Gazi Üniv. Müh. Mim. Fak. Der.
Cilt 24, No 4, 629-639, 2009

J. Fac. Eng. Arch. Gazi Univ.
Vol 24, No 4, 629-639, 2009

HİYERARŞİK BİR VERİTABANININ XML İLE MODELLENMESİ

Özgür YÜREKTEN ve Hasan Şakir BİLGE*
TÜBİTAK-UEKAE / G222 Birimi, 06680, Ankara
*Bilgisayar Mühendisliği, Mühendislik Fakültesi, Gazi Üniversitesi, 06570, Ankara
ozgur.yurekten@tubitak.gov.tr, bilge@gazi.edu.tr

(Geliş/Received: 24.06.2009 ; Kabul/Accepted: 23.10.2009)

ÖZET

Elektronik harp alanında kullanılan verilerin bulunduğu en büyük veri kaynağı EWIRDB kısaltmasıyla anılan Elektronik Harp Bütünleşik Tekrar Programlanır Veritabanı'dır. Bu veri tabanındaki veriler metin dosyaları şeklinde saklanmakta ve buradan okunarak yönetilmektedir. Bu hiyerarşik veri tabanı, her biri 2400'den fazla alandan oluşan kayıtları saklayabilmekte, dinamik olarak genişleyebilmekte ve belirli bir şablona göre eksik bilgiler içerebilmektedir. İlişkisel ve nesne-yönelimli veritabanları ile bu tür verilerin modellenmesi zor olduğu için miras bir sistem olarak durmakta ve verileri halen günümüzde metin dosyalarında saklanmaya devam etmektedir. İlişkisel ve nesne yönelimli veritabanları ile bu kadar büyük bir yapı modellendiğinde, bu modeli kullanan yazılımların geliştirilmesi karmaşıklaşmakta ve ortaya çıkan ürün kullanışsız olmaktadır. Bu çalışmada, EWIRDB veri tabanının daha etkin yönetimi ve kullanımının sağlanması hedeflenmiştir. Etkin yönetim için veriler, ilişkisel veya nesne-yönelimli veritabanlarına alternatif olarak, hiyerarşik yapıların yönetimi konusunda üstün özelliklere sahip XML veritabanına taşınmıştır.

Anahtar Kelimeler: Hiyerarşik veritabanı, XML veritabanı.

MODELING A HIERARCHICAL DATABASE WITH XML

ABSTRACT

Electronic Warfare Integrated Reprogramming DataBase (EWIRDB) is the largest data source in the area of electronic warfare. The information in this data source is stored and managed as plain files. The structure of this data source has the following characteristics; dynamic data expandability, ability to store over 2400 parameters, ability to accommodate incomplete data, supporting hierarchically modeled structure. It is difficult to design hierarchical data structure using the relational or object oriented database system; therefore EWIRDB remains as a legacy system and its data is still stored in plain files. When this large structure is modeled using the relational or object oriented databases, the complexity of development increases and the out-coming product is hard to use. The goal of this study is to enable EWIRDB data source to be managed and used more effectively. As an alternative of relational and object oriented database, XML database, which has superior capability of managing hierarchical data structures, is used for better management of data.

Keywords: Hierarchical data, XML database.

1. GİRİŞ (INTRODUCTION)

EWIRDB (Electronic Warfare Integrated Reprogramming DataBase), elektronik harp alanında 1970 yılından beri kullanılan, günümüzde de gelişerek kullanılmaya devam eden ve verileri hiyerarşik yapıda saklayan büyük bir veri kaynağıdır. Bu veri kaynağının verileri metin dosyaları şeklinde saklanmaktadır [1]. EWIRDB artık miras bir bilgi sistemidir. Bu gibi miras sistemler yeni ve düzenli olarak değişen şartlarda yeni iş isterlerinin karşılanmasına karşı direnç göstermektedir [2]. Bu sistemlerin yenileri ile değiştirilebilmesi için hem verilerin hem de yazılımların taşınması gerekmektedir [3, 4]. Miras sistemlerin veritabanları genellikle görev kritik veritabanlarıdır; yani her zaman kullanılabilir olmak zorundadırlar. Bunlar genellikle ilişkisel veritabanı kullanmazlar, ağ veya hiyerarşik veritabanı şeklinde veya düz dosya-

larda saklanmaktadırlar [5]. Geçmişte ilk olarak hiyerarşik veritabanları kullanılmasına rağmen, EWIRDB verileri hiyerarşik veritabanı olarak modellenmemiştir. Bunun yerine veriler düz dosyalarda saklanmıştır.

Bu çalışmada, EWIRDB verilerinin daha etkin yönetiminin ve kullanımının sağlanması hedeflenmiştir. Bu amaçla alternatif veritabanı yönetim sistemleri incelenmiştir. Bu konudaki diğer çalışmalara [1, 6-9] alternatif olarak verilerin yapısına en uygun olan XML (Extensible Markup Language) veritabanı [10] ile modelleme yapılmış ve bu veritabanının XML üzerinden yönetimi sağlanmıştır.

XML veritabanının seçilmesinde EWIRDB verilerinin hiyerarşik yapıda olması, dinamik olarak genişleyebilmesi ve 2400'ün üzerinde parametresi olan ağaçlar olarak ifade edilmesi [1, 6-9] öncelikli faktör olmuştur. EWIRDB verilerinin ilişkisel ve nesne yönelimli veritabanı yönetim sistemleri ile modelleme alternatifleri de incelenmiş, fakat yukarıda belirtilen faktörleri tam olarak destekleyemedikleri veya etkin bir çözüm sunamadıkları görülmüştür.

EWIRDB verileri sadece operasyonel olarak elektronik harpte kullanılmamakta, ayrıca elektronik harp sistemlerinin araştırılmasında, geliştirilmesinde, testlerinde, değerlendirilmesinde, modellenmesinde, elektronik harp sistemi simülasyonlarında ve eğitimlerinde de kullanılmaktadır [1]. Fakat EWIRDB veri dosyalarının anlaşılmasında ve kullanımında zorluklar bulunmaktadır [6]. Bunun için bu verilerin etkin kullanımına ihtiyaç duyulmaktadır.

2. MEVCUT ÇALIŞMALAR (RELATED WORKS)

EWIRDB veritabanının tekrar modellenmesi konusunda literatürde çok az çalışma bulunmaktadır. Bu çalışmalarda EWIRDB veritabanının hiyerarşik modelini ilişkisel, nesne-yönelimli ve ağ modellerine nasıl dönüştürüleceği üzerinde durulmuştur. EWIRDB veritabanı yapısının belirli bir bölümü üzerinde yapılan araştırmaların sonucunda elde edilen modeller gerçekleştirilmeye ve değerlendirilmeye çalışılmıştır.

Barnes [9] EWIRDB veritabanı yapısının nesne yönelimli olarak modellenmesi için yaptığı çalışmada veritabanı formatını açıklamıştır. Bu formatın nesne yönelimli olarak nasıl modellenebileceği konusunda veritabanının kavramsal analizini yapmıştır. Bu çalışmada EWIRDB yapısının seçilmesindeki nedenleri üç kategoride açıklamıştır:

- Bu veri tabanının elektronik harp için kritik bir veritabanı olması,
- Veritabanı yapısının etkin ve kullanılabilir olamaması,
- Veritabanının dosyalar içinde ve ilişkisel olarak saklanmasıdır.

Bu çalışmada veritabanının ilişkisel olarak tanımlanmış olduğu söylenmiş, fakat bu ilişkisel yapının standart ilişkisel yapı olmayıp süper sınıf, alt sınıf, genelleme, özelleştirme, soya çekim gibi nesne-yönelimli modellemenin özelliklerini de taşıyan gelişmiş ilişkisel model olduğu belirtilmiştir. Veritabanı, öncelikle özel olarak tanımlanmış gelişmiş ilişkisel modelle modellenmiş ve ardından nesne-yönelimli olarak modellenmesi yapılmıştır. Çalışmada veritabanının mantıksal olarak sadece bir kısmının modellenmesi hedeflenmiştir.

EWIRDB veritabanının tekrar modellenmesi konusundaki araştırmalardan biri de Coyne [1] tarafından yapılan çalışmadır. Bu çalışmada, Coyne EWIRDB veritabanının dosya yapısı, veritabanında bulunan verilerin nesne yönelimli olarak analiz edilmesi ve tasarlanması üzerinde durmuştur. EWIRDB yapısını, biçimini, kullanımını, bu verilerin nesnelerle nasıl ifade edilebileceğini ve nesnelerin birbirleri ile olan ilişkilerini tanımlamıştır. Nesneler ve aralarındaki ilişkileri tanımlarken nesne-yönelimli programlamanın soya çekim, genelleştirme, özelleştirme ve birleştirme gibi yeteneklerinden faydalanmıştır. EWIRDB veritabanının nesne-yönelimli olarak modellenebilmesi için elektronik harp konusunda alan uzmanı olmayı gerektirdiğini belirtmiş ve kendi çalışması ile elde edilen nesne yönelimli model ile alan uzmanı olmayanların da EWIRDB veritabanının yapısını anlamalarını kolaylaştıracağını söylemiştir. Bu çalışma, sadece analiz ve tasarım aktivitelerini içerdiği için nesne yönelimli olarak EWIRDB veritabanı modeli tamamlanmamıştır. Buna karşın çalışmada ortaya konulan model ile tüm EWIRDB veritabanını nesne yönelimli olarak gerçekleştirilebileceği savunulmuştur.

Coyne [1]'nin çalışması sonucu elde ettiği nesne-yönelimli veritabanı modelini Lee ve McKenna [6] kendi çalışmalarında gerçekleştirmişler ve test etmişlerdir. Modelin gerçekleştirilmesi sırasında Coyne [1]'nin tanımladığı modelde bazı hatalar bulunmuşlar ve bu hatalar giderildikten sonra EWIRDB veritabanı yapısının ancak yüzde 20'si üzerinde çalışmalarına devam etmişlerdir. Çalışma sonucunda tüm veritabanının gerçekleştirilmediği, nesneler arası ilişkilerin basitleşmediği ve karmaşıklığın halen devam ettiği belirtilmiştir. Ayrıca nesne yönelimli olarak modelin gerçekleştirilmesinde çoklu soya çekim özelliklerinin kullanıldığı ve bunun karmaşıklığı arttırdığı üzerinde durulmuştur. EWIRDB veritabanı için tanımlanan nesneler arasında bulunan ilişkiler netleştirilerek veritabanı yapısının daha anlaşılabilir hale geldiği ifade edilmiştir. EWIRDB veritabanı yapısındaki bir radar yeni bir alan ekleme ve bir alana değer girebilme özellikleri gerçekleştirilmemiş sadece dokuz çeşit sorgu formatı tanımlanmış ve bu sorguları çalıştıran uygulama hazırlanmıştır. Bu dokuz sorgu formatının EWIRDB içindeki her türlü veriye ulaşmayı sağlamadığı, fakat veritabanının genel

kullanım amacının büyük bir çoğunluğunu karşıladığı belirtilmiştir.

Bu konudaki diğer bir araştırma da Werre ve Diehl [8]'nin çalışmasıdır. Werre ve Diehl çalışmalarında, Coyne [1]'nin oluşturduğu modeli ağ veritabanı modeline çevirmeyi ve elde edilen ağ modeli ile EWIRDB veritabanı yapısının bir bölümünü gerçekleştirmeyi amaçlamışlardır. Fakat gerçekleştirilen modelin istenilen şekilde testi yapılamamış, ağ modelinin nesne yönelimli, ilişkisel veya hiyerarşik modeller ile karşılaştırılması gerçekleştirilememiştir.

EWIRDB veritabanının yeniden modellenmesi üzerine yapılan Scrivener ve Edwards [7]'ın çalışması, EWIRDB veritabanının ilişkisel olarak modellenmesine yöneliktir. Bu çalışmada EWIRDB yapısı öncelikle ilişkisel tablolarla ifade edilmiştir. Bu aşamada hiyerarşik verinin birçok soya çekim, genelleme ve özelleştirme ile ifade edilmesi gerektiği ve bunun ilişkisel olarak ifadesinde zorlukların olduğunu belirtmişlerdir. Çalışmada EWIRDB veritabanı yapısının sadece bir bölümü üzerinde modelleme gerçekleştirilmiştir. Yedi çeşit sorguyu gerçekleştirebilecek şekilde bir yazılım hazırlanmış ve test edilmiştir. Çalışmada; EWIRDB veritabanı için ilişkisel veri modelinin nesne yönelimli modelden daha kullanışsız olduğu ve nesne yönelimli yaklaşımın daha etkili ve daha kolay olabileceği belirtilmiştir.

EWIRDB veritabanını hiyerarşik modelden ilişkisel, nesne yönelimli ve ağ modellerine taşınması için yapılan tüm çalışmalarda [1, 6-9] teorik ve pratik olarak gerçekleştirilebilirlik üzerinde durulmuş, fakat hiçbir çalışmada tüm EWIRDB veritabanı modellenmemiş ve veritabanı üzerinde istenilen sorgunun gerçekleştirilebilmesi olanağı sunulmamıştır. Bu çalışmalarda [1, 6-9] ilişkisel veritabanı modeli için SQL, ağ veritabanı modeli için CODASYL-DML (Conference on Data Systems Language, Data Manipulation Language) ve nesne yönelimli veritabanı modeli için de OO-DML (Object Oriented Data Manipulation Language) arayüzü kullanılmıştır. Çalışmalar tüm veri modellerini (İlişkisel, Nesne-Yönelimli, Ağ, Hiyerarşik, Fonksiyonel) ve tüm veritabanı arayüzlerini (CODASYL-DML, SQL, OO-DML) destekleyecek M2DBMS (Multi-lingual, Multi-model DBMS) adında bir sistem üzerinde test edilmiştir. Çalışmalarda; veritabanının standart bir arayüzle sorgulanabilmesini sağlamak yerine birden fazla arayüzü destekleyecek bir sistem ortaya çıkarmaya çalışılmıştır.

Farklı problem alanlarındaki hiyerarşik veritabanlarının XML veritabanı olarak modellenebileceğini belirten çalışmalardan biri Some ve arkadaşlarının çalışmasıdır [11, 12]. Çalışmada, NASA (National Aeronautics and Space Administration)'nın bilimsel misyonunu gerçekleştirmesine yardımcı olacak, değerli ve çığır açacak teknolojilerin seçiminde, bununla beraber bu teknolojileri prototip seviyesinden kullanılabilir seviyesine gelinceye kadar olgunlaştırılmalarını takip edecek XML veritabanı ve bu veritabanını kullanan bir görev yazılımı anlatılmaktadır. Çalışma, genel olarak görev yazılımının geliştirilmesinde XML veritabanının seçimindeki gerekçeler üzerinedir. Çalışmalarında şu anki teknolojilerin NASA için fonksiyonel ve yapısal olarak önemini, teknolojilerin şu anki yeteneklerini, NASA'nın fonksiyonel/yapısal isterlerini karşılamaları için sahip olmaları gereken kriterleri belirlemişler ve takipte kullanılacak veritabanını tasarlamışlardır. Problem alanını yukarıda belirtilen isterleri karşılamak için dört parçaya bölmüşlerdir. Öncelikle NASA'nın örgütlenme şemasının tanımlanmasını ve organizasyondaki bölümlerin birbirleri ile ilişkilendirilmesini sağlamışlardır. Daha sonra NASA'nın misyonunu bağımsız olarak fonksiyonel ve yapısal olarak ayrıştırmışlar, her fonksiyonun ve yapının isterlerini nicel parametrelerle ifade etmişlerdir. Takip edilecek teknolojileri tanımlamışlar ve nicel yetenekleri belirlemişlerdir. Çalışmada, tanımlanan veri yapılarının modellenmesi öncesinde, analiz araçları tarafından kullanılabilecek standart bir arayüzü destekleyebilmesi ve kullanıcılar için standart sorgulama / veri girişi arayüzü olanağı sağlayabilmesi gibi kriterler ön planda tutulmuştur.

Bütün yukarıda belirtilen kriterler dikkate alındığında, verilerin hiyerarşik olarak ifade edilebilmesi, tüm alanların insanlar tarafından okunabilir ve makine tarafından anlaşılabilir olması, internet üzerinden ulaşılabilmesi, kullanıcı arayüzlerinin açık, sezgisel olarak kavranabilen ve kullanıcı dostu olması, güvenli erişilebilmesi, veri yapısının derinliğinin istenildiği kadar arttırılabilmesi, hızlı erişilebilmesi ve tarihsel verilere erişilebilmesi gibi yeni isterlerin de karşılanmasının gerekliliği üzerinde durulmuştur. Tanımlanan veri yapılarının ilişkisel veritabanı ile modellenmesi durumunda, hiyerarşik verilerin ancak normalizasyonla ilişkisel olarak ifade edilebildiği, çok fazla alan olduğu için normalizasyon sonucu veri yapısının karışacağı (normalizasyonla çok fazla tablo oluşacaktır) ve okunaklılığın kaybolacağı, veriler hiyerarşik yapıda olduğu için hiyerarşiye yeni bir alan, yeni bir seviye eklemenin zor olacağı (yeni alan ve hiyerarşi eklemek tasarlanan tabloların yapısında değişikliğe sebep olacaktır) ve kullanıcı arayüzlerinde hiyerarşik yapıyı görüntülemek için çok fazla işlem yapılması gerekeceği gibi kısıtlar ve zorluklar olduğu anlatılmıştır. Buna karşılık tanımlanan veri yapılarının ilişkisel veri tabanı yerine XML veritabanı ile modellenmesi durumunda verileri hiyerarşik olarak saklanmanın ve ulaşılmanın daha kolay olduğu, insanlar tarafından okunabilen ve makine tarafından anlaşılabilen yapıların tanımlanabildiği, daha anlaşılır ve kolay şekilde hiyerarşik verilerin kullanıcı arayüzünde gösterilebildiği, yeni alanların ve seviyelerin eklenebildiği belirtilmiştir. Çalışma sonucunda,

veri yapılarına uygun bir XML veritabanı tasarlanmış, tasarlanan XML veritabanını kullanan bir görev yazılımı hazırlanmış ve XQuery sorgulama dili ile analiz araçlarının veritabanına ulaşıp kullanabilecekleri bir arayüz sağlanmıştır.

XML veritabanları öğrenci veritabanı [13], biyoloji veritabanı [14] ve internet sayfası içeriği veritabanı [15] gibi değişik alanlarda da kullanılmıştır. Askeri alanda da XML ve internet teknolojilerinin kullanılması gerektiği belirtilmiştir [16, 17].

3. HİYERARŞİK VERİ MODELLEME ALTERNATİFLERİ (MODELING ALTERNATIVES OF HIERARCHICAL DATA)

3.1. İlişkisel Veritabanı İle Modelleme (Modeling with Relational Database)

Hiyerarşik verilerin ilişkisel veritabanlarında saklanabilmesi için çeşitli alternatifler bulunmaktadır. Hiyerarşik verilerin tablosal yapılara dönüştürülmesi öncelikli problemdir. Bu problemin aşılması için ilişkisel veritabanı oluşturmak için tanımlanan yöntemlerin göz önünde bulundurulması gerekmektedir. İlişkisel veritabanı tasarımı gerçekleştirilirken dikkat edilecek hususlar ve uygulanacak basamaklar normalizasyon olarak adlandırılır [19, 20].

Hiyerarşik verilerin ilişkisel veritabanı ile tasarlanması için alternatif modeller aşağıda açıklanmıştır.

3.1.1. Hiyerarşik verilerin birinci normal form ile modellenmesi (Modeling with first normal form)

Hiyerarşik verilerin birinci normal form ile modellenmesi verilerin hiyerarşi seviyesine ve sahip olunan eleman sayısına bağlıdır. Bir veya iki seviye hiyerarşisi bulunan ve az sayıda elemanı bulunan hiyerarşik veriler tablosal formata çok kolay şekilde dönüştürülebilir. Bu işlem için tüm elemanların ve elemanların özelliklerinin veritabanında bir tabloda sütun olarak tanımlanması yeterlidir. Az elemanı bulunan hiyerarşik veriler tablosal formata dönüştürüldükten sonra tüm sütunların veri tipleri belirlenmeli ve veri tiplerinde kısıtlamalar tanımlandıysa bu kısıtları gerçekleştirmek için tetikleyiciler tanımlanmalıdır. Burada ortaya çıkan en büyük problem hiyerarşilerin nasıl yönetileceğidir. Bir seçenek olarak elemanların hiyerarşi bilgileri de tablolarda saklanabilir. Bu şekilde birici normal formun kriterleri hala korunmuş olur. Fakat hiyerarşilerin mantıksal olarak yönetimi tetikleyiciler veya dış yazılımlar tarafından yapılması gerekir [17]. Birinci normal formda bulunan tablonun her satırına bir hiyerarşik veri saklanır. Hiyerarşik verinin tablosal forma dönüştürülmesi ve tablosal formda bulunan verinin hiyerarşik veriye dönüştürülmesi ayrı bir işlem olarak durmakta ve bu işlem dış yazılımlar tarafından sağlanmaktadır.

Çok fazla hiyerarşisi ve elemanı bulunan hiyerarşik verilerin ilişkisel veritabanında saklanabilmesi için ilişkisel veritabanının kısıtları dikkate alınmalıdır. Örneğin birçok veritabanı, bir tabloda 255'ten fazla sütunun tanımlanmasına izin vermemektedir. Hiyerarşisi ve elemanı fazla veriler için doğrudan bir tabloya dönüştürmek mümkün olmayacaktır. Bunun için ikinci normal form kurallarına uygun veritabanı tasarlanması gerekmektedir. EWIRDB veritabanının birinci normal form ile modellenebilmesi imkansızdır [17]. EWIRDB veritabanında 2400'ün üzerinde parametre bulunmakta ve bu parametrelerin bir tabloya sığdırılabilme imkanı bulunmamaktadır.

3.1.2. Hiyerarşik verilerin ikinci normal form ile modellenmesi (Modeling with second normal form)

Az elemanı bulunan hiyerarşik veriler için ikinci normal form bir seçenek iken, fazla elemanı olan hiyerarşik veriler için bir zorunluluktur. Hiyerarşisi ve elemanı fazla olan bir veri kaynağının, ortak birincil anahtarları kullanarak birden fazla tabloya dağıtılması gerekmektedir. Her bir hiyerarşik veri için bir birincil anahtar tanımlanır. Veri veritabanının kısıtlarına ve mantıksal gereksinimlere uygun olarak birden fazla parçaya bölünebilir. Bu işlemden sonra hiyerarşik veri, her tabloda bir kaydı bulunacak şekilde tablolara dağıtılır. Burada en önemli nokta, hiyerarşik verilerin ilişkisel veritabanında saklanırken artık tek parça olarak değil dağınık olarak bulunmasıdır. Verinin bütününe ulaşmak için birçok tablonun sorgulanması gerekmektedir. Sorgulama sonucunda, bütün bilgiler birleştirilerek tekrar hiyerarşik yapıya dönüştürülmesi işi biraz daha problemli olmaktadır.

EWIRDB veritabanının ikinci normal form ile modellenebilmesi mümkündür. EWIRDB veritabanında 2400'den fazla parametrenin bulunduğu ve bir tabloya en fazla 255 sütun eklenebildiği varsayılırsa yaklaşık on tablonun tasarlanması gerekecektir. Burada verilerin hiyerarşik bilgilerinin saklanmadığı düşünülmüştür. Elemanların hiyerarşi bilgisinin de tablolarda saklanacağı varsayıldığında en az yirmi tabloya ihtiyaç duyulacaktır. EWIRDB veritabanında bulunan her bir alanın birden fazla değeri bulunabilmektedir. Bunu desteklemek için ikinci normal form yetersiz kalmaktadır [17]. Çünkü bir alana elle bir veri girmek için diğer bütün alanların tekrar edilmesi gerekmektedir. İkinci normal form ancak bir elemanı tekrarlamayan yapıda hiyerarşisi olan veriler için kullanılabilir.

3.1.3. Hiyerarşik verilerin üçüncü normal form ile modellenmesi (Modeling with third normal form)

Az elemanı olan hiyerarşik veriler birinci ve ikinci normal form ile modellenebildiği gibi, üçüncü normal form ile de modellenebilir [17, 21]. Üçüncü normal formda, tüm alanların birincil anahtar ile ilişkili olması gerekmektedir. Bu normal form tekrarlanan

verilerin yönetimi için uygundur. Eğer bir hiyerarşik verinin tekrarlanan elemanları var ise üçüncü normal form ile modellenmesi gerekmektedir. Üçüncü normal formun tekrarlanan alanların yok edilmesine yönelik olduğu düşünülürse hiyerarşik verilerin analiz edilerek beraber tekrarlanan elemanların bir tabloda bulunması gerekmektedir. Hiyerarşik verinin analiz edilebilmesi için de sahip olunan verinin elemanlarının hangi amaçla ve nasıl birbirini etkilediğini bilmeyi gerektirmektedir. Yani ilgilenilen veri hangi alanda ise o alanda uzmanlığı gerektirmektedir.

Verilerin üçüncü normal formla modellenebilmesi az elemanı bulunan hiyerarşik veriler için kolaydır. Fakat çok fazla elemanı olan hiyerarşik verilerin parçalanması, başka problemlerle karşı karşıya kalınmasına sebep olur. Parçalama işlemi sonucunda küçük fakat çok fazla tablonun ortaya çıkması kaçınılmazdır. Örneğin, EWIRDB veritabanının üçüncü normal form ile modellenebilmesi için tüm tekrarların giderilmesi, tüm tekrarların giderilmesi için de her bir alan ve alanın hiyerarşi bilgisini saklayan bir tablo oluşturulması gerekmektedir. Bu durumda EWIRDB veritabanını üçüncü normal formla modelleyebilmek için 2400'ün üzerinde tablo oluşturmak gerekecektir. Çok sayıda küçük tablolarda hiyerarşik verinin saklanması verinin artık okunamaz hale gelmesine sebep olmaktadır. Verinin ilişkisel veritabanında saklanabilmesi için de bu zorunludur. Bir hiyerarşik verinin veritabanından sorgulanabilmesi için çok fazla tabloya ulaşılması gerekecektir. EWIRDB veritabanının 2400'ün üzerinde tablo ile modellenmesi hem az kullanışlı hem de performans açısından sorunlu hale getirecektir [17]. Üçüncü normal form ile 2400'ün üzerinde tablo tasarımı yapılması gerekirken dördüncü ve beşinci normal formlarla daha fazla tablo oluşturulacağı için bu modeller de kullanışsız olmaktadır.

3.1.4. Hiyerarşik verilerin ikili dosya olarak saklanması (Storing hierarchical data in binary file)

XML dokümanları tek parça olarak bir tabloda ikili dosya şeklinde saklanıp yönetilebilir. Bu yöntem XML dokümanlarının saklanmasından sonra dosyaların yine ikili dosya olarak sorgulanabilmesini sağlamaktadır. Dosyaların içeriklerini sorgulamak için yine ek uygulamalara ihtiyaç duyulmaktadır. Bilginin yönetilmesinin öncelikli olduğu durumlarda, ikili dosya olarak saklanan veritabanının kullanımı imkansızdır. Bir verinin bir elemanının değerini alabilmek için tüm dosya, veritabanından alınıp, işlenip ve sonuçlar dönecektir. Bu durumda, basit bir tasarımla oluşturulan veritabanının kullanımı zorlaşacaktır.

3.1.5. Hiyerarşik verilerin dinamik olarak saklanması (Storing hierarchical data in dynamic relational database)

İlişkisel veritabanlarının büyük hiyerarşik verileri yönetmek için tasarlanmadığı düşünüldüğünde, ilişkisel veritabanları hiyerarşik verileri dinamik yönetebilecek hale getirilmelidir. Bu yöntemle, hiyerarşiler desteklenmekte ve çok fazla tablo kullanmadan tasarlanabilmektedir. İlişkisel veritabanı içinde hiyerarşilerin tanımlandığı bir tablo ve ilişkilere uygun olarak veri girişi yapılabilecek bir tablo tanımlanmalıdır [17]. Dinamik ilişkisel veritabanında, öncelikle verilerin mümkün olan en az tabloda saklanması ve saklanan verilerin hiyerarşisinin ve doğruluğunun yönetilmesi için bir yapı tablosu tanımlanır. Yapı tablosunda her bir hiyerarşik eleman için bir kayıt bulunur. Bu elemanın veri tipi, adı, etiketi, hiyerarşideki yeri bulunur. Bu tablo XML dokümanları için tanımlanan XSD dokümanının görevini üstlenir. Yapı tablosu kullanılarak bir verinin her elemanı için veri girişi yapılabilecek bir tablo hazırlanır. Bu tabloya veri girişi yazı formatında yapılır. Bu modelde en büyük problem, tüm verilerin yazı formatında saklanmasıdır. XSD dokümanında çok fazla veri tipi tanımlanıp bunların doğruluğu kontrol edilebilirken bu modelde bu ancak yazılım ile kontrol edilebilmektedir. Bu modeldeki diğer bir problem de üçüncü normal formda olduğu gibi veriler dağınıktır.

Üçüncü normal formda veriler tablolara yayılmasına rağmen burada veriler kayıtlar halinde bulunmaktadır. Bir veriye erişebilmek için çok fazla kaydın sorgulanması ve bir araya getirilmesi gerekmektedir. Dinamik ilişkisel veritabanı modeli ile genişletilebilir özelliklere sahip veritabanı modeli tanımlanabilir, fakat dinamik modelin hiyerarşisini kontrol etme konusunda yetersizdir.

Hiyerarşik verilerin ilişkisel veritabanına çevrilmesinde karşılaşılan bütün bu güçlüklere rağmen, hiyerarşik verilerin daha düzgün şekilde yönetilmesi konusunda çalışmalar devam etmektedir. İlişkisel veritabanlarını bu kadar cazip yapan özellikler; veri yönetimi, güvenlik, iş tamamlama, indeksleme kabiliyetleridir. Bu özelliklerin ilişkisel veritabanları yerine, aynı özellikleri destekleyecek alternatif çözümlere de yönelim artmaktadır. Hiyerarşik verilerin yönetimi için nesne yönelimli veritabanları veya XML veritabanlarının kullanımı daha etkin bir çözüm olabilir. Çünkü bu veritabanları doğası gereği verilerin hiyerarşik olarak saklanabilmesine yardımcı olmaktadır.

3.2. Nesne Yönelimli Veritabanı ile Modelleme (Modeling with Object Oriented Database)

Nesne yönelimli veritabanları, nesne yönelimli programlama ile veritabanı yönetim sistemlerinin birleştirilmesi sonucu ortaya çıkan modellerdir. Nesne

yönelimli programlama dillerinde kullanılan nesneler, veritabanlarında saklanarak ilişkisel veritabanına benzer şekilde yönetilmesi sağlanır. Nesne yönelimli veritabanlarının geçmişi, ilişkisel veritabanlarının geçmişi ile paralellik göstermektedir. 1985'li yıllarda bu konuda çalışmalar hızlanmış, fakat ilişkisel veritabanı modelinin gölgesinde kalmıştır. Nesne yönelimli veritabanları, ilişkisel veritabanlarının tüm özelliklerine sahip olabilseler de bazı temel problemleri vardır. Örneğin, ilişkisel veritabanları kadar matematiksel temellere dayanmamakta ve standartlaşma konusunda çalışmalar olsa bile tam bir standartlaşma bulunmamaktadır. Nesne yönelimli veritabanlarında sorgulama için OQL (Object Query Language) kullanılmaktadır. Bu dil SQL diline benzer özelliklere sahiptir. Nesne yönelimli veritabanlarında indeksleme ve sorgulama yöntemleri ilişkisel veritabanlarından farklı olabilmektedir. Sorgulama için ilişkisel veritabanlarında olduğu gibi verilerin birleştirilmesi gerekmemekte, nesnelere olan referanslar kullanılarak verilere ulaşılmaktadır.

Nesne yönelimli veritabanlarında hiyerarşik verilerin modellenmesi için hiyerarşik veride bulunan her bir elemanın nesne olarak oluşturulması gerekmektedir. Oluşturulan bu nesneler veritabanında saklanarak yönetilmektedir. Veritabanında saklanan veriler nesneler olarak saklandığı için verilere ulaşabilmek için bu veritabanına bağımlılık ortaya çıkmaktadır. Ayrıca bu konuda bir standartlaşma olmadığı için bağımlılık kalıcı olabilmektedir. Veritabanında saklanan verilerin ikili yapıda nesnelerde saklanmasından dolayı verilerin transfer edilmesi için de bu veritabanına bağımlılık bulunmaktadır. Hiyerarşik verilerin nesnelere dönüştürülmesi, az elemanı olan veriler için çok kolay olabilirken, çok fazla elemana sahip hiyerarşik verilerin nesne yönelimli olarak modellenmesinin ardından, çok fazla nesne yapısı ortaya çıkacaktır. Örneğin, EWIRDB veritabanının nesne yönelimli olarak modellenmesi sonucu bu veritabanı yapısında tanımlanan parametre kadar nesnenin oluşturulması gerekecektir.

Nesne yönelimli veritabanı hiyerarşik verileri ilişkisel veritabanlarına göre daha kolay modelleyebilmektedir [7, 17]. Bu şekilde modellenen veriler üzerinde, ilişkisel veritabanı ile yapılabilecek bütün sorgulamalar yapılabilmektedir. Fakat herhangi bir hiyerarşide bulunan veriler üzerinde karışık sorgulamalar yapılması bu modelle de mümkün olmamaktadır. Örneğin herhangi bir hiyerarşideki elemanın 5 değerine sahip olan verilerin sorgulanması nesne yönelimli veritabanlarında mümkün olmamaktadır. Nesne yönelimli veritabanı olarak modellenen EWIRDB veritabanı yapısını yönetmek için hazırlanacak olan kullanıcı arayüzleri de çok fazla olacak ve çok fazla zamana ihtiyaç duyulacaktır [11, 12, 17]. Ayrıca nesne yönelimli veritabanı olarak modellenmesi durumunda veritabanında yeni bir alan ekleme özelliği bulunmayacaktır. Veritabanında yeni bir alan eklemek için tekrar tasarım yapılması, nesnelerin güncellenmesi ve eski verilerin yeni yapıya taşınması gerekecektir. Hiyerarşik veriler için nesne yönelimli veritabanlarının ilişkisel veritabanlarına göre çok fazla avantajı olmasına rağmen dinamik alan eklenememesi hiyerarşik verilerin genişletilebilirliğine engel olmaktadır [17]. Genişleyebilir EWIRDB verilerinin ilişkisel ve nesne yönelimli veritabanları ile modellenmesi birçok problemi ortaya çıkarmaktadır.

3.3. XML Veritabanı ile Modelleme (Modeling with XML Database)

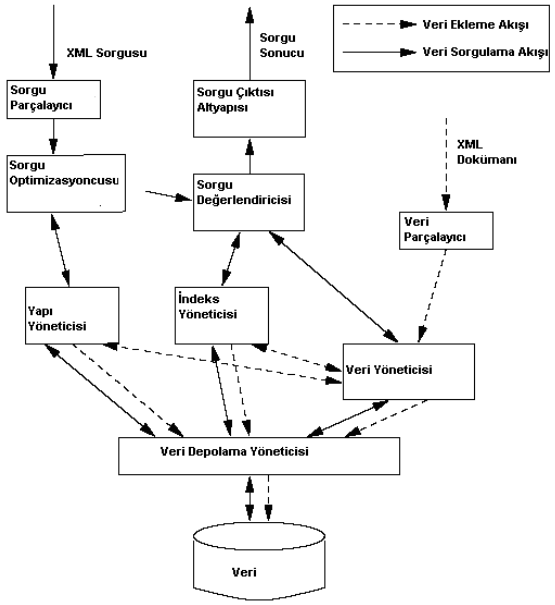
XML veritabanı hiyerarşik verilerin modellenmesinde karşılaşılan problemlerin çözümü için ortaya atılmış, XML formatındaki hiyerarşik verilerin saklanması, sorgulanması ve yönetilmesi için gerekli altyapıyı oluşturmuştur [11, 12, 17]. XML dokümanlarının veritabanı olarak yönetilmesi için iki strateji bulunmaktadır [11, 12, 17, 21]. Bu stratejiler aşağıda açıklanmıştır.

3.3.1. XML formatını destekleyen veritabanları ile modellenmesi (Modeling with XML enabled database)

Bu veritabanları, XML dokümanlarında bulunan verileri ilişkisel veya nesne yönelimli elemanlara dönüştürerek veritabanında saklarlar [22]. Bu verilerin veritabanında saklanma şekli kullanıcı tarafından kontrol edilemez. İlişkisel olarak saklayan veritabanları XML dokümanları üzerinde hem SQL dili ile hem de XQuery dili ile sorgulama imkanı vermektedirler. Nesne yönelimli veya ilişkisel modellerde XML dokümanının saklanması için veritabanlarında ek özellik olarak tanımlanmıştır. Bu veritabanlarının mimarisi XML dokümanlarına uygun olmadığı için kendi iç mimarilerine uygun olacak şekilde XML dokümanlarını yönetmektedir. Bu şekilde hiyerarşik verilerin modellenmesi için veritabanlarının altyapı kısıtları dikkate alınması gerekmektedir. Örneğin, hiyerarşik veriler ilişkisel tablolarda saklanıyorsa, hiyerarşik verilerin ilişkisel formata dönüştürülmesinde karşılaşılan problemler yine ortaya çıkacaktır. XML dokümanının nesne yönelimli olarak saklandığı veritabanlarında, hiyerarşik verilerin bu modele dönüştürülürken karşılaşılacak problemler yine aynı problemlerdir.

3.3.2. Doğal XML veritabanları ile modellenmesi (Modeling with native XML database)

Hiyerarşik verilerin doğrudan desteklenmesini sağlayacak bir model de doğal XML veritabanlarıdır [10, 18, 23]. Bu model mantıksal olarak hiyerarşik verilerin yönetilmesine, XML dokümanlarının elemanlarının bir birim olarak kabul edilmesine ve veritabanı fiziksel saklanma ortamı için bir modele dayandırmadan her türlü fiziksel ortamı destekleyebilecek bir ortam oluşturmaya odaklanmıştır.



Şekil 1. Doğal XML veritabanı mimarisi (Architecture of natural XML database) [23]

İlişkisel veya nesne yönelimli veritabanı modellerin kısıtlarına bağlı olmayan bir veritabanı oluşturarak hiyerarşik verilerin desteklenmesi sağlanmıştır. Doğal XML veritabanları ilişkisel veritabanlarının yetersiz kaldığı durumlarda öne çıkan bir seçenektir. XML dokümanlarının yönetimi sağladığı için verileri küçük parçalara ayırmak gibi sorun oluşmamaktadır. Veritabanlarında saklanan XML dokümanları ağaç şekline getirilerek indekslenebilmekte ve bu sayede normal dosya işlemlerinden daha fazla performanslı olmakta ve dosyalar daha az yer kaplayabilmektedir [24-26]. XML veritabanlarının ilişkisel veritabanları ile veri alışverişi yapabilmesi de mümkündür. XML veritabanlarında yönetilen verilerin kontrol altında ve istenilen formata dönüştürülmesi sağlanabilmektedir. Doğal XML veritabanlarının genel mimari örneği Şekil 1'de verilmiştir. XML dokümanı veritabanına eklenmek istendiğinde, doküman işlenerek (*Veri Parçalayıcı*) ağaç yapısı çıkarılmakta ve oluşturulan ağaç veri yöneticisine gönderilmektedir. *Veri yönetici* bu verileri kullanarak veritabanında saklanacak formata çevirmektedir. Bu format, ağacın elemanlarının bir birim olarak ifade edilmesini sağlayan küçük birimlerden oluşmaktadır. Oluşturulan bu birimler *veri depolama yöneticisi*ne gönderilmektedir. Aynı zamanda *veri yöneticisi*, ağaç halinde bulunan verinin ileride sorgulanabilmesi ve sorguların hızlı gerçekleştirebilmesi için gerekli bilgileri *sorgu değerlendiricisi*ne göndermektedir. Bu bilgi *indeks yöneticisi* tarafından da kullanılmaktadır. *Veri saklama yöneticisi* verileri ağacın elemanlarını ve özelliklerini tanımlanan şekilde saklar. *İndeks yöneticisi,* ağacın elemanları ve elemanların özellikleri üzerinde indeksler oluşturulabilmektedir. Bu indekslemede etiketler ve değerler birlikte değerlendirilmektedir. İndekslemelerde değerler üzerinde indeksleme yaparken veri tipi olarak sayı ve yazı tipleri kullanılmaktadır. Oluşturulan bu indeksleme, sorgulamalarda verilerin ve ağacın elemanlarının hızlı bulunması amacı ile kullanılmaktadır.

Sorgulamalar için işlem öncelikle *sorgu parçalayıcısı* ile başlamaktadır. XQuery ve XPath gibi sorgulama dili kullanılarak gönderilen sorgular burada matematiksel ağaç işlemlerinden geçirilerek anlamlandırılmaktadır. Parçalanan veri *sorgu optimizasyoncusu* tekrar organize edilmekte ve sorguda kullanılacak elemanlar ve eleman özellikleri *yapı yöneticisi* tarafından belirlenmektedir. Sorgu sonucu elde edilecek olan bilgiler, aynı zamanda *sorgu değerlendiricisi* tarafından da değerlendirilmektedir. Bu işlemlerin ardından fiziksel ortamda bulunan hiyerarşik veri elemanları bulunarak sorgu sonucu oluşturulmaktadır. Bu işlemler, ilişkisel veritabanlarının sorgulama mekanizmasına benzese bile XML veritabanları ile ilişkisel veritabanı sorgularına göre daha karmaşık işlemler yapabilmektedir. Örneğin, *yapı yöneticisi* sorgulamalarda kullanılacak bütün elemanları bulmak için kendi içinde ve verileri kullanarak bir sorgu yapmaktadır. XML dokümanlarının sorgulanmasında, dokümanda bulunan elemanların isimlerinin bilinmesine gerek yoktur. "Veritabanında bulunan ve adı X ile başlayan bütün etiketlerin değerlerini dön" gibi bir sorgulamada, sorgu sonucu dönmeden önce veritabanında verilen kurala uygun hangi etiketlerin bulunduğunun belirlenmesi gerekmektedir. Bu işlemden sonra sorgu sonuçları üretilmektedir.

İlişkisel veritabanlarında sorgu yapmak için hangi alanların kullanılacağının kesin olarak belirtilmesi gerekmektedir. Yukarıda örnek olarak verilen sorgunun ilişkisel veritabanlarında SQL dili kullanılarak gerçekleştirilmesi mümkün değildir. Bu tür sorgulamalar ancak yazılımlarda gerçekleştirilebilmektedir. XML veritabanlarında bu çeşit sorgular çok kolay bir şekilde gerçekleştirilebilmektedir. XML veritabanlarında bu tür sorgulamaların mümkün olması ve kullanılması bu veritabanını ön plana çıkarmaktadır. Fakat bu tür sorgular için XML veritabanlarının performanslarının geliştirilmesi gerekmektedir. Çünkü normal SQL sorgulamalarının yaptıkları işlemlerin yanında bu özelliğin gerçekleştirilmesi için ek bir katman ihtiyacı olmakta ve bu katman sorgu sonuçlarının raporlanmasında zaman kaybına sebep olmaktadır. EWIRDB veritabanının doğal XML veritabanları ile yönetilmesi, bu veritabanının bütün özelliklerinin desteklenmesini sağlamaktadır. EWIRDB veritabanının XML dokümanlarına dönüştürülerek bu veritabanına girdi olarak verilmesi ile yönetim sağlanabilecektir.

4. HİYERARŞİK VERİLERİN XML İLE MODELLENMESİ (MODELING OF HIERACHICAL DATA WITH XML)

EWIRDB ağaç hiyerarşilerinin XML dokümanlarına yazılması için XSD şeması hazırlanmıştır. XSD

```
<xs:complexType name="emitter">
  <xs:sequence>
    <xs:element ref="description" minOccurs="0"/>
    <xs:element ref="name" maxOccurs="unbounded" minOccurs="0"/>
    <xs:element ref="packageParameter" minOccurs="0"/>
    <xs:element ref="comment" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
    <xs:attribute name="elnot" type="xs:string"/>
</xs:complexType>

<xs:complexType name="parameter">
  <xs:sequence>
    <xs:element ref="commentReference" minOccurs="0"/>
  </xs:sequence>
    <xs:attribute name="treeNumber" type="xs:string"/>
    <xs:attribute name="name" type="xs:string"/>
    <xs:attribute name="suffixCode" type="xs:string"/>
    <xs:attribute name="sourceType" type="sourceType"/>
</xs:complexType>

<xs:complexType name="packageParameter">
  <xs:complexContent>
    <xs:extension base="parameter">
          <xs:sequence>
          <xs:element ref="parameter" maxOccurs="unbounded" minOccurs="0"
          </xs:sequence>
        <xs:attribute name="packageStatus" type="packageStatus"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="numericParameter">
  <xs:complexContent>
    <xs:extension base="parameter">
      <xs:sequence>
        <xs:element ref="maxValue"  minOccurs="0"/>
        <xs:element ref="minValue" minOccurs="0"/>
        <xs:element ref="resolution"  minOccurs="0"/>
        <xs:element ref="resolutionUnit"  minOccurs="0"/>
        <xs:element ref="unit"  minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="textParameter">
  <xs:complexContent>
    <xs:extension base="parameter">
      <xs:sequence>
        <xs:element ref="text" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Şekil 2. EWIRDB veritabanı verilerini XML dokümanlarına dönüştürmek için kullanılan XSD dokümanı (XSD document that is used for converting EWIRDB database to XML documents)

şemasının bir bölümü Şekil 2'de gösterilmiştir. XSD dosyası ile bir emiter; açıklaması (description), emiter adları (name), parametre paketi (packageParameter) ve yorumları (comment) olan nesne olarak modellenmiştir. Ayrıca emiterin birincil anahtar olarak "elnot" adında bir özelliği bulunmaktadır. Emiterin değişik ülkelerde değişik isimleri olduğu için birden fazla adı olabilmektedir. Ayrıca emiteri tanımlamak için bir açıklama ve emiter hakkında öğrenilen derslerin ve notların girilebileceği yorum parametreleri vardır. Parametre paketi parametreler ve parametre paketleri içeren yapıdır. Bu yapı sayesinde parametre ağacı oluşturulabilmektedir.

Parametre; sayısal (numericParameter), metin (textParameter) ve mantıksal (booleanParameter) parametre tiplerinden herhangi bir tanesi olabilmektedir. Örneğin, sayısal parametre tipi; birimi, minimum değerinin ve maksimum değerlerinin saklanabileceği alanlara sahiptir. Ayrıca bu parametre tipine girilebilecek değerin çözünülürlüğü ve çözünülürlük birimi tanımlanabilmektedir. Oluşturulan XSD dokümanında bazı bilgiler eleman özelliği, bazı bilgiler çocuk eleman olarak tanımlanmıştır. Hangi bilgilerin eleman özelliği olacağı değerin birden fazla değer alıp alamayacağına göre karar verilmiştir. Örneğin, hiyerarşi numaraları her bir eleman için tek olduğundan bunun özellik olmasına karar verilmiştir. XML dokümanının içindeki elemanların tipini belirlemek için XSD şemasında tanımlı olan etiketler kullanılmıştır.

Şekil 3'de örnek bir bölümü görülen XML dosyasında "A232W" birincil anahtarına sahip bir emiterin modeli bulunmaktadır. Emiterin "P/CW TREE" adında parametre paketi bulunmakta ve bu paketin içinde de "GENERAL DESCRIPTION" (Genel Açıklama) ve "SIGNAL POWER" (Sinyal Gücü) adında alt parametre paketleri bulunmaktadır. Bu alt paketlerde de değişik parametreler yer almaktadır. Bir emiterin tüm parametreleri ve yapısı bu şekilde XML yapısına dönüştürülmektedir. Bu XSD şeması kullanılarak dönüşümü yapılan XML dosyaları daha sonra XML veritabanına eklenmektedir.

4.1. Örnek Bir Uygulamanın Hazırlanması (Preparing Sample Application)

Bu projenin geliştirilmesinde Java 6.0 programlama dili, Apache Tomcat 5.5 sunucusu ve eXist veritabanı [10] kullanılmıştır. Veritabanında bulunan verilerin yönetilmesi için gerekli servislerin tanımlanmasında XQuery dili kullanılmıştır. İstemcilerden alınan istekler, XQuery diline dönüştürülerek veritabanında sorgulanmakta ve istek tekrar istemcilere gönderilmektedir. Geliştirilen örnek internet uygulaması, tanımlanan bu servisleri kullanarak verilerin sorgulanmasını ve görüntülenmesini sağlamaktadır. İnternet uygulamasının geliştirilmesinde JSF ve AJAX altyapısı birlikte kullanılmıştır. İnternet tarayıcısında görüntülenecek veriler kullanıcı isteği olduğunda sayfa yenilenmeden sunucuya bağlanarak yeni istekte bulunmaktadır. Şekil 4'de örnek uygulamanın ekran çıktısı verilmiştir. Seçilen bir verinin görüntülenmesi için kullanılan bu ekranda hiyerarşik yapı ağaç şeklinde solda görüntülenmektedir. Kullanıcı hiyerarşide bulunan bir elemanı seçtiğinde eğer bu elemanın alt hiyerarşisi yüklenmedi ise sunucudan bu veriler getirilmektedir. Sağ üst köşede bulunan tabloda seçilen elemandan birden fazla tekrar varsa tekrarlayan elemanların hepsini görüntülemektedir. Bu işlem için yine AJAX alt yapısı kullanılmıştır. Seçilen elemanın tekrarlı olan verileri, ekran yenilenmeden hemen gösterilmektedir.

5. SONUÇLAR (RESULTS)

EWIRDB elektronik harp verilerinin saklandığı en büyük veri kaynağıdır. Bu veri kaynağının verileri

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<emitter xmlns="http://www.gazi.edu.tr/bm/24290291/thesis/v1.0" elnot="A232W">
    <description>Adapted from EWIRDB data and generated pseudo values for bm24290291 thesis</description>
    <packageParameter treeNumber="000000000001.00" sourceType="MIXED" name="P/CW TREE">
        <parameter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="packageParameter"
            treeNumber="000000000010.00" sourceType="MIXED" name="GENERAL INFORMATION">
        <parameter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="packageParameter"
            treeNumber="000000000011.00" sourceType="MIXED" name="SIGNAL POWER">
            <parameter xsi:type="packageParameter" treeNumber="000000000111.00"
                suffixCode="++" sourceType="MIXED" name="CONSTANT POWER">
                <parameter xsi:type="numericParameter" treeNumber="000000000111.10"
                    suffixCode="++" sourceType="OBSERVED" name="PEAK POWER (EFFECTIVE RADIATED)">
                    <commentReference>C002</commentReference>
                    <maxValue>88.6</maxValue>
                    <minValue>88.6</minValue>
                    <resolution>2.0</resolution>
                    <resolutionUnit>DB</resolutionUnit>
                    <unit>DBW</unit>
                </parameter>
                <parameter xsi:type="numericParameter" treeNumber="000000000111.20" suffixCode="++"
                    sourceType="OBSERVED" name="PEAK POWER (AT TRANSMITTER)">
                    <commentReference>C002</commentReference>
                    <maxValue>145</maxValue>
                    <minValue>145</minValue>
                    <resolutionUnit>UNK</resolutionUnit>
                    <unit>KILOWATT</unit>
                </parameter>
                <parameter xsi:type="numericParameter" treeNumber="000000000111.10" suffixCode="++"
                    sourceType="ASSESSED" name="PEAK POWER (EFFECTIVE RADIATED)">
                    <commentReference>K003</commentReference>
                    <maxValue>80.1</maxValue>
                    <minValue>80.1</minValue>
                    <unit>DBW</unit>
                </parameter>
                <parameter xsi:type="numericParameter" treeNumber="000000000111.20" suffixCode="++"
                    sourceType="ASSESSED" name="PEAK POWER (AT TRANSMITTER)">
                    <commentReference>K003</commentReference>
                    <maxValue>145</maxValue>
                    <minValue>145</minValue>
                    <unit>KILOWATT</unit>
                </parameter>
                <parameter xsi:type="numericParameter" treeNumber="000000000111.21" suffixCode="++"
                    sourceType="ASSESSED" name="PEAK POWER AT ANT FEED PORT">
                    <commentReference>K003</commentReference>
                    <maxValue>91.5</maxValue>
                    <minValue>91.5</minValue>
                    <unit>KILOWATT</unit>
                </parameter>
```

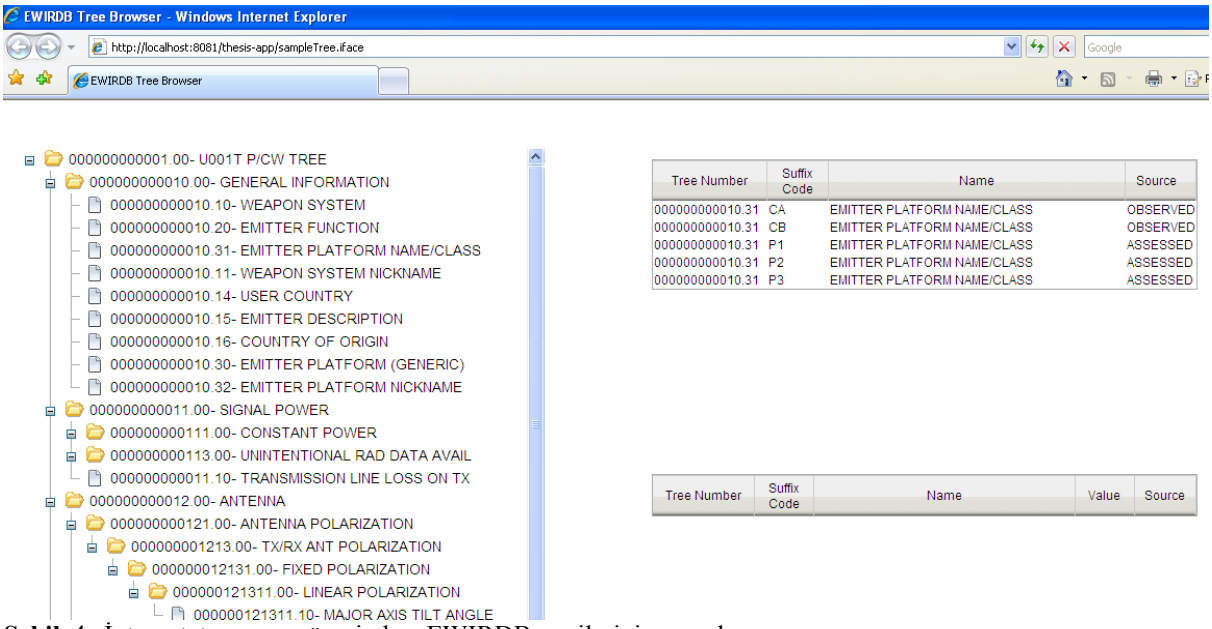
Şekil 3. EWIRDB verilerinin XML dokümanı olarak gösterimi (Representation of EWIRDB data as XML document)

dosyalar şeklinde saklanmakta ve yönetilmektedir. Bu verilerinin etkin yönetimi için veritabanı yönetim sistemleri incelenmiş ve veriler hiyerarşik yapısına uygun olan XML veritabanına taşınmıştır. Bu verilerin modellenmesinde ilişkisel veritabanı modeli ve nesne yönelimli veritabanı modeli etkin bir çözüm olmadığı görülmüştür.

EWIRDB veritabanı üzerinde daha önce yapılan çalışmalarda [1, 6-9] ilişkisel ve nesne yönelimli veritabanı modelleri denenmiş, fakat veritabanının en çok %20'si modellenebilmiş ve elde edilen veritabanlarında kısıtlı sorgular gerçekleştirilebilmiştir. Bu çalışma sonunda ise, veritabanının tamamı modellenmiş ve veritabanı üzerinde istenilen sorgunun çalıştırılabilmesi sağlanmıştır. Bu çalışmada, bunlara ek olarak XQuery sorgulama dili yardımı ile sorgu sonuçlarının istenilen formatta sunulması sağlanmıştır. Farklı alandaki hiyerarşik veriler üzerinde yapılan çalışmalarda [11, 12] da hiyerarşik verilerin XML veritabanları ile modellenmesinin daha uygun olduğu belirtilmiştir.

KAYNAKLAR (REFERENCES)

1. Coyne, K. M., "Design and Analysis of an Object-Oriented Database of Electronic Warfare Data", Yüksek Lisans Tezi, **Naval Postgraduate School**, Monterey, California, USA (1996).
2. Brodie, M. L., Stonebraker, M., "Migrating Legacy Systems: Gateways, Interfaces and the Incremental Approach", **Morgan Kaufman Series in Data Management Systems,** San Francisco, USA, (1995).
3. Tümay, A., Dinçer, K., Yürekten, Ö., Sungur M., Dikici, A, Tambağ, Y., "A Metadata Repository



Şekil 4. İnternet tarayıcısı üzerinden EWIRDB verilerinin sunulması (Presentation of EWIRDB data via an Internet browser)

Model to Integrate Heterogeneous Databases of Hardware Dependent Legacy Systems", **Fourth Biennial International Conference on Advances in Information Systems**, İzmir, Türkiye, LNCS 4243: 149-157 (2006).

4. Keller, A.M., Turner, P., "Migrating to Object Data Management", **OOPSLA Workshop on Legacy Systems and Object Technology**, Austin, Texas (1995).
5. Van Deursen, A., "Understanding Legacy Architectures", **Landelijk Architectuur Congres**, Zeist, Germany (2001).
6. Lee, J. J., McKenna, T. D., "The Object-Oriented Database and Processing of Electronic Warfare Data", Yüksek Lisans Tezi, **Naval Postgraduate School**, Monterey, California, USA (1996).
7. Scrivener, D. N., Edwards, R. D., "Reactivation of Relational Interface in M^2DBMS and Implementation of EWIR Database", Yüksek Lisans Tezi, **Naval Postgraduate School**, Monterey, California, USA (1996).
8. Werre, T. J., Diehl, B. A., "The Activation and Testing the Network CODASYL-DML Interface of the M^2DBMS Using the EWIR Database", Yüksek Lisans Tezi, **Naval Postgraduate School**, Monterey, California, USA (1996).
9. Barnes, G. B., "A Conceptual Approach to Object-Oriented Data Modeling", Yüksek Lisans Tezi, **Naval Postgraduate School**, Monterey, California, USA (1994).
10. Meier, W., "eXist: An Open Source Native XML Database", **Web Databases and Web Services, Lecture Notes in Computer Science**, 2593: 169-183 (2003).
11. Some, R.R., Czikmantory, A., Neff, J., Marshall, M., "XML Hierarchical Database for Missions and Technologies", **Proceedings of the 2004 IEEE Aerospace Conference**, 1: 292-303 (2004).
12. Some, R., Neff, J., "XML Technology Planning Database: Lessons Learned", **Proceedings of the 2005 IEEE Aerospace Conference**, 743-757 (2005).
13. Ida, M., Nozawa, T., Yoshikane, F., Miyazaki, K., Kita, H.: "Syllabus Database and Web Service on Higher Education", **7th International Conference on Advanced Communication Technology (ICACT 2005)**, Seoul, Korea, 1: 415-418 (2005).
14. Philippi, S., Köhler, J., "Using XML Technology for the Ontology-Based Semantic Integration of Life Science Databases", **IEEE Transactions on Information Technology in Biomedicine**, 8(2): 154-160 (2004).
15. Sokic, M., Matic, V., Bazent, A., "Web Content Management System Based on XML Native Database", **Proceedings of the 25th International Conference on Information Technology Interfaces,** 457-462 (2003).
16. Molitoris, J. J., "Use Of COTS XML and Web Technology for Current and Future C2 Systems", **Military Communications Conference**, 1: 221-226 (2003).
17. Yürekten, Ö., Dinçer, K., Akar, B., Sungur, M., Özbudak, E. K., "Migrating a Hierarchical Legacy Database Application onto an XML-Based Service-Oriented Web Platform", **The 21st International Symposium on Computer and Information Systems**, İstanbul, Türkiye, LNCS 4263: 645-654 (2006).
18. Erl, T., "Service-Oriented Architecture A Field Guide to Integrating XML and Web Services", **Prentice Hall**, (2004).
19. Ramakrishnan, R., Gehrke, J., "Database Management Systems", **McGraw-Hill** (2002).

20. Elmasri, R., Navathe, S. B., "Fundamentals of Database Systems", **Addison-Wesley** (2000).
21. Wang, F., Zaniolo, C., "Publishing and Querying the Histories of Archived Relational Databases in XML", **IEEE Proc. of the Fourth International Conference on Web Information Systems Engineering (WISE'03)**, (2003).
22. Jiang, H., Lu, H., Wang, W., Yu, J. X., "XParent: An Efficient RDBMS-Based XML Database System", **IEEE Computer Society Proceedings of the 18th International Conference on Data Engineering (ICDE)**, 335-336 (2002).
23. Jagadish, H. V., Al-Khalifa, S., Chapman, A., Lakshmanan, L. V. S., Nierman, A., Paparizos, S., Patel, J. M., Srivastava, D., Wiwatwattana, N., Wu, Y., Yu, C., "TIMBER: A Native XML Database", **The International Journal on Very Large Databases (VLDB)**, 11(4): 274-291 (2002).
24. Lu, H., Yu, J. X., Wang, G., Zheng, S., Jiang, H., Yu, G., Zhou, A., "What Makes the Differences: Benchmarking XML Database Implementations", **ACM Transactions on Internet Technology (TOIT)**, 5(1): 154-194 (2005).
25. Lawrence, R., "The Space Efficiency of XML", **Information and Software Technology**, 46: 753-759 (2004).
26. Salminen, A., Tompa, F. W., "Requirements for XML document database systems", **Proceedings of the 2001 ACM Symposium on Document Engineering**, Atlanta, Georgia, USA, 85 - 94 (2001).