YAZILIM MÜHENDİSLİĞİ DİYAGRAMLARININ KULLANIMINDAKİ BİLİŞSEL VE DAVRANIŞSAL ÖZELLİKLER

Ö. Kılıç[1], N. Çağıltay[2], G. Tokdemir[3]

[1]Atılım Üniversitesi, Yazılım Mühendisliği Bölümü, 06836 Ankara
e-posta: ozkankilic@atilim.edu.tr
[2]Atılım Üniversitesi, Yazılım Mühendisliği Bölümü, 06836 Ankara
e-posta: nergiz@atilim.edu.tr
[3]Atılım Üniversitesi, Bilgisayar Mühendisliği Bölümü, 06836 Ankara
e-posta: gtokdemir@atilim.edu.tr

ÖZET

Diyagramlar diğer disiplinlerde olduğu gibi yazılım mühendisliğinde de mühendislere, geliştirilen bilgi sisteminin tüm mimarisini sunmak amacıyla yaygın olarak kullanılmaktadır. Veri akış diyagramları (DFD) ve varlık ilişki diyagramları (ERD) yazılım mühendisliğinin ana gösterimleridir. Yazılım mühendisleri bu diyagramları bilgi sistemini anlamak, geliştirmek ve değerlendirmek için kullanırlar. Ancak bu diyagramları kavrama ve anlama sürecinde mühendislerin bilişsel ve davranışsal özelliklerini incelemek konusunda çok az çalışma yapılmıştır. Bu bildiride yazılım mühendislerinin ER ve DFD kullanılarak tasarlanmış bir sistemdeki hataları bulurken sergiledikleri bilişsel ve davranışsal özellikler incelenmiştir. Bu çalışmada bir göz-takip cihazı aracılığı ile anlama süreci ile ilgili toplanan çeşitli veriler, geçmişe-yönelik anketler ve katılımcı ile yapılan görüşme kayıtları deney verisi olarak kullanılmıştır.

ABSTRACT

Like in other disciplines, diagrams are widely used in software engineering as well in order to provide engineers with the overall architecture of the information system being developed. Data flow diagrams (DFD) and entity relationship diagrams (ERD) are one of the major notations in software engineering. Software engineers use these diagrams to understand, improve and evaluate information systems. However, few studies have been made to investigate engineers' cognitive and behavioral properties during diagram comprehension. In this paper, software engineers' cognitive and behavioral properties are investigated while they are detecting the defects in a system designed using ER and DFD. An eye tracking device and retrospective questionnaires are employed to collect experimental data in this study.

Anahtar Sözcükler: Yazılım Mühendisliği, Grafiksel Muhakeme, Hata Bulma, ERD-DFD

GİRİŞ

Diyagramlar bilgi sistemlerinde de geniş bir kullanım alanına sahiptirler. Literatürde de bu konuyla ilgili birçok çalışma mevcuttur. Larkin (1987) diyagramlarla gösterimin sözel anlatımlardan daha güçlü olduğunu belirtmiştir. Diyagramlar problem çözümü için gerekli bilgiyi toplayıp, artırırlar ve çözüm için gerekli düşüncenin dış dünyadan gelen bilgi ile geliştirilmesini mümkün kılarlar (Zhang, 1997). Bu bağlamda Yazılım Müdendisliği'nde veri akış diyagramları ve varlık ilişki diyagramları, sistem tasarımında geniş bir kullanım alanına sahiptirler. Veri akış diyagramları (DFD) bilgi sisteminin

yapıları arasındaki bilgi akışını gösteren, varlık ilişki diyagramları (ERD) ise sistemin veri modelini gösteren grafiksel gösterimlerdir.

DFD ve ERD diyagramları bilgi sistemlerinin geliştirilmesi sırasında, mimarisinin nasıl olması gerektiğiyle ilgili bilginin toplanmasını ve anlaşılmasını sağlarlar. Sistem geliştiriciler, sistem tasarımcısının grafiksel gösterimlerini kullanarak tüm sistemi anlamaya çalışırlar. Bu diyagramlar, şekil bulma ve bilişsel sembol operasyonlarını desteklediklerinden problem kümesi ile şekilsel gösterim arasında bir köprü kurarlar (Larkin & Simon, 1987). Bu diyagramların algılanmasından sonra, sistem geliştiriciler diyagramlardaki sistem uyumluluğunu, hataları ve yanlış gösterimleri bularak problem çözme işlemini gerçekleştirirler. Klemola ve Rilling (2002), yazılım geliştirmede sistem geliştiricilerin anlama ve bilişsel etkinliklerini modelleme konusunda çalışmışlardır. Ancak, yazılım diyagramlarının değerlendirilmesi sırasında insan davranışlarının ve bilişsel işlemlerin incelenmesi konusunda çok az çalışma yapılmıştır. Stenning ve diğerleri (2001), diyagramların anlamsal ve çözümsel verimliliğini muhakeme yapma yönünden incelemişlerdir. Wohlin ve Aurum diyagramlarda denekler tarafından bulunan hata sayısını, en çok bulunan hata tiplerini ve yanlış bulunan hataları çalışmıştır. Bu çalışmada, kontrol listesi tabanlı okuma tekniği 486 öğrenci üzerinde uygulanmış ve bu tekniğin ERD diyagramlarında faydalı olduğu bulunmuştur. Hungerford ve diğerleri (2004) ise yazılım kalitesini artırmak için yazılım diyagramlarının hatalarının bulunması konusunda çalışma yapmışlardır. Bu çalışmada, 12 tecrübeli yazılım geliştiricinin diyagramları incelerken kullandıkları arama yöntemleri protokol analiz yöntemi ile araştırılmıştır.

Görsel bir şekle bakarak bir iş yaparken, odaklanılan nokta ve odaklanılan süre ilgilenilen yeri belirler (Just and Carpenter, 1976). Odaklanılan nokta ve gözün izlediği yol, şeklin işlenmesinin ve anlaşılmasının ne kadar zor veya kolay olduğunu gösterir (Wickens & Holland, 2000). Goldberg ve Kotval (1999) uzun süreli göz odaklanmalarının şekilden bilgi çıkartılmasındaki zorluk derecesini gösterdiğini, aynı zamanda, gözün izlediği yolun, ilgilenilen yer, bilişsel yük ve çeşitli arama stratejilerinin de göstergesi olduğunu belirtmişlerdir. Göz arama tekniği bakılan grafikteki yer konusunda numerik veri sağlar ve bu teknikte gözün bekleme süresi, ve arama sırası da çalışılmıştır (Renshaw et. al., 2004).

Bu çalışmada, tecrübeli bir yazılım mühendisinin yazılım sistem tasarlama diyagramları olan DFD ve ERD diyagramlarındaki hata bulmaları konusu incelenmiştir. Çalışmada denek gereklilik okuma işlemi sırasında hiçbir şekilde desteklenmemişler (ad-hoc reading technique), ses ve göz hareketinin izlenmesi yoluyla veri toplanmıştır.

YÖNTEM

Bu çalışma sırasında kullanılan ihtiyaç dökümanı, Hungerford (2004)'un kullandığı döküman temel alınarak hazırlanmıştır. Çalışmada bir şirkete ait ihtiyaçlar ve ihtiyaçların karşılığı olarak geliştirilecek sisteme ait DFD ve ERD diyagramları katılımcıya verilerek bu diyagramlardaki hataları bulmaları istenmiştir. Bu işlem sırasında deneğin göz hareketleri ve sesi kaydedilerek hata bulma stratejileri incelenmiştir. Üç adet DFD diyagramı ve bir adet ERD diyagramı kullanılmıştır. DFD-0 sistemin en üst seviyedeki mimarisini gösterir ve sadece sistemin anlaşılmasına yardımcı olması için hatasız olarak deneklere sunulmuştur. Diğer iki DFD diyagramı ise, sistemin iki temel işleminin detaylı veri akışını göstermektedir. DFD diyagramlarına toplam 17 hata konulmuştur. ERD diyagramı ise sistemin varlık ilişkilerini göstermekte ve 12 hata bulundurmaktadır. Bu çalışmada hatalar üç grupta kategorize edilmiştir: MEF, eksik varlık veya eksik işlem; MRD, eksik ilişki veya eksik veri akışı; I, yanlış veri hata tiplerini göstermektedir. Tablo 1, ERD ve DFD diyagramlarında hataların kategorilerine göre dağılımlarını göstermektedir.

Tablo 1. ERD ve DFD diyagramları hata kategorileri ve hata sayısı

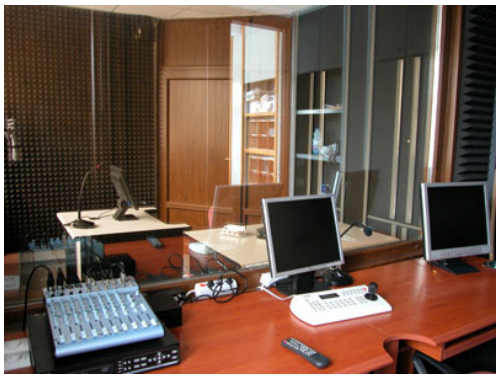
Kategori		**Her diyagramdaki hata sayısı**			
Kod	**Açıklama**	**ERD**	**DFD1**	**DFD2**	**Toplam**
MEF	eksik varlık / işlem	2	1	1	4
MRD	eksik ilişki / veri akışı	4	4	5	13
I	yanlış veri	6	3	3	12
	Toplam	12	8	9	29

Deney yapılmasından bir hafta önce, katılımcıya sistem ihtiyaç dökümanı ve diyagramlarda kullanılan notasyonların açıklaması gönderildi. Deney sırasında bu sisteme ait olan DFD ve ERD diyagramları bilgisayar ortamında katılımcıya sunularak, ihtiyaç dökümanıyla uymayan noktaların bulunması istendi. Bu sırada katılımcının göz hareketleri göz tarama cihazı Tobii (www.tobii.com) ile kayıt edilirken, aynı zamanda

sesli hata bulma veriside kayıt edildi. Tobii, Tobii 1750, Tobii x50 ve Tobii ET-17 cihazlarından oluşmaktadır. Ericsson ve Simon (1993)'in belirttiği gibi, bu çalışmada da hem hata bulma sırasındaki kendi kendine konuşma sesli protokolleri hem de çalışma sonrasında yapılan ankete ait sesli protokoller kullanılmış ve bu sesli veriler göz tarama verileriyle karşılaştırılmıştır.

Deney sırasında, gözlemci ayrı bir odada katılımcıyı izlemiş ve dinlemiş, hareketleri konusunda notlar almıştır. Test ve gözlemci odası Şekil 1 ve 2'de de görüldüğü gibi bu tür çalışmalar için özel olarak hazırlanmış bir odadır.

Şekil 1 Test Odası



Şekil 2 Gözlemci Odası

Hipotezler

Literatür çalışmasının sonuçlarına göre yazılım geliştiricilerin yazılım diyagramlarında hata bulmaları sırasında nasıl davrandıkları konusunda bir framework çalışmasına rastlanılmamıştır. Oysa bu diagramlar yazılım geliştirme süreci içinde son derece kritik bir öneme sahiptir. Bu durum göz önünde bulundurularak bu çalışmada aşağıdaki soruların cevaplanması hedeflenmiştir:

Tecrübeli bir yazılım mühendisinin bir yazılım diyagramına baktığında,

1. Belli bir bölgede gözünün odaklanması ile hata bulma sayısı ve hata yerinin ilişkisi nedir?
2. Sesli protokol tekniğiyle ve göz tarama cihazıyla elde edilen verinin arasındaki ilişki nedir?

Çalışmanın Modeli

Bu araştırmada, bir yazılım mühendisinin kendisine sunulan gösterimleri (ERD ve DFD) anlaması sürecinde nasıl davrandığı konusu incelenmiştir. Bu amaçla, katılımcının göz

hareketleri, davranışları, sözel ifadeleri kaydedilmiş, ayrıca kendisi ile yapılan görüşmelerdeki görüşleri alınmış ve detaylı olarak durumun incelenmesi sağlanmıştır. Deney yaklaşık iki saat sürmüştür. Deneyden hemen sonra katılımcı ile yarım saatlik bir görüşme yapılarak yaşadığı sorunlar ve anlama sürecindeki yaklaşımı konusunda bilgi alınmıştır.

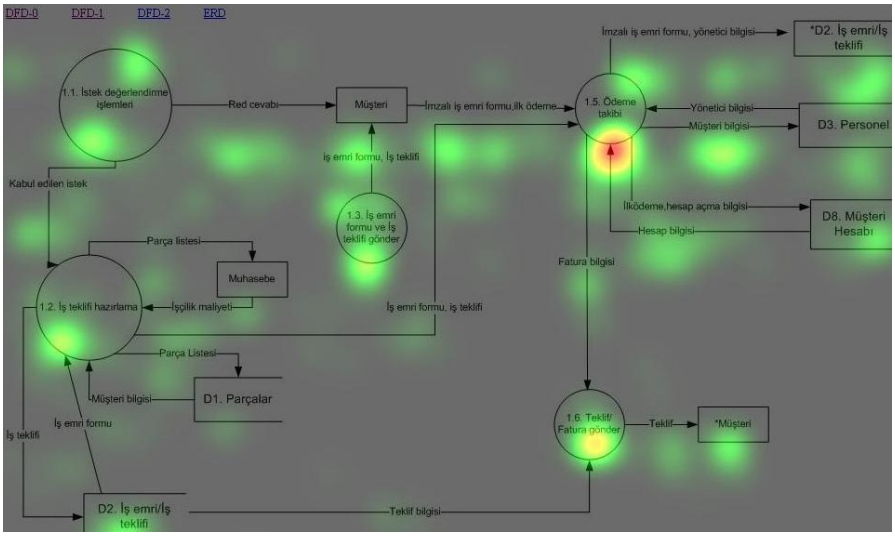
Katılımcı

Katılımcı, 32 yaşında bir bayandır. 9 yıldan beri yazılım mühendisi olarak çalışmaktadır.

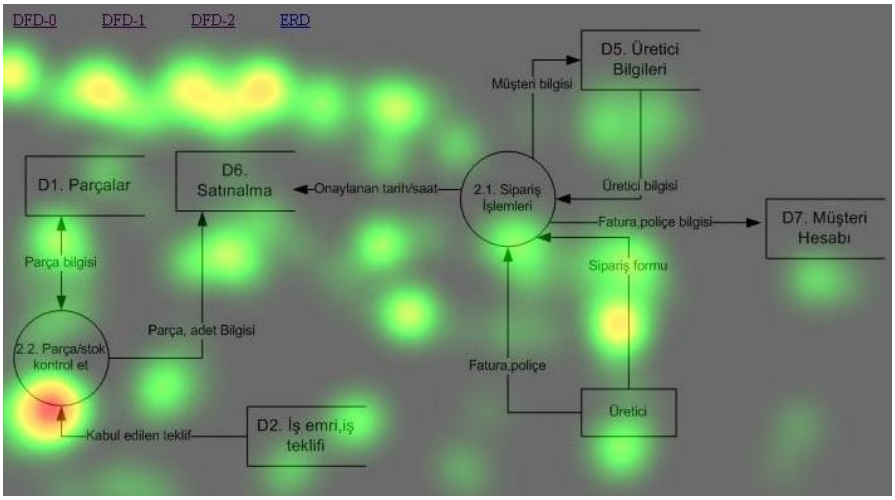
BULGULAR

Göz Takip Cihazı Bulguları:

Göz takip cihazının verilerine göre, gözün şekilleri tarama sırası yatayda ve dikeyde seri bir şekilde gerçeklemiştir. Katılımcı, hata olduğundan şüphelendiği bölgelerde odaklanmayı arttırmıştır. Şekil 3, Şekil 4 ve Şekil 5'te odaklanma bölgeleri görülmektedir.



Şekil 3 DFD1'de gözün odaklandığı noktalar (Yeşil>Sarı>Kırmızı odaklanma sırasındaki artışa gösteriyor)



Şekil 4 DFD2'de gözün odaklandığı noktalar (Yeşil>Sarı>Kırmızı odaklanma sırasındaki artışa gösteriyor)

Şekil 5 ERD'de gözün odaklandığı noktalar (Yeşil>Sarı>Kırmızı odaklanma sırasındaki artışa gösteriyor)

Bu grafikte katılımcının odaklanma derecesine göre aşağıdaki kodlama yapılmıştır:

1. Kırmızı bölgeler, odaklanmanın en yoğun olduğu bölgeler
2. Sarı bölgeler, odaklanmanın orta düzeyde olduğu bölgeler
3. Yeşil bölgeler, odaklanmanın az olduğu bölgeler

Hatalar ve Bulunma Zamanları:

Katılımcı, deneye başladıktan sonra yaklaşık bir saat sistemi anlamaya çalışmış, dokümanları tekrar okuyarak gösterimlerle karşılaştırmıştır. Bu sürenin sonucunda hataları bulmaya başlamıştır. Aşağıdaki tabloda bulunan hataların kodları ve bulunma süreleri verilmiştir.

Hata Kodu	Hata Tipi	Odaklanma	Hata bulma önceliği
DFD1_2	I	3	2
DFD1_5	I	2	3
DFD2_4	I	3	6
DFD2_5	I	3	7
ERD_6	I	1	8
ERD_9	I	2	9
DFD1_3	MRD	3	1
DFD1_4	MRD	2	4
DFD2_1	MRD	3	5
ERD_5	MRD	3	10

Katılımcı öncelikle veri akış gösterimlerine (DFD) yoğunlaşmıştır. Birinci DFD'deki toplam 3 yanlış veri (I) hatasından iki tanesini tanımlamıştır. Aynı gösterimdeki dört adet eksik ilişki / veri akışı (MRD) türündeki hatadan ikisini bulabilmiştir. Katılımcı, bu gösterimdeki toplam 8 hatadan dördünü tespit edebilmiştir.

İkinci veri akış şemasında ise, yanlış veri (I) türündeki toplam üç hatadan ikisini 3 seviyesinde yoğunlaşarak bulmuştur. Eksik ilişki/veri akışı (MRD) türündeki beş hatadan birisini tespit edebilmiştir. Katılımcı, ikinci gösterimdeki toplam 9 hatadan üçünü tespit etmiştir.

Varlık ilişki gösteriminde ise, toplam 12 hatadan üçünü tespit edebilmiştir. Bunlardan ikisi yanlış veri tipindeyken (I) diğeri eksik ilişki / veri akışı (MRD) tipindedir.

Genel olarak, Eksik varlık, işlem (MEF) türündeki hataların hiç birisini bulamamıştır.

DFD'lerdeki yanlış veri (I) ve eksik ilişki/veri akışı (MRD) türündeki hatalarda tespit edilmesi için ortalama benzer derecelerde yoğunlaşmıştır. Ancak ERD'lerde I türündeki hataları bulmak için çok daha uzun süre yoğunlaşmıştır.

Ses Kayıt Bulguları:

Odaklanma sırasında (kırmızı bölgeler) denek, hatadan emin olmaya ve problemi çözmeye çalıştığı için susma davranışı göstermiştir. Katılımcı, hata olduğuna dair ipucu bulduğunda ve hatadan emin olduğunda konuşmuş ve problem çözmesini anlatmıştır.

Röportaj Bulguları:

Katılımcı röportaj sırasında Yanlış veri (I) tipindeki hataları daha kolaylıkla bulduğunu belirtmiştir. Ancak göz tarama cihazının verilerine göre her iki hata türü için de benzer sürelerde yoğunlaşmıştır. Katılımcı hataları bulurken zorlanmış ve nasıl bir yöntem izleyeceğine uzun bir süre karar verememiştir. Bu durumu görüşme sırasında şöyle ifade etmiştir: "Nasıl yapacağımı bilemedim, okuyup tek tek baktım, bir numarayı bir yerde bulmama gerekiyor sandım"

TARTIŞMA ve SONUÇLAR

Hata olduğundan şüphelenilen bölgelere odaklanma dışındaki bölgelerde de göz, yatay ve dikey gezmesini sürdürmüştür. Bunun nedeni, katılımcının kısa dönemli belleğinde oluşturduğu görsel resmi tazeleme eğilimidir çünkü kısa dönemli bellekte görsel verinin unutulmaması için 0.5-1 saniye'de tekrar tazelenmelidir (Sperling, 1963).

Problem çözümü genellikle şu sırada olmuştur:

1. Hata olduğundan şüphelenme
2. Yazılı metinden kontrol etme
3. Şüphelenilen bölgeye odaklanıp kontrol etme
4. Hata raporlama

Katılımcının sözel olarak bildirdiği durumlar ile Göz Hareketlerini İzleme cihazının verileri arasında farklılıklar söz konusu olmuştur. Örneğin katılımcı DFD'lerde yanlış veri (I) tipindeki hataları bulmakta daha çok zorlandığını söylemiştir ancak Göz Hareketlerini İzleme cihazının veriler bulunan her iki türdeki hata için de benzeri

sürelerde yoğunlaşıldığını göstermiştir. Bu çalışmanın sonucuna göre, her iki gösterimde de (DFD ve ERD) eksik varlık/işlem türündeki hataların bulunmasının diğer hata türlerine göre daha zor olduğu söylenebilir.

Not: Bu çalismada bahsedilen/kullanılan Göz Hareketlerini İzleme cihazı TUBITAK SOBAG 104K098 nolu arastırma projesi ile sağlanmıştır. Çalışmamıza destek olan ODTU İnsan Bilgisayar Etkileşimi araştırma grubuna (http://hci.metu.edu.tr) teşekkür ederiz.

KAYNAKLAR

Zhang, J. (1997). "The Nature of External Representation in Problem Solving," *Cognitive Science*, vol. 21, no. 2, Apr. 1997, pp. 179-217.

Larkin, J.H., Simon, H.A. (1987). "Why a Diagram is (Sometimes) Worth Ten Thousand Words," Cognitive Science, vol. 11, no. 1, Jan.–Mar. 1987, pp. 65-99.

Klemola, T., Rilling, J. (2002). Modeling comprehension processes in software development. Proceedings of the First IEEE International Conference on Cognitive Informatics, 2002.

Stenning, K., Lemon, O. (2001). Aligning Logical and Psychological Perspectives on Diagrammatic Reasoning. Artificial Intelligence Review, 2001.

Hungerford, B., Hevner, A.R., Collins, R.W.(2004). Reviewing Software Diagrams: A Cognitive Study. IEEE Transactions in Software Engineering, co-authored with A.R. Hevner and R.W. Collins. Vol 30, No 2, February 2004.

Ericsson, K. A., & Simon, H. A. (1993). Protocol analysis; Verbal reports as data (revised edition). Cambridge, MA: Bradford books/MIT Press.

Renshaw, J. A., Finlay, J. E., Tyfa, D., Ward, R. D. (2004). Understanding visual influence in graph design through temporal and spatial eye movement characteristics, Interacting with computers, 16 (2004) 557-578.

Carpenter, P. A., Shah, P. (1998). A model of the perceptual and conceptual processes in graph comprehension. Journal of Experimental Psychology: Applied 4, 75-100.

Wickens, C. D., Holland, J. G. (2000). 3rd ed., Engineering Psychology and Human Performance, Longman, London.

Goldberg, J. H., Kotval, X. P. (1999). Computer interface evaluation using eye movements: methods and constructs. International Journal of Industrial Ergonomics 24, 631-645.

Sperling, G. (1963). A model fpr visual memory task. Human Faktors, 5, 19-31.

1. **Bu notasyonla çalışırken istediğiniz herhangi bir sırayı izlediniz mi, yoksa sistem sizi belli bir sırayı takip etmek zorunda bıraktı mı?**

   Soldan sağa/ rakamlarla ilişkilimi diye düşündüm
2. **Sizce DFD1 ne kadar karışıktı, 0-5 arasında değerlendiriniz (5: çok zor)**
   3.5 (0 en zor)
3. **Sizce DFD2 ne kadar karışıktı, 0-5 arasında değerlendiriniz (5: çok zor)**
   4 (0 en zor)
4. **Mavi Şirketi ile ilgili açıklama ve DFD arasındaki ilişki sizce nasıldı, uyumlu muydu, nasıl olmalıydı?**
   Açıktı detaylar ama örneğin iş emri-iş teklifini karıştırdım, ne demek istediğini anlamadım. Üreticinin fonskiyonunu anlamadım
5. **DFD deki işlemlerin gruplamaları sizce mantıklı mıydı, nasıl olmalıydı?**
   Evet ama bazı şeyler birleşebilirdi örneğin teklif/fatura gönderme ve teklif aynı şey gibi geldi
6. **Şekillerden hangilerini algılamak daha KOLAY oldu?**

Data store

7. **Şekillerden hangilerini algılamak daha ZOR oldu?**
   İşlemleri algılamak, ne olduğunu, her process de ne var, ne olmalı
8. **Farklı parçaları birleştirmeniz ya da karşılaştırmanız gerektiğinde bunları bir bütün olarak algılayabildiniz mi?, hayırsa neden?**

DFD1 ile DFD2 arasında ilişki olmalı mı? Sanki DFD2 DFD1 in içinde olmalıydı.

ER

1. **Hangi hataları kolaylıkla buldunuz? (MEF, MRD, I)**

1-N, M-N ilşkileri bulmak zordu

2. **Hangi hataları bulmakta zorluk çektiniz? (MEF, MRD, I)**
   İlişkiler kolay, attribute ile relationlar match etmiyordu

3. **Hataları kolaylıkla bulmayı sağlayan faktörler nelerdi**
   Açıklamalrın olması (relation için)
4. **Hataları bulmayı karmaşıklaştıran faktörler nelerdi?**
   notasyon
5. **Hataları bulurken ne yaptın hangi yolu izledin?**

Tablo bazlı baktım, tablolar arasındaki ilişkiler doğru mu, başka tablo olmalı mı. Entity lere baktım, relationların olduğunu sonra anladım

6. **Doğru bulduğu bir ER hatasını nasıl bulduğunu açıklat**
   İnheritence- Personel,görevli hatası. Açıklamalardan buldum
7. **Hata olarak gösterdiği ancak bizim hata listemizde bulunmayan bir ERD hatasını nerden buldugunu açıklat**

iş emri, iş teklifi karıştı,

8. **Bu notasyonla çalışırken istediğiniz herhangi bir sırayı izlediniz mi, yoksa sistem sizi belli bir sırayı takip etmek zorunda bıraktı mı?**
   En kolay anlayacağım yere baktım, örneğin personel, güvenlik etc..
9. **Sizce ERD ne kadar karışıktı, 0-5 arasında değerlendiriniz (5: çok zor)**
   5 bu notasyona alışık değilim
10. **Mavi Şirketi ile ilgili açıklama ve ERD arasındaki ilişki sizce nasıldı, uyumlu muydu,nasıl olmalıydı?**
    Notasyonu bilzsseydim daha iyi olurdu, uyumlu
11. **Şekillerden hangilerini algılamak daha KOLAY oldu?**
    Az şekil vardı, entity kolay
12. **Şekillerden hangilerini algılamak daha ZOR oldu?**
    O,1 çoklular
13. **Farklı parçaları birleştirmeniz ya da karşılaştırmanız gerektiğinde bunları bir bütün**

olarak algılayabildiniz mi?, hayırsa neden?

Hiyerarşi var, bakınca ne olduğu anlaşılıyori takibi kolay

14. **Dökümanlardan hangisi fazla, hangisi eksik, sizce nasıl olmalıydı?**
    -notasyon önceden gönderilmeliydi
    -attributlarla ER match etmeliydi veya verilseydi kafamda önceden canlandırabilirdim
    -ben çizseydim daha kolay olurdu
    -6. paragraf ayrı kalmış, ilkine konsaydı daha iyiydi, numalar çıksa daha iyi olurdu
    -DFD levellarının ne kadar detay taşıması gerektiği açık değil
    -UML deki ilk notasyon, işlemler aktarılırken yapılan, onlarla processleri açıklamak daha anlaşılır