

분산 시스템 구성관리를 위한 XML 매핑 프레임워크의 설계 및 구현

윤태웅, 민덕기
건국대학교 컴퓨터·정보통신공학과
{taewoong, dkmin}@konkuk.ac.kr

Design and Implementation Of XML Mapping Framework For Configuration Management in Distributed Systems

Taewoong Yun, Dugki Min
Department of Computer Science and Engineering, Konkuk University

요 약

분산 시스템 구성관리를 위해서는 여러 가지 구성환경정보를 로컬 파일 시스템에 저장하거나 네트워크를 통해서 다른 노드에 전달하는 방법이 필요하다. 본 논문에서는 구성관리를 위한 자바 객체와 XML을 사용한 분산시스템의 구성관리를 위한 영속성, 구조화, 재사용성을 가진 구성관리 프레임워크를 제시한다. 기존 연구에서는 문자열 타입만 제공하거나 자바 언어의 모든 타입을 빈즈 규약에 따라서 접근하였으나 본 연구에서는 재사용성과 성능향상을 위해서 구성관리에서 사용되는 일반타입, 리스트타입, 중첩타입으로 이루어진 구성정보 모델을 사용하였다. 또한 JDK버전과 XML파서 방식에 따른 성능 분석 결과를 제시한다. 또한 분산 시스템에서 이벤트 기반의 메시징 시스템의 이벤트를 전달하는 표준적인 API로 사용될 수 있다.

1. 서 론

분산 시스템의 관리에는 구성관리, 오류 관리, 성능 관리, 계정 관리, 보안관리 등 많은 영역이 있다[1]. 그 중에서 구성관리란 시스템에 존재하는 여러 가지 구성정보 및 환경정보를 관리하는 것을 말한다. 이러한 정보를 관리함에 있어서 파일을 기반으로 하는 방법에는 속성(Properties)파일[9]을 사용하거나 XML파일을 사용하는 방법[10]이 있다.

XML파일을 사용하는 경우 기존의 관리 어플리케이션에서는 XML파서를 사용하여 각 어플리케이션에 의존적인 파싱 로직을 구현하였다[10]. 이러한 방법의 문제점은 다른 관리 어플리케이션의 구성관리를 위해서는 다시 XML파서를 사용해서 또 다른 파싱 로직을 구현해야 한다는 점이다.

본 논문에서는 분산 시스템에서의 구성관리의 특징인 영속성, 구조성, 재사용성에 대해서 분석하고 기존의 연구되었던 구성관리 방법의 단점인 재사용성과 성능저하의 문제점을 해결하는 일반타입, 리스트타입, 중첩타입을 가진 구성관리 정보 모델을 제시하고 이를 위한 구성관리 프레임워크를 제시한다. 또한 자바JDK버전별 성능과 SAX기반 파서를 사용한 본 프레임워크의 성능향상 버전을 테스트 하였다.

이 논문의 구성은 2장에서는 분산 시스템의 구성관리의 특징에 대해서 분석하고 3장 관련연구에서는 객체 대 XML매핑의 방법론으로 기존에 연구되었던 SOAP프로토콜의 객체 바인딩, JDK 1.4.1의 자바빈즈 XMLEncoder, XMLDecoder와 IBM의 JiBX에 대해서 비교분석한다. 4장에서는 논문에서 제시하는 구성관리 프레임워크에 대해서 제시한다. 5장에서는 프레임워크의 성능 비교분석하며 SAX기반 파서를 통한 XML Decoding의 성능향상 방안을 제시하고 변형된 프레임워크를 제시하고 성능 분석을 하는 것으로 한다.

2. 분산 시스템 구성관리의 특징

분산 시스템 구성관리의 특징에는 영속성을 가져야 하며 구조

적이어야 하고 모든 구성관리 정보를 표현가능 해야 한다.

구성정보에는 시스템의 정적인 정보와 시스템의 상태를 표현하는 동적인 정보를 포함한다. 이러한 정보는 파일 시스템을 기반으로 영속성을 지켜야 하는 특징이 있다. 즉, 특정 노드가 관리하는 구성정보가 노드가 정지되었다가 다시 시작했을 경우 이전 상태를 복원해야 한다.

구성정보를 저장하는 파일에는 키와 값의 쌍으로 저장되는 속성(Property)파일이 있고 Structure기반의 XML파일 저장방법이 있다. Properties파일을 간단한 구성정보관리에는 효율적이지만 내부 정보의 재사용, 복잡한 관계를 표현하기에는 어렵다. XML파일은 속성파일에 비해 여러 가지 구조를 표현가능하다는 장점이 있으나 XML파서를 부가적으로 사용해야 하는 단점이 있다. 확장성이 중요한 분산 시스템에서는 XML파일기반을 주로 사용한다.

분산 시스템 구성관리의 또 다른 특징은 한번 작성된 구성관리 프레임워크를 다른 노드나 시스템에 사용하기 쉬운 재사용성을 가져야 한다. 재사용성이라는 특징은 구성관리가 관리하려는 정보에 독립적이어야 한다는 뜻이다. 관리하려는 정보가 어떠한 형태를 가져도 관리 가능한 구조가 되어야 한다.

3. 관련 연구

3.1 SOAP 프로토콜

SOAP 프로토콜[2]은 XML기반의 웹서비스에서 서비스 요청, 응답을 위한 프로토콜이다. 웹 메서드 호출에 있어서 파라미터가 객체일 경우 객체를 XML로 매핑을 지원한다. SOAP에서의 XML매핑은 기본 데이터 타입은 물론 구조체, 배열 그밖에 다양한 타입을 지원한다는 장점이 있다.

단점으로는 객체 대 XML매핑을 위한 프로토콜이 아닌 메서드 호출을 위한 프로토콜이기 때문에 XML구조가 복잡하고 사이즈가 큰 단점이 있다. 결론적으로 한정된 타입(기본타입, 벡터, 해쉬테이블 등등)만 필요로 하는 구성관리를 위한 프레임워크

로는 부적합하다. SOAP프로토콜 보다는 간단하면서 구성관리에 맞는 방법이 필요하다.

3.2 JDK1.4.1 자바빈즈의 XMLEncoder,XMLDecoder
 자바JDK1.4.1에 내장된 java.beans.XMLEncoder[3], java.beans.XMLDecoder[4]의 장점으로는 자바의 기본 데이터타입이외에 클래스등 자바 내장 타입을 전부 취급한다. 또한 XML으로 매핑하려는 객체가 자바 빈즈 규약을 지킬 경우 어떠한 객체라도 상태를 저장해 주는 장점이 있다.

단점으로는 자바 내장 타입을 전부 취급하기 때문에 성능이 낮은 단점이 있다. 그리고 분산시스템에서 사용되는 구성정보 클래스의 특성상 get,set의 프로퍼티 메서드를 가지지 않는 경우가 많은데 이러한 경우 get,set 프로퍼티 메서드를 부가적으로 코딩해야 하는 번거로움이 있다. 또한 매핑 이후 생성되는 XML사이즈가 본 논문에서 제시하는 XML파일 사이즈 보다 크다. 네트워크를 통해서 객체를 XML로 매핑시켜서 전달할 경우 XML파일의 크기가 중요하다.

3.3 IBM의 JiBX

JiBX[5]는 XML의 DTD로부터 자바객체로의 매핑을 지원하는 구조이다. 다른 파서를 이용한 객체 매핑의 경우 보다 빠른 성능을 가진다. 구성관리를 위한 프레임워크로 쓰이기에 적합한 구조이지만 매핑되는 데이터 타입이 스트림과 객체로 한정되어 있어서 실제 구성관리에서 사용되는 스트링을 제외한 다른 타입을 매핑 할 수 없기 때문에 구성관리에서 사용하기에는 부적합하다.

4. 구성관리 프레임워크

4.1 구성관리 방법

본 논문에서 제시하는 구성관리 방법은 자바 객체와 XML을 사용하는 방법이다. 어플리케이션에서는 자바 객체를 사용해서 구성정보에 접근하고 영속성을 위해서 이 자바 객체를 XML으로 저장한다. 이러한 자바 객체를 XML으로 변환하는 과정을 이후부터는 XML매핑이라고 부른다. 기존의 XML처리에서는 이러한 공통되고 표준화된 방법이 존재하지 않았기 때문에 매개마다 어플리케이션에서 필요로 하는 문법대로 XML을 파싱하는 파서를 개발해야 하는 단점이 있었다. 구성관리 프레임워크에서는 이러한 매핑 과정을 분리시켜 자동화하고 객체라는 인터페이스를 노출시키고 있다.

구성관리 프레임워크를 위한 객체 대 XML매핑을 그림 1과 같이 변환기(Transformer)에 의해서 이루어진다. 그림 1의 왼쪽 아래와 같이 객체로 만들어진 구성관리 정보 객체를 구성관리 프레임워크가 XML로 매핑 혹은 역변환 한다. 여기서 객체 이름과 속성 이름을 각각 Tag에 매핑되며 속성의 값은 XML의 Attribute로 매핑된다.

4.2 구성정보 모델

분산 시스템에서 사용하는 구성정보는 일반타입, 리스트 타입, 중첩 타입으로 표현가능하다. 일반 타입을 구성정보 클래스의 일반 타입으로 선언된 속성을 의미한다. 자바 언어의 기본타입(primitive)과 기본타입의 래퍼(wrapper)클래스를 말한다. 일단 타입인 경우는 각각 XML의 속성이름이 태그이름 속성 값이 태그 값으로 매핑된다.

리스트 타입은 일반타입의 리스트를 가진 벡터나 해쉬테이블로 구성되는 구성정보를 말한다. XML매핑 과정에서 Vector라는 태그 안에 Element태그로 쌓여진 일반 타입으로 매핑된다.

중첩 타입은 객체안에 다른 객체가 중첩된 타입이다. XML의 특성상 중첩구조를 잘 표현하므로 XML태그의 요소중첩으로 표현된다.

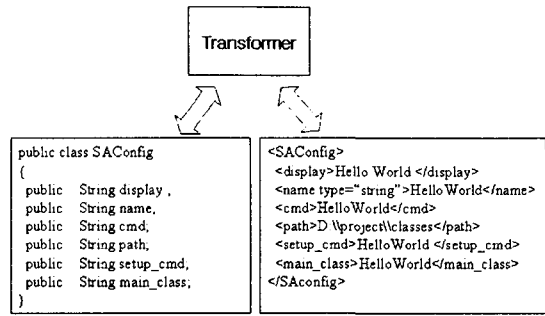


그림 1 객체 대 XML 매핑

4.3 구성정보 클래스 요구사항

구성관리 정보를 객체로 표현하는데 사용되는 구성정보 클래스는 다음과 같은 요구사항을 만족해야 한다. 1) 변환 가능한 표시(Markup) 인터페이스 Transformable을 상속받아야 한다. 2) 속성은 데이터 타입(Primitive), Wrapper 클래스, 일반 클래스, 벡터, 해쉬테이블만 허용한다. 4.2에서와 같이 구성정보의 특징상 위의 데이터 타입으로 표현 가능하다. 3) 속성의 변경자(Modify)는 리플렉션이 접근할 수 있는 public으로 선언되어야 한다. 4) "VECTOR", "ELEMENT", "START_CLASS", "configuration"은 XML태그 이름으로 미리 예약되어있으므로 속성이름으로 사용할 수 없다.

4.4 구성관리 프레임워크 설계

이러한 객체 대 XML매핑을 하는 변환기를 위한 프레임워크는 그림 2와 같다.

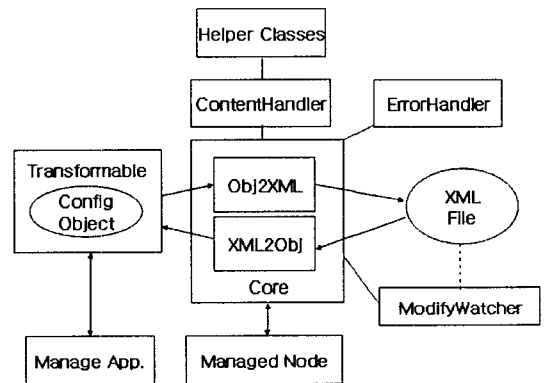


그림 2 구성관리 프레임워크

구성관리를 위한 프레임워크의 내부는 XML과 객체를 매핑 혹은 그 역변환을 하는 Core부분과 변환하려는 Transformable객체와 SAX기반의 XML 파서를 위한 ContentHandler와 ErrorHandler가 있다. SAX방식의 파서를 사용함에 있어서 XML 내부 속성의 기본 타입과 래퍼타입의 구분을 위해서 Helper 클래스가 사용된다. 또한 XML로 매핑된 정보를 변경 시 그 변경을 프레임워크 혹은 관리하는 어플리케이션에게 이벤트를 전달하기 위한 ModifyWatcher가 존재한다. 본 프레임워크에서는 SAX[6]기반의 XML파서와 DOM기반 XML파서를 선택해서 사용할 수 있는 구조이며 XML파일에서 객체를 역변환하는 기능

외에 분산시스템에서 각 노드의 관리정보를 전달할 수 있는 스트링기반의 XML에서 객체를 매핑하거나 역 변환하는 기능도 포함한다.

5. 테스트 및 성능분석 결과

본 논문의 프레임워크에 대한 성능분석은 자바 언어의 기본타입(primitive type)과 래퍼타입과 벡터와 해쉬테이블로된 중첩타입을 포함한 객체를 대상으로 encoding, decoding시간을 자바 버전별, SAX기반 XML파서와 DOM기반의 XML파서를 사용한 경우로 비교하였다. 테스트 XML파일 사이즈는 3KBytes이다. 테스트 환경은 펜티엄3-1G, 256M, Windows 2000 Server, JDK 1.3.1이다.

5.1 파일기반 Encoding 시간 비교

그림 3에서는 1)은 JDK1.3.1에서 본 프레임워크를 사용한 경우 구성정보 객체를 XML파일로 Encoding하는 경우, 2)는 JDK 1.4.1에서 본 프레임워크를 사용한 경우, 3)은 3.2 관련연구에서의 자바빈즈의 XMLEncoder를 사용한 경우를 비교하였다. 테스트 결과 본 프레임워크는 JDK1.3.1에서 최상의 성능 보인다. JDK1.4.1는 자체 라이브러리의 크기가 증가와 1),2)의 경우는 객체의 속성을 바로 접근해서 빠른 반면 3)의 경우는 객체의 속성을 빈즈 규약에 따라서 get에서 접근하기 때문에 낮은 성능을 보인다. 또한 3.2관련연구에서처럼 1.4.1에 java.beans패키지에 있는 방법은 모든 자료형을 다루기 때문에 성능이 낮게 측정되었다.

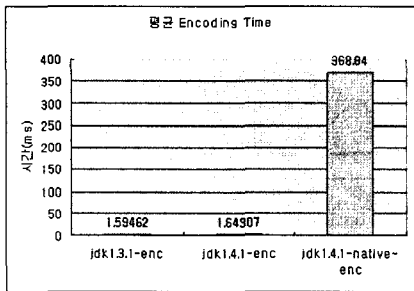


그림 3 JDK버전별 평균 Encoding 시간

5.2 파일기반의 Decoding 시간 비교

그림 4는 그림3과 같은 환경에서의 Decoding시간을 비교한 결과이다. Decoding시에 XML파서의 영향을 받는다. 역시 JDK1.3.1에서 최상의 성능을 보인다. 1)에서는 crimson파서 중 필요한 패키지만을 로딩해서 사용한 결과 JDK1.4.1에서의 XML파서 보다 좋은 성능을 보인다.

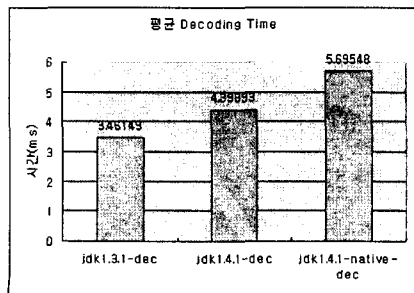


그림 4 JDK버전별 평균 Decoding 시간

5.3 DOM기반과 SAX기반 Decoding 시간 비교

그림3,4는 DOM기반의 XML파서를 사용한 경우이나 그림 5는 보다 빠른 성능을 위해서는 이벤트 방식의 SAX기반 파서를 사용해야 한다. SAX기반의 파서 선택에 있어서 기존의 jaxp-1.1(crimson parser[7]) 대신에 JWSDP-1.2의 JAXP(xerces parser[8]) 사용을 사용한 결과 0.5millisecond정도 향상된 성능을 보였다.

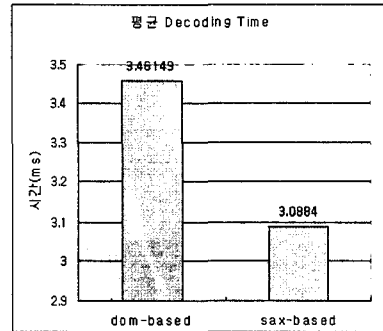


그림 5 XML방식별 평균 Decoding Time

7. 결론

본 논문에서는 분산 시스템에서 관리되는 구성관리 정보를 로컬에 저장하거나 이동된 다른 노드에 구성관리 정보를 전달하기 위한 XML매핑 프레임워크의 필요성 및 영속성, 구조성, 재사용성을 가진 프레임워크를 제시하였다. 문자열타입만 지원하는 IBM의 JiBX의 단점을 보완하고 자바언어의 모든 타입을 지원하는 SOAP프로토콜 및 자바빈즈 XMLEncoder,XMLDecoder의 단점을 보완한 구성정보관리를 위한 제한된 타입을 지원하는 구성정보 모델을 제시하였다. 또한 자바 JDK버전별 성능분석과 성능향상을 위한 방안으로 SAX기반 파서를 사용한 경우를 구현 및 테스트 하였다. 본 프레임워크를 사용해서 구성정보 관리에 있어서 XML파서 API를 알지 않더라도 XML매핑 혹은 그 역변환을 할 수 있게 되며 구성정보를 위한 XML표준 스키마를 제공한다. 또한 분산 시스템에서 이벤트 기반의 메시지 시스템의 이벤트를 다른 노드에게 XML형태로 전달하는 표준적인 API로 사용될 수 있다.

8. 참고 문헌

- [1] Heinz-Gerd Hegering, Integrated Management of Network Systems, Morgan Kaufmann Publishers, 1998.
- [2] Don Box, "Simple Object Access Protocol (SOAP) 1.1" available at URL :<http://www.w3.org/TR /SOAP/>
- [3] Java.beans.XMLEncoder available at "URL : <http://java.sun.com /j2se/1.4.1/docs/api/java/beans/XMLEncoder.html>"
- [4] Java.beans.XMLDecoder available at URL "<http://java.sun.com/j2se/1.4.1/docs/api/java/beans/XMLDecoder.html>"
- [5] Dennis M. Sosnoski, "XML and Java technologies: Data binding Part 3: JiBX architecture" available at URL: <http://www-106.ibm.com/developerworks/xml/library/x-databd3/>
- [6] SAX available at URL "<http://java.sun.com/webservices/docs/ea2/tutorial/doc/JAXPIntro4.html>"
- [7] Crimson Parser available at URL "<http://xml.apache.org/crimson/>"
- [8] Xerces Parser available at URL "<http://xml.apache.org/xerces2-j/index.html>"
- [9] Properties available at URL : "<http://java.sun.com/j2se/1.4.2/docs/api/java/util/Properties.html>"
- [10] Resin available at URL "<http://www.caucho.com/>"