İstanbul Beykent Üniversitesi
Yazılım Mühendisliği Bölümü
Yazılım Mühendisliği Tasarım Projesi

Final Raporu
2024 – GÜZ

Proje İsmi (kısaltması ile birlikte)

Grup Elemanları
No, Adı Soyadı

III Tasarım Dokümantasyonu (16 font)

6 Tasarım Hedeflerinin Tanımlanması (14 font)

Tasarım hedefleri, çözümlenecek sistemin en önemli özelliklerini içerir ve sistemin genel tasarımında oldukça etkilidir. Örneğin, bilgisayar oyunlarında öncelik, çözümün doğruluğundan (accuracy) ziyade hızının çok yüksek olmasıdır. Buradan da bir bilgisayar oyununda kullanılan motorun (physical engine), oyunun daha hızlı çalışmasını sağlayacağı düşünülür. Bu tip uygulamalarda geliştirilecek sistemin hedefi sistemin doğruluğu (accuracy) olmayacaktır. Bilimsel hesaplamaların yapıldığı farklı bir projede fiziksel hesaplamalar, hızı düşürecek bile olsa, en önemli kriter olarak belirlenecektir. Bu tür projelerde hedef, ne şekilde olursa olsun doğru hesaplamaların yapılmasıdır.

Tasarım hedefleri ile gereksinimler arasındaki önemli bir fark vardır: Gereksinimler, ürünün müşteriye kabul edilebilir olması için gerçekleştirilmesi gereken her şeydir. Tasarım hedefleri, mümkün olan en iyinin yapılması için gerçekleştirmesi istenilen özelliklerdir. Ayrıca, tasarımcılar hedeflerini belirlerken belirli kabul edilebilirlik şartlarını sağlamak zorunda değildir. Oysa gereksinimler müşterinin istekleridir.

Bu bölümde tanımlayacağınız herhangi bir kavramın hem gereksinimler belirlenirken hem de tasarım hedefinde verilmiş olması mümkündür. Bu nedenle herhangi bir tasarım hedefi, bir gereksinimin gerçekleştirilmesi ile sistemin mümkün olabildiği kadar hızlı çalışmasını sağlamak olabilir. Ayrıca tasarım hedefine göre sistemin belirli bir eşiğin altındaki hızı kabul edilmeyebilir.

Sistem tasarımında probleme uygun bazı kavramların birbirinin yerine geçebileceğinin belirlenmesi önemlidir. Örneğin:

i) fonksiyonellik (functionality) kullanılabilirliğe (usability) karşıttır. Bir sistem 100 fonksiyonlu olarak kullanılabilir mi? Bunun için ne büyüklükte büyük bir menü tasarımı daha uygundur?

ii) Düşük maliyet, (low cost) güçlülük (robustness) ile çelişir. Düşük maliyetli bir sistem, kullanıcı hatalı veri girişi yaptığında hataları (errors) kontrol etmeyebilir.

iii) Etkinlik (efficiency), taşınabilirliğe(portability) karşıdır.

iv) Hızlı geliştirme (rapid development), fonksiyonellik ile çelişir. Örneğin bir projeye geliştirme için 5 hafta verildiği ve bunun 5 programcı ile yapılacağı kabul edilsin. Tasarım süresi ise 2 hafta olsun. Bir sorun durumunda, teslim tarihinin uzatılması mümkün değil ise, bazı fonksiyonellikler azaltılacaktır. Bu durumda modeldeki tüm "use cases" lerin uygulanması mümkün olmayacak ya da proje müşteriye teslim edilemeyecektir.

v) Maliyet (cost), yeniden kullanılabilirliğe (reusability) karşıt olan bir özelliktir. Geçmişte, tasarımın yeniden kullanılabilir hale getirilmesi için ekstra çaba gerekiyordu. Kodlamada nesneler arasındaki pek çok özelliğin yeniden tasarımı yapılmalıydı. Günümüzde, tasarım şablonlarının kullanılmasıyla bu değişim oldukça modernleşti. Tasarım şablonları kullanılarak yeniden kullanılabilme durumu ucuzlamaktadır.

vi) ……..Bunların benzeri olarak örnekler çoğaltılabilir.

Uygulamanıza ait tasarım amaçları örnekleri Şekil 1 'e göre hazırlanacaktır. Tasarım amaçlarınızın neler olabileceği belirlendikten sonra bunların probleminize uygun olarak ifade edilmesi gerekmektedir.

Reliability	Modifiability	Maintainability	Understandability	Adaptability
Reusability	Efficiency	Portability	Traceability of requirements	Fault tolerance
Backward-compatibility	Cost-effectiveness	Robustness	High-performance	Good documentation
Well-defined interfaces	User-friendliness	Reuse of components	Rapid development	Minimum number of errors
Ease of learning	Readability	Ease of remembering	Ease of use	Increased productivity
Low-cost	Flexibility	……	……..	……..

Şekil 6.1: Farklı Tasarım Amaçları

7 Sınıf Diyagramları

Önceki raporda "use case" diyagramları ile açıklanan davranışsal betimlemelerin tümüne ait statik etkileşimlerini gösteren sınıf diyagramları çizilecektir (https://www.uml-diagrams.org/class-diagrams-overview.html). Bu bölümde mantıksal bakış açısına göre projenin tasarımına devam edilir; varlıklar (entities) sınıflar olarak tasarlandığında UML sınıf ve nesne diyagramları tasarlanmış olacaktır.

"use case" diyagramlarının açıklamalarına eş olarak tanımlanan sınıflar ve sınıflar arasındaki ilişkilerde "aggregation", composition", "association","generalization/specilization" ilişkileri mutlaka kullanılmak zorundadır. Bu bağlamda çalışmanın kapsamına / büyüklüğüne uygun sayıda sınıflar arası ilişkinin tanımlanması zorunludur.

8 Dinamik Model

Sistemin dinamik (etkileşim) modeli UML diyagramları ile "sequence diagrams" (https://online.visual-paradigm.com/diagrams/tutorials/sequence-diagram-tutorial/ ile betimlenir Her diyagramın açıklaması olmalıdır.

Herhangi bir raporda açıklaması olmayan bir şeklin anlamı olmaz.

9 Altsistem Ayrıştırması (Subsystem Decomposition)

Geliştirecek sistemin alt sistemlerinin durumunu gösteren yapısal bir diyagramdır. Sistemin ayrıştırması (decomposition) UML "deployment diagram" ile tasarlanabilir (https://www.uml-diagrams.org/deployment-diagrams-overview.html). Bu diyagram, bağlam (context) bakış açısına göre çözüm için önerilen tasarım özetini veren fonksiyonel yapıdır.

IV Test Planları (16 font) (Ana bölüm sayfa başıdır)

Test dokümantasyonu, yazılım projesinin önemli bir kısmıdır ve geliştirilecek projede yapılacak işlemlerin çoğu ayrıntılı olarak açıklanır. Bu bölümde test sürecinin nasıl ilerleyeceği belirlenir. Test planlamasının amacı en etkin ve verimli bir test sürecinin hazırlanarak sistemin çalışacağı ortamın yaratılmasıdır.

IEEE 829-Test Plan dokümantasyonunda yazılımın testi süreçlerinde

i) neler yapılması gerektiği,
ii) bunların hangi kalite standardına göre yapılacağı,
iii) test işlemlerinin hangi zaman ölçeğinde gerçekleştirileceği,
iv) test prosesini hedeflenenler doğrultusunda sağlamakla alınan riskler ve bunların nasıl gerçekleştirileceği açıklanır.

Bu maddelerin tanımlamaları aşağıdaki bölümler tarafından sağlanır.

10 Test Edilebilecek ve Test Edilemeyecek Özellikler

Bu bölüm, IEEE –829 standartına göre "Library Information System" isimli örnek bir yazılım projesinin alt sistemlerine göre aşağıdaki tablolar formatında hazırlanmıştır. Bunlar geliştirilecek yazılım sisteminde test edilecek alt sistemler ve bu alt sistemlerde test proseslerinin gerçekleştirileceği senaryolardır.

Tablo10.1: …….. Sisteminin Test Edilecek Alt Sistemleri /Özellikleri/….

Item Being Tested	Business Scenarios Being Tested
Borrowers sub-system	Creating a new library user
The circulation sub-system	Issuing an item
The circulation sub-system	Renewing an item
The circulation sub-system	Returning an item including overdue fine calculations
The catalogue sub-system	Cataloguing a book
The catalogue sub-system	Removing a book from stock

Aşağıdaki tablo da geliştirilecek yazılım sistemin test işlemine sokmayacağı özellikleri özetlemektedir. Bu alt sistemlerin ilgili özelliklerinin niçin test işlemine sokulmadığı mutlaka açıklanmalıdır.

Tablo 10.2: …….. Sisteminin Test Edilmeyecek Alt Sistemleri/Özellikleri/

Item Being Tested	Business Scenarios NOT Being Tested
Borrowers sub-system	Maintaining the user records
The circulation sub-system	Setting up of holiday dates
The catalogue sub-system	Assigning loan periods

Test edilecek ve edilmeyecek özelliklerin tümü belirlendikten ve kısaca açıklaması yapıldıktan sonra test prosesinin bir bütün olarak nasıl gerçekleştirileceği açıklanır. Bunlar aslında testin planlanması aşamasında hedeflenmiş olan *test düzeyleri*, test tipleri ve *test yöntemleri* olarak

i) Testin el ile ya da otomatik gerçekleştirileceği

ii) Beyaz kutu /kara kutu/ gri kutu testlerinden hangilerinin yapılacağıdır.

Proje gruplarının el ile gerçekleştirilecek testleri mutlaka bulunmalıdır. Aşağıda el ile test tanımlamasına bir örnek verilmiştir.

Tablo 10.3: Elle gerçekleştirilecek testlerin açıklaması

Manual Method Testing	Features Tested	Testing Type
Perform click on menu button and navigation button	Home interface	Functional / Usability
Perform click on next, previous, resume and stop button audio button	Voice Recording chapter lesson and example	Functional / Usability
Perform click on play, next, previous, resume and stop audio button	Audio control button	Functional / Usability
Perform drag and drop	Graphical games	Functional / Usability
Key in answer into text box and click submit button	Generate random question based on chapter topic	Functional / Usability

Test plan tipleri:

i) **Master Test Planı:** geliştirilecek yazılım ürünü için tüm test planlarını birleştiren tek bir üst düzey test planıdır.

ii) **Test Seviyesine Özel Test Planları:** Her test seviyesi için planlanır. Bunlar:

Unit Test Planı

Entegrasyon (Integration) Test Planı

Sistem Test Planı

Kabul (Acceptance) Test Planı olarak eklinde her test düzeyinde tanımlanır.

iii) **Test Tipine Özel Test Planları**: Performans Test Planı ve Güvenlik Test Planı gibi fonksiyonel olmayan, ana test türleri için planlardır.

11 Başarılı / Başarısız Testlerin Değerlendirme Ölçütleri

Önceki bölümlerde planlanmış ve tanımlanmış olan her düzeydeki testlerin aşağıdaki gibi bir örnek tablo ile başarılı /başarısız ölçütleri verilir.

Aşağıdaki örnek interaktif bir öğrenme kiti için sistemin alt sistemlerinin önceden tanımlanmış olan testlerinin (test tipi /test yöntemi) gerçekleştirilen işlemlerden sonra hedeflenen sonuçlarını özetlemektedir. Bu örneğin test işlemlerinin tümünün başarılı olması hedeflenmektedir. Bazı alt sistemlerde ise başarısızlıkla sonuçlanması hedeflenen özellikler tanımlanabilir. Bu örneklerde sonucun başarısız olması beklenir.

Tablo 11.1 : Başarılı / Başarısız Test Değerlendirmeleri

Module	Features	Expected Input	Expected Output	PASS/FAIL
Lesson	Voice Recording chapter lesson and example	User click lesson menu button	Perform Voice Recording Lesson and example based on chapter	PASS
	Audio control button	User Click Audio control button	Perform specific action based on button clicked such as next ,pause, resume and pervious	PASS
Exercise	Generate random question based on chapter topic	User clicked Exercise sub-menu button	Display random question	PASS
		User key in answers	Display editable textbox	PASS
		User clicked check button	Display correct or wrong symbols	PASS
		User click next button	Display new random question	PASS
Games	Generate random question based on chapter topic	User clicked sub-menu button	Display random question	PASS
		User clicked submit button	Display result	PASS
		User clicked OK button	Generate new question and show score	PASS
		Drag and drop	Placed object into correct place	PASS
Past Year Exam	Generate specific exam	User clicked Pass Year	Display exam past paper in a macromedia flash paper	PASS

12 Test Cases

Sistemin alt sistemlerine hangi testler uygulanacak ise her birinin bir test kimliği (test case id) olmalıdır. Her bir alt sistemin özelliklerine (features) hangi testin uygulanacağına (functional /unit/integration veya diğerleri) karar verilir.

Her bir *test case* için input specification (giriş betimlemeleri) ya data (veri) olarak alfanümerik değerler ya da tıklama ya da herhangi bir seçim yapma gibi kişisel eylemlerdir.

Her bir *test case* için beklenen çıktılar (expected outcomes) hatanın (error) kaynağı, hatanın önem derecesi (error severity) ya da hata tipleri (types of error) şeklinde elde edilecektir

Tablo 12.1 : Ayrıntılı Test Durumları ve Sonuçları

Burada sistem için yapılan tüm testler birer başlık altında daha detaylı incelenerek daha fazla açıklanacaktır.

Test Case ID: M-M-01

Test Amaçları: Kullanıcının oyunu başlatması.

Test Prosedürleri ve Sonuçları: Kullanıcı ana menüden eriştiği startgame butonuna bastıktan sonra yeni oyun başlat butonuna basar.

Test Verisi: Kullanıcı Girdisi.

Beklenen Sonuç: Ana menuden eriştiği start game butonundan newgame butonuna basarak yeni bir oyun dosyası oluşturur ve o dosya üzerinden oyunu başlatır.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: -

Hata Önemi: Önemli.

.....

Test Case ID: K-K-03

Test Amacı: Kullanıcın kaçınma aksiyonunu tanımlanan şekilde yapıp yapmadığını kontrol etmek

Test Prosedürleri ve Sonuçları:Kullanıcının kaçınma aksiyonunu gerçekleştirmesi için tanımlanmış olan tuşa basması

Test Verisi: Kullanıcı girdisi.

Beklenen Sonuç: Kullanıcının kaçınma aksiyonunu gerçekleştirecek olan tuşa basmış olması.

Gerçek Sonuç: Beklenen Sonuç ile aynı

Hata Türü: -

Hata Önemi: Önemli.

.....

Test Case ID: O-A-01

Test Amacı: Oyuncu karakterinin oyundaki aşılamayan engel ile etkileşimi.

Test Prosedürleri ve Sonuçları:Oyuncu karakterinin aşılamayan bir engel ile karşılaştırılması sonucunda belirlenen alanın ötesine geçirilmeye çalışması.

Test Verisi: Kullanıcı girdisi

Beklenen Sonuç: Oyuncu karakterinin aşılamayan bir engel ile karşılaştırılması sonucunda belirlenen alanı aşamaması

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü:-

Hata Önemi: Önemli.

.....