# YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ

Yazılım yaşam döngüsü; bir yazılımın üretilmesinden, yazılımın geliştirilip sonuçlanmasına kadar olan tüm aşamalar şeklinde tanımlanır. Yazılımla ilgili gereksinimler sürekli değiştiği için yazılım yaşam döngüsündeki aşamalar bir döngü halindedir. Yazılım yaşam döngüsünün beş tane temel adımları vardır. Bunlar **Planlama**, Yazılım yaşam döngüsünün başlangıcıdır. Proje için fizibilite çalışmalarının yapıldığı, gerekli olan ihtiyaçların belirlendiği, projenin planının oluşturulduğu aşamadır. **Çözümleme,** Yazılım işlevleri ve gereksinimlerinin ayrıntılı bir şekilde çıkarıldığı aşamadır. Bu aşamada UML diyagramları çizilir. **Tasarım,** Çözümleme aşamasından sonra belirlenen gereksinimlere yanıt verilir. Bu aşama iki alt yapıdan meydana gelmektedir. Bunlar \*\*Üst Seviye ve Mimari Tasarım: Mevcut sistem değil önerilen sistem ele alınır. \*\*Detaylı Tasarım: Yazılımı içeren bileşenleri ve bunların ayrıntıları ele alınır. **Gerçekleştirme,** Modüllerin kodlandığı, test edildiği ve kurulum çalışmalarının yapıldığı aşamadır. **Bakım,** Hataların giderildiği ve yeni eklentilerin yapıldığı aşamadır.

\*\*\* YAZILIM YAŞAM DÖNGÜSÜ MODELLERİ \*\*\*

**Gelişigüzel Model**, 1960 yıllarında kullanılan bir modeldir. Genellikle kişiye bağlı yazılım geliştirilir. Bu yüzden oluşturulan uygulamaların uygulanabilirliği ve bakımı oldukça zordur. Basit programlama yöntemidir. Gelişigüzel modelde belirlenmiş model ya da yöntem bulunmaz. Genellikle tek kişilik üretimlerde görülür.

**Barok Modeli ,**1970 yıllarında ortaya çıkan bir modeldir. Yazılım yaşam döngüsü temellerini doğrusal bir şekilde ele alır ve geliştirir. Aşamalar arası geri dönüşümü tanımlı değildir. Gerçekleştirelim aşamasına ağırlık veren modeldir. Bu yüzden günümüzde kullanımı tavsiye edilmemektedir. Bu model de belgeleme günümüz modellerden farklı olarak ayrı bir süreç olarak ele alır.

**Çağlayan Modeli,** Çağlayan modeli geçmiş yıllarda en yaygın kullanılan yazılım geliştirme modelidir. Geleneksel model olarak da bilinmektedir. İyi tanımlanmış ve az zaman gerektirin projeler için kullanımı uygun bir modeldir ama kullanımı günümüzde gittikçe azalmaktadır. Çağlayan modelinde yazılım, aşamaların tekrar edilmesiyle geliştirilir. Üretimi Çağlayan modeli, gereksinim tanımlama, sistem ve yazılım tasarımı, gerçekleştirme ve birim test, birleştirme ve sistem testi, işlem ve bakım aşamalarından oluşmaktadır. Gereksinim tanımlama aşaması, yapılacak olan projenin gereksinimlerinin belirlendiği aşamadır. Bu aşamada personel, donanım ve sistemin gereksinimleri belirlenir. Sistemin fizibilite çalışmaları yapılır. Kullanıcıların problemlerinin analizinin yapıldığı aşamadır. Bu aşamada proje planının oluşturulması adımlarından oluşur. Sistem ve yazılım tasarımı aşaması, gereksinimleri belirlenmiş olan projenin yapısal ve detay tasarımının oluşturulmasıdır. Gereksinimleri tanımlama aşamasındaki gereksinimlerin yanıtlandığı aşamadır. Yapılacak olan sistem için kullanılacak olan teknikler, gösterim, desen ve strateji ile ilgilidir. Bu aşama mimari tasarım, içsel ara yüzler, kullanıcı ara yüzü, tasarım araçları, tasarımın değerlendirilmesi, veri tasarımı gibi alt süreçleri de kapsar. Gerçekleştirme ve birim testi aşaması, tasarımı gerçekleşmiş olan proje sisteminin kodlara dönüştüğü aşamadır. Bu aşamada yazılım çalışmalarının yanı sıra kullanıcıya ulaştırılması sürecindeki bütün çalışmaları kapsar. Birleştirme ve sistem testi aşaması, yazılımın kodlaması bittikten sonra gerçekleştirilen doğrulama ve sınama aşamasıdır. Yapılmış olan yazılımın belirlenen özellikleri sağlayıp sağlamadığı ve yazılımın beklentilere uygun olup olmadığını kontrol etmek için statik ve dinamik sınama tekniklerinden faydalanılır. Statik teknikler, yazılımın tüm yazılım yaşam döngüsü boyunca gösterimlerin analiz ve kontrolleriyle ilgilenir. Dinamik teknikler ise gerçekleştirilmiş yazılımı içerir. İşlem ve bakım aşaması, bitmiş olan projenin ihtiyaçlar doğrultusunda yeni özelliklerin eklenildiği, hataların giderildiği aşamadır. Proje kullanılmaya başladıktan sonra yazılımın desteklemesini de kapsar. Yazılımın eksikliklerini gidermelerin yapıldığı aşamadır. Çağlayan modeli aşamalarında, bir aşama bitmediği sürece diğer aşamaya geçilmez. Her aşamada dokümantasyon yazılmalıdır ve test işlemleri yapılmalıdır. Aksi durumda yapılan aşama tamamlanamaz. Çağlayan modelinin avantajları, müşteri ve kullanıcılar tarafından da kolay anlaşılabilen adımlardan oluşur. Tekrarlamalar bir sonraki ve bir önceki adımlarda görülür, daha uzak adımlarda tekrarlamaların görülmesi çok nadirdir. Değişiklik yapma süreci yönetilebilir birimlere bölünmüştür. Gereksinim kısmı bittikten sonra sağlam bir temel oluşur. Proje yöneticileri için iş dağılımı yapması kolaydır. Çağlayan modeli aşamaları kolay anlaşılırdır. Çağlayan modelinin analiz ve tasarım aşamaları detaylı olduğundan dolayı bu adımlar uzun sürmektedir. Ancak kodlama ve test aşamaları kısa sürmektedir. Test aşamasından çıkan hata sayısı oldukça az sayıdadır. Çağlayan modeli dezavantajları, çağlayan modelinde üst aşamalarda yapılan hatalar yüzünden oldukça zaman kaybına yol açmaktadır. Ürünün ortaya çıkması için tüm aşamaların tamamlanmasını beklemek zorunda olunması diğer aşamaya geçilmeme riskini attırır. Projeyi bitirmek için belgelendirme kriteri vardır. Sistem geliştirilmesi süresince gereksinimler sürekli değişir. Çağlayan modeli gereksinimlerin çok iyi anlaşılabilir olduğu sürece kullanılmalıdır. İki ya da daha önceki aşamalara gitmek oldukça maliyetlidir. Bu nedenle gerektiğinde tüm aşamaları yeniden yapmak zordur. Karmaşık ve nesne yönelimli projeler için kullanımı uygun değildir. Devam eden ve uzun projelerde kullanımı yetersiz kalmaktadır. Değişime ve gelişime açık bir model değildir.

**V modeli**, çağlayan modeline göre daha gelişmiş yapıya sahiptir. Süreç adımlarını doğrusal bir şekilde ele almaz. Bunun yerine süreç adımlarının kodlama aşamasından sonra yukarıya doğru eğim alır ve V şeklini oluşturur. Yaşam döngüsü aşamaları arasındaki her bir ilişkiyi gösterir. Belirsizliklerin az olduğu ve iş tanımların belirgin olduğu BT projeleri için uygundur. Yapımı maliyetlidir. Sistem üzerinde değişiklik yapmak zordur. Yapımı uzun zaman alır. V modelinin temel çıktıları Kullanıcı modeli, geliştirme süresi boyunca kullanıcı ille ilişkileri tanımlamaktadır ve sistemin nasıl kabul edileceğine yönelik sınamalar ve planları ortaya koyar. Mimari model; sistem tasarımını ve oluşacak alt sistem ile tüm sistemin sınanma işleminin yapıldığı temel aşamadır. Gerçekleştirim model, Yazılım modellerin kodlanması ve sınanmasına ilişkin fonksiyonlardır. Belirsizliklerin az olduğu ve iş tanımların belirgin olduğu BT projeleri için uygundur. V modeli avantajları, Verification​​ ve​​ validation​​ planları erken aşamalarda belirlenir. Verification​​ ve validation teslim edilebilir tüm ürünlerde uygulanabilir. Proje yönetimini takip etmek kolaydır. Bu modelin kullanımı kolaydır. V modeli dezavantajları, aşamalar arasında tekrarlama yapılmaz. Aynı anda gerçekleşecek olaylara gerçekleşemez. Risk çözümlemeyle ilgili aktivite içermemektedir. Sistemi geliştirdikçe gereksinimler değişiklik gösterebilir.

**Evrimsel geliştirme modeli,** ilk tam ölçekli modeldir. Banka sistemleri gibi çok birimli organizasyonlarda kullanılan modeldir. Her aşamanın sonunda çıkan ara ürün, üretilen alan için tam işlevselliğe sahiptir. Yazılım gereksinimleri başlangıç aşamasında belirlenir. Maliyeti düşüktür. Karmaşık yapıya sahiptir. Sonradan sistem üzerinden değişiklik yapmak zordur. Yapımı uzun zaman alır. Yeniden kullanılabilirliği düşüktür. Evrimsel geliştirme modeli iki çeşidi vardır. Keşifçi geliştirme, buradaki hedef müşterinin problemlerini inceledikten sonra son ürünü teslim etmektir. İyi anlaşılmış problemleri inceler. Atılacak prototipleme, buradaki hedef sistemin problemlerini belirlemektir. İyi anlaşılmamış problemlerle başlar. Evrimsel geliştirme modelinin avantajları*,* sistemi kullanacak kişilerin problemlerini daha iyi kavramasını sağlar. Evrimsel geliştirme modelinin dezavantajları, düzenli bir şekilde teslim edilebilir ürün yoktur. Bakımı oldukça zordur. Sürekli yapılan güncellemeler sistemin yapısını bozar. Evrimsel geliştirme modelinin uygulanabileceği alanlar; kıza zamanlı projelerde, kullanıcı ara yüzü gibi büyük projelerin parçalarında, 500 LOC dan daha az olan küçük ve orta boyutlu sistemlerde kullanılır. Örnek olarak banka uygulamaları, belirli zaman aralıklarında eski sistemin güncellemesinin yapılması.

**Prototipleme modeli,** prototipleme de ilk olarak problemleri belirleyerek başlar. Problemler netleştikçe prototipi düzeltilir. Müşteri sistemden memnun olana kadar düzeltmelere devam edilir. Aşamaları döngüsel şeklinde devam eder. Bu modelde tasarım yapıldıktan sonra geliştirme süreci başlar. Belirli sıklıkla gereksinimler belirlenir. Maliyeti yüksektir. Sonradan sistem üzerinde değişiklik yapmak kolaydır. Yeniden kullanılabilirliği mümkündür. Yapımı uzun zaman alır. Prototipleme modelinin avantajları, sistemi kullanan kişi gereksinimleri görebilir. Yanlış anlaşılmaları engeller. Beklenilmeyen ve yeni problemleri belirler. Risk kontrolünü sağlar. Prototipleme modelinin dezavantajları, belgelendirilmesi olmayan prototiplerin oluşması. Düzeltme aşaması atlanırsa hataların çıkmasına sebep olur.

**Helezonik model,** belirli aralıklarla yazılım gereksinimleri belirlenir. Yapımı maliyetlidir ama başarı garantisi yüksektir. Bu modelin yapımı için uzmanlık gerekliliği vardır. Karmaşık bir yapıya sahiptir. Sonradan sistem üzerinde değişiklik yapılabilir. Yeniden kullanılabilirliği vardır. Yapımı uzun zaman alır. Helezonik model diğer bir ismiyle spiral model dört bölümden oluşur. Bu bölümler; planlama, risk yönetimi, üretim ve kullanıcı değerlendirmedir. Planlama, bu aşamada amaçlar belirlenir. Bir önceki adımda oluşan ürün ile birleştirmek işleminin yapılmasıdır. Risk analizi, risk analizleri yapılır ve alternatif fikirler değerlendirildiği aşamadır. Üretim, ürünün üretildiği ve geliştirildiği aşamadır. Kullanıcı değerlendirme, kullanıcılar ürünü değerlendirdiği aşamadır. Helezonik modelde riskler göz önünde tutulur. Her aşamanın önünden risk analizleri yapılır. Planlar, alternatifler belirlenir. Her aşama sonunda ara ürün belirlenir ve sonraki aşamanın planları yapılır. Bu modelde tasarım, bakım gibi aşamalar yoktur. Her döngü bir aşama sayılır. Helezonik geliştirme, Süreç art arda devam ederse spiral aktiviteler şeklinde gösterilir. Spiraller üzerindeki her halka bir aşamayı gösterir. Helezonik modelin avantajları, kullanıcılar sistemin gidişatını önceden görebilir. Ürünün gelişim aşamalarını küçük parçalara böler ve en riskli hangisiyse onunla başlar. Birçok yazılım modelini içine kapsar. Hatalar erken fark edilir. Helezonik modelin dezavantajları, riski düşük ve küçük projeler için maliyetlidir. Karmaşık bir yapıya sahiptir. Büyük projeler için kullanımı uygundur. Spiralleri sonsuza kadar gidebilir. Sistemi içten gelişimini sağlar. Öznel risk değerlendirilmesi yapılır.

**Formal sistem geliştirme modeli,** yazılımın gerçekleştirme ve tasarım aşamalarının matematiksel ifadeleri kullanarak çalışabilir programlamaya dönüştürülmesidir.Bu modelin temel amacı programı geliştirmektir. Kullanıcı sistemi kullanmaya başladığında karşısına çıkan hataları en düşük miktara indirir. Sistemi artırımlarla gelişimini sağlar. Devamlı tümleştirme vardır. Projeye başlamadan önce yazılım gereksinimleri belirlenir. Maliyeti yüksektir. Alanında uzmanlaşmış kişiler tarafından yapılabilir. Karmaşık bir yapıya sahiptir. Sonradan sistem üzerinde değişiklik yapmak zordur. Yapımı aşaması çok zaman alır. Formal sistem geliştirme modelinin avantajları, yazılımdaki belirsizlikleri, eksikleri belirler. Hatasız bir şekilde yazılım geliştirme modeli ortaya koyar. Karmaşık yapıya sahip değildir. Formal sistem geliştirme modelinin dezavantajları, maliyeti yüksek bir yöntemdir. Çok zaman alır. Sistem kullanılmaya başlanıldığında teknik olmayan personelle iletişim zorlaşır. Bu modelin temel bilgilerini az sayıda geliştirici sahip olduğu için yaygın eğitim gerekir.

**Yeniden kullanılmaya yönelik geliştirme modeli,** dışarıdan temin edilmiş veya daha önceden yazılmış olan projelerin geliştirilmesidir. Sistem gereksinimleri başlangıç aşamasında belirlenir. Maliyeti düşük ve başarı garantisi yüksek bir modeldir. Karmaşık bir yapıya sahiptir. Belirli bir süre sonra yeniden kullanılabilir. Yapım aşaması uzun zaman alabilir. Yeniden kullanılmaya yönelik geliştirme döngüsünün adımları sırasıyla şu şekildedir; gereksinim belirleme, bileşen analizi, gereksinim düzeltmeleri, yeniden kullanımla sistem tasarımı, geliştirme ve tümleştirme, gereksinim düzeltmeleri Yeniden kullanılmaya yönelik geliştirme modelinin avantajları, kaynaklar kontrol edilebilir. Maliyetini denetlemek mümkündür. Bu model basittir ve anlaması kolaydır. Sistemi kısa süre içerisinde geliştirilebilir. Yeniden kullanılmaya yönelik geliştirme modelinin dezavantajları, maliyeti yüksektir. Uzmanlık gerektirir. Problemleri anlamak zordur. Başarı garantisi yoktur.

**Artımlı geliştirme modeli,** müşterilerin ihtiyaç duyduğu yenilikleri sağlar. Belirli zaman aralıklarında sistemin belirli kısımlarını geliştirmeye, yenilikler eklemeye çalışıp teslim etmeye odaklı modeldir. Her yeni gelen kesim sisteme bir şeyler katmaya, geliştirmeye çalışır. Değişiklikler göz önünde bulundurulmaz. Kısıtlı sayıda çalışan bulunur. Belirli aralıklarla yazılım gereksinimleri belirlenir. Maliyeti düşük ve başarı garantisi yüksektir. Orta düzey basitliğe sahiptir. Sonradan sistem üzerinde değişiklik yapmak kolaydır. Yapım aşaması uzun zaman alabilir. Artımlı geliştirme modelinin avantajlar, yazılım için gerekli olan gereksinimler müşteri ile birlikte belirlenir. En önemli gereksinimi karşılayacak sistem kurulur. Tüm sistemin başarısız olma olasılığını azaltır. Sistem özellikleri daha fazla test edilebilir. Artımlı geliştirme modelinin dezavantajları, artırımları anlamak için tüm sistemi anlamak gerekir. Deneyimli, alanında uzman personel gerektirir. Artırımlar kendi içlerinde tekrar kullanılamazlar. Gereksinimleri aynı boyuttaki artırımlara atamak zor olabilir.

**Birleşik süreç modeli**, bu modele başlamadan ilk önce gereksinimler belirlenir. Başarı garantisi yüksek bir modeldir ancak maliyeti oldukça yüksektir. Karmaşık yapıya sahiptir. Sonradan değişiklik yapmak mümkündür. Yeniden kullanılabilirliği vardır. Uzun zaman alan bir modeldir. Başlangıç, ayrıntılandırma, tamamlama, yayım olmak üzere dört aşamadan oluşur. Başlangıç, fizibilite çalışmalarının yapıldığı, vizyon kararının alındığı aşamadır. projeye, devam edilip edilmeyeceğine de bu aşamada karar verilir. Ayrıntılandırma, gerçekçi çözümlerin oluşturulduğu, riskli olan yerlerin yinelemeli olarak oluşturulduğu aşamadır. Tamamlama, düşük riskli olan kısımların yinelemeli olarak oluşturulduğu aşamadır. Yayım, piyasaya sürme işlemidir. Birleşik süreç modeli avantajları, istek değiştirilmesine uyum sağlar. Büyük projeleri kolay bir şekilde çözümler. Oluşabilecek riskleri önceden belirler ve erkenden giderilir. Ürünü erken elde edilir. Fazla zaman almayan bir modeldir. Birleşik süreç modeli dezavantajları, karmaşık yapıya sahiptir. Riskleri yönetmek zordur. Maliyeti yüksektir.

**Çevrik model,** bu modelleme biçimi, prensipleri, değerleri sayesinde diğer geleneksel modellere göre daha kullanışlıdır. Ara yüzü modeli gibi modelleme tekniklerin nasıl kullanılacağını belirtir. Bu metodun kullanılacağı bazı durumlar şunlardır: müşterinin fikir değişikliğinin tahmin edilemez olması durumunda kullanılır. Analiz, tasarım, test etme aşamalarının ne kadar süreceği bilinmediği zaman kullanılır. Belirli aralıklarla yazılım gereksinimleri belirlenir. Maliyeti yüksek bir modeldir ama başarı garantisi yüksektir. Alanında uzmanlaşmış eğitim almış kişiler tarafından kullanılır. Karmaşık bir yapıya sahiptir. Sonradan sistem üzerinde değişiklik yapmak zordur. Belirli bir zaman sonra yeniden kullanılabilirliği vardır. Fazla zaman almayan modeldir. Çevrik model prensipleri; teslim gününün kısa tutup müşteriyi memnun etmek, koşullar değişse bile hemen adapte olmak, teknik başarının sağlanması, yapılan projenin güvenilir olması, iletişime önem verilip görüşlerin yüz yüze yapılması, sonradan eklenen gereksinimlerin dikkate alınması, modelin kolaylığı. Çevrik model avantajları, bu model için eğitim gerekmez. Adapte olmak hızlıdır. Ayrıntılı plan yerine iterasyon planları yapılır. Değişime, yeniliğe açık bir modeldir. Çevrik model dezavantajları, kurumsal yapıya uygulanamaz. İhtiyaçlar sürekli değiştiği için daha çok çalışma gerektirir. Takım üzerinde hedefe ulaşma baskısı oluşur. Projenin başarısı kötüye giderse sistemi yapan kişinin kariyerini de kötü etkiler.

**Kodla-düzelt modeli,** bu model ortada bir ürün yokken fikir üretilerek başlar ve ürün hazır hale gelene kadar devam eder. Maliyetli yöntemdir. Yapımı uzun zaman alır. Başarı garantisi düşük bir modeldir. Sonradan sistemde değişiklik yapılabilir. Başlangıç aşamasında gereksinimler belirlenir. Bu modeli kullanmak içi alanında uzman olmak değil ama belirli bir eğitim alınmış olası gerekmektedir. Belirli bir zaman sonra yeniden kullanılabilirliği düşük bir modeldir. Kodla-düzelt modelinin avantajları, önceden planlamaya ihtiyaç duymaz. Temel aşamalar hızlı bir şekilde geçilir. Küçük projelerde kullanılır. Bu modeli herkes kullanabilir. Kodla-düzelt modelinin dezavantajları, planlama yapılmaz. Yapılan hataların bulunması zordur. Bu nedenle kodların düzeltilmesi maliyetli olabilir. Kodlar kullanıcının ihtiyacını karşılamayabilir. Sonradan kodları değiştirmek zordur. Bitirme süresi belli değildir.

SCRUM NEDİR? SCRUM yazılım metotlarıyla uygulanabilecek proje yönetimi yaklaşımıdır. Karmaşık yapıdaki yazılımları küçük parçalara ayırarak geliştirmeye çalışır. Yönetimi ilk bakışta basit kurallara sahiptir. Gereksinimleri açıkça belirli değildir. Karmaşık yapıya sahip yazılımlar için uygundur. SCRUM sayesinde sürecin aksayan yönlerini çözüm bulunur. Günlük toplantılarda çalışanlar birbirlerinden sürekli haberdar olurlar. Böylece projede kopukluklar meydana gelmez. Başarılı yazılım geliştirme yöntemidir. Google, Microsoft, Facebook gibi ünlü şirketlerinde kullanılmaktadır.

Product Backlog: Müşteri ile görüşülüp gereksinim listesinin oluşturulduğu aşamadır. Sprint Backlog: Proje hazırlama aşamasıdır. 15-30 gün sürer. Scrum Daily Meeting: Ekibin günlük yaptığı toplantı aşamasıdır. Günlük toplantılarda ekip “dün ne yaptım? ““bugün ne yapacağım?” ve “engelleyen sorunlar var mı?” bu üç soruyu cevaplarlar.

SCRUM Temel Kavramları SCRUM üç temel kavramdan oluşur. Bunlar \*Roller: SCRUM yöneticisi, SCRUM takımı ve ürün sahibidir. \*Toplantılar Sprint Plannig Toplantıları, çalışanlar bir sonraki aşamada hangi işler yapacağını belirlerler. Belirlenen bu iş için maksimum süre bir aydır. Sprintte ne miktarda iş yapılacağı ekip karar verirler. Sprint Review Toplantıları, sprint süresi boyunca yapılmış olan çalışmalar konuşulur. Buradaki amaç iş birliğini sağlamaktır. Sprint Retrospective Toplantısı, Sprint çalışmaları konuşma, gözden geçirme toplantısı ile sprint planlamanın yapıldığı toplantı arasında yapılan toplantı çeşididir. \*Bileşenler Ürün Gereksinim Dokümanı = Yapılması gereken işlerin listesidir. Geçerli ve kullanışlı olması için devamlı bakım yapılması gerekmektedir. Bazı gereksinimler temizlenebilir. Sprint Dokümanı = product backlog’dan elde edilen görevleri içerir. Sprint dokümanı sadece çalışanlar tarafından değiştirilir. Sprint Kalan Zaman Grafiği= İterasyon boyunca işlerin ne kadar yapıldığı ile ne kadar yapılacağını gösterir. Böylece karşılaştırılmış olunur.

KAYNAKÇA

\* <https://www.startupnedir.com/scrum-nedir-2/>

\* <https://fikirjeneratoru.com/yazilim-proje-yonetimi-yontemleri/>

\* <https://www.codex.com.tr/yazilim-gelistirme-modelleri>

\* <http://www.aspmvcnet.com/tr/m/yazilim-muhendisligi/helezonik-tasarim-spiral-tasarim.html>

\* <http://web.firat.edu.tr/mbaykara/ytm2.pdf>