



***Ege Elektromobil  
BrakeAndDirection  
Komponent  
Gereksinim Spesifikasyon  
(SRS) Dökümanı***

*Atakan Ertekin*

*<http://www.egeelektromobiltakimi.com/>  
2025*

---

## Revizyonlar

Versiyon	Yazar(lar)	Versiyon Açıklaması	Tamamlanma Tarihi
1.0.0-0	Atakan Ertekin	BrakeAndDirectionManager komponentinin gereksinim dökümanı oluşturulmuştur. Temel gereksinimler eklenmiştir.	10.Şubat 2025
1.0.1-0	Ahmet Vedat		

## Gözdengeçiren & Onaylayan

### Gereksinim dökümanları onaylanan geçmiş versiyonlar

Onaylayan(lar)	Onaylanan Versiyon	İmza	Tarih

### Gereksinim dökümanlarının gözdengeçirenler

Gözdengeçiren(ler)	Onaylanan Gözdengeçirme	İmza	Tarih

## İÇİNDEKİLER

1. Giriş .....	3
2. Genel Açıklamalar .....	3
3. Fonksiyonel Gereksinimler.....	3
4. Arayüz Sınıf Tanımlamaları .....	<b>Hata! Yer işareti tanımlanmamış.</b>
5. Yazılım Komponentleri .....	<b>Hata! Yer işareti tanımlanmamış.</b>
6. Arayüz Tanımlamaları .....	<b>Hata! Yer işareti tanımlanmamış.</b>
7. Test Senaryoları .....	5
8. Görsel Modeller ve Diyagramlar .....	6
8.1 Temel Flowchart Diyagramları .....	6
8.2 Sıralama Diyagramları.....	6
9. Güncelleme Kayıt Defteri.....	<b>Hata! Yer işareti tanımlanmamış.</b>
10. Güncellenen Bütçe Kaydı.....	<b>Hata! Yer işareti tanımlanmamış.</b>
11. Ekler .....	7

# 1. Giriş

## 1.1 Giriş

Bu doküman, **BrakeAndDirectionManager** komponentinin fonksiyonlarını, gereksinimlerini, kullanılan arayüzleri ve test senaryolarını açıklamaktadır. Komponent, fren ve yön bilgilerini okuyarak yönetir ve ilgili arayüzler aracılığıyla sistemin diğer bileşenlerine iletir.

## 1.2 Dökümanın kapsamı

Bu döküman, **BrakeAndDirectionManager**'ın işleyişini tanımlayan işlevsel gereksinimleri, kullanılan arayüzleri ve doğrulama için test senaryolarını içermektedir. Sistem içindeki konumu ve diğer bileşenlerle olan etkileşimleri açıklanmaktadır.

## 1.3 Genel Bakış

**BrakeAndDirectionManager**, aracın fren ve yön kontrolünü sağlayan bir yazılım bileşenidir. Bu bileşen, **MotorDriverControlManager** üzerinden gelen fren ve yön bilgilerini işler, sistemde uygun güncellemeleri yapar ve **IBrakeStatus** ve **IDirectionStatus** arayüzleri aracılığıyla güncel bilgileri yayımlar.

# 2. Genel Açıklamalar

## 2.1 Temel Fonksiyon tanımlaması

**BrakeAndDirectionManager**'ın temel fonksiyonları:

- **Fren durumu okuma ve yönetme:** **MotorDriverControlManager**'dan gelen fren durumunu değerlendirerek sistemde güncellemeler yapar.
- **Yön durumu okuma ve yönetme:** **MotorDriverControlManager**'dan gelen yön bilgisini değerlendirerek güncel yön bilgisini belirler.
- **Fren ve yön durumlarını yayınlama:** **IBrakeStatus** ve **IDirectionStatus** arayüzleri üzerinden güncellenmiş durum bilgilerini paylaşır.
- **Hız yönetimi ile entegrasyon:** **SpeedControlManager\_ruBrakeUpdate()** çağrılarak fren durumundaki hız yönetimi sağlanır.

## 2.2 Fonksiyonel Hedefler

**BrakeAndDirectionManager**, aracın fren ve yön durumlarını periyodik olarak okuyup güncelleyerek, fren değişikliklerini algılar ve uygun aksiyonları gerçekleştirir. Yön değişikliklerini yalnızca durma veya geri vites senaryolarında uygular, hız kontrol yönetimiyle entegre çalışarak fren durumunu günceller ve hata senaryolarında uygun hata yönetimi mekanizmalarını sağlar.

# 3. Fonksiyonel Gereksinimler

Bu bölümde, **BrakeAndDirectionManager** komponentinin işlevlerini yerine getirebilmesi için gerekli olan gereksinimler detaylandırılmıştır. Gereksinimler, komponentin temel yapı taşlarını oluşturan **base class** tanımına dayanarak belirlenmiştir.

1. **BrakeAndDirectionManager**, aracın **fren ve yön yönetimini sağlayan** temel bir bileşendir. Bu bileşen, gelen **fren ve yön bilgilerini işler**, gerekli güncellemeleri yapar ve **IBrakeStatus** ile **IDirectionStatus** arayüzleri aracılığıyla sistemin diğer bileşenlerine bilgi sağlar.

- 1.1. Komponent, **başlangıç değerleriyle başlatılmalı** ve çalışma sürecinde bu değerleri yönetmelidir.
  - 1.2. Tüm **getter/setter fonksiyonları** ve **arayüz bağlantıları** eksiksiz çalışmalıdır.
  - 1.3. Güncellenen yön ve fren bilgileri **ilgili arayüzlere düzenli olarak yazılmalıdır**.
  - 1.4. **Hata yönetimi mekanizması** aktif olmalıdır.
2. BrakeAndDirectionManager, sistem başlatıldığında **varsayılan olarak ileri yön (DIRECTION\_FORWARD) ve fren devre dışı (BRAKE\_DISABLE) durumunda başlatılmalıdır**.
  - 2.1. Başlangıç değerleri aşağıdaki gibi olmalıdır:
    - 2.1.1. direction = DIRECTION\_FORWARD
    - 2.1.2. brake = BRAKE\_DISABLE
  - 2.2. Yeni değerler validasyon kurallarına göre güncellenmelidir.
  - 2.3. Başlangıçta, bu değerler **IBrakeStatus** ve **IDirectionStatus** arayüzlerine yazılmalıdır.
3. BrakeAndDirectionManager, **MotorDriverControlManager** bileşeninden **IBrakeAndDirection** arayüzü üzerinden **fren ve yön durumlarını belirli periyotlarla okuma** işlemi gerçekleştirmelidir.
  - 3.1. **readBrake()** fonksiyonu çağrılarak **fren durumu**
  - 3.2. **readDirection()** fonksiyonu çağrılarak **yön durumu** okunmalıdır.
  - 3.3. **Periyodik bir döngü içinde** bu kontroller yapılmalı ve **değişiklikler tespit edilmelidir**.
  - 3.4. Eğer bir değişiklik tespit edilirse, ilgili setter fonksiyonları kullanılarak güncelleme yapılmalıdır.
4. MotorDriverControlManager, fren durumu **BRAKE\_UNKNOWN** olarak raporladığında, bu durum bir hata olarak değerlendirilmelidir.
  - 4.1. Eğer **readBrake()** fonksiyonu **BRAKE\_UNKNOWN** döndürürse:
    - 4.1.1. Hata durumu **retVal** değişkenine kaydedilmelidir.
    - 4.1.2. **Error handling mekanizması devreye girmeli** ve hata kodu saklanmalıdır.
    - 4.1.3. **Sistemsal hata bildirimi** sağlanmalıdır.
5. MotorDriverControlManager, yön durumu **DIRECTION\_UNKNOWN** olarak raporladığında, bu durum bir hata olarak değerlendirilmelidir.
  - 5.1. Eğer **readDirection()** fonksiyonu **DIRECTION\_UNKNOWN** döndürürse:
    - 5.1.1. Hata durumu **retVal** değişkenine kaydedilmelidir.
    - 5.1.2. **Sistemsal hata bildirimi** sağlanmalıdır.
6. Aracın yön değişikliği **yalnızca fren aktifken (BRAKE\_ENABLE) yapılabilir**.
  - 6.1. Eğer **brake == BRAKE\_DISABLE** ise, yön değişikliğine **izin verilmemelidir**.
  - 6.2. Eğer **brake == BRAKE\_ENABLE** ise, yön değişikliğine izin verilmelidir.
  - 6.3. Yön değişikliği yapıldığında, yeni değer **IDirectionStatus** arayüzüne yazılmalıdır.
7. MotorDriverControlManager'dan alınan ve güncellenen fren ve yön bilgileri, **IBrakeStatus** ve **IDirectionStatus** arayüzlerine yazılmalıdır.
  - 7.1. Güncellenen **fren durumu, IBrakeStatus** arayüzüne yazılmalıdır.
  - 7.2. Güncellenen **yön durumu, IDirectionStatus** arayüzüne yazılmalıdır.

8. Fren durumundaki herhangi bir deęişiklik, **SpeedControlManager\_ruBrakeUpdate()** fonksiyonuna bildirilmelidir. Frenleme durumu, hız kontrolünü doğrudan etkileyen bir faktördür. Fren etkinleştirildiğinde veya devre dışı bırakıldığında, hız yönetimi bunu dikkate alarak uygun deęişiklikleri yapmalıdır.
- 8.1. **SpeedControlManager\_ruBrakeUpdate()** çağrılmalıdır.
- 8.2. Güncellenmiş hız durumu ilgili bileşenlere iletilmelidir.

## 4. Arayüz Sınıf Tanımlamaları

Abstract Sınıf	Tanım
Abstract_MotorDriver	Motorun hız, fren ve yön yönetimi için gerekli olan arayüzleri içerir.

## 1. Arayüz Tanımlamaları

### Abstract\_MotorDriver

Arayüz	Tanım
IBrakeAndDirection	Fren ve yön sinyallerinin kontrolünü sağlar.
IBrakeStatus	Fren durumunun yönetimini tanımlar.
IDirectionStatus	Motor yön bilgilerini okuma ve kontrol işlevlerini sağlar.
IMotorDriveError	Motor sürücü hatalarının tanımlanması ve raporlanması için kullanılır.

## 7. Test Senaryoları

### Senaryo A: Direction SIL TEST

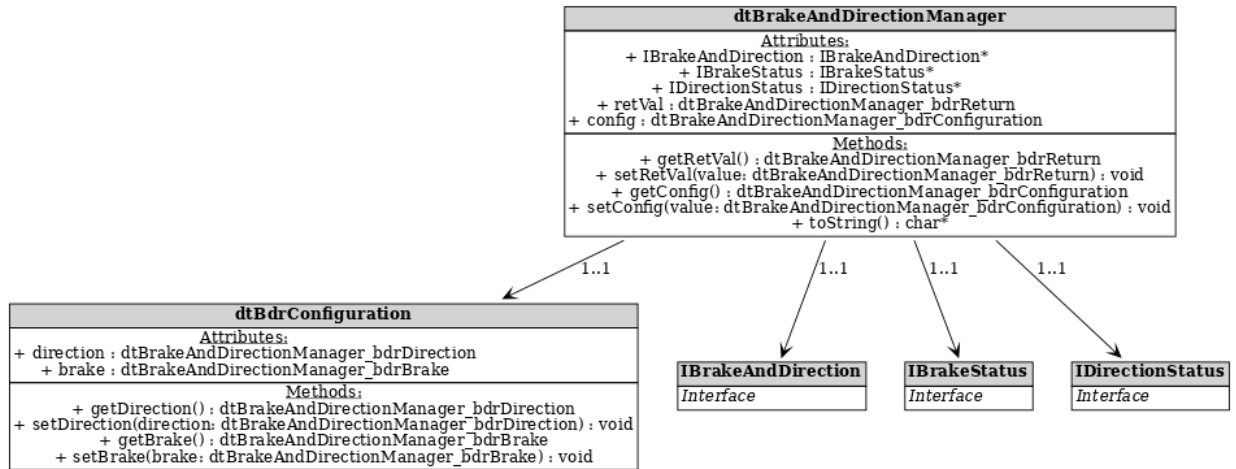
<Gerçekleşebilecek test senaryo açıklaması>

### Senaryo B: Brake SIL TEST

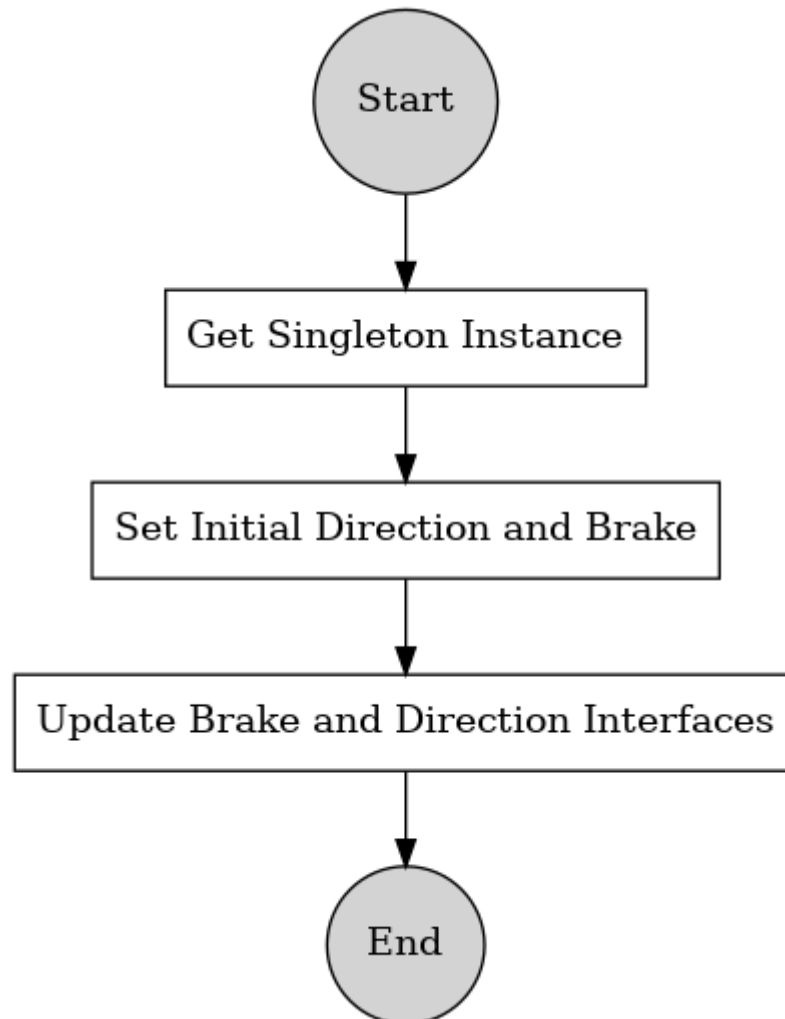
<Gerçekleşebilecek test senaryo açıklaması>

## 8. Görsel Modeller ve Diyagramlar

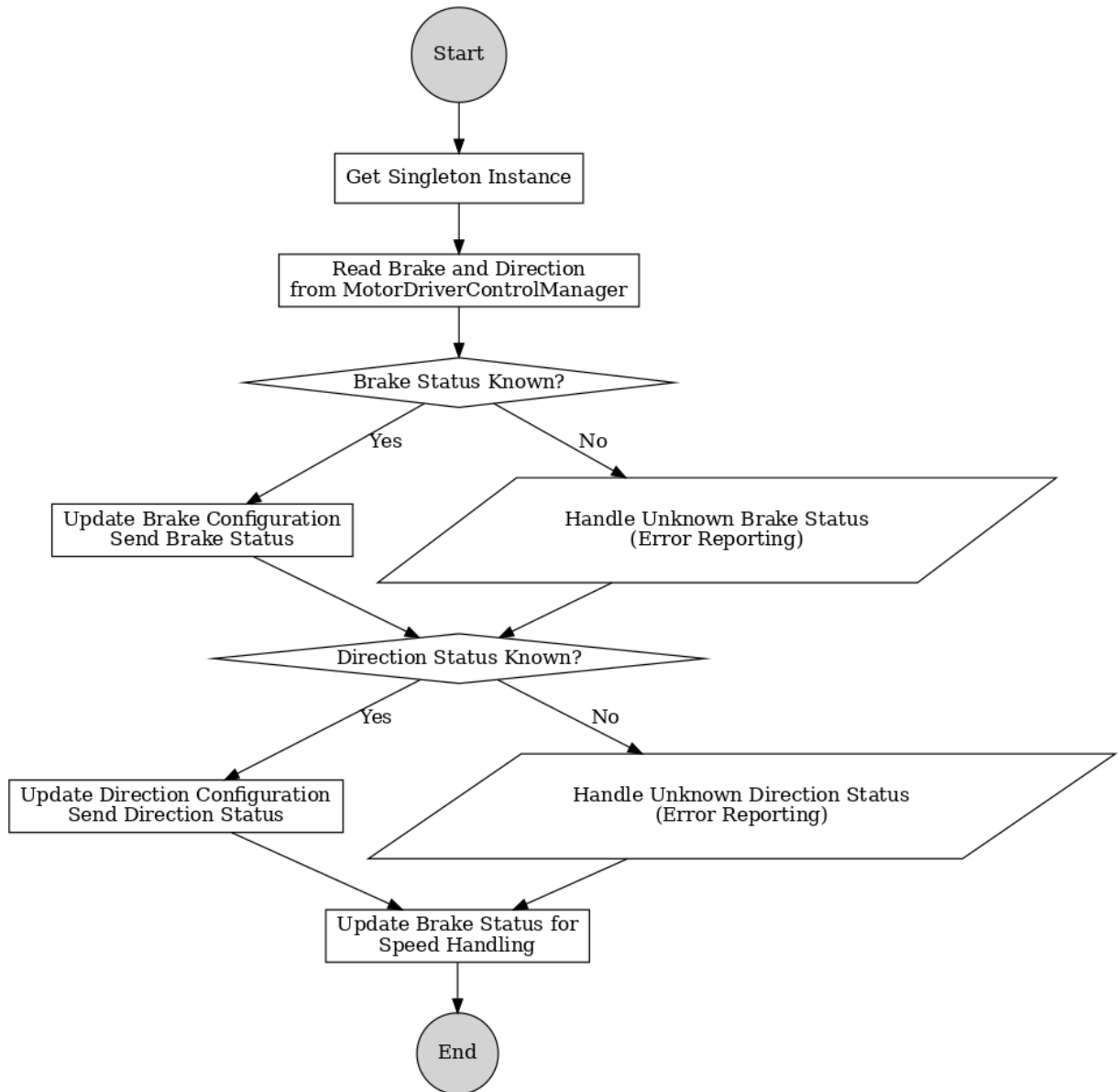
### 8.1 UML Diagram



### 8.2 BrakeAndDirectionManager\_ruInitialisation Activity Diagram



### 8.3 BrakeAndDirectionManager\_ruRefresh Activity Diagram



## 11. Ekler