T.C.

EGE ÜNİVERSİTESİ

Fen Bilimleri Enstitüsü

AKILLI ULAŞIM SİSTEMLERİ İÇİN CİHAZ YÖNETİM ÇERÇEVESİ

Can ÖZ

Danışman: Prof. Dr. N. Yasemin TOPALOĞLU

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Doktora Programı

İzmir

2024

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum "**AKILLI ULAŞIM SİSTEMLERİ İÇİN CİHAZ YÖNETİM ÇERÇEVESİ**" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

07 / 03 / 2024

Can ÖZ

ÖZET

AKILLI ULAŞIM SİSTEMLERİ İÇİN CİHAZ YÖNETİM ÇERÇEVESİ

ÖZ, Can

Doktora Tezi, Bilgisayar Mühendisliği Anabilim Dalı
Tez Danışmanı: Prof. Dr. N. Yasemin TOPALOĞLU
Mart 2024, 94 sayfa

Nesnelerin internetine ait uygulama alanlarından biri olan Akıllı Ulaşım Sistemleri (AUS), trafik güvenliğinin arttırılması, seyahat sürelerinin düzenlenmesi ve enerjinin verimli kullanılarak çevreye verilen zararın azaltılmasını amaçlar. Akıllı ulaşım sistemleri içinde farklı sağlayıcıların bulunması, uygulamaların birlikte çalışabilmesini ve yönetilmesini güçleştirir. Birlikte çalışabilme karmaşıklığını giderebilmek için cihaz yönetimine ait alanların, nesnelerin interneti yaklaşımlarıyla desteklenerek Toplu Taşıma için Bilgi Teknolojileri (*ITxPT*) birliği tarafından donanım, iletişim protokolü ve hizmet düzeyleri olarak standartlaştırılması hedeflenmiş olmasına rağmen pratik geliştirme süreçleri dışında yeteri kadar çalışma yapılmamıştır.

Bu tez kapsamında, toplu taşıma için bilgi teknolojilerini temel alarak ulaşım cihazlarının yönetilmesi için gereken verileri olay, durum ve kural ilişkisiyle birleştirerek verilerin daha anlamlı mesaj paketlerine dönüştürülmesini amaçlayan bir cihaz yönetim çerçevesi geliştirilmiştir. Bu doğrultuda farklı servis sağlayıcılara ait uygulamaların çalıştığı cihaz ağı içerisinde uygulamaların verilerini paylaşarak çalışabilmesi için bir ontoloji hazırlanmış ve uygulama verilerinin cihaz üzerinde kaydedilerek ulaşım sistemine ait durum değişimlerinde olayların oluşturulması sağlanmıştır. Geliştirilen cihaz yöneticisiyle farklı servislerin beraber çalışabildiği, uygulama geliştiriciler için yerel ağ ve sunucuyla haberleşme alt yapısının yalıtıldığı bir çerçeve sağlanmıştır.

Anahtar sözcükler: Nesnelerin İnterneti, Cihaz Yönetimi, Akıllı Ulaşım Sistemleri, Mesaj Odaklı Ara Birim Yazılımları, Birlikte Çalışabilirlik

ABSTRACT

DEVICE MANAGEMENT FRAMEWORK FOR INTELLIGENT TRANSPORTATION SYSTEMS

ÖZ, Can

Ph.D. in Computer Engineering
Supervisor: Prof. Dr. N. Yasemin TOPALOĞLU
March 2024, 94 pages

Intelligent Transportation Systems (ITS) are a subset of Internet of Things (IoT) applications that aim to enhance traffic safety, regulate travel times, and reduce environmental impact by promoting efficient energy use. The presence of multiple providers within ITS complicates the interoperability and management of applications. Although the Information Technology for Public Transport (ITxPT) association has aimed to standardize hardware, communication protocols, and service levels in the field of device management using IoT approaches to overcome interoperability challenges, practical development processes have not received sufficient attention outside of standardization efforts.

In this thesis, a device management framework is developed which aims to transform data into more meaningful message packets by combining the data required for managing transportation devices with event, state, and rule relationships based on information technologies for public transport. To this end, an ontology was developed to enable applications from different service providers to share data and operate within a device network. The framework also provides a layer that isolates communication with the local network and server for application developers, allowing different services to coexist.

Keywords: Internet of Things, Device Management, Intelligent Transportation Systems, Message Oriented Middlewares, Interoperability

ÖNSÖZ

Bu tezde, nesnelerin interneti alanlarından biri olan akıllı ulaşım sistemlerinde cihaz yönetimi üzerine odaklanılmıştır. Akıllı ulaşım sistemleri, nesnelerin internetinin sunduğu imkanlar ile ulaşım sistemlerinin daha güvenli, verimli ve sürdürülebilir olmasını hedefler. Farklı sağlayıcılara ait servislerin bir arada çalışabilmesi ve etkin bir şekilde yönetilebilmesi için ulaşım sistemlerine ait uygulama verilerinin izlenmesi ve birbirleriyle olan paylaşımın sağlanması önemlidir. Bu çalışmada ayrışık veri ve olaylar için mevcut standartlara uyan, ölçeklenebilir, mesaj odaklı bir cihaz yönetici çerçevesi geliştirilmiştir. Cihaz yönetici çerçevesi, toplu taşıma alanındaki bilgi ve iletişim teknolojilerini temel alarak ulaşım cihazlarının yönetimini kolaylaştırmayı ve cihaz üzerinde çalışan uygulamaların beraber çalışabilmesini amaçlamaktadır. Bu doğrultuda ulaşım sistemlerine ait durum değişimleri analiz edilerek cihaz ağı içerisinde bağlama ait yapılandırılmış veriler dağıtılmaktadır. Cihaz ağı içerisinde merkezi olarak konumlanan cihaz yöneticisi, uygulamaların hem cihaz içerisinde hem de sunucu ile olan haberleşmesinde evrensel standartlara uygun olarak nesnelerin interneti protokolleriyle desteklenmiştir. Farklı uygulamaların beraber çalışabildiği, genişleyebilir ve uygulamalara ait verinin izlenebildiği bir çerçeve geliştirilmiştir.

İZMİR

07/03/2024

CAN ÖZ

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

Şekil Sayfa

ŞEKİLLER DİZİNİ (devam)

Şekil Sayfa

ŞEKİLLER DİZİNİ (devam)

Şekil Sayfa

ÇİZELGELER DİZİNİ

Çizelge Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
AUS	Akıllı Ulaşım Sistemleri
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
COAP	Constrained Application Protocol
ETSI	European Telecommunications Standards Institute
GPS	Global Positioning System
HTTP	HyperText Transfer Protocol
IoT	Internet of Things
ITS	Intelligent Transport Systems
ITxPT	Information Technology for Public Transport
LAN	Local Area Network
M2M	Machine to Machine
MOM	Message Oriented Middleware
MQTT	Message Queuing Telemetry Transport
NFC	Near Field Communication
Nİ	Nesnelerin İnterneti
OBU	On Board Unit
RFID	Radio Frequency Identification
TLV	Tag Length Value

WSN	Wireless Sensor Network
UAB	Ulaştırma ve Altyapı Bakanlığı
XMPP	Extensible Messaging and Presence Protocol
VAL	Validator

1. GİRİŞ

Bilgi teknolojisindeki ilerlemeler, gündelik yaşamın her alanına etki etmiştir. 2011 yılında Almanya'da temelleri atılan Endüstri 4.0 (Industry 4.0) kavramı, sanayinin dijitalleşmesini hızlandırmıştır. Endüstri 4.0'ın temel hedefi üretimde verimliliği arttırmaktır. Bu amaçla üretim süreçlerinde kullanılan makinelerin, internet üzerinden birbirleriyle etkileşim içinde olması düşünülmüştür. Algılayıcılar (*sensor*) vasıtasıyla takip edilen durum değişimleri, yorumlanması için internet üzerinden bulut bilişim içerisinde (*cloud computing*) depolanmaktadır. Depolanan bilgilerin işlenmesiyle elde edilen yeni bilgilerin var olan sistemi daha akıllı hale getirerek birbirleriyle iletişim halinde bulunan nesnelerin interneti (*Internet of Things, IoT*) mimarisi tanımlanmıştır (Atzori et al., 2010).

Nesnelerin interneti farklı türlerdeki donanım, yazılım ve iletişim arayüzlerine sahip akıllı nesnelerin internet üzerinden birbirlerine bağlanmasını sağlar. En yalın haliyle nesnelerin interneti, makinelerden, insanlardan veya cihazlardan üretilen verilerin ağ üzerinden farklı sistemlere aktarılması olarak tanımlanmaktadır (Özdoğan, 2017). Uygulayıcılar, uyarıcılar ve birçok gömülü sistem cihazının beraber oluşturduğu bu ağ ile verimlilik, kullanılabilirlik, otomatik güncelleme ve kolay bakım gibi hedeflerle insan yaşamında kolaylıklar sağlanmaktadır (Giusto, 2010). Birbirine internet ağı üzerinden bağlanan cihazlar, akıllı olarak nitelendirilen sistem tanımlarını ortaya çıkartmıştır. Nesnelerin interneti; akıllı evler (Mocrii et al., 2018), bağlı arabalar, akıllı tarım (Lezoche et al, 2020), akıllı sağlık (Ray et al., 2019; Ahmadi et al., 2019) gibi birçok alanda kullanılırken son dönemlerde ulaşım sistemlerini akıllı hale getirmek için de tercih edilmeye başlamıştır. Akıllı Ulaşım Sistemleri (AUS, Intelligent Public Transport) olarak adlandırılan yaklaşım ile donanım ve servis seviyelerinde operasyonların daha verimli yönetilebilmesi ve kolay kontrol edilebilmesi hedeflenmiştir.

Akıllı ulaşım sistemlerinde cihazların yönetimi, bakımı ve farklı tür donanımlarla bütünleşik çalışabilmesi, kurulumdan çalışma anına kadar birçok gereksinimi açığa çıkartır. Bu sistemlerde ürünlerin hesaplama ve saklama kapasitelerinin sınırlı olması ve ağ iletişiminin standart yapıda olmaması temel problemlerdir. Bu sorunları giderebilmek için *ITxPT* (2017) adında toplu taşıma

araçlarına ait sistemleri düzenleyen ve servisleri tanımlayan bir standart ortaya çıkmıştır. Toplu ulaşım için bilgi teknolojileri standardıyla, farklı üreticilerin hazırlamış olduğu servislerin birlikte çalışabilmesi hedeflenmiştir. Böylece, servis sağlayıcıların farklı sistemler için aynı görevleri yapan farklı uygulama geliştirmelerinin önüne geçilmiştir.

Literatürde akıllı ulaşım sistemine ait yapılan çalışmalar, uygulamaların gruplanmasına göre hazırlanmıştır. Bazı çalışmalar uygulamaları servis tiplerine göre tanımlarken, bazı çalışmalar ise araçlar ve altyapı olarak ayrıma gitmiştir. ISO 14813-1 standardı, uygulamaların sınıflandırmasını ele alarak akıllı ulaşım sistemleri için standardı belirlemiştir. Bu doğrultuda alana özgü gerçekleştirimler önem kazanmaktadır. Sefer bilgileri, trafik yönetimi, toplu taşıma servisleri, yolcu bilgilendirme servisleri gibi farklı servis grupları, sunulacak hizmete ait özellikleri belirlemektedir. Alana özgü çalışmalar "Geleceğin Avrupa Otobüs Sistemleri" projelerine (Tozzi et al., 2016; Corazza et al., 2016) dayanan *ITxPT* kapsamında pratik geliştirme süreçleri olarak standartlaşsa da henüz yeteri kadar çalışma yapılmamıştır (Puerta et al., 2018; Corazza et al., 2018).

ITxPT gibi standartlar referans mimari yaklaşımları olarak ele alındığında, akıllı ulaşım sistemlerine ait uygulamaların araç içi uygulamaları, sunucu uygulamaları ve beraber çalışan uygulamalar olarak ayırmaktadır. Bu bağlamda akıllı ulaşım sistemlerinde uyumluluk ortak bir mimari ve veri modeli uygulandığında arttırılabilir. Kritik olmayan uygulamaların dağıtık mimari yaklaşımına, gerçek zamanlı uygulamaların merkezi mimari yaklaşımına göre yönetildiği bir çerçeve tanımı yapılmalıdır.

Bu tez çalışmasında ayrışık veri ve olayların yönetilmesi için ölçeklenebilir, evrensel olarak uygulanabilir; mesaj tabanlı bir cihaz yöneticisi çerçevesi geliştirilmiştir. Bu yönetici ham veri ve olayları alır, bunları uygulamalar tarafından tüketilmek üzere değerli mesajlara dönüştürür.

Geliştirilen çerçeve içerisinde tanımlanan yapılar ile farklı sağlayıcıların gerçekleştirdiği servisler beraber çalışabilmektedir. Bu uygulamaların etkileşim içerisinde olabilmesi için veri temelli iletişim modeli tercih edilmiştir. Nesnelerin

interneti yaklaşımlarına uygun olarak en yaygın kullanılan mesajlaşma biçimi abone/yayıncı modelidir. Ortak kullanılan veriler tanımlandıktan sonra abone/yayıncı mesajlaşma modeliyle gerçek zamanlı paylaşılması sağlanmıştır. Merkezi olarak cihaz yöneticisinde izlenen verilerin, olay ve durum değişimlerine duyarlı olarak daha anlamlı biçimde dağıtılmasıyla gerçek zamanlılık, beraber çalışabilme ve ölçeklenebilirlik kalite özelliklerinde çözüm üretilmesi hedeflenmiştir. Bu doğrultuda temel katkılarımız aşağıda özetlenmiştir.

- Cihaz yönetimi çerçevesi, cihaz yönetimi konusunu, nesnelerin interneti yaklaşımıyla yeniden değerlendirerek, mevcut *ITxPT* mimarisini genişletip, eksik servislerinin gerçekleştirimlerini içermektedir. Bu doğrultuda cihaz yöneticisi, uygulama bağımsız bir ara yazılım olarak hizmet verir, araç ağında dolaşan veriden anlamlı olaylar oluşturur. Bu şekilde, uygulamaların sistem durum verilerini beklemek zorunda kalmadan kendi görevlerine odaklanması sağlanmaktadır.
- Cihaz yöneticisi, veri tabanı desteğiyle uygulamaların yapılandırma verilerini saklar. Böylece uygulamaların belirli yapılandırma verilerini yönetmeleri için uygun ve standartlaştırılmış bir yol sunar.
- Cihaz yönetimi çerçevesinde, akıllı ulaşım sistemlerine ait araçlarda kullanılan verileri gruplayarak, durum değişimlerine duyarlı olaylar için bir ontoloji düşünülmüştür. Bu ontoloji ASN.1 (*Abstract Syntax Notation One*) kodlamasıyla serileştirilerek mesajların veri yükünün küçültülmesi sağlanmış, yeniden kullanılabilirlik ve değişken mesaj boyutu özellikleri desteklenmiştir.
- Farklı üreticilere ait varlıkların beraber kullanılabilmesi ve sistemin yönetilmesi için gerekli olan metotlar tanımlanmış ve bu veriler belirli başlıklar altında paylaşılmıştır. Böylelikle yeni olayların üretilmesi kolaylaşırken verilerin gerçek zamanlı durum ve uygulama verileri ile birleştirilmesi sağlanmıştır.

Tez çalışması süresince tamamlanan yayınlar aşağıda listelenmiştir:

- 2023 IEEE International Smart Cities Conference (ISC2)
  Message Based Terminal Manager for Public Transportation Systems
  (https://ieeexplore.ieee.org/abstract/document/10293500)
  Bucharest, Romania
- *2022 Bilgisayar Bilimleri ve Mühendisliği Dergisi* Nesnelerin İnterneti Yaklaşımıyla Konvoy Araçların Yönetimi, (https://dergipark.org.tr/en/download/article-file/2164030), Cilt-15 Sayı-1
- *2021 Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı (UBMK)* Ulaşım Sistemleri için Mesaj Tabanlı Cihaz İletişim Çerçevesi,
  (https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9558971)
  Ankara
- *2019 Akıllı Sistemlerde Yenilikler ve Uygulamaları Konferansı (ASYU)* Ulaşım Sistemleri için Mesaj Tabanlı Cihaz Yönetim Mimarisi, **İzmir**
- *2016 10. Ulusal Yazılım Mühendisliği Sempozyumu (UYMS),* Gelecek Nesil Gömülü Sistem Uygulamaları için Kullanıcı Etkileşimi Yaklaşımı Önerisi, (http://ceur-ws.org/Vol-1721/UYMS16_paper_33.pdf), **Çanakkale**

Tezin ikinci bölümünde, nesnelerin interneti ve akıllı ulaşım sistemleri tanıtılmıştır.

Akıllı ulaşım sistemlerinde cihaz yönetimi kapsamında literatürde yapılan çalışmalar, tezin üçüncü bölümünde incelenmiştir.

Dördüncü bölümde cihaz yönetim çerçevesinin genel özellikleri açıklanmıştır. Bu bölümde ağ seviyesindeki ve bildirim seviyesindeki gereksinimlere göre ihtiyaç duyulan gerçekleştirimler tanıtılmıştır. Olay yönetimi, servis keşfi, yapılandırma yönetimi, durum yönetimi ve kural yönetimi görevleri anlatılmıştır.

Beşinci bölümde ulaşım standardında belirtilen servisler gerçekleştirilmiş, konvoy araç problemi için hazırlanan düğüm servisi çalışmaları anlatılmıştır. Cihaz yönetim çerçevesinin uygulanması sefer yönetimi örnek olayı ile gerçekleştirilirken çerçevenin sunucu sorguları için desteklediği yaklaşım nesnelerin interneti protokolleriyle karşılaştırılmıştır.

Altıncı bölümde çalışmaların yapıldığı ortam ve çerçevenin kullanılması durumunda elde edilen kazanımlar açıklanmıştır.

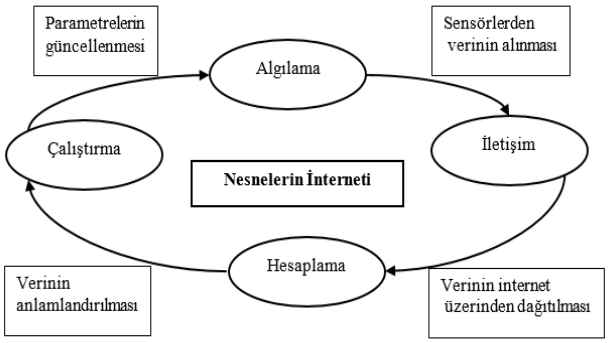
Tezin son bölümünde çalışmadan çıkarılan sonuçlar ve gelecek çalışmalar değerlendirilmiştir.

2. NESNELERİN İNTERNETİ VE AKILLI ULAŞIM SİSTEMLERİ

Nesnelerin İnterneti (Nİ) terimi ilk olarak 1999 yılında "*Procter & Gamble*" firmasından Kevin Ashton tarafından yapılan bir sunumda internetin her yerde bulunan algılayıcılar aracılığıyla fiziksel dünyaya bağlandığı bir sistemi tanımlamak için kullanmasıyla ortaya çıkmıştır (Ashton, 2009). Bilgi teknolojilerindeki son gelişmelerle birlikte nesnelerin interneti gündelik yaşantımıza oldukça hızlı bir şekilde girmiştir. Bilgisayar sistemlerinin internete bağlanması sonrasında mobil cihazlarla her alanda kullanılan internet, nesnelerden oluşan sistemlerin birbirlerine bağlanmasına olanak sağlamıştır. Nesnelerin interneti içinde mobil cihazlar, ev aletleri, araçlar, birbirinden farklı algılayıcılar ve işleticiler gibi çeşitli nesneler sayılabilir. Bu nesneler güncel iletişim teknolojileriyle internete bağlanırken kendi aralarında veri paylaşımında da bulunabilmektedir.

2.1 Nesnelerin İnternetine Genel Bakış

Nesnelerin interneti cihazları ve algılayıcıları geleneksel bilgisayarların aksine belirli bir amaca yönelik fonksiyonlara sahip olurlar. Desteklenen haberleşme protokolleriyle elde ettikleri verinin başka sistemler tarafından kullanılabilmesini sağlarlar. Şekil 2.1' de nesnelerin interneti ve akıllı nesneler için çalışma prensibi gösterilmiştir (Abdmeziem et al., 2016).



Şekil 2.1 Nesnelerin interneti cihazları işleme prensibi (Abdmeziem et al., 2016).

Stojkoska ve Trivodaliev (2017) bütün çevremizin birbirleriyle iletişim ve hesaplama yeteneklerine sahip akıllı objelerden oluştuğunu belirtmektedir. Algılayıcılardan ev uygulamalarına ve endüstriyel cihazlara kadar ulaşan bu karmaşık ortamdaki cihazlar nesnelerin interneti şemsiyesi altında toplanmaktadır. Vermesan ve Friess (2013), nesnelerin internetini ayrı cinsten nesnelerin herhangi bir zamanda ve yerde internet üzerinden bağlanabilmesine izin veren bir ağ olarak tanımlamıştır.

2.1.1 Nesnelerin internetinde temel bileşenler

Nesnelerin interneti için farklı bakış açıları içeren çeşitli çalışmalar, farklı tanımlamalar yapmıştır:

- Uluslararası Telekomünikasyon Birliğine (*International Telecommunication Union*, ITU) göre nesnelerin interneti; alandan, yerden ve zamandan bağımsız şekilde başka cihazlarla bağlantı kurabilen herhangi bir şeydir (ITU-T Y. 2060).
- Nesnelerin interneti; merkezi olmayan ve birbirlerinden bağımsız şekilde çalışabilen cihaz sisteminin algılama, işleme ve ağ yetenekleriyle donatılmasıdır (Kortuem et al., 2009).
- Nesnelerin interneti; düşük hesaplama kapasitesiyle insan etkileşimine gerek duymadan ürettikleri verileri işleyen ve transfer edebilen sensörler ve/veya işleticileri birbirine bağlayan bir altyapıdır (Dorsemaine et al., 2015).
- Nesnelerin interneti; kimlikleri, fiziksel özellikleri ve sanal kişilikleri olan ve akıllı arayüzlerle bilgi ağına sorunsuz şekilde bağlanabilen standart ve birlikte çalışabilen protokolleri temel alan kendi kendini yapılandırma yeteneklerine sahip dinamik bir küresel ağ altyapısıdır (Vermesan et al., 2011).
- Nesnelerin interneti; kimliğe, fiziksel ve sanal alanlara sahip farklı "şey"lerin oluşturduğu standart ve birlikte çalışabilen protokollere dayalı olarak kendi kendini yapılandırma yeteneklerine sahip, değişken yoğunluklu ve bağlanabilirliğe sahip küresel bir ağ ve hizmet altyapısıdır (Minerva et al., 2015).

Hesaplama, iletişim ve kimliklendirme yetenekleriyle zenginleşen küçük ölçekli elektronik cihazlar, nesnelerin interneti kavramını somut bir hale getirmiştir. Nesnelerin interneti sistemlerinde üç özellik ön plana çıkmaktadır:

- Algılama: sensörler ve kablosuz algılayıcılar ile herhangi bir yerden herhangi bir zamanda nesnelere ait bilgiler elde edilebilir. Çevremize gömülen bu donanımlarla, insanların gerçek dünyayla etkileşime girebilmesi fırsatı doğmuştur.
- Güvenli Aktarım: veriler genellikle kablolu veya kablosuz ağlar üzerinden aktarılır. Kablolu ağlar, daha yüksek güvenilirlik ve güvenlik sunarken, kablosuz ağlar daha esnek ve kurulumu daha kolaydır.
- Akıllı İşleme: ortaya çıkan veri bulut sistemleri gibi akıllı işleme teknolojileriyle nesnelerin interneti uygulamalarını desteklemektedir. Verinin saklanma ve işlenme yoğunluğuna göre zaman geçtikçe farklı yaklaşımlar ortaya çıkmıştır.

2.1.2 Nesnelerin interneti sistemleri uygulama alanları

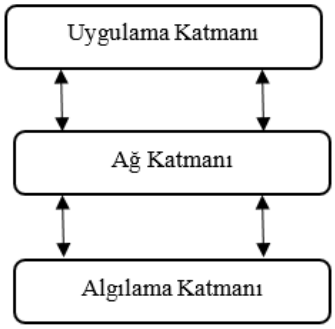
İnternete bağlanan milyarlarca cihaz ile nesnelerin interneti gündelik yaşama dokunan birçok uygulama alanını ortaya çıkartmıştır. Bunların en önemlileri aşağıda listelenmiştir. Şekil 2.2'de nesnelerin interneti uygulama alanları görülmektedir.

- Akıllı Şehirler: algılayıcıların, ağ yapılarının ve bulut sistemlerinin kullanımıyla vatandaşlar çevre hakkında gerçek zamanlı veriye ulaşabilmektedir.
- Akıllı Evler: eve kurulan uygun cihazlara mobil cihazlar üzerinden erişerek uzaktan ev yönetiminin yapılabilmektedir.
- Bağlı Araçlar: internete bağlanabilen araçların çalışma verileri sunucuya gönderilerek uzaktan takibi ve durum raporları oluşturulabilirken, birbirleriyle konuşabilen araçların belirli senaryolarda kaza önlemesi veya alarm göndermesi sağlanmaktadır.

Şekil 2.2 Nesnelerin İnterneti uygulama alanları (Erdoğmuş, 2016).

2.1.3 Nesnelerin interneti mimarileri

Geçtiğimiz yıllar boyunca birçok araştırmacı tarafından hem akademide hem de endüstride kabul edilen birbirinden farklı nesnelerin interneti mimari tanımları yapılmakla beraber ortak bir tanıma ulaşılamamıştır. Şekil 2.3'te görülen üç katmanlı üst seviye mimari en çok kabul edilen mimaridir (Wu et al., 2010). Çalışma içerisinde bu katmanlar algılama, ağ ve uygulama olarak adlandırılmıştır.



Şekil 2.3 Üç katmanlı nesnelerin interneti mimarisi (Wu et al., 2010).

Algılama Katmanı: çevremizdeki fiziksel özelliklerin algılanması görevindedir. RFID, GPS, NFC, WSN gibi teknolojilerini temel alır. Bu katman ayrıca bilginin taşınabilmesi için dijital sinyallere çevrilmesini de içerir. Doğrudan hissedilmeyen nesneler için mikroçipler kullanılarak bu dijitalleşme sağlanır (Sethi and Sarangi, 2017).

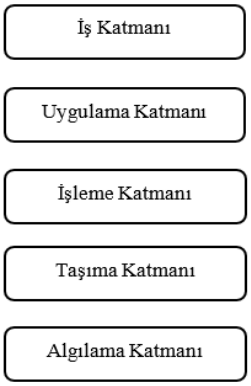
Ağ Katmanı: algılama katmanında elde edilen verinin işlenip uygulama katmanına gönderilmesinden sorumludur. Kablolu ve kablosuz birçok ağ teknolojisi (3G/4G/WiFi, Bluetooth, vb.) bu katmanda devreye girer. Çok fazla verinin taşınmasını gerektirdiğinden saklama ve işleme karmaşıklığını çözmek için bulut bilişim tercih edilebilir. Birçok büyük bilişim firmasının bulut çözümleri birçok uygulamada kullanılmaktadır.

Uygulama Katmanı: ağ katmanında işlenen verinin kullanıldığı yerdir. Akıllı ulaşım, akıllı ev, akıllı tarım gibi birçok farklı uygulama alanı bulunur. Uygulama geliştiriciler için mimarinin ön yüzüdür.

Uygulama geliştirenleri teknik katmanların karmaşıklığından ayırdığı için nesnelerin interneti mimarilerinin ana parçası ara yazılım (*middleware*) çözümleridir. Avrupa araştırma projesi olan HYDRA servis odaklı mimariyi (SOA, *Service Oriented Architecture*) temel alarak cihazlara ağ (*web*) servis komutlarıyla erişilmesini sağlayan bir ara yazılım yaklaşım örneğidir (Eisenhauer et al., 2010). Başka bir çalışmada cihazları uygulama katmanına taşıyan bir yaklaşım olarak nesnelerin ağı (WoT, *Web of Things*) tanımı ortaya atılmıştır (Guinard et al., 2010). Çalışmalarında olay güdümlü iletişim örüntüsü sağlamak için gerçek zamanlı ağ teknolojilerinin abone/yayımcı yaklaşımıyla http üzerinde uygulanmasını tartışmışlardır.

Cihazların ve uygulamaların belirli senaryolara göre tasarlanması farklı nesnelerin interneti uygulama alanları için beraber çalışmayı sağlayacak referans bir mimari eksikliğini ortaya çıkartmıştır. Bu problemi gidermek için ISO/IEC 30141(2018) ile kapsamlı bir kavramsal model tanımlanmıştır. Kavramsal model bakış açılarına göre fonksiyonel, sistem, bilgi, iletişim ve kullanım olarak mimari bileşenlerine parçalanmıştır. Bauer ve arkadaşları (2013), nesnelerin interneti referans mimari modeli (ARM, *IoT Reference Architecture Model*) tanımıyla beş alt model tanımında bulunmuştur. Böylece tanımlanan basit referans modelleri kullanılarak uygulama alanına özelleşen bir mimarinin nesnelerin interneti uygulama alanlarında kullanılması hedeflenmiştir. Şekil 2.4'te beş katmanlı nesnelerin interneti mimarisi görülmektedir. Taşıma katmanı algılama katmanından gelen verinin taşınması için kullanılacak kablolu ve kablosuz protokolleri

içermektedir. İşleme katmanı ara birim yazılım katmanıdır. Taşıma katmanından gelen büyük verinin kaydedilmesi anlamlandırılması için üst seviye veri işleme uygulamalarına sahiptir.



Şekil 2.4 Beş katmanlı nesnelerin interneti mimarisi (Bauer et al., 2013).

2.2 Nesnelerin İnterneti Protokolleri

Nesnelerin interneti uygulamalarında http, mqtt, coap, xmpp, mqtt ve websocket protokolleri yaygın olarak kullanılmaktadır.

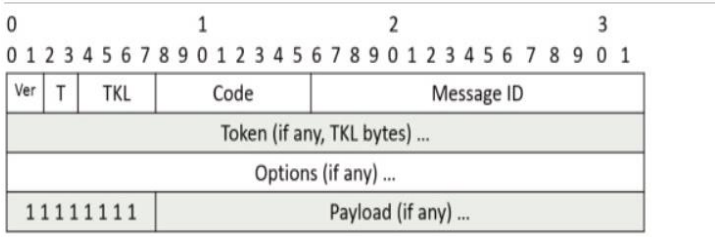
2.2.1 HTTP (HyperText Transfer Protocol)

HTTP, veri iletişimin istek/cevap mesajlarıyla sağlandığı uygulama katmanı protokolüdür (RFC2616). Web dünyasının temel taşı sayıldığı için farklı cihaz ve platformlar tarafından geniş destek görmektedir. Çeşitli uygulamalarla kolaylıkla entegre edilebilmesi nedeniyle nesnelerin interneti uygulamalarının daha geniş bir ekosistemle uyum sağlayabilmesine yardımcı olur. Ancak, sınırlı kaynaklara sahip küçük nesnelerin interneti cihazları için yüksek veri yükü ve düşük gecikme süresi nedeniyle uygun görülmez (Lombardi et al., 2021).

2.2.2 CoAP (Constrained Application Protocol)

CoAP, *IETF* (Internet Engineering Task Force) tarafından, Http'nin kısıtlı cihazlara göre özelleştirilen uygulama katmanı protokolüdür (Shelby et al., 2014). REST (*Representational State Transfer*) mimarisini temel alarak *UDP* üzerinden çalışmaktadır. Http benzeri istek/cevap örüntüsünde olmasına rağmen *UDP* tercihi

ile sunucu ve müşteri arasında iletilen paket yükü azaltılmıştır. Ancak *UDP* kullanımının mesaj gönderimlerini garanti etmemesinden dolayı açığa çıkan güvensiz durumu bakımından yeniden gönderim ve kaynak keşfetme mekanizmaları eklenmiştir (Silva et al., 2021). Şekil 2.5, *CoAP* için mesaj formatı görülmektedir. Her mesaj sabit uzunluklu 4 bayt ile başlar. İsteğe göre protokol seçenekleri eklenmektedir (Gupta. P., 2021).

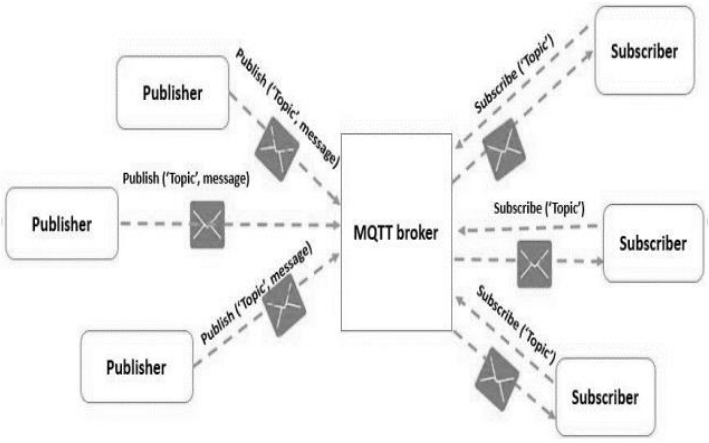


Şekil 2.5 *CoAP* için mesaj formatı (Gupta, 2021).

2.2.3 MQTT (Message Queue Telemetry Transport)

Nesnelerin interneti cihazlarında gerçek zamanlı mesaj değişimini destekleyen Http'den daha yalın kabul edilen yayıncı/abone protokolüdür (Locke, 2010). Gömülü cihazlar gibi kısıtlı donanımlar için *IBM* tarafından 1999 yılında geliştirilmiştir. Protokol tasarımı ağ bant genişliğini en aza indirmeye odaklanırken, mesaj gönderimi alt yapısı *TCP* üzerine kurulduğu için güvenilir mesaj iletimini hedeflemiştir.

Protokolün merkezinde bir aracı (*broker*) bulunur. Yayıncı ve abone arasında belirlenen başlıklar üzerinden mesaj iletimi yapılması sağlanır. Şekil 2.6'da *MQTT* yayıncı abone ve aracı arasındaki ilişki görülmektedir.



Şekil 2.6 *MQTT* yayıncı abone aracı yapısı (Cherradi et al., 2016).

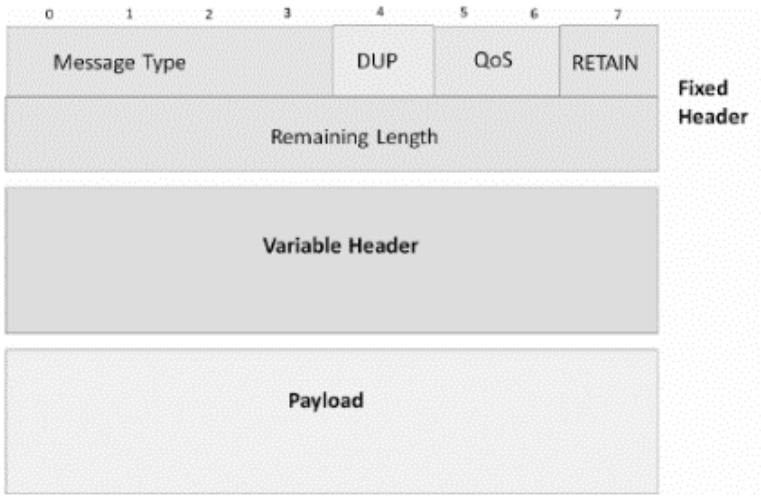
MQTT, mesaj iletimi için servis kalitesini *QoS* (*Quality of Services*) başlığı altında 3 tipte tanımlar.

- *QoS1* (At most once): Mesajlar saklanmaz ve cihazların kapanıp açılması durumunda yeniden gönderilmez. Mevcut TCP protokolü seviyesinde hizmet sağlarken mesaj kayıpları yaşanabilir.
- *QoS2* (At least once): Gönderici mesajına aracıdan cevap alana kadar mesajı saklar. Mesajların iletilmesi garanti edilirken aynı mesajın birden fazla gönderilmesi önlenmez.
- *QoS3* (Exactly once): İstemci ile aracı asında yapılan 4 aşamalı el sıkışma ile mesajın sadece bir kez alıcıya iletilmesi garanti edilir.

Mesaj başlıklarını yöneten aracı üzerinden yayınlayıcı ve aboneler mesaj üretip mesajları dinleyebilirler. Protokol spesifikasyonunda bulunan yardımcı işaretler uygulama geliştiriciler için ilk bağlantının kurulmasına ve bağlantı kopması durumunda yapılacak işlemlerin belirlenmesini kolaylaştırmaktadır.

- *Will Flag*: İstemcinin aracıya bağlantı kopması durumunda göndermek üzere belirlediği başlık ve mesajı belirtmek için kullanılır. İstemci çevrim dışı olduğu zaman belirtilen mesajı bekleyen alıcılara aracı tarafında bekletilen mesaj iletilmiş olur.

- *Clean Session*: Bağlantının sürekli koptuğu ortamlarda başlık üzerinde geçmişte gönderilen işaretçilerin kaldırılmasını sağlamak amacıyla kullanılmaktadır.
- *Retain*: Alıcının bir başlık altındaki mesajları dinleme isteği sonrasında aracının başlığa ait son mesajı iletmesi için kullanılır.

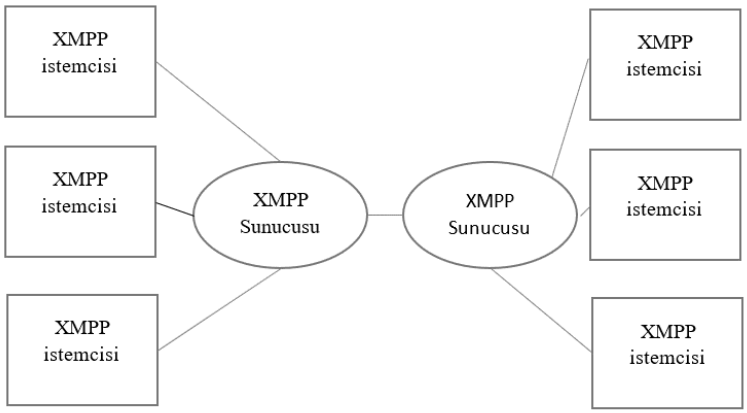


Şekil 2.7 *MQTT* mesaj formatı (Liu et al., 2020).

MQTT mesajının formatı Şekil 2.7'de yer almaktadır. *MQTT'* de her mesaj 2 byte sabit uzunluklu başlığa sahiptir. Değişken uzunlukta olabilen mesaj başlığı ise mesaj tipine göre kullanılmaktadır. Mesajın içeriği ise (*payload*) imaj, karakter ve ikili (*binary*) veri türünde gönderilebilir.

2.2.4 XMPP (Extensible Messaging and Presence Protocol)

XMPP, 1999 yılında *Jabber* yazılım birimi tarafından tanıtılan internet tabanlı platformlar üzerinde dağıtık bir şekilde çalışan bir iletişim protokolüdür (Bhowmik and Riaz, 2023). Veri iletişimi için *XML* kullanarak ses ve video taşınmasına da olanak sağlamaktadır. Sunucu ile istemciler arasındaki etkileşim Şekil 2.8'de görülmektedir.

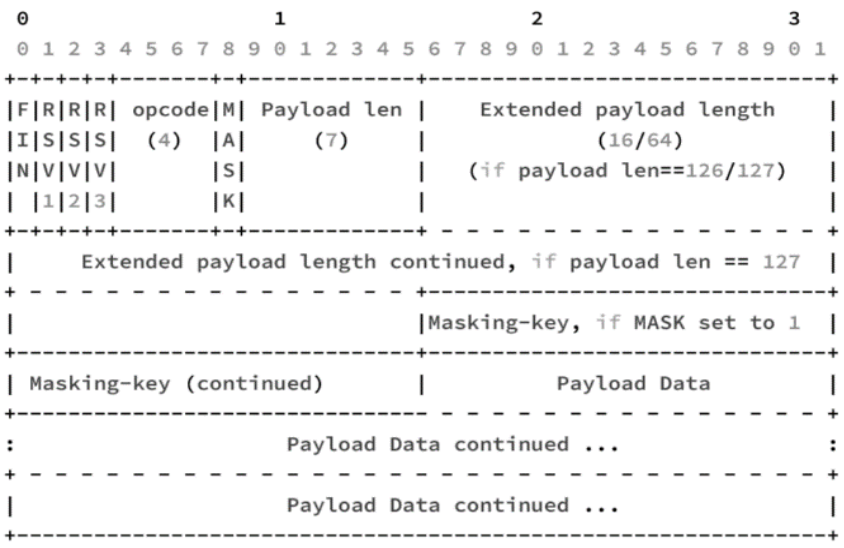


Şekil 2.8 XMPP mimarisi

XMPP' in, açık iletişim teknolojilerini desteklemesi ve dağıtık mimari de olması anlık mesajlaşma uygulamaları için tercih edilmesini sağlar. *MQTT*' de bulunan servis kalitesinin *XMPP*' de olmaması sebebiyle makinalar arası iletişim için uygun değildir.

2.2.5 WebSocket

WebSocket, açık olan bir *TCP* bağlantısı üzerinden çift yönlü mesaj gönderebilme işlemini yapabilen bir protokoldür (Fette and Melnikov, 2011). Protokol spesifikasyonu mevcut ağ teknolojilerini temel alarak istemci ile sunucu arasında açık bir bağlantının kurulmasına dayanmaktadır. İletişim için hazır olan bağlantı üzerinden sunucu veya istemci istediği anda mesaj gönderirken http protokolünde olduğu gibi gönderilen mesajlara cevap verme zorunluluğu bulunmamaktadır. Günümüzde sunucu servisleriyle hızlı etkileşim gösteren tarayıcılar ve mobil uygulamalar için kullanılmaktır.



Şekil 2.9 *WebSocket* mesaj formatı (Fette and Melnikov, 2011).

WebSocket mesaj formatı Şekil 2.9'da görülmektedir. Önemli yapı taşlarına ait açıklamalar aşağıda listelenmiştir:

- *FIN*: Mesajın son parça olduğunu gösterir.
- *OPCODE*: Mesajın tipini gösterir.
- *PAYLOAD LENGTH*: mesajın uzunluğunu gösterir.
- *PAYLOAD*: Gönderilen mesaj verisini gösterir.

WebSocket protokolü, bağlantı kurulumu sırasında el sıkışma ve veri aktarımı olarak iki aşamaya sahiptir (Hribernik and Kos, 2020). Bağlantı kurulması *Http* komutlarıyla başladığı için mevcut altyapılarla uyumludur. Bağlantı istemcinin "*HTTP GET*" sonrasında "*Upgrade*" isteği göndermesine sunucunu "*101 Protocol Change*" cevabı vermesiyle mevcut bağlantının kapatılıp yeni protokole geçilmesiyle kurulur. Bağlantı kurulumundan sonra istemci ile sunucu arasında çift yönlü haberleşmenin başladığı veri aktarımı aşamasına geçilir. Veri aktarımı mesaj paketleri üzerinden Çizelge 2.1'de açıklanan başlık tipleri ile sağlanmaktadır.

Çizelge 2.1 *WebSocket* komutları.

KOMUT	AÇIKLAMA
0	Mesaj paketinin devam ettiğini gösterir.
1	Mesaj paketinin UTF-8 karakter içeriğini gösterir.
2	Mesaj paketinin ikili veri iletimini gösterir.
8	Bağlantının kapatılması isteğini gösterir.
9	Bağlantının açık olduğunun sorgulanmasını sağlayan Ping görevini belirtir. Alıcı tarafın pong göndermesi beklenir.
10	Bağlantının açık olduğunun sorgulanmasını sağlayan Pong görevini belirtir. Ping karşılandığı zaman bu komutla cevap gönderilmelidir.

WebSocket iletişimi özellikle tekrar eden mesajlar düşünüldüğünde, *HTTP* kullanımına kıyasla hem mesaj boyutu hem de hız açısından oldukça verimlidir. *HTTP*, bir istek için 8 KB kullanılırken, *WebSocket* kullanımında ilk bağlantıdan sonra her mesajda sadece 2 byte harcanır.

2.3 Endüstriyel Nesnelerin İnterneti Mimarisi

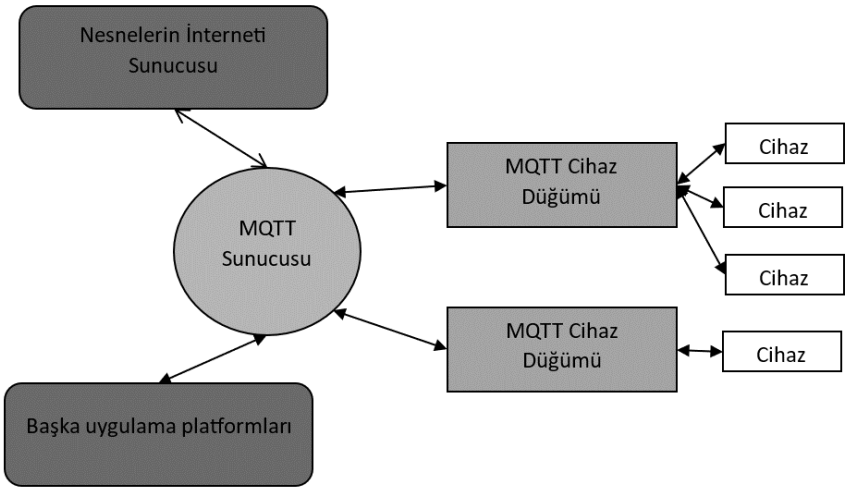
Endüstriyel Nesnelerin İnterneti (*Industrial Internet of Things, IIoT*) mimarisinin özel ihtiyaçlarını karşılamak amacıyla *Eclipse, MQTT* protokolünde kullanılan başlıkların yapısını, mesaj yüklerini ve iletişim parametrelerini standartlaştıran, *SparkPlug* adında bir spesifikasyon geliştirmiştir (Ramachandran, 2022). Mesaj paketi şablonu ve açıklamaları aşağıda listelenmiştir:

namespace/group_id/message_type/edge_node_id/[device_id]

- *Namespace*: Uygulanan spesifikasyon versiyonuna bağlı olarak "spAv1.0" veya "spBv1.0" şeklinde belirtilir.
- *Group_id*: Uç cihazların gruplanması için herhangi bir UTF-8 karakter dizisi olabilir.
- *Message_type*: Mesaj yükünün nasıl işleneceğini belirtir. Olası mesaj türleri şunlardır:

- *NBIRTH* – MQTT Uç Düğümler için Doğum Sertifikası
- *NDEATH* – MQTT UD için Ölüm Sertifikası
- *DBIRTH* – Cihazlar için Doğum Sertifikası
- *DDEATH* – Cihazlar için Ölüm Sertifikası
- *NDATA* – Düğüm veri mesajı
- *DDATA* – Cihaz veri mesajı
- *NCMD* – Düğüm komut mesajı
- *DCMD* – Cihaz komut mesajı
- *STATE* – Kritik uygulama durumu mesajı

Üretim sahasında bulunan bazı cihazların özel protokoller kullanması veya analog sinyaller göndermesi uç düğümler (*Edge of Nodes, EoN*) adı verilen cihazların kullanımını gerektirir. Uç düğüm cihazları, özel protokol verilerini ve analog sinyalleri *MQTT SparkPlug* veri türüne dönüştürür. *SparkPlug* uyumlu cihazlar doğrudan kendi tanım bilgileri üzerinden mesaj yayınlarken, modern protokoller üzerinden iletişim kuramayan eski makineler, uç düğüm cihazları aracılığıyla endüstriyel nesnelerin interneti ağına dahil edilebilir. Şekil 2.10'da tanımlanan elamanların üst seviye etkileşim şeması görülmektedir.



Şekil 2.10 *MQTT Sparkplug* üst seviye etkileşim şeması (Ramachandran, 2022).

SparkPlug spesifikasyonunda belirtilen mesaj tipleri ve bu mesajlara ait veri yüklerinin formatı cihaz katmanından uygulama ve sunum katmanına taşınan verilerin standartlaşması bakımından aşağıda listelenen avantajları kazandırır:

- Cihazların otomatik keşfi: Yeni cihazların ağda kolayca bulunabilmesini sağlar.
- Sürekli oturum farkındalığı: Cihazların bağlantı durumlarının takip edilmesini mümkün kılar.
- Belirleyici yük yapısı: Cihazlardan gelen verilerin anlaşılmasını ve işlenmesini kolaylaştırır.

2.4 Mesaj Odaklı Ara Birim Yazılım Mimarisi

Mesaj odaklı aracı birim yazılım mimarisi (*Message Oriented Middleware*, MoM), farklı uygulamaların birbirleriyle etkin bir şekilde iletişim kurması için güçlü bir altyapı sunan bilinen bir teknolojidir. İlk örnekleri internet çağının başlarında ortaya çıkmıştır. Banavar ve arkadaşları (1999), aracı yazılımın bağımsız uygulamaları birleştirmedeki yeteneklerini vurgulamıştır. Ara yazılımlar ile farklı bileşenlerin farklı yazılım teknolojileri kullanması sonucu ortaya çıkan heterojenlik ve karmaşık entegrasyon problemleri, hazırlanan ek soyutlama katmanıyla giderilir (Bernstein, 1996; Bandyopadhyay et al., 2011).

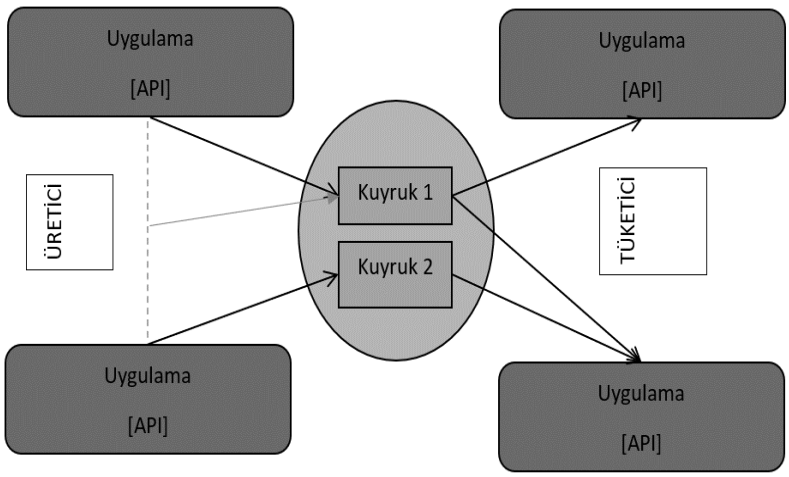
2.4.1 Yapı Taşları

Mesaj odaklı ara birim yazılım mimarisi içerisinde Şekil 2.11'de görülen temel olarak üç alt sistem bulunur (Chen et al., 2017; Yongguo et al., 2019):

- Mesajlar: Aracı yazılım düğümleri arasında veri paketleri olarak kullanılırlar. Bildirimler, olaylar, daha fazla veri talepleri veya hatta ikili veriler şeklinde olabilirler. Mesaj boyutu, aracı yazılım tarafından kısıtlanmadıkça herhangi bir sınır yoktur. Ağ üzerinden iletim için mesajların ağ paketi yüklerine bölünmesi aracı yazılımın sorumluluğundadır.
- Mesaj Kuyrukları: Mesajlar, düğümler arasında, veri ve yürütme akışını birbirinden ayıran aracı mesaj kuyrukları yardımıyla iletilir.

İşlenmeyi bekleyen mesajların sırasını tutarlar ve mesajlaşan taraflar arasında asenkron iletişim sağlarlar.

- Mesaj Aracısı: Merkezi bir sunucu olan mesaj aracısı, taraflar arasında mesaj alışverişini koordine eder. Aracının kendisine ait kuyrukları vardır, ancak ek olarak mesaj doğrulama, çeviri ve yönlendirme de yapar.



Şekil 2.11 Mesaj odaklı ara birim yazılımları genel mimarisi (Yongguo et al., 2019).

Ara birim yazılımlar, asenkron mesajlar aracılığıyla gevşek bağlı dağıtılmış yazılımları mümkün kılarken yayıncı/abone şeması bakımından çeşitli avantajlar sunar (Eugster, 2003):

- Eş zamanlama ayrılması: Mesaj göndericisi, mesajın geri dönüşünü beklemeye ihtiyacı olmadığı için dinleyici taraftan bağımsız olarak işine devam edebilir.
- Zaman ayrılması: İletişim kuran tarafların, mesaj alışverişine katılmak için aynı anda aktif olmaları gerekmez.
- Mantık ayrılması: Bilgi alışverişi için birbirlerinin yazılım yöntemlerini bilmeleri gerekmez.
- Mekân ayrılması: İletişim kuran tarafların mesajlaşması için konumlarının aynı olması gerekmez. Tarafların sadece mesaj aracısının konumunu bilmesi yeterlidir.

2.4.2 Ara birim yazılım mimarilerindeki problemler

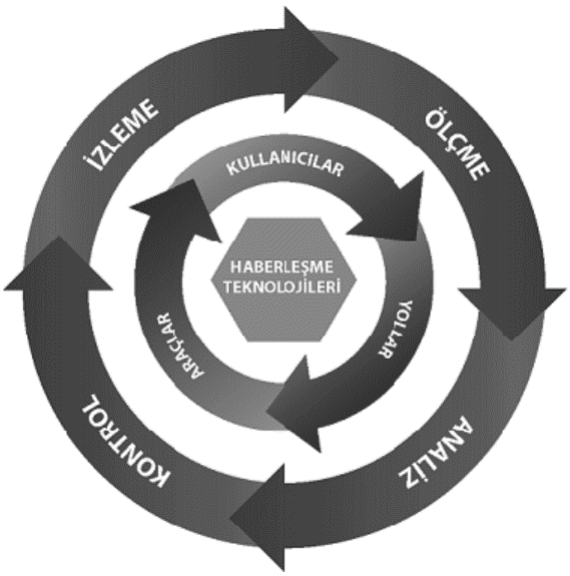
Nesnelerin interneti uygulamalarının farklı yeteneklere ve altyapılara sahip donanımlarla etkileşim kurabilmesi için haberleşme yaklaşımları, veri taşınması ve servislerin yetenekleri bakımından literatürde birçok mimari üzerine çalışılmıştır (Chaqfeh and Mohamed, 2012; Goovaerts, 2016; Ge et. al, 2017). Bu çalışmalarda tanımlanan problemler aşağıda özetlenmiştir:

- Beraber çalışabilme: Birçok heterojen cihazın birlikte iletişim kurması ve bilgi alışverişi yapması gerektiğinden nesnelerin interneti için ara yazılım katmanı önemlidir.
- Ölçeklenebilirlik: Nesnelerin interneti uygulamalarında desteklenmesi beklenen çok sayıda cihaz olduğundan, güvenilir bir ara katman gereklidir ve yeni cihazların eklenmesini yönetmelidir.
- Dinamik altyapı: Kısıtlı kaynaklara sahip olan birçok cihazı içermesi sebebiyle ağ içindeki mobil düğümler istedikleri zaman ayrılabilir veya katılabilirler. Nesnelerin interneti ara yazılım katmanı, ağın dinamik özelliğini yönetmek için esnek bir şekilde tasarlanmalıdır.
- Soyutlama sağlama: Akıllı ortamlar için ideal bir ara katman, heterojen cihazlar, arayüzler, veri akışı, fiziksellik ve geliştirme süreci bakımından birçok düzeyde soyutlama sağlamalıdır.
- Gerçek zamanlılık sağlama: Ulaşım veya sağlık gibi birçok nesnelerin interneti uygulaması için ortamlardan alınan veriden oluşturulacak bilgilerin gerçek zamanlı dağıtılması kritiktir.

Tanımlanan problemler arasından tez kapsamında gerçek zamanlılık, beraber çalışabilme ve ölçeklenebilirlik başlıklarında çözüm üretilmesi hedeflenmiştir.

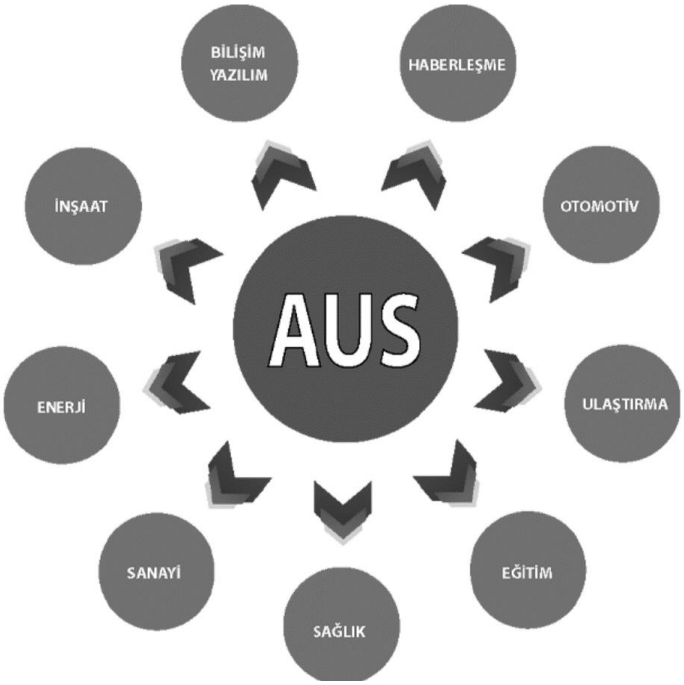
2.5 Akıllı Ulaşım Sistemleri

20. yüzyılın son çeyreğinde Akıllı Ulaşım Sistemleri (AUS) kavramı ortaya çıkmıştır (Alam et al., 2016). *AUS* içerisinde ulaşım yönetimi, kontrolü, altyapısı ve operasyonları gibi farklı alanlar bulunmaktadır. Günümüzde birçok şehirde devreye giren uygulamalarla ulaşımda güvenliğin arttırılması, trafiğin kontrol altına alınması, hava kirliliğinin takip edilmesi ve Şekil 2.12'de görüldüğü gibi ölçme ve izleme süreçleriyle kontrol altına alınması hedeflenmektedir.



Şekil 2.12 Akıllı ulaşım sistemleri yaşam döngüsü (UAB, 2023).

Teknolojinin ilerlemesiyle gündelik yaşantımızda kolaylıkla ulaşabildiğimiz güçlü hesaplama aygıtları, algılayıcılar, konum sistemleri, haberleşme teknolojileri ve veri işleme yaklaşımları ulaşım sistemlerinde de kullanılarak bütünüyle bir değişimi başlatmıştır. *AUS* ile karayolu, denizyolu, havayolu ve demiryolu sistemlerinde servis tanımları hazırlanmıştır. Bu sistemlerde tam anlamıyla veri akışı yönetimi; gerçek zamanlı olarak araçların yoğunluğunun takip edilmesi, hızların takibi ve araçların birbirine olan uzaklığının takibi ile sağlanabilir (Korablev et al., 2021). Akıllı ulaşım sistemleri uygulamalarının etkileşim içerisinde olduğu sektörler Ulaştırma ve Altyapı Bakanlığı tarafından tanımlanmıştır (Şekil 2.13).

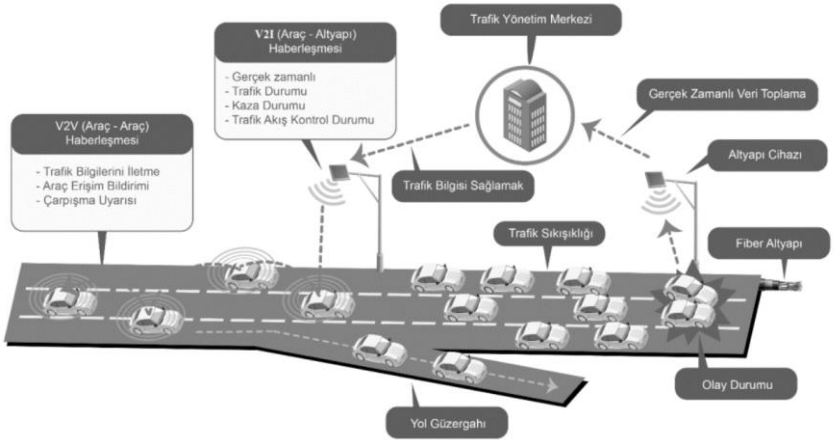


Şekil 2.13 Akıllı ulaşım sistemlerinin etkileşimindeki sektörler (UAB, 2023).

AUS, yolların, araçların ve insanların bağlandığı bütünleşik bir sistemdir. Doolan ve Muntean'a (2016) göre sadece trafik konularıyla değil aynı zamanda sosyal ve ekonomik konularla da ilgilidir. Akıllı ulaşım sistemlerinin gücü veri merkezli uygulamalardan gelmektedir. Akıllı cihazların ağında ulaşım altyapısı anında iletişim ve anında hesaplama gücüyle akan veriler üzerinde çalışır. Sharma ve Awasthi (2022) sistemi üç parçaya böler:

- Veri üretimi
- Verinin işlenmesi
- Verinin saklanmasıdır.

Ulaşım filolarındaki araçlardan elde edilen veri OBU (*on board unit*) olarak adlandırılan gömülü bilgisayarlardan elde edilir. *OBU*, farklı birçok algılayıcıdan veri toplar. Oldukça hızlı bir şekilde akan verinin işlenmesini ve gönderilmesini yönetir. Çeşitli uyarıcılarla hem araç içinde hem de sunucu tarafında uygun uyarılar üreterek araç takibinin verimliliğini sağlamaktadır.



Şekil 2.14 Akıllı ulaşım sistemleri iş birliği etkileşimi (UAB, 2023).

Akıllı ulaşım sistemlerinde iş birliğini belirleyerek ulaşım verimliliğini ve güvenliğini arttırmak amacıyla araçların birbirleriyle ve çevreyle etkileşim kurması Şekil 2.14'te görüldüğü gibi haberleşme türlerine ayrılmıştır. Gelişen haberleşme teknolojileri sayesinde yollarda meydana gelen problemlerin anında tespit edilmesi ve araçların yol kenarında bulunan algılayıcılarla etkileşimi daha konforlu seyahatin gerçekleştirilmesini sağlamaktadır.

Araçların haberleşmeleri dört türlü olabilir ve bağlı araçlar (*connected vehicles*) kavramını ortaya çıkartır:

1) V2V (Araç – Araç): Araçların birbirleriyle haberleşmesi

2) V2I (Araç – Altyapı): Araçların yol kenarı birimlerle haberleşmesi

3) I2I (Altyapı – Altyapı): Yol kenarı birimlerin birbirleriyle haberleşmesi

4) V2X (Araç – Her Şey): Araçların birbirleriyle, yayalar ile ve yol kenarı algılayıcılarla haberleşmesi

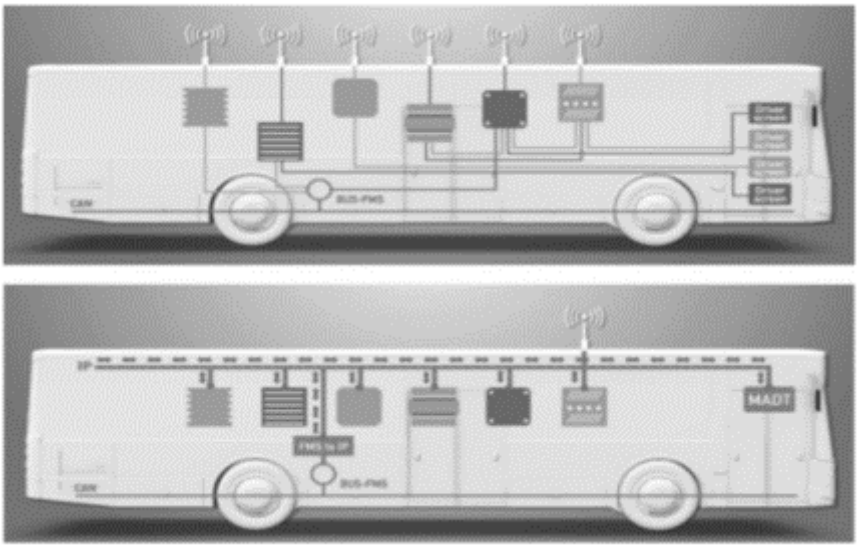
2.5.1 Bağlı araçlar

Bağlı araçlar, hem iç (yerleşik) hem de dış olmak üzere farklı türde algılayıcılar ve çoklu arabirim kartlarıyla donatılmış; hesaplama, depolama ve iletişim olanaklarına sahip, çevresini öğrenerek karar verebilen araçlardır. Bağlı araçların sayısı arttıkça, kablosuz teknolojiler (3GPP, IEEE 802.11p, Bluetooth vb.) ve algılayıcılar (radar, lidar vb.) ile donatılarak, verimli yönetim ve taşımacılık uygulamalarında araç akışlarını iyileştirmeyi, seyahat süresini en aza indirmeyi ve trafik sıkışıklığını önlemeyi amaçlamaktadır.

Araçlara yüklenen akıllı sistemler sayesinde gerçek zamanlı ölçümler ve güvenlik servisleri sisteme kazandırılır (El Zouka, 2016). Araç ağı içerisinde bulunan bu cihazlar birbirleriyle iletişim kurarak kendi kendine organize olan büyük bir algılama ağını oluşturmaktadır.

2.5.2 Toplu Ulaşım için Bilgi Teknolojisi

Farklı üreticilerin sağladığı sistemler ve cihazlar belirli teknolojiler kullanıldığında beraber çalışsa bile kurulum süreçlerinde çeşitli sorunlara sahiptir. Bilinen bu karmaşıklığın açık standartlar ile doğru şekilde yönetimi ve uyum sorunlarının giderilmesi için Toplu Ulaşım için Bilgi Teknolojisi (*Information Technology for Public Transport, ITxPT*) uluslararası standardı tanımlanmıştır. Toplu ulaşım alanında ekle/çıkar yaklaşımına göre beraber çalışabilen bilgi teknolojisi sistemi önerilmiştir. Bu doğrultuda tanımlanan mimari içerisinde farklı fonksiyonların yönetilmesi sağlanmış, kurulum ihtiyaçları, donanım ihtiyaçları ve yazılım ihtiyaçları tanımlanmıştır.



Şekil 2.15 Geleneksel mimari ile önerilen bilgi teknolojisi karşılaştırması (ITxPT, 2017).

Geleneksel mimari ile bilgi teknolojisinin önerdiği mimarinin karşılaştırması Şekil 2.15'te görülmektedir. Üstte bulunan şekilde araç veri hattına bağlı her bir donanımın belirli bir sürücü ekranı ve iletişim anteni bulunur. Yeniden organize edilen alttaki resimde toplu ulaşımda bilgi teknolojisi karmaşıklığı sadeleştirmiş, her donanımın bir IP ağı üzerinden birbirine bağlanmasını ve üretilen verilerin tanımlanan servislerle birbirleri arasında paylaşılmasını önermiştir. Böylece anten ve ekran sadeleşmiş, araç veri hattından alınan bilgiler IP ağına yönlendirilmiştir.

Mimarinin oluşturulma amacı beraber çalışmayı sağlamaktır. Bunun için sistemde bulunan her donanım birbirleriyle etkileşim halinde bulunurken özel olarak bir yapılandırma ve fazladan kaynak kullanımı ihtiyacı olmaması gerekir. Mimarinin önerdiği üç katman beraber çalışabilmeyi tanımlamaktadır. Bunlar; fiziksel, haberleşme ve servis katmanıdır.

- Fiziksel katman: Açık Sistemler Arası Bağlantı (OSI) modelinin fiziksel katmanına benzer şekilde araç üzerinde çalışan donanımların standartlarını belirlemek için kullanılır. Konnektör türleri, güç tüketim durumları, anten türleri gibi fiziksel arayüz tanımlarını içerir.

- Haberleşme katmanı: Fiziksel katmanda bulunan donanımların hem yerel ağ üzerinde hem de sunucu ile olan etkileşimde gerekli olan haberleşme modelini içerir. Ağ temelli iletişimde farklı servislerin bulunabilmesi ve servis tanımlarında gerekli olan alanların belirlenmesi haberleşme katmanında gösterilmiştir.
- Servis katmanı: Araç içerisinde gerekli olan servislerin detaylı olarak tanımlandığı katmandır. Envanter servisi, zaman servisi, konum servisi, araç veri hattını IP ağına aktaran servis, araç sefer bilgilerini sağlayan servis, araca binen/inen kişileri sayan servis, sürücü bilgi ekranı servisi, *MQTT* servisi mimari içerisinde tanımlanan esas servislerdir.

2.5.2.1 Otomatik Servis Keşfi

Mimari içerisinde önerilen haberleşme ve servis katmanı, ağ üzerinden bir dizi servisin sunulmasını sağlamıştır. Ancak servisin sunulması için gerekli olan ağın hazırlanması ve servislerin farklı yapılandırma parametrelerinin yönetilmesi manuel yapılandırma ihtiyacını gerektirir. Servis keşfi protokolleri, özel bir destek olmadan servislerin kullanımını basitleştirmek için kullanılır.

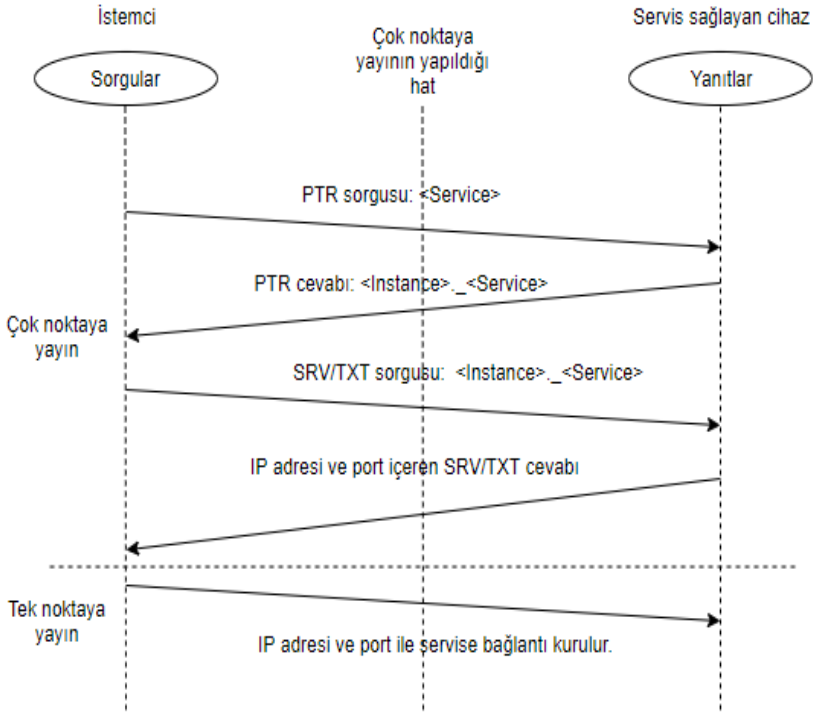
Bu doğrultuda Sıfır Yapılandırma (*Zero Configuration, Zeroconf*) olarak adlandırılan teknoloji, internete bağlı olmayan sistemler için ağ kurulmasını sunucuya gerek duymadan sağlamaktadır (Cheshire and Steinberg, 2005). Çok noktaya yayın DNS (Multicast DNS, mDNS) ve DNS hizmeti keşfi (DNS Service Discovery, DNS-SD) özellikleri, kapalı ağ gereksinimi olan gömülü sistem uygulama ortamları için uygundur.

mDNS, DNS' den farklı olarak çevrim dışı (kapalı) ağlarda, ağ adreslerinin isimlendirilmesi için kullanılmaktadır. DNS hizmet keşfi (DNS-SD) ise servis bilgilerinin paylaşılması görevindedir. Birbirini tamamlayan bu hizmetler ile cihaz ağına katılan yeni servislerin herhangi bir özel yapılandırma yapmadan birbirleriyle haberleşebilmesi sağlanmıştır.

RFC 6763 (2013) içerisinde tanımlanmış olan *mDNS*, IP ağı üzerindeki servislerin bulunması ve yayınlanması için geliştirilen bir standarttır. Standart DNS

programlama arayüzlerini, sunucularını ve paket formatlarını (PTR, SRV ve TXT kayıtları) kullanır. Şekil 2.16'da otomatik servis keşfi akış diyagramı görülmektedir.

- SRV: cihazın sunduğu servis ismi kaydı
- TXT: kullanıcı tanımlı açıklama kaydı
- PTR: servisin atama kaydı



Şekil 2.16 Otomatik servis keşfi akış diyagramı.

DNS hizmeti keşfine ait fonksiyonlar ağ içerisinde en az aşağıdaki özellikleri sağlamalıdır:

- Yayınlama servisi: Servisin isminin, URL'inin, port bilgisinin ve desteklediği protokollerin tanımlanmasına olanak sağlar.
- Güncelleme servisi: Tanımlanan servisin güncellenmesine olanak sağlar. Bunun için öncelikle eski girdilerin temizlenmesi sonrasında yeni özelliklerin yayınlanması gerekir.

- Sorgulama servisi: Var olan servislerin listelenmesine olanak sağlar.

Servislerin paylaşılabilmesi için SRV ve TXT kayıtlarını Çizelge 2.2'de gösterilmiştir. Çizelgede konum servisleri için gerçekleştirimine ait cümleler eklenmiştir.

Çizelge 2.2 Çok noktaya yayın için servis kayıt cümleleri.

<instance>._<service name>._<type>._ <protocol>.<domain>	**SRV Cümlesi**
[hostname]._gnss_location._itxpt_multicast._udp.local 3600 IN SRV 0 0 14005 [hostname]	**Konum Servisi SRV Cümlesi**
TTL class TXT "name=value"	**TXT Cümlesi**
[hostname]._gnss_location._itxpt_multicast._udp.local TTL class TXT "name=value"	**Konum Servisi TXT Cümlesi**

2.5.2.2 Önerilen servisler

Topu ulaşım için bilgi teknolojisi standardı içerisinde önerilen on tane servis vardır. Bazı servisler özel uygulamalar gerektirse de temel olan servisler ile çerçeve içerisinde bulunacak diğer uygulamalara hizmet sağlanmaktadır. Servislerin listesi aşağıda verilmiştir:

- Envanter Servisi (Inventory Service)
- Zaman Servisi (Time Service)
- Konum Servisi (Location Services)
- *VCG* Servisi (Vehicle Communication Gateway)
- *FMStoIP* Servisi (araç içi ve motor sensör verisini IP ağına aktarım servisi)
- *VehicletoIP* Servisi (FMS dışı sensör bilgilerini IP ağına aktarım servisi)
- *AVMS* Servisi (Advanced Vehicle Monitoring System)
- *APC* Servisi (Automatic Passenger Counting Service)
- *MADT* Servisi (Multi Application Driver Terminal Service)

- *MQTT* Servisi (*MQTT* bilgisini sağlayan servis)

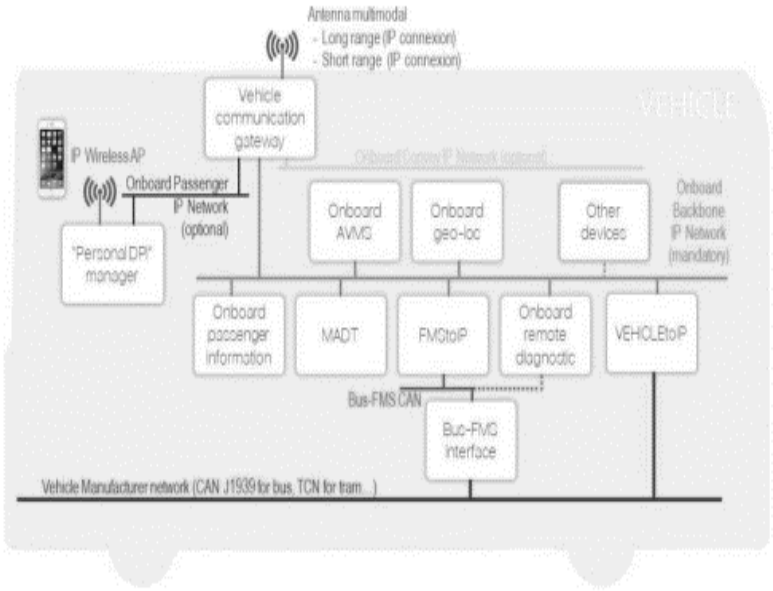
Zaman Servisi: Araç içinde çalışan servisler için saat eşitliğinden sorumludur. SNTP (RFC4330) standardı takip edilmiştir.

Konum Servisi: Araç içi birimlerde bağlam bilgisinin oluşturulması için ihtiyaç duyulan konum verisi, araç navigasyonu, yol güvenliği, sürücü asistan sistemleri ve araç içi eğlence sistemleri gibi çeşitli uygulamalarda kullanılmaktadır. Geleneksel yaklaşımlarda konum verisine bağlı her birim kendi donanım ve antenleri ile çalışmaktadır. Bu durum, kurulum ve bakım maliyetlerinin artmasına ve araç içinde edinilen konum bilgisinin farklı birimlerce farklı olması gibi sorunlara neden olmaktadır. Bu sorunların çözümü için konum servisinin geliştirilmesi önerilmektedir. Konum servisi, konum verisine bağlı olan cihazların tek noktadan standart şekilde veri almasını sağlayarak kurulum ve bakım maliyetlerinin azaltılmasına yardımcı olur.

AVMS Servisi: Araç içi sefer bilgileri, yolcu bilgilendirme sistemleri, yol güvenliği uygulamaları ve araç içi eğlence sistemleri gibi çeşitli uygulamalarda kullanılmaktadır. Geleneksel yöntemlerde, bu bilgiler merkezden edinilerek farklı yolcu bilgilendirme birimleri için ayrı ayrı gönderilmektedir. Bu durum hem araç içinde hem de sunucu tarafında veri trafiği oluşturmaktadır. Bu sorunun çözümü için, araç içi sefer bilgilerinin yardımcı uygulamalar tarafından tek noktadan alınmasını sağlayan bir servis geliştirilmesi önerilmektedir. Bu servis sayesinde, yardımcı uygulamalar ihtiyaç duydukları verileri tek noktadan alırken araç içerisinde dolaşan veri trafiği azalarak verimlilik ve performans artacaktır.

MQTT Servisi: Araç izleme verilerinin farklı servisler tarafından ayrı ayrı yönetilmesini engellemek için *MQTT* sunucusu (*broker*) kurulumu önerilmiştir. İzleme verilerinin hangi başlıklar ile hangi uç noktalara gönderileceği bu servis içerisinde tanımlanmıştır.

Standartta önerilen servislerin araç içi ağındaki iletişimi Şekil 2.17’de görülmektedir.



Şekil 2.17 Ağ üzerinden servislerin görünümü (ITxPT, 2017).

3. AKILLI ULAŞIM SİSTEMLERİNDE CİHAZ YÖNETİMİ

Ulaşım sistemlerindeki araçların internete bağlı olması, akıllı ulaşım sistemlerinin nesnelerin interneti uygulaması olarak değerlendirilmesini sağlar. Nesnelerin interneti ile internet bağlantısı fiziksel cihazlar üzerine doğru genişlemiştir. Algılayıcılar, işleticiler ve radyo frekansı etiketleri olarak değerlendirilen bu fiziksel cihazların tekil veya beraber oluşturduğu sistemler, teknolojinin ilerlemesiyle iletişim ve hesaplama yetenekleriyle donatılmıştır (Atzori et al., 2010). Bu doğrultuda akıllı ulaşım sistemlerindeki bağlı araçların yönetilmesi ve beraber çalışabilmesi için nesnelerin interneti sistemlerine ait benzer gereksinimleri karşılaması gerekmektedir.

3.1 Gereksinimler

Literatürde her alana uygun ve standart olarak kabul edilen bir nesnelerin interneti kurulumu tanımlanmamıştır. Nesnelerin interneti kurulumunun standartlaşma zorluklarının başlıca nedenleri şunlardır:

- Nesnelerin çeşitliliği: Nesnelerin interneti uygulamaları, çeşitli sensörler, veri toplama cihazları ve diğer akıllı cihazları kullanmaktadır. Bu cihazların her biri farklı iletişim teknolojileri, ölçüm birimleri ve veri formatları kullanmaktadır.
- Yönetim konuları: Nesnelerin interneti uygulamalarının farklı uç kullanıcılar tarafından farklı amaçlar için kullanılması sebebiyle altyapı, iletişim ve hizmetler bakımından birlikte yönetilebilmesi ihtiyacı ortaya çıkmaktadır.
- Kendi kendini yapılandırma ihtiyaçları: Nesnelerin interneti cihazları, kendi kendilerini yapılandırabilme özelliğine sahip oldukları zaman kurulum süreci basitleşebilir. Kısıtlı kaynaklara sahip olan cihazları içeren ortamlar için belirtilen yapılandırma desteği sağlanabilmelidir.
- Cihaz yönetimi için gereken fonksiyonlar: Nesnelerin interneti cihazları, farklı yönetim fonksiyonlarına ihtiyaç duyabilir. Bu fonksiyonlar arasında cihazın durumu, güvenliği, performansı ve enerji tüketimi gibi konular yer almaktadır.

Nesnelerin interneti kurulumunun standartlaştırılması için aşağıdaki öneriler yararlı olabilir:

- Nesnelerin sınıflandırılması: Nesnelerin interneti cihazları özelliklerine göre sınıflandırılarak ortak gereksinimlerin belirlenmesi sağlanır.
- Yönetim konuları için standartlar geliştirilmesi: Nesnelerin interneti uygulamaları için standart bilgi teknolojilerin kullanılmasıyla paydaşların ortak bir dil üzerinden etkileşim kurması sağlanır.
- Kendi kendini yapılandırma için standartlar geliştirilmesi: Nesnelerin interneti cihazlarının yapılandırma durumlarına duyarlı geri besleme kurallarının oluşturulması sağlanır.
- Cihaz yönetimi için standartlar geliştirilmesi: Nesnelerin interneti cihazlarının çalışma koşulları, durumları, güvenliği, performansı ve enerji tüketimi konuları bakımından takip edilmesiyle yönetim ihtiyaçlarında standartlaşma sağlanır.

Nesnelerin interneti kurulumunun standartlaşması alanındaki çalışmalar Bölüm 3.2' de incelenmiştir.

3.2 İlgili Çalışmalar

ETSI (European Telecommunications Standard Institute), OMA (OMA Device Alliance) ve ISO (Internation Standart Organization) uluslararası organizasyonların çalışmalarından yola çıkarak Pandey ve arkadaşları (2011) makineler arası ağ yönetimi konusunu tartışmışlardır. Bu çalışmada, otomatik yapılandırma yönetimi, hataya duyarlılık ve uzaktan yazılım güncelleme konularının önemi vurgulanmıştır. Sheng ve arkadaşları (2015) düşük kapasiteli hesaplama, iletişim ve saklama alanına sahip cihazlardan oluşan nesnelerin interneti ortamları için izleme ve yapılandırma konularını birlikte çalışabilirlik ve ölçeklenebilirlik bakımından incelemiştir.

Nesnelerin interneti cihaz yönetimi kapsamında uygulama katmanı için sıklıkla *MQTT* ve *CoAP* protokolleri kullanılır (Van den Abeele et al., 2014). OMA-DM (OMA Device Management) cihaz yönetimi için *CoAP* destekleyen ilk protokoldür (Mavromatis et al., 2019). *MQTT* ve *CoAP* incelemelerinde paket

gecikmesi ve transfer edilen mesaj sayılarına göre karşılaştırmalar belirgindir. Belirli bir eşiğe kadar *MQTT*'nin *CoAP*'a göre paketleri daha az gecikmelerle teslim edebildiği görülmektedir (Thangavel et al., 2014). Aynı ayarlarla transfer edilen mesaj sayısı üzerine (throughput) yapılan başka bir çalışmada *MQTT, CoAP* kullanımına göre yaklaşık 2 kat daha fazla mesaj kaydederken, *CoAP* http'ye göre 30 kat daha fazla mesaj kaydetmiştir (Joshi et al., 2017).

Geniş ağlarda cihazların yönetimini ve denetimini kolaylaştırmak için ağ yöneticisine yardımcı olan bir uygulama katmanı olan SNMP (*Simple Network Management Protocol*) protokolü tanımlanmıştır (Harrington et al., 2002). Basit ve çok kullanılan bir ağ teknoloji olmasından dolayı nesnelerin interneti cihazları ihtiyaçlarına özelleştirilerek cihazların izlenmesi için kullanılmıştır (Yahya et al., 2017). Lee (2023) çalışmasında farklı versiyonlarda çalışan cihazların SNMP üzerinde çalışan ortak bir platform üzerinden yönetilmesini sağlayan bir çözüm sunmaktadır. Hazırladığı servis platformu duruma göre hizmet sunabilen dinamik bir yapıdadır. Nesnelerin ve servislerin yönetimi konusunda Thoma ve arkadaşları (2014) çalışmalarında hem endüstriyel hem de akademik çalışmaları incelemiş ve nesnelerin interneti sistemlerindeki standartlaşma ihtiyacını hem verilerin tanımlanmasında hem verilerin taşınmasında hem de uygulama katmanında gerekli olduğunu belirtmişlerdir.

Radhika ve Malarvizhi (2012) kablosuz sensör ağları için, Raychoudhury ve arkadaşları (2013) yaygın ve mobil bilişim için ara birim yazılım yaklaşımlarını sınıflandırmıştır. Lien ve arkadaşları (2010), "*WordNet*" veri tabanını kullanarak semantik eşleştirme ile çalışan mesaj tabanlı ara birim yazılımı geliştirmişlerdir. Karşılaştırmalarını Java mesaj servisiyle yaptıkları zaman 0.41 kat performans kaybı gözlemlemişler, mevcut semantik ara birim yazılımlarına göre kıyaslama yaptıkları zaman 5 kat daha yüksek performans ölçümü elde etmişlerdir. Razzaque ve arkadaşları (2015) nesnelerin interneti için farklı ara birim yazılım katmanlarını incelemiştir. Bu çalışmalardan yola çıkarak ara yazılımlar uygulamaya özel, veri tabanı odaklı, servis odaklı ve mesaj odaklı olarak sınıflandırılmıştır.

Uygulamaya özel: Bu yaklaşımın temelinde belirli bir uygulamanın ihtiyaçlarına ve kaynak yönetimine odaklanması vardır. Akıllı ev uygulamalarına

odaklanan çalışmada uygulama özelinde geliştirilen fonksiyonlar ile uygulamaların belirli bağlamlar için başarımı hedeflenmiştir (Huebscher, 2004). MiLAN isimli bir başka ara yazılım ise uygulamalar için sensör ve ağ yönetimine yardımcı olan bir kural yönetimi üzerine özelleşmiştir (Heinzelman et al., 2004). Bu mimari yaklaşımı uygulamalara ve uygulama verisine doğrudan bağlı olduğu için nesnelerin interneti gereksinimlerinde belirttiğimiz çeşitlilik şartını karşılamamaktadır.

Servis odaklı: Servis odaklı bilişim, servislerin uygulama ayrıntılarından ve geliştirme modellerinden bağımsız olarak standart modeller ve protokoller üzerinden erişilebilir ve kullanılabilir olmasını hedefler. Bu, dağıtık uygulama bileşenleri arasında ortak çalışabilme ve gevşek bağlantı sağlamaya yardımcı olur. SStreaMWare, heterojen sensörlerden yayınlanan verilere erişmek için bir sorgulama hizmeti sağlayan servis odaklı ara birim yazılımıdır (Gurgen ve arkadaşları, 2008). İçerisinde bulunan sorgulama servisi, ağ geçidi servisi ve Proxy servisi ile sensör çeşitliliğini yönetme, ölçeklenebilirlik ve karmaşık sorgulama kıstaslarına odaklanan hibrit bir çözüm sağlamaktadır. Music arabirim yazılımı, dağıtık ortamlarda dinamik değişikliklerin meydana gelebileceği durumlarda sistemlerin çalışmasını desteklemek için kendi kendine esnek bir bileşen tabanlı mimari sunmaktadır (Rouvoy et al., 2009). Otomotiv iletişim çözümlerinin standartlaşmasında kilit rol alan AUTOSAR birliği otomotiv yazılımları için standartları tanımladıkları SOME/IP (Scalable service-Oriented MiddlewarE over IP, SOME/IP) protokolüyle veri taşımacılığı, serileştirme ve hizmet keşfi gibi sorunları çözen bileşenleri tanımlamışlardır (AUTOSAR, 2016).

Mesaj odaklı: Mesaj odaklı arabirim yazılımları bazı çalışmalarda yayınla/abone, mesaj kuyruğu veya olay tabanlı sistemler olarak kullanılmaktadır. Geleneksel olarak ayrışık sistemler arasında asenkron mesajlaşma, bağlaşımı koparma (decoupling) üzerine odaklanır. Artan büyük veri gereksinimleri sonrasında verimliliği arttırmak için Kafka (Goovaerts, 2016) ve RocketMQ (Ge et. al, 2017) gibi mesaj odaklı yazılımlar geliştirilmiştir. Bu çalışmalarla akıllı şehir, akıllı ulaşım vb. büyük veri kullanılan alanlarda mesaj odaklı ara birim yazılımlarında bulunan MQTT gibi çözümler güçlü veri aktarım hizmetleri sağlamaktadır.

Meng ve arkadaşları (2017), makinalar arasında veri paylaşımı için esnek bir makine – makine (M2M) mesajlaşma tasarım modeli önermişlerdir. Olay ve komut bildirimine ait önerdikleri yaklaşımı mikrodalga sensörleri üzerinden gıda kontrolü yapmak için kullanarak makine bağlantısı ve makine varlığı durum tespitlerini nesnelerin interneti algılama yaklaşımlarıyla destekleyerek güçlü bilgisayarlar ile kısıtlı kapasiteye sahip cihazların beraber çalışabilmesini göstermişlerdir.

Literatürde, akıllı ulaşım sistemine ait yapılan çalışmalar, uygulamaların gruplanmasına göre hazırlanmıştır. Bazı çalışmalar uygulamaları servis tiplerine göre tanımlarken bazı çalışmalar ise araçlar ve altyapı olarak ayrıma gitmiştir. Günümüzde ISO 14813-1 standardı uygulamaların sınıflandırmasını ele alarak akıllı ulaşım sistemleri için standardı belirlemiştir. Bu doğrultuda alana özgü gerçekleştirimler önem kazanmaktadır. Sefer bilgileri, trafik yönetimi, toplu taşıma servisleri, yolcu bilgilendirme servisleri gibi farklı servis grupları sunulacak hizmete ait özellikleri belirlemektedir. Alana özgü çalışmalar "Geleceğin Avrupa Otobüs Sistemleri" projelerini (Tozzi et al., 2016; Corazza et al., 2016) temel alarak geliştirilmiştir. Bu çalışmalar ulaşım sistemlerine ait enerji tüketimi, güvenli sürüş ve verimli yolcu bilgilendirme çalışma alanlarına standartlaşma getirirken bilgi teknolojilerinin farklı üreticiler tarafından kolaylıkla kullanılabilmesi için gerekli olan alt yapı ihtiyacını tanımlamaktadır.

Toplu Ulaşım için Bilgi Teknolojisi (*ITxPT*, 2017) uluslararası standardı, araç ağı içerisinde ekle/çıkar yaklaşımına göre beraber çalışabilen bilgi teknolojisi sistemi önermiştir. Bu doğrultuda tanımlanan mimari içerisinde farklı fonksiyonların yönetilmesi sağlanmış, kurulum, donanım ve yazılım ihtiyaçları tanımlanmıştır. Puerta ve arkadaşları (2018), standardı kullanarak sağlanan hizmetlerden verimli sürüş için yeni bir servis önerisinde bulunmuştur.

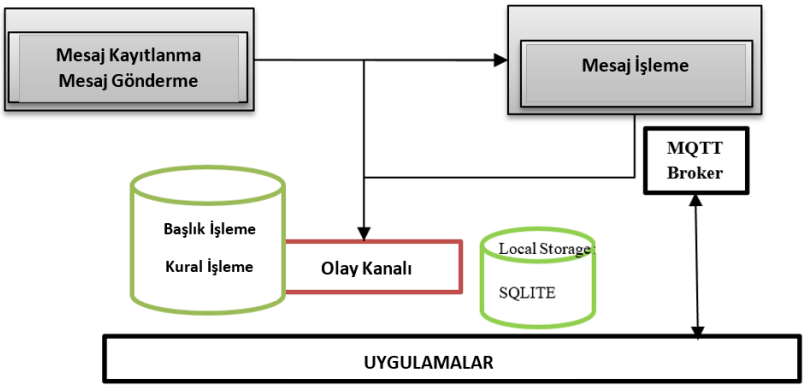
Sharma ve Awasthi (2022) akıllı ulaşım sistemleri mimarisini üç parçaya ayırır. Bunlar; veri üretimi, verinin işlenmesi ve verinin saklanmasıdır. Ulaşım filolarındaki araçlardan elde edilen veri OBU (*on board unit*) olarak adlandırılan gömülü bilgisayarlardan elde edilir ve haberleşme yetenekleriyle üzerinde bulunan birçok algılayıcıdan bilgi toplar.

4. GELİŞTİRİLEN CİHAZ YÖNETİM ÇERÇEVESİ

Bu bölümde geliştirilen cihaz yönetici çerçevesi ve bileşenleri açıklanmıştır. Bölüm 2.5'te incelenen akıllı ulaşım sistemleri, ulaşım sistemlerinin verimliliğini, güvenliğini ve sürdürülebilirliğini iyileştirmek için bilgi ve iletişim teknolojilerinden yararlanan sistemlerdir. Bu sistemlerin etkili bir şekilde çalışması için sistem içerisinde bulunan bileşenlerin kendi aralarında güvenilir ve verimli bir şekilde iletişim kurmaları gerekir. Akıllı ulaşım sistemleri kapsamında haberleşme gereksinimleri ve dağıtık uygulama bileşenleri tanımlanır (Alam et al., 2016; Korablev et al., 2021; Sharma ve Awasthi, 2022). Akıllı ulaşım sistemlerine özelleşen mimari tanımlarıyla donanım sadeliği, beraber çalışabilme, bütünlük ve bakım kolaylığı amaçlanmıştır.

Tez içerisinde, toplu ulaşım için bilgi standardında önerilen servislerin sağladığı veriler arasında koordinasyonu sağlayan bir çerçeve tanımı yapılmıştır. Geliştirilen cihaz yönetim çerçevesi esas mimariyi genişleterek veriden anlam çıkarmayı ve ağ içi uygulamaların görevlerini iletişim alt yapısını önemsemeden yapabilmelerini hedefler. Cihaz yöneticisi, mimaride önerilen standart teknolojileri desteklediği için, kolaylıkla farklı üreticiler tarafından kullanılabilecektir.

Şekil 4.1, toplu taşıma sistemlerinde bulunan uygulamaların veri kaynaklarına ve birbirleriyle ilişkilerine ait üst seviye çerçeve şemasını içerir. Algılayıcı (sensor) ve konum verileri, dağıtıcı uygulamalar ile sistemi beslerken, birbirlerinden bağımsız olarak çalışan servisler veri paylaşımlarını, veri değişim kayıtlarını ve durum güncellemelerini cihaz yöneticisi olarak adlandırılan uygulama birimi ile sağlamaktadır. Cihaz yöneticisi uygulamalar ile sistem içerisinde dolaşan veri arasında bulunmaktadır.



Şekil 4.1 Cihaz yöneticisi üst seviye mimarisi.

Cihaz yöneticisinin son uygulamalara ait verileri ve sensör verilerini birleştirilmesi için dört yardımcı birim tanımlanmıştır:

1) Yapılandırma birimi

2) Bildirim birimi

3) Durum yönetimi birimi

4) Kural yönetimi birimi

Ağ içerisinde dolaşan veri, cihaz yöneticisi tarafından karşılanıp ilgili kurallara göre anlamlaştırılarak ağ içerisinde yeniden dağıtılmaktadır. Cihaz yöneticisi üzerindeki veri akışı ve birimlerin etkileşim türleri:

- Veri toplama: Uygulamalar, veri toplama fonksiyonları aracılığıyla cihaz yöneticisi veri tabanına verilerini kaydeder. Bu veriler, cihazın kendisinden, cihaz üzerinden veya başka donanımlardan gelmektedir.
- Sistem durumu izleme: Bu veriler, öncelikle sistem durumunun belirlenmesi için işlenir. Sistem durumu, ulaşım ihtiyaçlarına göre özelleştirilmiştir. Sürücü girişi, sefer ve durak değişim bilgileri her uygulamanın görevlerini yapabilmesi için daha anlamlı şekilde izlenmektedir.

- Bildirim yönetimi: Uygulamaların cihaz dışından gelen verilere erişimi için sunucu ile olan haberleşme tanımlanmıştır. Senkron ve asenkron iletişim için fonksiyonlar hazırlanmıştır.
- Mesaj modeli: Durum değişimleri sırasında araç içerisinde dağıtılan mesaj paketlerin tanımı yapılmıştır. Her durum değişimine ait mesaj tanımları ve içeriği, cihaz yöneticine yüklenerek değişimlerin bağlama duyarlı olarak paylaşılması sağlanmıştır.

Araç ağı içerisinde bazı uygulamalar temel sistem durumuna göre görevlerini yerine getirirken, araç durum takibi ve alarm durumları için birden fazla sensör ve bağlam verisine duyarlı, olay üretimine ihtiyaç duyan uygulamalar da vardır. Bunlar için kural dosyaları oluşturulmuştur ve kural sözdizimi ulaşım sistemlerindeki ihtiyaçlara göre özelleştirilmiştir.

Cihaz yönetici çerçevesinin bileşenleri Bölüm 4.1’ de açıklanmıştır.

4.1 Ağ Yapılandırması

Akıllı ulaşım sistemlerinde, sistemde yer alan her aracın kendine ait bir birincil IP ağı (*primary network*) bulunmalıdır. Bu ağ, yazılım servislerinin birbirleriyle haberleşebilmesini sağlar ve ağ üzerindeki her servisin kendine ait bir ağ adresi bulunur. Ağ adresleri sınıflandırılarak kullanım alanlarına göre özelleştirilir. 1-255 arası sınıflandırılan ağ adresleri dağılımı aşağıda belirtilmiştir:

- A sınıfı 1-127
- B sınıfı 128-191
- C sınıfı 192-223
- D sınıfı 224-239
- E sınıfı 240-255

Ulaşım sistemlerinde, Bölüm 2.5.2’ de tanımlanan ulaşım bilgi teknolojisi standardına göre cihazlar için gerekli olan IP ağının C sınıfında olması yeterlidir. Çizelge 4.1’de belirtilen IP aralığı, araç ağı için kullanıma hazırlanmıştır.

Çizelge 4.1 OBU ağ kurulum bilgileri.

Aralık	Adres Sayısı	Sınıfı	Alt Ağ
192.168.0.0	256	C	255.255.255.0

Birden fazla donanımın bulunduğu ortamlarda IP atamaları üç yaklaşım ile gerçekleştirilebilir:

- Durağan Atama: Cihaza durağan kodlu olarak IP atanır. Aynı değerli IP'lerin kullanılması ve bakımın kolay yapılamaması durumları yüzünden kullanışlı değildir.
- Dinamik Atama: IP tanımlaması için ortamda bulunan IP dağıtma sunucusundan (*DHCP*) yazılım parçasına özel bir IP verilmesi talep edilir. Sunucunun görevi farklı yazılım parçalarına sistemde kullanılmayan IP dağıtmasıdır.
- Otomatik Atama: IPv4 için 169.254.0.0/16 Aralığı dinamik kullanım için ayrılmıştır.

Çalışmamızda araç ağında bulunan değişken yazılım hizmetlerine ulaşabilmek için dinamik IP atama yaklaşımı belirlenmiştir. Böylece sisteme dâhil olan her donanım, yönetici cihazın dağıttığı IP ile birbirlerini görürken, kurulum sırasında elle yapılandırma ihtiyacı ortadan kalkmıştır.

Belirsiz servis birimlerine kolaylıkla erişim sağlamak için yerel ağın içinde çok noktaya yayın DNS ve DNS keşfi yaklaşımları uygulanmıştır. Bölüm 4.3' te bu yetenekleri sağlamak için ihtiyaç duyulan protokoller ve uygulamalar tanımlanmıştır.

4.2 Bağlam Tanımı

Cihaz yönetici çerçevesinin farklı servis sağlayıcılara cihaz durumuna ait anlamlandırılmış veriyi sağlaması, uygulamaların beraber çalışabilmelerini kolaylaştırmaktadır. Veri ve olayların sistem içerisinde anlamlandırılması, sisteme özel bağlam tanımını gerekli kılar. Çizelge 4.2' de çerçeve için tanımlanan bağlam modeli yer almaktadır. Bağlamı oluşturan veri içerisinde konum, zaman damgası,

durum bilgisi bulunur. Ulaşım sistemleri için oluşan bütün olaylar içerisinde tanımlanan bağlam verisi birleştirilmiştir. Örneğin; sefer açılması sırasında üretilen olay paketi için sefer bilgilerinin yanında seferin başladığı konum bilgisi, zamanı ve sistem içerisindeki durumlar sefer açılması olayına eklenmiştir.

Bir uygulamanın ürettiği olaylar sistemin durumunu değiştirirken, diğer uygulamalar bu durum değişimlerinde kullanılabilecek yeni verileri oluştururlar. Cihaz ağı içerisinde paylaşılan olay paketlerinin uygulama çalışma prensibini değiştirmemesi için olay paketleri veri üzerinden kendini tanımlayacak şekilde hazırlanmıştır. Böylelikle, değişen her durum için uygulamalar yeni mesajları araç ağı içerisinde paylaşabilirler.

Değişken boyutlu olay paketlerinin paylaşılması, *TLV (Tag Length Value)* serileştirme metoduyla sağlanmıştır. Böylece sistem içerisindeki bütün verilerin boyutları paket içerisinden takip edilirken, tanımlanan veri başlıklarıyla bir sözlük oluşturulması sağlanmış, bölüm 4.3.2'de açıklaması yapılmıştır. Olayların sistem içerisinde taşıyacağı veri; konum ve zaman standart paketlerini, ayrıca duruma özel detay verileri içermektedir.

Çizelge 4.2 Bağlam modeli elemanları.

Konum	Başlık	Olay/Durum Tipi	Olay/Durum Özel Verisi

Pozisyon: Konum birimi üzerinden elde edilen enlem, boylam, yükseklik, hız, uydu sayısı, evrensel zamanı, kilometre sayacı verileridir.

Başlık: Başlık içerisinde uygulamaların çalışılan sisteme göre tanımlanmasını sağlayacak bilgiler bulunmaktadır. Uygulama tipi, sistem tipi, araç tanımı, durumu, zaman ve oturum zamanı bilgileri ile uygulama ve sistem bağımsız bütün verilerin gruplanmasını sağlamaktadır. Çizelge 4.3'te uygulama tipleri listelenmiştir.

Çizelge 4.3 Toplu ulaşım sistemleri için uygulama tipleri.

UYGULAMA TİPİ	
Uygulama Tipi No	**Uygulama Adı**
0	GPS Uygulaması
1	Alarm Yönetimi Uygulaması
2	Otobüs Yönetimi Uygulaması
3	Rota Uygulaması
4	Araç Bakım Uygulaması
5	Otomatik Pozisyon Yönetimi Uygulaması
6	Otomatik İnsan Sayacı Uygulaması

Olay/Durum Tipleri: Uygulama tiplerine göre oluşan durumları göstermektedir. Bazı durumlar her uygulama için ortak olsa da bazı durumlar uygulamalara göre özelleşebilir. Çizelge 4.4'te toplu ulaşım sistemleri için durum tipleri yer almaktadır.

Çizelge 4.4 Toplu ulaşım sistemleri için durum tipleri.

DURUM	
Durum No	**Durum Tipi**
0	Uygulama Başlaması
1	Gün Başı
2	Sefer Başlaması
3	Sefer Kapanması
4	Sürücü Kartı Takılması
5	Sürücü Kartı Çıkartılması
6	Durak Girişi
7	Durak Çıkışı

Mesaj Paketi Veri Modeli: Cihaz ağı içerisinde kaydedilen ve dağıtılan mesaj paketleri, bağlam tanımı için gerekli olan cihaz ve uygulama belirteçleri ile olay ve durum tiplerinin birleştirilmesiyle oluşturulur. Bu mesajlar için oluşturulan veri modeli elemanları aşağıda listelenmiştir:

- Cihaz Belirteci
    - Cihaz Küme Numarası (Güvenli çip içerisinde alınır)
    - Cihaz Numarası (Güvenli çip içerisinden alınır)
    - MAC Adresi
- Uygulama Belirteci

    - Uygulama Numarası (Cihaz Yöneticisi: ApplicationID)
    - Parametre Tanımı (Cihaz Yöneticisi: ParamID)
- Durum Belirteci
    - Araç Durumu (Cihaz Yöneticisi: StateType)
- Olay Belirteci
    - Ana Olay (Cihaz Yöneticisi: MainEvent)
    - Alt Olay (Cihaz Yöneticisi: SubEvent)

4.3 Çerçeve Bileşenleri

Yönetici çerçeve altı modüle ayrılmıştır:

1) Bildirim yönetimi modülü
2) Olay yönetimi modülü
3) Yapılandırma yönetimi modülü
4) Durum yönetimi modülü
5) Kural yönetimi modülü
6) Servis keşfi yönetimi modülü

4.3.1 Bildirim yönetimi modülü

Cihaz yöneticisi içerisinde uygulamalar ile sunucu arasında çift taraflı haberleşme birimi olarak çalışır. Oluşan olaylar sistem içerisinde bulunan uygulamalara bildirilir. Uygulama türünden bağımsız olarak aşağıdaki olaylar desteklenmektedir:

- Cihaz/servis bulunması
- Yapılandırma değişimleri
- Cihazların/servislerin uygulama çalışma durumları

Ağ içerisinde iletişim için mesajlar bir başlık altında gönderilecektir. Her başlığa ait ilgili mesajın alanları paylaşılan verileri ifade eder.

- init: source_name: mesaj gönderiminde gönderilen mesaja eklenecek kimlik kaynağı bilgisi
- Register: topic, callback_for_topic (source_name, topic, data, data_len)
- Send: topic, data, data_len

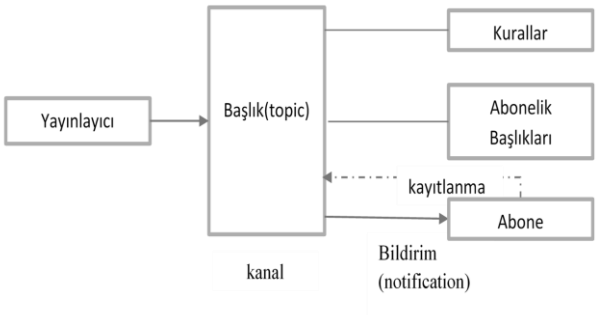
Geliştiricilerin kullanacağı şekilde çok noktaya haberleşme için yukarıda belirtilen metotlar C dilinde tanımlanmıştır. Hem cihaz içerisinde hem de aynı ağ içerisinde mesajlaşma Şekil 4.2'de gösterilen metotlar ile sağlanabilir.

```
extern int MSG_register(void *message_ctx, char *channel_name, int (*callback)(char*, char*, void*, int));
extern int MSG_dispatch(void *message_ctx);
extern int MSG_send(void *message_ctx, char* channel_name, void *data, int data_len);
extern int MSG_init(void **message_ctx, char *source_name, int is_blocking, int *messaging_socket);
extern int MSG_reinit(void *message_ctx, int *messaging_socket);
```

Şekil 4.2 Çok noktaya yayın metotları.

4.3.2 Olay yönetimi modülü

Olayın oluşturulması, işlenmesi ve bildirim yöneticisine yönlendirilmesi görevlerine sahip olan birimdir. Sistem içerisinde olay tabanlı ortak bir olay modeli sağlanır. Şekil 4.3'te olay yönetimi için hazırladığımız olay modeli gösterilmiştir.



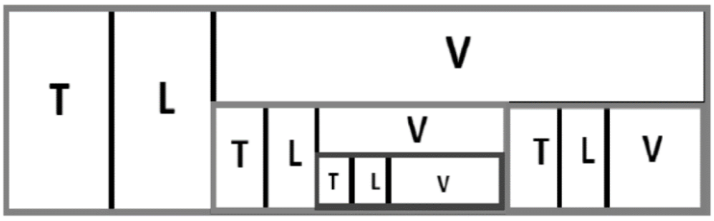
Şekil 4.3 Başlık tabanlı yayıncı/abone modeli.

Hem cihaz içerisinde hem de sunucu ile olan iletişimde eldeki verinin yapısal olarak çözümlenmesi için iki olay başlığı tanımı düşünülmüştür. Bunlar; ana olay (*mainEvent)* ve alt olay (*subEvent)* olarak adlandırılmış ve sayısal veri olarak tutulmaktadır.

Ana olaylar uygulama tiplerini gruplarken alt olaylar durum değişikliklerini içermektedir. Her ana olaya ait durumlar kendi kümesinde tanımlanmıştır. Anlamsal olarak benzer durum değişiklerinin ana olaylar altında eşleştirilmesiyle farklı alt olayların beraber değerlendirilmesi sağlanır.

Oluşturulan olay paketleri içeriğinin kendini tanımlayabilmesi ve kısıtlı disk boyutuna sahip cihaz üzerinde saklanabilmesi için ASN.1 (*Abstract Syntax Notation One*) kodlama standardı kullanılmıştır (ITU-T. 690, 1997). Spesifikasyonda verinin kodlama kuralları sınıflandırılmış, en sık tercih edilen BER (*Basic Encoding Rules*) formatı uygulanmıştır. BER formatının yapı taşları aşağıda listelenmiş, Şekil 4.5'te örneklenmiştir:

- TLV (Tag Length Value)
  - Bütün veriler TLV formatında kodlanır.
  - TAG: veri türünü belirtir.
  - LENGTH: verinin uzunluğunu belirtir.
  - VALUE: verinin kendisidir.



Şekil 4.4 ASN.1 kodlaması BER formatı (ITU-T 690, 1997).

Örneğin güzergâh kodu tanımı DF34 etiketiyle tanımlanmış, 3 byte ile ifade edilmiştir. Uygulama alanına ait veri sözlüğü Şekil 4.5'te görülmektedir.

HEADER		LENGTH
Main event	df01	1 byte
Sub event	df02	1 byte
Systemid	df03	1 byte
Busid	df04	3 byte
Samid	df05	3 byte
Status*1	df06	2 byte
Date	df07	4 byte
Sessionid	Df08	4 byte

GPS		LENGTH
Lat	Df20	4 byte
Long	Df21	4 byte
altitude	Df22	3 byte
speed	Df23	1 byte
svcount	Df24	1 byte
utctime	Df25	4 byte
accuracy	Df26	1 byte
source	Df27	1 byte
course	Df28	2 byte
Odometer	Df29	4 byte

ADDITIONAL		LENGTH
stop seq no	Df30	1 byte
Stop id	Df31	3 byte
Route code	Df32	3 byte
Direction	Df33	1 byte
Pathcode	Df34	3 byte
Driverid	Df35	3 byte
Driveralias	Df36	4 byte
Card insert time	Df37	4 byte
Trip code	Df38	3 byte
VDS Odometer	Df39	4 byte
Fuel Level	Df3A	4 byte
Stop entered time	Df3B	4 byte
Total Fuel Used	DF3C	4 byte
VDS Engine Speed	DF3D	4 byte
Door ID	DF4A	1 byte
Rule ID	DF4B	4 byte
Message id	DF4C	4 byte
APC Params*2 (max 5 door)	DF4D	30 byte
Tap Params*3 (max 3 type)	DF4E	12 byte
File id	DF40	4 byte
Terminal id	DF41	10 byte

Şekil 4.5 Araç içi paylaşılan veri sözlüğü.

Sistem içerisinde paylaşılan mesaj başlıkları ilgili olaylar ile eşleştirilerek araç içinde ve kayıtlarda olaya duyarlı mesajların oluşturulması sağlanmıştır. Örneğin, sefer olayları için başlangıç/bitiş alt olayları bulunur. Bu olay "*tripinfo*" başlığı altında farklı alanlarla veri paylaşımı yapabilmektedir. Başlık ile olayların eşleştirilmesi Çizelge 4.5'te gösterilmiştir.

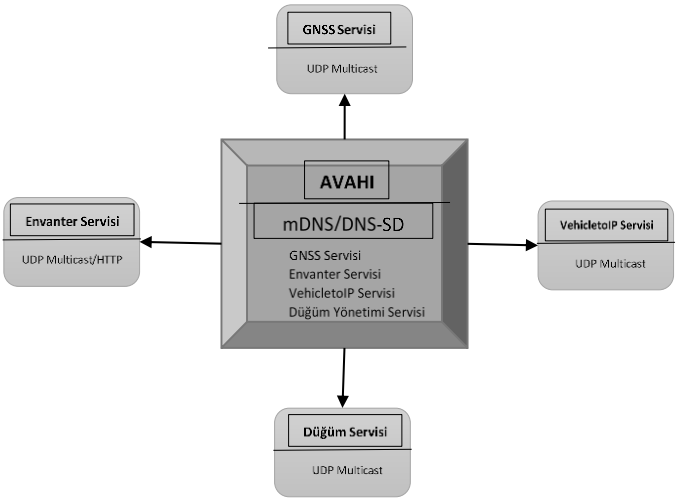
Çizelge 4.5 Olay ve başlık eşleştirmesi.

MAIN EVENT	SUB EVENT	TOPIC	DATA
Bus Management	Trip Opened	CHANNEL_TRIPINFO	TRP_DATA
Bus Management	Trip Ended	CHANNEL_TRIPINFO	TRP_DATA
Bus Management	Stop Entered	CHANNEL_STOPINFO	STP_DATA
Bus Management	Stop Left	CHANNEL_STOPINFO	STP_DATA

TRP_DATA: trip_status, trip_type, trip_type_code, display_route_code, route_start_name, route_end_name, direction, event_date_time, route_code, duty_no
STP_DATA:stop_name, is_at_stop, stop_seq, stop_code

4.3.3 Servis keşfi yönetimi modülü

Sisteme ait olan yeni cihazın bulunup, servislerinin kullanılmasını sağlayan birimdir. Çalışmamızda, çok noktaya yayın DNS (*multicast DNS, mDNS*) ve DNS hizmet keşfi (*DNS Service Discovery, DNS-SD*) hizmetleri Linux sistemlerde *Avahi* uygulaması ile sağlanır. avahi-daemon, avahi-autoipd, avahi-publish ve avahi-browse uygulamalarıyla otomatik dinamik ip belirlenmesi, servis paylaşımı ve servis keşfi özellikleri, araç içi ağ ortamında çalıştırılmaktadır. Şekil 4.6 uygulama kullanımını göstermektedir.



Şekil 4.6 Çalıştırılan sistem servisleri.

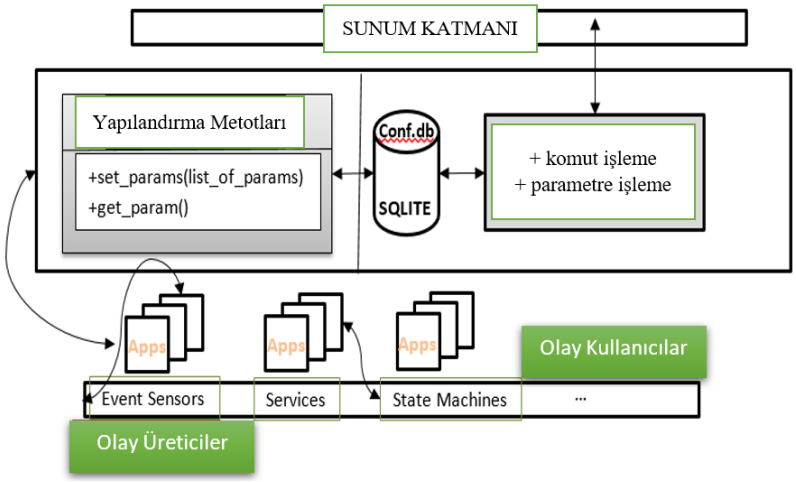
4.3.4 Yapılandırma yönetimi modülü

Yapılandırma yönetimi modülü, sistem içerisinde bulunan uygulamaların parametrelerini ve yapılandırma verilerini okuyan/yazan modüldür. Kapanıp açılmaları da göz önünde bulunduracak depolama ve yeniden yükleme yeteneklerine sahiptir.

Her uygulamanın kendine özel bir tanımı vardır (AppID). Bu id üzerinden parametre güncellenmesi veya parametrenin talep edilmesi yapılır. Uygulamalar paylaşmak istedikleri verileri "paramID = {string}" ve "paramValue = {string}" olarak yayınlar. Parametrelerin önceden tanımlanması gerekmemekte ve ilgili fonksiyon kullanıldığı zaman, cihaz yöneticisi parametre yoksa yaratacak, varsa değişim durumunu kontrol edecek, değişim durumunda kendi veri tabanını olay zamanı ile güncelleyecek şekilde hazırlanmıştır.

- set_params: [AppID, {paramId, paramValue}] (void)
- get_params:[AppID, {paramid}] => paramValue

Parametreler kendi içerisinde yapılandırma ve komut türlerine ayrılmıştır. Tanımlama aşamasında sistem içerisinde hangi türde kullanılması belirtilerek sistem içerisindeki kullanımı özelleştirilir. Şekil 4.7'de yapılandırma yönetimi etkileşimi gösterilmiştir.



Şekil 4.7 Yapılandırma yönetimi.

Parametreler değişime duyarlı olarak kullanılacağından, son durumları cihaz içerisinde Şekil 4.8'de görüldüğü gibi saklanmaktadır.

Record No	App ID	Param ID	Param Value	Last Update	Origin	Status

Örnek Konfigürasyon Verileri:

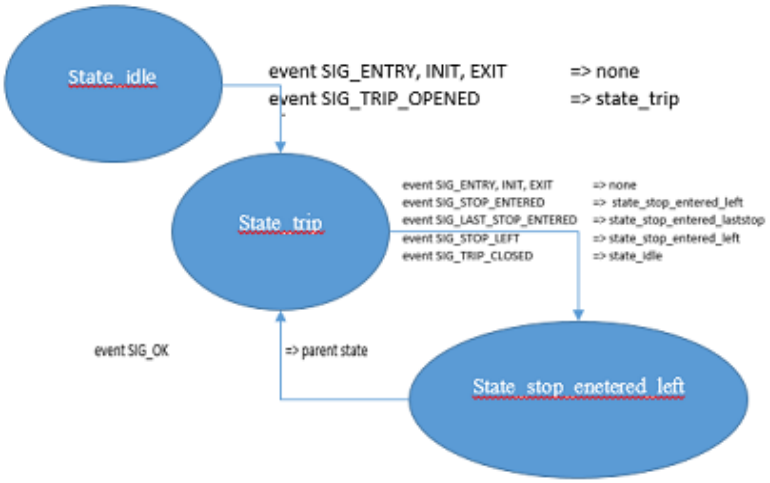
```
Record Number   APP_ID  CONF_ID VALUE   LAST_UPDATE ORIGIN  STATUS
-----   ------------------------------------------------------------
[0] ->  <1> <sleep_mode>    <0> <20211221141022>    <61CAE0FF 649EDECA> <0>
[1] ->  <1> <read_card> <1> <20191024162531>    <7FBE46D2 166ACB3B> <1>
[2] ->  <1> <driver_no_password>    <0> <20191024162531>    <7FBE46D2 6662987C> <1>
[3] ->  <1> <cico_mode> <4> <20210521105525>    <61CAE100 25414905> <0>
[4] ->  <1> <station_type>  <5> <20191024162531>    <7FBE46D2 2B46D8C>  <1>
[5] ->  <1> <confirm_state> <4> <20211228140030>    <61CAEE4E 74C02072> <0>
[6] ->  <1> <route_code>    <10001> <20211228140030>    <61CAEE4E 74C02072> <0>
[7] ->  <1> <driver_code>   <000001>    <20211228103030>    <61CABD16 4C6119FC> <0>
[8] ->  <1> <direction> <0> <20211228133030>    <61CAE746 505EE705> <0>
[9] ->  <1> <bus_id>    <55554> <20211228101030>    <61CAB866 35790A75> <0>
[10]    ->  <1> <sw_version>    <  TBL V2.6.2.3  >  <20211227173004>    <61CAE100 28D86B26> <0>
[11]    ->  <1> <test_status>   <0> <20211228104030>    <61CABF6E 3C8DD369> <0>
[12]    ->  <1> <ft_version>    <20211129171700>    <20211129180010>    <61CAE13B 309CD5A5> <0>
[13]    ->  <1> <para_version>  <20211201120800>    <20211201120849>    <61CAE13C 3FA9188D> <0>
[14]    ->  <1> <sam_version>   <ValSAM v2.9     >  <20371130094455>    <7FBE46D3 20F41263> <1>
[15]    ->  <1> <trip_no>   <0> <20191024162531>    <7FBE46D3 4212552>  <1>
[16]    ->  <1> <tstart_time>   <20211229061231>    <20211228140030>    <61CAEE4E 74C02072> <0>
[17]    ->  <1> <medialib_ver>  <V1.5.8>    <20211227173004>    <61CAE100 6E8A3426> <0>
[18]    ->  <1> <drvc_stat> <0> <20191024162531>    <7FBE46D3 3BE3267>  <1>
[19]    ->  <1> <data_stat> <0> <20211228113330>    <61CACBDA 47C6E722> <0>
[20]    ->  <1> <sys_id>    <105>   <20371130094455>    <7FBE46D3 628843BE> <1>
[21]    ->  <0> <prm_period>    <300>   <20191024162538>    <7FBE46D3 6A4E4E26> <1>
[22]    ->  <0> <tm_version>    <v6.3.1>    <20210414161330>    <61CAE100 2E21F86D> <0>
[23]    ->  <0> <tmslib_version>    <v1.2.7>    <20210326154538>    <61CAE100 AED81E0>  <0>
[24]    ->  <0> <temperature>   <0.00>  <20211222130904>    <61CAE100 21081F4E> <0>
[25]    ->  <0> <free_space>    <33780> <20211228132826>    <61CAE6CA 7E7F47FB> <0>
[26]    ->  <0> <free_memory>   <888220>    <20211229031310>    <61CBA816 799B16B9> <0>
[27]    ->  <0> <uptime>    <21>    <20211229030249>    <61CBA5A9 23769CE7> <0>
[28]    ->  <0> <platform>  <IMX>   <20191024162538>    <7FBE46D3 754E3D4D> <1>
[29]    ->  <0> <position_index>    <1> <20191024162538>    <7FBE46D3 69CF217B> <1>
[30]    ->  <0> <radius>    <1> <20191024162538>    <7FBE46D4 1310FD74> <1>
[31]    ->  <0> <altitude>  <1> <20191024162538>    <7FBE46D4 45F32E66> <1>
[32]    ->  <0> <board_version> <2.5>   <20191024162538>    <7FBE46D4 55FEE8DF> <1>
[33]    ->  <0> <resolution>    <480X272>   <20191024162538>    <7FBE46D4 4F90F095> <1>
[34]    ->  <0> <device_type>   <validator> <20191024162538>    <7FBE46D4 31B8B08D> <1>
[35]    ->  <0> <kernel_version>    <2.6.35.3-lionRaD-0045-gda66ae0>    <20371130074051>    <7FBE46D4 10097D64> <1>
```

Şekil 4.8 Yapılandırma yönetimi veri tabanı görünümü.

Veri tabanında biriken verilerin sunucuya gönderimi Bölüm 4.4'te telemetri gönderimi konusunda açıklanmıştır.

4.3.5 Durum yönetimi modülü

Hazırladığımız durum yönetimi modülüyle durum ve sinyaller eşleştirilerek durum değişimlerinin takip edilmesi sağlanmış, olaya duyarlı olarak mesajların araç ağı içerisinde dağıtılması sağlanmıştır. Cihaz yöneticisinin kullandığı ve cihaz içerisinde çalışan diğer uygulamaların kullanabileceği durum yönetimi modülünün veri akışı Şekil 4.9'da görülmektedir.



Şekil 4.9 Cihaz yöneticisi veri akışı değişim diyagramı.

Durum değişimleri ulaşım sistemlerine göre özelleştirildiğinde temel durumların (*OS, Operational States*) boşta, sürücü girişi, sefer girişi durumları olduğu belirlenmiştir. Bunların cihaz yöneticisine yüklenmesi için Çizelge 4.6'da gösterilen söz dizimi hazırlanmıştır.

Çizelge 4.6 Koşul durum olay söz dizimi.

OPERATIONAL STATES	Idle = OS.Idle Login = OS.Login Trip = OS.Trip
OS.<type> = <pre condition> | AESTList (attribute event state list)	
AESTList = **on <interested_attribute_name>**.<event_operator> STList [AESTList]	
event_operator = **IN** | **OUT** ('string')	
STList = NewState : Topic	
NewState = **operational_state** ('string')	
Topic = **topic_name** ('string')	
TRIPINFO= <trp_data>	
STOPINFO= <stp_data>	

Söz dizimi kullanılarak çalıştırılan kurallar Çizelge 4.7' de gösterilmiştir.

Çizelge 4.7 Koşul durum olay söz diziminin çalıştırılması.

OS.idle = **path_code** null | **on** DriverCard.Insert (data) : OS.login : SendMsg(DRIVERINFO
OS.login = **driver_card_data** not null | | **on** DriverCard.Removed(null): OS.idle : SendMsg(DRIVERINFO), **on** Trip.Opened(data) : OS.Trip : SendMSG(TRIPINFO)
OS.Trip = **path_code** not null | **on** Trip.Closed(null): OS.login : SendMSG(TRIPINFO), | **on** DriverCard.Removed(null): SendMsg(DRIVERINFO) : OS.idle, **on** Stop.Entered(data) : null : SendMSG(STOPINFO), **on** Stop.Left(data) : null : SendMSG(STOPINFO)

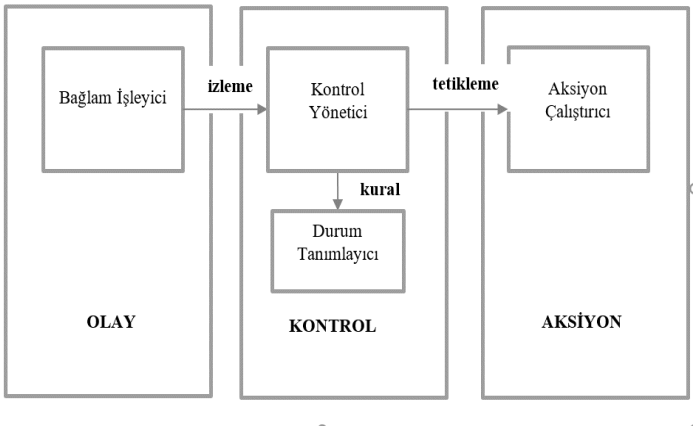
Girilen başlıklar için hangi mesajların gönderileceği başlık veri eşleştirilmesiyle sağlanmaktadır.

Durum yönetimi için program çalışma sırası şu şekildedir:

i. Eğer ilk OS belli değilse OS.idle koşul kontrol edilerek açılır.
ii. Eğer ilk OS belli ise ilgili OS koşul control edilerek açılır. İlgili olay noktaları yerel bağlama yüklenir, ilgili veri girdisi beklenir.
iii. İlgili veri yakalanınca olay türüne göre başlangıç/kapatma olayı sistem içerisinde belirtilen başlık mesajı detayıyla yayınlanır.

4.3.6 Kural yönetimi modülü

Cihaz ağı içerisinde durumlar, algılayıcılardan gelen veriden, geçmiş verinin değişimlerine göre oluşan alarm durumlarından ve pozisyon değişimleri üzerinden üretilir. Örneğin belirli bir pozisyon içinde, araç içindeki insan sayısının tanımlanan bir sayıyı aşması kontrolü birden fazla yazılım biriminin beraber çalışmasını ve özel uygulamaların gerçekleştirilmesini gerektirir. Hızlanma, yavaşlama, agresif dönüş, tork artışı, motorun açık olması, belirli bir alana girilmesi gibi değişken birçok kural noktası tanımlanabilir. Çerçevemizin temel özelliklerinden biri, yayıncı/abone çekirdeğinin, bildirim iletimi, abonelik dağıtımı ve iletişim protokolleriyle ilgili soyutlamalardan, anlamdan ve mekanizmalardan ayrılmasıdır. Şekil 4.10 bağlama duyarlı ortam için kural örüntüsünü gösterir.



Şekil 4.10 Bağlama duyarlı kural örüntüsü.

Cihaz yöneticisine gelen her mesaj incelenerek kural yöneticisine iletilir. Kural yöneticisine gelen her olay depoda tanımlanan kurallara göre olay tetiklenmesi için kontrol edilir. Oluşturulan olaylar ana ve alt olay ikilisiyle bağlam bilgisi yüklenerek dağıtılır. Çizelge 4.8 kural yapısını göstermektedir.

Çizelge 4.8 Kural yapısı.

RULE
Id
Name
Main event
Sub event1
Event1 publish
Sub event2
Event2 publish
Status bit
frequency
Alert level
Data
Data2

Her kuralın ilk durumu ve çalıştırıldıktan sonra sonlanma durumları olay üretimini gerektirir. Bunu gerçekleştirmek için kullanılan söz dizimin hız ve ivmelenme örnekleri Çizelge 4.9'da gösterilmiştir.

Çizelge 4.9 Hız ve ivmelenme verisi için oluşturulan söz dizimi.

SPV = speed
0:1 speed > threshold -> set state 1, generate sub event1
1:0 speed < threshold -> set state 0, generate sub event2
0:1 process_history -> set_state 1
1:2 speed.diff > threshold -> set state 2, generate sub event1
2:0 speed.diff < threshold -> set state 0, generate sub event2
Process_history: utctime != 0, utctime(diff) = 1 -> save_local_history

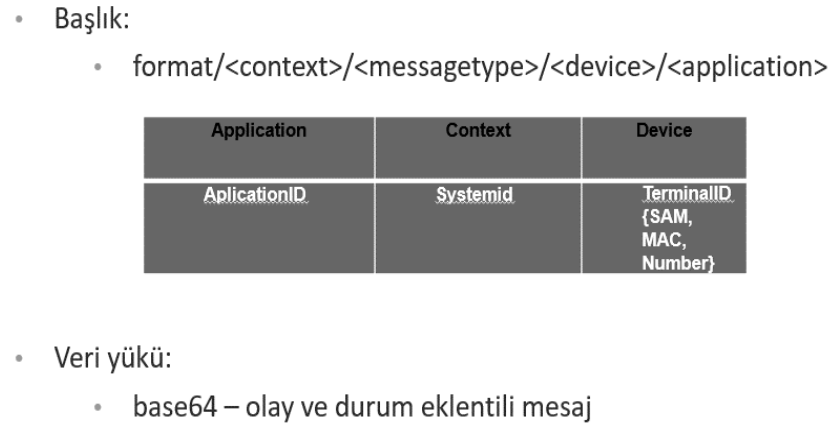
4.4 Mesaj Tabanlı İletişim Modeli

Geliştirdiğimiz çerçevede, cihazların sunucuyla iletişimi *MQTT* ve *WebSocket* protokolü ile sağlanmaktadır. Asenkron mesajlaşma katmanı *MQTT* başlık yapısını ve mesaj değişimlerini tanımlar. Senkron mesajlaşma *WebSocket* üzerinden çalışmaktadır.

4.4.1 Telemetri gönderimi

Cihaz yöneticisi telemetri verisi gönderimini, Bölüm 2.3'te tanıtılan *MQTT Sparkplug* spesifikasyonunu, ulaşım sistemleri gereksinimlerine özelleştirerek yönetir.

Bu doğrultuda *MQTT* başlık yapısı sadeleştirilmiştir: {format belirteci} / {bağlam belirteci} / {mesaj/komut türü} / {cihaz belirteci} / {uygulama belirteci}. Bölüm 4.3.4'te tanıtılan yapılandırma yönetimi bileşenleriyle Şekil 4.11'de görülen bağlama duyarlı mesaj başlık yapısı oluşturulur.



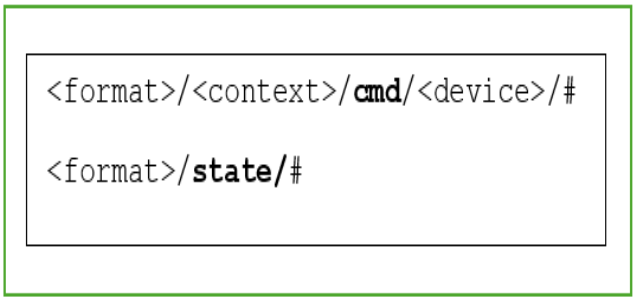
Şekil 4.11 Bağlama duyarlı mesaj başlık yapısı.

Bağlam belirteci ve cihaz belirteci için bölüm 4.2'de tanıtılan veri modeli elemanlarından faydalanılmıştır. Cihazların kullanım bölgelerini belirten sistem numarası ve araç numarası, kurulum aşamasında kullanılan güvenli çip üzerinden alınarak telemetri gönderimi içerisinde bağlam ve cihaz tanımı için kullanılır.

Endüstriyel cihazların aksine ulaşım sistemi cihazlarının tekil değerlendirilmesi yeterli olduğu için uç düğüm kümesi başlıklarına ihtiyaç ortadan kalkmıştır. Bu doğrultuda cihaz yöneticisinin duyarlı olduğu başlıklar Bölüm 2.3'te tanıtılan *MQTT Sparkplug* spesifikasyonuna göre listelendiği gibi sadeleşmiştir:

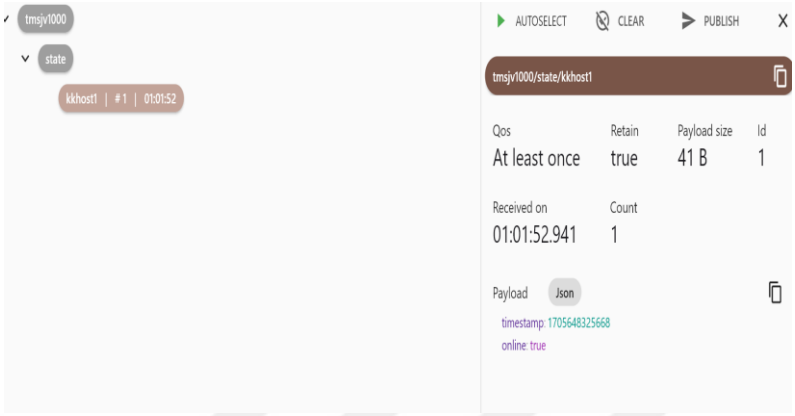
- BIRTH – Cihaz doğum mesajı
- DEATH – Cihaz ölüm mesajı
- STATE – Sunucu çevrim içi / çevrim dışı mesajı
- CMD – Cihaz komut mesajı
- DATA – Cihaz veri mesajı

Cihazlar içerisinde birden fazla uygulamanın takip edilebilmesi, başlık yapısı içine Bölüm 4.3.4'te tanıtılan uygulama belirteçleriyle sağlanır. Cihaz yöneticisi çalışmaya başladığı zaman *STATE* başlığı altında sunucunun çevrim içi bilgisini öğrenir ve kendisine iletilen veriyi sunucuya gönderime hazırlar. Bağlantı kurulduktan sonra dinlenen diğer başlıklar Şekil 4.12'de görülmektedir.



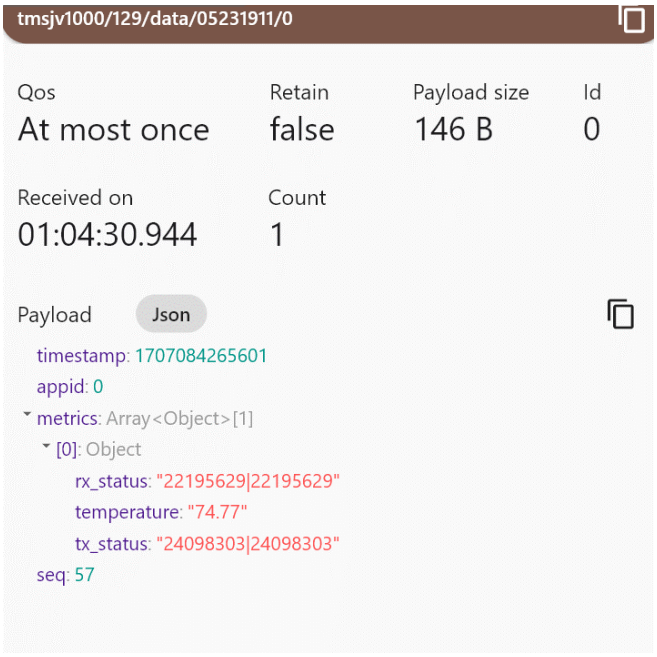
Şekil 4.12 Cihaz yöneticisinin dinlediği başlıklar.

Bölüm 2.2.3'te MQTT spesifikasyonunda belirtilen "*retain*" bayrağı, mesajın bağlantı sonrasında gelmesini garanti etmektedir. Karşılanan başlık yapısı ve veri yükü Şekil 4.13'te görülmektedir.



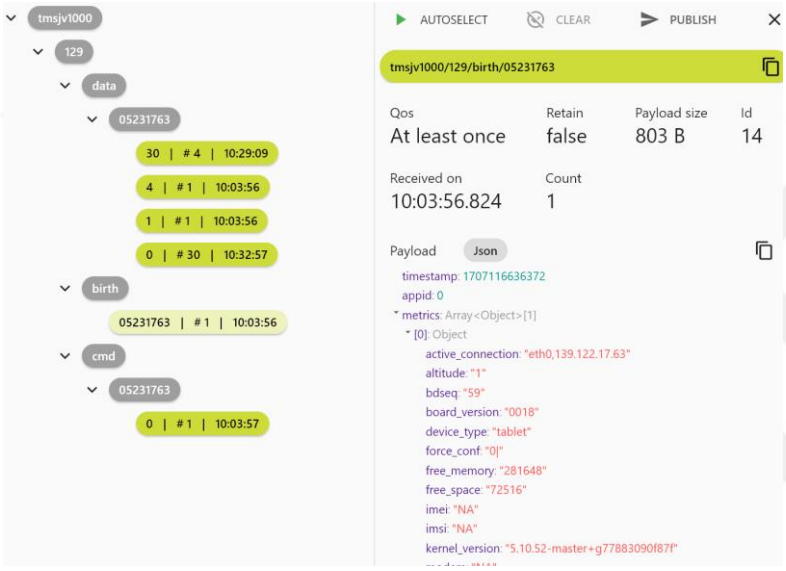
Şekil 4.13 Cihaz yöneticisi başlangıç paketi içeriği.

Mesajların içeriği; veri tanımlayıcısı, zaman damgası ve son gönderilen mesajın sıra numarası bilgisiyle gönderilir. Cihaz yöneticisinin kendi uygulaması üzerinden gönderdiği paket içeriği Şekil 4.14'te görülmektedir.



Şekil 4.14 Cihaz yöneticisi izleme verileri.

İlk gönderilen mesaj (*birth*) cihazın canlı olduğunu gösterir ve sunucu uygulaması tarafından cihazın canlı olduğunu göstermek için kullanılır. *MQTT* spesifikasyonunda bulunan *lwt* (*last will topic*) ile işaretlenen ölüm (*death*) mesajı cihazın çevrim dışı olmasını, MQTT aracısının yönetmesini sağlayarak ilgili cihazı takip eden bütün uygulamaların bu mesajı almasını garanti eder. Durum (*state*) başlığı ile sunucunun canlı olduğu takip edilerek cihaz mesajlarının gereksiz yere internete gönderilmesinin önüne geçilir. Şekil 4.15, cihaz yöneticisi üzerine kayıtlanan farklı uygulamaları göstermektedir.



Şekil 4.15 Cihaz yöneticisine kayıtlanan uygulamalar.

4.4.2 WebSocket üzerinden servis çağrımı

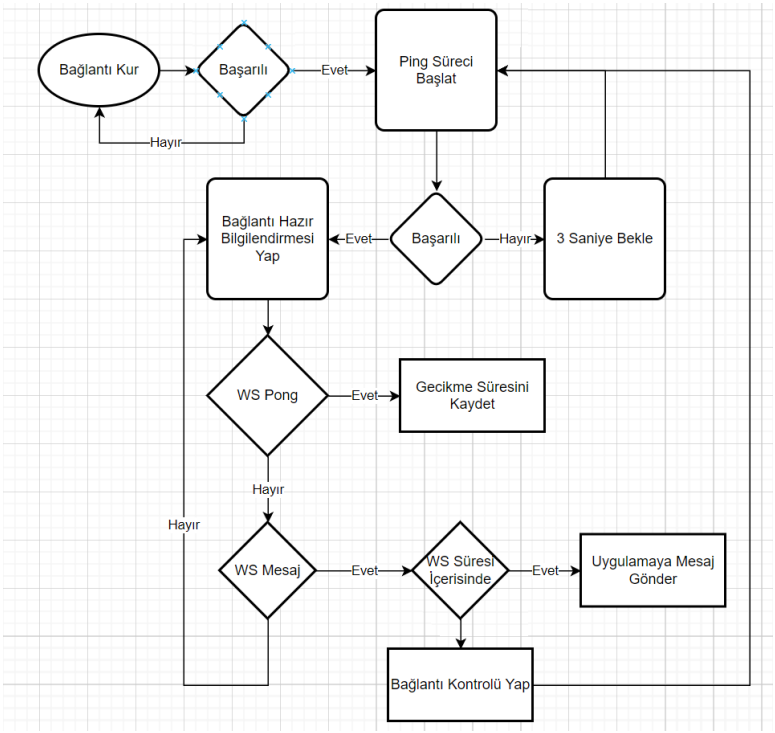
Cihaz yöneticisi, uygulamaların hızlı servis çağrımı yapabilmesi için bölüm 2.2.5' te tanıtılan *WebSocket* protokolünü kullanır. Kurulum anında *WebSocket* sunucusuna bağlantı isteği gönderilir ve bağlantının canlılığı cihaz yöneticisi çalışma süresi boyunca devamlı kontrol edilir. Arka planda devam eden bu kontrol sayesinde bağlantının kapanması veya komuta cevap alamamanın önüne geçilerek uygulamaların gerçek bağlantı durumlarının farkında olması sağlanır. Cihaz yöneticisi bağlantı durumunu kendisine bağlanan bütün uygulamalar için ilgili fonksiyon noktalarına geri bildirim olarak vermektedir.

WebSocket spesifikasyonunda belirtildiği üzere (Bölüm 2.2.5) protokolün 4 bit ile ifade edilen komut tipleri bulunmaktadır. Belirli aralıklarla gönderilen "*WS Ping*" ve cevap olarak karşılanan "*WS Pong*" komutları arasında harcanan gecikme ölçülerek bağlantının canlı olduğu kontrol edilir. Herhangi bir servis çağrımının zaman aşımı ile sonuçlanması durumunda, bu sorgulama yeniden görevlendirilir ve bağlantı kontrolü yeniden yapılmış olur.

WebSocket paketi alfa numerik karakterlerden tanımlanmıştır. Web servisi çağrımında fonksiyon adı ve servis adında geçen parametreler ile servise gönderilecek veri arka arkaya dizilmiş, "," ayracı ile birbirlerinden ayrılmıştır.

Paket örneği: WS,<datetime>,<funcname>,<urlparams>,<dynamic data>

WebSocket çağrımlarında gelen her cevap ayrı bir *callback(on_message)* içerisine geri gönderilmektedir. İlgili mesaj paketine göre cevaplar uygulama katmanına geri gönderilmektedir. Bağlantı kurulmasına, mesajların dinlenmesine ve bağlantı kontrolüne ait akış diyagramı Şekil 4.16'da görülmektedir.



Şekil 4.16 Cihaz yöneticisi WebSocket akış diyagramı.

5. ÖRNEK OLAYLAR

Bu bölümde cihaz yönetici çerçevesinin kullanımı örnek senaryolar üzerinden açıklanmıştır. Cihaz ağı içerisindeki uygulamaların birbirleriyle haberleşebilmesi için otomatik servis keşfi gereklidir. Geliştirilen cihaz yönetici çerçevesi ile çalışacak servislerin *Avahi* kullanımı üzerinden yapılması, mevcut standartta bulunan konum servisi ve standardı genişleten düğüm (*Node*) servisi sırasıyla Bölüm 5.1 ve 5.2'de anlatılmaktadır. Bölüm 5.3'te cihaz yönetimi çerçevesi kullanılarak sefer yönetimine ait örnek bir olay incelenmiştir. Bölüm 5.4, istek/cevap sunucu sorguları için cihaz yönetici çerçevesinde görevlendirdiğimiz *WebSocket* iletişim yaklaşımının, mevcut nesnelerin interneti protokolleriyle karşılaştırmasını içermektedir.

5.1 Konum Servisi Örnek Olayı

Bölüm 4.3.3'te açıklanan servis keşfi yaklaşımı Linux işletim sistemlerinde bulunan *Avahi* uygulaması ile gerçekleştirilmiş, cihaz ağı içerisinde bulunan uygulamaların servislere ait yapılandırma bilgilerine erişimleri sağlanmıştır. *Avahi* uygulamasının görevini yapabilmesi için işletim sisteminde servis klasörü altında ".service" dosyası bulunmalı, içerisinde SRV ve TXT kayıtları eklenmelidir. Örnek olarak konum verisini dağıtan uygulamanın yapılandırma bilgilerine erişmek için hazırlanan konum servisinin içeriği Şekil 5.1'de görülmektedir.

```
<?xml version="1.0" standalone='no'?>
<!DOCTYPE service-group SYSTEM "avahi-service.dtd">
<service-group>
        <name>_gnss_location.</name>
        <service>
                <type>_itxpt_multicast._udp</type>
                <port>14005</port>
                <domain-name>local</domain-name>
                <txt-record>multicast=239.255.42.21</txt-record>
                <txt-record>version=1</txt-record>
                <txt-record>txtversion=1</txt-record>
        </service>
</service-group>
```

Şekil 5.1 Konum servisi dosyası.

```
Found user 'avahi' (UID 998) and group 'avahi' (GID 998).
Successfully dropped root privileges.
avahi-daemon 0.8 starting up.
Successfully called chroot().
Successfully dropped remaining capabilities.
Loading service file /services/GNSS.service.
Loading service file /services/VehicletoIP.service.
Loading service file /services/module_inventory.service.
Joining mDNS multicast group on interface eth0.IPv6 with address fe80::66bc:e181:6efd:98a8.
New relevant interface eth0.IPv6 for mDNS.
Joining mDNS multicast group on interface eth0.IPv4 with address 139.122.100.1.
New relevant interface eth0.IPv4 for mDNS.
Network interface enumeration completed.
Registering new address record for fe80::66bc:e181:6efd:98a8 on eth0.*.
Registering new address record for 139.122.100.1 on eth0.IPv4.
Registering new address record for 192.168.2.16 on eth0.IPv4.
Registering new address record for 139.122.19.10 on eth0.IPv4.
Server startup complete. Host name is imx6ulval6.local. Local service cookie is 2940764403.
Service "imx6ulval6_inventory" (/services/module_inventory.service) successfully established.
Service "_vehicletoip." (/services/VehicletoIP.service) successfully established.
Service "_gnss_location." (/services/GNSS.service) successfully established.
```

Şekil 5.2 Avahi sevisi çalıştırma çıktısı.

Avahi uygulaması, servis klasörü altında bulunan servis dosyalarını sırasıyla çalıştırır. Sonuçta, Şekil 5.2'de görülen çıktı oluşmaktadır. Örnek olarak verilen konum servisi; enlem, boylam, yükseklik ve zaman değerlerini cihaz ağı içerisinde Şekil 5.3'te hazırlanan *XML* çıktısı olarak paylaşmaktadır.

```
Service: _gnss_location.:
Payload is:
<?xml version="1.0" encoding="UTF-8"?>
<GNSSLocationDelivery>
        <GNSSLocation>
        <latitude>38.46002</latitude>
        <longitude>27.18939</longitude>
        <time>372510</time>
        <speed>0</speed>
        <altitude>0</altitude>
        </GNSSLocation>
</GNSSLocationDelivery>
```

Şekil 5.3 Konum servisi verisi.

5.2 Düğüm Servisi Örnek Olayı

Bölüm 2.5.2'de tanıtılan toplu ulaşım için bilgi teknolojisi, cihaz ağı içerisinde hangi servislerin bulunması gerektiğini ve servislerin otomatik olarak bulunabilmesi için gereken yaklaşımları tanımlamıştır. Toplu ulaşım için bilgi teknolojisini temel alarak hazırladığımız cihaz yönetim çerçevesinde, ulaşım araçlarının yönetim konularından biri olan çoklu araç yönetimi için düğüm servisi hazırlanmış ve temel servislerin yanında düğüm servisi ile genişletilmiştir.

Toplu ulaşım araçları için bir kurulum tipi, birden fazla aracının dinamik olarak birleşebildiği konvoy sistemleridir. Ulaşım cihazları tek başına çalıştıkları zaman, belirlenen plaka ve otobüs numarası ile sunucu sisteminden izlenebilmektedir. Ancak bazı koşullarda bir veya daha fazla otobüs körük biçimiyle birbirleriyle birleştiği zaman aynı enerji hattında ve aynı ağ içerisinde bulunmasından dolayı birbirlerinin görevlerini karıştırmaktadır. Otomatik biçimde servis bulunması ile birden fazla otobüsün aynı ağ içinde buluşması durumunda sefere çıkan doğru otobüsün belirlenmesi ve otobüs içerisinde bulunan cihazların çalışma biçimlerinin ayarlanması sağlanmaktadır.

Belirlenen kapsam listesi şu şekildedir:

- Fiziksel yetenekleri ve görevleri bakımından MASTER ve SLAVE tanımlı cihazlar sistemde bulunur.
- Sürücü önünde bulunan, sunucu ile konuşabilen ve sefer yönetimi yapılmasını sağlayan OBU'lar MASTER kabul edilir. MASTER cihazın çalışma kurallarına göre yolcu tarafında bulunan ve ücret toplaması görevinde olan VALİDATOR'ler doğal SLAVE durumundadır. Benzer şekilde ağ içerisinde MASTER olmayan OBU'lar da SLAVE olarak tanımlanmaktadır.
- Sefer durumunda SLAVE cihazlar MASTER cihazın yapılandırmasına göre çalışmaktadır.
    - Uyku durumu
    - GPS bilgileri
    - Sefer bilgileri
        - Hat ve durak bilgileri

- Veri kayıtları
- Ağ geçidi özellikleri

Belirlenen kural listesi şu şekildedir:

- Bir ağ içerisinde sadece bir tane MASTER cihaz bulunur. Bir veya daha fazla SLAVE cihaz aynı ağ içerisinde bulunabilir.
- Uyku durumunda olan OBU, MASTER seçimi sırasında en düşük öncelikte kabul edilir.
- Sefer modunda uykuda olmayan birden fazla OBU ağ içerisindeyse, OBU lar arasından rasgele seçim yapılmaktadır.

Komut listesi şu şekilde sıralanmıştır:

- Kayıt Düğümü (Register Node)
  - Cihaz tipi
  - Kalp atışı (zaman damgası ile)
  - Cihaz sırası
- Master durumu isteme
- Master olduğunu yayınlama
- Master durumunu sorgulama

Çizelge 5.1’de düğüm servisi alanları tanımlanmıştır.

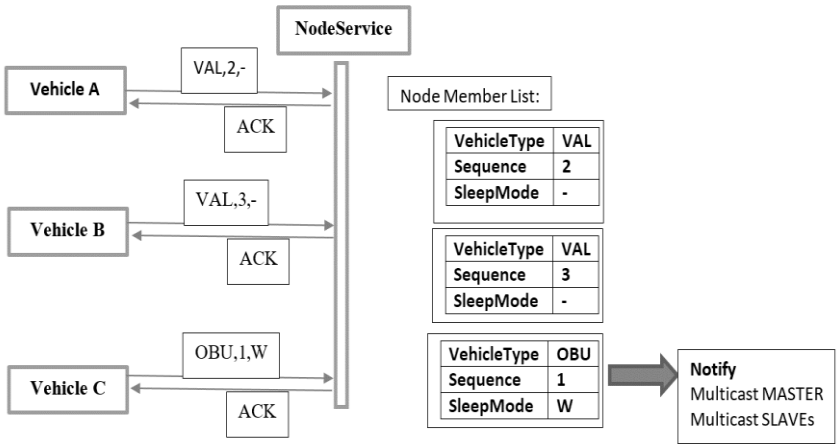
Çizelge 5.1 Düğüm servisi alan tanımları.

Alan	**Tanım**	**Değer**
Birim	[alan adını]'i gösteren özel bir tanımlayıcı	[alan adı]
Servis adı	Servisin uygulama tanımı	düğüm
Tip	Uygulama protokolünün adı (http, sntp, multicast,…)	multicast
Protokol	Servisi taşıyan protokolün adı (tcp veya udp)	udp
Alan	Kaydın geçerli olduğu alan adı	yerel
TTL	DNS geçerlilik süresi	3600
Sınıf	DNS sınıfı (internet için IN)	IN
"SRV"		"SRV"
Öncelik	Öncelik (düşük olan tercih edilir)	0 (ayarlanabilir)
Ağırlık	Aynı öncelik için ağırlık (ağır olan tercih edilir)	0 (ayarlanabilir)
Port	Servisin TCP veya UDP portu	9876
Hedef	Servisi sağlayan cihaz	[alan adı]

Sonuç olarak OBU'nun Master/Slave belirleme servisi şu şekilde cevaplanır:

[hostname]._node._ multicast._udp.local *3600 IN SRV 0 0 9876 [hostname]*

Şekil 5.4'te servisin içerisinde bulunan elemanları etkileşimi ve komutlara göre durum değişimleri gösterilmiştir.

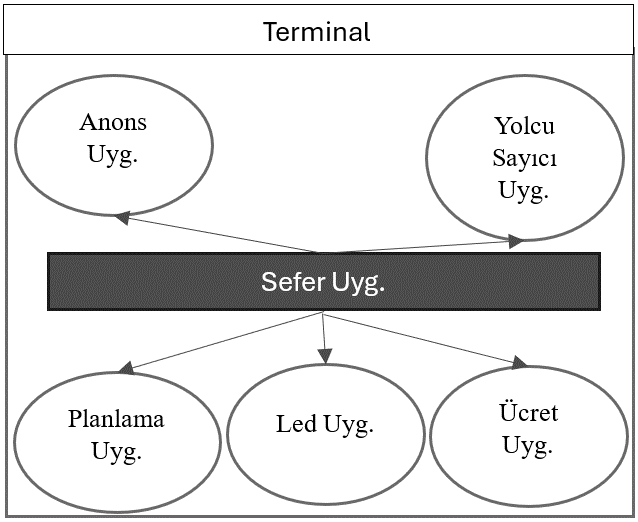


Şekil 5.4 Düğüm servisi obu validator etkileşimi.

5.3 Sefer Yönetimi Örnek Olayı

Cihaz yöneticisinin temel amacı daha verimli ve hızlı veri trafiği sağlamak üzere araç ağında kablolu/kablosuz ağ iletişimlerine dayalı sistemlerin kullanımını mümkün kılan bir çerçeve sağlamaktır. Bu bölümde sefer sürecini ele alarak cihaz yöneticisinin temel özellikleri ve kullanımı anlatılacaktır.

Sefer, toplu taşıma sistemlerinde oluşan verilerin değerlendirilmesi için temel bilgileri içermektedir. İçerisinde gidilen güzergahın bilgisi ve bu güzergâh içerisindeki durakların konumları bulunur. Sistemin çalışması boyunca hem aracın hem de yolcuların davranışlarına göre çeşitli veriler oluşmaktadır. Oluşacak verilere göre araç içerisinde sürücü ve yolcular için birçok bilgilendirme noktası açığa çıkar. Şekil 5.5'te bilgilendirme noktasında görevlendirilen birçok donanım ve uygulama gösterilmiştir. Bu farklı sistemlerin, sefer verilerini kendi içlerinde yönetebilmesi için ham verilere erişmesi ve değişimi takip ederek kendi görevlerini yerine getirmesi beklenmektedir.

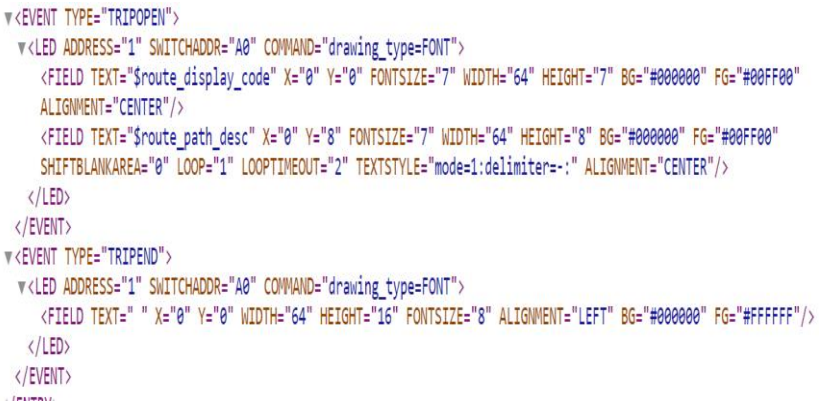


Şekil 5.5 Sefer verisi uygulamaları.

Bölüm 4.3.5'te hazırlanan söz dizimine göre cihaz yöneticisi validatör uygulamasının verilerine göre sefer paketi oluşturmaktadır. Kural, mevcut durumun sürücü girişinde olmasını beklerken sefer açılışı kontrolüne dayalıdır. Sefer açılışı, sefer başlangıç zamanını ve güzergâh kodunun karşılanmasına göre değerlendirilmiştir. İlgili ifade aşağıda gösterilmiştir.

[OS.login= **on** Trip.Opened(trip_time, path_code) : OS.Trip : SendMSG(TRIPINFO)]

Kural çalıştıktan sonra sefer paketi bölüm 4.2'deki hazırlanan bağlam modeline göre pozisyon, zaman damgası ve durum bilgisiyle genişletilerek sistem içerisine dağıtılır. Sefer takip eden uygulamanın sefer açması üzerine üretilen oturum bilgisi ile ağ içi ve sunucu için sefer açtı olayı üretilir. Cihaz yöneticisi bildirimlerine duyarlı olarak çalışan araç içi uygulamalarının, sefer ve durak değişimlerini dinlemesi için hazırlanan girdi dosyaları Şekil 5.6'da görülmektedir.



Şekil 5.6 Led uygulaması için sefer olayları yönetimi.

Cihaz yöneticisi üzerinden dağıtılan anlamlandırılmış mesajlar iki avantaj sağlamaktadır:

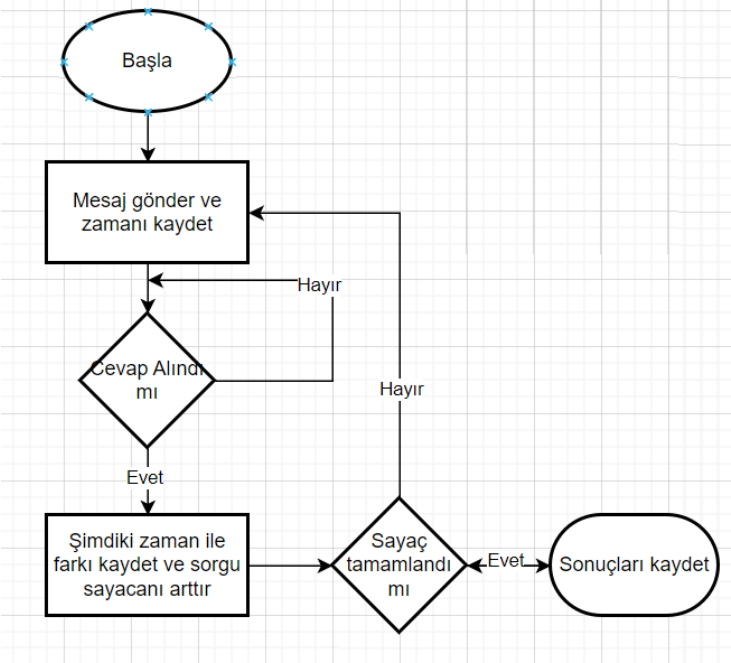
1) Bilgi sağlamlığı artar: Sensörlerden gelen veriler sefer durumuyla tek noktada ilişkilendirilerek bütün paydaşlar için eş verinin, anında kullanılmasını sağlar.

2) Veri filtreleme sağlanır: Ham verilerin filtrelenerek daha özet biçimde dağıtılmasıdır. Böylelikle, üst katmanlara iletilen mesajların boyutu ve sayısı azılır, ağ trafiği düşer. Mevcut çözümlerde, sistem paydaşlarının temel sefer verisine doğru zaman/konum değeriyle sahip olması ve düzenli olarak güncelliğinin kontrol edilmesi gerekirken, önerilen cihaz yönetimi yaklaşımıyla temel sefer verilerinin güncelliği, tek nokta üzerinden dağıtılması bakımından, bütün uygulamaların aynı veriye göre çalıştığı garanti edilir.

5.4 Servis Çağrımı Örnek Olayı

Akıllı ulaşım sistemlerinde yer alan servislerdeki durum değişikliklerinin sunucuya iletilebilmesi veya sunucunun bir komutu cihaza anında gönderilebilmesi için gerçek zamanlı iletişim gereklidir. Bu doğrultuda geliştirilen cihaz yöneticisi çerçevesine eklenen çift yönlü haberleşme alt yapısıyla servis çağrılarının hızlandırılması hedeflenmiştir. Ayrıca, cihaz yöneticisini kullanan uygulamalar, servis çağrımları öncesinde bağlantı durumunu kontrol ederken servis çağrımında geçen sürenin kontrolünü de yapabileceklerdir.

İletişim protokolleri karşılaştırmalarında sıklıkla tercih edilen yöntem, iletilen mesajın onayının alınmasıyla veri gönderimi arasındaki geçen sürenin ölçümü olan gidiş geliş (*round trip time, rtt*) süresidir (Postel, 1981). Gidiş geliş süresini ölçebilmek için cihaz yönetici çerçevesi içerisinde Şekil 5.7'de görülen algoritma kullanılmıştır.



Şekil 5.7 Gidiş geliş süresi ölçme algoritması.

Cihaz yöneticisinin performansını değerlendirmek amacıyla http, mqtt ve websocket protokolleri üzerinde Şekil 5.7'de görülen algoritma uygulanarak serviste geçen süreleri ihmal etmek için zaman eşleme servisi yazılmış ve her bir protokol üzerindeki sonuçlar karşılaştırılmıştır.

5.4.1 HTTP yöntemi

HTTP gibi istemci-sunucu protokollerinde oturumlar üç aşamadan oluşur:

i. İstemci TCP bağlantısı kurar.
ii. İstemci isteğini gönderir ve cevap bekler.
iii. Sunucu isteği işler, bir durum kodu ve uygun verileri içeren cevabını geri gönderir.

Anlık değişimlerin istemci tarafına iletilmesi için sunucu servisleri istemci tarafından sürekli sorgulanmalıdır. Sorgulama ve alınan cevaplar belli sırada paketlerin karşılığına göre işlenir. *TCP* bağlantı kuralları doğrultusunda ve toplu taşıma sistemlerinin ağırlıkla *3G/LTE* bağlantı hızları kullanması bakımından en basit servise ulaşması için standart iletişim gecikmelerini kabul etmesi gerekmektedir.

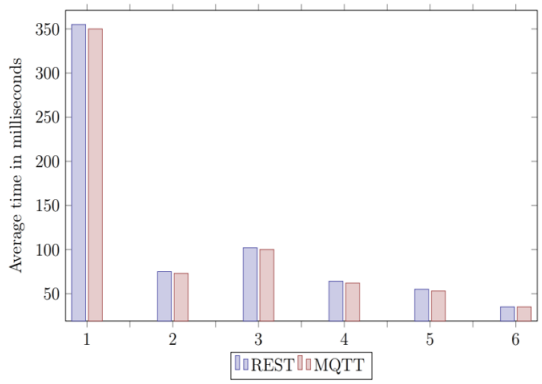
3G bağlantı hızında bağlantı kurulması ve cevap paketinin cihaza geri dönmesi ölçüldüğü zaman 3sn gecikme olduğu görülmüştür. Bu gecikme gerçek zamanlı iletişimin önünde önemli bir problem oluşturur. Diğer problem ise sürekli sorgulama ile sunucu sistemlerinde yükün gereksiz yere arttırılmasıdır.

5.4.2 MQTT yöntemi

Var olan *HTTP* yaklaşımı Manandhar (2017) çalışmasında *MQTT* üzerinden gönderilecek şekilde tasarlanmıştır. İstemci *MQTT* başlığı altında ilgili sunucu fonksiyonu cevap kodlarına kayıtlanır (örneğin 200 OK başlığı) ve başka bir başlık altında da sorgusunu gönderir.

[subscribe: /devices/post/response_id] [Publish: /devices/get/response_id "{status:200}"]

Çalışmasını *MQTT* ve *REST* tercihlerine göre cihazlarının kayıtlanması, sunucu tarafından cihaz listesinin çekilmesi, uygulama yönetimi ve nesnelerin interneti cihazlarında uygulama çalıştırılması başlıklarında değerlendirmiştir. Elde edilen sonuçlar Şekil 5.8’de gösterilmiştir.



1. Nİ cihaz kayıtlanması
2. Cihaz listesinin çekilmesi
3. Cİhaza uygulma yüklenmesi
4. Cihazda uygulama güncellenmesi
5. Cihazda uygulama silinmesi
6. Cİhazda uygulama çalıştırılması

Şekil 5.8 MQTT ve REST ortalama zaman karşılaştırması (Manandhar, 2017).

Önerilen yaklaşımı kullanarak 3G bağlantı hızında bağlantı kurulması ve cevap paketinin cihaza geri dönmesi ölçüldüğü zaman 1 sn gecikme olduğu görülmüştür. Gecikme kabul edilebilir seviyelere inmesine rağmen bir sorgu için hazırlanması gereken kod boyutu artmakta ve kısıtlı kapasiteye sahip cihazlarda yönetilmesi zorlaşmaktadır.

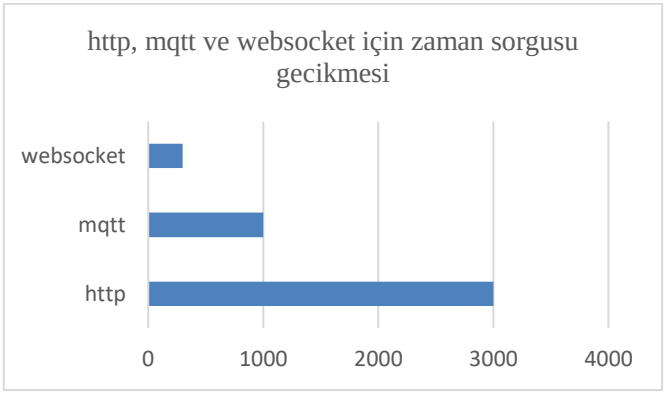
5.4.3 WebSocket ile önerilen yaklaşım

Bölüm 4.4.2'de tanıtılan *WebSocket* kullanımı, uygulamaların sunucu ile çift yönlü bağlantıda olması sağlayarak sunucudan gelecek anlık emirlerin dinlemesine olanak tanımaktadır. Aynı bağlantı üzerinden uygulamaların servis çağrımları gerçekleştirilmiştir.

Literatürde *WebSocket* ile *MQTT* karşılaştırmaları iletim gecikmesine, yazılan kod satırına ve gönderilen mesajın büyüklüğüne göre değişmektedir. Oliveira ve ark. (2018) çalışmalarında, iletim gecikmesini gömülü yazılım geliştirilen cihaz üzerinde gerçekleştirerek ortalama değerleri hesaplamışlardır. Yapılan çalışmanın sonucunda *WebSocket* için ortalama 10 ms altında iletim gecikmesi ölçülmüştür. *WebSocket* aksine *MQTT* kullanımında gönderilen mesaj boyutunun arttırılması durumunda gidiş geliş süresinin de aynı şekilde yükseldiği gözlemlenmiştir.

Önerdiğimiz yaklaşım kullanılarak zaman servisi çağrıldığında iletişim gecikme süresi *3G* bağlantı hızından 300 ms seviyelerine inmiştir. Benzer iletişim *4G* üzerinden kurulduğu zaman harcanan süre 100 ms içerisine inmektedir.

İletişim protokolleri karşılaştırıldığında Şekil 5.9'da görülen dağılım ortaya çıkmıştır.



Şekil 5.9 http, mqtt ve websocket için zaman sorgusu gecikmesi.

6. DEĞERLENDİRME VE TARTIŞMA

Nesnelerin interneti sistemlerinde standart iletişim protokollerinin ve ortak veri formatlarının benimsenmesi, cihazların insan müdahalesi olmadan birbirleriyle iletişim kurup iş birliği yaptığı makineler arası iletişim paradigmasının ortaya çıkmasını sağlamıştır (Pandey et al., 2011). Ancak makineler arası iletişimi tam olarak etkinleştirmek için, heterojen sistemler arasında tam bir uyumluluk sağlayan, iyi tanımlanmış arayüzlere sahip eksiksiz bir ağ mimarisi ve mimari üzerinde desteklenen servis tanımları gereklidir. Bu yöndeki güçlü bir standardizasyon çabası, Avrupa Telekomünikasyon Standartları Enstitüsü (ETSI) tarafından akıllı makineler arası iletişimleri teknik komitesi bünyesinde yürütülmektedir (Kushwah et al., 2020). Bu komite, makineler arası iletişimleriyle ilgili gereksinimleri, ağ oluşturmayı, protokolleri ve güvenlik gereksinimlerini ele almaktadır. Standardizasyon çabalarına rağmen, bilgi teknolojilerinde yapılan gelişmeler kısıtlı kaynaklara sahip gömülü sistem cihazlarının internete doğrudan bağlanmasıyla, makineler arası iletişim konularının nesnelerin interneti sistemlerinde yeniden değerlendirilmesini zorunlu kılmıştır. Dağıtık çalışan heterojen ortamlarda merkezi olarak verilerin yönetilmesi, sistem içerisinde akan verinin ortak bir yaklaşımla anlamlandırılmasını sağlamaktadır. Alt seviye donanımların haberleşmesindeki standartlaşma, kapalı devre çalışan ortamlarda bulunan uygulamaların daha kolay yönetilmesine yardımcı olmaktadır. Bu hedefle toplu taşıma sistemlerine özelleşen nesnelerin internetiyle uyumlu yaklaşımlar tanımlanmıştır. Toplu taşıma alanına özgü çalışmalar pratik geliştirme süreçleri olarak standartlaşsa da henüz yeteri kadar çalışma yapılmamıştır.

Bu doğrultuda çalışmamızda ulaşım sistemleri için bir cihaz yöneticisi çerçevesi hazırlanmıştır. Cihaz yöneticisi, merkezi şekilde uygulamalar arasında konumlanarak uygulamacıların ulaşım sistemi durum ve verilerine göre görevlerini yapabileceği bağlam verilerini oluşturmaktan sorumludur. Temel seviyede ulaşım sistemlerinde dolaşan veriler tanımlanmıştır. Bu veriler otobüs yönetimi, alarm, kafa sayıcı, yolcu bilgilendirme ve araç algılayıcı (sensor) verilerini içermektedir.

Cihaz yöneticisi, toplu ulaşım için bilgi teknolojisi standardı temel alınarak hazırlanmış ve tezde öncelikle standartta belirtilen servisler gerçekleştirilmiştir.

Mevcut servislerin yanında cihaz yönetimi için gerekli yeni servis tanımları yapılmıştır. Hazırlanan düğüm servisi, toplu ulaşım sistemlerinde bulunan OBU ve validatorler için çalıştırılmış, bu doğrultuda birden fazla OBU ve validator bulunan sistemlerde düğüm servisi ile cihaz tanımları otomatik olarak tespit edilebilmiş, ağ içerisindeki diğer cihazlara anlık değişimler ve mevcut cihaz tanımları aktarılabilmiştir. Konvoy araçlarının yerel ağ üzerinden çok noktaya yayın servisi ile yönetilmesi gömülü sistem uygulamaları için kolaylıkla geliştirilmesini ve yönetilmesini sağlarken, toplu ulaşım için bilgi teknolojileri standardına uygun olarak gönderilen olay tabanlı mesajlarla sistemin çalışma anı değişimlerine hızlı cevap vermesi sağlanmıştır. Düğüm servisi hem otobüs hem de tren formunda çalışan sistemlerde sefer durumlarına göre kullanıcı müdahalesi olmadan istenen hizmetleri vermesini sağlamaktadır.

Geliştirilen çerçevede, gömülü sistem uygulamalarının bağımsız ve beraber çalışabilmesi için cihaz yöneticisinin olay değişimlerine duyarlı olarak mesajlar oluşturması sağlanmıştır. Cihaz yöneticisi uygulamalar arasında merkezi olarak veri dinleyen ara birim yazılım konumdadır ve kullanımıyla ağ içerisinde oluşan bağlam bilgisi olay, durum ve kural düzeyinde birleştirilir. Veri temelli olay yaklaşımı ile birbirinden bağımsız çalışan uygulamaların sık paylaşılan verileri merkezi olarak işlenmektedir. Yerelde işlenen mesajlar ile sunucuya gereksiz veri gönderimi engellenmektedir. Böylece gerçek zamanlı verilerin cihaz içerisinde işlenmesi sağlanırken bağlama duyarlı olarak genişletilen olay paketleri hem ağ içerisinde hem de sunucu tarafından belirtilen kurallara göre dinamik olarak yönetilmektedir. Araç içerisindeki mesajlar, yayıncı/abone yaklaşımlarından araç ağı için en uygunu olan başlık tabanlı mesajlaşma ile dağıtılmaktadır. Bu yaklaşım cihaz içi uygulamaların anlaşılan başlıklarla birbirleriyle haberleşebilmesini sağlarken, cihaz yönetimi konusunda sunucuyu ilgilendiren olayların gönderilmesinde de aynı mesaj paketlerinin kullanılabilmesine yardımcı olmuştur. Böylece tek gerçekleştirim ile ortak altyapı kullanımı sağlanır. Bildirimler doğrudan mesaja kayıtlanan aboneye aracı üzerinden iletilirken, yönetici yazılım birimi bağlama göre değerlendirme yaparak yeni olayların tetiklenmesini kontrol etmektedir.

6.1 Kalite Ölçütleri

Bilgi ve iletişim teknolojilerin kullanılmasıyla genişleyen ulaşım sistemleri cihaz seviyesinden sunucu seviyesine kadar farklı katmanları içerir ve her katmanda farklı bileşenler (uygulamalar) görevlendirilmiştir. Bu doğrultuda akıllı ulaşım sistemleri yaklaşımlarının temel amacı katmanlar arasında ortak dilin belirlenmesi, donanım ve iletişim teknolojilerinin farklılığının ortadan kaldırılmasıdır. Hazırladığımız cihaz yöneticisi ulaşım sistemlerinin en alt katmanındaki donanımlar üzerinde çalışan uygulamaların koordinasyonu ve etkileşiminden sorumludur. Bu hedefle aşağıda listelenen kalite ölçütleri üzerinde durulmuştur.

- Gerçek zamanlılık: WebSocket desteği ile sunucu ile çift taraflı haberleşme kanalı açılarak uzun süren sunucu sorguların hızlı cevaplanması sağlanmıştır. Bölüm 5.4 tanıtılan literatürdeki çalışmalar ve diğer iletişim yaklaşımlarıyla olan karşılaştırmalar WebSocket kullanımının ulaşım cihazları ihtiyaçlarını gerçek zamanlı karşıladığı görülmüştür. 4G bağlantı hızında 500 ms altında iletişim hızları elde edilmiştir. Ayrıca, Bölüm 4.3.2'de açıklanan cihaz ağı içerisinde bulunan uygulamaların olay tabanlı mesaj paylaşımı yöntemi, araç durumlarının ve verilerinin değişim yayınlamasını çok noktaya aynı anda yapıldığını göstermektedir. Cihaz ağı içerisinde uygulamaların oluşturulan olaylara erişme süreleri 50ms içerisindedir.
- Evrensel standartlar: Literatürde nesnelerin interneti sistemlerine ait hazırlanan referans mimariler üst seviye katmanları gruplayarak her katmanın yeteneklerini belirlemektedir. Bu doğrultuda referans alınan toplu ulaşım için bilgi teknolojisi standardı, ulaşım sistemlerinde bulunan her servisin farklı üreticiler tarafından hazırlanabileceğini göz önünde bulundurarak teknoloji kullanımında standartlaşmanın önemini vurgular. Bu doğrultuda Bölüm 4.3.3'te tanıtılan servis keşfi yönteminin önerdiğimiz çerçevede kullanılmasıyla cihaz ağı içerisinde çalışan birbirinden farklı servislerin elle ayarlama gerektirmeden görev alabilmesini sağlamaktadır. Cihaz yöneticisi cihaz ağı içerisinde uygulamalar ile haberleşmeyi çok noktaya yayın (UDP, multicast) protokolü ile yaparak uygulama geliştiriciler için kolay bir arayüz sağlamaktadır. Uygulamaların verilerini sunucuya

göndermesi için cihaz yöneticisi iletişim yöntemi olarak Bölüm 4.4'te tanıtılan iletişim modelini kullanmakta ve verinin sunucular tarafından kolaylıkla işlenebilmesi için *MQTT Sparkplug* protokolünü desteklemektedir.

- Ölçeklenebilirlik ve Kolay Genişleme: Cihaz yöneticisi ömrü boyunca farklı uygulamaların verilerini paylaşması, olay ve durum değişimi yayınlaması için kullanılırken yeni eklenecek uygulamaların cihaz yöneticisine ait alt yapı haberleşme bilgisine ihtiyacı olmamıştır. Geliştirilen çerçevede çalıştırılan 5 ayrı uygulama, değişken veriler olan pozisyon ve durum değerlerini aynı değerleriyle erişmektedir.
- Veri zenginleştirme: Cihaz yöneticisi, Bölüm 4.2'de tanıtılan bağlam desteğiyle basit verilerin sistem durumuna göre daha anlamlı hale gelmesini sağlamıştır. Başlık tabanlı olarak dağıtılan olay paketleri cihaz tanımlayıcıları, pozisyon bilgileri, sistem durum bilgisini ve alarm durumlarını desteklemektedir.
- Bilgi sağlamlılığı: Araç içerisindeki uygulamaların dağıtık çalışması sebebiyle uygulamaların oluşturduğu olaylar için zaman ve pozisyon bilgisinin uygulamaya göre değişmemesi önemlidir. Böylece farklı uygulama olayları için mesajlar birbirleriyle tutarlı zamanı verecektir. Bu kapsamda her olay bağlam tanımını içeren verilerle birleştirilmiştir. Hazırladığımız cihaz yönetim çerçevesi, oluşan paketler için dağıtılan verinin bütün uygulamalarda aynı olmasını sağlamaktadır.
- Merkezi veri biriktirme: Uygulamalar verilerini başlık tabanlı mesajlaşma yöntemiyle dağıtmaktadır. Merkezi olarak yerleştirilen cihaz yöneticisi, verileri uygulamaların kullanmasına göre paketlerken sunucu üzerinden izlenebilmesini sağlamaktadır. Disk üzerinde paketlerin saklanması durumunda paket boyutları 1.5 kat küçülmüştür. Merkezi veri biriktirme yöntemiyle kuralların işlenebilmesi sağlanırken sunucuya gönderilen paketlerdeki değişimlerden uygulamaların yalıtılması sağlanmaktadır.

6.2 Cihaz Yöneticisinin Uygulandığı Donanım

Uygulama çalışmaları Kent Kart Ege Elektronik San. Tic. A.Ş. (Kentkart) Ar-Ge merkezinin ürettiği sürücü bilgisayarları üzerinde gerçekleştirilmiştir. Elektronik ücret toplama, araç takip, gerçek zamanlı yolcu bilgilendirme, planlama ve araç içi kamera sistemleri alanlarında toplu taşıma bilgi sistemleri üreten bir teknoloji firmasıdır. Yurt içinde 20'den fazla yurt dışında Amerika, İtalya, Katar, Pakistan gibi 10'dan fazla uluslararası bölgede hizmet vermektedir.

Tez kapsamında standartlaşan ulaşım servislerinin gerçekleştirimi ve mesaj odaklı cihaz yöneticisi özellikleriyle genişletilmesi, Kentkart'ın ürettiği toplu ulaşım donanımları (OBU) üzerinde hazırlanmıştır. OBU'nun sistem özellikleri Çizelge 6.1'de belirtilmiştir:

Çizelge 6.1 Kentkart OBU sistem özellikleri.

İşetim Sistemi	Linux
İşlemci	ARM Cortex A8
Hafıza	1 GB
Disk	512 MB
İletişim	Ethernet, Wlan, GPRS
Donanım Protokolleri	RS485, seri port, CAN

6.3 Cihaz Yöneticisinin Benzer Sistemler ile Karşılaştırılması

Cihaz yöneticisinin veri temelli iletişim yaklaşımı, Meng ve arkadaşlarının (2017) veri paylaşımı üzerine hazırladıkları mesajlaşma modeliyle karşılaştırıldığında, kurdukları modelin cihazın tespit edilmesi ve iletişimin kontrol edilmesiyle sınırlı olmasından ötürü daha ilkel makinelerin bulunduğu ortamlara uygun olduğu görülmüştür. Heterojen uygulamalardan gelen verileri merkezi olarak anlamlandıran cihaz yöneticisi sistemde çalışan diğer uygulamalara anlamlı mesajlar üretebilmektedir.

Akıllı ulaşım sistemleri için makineler arası haberleşme çözümü olarak hazırlan *ICSI* (*Intelligent Cooperative Sensing for Improved traffic efficiency*) ara yazılımı, temsili durum aktarımı (*REpresentational State Transfer*) modeliyle yeniden yapılandırma yeteneğini cihazlara kazandırmaktadır (Azzara et al., 2015). İletişim protokolü olarak tercih edilen *CoAP*, gerçek zamanlılık ve gönderilen mesajın sağlamlığı bakımından cihaz yöneticisi çerçevesinin kalite ölçütleri dışında kalmıştır.

Cihaz ağı içerisinde iki grup uygulama bulunur:

1) Sistem uygulamaları: Sistem durumuna göre kendi görevini tamamlayan yolcu bilgilendirme, anons, sürücü bilgilendirme gibi uygulamalarıdır. Bu uygulamalar sefer açılışı ve durak geçişlerine kayıtlanarak kendi görevlerini yerine getirirler.

2) Algılayıcı verilerini yöneten uygulamalar: Cihaz durumundan bağımsız algılayıcı verisini yayınlarlar, cihaz yöneticisi bu veriyi cihaz durumu ve pozisyon bilgisiyle ilişkilendirerek dağıtmaktadır.

Toplu ulaşım için bilgi teknolojisi standardı ilk grup uygulamalar için gerekli olan verileri tanımlayarak üreticilerin veri paylaşma arayüzlerini standartlaştırmaktadır. Standarda uyan uygulamaların bulunduğu sistemlerde bizim yapımıza benzer şekilde sonuca ulaşılır. Ancak, araçlar birçok farklı üreticilerin donanımlarından oluştuğu zaman yalınlaştırma noktası olarak cihaza eklenen cihaz yöneticisi farklı uygulamaların anlayacağı şekilde yapılandırılabilir.

İkinci grup uygulamalar için ulaşım standardının bir önerisi bulunmamaktadır. Dağıtık veri yönetimi kapsamında bu tarz kuralların sunucu tarafında sonradan yapılması önerilmiştir. Ancak gerçek zamanlı sistemlerde farklı verilere göre üretilen olaylar, sistem takibini ve sunucu işlerini kolaylaştırmaktadır.

6.4 Performans Değerlendirme

Toplu ulaşım araçları belirli bir rota boyunca belirli durakları takip ederler. Böylesi bir sistemde hem araç içinde hem de araç dışında durak takibi yapan birden

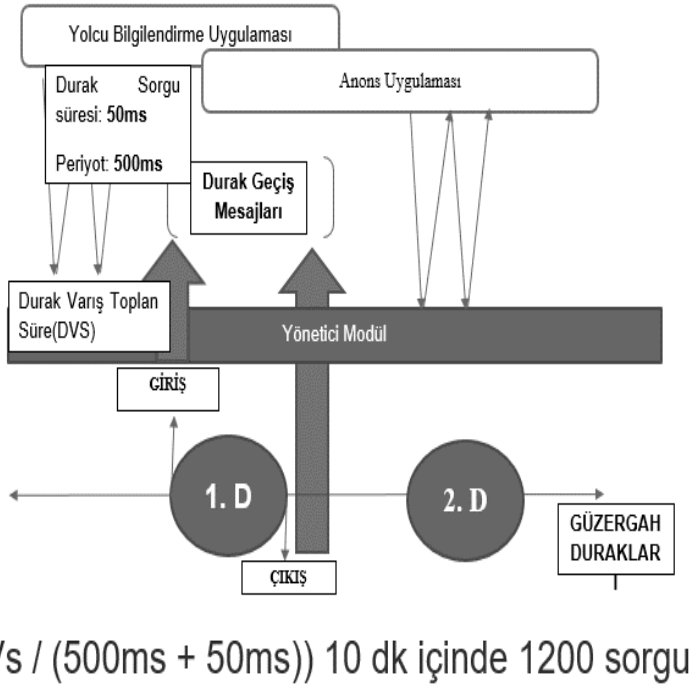
fazla uygulama çalışır. Araç içinde çalışan uygulamalara örnek olarak; durak geçişlerine göre kapı sensörlerinin çalışmasını izleyen uygulamalar, yolcu bilgilendirme kapsamında durak giriş/çıkış anonslarını veren uygulamalar ve led/lcd paneller üzerinde durak geçişlerini gösteren uygulamalar verilebilir. Araç dışında ise duraklara veya raylı sistem istasyonlarına konumlandırılmış led/lcd panelleri ve filo takibi yapan sunucu uygulamaları ile yolcuların elinde bulunan mobil uygulamalar verilebilir. Her iki durumda da uygulamaların görevlerini yapması için ihtiyaç duyulan veri aynı olsa da bu uygulamalar araç içi ve sunucu uygulamaları olarak farklılaşmaktadır.

Sistemde durak takibi yapılabilmesi için rota ve durak konum bilgilerinin bilinmesi, araç hareket ettikçe konum bilgisinin güncellenmesi ve ulaşılan durak bilgisine göre uygulamaların kendi görevlerini yerine getirmesi beklenir. Bu yetenekler kapsamında sistemde anons uygulaması, araç içi led uygulaması ve durak led uygulaması çalıştığı düşünülmüştür.

Araç içerisinde bulunan OBU, ağ geçidi özelliğinde bulunarak sistemden rota ve durak bilgilerini yükler, konum servisiyle haberleşir ve durak takibini yapan bir uygulamaya sahiptir. Yönetici uygulama mesaj kanalı üzerinden değişimleri yayınlarken çok noktaya yayın protokolüyle yerel ağ üzerinden gelen istekleri de cevaplamaktadır. Bu görevleri yerine getirmesi ve diğer donanımlarla konuşabilmesi kabul edilerek, mesaj tabanlı çerçeve yaklaşımının kullanılması karşılaştırılmıştır.

6.4.1 Çerçeve kullanımına bağlı karşılaştırma

Sistem içerisinde mesaj tabanlı çerçeve uygulanmadığı zaman çevresel uygulamalar durak değişim bilgilerini alabilmek için yönetici katman ile doğrudan haberleşmek durumundadır. İstek ve cevap sorguları ile oluşan ortam Şekil 6.1’de gösterilmiştir.

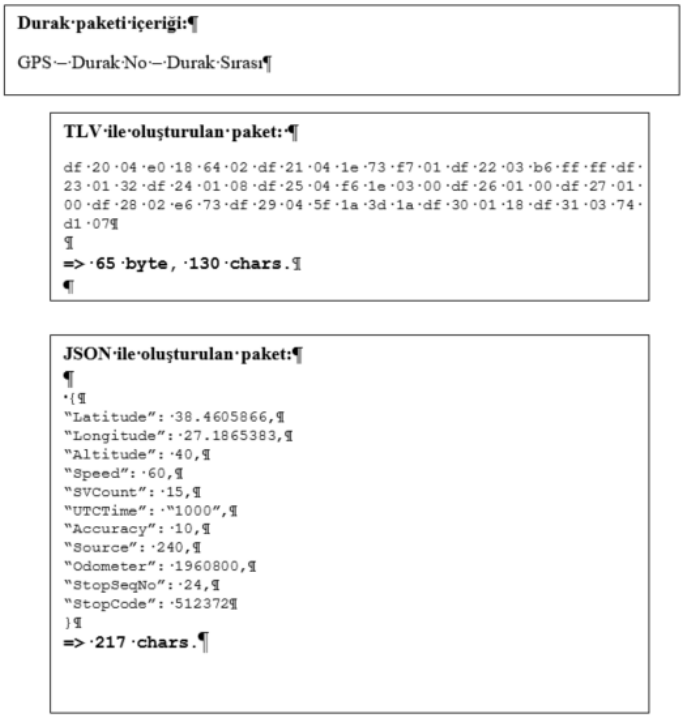


Şekil 6.1 Çerçevenin uygulanması.

Durak bilgisi edinme sorgusu ağ içerisinde 50 ms içerisinde cevaplanmaktadır. Durak geçişlerini takip etmek için 500 ms döngülerle sorgulama yapıldığında uygulama başına durağa ulaşana kadar geçen sürede (DV_s) sorgu sayısı (DV_s / (500 + 50)) olarak ortaya çıkar. Örneğin ortalama 10 dk içerisinde durak geçişlerini tespit etmek için sorgu süresi ihmal edildiğinde, 1200 sorgu da 1 verimle çalışmaktadır. Uygulamalar ve ihtiyaç duyulan veriler arttıkça yönetici birim üzerindeki hizmet süresi de artmaktadır.

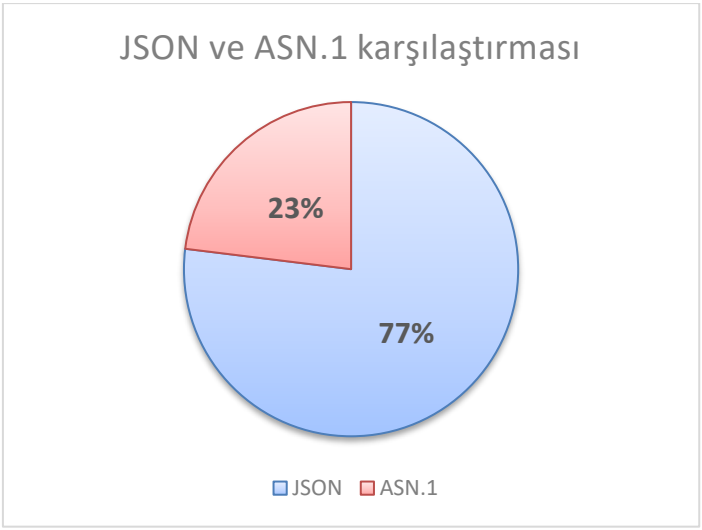
6.4.2 Paket türüne bağlı gecikme

Çerçeve içinde verileri serileştirme işi için *ASN.1* kodlamasının kullanılma sebebi, kendini tanımlama özelliği, kolay kullanılır olması ve kolay genişletilebilir olmasıdır. *ASN.1* kodlamasıyla sistemlere özel veri sözlüğünün oluşması ikincil katkıdır. Paket boyutlarına göre kıyaslama *MQTT* için sıklıkla tercih edilen *JSON* formatı ile yapılmıştır. Şekil 6.2 durak girişi için oluşturulan paketlerin karşılaştırması vardır.



Şekil 6.2 Durak paketi serileştirme yöntemi karşılaştırma.

Gönderilen veriler aynı olmasına rağmen ASN.1 tercih edildiğinde 1.5 kattan fazla veri harcaması azaltılmaktadır. Paket boyutları bakımından karşılaştırma sonuçları Şekil 6.3'te görülmektedir.



Şekil 6.3 Paket boyutlarına göre karşılaştırma sonucu.

7. SONUÇ

Tez kapsamında geliştirilen cihaz yönetim çerçevesi, akıllı ulaşım sistemlerinde yer alan cihazlar arası veri alışverişini ve birlikte çalışabilirliği destekleyerek, cihaz ağında dolaşan verinin sadeleşmesi, veri miktarının azaltılması, olaya duyarlı gerçek zamanlı paylaşılması ve uygulamaların birbirleriyle ve sunucu ile haberleşme altyapısını yalıtması açısından önemli iyileştirmeler sağlamaktadır. Akıllı ulaşım sistemlerine konu olan cihazlardaki uygulamalar, cihaz içerisindeki algılayıcılardan, araç motorundan ve çevresel uygulamalardan elde edilen veriler ile çalışmaktadır. Örneğin motor sensörünü denetleyen bir uygulama kendi tasarımına göre cihaz içerisinde oluşan durumları denetleyerek tespit ettiği bir arızaya karşı cihaz durumunu değiştirirken ilgili ekranlarda da uyarılar vermektedir. Birden çok algılayıcıdan elde edilen verilere göre karar alınabilmesi için dağıtık sistem tasarımı gerekir. Ağ içerisindeki verilerin olaya duyarlı biçimde yayınlanması, birçok uygulamanın bağımsız bir şekilde çalışmasını sağlar.

Geliştirilen cihaz yönetim çerçevesinin faydalarının görüldüğü bir diğer konu, verilerin uygulamalar arasında farklı şekilde anlamlandırılabilmesidir. Akıllı ulaşım sistemlerinde birden çok algılayıcı verisine göre özelleşerek çalışan uygulamalar, daha karmaşık olayları kendi servislerine göre anlamlandırabilir. Aynı zamanda bu verilerin ağırlıkları ve öncelikleri, farklı uygulamalara farklı şekilde yorumlanabilir. Örneğin bir uygulama için bir algılayıcıdan elde edilen değer yüksek olarak değerlendirilirken başka bir uygulama için düşük olarak kabul edilebilir. Bu verilerin cihaz içerisinde işlenmeden yayınlanması hem karmaşaya hem de cihaz ağında yoğun veri akışına neden olmaktadır. Tez kapsamında geliştirilen cihaz yönetim çerçevesi ile, bu problemi çözmek için uygulama bağımsız bir ara birim yazılım katmanı görevlendirilerek, ham veriler merkezi bir şekilde anlamlandırılmış ve araç içerisinde daha anlamlı olarak dağıtılması sağlanmıştır. Bu doğrultuda akıllı ulaşım sistemlerine ait referans mimariler incelenerek temel seviyede ulaşım sistemlerinde dolaşan otobüs yönetimi, alarm, kafa sayıcı, yolcu bilgilendirme ve araç algılayıcı (sensor) uygulama verileri tanımlanmıştır. Bu veriler, Bölüm 2.5.2'de tanıtılan toplu taşıma için bilgi teknolojisi standardına uygun olarak nesnelerin interneti mimarisi yaklaşımıyla

cihaz ağı içerisinde merkezi olarak işlenmektedir. Bölüm 3.1'de açıklanan nesnelerin interneti ortamları için cihaz yönetimi gereksinimlerine göre dağıtık servislerin evrensel standartlar ile birbirleriyle haberleşebilmesi, kolay kurulum ve bakım yapılabilmesi üzerinde durulmuştur. Cihaz yöneticisi, uygulamaların bağımsız ve beraber çalışabilmesi için olay değişimlerine duyarlı bir bağlam veri modeline sahiptir. Bağlam veri modeli, akıllı ulaşım sistemlerine ait uygulama ihtiyaçlarına göre değerlendirilerek pozisyon, zaman ve durum verisini içermektedir. Böylelikle cihaz ağı içerisinde oluşan her mesajın olaya duyarlı, hızlı ve veri üzerinden takip edilmesi sağlanmıştır. Üretilen mesajlar araç içerisinde tüketilirken uygulamaların veri ve durum değişimleri sunucu üzerinden takip edilebilmektedir. Geliştirilen cihaz yönetici çerçevesi, Bölüm 5.3'te açıklanan sefer yönetimi örnek olayı ile çalıştırılmış ve ağ içerisindeki mesajların anlamlandırılmasıyla iki avantaj elde edilmiştir. Algılayıcı verileri sefer durumu değişimiyle merkezi noktada ilişkilendirilerek ağ içerisinde çalışan bütün uygulamaların aynı veriyi kullanması sağlanırken paylaşılan verinin farklı uygulamalar için tekrar edilmesinin önüne geçilmiştir.

Cihaz yönetim çerçevesinin içinde yer alan ara birim yazılım katmanıyla elde edilen avantajlar aşağıda özetlenmiştir:

- Ağ bağlantılarının sadeleşmesi: Sistem içerisinde çalışan uygulamalar bütün ağın kontrolünden sorumlu değildir. Birbirinden bağımsız olan uygulamaların birbirleriyle doğrudan bir bağlantısı olmadığı için ağ bilgilerini bilmelerine de gerek olmamıştır. Geliştirilen cihaz yönetici çerçevesiyle, yeni kurulan bir bağlantı durumunda sistemin durumu ve o anki verisi doğrudan ara birim yazılımı üzerinden yapılmaktadır.
- Asenkron iletişim: Mesajlar, geliştirdiğimiz cihaz yöneticisi üzerinde saklandığı için noktadan noktaya iletişim ihtiyacı ortadan kalkmış, uygulamaların durum değişimlerine kayıtlanması sağlanmıştır. Sunucu ile haberleşme altyapısı cihaz yöneticisi içerisinde tutularak uygulamaların ayrı sunucu bağlantı açmalarının önüne geçilmiştir.
- Ölçeklenebilirlik: Veri modeli dinamik olarak mesajların büyümesine olanak sağladığı için mesajların değişmesi bütün sistemi etkilememiş,

uygulamaların artması veya mesajların içeriğinin genişlemesi durumunda geriye dönük uyumluluk problemleri engellenmiştir.

- Merkezi veri kontrolü: Sistem üzerinde akan veriler ara birim yazılım katmanı üzerinde biriktirildiği için sistem paydaşları istediği mesajı ilgili kurallar üzerinden almaktadır. Mesajları oluşturan kuralların değişmesi durumunda, güncellemenin cihaz üzerinde tek noktada yapılmasına uygun olarak hazırlanmıştır.

Cihaz yönetim çerçevesi, akıllı ulaşım sistemlerinde çalışan uygulamalar için cihaz ağı içinde ve sunucu ile olan haberleşme teknolojisini yalıtarak uygulamaların beraber çalışabildiği, kolay genişleyebilir bir ortam sunmaktadır. Bu doğrultuda, nesnelerin interneti haberleşme teknolojileri incelenerek gecikme süreleri hesaplanmıştır. Bölüm 5.4'te açıklanan servis çağrımı örnek olayı *MQTT* ve *WebSocket* kullanımına ilişkin değerlendirmeleri içermektedir.

Geliştirilen cihaz yöneticisi, ulaşım sistemlerinin en alt katmanındaki donanımlar üzerinde çalışan uygulamaların koordinasyonu ve yönetiminden sorumludur. Cihaz yöneticisi için ortaya koyduğumuz çerçeve, akıllı ulaşım sistemlerinin etkileşimde olduğu ulaştırma, enerji, sağlık ve çevre sektörleriyle beraber çalışarak kendi kendine karar verebilen ulaşım sistemlerinin gerçekleştirilmesine uygun hazırlanmıştır. Bu doğrultuda cihaz yönetim çerçevesinin akıllı çevre ve akıllı şehir uygulamalarına uyarlanabilmesi mümkündür.

Akıllı ulaşım sistemleri ihtiyaçlarına göre özelleşen cihaz yöneticisi çerçevesinin üzerinde gelecekte gerçekleştirilebilecek çalışmalar aşağıda özetlenmiştir:

- Farklı sektörlere ait verinin ulaşım sistemlerine ait uygulamalarda aynı anlamlarda kullanılabilmesi ve cihaz yöneticisi çerçevesinin kullanılabilirliğinin arttırılması için veriyi anlamsal olarak eşleştiren bir ontoloji tanımlanması gereklidir.

- Cihaz ağı içerisindeki uygulamaların kesintisiz olarak çalışabilmesi için çerçevenin mesaj üretiminde kullandığı kural girdilerinin çalışma anında işlenecek şekilde güncellenerek sunucu üzerinden yeni mesajların cihaz ağı içerisinde tetiklenmesi sağlanabilir.
- Akıllı çevre kapsamında kullanım alanı genişleyen elektrikli araçların bakım maliyetlerini kontrol edebilmek ve verimli kullanılabilmesi için veri üzerinden çıkarım yapan araçların entegrasyonu gerekmektedir. Bu doğrultuda cihaz yönetici çerçevesine çıkarım motoru eklenerek ulaşım araçlarının batarya ömürleri ve şarj durumları takip edilebilir.
- Ekonomik sürüş ihtiyaçları kapsamında sürücü performansının takibi ve aracın verimli kullanılması için senaryolar uygulanabilir.
- Cihaz yöneticisi çerçevesinin kritik durumları anlık olarak sürücüye bildirebilmesi için bir arayüz tasarımı gerekmektedir.

KAYNAKLAR DİZİNİ

Abdmeziem, M. R., Tandjaoui, D. and Romdhani, I., 2016, Architecting the internet of things: state of the art. Robots and Sensor Clouds, 55-75.

Ahmadi, H., Arji, G., Shahmoradi, L., Safdari, R., Nilashi, M. and Alizadeh, M., 2019, The application of internet of things in healthcare: a systematic literature review and classification. Universal Access in the Information Society, 18(4), 837-869.

Alam, M., Ferreira, J. and Fonseca, J., 2016, Introduction to intelligent transportation systems. Intelligent transportation systems: Dependable vehicular communications for improved road safety, 1-17.

Ashton, K., 2009, That 'internet of things' thing. RFID journal, 22(7), 97-114.

AUTOSAR, N., 2016, Some/Ip Protocol Specification. AUTOSAR FO Release Version, 1(0).

Atzori, L., Iera, A. and Morabito, G., 2010, The internet of things: A survey, Computer networks, 54(15), 2787-2805.

Azzara, A., Petracca, M. and Pagano, P., 2015, The icsi m2m middleware for iot-based intelligent transportation systems. In 2015 IEEE 18th International Conference on Intelligent Transportation Systems (pp. 155-160). IEEE.

Banavar, G., Chandra, T., Strom, R. and Sturman, D., 1999, A case for message oriented middleware. In Distributed Computing: 13th International Symposium, DISC'99 Bratislava, Slovak Republic September 27–29, 1999 Proceedings 13 (pp. 1-17). Springer Berlin Heidelberg.

Bauer, M., Bui, N., Loof, J. D., Magerkurth, C., Nettsträter, A., Stefa, J. and Walewski, J. W., 2013, IoT reference model. In Enabling Things to Talk (pp. 113-162). Springer, Berlin, Heidelberg.

Bandyopadhyay, S., Sengupta, M., Maiti, S. and Dutta, S., 2011, Role of middleware for internet of things: A study. International Journal of Computer Science and Engineering Survey, 2(3), 94-105.

KAYNAKLAR DİZİNİ (devam)

Bernstein, P. A., 1996, Middleware: a model for distributed system services. Communications of the ACM, 39(2), 86-98.

Bhowmik, R. and Riaz, M. H., 2023, An extended review of the application layer messaging protocol of the internet of things. Bulletin of Electrical Engineering and Informatics, 12(5), 3124-3133.

Chaqfeh, M. A., and Mohamed, N., 2012, Challenges in middleware solutions for the internet of things. In 2012 international conference on collaboration technologies and systems (CTS) (pp. 21-26). IEEE.

Chen, Y., Mao, T. and Yu, B., 2017, A reliable messaging middleware for financial institutions. In Proceedings of the 3rd International Conference on Communication and Information Processing (pp. 108-112).

Cherradi, G., El Bouziri, A. and Boulmakoul, A., 2016, Smart data collection based on iot protocols. JDSI, 16, 2509-2103.

Cheshire, S. and Steinberg, D., 2006, Zero configuration networking: The definitive guide. " O'Reilly Media, Inc.".

Corazza, M. V., Guida, U., Musso, A. and Tozzi, M., 2016, From EBSF to EBSF_2: A compelling agenda for the bus of the future: A decade of research for more attractive and sustainable buses. In 2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC) (pp. 1-6). IEEE.

Corazza, M. V., Magnalardo, S., Musso, A., Petracci, E., Tozzi, M., Vasari, D. and de Verdalle, E., 2018, Testing an innovative predictive management system for bus fleets: outcomes from the Ravenna case study. IET Intelligent Transport Systems, 12(4), 286-293.

Doolan, R., and Muntean, G. M., 2016, EcoTrec—A novel VANET-based approach to reducing vehicle emissions. IEEE Transactions on Intelligent Transportation Systems, 18(3), 608-620.

Dorsemaine, B., Gaulier, J. P., Wary, J. P., Kheir, N. and Urien, P., 2015,

KAYNAKLAR DİZİNİ (devam)

Internet of things: a definition & taxonomy. In 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies (pp. 72-77). IEEE.

Eisenhauer, M., Rosengren, P. and Antolin, P., 2010, Hydra: A development platform for integrating wireless devices and sensors into ambient intelligence systems. In The Internet of Things (pp. 367-373). Springer, New York, NY.

El Zouka, H. A., 2016, A secure interactive architecture for vehicular cloud environment. In 2016 IEEE International Conference on Smart Cloud (SmartCloud) (pp. 254-261). IEEE.

Erdoğmuş, P., 2016, Doğadan esinlenen optimizasyon algoritmaları ve optimizasyon algoritmalarının optimizasyonu. Düzce üniversitesi bilim ve teknoloji dergisi, 4(1), 293-304.

Eugster, P. T., Felber, P. A., Guerraoui, R. and Kermarrec, A. M., 2003, The many faces of publish/subscribe. ACM computing surveys (CSUR), 35(2), 114-131.

Fette, I. and Melnikov, A., 2011, Rfc 6455: The websocket protocol.

Gupta, P., 2021, A survey of application layer protocols for Internet of Things. In 2021 International Conference on Communication information and Computing Technology (ICCICT) (pp. 1-6). IEEE.

Harrington, D., Presuhn, R. and Wijnen, B., 2002, An architecture for describing simple network management protocol (SNMP) management frameworks (No. rfc3411).

Heinzelman, W. B., Murphy, A. L., Carvalho, H. S. and Perillo, M. A., 2004, Middleware to support sensor network applications. IEEE network, 18(1), 6-14.

KAYNAKLAR DİZİNİ (devam)

Hribernik, M. and Kos, A., 2020, Secure WebSocket Based Broker and Architecture for Connecting IoT Devices and Web Based Applications. IPSI Transactions on Advanced Research, 16(1).

Huebscher, M. C. and McCann, J. A., 2004, Adaptive middleware for context-aware applications in smart-homes. In Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing (pp. 111-116).

Ge, F., Xinhua, Z. and Chao, L., 2017, Study of message data subscription based on multi-application big data analysis. Netinfo Security, (11), 44-49.

Goovaerts, T., De Win, B. and Joosen, W., 2008, A comparison of two approaches for achieving flexible and adaptive security middleware. In Proceedings of the 2008 workshop on Middleware security (pp. 19-24).

Gurgen, L., Roncancio, C., Labbé, C., Bottaro, A. and Olive, V., 2008, SStreaMWare: a service oriented middleware for heterogeneous sensor data management. In Proceedings of the 5th international conference on Pervasive services (pp. 121-130).

Giusto, D., 2010, A. lera, g. morabito, l. atzori (eds.) the internet of things. Springer.

Guinard, D., Trifa, V. and Wilde, E., 2010, A resource oriented architecture for the web of things. In 2010 Internet of Things (IOT) (pp. 1-8). IEEE.

ISO/IEC 30141., 2018, Internet of Things (IoT)—Reference Architecture, Standard ISO/IEC 30141:2018, ISO, Mayıs. 2021. [Online]. Available: https://www. iso.org/standard/65695.html

ITU-T. 690, 1997, ITU-T Recommendation X. 690, Information technology-ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), International Telecommunication Union, 2008.

ITU-T Y. 2060, 2012, Overview of the Internet of things. Series Y: Global information infrastructure, internet protocol aspects and next-generation networks-Frameworks and functional architecture models, 2060-201206.

KAYNAKLAR DİZİNİ (devam)

ITxPT, 2017, Onboard Architecture Specification, Information Technology for Public Transport, S02v2.0_2017

Joshi, J., Rajapriya, V., Rahul, S. R., Kumar, P., Polepally, S., Samineni, R. and Tej, D. K., 2017,Performance enhancement and IoT based monitoring for smart home. In 2017 International Conference on Information Networking (ICOIN) (pp. 468-473). IEEE.

Korablev, V., Gugutishvili, D., Lepekhin, A. and Gerrits, B., 2021, Developing a traffic management system architecture model. Transportation research procedia, 54, 918-926.

Kortuem, G., Kawsar, F., Sundramoorthy, V. and Fitton, D., 2009, Smart objects as building blocks for the internet of things. IEEE Internet Computing, 14(1), 44-51.

Kushwah, R., Batra, P. K. and Jain, A., 2020, Internet of things architectural elements, challenges and future directions. In 2020 6th International Conference on Signal Processing and Communication (ICSC) (pp. 1-5). IEEE.

Lee, H. J., 2023, Dynamic Context Awareness of Universal Middleware based for IoT SNMP Service Platform. Tehnički glasnik, 17(2), 185-191.

Lien, Y. C. N. and Wu, W. J., 2010, A lexical database filter for efficient semantic publish/subscribe message oriented middleware. In 2010 Second International Conference on Computer Engineering and Applications (Vol. 2, pp. 154-157). IEEE.

Liu, X., Zhang, T., Hu, N., Zhang, P. and Zhang, Y., 2020, The method of Internet of Things access and network communication based on MQTT. Computer Communications, 153, 169-176.

Lezoche, M., Hernandez, J. E., Díaz, M. D. M. E. A., Panetto, H. and Kacprzyk, J., 2020, Agri-food 4.0: A survey of the supply chains and technologies for the future agriculture. Computers in industry, 117, 103187.

KAYNAKLAR DİZİNİ (devam)

Locke, D., 2010, Mq telemetry transport (mqtt) v3. 1 protocol specification. IBM developerWorks Technical Library.

Lombardi, M., Pascale, F. and Santaniello, D., 2021, Internet of things: A general overview between architectures, protocols and applications. Information, 12(2), 87.

Manandhar, S., 2017, MQTT based communication in IoT (Master's thesis).

Mavromatis, A., Colman-Meixner, C., Silva, A. P., Vasilakos, X., Nejabati, R. and Simeonidou, D., 2019, A software-defined IoT device management framework for edge and cloud computing. IEEE Internet of Things Journal, 7(3), 1718-1735.

Meng, Z., Wu, Z., Muvianto, C., and Gray, J., 2016, A data-oriented M2M messaging mechanism for industrial IoT applications. IEEE Internet of Things Journal, 4(1), 236-246.

Minerva, R., Biru, A. and Rotondi, D., 2015, Towards a definition of the Internet of Things (IoT). IEEE Internet Initiative, 1(1), 1-86.

Mocrii, D., Chen, Y. and Musilek, P., 2018, IoT-based smart homes: A review of system architecture, software, communications, privacy and security. Internet of Things, 1, 81-98.

Oliveira, G. M., Costa, D. C., Cavalcanti, R. J., Oliveira, J. P., Silva, D. R., Nogueira, M. B. and Rodrigues, M. C., 2018, Comparison between MQTT and WebSocket protocols for Iot applications using ESP8266. In 2018 Workshop on Metrology for Industry 4.0 and IoT (pp. 236-241). IEEE.

Özdoğan, O., 2017, Endüstri 4.0: dördüncü sanayi devrimi ve endüstriyel dönüşümün anahtarları. Pusula.

Pandey, S., Choi, M. J., Kim, M. S. and Hong, J. W., 2011, Towards management of machine to machine networks. In 2011 13th Asia-Pacific Network Operations and Management Symposium (pp. 1-7). IEEE.

Postel, J., 1981, Transmission control protocol (No. rfc793).

KAYNAKLAR DİZİNİ (devam)

Puerta, J., Brazález, A., Suescun, A., Iparraguirre, O. and Atutxa, U., 2018, Standardizing IT Systems on Public Transport: An Eco-Driving Assistance System Case Study. In International Workshop on Communication Technologies for Vehicles (pp. 149-158). Springer, Cham.

Ramachandran, A., 2022, Design of an Edge to Cloud IIoT Middleware Architecture. North Carolina State University.

Ray, P. P., Dash, D., and De, D., 2019, Edge computing for Internet of Things: A survey, e-healthcare case study and future direction. Journal of Network and Computer Applications, 140, 1-22.

Razzaque, M. A., Milojevic-Jevric, M., Palade, A. and Clarke, S., 2015, Middleware for internet of things: a survey. IEEE Internet of things journal, 3(1), 70-95.

Radhika, J. and Malarvizhi, S., 2012, Middleware approaches for wireless sensor networks: An overview. International Journal of Computer Science Issues (IJCSI), 9(6), 224.

Raychoudhury, V., Cao, J., Kumar, M. and Zhang, D., 2013, Middleware for pervasive computing: A survey. Pervasive and Mobile Computing, 9(2), 177-200.

RFC2616., The Internet Society Hypertext Transfer Protocol (HTTP), [online] Available: https://tools.ietf.org/html/rfc2616.

Rouvoy, R., Barone, P., Ding, Y., Eliassen, F., Hallsteinsen, S., Lorenzo, J. and Scholz, U., 2009, Music: Middleware support for self-adaptation in ubiquitous and service-oriented environments. Software engineering for self-adaptive systems, 164-182.

Sethi, P. and Sarangi, S. R., 2017, Internet of things: architectures, protocols, and applications. Journal of Electrical and Computer Engineering.

Sharma, S. and Awasthi, S. K., 2022, Introduction to intelligent transportation system: overview, classification based on physical architecture, and

KAYNAKLAR DİZİNİ (devam)

challenges. International Journal of Sensor Networks, 38(4), 215-240.

Shelby, Z., Hartke, K. and Bormann, C., 2014, The Constrained Application Protocol (CoAP). Technical Report, https://www.rfc-editor.org/info/rfc7252

Sheng, Z., Mahapatra, C., Zhu, C. and Leung, V. C., 2015, Recent advances in industrial wireless sensor networks toward efficient management in IoT. IEEE access, 3, 622-637.

Silva, D., Carvalho, L. I., Soares, J. and Sofia, R. C., 2021, A performance analysis of internet of things networking protocols: Evaluating MQTT, CoAP, OPC UA. Applied Sciences, 11(11), 4879.

Stojkoska, B. L. R. and Trivodaliev, K. V., 2017, A review of Internet of Things for smart home: Challenges and solutions. Journal of cleaner production, 140, 1454-1464.

Thangavel, D., Ma, X., Valera, A., Tan, H. X. and Tan, C. K. Y., 2014, Performance evaluation of MQTT and CoAP via a common middleware. In 2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP) (pp. 1-6). IEEE.

Thoma, M., Braun, T., Magerkurth, C. and Antonescu, A. F., 2014, Managing things and services with semantics: A survey. In 2014 IEEE Network Operations and Management Symposium (NOMS) (pp. 1-5). IEEE.

Tozzi, M., Bousse, Y., Karlsson, M. and Guida, U., 2016, A European initiative for more efficient and attractive bus systems: the EBSF_2 Project. Transportation Research Procedia, 14, 2640-2648.

Ulaştırma ve Altyapı Bakanlığı, "Akıllı Ulaşım Sistemleri (AUS)", https://hgm.uab.gov.tr/akilli-ulasim-sistemleri-aus (Erişim tarihi: 23 Eylül 2023)

Yahya, W., Basuki, A., Sakti, P. E. and Fernanda, F. F., 2017, Lightweight monitoring system for iot devices. In 2017 11th international conference on telecommunication systems services and applications (TSSA) (pp. 1-4).

KAYNAKLAR DİZİNİ (devam)

IEEE.

Yongguo, J., Qiang, L., Changshuai, Q., Jian, S. and Qianqian, L., 2019, Message-oriented middleware: A review. In 2019 5th International Conference on Big Data Computing and Communications (BIGCOM) (pp. 88-97). IEEE.

Van den Abeele, F., Hoebeke, J., Moerman, I. and Demeester, P., 2014, Fine-grained management of CoAP interactions with constrained IoT devices. In 2014 IEEE Network Operations and Management Symposium (NOMS) (pp. 1-5). IEEE.

Vermesan, O. and Friess, P. (Eds.)., 2013, Internet of things: converging technologies for smart environments and integrated ecosystems. River publishers.

Vermesan, O., Friess, P., Guillemin, P., Gusmeroli, S., Sundmaeker, H., Bassi, A. and Doody, P., 2011, Internet of things strategic research roadmap. Internet of things-global technological and societal trends, 1(2011), 9-52.

Wu, M., Lu, T. J., Ling, F. Y., Sun, J. and Du, H. Y., 2010, Research on the architecture of Internet of Things. In 2010 3rd international conference on advanced computer theory and engineering (ICACTE) (Vol. 5, pp. V5-484). IEEE.

TEŞEKKÜR

Tez çalışmam boyunca değerli görüş ve önerileriyle doğru yönde ilerlememi sağlayan, her konuda yardım ve ilgisini esirgemeyen değerli hocam ve tez danışmanım Prof. Dr. N. Yasemin TOPALOĞLU hocama teşekkürlerimi sunarım. Tez izleme komitesi toplantılarındaki kıymetli görüş ve önerilerle çalışmama katkılar sağlayan Prof. Dr. Oğuz DİKENELLİ ve Doç. Dr. Tuğkan TUĞLULAR hocalarıma teşekkür ederim. Bu çalışmalarım süresince benden desteğini esirgemeyen Kentkart ailesindeki çalışma arkadaşlarıma ve Kentkart Ege Elektronik San. Ve Tic. A.Ş Arge Merkezine teşekkürü bir borç bilirim.

Her zaman olduğu gibi tüm öğrenim hayatım boyunca beni destekleyen sevgili annem Müveddet ÖZ'e ve babam Ethem ÖZ'e teşekkürü bir borç bilirim.

Akademik yaşamımda bana sınırsız destek olan sevgili eşim Esra'ya ve biricik oğlum Batu'ya sonsuz teşekkür ederim.

07 / 03 / 2024

Can ÖZ

ÖZGEÇMİŞ

Kişisel Bilgiler

Ad : : Can

Soyad : Öz

Eğitim

Doktora : Ege Üniversitesi - Bilgisayar Mühendisliği [2012 - 2024]

Y.Lisans : Ege Üniversitesi - Bilgisayar Mühendisliği [2008 - 2011]

Lisans : Dokuz Eylül Üniversitesi - Bilgisayar Mühendisliği [2003- 2008]

Lise : Mehmet Emin Resülzade Anadolu Lisesi [1996 – 2003]

İş Deneyimi

Kentkart Ege Elektronik A.Ş. Yazılım Geliştirme Takım Lideri [2014 - ...]

Kentkart Ege Elektronik A.Ş. Yazılım Geliştirme Mühendisi [2009 - 2014]

Grundig Elektronik A.Ş. Yazılım Tasarım Mühendisi [2008 - 2009]

Akademik Yayınlar

2023 IEEE International Smart Cities Conference (ISC2)

Message Based Terminal Manager for Public Transportation Systems (https://ieeexplore.ieee.org/abstract/document/10293500) **Bucharest, Romania**

2022 Bilgisayar Bilimleri ve Mühendisliği Dergisi

Nesnelerin İnterneti Yaklaşımıyla Konvoy Araçların Yönetimi, (https://dergipark.org.tr/en/download/article-file/2164030) **Cilt-15 Sayı-1**

2021 Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı (UBMK)

Ulaşım Sistemleri için Mesaj Tabanlı Cihaz İletişim Çerçevesi, (https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9558971), **Ankara**

2019 Akıllı Sistemlerde Yenilikler ve Uygulamaları Konferansı (ASYU)

Ulaşım Sistemleri için Mesaj Tabanlı Cihaz Yönetim Mimarisi, (https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8946357), **İzmir**

2016 10. Ulusal Yazılım Mühendisliği Sempozyumu (UYMS)

Gelecek Nesil Gömülü Sistem Uygulamaları için Kullanıcı Etkileşimi Yaklaşımı Önerisi, (http://ceur-ws.org/Vol-1721/UYMS16_paper_33.pdf), **Çanakkale**