Nesnelerin İnterneti için Gerçek Zamanlı Tasarsız Veri Toplama Platformu

Araştırma Makalesi/Research Article

Vahid Khalilpour AKRAM, Orhan DAĞDEVİREN

Uluslararası Bilgisayar Enstitüsü, Ege Üniversitesi, İzmir, Türkiye
vahid.akram@ege.edu.tr, orhan.dagdeviren@ege.edu.tr
(Geliş/Received:30.05.2020; Kabul/Accepted:21.10.2020)
DOI: 10.17671/gazibtd.745598

Özet— Günümüzde, fiziksel dünyadan bilgi edinebilen ve haberleşme yeteneğine sahip birçok akıllı nesne birbirine bağlanarak küresel bir algılayıcı ağ altyapısı veya Nesnelerin İnternetini oluşturmaktadır. Nesnelerin İnterneti için üretilen cihazların hızla artması ve her cihazın geliştirme ortamı ve ara yüzünün farklı olması cihaza özel uygulamaların üretimine sebep olmuştur. Bu çalışmada Nesnelerin İnterneti için bir gerçek zamanlı tasarsız veri toplama, raporlama ve analiz platformu önerilmiştir. Geliştirilen platform, sensörlerden veri toplayan gömülü yazılımı, yerel veri toplama uygulaması, bulut üzerinde çalışan NodeJS tabanlı sunucu ve web tabanlı istemci yazılımı olmak üzere, dört ana bileşenden oluşmaktadır. Bileşenlerin tamamen bağımsız olması, tasarlanan platformun birçok uygulamada kolaylıkla kullanılmasını sağlamaktadır. Önerilen platformda sensörler arasında WiFi veya kablolu ağ gibi bir merkezi iletişim altyapısına gerek olmadan, sensörler arası bir kapsayan ağaç oluşturulur ve toplanan veriler ağaç üzerinden işleme merkezine ulaştırılır. Eklenen yeni sensörler, ağa bağlı mevcut düğümlerden birisini hedef düğüm olarak seçip topladığı verileri o düğüme gönderir. Her düğüm diğer düğümlerden topladığı verileri kendi hedef düğümüne göndererek verileri yerel işleme merkezine ulaştırır. Böylece ağın kapsama alanı kolay ve hızlı bir şekilde genişletilebilir. Veriler yerel işleme merkezinden bulut üzerinde çalışan bir sunucuya aktarılır. Son kullanıcılar web üzerinden sunucuya bağlanarak sensörlerin anlık verilerini ve bu verilerden üretilen analitik raporları görebilirler. 96 saat boyunca yapılan deneysel çalışmalarda, geliştirilen platform istikrarlı bir şekilde çalışarak toplanan verilerden anlık raporlar üretmiştir.

Anahtar Kelimeler— Nesnelerin İnterneti, Gerçek Zamanlı Veri Toplama, Tasarsız Ağlar, Sensör, Kapsayan Ağaç.

Real Time Ad-Hoc Data Collection Platform for Internet of Things

Abstract— Recently, many smart objects that can obtain information from the physical world and have the communication ability are connected to each other and form a global sensor network infrastructure or Internet of Things. The rapid increase of the developed devices for the Internet of Things and the specific development environment of each device have led to the implementation of device-specific applications. In this study, a real-time ad-hoc data collection, reporting and analysis platform is proposed for the Internet of Things. The developed platform consists of four main components, including embedded software that collects data from sensors, local data collection application, NodeJS-based server running on the cloud, and web-based client software. Each component is completely independent from the others which allows the platform to be easily adapted to many applications. With the proposed platform, a spanning tree is created between the sensors without the need for a central communication infrastructure such as WiFi or wired network, and the collected data is transmitted to the processing center over the tree. The new sensors select one of the existing nodes in the network as the target and send their collected data to that node. Each node sends the received data from the other nodes to its destination node which delivers the data to a local processing center. Thus, the coverage area of the network can be easily and quickly expanded. The data is transferred from the local processing center to a server running on the cloud. End users can connect to the server through the web and see the real time data of the sensors and analytical reports produced from this data. In the experimental studies carried out for 96 hours, the developed platform worked stably and produced instant reports from the collected data.

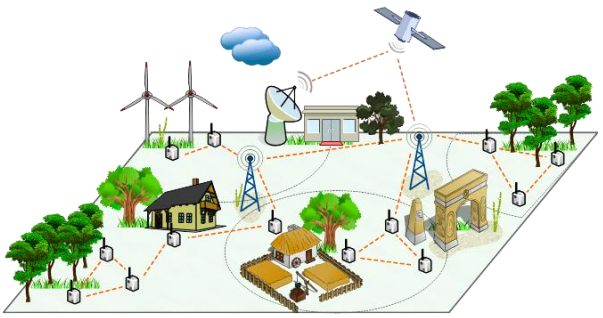
Keywords— Internet of Things, Real Time Data Collection, Ad-Hoc Networks, Sensor, Spanning Tree.

1. GİRİŞ (INTRODUCTION)

Bilgi teknolojileri alanında hızla gelişen teknolojilerden birisi olan bulut bilişim (Cloud Computing) teknolojisi, güvenli ve konumdan bağımsız depolama, hesaplama ve çeşitli uygulama servislerini sağlamaktadır. Diğer yandan, farklı sensörlere sahip, düşük maliyetli ve verimli enerji tüketen yeni nesil donanımların üretilmesiyle beraber Nesnelerin İnterneti (Internet of Things) yaygın bir şekilde algılama, izleme, kontrol ve akıllı yönetim sistemlerinde kullanılmaya başlamıştır. Bu iki teknolojiden yararlanan yeni nesil uygulamalar, farklı alanlarda önemli servisler ve hizmetleri sağlamaktalardır. Örneğin, sağlık hizmetlerinde uzaktan hasta bakım ve el hijyeni izleme sistemleri [1, 2], toplu ulaşımda otomatik araç yönetimi [3, 4] ve akıllı trafik yönetimi [5, 6], hava izleme ve erken uyarı sistemleri [7, 8], ve akıllı ev sistemleri [9, 10] nesnelerin interneti ve bulut bilişim teknolojilerinden yararlanmaya başlamıştır. Nesnelerin internetinde, sensörlerden gerçek zamanlı ve doğru veri toplama, önemli işlevlerden biridir. Genel olarak, sensör tabanlı sistemlerde, çevre, yapı veya bir varlığın çeşitli özellikleri sensörler tarafından algılanıp, WiFi, kablolu bir iletişim altyapısı, veya çok-atlamalı (multi-hop) bağlantılar üzerinden bir baz istasyonuna ve oradan bir veri işleme merkezine gönderilir. Veriler işleme merkezine ulaştıktan sonra, algılanan bilgilere dayanarak akıllı kararlar verilebilir veya verilen hizmetlerin kalitesi yükseltilebilir. Örneğin, bir tarım veya sera alanında havanın sıcaklığı, ortamın ışık yoğunluğu, rüzgarın şiddeti ve toprağın nemi, alanda dağıtılan sensörler tarafından ölçülüp, kontrol altında tutularak, bitkilerin verimliliği artırılabilir. Başka bir örnek olarak, ormanlarda yangını önlemek için ormanın çeşitli konumlarına hava sıcaklığını ölçen sensörler dağıtılabilir. Böylece sıcaklığı belli bir seviyenin üstüne çıkan bölgelerde gereken müdahaleler daha erken yapılabilir.

Genel olarak, nesnelerin internetinde, radyo mesajları üzerinden haberleşebilen, ortamda çeşitli özellikleri veya eylemleri algılayabilen, sınırlı bellek ve düşük işleme gücünü sahip ve çok az enerji tüketen donanımlar kullanılır. Bu cihazlar, topladıkları verileri WiFi, kablolu ağ veya GSM gibi farklı iletişim kanallarından bir merkezi işleme istasyonuna ulaştırırlar. Böyle altyapıların bulunmadığı, orman, deniz ve yeraltı gibi ortamlarda, veriler diğer düğümler aracıyla çok-atlamalı bağlantılar üzerinden işleme istasyonuna gönderilir. Çalışmamızda merkezi iletişim altyapısı olmayan ortamlar için tasarsız (ad-hoc) çok-atlamalı bağlantılar üzerinden veri toplayabilen bir platform önermekteyiz.

Şekil 1 nesnelerin internetini kullanan örnek bir tasarsız çevre izleme sistemini göstermektedir. Birbirinin radyo menzilinde bulunan cihazların arasında bir bağlantının olduğu varsayılabilir. Dolayısıyla, düğümlerin arasında veri iletimi için birden fazla patika bulanabilir. Toplanan veriler çok-atlamalı bağlantılar aracıyla veri işleme merkezine gönderilir. Birçok uygulamada sensör cihazların enerji kaynağı sınırlı olduğundan dolayı, verilerin enerji tasarruflu ve aynı zamanda en kısa patikadan iletilmesi gerekmektedir.



Şekil 1. Örnek bir çevre izleme sistemi
(An environment monitoring system)

Günümüzde nesnelerin internetini destekleyen, farklı donanımlar ve sensörler bulunmaktadır. Bu cihazların desteklediği işletim sistemi ve programlama dilleri farklı olduğundan dolayı, farklı donanımlar için farklı yazılımlar ve platformlar geliştirilmiştir. Genel olarak nesnelerin internetinden bir veri toplama platformu aşağıdaki ana işlevleri desteklemelidir:

- Tüm sensörlerden algılanan veriler bir işleme merkezinde toplanmalı ve elde edilen verilerden anlamlı analitik raporlar üretilmelidir.
- İletişim altyapısı veya çok atlamalı bağlantılar üzerinden enerji tasarruflu veri iletilmesi sağlanmalıdır.
- Düğümlerin uzaktan yönetimi sağlanmalıdır.
- Yeni düğümlerin eklenmesi desteklenmelidir.
- Mevcut düğümler veya bağlantılarda olası arızalar tespit edilmelidir.

Bu çalışmada nesnelerin interneti için bir veri toplama platformu geliştirilmiştir. Geliştirilen sistemde Telsiz Duyarga Ağlarında (TDAlarında) [20] kullanılan sensörlerden elde edilen veriler çok-atlamalı bağlantılar üzerinden bir baz istasyonuna gönderilir ve internet üzerinden bir sunucuya aktarılır. Kullanıcılar bulut üzerinde çalışan web sunucusuna bağlanarak gönderilen anlık veriler, çeşitli raporlar ve analitik grafikleri görebilirler.

İlerleyen bölümler şu şekilde organize edilmiştir. Bölüm 2'de ilgili çalışmalar ve benzer platformların özellikleri tartışılmıştır. Bölüm 3'te problemin tanımı ve ağ modeli verilmiştir. Bölüm 4 geliştirilen sistemin bileşenleri ve her bileşenin detaylarını kapsamaktadır. Bölüm 5'te deneysel çalışmada elde edilen sonuçlar anlatılmıştır. Bölüm 6'da sonuç ve gelecek çalışmalar verilmiştir.

2. İLGİLİ ÇALIŞMALAR (RELATED WORKS)

Nesnelerin İnterneti için tasarlanan platformların genel amacı, fiziksel cihazlar ile dijital dünya arasında bir bağlantı kurarak İnternet üzerinden çeşitli nesnelere erişim sağlamaktır. Genel olarak bu platformlar, sensörler tarafından algılanan verileri toplayıp depolamakla beraber anlaşılabilir bir şekilde kullanıcılara sunarlar. Ayrıca, bu platformlar kullanıcılar tarafından verilen talimatlara,

nesnelerin yürütebileceği komutları çevirip İnternet üzerinden cihazlara aktarırlar. Son yıllarda, Nesnelerin İnternetine artan ilgiden dolayı bu teknolojiyi destekleyen platformların sayısı hızla artmaktadır. Genel olarak, Nesnelerin İnterneti için tasarlanan platformları, alana özgü ve çok amaçlı gruplara ayırabiliriz. Alana özgü platformlar özel bir uygulama için tasarlanırlar ve o alanın ihtiyaçları doğrultusunda çeşitli servisler sunarlar. Örneğin, tarım ve hayvancılık için tasarlanan FarmBeats platformu çeşitli sensörler, kameralar ve dronlar aracıyla, ortamdan anlık veri toplayarak, hava sıcaklığı, rüzgar şiddeti, toprağın nemi ve PH değeri, ve hayvanların anlık konumu gibi verilerden yararlı haritalar ve raporlar üretebilir [17]. Başka bir örnek olarak, akıllı binalarda tüketilen enerjinin yönetimi için tasarlanan IoTEP platformu, binanın çeşitli konumlarına yerleştirilen sensörler aracıyla her odanın hava sıcaklığı ve odanın tükettiği anlık enerji miktarını ölçerek, odalarda enerjinin verimli tüketildiği ile ilgili çeşitli raporlar ve grafikler üretebilir [21].

Akıllı sera veya örtü altı tarım ortamlarında hassas bitkileri yetiştirmek için geliştirilen platform, sıcaklık, nem, su ve toprağın kimyasal değerleri, ışık seviyesi ve hava sirkülasyonu gibi tüm ölçümleri çok hassas sensörler aracıyla kontrol edip, sulama sistemi, pervane motorları, kalorifer veya uyarı sistemlerine gerekli komutları gönderebilir [22]. Kentsel hava kirliliği ve havadaki partikül maddeleri ölçmek için WiFi, 4G, Bluetooth ve LoRa gibi kablosuz iletişim teknolojileri destekleyen AirQ isimli bir platform tasarlanmıştır [23]. Bu platform taşınabilir ve düşük maliyetli cihazlar aracıyla hava kalitesini izleyerek, gerçek zamanlı ve konuma özgü hava kalite raporlarını web ve akıllı telefonlar için üretebilir. [24]'de sunulan IoMT isimli platform, hastane ve ev ortamında hastaların hayati değerlerini ölçüp sağlık çalışanlarına anlık raporlar üretebilir. Bu platformda, hastaların kalp atışı, vücut sıcaklığı ve hareket durumları gibi hayati değerler giyilebilir sensörler tarafından algılanır ve Node MCU veya Raspberry Pi cihazları üzerinde bulunan WiFi bağlantısı sayesinde bir bulut sistemine aktarılır. Ayrıca, hastanın bulunduğu ortama ışık, sıcaklık ve kızılötesi sensörler yerleştirilerek, ortamın bilgileri anlık olarak bulut üzerine aktarılır. Doktorlar ve hastane personeli bu bilgilere akıllı telefon veya web üzerinden erişerek hastanın durumunu sürekli kontrol altında tutabilirler.

Genel olarak, özel bir uygulama alanına tasarlanan platformlar, hedeflenen spesifik ortam veya uygulamalar için önemli avantajlar sağlayabilirler ancak bu platformların farklı uygulamalarda kullanılması çok zor ve hatta imkansız olabilir. Ayrıca, genel olarak bu platformlar özel nesneler ve belirli iletişim altyapıları için tasarlanırlar ve farklı amaçlar için üretilen sensörleri, aygıtları ve farklı iletişim kanallarını kullanamazlar.

Son yıllarda, Nesnelerin İnterneti ve bulut sistemlerini, uygulama alanından bağımsız olarak destekleyen platformların sayısı hızla artmaktadır. Örneğin KAA isimli bir platform, Nesnelerin İnterneti için tasarlanan açık kaynaklı ve çok amaçlı bir platformdur [25]. Bu platform cihazlara uzaktan erişim, veri analizi, depolama, görselleştirme ve bulut hizmetlerini sunmaktadır. Cihazların konfigürasyonu, cihazlar arası iletişimin kontrolü ve yazılımın uzaktan güncellenmesi KAA'nın sağladığı en önemli servislerdir. Veri depolama için, NoSQL tabanlı Cassandra Hadoop veya MongoDB veritabanı kullanılabilir. Bu platformun sunduğu ara katman gömülü SDK yazılımı, geliştiricinin cihazlarına yüklenmelidir. Dolayısıyla, bu platformun kullanıcıları sadece SDK tarafından desteklenen sınırlı sayıda cihazları kullanabilirler. SDK programı minimum 10 kb RAM ve 40 kb ROM bellek kullandığı için bu platformu kullanan geliştiricilerin kullanabileceği bellek miktarı sınırlı olabilir.

Başka bir örnek olarak, IBM IoT ve bulut platformu, nesneler ile uygulamalar arasında bağlantı kurarak, karmaşık endüstri çözümlerine kolaylık sağlayabilir [26]. Bu platformun omurgası IDentity as a Service (IDaaS) mimarisidir. Bu platform, ARM, Arduino ve Intel Galileo gibi çeşitli aygıtları MQTT mesajlaşma protokolü aracıyla IBM bulutuna bağlayabilir. RESTful ve gerçek zamanlı API'ler, aygıtlardan gelen verileri IBM bulut sistemine aktarırlar ve çeşitli uygulamalar bulut üzerinden bu verilere ulaşabilirler. IBM bulut sistemi, NoSQL, Dash ve TimeSeries veri tabanlarını desteklemektedir. Ayrıca aygıtların konfigürasyonu, yönetimi, yazılım güncellemesi ve durumlarının izlenmesi gibi çeşitli işlemler bu platform üzerinden yapılabilir. IBM platformu profesyonel ve büyük çaplı uygulamalar için geliştirilmiştir; bu yüzden bu platformda küçük ve orta çaplı uygulamaların geliştirilmesi zor olabilir.

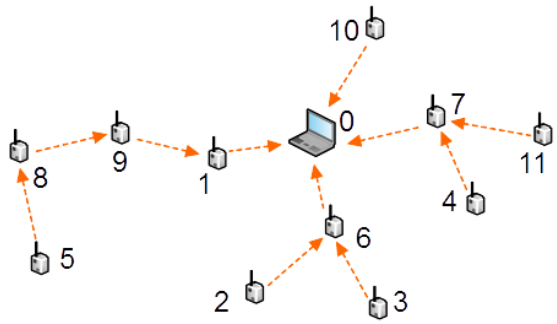
Oracle tarafından geliştirilen, Oracle IoT platformu, sensörler, ağ geçitleri ve uygulamalar arasında bağlantı kurabilir ve cihazlardan gönderilen verileri bulut üzerinde depolayabilir [27]. Bu platform toplanan verilerin ticari değerini analiz edebilir ve filtreleme, korelasyon ve anormallik algılama algoritmaları ile büyük hacimli verilerin sorgulamasını, analizini ve görselleştirilmesini kolaylaştırır. Oracle IoT platformu, cihazlar üzerinde Java için en az 11 MB bellek alanına ihtiyaç duymaktadır. Bu nedenle bu platformun desteklediği cihazların sayısı ve geliştiricilerin cihazlar üzerinde kullanabileceği bellek miktarı sınırlıdır.

Nesnelerin İnterneti için önde gelen platformlardan bir diğeri Thing Speak platformudur [28]. Thing Speak bulut teknolojisine dayalı bir platform olarak, gerçek zamanlı veri toplama, analiz ve raporlama işlemlerini desteklemektedir. Ayrıca, Thing Speak platformu ioBridge, Arduino, Twilio, Twitter, ThingHTTP, ve Matlab gibi üçüncü taraf uygulamalarla entegrasyon olanaklarını sağlamaktadır. Sensörlerden toplanan veriler, cihazın durumu ve cihazın konum bilgileri ile buluta aktarılır ve bulut üzerinden çeşitli uygulamalara sunulur. Bu platformun en önemli dezavantajı, aynı zamanda platforma bağlanabilen cihazların sınırlı olmasıdır.

Çalışmamızda önerilen platform, uygulama, iletişim altyapısı ve cihazlardan bağımsız olarak, farklı sensörler ve cihazları destekleyip, esnek bir uygulama, görselleştirme ve raporlama sistemini sağlamaktadır. Ayrıca, önerilen platformda, çok atlamalı bağlantıları ve tasarsız ağları desteklendiği için bu platformda tüm düğümlerin WiFi veya GSM üzerinden İnternete bağlanmalarına gerek kalmamaktadır. Böylece, büyük bir alanda tek bir düğümün İnternete bağlı olması, tüm diğer sensörler ve düğümlerin uzaktan erişimine yeterli olabilir.

3. PROBLEM TANITIMI VE AĞ MODELİ (PROBLEM FORMULATION AND NETWORK MODEL)

Gerçek uygulama senaryolarında, algılama alanındaki kritik bölgelerin anlık ve doğru izlenmesi büyük önem taşımaktadır. Genellikle, tüm uygulamalarda izleme alanı farklı bölgelere ayrılır ve her bölgeye bağımsız sensörler konumlandırılır. Birçok senaryoda (örneğin çiftlik veya orman alanlarında) algılama alnında yerel ağ veya WiFi gibi merkezi iletişim altyapıları bulunmamaktadır. Böyle alanların çok büyük olabileceği ve düğümlerin radyo haberleşme menzillerinin sınırlı olduğundan dolayı, baz istasyonundan uzak düğümler, aradaki diğer düğümleri kullanarak, çok-atlamalı bağlantılar üzerinden verilerini iletirler. Bu durumda düğümlerin arasında bir yönlendirme mekanizması bulunmalıdır. Algılanan tüm verilerin baz istasyonuna iletilmesi gerektiğinden dolayı, düğümler arasında kökü baz istasyonu olan bir kapsayan ağaç (spanning tree) kullanılabilir. Böylece ağaçta bulunan her düğüm kendi ürettiği veya çocuklarından gelen verileri ebeveyn düğümüne ileterek, tüm verileri baz istasyonuna ulaştırabilir. Şekil 2, örnek bir ağda düğümler arasında oluşan kapsayan ağacı göstermektedir. Örneğin 5 nolu düğümün algıladığı verilere 8, 9, ve 1 nolu düğümler üzerinden baz istasyonuna ulaşabilir.



Şekil 2: Kapsayan ağaç üzerinden veri iletimi
(Data forwarding over spanning tree)

Bu çalışmada, düğümlerin arasında merkezi bir iletişim altyapısının olmadığı varsayılmaktadır. Geliştirilen platformda düğümler arası bir kapsayan ağaç oluşturulur ve her düğüm verilerini çok-atlamalı bağlantılar üzerinden baz istasyonuna ulaştırır. Geliştirilen sistemde, paketlerin yönlendirilmesi bir kapsayan ağaç aracıyla yapılmaktadır. Her düğüm çalışmaya başladıktan sonra ağda bulunan düğümlerden birisini ebeveyn düğümü olarak seçip, verilerini bu düğüme iletir. Her düğüm diğer düğümlerden gelen verileri ebeveyn düğümüne ileterek verilerin baz istasyonuna ulaşmasını sağlar. Düğümler, ebeveyn düğümlerinin kimliğini, hedef düğüm olarak mesaja ekleyip, tüm mesajları toplu iletim (broadcast) olarak iletir. Böylece düğümler aynı zamanda komşularına kendilerinin hayatta olduklarını da bildirirler. Dolayısıyla bir ebeveyn düğüm bozulduğu zaman, onun çocukları başka ebeveyn düğüm bulmaya başlayabilirler.

Önerilen platformda, ağa yeni düğümlerin eklenmesi için o düğümleri ağa bağlı herhangi bir düğümün radyo menziline yerleştirmek yeterlidir. Bu nedenle, yeni düğümler eklenerek, sistem hızlı bir şekilde uygulama alanında genişletilebilir. Algılanan verilerin yanısıra, RSSI (Received Signal Strength Indicator) ve düğümün pilinden gelen voltaj değerleri de baz istasyonuna gönderilerek, kullanıcıya düğümlerin durumuyla ilgili anlık raporlar sunulabilmektedir. Kullanıcılar geliştirilen ara yüzleri kullanarak, herhangi bir düğümün detaylı bilgilerini ve durumunu görebilirler.

Bir düğüm tarafından belli bir süre mesaj gelmediği zaman, düğüm sistem tarafından bozuk olarak rapor edilir ve böylece kullanıcılar bozulan düğümler ve bağlantılar hakkında anlık bilgi alabilirler. Geliştirilen sistemde aşağıdaki varsayımlar bulunmaktadır:

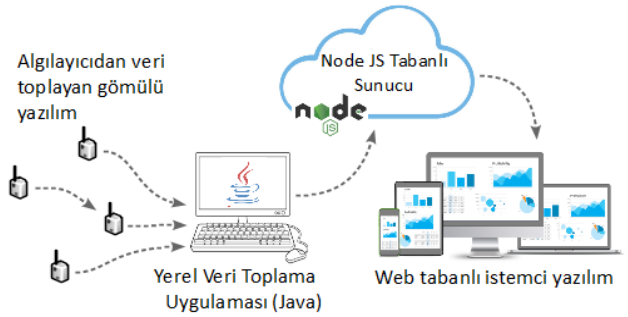
1- Her düğüm eşsiz bir kimlik numarası veya adrese sahiptir.
2- Tüm düğümlerde radyo haberleşme cihazı bulunmaktadır ve düğümler arası iletişim çift taraflıdır.
3- Düğümlerin üzerinde birden fazla sensör bulunabilir.
4- Düğümlerin enerji kaynağı ve bellek kapasiteleri sınırlıdır ve ağda en az bir geçit bulunmaktadır.
5- Sensörler tarafından gönderilen veriler sınırsız kullanıcı tarafından izlenebilir.

Düğümlerin gönderdiği veriler çeşitli analitik raporlar şeklinde kullanıcıya sunulabilir ve bu verilere aynı zamanda birden fazla kullanıcı İnternet üzerinden erişebilir. Ayrıca, kullanıcılar farklı bölgelerde, farklı amaçlar veya uygulamalar için sensörlerin verilerini aynı platformdan görebilirler. Bu varsayımlar altında tasarlanan platformun detayları ilerleyen bölümde verilmiştir.

4. TASARLANAN VERİ TOPLAMA PLATFORMU (DEVELOPED DATA GATHERING PLATFORM)

Tasarlanan veri toplama platformu, donanımlar üzerinde çalışan gömülü yazılım, yerel veri toplama uygulaması, NodeJS tabanlı sunucu ve web tabanlı bir istemci uygulaması olmak üzere 4 ana bileşenden oluşmaktadır. Gömülü yazılım, sensörlerin üzerinde çalışarak ortamın verilerini algılayıp, tek veya çok atlamalı bağlantılar üzerinden bir yerel veri toplama merkezine aktarır. Yerel veri toplama merkezi ortamda bulunan sensörlerden gelen verileri toplayıp gereken biçimlendirme ve birleştirme işlemlerini yaptıktan sonra İnternet veya yerel ağ bağlantıları üzerinden REST API aracıyla sunucuya aktarır. Sunucu gelen verileri web üzerinden bağlanan kullanıcılara aktarır. İstemci tarafında çalışan web

uygulaması, JavaScript kodları aracıyla gelen verileri çeşitli raporlar ve grafikler şeklinde kullanıcıya sunar. Şekil 3 tasarlanan sistemin genel mimarisini göstermektedir.



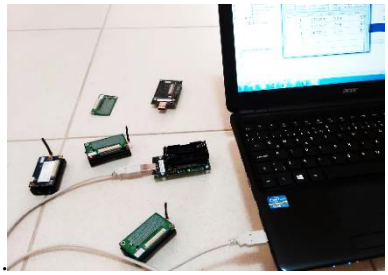
Şekil 3: Geliştirilen platformun genel mimarisi
(General architecture of developed platform)

Şekil 3'te gösterilen mimarinin en önemli avantajı bileşenlerin birbirinden tamamen bağımsız olmasıdır. Başka bir deyişle, gömülü yazılım, yerel veri toplama, sunucu ve web tabanlı istemci uygulamaları tamamen bağımsız bir şekilde çalışarak, platformun farklı alanlara kolaylıkla uygulanabileceğini sağlamaktalardır. Örneğin TDA yerine, Arduino, Node MCU veya cep telefonların sensörlerinden veri toplamak için gömülü yazılım dışında diğer bileşenler hemen hemen değişmeyecektir.

4.1. Gömülü Yazılım (Embeded Software)

Bu çalışmada ortamdan veri toplamak için IRIS [29], MDA100 [30] ve MIB520 [31] donanımları kullanılmıştır. Her IRIS düğümü bellek, işlemci, radyo haberleşme modülü ve geliştirme portundan oluşmaktadır. IRIS düğümleri Atmel Atmega1281 tabanlı, hesaplama ve radyo iletişimi aynı zamanda yapabilen ve düşük enerji tüketen bir mikroişlemciye sahiplerdir. Her IRIS düğümü 8 kB RAM bellek, 128 kB flash bellek ve 2.4 MHz bir radyo haberleşme modülüne sahiptir. Bu düğümler TinyOS [32] işletim sistemini desteklemektelerdir. TinyOS işletim sisteminin uygulamaları NesC programlama dili ile geliştirilebilir. IRIS düğümleri üzerinde bulunan 51 pinlik geliştirme konektörü analog girdiler, dijital I/O, I2C, SPI ve UART ara yüzlerini desteklemektedir. Bu ara yüzleri aracıyla, düğüm çeşitli sensörler veya MIB520 gibi başka donanımlara kolaylıkla bağlanabilir. Örneğin, bu konnektöre MDA100 isimli bir sıcaklık ve ışık sensörü bağlanabilir. IRIS düğümleri, MIB520 geçit aracıyla bilgisayarın USB portuna bağlanıp, ortamdan algıladıkları veya diğer düğümlerden gelen verileri bilgisayarda çalışan bir programa aktarabilirler. Şekil 4, deneylerde kullanılan IRIS, MDA100 ve MIB520 donanımlarını göstermektedir.

Tasarlanan platformda her IRIS düğümü çalışmaya başladıktan hemen sonra periyodik olarak Probe mesajı yayarak, ağa bağlı olan başka bir düğümü aramaya başlar. Her düğüm topladığı verileri baz istasyonuna iletmek için, ağa daha önce bağlanan başka bir hedef düğümünü kullanır. Bilgisayara bağlı olan baz istasyonu her zaman ağa bağlı olduğu için gelen tüm Probe mesajlara ACK mesajı göndererek diğer düğümlerin hedef düğümü olarak seçilir. Bir düğüm ACK mesajı aldıktan sonra, mesajın göndericisini hedef düğüm olarak seçip topladığı tüm verileri o düğüme aktarır. Hedef düğümü bulunan bir düğüm ağa bağlı sayılır ve gelen Probe mesajlara ACK mesajı gönderir ve böylece yeni eklenen düğümlerin hedef düğümü olarak seçilebilir. Böylece baz istasyonun radyo haberleşme menzilinde bulunmayan düğümler de çok atlamalı bağlantılar aracıyla verilerini, baz istasyonuna iletebilirler. Ortada bulunan hedef düğümler, diğer düğümlerden gelen verileri kendi hedef düğümlerine aktararak verilerin baz istasyonuna ulaşmasını sağlarlar. Bu durumda, ağ alanını genişletmek için yeni bir düğümü ağa bağlı olan mevcut bir düğümün radyo haberleşme menziline konumlandırmamız yeterli olur ve böylece ağ alanı kolaylıkla genişletilebilir.

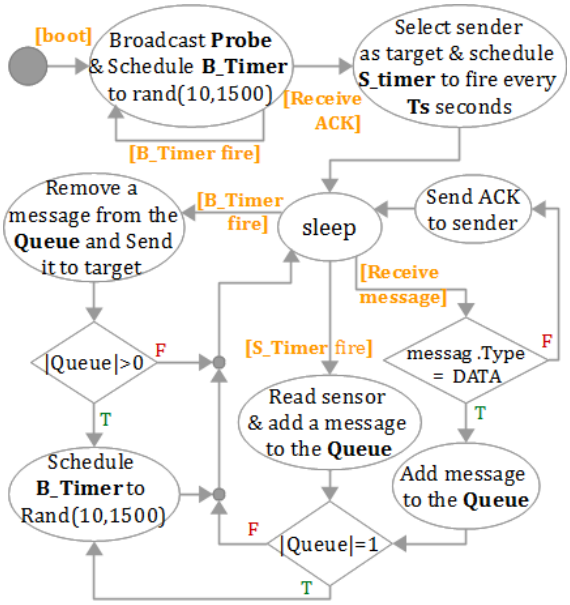


Şekil 4. Deneylerde kullanılan IRIS, MDA100 ve MIB520 donanımları
(Used IRIS, MDA100, and MIB520 devices in the experiments)

Hedef düğümünü seçtikten sonra, her düğüm periyodik olarak ortamın sıcaklık ve ışık seviyesini ölçüp, kendi pilinin voltaj değeri ile beraber hedef düğümüne gönderir. Hedef düğümler gelen mesajları bir kuyruğa ekleyerek kendi mesajlarıyla beraber bir sonraki düğüme aktarırlar. Gönderilen paketlerin çakışma olasılığını düşürmek için her düğüm bir mesajı gönderdikten sonra kuyruğunda bekleyen sonraki mesajı göndermeden önce rastgele bir süre bekler. Dolayısıyla her düğümde sensörlerden verileri toplamak için bir Timer ve kuyrukta bekleyen mesajları göndermek için başka bir Timer bulunmalıdır. Uygulamada BTimer ismi verdiğimiz timer 10 ms ile 1500 ms arasında rastgele sürelerde tetiklenir ve eğer kuyrukta bir mesaj varsa, o mesajı hedef düğüme aktarır. STimer isimli başka bir timer de dakikada bir kez tetiklenir ve sensörden gelen verileri bir mesaj içinde kuyruğa ekler.

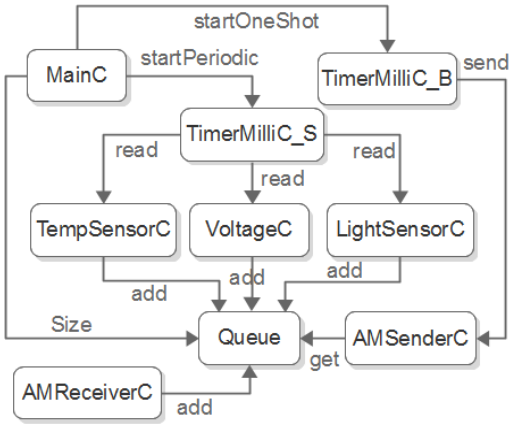
Şekil 5, tasarlanan gömülü yazılımın etkinlik şemasını göstermektedir. Düğüm çalışmaya başladıktan sonra, ACK mesajı almadığı sürece, periyodik olarak, Probe mesajı yaymaya devam eder. ACK mesajını aldıktan sonra, göndericiyi hedef düğüm olarak seçip, STimer'i her Ts saniyede bir kez (denemelerde 60 s seçilmiş) tetiklenmek üzere başlatır ve uyku durumuna geçer. STimer tetiklendiği zaman, düğüm sensör verilerini okuyup bu verileri bir mesaj olarak kuyruğa ekler. Eğer mesajı ekledikten sonra kuyrukta tek bir mesaj bulunursa, BTimer 10 ile 1500 ms arasında rastgele bir süreden sonra, bir kez tetiklenmek üzere, başlatılır. BTimer tetiklendiği zaman bir mesaj kuyruktan alınır ve hedef düğüme gönderilir ve eğer kuyrukta başka mesaj varsa BTimer tekrar (benzer bir

şekilde) ayarlanır. Her düğüm bir DATA mesajı aldığı zaman o mesajı kuyruğa ekler ve gerekirse, mesajı göndermek için BTimer'i ayarlar. Eğer gelen mesajın tipi DATA değilse, o zaman bir Probe mesajı alınmıştır ve bu durumda düğüm göndericiye bir ACK mesajı göndererek yeni düğümün ağa katılmasını sağlar. .



Şekil 5. Gömülü yazılımın etkinlik şeması
(Activity diagram of embedded software)

Şekil 6 tasarlanan gömülü yazılımın en önemli bileşenlerini göstermektedir. Uygulamanın ana bileşeni olan MainC, TimerMiliC_S bileşeninin periyodik ve sabit sürelerde tetiklenmesini sağlar. TimerMiliC_S bileşeni her tetiklendiğinde sıcaklık, ışık ve voltaj sensörlerini okuyan bileşenlerden read fonksiyonunu çağırır. Okuma işlemi tamamlandığında her bileşen add metodunu çağırarak verisini kuyruğa ekler. MainC bileşeni kuyruğun boyutunu Size metoduyla kontrol eder ve gerekirse TimerMiliC_B bileşeninden startOneShot metodunu çağırarak kuyruktaki mesajların gönderilmesini başlatır. TimerMiliC_B bileşeni tetiklendiği zaman AMSenderC bileşeninden send metodunu çağırarak bir mesajı kuyruktan alıp hedef düğüme gönderir. Düğüme gelen mesajları, AMReceiverC bileşeni alıp add metodunu çağırarak kuyruğa ekler.



Şekil 6. Gömülü yazılımın ana bileşenleri
(Main components of embedded software)

Düğümler arasında gönderilen mesajlar ve bu mesajlarda kullanılan sabit değerler aşağıdaki sözde kodda verilmiştir. Deneysel çalışmalarda, PROBE, DATA ve ACK mesaj tipleri ve TEMP, LIGHT ve VOLT veri tipleri bulunmaktadır. Gönderilen her mesajda, alıcı ve göndericinin kimlik numarasıyla beraber, mesaj tipi, sıcaklık, ışık ve voltaj değerleri bulunmaktadır.

```
#define Message Type: PROBE=1, DATA =2, ACK=3, TEMP=4,
LIGHT=5, VOLT=6;
struct MESSAGE {
  uint8 type;
  int16 sender, receiver;
  int16 lightdata, tempdata,  voltdata;
};
```

Her düğüm çalışmaya başladıktan hemen sonra radyo haberleşme donanımını açmaya çalışır. Bu donanım açıldığında aşağıdaki verilen AMControl.startDone metodu çağrılır. Eğer radyo modülü başlatılırken bir hata oluşursa AMControl.start metodu tekrar çağrılır. Eğer radyo haberleşme cihazı sorunsuz çalışmaya başlarsa, düğüm startPeriodic metodunu çağırarak BTimer'in, 10 ms ile 1500 ms arasında rastgele bir süreden sonra tetiklenmesini sağalar.

```
event void AMControl.startDone(err) {
  if (err != SUCCESS)       call AMControl.start();
  else       call BTimer.startPeriodic( rand(10,1500) );
}
```

BTimer tetiklendiği zaman aşağıda verilen fire fonksiyonunu çağırılır. Eğer daha önce bir hedef düğüm bulunmadıysa (target değişkeni -1 ise), fire metodunda, bir PROBE mesajı yayımlanır. Aksı takdirde eğer, bir hedef düğüm seçilmiş ve mesaj kuyruğunda en az bir mesaj varsa, kuyruktan bir mesaj alınır ve hedef düğüme gönderilir. Mesaj gönderildikten sonra eğer kuyrukta başka bir mesaj varsa BTimer 10 ms ile 1500 ms arasında rastgele bir süre sonra tetiklenmek üzere tekrar ayarlanır.

```
event void BTimer.fired() {
    if (target == -1)
        send(PROBE , null, AM_BROADCAST_ADDR);
    else if (queue.size()>0){
        msg=queue.get( );
        send(DATA , msg , AM_BROADCAST_ADDR);
        if (queue.size()>0)
            call BTimer.startOneShot( rand(10,1500) );
    }
}
```

Komşularından mesaj alan her düğümde aşağıda verilen Receive.receive fonksiyonu çağrılır ve mesaj parametre olarak bu fonksiyona gönderilir. Hedef düğümü olmayan bir düğüm ACK mesajı alırsa, göndericiyi hedef düğüm olarak seçip sensörlerden veri okumak amacıyla STimer'i dakikada bir kez tetiklenmek üzere başlatır. Daha önce hedef düğümü seçen bir düğüm DATA mesajı alırsa mesajı kuyruğa ekleyip gerekirse (kuyrukta tek bir mesaj varsa) BTimer'in rastgele bir süre sonra tetiklenmesini sağlar. Hedef düğümü bulunan bir düğüm PROBE mesajı alırsa, göndericiye bir ACK mesajı gönderir.

```
event Message AMReceive.receive(pkt) {
  if (target == -1){
       if (pkt.type==ACK){target=pkt.sender;
           call STimer.startPeriodic( 60000 );}
       }
   else{
       if (pkt.type==DATA && pkt.receiver==MY_ID){
            pkt.receiver=target;
            queue.add ( pkt );
            if (queue.size()==1)
                call BTimer.startOneShot( rand(10,1500) );
       }
       else if (pkt.type==PROBE)    send(ACK , null , pkt.sender);
  }
  return pkt;
}
```

STimer başladıktan sonra aşağıda verilen STimer.fire metodu her dakikada bir kez çağrılır ve sensörlerin anlık verileri okunur. Her sensörden veri okuma işlemi tamamlandığı zaman ilgili sensör bileşenin readDone metodu çağırılır ve elde edilen veri kuyruğa eklenir.

```
event void STimer.fired() {
           call Read.read();
           call TempReader.read();
           call ReadBatteryVoltage.read();
}
event void VoltSensorC.readDone(err, data){
    if (err == SUCCESS)    queue.add( VOLT , data );
  }
event void TempSensorC.readDone(err, data){
    if (err == SUCCESS)    queue.add( TEMP , data );
}
event void LightSensorC.readDone(err, data){
    if (err == SUCCESS)    queue.add( TEMP , data );
}
```

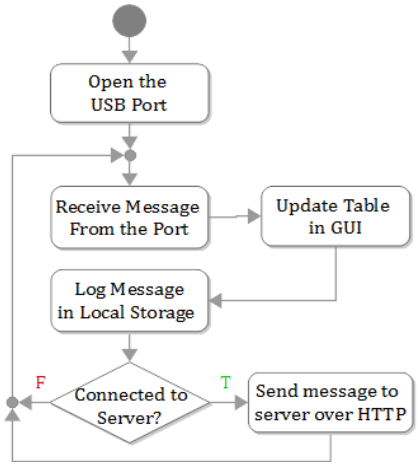
Her düğüm, komşuları veya hedef düğümüne bir mesaj göndermek için aşağıda verilen send metodunu çağırır. Bu metotta bir mesaj oluşturulur, mesajın tipi ve verileri parametre olarak gelen veriler ile doldurulur ve AMSend.send metodu çağrılarak radyo modülü aracıyla başka düğüme gönderilir.

```
void send( type, pkt, receiver){
    AMSend.send(receiver, pkt, sizeof(MESSAGE));
    call Leds.led1Toggle();
}
```

Düğümler tarafından gönderilen veriler tek veya çok atlamalı bağlantılar üzerinden baz istasyonuna ulaşır ve bilgisayara aktarılır. Bu veriler bilgisayarda bir uygulama aracıyla okunup gereken birleştirmeler yapıldıktan sonra İnternet aracıyla bulut üzerine aktarılır. Yerel veri toplama uygulamasının detayları bir sonraki bölümde verilmiştir.

4.2. Yerel Veri Toplama Uygulaması (Local Data Gathering Application)

Geliştirilen platformda, ortamdan toplanan veriler yerel bilgisayarda çalışan bir Java uygulamasına aktarılır. Bu uygulama sensörler tarafından gönderilen verileri, USB porta bağlı MIB520 geçit aracıyla okuyup, gereken düzeltmeleri yaptıktan sonra bulut üzerinde çalışan sunucuya aktarır. Şekil 7, yerel veri toplama uygulamasının akış şemasını göstermektedir. Uygulama çalışmaya başladıktan sonra geçidin bağlı olduğu portu açıp dinlemeye başlar. Her yeni mesaj geldiğinde, mesaj ara yüzde bulunan tabloda gösterilir ve yerel kayıt (log) dosyasına eklenir. Eğer uygulama sunucuya bağlı ise mesaj internet üzerinden sunucuya gönderilir.



Şekil 7. Yerel veri toplama uygulamanın akış şeması
(Flow Chart of local data gathering software)

Porttan veri beklerken, ara yüzüyle kullanıcının etkileşiminin devam etmesi için, port açma ve porttan veri okuma işlemleri Thread sınıfından türev alan PortListener sınıfında yapılmaktadır. Bu sınıfın yapıcı metodu porttan veri okumak için BuildSource. makePacketSource metodunu çağırarak bir PacketSource nesnesi üretir.

```
public class PortListener extends Thread {
  private PacketSource SerialPortReader;
  private Frame window;

  public PortListener(Frame frm, String[] args, String source) {
    window = frm;
    SerialPortReader = BuildSource.makePacketSource(source);
  }

  @Override
  public void run() {
      while (true) {
          pkt = SerialPortReader.readPacket();
          window.processMessage(pkt);
      }
  }
}
```

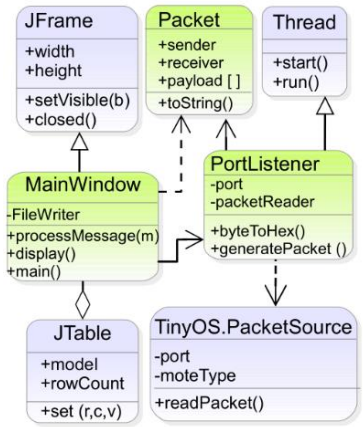
TinyOS işletim sisteminin Java kütüphanesinde sunulan BuildSource.makePacketSource metodu, geçidin port numarası ve bu porta bağlı donanımın (IRIS düğümü ve MIB520 geçidi) bilgilerini string formatında alır ve bir PacketSource nesnesi üretir. Bu nesne aracıyla geçit tarafından gönderilen paketler byte dizisi olarak alınır. Sınıfın Run metodunda bir sonsuz döngü içerisinde reader.readPacket() metodu çağırılır. Bir paket gelene

kadar Thread bu noktada bekletilir. Paket geldikten sonra generatePacket metodu çağrılarak, veriler byte dizisi formatından nesne formatına çevrilir ve Window.processMessage metodu çağrılarak ara yüz nesnesine gönderilir. Tasarlanan uygulamanın ara yüzünde, kullanıcı geçidin bağlı olduğu port bilgilerini belirleyerek geçide bağlanır ve her sensörden gelen veriyi bir tabloda görebilir. Aşağıda verilen ara yüzü sınıfında, jButton1ActionPerformed metodu çağrıldığı zaman (kullanıcı connect butonunu tıkladığı zaman) bir PortListener nesnesi oluşturulur ve bu nesnenin start metodu çağrılarak geçitten gelen paketlerin okunması başlatılır.

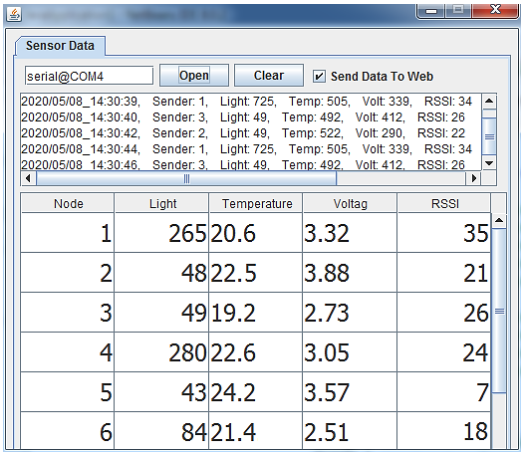
```
public class Frame extends JFrame {
   private    PortListener serialPortGateWay;
   private    FileWriter writer;

   private void StartButtonClick(ActionEvent evt) {
       serialPortGateWay = new PortListener(this, new
       String[]{"-comm", PortNumber.getText() + ":iris"});
       serialPortGateWay.start();
       writer = new FileWriter("log.txt", true);
    }

synchronized   void processMessage(Packet pkt) {
  UpdateTable(jTable1, pkt)
  writer.write(pkt.toString());
  if(jCheckBox1.isSelected())
            sendToWeb(pkt);
}
private  void sendToWeb(Packet pkt) {
      URL url = new URL("http:// "+server+"/submit_data?"+
      pkt.toString());
      HttpURLConnection con = url.openConnection();
      con.setRequestMethod("GET");
      con.setRequestProperty("Content-type", "text/xml");
      con.connect();
      con.getResponseCode();
      con.disconnect();
   }
```

PortListener, geçitten okunan her paketi processMessage metoduna gönderir. Bu metotta paketin içindeki bilgiler tabloya aktarılır ve eğer uygulama sunucuya bağlı ise sendToWeb metodu çağrılarak, paket sunucuya gönderilir. sendToWeb metodu sunucuda "/submit_data" adresine paket bilgilerini içeren bir GET mesajı gönderir ve bağlantıyı hemen kapatır.

Yerel veri toplama uygulamasının sınıf diyagramı Şekil 8'de gösterilmiştir. Bu şekilde mavi renk ile gösterilen sınıflar Java veya TinyOS kütüphanesinin standart sınıfları ve yeşil renk ile gösterilen sınıflar geliştirilen uygulamanın sınıflarıdırlar. MainWindow sınıfı uygulamanın başlangıç noktası ve aynı zamanda ara yüzüdür. Bu sınıf PortListener sınıfını kullanarak paketleri USB portundan alıp ekranda gösterir. PortListener sınıfı PacketSource nesnesini kullanarak paketleri porttan okuyup, MainWindow sınıfına aktarır. Uygulamanın örnek ekran görüntüsü Şekil 9'da verilmiştir.



Şekil 8. Yerel veri toplama uygulamanın sınıf diyagramı
(Class diagram of local data gathering software)



Şekil 9. Yerel veri toplama uygulamanın ekran görüntüsü.
(Screenshot of local data gathering software)

4.3. Node JS Tabanlı Sunucu (NodeJS Based Server)

Son yıllarda NodeJS hızlı, güvenilir ve verimli web sunucuların geliştirmesini sağlayarak popüler bir arka uç teknolojisine dönüşmektedir. NodeJS etkili, aktif, açık kaynaklı ve devasa bir JavaScript tabanlı kütüphaneye sahiptir. Genel olarak, NodeJS sürümlerinin arasında büyük uyumsuzluk problemi yaşanmamaktadır ve bu platformda geliştirilen uygulamalar yeni eklenen modüller ile güncellenebilmektedir. NodeJS'in hafif yapısı sayesinde, projenin birden fazla örneğini farklı sunuculara dağıtarak uygulama yatay olarak ölçeklendirilebilir. JavaScript dilini kullanan NodeJS, web tabanlı uygulamalarda arka ve ön uçtaki uygulamaların ortak bir dil kullanmalarını sağlamaktadır. NodeJS, bloklanmayan işlemleri desteklediği için, veritabanından veri okuma gibi IO işlemleri asenkron olarak yapılabilir. Böylece, aynı zamanda çok sayıda istemciye yanıt verilebilir. NodeJS 2009'da oluşturulduğundan itibaren açık kaynak bir ortam olarak, birçok şirket ve yazılımcı tarafından incelenip, test edilmiştir. Dolayısıyla NodeJS kütüphanesinde bulunan çok sayıda önemli modüller kabul edilebilir bir güvenilirlik düzeyine erişmişlerdir. Bu sebeplerden dolayı, web tabanlı sunucumuzu NodeJS ile geliştirdik.

Yerel veri toplama uygulaması, HTTP protokolü üzerinden bir GET mesajıyla topladığı verileri bulut üzerinde NodeJS [33] tabanlı bir sunucuya aktarır. Aşağıda gösterildiği gibi, NodeJS uygulamasında, express, http ve websocket modüllerini kullanılarak, aynı zamanda hem web ve hem websocket sunucuları oluşturulmuştur. Websocket üzerinden bağlanan her istemcinin bağlantısı clist dizisine eklenir ve ilerleyen zamanlarda Java uygulamasından gelen veriler bu dizide bulunan istemcilere aktarılır. Böylece, kullanıcılar tarayıcı sayfalarını yenilemeden, güncel sensör verilerini görebilirler.

```
var express = require('express');
var http = require('http');
var app = express();
var WebSocket= require('websocket').server;

var server = app.listen(80);

var clist=[];
var httpserv=http.createServer(function(req,res){});
httpserv.listen(8080);
var websocketserver= new WebSocket({httpServer: httpserv});
websocketserver.on('request', function(req){
    clist.push(req.accept(null, request.origin));
});
```

REST API'leri uygulamak için express kütüphanesi kullanılarak "/" ve "/submit_data" olmak üzere iki ana adreste GET isteklerine yanıt verilmektedir. Aşağıda gösterildiği gibi, "/" adresine gelen tüm GET isteklerine, uygulamanın ana ara yüzünü içeren index.html sayfası gönderilmektedir. "/submit_data" adresine gelen tüm GET isteklerinin query bölümünden, paket bilgileri okunup, JSON formatında veri tabanına ve sunucuya bağlı olan tüm istemcilere aktarılır.

```
var mysql = require("mysql");
var database = mysql.createConnection({
host: "localhost",  user: "admin",   password: "admin123*" });
database.connect(function(err) {    if (err) throw err;   });
app.get('/', function(req,res){res.sendFile("index.html" )});
app.get('/submit_data/', function (req, res) {
  res.status(200).send('ok');
  database.query("insert into messages (SenderID, Data)
  values("+req.query.sender+","+ JSON.stringify([req.query])));
  });
 if (wsconnection.length > 0) clist.forEach(async function(c) {
        c.send(JSON.stringify([req.query]));});
 });
```

Tüm alınan mesajlar aynı sunucuda bulunan bir MySQL veritabanına eklenir. NodeJS'in MySQL kütüphanesi kullanılarak, MySQL veritabanına bir bağlantı kurulur ve gelen her mesaj SQL Insert komutuyla veri tabanında message tablosuna eklenir.

4.4. Web Tabanlı İstemci Uygulaması (Web Based Client Application)

Tasarlanan sistemin son kullanıcıları, ortamdan toplanan verilere web sayfaları üzerinden erişebilirler. Web tabanlı istemci uygulaması JavaSript ve HTML dilinde geliştirilmiştir. GoogleChart kütüphanesi kullanılarak, toplanan veriler çeşitli grafikler şeklinde kullanıcıya sunulur. Kullanıcı, tarayıcı üzerinden, http protokolüyle sunucunun "/" adresine bir GET mesajı gönderdikten sonra index.html sayfasına erişir. Bu sayfa kullanıcının tarayıcısına yüklendikten sonra aşağıda gösterilen JavaScript fonksiyonunu çağırarak, sunucuya 8080 port üzerinden bir websocket bağlantısı kurar.

```
function initWebSocket() {
    output = document.getElementById("packetlog");
    websocket = new WebSocket("ws://"+server+":8080/");
    websocket.onmessage = function (event) {
        sensordata = JSON.parse(event.data);
        sensordata.forEach(addData);
    };
}
```

Bağlantı kurulduktan sonra, websocket üzerinden gelen paketler için addData fonksiyonu çağırılır. Aşağıda gösterildiği gibi, addData fonksiyonu sensörün kimlik numarasını çevrimiçi (online) cihazların listesine ekledikten sonra (eğer daha önce eklenmediyse) paketin verilerini DataSet isimli iki boyutlu diziye aktarır. Bu dizinin her satırı bir veri türünü (Örneğin ışık, sıcaklık, RSSI, vs.) tutmaktadır. Her satırda ilgili veriyi içeren ve farklı sensörlerden gelen değerler bulunmaktadır. Veriler DataSet'e aktarıldıktan sonra GoogleChart kütüphanesi aracıyla farklı grafiklerde gösterilirler.

```
function addData(dt) {
  if (getSensor(dt.id) == null)  addSensor(dt.id + "");
  addToDataSet(DataSet, dt);
  var d=google.visualization.arrayToDataTable(DataSet[0]);
  lightChart.draw(d);
  d = google.visualization.arrayToDataTable(DataSet[1]);
  tempChart.draw(ldata);
}
```

Şekil 10. İstemci uygulamanın örnek ekran çıktısı
(Screenshot of client application)

Analitik grafikleri oluşturmak için load fonksiyonunda, GoogleChart kütüphanesi web sayfasına yüklenir. Yükleme işlemi tamamlandığında drawChart fonksiyonu çağrılır ve her veri türü için LineChart sınıfından bir nesne oluşturulur. Bu nesne bir HTML etiketine atanır ve verileri grafik şeklinde gösterir. Tasarlanan web sayfasının örnek

ekran çıktısı Şekil 10'da gösterilmiştir. Kullanıcılar bu grafikler aracıyla sensörlerden gelen anlık verileri görebilirler.

```
google.charts.load('current', { 'packages': ['corechart'] });
google.charts.setOnLoadCallback(drawChart);
function drawChart() {
    lightChart = new google.visualization.LineChart(
    document.getElementById('light_chart_div'));
    tempChart = new google.visualization.LineChart(
    document.getElementById('temp_chart_div'));
}
```

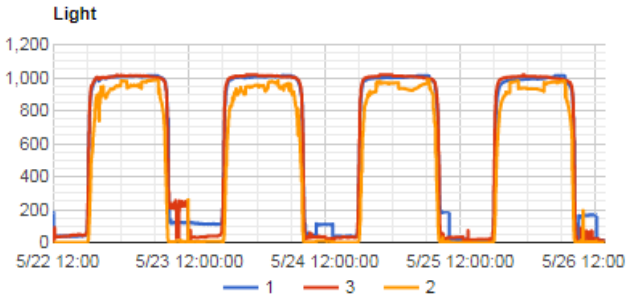
5. DENEYSEL ÇALIŞMLAR (EXPERIMENTS)

Geliştirilen platformu değerlendirmek için farklı odalara IRIS düğümlerini yerleştirip, ortamın sıcaklık ve ışık seviyelerini, düğümlerin voltaj değerlerini ve gönderilen paketlerin RSSI değerlerini 96 saat boyunca izledik. Sensörler her dakikada bir kez algıladıkları verileri tek veya çok atlamalı bağlantılar üzerinden 3 dBm gönderme gücüyle baz istasyonuna aktardılar. Deneylerde, 10 dakikalık süre zarfında sensörler tarafından gönderilen veriler Tablo 1'de gösterilmiştir.

Tablo 1. Sensörlerden gelen mesajların içeriği
(Received messages from the sensors)

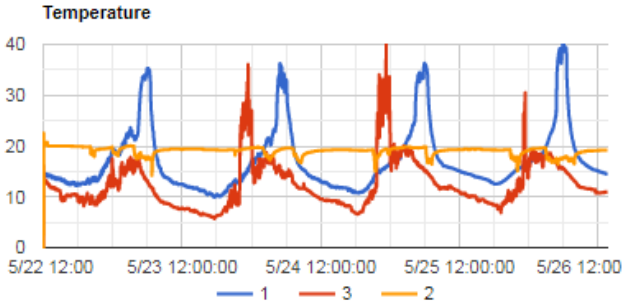
Time	ID	Light	Temp	Volt	RSSI
2020/04/22_00:00:00	1	36	14	3.55	12
2020/04/22_00:00:02	2	0	21.1	3.48	20
2020/04/22_00:00:02	3	28	14.2	3.41	15
2020/04/22_00:01:00	2	0	21	3.47	20
2020/04/22_00:01:01	3	29	14.1	3.41	15
2020/04/22_00:01:03	1	36	14	3.55	9
2020/04/22_00:02:02	1	37	14.2	3.55	12
2020/04/22_00:02:04	2	0	20.9	3.47	20
2020/04/22_00:02:05	3	30	13.7	3.41	15
2020/04/22_00:03:00	1	37	14.2	3.55	12
2020/04/22_00:03:02	3	30	13.7	3.40	15
2020/04/22_00:03:02	2	0	20.8	3.47	20
2020/04/22_00:04:01	2	0	20.8	3.47	20
2020/04/22_00:04:02	3	33	13.8	3.40	15
2020/04/22_00:04:04	1	37	14.3	3.55	12
2020/04/22_00:05:00	3	29	13.9	3.40	15
2020/04/22_00:05:01	2	0	20.8	3.47	21
2020/04/22_00:05:02	1	37	14.1	3.55	12
2020/04/22_00:05:04	3	29	13.9	3.40	15
2020/04/22_00:06:01	1	36	14	3.54	13
2020/04/22_00:06:03	2	0	20.7	3.47	20
2020/04/22_00:06:03	3	29	13.7	3.40	15
2020/04/22_00:07:00	1	36	14.3	3.54	12
2020/04/22_00:07:01	3	30	13.5	3.40	14
2020/04/22_00:07:02	2	0	20.7	3.47	20
2020/04/22_00:08:00	3	30	13.6	3.40	21
2020/04/22_00:08:01	2	0	20.6	3.47	21
2020/04/22_00:08:03	1	36	14.2	3.54	12
2020/04/22_00:09:02	1	37	14.2	3.54	12
2020/04/22_00:09:04	2	0	20.6	3.47	20
2020/04/22_00:09:05	3	30	13.7	3.40	15
2020/04/22_00:10:00	1	37	14.2	3.54	12
2020/04/22_00:10:02	2	0	20.6	3.47	21
2020/04/22_00:10:04	3	30	13.8	3.40	15
2020/04/22_00:10:06	1	37	443	318	12

Şekil 11 üç sensörden gelen ışık seviyelerini göstermektedir. Bu şekilde görüldüğü gibi, tüm sensörlerden gelen veriler gündüz ve gece periyotlarını net bir şekilde göstermektedir. Gece saatlerinde ortamda bulunan yapay ışıklar da sensörler tarafından tespit edilmiştir.

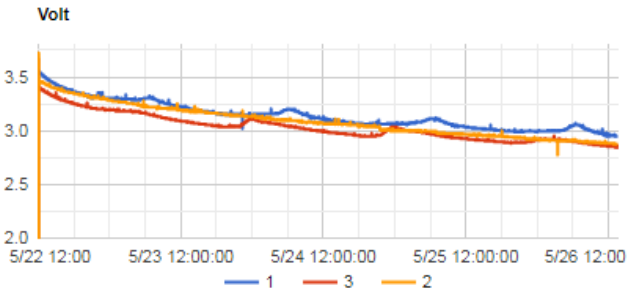


Şekil 11. Sensörlerden gelen ışık değerleri
(Received light data from the sensors)

Şekil 12 sensörlerin gönderdiği sıcaklık verilerini göstermektedir. Bu şekilde görüldüğü gibi 2 nolu sensörün bulunduğu odanın sıcaklığı hem gece ve hem gündüz 20 derecenin civarında ve sınırlı bir aralıkta değişmiştir. 2 ve 3 nolu düğümlerin bulunduğu odaların sıcaklığı, gece saatleri 10 dereceye düşerken gündüz saatleri 40 dereceye kadar yükselmiştir. 2 ve 3 nolu düğümlerin konumları güneş aldığından dolayı, öğlen saatlerinde bu sensörler yüksek sıcaklık verileri göndermişlerdir.



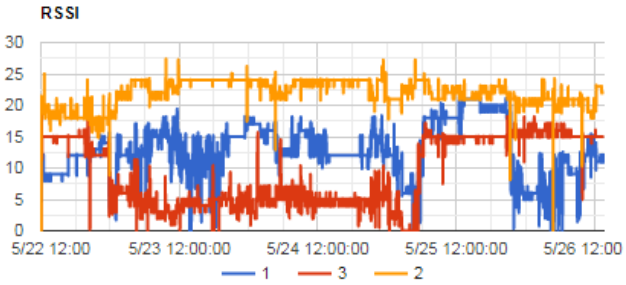
Şekil 12. Sensörlerden gelen sıcaklık değerleri
(Received temperature data from the sensors)



Şekil 13. Sensörlerden gelen voltaj değerleri
(Received voltage data from the sensors)

Şekil 13, düğümlerin gönderdiği voltaj değerlerini göstermektedir. Bu şekilde görüldüğü gibi, sensörlerin voltaj değerleri 3.5 civarından başlayıp deney boyunca yavaşça 2.8 volta kadar düşmüştür. Şekil 13'e göre sensörler, pillerinde bulunan enerjiyi hemen hemen doğrusal bir şekilde tüketmişlerdir. Şekil 14 alınan paketlerin RSSI değerlerini göstermektedir. Bu şekilde görüldüğü gibi 2 nolu düğümden gelen paketlerin RSSI değerleri daha yüksektir. Dolayısıyla, 2 nolu düğümün

hedef düğümüyle arasındaki mesafenin düşük olduğu veya aralarında hiçbir engelin olmadığı söylenebilir. 3 nolu düğümün gönderdiği paketlerin RSSI değeri genel olarak diğer düğümlerden düşük olduğundan dolayı, bu düğümün hedef düğümü ile arasındaki mesafenin yüksek veya aralarında bir engel bulunduğu söylenebilir. Genel olarak, paketlerin RSSI değerleri ortamda bulunan farklı faktörlerden etkilenebilir. Örneğin insanların hareket etmesi, ortamda bulunan nesnelerin değişmesi veya diğer cihazlardan gönderilen sinyaller, RSSI değerlerini etkileyebilir. Bu yüzden gönderilen RSSI değerlerinin anlık değişim oranı yüksek olabilir.



Şekil 14. Sensörlerden gelen RSSI değerleri
(Received RSSI data from the sensors)

6. SONUÇLAR (CONCLUSIONS)

Geleceğin önemli teknolojilerinden biri olan Nesnelerin İnterneti, Endüstri 4.0'ın en kritik bileşenleri içinde yer almaktadır. Hızla gelişen küçük, radyo iletişimi yapan, enerji tasarruflu aygıtlar ve sensörlerle beraber, Nesnelerin İnterneti yaygın bir şekilde izleme, kontrol, haberleşme ve otomasyon sistemlerinde kullanılmaya başlamıştır.

Bu çalışmada Nesnelerin İnterneti için bir veri toplama, raporlama ve analiz platformu geliştirilmiştir. Geliştirilen sistem, sensörlerden veri toplayan gömülü yazılımı, yerel veri toplama uygulaması, bulut üzerinde çalışan NodeJS tabanlı sunucu ve web tabanlı istemci yazılımı olmak üzere, dört ana bileşenden oluşmaktadır. Tasarlanan sistemde, donanımların üzerinde gömülü olarak çalışan uygulama verileri tek veya çok atlamalı bağlantılar üzerinden bir yerel bilgisayarda çalışan Java uygulamasına aktarır. Bu veriler Java uygulamasından, HTTP protokolü aracıyla bulut üzerinde çalışan bir sunucuya gönderilir. Son kullanıcılar, bir web tarayıcısıyla sunucuya bağlandıktan sonra, sensörlerden gönderilen verileri anlık olarak, çeşitli grafikler aracıyla izleyebilirler. Tasarlanan sistemi kullanarak, TDAlar üzerinde yapılan deneysel çalışmada, 96 saat boyunca, ortamın ışık ve sıcaklık, düğümlerin voltaj değerleri ve gönderilen sinyallerin RSSI değerleri ölçüp, çeşitli analitik grafikler oluşturduk. Önerilen sistemde bileşenlerin birbirinden tamamen bağımsız olması, platformun farklı alanlara kolaylıkla uygulamasını sağlamaktadır. Dolayısıyla, tasarlanan platform sınırlı güncellemeler ardından, çeşitli alanlar ve farklı uygulamalarda kullanılabilir.

Gelecek çalışmalarda, Arduino, Node MCU ve cep telefonlarının sensörlerinden gelen verilerin desteklenmesi, aynı değeri ölçen farklı sensör ve aygıtlardan gelen verilerin karşılaştırılması, donanımların hassasiyet analizi ve yapay zeka teknikleri ile toplanan verilerden anlamlı bilgilerin üretilmesi hedeflenmiştir. Ayrıca, çeşitli platformların kullanım kolaylığı, performansı, avantajları, zaafiyetleri, desteklediği donanımlar, diller ve protokolleri karşılaştırmak için, benzer bir uygulamanın [17-28]'de sunulan platformlarda geliştirilmesi hedeflenmiştir. Son olarak, [34]'te verilen çalışmaya benzer bir mobil ara yüzün tasarlanması ve topolojinin bağlılık durumunu kontrol etmek için [35, 36, 37]'deki yöntemler ve modüllerin bütünleştirilmesi hedeflenmektedir.

KAYNAKLAR (REFERENCES)

[1] Yu, Lei, Yang Lu, and XiaoJuan Zhu. "Smart hospital based on internet of things", *Journal of Networks*, 7(10), 1654, 2012.

[2] N. Karimpour, B. Karaduman, A. Ural, M. Challenger, and O. Dagdeviren, "IoT based Hand Hygiene Compliance Monitoring", **2019 International Symposium on Networks, Computers and Communications (ISNCC)**, pp. 1-6. IEEE, 2019.

[3] M. Kamal, M. Atif, H. Mujahid, T. Shanableh, A. Al-Ali, and A. Al Nabulsi. "IoT based smart city bus stops", *Future Internet*, 11(11), 227, 2019.

[4] J. Jalaney, R. S. Ganesh, "Review on IoT Based Architecture for Smart Public Transport System", *International Journal of Applied Engineering Research*, 14(2), 466-471, 2019.

[5] P.Kuppusamy, , R. Kalpana, P. Venkateswara, "Optimized traffic control and data processing using IoT", *Cluster Computing*, 22(1) 2169-2178, 2019.

[6] R. A. Dex, S. Djahel, "An IoT Enabled Traffic Light Controllers Synchronization Method for Road Traffic Congestion Mitigation", **2019 IEEE International Smart Cities Conference (ISC2)**, pp. 709-715. IEEE, 2019.

[7] P. Kapoor, F.A. Barbhuiya, "Cloud Based Weather Station using IoT Devices.", **IEEE Region 10 Conference TENCON 2019**, 2357-2362. IEEE, 2019.

[8] F.J. Joseph, "IoT Based Weather Monitoring System for Effective Analytics", *International Journal of Engineering and Advanced Technology*, 8(4), 311-315, 2019.

[9] O. Hamdan, H. Shanableh, I. Zaki, A. R. Ali,T. Shanableh, "IoT-based interactive dual mode smart home automation", **2019 IEEE International Conference on Consumer Electronics (ICCE)**, 1-2. IEEE, 2019.

[10] L. Özgür, V. K. Akram, M. Challenger, O. Dağdeviren, "An IoT based smart thermostat", **5th International Conference on Electrical and Electronic Engineering (ICEEE)**, 252-256, IEEE, 2018.

[11] A. P. Plageras, E. P. Kostas, S. Christos, H. Wang, B. Gupta, "Efficient IoT-based sensor BIG Data collection–processing and analysis in smart buildings", *Future Generation Computer Systems* 82, 349-357, 2018.

[12] M. Huang, A. Liu, T. Wang, C. Huang, "Green data gathering under delay differentiated services constraint for internet of things", *Wireless Communications and Mobile Computing*, 2018.

[13] A. Coates, M.Hammoudeh, K. G. Holmes, "Internet of things for buildings monitoring: Experiences and challenges", **International Conference on Future Networks and Distributed Systems**, 2017.

[14] R. Deng, Z. Zhang, J. Ren, and H. Liang, "Indoor Temperature Control of Cost-Effective Smart Buildings via Real-Time Smart Grid Communications", **IEEE Global Communications Conference (GLOBECOM)**, 1–6, 2016.

[15] R. Almeida, R. Oliveira, D. Sousa, M. Luis, Senna, C.Sargento, S, "A multi-technology opportunistic platform for environmental data gathering on smart cities", **IEEE Globecom Workshops (GC Wkshps)**, 1-7, 2017, IEEE.

[16] A. Alavi, A. Rahimian, K. Mehran, J. Alaleddin, "An IoT-based data collection platform for situational awareness-centric microgrids", **IEEE Canadian conference on electrical & computer engineering (CCECE)**, 1-4. IEEE, 2018.

[17] D. Vasisht, , Z. Kapetanovic, J. Won, X. Jin, R. Chandra, S. Sinha, A. Kapoor, M. Sudarshan, S. Stratman, "Farmbeats: An iot platform for data-driven agriculture", **14th Symposium on Networked Systems Design and Implementation (NSDI 17)**, 515-529, 2017.

[18] L. Jieyin, S. Zhou, H. Liu, X. Zhao, "LEO IoT based big data management and analysis platform design for intermodal containers", **IOP Conference Series: Materials Science and Engineering**, 715(1), 012029. IOP Publishing, 2020.

[19] A.F. Fuentes, E. Tamura, "LoRa-Based IoT Data Monitoring and Collecting Platform", **Ibero-American Congress on Information Management and Big Data**, 80-92. Springer, Cham, 2019.

[20] S. Khan, A. S. K. Pathan, N. A. Alrajeh, **Wireless sensor networks: Current status and future trends**, CRC press, 2016.

[21] F. Terroso-Saenz, , A.González-Vidal, , A. P.Ramallo-González, , & A. F.Skarmeta, "An open IoT platform for the management and analysis of energy data". *Future Generation Computer Systems*, 92, 1066-1079, 2019.

[22] M. A.Zamora, J.Santa, J. A.Martínez, V.Martínez, A. F.Skarmeta, "Smart farming IoT platform based on edge and cloud computing", *Biosystems engineering*, 177, 4-17, 2019.

[23] V. Choudhary, J. H. Teh, V. Beltran, H. B. Lim, "AirQ: A Smart IoT Platform for Air Quality Monitoring". **2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)**, pp. 1-2, IEEE, 2020.

[24] A. Rashed, , A. Ibrahim, , A. Adel, , B. Mourad, , A. Hatem, , M. Magdy, A. Khattab, "Integrated IoT medical platform for remote healthcare and assisted living", **2017 Japan-Africa Conference on Electronics, Communications and Computers** (JAC-ECC), pp. 160-163, IEEE, 2017.

[25] Internet: KAA Project, http://www.kaaproject.org.

[26] Internet: IBM IoT, https://internetofthings.ibmcloud.com.

[27] Internet: Oracle IoT, https://cloud.oracle.com/iot.

[28] Internet: Thing Speak, https://thingspeak.com

[29] Internet: Memsic Wirless Snsor Networks, IRIS datasheet, http://www.memsic.com/userfiles/files/Datasheets/WSN/IRIS_Datasheet.pdf,16.05.2020.

[30] Internet: mda100 sensor board, https://www.memsic.com/wireless-sensor-networks/mda100, 16.05.2020

[31] Internet: CMT mib520 usb gateway, http://www.cmt-gmbh.de/Produkte/WirelessSensorNetworks/MIB520.html,16.05.2020.

[32] M. Amjad, , M. Sharif, M. K. Afzal, S. W. Kim, "TinyOS-new trends, comparative views, and supported sensing applications: A review", *IEEE Sensors Journal*, 16(9), 2865-2889, 2016.

[33] S. Tilkov, S.Vinoski, "Node. js: Using JavaScript to build high-performance network programs", *IEEE Internet Computing*, 14(6), 80-83, 2010.

[34] U. Çabuk, O. Dağdeviren, Y. Yiğit, M. Süvari, "Gömülü Sistemler İçin Android Tabanlı Bir Mikroişlemci Programlama Yazılımı ve Arayüzü", *Bilişim Teknolojileri Dergisi*, 11 (4), 321-332, 2018.

[35] O. Dağdeviren, V. Akram, "Energy-efficient bridge detection algorithms for wireless sensor networks", *International Journal of Distributed Sensor Networks*, 9(4), 867903, 2013.

[36] O. Dagdeviren, ,V. K. Akram, "Pack: Path coloring based k-connectivity detection algorithm for wireless sensor networks", *Ad Hoc Networks*, 64, 41-52, 2017.

[37] O. Dağdeviren, V. Akram, "TinyOS Tabanlı Telsiz Duyarga Ağları için Bir Konumlandırma ve k-Bağlılık Denetleme Sistemi", *Bilişim Teknolojileri Dergisi*, 10(2), 139-152, 2017.