
Sağlık-NET Projesi

Yeni Başlayanlar İçin HL7 Kılavuzu



T.C. Sağlık Bakanlığı
Bilgi İşlem Dairesi Başkanlığı

Tarih : 22.04.2008
Sürüm : 2.0

Doküman Tarihçesi

Sürüm	Tarih	Değişiklik	Sayfalar
1.0	11 Nisan 2008	Yeni Yayın	1-88
1.1	22 Nisan 2008	Saglik-NET Web Servisleri Ornek İstemci Uygulaması bölümü güncellendi	61-74
2.0	14 Mayıs 2008	“Ek 1 Java istemcisi için gerekli kodlar” Bölüm 4.4 başlığı olarak doküman içine alındı.	74-92
2.0	14 Mayıs 2008	.NET C# İstemci Uygulaması örneği Bölüm 5 olarak dokümana eklendi.	93-107

İÇİNDEKİLER

SAĞLIK-NET PROJESİ.....	1
1 GİRİŞ	4
2 HEALTH LEVEL SEVEN (HL7)	5
2.1 HL7 TARİHÇE.....	5
2.2 HL7 SÜRÜM 2.x.....	6
2.2.1 HL7 sürüm 2.x'teki sorunlar	8
2.3 HL7 SÜRÜM 3	9
2.3.1 HL7 Mesaj Geliştirme Çerçeve.....	10
2.3.2 HL7 Rafine, Lokalizasyon ve Sınırlandırma Yönergeleri (Refinement, Constraint and Localization Guidelines)	14
2.3.3 HL7 Şablonları.....	15
2.3.4 HL7 Klinik Doküman Mimarisi	16
2.3.5 HL7 Referans Bilgi Modeli.....	19
2.3.6 HL7 v3 Oylama Web Sitesi	21
2.3.7 HL7 İletişim Spesifikasyonları	24
2.4 HL7 ARAÇLARINI KULLANARAK ÖRNEK BİR MSVS MUAYENE HL7 V3 MESAJI GELİŞTİRME.....	26
2.4.1 Kurulum İçin Hazırlık	26
2.4.2 Kurulum	27
2.4.3 Kullanım	35
2.4.4 HL7 RMIM Designer Visio Kullanımı	35
2.4.5 HL7 RoseTree Kullanımı	39
2.4.6 HL7 v3 Generator Tool Kullanımı.....	44
3 WEB SERVİSLER VE WEB SERVİSİ SUNUCUSU/İSTEMCİSİ UYGULAMASI.....	48
3.1 KULLANILAN SENARYO	48
3.2 WEB SERVİSİ GELİŞTİRME ORTAMININ HAZIRLANMASI VE KURULMASI	48
3.3 USBS TARAFINDA ÇALIŞACAK WEB SERVİSİN GELİŞTİRİLMESİ	51
3.4 AHBS TARAFINDA ÇALIŞACAK İSTEMCİNİN GELİŞTİRİLMESİ.....	55
3.5 GİDİP GELEN SOAP MESAJLARININ TCPMONITOR YARDIMIYLA GÖRÜNTÜLENMESİ	57
3.5.1 TCPMonitor'un başlatılması.....	57
4 SAĞLIK-NET WEB SERVİSLERİ ÖRNEK JAVA İSTEMCİ UYGULAMASI.....	60
4.1 GİRİŞ	60
4.2 GEREKSİNİMLER	62
4.3 İSTEMCİ KODU ÜRETİMİ	63
4.3.1 Ortamın hazırlanması	63
4.3.2 Kodların Üretilmesi.....	64
4.3.3 Güvenlik (Security) Eklenmesi	65
4.3.4 İstemcinin kullanılması	68
4.3.5 Build.xml Dosyasına Yakından Bakış	72
4.3.6 İstemciye (SaglikNetClient.java) Yakından Bakış	73
4.4 JAVA İSTEMCİSİ İÇİN GEREKLİ KODLAR	74
BUILD.XML.....	74
CLIENT.AXIS2.XML	81
ÖRNEK MUAYENE MESAJI	86
5 SAĞLIK-NET WEB SERVİSLERİ ÖRNEK .NET C# İSTEMCİ UYGULAMASI	93
5.1 GEREKSİNİMLER	93
5.2 İSTEMCİ KODU ÜRETİMİ	93

1 Giriş

T.C. Sağlık Bakanlığı Bilgi İşlem Daire Başkanlığı (BİDB) tarafından yürütüçülüğü gerçekleştirilen Ulusal Sağlık Bilgi Sistemi, Sağlık-NET, sağlık kurumlarında üretilen her türlü veriyi, doğrudan üretildikleri yerden, standartlara uygun şekilde toplamayı, toplanan verilerden tüm paydaşlar için uygun bilgiler üreterek sağlık hizmetlerinde verim ve kaliteyi artırmayı hedefleyen, entegre, güvenli, hızlı ve genişleyebilen bir bilgi sistemidir. Bu veri toplamada kullanılacak standardlar içerik olarak HL7 v3¹ ve iletişim protokolü olarak Web Servisleridir. Diğer bir deyişle medikal bilgi sistemleri Sağlık-NET'e bilgileri HL7 v3 formatında Web Servis Teknolojileri kullananak göndereceklerdir. Bu amaçla Sağlık-NET sistemi medikal bilgi sistemi istemcilerine Web Servis arayüzü (Sağlık-NET Web Servisleri) açmıştır. Bu Web Servis arayuzlerinin istemciler tarafından nasıl kullanılacağını detaylı olarak açıklayan "Uzak Saha Uygulamaları İçin HL7 Mesajları Entegrasyon Dokümanı"² BİDB tarafından yayınlanmıştır.

Bu doküman, HL7 Mesajları Entegrasyon Dokümanı'na bir ek mahiyetinde olup, Sağlık-NET Web Servislerine bağlanmak için yeterli HL7 ve HL7 v3 bilgisi olmayan uzak saha uygulamacılarına giriş bilgisi içermektedir. Ek olarak Web Servisleri hakkında genel bilgi ve Sağlık-NET Web Servislerine Java Programlama dili kullanarak örnek bir istemci uygulama, bu dokümanda sunulmaktadır. Bu doküman sadece uzak saha uygulamacılarına yönelik olmayıp, HL7 v3 hakkında detaylı bilgi edinmek isteyen kişilere bir yönlendirci kılavuzdur.

Dokümanın içeriği şu şekilde düzenlenmiştir. Öncelikle HL7 ve HL7 v3 bilgisi sunulmaktadır. Daha sonra Web servisleri ve örnek bir Web Servisi sunucusu/istemci uygulaması sunulmuştur. En son olarak Sağlık-NET Web Servislerine Java Programlama dili ile bir istemci yazılımı uygulaması kılavuzu sunulmuştur.

¹ <http://www.hl7.org/>

² http://www.sagliknet.saglik.gov.tr/portal_pages/notlogin/saglikcilar/saglikcilar_teknikstandart_hl7.htm

2 Health Level Seven (HL7)

Health Level Seven (HL7), Amerikan Ulusal Standardlar Enstitüsü (American National Standards Institute; ANSI³) tarafından akredite edilmiş, sağlık bilişimi alanında standard geliştiren bir organizasyondur (SDO: Standards Developing Organization). Temel olarak sağlık bilgilerinin elektronik olarak iletişimini sağlayan mesajlaşma standardlarını üretir. HL7 ismi Açık Sistem Bağlantı Modeli'ndeki (Open System Interconnection; OSI) en üst seviye olan Uygulama Seviyesi'nden (Application Level) gelmektedir. Uygulama seviyesi, iletişimini yapılacak verinin tanımı, veri alışverişinin zamanlaması ve belirli hataların uygulamaya tanıtılmasını sağlar.

HL7'in şu anki güncel sürümü sürüm üç'tür (v3) ve bu sürüm henüz oylama aşamasındadır. HL7 organizasyonu v3larındaki ürettiği standardları HL7 v3 Oylama (Ballot) Web Sitesinde⁴ yayımlamaktadır ve bu web sitesini genelde altı aylık periyodlar ile güncellemektedir. Şu an itibarıyle güncel olan oylama Mayıs 2008 sürümudur. Bundan önceki sürüm olan ve "Sürüm 2.x (v2.x)" olarak adlandırılan sürümleri aslında şu an dünyada en çok kullanılan eSağlık standardıdır. Ama Bölüm 2.2.1 de anlatıldığı gibi HL7 organizasyonu bu sürümde sıkıntılıları görerek "Sürüm 2.x" standardlarından çok bağımsız bir sürüm çıkarmıştır. Bu bölümün organizasyonu şu şekildedir. Önce bölüm 2.2 de HL7 v2.x standardlarından kısaca bahsedilip bu sürümdeki sorunlar anlatılacaktır. Daha sonra, bölüm 2.3 de, HL7 v3 sunulacaktır. Bu bölümde genel olarak, HL7 v3, bölüm 2.2.1 de bahsedilen sorunları nasıl çözdüğü, HL7 mesaj geliştirme çerçevesi, HL7 tarafından yayınlanan mesaj standardlarının nasıl kullanılabileceği, HL7 Referans Bilgi Modeli, HL7 İletişim Spesifikasyonları, HL7 Klinik Doküman Mimarisi ve HL7 Oylama Web Sitesinin nasıl kullanılacağı anlatılmıştır.

2.1 HL7 Tarihçe

Bilişim Teknolojilerinin sağlık alanında kullanılmaya başlaması ile birlikte sağlık organizasyonları yazılım şirketlerinden kendi ihtiyaçlarına göre yazılım almaya başladilar. Bu durumda yaşanan en büyük sıkıntı hali hazırda kullanılan yazılımin yeni alınan yazılım ile entegrasyonu idi. İlk aşamada getirilen çözüm sadece iki sistemi entegre etmeye yönelik özel bir çözümü ve ileride başka sistemler alınacağı düşünülürse alınan her sistem için harcanan efor katlanarak artıyordu. Bu sorunu çözme amaçlı olarak, HL7 konsorsiyumu çalışmalarına 1987 yılında başlamış olup 1987-1990 yılları arasında v2.1'i çıkarmıştır. İlk başlarda bu standardın amacı klinik veriden çok Hasta Özlük, Hasta Hesap gibi yönetimsel veri standardları üzerindeydi. Daha sonra 1994 yılında sürüm 2.2, 1997 yılında sürüm 2.3, ve 2003 yılında ise sürüm 2.5 çıkarılmıştır. Her yeni sürümde genel olarak HL7'in kapsama alanı genişletilmiştir.

2000'li yılların sonuna doğru bölüm 2.2.1 de anlatılan sorunları gören HL7 organizasyonu ilerde oluşabilecek tehlikenin farkına varıp, v2.x standardlarını iyileştirmek yerine, sil baştan yeni XML ve Nesne Tabanlı bir standardın ilk adımlarını atmaya başladı ve bu standardı Sürüm 3 olarak isimlendirilmiştir.

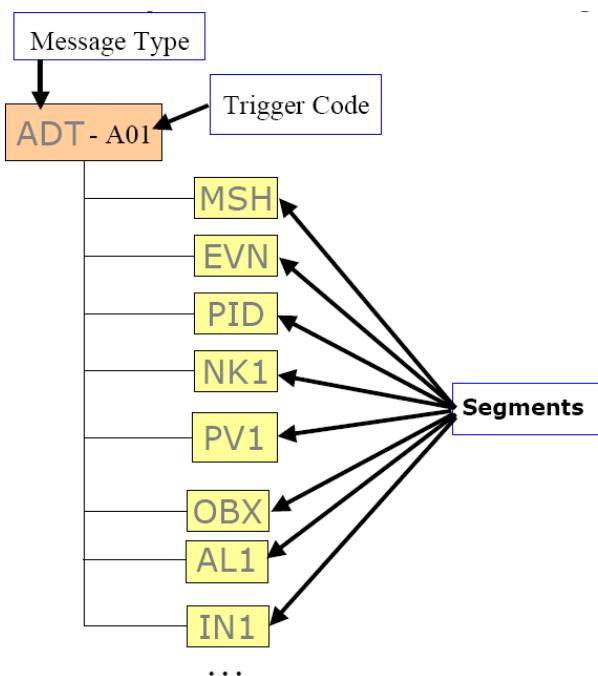
³ <http://www.ansi.org>

⁴ <http://www.hl7.org/v3ballot/html/welcome/environment/index.htm>

2.2 HL7 sürüm 2.x

HL7 v2.x⁵ genel olarak bir sağlık kurumunda olabilecek tüm olayları (trigger events: tetikleyici olaylar) ve bu olaylarda iletişimini yapılacak olan mesajları tanımlar. Mesaj formatı olarak genellikle EDI (Electronic Data Interchange) kullanılır.

HL7 v3'te en temel kavram tetikleyici olaylardır (trigger events). Bu olayları HL7 üç haneli bir kod olarak listeler. Örnek olarak "A02 Hasta transferi". Daha sonra herbir tetikleyici olayda kullanılacak olan mesajları tanımlar. Örneğin A02 olayında ADT (Admit, Discharge, Transfer) mesajı gönderilir ve ACK (Acknowledgement) mesajı alınır. Görüldüğü üzere herbir mesaj içinde üç haneli bir isimlendirme kullanılır. HL7 v2.x'te mesajlar, dilimlerden (segment) oluşmaktadır. Aşağıdaki Şekil 1, örnek bir HL7 V2.x mesajı göstermektedir.



Şekil 1 HL7 v2.x mesaj yapısı

Bu dilimler asıl verinin bulunduğu veri alanlarının (data field) mantıksal olarak gruplanması olarak düşünülebilir. Örneğin, Şekil 1'deki MSH dilimi mesaj başlığı veri alanlarını ve PID dilimi hasta kimlik veri alanlarını tutar. HL7 v2.x mesaj tanımları için kendi kullandığı Soyut Mesaj Sözdizimini (Abstract Message Syntax) kullanır. Şekil 2'de ADT mesajı için bir örnek bulunmaktadır. Burada sırasıyla bu mesajda hangi dilimlerin bulunması gerektiği, hangilerinin opsyonel ("[]") karakterleri ile) ve hangilerinin kendini yineleyebildiği ("{}") karakterleri ile) anlatılmaktadır. Ayrıca bu dilimler hakkında detaylı bilginin, standardın hangi bölümünde olduğu "Chapter" sütünunun altında söylenmektedir.

⁵ <http://www.hl7.org/Library/standards.cfm>

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

HL7 v2.x uygulamacıların kendi dilimlerini yaratmalarına izin vermektedir. Buradaki tek kısıt yeni yaratılan dilim isminin ilk karakterinin "Z" olmasıdır.

ADT^A01	ADT Message	Status	Chapter
MSH	Message Header	2	
	Software Segment	Required Segments	2
EVN	Event Type	3	
PID	Patient Identification	3	
[PD1]	Additional Demographics	3	
	Role	Optional Segments	15
	Next of Kin / Associated Parties	3	
PV1	Patient Visit	3	
[PV2]	Patient Visit – Additional Info	3	
	Role	Optional & Repeating Segments	15
	Disability Information	3	
	Observation/Result	7	
	Allergy Information	3	
	Diagnosis Information	6	
	Diagnosis Related Group	6	
	continued on next slide		
	...		

Şekil 2 Soyut Mesaj Sözdizimi Örneği

Herbir dilim birden fazla veri alanından (data field) oluşmaktadır. Veri alanları asıl verinin durduğu yerlerdir. Örnek olarak Şekil 3'te PID diliminin veri elemanları bulunabilir. Mesela beşinci veri elemanı "Hasta İsmi" (Patient Name) veri alanına bakacak olursak, bu alanın uzunluğu 250 karakter (LEN sütunu), veri tipi XPN: Extended Patient Name (DT sütunu) ve bu alan zorunlu (OPT sütündeki R değeri) ve kendini yineleyebilmektedir (RP sütunundaki Y değeri).

HL7 Attribute Table - PID							
SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	4	SI	O			00104	Set ID - PID
2	20	CX	B			00105	Patient ID
3	250	CX	R	Y		00106	Patient Identifier List
4	20	CX	B	Y		00107	Alternate Patient ID - PID
5	250	XPN	R	Y		00108	Patient Name
6	250	XPN	O	Y		00109	Mother's Maiden Name
7	26	TS	O			00110	Date/Time of Birth
8	1	IS	O		0001	00111	Administrative Sex
9	250	XPN	B	Y		00112	Patient Alias
10	250	CE	O	Y	0005	00113	Race
11	250	XAD	O	Y		00114	Patient Address
12	4	IS	B		0289	00115	County Code
13	250	XTN	O	Y		00116	Phone Number - Home
14	250	XTN	O	Y		00117	Phone Number - Business
15	250	CE	O		0296	00118	Primary Language
16	250	CE	O		0002	00119	Marital Status
17	250	CE	O		0006	00120	Religion
18	250	CX	O			00121	Patient Account Number
19	16	ST	B			00122	SSN Number - Patient
20	25	DLN	O			00123	Driver's License Number - Patient

Şekil 3 PID veri elemanları

```
MSH|^~\&|REG|TGH|LAB|TML|200502151126||ADT^A01|M12345  
P|2.5|<cr>  
EVN|A01|200502151126|<cr>  
PID|1||PATID1234^5^M11||FORBES^JAMES^H||19670329|M||C|  
1200 ELM STREET^ON^M5G1Z6|GL| (416)555-  
1212|(416)555-3434||S||X45 ^2^M10|123456789|987654^ON|<cr>  
NK1|1|FORBES^SADIE^K|WIFE|||CP^Contact person|<cr>  
PV1|1|I|0^2012^01|E|||004777^LIBAUER^SARA^J.||TRMA|||A  
DM|A0|<cr>
```

PID-5 Patient Name (XPN)

Şekil 4 Örnek HL7 v2.x Mesajı

Şekil 4'te örnek bir ADT mesajı verilmektedir. Görüldüğü üzere dilimler mesajın her bir satırını oluşturmaktadır ve bir dilim içindeki veri alanları "| " karakteri ile birbirlerinden ayrılmaktadır. Daha da derine incelemek olursak her bir veri alanı, alt alanlardan oluşmaktadır ve bu alt alanlar birbirlerinden "^" karakteri ile ayrırlar. Örnek olarak Hasta İsmi (Patient Name) veri alanındaki "FORBES^JAMES^H" değerinin alt alanları sırasıyla FORBES, JAMES ve H'dir.

HL7 v2.x, alt seviye'de yani iletişim protokollerinde neler kullanılması gerektiği hakkında kesin kurallar vermez. Ama kullanıcılar TCP/IP, X3.28 ve RS-232 tabanlı uygulama kılavuzları sunar.

2.2.1 HL7 sürüm 2.x'teki sorunlar

HL7 v2.x'teki temel sorunlar dört ana başlık altında incelenebilir:

- Mesaj uyumsuzluğu:** HL7 mesajlarını geliştiren HL7 Teknik Komiteleri, mesajları geliştirirken kuralları belli olan formal bir metodoloji kullanmaktadır. Her komite kendine göre "ad hoc" olarak nitelendirilen bir süreç ile mesaj tanımı üretmektedir. Bu durum komiteler arası "tekrar-kullanımı" oldukça azalmış ve aynı bilgi (information) için farklı veri formatlarının (data format) üretilmesine yol açmıştır. Böylece HL7'ın kendi komiteleri arasında uyumsuz olan mesajlar üretmişlerdir.
- Opsyonellik Sorunu:** HL7 v2.x'te tüm uygulamaların bilgi ihtiyacını karşılamak ve herkesin uyumluluğunu kolaylaştmak amacıyla, mesaj tanımlarında çoğu dilimi ve bu dilimlerdeki veri alanlarını opsyonel olarak tanımlamıştır. Şekil 3'te bu durum gözlemlenebilir. Ayrıca uygulamacıların kendi ihtiyaçlarına göre kendi dilimlerini Z kodu kullanarak yaratmalarına izin vermektedir. Bu durum belli bir süre sonra iki HL7 uyumlu olduğunu söyleyen yazılımın farklı veri modellerine sahip olmalarına ve dolayısıyla birbirleri ile başarılı mesaj alış verişinde bulunamamalarına yol açmıştır. Bu sebeple kişiler HL7'yi standard olarak sorgulamaya başlamıştır.
- Uyumluluk Beyanı:** HL7 v2.x'te uygulamaların HL7'a nasıl ve ne kadar uyduklarını söyleyebilecekleri bir mekanizma bulunmamaktadır. HL7 v2.x'e uyumlu olmak için şu sayıda

mesaj mutlaka sağlanmalıdır diye bir kural yoktur. Bu durum sadece bir-iki mesajı uygulayan yazılımların bile HL7'a uyumlu olduklarını iddia etmelerine yol açmıştır. Bu sebeple literatürde HL7-vari (HL7-ish) gibisinden kavramlar ortaya çıkmıştır. Bu sebeple yine iki HL7 uyumlu olduğunu söyleyen yazılım konuşamamaya başlamıştır.

4. **Bilgisayar'ın işleyebileceği bir Şema'nın bulunması:** HL7 v2.x mesajları yukarıda da belirtildiği üzere Soyut Mesaj Sözdizimleri kullanılarak tanımlanmıştır. Bir uygulayıcı, uyumlu mesaj üretmek için standardın bulunduğu yüzlerce sayfalık dokümanı okuyup tek tek herbir veri alanını kontrol etmek zorunda kalmaktadır. Bu durum uygulamacılara büyük sorunlar çıkarmaktadır.

2.3 HL7 Sürüm 3

Bölüm 2.2.1'de bahsedilen sorunları gören HL7 organizasyonu, yepyeni temeller üzerine kurulu bir v3 geliştirmeye başladı. HL7 v3'teki mesajlar sıfırdan geliştirilecekti ve HL7 v2.x ile uyumluluğu bu aşamada düşünülmeyecekti. HL7 v3, v2.x'in sorunlarını şu şekilde çözmeyi amaçlıyordu:

1. **Mesaj uyumsuzluğu:** Öncelikle 1999 yılında bir Mesaj Geliştirme Çerçevesi (Message Development Framework⁶) dokümanı yayınladı. Bu doküman HL7 v3'ün temeli niteliğinde olup HL7 teknik komitelerinin beraber/hamonize bir şekilde çalışması için formal bir mesaj geliştirme metodolojisi tanımlıyordu. Bu metodoloji nesne tabanlı bir metodoloji olup temeli Referans Bilgi Modeli'ne (Reference Information Model – RIM⁷) dayanıyordu. Genel olarak bir bilgi modeli, kullanıldığı alandaki kavramları, kavramların özelliklerini ve kavramlar arasındaki ilişkileri sınıf ve özellikler ile belirler. Bu bakış açısından RIM sağlık alanında kullanılan tüm bilgiyi modelleyebilecek özelliktedir. Bu metodolojiye göre ilk aşamada Teknik Komiteler beraberce bir RIM geliştirecek ve daha sonra her teknik komite kendi alanında kullanacağı mesajları bu RIM'i kullanarak yine formal bir metodoloji kullanarak geliştirecekti. Daha detaylı bilgi bölüm 2.3.1'de sunulmaktadır. Bu yol ile teknik komitelerin geliştireceği mesajlar uyumlu olacaktır.
2. **Bilgisayar'ın işleyebileceği bir Şema'nın bulunması:** HL7 v3 mesajları format olarak XML kullanmaktadır. Bu XML mesajlarının şeması için de XML Şeması⁸ (XSD: XML Schema Definition) kullanılmaktadır. XML Şeması, bilgisayar tarafından işlenebilen ve bir HL7 v3 mesajının yaratılmasında ve uyumluluk kontrolünde kullanılabilen dokümanlardır.
3. **Opsyonellik Sorunu:** HL7 Mesaj Geliştirme Çerçevesi dokümanını kullanarak mesaj geliştirmeye başladı. Teknik Komiteler bu dokümanda anlatılan formal metodolojiyi kullanarak kendi alanlarındaki (mesela Laboratuvar, Radyoloji, Hasta Yönetim) mesajları üretmeye başladılar ve uygulamacıların kullanımı için mesaj XSD şemaları sağlamaktadırlar. HL7 v3 bu mesaj

⁶ <http://www.hl7.org/library/mdf99/mdf99.pdf>

⁷ <http://www.hl7.org/v3ballot/html/infrastructure/rim/rim.htm>

⁸ <http://www.w3.org/XML/Schema>

şemalarına ek olarak, bu mesaj şemalarının nasıl kullanılacağına ve uygulanacağına dair kullanıcılarla/uygulayıcılarla formal bir metodoloji sunmaktadır. Bu metodolojinin adı “Rafine Etme, Sınırlandırma ve Lokalizasyon (Refinement, Constraint and Localization)⁹” olup temel fikri HL7 tarafından sunulan bir mesaj XSD’sini sınırlandırma üzerinedir. Sınırlandırmaya ek olarak bu kullanımın kullanıcılar tarafından detaylı olarak Profil dokümanlarında anlatılmasını beklemektedir. Bu şekilde opsyonellik sorununu ortadan kaldırmayı planlamaktadır. Unutulmamalıdır ki opsyonellik sorununun ortadan kaldırması demek bir mesajda opsyonel alan olmayacak anlamına gelmez, daha çok bu opsyonel alanların bir entegrasyon projesinde sorun çıkarmaması anlamına gelir.

4. **Uyumluluk Beyanı:** HL7 v3’ün, bölüm 2.3.1’te de belirtildiği gibi, metodolojisi nesne tabanlıdır. Bu metodolojide, kısaca, önce bir storyboard tanımı verilir. Storyboardlar metin halde sağlık alanındaki herhangi bir olayı anlatır. Bu şekilde sağlık alanındaki gereksinimlerin belirlenmesine bir adım atılmış olur. Daha sonra ise bu storyboard’da bulunan uygulama rolleri (application role) ve etkileşimler (interaction) belirlenir. En sonunda ise bu etkileşimde kullanılacak olan mesaj formatları tasarılanır. HL7 v3’te Uyumluluk Beyanı (Conformance Statement) kavramını ortaya koymuştur. Artık HL7 v3 kullanıcıları, HL7 v3’ün hangi uygulama rolünü uyguladıklarını bir Uyumluluk Beyanı dokümanı içerisinde belirteceklerdi. Bir uygulama rolüne uymak, o rol'e ait olan tüm etkileşimleri desteklemek yani o etkileşimlerdeki mesajları alıp verecek özellikte olmak demektir. Daha sonra, HL7 bu kuralı biraz gevşetti ve uygulamacılardan hangi etkileşimleri desteklediklerini belirtmelerini istedi. Bu şekilde, uygulamacıların HL7’i ne kadar ve nasıl destekledikleri belirlenebilecekti.

2.3.1 HL7 Mesaj Geliştirme Çerçeve

Şekil 5’tे görüldüğü üzere, mesaj geliştirmenin başlangıç noktası HL7’in storyboard olarak adlandırıldığı ve sağlık alanındaki veri alışverişi sebebiyet veren olayların metin olarak anlatıldığı metinlerin toplanması ile başlar. Bu storyboard’larda ayrıca nasıl bir veri alışverişi olduğu, hangi olay bu mesajı tetiklediği, alışverişin hangi sistemler arasında olduğu ve mesajın içinde neler olduğu metin olarak bulunmaktadır. Örnek olarak:

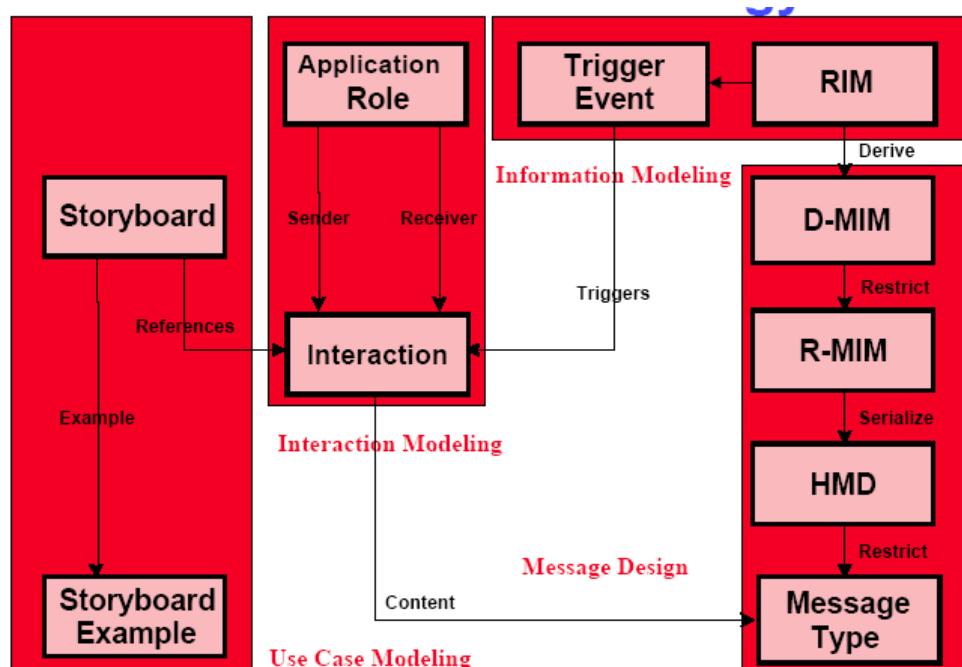
“Neville and Nelda Nuclear, a retired couple living in Home Town, State A, USA, are returning home after a trip to A Foreign Country where they attended a family reunion. On the morning of April 14, Neville and Nelda check out of their hotel in A Foreign Country and take a taxi to the airport where they take flight 556 to Any Town, State B, USA. Except for Neville's persistent cough, the couple enjoyed a long and uneventful flight.

After arriving at Any Town, State B, USA, some passengers of flight 556 stay in Any Town, while the Nuclears, along with others, transfer to flight 1886 to take their final leg home. There are also new passengers from Any Town and Home Town flying on flight 1886. Neville has taken a turn for the worse and is feeling quite ill. Nelda too begins to complain of flu-like symptoms. Late that evening, it is clear that Neville is very ill. He is admitted to the hospital and receives a presumptive diagnosis of SARS.

⁹ <http://www.hl7.org/v3ballot/html/infrastructure/conformance/conformance.htm>

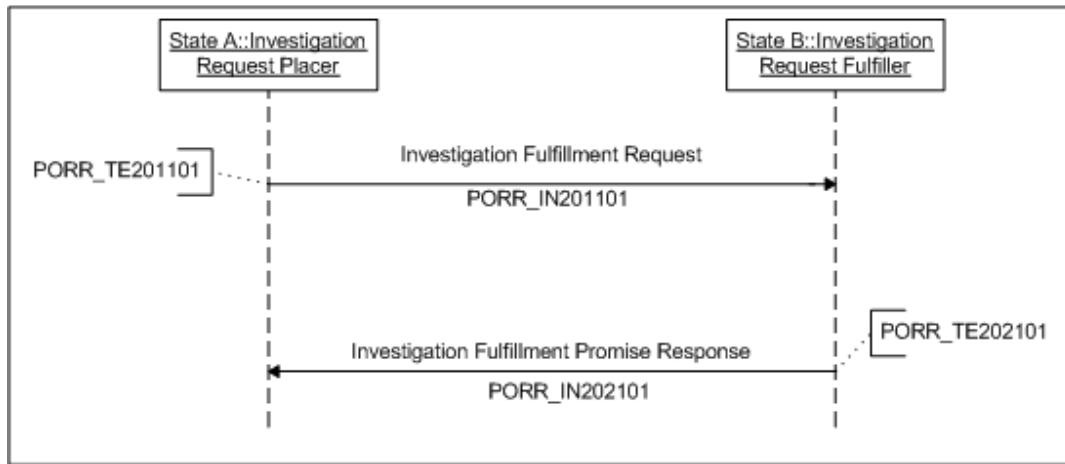
Once the case is reported to the State A Health Department (HD), the State A HD begins its investigation. Upon discovery of Neville's recent travel, State A HD requests the passenger manifests for the flights. State A HD parses the manifest data and determines that some passengers on the flight reside in State B. State A HD places a request for investigations to state B public health authorities using the [Investigation Fulfillment Request](#). State B HD responds with an [Investigation Fulfillment Promise Response](#)."

Tanımdan da görüldüğü üzere, metin içinde tamamen sanal isimler kullanılmıştır. *Neville* ve *Nelda Nuclear* isimli iki kişinin SARS hastalığına yakalandığı şüpheyi üzerine, *State A* hastanesi *State B* hastanesinden "Investigation Fulfillment Request (Tahkikat Gerçekleştirme Talebi)" mesajı ile tahkikat gerçekleştirilmemesini talep ediyor.



Şekil 5 HL7 v3 Mesaj Geliştirme Çerçeve

Daha sonra bu storyboard metnini gösteren UML Etkileşim Diagramı çizilerek HL7 Etkileşimleri ve HL7 Uygulama Rollerleri belirlenir. HL7 Etkileşimlerinin ve Uygulama Rollerinin asıl amacı metinsel bilgi içeren storyboardları formalize etmektir. Şekil 6'de bu storyboard'un UML Etkileşim Diyagramı gösterilmektedir.



Şekil 6 Etkileşim Diyagramı

Bir alandaki tüm storyboard'lar ve tüm etkileşimler belirlendikten sonraki adım bu etkileşimlerde kullanılacak olan mesajların içeriğinin belirlenmesidir. Bu içerik belirlenirken temel olarak storyboard'larda anlatılan bilgiler kullanılır. İçerik belirleme metodolojisi Şekil 5'in sağ tarafında gösterildiği gibi gerekli sınıfların RIM'den kopyalanmasıyla başlar. Daha önce belirtildiği gibi, RIM sağlık bilişimi alanında kullanılacak olan bilgiyi Sınıflar ve bu sınıfların Özellikleri olarak belirler. Bu kopyalanan sınıflar yeni bir bilgi modeli oluşturur ve bu bilgi modeline Alan Mesaj Bilgi Modeli (Domain Message Information Model: DMIM) denir. DMIM o anki mesaj üretecek Teknik Komitenin alanında kullanılacak olan tüm mesajların içeriğinin temelini oluşturur. Daha sonraki adım, bu büyük bilgi kümesinden tek tek mesajların içeriğini oluşturacak bilgi modellerinin üretimesidir. Bu yeni bilgi modelinin adı Rafine Edilmiş Mesaj Bilgi Modelidir (Refined Message Information Model: RMIM) ve üretilerek bir mesajın çatısını oluşturur. RMIM, Teknik Komiteler tarafından oluşturulurken, yani o anki tasarımları yapılacak mesajın içeriği belirlenirken, DMIM'den gerekli sınıflar çekilir. Bu sınıflara bu aşamada HL7 tarafından belirlenen sınırlandırımlar uygulanır. HL7 v3, altı çeşit sınırlandırma yolu sunmaktadır¹⁰:

1. Görünüş Kısıtları (Appearance Constraints): RIM'deki sınıflar sağlık'taki her türlü bilgiyi modelleme amaçlı olduğu ve geniş bir bilgi alanını kapsayabilmek için çok fazla sayıda özellik (attribute) içermektedir. Bazı özellikler o anki tasarımları yapan teknik komitenin işine yaramayabilir. Örneğin, RIM'deki Gözlem (Observation) sınıfı, Hasta Yönetim alanında kullanılırken, Gözlem sınıfının "kesilebilir mi göstergesi (interruptableInd)" özelliği kullanılmayabilir. Bu sebeple bu sınıf DMIM'e kopyalanırken interruptableInd özelliği olmadan eklenir. Bu bir görünüş kısıtidır.
2. Eleman Kendini-Tekrarlama (Cardinality) Kısıtları: Bu kısıtlar bir özelliğin kendini yineleme sayısının azaltılmasını sağlayan kısıtlardır.
3. Tip (Type) Kısıtları: HL7 v3, çok kapsamlı bir veri tipi kütüphanesi sağlamaktadır¹¹. Ek olarak sağlanan bu veri tipleri arasında bir hiyerarşi bulunmaktadır. Örneğin, Organizasyon İsmi

¹⁰ <http://www.hl7.org/v3ballot/html/infrastructure/conformance/conformance.htm>

¹¹ http://www.hl7.org/v3ballot/html/infrastructure/datatypes_r2/datatypes_r2.htm

(Organization Name) ve Kişi İsmi (Person Name) veri tipleri aynı seviyede olup, Varlık İsmi (Entity Name) veri tipinin hemen altında yer almaktadır. Bu sebeple, DMIM'de veri tipi Varlık İsmi olan bir özellik, RMIM'e kopyalanırken veri tipi Kişi İsmi'ne veya Organizasyon İsmi'ne sınırlanır.

4. Sözlük (Vocabulary) Kısıtları: HL7 v3 mesajlarda kullanılmak üzere Sözlük Alanları (Vocabulary Domain) belirler. Bu alanlar kodlu değerler değil, kavramlar (concept) içerirler. Daha sonra uygulamacılar, bu kavramlar için kullanacakları kodlu değer kümelerini (Value Set) bu sözlük alanlarına atarlar. Örneğin, HL7, YönetimselCinsiyet (AdministrativeGender) isimli ve içerisinde Erkek, Kadın, Belirsiz gibi kavramlar içeren Sözlük Alanı tanımlar. Varsayılmak ki, o anki uygulamadaki mesajlarda Erkek için 'E', Kadın için 'K' ve Belirsiz için 'B' kullanılacak. Bu durumda {Erkek, Kadın, Belirsiz} kümesi sözlük alanını oluştururken, {E, K, B} atanmış değer kümesini oluşturmaktadır. RIM'deki her kodlu özellik için bir Sözlük Alanı belirlenmiştir.

Bunlara ek olarak kodlu değer içeren her bir özelliğin "kodlama gücü" (coding strength) vardır. Bu güç için iki olasılık belirlenmiştir: CNE (kodlamada istisna olamaz: coded no exception) ve CWE (kodlamada istisna olabilir: coded with exception). Güçü CWE olan bir özelliğin sözlük alanındaki kavram kümesi ihtiyaca göre genişletilebilir. Yani mesaj bir uygulamada kullanılırken eğer o mesajın bir kodlu özelliğinin sözlük alanı gereklili bir kavramı içermiyorsa bu kavram eklenebilir. CNE olma durumunda ise sözlük alanının kavram kümesi kesinlikle değiştirilemez.

Sözlük kısıtları söz konusu olunca, DMIM'den bir sınıf RMIM'e kopyalanırken, şu sınırlandırmalar olabilir:

- Bir Sözlük Alanı'nın kavram kümesi küçültülebilir.
- Bir değer kümesi bir sözlük alanına atanabilir. Yani bu atamada bir çeşit sınırlama yoludur.
- Bir değer kümesinin kod seti küçültülebilir.
- Güçü CWE olan bir özelliğin gücü CNE'ye çevrilebilir.

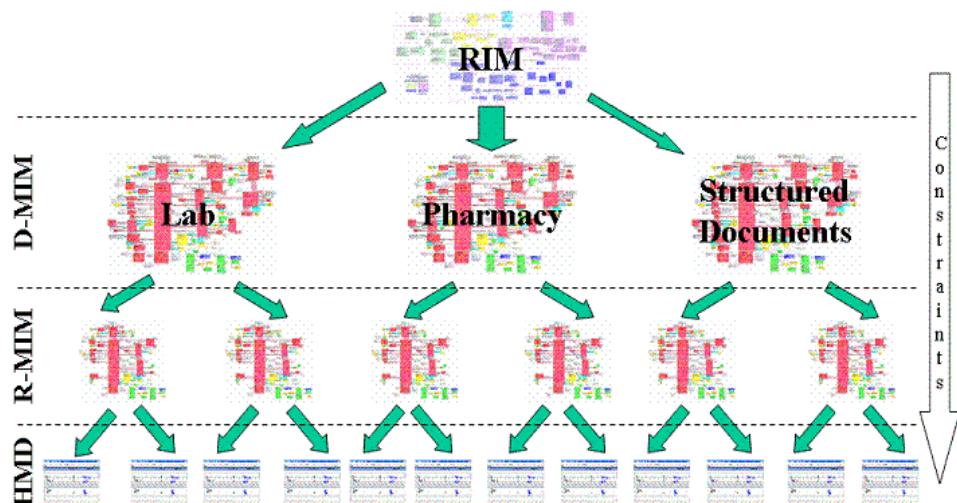
5. Diğer Değer Kısıtları: Bu kısıtlamalar bazı özelliklere sabit değerler atanmasını veya değer sınırı koyulmasını (örneğin, yaş özelliğinin değeri 150'den büyük olamaz gibi) içerir.
6. Detaylı açıklamalar (Annotations): Burada o anki özellik yada sınıf için daha da fazla metinsel açıklama getirilir.

RMIM aşaması kendini yineleyen bir aşamadır her bir yinelemede yukarıda bahsedilen sınırlandırmalar bazları kullanılır ve o anki tasarımları yapan Teknik Komitenin geliştirilen RMIM'in belli bir olgunluğa eriştiğine karar verene kadar devam eder.

RMIM'ler birer bilgi modelidir ve bir mesaj formatına yani XSD'ye nasıl çevrilir bilgisini içermez. Bu sebeple XSD'ye çevrimi kolaylaştıracak olan bir tablo formatına ihtiyaç vardır. Bu tablo formatına Hiyerarşik Mesaj Tanımı (Hierarchical Message Description) denir. RMIM'lerin tamamlanmasından sonra, RMIM'lerden HMD'ler üretilir. Bu geçiş aşamasında da yukarıda bahsedilen altı sınırlama kullanılabilir. Göründüğü üzere bilgi modelleri arasında bir seviye vardır. En üstte RIM ve daha sonra sırasıyla DMIM, RMIM ve HMD bulunmaktadır ve her bir seviyedeki bilgi modeli kendinden bir üstteki bilgi modelinden yaratılmaktadır. Bu geçişlerin herbirinde yukarıda bahsedilen altı sınırlama çeşidi uygulanır. En son adım olarak yazılım araçları ile üretilen HMD'ler, XSD'ye çevrilir.

Etkileşimlerde kullanılacak içeriğin tanımlanması aşamalarını anlatan diğer bir figür de Şekil 7'de sunulmaktadır.

Static Model Hierachy



Şekil 7 Mesaj Tanımlama Çerçeve

2.3.2 HL7 Rafine, Lokalizasyon ve Sınırlandırma Yönetgeleri (Refinement, Constraint and Localization Guidelines)

HL7 v3, uygulamacılara mesaj XSD şemaları sunmaktadır. Uygulamacılar bunları olduğu gibi kullanabilirler. Ama HL7 dahil her standardın iki sorunu vardır:

1. Standard tarafından sunulan orjinal mesaj şeması o anki uygulamaya göre çok generic olabilir, bu yüzden sınırlandırma gerektirebilir
2. Sunulan mesaj şemaları o anki uygulamanın tüm veri gereksinimlerini karşılayamayabilir, bu yüzden genişletilmesi (mesela yeni eleman eklenmesi) gerekebilir

Bu sorunları gören HL7 organizasyonu, sürüm 3 standardını çıkarırken teknik komitelerin kullanacağı metodolojiye ek olarak, uygulamacıların da kullanacağı ve yukarıdaki sorunları çözmeye yönelik formal bir metodoloji sunmaktadır. Bu metodoloji "HL7 Rafine, Lokalizasyon ve Sınırlandırma Yönetgeleri¹²" nde bulunmaktadır.

Sunulan metodolojinin temel mantığı, uygulamacıların HL7'in sunduğu orjinal XSD mesaj şemalarını elliinden geldiğince kullanmalarıdır. Eğer bu mümkün değilse, kullanıcı bir üst seviye olan HMD seviyesine çıkış yaparak gerekli sınırlandırmaları yapıp, kendi XSD şemalarını üretebilir. Eğer bu da mümkün

¹² <http://www.hl7.org/v3ballot/html/infrastructure/conformance/conformance.htm>

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

değilse, uygulamacılar RMIM seviyesine çıkış sınırlandırmalarını yapıp buradan kendi mesaj XSD şemalarını üretebilirler. Bu geçişlerde bölüm 2.3.1'de bahsedilen sınırlandırma çeşitleri kullanılabilir. Bu sınırlandırma yolları ile kullanıcı tarafından yeni üretilen XSD şemasına uygun bir mesaj aynı zamanda HL7'in sunduğu orjinal mesaj şemalarına da uyacaktır. Bu şekilde sınırlandırma sorunu aşılmış olmaktadır.

HL7 v3'ün sunduğu mesajlar oldukça kapsamlı mesajlardır. O anki uygulamada gerekli olan bir elemanın eksikliği genelde çok olası gözükmektedir. Eğer kullanıcının bir eleman eklemesi gerekiyorsa bunu RMIM seviyesinde yapabilir ama bu çeşit genişletmeler HL7 tarafından informal olarak kabul edilmektedir.

HL7, kullanıcılarından yaptıkları sınırlandırmaları veya genişletmeleri Profil denen dokümanlarda anlatmalarını beklemektedir. Bu yönde beş çeşit profil tipi bulunmaktadır:

1. Sınırlama Profili: Bu dokümanda kullanılan mesajlarda hangi sınırlamalar kullanıldığı anlatılmaktadır.
2. Uygulanabilir Profil: Bu doküman genelde Sınırlama Profillerinden daha sınırlıdır ve hiç bir opsyonel eleman tanımına izin vermez. Amacı tak-çalıştır (plug-and-play) birlikte çalışabilirliği sağlamaya yönelikir.
3. Uyumluluk Profili: Bu profil ya bir sınırlandırma ya da bir uygulanabilir profilden oluşmaktadır ve uygulamacıların kendi sistemlerinde HL7 v3'e nasıl uyduklarını anlatmalarını sağlar.
4. Localizasyon Profili: Bu profil ile bir bölgede uygulması gereken HL7 mesajları anlatılır. Ayrıca genişletilmiş elemanlar belirtilir.
5. Açıklama Profili: Bu profil tipi ile uygulamacılar kendi yazılımlarını yukarıdaki profillere ek olarak daha detaylı olarak metin halinde açıklarlar.

Bu profillerin içlerinde neler olması gereği henüz HL7 tarafından formal olarak belirlenmemiştir. Ama aşağıdaki bilgilerin sunulması önerilmektedir:

- HL7 mesajlarının kullanımının nasıl olduğunu metinsel olarak anlatan Storyboard'lar ve bu Storyboard'lara uygun olarak çizilen UML Kullanım-Durumu (Use-Case) ve Etkileşim (Interaction) şemaları
- Desteklenen HL7 v3 etkileşimleri (interactions)
- Herbir etkileşim için değişimi yapılan mesajlar ve bu mesajlarda kullanılan sınırlamalar
- Var ise, ayrıca yapılan informal (standarda uygun olmayan) mesaj genişletimleri (extensions)
- Kullanılan transport protokollerı

2.3.3 HL7 Şablonları

HL7'in önerdiği sınırlandırma çeşitlerine ek sınırlamalar da mevcuttur. Mesela "Hasta'ya yüksek tansiyon tanısı konmuşsa kan basıncı 12/8'den yüksek olmalıdır". Bu gibi sınırlamalar mevcut XSD

kuralları ile belirtilememektedir. Bu sebeple HL7 organizasyonu, şablonlar projesini¹³ başlatmıştır. Şablonlar ile bir mesaja ek sınırlar getirilebilir. Şablonlar projesi şu an tasarım aşamasındadır ve şablonun içeriğini belirtmek için Schematron¹⁴ dilinin kullanılması da olasılıklar arasındadır.

2.3.4 HL7 Klinik Doküman Mimarisi

HL7 Klinik Doküman Mimarisi¹⁵ (Clinical Document Architecture:CDA) iletimi yani alışveriş yapacak olan klinik dokümanların formatını ve anlamını tanımlayan bir doküman standartıdır. Klinik dokümanlar şu özelliklere sahiptirler:

- Kalıcılık (Persistence): Belli bir yerde (Veri tabanı yahut dosya sistemi) değiştirilmeden belli bir süre boyunca olduğu gibi saklanması
- Yönetilebilirlik (Sterwardship): Bir organizasyon tarafından yönetimi sağlanması
- Onaylama (Authentication): Bir şahıs tarafından kanunen onaylanmalıdır
- Bütünlük (Wholeness): Yapılan kanuni onaylama dokümanın bir parçasına değil bütününe aittir
- İnsan tarafından okunabilirlik (Human Readibility): Doküman bir insan tarafından okunabilir olmalıdır

Örnek olarak elektronik hasta kayıtları bir klinik doküman olarak düşünülebilir. CDA dokümanları XML kullanılarak tasarlanmış olup bilgi içeriğini diğer HL7 v3 mesajları gibi RIM'den almıştır. Unutulmamalıdır ki CDA'nın kapsamı iletimi (exchange) yapılacak olan klinik dokümandır. HL7, doküman'ın bir doküman denetim sisteminde nasıl tutulacağına karışmaz. CDA dokümanları bir HL7 mesajı içinde gönderilebileceği gibi ayrıca (Integrating Healthcare Enterprise: Cross Enterprise Document Sharing¹⁶ (IHE-XDS) Profilinde olduğu gibi) başka iletişim protokollerini de kullanılabılır.

Bir CDA dokümanı “<ClinicalDocument>” elemanları ile sarılmış, bir başlık ve bir gövde içeren dokümanlardır. Başlık kısmı “<ClinicalDocument>” ile “<structuredBody>” elemanları arasında kalan kısımdır. Gövde kısmı temel olarak “<section>” elemanlarından oluşur ve bu elemanlar, insan okuması için düz metin içeren bir “<text>” elemanını ve ek olarak bu düz metnin bilgisayar tarafından işlenmesini sağlayan girdileri de (“<entry>”) içerebilir. Şekil 8'de örnek bir CDA dokümanı sunulmaktadır.

¹³ <http://www.hl7.org/v3ballot/html/infrastructure/templates/templates.htm>

¹⁴ <http://www.schematron.com/>

¹⁵ <http://www.hl7.org/v3ballot/html/infrastructure/cda/cda.htm>

¹⁶ <http://www.ihe.net/Profiles/index.cfm#IT>

```
<ClinicalDocument>
  ... CDA Header ...
  <structuredBody>
    <section>
      <text>...</text>
      <observation>...</observation>
      <substanceAdministration>
        <supply>...</supply>
      </substanceAdministration>
      <observation>
        <externalObservation>...
        </externalObservation>
      </observation>
    </section>
    <section>
      <section>...</section>
    </section>
  </structuredBody>
</ClinicalDocument>
```

Şekil 8 Örnek CDA dokümanı

CDA'yı uygulayan kişiler sağlanan orjinal XSD'yi kullanabilecekleri gibi daha da fazla sınırlandırma yapabilirler. Bu sınırlandırmayı "<section>" seviyesinde şablonlar ya da ek olarak "<entry>" seviyesinde şablonlar uygulayarak yapabilirler. CDA ismindeki Mimari (Architecture) kelimesi de bu sınırlandırma mantığından gelmektedir. Eğer hiç bir sınırlandırma uygulanmayan XSD kullanılıyorsa bu uygulama Seviye 1'dedir. Eğer "<section>" seviyesinde şablonlar uyguluyorsa Seviye 2'dedir. Ek olarak "<entry>" seviyesinde şablonlar uyguluyorsa Seviye 3'dedir. Görüldüğü üzere bir uygulamanın CDA uyumluyum demesi için en temel gereksinim üretilen CDA XML dokümanlarının HL7 tarafından yayınlanan orjinal CDA XSD Şemasına uygun olması gerekmektedir.

2.3.4.1 CDA Başlığı (Header)

Başlık bölümündeki bilgiler bir CDA dokümanın işlenmesinde işe yarayacak bilgilerdir. Kısaca şu bilgiler tutulur:

- Doküman id'si
- Dokümanın ne hakkında olduğunu gösteren kodu
- Dokümanın ismi
- Dokümanın yaratılış zamanı
- Dokümanın gizlilik kodu
- Doküman dili
- Kim tarafından onaylandığını ve kim tarafından yazıldığı
- Dokümandan sorumlu olan organizasyonu
- Doküman alıcısı
- Hangi hasta için olduğu
- Diğer hangi dokümanlarla ilgili olduğu

2.3.4.2 CDA Gövdesi (Body)

Gövde kısmında asıl iletilecek olan veri “`<section>`” elemanları içerisinde tutulur. Herbir “`<section>`” elementinin id’si, kodu, ismi, metin kısmı, dili, gizlilik kodu, yazarı ve öznesi bulunmaktadır. Şekil 9’da örnek bir CDA “`<section>`” elemanı sunulmaktadır.

```
<section ID="SECT001">
  <code code="10164-2" codeSystem="2.16.840.1.113883.6.1"
    codeSystemName="LOINC"/>
  <title>History of Present Illness</title>
  <text>Mr. Smith is a 57 year old male presenting with
    chest pain. He sustained a myocardial infarction 3 years
    ago, ...
  </text>
</section>

...
<section ID="SECT003">
  <code code="10153-2" codeSystem="2.16.840.1.113883.6.1"
    codeSystemName="LOINC"/>
  <title>Past Medical History</title>
  <text>History of coronary artery disease, as noted
    <linkHtml href="#SECT001">above</linkHtml>.</text>
</section>
```

Şekil 9 Örnek CDA Section Elemanı

İnsan tarafından okunabilir “`<text>`” elemanına ek olarak, bu metinde anlatılan bilgiyi bilgisayar tarafından işlenmesini kolaylaştıracak olan bir veya daha fazla “`<entry>`” elemanı tanımlanabilir. CDA kişilere yardımcı olması açısından kendi “`<entry>`” tipleri yaratmıştır. Bunlar arasında Gözlem (Observation), Prosedür (Procedure) gibi bilgiler bulunmaktadır.

```
<section>
  <code code="10153-2"
    codeSystem="2.16.840.1.113883.6.1"
    codeSystemName="LOINC"/>
  <title>Past Medical History</title>
  <text>
    There is a history of <content ID="al">Asthma</content>
  </text>
  <entry>
    <observation classCode="OBS" moodCode="EUN">
      <code code="195967001"
        codeSystem="2.16.840.1.113883.6.96"
        codeSystemName="SNOMED CT"
        displayName="Asthma">
        <originalText>
          <reference value="#al"/>
        </originalText>
      </code>
      <statusCode code="completed"/>
    </observation>
  </entry>
</section>
```

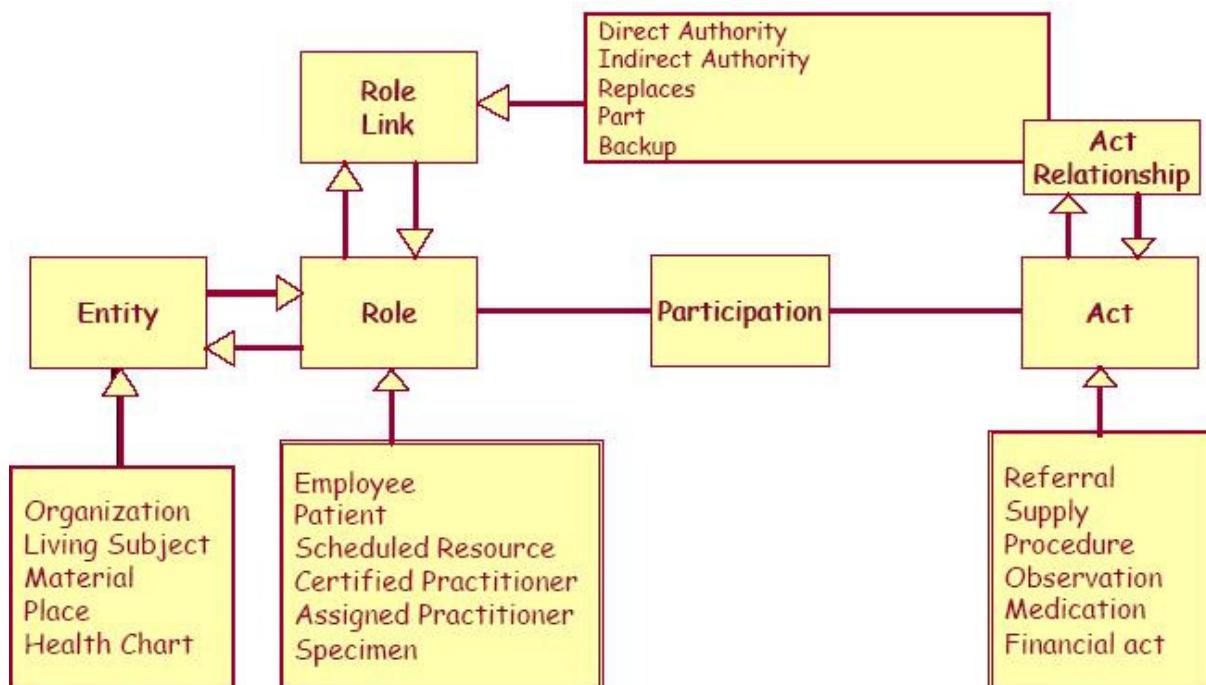
Şekil 10 Entry içeren CDA Dokümanı Örneği

Şekil 10'da “<entry>” içeren bir örnek sunulmaktadır. Görüldüğü gibi düz metin “<text>” bölgesindeki Astım kelimesi bir “<observation>” entry'si ile SNOMED CT kullanılarak kodlanmıştır.

2.3.5 HL7 Referans Bilgi Modeli

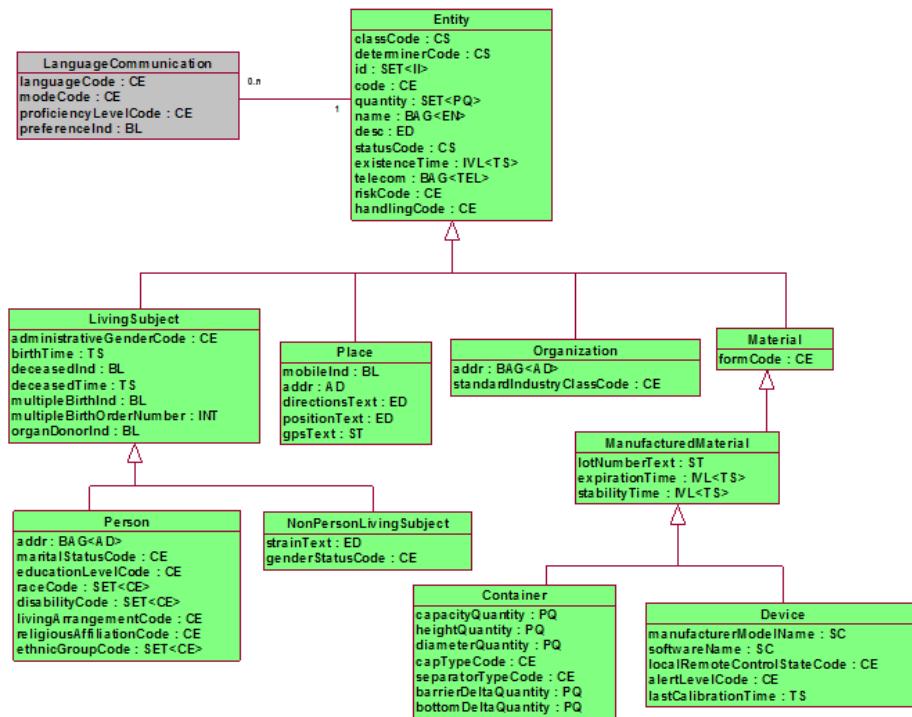
HL7 Referans Bilgi Modeli (Reference Information Model: RIM) sağlık bilişimindeki olsası tüm mesajların alt yapısını oluşturan bir bilgi modelidir. Temel amacı Teknik Komitelerin mesaj geliştirmelerinde aynı yerden başlamalarını sağlamak böylece tekrar-kullanımı artırıp farklı alanlardaki mesajlar arasındaki olsası uyumsuzluğu engellemektir. RIM çok kapsamlı bir bilgi modeli olmasına karşın temel olarak altı sınıf ve bu sınıfların özelleştirilmiş altsınıflarından oluşmaktadır. Şekil 11'de bu altı sınıf sunulmaktadır:

1. Act: En temel sınıf Act sınıfıdır. Bu sınıf sağlık alanındaki olabilecek Muayene, İzlem, Tetkik gibi tüm aktiviteleri temsil eder.
2. Entity: Canlı veya cansız tüm varlıklar bu sınıf ile temsil edilir. Şahıs, Materyal bu sınıfın olsası altsınıflarıdır.
3. Role: Bu sınıf belirlenmiş bir Entity sınıfının rolünü/yetisini gösterir. Mesela bir Şahıs Entity'sinin rolü Doktor iken başka bir Şahıs Entity'sinin rolü Hasta olabilir.
4. Participation: Bu sınıf belli bir Role sınıfının, bir Act'e nasıl katıldığını belirler. Diğer bir deyişle, bir rol, bir activiteye nasıl katılıyor. Mesela Hasta Rolü, Muayene Act'ine muayene edilen olarak katılırken, Doktor Rolü muayeneyi gerçekleştiren olarak katılabilmektedir.
5. ActRelationship: Bu sınıf Act'ler arasındaki ilişkiyi tanımlar. Mesela bir Muayene Act'i sonucu Tetkik Act'i istenebilir. Bu durumda bu iki Act arasında tipi “Sebep-Reason” olan bir ActRelationship konulmalıdır.
6. RoleLink: Bu sınıf Role sınıfları arasındaki ilişkiyi belirler.



Şekil 11 RIM Temel Sınıfları

RIM kurulurken temelde olacak şekilde önce bu altı sınıf oluşturuldu. Daha sonra diğer olası sınıflar bu altı sınıfın özelleştirilmiş altsınıfları olarak RIM'e yerleştirilmeye başlandı. Örnek olarak Şekil 12'da Entity sınıfının altsınıfları gösterilmektedir. Bölüm 2.3.1'de bahsedildiği gibi, Teknik Komiteler kendi alanlarında kullanılmak üzere geliştirecekleri mesajlar için önce bir DMIM oluşturmakdadırlar. Bu DMIM'leri, RIM'den gerekli sınıfları kopyalayarak oluşturmaktadır.



Şekil 12 Entity sınıfının alt sınıfları

2.3.6 HL7 v3 Oylama Web Sitesi

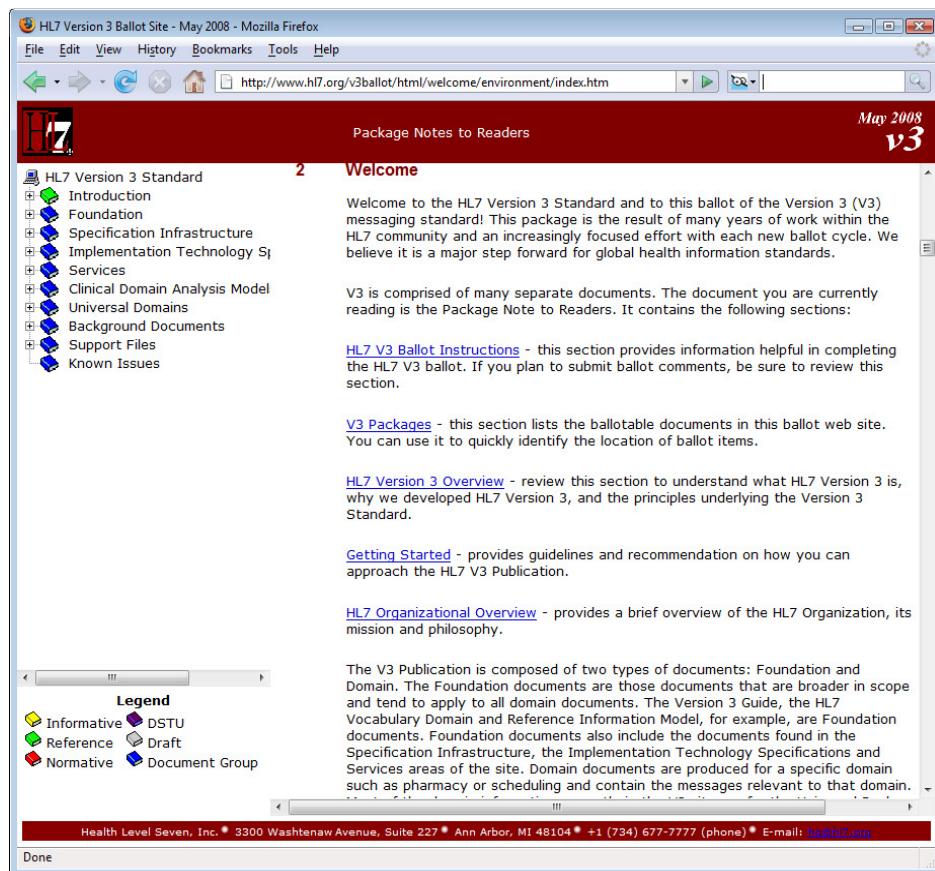
Önceden de belirtildiği gibi HL7 v3 standarı henüz oylama aşamasındadır. HL7 organizasyonu, v3 standartını genelde altı aylık periyodlarla güncelliyor ve v3 oylama Web Sitesinde¹⁷ yayımlamaktadır. Şekil 13'de bu sitenin giriş sayfası gösterilmektedir. Sol tarafataki ağaç yapısıyla okuyucular sitede dolaşabilirler. Uyulma zorunluluğuna göre düşünüldüğünde beş çeşit doküman vardır ve bu çeşitleri sayfanın sol tarafındaki ağaç yapısının altındaki renkler ile de gösterilmektedir:

1. Informative Dokümanlar: Bu dokümanlar sarı renk ile gösterilir ve bilgi verme amacındadır. Uygulamacılara kural koyma niteliğinde değildirler.
2. Normative Dokümanlar: Bu dokümanlar kırmızı renk ile gösterilir ve standartın kendini oluşturmaktadır. Diğer bir deyişle, uygulamacılar bu dokümanlarda belirtilen kurallara uymak zorundadır.
3. Draft Dokümanlar: Henüz tamamlanmamış üzerinde çalışmalar halen devam etmekte olan ve gri renk ile gösterilen dokümanlardır.
4. Reference Dokümanlar: Bu dokümanlar kişilerin standartı daha iyi anlamalarına yardımcı olan dokümanlardır. Mesela web sistesindeki Glossary (Sözlük) bölümü bir referans dokümandır.

¹⁷ <http://www.hl7.org/v3ballot/html/welcome/environment/index.htm>

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

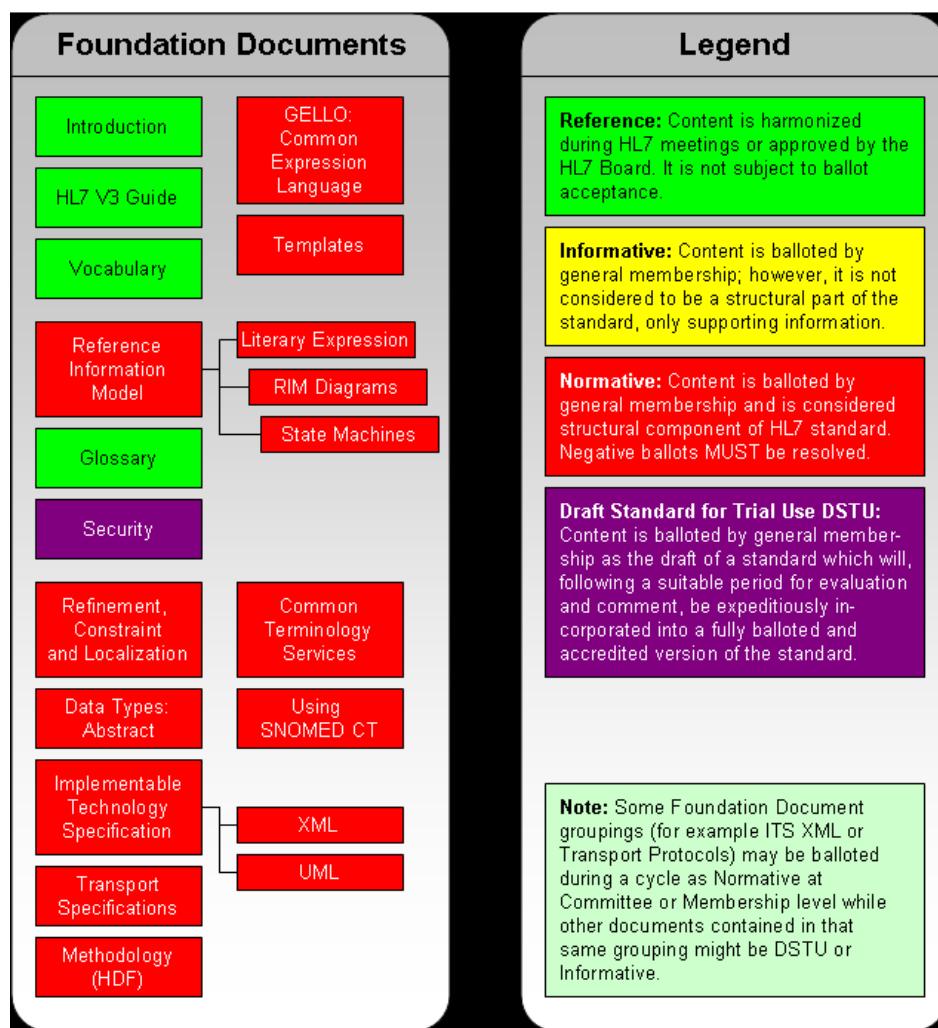
5. DSTU (Draft Standard for Trial Use) Dokümanlar: İleride normative olacak ama henüz uygulama aşamasında olan dokümanlardır.



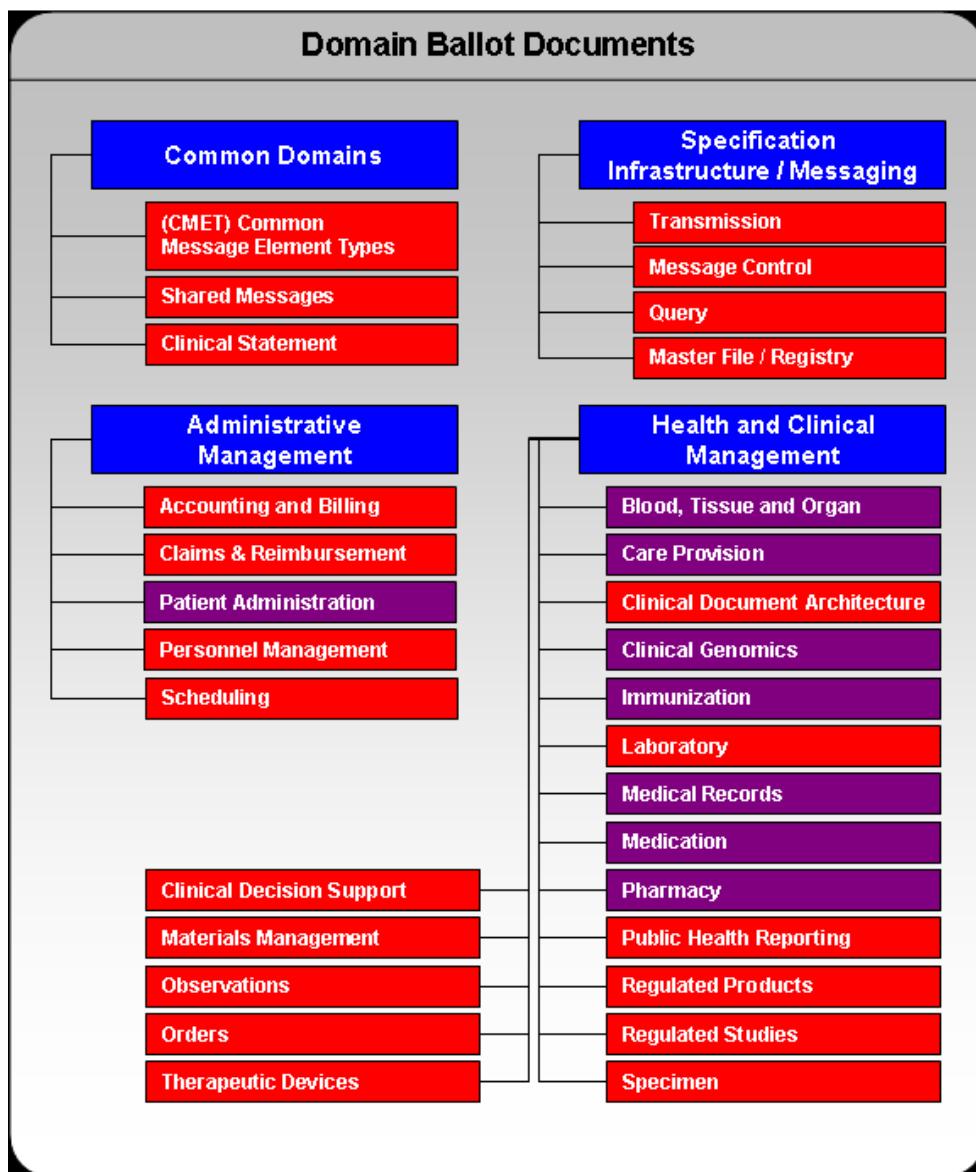
Şekil 13 HL7 v3 Oylama Sitesi Giriş Sayfası

Bu doküman ayrimına ek olarak kullanım durumuna göre de dokümanlar ikiye ayrılırlar:

1. Temel (Foundation Dokümanlar): Bu dokümanlar belli bir sağlık alanına özel olmayıp mesaj geliştirme aşamasında Teknik Komitelere veya uygulamacılara yardımcı olan dokümanlardır. Referans Bilgi Modeli, Mesaj Geliştirme Çerçeve dokümanları bu kategoriye örnektir. Şekil 14'da temel dokümanlar sunulmaktadır.
2. Alan (Domain Dokümanlar): Eczane, Laboratuvar veya Hasta Yönetim gibi bir Alana yönelik dokümanlardır. Bu dokümanların içerisinde belli bir alanda Teknik Komiteler tarafından geliştirilmiş DMIM'ler, RMIM'ler, HMD'ler ve mesaj şemaları sunulmaktadır. Ayrıca storyboard'lar, uygulama roller ve etkileşimler de sunulmaktadır. Şekil 15'da şu an HL7 v3 tarafından kapsanan alanlar sunulmaktadır.



Şekil 14 Temel Dokümanlar



Şekil 15 Alan Dokümanları

2.3.7 HL7 İletişim Spesifikasyonları

V3'ten önceki sürümlerde HL7, birlikte çalışabilirlik katmanlarının (interoperability stack) alt seviyelerine fazla karışmazdı. Ama V3 ile beraber, HL7 kullanılacak iletişim protokolünün özellikleri hakkında da yönnergeler çıkarmaktadır. Hatta "HL7 is not level 7 any more (HL7 artık sadece seviye 7 değil)" denmeye başlanmıştır. Bu yönde ilk olarak "Soyut Transport Spesifikasyonu"¹⁸ (Abstract Transport Specification) yayınlanmıştır. Bu spesifikasyonda, HL7 mesajlarının ilettilmesinde kullanılacak olan bir transport protokolünün olması gereken özellikleri sunulmuştur. Örneğin güvenlilik, adresleme ve güvenilirlik. Daha sonra teknik komiteler bu gereksinimleri karşılayan protokollerini birer profil halinde sunmuşlardır:

¹⁸ <http://www.hl7.org/v3ballot/html/infrastructure/transport/transport-abstract.htm>

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

1. ebXML Profili¹⁹: Soyut Mesaj Spesifikasyonunda belirtilen gereksinimler ebXML Mesajlaşma Standardı kullanılarak nasıl karşılanır bilgisini anlatan profildir.
2. Web Servis Profili²⁰: Soyut Mesaj Spesifikasyonunda belirtilen gereksinimler Web Servisleri kullanılarak nasıl karşılanır bilgisini anlatan profildir.
3. Asgari Alt Katman Protokolü Profili (Minimal Lower Layer Protocol Profile)²¹: Soyut Mesaj Spesifikasyonunda belirtilen gereksinimler en basitinden TCP/IP protokolü ile nasıl karşılanır bilgisini anlatan profildir.

Bu protokollerden kullanılması en olası olanı Web Servis Profilidir. Bu profilde bir HL7 v3 mesajı alacak olan bir Web Servisinin dört çeşit gereksinimi anlatılmaktadır:

1. Temel (Basic) Gereksinimler: WSDL ve SOAP mesajlarının nasıl olması gerektiği hakkında detaylı bilginin verildiği bölümdür.
2. Adresleme (Addressing) Gereksinimleri: Bu Web Servisine ulaşmak için gerekli adresleme bilgilerinin gereksinimlerinin WS-Addressing²² ile nasıl başarılıacağını anlatır.
3. Güvenlik (Security) Gereksinimleri: Uygulanacak olan Web Servisinin kullanacağı güvenlik standardlarının neler olduğu ve nasıl bir güvenlik kurulumu yapılması gerektiği bu bölümde anlatılır. WS-Security²³, WS-Trust²⁴, WS-SecureConversation²⁵ ve WS-SecurityPolicy²⁶ standardlarında dayalıdır.
4. Güvenilirlik (Reliability) Gereksinimleri: Güvenilirlik için gereken standardların ve bu standardların nasıl kullanılacağı detayları olarak anlatıldığı bölümdür. WS-ReliableMessaging²⁷ tabanlıdır.

¹⁹ <http://www.hl7.org/v3ballot/html/infrastructure/transport/transport-ebxml.htm>

²⁰ <http://www.hl7.org/v3ballot/html/infrastructure/transport/transport-wsprofiles.htm>

²¹ <http://www.hl7.org/v3ballot/html/infrastructure/transport/transport-mllp.htm>

²² <http://www.w3.org/Submission/ws-addressing/>

²³ <http://www.oasis-open.org/committees/wss/>

²⁴ <http://specs.xmlsoap.org/ws/2005/02/trust/WS-Trust.pdf>

²⁵ <http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512>

²⁶ <http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

²⁷ <http://www.ibm.com/developerworks/library/specification/ws-rm/>

2.4 HL7 Araçlarını kullanarak örnek bir MSVS Muayene HL7 v3 mesajı geliştirme

Bu bölümde bir HL7 mesajının, HL7'İN sağladığı yazılım araçları ile nasıl tasarlandığı sunulmaktadır. HL7 Teknik Komiteleri mesaj tanımlamak için bu araçları kullanmaktadır. Ayrıca kullanıcılar bölüm 2.3.2 de bahsedilen yönergeleri bu araçları kullanarak gerçekleştirebilirler.

2.4.1 Kurulum İçin Hazırlık

Bu başlık altında, HL7 araçlarının kurulumu için gerekli olan tüm dosyalar ve bu kurulum dosyalarının nerelerden edinilebileceği listelenecaktır. Asıl kurulum ise bir sonraki başlıkta anlatılacaktır.

Kurulum için gerekli olan yazılımlar aşağıda listelenmiştir. Microsoft Office Visio dışında tüm yazılımların indirebileceği adres: <http://www.hl7.org.au/HL7-V3-Resrcs.htm>. Visio dışında tüm araçlar ücretsiz olarak dağıtılmaktadır.

1. Microsoft Office Visio
 - HL7 Rafine Mesaj Bilgi Modeli (Refined Message Information Model - R-MIM) geliştirilmesi Visio üzerinden yapılmaktadır.
 - HL7 sürüm olarak 2002'yi tavsiye etse de, farklı bilgisayarlaraya yapılan kurulumlar sayesinde görüldü ki 2003 ve 2007 sürümünde de HL7 RMIM Designer sorunsuz olarak çalışmaktadır. Ancak 2002'den önceki sürümler desteklenmemektedir.
 - Kurulacak Visio'nun dili önemli değildir.
2. HL7 RIM Repository
 - RIM deposunun Microsoft Access veri tabanı formatındaki en son sürümü.
 - Adres: http://hl7projects.hl7.nscee.edu/frs/?group_id=32
3. HL7 RMIM Designer Visio
 - RMIM Designer Visio'ya bir eklenti olarak yüklenip, RIM'den RMIM türetmeyi sağlayan görsel bir dizayn aracıdır.
 - Adres: http://hl7projects.hl7.nscee.edu/frs/?group_id=21
4. HL7 RoseTree
 - RoseTree girdi olarak RMIM alıp, HL7 Hiyerarşik Mesaj Tanımı (Hierarchical Message Definition - HMD) üreten, tek başına çalışan bir uygulamadır.
 - Adres: http://hl7projects.hl7.nscee.edu/frs/?group_id=31
5. HL7 v3 Generator Tool
 - HL7 v3 Generator Tool girdi olarak HMD alıp, HL7 v3 mesaj tanımlarını XSD şemaları, tablo görüntüleri ve excel dosyaları olarak üreten, komut satırından çalışan bir uygulamadır.
 - Bu araç Java Runtime 1.4.2 ve yukarısını istemektedir. 1.4.2 sürümüne <http://java.sun.com/j2se/1.4.2/download.html> adresinden, en son sürüm olan Java 6 Runtime Environment'e ise <http://java.sun.com/javase/downloads/index.jsp> adresinden ulaşılabilir.
 - Adres: http://hl7projects.hl7.nscee.edu/frs/?group_id=24

2.4.2 Kurulum

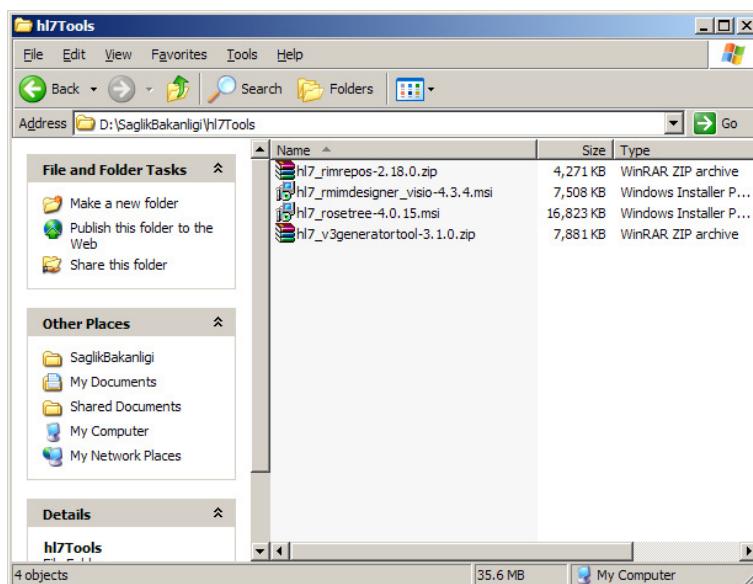
Bu başlık altında kurulum için gerekli olan tüm işlemler, adımlar halinde ve ekran görüntüleri desteğiyle anlatılacaktır.

Adım 1:

- Microsoft Office Visio 2002, 2003 veya 2007 standart konfigürasyon ile kurulur.
- Java Runtime Environment yoksa eğer, o da standart konfigürasyon ile kurulur.

Adım 2:

- Bir önceki adımda 2, 3, 4 ve 5 no'lu maddeler olarak listelenen araçların tümü bir klasöre atılır. Hepsinin aynı klasörde olması bir zorunluluk değildir. Burada HL7 v3 Generator Tool – Java ilişkisinden kaynaklanan tek bir koşul vardır, o da araçların yüklediği klasörün ve onun üzerindeki tüm klasörlerin isimlerinde Türkçe karakter bulunmamasıdır.
- Bu dokümandaki örnekte tüm araçlar “D:\SaglikBakanligi\hl7Tools” klasörüne yüklenmiştir.

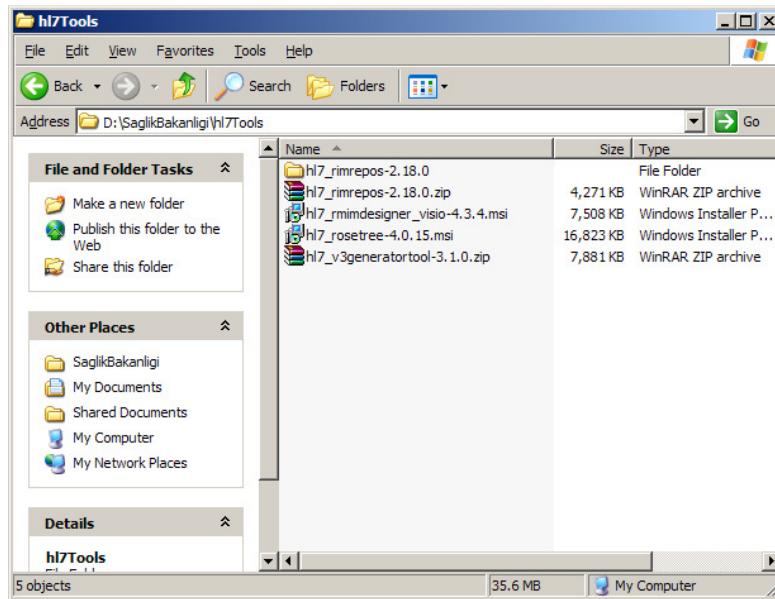


Şekil 16 Dizin formatı

Adım 3:

- RIM Repository'yi içeren “hl7_rimrepos-2.18.0.zip” dosyası açılarak içeriği çıkartılır (“extract”).

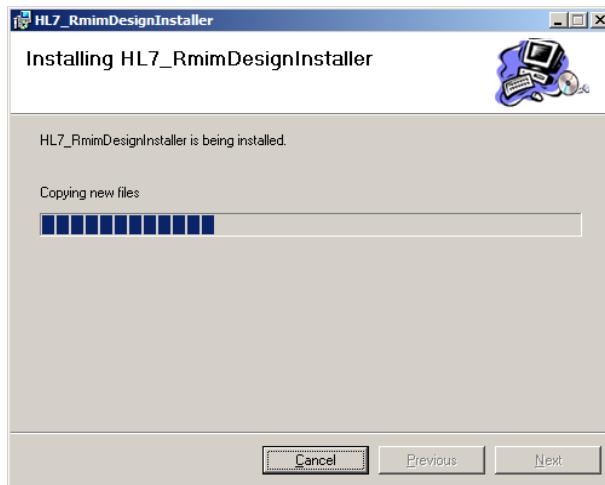
T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



Şekil 17 RIM Deposunun açılmış hali

Adım 4:

- hl7_rmimdesigner_visio-4.3.4.msi dosyası kurulur. Bu aşamada tek yapılması gereken "next" komutlarıyla ilerlemektir.



Şekil 18 RMIM Designer kurulumu

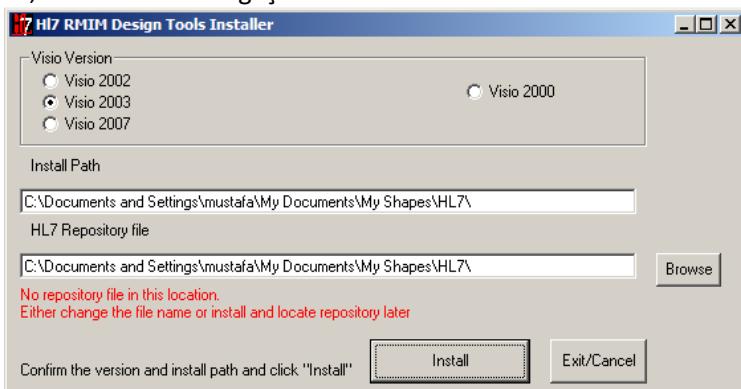
- Bu dosya aslında RMIM Designer'ın kendisini değil de, RMIM Designer'ı asıl yükleyecek olan kurucuyu yüklemektedir.
- Windows'un "Başlat" menüsünün üst kısmında "HL7_RMIMDesignerInstaller" adında yeni bir kısayol eklenmiştir. Bu kısayol çalıştırılarak asıl kurulum işlemine geçilir.

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



Şekil 19 RMIM Designer Başlatılması

- Bu kısayolun seçilmesiyle asıl yükleyici çalıştırılır ve bu yükleyici açılır açılmaz sistemde yüklü olan Microsoft Visio'yu bulmaya çalışır. Visio versiyonu 2003 veya 2007 ise desteklenmediğine dair bir uyarı mesajı döner, önemsenmeden geçilmelidir.

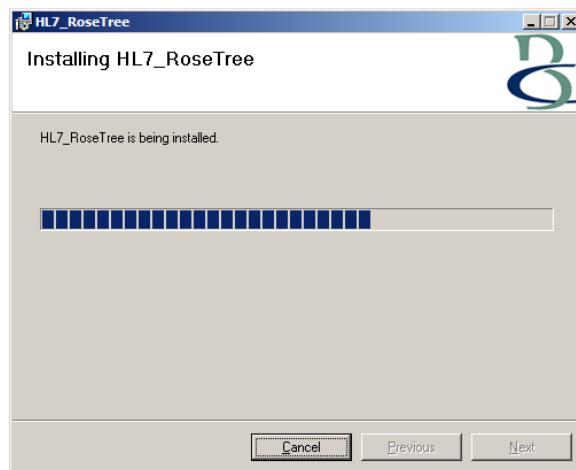


Şekil 20 RMIM Designer Konfigurasyonu

- Şekil 20'de görüldüğü gibi Visio sürümünün düzgün olarak seçildiğine emin olduktan sonra "Install" düğmesine tıklanır. Bu aşamada Visio'nun açık olmaması gereklidir. Kurulum çok kısa sürede tamamlanarak bir geri bildirim mesajı döner.

Adım 4:

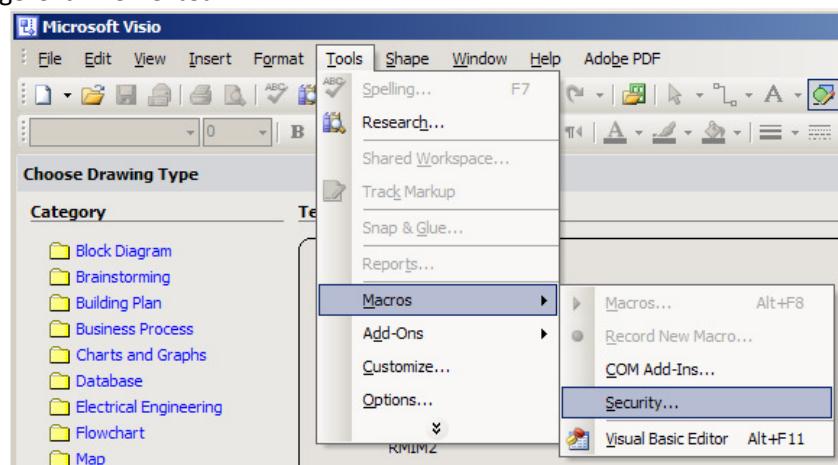
- Şu ana kadar kurulması gereken 4 araçtan 2'si başarıyla kurulmuştur. Bu adımda kurulacak olan yazılım RoseTree'dir.
- Ana klasörümüzde yer alan hl7_rosetree-4.0.15.msi dosyası çalıştırılır. Yine standard bir şekilde "next" adımları ile ilerlenerek kurulum tamamlanır.



Şekil 21 RoseTree kurulumu

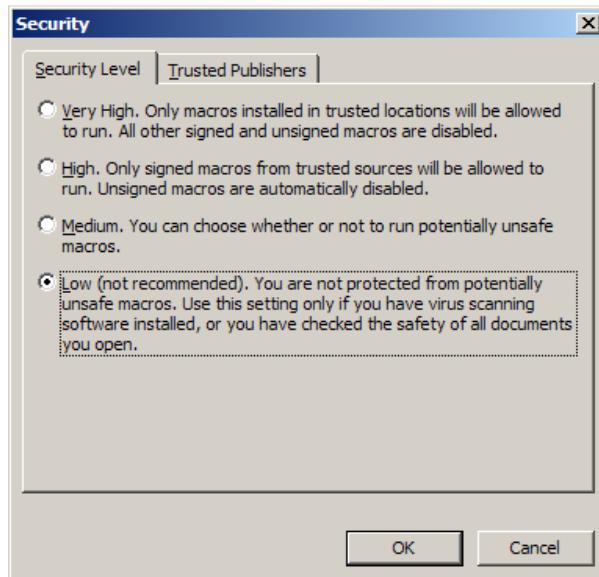
Adım 5:

- Gerekli olan 4 yazılımdan 3'ü başarıyla kuruldu. Bu adımda RMIM Designer'in Visio içinde düzgün bir şekilde çalışıp çalışmadığından emin olmaya yönelik bazı aktiviteler yapılacaktır.
- Microsoft Office Visio başlatılır.
- RMIM Designer menüsüne geçilmeden önce yapılması gereken kritik bir işlem vardır. RMIM Designer'ı çalıştırabilmek için Visio'nun "Macro"ları çalıştırmasına izin vermek gereklidir.
- Bunun için gerekli ayarlar Visio 2003'te Tools → Macros → Security altında yer almaktadır. Visio 2007'de de yine benzer bir menüde ayarlar bulunabilir. Visio 2002 HL7'nin açıklamalarına göre bu ayarlamayı gerektirmemektedir.



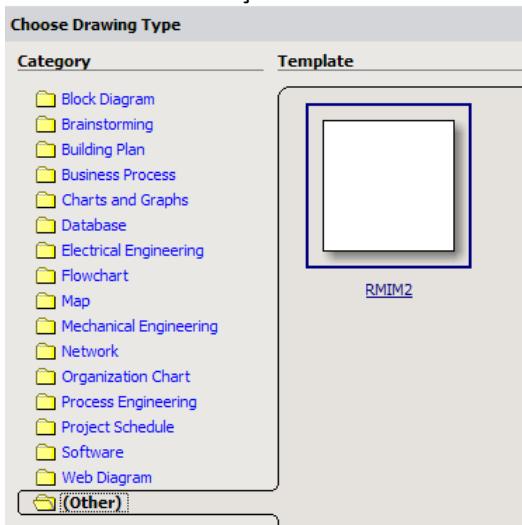
Şekil 22 Macroların etkinleştirilmesi

- Bu menü seçilir. Açılan yeni pencerede güvenlik seviyesi ("Security Level") olarak en düşük seviye ("Low") seçilir. Ayarlar kaydedildikten sonra Visio kapatılarak yeniden başlatılır.



Şekil 23 Makroların etkinleştirilmesi - 2

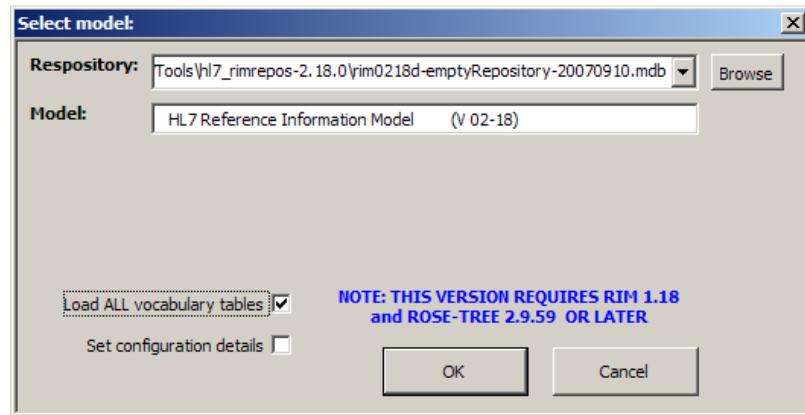
- Yeniden başlatılan Visio ekranında sol blokta kategoriler listelenir. Bu kategorilerden “Other” seçilir. Sağda açılan şablon ekranında ise “RMIM2” seçilir.



Şekil 24 HL7 RMIM Stencil'inin seçilmesi

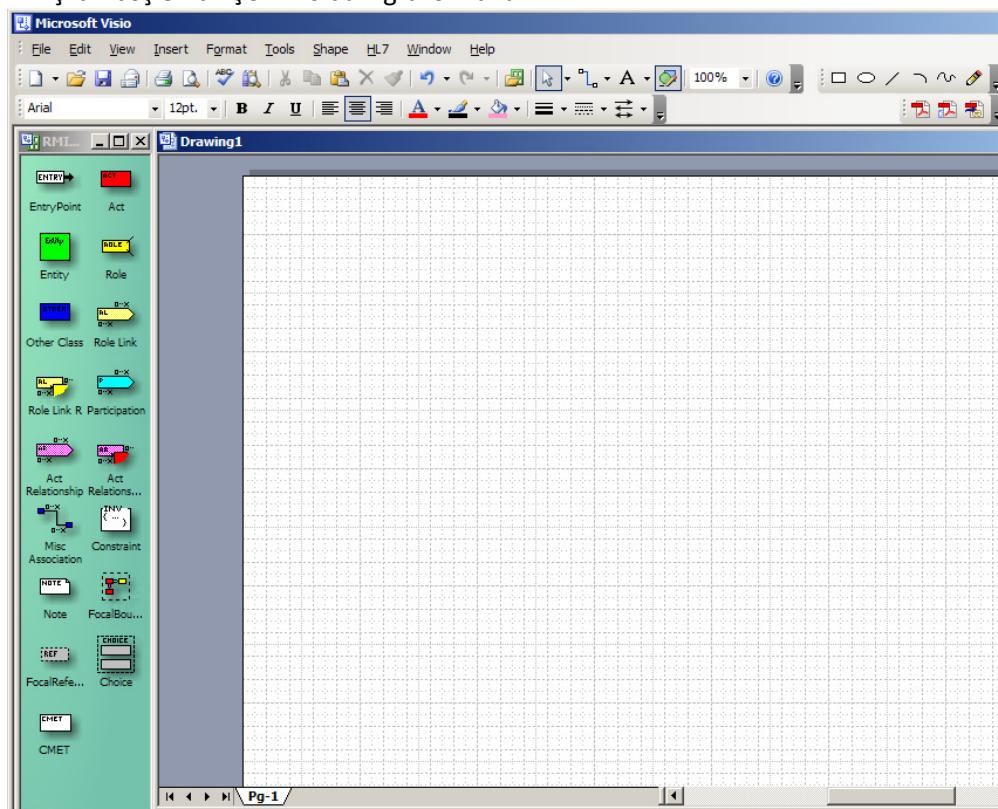
- RMIM2 şablonu seçildikten sonra Visio bir iki uyarı mesajı dönebilir, önemsizdir geçilebilir.
- Yeni açılan ufak bir pencerede RMIM Designer, RIM repository'nin adresini ister. Hatırlanacağı üzere, Adım 3'te RIM repository “D:\SaglikBakanligi\hl7Tools\hl7_rimrepos-2.18.0” klasörüne kaydedilmiştir. Aşağıdaki figürde görülen “browse” butonu seçilerek, klasörde bulunan Access veri tabanı dosyası (.mdb uzantılı) seçilir.

T.C. Sağlık Bakanı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



Şekil 25 RIM Deposunun seçilmesi

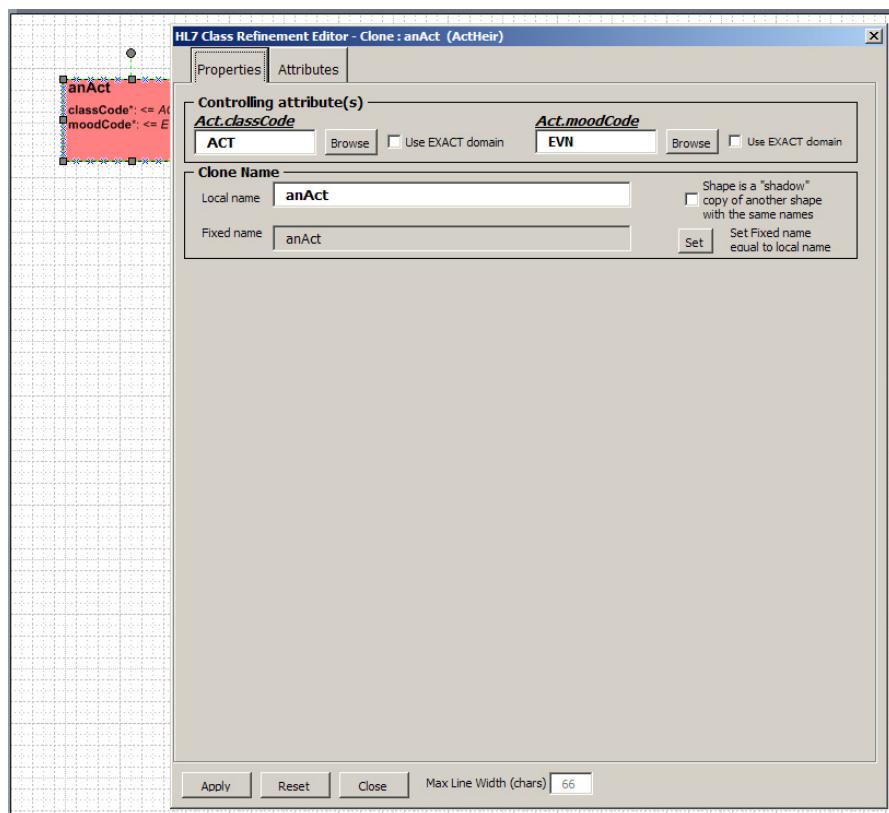
- “Repository” seçildikten sonra “Model:” alanın dolması için, “Repository” alanı seçiliyken bir kez “Tab” tuşuna basılır. “Model:” alanının figürde görüldüğü üzere dolması beklenir.
- Figürde sol alta yer alan seçeneklerden “Load ALL vocabulary tables” seçeneği işaretlenir.
- Son olarak “OK” tuşuna basılarak RMIM tasarımcı ekranının açılması beklenir. Bu işlem biraz zaman alabilir. Açılan boş ekran Şekil 26’da gibi olmalıdır.



Şekil 26 RMIM Designer başlangıç sayfası

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

- Bu figürde sol tarafta RIM Repository'nin içerisinde var olan sınıflar bulunmaktadır. Tasarım bir sonraki başlıkta anlatılacaktır ancak kurulumun başarılı olduğundan emin olmak için, sol bloktan "Act", "Entity" veya "Role" gibi sınıflardan biri seçilerek sağ taraftaki boş ekran sürüklenir.
- Örneğin, bir "Act" sağ tarafa sürüklendiği durumda, Şekil 27'de görüldüğü gibi bir "Class Refinement Editor" açılıyor ise kurulum başarılı olmuş demektir.



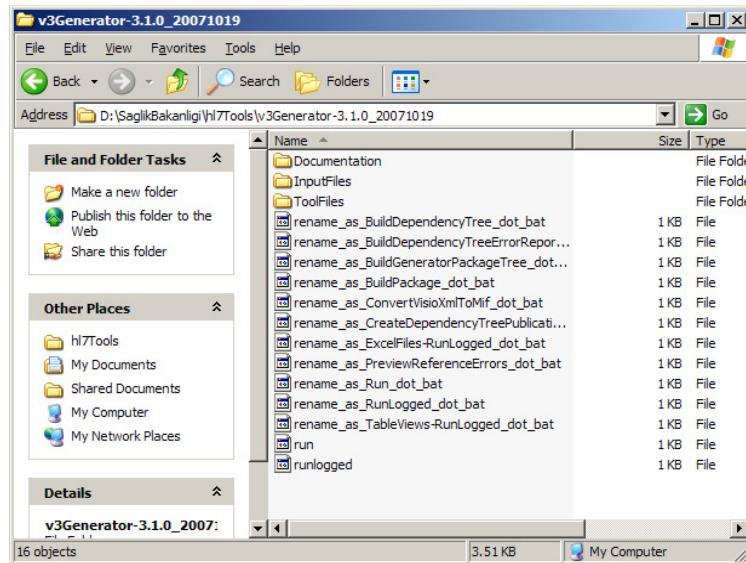
Şekil 27 Bir Act sınıfı ekleme

- Değişiklikler kaydedilmeden Visio kapatılabilir. Kurulumla ilgili son bir adım kaldı.

Adım 6:

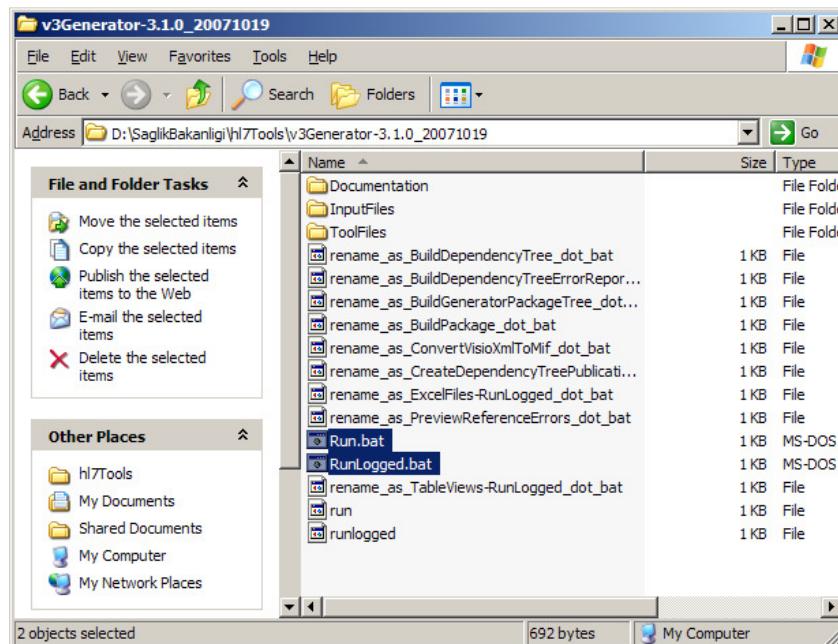
- İlk başlıkta tanıtılan 4 araçtan 3'ünün kurulumu bitmişti, şimdi sıra sonucusunda. Bu adımda kurulacak olan yazılımın adı HL7 v3 Generator Tool.
- Ana klasörümüzde yer alan "hl7_v3generatortool-3.1.0.zip" dosyası açılarak bulunduğu klasöre boşaltılır ("extract").
- Boşaltma işleminin sonunda "D:\SaglikBakanligi\hl7Tools\v3Generator-3.1.0_20071019" klasörüne uygulama yüklenmiş oldu.

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



Şekil 28 v3Generator kurulumu

- Kurulumun tamamlanması için Şekil 28'de görülen dosyalardan iki tanesinin yeniden isimlendirilmesi gerekmektedir. Esasen bu dosaların tamamı da dosya isimlerinde açıklandığı gibi yeniden isimlendirilebilir. Bu aşamada sadece kullanılacak olanların isimlendirilmesi yapılacaktır.
- “rename_as_Run_dot_bat” isimli dosya “Run.bat”, “rename_as_RunLogged_dot_bat” isimli dosya ise “RunLogged.bat” şeklinde yeniden isimlendirilir.



Şekil 29 v3Generator kurlumu

- Bu adımla birlikte kurulum işlemi son bulmuştur. Artık HL7 araçları kullanılmaya hazır hale gelmiştir.

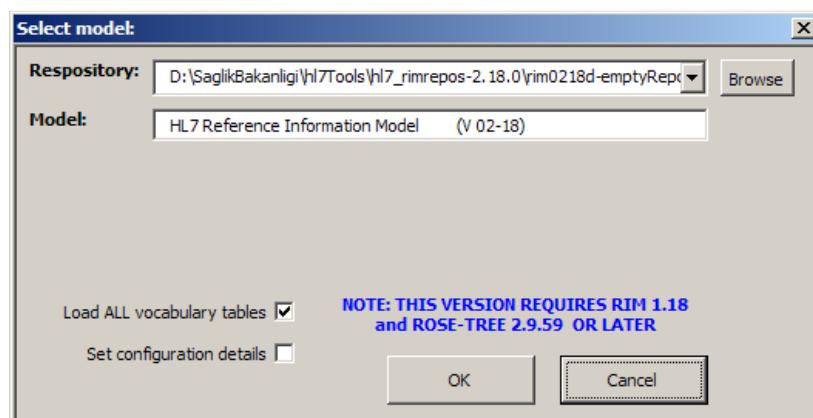
2.4.3 Kullanım

Bu başlık altında, kurulumu tamamlanan HL7 araçlarının kullanımı anlatılacaktır.

1. HL7 RMIM Designer Visio: RIM → D-MIM ve D-MIM → R-MIM aşamalarını tek bir aracılıkla gerçekleştirir.
2. HL7 RoseTree: R-MIM → HMD aşamasını gerçekleştirir.
3. HL7 v3 Generator Tool: HMD → XSD Şema aşamasını gerçekleştirir.

2.4.4 HL7 RMIM Designer Visio Kullanımı

HL7 RMIM Designer'ın kullanımı Visio'ya entegre yapısı sayesinde oldukça kolaydır. Kullanıma başlamak için Microsoft Visio açılır. Kurulum aşamasında zaten Access dosyası halindeki RIM Repository RMIM Designer'a parametre olarak verildiğinden, açılan ufak pencerede "OK" demek yeterlidir. Visio ayarları saklamaktadır.



Şekil 30 RMIM Designer kullanımı

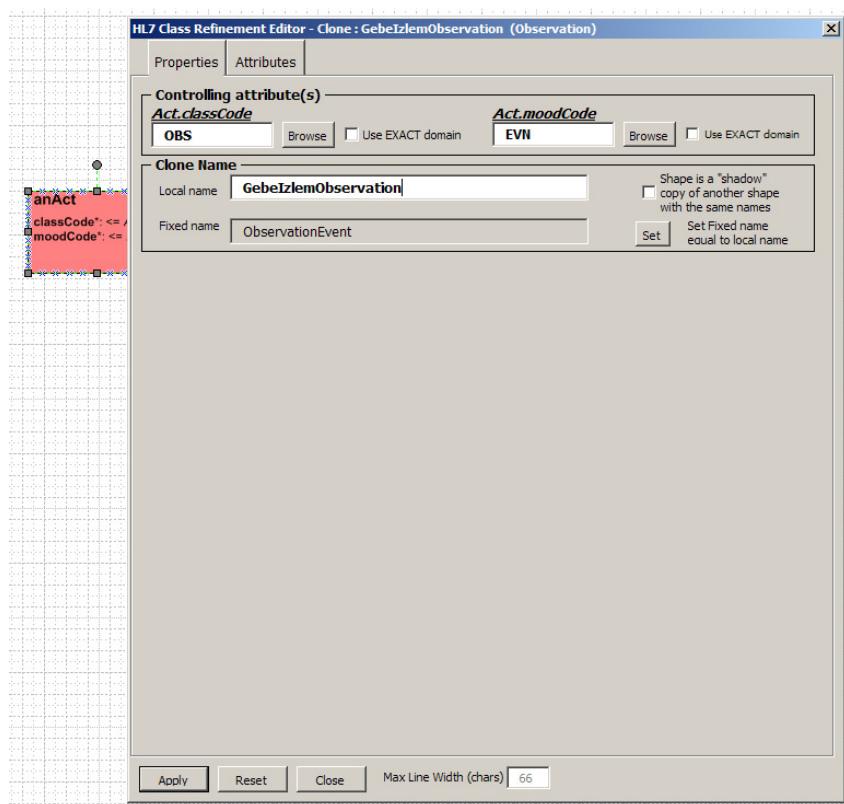
Daha sonra Visio tasarım ekranı açılır. Bu ekranda tasarım RIM, Mesaj Geliştirme Çerçeve (MDF) ve HL7 Geliştirme Çerçeve'ne (HDF) uygun bir şekilde yapılır.

Sol blokta yer alan RIM sınıflarından ihtiyaç duyulanlar seçili, sağ taraftaki çizim alanına taşınır. Her yeni taşınan sembolden sonra, o sınıfın ait yeni bir kopya oluşturulur ve bu kopya sınıfın detaylı özellikleri kullanıcıdan otomatik olarak açılan "Class Refinement Editor" penceresi vasıtasyyla istenir.

Örneğin aşağıdaki figürde Gebe İzlem MSVS'sini modellemek amacıyla ilk olarak "Act" sınıfının bir kopyası oluşturulmuştur. Bu "Act" kopyası aslında bir "Observation" şeklinde modelleneneğinden, figürdeki pencerede Act.classCode değeri olarak "OBS" seçilmiştir. Act.moodCode değeri ise event anlamına gelen "EVN" seçilmiştir. Burada tüm sınıflar için var olan classCode ve moodCode gibi HL7 tarafından tanımlanmış sözlükler eğitimler kapsamında RIM konusu işlenirken anlatılacaktır.

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Kopyalanan sınıf için yerel bir isim vermek mümkündür. Örneğin Şekil 31'de "GebelzlemObservation" tercih edilmiştir.

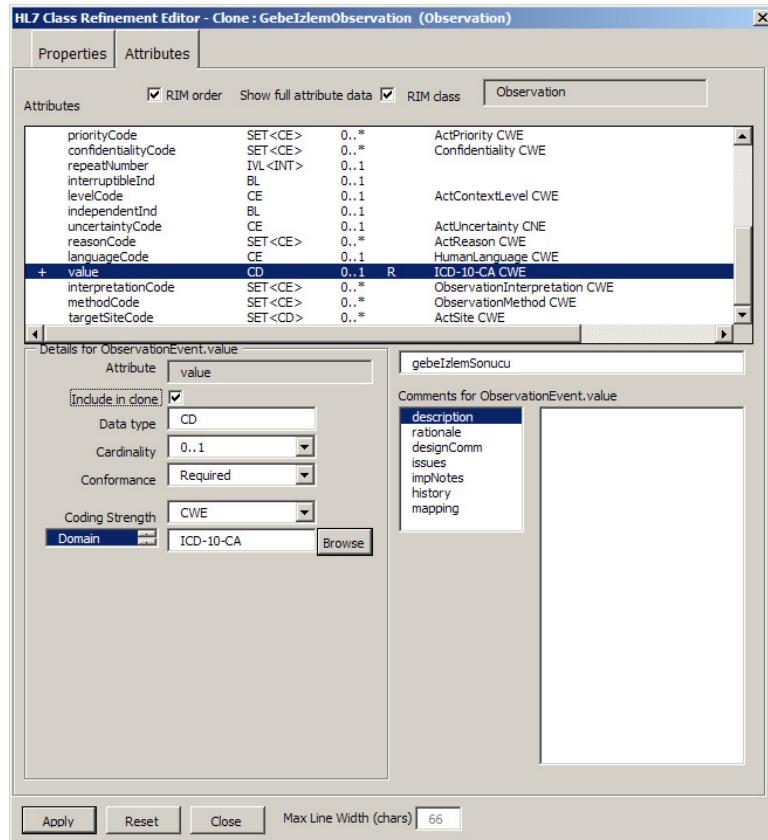


Şekil 31 RMIM Act Sınıfı Ekleme

Kopya sınıfla ilgili temel özellikler seçildikten sonra "Class Refinement Editor" penceresinin "Attributes" sekmesi seçilmelidir. Bu pencerede kopyalanan sınıfın içereceği elemanlara ait özelliklerin konfigürasyonu yapılır. Bu pencerede listelenen elemanlar çift tıklanarak veya "include in clone" seçeneği seçilerek kopya sınıfa dahil edilebilir. Bir eleman bu şekilde dahil edildikten sonra bazı özelliklerinin ayarlanması beklenir. Aşağıdaki figürde bu özellikler görülebilir.

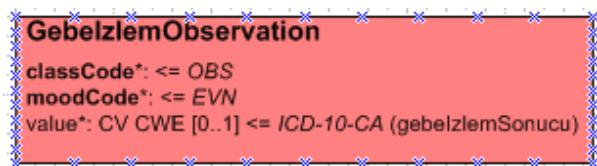
Bu figürde varsayılan olarak seçili olmayan "value" elemanı "GebelzlemObservation" kopya sınıfına dahil edilmiştir. Veri tipi varsayılan olarak "ANY" iken, kısıtlandırılarak "Coded Value" (CV) seçilmiştir. Veri tipleri yine eğitimler kapsamında detaylı olarak anlatılacaktır.

Bunların dışında bu elemanın zorunlu mu yoksa seçimi mi olduğunu, kaç adet bulunması gerektiğini, eğer kodlanmış bir değer ise hangi "Vocabulary Domain"ı kullanacağını belirtmek bu ekran sayesinde mümkündür. Elemana tipki kopya sınıfta olduğu gibi yerel bir isim vermek de mümkündür. Örneğin burda "gebelzlemSonucu" olarak adlandırılan elemanın bulunması zorunlu bir eleman olduğu, değerlerini ise ICD-10 sınıflandırma sisteminden alacağı belirtilmiştir. Tüm elemanların özellikleri bu şekilde ayarlandıktan sonra yapılması gereken işlem "Apply" tuşuna basmaktır.



Şekil 32 RMIM Act Sınıfının Özelliklerini Seçme

Bu ufak modelleme işleminin sonucu Şekil 33'deki gibi görüntülenir Visio tarafından:

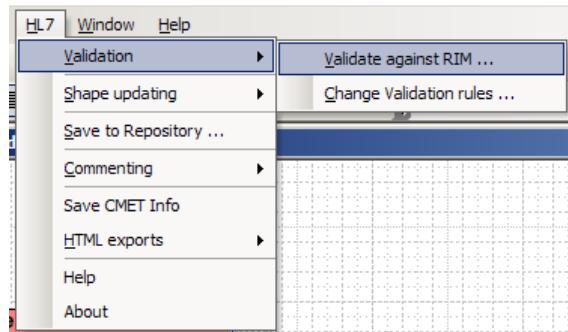


Şekil 33 Gebelzlem Sınıfı

Bu şekilde bir mesaja ait tüm sınıflar ve sınıflar arasındaki ilişkiler modellenerek tasarım işlemi sonlandırılır. Son olarak mesaja bir adet "Entry Point" eklenmelidir.

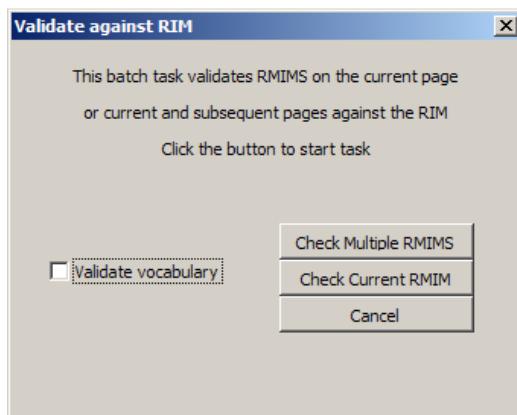
Tasarım işlemi sonlandıktan sonra öncelikli olarak File → Save As menüsü yardımıyla Visio dosyası saklanmalıdır. Bu işlemin en başta yapılması bilgi kaybı riskine daha güvenlidir.

Daha sonra, oluşturulan mesajın RIM Repository'ye kaydedilme aşaması başlar. Bunun için öncelikle oluşturulan mesajın "validate" edilmesi gereklidir. Seçilmesi gereken menü Şekil 34'de gösterilmiştir:



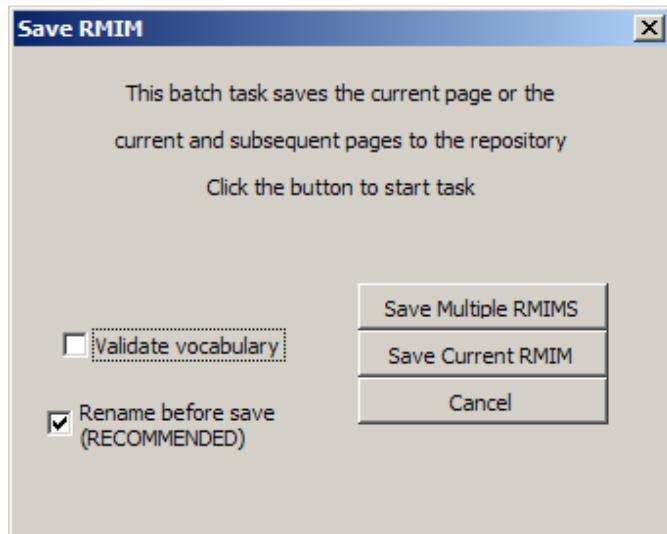
Şekil 34 Kontrol Etme

Açılan pencerede (aşağıda görüldüğü gibi), eğer tasarımda HL7 tarafından tanımlanmamış “Vocabulary Domain” ler kullanılıyorsa (ki USVS’den mesaj türetme konusunda bu böyledir), “Validate Vocabulary” seçeneği boş bırakılarak “Check Current RMIM” düğmesine basılır.



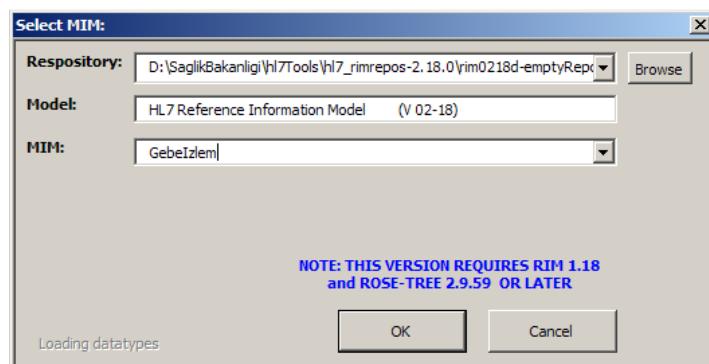
Şekil 35 RMIM Seçme

Doğrulama işlemi eğer varsa hataları listeler ve bunların düzeltilmesi için grafiksel olarak yardımcı olur. Hatalara çift tıklayarak model üzerinde hata görüntülenebilir. Hatalar düzeltildikten sonra RIM Deposuna kaydetme işlemine geçilebilir. Yapılması gereken HL7 → Save to Repository ... menüsünün seçilmesidir. Bu menünün seçilmesiyle açılan pencerede yine “Validate Vocabulary” seçeneği boş bırakılır ve “Save Current RMIM” tuşuna basılır.



Şekil 36 RMIM Seçme Konfigurasyonu

Açılan yeni pencere Mesaj Bilgi Modeli’ne (MIM) bir isim verilmesini bekler. Bu örnekte “Gebelzlem” ismi verilir ve “OK” tuşuna basılır.



Şekil 37 RIM Deposunu Seçme

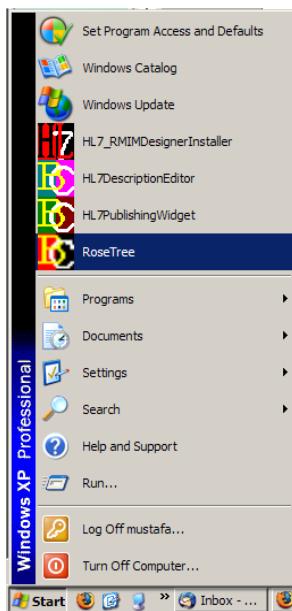
OK tuşuna basıldıktan sonra kaydedildi (“saved”) uyarısının çıkması beklenir. Bu son adımla HL7 RMIM Designer’ın kullanımı sonlanır.

2.4.5 HL7 RoseTree Kullanımı

RoseTree RMIM Designer ile başlanan modelleme işlemine devam etmeyi sağlar ve RMIM’den HMD üretilmesini sağlar.

RoseTree’yi başlatmak için Başlat menüsünden RoseTree’yi seçmek yeterlidir.

T.C. Sağlık Bakanı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



Şekil 38 RoseTree başlatma

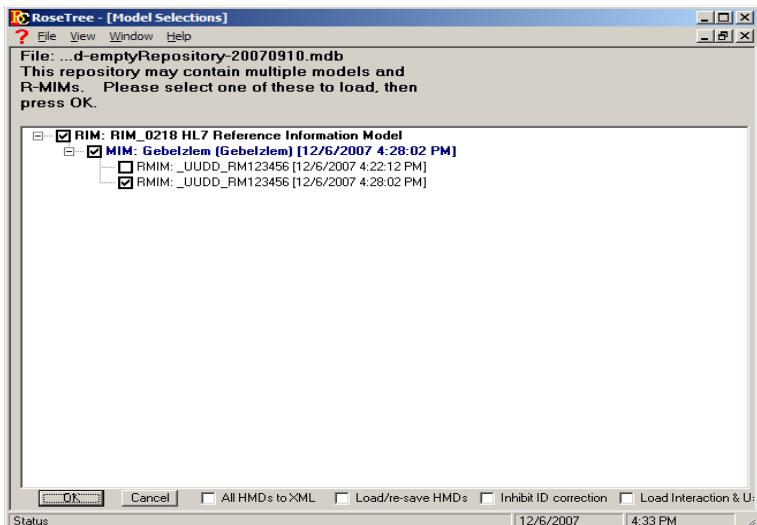
RoseTree açıldıktan sonra File → Open menüsü ile RIM Repository'sinin bulunduğu Access veri tabanı dosyasını göstermek gereklidir.



Şekil 39 RIM deposunu seçme

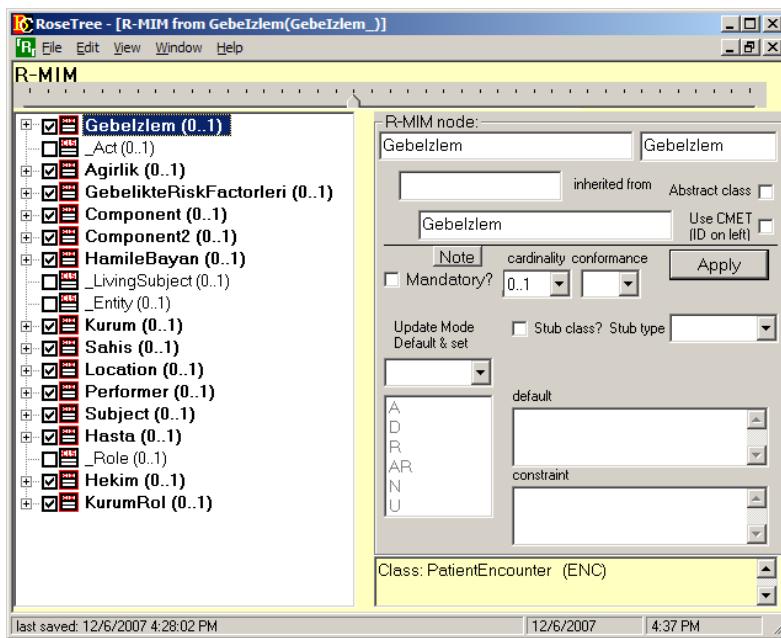
T.C. Sağlık Bakanı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Daha sonra güncellenen pencerede en son eklenen modelleri görmek mümkündür. Bu pencerede en alt seviye çocuklardan birini seçmek gerekmektedir. Gebelzlem örneğinde en alt seviyede farklı zamanlarda kaydetmekten kaynaklanan iki çocuk vardır. Aşağıdaki figürde görüldüğü üzere en güncel çocuk seçilmiştir. Daha sonra yapılması gereken sol alt köşede yer alan "OK" tuşuna basmaktadır.



Şekil 40 RMIM'i seçme

Bu işlem sonucunda geliştirilmiş olan R-MIM RoseTree uygulamasında Şekil 41'de görüldüğü üzere açılır:



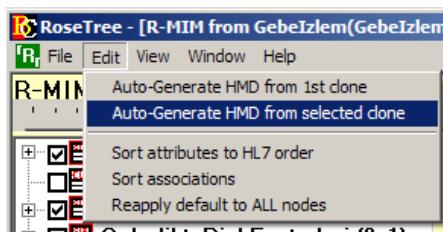
Şekil 41 RMIM'im uyarlanması

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Geliştirilen R-MIM üzerinde HL7 tarafından tanımlanan her türlü kısıtlama (“constraining”) işlemi yapmak mümkündür. Şekilde de görüldüğü üzere, sol bloktan seçilen model elemanlarının özellikleri sağ blokta görüntülenmekte ve aynı zamanda değiştirilebilmektedir.

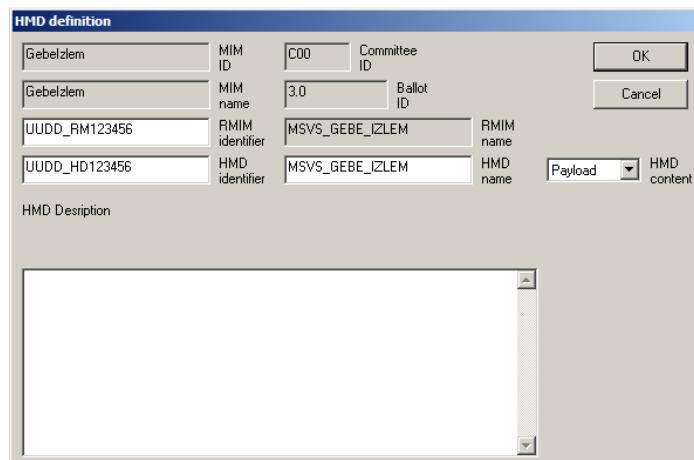
Bu aşamada değişiklik yapılrsa eğer sağ blokta yer alan “Apply” tuşu ile kaydedilmelidir. Daha sonra yapılması gereken ise HMD’nin oluşturulmasıdır. HMD’nin kendine has bir XML yapısı vardır ancak bunun detaylarının bilinmesi gerekli değildir.

HMD çıkarmak için sol blokta en tepede yer alan eleman seçildikten sonra Edit menüsünden “Auto-Generate HMD from selected clone” veya direk olarak “Auto-Generate HMD from first clone” menüsü Şekil 42’deki gibi seçilir:



Şekil 42 HMD Üretme

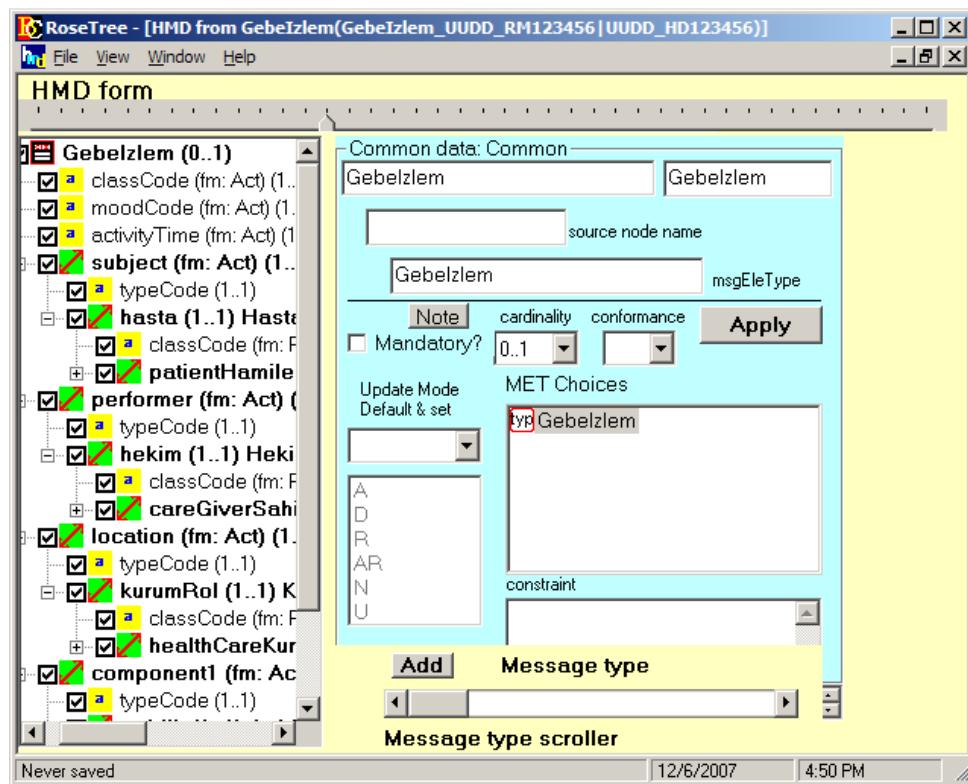
Bu menü HMD üretmeye başlar ve bir pencere ile kullanıcıdan HMD’ye ait bazı bilgiler ister:



Şekil 43 HMD İsimlendirme

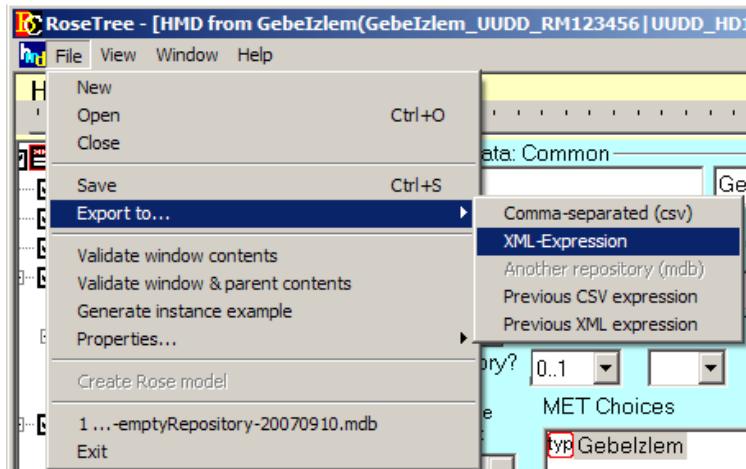
Bu ekranı ismi boş ya da yarımlar olarak gelen iki alan vardır: HMD identifier ve HMD name. Bunlar da yukarıda görüldüğü gibi isimlendirilebilir. Daha sonra OK tuşuna basılır.

Yeni açılan RoseTree ekranı bu sefer modelin HMD görüntüsünü gösterir:



Şekil 44 HMD Uyarlama

Bu ekranda da yine HMD seviyesinde yapılması mümkün olan kısıtlamaları yapmak mümkündür, örneğin bir alanın zorunlu olup olmadığını seçme gibi. Değişiklikler tamamlandıktan sonra RoseTree ile ilgili son adıma gelir; yani HMD'nin XML formatında kaydedilmesi. Bunun için yapılması gereken Şekil 45'te gösterilen menünün seçilmesidir:



Şekil 45 HMD'yi XML dosyasına kaydetme

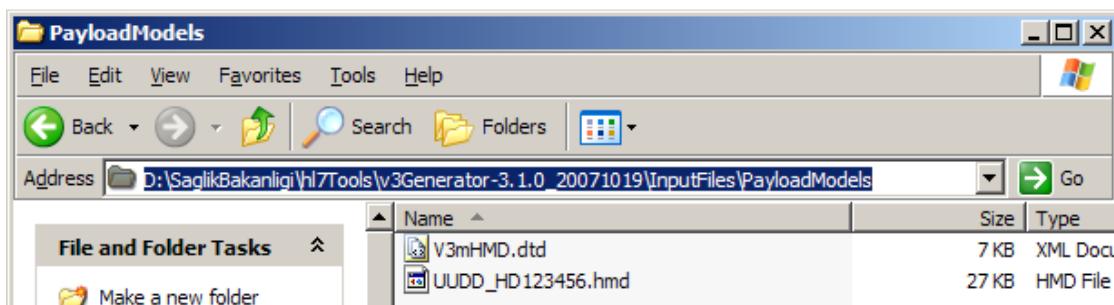
T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Bu ekran .hmd dosyasının kaydedileceği yeri sorar, herhangi bir adres verilebilir. Bu örnekte yine ana klasör olan “D:\SaglikBakanligi\hl7Tools” adresine kaydedilir. Kaydetme işleminden sonra RoseTree kapatılabilir.

2.4.6 HL7 v3 Generator Tool Kullanımı

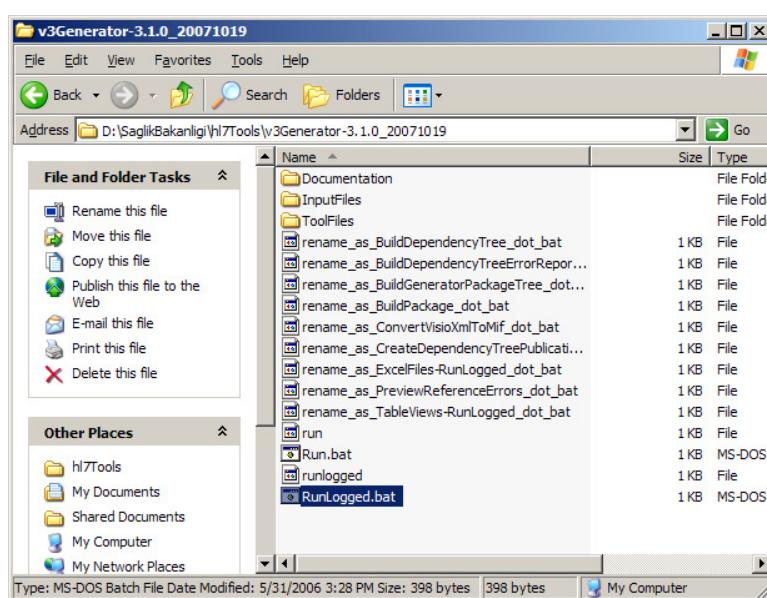
HL7 v3 Generator Tool, RoseTree'nin ürettiği HMD dosyasını girdi olarak alıp, XSD şemaları, tablo görüntüleri ve excel dosyaları olarak üretebilen, komut satırından çalışan bir uygulamadır.

Bu aşamada yapılması gerekenler oldukça basittir. Bir önceki adımda üretilen HMD dosyası, HL7 v3 Generator Tool ana klasörü altında, InputFiles altında PayloadModels klasörüne kopyalanır:



Şekil 46 v3Generator kullanımı

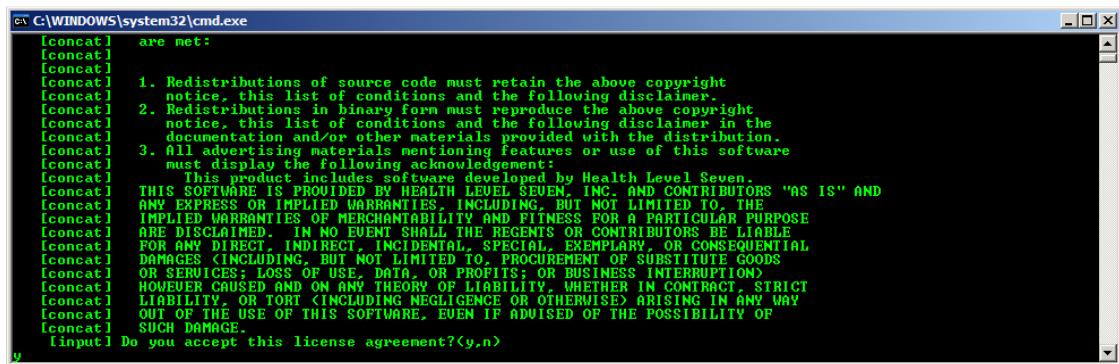
Daha sonra, HL7 v3 Generator Tool ana klasörüne gidilerek kurulum aşamasında oluşturulan “RunLogged.bat” komutu çalıştırılır.



Şekil 47 v3Generator çalıştırılması

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Açıldıktan hemen sonra uygulama size lisansı kabul edip etmediğinizi sorar, komut satırına "y" yazılıp devam edilir:



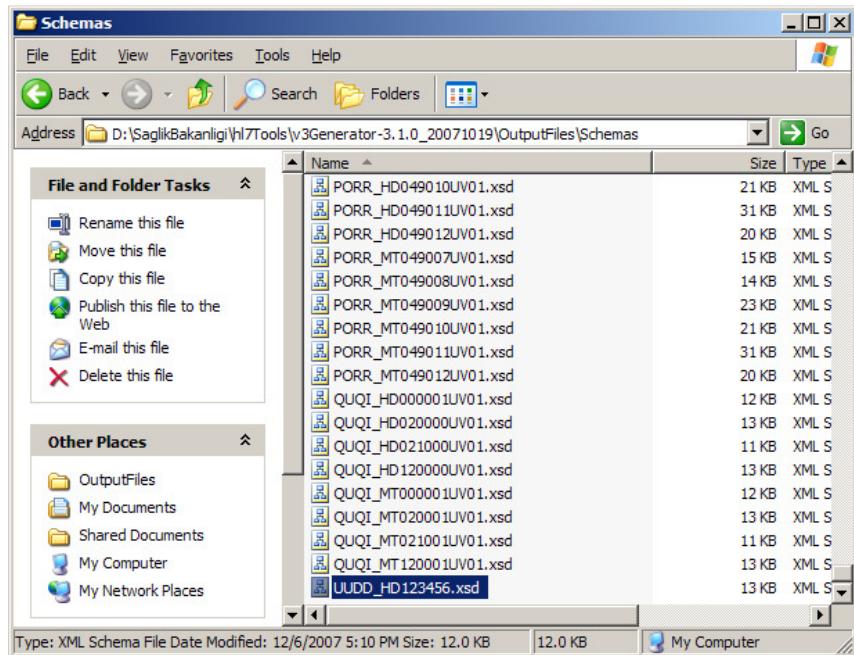
```
C:\WINDOWS\system32\cmd.exe
[concat] are met:
[concat]
[concat] 1. Redistributions of source code must retain the above copyright
[concat] notice, this list of conditions and the following disclaimer.
[concat] 2. Redistributions in binary form must reproduce the above copyright
[concat] notice, this list of conditions and the following disclaimer in the
[concat] documentation and/or other materials provided with the distribution.
[concat] 3. All advertising materials mentioning features or use of this software
[concat] must display the following acknowledgement:
[concat]   This product includes software developed by Health Level Seven.
[concat] THIS SOFTWARE IS PROVIDED BY HEALTH LEVEL SEVEN, INC. AND CONTRIBUTORS "AS IS" AND
[concat] ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
[concat] IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
[concat] ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
[concat] FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
[concat] DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
[concat] OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
[concat] HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
[concat] LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
[concat] OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
[concat] SUCH DAMAGE.
[concat] Input] Do you accept this license agreement?<y,n>
y
```

Şekil 48 v3Generator lisans kabulü

Lisansın kabul edilmesiyle XSD şema oluşturma işlemi başlatılmış olur. Bu adım biraz zaman alan bir işlemidir. Modelin büyüklüğüne göre değişmekle birlikte, 10 dakikadan fazla sürmesi normaldir. Uygulamanın işini bitirdiği komut satırında kayan yazılan sonlanmasıyla anlaşılabilir. Aynı zamanda uygulama şema oluşturma görevini bitirdiğini yazılı olarak da belirtir.

Uygulamanın sonlanmasıyla birlikte HL7 v3 Generator Tool ana klasörü altında iki yeni klasör oluştur: "OutputFiles" ve "TemporaryFiles".

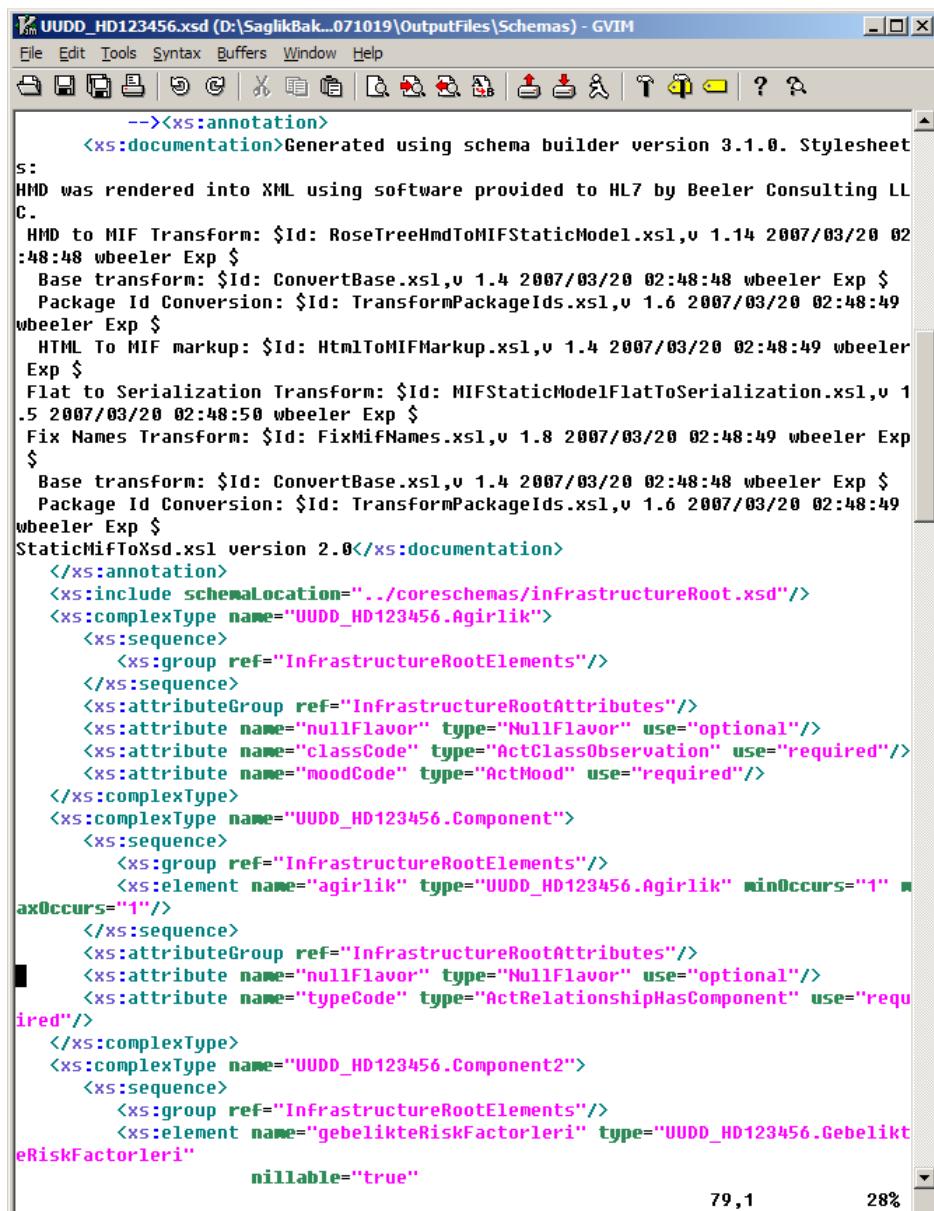
Oluşturulan XSD şeması OutputFiles altındaki Schemas klasörü altında, girdi olarak verilen HMD dosyasıyla aynı isimde oluşturulur. Burada anlatılan örnekte "UUDD_HD123456.xsd".



Şekil 49 v3Generator'a input verilmesi

XSD şeması herhangi bir XSD editörü ile veya yoksa, herhangi bir düz metin editörü ile açılıp incelenebilir, R-MIM ve HMD ile uygunluğu kontrol edilebilir. Bu adım HL7 araçları ile mesaj oluşturma dokümanının sonudur. Aşağıda oluşturulan XSD dosyasından bir kesit bulmak mümkündür. Bu dokümdanda kurulan tüm yazılımlar ve üretilen mesajlar Çıktı 1 – Ara Gelişme Raporu'na ek olarak sunulan CD içerisinde mevcuttur.

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)



The screenshot shows a GVIM window displaying an XML schema document named UUDD_HD123456.xsd. The code is color-coded to highlight XML tags and attributes. The schema defines several complex types, including 'Agirlik' and 'Component', which have attributes like 'nullFlavor' and 'type'. It also includes a 'GebelikteRiskFactorleri' element. The XML is generated using schema builder version 3.1.0, as indicated in the documentation section.

```
--><xs:annotation>
<xs:documentation>Generated using schema builder version 3.1.0. Stylesheet
s:
HMD was rendered into XML using software provided to HL7 by Beeler Consulting LL
C.
HMD to MIF Transform: $Id: RoseTreeHmdToMIFStaticModel.xsl,v 1.14 2007/03/20 02
:48:48 wbeeler Exp $
Base transform: $Id: ConvertBase.xsl,v 1.4 2007/03/20 02:48:48 wbeeler Exp $
Package Id Conversion: $Id: TransformPackageIds.xsl,v 1.6 2007/03/20 02:48:49
wbeeler Exp $
HTML To MIF markup: $Id: HtmlToMIFMarkup.xsl,v 1.4 2007/03/20 02:48:49 wbeeler
Exp $
Flat to Serialization Transform: $Id: MIFStaticModelFlatToSerialization.xsl,v 1
.5 2007/03/20 02:48:50 wbeeler Exp $
Fix Names Transform: $Id: FixMifNames.xsl,v 1.8 2007/03/20 02:48:49 wbeeler Exp
$
Base transform: $Id: ConvertBase.xsl,v 1.4 2007/03/20 02:48:48 wbeeler Exp $
Package Id Conversion: $Id: TransformPackageIds.xsl,v 1.6 2007/03/20 02:48:49
wbeeler Exp $
StaticMifToXsd.xsl version 2.0</xs:documentation>
</xs:annotation>
<xs:include schemaLocation="../coreschemas/infrastructureRoot.xsd"/>
<xs:complexType name="UUDD_HD123456.Agirlik">
<xs:sequence>
<xs:group ref="InfrastructureRootElements"/>
</xs:sequence>
<xs:attributeGroup ref="InfrastructureRootAttributes"/>
<xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>
<xs:attribute name="classCode" type="ActClassObservation" use="required"/>
<xs:attribute name="moodCode" type="ActMood" use="required"/>
</xs:complexType>
<xs:complexType name="UUDD_HD123456.Component">
<xs:sequence>
<xs:group ref="InfrastructureRootElements"/>
<xs:element name="agirlik" type="UUDD_HD123456.Agirlik" minOccurs="1" m
axOccurs="1"/>
</xs:sequence>
<xs:attributeGroup ref="InfrastructureRootAttributes"/>
<xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>
<xs:attribute name="typeCode" type="ActRelationshipHasComponent" use="requ
ired"/>
</xs:complexType>
<xs:complexType name="UUDD_HD123456.Component2">
<xs:sequence>
<xs:group ref="InfrastructureRootElements"/>
<xs:element name="gebelikteRiskFactorleri" type="UUDD_HD123456.Gebelikt
eRiskFactorleri"
nillable="true"
```

Şekil 50 v3Generator sonucu

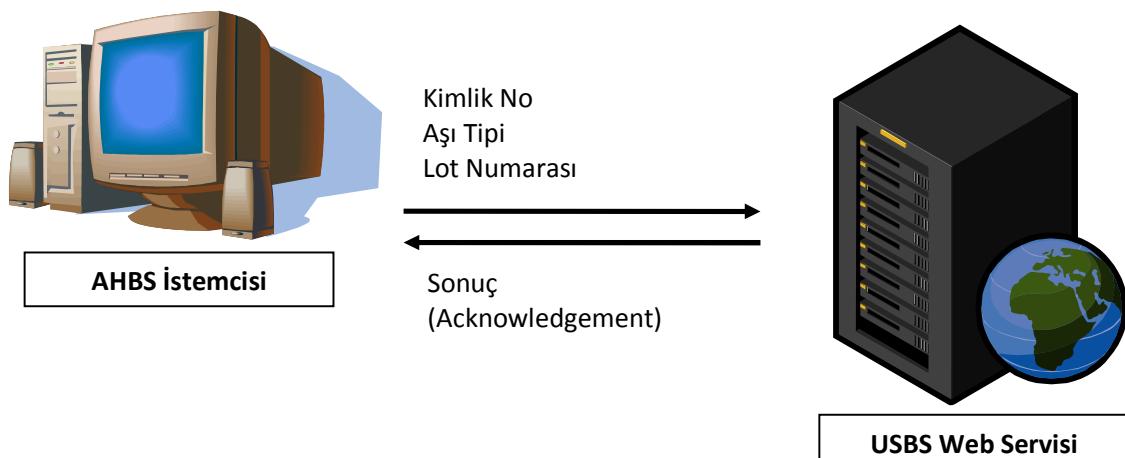
3 Web Servisler ve Web Servisi Sunucusu/İstemicisi Uygulaması

Web Servisleri, genel bir tanımla: "Web üzerinden SOAP (Simple Object Access Protokolü) ile erişilebilinen, arayüzü (diğer bir deyişle, ne mesaj alıp ne mesaj döndüğü, hangi operasyonları ve hangi bindingleri desteklediği, hangi makinede bulunduğu) WSDL (Web Service Description Language) ile anlatılan uygulamalara" denir. Bu bağlamda Web'de hasta Elektronik Sağlık Kaydı sorgulanmasına izin veren bir uygulamadan basit bir havadurumu bilgisi dönen bir uygulamaya kadar her şey bir Web Servisi olarak düşünülebilir. Gerçekleştirilecek olan senaryo adım adım detaylı olarak anlatılmıştır. Okunabilirliği artırmak için, gerekli/detaylı bilgiler ilgili adımda belirtilmiştir.

3.1 Kullanılan Senaryo

Bu senaryoda (Şekil 51) AHBS, Aile Hekimliğine gelen her bir hastaya yapılan aşayı USBS'ye bildirmesi düşünülmüştür. AHBS'nin göndereceği Veri Seti şu şekildedir:

- Hasta TC Kimlik Numarası
- Aşı Tipi
- Lot Numarası



Şekil 51 Uygulanacak Senaryo

Bu bağlamda USBS'de bir Aşı Bilgisi kabul Web Servisi bulunacak ve AHBS'de bu Web Servisini uyandıracak olan istemci bulunacaktır. AHBS istemcisi aşı bilgisini kendinde bulunan bir veritabanından çekip USBS'ye gönderecektir. USBS bu mesajı aldığı zaman önce Kimlik bilgisini MERNIS ile kontrol edecek, başarılı ise kendi "Aşı Veri Tabanına" kaydedecek ve daha sonra da bir acknowledgement dönecektir.

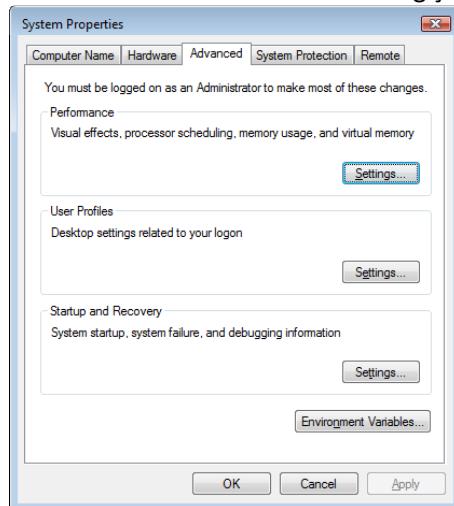
3.2 Web Servisi Geliştirme Ortamının Hazırlanması ve Kurulması

Bu dokümanda Tomcat 5.0.28 ve Axis 1.2.1 kullanılmıştır. Tomcat bir Web Server uygulamasıdır ve HTTP üzerinden gelen isteklere cevap verme yetisindedir. Ama SOAP mesajlarını işleme özelliği yoktur. Bunun için Axis denen SOAP işleyicisinin Tomcat'e eklenmesi gerekmektedir. Axis ayrıca kişilerin Web Servis geliştirmesinde kolaylık sağlamak amacıyla yardımcı programlar sağlamaktadır. Bu programlar sayesinde uygulamaclar SOAP mesajlarının detayları ile uğraşmadan "Uygulama Seviyesinde" kalarak kendi Web

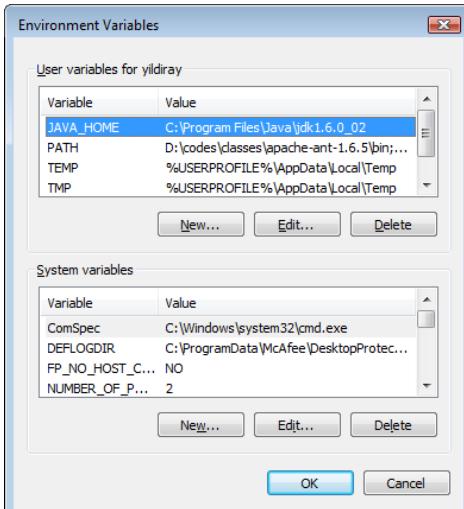
T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

Servislerini yada istemcilerini geliştirebilmektedir. Web Servisinin ve istemcisinin geliştirilmesinde kullanılacak ortamın (web server ve SOAP işleyicisi: Tomcat ve Axis) kurulması şu şekildedir:

1. Öncelikle bir command prompt açarak java ve javac komutlarının çalışıp çalışmadığını kontrol edin. Eğer çalışmıyor ise PATH ortam değişkenine kurulumunu yaptığıınız Java'nın path'ini ekleyin. Bunun için "Bilgisayarım"ı sağ tıklayın, buradan "Özellikler" seçin. Çıkan pencerede "Gelişmiş" sekmesine gidin. Bu sekmenin altında "Ortam Değişkenleri" ni göreceksiniz (Şekil 52).



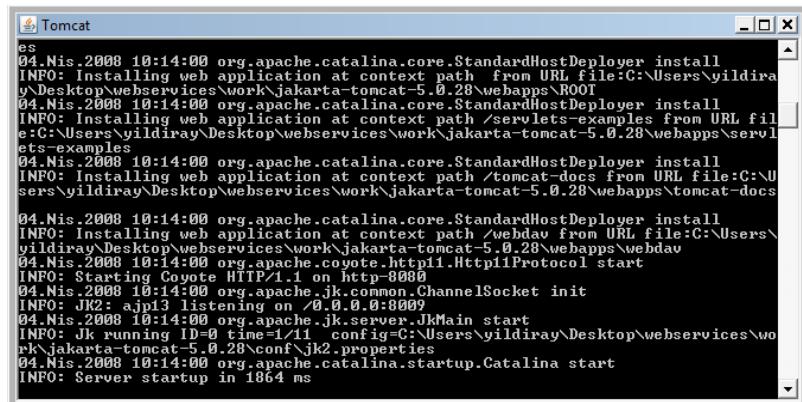
Şekil 52 Ortam Değişkenleri



Şekil 53 JAVA_HOME ortam değişkeni

2. Tam buradayken JAVA_HOME ortam değişkeninin de olup olmadığını kontrol edin. Eğer yoksa bu değişkeninde tanımlanması gerekmektedir (Şekil 53).
3. "webservices" isimli bir dizin açın.

4. “webservices” dizininin içine gidin. Doküman ilerlerken yapılacak işlemleri kolaylaştırma amaçlı scriptler sunulacaktır. Bu scriptler çalışırken “work” diye bir dizini referans almaktadır. Bu sebeple o anki “webservices” dizininde “work” diye bir dizin daha açın.
5. jakarta-tomcat-5.0.28.zip dosyasını “work” dizininin altına açın.
6. axis-bin-1.2.1.zip’i “work” dizininin altına açın.
7. “work\axis-1.2.1\webapps\axis” dizinini “work\jakarta-tomcat-5.0.28\webapps” altına kopyalayın. Böylece elinizde “work\jakarta-tomcat-5.0.28\webapps\axis” diye bir dizin bulunsun.
8. Tomcat ve Axis kurulumu tamamlanmıştır. Daha sonra bu kurulumu denemek için önce Tomcat’ı çalıştırın. Bunu “work\jakarta-tomcat-5.0.28\bin\startup.bat” scriptini kullanarak yapın (Şekil 54).



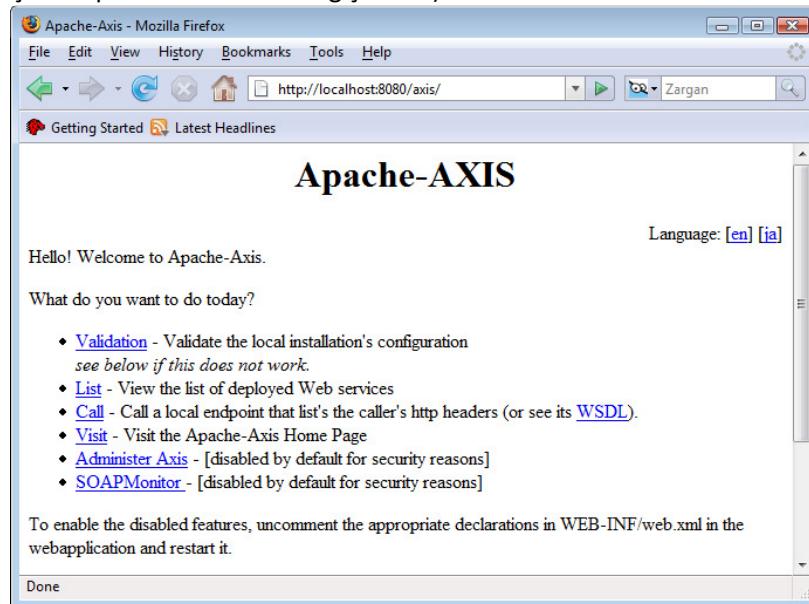
```

Tomcat
es
04.Nis.2008 10:14:00 org.apache.catalina.core.StandardHostDeployer install
INFO: Installing web application at context path from URL file:C:\Users\yildiray\Desktop\webservices\work\jakarta-tomcat-5.0.28\webapps\ROOT
04.Nis.2008 10:14:00 org.apache.catalina.core.StandardHostDeployer install
INFO: Installing web application at context path /servlets-examples from URL file:C:\Users\yildiray\Desktop\webservices\work\jakarta-tomcat-5.0.28\webapps\servlets-examples
04.Nis.2008 10:14:00 org.apache.catalina.core.StandardHostDeployer install
INFO: Installing web application at context path /tomcat-docs from URL file:C:\Users\yildiray\Desktop\webservices\work\jakarta-tomcat-5.0.28\webapps\tomcat-docs
04.Nis.2008 10:14:00 org.apache.catalina.core.StandardHostDeployer install
INFO: Installing web application at context path /webdav from URL file:C:\Users\yildiray\Desktop\webservices\work\jakarta-tomcat-5.0.28\webapps\webdav
04.Nis.2008 10:14:00 org.apache.coyote.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
04.Nis.2008 10:14:00 org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8009
04.Nis.2008 10:14:00 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=1/11 config=C:\Users\yildiray\Desktop\webservices\work\jakarta-tomcat-5.0.28\conf\jk2.properties
04.Nis.2008 10:14:00 org.apache.catalina.startup.Catalina start
INFO: Server startup in 1864 ms

```

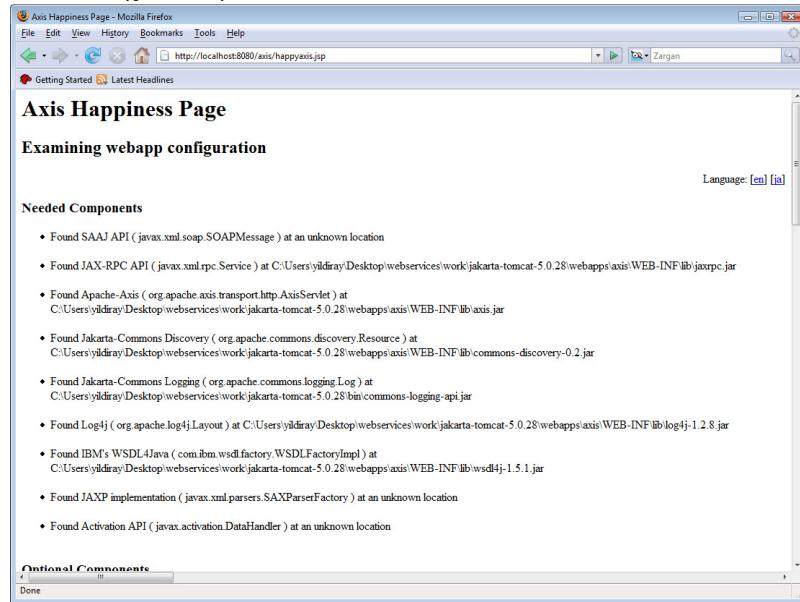
Şekil 54 Tomcat çalışma ekranı

9. Daha sonra bir browser açarak <http://localhost:8080/axis> adresine bağlanın (Şekil 55). (Önceden bir Tomcat kurulumu var ise, yeni kurulan Tomcat’ın çıkışmaması için “work\jakarta-tomcat-5.0.28\conf\server.xml” dosyasının değiştirilmesi gerekmektedir”. Buradaki 8080, 8443, 8005 değerlerini başka bir port numarası ile değiştirebilirsiniz.)



Şekil 55 Axis giriş sayfası

10. Bu yeni çıkan pencerede “Validation” linkine tıklayıp, tüm zorunlu componentlerin olup olmadığını kontrol edin (Şekil 56).



Şekil 56 Axis validasyon sayfası

11. Kurulumunuz tamamlanmıştır.

3.3 USBS tarafından çalışacak Web Servisin geliştirilmesi

Öncelikle şunu belirtmekte fayda var: “webservices” dizininde bulunan scriptler, Axis ile Web servis geliştirmeyi kolaylaştırmak amacıyla yazılmıştır. Bir Web servisi uygulaması önce yapılacak Web Servisinin WSDL dokümanını yazmak ile başlar. Bu WSDL dokümanı uygulamalar arasında kontrat niteliğinde olup sadece arayüzü tanımlar. Dikkat edilmesi gereken nokta bu WSDL’ın içindeki “endpoint” kısmı gerçek/ulaşılabılır bir URL belirtmemektedir. O an henüz Web Servis uygulanmadığı için öylesine bir değer girmedektedir. İleriki zamanlarda, senaryomuzdan devam edeceğimizde, USBS, Web Servisini geliştirmeyi bitirdiğinde, gerçek WSDL dokümanını üretip, AHBS’ye gönderecektir. Görüldüğü üzere, WSDL dokümanlarının iki kullanım şekli bulunmaktadır: (1) İlk aşamada Web Servisinin kendini geliştirmek, (2) Geliştirilmesi bitmiş bir Web Servisinin istemcisini geliştirmek. Bu bölümde Web Servisinin kendisinin, yani USBS tarafının, geliştirilmesi anlatılmaktadır. Bu Web Servisin geliştirilmesi için aşağıdaki adımları takip ediniz:

1. “webservices” dizinin içinei Example.wsdl dosyasını düzenleyiniz. Bunun için WSDL dosyasının içeriği aşağıdaki şekilde olması gerekmektedir:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
  targetNamespace="example.srdc.com.tr"
  xmlns:apachesoap="http://xml.apache.org/xml-soap"
  xmlns:impl="example.srdc.com.tr"
  xmlns:intf="example.srdc.com.tr"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
```

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
<wsdl:message name="vaccine">
  <wsdl:part name="kimliknumarası" type="xsd:string"/>
  <wsdl:part name="asitipi" type="xsd:string"/>
  <wsdl:part name="lotnumarası" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="acknowledgement">
  <wsdl:part name="ack" type="xsd:string"/>
</wsdl:message>
<wsdl:portType name="Example">
  <wsdl:operation name="asiKabul" parameterOrder="kimliknumarası asitipi lotnumarası">
    <wsdl:input message="impl:vaccine" name="vaccine"/>
    <wsdl:output message="impl:acknowledgement" name="acknowledgement"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ExamplePortSoapBinding" type="impl:Example">
  <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="asiKabul">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="vaccine">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <!-- Example.asiKabul -->
    </wsdl:input>
    <wsdl:output name="acknowledgement">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <!-- Example.acknowledgement -->
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="ExampleService">
  <wsdl:port binding="impl:ExamplePortSoapBinding" name="ExamplePort">
    <wsdlsoap:address location="http://localhost"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

2. Öncelikle “webservices” dizinindeki “wsdl2java.bat” dosyasını düzenlemek gerekmektedir. Axis, WSDL2Java isimli bir tool sunmaktadır. Bu tool, kullanıcıların bir Web servisinin WSDL dokümanını kullanarak gerek istemci kaynak kodunu gerekse Web servisinin kendisinin kaynak kodunun (server tarafının) üretilmesini sağlamaktadır. İlk aşamada server tarafının kaynak kodlarını üretmemiz gerekmektedir. Bunun için WSDL2Java’ya “--server-side” opsyonu verilmelidir. Bu sebeple wsdl2java.bat dosyasının içeriği Şekil 57 deki gibi olmalıdır:

```

set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\
set CP=.;%LIBDIR%axis.jar;%LIBDIR%axis-ant.jar;%LIBDIR%commons-discovery-0.2.jar;%LIBDIR%commons-logging-1.0.4.jar;
;%LIBDIR%jaxrpc.jar;%LIBDIR%log4j-1.2.8.jar;%LIBDIR%saaj.jar;%LIBDIR%wsdl4j-1.5.1.jar
REM set WSDL=http://localhost:8080/axis/services/ExamplePort?wsdl
set WSDL=Example.wsdl

java -cp %CP% org.apache.axis.WSDL2Java --server-side %WSDL%
~
~
~
```

Şekil 57 wsdl2java.bat

3. Daha sonra wsdl2java.bat dosyasını çift tıklayarak çalıştırın. Çalışma bittiğinde “webservices” dizini altında “tr” isimli bir dizin oluşacaktır. Bu dizin, üretilen Java sınıflarını içermektedir. En alt seviyeye kadar indiğinizde beş tane Java sınıfı ve ek olarak iki tane WSDL dosyası görülmektedir. WSDL2Java tool’u aslında, özellikle istemci Java sınıflarını üretmek içindir. Ne olursa olsun hep istemci kaynak kodlarını üretir. Ek olarak “--server-side” opsyonu verildiğinde, Web servis kaynak kodlarını da üretir. Bu yüzden buradaki istemci’ye ait olan kaynak kodlarını silmek gerekmektedir. Bu sınıflar şunlardır: ExamplePortSoapBindingStub, ExampleService ve ExampleServiceLocator. Diğer kalan iki sınıf Web Servisinin kendisine aittir.
4. WSDL2Java tool’u kaynak kodundaki methodlarının içeriği boş olan Java sınıfları üretir. Bu tür sınıflara template adı verilmektedir. Uygulamacılar bu methodların içerisinde kendilerine göre doldurmaları gerekmektedir. Bu işleme “overriding” denir. Mesela bizim örneğimizde ExamplePortSoapBindingImpl sınıfının içerisinde aşağıdaki şekilde içeriği boş olan bir method bulunmaktadır:

```

public java.lang.String asiKabul(java.lang.String kimliknumarası,
        java.lang.String asitipi,
        java.lang.String lotnumarası)
    throws java.rmi.RemoteException {
    return null;
}
```

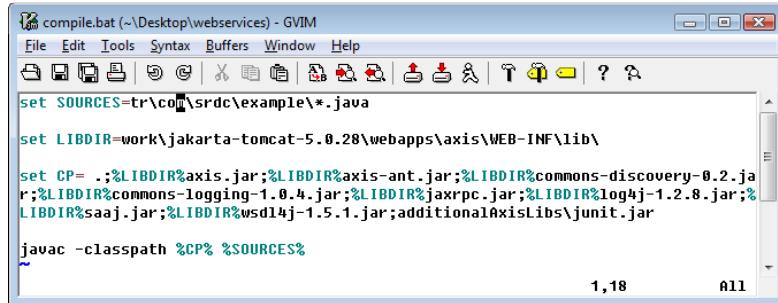
USBS Web Servisini geliştirecek olan uygulamacı bunun içini dolduracaktır. Önce “kimliknumarası” değişkenini kullanarak, MERNİS’i sorgulayacak daha sonra MERNİS’ten “geçerlidir” yanıtını alırsa bu aşı bilgisini kendi AŞI veri tabanına koyacaktır. Mesela bu overriding sonucunda yukarıdaki method aşağıdaki gibi olabilir (NOT: buradaki diğer kullanılan MERNIS yada ASIDB sınıfları örnek olup gerçek sınıflar değildir. Bu aşamada sadece “return null;” satırını “return kimliknumarası+”-CA”; olarak değiştirin)

```

public java.lang.String asiKabul(java.lang.String kimliknumarası,
        java.lang.String asitipi,
        java.lang.String lotnumarası)
    throws java.rmi.RemoteException {
    if(MERNİS.kimlikKontrol() == true) {
        ASIDB.insert("INSERT INTO ASIDB VALUES("+kimliknumarası+","+asitipi+","+lotnumarası+ ")");
        return "CA";
    } else
        return "CE";
```

}

- Daha sonra bu Web servisinin kaynak kodlarını derleyin. Bunun için “webservices” dizinindeki compile.bat scriptini kullanabilirsiniz. Compile script’inin içeriği Şekil 58 deki gibi olmalıdır:



```

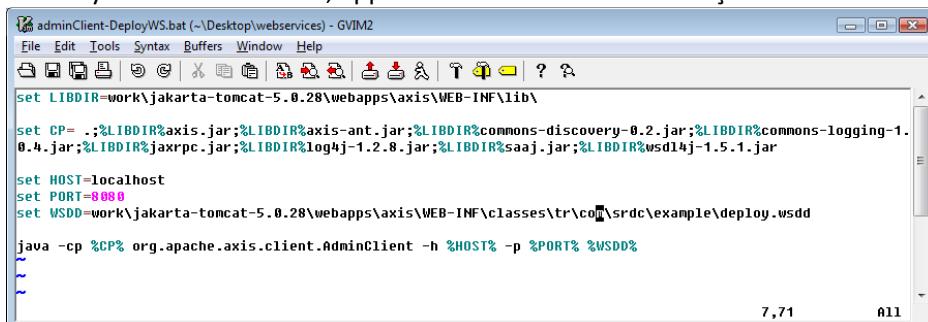
set SOURCES=tr\com\srdc\example\*.java
set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\
set CP=.;%LIBDIR%axis.jar;%LIBDIR%axis-ant.jar;%LIBDIR%commons-discovery-0.2.jar;%LIBDIR%commons-logging-1.0.4.jar;%LIBDIR%jaxrpc.jar;%LIBDIR%log4j-1.2.8.jar;%LIBDIR%saaj.jar;%LIBDIR%wsdl4j-1.5.1.jar;additionalAxisLibs\junit.jar
javac -classpath %CP% %SOURCES%

```

Şekil 58 compile.bat

- Derlenen bu sınıfların bir Web servis olarak açılabilmesi için Tomcat kurulumuna tanıtılması gerekmektedir. Bu işleme deployment denmektedir:

- Öncelikle derlenmiş sınıfların Tomcat’ın ulaşabileceği bir dizine konulması gerekmektedir. Bunun için “webservices\tr” dizinini kesip, “work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\classes” altına kopyalayın ve Tomcat’ı tekrar açıp-kapatın.
- Şu an Tomcat bu sınıflara ulaşabilir hale gelmiştir ama bu sınıfların bir Web Servise ait olduğunu bilmemektedir. Bunun için “webservices” dizinindeki “adminClient-DeployWS.bat” scripti kullanılabilir. Bu script’İN yaptığı iş WSDL2Java’NIN ürettiği deploy.wsdd’yi Tomcat’E göndermektir. WSDD (Web Service Deployment Descriptor) dosyaları Web Servisleri, application serverlara tanıtmak için kullanılır.



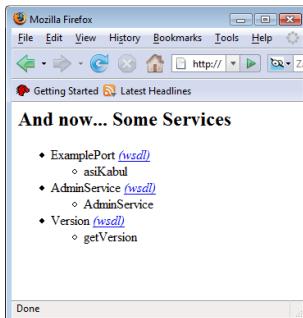
```

set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\
set CP=.;%LIBDIR%axis.jar;%LIBDIR%axis-ant.jar;%LIBDIR%commons-discovery-0.2.jar;%LIBDIR%commons-logging-1.0.4.jar;%LIBDIR%jaxrpc.jar;%LIBDIR%log4j-1.2.8.jar;%LIBDIR%saaj.jar;%LIBDIR%wsdl4j-1.5.1.jar
set HOST=localhost
set PORT=8080
set WSDD=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\classes\tr\com\srdc\example\deploy.wsdd
java -cp %CP% org.apache.axis.client.AdminClient -h %HOST% -p %PORT% %WSDD%

```

Şekil 59 adminClient-DeployWS.bat

- Deployment tamamlandıktan sonra USB’sının Web Servisi uygulaması bitmiştir. Emin olmak için <http://localhost:8080/axis> anasayfasındaki “List” linkinden bu kontrol edilebilir (Şekil 60).



Şekil 60 Uygulanmış Web Servis

3.4 AHBS Tarafında Çalışacak İstemcinin Geliştirilmesi

USBS Web Servisin kendisinin uygulamasını bitirdikten sonra AHBS bu Web Servisi uyandıracak olan istemciyi geliştirmeye başlar. Bunun için uygulanmış Web Servisine ait WSDL dokümanının USBS tarafından AHBS'ye iletilmesi gerekmektedir. AHBS izlediği adımlar yani bir istemci (client) geliştirmek için gereken adımlar aşağıdaki şekildedir:

1. Axis'in WSDL2Java tool'unu bu sefer aşağıdaki şekilde “--server-side” opsyonu vermeden çalıştırın (Şekil 61). (NOT1: Buradaki başında REM ibaresi olan satırlar deactivate olmuş satırlardır) (NOT2: USBS'nin geliştirdiği Web Servisin asıl WSDL URL'i şekildeki script'te de görüldüğü üzere <http://localhost:8080/axis/services/ExamplePort?wsdl> dir)

The screenshot shows a Windows command prompt window titled 'wsdl2java.bat (~\Desktop\webservices) - GVIM'. The window contains the following command line:


```
set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\
set CP= .;%LIBDIR%axis.jar;%LIBDIR%axis-ant.jar;%LIBDIR%commons-discovery-0.2.jar;%LIBDIR%commons-logging-1.0.4.jar;%LIBDIR%jaxrpc.jar;%LIBDIR%log4j-1.2.8.jar;%LIBDIR%saaj.jar;%LIBDIR%wsdl4j-1.5.1.jar
set WSDL=http://localhost:8080/axis/services/ExamplePort?wsdl
REM set WSDL=Example.wsdl
REM java -cp %CP% org.apache.axis.wsdl.WSDL2Java --server-side %WSDL%
java -cp %CP% org.apache.axis.wsdl.WSDL2Java -t %WSDL%
```

Şekil 61 wsdl2java.bat

2. “webservices” dizininde yeni yaratılmış “tr” klasörünü göreceksiniz. Bu klasör istemcinin kullanacağı template kaynak kodlarını içermektedir. Aynı server kaynak kodunda olduğu gibi burada üretilen methodların içeriği boştur. Uygulamacı bunların içini override etmesi yani doldurması gerekmektedir. Senaryo örneğinden devam edecek olursak, AHBS kendi veri tabanına bağlanıp, hasta kimlik numarasını, aşı tipini ve lot numarasını çekip Web servisine gönderecektir.
3. “webservices\tr” dizininin en alt seviyesine gidildiğinde burada beş tane istemci kaynak kodu görülmektedir. Burada üzerinde değişiklik yapılacak ana sınıf “ExampleServiceTestCase” dir. Bu kodun içinde görüldüğü üzere aşağıdaki method bulunmaktadır:

```
public void test1ExamplePortAsiKabul() throws Exception {
    tr.com.srdc.example.ExamplePortSoapBindingStub binding;
    try {
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
binding = (tr.com.srdc.example.ExamplePortSoapBindingStub)
    new tr.com.srdc.example.ExampleServiceLocator().getExamplePort();
}
catch (javax.xml.rpc.ServiceException jre) {
    if(jre.getLinkedCause() != null)
        jre.getLinkedCause().printStackTrace();
    throw new junit.framework.AssertionFailedError("JAX-RPC ServiceException caught: " + jre);
}
assertNotNull("binding is null", binding);
binding.setTimeout(60000);
java.lang.String value = null;
value = binding.asiKabul(new java.lang.String(), new java.lang.String(), new java.lang.String());
}
```

Bu method boş inputlar göndermektedir. İstemciyi uygulayacak kişinin bu inputlara girilecek olan değerleri dolduracak olan kodu kendi yazması gerekmektedir. Örnegimizde istemci bu değerleri kendi veri tabanından çekecektir. Örnek bir uygulama aşağıdaki gibi olacaktır:

```
public void test1ExamplePortAsiKabul() throws Exception {
    tr.com.srdc.example.ExamplePortSoapBindingStub binding;
    try {
        binding = (tr.com.srdc.example.ExamplePortSoapBindingStub)
            new tr.com.srdc.example.ExampleServiceLocator().getExamplePort();
    }
    catch (javax.xml.rpc.ServiceException jre) {
        if(jre.getLinkedCause() != null)
            jre.getLinkedCause().printStackTrace();
        throw new junit.framework.AssertionFailedError("JAX-RPC ServiceException caught: " + jre);
    }
    assertNotNull("binding is null", binding);
    binding.setTimeout(60000);

    String kimlikno=HASTADB.retrieve("SELECT ID FROM HASTATABLOSU");
    String asitipi=HASTADB.retrieve("SELECT ASITIPI FROM HASTATABLOSU WHERE ID="+kimlikno);
    String lotnumarası=HASTADB.retrieve("SELECT LOTNUMARASI FROM HASTATABLOSU WHERE ASITIPI='"+asitipi');

    java.lang.String value = null;
    value = binding.asiKabul(kimlikno, asitipi, lotnumarası);
    System.out.println(" $$$ SONUC: "+value);
}
```

(NOT: Buradaki HASTADB sınıfı gerçek olmayan sadece örnek vermek için kullanılmış bir sınıfır)

(NOT2: Şimdilik bu adım için aşağıdaki satırları:

```
String kimlikno=HASTADB.retrieve("SELECT ID FROM HASTATABLOSU");
String asitipi=HASTADB.retrieve("SELECT ASITIPI FROM HASTATABLOSU WHERE ID="+kimlikno);
String lotnumarası=HASTADB.retrieve("SELECT LOTNUMARASI FROM HASTATABLOSU WHERE ASITIPI='"+asitipi');
```

bu satırlar ile değiştirin:

```
String kimlikno="12345678910";
String asitipi="VEREM";
```

String lotnumarası="1";

4. Ek olarak WSDL2Java'nın ürettiği kaynak kodlarda Java'nın ilk çalışma esnasında çalıştıracağı "main" fonksiyonu bulunmamaktadır. Aşağıdaki "main" fonksyonunu ExampleServiceTestCase sınıfına ekleyin:

```
public static void main(String argv[]) {  
    try {  
        ExampleServiceTestCase estc=new ExampleServiceTestCase("");  
        estc.test1ExamplePortAsiKabul();  
    } catch(Exception ex) {  
    }  
}
```

5. Daha sonra bu istemci kodunu compile.bat scriptini kullanarak derleyin.
6. Son olarak invoke.bat scriptini kullanarak istemciyi çalıştırıp Web Servisini uyandırabilirsiniz. invoke.bat scripti Şekil 62'de sunulmaktadır.

```
C:\Select Administrator: C:\Windows\system32\cmd.exe  
C:\Users\yildiray\Desktop\webservices>invoke.bat  
C:\Users\yildiray\Desktop\webservices>set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\  
C:\Users\yildiray\Desktop\webservices>set CP=.;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\axis.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\axis-ant.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\commons-discovery-0.2.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\commons-logging-1.0.4.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\jaxrpc.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\log4j-1.2.8.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\saaj.jar;work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\wsdl4j-1.5.1.jar;additionalAxislibs;junit.jar tr.com.srdc.example.ExampleServiceTestCase  
$55 SONUC: 12345678910-C4  
C:\Users\yildiray\Desktop\webservices>
```

Şekil 62 Web Servisin çağrılması

3.5 Gidip gelen SOAP Mesajlarının TCPMonitor yardımcıyla görüntülenmesi

Axis ayrıca TCPMonitor isimli TCP/IP üzerinden gidip/gelen mesajları görüntüleyebileceğiniz bir tool sağlamaktadır. Bu tool bir aracı olarak görev yapmakta olup, SOAP mesajlarını görebilmek için istemcinin göndereceği SOAP mesajını Web Servis yerine TCPMonitor'e göndermek gerekmektedir. TCPMonitor'ün de çalışmaya başlarken kendine gelen mesajları Web Servisine yönlendirmesi için konfigüre edilmesi gerekmektedir.

3.5.1 TCPMonitor'un başlatılması

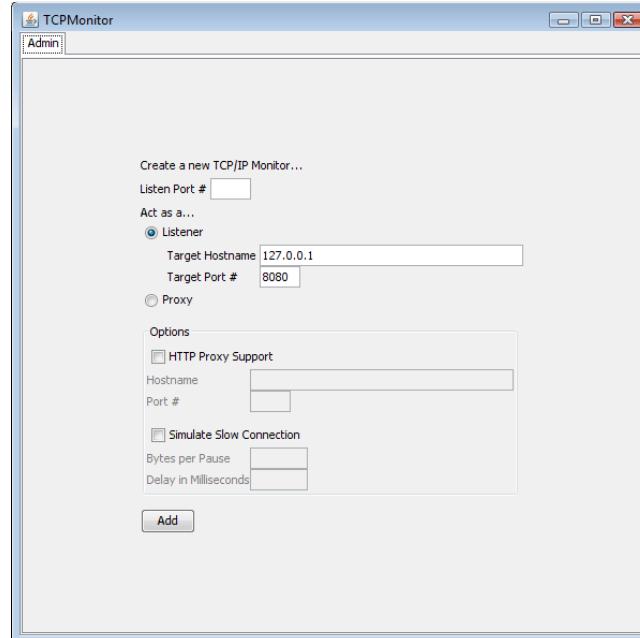
```
set LIBDIR=work\jakarta-tomcat-5.0.28\webapps\axis\WEB-INF\lib\
```

```
set CP=.;%LIBDIR%axis.jar;%LIBDIR%axis-ant.jar;%LIBDIR%commons-discovery-0.2.jar;%LIBDIR%commons-logging-1.0.4.jar;%LIBDIR%jaxrpc.jar;%LIBDIR%log4j-1.2.8.jar;%LIBDIR%saaj.jar;%LIBDIR%wsdl4j-1.5.1.jar
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

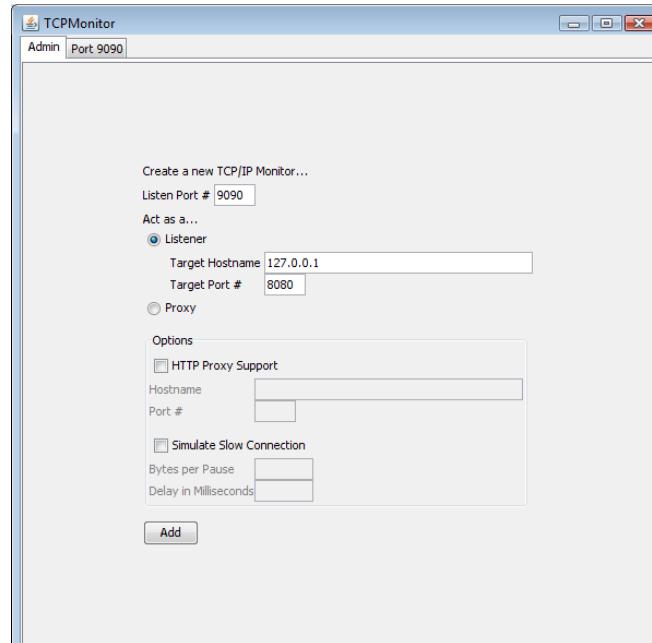
`java -cp %CP% org.apache.axis.utils.tcpmon`

Yukarıdaki script ile TCPMonitor çalıştırılır. Şekil 63 deki ekran görünecektir.



Şekil 63 TCPMonitor Açılışı

Burada önce TCPMonitor'un 9090 portunu dinleyip gelen istekleri Web Servislerin çalıştığı 8080 portuna yönlendirmesi gerekmektedir. Bunun için aşağıdaki gibi değerlerin girilip "Add" tuşuna basılmalıdır.



Şekil 64 TCPMonitor konfigürasyonu

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

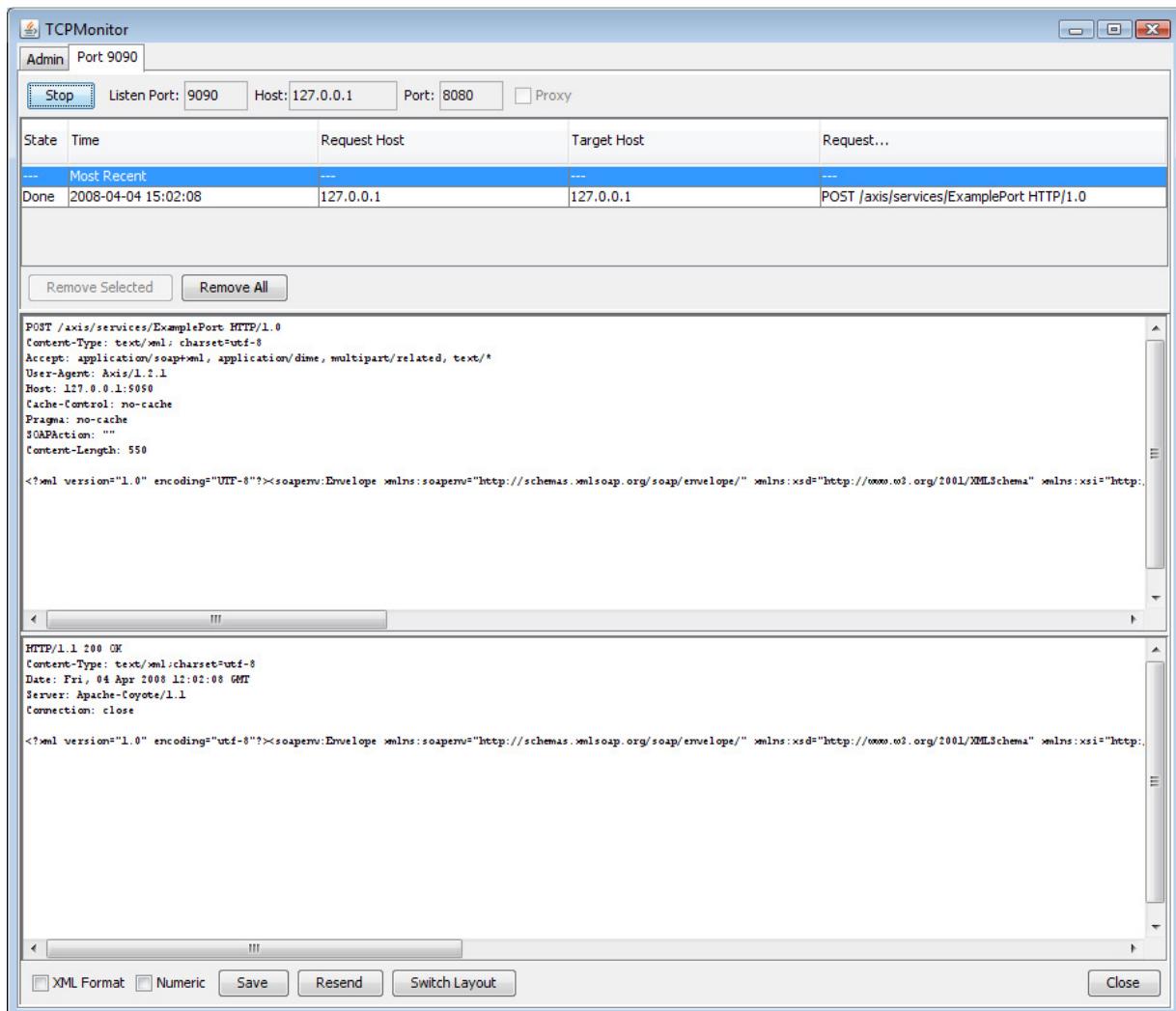
Daha sonra Web Servis istemcimize isteklerin 8080 portuna değil SOAP mesajlarını 9090 portuna göndermesi gerekiği söylenmelidir. Bunun için ExampleServiceLocator sınıfı açılıp içerisindeki şu satır:

```
private java.lang.String ExamplePort_address = "http://localhost:8080/axis/services/ExamplePort";
```

aşağıdaki satır ile değiştirilmelidir.

```
private java.lang.String ExamplePort_address = "http://localhost:9090/axis/services/ExamplePort";
```

Daha sonra istemci kodumuzu tekrar compile.bat ile derleyip, invoke.bat ile çalıştığımızda gidip gelen SOAP mesajları TCPMonitor'de Şekil 65 deki gibi görülecektir.



Şekil 65 TCPMonitor Çalışması

4 Sağlık-NET Web Servisleri Örnek Java İstemci Uygulaması

4.1 Giriş

Dokümanın bu bölümünde Sağlık-NET Web Servislerini çağrırmak için nasıl bir örnek istemci yazılabileceği anlatılmaktadır. Bu geliştirilecek istemcinin 2 katmandan olduğu düşünülebilir: HL7 Katmanı Uygulaması ve Web Servisi Katmanı Uygulaması. Web Servisi Katmanı Uygulaması axis yardımıyla web service wsdl dosyalarından java kodlarının üretilmesiyle oluşmaktadır. Bu uygulama web servisi çağrırmak için transport seviyesinde gerekli altyapıyı sağlarken tek başına yeterli değildir. Sağlık-NET Web servisleri HL7 Mesajlarını kullandığı için üstüne HL7 Katmanı Uygulaması gerekmektedir. Bu seviye, gidecek HL7 mesajlarının oluşturulması ve mesajlara güvenlik eklenmesinden sorumludur.

HL7 Katmanı Uygulaması, gönderilmesi gereken xml mesaj dosyasını, hangi tür USBS servisinin çağrılaçğını, birinci ya da ikinci operasyondan hangisinin çağrılaçğını ve mesaj için bir uuid değerini argüman olarak alır. Genel işleyişi şu şekildedir.

Birinci operasyon çağrılıyorsa:

*Xml mesajını dosyadan okur, mesajı java nesnelerine yükler, mesajın **uuid değerini** argümanda verilen **uuid değeri ile** değiştirir, servisi bu nesne ile çağrıır ve sonucu döner. Eğer sonuç CA dönmüşse (başarılı bir çağrıma olmuşa), **uuid değerini servisin ikinci operasyonunu çağrırmak için kaydeder.***

İkinci operasyon çağrılıyorsa:

*Xml mesajını dosyadan okur, java nesnelerine yükler, durumu kontrol edilmek istenen mesajın **uuid değerini** argümanda verilen **uuid değeri ile** günceller, servisi bu nesne ile çağrıır ve sonucu döner.*

İstemciyi yazmak için java dili kullanılmıştır. İstemcinin Sağlık-NET Web Servislerini çağrıabilecek duruma gelmesi aşağıdaki 4 ana adımdan oluşmaktadır:

- Ortamın hazırlanması

Bu adımda gerekli dizin yapısının oluşturulması ve uygun altyapının kurulması gerçekleşmektedir. Bu adım tamamlandığında aşağıdaki dizin yapısının olması beklenmektedir.

```
[USBS_Istemci]
  |---> conf
    |--->client.axis2.xml
    |--->client_conf.properties
  |---> resources
    |--->instances
    |--->wsdl
      |--->gerekli wsdl dosyaları
  |---> src
    |--->client
    |--->v3
  |---> lib
```

```
|--->axis  
|--->rampart  
|---> build.xml
```

Bu yapıda **[USBS_Istemci]/conf** dizini altında sistemin çalışması için gerekli ayarlama dosyaları bulunmaktadır. **client.axis2.xml** dosyası axis2 kütüphaneleri tarafından otomatik kullanılırken, **client_conf.properties** dosyası da çağrılabilecek web servisinin adresi, portu ve kullanıcı şifre bilgileri yer almaktadır.

[USBS_Istemci]/resources dizini altında ise sistem tarafından kullanılan girdi dosyaları bulunmaktadır. Bunlardan ilki **[USBS_Istemci]/resources/instances** dizininde bulunan web servislerini çağrıırken servislere girdi olarak verilen örnek mesajlardır. Farklı mesajlar gönderilmek istendiğinde buradaki mesajlar değiştirilebilir ya da yeni mesajlar eklenip build.xml dosyasında verilen argümanlar değiştirilebilir. **[USBS_Istemci]/resources/wsdl** dizinin altında ise WSDL2Java kod üreticisinin Web Servisleri Katmanı Uygulaması kodlarını oluştururken kullanılacak wsdl dosyaları bulunmalıdır.

[USBS_Istemci]/src dizinin altında hem HL7 Katmanı Uygulaması (**[USBS_Istemci]/src/client**) hem de Web Servisleri Katmanı Uygulaması (**[USBS_Istemci]/src/v3**) kodları bulunmaktadır. Web Servisleri Katmanı Uygulaması kodları WSDL2Java çalıştırıldığında otomatik olarak burada üretilir. HL7 Katmanı Uygulaması ise kullanıcı tarafından yazılıp **[USBS_Istemci]/src/client** klasörünün altına koyulmalıdır.

Son olarak **[USBS_Istemci]/lib** dizini ise sistem tarafından kullanılan axis ve rampart kütüphanelerini içermektedir.

- Kodların üretilmesi (WSDL2Java)

Bu adımda **[USBS_Istemci]/resources/wsdl** dizininin altındaki wsdl dosyasının belirttiği web servislerine uygun WebService Katmanı Uygulamasının kodları otomatik olarak axis2 tarafından çıkartılmaktadır. Ancak bu süreçte bazı hatalarla karşılaşlığımızdan dolayı (bkzn. 4.3.2) kodlarda bir miktar değişiklik gerekmektedir.

- Güvenlik Eklenmesi

Sağlık-Net Web Servislerinin çağrılmaması için giden mesajın Header kısmına güvenlik bilgilerinin eklenmesi gerekmektedir. Bu adımda HL7 Katmanı Uygulamasının kullanıcı adı ve şifre bilgilerinin mesajın Header bölümüne yerleştirilmesini sağlamak için hangi değişikliklerin yapılması gerektiği anlatılmaktadır.

- HL7 Katmanı Uygulamanın Kullanılması

Artık tüm altyapı hazır olduktan sonra geriye HL7 Katmanı Uygulamasının doğru argümanlar verilerek kullanılması kalmaktadır. Bu adımda ant ve build.xml dosyaları yardımıyla hangi servisin hangi mesajlarla çağrılabileceğini belirleyerek HL7 Katmanı Uygulamasının web servisini çağrıması sağlanmaktadır.

4.2 Gereksinimler

USBS Web servislerini çağrımak için kullanılan java dilindeki istemci kodları web servislerin wsdl dosyalarından açık kaynak araçlar olan “Apache Axis2/Java” ve “Apache Ant” yardımıyla üretilmiştir. Bu dokümdan anlatılan yöntemle Web servisi istemcisi oluşturmak için gerekli araçlar ve versiyonları aşağıda verilmiştir.

- Java Platform Standard Edition (Java SE) Development Kit 6²⁸
 - Java, Sun Microsystems tarafından geliştirilmeye başlanmış açık kodlu, nesneye yönelik, platformdan bağımsız, yüksek performanslı, çok işlevli, yüksek seviye, adım adım işletilen (interpreted) bir dildir.²⁹
 - Problemle karşılaşılması durumunda “Java SE Runtime Environment 6 Update 3”³⁰ versiyonu kullanılması önerilir.
- Apache Ant³¹
 - Apache Ant yazılım geliştirme sürecini otomatikleştirmek için bir araçtır. “Make”e benzer olmasına karşın Java dilinde yazılmış ve Java platformu gerektirmektedir. Java projeleri için çok uygundur.
- Apache Axis2/Java v1.3³²
 - Apache Axis2 Web servisleri için açık kaynak, XML'e dayalı bir geliştirme iskeleti sunmaktadır. Geniş çaplı kullanılmış olan Apache Axis SOAP yapısının tamamen yeniden tasarlanmış ve yazılmış versiyonudur.
 - **Axis2'nin 1.3 versiyonu olması zorunludur.** Diğer versiyonlarında problemlerle karşılaşılmıştır.
- Rampart 1.3³³

²⁸ <http://java.sun.com/javase/downloads/index.jsp>

²⁹ [http://tr.wikipedia.org/wiki/Java_\(programlama_dili\)](http://tr.wikipedia.org/wiki/Java_(programlama_dili))

³⁰ https://cds.sun.com/is-bin/INTERSHOP.enfinity/WFS/CDS-CDS_Developer-Site/en_US/-/USD/ViewProductDetail-Start?ProductRef=jdk-6u3-oth-JPR@CDS-CDS_Developer

³¹ <http://ant.apache.org/>

³² <http://ws.apache.org/axis2/>

³³ http://ws.apache.org/axis2/modules/rampart/1_3/security-module.html

- Apache Software Foundation tarafından Axis2 için geliştirilen WS-Security standartı uygulamasıdır. Web Servisleri için güvenlik (security) özellikleri sağlamaktadır.
- **Rampart'ın mutlaka 1.3 versiyonu kullanılması gerekmektedir.** Diğer versyonlarla denendiğinde güvenli web servis istemcisi üretmekte sorunlarla karşılaşılmıştır.

4.3 İstemci Kodu Üretimi

Bu bölümde örnek bir USBS Web servisi olarak Muayene Web servisi için istemci kodu üretimi anlatılmaktadır. Bu bölüm yukarıdaki gereksinimler kısmından **Java** ve **Apache Ant** araçlarının halihazırda kurulmuş ve kullanılmaya hazır olduğunu varsaymaktadır. **Apache Axis2'nin** ve **Rampart** modülünün kullanılması ise raporda anlatılacaktır.

4.3.1 Ortamın hazırlanması

- Bilgisayarda istenilen bir yerde yeni bir klasör oluşturulur ve istenilen bir isim verilir. Bu dokümda buradan itibaren **[USBS_Istemci]** denildiğinde bu oluşturulmuş olan ana klasör kastedilecektir.
- Bölüm 4.4'de verilen ant build dosyası (build.xml) **[USBS_Istemci]** klasörünün altına kopyalanır.
- **[USBS_Istemci]**'nin altına **[USBS_Istemci]/resources/wsdl/Muayene** yolu oluşturacak şekilde gerekli klasörler oluşturulur.
 - **[USBS_Istemci]** klasörünün içine gidilir ve resources adında yeni bir klasör oluşturulur.
 - resources klasörünün içine gidilir ve wsdl adında yeni klasör oluşturulur.
 - wsdl klasörünün içine gidilir ve Muayene adında yeni klasör oluşturulur.
- Muayene wsdl dosyası ve ilgili xsd dosyaları (MCCI_IN000001TR01.xsd, MCCI_IN000002TR01.xsd, POCD_MT000005TR01.xsd, QUCD_MT000001TR01.xsd, QUQI_IN000001TR01.xsd ve QUQI_IN000002TR01.xsd) yeni oluşturulan Muayene klasörünün altına kopyalanır.
 - Muayene wsdl dosyası (MCCI_AR000001TR01.wsdl) ve ilgili xsd dosyaları kopyalanır.
 - **[USBS_Istemci]/resources/wsdl/Muayene** dizininin altına gidilir.
 - Kopyalanan dosyalar bu dizine yapıştırılır.
- **[USBS_Istemci]/resources/wsdl/coreschemas** yolu oluşturacak şekilde gerekli klasörler oluşturulur.
 - **[USBS_Istemci]/resources/wsdl** dizinine gidilir ve coreschemas adında yeni bir klasör oluşturulur.

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

- USBS web servislerinin kullandığı ortak xsd dosyaları (*datatype.xsd*, *datatype-base.xsd*, *infrastructureRoot.xsd*, *NarrativeBlock.xsd*, *USBSVoc.xsd* ve *voc.xsd*) **[USBS_Istemci]/resources/wsdl/coreschemas** dizinine kopyalanır.
 - Ortak xsd dosyaları kopyalanır.
 - **[USBS_Istemci]/ resources/wsdl/coreschemas** dizininin altına gidilir.
 - Kopyalanan dosyalar bu dizine yapıştırılır.
- [USBS_Istemci]'nin altına **[USBS_Istemci]/lib/axis** ve **[USBS_Istemci]/lib/rampart** klasörleri oluşturulur.
 - [USBS_Istemci] klasörünün içine gidilir ve lib adında yeni bir klasör oluşturulur.
 - lib klasörünün içine gidilir ve axis adında yeni klasör oluşturulur.
 - lib klasörünün içinde rampart adında yeni klasör oluşturulur.
- Axis2 v1.3 http://apache.karegen.com/ws/axis2/1_3/axis2-1.3-bin.zip adresinden indirilir ve zip dosyasının içinde axis2-1.3/lib klasörünün altındaki tüm dosyalar **[USBS_Istemci]/lib/axis** dizininin altına kopyalanır.
- Rampart modülü http://www.apache.org/dyn/mirrors/mirrors.cgi/ws/rampart/1_3/rampart-1.3.zip adresinden indirilir ve zip dosyasının içindeki rampart-1.3/lib klasörünün altındaki tüm dosyalar **[USBS_Istemci]/lib/rampart** dizininin altına kopyalanır.
- **[USBS_Istemci]** klasörünün altına **conf** klasörü oluşturulur.
- Bölüm 4.4'deki client.axis2.xml dosyası **[USBS_Istemci]/conf** dizininin altına kopyalanır.
 - **[USBS_Istemci]/conf** dizinine gidilir.
 - Bu dizinde client.axis2.xml adında yeni bir dosya oluşturulur.
 - Bölüm 4.4'deki içerik bu yeni dosyanın içine kopyalanır ve kaydedilir.
- **[USBS_Istemci]/src/client** yolu oluşturacak şekilde gerekli klasörler oluşturulur.
 - **[USBS_Istemci]** klasörünün altında **src** adında bir klasör oluşturulur.
 - **[USBS_Istemci]/src** dizininin içine gidilip client adında yeni bir klasör oluşturulur.

4.3.2 Kodların Üretilmesi

- Komut satırı açılıp **[USBS_Istemci]** klasörüne gidilir.

- Komut satırında **ant generate.client.codes1** yazılmıştır. Bu komut bazı bilgisayarlarda “BUILD FAILED
[USBS_Istemci]\build.xml:159: java.lang.OutOfMemoryError: Java heap space” hatasına neden olabilmektedir. **Bu hata ile karşılaşılısa bile kodlar doğru olarak üretilmiştir.**
- build.xml dosyası açılmıştır. **generate.client.codes1**’in içeriğine bakıldığından, dataBindingName’ın jaxbri olarak belirlendiği görülmektedir. JAXB xsd dosyalarından yüksek seviye programlama dilinde bu xsd dosyalarına karşılık gelecek kodlar üreten piyasadaki binding aracından bir tanesidir. Axis2 nin birincil kullandığı ADT binding yerine JAXB kullanılmasının nedeni, ADT’nin verilen xsd’ler için kod üretirken hata vermesi dolayısıyladır.
- Komut çalıştığından üretilecek kodlar **[USBS_Istemci]/src/v3** klasörünün altında bulunmaktadır.
- Kodlar üretilmiş olmasına rağmen, axis2-1.3 versiyonunda önceki versiyonlardan farklı olarak otomatik üretilecek kodlarda yapılan bir değişiklikten dolayı USBS servislerinin güvenlik ile çağrılmamasında problemler çıkmaktadır. Bu sebeple üretilecek kodları kullanabilmek için aşağıdaki değişiklikler yapılmalıdır.
- **[USBS_Istemci]/src/v3/hl7_org/Muayene** yolu izlenerek üretilecek kodlar bulunur. **MCCI_AR000001TR_ServiceStub.java** dosyası açılır.
 - Dosyanın içindeki tüm (6 adet) **QName("urn:hl7-org:v3")** stringleri **QName("")** ile değiştirilir.
 - Dosyanın içindeki tüm (4 adet) **factory.createOMNamespace("")** stringleri **factory.createOMNamespace("urn:hl7-org:v3")** ile değiştirilir.
 - Dosya kaydedilip kapatılır.
- Bu değişikliklerden sonra istemci alt yapısı kullanılmaya hazırlanır.

4.3.3 Güvenlik (Security) Eklenmesi

- **[USBS_Istemci]/src/client** dizinine gidilir. **CustomSecurityHandler.java** adında bir dosya oluşturulur ve dosyanın içine aşağıdaki kod yazılmıştır, dosya kaydedilir.
 - **[USBS_Istemci]/src/client** dizinine gidilir.
 - Bu dizinde **CustomSecurityHandler.java** adında yeni bir dosya oluşturulur.
 - Aşağıdaki tablodaki içerik bu yeni dosyanın içine kopyalanır ve kaydedilir.

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
package client;

import org.apache.axiom.soap.SOAPEnvelope;
import org.apache.axiom.soap.impl.builder.StAXSOAPModelBuilder;

import org.apache.axis2.AxisFault;
import org.apache.axis2.context.MessageContext;
import org.apache.axis2.handlers.AbstractHandler;

import org.apache.ws.security.message.WSSecHeader;
import org.apache.ws.security.message.WSSecTimestamp;
import org.apache.ws.security.message.WSSecUsernameToken;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;

import java.io.*;

import java.util.ResourceBundle;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamException;
import javax.xml.stream.XMLStreamReader;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

public class CustomSecurityHandler extends AbstractHandler {
    public InvocationResponse invoke(MessageContext messageContext)
        throws AxisFault {
        ResourceBundle resourceBundle =
        ResourceBundle.getBundle("client_conf");
        String username = resourceBundle.getString("username");
        String password = resourceBundle.getString("password");

        SOAPEnvelope soapEnvelope = messageContext.getEnvelope();
        soapEnvelope.getHeader().detach();

        WSSecUsernameToken wssAddUsernameToken = new WSSecUsernameToken();
        wssAddUsernameToken.setPasswordType(
            "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText");
        wssAddUsernameToken.setUserInfo(username, password);
        wssAddUsernameToken.addNonce();
        wssAddUsernameToken.addCreated();

        //get the factory
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
        dbf.setNamespaceAware(true);

        String uuid = java.util.UUID.randomUUID().toString() + ".xml";
        Document dom = null;

        try {
            //Using factory get an instance of document builder
            DocumentBuilder db = dbf.newDocumentBuilder();

            //parse using builder to get DOM representation of the XML file
            String myString = soapEnvelope.toString().trim();
```

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
InputStream is = new ByteArrayInputStream(myString.getBytes("UTF-8"));
    dom = db.parse(is);
} catch (ParserConfigurationException pce) {
    pce.printStackTrace();
} catch (SAXException se) {
    se.printStackTrace();
} catch (IOException ioe) {
    ioe.printStackTrace();
}

WSSecHeader wsSecHeader = new WSSecHeader();
wsSecHeader.setMustUnderstand(false);
wsSecHeader.insertSecurityHeader(dom);

WSSecTimestamp wsSecTimestamp = new WSSecTimestamp();
wsSecTimestamp.setTimeToLive(300);

wsSecTimestamp.build(dom, wsSecHeader);
wssAddUsernameToken.build(dom, wsSecHeader);

// let's first read our soap xml files in to StAXSOAPModelBuilder
String result = dom2String(dom);
System.out.println("result:::" + result);

StringReader soapFileReader = new StringReader(result);
XMLStreamReader parser = null;

try {
    parser = XMLInputFactory.newInstance()
        .createXMLStreamReader(soapFileReader);
} catch (XMLStreamException e) {
    e.printStackTrace();
}

StAXSOAPModelBuilder builder = new StAXSOAPModelBuilder(parser, null);
SOAPEnvelope envelope = (SOAPEnvelope) builder.getDocumentElement();
messageContext.setEnvelope(envelope);

return InvocationResponse.CONTINUE;
}

// This method writes a DOM document to a string
public static String dom2String(Document doc) {
    try {
        // Prepare the DOM document for writing
        Source source = new DOMSource(doc);

        // Prepare the output file
        StringWriter writer = new StringWriter();
        Result result = new StreamResult(writer);

        // Write the DOM document to the string
        Transformer xformer = TransformerFactory.newInstance()
            .newTransformer();
        xformer.transform(source, result);

        return writer.toString();
    } catch (TransformerConfigurationException e) {
    } catch (TransformerException e) {
    }

    return null;
}
}
```

- [USBS_Istemci]/conf klasörüne gidilip **client_conf.properties** adında bir dosya oluşturulur ve dosyanın içine aşağıdaki kod yazılıp, dosya kaydedilir.
 - [USBS_Istemci]/conf dizinine gidilir.
 - Bu dizinde **client_conf.properties** adında yeni bir dosya oluşturulur.
 - Aşağıdaki içerik bu yeni dosyanın içine kopyalanır ve kaydedilir.

```
address.server = http://www.sagliknet.saglik.gov.tr:7750  
username = ALINAN_KULLANICI_ADI  
password = ALINAN_SIFRE
```

4.3.4 İstemcinin kullanılması

Yukarıdaki işlemlerden sonra USBS Muayene Web Servisi istemcisi kullanıma hazırdır. Aşağıda nasıl kullanılabileceğini gösteren bir örnek verilmektedir.

- [USBS_Istemci]/src/client klasörüne gidilir, SaglikNetClient.java adında bir dosya oluşturulur ve dosyanın içine aşağıdaki kod yazılıp, dosya kaydedilir.
 - [USBS_Istemci]/src/client dizinine gidilir.
 - Bu dizinde SaglikNetClient.java adında yeni bir dosya oluşturulur.
 - Aşağıdaki tablolardaki içerik bu yeni dosyanın içine kopyalanır ve kaydedilir.

```
package client;

import java.io.File;
import java.io.FileWriter;

import java.util.ResourceBundle;

import javax.xml.XMLConstants;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.Marshaller;
import javax.xml.stream.XMLStreamWriter;
import javax.xml.validation.Schema;
import javax.xml.validation.SchemaFactory;

public class SaglikNetClient {
    public static final String COMMIT_ERROR = "CE";
    public static final int TIMEOUT = 15000;

    private String fileName;
    private String serviceName;
    private String uuid;
    private boolean isFirst;

    //    private boolean inspectMessages;
    private String serverAddress = null;

    public SaglikNetClient(String fileName, String serviceName,
                          boolean isFirst, String uuid) {
        init();
        this.fileName = fileName;
        this.serviceName = serviceName;
        this.isFirst = isFirst;
        this.uuid = uuid;
    }
}
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
private void init() {
    ResourceBundle resourceBundle =
ResourceBundle.getBundle("client_conf");
    serverAddress = resourceBundle.getString("adress.server");
}

private String firstWebService(String endpointURL, String fileName)
throws java.lang.Exception {
    javax.xml.bind.JAXBContext context;
    javax.xml.bind.Unmarshaller unmarshaller;
    javax.xml.bind.Marshaller marshaller;
    javax.xml.bind.JAXBElement object;

    String result = "";

    v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub stub16 = new
v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub(endpointURL);
    stub16._getServiceClient().getOptions()
        .setTimeOutInMilliSeconds(TIMEOUT);

    context =
javax.xml.bind.JAXBContext.newInstance(v3.hl7_org.Muayene.MCCIIN000001TR01Messa
ge.class);
    unmarshaller = context.createUnmarshaller();
    object = (javax.xml.bind.JAXBElement) unmarshaller.unmarshal(new
java.io.File(
        fileName));

    v3.hl7_org.Muayene.MCCIIN000001TR01Message message16 =
(v3.hl7_org.Muayene.MCCIIN000001TR01Message) object.getValue();
    v3.hl7_org.Muayene.MCCIIN000001TR01ControlActEvent
controlActEvent16 =
    message16.getControlActEvent();
    v3.hl7_org.Muayene.MCCIIN000001TR01Subject subject16 =
controlActEvent16.getSubject();
    subject16.getExamination().getId().setExtension(this.uuid);

    v3.hl7_org.Muayene.MCCIIN000002TR01Message output16 =
stub16.MCCI_AR000001TR_MCCI_IN000001TR(message16);
    result = output16.getAcknowledgement().getTypeCode().getCode();

    if (result.equals(COMMIT_ERROR)) {
        return result + " - " +
            output16.getControlActEvent().getValue().getReason().get(0)
                .getDetectedIssueEvent().getCode().getDisplayName();
    } else {
        return result;
    }
}

private String secondWebService(String endpointURL, String fileName)
throws java.lang.Exception {
    javax.xml.bind.JAXBContext context;
    javax.xml.bind.Unmarshaller unmarshaller;
    javax.xml.bind.JAXBElement object;

    v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub stub16 = new
v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub(endpointURL);
    stub16._getServiceClient().getOptions()
        .setTimeOutInMilliSeconds(TIMEOUT);
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
        context =
javax.xml.bind.JAXBContext.newInstance(v3.hl7._org.Muayene.MCCIIN000001TR01Message.class);
        unmarshaller = context.createUnmarshaller();
        object = (javax.xml.bind.JAXBElement) unmarshaller.unmarshal(new
java.io.File(
                fileName));
        v3.hl7._org.Muayene.QUQIIN000001TR01Message message16 =
(v3.hl7._org.Muayene.QUQIIN000001TR01Message) object.getValue();
        v3.hl7._org.Muayene.QUQIIN000001TR01ControlActEvent controlActEvent16 =
        message16.getControlActEvent();
        v3.hl7._org.Muayene.QUQIIN000001TR01QueryByParameter queryByParameter16 =
        controlActEvent16.getQueryByParameter();
        queryByParameter16.getClinicalDocumentId().getValue()
                .setExtension(this.uuid);
        v3.hl7._org.Muayene.QUQIIN000002TR01Message output16 =
stub16.MCCI_AR000001TR_QUQI_IN000001TR(message16);
        return output16.getAcknowledgement().getTypeCode().getCode() + ":" +
output16.getControlActEvent().getQueryAck().getQueryResponseCode()
                .getCode();
    }
    public String execute() {
        String endpointURL = serverAddress + "/Muayene/services/MCCI_AR000001TR_Service";
        if (isFirst) {
            try {
                return firstWebService(endpointURL, fileName);
            } catch (Exception e) {
                e.printStackTrace();
            }
        } else {
            try {
                return secondWebService(endpointURL, fileName);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return "FAILED";
    }
    public static void main(String[] argv) {
        try {
            String fileName = argv[0];
            String serviceName = argv[1];
            String firstOrSecond = argv[2];
            String uuid = argv[3];
            try {
                SaglikNetClient testcase = new SaglikNetClient(fileName,
                        serviceName,
                        Boolean.valueOf(firstOrSecond),
                        uuid);
                testcase.execute();
            } catch (Exception e) {
                e.printStackTrace();
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        } {}
    }
}
```

- **[USBS_Istemci]/resources** klasörünün altına kullanılmak istenen web servisi inputu Muayene.xml dosyasının içine kaydedilir. Örnek bir muayene mesajı Ekte bulunabilir.
 - **[USBS_Istemci]/resources** dizinine gidilir.
 - Muayene.xml adında yeni bir dosya oluşturulur.
 - Kullanılmak istenen web servisi girdisi bu dosyanın içine yazılır ve dosya kaydedilir.
- Komut satırı açılıp **[USBS_Istemci]** klasörüne gidilir. Burada **ant compile2** komutu ile tüm kod derlenir ve daha sonra **ant run.test** komutu ile web servisi istemcisi çalıştırılır.
 - Komut satırı açılır.
 - **[USBS_Istemci]** klasörüne gidilir. (örnek “cd c:\codes\USBS_Istemci”)
 - Komut satırına **ant compile2** yazılır ve Enter'a basılır.
 - Komut satırına **ant run.test** yazılır ve Enter'a basılır. Bu komutla birlikte USBS web servisi çağrılmış bulunmaktadır.

4.3.5 Build.xml Dosyasına Yakından Bakış

Bir üstteki bölümde web servisini çağırmak için kullanılan build.xml dosyasındaki **run.test** görevi aşağıdaki gibidir.

```
<target name="run.test">
  <java fork="yes" className="client.SaglikNetClient">
    <classpath refid="classpath"/>
    <jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/>
    <arg value="${resources.dir}/Muayene.xml"/>
    <arg value="Muayene"/>
    <arg value="true"/>
    <arg value="b89a6520-0494-11dd-95ff-0800200c9a66"/>
  </java>
</target>
```

className hangi java dosyasının çalıştırılacağını belirtmektedir. Örnekte bir önceki bölümde yazdığımız client package'ının altındaki SaglikNetClient.java dosyasının main methodunun çağrılacağı belirtilmektedir. Eğer başka bir istemci kullanılmak istenirse burdaki className yeni istemciyi gösterecek şekilde değiştirilmelidir.

<jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/> satırı ise axis2'nin web servisini çağıırırken konfigurasyon için hangi dosyayı kullanacağını gösterir. Bu kısımda değişiklik yapmaya gerek yoktur.

Sonraki **arg value** ile başlayan 4 satır ise istemciye verilen argümanlardır. Sırasıyla hangi xml instance'ı kullanılacağı, hangi USBS web servisinin çağrılacağı, birinci ya da ikinci operasyondan hangisinin çağrılacağı (true ise birinci operasyon çağrılmaktadır.) ve kullanılacak uuid değeri belirtilmektedir.

Gerekli olduğunda bu argümanlar değiştirilip, dosya kaydedildikten sonra **ant run.test** komutu ile USBS web servisi istenilen yeni argümanlarla çağrılabılır.

4.3.6 İstemciye (`SaglikNetClient.java`) Yakından Bakış

```
(1) v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub stub16 = new  
     v3.hl7_org.Muayene.MCCI_AR000001TR_ServiceStub(endpointURL);  
(2) stub16._getServiceClient().getOptions()  
     .setTimeOutInMilliSeconds(TIMEOUT);
```

(1) numaralı satır Muayene Web Servisi için bir Stub objesi oluşturur, bunun için de servisin bulunduğu adresi parametre olarak alır. Buradaki Stub, Axis2 tarafından otomatik olarak oluşturulmuştur. (2) numaralı satır bu Stub için “timeout” süresini ayarlar, zorunlu değildir ancak olası gecikmeleri ele almak için 15 saniye gibi bir değer verilmiştir.

```
(3) context =  
     javax.xml.bind.JAXBContext.newInstance(v3.hl7_org.Muayene.MCCIIN000001TR01Message.class);  
(4) unmarshaller = context.createUnmarshaller();  
(5) object = (javax.xml.bind.JAXBElement) unmarshaller.unmarshal(new  
     java.io.File(fileName));
```

Bu 3 satır kod “fileName” ile yolu belirtilen Muayene mesajı XML dosyasını okur ve obje modelini çıkarır. Bu işleme “unmarshalling” denir. Bu kısmı halleden kütüphane JAXB’dir.

```
(6) v3.hl7_org.Muayene.MCCIIN000001TR01Message message16 =  
     (v3.hl7_org.Muayene.MCCIIN000001TR01Message) object.getValue();  
(7) v3.hl7_org.Muayene.MCCIIN000001TR01ControlActEvent controlActEvent16 =  
     message16.getControlActEvent();  
(8) v3.hl7_org.Muayene.MCCIIN000001TR01Subject subject16 =  
     controlActEvent16.getSubject();  
(9) subject16.getExamination().getId().setExtension(this.uuid);
```

Bu 4 satır ise, obje modelinden message16 adında bir v3.hl7_org.Muayene.MCCIIN000001TR01Message objesi oluşturur ve daha sonra Muayene klinik dokümanının “id” elemanının “extension” alanına verilen **UUID değerini** yerleştirir. Verilen bu UUID değerinin sorgulama işlemi için saklanması gerekmektedir.

```
(10) v3.hl7_org.Muayene.MCCIIN000002TR01Message output16 =  
     stub16.MCCI_AR000001TR_MCCI_IN000001TR(message16);  
(11) result = output16.getAcknowledgement().getTypeCode().getCode();
```

(10) numaralı satır, mesajın bulunduğu message16 objesini (1) numaralı satırda oluşturulan stub16 objesine geçirerek veri gönderim operasyonu olan MCCI_IN000001TR operasyonunu çağırır. Dönen cevap ise output16 adında bir v3.hl7_org.Muayene.MCCIIN000002TR01Message objesi oluşturur. (11) numaralı satırda dönen cevabın kodu result adında bir String objesine yazılır.

```
(12) if (result.equals(COMMIT_ERROR)) {  
(13)     return result + " - " +  
             output16.getControlActEvent().getValue().getReason().get(0)  
                 .getDetectedIssueEvent().getCode().getDisplayName();  
(14) } else {  
(15)     return result;  
(16) }
```

Bu son satırlarda ise, result objesinde saklanan cevap kodu kontrol edilir ve eğer değeri “CE” ise, (13)'üncü satırda bu hatanın detayı result objesine eklenir. Değer “CE” değil de başka bir seyse, (“CA” olmasını bekleriz), ekleme yapmadan kodu döner.

4.4 Java İstemcisi İçin Gerekli Kodlar

build.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="SaglikNetTest" default="build" basedir=".">
    <description>
        Saglik-NET Test Client
    </description>
    <property name="src.dir" location="src"/>
    <property name="build.dir" location="build"/>
    <property name="lib.dir" location="lib"/>
    <property name="conf.dir" location="conf"/>
    <property name="temp.dir" location="temp"/>
    <property name="resources.dir" location="resources"/>
    <property name="out.dir" location="out"/>
    <property name="project.properties" value="${conf.dir}/project.properties"/>
    <property file="${project.properties}"/>
    <path id="classpath">
        <pathelement location="${build.dir}"/>
        <pathelement location="${conf.dir}"/>
        <pathelement location="${temp.dir}"/>
        <fileset dir="${lib.dir}">
            <include name="**/*.jar"/>
        </fileset>
    </path>
    <target name="init">
        <mkdir dir="${build.dir}"/>
    </target>
    <target name="compile" depends="init" description="compiles the source codes and puts under build dir">
        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/AgizveDisSagligi" destdir="${build.dir}" debug="true"
        debuglevel="lines,vars,source">
            <classpath refid="classpath"/>
        </javac>
        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Asi" destdir="${build.dir}" debug="true"
        debuglevel="lines,vars,source">
            <classpath refid="classpath"/>
        </javac>
        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/BebekCocukBeslenme" destdir="${build.dir}"
        debug="true" debuglevel="lines,vars,source">
            <classpath refid="classpath"/>
        </javac>
        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/BebekCocukIzlem" destdir="${build.dir}" debug="true"
        debuglevel="lines,vars,source">
            <classpath refid="classpath"/>
        </javac>
    </target>

```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/BebekCocukPsikososyalIzlem" destdir="${build.dir}"  
debug="true" debuglevel="lines,vars,source">  
    <classpath refid="classpath"/>  
    </javac>  
    <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/BulasiciHastalikKesinVakaBildirimi"  
destdir="${build.dir}" debug="true" debuglevel="lines,vars,source">  
        <classpath refid="classpath"/>  
        </javac>  
        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/BulasiciHastalikOlasiVakaBildirimi"  
destdir="${build.dir}" debug="true" debuglevel="lines,vars,source">  
            <classpath refid="classpath"/>  
            </javac>  
            <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Diyabet" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                <classpath refid="classpath"/>  
                </javac>  
                <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/DogumBildirim" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                    <classpath refid="classpath"/>  
                    </javac>  
                    <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Gebelik" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                        <classpath refid="classpath"/>  
                        </javac>  
                        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/GebePsikososyalIzlem" destdir="${build.dir}"  
debug="true" debuglevel="lines,vars,source">  
                            <classpath refid="classpath"/>  
                            </javac>  
                            <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/GebelikSonucu" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                                <classpath refid="classpath"/>  
                                </javac>  
                                <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/HastaOzlukBilgileri" destdir="${build.dir}"  
debug="true" debuglevel="lines,vars,source">  
                                    <classpath refid="classpath"/>  
                                    </javac>  
                                    <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/KadinIzlem" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                                        <classpath refid="classpath"/>  
                                        </javac>  
                                        <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Kanser" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                                            <classpath refid="classpath"/>  
                                            </javac>  
                                            <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Lohusalizlem" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                                                <classpath refid="classpath"/>  
                                                </javac>  
                                                <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Muayene" destdir="${build.dir}" debug="true"  
debuglevel="lines,vars,source">  
                                                    <classpath refid="classpath"/>  
                                                    </javac>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/OlumBildirim" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/TetkikSonucu" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/VatandasYabanciKayit" destdir="${build.dir}"
debug="true" debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/VatansizKayit" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/YatanHasta" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/YenidoganKayit" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
<javac deprecation="on" srcdir="${src.dir}/client" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
    <classpath refid="classpath"/>
</javac>
</target>
<target name="compile2" depends="init" description="compiles the source codes and puts under build dir">
    <javac deprecation="on" srcdir="${src.dir}/v3/hl7_org/Muayene" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
        <classpath refid="classpath"/>
    </javac>
    <javac deprecation="on" srcdir="${src.dir}/client" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
        <classpath refid="classpath"/>
    </javac>
</target>
<target name="compile0" depends="init" description="compiles the source codes and puts under build dir">
    <javac deprecation="on" srcdir="${src.dir}/client" destdir="${build.dir}" debug="true"
debuglevel="lines,vars,source">
        <classpath refid="classpath"/>
    </javac>
</target>
<target name="build" depends="init, compile" description="Install the software"/>
<target name="clean" description="cleans up class-files">
    <delete dir="${build.dir}"/>
</target>
<target name="declare">
    <taskdef name="codegen" classname="org.apache.axis2.tool.ant.AntCodegenTask"
classpathref="classpath"/>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
</target>
<target name="generate.client.codes1" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Muayene/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Muayene" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes2" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/15-49yasKadinIzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.KadinIzlem" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes3" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/AgizveDisSagligi/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.AgizveDisSagligi" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes4" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Asi/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Asi" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes5" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukBeslenme/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.BebekCocukBeslenme" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes6" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukPsikososyalIzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.BebekCocukPsikososyalIzlem" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes7" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukIzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.BebekCocukIzlem" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes8" depends="declare">
    <codegen
wsdlfilename="${resources.dir}/wsdl/BulasiciHastalikKesinVakaBildirimi/MCCI_AR000001TR01.wsdl"
namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.BulasiciHastalikKesinVakaBildirimi"
databindingName="jaxbri"/>
</target>
<target name="generate.client.codes9" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/BulasiciHastalikOlasiVakaBildirimi/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.BulasiciHastalikOlasiVakaBildirimi"
    databindingName="jaxbri"/>
</target>
<target name="generate.client.codes10" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Diyabet/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Diyabet" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes11" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/DogumBildirim/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.DogumBildirim" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes12" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/GebePsikososyalIzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.GebePsikososyalIzlem" databindingName="jaxbri"/>
</target>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<target name="generate.client.codes13" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Gebelzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Gebelzlem" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes14" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/GebelikSonucu/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.GebelikSonucu" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes15" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/HastaOzlukBilgileri/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.HastaOzlukBilgileri" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes16" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Kanser/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Kanser" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes17" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Lohusalzlem/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.Lohusalzlem" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes18" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/TetkikSonucu/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.TetkikSonucu" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes19" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/VatandasYabanciKayit/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.VatandasYabanciKayit" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes20" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/VatansizKayit/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.VatansizKayit" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes21" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/YatanHasta/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.YatanHasta" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes22" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/YenidoganKayit/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.YenidoganKayit" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes23" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/OlumBildirim/MCCI_AR000001TR01.wsdl"
    namespaceToPackages="urn:hl7-org:v3=v3.hl7_org.OlumBildirim" databindingName="jaxbri"/>
</target>
<target name="generate.client.codes" depends="declare">
    <codegen wsdlfilename="${resources.dir}/wsdl/Muayene/MCCI_AR000001TR01.wsdl"
    output="${src.dir}/Muayene" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
    <codegen wsdlfilename="${resources.dir}/wsdl/15-49yasKadinIzlem/MCCI_AR000001TR01.wsdl"
    output="${src.dir}/15-49yasKadinIzlem" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
    <codegen wsdlfilename="${resources.dir}/wsdl/AgizveDisSagligi/MCCI_AR000001TR01.wsdl"
    output="${src.dir}/AgizveDisSagligi" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<codegen wsdlfilename="${resources.dir}/wsdl/Asi/MCCI_AR000001TR01.wsdl" output="${src.dir}/Asi"
databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukBeslenme/MCCI_AR000001TR01.wsdl"
output="${src.dir}/BebekCocukBeslenme" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukPsikosyallzlem/MCCI_AR000001TR01.wsdl"
output="${src.dir}/BebekCocukPsikosyallzlem" databindingName="jaxbri" testcase="true"
generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/BebekCocukIzlem/MCCI_AR000001TR01.wsdl"
output="${src.dir}/BebekCocukIzlem" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen
wsdlfilename="${resources.dir}/wsdl/BulasiciHastalikKesinVakaBildirimi/MCCI_AR000001TR01.wsdl"
output="${src.dir}/BulasiciHastalikKesinVakaBildirimi" databindingName="jaxbri" testcase="true"
generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/BulasiciHastalikOlasiVakaBildirimi/MCCI_AR000001TR01.wsdl"
output="${src.dir}/BulasiciHastalikOlasiVakaBildirimi" databindingName="jaxbri" testcase="true"
generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/Diyabet/MCCI_AR000001TR01.wsdl"
output="${src.dir}/Diyabet" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/DogumBildirim/MCCI_AR000001TR01.wsdl"
output="${src.dir}/DogumBildirim" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/GebePsikosyallzlem/MCCI_AR000001TR01.wsdl"
output="${src.dir}/GebePsikosyallzlem" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/GebelikSonucu/MCCI_AR000001TR01.wsdl"
output="${src.dir}/GebelikSonucu" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/HastaOzlukBilgileri/MCCI_AR000001TR01.wsdl"
output="${src.dir}/HastaOzlukBilgileri" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/Kanser/MCCI_AR000001TR01.wsdl"
output="${src.dir}/Kanser" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/Lohusalzlem/MCCI_AR000001TR01.wsdl"
output="${src.dir}/Lohusalzlem" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/TetkikSonucu/MCCI_AR000001TR01.wsdl"
output="${src.dir}/TetkikSonucu" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/VatandasYabanciKayit/MCCI_AR000001TR01.wsdl"
output="${src.dir}/VatandasYabanciKayit" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/VatansizKayit/MCCI_AR000001TR01.wsdl"
output="${src.dir}/VatansizKayit" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/YatanHasta/MCCI_AR000001TR01.wsdl"
output="${src.dir}/YatanHasta" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/YenidoganKayit/MCCI_AR000001TR01.wsdl"
output="${src.dir}/YenidoganKayit" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
<codegen wsdlfilename="${resources.dir}/wsdl/OlumBildirim/MCCI_AR000001TR01.wsdl"
output="${src.dir}/OlumBildirim" databindingName="jaxbri" testcase="true" generateservicexml="true"/>
</target>
<target name="run.test">
<java fork="yes" className="client.SaglikNetClient">
<classpath refid="classpath"/>
<jvmarg value="-Daxis2.xml=../conf/client.axis2.xml"/>
<arg value="${resources.dir}/Muayene.xml"/>
<arg value="Muayene"/>
<arg value="true"/>
```

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<!--arg value="true"-->
<arg value="b89a6520-0494-11dd-95ff-0800200c9a66"/>
</java>
</target>
<target name="run.testWS1">
<java fork="yes" className="client.SaglikNetClient">
<classpath refid="classpath"/>
<jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/>
<!--jvmarg value="-Daxis2.repo=./axis2_repository"-->
<arg value="${resources.dir}/instances/birim/true/Lohusalzlem.xml"/>
<!--Instance file path-->
<arg value="Lohusalzlem"/>
<!-- Service name to be called -->
<arg value="true"/>
<!-- First service? -->
<!--arg value="true"-->
<arg value="951c1f6f-24d8-448b-8624-0ae55ccd5cd8"/>
<!--uuid for the first service-->
</java>
</target>
<target name="run.testWS2">
<java fork="yes" className="client.SaglikNetClient">
<classpath refid="classpath"/>
<jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/>
<!--jvmarg value="-Daxis2.repo=./axis2_repository"-->
<arg value="${resources.dir}/instances/query/Query.xml"/>
<!--Instance file path-->
<arg value="BulasiciHastalikKesinVakaBildirimi"/>
<!-- Service name to be called -->
<arg value="false"/>
<!-- First service? -->
<!--arg value="true"-->
<arg value="7c9bca67-2709-4369-9ed1-f656eb6464bc"/>
<!--uuid for the second service-->
</java>
</target>
<target name="run.masTestWS1">
<java fork="yes" className="client.SaglikNetMassiveTest">
<classpath refid="classpath"/>
<jvmarg value="-Xms256m"/>
<jvmarg value="-Xmx256m"/>
<jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/>
<arg value="WS1"/>
<!-- Which WS? -->
<arg value="true"/>
<!-- If it is WS1, true or false instances? -->
<arg value="${resources.dir}/instances/birim"/>
<!-- If it is WS1, the path to the main instances folder -->
</java>
</target>
<target name="run.masTestWS2">
<java fork="yes" className="client.SaglikNetMassiveTest">
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<classpath refid="classpath"/>
<jvmarg value="-Xms256m"/>
<jvmarg value="-Xmx256m"/>
<jvmarg value="-Daxis2.xml=./conf/client.axis2.xml"/>
<arg value="WS2"/>
<!-- Which WS? -->
<arg value="${resources.dir}/instances/query/Query.xml"/>
<!-- If it is WS2, the path to the query instance -->
<arg value="${out.dir}/WS1-TRUE-20080404-144027.csv"/>
<!-- If it is WS2, the path to the output of masTestWS1 -->
</java>
</target>
<!--
    "Kadinizlem";
    "AgizveDisSagligi";
    "Asi";
    "BebekCocukBeslenme";
    "BebekCocukIzlem" ;
    "BebekCocukPsikososyallzlem" ;
    "BulasiciHastalikKesinVakaBildirimi" ;
    "BulasiciHastalikOlasiVakaBildirimi" ;
    "Diyabet" ;
    "DogumBildirim" ;
    "Gebelzlem" ;
    "GebePsikososyallzlem" ;
    "GebelikSonucu" ;
    "HastaOzlukBilgileri" ;
    "Kanser" ;
    "Lohusalzlem" ;
    "Muayene" ;
    "OlumBildirim" ;
    "TetkikSonucu" ;
    "VatandasYabanciKayit" ;
    "VatansizKayit" ;
    "YatanHasta" ;
    "YenidoganKayit" ;
    -->
</project>
```

client.axis2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
!
! Copyright 2006 The Apache Software Foundation.
!
! Licensed under the Apache License, Version 2.0 (the "License");
! you may not use this file except in compliance with the License.
! You may obtain a copy of the License at
!
!   http://www.apache.org/licenses/LICENSE-2.0
!
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

! Unless required by applicable law or agreed to in writing, software
! distributed under the License is distributed on an "AS IS" BASIS,
! WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
! See the License for the specific language governing permissions and
! limitations under the License.

!-->

```
<axisconfig name="AxisJava2.0">

    <!-- ===== -->
    <!-- Parameters -->
    <!-- ===== -->
    <parameter name="hotdeployment" locked="false">true</parameter>
    <parameter name="hotupdate" locked="false">false</parameter>
    <parameter name="enableMTOM" locked="false">false</parameter>

    <!--During a fault, stacktrace can be sent with the fault message. The following flag will control -->
    <!--that behaviour.-->
    <parameter name="sendStacktraceDetailsWithFaults" locked="false">true</parameter>

    <!--If there aren't any information available to find out the fault reason, we set the message of the
exception-->
    <!--as the faultreason/Reason. But when a fault is thrown from a service or some where, it will be -->
    <!--wrapped by different levels. Due to this the initial exception message can be lost. If this flag-->
    <!--is set then, Axis2 tries to get the first exception and set its message as the faultreason/Reason.-->
    <parameter name="DrillDownToRootCauseForFaultReason" locked="false">false</parameter>

    <!--This is the user name and password of admin console-->
    <parameter name="userName" locked="false">admin</parameter>
    <parameter name="password" locked="false">axis2</parameter>

    <!--To override repository/services you need to uncomment following parameter and value SHOULD be
absolute file path.-->
    <!--<parameter name="services" locked="false">service</parameter>-->
    <!--To override repository/modules you need to uncomment following parameter and value SHOULD be
absolute file path-->
    <!--<parameter name="modules" locked="false">modules</parameter>-->

    <!--Following params will set the proper context paths for invocations. All the endpoints will have a
commons context-->
    <!--root which can be used using the following contextRoot parameter-->
    <!--<parameter name="contextRoot" locked="false">axis2</parameter>-->

    <!--Our HTTP endpoints can handle both REST and SOAP. Following parameters can be used to distinguish
those endpoints-->
    <!--<parameter name="servicePath" locked="false">services</parameter>-->
    <!--<parameter name="restPath" locked="false">rest</parameter>-->

    <!--Set the flag to true if you want to enable transport level session management-->
    <parameter name="manageTransportSession" locked="false">false</parameter>

    <!--Following two parameters will be used to handle REST in Axis2. The default settings will make Axis2 to
have two-->
```

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<!--different endpoints, one for REST (AxisRESTServlet) one for SOAP message handling (AxisServlet). But
following-->
<!--parameters help to tweak the message handling of two main servlets. -->

<!-- If the enableRESTInAxis2MainServlet is true, then Axis2MainServlet will handle both SOAP and REST
messages -->
<parameter name="enableRESTInAxis2MainServlet" locked="true">true</parameter>

<!-- Following parameter will completely disable REST handling in both the servlets-->
<parameter name="disableREST" locked="true">false</parameter>

<!-- This will disable the separate servlet we have for REST handling. -->
<parameter name="disableSeparateEndpointForREST" locked="true">false</parameter>

<!-- ===== -->
<!-- Message Receivers -->
<!-- ===== -->
<!--This is the Default Message Receiver for the system , if you want to have MessageReceivers for -->
<!--all the other MEP implement it and add the correct entry to here , so that you can refer from-->
<!--any operation -->
<!--Note : You can override this for particular service by adding the same element with your requirement-->
<messageReceivers>
    <messageReceiver mep="http://www.w3.org/2004/08/wsdl/in-only"
        class="org.apache.axis2.receivers.RawXMLINOnlyMessageReceiver"/>
    <messageReceiver mep="http://www.w3.org/2004/08/wsdl/in-out"
        class="org.apache.axis2.receivers.RawXMLINOutMessageReceiver"/>
</messageReceivers>

<!-- ===== -->
<!-- Target Resolvers -->
<!-- ===== -->
<!-- Uncomment the following and specify the class name for your TargetResolver to add -->
<!-- a TargetResolver. TargetResolvers are used to process the To EPR for example to -->
<!-- choose a server in a cluster -->
<!--<targetResolvers>-->
    <!--<targetResolver class="" />-->
<!--</targetResolvers>-->

<!-- ===== -->
<!-- Transport Ins -->
<!-- ===== -->
<transportReceiver name="http"
    class="org.apache.axis2.transport.http.SimpleHTTPServer">
    <parameter name="port" locked="false">6060</parameter>
    <!--If you want to give your own host address for EPR generation-->
    <!--uncomment following parameter , and set as you required.-->
    <!--<parameter name="hostname" locked="false">http://myApp.com/ws</parameter>-->
</transportReceiver>
```

```
<transportReceiver name="tcp"
    class="org.apache.axis2.transport.tcp.TCPServer">
    <parameter name="port" locked="false">6061</parameter>
    <!--If you want to give your own host address for EPR generation-->
    <!--uncomment following parameter , and set as you required.-->
    <!--<parameter name="hostname" locked="false">tcp://myApp.com/ws</parameter>-->
</transportReceiver>

<!-- ===== -->
<!-- Transport Outs -->
<!-- ===== -->

<transportSender name="jms"
    class="org.apache.axis2.transport.jms.JMSSender"/>
<transportSender name="tcp"
    class="org.apache.axis2.transport.tcp.TCPTransportSender"/>
<transportSender name="local"
    class="org.apache.axis2.transport.local.LocalTransportSender"/>
<transportSender name="http"
    class="org.apache.axis2.transport.http.CommonsHTTPTransportSender">
    <parameter name="PROTOCOL" locked="false">HTTP/1.1</parameter>
    <parameter name="Transfer-Encoding" locked="false">chunked</parameter>
</transportSender>
<transportSender name="https"
    class="org.apache.axis2.transport.http.CommonsHTTPTransportSender">
    <parameter name="PROTOCOL" locked="false">HTTP/1.1</parameter>
    <parameter name="Transfer-Encoding" locked="false">chunked</parameter>
</transportSender>

<!-- ===== -->
<!-- Phases -->
<!-- ===== -->
<phaseOrder type="InFlow">
    <!-- System pre-defined phases -->
    <phase name="Transport">
        <handler name="RequestURIBasedDispatcher"
            class="org.apache.axis2.engine.RequestURIBasedDispatcher">
            <order phase="Transport"/>
        </handler>
        <handler name="SOAPActionBasedDispatcher"
            class="org.apache.axis2.engine.SOAPActionBasedDispatcher">
            <order phase="Transport"/>
        </handler>
    </phase>
    <phase name="Security"/>
    <phase name="PreDispatch"/>
    <phase name="Dispatch" class="org.apache.axis2.engine.DispatchPhase">
        <handler name="AddressingBasedDispatcher"
            class="org.apache.axis2.engine.AddressingBasedDispatcher">
            <order phase="Dispatch"/>
        </handler>
    </phase>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<handler name="SOAPMessageBodyBasedDispatcher"
        class="org.apache.axis2.engine.SOAPMessageBodyBasedDispatcher">
    <order phase="Dispatch"/>
</handler>
<handler name="InstanceDispatcher"
        class="org.apache.axis2.engine.InstanceDispatcher">
    <order phase="Dispatch"/>
</handler>
</phase>
<!-- System pre defined phases -->
<!-- After Postdispatch phase module author or or service author can add any phase he want
-->
<phase name="OperationInPhase"/>
</phaseOrder>
<phaseOrder type="OutFlow">
    <!-- user can add his own phases to this area -->
    <phase name="OperationOutPhase"/>
    <!--system predefined phase-->
    <!--these phase will run irrespective of the service-->
    <phase name="PolicyDetermination"/>
    <phase name="MessageOut"/>
    <phase name="Security"/>
    <phase name="CustomSecurityPhase">
        <handler name="CustomSecurityHandler" class="client.CustomSecurityHandler"/>
    </phase>
</phaseOrder>
<phaseOrder type="InFaultFlow">
    <phase name="PreDispatch"/>
    <phase name="Dispatch" class="org.apache.axis2.engine.DispatchPhase">
        <handler name="RequestURIBasedDispatcher"
                class="org.apache.axis2.engine.RequestURIBasedDispatcher">
            <order phase="Dispatch"/>
        </handler>

        <handler name="SOAPActionBasedDispatcher"
                class="org.apache.axis2.engine.SOAPActionBasedDispatcher">
            <order phase="Dispatch"/>
        </handler>

        <handler name="AddressingBasedDispatcher"
                class="org.apache.axis2.engine.AddressingBasedDispatcher">
            <order phase="Dispatch"/>
        </handler>

        <handler name="SOAPMessageBodyBasedDispatcher"
                class="org.apache.axis2.engine.SOAPMessageBodyBasedDispatcher">
            <order phase="Dispatch"/>
        </handler>
        <handler name="InstanceDispatcher"
                class="org.apache.axis2.engine.InstanceDispatcher">
            <order phase="Dispatch"/>
        </handler>
```

```
        </handler>
    </phase>
    <!-- user can add his own phases to this area -->
    <phase name="OperationInFaultPhase"/>
</phaseOrder>
<phaseOrder type="OutFaultFlow">
    <!-- user can add his own phases to this area -->
    <phase name="OperationOutFaultPhase"/>
    <phase name="PolicyDetermination"/>
    <phase name="MessageOut"/>
</phaseOrder>
</axisconfig>
```

Örnek Muayene Mesajı

```
<?xml version="1.0" encoding="UTF-8"?>
<hl7:MCCI_IN000001TR01 xsi:schemaLocation="muayene.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:hl7="urn:hl7-org:v3">
    <hl7:id root="2.16.840.1.113883.3.129.2.1.2" extension="a"/>
    <hl7:creationTime/>
    <hl7:responseModeCode code="Q"/>
    <hl7:interactionId root="2.16.840.1.113883.3.129.2.1.1" extension="MCCI_IN000001TR01"/>
    <hl7:processingCode/>
    <hl7:processingModeCode code="T"/>
    <hl7:acceptAckCode code="AL"/>
    <hl7:receiver typeCode="RCV">
        <hl7:device>
            <hl7:id root="2.16.840.1.113883.3.129.1.1.5" extension="USBS"/>
        </hl7:device>
    </hl7:receiver>
    <hl7:sender typeCode="SND">
        <hl7:device>
            <hl7:id root="2.16.840.1.113883.3.129.1.1.5" extension="a"/>
        </hl7:device>
    </hl7:sender>
    <hl7:controlActEvent>
        <hl7:subject>
            <hl7:examination classCode="DOCLIN" moodCode="EVN">
                <hl7:id root="2.16.840.1.113883.3.129.2.1.3" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
                <hl7:code code="MUAYENE" displayName="Muayene MSVS" (Vatandaş/Yabancı)" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.2.2.1" codeSystemName="Döküman Tipi"/>
                    <hl7:effectiveTime value="20080311151312"/>
                    <hl7:confidentialityCode code="1" displayName="Normal" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.1.2.77" codeSystemName="Gizlilik"/>
                        <hl7:languageCode code="tr-TR"/>
                        <hl7:versionNumber value="1"/>
                        <hl7:recordTarget typeCode="RCT" contextControlCode="OP">
                            <hl7:patientRole classCode="PAT">
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<hl7:id root="2.16.840.1.113883.3.129.1.1.1">
extension="12345678901"/>
<hl7:patient classCode="PSN" determinerCode="INSTANCE">
<hl7:name>
<hl7:family>Topal</hl7:family>
<hl7:given>Ali</hl7:given>
</hl7:name>
<hl7:administrativeGenderCode code="1" displayName="Erkek" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.1.2.21" codeSystemName="Cinsiyet"/>
<hl7:birthTime value="19790311"/>
<hl7:raceCode code="1" displayName="TC" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.1.2.52" codeSystemName="Uyruk"/>
</hl7:patient>
</hl7:patientRole>
</hl7:recordTarget>
<hl7:author typeCode="AUT" contextControlCode="OP">
<hl7:time value="20080311151327"/>
<hl7:assignedAuthor classCode="ASSIGNED">
<hl7:id root="2.16.840.1.113883.3.129.1.1.1">
extension="26930056831"/>
</hl7:assignedAuthor>
</hl7:author>
<hl7:custodian typeCode="CST">
<hl7:assignedCustodian classCode="ASSIGNED">
<hl7:representedHealthcareOrganization classCode="ENT" determinerCode="INSTANCE">
<hl7:id root="2.16.840.1.113883.3.129.1.1.6">
extension="10121"/>
</hl7:representedHealthcareOrganization>
</hl7:assignedCustodian>
</hl7:custodian>
<hl7:primaryInformationRecipient typeCode="PRCP">
<hl7:recipient classCode="ASSIGNED">
<hl7:representedMinistryOfHealth classCode="ORG" determinerCode="INSTANCE">
<hl7:id root="2.16.840.1.113883.3.129.1.1.6">
extension="5881"/>
</hl7:representedMinistryOfHealth>
</hl7:recipient>
</hl7:primaryInformationRecipient>
<hl7:component typeCode="COMP" contextConductionInd="true">
<hl7:structuredBody classCode="DOCBODY" moodCode="EVN">
<hl7:component2 typeCode="COMP" contextConductionInd="true">
<hl7:dischargeDataset classCode="DOCSECT" moodCode="EVN">
<hl7:id root="2.16.840.1.113883.3.129.2.1.4" extension="3F2504E0-4F89-11D3-9A0C-0305E82C3301"/>
<hl7:code code="CIKIS" displayName="Çıkış Verisi" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.2.2" codeSystemName="Veriseti"/>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<hl7:text> Okunabilir bilgi</hl7:text>
<hl7:component2 typeCode="COMP">
contextConductionInd="true">
<hl7:dischargeSection>
classCode="DOCSECT" moodCode="EVN">
<hl7:id>
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C3301"/>
<hl7:code code="CIKIS">
displayName="Çıkış Verisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
<hl7:text>Component2:>
Okunabilir bilgi</hl7:text>
<hl7:component>
typeCode="COMP" contextConductionInd="true">
<hl7:discharge>
classCode="ACT" moodCode="EVN">
<hl7:code code="2" displayName="Taburcu (Yatan Hasta)" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.1.2.9" codeSystemName="Çıkış Şekli"/>
<hl7:effectiveTime value="200803111642"/>
</hl7:discharge>
</hl7:component>
</hl7:dischargeSection>
</hl7:component2>
</hl7:dischargeDataset>
</hl7:component2>
<hl7:component3 typeCode="COMP">
contextConductionInd="true">
<hl7:examinationDataset classCode="DOCSECT">
moodCode="EVN">
<hl7:id>
root="2.16.840.1.113883.3.129.2.1.4" extension="3F2504E0-4F89-11D3-9A0C-0305E82C3301"/>
<hl7:code code="MUAYENE">
displayName="Muayene Veriseti" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.2.2.2"
codeSystemName="Veriseti"/>
<hl7:text>Muayene Veriseti</hl7:text>
<hl7:author typeCode="AUT">
contextControlCode="OP">
<hl7:doctor>
classCode="ASSIGNED">
<hl7:id>
root="2.16.840.1.113883.3.129.1.1.1" extension="26930056831"/>
<hl7:doctor>
</hl7:author>
<hl7:component1 typeCode="COMP">
contextConductionInd="true">
<hl7:examProtocolNoSection>
classCode="DOCSECT" moodCode="EVN">
<hl7:id>
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C3301"/>
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<hl7:code
code="PROTOKOLNO" displayName="Protokol No Bilgisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
<hl7:text>Protokol No
Bilgisi</hl7:text>
<hl7:component
typeCode="COMP" contextConductionInd="true">

<hl7:examProtocolNo classCode="ACT" moodCode="EVN">
<hl7:id
root="2.16.840.1.113883.3.129.1.1.4" extension="123456"/>
<hl7:examProtocolNoSection>
</hl7:component>
<hl7:component7 typeCode="COMP"
contextConductionInd="true">
<hl7:procedureSection
classCode="DOCSECT" moodCode="EVN">
<hl7:id
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
<hl7:code
code="MUDAHALE" displayName="Müdehale Verisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
<hl7:text>MUDAHALE</hl7:text>
<hl7:component
typeCode="COMP" contextConductionInd="true">
<hl7:procedure
classCode="PROC" moodCode="EVN">
<hl7:code code="300070" displayName="MEMBRAN FİLTRELİ KAN POMPONENTİ AYRİŞTIRMA"
codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.1.2.2" codeSystemName="BÜT"/>
<hl7:procedure>
</hl7:component>
<hl7:procedureSection>
</hl7:component7>
<hl7:component8 typeCode="COMP"
contextConductionInd="true">
<hl7:diagnosisSection
classCode="DOCSECT" moodCode="EVN">
<hl7:id
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
<hl7:code code="TANI"
displayName="Tanı Verisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
<hl7:text>Tanı</hl7:text>
<hl7:component
typeCode="COMP" contextConductionInd="true">
<hl7:diagnosis
classCode="OBS" moodCode="EVN">
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<hl7:code code="ANATANI" displayName="Ana Tanı" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.6" codeSystemName="Tanı Tipi"/>

<hl7:value code="M800-M998" displayName="Neoplazmların Morfolojisi" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.6.3" codeSystemName="ICD-10"/>
                                         </hl7:diagnosis>
                                         </hl7:component>
                                         </hl7:diagnosisSection>
                                         </hl7:component8>
                                         <hl7:component9 typeCode="COMP">

contextConductionInd="true">
                                         <hl7:examinationSection

classCode="DOCSECT" moodCode="EVN">
                                         <hl7:id
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
                                         <hl7:code
code="MUAYENE" displayName="Muayene Verisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
                                         <hl7:text>Muayene
                                         <hl7:component
                                         typeCode="COMP" contextConductionInd="true">
                                         <hl7:encounter

classCode="ENC" moodCode="EVN">
                                         <hl7:effectiveTime>
                                         <hl7:low value="200803110900"></hl7:low>
                                         <hl7:high value="200803111000"></hl7:high>
                                         </hl7:effectiveTime>
                                         <hl7:location typeCode="LOC">
                                         <hl7:clinic classCode="SDLOC">
                                         <hl7:code code="2" displayName="Aile Hekimliği" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.1.2.1" codeSystemName="Klinikler"/>
                                         </hl7:clinic>
                                         </hl7:location>
                                         </hl7:encounter>
                                         </hl7:component>
                                         </hl7:examinationSection>
                                         </hl7:component9>
                                         </hl7:examinationDataset>
                                         </hl7:component3>
                                         <hl7:component4 typeCode="COMP">

contextConductionInd="true">
```

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
<hl7:receptionDataset classCode="DOCSECT"
moodCode="EVN">
    <hl7:id
root="2.16.840.1.113883.3.129.2.1.4" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
        <hl7:code code="KABUL"
displayName="Kabul Veriseti" codeSystemVersion="1.0" codeSystem="2.16.840.1.113883.3.129.2.2.2"
codeSystemName="Veriseti"/>
            <hl7:text>Kabul Veriseti</hl7:text>
            <hl7:component3 typeCode="COMP"
contextConductionInd="true">
                <hl7:registrationSection
classCode="DOCSECT" moodCode="EVN">
                    <hl7:id
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
                        <hl7:code code="KABUL"
displayName="Kabul Verisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
                            <hl7:text>Kabul</hl7:text>
                            <hl7:component
typeCode="COMP" contextConductionInd="true">
                                <hl7:registration
classCode="ACT" moodCode="EVN">
                                    <hl7:code code="3" displayName="Diğer" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.1.2.7" codeSystemName="Kabul Şekli"/>
                                    <hl7:effectiveTime value="200803101500"/>
                                    </hl7:registration>
                                    </hl7:component>
                                </hl7:registrationSection>
                            </hl7:component3>
                            <hl7:component4 typeCode="COMP"
contextConductionInd="true">
                                <hl7:caseTypeSection
classCode="DOCSECT" moodCode="EVN">
                                    <hl7:id
root="2.16.840.1.113883.3.129.2.1.5" extension="3F2504E0-4F89-11D3-9A0C-0305E82C330"/>
                                    <hl7:code
code="VAKATURU" displayName="Vaka Türü Bilgisinin Olduğu Bölüm" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.2.2.3" codeSystemName="Veri Kısmı"/>
                                    <hl7:text>Vaka
Turu</hl7:text>
                                    <hl7:component
typeCode="COMP" contextConductionInd="true">
                                        <hl7:caseType
classCode="ACT" moodCode="EVN">
                                            <hl7:code code="1" displayName="Normal" codeSystemVersion="1.0"
codeSystem="2.16.840.1.113883.3.129.1.2.53" codeSystemName="Vaka Türü"/>
                                            </hl7:caseType>

```

91

T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
</hl7:component>
</hl7:caseTypeSection>
</hl7:component4>
</hl7:receptionDataset>
</hl7:component4>
</hl7:structuredBody>
</hl7:component>
</hl7:examination>
</hl7:subject>
</hl7:controlActEvent>
</hl7:MCCI_IN000001TR01>
```

5 Sağlık-NET Web Servisleri Örnek .NET C# İstemci Uygulaması

Bu dokümanın amacı Sağlık Bakanlığı Bilgi İşlem Daire Başkanlığı tarafından yürütüçülüğü gerçekleştirilen Ulusal Sağlık Bilgi Sistemi ile iletişime geçecek örnek bir istemci yazılımını .NET C# programlama dili kullanarak geliştirmektir.

5.1 Gereksinimler

İstemci geliştirilirken aşağıdaki araçlar kullanılmıştır:

1. Microsoft Visual Studio 2005³⁴ (VS2005)
2. Microsoft Web Service Enhancements 3.0³⁵ (WSE 3.0)

Öncelikle Visual Studio 2005 kurulumu yapılmalıdır. Daha sonra Visual Studio en az bir kere çalıştırıldıkten sonra Web Service Enhancements 3.0 kurulabilir. WSE 3.0 kurulumun amacı USBS'de güvenlik için kullanılan WS-Security UsernameToken Profiline uygun mesajların istemci tarafından gönderimini sağlamaktır.

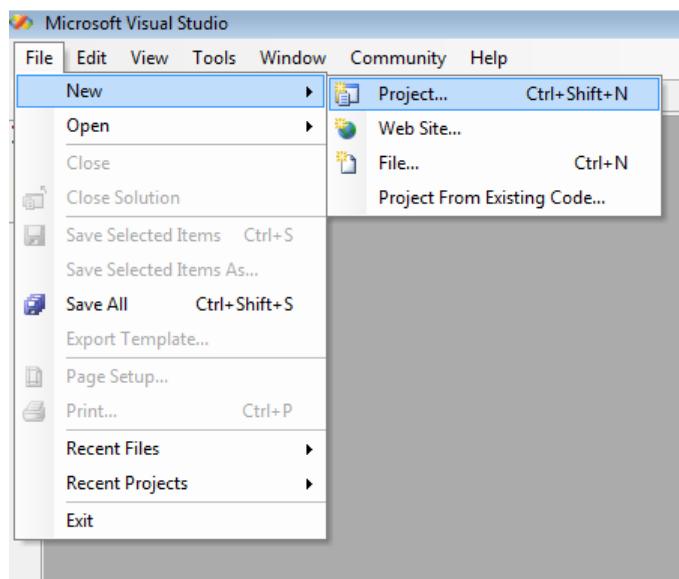
5.2 İstemci Kodu Üretimi

İstemci geliştirilirken aşağıdaki adımların gerçekleştirilmesi gerekmektedir.

1. VS2005'i başlatınız.
2. Şekil 66'te görüldüğü gibi yeni bir proje yaratınız.

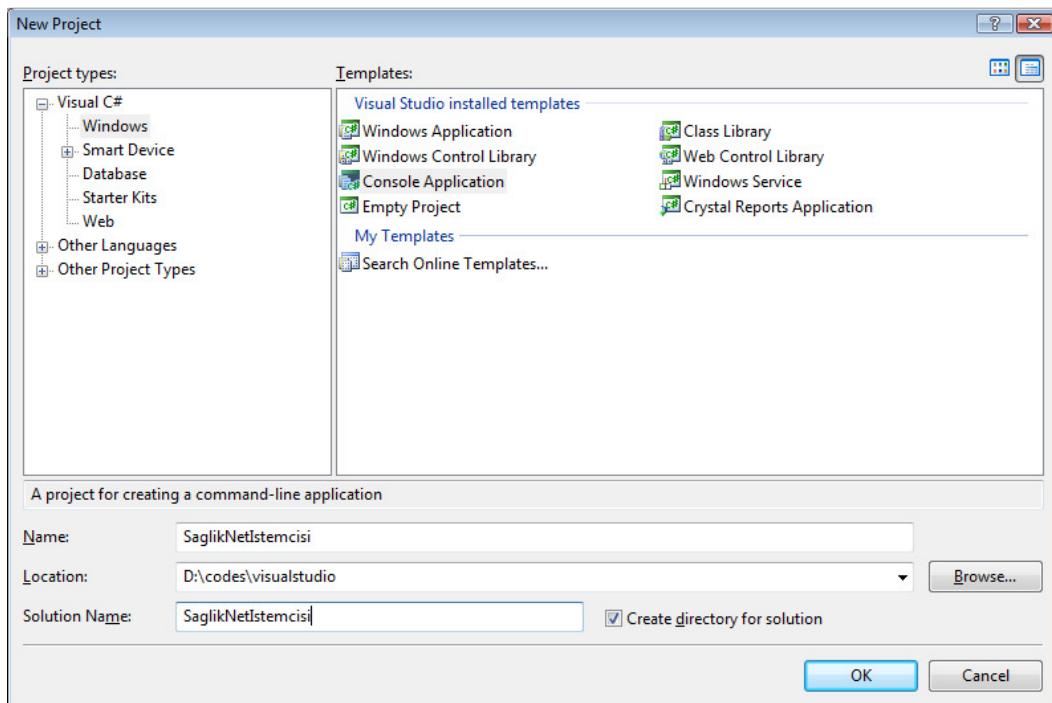
³⁴ <http://www.microsoft.com/turkiye/net/default.mspx>

³⁵ <http://www.microsoft.com/downloads/details.aspx?familyid=018a09fd-3a74-43c5-8ec1-8d789091255d&displaylang=en>



Şekil 66 Yeni Proje Yaratılması

3. Projenin tipini “Console Application” olarak seçiniz ve bir isim veriniz. Şekil 67’te görüldüğü üzere projenin ismi “SaglikNetistemci” olarak belirlenmiştir.

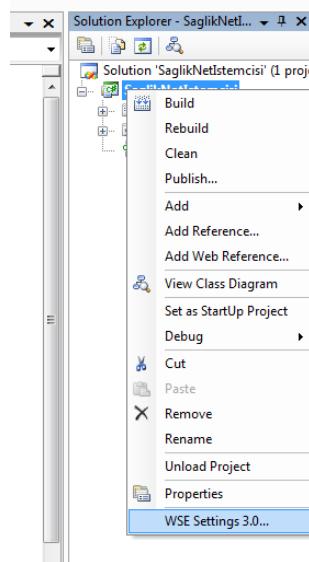


Şekil 67 Proje Tipinin Belirlenmesi

4. Daha sonra geliştirilen istemcinin “WSE 3.0 İstemci Konfigürasyonunun” yapılması gerekmektedir. Bu işlem aşağıdaki şekilde yapılabilir

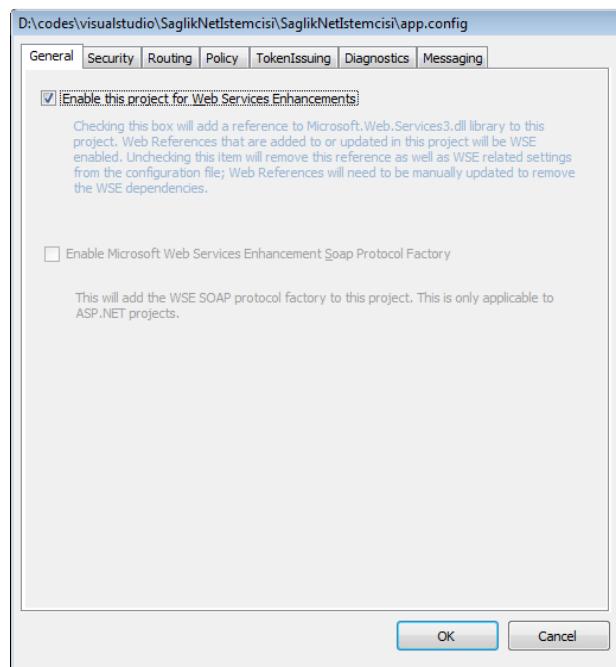
T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

- a. Öncelikle projenin WSE 3.0 kullanması yönünde etkinleştirilmelidir. Bunun için Şekil 68'teki gibi projeyi sağ tıklayıp, "WSE Settings 3.0..." seçiniz.



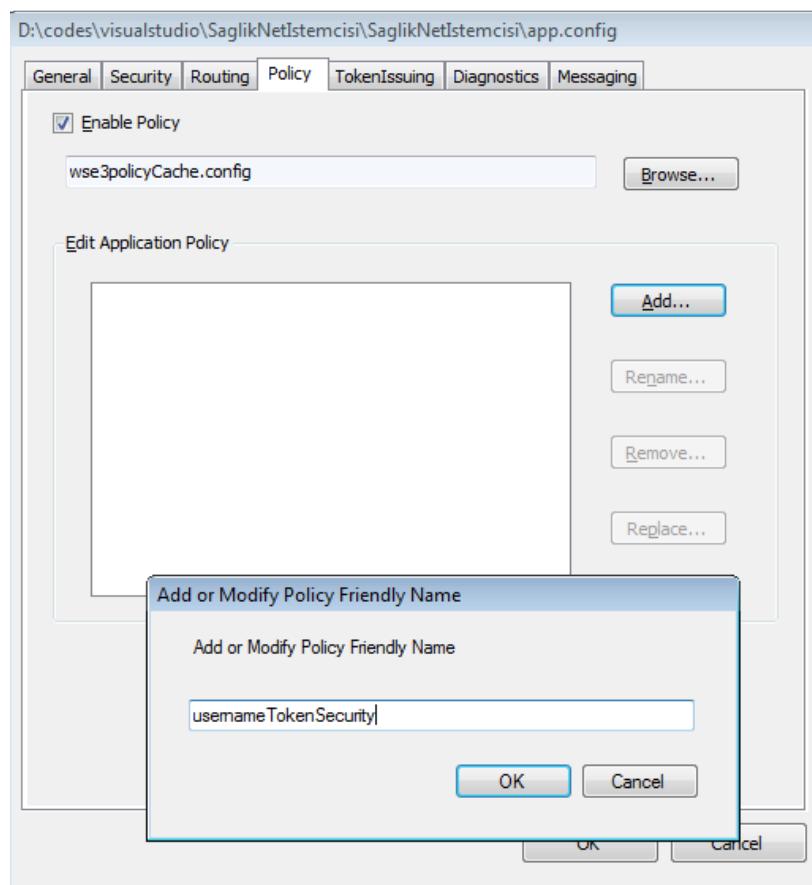
Şekil 68 WSE 3.0 Konfigürasyonu

- b. Daha sonra çıkan pencerenin General sekmesindeki "Enable this project for Web Service Enhancements" seçeneğini tıklayınız.



Şekil 69 WSE 3.0 Etkinleştirilmesi

- c. Şekil 70'te görüldüğü gibi daha sonra Policy sekmesini tıklayın. Burada "Enable Policy" seçeneğini seçiniz ve "Edit Application Policy" grubunun altındaki "Add..." düğmesine basarak çıkan pencerede policy dosyasına bir isim veriniz. Örnekte "usernameTokenSecurity" ismi kullanılmıştır.



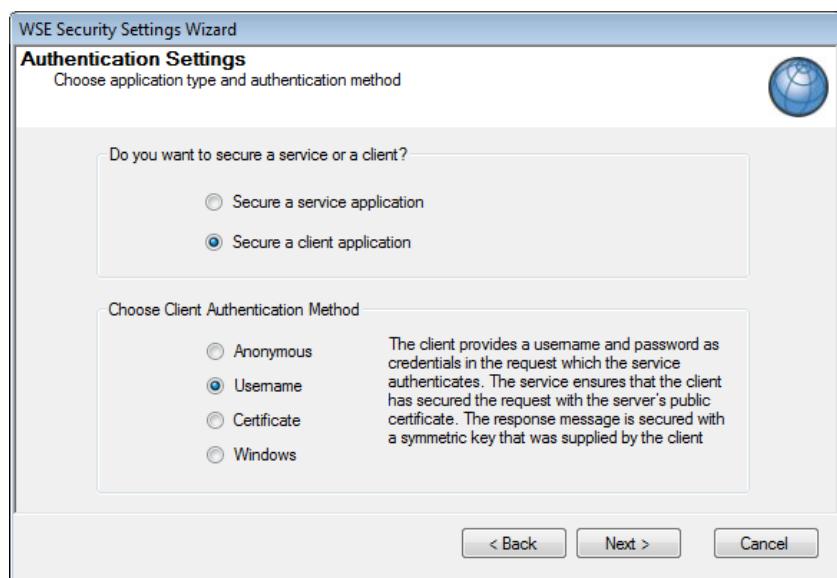
Şekil 70 WSE 3.0 Policy Ekleme

- d. Şekil 70'te görülen "Add or Modify Policy Friendly Name" penceresindeki OK tuşu basıldıktan sonra çıkan Sihirbaz (Wizard) penceresinde önce Next tuşuna basınız.



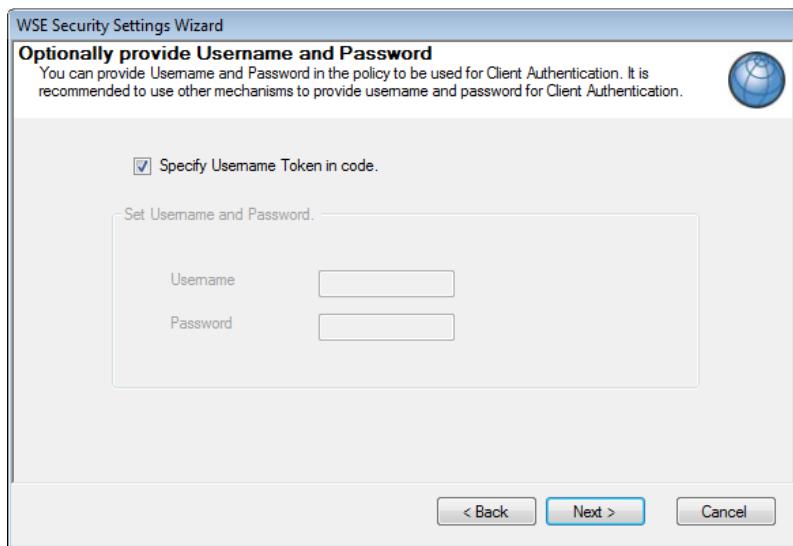
Şekil 71 Policy Konfigürasyon Sihirbazı

- e. Daha sonra Şekil 72'te görüldüğü üzere "Secure a client application" ve "Username" seçeneklerini tıklayıp Next tuşuna basınız.



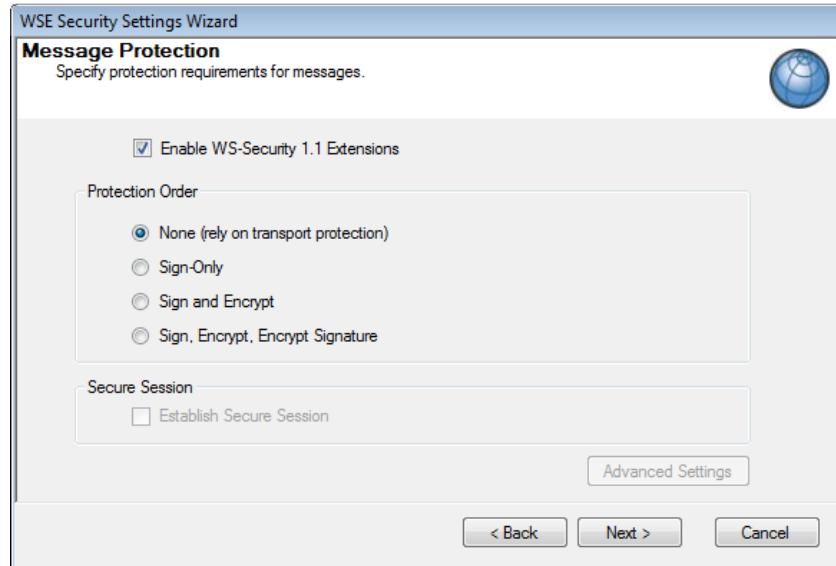
Şekil 72 Kimlik Belileme Ayarları

- f. Şekil 73'te görüldüğü gibi "Specify Username Token in code" seçeneğini tıklatıp Next tuşuna basınız.



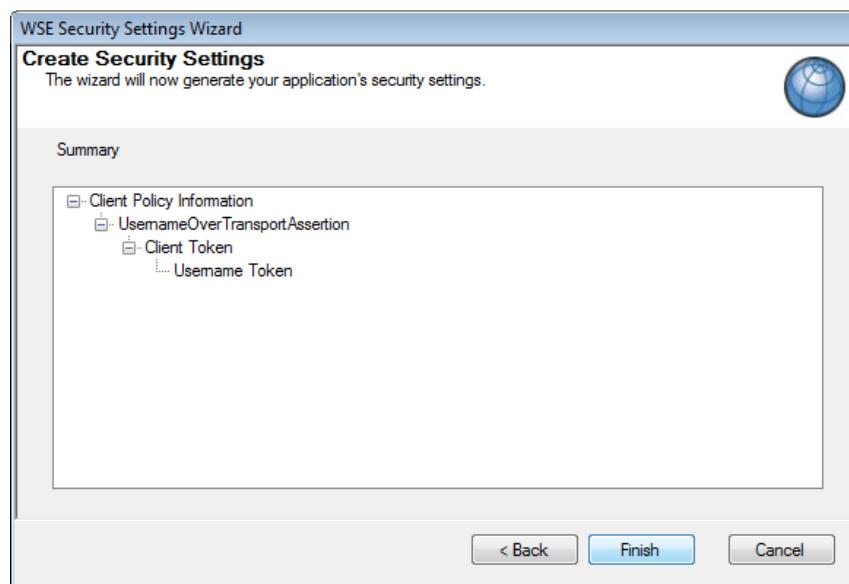
Şekil 73 Seçimli Kullanıcı adı ve Şifre Ayarları

- g. Daha sonraki adımda aşağıdaki Şekil 74'te görüldüğü gibi "None (rely on transport security protection)" ve "Enable WS-Security 1.1 Extensions" seçeneklerini seçip Next tuşuna basınız.



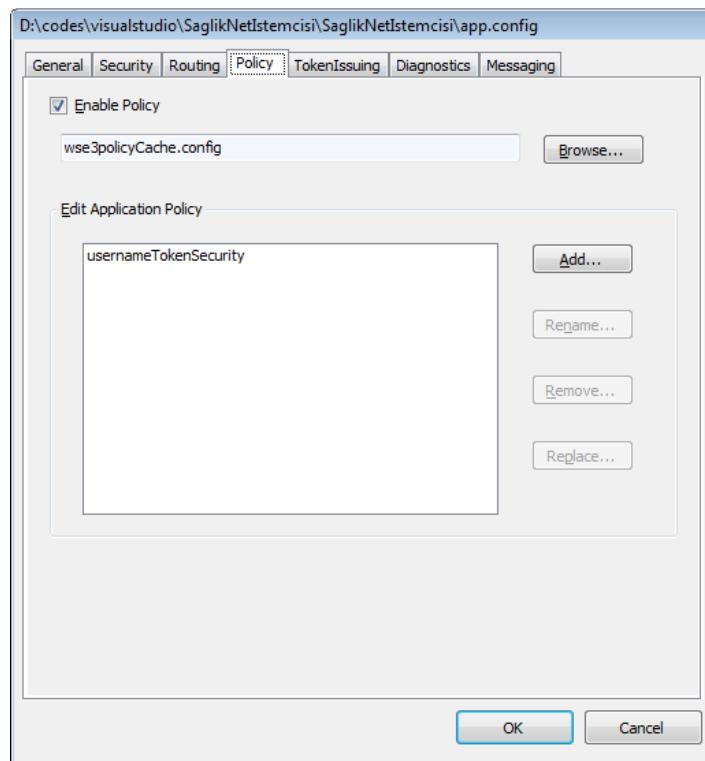
Şekil 74 Mesaj Koruma Ayarları

- h. Sihirbazın en son adımı Şekil 75'te gösterilmektedir. Bu adımda Finish tuşuna basınız.



Şekil 75 Güvenlik Ayarlarının Yaratılması

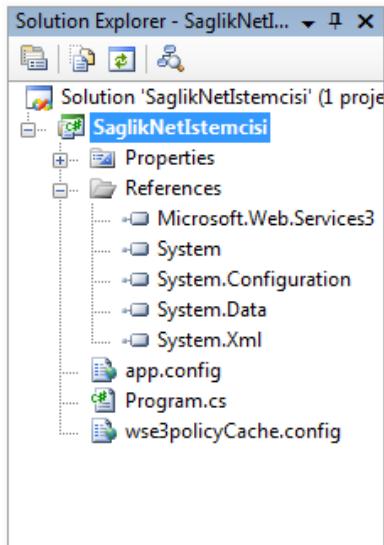
5. Sihirbazın kapanmasından sonra Policy sekmesi Şekil 76'te görüldüğü gibi olacaktır. Bu adımda istemcinin konfigürasyonu tamamlanmış olup OK tuşuna basılabilir.



Şekil 76 Policy'nin Yaratılması

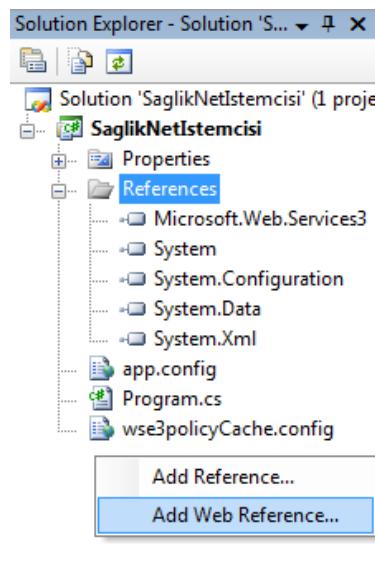
T.C. Sağlık Bakanlığı, BiDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

6. Bu adımlar sonucu projede yeni iki dosya yaratılmıştır. Bunlar app.config ve WSE 3.0 konfigürasyonunun durduğu wse3policyCache.config'dır.



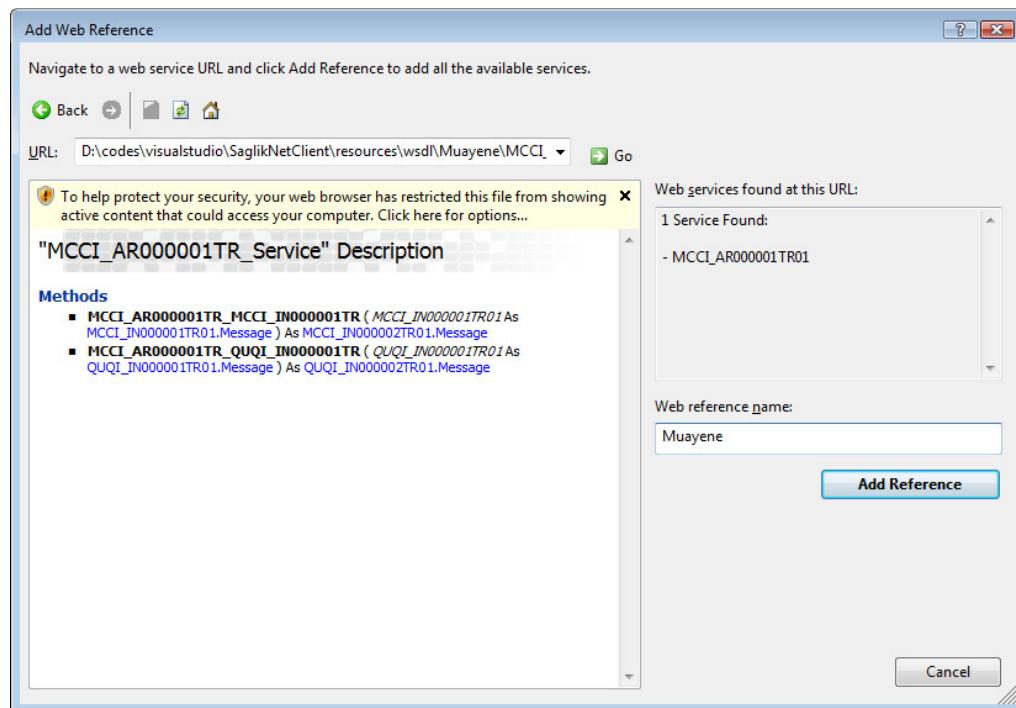
Şekil 77 Yeni Konfigürasyon Dosyaları

7. Bu adımdan sonra herbir USBS Web Servisi, istemci projesine Web Referansı olarak eklenebilir. Dokümda Muayene Web Servisi kullanılmıştır. Web Referansı'nı eklemek için "Solution Explorer" in herhangi bir yerini sağ tıklayarak "Add Web Reference" seçeneğini tıklayınız.



Şekil 78 Web Referanslarının Eklenmesi

8. Şekil 79'teki gibi çıkan pencerede Muayene Web Servisinin WSDL dokümanın dosya yerini veya URL'ini giriniz ve bir isim vererek (mesela Muayene) "Add Reference" tuşuna basınız.



Şekil 79 Muayene Web Servisi Referasının Eklenmesi

9. Bu adım itibarı ile Muayene Web Servisinin gerekli stub kaynak kodları oluşturulmuştur. Bundan sonra yapılacak işlemler istemcimizi bu yaratılan kodları kullanmasına yönelik bir üst seviyedeki kodların yazılmasıdır. Bu üst seviye diye nitelendirilen bölümde HL7 v3 mesajı oluşturulacak gerekli UUID değerleri set edilecek ve Muayene Web Servisinin hangi operasyonunun (veri bildirim mi yoksa sorgulama mı?) çağrırlaçığı belirlenecektir.
10. Geliştirilen örnek istemci gönderilecek olan HL7 v3 mesajını bir dosyadan okuyup gerekli obje modelini oluşturmaktadır. Yazılımcılar mesaj obje modelini eğer bir dosyadan okumadan yaratacaklarsa bir sonraki adıma geçebilirler. Mesaj obje modelini bir XML dosyasından yaratmak için Reference.cs dosyasında aşağıdaki değişiklikleri yapmalıdır:
 - a. MCCI_IN000001TR01Message sınıfının başına aşağıdaki satırı ekleyiniz.

```
[System.Xml.Serialization.XmlRootAttribute("MCCI_IN000001TR01", Namespace = "urn:hl7-org:v3", IsNullable = false)]
```
 - b. MCCI_IN000002TR01Message sınıfının başına aşağıdaki satırı ekleyiniz.

```
[System.Xml.Serialization.XmlRootAttribute("MCCI_IN000002TR01", Namespace = "urn:hl7-org:v3", IsNullable = false)]
```
 - c. QUQI_IN000001TR01Message sınıfının başına aşağıdaki satırı ekleyiniz.

T.C. Sağlık Bakanlığı, BİDB, Yeni Başlayanlar İçin HL7 Kılavuzu (Sürüm 2.0)

```
[System.Xml.Serialization.XmlRootAttribute("QUQI_IN000001TR01", Namespace = "urn:hl7-org:v3", IsNullable = false)]
```

- d. QUQI_IN000002TR01Message sınıfının başına aşağıdaki satırı ekleyiniz.

```
[System.Xml.Serialization.XmlRootAttribute("QUQI_IN000002TR01", Namespace = "urn:hl7-org:v3", IsNullable = false)]
```

11. Bu üst seviye için aşağıdaki Tablo 1'teki kaynak kod kullanılır. Kaynak kodda da görüldüğü üzere Main fonksyonu “isDataTransmission” değişkeninin değerine göre ya “firstOperation” fonksyonunu yada “secondOperation” fonksyonunu çağırmaktadır.

Tablo 1 Program.cs Kaynak Kodu

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Xml.Serialization;
using System.Xml;
using System.IO;
using SaglikNetIstemcisi.Muayene;
using Microsoft.Web.Services3;
using Microsoft.Web.Services3.Design;
using Microsoft.Web.Services3.Security.Tokens;

namespace SaglikNetIstemcisi
{
    class Program
    {
        static void Main(string[] args)
        {
            String dataTransmissionFilePath =
"D:\\codes\\visualstudio\\saglikNetClient\\resources\\instances\\Muayne.xml";
            String queryFilePath =
"D:\\codes\\visualstudio\\saglikNetClient\\resources\\instances\\query\\sorgulama.xml";
            bool isDataTransmission = false;
            if (isDataTransmission)
            {
                String guid = Guid.NewGuid().ToString();
                String code = firstOperation(dataTransmissionFilePath,
guid);
                Console.WriteLine(guid);
                Console.WriteLine(code);
            }
            else
            {
                String code = secondOperation(queryFilePath, "4a561ce8-
2856-4c50-9650-5338b9265ae2");
                Console.WriteLine(code);
            }
        }
    }
}
```

```
    static String firstOperation(String filePath, String UUID)
    {
        XmlSerializer serializer = new
XmlSerializer(typeof(MCCI_IN000001TR01Message));
        FileStream fs = new FileStream(filePath,
 FileMode.OpenOrCreate);
        TextReader reader = new StreamReader(fs);

        MCCI_IN000001TR01Message input =
(MCCI_IN000001TR01Message)serializer.Deserialize(reader);

        input.controlActEvent.subject.examination.id.extension =
UUID;
        MCCI_AR000001TR_ServiceWse stub = new
MCCI_AR000001TR_ServiceWse();

        UsernameToken token = new
UsernameToken("SAGLIK_BAKANLIGINDAN_ALINAN_USERNAME",
"SAGLIK_BAKANLIGINDAN_ALINAN_PASSWORD",
>PasswordOption.SendPlainText);

        stub.SetClientCredential(token);
        stub.SetPolicy("usernameTokenSecurity");

        MCCI_IN000002TR01Message response =
stub.MCCI_AR000001TR_MCCI_IN000001TR(input);
        return response.acknowledgement.typeCode.code;
    }

    static String secondOperation(String filePath, String UUID)
    {
        XmlSerializer serializer = new
XmlSerializer(typeof(QUQI_IN000001TR01Message));
        FileStream fs = new FileStream(filePath,
 FileMode.OpenOrCreate);
        TextReader reader = new StreamReader(fs);

        QUQI_IN000001TR01Message input =
(QUQI_IN000001TR01Message)serializer.Deserialize(reader);

        input.controlActEvent.queryByParameter.clinicalDocumentId.value.extension =
UUID;

        MCCI_AR000001TR_ServiceWse stub = new
MCCI_AR000001TR_ServiceWse();

        UsernameToken token = new
UsernameToken("SAGLIK_BAKANLIGINDAN_ALINAN_USERNAME",
"SAGLIK_BAKANLIGINDAN_ALINAN_PASSWORD",
>PasswordOption.SendPlainText);
```

```
        stub.SetClientCredential(token);
        stub.SetPolicy("usernameTokenSecurity");

        QUOI_IN000002TR01Message response =
stub.MCCI_AR000001TR_QUOI_IN000001TR(input);
        String result = response.acknowledgement.typeCode.code +
" :" + response.controlActEvent.queryAck.queryResponseCode.code;
        return result;

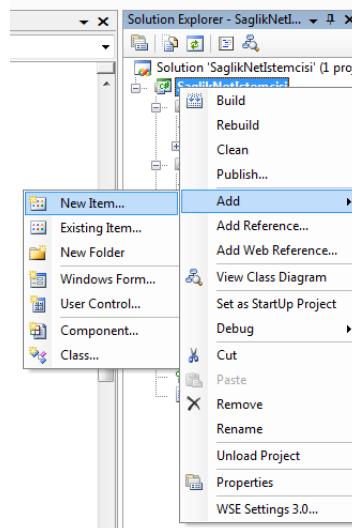
    }
}
```

12. Kaynak kod bu durumda çalıştırılırsa USBS Web servisinden “mustUnderstand” özelliği ile ilgili bir hata olacaktır. Bunun sebebi giden SOAP mesajının “Security Header” bölümündeki “mustUnderstand” özelliğinin değerinin 1 olarak set edilmesidir. USBS Web Servislerini başarılı bir şekilde çağırmak için bu değer 0 olarak set edilmeli veya mustUnderstand özelliği silinmelidir. Bunun için ilk akla gelen çözüm aşağıdaki satırı kullanarak bu değeri sıfırlamaktır. Ama VS2005 bu değeri göz ardı edip ne olursa olsun mustUnderstand özelliği için 1 değeri göndermektedir.

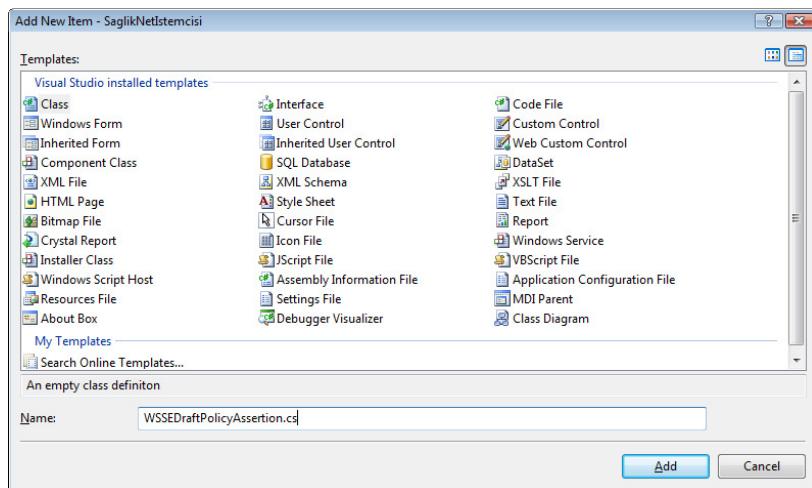
```
stub.RequestSoapContext.Security.EncodedMustUnderstand = "0";
```

Bu sorunu çözmek için WSE 3.0'ın SOAPFilter mekanizması kullanılmalıdır. SOAPFilter mekanizması istemciden gönderilecek olan SOAP mesajına ulaşım gereklili deşikliklerin yapılabilmesini sağlamaktadır. Bunun için gerçekleştirilecek adımlar şu şekildedir.

- a. Önce projemize WSSEDraftPolicyAssertion.cs isminde yeni bir sınıf ekleyiniz. Bunun için sırasıyla Şekil 80 ve Şekil 81'te gösterilen adımlar izlenebilir.



Şekil 80 Yeni Sınıf Ekleme



Şekil 81 Eklenecek Sınıfın İsmini Belirleme

- Bu sınıf genel olarak gönderilecek olan SOAP mesajına ulaşır mustUnderstand özelliğini silmektedir. Bu sınıfın içeriği aşağıdaki Tablo 2'teki gibidir.

Tablo 2 WSSEDraftPolicyAssertion.cs Kaynak Kodu

```
using System;
using Microsoft.Web.Services3.Design;
using Microsoft.Web.Services3;
using System.Xml;
using System.Xml.XPath;
using System.Collections.Generic;

namespace SaglikNetIstemcisi
{
    public class WSSEDraftPolicyAssertion : PolicyAssertion
    {

        public override Microsoft.Web.Services3.SoapFilter
CreateServiceInputFilter(FilterCreationContext context)
        {
            return null;
        }

        public override Microsoft.Web.Services3.SoapFilter
CreateServiceOutputFilter(FilterCreationContext context)
        {
            return null;
        }

        public override Microsoft.Web.Services3.SoapFilter
CreateClientInputFilter(FilterCreationContext context)
        {
            return null;
        }
    }
}
```

```
    public override Microsoft.Web.Services3.SoapFilter
CreateClientOutputFilter(FilterCreationContext context)
{
    return new WSSEDraftClientOutputFilter();
}

    public override void ReadXml(XmlReader reader,
IDictionary<string, Type> extensions)
{
    if (reader == null)
        throw new ArgumentNullException("reader");
    if (extensions == null)
        throw new ArgumentNullException("extensions");

    bool isEmpty = reader.IsEmptyElement;

    reader.ReadStartElement("WSSEDraftAssertion");
    if (!isEmpty)
    {
        reader.Skip();
    }
}

    public override IEnumerable<KeyValuePair<string, Type>>
GetExtensions()
{
    return new KeyValuePair<string, Type>[] { new
KeyValuePair<string, Type>("WSSEDraftAssertion", this.GetType()) };
}

    class WSSEDraftClientOutputFilter : SoapFilter
{
    internal WSSEDraftClientOutputFilter()
    {

    }

    public override SoapFilterResult
ProcessMessage(SoapEnvelope envelope)
{
    XmlNodeList nodeList =
envelope.Header.GetElementsByTagName("Security",
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd");
    XmlNode node = nodeList.Item(0);
    XmlAttributeCollection attributes = node.Attributes;
    attributes.RemoveAll();
    return SoapFilterResult.Continue;
}
}
}
```

- c. Daha sonra yapılması gereken iş wse3policyCache.config dosyasının bu sınıfı çağırmasına yönelik değiştirilmesidir. Aşağıdaki Tablo 3'te gösterilen XML içeriği wse3policyCache.config dosyasının içine kopyalanabilir.

Tablo 3 wse3policyCache.config İçeriği

```
<policies
  xmlns="http://schemas.microsoft.com/wse/2005/06/policy">
  <extensions>
    <extension name="usernameOverTransportSecurity"
      type="Microsoft.Web.Services3.Design.UsernameOverTransportAssertion,
      Microsoft.Web.Services3, Version=3.0.0.0, Culture=neutral,
      PublicKeyToken=31bf3856ad364e35" />
    <extension name="requireActionHeader"
      type="Microsoft.Web.Services3.Design.RequireActionHeaderAssertion
      , Microsoft.Web.Services3, Version=3.0.0.0, Culture=neutral,
      PublicKeyToken=31bf3856ad364e35" />
    <extension name="WSSEDraftAssertion"
      type="SaglikNetIstemcisi.WSSEDraftPolicyAssertion,SaglikNetIstemicisi"/>
  </extensions>
  <policy name="usernameTokenSecurity">
    <usernameOverTransportSecurity />
    <requireActionHeader />
    <WSSEDraftAssertion />
  </policy>
</policies>
```

- d. Bu adımdan sonra örnek istemcimizin uygulaması bitmiş olup, proje derlenip çalıştırılabilir.