CA Application Performance Management

ChangeDetector 사용자 안내서

릴리스 9.5

포함된 도움말 시스템 및 전자적으로 배포된 매체를 포함하는 이 문서(이하 "문서")는 정보 제공의 목적으로만 제공되며 CA 에 의해 언제든 변경 또는 취소될 수 있습니다.

CA 의 사전 서면 동의 없이 본건 문서의 전체 혹은 일부를 복사, 전송, 재생, 공개, 수정 또는 복제할 수 없습니다. 이 문서는 CA 의 기밀 및 독점 정보이며, 귀하는 이 문서를 공개하거나 다음에 의해 허용된 경우를 제외한 다른 용도로 사용할 수 없습니다: (i) 귀하가 이 문서와 관련된 CA 소프트웨어를 사용함에 있어 귀하와 CA 사이에 별도 동의가 있는 경우, 또는 (ii) 귀하와 CA 사이에 별도 기밀 유지 동의가 있는 경우.

상기 사항에도 불구하고, 본건 문서에 기술된 라이선스가 있는 사용자는 귀하 및 귀하 직원들의 해당 소프트웨어와 관련된 내부적인 사용을 위해 합당한 수의 문서 복사본을 인쇄 또는 제작할 수 있습니다. 단, 이 경우 각 복사본에는 전체 CA 저작권 정보와 범례가 첨부되어야 합니다.

본건 문서의 사본 인쇄 또는 제작 권한은 해당 소프트웨어의 라이선스가 전체 효력을 가지고 유효한 상태를 유지하는 기간으로 제한됩니다. 어떤 사유로 인해 라이선스가 종료되는 경우, 귀하는 서면으로 문서의 전체 또는 일부 복사본이 CA 에 반환되거나 파기되었음을 입증할 책임이 있습니다.

CA 는 관련법의 허용 범위 내에서, 상품성에 대한 묵시적 보증, 특정 목적에 대한 적합성 또는 권리 위반 보호를 비롯하여(이에 제한되지 않음) 어떤 종류의 보증 없이 본 문서를 "있는 그대로" 제공합니다. CA 는 본 시스템의 사용으로 인해 발생되는 직, 간접 손실이나 손해(수익의 손실, 사업 중단, 영업권 또는 데이터 손실 포함)에 대해서는 (상기 손실이나 손해에 대해 사전에 명시적으로 통지를 받은 경우라 하더라도) 귀하나 제 3 자에게 책임을 지지 않습니다.

본건 문서에 언급된 모든 소프트웨어 제품의 사용 조건은 해당 라이선스 계약을 따르며 어떠한 경우에도 이 문서에서 언급된 조건에 의해 라이선스 계약이 수정되지 않습니다.

본 문서는 CA 에서 제작되었습니다.

본 시스템은 "제한적 권리"와 함께 제공됩니다. 미합중국 정부에 의한 사용, 복제 또는 공개는 연방조달규정(FAR) 제 12.212 조, 제 52.227-14 조, 제 52.227-19(c)(1)호 - 제(2)호 및 국방연방구매규정(DFARS) 제 252.227-7014(b)(3)호 또는 해당하는 경우 후속 조항에 명시된 제한 사항을 따릅니다.

Copyright © 2013 CA. All rights reserved. 본 시스템에서 언급된 모든 상표, 상호, 서비스 표시 및 로고는 각 해당 회사의 소유입니다.

CA Technologies 제품 참조

이 문서에서는 다음과 같은 CA Technologies 제품과 기능을 참조합니다.

- CA Application Performance Management(CA APM)
- CA Application Performance Management ChangeDetector(CA APM ChangeDetector)
- CA Application Performance Management ErrorDetector(CA APM ErrorDetector)
- CA Application Performance Management for CA Database Performance(CA APM for CA Database Performance)
- CA Application Performance Management for CA SiteMinder®(CA APM for CA SiteMinder®)
- CA Application Performance Management for CA SiteMinder®Application Server Agents(CA APM for CA SiteMinder® ASA)
- CA Application Performance Management for IBM CICS Transaction Gateway(CA APM for IBM CICS Transaction Gateway)
- CA Application Performance Management for IBM WebSphere Application Server(CA APM for IBM WebSphere Application Server)
- CA Application Performance Management for IBM WebSphere Distributed Environments(CA APM for IBM WebSphere Distributed Environments)
- CA Application Performance Management for IBM WebSphere MQ(CA APM for IBM WebSphere MQ)
- CA Application Performance Management for IBM WebSphere Portal(CA APM for IBM WebSphere Portal)
- CA Application Performance Management for IBM WebSphere Process Server(CA APM for IBM WebSphere Process Server)
- CA Application Performance Management for IBM z/OS®(CA APM for IBM z/OS®)
- CA Application Performance Management for Microsoft SharePoint(CA APM for Microsoft SharePoint)
- CA Application Performance Management for Oracle Databases(CA APM for Oracle Databases)
- CA Application Performance Management for Oracle Service Bus(CA APM for Oracle Service Bus)

- CA Application Performance Management for Oracle WebLogic Portal(CA APM for Oracle WebLogic Portal)
- CA Application Performance Management for Oracle WebLogic Server(CA APM for Oracle WebLogic Server)
- CA Application Performance Management for SOA(CA APM for SOA)
- CA Application Performance Management for TIBCO BusinessWorks(CA APM for TIBCO BusinessWorks)
- CA Application Performance Management for TIBCO Enterprise Message Service(CA APM for TIBCO Enterprise Message Service)
- CA Application Performance Management for Web Servers(CA APM for Web Servers)
- CA Application Performance Management for webMethods Broker(CA APM for webMethods Broker)
- CA Application Performance Management for webMethods Integration Server(CA APM for webMethods Integration Server)
- CA Application Performance Management Integration for CA CMDB(CA APM Integration for CA CMDB)
- CA Application Performance Management Integration for CA NSM(CA APM Integration for CA NSM)
- CA Application Performance Management LeakHunter(CA APM LeakHunter)
- CA Application Performance Management Transaction Generator(CA APM TG)
- CA Cross-Enterprise Application Performance Management
- CA Customer Experience Manager(CA CEM)
- CA Embedded Entitlements Manager(CA EEM)
- CA eHealth® Performance Manager(CA eHealth)
- CA Insight™ Database Performance Monitor for DB2 for z/OS®
- CA Introscope®
- CA SiteMinder®
- CA Spectrum® Infrastructure Manager(CA Spectrum)
- CA SYSVIEW® Performance Management(CA SYSVIEW)

CA 에 문의

기술 지원팀에 문의

온라인 기술 지원 및 지사 목록, 기본 서비스 시간, 전화 번호에 대해서는 http://www.ca.com/worldwide에서 기술 지원팀에 문의하십시오.

목차

제 1 장: CA Application Performance Management ChangeDetector 개요

CA Application Performance Management ChangeDetector 를 사용하면 CA Introscope 가 응용 프로그램 파일 및 구성의 변경 사항을 모니터링하고 보고할 수 있습니다. CA APM ChangeDetector 로 프로덕션 웹 응용 프로그램의 변경 사항을 감지할 수 있으므로 웹 응용 프로그램 성능 문제의 근본 원인이 변경으로 인한 것인지 확인하는 데 큰 도움이 됩니다. 변경이 문제의 원인이라는 것을 확인했으면 CA APM ChangeDetector 를 사용하여 문제를 진단할 수 있습니다. CA APM ChangeDetector 에는 성능 문제의 원인으로 의심되는 코드, 응용 프로그램 서버 구성 및 연결된 시스템 구성의 변경 사항을 시각화하는 기능이 있습니다.

응용 프로그램의 매우 다양한 부분에서 변경이 발생하기 때문에, 이에 대처할 수 있도록 Environment Performance Agent(EPAgent)에서도 CA APM ChangeDetector 가 지원됩니다. EPAgent 를 사용하면 거의 모든 정보 출처에서 응용 프로그램 성능 정보를 수집하고 사용자 환경과 관련된 변경 데이터를 모니터링할 수 있습니다. EPAgent 에 대한 자세한 내용은 *CA APM Environment Performance Agent 구현 안내서*를 참조하십시오.

이 장에서는 CA APM ChangeDetector 에 대해 소개하며, 이 구성 요소를 사용자 CA Introscope 환경에 통합하는 방법과 함께 일반적인 사용 시나리오에 대해 설명합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

안내서 정보

이 안내서에는 CA APM ChangeDetector 그래픽 사용자 인터페이스 구성 요소의 설치, 구성, 배포 및 사용에 대한 정보가 포함되어 있습니다.

따로 언급되어 있지 않는 한 이러한 정보는 모든 플랫폼에 적용되는 것으로 간주할 수 있습니다. 예를 들어 .NET 플랫폼만 대상으로 하는 단원에는 아래와 같은 참고 사항이 명시됩니다.

참고: 이 단원의 내용은 .NET 플랫폼에만 적용됩니다.

이 안내서에 사용되는 디렉터리 명명 규칙

이 안내서에서는 설치 디렉터리에 대해 다음과 같은 명명 규칙을 사용합니다.

규칙:

<EM_Home>

의미:

Enterprise Manager 가 설치된 디렉터리로 대개 Program Files 디렉터리 아래에 있습니다.

규칙:

<Workstation_Home>

의미:

Workstation 이 설치된 디렉터리로 대개 Program Files 디렉터리 아래에 있습니다.

규칙:

<Agent_Home>

의미:

CA Introscope 에이전트가 설치된 디렉터리로 대개 CA APM Agent 디렉터리에 해당합니다.

규칙:

<ProductName_Home>

의미:

타사 제품이나 응용 프로그램의 설치 디렉터리입니다. 예를 들어 WebLogic 을 사용하는 경우 응용 프로그램 서버의 설치 디렉터리입니다.

규칙:

FileName<VersionNumber><Operating System or other identifier>.FileType

의미:

구체적인 식별 정보를 포함하는 파일 이름입니다.

예를 들어 Unix 운영 체제에서 CA APM ChangeDetector 8.1 의 tar 패키지에서 파일을 추출하는 경우 다음 파일을 다운로드하게 됩니다.
ChangeDetector8.1.0.0.unix.tar

이 경우 본 안내서에서는 해당 파일을 다음과 같이 표기합니다.
ChangeDetector<VersionNumber>.unix.tar

CA APM ChangeDetector 정보

CA APM ChangeDetector 는 응용 프로그램 환경의 변경 사항을 모니터링하는 데 사용할 수 있는 CA Introscope 확장 기능의 집합입니다. CA APM ChangeDetector 는 CA Introscope 에 직접 통합되어 오버헤드가 낮은 실시간 변경 감지 기능을 제공합니다. 프로덕션 환경에서 문제가 발생한 경우 CA Introscope 사용자는 CA APM ChangeDetector 를 사용하여 응용 프로그램 변경 사항과 응용 프로그램 성능의 변경 사항을 상호 연관시켜 변경이 문제의 원인인지 식별할 수 있습니다.

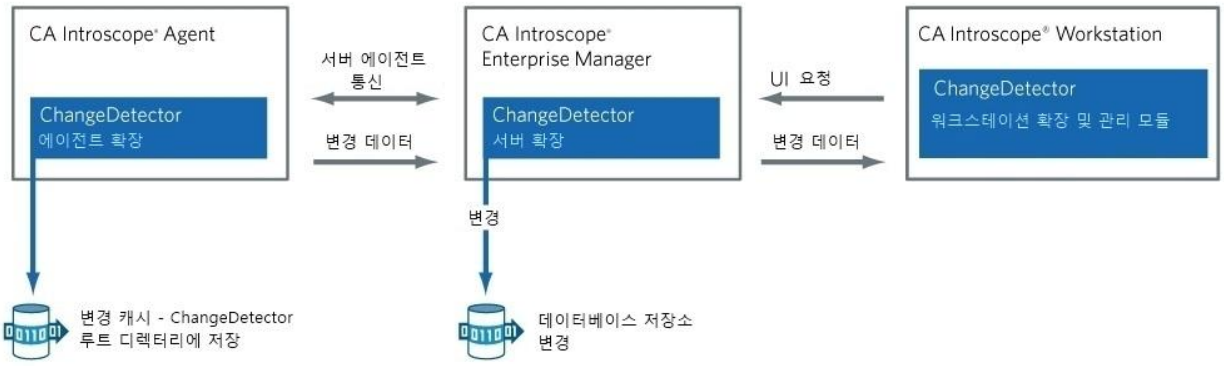
CA APM ChangeDetector 에는 에이전트에서 여러 기간에 걸쳐 응용 프로그램 코드, 구성 및 환경의 차이점을 보고하는 기능이 있으므로 특정 기간 동안 응용 프로그램의 한 인스턴스가 어떻게 달라졌는지 확인할 수 있습니다. 예를 들어 오늘과 어제 또는 오늘과 지난 주 사이의 응용 프로그램 차이점을 확인할 수 있습니다.

CA APM ChangeDetector 를 설치하면 사용할 수 있는 기능:

- 파일(텍스트 및 바이너리), 아카이브, 시스템 속성, 응용 프로그램 코드, 데이터베이스 테이블 및 특정 데이터베이스 쿼리 결과 집합을 포함한 변경 이벤트의 그래픽 뷰
- 변경 사항이 감지된 시간, 변경의 특성, 텍스트 파일 버전 간의 차이점 등을 비롯한 변경 이벤트 세부 정보
- 변경 데이터를 종합적으로 보여 주는 CA APM ChangeDetector 대시보드
- 변경 사항의 계층적 뷰와 기록 뷰(오늘, 어제, 지난 주)
- 응용 프로그램의 변경 사항을 요약하여 보여 주는 변경 보고서

CA APM ChangeDetector 와 사용자의 CA Introscope 환경

다음 그림에서는 CA APM ChangeDetector 구성 요소가 CA Introscope 와 상호 작용하는 방식을 보여 줍니다.



CA APM ChangeDetector 사용 시나리오

다음 시나리오에서는 CA APM ChangeDetector 를 사용하여 변경 사항을 감지하고, 변경 사항과 문제를 상호 연관시키고, 변경된 내용을 확인하고, 문제 해결 방법을 결정하는 과정을 설명합니다.

변경이 문제의 원인인지 확인: 우울한 월요일 아침

월요일 아침 9 시에 은행의 온라인 뱅킹 응용 프로그램에서 성능 문제가 발생하기 시작했습니다. 로그인을 비롯한 다른 중요 트랜잭션 응답 시간이 길어져 SLA 수준을 유지하지 못한다는 CA Introscope 경고가 IT 팀에 전달됩니다. 성난 사용자들은 고객 서비스 센터에 전화를 걸어 은행 계좌에 액세스할 수 없다는 불만을 토로하기 시작합니다. 상담 센터 관리자는 문제를 응용 프로그램 지원 그룹으로 승격시켜 즉시 조사에 착수하도록 만듭니다.

"무엇이 바뀐 것인가요? 지난 주 내내 응용 프로그램은 아무 문제 없이 작동했습니다."

응용 프로그램 지원 팀은 CA Introscope 대시보드에서 성능 저하를 검토합니다. 먼저 성능 그래프를 살펴봅니다. CA APM ChangeDetector 가 설치되어 있으므로 CA APM ChangeDetector 차트에 직접 통합된 형태로 변경 데이터를 확인할 수 있습니다. 지원 팀은 차트에 제공된 변경 주석과 세부 정보를 검토하여 현재의 성능 문제에 앞서 지난 주말 동안 일련의 변경이 이루어진 것을 알게 됩니다. 보다 상세한 조사를 통해 응용 프로그램의 유지 관리 기간인 토요일 밤에 변경이 발생한 것을 확인합니다. 지원 팀은 CA Introscope Investigator 의 트리 뷰를 클릭하여 다양한 응용 프로그램 인스턴스의 성능 그래프를 검토합니다. 유사한 여러 그래프를 통해 동일한 시간대에 각 인스턴스에서 유사한 변경 사항이 발생했다는 것이 명확해집니다.

변경 사항으로 인해 문제가 발생한 것으로 보고 응용 프로그램 지원 팀은 더욱 심도 있는 조사를 합니다. CA Introscope Investigator 에서 응용 프로그램의 각 구성 요소를 클릭하면 CA APM ChangeDetector 뷰에 각 구성 요소에서 발생한 변경 사항이 표시됩니다. 응용 프로그램 파일을 추적하여 구성 파일 3 개를 포함한 37 개 파일이 변경되었다는 사실을 알게 됩니다. 지원 팀은 CA APM ChangeDetector 를 사용하여 각 변경 사항이 수정에 따른 것이며 유지 관리 기간 동안 감지되었음을 확인합니다. 응용 프로그램 서버 구성 파일 중 하나를 선택하여 서버의 현재 버전과 변경 전에 존재한 파일의 이전 버전을 비교합니다.

그러자 원인이 명확해집니다. 100 으로 설정되어 있던 데이터베이스 연결 풀 매개 변수가 10 으로 변경되어 있습니다. 이 정보를 가지고 응용 프로그램 지원 관리자는 개발 관리자에게 전화를 걸어 이것이 입력 실수라는 것과 성능 저하가 데이터베이스 연결의 예기치 않은 변경 때문이라는 것을 확인합니다.

문제가 되기 전에 변경 사항 감지: 현장에서 포착

CA Introscope 경고가 나타납니다. 응용 프로그램 지원 팀에게 Java 응용 프로그램에서 변경 사항이 발생했다는 정보가 전달됩니다. 예정된 변경이 없었기 때문에(예정된 유지 관리 기간이 아님) 응용 프로그램 지원 관리자는 성능 그래프 조사를 시작합니다. 응용 프로그램은 현재 정상적으로 작동하지만 그래프는 몇 분 전에 몇 가지 변경 사항이 발생했음을 보여 줍니다. 커서로 변경 사항을 가리키면 변경 세부 정보가 표시됩니다. 시스템 변수가 수정된 것으로 나타납니다.

응용 프로그램 지원 관리자는 CA Introscope Investigator 탐색을 시작하여 다양한 응용 프로그램 구성 요소의 변경 사항을 조사합니다. CA APM ChangeDetector 에서 응용 프로그램의 변경 사항이 트리 뷰로 나타납니다. 변경 사항을 조사하면서 응용 프로그램 지원 관리자는 응용 프로그램 및 구성 파일이나 APM 데이터베이스 구성에서 발생한 변경 사항이 없다는 것을 알게 됩니다. 대신 이전 CA Introscope 경고를 확인하면서 JVM 및 응용 프로그램 서버의 4 가지 시스템 속성이 변경된 것을 알게 됩니다.

응용 프로그램 지원 관리자는 감지된 변수 변경 사항을 각각 클릭하여 해당 변수 이름, 변경이 감지된 시간, 변경된 값을 표시합니다. Java 힙 크기 변수가 줄어들어 응용 프로그램의 로드가 증가하면 문제가 될 수 있는 것으로 나타납니다. UNIX 시스템 관리자와 함께 점검하여 관리 업무가 수행되었고 변경 사항의 잠재적 효과가 실현되지 않았음을 확인합니다. 변경 사항을 원래 값으로 되돌려 잠재적 성능 문제를 방지합니다.

제 2 장: CA APM ChangeDetector 설치 및 구성

이 장에서는 구성 마법사를 사용하거나 수동으로 XML 구성 파일을 편집하여 CA APM ChangeDetector 데이터 원본을 설치하고 구성하는 방법에 대해 설명합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

CA APM ChangeDetector 설치 및 설정

CA APM ChangeDetector 는 에이전트를 설치하는 동안 설치되어, 사용되도록 설정됩니다. CA APM ChangeDetector 는 독립 실행형 설치 관리자를 사용하거나 자동 모드 설치 옵션을 사용하여 설치할 수 있습니다. CA APM ChangeDetector 를 설치한 경우 구성이 필요합니다.

다음 단계를 수행하십시오.

1. 에이전트를 설치합니다.

   에이전트 설치에 대한 자세한 내용은 *CA APM .Net 에이전트 구현 안내서* 또는 *CA APM Java Agent 구현 안내서*를 참조하십시오.CA APM ChangeDetector 는 기본적으로 사용되도록 설정됩니다.

a. *<EM_Home>/config* 디렉터리에 있는 *IntroscopeEnterpriseManager.properties* 파일을 열고 다음 속성을 *false* 로 설정한 후 Enterprise Manager 를 다시 시작하면 CA APM ChangeDetector 를 사용하도록 설정할 수 있습니다.

```
introscope.changeDetector.disable=false
```

중요! AIX 환경에서 WebSphere 6.1 또는 7.0 응용 프로그램 및 Oracle DB 를 기반으로 CA APM ChangeDetector 를 실행하는 경우 다음과 유사한 예기치 않은 예외가 발생할 수 있습니다.

java.security.AccessControlException: Access denied (java.net.SocketPermission hostname:port connect,resolve) at java.security.AccessController.checkPermission(AccessController.java:104)

이 예외를 방지하려면 Java 보안 권한을 허용하도록 *<WebSphere_Home>/properties/server.policy* 를 수정하거나 WebSphere Admin 콘솔의 "Enable application security"(응용 프로그램 보안 사용) 확인란에 대한 선택을 취소하십시오.

2. CA APM ChangeDetector 를 구성합니다.

CA APM ChangeDetector 를 구성하기 전에 다양한 구성 옵션을 살펴보고 CA APM ChangeDetector 를 구성하기 전에 (페이지 21)를 참조하십시오.

참고: CA APM ChangeDetector 는 Windows(cdnativefile.dll) 및 Linux(libcdnativefile.so) 운영 체제에 대해 변경된 데이터와 함께 파일 소유자 데이터를 보고하는 기본 라이브러리를 설치합니다. 지원되는 플랫폼 버전 목록은 *CA APM Compatibility Guide(CA APM 호환성 안내서)*를 참조하십시오.

CA APM ChangeDetector 를 구성하기 전에

CA APM ChangeDetector 를 구성하기 전에 다양한 구성 옵션을 살펴보고 자신의 환경에 가장 적합한 방법을 선택해야 합니다.

- CA APM ChangeDetector 구성 마법사 사용 (페이지 23) - 설정을 구성할 때 나타나는 일련의 안내 페이지에서 정보를 입력하여 CA APM ChangeDetector 를 구성할 수 있습니다.
- CA APM ChangeDetector 구성 파일 수정 (페이지 34) - 이 방법은 모든 플랫폼에서 사용할 수 있습니다. 이 방법에서는 XML 파일을 수동으로 편집하여 CA APM ChangeDetector 를 구성할 수 있습니다. CA APM ChangeDetector 패키지에는 이 방법을 선택한 경우 참조할 수 있는 샘플 XML 파일이 들어 있습니다.

다음과 같은 개념에도 익숙해져야 합니다.

- 여러 CA APM ChangeDetector 구성 파일 사용 (페이지 21)
- 데이터 원본 이해 (페이지 22)

추가 정보:

예제 구성 파일 (페이지 89)

여러 CA APM ChangeDetector 구성 파일 사용

CA APM ChangeDetector 구성 파일 하나를 지정하는 대신 파일 여러 개가 들어 있는 디렉터리를 지정할 수 있습니다. 디렉터리를 지정하면 CA APM ChangeDetector 가 시작할 때 해당 디렉터리에서 유효한 모든 구성 파일을 읽습니다.

CA APM ChangeDetector 구성 디렉터리를 지정하려면

- *IntroscopeAgent.profile* 을 편집하여 다음 속성을 설정합니다.
  `introscope.changeDetector.profileDir=<path to ChangeDetector directory>`

  이 속성의 기본값은 다음과 같습니다.
  `<path to IntroscopeAgent.profile>/changeDetector_profiles.`

  다음 속성에 단일 파일을 지정할 수도 있습니다.
  `introscope.changeDetector.profile`

  이 경우 CA APM ChangeDetector 에서 해당 파일과 지정된 구성 디렉터리의 모든 파일을 읽습니다.

데이터 원본 이해

데이터 원본은 파일, 데이터베이스 테이블 열 값, 런타임 클래스 인스턴스 등과 같은 데이터 원본 유형에 따라 유형이 정의되는 리소스 그룹입니다. 리소스는 이름(예: *C:\Introscope\Introscope Enterprise Manager.exe*), 데이터베이스의 *buffer_pool* 열 이름 또는 *java.lang.System* 으로 식별됩니다. 데이터 원본 하나에 리소스 여러 개가 있을 수 있으며, 리소스의 값은 여러 개일 수 있지만 한 번에 한 값만 가질 수 있습니다.

참고: 데이터 원본 내에서 리소스 이름은 고유해야 합니다.

데이터 원본은 다음 두 방법 중 하나를 사용하여 정의합니다.

- CA APM ChangeDetector 구성 마법사 사용 (페이지 23)
- CA APM ChangeDetector 구성 파일 수정 (페이지 34)

CA APM ChangeDetector 지원 에이전트의 경우 구성 파일에서 지정한 데이터 원본이 Investigator 의 "변경" 탭에 나타납니다. 변경 데이터 보기에 대한 자세한 내용은 CA APM ChangeDetector 데이터 보기 (페이지 63)를 참조하십시오.

각 데이터 원본 정의는 *datasource-type* 태그 안에 있어야 하며 *name* 및 *class* 특성을 포함해야 합니다. *name* 은 원하는 대로 지정할 수 있으며 나중에 구성 파일에서 참조됩니다. *class* 특성은 이 유형의 데이터 원본에 대한 datasource-instance 정의를 구문 분석하는 데 사용할 클래스를 정의합니다. 클래스에서는 다음 인터페이스를 구현해야 합니다.

`com.wily.cd.agent.config.IDataSourceConfig`

datasource-type 요소는 사용할 수 있는 데이터 원본의 유형을 에이전트에 알려 줍니다. *datasource-type* 요소를 활용하려면 *datasource-instance* 요소도 정의합니다. 이 요소 각각은 에이전트가 모니터링하는 물리적 데이터 원본에 해당합니다.

참고: 데이터 원본 인스턴스에는 이름이 같은 리소스 여러 개가 있을 수 없습니다. 예를 들어 이름이 같은 *javaenv* 속성과 같이 한 데이터 원본 인스턴스에 전체 경로와 이름이 같은 파일이 중복되는 것은 유효하지 않습니다. CA APM ChangeDetector 의 모든 데이터는 데이터 원본 인스턴스로 구성됩니다.

ChangeDetector 구성 마법사 사용

구성 마법사를 사용하여 CA APM ChangeDetector 를 구성할 수 있습니다. 필요한 경우 *ChangeDetector-config.xml* 을 업데이트하여 구성할 수도 있습니다. 이에 대해서는 예제 구성 파일 (페이지 89)을 참조하십시오.

구성 마법사로 다음 작업을 수행할 수 있습니다.

- 구성 마법사 실행 (페이지 23)
- 구성 마법사로 데이터 원본 추가 (페이지 24)
- 구성 마법사로 데이터 원본 수정 (페이지 25)
- 구성 마법사로 데이터 원본 제거 (페이지 26)

참고: CA APM ChangeDetector 구성 파일 여러 개를 사용할 수 있습니다. 자세한 내용은 여러 CA APM ChangeDetector 구성 파일 사용 (페이지 21)을 참조하십시오.

구성 마법사 실행

구성 마법사를 사용하면 XML 을 수동으로 편집하는 대신 그래픽 사용자 인터페이스를 사용하여 데이터 원본 속성을 설정할 수 있습니다.

구성 마법사를 실행하려면

1. CA APM ChangeDetector 를 설치한 후 <*Workstation_Home*>/*tools* 로 이동하고 명령 프롬프트에서 *configwizard.bat* 를 실행하여 구성 마법사를 시작합니다.

   참고: *JAVA_HOME* 이 설정되어 있지 않으면 명령 프롬프트에서 JRE 경로를 인수로 사용하여 *configwizard.bat* 를 실행합니다. 예제: *configwizard.bat s:\sw\sun\jre*

2. 구성 마법사에서 다음 작업을 수행할 수 있습니다.
   - 구성 마법사로 데이터 원본 추가 (페이지 24)
   - 구성 마법사로 데이터 원본 수정 (페이지 25)
   - 구성 마법사로 데이터 원본 제거 (페이지 26)

구성 마법사로 데이터 원본 추가

구성 마법사를 사용하여 다음과 같은 데이터 원본을 추가할 수 있습니다.

- 마법사로 어셈블리 모니터 구성 (페이지 26)

  참고: .NET 플랫폼에만 해당하는 내용입니다.
- 마법사로 Java 클래스 모니터 구성 (페이지 28)

  참고: Java 플랫폼에만 해당하는 내용입니다.
- 마법사로 데이터베이스 모니터 구성 (페이지 29)
- 마법사로 파일 시스템 모니터 구성 (페이지 30)
- 마법사로 .NET 환경 변수 모니터 구성 (페이지 33)

  참고: .NET 플랫폼에만 해당하는 내용입니다.
- 마법사로 Java 시스템 속성 모니터 구성 (페이지 33)

  참고: Java 플랫폼에만 해당하는 내용입니다.

데이터 원본을 추가하려면

1. 구성 마법사를 실행 (페이지 23)합니다.
2. 원하는 플랫폼을 선택하고 "다음"을 클릭합니다.
3. "새 CA APM ChangeDetector 구성 파일 만들기"를 선택하고 "다음"을 클릭합니다.

   이 마법사 페이지에서 새 구성 파일을 만들거나 기존 구성 파일을 수정할 수 있습니다.
4. 옵션을 선택하고 "다음"을 클릭합니다.

   여기서 만든 데이터 원본이 마법사 창의 "현재 데이터 원본" 섹션에 나타납니다. 이제 언제든지 데이터 원본을 편집하고 삭제할 수 있습니다.

   데이터 원본을 추가하면 현재 데이터 원본에 나타납니다. 계속해서 다양한 필드와 요소를 편집할 수 있으며, 필요한 경우 데이터 원본을 제거할 수 있습니다.

5. 추가하려는 데이터 원본 유형을 선택하고 "다음"을 클릭합니다.
6. 추가하려는 데이터 원본의 이름을 입력하고 "다음"을 클릭합니다.
   옵션은 이전 단계에서 지정한 플랫폼에 따라 달라집니다.
   - 적용 가능한 모든 데이터 원본을 계속 구성하고(마법사를 사용한 데이터 원본 구성 (페이지 26) 참조) 필요한 경우 선택적인 CA APM ChangeDetector 워크스테이션 및 Enterprise Manager 속성을 정의합니다(선택적 구성 속성 (페이지 59) 참조).
7. 모든 데이터 원본을 추가했으면 "다른 이름으로 저장"을 클릭하여 XML 파일로 저장합니다.

구성 마법사로 데이터 원본 수정

데이터 원본을 추가한 경우 데이터 원본을 수정할 수 있습니다.

참고: 이전 버전에서 업그레이드하는 경우 원래 파일의 백업을 저장할 수도 있고 업그레이드하면서 기존 파일을 덮어쓸 수도 있습니다.

데이터 원본을 수정하려면

1. 구성 마법사를 실행 (페이지 23)합니다.
2. 플랫폼을 선택하고 "다음"을 클릭합니다.
3. "Modify existing ChangeDetector"(기존 ChangeDetector 수정)을 선택하고 "다음"을 클릭합니다.
4. 수정할 구성 파일을 선택합니다.
5. 마법사의 왼쪽 창에서 데이터 원본을 선택하여 수정합니다.

   각 데이터 원본에는 수정이 가능한 다양한 필드와 요소가 있습니다. 수정하려면 필드를 변경하거나 요소를 선택한 다음 수정합니다.
6. 변경한 후 "저장"을 클릭한 다음 "끝내기"를 클릭합니다.

구성 마법사로 데이터 원본 제거

데이터 원본을 추가한 경우 데이터 원본을 제거할 수 있습니다.

데이터 원본을 제거하려면

1. 구성 마법사를 실행 (페이지 23)합니다.
2. 원하는 플랫폼을 선택하고 "다음"을 클릭합니다.
3. "선택한 데이터 원본 제거"를 선택하고 "다음"을 클릭합니다.
4. 데이터 원본을 제거할 구성 파일을 선택합니다.
5. 마법사의 왼쪽 창에서 데이터 원본을 선택하여 제거합니다.
6. 모두 변경한 후 "저장"을 클릭한 다음 "끝내기"를 클릭합니다.

마법사를 사용한 데이터 원본 구성

구성 마법사를 사용하여 다음과 같은 데이터 원본을 구성할 수 있습니다. 새 구성 파일에 데이터 원본을 추가한 후 "다른 이름으로 저장"을 클릭하여 이름을 지정하고 XML 파일을 저장합니다.

- 마법사로 어셈블리 모니터 구성 (페이지 26)

  참고: 어셈블리 모니터는 .NET 플랫폼에만 적용 가능합니다.
- 마법사로 Java 클래스 모니터 구성 (페이지 28)
- **참고:** Java 클래스 모니터는 Java 플랫폼에만 적용 가능합니다.
- 마법사로 데이터베이스 모니터 구성 (페이지 29)
- 마법사로 .NET 환경 변수 모니터 구성 (페이지 33)

  참고: .NET 환경 변수 모니터는 .NET 플랫폼에만 적용 가능합니다.
- 마법사로 Java 시스템 속성 모니터 구성 (페이지 33)

  참고: Java 시스템 속성은 Java 플랫폼에만 적용 가능합니다.

마법사로 어셈블리 모니터 구성

.NET 플랫폼의 경우 어셈블리 모니터 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 어셈블리 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.

   마법사의 다음 페이지가 나타납니다.

2. 다음과 같은 속성 구성 정보를 입력하고 "다음"을 클릭합니다.

   - **반복당 클래스 수** - 정의된 반복당 로드되는 클래스의 수입니다. 값이 작을수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **반복 간 지연** - 클래스가 모니터링되지 않을 때 반복 사이에 경과된 시간(초, 분 또는 시)입니다. 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **최초 대기 시간** - 클래스를 모니터링하기 전까지 대기하는 초기 시간(초, 분 또는 시)입니다.

   마법사의 다음 페이지가 나타납니다.

3. 다음 단계에 따라 어셈블리와 클래스에 대한 제외 요소를 추가합니다.

   a. "새 제외 요소 추가"를 선택합니다.

   b. "다음"을 클릭합니다.

   c. 해당 필드에 제외 패턴을 입력합니다.

   d. 포함 패턴을 추가하려면 "추가"를 클릭하고 해당 필드에 패턴을 입력합니다. 이 단계를 필요한 만큼 반복합니다.

   e. "다음"을 클릭합니다.

   f. 위 단계를 반복하여 더 많은 제외 패턴을 추가하거나 "완료"를 클릭합니다.

   참고: 첫 번째 제외 요소는 regex 와 일치하는 어셈블리를 모니터링하지 않도록 제외하고 두 번째 제외 요소는 regex 와 일치하는 클래스를 모니터링하지 않도록 제외합니다.
   제외 패턴을 사용하여 검색 범위를 좁히고 제외 패턴에 대한 예외가 있는 경우 포함 패턴을 추가합니다. 예를 들어 제외 패턴이 ".*"이고 포함 패턴은 "java\.*"일 수 있습니다.
   제외 패턴이나 포함 패턴에 정규식을 사용하십시오. 다음은 몇 가지 예제입니다.
   foo
   .*bar.
   .*
   com\.wily\.(.*)

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

마법사로 Java 클래스 모니터 구성

Java 플랫폼의 경우 Java 클래스 모니터 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 java 클래스 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.

   마법사의 다음 페이지가 나타납니다.

2. 다음과 같은 속성 구성 정보를 입력하고 "다음"을 클릭합니다.

   - **반복당 클래스 수** - 값이 작을수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **반복 간 지연** - 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.

   마법사의 다음 페이지가 나타납니다.

3. "새 제외 요소 추가"를 선택하고 "다음"을 클릭합니다.
4. 해당 필드에 제외 패턴을 입력합니다.
5. 포함 패턴을 추가하려면 "추가"를 클릭하고 해당 필드에 패턴을 입력합니다. 이 단계를 필요한 만큼 반복하고 "다음"을 클릭합니다.
6. 위 단계를 반복하여 더 많은 제외 패턴을 추가하거나 "완료"를 클릭합니다.

   참고: 제외 패턴을 사용하여 검색 범위를 좁히고 제외 패턴에 대한 예외가 있는 경우 포함 패턴을 추가합니다. 예를 들어 제외 패턴이 ".*"이고 포함 패턴은 "java\.*"일 수 있습니다.
   제외 패턴이나 포함 패턴에 정규식을 사용하십시오. 다음은 몇 가지 예제입니다.
   foo
   .*bar.
   .*
   com\.wily\.(.*)

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

마법사로 데이터베이스 모니터 구성

마법사를 사용하여 데이터베이스 모니터 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 데이터베이스 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.

   마법사의 다음 페이지가 나타납니다.

2. Java 플랫폼에 대한 다음과 같은 데이터베이스 정보를 입력하고 "다음"을 클릭합니다.

   - **JDBC 드라이버** - 이 데이터베이스에 대한 드라이버입니다. 예제: oracle.jdbc.driver.OracleDriver
   - **JDBC 드라이버 클래스 경로** - 데이터베이스 드라이버의 경로입니다("찾아보기"로 선택할 수 있음).
   - **JDBC URL** - 데이터베이스의 JDBC URL 을 입력합니다.
   - **사용자 이름** - 데이터베이스를 사용하는 사용자 이름을 입력합니다.
   - **암호** - 데이터베이스의 암호를 입력합니다. 암호는 자동으로 구성 파일에 알아보기 힘들게 기록됩니다.
   - **암호 확인** - 확인을 위해 암호를 다시 입력합니다.
   - **일정 유형** - "반복" 또는 "시작 후"가 있습니다. 거의 변경되지 않았거나 시작 시에만 응용 프로그램에서 확인하는 데이터베이스인 경우 "시작 후"를 선택하십시오.
   - **반복 간격** - 반복 일정인 경우 간격을 선택합니다. 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **반복 간 지연** - 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.

   마법사의 다음 페이지가 나타납니다.

3. .NET 플랫폼에 대한 다음과 같은 데이터베이스 정보를 입력하고 "다음"을 클릭합니다.

   - **URL** - 데이터베이스의 URL 을 입력합니다.
   - **사용자 이름** - 데이터베이스를 사용하는 사용자 이름을 입력합니다.
   - **암호** - 데이터베이스의 암호를 입력합니다. 암호는 자동으로 구성 파일에 알아보기 힘들게 기록됩니다.
   - **암호 확인** - 확인을 위해 암호를 다시 입력합니다.

   참고: .NET 플랫폼에서 URL 속성에 사용자 이름과 암호를 지정할 수도 있고 값을 지정하지 않을 수도 있습니다. 마법사는 URL 에서 이러한 값을 확인하지 않습니다.

   - **일정 유형** - "반복" 또는 "시작 후"가 있습니다. 거의 변경되지 않았거나 시작 시에만 응용 프로그램에서 확인하는 데이터베이스인 경우 "시작 후"를 선택하십시오.
   - **반복 간격** - 반복 일정인 경우 간격을 선택합니다. 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.

   마법사의 다음 페이지가 나타납니다.

4. SQL 문을 추가하려면 "새 SQL 문 추가"를 선택하고 "다음"을 클릭합니다.

   이 단계를 반복하여 더 많은 SQL 문을 추가하거나, 그렇지 않은 경우 "완료"를 클릭합니다. 예:

   ```
   SELECT name, value FROM v$parameter
   ```

   참고: 마법사는 SQL 문의 유효성을 검사하지 않습니다. 데이터베이스에 대해 유효한 SQL 문을 사용해야 합니다.

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

마법사로 파일 시스템 모니터 구성

마법사를 사용하여 파일 시스템 모니터 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 파일 시스템 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.

   마법사의 다음 페이지가 나타납니다.

2. "새 스캔-디렉터리 요소 추가"를 선택하고 "다음"을 클릭합니다.

3. 디렉터리 이름과 필수 필드를 입력합니다. 여기에는 "디렉터리 이름", "재귀", "파일 집합" 및 "사용"이 있습니다.

   - **디렉터리 이름** - CA APM ChangeDetector 로 검색할 디렉터리의 경로를 입력합니다. 예제: *C:\\WebLogic\\myApplicationHome*
   - **재귀** - True 값을 사용하면 CA APM ChangeDetector 가 지정된 디렉터리 아래의 모든 하위 폴더에서 찾습니다.
   - **파일 집합** - 이 scan-directory 와 연결된 파일 집합을 선택합니다.
   - **사용** - 테스트나 기타 용도로 scan directory 를 사용하지 않도록 설정할 수 있습니다. 하지만 scan directory 를 사용하도록 설정한 후 사용하지 않도록 설정하면 이 scan directory 요소에 포함된 파일이 CA APM ChangeDetector 에서 삭제된 것으로 보고됩니다.

4. 제외 패턴을 추가하려면 "새로 추가"를 클릭하고 패턴을 입력하십시오. 이 단계를 필요한 만큼 반복합니다.

5. "파일 집합 만들기 또는 수정"을 클릭하여 기존 파일 집합을 추가하거나 편집합니다.

   - "새 파일 집합 추가"를 선택하고 "다음"을 클릭합니다.
   - 파일 집합 이름을 입력하고 "다음"을 클릭합니다.

6. "다음"을 클릭하고 위 단계를 반복하여 더 많은 scan-directory 요소를 추가한 후 "완료"를 클릭합니다.

7. 포함/제외 요소를 입력하여 검색 범위를 좁히고 제외 패턴에 대한 예외가 있는 경우 포함 패턴을 추가합니다.

   - "새 제외 요소 추가"를 선택하고 "다음"을 클릭합니다.
   - 해당 필드에 제외 패턴을 입력합니다.
   - 포함 패턴을 입력하려면 "추가"를 클릭하고 해당 필드에 포함 패턴을 입력합니다. 이 단계를 필요한 만큼 반복합니다.

8. 다음 정보를 입력하고 "다음"을 클릭합니다.

   - **반복당 파일** - 값이 작을수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **파일 반복 간 지연** - 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.
   - **업로드할 최대 파일 크기** - 서버에 업로드되는 최대 ASCII 파일 크기입니다. 업로드된 ASCII 파일은 CA APM ChangeDetector 의 차이 뷰에서 확인할 수 있습니다.

   참고: 다음 아카이브 속성은 Java 플랫폼에서만 사용할 수 있습니다.

   - **다이제스트 사용** - "Always"(항상), "Never"(없음) 또는 "Needed"(필요함)를 선택합니다. 다이제스트는 MD5 와 같은 형태의 메시지 다이제스트입니다. 이 기능을 사용하면 CA APM ChangeDetector 가 해시 비교를 수행하여 touch 에 의한 파일 변경 사항을 무시할 수 있습니다. "Always"(항상)을 선택하면 성능에 영향을 미칠 수 있습니다. "Needed"(필요함)를 선택하면 타임스탬프와 파일 크기가 변경된 경우에만 다이제스트가 사용됩니다.
   - **아카이브 확대** - *.zip* 또는 *.jar* 등의 아카이브에서 개별 파일을 검색하려면 True 를 선택하십시오.
   - **반복당 아카이브 수** - "아카이브 확대"를 선택한 경우 이 값을 지정합니다. 값이 작을수록 검색에 필요한 시간은 늘어납니다. 아카이브 파일은 전체가 검색되므로 아카이브에 많은 수의 파일이 있는 경우 이 수를 제한해야 합니다.
   - **아카이브 반복 간 지연** - 값이 클수록 CPU 사용량은 줄어들지만 검색에 필요한 시간은 늘어납니다.

9. 파일 집합을 추가하고 이 섹션에 있는 "완료"를 선택한 후 "다음"을 클릭합니다.

   참고: 모니터링되는 파일 시스템에는 읽기 권한이 있어야 하며 파일 시스템이 네트워크에 있는 경우에는 네트워크가 작동해야 CA APM ChangeDetector 에서 파일을 감지할 수 있습니다.

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

마법사로 .NET 환경 변수 모니터 구성

.NET 플랫폼의 경우 환경 변수 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 파일 시스템 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.
2. "새 제외 요소 추가"를 선택하고 "다음"을 클릭하여 제외 요소를 추가합니다.

   해당 필드에 제외 패턴을 입력할 수 있습니다. 제외 패턴을 사용하여 검색 범위를 좁히고 제외 패턴에 대한 예외가 있는 경우 포함 패턴을 추가합니다. 예를 들어 제외 패턴이 ".*"이고 포함 패턴은 "java\.*"일 수 있습니다.

   제외 패턴이나 포함 패턴에 정규식을 사용하십시오. 다음은 몇 가지 예제입니다.

   foo

   .*bar.

   (.*)

   java\.*
3. "새로 추가"를 선택하고 "다음"을 클릭하여 포함 패턴을 추가합니다.

   해당 필드에 포함 패턴을 입력할 수 있습니다.
4. 계속해서 위 단계를 반복하여 포함/제외 패턴을 필요한 만큼 추가한 후 "완료"를 클릭합니다.

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

마법사로 Java 시스템 속성 모니터 구성

Java 플랫폼의 경우 Java 시스템 속성 모니터 데이터 원본을 추가하거나 수정할 수 있습니다.

마법사로 Java 시스템 모니터 데이터 원본을 구성하려면

1. 영숫자 문자를 사용하여 데이터 원본 이름을 입력하고 "다음"을 클릭합니다.
2. "새 제외 요소 추가"를 선택하고 "다음"을 클릭하여 제외 요소를 추가합니다.

   해당 필드에 제외 패턴을 입력할 수 있습니다. 제외 패턴을 사용하여 검색 범위를 좁히고 제외 패턴에 대한 예외가 있는 경우 포함 패턴을 추가합니다. 예를 들어 제외 패턴이 ".*"이고 포함 패턴은 "java\.*"일 수 있습니다.

   제외 패턴이나 포함 패턴에 정규식을 사용하십시오. 다음은 몇 가지 예제입니다.

   foo

   .*bar.

   (.*)

   java\.*
3. "새로 추가"를 선택하고 "다음"을 클릭하여 포함 패턴을 추가합니다.

   해당 필드에 포함 패턴을 입력할 수 있습니다.

추가 정보:

구성 마법사로 데이터 원본 추가 (페이지 24)
구성 마법사로 데이터 원본 수정 (페이지 25)

CA APM ChangeDetector 구성 파일 수정

XML 구성 파일을 편집하여 구성 설정을 수동으로 수정할 수 있습니다. 이 방법은 모든 플랫폼에서 사용할 수 있습니다. CA APM ChangeDetector 패키지에는 이 방법을 선택한 경우 참조할 수 있는 샘플 XML 파일이 들어 있습니다.

추가 정보:

예제 구성 파일 (페이지 89)

ChangeDetector-config.xml 파일 정보

ChangeDetector-config.xml 을 기반으로 사용자 지정 구성 파일을 만들어 CA APM ChangeDetector 로 모니터링할 변경 사항의 *유형을 지정*할 수 있습니다. CA APM ChangeDetector 에서 변경 사항은 데이터 원본별로 그룹화됩니다. 이 파일을 사용하면 다음과 같은 구성을 변경할 수 있습니다.

- 데이터베이스 모니터 속성 수동 구성 (페이지 36) - *database* 데이터 원본은 호환 데이터베이스에서 변경 데이터를 수집하는 방법을 정의합니다.
- 파일 시스템 모니터 속성 수동 구성 (페이지 39) - *file* 데이터 원본은 변경 사항을 모니터링할 파일을 정의합니다.
- Java 클래스 모니터 속성 수동 구성 (페이지 46) - *classmonitor* 데이터 원본은 모니터링할 Java 클래스를 정의합니다.

  참고: Java 플랫폼에만 해당하는 내용입니다.
- Java 시스템 속성 모니터 수동 구성 (페이지 47) - *javaenv* 데이터 원본은 모니터링할 Java 프로세스 시스템 속성을 정의합니다.

  참고: Java 플랫폼에만 해당하는 내용입니다.
- 어셈블리 모니터 속성 수동 구성 (페이지 48) - .NET 환경에 대한 어셈블리 모니터를 나타내는 *classmonitor* 데이터 원본은 CA APM ChangeDetector 에서 모니터링할 어셈블리를 알려 줍니다.

  참고: .NET 플랫폼에만 해당하는 내용입니다.
- .NET 환경 변수 모니터 속성 수동 구성 (페이지 52) - .NET 환경에 대한 시스템 모니터 속성을 나타내는 *javaenv* 데이터 원본은 CA APM ChangeDetector 에서 모니터링할 시스템 속성을 알려 줍니다.

  참고: .NET 플랫폼에만 해당하는 내용입니다.

중요! 전체 파일 경로를 사용하여 파일을 검색하면 CPU 사용률이 급등할 수 있습니다. 이를 피하기 위해서는 전체 경로 파일 이름만 사용하여 검색해야 합니다. 전체 경로 파일 이름만 사용하여 검색하려면 다음과 같이 *fullpath="true"* 속성을 추가하여 *ChangeDetector-config.xml* 파일을 편집하십시오.

```
<scan-directory recursive="true" name="/opt/Oracle" fileset="default"
enabled="true" fullpath="true" > </scan-directory>
```

구성 파일에서 시스템 또는 에이전트 속성 사용

구성 파일의 모든 XML 특성에서 시스템 속성 또는 에이전트 프로필 속성을 값으로 사용할 수 있습니다. 이렇게 하면 CA APM ChangeDetector 구성 파일에서 런타임에 확인되는 속성을 사용할 수 있습니다. 한 값에 시스템 속성과 에이전트 프로필 속성이 모두 지정된 경우 시스템 속성이 우선합니다.

다음 예를 참조하십시오.

```
<scan-directory name="${APPLICATION_HOME}/bin/" recursive="true"
fileset="default"/>
```

이 예에서 *${APPLICATION_HOME}*은 런타임에 현재 값으로 바뀝니다.

참고: 이러한 속성의 값은 런타임에 유효한 시스템 속성이나 에이전트 프로필 속성으로 매핑되어야 합니다.

추가 정보:

예제 구성 파일 (페이지 89)

데이터베이스 모니터 속성 수동 구성

database 데이터 원본 인스턴스는 CA APM ChangeDetector 가 호환 데이터베이스에서 변경 데이터를 수집하는 방법을 지정합니다. 데이터베이스에 대해 지원되는 JDBC 또는 OLEDB 드라이버가 있어야 합니다.

참고: 사용하는 플랫폼에 따라 구성 속성이 조금씩 달라집니다. 특정 플랫폼에서만 사용할 수 있는 특성은 이 단원에서 해당 내용이 참고 사항으로 명시됩니다.

참고: 이 요소는 예제 Java ChangeDetector-config.xml 파일 (페이지 90)에서 볼 수 있는 사용자 지정 구성 파일 예제에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="Orcl_on_aserver" type="database" version="8.0"
driver="oracle.jdbc.driver.OracleDriver"
driverClasspath="C:\\somePathTo\\classes12.zip"
url="jdbc:oracle:thin:@aserver:1521:orcl" username="a3f973777b9d"
password="f478831d9bcd65" isClearText="false" >
SQL Server
SELECT name, value FROM v$parameter
</sql>
<schedule type="repetitive" interval="1" unit="min" />
</datasource-instance>
```

클래스 경로 구분 기호(; 또는 :) 중 하나를 사용하여 데이터베이스 드라이버에 대한 클래스 경로를 여러 개 지정할 수 있습니다.

데이터베이스에 대한 사용자 이름 및 암호를 지정할 수 있습니다(있는 경우). 하지만 사용자 이름과 암호 중 하나만 지정할 수는 없습니다.

참고: 보안상의 이유로 사용자 이름 및 암호로 입력한 값은 알아보기 힘든 값으로 자동으로 바뀝니다. 값을 변경하려면 먼저 *isClearText="true"* 속성을 설정하고 원하는 대로 변경하십시오. 다음 번에 에이전트를 실행하면 *isClearText* 속성이 자동으로 *false* 로 다시 설정됩니다.

데이터베이스 모니터를 사용할 경우 모든 *datasource-instance* 요소에서 정의해야 하는 *name* 및 *type* 특성과 함께 *datasource-instance* 요소에 다음과 같은 특성을 설정해야 합니다.

- *driver* - 사용할 JDBC 호환 드라이버의 클래스 이름입니다. 이 예제에서는 Oracle 데이터베이스에 연결하므로 이 사용자 지정 구성 파일에 지정된 드라이버는 다음과 같습니다.
  `oracle.jdbc.driver.OracleDriver`

참고: 이 특성은 Java 플랫폼에서만 사용할 수 있습니다.

- *driverClasspath* - driver 특성에 참조된 드라이버가 들어 있는 아카이브의 경로입니다. 플랫폼에 대한 클래스 경로 구분 기호를 사용하여 드라이버를 여러 개 지정할 수 있습니다.

참고: 이 특성은 Java 플랫폼에서만 사용할 수 있습니다.

- *url* - 데이터베이스에 연결하는 데 사용할 JDBC URL 입니다.
- *username* - 데이터베이스에 연결하는 데 사용할 사용자 이름입니다. 이 값은 알아보기 힘들게 기록됩니다. 이는 에이전트를 다시 시작할 때 *cleartext* 속성이 *true* 로 설정되는 경우에도 마찬가지입니다.
- *password* - 데이터베이스에 연결하는 데 사용할 암호입니다. 이 값은 알아보기 힘들게 기록됩니다. 이는 에이전트를 다시 시작할 때 *cleartext* 속성이 *true* 로 설정되는 경우에도 마찬가지입니다.
- *isClearText* - 사용자 이름과 암호를 알아보기 힘들게 기록할지 여부를 정의합니다. 이 값이 *true* 이면 사용자 이름과 암호가 알아보기 힘들게 바뀌고 구성 파일을 이 바뀐 값으로 덮어씁니다.

보안상의 이유로 이 속성은 기본적으로 *false* 로 설정됩니다. 이 속성을 *true* 로 설정한 다음 에이전트를 다시 시작하면 속성이 다시 *false* 로 설정됩니다.

database 유형의 datasource-instance 요소 내에 두 가지 최상위 요소 *sql* 및 *schedule* 을 정의할 수 있습니다.

sql 요소는 테이블에서 모니터링하려는 정보를 수집하는 데 사용되는 SQL 문입니다. 이 유형의 *datasource-instance* 요소 내에서 사용할 수 있는 SQL 요소 수에는 제한이 없습니다.

참고: 이 SQL 문의 결과에서 열은 2 개만 생성되어야 하며 정의된 SQL 요소 수에 관계없이 첫 번째 열의 값은 고유(기본 키)해야 합니다. *resultset* 의 첫 번째 열에는 NULL 값을 사용할 수 없습니다. 두 번째 열이 NULL 값이면 해당 행이 없는 것으로 처리됩니다. 즉, *resultset* 에 해당 행이 존재하지 않습니다.

schedule 요소는 데이터베이스 모니터가 변경 내용을 검색하는 빈도를 정의합니다. 이 요소에서는 *type* 특성을 정의해야 합니다. 이 특성의 유효한 값은 다음과 같습니다.

- *repetitive - type* 특성의 값이 *repetitive* 이면 *interval* 및 *unit* 특성을 정의해야 합니다. *interval* 특성은 정수 값이어야 합니다. *unit* 특성은 *minute*, *hour*, *sec* 중 하나여야 합니다. 이 두 특성이 CA APM ChangeDetector 에서 데이터베이스의 변경 사항을 검색하는 빈도를 지정합니다. 예제에서는 CA APM ChangeDetector 가 1 분마다 한 번씩 검색합니다. 10 초마다 한 번씩 검색하려면 *interval* 값으로 *10* 을 입력하고 *unit* 값으로 *sec* 를 지정하십시오.
- *post-startup - type* 특성 값이 *post-startup* 이면 다른 특성을 정의할 필요가 없습니다. 이 *type* 특성 값은 에이전트가 시작된 후 CA APM ChangeDetector 가 제공된 SQL 문을 한 번만 검색하도록 지정합니다.

파일 시스템 모니터 속성 수동 구성

CA APM ChangeDetector 파일 모니터링 시스템을 사용하면 요구 사항과 I/O 오버헤드 허용치가 서로 다른 여러 조직에서 파일 변경 사항을 모니터링하는 작업을 제어할 수 있습니다. 이를 통해 변경 사항이 발생한 시점부터 변경 사항이 감지된 시점까지 처리 성능과 I/O 비용 및 시간 사이의 균형을 유지할 수 있습니다.

참고: 파일 모니터링 속성은 대/소문자를 구분합니다. 예를 들어 모니터링하려는 디렉터리의 이름이 *i18n* 이고 *scan-directory name* 속성이 *I18N* 으로 설정되어 있으면 CA APM ChangeDetector 가 디렉터리를 찾지 못합니다.

참고: 이 요소는 예제 Java ChangeDetector-config.xml 파일 (페이지 90)에서 볼 수 있는 사용자 지정 구성 파일 예제에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="C_Drive" type="file" version="8.0">
      <!-- datasource-instance specific XML here -->
      <property name="explodeArchiveFiles" value="true" />
      <!-- Accepted units are hour, min, sec -->
      <property name="delayBetweenIterations" value="1" unit="sec" />
      <property name="filesPerIteration" value="1" />
      <property name="delayBetweenArchiveIterations" value="10" unit="sec" />
      <property name="archiveFilesPerIteration" value="1" />
      <!-- Accepted units are bytes, KBytes, MBytes -->
      <property name="maxFileSizeToUpload" value="4" unit="KB" />
      <property name="useDigest" value="needed" />
      <fileset name="default">
         <exclude pattern="(.*)\.err" />
         <exclude pattern="(.*)\.log" />
         <exclude pattern="(.*)\.lok" />
         <exclude pattern="(.*)\.tlog" />
         <exclude pattern="(.*)\.log0(.*)" />
      </fileset>
      <fileset name="NoCode">
         <include-fileset name="default" />
         <exclude pattern="(.*)\.jar" >
             <include pattern="(.*)wily(.*)\.jar" />
         </exclude>
         <exclude pattern="(.*)\.zip" />
      </fileset>
      <!--  typically, the wily agent is installed in the "wily" directory. -->
      <scan-directory name="wily" recursive="true"
         fileset="default">
         <exclude name="data" />
      </scan-directory>
      <!-- scan the java home directory, as specified in the java.home system property
-->
      <scan-directory recursive="true" fileset="default"
         name="${java.home}" enabled="false">
         <exclude name="lib/zi" />
      </scan-directory>
      <!--  directories to be scanned in a typical jboss installation -->
      <scan-directory name="." recursive="true" fileset="default"
         enabled="false">
         <exclude name="log" />
         <exclude name="tmp" />
      </scan-directory>
      <!--  test scan directory -->
      <scan-directory recursive="true" fileset="NoCode" name="."
         enabled="true" />
      <!--  directories to be scanned in a typical WebSphere 5.0ee installation -->
      <!--  basically exclude the following dirs:
         _uninst, _uninstPME, BRBeans, classes, installableApps, logs, temp, tranlog,
wstemp -->
```

```
    <scan-directory recursive="true" name="." fileset="default"
       enabled="false">
       <exclude name="_uninst" />
       <exclude name="_uninstPME" />
       <exclude name="logs" />
       <exclude name="temp" />
       <exclude name="tranlog" />
       <exclude name="wstemp" />
    </scan-directory>
</datasource-instance>
```

File 데이터 원본 인스턴스 유형에서 요소 정의

file 유형의 datasource-instance 요소 내에서 다음 최상위 요소 3 개를 정의할 수 있습니다.

- property (페이지 41)
- fileset (페이지 44)
- scan-directory (페이지 45)

file 유형의 datasource-instance 는 구성 파일의 시작 부분에서 파일 시스템 모니터로 정의됩니다.

파일 변경 사항 모니터링 시스템은 *file* 데이터 원본 인스턴스 구성에 지정된 모든 파일을 순차적으로 검색합니다. CPU 는 작업 단위의 수집을 위해 할당된 후 다른 작업에 사용할 수 있도록 해제됩니다.

property 요소

각 *property* 요소에는 *name* 특성과 *value* 특성이 있어야 합니다. 여기에 정의된 일부 속성은 *units* 특성도 포함할 수 있습니다.

참고: .NET 플랫폼은 아카이브를 지원하지 않으므로 모든 아카이브 관련 속성은 Java 에서만 정의할 수 있습니다.

이 유형의 datasource-instance 에서는 다음과 같은 속성을 사용할 수 있습니다.

- *explodeArchiveFiles*

  Java Agent 를 사용하는 경우에만 이 속성을 적용할 수 있습니다. 이 속성은 .NET 에이전트를 실행하는 환경과 호환되지 않습니다.

  이 속성의 value 특성에는 *true* 또는 *false* 를 사용할 수 있습니다.

  value 가 *true* 로 설정되어 있으면 CA APM ChangeDetector 가 아카이브 파일의 콘텐츠를 보고합니다. CA APM ChangeDetector 가 모니터링하는 아카이브 파일에서 변경 사항이 발생하면 최소한 2 개의 변경 이벤트가 전송됩니다. 하나는 아카이브 파일이 수정된 것에 대한 이벤트이고, 다른 하나는 아카이브 파일 내의 콘텐츠 파일이 수정된 것에 대한 이벤트입니다. 수정된 콘텐츠 파일 역시 아카이브 파일인 경우 CA APM ChangeDetector 는 해당 아카이브 파일을 열고 중첩된 아카이브의 콘텐츠 파일에 대한 변경 이벤트를 전송하는 식으로 중첩된 아카이브를 계속 처리합니다.

  ZIP 및 GZIP 파일 형식만 아카이브로 지원됩니다(예: *zip, gzip, jar, ear, war, rar, sar*). CA APM ChangeDetector 는 *tar* 파일을 지원하지 않습니다.

  explodeArchiveFiles 값을 false 로 설정하면 CA APM ChangeDetector 가 아카이브 내부를 검색하지 않습니다. 따라서, 아카이브의 모든 변경 사항이 아카이브 자체의 수정/추가/삭제로 나타납니다.

  기본값은 *false* 입니다.

- *delayBetweenIterations*

  value 값으로 정수를 사용할 수 있으며 units 는 *sec, min* 또는 *hour* 입니다.

  이 속성은 파일 큐에 대한 각 반복 사이의 대기 시간을 정의하며 *filesPerIteration* 속성과 연결되어 있습니다.

  기본값은 3 초입니다.

- *filesPerIteration*

  value 값으로 정수를 사용하여 *delayBetweenIterations* 속성에 명시된 작업 단위에 대한 파일의 수를 정의할 수 있습니다.

  이 속성은 CPU 할당을 해제하기 전에 검색할 파일 수를 지정합니다. 반복당 파일 수가 10 인 경우 CA APM ChangeDetector 는 10 개 파일에서 변경 사항을 검색한 다음 *delayBetweenIterations* 속성의 정의에 따라 대기합니다. 이 대기 시간이 지난 후 다시 또 다른 10 개 파일을 검색하고 대기 상태로 돌아가는 작업을 반복합니다.

  기본값은 5 입니다.

- *delayBetweenArchiveIterations*

  value 값으로 정수를 사용할 수 있으며 units 는 *sec, min* 또는 *hour* 입니다.

  이 속성은 아카이브 큐의 대기 시간을 제어합니다. CA APM ChangeDetector 가 파일을 검색할 때 아카이브로 인식되는 파일이 있으면 CA APM ChangeDetector 는 해당 파일을 아카이브 큐에 배치하여 파일의 콘텐츠가 검사되게 합니다.

  이 속성은 *explodeArchiveFiles* 가 *true* 로 설정된 경우에만 의미가 있습니다. 그렇지 않으면 아카이브 파일이 일반 파일로 처리됩니다.

  기본값은 10 초입니다.

- *archiveFilesPerIteration*

  동일한 특성을 *filesPerIteration* 속성으로 허용합니다. *filesPerIteration* 속성과 마찬가지로, *archiveFilesPerIteration* 은 반복당 콘텐츠가 검색되는 아카이브의 수를 제어하는 아카이브 큐 속성입니다. 이 속성은 *explodeArchiveFiles* 가 *true* 로 설정된 경우에만 의미가 있습니다. 그렇지 않으면 아카이브 파일이 일반 파일로 처리됩니다.

  기본값은 1 입니다.

  참고: 아카이브의 전체 콘텐츠는 한 번에 업로드됩니다. 여기에 *filesPerIteration* 값은 적용되지 않습니다. 검색 중인 아카이브 내에서 아카이브가 발견되면 중첩된 아카이브가 아카이브 큐에 추가됩니다.

- *maxFileSizeToUpload*

  value 특성으로 정수를 사용할 수 있으며 units 로 *B*, *KB* 또는 *MB* 를 사용할 수 있습니다. 이 속성은 콘텐츠를 서버로 전송할 파일의 최대 크기를 정의합니다. 현재는 이 속성으로 정의된 파일 크기보다 작은 ASCII 파일만 전송됩니다.

  기본값은 50 KB 입니다.

- *useDigest*

  이 속성은 변경 사항을 감지할 때 메시지 다이제스트(예: MD5)를 사용할 방법을 정의합니다. 다이제스트를 사용하면 CA APM ChangeDetector 가 해시 비교를 수행하여 touch 에 의한 파일 변경 사항을 무시할 수 있습니다. 이 속성의 값은 다음과 같습니다.

  - never - 다이제스트를 사용하지 않습니다.
  - always - 항상 다이제스트 비교를 수행합니다.
  - needed - 타임스탬프와 파일 크기가 변경된 경우에만 다이제스트를 사용합니다(기본값).

fileset 요소

다음은 *fileset* 요소에 대한 예제입니다.

```
<fileset name="default">
         <exclude pattern="(.*)\.err" />
         <exclude pattern="(.*)\.log" />
         <exclude pattern="(.*)\.lok" />
         <exclude pattern="(.*)\.tlog" />
         <exclude pattern="(.*)\.log0(.*)" />
</fileset>
<fileset name="NoCode">
         <include-fileset name="default" />
         <exclude pattern="(.*)\.jar" >
             <include pattern="(.*)wily(.*)" />
         </exclude>
         <exclude pattern="(.*)\.zip" />
</fileset>
<fileset name="all">
         <exclude pattern="(.*)>
             <include pattern="(.*) />
         </exclude>
</fileset>
```

fileset 요소는 검색에서 포함하거나 제외할 파일의 패턴을 정의합니다. fileset 요소에서 다음과 같은 하위 요소를 사용할 수 있습니다.

- *exclude*
- *include-fileset*

exclude 요소에는 *pattern* 특성이 정의되어 있어야 합니다. *exclude* 요소는 *include* 요소를 하위 노드로 포함하지 않거나 하나 이상 포함할 수 있습니다. *include* 요소에는 *pattern* 특성이 정의되어 있어야 합니다. *include* 요소의 용도는 *exclude* 요소를 재정의하는 것입니다. 위 예제의 *NoCode* 파일 집합에서는 *.jar* 로 끝나는 모든 파일 이름을 제외하고자 합니다. 단 이름에 *wily* 가 있는 파일 이름은 예외입니다.

include-fileset 요소에는 *name* 특성이 정의되어 있어야 합니다. *include-fileset* 요소를 사용하는 경우 포함하려는 파일 집합이 이전에 정의되어 있어야 합니다. 위 예제에서 *NoCode* 파일 집합은 *default* 파일 집합을 포함합니다. 만약 두 파일 집합의 순서를 바꾸어 default 가 두 번째로 정의되는 파일 집합이 되면 잘못된 구성이 됩니다. fileset 요소에서 *include-fileset* 요소를 사용하는 경우 참조된 파일 집합의 모든 포함 패턴과 제외 패턴은 마스터 요소의 일부가 됩니다. 위 예제에서 "*NoCode*" 파일 집합은 **.jar, *.zip, *.err, *.log, *.lok* 등의 제외 패턴을 포함합니다.

fileset 요소 내에서 *exclude* 및 *include-fileset* 요소의 순서는 중요하지 않습니다. 파일이 정의된 제외 패턴과 일치하지 않으면 검색에 해당 파일이 포함됩니다. 파일이 정의된 제외 패턴과 일치하면 해당 파일이 제외 패턴 내부에 정의된 포함 패턴과 일치하는지 여부를 확인합니다. 포함 패턴과 일치하는 경우 검색에 해당 파일이 포함됩니다. 포함 패턴과 일치하지 않으면 해당 파일이 검색에서 제외됩니다.

scan-directory 요소

scan-directory 요소는 검색할 디렉터리를 정의합니다. *scan-directory* 요소의 특성은 *name*, *recursive*, *fileset* 및 *enabled* 입니다.

- *name* 특성은 검색할 디렉터리를 정의하며 필수입니다. 예를 들어 *c:\\test* 또는 *C:/test* 와 같은 디렉터리를 정의합니다.
- *recursive* 특성은 *true/false* 값을 사용하고 파일 시스템 모니터가 검색하는 모든 디렉터리를 재귀적으로 검색해야 하는지 여부를 정의합니다. 기본값은 *true* 입니다.
- fileset 특성은 이 데이터 원본 인스턴스에서 사용할 파일 집합을 정의합니다. 지정된 fileset 은 구성 파일의 앞부분에 정의되어야 합니다. 이 특성은 필수 특성입니다.
- *enabled* 특성은 *true/false* 값을 사용하고 *scan-directory* 요소를 사용하거나 사용하지 않도록 설정합니다.

scan-directory 요소는 *exclude* 자식 요소를 포함할 수 있습니다. *exclude* 요소에는 *name* 특성이 필요합니다. *name* 특성은 디렉터리에 매핑되어야 합니다.

참고: name 특성에 지정된 값은 정규식일 수 없습니다. 이 값은 실제 디렉터리를 매핑하는 리터럴 문자열이어야 합니다. 패턴에 기반하여 항목을 제외하려면 사용하는 *fileset* 요소를 수정하십시오.

다음은 예제 Java ChangeDetector-config.xml 파일 (페이지 90)에 정의되어 있는 이 요소의 예입니다.

```
<scan-directory name="." recursive="true" fileset="default"
   enabled="false">
   <exclude name="log" />
   <exclude name="tmp" />
</scan-directory>
```

Java 클래스 모니터 속성 수동 구성

참고: Java 플랫폼에만 해당하는 내용입니다.

classmonitor 데이터 원본 인스턴스는 모니터링할 Java 클래스를 CA APM ChangeDetector 에 알려 줍니다. 클래스 이름과 클래스의 로더가 리소스 이름으로 사용되고 클래스 정의와 일치하는 바이트 배열이 리소스 값으로 사용됩니다.

CA APM ChangeDetector 인스턴스당 이 유형의 *datasource* 인스턴스 하나만 사용할 수 있습니다.

참고: Java 는 요청할 때 로드되는 방식으로 작동하기 때문에 클래스가 실행되고 있는 바이너리의 일부가 아닌지, 또는 클래스가 아직 로드되지 않았는지를 CA APM ChangeDetector 에서는 파악할 수 없습니다. 따라서 *classmonitor* 데이터 원본은 삭제 변경 사항을 생성하지 않습니다.

참고: 이 요소는 예제 Java ChangeDetector-config.xml 파일 (페이지 90)에서 볼 수 있는 사용자 지정 구성 파일 예제에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="Java class monitor" type="classmonitor"  version="8.0">
      <property name="delayBetweenIterations" value="2" unit="sec"/>
      <property name="classesPerIteration" value="100" />
      <exclude pattern="foo" />
      <exclude pattern=".*bar.*">
         <include pattern="hello" />
         <include pattern=".*world.*" />
      </exclude>
   </datasource-instance>
```

exclude 요소의 구문은 Java 시스템 속성 모니터와 동일합니다.

추가 정보:

Java 시스템 속성 모니터 수동 구성 (페이지 47)

Classmonitor 데이터 원본 인스턴스 유형에서 요소 정의

참고: Java 플랫폼에만 해당하는 내용입니다.

classmonitor 유형의 데이터 원본 인스턴스에는 다음과 같은 2 가지 property 요소를 정의할 수 있습니다. 두 *property* 요소 각각에는 *name* 특성과 *value* 특성이 있어야 합니다.

- *delayBetweenIterations*

  value 값으로 정수를 사용할 수 있으며 units 는 *sec, min* 또는 hour 입니다.

  이 속성은 클래스 큐에 대한 각 반복 사이의 대기 시간을 정의하며 *classesPerIteration* 속성과 연결되어 있습니다.

  기본값은 2 초입니다.

- *classesPerIteration*

  value 값으로 정수를 사용하여 *delayBetweenIterations* 속성에 명시된 작업 단위에 대한 클래스의 수를 정의할 수 있습니다.

  이 속성은 CPU 할당을 해제하기 전에 검색할 클래스 수를 지정합니다. 반복당 클래스 수가 10 인 경우 CA APM ChangeDetector 는 10 개 클래스에서 변경 사항을 검색한 다음 *delayBetweenIterations* 속성의 정의에 따라 대기합니다. 이 대기 시간이 지난 후 다시 또 다른 10 개 클래스를 검색하고 대기 상태로 돌아가는 작업을 반복합니다.

  기본값은 100 입니다.

Java 시스템 속성 모니터 수동 구성

참고: Java 플랫폼에만 해당하는 내용입니다.

javaenv 데이터 원본 인스턴스는 모니터링할 Java 시스템 속성을 CA APM ChangeDetector 에 알려 줍니다. CA APM ChangeDetector 는 Java 프로그램 실행 중에 발생하는 런타임 변경 사항을 모니터링하는 것이 아니라 프로세스가 다시 시작될 때마다 그 동안 변경된 Java 시스템 속성 변경 사항을 모니터링합니다. 따라서 이 유형의 수집은 시작할 때 한 번만 실행됩니다. 속성 이름은 리소스 이름으로 사용되고 속성 값은 이러한 리소스의 값을 구성합니다.

참고: 이 요소는 예제 Java ChangeDetector-config.xml 파일 (페이지 90)에서 볼 수 있는 사용자 지정 구성 파일 예제에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="Java system properties" type="javaenv"  version="8.0">
      <exclude pattern="foo" />
      <exclude pattern=".*bar.*">
         <include pattern="hello" />
         <include pattern=".*world.*" />
      </exclude>
   </datasource-instance>
```

javaenv 데이터 원본 내에서 *exclude* 요소만 최상위 요소로 정의할 수 있습니다. *exclude* 요소는 하위 자식을 포함할 수 있습니다.

CA Introscope 에 표시하지 않으려는 속성을 제외하려면 *exclude* 요소를 사용하십시오. *exclude* 요소에는 *pattern* 특성이 정의되어 있어야 합니다. 예를 들어 CA Introscope 에 *foo* 속성을 표시하지 않으려면 예제에 표시된 *exclude* 요소를 사용하십시오.

exclude 요소에는 *include* 하위 노드를 원하는 수만큼 정의할 수 있습니다. *exclude* 요소와 마찬가지로, *include* 요소에는 *pattern* 특성이 정의되어 있어야 합니다. *include* 하위 노드는 *exclude* 요소의 동작을 재정의하는 데 사용됩니다. 예제에서 볼 수 있는 것처럼 *.*bar.** 와 일치하는 모든 속성이 제외되지만 *hello* 또는 *.*world.** 와 일치하는 속성은 예외입니다.

어셈블리 모니터 속성 수동 구성

참고: .NET 플랫폼에만 해당하는 내용입니다.

classmonitor 데이터 원본은 .NET 환경에 대한 어셈블리 모니터를 나타내며 CA APM ChangeDetector 가 모니터링할 어셈블리를 지정합니다.

어셈블리 모니터는 메서드를 한 번에 하나씩 로드합니다. 메서드에는 다음과 같은 메타데이터가 포함됩니다.

- 어셈블리 이름
- 버전
- 클래스
- 메서드
- 메서드 서명

이름이 같은 어셈블리의 서로 다른 버전 간에 메타데이터의 변경 사항을 모니터링할 수 있습니다. 예를 들어 *cd_sample.dll 1.0.0* 과 *cd_sample.dll version 1.0.1* 에 포함된 클래스를 동일한 메타데이터의 서로 다른 버전으로 처리하게 됩니다. 데이터 변경 사항이 모니터링되고 Workstation Investigator 에 표시됩니다. 하지만 어셈블리 이름이 변경되면 어셈블리가 새로운 리소스로 간주되므로 어셈블리 내의 클래스가 새 리소스가 되어 추가적인 이벤트로 처리됩니다.

참고: 이 요소는 예제 .NET ChangeDetectorDotnet-config.xml 파일 (페이지 96)에 예제로 제공된 사용자 지정 구성 파일에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="Assembly Monitor" type="classmonitor" version="8.0">

    <property name="initialWaitTime" value="30" unit="sec" />
    <property name="delayBetweenIterations" value="2" unit="min" />
    <property name="classesPerIteration" value="5" />
    <excludeassembly pattern=".\mscorlib\.dll"/>
    <excludeassembly pattern=".\System\.dll"/>
    <excludeassembly pattern=".\System\.Xml\.dll"/>
    <excludeassembly pattern=".\System\.Web\.dll"/>
    <excludeassembly pattern=".\System\.Configuration\.dll"/>
    <excludeassembly pattern=".\wily\.."/>
    <excludeassembly pattern=".\Microsoft\.JScript\.dll"/>
    <excludeassembly pattern=".\VJSharpCodeProvider\.dll"/>
    <excludeassembly pattern=".\System\.Data\.dll"/>
    <excludeassembly pattern=".\Oracle\.DataAccess\.dll"/>
    <excludeassembly pattern=".\System\.Web\.Mobile\.dll"/>
    <excludeassembly pattern=".\System\.ServiceModel\.dll"/>
    <excludeassembly pattern=".\SMDiagnostics\.dll"/>
    <excludeassembly pattern=".\System\.Drawing\.dll"/>
    <excludeassembly pattern=".\System\.Web\.RegularExpressions\.dll"/>
    <excludeassembly pattern=".\Microsoft\.VisualBasic\.dll"/>
    <excludeassembly pattern=".\CppCodeProvider\.dll"/>
```

```
    <excludeassembly pattern=".\System\.EnterpriseServices\.dll"/>
    <excludeassembly pattern=".\System\.Transactions\.dll"/>

    <exclude pattern="com\.wily\.(.*)"/>

</datasource-instance>
```

이러한 요소의 구문은 다음과 같습니다.

- *excludeassembly*

  <!-- exclude assemblies -->

  <excludeassembly pattern=".\mscorlib\.dll"/>

  <excludeassembly pattern=".\System\.dll"/>

  <excludeassembly pattern=".\System\.Xml\.dll"/>

  <excludeassembly pattern=".\System\.Web\.dll"/>

  <excludeassembly pattern=".\System\.Configuration\.dll"/>

  <excludeassembly pattern=".\wily\.."/>

  <excludeassembly pattern=".\Microsoft\.JScript\.dll"/>

  <excludeassembly pattern=".\VJSharpCodeProvider\.dll"/>

  <excludeassembly pattern=".\System\.Data\.dll"/>

  <excludeassembly pattern=".\Oracle\.DataAccess\.dll"/>

  <excludeassembly pattern=".\System\.Web\.Mobile\.dll"/>

  <excludeassembly pattern=".\System\.ServiceModel\.dll"/>

  <excludeassembly pattern=".\SMDiagnostics\.dll"/>

  <excludeassembly pattern=".\System\.Drawing\.dll"/>

  <excludeassembly pattern=".\System\.Web\.RegularExpressions\.dll"/>

  <excludeassembly pattern=".\Microsoft\.VisualBasic\.dll"/>

  <excludeassembly pattern=".\CppCodeProvider\.dll"/>

  <excludeassembly pattern=".\System\.EnterpriseServices\.dll"/>

  <excludeassembly pattern=".\System\.Transactions\.dll"/>

- exclude

  <exclude pattern="com\.wily\.(.)"/>*

- include

  <excludeassembly pattern=".\System\.dll"/>

  <exclude pattern="abc\.xyz\.(.)">*

  <include pattern="abc\.xyz\.asdf\.(.)"/> </exclude>*

classmonitor 유형의 데이터 원본 인스턴스에는 다음과 같은 property 요소를 정의할 수 있습니다. 두 *property* 요소 각각에는 *name* 특성과 *value* 특성이 있어야 합니다.

- *initialWaitTime*

  value 값으로 정수를 사용할 수 있으며 units 는 sec, min 또는 hour 입니다.

  이 속성은 마지막 어셈블리를 로드한 후 에이전트가 클래스를 검색하기 전에 대기하는 시간을 정의합니다.

- *delayBetweenIterations*

  value 값으로 정수를 사용할 수 있으며 units 는 sec, min 또는 hour 입니다.

  이 속성은 클래스 큐에 대한 각 반복 사이의 대기 시간을 정의하며 classesPerIteration 속성과 연결되어 있습니다.

  기본값은 2 초입니다.

- *classesPerIteration*

  value 값으로 정수를 사용하여 delayBetweenIterations 속성에 명시된 작업 단위에 대한 클래스의 수를 정의할 수 있습니다.

  이 속성은 CPU 할당을 해제하기 전에 검색할 클래스 수를 지정합니다. 반복당 클래스 수가 10 인 경우 CA APM ChangeDetector 는 10 개 클래스에서 변경 사항을 검색한 다음 delayBetweenIterations 속성의 정의에 따라 대기합니다. 이 대기 시간이 지난 후 다시 또 다른 10 개 클래스를 검색하고 대기 상태로 돌아가는 작업을 반복합니다.

  기본값은 100 입니다.

.NET 환경 변수 모니터 속성 수동 구성

참고: .NET 플랫폼에만 해당하는 내용입니다.

.NET 환경 변수 모니터 데이터 원본(javaenv) 인스턴스는 CA APM ChangeDetector 가 모니터링할 환경 변수를 지정합니다. CA APM ChangeDetector 는 응용 프로그램 실행 중에 발생하는 런타임 변경 사항을 모니터링하는 것이 아니라 프로세스가 다시 시작될 때마다 그 동안 변경된 환경 변수 변경 사항을 모니터링합니다. 따라서 이 유형의 수집은 시작할 때 한 번만 실행됩니다. 변수 이름은 리소스 이름으로 사용되고 변수 값은 이러한 리소스의 값을 구성합니다.

참고: 이 요소는 예제 .NET ChangeDetectorDotnet-config.xml 파일 (페이지 96)에 예제로 제공된 사용자 지정 구성 파일에 정의되어 있습니다. 이 예에서 제공하는 예제 데이터는 사용자 데이터의 콘텐츠를 반영할 수도 있고 반영하지 못할 수도 있습니다. 다른 버전의 CA APM ChangeDetector 를 사용하는 경우 예제 데이터를 적절한 콘텐츠로 대체하십시오.

```
<datasource-instance name="System Properties" type="javaenv"  version="8.0">
      <exclude pattern="foo" />
      <exclude pattern=".*bar.*">
         <include pattern="hello" />
         <include pattern=".*world.*" />
      </exclude>
   </datasource-instance>
```

javaenv 데이터 원본 내에서 exclude 요소만 최상위 요소로 정의할 수 있습니다. exclude 요소는 하위 자식을 포함할 수 있습니다.

CA Introscope 에 표시하지 않으려는 속성을 제외하려면 exclude 요소를 사용하십시오. exclude 요소에는 pattern 특성이 정의되어 있어야 합니다. 예를 들어 CA Introscope 에 foo 속성을 표시하지 않으려면 예제에 표시된 exclude 요소를 사용하십시오.

exclude 요소에는 include 하위 노드를 원하는 수만큼 정의할 수 있습니다. exclude 요소와 마찬가지로, include 요소에는 pattern 특성이 정의되어 있어야 합니다. include 하위 노드는 exclude 요소의 동작을 재정의하는 데 사용됩니다. 예제에서 볼 수 있는 것처럼 .*bar.*와 일치하는 모든 속성이 제외되지만 hello 또는 .*world.*와 일치하는 속성은 예외입니다.

기존 ChangeDetector-config.xml 파일 업그레이드

CA APM ChangeDetector 구성 파일을 업그레이드하면 자동으로 새 형식으로 업그레이드됩니다.

참고: 기존 구성 파일을 원래 이름 *.bak* 로 업그레이드하려면 구성 파일과 구성 파일의 현재 디렉터리에 대한 쓰기 액세스 권한이 있어야 합니다.

CA APM ChangeDetector 는 기존 구성 파일의 이름을 변경하고 업그레이드된 파일로 기존 구성 파일을 대체합니다. 기존 파일의 이름을 변경할 수 없거나 디렉터리에 쓸 수 없으면 새 업그레이드 구성 파일이 임시 디렉터리에 저장되고 기존 파일을 새 파일로 덮어쓰라는 지침이 나타납니다.

에이전트 구성 파일 수정

IntroscopeAgent.profile 을 수정하여 CA APM ChangeDetector 에이전트 확장 ID, CA APM ChangeDetector 구성 파일 경로 및 장애 조치 메커니즘을 지정합니다. 이러한 수정 사항이 항상 필요한 것은 아니며 자동 ID 할당을 대신 사용할 수 있습니다.

ChangeDetector 에이전트 ID 및 구성 파일 경로 설정(선택 사항):

1. 각 ChangeDetector 확장 에이전트 인스턴스에서 *IntroscopeAgent.profile* 을 기억하기 쉬운 새 이름(예: *IntroscopeAgentBackup.profile*)을 사용하여 별도의 디렉터리에 백업합니다.
2. *IntroscopeAgent.profile* 을 편집하여 다음 속성을 설정합니다.

   introscope.changeDetector.agentID=<ChangeDetector agent ID>

   agentID 속성은 영숫자와 밑줄(_) 및 하이픈(-)만을 포함할 수 있습니다.

   이 ID 는 CA APM ChangeDetector 에이전트의 전역 ID 에 해당하며 Enterprise Manager 또는 Enterprise Manager 클러스터에 연결하는 모든 에이전트에서 고유해야 합니다.
3. 각 ChangeDetector 확장 에이전트 인스턴스에서 구성 파일의 경로(파일 이름 포함)를 포함하도록 *introscopeAgent.profile* 을 편집합니다.

   introscope.changeDetector.profile=<path to ChangeDetector-config.xml>

   이름을 변경하지 않고 예제 구성 파일을 사용한 경우 파일 이름은 *ChangeDetector-config.xml* 입니다. 이름을 변경한 경우에는 해당 이름을 지정하십시오.

경로 이름을 입력할 때는 절대 경로 이름과 백슬래시 두 개를 사용합니다. 예를 들어 WebLogic 응용 프로그램 서버를 참조하는 경우 속성 값에 입력하는 경로는 다음과 같습니다.
<ProductName_Home>\\<Agent_Home>

참고: 각 에이전트에 대한 구성 파일의 경로를 지정해야 합니다. *IntroscopeAgent.profile* 에서 두 개 이상의 에이전트를 실행하는 경우 명령줄에서 이 속성을 정의하십시오. 예를 들어 응용 프로그램 서버의 시작 스크립트에서 속성을 정의합니다. 이렇게 하면 CA Introscope 가 CA APM ChangeDetector 에이전트 확장 구성을 구분할 수 있습니다. 새 IntroscopeAgent 프로필을 만드는 방법에 대한 자세한 내용은 *CA APM Java Agent 구현 안내서*를 참조하십시오.

추가 정보:

ChangeDetector 에이전트 ID 명명 옵션 (페이지 57)

부하가 분산된 환경에서 CA APM ChangeDetector 구성

CA APM ChangeDetector 에이전트 확장을 부하가 분산된 에이전트 환경에서 작동하도록 구성할 수 있습니다. 이렇게 하려면 CA APM ChangeDetector 에이전트 확장이 있는 에이전트를 MOM(Manager of Managers)에 연결하도록 구성합니다. 그러면 MOM 이 weight 속성(*introscope.enterprisemanager.clustering.login.em1.weight*) 구성에 따라 에이전트 부하를 적절한 수집기로 분산시킵니다.

부하 분산을 위해 CD 를 구성하려면

- ChangeDetector 에이전트를 MOM 에 연결하도록 구성하려면 다음 속성을 설정하십시오.

```
introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=<MOM Host>
introscope.agent.enterprisemanager.transport.tcp.port.DEFAULT=5001
introscope.agent.enterprisemanager.transport.tcp.socketfactory.DEFAULT=com.wily.isengard.postofficehub.link.net.DefaultSocketFactory
```

- MOM 의 부하 분산을 구성하려면 다음 속성을 설정하십시오.

  (수집기 1)

```
introscope.enterprisemanager.clustering.login.em1.host=sqw32vserv12
introscope.enterprisemanager.clustering.login.em1.port=5001
introscope.enterprisemanager.clustering.login.em1.publickey=internal/server/EM.public
introscope.enterprisemanager.clustering.login.em1.weight=50
```

(수집기 2)

```
introscope.enterprisemanager.clustering.login.em2.host=sqw32vserv10
introscope.enterprisemanager.clustering.login.em2.port=5002
introscope.enterprisemanager.clustering.login.em2.publickey=internal/server/EM.public
introscope.enterprisemanager.clustering.login.em2.weight=50
```

에이전트 속성에 대한 자세한 내용은 *CA APM Java Agent 구현 안내서* 또는 *CA APM .Net 에이전트 구현 안내서*를 참조하십시오. Enterprise Manager 속성에 대한 자세한 내용은 *CA APM 구성 및 관리 안내서*를 참조하십시오.

에이전트 장애 조치 메커니즘을 위한 CA APM ChangeDetector 구성

에이전트 장애 조치 메커니즘과 함께 작동하도록 CA APM ChangeDetector 에이전트 확장을 설치한 경우 다음 절차에 따라 기본 및 대체 Enterprise Manager 수집기를 구성하십시오.

장애 조치 상황에서 CA APM ChangeDetector 에이전트 확장은 대체 Enterprise Manager 로 지정된 Enterprise Manager 로 변경 데이터를 보냅니다. 하지만 대체 Enterprise Manager 의 변경 데이터베이스(changes.db)는 기본 Enterprise Manager 와 동기화 상태가 아니므로 CA APM ChangeDetector 가 종료됩니다. 이 상태에서도 CA Introscope 에이전트는 계속 실행되므로 CA APM ChangeDetector 는 기본 Enterprise Manager 가 복구될 때까지 계속 연결을 시도합니다.

참고: CA Introscope 에이전트가 MOM 에 직접 연결된 경우 변경 사항이 보고되지 않습니다.

에이전트 장애 조치 메커니즘을 위해 CD 를 구성하려면

IntroscopeAgent.profile 에서 다음 속성을 구성합니다.

- 에이전트가 CA APM ChangeDetector 를 지원할 수 있도록 MOM 이 아닌 기본 Enterprise Manager 수집기를 지정합니다. 다음 속성을 설정합니다.

```
introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=<primary EM collector host>
introscope.agent.enterprisemanager.transport.tcp.port.DEFAULT=<primary EM collector port>
introscope.agent.enterprisemanager.transport.tcp.socketfactory.DEFAULT=com.wily.isengard.postofficehub.link.net.DefaultSocketFactory
```

- MOM Enterprise Manager 를 대체 Enterprise Manager 로 지정합니다. 다음 속성을 설정합니다.

```
introscope.agent.enterprisemanager.transport.tcp.host.FALLBACK =<MOM EM collector host>
introscope.agent.enterprisemanager.transport.tcp.port.FALLBACK =<MOM EM collector port>
introscope.agent.enterprisemanager.transport.tcp.socketfactory.FALLBACK=com.wily.isengard.postofficehub.link.net.DefaultSocketFactory
```

- 다음 속성을 설정하여 연결 순서와 재시도 간격을 지정하십시오.

```
introscope.agent.enterprisemanager.connectionorder=DEFAULT, FALLBACK
introscope.agent.enterprisemanager.failbackRetryIntervalInSeconds=<failover retry time in seconds>
```

에이전트 속성에 대한 자세한 내용은 *CA APM Java Agent 구현 안내서* 또는 *CA APM .NET 에이전트 구현 안내서*를 참조하십시오.

.NET 에서 개별적인 구성 파일로 여러 응용 프로그램 실행

.NET 에서 IIS 를 통해 여러 응용 프로그램을 실행하며 각 응용 프로그램이 개별적으로 변경 사항을 보고하게 만들려는 사용자는 쉼표로 구분된 목록으로 각 응용 프로그램의 구성 폴더를 지정해야 합니다. 각 응용 프로그램에는 고유한 ChangeDetector config 폴더가 있어야 합니다. 이 폴더에는 해당 응용 프로그램의 ChangeDetector 구성 XML 이 있습니다. 경로에서 마지막 폴더 이름은 응용 프로그램 이름(AppDomain 표시 이름) 부분과 일치해야 합니다.

예를 들어 BalloonShop 및 Petshop 의 변경 사항을 모니터링하고 있으며 각 응용 프로그램에 고유한 구성 폴더가 있는 경우 *introscope.changeDetector.profileDir* 은 다음 예와 같이 구성해야 합니다.

```
introscope.changeDetector.profileDir=S:\sw\CA_Wily\wily_dotnet\Introscope<VersionNumber>\wily\CD-config\balloonshop,S:\sw\CA_Wily\wily_dotnet\Introscope<VersionNumber>\wily\CD-config\petshop
```

CA APM ChangeDetector 사용 안 함

CA APM ChangeDetector 를 사용하지 않도록 설정하려면

다음 속성을 false 로 설정하십시오.

```
introscope.changeDetector.enable=false
```

ChangeDetector 에이전트 ID 명명 옵션

설치하는 각 CA APM ChangeDetector 에이전트 확장에서 ChangeDetector 에이전트 ID 는 고유해야 합니다. CA Introscope 에이전트가 하나만 있는 단순한 환경에서는 CA APM ChangeDetector 가 자동으로 ID 를 할당할 수 있습니다.

더 복잡한 환경에서는 다른 방법을 사용하여 고유한 ChangeDetector 에이전트 ID 를 얻을 수 있습니다. 어떤 옵션을 사용할지는 다음에 따라 달라집니다.

- CA APM ChangeDetector 지원 에이전트가 여러 개 있는지 여부
- 에이전트 각각에 고유한 프로필 파일(IntroscopeAgent.profile)이 있는지, 아니면 모든 에이전트가 동일한 프로필 파일을 사용하는지 여부
- CA APM ChangeDetector 에이전트 확장에서 자동 ID 생성을 사용할 것인지 여부
- java 시스템 속성, .NET 환경 변수 모니터 또는 에이전트 속성 중 CA APM ChangeDetector 에이전트 확장을 식별하는 데 사용할 항목

참고: 동일한 CA Introscope 에이전트에서 동일한 ID 의 CA APM ChangeDetector 에이전트 확장이 두 개 이상 실행 중인 경우 두 번째 인스턴스가 시작될 때 자동으로 종료되고 오류 메시지가 출력됩니다.

다음 표에서는 권장 옵션 몇 가지를 요약하여 보여 줍니다.

환경	권장 에이전트 ID 설정
각 CA APM ChangeDetector 지원 에이전트가 고유한 디렉터리에 위치한 고유한 에이전트 프로필을 사용하는 경우	■ CA APM ChangeDetector 의 자동 ID 할당을 사용합니다. 이 경우 다른 조치가 필요하지 않습니다.
CA APM ChangeDetector 지원 에이전트 여러 개가 동일한 에이전트 프로필을 사용하는 경우	■ Java 시작 명령에서 -D 매개 변수를 사용하여 *introscope.changeDetector.agentID* 속성을 지정합니다. ■ Java 명령줄이나 *IntroscoptAgent.profile* 에서 유용한 Java 시스템 속성, .NET 환경 변수 모니터 또는 에이전트 프로필 속성에 기반한 식을 사용하여 런타임에 해결합니다.

추가 정보:

-D Java 시스템 매개 변수를 사용하여 ID 할당 (페이지 58)
Java 시스템 또는 에이전트 프로필 속성에 기반한 ID 할당 (페이지 58)

자동 ID 할당 사용

CA APM ChangeDetector 지원 에이전트마다 IntroscopeAgent.profile 이 하나씩 있는 경우에만 자동 ID 할당을 사용할 수 있습니다.

이 경우 Java 시스템 명령이나 *IntroscopeAgent.profile* 파일에 *introscope.changeDetector.agentID* 속성을 정의하지 않으면 CA APM ChangeDetector 가 이 속성에 대한 고유한 값을 생성합니다. 이 값은 CA APM ChangeDetector 루트 디렉터리에 있는 ".id" 파일에 저장됩니다. 이 파일을 삭제하거나 수정하지 마십시오.

참고: CA APM ChangeDetector 루트 디렉터리(change_detector)는 기본적으로 *IntroscopeAgent.profile* 파일이 들어 있는 디렉터리에 생성됩니다. 하지만 *introscope.changeDetector.rootDir* 속성을 정의하여 다른 루트 디렉터리를 지정할 수 있습니다.

-D Java 시스템 매개 변수를 사용하여 ID 할당

Java 명령줄에서 CA APM ChangeDetector 에이전트 확장에 대한 ID 를 정의할 수 있습니다. 이 작업을 수행하려면 다음 예에서 구성한 것처럼 -D 매개 변수와 *introscope.changeDetector.agentID* 속성을 사용하십시오.

```
java -Dintroscope.changeDetector.agentID=<value>
```

Java 시스템 또는 에이전트 프로필 속성에 기반한 ID 할당

원하는 Java 시스템 속성이나 에이전트 프로필 속성에 기반하여 런타임에 ChangeDetector 에이전트 ID 를 동적으로 할당할 수 있습니다. 환경이 복잡하며 이미 다른 속성을 사용하여 다양한 프로세스를 구분하고 있는 경우 이 방법을 사용할 수 있습니다.

다른 속성에 기반하여 ChangeDetector 에이전트 ID 를 할당하려면(옵션 1)

- *IntroscopeAgent.profile* 파일에서 에이전트 ID 속성의 식을 다음 예제와 같이 입력합니다. 여기서 접두사와 접미사는 간단한 상수 문자열입니다.

  ```
  introscope.changeDetector.agentID=<prefix>${any Java system property}<suffix>
  introscope.changeDetector.agentID=<prefix>${any Agent profile property}<suffix>
  ```

  접두사나 접미사 중 하나만 사용하거나 둘 모두를 사용할 수 있습니다.

다른 속성에 기반하여 ChangeDetector 에이전트 ID 를 할당하려면(옵션 2)

- 다음 예에서 볼 수 있는 것처럼, Java 시작 명령에서 위와 유사한 식을 사용합니다. 여기에서 접두사와 접미사는 간단한 상수 문자열입니다.

  ```
  java -Dintroscope.changeDetector.agentID=<prefix>${any Java system property}<suffix>
  java -Dintroscope.changeDetector.agentID=<prefix>${any Agent profile property}<suffix>
  ```

  접두사나 접미사 중 하나만 사용하거나 둘 모두를 사용할 수 있습니다.

선택적 구성 속성

CA APM ChangeDetector 에는 CA Introscope 에이전트, Workstation 및 Enterprise Manager 구성 파일에서 설정할 수 있는 선택적 속성이 있습니다.

선택적 에이전트 속성

IntroscopeAgent.profile 에서 다음 속성을 설정합니다.

- *introscope.changeDetector.rootDir*

  CA APM ChangeDetector 루트 디렉터리의 위치를 지정합니다. 해당 위치가 없는 경우 자동으로 생성됩니다. 이 위치를 지정하지 않으면 루트 디렉토리는 기본적으로 *<Agent_Home>*입니다.

  참고: CA APM ChangeDetector 에서는 루트 디렉터리를 사용하여 일반적인 처리에 필요한 파일을 생성합니다. 따라서 이 디렉터리를 삭제해서는 안 됩니다. 디렉터리 위치를 정의하는 속성은 선택 사항입니다.

- *introscope.changeDetector.compressEntries.enable*

  이 값을 false 로 설정하면 데이터 압축이 사용되지 않습니다. 데이터 압축을 사용하면 CA APM ChangeDetector 데이터 버퍼에 대한 압축이 허용됩니다. 시작할 때 메모리 사용량 문제가 발생하면 이 속성을 설정하는 것이 좋습니다. 기본값은 *true* 입니다. 이 값을 *false* 로 설정한 경우 설정을 적용하려면 응용 프로그램을 다시 시작해야 합니다.

- *introscope.changeDetector.compressEntries.batchSize*

  이 속성은 *introscope.changeDetector.compressEntries.enable* 속성과 함께 사용됩니다. 이 속성은 압축 작업의 배치 크기를 정의합니다. 기본값은 1000 입니다.

- *introscope.changeDetector.isengardStartupWaitTimeInSec*

  이 속성은 기본적으로 사용할 수 있습니다. 이 속성을 사용하면 에이전트가 시작된 후 Enterprise Manager 에 연결을 시도하기 전까지 대기해야 하는 시간을 초 단위로 지정할 수 있습니다. 기본값은 15 입니다.

- *introscope.changeDetector.waitTimeBetweenReconnectInSec*

  이 속성은 기본적으로 사용할 수 있습니다. 이 속성을 사용하면 에이전트가 Enterprise Manager 에 다시 연결하려고 시도하기 전까지 대기해야 하는 시간을 초 단위로 지정할 수 있습니다. 기본값은 10 입니다.

선택적 Workstation 속성

IntroscopeWorkstation.properties 에서 다음 속성을 설정할 수 있습니다.

- *introscope.changeDetector.defaultDataWindowValue=<value>*

  Investigator 에서 라이브 데이터 보기에 사용할 시간을 지정합니다. 이 속성과 *defaultDataWindowUnit* 속성을 함께 사용하여 라이브 데이터 보기에 사용할 시간 단위 값을 지정합니다. 기본값은 1 입니다.

- *introscope.changeDetector.defaultDataWindowUnit*=<value>

  Investigator 에서 라이브 데이터 보기에 사용할 시간 단위를 지정합니다. 이 속성은 *defaultDataWindowValue* 속성과 함께 사용됩니다. 이 속성의 유효한 값은 *seconds*, *minutes*, *hours*, *days*, *weeks* 및 *months* 입니다. 기본값은 *weeks* 입니다. days 또는 months 를 지정한 경우 사용되는 실제 시작 날짜는 사용자의 로캘에 따라 달라집니다. 예를 들어 한 주가 일요일에 시작하는 로캘도 있고 월요일에 시작하는 로캘도 있습니다.

- *introscope.changeDetector.useCh*angeTime=false

  기본적으로 CA APM ChangeDetector 는 파일 변경 사항에 대한 감지된 시간을 표시하고 수정 시간은 표시하지 않습니다. 감지된 시간과 수정 시간을 모두 표시하려면 이 속성을 *true* 로 설정하십시오. 이 속성은 파일 변경 사항에만 적용되고 시스템 속성, 환경 변수, 데이터베이스 또는 java 클래스 변경 사항에는 적용되지 않습니다.

CA APM ChangeDetector 데이터를 보내도록 EPAgent 플러그 인 구성

참고: Java 플랫폼에만 해당하는 내용입니다.

EPAgent 플러그 인이나 확장에 CA APM ChangeDetector 를 사용하는 경우 EPAgent 를 구성하십시오.

CA APM ChangeDetector 에서는 STDOUT 로 출력되는 XML 을 사용하여 Enterprise Manager 로 데이터를 보내므로 CA APM ChangeDetector 가 인식할 수 있는 형식으로 EPAgent 데이터 형식을 지정해야 합니다.

```
<changeData dataSource="dataSource name">
<resource name="resource1" value="resource1 value"/>
<resource name="resource2" value="resource2 value"/>
</changeData>
```

참고: ChangeDetector XML 을 STDOUT 로 출력할 때는 모두 한 행에 출력되어야 합니다. 위 예제 XML 은 읽기 쉽도록 여러 행으로 나눈 것입니다. STDOUT 에 한 행으로 출력하면 다음과 같이 보입니다.

<changeData dataSource="dataSource name"><resource name="resource1" value="resource1 value"/><resource name="resource2" value="resource2 value"/></changeData>

각 changeData 요소에는 dataSource 특성이 정의되어 있어야 합니다. 이 특성은 CA APM ChangeDetector 데이터를 볼 때 CA Introscope Workstation 에 표시되는 이름입니다. 각 changeData 요소는 특정 데이터 원본에 연결됩니다. 두 개 이상의 데이터 원본에 대한 CA APM ChangeDetector 데이터를 보고하려면 각 changeData 요소를 별도의 행에 출력해야 합니다.

changeData 요소에 포함될 수 있는 resource 요소의 수에는 제한이 없습니다. 각 resource 요소에는 두 가지 특성이 정의되어 있어야 합니다. name 특성은 보고하려는 리소스의 이름에 해당하고, value 특성은 보고하려는 리소스의 현재 값에 해당합니다.

changeData 요소를 출력할 때마다 모든 알려진 리소스의 현재 상태를 포함시켜야 합니다.

- 특정 데이터 원본에 대해 마지막으로 changeData 요소를 출력할 때 포함되지 않았던 리소스를 포함시킨 경우 해당 리소스는 시스템에 추가되는 것으로 간주됩니다.
- 두 실행 사이에 리소스의 value 특성이 변경되면 리소스가 수정된 것으로 간주됩니다.
- 특정 데이터 원본에 대해 마지막으로 changeData 요소를 출력할 때 포함되었던 리소스를 포함시키지 않은 경우 해당 리소스는 시스템에 삭제되는 것으로 간주됩니다.

선택적 Enterprise Manager 속성

IntroscopeEnterpriseManager.properties 에서 다음 속성을 설정할 수 있습니다.

- *introscope.changeDetector.storage.purge.maxDataAgeInDays*=<value>

  데이터 저장소에서 변경 사항을 삭제하기 전까지 보관하는 최대 기간(일)을 지정합니다. 기본값은 90 입니다.

- *introscope.changeDetector.storage.purge.offsetHour*=<value>

  제거(purge)를 사용하는 경우 실행할 시간을 지정합니다. 이 속성을 비워 두는 경우 기본값은 4(오전 4 시)입니다.

- *introscope.changeDetector.storage.perst.dbfile*=<path to storage file>

  ChangeDetector 데이터에 사용되는 저장소 파일의 위치를 지정합니다. 상대 경로 또는 완전하게 지정된 경로를 사용할 수 있습니다. 기본값은 *{$EM_INSTALL}/data/changes.db* 입니다.

- *introscope.changeDetector.jdbc.maxNumRows*=<value>

  처리하도록 허용되는 SQL 쿼리의 최대 행 수를 지정합니다. 수행된 쿼리는 Enterprise Manager 에 보고된 모든 변경 사항을 행 형식으로 표시하여 반환합니다. 변경 사항은 *changes.db* 파일에서 보고됩니다. 기본값은 10000 입니다.

제 3 장: CA APM ChangeDetector 데이터 보기

CA APM ChangeDetector 는 CA Introscope 에 직접 통합되어 응용 프로그램 환경의 변경 사항을 쉽게 모니터링할 수 있도록 CA Introscope 를 확장합니다.

CA APM ChangeDetector 가 설치되어 있으면 CA Introscope 에서 다음과 같은 변경 사항을 확인할 수 있습니다.

- CA Introscope 콘솔에서 CA APM ChangeDetector 대시보드에 변경 데이터의 종합적인 뷰가 표시됩니다.
- Workstation 에서 "변경" 탭에 계층적 트리 뷰나 테이블 뷰로 변경 데이터가 표시됩니다.
- Workstation 에서 변경 뷰어를 사용하여 어떤 변경 사항을 볼 것인지 지정하는 옵션을 설정할 수 있습니다.
- CA Introscope 의 메트릭 그래프와 대시보드에서 주석으로 변경 이벤트를 식별할 수 있습니다.
- CA Introscope 보고서에서 변경 기록을 확인할 수 있습니다.

참고: 모든 CA Introscope 에이전트가 Enterprise Manager 에 연결되기 전에 먼저 MOM 에 연결하면 Workstation 에 잘못된 변경 사항 수와 CA Introscope 에이전트 수가 표시됩니다. 몇 분 후에 다른 노드를 클릭하거나 다른 Investigator 를 열면 문제가 해결됩니다.

이 섹션은 다음 항목을 포함하고 있습니다.

CA Introscope 에서 변경 데이터 보기 (페이지 63)
차트 및 보고서에서 변경 데이터 보기 (페이지 76)

CA Introscope 에서 변경 데이터 보기

CA APM ChangeDetector 를 설치하고 구성할 때 모니터링할 CA Introscope 에이전트를 식별하고 CA APM ChangeDetector 에서 수집할 변경 데이터의 유형을 지정합니다. 또한 Workstation 을 사용하여 변경 데이터를 보고 보고서를 생성할 수 있습니다.

Workstation 의 다음 위치에서 변경 데이터를 볼 수 있습니다.

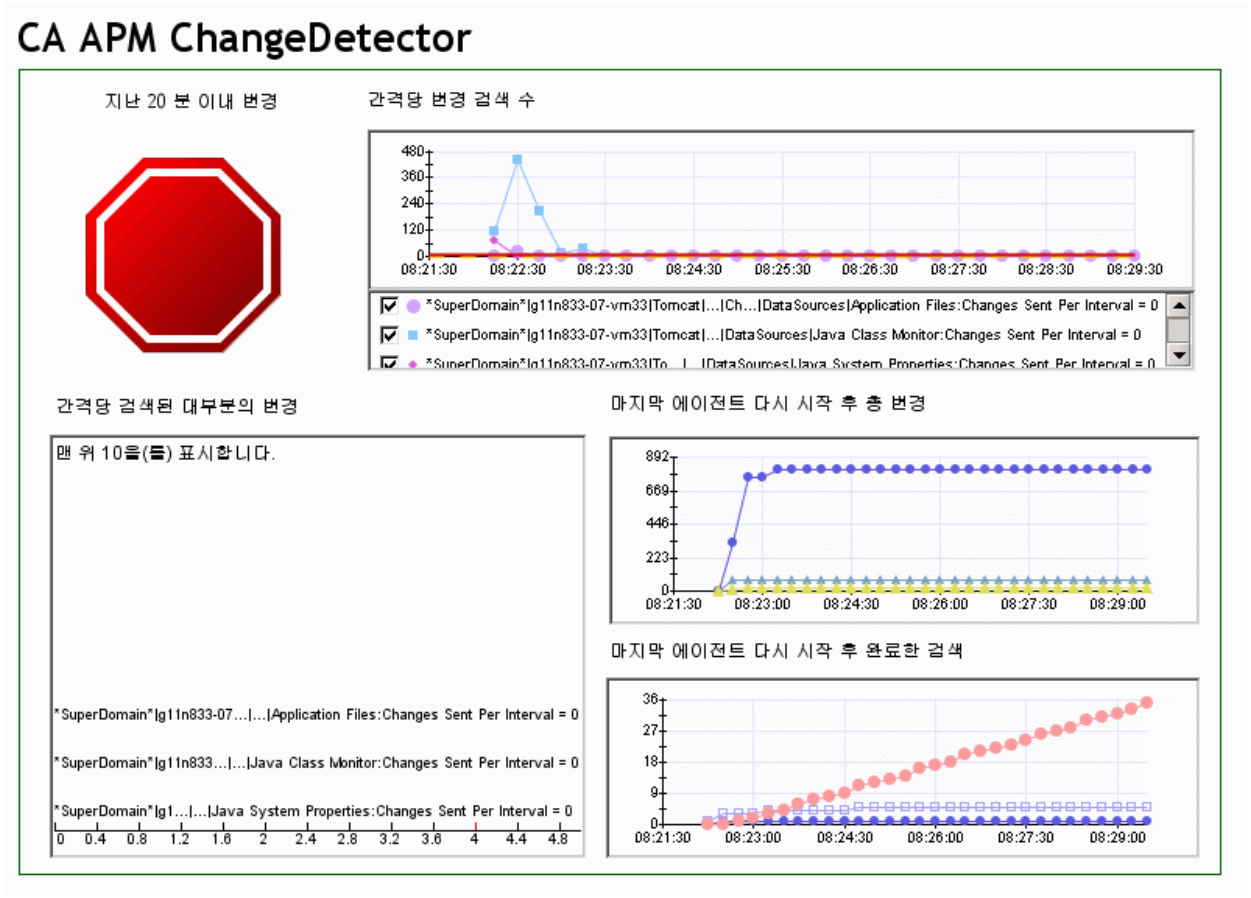
- CA APM ChangeDetector 대시보드 (페이지 65)
- Investigator 의 트리 뷰 (페이지 66)
- Investigator 의 테이블 뷰 (페이지 73) 트리 뷰에서 선택한 리소스에 대한 정보가 동일하게 테이블 뷰에도 표시되지만 그 형식은 다릅니다.

Investigator 의 "변경" 테이블에는 "Launch Change Viewer"(변경 뷰어 시작) 단추가 포함됩니다. 변경 뷰어에는 어떤 변경 데이터를 볼 것인지 세밀하게 제어할 수 있는 옵션이 있습니다.

참고: 모든 CA Introscope 에이전트가 Enterprise Manager 에 연결되기 전에 먼저 MOM 에 연결하면 Workstation 에 잘못된 변경 사항 수와 CA Introscope 에이전트 수가 표시됩니다. 몇 분 후에 다른 노드를 클릭하거나 다른 Investigator 를 열면 문제가 해결됩니다.

CA APM ChangeDetector 대시보드에서 변경 데이터 보기

CA Introscope 콘솔에는 변경 데이터의 종합적인 뷰가 표시되는 CA APM ChangeDetector 대시보드가 있습니다.



대시보드에는 변경 데이터에 대한 다음과 같은 뷰가 제공됩니다.

- 경고 표시기에는 지난 20 분 동안의 변경 사항이 표시됩니다.

  참고: CA APM ChangeDetector 는 지난 20 분 동안 변경 사항이 있었는지 여부를 확인합니다. 하지만 이것은 지난 20 분 동안의 *누적* 값이 아닙니다. 예를 들어 지난 20 분 동안 변경 사항 10 개가 있지만 각 변경 사항이 서로 다른 간격에 전송되었다면 노란색으로 표시될 수 있습니다. 한 간격에 5 개 이상의 변경 사항이 있는 경우에만 빨간색으로 표시됩니다.

- 간격당 검색된 변경 사항 수를 보여 주는 그래프와 CA Introscope 에이전트를 마지막으로 다시 시작한 이후의 변경 사항 수를 보여 주는 그래프가 있습니다.
- CA Introscope 에이전트를 마지막으로 다시 시작한 이후로 완료된 검색 수를 보여 주는 그래프가 있습니다.

CA APM ChangeDetector 열기

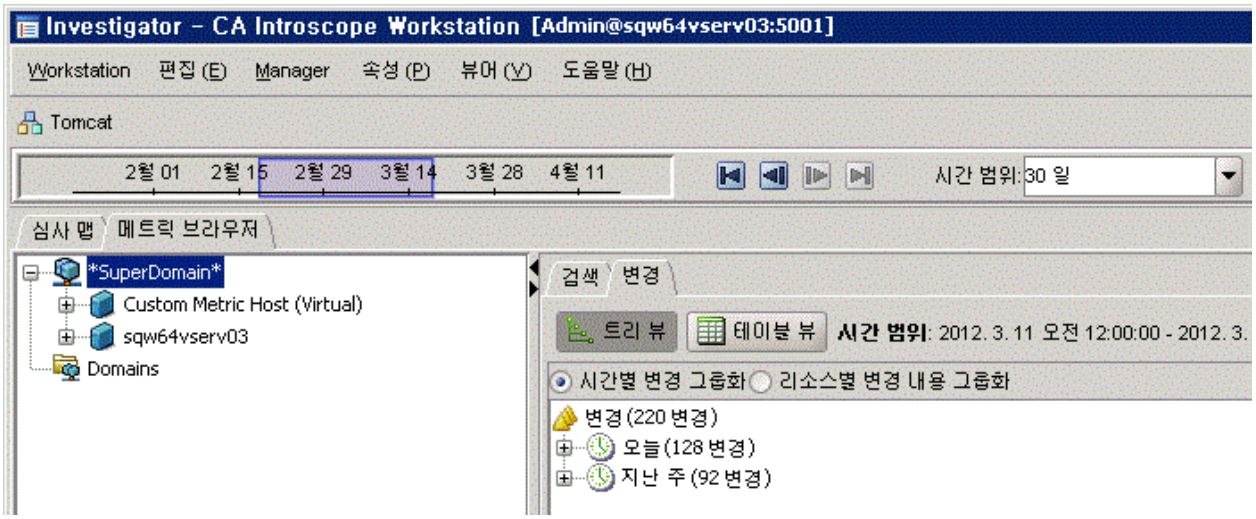
CA APM ChangeDetector 가 설치되어 있으면 CA Introscope Workstation 에 "변경" 탭이 나타납니다.

CA APM ChangeDetector 를 열려면

- 새 CA Introscope Investigator 를 열고 "변경" 탭을 클릭합니다. CA APM ChangeDetector 가 트리 뷰 모드에서 열립니다.

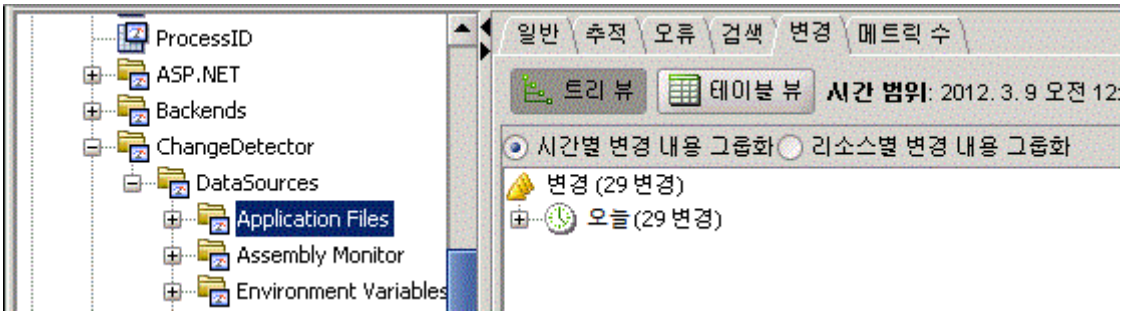
트리 뷰에서 변경 데이터 보기

Investigator 의 CA APM ChangeDetector 트래 뷰에는 *변경* 루트 노드에서 날짜와 시간순으로 그룹화된 변경 데이터가 계층적으로 표시됩니다. 트리 뷰가 기본 Investigator 뷰입니다.



특정 CA Introscope 에이전트의 변경 데이터를 표시할 수도 있습니다. 이렇게 하려면 CA Introscope Investigator 트리에서 해당 ChangeDetector 노드나 CA Introscope 에이전트의 ChangeDetector 데이터 원본 중 하나를 클릭하십시오.

ChangeDetector 데이터 원본이 중단된 경우 더 이상 "일반" 탭에 메트릭 데이터가 표시되지 않지만 "변경" 탭에는 변경 데이터가 계속 표시됩니다.

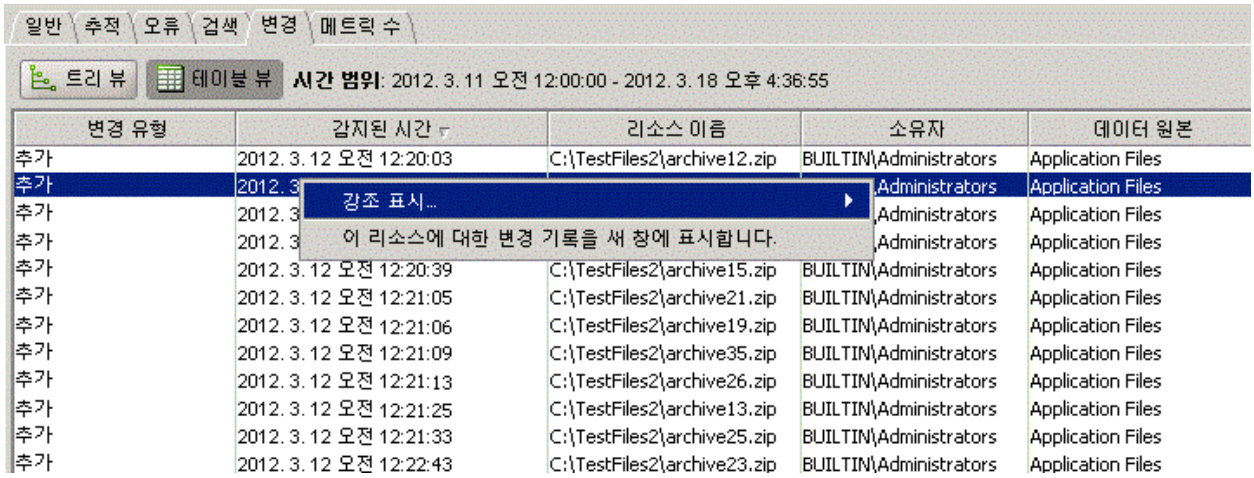


Introscope "변경" 탭에는 지난 주의 변경 데이터와 세션 중에 나타난 라이브 데이터가 표시됩니다. 다른 시간 범위를 선택하면 해당 시간 범위의 변경 데이터가 표시됩니다.

ChangeDetector 구성 매개 변수를 사용하여 변경 데이터를 표시하는 기본 시간 간격을 수정할 수 있습니다. 자세한 내용은 13 페이지의 *ChangeDetector 설치 및 구성*을 참조하십시오.

트리 뷰에서 시간이나 리소스를 기준으로 변경 데이터를 그룹화할 수 있습니다.

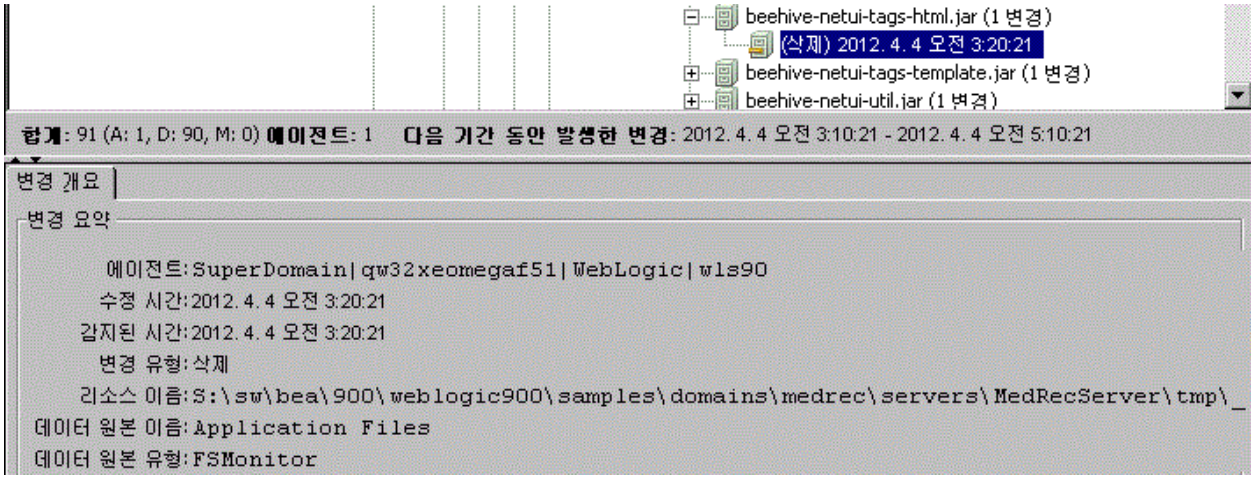
변경 사항 여러 개를 포함하는 노드를 클릭하면 아래쪽 창에 해당 노드 아래에 있는 모든 변경 이벤트가 표 형식으로 나열됩니다.



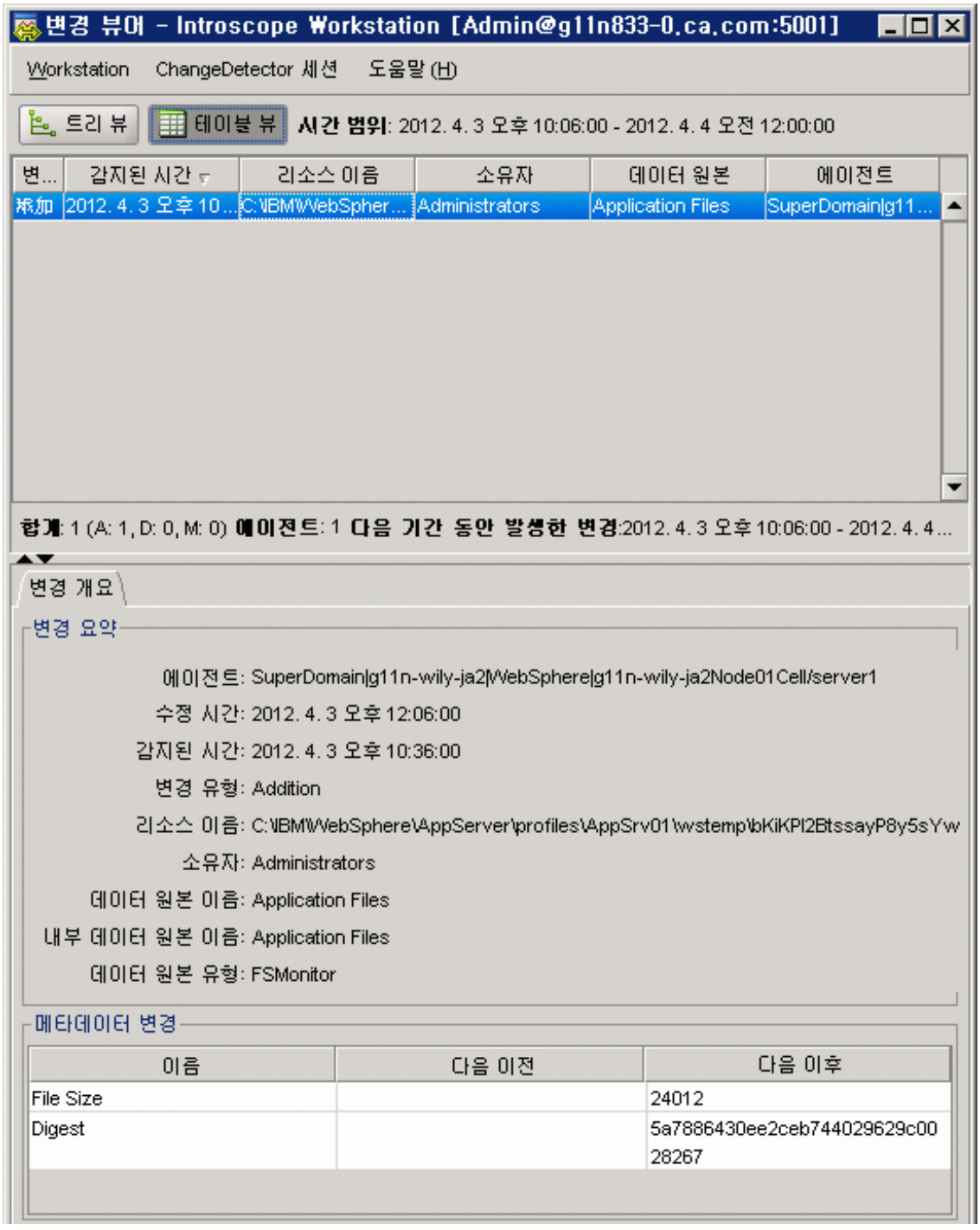
변경 유형	감지된 시간	리소스 이름	소유자	데이터 원본
추가	2012. 3. 12 오전 12:20:03	C:\TestFiles2\archive12.zip	BUILTIN\Administrators	Application Files
추가	2012. 3		Administrators	Application Files
추가	2012. 3		Administrators	Application Files
추가	2012. 3		Administrators	Application Files
추가	2012. 3. 12 오전 12:20:39	C:\TestFiles2\archive15.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:05	C:\TestFiles2\archive21.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:06	C:\TestFiles2\archive19.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:09	C:\TestFiles2\archive35.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:13	C:\TestFiles2\archive26.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:25	C:\TestFiles2\archive13.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:33	C:\TestFiles2\archive25.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:22:43	C:\TestFiles2\archive23.zip	BUILTIN\Administrators	Application Files

아래쪽 창에서 변경 이벤트를 마우스 오른쪽 단추로 클릭하고 다음과 같은 보기 옵션을 선택합니다.

- 테이블 강조 표시를 사용하여 특정 유형의 변경 사항을 보다 쉽게 확인하려면 "강조 표시"를 선택합니다(테이블 강조 표시 사용 (페이지 75) 참조).
- 변경 사항에 대한 트리 뷰를 열고 "변경 개요" 탭에 특정 변경 데이터를 표시하려면 "부모 뷰에서 이 변경 사항을 선택"을 클릭합니다.



- 선택한 리소스의 변경 기록을 표시하는 테이블 뷰를 별도의 창에서 열려면 "이 리소스에 대한 변경 기록을 새 창에 표시합니다"를 클릭합니다.



"변경 개요" 탭에 정보 표시

트리에서 단일 변경 사항을 선택할 때마다 "변경 개요" 탭이 열립니다. "변경 개요" 탭에는 항상 "요약" 패널이 나타나며, 필요에 따라 "MetaData"(메타데이터)와 "Details"(세부 정보) 패널이 나타날 수 있습니다.

- **변경 요약**

  단일 변경 사항을 선택할 때마다 "변경 개요" 탭에 나타납니다. 에이전트 ID, 수정 시간(파일에만 해당 - 아래 참고 참조), 감지된 시간, 변경 유형, 리소스 및 데이터 원본 이름, 데이터 원본 유형 등의 변경 사항에 대한 기본 정보가 이 요약에 표시됩니다.

- **메타데이터 변경**

  변경 데이터에 메타데이터가 포함된 경우 "변경 개요" 탭에 나타납니다. 예를 들어 파일 변경 사항의 메타데이터에는 마지막 수정 시간과 파일 크기가 포함됩니다.

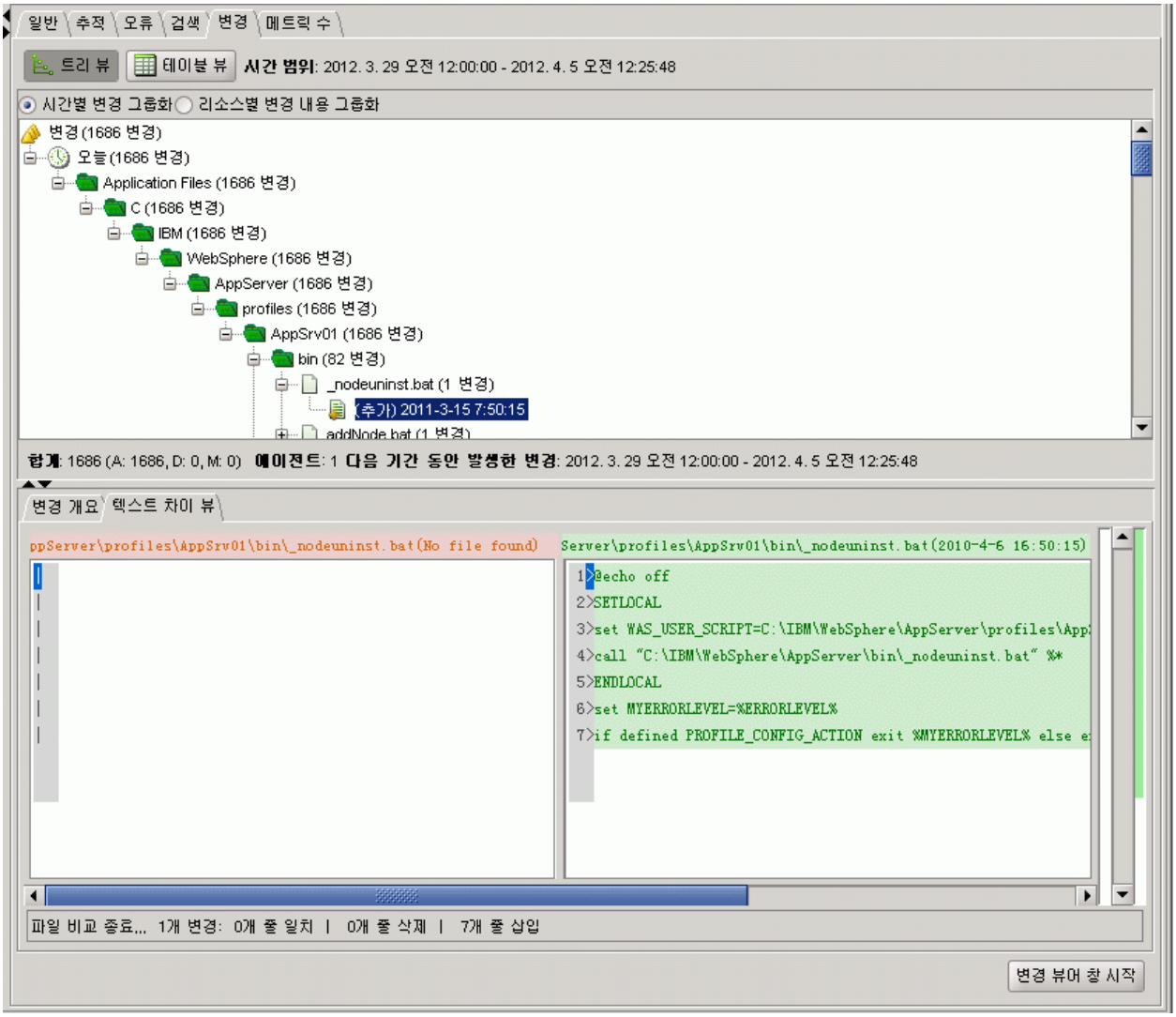
- **변경 세부 정보**

  이전/이후 테이블에 변경 세부 정보를 표시할 수 있는 경우에만 "변경 개요" 탭에 나타납니다. 예를 들어 환경 속성, 데이터베이스 특성 및 java 클래스와 같은 데이터 형식이 그에 해당합니다.

참고: 파일의 경우 선택적인 Workstation 속성이 설정되어 있으면(선택적 구성 속성 (페이지 59) 참조) CA APM ChangeDetector 가 변경 사항이 검색된 시간과 파일이 마지막으로 수정된 시간을 구분할 수 있습니다. CA APM ChangeDetector 지원 에이전트를 처음 시작한 경우 파일의 마지막 수정 시간이 라이브 시간 창(기본적으로 7 일) 이전이면 파일에 대한 추가 이벤트가 라이브 모드에서 나타나지 않을 수 있습니다. 하지만 "시간 범위" 드롭다운 목록에서 다른 시간 창을 선택하여 현재 기간에 표시되지 않는 데이터에 액세스할 수 있습니다.
경우에 따라 "감지된 시간"과 "수정 시간"의 시간 차이가 전체 검색에 걸리는 시간보다 길다면 CA APM ChangeDetector 에서 파일 변경 사항을 감지하는 데 전체 검색보다 더 오랜 시간이 걸릴 수 있습니다. 이 문제가 발생하는 이유는 여러 가지입니다. 예를 들어 파일 이름이 바뀌고 기존 파일을 덮어썼을 수 있습니다. CA APM ChangeDetector 에서는 이 작업을 덮어쓴 파일에 대한 수정 변경 이벤트로 인식하지만 파일의 마지막 수정 타임스탬프는 이름이 바뀌기 전의 타임스탬프가 계속 유지되기 때문에(운영 체제에 따라 다를 수 있음) 이러한 문제가 발생하게 됩니다. 또한 일부 운영 체제에서는 사용자가 시스템 유틸리티를 사용하여 파일의 마지막 수정 타임스탬프를 수정할 수 있으므로 이러한 문제가 발생할 수 있습니다.
수정 시간을 표시하도록 CA APM ChangeDetector 를 구성하려면 선택적 구성 속성 (페이지 59)을 참조하십시오.

"텍스트 차이 뷰" 탭에서 정보 보기

ChangeDetector "텍스트 차이 뷰" 탭을 사용하여 텍스트 파일 콘텐츠의 차이점을 식별할 수 있습니다.



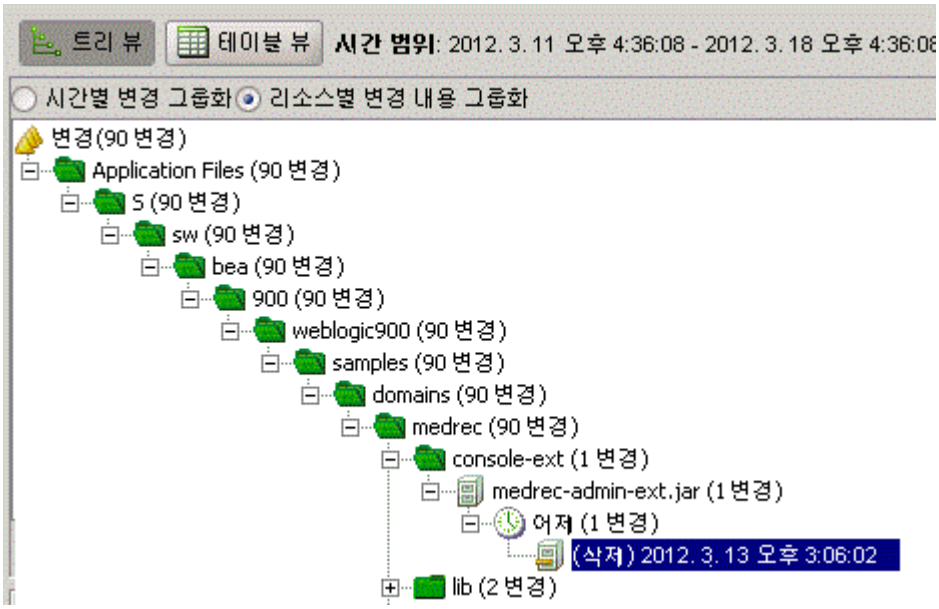
다음 조건을 충족하는 변경 사항을 선택하면 ChangeDetector 트리 뷰나 테이블 뷰에 "텍스트 차이 뷰" 탭이 나타납니다.

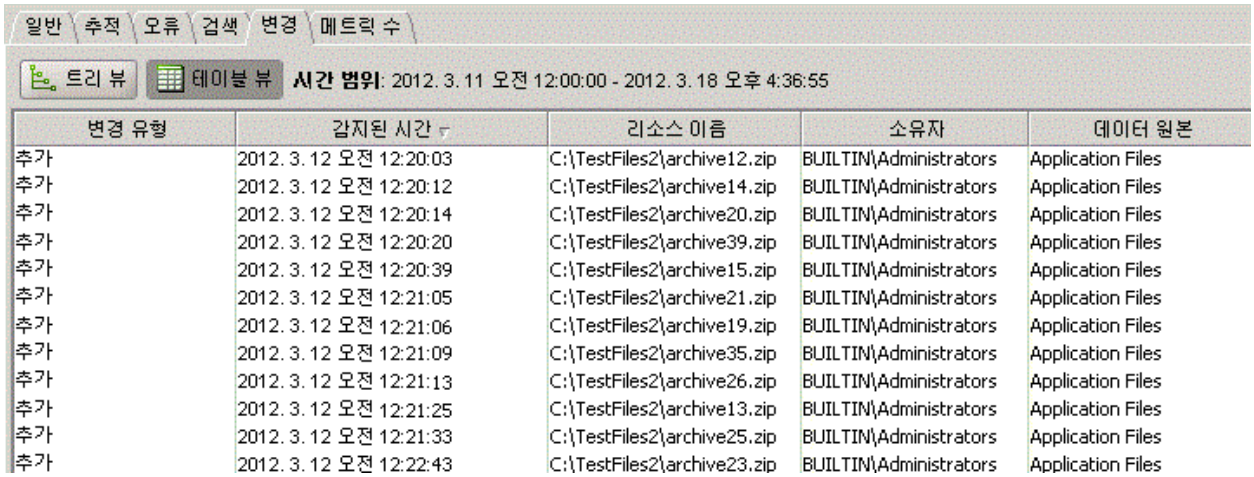
- 텍스트 파일(예: 구성 파일)
- 구성 파일(*ChangeDetector-config.xml*)에 정의된 *maxFileSizeToUpload* 속성 값보다 작은 파일 크기

크기가 maxFileSizeToUpload 값보다 큰 텍스트 파일은 Workstation "텍스트 차이 뷰"에 표시되지 않습니다.

테이블 뷰에서 변경 데이터 보기

Investigator 에서 리소스를 선택한 경우 트리 뷰나 테이블 뷰에서 리소스의 변경 데이터를 볼 수 있습니다. 트리 뷰에서 삭제나 수정과 같은 변경 이벤트를 선택하면 선택한 이벤트의 세부 정보를 표시하는 행이 테이블 뷰에서 강조 표시됩니다.



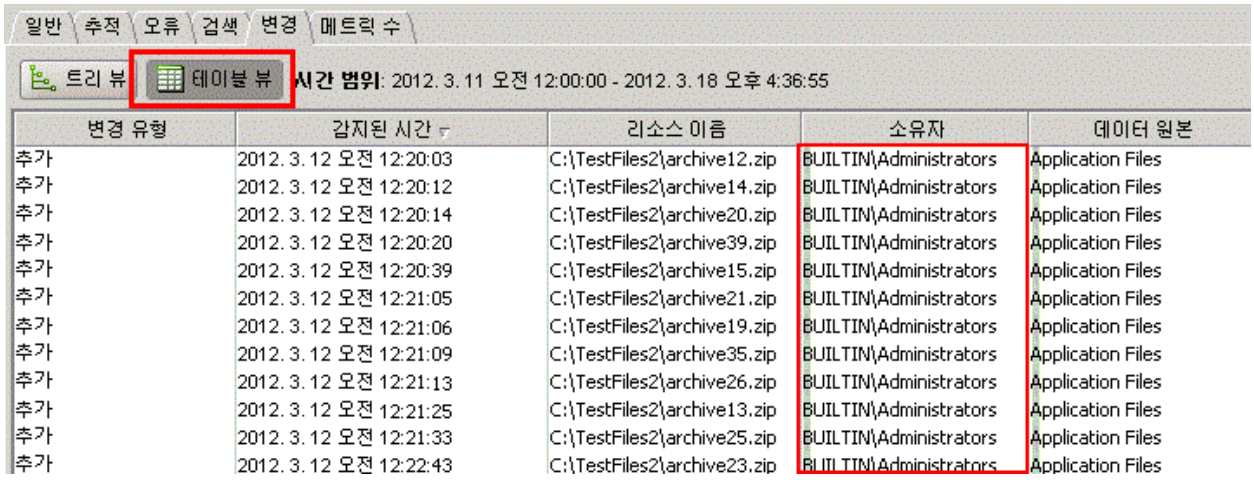


변경 유형	감지된 시간	리소스 이름	소유자	데이터 원본
추가	2012. 3. 12 오전 12:20:03	C:\TestFiles2\archive12.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:12	C:\TestFiles2\archive14.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:14	C:\TestFiles2\archive20.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:20	C:\TestFiles2\archive39.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:39	C:\TestFiles2\archive15.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:05	C:\TestFiles2\archive21.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:06	C:\TestFiles2\archive19.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:09	C:\TestFiles2\archive35.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:13	C:\TestFiles2\archive26.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:25	C:\TestFiles2\archive13.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:33	C:\TestFiles2\archive25.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:22:43	C:\TestFiles2\archive23.zip	BUILTIN\Administrators	Application Files

테이블 뷰를 사용하여 "변경 유형", "감지된 시간", "리소스 이름", "데이터 원본" 및 "에이전트 ID" 같은 다양한 축에 표시된 데이터를 정렬할 수 있습니다.

테이블 뷰에서 소유자 데이터 보기

CA APM ChangeDetector 는 파일 시스템 모니터 데이터 원본의 변경 사항을 커밋한 사용자를 식별합니다. 이 정보를 보려면 테이블 뷰에서 파일 시스템 데이터 원본을 표시하십시오. "소유자" 열에 사용자 ID 정보가 표시됩니다.

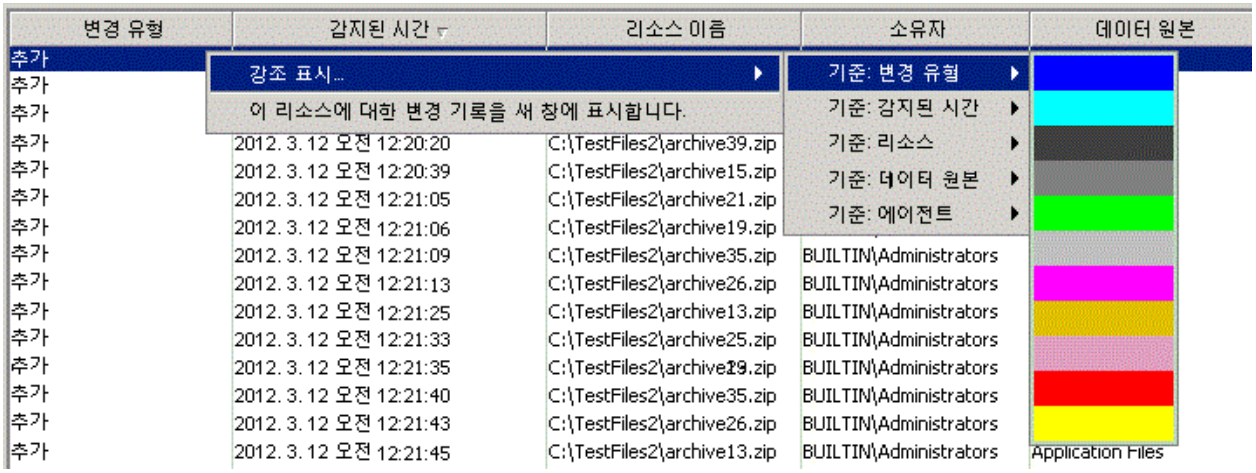


변경 유형	감지된 시간	리소스 이름	소유자	데이터 원본
추가	2012. 3. 12 오전 12:20:03	C:\TestFiles2\archive12.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:12	C:\TestFiles2\archive14.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:14	C:\TestFiles2\archive20.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:20	C:\TestFiles2\archive39.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:20:39	C:\TestFiles2\archive15.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:05	C:\TestFiles2\archive21.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:06	C:\TestFiles2\archive19.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:09	C:\TestFiles2\archive35.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:13	C:\TestFiles2\archive26.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:25	C:\TestFiles2\archive13.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:21:33	C:\TestFiles2\archive25.zip	BUILTIN\Administrators	Application Files
추가	2012. 3. 12 오전 12:22:43	C:\TestFiles2\archive23.zip	BUILTIN\Administrators	Application Files

다른 데이터 원본을 표시한 경우 테이블 뷰에 "소유자" 열이 아무 데이터 없이 나타납니다.

테이블 강조 표시 사용

특정 유형의 변경 데이터를 확인하기 쉽게 만들려면 색을 사용하여 테이블 뷰에서 변경 유형, 감지된 시간, 리소스, 데이터 원본 또는 에이전트별로 이벤트를 강조 표시할 수 있습니다.



형광펜을 적용하려면

1. 강조 표시할 특성이 있는 이벤트를 선택합니다. 예를 들어 특정 리소스가 있는 이벤트나 수정 이벤트를 선택합니다.
2. 이벤트를 마우스 오른쪽 단추로 클릭하고 "강조 표시"와 강조 표시할 특성(변경 유형, 감지된 시간, 리소스, 데이터 원본 또는 에이전트)을 선택합니다.

   "By Time Changed"(수정 시간별)를 선택한 경우 날짜 범위도 지정해야 합니다.
3. 이 형광펜의 색을 선택합니다.

   이제 선택한 특성이 있는 모든 이벤트가 강조 표시됩니다.

형광펜 작업

- 형광펜은 정의한 순서대로 우선 순위에 따라 적용되며 중첩되지 않습니다. 예를 들어 변경 유형을 기준으로 빨간색 형광펜을 정의한 경우 해당 변경 유형의 모든 이벤트가 빨간색으로 강조 표시됩니다. 그런 다음 리소스 기준으로 노란색 형광펜을 정의하면 빨간색으로 강조 표시된 이벤트는 색이 변하지 않습니다. 노란색 형광펜은 선택한 리소스와 일치하는 나머지 이벤트에 적용됩니다.

- 형광펜의 우선 순위 순서를 변경할 수 있습니다.
- 형광펜 각각을 개별적으로 제거하거나 한 번에 모두 제거할 수 있습니다.

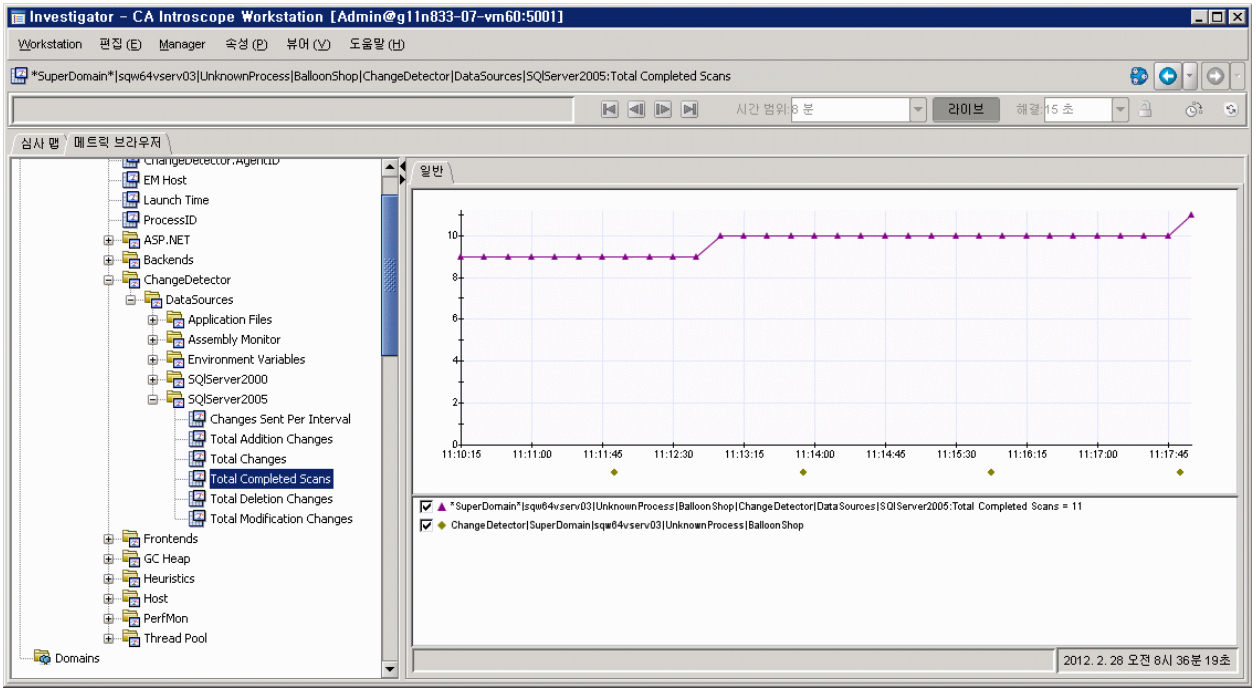
  이 예제의 "강조 표시" 메뉴를 살펴보십시오. 여기에 정의되어 있는 세 개의 형광펜은 리소스, 감지된 시간 및 데이터 원본 기준입니다.

차트 및 보고서에서 변경 데이터 보기

CA APM ChangeDetector 는 변경 데이터에 대한 정보를 CA Introscope 차트에 통합하며 지난 24 시간 동안 시스템에서 발생한 모든 변경 사항을 표시하는 보고서 템플릿을 포함하고 있습니다.

통합된 CA APM ChangeDetector 차트

CA APM ChangeDetector 는 변경 데이터에 대한 정보를 CA Introscope 차트에 통합합니다. 그림에서 볼 수 있는 것처럼, 변경 데이터는 차트의 X 축 아래에 차트 주석으로 나타나고 도구 설명으로도 나타납니다.



CA APM ChangeDetector 차트 주석 보기

CA APM ChangeDetector 차트 주석에는 범례에 매핑된 기호와 색이 사용되므로 변경 사항이 어디에서 발생했는지 쉽게 식별할 수 있습니다. 다음은 ChangeDetector 차트 주석의 몇 가지 기능입니다.

- 마우스 포인터로 주석을 가리키면 자세한 정보를 표시하는 도구 설명이 나타납니다.

  차트 주석 여러 개가 서로 지나치게 가깝게 배치되면 구분하기 어려울 수 있으므로 CA APM ChangeDetector 는 최대 다섯 개의 주석에 대한 도구 설명을 표시합니다. 이러한 도구 설명은 한 번에 모두 볼 수 있는 형태로 누적되어 표시됩니다. 주석이 다섯 개를 넘는 경우 창 맨 위에 주석의 총 개수를 알려 주는 메시지가 표시되고 처음 다섯 개 주석만 나타납니다.
- 차트 주석을 두 번 클릭하면 변경 뷰어가 열립니다. 이 변경 뷰어에는 주석 양쪽에 있는 두 그리드 선 사이에서 발생한 변경 사항이 나열되고 주석 자체에 해당하는 이벤트가 강조 표시됩니다.
- 차트에 나타나는 주석은 차트에 나타나는 메트릭에 따라 달라집니다. 예를 들어 차트의 메트릭이 CA Introscope 에이전트 A, B 및 C 에 있고 에이전트 A 및 B 만 CA APM ChangeDetector 를 지원하는 경우 지정된 기간에 대한 에이전트 A 및 B 의 변경 주석만 나타납니다.
- 지정된 기간 동안 변경 사항이 없으면 변경 데이터가 나타나지 않고 차트 아래쪽에 변경 주석이 표시되지 않습니다.

CA Introscope 차트에 대한 자세한 내용은 *CA APM Workstation 사용자 안내서*를 참조하십시오.

표시할 차트 주석 지정

차트에 표시할 주석을 지정할 수 있습니다. 예를 들어 특정 데이터 원본이나 리소스의 주석만 표시하거나, 한 CA Introscope 에이전트에서 수집된 변경 사항을 다른 CA Introscope 에이전트에서 수집된 메트릭과 상호 연관시킬 수 있습니다.

표시할 변경 사항은 다음 두 가지 방법으로 지정할 수 있습니다.

- Investigator 에서 "속성" > "Chart Annotation Settings"(차트 주석 설정)로 이동합니다.
- 차트를 마우스 오른쪽 단추로 클릭하고 "Chart Annotation Settings"(차트 주석 설정)를 선택합니다.

차트에서 주석을 지정하려면

1. "Chart Annotation Settings"(차트 주석 설정) 대화 상자를 엽니다.
2. "Default Settings"(기본 설정) 또는 "Use Custom Settings"(사용자 지정 설정 사용)을 선택합니다.

   Default Settings(기본 설정)

   차트에 메트릭을 표시하는 CA APM ChangeDetector 지원 에이전트에 대해서만 차트 주석을 표시합니다.

   예를 들어 차트의 메트릭이 CA Introscope 에이전트 A, B 및 C 에 있고 에이전트 A 및 B 만 CA APM ChangeDetector 를 지원하는 경우 지정된 기간에 대한 에이전트 A 및 B 의 변경 주석만 나타납니다. 차트에 에이전트 D 의 메트릭이 포함되어 있지 않으므로 에이전트 D 의 변경 사항은 CA APM ChangeDetector 지원 여부와 관계없이 차트에 나타나지 않습니다.

   Custom Settings(사용자 지정 설정)

   차트 주석으로 레이블을 지정할 CA APM ChangeDetector 구성 요소를 선택합니다.

   참고: 주석 옵션에 대한 변경 사항은 현재 차트에만 적용됩니다. 차트를 벗어나면 변경 사항이 유지되지 않습니다.

기본 제공 CA APM ChangeDetector 보고서 실행

CA APM ChangeDetector 는 지난 24 시간 동안 시스템에서 발생한 모든 변경 사항을 표시하는 보고서 템플릿을 포함하고 있습니다. 이 보고서는 CA Introscope 에이전트별로 그룹화되며 에이전트별 요약과 보고서 요약을 포함합니다.

지난 24 시간 동안 변경 보고서를 실행하려면

1. "Workstation" > "보고서 생성"을 선택하여 "보고서 템플릿 선택" 대화 상자를 엽니다.

   목록에 *지난 24 시간 동안 변경* 보고서 템플릿이 포함되어 있습니다.
2. "지난 24 시간 동안 변경"을 지정하고 "선택"을 클릭하여 "보고서 생성" 대화 상자를 엽니다.
3. 보고서의 시작 날짜와 종료 날짜를 지정합니다.

   참고: 보고서의 시간 범위는 보고서를 생성 중인 Workstation 의 표준 시간대에 따라 계산됩니다.

4. CA APM ChangeDetector 지원 에이전트 목록에서 에이전트를 선택하거나 다른 에이전트 표현식을 지정하여 템플릿 에이전트 표현식을 재정의합니다.
5. "미리 보기 생성"을 클릭하십시오.

   "미리 보기"에 보고서 제목 페이지가 표시됩니다.
6. "미리 보기" 단추를 사용하여 보고서 출력을 조작하고 보고서를 저장합니다.

CA Introscope 보고서에 CA APM ChangeDetector 요소 추가

기본 제공 *지난24 시간 동안 변경* 보고서 템플릿과 더불어 CA APM ChangeDetector 에는 CA APM ChangeDetector 요소를 추가하여 사용자 지정 보고서를 만들 수 있는 CA Introscope 보고 기능이 포함되어 있습니다.

사용자 지정 CA APM ChangeDetector 보고서를 만들려면

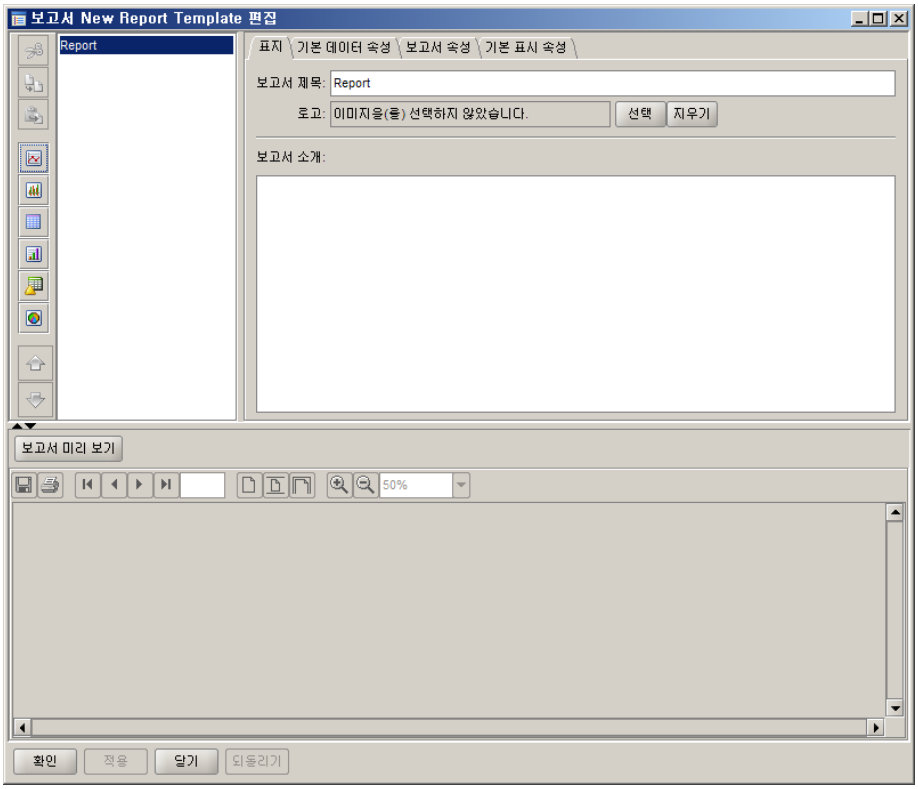
1. Workstation Investigator 에서 "파일" > "새 관리 모듈 편집기"를 선택합니다.
2. "요소" > "보고서 템플릿 새로 만들기"로 이동합니다.
3. 새 보고서 템플릿의 이름을 입력하고 보고서 이름이 고유하도록 "고유성 강제"를 클릭합니다.

   "고유성 강제"를 선택하면 CA Introscope 에서 사용자가 입력한 고유하지 않은 이름에 숫자를 붙여 고유한 이름으로 바꿉니다. 추가된 숫자는 보고서 템플릿이 생성된 후 이를 관리 모듈 편집기에서 볼 때 나타납니다. "고유성 강제"를 선택하지 않은 경우 동일한 보고서 템플릿 이름이 있으면 CA Introscope 는 오류 메시지를 표시하고 보고서를 생성하지 않습니다.
4. 보고서에 포함할 관리 모듈을 선택하고 "확인"을 클릭합니다.

   "보고서 템플릿 새로 만들기" 대화 상자가 열립니다.
5. "활성" 확인란을 클릭하여 보고서 템플릿을 활성화하여 CA Introscope 콘솔, Investigator 및 관리 모듈 편집기의 보고서 템플릿 목록에 나타나게 만듭니다.

6. 템플릿 편집기 열기를 클릭합니다.

   보고서 템플릿이 열리고 보고서 목차에 사용 가능한 옵션이 표시됩니다.



   참고: 변경 데이터에 대한 보고서 옵션만 포함하거나, 보고서에 메트릭 그래프와 함께 옵션을 추가하여 해당 보고서의 시간 프레임 동안 발생한 변경 사항을 표시할 수 있습니다.

7. "표지" 탭을 클릭하여 보고서의 용도를 지정합니다.

   보고서 제목

   생성된 보고서의 제목을 입력합니다. 제목은 제목 페이지에 목차와 함께 나타납니다.

   로고

   보고서와 연결할 로고를 선택합니다.

   보고서 소개

   보고서의 개요를 간략하게 설명하는 텍스트를 입력합니다.

8. "기본 데이터 속성" 탭을 클릭하여 속성을 지정합니다.

   기본적으로 데이터 속성은 수정할 수 없습니다. 하지만 파일의 실제 변경 시간 대신 변경이 검색된 시간을 표시하기 위해 "변경 시간 사용"을 해제할 수 있습니다.

   기본 시간 범위를 수정하려면 "템플릿 기본 시간 범위"에서 다음 필드를 변경하십시오.

   시작 시간 및 종료 시간

   시간 범위를 지정할 때 특정 시작 날짜와 종료 날짜를 지정하거나 *24 시간*과 같은 시간 기간을 지정할 수 있습니다.

   보고서의 시간 범위를 다음 방법 중 하나로 지정할 수 있습니다.

   a. 특정 시작 및 종료 날짜와 시간을 입력하거나 달력 아이콘을 클릭하여 시작 및 종료 날짜를 선택합니다.

   b. "시작 시간"을 비워 두고 "기간" 및 "단위" 매개 변수를 사용하여 보고서 실행 기간을 지정합니다.

   c. "종료 시간"을 비워 두고 "기간" 및 "단위" 매개 변수를 사용하여 보고서 실행 기간을 지정합니다.

   d. "종료 시간"에 현재를 입력하고 "기간" 및 "단위" 매개 변수를 사용하여 현재 시점을 기준으로 보고서를 실행할 과거 시간을 지정합니다.

   Duration(기간)

   숫자를 입력하여 보고서 실행 기간을 지정합니다. 이 숫자는 "단위" 값과 함께 적용됩니다. 예를 들어 "단위"가 *시간*인 경우 "기간"에 대해 *24* 를 입력할 수 있습니다.

   참고: "기간" 및 "단위" 매개 변수가 "시작 시간" 및 "종료 시간"과 함께 적용되는 방식에 대해서는 "시작 시간" 및 "종료 시간"의 설명을 참조하십시오.

   단위

   드롭다운 목록에서 시간 단위를 선택합니다. 설정은 *분*, *시간*, *일* 또는 *주*입니다.

9. "표시 속성" 탭을 클릭하여 보고서를 생성한 후 보고서의 그래프와 테이블이 표시될 방식을 결정하는 속성을 설정합니다.

   정렬 기준

   이 행을 클릭하여 목록을 열고 "타임스탬프", "데이터 원본", "리소스" 또는 "변경 유형" 중에서 선택합니다. 보고서의 모든 변경 사항은 처음에 에이전트 ID 별로 그룹화됩니다. "정렬 기준" 필드는 각 에이전트에서 보조 그룹입니다.

   정렬 순서

   이 행을 클릭하여 목록을 열고 "오름차순" 또는 "내림차순"을 선택합니다.

   최대 행 수

   보고서에 표시할 최대 행 수를 입력합니다.

보고서 작성 및 생성에 대한 자세한 내용은 *CA APM Workstation 사용자 안내서*를 참조하십시오.

제 4 장: CA APM ChangeDetector 메트릭

CA APM ChangeDetector 는 Enterprise Manager 및 CA Introscope 에이전트에 대한 지원 가능성 메트릭을 보고합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

Enterprise Manager 에 대한 CA APM ChangeDetector 지원 가능성 메트릭

Enterprise Manager 에 대한 지원 가능성 메트릭은 *Enterprise Manager|ChangeDetector|DataStore(데이터 저장소)* 노드에 있는 가상 에이전트 아래에서 찾을 수 있습니다.

이 메트릭에 대한 자세한 내용은 다음 항목을 참조하십시오.

Avg Time For Insertion (ms)(삽입 평균 시간(ms))

하나의 CA APM ChangeDetector 데이터 포인트 하나를 Enterprise Manager 측 데이터베이스에 삽입하는 데 걸리는 평균 시간(밀리초)입니다.

Datastore Used (%)(데이터 저장소 사용 백분율)

CA APM ChangeDetector 데이터를 저장하기 위해 할당된 데이터베이스 공간의 백분율입니다. 이 값이 100%에 도달한 후 다른 데이터 포인트를 삽입하려면 데이터베이스에 공간을 추가해야 합니다.

Number of Insertions(삽입 수)

데이터베이스에 삽입된 CA APM ChangeDetector 변경 사항의 수입니다.

Number of known agents(알려진 에이전트 수)

Enterprise Manager 가 데이터를 보고할 수 있는 CA APM ChangeDetector 지원 CA Introscope 에이전트의 수입니다. 이 값은 사용자 권한 및 에이전트 연결에 따라 달라질 수 있습니다.

수정 횟수(데이터베이스)

현재 데이터베이스에 저장된 CA APM ChangeDetector 변경 사항의 수입니다.

Size of Datastore(데이터 저장소 크기)

데이터 저장소의 바이트 단위 크기입니다.

참고: *"Avg Time For Insertion (ms)"*(삽입 평균 시간(ms)) 및 *"Number of Insertions"*(삽입 수) 메트릭 값은 Enterprise Manager 가 다시 연결되면 0 으로 재설정되므로 이러한 메트릭 값은 Enterprise Manager 수명 기간 동안에만 유효합니다.

수정 횟수(CA Introscope)

각 CA Introscope 에이전트에서 보낸 CA APM ChangeDetector 변경 사항의 수입니다.

CA Introscope 에 대한 CA APM ChangeDetector 지원 가능성 메트릭

CA APM ChangeDetector 는 검색이 완료되었거나 데이터 포인트가 전송된 경우 구성 중에 정의된 모든 데이터 원본에 대해 모든 에이전트의 CA Introscope 에이전트 지원 가능성 메트릭을 보고합니다. 모든 메트릭 총계는 CA Introscope 에이전트의 수명 기간 동안에만 유효합니다. 즉, CA Introscope 에이전트가 다시 연결되면 모든 카운터가 0 으로 재설정됩니다.

에이전트 지원 가능성 메트릭은 Workstation Investigator 트리의 ChangeDetector|데이터 원본 노드에서 볼 수 있습니다.

Changes Sent Per Interval(간격당 전송된 변경 사항 수)

CA Introscope 에서 간격당 전송한 변경 사항의 수입니다. 간격이 바뀔 때마다 카운터가 0 으로 재설정됩니다.

Total Addition Changes(추가 변경 사항의 총 수)

이 데이터 원본에 대해 CA Introscope 에서 보낸 총 추가 변경 사항 수입니다.

Total Completed Scans(완료된 검색의 총 수)

이 데이터 원본에 대해 완료된 검색의 수입니다.

Total Changes(변경 사항의 총 수)

이 데이터 원본에 대해 에이전트에서 보낸 총 변경 사항 수입니다.

Total Deletion Changes(삭제 변경 사항의 총 수)

이 데이터 원본에 대해 CA Introscope 에서 보낸 총 삭제 변경 사항 수입니다.

Total Modification Changes(수정 변경 사항의 총 수)

이 데이터 원본에 대해 CA Introscope 에서 보낸 총 수정 변경 사항 수입니다.

부록 A: 예제 구성 파일

CA APM ChangeDetector 설치에는 표준 구성 파일이 예제로 포함되어 있습니다.

구성 마법사를 사용하면 CA APM ChangeDetector 를 자신의 환경에 맞게 쉽게 구성 (페이지 19)할 수 있습니다.

사용자 환경에 맞게 수정된 구성 파일이 필요한 경우 *ChangeDetector-config.xml* 또는 *ChangeDetector-configDotnet.xml* 을 기반으로 사용자 지정 CA APM ChangeDetector 구성을 생성할 수 있습니다.

이 장에서는 사용자 지정 구성을 생성하는 방법에 대해 자세히 설명하고 사용자 지정 구성 파일의 예를 제공합니다.

이 섹션은 다음 항목을 포함하고 있습니다.

예제 Java ChangeDetector-config.xml 파일

다음은 가능한 Java 용 사용자 지정 CA APM ChangeDetector 구성의 예입니다.

참고: 다음 XML 예는 예제 데이터입니다. 이것은 현재의 예제 Java ChangeDetector-config.xml 파일 콘텐츠를 나타내거나 이전 버전의 CA APM ChangeDetector 콘텐츠를 포함할 수 있습니다.

```
<change-detector>
<!--
#############################################################
# Introscope ChangeDetector Configuration
#
# CA Wily Introscope(R) ChangeDetector Version 8.1
# Copyright (c) 2008 CA. All Rights Reserved.
# Introscope(R)는 CA 의 등록 상표입니다.
#############################################################
-->

    <!-- ======================================================= -->
    <!-- FILE CHANGE MONITORING                                  -->
    <!-- ======================================================= -->
    <!--
    검색 디렉터리 구성:
    반드시 수정해야 하는 유일한 구성 요소는
    <scan-directory> 요소의 'name' 특성으로,
    이 <datasource-instance> 요소의 맨 아래 부근에 있습니다.
    이 'name' 특성에는 ChangeDetector 로 검색하려는
    디렉터리의 경로를 입력해야 합니다. 일반적으로
    이 경로는 응용 프로그램 서버의 웹 응용 프로그램
    배포 디렉터리 및 구성 디렉터리입니다.

    파일 유형 사용자 지정:
    기본적으로 현재 작업 디렉터리에서 아래의
    <fileset> 요소로 정의된 파일 유형을 검색합니다.
    이러한 파일 유형은 변경할 경우 응용 프로그램의
    성능에 영향을 미칠 가능성이 높은 요소를 나타냅니다.
    특수한 파일 유형을 검색해야 하는 경우가 아니라면
    이 요소를 수정할 필요가 없습니다. -->
        <datasource-instance name="Application Files" type="file" version="8.1">

                <property name="explodeArchiveFiles" value="false" />

                <!-- Accepted units are hour, min, sec -->
                <property name="delayBetweenIterations" value="3" unit="sec" />

                <property name="filesPerIteration" value="5" />
```

```
                <property name="delayBetweenArchiveIterations" value="10"
unit="sec" />

                <property name="archiveFilesPerIteration" value="1" />

                <!-- Accepted units are bytes, KBytes, MBytes -->
                <property name="maxFileSizeToUpload" value="50" unit="KB" />

                <property name="useDigest" value="needed" />

                <!-- Scans typical files used for web application config -->
                <fileset name="config">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$"/>
                                <include pattern="(.*)\.cmd$"/>
                                <include pattern="(.*)\.sh$"/>
                                <include pattern="(.*)\.bat$"/>
                        </exclude>
                </fileset>

                <!-- Scans typical files that contain web application code -->
                <fileset name="webElements">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.jsp$"/>
                                <include pattern="(.*)\.cfm$"/>
                                <include pattern="(.*)\.js$"/>
                                <include pattern="(.*)\.html$"/>
                        </exclude>
                </fileset>

                <!-- Scans java archives -->
                <fileset name="archives">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.zip$"/>
                                <include pattern="(.*)\.jar$"/>
                                <include pattern="(.*)\.ear$"/>
                                <include pattern="(.*)\.war$"/>
                        </exclude>
                </fileset>

                <!-- Scans only the types of files defined in config, webElements,
and archive filesets -->
                <fileset name="default">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$" />
                                <include pattern="(.*)\.cmd$" />
                                <include pattern="(.*)\.sh$" />
                                <include pattern="(.*)\.bat$" />
                                <include pattern="(.*)\.jsp$" />
```

```
                                <include pattern="(.*)\.cfm$" />
                                <include pattern="(.*)\.js$" />
                                <include pattern="(.*)\.html$" />
                                <include pattern="(.*)\.zip$" />
                                <include pattern="(.*)\.jar$" />
                                <include pattern="(.*)\.ear$" />
                                <include pattern="(.*)\.war$" />
                        </exclude>
                </fileset>

                <!-- Scans only the types of files defined in config and webElements
-->
                <fileset name="defaultNoArchives">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$" />
                                <include pattern="(.*)\.cmd$" />
                                <include pattern="(.*)\.sh$" />
                                <include pattern="(.*)\.bat$" />
                                <include pattern="(.*)\.jsp$" />
                                <include pattern="(.*)\.cfm$" />
                                <include pattern="(.*)\.js$" />
                                <include pattern="(.*)\.html$" />
                        </exclude>
                </fileset>

                <!-- Scans everything except log files -->
                <fileset name="noLogs">
                        <exclude pattern="(.*)\.err$" />
                        <exclude pattern="(.*)\.log$" />
                        <exclude pattern="(.*)\.lok$" />
                        <exclude pattern="(.*)\.tlog$" />
                        <exclude pattern="(.*)\.log0(.*)" />
                </fileset>

        <!--
        name 특성의 값을 모니터링하려는 디렉터리를 가리키도록 변경
        참고: 즉시 사용 가능한 구성에서는 "default" 파일 집합을 사용합니다.  java 아카이브
파일을 모니터링하지 않으려면
        대신 "defaultNoArchive" 파일 집합을 사용하십시오. 또는, 선택한 파일을 모니터링하도록
fileset 요소를 사용자 지정하고
        아래의 scan-directory 요소를 바꾸거나 새로 추가할 수 있습니다.
        또한 name 특성(또는 다른 특성) 값으로 Java 시스템 속성이나 Introscope Agent 속성
값을
        사용할 수 있습니다. 예: name="${MY_APP_HOME}/filesICareAbout/"-->
                <scan-directory recursive="true" name="." fileset="default"
                        enabled="true">
                </scan-directory>
        </datasource-instance>
```

```
    <!-- ========================================================= -->
    <!-- DATABASE CHANGE MONITORING                                -->
    <!-- ========================================================= -->
    <!--
    DB 모니터에 대한 아래의 예제 설정에서는 Oracle v$parameter
    테이블에서 10 분마다 한 번씩 name/value 쌍을 검색합니다.

    참고: 이 데이터 원본 인스턴스에는
    사용자 환경에 적합한 연결 매개 변수가 필요하기 때문에
    기본적으로 주석으로 처리됩니다. 사용자의 데이터베이스 설정과 관련된 값을
    입력하십시오. -->
    <!--
    <datasource-instance name="Oracle DB" type="database"
driver="oracle.jdbc.driver.OracleDriver"
driverClasspath="C:\\somePathTo\\yourOracleDriver.zip"
                        url="jdbc:oracle:thin:@yourdbserver:1521:orcl"
username="username"
                        password="password" version="8.1">
                SQL Server
                        SELECT name, value FROM v$parameter
                </sql>
                <schedule type="repetitive" interval="10" unit="min"/>
        </datasource-instance>
        -->


    <!-- ========================================================= -->
    <!-- JAVA SYSTEM PROPERTIES MONITORING                         -->
    <!-- ========================================================= -->
    <!--
        기본적으로 모든 속성이 Java 시스템 속성 모니터에
    포함되고 모니터링됩니다.

    속성을 제외하려면 exclude 노드를 추가해야 합니다. 기존 제외 항목을
    재정의하려면 해당 exclude 요소 내부에 include 요소를
    중첩시켜야 합니다.  아래의 예제에서는 "windows."로 시작하는 속성을 예외로 하고
    모든 속성을 제외합니다.
                <exclude pattern=".*">
                        <include pattern="java\.*"/>
                </exclude> -->
        <datasource-instance name="Java System Properties" type="javaenv"
version="8.1">
        </datasource-instance>


    <!-- ========================================================= -->
    <!-- JAVA CLASS MONITOR                                        -->
    <!-- ========================================================= -->
```

```
        <!--
        기본적으로 응용 프로그램 서버 클래스는 아래의 exclude 요소를 사용하여 제외됩니다.
        exclude 요소의 pattern 특성은 네임스페이스를 포함한 클래스 이름의
        일치 여부를 확인하는 데 사용되는 정규식을 정의합니다.

        특정 제외 항목을 재정의하려면 해당 exclude 요소 내부에 include 요소를
        중첩시켜야 합니다.  아래의 예제에서는 네임스페이스 내부의 클래스를
        예외로 하고 모든 클래스를 제외합니다.
                    <exclude pattern=".*">
                            <include pattern="java\.*"/>
                    </exclude>

            참고:   현재 ChangeDetector는 JVM당 하나의 classmonitor 데이터 원본
                            인스턴스만 지원합니다.

            참고:   어떤 제외 패턴과도 일치하지 않는 모든 클래스가
                            Java 클래스 모니터에 포함되고 모니터링됩니다.

            -->
            <datasource-instance name="Java Class Monitor" type="classmonitor"
    version="8.1">
                    <!-- Accepted units are hour, min, sec -->
                    <property name="delayBetweenIterations" value="2" unit="sec" />
                    <property name="classesPerIteration" value="100" />

                    <!-- exclude classes from Wily -->
                    <exclude pattern="com\.wily\.(.*)"/>

                    <!--
                            아래에서는 일부 응용 프로그램 서버의 클래스를 건너뜁니다.
                            이러한 클래스를 포함시키면 거의 변경되지 않고
                            응용 프로그램 성능과 관련이 없는 많은 수의
                            클래스를 추적하게 됩니다.
                    -->

                    <!-- exclude classes from BEA -->
                    <exclude pattern="weblogic\.(.*)"/>
                    <exclude pattern="com\.bea\.(.*)"/>

                    <!-- exclude classes from IBM -->
                    <exclude pattern="com\.ibm\.(.*)">
                            <include pattern="com\.ibm\.(.*)jdbc(.*)"/>
                    </exclude>

                    <!-- exclude classes from SAP -->
                    <exclude pattern="com\.sap\.(.*)"/>

                    <!-- exclude classes from Oracle -->
                    <exclude pattern="oracle\.*">
```

```
                        <include pattern="oracle\.jdbc\.(.*)"/>
                    </exclude>

                    <!-- exclude classes from Sun -->
                    <exclude pattern="com\.sun\.enterprise\.(.*)"/>
            </datasource-instance>

            <!-- ========================================================== -->
        <!-- CONFIGURATION PROPERTIES - do not modify datasource-type's -->
        <!-- ========================================================== -->
            <datasource-type name="file"
class="com.wily.rave.agent.ds.file.FileDataSourceConfig" />
            <datasource-type name="database"
class="com.wily.rave.agent.ds.db.DBDataSourceConfig" />
            <datasource-type name="javaenv"
class="com.wily.rave.agent.ds.sysprop.SysPropDataSourceConfig" />
            <datasource-type name="classmonitor"
class="com.wily.rave.agent.ds.classmonitor.RuntimeClassMonitorConfig" />

</change-detector>
```

예제 .NET ChangeDetectorDotnet-config.xml 파일

다음은 가능한 .NET 용 사용자 지정 CA APM ChangeDetector 구성의 예입니다.

참고: 다음 XML 예는 예제 데이터입니다. 이것은 현재의 예제 .NET ChangeDetectorDotnet-config.xml 파일 콘텐츠를 나타내거나 이전 버전의 CA APM ChangeDetector 콘텐츠를 포함할 수 있습니다.

```
<change-detector>
<!--
############################################################
# Introscope ChangeDetector Configuration
#
# CA Wily Introscope(R) ChangeDetector Version 8.1
# Copyright (c) 2008 CA. All Rights Reserved.
# Introscope(R)는 CA의 등록 상표입니다.
############################################################
-->
    <!-- ======================================================= -->
    <!-- ENVIRONMENT VARIABLES MONITORING                        -->
    <!-- ======================================================= -->
    <!--
        기본적으로 모든 환경 변수가 환경 변수 모니터에
        포함되고 모니터링됩니다.

    변수를 제외하려면 exclude 노드를 추가해야 합니다. 기존 제외 항목을
    재정의하려면 해당 exclude 요소 내부에 include 요소를
    중첩시켜야 합니다.  아래의 예제에서는 "windows."로 시작하는 속성을 예외로 하고
    모든 속성을 제외합니다.
                <exclude pattern=".*">
                       <include pattern="windows\.*"/>
                </exclude> -->

     <datasource-instance name="Environment Variables" type="javaenv"
version="8.1">
        <exclude pattern="foo" />
        <exclude pattern=".*bar.*">
          <include pattern="hello" />
          <include pattern=".*world.*" />
        </exclude>
     </datasource-instance>

    <!-- ======================================================= -->
    <!-- FILE CHANGE MONITORING                                  -->
    <!-- ======================================================= -->
    <!--
    검색 디렉터리 구성:
    반드시 수정해야 하는 유일한 구성 요소는
```

```
    <scan-directory> 요소의 'name' 특성으로,
    이 <datasource-instance> 요소의 맨 아래 부근에 있습니다.
    이 'name' 특성에는 ChangeDetector 로 검색하려는
    디렉터리의 경로를 입력해야 합니다. 일반적으로
    이 경로는 응용 프로그램 서버의 웹 응용 프로그램
    배포 디렉터리 및 구성 디렉터리입니다.

    파일 유형 사용자 지정:
    기본적으로 현재 작업 디렉터리에서 아래의
    <fileset> 요소로 정의된 파일 유형을 검색합니다.
    이러한 파일 유형은 변경할 경우 응용 프로그램의
    성능에 영향을 미칠 가능성이 높은 요소를 나타냅니다.
    특수한 파일 유형을 검색해야 하는 경우가 아니라면
    이 요소를 수정할 필요가 없습니다. -->
        <datasource-instance name="Application Files" type="file" version="8.1">

                <property name="explodeArchiveFiles" value="false" />

                <!-- Accepted units are hour, min, sec -->
                <property name="delayBetweenIterations" value="3" unit="sec" />

                <property name="filesPerIteration" value="5" />

                <property name="delayBetweenArchiveIterations" value="10"
unit="sec" />

                <property name="archiveFilesPerIteration" value="1" />

                <!-- Accepted units are bytes, KBytes, MBytes -->
                <property name="maxFileSizeToUpload" value="50" unit="KB" />

                <property name="useDigest" value="needed" />

                <!-- Scans typical files used for web application config -->
                <fileset name="config">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$"/>
                                <include pattern="(.*)\.cmd$"/>
                                <include pattern="(.*)\.sh$"/>
                                <include pattern="(.*)\.bat$"/>
                        </exclude>
                </fileset>

                <!-- Scans typical files that contain web application code -->
                <fileset name="webElements">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.asp$"/>
                                <include pattern="(.*)\.asm$"/>
                                <include pattern="(.*)\.js$"/>
```

```
                                <include pattern="(.*)\.html$"/>
                        </exclude>
                </fileset>

                <!-- Scans archives -->
                <fileset name="archives">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.zip$"/>
                        </exclude>
                </fileset>

                <!-- Scans only the types of files defined in config, webElements,
and archive filesets -->
                <fileset name="default">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$" />
                                <include pattern="(.*)\.cmd$" />
                                <include pattern="(.*)\.sh$" />
                                <include pattern="(.*)\.bat$" />
                                <include pattern="(.*)\.asp$" />
                                <include pattern="(.*)\.asm$" />
                                <include pattern="(.*)\.js$" />
                                <include pattern="(.*)\.html$" />
                                <include pattern="(.*)\.zip$" />
                                <include pattern="(.*)\.profile$" />
                        </exclude>
                </fileset>

                <!-- Scans only the types of files defined in config and webElements
-->
                <fileset name="defaultNoArchives">
                        <exclude pattern="(.*)">
                                <include pattern="(.*)\.xml$" />
                                <include pattern="(.*)\.cmd$" />
                                <include pattern="(.*)\.sh$" />
                                <include pattern="(.*)\.bat$" />
                                <include pattern="(.*)\.asp$" />
                                <include pattern="(.*)\.asm$" />
                                <include pattern="(.*)\.js$" />
                                <include pattern="(.*)\.html$" />
                        </exclude>
                </fileset>

                <!-- Scans everything except log files -->
                <fileset name="noLogs">
                        <exclude pattern="(.*)\.err$" />
                        <exclude pattern="(.*)\.log$" />
                        <exclude pattern="(.*)\.lok$" />
                        <exclude pattern="(.*)\.tlog$" />
```

```
                <exclude pattern="(.*)\.log0(.*)" />
            </fileset>

        <!--
        scan-directory:
        name 특성의 값을 모니터링하려는 디렉터리를 가리키도록 변경
        참고: 즉시 사용 가능한 구성에서는 "default" 파일 집합을 사용합니다. 또는, 선택한 파일을
모니터링하도록 fileset 요소를
        사용자 지정하고 아래의 scan-directory 요소를 바꾸거나
        새로 추가할 수 있습니다. 또한 "name" 특성(또는 다른 특성) 값으로 환경 변수나 Introscope
Agent 속성 값을
        사용할 수 있습니다. 예제: name="${MY_APP_HOME}/filesICareAbout/"
        이 값은 응용 프로그램(에이전트 아님)의 작업 디렉터리에 상대적임을 의미하는
        점 하나(.)일 수 있으며, 그렇지 않은 경우 전체 경로여야 합니다. 아래 예제에서는 에이전트
설치에 일반적으로 사용되는 introscope8.1 폴더를
        에이전트 설치
        -->

            <scan-directory recursive="true" name="C:/Program Files/CA
Wily/Introscope8.1/wily" fileset="default"
                enabled="true">
            </scan-directory>
        </datasource-instance>


    <!-- ========================================================= -->
    <!-- ASSEMBLY MONITOR                                          -->
    <!-- ========================================================= -->
    <!--
    기본적으로 Windows 시스템 클래스는 아래의 exclude 요소를 사용하여 제외됩니다.
    exclude 요소의 pattern 특성은 네임스페이스를 포함한 클래스 이름의
    일치 여부를 확인하는 데 사용되는 정규식을 정의합니다.

    특정 제외 항목을 재정의하려면 해당 exclude 요소 내부에 include 요소를
    중첩시켜야 합니다.  아래의 예제에서는 네임스페이스 내부의 클래스를
    예외로 하고 모든 클래스를 제외합니다.
            <exclude pattern=".*">
                <include pattern="system\.*"/>
            </exclude>

        참고:   현재 ChangeDetector 는 어셈블리 모니터 데이터 원본의
                인스턴스 하나만 지원합니다.

        참고:   어떤 제외 패턴과도 일치하지 않는 모든 클래스가
                어셈블리 모니터에 포함되고 모니터링됩니다.

        -->
```

```
        <datasource-instance name="Assembly Monitor" type="classmonitor"
version="8.1">

                <!-- The initial wait on startup and when an assembly is loaded before
scanning -->
                 <!-- Accepted units are hour, min, sec -->
                <property name="initialWaitTime" value="30" unit="sec" />

                 <!-- Accepted units are hour, min, sec -->
                <property name="delayBetweenIterations" value="2" unit="min" />
                <property name="classesPerIteration" value="5" />

                <!--
                        아래에서는 시스템 어셈블리의 클래스를 건너뜁니다.
                        이러한 클래스를 포함시키면 거의 변경되지 않고
                        응용 프로그램 성능과 관련이 없는 많은 수의
                        클래스를 추적하게 됩니다.
                        모니터링할 필요가 없는 어셈블리를
                        <excludeassembly> 태그를 사용하여 추가합니다.
                -->

                <!-- exclude assemblies -->
                <excludeassembly pattern=".\\mscorlib\.dll"/>
                <excludeassembly pattern=".\\System\.dll"/>
                <excludeassembly pattern=".\\System\.Xml\.dll"/>
                <excludeassembly pattern=".\\System\.Web\.dll"/>
                <excludeassembly pattern=".\\System\.Configuration\.dll"/>
                <excludeassembly pattern=".\\wily\.."/>
                <excludeassembly pattern=".\\Microsoft\.JScript\.dll"/>
                <excludeassembly pattern=".\\VJSharpCodeProvider\.dll"/>
                <excludeassembly pattern=".\\System\.Data\.dll"/>
                <excludeassembly pattern=".\\Oracle\.DataAccess\.dll"/>
                <excludeassembly pattern=".\\System\.Web\.Mobile\.dll"/>
                <excludeassembly pattern=".\\System\.ServiceModel\.dll"/>
                <excludeassembly pattern=".\\SMDiagnostics\.dll"/>
                <excludeassembly pattern=".\\System\.Drawing\.dll"/>
                <excludeassembly
pattern=".\\System\.Web\.RegularExpressions\.dll"/>
                <excludeassembly pattern=".\\Microsoft\.VisualBasic\.dll"/>
                <excludeassembly pattern=".\\CppCodeProvider\.dll"/>
                <excludeassembly pattern=".\\System\.EnterpriseServices\.dll"/>
                <excludeassembly pattern=".\\System\.Transactions\.dll"/>

                <!-- exclude classes from Wily -->
                <!-- Add classes you wish to exclude in exclude patterns -->
                <exclude pattern="com\.wily\.(.*)"/>
```

```
        </datasource-instance>


    <!-- ========================================================= -->
    <!-- DATABASE CHANGE MONITORING                                -->
    <!-- ========================================================= -->
    <!--
    DB 모니터에 대한 아래의 예제 설정에서는 Oracle v$parameter
    테이블에서 10 분마다 한 번씩 name/value 쌍을 검색합니다.

    참고: 이 데이터 원본 인스턴스에는
    사용자 환경에 적합한 연결 매개 변수가 필요하기 때문에
    기본적으로 주석으로 처리됩니다. 사용자의 데이터베이스 설정과 관련된 값을
    입력하십시오. -->
<!--
       <datasource-instance name="SQl Server DB" type="database"
                            url="Provider=SQLOLEDB;Data Source=localhost;
Integrated Security=SSPI;Initial Catalog=northwind" version="8.1">
                SQL Server
                        SELECT name, value FROM sampletable
                </sql>
                <schedule type="repetitive" interval="10" unit="min"/>
        </datasource-instance> -->


        <!-- ========================================================= -->
        <!-- CONFIGURATION PROPERTIES - do not modify datasource-type's -->
        <!-- ========================================================= -->
        <datasource-type name="javaenv"
class="com.wily.rave.agent.ds.sysprop.SysPropDataSourceConfig" />
        <datasource-type name="file"
class="com.wily.rave.agent.ds.file.FileDataSourceConfig" />
        <datasource-type name="classmonitor"
class="com.wily.rave.agent.ds.classmonitor.RuntimeAssemblyMonitorConfig" />
        <datasource-type name="database"
class="com.wily.rave.agent.ds.db.DBDataSourceConfig" />

</change-detector>
```

부록 B: 질문과 대답

maxFileSizeToUpload 구성 파일 속성을 4 MB 로 설정했음에도 CA APM ChangeDetector 가 9 MB 아카이브 파일을 감지하고 해당 파일의 변경 사항을 보고했습니다. 이것은 버그입니까?

버그가 아닙니다. *maxFileSizeToUpload* 속성은 텍스트 파일에만 적용되며 아카이브 파일에는 적용되지 않습니다. CA APM ChangeDetector 는 전체 아카이브 파일의 콘텐츠를 업로드하는 대신 아카이브 파일을 열고 각 파일을 개별적으로 처리합니다.

응용 프로그램에서 실행되고 있는 코드에서 클래스를 삭제했지만 CA APM ChangeDetector 가 삭제를 감지하지 못했습니다. 이유가 무엇입니까?

CA APM ChangeDetector 는 코드의 추가 및 수정만 감지하며, 코드 삭제는 감지하지 않습니다.

CA APM ChangeDetector 가 모니터링되는 파일 시스템에서 크기가 0 인 파일을 표시하는 이유는 무엇입니까?

모니터링되는 파일 시스템에 읽기 권한이 없거나, 해당 파일 시스템이 네트워크 폴더에 있는 경우 네트워크가 다운된 것일 수 있습니다.