거나 Heap Dump, System Dump를 생성할 수 있는 기능 제공
- 메모리 누수가 발생하는 중단된 애플리케이션 리스트 MBean 제공

애플리케이션의  
메모리 누수  
가능성을 감소

충분한 정보를 취합  
누수가 감지되면  
애플리케이션 수정 지원

메모리 누수가 발생하는  
중단된 애플리케이션 리스  
트 제공

Table 1. *com.ibm.ws.runtime.component.MemoryLeakConfig.detectAppCLLeaks*

Information	Value
Name	com.ibm.ws.runtime.component.MemoryLeakConfig.detectAppCLLeaks
Description	When the server is shutting down or an application stops, WebSphere Application Server determines the classloaders that have leaked and issues warnings and other additional information that aids in debugging the memory leak. See also the <a href="#">Improved classloader leak detection</a> PMR.
Default	false

Table 2. *com.ibm.ws.runtime.component.MemoryLeakConfig.clearAppCLLeaks*

Information	Value
Name	com.ibm.ws.runtime.component.MemoryLeakConfig.clearAppCLLeaks
Description	Enable proactive classloader leak mediation and fixing. When this property is set to true, WebSphere Application Server mediates on behalf of the application to remedy any classloader leaks that are detected.
Default	false

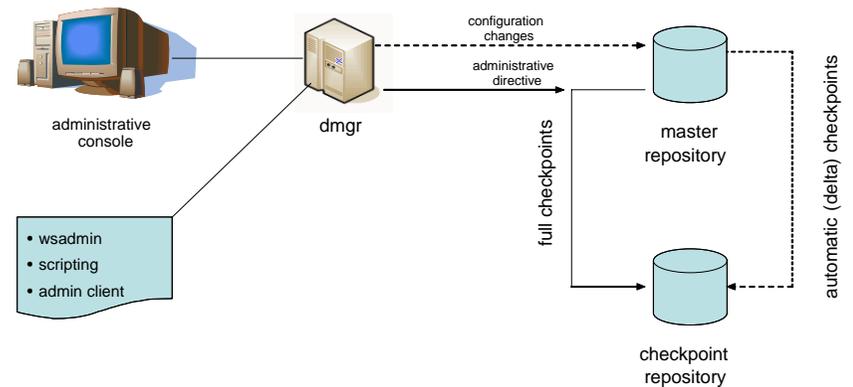
# 설정 변경 실시간 백업/복구

관리적으로 관리 콘솔에서 전체 설정 저장의 복사본을 저장할 뿐만 아니라 자동으로 실시간 백업을 보존할 수 있어서 이를 통해 필요 시 즉시 되돌리기 가능

## ▪ 변경사항이 생길 때마다 자동적으로 설정이 저장됨

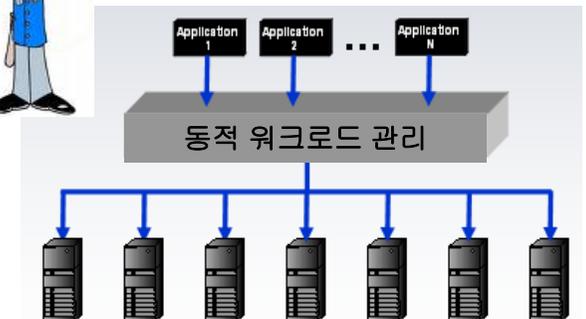
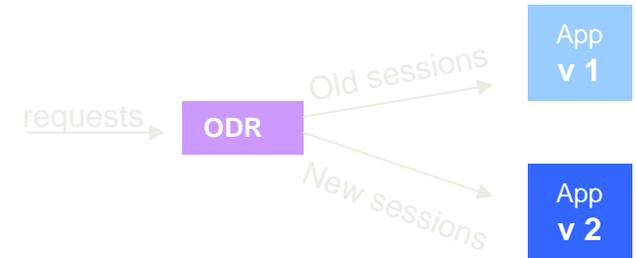
- 개별 설정 변경에 따라 파일의 백업 복사본으로 이루어진 체크포인트 변화
- **설정 상의 이슈가 발생했을 때 즉시 되돌리기 기능이 사용 가능**
- 설정 변경 관리 및 감사(Audit) 용도로 활용 가능
- 파일로 저장 설정을 남길 뿐만 아니라 저장된 설정의 각각의 상태를 관리 콘솔에서 직접적으로 확인 및 제어 가능

Select	Name	Documents	Type	Sequence	Description
<input type="checkbox"/>	<a href="#">Delta-1241111972367</a>	1	DELTA	1241111972367	Autosave delta image
<input type="checkbox"/>	<a href="#">Delta-1241112253617</a>	1	DELTA	1241112253617	Autosave delta image
<input type="checkbox"/>	<a href="#">Delta-1241112254744</a>	1	DELTA	1241112254744	Autosave delta image
<input type="checkbox"/>	<a href="#">tesCheck</a>	352	FULL	1241111957635	20090501
<input type="checkbox"/>	<a href="#">testCheck2</a>	353	FULL	1241112268240	20090501
Total 5					



# 요약 - 지능화된 관리의 이점

- 중단 없는 애플리케이션 업그레이드
  - 애플리케이션 서버를 위한 애플리케이션 버전 관리
- 장애를 피하기 위한 사전 조치
  - 애플리케이션 서버 자가관리(Health Management)
- 지능적인 라우팅
  - 동적 워크로드 관리
- 복원력 강화를 통한 가용성 향상



# 새로운 IBM WAS V8.5

기존 WAS와 같지만 또 다른 새로운 차세대 WAS인 IBM WAS V8.5

## 개발자 경험



빠르고, 유연하고,  
단순화된  
애플리케이션 개발

- 아주 가벼운 WAS 서버인 Liberty Profile 제공
- 확장된 툴링과 WAS 툴링 번들
- JDK7 지원
- OSGi 프로그래밍 모델 강화
- OSGi 애플리케이션에서 EJB 지원
- 마이그레이션 툴킷
- 웹 2.0 & 모바일 툴킷: IBM Worklight 통합
- SCA OASIS 프로그래밍 모델

## 애플리케이션 복원력



지능적인 관리  
& 강화된  
복원력

- 애플리케이션 에디션 관리
- 애플리케이션 서버 자가관리(Health Management)
- 새로운 지능적 라우팅 능력
- 메시징 인프라 복원력
- 동적 클러스터링
- WAS에서 메모리 누수(Memory Leak) 감지 및 방지

## 작업과 제어



작업과 보안, 제어,  
통합을 향상

- 선택적인 JDK 활용
- WebSphere 배치(Batch) 강화
- 관리자 보안 감사
- OSGi Blueprint 보안 향상
- Cross Component Trace (XCT)
- IBM Support Assistant 강화
- 보다 향상된 로그와 추적 필터링

# 최신 JDK 7.0 지원 및 선택적으로 사용 가능

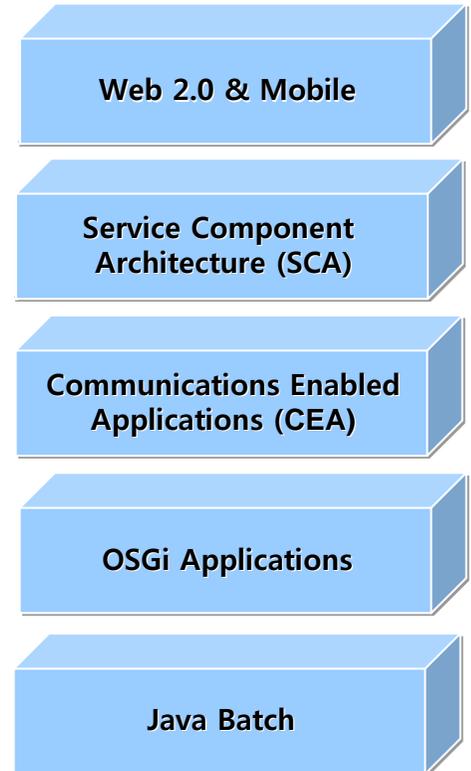
가장 최신의 JDK 7.0 을 지원하며 개발과 운영환경에서 상황에 따라 가장 적합한 JDK 를 선택해서 사용 가능 (JDK 6 또는 7)

- WAS V8.5에서 선택적인 JDK 적용을 새롭게 소개
  - Java 6를 지속적으로 사용하면서 일부 환경에서는 Java 7을 사용 가능
  - Java 6를 유지하면서 일부 부분집합은 Java 7으로 토폴로지 구성 가능
  - 필요할 경우 Java 7과 Java 6를 스위치해서 사용 가능
- Java 7의 장점
  - 호환성 – 이전 버전 프로그램은 변화 없이 Java 7에서 수행 가능
  - 생산성 – 반복적인 코드 사용을 줄이고 베스트 코딩 프랙티스 사용 촉진
  - String을 바로 switch 문에서 사용 가능, Binary literal(0b101010), multi-catch 사용 가능
  - Automatic Resource Management – 명시적인 close 없이도 Scope를 벗어나면 자동으로 자원 반환
  - 성능 – 새로운 concurrency API – asynchronous I/O API
  - 보편성 – JVM에서 동적 Language의 성능 향상
  - 통합성 – Java 7에서 새롭게 유연한 파일 시스템 API 포함(JSR203)

# Java EE 6.0 및 다양한 프로그래밍 모델 지원

최신의 표준 자바 개발 환경(Java EE 6)을 지원하며, 다양한 통합 표준기반 프로그래밍 모델을 함께 제공

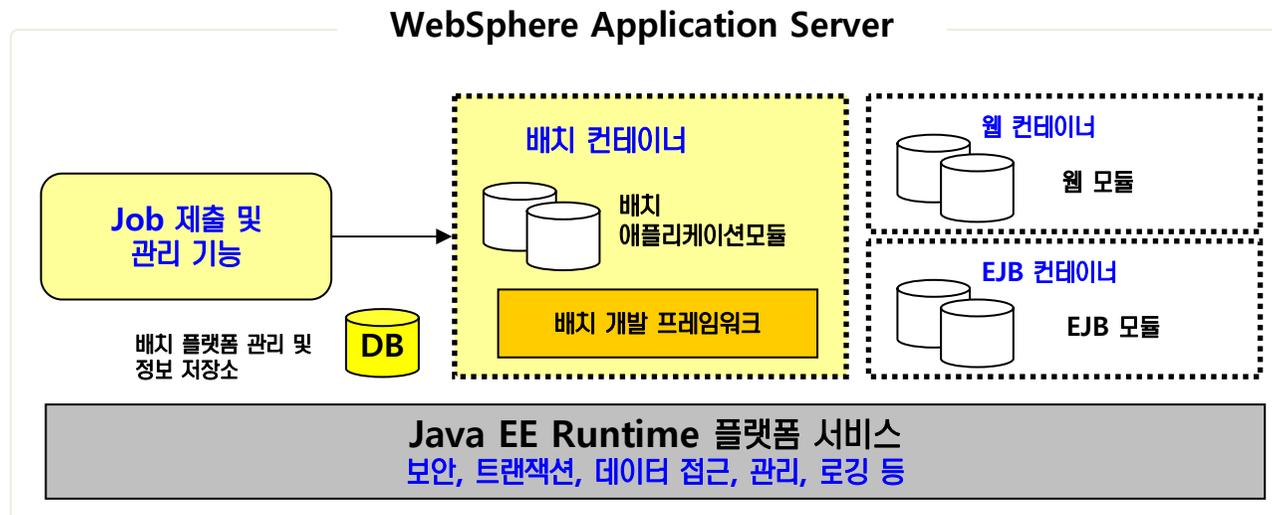
- Enterprise JavaBeans (EJB) 3.1
  - 애플리케이션 서버 외부에서 테스트, 새로운 타이머 지원, 비동기 지원 강화, WAR 파일 안에 패키지 가능 등을 포함하여 단순화를 통한 향상된 개발 생산성 제공
- Contexts and Dependency Injection for Java (CDI) 1.0
  - 웹과 비즈니스 로직 계층의 단순화되고 강화된 연계를 통하여 보다 빠른 가치 제공 가능
- JPA(Java Persistence API) 2.0
  - 동적 쿼리 구조, 향상된 락킹과 매핑 지원을 통한 개발자들의 사용성 및 성능 향상
- Java Servlet 3.0
  - SIP 과 COMET 과 같은 새로운 비동기 프로토콜 지원, 어노테이션 지원, 3rd-Party 프리젠테이션 프레임워크와의 쉬운 통합을 통한 강화된 가치 제공



# Java Batch

배치만을 위하여 별도의 인프라와 운영 비용을 들일 필요 없이, OLTP 와 통합된 환경을 제공하여 배치 애플리케이션의 빠른 개발 및 배포 가능

- 더 낮은 TCO: 공유되는 비즈니스 로직과 인프라를 활용하여 배치와 OLTP 워크로드를 동시에 처리
- 강화된 개발 생산성: 배치 라이프사이클 관리를 위해 사전 통합된 애플리케이션 프레임워크, Java 배치 프로그래밍 모델과 도구
- 자동화와 관리: 보안, 안정성, 고가용성 등을 위하여 추가된 체크포인트, 재시작 기능과 같은 컨테이너 관리 서비스이며, OLTP 애플리케이션과 배치 작업을 통합관리
- 병렬처리: 병렬 배치 작업의 파티셔닝과 배정, 관리와 모니터링을 위한 인프라 제공



# IBM WAS V8.5 오퍼링

## IBM WAS V8.5 오퍼링 – Liberty Profile 및 Intelligent Management

<p><b>WAS for Developers</b></p> <p>Tools Edition +Liberty Profile</p> <p>운영환경 WAS 위에서 구동될 수 있는 혁신적인 애플리케이션들을 효율적으로 개발할 수 있도록 지원</p> <p>Eclipse Adapter를 포함하여 개발자 PC에서는 무상 사용 가능</p>	<p><b>WAS Hypervisor Edition</b></p> <p>+Liberty Profile +Intelligent Mgmt</p> <p>즉시 VMware와 같은 가상화된 환경에서 수행될 수 있도록 최적화</p>	<p><b>WAS ND</b></p> <p>Tools Edition +Liberty Profile +Intelligent Mgmt</p> <p>미션 크리티컬한 애플리케이션에 대한 고가용성 및 고도화된 성능, 관리 능력을 제공 가능</p>	<p><b>WAS for z/OS</b></p> <p>+Liberty Profile +Intelligent Mgmt</p> <p>z/OS Sysplex를 완전히 활용하여 높은 보안성, 안정성, 자원 효율성 등 제공</p>
<p><b>WAS</b></p> <p>Tools Edition +Liberty Profile</p> <p>최대 5개의 애플리케이션 서버 프로파일에 대하여 웹 티어 클러스터링과 페일오버를 포함하여 적당한 사이즈 업무에 대한 보안, 고성능 트랜잭션 엔진을 제공</p>			
<p><b>WAS Express</b></p> <p>+Liberty Profile</p> <p>동적 웹 사이트나 애플리케이션을 구축하기 위한 낮은 비용의 준비된 솔루션</p>		<p><b>WAS CE</b></p> <p>사전 투자 비용이 필요 없는 오픈 소스 기반의 엔진 제공</p>	

공통 코드 기반으로 개발

# 목차

- 시장의 트렌드
- WAS V8.5
  - Liberty Profile을 통한 개발자 경험 향상
  - 애플리케이션 복원력
  - 작업과 제어 향상

## ■ 요약

# 차이가 명확하게 보이나요?



## WAS라고 다 똑같은 WAS가 더 이상 아닙니다

- IBM WAS V8.5는 개발자들을 위한 **경량화된 IBM WAS Liberty Profile**을 제공합니다. 약 50MB 정도의 하드디스크만 차지하며, 구동시간이 5초도 걸리지 않는 아주 가벼운 WAS입니다.
- 애플리케이션 관리 기능을 통해 가동시간 **중단 없이 애플리케이션 업그레이드**가 가능합니다.
- 자가 보호 및 치유 기능을 통하여 **미리 문제를 감지하고 조치**를 취할 수 있습니다.
- WAS의 상태를 체크하여 최고의 성능과 가용성을 낼 수 있도록 **지능적으로 요청을 동적 분배**할 수 있습니다.



감사합니다

