Yazılım Projelerinde Büyüklük Tahmini

Emin BORANDAĞ[1], Fatih YÜCALAR[1],Önder ŞAHİNASLAN[2]

[1]Maltepe Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Yazılım Mühendisliği Bölümü
[2]Maltepe Üniversitesi, Bilişim Bölümü

eminb@maltepe.edu.tr, fatihy@maltepe.edu.tr, onder@maltepe.edu.tr

Özet: Yazılım projelerinde, geliştirilecek yazılımın büyüklüğü ve harcanacak emeğin kestirimi, projelerin doğru planlanması ve hesaplanması açısından çok önemlidir. Yazılım geliştirme sürecinin başında, büyüklük, emek ve maliyet kestirimleri geliştiricilerin ve yöneticilerin karşılaştığı en önemli problemlerdir. Yazılım proje yönetiminde çok önemli olan ölçme ve bu kavram çerçevesinde yapılanan kestirim yöntemleri aracılığı ile zaman ve işgücü gibi planlamaların yapılabilme gereği açıktır. Bu çalışmada bir yazılım projesinin büyüklüğü, işlev puanı yöntemi ile hesaplanmıştır. Aynı yazılım projesi, farklı yazılım grupları tarafından gerçekleştirilmiştir. Yazılım projesi için yapılan ilk tahminleme ile yazılım gruplarının gerçekleştirmiş olduğu çalışma sonunda elde edilen veriler karşılaştırılmış ve sonuçlara yer verilmiştir.

Anahtar Sözcükler: Yazılım Proje Yönetimi, Büyüklük Kestirimi, İşlev Puanı Analizi

Size Estimation in Software Projects

Abstract: Estimating the size of software and the effort to be spent to develop it is very important for the accurate planning and calculation of software projects. At the beginning of the software development process, size, effort and cost estimations are the most important problems that developers and administrators face. Necessity of making plans such as time and effort planning through the concept of measuring that is so important in software project administration and the estimation methods takes shape around this concept is obvious. In this study, the size of the software project was calculated through the function point's method. The same software project, implemented by different software development teams. Early estimations made for the software project and the data obtained as a result of the teams' studies are compared and comparison results are provided.

Keywords: Software Project Management, Size Estimation, Function Point Analysis

1. Giriş

Her yazılım projesinin temel hedefi, müşterinin ihtiyaçlarını karşılayan, öngörülmüş bütçe ile zamanında teslim edilen hatasız bir yazılım geliştirmektir.

Yazılımda ölçüm yöntemlerinin kullanılması, yazılım sektöründe gittikçe önem kazanmaktadır. Yazılım ölçümü, yazılım projesini anlamak ve modellemek, yazılım projelerinin yönetilmesine yol göstermek ve yazılım süreç geliştirme çalışmalarını yön vermek açısından yazılım şirketleri için çok önemlidir. Bu amaç doğrultusunda bildirinin ikinci bölümünde yazılım büyüklük yöntemlerinden, üçüncü bölümde yazılım projesinin kapsamında bahsedilecektir. Dördüncü bölümde, projenin başlangıcında yapılan işlevsel büyüklük kestirimi ve geliştirilmesi sonrasında elde edilen ölçütler anlatılmıştır. Son bölümde ise sonuçlar ve önerilere yer verilmiştir.

2. Yazılım Büyüklük Kestirim Yöntemleri

Yazılımın ölçülebilmesi, harcanılan zaman, emek, proje büyüklüğü ve kalite gibi faktörlerin belirlenmesine olanak sağlamaktadır. Organizasyonlar, bu verilere dayanarak ileride alacakları projeler için kestirim yapabilme imkânı bulabileceklerdir. Yazılım projelerinde kaliteyi arttırmak, her şeyden önce doğru ölçme yöntemlerine bağlıdır.

Yazılım büyüklük kestiriminde kullanılan yöntemler; teknik büyüklük kestirim yöntemleri ve işlevsel büyüklük kestirim yöntemleri olarak sınıflandırılmıştır. Teknik büyüklük kestirim yönteminde en çok bilinen yöntem Satır Sayısı (Lines of Code - LOC) yöntemidir [1]. Uygulamanın büyüklüğünü anlamak için bilgisayar programlarındaki kodların satırlarını sayma en geleneksel ve en yaygın şekilde kullanılan yazılım ölçümüdür. Kolaylığı ve doğrudan ölçülebilirliği açısından en fazla kullanılan yazılım ölçme yöntemi, satır sayısıdır.

Ancak kullanılan bu yöntemin bazı dezavantajları vardır. Programlama dili farkı, deneyim farkı gibi nedenlerle LOC yöntemi projenin büyüklüğünü tahminleme de direkt olarak kullanılan bir yöntem değildir. Bu nedenle İşlevsel Büyüklük Ölçümü (Functional Size Measurement - FSM), yöntemleri kullanılmaktadır. Bu yöntemde yazılımın işlevselliğini temel alınmaktadır.

2.1 İşlevsel Büyüklük Kestirim Yöntemleri

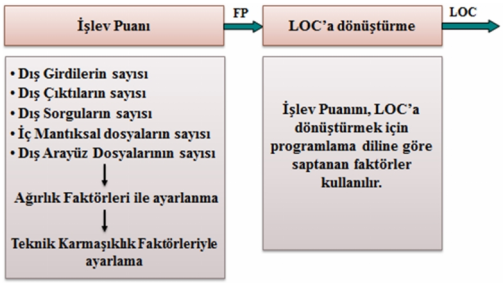
İlk olarak İşlev Puanı (Function Points) ve İşlev Puan Analizi (Function Points Analysis - FPA) 1979 yılında IBM'in satır sayısına alternatif olarak yazılım büyüklük ölçümü için Allan Albrecht tarafından ortaya çıkartılmıştır. 1983'de ise, Allan Albrecht ve John Gaffney tarafından Yönetim Bilgi Sistemlerinin büyüklüğünü ölçmek için FSM yöntemi geliştirilmiştir [2]. Daha sonra farklı kitleler tarafından orijinal FPA yöntemi üzerinde yapılan oynamalarla, aralarında ölçüm yöntemi farklı birçok FSM yöntemi geliştirilmiştir. Aşağıda bazı işlevsel büyüklük kestirim yöntemleri verilmiştir:

- İşlev Puanı (Function Points - FP),
- IFPUG İşlev Puanı Analizi (IFPUG Function Points Analysis – IFPUG FPA),
- Mark II İşlev Puanı (Mark II Function Points – MK II FP),
- Nesma İşlev Puanı (Nesma Function Points),
- Tam İşlev Puanı (Full Function Points – FFP),
- COSMIC Tam İşlev Puanı (COSMIC Full Function Points – COSMIC FFP),
- Nesne Puanı (Object Points),
- Nesne-Tabanlı İşlev Puanı (Object-Oriented Function Points – OO FP),
- Nesne-Tabanlı Yöntem İşlev Puanı (Object-Oriented Method Function Points – OOmFP)

2.2 İşlev Puanı (Function Points)

Bu yaklaşım; verimliliğin, üretilen işlev puanına göre adam-ay olarak belirlenmesini öngörür.

Eğer proje ile ilgili girdi çıktı gibi özellikler tahmin edilebiliyorsa, bunlar kullanılarak geliştirilecek sisteme ait bir İşlev Puanı hesabı yapılabilir ve sonuçlar Satır Sayısına (LOC) çevrilebilir. Bu satır sayısından maliyet, emek ve süre tahmini yapılabilir. İşlev puanı dönüşüm süreci, Şekil 1'de gösterilmiştir [3].



Şekil 1. İşlev Puanı Dönüşüm Süreci

İşlev Puanı'nın hesaplanması ve Satır Sayısı'na dönüştürülmesi süreci beş adımdan oluşmaktadır.

Adım-1: İşlev Puanında sistemin işlevselliği 5 ayrı bileşenle incelenmektedir:

- ***Dış Girdiler:*** Uygulamanın dışından uygulamanın içine doğru olan süreçleri ve işlenebilir verileri gösterir. Veri genellikle uygulamaya içine eklenebilir, silinebilir veya güncellenebilir. Dış girdilere örnek olarak; kullanıcının bilgi girişi yaptığı veri giriş ekranları ve mantıksal dâhili dosyalar verilebilir.
- ***Dış Çıktılar:*** Verinin uygulama sınırları içinden dışarı çıkmasına izin veren süreç veya işlemlerdir. Dış çıktılara örnek olarak; raporlar, doğrulama mesajları ve ekran çıktıları verilebilir.
- ***Dış Sorgular:*** Kullanıcı isteği doğrultusunda alınan hızlı veri çıkışlarıdır Dış sorgular dosyada saklanan veriyi değiştirmez veya güncellemez. Sadece bilgiyi okurlar.
- ***İç Mantıksal Dosyalar:*** Uygulama sınırları ile birlikte verilerin saklandığı mantıksal bir dosyadır. İç mantıksal dosyalara örnek olarak, dâhili kullanıcı verileri, saklanan veriler verilebilir.
- ***Dış Arayüz Dosyaları:*** Başka bir uygulama sistemi ile olan paylaşımı ifade eder.

Tablo 1. İşlev Puanı Karmaşıklık Tablosu

Bileşenler	Basit	Orta	Karmaşık
Dış Girdiler	3	5	6
Dış Çıktılar	4	6	7
Dış Sorgular	3	5	6
İç Mantıksal Dosya	7	13	15
Dış Arayüz Dosya	5	9	10

Adım-2: Düzeltilmemiş İşlev Puanı'nın (Unadjusted Function Points - UFPs) hesaplanması:

$$UFP = [Dış\ Girdiler\ x\ W(1)] + [Dış\ Çıktılar\ x\ W(2)] + [Dış\ Sorgular\ x\ W(3)] + [İç\ Mantıksal\ Dosyalar\ x\ W(4)] + [Dış\ Arayüz\ Dosyaları\ x\ W(5)]$$

Her bir bileşenin zorluk derecesi basit, orta ve karmaşık gibi Tablo 1'de verilen rakamsal değerlere bağlı olarak ölçülebilmektedir. Bu ölçülen değerler toplanarak Düzeltilmemiş İşlev Puanı'nı oluşturmaktadır.

Adım-3: Teknik Karmaşıklık Faktörünün (Technical Complexity Factor - TCF) hesaplanması:

Tablo 2'de verilen 14 genel sistem özelliği kullanılarak sistemin beklenilen uygulama zorluğu için ilave bir Teknik Karmaşıklık Faktörü (TCF) hesaplanır.

Tablo 2. Genel Sistem Özellikleri

	Genel Sistem Özellikleri	Kısa Açıklama
1	Veri İletişimleri	Sistemin uygulaması ile bilgi değişimi veya transferinde yardımcı olmak için kaç tane iletişim aracı vardır?
2	Dağıtılan Veri/İşleme	Dağıtılan bilgi ve işleme fonksiyonları nasıl idare edilmektedir?
3	Performans	Hedefler, yanıtlama zamanı ve iş çıkarma performansı önemli midir?
4	Çok Kullanılan Konfigürasyon	Uygulamanın idare edileceği mevcut donanım platformu ne kadar yoğun kullanılmaktadır?
5	İşlem Oranı	İşlem oranı yüksek midir?
6	Çevrimiçi Veri Girişi	Hangi oranda bilgi çevrimiçi girilmektedir?
7	Son Kullanıcı Verimliliği	Uygulama son kullanıcı verimliliği için mi tasarlanmıştır?
8	Çevrimiçi Güncelleme	Kaç veri dosyası çevrimiçi güncellenmektedir?
9	Karmaşık İşlem Yapma	Dâhili işlem yapma karmaşık mıdır?
10	Yeniden Kullanılabilirlik	Uygulama yeniden kullanılabilir olması için mi tasarlanmıştır?
11	Dönüştürme/Kurulum Kolaylığı	Sistemde otomatik dönüşüm ve kurulum da dâhil edilmiş midir?
12	İşlevsel Kolaylık	Yedekleme, başlatma ve kurtarma gibi operasyonlar ne kadar otomatiktir?
13	Çoklu Saha Kullanımı	Uygulama çoklu örgüte sahip çoklu sahalar için özellikle mi tasarlanmış, geliştirilmiş ve desteklenmiştir?
14	Değişimi Kolaylaştırma	Uygulama kullanıcı tarafından kullanım kolaylığı ve değişimi kolaylaştırmak için özel olarak mı tasarlanmış, geliştirilmiş ve desteklenmiştir?

14 genel sistem özelliği için verilen her bir soruya 0 ile 5 arasında değerler verilir ve bu değerler toplanarak Etki Derecesi (Degree of Influence - DI) hesaplanır.

0: hiç yok ya da etkisiz,
1: önemsiz etki,
2: az etkili,
3:orta düzeyde etkili
4: önemli düzeyde etkili,
5: güçlü etki

$$DI = \Sigma_{i=1..14}\, Cevap_i$$
$$TCF = 0{,}65 + 0{,}01 \times DI$$

Adım-4: İşlev Puanı aşağıda verilen formül kullanılarak hesaplanır:

$$FP = UFP \times TCF$$

İşlev Puanı'nı, Satır Sayısına dönüştürmek için aşağıdaki formülden yararlanılır.

$$LOC = FP \times Prog.\ Dili\ LOC\ Katsayısı$$

Tablo 3. Bazı Programlama Dillerinin LOC/FP Oranları

Programlama Dili	LOC/FP
C	128
C ++	53
COBOL	107
FORTRAN	105
DELPHI 5	18
JAVA 2	46
VISUAL BASIC 6	24
SQL	13
Dördüncü Kuşak Diller	20
Nesne Tabanlı Diller	30

3. Yazılım Projesi Kapsamı

Yazılım projesi toplam yedi modülden oluşan bir Windows uygulamasıdır. Programa ilişkin modüller aşağıda verilmektedir.

a) Kullanıcı Giriş Ekranı
b) Ürün Arama Listeleme Ekranı
   – Arama Kriterleri (Ürün Kodu, Ürün Adı, Kategorilere Göre Arama)
   – Listeleme (Ürün Kodu, Ürün Adı, Kategori Sil, Stok Durumu, Aktiflik)

c) Stok Giriş Güncelleme ve Silme Ekran
   - Ürün Adı, Kategori, Adet, Stok Giriş Tarihi, Hangi Bölüme Gönderilmiş, Aktif

d) Kişisel Bilgiler
   - Ad, Soyad, Bölüm, Unvan (Dışarıdan)

e) Kategori Bilgileri ve Demirbaş Bilgileri Giriş Ekranı

f) Personel Üzerine Demirbaş verilmesi
   - Personel unvanına göre, adına, soyadına ve bölümüne göre arama yapabilmektedir.
   - Personel üzerine demirbaş verme işlemleri yapılabilmektedir.

g) Listeme Raporlama
   - Stok ismine göre, stok tipine göre, stok türlerine göre arama yapılabilmektedir
   - Personel üzerindeki stokları listeleyebilmektedir.

4. Projenin Başlangıcında Yapılan İşlevsel Büyüklük Kestirimi

Proje ile ilgili büyüklük hesabı, işlev puanı yöntemi kullanılarak yapılmıştır. Sistemin işlevselliği 5 ayrı bileşen dikkate alınarak belirlenmiştir. Tablo-4'de hangi düzeyde kaç adet bileşen olduğu gösterilmiştir.

Tablo 4. Karmaşıklık Düzeylerine Göre Bileşen Sayıları

Bileşenler	Basit	Orta	Karmaşık
Dış Girdiler	2	2	0
Dış Çıktılar	0	3	1
Dış Sorgular	0	0	0
İç Mantıksal Dosya	0	1	0
Dış Arayüz Dosya	1	0	0

UFP = [Dış Girdiler x W(1)] +
[Dış Çıktılar x W(2)] +
[Dış Sorgular x W(3)] +
[İç Mantıksal Dosyalar x W(4)] +
[Dış Arayüz Dosyaları x W(5)]

UFP = [(2*3) + (2*5)] + [(3*6) + (1*7)] +
[1*13] + [1*5] = 59

DI = 2 + 2 + 3 + 2 + 2 + 3 + 4 + 3 + 1 +
3 + 2 + 1 + 1 + 5 = 34

TCF = 0,65 + (0,01 * 34) = 0,99

FP = UFP * TCF = 58,41

VB programlama dili ile tahmini proje satır sayısı;

- **LOC** = 58,41 * 24 = 1401,84 olarak bulunmuştur.

Proje için harcanan tahmini emeğin hesaplanmasında COCOMO yöntemi [4] kullanılmıştır. COCOMO yöntemi ile tahmini emek aşağıda verilen formül kullanılarak hesaplanabilir.

$$Emek = 2.4\ (KLOC)^{1.05}$$

Projenin tahmini emeği;

Emek = $2.4\ (1{,}402)^{1{,}05} = 3{,}42$ adam-ay

Proje geliştirme süresi ise;

Süre = $2.5\ (3{,}42)^{0{,}38} = 3{,}99$ ay olarak bulunmuştur.

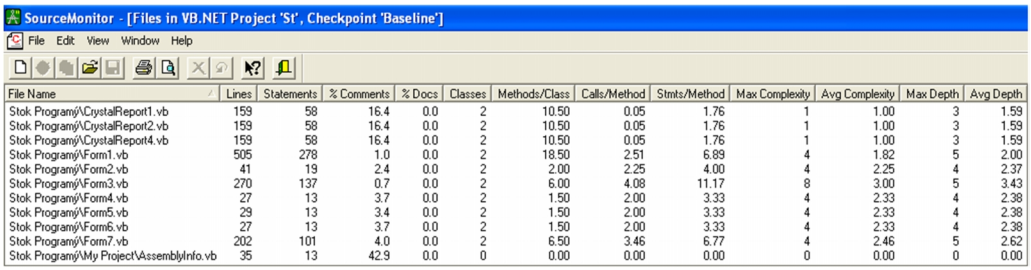
$$N = Emek\ /\ Geliştirme\ Zamanı$$

Formülü kullanılarak projenin yaklaşık olarak kaç kişi ile geliştirileceği bulunabilir:

$$N = 3{,}42 / 3{,}99 = 0{,}86 \approx 1 \text{ Kişi}$$

4.1 Projenin Geliştirilmesi Sonrasında Elde Edilen Ölçütler

Aynı yazılım projesi, üç farklı yazılım ekibi tarafından gerçekleştirilmiştir. Bu yazılım ekipleri aynı teknolojik altyapıyı kullanarak bu yazılım projesini geliştirmişlerdir. Yapılan çalışmalar sonucunda projeye ilişkin veriler, "SourceMonitor V3.3" [5] kullanılarak elde edilmiştir. Şekil 2'de "SourceMonitor" programına ilişkin arayüz görülmektedir.



File Name	Lines	Statements	% Comments	% Docs	Classes	Methods/Class	Calls/Method	Stmts/Method	Max Complexity	Avg Complexity	Max Depth	Avg Depth
Stok Programý\CrystalReport1.vb	159	58	16.4	0.0	2	10.50	0.05	1.76	1	1.00	3	1.59
Stok Programý\CrystalReport2.vb	159	58	16.4	0.0	2	10.50	0.05	1.76	1	1.00	3	1.59
Stok Programý\CrystalReport4.vb	159	58	16.4	0.0	2	10.50	0.05	1.76	1	1.00	3	1.59
Stok Programý\Form1.vb	505	278	1.0	0.0	2	18.50	2.51	6.89	4	1.82	5	2.00
Stok Programý\Form2.vb	41	19	2.4	0.0	2	2.00	2.25	4.00	4	2.25	4	2.37
Stok Programý\Form3.vb	270	137	0.7	0.0	2	6.00	4.08	11.17	8	3.00	5	3.43
Stok Programý\Form4.vb	27	13	3.7	0.0	2	1.50	2.00	3.33	4	2.33	4	2.38
Stok Programý\Form5.vb	29	13	3.4	0.0	2	1.50	2.00	3.33	4	2.33	4	2.38
Stok Programý\Form6.vb	27	13	3.7	0.0	2	1.50	2.00	3.33	4	2.33	4	2.38
Stok Programý\Form7.vb	202	101	4.0	0.0	2	6.50	3.46	6.77	4	2.46	5	2.62
Stok Programý\My Project\AssemblyInfo.vb	35	13	42.9	0.0	0	0.00	0.00	0.00	0	0.00	0	0.00

Şekil 2. Source Monitor Programı

Her yazılım ekibinin gerçekleştirmiş olduğu yazılım projesine ait Kod Satır Sayısı (Lines of Code - LOC) değerleri Tablo 5'de sunulmuştur.

Tablo 5. Yazılım Projelerine İlişkin Kod Satır Sayıları

Grup	Kod Satır Sayısı
A Grubu	2331
B Grubu	1074
C Grubu	1254

Proje başında FP yöntemi kullanılarak tahmin edilen proje büyüklüğü 1402 satır olarak elde edilmişti. Aynı yazılım projesi, üç farklı ekip tarafından gerçekleştirildikten sonra, projelerin büyüklüğüne bakıldığında, bu üç yazılım projesinin ortalama olarak 1553 kod satırından oluştuğu görülmektedir. Gerçekleştirilen yazılım projeleri ile tahmin edilen proje büyüklüğü arasında yaklaşık %10'luk bir sapma vardır. Yazılım projeleri ile ilgili geliştirme zamanlarına bakıldığında; A Grubu 1 adam-ay, B Grubu 15 adam-gün, C Grubu ise 12 adam-günlük sürelerde projeyi geliştirmişlerdir.

5. Sonuç ve Öneriler

Yazılım projeleri için başlangıçta harcanacak emek, maliyet ve zaman tahminlerinin planlanması noktasında temel girdi projenin büyüklüğüdür. Burada sunulan çalışmada yazılım projelerinde büyüklük kestirimi konusu ele alınmıştır. Yazılım büyüklük kestirim yöntemi olarak İşlev Puanı (FP) kullanılmıştır. Aynı yazılım projesi üç farklı ekip tarafından gerçekleştirilmiştir.

Çalışma sonucunda elde edilen bulgular aşağıda verilmektedir:

- Büyük ölçekli yazılım projelerinde olduğu gibi; küçük ölçekli yazılım projelerinde, büyüklüğü tahmin etmek için FP yönteminin kullanılabilir olduğu görülmüştür.
- İleri ki aşamada yapılacak çalışmalarda, nesne-tabanlı diller göz önünde bulundurularak, farklı bir yazılım büyüklük kestirim yönteminin oluşturulması amaçlanmaktadır.

6. Kaynaklar

[1] Fenton, N. E., *"Software Measurement: A Necessary Scientific Basis",* IEEE Transactions on Software Engineering, Vol.20(No.3), 199-206, March, (1994).

[2] Fetcke, T., Abran, A., & Dumke, R., "*A Generalized Representation for Selected Functional Size Measurement Methods"*, 11th International Workshop on Software Measurement, Montreal, Canada, (2001).

[3] Symons, C. R., *"Function Point Analysis: Difficulties and Improvements"* IEEE Transactions on Software Engineering, Vol. SE-14, No. 1, Jan. 1988, S. 2-11. Congress/Conference/Publication (1988)

[4] Hughes, B., & Cotterell, M., *"Software Project Management"*, 5th Edition. McGraw-Hill Education, (2009).

[5] SourceMonitor 3.3 Kurulum Adresi, http://www.campwoodsw.com/sourcemonitor.html, (2012).