ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

BİÇİMSEL DİLLER VE OTOMATA

21 Nisan 2020

Doç. Dr. Ahmet Yazıcı

ISO-19450 OBJECT PROCESS MODELING(OPM); Opcat, xml

BAGGAGE HANDLING SYSTEM

152120171007 – Muzaffer Arda Uslu

152120171017 – Gökhan Samet Albayrak

152120171023 – Onur Akkepenek

152120171029 – İsmail Demircan

İçindekiler

1. Genel Giriş 3
   1.1. OPM 3
      1.1.1. OPM Tarihçe 3
      1.1.2. OPM'de Diyagram 4
      1.1.3. OPM'in Faydaları 4
   1.2. Neden Kavramsal Modelleme 4
   1.3. Dizayn ve Temel Ögeler 5
      1.3.1. Modelleme 5
   1.4. Links 8
      1.4.1. Structural Links (Yapısal Bağlantı) 8
      1.4.2. Procedural Links (Yordamsal Bağlantı) 9
      1.4.3. Event and condition 9
   1.5. OPM & SysML 9
   1.6. OPM & UML 9
   1.7. ISO-19450 9
   1.8. OPCAT 10
   1.9. XML 10
   1.10. OPM Kullanım Alanları 10
   1.11. Proje Örnekleri 11
2. Problem Özellikleri ve Çözüm Yaklaşımı 12
   2.1. Problem Çözümü 12
   2.2. Uygulama Yazılımı 17
3. Proje Ekibi Değerlendirmesi 23
   3.1.Grup Bilgisi 23
      3.1.1. Grup Koordinatörü ve Görev Paylaşımı 23
      3.1.2. Grup Üyelerinin Adam-Saat Süreleri 23
   3.2 Yapılan Toplantı Raporu 23
4. Sonuçlar 27
5. Kaynakça 29

1. Genel Giriş

Günümüzde çoğu mühendislik alanları farklı diller kullanılıyor:

- Makina Mühendisliği -> Makine Çizimleri
- İnşaat Mühendisliği -> Kendine özgü çizimler
- Bilgisayar Mühendisliği -> UML
- Elektrik ve Elektronik Mühendisliği -> Kendine özgü çizimler
- Peki Sistem Mühendisleri hangi dili kullanıyor?

Sistem mühendisleri sistem mimarilerini ve tasarımını kavramsal olarak iletmek için grafik biçimsel dil kullanıyorlar.

2 dile sahiptirler:

-SysML (Systems Modelling Language)

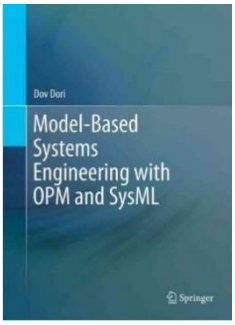
-**OPM** (Object Process Methodology)

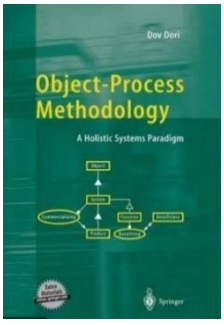
1.1. OPM

- OPM bilgiyi yakalama ve sistemi tasarlamada kullanılan kavramsal modelleme dilidir. Durum bilgisi olan objelerin minimal evrensel ontolojisine ve bu objeleri dönüştüren işlemlere dayanır.
- OPM, çok çeşitli alanlarda yapay ve doğal sistemlerin işlevini, yapısını ve davranışını resmi olarak belirlemek için kullanılabilir.

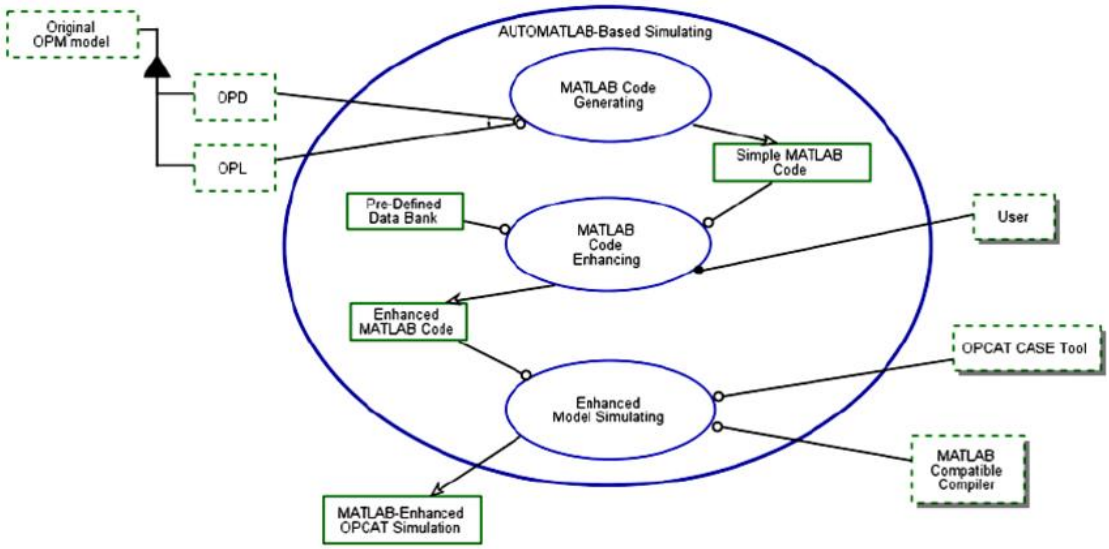
1.1.1. OPM Tarihçe

- Dov Dori tarafından tasarlandı ve geliştirildi.
- İlk fikirler 1995 yılında ortaya çıktı.
- İlk kitap 2002 yılında yayımlandı (Object-Process Methodology: a Holistic Systems Paradigm).
- OPM o zamandan beri Anlambilim ağından savunma ve moleküler biyolojiye kadar birçok alanda uygulanmıştır.
- SysML dilini de kapsayan ikinci kitap ise 2016 yılında yayımlandı.
- Ağustos 2014'te ISO TC184 / SC5'in beş yıllık çalışmasının ardından ISO, OPM'yi ISO / PAS 19450 olarak kabul etti.
- SysML dili OPM den 8 yıl önce standart kabul edildiğinden ve birinci sınıf satıcılar tarafından desteklendiği için daha çok benimseniyor fakat OPM de akademi ve endüstri alanında hızla kabul görülen bir dildir.





1.1.2. OPM'de Diyagram

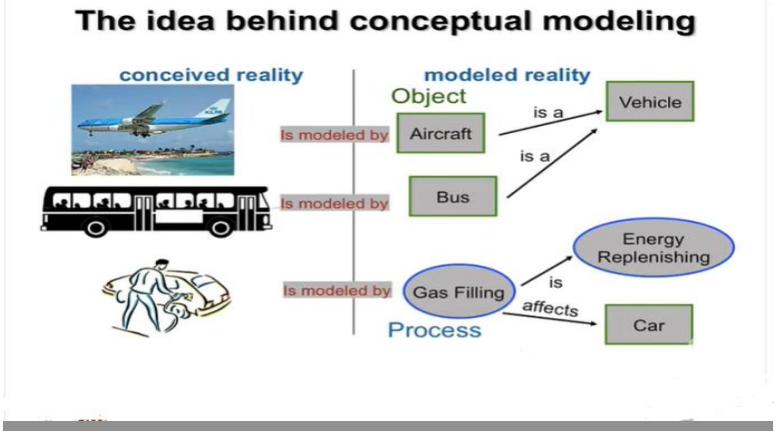


1.1.3. OPM'in Faydaları

- OPM'in sadece bir çeşit diyagramı vardır; OPD. Bu nedenle basittir ve temelinde öğrenilmesi kolaydır.
- Karmaşık senaryoları modellemek için şaşırtıcı derecede basit bir çerçeve sunar.

1.2. Neden Kavramsal Modelleme

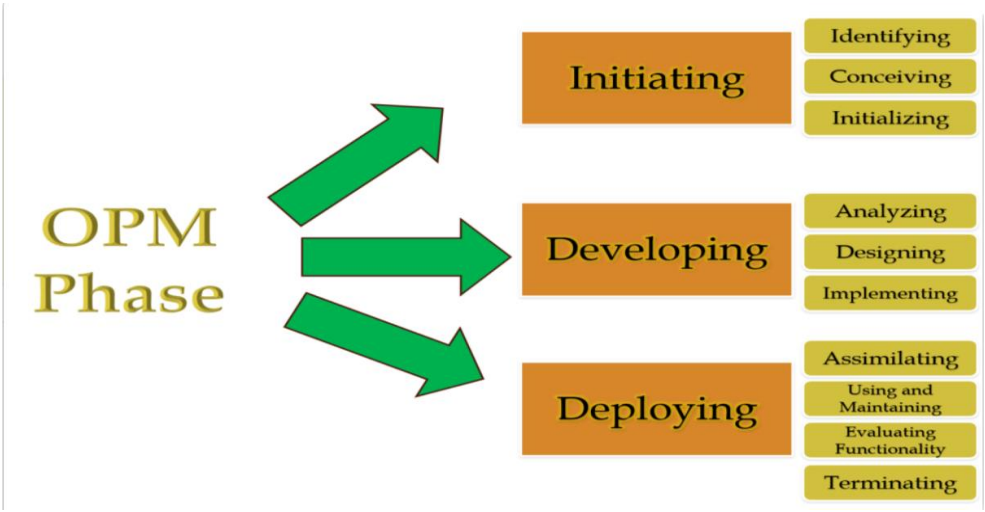
- Sistemin zihinsel bir resmini oluşturur.
- Sözsüz araçlarla tasarım sistemleri oluşturur.
- Kavramları başkalarına iletmeyi sağlar. (İnsanların sahip olduğu bilgileri açıkça ve net bir şekilde aktarmayı sağlar.)
- Sözsüz bilgiyi belirgin bilgiye dönüştürür. (Birçok insan bilgiye sahip fakat tam olarak açıklama durumuna sahip değiller.)
- Görsel ve metinsel şekilcilik anlayışı ile durumu izah eder.



1.3. Dizayn ve Temel Ögeler

1.3.1. Modelleme

- Nesne Süreç Metodolojisi (OPM) herhangi bir sistemde bulunan iki yönü birleştiren (yapısı ve davranışı) bir sistem modelleme paradigmasıdır.
- OPD (Object-Process Diagrams) görsel/grafiksel tabanlıdır.
- OPL (Object-Process Language) kısmı ise İngilizcenin bir alt kümesindeki karşılık gelen cümleler kümesini içeren metinsel tabanlıdır.
- OPD ve OPL oluşturmak için OPCAT adlı patentli bir yazılım paketi ücretsiz olarak mevcuttur.



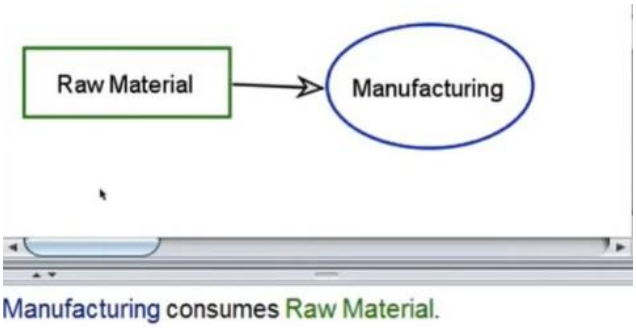
Yapı: Nesneler ve aralarındaki yapısal ilişkiler aracılığıyla temsil edilir

(Bütün-kısım ve “is -a” ilişkisi gibi).

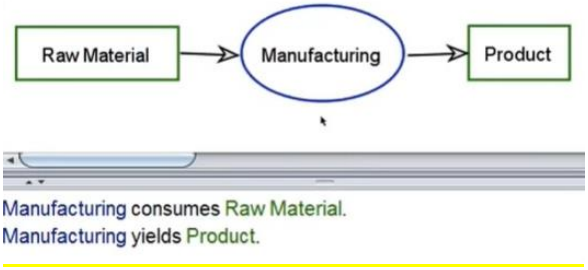
Davranış: Süreçler ve nesneleri nasıl dönüştürdükleri ile temsil edilir.

Yukarda da görüldüğü gibi obje ve süreç bu simgelerle temsil edilir.

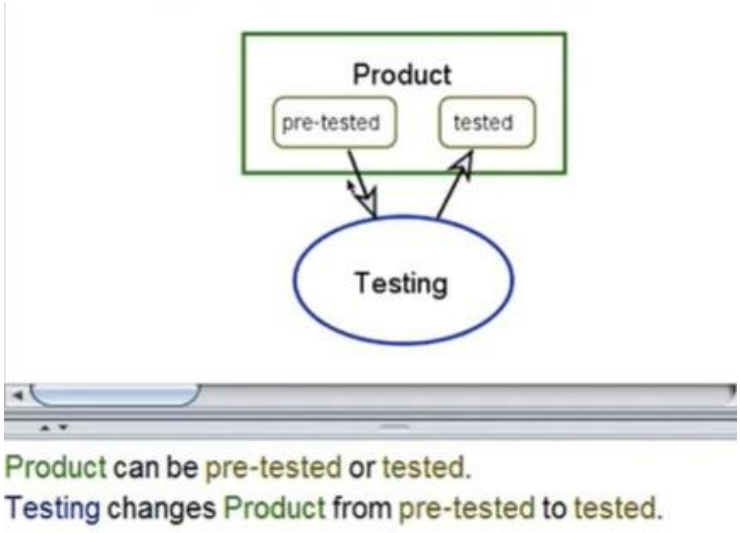
Dönüştürmekten kastımız nesneleri nasıl oluşturdukları veya tükettikleri veya bir nesnenin durumlarını nasıl değiştirdikleri.



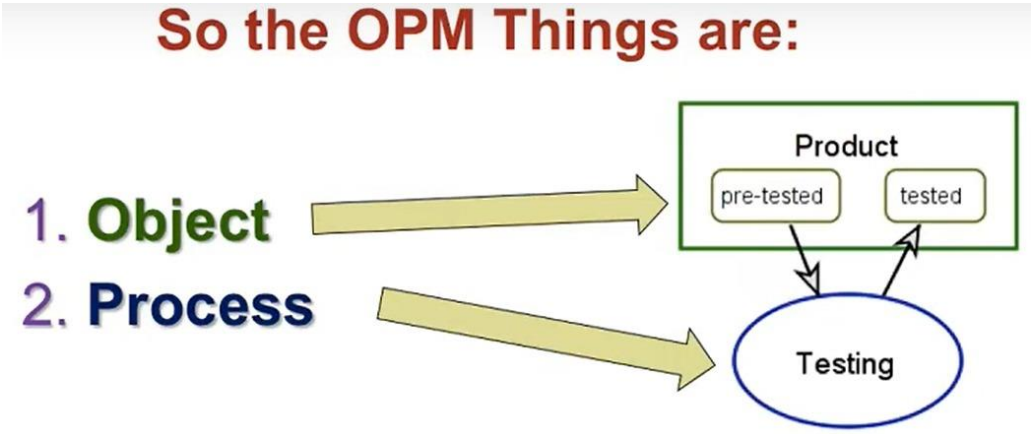
Sürecin nesneyi tüketmesi temsil edilmiştir.



Sürecin nesneyi nasıl yarattığı temsil edilmiştir.



Sürecin nesnenin durumunu nasıl değiştirdiği temsil edilmiştir.



Yapı ve davranış sadece kavramdan ibarettir.

OPM çift modludur ve sistem modelini aynı anda iki farklı yöntemle temsil eder.

1.3.1.1. OPD

- OPM ‘in tek diyagramıdır. Diyagram türünün bu eşsizliği OPM’in sadeliğine katkıda bulunur özellikle 14 çeşit diyagramı olan UML ve 9 çeşit diyagramı olan SysML ile keskin bir tezat oluşturur.
- OPD, nesneleri, süreçleri ve aralarındaki bağlantıları grafiksel olarak tanımlar.
- Bağlantılar (Links) yapısal ve prosedürel olabilir.
- Yapısal bağlantılar, nesneleri nesnelere veya süreçleri süreçlere bağlar, bu sayede statik sistem yönünü ifade eder. (Sistemin nasıl yapılandırıldığı). Yordamsal bağlantılar, nesneleri işlemlere bağlar, bu sayede dinamik sistem yönünü ifade eder (Sistem zaman içinde nasıl değiştiğini).

1.3.2.2. OPL

- Her OPD yapısı (yani, bir veya daha fazla bağlantıyla birbirine bağlanan iki veya daha fazla şey) OPL'deki bir cümleye çevrilir, bu cümle İngilizcenin bir alt kümesidir.
- OPL'in gücü, insanlar tarafından okunabileceği, ancak bilgisayarlar tarafından da yorumlanabileceği gerçeğinde yatmaktadır.

- Her model olgusu hem grafiksel hem de metinsel olarak ifade edildiğinden, teknik olmayan paydaşların kolayca erişebilmelerini sağlayarak, sistem gereksinimlerinin ortaya çıkarılması, tasarlanması ve geliştirilmesinin erken, kritik aşamalarında yer almasını sağlar.

OPM consists of OPM Language and OPM Methodology.
OPM Model consists of OPD Set and OPL Spec.
Modeler is physical.
Modeler handles OPM Modeling.
OPM Modeling requires OPM Methodology and OPM Language.
OPM Modeling yields OPM Model.

1.4. Links

Linkler 3 gruba ayrılır.

- Structural Links
- Procedural Links
- Event and condition

1.4.1. Structural Links (Yapısal Bağlantı)

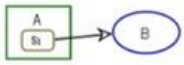
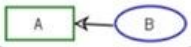
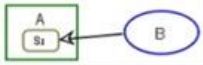
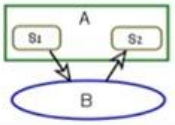
Yapısal ilişkiyi temsil eder. Yapısal bir ilişki, sistemde en azından bir süre devam eden bir ilişkiyi belirtecektir. Yapısal bağlantılar sistemdeki statik, zamandan bağımsız, uzun süreli ilişkileri belirtir.

Symbol	Name	OPL	Allowed Source-to-Destination connections	Semantics/ Effect on the system flow/ Comments
▲	Aggregation-Participation	A consist of B.	Object-Object Process- Process	Whole -Part
▲	Exhibition-Characterization	A exhibits B.	Object-Object Object-Process Process-Object Process- Process	
△	Generalization-Specialization	B is an A. (objects) B is A. (processes)	Object-Object Process- Process	
◭	Classification-Instantiation	B is an instance of A.	Object-Object Process- Process	
→ ⇄	Tagged structural links: Unidirectional Bidirectional	According to text added by user	Object-Object Process- Process	Describes structural information.

Yapısal bir bağ, nesne -nesne ve işlem – işlem gibi bağlantıları birbirine bağlar ancak Exhibition-Characterization bağlantısı dışında nesne – işlem bağlantısı yapamaz.

1.4.2. Procedural Links (Yordamsal Bağlantı)

Prosedürel ilişkiyi belirler. Prosedürel bir ilişki, nesneleri dönüştüren süreçlerin zamana bağlı veya koşullu tetiklenmesini belirleyerek sistemin işlevini yerine getirmek için nasıl çalıştığını belirleyecektir.

Name	Symbol	OPL	Semantics
Consumption Link		B consumes A.	Process B consumes Object A.
State-Specified Consumption Link		B consumes s1 A.	Process B consumes Object A when it is at State s1.
Result Link		B yields A.	Process B creates Object A.
State-Specified Result Link		B yields s1 A.	Process B creates Object A at State s1.
Input-Output Link Pair		B changes A from s1 to s2.	Process B changes the state of Object A from State s1 to State s2.
Effect Link		B affects A.	Process B changes the state of Object A;

1.4.3. Event and condition

Olay – koşul – eylem paradigması OPM için kontrol akışı sağlar. Olay, bir nesnenin oluşturulduğu veya bir nesnenin belirli bir duruma girdiği bir noktadır.

1.5. OPM & SysML

- OPM ve SysML, sistem modellemesine iki farklı yaklaşımı temsil eder.
- SysML, bağımsız olarak türetilen dokuz adet diyagram kullanır.
- OPM ise sadece nesne – süreç diyagramı kullanır. OPM de birkaç modelin entegre edilmesi ihtiyacı daha karmaşık olabilir.
- SysML grafikler ile modellenirken OPM grafikler ve metinsel olarak (bimodal) modellenir.
- SysML UML'in profil mekanizması kullanılarak Birleşik Modelleme Dili'nin(UML) bir uzantısı olarak tanımlanır. OPM durumsal nesnelerin ve onları oluşturan süreçlerin minimal ontolojisi (Minimal Universal Ontology) ile tanımlanır.

1.6. OPM & UML

OPM ve UML arasındaki farklar analiz ve tasarım aşamalarında oldukça algılanabilir.

- UML çok modelli bir yapıda olurken OPM ise tek bir modelleme üstünde durur.
- OPM de sistemin davranışları ve yapıları yan yana belirtilirken UML de ise sistemin davranış ve yapıları farklı ve ayrı görüntülerde belirtilir.
- OPM farklı görüntüleri birleştirme ihtiyacı duymazken UML de ise bu ihtiyacı duyar.

1.7. ISO-19450

- ISO 19450'de belirtildiği gibi OPM notasyonu, sistemlerin resmi sözdizimi ve anlambilim yoluyla kavramsal modellemesini destekler. Bu formalite, sistem mimarlığı, mühendislik, geliştirme, yaşam döngüsü desteği, iletişim ve evrim dahil olmak üzere genel olarak model tabanlı sistem mühendisliğinin (MBSE) temelini oluşturur. OPM, çok disiplinli bir

ortamda çalışmayı sağlayarak, yapım, test, entegrasyon ve günlük bakım altında sistemin ortak bir görünümünü kolaylaştırır.

1.8. OPCAT

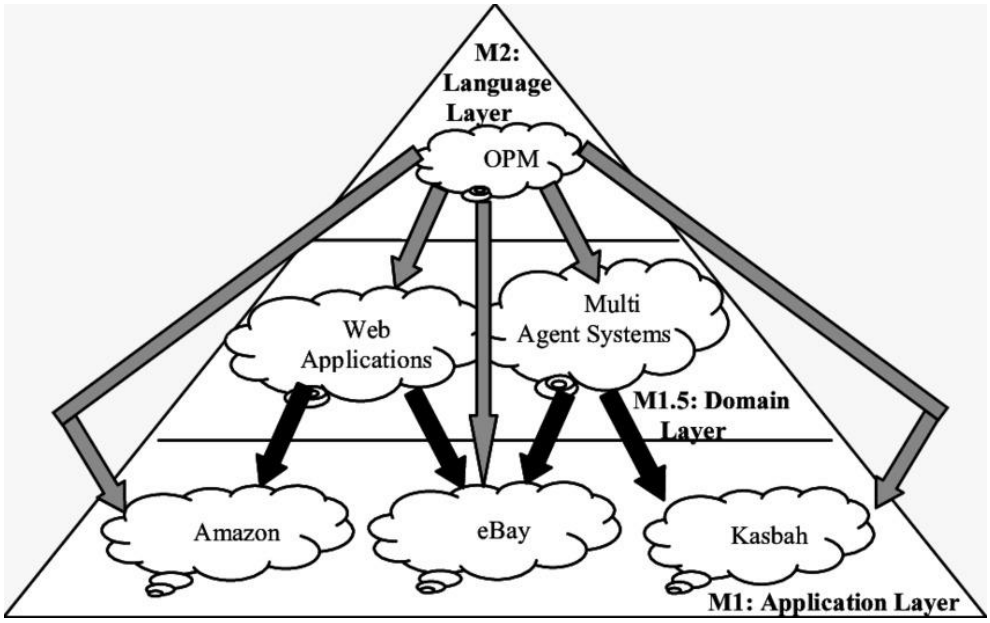
- OPCAT, eşzamanlı, senkronize ve ayrık zamanlı yürütme yoluyla görsel modelleme ve model simülasyonunu mümkün kılar (Yaroker vd., 2013). Yürütme, incelenen sistemin altında yatan mekanizmaları analiz ederek sistemin davranışının anlaşılmasını ve modelleme hatalarının saptanmasını sağlar**.**

1.9. XML

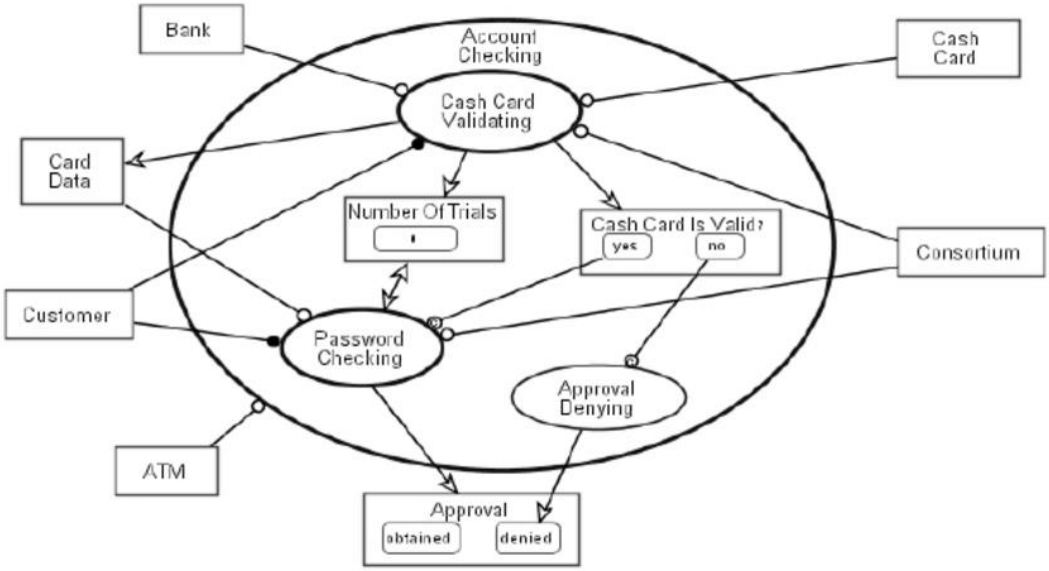
- XML (Extensible Markup Language ya da Türkçesiyle Genişletilebilir İşaretleme Dili), hem insanlar hem bilgi işlem sistemleri tarafından kolayca okunabilecek dokümanlar oluşturmaya yarayan bir işaretleme dilidir. W3C tarafından tanımlanmış bir standarttır. Bu özelliği ile veri saklamanın yanında farklı sistemler arasında veri alışverişi yapmaya yarayan bir ara format görevi de görür. SGML'in basitleştirilmiş bir alt kümesidir.

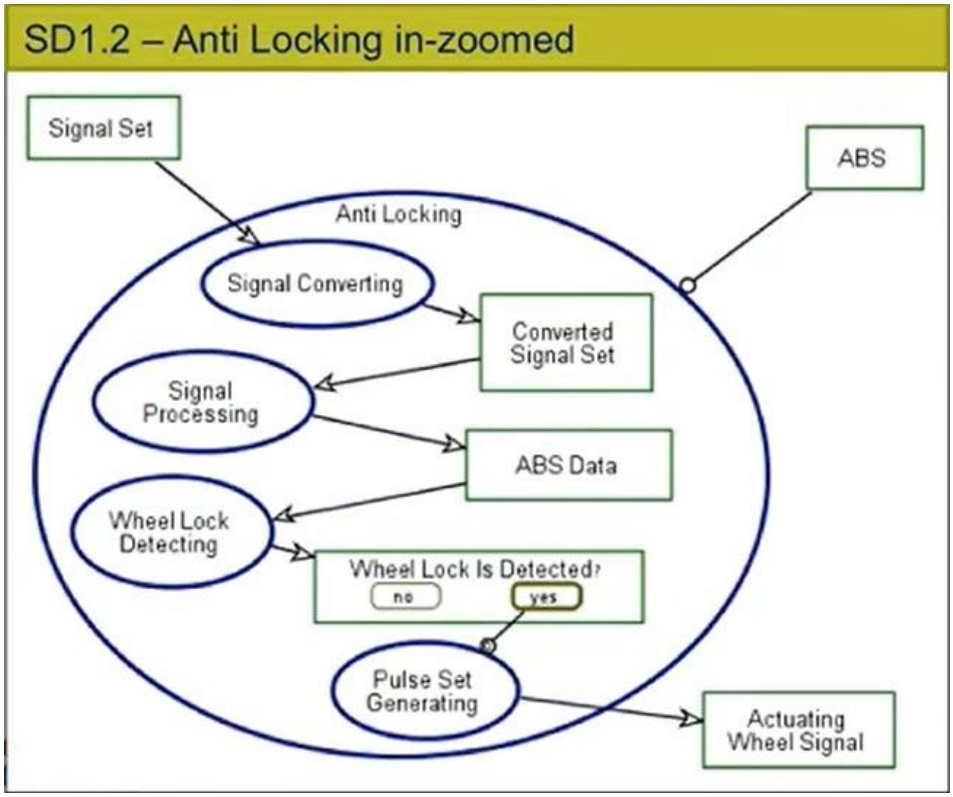
1.10. OPM Kullanım Alanları

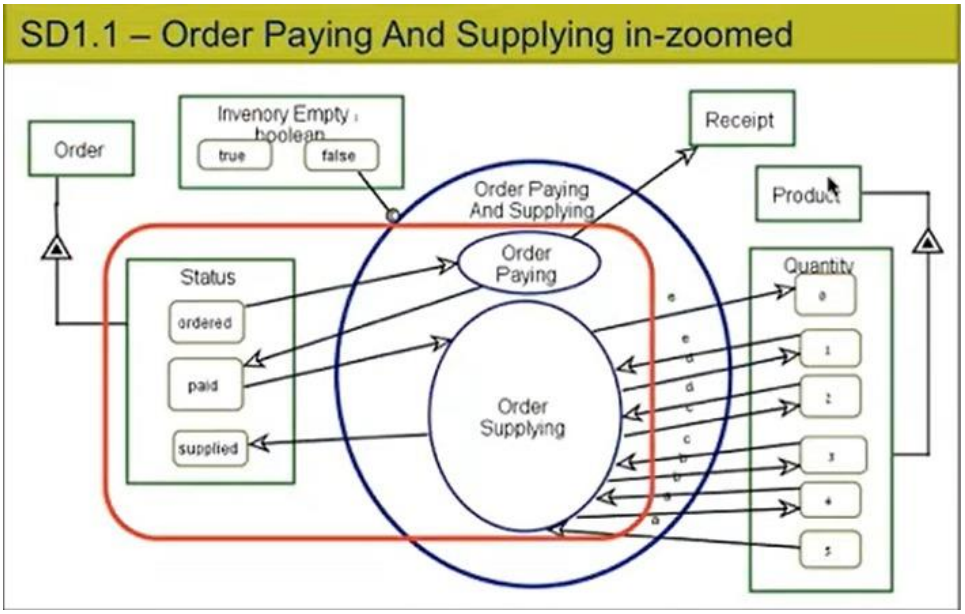
- OPM üzerine ilk makaleden bu yana (Dori, 1995), iki kitap (Dori 2002; 2016) ve düzinelerce makale, çok çeşitli çok disiplinli insan yapımı sosyo-teknik sanayi ve bilgi sistemlerinde ve biyolojik sistemlerde yayınlanmıştır. ISO 19450 (2015) olarak OPM, Aralık 2015'te Uluslararası Standardizasyon Örgütü (ISO) OPM'yi ISO 19450 olarak tanıdı.OPM uygulayan alanlar arasında moleküler biyoloji (Somekh ve diğerleri, 2014), tıp (Wachs ve diğerleri, 2014), uydu iletişim yazılımı geliştirme (Dori ve Thipphayathetthana, 2015), sosyo-teknik sistemler (Osorio ve diğerleri, 2011) bulunmaktadır.), veri depolama (Dori ve diğerleri, 2008), ajan teknolojisi (Sturm ve diğerleri, 2010) ve havacılık (Mordecai ve diğerleri, 2016).



1.11. Proje Örnekleri







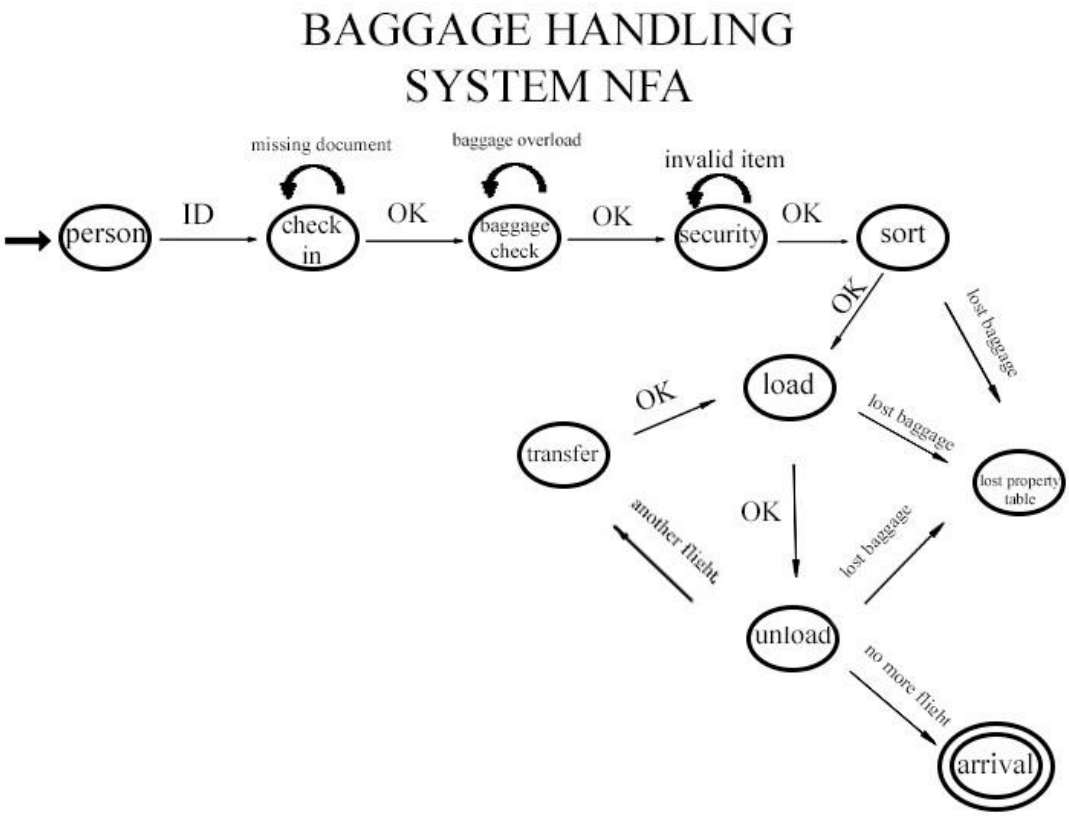
2. Problem Özellikleri ve Çözüm Yaklaşımı

Proje konusu olarak Bagaj Taşıma Sistemini ele alındı. Bagaj Taşıma Sistemi havaalanlarında kontrol edilen bagajların hareket eden bir bant vasıtasıyla uçağa taşıma ve uçaktan bagaj teslim noktasına taşınma modelidir. Bu sistem Biçimsel Diller ve Otomata dersinde Sonlu Devinim Otomata (NFA) konusu ile alakalıdır.

Yapılan uygulama OPCAT üzerinde modellenerek C# üzerinde simülasyonu yapıldı.

2.1. Problem Çözümü

Havaalanına gelen bir yolcunun "ID"'si alındıktan sonra CHECK-IN aşamasına "OK" ise BAGGAGE CHECK aşamasına geçer. Eğer CHECK-IN aşamasında "MISSING DOCUMENT" gelirse tekrar CHECK-IN aşamasına döner. BAGGAGE CHECK işlemininde "OK" gelirse SECURITY aşamasına geçmektedir, işlem başarısız olursa CHECK-IN aşamasına geri döner. SECURITY aşamasında "OK" gelirse SORT aşamasına geçer, "INVALID ITEM" gelirse tekrar SECURITY aşamasında döner. SORT aşamasında "OK" gelirse bagaj LOAD aşamasına geçer. LOAD aşamasında "OK" gelirse unload aşamasına geçer. UNLOAD aşamasında "NO MORE FLIGHT" gelirse ARRIVAL aşamasına geçer ve **biter**. Eğer bu üç aşamada (SORT, LOAD, UNLOAD) "LOST BAGGAGE" geilirse LOST PROPERTY TABLE aşamasına geçer. Yolcunun aktarmalı uçuşu varsa UNLOAD aşamasına "ANOTHER FLIGHT" gelir ve TRANSFER aşamasına geçer. TRANSFER aşamasına "OK" gelirse LOAD aşamasına geçer. Bu döngü aktarmalı uçuş kalmayana kadar devam eder.



NFA KABUL DURUMU (iki adet örnek verilmiştir):

ID OK OK OK OK OK NO-MORE-FLIGHT,

ID OK OK OK OK OK ANOTHER-FLIGHT OK OK NO-MORE-FLIGHT

DFA RET DURUMU (iki adet örnek verilmiştir):

ID OK OK OK OK OK

ID OK OK OK OK LOST-BAGGAGE

∑ {ID, OK, missing document,baggage overload,invalid item,lost baggage,no more flight,another flight}

& (person,ID) ={ check-in }

& (check-in,missing document)={ check-in}

& (check-in,OK) ={ baggage check }

& (baggage check,baggage overload)={check-in}

& (baggage check,OK)={security}

& (security,invalid item)={security}

& (security,OK)={sort}

& (sort,lost baggage)={lost property table}

& (sort,OK)={load}

& (load,lost baggage) = {lost property table}

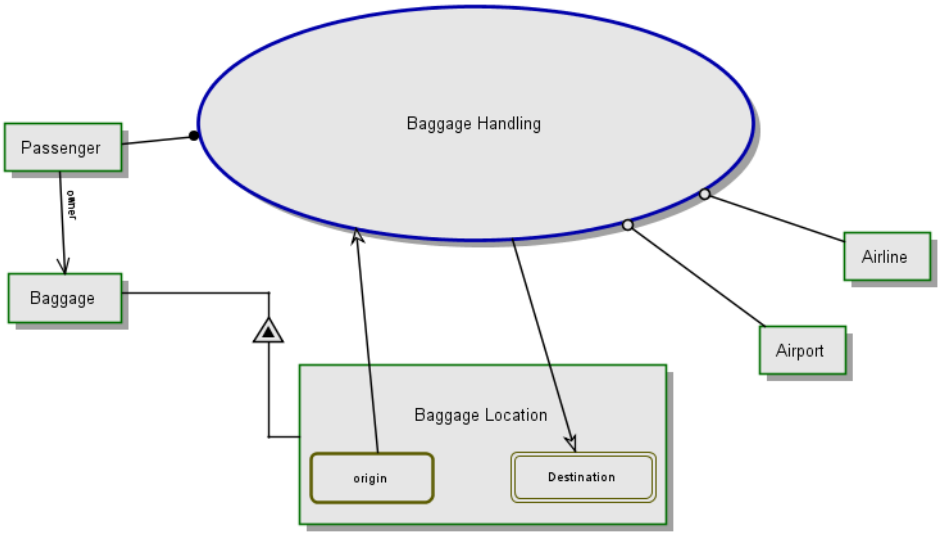
& (load,OK) = {unload}

& (unload,lost baggage) = {lost property table}

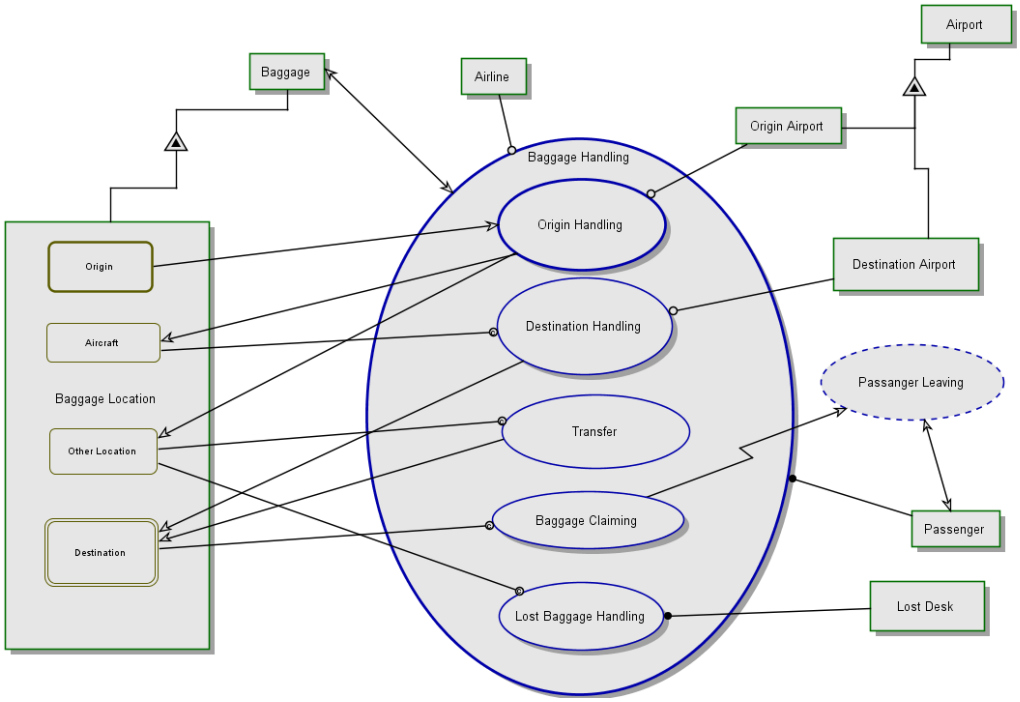
& (unload,no more flight) = {arrival}

& (unload,another flight) = {transfer}

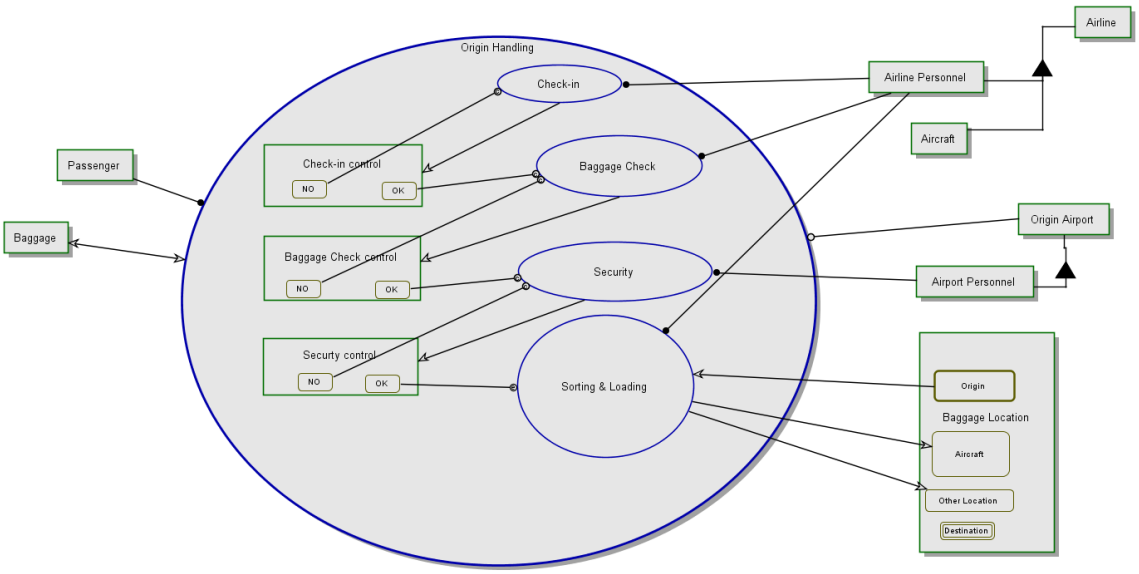
& (transfer, OK) = {load}



Passenger is physical.
Passenger owner Baggage.
Passenger handles Baggage Handling.
Baggage is physical.
Baggage exhibits Baggage Location.
    Baggage Location is physical.
    Baggage Location can be origin or Destination.
        origin is initial.
        Destination is final.
Airport is physical.
Airline is physical.
Baggage Handling is physical.
Baggage Handling requires Airport and Airline.
Baggage Handling changes Baggage Location from origin to Destination.



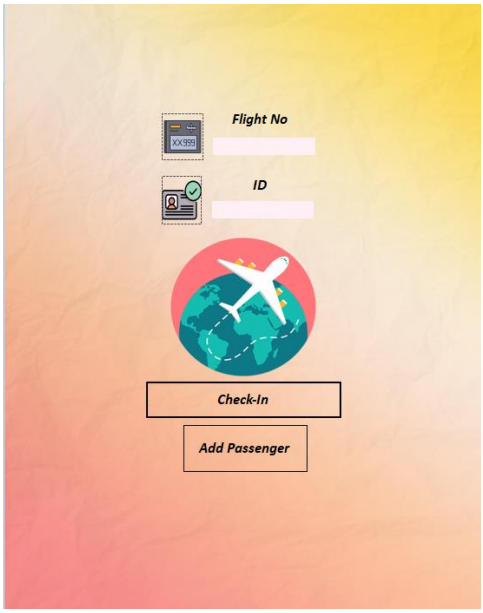
Passenger is physical.
Passenger handles Baggage Handling.
Airport is physical.
Airport exhibits Origin Airport and Destination Airport.
    Origin Airport is physical.
    Destination Airport is physical.
Airline is physical.
Lost Desk is physical.
Lost Desk handles Lost Baggage Handling.
Baggage is physical.
Baggage exhibits Baggage Location.
    Baggage Location is physical.
    Baggage Location can be Origin, Aircraft, Destination, or Other Location.
        Origin is initial.
        Destination is final.
Passanger Leaving is environmental.
Passanger Leaving affects Passenger.
Baggage Handling is physical.
Baggage Handling consists of Origin Handling, Destination Handling, Lost Baggage Handling, Baggage Claiming, and Transfer.
Baggage Handling requires Airline.
Baggage Handling affects Baggage.
Baggage Handling zooms into Origin Handling, Destination Handling, Transfer, Baggage Claiming, and Lost Baggage Handling.
    Origin Handling is physical.
    Origin Handling requires Origin Airport.
    Origin Handling consumes Origin Baggage Location.
    Origin Handling yields either Other Location Baggage Location or Aircraft Baggage Location.
    Destination Handling is physical.
    Destination Handling occurs if Baggage Location is Aircraft.
    Destination Handling requires Destination Airport.
    Destination Handling yields Destination Baggage Location.
    Transfer occurs if Baggage Location is Other Location.
    Transfer yields Destination Baggage Location.
    Baggage Claiming is physical.
    Baggage Claiming occurs if Baggage Location is Destination.
    Baggage Claiming invokes Passanger Leaving.
    Lost Baggage Handling is physical.
    Lost Baggage Handling occurs if Baggage Location is Other Location.



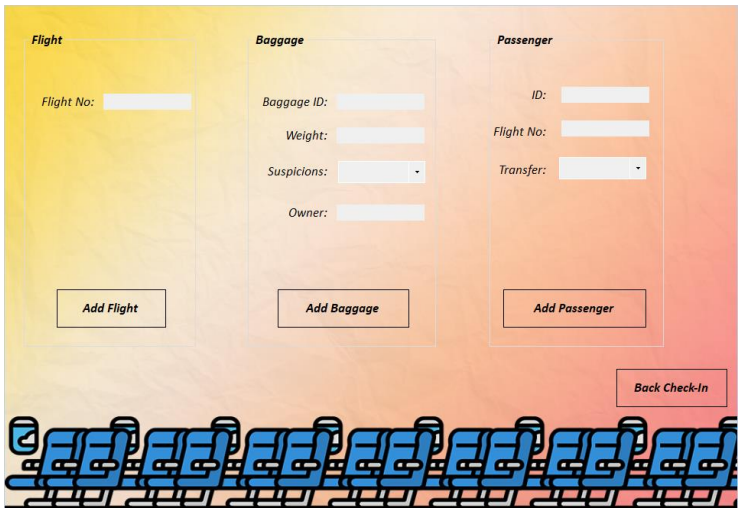
Passenger is physical.
Passenger handles Origin Handling.
Airline is physical.
Airline consists of Airline Personnel and Aircraft.
    Airline Personnel is physical.
    Airline Personnel handles Baggage Check, Sorting & Loading, and Check-in.
    Aircraft is physical.
Baggage Location is physical.
Baggage Location can be Origin, Aircraft, Destination, or Other Location.
    Origin is initial.
    Destination is final.
Origin Airport is physical.
Origin Airport consists of Airport Personnel.
    Airport Personnel is physical.
    Airport Personnel handles Security.
Baggage is physical.
Origin Handling is physical.
Origin Handling exhibits Check-in control, Baggage Check control, and Securty control.
Origin Handling consists of Check-in, Sorting & Loading, Baggage Check, and Security.
Origin Handling requires Origin Airport.
Origin Handling affects Baggage.
Origin Handling zooms into Check-in, Baggage Check, Security, and Sorting & Loading, as well as Securty control, Baggage Check control, and Check-in control.
    Securty control can be NO or OK.
    Securty control triggers Sorting & Loading when it enters OK.
    Baggage Check control can be NO or OK.
    Check-in control can be OK or NO.
    Check-in occurs if Check-in control is NO.
    Check-in yields Check-in control.
    Baggage Check occurs if Check-in control is OK and Baggage Check control is NO.
    Baggage Check yields Baggage Check control.
    Security occurs if Securty control is NO and Baggage Check control is OK.
    Security yields Securty control.
    Sorting & Loading requires OK Securty control.
    Sorting & Loading changes Baggage Location from Origin to Other Location and Baggage Location from Origin to Aircraft.

2.2.Uygulama Yazılımı

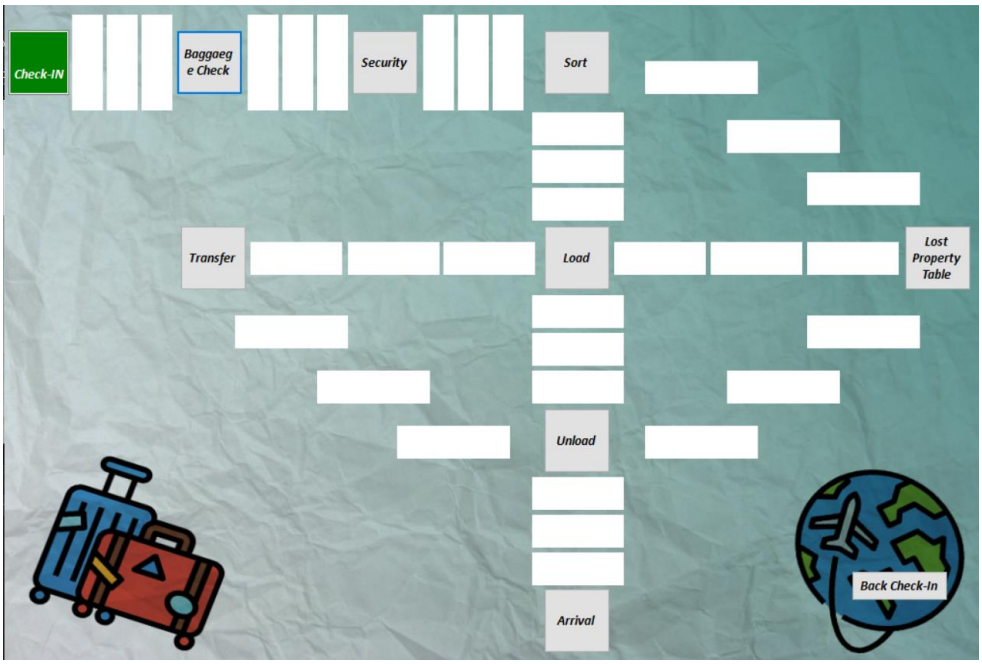
https://bitbucket.org/152120171007/baggage-handling-system/src/develop/



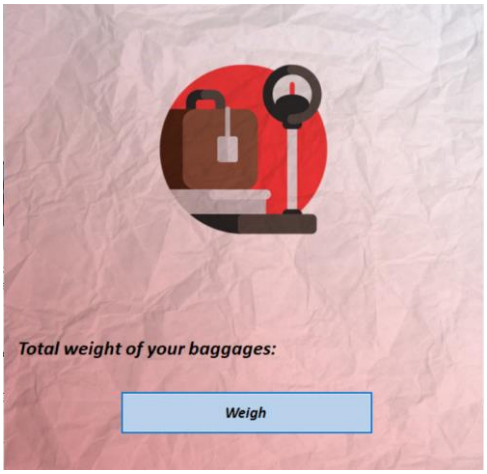
Simülasyonun karşımıza çıkan ilk arayüzü yukarıdaki görseldir. Bu arayüz de yolcunun uçuş numarası ve kimlik numarasına göre check-in işlemini gerçekleştiriyor. Eğer yolcu biletini henüz almamış ise biletini alması için add passenger arayüzüne geçiş yapmak zorundadır.



Add Passenger arayüz tasarımı yukarıdaki görseldir. Burada kişinin bilgileri alınır ve yeni yolcu oluşturulur.Back Check -In kısmı ile de yolcu check-in kısmına yönlendirilir.

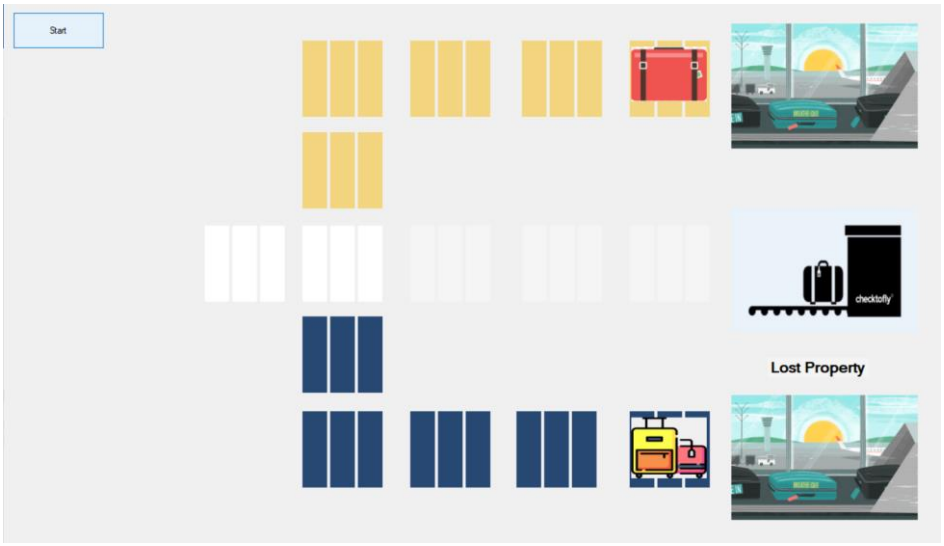


Check-In işlemi tamamlandıktan sonra Baggage Handling System'in arayüzü karşımıza çıkmaktadır. Bu sistemde genel amaç bagajların başlangıç konumundan varış konumuna nasıl taşındığını göstermektir.



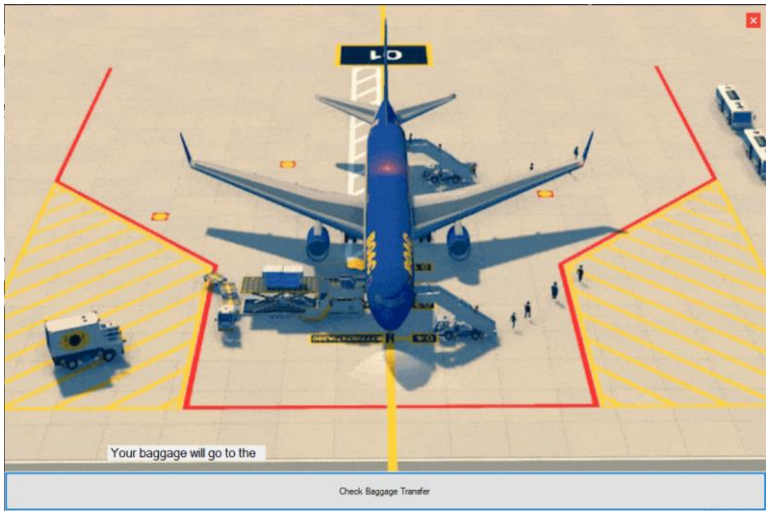
Check-in kısmını başarıyla geçen yolcu bu arayüz de bagajının ağırlığını kontrol ettirir.Eğer belirlenen kilogram sınırını aşmış ise fazladan bagaj hakkı satın alır.

Baggage Check işlemini başarıyla geçen bagaj Securtiy arayüzüne gelir ve içerisinde şüpheli bir eşya var mı yok mu kontrol edilir. Eğer şüpheli bir eşya var ise çıkarılması beklenir.

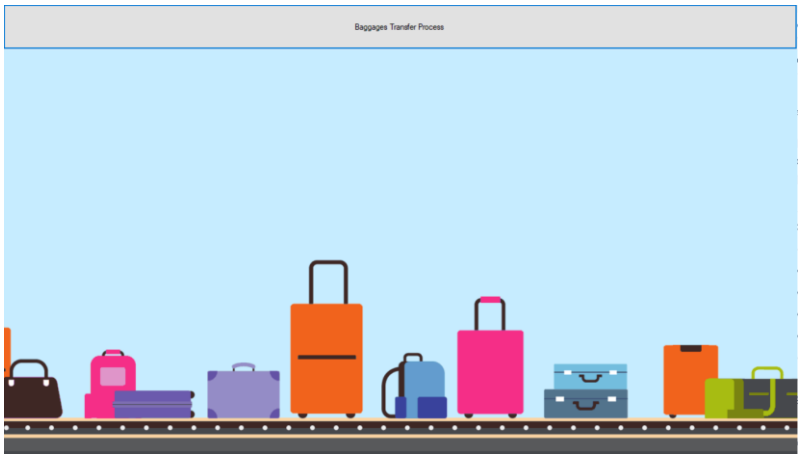


Securtiy işlemini başarıyla geçen bagaj bu arayüz de Sort işlemine tabi tutulur. Bu Sort işlemi sırasında kaybolan bir bagaj var ise Lost Property Table kısmına yönlendirilir.

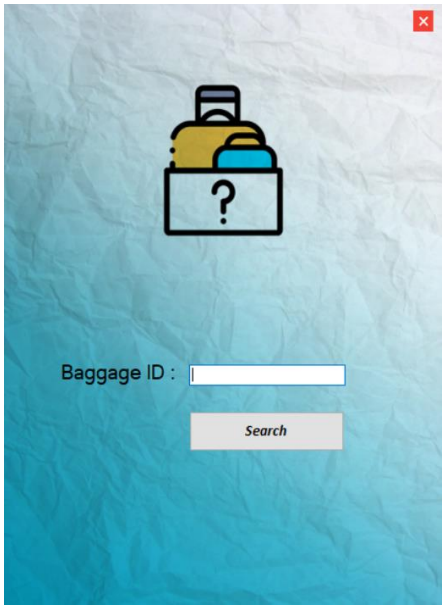
Sort işlemini başarıyla geçen bagaj bu arayüz de uçağa yüklenir. Eğer yükleme aşamasında kaybolan bir bagaj var ise Lost Property Table kısmına yönlendirilir.



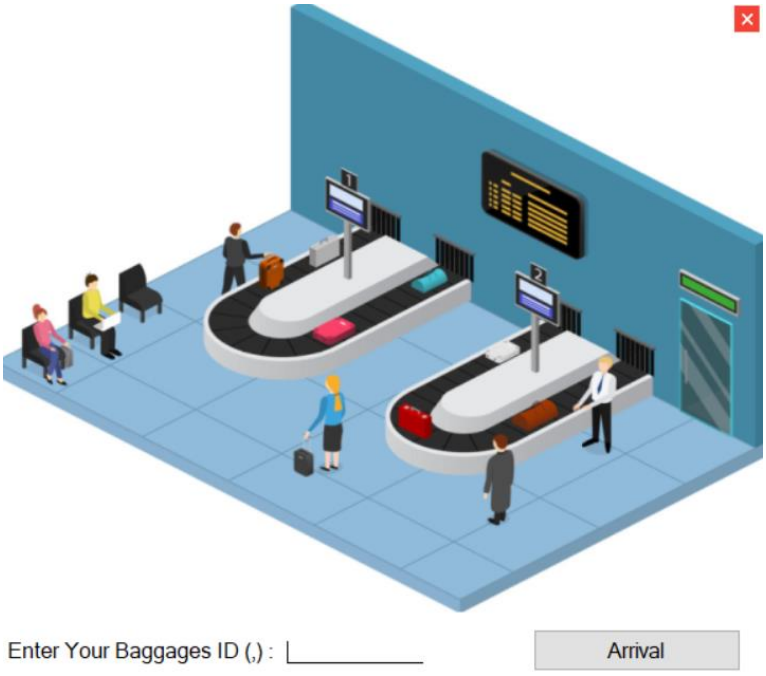
Uçak varış noktasına geldiğinde bu arayüz de bagajların indirme işlemi yapılır.



Transfer olacak bagaj burada bantlar üzerinde valiz taşıma tabi tutulur.



Kaybolan bagajlar bu kısma gelerek Baggage ID bilgisine göre yolcuya verilir.



Bütün işlemlerden başarıyla geçen bagaj Baggage Claim'e gelerek yolcuya verilir.

3. Proje Ekibi Değerlendirmesi

3.1.Grup Bilgisi

3.1.1. Grup Koordinatörü ve Görev Paylaşımı

Grup Koordinatörü

İsmail DEMİRCAN

Görev Paylaşımı (Bu zamana kadar tüm görev paylaşımları yazılmıştır.)

İsmail DEMİRCAN:
OPM'in Kullanım Alanları, ISO-19450, Proje Araştırması, Proje Araştırması, State-Transition ve Çizimi, Görev Dağılımı ve Denetleme, OPM & UML Farkı, OPCAT Kullanımı, Diyagram Dili, Simülasyon, Rapor
Gökhan Samet ALBAYRAK:
XML, OPM'in Tarihçesi, Proje Araştırması, Proje Araştırması, State-Transition, Linkler, OPCAT Kullanımı ve Çizimi, Diyagram Dili, Simülasyon, Rapor

Onur AKKEPENEK:
OPM & SysML, OPM'in Faydaları, Proje Araştırması, Proje Araştırması, State-Transition, OPCAT Kullanımı ve Çizimi, Diyagram Dili, Simülasyon, Rapor
Muzaffer Arda USLU:
Neden Kavramsal Modelleme, OPD & OPL, Proje Araştırması, Proje Araştırması, State-Transition ve Çizimi, OPCAT Kullanımı, Diyagram Dili, Simülasyon, Rapor

3.1.2. Grup Üyelerinin Adam-Saat Süreleri

İsmail DEMİRCAN: 58 Adam-Saat
Gökhan Samet ALBAYRAK: 56 Adam-Saat
Onur AKKEPENEK: 56 Adam-Saat
Muzaffer Arda USLU: 57 Adam-Saat
(Bu zamana kadar toplam saat verilmiştir.)

3.2 Yapılan Toplantı Raporu

Toplantı 1 Tarih: 10.03.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

Toplantı esnasında grup üyeleri konu hakkındaki fikirlerini belirtti. Projeye dair oluşan sorular ve kilit noktalar konuşuldu.

ve şu kararlar alındı:

- Discord ve WhatsApp grupları oluşturulmasına karar verildi.
- Konular alt başlıklara ayrıldı ve grup üyelerine dağıtıldı.

Toplantı 2 Tarih: 21.03.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Toplantımız Covid-19 salgını nedeniyle internet ortamından gerçekleştirildi. Yapılan araştırmalar sonucu ön rapor hazırlandı.
- Sunumun ön rapor teslim edildikten sonra hazırlanmasına karar verildi.

Toplantı 3 Tarih: 08.04.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

Toplantı esnasında grup üyeleri proje hakkındaki fikirlerini belirtti. Projeye dair oluşan sorular ve kilit noktalar konuşuldu.

ve şu kararlar alındı:

- Projeyle ilgili görev dağılımı yapıldı ve herkesin kendi konusunu araştırmasına karar verildi.
- Temel OPCAT kullanımının öğrenilmesine karar verildi.
- Sonlu Devinimli Otomatalar konusunun tekrar edilmesi kararı alındı.

Toplantı 4 Tarih: 19.04.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Araştırmalar sonucu genel tasarım oluşturuldu.
- Yapılan tasarımın State-Transition diyagramı oluşturuldu.
- OPCAT üzerinden çizimine başlanıldı.

Toplantı 5 Tarih: 21.04.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- OPCAT çizimi bittikten sonra C# uygulaması yapılmasına karar verildi.
- Genel tablo gözden geçirildi.

Toplantı 6 Tarih: 26.04.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- OPCAT çizimi state diagram göz önüne alınarak bitirildi.
- Simülasyon için gerekli kaynak ve bilgiler için araştırma yapılmasına karar verildi.

Toplantı 7 Tarih: 03.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Simülasyonun dataları için XML kullanılmasına karar verildi.
- XML hakkında herkesin araştırma yapılmasına karar verildi.

- Sürüm kontrol sistemi kullanılmasına karar verildi.

Toplantı 8 Tarih: 10.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Proje tasarımı için dizayn fikirleri alındı.
- Projenin SOLİD kavramlarına uygun olarak yapılmasına karar verildi.

Toplantı 9 Tarih: 17.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Proje için görev dağılımı yapıldı (class dağılımı).
- Kullanılacak dizayn belirlendi ve uygulamaya konuldu.

Toplantı 10 Tarih: 20.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Proje için verilen görevlerin yapılıp yapılmadığı kontrol edildi.
- Proje geliştirilmeye devam edildi.

Toplantı 11 Tarih: 22.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Projenin simülasyonu için 22.05.2020- 27.05.2020 tarihleri arasında her gün Discord üzerinden toplanılmasına karar verildi.

Toplantı 12 Tarih: 27.05.2020

Katılımcılar:

Muzaffer Arda USLU

Gökhan Samet ALBAYRAK

Onur AKKEPENEK

İsmail DEMİRCAN

- Simülasyonun son hali kontrol edilip gerekli düzeltmeler yapıldı ve rapor yazıldı.

4. Sonuçlar

Bagaj taşıma sisteminin modellenmesinde OPCAT kullanılmasının daha işlevsel olduğu sonucuna varılmıştır ve belli başlı tasarımlara başlanmıştır. Tasarımların tamamlanmasından sonrasındaki aşamada C# üzerinden simülasyon gerçekleştirilmesi yapıldı. Yaptığımız bu simülasyon sürecinde en zorlandığımız kısım OPCAT modellemesi oldu. Model oluşturmanın önemini kavradığımız bu projede doğru ve iyi hazırlanmış bir modelin kodlanmasının çok daha kolay ve anlaşılır olduğunu fark ettik. Takım olarak bir proje çıkarırken birbirimizin açıklarını kapattık ama versiyon kontrol sistemi kullanımında yüksek tecrübeye sahip olmadığımızdan dolayı bazı sorunları çözmemiz zaman aldı. Codefator.io ile de kodumuzun düzenini test ettik. Sonuç olarak, yapılan bu projede kendimize birçok yeni şey kattığımızı düşünüyoruz. Mesela; temel OPCAT kullanımı, Nesne Süreç Metodolojisi(OPM), XML dosya işlemleri ve takım çalışması.

5. Ek 1: Kodlar

```
/**
* @brief    : Bu fonksiyon kaybolan bagajın hangi havalimanın da kaybolduğunu belirler ve kaybolduğu havalimanından teslim alınmasını sağlar.
* Eğer sort ya da load süreçlerinde bagaj kaybolursa bagajın konumu origin aiport, eğer unload sürecinde bagaj kaybolursa destination aiport
* olarak konuma atama yapılıyor. Kaybolan bagaj numarası yolcunun bagajının ID'si ile eşleşirse bagajı teslim alabilr.
*/
1 reference
private void btnFindLostBaggage_Click(object sender, EventArgs e)
{
    string baggageControl = "";
    Airport airport = new Airport();
    if (HandlingSystem.LostPropertyWay == 9 || HandlingSystem.LostPropertyWay == 10)
        airport.Location = "Origin Airport";
    else
        airport.Location = "Destination Airport";
    txtBoxBaggageID.Text = "," + txtBoxBaggageID.Text;
    for (int i = 0; i < Airline.passengerList[HandlingSystem.index].Baggages.Count(); i++)
    {
        if (Airline.passengerList[HandlingSystem.index].Baggages[i].Owner == "")
        {
            baggageControl += "," + Airline.passengerList[HandlingSystem.index].Baggages[i].BaggageID.ToString();
            Airline.passengerList[HandlingSystem.index].Baggages[i].BaggageLocation = airport.Location;
        }
    }
    if (baggageControl == txtBoxBaggageID.Text)
    {
        MessageBox.Show("You can take the your baggages.", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);
        Lost_Timer.Start();
    }
    else
        MessageBox.Show("No baggage for this number was found.", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
    txtBoxBaggageID.Text = "";
}
```

```
/**
* @brief   : Bu fonksiyon passengerXML dosyasinda kayitli olan yolculari okur ve Passenger Listesine kaydeder.
* @param   : Bu fonksiyon, icerisinde Passenger tutan bir adet Liste parametresine sahiptir.
*/
1 reference
public static void readPassengerXMLFile(List<Passenger> passengerList)
{
    XDocument xDoc = XDocument.Load(@"data/passengerXML.xml");
    XElement passengerRootElement = xDoc.Root;
    foreach (XElement passenger in passengerRootElement.Elements())
    {
        passengerNo = Int32.Parse(passenger.FirstAttribute.Value);
        List<Baggage> tempBaggageList = new List<Baggage>();

        XElement baggageRootElement = passenger.Element("baggageList");
        foreach (XElement baggage in baggageRootElement.Elements())
        {
            Baggage bagg = new Baggage(bool.Parse(baggage.Element("suspicions").Value),
            double.Parse(baggage.Element("baggageWeight").Value), baggage.Element("baggageID").Value,
            baggage.Element("owner").Value, baggage.Element("baggageLocation").Value);
            tempBaggageList.Add(bagg);
        }
        Passenger pass = new Passenger(passenger.Element("passengerID").Value, passenger.Element("flightNo").Value,
            bool.Parse(passenger.Element("extraBaggageAllowance").Value), bool.Parse(passenger.Element("transfer").Value)
            , tempBaggageList);
        passengerList.Add(pass);
    }
    passengerNo++;
}
```

```
/**
* @brief   : Bu fonksiyon passengerXML dosyasina yeni bir yolcuyu ve yolcunun bagajlarini kaydeder.
* @param   : Bu fonksiyon, Passenger tipinde bir adet parametreye sahiptir.
*/
1 reference
public static void saveXMLFile(Passenger psg)
{
    XDocument xDoc = XDocument.Load(@"data/passengerXML.xml");
    XElement rootElement = xDoc.Root;
    XElement newElementPassenger = new XElement("passenger");
    XAttribute passengerAttribute = new XAttribute("passengerNo", passengerNo.ToString());
    passengerNo++;
    XElement passengerID = new XElement("passengerID", psg.ID1);
    XElement passengerFlightNo = new XElement("flightNo", psg.FlightNo1);
    XElement passengerExtraBaggageAllowance = new XElement("extraBaggageAllowance", psg.ExtraBaggageAllowance1);
    XElement passengerTransfer = new XElement("transfer", psg.Transfer);
    XElement baggageListElement = new XElement("baggageList");
    for (int i = 0; i < psg.Baggages.Count; i++)
    {
        XElement newElementBaggage = new XElement("baggage");
        XAttribute bagggeAttribute = new XAttribute("baggageNo", (i + 1).ToString());
        XElement baggagesuspicions = new XElement("suspicions", psg.Baggages[i].Suspicios);
        XElement baggageID = new XElement("baggageID", psg.Baggages[i].BaggageID);
        XElement baggageWeight = new XElement("baggageWeight", psg.Baggages[i].Weight);
        XElement baggageOwner = new XElement("owner", psg.Baggages[i].Owner); ;
        XElement baggageLocation = new XElement("baggageLocation", psg.Baggages[i].BaggageLocation); ;
        newElementBaggage.Add(bagggeAttribute, baggagesuspicions, baggageID, baggageWeight, baggageOwner, baggageLocation);
        baggageListElement.Add(newElementBaggage);
    }
    newElementPassenger.Add(passengerAttribute, passengerID, passengerFlightNo, passengerExtraBaggageAllowance,
        passengerTransfer, baggageListElement);
    rootElement.Add(newElementPassenger);
    xDoc.Save(@"data/passengerXML.xml");
}
```

6. Kaynakça

- https://samim.io/p/2020-01-07-object-process-methodology-opm-is-a-conceptual-model/
- https://pdfs.semanticscholar.org/47d4/bc987c3515b302ec200f1bb082c96e92d790.pdf
- https://en.wikipedia.org/wiki/Object_Process_Methodology
- https://tr.wikipedia.org/wiki/XML
- https://www.youtube.com/watch?v=X8io71hTg8A&t=2919s
- https://www.iso.org/obp/ui/#iso:std:iso:pas:19450:ed-1:v1:en
- https://www.researchgate.net/figure/The-transition-diagram-of-the-ATM-behaviors_fig2_220636950
- https://people.engr.ncsu.edu/efg/210/s99/Notes/fsm/
- https://www.youtube.com/watch?v=FTFnf6y0HiA&t=2679s