Sun Java System Message Queue 3.7 UR1 기술 개요

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

부품 번호: 820–0532
2007년 2월

Copyright 2007 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. 모든 권리는 저작권자의 소유입니다.

Sun Microsystems, Inc.는 이 문서에 설명된 제품의 기술 관련 지적 재산권을 소유합니다. 특히 이 지적 재산권에는 하나 이상의 미국 특허권이 포함될 수 있으며, 미국 및 다른 국가에서 하나 이상의 출원 중인 특허권이 제한 없이 포함될 수 있습니다.

미국 정부의 권리 – 상용 소프트웨어정부 사용자는 Sun Microsystems, Inc.의 표준 사용권 계약과 해당 FAR 규정 및 보충 규정을 준수해야 합니다.

이 배포판에는 타사에서 개발한 자료가 포함되어 있을 수 있습니다.

제품 중에는 캘리포니아 대학에서 허가한 Berkeley BSD 시스템에서 파생된 부분이 포함되어 있을 수 있습니다. UNIX는 미국 및 다른 국가에서 X/Open Company, Ltd.를 통해 독점적으로 사용권이 부여되는 등록 상표입니다.

Sun, Sun Microsystems, Sun 로고, Solaris 로고, Java Coffee Cup 로고, docs.sun.com, Java 및 Solaris는 미국 및 다른 국가에서 Sun Microsystems, Inc.의 상표 또는 등록 상표입니다. 모든 SPARC 상표는 사용 허가를 받았으며 미국 및 다른 국가에서 SPARC International, Inc.의 상표 또는 등록 상표입니다. SPARC 상표를 사용하는 제품은 Sun Microsystems, Inc.에서 개발한 구조에 기반을 두고 있습니다.

OPEN LOOK 및 Sun™ 그래픽 사용자 인터페이스(GUI)는 Sun Microsystems, Inc.가 자사의 사용자 및 정식 사용자용으로 개발했습니다. Sun은 컴퓨터 업계를 위한 시각적 또는 그래픽 사용자 인터페이스(GUI)의 개념을 연구 개발한 Xerox사의 선구적인 노력을 높이 평가하고 있습니다. Sun은 Xerox와 Xerox 그래픽 사용자 인터페이스(GUI)에 대한 비독점적 사용권을 보유하고 있습니다. 이 사용권은 OPEN LOOK GUI를 구현하는 Sun의 정식 사용자에게도 적용되며 그렇지 않은 경우에는 Sun의 서면 사용권 계약을 준수해야 합니다.

이 설명서에 포함된 정보 및 이 설명서에서 다루는 제품은 미국 수출 관리법이 적용되며 다른 국가의 수출 또는 수입 관리법의 적용을 받을 수 있습니다. 이 제품과 정보를 직간접적으로 핵무기, 미사일 또는 생화학 무기에 사용하거나 핵과 관련하여 해상에서 사용하는 것은 엄격하게 금지합니다. 미국 수출 금지 국가 또는 금지된 개인과 특별히 지정된 국민 목록을 포함하여 미국 수출 금지 목록에 지정된 대상으로의 수출이나 재수출은 엄격하게 금지됩니다.

설명서는 "있는 그대로" 제공되며, 법률을 위반하지 않는 범위 내에서 상품성, 특정 목적에 대한 적합성 또는 비침해에 대한 묵시적인 보증을 포함하여 모든 명시적 또는 묵시적 조건, 표현 및 보증을 배제합니다.

070428@17039

목차

머리말 7

1 메시징 시스템: 소개 17
메시지 지향 미들웨어(Message-Oriented Middleware(MOM)) 17
MOM 표준으로서의 JMS 22
JMS 메시징 객체 및 패턴 22
관리 대상 객체 24
▼ 관리 대상 객체를 대상으로 사용하는 방법 25
Message Queue: 요소 및 기능 26
Message Queue 서비스 26
브로커에 연결 27
브로커 28
클라이언트 런타임 지원 29
관리 30
Message Queue 서비스 확장 30
활성화 기술로서의 Message Queue 31
Message Queue 기능 요약 32

2 클라이언트 프로그래밍 모델 33
설계 및 성능 33
메시징 도메인 34
지점간 메시징 34
게시/가입 메시징 36
도메인별 API 및 통합 API 38
프로그래밍 객체 39
연결 팩토리 및 연결 40
세션 41

메시지 42
메시지 생성 44
메시지 소비 45
동기식 및 비동기식 소비자 45
선택기를 사용하여 메시지 필터링 45
영구 가입자 사용 46
요청-응답 패턴 46
안정적인 메시징 47
확인 48
트랜잭션 49
영구 저장소 50
시스템에서의 메시지 경로 50
메시지 생성 51
메시지 처리 및 경로 지정 51
메시지 소비 52
메시지 수명 끝 52
SOAP 메시지 작업 52
▼ 안정적인 SOAP 메시징을 수행하는 방법 53
Java 및 C 클라이언트 53

3 Message Queue 서비스 55
구성 요소 서비스 55
연결 서비스 57
대상 및 라우팅 서비스 58
지속성 서비스 61
보안 서비스 62
모니터링 서비스 64
관리 도구 및 작업 66
관리 도구 66
개발 환경 지원 68
프로덕션 환경 지원 69
메시징 작업 확장 70

4 브로커 클러스터 71
클러스터 구조 71

메시지 전달 72
대상 속성 73
클러스터링 및 대상 74
클러스터 구성 78
클러스터 동기화 79

5 Message Queue와 J2EE 81
JMS/J2EE 프로그래밍: Message-Driven Bean 81
J2EE Application Server 지원 83
JMS 자원 어댑터 83

A Message Queue의 선택적 JMS 기능 구현 85
선택적 기능 85

B Message Queue 기능 87
기능 목록 88

용어집 103

색인 107

머리말

이 **Sun Java System Message Queue 3.7 UR1 기술 개요** 설명서에서는 Message Queue 메시징 서비스의 기술, 개념, 구조, 기능 및 특징을 소개합니다.

따라서 이 설명서는 Message Queue 설명서 세트에 있는 다른 문서의 기초가 되며 먼저 읽어 보는 것이 좋습니다.

대상

이 안내서는 응용 프로그램 개발자, 관리자를 비롯하여 Message Queue 제품을 사용하거나 그 기술, 개념, 구조, 기능 및 특징을 이해하려는 사용자를 대상으로 합니다.

- 응용 프로그램 개발자는 Message Queue 서비스를 사용하여 다른 클라이언트 응용 프로그램과 메시지를 교환하는 Message Queue 클라이언트 응용 프로그램의 작성을 담당합니다.
- Message Queue 메시징 서비스를 설정하고 관리하는 것은 관리자의 책임입니다. 이 문서는 메시징 시스템에 대한 지식이나 이해를 전제로 하지 않습니다.

이 문서는 Message Queue 서비스로 구현되는 JMS(Java Message Service) 사양에 대한 지식을 전제로 하지 않습니다.

알아야 할 사항

이 문서를 읽기 위한 전제 조건은 없습니다. Message Queue 개발 및 관리 설명서를 읽기 전에 이 책을 읽어 Message Queue의 기본적인 개념을 이해하십시오.

구성

이 설명서는 처음부터 끝까지 읽도록 되어 있으며, 각 장은 이전 장에 수록된 내용을 바탕으로 합니다. 다음 표에서는 각 장의 내용을 간단히 설명합니다.

표 P–1 책의 내용과 구성

장	설명
1 장	메시징 미들웨어 기술에 대해 소개하며, JMS 표준 그리고 Message Queue 서비스에서의 해당 표준 구현에 대해 설명합니다.
2 장	JMS 프로그래밍 모델 및 Message Queue 클라이언트 런타임을 사용하여 JMS 클라이언트를 만드는 방법에 대해 설명합니다. C++ 클라이언트 및 SOAP 메시지 전송에 대한 런타임 지원을 설명합니다.
3 장	관리 작업 및 도구를 소개하며 연결, 라우팅, 지속성, 보안 및 모니터링의 구성에 사용되는 브로커 서비스에 대해 설명합니다.
4 장	Message Queue 브로커 클러스터의 구조 및 사용에 대해 설명합니다.
5 장	J2EE 플랫폼 환경에서 JMS 지원을 구현한 결과를 살펴봅니다.
부록 A	Message Queue 제품이 JMS 선택 항목을 처리하는 방법을 설명합니다.
부록 B	Message Queue 기능을 차례로 소개하고 이러한 기능을 구현하는 데 필요한 단계를 요약하며 자세한 내용을 위한 참조 정보를 제공합니다.
용어집	Message Queue 사용 중에 접할 수 있는 용어와 개념에 대한 정보를 제공합니다.

관련 문서

Message Queue 설명서 세트를 구성하는 문서는 다음 표에 일반적으로 사용되는 순서에 따라 나열되어 있습니다.

표 P–2 Message Queue 설명서 세트

문서	대상	설명
Sun Java System Message Queue 3.7 UR1 Installation Guide	개발자와 관리자	Solaris, Linux 및 Windows 플랫폼에서 Message Queue 소프트웨어를 설치하는 방법을 설명합니다.

표 P–2 Message Queue 설명서 세트 *(계속)*

문서	대상	설명
Sun Java System Message Queue 3.7 UR1 릴리스 노트	개발자와 관리자	새로운 기능, 제한, 알려진 버그 및 기술 노트에 관한 설명이 포함되어 있습니다.
Sun Java System Message Queue 3.7 UR1 기술 개요	개발자와 관리자	Message Queue 개념, 기능 및 구성 요소에 대해 설명합니다.
Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients	개발자	Message Queue의 JMS 또는 SOAP/JAXM 사양 구현을 사용하는 Java 클라이언트 프로그램 개발자를 위한 빠른 시작 자습서와 프로그래밍 정보를 제공합니다.
Sun Java System Message Queue 3.7 UR1 관리 설명서	관리자, 개발자에게도 권장	Message Queue 관리 도구를 사용한 관리 작업 수행 시 필요한 배경 및 정보를 제공합니다.
Sun Java System Message Queue 3.7 UR1 Developer's Guide for C Clients	개발자	Message Queue에서 C로 구현된 JMS API(C-API)를 사용하는 C 클라이언트 프로그램 개발자를 위한 프로그래밍 및 참조 설명서를 제공합니다.

온라인 도움말

Message Queue 3.7 UR1는 Message Queue 메시지 서비스 관리 작업을 수행하기 위한 명령줄 유틸리티를 포함하고 있습니다.

또한 Message Queue 3.7 UR1 에는 그래픽 사용자 인터페이스(GUI) 관리 도구인 관리 콘솔(`imqadmin`)이 포함되어 있습니다. 상황에 맞는 도움말이 관리 콘솔에 포함되어 있습니다. **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 "Administration Console Online Help"을 참조하십시오.

JavaDoc

JavaDoc 형식의 JMS 및 Message Queue API 설명서는 다음 위치에 있습니다.

플랫폼	위치
Solaris	`/usr/share/javadoc/imq/index.html`
Linux	`/opt/sun/mq/javadoc/index.html`
Windows	`IMQ_HOME/javadoc/index.html`

이 설명서는 모든 HTML 브라우저에서 볼 수 있습니다. 표준 JMS API 설명서를 비롯하여 메시징 응용 프로그램 개발자에게 유용한, Message Queue 관리 대상 객체를 위한 Message Queue 전용 API에 대한 내용도 포함하고 있습니다.

클라이언트 응용 프로그램의 예

Message Queue에서는 개발자에게 도움이 되도록 다양한 클라이언트 응용 프로그램의 예를 제공합니다.

Java 클라이언트 응용 프로그램의 예

Java 클라이언트 응용 프로그램의 예는 플랫폼별로 다음 디렉토리에 있습니다. 이 디렉토리와 각 하위 디렉토리에 있는 README 파일을 참조하십시오.

플랫폼	위치
Solaris	/usr/demo/imq/
Linux	/opt/sun/mq/examples
Windows	IMQ_HOME/demo/

C 클라이언트 프로그램의 예

C 클라이언트 응용 프로그램의 예는 플랫폼별로 다음 디렉토리에 있습니다. 이 디렉토리에 있는 README 파일을 참조하십시오.

플랫폼	위치
Solaris	/opt/SUNWimq/demo/C/
Linux	/opt/sun/mq/examples/C/
Windows	IMQ_HOME/demo/C/

JMS(Java Message Service) 사양

다음 위치에서 JMS 사양을 확인할 수 있습니다.

(http://java.sun.com/products/jms/docs.html)

사양에는 샘플 클라이언트 코드가 포함되어 있습니다.

디렉토리 변수 규칙

Message Queue에서는 세 가지 디렉토리 변수를 활용합니다. 그 설정 방법은 플랫폼별로 달라집니다. 표 P–3에서는 이러한 변수를 소개하고 이들이 Solaris, Linux 및 Windows 플랫폼에서 어떻게 사용되는지를 설명합니다.

주 – 표 P–3의 정보는 Message Queue의 독립 실행형 설치에만 적용됩니다. Message Queue가 설치되어 Application Server 설치의 일부로 실행될 경우, 디렉토리 변수의 값이 다르게 설정됩니다. `IMQ_HOME`이 *appServer_install_dir*`/imq`로 설정됩니다(여기서 *appServer_install_dir*은 Application Server 설치 디렉토리). 그리고 `IMQ_VARHOME`은 *appServer_domainName_dir* `/imq`로 설정됩니다(여기서 *appServer_domainName_dir*은 Message Queue 브로커를 시작하는 도메인의 도메인 디렉토리).

표 P–3 디렉토리 변수 규칙

변수	설명
`IMQ_HOME`	Message Queue 설명서에서 Message Queue 기본 디렉토리(루트 설치 디렉토리)를 가리킬 때 사용합니다. ■ Solaris 및 Linux에는 루트 Message Queue 설치 디렉토리가 없습니다. 따라서 Message Queue 설명서에서 `IMQ_HOME`은 Solaris 및 Linux에서의 파일 위치를 가리키지 않습니다. ■ Windows의 경우, 루트 Message Queue 설치 디렉토리는 Message Queue 번들의 압축을 푼 디렉토리로 설정됩니다.
`IMQ_VARHOME`	Message Queue에서 임시 또는 동적으로 작성된 구성 및 데이터 파일이 저장되는 `/var` 디렉토리입니다. 임의의 디렉토리를 가리키는 환경 변수로 설정될 수 있습니다. ■ Solaris에서 `IMQ_VARHOME`의 기본값은 `/var/imq` 디렉토리입니다. ■ Solaris에서 Sun Java System Application Server 평가판의 `IMQ_VARHOME` 기본값은 `IMQ_HOME/var` 디렉토리입니다. ■ Linux에서 `IMQ_VARHOME`의 기본값은 `/var/opt/sun/mq` 디렉토리입니다. ■ Windows에서 `IMQ_VARHOME`의 기본값은 `IMQ_HOME/var` 디렉토리입니다.

표 P–3 디렉토리 변수 규칙 *(계속)*

변수	설명
IMQ_JAVAHOME	Message Queue 실행 가능에 필요한 Java runtime(JRE)의 위치를 가리키는 환경 변수입니다. ■ Solaris에서 IMQ_JAVAHOME은 최신 버전의 JDK를 찾지만, 사용자가 선호하는 JRE의 위치로 값을 설정할 수도 있습니다. ■ Linux에서는 Message Queue가 맨 처음 최신 버전의 JDK를 찾지만, 사용자가 선호하는 JRE의 위치로 IMQ_JAVAHOME의 값을 설정할 수도 있습니다. ■ Windows에서는 기존 Java 런타임을 가리키도록 IMQ_JAVAHOME이 설정됩니다(시스템에 지원되는 버전이 있는 경우). 지원되는 버전이 없으면 설치됩니다.

이 설명서에서 IMQ_HOME, IMQ_VARHOME 및 IMQ_JAVAHOME은 플랫폼별 환경 변수 표시나 구문(예: UNIX의 $IMQ_HOME) **없이** 표시됩니다. 경로 이름에는 일반적으로 UNIX 디렉토리 구분자 표시(/)를 사용합니다.

표기 규칙

다음 표에서는 이 설명서에서 사용하는 표기 규칙을 소개합니다.

표 P–4 표기 규칙

활자체	의미	예
AaBbCc123	명령, 파일 및 디렉토리 이름 그리고 화면상의 컴퓨터 출력	해당 .login 파일을 편집합니다. 모든 파일을 나열하려면 ls -a를 사용합니다. machine_name% you have mail.
AaBbCc123	화면상의 컴퓨터 출력이 아닌, 사용자가 입력하는 내용	machine_name% **su** Password:
aabbcc123	자리 표시자: 실제 이름이나 값으로 대체	파일을 제거하는 명령은 rm *filename*입니다.

표 P–4 표기 규칙 *(계속)*

활자체	의미	예
AaBbCc123	책 제목, 새 용어 및 강조할 용어	**User's Guide**의 6장을 참조하십시오. **캐시는** 로컬에 저장되는 복사본입니다. 파일을 저장하지 **마십시오**. **주:** 일부 강조되는 항목은 온라인에서 굵게 표시됩니다.

명령 예의 쉘 프롬프트

다음 표는 C 쉘, Bourne 쉘, Korn 쉘 및 Windows 운영 체제에서의 기본 UNIX® 시스템 프롬프트와 수퍼유저 비밀번호입니다.

표 P–5 쉘 프롬프트

쉘	프롬프트
C 쉘	`machine_name%`
수퍼유저용 C 쉘	`machine_name#`
Bourne 쉘 및 Kron 쉘	`$`
수퍼유저를 위한 Bourne 쉘과 Korn 쉘	`#`
Windows	C:\

기호 규칙

다음 표에서는 이 설명서에서 사용할 수 있는 기호에 대해 설명합니다.

표 P–6 기호 규칙

기호	설명	예	의미
`[ ]`	선택적 인수와 명령 옵션을 포함합니다.	`ls [-l]`	`-l` 옵션은 필요하지 않습니다.
`{ | }`	필수 명령 옵션에 대한 일련의 선택 항목을 포함합니다.	`-d {y|n}`	`-d` 옵션에서는 `y` 인수 또는 `n` 인수를 사용해야 합니다.

표 P–6 기호 규칙 *(계속)*

기호	설명	예	의미
`${ }`	변수 참조를 나타냅니다.	`${com.sun.javaRoot}`	`com.sun.javaRoot` 변수의 값을 참조합니다.
-	여러 키를 동시에 누릅니다.	Control-A	Ctrl 키를 누른 채로 A 키를 누릅니다.
+	여러 키를 연속으로 누릅니다.	Ctrl+A+N	Ctrl 키를 눌렀다가 놓은 다음 후속 키를 누릅니다.
→	그래픽 사용자 인터페이스에서 메뉴 항목 선택을 나타냅니다.	파일 → 새로 만들기 → 템플리트	파일 메뉴에서 새로 만들기를 선택합니다. 새로 만들기 하위 메뉴에서 템플리트를 선택합니다.

설명서, 지원 및 교육

Sun 웹 사이트에서는 다음 추가 자료에 대한 정보를 제공합니다.

- 설명서(`http://www.sun.com/documentation/`)
- 지원(`http://www.sun.com/support/`)
- 교육(`http://www.sun.com/training/`)

Sun 제품 설명서 검색

docs.sun.com 웹 사이트에서 Sun 제품 설명서를 검색하는 것 외에도 검색 필드에 다음 구문을 입력하여 검색 엔진을 사용할 수 있습니다.

search-term `site:docs.sun.com`

예를 들어 “broker” 를 검색하려면 다음과 같이 입력합니다.

```
broker site:docs.sun.com
```

검색에 다른 Sun 웹 사이트를 포함시키려면(예: java.sun.com, www.sun.com 및 developers.sun.com) 검색 필드에서 "`docs.sun.com`" 대신 " `sun.com`"을 사용합니다.

타사 웹 사이트 참조

이 문서에는 관련된 정보를 추가로 얻을 수 있는 타사 URL도 나와 있습니다.

주 – Sun은 이 문서에 언급된 타사 웹 사이트의 사용 가능성에 대해 책임지지 않습니다. Sun은 해당 사이트 또는 자원에 있거나 사용 가능한 내용, 광고, 제품 또는 기타 자료에 대해 보증하지 않으며 책임 또는 의무를 지지 않습니다. Sun은 해당 사이트나 자원을 통해 사용 가능한 내용, 상품 또는 서비스의 사용과 관련해 발생했거나 발생했다고 간주되는 손해 또는 손실에 대해 책임이나 의무를 지지 않습니다.

사용자 의견 환영

Sun은 본 설명서의 개선을 위해 지속적으로 노력하고 있으며 고객의 의견과 제안을 환영합니다. 사용자 의견을 보내시려면 `http://docs.sun.com`에서 의견 보내기를 누릅니다. 온라인 양식에 설명서 전체 제목과 부품 번호를 입력합니다. 부품 번호는 7자리 또는 9자리 숫자이며, 해당 설명서의 제목 페이지나 문서의 URL에서 찾을 수 있습니다. 예를 들어 이 설명서의 부품 번호는 820-0532입니다.

◆ ◆ ◆ 1 장

1 메시징 시스템: 소개

Sun Java™ System Message Queue는 JMS(Java Message Service) 표준을 구현 및 확장하는 메시징 미들웨어 제품입니다. 이 문장의 의미를 완전히 이해한다면 26 페이지 “Message Queue: 요소 및 기능” 절부터 시작할 수 있습니다. 그렇지 않은 경우 처음부터 시작해야 합니다.

이 장에서는 Message Queue와 같은 제품의 기반이 되는 메시징 기술과 Message Queue에서 JMS 사양을 구현 및 확장하여 이 기술을 표준화하는 방법에 대해 설명합니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 17 페이지 “메시지 지향 미들웨어(Message-Oriented Middleware(MOM))”
- 22 페이지 “MOM 표준으로서의 JMS”
- 26 페이지 “Message Queue: 요소 및 기능”
- 30 페이지 “Message Queue 서비스 확장”
- 31 페이지 “활성화 기술로서의 Message Queue”

메시지 지향 미들웨어(Message-Oriented Middleware(MOM))

기업, 공공 기관 및 기술은 지속적으로 변화하므로 이러한 기관 및 기술에 사용되는 소프트웨어 시스템은 그러한 변화를 수용할 수 있어야 합니다. 기업을 합병하거나 서비스를 추가하거나 사용 가능한 서비스를 확장한 후 정보 시스템을 다시 구축하기가 어려울 수 있습니다. 새 구성 요소를 통합하거나, 기존 구성 요소를 가능한 한 효과적으로 확장하는 것이 가장 중요합니다. 이기종 구성 요소를 통합하는 가장 쉬운 방법은 이기종 구성 요소를 동종 요소로 다시 만드는 것이 아니라 차이점과 상관없이 구성 요소 간의 통신을 가능하게 해 주는 계층을 제공하는 것입니다. 이 계층을 **미들웨어**라 하며, 이는 독립적으로 개발되어 서로 다른 네트워크 플랫폼에서 실행되는 소프트웨어 구성 요소(응용 프로그램, Enterprise Java Bean, 서블릿 및 기타 구성 요소) 간의 상호 작용을 가능하게 해 줍니다. 이 상호 작용이 가능할 때 네트워크가 컴퓨터가 될 수 있습니다.

그림 1–1에 표시된 것처럼 개념적으로 미들웨어는 응용 프로그램 계층과 플랫폼 계층(운영 체제 및 기본 네트워크 서비스) 사이에 있습니다.

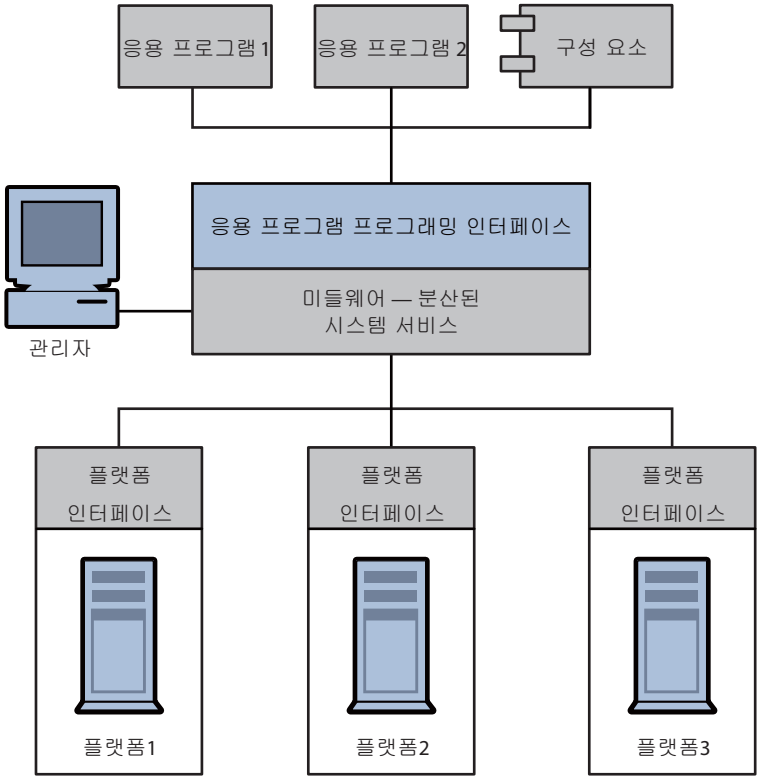

그림 1–1 미들웨어

서로 다른 네트워크 노드에 분산된 응용 프로그램은 다른 응용 프로그램을 호스트하는 운영 환경의 세부 정보나 다른 응용 프로그램에 연결해 주는 서비스를 고려할 필요 없이 응용 프로그램 인터페이스를 사용하여 통신합니다. 또한 관리 인터페이스를 제공하여 상호 연결된 새 가상 응용 프로그램 시스템의 안정성과 보안을 실현할 수 있습니다. 시스템의 성능을 측정, 조정하고 기능적인 손실 없이 확장할 수 있습니다.

미들웨어는 다음과 같은 범주로 그룹화할 수 있습니다.

- RPC(Remote Procedure Call) 기반 미들웨어 - 한 응용 프로그램의 프로시저를 사용하여 원격 응용 프로그램의 프로시저를 로컬 호출처럼 호출할 수 있게 해 줍니다. 미들웨어는 원격 프로시저를 찾아서 호출자가 알기 쉽게 사용할 수 있도록 해 주는 연결 메커니즘을 구현합니다. 일반적으로 이 유형의 미들웨어는 프로시저 기반 프로그램을 처리했지만, 지금은 객체 기반 구성 요소도 포함합니다.
- ORB(Object Request Broker) 기반 미들웨어 - 응용 프로그램의 객체를 배포하여 이기종 네트워크 간에 공유할 수 있게 해 줍니다.

- MOM(Message Oriented Middleware) 기반 미들웨어 - 분산 응용 프로그램 간에 메시지를 보내고 받으면서 데이터를 전달하고 교환할 수 있게 해 줍니다.

이러한 모든 모델에서 한 소프트웨어 구성 요소가 네트워크를 통해 다른 구성 요소의 동작에 영향을 줄 수 있습니다. RPC 및 ORB 기반 미들웨어에서는 시스템의 구성 요소를 밀접하게 연결하는 반면, MOM 기반 시스템에서는 구성 요소를 더 느슨하게 연결할 수 있습니다. RPC 또는 ORB 기반 시스템에서는 한 프로시저에서 다른 프로시저를 호출할 때 호출된 프로시저가 반환될 때까지 대기하였다가 다른 작업을 수행할 수 있습니다. 앞에서 설명한 것처럼 이러한 모델에서는 미들웨어가 부분적으로 수퍼링커의 역할을 하여 호출된 프로시저를 네트워크에서 찾고 네트워크 서비스를 통해 함수 또는 메소드 매개 변수를 프로시저에 전달한 다음 결과를 반환합니다.

MOM 기반 시스템에서는 그림 1–2에 표시된 것처럼 비동기식 메시지 교환을 통해 통신을 가능하게 합니다.

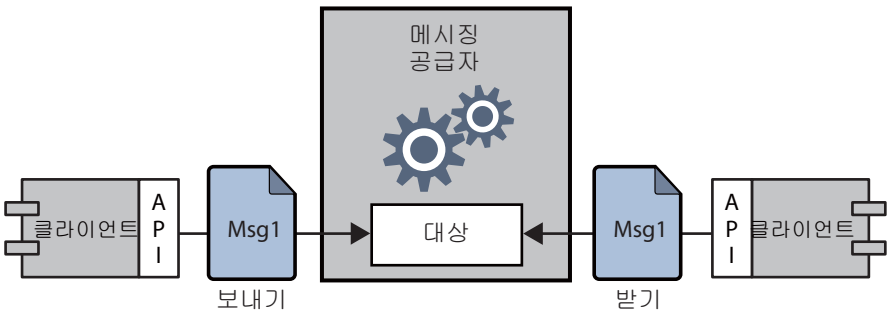

그림 1–2 MOM 기반 시스템

메시지 지향 미들웨어(Message Oriented Middleware)는 메시징 공급자를 사용하여 메시징 작업을 중재합니다. MOM 시스템의 기본 요소는 클라이언트, 메시지 및 MOM 공급자이며 여기에는 API와 관리 도구가 포함됩니다. MOM 공급자는 서로 다른 구조를 사용하여 메시지를 라우팅하고 전달할 수 있습니다. MOM 공급자는 중앙 집중식 메시지 서버를 사용하거나 각 클라이언트 컴퓨터에 라우팅 및 전달 기능을 배포할 수 있습니다. 일부 MOM 제품에서는 이 두 가지 방법을 결합하여 사용합니다.

클라이언트는 MOM 시스템을 통해 API를 호출하여 공급자가 관리하는 대상에게 메시지를 보냅니다. API 호출은 공급자 서비스를 호출하여 메시지를 라우팅하고 전달합니다. 메시지를 보낸 후 클라이언트는 다른 작업을 계속해서 수행할 수 있습니다. 이때 공급자는 수신하는 클라이언트가 메시지를 검색할 때까지 해당 메시지를 보관합니다. 메시지 기반 모델은 공급자의 중재와 연계하여 느슨하게 연결된 구성 요소의 시스템이 만들어질 수 있게 합니다. 그런 시스템은 개별 구성 요소나 연결이 실패하더라도 중단 없이 계속해서 안정적으로 작동할 수 있습니다.

메시징 공급자를 사용하여 클라이언트 간에 메시징을 중재할 경우 관리 인터페이스를 추가하여 성능을 모니터링 및 조정할 수 있다는 또 다른 이점이 있습니다. 따라서 클라이언트 응용 프로그램은 메시지 발신, 수신 및 처리와 관련된 것을 제외한 모든

문제에서 효과적으로 벗어날 수 있습니다. MOM 시스템 구현은 코드에서 담당하고 상호 운용성, 안정성, 보안, 확장성, 성능 등과 관련된 문제는 관리자가 해결해야 합니다.

지금까지는 메시지 지향 미들웨어를 사용하여 분산 구성 요소를 연결할 때의 장점에 대해 설명했습니다. 이 경우 몇 가지 단점도 존재합니다. 그 중 하나는 바로 느슨한 연결입니다. RPC 시스템에서 호출 함수는 호출된 함수가 작업을 완료할 때까지 반환되지 않습니다. 비동기식 시스템에서 호출 클라이언트는 작업을 처리하는 데 필요한 자원이 부족하여 호출된 구성 요소가 실패할 때까지 계속 수신자에게 작업을 로드할 수 있습니다. 물론 성능을 모니터링하고 메시지 흐름을 조정하여 이러한 조건을 최소화하거나 방지할 수 있지만 RPC 시스템에서는 그런 작업이 필요하지 않습니다. 중요한 것은 각 시스템 종류의 장점과 단점을 이해하는 것입니다. 각 시스템은 서로 다른 작업에 적합합니다. 정확하게 필요한 동작을 얻기 위해 두 종류의 시스템을 결합해야 하는 경우도 있습니다.

그림 1–3에서는 MOM 시스템에서 두 RPC 기반 시스템 간의 통신을 가능하게 하는 방법을 보여 줍니다. 그림의 왼쪽은 성능 향상을 위해 서로 다른 네트워크 노드에 클라이언트, 서버 및 데이터 저장소 구성 요소를 배포하는 응용 프로그램을 나타냅니다. 이것은 할인 항공 예약 시스템인데,최종 사용자는 이 서비스를 유료로 이용하여 주어진 목적지와 시간에 이용 가능한 가장 저렴한 요금을 찾을 수 있습니다. 데이터 저장소에는 등록된 사용자와 이 프로그램에 참여하는 항공사에 대한 정보가 보관됩니다. 사용자의 요청이 있을 경우 서버의 논리는 참여하는 항공사에서 가격을 쿼리하여 정보를 정렬한 다음 가장 저렴한 세 가지 요금을 사용자에게 표시합니다. 그림의 오른쪽은 참여하는 항공사 중 한 항공사의 티켓/예약 시스템을 나타내는 RPC 기반 시스템을 보여 줍니다. 그림의 오른쪽은 할인하는 사람이 연결된 항공사 수만큼 복제됩니다. 그런 각 항공사에 대해 데이터 저장소는 이용 가능한 항공편에 대한 정보(좌석, 비행 시간, 가격)를 보관합니다. 서버 구성 요소는 최종 사용자가 입력하는 데이터에 따라 해당 정보를 업데이트합니다. 또한 항공사 서버는 MOM 서비스에 가입하여 할인 예약 시스템에 대한 정보 요청을 수락하고 좌석 및 가격 정보를 반환합니다. 고객이 PanWorld의 할인 티켓 구매를 결정할 경우 해당 시스템의 서버 구성 요소는 데이터 저장소에서 해당 정보를 업데이트한 다음 요청자를 위한 티켓을 생성하거나 할인 서비스에 티켓을 생성하라는 메시지를 보냅니다.

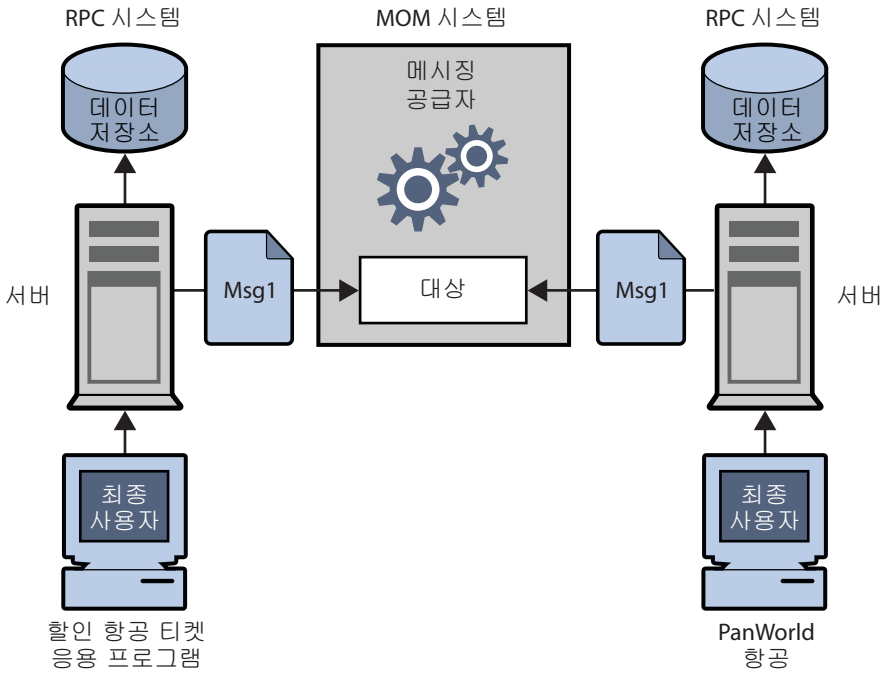

그림 1–3 RPC와 MOM 시스템 결합

이 예에서는 RPC 시스템과 MOM 시스템 간의 몇 가지 차이점에 대해 설명합니다. 분산 구성 요소가 연결되는 방식의 차이점에 대해서는 이미 설명했습니다. 다른 차이점으로는, RPC 시스템의 경우 클라이언트와 서버 구성 요소를 배포하고 연결하는 데 사용되며 이때 클라이언트가 주로 최종 사용자인 반면, MOM 시스템에서는 클라이언트가 메시징을 통해서만 상호 운용 가능한 이기종 소프트웨어 구성 요소인 경우가 많습니다.

MOM 시스템의 보다 심각한 문제는 MOM이 독점적인 제품으로 구현된다는 사실입니다. SuperMOM-X를 사용하는 회사에서 SuperMOM-Y를 사용하는 회사를 인수한다면 어떻게 될까요? 이 문제를 해결하려면 표준 메시징 인터페이스가 필요합니다. SuperMOM-X와 SuperMOM-Y 모두 이 인터페이스를 구현한다면, 한 시스템에서 실행하도록 개발된 응용 프로그램이 다른 시스템에서도 실행될 수 있습니다. 그러한 인터페이스는 쉽게 학습할 수 있으면서 고급 메시징 응용 프로그램을 지원하는 데 충분한 기능을 제공해야 합니다. 그러한 목표 아래 1998년, JMS(Java Message Service) 사양이 발표되었습니다. 다음 절에서는 JMS의 기본 기능을 소개하고, 어떻게 기존의 독점적인 MOM 제품의 공통 요소를 포함하면서 차이점과 추가 확장을 수용하도록 표준이 개발되었는지를 설명합니다.

MOM 표준으로서의 JMS

JMS(Java Messaging Service) 사양은 원래 Java 응용 프로그램이 기존 MOM 시스템에 액세스할 수 있도록 개발되었습니다. JMS 사양은 첫 선을 보인 이후 기존의 많은 MOM 공급업체에 의해 채택되었으며 물론 비동기식 메시징 시스템으로 구현되었습니다.

JMS 사양의 설계자들은 이 사양을 개발할 때 다음과 같이 기존 메시징 시스템의 필수적인 요소를 수용하고자 했습니다.

- 메시지 경로를 지정하고 전달하는 메시징 공급자 개념
- 별개의 메시징 패턴 또는 도메인(예: 지점간 메시징 및 게시/가입 메시징)
- 동기식 및 비동기식 메시지 수신 기능
- 안정적인 메시지 전달 지원
- 공통 메시지 형식(예: 스트림, 텍스트, 바이트)

공급업체는 JMS 인터페이스를 구현하는 라이브러리, 메시지를 라우팅하고 전달하는 기능, 메시징 서비스를 관리, 모니터링하고 조정하는 관리 도구 등으로 구성되는 *JMS* **공급자**를 제공하여 JMS 사양을 구현합니다. 라우팅 및 전달 기능은 중앙 집중식 메시지 서버나 브로커에서 수행되거나, 각 클라이언트 런타임의 일부인 기능을 통해 구현될 수 있습니다.

또한 JMS 공급자는 다음과 같은 다양한 역할을 수행할 수 있습니다. JMS 공급자를 독립 실행형 제품으로 만들거나 대용량 분산 런타임 시스템의 내장 구성 요소로 만들 수 있습니다. 독립 실행형 제품으로 만든 JMS 공급자는 엔터프라이즈 응용 프로그램 통합 시스템의 백본을 정의하는 데 사용될 수 있고, 응용 프로그램 서버에 내장된 JMS 공급자는 구성 요소 간 메시징을 지원할 수 있습니다. 예를 들어 J2EE는 JMS 공급자를 사용하여 Message-Driven Bean을 구현하고 EJB 구성 요소가 메시지를 보내고 받을 수 있도록 합니다.

기존 시스템의 모든 기능을 포함하는 표준을 만들었다면 학습하거나 구현하기 어려운 시스템이 생성되었을 것입니다. 대신, JMS에서는 메시징 개념과 기능의 공통 분모를 정의했습니다. 그 결과 배우기 쉽고 JMS 공급자 간 JMS 응용 프로그램의 이식성을 최대화하는 표준이 생성되었습니다. JMS는 프로토콜 표준이 아니라 API 표준입니다. JMS 클라이언트를 공급업체 간에 쉽게 이동할 수 있습니다. 그러나 일반적으로 서로 다른 JMS 공급업체 간에는 직접 통신할 수 없습니다.

다음 절에서는 JMS 사양에 정의된 기본 객체와 메시징 패턴에 대해 설명합니다.

JMS 메시징 객체 및 패턴

메시지를 보내거나 받으려면 메시지 브로커로도 구현되는 JMS 공급자에 JMS 클라이언트를 맨 먼저 연결해야 합니다. 그러면 클라이언트와 브로커 간에 통신 채널이 열립니다. 그런 다음 클라이언트는 메시지를 작성, 생성 및 소비하는 세션을 설정해야 합니다. 세션을 클라이언트와 브로커 간의 특정 대화를 정의하는 메시지 스트림으로 간주할 수 있습니다. 클라이언트 자체는 **메시지 제작자** 또는 **메시지 소비자**입니다.

메시지 제작자는 브로커가 관리하는 **대상**에게 **메시지**를 보냅니다. 메시지 소비자는 해당 대상에 액세스하여 메시지를 소비합니다. 메시지는 헤더, 선택적 등록 정보 및 본문으로 구성됩니다. 본문에는 데이터가 저장되고, 헤더에는 브로커가 메시지를 라우팅하고 관리하는 데 필요한 정보가 들어 있으며, 등록 정보는 클라이언트 응용 프로그램이나 공급자가 자체 메시지 처리 요건에 맞게 정의할 수 있습니다. JMS 사양을 구성하는 기본 객체는 연결, 세션, 대상, 메시지, 제작자 및 소비자입니다.

클라이언트 응용 프로그램은 이러한 기본 객체를 사용하여 두 가지 메시징 패턴 또는 도메인을 통해 메시지를 주고 받을 수 있습니다. 그림 1–4를 참조하십시오.

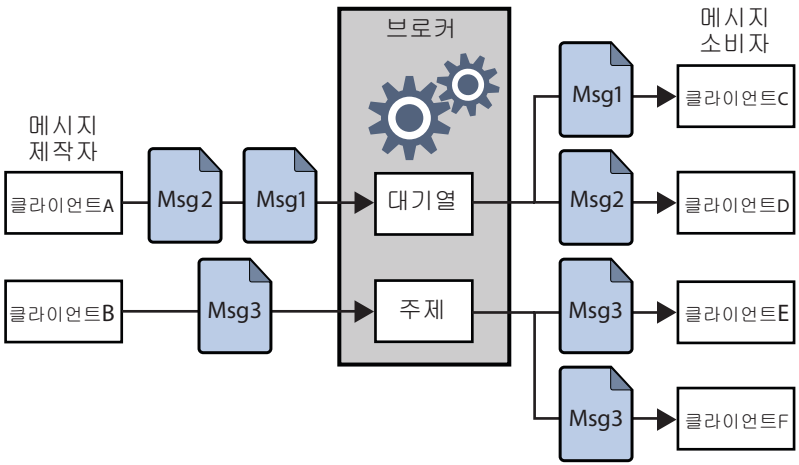

그림 1–4 JMS 메시징 패턴

클라이언트 A와 B는 서로 다른 두 대상을 경유하여 클라이언트 C, D 및 E에게 메시지를 보내는 메시지 제작자입니다.

- 클라이언트 A, C 및 D 간의 메시징은 지점 간 패턴을 보여 줍니다. 클라이언트는 이 패턴을 사용하여 메시지를 대기열 대상에게 보냅니다. 이때 한 수신자만 대기열 대상으로부터 메시지를 받을 수 있습니다. 해당 대상에 액세스하는 다른 수신자는 해당 메시지를 받을 수 없습니다.
- 클라이언트 B, E 및 F 간의 메시징은 게시/가입 패턴을 보여 줍니다. 클라이언트는 이 브로드캐스트 패턴을 사용하여 주제 대상에게 메시지를 보냅니다. 이때 모든 소비자 가입자가 해당 메시지를 검색할 수 있습니다. 각 가입자는 고유의 메시지 복사본을 갖습니다.

도메인의 메시지 소비자는 메시지를 동기식으로 받을지 비동기식으로 받을지 여부를 선택할 수 있습니다. 동기식 소비자는 메시지를 검색하기 위해 명시적 호출을 생성하고, 비동기식 소비자는 보류 중인 메시지를 전달하기 위해 호출되는 콜백 메소드를 지정합니다. 또한 소비자는 들어오는 메시지에 대한 선택 기준을 지정하여 메시지를 필터링할 수 있습니다.

관리 대상 객체

JMS 사양에서는 모든 가능성을 고갈시키지 않으면서 기존 MOM 시스템의 많은 요소를 결합하는 표준을 탄생시켰습니다. 즉, 차이점과 추가 확장을 수용할 수 있는 확장 체계를 마련하고자 했습니다. JMS는 개별 공급자가 다양한 메시징 요소를 정의하고 구현할 수 있는 가능성을 남겨 두었습니다. 이러한 요소로는 로드 균형 조정, 표준 오류 메시지, 관리 API, 보안, 기본 와이어 프로토콜, 메시지 저장소 등이 있습니다. 다음 26 페이지 “Message Queue: 요소 및 기능” 절에서는 Message Queue가 그러한 많은 요소를 구현하고 JMS 사양을 확장하는 방법에 대해 설명합니다.

JMS에서 완전히 정의하지 않은 두 메시징 요소는 연결 팩토리와 대상입니다. 이러한 요소는 JMS 프로그래밍 모델의 기본 요소이지만, 공급자가 이러한 객체를 정의하고 관리하는 방식에 많은 차이가 있으므로 공통 정의를 생성하는 것은 가능하지도 않고 바람직하지도 않습니다. 따라서 이 두 객체는 프로그래밍 방식으로 만들지 않고 관리 도구를 사용하여 만들어 구성하는 것이 일반적입니다. 그러면 두 객체가 객체 저장소에 저장되고 JMS 클라이언트가 표준 JNDI 조회를 통해 액세스할 수 있습니다.

- **연결 팩토리 관리 대상 객체**는 클라이언트를 브로커에 연결하는 데 사용됩니다. 연결 팩토리 관리 대상 객체는 연결 처리, 클라이언트 아이디, 메시지 헤더 대체, 안정성, 흐름 제어 등과 같은 메시징 동작의 특정 측면을 관리하는 공급자별 정보를 캡슐화합니다. 지정된 연결 팩토리에서 파생되는 모든 연결은 해당 팩토리에 대해 구성된 동작을 나타냅니다.
- **대상 관리 객체**는 브로커에서 물리적 대상을 참조하는 데 사용됩니다. 관리 대상 객체는 공급자별 이름 지정(주소 구문) 규약을 캡슐화하고 대상이 사용되는 메시징 도메인(대기열 또는 주제)을 지정합니다.

JMS 클라이언트는 관리 대상 객체를 조회할 필요가 없으며, 이러한 객체를 프로그래밍 방식으로 만들어 브로커의 메모리에 저장할 수 있습니다. 프로토타입을 신속하게 제작하려면 이러한 객체를 프로그래밍 방식으로 만드는 것이 가장 쉽습니다. 그러나 프로덕션 환경에서 배포할 경우에는 중앙 저장소에서 관리 대상 객체를 조회하여 메시징 동작을 훨씬 쉽게 제어하고 관리할 수 있습니다.

- 관리자는 연결 팩토리 객체에 대해 관리 대상 객체를 사용하여 이러한 객체를 재구성함으로써 메시징 성능을 조정할 수 있습니다. 그러면 다시 코딩 작업을 하지 않고도 성능을 향상시킬 수 있습니다.
- 관리자는 물리적 대상에 대해 관리 대상 객체를 사용하여 클라이언트에게 사전 구성된 객체에 액세스하도록 요구함으로써 브로커에서 이러한 대상의 증가를 제어할 수 있습니다.
- 관리 대상 객체는 개발자가 공급자별 구현 세부 정보를 보지 못하도록 차단하고, 개발자가 특정 공급자를 위해 개발하는 코드를 약간만 변경하거나 변경하지 않고 다른 공급자에게 이식할 수 있도록 해 줍니다.

관리 대상 객체를 사용하면 그림 1–5에 표시된 것처럼 기본 JMS 응용 프로그램에 최종 기능 하나가 추가됩니다.

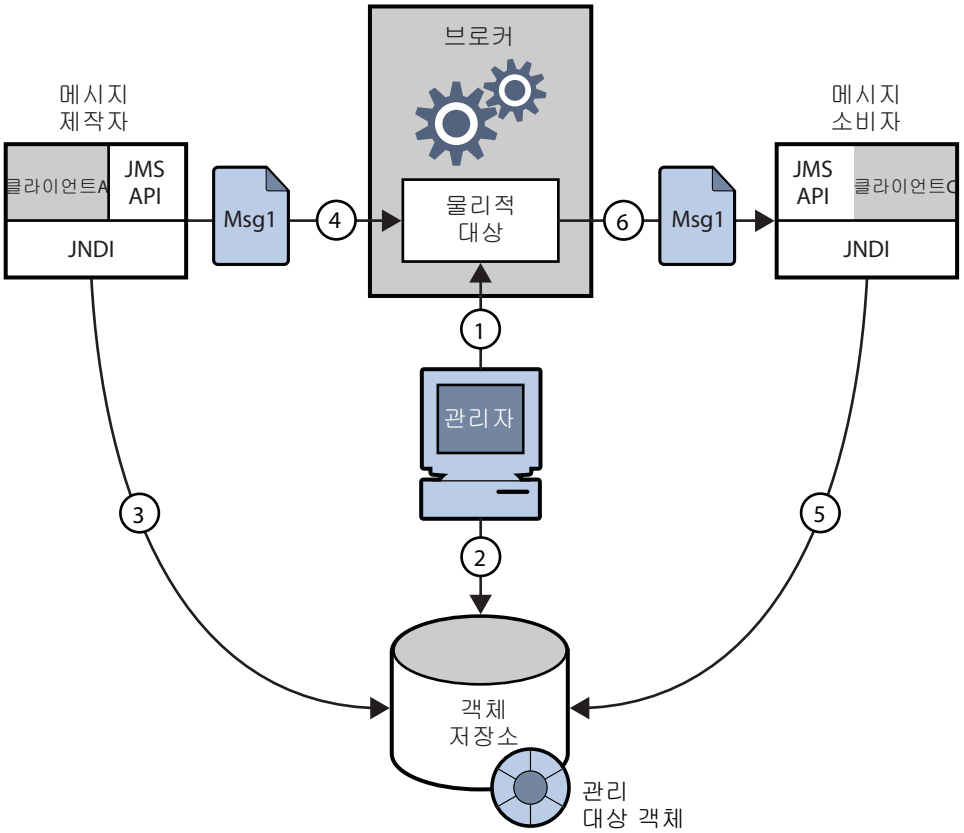

그림 1–5 JMS 응용 프로그램의 기본 요소

그림 1–5에서는 메시지 제작자와 메시지 소비자가 대상 관리 대상 객체를 사용하여 해당 물리적 대상에 액세스하는 방법을 보여 줍니다. 표시된 단계는 관리자와 클라이언트 응용 프로그램이 이 메커니즘을 사용하여 메시지를 보내고 받기 위해 수행해야 하는 작업을 나타냅니다.

▼ 관리 대상 객체를 대상으로 사용하는 방법

1. **관리자는 브로커에 물리적 대상을 만듭니다.**
2. **관리자는 대상 관리 대상 객체를 만든 다음 해당하는 물리적 대상의 이름과 유형(대기열 또는 주제)을 지정하여 객체를 구성합니다.**
3. **메시지 제작자는 JNDI 조회 호출을 사용하여 관리 대상 객체를 조회합니다.**
4. **메시지 제작자는 대상에게 메시지를 보냅니다.**

5 **메시지 소비자는 메시지를 받을 대상 관리 대상 객체를 조회합니다.**

6 **메시지 소비자는 대상으로부터 메시지를 받습니다.**

연결 팩토리 관리 대상 객체를 사용하는 과정도 비슷합니다. 관리자는 관리 도구를 사용하여 연결 팩토리 관리 대상 객체를 만들어 구성합니다. 클라이언트는 연결 팩토리 객체를 조회한 후 이를 사용하여 연결을 만듭니다.

관리 대상 객체를 사용하면 메시징 처리 과정에 몇 단계가 추가되지만, 메시징 응용 프로그램의 견고성과 이식성이 향상됩니다.

Message Queue: 요소 및 기능

지금까지는 메시지 지향 미들웨어의 요소에 대해 설명하고 JMS를 사용하여 MOM 응용 프로그램에 이식성을 추가하는 방식에 대해 설명했습니다. 이제 Message Queue에서 JMS 사양을 구현하는 방법과 안정적이고 안전하며 확장 가능한 메시징 서비스를 제공하기 위해 사용하는 기능과 도구에 대해 설명하겠습니다.

우선 많은 JMS 공급자와 마찬가지로 Message Queue를 독립 실행형 제품으로 사용하거나, 비동기식 메시징을 제공하기 위해 J2EE 응용 프로그램 서버에 내장된 활성화 기술로 사용할 수 있습니다. Message Queue가 J2EE에서 수행하는 역할에 대한 자세한 내용은 5 장를 참조하십시오. 다른 JMS 공급자와 다르게, Message Queue는 JMS 참조 구현으로 지정되었습니다. 이러한 지정은 Message Queue가 올바르고 완벽한 JMS 구현임을 입증합니다. 또한 Message Queue 제품이 향후의 JMS 개정 및 확장을 통해 최신 상태로 유지됨을 보장합니다.

Message Queue 서비스

JMS 공급자로서 Message Queue는 JMS 인터페이스를 구현하고 관리 서비스와 제어를 수행하는 **메시징 서비스**를 제공합니다. 지금까지 메시지 전달에서 브로커의 역할을 중심으로 JMS 공급자에 대해 설명했습니다. 그러나 JMS 공급자는 브로커 이외에 안정적이고 안전하며 확장 가능한 메시징을 제공하기 위한 많은 요소를 포함해야 합니다. 그림 1–6에서는 Message Queue 메시지 서비스를 구성하는 요소를 보여 줍니다. 그 중에는 다양한 연결 서비스(각기 다른 프로토콜 지원), 관리 도구를 비롯하여 메시징, 모니터링 및 사용자 정보용 데이터 저장소도 있습니다. Message Queue 서비스는 그림에 회색으로 표시된 모든 요소를 포함하고 있습니다.

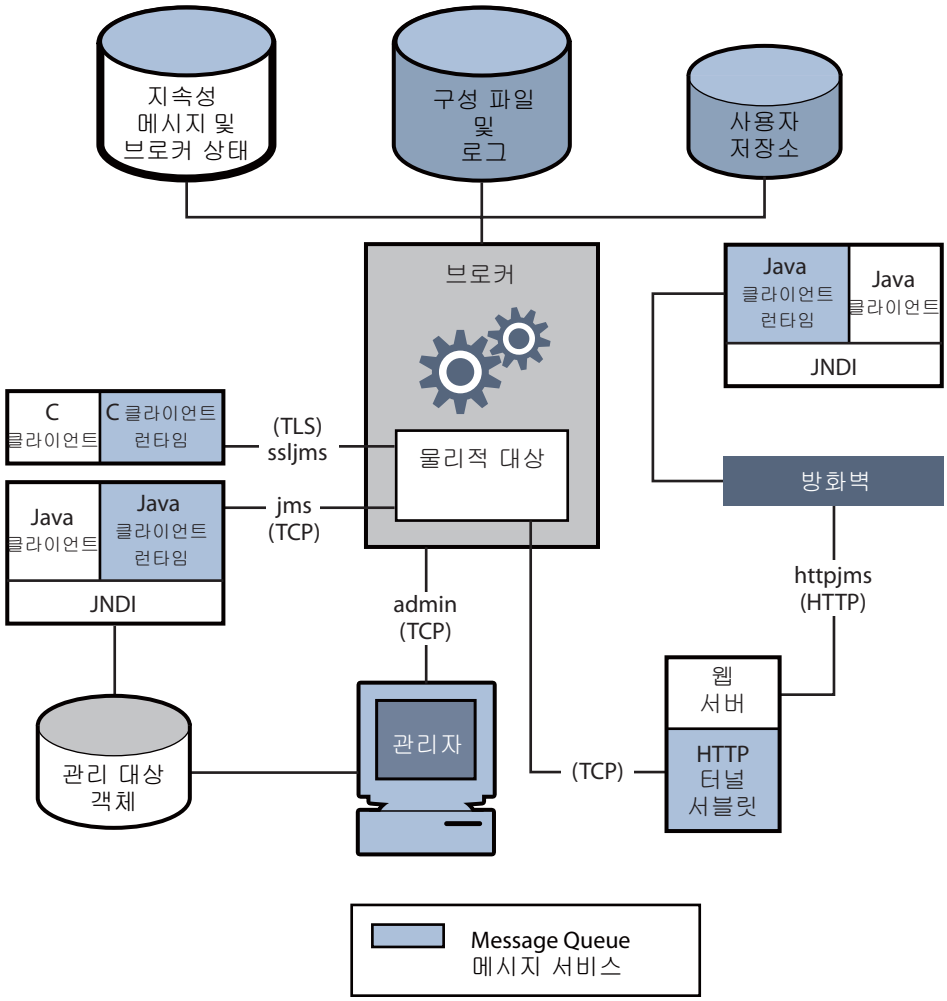

그림 1–6 Message Queue 서비스

위와 같이, 전 기능 JMS 공급자는 기본 JMS 모델보다 더 복잡합니다. 다음 절에서는 위에서 보여준 Message Queue 서비스 요소에 대해 설명합니다. 이러한 요소는 브로커, 클라이언트 런타임 지원 및 관리라는 세 가지 범주로 분류될 수 있습니다.

브로커에 연결

그림 1–6에 표시된 것처럼 응용 프로그램 클라이언트와 관리 클라이언트 모두 브로커에 연결할 수 있습니다. JMS 사양에서는 공급자가 특정 와이어 프로토콜을 구현하도록 규정하지 않습니다. 응용 프로그램 클라이언트와 관리 클라이언트가 브로커에 연결하는

데 사용하는 Message Queue 서비스는 현재 TCP, TLS, HTTP 또는 HTTPS 프로토콜의 상위 계층에 있습니다. HTTP의 상위 계층에 있는 서비스를 사용하면 방화벽을 통해 메시지를 전달할 수 있습니다.

- JMS 지원을 제공하며 클라이언트와 브로커와의 연결을 가능하게 해 주는 서비스(`jms`, `ssljms`, `http` 또는 `https`)는 `NORMAL` 서비스 유형이며 `TCP`, `TLS`, `HTTP` 또는 `HTTPS` 프로토콜의 상위 계층에 있습니다.
- 관리자를 브로커에 연결하는 데 사용하는 서비스(`admin`, `ssladmin`)는 `ADMIN` 서비스 유형이며 `TCP` 또는 `TLS` 프로토콜의 상위 계층에 있습니다.

기본적으로 브로커를 시작하면 `jms`와 `admin` 서비스가 실행됩니다. 또한 이러한 연결 서비스 중 어느 것이라도 또는 전부 실행하도록 브로커를 구성할 수 있습니다. 각 서비스는 특정 인증 및 권한 부여(액세스 제어) 기능을 지원하며 다중 스레드 방식으로서 다중 연결을 지원합니다.

연결이 실패할 경우 Message Queue 서비스는 동일한 브로커 또는 다른 브로커(이 기능이 사용 가능하도록 설정된 경우)에 대한 클라이언트 연결을 자동으로 다시 시도할 수 있습니다. 자세한 내용은 부록 B의 자동 재연결 기능에 대한 설명을 참조하십시오.

클라이언트는 연결을 가져오는 연결 팩토리를 만들 때 연결 런타임 지원을 구성할 수 있습니다. 옵션을 사용하여 연결할 브로커, 재연결 처리 방법, 메시지 흐름 제어 등을 지정할 수 있습니다. 연결을 구성하는 방법에 대한 자세한 내용은 40 페이지 “연결 팩토리 및 연결”을 참조하십시오.

브로커

메시지 서비스의 핵심은 안정적으로 메시지를 라우팅, 전달하고 사용자를 인증하며 성능 모니터링을 위한 데이터를 수집하는 브로커입니다.

- 메시지를 라우팅하고 전달하기 위해 브로커는 들어오는 메시지를 각각의 대상에 배치하고 이 대상을 드나드는 메시지 흐름을 관리합니다.
- 안정적인 전달을 위해 브로커는 영구 저장소를 사용하여 상태 정보와 영구 메시지를 수신 시점까지 저장합니다. 브로커 또는 연결이 실패할 경우 브로커는 저장된 정보를 사용하여 브로커의 상태를 복원한 다음 작업을 다시 시도할 수 있습니다.
- 교환되는 데이터의 보안을 위해 브로커는 인증된 연결을 사용합니다. SSL과 같은 보안 프로토콜을 통해 데이터를 암호화할 수도 있습니다. 또한 브로커는 사용자 그리고 사용자가 액세스할 수 있는 데이터 또는 작업에 대한 정보를 보관하는 저장소를 사용하고 관리합니다. 브로커는 서비스를 요청하는 사용자를 인증하고 이 저장소에서 정보를 조회하여 사용자가 수행할 작업에 대한 권한을 부여합니다.
- 브로커는 시스템을 모니터링하기 위해 관리자가 액세스하여 성능을 측정하고 브로커를 조정할 수 있는 메트릭 및 진단 정보를 생성합니다. 메트릭 정보를 프로그래밍 방식으로 사용하여 응용 프로그램이 메시지 흐름과 패턴을 조정하여 성능을 향상시키게 할 수도 있습니다.

Message Queue 서비스는 관리자가 브로커 지원을 구성하는 데 사용할 수 있는 다양한 관리 도구를 제공합니다. 자세한 내용은 30 페이지 "관리"를 참조하십시오.

클라이언트 런타임 지원

클라이언트 런타임 지원은 Message Queue 클라이언트를 구축할 때 연결되는 라이브러리에 제공됩니다. 클라이언트 런타임은 클라이언트의 일부가 되는 Message Queue 서비스 비트로 간주할 수 있습니다. 예를 들어 클라이언트 코드에서 메시지를 보내는 API 호출을 만들 경우, 이 라이브러리에서 호출되는 코드는 브로커의 물리적 대상으로 메시지를 전달할 때 사용될 프로토콜에 적합한 메시지 비트를 패키지화합니다.

Java 및 C 클라이언트 지원

JMS 공급자는 Java 클라이언트만 지원해야 합니다. 그러나 그림 1–6에 표시된 것처럼 Message Queue 클라이언트는 Java 또는 공급자별 C API를 사용하여 메시지를 보내거나 받을 수 있습니다. 이러한 인터페이스는 Java 또는 C 런타임 라이브러리에 구현되는데, 브로커와의 연결을 만들고 요청된 연결 서비스에 적절하게 비트를 패키지화하는 실제 작업을 담당합니다.

- Java 클라이언트 런타임은 Java 클라이언트에 브로커와의 상호 작용에 필요한 객체를 제공합니다. 이러한 객체에는 연결, 세션, 메시지, 메시지 제작자 및 메시지 소비자가 포함됩니다.
- C 클라이언트 런타임은 C 클라이언트에 브로커와의 상호 작용에 필요한 기능과 구조를 제공합니다. C 클라이언트 런타임은 JMS 프로그래밍 모델의 절차상 버전을 지원합니다. C 클라이언트는 JNDI를 사용하여 관리 대상 객체에 액세스할 수 없지만, 연결 팩토리와 대상을 프로그래밍 방식으로 만들 수 있습니다.

Message Queue 서비스는 레거시 C 및 C++ 응용 프로그램이 JMS 기반 메시징에 참여할 수 있도록 C API를 제공합니다. 이러한 두 API에서 제공하는 기능에는 많은 차이가 있으며, 그에 대한 자세한 내용은 53 페이지 "Java 및 C 클라이언트"를 참조하십시오.

JMS 사양이 Java 클라이언트 전용 표준이라는 점을 명심하십시오. C 지원은 Message Queue 공급자에만 해당되며 다른 공급자에게 연결할 클라이언트 응용 프로그램에서는 사용할 수 없습니다.

Java 클라이언트에 대한 SOAP 지원

Message Queue Java 클라이언트는 JMS 메시지로 래핑된 SOAP 메시지를 보내고 받을 수도 있습니다. SOAP(Simple Object Access Protocol)를 사용하면 분산 환경의 두 피어 간에 구조화된 데이터를 교환할 수 있습니다. 교환되는 데이터는 XML 스키마를 통해 지정됩니다.

Sun SOAP 처리는 현재 지점 간 모델 사용으로 제한되었으며, 안정성이 보장되지 않습니다. SOAP 메시지를 JMS 메시지로 래핑한 다음 브로커를 사용하여 라우팅하면 안정적인 전달을 보장하고 주제 및 지점 간 도메인 사용이 모두 가능한 전 기능 Message

Queue 메시징을 이용할 수 있습니다. Message Queue는 메시지 제작자가 SOAP 메시지를 JMS 메시지로 래핑하고 메시지 소비자가 JMS 메시지에서 SOAP 메시지를 추출하는 데 사용할 수 있는 유틸리티 루틴을 제공합니다.

SOAP 메시지 처리에 대한 자세한 내용은 52 페이지 “SOAP 메시지 작업”을 참조하십시오.

관리

Message Queue 서비스는 다음 작업을 수행하는 데 사용할 수 있는 명령줄 도구를 제공합니다.

- 브로커 시작 및 구성
- 대상 만들기 및 관리, 브로커 연결 관리, 브로커 자원 관리
- JNDI 객체 저장소에서 관리 대상 객체 추가, 나열, 업데이트 및 삭제
- 파일 기반 사용자 저장소 채우기 및 관리
- 영구 저장소 용도로 JDBC 호환 데이터베이스 만들기 및 관리

GUI 기반 관리 콘솔을 사용하여 다음 명령줄 기능을 수행할 수도 있습니다.

- 브로커에 연결하여 관리
- 물리적 대상 만들기 및 관리
- 객체 저장소에 연결, 저장소에 객체 추가 및 객체 관리

Message Queue 서비스 확장

클라이언트 수 또는 연결 수가 증가하면 병목 현상 해소 또는 성능 향상을 위해 메시지 서비스 확장이 필요할 수 있습니다. Message Queue 메시지 서비스는 사용자의 필요에 따라 많은 확장 옵션을 제공합니다. 이러한 옵션은 편의상 다음과 같은 범주로 정렬될 수 있습니다.

- **수직 확장**은 처리 능력을 늘리고 사용 가능한 자원을 확장하는 것입니다. 프로세서 또는 메모리를 추가하거나, 공유 스레드 모델로 전환하거나, Java VM을 64비트 모드로 실행하여 수직 확장을 할 수 있습니다.

  지점 간 도메인을 사용하는 경우에는 여러 소비자가 하나의 대기열에 액세스하도록 허용하여 소비자 측면을 확장할 수 있습니다. 이 방법을 사용할 경우 최대 활성 및 백업 소비자 수를 지정할 수 있습니다. 또한 로드 균형 조정 메커니즘에서는 소비자의 현재 용량과 메시지 처리 속도를 고려합니다. 이는 Message Queue의 기능입니다. (JMS 사양은 단일 소비자가 대기열에 액세스할 때의 메시징 동작을 정의합니다. 여러 소비자의 액세스가 허용되는 대기열에 대한 동작은 공급자별로 다릅니다. 이 확장 옵션에 대한 자세한 내용은 Message Queue 개발 안내서를 참조하십시오.)
- **상태 없는 수평 확장**은 브로커를 추가한 다음 해당 브로커에 기존 클라이언트를 재배포하여 이루어집니다. 이 방법은 쉽게 구현할 수 있지만, 메시징 작업이 독립 작업 그룹으로 분류될 수 있는 경우에만 적합합니다.

- **상태 있는 수평 확장**은 브로커를 **클러스터**에 연결하여 이루어집니다. 브로커 클러스터에서 각 브로커는 로컬 응용 프로그램 클라이언트뿐만 아니라 클러스터에 있는 모든 다른 브로커에도 연결됩니다. 브로커는 동일한 호스트에 있거나 네트워크에 분산될 수 있습니다. 대상 및 소비자에 대한 정보가 클러스터에 있는 모든 브로커에 복제됩니다. 대상 또는 가입자에 대한 업데이트가 함께 전파되므로, 각 브로커는 직접 연결된 제작자로부터 클러스터의 다른 브로커에 연결된 소비자로 메시지를 라우팅할 수 있습니다. 백업 소비자가 사용되는 환경에서 하나의 브로커나 연결이 실패할 경우, 액세스할 수 없는 소비자에게 보낸 메시지를 다른 브로커의 백업 소비자에게 전달할 수 있습니다.

  브로커 또는 연결이 실패할 경우 영구 항목(대상 및 영구 가입)에 대한 상태 정보가 동기화되지 않을 수 있습니다. 예를 들어 클러스터된 브로커가 중단되어 대상이 클러스터에 있는 다른 브로커에 생성되고 첫 번째 브로커가 다시 시작될 경우, 해당 브로커는 새 대상에 대해 알 수 없습니다. 이 문제를 사전에 방지하기 위해 클러스터의 한 브로커를 **마스터 브로커**로 지정할 수 있습니다. 이 브로커는 **마스터 구성 파일**에서 대상 및 영구 가입에 대한 모든 변경을 추적하고, 클러스터에서 일시적으로 오프라인 상태인 브로커를 업데이트합니다. 자세한 내용은 4 장를 참조하십시오.

  마스터 브로커를 사용하더라도 Message Queue는 브로커 또는 연결이 실패할 경우 서비스 가용성만 제공하며 데이터 가용성은 제공하지 않습니다. 예를 들어 클러스터된 브로커를 사용할 수 없는 경우, 해당 브로커에 보관된 모든 영구 메시지는 브로커가 복구될 때까지 사용할 수 없습니다. 현재는 SunCluster Message Queue 에이전트를 통해서만 데이터 가용성이 제공됩니다. 이 경우 영구 저장소는 공유 파일 시스템에 유지됩니다. 브로커가 실패할 경우 두 번째 노드의 Message Queue 에이전트가 공유 저장소를 인계할 브로커를 시작합니다. 클라이언트가 해당 브로커에 다시 연결되므로 영구 데이터에 대한 액세스와 서비스가 지속됩니다.

활성화 기술로서의 Message Queue

Java 2 Platform, Enterprise Edition(J2EE 플랫폼)은 Java 프로그래밍 환경에서 분산된 구성 요소 모델의 사양입니다. J2EE 플랫폼의 요구 사항 중 하나는 분산 구성 요소가 안정적인 비동기식 메시지 교환을 통해 상호 작용할 수 있어야 한다는 것입니다. 이 기능을 제공하는 JMS 공급자는 다음 두 가지 역할을 담당할 수 있습니다. 이 기능을 사용하여 서비스를 제공하고 JMS 메시지를 소비할 수 있는 EJB(Enterprise Java Bean) 구성 요소의 특수 유형인 MDB(Message-Driven Bean)를 지원할 수 있습니다.

J2EE 호환 응용 프로그램 서버는 해당 JMS 공급자가 제공하는 자원 어댑터를 사용해야 그 공급자의 기능을 활용할 수 있습니다. Message Queue에서는 그러한 자원 어댑터를 제공합니다. Application Server 환경에 배포되고 실행되는 MDB를 비롯한 J2EE 구성 요소는 플러그인된 JMS 공급자 지원을 활용하여 내부에서 또는 외부 JMS 구성 요소와 JMS 메시지를 교환할 수 있습니다. 따라서 분산 구성 요소에 강력한 통합 기능을 제공하게 됩니다.

Message Queue 자원 어댑터에 대한 자세한 내용은 5 장을 참조하십시오.

Message Queue 기능 요약

Message Queue는 JMS 사양의 요구 사항을 훨씬 능가하는 성능과 기능을 갖추고 있습니다. 이러한 기능을 사용하여 Message Queue는 24시간 중차대한 작업으로 무수히 많은 메시지를 교환하는 많은 수의 분산 구성 요소로 구성된 시스템을 통합할 수 있습니다. 이 기능에 대한 요약은 부록 B을 참조하십시오.

◆ ◆ ◆ 2 장

2 클라이언트 프로그래밍 모델

이 장에서는 Message Queue 클라이언트 프로그래밍의 기본적인 내용에 대해 설명합니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 34 페이지 "메시징 도메인"
- 39 페이지 "프로그래밍 객체"
- 44 페이지 "메시지 생성"
- 45 페이지 "메시지 소비"
- 46 페이지 "요청-응답 패턴"
- 47 페이지 "안정적인 메시징"
- 50 페이지 "시스템에서의 메시지 경로"
- 53 페이지 "Java 및 C 클라이언트"

이 장에서는 Java 클라이언트의 설계와 구현을 중심으로 설명합니다. 일반적으로 C 클라이언트 설계는 Java 클라이언트 설계와 거의 비슷합니다. 이 장의 마지막 절에서는 Java 클라이언트와 C 클라이언트의 차이점에 대해 요약합니다. Message Queue 클라이언트 프로그래밍에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients** 및 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for C Clients**를 참조하십시오.

3 장에서는 Message Queue 서비스를 사용하여 메시징 성능을 지원, 관리 및 조정하는 방법에 대해 설명합니다.

설계 및 성능

Message Queue 응용 프로그램의 동작은 클라이언트 설계, 연결 구성, 브로커 구성, 브로커 조정, 자원 관리와 같은 여러 가지 요소에 따라 다릅니다. 이 중 일부는 개발자에게 책임이 있고 나머지는 관리자와 관련되어 있습니다. 그러나 개발자는 Message Queue 서비스에서 응용 프로그램 설계를 지원하고 확장하는 방법을 잘 알고 있어야 하고, 관리자는 응용 프로그램을 조정할 때 설계 목표를 잘 알고 있어야 합니다. 메시징 동작은 재설계, 신중한 모니터링 및 조정을 통해 최적화될 수 있습니다. 따라서 우수한 Message

Queue 응용 프로그램 만들기의 핵심은 개발자와 관리자가 응용 프로그램 라이프사이클의 각 단계에서 실현될 수 있는 내용을 이해하고 원하는 동작과 관찰된 동작에 대한 정보를 공유하는 것입니다.

메시징 도메인

구성 요소와 응용 프로그램은 메시징 미들웨어를 통해 메시지를 생성하고 소비하면서 통신할 수 있습니다. JMS API에서는 이 통신을 제어하는 두 가지 패턴, 즉**메시징 도메인** **지점간 메시징** 및 **게시/가입 메시징**을 정의합니다. JMS API는 이 패턴을 지원하도록 구성되어 있습니다. 연결, 세션, 제작자, 소비자, 대상, 메시지와 같은 기본 JMS 객체는 두 도메인 모두에서 메시징 동작을 지정하는 데 사용됩니다.

지점간 메시징

지점간 도메인에서는 메시지 제작자를 **발신자**라고 하고 소비자를 **수신자**라고 합니다. 메시지 제작자와 소비자는 **대기열**이라는 대상을 통해 메시지를 교환합니다. 즉, 발신자가 대기열에 메시지를 **생성**하고 수신자가 대기열의 메시지를 **소비**합니다.

그림 2–1에서는 지점간 도메인에서 가장 간단한 메시징 작업을 보여 줍니다. `MyQueueSender`는 `Msg1`을 대기열 대상 `MyQueue1`로 보냅니다. 그러면 `MyQueueReceiver`가 `MyQueue1`에서 메시지를 가져옵니다.

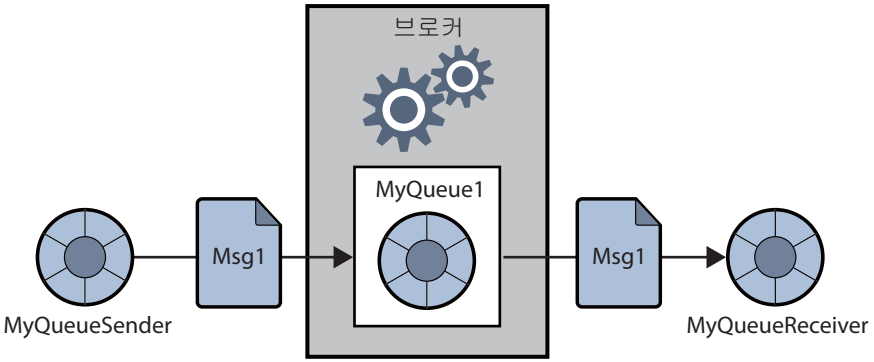

그림 2–1 간단한 지점간 메시징

그림 2–2에서는 더 복잡한 지점간 메시징 그림을 사용하여 이 도메인에서의 가능성을 보여 줍니다. 발신자인 `MyQSender1`과 `MyQSender2`가 동일한 연결을 사용하여 메시지를 `MyQueue1`로 보냅니다. `MyQSender3`은 추가 연결을 사용하여 메시지를 `MyQueue1`로 보냅니다. 수신하는 쪽에서는 `MyQReceiver1`이 `MyQueue1`의 메시지를 소비하며 `MyQReceiver2`와 `MyQReceiver3`은 연결을 공유하면서 `MyQueue1`의 메시지를 소비합니다.

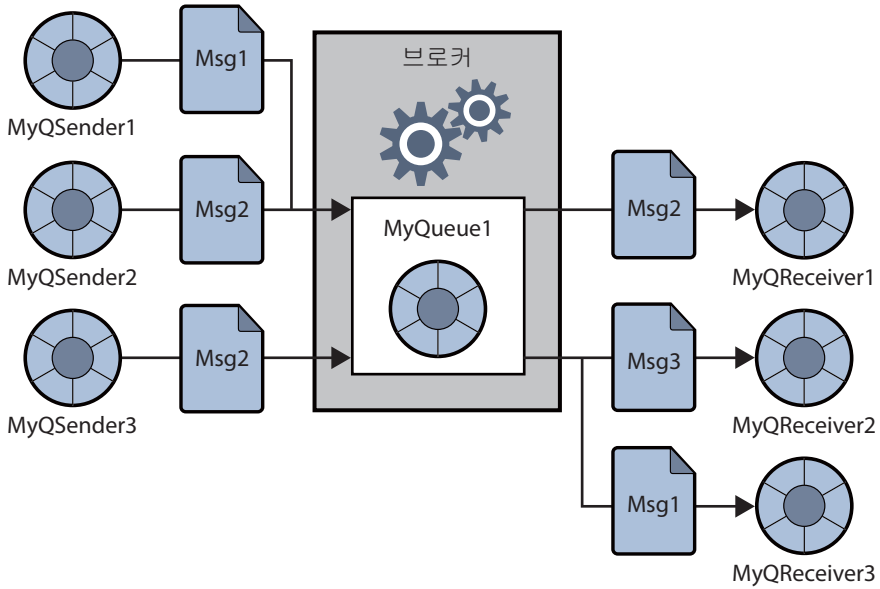

그림 2–2 복잡한 지점간 메시징

더욱 복잡한 이 그림은 지점간 메시징에 대한 많은 추가 사항을 나타냅니다.

- 여러 제작자가 동일한 대기열에 메시지를 보낼 수 있습니다. 제작자는 하나의 연결을 공유하거나 여러 연결을 사용할 수 있지만, 모두 동일한 대기열에 액세스 가능합니다.
- 여러 수신자가 동일한 대기열의 메시지를 소비할 수 있지만, 각 메시지는 한 명의 수신자에 의해서만 소비될 수 있습니다. 따라서 `Msg1`, `Msg2` 및 `Msg3`은 서로 다른 수신자가 소비합니다(Message Queue의 확장).
- 수신자는 하나의 연결을 공유하거나 여러 연결을 사용할 수 있지만, 모두 동일한 대기열에 액세스 가능합니다. (Message Queue의 확장).
- 발신자와 수신자는 타이밍에 구애 받지 않습니다. 즉, 수신자는 클라이언트가 메시지를 보낸 시점에서의 실행 여부와 관계없이 메시지를 불러올 수 있습니다.
- 발신자와 수신자를 런타임에 동적으로 추가 및 삭제할 수 있으므로 필요에 따라 메시징 시스템을 확대하거나 축소할 수 있습니다.
- 메시지는 보낸 순서대로 대기열에 배치되지만 소비되는 순서는 메시지 만료일, 메시지 우선 순위, 메시지 소비 시 선택기 사용 여부와 같은 요소에 따라 다릅니다.

지점간 모델은 다음과 같은 많은 장점이 있습니다.

- 여러 수신자가 동일한 대기열의 메시지를 소비할 수 있으므로, 메시지가 수신되는 순서가 중요하지 않은 경우 메시지 소비 로드 균형을 조정할 수 있습니다(Message Queue의 확장).
- 대기열을 대상으로 하는 메시지는 수신자가 없더라도 항상 보관됩니다.

- Java 클라이언트에서는 대기열 브라우저 객체를 사용하여 대기열의 내용을 조사할 수 있습니다. 그런 다음 이 조사를 통해 수집한 정보를 기반으로 메시지를 소비할 수 있습니다. 즉, 소비 모델은 일반적으로 선입선출(FIFO)이지만 소비자는 원하는 메시지를 알고 있는 경우 메시지 선택기를 사용하여 대기열의 헤드에 없는 메시지를 소비할 수 있습니다. 또한 관리 클라이언트도 대기열 브라우저를 사용하여 대기열의 내용을 모니터링할 수 있습니다.

게시/가입 메시징

게시/가입 도메인에서는 메시지 제작자를 **게시자**라고 하고 메시지 소비자를 **가입자**라고 합니다. 게시자와 가입자는 **주제**라는 대상을 통해 메시지를 교환합니다. 즉, 게시자는 주제에 대한 메시지를 생성하고 가입자는 주제에 **가입**한 다음 해당 주제에서 메시지를 **소비**합니다.

그림 2–3에서는 게시/가입 도메인의 간단한 메시징 작업을 보여 줍니다. `MyTopicPublisher`는 `Msg1`을 대상인 `MyTopic`에 게시합니다. 그러면 `MyTopicSubscriber1`과 `MyTopicSubscriber2`가 각각 `MyTopic`에서 `Msg1`의 복사본을 받습니다.

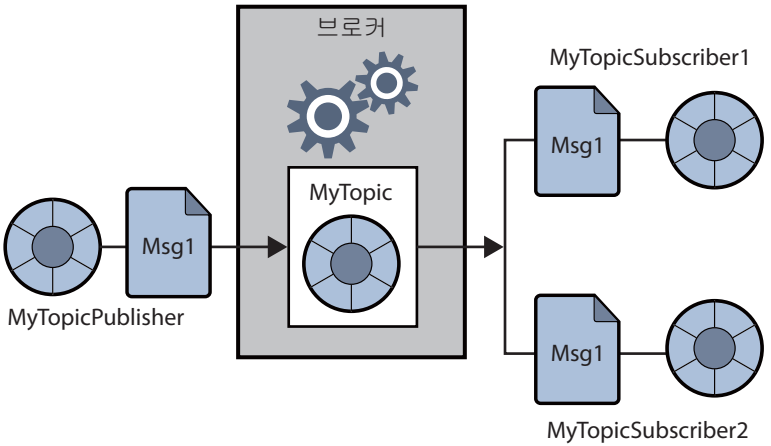

그림 2–3 간단한 게시/가입 메시징

게시/가입 모델에는 여러 명의 가입자가 필요하지 않지만, 그림에서는 이 도메인을 사용하여 메시지를 브로드캐스트할 수 있다는 사실을 강조하기 위해 두 명의 가입자를 표시했습니다. 주제에 대한 모든 가입자가 해당 주제에 게시된 메시지의 복사본을 갖습니다.

가입자는 비영구 가입자일 수도 있고 영구 가입자일 수도 있습니다. 브로커는 모든 활성 가입자에 대한 메시지를 보관하지만, 활성 가입자가 영구 가입자인 경우에는 비활성 가입자에 대한 메시지만 보관합니다.

그림 2–4에서는 이 패턴이 제공하는 가능성을 설명하기 위해 더 복잡한 게시/가입 메시징 그림을 보여 줍니다. 여러 명의 제작자가 `Topic1` 대상에 메시지를 게시합니다. 여러 명의 가입자가 `Topic1` 대상의 메시지를 소비합니다. 가입자가 선택기를 사용하여 메시지를 필터링하지 않는 한, 각 가입자는 선택한 주제에 게시된 모든 메시지를 얻게 됩니다. 그림 2–4에서 `MyTSubscriber2`는 `Msg2`를 필터링했습니다.

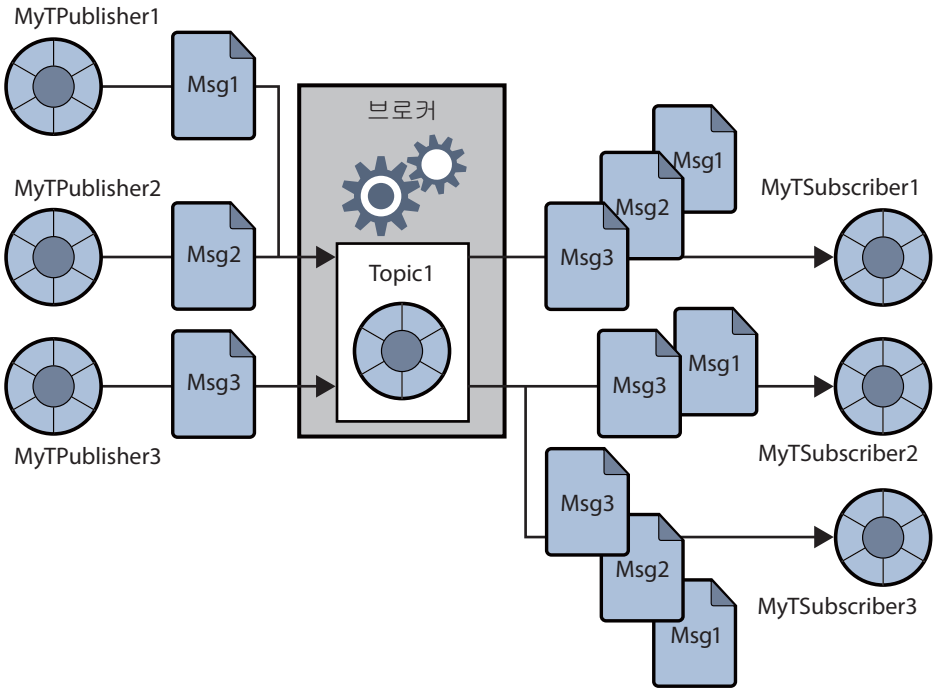

그림 2–4 복잡한 게시/가입 메시징

더욱 복잡한 이 그림은 게시/가입자 메시징에 대한 많은 추가 사항을 나타냅니다.

- 여러 제작자가 동일한 주제에 메시지를 게시할 수 있습니다. 제작자는 하나의 연결을 공유하거나 여러 연결을 사용할 수 있지만, 모두 동일한 주제에 액세스 가능합니다.
- 여러 가입자가 동일한 주제의 메시지를 소비할 수 있습니다. 선택기를 사용하여 메시지를 필터링하거나 사용하기 전에 메시지가 만료된 경우가 아니면, 가입자는 주제에 게시된 모든 메시지를 검색합니다.
- 가입자는 하나의 연결을 공유하거나 여러 연결을 사용할 수 있지만, 모두 동일한 주제에 액세스 가능합니다.
- 영구 가입자는 활성 가입자이거나 비활성 가입자일 수 있습니다. 브로커는 비활성 가입자에 대한 메시지를 보관합니다.
- 게시자와 가입자를 런타임에 동적으로 추가 및 삭제할 수 있으므로 필요에 따라 메시징 시스템을 확대하거나 축소할 수 있습니다.

- 메시지는 보낸 순서대로 주제에 게시되지만 소비되는 순서는 메시지 만료일, 메시지 우선 순위, 메시지를 소비하는 데 선택기가 사용되는지 여부 등과 같은 요소에 따라 다릅니다.
- 게시자와 가입자는 타이밍에 구애를 받습니다. 즉, 주제 가입자는 가입한 이후에 게시된 메시지만 소비할 수 있습니다.

게시/가입 모델의 가장 큰 이점은 메시지를 가입자에게 브로드캐스트할 수 있다는 점입니다.

도메인별 API 및 통합 API

JMS API는 지점간 도메인이나 게시/가입 도메인을 구현하는 데 사용할 수 있는 인터페이스와 클래스를 정의합니다. 표 2–1의 2열과 3열에는 **도메인별** API가 표시되어 있습니다. JMS API는 일반 메시징 클라이언트를 프로그래밍할 수 있는 추가 **통합** 도메인을 정의합니다. 이런 클라이언트의 동작은 메시지를 생성하고 소비하는 대상의 유형에 따라 결정됩니다. 메시징은 대상이 대기열인 경우에는 지점간 패턴에 따라 동작하고, 대상이 주제인 경우에는 게시/가입 패턴에 따라 동작합니다.

표 2–1 JMS 프로그래밍 도메인 및 객체

기본 유형(통합 도메인)	지점간 도메인	도메인 게시/가입
`Destination`(대기열 또는 주제)	`Queue`	`Topic`
`ConnectionFactory`	`QueueConnectionFactory`	`TopicConnectionFactory`
`Connection`	`QueueConnection`	`TopicConnection`
`Session`	`QueueSession`	`TopicSession`
`MessageProducer`	`QueueSender`	`TopicPublisher`
`MessageConsumer`	`QueueReceiver`	`TopicSubscriber`

통합 도메인은 JMS 버전 1.1에서 소개되었습니다. 이전의 1.02b 사양을 준수해야 할 경우 도메인별 API를 사용할 수 있습니다. 또한 도메인별 API를 사용하면 대기열 대상에 영구 가입자를 만드는 등 특정 유형의 프로그래밍 오류를 방지하는 깨끗한 프로그래밍 인터페이스의 장점이 있습니다. 하지만 도메인별 API는 동일한 트랜잭션이나 동일한 세션에서 지점간 및 게시/가입 작업을 결합할 수 없다는 단점이 있습니다. 이러한 결합이 필요한 경우 통합 도메인 API를 선택해야 합니다. 두 도메인 결합의 예는 46 페이지 “요청-응답 패턴”을 참조하십시오.

프로그래밍 객체

연결 팩토리, 연결, 세션, 제작자, 소비자, 메시지, 대상과 같이 JMS 메시징을 구현하는 데 사용되는 객체는 프로그래밍 도메인 전체에서 동일하게 유지되어야 합니다. 이 객체는 그림 2–5에 나와 있습니다. 이 그림에서는 연결 팩토리 객체부터 시작하여 객체가 파생되는 방법을 위에서 아래로 보여 줍니다.

객체 저장소에는 두 개의 객체(연결 팩토리와 대상)가 있습니다. 이는 이 객체들이 일반적으로 관리 대상 객체로 생성, 구성되고 관리됨을 나타냅니다. 이 장에서는 연결 팩토리와 대상이 프로그래밍 방식이 아니라 관리 목적으로 생성된다고 가정합니다.

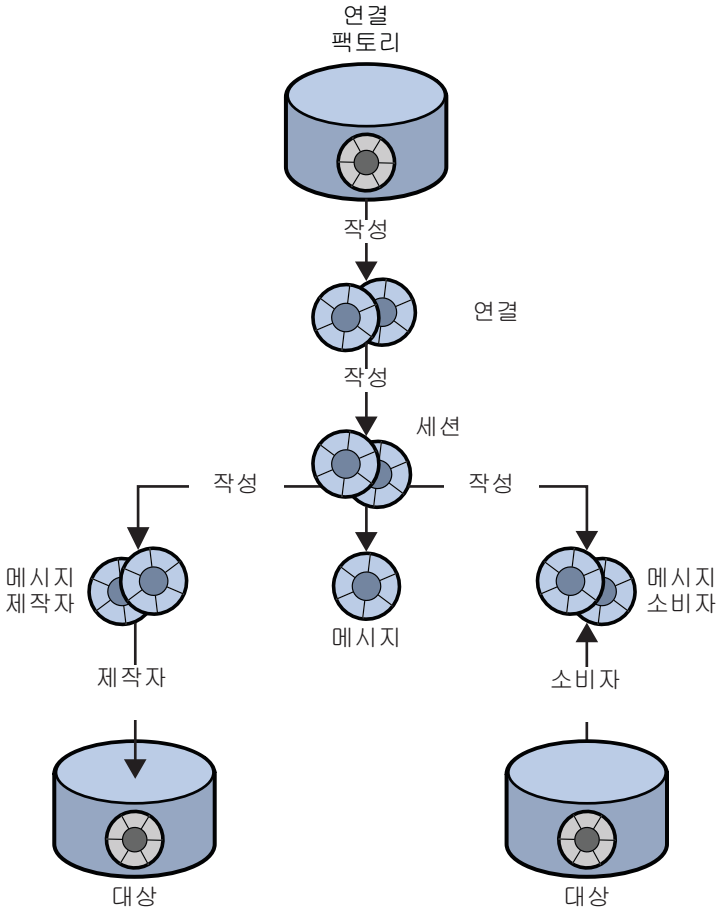

그림 2–5 JMS 프로그래밍 객체

표 2–2에서는 메시지를 보내고 받는 데 필요한 단계를 요약합니다. 1-6단계는 발신자와 수신자 모두에게 동일합니다.

표 2–2 메시지 생성 및 소비

메시지 생성	메시지 소비
1. **관리자가 연결 팩토리 관리 대상 객체를 만듭니다.**	
2. **관리자가 물리적 대상과 해당 대상을 참조하는 관리 대상 객체를 만듭니다.**	
3. **클라이언트가 *JNDI* 조회를 통해 연결 팩토리 객체를 가져옵니다.**	
4. **클라이언트가 *JNDI* 조회를 통해 대상 객체를 가져옵니다.**	
5. **클라이언트가 연결을 만들고 이 연결에 관한 등록 정보를 설정합니다.**	
6. **클라이언트가 세션을 만들고 메시징 안정성을 제어하는 등록 정보를 설정합니다.**	
7. 클라이언트가 메시지 제작자를 만듭니다.	클라이언트가 메시지 소비자를 만듭니다.
8. 클라이언트가 메시지를 작성합니다.	클라이언트가 연결을 시작합니다.
9. 클라이언트가 메시지를 보냅니다.	클라이언트가 메시지를 받습니다.

다음 절에서는 연결, 세션, 메시지, 대상과 같이 제작자와 소비자가 소비하는 객체에 대해 설명합니다. 그런 다음 메시지의 생성과 소비를 설명하여 JMS 객체에 대한 내용을 마치겠습니다.

연결 팩토리 및 연결

클라이언트는 **연결 팩토리** 객체(`ConnectionFactory`)를 사용하여 **연결**을 만듭니다. 연결 객체(`Connection`)는 클라이언트와 브로커 간의 활성 연결을 나타냅니다. 이 연결 객체는 기본적으로 시작되거나 이 클라이언트의 관리자가 명시적으로 시작하는 기본 연결 서비스를 사용합니다.

연결이 만들어지면 통신 자원 할당 및 클라이언트 인증이 이루어집니다. 이는 비교적 중량급 객체이며, 대부분의 클라이언트는 단일 연결을 사용하여 모든 메시징을 수행합니다. 연결은 동시 사용을 지원합니다. 제작자와 소비자는 그 수의 제한 없이 동일한 연결을 공유할 수 있습니다.

연결 팩토리를 만들 때 해당 등록 정보를 설정하여 연결 팩토리에서 파생되는 모든 연결의 동작을 구성할 수 있습니다. Message Queue의 경우 등록 정보를 통해 다음과 같은 정보를 지정합니다.

- 브로커가 있는 호스트의 이름, 원하는 연결 서비스, 클라이언트가 해당 서비스에 액세스하는 데 사용하는 포트
- 연결이 실패할 경우 브로커에 자동으로 다시 연결하는 방법. 이 기능은 연결이 끊어진 경우에 클라이언트를 동일한 브로커나 다른 브로커에 다시 연결합니다. 데이터 페일오버는 보장되지 않습니다. 다른 브로커에 다시 연결할 때 지속성 메시지와 기타 상태 정보가 손실될 수 있습니다.)

- 브로커가 영구 가입을 추적하는 데 필요한 클라이언트의 아이디
- 연결을 시도하는 사용자의 기본 이름 및 비밀번호. 이 정보는 연결 시 비밀번호를 지정하지 않은 경우에 사용자를 인증하고 작업 권한을 부여하는 데 사용됩니다.
- 안정성을 고려하지 않는 클라이언트에 대해 브로커 확인을 억제할지 여부
- 브로커와 클라이언트 런타임 간의 제어 및 페이로드 메시지의 흐름을 관리하는 방법
- 대기열 찾아보기 처리 방법(Java 클라이언트만 해당)
- 특정 메시지 헤더 필드를 대체할지 여부

클라이언트 응용 프로그램을 시작할 때 사용하는 명령줄에서 연결 팩토리 등록 정보를 대체할 수 있습니다. 해당 연결에 대한 등록 정보를 설정하여 주어진 연결에 대한 등록 정보를 대체할 수도 있습니다.

연결 객체를 사용하여 **세션** 객체를 만들거나, 예외 수신자를 설정하거나, JMS 버전 및 공급자 정보를 가져올 수 있습니다.

세션

연결이 클라이언트와 브로커 간의 통신 채널을 나타내는 경우 세션에 클라이언트와 브로커 간의 단일 대화가 표시됩니다. 일반적으로 세션 객체를 사용하여 메시지, 메시지 제작자 및 메시지 소비자를 만듭니다. 세션을 만들 경우 많은 **확인** 옵션 또는 트랜잭션을 통해 안정적인 전달을 구성합니다. 자세한 내용은 47 페이지 “안정적인 메시징”을 참조하십시오.

JMS 사양에 따르면 세션은 메시지 생성 및 소비를 위한 단일 스레드 컨텍스트입니다. 단일 세션에 대해 여러 메시지 제작자와 소비자를 만들 수 있지만, 해당 제작자와 소비자를 순차적으로 사용해야 합니다. 스레딩 구현은 Java 클라이언트와 C 클라이언트 간에 약간 다릅니다. 스레딩 구현 및 제한에 대한 자세한 내용은 해당 개발 안내서를 참조하십시오.

세션 객체를 사용하여 다음을 수행할 수도 있습니다.

- 관리 대상 객체를 사용하여 대상을 정의하지 않는 클라이언트에 대한 대상 만들기 및 구성
- 임시 주제와 대기열 만들기 및 구성. 임시 주제와 대기열은 요청-응답 패턴의 일부로 사용됩니다. 46 페이지 “요청-응답 패턴”을 참조하십시오.
- 트랜잭션 처리 지원
- 메시지 생성 또는 소비를 위한 일련 순서 정의
- 비동기식 소비자에 대한 메시지 수신기 실행 일련화
- 대기열 브라우저를 만듭니다. (Java 클라이언트만 해당)

메시지

메시지는헤더, 등록 정보 및 본문으로 구성됩니다.이 구조를 이해해야 메시지를 적절하게 작성하고 특정 메시징 동작을 구성할 수 있습니다.

메시지 헤더

헤더는 모든 JMS 메시지에서 필수입니다. 헤더에는 10개의 사전 정의된 필드가 포함되어 있습니다. 이러한 필드에 대한 목록과 설명은 표 2–3을 참조하십시오.

표 2–3 JMS 정의 메시지 헤더

헤더 필드	설명
JMSDestination	메시지를 보낼 대상 객체의 이름을 지정합니다. (공급자가 설정)
JMSDeliveryMode	메시지가 지속성 메시지인지 여부를 지정합니다. (기본적으로 공급자가 설정하거나, 클라이언트가 제작자 또는 개별 메시지에 대해 명시적으로 설정)
JMSExpiration	메시지가 만료되는 시간을 지정합니다. (기본적으로 공급자가 설정하거나, 클라이언트가 제작자 또는 개별 메시지에 대해 설정)
JMSPriority	0(낮음)-9(높음) 범위에서 메시지의 우선 순위를 지정합니다. (기본적으로 공급자가 설정하거나, 클라이언트가 제작자 또는 개별 메시지에 대해 명시적으로 설정)
JMSMessageID	공급자 설치 컨텍스트 내에서 메시지에 대한 고유 아이디를 지정합니다. (공급자가 설정)
JMSTimestamp	공급자가 메시지를 받은 시간을 지정합니다. (공급자가 설정)
JMSCorrelationID	클라이언트가 두 메시지 간의 통신을 정의하는 데 사용하는 값(클라이언트가 필요에 따라 설정)
JMSReplyTo	소비자가 회신을 보낼 대상을 지정합니다. (클라이언트가 필요에 따라 설정)
JMSType	메시지 선택기에서 평가할 수 있는 값(클라이언트가 필요에 따라 설정)
JMSRedelivered	메시지가 이미 전달되었지만 확인되지 않았는지 여부를 나타냅니다. (공급자가 설정)

이 표에서 알 수 있듯이 메시지 헤더 필드는메시지 식별, 메시지 라우팅 구성, 메시지 처리 정보 제공과 같은 다양한 목적으로 사용됩니다.

가장 중요한 필드 중 하나인 JMSDeliveryMode가 메시지 전달의 안정성을 결정합니다. 이 필드는 메시지가 지속성 메시지인지 여부를 나타냅니다.

- **지속성 메시지**정확히 한 번의 전달 및 성공적인 소비이 보장됩니다. 지속성 메시지는 메시지 서비스 오류 발생 시에도 손실되지 않습니다.
- **비지속성 메시지**는 최대 한 차례 전달이 보장됩니다. 비지속성 메시지는 메시지 서비스 오류 발생 시 손실됩니다.

일부 메시지 헤더 필드는 공급자(브로커 또는 클라이언트 런타임) 또는 클라이언트에 의해 설정됩니다. 메시지 제작자는 특정 메시징 동작을 가져올 헤더 값을 구성해야 합니다. 메시지 소비자는 헤더 값을 읽고 메시지 경로가 지정된 방법과 향후의 메시지 처리에 필요한 내용을 이해해야 합니다.

헤더 필드(`JMSDeliveryMode`, `JMSExpiration` 및 `JMSPriority`)는 세 가지 수준에서 설정할 수 있습니다.

- 연결 팩토리에서 파생되는 모든 연결로부터 생긴 메시지
- 생성된 각 메시지
- 특정 메시지 제작자가 만든 모든 메시지

이러한 필드가 여러 수준으로 설정되는 경우, 연결 팩토리에 설정된 값이 개별 메시지에 설정된 값을 대체하며 주어진 메시지에 설정된 값이 메시지 제작자에 설정된 값을 대체합니다.

메시지 헤더 필드의 이름은 언어 구현에 따라 다릅니다. 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients** 또는 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for C Clients**를 참조하십시오.

메시지 등록 정보

메시지는 등록 정보 이름과 등록 정보 값 쌍으로 지정되는 등록 정보라는 선택적 헤더 필드를 포함할 수도 있습니다. 클라이언트와 공급자는 이 등록 정보를 통해 메시지 헤더를 확장할 수 있으며, 등록 정보에는 클라이언트나 공급자가 메시지를 식별하여 처리하는 데 유용한 정보가 들어 있을 수 있습니다. 수신 클라이언트는 메시지 등록 정보를 사용하여 지정된 기준에 맞는 메시지만 전달하도록 요청할 수 있습니다. 예를 들어 소비자 클라이언트는 뉴저지에 있는 시간제 직원에 대한 급여 메시지에 대해서만 관심 분야를 표시할 수 있습니다. 공급자는 지정된 기준에 맞지 않는 메시지를 전달하지 않습니다.

JMS 사양에서는 아홉 가지 표준 등록 정보를 정의합니다. 이러한 등록 정보 중 일부는 클라이언트가, 다른 일부는 공급자가 설정합니다. 등록 정보의 이름은 예약된 문자 "JMSX"로 시작됩니다.클라이언트나 공급자는 이러한 등록 정보를 사용하여 메시지를 보낸 사람, 메시지 상태, 메시지 전달 빈도, 메시지가 전달된 시간 등을 확인할 수 있습니다. 이러한 등록 정보를 통해 공급자는 메시지 경로를 지정하고 진단 정보를 제공할 수 있습니다.

Message Queue에서는 메시지 등록 정보도 정의합니다. 메시지 등록 정보는 메시지를 전달할 수 없을 경우의 메시지 처리 방법과 압축된 메시지를 식별하는 데 사용됩니다. 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients**의 "Managing Message Size"를 참조하십시오.

메시지 본문

메시지 본문에는 클라이언트가 교환하고자 하는 데이터가 포함되어 있습니다.

표 2–4에 나와 있는 것처럼 JMS 메시지의 유형에 따라 본문에 포함되는 내용과 소비자가 본문을 처리하는 방법이 결정됩니다. 세션 객체에는 각 메시지 본문 유형에 대한 생성 메소드가 포함되어 있습니다.

표 2–4 메시지 본문 유형

유형	설명
`StreamMessage`	본문이 Java 프리미티브 값의 스트림을 포함하는 메시지. 이 메시지는 순차적으로 채워지고 읽혀집니다.
`MapMessage`	본문에 일련의 이름-값 쌍을 포함하는 메시지. 항목 순서는 정의되지 않습니다.
`TextMessage`	본문에 Java 문자열을 포함하는 메시지(예: XML 메시지)
`ObjectMessage`	본문에 일련화된 Java 객체를 포함하는 메시지
`BytesMessage`	본문에 해석되지 않은 바이트의 스트림이 포함된 메시지
`Message`	헤더와 등록 정보는 있지만 본문이 없는 메시지

Java 클라이언트는 클라이언트 런타임에서 전송 중인 메시지의 본문을 압축하도록 등록 정보를 설정할 수 있습니다. 소비자측의 Message Queue 런타임에서 메시지를 전달하기 전에 압축을 풉니다.

메시지 생성

메시지 제작자는 연결 및 세션 컨텍스트 내에서 메시지를 전송하거나 게시합니다. 메시지 생성은 매우 간단합니다. 클라이언트는 메시지 제작자 객체(`MessageProducer`)를 사용하여 물리적 **대상**(API에서는 대상 객체로 표현됨)으로 메시지를 보냅니다.

제작자를 만들 때 모든 제작자의 메시지가 전송되는 기본 대상을 지정할 수 있습니다. 지속성, 우선 순위, 수명 등을 제어하는 메시지 헤더 필드에 대한 기본값을 지정할 수도 있습니다. 그러면 이 기본값은 메시지를 보낼 때 대체 대상을 지정하거나 지정된 메시지의 헤더 필드에 대한 대체 값을 설정하여 기본값을 대체하지 않는 한 해당 제작자가 만든 모든 메시지에서 사용됩니다.

메시지 제작자는 `JMSReplyTo` 메시지 헤더 필드를 설정하여 요청-응답 패턴을 구현할 수도 있습니다. 자세한 내용은 46 페이지 “요청-응답 패턴”을 참조하십시오.

메시지 소비

메시지 소비자는 연결 및 세션 컨텍스트 내에서 메시지를 받습니다. 클라이언트는 메시지 소비자 객체(MessageConsumer)를 사용하여 지정된 물리적 대상(API에서는 대상 객체로 표현됨)으로부터 메시지를 받습니다.

다음 세 가지 요소는 브로커가 소비자에게 메시지를 전달하는 방법에 영향을 미칩니다.

- 소비이 동기식인지 비동기식인지 여부
- 선택기를 사용하여 들어오는 메시지를 필터링하는지 여부
- 가입자가 영구 가입자인지 여부(메시지가 주제 대상에서 소비되는 경우)

메시지 전달 및 클라이언트 설계에 영향을 미치는 다른 중요한 요소로는 소비자에게 필요한 안정성 수준이 있습니다. 47 페이지 “안정적인 메시징”을 참조하십시오.

동기식 및 비동기식 소비자

메시지 소비자는 동기식 또는 비동기식 메시지 소비를 지원할 수 있습니다.

- 동기식 소비는 소비자가 메시지 전달을 명시적으로 요청한 다음 메시지를 소비해야 함을 의미합니다.

  메시지를 요청하는 데 사용되는 메소드에 따라 동기식 소비자는 메시지가 도착할 때까지 무한히 대기하거나, 지정된 시간 동안 메시지를 기다리거나, 소비 준비가 된 메시지가 없는 경우 즉시 돌아가도록 선택할 수 있습니다. ("소비됨"은 해당 객체를 클라이언트에서 즉시 사용할 수 있음을 의미합니다. 성공적으로 전송되었지만 브로커가 처리를 마치지 않은 메시지는 소비할 수 없습니다.)
- 비동기식 소비는 메시지가 소비자용으로 등록된 메시지 수신기 객체(MessageListener)로 자동 전달됨을 의미합니다. 세션 스레드가 메시지 수신기 객체의 onMessage() 메소드를 호출하면 클라이언트가 메시지를 소비합니다.

선택기를 사용하여 메시지 필터링

메시지 소비자는 메시지 선택기를 사용하여 등록 정보가 특정 선택 기준과 일치하는 메시지만 메시지 서비스에서 전달하게 할 수 있습니다. 소비자를 만들 때 이 기준을 지정합니다.

선택기는 SQL-like 구문을 사용하여 메시지 등록 정보에 대해 일치시킵니다. 예를 들면 다음과 같습니다.

```
color =   red’
size > 10
```

Java 클라이언트는 대기열을 찾아볼 때 선택기를 지정할 수도 있습니다. 그러면 소비 대기 중인 선택된 메시지를 확인할 수 있습니다.

영구 가입자 사용

세션 객체를 사용하여 주제에 대한 영구 가입자를 만들 수 있습니다. 브로커는 가입자가 비활성화된 경우에도 이러한 종류의 가입자에 대한 메시지를 보관합니다.

브로커는 가입자의 상태를 유지하였다가 가입자가 다시 활성화되면 메시지를 다시 전달해야 하므로, 지정된 가입자가 들어오고 나가는 것을 식별할 수 있어야 합니다. 가입자의 아이디는 가입자를 만든 연결의 `ClientID` 등록 정보와 가입자를 만들 때 지정한 가입자 이름으로부터 구성됩니다.

요청-응답 패턴

동일한 연결 또는 세션(통합 API를 사용하는 경우)에서 제작자와 소비자를 결합할 수 있습니다. 또한 JMS API를 통해 임시 대상을 사용하여 메시징 작업에 대한 요청-응답 패턴을 구현할 수 있습니다.

요청-응답 패턴을 설정하려면 다음 작업을 수행해야 합니다.

1. 소비자가 응답을 보낼 수 있는 임시 대상을 만듭니다.
2. 보낼 메시지에서 메시지 헤더의 `JMSReplyTo` 필드를 해당 임시 대상으로 설정합니다.

메시지 소비자가 메시지를 처리할 때 메시지의 `JMSReplyTo` 필드를 검사하여 응답이 필요한지를 결정하고 지정된 대상에게 응답을 보냅니다.

요청-응답 메커니즘은 제작자가 응답 대상에 대한 관리 대상 객체를 설정해야 하는 번거로움을 덜어주고, 소비자가 요청에 쉽게 응답할 수 있게 해 줍니다. 이 패턴은 제작자가 계속하기 전에 요청이 처리되었는지 확인해야 하는 경우에 유용합니다.

그림 2–6에서는 주제에 메시지를 보내고 임시 대기열에서 응답을 받는 요청-응답 패턴에 대해 설명합니다.

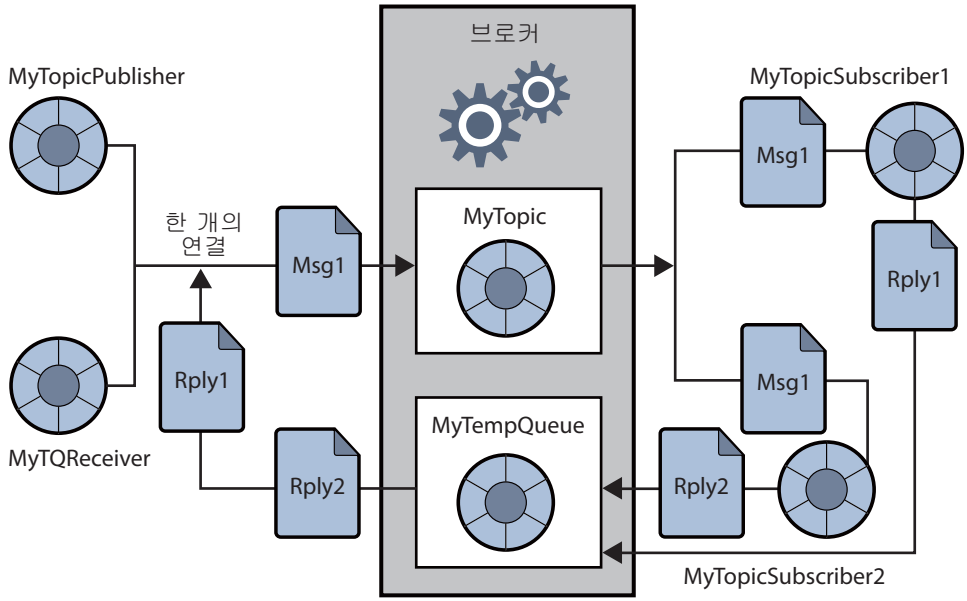

그림 2–6 요청/응답 패턴

그림에 표시된 것처럼 `MyTopicPublisher`는 `Msg1`을 대상 `MyTopic`에 생성합니다. `MyTopicSubsriber1`과 `MyTopicSubscriber2`는 메시지를 받고 `MyTempQueue`에 응답을 보냅니다. `MyTQReceiver`는 이 대기열에서 응답을 검색합니다. 이 패턴은 다수의 클라이언트에 가격 정보를 게시하고 순차적 처리를 위해 (응답) 주문을 대기열에 넣는 응용 프로그램에 유용할 수 있습니다.

임시 대상은 대상을 만든 연결 기간 동안만 지속됩니다. 제작자는 임시 대상에 보낼 수 있지만, 대상을 만든 동일한 연결에 의해 생성되는 것은 임시 대상에 액세스할 수 있는 소비자뿐입니다.

요청/응답 패턴은 임시 대상 만들기에 따라 다르므로 다음과 같은 경우에는 이 패턴을 사용하지 않아야 합니다.

- 임시 대상을 만드는 연결이 응답을 보내기 전에 종료될 수 있는 경우
- 임시 대상에 지속성 메시지를 보내야 하는 경우

안정적인 메시징

메시지 전달은 두 개의 홉에서 발생합니다. 첫 번째 홉은 제작자의 메시지를 브로커의 물리적 대상으로 가져오고, 두 번째 홉은 물리적 대상의 메시지를 소비자에게 가져옵니다. 따라서 브로커로 이동하는 홉, 브로커가 실패할 때 브로커 메모리에 있는 홉 그리고 브로커에서 소비자로 이동하는 홉 중 하나에 해당할 경우 메시지가 손실될 수

있습니다. 안정적인 전달에서는 이러한 경우에도 전달이 실패하지 않습니다. 비지속성 메시지는 브로커가 실패할 경우 항상 손실될 수 있으므로 안정적인 전달은 지속성 메시지에만 적용됩니다.

다음 두 메커니즘을 사용하여 안정적인 전달을 확실히 합니다.

- 클라이언트는 **확인** 또는 **트랜잭션**을 사용하여 메시지가 성공적으로 생성 및 소비되는지 확인합니다.
- 브로커는 영구 저장소에 메시지를 저장할 수 있으므로 메시지가 소비되기 전에 브로커가 실패할 경우 저장된 메시지 복사본을 검색하여 다시 작업할 수 있습니다.

다음 절에서는 안정성을 보장하는 이러한 2가지 측면에 대해 설명합니다.

확인

확인은 클라이언트와 메시지 서비스 간에 안정적인 메시지 전달을 확인하기 위해 보내는 메시지입니다. 확인은 제작자와 소비자에서 서로 다르게 사용됩니다.

메시지 생성 시, 브로커는 메시지를 받아서 대상에 넣고 영구적으로 저장했음을 확인합니다. 제작자의 `send()` 메소드는 이 확인을 받을 때까지 차단됩니다. 이러한 확인은 지속성 메시지를 보낼 때 클라이언트에 대해 투명합니다.

메시지 소비 시, 클라이언트가 대상으로부터의 메시지 전달을 수신하고 메시지를 소비했음을 확인한 다음 브로커가 해당 대상에서 메시지를 삭제합니다. JMS는 서로 다른 안정성 정도를 나타내는 다양한 확인 모드를 지정합니다.

- `AUTO_ACKNOWLEDGE` 모드에서 세션은 클라이언트에서 소비하는 각 메시지를 자동으로 확인합니다. 브로커가 소비된 각 메시지에 대해 클라이언트 확인을 처리했음을 확인할 때까지 기다리는 동안 세션 스레드는 차단됩니다.
- `CLIENT_ACKNOWLEDGE` 모드에서 클라이언트는 메시지 객체의 `acknowledge()` 메소드를 호출하여 하나 이상의 메시지를 소비한 후 명시적으로 확인합니다. 이것은 세션이 메소드의 이전 호출 이후에 세션에 의해 소비된 모든 메시지를 확인하기 때문입니다. 또한 브로커가 클라이언트 확인을 처리했음을 확인할 때까지 세션 스레드는 차단됩니다.

  Message Queue는 클라이언트가 하나의 메시지에 대해서만 수신을 확인할 수 있는 메소드를 제공하여 이 모드를 확장합니다.
- `DUPS_OK_ACKNOWLEDGE` 모드에서 세션은 메시지를 10개 소비한 후 확인합니다. 이 모드에서는 브로커 확인이 필요하지 않으므로 브로커 확인이 완료될 때까지 세션 스레드가 차단되지 않습니다. 이 모드에서는 메시지가 손실되지 않지만 이름이 `DUPS_OK`이기 때문에 중복 메시지가 수신될 수 있습니다.

안정성보다는 성능을 더 고려하는 클라이언트를 위해 Message Queue 서비스는 `NO_ACKNOWLEDGE` 모드를 제공하여 JMS API를 확장합니다. 이 모드에서는 브로커가 클라이언트 확인을 추적하지 않기 때문에 소비자 클라이언트에서 메시지를 성공적으로

처리했는지 확인할 수 없습니다. 이 모드를 선택하면 비영구 가입자에게 보내는 비지속성 메시지에 대한 성능이 향상될 수 있습니다.

트랜잭션

트랜잭션은 하나 이상의 메시지 생성 및/또는 소비를 기본 단위로 묶는 방법입니다. 위에서 설명한 클라이언트 및 브로커 확인 프로세스는 트랜잭션에도 적용됩니다. 이 경우 클라이언트 런타임과 브로커 확인이 트랜잭션 수준에서 암시적으로 수행됩니다. 트랜잭션이 완결되면 브로커 확인이 자동으로 보내집니다.

세션은 **트랜잭션된** 세션으로 구성할 수 있으며, JMS API는 트랜잭션 초기화, 완결 또는 롤백을 위한 메소드를 제공합니다.

트랜잭션 내부에서 메시지를 생성하거나 소비하면 메시지 서비스는 다양한 발신 및 수신을 추적하여, JMS 클라이언트가 트랜잭션을 완결하도록 호출한 경우에만 작업을 완료합니다. 트랜잭션 내부에서 특정 발신 또는 수신 작업이 실패할 경우 예외가 발생합니다. 클라이언트 코드는 예외를 무시하거나 작업을 다시 시도하거나 전체 트랜잭션을 롤백하는 방법으로 예외를 처리할 수 있습니다. 트랜잭션이 완결되면 성공적인 작업이 모두 완료됩니다. 트랜잭션이 롤백되면 성공적인 작업이 모두 취소됩니다.

트랜잭션의 범위는 항상 단일 세션입니다. 즉, 단일 세션 컨텍스트에서 수행되는 하나 이상의 제작자 또는 소비자 작업을 묶어 단일 트랜잭션으로 분류할 수 있습니다. 트랜잭션 범위가 단일 세션에 국한되므로 메시지 생성과 소비를 모두 총괄하는 종단간 트랜잭션은 만들 수 없습니다.

또한 JMS 사양은 **분산** 트랜잭션을 지원합니다. 즉, 메시지 생성 및 소비는 데이터베이스 시스템과 같은 다른 자원 관리자가 관련된 작업을 포함하는, 더 크고 분산된 트랜잭션의 일부가 될 수 있습니다. Java Systems Application Server에 제공된 것과 같은 트랜잭션 관리자는 분산 트랜잭션을 지원할 수 있어야 합니다.

분산 트랜잭션의 경우, 분산 트랜잭션 관리자는 JTA(Java Transaction API)인 XA Resource API 사양에 정의된 2단계 완결 프로토콜을 사용하여 여러 자원 관리자(메시지 서비스, 데이터베이스 관리자 등)가 수행하는 작업을 추적하고 관리합니다. Java에서 자원 관리자 및 분산 트랜잭션 관리자 간의 상호 작용은 JTA 사양에서 설명합니다.

분산 트랜잭션 지원은 메시징 클라이언트가 JTA에서 정의된 `XAResource` 인터페이스를 통해 분산 트랜잭션에 참여할 수 있음을 의미합니다. 이 인터페이스는 2단계 완결을 구현하는 여러 메소드를 정의합니다. 클라이언트측에서 API 호출이 이루어지는 동안 JMS 메시지 서비스는 분산 트랜잭션 내부의 다양한 발신 및 수신 작업을 추적하고 트랜잭션 상태를 추적하며 JTS(Java Transaction Service)가 제공하는 분산 트랜잭션 관리자와의 조정을 통해서만 메시징 작업을 완료합니다. 로컬 트랜잭션과 마찬가지로 클라이언트는 예외를 무시하거나 작업을 다시 시도하거나 전체 분산 트랜잭션을 롤백하는 방법으로 예외를 처리할 수 있습니다.

주 – Message Queue는 Java Enterprise Edition 플랫폼에서 JMS 공급자로 사용되는 경우에 한해 분산 트랜잭션을 지원합니다. 분산 트랜잭션 사용 방법에 대한 자세한 내용은 Application Server 공급자가 제공한 설명서를 참조하십시오.

영구 저장소

안정성의 또 다른 측면은 메시지가 소비자에게 전달될 때까지 브로커가 지속성 메시지를 잃어버리지 않아야 한다는 것입니다. 즉, 메시지가 물리적 대상에 전달되면 브로커는 이를 영구 **데이터 저장소**에 저장해야 합니다. 어떤 이유로 브로커가 중단되는 경우, 브로커는 메시지를 나중에 복구하여 해당 소비자에게 전달할 수 있습니다.

브로커는 영구 가입을 영구히 저장해야 합니다. 그렇지 않으면, 오류 발생 시 브로커는 메시지가 주제 대상에 도달한 다음에 활성화되는 영구 가입자에게 메시지를 전달할 수 없습니다.

메시지 전달을 보장하려는 메시징 응용 프로그램은 메시지가 지속성을 갖도록 지정하고, 이를 영구 가입이 설정된 주제 대상이나 대기열 대상 중 하나로 전달해야 합니다.

3 장에서는 Message Queue 서비스에서 제공한 기본 메시지 저장소 그리고 관리자가 대체 저장소를 설정하고 구성하는 방법에 대해 설명합니다.

시스템에서의 메시지 경로

지금까지 설명한 자료를 요약하여 이 절에서는 Message Queue 서비스를 사용하여 제작자에서 소비자로 메시지를 전달하는 방법에 대해 설명합니다. 완벽한 이해를 위해서는 더 자세한 정보가 필요합니다. 이러한 전달 단계에서 시스템에 의해 처리된 메시지는 두 가지 범주로 구분됩니다.

- **페이로드 메시지** - 제작자가 소비자에게 보낸 메시지
- **제어 메시지** - 페이로드 메시지가 성공적으로 전달되었음을 확인하고 연결을 통한 메시지 흐름을 제어하기 위해 브로커와 클라이언트 런타임 간에 전달되는 개인 메시지

메시지 전달은 그림 2–7에 설명되어 있습니다.

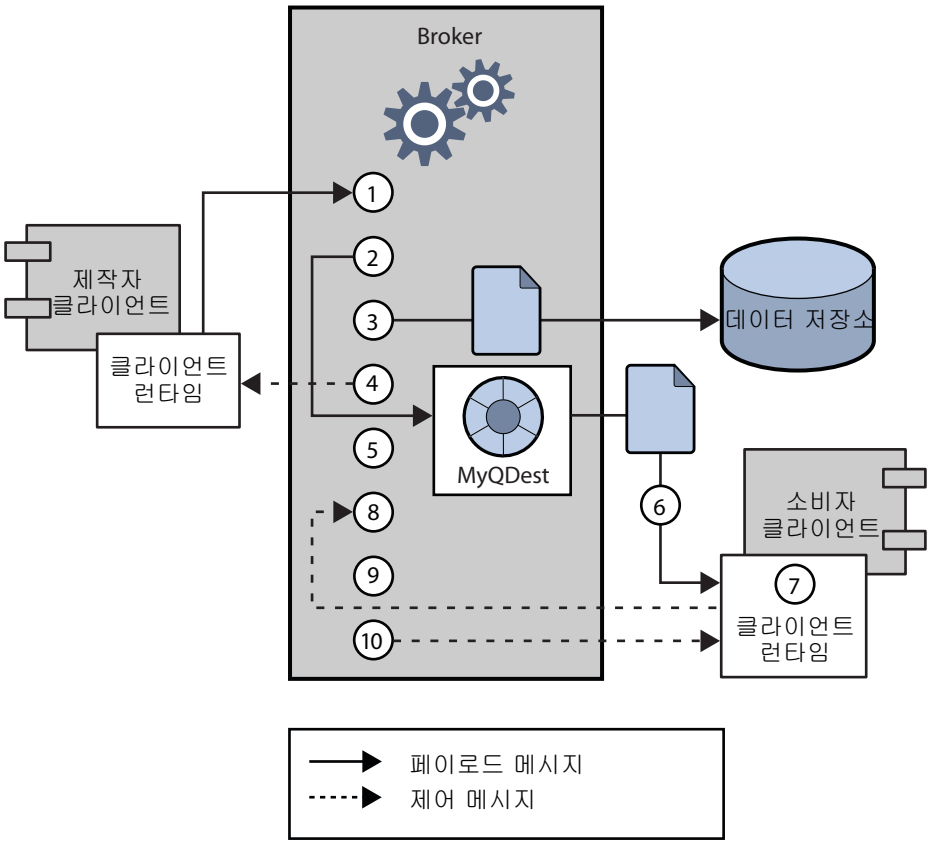

그림 2–7 메시지 전달 단계

안정적으로 전달되는 지속성 메시지의 메시지 전달 단계는 다음과 같습니다.

메시지 생성

1. 클라이언트 런타임이 연결을 통해 메시지 제작자에서 브로커로 메시지를 전달합니다.

메시지 처리 및 경로 지정

2. 브로커가 연결을 통해 메시지를 읽고 적절한 대상에 저장합니다.

3. 브로커가 (지속성) 메시지를 데이터 저장소에 저장합니다.

4. 브로커가 메시지 제작자의 클라이언트 런타임에게 메시지 수신 확인을 보냅니다.

5. 브로커가 메시지 경로 지정을 결정합니다.

6. 브로커가 대상의 메시지를 해당 연결에 기록하여 소비자에 대한 고유 식별자 태그를 붙입니다.

메시지 소비

7. 메시지 소비자의 클라이언트 런타임이 연결에서 메시지 소비자로 메시지를 전달합니다.

8. 메시지 소비자의 클라이언트 런타임이 메시지 소비에 대한 확인을 브로커로 보냅니다.

메시지 수명 끝

9. 브로커가 클라이언트 확인을 처리하고 모든 확인이 수신되면 (지속성) 메시지를 삭제합니다.

10. 브로커가 소비자의 클라이언트 런타임에서 클라이언트 확인이 처리되었는지 확인합니다.

관리자가 대상에서 메시지를 삭제한 경우 또는 관리자가 영구 가입을 제거하거나 다시 정의한 경우, 메시지가 소비되기 전에 브로커가 이를 삭제할 수 있으므로 주제 대상의 메시지가 전달되지 않고 제거되는 상황이 벌어집니다. 이때 브로커가 메시지를 삭제하지 않고 **사용 불능 메시지 대기열**이라는 특수 대상에 메시지를 저장하도록 할 수도 있습니다. 메시지는 만료되는 경우, 메모리 제한으로 인해 제거되거나 클라이언트에서 예외가 발생하여 전달에 실패하면 사용 불능 메시지 대기열에 저장됩니다. 사용 불능 메시지 대기열에 메시지를 저장하여 시스템 문제를 해결하고 특정 상황에서 메시지를 복구할 수 있습니다.

SOAP 메시지 작업

SOAP(29 페이지 "Java 클라이언트에 대한 SOAP 지원" 참조)를 사용하면 분산 환경에서 두 피어 간에 구조화된 데이터(XML 스키마에서 지정)를 교환할 수 있습니다. Sun의 SOAP 구현에서는 현재 안정적인 SOAP 메시징과 SOAP 메시지 게시를 지원하지 않습니다. 그러나 Message Queue 서비스를 사용하여 안정적인 SOAP 메시징을 수행할 수 있으며, 원하는 경우 SOAP 메시지를 게시할 수 있습니다. Message Queue 서비스는 SOAP 메시지를 직접 전달하지 않습니다. 그러나 SOAP 메시지를 JMS 메시지로 래핑하고, 이러한 메시지를 일반 JMS 메시지처럼 생성 및 소비하며, JMS 메시지에서 SOAP 메시지를 추출할 수 있습니다.

Message Queue에서는 `javax.xml.messaging`과 `com.sun.messaging.xml`의 두 가지 패키지를 통해 SOAP 지원을 제공합니다. 이러한 라이브러리에서 구현된 클래스를 사용하여 SOAP 메시지를 수신하고, SOAP 메시지를 JMS 메시지로 래핑하며, JMS 메시지에서 SOAP 메시지를 추출할 수 있습니다. J2EE 플랫폼은 SOAP 메시지를 어셈블 및 역어셈블하는 데 사용할 수 있는 `java.xml.soap` 패키지를 제공합니다.

▼ 안정적인 SOAP 메시징을 수행하는 방법

1. **`java.xml.soap` 패키지에 정의된 객체를 사용하여 SOAP 메시지를 구성하거나, `javax.xml.messaging` 패키지에 정의된 서블릿을 사용하여 SOAP 메시지를 수신하거나, JAX-RPC와 같은 웹 서비스를 사용하여 SOAP 메시지를 수신합니다.**
2. **Message Transformer 유틸리티를 사용하여 SOAP 메시지를 JMS 메시지로 변환합니다.**
3. **JMS 메시지를 원하는 대상으로 보냅니다.**
4. **JMS 메시지를 비동기식 또는 동기식으로 소비합니다.**
5. **JMS 메시지를 소비한 후 Message Transformer 유틸리티를 사용하여 해당 메시지를 SOAP 메시지로 변환합니다.**
6. **`java.xml.soap` 패키지에 정의된 SAAJ API를 사용하여 SOAP 메시지를 역어셈블합니다.**
   SOAP 메시지 및 처리 방법에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients**의 5 장, "Working with SOAP Messages"를 참조하십시오.

Java 및 C 클라이언트

Message Queue에서는 레거시 C 및 C++ 응용 프로그램을 사용하여 JMS 기반 메시징에 참여할 수 있도록 메시징 서비스에 C API를 제공합니다.

JMS 프로그래밍 모델은 Message Queue C 클라이언트 설계의 기초입니다. **Sun Java System Message Queue 3.7 UR1 Developer's Guide for C Clients**에서는 C 데이터 유형 및 함수가 이 모델을 구현하는 방법에 대해 설명합니다.

Java 인터페이스와 마찬가지로 C 인터페이스도 다음 기능을 지원합니다.

- 게시/가입 및 지점간 연결
- 동기식 및 비동기식 수신
- `CLIENT`, `AUTO` 및 `DUPS_OK` 확인 모드
- 로컬 트랜잭션
- 세션 복구
- 임시 주제 및 대기열
- 메시지 선택기

그러나 Java Message Service 사양은 *Java* 클라이언트에 대해서만 표준으로 사용됩니다. 즉, C Message Queue API는 Message Queue 공급자 전용이므로 다른 JMS 공급자가 사용할 수 없습니다. 다른 JMS 공급자는 C 클라이언트를 포함하는 메시징 응용 프로그램을 처리할 수 없습니다.

C 인터페이스는 다음 기능을 지원하지 않습니다.

- 관리 대상 객체 사용
- 맵, 스트림 또는 객체 메시지 유형
- 소비자 기반 흐름 제어
- 대기열 브라우저
- JMS 응용 프로그램 서버 기능(연결 소비자, 분산 트랜잭션)
- SOAP 메시지 송수신
- 압축된 JMS 메시지 송수신
- 자동 재연결 또는 페일오버 - 연결 실패 시, 클라이언트 런타임에서 브로커에 자동으로 다시 연결
- `NO_ACKNOWLEDGE` 모드

◆ ◆ ◆ 3 장

3 Message Queue 서비스

Message Queue 클라이언트 성능은 클라이언트 설계 및 Message Queue 서비스를 구성하고 관리하는 방법에 따라 다릅니다. 이 장에서는 1장에서 소개한 Message Queue 서비스에 대해 자세히 설명합니다. 서비스 구성 요소를 검토하고, 이러한 구성 요소를 구성하는 데 사용하는 도구를 소개하며, 여러 환경에서 메시지 서비스를 관리하는 데 필요한 작업을 요약합니다. 이 장은 다음 절로 구성되어 있습니다.

- 55 페이지 "구성 요소 서비스"
- 66 페이지 " 관리 도구 및 작업"
- 70 페이지 "메시징 작업 확장"

구성 요소 서비스

그림 3–1은 Message Queue 서비스를 보여 줍니다. 2 장에서는 프로그래밍 모델 그리고 클라이언트가 Java 및 C API를 사용하여 클라이언트 런타임(메시지 서비스 중 클라이언트 응용 프로그램에 액세스할 수 있는 부분)과 상호 작용하는 방법에 대해 설명합니다. 이 장에서는 관리자가 액세스할 수 있는 메시지 서비스의 구성 요소 및 서비스에 대해 중점적으로 설명합니다.

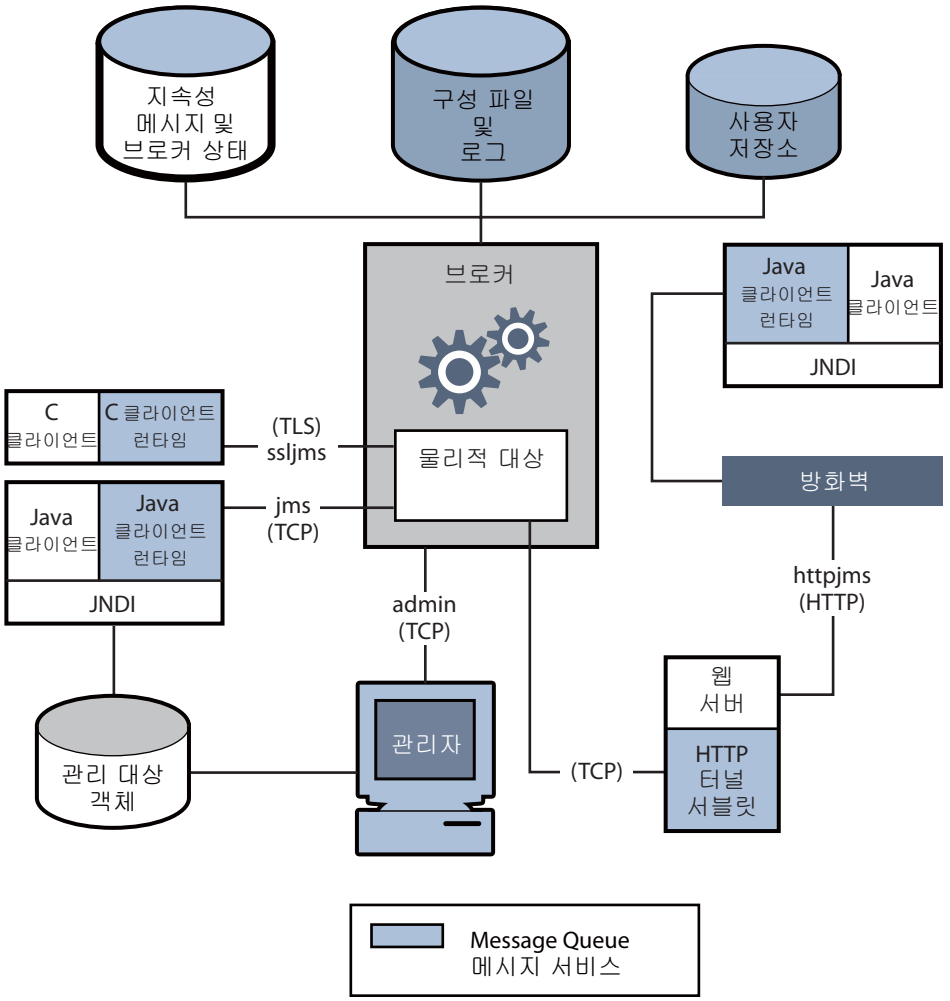

그림 3–1 Message Queue 서비스

브로커 등록 정보를 설정하여 Message Queue 서비스를 제어합니다. 이러한 등록 정보는 특정 등록 정보의 영향을 받는 서비스 또는 브로커 구성 요소에 따라 여러 범주로 나뉩니다. 브로커 서비스에는 다음이 포함됩니다.

- **연결 서비스** - 브로커와 클라이언트 간의 물리적 연결을 관리하면서 보내고 받는 메시지 전송을 담당합니다.
- **라우팅 서비스** - 메시지 서비스에서 사용하는 제어 메시지뿐만 아니라 JMS 메시지를 라우팅하고 전달함으로써 안정적인 전달을 지원합니다.
- **지속성 서비스** - 영구 저장소에 대한 데이터 쓰기 및 영구 저장소로부터의 데이터 검색을 관리합니다.

- **보안 서비스** - 브로커에 연결하는 사용자를 인증하고 사용자 작업에 권한을 부여합니다.
- **모니터링 서비스** - 메트릭 및 진단 정보를 생성하고 이러한 정보를 지정한 출력 채널에 기록합니다.

다음 절에서는 이러한 각 서비스에 대해 설명하고 필요 사항에 맞게 서비스를 사용자 정의하는 데 사용하는 등록 정보를 요약합니다.

브로커 등록 정보가 여러 구성 파일에 정의되며, 브로커 시작 시 사용한 명령줄에서 정의될 수도 있습니다. **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 4 장, "Configuring a Broker"에서는 이 구성 파일에 대해 소개하며, 어떤 파일의 등록 정보 값이 다른 파일에 설정된 값을 대체하기 위해 사용되는 우선 순위도 설명합니다. 시작 명령으로 설정한 등록 정보는 다른 모든 설정에 우선합니다.

연결 서비스

연결 관련 등록 정보를 사용하여 브로커와 해당 브로커의 클라이언트 간의 물리적 연결을 구성하고 관리할 수 있습니다. Message Queue 클라이언트에서 사용할 수 있는 연결 서비스의 이름, 유형, 기본 프로토콜 등에 대해서는 27 페이지 "브로커에 연결"을 참조하십시오. 연결 서비스는 다중 스레드되며, 브로커의 포트 매퍼가 동적으로 할당하거나 관리자가 정적으로 할당할 수 있는 전용 포트를 통해 사용할 수 있습니다. 기본적으로 브로커를 시작하면 `jms` 및 `admin` 서비스가 시작되고 실행됩니다.

모든 연결에는 양측이 있기 때문에 연결 구성은 양측에서 이루어지며 양측에서 함께 조정되어야 합니다.

- 클라이언트는 기본값이 아닌 연결 서비스, 호스트 및 포트를 요청하고, 다른 브로커와의 재연결이 필요한 경우 연결할 브로커 목록을 지정하며, 재연결 동작을 구성하기 위해 연결 팩토리 객체의 특정 속성을 구성해야 합니다. 또한 클라이언트는 실패한 연결을 테스트할 핑 간격을 지정할 수 있습니다.
- 한편 관리자는 브로커 등록 정보를 사용하여 기본값이 아닌 연결 서비스를 활성화하고, 필요한 경우 정적 포트를 할당하며, 스레딩을 구성하고, 여러 네트워크 카드가 사용되는 경우 연결할 호스트를 지정합니다. 또한 관리자는 클라이언트 액세스 가능 여부를 테스트할 핑 간격을 지정할 수 있는데, 이 기능은 자원 관리에 유용합니다.

클라이언트는 방화벽을 통과하여 Message Queue 서비스에 연결할 수 있습니다. 이렇게 하려면 방화벽 관리자에게 특정 포트를 열어 놓도록 요청한 다음 해당 (정적) 포트에 연결하거나 부록 B에 요약된 것처럼 HTTP 또는 HTTPS 서비스를 사용하면 됩니다.

각 연결 서비스는 특정 인증 및 권한 부여 기능도 지원합니다. 자세한 내용은 62 페이지 "보안 서비스"를 참조하십시오.

포트 매퍼

연결 서비스는 브로커의 주 포트인 7676에 있는 일반 **포트 매퍼**를 통해 동적으로 포트를 할당합니다. Message Queue 클라이언트 런타임이 브로커와의 연결을 설정할 때 먼저 포트 매퍼에 접속하여 선택된 연결 서비스의 포트 번호를 요청합니다.

`jms`, `ssljms`, `admin` 및 `ssladmin` 연결 서비스를 구성할 때 **정적** 포트 번호를 할당하여 포트 매퍼를 대체할 수 있습니다. 하지만 정적 포트는 대개 방화벽을 통한 연결 등 특수 상황에서만 사용되므로 일반적으로 권장되지 않습니다.

스레드 풀 관리

각 연결 서비스는 다중 스레드 방식으로서, 다중 연결을 지원합니다. 이러한 연결에 필요한 스레드는 풀에 있는 브로커에서 유지 관리합니다. 할당 방법은 최소 및 최대 스레드 값에 대해 지정한 값 및 선택한 스레딩 모델에 따라 다릅니다.

브로커 등록 정보를 설정하여 스레드의 최소 및 최대 개수를 지정할 수 있습니다. 연결 시 스레드가 필요하면 해당 연결을 지원하는 서비스의 스레드 풀에 스레드가 추가됩니다. 최소 개수는 할당 가능한 스레드 수를 지정합니다. 사용 가능한 스레드 수가 이 최소 임계값을 초과하는 경우 시스템은 최소 임계값에 다시 도달할 때까지 스레드를 종료시켜 여유 스레드를 확보하는 방법으로 메모리 자원을 절약합니다. 로드량이 많은 경우 풀의 최대 수에 도달할 때까지 스레드 수가 증가할 수 있습니다. 이러한 경우 스레드를 사용할 수 있을 때까지 새 연결이 거부됩니다.

선택한 스레딩 모델에서는 스레드가 단일 연결 전용인지 또는 여러 연결에서 공유할 것인지를 지정합니다.

- 전용 모델에서는 브로커에 대한 각 연결에서 받는 메시지용과 보내는 메시지용의 스레드 두 개가 필요합니다. 그러면 가능한 연결 수가 제한되지만, 우수한 성능이 제공됩니다.
- 공유 모델 연결은 메시지를 보내거나 받을 때 공유 스레드에서 처리됩니다. 각 연결에는 전용 스레드가 없기 때문에 이 모델에서는 가능한 연결 수가 늘어나지만, 스레드 관리에 일부 오버헤드가 추가되어 성능에 영향을 줍니다.

대상 및 라우팅 서비스

클라이언트가 브로커에 연결된 후에는 메시지 라우팅 및 전달을 계속할 수 있습니다. 이 단계에서 브로커는 여러 종류의 물리적 대상을 작성 및 관리하여 메시지의 원활한 흐름을 보장하고 자원을 효율적으로 관리합니다. 브로커에서 라우팅 및 대상과 관련된 브로커 등록 정보를 사용하여 사용자의 응용 프로그램 요구 사항에 적합한 방식으로 이러한 작업을 관리합니다.

브로커의 **물리적 대상**은 메시지가 메시지 소비자에게 전달되기 전에 저장되는 메모리상의 위치입니다. 물리적 대상에는 다음 네 가지 종류가 있습니다.

- **관리 작성 대상**은 관리자가 GUI(imqadmin) 또는 imqcmd 유틸리티를 사용하여 작성합니다. 이 대상은 프로그래밍 방식으로 작성된 논리 대상이나 관리자가 작성하고 클라이언트가 조회하는 대상 관리 대상 객체에 해당합니다. imqcmd 유틸리티를 사용하여 각 관리 작성 대상의 등록 정보를 설정하고 업데이트할 수 있습니다.
- **자동 작성 대상**은 메시지 소비자나 제작자가 존재하지 않는 대상에 액세스를 시도할 때마다 브로커에서 자동으로 작성됩니다. 일반적으로 개발 과정에서 사용됩니다. 이러한 대상의 작성을 허용하지 않도록 브로커 등록 정보를 설정할 수 있습니다. 브로커 등록 정보를 설정하여 특정 브로커의 모든 자동 작성 대상을 구성할 수 있습니다.

  자동 작성 대상은 사용자 클라이언트가 없거나 메시지를 더 이상 포함하지 않는 경우처럼 더 이상 사용되지 않을 때 브로커에 의해 자동으로 삭제됩니다. 브로커를 다시 시작할 경우 브로커는 지속성 메시지를 포함하는 경우에만 이 종류의 대상을 다시 작성합니다.
- 임시 대상은 메시지의 회신을 받을 대상이 필요한 클라이언트가 프로그래밍 방식을 통해 명시적으로 작성하고 삭제합니다. 이 이름이 의미하는 것처럼 이러한 대상은 임시 대상이며, 그 대상이 필요한 연결이 지속되는 동안에 한해 브로커가 유지 관리합니다.

  임시 대상은 영구적으로 저장되지 않으며 브로커를 다시 시작해도 다시 작성되지 않지만, 관리 도구에는 표시됩니다.
- **사용 불능 메시지 대기열**은 브로커 시작 시 자동으로 작성되고, 진단 목적으로 사용 불능 메시지를 저장할 때 사용되는 특수한 대상입니다. imqcmd 유틸리티를 사용하여 사용 불능 메시지 대기열의 등록 정보를 설정할 수 있습니다.

대상 관리

대상을 관리할 때 imqcmd 유틸리티를 사용합니다. 대상 관리는 다음 여러 작업을 포함합니다.

- 대상 작성, 일시 중지, 다시 시작 또는 삭제
- 브로커의 모든 대상 나열
- 대상의 상태 및 등록 정보에 대한 정보 표시
- 대상의 메트릭 정보 표시
- 대상의 메시지를 지속적으로 처리하는 데 사용되는 디스크 공간 압축
- 물리적 대상의 등록 정보 업데이트

관리 작업은 관리 작성, 자동 작성, 임시 또는 사용 불능 메시지 대기열과 같이 관리되는 대상의 종류에 따라 달라집니다. 예를 들어 임시 대상은 명시적으로 삭제될 필요가 없습니다. 자동 작성된 등록 정보는 해당 브로커의 모든 자동 작성 대상에 적용되는 브로커 구성 등록 정보를 사용하여 구성됩니다.

물리적 대상 구성

최적의 성능을 얻기 위해 물리적 대상을 작성하거나 업데이트할 때 등록 정보를 설정할 수 있습니다. 설정할 수 있는 등록 정보는 다음을 포함합니다.

- 대상의 유형과 이름
- 대상의 개별 및 종합 제한(최대 메시지 수, 최대 전체 바이트 수, 메시지별 최대 바이트 수, 최대 제작자 수)
- 개별 및 종합 제한 초과 시 브로커에서 수행해야 하는 작업
- 일괄적으로 전달되는 최대 메시지 수
- 대상의 사용 불능 메시지를 사용 불능 메시지 대기열로 전송할지 여부
- 대상을 클러스터의 다른 브로커로 복제할지 여부(클러스터된 브로커의 경우)

대기열 대상의 경우 최대 백업 소비자 수를 구성할 수도 있으며, 로컬 대기열로 전달이 권장되는지 여부를 지정(클러스터된 브로커의 경우)할 수 있습니다.

사용 불능 메시지 대기열의 제한 및 동작을 구성할 수도 있습니다. 그러나 이러한 대기열의 기본 등록 정보는 표준 대기열의 등록 정보와 다릅니다.

메모리 관리

대상은 처리하는 메시지의 수 및 크기, 등록하는 소비자의 수 및 지속성에 따라 많은 양의 자원을 소비할 수 있으므로 메시지 서비스의 성능 및 안정성을 보장하도록 면밀하게 관리해야 합니다.

등록 정보를 설정하여 브로커가 받는 메시지로 넘치는 것을 방지하고 브로커의 메모리 부족을 방지할 수 있습니다. 자원이 부족하게 될 때 브로커는 세 가지 수준의 메모리 보호, 즉 대상 제한, 시스템 전체 제한 및 시스템 메모리 임계값을 사용하여 메시지 서비스를 계속 작동시킵니다. 원칙적으로, 대상 제한 및 시스템 전체 제한이 적절히 설정되면 중요 시스템 메모리 임계값을 초과하지 않습니다.

대상 메시지 제한

대상 속성을 사용하여 각 대상의 메모리 및 메시지 흐름을 관리할 수 있습니다. 예를 들어 대상에 허용되는 최대 제작자 수, 최대 메시지 수(또는 크기) 및 단일 메시지의 최대 크기를 지정할 수 있습니다.

또한 이러한 제한에 도달하면 제작자 속도 줄이기, 가장 오래된 메시지 삭제, 가장 낮은 우선 순위의 메시지 삭제 또는 최신 메시지 거부와 같이 브로커의 응답 방법을 지정할 수도 있습니다.

시스템 전체 메시지 제한

등록 정보를 사용하여 브로커의 모든 대상에 적용할 제한을 설정할 수 있습니다. 전체 메시지 수 및 모든 메시지에 소비되는 메모리를 지정할 수 있습니다. 시스템 전체 메시지 제한 중 하나에 도달하면 브로커는 새 메시지를 거부합니다.

시스템 메모리 임계값

마지막으로 등록 정보를 사용하여 메모리 과부하 방지를 위한 브로커 조치의 수위를 높이는 임계값을 설정할 수 있습니다. 취할 수 있는 조치는 다음과 같이 메모리 자원 상태에 따라 달라집니다. **초록**(사용 가능한 메모리 충분), **노랑**(브로커 메모리 감소 중), **주황**(브로커 메모리 부족), **빨강**(브로커가 사용할 수 있는 메모리 없음). 브로커의 메모리 상태가 **초록**에서 **빨강**으로 진행될수록 브로커는 점차 높은 수준의 조치를 취합니다.

- 데이터 저장소에서 지속성 메시지의 메모리 내 복사본을 삭제합니다.
- 비지속성 메시지의 제작자를 억제한 뒤 결국 브로커로 향하는 메시지 흐름을 중지시킵니다. 지속성 메시지 흐름은 브로커가 각 메시지를 확인해야 하기 때문에 자동으로 제한됩니다.

지속성 서비스

오류 발생 시 브로커를 복구하려면 메시지 전달 작업 상태를 다시 작성해야 합니다. 이렇게 하려면 데이터 저장소에 상태 정보를 저장해야 합니다. 브로커는 다시 시작할 때 저장된 데이터를 사용하여 대상 및 영구 가입을 다시 작성하고, 지속성 메시지를 복구하며, 열린 트랜잭션을 롤백하고, 전달되지 않은 메시지의 라우팅 테이블을 다시 작성합니다. 그런 다음 메시지 전달을 다시 시작합니다.

Message Queue 서비스는 파일 기반 및 JDBC 호환 지속성 모듈(그림 3–2 참조)을 지원하며 기본적으로 파일 기반 지속성을 사용합니다.

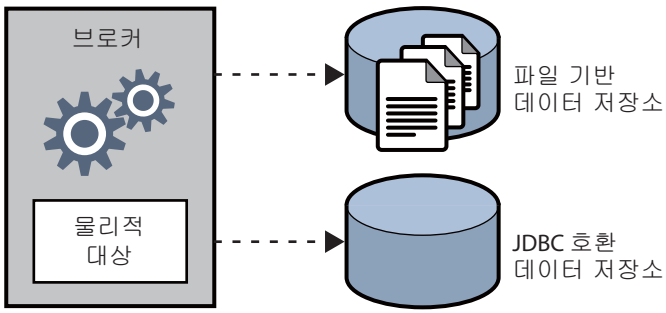

그림 3–2 지속성 지원

파일 기반 지속성

파일 기반 지속성은 개별 파일을 사용하여 영구 데이터를 저장하는 메커니즘입니다. 파일 기반 지속성을 사용하면 브로커 등록 정보를 설정하여 다음을 수행할 수 있습니다.

- 메시지 추가 및 제거 시 단편화를 해소하기 위한 데이터 저장소 압축
- **기록할 때마다** 메모리 상태에서 물리적 저장 장치와 동기화. 이렇게 하면 시스템 충돌로 인한 데이터 손실을 줄일 수 있습니다.
- 데이터 저장소 파일에 대한 메시지 할당 관리 및 파일 관리와 저장에 필요한 자원 관리

일반적으로 파일 기반 지속성은 JDBC 기반 지속성보다 빠르지만, JDBC 호환 저장소가 제공하는 중복 및 관리 제어 기능을 선호하는 사용자도 있습니다.

JDBC 기반 지속성

JDBC 기반 지속성에서는 Java Database Connectivity(JDBC™) 인터페이스를 사용하여 브로커를 JDBC 호환 데이터 저장소에 연결합니다. 브로커가 JDBC 드라이버를 통해 데이터 저장소에 액세스하게 하려면 다음을 수행해야 합니다.

- JDBC 관련 브로커 구성 등록 정보 설정. 사용되는 JDBC 드라이버를 지정하고, 브로커를 JDBC 사용자로 인증하며, 필요한 테이블을 만드는 경우 등에 이 등록 정보를 사용할 수 있습니다.
- `imqdbmgr` 유틸리티를 사용하여 적합한 체제를 갖는 데이터 저장소 작성

이러한 작업을 완료하는 모든 절차와 관련 구성 등록 정보에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 4 장, “Configuring a Broker”을 참조하십시오.

보안 서비스

Message Queue 서비스는 각 브로커 인스턴스에 대한 인증 및 권한 부여(액세스 제어)를 지원하고 암호화 기능도 지원합니다.

- **인증**은 검증된 사용자만 브로커에 연결할 수 있게 합니다.
- **권한 부여**는 자원에 액세스하고 특정 작업을 수행할 권한을 갖는 사용자나 그룹을 지정합니다.
- **암호화**는 연결을 통한 전달 중 메시지가 훼손되지 않게 보호합니다.

인증 및 권한 부여는 사용자 이름, 비밀번호 및 **그룹** 멤버쉽 등 메시징 시스템의 사용자에 대한 정보가 있는 저장소에 따라 다릅니다. 또한 사용자 또는 그룹의 특정 작업에 대한 권한을 부여하기 위해 브로커는 사용자 또는 그룹이 수행할 수 있는 작업을 지정하는 **액세스 제어 등록 정보 파일**을 확인해야 합니다. 브로커에서 사용자를 인증하고 작업 권한을 부여할 때 필요한 정보를 설정해야 합니다.

그림 3–3에서는 브로커에서 인증 및 권한 부여를 수행할 때 필요한 구성 요소를 보여 줍니다.

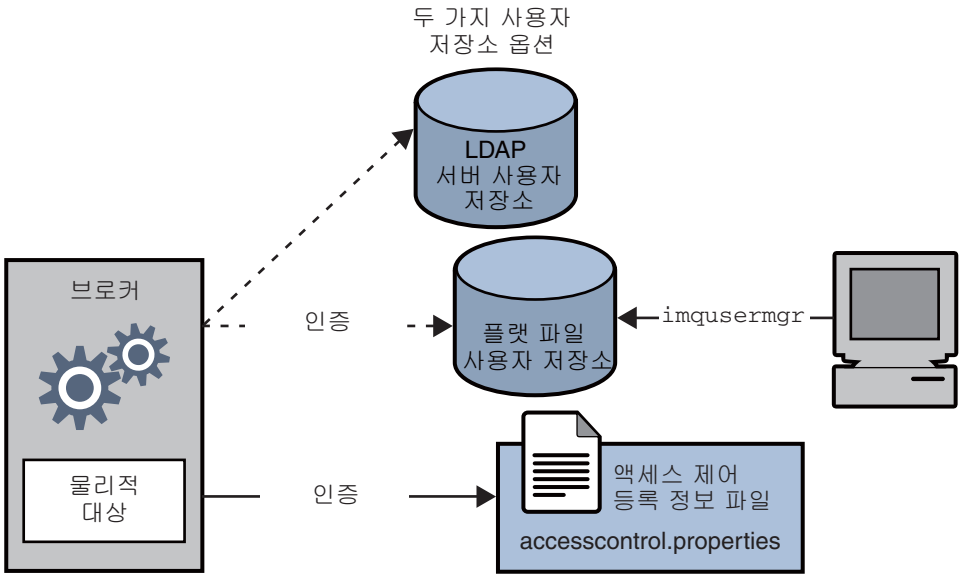

그림 3–3 보안 관리자 지원

그림 3–3과 같이, Message Queue 서비스와 함께 제공된 플랫 파일 사용자 저장소에 사용자 데이터를 저장할 수 있습니다. 또는 기존의 LDAP 저장소에 플러그인할 수 있습니다. 브로커 등록 정보를 설정하여 선택 사항을 나타냅니다.

- 플랫 파일 저장소를 선택할 경우 저장소 관리에 `imqusermgr` 유틸리티를 사용해야 합니다. 이 옵션은 사용하기 쉬우며 기본적으로 제공됩니다.
- 기존 LDAP 서버를 사용하려면 LDAP 공급업체에서 제공하는 도구를 사용하여 사용자 저장소를 채우고 관리합니다. 또한 브로커가 LDAP 서버에서 사용자 및 그룹 정보를 쿼리할 수 있도록 브로커 인스턴스 구성 파일의 등록 정보를 설정해야 합니다.

  확장성이 중요하거나 여러 브로커에서 공유할 저장소가 필요한 경우 LDAP 옵션이 편리합니다. 브로커 클러스터를 사용하는 경우에도 마찬가지입니다.

인증 및 권한 부여

클라이언트가 연결을 요청할 경우 이 클라이언트는 사용자 이름 및 비밀번호를 제공해야 합니다. 브로커는 지정된 이름과 비밀번호를 사용자 저장소에 저장된 이름 및 비밀번호와 비교합니다. 클라이언트에서 브로커로 비밀번호가 전송될 때 이 비밀번호는 기본 64 인코딩이나 메시지 다이제스트(MD5) 해싱을 사용하여 인코딩됩니다. MD5는 플랫 파일 저장소에 사용되며, 기본 64는 LDAP 저장소에 필요합니다. LDAP를 사용하는 경우 보안 TLS 프로토콜을 사용할 수 있습니다. 각 연결 서비스가 개별적으로 사용하는 인코딩 유형을 구성하거나 브로커 전체에 대한 인코딩을 설정하기 위해 브로커 등록 정보를 설정할 수 있습니다.

사용자가 어떤 작업을 수행하려고 하면, 브로커는 사용자의 이름 및 그룹 멤버쉽(사용자 저장소에 있음)을 해당 작업 액세스를 위해 지정된 사용자의 이름과 그룹 멤버쉽(액세스 제어 등록 정보 파일에 있음)과 대조 확인합니다. 액세스 제어 등록 정보 파일은 다음 작업에 대한 사용자 또는 그룹의 권한을 지정합니다.

- 브로커에 연결
- 대상 액세스: 소비자, 제작자 또는 특정 대상이나 모든 대상에 대한 대기열 브라우저 작성
- 대상 자동 작성

브로커 등록 정보를 설정하여 다음 정보를 지정할 수 있습니다.

- 액세스 제어의 사용 여부
- 액세스 제어 파일의 이름
- 비밀번호 인코딩 방법
- 클라이언트가 브로커의 인증 요청에 응답하는 데 걸리는 시간
- 보안 연결에 필요한 정보

암호화

클라이언트와 브로커 사이에 전송되는 메시지를 암호화하려면 SSL(Secure Socket Layer) 표준 기반의 연결 서비스를 사용해야 합니다. SSL은 SSL 사용 가능 브로커와 SSL 사용 가능 클라이언트 사이에 암호화된 연결을 설정함으로써 연결 수준에서의 보안을 제공합니다.

브로커 등록 정보를 설정하여 사용되는 SSL 키 저장소의 보안 등록 정보와 비밀번호 파일의 이름 및 위치를 지정할 수 있습니다.

모니터링 서비스

브로커는 응용 프로그램 및 브로커 성능을 모니터링하고 진단할 구성 요소를 포함합니다. 여기에는 다음 항목이 포함됩니다.

- 데이터를 생성하는 구성 요소, 메트릭 생성자 및 이벤트를 기록하는 브로커 코드
- 여러 출력 채널을 통해 정보를 기록하는 로거 구성 요소
- 메트릭 정보를 포함하는 JMS 메시지를 JMS 모니터링 클라이언트가 소비할 수 있도록 주제 대상에게 보내는 메시지 제작자

그림 3–4에서는 일반적인 체계를 보여 줍니다.

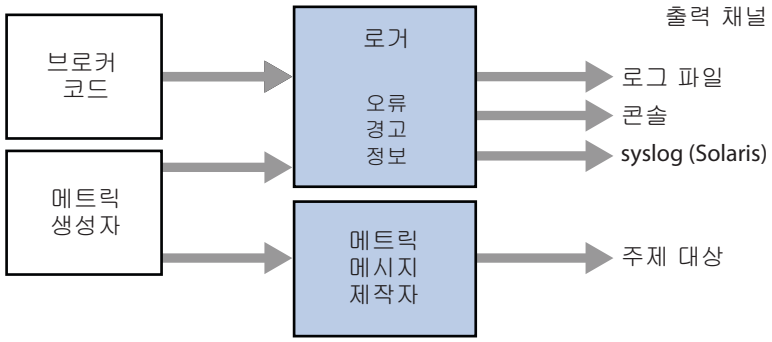

그림 3–4 모니터링 서비스 지원

메트릭 생성자

메트릭 생성자는 브로커 내부 및 외부의 메시지 흐름, 브로커 메모리의 메시지 수 및 이 메시지가 소비하는 메모리, 열려 있는 연결 수, 사용 중인 스레드 수 등과 같은 브로커 활동 정보를 제공합니다.

브로커 등록 정보를 설정하여 메트릭 데이터 생성을 설정 또는 해제하고 메트릭 보고서 생성 빈도를 지정할 수 있습니다.

로거

Message Queue 로거는 브로커 코드 및 메트릭 생성자가 생성한 정보를 가져온 다음, 오류 발생 시 표준 출력(콘솔), 로그 파일 및 `syslog` 데몬 프로세스(Solaris™의 경우)에 정보를 기록합니다.

로거에서 수집된 정보의 유형과 각 출력 채널에 기록된 유형을 지정하도록 브로커 등록 정보를 설정할 수 있습니다. 로그 파일의 경우 로그 파일을 닫고 출력을 새 파일로 롤오버하는 지점을 지정할 수도 있습니다. 로그 파일이 지정된 크기나 표시 시간에 도달하면 이 파일을 저장하고 새 로그 파일을 작성합니다.

로거를 구성하는 방법 및 로거를 사용하여 성능 정보를 얻는 방법에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 "Configuring and Using Broker Logging"을 참조하십시오.

메트릭 메시지 제작자

그림 3–4에 나타난 메트릭 메시지 제작자는 일정 간격으로 메트릭 생성자로부터 정보를 받아서 메시지에 기록한 다음, 메시지에 포함된 메트릭 정보 유형에 따라 여러 메트릭 주제 대상 중 하나로 보냅니다.

이러한 메트릭 주제 대상에 가입한 Message Queue 클라이언트는 메시지를 소비하고 메시지에 포함된 메트릭 데이터를 처리할 수 있습니다. 이렇게 하면 개발자는 사용자 정의 모니터링 도구를 작성하여 메시징 응용 프로그램을 지원할 수 있습니다. 각 메트릭

메시지 유형에서 보고하는 메트릭 수량에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 18 장, “Metrics Reference”를 참조하십시오. 메트릭 메시지 생성을 구성하는 방법에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for Java Clients**의 4 장, “Using the Metrics Monitoring API” 및 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Writing an Application to Monitor Brokers”을 참조하십시오.

관리 도구 및 작업

이 절에서는 Message Queue 서비스를 구성하는 데 사용하는 도구 및 개발이나 프로덕션 환경에 대한 지원을 완료하는 데 필요한 작업에 대해 설명합니다.

관리 도구

그림 3–5에서는 클라이언트 연결을 제외한 메시지 서비스를 보여 주며, 브로커 구성 요소와 이 구성 요소를 관리하는 데 사용되는 도구에 중점을 두었습니다.

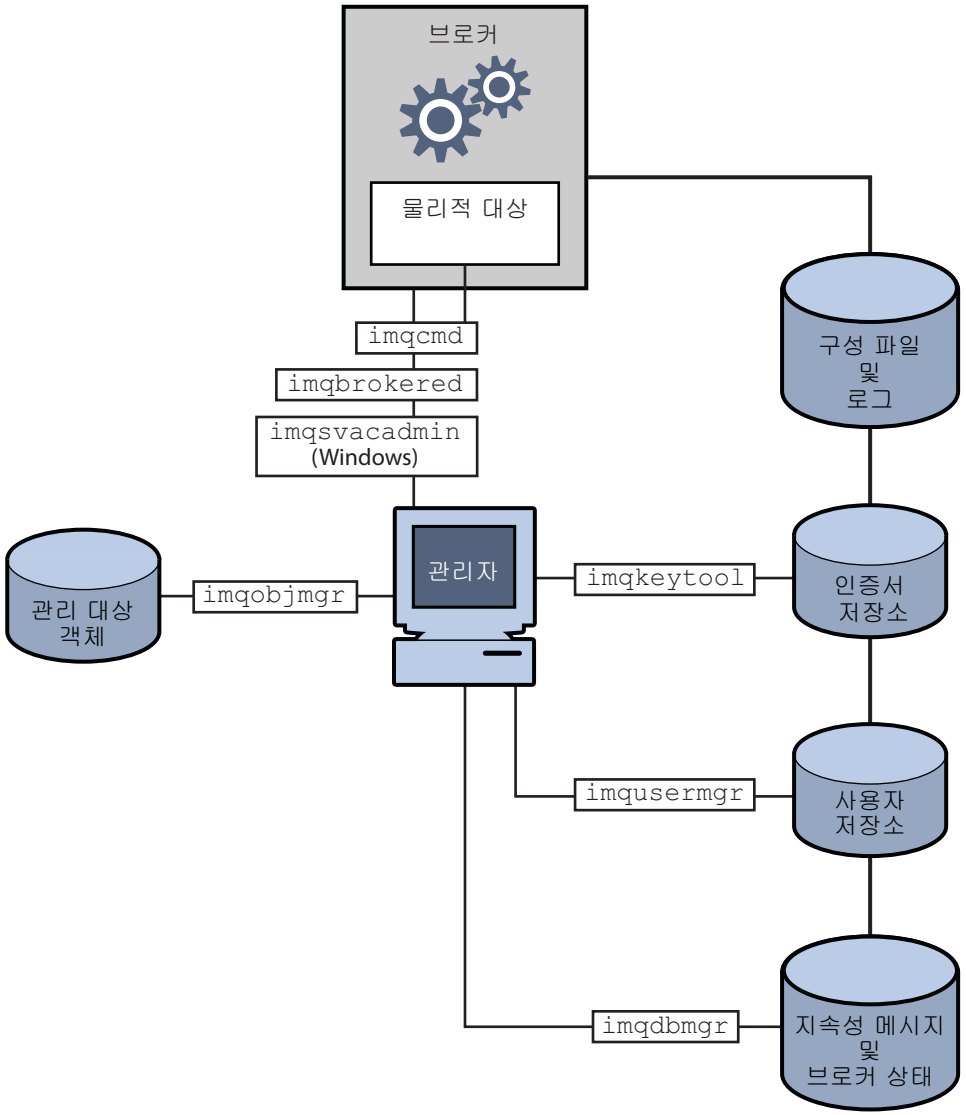

그림 3–5 관리 도구

다음 명령줄 도구를 사용하여 Message Queue 서비스를 구성하고 관리할 수 있습니다.

- 브로커를 시작하려면 `imqbrokerd` 유틸리티를 사용합니다. 클러스터에서 브로커의 연결 여부 및 추가적인 시작 구성 정보를 지정하기 위해 `imqbrokerd` 명령에 옵션을 사용할 수 있습니다.
- 브로커를 시작했으면 `imqcmd` 유틸리티를 사용하여 물리적 대상을 작성, 업데이트 및 삭제하고 브로커와 그 연결 서비스를 제어하며 브로커의 자원을 관리합니다.

- JNDI 객체 저장소에서 관리 대상 객체를 추가, 나열, 업데이트 및 삭제하려면 `imqobjmgr` 유틸리티를 사용합니다.
- 사용자 인증 및 권한 부여를 위해 파일 기반 사용자 저장소를 채우려면 `imqusermgr` 유틸리티를 사용합니다.
- 영구 저장소에 사용하는 JDBC 호환 데이터베이스를 작성하고 관리하려면 `imqdbmgr` 유틸리티를 사용합니다. (기본 제공된 파일 저장소에서는 외부 관리가 필요하지 않습니다.)
- SSL 인증에 사용되는, 자체 서명된 인증서를 생성하려면 `imqkeytool` 유틸리티를 사용합니다.
- 브로커를 Windows 서비스로 설치, 쿼리하고 제거하려면 `imqsvcadmin` 유틸리티를 사용합니다.

GUI 기반 관리 콘솔은 `imqcmd`와 `imqobjmgr` 유틸리티의 일부 기능을 결합한 것입니다. 이러한 결합 기능을 사용하여 다음을 수행할 수 있습니다.

- 브로커에 연결하여 관리
- 물리적 대상 작성 및 관리
- 객체 저장소에 연결, 저장소에 객체 추가 및 객체 관리

개발 환경 지원

클라이언트 구성 요소를 개발할 때 관리 작업을 최소화시키는 것이 가장 좋습니다. 이러한 관리 작업을 최소화할 수 있도록 설계된 Message Queue 제품은 별도의 조정 없이 사용할 수 있으므로 무리 없이 브로커를 시작할 수 있습니다. 개발에 중점을 둔 경우에는 다음과 같은 방법을 사용합니다.

- 데이터 저장소(기본 제공된 파일 지속성), 사용자 저장소(파일 기반) 및 액세스 제어 등록 정보 파일로 구성된 기본 구현을 사용합니다. 이는 개발 테스트에 적합합니다. 기본 사용자 저장소는 설치 후 즉시 브로커를 사용할 수 있도록 기본 항목으로 작성됩니다. 기본 사용자 이름(`guest`) 및 비밀번호(`guest`)를 사용하여 클라이언트를 인증할 수 있습니다.
- 단순 파일 시스템 객체 저장소용으로 디렉토리를 작성하여 사용하고, 여기에 관리 대상 객체를 저장합니다. 저장소 작성을 선호하지 않는 경우에는 관리 대상 객체를 직접 코드로 인스턴스화할 수도 있습니다.
- 브로커에 물리적 대상을 명시적으로 작성하지 않고 자동으로 작성된 물리적 대상을 사용합니다. 자세한 내용은 해당 개발 안내서를 참조하십시오.

프로덕션 환경 지원

프로덕션 환경에서 메시지 서비스 관리는 응용 프로그램 성능 그리고 엔터프라이즈 확장, 가용성 및 보안 요구 사항 해결 측면에서 중요한 역할을 합니다. 이러한 환경에서는 관리자가 더 많은 작업을 수행해야 합니다. 이러한 작업은 대략 설정 및 유지 관리 작업으로 나뉩니다.

설정 작업

일반적으로 다음 설정 작업을 수행해야 합니다.

- 관리 액세스 보안

  파일 기반 또는 LDAP 사용자 저장소 사용 여부에 따라 관리자는 `admin` 그룹에 속해야 하고 보안 비밀번호를 가지고 있어야 합니다. 필요한 경우 관리자용 브로커로의 보안 연결을 작성합니다.

- 클라이언트 액세스 보안

  파일 기반 또는 LDAP 사용자 저장소 사용 여부에 따라 메시지 서비스에 액세스할 수 있는 사용자 이름을 사용자 저장소에 입력하고 액세스 제어 등록 정보 파일을 편집하여 사용자에게 적절한 권한을 부여합니다. 필요한 경우 SSL 기반 연결 서비스를 설정합니다. 인증되지 않은 연결을 방지하려면 "guest" 사용자 비밀번호를 변경해야 합니다.

- 물리적 대상 작성 및 구성

  브로커 자원에서 메시지 수와 메시지에 할당된 메모리 양을 지원할 수 있도록 대상 속성을 설정합니다.

- 관리 대상 객체 작성 및 구성

  LDAP 객체 저장소를 사용하려는 경우 저장소를 구성하고 설정해야 합니다. 연결 팩토리 및 대상 관리 대상 객체를 작성하고 구성합니다.

- 상태 있는 수평 확장이 필요한 경우 브로커 클러스터 작성

  중앙 구성 파일을 작성하고 마스터 브로커를 지정합니다.

유지 관리 작업

브로커 자원을 모니터링 및 제어하고 응용 프로그램 성능을 조정하려면 응용 프로그램을 배포한 후 다음을 수행해야 합니다.

- 응용 프로그램 클라이언트 지원 및 관리
  - 대상, 영구 가입과 트랜잭션의 모니터링 및 관리
  - 자동 작성 기능 비활성화
  - 사용 불능 메시지 대기열 모니터링 및 관리
- 브로커 모니터링 및 조정
  - 실패한 브로커 복구
  - 브로커 모니터링, 조정 및 재구성

- 브로커 메모리 자원 관리
- 필요한 경우 클러스터 확장

- 관리 대상 객체 관리

  필요에 따라 추가 관리 대상 객체를 작성하고, 연결 팩토리 속성을 조정하여 성능 및 처리 능력을 향상시킵니다.

메시징 작업 확장

Message Queue 서비스는 브로커를 연결하고 브로커의 상태 정보 공유를 허용함으로써 수평으로 확장될 수 있습니다. 이렇게 하면 어떤 단일 브로커가 원격 대상에 액세스하여 더 많은 클라이언트를 서비스할 수 있습니다. 자세한 내용은 4 장을 참조하십시오.

◆ ◆ ◆ 4 장

4 브로커 클러스터

Message Queue는 **브로커 클러스터**, 즉 함께 작업하면서 클라이언트에 메시지 전달 서비스를 제공하는 브로커의 그룹을 사용하도록 지원합니다. 클러스터를 사용하여 관리자는 다중 브로커 간에 클라이언트 연결을 분산하는 방식으로 메시지 트래픽의 볼륨에 대한 메시징 작업 크기를 조절할 수 있습니다.

이 장에서는 그러한 브로커 클러스터의 구조와 내부 기능에 대해 설명합니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 71 페이지 "클러스터 구조"
- 72 페이지 "메시지 전달"
- 78 페이지 "클러스터 구성"
- 79 페이지 "클러스터 동기화"

브로커 클러스터는 데이터 가용성이 아닌 서비스 가용성을 제공합니다. 클러스터에 있는 브로커에서 장애가 발생하면 해당 브로커에 연결된 클라이언트는 클러스터에 있는 다른 브로커에 다시 연결할 수 있지만, 대체 브로커에 다시 연결될 때 일부 데이터가 손실될 수 있습니다.

클러스터 구조

그림 4–1에서는 Message Queue의 브로커 클러스터 구조를 보여 줍니다. 클러스터의 각 브로커는 나머지 모든 브로커와 직접 연결되어 있습니다. 각 클라이언트(메시지 제작자 또는 소비자)에는 직접 통신하는 단일 **홈 브로커**가 있어 클러스터에 해당 브로커만 있는 것처럼 메시지를 송수신합니다. 배후에서 홈 브로커는 연결된 모든 클라이언트에 전달 서비스를 제공하기 위해 다른 브로커와 함께 작업합니다.

클러스터에서 서비스 가용성은 브로커가 대상 및 영구 가입자에 대한 정보를 공유할 수 있는지 여부에 따라 달라집니다. 클러스터된 브로커에서 장애가 발생하면 이러한 상태 정보는 동기화되지 않을 수 있습니다. 클러스터 내에서 하나의 브로커를 **마스터 브로커**로 지정하면 이러한 가능성을 방지할 수 있습니다. 마스터 브로커는 클러스터의

지속성 항목(대상 및 영구 가입)에 대한 변경 사항을 추적하는 **구성 변경 기록**을 유지 관리합니다. 이 기록은 변경 사항이 생길 때 오프라인 상태였던 브로커에게 이러한 변경 정보를 전파하는 데 사용됩니다.

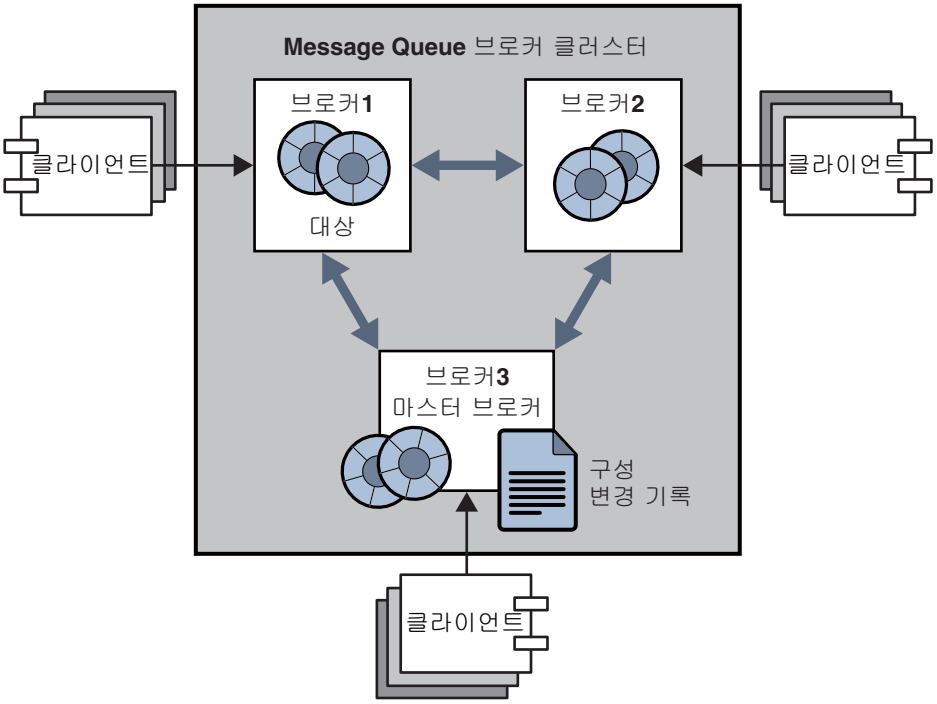

그림 4–1 클러스터 구조

다음 절에서는 하나 이상의 브로커가 오프라인이었던 경우라도 클러스터 내에서 메시지 전달이 이루어지는 방식 및 브로커의 구성 및 동기화 방식에 대해 설명합니다.

메시지 전달

클러스터 구성에서 브로커는 대상 및 메시지 소비자에 대한 정보를 공유합니다. 각 브로커는 다음 정보에 대해서도 알고 있습니다.

- 클러스터에 있는 모든 물리적 대상의 이름, 유형 및 속성
- 각 메시지 소비자의 이름, 위치 및 관심
- 위의 정보에 대한 업데이트(삭제, 추가 또는 재구성)

이러한 정보를 가지고 각 브로커는 자신과 직접 연결된 메시지 제작자로부터 원격 메시지 소비자에게 메시지를 라우팅할 수 있습니다. 제작자의 홈 브로커는 소비자의 홈 브로커와는 다른 기능을 합니다.

- 제작자의 홈 브로커는 해당 제작자가 제작한 메시지의 지속적인 처리 및 라우팅, 로깅, 트랜잭션 관리와 소비자 클라이언트로부터의 확인 처리를 담당합니다.
- 소비자의 홈 브로커는 소비자 정보의 지속적인 처리를 담당하고 소비자에게 메시지를 전달할 뿐 아니라 여전히 소비자가 사용 가능한지 그리고 메시지가 성공적으로 소비되었는지 여부를 제작자의 브로커에게 알립니다.

클러스터된 브로커는 클러스터 내의 메시지 트래픽을 최소화하기 위해 함께 작동합니다. 예를 들어 원격 브로커에 같은 주제 대상에 대해 두 개의 동일한 가입이 있으면 메시지는 회선을 통해 한 번만 전송됩니다. 로컬 소비자에 대한 전달이 원격 소비자에 대한 전달보다 높은 우선 순위를 갖도록 대상 등록 정보를 설정하여 트래픽을 더 줄일 수 있습니다.

클라이언트와 브로커 간에 암호화된 보안 메시지를 전달하는 경우 브로커 간의 메시지 전달을 보호하도록 클러스터를 구성할 수 있습니다.

대상 속성

클러스터된 브로커에 있는 물리적 대상에 대해 설정된 속성은 클러스터에 있는 해당 대상의 모든 인스턴스에 적용됩니다. 그러나 이러한 속성에서 지정한 제한 중 일부는 클러스터 전체에 적용되고, 나머지는 개별 대상 인스턴스에 적용됩니다. 표 4–1에서는 물리적 대상에 대해 설정 가능한 속성과 그 범위를 보여 줍니다.

표 4–1 클러스터된 브로커에 있는 물리적 대상의 등록 정보

등록 정보 이름	범위
maxNumMsgs	각 브로커즉, 클러스터 전체에 걸쳐 제작자를 배포하면 소비하지 않은 전체 메시지에 대한 제한이 증가하게 됩니다.
maxTotalMsgBytes	각 브로커즉, 클러스터 전체에 제작자를 배포하면 소비하지 않은 메시지용으로 예약된 전체 메모리에 대한 제한이 증가하게 됩니다.
lmitBehavior	전역
maxBytesPerMsg	각 브로커
maxNumProducers	각 브로커
maxNumActiveConsumers	전역
maxNumBackupConsumers	전역
consumerFlowLimit	전역
localDeliveryPreferred	전역

표 4–1 클러스터된 브로커에 있는 물리적 대상의 등록 정보 (계속)

등록 정보 이름	범위
isLocalOnly	전역
useDMQ	각 브로커

클러스터링 및 대상

대상이 관리자에 의해 작성되는지, 자동으로 작성되는지 아니면 임시 대상인지 여부에 따라 클러스터 내에서 대상이 전파되는 방식 그리고 연결 또는 브로커 오류 발생 시 대상이 처리되는 방식이 달라집니다. 다음 하위 절에서는 대상이 작성되는 시점과 그 복제 방법을 알 수 있는 몇 가지 사용 예에 대해 알아봅니다. 여기에는 다음이 포함됩니다.

- 74 페이지 "회신 모델을 사용하여 대기열로 생성"
- 75 페이지 "자동 작성 대상에 생성"
- 76 페이지 "주제 대상에 생성"
- 77 페이지 "연결 또는 브로커 오류 발생 시 대상 처리"

회신 모델을 사용하여 대기열로 생성

아래 그림에서는 클라이언트가 대기열에 생성하고 회신 모델을 사용할 경우 대상이 어떻게 작성되고 복제되는지 보여 줍니다.

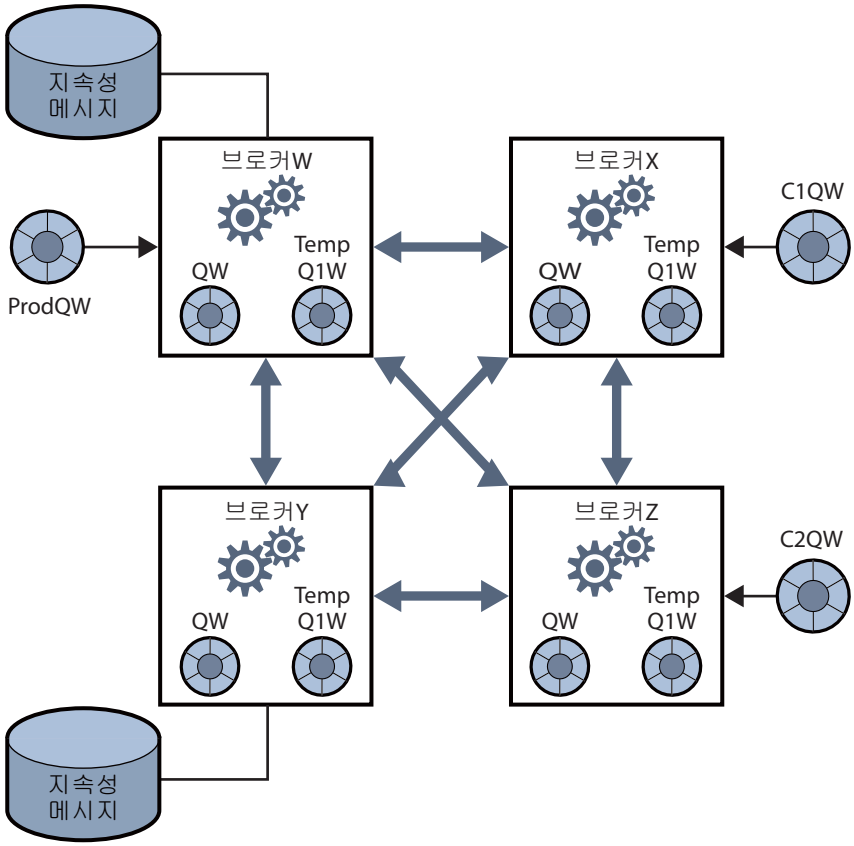

그림 4–2 클러스터에서의 대상 복제: 회신을 사용하는 대기열

1. 관리자는 물리적 대상인 `QW`를 만듭니다. 작성 시점에 클러스터 전체에 걸쳐 대기열이 복제됩니다.
2. 제작자 `ProdQW`는 대기열 `QW`로 메시지를 보내고 임시 대기열 `TempQ1W`로 회신을 보내는 회신 모델을 사용합니다. (임시 대기열은 응용 프로그램에서 임시 대상을 만들고 소비자를 추가할 때 만들어지고 복제됩니다.)
3. 홈 브로커 `BrokerW`는 `QW`에 전송된 메시지를 지속적으로 처리하고 이 메시지의 선택 기준을 충족하는 첫 번째 활성 소비자에게로 메시지 경로를 지정합니다. 메시지를 받을 준비가 된 소비자에 따라 메시지는 `BrokerX`에 있는 소비자 `C1QW`나 `BrokerY`에 있는 소비자 `C2QW`에게 전달됩니다. 메시지를 받은 소비자는 대상 `TempQ1W`로 회신을 보냅니다.

자동 작성 대상에 생성

다음 그림에서는 제작자가 존재하지 않는 대상에게 메시지를 보내고 대상을 자동으로 작성해야 하는 경우에 대상이 어떻게 작성되고 복제되는지를 보여 줍니다.

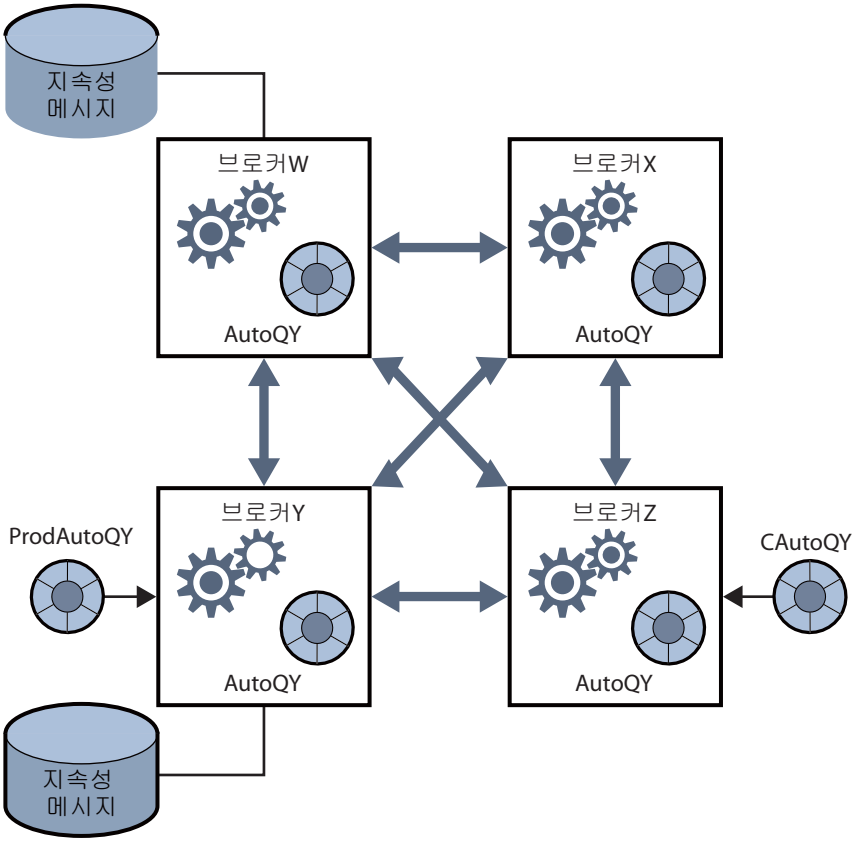

그림 4–3 클러스터에서의 대상 복제: 자동 작성된 대상

1. 제작자 ProdAutoQY는 브로커에 없는 대상인 AutoQY로 메시지를 보냅니다.
2. 브로커는 지속적으로 메시지를 처리하고 대상 AutoQY를 만듭니다.

   자동 생성된 대상이 클러스터 전체에 걸쳐 자동으로 복제되지는 않습니다. 소비자가 대기열 AutoQY로부터 메시지를 받도록 선택한 경우에만 소비자의 홈 브로커는 대상 AutoQY를 만들고 소비자에게 메시지를 전달합니다. 소비자가 자동 작성 대상을 만들 때 대상은 클러스터 전체로 복제됩니다. 이 예에서 소비자 CAutoQY가 대상을 작성할 때 복제가 이루어집니다.

주제 대상에 생성

다음 그림에서는 클라이언트가 관리자에 의해 작성된 주제 대상에 메시지를 게시할 때 클러스터에서 대상이 작성되고 복제되는 방법을 보여 줍니다.

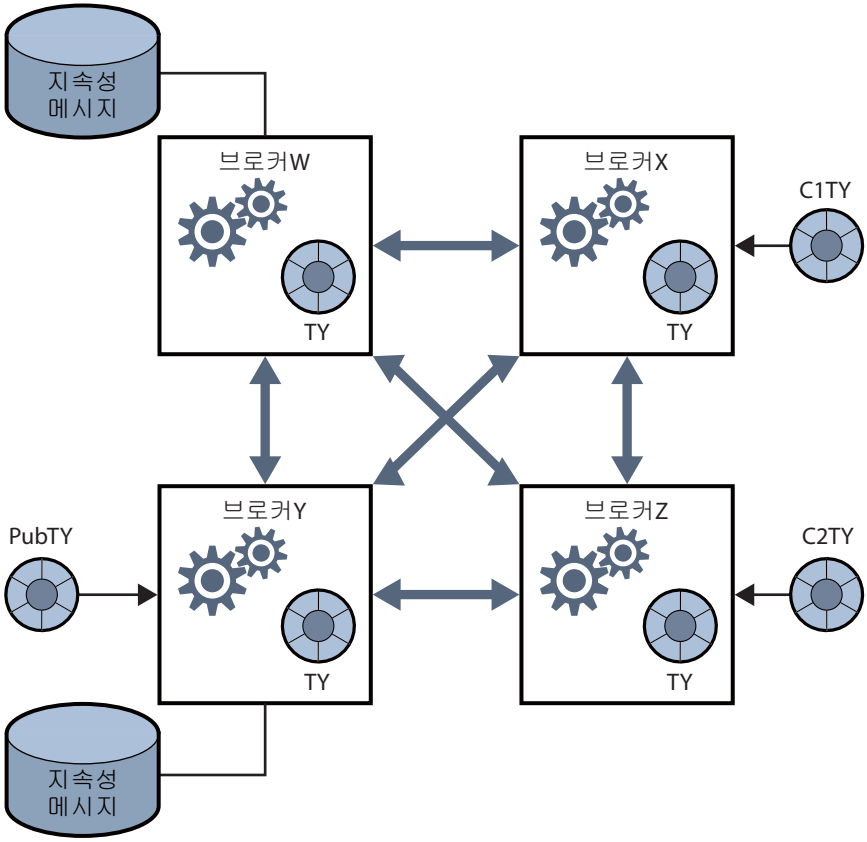

그림 4–4 클러스터에서의 대상 복제: 주제에 게시

1. 관리자가 물리적 주제 대상인 TY를 만듭니다. 관리 작성 대상인 TY는 대상이 사용되기 전에 브로커 클러스터 전체로 복제됩니다.
2. 게시자 PubTY는 주제 TY로 메시지를 보냅니다.
3. 홈 브로커 BrokerY는 TY에 게시된 모든 메시지를 지속적으로 처리하고 이 메시지의 선택 기준에 일치하는 모든 주제 가입자에게로 메시지 경로를 지정합니다. 이 예에서 C1TY 및 C2TY는 주제 TY에 가입됩니다.

연결 또는 브로커 오류 발생 시 대상 처리

표 4–2에서는 클러스터에서 여러 종류의 대상이 복제되고 삭제되는 방법에 대해 설명합니다.

표 4–2 클러스터에서 대상 처리

대상	전파 및 삭제
관리 작성	대상이 만들어지면 클러스터 내에서 전파되고 각 브로커는 대상 정보를 지속적으로 저장합니다. 관리자가 명시적으로 대상을 삭제하면 해당 대상은 삭제됩니다. 마스터 브로커가 있으면 작성 및 삭제 기록이 마스터 브로커에 저장되어 클러스터 내의 브로커가 상태 정보를 동기화할 수 있습니다.
임시	대상이 만들어지면 클러스터 곳곳으로 전파됩니다. 임시 대상과 관련된 소비자가 다시 연결되도록 허용되면 대상은 소비자의 홈 브로커에 지속적으로 저장됩니다. 그렇지 않은 경우 대상이 저장될 수 없습니다. 소비자의 연결이 끊어지면 모든 브로커에서 대상이 삭제됩니다. 소비자의 홈 브로커가 충돌하여 소비자가 다시 연결되도록 허용된 경우 이 소비자와 관련된 임시 대상이 모니터링됩니다. 소비자 클라이언트가 특정 시간 내에 다시 연결되지 않으면 클라이언트에서 오류가 발생한 것으로 가정하고 대상이 삭제됩니다.
자동 작성	제작자가 만들어졌으나 대상이 없는 경우 대상은 제작자의 홈 브로커에 만들어집니다. 존재하지 않는 대상에 대해 소비자가 만들어지면 소비자 및 대상 정보는 클러스터 전체로 전파됩니다. 자동 작성 대상은 관리자에 의해 명시적으로 삭제되거나 다음과 같은 경우 각 브로커에 의해 자동으로 삭제됩니다. ■ 특정 시간 동안 소비자 또는 메시지가 없는 경우 ■ 브로커를 다시 시작하거나 해당 대상에 대한 메시지가 없는 경우

클러스터 구성

시작 시 클러스터의 브로커 간에 연결을 설정하려면 각 브로커는 다른 모든 브로커(마스터 브로커 포함)에 대한 호스트 이름 및 포트 번호를 전달 받아야 합니다. 이 정보는 클러스터의 모든 브로커에 대해 동일해야 하는 일련의 **클러스터 구성 등록 정보**에 의해 지정됩니다. 각 브로커에 대해 개별적으로 구성 등록 정보를 지정할 수 있지만 이 방법은 오류가 발생하기 쉽고 클러스터 구성의 일관성이 손상될 수 있습니다. 대신, 시작 시 각 브로커가 참조하는 하나의 중앙 **클러스터 구성 파일**에 모든 구성 등록 정보를 두는 것이 좋습니다. 이렇게 하면 모든 브로커가 동일한 구성 정보를 공유하게 됩니다.

클러스터 구성 등록 정보에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Cluster Configuration Properties”를 참조하십시오.

주 – 원래 클러스터 구성 파일은 클러스터 구성 용도로 만들어졌지만 클러스터의 모든 브로커에서 공유하는 다른 등록 정보를 저장하기에도 편리한 장소입니다.

클러스터 동기화

클러스터의 구성이 변경될 때마다 변경에 대한 정보가 클러스터의 모든 브로커로 자동 전파됩니다. 다음과 같은 경우에 클러스터 구성이 변경됩니다.

- 클러스터 브로커 중 하나에서 대상이 작성되거나 삭제된 경우
- 대상의 등록 정보가 변경된 경우
- 메시지 소비자가 해당 홈 브로커에 등록된 경우
- 메시지 소비자와 홈 브로커 사이의 연결이(명시적으로 또는 클라이언트, 브로커 또는 네트워크 오류로 인해) 끊기는 경우
- 메시지 소비자가 특정 주제에 대한 영구 가입을 설정하는 경우

이러한 구성 변경 정보는 변경 시 온라인 상태인 클러스터의 모든 브로커로 즉시 전파됩니다. 하지만 오프라인 상태의 브로커(예: 충돌한 브로커)는 변경이 발생할 때 변경에 대한 알림을 수신하지 않습니다. 오프라인 상태인 브로커를 수용하기 위해 Message Queue는 작성되거나 삭제된 모든 지속성 항목(대상 및 영구 가입)을 기록하여 클러스터에 대한 **구성 변경 기록**을 유지 관리합니다. 오프라인 상태인 브로커가 다시 온라인 상태로 되면(또는 새 브로커가 클러스터에 추가되는 경우) 해당 브로커는 대상 및 영구 가입자에 대한 정보를 이 기록에서 참조한 다음 다른 브로커와 현재 활성 메시지 소비자에 대한 정보를 교환합니다.

클러스터에서 **마스터 브로커**로 지정된 한 브로커가 구성 변경 기록의 유지 관리를 담당합니다. 다른 브로커는 마스터 브로커 없이는 초기화를 완료할 수 없으므로 클러스터 내에서 마스터 브로커가 항상 먼저 시작되어야 합니다. 마스터 브로커가 오프라인 상태로 되면 다른 브로커가 구성 변경 기록에 액세스할 수 없으므로 구성 정보를 클러스터 전체에 전파할 수 없습니다. 이 상태에서 대상이나 영구 가입을 작성, 재구성, 삭제하려는 경우 또는 영구 가입 재활성화와 같은 관련 작업을 시도할 경우 예외가 발생합니다. 하지만 비관리 메시지 전달은 계속해서 정상적으로 작동합니다. 마스터 브로커 및 구성 변경 기록의 사용은 선택 사항입니다. 마스터 브로커 및 구성 변경 기록은 클러스터 구성 변경 또는 브로커 오류 발생 후 클러스터 동기화를 고려하는 경우에만 필요합니다.

◆ ◆ ◆ 5 장

5 Message Queue와 J2EE

Java 2 Platform, Enterprise Edition(J2EE 플랫폼)은 다중 계층 및 서버의 시스템 기능에 의존하는 클라이언트 엔터프라이즈 응용 프로그램을 호스트하는 표준 서버 플랫폼의 사양입니다. J2EE 플랫폼의 요구 사항 중 하나는 분산 구성 요소가 안정적인 비동기식 메시징을 통해 상호 작용할 수 있어야 한다는 것입니다. 이러한 상호 작용은 JMS 공급자를 사용하면 가능합니다. 실제로 Message Queue는 J2EE 플랫폼의 참조 JMS 구현입니다.

이 장에서는 J2EE 플랫폼 환경에서 JMS 지원을 구현한 결과를 살펴봅니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 81 페이지 "JMS/J2EE 프로그래밍: Message-Driven Bean"
- 83 페이지 "J2EE Application Server 지원"

Message Queue를 J2EE 호환 응용 프로그램 서버를 위한 JMS 공급자로 사용하는 것에 대한 자세한 내용은 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 17 장, "JMS Resource Adapter Property Reference"를 참조하십시오.

JMS/J2EE 프로그래밍: Message-Driven Bean

2 장에서 소개한 일반적인 JMS 클라이언트 프로그래밍 모델 외에도 J2EE 플랫폼 응용 프로그램 컨텍스트에서 사용하는, 보다 특수화된 JMS 버전이 있습니다. 이 특수화된 클라이언트를 *Message-Driven Bean*이라고 부르며, EJB(Enterprise JavaBeans) 구성 요소 중 하나로서EJB 2.0 이상 사양(`http://java.sun.com/products/ejb/docs.html`)에 설명되어 있습니다.

Message-Driven Bean은 비동기식 메시징을 제공합니다. 다른 EJB 구성 요소(Session Bean과 Entity Bean)는 표준 EJB 인터페이스를 통한 동기식 호출만 가능합니다. 그러나 엔터프라이즈 응용 프로그램에서는 서버측 구성 요소가 서버 자원을 독점하는 일 없이 통신할 수 있도록 비동기식 메시징이 필요할 때가 많습니다. 서버측 구성 요소가 응용 프로그램 이벤트에 응답해야 하는 응용 프로그램에서는 메시지 제작자와 밀접하게

연결되지 않으면서 메시지를 수신하고 소비할 수 있는 EJB 구성 요소가 필요합니다. 또한 엔터프라이즈 응용 프로그램에서 이러한 기능은 로드 증가에 따라 확장되어야 합니다.

MDB(Message-Driven Bean)는 특수한 EJB 컨테이너가 지원하는 EJB 구성 요소이며, 해당 컨테이너는 지원하는 구성 요소에 대해 분산 서비스를 제공합니다.

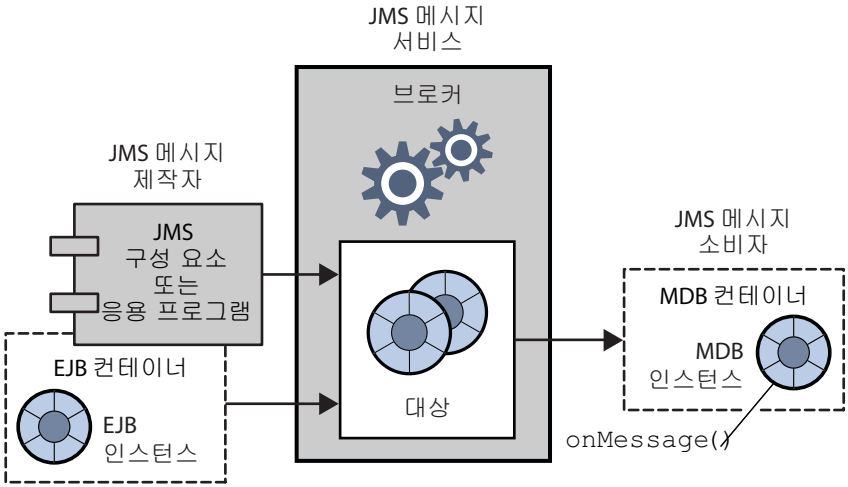

그림 5–1 MDB와의 메시지

- JMS MDB(message driven bean)는 JMS `MessageListener` 인터페이스를 구현하는 EJB입니다. `onMessage` 메소드(MDB 개발자가 작성)는 MDB 컨테이너가 메시지를 수신할 때 호출됩니다. `onMessage()` 메소드는 표준 `MessageListener` 객체의 `onMessage()` 메소드처럼 이 메시지를 소비합니다. MDB에서는 다른 EJB 구성 요소에서처럼 메소드를 원격 호출하지 않습니다. 따라서 그와 연관된 홈 또는 원격 인터페이스가 없습니다. MDB는 단일 대상에서 온 메시지를 소비할 수 있습니다. 그림 5–1에서 확인할 수 있듯이 독립형 JMS 응용 프로그램, JMS 구성 요소, EJB 구성 요소 또는 웹 구성 요소에서 메시지를 생성할 수 있습니다.
- 특수한 EJB 컨테이너가 MDB를 지원합니다. MDB의 인스턴스를 만들고 이 인스턴스가 비동기식으로 메시지를 소비하도록 설정합니다. 이 컨테이너는 메시지 서비스와 연결을 설정하고(인증 포함), 지정된 대상과 연관된 세션 풀을 생성하며 풀링된 세션 간의 메시지 배포를 관리합니다. 이 컨테이너는 MDB 인스턴스의 라이프사이클을 제어하므로 MDB 인스턴스 풀에서 받는 메시지 로드를 수용할 수 있도록 관리합니다.

  MDB와 관련된 배포 설명자는 컨테이너가 메시지 소비 설정 시 사용하는 연결 팩토리 및 대상 속성을 지정합니다. 이 배포 설명자는 배포 도구에서 컨테이너 구성 시 필요로 하는 다른 정보를 포함할 수도 있습니다. 이 컨테이너 각각은 단일 MDB 인스턴스를 지원합니다.

J2EE Application Server 지원

J2EE 아키텍처에서 EJB 컨테이너는 J2EE Application Server가 호스팅합니다. Application Server는 트랜잭션 관리자, 지속성 관리자, 이름 서비스 그리고 메시징과 MDB의 경우 JMS 공급자 등 다양한 컨테이너에서 필요로 하는 자원을 제공합니다.

Sun Java System Application Server의 경우 JMS 메시징 자원은 Sun Java System Message Queue에서 제공합니다.

- Sun Java System Application Server 7.0의 경우 Message Queue 메시징 시스템이 Application Server에 원시 JMS 공급자로 통합되어 있습니다.
- Sun J2EE 1.4 Application Server의 경우 Message Queue가 Application Server에 내장된 JMS 자원 어댑터로 플러그인되어 있습니다.

Application Server의 향후 릴리스에서는 Message Queue가 표준 자원 어댑터 배포 및 구성 방법을 사용하는 Application Server에 플러그인됩니다.

J2EE 아키텍처에 대한 자세한 내용은 J2EE 플랫폼 사양(`http://java.sun.com/j2ee/download.html#platformspec`)을 참조하십시오.

JMS 자원 어댑터

자원 어댑터는 J2EE 1.4를 준수하는 Application Server에 추가 기능을 플러그인하는 표준화된 방법입니다. J2EE Connector Architecture(J2EECA) 1.5 사양에 의해 정의되는 이 표준에 따라 Application Server가 표준 방식으로 외부 시스템과 상호 작용할 수 있습니다. 외부 시스템은 EIS(Enterprise Information System)를 비롯하여 JMS 공급자와 같은 메시징 시스템을 포함할 수 있습니다. Message Queue에는 Application Server가 Message Queue를 JMS 공급자로 사용할 수 있도록 JMS 자원 어댑터가 포함되어 있습니다.

JMS 자원 어댑터를 Application Server에 플러그인하면 Application Server에서 배포되어 실행 중인 J2EE 구성 요소가 JMS 메시지를 교환할 수 있습니다. 이러한 구성 요소에 필요한 JMS 연결 팩토리 및 대상 관리 객체는 J2EE Application Server 관리 도구를 사용하여 만들고 구성할 수 있습니다.

그러나 브로커 관리 및 물리적 대상 관리와 같은 다른 관리 작업은 J2EECA 사양에 포함되어 있지 않으므로 공급자 고유의 도구를 통해서만 수행할 수 있습니다.

Message Queue 자원 어댑터는 Sun J2EE 1.4 Application Server에 통합되어 있습니다. 그러나 다른 J2EE 1.4 Application Server에서는 아직 인증되지 않았습니다.

Message Queue 자원 어댑터는 운영 체제에 종속된 디렉토리에 있는 단일 파일(`imqjmsra.rar`) 입니다(**Sun Java System Message Queue 3.7 UR1 관리 설명서**의 17 장, “JMS Resource Adapter Property Reference”). `imqjmsra.rar` 파일은 자원 어댑터 배포 설명자(`ra.xml`)를 비롯하여 어댑터를 사용하기 위해 Application Server에서 필요한 JAR 파일도 포함합니다.

모든 J2EE -1.4 호환 Application Server에서 해당 Application Server와 함께 제공되는 자원 어댑터 배포 및 구성 지침에 따라 Message Queue 자원 어댑터를 사용할 수 있습니다. 상용 J2EE 1.4 Application Server가 출시되고 Message Queue 자원 어댑터가 이러한 Application Server에서 인증되면 Message Queue 문서에서 관련 배포 및 구성 절차에 대한 구체적인 정보를 제공할 것입니다.

◆ ◆ ◆ 부 록 A

Message Queue의 선택적 JMS 기능 구현

JMS 사양은 선택 사항인 특정 항목을 나타냅니다. 각 JMS 공급자(공급업체)가 해당 항목의 구현 여부를 선택합니다. 이 부록에서는 Message Queue 제품이 JMS 선택 항목을 처리하는 방법에 대해 설명합니다.

표 A–1에서는 Message Queue 서비스가 JMS 선택 항목을 처리하는 방법을 설명합니다.

선택적 기능

표 A–1 선택적 JMS 기능

JMS 사양 섹션	설명 및 Message Queue 구현
3.4.3 JMSMessageID	"메시지 ID가 메시지의 크기를 생성하고 늘리려고 하므로 일부 JMS 공급자는 응용 프로그램에서 메시지 ID를 사용하지 않는다는 힌트가 있으면 메시지 오버헤드를 최적화할 수 있습니다. JMS Message Producer는 메시지 아이디 비활성화를 위한 힌트를 제공합니다." **Message Queue 구현**: 제품에서는 메시지 ID 생성을 비활성화하지 않습니다(MessageProducer의 모든 setDisableMessageID() 호출이 무시됨). 모든 메시지에는 유효한 MessageID 값이 있습니다.
3.4.12 메시지 헤더 필드 대체	"JMS는 관리자가 이러한 헤더 필드 값을 대체하는 방법에 대해 특별히 정의하지 않습니다. JMS 공급자가 이 관리 옵션을 지원해야 하는 것은 아닙니다." **Message Queue 구현**: Message Queue 제품은 클라이언트 런타임 구성을 통해 메시지 헤더 필드 값의 관리 대체를 지원합니다(42 페이지 "메시지 헤더" 참조).

표 A–1 선택적 JMS 기능 *(계속)*

JMS 사양 섹션	설명 및 Message Queue 구현
3.5.9 JMS 정의 등록 정보	"JMS는 JMS 정의 등록 정보를 위한 'JMSX' 등록 정보 이름 접두어를 예약합니다." "별도로 명시되지 않은 경우 이러한 등록 정보 지원은 선택 사항입니다." **Message Queue 구현**: JMS 1.1 사양에서 정의한 JMSX 등록 정보는 Message Queue 제품에서 지원됩니다(**Sun Java System Message Queue 3.7 UR1 관리 설명서**의 부록 B, "Message Queue 인터페이스의 안정성" 참조).
3.5.10 공급자별 등록 정보	"JMS는 공급자별 등록 정보를 위한 'JMS_<*vendor_name*>' 등록 정보 이름 접두어를 예약합니다." **Message Queue 구현**: 공급자별 등록 정보는 공급자 고유 클라이언트에서 JMS 사용을 지원하는 데 필요한 특수 기능을 제공하는 것을 목적으로 합니다. JMS 간 메시징에는 사용할 수 없습니다.
4.4.8 분산 트랜잭션	"JMS에서는 공급자가 분산 트랜잭션을 지원하지 않아도 됩니다." **Message Queue 구현**: 이번 Message Queue 제품 릴리스에서는 분산 트랜잭션이 지원됩니다(49 페이지 "트랜잭션" 참조).
4.4.9 다중 세션	"PTP <지점간 배포 모델>의 경우, JMS는 동일한 대기열에 대한 동시 QueueReceivers의 의미를 지정하지 않지만, 공급자가 이를 지원하는 것을 금지하지는 않습니다." 자세한 내용은 JMS 사양의 5.8절을 참조하십시오. **Message Queue 구현**: Message Queue 구현에서는 다중 소비자로 대기열을 전달할 수 있습니다. 자세한 내용은 34 페이지 "지점간 메시징"을 참조하십시오.

◆ ◆ ◆ 부 록 B

Message Queue 기능

Message Queue 서비스는 안정적이고 유연한 비동기식 메시지 전달을 위해 JMS 1.1 사양을 완전히 구현합니다. JMS 호환성 관련 문제에 대한 자세한 내용은 부록 A을 참조하십시오. 그러나 Message Queue는 JMS 요구 사항을 훨씬 능가하는 성능과 기능을 갖추고 있습니다. 이러한 기능을 사용하여 24시간 중차대한 작업으로 무수히 많은 메시지를 교환하는 많은 수의 분산 구성 요소로 구성된 시스템을 통합하고 모니터링할 수 있습니다.

이 설명서에서는 Message Queue 서비스에 대한 설명으로 이러한 기능을 소개합니다. 사용자의 편의를 위해 이 부록에서는 Message Queue 기능을 요약했습니다. 이 설명서에는 각 기능이 간략하게 설명되어 있으며 기능을 사용하는 데 필요한 작업이 요약되어 있습니다. 이러한 기능을 소개하는 이 설명서의 절과 이러한 기능을 자세히 설명하는 Message Queue 설명서 세트의 특정 문서에 참조가 제공되어 있습니다.

부록 B에 알파벳순으로 나열된 Message Queue의 기능은 크게 다음 범주로 나눌 수 있습니다.

- 통합 지원
  - HTTP 연결
  - 보안 연결
  - C 클라이언트 지원
  - SOAP 지원
  - J2EE 자원 어댑터
- 보안
  - 인증
  - 권한 부여
  - 암호화(보안 연결 참조)
- 확장성
  - 스레드 관리
  - 브로커 클러스터
  - 다중 소비자로의 대기열 전달

- 가용성
  - 메모리 자원 관리
  - 클라이언트에 대한 메시지 흐름 제어
  - 자동 재연결
  - 안정적인 데이터 지속성
  - 연결 핑
- 관리성
  - 관리 도구
  - 메시지 기반 모니터링 API
  - 조정 가능한 성능
  - 구성 가능한 물리적 대상
  - 브로커 구성
  - 사용 불능 메시지 대기열
- 성능
  - 메시지 압축
  - 조정 가능한 성능
  - 구성 가능한 물리적 대상
- 유연한 서버 구성
  - 구성 가능한 지속성
  - LDAP 서버 지원
  - JNDI 서버 지원

기능 목록

표 B–1 Message Queue 기능

기능	설명 및 참조
관리 도구	Message Queue 서비스에는 대상, 트랜잭션, 영구 가입, 관리 대상 객체 저장소, 사용자 저장소, JDBC 호환 데이터 저장소와 서버 인증서를 관리하기 위한 GUI 및 명령줄 도구가 포함되어 있습니다. **참조** 66 페이지 “관리 도구” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 1 장, “Administrative Tasks and Tools”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
인증	브로커와 연결하려는 사용자를 인증합니다. Message Queue 서비스를 사용하여 사용자는 사용자 저장소에 저장된 값과 사용자 이름 및 비밀번호를 확인하여 브로커에 연결할 수 있습니다. 저장소는 Message Queue 또는 LDAP 저장소(LDAP v2 또는 v3 프로토콜)와 함께 제공되는 플랫 파일 저장소일 수 있습니다. **사용 방법** 1. 사용자 저장소를 만들거나 기본 인스턴스를 사용합니다. 2. `imqusermgr` 도구를 사용하여 저장소를 채웁니다. **참조** 63 페이지 "인증 및 권한 부여" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 7 장, "Managing Security"
권한 부여	특정 작업을 수행하도록 사용자에게 권한을 부여합니다. Message Queue 서비스를 사용하여 사용자 및 사용자 그룹이 수행할 수 있는 작업을 지정하는 액세스 제어 등록 정보 파일을 만들 수 있습니다. 클라이언트가 연결을 만들거나 제작자 또는 소비자를 만들거나 대기열을 찾는 경우 브로커는 이 파일을 확인합니다. **사용 방법** 브로커 인스턴스에 대해 자동으로 만들어진 액세스 제어 등록 정보 파일을 편집합니다. **참조** 63 페이지 "인증 및 권한 부여" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 7 장, "Managing Security"

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
자동 재연결	관리자는 연결 실패 시 자동 재연결을 사용할 수 있도록 연결 팩토리 관리 대상 객체의 연결 속성을 설정합니다. 같은 브로커로 다시 연결되거나 또는 클러스터가 사용 중이면 클러스터 내의 다른 브로커로 다시 연결될 수 있습니다. 재연결 시도 횟수 및 시도 간격을 지정할 수 있습니다. 클러스터된 브로커의 경우 브로커 목록을 통한 반복 횟수 및 특정 순서로 목록 반복 여부를 지정할 수 있습니다. **참조** 57 페이지 “연결 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 8 장, “관리 대상 객체 관리”
브로커 클러스터	관리자는 브로커 인스턴스를 브로커 클러스터로 그룹화하여 많은 브로커 인스턴스 간의 클라이언트 연결 및 메시지 전달의 균형을 조정할 수 있습니다. **사용 방법** 1. 클러스터에 있는 각 브로커의 클러스터 구성 등록 정보를 지정합니다. 구성 파일을 사용하거나 각 브로커의 등록 정보를 설정하여 지정할 수 있습니다. 2. 마스터 브로커가 있으면 마스터 브로커를 시작합니다. 3. 클러스터에 있는 다른 브로커를 시작합니다. **참조** 4 장 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 9 장, “Working With Broker Clusters”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
브로커 구성	관리자는 Message Queue 서비스 성능을 조정하도록 브로커 등록 정보를 설정할 수 있습니다. 여기에는 라우팅 서비스, 지속성 서비스, 보안, 모니터링 및 관리 대상 객체 관리가 포함됩니다. **참조** 3 장 **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 4 장, “Configuring a Broker”
C 클라이언트 지원	C 클라이언트는 Message Queue 메시징 서비스를 사용하여 메시지를 보내고 받을 수 있습니다. C API에서는 기존 C 및 C++ 응용 프로그램이 JMS 기반 메시징에 참여할 수 있습니다. Message Queue의 C API는 관리 대상 객체, 맵, 스트림 또는 객체 메시지 본문 유형, 분산 트랜잭션 및 대기열 브라우저 사용한다는 점을 제외하면 표준 JMS 기능의 대부분을 지원하는 C 클라이언트 런타임에서 지원됩니다. 또한 C 클라이언트 런타임은 대부분의 Message Queue 엔터프라이즈 기능을 지원하지 않습니다. **참조** 53 페이지 “Java 및 C 클라이언트” **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for C Clients**
압축 메시지	Java 클라이언트는 전송 중인 메시지를 클라이언트 런타임에서 압축하도록 메시지 등록 정보를 설정할 수 있습니다. 소비자측의 런타임에서 소비자에게 메시지를 전달하기 전에 메시지의 압축을 풉니다. 압축 메시지가 실제로 성능을 향상시키는지를 결정하는 데 사용할 수 있는 추가 등록 정보가 제공됩니다. **참조** 43 페이지 “메시지 본문” **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for Java Clients**의 “Managing Message Size”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
구성 가능한 지속성	관리자는 Message Queue 또는 Oracle 8i와 같은 JDBC 호환 데이터베이스와 함께 제공되는 파일 기반 영구 저장소를 사용하도록 브로커를 구성할 수 있습니다. **사용 방법** 파일 시스템 영구 저장소 또는 JDBC 호환 저장소와 관련된 브로커 등록 정보를 설정합니다. **참조** 61 페이지 “지속성 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Configuring a Persistent Data Store”
구성 가능한 물리적 대상	관리자는 대상을 만들 때 물리적 대상 등록 정보를 설정하여 일부 메시징 동작을 정의할 수 있습니다. 다음 동작은 모든 대상에 대해 구성할 수 있습니다. 소비되지 않은 최대 메시지 수 또는 그 메시지에 허용되는 최대 메모리 양, 메모리 제한에 도달한 경우 브로커가 거부해야 하는 메시지, 최대 제작자 및 소비자 수, 최대 메시지 크기, 일괄적으로 전달되는 최대 메시지 수, 대상이 로컬 소비자에게만 전달 가능한지 여부, 대상의 사용 불능 메시지를 사용 불능 메시지 대기열로 이동 가능한지 여부 **참조** 58 페이지 “대상 및 라우팅 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 6 장, “Managing Physical Destinations”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
연결 핑	관리자는 연결 팩토리 속성을 설정하여 클라이언트 런타임에서 브로커로의 핑 작업 빈도를 지정할 수 있습니다. 이렇게 하면 클라이언트는 실패한 연결을 우선적으로 감지할 수 있습니다. **참조** 57 페이지 "연결 서비스" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 "Connection Services"
사용 불능 메시지 대기열	Message Queue 메시지 서비스는 사용 불능 메시지 대기열을 만들어 만료되거나 브로커에서 처리할 수 없는 메시지를 보관합니다. 대기열의 내용을 확인하여 시스템 성능을 모니터링하거나 조정하고 시스템 성능 관련 문제를 해결할 수 있습니다. **참조** 58 페이지 "대상 및 라우팅 서비스" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 6 장, "Managing Physical Destinations"

표 B–1 Message Queue 기능 *(계속)*

기능	설명 및 참조
HTTP 연결	Java 클라이언트는 브로커와의 HTTP 연결을 생성할 수 있습니다. HTTP 전송을 사용하면 방화벽을 통해 메시지를 전달할 수 있습니다. Message Queue는 웹 서버 환경에서 실행되는 HTTP 터널 서블릿을 사용하여 HTTP 지원을 구현합니다. 클라이언트가 생성하는 메시지는 클라이언트 런타임에서 HTTP 요청으로 래핑된 다음 HTTP를 통해 방화벽을 거쳐 터널 서블릿으로 전달됩니다. 터널 서블릿은 HTTP 요청으로부터 JMS 메시지를 추출하고 TCP/IP를 통해 메시지를 브로커로 전달합니다. **사용 방법** 1. 웹 서버에 HTTP 터널 서블릿을 배포합니다. 2. 브로커의 `httpjms` 연결 서비스를 구성하고 브로커를 시작합니다. 3. HTTP 연결을 구성합니다. 4. 브로커로 HTTP 연결을 가져옵니다. (Java 클라이언트만 해당) **참조** 27 페이지 “브로커에 연결” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 부록 C, “HTTP/HTTPS 지원”
대화형 모니터링	관리자는 `imqcmd metrics` 명령을 사용하여 원격으로 브로커를 모니터링할 수 있습니다. 모니터링된 데이터에는 JVM 메트릭, 브로커 메시지 흐름, 연결, 연결 자원, 메시지, 대상 메시지 흐름, 대상 소비자, 대상 자원 사용이 포함됩니다. **참조** 64 페이지 “모니터링 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 10 장, “Monitoring a Broker”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
J2EE 자원 어댑터	Message Queue는 J2EE 호환 Application Server로 플러그인될 수 있는 자원 어댑터를 제공합니다. Message Queue를 JMS 공급자로 사용하여 Application Server는 안정적인 비동기식 메시지를 통해 Application Server에서 상호 작용할 수 있는 분산 구성 요소 실행이라는 J2EE 요구 사항을 충족합니다. **사용 방법** 어댑터 속성을 설정하여 어댑터를 구성합니다. **참조** 83 페이지 "J2EE Application Server 지원" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 17 장, "JMS Resource Adapter Property Reference"
JNDI 서비스 공급자 지원	클라이언트는 JNDI API를 사용하여 관리 대상 객체를 조회할 수 있습니다. 관리자는 `imqobjmgr` 유틸리티를 사용하여 JNDI를 통해 액세스 가능한 객체 저장소에서 관리 대상 객체를 추가, 나열, 업데이트하고 삭제할 수 있습니다. **참조** 66 페이지 "관리 도구" **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 8 장, "관리 대상 객체 관리"

표 B–1 Message Queue 기능 *(계속)*

기능	설명 및 참조
LDAP 서버 지원	관리자는 관리 대상 객체 저장소와 인증 및 권한 부여에 필요한 사용자 정보 저장을 위해 LDAP 서버를 사용할 수 있습니다. 기본적으로 Message Queue는 이러한 데이터를 위해 파일 기반 저장소를 제공합니다. **관리 대상 객체 사용 방법** 1. 공급업체에서 제공한 도구를 사용하여 사용자 저장소를 채우고 관리합니다. 2. LDAP 관련 브로커 등록 정보를 설정합니다. 3. 관리 사용자에 대한 액세스 제어를 설정합니다. **참조** 62 페이지 “보안 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 7 장, “Managing Security” **사용자 저장소 사용 방법** 1. 공급업체에서 제공한 도구를 사용하여 LDAP 서버를 설정합니다. 2. LDAP 관련 브로커 등록 정보를 설정하여 초기 컨텍스트 및 저장소 위치를 정의합니다. 3. LDAP 서버 작업을 보호하기 위한 LDAP 관련 브로커 등록 정보를 설정합니다. **참조** **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 7 장, “Managing Security”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
메모리 자원 관리	관리자는 다음 동작을 구성할 수 있습니다. 1. 최대 제작자 수, 메시지의 최대 크기 및 하나의 메시지에 대한 최대 크기를 지정하도록 대상에서 등록 정보를 설정합니다. 2. 메시지 흐름을 제어하도록 대상에서 등록 정보를 설정합니다. 3. 각 대상에 대한 메시지 흐름을 관리하도록 대상에서 등록 정보를 설정합니다. 4. 해당 브로커의 모든 대상에서 메시지 제한을 지정하도록 브로커에서 등록 정보를 설정합니다. 5. 브로커가 메모리 과부하 방지 조치의 수위를 더 높이게 되는, 사용 가능 시스템 메모리의 임계값을 지정하도록 브로커에서 등록 정보를 설정합니다. 조치는 다음과 같이 메모리 자원 상태에 따라 달라집니다. **참조** 58 페이지 “대상 및 라우팅 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 4 장, “Configuring a Broker”
메시지 압축	개발자는 메시지를 보내기 전에 클라이언트 런타임에서 메시지를 압축하도록 메시지 헤더 등록 정보를 설정할 수 있습니다. 클라이언트 런타임은 소비자에게 메시지를 전달하기 전에 메시지의 압축을 풉니다. **참조** 43 페이지 “메시지 등록 정보” **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for Java Clients**의 “Message Compression”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
클라이언트에 대한 메시지 흐름 제어	관리자나 개발자는 다양한 흐름 제한 및 측정 체계를 지정하여 페이로드 및 제어 메시지의 충돌을 최소화하고 그로 인해 메시지의 처리량을 최대화하도록 연결을 구성할 수 있습니다. **사용 방법** 연결 팩토리 관리 대상 객체의 흐름 제어 속성을 설정(관리자)하거나 연결 팩토리의 흐름 제어 등록 정보를 설정(개발자)합니다. **참조** 40 페이지 “연결 팩토리 및 연결” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Connection Services” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “연결 팩토리 속성”
메시지 기반 모니터링 API	Java 클라이언트는 모니터링 API를 사용하여 사용자 정의 모니터링 응용 프로그램을 만들 수 있습니다. 모니터링 응용 프로그램은 특수한 메트릭 주제 대상에서 메트릭 메시지를 검색하는 소비자입니다. **사용 방법** 1. 메트릭 모니터링 클라이언트를 작성합니다. 2. 브로커의 메트릭 메시지 제작자를 구성하도록 브로커 등록 정보를 설정합니다. 3. 메트릭 주제 대상에 대한 액세스 제어를 설정합니다. 4. 모니터링 클라이언트를 시작합니다. **참조** 64 페이지 “모니터링 서비스” **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for Java Clients**의 4 장, “Using the Metrics Monitoring API” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 10 장, “Monitoring a Broker”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
다중 소비자로의 대기열 전달	클라이언트는 특정 대기열에 대해 여러 소비자를 등록할 수 있습니다. 관리자는 대기열에 대한 최대 활성 소비자 수 및 최대 백업 소비자 수를 지정할 수 있습니다. 브로커는 시스템 크기를 조정할 수 있도록 등록된 소비자들 간에 로드 균형을 조정하면서 메시지를 분산합니다. **사용 방법** 물리적 대상 등록 정보인 `maxNumActiveConsumers` 및 `maxNumBackupConsumers`를 설정합니다. **참조** 34 페이지 “지점간 메시징” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 15 장, “Physical Destination Property Reference”
안정적인 데이터 지속성	안정성을 얻으려면 `imq.persist.file.sync.enabled` 등록 정보를 true로 설정하여 운영 체제에서 데이터를 영구 저장소에 동기식으로 기록해야 합니다. 이렇게 하면 시스템 중단으로 인한 데이터 손실을 방지할 수 있지만 성능은 저하됩니다. 데이터가 손실되지는 않지만 클러스터된 브로커에서 현재 이 데이터를 공유하지 않으므로 클러스터 내의 다른 브로커가 사용할 수 없게 됩니다. 시스템 백업 시 브로커는 안정적으로 작업을 다시 시작할 수 있습니다. **참조** 61 페이지 “지속성 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Persistence Properties” .

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
보안 연결	클라이언트는 TCP/IP 및 HTTP 전송을 통해 SSL(Secure Socket Layer) 표준에 기반한 메시지를 안전하게 전송할 수 있습니다. 이러한 SSL 기반 연결 서비스를 사용하면 클라이언트와 브로커 간에 전송되는 메시지를 암호화할 수 있습니다. SSL 지원은 자체 서명된 서버 인증서를 기반으로 합니다. Message Queue는 개인/공용 키 쌍을 생성하고 자체 서명 인증서에 공용 키를 포함시키는 유틸리티를 제공합니다. 이 인증서는 브로커와의 연결을 요청하는 클라이언트로 전달되고 클라이언트는 해당 인증서를 사용하여 암호화된 연결을 설정합니다. **사용 방법** 1. 자체 서명되었거나 서명된 인증서를 생성합니다. 2. 보안 서비스를 사용하도록 설정합니다. 3. 브로커를 시작합니다. 4. 클라이언트 보안 연결 등록 정보를 구성하고 클라이언트를 실행합니다. **참조** 27 페이지 “브로커에 연결” 62 페이지 “보안 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 7 장, “Managing Security” **Sun Java System Message Queue 3.7 UR1 Developer’s Guide for C Clients**의 “Working With Secure Connections”

표 B–1 Message Queue 기능 (계속)

기능	설명 및 참조
SOAP 지원	클라이언트는 SOAP(XML) 메시지를 받을 수 있으며 받은 메시지를 JMS 메시지로 래핑한 다음 Message Queue를 이용해 JMS 메시지처럼 교환할 수 있습니다. 클라이언트는 특수 서블릿을 사용하여 SOAP 메시지를 받고, 유틸리티 클래스를 사용하여 SOAP 메시지를 JMS 메시지로 래핑하며, 다른 유틸리티 클래스를 사용하여 JMS 메시지에서 SOAP 메시지를 추출할 수 있습니다. 클라이언트는 표준 SAAJ 라이브러리를 사용하여 SOAP 메시지를 어셈블 및 역어셈블할 수 있습니다. **참조** 52 페이지 “SOAP 메시지 작업” “Working With SOAP Message,” **Sun Java System Message Queue 3.7 UR1 Developer's Guide for Java Clients**의 5 장, “Working with SOAP Messages”
스레드 관리	관리자는 특정 연결 서비스에 할당된 최대 및 최소 스레드 수를 지정할 수 있습니다. 또한, 관리자는 유휴 연결 전용 스레드를 다른 연결에서 사용하도록 허용하는 공유 스레드 모델을 사용하여 연결 서비스에서 처리 능력을 증가시킬 수 있는지 여부도 결정할 수 있습니다. **사용 방법** 연결 서비스 스레드 관련 등록 정보를 설정합니다. **참조** 58 페이지 “스레드 풀 관리” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 4 장, “Configuring a Broker”

표 B–1 Message Queue 기능 *(계속)*

기능	설명 및 참조
조정 가능한 성능	관리자는 메모리 사용, 스레딩 자원, 메시지 흐름, 연결 서비스, 안정성 매개 변수 및 메시지 처리량과 시스템 성능에 영향을 미치는 기타 요소를 조정하도록 브로커 등록 정보를 설정할 수 있습니다 **참조** 64 페이지 “모니터링 서비스” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 “Monitoring Services” **Sun Java System Message Queue 3.7 UR1 관리 설명서**의 11 장, “Analyzing and Tuning a Message Service”

용어집

이 용어집에서는 Message Queue 사용 중에 접할 수 있는 용어와 개념에 대한 정보를 제공합니다. Sun Java System에서 사용되는 모든 용어를 설명하는 용어집은 **Sun Java Enterprise System Glossary** 를 참조하십시오.

JMS 공급자 메시징 시스템을 위한 JMS 인터페이스를 구현하고 해당 시스템을 구성하고 관리하는 데 필요한 관리 및 제어 기능을 추가하는 제품

관리 대상 객체 사전 구성된 객체. 공급자별 구현 세부 정보을 캡슐화하는 연결 팩토리 또는 대상으로서 관리자가 하나 이상의 JMS 클라이언트에서 사용하도록 생성합니다. 관리 대상 객체를 사용하면 JMS 클라이언트가 공급자 독립성을 갖게 됩니다. 관리 대상 객체는 관리자가 JNDI 이름 공간에 배치하며 JMS 클라이언트가 JNDI 조회를 사용하여 액세스합니다.

권한 부여 사용자가 연결 서비스나 대상과 같은 메시지 서비스 자원에 액세스하여 메시지 서비스에서 지원하는 특정 작업을 수행할 수 있는지 여부를 메시지 서비스가 결정하는 과정

그룹 연결, 대상 및 특정 작업에 대한 액세스 권한을 부여하기 위해 Message Queue 클라이언트의 사용자가 속하는 그룹

대기열 관리자가 지점간 전달 모델을 구현하기 위해 생성하는 객체. 메시지를 소비하는 클라이언트가 비활성 상태이더라도 대기열은 항상 메시지 보관이 가능합니다. 대기열은 제작자와 소비자 사이의 중간 저장소 역할을 합니다.

대상 생성된 메시지가 라우팅 및 이후 소비자로의 전달을 위해 이동하는 Message Queue 브로커상의 물리적 대상. 물리적 대상은 JMS 클라이언트가 자신이 누구를 위해 메시지를 생성하며 누구로부터 받은 메시지를 소비하는지 그 대상을 지정할 때 사용하는 관리 객체에 의해 식별 및 캡슐화됩니다.

데이터 저장소 브로커가 필요로 하는 정보(영구 가입, 대상 관련 데이터, 지속성 메시지, 감사 데이터 등)가 영구적으로 저장되는 데이터베이스

도메인 JMS 클라이언트가 JMS 메시징 작업을 프로그래밍할 때 사용하는 객체 집합. 두 가지 프로그래밍 도메인이 있습니다. 지점간 전달 모델을 위한 도메인과 게시/가입 전달 모델을 위한 도메인입니다.

메시지 메시징 클라이언트가 소비하는 비동기 요청, 보고서 또는 이벤트. 메시지는 헤더(필드 추가 가능)와 본문으로 구성됩니다. 메시지 헤더는 표준 필드 및 선택적 등록 정보를 지정합니다. 메시지 본문은 전송되는 데이터를 포함합니다.

메시지 서비스 분산 구성 요소 또는 응용 프로그램 간에 안정적인 비동기식 메시지 교환을 제공하는 미들웨어 서비스. 브로커, 클라이언트 런타임, 브로커가 자체 기능을 수행하는 데 필요한 여러 데이터 저장소 그리고 브로커를 구성, 모니터링하고 성능을 조정하는 데 필요한 관리 도구가 포함됩니다.

메시징 엔터프라이즈 응용 프로그램이 사용하는 비동기 요청, 보고서 또는 이벤트 시스템으로, 느슨하게 연결된 응용 프로그램이 안정적이고 안전하게 정보를 전송할 수 있게 합니다.

브로커 메시지 라우팅, 전달, 지속성, 보안 및 로깅을 관리하며 성능 및 자원 사용을 모니터링하고 조정할 인터페이스를 제공하는 Message Queue 실체

비동기식 메시징 메시지를 수신할 소비자의 준비 여부와 관계없이 메시지가 전송되는 메시지 교환. 즉 메시지 발신자는 발신 메소드가 반환될 때까지 기다릴 필요 없이 다른 작업을 진행할 수 있습니다. 메시지 소비자가 작업 중이거나 오프라인 상태인 경우, 일단 메시지가 전송된 다음 소비자가 준비되었을 때 수신됩니다.

사용 불능 메시지 정상 처리나 명시적 관리자 조치가 아닌 다른 이유로 시스템에서 제거된 메시지. 메시지가 만료되었거나, 메모리 제한 초과로 인해 대상에서 제거되었거나, 전달 시도가 실패했기 때문에 사용 불능으로 간주될 수 있습니다. 사용 불능 메시지를 사용 불능 메시지 대기열에 저장하도록 선택할 수 있습니다.

사용 불능 메시지 대기열 브로커 시작 시 자동으로 작성되어, 진단 용도로 사용 불능 메시지를 저장하는 데 사용되는 특별한 대상

선택기 메시지를 정렬하고 경로를 지정하기 위해 사용된 메시지 헤더 등록 정보. 메시지 서비스는 메시지 선택기에 지정된 기준에 따라 메시지 필터링 및 경로 지정을 수행합니다.

세션 메시지를 보내고 받는 단일 스레드 컨텍스트. 대기열 세션이거나 주제 세션이 될 수 있습니다.

소비자 대상으로부터 메시지를 수신하기 위해 세션에서 작성한 객체(`MessageConsumer`). 지점 간 전달 모델의 소비자는 수신자 또는 브라우저(`QueueReceiver` 또는 `ueueBrowser`)이고, 게시/가입 전달 모델의 소비자는 가입자(`TopicSubscriber`)입니다.

암호화 연결을 통한 전달 중 메시지가 훼손되지 않게 보호하는 메커니즘

연결 페이로드 메시지 및 제어 메시지를 모두 전달할 때 클라이언트와 브로커 간에 사용되는 통신 채널

연결 팩토리 클라이언트가 브로커와의 연결을 생성할 때 사용하는 관리 대상 객체. `ConnectionFactory` 객체, `QueueConnectionFactory` 객체 또는 `TopicConnectionFactory` 객체일 수 있습니다.

인증 검증된 사용자만 브로커에 연결할 수 있게 하는 과정

전달 모델 메시지가 전달되는 모델로서, 지점간 모델 또는 게시/가입 모델이 있습니다. JMS에는 각각 특정 클라이언트 런타임 객체와 특정 대상 유형(대기열 또는 주제)을 사용하는 별도의 프로그래밍 도메인과 통합 프로그래밍 도메인이 있습니다.

전달 모드 메시징의 신뢰성 지표. 메시지가 단 한 번만 전달되어 성공적으로 소비되는지(지속성 전달 모드) 또는 최대 1회 전달되는지(비지속성 전달 모드) 여부

제작자 대상으로의 메시지 전송에 사용되는 세션에서 작성한 객체(`MessageProducer`). 지점간 전달 모델에서 제작자는 발신자(`QueueSender`)이며, 게시/가입 전달 모델의 제작자는 게시자(`TopicPublisher`)입니다.

주제 관리자가 게시/가입 전달 모델을 구현하기 위해 생성하는 객체. 주제는 자신에게 전달된 메시지의 수집 및 배포를 담당하는, 내용 계층상의 노드로 간주할 수 있습니다. 메시지 게시자와 메시지 가입자는 중간에 있는 주제를 통해 구분됩니다.

클라이언트 메시징 서비스를 통해 다른 클라이언트와 상호 작용하면서 메시지를 주고 받는 응용 프로그램(또는 소프트웨어 구성 요소). 클라이언트는 제작자 클라이언트나 소비자 클라이언트 또는 두 클라이언트 모두가 될 수 있습니다.

클라이언트 런타임 메시징 클라이언트에게 Message Queue 메시지 서비스와의 인터페이스를 제공하는 Message Queue 소프트웨어. 클라이언트 런타임은 클라이언트가 대상에게 메시지를 보내고 대상으로부터 메시지를 받는 데 필요한 모든 작업을 지원합니다.

클라이언트 식별자 클라이언트를 대신하여 연결 및 그 객체를 Message Queue 브로커가 관리하는 상태와 연관시키는 식별자

클러스터 확장 가능한 메시징 서비스를 제공하는, 상호 연결된 둘 이상의 브로커

트랜잭션 완료하거나 또는 완전히 롤백해야 하는 기본 작업 단위

확인 안정적으로 전달될 수 있도록 클라이언트와 브로커 간에 교환되는 제어 메시지. 일반적인 두 가지 확인 유형은클라이언트 확인과 브로커 확인입니다.

색인

A

Application Server 및 Message Queue, 83-84
AUTO_ACKNOWLEDGE 모드, 48

B

BytesMessage 유형, 44

C

C 클라이언트, 29, 53, 91
CLIENT_ACKNOWLEDGE 모드, 48

D

DUPS_OK_ACKNOWLEDGE 모드, 48

E

EJB 컨테이너, 82

H

HTTP 연결, 94

I

imqbrokerd 유틸리티, 67
imqcmd 유틸리티, 67
imqdbmgr 유틸리티, 68
imqkeytool 유틸리티, 68
imqobjmgr 유틸리티, 68
imqsvcadmin 유틸리티, 68
imqusermgr 유틸리티, 63, 68

J

J2EE 응용 프로그램
  EJB 사양, 81
  JMS 및, 22, 81
  Message-Driven Bean
    참조 Message-Driven Bean
  Message Queue 및, 31
J2EE 자원 어댑터, 95
Java 클라이언트, 29, 53
JDBC 지원
  관리, 68
  정보, 62
JMS
  Message Queue의 선택적 기능, 85
  공급자, 22
  도메인 및 API, 38
  런타임 지원, 29
  메시지 우선 순위, 표준, 43
  메시징 객체, 23
  메시징 패턴, 23
  사양, 22

JMS *(계속)*
예약 등록 정보, 86
JMS 응용 프로그램, 39
JMS 클라이언트, 53
JMSCorrelationID 메시지 헤더 필드, 42
JMSDeliveryMode 메시지 헤더 필드, 42, 43
JMSDeliveryMode 메시지 헤더 필드, 42
JMSDestination 메시지 헤더 필드, 42
JMSExpiration 메시지 헤더 필드, 43
JMSExpiration 메시지 헤더 필드, 42
JMSMessageID, 85
JMSMessageID 메시지 헤더 필드, 42
JMSPriority 메시지 헤더 필드, 43
JMSPriority 메시지 헤더 필드, 42
JMSRedelivered 메시지 헤더 필드, 42
JMSReplyTo 메시지 헤더 필드, 44
JMSReplyTo 메시지 헤더 필드, 42
JMSTimestamp 메시지 헤더 필드, 42
JMSType 메시지 헤더 필드, 42
JNDI 지원, 95

L
LDAP 서버 지원, 95
LDAP 저장소, 63

M
MapMessage 유형, 44
MDB, **참조** message-driven beans
MDB 컨테이너, 82
Message-Driven Bean
Application Server 지원, 83-84
MDB 컨테이너, 82
배포 설명자, 82
정보, 82
Message Queue
Application Server 및, 83-84
JMS 선택적 기능, 85
개발 환경, 68
기능 개요, 87
프로덕션 환경, 69
Message 유형, 44

O
ObjectMessage 유형, 44

S
SOAP 메시지, 52
SOAP 지원, 29, 101
SSL
기능 설명, 100
자체 서명된 인증서, 68
정보, 64
SSL(Secure Socket Layer) 표준, **참조** SSL
StreamMessage 유형, 44

T
TextMessage 유형, 44
TLS 프로토콜, 63

X
XA 자원 관리자, **참조** 분산 트랜잭션

가
가입자
소개, 36
영구, 37, 46, 50

객
객체 요청 브로커, 18

게
게시, 36
게시/가입 메시징, 36

관
관리 대상 객체
  관리, 68
  사용, 25
  소개, 24
관리 도구, 30
관리 작성 대상, 59

구
구성 요소
  EJB, 81
  MDB, 82

권
권한
  Message Queue 작업, 64
  액세스 제어 등록 정보 파일, 64
권한 부여
  참조 액세스 제어 파일
  사용 및 참조, 89
  정보, 63-64
  필요한 구성 요소, 62

기
기본 제공 지속성, 61

대
대기열, 41
대기열 브라우저, 36, 41
대상
  관리, 59
  구성, 60
  만들기, 41
  임시, 41, 46
  제한, 60
  종류, 59
대상 관리 객체
  JMS 프로그래밍 객체, 45
  정의, 24

데
데이터 저장소, 50
  JDBC 액세스 가능, 62
  정보, 61
  플랫 파일, 61

라
라우팅 서비스, 56

로
로거
  정보, 65
  출력 채널, 65
로깅, **참조** 로거

마
마스터 브로커, 78, 79

메
메모리 관리, 60, 97
메시지
  JMS, 42
  JMS 우선 순위, 43
  `JMSReplyTo` 헤더 필드, 46
  SOAP, 52
  게시, 36
  대상, 42
  만료, 42
  본문, 44
  본문 유형, 44
  브로드캐스팅, 38

메시지 *(계속)*
생성 및 소비, 40
선택, 42, 45
소비, 45
소비 로드 균형 조정, 36
수신기, 45
아이디, 42
안정적 전달, 48
안정적인 전달, 48
압축, 44, 91, 97
우선 순위, 42, 43
재전송 플래그, 42
저장소, 50
전달 모드, 42
제어, 50
지속성, 42
처리, 51
타임스탬프, 42
통신, 설정, 42
페이로드, 50
헤더
참조 메시지 헤더 필드
회신 대상, 42
메시지 서비스
관리, 30
구성 요소, 55
메모리 관리, 97
소개, 26
확장, 30
메시지 소비자, **참조** 소비자
메시지 수신기, **참조** 수신기
메시지 제작자, **참조** 제작자
메시지 지향 미들웨어, 17, 18
메시지 헤더 필드
JMS 메시지, 42-43
대체, 41, 85
메시징 공급자, 20
메시징 도메인
API 및, 38
게시/가입, 36
소개, 34
지점간, 34
메트릭
데이터
참조 브로커 메트릭
메시지, 65-66
메시지 제작자, 65-66
보고서, 65

모
모니터링 API, 98
모니터링 서비스, 57

물
물리적 대상
관리, 59, 67
구성, 60
만들기, 41
임시, 41, 46
제한, 60
종류, 59

미
미들웨어, 17, 18

방
방화벽, 57, 58

보
보안, 62, 100
보안 서비스, 57

분
분산 트랜잭션
참조 XA 연결 팩토리

분산 트랜잭션 *(**계속**)*
JMS 요구 사항 및, 86
XA 자원 관리자, 49
정보, 49

브
브로커
GUI 기반 관리, 68
windows 서비스, 68
개발 환경, 68
관리, 68
관리 도구, 66
다시 시작, 61
등록 정보, 57
로깅
참조 로거
마스터 브로커, 78, 79
메모리 관리, 60, 61
메트릭
참조 브로커 메트릭
모니터링, 94
모니터링 API, 98
방화벽, 연결, 57
상호 연결
참조 브로커 클러스터
서비스 사용, 57
성능, 조정, 102
소개, 28
시작, 67
연결, 28
오류 복구, 61
유지 관리, 69
자동 재연결, 40, 90
제한 동작, 60
프로덕션 환경, 69
브로커 클러스터
구성 변경 기록, 79
구조, 72
마스터 브로커, 78, 79
사용 및 참조, 90
정보 전파, 79
클러스터 구성 등록 정보, 78
클러스터 구성 파일, 78
브로커 확인
메시지 소비 및, 48
억제, 41

사
사용 불능 메시지 대기열
사용 및 참조, 93
정보, 59
사용자, 관리, 68
사용자 데이터, 63

선
선택기, 45

설
설계 및 성능, 34

성
성능, 102
성능 및 설계, 34

세
세션
JMS 클라이언트 확인, 48
JMS 프로그래밍 객체, 41
스레딩, 41
트랜잭션, 48

소
소비자
JMS 클라이언트, 23
JMS 프로그래밍 객체, 45
동기식, 45

소비자 *(계속)*
비동기식, 45
소비 로드 균형 조정, 36
영구, 41
전달, 45
한 대기열에서 다중, 99

수
수신기
JMS 프로그래밍 객체, 45
MDB 및, 82
수신자, 일련화, 41

스
스레드 관리, 101
스레딩 모델, 58

안
안정적인 전달
JMS 사양, 47-50
데이터 지속성, 99

암
암호화, 64

액
액세스 제어, 64
액세스 제어 파일, 62

연
연결 객체, 40
연결 서비스, 28
HTTP 지원, 94
연결 서비스 *(계속)*
관리, 67
구성, 57
메시지 흐름, 98
보안, 100
스레드 관리, 101
자동 재연결, 90
정보, 56
포트 매퍼
참조 포트 매퍼
핑 서비스, 93
연결 팩토리 관리 대상 객체
JMS 프로그래밍 객체, 40
정의, 24

영
영구 가입, 50
영구 데이터 저장소, 50

요
요청-응답 패턴, 46

인
인증
사용 및 참조, 89
정보, 63-64
필요한 구성 요소, 62

임
임시 대상, 46, 59

자
자동 작성 대상, 59
자원 어댑터, 31, 83, 95
자체 서명된 인증서, 68

전
전달, 안정적, **참조** 안정적인 전달
전달 모드, 42

제
제어 메시지, 50
제작자
  JMS 클라이언트, 23
  JMS 프로그래밍 객체, 44
  만들기, 44

주
주제, 41

지
지속성
  구성 가능, 92
  기본 제공, 61
  데이터, 99
  플러그인
    참조 플러그인 지속성
지속성 서비스, 56
지점간 메시징, 34

컨
컨테이너
  EJB, 82
  MDB, 82

클
클라이언트
  C 및 C++, 29, 53, 91
  Java, 29, 53
  런타임 지원, 29
클라이언트 인증, 40
클라이언트 확인, 48
클러스터 구성 등록 정보, 78
클러스터 구성 파일, 78

타
타임스탬프, 42

통
통합 API, 38

트
트랜잭션
  분산
    참조 분산 트랜잭션
  처리, 49

페
페이로드 메시지, 50

포
포트, 동적 할당, 58
포트 매퍼, 58

플
플러그인 지속성, 62