
제 7 장 WebSphere Product Center 서비스 관리

서비스 유형

완전한 WebSphere Product Center 시스템은 동시에 실행 중인 다음 서비스로 구성됩니다.

admin	관리 서버는 원격 시스템에서 모듈을 시작/중지합니다.
appsvr	Application Server는 JSP(Java Server Page)를 제공합니다.
eventprocessor	이벤트 프로세서는 모든 모듈 간에 이벤트를 디스패치합니다.
queuemanager	대기열 관리자는 WebSphere Product Center 외부로 문서를 전송합니다.
scheduler	스케줄러는 백그라운드 작업을 실행합니다.
workflow	워크플로우 엔진

admin_properties.xml 및 클러스터링

워크스테이션의 클러스터에서 서비스를 실행할 수 있습니다. 클러스터의 다른 시스템은 admin_properties.xml 파일에 정의되어 있습니다.

\$TOP/etc/default/admin_properties.xml

참고: 추가 정보는 admin_properties.xml에 제공됩니다. admin_properties.xml 파일에 나열된 모든 시스템에서 각 서비스를 실행할 수 있습니다.

일반 WebSphere Product Center 클러스터에는 Application Server, WebSphere Product Center 서버의 지원 RMI 레지스트리 유틸리티 및 보조 서버의 나머지 WebSphere Product Center 구성요소가 포함될 수 있습니다.

기본 서버를 실패복구할 경우, 보조 서버에서 이전에 실행하지 않았던 서비스는 중단 시간을 최소화하여 최소의 노력으로 온라인 서비스를 다시 제공할 수 있습니다.

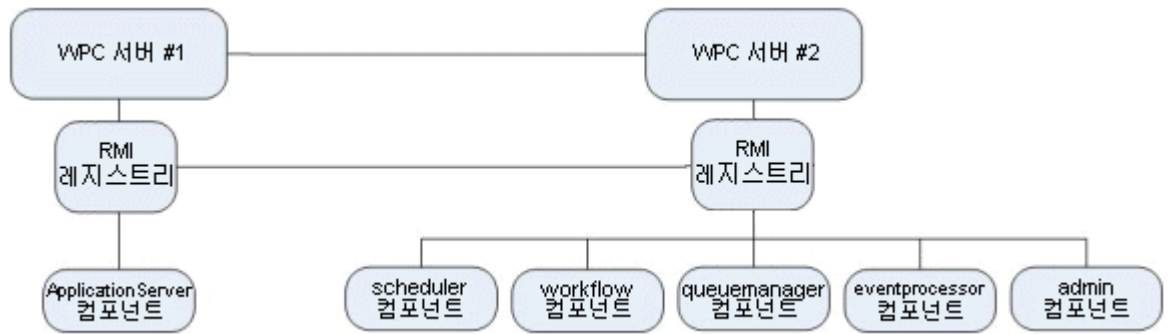


그림 4 - 일반 WebSphere Product Center 클러스터

서비스 이름 - 자세한 이름 및 축약 이름

각 서비스는 서비스 이름으로 고유하게 식별됩니다. 서비스 이름은 고유해야 합니다. (서비스는 동일한 이름의 다른 서비스가 클러스터의 시스템에서 실행 중인 경우 시작을 거절합니다.)

각 서비스는 서비스 이름이 다른 경우 여러 시스템에서 실행될 수 있습니다.

'admin' 및 'appsrvr' 서비스의 이름은 시스템에서 수정합니다.

admin의 경우 admin_<machine name>(예: 'admin_server1')

appsrvr의 경우 appsrvr_<machine name>(예: 'appsrvr_server1')

다른 서비스의 경우 임의의 이름을 선택하십시오. 선택한 이름은 실제로는 서비스의 축약 이름입니다.

내부적으로 자세한 이름은 이 축약 이름을 사용하여 빌드됩니다.

rmi://<machine name>:<rmi port>/<db user name>/<service type>/<service short name>

예/:

'server1' 시스템의 'scheduler' 서비스를 실행, rmi 포트 17507 사용, 데이터베이스 사용자 'pauadm'에 연결, 'sch1' 서비스 이름을 지정 중이면 자세한 이름은 다음과 같습니다.

rmi://server1:17507/pauadm/scheduler/sch1

또 다른 스케줄러(sch2)가 동일한 사용자 및 포트의 서버 2에서 실행되면, 자세한 이름은 다음과 같습니다.

rmi://server2:17507/pauadm/scheduler/sch2

서비스 유형의 메모리 플래그 설정

WebSphere Product Center 서비스의 메모리 플래그는 WebSphere Product Center 설치 디렉토리에 있는 WebSphere Product Center 초기화 스크립트에 설정되어 있습니다.

<설치 위치>/setup/init_ccd_vars.sh

WebSphere Product Center 서비스에 다음 메모리 플래그 설정을 사용할 것을 권장합니다.

```
export ADMIN_MEMORY_FLAG='-Xmx64m -Xms48m'
```

```
export APPSVR_MEMORY_FLAG='-Xmx512m -Xms64m'
```

```
export EVENTPROCESSOR_MEMORY_FLAG='-Xmx64m -Xms48m'
```

```
export QUEUEMANAGER_MEMORY_FLAG='-Xmx64m -Xms48m'
```

```
export SCHEDULER_MEMORY_FLAG='-Xmx1024m -Xms48m'
```

```
export WORKFLOWENGINE_MEMORY_FLAG='-Xmx64m -Xms48m'
```

RMI – 원격 메소드 호출

서비스 등록은 RMI(Java Remote Method Invocation)를 통해 수행됩니다. 서비스를 실행하기 전에 RMI가 시스템에서 시작되었는지 확인하십시오.

RMI 상태

클러스터에서 실행 중인 모든 서비스 목록을 확보하려면, 다음 스크립트를 실행하십시오.

```
$TOP/bin/go/rmi_status.sh
```

이 스크립트는 클러스터의 모든 시스템에서 RMI 데몬과 연결하여 각 시스템에서 로컬 서비스 목록을 확보합니다. 자세한 이름 목록을 리턴합니다.

로그 파일

각 서비스는 런타임 로그 파일을 생성합니다.

```
$TOP/logs/<service>/<서비스 이름>/svc.out
```

예/:

스케줄러 이름 'sch1'은 \$TOP/logs/scheduler/sch1에서 런타임 로그 파일 svc.out을 생성합니다.

서비스를 시작한 후에는 로그 파일을 점검하여 모든 것이 문제점없이 시작되었는지 확인할 것을 권장합니다.

서비스 시작

다음 절에서는 로컬 스크립트를 사용하여 서비스를 제어하는 방법을 설명합니다. 서비스를 사용하기 전에 RMI 레지스트리는 서비스를 사용하여 시스템에서 시작되어야 합니다.

RMI를 시작하려면, 스크립트를 실행하십시오.

```
$TOP/bin/go/start/start_rmiregistry.sh
```

로컬 시스템에서 서비스 시작

로컬 시스템에서 서비스를 시작하는 가장 간단한 방법은 \$TOP/bin/go/start/ 디렉토리에서 스크립트를 사용하는 것입니다.

스크립트	설명
start_admin.sh	관리 서비스를 시작합니다.
start_appsvr.sh	Application Server를 시작합니다.
start_eventprocessor.sh	이벤트 프로세서를 시작합니다.
start_queuemanager.sh	대기열 관리자를 시작합니다.
start_rmiregistry.sh	RMI 레지스트리를 시작합니다.
start_scheduler.sh	스케줄러를 시작합니다.
start_workflowengine.sh	워크플로우 엔진을 시작합니다.

이러한 스크립트 각각(start_admin.sh, start_appsvr.sh 및 start_rmiregistry.sh 제외)은 선택적 인수로 서비스 이름을 가져올 수 있습니다.

```
-svc_name=<service name>
```

admin 및 appsvr 서비스는 기본 이름을 사용합니다(admin_<machine name> 및 appsvr_<machine name>). 다른 이름을 지정하면 효과가 없습니다.

서비스 이름이 지정되지 않으면 기본 이름을 사용합니다.

스케줄러의 경우 "scheduler"

이벤트 프로세서의 경우 "eventprocessor"

대기열 관리자의 경우 "queuemanager"

워크플로우 엔진의 경우 "workflow"

참고: 이미 실행 중인 로컬 서비스의 이름을 사용하여 로컬 서비스가 시작된 경우, 이전 로컬 서비스가 먼저 중단됩니다. 따라서 서비스를 '다시 시작'하는 데에도 스크립트가 사용될 수 있습니다(먼저 중단한 후 다시 시

작).

예/:

"sch1" 이름으로 스케줄러를 시작하려면 다음을 수행하십시오.

```
$STOP/bin/go/start/start_scheduler.sh -svc_name=sch1
```

기본 이름으로 스케줄러를 시작하려면 다음을 수행하십시오.

```
$STOP/bin/go/start/start_scheduler.sh
```

서비스 중단

서비스를 중단하면 서비스가 종료되며 사용 불가능하게 됩니다.

예를 들어, 스케줄러가 작업을 실행 중이면, 작업은 프로세스 중간에 중단됩니다.

로컬 시스템에서 서비스 중단

여기의 구조는 해당 시작 구조를 반영합니다.

\$STOP/bin/go/abort/ 디렉토리에서 스크립트를 사용하십시오.

스크립트	설명
abort_admin.sh	관리 서비스를 중단합니다.
abort_appsvr.sh	Application Server를 중단합니다.
abort_eventprocessor.sh	이벤트 프로세서를 중단합니다.
abort_queuemanager.sh	대기열 관리자를 중단합니다.
abort_rmiregistry.sh	RMI 레지스트리를 중단합니다.
abort_scheduler.sh	스케줄러를 중단합니다.
abort_workflowengine.sh	워크플로우 엔진을 중단합니다.

이러한 스크립트 각각(abort_admin.sh, abort_appsvr.sh 및 abort_rmiregistry.sh 제외)은 선택적 인수로 서비스 이름을 가져올 수 있습니다.

```
-svc_name=<service name>
```

참고: RMI를 중단하면 원격 시스템에서 서비스에 연결할 수 없습니다.

서비스 중지

서비스를 중지하면 순조롭게 종료하기 위한 서비스가 요청됩니다. 서비스가 "블록화"되면, 시스템 종료 프로시저를 전혀 실행하지 않을 수도 있습니다. 스케줄러는 현재 실행 중인 모든 작업 실행을 종료할 때까지 중지되지 않습니다.

로컬 시스템에서 서비스 중지

여기의 구조는 시작 구조를 반영합니다.

\$STOP/bin/go/stop/ 디렉토리에서 스크립트를 사용하십시오.

스크립트	설명
stop_admin.sh	관리 서비스를 중지합니다.
stop_appsvr.sh	Application Server를 중지합니다.
stop_eventprocessor.sh	이벤트 프로세서를 중지합니다.
stop_queuemanager.sh	대기열 관리자를 중지합니다.
stop_scheduler.sh	스케줄러를 중지합니다.
stop_workflowengine.sh	워크플로우 엔진을 중지합니다.

이러한 스크립트 각각(abort_admin.sh, abort_appsvr.sh 및 abort_rmiregistry.sh 제외)은 선택적 인수로 서비스 이름을 가져올 수 있습니다.

-svc_name=<service name>

중단 및 중지에 관한 중요한 참고

중지와 중단 중 어느 것을 사용해야 합니까?

중단	서비스가 종료될 것임을 보증하나 현재 실행 중인 타스크가 인터럽트되지 않을 것임을 보증할 수는 없습니다.
중지	서비스가 중지된 "경우" 현재 실행 중인 모든 타스크가 먼저 중지된 후 순조롭게 중지될 것임을 보증합니다.

모든 WebSphere Product Center 모듈 시작

로컬 시스템에서 WebSphere Product Center 시작

\$STOP/bin/go/start/start_local.sh 스크립트를 실행하십시오.

이렇게 하면 RMI 레지스트리와 다음 서비스도 시작됩니다.

- 관리 'admin_<machine name>'
- Application Server 'appsvr_<machine name>'
- 이벤트 프로세서 'eventprocessor'
- 대기열 관리자 'queuemanager'
- 스케줄러 'scheduler'
- 워크플로우 'workflow'

참고: 시작하기 전에 로컬 시스템에서 기존 시스템을 종료(kill)시키십시오.

로컬 시스템에서 WebSphere Product Center 중단

\$STOP/bin/go/abort/abort_local.sh 스크립트를 실행하십시오.

로컬 시스템에서 시작된 모든 서비스가 중단됩니다. RMI 레지스트리가 중단됩니다.

로컬 시스템에서 WebSphere Product Center 중지

\$STOP/bin/go/stop/stop_local.sh 스크립트를 실행하십시오.

로컬 시스템에서 시작된 모든 서비스가 중지됩니다. 기본적으로 RMI 레지스트리가 다른 서비스로 중지됩니다. 실행 중인 RMI 레지스트리를 유지하려면, 다음 옵션을 전달하십시오.

--kill_rmi=no

참고: "kill_rmi=no" 앞에 두 개의 대시 기호가 있습니다.

서비스 상태

축약형 서비스 상태 확보

짧은 상태의 서비스를 가져오려면, 다음 매개변수를 전달하십시오.

-cmd=check -svc=<service name>

예:

스케줄러 상태를 가져오려면 다음을 실행하십시오.

rootadmin.sh -cmd=check -svc=scheduler

짧은 상태는 다음 중 하나일 수 있습니다.

실행 중	서비스가 실행되고 있으며 "하트 비트" 기능에 응답 중입니다.
찾을 수 없음	서비스를 찾을 수 없습니다. 서비스가 시작되지 않았거나 기능을 중지했을 수 있습니다.
찾았으나 응답하지 않음	RMI 레지스트리에 등록 중인 서비스를 찾았으나 "하트 비트" 기능에 응답하지 않습니다. 서비스를 다시 시작해야 할 수 있습니다.

세부형 서비스 상태 확보

세부형 서비스 상태를 가져오려면, 다음 매개변수를 전달하십시오.

-cmd=status -svc=<service name>

아무 브라우저를 사용하여 볼 수 있는 **html** 파일이 생성됩니다. 터미널에서 **lynx**를 사용하여 출력을 형식화하고자 할 수 있습니다.

예/:

스케줄러 상태를 가져오려면 다음을 실행하십시오.

```
rootadmin.sh -cmd=status -svc=scheduler > /tmp/sch_status.html;  
lynx /tmp/sch_status.html
```

또는

```
rootadmin.sh -cmd=status -svc=scheduler > /tmp/sch_status.html;  
lynx -dump /tmp/sch_status.html
```

참고: 위의 예에서 사용된 ">"는 상태 세부사항을 파일 출력 위치에 지정합니다.

상태는 서비스에서 실행 중인 다른 스레드의 개요와 서비스에서 현재 가져온 데이터베이스 연결 상태도 제공합니다.