PROGRAMLAMADA YENİ EĞİLİMLER

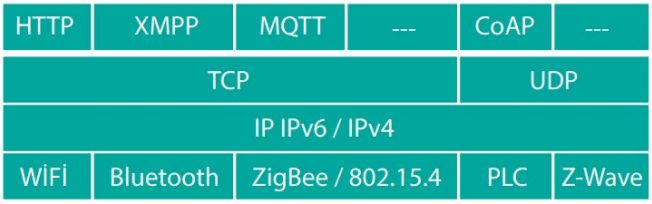
- **Çevik yazılım geliştirme yöntemleri**, değişikliklere daha hızlı adapte olmayı hedefleyerek yazılım projelerinde daha başarılı bir yöntem olarak öne çıkmaktadır. Ayrıca çevik yöntemler, yazılımı talep edenler ile yazılımı geliştirenler arasında sürekli iletişim ve ortak hedef oluşturmayı gerektirdiğinden, tüm paydaşların memnuniyetini de arttırmaktadır. Bu yöntemlerden en çok kullanılanı Scrum'dır.
- **Scrum**, öğrenmesi kolay ve esnek bir çevik yazılım geliştirme yöntemidir. En temel özellikleri **şeffaf, gözleme dayalı ve tekrarlamalı** (iterative) olmasıdır. Yazılımda değer üreten kısımlar daha önce üretilir, ekibin birim zamanından elde edilen fayda miktarı artar.
- **Yazılım geliştirme süreci** temel olarak **tasarım, geliştirme, test** ve **dağıtma** aşamalarını içerir.
- 1980'lerden 2000'li yıllara kadar uygulanan yazılım geliştirme süreçlerinin birçoğu **şelale (waterfall)** yaklaşımını kullanmıştır. Şelale tipi süreçlerde, tüm temel yazılım geliştirme faaliyetleri önceden belirlenen sabit bir zaman aralığında ve belirlenmiş bir sıra ile yapılır. Tüm yazılım geliştirme işlemleri bitmeden **test aşamasına geçilmez**! Gereksinim analizi bitmeden yazılım geliştirilmeye başlanmaz!
- Scrum yönteminin ilk prototipleri 1990'lı yılların başında **Ken Schwaber** tarafından kendi şirketinde denenmiştir. 1995 yılında **Jeff Sutherland** ile birlikte çalışarak "Scrum Software Development Process" (Scrum Yazılım Geliştirme Süreci) isimli bir makale yazarak Oopsla konferansında sunmuştur.
- **Çevik Yazılım Geliştirme Manifestosu:** 1) Süreçler ve araçlardan ziyade bireyler ve etkileşimlere 2) Kapsamlı dokümantasyondan ziyade çalışan yazılıma 3) Sözleşme pazarlıklarından ziyade müşteri ile işbirliğine 4) Bir plana bağlı kalmaktan ziyade değişime karşılık vermeye kanaat getirdik.
- **Çevik Yazılım Geliştirme Yöntemleri:** Adaptive software development (ASD) • Agile modeling • Agile Unified Process (AUP) • Crystal Clear methods • Disciplined agile delivery • Dynamic systems development method (DSDM) • Extreme programming (XP) • Feature-driven development (FDD) • Lean software development • Kanban • Rapid application development (RAD) • Scrum • Scrumban
- **Çevik Yazılım Geliştirme Pratikleri**
  - **Test güdümlü programlama:** Bu yöntemde yazılımcı, kodu doğrudan yazmaya başlamak yerine, öncelikle koddan bekleneni test edecek olan test kod parçacıklarını yazarak işe başlar. Yazılım parçası her defasında ilgili testlere tabi tutulur. Testler başarılı olana kadar bu işlem devam eder.
  - **Kod yeniden yapılandırma:** Mevcut kodun davranışını değiştirmeden yapısal değişiklik yapma faaliyetidir.
  - **Sürekli entegrasyon:** Yazılım üzerinde yapılan değişikliklerin herhangi bir bozulmaya sebep olup olmadığının **önceden belirlenmesini** hedefleyen bir pratiktir.
  - **Eşli programlama:** İki yazılımcının aynı iş üzerinde beraber çalışmasıdır. Yazılımcılardan bir tanesi kodu yazarken, diğeri gözlemleyerek kodu anında gözden geçirir. Yazılımcılar ara ara rol değişikliği de yaparlar. Bu yöntem tek başına çalışmaya göre daha maliyetli olsa da, yapılan araştırmalara göre hata oranını düşürdüğü ve yazılım kalitesini arttırdığı gözlenmiştir.
- **Scrum**, insanların karışık ve adaptasyona açık problemleri ele alabilmek için en yüksek değere sahip ürünü, üretken ve yaratıcı bir şekilde geliştirmesini sağlayan bir iskelettir. Scrum, sadece yazılım ürünü geliştirmek için ortaya konmuş bir yöntem değildir. Herhangi değer üreten bir ürün geliştirmesi için de kullanılabilir. Ancak, en çok yazılım projelerinde uygulanır.
- **Scrum'da** üç temel kavrama önem verilmektedir: 1) Şeffaflık, 2) Gözlem, 3) Adaptasyon.
- **Scrum ekibi**: 1) Ürün Sahibi (Product Owner), 2) Geliştirme Ekibi (Development Team), 3) Scrum Uzmanı'ndan (Scrum Master) oluşur.
  - **Ürün sahibi** tek bir kişi olmalıdır. Yapılan işlerin bittiğini kabul eden kişi Ürün Sahibidir.
  - **Geliştirme Ekibi**, her bir iterasyon sonrasında, **ürünün sürüme çıkabilir bir kısmını** teslim etmekten sorumlu kişilerden oluşur.
  - **Scrum Uzmanı**, Scrum'ın **doğru anlaşılması ve doğru uygulanmasını** sağlayan kişidir. Bunun için Scrum kurallarından, rehberlerden ve pratiklerden faydalanır. Hizmetkâr-lider olarak da adlandırılır.
- **Scrum etkinlikleri** düzenlilik sağlamak ve fayda arttırımı için yapılır.
  Bu etkinlikler süreç içinde yeterli sayıda olmalı ve başka toplantı, etkinlik vb. yapılmamalıdır.
- **Sprint**, belirli bir zaman aralığını temsil eder. Bu aralığın uzunluğu **en fazla bir ay** olabilir. Bir Sprint biter bitmez hemen diğer Sprint başlar, **arada boşluk yoktur**. Sprint dışı bir zaman olmadığı için, diğer Scrum etkinlikleri Sprint içinde yapılır.
  Bunlar: 1) **Sprint Planlama**, 2) **Günlük Scrum**, 3) **Sprint Değerlendirme**, 4) **Sprint Retrospektifi**.
  - **Sprint Planlama:** Sprintin en başında yapılan etkinliktir. Planlamada tüm Scrum Ekibi mevcut Sprint içinde yapılacak işi planlarlar. Bir aylık Sprint için planlama toplantısı en fazla 8 saat, iki haftalık bir Sprint için ise 4 saat olabilir. Toplantıda temel olarak, içinde bulunulan Sprint süresince kullanılabilir ürün parçası olarak neler yapılacağı ve bu ürün parçalarını oluşturmak için yapılacak işlerin nasıl gerçekleştirileceği sorularına cevap aranır.
  - **Günlük Scrum:** Bu toplantı, Geliştirme Ekibinin **son bir gün içinde** yaptığı faaliyetler ve planladığı faaliyetler hakkında bilgi alış verişini yaptığı toplantıdır, en fazla 15 dakika sürmelidir. Toplantı **her gün aynı saatte**, önceden belirlenen bir yerde (tercihen Scrum Ekip Panosu yanında) ve ayakta yapılmalıdır.
  - **Sprint Değerlendirme:** Sprint'in sonunda, ekibin geliştirdiği **ürünü kontrol ettiği** bir toplantıdır.
  - **Sprint Retrospektif:** Sprint içinde en son yapılan etkinliktir. Bu toplantıda Scrum Takımı tüm bir Sprint'in değerlendirmesini yapar. Toplantı **üç saat** ile sınırlandırılmalıdır. Retrospektif toplantısında odaklanılan şey sürekli iyileştirmedir ve temel olarak bir sonraki Sprint sürecinde nelerin daha iyi yapılabileceği tartışılır. Sprint Retrospektif toplantısı ile Sprint biter ve Sprintler arası boşluk olmadığından hemen bir sonraki Sprint için planlama toplantısına başlanır.
- **Kızgın-Üzgün-Mutlu yöntemi**: Retrospektif toplantısında, ekipte herkes o Sprint içinde kendisini kızdıran, üzen ve mutlu eden şeyleri küçük kâğıtlara not eder. Herkes kızgın, üzgün ve mutlu olduğu durumları yazdığı not kâğıtlarını tahtaya yapıştırıp anlattıktan sonra ise ekip iyileştirme yapmak adına alacağı aksiyon planlarını belirler.
  Bu yöntemin bir benzeri de, Yapmaya Başlamalıyız-Yapmayı Bırakmalıyız-Yapmaya Devam Etmeliyiz şeklindedir.
- **Scrum Eserleri ve Çıktıları:** 1) **Ürün İş Listesi**, 2) **Sprint İş Listesi**, 3) **Ürün Parçası**, 4) **Takım Hızı**, 5) **Bitti Tanımı**.

- **Ürün İş Listesi:** Ürünün, önceliğe göre sıralanmış özellik listesidir. Üründen beklenen tüm gereksinimler için tek bilgi kaynağı burasıdır. Ürün Sahibi, Ürün İş Listesinden sorumlu tek kişidir. Listenin sıralaması, içeriği ve erişilebilirliğinden sorumludur. Geri bildirimler, Ürün İş Listesi içinde yer bulurlar. Ürün var olduğu müddetçe Ürün İş Listesi de var olacaktır.
- **Sprint İş Listesi** belirlenen Sprint hedefi doğrultusunda seçilmiş Ürün İş Listesi kalemleridir. Sprint hedefine ulaşmak için gerekli tüm işler mevcuttur. Böylelikle şeffaflık sağlanır. Bu işlerin hepsinin mevcut tahmini, kalan efor bilgileri açık bir şekilde görülebilir durumdadır.
- Scrum Ekiplerinin kullandığı ve Sprint gidişatının bir bakışta özet olarak anlaşılabildiği grafiğe **Aşağı-Tüketim (Burn-Down)** grafiği adı verilir. Bu grafik her Günlük Scrum toplantısında güncellenir.
- **Ürün Parçası:** Sprint sonunda tamamlanan Ürün İş Listesi kalemleri ile daha önce bitirilmiş Sprintlerdeki Ürün Parçalarının değerlerinin toplamıdır. **Her Sprint sonunda bir Ürün Parçası** kullanılabilir olarak hazır ve "Bitti" olarak tanımlanmış bir şekilde ortaya çıkmalıdır.
- **Takım Hızı:** Sprint boyunca yapılan işlerin toplam puanı, ekibin hızı olarak tanımlanır. Birkaç Sprint geçtikten sonra ekibin hız değeri oturacaktır. Bu değer daha sonra planlama toplantılarında yol gösterici olur.
- **Bitti Tanımı:** Ürün İş Listesi kalemleri "Bitti" olarak nitelendirildiğinde tüm ekip aynı şeyi anlamalıdır. Bu bitti tanımının neleri içerdiği önceden belirlenmelidir. Tüm iş kalemlerinin aynı "Bitti" özelliklerine sahip olması ürünün kalitesi açısından çok önemlidir.
- **Nesnelerin İnterneti** kavramı ilk kez **Kevin Ashton** tarafından ortaya atılmıştır. Kevin Ashton radyo frekansı tanımlama (Radio Frequency Identification – **RFID**) üzerinde çalışan bir İngiliz teknoloji öncüsüdür.
- **Ağ Geçidi:** Farklı ağların, farklı teknolojilerin birbirleriyle iletişimini sağlayan ağ donanımlarıdır. Bir başka deyişle, **ağlar arası tercümanlık** görevini üstlenen cihazlardır.
- **Sensör:** Fiziksel ortamdan, yani çevreden gelen çeşitli türdeki girdileri algılayan ve bu algıyı **elektronik sinyallere çeviren** aygıttır. Bu girdiler; ısı, hareket, nem, basınç veya çok sayıda diğer çevresel olgudan herhangi biri olabilir. Çıktı genellikle, sensör aracılığıyla insan tarafından okunabilir gösterime dönüştürülen veya daha karmaşık işlemler için bir ağ üzerinden iletilen elektronik sinyallerdir.
- **Sensör Çeşitleri**
  - **Akustik ve ses sensörleri:** Havada veya denizaltında, **titreşim üreten nesnelerin algılanması**nı sağlar (Mikrofon, hidrofon).
  - **Çevresel sensörler:** Çevresel **meteorolojik değişimleri izlemek** için kullanılırlar (Yağmur sensörü, kar sensörü, nem sensörü).
  - **Kimyasal sensörler:** Gazların, sıvıların veya elementlerin tespiti için kullanılırlar. (pH sensörü, kimyasal gaz sensörleri, CO_2 sensörü)
  - **Yaklaşma ve varlık bildirim sensörleri:** Ortamdaki hareketleri algılamada kullanılırlar (Otomobil park etme yakınlık sensörleri)
  - **Elektrik ve Manyetik sensörler:** Elektrik ve manyetik alanlardaki değişiklikleri algılayabilir. (**Galvanometre**, metal detektörü)
  - **Otomotiv sensörler:** Taşıtlarda kullanılanılır. (Hız ölçücü radar, araç içi ısı sensörleri, benzin uyarı ve lastik basınç sensörleri)
  - **Termal ve ısı sensörleri:** Ortam ısısını ve sıcaklığını algılamak için kullanılırlar (**Termokapıl**, kalorimetre, **gordon** sensörü)
  - **Optik sensörler:** Ortamda bulunan **ışık miktarını** algılamak için kullanılırlar (Foto diyot, fototransistor)
  - **Mekanik sensörler:** Ortam titreşimini, yer değişimleri ölçmek için kullanılmaktadırlar (Yükseklik sensörleri)
  - **Biyosensörler:** Elektrokimyasal ölçüm yaparlar. Sağlık ve besin endüstrisinde kullanılırlar. (Kalp atış sensörleri, kızıl ötesi sensörler)
- **Aktif sensörler**, veri üretebilmek için bir **güç kaynağına ihtiyaç duyar**ken;
  Pasif sensörler ise, doğrudan verileri ortamdaki değişimlerden elde edebilmekte, bir **güç kaynağına ihtiyaç duymamaktadır**.
- **RFID Teknolojisi:** Otomatik tanımlama, nesnelerin belirlenmesi ilkesi ile ortaya çıkmış bir teknoloji olarak ifade edilebilir. RFID, Otomatik Tanımlama ve Veri Yakalama (Automatic Identification and Data Capture – AIDC) için bir yöntemdir. RFID, bir nesnenin **ne olduğunu, nerede olduğunu ve durumunu** kolaylıkla söyleyebilir; bu da Nesnelerin İnterneti teknolojisine önemli bir katkı sağlar.

Header	EPC Maneger Number	Object Class	Serial Number
Önbilgi	EPC İşletme Numarası	Ürün tipi	Seri Numarası
8 bit	28 bit	24 bit	36 bit
96 bit			

- **Gömülü Sistemler:** RFID ve sensörler, ortamdaki değişimleri sinyale dönüştürmektedir. Gömülü sistemler de, **bu sinyalleri veriye çevirir ve internet ağına iletir**. Bağımsız bir sistem olabildiği gibi büyük bir sistemin parçası da olabilir.
- **Gömülü sistemlerin temel bileşenleri**
  - **Donanım:** Genelde analog sinyali sayısal sinyale çeviren A/D çevirici, mikro işlemci/kontrolcü, bellek, arabirimlerin protokollerini işleyen arabirim kontrolörleri (USB, WiFi, Bluetooth vb.) ve sensörlerin bağlanabileceği portlardan oluşmaktadır.
  - **Uygulama Yazılımı:** Sadece donanım aygıtlarının sınırlı ve temel işlevlerini kontrol etmek için kullanılır ve genellikle kullanıcı girişine ihtiyaç duymaz. Yazılım çoğunlukla kullanıcıdan bağımsız işlemekte ve fonksiyonları harici kontrollerle etkinleştirilmektedir.
  - **Gömülü sistemin işletim sistemi:** Bir bilgisayarın donanım ve kaynaklarının kullanıcı programları tarafından paylaşılması ve yönetilmesini sağlar. Etrafımızda gördüğümüz hemen her akıllı cihaz üzerinde koşan bir işletim sistemi bulunmaktadır. Ancak temel görevi, bir ölçümü alıp bunu GPRS, Bluetooth veya WiFi vasıtasıyla bir noktaya iletecek termostat kontrolü yapan bir akıllı modül için akıllı telefon üzerinde koşan bir işletim sistemi gibi karmaşık bir sisteme gerek yoktur.
- **Gömülü Sistem Örneği:** Bir yangın algılayıcısı için; bilgisayar yerine ısı ve gaz sensörlerinin bağlanabildiği bir gömülü sistem, gerekli yazılımla beraber tüm yangın alarm işini rahatlıkla üstlenebilmektedir.
- Günümüz bilgisayarlarında kullanılan LINUX, Windows gibi işletim sistemleri **gereksinimler ölçüsünde basite indirgenerek** gömülü sistemlere entegre edilebilmektedir. Böylece gömülü sistemler üzerinde çalışan işletim sistemleri çok küçük bir alan kaplamakta, daha az güç ve daha az bellek kullanarak gerekli hız gereksinimlerini sağlanabilmektedir.
- **Gerçek zamanlı işletim sistemi:** Gerçek zamanlı işletim sistemlerindeki (Real-time Operating System–**RTOS**) temel olgu, bu sistemlerin **çok hassas bir zamanlama yeteneğine sahip olması** ve **yüksek düzeyde güvenilirlik** ile uygulamaları çalıştırmak için özel olarak tasarlanmış olmalarıdır.

- **Arabalardaki hava yastığı sistemi**, zamanlamanın önemi açısından örnek verilebilecek bir sistemdir. Zamanlamada oluşabilecek en küçük bir hata, hava yastığının erken veya geç açılmasına neden olacak ve bunun sonucu oluşabilecek zararın telafisi mümkün olmayabilecektir. Bu açıdan gerçek zamanlı bir işletim sistemine ihtiyaç vardır.
- **IAB (Internet Architectural Board):** Küresel iletişimin ve İnternetin gelişimindeki teknik yönlendirmeyi sağlamak için oluşturulan organizasyondur. İnternet adresi: https://www.iab.org/
- **RFC (Request For Comment):** İnternet üzerindeki **tüm standartların ve protokollerin tanımlandığı** dokümantasyon arşividir. Her protokolün bir RFC geçmişi bulunmaktadır.
- **Temel İletişim Altyapıları**
  - **Nesneden Nesneye İletişim:** Bu modelde **iki veya ikiden fazla nesne, üçüncü bir uygulama servisine ihtiyaç duymadan** birbirleriyle doğrudan iletişim içindedirler. IP veya Internet gibi genel ağ protokolleri kullanılabildiği gibi, Bluetooth, Z-Wave ve ZigBee protokolleri de kullanılabilir. Düşük veri hızıyla çalışabilecek ve küçük veri paketlerinin yeterli olabileceği cihazlar arasında iletişim kurmak için geliştirilmiştir.
  - **Nesne Bulut İletişimi:** Bu modelde her bir nesne, veri alışverişinde bulunmak ve ileti trafiğini kontrol etmek için uygulama hizmeti sağlayıcısı gibi doğrudan bir İnternet bulut hizmetine bağlanır. İletişimde **kablolu Ethernet ağı veya Wi-Fi** teknolojisi kullanılır.
  - **Nesne-Ağ Geçidi İletişim Modeli:** Bu modelde nesne ile bulutta sağlanacak hizmet arasında konuşlanmış bir **Uygulama Katmanı Ağ geçidi (UKA) aracılık hizmeti** sağlanmaktadır. Nesne, bir bulut hizmetine erişmek için bir UKA servisi aracılığıyla kanal olarak bağlanır. Yani cihaz ve bulut hizmeti arasında aracı olarak görev yapan bir uygulama yazılımı vardır. Bu yazılım da yerel ağdaki bir ağ geçidinde konuşlanmıştır. Günümüzde en geniş kapsamlı ve esnek iletişim modeli olarak bilinir ve **ağ geçidi yükümlülüğünü çoğunlukla akıllı telefonların aldığı** gözlenmektedir.
  - **Arka Uç Veri Paylaşım Modeli:** Kullanıcıların akıllı nesne verilerini bir bulut hizmetinden ve farklı kaynaklardan gelen verilerle birlikte işlemesine ve analiz etmesine olanak tanıyan bir iletişim mimarisidir. Bu mimari, "yüklenen sensör verisine üçüncü taraflara erişim izni verme isteğini" destekler. Bir anlamda bulut servisleri arasında da verinin dağıtımını destekler.
- **Nesnelerin İnternetinde Temel Protokoller**

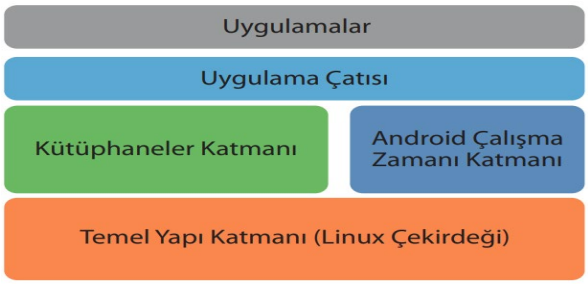


  - **IPv6**: IPv4 protokolünde desteklenen 32 bitlik adres alanı, **128 bit** alan ile yer değiştirmiştir. Böylece 2^{128} = 3.4E38 değişik cihaz tanımlanabilecektir. **IPSec** desteği ile güvenlik sağlanabilmektedir. **Akış kontrolü** ile Servis Kalitesi temelli yönlendirmeye olanak sağlamaktadır.
- **Uygulama Katmanı Protokolleri**
  - **MQTT (Message Queing Telemetri Transport):** Veri iletimi için yayınlama/abone mantığını kullanmaktadır. TCP/IP yığıtının üzerinde çalışır ve asenkron haberleşme protokolüdür. Duyurmak istediğiniz ilginç bir veriniz olduğunda nasıl bu veriyi ilan tahtasına asarsanız, bu protokol de aynı şekilde işlemektedir. Böylece, konunuzla ilgilenen herkes mesajınızı okuyabilir. Aracılar (broker) ve müşteriler (client) şeklinde iki katılımcı cihaz bulunur. Müşteriler, verilere erişebilecek veya değiştirebilecek cihazlardır. Aracılar ise verileri barındıran ve aktaran cihazlardır. En önemli özelliği basit olması ve kaynak tüketiminde ekonomik olmasıdır.
  - **CoAP (Constrained Application Protocol):** REST (Representational State Transfer) modelini temel alan bir web transfer protokolüdür. Küçük başlık boyutu nedeniyle ağırlıklı olarak M2M (cihazdan cihaza) hafif iletişim için kullanılır. CoAP, HTTP'yi taklit eder. CoAP, UDP protokolünün hızından ve basitliğinden yararlanmaktadır.
  - **XMPP (Extensible Messaging and Presence Protokol):** Jaber tabanlı açık kaynak kodlu bir protokol olup anlık mesajlaşma, çoklu sohbet ve görüntülü görüşme için geliştirilmiştir. **Uçtan uca şifreleme tekniğ**ine olanak sunarak üst düzey güvenliği sağlayabilmektedir.
- **Veri – Bağlantı Katmanı Protokolleri**
  - **ZigBee**: Düşük maliyetli, düşük güçlü, kablosuz, cihazdan cihaza ağların ihtiyaçlarını karşılamak için açık bir küresel standart olarak geliştirilen bir iletişim teknolojisidir. Sistem yapısı ZigBee koordinatörü, yönlendirici ve uç cihazlar olmak üzere üç temel cihaz türünün bir araya gelmesiyle oluşur. ZigBee standardı IEEE 802.15.4 fiziksel radyo spesifikasyonunda ve 2.4 GHz, 900 MHz ve 868 MHz dâhili lisanssız bantlarda çalışır. Özellikle pille çalışan düşük maliyetli cihazlar için tasarlanmış, paket tabanlı bir radyo protokolüdür. Birbirini 70-100 metre mesafe içerisinde doğrudan gören cihazların iletişiminde problemsiz çalışır. ZigBee hemen her türlü topoloji için rahatlıkla kullanılır.
  - **Z-Wave:** 2004 yılında ortaya çıkan ve günümüz ev otomasyon sistemlerinde yaygın bir kullanıma ulaşmış, radyo sinyalleriyle haberleşme esasına dayalı bir protokol ve bu protokolü destekleyen cihaz teknolojisini tanımlamaktadır. Z-Wave cihazlar Amerika'da 908.42 MHz, Avrupa'da ise 868.42 MHz iletişim bandında çalışmaktadırlar. Bir Z-Wave ağı teoride 232 düğümü desteklemekte, ancak pratik uygulamalarda düğüm sayısının 30-40 düğümü aşması durumunda iletişim problemleriyle karşılaşıldığı not edilmektedir. Bu sistemi destekleyen cihazlar bir örgü (mesh) yapısı içerisinde birbirleriyle ortak ağ oluşturmaktadırlar.
  - **Bluetooth:** Kısa mesafeli kablosuz cihazdan cihaza iletişim temellenmesinde en yaygın kullanılan protokoldür. Farklı elektronik cihazlar arasında veri aktarımı yapmak için kullanılan kablosuz bir teknolojidir.
- **W3C (**World Wide Web Consortium, WWW**)**, internet sitelerinde standartları belirlemek için kurulmuş organizasyondur.

- **HTML dili** kullanılarak kendi kendine çalışabilen programlar üretilemeyeceği için HTML tam olarak bir programlama dili sayılmaz.
- Notepad++ programında; Internet Explorer ön izlemesi için (Ctrl+Alt+Shift+I), Firefox ön izlemesi için ise (Ctrl+Alt+Shift+X) tuş kombinasyonu kullanılır.
- **<!doctype html> bildirimi**, bu sayfanın bir HTML5 sayfası olarak işlenmesi gerektiğini web tarayıcıya bildirir.
- **<h1>** en büyük başlığı, **<h6>** ise en küçük başlığı gösterir.
- **<p> etiketi**, web sayfasına paragraf içeriği eklemek için kullanılan HTML etiketidir.
- **<a> etiketi**, web sayfalarına bağlantı (link) vermek için kullanabileceğimiz HTML etiketidir.
- **Metin biçimlendirme etiketleri**
  - **:** Metni kalınlaştırmak için kullanılır, aynı şekilde **<b>** etiketi de aynı etkiyi verir.
  - **:** Metni eğik hale getirmek için kullanılır, aynı şekilde **<i>** etiketi de aynı etkiyi verir.
  - **<sub>:** Metni alt yazı hâline getirir.
  - **<sup>:** Metni üst yazı hâline getirir.
  - **<del>:** Metnin üzerine çizgi çizer.
  - **<u>:** Metnin altına çizgi çizer.
  - **<small>:** Metnin fontunu küçültür.
  - **<big>:** Metnin fontunu büyütür.
  - **<tt>:** Metnin daktilo fontunda yazılmasını sağlar.
  - **<cite>:** Metni alıntı olarak yazdığımızda bu etiketi kullanırız.
  - **<abbr>:** Kısaltma şeklinde yazılan bir kelimenin tanımını verir, "title" özelliği ile yazılan bilgi fare imleci ile kısaltılmış kelimenin üzerine gelince tanım olarak ortaya çıkar.
  - **
:** Bu etiket boş etikettir. Bir satırlık boşluk oluşturmak için kullanılır.
  - **<hr />:** Bu etiket boş etikettir. Bir satırlık yatay bir çizgi oluşturmaya yarar.
- **Sıralı liste** oluşturmak için **<ol>, sırasız liste** oluşturmak için **<ul>** etiketlerini kullanmalıyız. Her iki listenin içerisine <li> etiketini kullanarak eleman ekleyebiliriz.
- **Sıralı liste için liste türleri**
  - **1:** Onluk tabanda numaralama (1,2,3,4,...)
  - **i:** Küçük Romen rakamları (i,ii,iii,iv,...)
  - **I:** Büyük Romen rakamları (I,II,III,IV,...)
  - **a:** Küçük harflerle alfabetik (a,b,c,...)
  - **A:** Büyük harflerle alfabetik (A,B,C,...)
- **Sırasız liste için liste türleri**
  - **disc:** İçi **dolu bir daire** görüntüler.
  - **circle:** İçi **boş bir daire** görüntüler.
  - **square:** İçi **dolu ya da boş bir kare** görüntüler.
- Sıralı listelerde "start" özelliği kullanılarak, listenin başlangıç değerini de belirtme imkanı vardır.
- **Tanım listesi <dl>** etiketlerinin arasında tanımlanır.
  Liste içerisinde alt başlık oluşturmak için **<dt>**, bu başlığa ait bilgi vermek için ise **<dd>** etiketi kullanılır.
- **Tablo oluşturmak için <table>** etiketini kullanmak gerekir. Satır ve sütunlar ise **<tr>** ve **<td>** etiketleri ile oluşturulur. **Sütunları birleştirmek** için "**colspan**", **satırları birleştirmek için** ise "**rowspan**" özelliği, birleştirilecek sütun ve/veya satır sayısı belirtilerek ayarlanır.
- **HTML5 Semantik Etiketleri**
  - **<header>:** sayfanın tanımı, sayfanın başlığı gibi bilgileri içerir.
  - **<nav>:** sayfa içerisindeki bağlantılar veya site dışı bağlantılar için kullanılır.
  - **<article>:** bir makale ya da yazı alanını temsil eder.
  - **<section>:** sayfa içinde bir kısmı belli eder, bölümleme öğesi olarak kullanılır.
  - **<aside>:** yan menüyü belirtmek amaçlı kullanılır.
  - **<footer>:** bir sayfa veya bir kısım için alt bilgi belirten bölümdür.
  - **<address>:** iletişim bilgilerinin yer aldığı bölümdür.
- **<details>** etiketi detay bilgilerini içerirken **<summary>** etiketi sayfanın ya da yazının özet bilgisini içerir.
- **HTML5 Ses, Video ve Görüntü Etiketleri**
  - **<img>:** sayfalarda resim göstermek için kullanılır. <figure> etiketi içerisinde <figcaption> etiketi ile birlikte kullanılabilir. <figcaption> etiketi, resmin içeriği hakkında bilgi sunulan etikettir. Resim hakkında açık olmayacak şekilde, sadece arama motorlarına yönelik bilgi sunmak istersek, <img> etiketinin alt="" özelliğini kullanmamız gerekir.
  - **<video>:** sayfamıza video oynatıcıyı eklemek için kullanılır. Video oynatıcımızın genişliğini "width", yüksekliğini ise "height" özellikleri ile belirleyebiliriz. HTML kodunda geçen **"controls"** kelimesi video oynatıcının kontrollerinin açık olarak kullanıcıya sunulacağını gösterir. Video etiketinin özellikleri şunlardır:
    - **Src:** videonun kaynak adresini belirtir.
    - **Autoplay:** video hazır olur olmaz oynamaya başlasın mı başlamasın mı ayarıdır.
    - **Loop:** video bittiğinde yeniden başlayıp başlamayacağına burada karar verilir.
    - **Muted:** video ilk açıldığında sesi kapalı olup olmayacağı ayarlanır.
    - **Poster URL:** video başlamada önce, video oynatılana kadar ekranda bir resim göstermemizi sağlar.

        - **Preload auto:** sayfa yüklendiğinde videonun otomatik yüklenip yüklenmeyeceğini ayarlayabiliriz.
    - **<audio>:** sayfamıza ses oynatıcıyı eklemek için kullanılır. Özellikleri video etiketi ile benzerdir.
- **HTML5 Giriş Tipleri**
    - **Tarih Giriş Tipi (<input type="date">):** Giriş alanından tarih seçmenize yardımcı olur.
    - **Tarih-Zaman Giriş Tipi (<input type="datetime-local">):** Kullanıcının tarih ve zaman seçmesini sağlayan giriş şeklidir.
    - **E-Posta Giriş Tipi (<input type="email">):** Giriş alanından bir elektronik posta alınması için kullanılır.
    - **Ay-Yıl Giriş Tipi (<input type="month">):** Giriş alanından ay ve yılın seçilmesine olanak veren giriş tipidir.
    - **Zaman Giriş Tipi (<input type="time">):** Giriş alanından bir saat seçilmesine olanak veren giriş tipidir.
    - **Hafta-Yıl Giriş Tipi (<input type="week">):** Giriş alanından herhangi bir haftanın ve yılın seçilmesine olanak veren giriş tipidir.
    - **İnternet Kaynak Belirteci (URL) Giriş Tipi (<input type="url">):** Giriş alanının bir URL bağlantısı içerdiği durumlarda kullanılır.
    - **Telefon Giriş Tipi (<input type="tel">):** Son kullanıcının telefon numarası girmesini sağlayan giriş tipidir.
    - **Arama Giriş Tipi (<input type="search">):** Kullanıcıların arama kelimeleri girmesini sağlayan etiket tipidir.
    - **Aralık Giriş Tipi (<input type="range">):** Kullanıcının belirli aralıkta olan bir sayıyı seçmesini sağlar. Ayrıca "min" ve "max" özelliğinin değerlerini ayarlayarak hangi sayıların kabul edilebileceğini, "value" özelliği ile başlangıç değerini belirleriz.
    - **Sayısal Bilgi Giriş Tipi (<input type="number">):** Giriş alanı sayısal bir değer gerektirdiği zaman kullanılır. "min" ve "max" özelliklerinin değerlerini ayarlayarak hangi sayıların kabul edilebileceğini belirtebilirsiniz. Ayrıca "step" özelliğini ayarlayarak yukarı aşağı tuşlarına basıldığında değerin kaçar kaçar artacağını veya azalacağını belirlemeniz mümkündür.
    - **Renk Bilgisi Giriş Tipi (<input type="color">):** Kullanıcının giriş alanından bir renk seçebilmesini sağlar.
- **<canvas> etiketi**, betik (script) programlama (genellikle JavaScript) yardımı ile **grafik ve çizim alanları oluşturmak** için kullanılan etikettir. **Herhangi bir özelliği yoktur**. Yalnızca üzerine çizilebilir bir alan oluşturmaktadır.
- **<mark> etiketi,** yazı içerisinde vurgulanan metni tanımlar. Varsayılan vurgulama rengi sarıdır.
- **<meter> etiketi,** bilinen bir aralık içindeki değeri görsel olarak verir. (Loading kutusu gibi.) Etiketin içerisinden ayarlayabileceğimiz "min" ve "max" özellikleri ile elemanın alabileceği en küçük ve en büyük değerleri verebiliriz. Ayrıca, "value" özelliği ile o anki değeri girebiliriz.
- **<progress> etiketi,** bir işlemin ilerleme sürecini görsel olarak verir. Etiketin değerlerini "progress value" özelliğini ayarlayarak verebiliriz. Ayrıca, "max" özelliği ile alabileceği en yüksek değeri ayarlarız.
- **<datalist> etiketi,** giriş elemanları için önceden tanımlanmış seçenekler listesini içerisinde barındırır ve bu liste içerisindeki elemanlara **otomatik tamamlama özelliği** verilmesini sağlar.
- **<output> etiketi,** hesaplama sonucunu tanımlar.
- HTML sayfalarının "**geolocation**" desteği ile sayfamızı görüntüleyen kullanıcının konumunu, javascript kodu yardımıyla alabiliriz.
- **CSS**, İngilizce "Cascading Style Sheet" (Basamaklı Stil Sayfaları) kelimelerinin baş harflerinden oluşan bir kısaltmadır.
- **CSS etiketleri,** <style type = "text/css">…</style> etiketleri arasına yazılır.
- **Yuvarlatılmış Çerçeve Görünümü:** "**border-radius:**" özelliği ile html etiketlerin köşelerine farklı yuvarlatma değerleri verilebilir.
- **Metin Gölgelendirme:** "**text-shadow:**" ile sağlanır. Örnek olarak text-shadow: 3px 3px 3px #FF0000; verilebilir.
- **Geçişler (transition):** Bu özellik ile bir betik dili (javascript) veya ek bir özellik kullanmadan, stil uyguladığımız elemanın fare ile etkileşime girmesini sağlayabiliriz.
- **Çoklu Kolonlar:** Metinlerin normal akışlarının dışında gazete veya dergilerde olduğu gibi çok kolonlu olarak gösterilmesini sağlar.
    - **column-count:** Metnin kaç kolona bölüneceğini belirler.
    - **column-gap:** Kolonlar arasında ne kadar mesafe olacağını belirler.
    - **column-rule:** Kolonlar arasında çizgi olup olmayacağını belirler.
    - **column-rule-color:** Kolonlar arasındaki çizginin rengini belirler.
    - **column-rule-style:** Kolonlar arasındaki çizginin şeklini belirler.
    - **column-rule-width:** Kolonlar arasındaki çizginin kalınlığını belirler.
    - **column-width:** Kolonların genişliğinin ne kadar olacağını belirler.
- **Renk Geçişli Görünüm:** Kullanımı linear-gradient(Renk1, Renk2); şeklindedir.
- **Dairesel renk geçişi** sağlamak için, radial-gradient(Renk_1,Renk_2 , … , Renk_n); şeklinde bir css komutu yazılabilir.
- **İşletim Sistemi**: Bilgisayar, oyun konsolu, cep telefonu, araba, beyaz eşya vb. cihazlarda çalışan, donanım kaynaklarını yöneten ve çeşitli uygulama yazılımları için yaygın servisleri sağlayan bir yazılımlar bütünüdür. Popüler olan mobil işletim sistemlerine örnek olarak Android, iOS, Windows Mobile, BlackBerry OS, Symbian vb. gösterilebilir.
- **Mobil Uygulama Nedir?** Mobil cihazlar için tasarlanmış ve kodlanmış yazılımlara mobil uygulama denir.
- **Mobil Uygulama Ne Değildir?** Mobil uygulama, ***mobil bir web sitesi değildir***.
- Aynı web sayfasının kademeli olarak ekran boyutlarına küçültülüp büyütülebildiği tasarımlara **responsive tasarım** denir.
- **Düşük Seviyeli Programlama Dili**, komut kümesinde hiç soyutlama imkânı vermeyen ya da kısıtlı miktarda soyutlama sağlayan anlamına gelir. Burada kullanılan "düşük" kelimesi, programlama diliyle makine dili arasında bulunmayan ya da az bulunan bir soyutlama imkânını temsil etmektedir. Bu anlamda düşük seviyeli programlama dilleri "donanıma yakın" olarak da bilinmektedir.
- **API (Uygulama Programlama Arayüzü)**, **bir yazılımın başka bir yazılımda tanımlanmış fonksiyonlarını kullanabilmesi** için oluşturulmuş bir tanımlar bütünüdür. Örneğin, YouTube API'si kullanarak uygulama içerisinde YouTube videoları yayımlanabilmekte, Turkcell API'si kullanarak mobil uygulama içerisinde Turkcell'in sunduğu SMS gönderme, kimlik doğrulama, konum sorgulama, mobil ödeme vb. hizmetlerini kullanabilen bir uygulama yazılabilmektedir.
- **Internet of Things – IoT:** Temel haliyle, nesnelerin internet üzerinde veri alışverişi yapabilmesi ve internete bağlı olan diğer tüm cihazlarla iletişim kurarak nesneler ekosistemi içerisinde yaşayabilmesidir.
- **Çoklu Platform (Cross Platform) yazılımları**, birden fazla işletim sisteminde dağıtımı olan yazılımlardır.

- **Yazılım Geliştirme Kiti (Software Development Kit - SDK):** Belli bir yazılım paketi, donanım platformu, bilgisayar sistemi, oyun konsolu, işletim sistemi veya benzeri bir platform için uygulama üretmeyi sağlayan yazılım geliştirme araçlarından oluşan yazılımdır.
- **Plug-in:** Kendi başına çalışabilen yapı için, çok özel bir alanda gereklilik üzerine geliştirilen, yapıya yeni özellikler ekleyen yazılımdır.
- **Mobil Uygulama Türleri**
  - **Yerel (Native) Uygulamalar:** Uygulamanın cihaza özgü yerel dille yazılarak hazırlanmasıdır. Belli bir platforma özel, genellikle platform sağlayıcının üretmiş olduğu SDK araçları ve programlama dili ile geliştirilmiş uygulamalardır.
    - **Platform Tabanlı Yerel Uygulamalar**: Platforma ait işletim sistemlerini üretenlerin sunduğu uygulama geliştirme ortamları ve dillerinin kullanılmasıyla yazılan uygulamalardır. Google Android -> Android Studio -> Java, Apple iOS -> XCode -> Swift, Microsoft Windows Phone -> Visual Studio -> C#. Platform tabanlı yerel uygulamalar, donanımın bütün yeteneklerine erişme imkânı sağlar ve hız bakımından yüksek performans gösterirler. **Dezavantaj olarak ise** işletim sistemlerinin her birinin yapısına, geliştirme ortamına ve programlama diline hâkimiyet gerektirirler.
    - **Çoklu Platform Yerel Uygulamalar:** Bu kategoride yazılan uygulamalar tek bir geliştirme ortamı ve tek bir programlama dili kullanarak çeşitli platformlara aynı anda çıktı üreten uygulamalardır. Bu alanda en bilinen örnekler, Xamarin, Titanium ve Smartface App Studio gibi araçlardır. (Flutter) Bu platformların en büyük avantajı, bilinmesi gereken tek bir çatı yapısının, geliştirme ortamının ve programlama dilinin olmasıdır.
  - **Web Tabanlı Uygulamalar:** Açılan uygulamanın içerisinde bir web sitesi çalışmaktadır. Web uygulamalarında ise HTML5, JavaScript ve CSS3 gibi farklı diller kullanılmaktadır. En büyük avantajları, bilinen İnternet teknolojileriyle sunucu üzerinden yayın yapılması sayesinde istenilen zamanda müdahâle edilebilmesi ve her cihazda çalışabildiği için ayrı ayrı programlama gerekmemesidir. **Dezavantajları olarak ise** aslında cihazda çalışan bir uygulama olmaması ve tarayıcı üzerinde çalıştığı için bazı deneyim, performans kayıplarının yaşanması, yerel uygulama gibi esnekliğin olmaması ve çevrimdışı çalışamaması olarak sıralanabilir.
  - **Melez uygulamalar:** Temel olarak tek bir kod çıktısı ile birden fazla platformda uygulama çalıştırılabilmek amacıyla geliştirilmiştir. Uygulamanın melez olması, açıklanabilir web uygulamaları ile yerel uygulamaların karışımı olarak açıklanabilir. Adobe'un açık kaynak bir aracı olan PhoneGap, sağladığı çeşitli plug-in destekleriyle bazı cihaz donanımlarına erişim sağlamaktadır.
- **Mobil Uygulama Geliştirme Platformları**
  - **Android:** Google ve Open Handset Alliance tarafından kodlanmış, Linux İşletim Sistemi tabanlı, mobil cihazlar için geliştirilmiş açık kaynak kodlu bir işletim sistemidir.



    - **Temel Yapı (Linux Çekirdeği) Katmanı:** Android sisteminin en alt katmanında "Linux Çekirdeği (Kernel)" kullanılır. Linux çekirdeğinin doğrudan kaynak sağladığı yapılar olarak güvenlik, hafıza ve süreç kontrolü, dosyalama ve bağlantı için girdi/çıktı işlemleri, cihaz sürücüleri sayılabilir.
    - **Kütüphaneler Katmanı:** Android mimarisinin en önemli katmanlarından biri olan kütüphaneler katmanında, C dili ile yazılmış yerel sistem kütüphaneleri, internet tarayıcı motorlarının çalışması için Webkit, görüntüleme kontrolünü yapan yüzey yöneticisi (Surface Manager), grafik işlemleri için OpenGL, ses-video işlemleri için Media Framework, veri yapıları kontrolü için SQLite gibi yapılar bulunmaktadır.
    - **Android Çalışma Zamanı (Android Runtime) Katmanı:** Android Çalışma Zamanı Katmanı, Linux çekirdeği kütüphaneleriyle Java kütüphanesinin birleştiği bölümdür. İki önemli bileşen vardır. Bunlar temel Java kütüphaneleri ve sanal makinedir. Uygulamalar sanal makine tarafından çalıştırılmaktadır. Android için uygulama geliştirme dili Java'dır. Java ile yazılan uygulamalar alınır, Java kodları derlenerek bayt kod (bytecode) dosyalarına çevrilir. Bu dosyalar, **dex dosyasına** çevrilerek **Dalvik Sanal Makinesi**'nin çalıştırabileceği şekildeki dosyalar haline gelir.
    - **Uygulama Çatısı (Application Framework) Katmanı:** Uygulamaların işletim sistemiyle etkileşimini sağlayacak olan sınıf ve servislerin bulunduğu katmandır.
      - Aktivite Yöneticisi (Activity Manager): Aktivitelerin çalıştırılması, durdurulması gibi durumların yönetiminden sorumludur.
      - Pencere Yöneticisi (Window Manager): Tüm uygulamaların görüntülenen ekranlarının yönetilmesinden sorumludur.
      - İçerik Sağlayıcı (Content Provider): Uygulamalar arasında veri paylaşımı ve erişimi yapmaktan sorumludur.
      - Görüş Sistemi (View System): Kullanıcı arayüzü oluşturulmasından ve yönetilmesinden sorumlu olan servistir.
      - Paket Yöneticisi (Package Manager): Cihaz içerisinde kurulu uygulama, paketlerinden (apk dosyaları) sorumludur.
      - Telefon Yöneticisi (Telephony Manager): Cihaz yönetiminden sorumlu olan servistir. Uygulama içerisinde cihazla alakalı bir işleve aracılık etmektedir. Android telefona gelen bir aramanın tespit edilmesi buna örnek verilebilir.
      - Lokasyon Yöneticisi (Location Manager): Cihaz içerisinde bulunan Lokasyon (Global Pozisyonlama) erişimini sağlar, yönetir.
      - Kaynak Yöneticisi (Resource Manager): Uygulamalar içerisindeki resim, dosya ve kaynakların yönetiminden sorumludur.
      - Bildirim Yöneticisi (Notification Manager): Cihaz üzerindeki bildirimlerden (bilgi mesajı) sorumlu olan servistir.
    - **Uygulamalar (Applications) Katmanı:** Android işletim sistemi mimarisinde en üst katmandır.

- **iOS:** Apple firmasının mobil platformları için geliştirdiği işletim sistemidir. Temel programlama dili Objective-C'dir. Haziran 2014'ten itibaren yerini Swift'e bırakacak duruma gelmiştir. Beş katmandan oluşan iOS sistem mimarisi katmanlarını, Çekirdek İşletim Sistemi (Core OS), Çekirdek Servisler (Core Services), Medya, Cocoa Touch ve Uygulamalar katmanları olarak sıralayabiliriz.
  - **Çekirdek İşletim Sistemi (Core OS) Katmanı:** İşletim sisteminin temelini oluşturan katmandır. **Donanıma en yakın** katman olup alt seviye arayüzler, güvenlik çatıları, güç yönetimi, dosya sistemi ve diğer donanıma yakın birimlerin olduğu katmandır.
  - **Çekirdek Servisler (Core Services) Katmanı:** Uygulamalar için gerekli olan temel sistem servislerinin sunulduğu bölümdür. Kullanıcı giriş işlemleri, depolama, lokasyon işlemleri, kullanıcı adres defteri, temel telefon işlemleri, sistem ayarları gibi temel işlemlerle iletişime geçmemize olanak tanır.
  - **Medya Katmanı:** Cihazda ses, görüntü ve video işlemleri ile ilgili kütüphanelerin bulunduğu önemli bir katmandır. Bu katmanda yer alan yapılar; Çekirdek Grafikler, Medya Yöneticisi, Çekirdek Medya – Ses – Video, OpenGL olarak sıralanabilir.
  - **Cocoa Touch Katmanı:** Kullanıcıya yakın olan, kullanıcıyla iletişime olanak veren görsel arabirimleri sağlayan sınıfların yer aldığı kısımdır. Dokunmatik ekran, bildirimler, çoklu-işlevsellik gibi önemli servisler burada yer alır.
  - **Uygulamalar (Applications) Katmanı:** Android işletim sistemine benzer şekilde en üst katmandır.
- **Windows:** Mobil cihazlarla ilgili olarak ilk ürününü 1992 yılında geliştirmeye başlayan ve 1996 yılında piyasaya süren Microsoft, o dönemlerin ilk mobil işletim sistemi olan **Windows CE**'yi piyasaya sürmüştür. Bu işletim sistemi, mimari olarak Windows 95, kullanıcı arabirimi olarak da **WinPad** kullanmıştır. Windows Phone 8'de **C++** kullanarak uygulama yazılabilmektedir.

- **Open Handset Alliance:** Mobil cihazlar için açık standartlara dayanarak bir araya gelen **seksen dört firmanın** oluşturduğu bir uluslararası birliktir. Üye firmalar arasında Google'ın dışında HTC, Sony, Dell, Intel, Motorola, Qualcomm, Texas Instruments, Samsung Electronics, LG Electronics, T-Mobile, Sprint, Nvidia ve Wind River Systems gibi firmalar bulunmaktadır.
- **ROM:** Android cihazlarında işletim sistemi yüklenmesi için ayrılmış alandır. Bu kısma gerek cihaz üreticilerinin hazırladığı işletim sistemi gerekse üçüncü kişilerin hazırladığı özel ROM'ları ekleyerek cihazdaki işletim sistemi yükseltilebilir veya alçaltılabilmektedir.
- **Genel Kamu Lisansı Çekirdeği:** Sistem araçlarını, kütüphanelerini ve son kullanıcı yazılımını içeren özgür belgeleme lisansı tasarısı kapsamında birçok yerde kullanılan, özgür bir yazılım lisansıdır.
- **Apache Lisansı:** Apache Yazılım Derneği tarafından yayımlanan bir özgür yazılım lisansıdır. Özgür ve açık kaynak kodlu yazılımın geliştirilmesi için kaynak kodlarının kullanımına izin vermektedir.
- **Bayt kod (Bytecode):** Taşınabilir kod olarak bilinen bayt kod, bir yorumlayıcı tarafından çalıştırılabilir ve aynı zamanda makine diline derlenebilir durumdaki komut setini ifade etmektedir.
- **Dalvik Sanal Makinesi:** Android işletim sisteminde uygulamaların yüklenmesi esnasında **önbellekleme yapılmasına olanak sağlayan** bir sanal makine kurgusudur. Bu sayede uygulamalar farklı işlemci ve bellek kullanan cihazlarda sorunsuz çalışabilmektedir.
- **.apk Dosyası:** İngilizce açılımıyla Android Application Package olan ve Android platformunda kullanılan ön tanımlı paket dosya formatıdır. Android uygulamasının çalıştırılabilir dosya formatı olarak kullanılmaktadır.
- **Android Sanal Cihaz Yöneticisi (Android Virtual Device - AVD Manager):** Uygulamanın geliştirici için bilgisayarında Android işletim sistemi yüklü sanal cihazlar oluşturabilmesini ve bu cihazların yönetiminin yapılabilmesini sağlamaktadır.
- **Windows CE:** İngilizce Compact Editional olarak bilinen, Microsoft tarafından taşınabilir cihazlar için üretilmiş bir işletim sistemidir.
- **Silverlight ve XNA:** Microsoft tarafından geliştirilen **animasyon, vektör, 3D, grafik ve görüntü oynatma** imkânı sağlayan geliştirme ortamıdır. XNA ise Xbox uygulamalarını Windows platformlarında çalıştırmak için Microsoft'un geliştirdiği bir çatıdır.
- **.NET:** Microsoft'un, programlama dilinden ve çalıştırılacak sistemden bağımsız olarak uygulama geliştirmeyi amaçlayan platformudur.
- **Oyun**, bir faaliyete ilişkin düşünce ve duyguları ortaya çıkaran, eğlence odaklı bir aktivitedir.
- **Dijital Oyun Türleri**
  - **Aksiyon Oyunları:** Aksiyon filmlerinin benzeri bir oyun türüdür. Hareket, hız, savaş ve şiddet unsurlarını da içerisinde barındırır.
  - **Birinci Şahıs Nişancı Oyunları:** Oyuncu karakterinin **kendi gözünden oynandığı** dijital oyun türüdür.
  - **Macera Oyunları:** **Araştırma, keşif, bulmaca** çözme gibi unsurları içeren, hikâye odaklı bir dijital oyun türüdür.
  - **Dövüş Oyunları:** Oyuncuların ilgili senaryoyu belli bir karakterle veya oyunda sunulan **karakter seçeneklerinin birine bürünerek** oynadıkları dövüş odaklı oyunlardır.
  - **Platform Oyunları:** Genelde oyuna konu olan kahramanın **çeşitli engeller üzerinden** geçirilerek bölümün tamamlanması ve bir sonraki bölüme geçilmesi üzerine kurulmuş bir **algoritmik yapıya dayanan** oyun türüdür.
  - **Bilmece Oyunları:** Yanıtlanması için sorulan şaşırtıcı ve eğlendirici soruların kullanıcıya yöneltilerek oynandığı oyun türüdür.
  - **Bulmaca / Zekâ Oyunları:** Kişinin görsel okuryazarlığını geliştiren bir platform üzerinde oynanan dijital oyun türüdür.
  - **Rol Yapma Oyunları:** Oyuncunun belli bir durumda, **bir karakterin rolünü üstlendiği**, var olan bilgi ve kaynakları kullanarak bu karakterin başına gelen sorunları çözdüğü oyunlardır.
  - **Simülasyon Oyunları:** **Gerçek hayatta** yapılan bazı işlerin oyun hâline getirilmesi şeklinde tasarlanan oyunlardır.
  - **Strateji Oyunları:** Stratejik kararlarla oynanan dijital oyunlardır. En bilinen strateji oyunu türü **satranç** oyunudur.
  - **Spor Oyunları:** Genellikle **bir spor alanından taklit edilerek** hazırlanan oyunlardır.
  - **Mantıksal Oyunlar:** Oyuncuların belli **mantık hesaplamaları** yaparak ilerleme sağlayabildiği dijital oyun türüdür.
  - **Matematik Oyunları:** Oyuncuların matematiksel düşünme ve pratik hesap yapabilme yeteneklerini arttıran dijital oyun türüdür.
  - **Eğitsel Oyunlar:** Eğitim amaçlı tasarlanmış dijital oyun türüdür.
  - **Çevrimiçi Oyunlar:** İnternet üzerinden bağlanılarak oynanan dijital oyunlardır.

- **Dijital oyun geliştirme aşamaları;**
  1) fikir ve senaryo belirleme, 2) hedef kitle, 3) hikâye, 4) oynanabilirlik, 5) rekabet ortamı, 6) donanım kaynakları tüketimi.

- **Kullanıcı Arayüzü (User Interface – UI) & Kullanıcı Deneyimi (UI & UX):** Kullanıcı Arayüzü, aslında arayüz tasarımı olarak bilinir. Örneğin, bir televizyon kumandasının üzerindeki tuşlar bir arayüz tasarımı örneğidir. Tuşların yerleri, büyüklükleri, renkleri tasarımcıların verdiği kararlar sonucu uygulanır. Kullanıcı Deneyimi (User Experience – UX) ise nasıl göründüğü ile ilgili değil, kullanıcının nasıl hissettiğiyle ilgilenmektedir. Kullanıcı deneyiminin amacı yapılmak istenilen işlemin basit, sorunsuz ve kolay bir şekilde yapılmasını sağlamaktır.
- **Oyun motoru**, kişilerin veya şirketlerin **oyun yapımında kullandıkları ücretli veya ücretsiz yazılımlar**a verilen genel bir isimdir.
- Modellemenin üç boyutlu ortamlarının gerçeğe yakın ve oyun türüne uygun olacak şekilde hazırlanması için bünyesinde barındırdığı "**model editör**" kullanılır.
- **Detay seviyesi**, kameraya yakın modellerin daha detaylı görünmesi, uzakta olan modellerin detay seviyesinin daha düşük olması mantığına dayanmaktadır.
- **Oyun tasarımı ve programlama aşamaları:** 1) **Modelleme**, 2) **Kaplama**, 3) **Animasyon**, 4) **Programlama**.
- **Modelleme aşamasında kullanılan başlıca terimler**
  - **Varlık (Asset):** 3ds Max, Photoshop vb. harici programlarda üretilmiş modeller, dokular, ses efektleri ve animasyonlar gibi içeriklerin tamamına verilen genel bir isimdir.
  - **Poligon Sayısı (Polygon Count):** Üç boyutlu ortamda hesaplanan modelde kullanılan toplam üç kenarlı poligon sayısıdır.
  - **Üçgen (Triangle):** Bir üç boyutlu modelin, üç kenarından oluşan poligon yüzeyini ifade eder.
  - **Dörtgen (Quad):** Bir üç boyutlu modelin, dört kenarından oluşan poligon yüzeyidir.
  - **Gerçek Zamanlı Sonuçlandırma (Real Time Rendering):** Oyunların, içeriklerini belli bir hızda görselleştirerek sonuçlandırması gerekmektedir.
  - **İyileştirme (Optimization):** Bu terim en uygun hale getirme olarak tanımlanabilir.
  - **Detay Modelleri (LOD Models):** Oyunun farklı çözünürlüklerinde kullanılan sürümleridir. Bu modeller, nesnenin kameraya olan uzaklığının temel alınmasıyla poligon sayısı açısından farklı şekilde ifade edilmesinden oluşmaktadır.
  - **Siluet (Silhoulette):** Bir nesnenin silueti, onun ana formunu oluşturmaktadır. Bu sayede duvara vuran gölgeden nesnenin ne olduğu anlaşılabilmektedir.
- **Kaplama aşamasında kullanılan başlıca terimler**
  - **UV Haritası (UV Map):** Üç boyutlu nesnelerle iki boyutlu **dokular** arasında köprü görevi gören, çizilen detayların objenin neresinde ve nasıl görüneceğinin koordinatları ile belirlendiği kısımdır.
  - **Dağılım Haritası (Diffuse Map):** Nesnenin ana rengini oluşturan ve **tek renk bilgisi** barındıran haritaya verilen addır.
  - **Yükselti Haritası (Bump Map):** Sonlandırma (render) süresi oluşturmadan nesneye yükseklik ve derinlik katma hilesi sağlar.
  - **Normal Harita (Normal Map):** Yükselti haritası gibi, geometrik olmayan detayların oluşturulmasını sağlamaktadır.
  - **Yansıma Haritası (Specular Map):** Nesnelere parlaklık katmada kullanılmaktadır. Haritadaki siyah bölgeler parlamazken beyaz bölgeler parlama özelliği göstermektedir.
  - **Saydamlık Haritası (Alpha Map):** Oluşturulan grafiklerde, saydamlık bilgisi içeren kanala "**Alfa**" ismi verilmektedir. Bu, modelin hangi bölgesinin saydam olacağını belirleyen kanaldır. Siyah bölgeler opakken, beyaz alanlar şeffaf olmaktadır.
  - **Yerdeğişim Haritası (Displacement Map):** Normal haritaya benzer bir şekilde modele detay eklemekte kullanılmaktadır. Normal haritadan farklı olarak modeli fiziksel olarak değiştirir.
  - **Yayılma Haritası (Emissive Map):** Yansıyan ışıktan oluşan parlama yerine kendisi ışık kaynağı olan nesnelerin, ışık yayma şeklini tayin eden haritadır.
  - **Işık Haritası (Light Map):** Olabildiğince fazla işlem ışık haritasına taşınarak simule edilmektedir. Bir kısmı da dokular üzerine boyanarak istenilen etki sağlanmaktadır.
  - **Doku Döşeme (Tiling Textures):** Dokular belli yönlerde sonsuza kadar tekrar edilerek döşenebilir. Duvarlar, su, yer yüzeyleri gibi alanlarda çok kullanılmaktadır.
  - **Çıkartmalar (Decals):** Saydamlık barındıran dokulardır. Mermi izleri, sıçrayan kanlar, posterler, çöpler ve buhar… gibi.
  - **Kaplamalar (Shaders):** Bütün doku çalışması bittiği zaman tüm unsurlar kaplama adı verilen bir nesne haline getirilmektedir.
  - **Doku Çözünürlüğü (Texture resolution):** Bir dokunun piksel cinsinden boyutlarını ifade etmektedir.
- **Animasyon aşamasında kullanılan başlıca terimler**
  - **Donatma (Ringing):** Animasyona kemik sistemi oluşturmak için kullanılmaktadır.
  - **Giydirme (Skinning):** Oluşturulan kemik sistemiyle modelin gövdesinin tüm etkileşimini barındıran süreçtir.
  - **Tek Animasyon (one-off Animation):** Bir nesnenin veya karakterin **özel animasyonlarına** verilen isimdir. Silah çekme, top fırlatma gibi örnekler verilebilir.
  - **Döngüsel Animasyonlar (Looping Animations):** Oyuncunun fark etmediği, **devamlılık içeren** animasyonlardır. (Koşma, Yürüme)
  - **Soket (Socket):** İkincil animasyonlarda kullanılabilmesi için, kemik sistemine ve kontrol teçhizatlarına yerleştirilen geçici nesnedir.
  - **Eklenebilen Animasyonlar (Additive Animations):** Temel hareketlerin ortasına yerleşip, onlarla gizlice karışabilen animasyonlara verilen isimdir.
- **Programlama aşamasında kullanılan başlıca terimler**
  - **Oyun Motoru (Game Engine):** Oyun geliştirmek için kullanılan temel sistemlerdir.
  - **Fizik Motoru (Physics Engine):** Oyundaki fizik kurallarının etkisinin görülebildiği yapıları fizik motoru belirlemektedir. Bir oyundaki fizik kurallarının etkisini görebilmek için binlerce satır kod yazmak gerekmektedir. Ancak oyun motorlarında bu optimizasyon otomatik ve değiştirilebilir bir şekilde gelmektedir.
  - **Oyun Döngüsü (Game Loop):** Herhangi bir etkileşimin olmadığı durumlarda bile, oyunun akıcı bir şekilde çalışmasını sağlar.
  - **Oyun Durumu (Game State):** Dijital oyunun içsel mantığını oluşturan ve zamanlayıcılar, skorlar, seviyeler, bölümler gibi önemli aşamaların gidişatını izleyen sistemdir.

  - **Kullanıcı Arayüzü (GUI):** Oyuncuların ekranda gördüğü arayüze verilen genel addır.
  - **Yapay Zekâ (AI):** Cihazın oyundaki karakter ve sistemleri yönettiği karar alma mekanizmasıdır.
  - **Yol İşaretleri (Waypoints):** Yapay zekânın anlayabileceği en basit hâliyle oluşturulan seviyelere verilen isimdir.
  - **Kod Yazma (Scripting):** Yüksek seviyeli dillerle kod yazma ve oyundaki tüm fonksiyonel yapıyı ortaya çıkarmaktır.
- **Oyun motorları** dikkate alındığında en büyük dezavantaj olan maliyet, açık kaynak kod ve ücretsiz birtakım yazılımlarla azaltılmaktadır.
- **Popüler Oyun Motorları**
  - Unity: **Üç boyutlu oyun motoru** olan Unity, gelişmiş özelliklere sahip üç boyutlu dijital oyunların cihaza kurulmadan oynanmasını sağlayan bir yapıya sahiptir. Unity 3D motorunu kullanan oyunlar, Unity Web Eklentisi sayesinde hiçbir kurulum işlemi olmadan bir tarayıcı üzerinden çalışabilmektedir. Bir başka kolaylık ise, Unity kullanılarak geliştirilen dijital oyunun herhangi bir altyapı değişikliğine gerek olmadan farklı platformlara (PC, Mac, Web, PS4, iOS, Android, Windows Phone) uygun olarak derlenebilmesidir. Diğer oyun motorlarında grafik ile kod kısımları ayrılmışken, Unity'de grafik ile kod **birlikte** çalışmaktadır. **C#** ve **JavaScript** gibi yüksek seviyeli programlama dillerini destekler.
  - Unreal Engine: **Epic Games** firması tarafından geliştirilen Unreal Engine, piyasadaki çoğu **birinci şahıs nişancı (FPS) türü oyunların** yapımında kullanılmıştır. Programlama dili olarak **C++** kullanılsa da birçok kütüphane ve script dillerini desteklemektedir. Çoklu platform desteği de vardır.
  - CryEngine: Üç Türk kardeşin kurduğu **Crytek** adlı firma tarafından geliştirilmiş oyun motorudur. Yüksek grafik kalitesini sağlarken **dışarıdan fizik motoru veya kütüphane ihtiyacının duyulmaması** CryEngine motorunu popüler yapan özelliklerindendir. Dezavantaj olarak, geliştirilen dijital oyunların yüksek kapasiteli donanım gereksinimi olması gösterilebilir. CryEngine, grafik olarak daha gelişmiş yapısıyla **C++** dil yapısını iyi bilmeyi gerektirmektedir.
  - CopperCube: **Ambiera** firması tarafından geliştirilen oldukça yeni grafik editörü ve oyun motorudur. Piyasadakilerden farklı olarak hiçbir teknik kodlama ve profesyonel üç boyut bilgisi gerektirmeden üç boyutlu dijital oyunlar oluşturulabilmesini sağlamaktadır.
  - Microsoft XNA: 2006 yılında Microsoft tarafından oyun programlama üzerine **.NET çatısını** kullanarak iki boyutlu ve üç boyutlu oyunlar geliştirmek için piyasaya sürdüğü bir başka çatıdır. .NET dillerinden herhangi birinin (C#, VB.NET, C++.NET) bilinmesiyle XNA ile oyun programlamaya başlanabilecek bir ortam sunulmuştur. OpenGL, DirectX gibi aşağı seviyede fonksiyonlarla uğraşmaya gerek yoktur. Uygulamalar Windows, XBOX, Windows Phone platformlarında çalıştırılabilir.
- **Grafik işlemciler** ilk olarak sadece grafik tabanlı işlemler için kullanılıyordu. Sonraları grafik işlemcilerin çeşitli nümerik işlemleri de yapabileceği fark edilmiştir. Günümüzde **biyoinformatik**, **hesaplamaya dayalı finans**, **sayısal analitik**, **moleküler dinamik**, **medikal görüntüleme**, **hava iklim tahmini** gibi birçok uygulama alanında kullanılmaktadır.
- **Paralel programlamada**, hesaplama ortamında eşzamanlı olarak çalışan birden fazla komut işleyici vardır. Bir başka deyişle problem, birden fazla hesaplama birimi tarafından eşzamanlı olarak çözülür.
- **Merkezi işlemci birimi (CPU):** Bilgisayarın en önemli parçasıdır ve aritmetik ve mantıksal operasyonları gerçekleştirir.
- **İşlemcilerin içindeki elektronik devreler**, bir tür saat vuruşu mekanizması ile hareket ederler. Saat vuruşu, dijital bir elektrik sinyalinin 1'den 0'a inişi ya da 0'dan 1'e çıkışı ile tanımlanır. Bir saniye içerisinde bu saat vuruşlarından kaç tane olduğu, o işlemcinin frekansını belirler. Her saat vuruşu yeni bir komut çalıştırılması anlamına geldiğinden, bir işlemcinin frekansı ne kadar yüksekse, birim zamanda çalıştırabildiği komut sayısı da o kadar yüksektir.
- **Grafik işlemci birimi (GPU):** **Görüntü işleme komutlarını çalıştırmak için** özelleşmiş bir elektronik devredir. GeForce 3 serisi DirectX 8.0 standardını uygulayan ilk yonga olmuştur. OpenGL API ve DirectX ile birlikte GPU'lar kabiliyetlerini arttırmaya başlamışlardır. Bilgisayar programcıları bilgi üzerinde istedikleri hesaplamaları grafik işlemci birimine de yaptırabilirler. Böylece, GPU ile sadece piksellerin renk hesaplaması değil, farklı nümerik hesaplamaların yapılması da mümkün hâle gelmiştir. GPU ile iletişimi sağlayan **OpenGL ve DirectX** kütüphanelerinin yapısının öğrenilmesi, GPU programlama yapabilmek için oldukça önemlidir. GPU ile veri okuma işlemleri çok hızlı bir şekilde yapılabilmektedir.
- **GPU hesaplamanın erken dönemlerindeki kısıtlar**: 1) işlem sonuçlarının hafızaya nasıl kaydedileceği, 2) ondalıklı sayılar ile hesaplamanın nasıl yapılabileceği, 3) derleme hatalarının nasıl ayıklanabileceğidir.
- **CPU-GPU Karşılaştırılması:** CPU programlama genellikle sınırlı sayıda kuyrukların çalışacağı uygulamalar için optimize edilmiştir. GPU'lar uzun hesaplama komutlarının hâkim olduğu çoklu kuyruklu uygulamalarda kullanılmaktadır. GPU'lar kuyruk işleme, veri önbelleği, sanal bellek yönetimi gibi konularda gelişmeye devam etmektedir. GPU'larda aritmetik işlem yapan birimler, CPU'lara göre daha fazla sayıda bulunmaktadır. Bu nedenle çok fazla aritmetik işlem gerektiren operasyonları GPU'lar daha hızlı yapabilmektedir. Ayrıca, GPU'lar ile ondalıklı sayı işlemleri günümüzde CPU'lardan 20 kat kadar daha hızlı yapılabilmektedir.
- **CUDA (Compute Unified Device Architecture),** GPU programlama için NVIDIA firmasının sunduğu C, C++, Fortran gibi programlama dilleriyle birlikte kullanılabilen mimari ve teknolojidir. CUDA'nın genel amaçlı GPU programlamada sunduğu çeşitli avantajları vardır. Bunlardan ilki, hafızadan rastgele seçilen bir adresin CUDA kodu ile okunabilmesidir. Bir diğeri ise bit seviyesinde işlemler ile tam sayı ve ondalıklı sayı işlemler için tam destek sağlamasıdır.
- **CUDA Kısıtları:** Standart C kütüphanelerini tamamen desteklemez!!! Standart C'de derlenebilen bazı kodlar, CUDA C'de derlenememektedir. Ayrıca gerçekleme (rendering) dillerinden OpenGL ile çalışabilirliği tek yönlüdür. Yani, OpenGL kayıtlı CUDA hafızasına ulaşabilirken, **CUDA OpenGL hafızasına erişememektedir.**
- **Heterojen Hesaplama Ortamı:** CPU ve GPU'nun birlikte kullanıldığı hesaplama ortamlarına denir.
- C dilinde yazılmış bir kod, NVIDIA derleyicisinde "nvcc" komutu kullanılarak derlenebilir.
- **cudaDeviceSynchronize():** CUDA'da kuyruklar ile paralel işlem yaparken, ana (main) fonksiyonda bütün kuyrukların **sonucunu bekleyebilmek** için kullanılan komuttur.

- **Büyük Veri:** Sadece hacimsel olarak verinin çok fazla olduğu anlamına gelmez. Elimizdeki verinin büyük veri olarak tanımlanabilmesi için verinin karmaşık yapıda ve hacimsel olarak büyük olması gerekir.
- **Büyük Veri Bileşenleri:** Çeşitlilik (variety), Değer (value), Doğrulama (verification), Veri büyüklüğü (volume) ve Hızdır (velocity). (5V)
  - **Çeşitlilik (Variety):** Veri, birçok farklı kaynaklardan derlendiği için kendi içinde çeşitlilik ve farklılık göstermektedir.
  - **Hız (Velocity):** Hızla gelen veriyi işleyebilmek için, aynı zamanda uygulamaların işlem yapabilme hızının da artması gerekmektedir. Bazı uygulamalar için verinin ömrü kısadır. Ömrü dâhilinde işlenemeyen veri geçerliğini yitirecektir.
  - **Veri büyüklüğü (Volume):** Organizasyonların her geçen yıl artan büyük hacimdeki verileri nasıl depolayacağını, uygulamalarında nasıl kullanacağını planlaması gerekmektedir. Gerekli önlemler alınmazsa kurumların bu veriyle başa çıkması zorlu olacaktır.
  - **Doğrulama (Verification):** Veriler hızlı bir şekilde gelip büyük bir hacim oluşturduğu için, verinin güvenilirliğinin önemi de artmaktadır. Verinin kaynağı bilinmeli ve yalnızca doğru kişilerle paylaşılmalıdır, istenmeyen kişilerden gizlenmelidir. Doğrulanmamış bilgilerin kullanılması, hem kullanıcılar hem de servis sağlayıcılar açısından sorunlara neden olabilir.
  - **Değer (Value):** Doğru ve güvenilir bir şekilde edinilmiş verinin servis sağlayıcılar için önemi büyüktür. Büyük veri birtakım üretim analiz işlemlerinden geçtikten sonra ortaya çıkan sonuçlar servis sağlayıcılar için önemlidir.
- **Büyük Veri Analitiği:** Farklı veri tiplerindeki büyük veriyi işleyerek anlamlı sonuçlar çıkarma işlemidir. Çeşitli tiplerdeki verileri içeren büyük veri setlerini işleyerek bu veri setlerinden çeşitli örüntüleri, veriler arasındaki ilişkileri, müşteri tercihlerini, market eğilimleri gibi yararlı iş bilgilerini açığa çıkarma işlemidir.
- **Büyük veri analitiğinin ana amacı**, büyük veriyi analiz ederek veri sahiplerine işleri hakkında **daha doğru kararlar verebilmelerine yardımcı** olmaktır. Büyük Veri İşlemleri, kendi içerisinde Veri Yönetimi ve Veri Analitiği olarak ikiye ayrılır.
- **Veri Yönetimi**
  - **Veri toplama ve kayıt:** İlk olarak verilerin, veri kaynağından toplanması gerekmektedir. Sosyal ağ, sensörler, ses kayıt cihaz vb. çeşitli kaynaklardan gelen veriler toplanmakta ve kayıt altına alınmaktadır.
  - **Çıkartım, temizleme ve belirtim:** Genellikle toplanan veri, analiz için hazır bir formatta değildir. Öncelikle veri analizinde kullanılabilir hale getirilmesi gerekir.
  - **Birleştirme ve gösterim:** Veriler genellikle farklı kaynaklardan geldikleri için, kullanıma hazır hale getirmek amacıyla öncelikle birleştirilmeleri gerekmektedir.
- **Veri Analitiği**
  - **Modelleme ve analiz:** Çeşitli veri modelleme teknikleri kullanılarak sıklıkla kullanılan örüntüler ve veriler arasındaki ilişkiler bulunabilir**.** Bu şekilde büyük veri üzerinden ilişkisel bilgi çıkarımı yapılır.
  - **Anlamlandırma:** Büyük veri analizinde son aşama modelleme ya da sorgu sonucunda açığa çıkan sonuçların anlamlandırılarak ilgili kişilere (örneğin; şirket yöneticisi, pazarlama direktörü vb.) sunulmasıdır.
- **Metin Analitiği:** Metin analitiği, bir diğer adıyla metin madenciliği, bir **metinden bilgi çıkarma tekniği**dir. Sosyal ağlardaki paylaşımlar, elektronik postalar, forumlara girilen yorumlar, çeşitli dokümanlar ve haberler şirketler tarafından kullanılan metin verisi örnekleridir. **Metin analitiğinde** çeşitli yöntemler kullanılarak **büyük veriden anlamlı bilgiler** çıkarılabilmektedir. **Metin analitiği örneklerinden birisi** de **metin özetleme**dir. Metin özetlemede bir ya da daha fazla doküman kullanılarak, orijinal büyük metnin önemli noktaları özetlenir. Metin analitiğinin bir diğer uygulaması da soru cevaplama teknikleridir.
- **Güven analizi ve Tercih madenciliği** de metin analitiği konularından biridir. Tercih madenciliğiyle **müşterilerden çeşitli bilgiler toplanıp analiz edilerek** şirketler için anlamlı bilgiler çıkarılmaktadır.
- **Ses analitiği**, ses verisinden bilgi çıkarma işlemi olarak tanımlanabilir. Ses analitiği genellikle çağrı merkezlerinde ya da sağlık hizmetlerinde kullanılmaktadır.
- **Video içerik analitiği**, videodan otomatik olarak anlamlı bilgi çıkarma işlemidir. Birincil kullanım alanı, güvenlik ve izleme sistemleridir. Diğer bir kullanım alanı da iş zekâsı uygulamalarıdır. Örneğin, **marketlerde hangi reyonlara daha fazla müşteri uğruyor**, bir ürüne hangi üründen sonra yöneliyor gibi…
- **Hadoop:** Büyük veri setlerinin dağıtık olarak saklanması ve üzerinde işlem yapılması için gerçekleştirilmiş Java dilinde yazılmış açık kaynak kodlu bir yazılım çerçevesidir. Temelde iki ana parçadan oluşmaktadır. İlk parça, **verinin saklandığı bölüm** olan Hadoop Dağıtık Dosya Sistemi (HDFS)'dir. İkinci parça ise **verinin işlendiği bölümdür** (MapReduce).
- **Hadoop'da HDFS katmanı** temelde iki parçadan oluşmaktadır. Bunlar **Ad Düğümü** ve **Veri Düğümü**'dür. Ad düğümünde verinin kendisi değil, veri hakkında bilgiler saklanmaktadır. Veri düğümü ise asıl verinin tutulduğu kısımdır.
- **MapReduce**, dağıtık mimari içerisinde bulunan **dosyaları paralel şekilde analiz ederek**, sonuçları çıkaran ve gönderen sistemlerdendir. İş ve Görev Takipçisi şeklinde iki parçadan oluşur. Yapılması gereken işler, **İş Takipçisi** ile yönetilir ve **Görev Takipçisi** ile gerçekleştirilir.
- **Hadoop dışında diğer büyük veri işleme projeleri:** Apache **Pig**, Apache **Hive**, Apache **Spark**, Apache **Storm**.
- **Anaçatı bilgisayar**, yüzlerce kullanıcıya eş zamanlı olarak farklı hizmetler verebilen, büyük işlem gücü, yüksek girdi-çıktı kapasiteli ve pahalı bir bilgisayardır. İçerisinde çok fazla sayıda işlemci ve sabit disk barındırır.
- **İlk gerçek ticarileşmiş bulut bilişim hizmeti** olan **Amazon S3**, **2006** yılında hizmete girmiştir.
- **Bulut bilişim sistemleri** erişim modellerine göre dört sınıfa ayrılır: 1) **Genel** bulut, 2) **Topluluk** bulut, 3) **Özel** bulut, 4) **Karma** bulut.
  - **Genel Bulut:** Depolama, yazılım ve diğer kaynaklar, hizmet sağlayan şirket tarafından, **kullandığın kadar öde modeli** ile ya da ücretsiz olarak **son kullanıcılara** sunulur. (Amazon EC2, Google App Engine, SalesForce.com)
  - **Topluluk Bulut:** Ortak ihtiyaçları olan **özel bir topluluk**taki çeşitli organizasyonlar arasında altyapının paylaşılmasına imkân verir.
  - **Özel Bulut:** **Sadece tek bir organizasyon** için işletilen bulut yapılarıdır. Altyapı, şirket tarafından veya dışarıdaki bir kurum tarafından yönetilebilir.
  - **Karma Bulut:** İki veya daha fazla özel, topluluk veya genel bulut yapılarının birleşmesinden oluşur. Bu tür yapılarda, kurumlar özel bulut mimarilerinde kendilerine ait özel bilgileri tutarlarken, yapı içindeki genel bulut yapısı ile son kullanıcıya ulaşırlar.

- **Bulut bilişimde her şey kullanıcıya hizmet olarak sunulur**. Servisler, belirli işleri yapması için otomatikleştirilmiş alt yordamların toplamı olarak tarif edilebilir. Servislerin hizmet olarak sunulmasını üç ana başlık altında toplayabiliriz.

BULUT BİLİŞİM	KULLANICI
Hizmet olarak Yazılım: E-posta, CRM, ERP	Son Kullanıcı
Hizmet olarak Platform: Yazılım Çalıştırma	Geliştiriciler
Hizmet olarak Altyapı: Önbellek, Ağ Altyapısı, Sunucu Yönetimi	IT Operatörleri

- **Bulut Bilişim Hizmet Modelleri**
  - **Hizmet olarak Altyapı (Infrastructure as a Service - IaaS):**
    Teknoloji firmaları veya geliştiriciler için **disk alanı** ya da hesap yapabilme gibi kaynakları sunar.
    (Bluelock, CSC, Go Grid, IBM, OpenStack, Rackspace, Savvis, Terremark, VMWare, **Amazon EC2**)
  - **Hizmet olarak Platform (Platform as a Service - PaaS):**
    Teknoloji firmaları ve geliştiriciler için bulut bilişime **hazır çözümler üretilmesini sağlayan yapıları**n sunulduğu modeldir.
    (Amazon Elastic Beanstalk, Microsoft Azure, Centurylink Appfog, CloudControl dotCloud, Engine Yard, Google App Engine, IBM Bluemix, Pivotal Cloud Foundry, Red Hat OpenShift, Salesforce, **Heroku**)
  - **Hizmet olarak Yazılım (Software as a Service - SaaS):**
    Belirli bir amaç için geliştirilmiş yazılımların son kullanıcılara sunulmasını sağlayan yapılardır. CRM, ERP, finans, muhasebe, e-posta gibi yazılımlar bulut üzerinden dağıtılır. (Salesforce, **Gmail**, Workday, Netsuite, ServiceNow, Athenahealth, Microsoft Office Online)