Bilgisayar Bilimleri ve Mühendisliği Dergisi

Cilt-18
Sayı-2
2025

Çok Boyutlu Yazılım Geliştirme Verimlilik Değerlendirme Modeli Önerisi ve Uzaktan Çalışma Ortamlarında Kullanım Örnekleri

Multi-Dimensional Software Development Efficiency Assessment Model Proposal and Its Evaluation in Remote Work Environments

Haluk ALTUNEL
Bilkent Üniversitesi
Bilgisayar Mühendisliği
Ankara Türkiye
altunel@bilkent.edu.tr
ORCID: 0000-0003-1103-3644

Öz

Geleneksel yazılım geliştirme projelerinde verimlilik, yalnızca girdi-çıktı oranı üzerinden değerlendirilen, çoğu zaman tek boyutlu bir yaklaşımla ele alınmaktadır. Bu bakış açısı, kod kalitesi, bakım yükü ve müşteri memnuniyeti gibi kritik unsurları göz ardı ederek, verimliliğin çok boyutlu doğasını tam olarak yansıtamamaktadır. Bu çalışmada, yazılım geliştirme süreçlerindeki verimliliği somut verilerle ölçülebilir hale getirmeyi amaçlayan, dokuz temel boyuttan oluşan kapsamlı bir model önerilmiştir.

Modelin uygulanabilirliğini ve geçerliliğini test etmek amacıyla, üç farklı kurumda üçer yıllık döngüleri kapsayan bir örnek olay çalışması gerçekleştirilmiştir. Elde edilen bulgular, önerilen modelin gerçek hayattaki yazılım geliştirme süreçleriyle uyumlu olduğunu ve kurumlardaki gözlemlerle tutarlı sonuçlar üretebildiğini göstermektedir. Bu yönüyle, önerilen model, literatürdeki tek boyutlu verimlilik yaklaşımlarından önemli ölçüde ayrışmaktadır. Verimlilik sonuçlarının, kurum kültürü, takım yapısı, kullanılan teknolojiler ve teknik borç gibi bağlamsal faktörlerden etkilendiği görülmüştür. Bu nedenle, modelin sağladığı değerlerin her kuruma özgü olduğu ve doğrudan kurumlar arası karşılaştırma yerine, kurumlardaki zaman içindeki değişimin incelenmesinin daha anlamlı sonuçlar vereceği vurgulanmıştır. Önerilen model, yapay zekanın yazılım geliştirmeye entegrasyonu ve farklı iş modellerinin verimliliğe olan etkileri gibi güncel konuların incelenmesi için bir temel sunmaktadır. Bu çalışma, gelecekteki araştırmalar için verimlilik analizlerinde bir zemin hazırlamakta ve yazılım verimliliği alanında çok boyutlu ve bağlama duyarlı yaklaşımların önemini ortaya koymaktadır.

Anahtar sözcükler: Yazılım geliştirme, verimlilik, üretkenlik, yazılım metrikleri, yapay zeka, uzaktan çalışma

Abstract

In traditional software development, efficiency is often evaluated through a one-dimensional approach that focuses solely on the input-output ratio. This perspective overlooks critical factors such as code quality, maintenance burden, and customer satisfaction, failing to capture the multidimensional nature of efficiency. This study proposes a comprehensive, nine-dimensional model designed to make software development efficiency and productivity measurable using concrete data.

To test the model's applicability and validity, a case study was conducted across three different organizations,

Makale Bilgileri	**Article Info**
Türü: Derleme	Type: Survey
Geliş tarihi: 15.06.2025	Received date: 15.06.2025
Kabul tarihi: 30.10.2025	Accepted date: 30.10.2025

Atıf/ to Cite (IEEE): H. ALTUNEL: Çok Boyutlu Yazılım Geliştirme Verimlilik Değerlendirme Modeli Önerisi ve Uzaktan Çalışma Ortamlarında Kullanım Örnekleri : Bilgisayar Bilimleri ve Mühendisliği Dergisi, Cilt 18 2025 Sayı-2 1-12. Sf: 69-80, : **DOI: 10.54525/bbmd.1719996**

covering three-year periods within each. The findings demonstrate that the proposed model is consistent with real-world software development processes and produces results that align with observations within these organizations. In this regard, the model significantly departs from the one-dimensional efficiency approaches prevalent in existing literature.

The results indicate that efficiency is influenced by contextual factors such as company culture, team structure, technologies used, and technical debt. Therefore, it is emphasized that the values provided by the model are specific to each organization. Instead of direct comparison, analyzing changes over time within a single organization yields more meaningful results. The proposed model provides a foundation for examining current topics, such as the integration of artificial intelligence into software development and the impact of different business models on efficiency. This work establishes a basis for future research and highlights the importance of adopting a multidimensional, context-aware approach to software efficiency.

Keywords*:* Software development, efficiency, productivity, software metrics, artifical intelligence, remote working environment.

1. Giriş

Yazılım geliştirme projeleri, temelinde üretim projeleridir ve genel olarak yeni bir ürün veya yeni fonksiyonaliteler ortaya çıkartmayı hedefler. Üretimin olduğu ortamlarda girdi ve çıktıdan söz edilebilir. Yazılım geliştirme sistemlerinde temel girdi insanın zekası ve emeğidir. Çıktı ise hedeflenen ürün veya buna bağlı fonksiyonalitedir. Üretim sistemi bakış açısı ile bu tür bir sistemin çıktısının girdisine oranına bakılarak en yalın hali ile verimlilik hesaplamak mümkündür. Bunun sonucu olarak, yazılım geliştirme dünyasında verimlilik ve üretkenlik ölçümleri genel olarak tek boyutlu olarak hesaplanmıştır. Öte yandan yazılım geliştirmedeki diğer çıktılar olan kodun kalitesi, canlıda çıkan hatalar, bakım yükü, kodun güvenliği ve müşteride yarattığı memnuniyet ise verimliliğin parçası olarak görülmemiş sadece verimliliğe etki eden faktörler olarak değerlendirilmiştir. Ayrıca yazılımı üreten insanların verimliliği ise yazılım geliştirme verimlilik değerlendirme modellerinin bir parçası olarak değerlendirilmemiştir. Halbuki yazılım geliştirmede verimlilik ve üretkenlik çok boyutludur ve farklı boyutları da dahil edilerek ele alınmalıdır.

Son yıllarda yazılım geliştirme ekosistemini önemli ölçüde etkileyen dış etkenler bulunmaktadır. Bunlardan ilki yapay zekanın yazılım geliştirmedeki kullanımının yaygınlaşmasıdır. Diğeri de pandeminin de etkisi ile uzaktan çalışmanın yaygın şekilde endüstride kullanılmaya başlamasıdır [1]. Her ikisi de yazılım geliştirme alışkanlıklarını değiştirmekte, verimliliğe ve üretkenliğe yeniden ve daha geniş bir açıdan bakmayı gerekli kılmaktadır. Dolayısı ile bu çalışmada yazılım geliştirme alanında çok boyutlu bir verimlilik değerlendirme modeli önerilmiştir. Toplam 9 boyut belirlenmiş ve bu boyutların ölçümleri için formüller ortaya konmuştur. Önerilen çok boyutlu verimlilik modeli 3 farklı kurumda uzaktan çalışma koşullarında denenmiş ve kullanılabildiği görülmüştür.

Makale temel olarak 4 bölüm halinde yapılandırılmıştır. Sıradaki bölümünde yazılım geliştirmede verimlilik ölçümüne dair literatürdeki çalışmalar özetlenmiştir. Üçüncü bölümde önerilen model ve bunun bileşenlerine yer verilmiştir. Dördüncü bölümde uygulamaya dair 3 örnek ele alınmış ve modelin uygulama sonuçları değerlendirilerek karşılaştırılmıştır. Çalışmanın sonuçları özetlenerek, çalışmanın gelişmesine dönük sonraki adımlara yer verilerek kapanış yapılmıştır.

2. Kaynak Taraması

2.1 Yazılım Geliştirme Verimlilik Değerlendirme Modelleri

Yazılım geliştirme alanındaki üretkenlik ve verimlilik ölçme ve değerlendirme modellerinin belirlenmesi için aşağıdaki araştırma sorularına yanıt aranmıştır:

AS (Araştırma Sorusu)-1: Yazılım geliştirmede hangi verimlilik modelleri ve bunlara bağlı hangi metrikler tanımlanmıştır?

AS (Araştırma Sorusu)-2: Yazılım geliştirmede hangi üretkenlik modelleri ve bunlara bağlı hangi metrikler tanımlanmıştır?

Bu sorulara literatürde verilen yanıtların takibi amacı ile sorgu komutları oluşturularak araştırma kütüphanelerinde tarama yapılmıştır.

Sorgu Komutu-1: "Software Development" AND "Productivity Model".

Sorgu Komutu-2: "Software Development" AND "Efficiency Model".

Web of Science (WoS) ana araştırma kütüphanesi olarak kullanılmıştır. Google Scholar da tamamlayıcı kütüphane olarak taramaya dahil edilmiştir. Taramalar Eylül 2025'de yapılarak sonuçları Tablo 1'de özetlenmiştir. Taramalarda güncel çalışmalara yer verilmesi ve de teknolojideki değişimin etkilerinin yansımasının görülebilmesi amacı ile 2020 ve sonrasındaki yayınlara dair sorgu sonuçları da Tablo 1'de yer almaktadır. Ayrıca odaklanılacak kümenin belirlenmesi amacı ile atıf alan yayınlara dair sayılar da Tablo 1'e eklenmiştir. Bu makalenin ana amacı sistematik literatür taraması olmadığı için tarama çalışması diğer araştırma kütüphanelerine genişletilmemiştir.

Çizelge-1: Sorgu Komutları Sonucu Listelenen Yayın Sayıları

	WoS	Google Scholar	WoS (2020 ve sonrası)	Google Scholar (2020 ve sonrası)	WoS (2020 ve sonrası olup atıf alan)	Google Scholar (2020 ve sonrası olup atıf alan)
Sorgu Komutu-1: "Software Development" AND "Productivity Model"	101	60	17	38	12	24
Sorgu Komutu-2: "Software Development" AND "Efficiency Model"	109	43	17	11	12	8

Çizelge-1'deki çalışmalardan 2020 ve sonrasında yayınlanan ve atıf alan çalışmalara odaklanılmıştır. Tek bir liste oluşturulması amacı ile ilk adım olarak Sorgu Komutu 1 ve 2'de ortak yer alan çalışmalar tekleştirilmiştir. İkinci adım olarak listede tüm çalışmalar, bilimsel titizlik ve konuyla ilgisi açısından gözden geçirilerek odaklanılacak çalışma sayısı 18'e indirilmiştir ve bu çalışmalar Tablo 2'de yer almaktadır.

Çizelge-2'de yer alan çalışmalardan 7 tanesi verimlilik modeli veya verimlilik ölçümü önermesinde bulunmuştur. Yazılım üretkenliğinin göstergesi olarak üretilen kod miktarı için Kod Satır Sayısı (KSS) ve üretilen fonksiyonalite için Fonksiyon Nokta (FN) ölçümleri önerilmiştir [2]. Bunlara ilave olarak toplam emek, üretilen Kullanım Durumu Noktaları (KDN) analiz edilerek üretkenlik ölçümü için bu ikisinin oranını olan Üretkenlik Faktörü (ÜF) ortaya konmuştur [3]. Çevik yöntemlerin kullanılması durumunda işlerin kıyaslaması için kullanılabilecek Göreceli Emek kavramı ortaya atılmıştır [4]. Fonksiyonel Boyut (FB) yazılımın fonksiyonel boyutu olarak üretkenliğin temel boyutu olarak öne çıkartılmıştır [5]. Açık kaynaklı projelerde ekosistemdeki katılımcıların, ürettiği kod deposu oluşturma, kod gönderme, sorun bildirme, yorum yapma gibi bilgilerin toplamını ele alan Açık Kaynak Yazılım Ekosistemi Toplam Üretkenliği (AKYETÜ) kavramı üretilmiştir ancak formüle edilmemiştir [6, 7]. Yazılım projelerindeki üretkenlik, üretim çıktısının emeğe oranı olarak önerilmiş ancak sadece bu açıdan değerlendirmenin yeterli olmayacağı, projedeki insanların, geliştirilen ürünün ve organizasyonun da etkisinin olduğu değerlendirilmiştir [8].

Üretkenliğe ve verimliğe etki eden faktörler farklı çalışmalara konu olmuş ve temel olarak yazılım geliştirme yöntemi, proje karmaşıklığı, ekip otonomisi, yazılımın geliştirildiği programlama dilleri, yazılımın mimarisi, DevOps altyapısı, iş alanı ekibin motivasyonu, yetkinliği ve tecrübe seviyesinin ön planda olduğu gösterilmiştir [9-17]. Son yıllarda yazılım geliştirmede yaygınlığı giderek artan yapay zeka kullanımının üretkenliğe etkileri ortaya konmuş ancak ayrı bir değerlendirme modeli önerilmemiştir [18, 19].

2.2 Verimlilik Değerlendirmenin Uzaktan Çalışma Ortamlarındaki Kullanım Örnekleri

Teknoloji alanındaki gelişmelerin sonucu olarak uzaktan çalışma fikri 2000'li yılların başında tartışılmaya başlanan bir kavram haline gelmiştir. İngiltere'de bu modelle çalışanların 1997-2002 yılları arasındaki deneyimleri derlendiğinde çalışma saatlerinin ve mekânsal esnekliğin getirdiği avantajların yanında stres seviyesinin artması ve sınırların bulanıklaşması sorunlarını beraberinde getirmesi sonucuna ulaşılmıştır [20]. Benzer avantajların yanında uzaktan çalışmanın sosyal ilişkiler, iş ortamındaki güven ve çalışanlar arası bilgi paylaşımı noktasında zorluklara sebep olduğu da ortaya konmuştur [21]. Uzaktan ekip yönetmenin farklı hazırlıklar gerektirdiği, çalışanlara daha sık ve veriye dayalı geri bildirim verilmesi gerekliliği de yine aynı çalışmada tespit edilmiştir.

Çizelge-2: Odaklanılan Yayınlar

Yayın Adı	Yayın Tipi (Dergi/Konferans)	Yayın Yılı	Atıf Sayısı
Enhanced framework for ensemble effort estimation by using recursive-based classification	D	2021	14
Towards an evidence-based theoretical framework on factors influencing the software development productivity	D	2020	22
Framework of Software Developers Engagement Antecedents and Productivity - A Review	K	2020	19
Low-Code Versus Code-Based Software Development: Which Wins the Productivity Game?	D	2022	31
A Cost Estimating Method for Agile Software Development	D	2021	16
Parametric Software Reliability Growth Model with Testing Effort : A Review	K	2021	6
The Measurement of the Software Ecosystem's Productivity with GitHub	K	2021	9
How to Evaluate the Productivity of Software Ecosystem: A Case Study in GitHub	D	2020	9
Placing Trust in Automated Software Development Processes	D	2022	1
Tool for Measuring Productivity in Software Development Teams	D	2021	6
Studying Task Processes for Improving Programmer Productivity	D	2021	13
Software Development Productivity Model: Validation through Expert Review	K	2021	4
Using COSMIC to measure functional size of software: a Systematic Literature Review	K	2022	4
A Process Model for Component-Based Model-Driven Software Development	D	2020	15
Evaluation of the implementation of a subset of ISO/IEC 29110 Software Implementation process in four teams of undergraduate students of Ecuador. An empirical software engineering experiment	D	2020	21
Factors influencing vendor organizations in the selection of DevOps for global software development: an exploratory study using a systematic literature review	D	2023	11
Hints for Generative AI Software Development	D	2024	7
Evaluation of the Code Generated By Large Language Models: The State of the Art	K	2025	2

Yazılım geliştirme alanı özelinde uzaktan çalışmaya dair araçlar ve altyapı, bilgi paylaşımı, uzaktan eşli programlama, büyük grup toplantıları, farkındalık ve sosyal etkileşim zorluklarının ön plana çıktığı tespit edilmiştir [22]. Uzaktan çalışan yazılım profesyonellerinin zihinsel iş yüklerinin yüksek olduğu ve bunun arka planında yeni teknolojileri öğrenme zorunluluğu ve zaman kısıtlarının öne çıktığı anlaşılmıştır [23]. Diğer yandan küçük takımların daha yüksek verimlilik sağladığını; takım üyelerinin motivasyonu, iletişimi ve yeniliğe destek gibi

başlıkların verimlilik üzerinde etkili olduğu sonucuna varılmıştır [24]. Üretimin tek boyutlu olmadığı, üretilen yazılımın kalitesi, güvenliği, performansı ve bakım yükü gibi fonksiyonel olmayan boyutların da üretkenlikle birlikte değerlendirilmesi gerekmektedir [25]. Çevik yöntemlerin ve bu yöntemlerle çalışmanın uygun şartlar ve takım yapısı oluşturulduğunda yazılım üretkenliği ve verimliliği üzerinde pozitif korelasyonla katkı yaptığı belirlenmiştir [26].

Pandemi dönemi ve sonrasında yazılım geliştirme alanında uzaktan çalışma ve buna bağlı etkiler daha çok ilgi çeker ve araştırılır hale gelmiştir. Nepal'de pandemi sürerken yapılan bir anket çalışmasında katılımcı olan teknoloji sektörü profesyonellerinin %55'i üretkenliklerinin düştüğünü ve bunun nedeninin zamanında geri bildirim alınamaması ve ev ortamında dikkat dağıtıcı unsurlar olarak öne çıkmıştır [27]. Polonya'da çoğunluğunu yazılım geliştirme alanında uzmanların oluşturduğu bilişim teknolojileri profesyonelleri arasında yapılan ankette uzaktan çalışmanın ofisten çalışmaya göre verimli olduğu fikrini tercih edenlerin oranı %70' den fazladır [28]. Aynı çalışmada ofisten çalışmanın özellikle yeni veya az tecrübeli çalışanların takım dinamiklerini anlama ve işi öğrenme yönünden fayda sağladığı belirlenmiştir. Uluslararası bir şirketin Avrupa ve Amerika'da çalışan yazılım geliştiricilerinin ürettiği kod verileri, ortak çalışma platformu verileri (Slack vb.), toplantı ve eposta verileri analiz edilerek, kod üretiminin arttığını, buna karşın çalışma süresinin ve toplantı sayısının da paralel şekilde arttığını göstermiştir [29]. Amerika'daki yazılım geliştirme alanı profesyonelleri ile yapılan anket ve görüşmelerin derlenmesi ile üretkenliğin ve yenilikçiliğin uzaktan çalışma ortamında arttığı, ancak bunun ön koşulunun güçlü bilgi paylaşımı altyapısının bulunması olduğu tespit edilmiştir [30]. 35 farklı ülke ve farklı kıtalardan katılımcıların dahil olduğu bir başka çalışmada yazılım geliştirmede teknik rollerde olan çalışanların üretkenliği artarken iş analizi, proje yönetimi gibi paydaş etkileşimi gerektiren rollerde çalışanların üretkenliklerinin azaldığı ve bu nedenle projelerin yaklaşık %47'sinin uzaktan çalışma nedeni ile geciktiği raporlanmıştır [31]. Proje yönetiminin uzaktan çalışma ortamında sanal platformlara taşınması ile çevik yöntemlerin bu duruma uyarlanması, sorumluluk bilincinin yerleşmesi için daha sık retrospektif yapılması ve verimliliği arttırmak için daha fazla otomasyon yapılması önerilmiştir [32]. Büyük bölümü Brezilya ve Hindistan'da uzaktan çalışan yazılım geliştiriciler ve test uzmanları ile yapılan bir araştırmaya göre iş birliği ve verimliliği arttırmanın iletişim kanallarının arttırarak ve sık sık hizalanma toplantıları yaparak mümkün olabileceği sonucuna ulaşılmıştır [33].

Farklı ülkelerde yapılan çalışmaların ortak noktası verimlilik modellerinden ziyade, iletişim ve iş birliği odağında ilerlemiş ve bunları destekleyecek toplantı vb. ritüeller ile dijital iş birliği platformlarının daha etkin kullanımı öne çıkartılmıştır. Öte yandan yazılım geliştiren ekiplerde verimliliğin ölçümü ve bu ölçümlerin değerlendirme modeli haline getirildiği ve uzaktan çalışan ekiplerde kullanımına dair örneklere rastlanmamıştır.

3. Çok Boyutlu Verimlilik Değerlendirme Modeli

Üretkenlik elde edilen çıktı miktarına odaklanır. Yazılım geliştirme alanında üretkenlik çıktı olarak ana ürüne eklenen özellikler veya müşteriye sunulan fonksiyonlar olduğu için birim zamanda üretilen fonksiyon veya özellik üzerinden ölçüm yapılır. Bunlara ek olarak müşterinin görmediği ancak ürünün/çözümün teknik olarak fonksiyonel olmayan ihtiyaçlarının karşılamasına dönük tüm başlıklar da üretkenliğin konusudur. Üretkenlik "ne kadar" üretildiği ile ilgilenir. Bu amaçla birim zamanda üretilen Fonksiyon Nokta (FP) Sayısı veya çevik yöntemlerle çalışan takımlarda Kullanıcı Hikayesi kullanılır. Bu çalışmada, çağlayan ve çevik her iki metodolojiyi tek çatı altında eritmek amacı ile Üretim Çıktısı kavramı kullanılacaktır. Üretim Çıktısı, belli bir zaman diliminde tamamlanan iş adedini temsil eder. Diğer taraftan Üretim Çıktısı olarak kod satır sayısını veya yazılımın teknik metriklerini ölçmeyi deneyen yaklaşımlar olsa da bunlar ana üretim hedefi olarak değil, üretimin yöntemi ve verimlilik faktörlerine dönük analizde kullanılmalıdır [8].

Verimlilik çıktının "nasıl" elde edildiğine odaklanır. Üretim verimliliği ise birim zamanda üretilen iş çıktısının harcanan efora oranıdır ve Formül 1.a ve 1.b'de yer almaktadır. Formül 1.a, çağlayan modeli kullanan yazılım geliştirme ekiplerine dönük hazırlanmıştır. Formülde üretilen fonksiyon nokta sayısının bunları üretmek için harcanan efora oranı verilmiştir ve birimi FP/Kişi-Gün'dür. Formül 1.b, çevik yöntemleri kullanan ekiplere yönelik hazırlanmıştır. Formülde üretilen kullanıcı hikayesi sayısının (User Story) bunları üretmek için harcanan efora oranı gösterilmiştir ve birimi US/Kişi-Gün'dür. Yazılım üretimi esas olarak insan emeği ile yapıldığı için temel girdi, insan eforu olarak alınmıştır. İnsan eforu, kişi-gün veya kişi-saat olarak ölçülür. Yazılım geliştirme projelerinde ister çağlayan ister çevik yöntemler kullanılsın, çalışanlar üzerinde çalıştıkları iş paketleri için efor girişi yaparak harcadıkları maliyetleri yapılan iş ile ilişkilendirirler. Bunu yaparken Jira, Asana, ClickUp ve bunlara benzer araçlar kullanılır. Dolayısı ile üretim verimliliği ölçümü için endüstrideki firmalarda veri elde edilmesi mümkündür.

$$Üretim\ Verimliliği = \frac{\sum Üretilen\ Fonksiyon\ Nokta\ Sayısı}{\sum Harcanan\ Efor} \quad (1.a)$$

$$Üretim\ Verimliliği = \frac{\sum Üretilen\ Kullanıcı\ Hikayesi}{\sum Harcanan\ Efor} \quad (1.b)$$

Yazılım geliştirme alanında üretimin gerçekleşmesi için oluşan insan eforu dışındaki maliyetin de bir başka bağımsız boyut olarak ele alınması gerekir. Maliyeti belirleyen unsurlar, dış firmalardan tedarik edilen ilave lisanslar, proje özelinde ihtiyaç duyulan donanımlar ve

varsa diğer ek maliyetlerden oluşur. Üretim Çıktısı başına yapılan harcama, maliyet odaklı verimlilik değerini verir ve Formül 2 ile gösterilmiştir. Buradaki üretim çıktısı, çağlayan çalışan ekiplerde Fonksiyon Nokta (FP) ve çevik çalışan ekiplerde Kullanıcı Hikayesi (US) olarak dikkate alınır. Formülün paydasında yer alan Toplam Üretim Maliyeti, para birimi ile gösterilir ve üretimin yapılması için gereken maliyetlerin toplamıdır. Bu maliyetin içinde satın alınan ek donanımlar, ek yazılım lisansları, dış firmalara yaptırılan fonksiyonlara dair maliyetler yer alır. Farklı zamanlarda oluşan maliyetlerin kıyaslamasında kullanılabilmesi için Net Bugünkü Değer (Net Present Value-NPV) kullanılmıştır.

$$Maliyet\ Verimliliği = \frac{\sum Üretim\ Çıktısı}{\sum Toplam\ Üretim\ Maliyeti} \quad (2)$$

Üretilen kod miktarının planlanan kod miktarına oranı, teknik odaklı verimlilik değerlendirmesi için kullanılabilir. Ancak bu noktada kullanılan yazılım dili, kütüphaneler ve daha önce hazırlanmış altyapıların üretilen kod satır sayısına doğrudan etkisi vardır. Bu nedenle bu alan ayrıca ölçüm için önerilmemiştir.

Yazılım geliştirme alanında yapılan üretimin standart ve tekrarlı olmaması, üretilen her bir çıktının farklı teknik karmaşıklık içermesi, ölçümlerde kullanılan Üretim Çıktısının ağırlıklandırılmasını ve kıyaslanabilir hale gelmesini gerektirmektedir. Bu amaçla farklı yaklaşımlar bulunmaktadır ve özellikle yazılım büyüklüğü tahminleme çalışmalarında yararlanılır [34]. Bunlardan COSMIC Fonksiyonel Nokta yaygın kullanımı ve ISO19761 ile standart haline dönüşmüş olması nedenleri ile tercih edilmiştir [35]. Bu yöntem, yazılım geliştirmenin analiz aşamasında belirlenen veri işlemleri, giriş/çıkış fonksiyonları, sorguları ve değişiklikleri esas alır.

Verimin çıktılara dayalı olması yeterli değildir. Diğer boyut olarak çalışan verimliliği de dahil edilmelidir. Bu amaçla uzaktan çalışanları takip amacı ile kullanılan yazılımlar kullanılır ve bunlar pandemi dönemi ve sonrasında oldukça popüler hale gelmiştir [36]. Çalışan verimliliğini takip edip ölçen Teramind, AktivTrack, Hubstaff, Berqun gibi ürünler pandemi ve sonrasındaki dönemde hızla yaygınlaşmıştır. Bu ürünlerin yardımı ile çalışanların bilgisayarda yürüttükleri aktiviteler takip edilebilmekte ve kategorize edilerek, verimli ve verimsiz olarak sınıflanabilmektedir. Dolayısı ile çalışanların günlerinin ne kadarını verimli olarak sınıflanan aktivitelerle geçirebildikleri tespit edilmektedir ve Formül 3'te gösterildiği üzere Çalışan Verimliliği olarak ölçülebilmektedir. Örneğin Word, Excel gibi uygulamalarda geçirilen süreler, eposta uygulamalarında geçirilen süreler genel olarak verimli sınıflanmaktadır. Ancak bu doğrudan üretim çıktısı oluşturmak için olmayabilir. Bu tür yazılımlar, çalışanların üretim çıktısı oluşturmak için kullandıkları zamanı ayırt edememektedir. Dolayısı ile Formül 3 ile ölçülen Çalışan Verimliliği, Formül 1 ile ölçülen Üretim Verimliliğinden bağımsız bir boyuttur.

$$Çalışan\ Verimliliği = \frac{\sum Verimli\ Aktivite\ Süresi}{\sum Toplam\ Aktivite\ Süresi}.100 \quad (3)$$

Yazılım geliştirme yapanlar özelinde geliştirilen koda dair farklı metrikleri, DevOps hatları üzerinden toplanarak yapılan kodlamanın verimliliği bir başka boyut olarak değerlendirebilir. Bu amaçla DevOps hatlarındaki aktivitelere ve Git vb. kod depolarına erişerek buradaki kayıtları analiz eden Haystack, Allstack, Pluralsight Flow Jellyfish, Oobeya gibi ürünler bulunmaktadır. Bu araçların ölçtüğü metrikler çeşitlilik gösterse de ortak payda olarak kodun verimliliği olarak nitelendirilebilecek bir metrik daha ölçülebilir. Bu metrik kodun yeni bir kod olup olmamasına bakarak, kodlama verimliliğine dair bir ölçüm yapılmasına imkan vermektedir. Yazılım geliştirme faaliyeti sonrasında kod deposuna eklenen, değişen veya silinen kodun kendi içindeki dağılımı üzerinden kodlama verimliliği belirlenebilir. Yeni kod ve başkasına ait kodun gözden geçirilip iyileştirilmesi verimli aktiviteler iken, eski kodların değişimini değerlendirme kısmı teknik olarak dikkat gerektirmektedir. Eski kod üzerinde yapılan değişiklik yeniden düzenleme sebepli olduğu gibi, çıkan hataların düzeltilmesi amacı ile yapılan düzenlemeleri de içerebilir. Bu nedenle verimli ve verimsiz ayrımı yapılması gereklidir ve Formül 4 ile ifade edilmiştir. Kodlama verimliliği formülündeki KSS: Kod Satır Sayısını ifade eder. Otomatik üretilen veya dış kütüphanelerden alınan kodlar KSS'ye dahil edilmez. Kodlama verimliliği ölçümü % değer olarak verilir. Bunun ayrımını hem süre hem de yapılan değişikliğin hatalarla ilişkisine bakarak belirlemek mümkündür.

$$Kodlama\ Verimliliği = \frac{\sum (Yeni\ KSS + Düzenlenen\ KSS)}{\sum Tüm\ KSS}.100 \quad (4)$$

Yazılım kalite boyutuna dair etkiyi değerlendirmek için kodun kurallara ve belirlenen standartlara uyumlu yazıldığını tespit etmek amacı ile kodun kurallara uyum oranı Formül 5'te gösterildiği gibi % olarak hesaplanabilir. Bu amaçla statik kod analizi yapan Sonarqube, Codacy, PVC Studio türü araçlar kullanılabilir. Birim test yazılması ve buna dair kapsama oranının ölçülmesi de yaygın bir pratiktir, hatta DevOps hatlarında bir kalite kapısı haline dönüştürülebilir [37, 38]. Ancak birim test kapsama oranı tek başına kaliteye dair değerlendirme için yeterli olmayacaktır. Birim testin kodun iş mantığını içeren karmaşık bölümlerine yazılması ile daha basit standart bölümlerine yazılması arasında fark olacaktır. Dolayısı ile birim test kapsama oranının verimlilik değerlendirmesinde ayrı bir boyutu temsil etmesi mümkün değildir.

$$Kodlama\ Kurallarına\ Uyum = \frac{\sum Uyulan\ Kurallar}{\sum Tüm\ Kurallar}.100 \quad (5)$$

Yazılımın güvenlik boyutunu değerlendirmek için benzer şekilde kod güvenliği analizi yapılmalıdır, bu esnada kod içindeki güvenlik zafiyetleri değerlendirilir, bunlar yüksek, orta ve düşük olarak kategorize edilerek raporlanır. Kodlama güvenliği için Fortify, Checkmarx, Aikido Security

gibi araçları kullanarak OWASP tarafından yayınlanan Top10 listesini esas alan kodlama güvenlik zafiyetleri belirlenir [39]. Güvenlik boyutuna dair farklı ölçüm yöntemleri ve skorlarını sunan modeller bulunmakta ve bunlar, Sonarqube gibi ürünlerde yer almaktadır. Kodlama açısından güvenlik verimliliği Formül 6'da gösterilmiştir. Güvenlik zafiyet puanının hesaplanmasında yüksek, orta ve düşük güvenlik zafiyetleri sırasıyla 3, 2 ve 1 katsayıları ile ağırlıklandırılarak yansıtılır.

$$Güvenli\ Kodlama\ Verimliliği = 100 - \frac{\sum Güvenlik\ Zaafiyeti\ Puanı}{\sum Tüm\ KSS}.100 \quad (6)$$

Üretilen çıktının kalitesinin tespiti için testlerde çıkan hataların kod satır sayısına oranı ile hata yoğunluğu (Defect Density) kullanılır [40]. Hataların bağımsız test veya kalite ekipleri tarafından tespit edildiği ve bunların sistem testleri ve regresyon testleri esnasında tespit edilmeleri halinde, yazılım geliştirmede bağımsız bir boyut olarak değerlendirilebilir. Bu noktada hata yoğunluğu ayrı bir boyut olarak ele alınmalıdır ve Formül 7 ile gösterilmiştir.

$$Hata\ Verimiliği = 100 - \frac{\sum Hata\ Sayısı}{\sum Tüm\ Kod\ Satır\ Sayısı}.100 \quad (7)$$

Tüm geliştirme aşamaları tamamlanarak canlı ortama kurulan bir yazılım ürünün işletimine dair bakım eforu ve operasyonel maliyetler ile canlı ortamda ortaya çıkan olay kayıtları da verimlilik açısından önem taşır. Proje bazlı ve teslimat odağı ile uzaktan çalışan ekiplerde çoğunlukla bu boyut göz ardı edilse de özellikle verimlilik analizinde önem taşır. Uzaktan çalışma ile farklı coğrafyalarda yazılım üreten ekiplerin çoğunlukla canlı ortama kadar olan taahhütlerine odaklandıkları bir gerçektir. Çalışan ve testleri geçen yazılımın canlıya çıkışı sonrasında yaşananlara yeterince önem verilmediği, bakım için harcanan 3 yıllık eforun toplam proje eforunun %75-90 seviyesine kadar yükseldiği görülmektedir [41]. Bu noktada yazılım bakımı ve teknik işletim operasyonları için harcanan 3 yıllık ortalama efor ile toplam proje eforunun oranlanması ile bakım verimliliği hesaplanır ve Formül 8 ile ifade edilir. Bakım eforunda canlı ortamda oluşan olay kayıtlarının yarattığı efor da yer alır ve toplam bakım eforu bütünsel hesaplanmalıdır. Eğer canlıya geçerken teknik bir borç bırakıldı ise buna dair etki canlı ortamdaki bakım maliyetinde kendini gösterecektir.

$$Bakım\ Verimliliği = 100 - \frac{\sum Yazılım\ bakım\ ve\ teknik\ işletim\ eforu}{\sum Proje\ Geliştirme\ Eforu}.100 \quad (8)$$

Verimlilik değerlendirmesinin sağlıklı yapılabilmesi için farklı teknik boyutların yanında müşteri memnuniyeti boyutunun da eklenmesi gereklidir. Yapılan teknik ölçüm ve analizlerin müşteri beklentilerini karşılamadaki etkisi, müşteri memnuniyetine katkısı ve özellikle müşterinin tavsiye etme potansiyeli de ölçülerek tüm boyutlardaki değişimle ilişkilendirilmedir. Bu amaçla müşteriler nezdindeki durumu değerlendirmek amacı ile bağımsız ve periyodik anketler kullanılabilir. Anketler tasarlanırken müşteri memnuniyetini en net şekilde alacak sorular sorulmalı ve sonuçlar Likert Ölçeği ile hazırlanmış anketler kullanılarak müşteri memnuniyeti ölçülerek değerlendirilir [42]. Müşteri memnuniyeti % olarak hesaplanır ve Formül 9 ile gösterilir. Net tavsiye skoru da bu ölçümde müşteri memnuniyetine dair gösterge olması sebebi ile verimlilik analizi için ayrıca bir boyut olarak eklenmesine gerek bulunmamaktadır.

$$Müşteri\ Memnuniyeti = \frac{\sum Memnun\ Müşteri\ Sayısı}{\sum Tüm\ Müşteri\ Sayısı}.100 \quad (9)$$

Tüm bu 9 boyutu bir araya getirince Çok Boyutlu Yazılım Geliştirme Verimlilik Modeli oluşmaktadır. Her bir boyuta dair ölçüm farklı formüllere ve verilere dayanarak yapıldığı için boyutlar birbirinden bağımsızdır. Bu boyutlara dair ölçümlerin değerlendirilmesi için radar grafiği kullanılır, örneği Şekil 1'de yer almaktadır. Radar grafiğinde, düzenli ölçümlerle yapılan kurumlarda, boyutlardaki değişim ve boyutlar arası etkileşim daha net bir şekilde izlenebilir.

Yazılım geliştirme yaşam döngüsünde doğrudan ölçülen ve türev olarak elde edilen farklı metrikler de bulunmasına rağmen, yaygın kullanımı nedeniyle bu 9 boyut ve buna dair metrikler seçilmiştir. Modelin tüm boyutlarına dair ölçümler veya bu ölçümleri yapacak araçlar ve altyapılar bulunmayabilir. 9 boyutun 5 veya daha fazlası ölçülebildiğinde model, değerlendirmeye uygun bir çerçeve sunacaktır. Ölçülen boyutların zamana bağlı değişimi, yazılım geliştirmeye dair verimlilik değişimini değerlendirmek için gereklidir. Zamana bağlı değişimin dikkatlice irdelenmesi ile boyutlardaki değişimler ve bunların aralarındaki ilişkilerin incelenmesi mümkün olacaktır.

Şekil-1: Çok Boyutlu Yazılım Geliştirme Verimlilik Modeli Radar Grafiği.

4. Modelin Uygulanması ve Sonuçları

Farklı dönemlerde 3 farklı kurumda model kullanılmıştır. Kurumlar, en az 100 ve daha fazla yazılım geliştiricisi olan, ürün ve proje geliştiren firmalardır. Her bir kurumun ana işkolu teknoloji ürünlerinin geliştirilmesidir. Teknoloji ürünleri sadece yazılım odaklı olabildiği gibi donanım ve yazılımın bir arada olduğu ürün veya projeler de olabilir. Kurumlar, mahremiyetleri nedeni ile A, B ve C olarak

anılacaktır ve firma detaylarına yer verilmeyecektir. Her 3 kurum da Türkiye merkezli olup, yurtiçi ve yurtdışındaki müşterilerle çalışmaktadır. A ve B firmaları işletmeden işletmeye (B2B) alanda teknoloji geliştirirken, C firması işletmeden tüketiciye (B2C) alanda teknoloji geliştirmektedir. Her 3 kurumun çalıştığı alanlar ve hedef müşteri kitleleri birbirinden farklıdır.

4.1. Kurum A'da Model Uygulaması

Kurum A'da modelin kullanımı esnasında tam zamanlı uzaktan çalışma yürütülmüştür. Bu kurum genel olarak çevik yaklaşımları benimsemiştir ve proje bazlı çalışmaktadır. Projeler temelde müşteri talepleri ile şekillenmektedir. Çalışmada toplam 6 boyuta dair veri ölçümleri yapılmıştır. Bu ölçümler, uzaktan çalışmaya başlandığı dönem olan 2020 ile uzaktan çalışmanın devam ettiği 2022 arasında yapılmıştır. İlk ölçüm olan 2020 ve son ölçüm olan 2022 yıllarına dair verilerin olduğu radar görüntüsü Şekil 2'de yer almaktadır.

İlk ve son ölçüm değerleri kıyaslandığında üretim verimliliği ve maliyet verimliliğinde iyileşme olurken, kodlama kurallarına uyum ve güvenli kodlama verimliliği alanlarında gerileme olmuştur. Bu dört boyut birlikte değerlendirildiğinde Kurum A'daki üretim hızı artarken, kod kalitesi ve güvenliği alanlarında düşüş görülmektedir. Bakım verimliliği alanındaki değişim çok sınırlıdır. Müşteri memnuniyetinin ise belirgin şekilde arttığı gözlenmiştir. Model'e göre bu kurumun üretim hızını arttırarak müşteri memnuniyetine katkı sağladığı ancak kod kalitesi ve güvenliği alanlarında gerilemelere sebep olduğu söylenebilir. Bu sonuca etki edebilecek üretim dışı farklı etkenler de söz konusudur. Örneğin uzaktan çalışmaya uyumun artması, müşteri taleplerinin niteliği ve niceliği, çalışan devinimi, kurumun yer aldığı sektörün dinamikleri. Kurum özelinde tüm bu etkenleri sabit tutarak veri elde edilmesi mümkün olmadığı için, verimlilik metriklerinin zamana ve birbirlerine karşı değişimi gözlenmiştir. Değişimi tetikleyen dinamiklerin netleştirilmesi için daha steril koşullarda farklı bir çalışma yapılması gereklidir.

Şekil-2: Kurum A'nın ilk ölçüm olan 2020 değerleri ve son ölçüm olan 2022 değerleridir.

4.2. Kurum B'de Model Uygulaması

Kurum B'de de uzaktan çalışmanın tamamen uygulandığı 2021 ve 2023 dönemleri arasında çalışma yürütülmüştür. Bu kurum müşteri talepleri ile şekillenen projeler odaklı çalışmaktadır, eş zamanlı ürünleştirme çalışmaları da yürütülmekte olup bu alandaki çalışmalar sınırlıdır. Kurum hem çevik hem de çağlayan çalışma modellerini benimsemiştir. İlk ve son ölçüm yıllarına dair verilerin olduğu radar görüntüsü Şekil 3'te yer almaktadır. Toplam 8 boyutta ölçüm yapılmıştır. Güvenli Kodlama Verimliliği için ölçüm bu alandaki araçlardan herhangi bir tanesinin bulunmaması nedeni ile yapılamamıştır. Yıllık ölçümler yapılırken ilgili yılda gerçekleştirilen tüm projeler ve bu projeler için geliştirilen tüm kodlar dikkate alınmıştır. Bu nedenle yıllık veriler ilgili yılın tamamını içermektedir ve yıl sonunda toplanmıştır.

Şekil-3: Kurum B'nin ilk ölçüm olan 2021 değerleri ve son ölçüm olan 2023 değerleridir.

Kurum B için ilk ve son ölçüm verileri karşılaştırıldığında, üretim verimliliği ve maliyet verimliliğinin arttığı görülmektedir. Buna karşın çalışan verimliliğinin belirgin şekilde azaldığı görülmektedir. Kodlama verimliğinin sınırlı düştüğü, kodlama kurallarına uyumun ise daha belirgin gerilediği gözlenmektedir. Hata verimliliğinin düşerken, bakım verimliliği sınırlı oranda gerilemiştir. Müşteri memnuniyetinde kısmı bir artış görülmektedir. Modele göre Kurum B'de de yazılım üretimi hızı artmış ve üretim ve maliyet verimliliği yükselmiştir. Bu da müşteri memnuniyetine kısmı bir artış olarak yansımıştır. Kodlama verimliliği ve kodlama kurallarına uyum ise gerilemiştir. Ana sebebin üretim hızına dönük beklentinin kodlama üzerinde yarattığı baskı kaynaklı olduğu değerlendirilmektedir. Kurum B özelinde en belirgin değişim çalışan verimliliğindeki düşüştür. Çalışanların daha uzun sürelerde çalıştığı ve bunun üretim verimliliğine olumlu yansıdığı görülmüştür. Bunun arkasında projelerin teslim sürelerine yetiştirilmesi baskısı bulunmaktadır. Öte yandan bilgisayar başında geçen süre arttığı için dinlenebilmek amaçlı film izleme gibi farklı aktivitelerin de bilgisayarlarda yapılmaya başlandığı ve bunun da verimsiz aktivitelerin oranını önemli ölçüde arttırdığı ve çalışan verimliliği oranını negatif etkilediği tespit edilmiştir. Diğer yandan ölçülen 8 verimlilik metriği yanında üretim dışı

faktörlerin de etki edebilmesi söz konusudur. Deneysel bir ortam kurulamadığı için tüm dış faktörlerin sabit tutulması mümkün olmamıştır. Ayrıca verimlilik metriklerinin kendi aralarındaki korelasyonlarını ortaya çıkartmak için yeterli veri olmadığı için bu analize yer verilememiştir. Bu başlık daha sonraki çalışmaların konusu olarak planlanmıştır.

4.3. Kurum C'de Model Uygulaması

Kurum C'nin ölçümlerinin yapıldığı dönem 2022 ve 2024 yılları arasındadır. Kurum C, 2022'de tam zamanlı uzaktan çalışma yürütmüş, 2023 itibarı ile isteğe bağlı hibrit modele geçmiştir. Bu dönemde dileyen çalışan, uzaktan çalışabilirken, dileyen çalışan haftada 2 gün ofise giderek çalışmıştır. Melez çalışmayı tercih edenlere şirket mali yan haklarla destek olmuştur. Bunun sonucunda şirketin 17%'si melez çalışmayı tercih etmiş, kalan çoğunluk ise uzaktan çalışmayı sürdürmüştür. Şirket ürün bazlı yapılanmış, ürünlere yeni fonksiyonlar ekleyen veya teknolojik altyapısını yenileyen projeler geliştirmektedir. Ürünler, platform model ile yapılandırılmış web ve mobilden son müşteri tarafından doğrudan erişilebilir durumdadır. İki yıla dair verilerin olduğu radar görüntüsü Şekil-4'te yer almaktadır.

Kurum C'nin ölçümleri incelendiğinde ve ilk yıl ile son yıl karşılaştırıldığında üretim verimliliği ve maliyet verimliliğindeki artış oranı oldukça yüksektir. Çalışan verimliliği de önemli oranda artmıştır. Kodlama verimliliği de yükselirken, kod kurallarına uyumdaki iyileşme sınırlı kalmıştır. Bakım verimliliği artmış, müşteri memnuniyeti önemli oranda yükselmiştir. Ölçülen verimlilik boyutlarının tamamında iyileşme söz konusudur. Bu veriler diğer 2 kurumdan ayrışmaktadır. Kurum C'deki pozitif değişime daha yakından bakıldığında, 2023 yılında başlatılan platform yenileme çalışmalarının etkisi olduğu görülmektedir. Var olan ürün platformları, yeni teknolojilerle yenilenmeye başlanmış ve 2024 yılında büyük ölçüde bu dönüşüm tamamlanmıştır. Fonksiyonel olarak yeniliğin sınırlı miktarda eklendiği, bunun yerine öncelikli olarak teknik iyileştirmelerin yapıldığı bir ortamda çalışanlar, bildikleri bir ürünü yenilemişlerdir. Bu da çalışan verimliliği, kodlama verimliliği, kodlama kurallarına uyum alanlarında da iyileşme fırsatı vermiştir. Üründeki iyileşme müşteri memnuniyetine doğrudan yansımıştır. Bunun yanında dış etkenlerin, müşteri beklentilerinin ve sektörün değişiminin de sonuçlara etkisi olması muhtemeldir. Bu etkilerden izole edilerek veri elde edilmesi mümkün olmadığı için verimlilik metriklerinin kendi aralarındaki korelasyonlarını ortaya çıkartmak Kurum C için de mümkün olmamıştır.

Şekil-4: Kurum C'nin ilk ölçüm olan 2022 değerleri ve son ölçüm olan 2024 değerleridir.

Her üç kurumun da ölçüm değerleri Çizelge-3'te sunulmuştur. Bu ölçümler, analiz dönemi boyunca (2025 yılı itibarıyla devam eden) uzaktan veya karma çalışma modelleri bağlamında gerçekleştirilmiştir. Bu nedenle, elde edilen değerler, söz konusu çalışma düzenlerinin gerçek dünyadaki temsiliyetini yansıtmaktadır. Ancak, kurumların kendi içinde ölçülen verimlilik değerlerinin değişiminde, tek başına uzaktan veya karma çalışma modelinin belirleyici tek etken olmadığı kabul edilmelidir. Daha önce bahsedilen diğer dış etkenlerin de doğrudan veya dolaylı etkileri bulunmaktadır. Bu çok değişkenli etkinin analiz edilmesi, sonuçların iç geçerliliğini tam olarak tesis etmek için ayrı bir çalışma gerektirmektedir. Bu amaçla, bulguların geçerliliğini artırmak için iki adımlı bir yaklaşım planlanmaktadır. İlk adım olarak, aynı kurum içindeki benzer özelliklere sahip, ancak farklı çalışma modelleri (uzaktan/karma) uygulayan ekiplerin verimlilik değişimlerini inceleyerek kontrollü bir kıyaslama imkanı sağlanacaktır. İkinci adımda ise, tam zamanlı ofisten çalışan ekiplerdeki verimlilik ölçümleri zamana bağlı olarak ayrı bir çalışmayla analiz edilecektir. Bu karşılaştırmalı analiz, ofisten çalışmanın getirdiği iletişim ve fiziksel etkileşim farkının verimlilik boyutlarına olan özel etkisini izole ederek, mevcut bulgularımızın temsil gücünü ve geçerliliğini daha sağlam temellere oturtacaktır.

Çizelge-3: Kurumlar için 9 Boyuta dair İlk ve Son Ölçüm Değerleri

	1.Üretim Verimliliği (FP/ Kişi-gün)	2.Maliyet Verimliliği (FP/ 10 Bin TL)	3.Çalışan Verimliliği (%)	4.Kodlama Verimliliği (%)	5.Kodlama Kurallarına Uyum (%)	6.Güvenli Kodlama Verimliliği (%)	7.Hata Verimliliği (%)	8.Bakım Verimliliği (%)	9.Müşteri Memnuniyeti (%)
Kurum A (İlk Ölçüm)	0.9	1.4			95%	81%		20%	75%
Kurum A (Son Ölçüm)	1.1	1.7			92%	79%		21%	82%
Kurum B (İlk Ölçüm)	0.6	0.9	67%	75%	92%		67%	19%	62%
Kurum B (Son Ölçüm)	0.7	1.1	59%	74%	89%		65%	18%	65%
Kurum C (İlk Ölçüm)	0.4	1.3	82%	72%	93%			17%	47%
Kurum C (Son Ölçüm)	0.7	1.2	88%	90%	95%			22%	61%

Verimlilik boyutlarının kendi aralarındaki ilişkisini incelemek için daha geniş bir veri setine ihtiyaç duyulmaktadır. Diğer yandan bu üç kurumda tüm boyutların hepsinin aynı anda ölçülebilmesi mümkün olmamıştır. Özel yazılım ürünü gerektiren güvenli kodlama verimliliği ve çalışan verimliliği gibi alanlarında tüm kurumlarda veri bulunmamaktadır. Verilerin toplanabileceği araçların yaygınlaşma hızı da bu ölçümlerin sağlıklı yapılması için anahtar konumundadır. Bu nedenlerle istatistiksel olarak anlamlı ilişkiler tespit edebilmek amacı ile verimlilik boyutları arasındaki korelasyon, daha yaygın ölçüm imkanı olan boyutlardan başlanarak ayrı bir çalışmada ele alınacaktır.

Ölçüm sonuçları genel olarak incelendiğinde, modelin dokuz boyutundan ölçülebilenlerin olduğu ve bu ölçümlerin yıllar içindeki değişiminin güvenilir bir şekilde takip edilebildiği görülmüştür. Bu durum, modelin sektörde yaygın olarak kullanılan mevcut araçlar ve altyapılardan elde edilen ham verilerle uyumlu olduğunu ve pratikte uygulanabilirliğini göstermektedir. Ayrıca, verimliliğin tek bir boyutla incelenemeyeceği; farklı boyutlarla ele alınmasının ve zamana bağlı değişiminin değerlendirilmesinin daha sağlıklı olacağı sonucu pekişmiştir.

Bu çok boyutlu verimlilik modeli, kurumlardaki yetkililere sadece gözlem yapma imkanı sunmakla kalmaz, aynı zamanda somut karar verme noktalarında da kritik bir temel sağlar. Ölçülen verimlilik değerleri, her ne kadar kuruma özgü olsa da zaman içindeki kendi değişimleri üzerinden referans değerler oluşturur. Örneğin, bir boyutun trendinde gözlemlenen belirgin bir düşüş, yöneticilere "ekibe ek kaynak alımı", "eğitim/mentorluk ihtiyacı" veya "teknik borcu temizleme" gibi konularda proaktif kararlar almaları için kanıt sunar. Bu sürekli izleme, kararların deneye dayalı olmasına olanak tanır. Bu ölçme yöntemini uygulamak, yazılım geliştiren kurum için elbette bir maliyet getirecektir. Ancak bu maliyet, elde edilecek somut faydalarla dengelenmektedir:

Risk Azaltma ve Erken Uyarı: Verimlilik boyutlarındaki anormal değişimler, potansiyel proje gecikmeleri veya kalite sorunları için erken uyarı görevi görerek, yüksek maliyetli kriz müdahalelerinin önüne geçer.

Yatırımın Geri Dönüşü (ROI): Yönetim, bu verileri kullanarak yeni bir araç, teknoloji adaptasyonu veya ekibe yeni bir çalışan alımı gibi yatırımların verimlilik boyutlarına etkisini somut olarak ölçebilir ve böylece kaynakların doğru yerlere yönlendirilip yönlendirilmediğini değerlendirebilir.

Sonuç olarak, verimlilik boyutlarında çıkan değerler kuruma özgü olmakla birlikte, bu değerlerin zamana bağlı değişiminin analizi, kurumun içinde bulunduğu bağlamdaki (teknolojik borç, ekip devinimi, kültürel adaptasyon) değişimin sonuçlara etkisini de ortaya koyar ve stratejik yönetim kararlarını destekler.

5. Sonuç

Bu çalışma ile yazılım geliştirme verimlilik değerlendirme modeli 9 temel boyut ile somut veriye dayalı şekilde ölçülebilir şekilde önerilmiştir. Modelin uygulaması 3 farklı kurumda, 3'er yıllık döngüleri içine alacak şekilde uygulanmış, ölçülebilen boyutlar ile gerçek hayattaki karşılıkları kıyaslanmıştır. Buna göre modelin uygulanabilir olduğu ve sonuçlarının kurumlardaki tespitlerle uyumlu olduğu görülmüştür. Çok boyutlu verimlilik değerlendirme modeli literatürdeki tek boyutlu diğer çalışmalardan ayrışmaktadır.

Modelin daha fazla kurumda kullanılması ile farklı iş modelleri, farklı sektörler, farklı ülkelerdeki uygulama sonuçlarının benzerlik ve farklılıkları değerlendirilecektir. Bu ilerleyen aşamalar için planlanmıştır. Ayrıca modelin boyutları arasındaki ilişki için üç kurumun verileri ile istatistiksel olarak anlamlı bir korelasyon çalışması mümkün olmamıştır. Hangi boyutun diğerlerine nasıl etkisi olduğunu anlamak için daha fazla örneğe ihtiyaç bulunmaktadır.

Verimliliğe dair kurum kültürü, takım yapılanması, kullanılan metodolojiler, teknolojiler, çalışan tecrübesi ve demografisi gibi başlıklar sonuçları etkileyebilir [43]. Bu amaçla verimlilik odağı ilgi çeken bir alan olup birçok çalışma üretilmiştir [44]. Ayrıca üzerinde çalışılan projelerin niteliği, teknik mimarisi ve iş alanı da verimliliğe doğrudan etki yapabilir. Bir diğer önemli etken de ürün odaklı çalışılan projelerde, ürünün yaşam döngüsünde bulunduğu aşama ve bu aşamaya uygun gerçekleştirilen projelerin de üretkenliğe ve verimliliğe farklı boyutlarda etkileri olacaktır [45]. Dolayısı ile verimlilik boyutlarındaki iyileşme veya kötüleşme sadece tek bir boyuta bağlı olmayabilir. Ancak farklı etkileri ayrıştırıp analiz edebilmek için modelin daha geniş kitlelerce kullanılması gerekmektedir.

Yapay zekanın yazılım geliştirme alanındaki ağırlığı ve etkisi birçok sektörde olduğu gibi her geçen gün artmaktadır. Verimlilik boyutları incelenirken yapay zekanın yazılım geliştirme adımlarındaki etkisi de ayrı bir başlık olarak değerlendirilmelidir. Yapay zeka kullanılarak yazılım geliştirmenin, modelde belirlenen 9 boyuta doğrudan veya dolaylı etkisi olacaktır. Bu alanda ayrı bir çalışma planlanmıştır.

Kaynakça

[1] Fan, W. Moen, P. "Working more, less or the same during COVID-19? A mixed method, intersectional analysis of remote workers", Work and Occupations, Vol. 49, No. 2, (pp. 143-186), 2022

[2] Chapetta, W. A. Travassos, G. H. "Towards an evidence-based theoretical framework on factors influencing the software development productivity", Empirical Software Engineering, 25, 3501–3543, 2020.

[3] Trigo, A. Varajão, J. Almeida, M. "Low-Code Versus Code-Based Software Development: Which Wins the Productivity Game?," IT Professional, 24(5), 61-68, 2022.

[4] Butt, S. A. Misra, S. Piñeres-Espitia, G. Ariza-Colpas, P. Sharma, M. M."A Cost Estimating Method for Agile Software Development," in Computational Science and Its Applications – ICCSA 2021, 12955, 2021.

[5] Ahmad, M. Z. Ahmad, N. "Parametric Software Reliability Growth Model with Testing Effort: A Review," 2021 International Conference on Computational Performance Evaluation (ComPE), 899-904, 2021.

[6] Liao, Z. Zhao, Y. Liu, S. Zhang, Y. Liu, L. Long, J. "The Measurement of the Software Ecosystem's Productivity with GitHub," Computer Systems Science and Engineering, 36(1), 239-258, 2021.

[7] Liao, Z. Qi, X. Zhang, Y. Fan, X. Zhou, Y. "How to Evaluate the Productivity of Software Ecosystem: A Case Study in GitHub," 2020.

[8] Mota, J. S. Tives, H. A. Canedo, E. D. "Tool for Measuring Productivity in Software Development Teams", Information, 12(10), 396, 2021.

[9] Hussain, A. Raja, M. Vellaisamy, P. Krishnan, S. Rajendran, L. "Enhanced framework for ensemble effort estimation by using recursive-based classification," IET Software, 2021.

[10] Alsunki, A. A. M. Ali, M. A. M. Jaharadak, A. A. Tahir, N. Md "Framework of Software Developers Engagement Antecedents and Productivity - A Review," 2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA), 302-307, 2020.

[11] Michael, J. B. "Placing Trust in Automated Software Development Processes," Computer, 55(7), 78-81, 2022.

[12] Jalote, P. Kamma, D. "Studying Task Processes for Improving Programmer Productivity," IEEE Transactions on Software Engineering, 47(4), 801-817, 2021.

[13] Nordin, A. A. M. Latih, R. Ali, N. M. "Software Development Productivity Model: Validation through Expert Review," 2021 International Conference on Electrical Engineering and Informatics (ICEEI), 1-6, 2021.

[14] Martino, V. L. Gravino, C. "Using COSMIC to measure functional size of software: a Systematic Literature Review," 2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 225-228, 2022.

[15] Alrubaee, U. A. Cetinkaya, D. Liebchen, G. Dogan, H. "A process model for component-based model-driven software development," Information, 11(6), 302, 2020.

[16] Castillo-Salinas, L. Sanchez-Gordon, S. Villarroel-Ramos, J. Sánchez-Gordón, M. "Evaluation of the implementation of a subset of ISO/IEC 29110 Software Implementation process in four teams of undergraduate students of Ecuador. An empirical software engineering experiment," Computer Standards & Interfaces, 70, 103430, 2020.

[17] Khan, S. U., Khan, A. W., Khan, F., Khan, J., & Lee, Y. (2023). Factors influencing vendor organizations in the selection of DevOps for global software development: an exploratory study using a systematic literature review. Cognition, Technology & Work, 25(4), 411-426.

[18] Ebert, C., Arockiasamy, J. P., Hettich, L., & Weyrich, M. (2024). Hints for generative AI software development. IEEE Software, 41(5), 24-33.

[19] Ying, Z. Towey, D. Zhang, Y. "Evaluation of the Code Generated By Large Language Models: The State of the Art," 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC), 440-449, 2025.

[20] Hardill, I. Green, A. "Remote working—altering the spatial contours of work and home in the new economy", New Technology Work and Employment, 18(3), 212-222, 2003.

[21] Hislop, D. Axtell, C. Daniels, K. "The challenge of remote working", 2008.

[22] Sharp, H. Barroca, L. Deshpande, A. Gregory, P. Taylor, K. "Remote working in an Agile team", Agile Research Network, 1-15, 2016.

[23] Zulfany, A. H. Dewi, R. S. Partiwi, S. G. "Analyzing mental workload of remote worker by using SWAT methodology (case study: Remote software engineer)", IOP Conference Series: Materials Science and Engineering, Cilt 598, Sayı 1, Ağustos 2019.

[24] Sudhakar, G. Farooq, A. Patnaik, S. "Measuring productivity of software development teams", Serbian Journal of Management, 7(1), 65-75, 2012.

[25] Delaney, S. Schmidt, D. "A productivity framework for software development literature review", Proceedings of the 2nd International Conference on Software Engineering and Information Management, (pp. 69-74), Ocak 2019.

[26] Iqbal, J. Omar, M. Yasin, A. "An empirical analysis of the effect of agile teams on software productivity", 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), (pp. 1-8), Ocak 2019.

[27] Nordin, A. A. M. Latih, R. Ali, N. M. "Software Development Productivity Model: Validation through Expert Review", 2021 International Conference on Electrical Engineering and Informatics (ICEEI), (pp. 1-6), Ekim 2021.

[28] Rot, A. Sobinska, M. Busch, P. "Programming Teams in Remote Working Environments: an Analysis of Performance and Productivity", 2023 13th International Conference on Advanced Computer Information Technologies (ACIT), (pp. 376-381), Eylül 2023.

[29] Smite, D. Moe, N. B. Klotins, E. Gonzalez-Huerta, J. "From forced Working-From-Home to voluntary working-from-anywhere: Two revolutions in telework", Journal of Systems and Software, 195, 111509, 2023.

[30] Nwankpa, J. K. Roumani, Y. F. "Remote work, employee productivity and innovation: the moderating roles of knowledge sharing and digital business intensity", Journal of Knowledge Management, 28(6), 1793-1818, 2024.

[31] Nguyen-Duc, A. Khanna, D. Le, G. H. Greer, D. Wang, X. Zaina, L. M. P. Abrahamsson, "Work-from-home impacts on software project: A global study on software development practices and stakeholder perceptions", Software: Practice and Experience, 54(5), 896-926, 2024.

[32] Somanathan, S. "Optimizing Agile Project Management for Virtual Teams: Strategies for Collaboration, Communication, and Productivity in Remote Settings", International Journal of Applied Engineering & Technology, 5, 2023.

[33] Jansen, F. Souza Santos, R. De "Remote Communication Trends Among Developers and Testers in Post-Pandemic Work Environments", 2024 IEEE International Conference on Software Maintenance and Evolution (ICSME), Flagstaff, AZ, A.B.D., (pp. 672-677), 2024, doi: 10.1109/ICSME58944.2024.00071.

[34] Pfleeger, S. L. Wu, F. Lewis, R. Software cost estimation and sizing methods: issues, and guidelines, Cilt 269, Rand Corporation, 2005.

[35] Common Software Measurement International Consortium (COSMIC), “COSMIC Software Sizing - open standard for software size measurement”, https://cosmic-sizing.org/, 2025.

[36] Bogdanović, D. Sladojević, S. Arsenović, M. Anderla, A. “Multi-Criteria Analysis of Characteristics of Remote Employee Monitoring Systems”, 21st International Symposium INFOTEH-JAHORINA (INFOTEH), (pp. 1-6), 2022.

[37] Wilkes, B. Milani, A. M. P. Storey, M. A. “A framework for automating the measurement of devops research and assessment (DORA) metrics”, 2023 IEEE International Conference on Software Maintenance and Evolution (ICSME), (pp. 62-72), Ekim 2023.

[38] Altunel, H. Say, B. “Software product system model: a customer-value oriented, adaptable, devops-based product model”, SN Computer Science, 3(1), 38, 2022.

[39] OWASP Foundation, “OWASP Top Ten”, https://owasp.org/www-project-top-ten/, 2025.

[40] Dayyala, N. Walstrom, K. A. Bagchi, K. K. Udo, G. “Factors impacting defect density in software development projects”, International Journal of Information Technologies and Systems Approach (IJITSA), 15(1), 1-23, 2022.

[41] Butt, S. A. Melisa, A. C. Misra, S. “Software product maintenance: A case study”, International Conference on Computer Information Systems and Industrial Management, (pp. 81-92), Cham: Springer International Publishing, Temmuz 2022.

[42] Giro Manzano, P. Customer Satisfaction Measurement: strategies, methodologies and factors influencing customer satisfaction measures, 2021.

[43] Cleaver, K. Teamwork Quality and Success in Remote, Hybrid, and Proximal Agile Software Development (ASD) Teams, Doktora Tezi, The Chicago School of Professional Psychology, 2024.

[44] Hernández, G. Martínez, Á. Jiménez, R. Jiménez, F. “Productivity metrics for an agile software development team: A systematic review”, TecnoLógicas, 22(SPE), 63-81, 2019.

[45] Altunel, H. “Product life cycle based project management model”, The Journal of Modern Project Management, Vol. 4, No. 3, 2017.