

Oracle9i Application Server의 기본적인 관리 방안

글 ||장성우| 한국오라클 BD본부 WAS팀| sungwoo.chang@oracle.com

웹 애플리케이션 서버 위에서 동작하는 다양한 애플리케이션들을 관리하기 위해서는 효율적이고 편리한 관리 도구들이 있어야 한다. 본 장에서는 **Oracle9iAS**에서 제공하는 관리 도구들을 통해 **Oracle9iAS** 인스턴스를 관리하는 다양한 방법에 대해서 소개한다.

Oracle9iAS 인스턴스 관리 방안

Oracle9iAS를 구동시키는 형태에는 크게 다음과 같은 3가지의 방법이 있다 :

- 1 **OC4J만을 구동시켜 활용하는 방법** : 가장 간단한 형태의 구동 방법으로서 **OC4J** 상에 **J2EE** 애플리케이션만을 수행시키고자 할 때 많이 사용하는 방법이다. 가장 사용이 단순하며 또 응답 성능도 가장 빠른 경향이 있다.
- 1 **OHS+OC4J를 함께 구동시키는 방법** : 웹 서버와 애플리케이션 서버를 동시에 구동시켜 일단 모든 요청을 웹 서버가 접수하고 프로그램 실행에 대한 요청만 다시 애플리케이션 서버로 전달하여 해당 애플리케이션을 실행하고 그 결과를 다시 웹 서버를 통해 사용자 브라우저로 전달하는 형태의 수행을 지원하기 위한 구동 방법이다.
- 1 **Infrastructure를 통해 OHS+OC4J를 구동하는 방법(클러스터 사용의 경우)** : 다중 클러스터나 포탈, SSO 등을 사용하는 경우 리파지토리를 통해 해당 콘텐츠와 제어 정보를 관리하는데 이를 활용하기 위해서는 먼저 **metadata repository**를 구동시킨 후에 **OHS, OC4J**를 구동시켜야 한다.

이 각각의 경우에 대해서 실제 관리 방법에 대해서 살펴 보기로 한다.

Standalone OC4J의 관리 방법

OC4J 시작하기

OC4J를 실행하기 위해서는 해당 파일인 **oc4j.jar** 파일을 실행시켜 주기만 하면 된다 :

Java - jar oc4j.jar

하지만, 좀 더 편리하게 사용하기 위해서는 다음과 같이 **startoc4j.sh**를 만들어서 실행시켜 주면 좋다. 이때, **nohup{시작시각}.out**은 **OC4J**가 콘솔에 뿌리는 메시지를 담는 로그 파일로서 **j2ee/home/log** 디렉토리에 두도록 한다.

```
% cat startoc4j.sh
```

```
#!/usr/bin/sh
OC4J_START_TIME=`date +%Y%m%d_%H%M%S`
OC4J_NOHUP_LOG=nohup$OC4J_START_TIME.out
nohup java -jar oc4j.jar > log/$OC4J_NOHUP_LOG &
```

OC4J 종료하기

OC4J의 종료는 **admin.jar**를 다음과 같이 실행시켜서 수행한다 :

```
java -jar admin.jar ormi://localhost:23791 admin welcome - stop
```

또는 **ps**로 **java** 프로세스를 찾아서 **`kill - 9`** 명령으로 종료해 주어도 된다.

새 애플리케이션 등록 하기

이제 **OC4J** 위에 실행시키고 싶은 애플리케이션으로서 **foo.ear**이 있다고 하자. 그리고 이 애플리케이션이 다음과 같은 기본 가정을 가지고 있다고 하자.

- 애플리케이션이 **fooApp**이라는 이름을 가진다
- **foo.ear**에 포함된 웹 모듈 이름이 **foo-web**이다.
- 브라우저에서 **/fooroot** 라는 **url** 패턴을 가지고 들어온다.
- **RMI** 포트가 **23791** 이고(**rmi.xml**) **admin**의 비밀번호가 **welcome** 이다.

이 애플리케이션을 디플로이(애플리케이션 서버에 원하는 애플리케이션을 로딩시켜 실행시킬 수 있도록 준비하는 행위를 말함)하기 위해서는 다음 두 명령을 순차적으로 실행해 주면 된다 :

(1) **java -jar admin.jar ormi://localhost:23791 admin welcome - deploy - file foo.ear - deploymentName fooApp**

(2) **java -jar admin.jar ormi://localhost:23791 admin welcome - bindWebApp fooApp foo-web default-web-site /fooroot**

OHS+OC4J 구동 방법

OHS와 **OC4J**가 함께 수행되는 경우에는 **OC4J**를 개별적으로 실행시킬 수 없다. 이 경우에는 '**dcmctl**'이라고 하는 **command utility**를 통해서 인스턴스의 시작과 중지 작업을 실행해 주어야 한다.

Oracle9iAS의 구동을 위해서는 먼저 '**dcmctl**'을 이용하여 웹 서버(**OHS**)와 애플리케이션 서버(**OC4J**)를 구동시켜야 한다. 구동시키는 방법은 다음과 같다 :

시작하기

1. OHS 구동

dcmctl start - ct ohs - v - d

2. OC4 구동

dcmctl start - co OC4J - v - d

내리기

1. OC4J 중지

dcmctl stop - co OC4J_DAS - v - d

2. OHS 중지

dcmctl stop - ct ohs - v - d

Infrastructure를 통한 OHS+OC4J 구동 방법

Single Sign-on, 포탈, 클러스터링 등의 높은 수준의 기능을 활용하기 위해서는 **infrastructure**라고 불리는 **metadata repository**가 함께 실행되어야 한다. **Infrastructure** 안에는 기본적인 **cluster configuration** 정보 뿐만 아니라 **LDAP directory**, **SSO information**, 포탈에서 관리하는 메타 정보와 콘텐츠 등이 저장, 관리되어 지게 된다.

Infrastructure를 포함하는 **Oracle9iAS**의 관리 방법은 다음과 같다 :

시작하기

1. Metadata repository

- 데이터베이스 리스너를 띄운다: **lsnrctl start**
- **Metadata repository** 데이터베이스를 띄운다: **sqlplus / as sysdba → startup**

2. OID

- **OID monitor**를 띄운다: **oidmon start**

- **ldapd**를 띄운다: **oidctl connect=iasdb server=oidldapd instance=1 start**
'**ps - ef |grep oid**' 명령을 통해 다음과 같이 세 개의 프로세스가 보여야 한다.

```
oidldapd - p 389 - i 1 - conf 0 - sport 636 - sslenable 2 key=xxxxx
oidldapd
oidmon start
```

3. EM 서버

- **DISPLAY** 변수를 설정한다: **export DISPLAY=127.0.0.1:0.0**
- **EM** 서버를 띄운다: **emctl start**

4. HTTP

```
dcmctl start - ct ohs - v - d
```

5. OC4J_DAS

```
dcmctl start - co OC4J_DAS - v - d
```

내리기

1. OC4J_DAS

```
dcmctl stop - co OC4J_DAS - v - d
```

2. HTTP

```
dcmctl stop - ct ohs - v - d
```

3. EM 서버

```
emctl stop
```

4. OID

- **ldapd**를 내린다: **oidctl connect=iasdb server=oidldapd instance=1 stop**
이 작업은 시간이 다소 걸린다. '**ps - ef |grep oid**' 로 보았을때 **oidldapd** 프로세스 두 개가 모두 내려가야 한다.
- **oidmon**을 내린다: **oidmon stop**
반드시 **ldapd**가 내려간 후 실행한다.

5. Metadata repository

- **Metadata repository** 데이터베이스를 내린다: **sqlplus sys/<password> as sysdba → shutdown immediate**
- 데이터베이스 리스너를 내린다: **lsnrctl stop**

Oracle9iAS 관리 시 주의할 점

보통 숙련된 관리자가 아니라면 **Oracle9iAS** 운용 과정에서 다음과 같은 실수를 저지르기 쉽다 :

- 1 한 기계에 **Oracle9iAS Instance**를 여러 개 설치할 경우에는 반드시 같은 **OS** 사용자의 다른 **ORACLE_HOME**에 설치하여야 하는데 이를 무시하고 설치할 가능성이 있다.
- 2 **dcmctl** 명령어 수행시 **ORACLE_HOME**을 확인하지 않고 하는 경우 다른 인스턴스의 **dcmctl**이 실행될 수 있다.
- 3 클러스터에 **Join**된 인스턴스 환경 설정 변경이나 **.ear** 디플로이 작업 시 반드시 클러스터에 속한 모든 인스턴스의 **EM** 서버가 떠 있는지 확인하고 작업을 해야 하는데 이를 간과하기 쉽다.
- 4 운용시에는 **/var/tmp**, **/tmp** 공간을 사용한다. 특히 '**dcmctl deployApplication [redeployApplication]**' 명령을 사용할 때는 이곳에 여유 공간을 충분히 둔다.(최소 **20M** 이상)

Oracle9iAS 인스턴스가 매뉴얼대로 작동하지 않는 경우는 위에서 열거한 바와 같이 비숙련된 관리자의 실수에서 비롯된 것이 대부분이다. 아래의 내용 가운데에는 **Oracle9iAS Administrator's Guide**의 내용과 맞지 않는 부분이 있는데 이는 관리자의 실수로 **Oracle9iAS** 인스턴스가 원하는 동작을 하지 않을 경우 비상시에 사용할 수 있는 팁들이다.

1. dcmctl 명령은 반드시 -v -d 옵션을 뒤에 붙여 사용한다.

2. dcmctl 명령이 timeout exception을 발생시키는 것은 대부분의 경우에 무해하며, 다만 명령의 성공 여부를 알기 위해 'dcmctl getReturnStatus -v -d' 혹은 'dcmctl getState -v -d'를 통해 확인한다.

3. OC4J 환경 설정을 변경 하기 전에는 반드시 'dcmctl saveInstance ...' 명령으로 백업을 받아둔다.

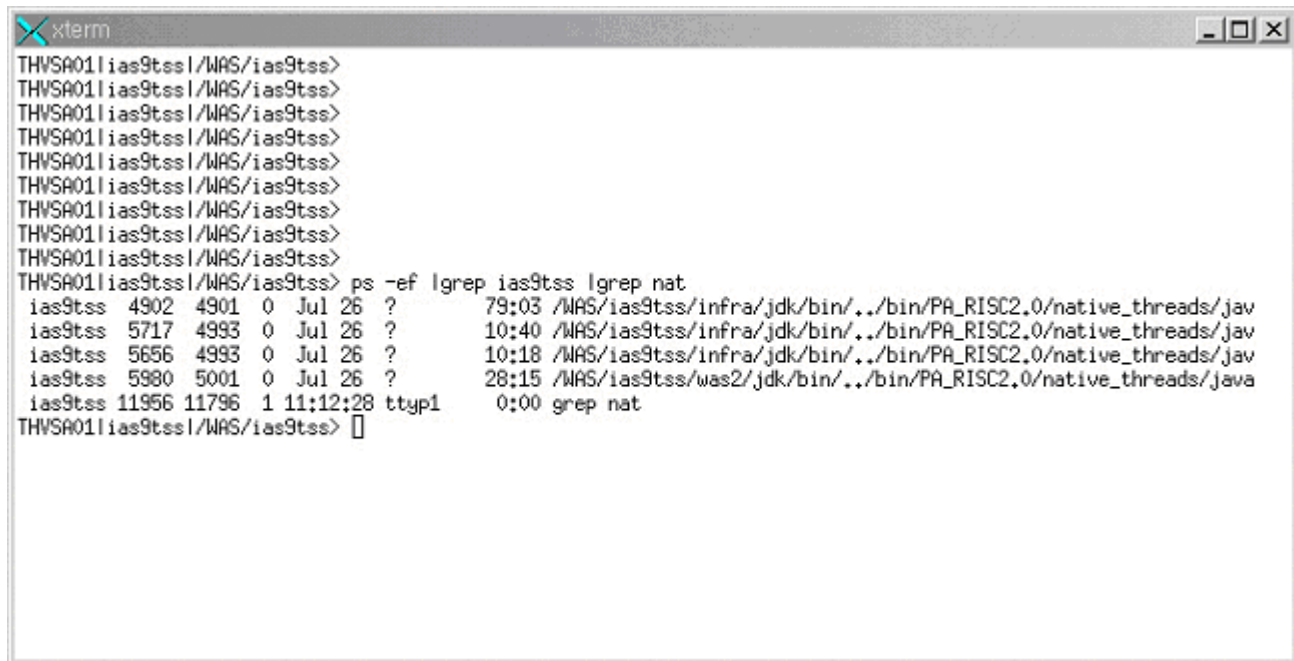
OC4J의 실행에 필요한 환경 변수들의 값은 repository에 저장되어지게 된다. 이 값을 변경시키기 전에 반드시 'dcmctl saveInstance -dir full- directory- pathname- to- store- repository' 명령을 통해 이전 repository 값을 백업 받아 놓는다. 이는 환경 변수 변경시 사용자의 실수로 인한 변수 설정 오작동 등의 경우에 백업 받아 놓았던 repository를 이용하여 원래 상태로 복구할 수 있도록 지원하기 위해서이다.

4. EM 서버가 동작하는지 확인하는 방법

'netstat -na |grep 1810' 혹은 'ps -u <OS user> |grep perl'

5. 자바 프로세스를 찾는 방법

dcmctl을 통해 OC4J 인스턴스를 띄울 경우 java 프로세스의 실행 경로가 매우 길어져서 'ps -ef |grep java'로는 찾아지지 않는 경우가 많다. dcmctl이 띄우는 java 프로세스의 실행 경로에는 native_threads가 포함되어 있으므로 'ps -ef |grep nat' 명령으로 자바 프로세스를 찾을 수 있다.



```

xterm
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss>
THVSA01|ias9tss|/WAS/ias9tss> ps -ef |grep ias9tss |grep nat
ias9tss 4902 4901 0 Jul 26 ? 79:03 /WAS/ias9tss/infra/jdk/bin/./bin/PA_RISC2.0/native_threads/jav
ias9tss 5717 4993 0 Jul 26 ? 10:40 /WAS/ias9tss/infra/jdk/bin/./bin/PA_RISC2.0/native_threads/jav
ias9tss 5656 4993 0 Jul 26 ? 10:18 /WAS/ias9tss/infra/jdk/bin/./bin/PA_RISC2.0/native_threads/jav
ias9tss 5980 5001 0 Jul 26 ? 28:15 /WAS/ias9tss/was2/jdk/bin/./bin/PA_RISC2.0/native_threads/java
ias9tss 11956 11796 1 11:12:28 ttyt1 0:00 grep nat
THVSA01|ias9tss|/WAS/ias9tss>
  
```