Araştırma (Bilgi toplayanlar)  
Filtreleme (Toplanan bilgileri farklı gruplara ayırma)  
Sentezleme (Farklı gruplardaki bilgileri birleştirerek bir yapı oluşturma)  
Yazılım Geliştirme (İstenilen yazılımın 3.adımda oluşturulan yapıya uygun geliştirilmesi)

En alt katmandan sürekli bilgi güncellemesi yapılır (4 hafta boyunca)

yazılım ürünleri,çevik yazılım, özellikler senaryo ve hikayeler (3.kısım), yazılım mimarisi, bulut temelli yazılım, mikroservis mimarileri, güvenlik ve gizlilik

1.SLAYT KEYLER - YAZILIM ÜRÜNLERİ

• Yazılım ürünleri, geniş bir müşteri yelpazesi için yararlı olması muhtemel genel işlevleri içeren yazılım sistemleridir.

• Ürün yazılım mühendisliğinde, ürünün parçası olması gereken özelliklerin belirlenmesinden ve bu özelliklerin uygulanmasından aynı firma sorumludur.

• Yazılım ürünleri, müşterinin bilgisayarlarında çalışan bağımsız sistemler, hibrit sistemler veya hizmet tabanlı sistemler olarak teslim edilebilir. Hibrit sistemlerde, bazı özellikler yerel olarak uygulanır ve diğerlerine İnternet üzerinden erişilir. Hizmet tabanlı ürünlerde tüm ürün özelliklerine uzaktan erişilir.

• Bir ürün vizyonu, neyin geliştirileceğini, ürün için hedef müşterilerin kimler olduğunu ve geliştirmekte olduğunuz ürünü neden satın almaları gerektiğini kısa ve öz bir şekilde açıklamalıdır.

• Etki alanı deneyimi, ürün deneyimi, müşteri deneyimi ve deneysel bir yazılım prototipinin tümü, ürün vizyonunun geliştirilmesine katkıda bulunabilir.

Ürün yöneticilerinin temel sorumlulukları, ürün vizyon sahipliği, ürün yol haritası geliştirme, kullanıcı hikayeleri ve ürün biriktirme listesi oluşturma, müşteri ve kabul testi ve kullanıcı arayüzü tasarımıdır.

• Ürün yöneticileri işletme, yazılım geliştirme ekibi ve ürün müşterileri arasındaki arayüzde çalışır. Bu gruplar arasındaki iletişimi kolaylaştırırlar.

• Kendi fikirlerinizi geliştirmek ve planlanan ürün özelliklerini potansiyel müşterilere göstermek için her zaman bir ürün prototipi geliştirmelisiniz.

2.SLAYT KEYLER - ÇEVİK YAZILIM MÜHENDİSLİĞİ

Yazılım ürünleri geliştirmenin en iyi yolu, hızlı ürün geliştirme ve teslim etmeye yönelik çevik yazılım mühendisliği yöntemlerini kullanmaktır.

Çevik yöntemler, yinelemeli geliştirme ve geliştirme süreci sırasında genel giderlerin en aza indirilmesi üzerine kuruludur.

• Ekstrem programlama (XP), kullanıcı hikayeleri, önce test geliştirme ve sürekli entegrasyon gibi çevik geliştirme uygulamalarını tanıtan etkili bir çevik yöntemdir. Bunlar artık ana akım yazılım geliştirme faaliyetleridir.

Scrum, çevik planlama ve yönetime odaklanan çevik bir yöntemdir.

XP'den farklı olarak, kullanılacak mühendislik uygulamalarını tanımlamaz. Geliştirme ekibi, geliştirilmekte olan ürün için uygun olduğuna inandıkları her türlü teknik uygulamayı kullanabilir.

• Scrum'da, yapılacak iş, bir ürün biriktirme listesinde (tamamlanacak iş öğelerinin bir listesi) tutulur. Yazılımın her artışı, ürün biriktirme listesindeki bazı iş öğelerini uygular.

Sprintler, bir ürün artışının geliştirildiği sabit zamanlı aktivitelerdir (genellikle 2-4 hafta). Artımlar 'potansiyel olarak sevk edilebilir' olmalıdır, yani teslim edilmeden önce daha fazla çalışmaya ihtiyaç duymamalıdırlar.

• Kendi kendini organize eden bir ekip, yapılacak işi ekip üyeleri arasında tartışma ve anlaşma yoluyla organize eden bir geliştirme ekibidir.

• Ürün biriktirme listesi, sprintler ve kendi kendini organize eden ekipler gibi Scrum uygulamaları, Scrum'ın diğer yönleri kullanılmasa bile herhangi bir çevik geliştirme sürecinde kullanılabilir.

3.SLAYT KEYLER- ÖZELLİKLER SENARYOLAR VE HİKAYELER

Bir yazılım ürünü özelliği, bir kullanıcının ürünü kullanırken ihtiyaç duyabileceği veya isteyebileceği bir şeyi uygulayan bir işlevsellik parçasıdır.

• Ürün geliştirmenin ilk aşaması, her bir özelliği tanımladığınız ve işlevselliğinin kısa bir tanımını verdiğiniz ürün özelliklerinin listesini belirlemektir.

• Kişiler, ürününüzü kullanabileceğini düşündüğünüz bir kullanıcı türünün karakter portresini oluşturduğunuz 'hayali kullanıcılar'dır.

• Bir karakter açıklaması, tipik bir ürün kullanıcısının 'resmini' çizmelidir. Eğitim geçmişlerini, teknoloji deneyimlerini ve neden ürününüzü kullanmak isteyebileceklerini açıklamalıdır.

• Senaryo, kullanıcının yapmak istediği bir şeyi yapmak için ürün özelliklerine eriştiği bir durumu açıklayan bir anlatımdır.

Senaryolar her zaman kullanıcının bakış açısıyla yazılmalı ve tanımlanmış kişilere veya gerçek kullanıcılara dayanmalıdır.

• Kullanıcı öyküleri, bir kullanıcının bir yazılım sisteminden istediği şeyi yapılandırılmış bir şekilde ortaya koyan daha ayrıntılı anlatılardır.

• Kullanıcı öyküleri, bir senaryoyu genişletmenin ve ayrıntı eklemenin bir yolu olarak veya sistem özelliklerinin açıklamasının bir parçası olarak kullanılabilir.

• Özellik belirleme ve tasarımdaki kilit etkiler, kullanıcı araştırması, etki alanı bilgisi, ürün bilgisi ve teknoloji bilgisidir.

• Bu anlatılarda kullanıcı eylemlerini öne çıkararak ve bu eylemleri desteklemek için ihtiyaç duyduğunuz özellikleri düşünerek senaryolardan ve öykülerden özellikleri belirleyebilirsiniz.

4. SLAYT KEYLER - YAZILIM MİMARİSİ

• Yazılım mimarisi, bileşenlerinde, bunların birbirleriyle ve çevreyle olan ilişkilerinde ve tasarımına ve evrimine rehberlik eden ilkelerde somutlaşan bir sistemin temel organizasyonudur.

• Bir yazılım sisteminin mimarisi, güvenilirlik, verimlilik ve güvenlik gibi işlevsel olmayan sistem özellikleri üzerinde önemli bir etkiye sahiptir.

• Mimari tasarım, ürününüz için kritik olan konuları anlamayı ve bileşenleri ve bunların ilişkilerini gösteren sistem açıklamaları oluşturmayı içerir.

• Mimari açıklamaların temel rolü, geliştirme ekibinin sistem organizasyonunu tartışması için bir temel sağlamaktır. Gayri resmi mimari diyagramlar, hızlı ve çizilmesi ve paylaşılması kolay olduğu için mimari tanımlamada etkilidir.

• Sistem ayrıştırma, mimari bileşenleri analiz etmeyi ve bunları daha ince taneli bileşenler kümesi olarak temsil etmeyi içerir.

Karmaşıklığı en aza indirmek için endişeleri ayırmalı, işlevsel tekrardan kaçınmalı ve bileşen arayüzlerine odaklanmalısınız.

• Web tabanlı sistemler genellikle kullanıcı arayüzü katmanları, uygulamaya özel katmanlar ve bir veritabanı katmanı içeren ortak bir katmanlı yapıya sahiptir.

• Bir sistemdeki dağıtım mimarisi, o sistemdeki sunucuların organizasyonunu ve bileşenlerin bu sunuculara tahsisini tanımlar.

• Çok katmanlı istemci-sunucu ve hizmet odaklı mimariler, web tabanlı sistemler için en yaygın kullanılan mimarilerdir.

• Veritabanı ve bulut teknolojileri gibi teknolojiler hakkında karar vermek, mimari tasarım sürecinin önemli bir parçasıdır.

5.SLAYT KEYLER - BULUT TABANLI YAZILIM

Bulut, kendi kullanımınız için kiralayabileceğiniz çok sayıda sanal sunucudan oluşur. Siz ve müşterileriniz bu sunuculara internet üzerinden uzaktan erişir ve kullanılan sunucu süresi kadar ödeme yaparsınız.

• Sanallaştırma, birden çok sunucu örneğinin aynı fiziksel bilgisayarda çalıştırılmasına izin veren bir teknolojidir. Bu, bulutta dağıtım için yazılımınızın yalıtılmış örneklerini oluşturabileceğiniz anlamına gelir.

• Sanal makineler, kendi işletim sisteminizi, teknoloji yığınınızı ve uygulamalarınızı çalıştırdığınız fiziksel sunucu kopyalarıdır.

• Konteynerler, sanal sunucuların hızlı bir şekilde çoğaltılmasına ve konuşlandırılmasına izin veren hafif bir sanallaştırma teknolojisidir. Tüm konteynerler aynı işletim sistemini çalıştırır. Docker şu anda en yaygın kullanılan konteyner teknolojisidir.

• Bulutun temel bir özelliği, 'her şeyin' bir hizmet olarak sunulabilmesi ve internet üzerinden erişilebilmesidir. Bir hizmet, sahip olunmak yerine kiralanır ve diğer kullanıcılarla paylaşılır.

Hizmet olarak altyapı (laaS), bilgi işlem, depolama ve diğer hizmetlerin bulut üzerinden sunulduğu anlamına gelir. Kendi fiziksel sunucularınızı çalıştırmanıza gerek yoktur.

• Hizmet olarak platform (PaaS), yazılımınızı talebe göre otomatik olarak ölçeklendirmeyi mümkün kılmak için bir bulut platformu satıcısı tarafından sağlanan hizmetlerin kullanılması anlamına gelir.

• Hizmet olarak yazılım (SaaS), uygulama yazılımının kullanıcılara bir hizmet olarak sunulması anlamına gelir. Bunun, daha düşük sermaye maliyetleri gibi kullanıcılar ve yeni yazılım sürümlerinin daha basit dağıtımı gibi yazılım satıcıları için önemli faydaları vardır.

• Çok kiracılı sistemler, tüm kullanıcıların çalışma zamanında bireysel ihtiyaçlarına göre uyarlanabilen aynı veritabanını paylaştığı SaaS sistemleridir. Çok eşgörünümlü sistemler, her kullanıcının kendi ayrı veritabanına sahip olduğu SaaS uygulamalarıdır.

• Bulut tabanlı yazılım için temel mimari konular, kullanılacak bulut platformu, çok kiracılı mı yoksa çok eşli veritabanı mı kullanılacağı, ölçeklenebilirlik ve dayanıklılık gereksinimleri ve sistemde temel bileşenler olarak nesnelerin veya hizmetlerin kullanılıp kullanılmayacağıdır.

6.SLAYT KEYLER - MİKROSERVİS MİMARİSİ

Mikro hizmet, kendi sürecinde çalışan ve hafif protokoller kullanarak diğer mikro hizmetlerle iletişim kuran bağımsız ve bağımsız bir yazılım bileşenidir.

• Bir sistemdeki mikro hizmetler, farklı programlama dilleri ve veritabanı teknolojileri kullanılarak uygulanabilir.

• Mikroservislerin tek sorumluluğu vardır ve sistemdeki diğer mikroservisleri değiştirmeye gerek kalmadan kolayca değiştirilebilecek şekilde tasarlanmalıdır.

• Mikro hizmet mimarisi, sistemin iletişim kuran mikro hizmetlerden oluşturulduğu bir mimari tarzdır. Her mikro hizmetin kendi kapsayıcısında çalışabileceği bulut tabanlı sistemler için çok uygundur.

• Bir mikro hizmet sisteminin mimarlarının en önemli iki sorumluluğu, sistemin mikro hizmetlere göre nasıl yapılandırılacağına karar vermek ve mikro hizmetlerin nasıl iletişim kurması ve koordine edilmesi gerektiğine karar vermektir.

İletişim ve koordinasyon kararları, mikro hizmet iletişim protokollerine, veri paylaşımına, hizmetlerin merkezi olarak koordine edilip edilmeyeceğine ve hata yönetimine karar vermeyi içerir.

• RESTful mimari stili, mikro hizmet tabanlı sistemlerde yaygın olarak kullanılmaktadır. Hizmetler, GET, POST, PUT ve DELETE HTTP fiillerinin hizmet işlemleriyle eşleneceği şekilde tasarlanmıştır.

•RESTful stili, bir mikro hizmet mimarisinde XML veya daha genel olarak JSON kullanılarak temsil edilebilen dijital kaynaklara dayalıdır.

• Sürekli dağıtım, bir hizmet değişikliği yapılır yapılmaz hizmetin yeni sürümlerinin üretime alındığı bir süreçtir. Yeni sürümün "üretim kalitesinde" olup olmadığını kontrol etmek için otomatik testlere dayanan tamamen otomatik bir süreçtir.

• Sürekli dağıtım kullanılıyorsa, yeni dağıtılan bir hizmette sorunlar bulunursa daha eski bir sürüme geçebilmek için dağıtılan hizmetlerin birden çok sürümünü korumanız gerekebilir.

7.SLAYT KEYLER GÜVENLİK VE GİZLİLİK

Güvenlik, bir yazılım sisteminin kullanılabilirliğini, sistemin ve/veya verilerinin bütünlüğünü ve gizli bilgilerin çalınmasını tehdit edebilecek kötü niyetli saldırılara karşı kendini koruma yeteneği ile ilgili teknik bir kavramdır.

• Yazılım ürünlerine yönelik yaygın saldırı türleri arasında enjeksiyon saldırıları, siteler arası komut dosyası çalıştırma saldırıları, oturumu ele geçirme saldırıları, hizmet reddi saldırıları ve kaba kuvvet saldırıları yer alır.

• Kimlik doğrulama, kullanıcının bildiği bir şeye, kullanıcının sahip olduğu bir şeye veya kullanıcının bazı fiziksel özelliklerine dayalı olabilir.

• Birleştirilmiş kimlik doğrulama, kimlik doğrulama sorumluluğunu Facebook veya Google gibi bir üçüncü tarafa veya bir işletmenin kimlik doğrulama hizmetine devretmeyi içerir.

• Yetkilendirme, kullanıcının doğrulanmış kimliğine dayalı olarak sistem kaynaklarına erişimi kontrol etmeyi içerir. Erişim kontrol listeleri, yetkilendirmeyi uygulamak için en yaygın kullanılan mekanizmadır.

• Simetrik şifreleme, bilgilerin aynı gizli anahtarla şifrelenmesini ve şifresinin çözülmesini içerir. Asimetrik şifreleme, bir anahtar çifti kullanır - bir özel anahtar ve bir genel anahtar. Genel anahtar kullanılarak şifrelenen bilgilerin şifresi yalnızca özel anahtar kullanılarak çözülebilir.

Simetrik şifrelemede önemli bir sorun anahtar değişimidir. Web trafiğini güvence altına almak için kullanılan TLS protokolü, paylaşılan bir anahtar oluşturmak için kullanılan bilgileri aktarmak için asimetrik şifreleme kullanarak bu sorunun üstesinden gelir.

• Ürününüz hassas kullanıcı verilerini saklıyorsa, bu verileri kullanımda değilken şifrelemeniz gerekir.

• Bir anahtar yönetim sistemi (KMS), şifreleme anahtarlarını depolar. Bir işletmenin binlerce hatta milyonlarca anahtarı yönetmesi ve eski bir şifreleme anahtarı kullanılarak şifrelenmiş geçmiş verilerin şifresini çözmesi gerekebileceğinden, bir KMS kullanmak önemlidir.

• Gizlilik, insanların kişisel bilgilerinin başkalarına açıklanması konusunda ne hissettikleri ile ilgili sosyal bir kavramdır. Farklı ülkeler ve kültürler, hangi bilgilerin gizli olması ve olmaması gerektiği konusunda farklı fikirlere sahiptir.

• Kişisel mahremiyeti korumak için birçok ülkede veri koruma yasaları çıkarılmıştır.

Kullanıcı verilerini yöneten şirketlerin bunları güvenli bir şekilde saklamasını, kullanıcıların izni olmadan kullanılmamasını veya satılmamasını ve kullanıcıların sistem tarafından tutulan kişisel verileri görüntülemesine ve düzeltmesine izin vermesini şart koşuyorlar.