Sun Java™ System

Message Queue 3
기술 개요

2005Q1

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

부품 번호: 819-2222

목차

그림 목차

표 목차

머리말

이 문서 Sun Java™ System Message Queue 3 2005Q1 *기술 개요*는 Message Queue 메시징 서비스의 기술, 개념, 구조, 기능 및 특징을 소개합니다.

따라서 *Message Queue 기술 개요*는 Message Queue 설명서 세트 내에서 다른 문서의 기초를 제공합니다. Message Queue 설명서 세트에 있는 다른 문서를 읽기 전에 이 문서를 읽어야 합니다.

머리말은 다음과 같은 절로 구성되어 있습니다.

대상

이 안내서는 관리자, 응용 프로그램 개발자 및 Message Queue 제품을 사용하거나 제품의 기술, 개념, 구조, 기능 및 특징을 이해하려는 사용자를 위한 것입니다.

Message Queue 관리자는 Message Queue 메시징 시스템, 특히 이 시스템의 핵심인 Message Queue 메시지 서버를 설정하고 관리합니다. 이 문서는 메시징 시스템에 대한 지식이나 이해를 전제로 하지 않습니다.

응용 프로그램 개발자는 Message Queue 서비스를 사용하여 다른 클라이언트 응용 프로그램과 메시지를 교환하는 Message Queue 클라이언트 응용 프로그램을 작성하는 일을 담당합니다. 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. 다만, Message Queue 개발 및 관리 안내서를 읽기 전에 이 책을 읽고 기본적인 Message Queue 개념을 이해하십시오.

구성

이 안내서는 처음부터 끝까지 읽도록 되어 있으며 각 장은 이전 장에 수록된 내용을 기반으로 구성됩니다. 다음 표에서는 각 장의 내용을 간단히 설명합니다.

표 1 내용 및 구성

장	설명
1장, "개념적 기초"	엔터프라이즈 메시징 시스템을 설명하고 Java Message Service 개념 및 용어를 소개하여 Message Queue에 대한 개념적 배경 지식을 제공합니다.
2장, "Message Queue 소개"	Message Queue 서비스의 구조를 설명하고 엔터프라이즈급 특징 및 기능을 설명하여 이 서비스를 소개합니다.
3장, "안정적인 메시지 전달"	Message Queue 서비스가 메시징 응용 프로그램에 대해 안정적인 메시지 전달을 제공하는 방식을 설명합니다.
4장, "메시지 서버"	다양한 브로커 구성 요소 및 기능을 설명하여 브로커의 내부 구조를 설명합니다. 개발 및 작업 환경에서 Message Queue를 사용하는 여러 가지 접근방식을 설명합니다.
5장, "브로커 클러스터"	Message Queue 브로커 클러스터의 구조 및 내부 기능을 설명합니다.
6장, "Message Queue 및 J2EE"	J2EE 플랫폼 환경에서 JMS 지원을 구현한 결과를 살펴봅니다.
부록 A, "선택적 JMS 기능의 Message Queue 구현"	Message Queue 제품이 JMS 선택 항목을 처리하는 방법을 설명합니다.
용어집	Message Queue 사용 중에 접할 수 있는 용어와 개념에 대한 정보를 제공합니다.

본 설명서에 사용된 규칙

이 절에서는 본 설명서에 사용되는 표기 규칙에 관한 정보를 제공합니다.

활자체 규약

표 2 문서 표기 규칙

형식	설명
기울임꼴	기울임꼴 텍스트는 위치 표시자를 나타냅니다. 기울임꼴 텍스트로 표시된 부분은 적절한 절 또는 값으로 대체합니다. 기울임꼴 텍스트는 문서 제목, 강조 또는 소개할 단어나 구를 지정할 때도 사용됩니다.
고정 폭	고정 폭 텍스트는 코드의 예, 명령줄에 입력하는 명령, 디렉토리/파일/경로 이름, 오류 메시지 텍스트, 클래스 이름, 메소드 이름(서명의 모든 요소 포함), 패키지 이름, 예약어, URL을 나타냅니다.
[]	대괄호는 명령줄 구문에 선택적으로 사용되는 값을 나타냅니다.
모두 대문자	모두 대문자로 표시된 텍스트는 파일 시스템 유형(GIF, TXT, HTML 등), 환경 변수(IMQ_HOME) 또는 머리 글자(Message Queue, JSP)를 나타냅니다.
키+키	동시 키 입력은 더하기 기호와 함께 표시됩니다. Ctrl+A는 두 키를 동시에 누르라는 의미입니다.
키-키	연속적인 키 입력은 하이픈과 함께 표시됩니다. Esc-S는 Esc를 누르고 이를 놓은 다음 S키를 누르라는 의미입니다.

디렉토리 변수 규칙

Message Queue에서는 세 가지 디렉토리 변수를 사용하며 설정 방법은 플랫폼에 따라 다릅니다. 표 3에서는 이러한 변수를 설명하고 이들이 Solaris™, Windows, Linux 플랫폼에 사용되는 방법을 개괄적으로 설명합니다.

표 3 Message Queue 디렉토리 변수

변수	설명
IMQ_HOME	일반적으로 Message Queue 설명서에서 Message Queue 기본 디렉토리(루트 설치 디렉토리)를 참조할 때 사용됩니다. • Solaris에는 루트 Message Queue 설치 디렉토리가 없습니다. 따라서 Message Queue 설명서에서 Solaris의 파일 위치를 참조하는 경우에는 IMQ_HOME이 사용되지 않습니다. • Solaris에서 Sun Java System Application Server의 루트 Message Queue 설치 디렉토리는 Application Server 기본 디렉토리 아래에 있는 /imq입니다. • Windows의 경우, 루트 Message Queue 설치 디렉토리는 Message Queue 설치 프로그램에서 설정합니다(기본값은 C:\Program Files\Sun\MessageQueue3). • Windows의 경우 Sun Java System Application Server의 루트 Message Queue 설치 디렉토리는 Application Server 기본 디렉토리 아래의 /imq입니다. • Linux에는 루트 Message Queue 설치 디렉토리가 없습니다. 따라서 Message Queue 설명서에서 Linux의 파일 위치를 참조하는 경우에는 IMQ_HOME이 사용되지 않습니다.
IMQ_VARHOME	Message Queue 임시 또는 동적으로 작성된 구성 및 데이터 파일이 저장되는 /var 디렉토리입니다. 모든 디렉토리를 가리키는 환경 변수로 설정할 수 있습니다. • Solaris에서 IMQ_VARHOME의 기본값은 /var/imq 디렉토리입니다. • Solaris에서 Sun Java System Application Server 평가판의 IMQ_VARHOME 기본값은 IMQ_HOME/var 디렉토리입니다. • Windows에서 IMQ_VARHOME의 기본값은 IMQ_HOME\var 디렉토리입니다. • Windows에서 Sun Java System Application Server의 IMQ_VARHOME 기본값은 IMQ_HOME\var 디렉토리입니다. • Linux에서 IMQ_VARHOME의 기본값은 /var/opt/imq 디렉토리입니다.

표 3 Message Queue 디렉토리 변수*(계속)*

변수	설명
IMQ_JAVAHOME	Message Queue 실행 파일에 필요한 Java™ runtime (JRE)의 위치를 가리키는 환경 변수입니다. • Solaris에서 IMQ_JAVAHOME은 다음과 같은 순서로 Java 런타임을 찾지만, 선택적으로 사용자는 필요한 JRE가 있는 적절한 위치로 값을 설정할 수 있습니다. Solaris 8 또는 9: /usr/jdk/entsys-j2se /usr/jdk/jdk1.5.* /usr/jdk/j2sdk1.5.* /usr/j2se Solaris 10: /usr/jdk/entsys-j2se /usr/java /usr/j2se • Linux에서 Message Queue가 먼저 다음 순서로 Java 런타임을 찾지만, 선택적으로 사용자는 IMQ_JAVAHOME의 값을 필요한 JRE가 있는 적절한 위치로 설정할 수 있습니다. /usr/jdk/entsys-j2se /usr/java/jre1.5.* /usr/java/jdk1.5.* /usr/java/jre1.4.2* /usr/java/j2sdk1.4.2* • Windows에서 IMQ_JAVAHOME의 기본값은 IMQ_HOME\jre이지만, 선택적으로 사용자는 필요한 JRE가 있는 적절한 위치로 값을 설정할 수 있습니다.

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

관련 문서

Message Queue에는 이 설명서 외에도 추가 설명서 자원이 제공됩니다.

Message Queue 설명서 집합

Message Queue 설명서 집합을 구성하는 문서는 표 4에서 일반적으로 사용되는 순서에 따라 나열되어 있습니다.

표 4 Message Queue 설명서 집합

문서	대상	설명
Message Queue 설치 설명서	개발자와 관리자	Solaris, Linux, Windows 플랫폼에서 Message Queue 소프트웨어를 설치하는 방법을 설명합니다.
Message Queue 릴리스 노트	개발자와 관리자	새로운 기능, 제한, 알려진 버그 및 기술 노트에 관한 설명이 포함되어 있습니다.
Message Queue 관리 설명서	관리자, 개발자에게도 권장	Message Queue 관리 도구를 사용한 관리 작업 수행 시 필요한 배경 및 정보를 제공합니다.
Java 클라이언트용 Message Queue 개발 안내서	개발자	JMS 및 SOAP/JAXM 사양의 Message Queue 구현을 사용하는 Java 클라이언트 프로그램 개발자를 위한 빠른 시작 자습서와 프로그래밍 정보를 제공합니다.
C 클라이언트용 Message Queue 개발자 안내서	개발자	Message Queue 메시지 서비스에 대한 C 인터페이스(C-API)를 사용하는 C 클라이언트 프로그램 개발자를 위한 프로그래밍 및 참조 설명서를 제공합니다.

온라인 도움말

Message Queue는 Message Queue 메시지 서비스 관리 작업을 수행하는 명령줄 유틸리티를 포함합니다. 이 유틸리티에 대한 온라인 도움말을 보려면 *Message Queue 관리 설명서*를 참조하십시오.

또한 Message Queue에는 그래픽 사용자 인터페이스(GUI) 관리 도구인 관리 콘솔(`imqadmin`)이 포함되어 있습니다. 컨텍스트 관련 온라인 도움말은 관리 콘솔에 포함되어 있습니다.

JavaDoc

JavaDoc 형식의 Message Queue Java 클라이언트 API(JMS API 포함) 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 Netscape, Internet Explorer와 같은 모든 HTML 브라우저로 볼 수 있습니다. 표준 JMS API 설명서 및 Message Queue 관리 객체에 대한 Message Queue별 API를 포함하며(*Java 클라이언트용 Message Queue 개발 안내서*의 3장 참조), 이는 메시징 응용 프로그램 개발자에게 도움이 됩니다.

클라이언트 응용 프로그램의 예

샘플 클라이언트 응용 프로그램 코드를 제공하는 여러 응용 프로그램의 예가 운영 체제에 따라 다른 디렉토리에 포함되어 있습니다(*Message Queue 관리 설명서* 참조).

해당 디렉토리와 각 하위 디렉토리에 있는 README 파일을 참조하십시오.

JMS(Java Message Service) 사양

다음 위치에서 JMS 사양을 확인할 수 있습니다.

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

이 사양에는 클라이언트 코드 샘플이 포함되어 있습니다.

관련 타사 웹 사이트 참조

이 문서에 있는 타사 URL에서는 관련 추가 정보를 제공합니다.

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

사용자 의견 환영

Sun은 본 설명서의 개선을 위해 지속적으로 노력하고 있으며 고객의 의견과 제안을 환영합니다.

사용자 의견을 나누시려면 http://docs.sun.com으로 이동하여 의견 보내기를 누르십시오. 온라인 양식에서 문서 제목과 부품 번호를 입력하십시오. 부품 번호는 해당 설명서의 제목 페이지나 문서 맨 위에 있는 7자리 또는 9자리 숫자입니다.

사용자 의견을 제출할 때 해당 양식에 영문 설명서의 제목과 부품 번호를 입력해야 할 수도 있습니다. 본 설명서의 영문 제목과 부품 번호는 Sun Java System Message Queue 3 Technical Overview 2005Q1(819-0069)입니다.

1장

개념적 기초

Sun Java™ System Message Queue(Message Queue)는 기업 전체에 분산된 응용 프로그램 및 구성 요소를 통합할 수 있는 안정적인 *비동기식 메시징*을 제공합니다. 다양한 플랫폼 및 운영 체제에서 실행 중인 프로세스가 서비스에 연결하여 서로 상호 작용할 수 있습니다.

Message Queue는 JMS(Java™ Message Service) 개방형 표준을 구현하는 표준 기반 *메시징* 솔루션입니다. 또한 Message Queue는 대규모 엔터프라이즈 배포에 필요한 상호 운용성, 보안, 확장성, 가용성, 관리성 및 기타 기능을 제공합니다.

이 장에서는 Message Queue에 대한 개념적 기초를 제공하며 다음 내용으로 구성되어 있습니다.

JMS 개념 및 용어에 이미 익숙하다면 2장, "Message Queue 소개"로 건너뛸 수 있습니다.

엔터프라이즈 메시징 시스템

엔터프라이즈 메시징 시스템에서 독립 분산 응용 프로그램이나 응용 프로그램 구성 요소는 *메시지*를 통해 상호 작용할 수 있습니다. 동일한 호스트나 네트워크에 있거나 또는 인터넷을 통해 느슨하게 연결되어 있는 이 구성 요소들은 메시징을 사용하여 데이터를 전달하고 각자의 기능을 조정합니다.

많은 수의 구성 요소가 동시에 메시지를 교환하고 고용량의 처리량을 지원하기 위해서는 메시지 발신이 사용자의 즉시 수신 가능 여부에 따라 결정되어서는 안 됩니다. 메시지 사용자가 작업 중이거나 오프라인 상태인 경우, 시스템은 사용자가 온라인이 될 때 메시지 수신이 가능하도록 해야 합니다. 이러한 메시지 송수신의 분리를 비동기식 메시지 전달이라고 합니다.

비동기식 메시징 모델은 복잡한 시스템을 통합하는 작업에 매우 적합합니다. 이 모델은 작업 수행 과정에서 한 구성 요소가 다른 구성 요소를 방해할 수 없게 되어 있습니다. 비동기식 메시징은 동기식 시스템에서 가능한 일부 제어 기능을 포기하지만 구성 요소의 상호 작용에 상당한 유연성을 제공하며 한 구성 요소가 실패하더라도 전체 구성 요소의 실패로 연결되지 않으므로 견고성이 더해집니다.

엔터프라이즈 메시징 시스템의 요구 사항

일반적으로 엔터프라이즈 응용 프로그램 시스템은 24시간 중차대한 작업으로 무수히 많은 메시지를 교환하는 많은 수의 분산 구성 요소로 구성됩니다. 비동기식 메시징 지원을 비롯하여 이러한 시스템을 지원하려면 엔터프라이즈 메시징 시스템이 다음 요구 사항을 만족시켜야 합니다.

안정적인 전달. 구성 요소 간에 전달되는 메시지는 네트워크나 시스템 오류로 인해 손실되지 않아야 합니다. 즉 시스템에서 메시지 전달을 보장할 수 있어야 합니다.

보안. 메시징 시스템은 사용자 인증, 메시지 및 자원에 대한 인증된 액세스, 회선을 통한 암호화와 같은 기본 보안 기능을 지원해야 합니다.

확장성. 메시징 시스템은 성능이나 메시지 처리량이 실제로 저하되지 않으면서 로드 증가(사용자 수 및 메시지 수 증가)를 수용할 수 있어야 합니다. 업무와 응용 프로그램이 늘어나면 이는 매우 중요한 요구 사항이 됩니다.

가용성. 메시징 시스템은 거의 다운 타임 없이 작동해야 합니다. 즉 오류 발생 시 시스템이 메시징 서비스를 계속 제공하기에 충분한 중복을 포함해야 함을 의미합니다.

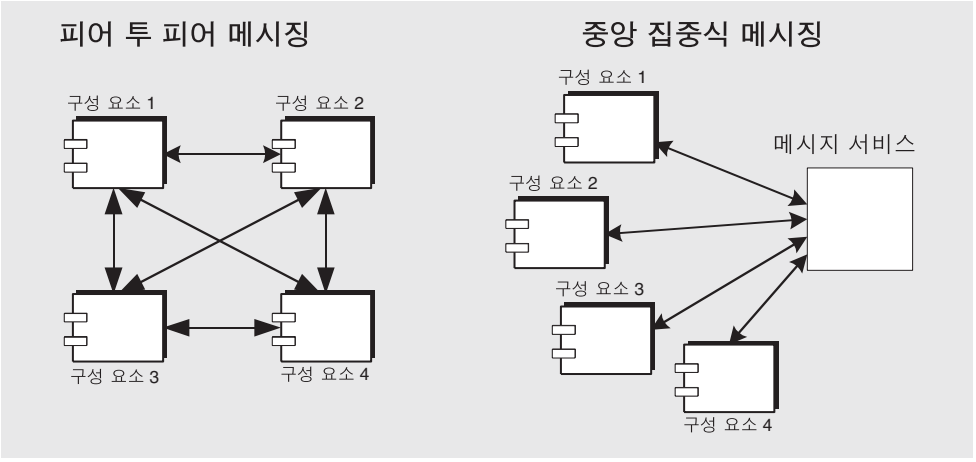
관리성. 메시징 시스템은 메시지 전달을 모니터하고 관리할 도구를 제공해야 합니다. 관리자는 시스템 자원을 최적화하고 시스템 성능을 조정할 수 있어야 합니다.

중앙 집중식(MOM) 메시징

메시지 대기열은 그림 1-1에 표시된 대로 중앙 집중식 메시징 시스템을 사용합니다. 이러한 시스템에서 각 메시징 구성 요소는 단일 중앙 메시지 서비스와의 연결을 유지 관리합니다. 구성 요소는 잘 정의된 인터페이스를 통해 메시지 서비스와 상호 작용합니다.

모든 메시징 구성 요소가 다른 모든 구성 요소와의 연결을 유지 관리하는 대체 피어 투 피어 시스템은 그림의 왼쪽에 있습니다. 피어 투 피어 시스템에서는 빠르고 안정적이며 보안이 유지된 상태로 전달을 할 수 있지만, 안정성과 보안을 지원하는 코드가 각 구성 요소마다 존재해야 합니다. 메시지의 송수신은 긴밀하게 결합되어 있으므로 비동기식 전달이 어려워집니다. 구성 요소가 시스템에 추가되면 연결 수가 기하학적으로 늘어나므로 시스템이 제대로 확장되지 않습니다. 또한 중앙 집중식 관리는 피어 투 피어 시스템에서 문제가 됩니다.

그림 1-1 중앙 집중식 메시징과 피어 투 피어 메시징 비교



엔터프라이즈 메시징의 권장되는 접근 방식인 중앙 집중식 시스템에서 메시지 서비스는 구성 요소 간에 메시지 경로 지정 및 전달을 제공하고 안정적인 전달 및 보안을 담당합니다. 이 시스템의 구성 요소는 느슨하게 결합되어 있으므로 비동기식 메시징을 달성하기 쉽습니다.

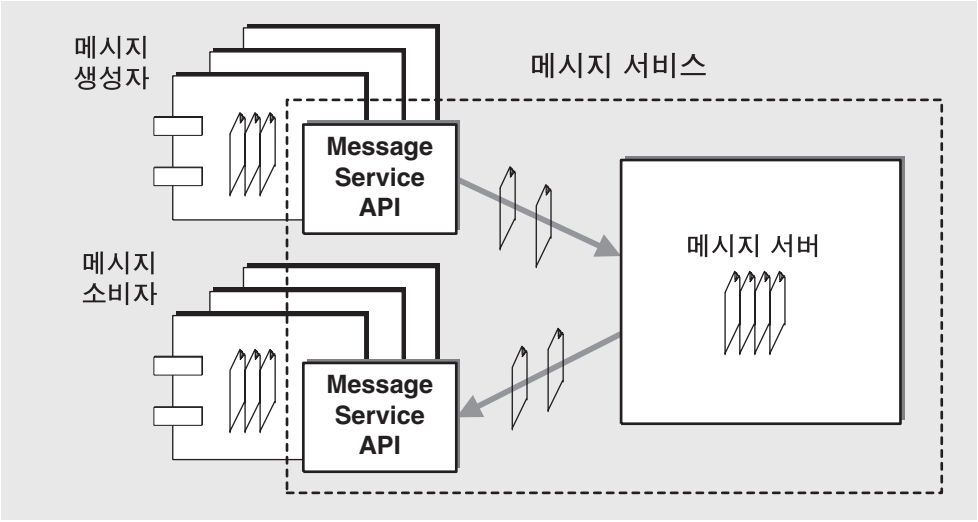
메시징 구성 요소가 시스템에 추가되면 선형적으로 연결 수가 증가되므로, 메시지 서비스를 확장하여 시스템을 보다 쉽게 확장할 수 있습니다. 중앙 메시지 서비스는 메시징 클라이언트를 연결할 뿐만 아니라 동작을 구성하고 성능을 모니터하며 서비스를 조정하여 각 메시징 클라이언트의 요구를 만족시킬 수 있는 관리 인터페이스를 제공합니다.

기본 메시지 서비스 구조

중앙 집중식 메시징 시스템의 기본 구조는 그림 1-2와 같으며 일반 *메시지 서비스*를 통해 메시지를 교환하는 메시지 *생성자*와 메시지 *사용자*로 구성됩니다. 메시지 생성자와 사용자는 그 수의 제한 없이 동일한 메시징 구성 요소(또는 응용 프로그램)에 위치할 수 있습니다.

메시지 생성자는 메시지 서비스 프로그래밍 API를 사용하여 메시지를 *메시지 서버*로 전송합니다. 메시지 서버는 메시지에 대해 인터레스트를 등록한 하나 이상의 메시지 사용자에게 메시지를 경로 지정하고 전달합니다. 사용자는 메시지 서비스 프로그래밍 API를 사용하여 메시지를 수신합니다. 메시지 서비스는 관련된 모든 사용자에게 메시지가 전달되도록 보장하는 역할을 합니다.

그림 1-2 메시지 서비스 구조



이 프로세스를 은유적으로 표현할 때 가장 적절한 표현은 편지 교환이라고 할 수 있습니다. 편지가 최종 수신자의 주소로 지정되어 있어도 편지는 우체국을 통해 발송되므로 수신인이 우편함을 통해 받아볼 때까지 여러 중간 위치에 머물게 됩니다.

JMS(Java Message Service) 기초

Message Queue는 JMS(Java Message Service) 개방형 표준을 구현하는 엔터프라이즈 메시징 시스템 즉, *JMS 공급자*입니다. 따라서 JMS 개념은 Message Queue 서비스의 작업 방식을 이해하는 데 기본이 됩니다.

JMS 사양은 안정적인 비동기식 메시징에 적용되는 일련의 규칙과 의미의 집합을 규정하며 메시지 구조, 프로그래밍 모델 및 API를 정의합니다.

이 절에서는 이 책의 나머지 장을 이해하는 데 필요한 JMS 개념 및 용어를 설명합니다. 이 절은 다음 내용으로 구성되어 있습니다.

JMS 메시지 구조

Message Queue에서 데이터는 JMS 메시지를 사용하여 교환됩니다. JMS 사양에 따라 생성자 *클라이언트*에 의해 작성된 메시지는 헤더, 등록 정보 및 본문 등 세 부분으로 구성됩니다.

헤더

헤더는 모든 JMS 메시지에서 필수입니다. 헤더 필드에는 메시지 경로 지정 및 식별에 사용되는 값이 포함됩니다.

헤더 값은 다음과 같은 여러 가지 방법으로 설정할 수 있습니다.

- 메시지 생성 또는 전달 프로세스 중 JMS 공급자에 의해 자동으로
- 메시지 생성자를 작성할 때 지정된 설정을 통해 생성자 클라이언트에 의해
- 메시지 단위의 메시지에서 생성자 클라이언트에 의해

JMS에 의해 정의된 헤더 필드에 대한 자세한 내용은 *Message Queue Developer's Guide for Java Clients* 또는 *Message Queue Developer's Guide for C Clients*를 참조하십시오. 이러한 헤더 필드를 사용하여 메시지의 대상, 만료 시간, 메시지 우선 순위 등을 정의할 수 있습니다.

등록 정보

메시지는 *등록 정보*라는 선택적 헤더 필드를 포함할 수 있습니다. 등록 정보는 등록 정보 이름과 등록 정보 값 쌍으로 지정됩니다. 메시지 헤더의 확장으로 생각할 수 있는 등록 정보는 데이터를 작성한 프로세스에 대한 정보, 데이터가 작성된 시간 및 데이터 각 부분의 구조를 포함할 수 있습니다. 또한 JMS 공급자는 메시지의 압축 여부 또는 수명이 다했을 때 메시지 처리 방법 등 메시지 처리에 영향을 미치는 등록 정보를 추가할 수 있습니다.

JMS 공급자는 메시지 등록 정보를 *선택기*로 사용하여 메시지를 정렬하고 경로 지정할 수 있습니다. 생성자 클라이언트는 메시지에 응용 프로그램별 등록 정보를 포함시킬 수 있으며 사용자 클라이언트는 등록 정보가 특정 값을 갖는 메시지만 수신하도록 선택할 수 있습니다. 예를 들어, 사용자 클라이언트는 뉴저지에 있는 시간제 직원에 대한 급여 메시지에 대해서만 관심 분야를 표시할 수 있습니다. 지정된 선택 기준을 만족하지 않는 메시지는 클라이언트로 전달되지 않습니다.

선택기는 사용자 클라이언트 작업을 간소화하고 해당 메시지가 필요하지 않은 클라이언트로 메시지를 전달하는 오버헤드를 없앱니다. 그러나 선택 기준을 처리해야 하는 메시지 서비스에 일부 오버헤드가 추가됩니다. 메시지 선택기 구문과 의미는 JMS 사양에 설명되어 있습니다.

메시지 본문 유형

JMS 메시지의 유형은 표 1-1에 지정된 대로 본문의 내용을 결정합니다.

표 1-1 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. 예를 들어, XML 메시지
ObjectMessage	본문에 일련화된 Java 객체를 포함하는 메시지
BytesMessage	본문에 해석되지 않은 바이트의 스트림이 포함된 메시지

JMS 프로그래밍 모델

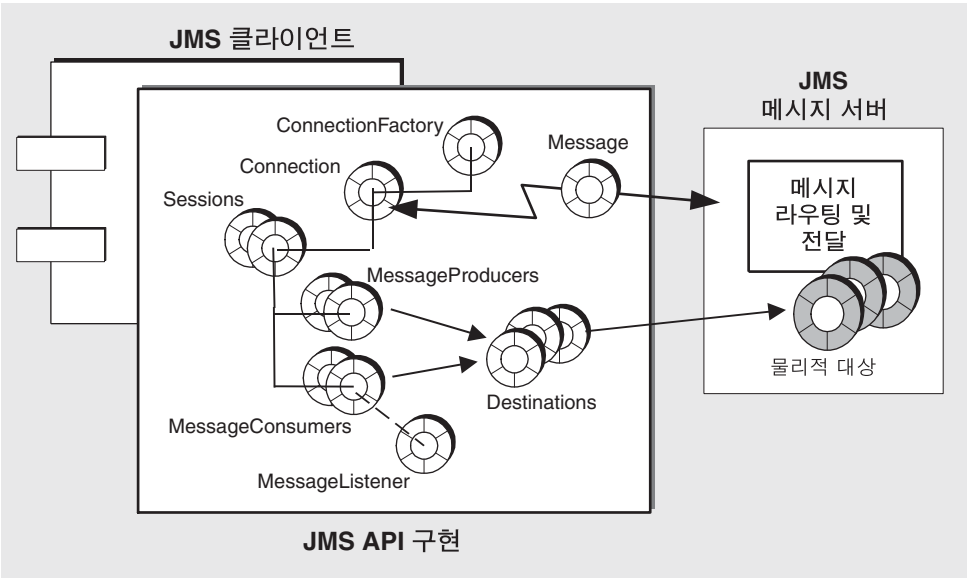
JMS 프로그래밍 모델은 비동기식 메시징 서비스의 구조를 지원합니다. JMS 클라이언트는 JMS 메시지 서비스를 통해 메시지를 교환합니다. JMS 공급자는 JMS 메시징을 수행하는 데 필요한 객체를 준비합니다. 이러한 객체는 JMS API(Application Programming Interface)를 구현합니다.

이 절에서는 JMS 메시징에 필요한 프로그래밍 객체를 설명하고 메시지를 송수신하는 데 사용되는 전달 모델(지점간 및 게시/가입)을 소개합니다.

프로그래밍 객체

메시지 전달을 위해 JMS 클라이언트 설정에 사용되는 객체는 그림 1-3에 나와 있습니다.

그림 1-3 JMS 프로그래밍 객체



JMS 프로그래밍 모델에서 JMS 클라이언트는 *연결 팩토리*객체(ConnectionFactory)를 사용하여 JMS 메시지 서버와 메시지를 송수신할 *연결*을 만듭니다. 연결 객체(Connection)는 클라이언트와 메시지 서버 간의 활성 연결을 나타냅니다.

연결되면 통신 자원 할당 및 클라이언트 인증이 이루어집니다. 이는 비교적 중량급 객체이며 대부분의 클라이언트는 단일 연결을 사용하여 모든 메시징을 수행합니다.

연결은 *세션* 객체(Session)를 생성할 때 사용합니다. 세션은 메시지 생성 및 사용을 위한 단일 스레드 컨텍스트입니다. 메시지 생성을 비롯하여 메시지를 보내고 받는 메시지 생성자 및 사용자를 생성할 때 사용하며, 전달할 메시지의 일련 순서를 정의합니다. 세션은 여러 *관리 대상 객체* 옵션이나 트랜잭션을 통해 안정적인 전달을 지원합니다.

클라이언트는 메시지 생성자 객체(MessageProducer)를 사용하여 API에서 대상 객체로 표시되는 지정된 물리적 *대상*으로 메시지를 보냅니다. 메시지 생성자는 생성자가 물리적 대상으로 보낼 모든 메시지에 적용되는 전달 모드(지속성 메시지 및 비지속성 메시지), 우선 순위 및 수명 값과 같은 기본 메시지 헤더 값을 지정할 수 있습니다.

그와 비슷하게 클라이언트는 메시지 사용자 객체(MessageConsumer)를 사용하여 API에서 대상 객체로 표시되는 지정된 물리적 대상으로부터 메시지를 받습니다. 메시지 전달 모델에 따라 *Queue*와 *Topic*의 2가지 대상 유형이 있습니다.

메시지 사용자는 메시지 선택기를 사용하여 메시지 서비스가 특정 선택 기준과 일치하는 등록 정보를 갖는 메시지만 전달하도록 할 수 있습니다.

메시지 사용자는 동기식 또는 비동기식 메시지 사용을 지원할 수 있습니다.

- 동기식 사용은 사용자가 메시지 전달을 명시적으로 요청한 다음 메시지를 사용해야 함을 의미합니다.
- 비동기식 사용은 메시지가 사용자용으로 등록된 메시지 수신기 객체(MessageListener)로 자동 전달됨을 의미합니다. 세션 스레드가 메시지 수신기 객체의 onMessage() 메소드를 호출하면 클라이언트가 한 메시지를 사용합니다.

프로그래밍 도메인: 메시지 전달 모델

JMS는 서로 다른 2가지 메시지 *전달 모델*인 지점간 모델과 게시/가입 모델을 지원합니다.

지점간(Queue 대상) 메시지는 생성자로부터 단일 사용자에게 전달됩니다. 이 전달 모델에서 대상 유형은 *대기열(Queue)*입니다. 메시지는 먼저 대기열 대상으로 전달된 다음, 대기열로부터 해당 대기열에 등록된 사용자 중 하나에게 한 번에 하나씩 전달됩니다. 생성자 수의 제한 없이 생성자는 대기열 대상에게 메시지를 보낼 수 있으며, 각 메시지는 반드시 한 사용자에게만 전달되어 성공적으로 사용됩니다. 대기열 대상에 등록된 사용자가 없는 경우, 대기열은 받은 메시지를 보관했다가 사용자가 대기열에 등록하면 메시지를 전달합니다.

게시/가입(Topic 대상) 단일 생성자가 사용자 수의 제한 없이 메시지를 전달합니다. 이 전달 모델에서 대상 유형은 *주제(Topic)*입니다. 메시지는 먼저 주제 대상으로 전달된 다음, 해당 주제에 *가입*한 *모든* 활성 사용자에게 전달됩니다. 생성자 수의 제한 없이 생성자는 특정 주제 대상으로 메시지를 보낼 수 있으며, 각 메시지는 가입한 사용자 수의 제한 없이 전달될 수 있습니다.

또한 주제 대상은 *영구 가입*을 지원합니다. 영구 가입은 주제 대상에 등록되었지만 메시지가 대상에 도달하는 시점에 비활성될 수 있는 사용자를 의미합니다. 나중에 활성화된 사용자는 메시지를 수신합니다. 주제 대상에 대해 등록된 사용자가 없는 경우, 해당 주제는 영구 가입이 있는 비활성 사용자에 대해서만 메시지를 보관합니다.

이러한 2가지 메시지 전달 모델은 표 1-2에 표시된, 서로 조금씩 다른 의미를 갖는 다른 프로그래밍 *도메인*을 나타내는 3가지 API 객체 집합을 사용하여 처리됩니다.

표 1-2 JMS 프로그래밍 도메인 및 객체

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

* 프로그래밍 방식에 따라 특정 대상 유형을 지정해야 합니다.

통합 도메인은 JMS 버전 1.1에서 소개되었습니다. 1.02b 이전 사양을 준수해야 할 경우 도메인별 API를 사용할 수 있습니다. 또한 도메인별 API를 사용하면 대기열 대상에 대한 영구 가입자 작성 등의 특정 유형 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 선택하십시오.

Message Queue 제품에 포함된 예제 응용프로그램을 비롯한 Message Queue 설명서의 많은 코드 예제는 개별 프로그래밍 도메인을 사용합니다.

안정적인 메시징

메시지의 *전달 모드*는 지속성 또는 비지속성으로 설정될 수 있으며 이 모드에서 메시지 전달의 안정성을 관리합니다.

- *지속성 메시지*는 단 한 차례 전달 및 성공적인 사용이 보장됩니다. 지속성 메시지는 메시지 서비스 오류 발생 시에도 손실되지 않습니다. 이 메시지에서는 안정성이 중요합니다.
- *비지속성 메시지*는 최대 한 차례 전달이 보장됩니다. 비지속성 메시지는 메시지 서비스 오류 발생 시 손실됩니다. 이러한 메시지의 경우 안정성은 중요한 사항이 아닙니다.

지속성 메시지의 경우 안정성 보장에는 2가지 측면이 있습니다. 하나는 확인과 트랜잭션을 사용하여 메시지 생성 및 사용이 성공적으로 이루어지게 하는 것입니다. 다른 하나는 지속성 저장소에 메시지를 보관하여 메시지를 사용자에게 전달할 때까지 메시지 서비스에서 지속성 메시지가 손실되지 않게 하는 것입니다.

다음 절에서는 안정성을 보장하는 이러한 2가지 측면에 대해 설명합니다.

확인/트랜잭션

안정적인 메시징은 메시지 생성자에서 메시지 서버의 물리적 대상으로, 그리고 해당 물리적 대상에서 메시지 사용자에게로 지속성 메시지가 성공적으로 전달될 수 있는지 여부에 따라 결정됩니다. 이러한 안정성은 JMS 세션이 지원하는 2가지 일반 메커니즘인 *관리 대상 객체* 또는 *트랜잭션* 중 하나를 사용하여 실현할 수 있습니다. 트랜잭션의 경우, 로컬 트랜잭션 또는 분산(분산 트랜잭션 관리자의 제어 하에서) 트랜잭션이 될 수 있습니다.

확인

확인은 안정적인 전달을 위해 클라이언트와 메시지 서비스 간에 전송되는 메시지입니다.

메시지 생성 시, 메시지 서비스는 메시지 전달을 수신하고 메시지를 대상에 넣고 영구적으로 저장했음을 확인합니다. 생성자의 `send()` 메소드는 확인이 반환할 때까지 차단됩니다.

메시지 사용 시, 클라이언트가 대상으로부터의 메시지 전달을 수신하고 메시지를 사용했음을 확인한 다음 메시지 서비스가 해당 대상에서 메시지를 삭제합니다. JMS는 다양한 안정성 정도를 나타내는 다양한 확인 모드를 지정합니다. 이러한 모드 중 일부에서는 메시지 서버가 메시지를 삭제하여 메시지를 다시 전달할 수 없음을 확인할 때까지 클라이언트는 메시지 서버에 대한 대기를 차단합니다.

로컬 트랜잭션

세션을 *트랜잭션된* 것으로 구성할 수 있는데, 이 경우 하나 이상의 메시지 생성 및/또는 사용을 *트랜잭션*이라는 기본 단위로 분류할 수 있습니다. JMS API는 트랜잭션을 시작, 완결 또는 롤백하는 메소드를 제공합니다.

트랜잭션 내부에서 메시지를 생성하거나 사용하면 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 작업이 모두 완료됩니다. 트랜잭션이 롤백되면 성공적인 작업이 모두 취소됩니다.

로컬 트랜잭션의 범위는 항상 단일 세션입니다. 즉 단일 세션 컨텍스트에서 수행되는 하나 이상의 생성자 또는 사용자 작업을 묶어 단일 로컬 트랜잭션으로 분류할 수 있습니다.

트랜잭션 범위가 단일 세션에 국한되므로 메시지 생성과 사용을 모두 총괄하는 종단간 트랜잭션은 만들 수 없습니다. 즉, 대상으로 메시지를 전달하는 것과 나중에 메시지를 클라이언트로 전달하는 것을 같은 트랜잭션으로 분류할 수 없습니다.

분산 트랜잭션

또한 JMS 사양은 *분산* 트랜잭션을 지원합니다. 즉 메시지 생성 및 사용은 데이터베이스 시스템과 같은 다른 자원 관리자가 관련된 작업들을 포함하는 더 크고 분산된 트랜잭션의 일부가 될 수 있습니다. 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 브로커는 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.

영구 저장소

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

또한 메시지 서버는 영구 가입도 지속적으로 저장해야 합니다. 그렇지 않으면, 오류 발생 시 메시지 서버는 메시지가 주제 대상에 도달한 다음에 활성화되는 영구 가입자에게 메시지를 전달할 수 없습니다.

메시지 전달을 보장하려는 메시징 응용프로그램은 메시지가 지속성을 갖도록 지정하고 이들을 영구 가입을 갖는 주제 대상이나 대기열 대상 중 하나로 전달해야 합니다.

JMS 관리 대상 객체

JMS 프로그래밍 모델에서 사용된 2가지 객체인 연결 팩토리와 대상은 공급자의 JMS 사양 구현에 따라 달라질 수 있습니다.

- *연결 팩토리 객체*는 공급자가 메시지 전달 시 사용하는 프로토콜 및 메커니즘에 따라 동작이 달라지는 연결을 작성하는 데 사용됩니다.
- *대상 객체*는 브로커의 물리적 대상의 이름을 지정하는 데 사용되는데 메시지 서버에 있는 물리적 대상의 특정 이름 지정 규약 및 기능에 따라 다릅니다.

클라이언트 이식을 허용하면서 이러한 객체 정의에 있어서 공급자에게 최대한의 유연성을 제공하기 위해 JMS 사양은 공급자별 정보를 캡슐화하는 연결 팩토리 및 대상에 대한 *관리 대상 객체*를 정의합니다. 이러한 객체는 관리자가 작성 및 구성하고 JNDI 이름 영역(객체 저장소)에서 저장하며 클라이언트가 표준 JNDI 조회 코드를 통해 액세스합니다.

관리 대상 객체를 사용하면 JMS 클라이언트는 공급자별 객체를 조회하고 참조할 때 논리적 이름을 사용할 수 있습니다. 그렇게 되면 클라이언트 코드는 공급자가 사용하는 특정 이름 또는 주소 지정 구문이나 구성 가능한 등록 정보를 알아 둘 필요가 없습니다. 따라서 코드 공급자 독립성을 갖게 됩니다.

42페이지의 "관리 대상 객체" 절에서는 메시지 대기열에서 사용된 관리 대상 객체에 대한 추가 정보를 제공합니다.

주 JMS 사양에서는 JNDI 조회를 사용하여 관리 대상 객체에 액세스하지 않아도 됩니다. 클라이언트 코드는 연결 팩토리 및 대상 객체를 인스턴스화하고 해당 속성에 대한 값을 설정합니다. 하지만 클라이언트 코드를 다른 공급자에게 이식할 수 없음을 의미합니다.

2장

Message Queue 소개

Message Queue는 JMS 1.1 사양을 준수하는 안정적인 비동기 메시징 서비스입니다. 또한 대규모 엔터프라이즈 배포 요구를 해결하기 위해 Message Queue는 JMS 사양 요구 사항을 능가하는 많은 기능을 제공합니다.

이 장에서는 Message Queue 서비스 구조를 설명하고 엔터프라이즈 특성 및 기능을 소개합니다. 이 장은 다음 내용으로 구성되어 있습니다.

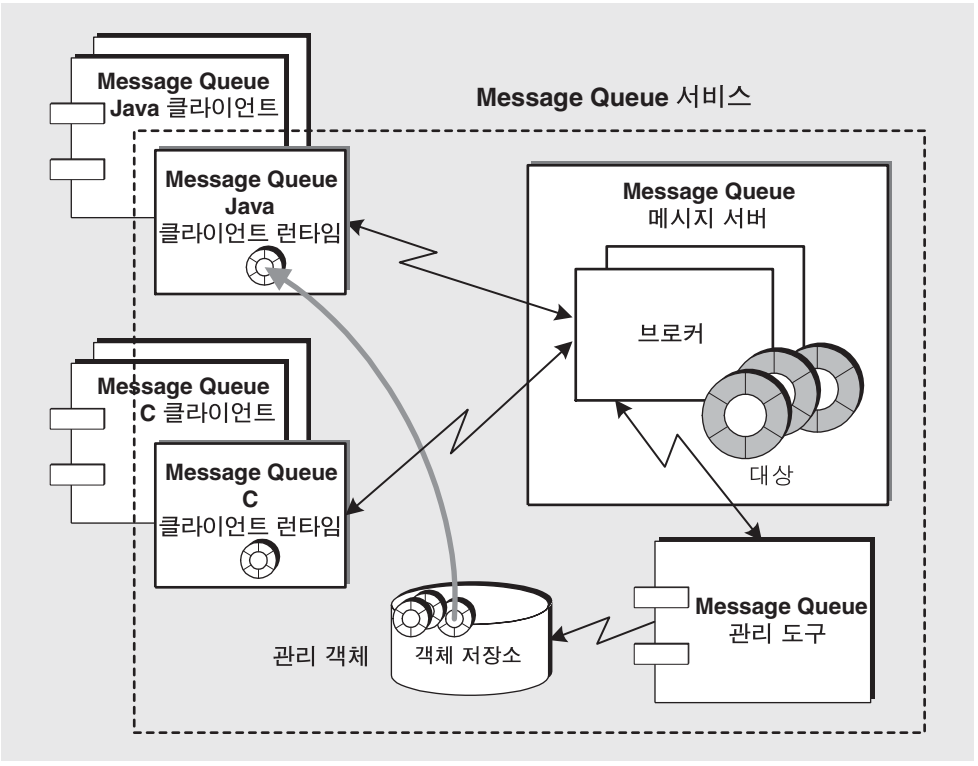
메시지 서비스 구조

Message Queue 서비스는 다음 요소로 구성되어 있습니다.

- 37페이지의 "메시지 서버"
- 37페이지의 "클라이언트 런타임"
- 42페이지의 "관리 대상 객체"
- 45페이지의 "관리 도구"

그림 2-1은 이러한 요소들이 함께 작동하는 방식을 보여줍니다.

그림 2-1 Message Queue 서비스 구조



그림과 같이 Message Queue 클라이언트는 Java 또는 C API를 사용하여 메시지를 주고받습니다. 이러한 API는 브로커와의 연결을 생성하고 요청된 연결 서비스에 적절하게 비트를 패키지화하는 실제 작업을 하는 Java 또는 C 클라이언트 런타임 라이브러리에 구현됩니다. 응용 프로그램이 관리 대상 객체를 사용하는 경우 클라이언트 런타임은 객체 저

장소에서 이러한 객체를 찾아서 사용하여 연결을 구성하고 물리적 대상을 찾습니다. 브로커는 메시지 경로를 지정하고 전달합니다. 관리자는 Message Queue 관리 도구를 사용하여 브로커를 관리하고 관리 대상 객체를 객체 저장소에 추가합니다.

다음 절에서 이러한 요소 각각에 대해 간단히 설명합니다.

메시지 서버

메시지 서버는 하나 이상의 브로커로 구성되어 메시지 경로 지정 및 전달을 수행하는 Message Queue 서비스의 핵심입니다.

메시지 서버는 단일 *브로커* 또는 경로 지정 및 전달 서비스를 수행하기 위해 브로커 *클러스터*로서 함께 작업하는 브로커 집합으로 구성됩니다. 브로커는 다음 작업을 수행하는 과정입니다.

- 사용자의 *인증* 및 수행할 작업의 *권한 부여*
- 클라이언트와의 통신 채널 설정
- 생성자 클라이언트로부터 메시지를 수신하여 각 물리적 대상에 저장
- 하나 이상의 사용자 클라이언트로 메시지 경로 지정 및 전달
- 안정적인 전달 보장
- 시스템 성능을 모니터하기 위한 데이터 제공

메시지 서버, 해당 내부 구성 요소 및 수행하는 기능에 대한 자세한 내용은 4장, "메시지 서버"(71페이지)를 참조하십시오.

Message Queue 엔터프라이즈판은 상호 연결된 다중 브로커 인스턴스로 구성되어, 메시지 서버가 메시지 트래픽 볼륨에 따라 작업 크기를 조절할 수 있는 브로커 클러스터의 사용을 지원합니다. 구조 및 클러스터 구성 문제에 대한 자세한 내용은 5장, "브로커 클러스터"를 참조하십시오.

클라이언트 런타임

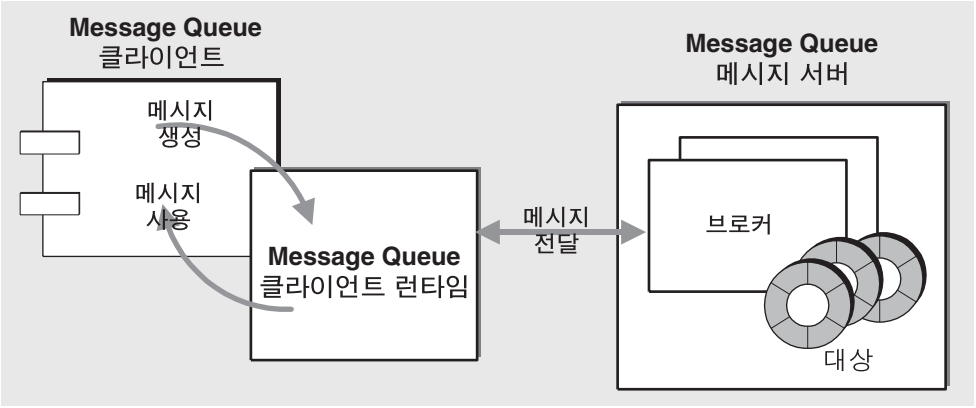
Message Queue *클러스터*는 클라이언트 응용 프로그램에 Message Queue 서비스에 대한 인터페이스를 제공합니다. 클라이언트 런타임은 Message Queue 클라이언트가 메시지를 생성(대상으로 메시지 전송)하고 메시지를 사용(대상으로부터 메시지 검색)하는 데 필요한 모든 작업을 지원합니다.

36페이지의 그림 2-1과 같이 Message Queue 클라이언트 런타임에는 두 가지 언어 구현이 있습니다 .

- **Java 클라이언트 런타임.** JMS API를 구현하고 Message Queue 메시지 서버와 상호 작용하는 데 필요한 모든 객체와 함께 Java 클라이언트 응용 프로그램 및 구성 요소를 제공합니다. 이러한 인터페이스 객체에는 연결, 세션, 메시지, 메시지 생성자 및 메시지 사용자가 포함됩니다.
- **C 클라이언트 런타임.** Message Queue 서버와 상호 작용하는 데 필요한 C 프로그래밍 인터페이스와 함께 C 클라이언트 응용 프로그램 및 구성 요소를 제공합니다. C 클라이언트 런타임은 JMS API 메시징 모델의 절차상 버전을 지원합니다.

그림 2-2는 Message Queue 클라이언트와 메시지 서버 간 클라이언트 런타임에서 수행하는 주요한 역할을 보여줍니다. 메시지 *전달*은 클라이언트 런타임과 메시지 서버 간의 상호 작용인데 반해, 메시지 *생성* 및 *사용*은 클라이언트와 클라이언트 런타임 간의 상호 작용과 관련됩니다.

그림 2-2 클라이언트 런타임 및 메시징 작업



클라이언트 런타임은 다음 기능을 수행합니다.

- 메시지 서버로 메시지 전달 관리
- 연결 설정
- 클라이언트의 아이디 설정
- 클라이언트 확인 구현
- 연결 상에서 메시지 흐름 제어
- 생성자 클라이언트가 설정한 메시지 헤더 값 대체 가능

다음 하위 절에서는 클라이언트 런타임 기능을 간단히 설명합니다. 클라이언트 런타임 동작의 일부 측면은 연결 팩토리 객체의 등록 정보를 구성하여 사용자 정의할 수 있습니다.

연결 처리

연결 처리 동작을 구성하려면 클라이언트가 연결하려는 브로커의 호스트 이름과 포트 및 원하는 연결 서비스 유형을 지정해야 합니다. 클러스터의 일부인 브로커와 연결된 경우 연결할 주소 목록을 지정해야 합니다. 한 브로커가 온라인이 아닌 경우 클라이언트 런타임은 사용자를 클러스터의 다른 브로커로 연결할 수 있습니다.

엔터프라이즈판에서 클라이언트 런타임은 연결이 실패할 경우 자동으로 브로커에 다시 연결할 수 있습니다. 재연결은 동일한 브로커에 대한 것일 수 있으며, 클라이언트가 클러스터의 일부인 브로커에 연결된 경우 원래 연결과 다른 브로커에 대한 것일 수 있습니다.

브로커 인스턴스가 공유되는 고가용성 영구 저장소(Sun Cluster와 Message Queue의 통합을 통해 사용 가능)를 사용하지 않는 경우, 다른 브로커 인스턴스로 재연결되면 실패한(또는 연결이 해제된) 브로커가 보유하는 지속성 메시지 및 기타 상태 정보가 손실될 수 있습니다. 즉, 재연결은 연결 페일오버를 제공하지만 데이터 가용성은 제공하지 않습니다.

클라이언트 아이디

응용 프로그램이 유용하다고 인식할 경우 클라이언트 아이디를 아무 연결에나 설정할 수 있는데 영구 가입자를 식별할 수 있도록 설정되어야 합니다.

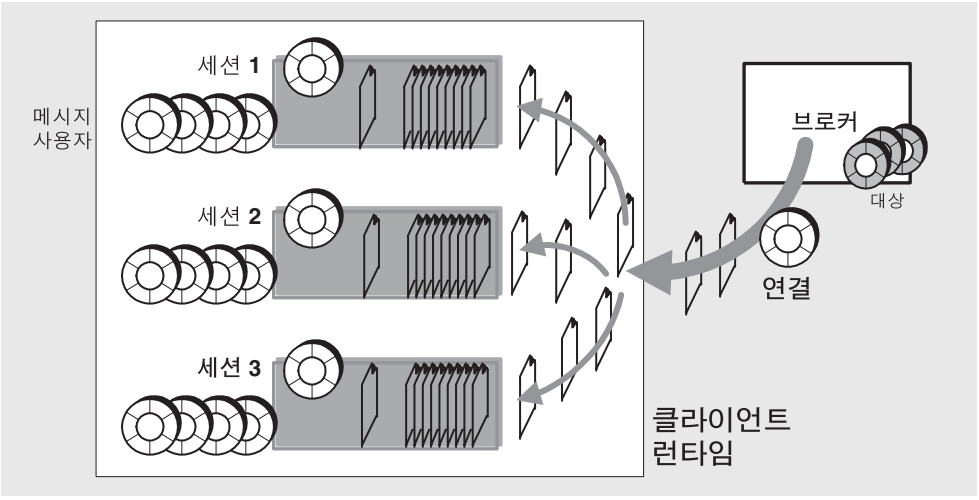
영구 가입을 추적하기 위해 브로커는 고유한 클라이언트 아이디를 사용합니다. 클라이언트 아이디는 메시지가 주제 대상으로 전달될 시점에 비활성인 영구 가입자를 식별하는 데 사용됩니다. 브로커는 이러한 가입자에게 전달된 메시지를 보관하였다가 가입자가 활성화되면 해당 메시지를 사용할 수 있게 합니다.

따라서 배포된 응용 프로그램에서 영구 가입을 사용할 때마다 클라이언트 식별자를 설정해야 합니다. Message Queue 기능을 사용하면 클라이언트 아이디를 지정할 때 특수 변수 이름 구문을 사용할 수 있으므로 해당 객체가 관리자에 의해 작성되었는지 아니면 프로그램 방식으로 작성되었는지 여부에 상관없이 연결 팩토리 객체에서 얻은 각 연결에 대해 다른 클라이언트 아이디를 얻을 수 있습니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

사용자에게 메시지 배포

연결을 통해 브로커가 전달한 메시지는 클라이언트 런타임이 수신하고 적절한 Message Queue 세션으로 분배됩니다. 이 경우 메시지는 40페이지의 그림 2-3에 표시된 대로 해당 메시지 사용자가 사용할 수 있도록 대기열에 쌓입니다.

그림 2-3 Message Queue 클라이언트 런타임으로의 메시지 전달



메시지는 각 세션 대기열에서 한 번에 하나씩 동기식(`receive()` 메소드를 호출하는 클라이언트 스레드를 통해) 또는 비동기식(메시지 수신기 객체의 `onMessage()` 메소드를 호출하는 세션 스레드를 통해)으로 사용됩니다(세션은 단일 스레드로 구성).

클라이언트 런타임으로 전달된 메시지의 흐름은 사용자 수준별로 측정됩니다. 연결 팩토리 등록 정보를 적절하게 조정하여 한 세션으로 전달된 메시지가 동일한 연결의 다른 세션으로의 메시지 전달에 부정적인 영향을 미치지 않도록 메시지의 흐름을 조절할 수 있습니다.

안정적인 메시지 전달

클라이언트 런타임은 안정적으로 메시지를 전달하는 데 중요한 역할을 합니다. JMS 사양의 클라이언트 확인 및 트랜잭션 모드를 지원하고 안정적인 전달을 보장하기 위해 사용된 다양한 브로커 확인 동작을 제어합니다.

JMS 사양은 여러 수준의 안정성을 위해 제공하는 많은 클라이언트 확인 모드를 설명합니다. 이러한 확인 모드와 Message Queue에 의해 구현되는 추가 모드는 메시지 사용 컨텍스트에서 설명됩니다(63페이지의 "클라이언트 확인" 참조).

지속성 메시지 및 안정적인 전달 시, 브로커는 일반적으로 단 한 번의 메시지 사용을 보장하기 위해 사용된 작업을 완료했을 때 클라이언트 런타임으로 확인을 보냅니다. 이러한 브로커 확인을 억제하는 연결 팩토리 등록 정보를 사용하면 네트워크 대역폭 및 처리를 줄일 수 있습니다. 하지만 이렇게 브로커 확인을 억제하면 안정적인 전달을 보장하지 못합니다.

메시지 흐름 제어

클라이언트 런타임은 연결에서 메시지 흐름의 게이트키퍼입니다. 연결에서 흐르는 일반 JMS 페이로드 메시지를 비롯하여 Message Queue는 안정적인 전달을 보장하고 연결에서의 메시지 흐름을 관리하며 다른 제어 기능을 수행하는 데 사용되는 다양한 제어 메시지를 보냅니다.

페이로드 메시지와 제어 메시지는 동일한 연결에 대해 경쟁하므로 충돌하여 정체를 일으킬 수 있습니다. 클라이언트 런타임은 구성 가능한 다양한 흐름 제한 및 측정 체계를 적용하여 페이로드 및 제어 메시지의 충돌을 최소화하고 그로 인해 메시지 처리량을 최대화합니다.

메시지 헤더 값 대체

클라이언트 런타임은 메시지의 지속성, 수명 및 우선 순위를 지정하는 JMS 메시지 헤더 필드를 대체할 수 있습니다.

Message Queue에서는 연결 수준에서 메시지 헤더를 대체할 수 있습니다. 지정된 연결 컨텍스트에서 생성된 모든 메시지에 대해 대체 내용을 적용합니다.

메시지 헤더 값을 대체하는 클라이언트 런타임의 기능은 Message Queue 관리자에게 메시지 서버의 자원에 대한 더 많은 제어 권한을 부여합니다. 하지만 이러한 필드를 대체할 경우 응용 프로그램별 요구 사항(예: 메시지 지속성)에 저촉될 위험이 있습니다. 따라서 이러한 기능은 담당 응용 프로그램 사용자 및 설계자와 상의해서 사용해야 합니다.

기타 기능

클라이언트 런타임은 몇 가지 다른 관련 기능을 수행합니다.

- **대기열 찾아보기 특성.** 대기열 대상의 내용을 찾아볼 때 한 번에 검색될 메시지 수와 메시지를 대기할 시간에 대하여 클라이언트 런타임을 구성할 수 있습니다.
- **메시지 압축.** Java 클라이언트 런타임은 메시지 생성 시 메시지를 압축하고 메시지 사용 시 메시지 압축을 풀 수 있습니다. 이러한 압축 또는 압축 풀기의 발생 여부는 클라이언트가 메시지를 작성할 때 메시지 헤더에 설정한 Message Queue별 메시지 등록 정보에 따라 달라집니다.

관리 대상 객체

관리 대상 객체는 연결 및 대상에 대한 공급자별 구현 및 구성 정보를 캡슐화합니다. 관리 대상 객체는 프로그램 방식으로 작성되거나 관리 도구를 사용하여 작성 및 구성되고 객체 저장소에 저장되며 표준 JNDI 조회 코드를 통해 클라이언트 응용 프로그램에서 액세스할 수 있습니다.

Message Queue는 다음 테이블에 표시된 관리 대상 객체 유형을 제공합니다.

표 2-1 Message Queue 관리 대상 객체 유형

유형	설명
대상	브로커의 물리적 대상을 나타냅니다. 브로커에 있는 물리적 대상의 공급자별 이름을 포함합니다. 메시지 사용자 및/또는 메시지 생성자 객체는 대상 관리 객체를 사용하여 해당 물리적 대상에 액세스합니다.
연결 팩토리	클라이언트 응용 프로그램과 Message Queue 메시지 서버 간에 물리적 연결을 설정합니다. 또한 물리적 연결 동작을 제어하는 Message Queue 클라이언트 런타임을 구성합니다. 연결 팩토리 관리 대상 객체의 속성 값을 설정할 때 설정하는 모든 연결에 적용할 등록 정보를 지정합니다.
XA 연결 팩토리	분산 트랜잭션을 지원하는 물리적 연결을 설정하는 데 사용됩니다(32페이지의 "분산 트랜잭션" 참조). XA 연결 팩토리 객체는 일반 연결 팩토리 객체와 동일한 속성 집합을 공유하지만 분산 트랜잭션을 지원하는 데 필요한 추가 메커니즘을 활성화합니다.
SOAP 종점	SOAP 메시지의 최종 대상을 식별하며 SOAP 메시지를 수신할 수 있는 서블릿의 URL입니다. SOAP 종점 관리 대상 객체를 구성하여 다중 URL을 지정할 수 있습니다. 또한 객체와 연결된 조회 이름과 객체 저장소 속성을 지정합니다.

JNDI를 통해 관리 대상 객체 사용

JMS 사양에서는 JMS 클라이언트가 JNDI 이름 공간에서 관리 대상 객체를 조회할 것을 요구하지는 않지만, 그렇게 할 경우 뚜렷한 장점이 있습니다. 즉, 단일 제어 소스를 허용하고 다시 코드화할 필요 없이 연결(클라이언트 런타임 동작)을 구성 및 재구성할 수 있으며 다른 JMS 공급자에게 클라이언트를 이식할 수 있습니다.

관리 대상 객체를 사용하면 Message Queue 서비스를 매우 쉽게 제어하고 관리할 수 있습니다.

- 관리자는 클라이언트 응용 프로그램이 사전 구성된 연결 팩토리 객체에 액세스하도록 요구함으로써 클라이언트 런타임의 동작을 지정할 수 있습니다.
- 관리자는 클라이언트 응용 프로그램이 기존 물리적 대상과 일치하는 사전 구성된 대상 관리 객체에 액세스하도록 요구함으로써 물리적 대상의 증가를 제어할 수 있습니다.

즉, 관리 대상 객체를 사용하면 Message Queue 관리자는 메시지 서비스 구성 세부 정보를 제어하면서 동시에 클라이언트 응용 프로그램이 공급자 독립성을 갖게 할 수 있습니다.

관리 대상 객체를 사용한다는 것은 프로그래머가 공급자별 구문과 객체 이름 지정 규약이나 공급자별 구성 등록 정보에 대하여 알 필요가 없음을 의미합니다. 실제로 관리 대상 객체가 읽기 전용이 되게 지정함으로써 관리자는 관리 대상 객체가 처음 작성되었을 때 설정된 관리 대상 객체 속성 값을 클라이언트 응용 프로그램이 변경할 수 없게 할 수 있습니다.

클라이언트 응용 프로그램에서 연결 팩토리와 대상 관리 객체 모두를 자체적으로 인스턴스화할 수 있지만, 이는 관리 대상 객체의 기본 목표에 어긋납니다. Message Queue 관리자는 응용 프로그램에서 필요로 하는 브로커 자원을 제어하고 메시징 성능을 조정해야 합니다. 또한 관리 대상 객체를 직접 인스턴스화하면 클라이언트 응용 프로그램이 공급자에게 종속됩니다.

이러한 주장에도 불구하고 응용 프로그램은 종종 관리 제어가 문제가 되지 않는 개발 환경에서 관리 대상 객체를 인스턴스화합니다.

객체 저장소

Message Queue 관리 대상 객체는 JNDI 조회를 통해 클라이언트 응용 프로그램에서 액세스할 수 있는 객체 저장소(36페이지의 그림 2-1 참조)에 저장됩니다. Message Queue는 두 가지 유형의 객체 저장소, 즉, 표준 LDAP 디렉토리 서버와 파일 시스템 객체 저장소를 지원합니다.

LDAP 서버 객체 저장소 LDAP 서버는 작업 메시징 시스템에 권장되는 객체 저장소입니다. LDAP 구현은 여러 공급업체에서 제공하며 분산 시스템에서 사용할 수 있도록 디자인되어 있습니다. LDAP 서버는 작업 환경에 유용한 보안 기능도 제공합니다.

파일 시스템 객체 저장소 Message Queue는 파일 시스템 객체 저장소를 지원합니다. 이 저장소는 생산 시스템에는 권장되지 않지만 개발 환경에서는 매우 사용하기 쉽다는 장점이 있습니다. LDAP 서버를 설정할 필요 없이 로컬 파일 시스템에 디렉토리를 만들기만 하면 됩니다. 그러나 클라이언트가 여러 컴퓨터 노드에 배포된 경우 이 클라이언트들이 객체 저장소가 위치한 디렉토리에 액세스할 수 없으면 파일 시스템 저장소를 중앙 집중식 객체 저장소로 사용할 수 없습니다.

관리 도구

Message Queue 관리 도구는 일련의 명령줄 유틸리티와 그래픽 사용자 인터페이스(GUI) 관리 콘솔로 구성됩니다.

명령줄 유틸리티 Message Queue는 브로커 시작 및 관리, 물리적 대상 작성 및 관리, 관리 대상 객체 관리 그리고 더 전문화된 기타 관리 작업의 수행과 같은 모든 Message Queue 관리 작업을 수행하는 일련의 명령줄 유틸리티를 제공합니다. 모든 명령줄 유틸리티는 공통 형식, 구문 규약 및 옵션을 공유합니다. 명령줄 유틸리티 사용에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

관리 콘솔 이 콘솔은 Message Queue 명령줄 유틸리티 기능의 일부를 제공합니다. 관리 콘솔을 사용하여 브로커를 관리하고 물리적 대상을 작성 및 관리하며 관리 대상 객체를 관리할 수 있습니다. 하지만 일부 명령줄 유틸리티의 더욱 전문화된 작업은 수행할 수 없습니다. 예를 들어, 관리 콘솔을 사용하여 브로커를 시작하거나 브로커 클러스터를 작성하거나 사용자 저장소를 관리할 수 없습니다. 이러한 작업은 Message Queue 명령줄 유틸리티를 사용하여 수행해야 합니다.

*Message Queue 관리 설명서*에서 제공하는 간단한 실습 자습서를 통해 관리 콘솔 기능을 익히고 이 기능을 사용하여 기본적인 작업을 수행하는 방법을 학습할 수 있습니다.

관리 콘솔과 일부 명령줄 유틸리티를 사용하여 브로커 및 물리적 대상을 원격 관리할 수 있습니다.

제품 기능

Message Queue 서비스 및 이전 절에서 설명한 구조는 안정적인 비동기식의 유연한 메시지 전달을 위해 JMS 1.1 사양을 완전히 구현합니다. JMS 호환성 관련 문제에 대한 설명서를 보려면 *부록 A, "선택적 JMS 기능의 Message Queue 구현"*을 참조하십시오.

하지만 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 시스템과 통합할 수 있습니다.

아래에서 설명하는 Message Queue의 엔터프라이즈 기능은 다음과 같은 범주로 분류됩니다.

통합 지원 기능

Message Queue를 사용하면 여러 전송 프로토콜에 대한 지원, Message Queue 서비스에 대한 C 클라이언트 인터페이스, SOAP(XML) 메시지 지원 및 플러그 가능한 J2EE 자원 어댑터를 포함하므로 기업 전체에서 이종 응용 프로그램 및 구성 요소를 통합할 수 있습니다.

다중 전송 지원

Message Queue는 TCP와 HTTP 등의 다양한 전송 방식을 통해 그리고 보안 연결을 사용하여 클라이언트가 Message Queue 메시지 서버와 상호 작용하는 기능을 지원합니다.

HTTP 연결 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 HTTP에서 방화벽을 통해 터널 서블릿으로 전달됩니다. 터널 서블릿은 HTTP 요청으로부터 메시지를 추출하고 TCP/IP를

통해 메시지를 브로커로 전달합니다. 유사한 방식으로 Message Queue는 HTTPS 터널 서블릿을 사용하여 보안 HTTP 연결을 지원합니다. HTTP 연결 구조에 대한 자세한 내용은 76페이지의 "HTTP/HTTPS 지원"을 참조하십시오. HTTP/HTTPS 연결 설정 및 구성에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

보안 연결 Message Queue는 TCP/IP 및 HTTP 전송을 통해 SSL(Secure Socket Layer) 표준에 기반한 메시지를 안전하게 전송합니다. 이러한 SSL 기반 연결 서비스를 사용하면 클라이언트와 브로커 사이에서 보내는 메시지를 *암호화*할 수 있습니다.

SSL 지원은 자체 서명한 서버 인증서에 기반합니다. 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. 자체 서명한 인증서를 작성하여 SSL 기반 연결 서비스를 사용하는 방법에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

C 클라이언트 인터페이스

Message Queue는 Java 언어 메시징 클라이언트 지원뿐만 아니라 Message Queue 서비스에 대한 C 언어 인터페이스를 제공합니다. C API를 사용하여 기존 C 응용 프로그램 및 C++ 응용 프로그램은 JMS 기반 메시징에 참여할 수 있습니다. 하지만 Message Queue의 C API를 사용하는 클라이언트는 다른 JMS 공급자로 이식될 수 없습니다.

Message Queue의 C API는 관리 대상 객체와 맵, 스트림 또는 객체 메시지 본문 유형, 분산 트랜잭션 및 대기열 브라우저를 사용한다는 점을 제외하면 표준 JMS 기능의 대부분을 지원하는 C 클라이언트 런타임에서 지원됩니다. 또한 C 클라이언트 런타임은 대부분의 Message Queue 엔터프라이즈 기능을 지원하지 않습니다.

C API의 기능과 C API가 C 데이터 유형 및 함수를 사용하여 JMS 프로그래밍 모델을 구현하는 방식에 대한 자세한 내용은 *C 클라이언트용 Message Queue 개발 안내서*를 참조하십시오.

SOAP(XML) 메시징 지원

Message Queue는 SOAP(Simple Object Access Protocol) 사양을 준수하는 메시지 생성 및 전달을 지원합니다. SOAP을 사용하면 분산 환경에서 피어 간에 구조화된 XML 데이터 또는 SOAP 메시지를 교환할 수 있습니다. SOAP 메시지는 첨부 파일을 포함할 수도 있는 XML 문서입니다. 이 첨부 파일이 XML 형식일 필요는 없습니다.

SOAP 메시지가 XML로 코드화된다는 사실은 SOAP 메시지가 플랫폼 독립성을 갖게 합니다. 이러한 메시지를 사용하여 레거시 시스템에서 데이터에 액세스하고 엔터프라이즈 간에 데이터를 공유할 수 있습니다. 또한 XML이 제공하는 데이터 통합은 이 기술이 웹 서비스와 같은 웹 기반 컴퓨팅의 특성을 갖게 만듭니다. 방화벽은 SOAP 패킷을 인식하고 SOAP 메시지 헤더에 표시된 정보에 기반하여 메시지를 필터링할 수 있습니다.

Message Queue는 SAAJ(SOAP with Attachments API for Java) 사양을 구현합니다. SAAJ는 SOAP 메시징을 위한 프로그래밍 모델을 지원하고 SOAP 메시지를 구성, 전송, 수신 및 조사하기 위해 사용할 수 있는 Java 객체를 제공하기 위해 구현할 수 있는 응용 프로그래밍 인터페이스입니다. SAAJ는 다음 2가지 패키지를 정의합니다.

- `javax.xml.soap`: SOAP 메시지의 부분을 정의하고 SOAP 메시지를 어셈블 및 역어셈블하려면 이 패키지의 객체를 사용합니다. 또한 이 패키지를 사용하여 공급자의 지원 없이 SOAP 메시지를 전송할 수 있습니다.
- `javax.xml.messaging`: 공급자를 사용하여 SOAP 메시지를 보내고 SOAP 메시지를 받으려면 이 패키지의 객체를 사용합니다.

Message Queue는 SOAP 메시지를 JMS 메시지로 변환하고 *그 역으로도 변환하는* 유틸리티를 제공합니다. 이러한 유틸리티를 사용하면 SOAP 메시지를 서블릿에서 수신하여 JMS 메시지로 변환하면 Message Queue 서비스에서 JMS 사용자에게로 전달하고 다시 SOAP 메시지로 변환하여 SOAP 종점으로 전달할 수 있습니다. 즉, Message Queue는 SOAP 종점 간에 안정적으로, 비동기식으로 SOAP 메시지를 교환하는 기능, 더 간단하게는 SOAP 메시지를 Message Queue 가입자에게 게시하는 기능을 지원합니다.

자세한 내용은 *Java 클라이언트용 Message Queue 개발 안내서*를 참조하십시오.

J2EE 자원 어댑터

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 서로 상호 작용할 수 있도록 하는 것입니다. 즉, J2EE 플랫폼은 JMS 지원을 요구합니다.

이러한 지원은 JMS 메시지를 사용할 수 있는 EJB(Enterprise Java Bean) 구성 요소의 특수 유형인 MDB(Message-Driven Bean)를 사용하는 J2EE 프로그래밍 모델에서 제공됩니다. J2EE 호환 Application Server는 JMS 메시징을 지원하는 MDB 컨테이너를 제공해야 합니다. 이 작업은 Application Server에 JMS 자원 어댑터를 플러그인하면 가능합니다. Message Queue는 이러한 자원 어댑터를 제공합니다.

Message Queue 자원 어댑터를 Application Server에 플러그인하면 Application Server 환경에서 배포되고 실행 중인 MDB를 비롯한 J2EE 구성 요소는 내부 및 외부 JMS 구성 요소와 JMS 메시지를 교환할 수 있습니다. 따라서 분산 구성 요소에 강력한 통합 기능을 제공하게 됩니다.

Message Queue 자원 어댑터에 대한 내용은 6장, "Message Queue 및 J2EE"를 참조하십시오.

보안 기능

대부분의 엔터프라이즈 응용 프로그램에서 저장 및 전송 중인 메시지 데이터를 보호하는 것이 중요합니다. Message Queue는 사용자 인증, 자원에 대한 액세스 제어 및 메시지 암호화를 비롯하여 많은 수준에서 보안을 제공합니다.

인증 Message Queue는 사용자의 비밀번호 기반의 인증을 지원합니다. 플랫 파일 또는 LDAP 사용자 저장소에 저장된 비밀번호에 기반하여 메시지 서버에 대한 연결이 사용자에게 허가됩니다. 모든 연결 시도(사용자 및 호스트 컴퓨터)에 대한 정보가 기록되므로 추적할 수 있습니다.

권한 부여 액세스 제어 목록(ACL)을 사용하면 브로커 연결 및 물리적 대상에 대한 액세스에 대해 구성 가능하고 세밀한 제어가 가능합니다. 사용자 및 그룹 액세스가 모두 지원됩니다. 권한 부여는 브로커별로 수행되므로 각 브로커는 서로 다른 액세스 제어 파일을 가질 수 있습니다.

암호화 SSL 지원을 사용하면 메시지 서버와 해당 클라이언트 간의 모든 메시지 트래픽(TCP/IP 또는 HTTP 연결을 통해서든)을 완전한 SSL 구현을 사용하여 암호화할 수 있습니다.

사용자 저장소 채우기, 액세스 제어 목록 관리 및 SSL 지원 설정에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

확장성 기능

Message Queue를 사용하면 사용자, 클라이언트 연결 및 메시지 로드가 증가함에 따라 응용 프로그램의 크기를 조정할 수 있습니다.

확장 가능한 연결 용량

Message Queue 브로커는 수 천 개의 연결을 동시에 처리할 수 있습니다. 기본적으로 각 연결은 전용 브로커 스레드에 의해 처리됩니다. 따라서 연결이 유휴 상태일 때도 해당 스레드를 독점하므로 다중 연결이 동일한 스레드를 공유할 수 있도록 연결 서비스를 구성할 수 있습니다. 이러한 공유 스레드 풀 모델은 브로커가 지원할 수 있는 연결 수를 현저하게 증가시킬 수 있습니다. 자세한 내용은 75페이지의 "스레드 풀 관리자"를 참조하십시오.

브로커 클러스터

브로커를 통해 전달되는 연결 수와 메시지 수가 증가함에 따라 Message Queue 서버에 추가 브로커 인스턴스를 추가하여 추가 로드를 관리할 수 있습니다. 브로커 클러스터는 많은 브로커 인스턴스 간의 클라이언트 연결 및 메시지 전달의 균형을 조정하여 메시지 서버를 크게 확장할 수 있습니다. 브로커 인스턴스는 동일한 호스트에 있거나 네트워크에 분산될 수 있습니다. 클러스터링은 비즈니스 요구가 증대됨에 따라 메시지 처리량을 향상시키고 메시지 대역폭을 확장하기에 이상적인 방법입니다. 브로커 클러스터는 5장, "브로커 클러스터"(91페이지)에서 소개하고 *Message Queue 관리 설명서*에서 자세히 설명합니다.

다중 사용자로의 대기열 전달

JMS 사양에 따라 대기열 대상의 메시지는 단일 사용자에게로만 전달될 수 있습니다. Message Queue를 사용하면 다중 사용자가 한 대기열을 사용하여 등록할 수 있습니다. 그런 다음 브로커는 메시지를 등록된 여러 사용자에게 분산하여 로드 균형을 조정하고 시스템 크기를 조정할 수 있습니다.

다중 사용자에게로의 대기열 전달 구현은 구성 가능한 로드 균형 조정 방식을 사용합니다. 이 방식을 사용하면 최대 활성 사용자 수 및 실패 시 활성 사용자를 대신하기 위해 대기하는 최대 백업 사용자 수를 지정할 수 있습니다. 또한 로드 균형 조정 메커니즘은 사용자의 현재 용량과 메시지 처리 속도를 고려합니다.

로드 균형 조정 대기열 전달에 대한 자세한 내용은 61페이지의 "다중 사용자로의 대기열 전달"을 참조하십시오.

가용성 기능

Message Queue는 서비스 중단 시간을 최소화하는 많은 기능을 제공합니다. 이 범위는 오류를 방지하기 위한 메커니즘에서 고가용성을 제공하기 위해 Sun Cluster와의 통합을 가능하게 하는 메커니즘에 이르기까지 다양합니다.

메시지 서비스 안정성

메시지 서비스의 가용성을 보장하는 가장 효과적인 방법 중 하나는 고성능을 제공하고 오류를 최소화하는 서비스를 제공하는 것입니다. Message Queue는 메모리 과부하 또는 성능 정체를 피하는 메커니즘을 제공합니다. 이러한 메커니즘은 메시지 서버와 클라이언트 런타임에서 모두 작동합니다.

메시지 서버 자원 관리 메시지 서버는 메모리 및 CPU 자원에서 제한되므로 대처할 수 없거나 불안정할 수 있는 상태까지 과부하될 수 있습니다. 이러한 경우는 일반적으로 메시지 생성 속도가 사용 속도를 훨씬 능가할 때 발생합니다. 이러한 상황을 피하기 위해 개별 물리적 대상 수준 및 시스템 차원 수준에서 브로커를 구성하여 메모리가 넘치는 것을 방지할 수 있습니다. 자세한 내용은 79페이지의 "메모리 자원 관리"를 참조하십시오.

클라이언트 런타임 메시지 흐름 제어 또한 Message Queue는 클라이언트 런타임으로 메시지 전달을 제어하는 메커니즘을 제공합니다. 흐름 제어 메커니즘을 사용하여 클라이언트가 메모리를 고갈시키지 않도록 하면서 클라이언트 런타임으로의 메시지 전달을 최적화할 수 있습니다. 자세한 내용은 41페이지의 "메시지 흐름 제어"를 참조하십시오.

메시지 서버에 자동 재연결

Message Queue는 자동 재연결 기능을 제공합니다. 메시지 서버와 클라이언트 간 연결이 실패하면 Message Queue는 연결 재설정을 시도하면서 클라이언트 상태를 유지합니다. 대부분의 경우 일단 연결이 다시 설정되면 메시지 생성 및 사용이 투명하게 재개됩니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

Sun Cluster를 통한 고가용성

Message Queue의 브로커 클러스터링은 확장성이 뛰어난 메시지 서버를 제공하지만 클러스터의 한 브로커 인스턴스에서 다른 인스턴스로의 페일오버를 지원하지 않습니다. 하지만 Message Queue는 Sun Cluster 소프트웨어와 통합하여 고가용성 메시지 서버를 제공할 수 있습니다. Message Queue용으로 개발된 Sun Cluster 에이전트를 사용하면 Sun Cluster는 브로커가 실패해도 메시지 서버를 사실상 중단 시간 없이 바로, 투명하게 복원하여 상태 데이터가 전혀 손실되지 않게 합니다.

관리성 기능

Message Queue는 메시지 서비스를 모니터 및 관리하고 메시지 서비스 성능을 조정하는 데 사용할 수 있는 많은 기능을 제공합니다.

강력한 관리 도구

Message Queue는 Message Queue 메시지 서버를 관리하고 대상, 트랜잭션, 영구 가입 및 보안을 관리하기 위한 명령줄 및 GUI 도구를 모두 제공합니다(45페이지의 "관리 도구" 참조).

또한 Message Queue는 메시지 서버의 원격 모니터링 및 관리를 비롯하여 JMS 관리 대상 객체, 사용자 저장소, 플러그인 JDBC 호환 데이터 저장소 및 자체 서명한 서버 인증서를 관리하는 도구를 지원합니다. 이러한 관리 도구 사용에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

메시지 기반 모니터링 API

Message Queue는 사용자 정의 모니터링 응용 프로그램을 작성하는 데 사용할 수 있는 간단한 JMS 기반 모니터링 API를 제공합니다. 이러한 모니터링 응용 프로그램은 특수한 주제 대상에서 메트릭 메시지를 검색하는 사용자입니다. 메트릭 메시지는 Message Queue 브로커에서 제공하는 모니터링 데이터를 포함합니다(86페이지의 "메트릭 메시지 생성자 (엔터프라이즈판)" 참조).

각 메트릭 메시지 유형에서 보고하는 메트릭 수량에 대한 자세한 내용은 메트릭 메시지를 사용할 Message Queue 클라이언트 개발 방법을 설명하는 *Java 클라이언트용 Message Queue 개발 안내서*를 참조하십시오. 메트릭 메시지 생성을 구성하는 방법에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

조정 가능한 성능

Message Queue는 메시지 서버와 클라이언트 런타임을 모두 조정하여 최적의 성능을 얻는 여러 가지 방법을 제공합니다. 주요 자원을 모니터하고 메모리 사용, 스레딩 자원, 메시지 흐름, 연결 서비스, 안정성 매개 변수 및 메시지 처리량과 시스템 성능에 영향을 미치는 기타 요소를 조정할 수 있습니다. 메시지 서비스 성능을 조정하는 방법에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

유연한 서버 구성 기능

Message Queue를 사용하여 지속성 객체, 사용자 정보 및 관리 대상 객체를 저장하는 방식을 선택할 수 있습니다.

구성 가능한 지속성

메시지 전달을 보장하기 위해 Message Queue는 메시지를 사용할 때까지 메시지 및 기타 지속성 객체를 저장합니다. Message Queue는 고성능 파일 기반 영구 저장소를 제공할 뿐 아니라 구성 가능한 지속성을 지원합니다. 따라서 Oracle 8i와 같은 내장 또는 외장 JDBC 호환 데이터베이스에 지속성 메시지를 저장할 수 있습니다. 자세한 내용은 81페이지의 "지속성 관리자"를 참조하십시오.

LDAP 서버 지원

Message Queue는 인증 및 권한 부여를 위해 필요한 관리 대상 객체 및 사용자 정보 모두에 대해 파일 기반 저장 장치를 제공합니다. 하지만 Message Queue는 관리 대상 객체 저장소 및 사용자 저장소에 대한 LDAP 서버 사용도 지원합니다. LDAP 서버는 이러한 정보를 저장하고 검색하는 보다 안전한 표준 방식을 제공하므로 생산 시스템에 권장됩니다. 관리 대상 객체 저장소 및 사용자 저장소에 대한 LDAP 서버 사용에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

제품 판

Message Queue는 엔터프라이즈판과 플랫폼판의 두 버전으로 사용할 수 있습니다. 두 버전 모두 JMS 사양을 완전히 구현하지만 각각은 다른 기능 집합과 용량을 가집니다. 다음 표에 기능 집합이 비교되어 있습니다. 기능에 대한 설명은 46페이지의 "제품 기능"을 참조하십시오.

표 2-2 기능 비교: 엔터프라이즈판과 플랫폼판

엔터프라이즈판	플랫폼판
고급 통합 지원 기능	
HTTP 지원, TCP 지원	TCP 지원
SSL(Secure Socket Layer) 표준에 기반한 보안 연결	SSL(Secure Socket Layer) 표준에 기반한 보안 연결
C 언어 API, Java API	Java API (C API는 시험 사용권으로만 사용할 수 있습니다.)
SOAP(XML) 메시징 지원	SOAP(XML) 메시징 지원
J2EE 자원 어댑터	J2EE 자원 어댑터
보안 기능	
플랫 파일 또는 LDAP 사용자 저장소에서 인증, 액세스 제어 파일을 사용하여 권한 부여 및 SSL 암호화.	엔터프라이즈판과 동일
확장성 기능	
확장 가능한 연결 용량	고정 연결 용량
메시지 서버를 브로커 클러스터로 구현할 수 있음	단일 브로커 메시지 서버
(대기열 당)무제한 수의 메시지 사용자에게 대기열 전달	(대기열 당)최대 3명의 메시지 사용자에게 대기열 전달

표 2-2 기능 비교: 엔터프라이즈판과 플랫폼판*(계속)*

엔터프라이즈판	플랫폼판
가용성 기능	
메모리 자원 관리 및 메시지 흐름 제어를 통한 메시지 서비스 안정성	엔터프라이즈판과 동일
클러스터의 다른 브로커로 클라이언트 연결 페일오버 또는 동일한 브로커로 자동 재연결.	클라이언트 연결 페일오버 없음. 동일한 브로커로의 자동 재연결이 허용됨
Sun Cluster를 통한 고가용성	Sun Cluster를 통한 고가용성
관리성 기능	
강력한 관리 도구	강력한 관리 도구
관리 도구 및 로깅을 비롯한 메시지 기반 모니터링 API	관리 도구 및 로깅. 하지만 메시지 기반 모니터링 API는 없음
조정 가능한 성능	조정 가능한 성능
유연한 서버 구성 기능	
플러그 가능 지속성	플러그 가능 지속성
LDAP 서버 지원	LDAP 서버 지원

플랫폼판 및 엔터프라이즈판의 사용권 성능은 아래에서 설명합니다.

엔터프라이즈판

Message Queue 엔터프라이즈판을 사용하여 엔터프라이즈 작업 환경에서 메시지 응용 프로그램을 배포하고 실행할 수 있습니다. 엔터프라이즈판은 메시징 응용 프로그램과 구성 요소의 개발 디버깅, 로드 테스트에도 사용할 수 있습니다. 엔터프라이즈판에는 사용되는 CPU 수에 기반하여 무기한 영구 사용권이 있습니다. 사용권은 다중 브로커 메시지 서비스의 브로커 수에 제한이 없습니다.

플랫폼판

Message Queue 플랫폼판의 경우 메시지 서버가 지원하는 클라이언트 연결의 수에 제한이 없습니다. 이 기능은 기본 사용권 또는 90일 시험 사용권과 함께 제공됩니다.

- **기본 사용권**은 무기한 사용할 수 있습니다. 기본 사용권을 갖는 플랫폼판은 생산 요구가 적은 환경에서 JMS 공급자로 사용할 수 있습니다. 이 사용권은 엔터프라이즈판 기능을 포함하지 *않습니다*.
- **90일 시험 엔터프라이즈 사용권**은 기본 사용권에 포함되지 않은 모든 엔터프라이즈판 기능을 포함합니다. 하지만 소프트웨어는 사용권에 90일의 제한을 두기 때문에 엔터프라이즈판에서 사용 가능한 기능 평가에 적합합니다. 90일 시험 엔터프라이즈 사용권 사용에 대한 지침은 *Message Queue 관리 설명서*에 설명된 시작 옵션을 참조하십시오.

플랫폼판은 Sun 웹 사이트에서 무료로 다운로드할 수 있으며 Sun Java System Application Server 플랫폼과 함께 제공됩니다. 플랫폼판에서 엔터프라이즈판으로 Message Queue를 업그레이드하는 지침은 *Message Queue 설치 설명서*에서 찾아볼 수 있습니다.

주 Message Queue의 모든 판에 대해, 제품의 일부 즉, Message Queue 클라이언트 런타임을 상업적인 용도로 무료 재배포할 수 있습니다. 하지만 제품에 있는 다른 모든 파일은 재배포할 수 *없습니다*. 무료로 재배포할 수 있는 부분을 사용하여 정식 사용자는 사용권 사용료를 내지 않고 제 3자에게 판매할 수 있는 Message Queue 클라이언트 응용 프로그램(Message Queue 서비스에 연결할 수 있는 응용 프로그램)을 개발할 수 있습니다. 제 3자는 Message Queue를 구입하여 Message Queue 메시지 서버에 액세스하거나 Message Queue 메시지 서버가 설치되어 실행 중인 다른 타사에 연결해야 합니다.

Sun 제품 컨텍스트의 Message Queue

Message Queue는 응용 프로그램에 의해 직접적으로 사용되는 미들웨어일 뿐 아니라 다른 미들웨어 및 Sun에서 제공한 다른 서버 및 응용 프로그램에서도 사용됩니다. 따라서 Message Queue는 Solaris 및 Java Enterprise System에서 제공되고 Sun Java System Application Server에서도 제공되고 있습니다.

Application Server에서 Message Queue는 J2EE 플랫폼이 JMS 공급자에게 제공하는 JMS 요구 사항을 만족시키며 Application Server가 호스트하는 응용 프로그램에 의해 직접적으로 사용됩니다. 자세한 내용은 6장, "Message Queue 및 J2EE"(97페이지)를 참조하십시오.

3장

안정적인 메시지 전달

이 장에서는 Message Queue 서비스가 안정적인 메시지 전달을 제공하는 방법을 설명합니다. 메시지를 해당 사용자에게 경로 지정하고 전달할 때 그리고 메시지가 전달되었음을 보장할 때 사용되는 다양한 메커니즘을 설명하면서 시스템을 통한 메시지의 경로를 추적합니다.

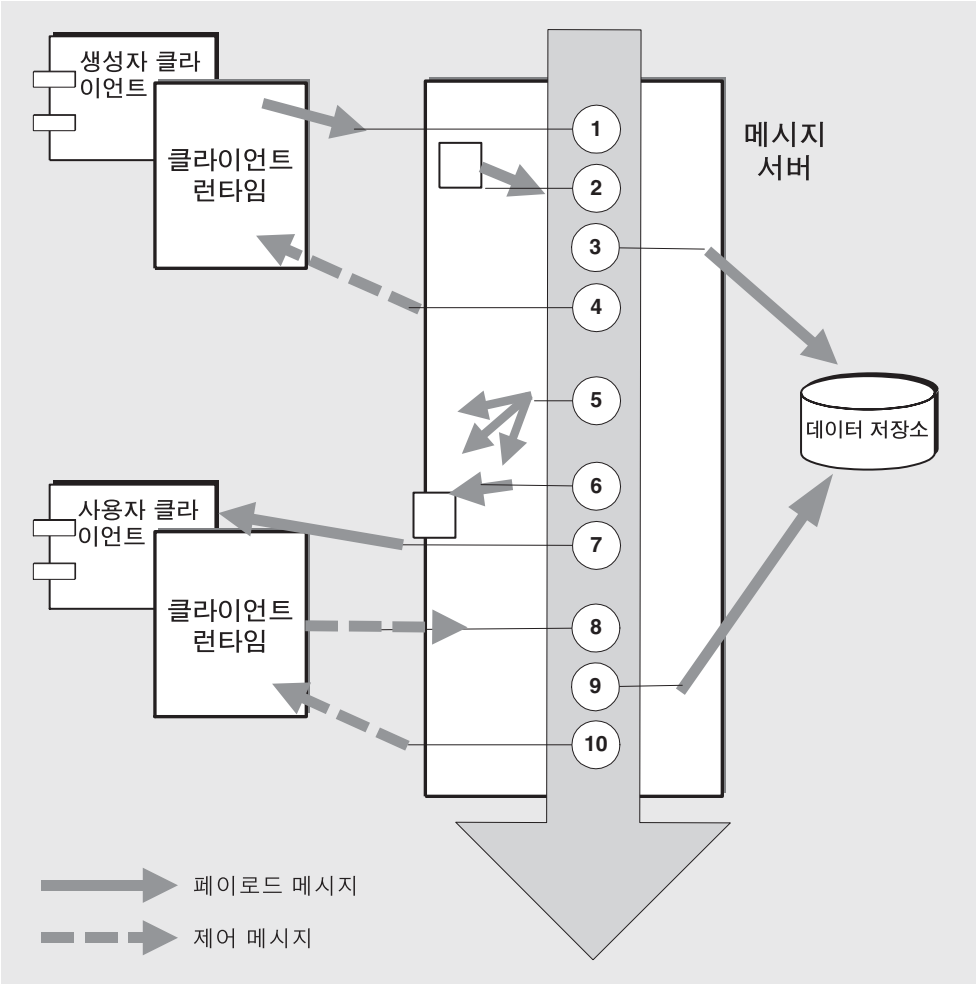
이 절은 다음 내용으로 구성되어 있습니다.

이 장에는 개발자 및 관리자 모두가 관심을 가질 자료가 있으며 2장, "Message Queue 소개"의 내용을 보완합니다.

시스템에서의 메시지 경로

Message Queue 메시지 서비스를 통한 메시지 생성자에서 메시지 사용자로의 메시지 전달은 그림 3-1에서 보여줍니다. 다음에 오는 하위 절에서는 전달 과정의 각 단계에 대한 자세한 설명을 제공합니다.

그림 3-1 메시지 전달 단계



안정적으로 전달되는 지속성 메시지의 메시지 전달 단계는 다음과 같습니다.

메시지 생성

1. 클라이언트 런타임이 연결을 통해 메시지 생성자에서 메시지 서버로 메시지를 전달합니다.

메시지 처리 및 경로 지정

2. 메시지 서버가 연결을 통해 메시지를 읽어 들여 적절한 대상에 저장합니다.
3. 메시지 서버가 (지속성) 메시지를 데이터 저장소에 저장합니다.
4. 메시지 서버가 메시지 생성자의 클라이언트 런타임에게 메시지 수신 확인을 보냅니다.
5. 메시지 서버가 메시지 경로 지정을 결정합니다.
6. 메시지 서버가 대상에서 들어온 메시지를 해당 연결에 기록합니다.

메시지 사용

7. 메시지 사용자의 클라이언트 런타임이 연결에서 메시지 사용자로 메시지를 전달합니다.
8. 메시지 사용자의 클라이언트 런타임이 메시지 사용에 대한 확인을 메시지 서버로 보냅니다.

메시지 수명 끝

9. 메시지 서버는 클라이언트 확인을 처리하여, 대상과 데이터 저장소 양쪽에서 (지속성) 메시지를 삭제합니다.
10. 클라이언트 확인이 처리되었으므로 메시지를 다시 전달할 수 없음에 대한 확인을 메시지 서버가 사용자의 클라이언트 런타임으로 보냅니다.

이러한 전달 단계에서 시스템에 의해 처리된 메시지는 두 가지 범주로 구분됩니다.

- **페이로드 메시지**. 생성자 클라이언트가 사용자 클라이언트로 보낸 JMS 메시지
- **제어 메시지**. 페이로드 메시지가 성공적으로 전달되었음을 보장하고 연결을 통한 메시지 흐름을 제어하기 위해 메시지 서버와 클라이언트 런타임 간에 전달된 확인 및 기타 비 페이로드 메시지

메시지 전달 처리

생성자에서 사용자에게 메시지를 전달하는 과정에서 Message Queue 서비스에 의한 메시지 처리는 그림 3-1 다음의 단계 설명에서 표시된 대로 여러 단계에서 진행됩니다.

단계는 다음과 같습니다.

- 60페이지의 "메시지 생성"
- 61페이지의 "메시지 처리 및 경로 지정"
- 63페이지의 "메시지 사용"
- 66페이지의 "메시지 수명 끝"

다음 절에서 이러한 단계에 대해 설명합니다.

메시지 생성

메시지 생성 단계에서는 클라이언트가 메시지를 작성하고 연결을 통해 클라이언트 런타임이 브로커의 대상으로 전달합니다.

메시지의 전달 모드가 지속성(브로커가 실패한 경우에도 단 한 차례 전달 보장)으로 설정되어 있었다면 브로커는 기본적으로 제어 메시지(브로커 확인)를 다시 클라이언트 런타임으로 전달합니다. 이 브로커 확인은 브로커가 대상으로 메시지를 전달했고 브로커의 데이터 저장소에 저장했음을 나타냅니다. 브로커 확인을 수신할 때까지 클라이언트 스레드가 차단됩니다.

메시지의 전달 모드가 비지속성으로 설정되어 있었다면 브로커는 기본적으로 브로커 확인을 클라이언트 런타임으로 다시 보내지 않으므로 클라이언트 스레드가 차단되지 않습니다. 하지만 브로커가 비지속성 메시지를 수신했는지 여부를 알아야 한다면 브로커 확인을 활성화할 수 있습니다. 실제로, 대상 메모리 제한에 도달할 때 브로커의 메시지 생성 속도를 낮추려면 브로커 확인을 활성화해야 합니다(80페이지의 "대상 메시지 제한" 참조).

메시지 처리 및 경로 지정

브로커는 들어오는 JMS 페이로드 메시지를 받을 때 메시지를 대상에 저장한 다음 해당 사용자에게로 경로 지정합니다.

일반적으로 모든 메시지는 전달되거나 만료될 때까지 물리적 대상(메모리)에 남아 있습니다. 하지만 브로커에 오류가 발생할 경우 이 메시지는 손실됩니다. 메시지가 지속성이면 브로커는 메시지를 데이터베이스 또는 파일 시스템에 저장하고 오류가 발생하면 복구합니다.

메시지 처리는 다음 절에서 설명하는 것처럼 대상 유형(대기열 또는 주제)에 따라 다릅니다. 또한 관리자가 물리적 대상을 작성할 때 대상에 대해 설정한 대상 등록 정보에 따라 달라집니다.

대기열 대상

대기열 대상은 지점간 메시징에서 사용되며 이 경우 메시지는 한 사용자에게만 전달되고 한 사용자에 의해서만 사용됩니다.

대기열의 메시지는 단일 사용자에게만 전달되지만 Message Queue에서는 다중 사용자가 하나의 대기열에 등록할 수 있습니다. 그런 다음 브로커는 메시지를 등록된 여러 사용자에게 분산하여 사용자 간에 로드 균형을 조정합니다.

기본 경로 지정 메커니즘

메시지가 생성자로부터 도달하면 대기열로 들어갑니다. 각 메시지가 대기열의 맨 앞에 도달하면 대기열에 등록된 단일 사용자로 경로 지정됩니다. 메시지가 대기열의 맨 앞에 도달하는 순서는 도착 순서 및 우선 순위에 따라 결정됩니다.

메시지에 선택기 등록 정보 값이 설정되어 있는 경우 브로커는 이 값을 등록된 사용자가 지정한 선택기 값과 비교하여 선택기 값이 일치하는지 확인한 다음 해당 메시지를 사용자에게 경로 지정합니다.

다중 사용자로의 대기열 전달

다중 사용자로의 대기열 전달 구현에서는 다음과 같은 여러 가지 대기열 대상 등록 정보을 기반으로 구성 가능한 로드 균형 조정 방법을 사용합니다.

- 로드 균형 조정된 대기열 전달에서 활성 상태가 될 수 있는 최대 사용자 수를 설정할 수 있습니다.
- 활성 사용자 중 하나가 실패할 경우 대신할 백업 사용자의 최대 수를 설정할 수 있습니다.

사용자 수가 두 등록 정보의 합계를 초과할 경우 새 사용자가 거부됩니다(Message Queue 플랫폼판은 대기열당 최대 3명의 사용자(2명의 활성 사용자와 1명의 백업 사용자)를 지원하고 Message Queue 엔터프라이즈판은 제한이 없음).

로드 균형 조정 메커니즘에서는 여러 사용자의 메시지 사용 속도를 고려합니다. 구성 가능한 크기(대기열 대상의 사용자 흐름 제한 등록 정보)의 일괄 처리에서 대기열 대상의 메시지를 사용 가능한 새 활성 사용자에게 (대기열에 등록된 순서에 따라) 경로 지정합니다. 이 메시지를 전달한 후 사용자가 사용 가능해지면 대기열에 도착하는 추가 메시지를 일괄 처리로 사용자에게 경로 지정합니다. 사용자가 이전에 전달 받은 메시지의 구성 가능한 비율을 사용하면 사용 가능해지는 것입니다. 즉, 각 사용자의 디스패치 속도는 사용자의 현재 용량과 메시지 처리 속도에 따라 다릅니다.

활성 사용자가 실패한 경우 첫 번째 백업 사용자가 활성화되어 실패한 사용자의 작업을 인수합니다. 이러한 메커니즘 때문에, 대기열 대상에 둘 이상의 활성 사용자가 있는 경우 메시지 사용 순서가 지켜지지 않을 수 있습니다.

메시지 생성 속도가 느린 경우 브로커가 활성 사용자들에게 메시지를 고르지 않게 디스패치할 수 있습니다. 활성 사용자가 필요 이상으로 많은 경우 메시지를 받지 못하는 사용자도 있을 수 있습니다.

브로커 클러스터 환경에서는 다중 사용자로의 전달 시 로컬 사용자를 우선하도록 설정할 수 있습니다. 대기열 대상 등록 정보를 사용하여 생성자의 홈 브로커, 즉 생성자가 메시지를 보낸 브로커(로컬 브로커)에 사용자가 없는 경우에만 메시지를 원격 사용자에게 전달하도록 지정할 수 있습니다. 그러면 *원격 사용자*의 홈 브로커를 통해 원격 사용자에게 경로 지정할 때 처리량이 떨어질 수 있는 상황에서 성능을 향상시킬 수 있습니다.

주제 대상

주제 대상은 대상에서 인터레스트를 등록한 모든 사용자에게 메시지가 전달되는 게시/가입 메시징에서 사용합니다.

기본 경로 지정 메커니즘

생성자로부터 메시지가 도착하면 해당 주제에 가입한 모든 사용자에게 전달됩니다. 사용자가 해당 주제에 대한 영구 가입에 등록한 경우, 메시지가 도달했을 때 사용자는 활성 상태(메시지를 받기 위한)가 아니어도 됩니다. 브로커는 이 사용자가 다시 한 번 활성화될 때까지 메시지를 보관했다가 전달합니다.

메시지에 선택기 등록 정보 값이 설정되어 있는 경우 브로커는 이 값을 등록된 사용자가 지정한 선택기 값과 비교하여 선택기 값이 일치하는지 확인한 다음 해당 메시지를 사용자에게 경로 지정합니다.

영구 가입 및 클라이언트 식별자

오직 한 사용자만 한 주제에 대한 영구 가입을 가질 수 있습니다. 이 사용자가 메시지 서버에 대한 연결을 열고 닫을 때 사용자의 아이디가 동일해야 합니다. 각 영구 가입이 한 사용자에게만 해당하는지 확인하기 위해 *클라이언트 식별자*가 사용됩니다.

클라이언트 식별자는 클라이언트를 대신하여 메시지 서버에서 관리하는 상태 정보를 메시지 서버에 대한 클라이언트 연결과 연관시킵니다. 정의에 따라 클라이언트 식별자는 고유합니다.

영구 가입을 작성하려면 클라이언트 식별자를 클라이언트가 JMS API 메소드 호출을 사용하여 프로그래밍 방식으로 설정하거나 클라이언트가 사용하는 연결 팩토리 객체에 관리 방식으로 구성되어야 합니다.

메시지 사용

메시지가 경로 지정되었으면 메시지는 해당 사용자에게 전달됩니다. 사용자가 페이로드 메시지를 수신하면 사용자 클라이언트 런타임은 클라이언트가 메시지를 수신하고 처리했음에 대한 확인을 브로커에게 보냅니다. 브로커는 대상에서 메시지를 삭제하기 전에 이 클라이언트 확인을 기다립니다. 클라이언트 확인은 개별 메시지, 메시지 그룹 또는 트랜잭션에 적용할 수 있습니다.

클라이언트 확인

JMS 사양에 따라 클라이언트는 세션을 작성할 때 세 가지 기본 확인 모드 중 하나를 지정할 수 있습니다. 원하는 메시지 전달 안정성에 따라 모드를 선택합니다.

Message Queue는 `NO_ACKNOWLDEGE` 모드를 추가하여 클라이언트 확인 모드 집합을 확장합니다. 기본 및 확장 모드는 다음 하위 절에서 설명합니다.

AUTO_ACKNOWLEDGE 모드

AUTO_ACKNOWLEDGE 모드에서 세션은 클라이언트에서 사용하는 각 메시지를 자동으로 확인합니다. 또한 브로커가 사용된 각 메시지에 대해 클라이언트 확인을 처리했음을 확인할 때까지 기다리는 동안 세션 스레드는 차단됩니다. 이러한 확인을 브로커 확인이라고 합니다.

- 메시지 수신기에 의한 비동기식 메시지 사용의 경우 메시지는 onmessage() 메소드가 반환된 다음 확인됩니다.
- 동기식 메시지 사용의 경우 메시지는 receive() 메소드가 반환되기 직전에 확인됩니다. 이 경우 메시지가 사용되기 전에 시스템에 오류가 발생하면 메시지는 부분적으로 처리되지만 손실될 수 있습니다. 안정성을 높이려면 시스템 오류가 발생할 경우 메시지 손실이 없게 해주는 CLIENT_ACKNOWLEDGE 모드 또는 트랜잭션된 세션을 사용하십시오.

CLIENT_ACKNOWLEDGE 모드

CLIENT_ACKNOWLEDGE 모드는 클라이언트에게 대부분의 제어 기능을 부여합니다. 이 모드에서 클라이언트는 하나 이상의 메시지를 사용한 후 명시적으로 확인합니다. 확인은 클라이언트가 메시지 객체의 acknowledge() 메소드를 호출할 때 발생하므로 세션에서 이전 메소드 호출 이후 해당 세션에서 사용한 모든 메시지를 확인할 수 있게 합니다. 따라서 메시지가 사용된 *순서*와 관련 없이, 세션의 많은 메시지 수신기에서 비동기식으로 사용된 메시지를 포함할 수 있습니다.

또한 브로커가 클라이언트 확인을 처리했음을 확인해주는, 사용된 메시지의 일괄 처리에 대한 브로커 확인이 반환되길 기다리는 동안 세션 스레드를 차단합니다.

클라이언트 확인 및 브로커 확인은 대개 일괄 처리되므로(하나씩 전달되지 않음), CLIENT_ACKNOWLEDGE 모드를 AUTO_ACKNOWLEDGE 모드와 비교해볼 때 일반적으로 연결 대역폭을 보존하고 브로커 확인에 필요한 오버헤드를 줄입니다. 물론 이 모드에서 클라이언트가 각 메시지를 확인할 경우 일괄 처리가 일어나지 않으며 확인이 하나씩 전송됩니다.

주 Message Queue는 또한 표준 동작이 아닌, 메소드를 호출하는 *개별* 메시지만 확인할 수 있는 CLIENT_ACKNOWLEDGE 모드에서 사용할 수 있는 특정 메소드를 제공합니다. 이는 *Java 클라이언트용 Message Queue 개발 안내서*에 설명된 프로그래밍 방법을 사용하면 가능합니다.

DUPS_OK_ACKNOWLEDGE 모드

DUPS_OK_ACKNOWLEDGE 모드에서 세션은 메시지를 10개 사용한 후 확인합니다. 이 값은 현재 구성 가능하지 않습니다. AUTO_ACKNOWLEDGE 또는 CLIENT_ACKNOWLEDGE 모드와 달리 DUPS_OK_ACKNOWLEDGE 모드에서는 브로커 확인을 요청하지 않으므로 브로커 확인을 기다릴 때까지 세션 스레드가 차단되지 않습니다.

이는 메시지가 한 번만 전달되어 사용됨을 보장하지 않는다는 의미입니다. 일반적으로 메시지는 그렇게 자주 다시 전달되지 않으며 실패 시에만 메시지가 전달됩니다. 이 경우 브로커는 자신이 전달한 메시지에 대한 클라이언트 확인을 받지 않습니다. 클라이언트가 중복 전달을 걱정하지 않는다면 DUPS_OK_ACKNOWLEDGE 모드를 사용할 수 있습니다.

클라이언트 확인은 일괄 처리되고 클라이언트 스레드가 차단되지 않으므로 메시지 처리 능력은 일반적으로 다른 모드보다 훨씬 높습니다.

NO_ACKNOWLEDGE 모드

NO_ACKNOWLEDGE 모드에서는 브로커가 클라이언트를 대신하여 클라이언트 확인을 수행하므로 사용자 클라이언트에 의해 메시지가 성공적으로 처리되었는지 보장하지 않습니다.

안정적인 전달은 중요하지 않고 메시지 처리 능력이 중요할 때 이 모드를 사용하십시오. 즉, 메시지가 짧은 간격 동안 주기적으로 전송되어 메시지 로드가 높고 메시지 손실이 크게 문제가 되지 않는 경우일 수 있습니다.

이 모드는 JMS 사양을 확장해야 하므로 다른 JMS 공급자와 함께 작동할 필요가 없는 클라이언트에 의해서만 사용되어야 합니다.

트랜잭션

위에서 설명한 클라이언트와 브로커 확인 과정은 JMS 메시지 전달이 트랜잭션으로 그룹화된 경우에도 적용됩니다. 이 경우 클라이언트와 브로커 확인 과정은 트랜잭션 수준에서 수행되므로 트랜잭션과 관련된 모든 메시지가 포함됩니다. 트랜잭션이 완결되면 브로커 확인이 자동으로 보내집니다.

브로커는 트랜잭션을 추적하면서 트랜잭션 완결 또는 실패 시 롤백이 가능하게 합니다. 또한 이 트랜잭션 관리는 더 큰 규모의 분산 트랜잭션에 포함되는 로컬 트랜잭션을 지원합니다(32페이지의 "분산 트랜잭션" 참조). 브로커는 트랜잭션이 완결될 때까지 그 상태를 추적합니다. 브로커가 시작되면 이 브로커는 아직 완결되지 않은 모든 트랜잭션을 검사하며, 수동으로 해결해야만 하는 PREPARED 상태의 트랜잭션을 제외하고 모든 트랜잭션을 롤백하도록 기본 설정되어 있습니다.

Message Queue는 XA 연결 팩토리를 통해 분산 트랜잭션 지원을 구현합니다. 이를 통해 XA 연결을 생성하고, XA 연결을 통해 XA 세션을 생성할 수 있습니다. 또한 분산 트랜잭션을 지원하려면 타사의 JTS(Java Transaction Service)나 J2EE 호환 Application Server (JTS를 제공하는)가 필요합니다.

메시지 수명 끝

브로커는 메시지가 전달된 후 대상 메모리에서 메시지를 삭제합니다. 하지만 메시지가 제대로 전달되지 않은 상태에서 삭제되는 경우가 있습니다. 다음 하위 절에서는 메시지가 삭제되는 경우를 설명합니다.

메시지의 정상 삭제

정상적인 상황에서 브로커는 클라이언트 확인을 수신하여 메시지가 성공적으로 전달되었을 때 대상 메모리에서 메시지를 삭제합니다.

브로커가 메시지를 사용자에게 전달할 때 메시지를 전송으로 표시하지만 실제로는 메시지의 수신 및 사용 여부를 알지 못합니다. 따라서 브로커는 메시지를 물리적 대상 및 영구 저장소에서 삭제하기 전에 클라이언트 확인을 기다립니다.

메시지가 주제로 전송되면 브로커는 메시지를 전달한 각 메시지 사용자로부터 클라이언트 확인을 받을 때까지 메시지를 삭제하지 않습니다. 어떤 주제에 영구 가입 시, 브로커는 각 메시지를 해당 대상에 보존하고 각 영구 가입자가 활성 사용자가 되면 메시지를 전달합니다. 브로커는 클라이언트 확인을 수신하면 이를 기록하고 모든 확인을 수신한 후에만 (그 전에 메시지가 만료되지 않으면) 메시지를 삭제합니다. 클라이언트 확인 모드에 따라 브로커는 브로커 확인을 다시 클라이언트로 전달하여 클라이언트 확인을 수신했음을 확인해줄 수 있습니다.

브로커 또는 연결이 실패한 경우 브로커는 클라이언트 확인을 수신하지 못했으므로 이전에 전달했지만 확인되지 않은 모든 메시지를 다시 전달하여 이들 메시지를 재전송 플래그로 표시합니다. 예를 들어, 대기열 사용자가 메시지 수신을 확인하기 전에 오프라인 상태가 되고 다른 사용자(같은 사용자라도 됨)가 연이어 대기열에 등록하면 브로커는 확인되지 않은 메시지를 새 사용자에게 다시 전달하고 재전송 플래그로 표시합니다. 이러한 상황에서 메시지 재전송에 대하여 걱정하는 클라이이언트 응용 프로그램은 이 플래그에 대한 메시지를 확인해야 합니다.

주 클라이언트가 수신했지만 아직 확인하지 않은 메시지의 재전송을 명시적으로 요청할 수 있는 JMS API(복구 세션)가 있습니다. 이러한 메시지를 다시 전달할 때 브로커는 해당 메시지를 재전송 플래그로 표시합니다.

메시지의 비정상적 삭제

메시지를 전달할 수 없을 때 메시지 전달을 방해한 상황에 따라서 메시지는 삭제되거나 사용 불능 메시지 대기열에 보관됩니다.

다음과 같은 상황에서 메시지는 성공적으로 전달되어 사용되기 전에 브로커에 의해 삭제됩니다.

- 관리 도구를 사용하여 하나 이상의 메시지 대상을 삭제합니다.
- 영구 가입을 제거하거나 재정의하여, 전달 가능성이 없는 상태에서 주제 대상에 메시지를 그대로 둡니다.

하지만 다음과 같은 상황에서는 메시지가 사용 불능으로 간주되며 구성한 동작에 따라서 삭제되거나 사용 불능 메시지 대기열에 보관됩니다.

- 메시지 헤더에 설정된 `JMSExpiration` 값을 초과하여 메시지가 만료됩니다.
- 대상이 메모리 제한 임계값을 초과했기 때문에 대상의 "제거" 제한 동작이 호출됩니다.
- 메시지를 사용할 수 없기 때문에(클라이언트가 예외를 생성) 메시지 전달이 실패합니다.

이러한 메시지를 보유하고 사용 불능 메시지 대기열에 보관하도록 선택할 수 있습니다. 사용 불능 메시지 대기열에 메시지를 보관할 때 브로커는 Message Queue별 등록 정보 값을 메시지에 기록하여 해당 대기열에 보관한 시간과 이유를 지정합니다.

따라서 진단 용도로 사용 불능 메시지 대기열에서 메시지를 검색할 수 있습니다. 자세한 내용은 79페이지의 "사용 불능 메시지 대기열"을 참조하십시오.

성능 문제

메시지 전달의 안정성이 높아질수록 이를 실현하기 위해 더 많은 오버헤드와 대역폭이 필요합니다. 안정성과 성능 간의 균형은 설계 시 고려해야 할 중요한 사항입니다. 비지속성 메시지를 생성하고 사용하도록 선택함으로써 *성능*을 극대화할 수 있습니다. 한편 지속성 메시지를 생성 및 사용하고 트랜잭션된 세션을 사용할 경우 안정성을 극대화할 수 있습니다. 이 둘 간에는 각 응용 프로그램의 필요 사항에 다양한 옵션이 존재합니다.

예를 들어, 메시지를 처리할 수 있는 속도는 메시징 응용 프로그램 설계, 메시지 서버의 구성 및 클라이언트 런타임의 구성을 비롯한 많은 요인의 산물입니다. 이러한 요인은 매우 개별적이지만 상호 작용하여 성능을 최대화하는 작업을 복잡하게 만들 수 있습니다.

이 절에서는 안정성과 성능간의 균형을 찾는 데 필요한 몇 가지 요인을 간단히 검토합니다.

전달 모드 전달 모드는 메시지가 최대 한 번(비지속성) 또는 단 한 차례(지속성) 전달될지를 지정합니다. 지속성 메시지를 관리하려면 연결을 통해서 흐르는 브로커 확인 메시지를 사용해야 하고 브로커 확인을 수신할 때까지 기다리면서 차단하는 클라이언트 확인 모드를 사용해야 합니다. 처리 능력을 높이기 위해 브로커 확인을 억제하도록 클라이언트 런타임을 설정할 수 있지만 이 경우 지속성 메시지가 단 한 차례 전달됨을 보장할 수 없습니다.

클라이언트 확인 모드 네 가지 클라이언트 확인 모드 각각은 서로 다른 처리 및 대역폭 오버헤드 수준을 필요로 합니다. `AUTO_ACKNOWLEDGE` 모드는 가장 많은 오버헤드를 사용하지만 메시지 단위로 안정성을 보장하고 `CLIENT_ACKNOWLEDGE` 모드는 확인을 일괄 처리하므로 대역폭 오버헤드가 덜 요구되는 반면, `DUPS_OK_ACKNOWLEDGE` 모드는 가장 적은 오버헤드를 사용하지만 메시지의 중복 전달을 허용합니다. `NO_ACKNOWLEDGE` 모드는 메시지가 손실될 수 있지만 최고의 성능을 제공합니다.

클라이언트 응용 프로그램 설계 세션의 대기열에 있는 메시지 수는 각 사용자의 메시지 로드와 세션을 사용하는 메시지 사용자 수의 함수입니다. 클라이언트가 메시지를 생성하거나 사용할 때 지연되는 경우 일반적으로 응용 프로그램을 재설계하여 메시지 생성자와 사용자를 더 많은 세션에 분산시키거나 세션을 더 많은 연결에 분산시킴으로써 성능을 향상시킬 수 있습니다. 성능에 영향을 미치는 설계 문제는 *Java 클라이언트용 Message Queue 개발 안내서* 및 *C 클라이언트용 Message Queue 개발 안내서*에서 설명합니다.

메시지 흐름 측정 연결 대역폭에 대한 제어 메시지와 페이로드 메시지 간 경합은 클라이언트 런타임에서 관리할 수 있습니다. 클라이언트 런타임을 적절하게 구성하면 브로커 확인 전달 속도를 향상시켜, 차단된 세션 스레드를 해제하고 메시지 사용 속도를 높일 수 있습니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

메시지 흐름 제한 클라이언트 런타임 자원 제한에 도달하면 메시지 사용 속도가 느려질 수 있습니다. 하나 이상의 사용자가 사용할 때까지 대기하면서, 클라이언트 런타임에 보관되는 메시지 수를 제한하면 이러한 자원 제한을 피할 수 있습니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

4장

메시지 서버

37페이지의 "메시지 서버"에서 소개된 Message Queue 메시지 서버는 단일 *브로커* 또는 메시지 경로 지정 및 전달을 수행하기 위해 함께 작동하는 일련의 브로커(브로커 클러스터)로 구성됩니다.

이 장에서는 브로커의 내부 구조를 설명하고 다양한 구성 요소를 설명하며 개발 및 작업 환경에서 이를 구성 및 관리하는 데 필요한 단계를 보여줍니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 72페이지의 "브로커 구조"
- 74페이지의 "브로커 구성 요소"
- 87페이지의 "개발 및 작업 환경"

브로커의 기능 부분을 이해하면 원하는 브로커 동작을 구성하고 작업 크기를 조정하며 성능을 최적화하는 데 도움이 될 것입니다. 따라서 이 장은 응용 프로그램 개발자보다는 관리자가 더 관심을 가질 내용입니다.

브로커 구조

메시지 전달을 수행하려면 브로커가 클라이언트와의 통신 채널을 설정하고, 인증 및 권한 부여를 수행하며, 메시지 경로를 올바르게 지정하고, 안정적인 전달을 보장하며, 시스템 성능을 모니터링할 데이터를 제공해야 합니다.

이렇게 복잡한 기능들을 수행하기 위해 브로커는 각각의 전달 과정에서 특별한 역할을 맡는 다양한 내부 구성 요소를 사용합니다. 주요 브로커 구성 요소는 그림 4-1에서 확인할 수 있으며, 표 4-1에서도 간략히 설명합니다. 메시지 라우터 구성 요소가 주요 메시지 경로 지정 및 전달 서비스를 수행하고 다른 구성 요소는 메시지 라우터가 종속되는 중요한 지원 서비스를 제공합니다.

그림 4-1 브로커 구성 요소

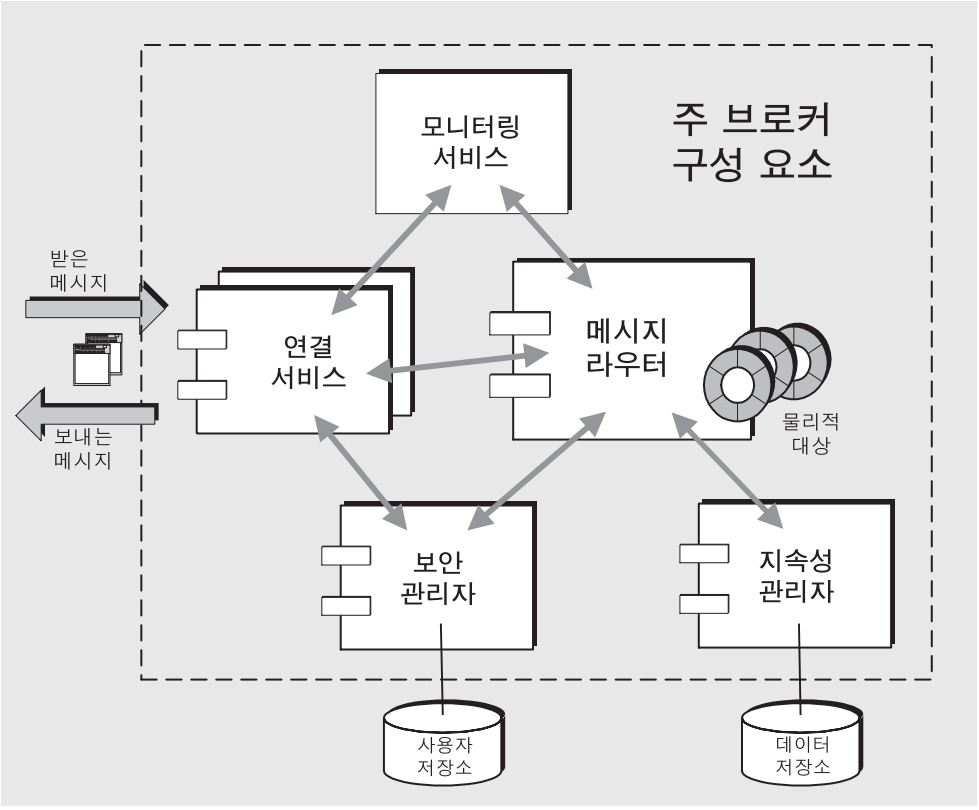


표 4-1 주요 브로커 구성 요소 및 기능

구성 요소	설명/기능
연결 서비스	브로커와 클라이언트 사이의 물리적 연결을 관리하면서 받고 보내는 메시지 전송을 담당합니다.
메시지 라우터	메시지 경로 지정 및 전달을 관리합니다. 여기에는 JMS 메시지를 비롯하여 Message Queue 메시징 시스템이 JMS 메시지 전달을 지원하기 위해 사용하는 제어 메시지도 포함됩니다.
지속성 관리자	영구 저장소에 대한 데이터 쓰기를 관리하여 시스템 오류로 인해 JMS 메시지 전달 오류가 발생하지 않게 합니다.
보안 관리자	브로커와의 연결을 요청하는 사용자에게 인증 서비스를 제공하고 인증된 사용자에게 권한 부여 서비스(액세스 제어)를 제공합니다.
모니터링 서비스	브로커를 모니터링 및 관리할 때 사용할 수 있는 다양한 출력 채널에 기록 가능한 메트릭 및 진단 정보를 생성합니다.

브로커를 구성할 때 실제로는 로드 상태, 응용 프로그램 복잡성 등에 따라 브로커 성능을 최적화하기 위해 이러한 서비스를 구성하는 것입니다.

브로커 구성 요소

다음 절에서는 그림 4-1에 표시된 각 브로커 구성 요소와 해당 기능 및 동작을 설명합니다. 각 등록 정보 및 구성 절차에 대해서는 *Message Queue 관리 설명서*를 참조하십시오.

연결 서비스

Message Queue 브로커는 응용 프로그램 클라이언트 및 관리 클라이언트 모두와의 통신을 지원합니다. 각 연결 서비스는 서비스 유형 및 프로토콜 유형을 통해 지정됩니다.

서비스 유형 서비스가 JMS 메시지 전달(NORMAL)을 제공하는지 또는 관리 도구를 지원하는 Message Queue 관리 서비스(ADMIN)를 제공하는지 지정합니다.

프로토콜 유형 서비스를 지원하는 기본 전송 프로토콜 계층을 지정합니다.

현재 Message Queue 브로커에서 사용할 수 있는 연결 서비스가 표 4-2에 표시되어 있습니다.

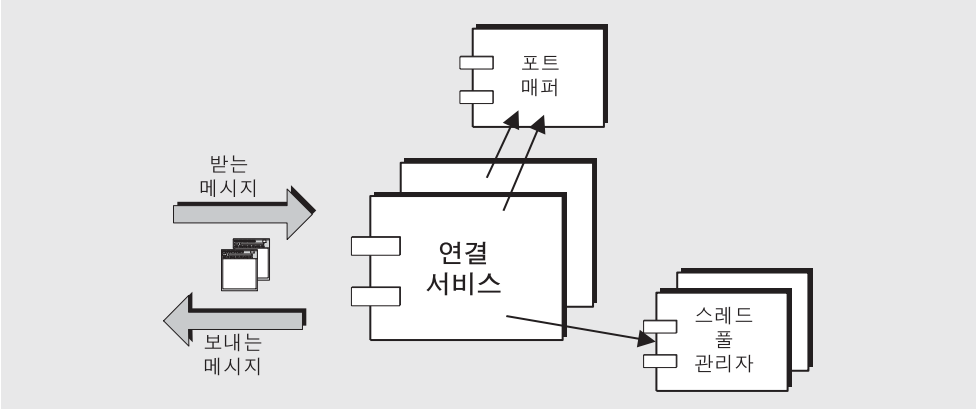
표 4-2 브로커가 지원하는 연결 서비스

서비스 이름	서비스 유형	프로토콜 유형
jms	NORMAL	TCP
ssljms	NORMAL	TLS(SSL 기반 보안)
httpjms(엔터프라이즈판)	NORMAL	HTTP
httpsjms(엔터프라이즈판)	NORMAL	HTTPS(SSL 기반 보안)
admin	ADMIN	TCP
ssladmin	ADMIN	TLS(SSL 기반 보안)

이 연결 서비스 중 어느 것이라도 또는 전부 실행하도록 브로커를 구성할 수 있습니다. 각 연결 서비스는 특정 인증 및 권한 부여(액세스 제어) 기능을 지원합니다(82페이지의 "보안 관리자" 참조). 각 연결 서비스는 다중 스레드 방식으로서, 다중 연결을 지원합니다.

각 연결 서비스는 특정 포트에서 사용 가능하며, 브로커의 호스트 이름과 포트 번호로 지정됩니다. 포트는 동적으로 할당될 수도 있고 사용자가 연결 서비스에 사용 가능한 포트를 직접 지정할 수도 있습니다. 일반 체계가 그림 4-2에 나타나 있습니다.

그림 4-2 연결 서비스 지원



포트 매퍼

연결 서비스는 일반 *포트 매퍼*에 의해 포트에 지정됩니다. 포트 매퍼 자체는 표준 포트 번호 7676에 위치합니다. Message Queue 클라이언트 런타임이 브로커와 연결을 설정할 때 먼저 포트 매퍼에 접속하여 원하는 연결 서비스의 포트 번호를 요청합니다.

`jms`, `ssljms`, `admin` 및 `ssladmin` 연결 서비스를 구성할 때 이들 연결 서비스에 대하여 *정적* 포트 번호를 지정하여 포트 매퍼를 대체할 수 있습니다. 하지만 정적 포트는 대개 방화벽을 통한 연결 등 특수 상황에서만 사용되므로(76페이지의 "HTTP/HTTPS 지원" 참조) 일반적으로 권장되지 않습니다.

스레드 풀 관리자

각 연결 서비스는 다중 스레드 방식으로서, 다중 연결을 지원합니다. 이 연결에 필요한 스레드는 *스레드 풀 관리자* 구성 요소가 관리하는 스레드 풀에서 유지 관리됩니다. 스레드 풀 관리자를 구성하여 스레드 풀에서 유지 관리되는 최소 스레드 수와 최대 스레드 수를 설정할 수 있습니다. 연결 시 스레드가 필요하면 스레드 풀에 해당 스레드가 추가됩니다. 최소 수를 초과할 경우, 시스템은 최소 임계값에 도달할 때까지 스레드를 종료시켜 여유 스레드를 확보하는 방법으로 메모리 자원을 절약합니다. 스레드 풀의 스레드는 단일 연결 전용으로 사용되거나 필요에 따라 여러 연결에 지정될 수 있습니다.

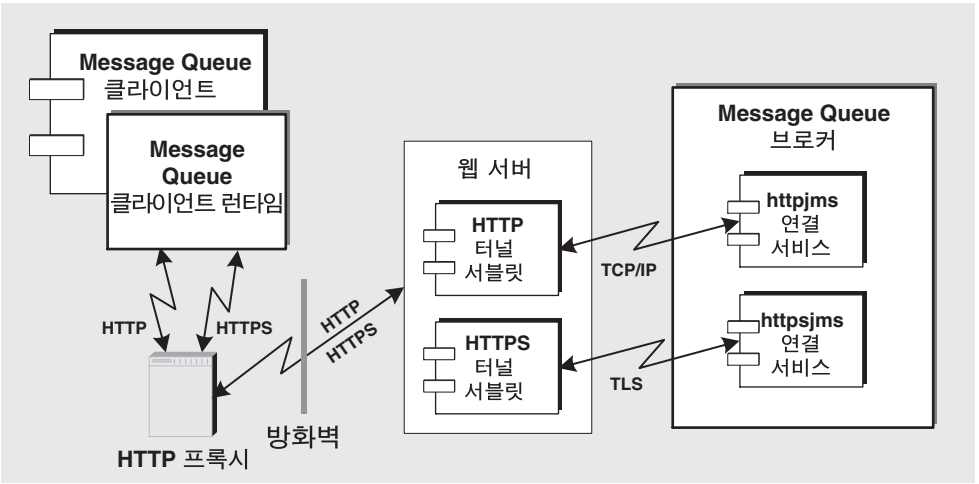
HTTP/HTTPS 지원

HTTP/HTTPS 지원을 통해 Message Queue 클라이언트는 직접 TCP 연결이 아닌 HTTP 프로토콜을 사용하여 브로커와 상호 작용할 수 있습니다. 클라이언트가 방화벽으로 브로커와 분리되어야 할 경우 방화벽을 통해 통신을 허용하므로 HTTP/HTTPS 서비스를 사용할 수 있습니다.

주 HTTP/HTTPS 지원은 Java 클라이언트에 대해 사용할 수 있지만 C 클라이언트에는 사용할 수 없습니다.

그림 4-3에서는 HTTP/HTTPS 지원과 관련된 주요 구성 요소를 보여줍니다.

그림 4-3 HTTP/HTTPS 지원 구조



- Message Queue 클라이언트쪽에서는 HTTP 또는 HTTPS 전송 드라이버가 JMS 메시지를 HTTP 요청으로 캡슐화하고 이러한 요청이 정확한 순서로 웹 서버에 전송되는지 확인합니다.
- 클라이언트는 필요한 경우 HTTP 프록시 서버를 사용해서 웹 서버와 통신할 수 있습니다.

- HTTP 또는 HTTPS 터널 서블릿(모두 Message Queue와 함께 제공)이 웹 서버에 로드되고 JMS 메시지를 브로커로 전달하기 전에 클라이언트 HTTP 요청으로부터 추출하는 데 사용됩니다. 또한 HTTP/HTTPS 터널 서블릿은 브로커 응답을 클라이언트에 다시 보냅니다. 한 HTTP/HTTPS 터널 서블릿을 사용해서 여러 브로커에 액세스할 수 있습니다.
- 브로커쪽에서는 터널 서블릿으로의 브로커 연결은 브로커 시작 시 설정됩니다. 메시지가 HTTP 또는 HTTPS 터널 서블릿에서 전송될 때 `httpjms` 또는 `httpsjms` 연결 서비스는 각각 메시지를 분해하여 브로커의 메시지 라우터 구성 요소로 제출합니다.

그림 4-3에 나타난 HTTP와 HTTPS의 구조는 매우 유사합니다. 가장 큰 차이는 HTTPS (`httpjms` 연결 서비스)의 경우 터널 서블릿이 클라이언트와 브로커 모두에 대해 보안 연결을 갖는다는 점입니다.

Message Queue의 HTTPS 터널 서블릿은 연결을 요청하는 모든 브로커에게 자체 서명된 인증서를 전달합니다. 인증서는 브로커가 HTTPS 터널 서블릿에 대해 암호화된 연결을 설정할 때 사용합니다. 이 연결이 설정되고 나면 Message Queue 클라이언트와 터널 서블릿 사이의 보안 연결을 처리할 수 있습니다.

`httpjms` 및 `httpsjms` 서비스는 *Message Queue 관리 설명서*에 설명된 등록 정보를 사용하여 구성됩니다.

메시지 라우터

지원되는 연결 서비스를 사용하여 클라이언트와 브로커 사이에 연결이 설정되면 경로 지정 및 메시지 전달을 할 수 있습니다.

Message Queue 메시징은 2단계 메시지 전달을 기반으로 합니다. 먼저 생성자 클라이언트에서 브로커의 물리적 대상으로 메시지가 전달되고, 두 번째로는 브로커의 대상에서 하나 이상의 사용자 클라이언트로 메시지가 전달됩니다. 메시지 라우터는 도착하는 메시지를 적절한 대상에 넣은 다음 메시지를 적절한 사용자에게로 경로 지정하고 전달하는 프로세스를 관리합니다.

이 절에서는 여러 가지 종류의 대상과 이러한 대상에 대한 메모리 자원 관리를 개별적 및 집합적으로 설명합니다. 메시지 경로를 지정하고 전달하는 메커니즘에 대한 설명은 3장, "안정적인 메시지 전달"에서 찾아볼 수 있습니다.

물리적 대상

물리적 대상은 브로커의 물리적 메모리에서의 위치를 나타내며, 여기서는 받은 메시지가 사용자 클라이언트에게 경로 지정되기 전에 저장됩니다.

대상은 관리 작성, 자동 작성, 임시 및 사용 불능 메시지 대기열 등 대상의 작성 방식과 용도에 따라 여러 범주로 구분됩니다.

관리 작성 대상

관리 작성 대상은 관리자가 Message Queue 관리 도구를 사용하여 작성합니다. 이들 대상은 프로그램 방식으로 작성된 논리 대상이나 클라이언트에 의해 조회되는 대상 관리 객체에 해당합니다.

Message Queue 메시지 서버는 메시징 시스템에서 중앙 허브가 되므로 이 서버의 성능 및 안정성이 성공적인 엔터프라이즈 응용 프로그램에 있어 중요한 요소입니다. 대상은 (처리하는 메시지의 수 및 크기, 등록하는 메시지 사용자의 수 및 지속성에 따라) 많은 자원을 사용할 수 있으므로 메시지 서버 성능 및 안정성을 보장하도록 면밀하게 관리해야 합니다. 따라서 관리자가 응용 프로그램을 대신하여 대상을 만들고 대상을 모니터하며 필요 시 자원 요구 사항을 재구성하는 것이 표준 방식입니다.

자동 작성 대상

자동 작성 대상은 필요할 때 관리자가 개입하지 않아도 브로커에 의해 자동으로 작성됩니다. 특히, 자동 작성 대상은 메시지 사용자 또는 메시지 생성자가 존재하지 않는 대상에 액세스를 시도할 때마다 작성됩니다. 이들 대상은 일반적으로 개발 및 테스트 주기 중, 대상을 동적으로 작성해야 할 때 사용됩니다. *자동 작성* 기능을 사용 가능하거나 사용 불가하도록 브로커를 구성할 수 있습니다.

대상이 자동으로 작성될 경우, (동일한 대상 이름을 사용하는) 서로 다른 클라이언트 응용 프로그램이 충돌하거나 (대상 지원에 필요한 자원 때문에) 시스템 성능이 저하될 수 있습니다. 이런 이유로 자동 작성 대상은 더 이상 사용되지 않을 경우, 즉 더 이상 메시지 사용자 클라이언트가 없거나 어떤 메시지도 포함하지 않는 경우 브로커에 의해 자동으로 삭제됩니다. 브로커가 다시 시작되면 지속성 메시지가 있는 경우에만 자동 작성 대상을 다시 작성합니다.

임시 대상

임시 대상은 다른 클라이언트에게 보낸 메시지의 응답을 받을 대상이 필요한 클라이언트가 (JMS API를 사용하여) 명시적으로 만들고 삭제합니다. 이러한 대상은 작성 시 해당 연결이 지속되는 동안 브로커에 의해 유지 관리됩니다. 임시 대상은 관리자가 삭제할 수 없습니다. 그리고 사용 중이라면, 즉 활성 메시지 사용자가 있는 경우에는 클라이언트 응용 프로그램에서도 삭제할 수 없습니다. 임시 대상은 (지속성 메시지가 있는) 관리 작성되거나 자동 작성된 대상과 달리 영구 저장되지 않으며 브로커가 다시 시작할 때 다시 작성되지는 않지만, Message Queue 관리 도구에 표시될 수 있습니다.

사용 불능 메시지 대기열

*사용 불능 메시지 대기열*은 브로커 시작 시 자동으로 작성되어, 진단 용도로 사용 불능 메시지를 저장하는 데 사용되는 특별한 대상입니다. *사용 불능 메시지*는 정상 처리 또는 명시적 관리 조치가 아닌 다른 이유로 인해 시스템에서 제거되는 메시지입니다. 메시지가 만료되었거나, 메모리 제한 넘침으로 인해 대상에서 제거되었거나, 전달 시도 실패로 인해 사용 불능으로 간주될 수 있습니다.

사용 불능 메시지 대기열에 메시지를 넣는 방법은 2가지가 있습니다.

- 메시지를 삭제하지 않고 사용 불능 메시지 대기열에 넣도록 대상을 구성할 수 있습니다.
- 또한 클라이언트 개발자는 메시지를 작성할 때 메시지를 사용할 수 없게 될 경우 메시지를 사용 불능 메시지 대기열에 넣을지 여부를 결정하는 등록 정보 값을 설정할 수 있습니다.

메시지가 사용 불능 메시지 대기열에 놓이면 추가 등록 정보의 정보가 기록되므로 사용 불능 원인에 대한 정보를 볼 수 있습니다.

메모리 자원 관리

메시지 서버는 메모리, CPU 사이클 등 자원이 제한되어 있습니다. 따라서, 브로커가 지원하는 메시징 응용 프로그램의 사용 패턴에 따라 메시지 서버가 넘치게 되어 응답하지 않거나 불안정하게 될 수 있습니다. 특히, 대상에 대한 메시지 생성 속도가 사용 속도보다 훨씬 빠를 때 문제가 될 수 있습니다.

메시지 라우터에는 메모리 자원을 관리하고 브로커의 메모리 부족을 방지하기 위한 메커니즘이 있습니다. 이 메커니즘은 3가지 수준의 메모리 보호 즉, 대상 메시지 제한, 시스템 전체 제한 및 시스템 메모리 임계값을 사용하여 자원이 부족하게 될 때 시스템을 계속 작동시켜 줍니다.

대상 메시지 제한

메시지가 대상에 오래 머물게 되면 메모리 자원이 문제가 되기도 합니다. 대상에 메모리를 너무 많이(시스템 메모리만 너무 많음) 또는 너무 적게(메시지가 거부됨) 할당하는 것은 모두 좋지 않습니다.

각 대상의 로드 요구에 기반하여 유연성을 허용하기 위해 각 대상에 대한 메모리 자원 및 메시지 흐름을 관리하는 등록 정보를 설정할 수 있습니다. 예를 들어, 대상에 허용되는 최대 생성자 수, 대상에 허용되는 최대 메시지 수(또는 크기) 및 단일 메시지의 최대 크기를 지정할 수 있습니다.

또한 이러한 제한에 도달할 경우 메시지 라우터가 수행하는 네 가지 응답을 지정할 수 있습니다. 네 가지 제한 동작은 다음과 같습니다.

- 메시지 생성자의 속도 줄이기
- 가장 오래된 메시지 삭제
- 보존 기간을 기준으로 우선 순위가 가장 낮은 메시지 삭제
- 최신 메시지 거부

시스템 전체 메시지 제한

시스템 전체 메시지 제한은 두 번째 보호 집합을 구성합니다. 총 메시지 수, 모든 메시지가 사용하는 메모리 등과 같이 브로커의 모든 대상에 한꺼번에 적용되는 시스템 전체 제한을 지정할 수 있습니다. 시스템 전체 메시지 제한에 도달하면 메시지 라우터는 새 메시지를 거부합니다.

시스템 메모리 임계값

시스템 메모리 임계값은 세 번째 보호 집합입니다. 브로커가 메모리 과부하 방지를 위한 조치의 수위를 점점 더 높이게 되는 사용 가능한 시스템 메모리의 임계값을 지정할 수 있습니다. 조치는 메모리 자원 상태, 즉 초록(사용 가능한 메모리 충분), 노랑(브로커 메모리 감소 중), 주황(브로커 메모리 부족) 및 빨강(브로커가 사용 가능한 메모리 없음)에 따라 달라집니다. 브로커의 메모리 상태가 초록에서 노랑 및 주황을 거쳐 빨강으로 변하면 브로커는 다음과 같이 점점 더 높은 수준의 조치를 수행합니다.

- 데이터 저장소에서 지속성 메시지의 메모리 내 사본 삭제
- 비지속성 메시지의 생성자 억제한 뒤 결국 브로커로 향하는 메시지 흐름을 중지시킴. 지속성 메시지 흐름은 브로커가 각 메시지가 확인해야 하기 때문에 자동으로 제한됨.

두 조치 모두 성능을 떨어뜨립니다.

시스템 메모리 임계값에 도달하는 경우는 대상별 메시지 제한과 시스템 전체 메시지 제한을 잘못 설정했기 때문입니다. 임계값만으론 잠재적 메모리 과부하를 제때에 잡을 수 없는 경우도 있습니다. 따라서 이 기능에만 의존하지 말고, 대상을 개별적 및 전체적으로 구성하여 메모리 자원을 최적화하는 것이 좋습니다.

대상 기반 제한 및 동작은 세밀하게 모니터하고 조정하면 메시지 유입과 유출의 균형을 유지하여 시스템 과부하가 발생할 수 없도록 하는 데 사용할 수 있습니다. 이 메커니즘은 오버헤드를 사용하고 메시지 처리량을 제한하는 단점이 있지만 운영 무결성을 유지할 수 있습니다.

지속성 관리자

오류 발생 시 브로커를 복구하려면 메시지 전달 작업 상태를 다시 작성해야 합니다. 그러기 위해서는 모든 지속성 메시지와 기본적인 경로 지정 및 전달 정보를 데이터 저장소에 저장해야 합니다. 또한 복구하려면 브로커는 다음을 수행할 수 있어야 합니다.

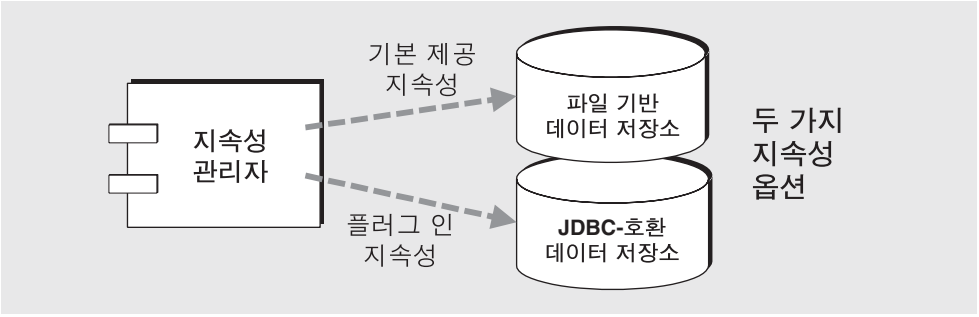
- 대상 다시 작성
- 각 주제의 영구 가입 목록 복원
- 각 메시지의 확인 목록 복원
- 완결된 모든 트랜잭션 상태 복제

지속성 관리자는 이 모든 상태 정보의 저장 및 복원을 관리합니다.

브로커가 다시 시작되면 지속성 관리자가 관리하는 데이터를 사용하여 대상 및 영구 가입을 다시 작성하고 지속성 메시지를 복구하며 열린 트랜잭션을 롤백하고 전달되지 못한 메시지의 경로 지정 테이블을 다시 작성합니다. 그런 다음 메시지 전달을 다시 시작할 수 있습니다.

Message Queue는 기본 제공 및 플러그 인 지속성 모듈을 모두 지원합니다(그림 4-4 참조). 기본 제공 지속성은 파일 기반 데이터 저장소입니다. 플러그 인 지속성은 JDBC™(Java Database Connectivity) 인터페이스를 사용하며 JDBC 호환 데이터 저장소가 필요합니다. 일반적으로 기본 제공 지속성은 플러그 인 지속성보다 더 빠릅니다. 그러나 JDBC 호환 데이터베이스 시스템에서 제공하는 중복 및 관리 기능을 선호하는 사용자도 있습니다.

그림 4-4 지속성 관리자 지원



기본 제공 지속성 기본 Message Queue 영구 저장소 솔루션은 개별 파일을 사용하여 지속성 데이터를 저장하는 파일 기반 데이터 저장소입니다. 메시지를 추가 및 제거할 때 단편화를 줄이기 위한 방법으로 파일 기반 데이터 저장소를 압축할 수 있습니다. 안정성을 최대화하려면 지속성 작업이 메모리 상태에서 물리적 저장 장치와 동기화되도록 지정할 수 있습니다. 이렇게 하면 시스템 충돌로 인한 데이터 손실을 없앨 수 있지만 성능은 느려집니다. 데이터 저장소에는 중요 정보나 소유 정보를 포함하는 메시지가 있을 수 있기 때문에 데이터 저장소 파일을 인증되지 않은 액세스로부터 보호하는 것이 좋습니다.

플러그 인 지속성 JDBC 드라이버를 통해 액세스 가능한 데이터 저장소를 모두 액세스하도록 브로커를 설정할 수 있습니다. 이 경우 여러 JDBC 관련 브로커 구성 등록 정보를 설정하고 Message Queue의 데이터베이스 관리자 유틸리티를 사용하여 적합한 체계를 갖는 데이터 저장소를 만들어야 합니다. 절차 및 관련 구성 등록 정보는 *Message Queue 관리 설명서*에 자세히 설명되어 있습니다.

보안 관리자

Message Queue 는 인증 및 권한 부여(액세스 제어) 기능을 제공하며 암호화 기능도 지원합니다.

- *인증*은 검증된 사용자만 메시지 서버에 연결할 수 있게 합니다.
- *권한 부여*는 메시지 서비스에서 지원하는 특정 작업을 수행하기 위해 연결 서비스 또는 대상 등의 자원에 액세스할 수 있는 권한을 갖는 사용자를 지정합니다.
- *암호화*는 연결을 통한 전달 중 메시지가 훼손되지 않게 보호합니다.

인증 및 권한 부여 기능은 사용자 저장소(84페이지의 그림 4-5 참조), 즉 메시징 시스템 사용자에 대한 정보(예: 아이디, 비밀번호, *그룹* 멤버쉽)를 포함하는 파일, 디렉토리 또는 데이터베이스에 따라 달라집니다. 아이디와 비밀번호는 브로커와의 연결 요청 시 사용자를 인증할 때 사용됩니다. 대상에 대한 메시지 생성/사용과 같은 작업 권한을 부여할 때 아이디와 그룹 멤버쉽이 액세스 제어 파일과 함께 사용됩니다.

Message Queue 제공 사용자 저장소를 채우거나 기존 LDAP 사용자 저장소를 브로커에 플러그 인할 수 있습니다. 플랫 파일 사용자 저장소는 쉽게 사용할 수 있지만 보안 공격에 취약하므로 평가 및 개발 용도에 한해 사용해야 합니다. LDAP 사용자 저장소는 안전하므로 작업 환경에 가장 적합합니다.

다음 하위 절에서는 인증, 권한 부여 및 암호화를 설명합니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

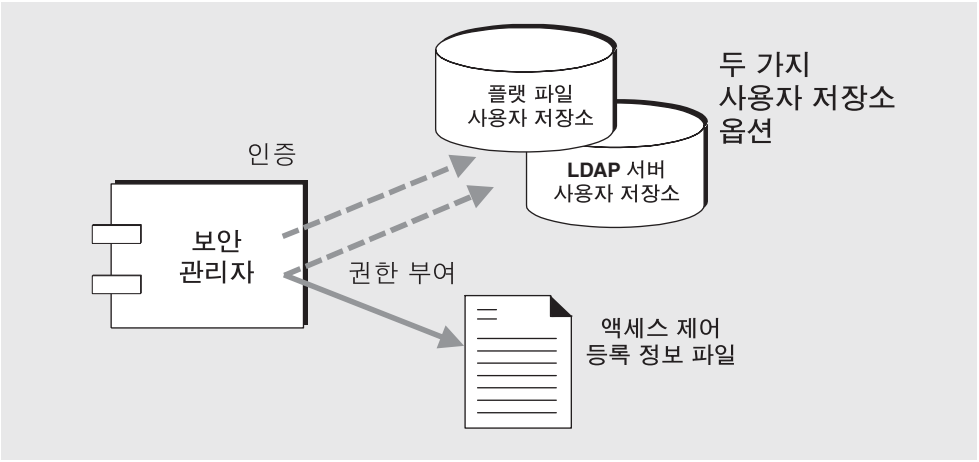
인증

Message Queue 보안은 비밀번호 기반의 인증을 지원합니다. 클라이언트가 브로커와의 연결을 요청할 경우 이 클라이언트는 아이디와 비밀번호를 제출해야 합니다. 보안 관리자는 클라이언트가 제출한 아이디와 비밀번호를 사용자 저장소에 저장된 정보와 비교합니다. 클라이언트가 브로커에게 비밀번호를 전송할 때 이 비밀번호는 기본-64 인코딩이나 메시지 다이제스트(MD5) 중 한 가지를 사용하여 암호화됩니다. 보다 안전한 전송에 대해서는 85페이지의 "암호화"를 참조하십시오. 별도로 각 연결 서비스가 사용하는 인코딩 유형을 구성하거나 브로커 전체에 대한 인코딩을 설정할 수 있습니다.

권한 부여

클라이언트 응용 프로그램 사용자가 인증되면 이 사용자는 여러 Message Queue 관련 작업을 수행할 권한을 갖습니다. 보안 관리자는 사용자 기반 및 그룹 기반 액세스 제어를 모두 지원합니다. 사용자 저장소에 있는 아이디 또는 해당 사용자가 속한 그룹에 따라 이 사용자는 특정 Message Queue 작업을 수행할 권한을 갖습니다. 액세스 제어 등록 정보 파일에서 액세스 제어를 지정합니다(그림 4-5 참조).

그림 4-5 보안 관리자 지원



사용자가 어떤 작업을 수행하려 하면 보안 관리자는 (액세스 제어 등록 정보 파일에 있는) 해당 작업 액세스를 위해 지정된 아이디 및 그룹 멤버쉽을 (사용자 저장소에 있는) 아이디 및 그룹 멤버쉽과 대조 확인합니다. 액세스 제어 등록 정보 파일은 다음 작업에 대한 권한을 지정합니다.

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

사용자 저장소에서 그룹 및 이 그룹의 사용자를 정의할 수 있습니다(단, 플랫 파일 사용자 저장소에서는 그룹 기능이 완전히 지원되지 않음). 그런 다음, 액세스 제어 등록 정보 파일을 편집하여 생성, 사용 또는 찾아보기 목적으로 사용자 또는 그룹이 액세스할 수 있는 대상을 지정할 수 있습니다. 개별 대상이나 모든 대상에 대한 액세스 권한을 특정 사용자나 그룹에게만 부여할 수 있습니다.

또한 브로커가 대상을 자동으로 작성할 수 있도록 구성된 경우(78페이지의 "자동 작성 대상" 참조), 액세스 제어 등록 정보 파일을 편집하여 브로커가 특정 사용자를 위해 대상을 자동으로 작성할 수 있도록 제어할 수 있습니다.

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.

모니터링 서비스

브로커는 로그 작업을 모니터링하고 진단할 다양한 구성 요소를 포함합니다. 다음은 그러한 구성 요소의 예입니다.

- 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 사용할 수 있도록 주제 대상에게 보내는 메시지 생성자

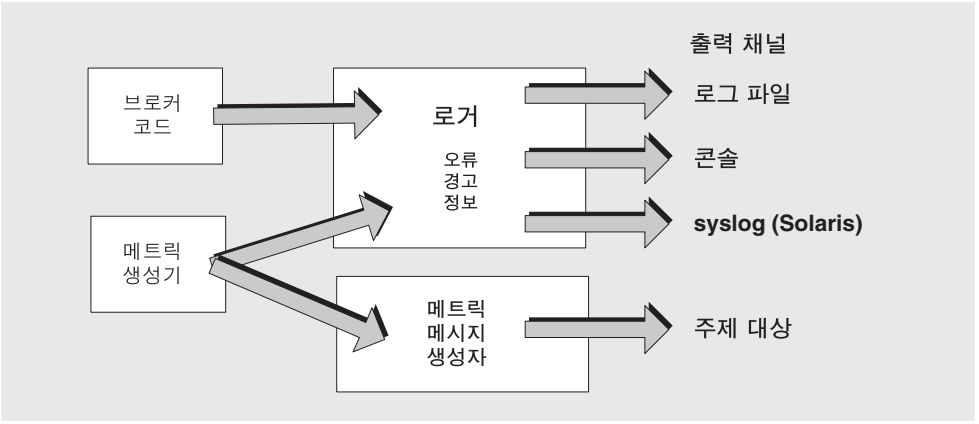
86페이지의 그림 4-6은 일반 체계를 보여줍니다.

메트릭 생성자

그림 4-6에 표시된 메트릭 생성자는 브로커 내부 및 외부로의 메시지 흐름, 브로커 메모리의 메시지 수 및 이 메시지가 사용하는 메모리, 열려 있는 연결 수, 사용 중인 스레드 수 등과 같은 브로커 활동 정보를 제공합니다.

메트릭 데이터 생성을 설정 또는 해제하고 메트릭 보고서 생성 빈도를 지정할 수 있습니다.

그림 4-6 모니터링 서비스 지원



로거

그림 4-6에 표시된 Message Queue 로거는 브로커 코드 및 메트릭 생성자가 생성한 정보를 가져와서 표준 출력(콘솔), 로그 파일, `syslog` 데몬 프로세스(Solaris™ 플랫폼의 경우) 등과 같은 여러 출력 채널에 해당 정보를 기록합니다.

로거에서 수집된 정보의 유형과 각 출력 채널에 기록된 유형을 지정할 수 있습니다. 로그 파일의 경우 로그 파일을 닫고 출력을 새 파일로 롤오버하는 지점을 지정할 수 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 *Message Queue 관리 설명서*를 참조하십시오.

메트릭 메시지 생성자(엔터프라이즈판)

그림 4-6에 나타난 메트릭 메시지 생성자 구성 요소는 메트릭 생성자 구성 요소로부터 일정 간격으로 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 대상의 메시지를 사용하고 메시지에 포함된 메트릭 정보를 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 메트릭 메시지를 사용할 Message Queue 클라이언트 개발 방법을 설명하는 *Java 클라이언트용 Message Queue 개발 안내서*를 참조하십시오. 메트릭 메시지 생성을 구성하는 방법에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

개발 및 작업 환경

메시징 응용 프로그램을 개발 및 테스트할 뿐만 아니라 작업 환경에서 이러한 응용 프로그램을 배포 및 관리하려면 Message Queue 서비스에서 제공하는 메시징 기반 구조가 필요합니다.

이 절에서는 개발 및 작업 환경에서 Message Queue을 사용하는 다양한 접근 방식을 소개합니다. 이 장은 다음 내용으로 구성되어 있습니다.

- 87페이지의 "개발 환경 및 작업"
- 88페이지의 "작업 환경 및 작업"

관리자가 작업 환경 및 작업을 설정하고 관리하는 일을 담당하지만 개발자가 이 환경의 일부를 설정 및 구성하고 클라이언트가 예상대로 작동하는 지 여부를 판단할 때 필요한 정보를 관리자에게 제공하려면 이러한 환경을 이해하는 것이 중요합니다.

개발 환경 및 작업

개발 환경에서의 작업은 Message Queue 클라이언트 응용 프로그램을 프로그래밍하는 데 중점을 둡니다. Message Queue 메시지는 주로 테스트 용도로 필요합니다.

표준 구성

Message Queue 제품은 표준 사용되도록 설계되었습니다. 기본값으로 브로커를 시작할 수 있으며 기본 데이터 저장소, 사용자 저장소 및 액세스 제어 등록 정보 파일이 제공됩니다.

기본 사용자 저장소는 관리자가 개입하지 않고도 설치 후 바로 Message Queue 브로커를 사용할 수 있게 해주는 기본 항목으로 작성됩니다. 즉, 브로커를 사용하기 위해 초기 아이디/비밀번호를 설정할 필요가 없습니다. 기본 아이디(`guest`) 및 비밀번호(`guest`)를 사용하여 클라이언트 사용자를 인증할 수 있습니다.

또한 새 응용 프로그램 개발 과정을 안내하는 여러 샘플 응용 프로그램도 제공됩니다.

개발 실습

개발 환경에서는 유연성이 강조되며 일반적으로 다음과 같은 방법을 적용합니다.

- 관리는 주로 개발자가 테스트에 사용할 브로커를 시작하는 정도로 최소화합니다.
- 데이터 저장소(기본 제공 파일 기반 지속성), 사용자 저장소(파일 기반 사용자 저장소) 및 액세스 제어 등록 정보 파일의 기본 구현을 사용합니다. 이 기본 구현은 일반적으로 개발 테스트에 적합합니다.
- 관리 대상 객체를 저장하는 (해당 용도의 디렉토리를 작성하여) 단순 파일 시스템 객체 저장소를 사용하거나 클라이언트 코드에서 관리 대상 객체를 인스턴스화하므로 객체 저장소를 전혀 사용하지 않습니다.
- 다중 브로커 테스트를 수행 중이라면 마스터 브로커는 필요하지 않을 것입니다(94페이지의 "클러스터 동기화" 참조).
- 일반적으로 대상은 명시적으로 만들지 않고 자동 작성된 대상을 사용합니다.

작업 환경 및 작업

작업 환경에서 성능을 최대화하려면 응용 프로그램을 배포, 관리 및 조정해야 합니다. 이러한 상황에서는 메시징 기반 구조를 관리하고 조정하는 것이 중요한(매우 중요한 것은 아니더라도) 요구 사항입니다.

또한 배포는 규모 및 가용성 모두에서 엔터프라이즈 요구 사항을 지원해야 합니다. 이를 위해 성능을 조정하고 시스템을 확장하여 증가하는 로드를 만족시키고 메시지 서비스 및 응용 프로그램별 자원을 매일 모니터하고 관리하면서 메시지 서비스의 구성 및 설정을 사용자 정의해야 합니다. 따라서, 관리 작업은 메시징 시스템 및 이 시스템이 지원해야 하는 응용 프로그램의 복잡성에 따라 달라집니다. 다음 절에서는 관리 작업을 설정 작업과 유지 보수 작업으로 분류합니다. 이러한 작업을 수행하는 데 필요한 절차는 *Message Queue 관리 설명서*를 참조하십시오.

설정 작업

작업 환경에서는 보안 액세스 설정, 연결 팩토리 및 대상 객체 구성, 클러스터 설정, 영구 저장소 구성 및 메모리 관리가 필요합니다.

➤ **작업 환경을 설정하는 방법**

보통 다음 설정 작업 중 전부는 아니더라도 최소한 일부는 수행해야 합니다.

- **관리 액세스 보안**(관리 도구 사용 보호)
  - admin 연결 서비스가 활성 상태인지 확인합니다.
  - 권한 부여: 특정 개인 또는 admin 그룹에 대해 admin 연결 서비스에 액세스할 수 있게 합니다.
  - 그룹에 대해 권한 부여할 경우 관리자가 admin 그룹에 속해 있는지 확인합니다.
    - 파일 기반 사용자 저장소: 기본 admin 그룹을 가집니다. 관리자가 admin 그룹에 속해 있는지 확인합니다. 또는 기본 admin 사용자를 사용하는 경우 admin 비밀번호를 변경합니다.
    - LDAP 사용자 저장소: 관리자가 admin 그룹에 속하는지 확인합니다.
- **클라이언트 액세스 보안**:
  - 인증: 파일 기반 사용자 저장소에 항목을 만들거나 브로커가 기존 LDAP 사용자 저장소를 사용하도록 구성합니다.

    (최소한 관리 기능은 비밀번호로 보호합니다.)
  - 권한 부여: 액세스 제어 등록 정보 파일에서 액세스 설정을 수정합니다.
  - 암호화: SSL 기반 연결 서비스를 설정합니다.
- **물리적 대상 작성**
- **관리 대상 객체 작성**:
  - LDAP 객체 저장소를 구성하거나 설정합니다.
  - 연결 팩토리 및 대상 관리 객체를 작성합니다.
- **필요한 경우 브로커 클러스터 작성**:
  - 중앙 구성 파일을 작성합니다.
  - 마스터 브로커를 지정합니다.
- **플러그 인 지속성을 사용하도록 브로커 구성**
- **메모리 관리 구성**

  메시지에 할당된 메시지 수와 메모리 양이 사용 가능한 브로커 메모리 자원에 적합하도록 대상 속성을 설정합니다.

유지 보수 작업

작업 환경에서는 Message Queue 메시지 서버 자원을 모니터하고 제어해야 합니다. 응용 프로그램 성능, 신뢰성 및 보안이 매우 중요하며, Message Queue 관리 도구를 사용하여 다음과 같은 다양한 작업을 지속적으로 수행해야 합니다.

➤ **작업 환경을 유지 보수하는 방법**

- **응용 프로그램 동작 관리**
  - 브로커의 자동 작성 기능을 비활성화합니다.
  - 응용 프로그램을 대신하여 물리적 대상을 작성합니다.
  - 대상에 대한 사용자 액세스를 설정합니다.
  - 대상을 모니터하고 관리합니다.
  - 영구 가입을 모니터하고 관리합니다.
  - 트랜잭션을 모니터하고 관리합니다.
- **브로커 모니터 및 조정**
  - 브로커 메트릭을 사용하여 브로커를 조정 및 재구성합니다.
  - 브로커 메모리 자원을 관리합니다.
  - 클러스터에 브로커를 추가하여 로드 균형을 조정합니다.
  - 실패한 브로커를 복구합니다.
- **관리 대상 객체 관리:**
  - 필요에 따라 연결 팩토리 및 대상 관리 객체를 추가로 작성합니다.
  - 연결 팩토리 속성 값을 조정하여 성능 및 처리 능력을 향상시킵니다.

5장

브로커 클러스터

Message Queue Enterprise Edition은 메시지 전달 서비스를 클라이언트에 제공하기 위해 함께 작업하는 브로커 그룹인 *브로커 클러스터*의 사용을 지원합니다. 클러스터를 사용하여 메시지 서버는 다중 브로커 간에 클라이언트 연결을 분배하는 방식으로 메시지 트래픽의 볼륨에 대한 작업 크기를 조절할 수 있습니다.

이 장에서는 이러한 브로커 클러스터의 구조 및 내부 기능에 대해 설명합니다. 이 장은 다음 내용으로 구성되어 있습니다.

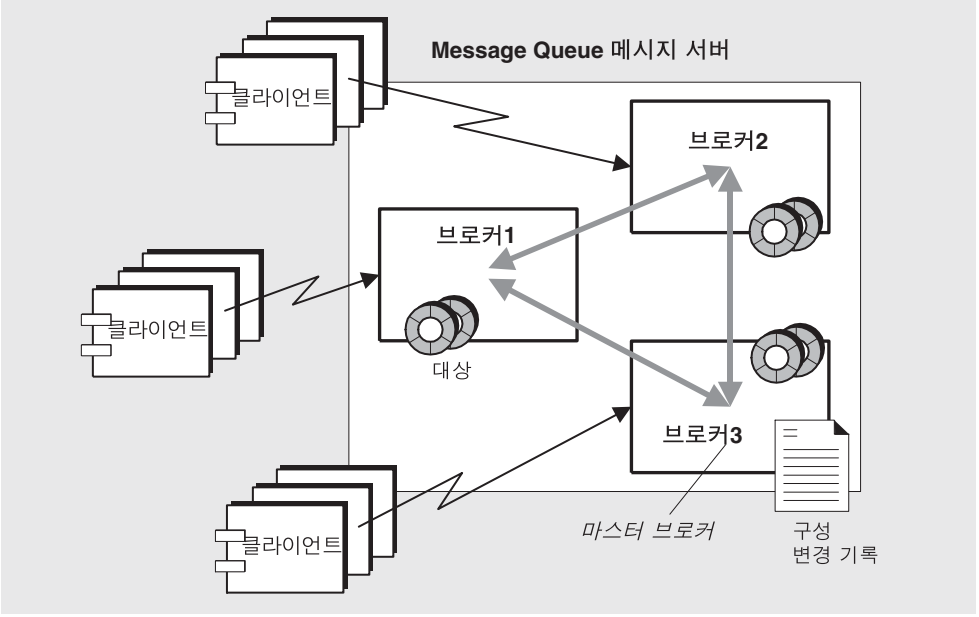
브로커 클러스터를 구성 및 관리하는 관리자이거나 클러스터를 사용하여 메시징 응용 프로그램을 테스트해야 하는 개발자라면 이 장을 읽어야 합니다.

클러스터 구조

그림 5-1은 브로커 클러스터에 대한 Message Queue의 구조를 보여줍니다. 클러스터 내 각 브로커는 다른 모든 브로커에게 직접 연결됩니다. 각 클라이언트(메시지 생성자 또는 사용자)에는 직접 통신하는 단일 *홈 브로커*가 있어 서버에 해당 브로커만 있는 것처럼 메시지를 송수신합니다. 안보이는 곳에서 홈 브로커는 연결된 모든 클라이언트에 전달 서비스를 제공하는 로드를 공유하기 위해 클러스터의 다른 브로커와 함께 작업합니다.

클러스터 내에서 한 브로커를 *마스터 브로커*로 지정할 수 있습니다. 마스터 브로커는 클러스터의 지속성 항목(대상 및 영구 가입)에 대한 변경 사항이 기록되는 *구성 변경 기록*을 유지 관리합니다. 이 레코드는 변경이 발생할 당시 오프라인으로 있었던 브로커에게 이러한 변경 정보를 전파하는 데 사용됩니다. 자세한 내용은 아래의 "클러스터 동기화"를 참조하십시오.

그림 5-1 클러스터 구조



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

메시지 전달

클러스터 구성에서 각 대상은 클러스터의 모든 브로커에 복제됩니다. 각 브로커는 다른 모든 브로커의 대상에 등록된 메시지 사용자에 대해 알고 있습니다. 따라서 각 브로커는 자신과 직접 연결된 메시지 생성자로부터 원격 메시지 사용자에게 메시지를 전달하고, 원격 생성자로부터 자신과 직접 연결된 사용자에게 메시지를 전달할 수 있습니다.

메시지 생성자 자신의 홈 브로커는 모든 저장소 및 경로 지정을 처리하고 해당 생성자가 보낸 메시지에 대한 모든 클라이언트의 확인을 처리합니다. 클러스터 내 메시지 트래픽을 최소화하기 위해 메시지는 대상 브로커에 연결된 사용자에게 전달될 때만 한 브로커에서 다른 브로커로 전송됩니다. 경우에 따라(예: 다중 사용자에게 대기열 전달) 로컬 사용자에 대한 전달이 원격 사용자에 대한 전달보다 높은 우선 순위를 갖도록 지정하여 트래픽을 더 줄일 수 있습니다. 클라이언트와 메시지 서버 간에 암호화된 보안 메시지 전달이 필요한 경우 브로커 간의 메시지 전달도 보안하도록 클러스터를 구성할 수 있습니다.

주 몇 가지 예외가 있지만 클러스터의 대상 등록 정보는 개별 대상 인스턴스가 아니라 전체 클러스터에 집합적으로 적용됩니다. 특정 대상 등록 정보에 대한 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

클러스터 구성

시작 시 클러스터의 브로커 간에 연결을 설정하려면 각 브로커는 다른 모든 브로커(마스터 브로커 포함, 있을 경우)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 *Message Queue 관리 설명서*를 참조하십시오.

주 원래 클러스터 구성 파일은 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. 이러한 구성 변경의 경우는 다음과 같습니다.

- 클러스터 브로커 중 하나에 대상이 작성되거나 삭제된 경우
- 메시지 사용자가 해당 홈 브로커에 등록된 경우
- 메시지 사용자와 홈 브로커 사이의 연결이(명시적으로 또는 클라이언트, 브로커 또는 네트워크 오류로 인해) 끊기는 경우
- 메시지 사용자가 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. 오프라인된 브로커를 수용하기 위해 Message Queue는 작성되거나 삭제된 모든 지속성 항목(대상 및 영구 가입)을 기록하여 클러스터에 대한 *구성 변경 기록*을 유지 관리합니다. 오프라인된 브로커가 다시 온라인 상태로 되면(또는 새 브로커가 클러스터에 추가되면) 해당 브로커는 대상 및 영구 가입자에 대한 정보를 이 레코드에서 참조한 다음 다른 브로커들과 현재 활성 메시지 사용자에 대한 정보를 교환합니다.

*마스터 브로커*로 지정된 클러스터의 한 브로커가 구성 변경 기록을 유지 관리합니다. 다른 브로커는 마스터 브로커 없이는 초기화를 완료할 수 없으므로 클러스터 내에서 마스터 브로커가 항상 처음으로 시작되어야 합니다. 마스터 브로커가 오프라인이 되면 다른 브로커가 구성 변경 기록에 액세스할 수 없으므로 구성 정보를 클러스터 전체에 전달할 수 없습니다. 이 상태에서 대상이나 영구 가입을 작성하거나 삭제하려는 경우 또는 영구 가입 재활성화와 같은 관련 작업을 시도할 경우 예외가 발생합니다. 하지만 비관리 메시지 전달은 계속해서 정상적으로 작동합니다.

배포 환경

브로커 클러스터의 사용은 개발 환경에서 배포되었는지 아니면 작업 환경에서 배포되었는지에 따라 달라집니다.

개발 환경

클러스터를 테스트 용도로 사용하고 확장성 및 브로커 복구를 심각하게 고려하지 않아도 되는 개발 환경에서는 마스터 브로커가 별로 필요하지 않습니다. 테스트 환경에서는 종종 대상이 자동 작성되고(78페이지의 "자동 작성 대상" 참조), 이 대상에 대한 영구 가입은 테스트 중인 응용 프로그램에서 작성하고 삭제합니다. 마스터 브로커가 없는 경우 Message Queue는 다른 브로커를 시작하기 위해 마스터 브로커가 반드시 실행 중이어야 할 필요는 없으며, 대상 및 영구 가입을 변경할 수 있고 이 변경 사항은 실행 중인 모든 브로커에게 전달될 수 있습니다. 그러나 브로커가 오프라인되었다가 나중에 복원될 경우 오프라인 상태였을 때 변경된 사항은 동기화하지 않습니다. 마스터 브로커를 사용하도록 환경을 재구성할 경우 Message Queue는 일반 요구 사항을 다시 적용합니다.

작업 환경

확장성 및 브로커 복구를 심각하게 *고려해야 하는* 작업 환경에서는 반드시 마스터 브로커를 사용하고 구성 변경 기록을 유지 관리해야 합니다. 그래야 브로커가 오프라인되었다가 다시 복원될 경우 오프라인이었을 때 변경된 사항과 동기화됩니다.

실제로 구성 변경 기록을 정기적으로 백업하여 사고에 의한 기록 손상 및 마스터 브로커 오류를 방지하는 것이 바람직합니다. Message Queue는 구성 변경 기록을 백업하고 복구할 수 있는 명령줄 옵션을 제공합니다. 필요하면 마스터 브로커 역할을 하는 브로커를 변경할 수도 있습니다. 자세한 내용은 *Message Queue 관리 설명서*를 참조하십시오.

6장

Message Queue 및 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 메시지 교환을 통해 다른 구성 요소와 서로 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 살펴보겠습니다. 이 절은 다음 내용으로 구성되어 있습니다.

이 장에서는 J2EE 구성 요소 프로그래밍 및 배포를 모두 다루기 때문에 응용 프로그램 개발자와 관리자 모두가 관심을 가질 내용입니다.

JMS/J2EE 프로그래밍: Message-Driven Bean

28페이지의 "JMS 프로그래밍 모델"에서 소개한 일반적인 JMS 클라이언트 프로그래밍 모델 외에도 J2EE 플랫폼 응용 프로그램 컨텍스트에서 사용하는, 보다 특수화된 JMS 버전이 있습니다. 이 특수화된 JMS 클라이언트를 *Message-Driven Bean*이라고 부르며, EJB 2.0 사양(`http://java.sun.com/products/ejb/docs.html`)에 지정된 EJB(Enterprise JavaBeans) 구성 요소 계열 중 하나입니다.

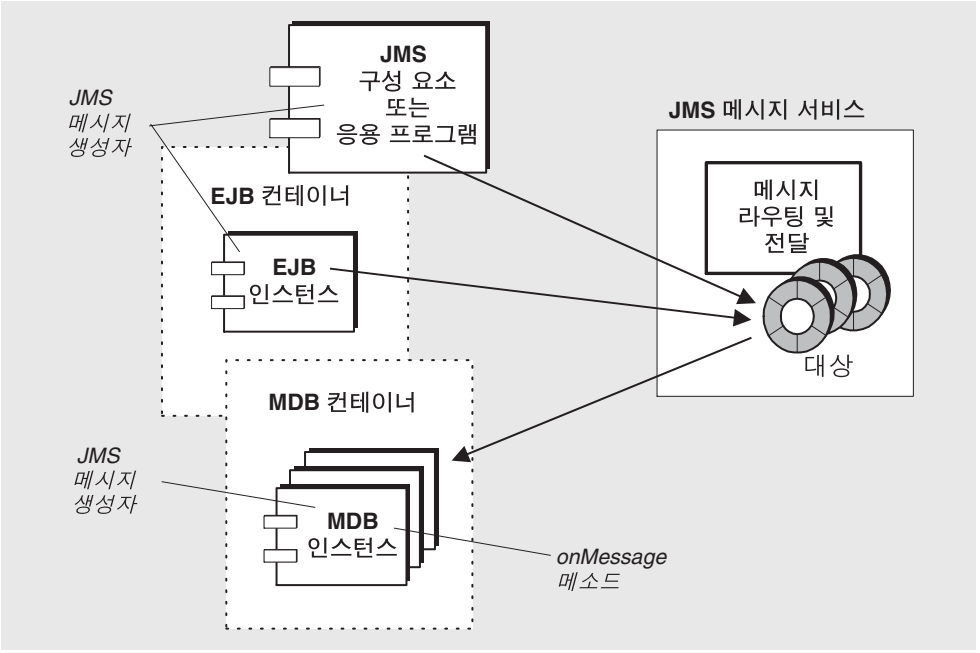
Message-Driven Bean이 필요한 이유는 다른 EJB 구성 요소(Session Bean과 Entity Bean)들이 동기식 호출만 가능하기 때문입니다. 이 EJB 구성 요소들은 표준 EJB 인터페이스를 통해서만 액세스할 수 있으므로 비동기식으로 메시지를 수신할 메커니즘이 없습니다.

그러나 비동기식 메시징은 많은 엔터프라이즈 응용 프로그램에서 필요합니다. 그러한 응용 프로그램 중 대부분에서는 서버측 구성 요소가 서버 자원을 독점하지 않으면서 상호 통신하고 응답할 수 있어야 합니다. 즉 메시지 생성자와 밀접하게 연결되지 않으면서 메시지 수신 및 사용이 가능한 EJB 구성 요소가 필요합니다. 이 기능은 서버측 구성 요소가 응용 프로그램 이벤트에 응답해야 하는 모든 응용 프로그램에서 필요합니다. 또한 엔터프라이즈 응용 프로그램에서도 로드 증가에 따라 이 기능이 확장되어야 합니다.

MDB(Message-Driven Bean)는 특정 EJB 컨테이너(지원하는 구성 요소에 대해 분산 서비스를 제공하는 소프트웨어 환경)가 지원하는 특정 EJB 구성 요소입니다.

Message-Driven Bean MDB는 JMS `MessageListener` 인터페이스를 구현하는 JMS 메시지 사용자입니다. `onMessage` 메소드(MDB 개발자가 작성)는 MDB 컨테이너가 메시지를 수신할 때 호출됩니다. 이 `onMessage()` 메소드는 표준 `MessageListener` 객체의 `onMessage()` 메소드처럼 메시지를 사용합니다. 다른 EJB 구성 요소에서처럼 MDB에 대해 메소드를 원격 호출하지 않으므로 MDB와 관련된 홈 또는 원격 인터페이스는 없습니다. MDB는 단일 대상으로부터의 메시지를 사용할 수 있습니다. 그림 6-1에서 확인할 수 있듯이 독립형 JMS 응용 프로그램, JMS 구성 요소, EJB 구성 요소 또는 웹 구성 요소에서 메시지를 생성할 수 있습니다.

그림 6-1 MDB와의 메시지



MDB 컨테이너 MDB 인스턴스를 생성하고 비동기식으로 메시지를 사용하도록 설정하는 역할을 하는 특수화된 EJB 컨테이너가 MDB를 지원합니다. 여기에는 메시지 서비스와의 연결 설정(인증 포함), 지정된 대상과의 세션 풀 생성, 세션 풀 및 관련 MDB 인스턴스 사이에서의 수신 메시지 배포 관리가 포함됩니다. 이 컨테이너는 MDB 인스턴스의 라이프사이클을 제어하므로 MDB 인스턴스 풀이 받는 메시지 로드를 수용할 수 있도록 관리합니다.

MDB와 관련 있는 배포 설명자는 컨테이너가 메시지 사용 설정 시 사용하는 관리 대상 객체, 즉 연결 팩토리와 대상의 JNDI 조회 이름을 지정합니다. 또한 배포 설명자는 배포 도구가 컨테이너 구성 시 필요한 다른 정보를 포함할 수 있습니다. 이 컨테이너 각각은 단일 MDB로만 이루어진 인스턴스를 지원합니다.

J2EE Application Server 지원

J2EE 구조((http://java.sun.com/j2ee/download.html#platformspec)의 J2EE 플랫폼 사양 참조)에서 EJB 컨테이너는 J2EE Application Server가 호스팅합니다. Application Server는 트랜잭션 관리자, 지속성 관리자, 이름 서비스, JMS 공급자(메시징 및 MDB의 경우) 등과 같은 다양한 컨테이너가 필요로 하는 자원을 제공합니다.

Sun Java System Application Server에서 JMS 메시징 자원은 Sun Java™ System Message Queue가 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 내장 JMS 자원 어댑터로 플러그 인되어 있습니다.
- Application Server의 향후 릴리스에서는 Message Queue가 표준 자원 어댑터 배포 및 구성 방법을 사용하는 Application Server에 플러그 인됩니다.

JMS 자원 어댑터

*자원 어댑터*는 J2EE 1.4를 준수하는 Application Server에 추가 기능을 플러그 인하는 표준화된 방법입니다. 표준은 J2EE Connector Architecture(J2EECA) 1.5 사양에 의해 정의됩니다. 이 구조를 통해 J2EE 1.4를 준수하는 Application Server는 표준 방식으로 외부 시스템과 상호 작용할 수 있습니다. 외부 시스템은 다양한 EIS(Enterprise Information System)를 비롯하여 JMS 공급자와 같은 다양한 메시징 시스템을 포함할 수 있습니다. Message Queue는 Application Server가 Message Queue를 JMS 공급자로 사용할 수 있도록 하는 JMS 자원 어댑터를 포함합니다.

J2EECA 1.5를 통해 가능하게 된 표준화된 상호 작용으로는 다양한 종류의 Message-Driven Bean 컨테이너에 대한 지원을 비롯하여 연결 풀링, 스레드 풀링, 트랜잭션 및 보안 컨텍스트 전파를 들 수 있습니다. 또한 이 사양은 연결 팩토리 및 기타 관리 객체를 만드는 표준화된 방식도 포함하고 있습니다.

JMS 자원 어댑터를 Application Server에 플러그 인하면 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 메시지 서버 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, 아직 다른 J2EE 1.4 Application Server에서는 인증되지 않았습니다.

Message Queue 자원 어댑터는 단일 파일(`imqjmsra.rar`)이며, 이 파일이 있는 디렉토리는 운영 체제에 따라 다릅니다(*Message Queue 관리 설명서* 참조). `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.

부록 A

선택적 JMS 기능의 Message Queue 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 방법을 설명합니다.

이 자료는 응용 프로그램 개발자와 가장 관련이 있습니다.

각 JMS 선택 항목 처리는 표 A-1에 나타나 있습니다.

표 A-1 선택적 JMS 기능

JMS 사양 섹션	설명 및 Message Queue 처리
3.4.3 `JMSMessageID`	"메시지 ID는 메시지 작성 및 메시지의 크기 증가에 어느 정도 영향을 미치기 때문에 응용 프로그램에서 메시지 ID를 사용하지 않는다는 힌트가 있을 경우 일부 JMS 공급자가 메시지 오버헤드를 최적화할 수 있습니다. JMS Message Producer는 메시지 아이디 비활성화를 위한 힌트를 제공합니다." **Message Queue 구현**: 제품은 메시지 ID 생성을 비활성화하지 않습니다(`MessageProducer`의 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 공급자는 이 관리 옵션을 지원하지 않아도 됩니다." **Message Queue 구현**: Message Queue 제품은 클라이언트 런타임 구성을 통해 메시지 헤더 필드 값의 관리 대체를 지원합니다 (41페이지의 "메시지 헤더 값 대체" 참조).

표 A-1 선택적 JMS 기능*(계속)*

JMS 사양 섹션	설명 및 Message Queue 처리
3.5.9 JMS 정의 등록 정보	"JMS는 JMS 정의 등록 정보에 대한 'JMSX' 등록 정보 이름 접두어를 예약합니다." "별도로 명시되지 않은 경우 이러한 등록 정보 지원은 선택 사항입니다." **Message Queue 구현**: JMS 1.1 사양에 정의된 JMSX 등록 정보는 Message Queue 제품에서 지원합니다(*Message Queue 관리 설명서* 참조).
3.5.10 공급자별 등록 정보	"JMS는 공급자별 등록 정보에 대한 '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. 이러한 기능은 JMS간 메시징에 사용되지 않습니다. Message Queue 3은 공급자별 등록 정보를 사용하지 않습니다.
4.4.8 분산 트랜잭션	"JMS에서는 공급자가 분산 트랜잭션을 지원하지 않아도 됩니다." **Message Queue 구현**: 분산 트랜잭션은 Message Queue 제품의 본 릴리스에서 지원됩니다(65페이지의 "트랜잭션" 참조).
4.4.9 다중 세션	"PTP <지점간 배포 모델>의 경우 JMS는 동일한 대기열에 대한 동시 QueueReceivers의 의미를 지정하지 않지만, 공급자가 이를 지정하는 것을 금지하지는 않습니다." 자세한 내용은 JMS 사양의 5.8절을 참조하십시오. **Message Queue 구현**: Message Queue 구현에서는 다중 사용자로의 대기열 전달을 지원합니다. 자세한 내용은 61페이지의 "다중 사용자로의 대기열 전달"을 참조하십시오.

용어집

이 용어집에서는 Message Queue 사용 중에 접할 수 있는 용어와 개념에 대한 정보를 제공합니다.

관리 대상 객체 사전 구성된 객체로서, 공급자별 구현 세부 정보를 캡슐화하고 관리자가 작성하여 하나 이상의 JMS 클라이언트가 사용하는 연결 팩토리 또는 대상. 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 액세스를 인증할 수 있도록 Message Queue 클라이언트의 사용자가 속하는 그룹

Queue 관리자가 지점간 전달 모델을 구현하기 위해 생성하는 객체. 메시지를 사용하는 클라이언트가 비활성 상태이더라도 대기열은 항상 메시지 보관이 가능합니다. 대기열은 생성자와 사용자 사이의 중간 저장소 역할을 합니다.

대상 생성된 메시지가 경로 지정 및 이후 사용자로의 전달을 위해 이동하는 Message Queue 메시지 서버상의 물리적 대상. 물리적 대상은 클라이언트가 자신이 누구를 위해 메시지를 생성하며 누구로부터 받은 메시지를 사용하는지 그 대상을 지정할 때 사용하는 관리 대상 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. 지점간 전달 모델을 위한 도메인과 게시/가입 전달 모델을 위한 도메인 등 두 가지 유형의 프로그래밍 도메인이 있습니다.

메시지 메시징 클라이언트가 사용하는 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 서버 클라이언트와의 연결, 메시지 처리 및 경로 지정, 지속성, 보안 및 모니터링을 포함하여 Message Queue 서비스에 대해 중앙 집중식 전달 서비스를 제공하는 하나 이상의 브로커. 메시지 서버는 생성자 클라이언트가 보내는 메시지를 받고 또한 사용자 클라이언트에게 그 메시지를 전달하는 물리적 대상을 관리합니다.

메시지 서비스 분산 구성 요소 또는 응용 프로그램 간에 안정적인 비동기식 메시지 교환을 제공하는 미들웨어 서비스. 메시지 서버, 클라이언트 런타임 및 메시지 서버가 자체 기능을 수행하는 데 필요한 여러 데이터 저장소가 포함됩니다.

메시징 엔터프라이즈 응용 프로그램이 사용하는 비동기 요청, 보고서 또는 이벤트 시스템으로서, 느슨하게 연결된 응용 프로그램들이 신뢰성 있고 안전하게 정보를 전송할 수 있게 합니다.

브로커 메시지 경로 지정, 전달, 지속성, 보안 및 로깅을 관리하고 성능 및 자원 사용을 모니터하고 조정할 수 있는 인터페이스를 제공하는 Message Queue 실체

비동기식 메시징 메시지 전송이 메시지를 수신할 사용자가 준비되었는지 여부에 의존하지 않는 메시지의 교환. 즉, 메시지 발신자가 발신 메소드가 반환될 때까지 기다릴 필요 없이 다른 작업을 진행할 수 있습니다. 메시지 사용자가 작업 중이거나 오프라인 상태인 경우 사용자가 준비되었을 때 메시지가 전송된 다음 수신됩니다.

사용 불능 메시지 정상 처리 또는 명시적 관리자 조치가 아닌 다른 이유로 해서 시스템에서 제거된 메시지. 메시지는 만료되었거나, 메모리 제한이 넘쳐서 대상에서 제거되었거나, 전달 시도 실패로 인해 사용 불능으로 간주될 수 있습니다. 사용 불능 메시지 대기열에 사용 불능 메시지를 저장할 것을 선택할 수 있습니다.

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

사용자 대상으로부터의 메시지 수신에 사용되는 세션에서 작성한 객체(MessageConsumer). 지점간 전달 모델의 사용자는 수신기 또는 브라우저(`QueueReceiver`나 `QueueBrowser`)이고, 게시/가입 전달 모델의 사용자는 가입자(`TopicSubscriber`)입니다.

생성자 세션에서 생성한 객체(메시지 생성자)로서 대상에게 메시지를 보낼 때 사용합니다. 지점간 전달 모델에서 생성자는 발신자(QueueSender)이며, 게시/가입 전달 모델의 생성자는 게시자(TopicPublisher)입니다.

선택기 메시지를 정렬하고 경로 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.

암호화 연결을 통한 전달 중 메시지가 훼손되지 않게 하는 메커니즘.

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

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

전달 모델 메시지가 전달되는 모델로서, 지점간 모델 또는 게시/가입 모델이 있습니다. JMS에서는 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 차례 전달되어 성공적으로 사용되는지(지속성 전달 모드) 또는 최대 1회 전달되는지(비지속성 전달 모드) 여부

Topic 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 다른 클라이언트와 상호 작용하면서 메시지 서비스를 사용하여 메시지를 교환하는 응용 프로그램(또는 소프트웨어 구성 요소). 클라이언트는 생성자 클라이언트나 사용자 클라이언트 또는 둘 다 될 수 있습니다.

클라이언트 런타임 메시징 클라이언트에게 Message Queue 메시지 서버와의 인터페이스를 제공하는 Message Queue 소프트웨어. 클라이언트 런타임은 클라이언트가 대상에게 메시지를 보내고 대상으로부터 메시지를 받는 데 필요한 모든 작업을 지원합니다. 클라이언트 런타임은 ConnectionFactory 등록 정보를 설정하여 구성됩니다.

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

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

트랜잭션 완료하거나 완전히 롤백해야 하는 작업 기본 단위

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

인증 검증된 사용자만 메시지 서버에 연결할 수 있게 하는 과정

JMS 공급자 메시징 시스템을 위한 JMS 인터페이스를 구현할 뿐만 아니라 해당 시스템을 구성하고 관리하는 데 필요한 관리 및 제어 기능을 추가한 제품

색인

가

다

라

마

바

사

아

자

카

타

파

하

A

C

D

H

I

J

L

M

N

S

T

X