Avrupa Bilim ve Teknoloji Dergisi
Sayı 35, S. 489-496, Nisan 2022

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 35, pp. 489-496, April 2022

Research Article

Blok Zincir ve Ulak (Oracles) İle Parametrik İşlemlerin Gerçekleştrilmesi

Mansur Beştaş[1*]

[1*] Siirt University, Siirt, Turkey, (ORCID: 0000-0002-8192-2044), mansur@siirt.edu.tr

(İlk Geliş Tarihi 26 Mart 2022 ve Kabul Tarihi 30 Nisan 2022)

(**DOI:** 10.31590/ejosat.1093723)

ATIF/REFERENCE: Beştaş, M. (2022). Blok Zincir ve Ulak (Oracles) İle Parametrik İşlemlerin Gerçekleştrilmesi. *Avrupa Bilim ve Teknoloji Dergisi*, (35), 489-496.

Öz

Blok zincir, verinin kurcalamaya karşı güvenli olarak saklanmasına yönelik geliştirilmiş bir teknolojisidir. Blok zincir teknolojisinin daha etkin kullanılması ve merkezi olmayan uygulamaların ortaya çıkışı akıllı sözleşmeler aracılığı ile gerçekleşmiştir. Ancak akıllı sözleşmeler üzerinde çalıştıkları blok zincir dışından veri alamamaktadır. Bu problem gerçek yaşam problemlerinin çözümü adına projelerin ortaya çıkışını sınırlandırmaktadır. Örneğin döviz bilgileri ile ilgili bir uygulamada gerekli veri kaynağı blok zincir dışıdır. Tam bu noktada blok zincir ile dış dünya arasında bir köprüye ihtiyaç duyulmaktadır. Blok zincir ulakları blok zincir içi ve dışı verileri arasındaki iletişim boşluğunu doldurmaktadır.
Bu çalışmada blok zincir teknolojisi ve akıllı sözleşmeler hakkında bilgi verilmiştir. Ardından Blok zincir ulakları tanıtılmış olup, ulak teknolojisi mimarisi, akademik çalışma istatistiği, bu alandaki belli başlı ticari girişimler hakkında bilgi verilmiştir. Çalışmanın son bölümünde blok zincir ulak sisteminin oluşturulması için örnek çalışma yapılmıştır. Yapılan çalışmada blok zincir içine ve dışına verilerin aktarılması için gerekli bir sistemin temel taşını oluşturan mimari teorik ve uygulamalı olarak ortaya koyulmuştur.

Anahtar Kelimeler: Blok zincir, Akıllı sözleşme, Blok zincir ulakları.

Performing Parametric Transactions with Blockchain and Oracles

Abstract

Blockchain is a technology developed for the tamper-proof storage of data. The more effective use of blockchain technology and the emergence of decentralized applications have been realized through smart contracts. However, smart contracts cannot receive data from outside the blockchain they are working on. This problem limits the emergence of projects for the solution of real-life problems. For example, in an application related to currency trading, the required data source is off-blockchain. At this point, a bridge is needed between the blockchain and the off-blockchain world. Blockchain oracles fill the communication gap between on- and off-blockchain data.
In this research, served information about blockchain technology and smart contracts are given. Then, Blockchain oracles were explained, and information was given about the Blockchain oracle technology architecture, academic study statistics, and major commercial initiatives in this field. In the last section of the research, a case study was demonstrated for the creation of the Blockchain oracles system. In the research, the architecture that constitutes the pilar of a system necessary for transferring data into and out of the blockchain has been revealed theoretically and practically.

Keywords: blockchain, Smart contract, Blockchain oracles

[*] Corresponding Author: mansur@siirt.edu.tr

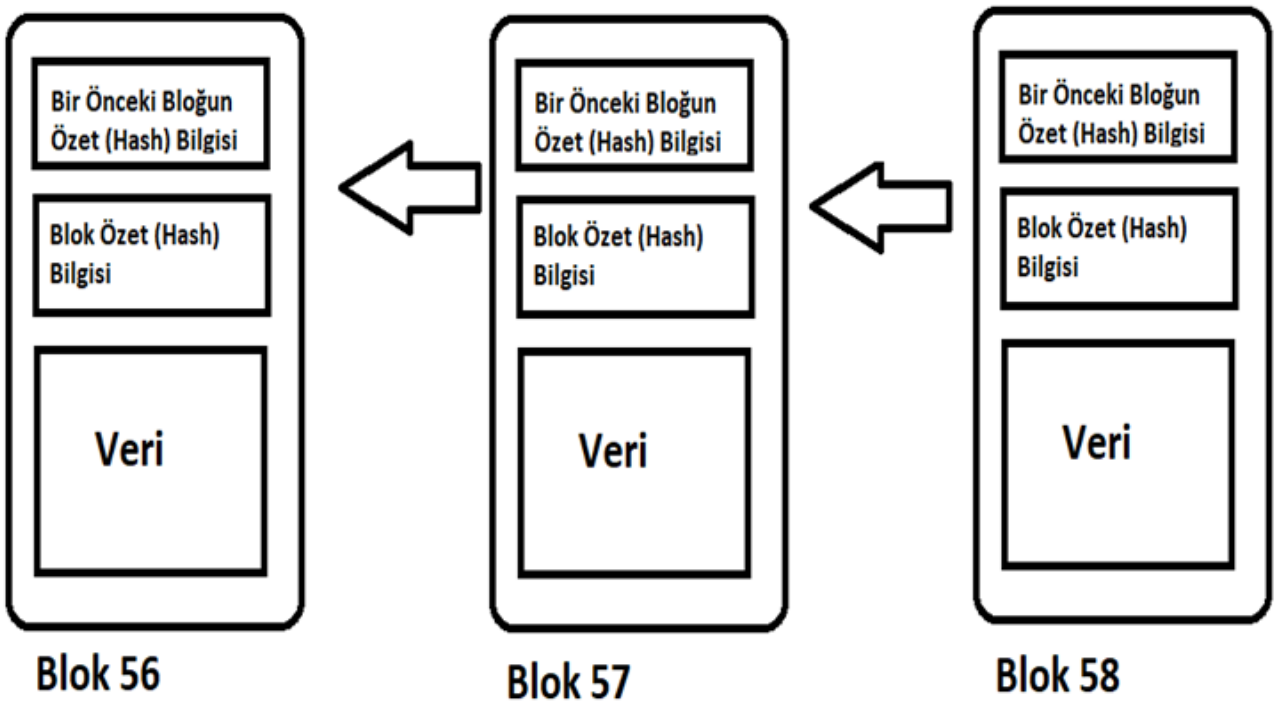
1. Giriş

Teknoloji, işletmelerin iş yapış şeklini ve günlük hayatımızı sürekli olarak değiştirmektedir. Teknolojinin temel unsurlarından biri verinin yönetilmesidir. Teknolojinin getirdiği kolaylıklardan biri işlemlerin insan unsurunun azaltılması ve süreç içerisinde elde edilen verinin kaydedilmesi, değiştirilmediğinden emin olmaktır. Verinin yönetilmesi ve değiştirilmediğinden emin olma ihtiyaının getirdiği bir sonuç olarak blok zincir teknolojisi ortaya çıkmıştır. Blok zincir teknolojisi temel olarak verinin güven içerisinde saklanma ihtiyacını çözmektedir. Ancak zaman içerisinde sadece veri değil yazılım kodlarının da değiştirilmediğinin güvence altına alınma ihtiyacı ortaya çıkmıştır. Blok zincir yazılım alanında bu ihtiyacı akıllı sözleşme ile çözmüştür. Akıllı sözleşme, blok zincir üzerinde saklanan ve önceden belirlenmiş şartlara göre davranan program kodlarıdır (Szabo, 1994). Blok zincir teknolojisi ve akıllı sözleşmeler tasarım gereği üzerinde çalıştığı ağ üzerindeki bilgilere ulaşmaktadır. Ancak gerçek hayat problemleri birden fazla kaynaktan bilgi edinimini zorunlu kılmaktadır. Blok zincir ağı dışından veri elde edinimi daha önceden belirlenmiş veri kaynakları (blok zincir ulakları) ile gerçekleştirilmektedir.

Bu çalışmada blok zincir teknoloji üzerinde oluşturulan akıllı sözleşmenin ağ harici veri kaynağına erişim yöntemi ele alınmıştır. Bu kapsamda blok zincir teknolojisi incelenmiştir. Akıllı sözleşmelerin, blok zincir temelinde oluşturulmuş uygulamalardaki rolü ele alınmıştır. Teknik açıdan blok zincir ulakları (Blockchain Oracles) ile akıllı sözleşme iletişiminin sağlanması konusuna değinilmiştir.

2. Blok Zincir Teknolojisi

Blok zincir teknolojisinin temelini dijital belgelerin zaman damgası ile korunması ve değiştirilmesinin önüne geçilmesi amacıyla geriye dönük olarak verinin takibinin yapılabileceği zincir yöntemi oluşturmaktadır (Haber ve Stornetta, 1991). Daha sonra güvenliğin arttırılması amacıyla yönteme Merkle ağaçları eklenmiştir (Merkle, 1987). Blok zincir, temel olarak bilgilerin blok adı verilen bilgi paketlerinde tutulması ve bu paketlerin kriptolojik özetinin bir sonraki blok üzerinde tutulmasıdır. Böylelikle herhangi bir blok üzerinde veri değişmesi durumunda kendisinden sonra gelen bloklarda bulunan özet bilgisinin uyuşmamasından dolayı verinin değiştirildiği kolaylıkla tespit edilebilmektedir (Şekil 1). Bu teknoloji Bitcoin dijital paranın ortaya çıkışına kadar aktif bir şekilde kullanılmamıştır (Nakamoto, 2008).



Şekil 1. Blok zincir yapısı (Figure 1. Blockchain structure) (Nakamato, 2008)

3. Akıllı Sözleşme

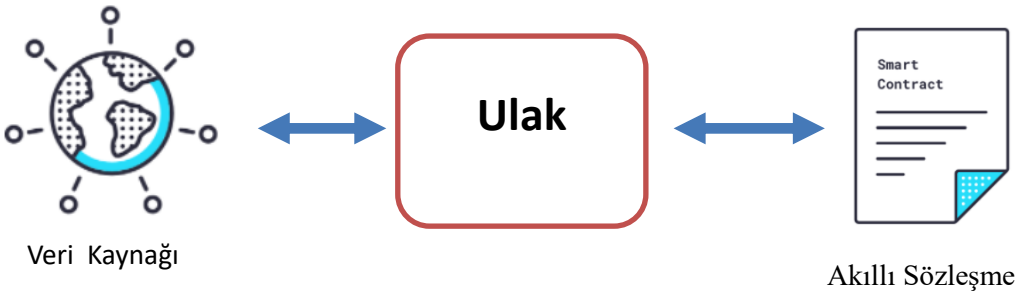
Blok zincir teknolojisi bilgi saklama amaçlı gerçekleştirilmiştir. Bitcoin ile veri saklanırken dijital bir varlığın sahipliğinin gerçekleştirilmesi ve takibi süreci gerçekleştirilmektedir. Her ne kadar Blok zincir teknolojisi veri saklanması ve doğrulanması yönüyle başarılı bir yöntem olsa da tasarım olarak tek görevi yerini getirmektedir. Bu durum gerçek hayat problemlerinin çözümünde yeterli olmamaktadır (Liu ve Fraser, 2018; Mittal, 2017). Gerçek hayat problemlerinde daha önceden belirtilmiş olan şartlara göre yine daha önceden belirlenmiş kurallara göre sonuçların elde edilmesi ihtiyacı bulunmaktadır. Bu problemin çözümü olarak ortaya çıkan ilk örnek Ethereum projesidir. Ethereum projesi blok zincir üzerinde yerleştirdiği Ethereum Virtual Machine (EVM)

içerisinde kod betiklerinin çalıştırılmasını sağlamıştır. EVM üzerinde çalıştırılan kod betikleri Solidity programlama ile kodlanmaktadır. EVM üzerinde gerçekleştirilen işlemlerden kaynaklı ihtiyaç duyulan işlemsel ve depolama süreçleri için GAS adı verilen işlem ücreti alınmaktadır (Clack, Bakshi ve Braine, 2016; Mittal, 2017).

Akıllı sözleşmelerin kullanımının artması sonucunda merkezi olmayan uygulamalar (Decentralized Applications - DApps) ortaya çıkmıştır (Taş ve Tanrıöver; 2019). Merkezi olmayan uygulamalar merkezi işlem birimine ihtiyaç duymayan uygulamalardır. Akıllı sözleşmelerin yoğunlukla kullanıldığı alanlar; tedarik zinciri, sağlık, finans, sigortacılık ve perakendedir (Allison, 2016; Jamil, Hang, Kim ve Kim, 2019; Sharma, 2018; Aylak, 2022). Özellikle Endüstri 4.0 ve beraberinde getirdiği teknolojik ihtiyaçlar akıllı sözleşmelerin kullanımını desteklemektedir (Dikilitaş, Toka ve Sayar, 2021).

4. Blok Zincir Ulakları (Blockchain Oracles)

Blok zincir teknolojisi ve uygulamaları, veri güvenliği kapsamında incelenmiş ve birçok çalışma ile desteklenmiştir (Ramachandran ve Kantarcioglu, 2017; Azaria, Ekblaw, Vieira ve Lippman, 2016). Ancak gerçek hayat örnekleri açısından bakıldığında bir ürünün üretildiği yerden tedarik zincirine dâhil olması ve son kullanıcıya kadar olan süreçte ürünün takibi birden fazla verinin blok zincir üzerinde kayıt işlemini gerektirmektedir (Mik, 2017). Blok zincir teknolojisi ve üzerinde çalışan kontratlar, mimari gereği blok zincir ağının dışından veri alamamaktadır. Ancak gerçek hayat problemlerini çözen yazılımların birçoğu dışarıdan veri almaya ihtiyaç duymaktadır. Örnek olarak IOT kaynağından gelecek bir parametrik verinin blok zincir üzerine aktarılması bir problemdir. Verinin blok zincir üzerine kayıt işlemi bir aracıya ihtiyaç duyularak gerçekleştirilmektedir.



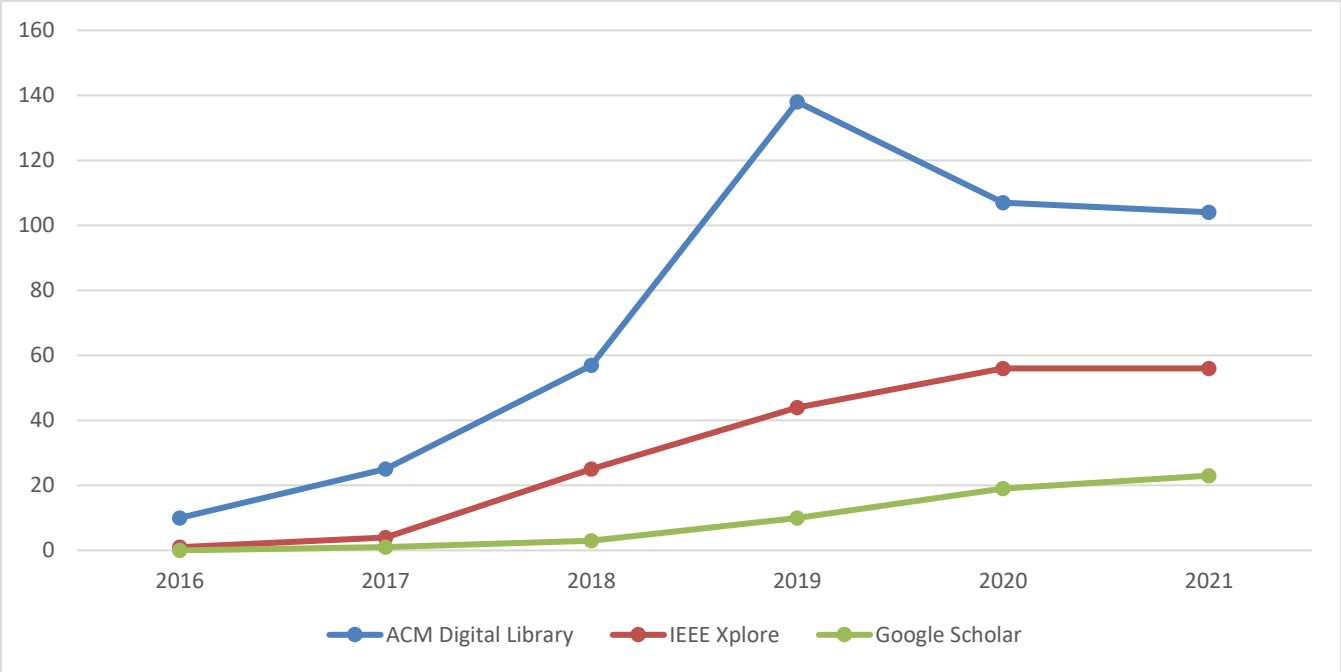
Şekil 2. Ulak veri akış süreci (Figure 2. Blockchain oracle data flow) (Cryptopedia, 2022)

Blok zincir dışından veri alma ihtiyacı blok zincir ulakları (Blockchain Oracles) tarafından giderilmektedir (Curran, 2018). Blok zincir ulakları, blok zincir teknolojisinin üzerinde yeni çalışılmaya başlanmış bir konudur. Tablo 1'de belirlenmiş arama yöntemine göre elde edilmiş sonuçlar Şekil 3'de yıllara ve veri tabanına göre İngilizce kaynaklarda bu alanda yapılmış çalışmalar aşağıda belirtilmiş sorgu ile sorgulanmıştır. Sonuçlardan da anlaşılacağı üzere giderek ilgi alan bir konu olduğu görülmektedir. Aynı sorgu Türkçe için gerçekleştirildiğinde herhangi bir sonuç elde edilmemektedir.

(("blockchain" OR "block chain" OR "block-chain") AND ("oracles" OR "oracle" OR "middle-ware" OR "middleware" OR "middle ware" OR "datafeed" OR "data feed" OR "data-feed")))

Tablo 1. Veri tabanlarına göre sorgu sonuçları (Table 1. Query results by databases)

Veri Tabanı	Filtre	Sonuç sayısı
ACM Digital Library	Başlık, Özet, Anahtar kelimeler	441
IEEE Xplore	Başlık, Özet, Anahtar kelimeler	186
Google Scholar	Başlık	56
Toplam		**683**



Şekil 3. Yıllara göre veri tabanı sonuçları (Figure 3. Database results by years)

Ulaklar, blok zincir ve blok zincir dışı ortam arasında köprü durumundadır. Ulaklar, veri kaynağının kendisi değildir (Şekil 2). Ulaklar ağ dışı verileri akıllı kontratlara sunar. Ancak kimi ulaklar dış kaynaklara veri gönderme yeteneğine de sahiptir (Dale, 2019). Bu nedenle çeşitli türlerde ulaklar bulunmaktadır. Ulaklar veri kaynaklarının türüne, veri aktarım yönüne ve merkezilik özelliklerine göre çeşitlenmektedir. Ulakların veri kaynağı yazılım, donanım veya insan olabilir. Genellikle veri kaynağı olarak yazılım ve donanım kaynakları kullanılmaktadır. Yazılım kaynakları çeşitli yazılım, veri tabanı, borsa ve uçuş bilgileri olabilmektedir. Donanım kaynakları ağırlıklı olarak IoT, RFID ve sensörlerdir. Veri, blok zincir ağının dışından akıllı kontrata doğru ilerliyorsa *gelen ulak* eğer dışarı yönlü veri akışı var ise *giden ulak* türü üzerinde işlem görür (Curran, 2018; Beniiche, 2020). Merkezi niteliğe sahip ulaklar, tek bir veri sağlar ve tek kaynak ile iletişim halindedir. Tek nokta ile iletişimde halinde olmak riskli bir durumdur. Tek noktanın işlem yapamaması durumunda veri akışında kesinti gerçekleşecektir. Merkezi olmayan ulaklar, veri kaynağı olarak birden fazla noktayı kullanmaktadır ve gelen verinin doğruluğu, tutarlılığı kontrol edilebilmekte ve olası bir veri kesintisine daha dayanıklı olabilmektedir.

Blok zincir ulakları gerçek hayat problemleri için önemli olmasından dolayı, bu alanda akademik çalışmalara yanı sıra ticari girişimler de ortaya çıkmıştır. Bu alanda ortaya çıkan girişimler şunlardır:

Provable.xyz: 2018 yılında Oraclize.it ismi ile kurulmuştur. Daha sonra ismini değiştirmiştir. Ethereum, Rootstock, R3 Corda, Hyperledger Fabric ve EOS gibi platformlara hizmet vermektedir. Merkezi niteliğe sahiptir. Ancak istenildiğinde birden fazla veri kaynağı desteği vermektedir (Provable, 2018).

ChainLink: San Francisco merkezli olarak 2017 yılında kurulmuştur. Merkezi olmayan bir nitelikte çalışmaktadır. Aave, Synthetix ve yearn.finance tarafından desteklenmektedir (ChainLink, 2017).

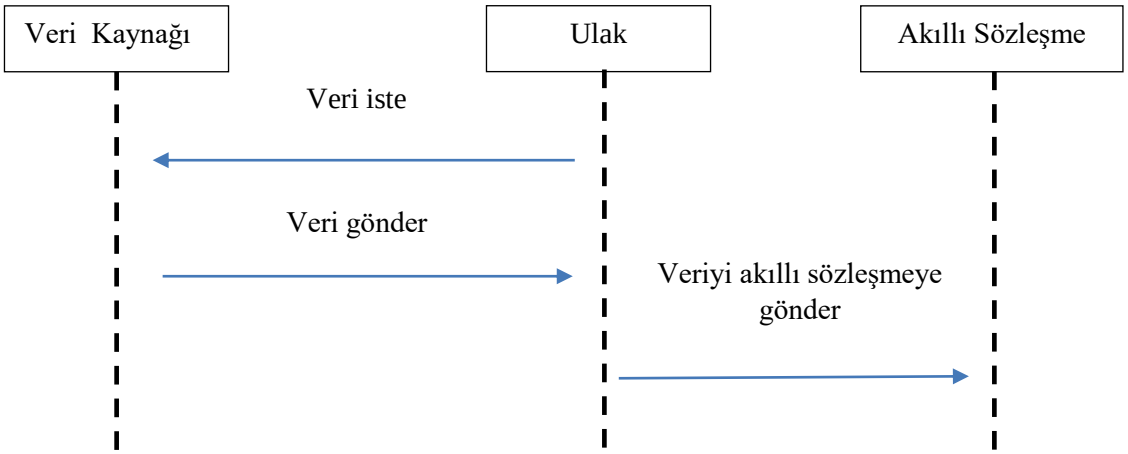
Tellor: Dijital para çiftlerinin zincir dışı veri noktaları ile iletişim sağlayan merkezi bir yapıya sahip olmayan ve 2020 yılında faaliyete başlayan bir platformdur (Tellor, 2018).

DIA: Merkezi olmayan blok zincirler arası işlem yapabilme kapasitesine sahip açık kaynak kodlu ve 2018 yılında faaliyete geçmiş bir platformdur. Ethereum, BSC, Polygon, xDaiChain, Astar, Avalanche, Aurora, Metis, Moonbeam, Fantom ve Near blok zincirlerini desteklemektedir (DIA, 2018).

API3: Proje 2020 yılında faaliyete geçmiştir. Blok zincir tabanlı merkezi yazılımların API'ler aracılığı ile veri elde etmesini sağlamayı amaçlamış bir platformdur. Fantom, Polygon, Tomochain gibi blok zincirleri desteklemektedir (API3, 2020).

5. Tasarım ve Uygulama

Blok zincir dışı verinin ulak aracılığı ile akıllı sözleşmeye aktarımı bir problemdir. Bu çalışmada aktarım probleminin temel seviyede tasarımının ortaya koyulması ve örnek bir olay üzerinden çalışma sisteminin ortaya koyulması gerekmektedir. Bu çalışmada Şekil 4'te veri akış sıra diyagramı ortaya koyulmuş olan bir çalışma yapılmıştır. Akış diyagramına göre ulak görevini üstlenen yazılım veri kaynağından veri istemektedir. Veri kaynağından elde edilen bilgi blok zincir üzerinde bulunan akıllı sözleşmeye gönderilmektedir. Akıllı sözleşme üzerinde saklanan bilgi, kendisi ile iletişime geçen diğer sözleşmelere veriyi aktarmaktadır.



Şekil 4. Ulak veri akış sıra diyagramı (Figure 4. Oracle data flow sequence diagram)

Yapılan çalışmada belirtilen veri akış diyagramı hayata geçirilmiştir. Seçili bir şehre ait olan hava durumu bilgisinin seçili web sitesinden alınmıştır. Şekil 5'te verilmiş olan Javascript kodlarından görüleceği üzere her 5 dakikada bir seçili web sitesine *gettemp* fonksiyonu aracılığı ile sorgu gönderilmekte ve seçili şehre ait hava sıcaklığı bilgisi getirilmektedir.

```
async function updatedata(city, temperature){
    let tx_builder = contract.methods.updateWeather(city, temperature);
    let encoded_tx = tx_builder.encodeABI();
    let transactionObject = {
        gas: 200000,
        data: encoded_tx,
        from: WALLET_ADDRESS,
        to: CONTRACT_ADDRESS,
    };
    let signed = await web3.eth.accounts.signTransaction(transactionObject, PRIVATE_KEY);
    let received = await web3.eth.sendSignedTransaction(signed.rawTransaction);
    return received["transactionHash"];
}
async function gettemp(sehir){
    var sonuc;
    await $.ajax({
        url: "https://www.hurriyet.com.tr/hava-durumu/"+sehir+"/",
        dataType: 'text',
        success: function(data) {
            var elements = $("<p>").html(data)[0].getElementsByClassName("weather-detail-
hightemp")[0].innerText;
            elements = elements.replace("°C", "")
            sonuc = elements;
        }
   });
   return sonuc;
};
$(document).ready(function()
{
    setInterval(function()
```

```
    {
        load().then( async function(){
            sehir = "siirt";
            var temperature = await gettemp(sehir);
            var transactionHash = await updatedata(sehir, temperature)
            updateStatus(sehir+ " "+ temperature + " " + transactionHash);
        })
    }, 30000);
})
```

Şekil 5. Ulak yazılımının kodları (Figure 5. Oracle application codes)

Elde edilen bilgi *updatedata* fonksiyonu aracılığı ile akıllı sözleşmeye gönderilmiştir. Updatedata fonksiyonun 73 nolu satırında görüleceği üzere akıllı sözleşme üzerinde bulunan *updateWeather* fonksiyonu ile iletişime geçilmektedir.

```
pragma solidity 0.8.12;
contract WeatherOracle {
  address public oracleAddress;

  constructor (address _oracleAddress) {
    oracleAddress = _oracleAddress;
  }
  struct Datas {
    string city;
    string temperature;
  }
  mapping (bytes32 => Datas) private datas;
  event WeatherUpdate (
    string city,
    string temperature
  );
    function stringToBytes32(string memory source) private pure returns (bytes32 result) {
        bytes memory tempEmptyStringTest = bytes(source);
        if (tempEmptyStringTest.length == 0) {
            return 0x0;
        }
        assembly {
                result := mload(add(source, 32))
        }
    }
  function updateWeather (
    string memory _citycode,
    string memory _city,
    string memory _temperature
  )
  public
  {
    require(msg.sender == oracleAddress);
    datas[stringToBytes32(_citycode)].city = _city;
```

```
    datas[stringToBytes32(_citycode)].temperature = _temperature;

    emit WeatherUpdate (
      _city,
      _temperature
    );
  }
  function getData(string memory _citycode) public view returns (string memory, string
memory){
    return (datas[stringToBytes32(_citycode)].city,
datas[stringToBytes32(_citycode)].temperature);
  }
}
```

Şekil 6. Akıllı sözleşme kodları (Figure 6. Smart contract codes)

Şekil 6'te akıllı sözleşme kodları görülebilir. Akıllı sözleşmede verilerin kaydedilmesi için *updateWeather* fonksiyonu oluşturulmuştur. Bu fonksiyon başka kontratlar tarafından *getData* fonksiyonu ile çağrılabilmektedir.

6. Sonuç

Gelişen teknoloji hayatın ihtiyaç duyulan tüm alanlarında çözüm olabilecek şekilde yer edinmektedir. Blok zincir teknolojisinin günlük hayatta bulunan problemlerin çözümü amacıyla kullanımı gittikçe artmaktadır. Bu nedenle bireylerin ve işletmelerin ihtiyaçlarına yönelik geliştiren yazılımlara blok zincir teknolojisinin entegrasyonu da artmaktadır. Tedarik zinciri, enerji, finans, sağlık vb. alanlarda blok zincir teknolojisinin kullanımının temel gerekliliklerinden biri süreçlere ait verilerin blok zincir üzerinde aktarılmasıdır. Bu noktada ulak yazılımlarının varlığı blok zincir temelinde geliştirilen projelerin kaçınılmaz bir öğesidir. Ulak yazılımlarının gerekliliği ve mimarisinin anlaşılması bu alanda çalışma yapan araştırmacıların göz ardı edemeyeceği bir olgudur. Blok zincir teknolojisi ve akıllı sözleşme ile ilgili Türkçe yapılan çalışmalarda genellikle zincir dışından gelen verilerin araştırmacılar tarafından sadece akış şeması üzerinde bir ok olarak göstermektedir. Bu çalışmada, ulak yazılımlarının veri süreçleri açıklanmıştır. Bu çalışma aracılığı ile blok zincir alanında ulak yazılımları ile literatür eksikliğinin gidereceği öngörülmektedir. Ayrıca gerçekleştirilen uygulama ile pratikte bir ulak yazılımının gerçekleştirilme adımları gösterilmiştir. Ancak bu çalışmada blok zincir ulaklarının dışarıdan içeriye yönlü ve tek veri kaynağından parametrik verinin gelecek şekilde tasarım gerçekleştirilmiştir. Bu çalışmada hayata geçirilen uygulama her nekadar blok zincir ulaklarının tüm çalışma türlerinin kapsamasa dahi bu alanda çalışma yapacak araştırmacılara yol göstereceği düşünülmektedir.

Ek Bilgi: Çalışmada kullanılan kodlar https://github.com/digifelis/oracle adresinde paylaşılmıştır.

Kaynakça

Allison, I. (2016). Skuchain: Here's how blockchain will save global trade a trillion dollars. *International Business Times*, 1–5. https://www.ibtimes.co.uk/skuchain-heres-how-blockchain-will-save-global-trade-trillion-dollars-1540618. Erişim tarihi: 28.12.2021

API3 (2020) API3, https://api3.org/. Erişim tarihi: 28.02.2022

Aylak, B. L. (2022). The Effects of the Applications of Blockchain Technology on the Logistics sector . Avrupa Bilim ve Teknoloji Dergisi , Ejosat Özel Sayı 2022 (ICAENS) , 148-152 . DOI: 10.31590/ejosat.1077800

Azaria, A., Ekblaw, A., Vieira, T., & Lippman, A. (2016, August). Medrec: Using blockchain for medical data access and permission management. In *2016 2nd international conference on open and big data (OBD)* (pp. 25-30). IEEE.

Beniiche, A. (2020). A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*.

ChainLink (2017). ChainLink, https://chain.link/. Erişim tarihi: 28.02.2022

Clack, C. D., Bakshi, V. A. ve Braine, L. (2016). Smart contract templates: foundations, design landscape and research directions. *arXiv preprint arXiv:1608.00771*.

Cryptopedia (2022), What Is Chainlink in 5 Minutes. https://www.gemini.com/cryptopedia/what-is-chainlink-and-how-does-it-work. Erişim tarihi: 27.02.2022

Curran, B. (2018) What Are Oracles? Smart Contracts, Chainlink & "The Oracle Problem. https://blockonomi.com/oracles-guide. Erişim tarihi:27.02.2022

Dale, O. (2019) What Is Chainlink? Guide to The Decentralized Oracle Network. https://blockonomi.com/chainlink-guide/. Erişim tarihi:27.02.2022

Dia (2018). Dia, https://www.diadata.org/. Erişim tarihi: 28.02.2022

Dikilitaş, Y. , Toka, K. O. & Sayar, A. (2021). Current Research Areas in Blockchain . Avrupa Bilim ve Teknoloji Dergisi , Ejosat Özel Sayı 2021 (HORA) , 488-492 . DOI: 10.31590/ejosat.977320

Haber, S. ve Stornetta, W. S. (1991). How to time-stamp a digital document. *Journal of Cryptology*, *3*(2), 99–111. doi:10.1007/BF00196791

Jamil, F., Hang, L., Kim, K. ve Kim, D. (2019). A Novel Medical Blockchain Model for Drug Supply Chain Integrity Management in a Smart Hospital. *Electronics* . doi:10.3390/electronics8050505

Liu, M. ve Fraser, J. (2018). *Origin White Paper*. https://www.originprotocol.com/en/whitepaper. Erişim tarihi: 23.12.2021

Merkle, R. C. (1987). A Digital Signature Based on a Conventional Encryption Function. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *293 LNCS*, 369–378. doi:10.1007/3-540-48184-2_32

Mik, E. (2017). Smart contracts: terminology, technical limitations and real world complexity. *Law, Innovation and Technology*, *9*(2), 269-300.

Mittal, V. (2017). A survey of Blockchain Technologies for Open Innovation. *World Open Innovation Conference 2017*, (November), 1–27.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 21260.

Provable (2018). Provable, https://provable.xyz/. Erişim tarihi: 28.02.2022

Ramachandran, A., & Kantarcioglu, D. (2017). Using blockchain and smart contracts for secure data provenance management. *arXiv preprint arXiv:1709.10000*.

Sharma, A. (2018). 5 Trends Shows How Blockchain Is Changing Social Media | Hacker Noon. *Hackernoon*. 24 Ağustos. https://hackernoon.com/5-trends-shows-how-blockchain-is-changing-social-media-ba50c975c041. Erişim tarihi: 26.11.2021

Szabo, N. (1994). *Micropayments and Mental Transaction Costs*. 14

Taş, R & Tanrıöver , Ö. Ö., "Building A Decentralized Application on the Ethereum Blockchain," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), 2019, pp. 1-4, doi: 10.1109/ISMSIT.2019.8932806.

Tellor (2020). Tellor, https://tellor.io/. Erişim tarihi: 28.02.2022