

# İşletim Sistemleri

## Bölüm2: İşletim Sistemi Yapıları

M.Ali Akcayol Gazi Üniversitesi Bilgisayar Müh. Ders sunularıdır

Bu dersin sunumları, "Abraham Silberschatz, Greg Gagne, Peter B. Galvin, Operating System Concepts 9/e, Wiley, 2013." kitabı kullanılarak hazırlanmıştır.

### Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

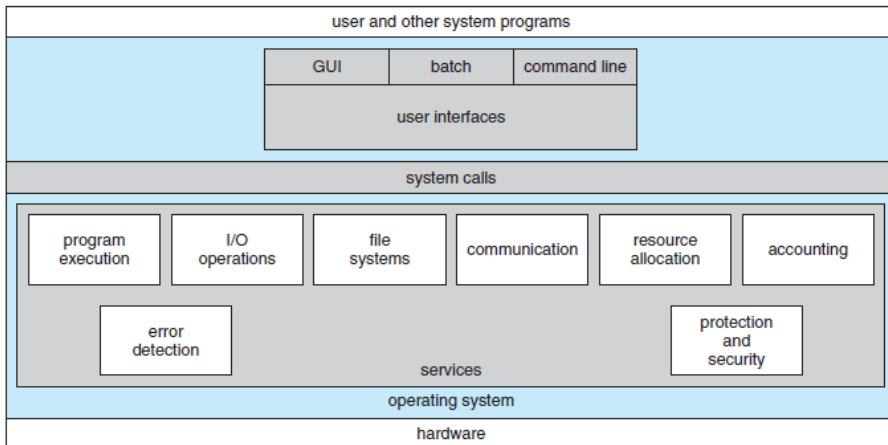
## İşletim sistemi servisleri

- İşletim sistemi uygulama programlarının programlarının çalışması için ortam sağlar.
- İşletim sistemleri, sağladığı servisler, arayüzler , sunduğu bileşenler ve bileşenler arasındaki ilişkilerle değerlendirilirler.
- İşletim sistemleri, programcıların programları kolay bir şekilde geliştirilebilmeleri için servisleri ve bileşenleri sağlar.
- İşletim sistemlerinin sağladığı servisler farklılık gösterir.

3

## İşletim sistemi servisleri

- İşletim sistemleri, ortak servislerle birlikte kendine özgü servislere de sahiptir.



4

## İşletim sistemi servisleri

- Kullanıcıya yardımcı olmak için sağlanan fonksiyonlar:
  - **Kullanıcı arayüzü:** Tüm işletim sistemleri kullanıcı arayüzüne sahiptir. Komut satırı arayüzü (**command-line interface - CLI**), **metin** komutları alır. Batch arayüzü (**batch interface**), **komutlar** dosya içerisinde sağlanır. Grafik kullanıcı arayüzü (**graphical user interface - GUI**), en **esnek** ve yaygın kullanılandır.
  - **Program çalıştırma:** Sistem bir programı hafızaya **yükleme**, **çalıştırma** ve **sonlandırma** (hatalı veya hatasız) işlemlerini yapar.
  - **I/O işlemleri:** Programlar I/O cihazlarına ihtiyaç duyabilirler. Kullanıcılar koruma ve etkinlik için doğrudan I/O cihazlarını kullanamazlar. İşletim sistemleri, **I/O cihazları için gerekli işlevleri sağlarlar**.
  - **Dosya sistemi işlemi:** Programlar dosyalara veya dizinlere erişim, okuma ve yazma yapmak isteyebilirler. İşletim sistemleri **dosya erişimlerini düzenleyen işlevleri sağlarlar**.

5

## İşletim sistemi servisleri

- Kullanıcıya yardımcı olmak için sağlanan fonksiyonlar:
  - **İletişim:** Bir process başka bir process'le haberleşmeye ihtiyaç duyabilir. Bu aynı bilgisayardaki process'ler arasında veya ağdaki bilgisayarlardaki process'ler arasında olabilir. **İletişim paylaşılmış hafıza (shared memory) alanlarında mesaj göndererek yapılabilir**.
  - **Hata denetimi:** İşletim sistemi oluşan hataları sürekli izlemek, raporlamak ve mümkünse düzeltmek zorundadır.

6

## İşletim sistemi servisleri

- Sistemin etkin çalışması için kullanılan fonksiyonlar:
  - **Kaynak tahsisi:** Çok sayıda kullanıcı veya iş aynı anda çalışır ve kaynakların bunlara tahsis edilmesi gereklidir.  
Bu kaynaklar, CPU time, hafıza, dosya, I/O cihazları olabilir.
  - **Hesap oluşturma:** İşletim sistemi, kaynakların kullanımını kullanıcı bazında tutabilir.  
Bu ücretlendirme veya istatistiksel çalışma için gerekli olabilir.
  - **Koruma ve güvenlik:** Sistem kaynaklarına kullanıcıların erişimi denetlenir ve kullanıcı bazında yetkilendirilir.

7

## Konular

- İşletim sistemi servisleri
- **Kullanıcı ve işletim sistemi arayüzü**
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

8

## Kullanıcı ve işletim sistemi arayüzü

- İşletim sistemleri, kullanıcılara iki farklı arayüz sağlarlar:

- **Command-line interface** (command interpreter)
  - Kullanıcı doğrudan komutları yazar.
  - Kullanımı ve öğrenmesi zordur.
- **Graphical user interface** (GUI)
  - Kullanıcı farklı bileşenlerle komutları oluşturabilir.
  - Kullanımı ve öğrenmesi kolaydır.

9

## Kullanıcı ve işletim sistemi arayüzü

### **Command interpreter** (Komut Yorumlayıcısı)

- Bazı işletim sistemleri, **command interpreter**'ı kernel kısmında bulundundur.
- **Windows** ve **UNIX** gibi işletim sistemleri ise command interpreter'a ayrı bir program olarak bakar.
- Bazı işletim sistemleri ise, **birden fazla command interpreter**'a sahiptir.
- **Interpreter**'lar **shell** olarak adlandırılır.
- Kullanıcılar, UNIX ve Linux'da Bourne Shell, C Shell, Korn Shell, ... gibi farklı kabuklardan birisini seçebilirler.
- **Command interpreter** tarafından **create, delete, list, print, copy, execute, ...** işlemleri yapılır.
- Command interpreter, komutlara ait **kodlara sahip olabilir** veya her komut için **ayrı sistem programı** olabilir.

10

## Kullanıcı ve işletim sistemi arayüzü

### GUI

- Bazı işletim sistemlerinin arayüzü, menülere sahiptir ve mouse tabanlı işlemlere izin veren **desktop** şeklinde oluşturulmuştur.
- Programlara, dosyalara, dizinlere, sistem fonksiyonlarına **icon** kullanılarak erişilebilir veya programlar çalıştırılabilir.
- Herhangi bir dizin (**folder**) içerisindeyken mouse ile menüye ulaşıp işlem yapılabilir.
- **İlk GUI, 1973 yılında Xerox Alto bilgisayarda kullanılmıştır.**
- İlk yaygın kullanılan GUI, Apple Macintosh bilgisayarlarda 1980'li yıllarda kullanılmaya başlanmıştır.
- Microsoft'un **Windows 1.0** işletim sistemi MS-DOS işletim sistemine arayüz eklenerek geliştirilmiştir.
- **Mobil cihazlar, parmak ile kullanılan arayüzlere sahiptir.**

11

## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- **Sistem çağrıları**
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

12

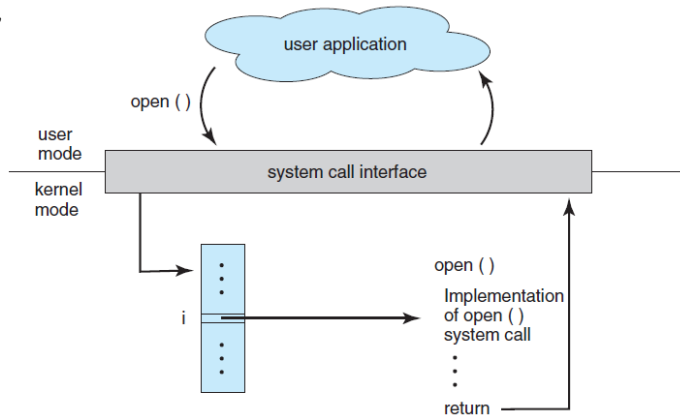
## Sistem çağrıları

- **Sistem çağrıları, servislerin işletim sistemi tarafından kullanılabilmesi için arayüz sağlarlar.**
- Bu çağrıların kullandığı program blokları genellikle C veya C++ ile yazılır. **Düşük seviyeli işlemler (donanım erişimi) için assembly dili kullanılır.**
- Bir dosyayı açıp (sistem çağrısı), içeriğini okumak (sistem çağrısı) ve kapatmak (sistem çağrısı) ayrı ayrı sistem çağrıları gerektir.
- İşlemler sırasında oluşan hatalar (dosya bulunamadı, okuma hatası, ...) sistem çağrıları tarafından yönetilir.
- **Uygulama geliştirici, işletim sistemindeki API** (application programming interface) (Windows API, POSIX API, Java API) **tarafından sağlanan fonksiyonları** (dosya aç, oku, yaz, ...) **kullanır.**
- **Programcı**, bu fonksiyonlara programlama dilinin sağladığı **kütüphaneler aracılığıyla erişir.**

13

## Sistem çağrıları

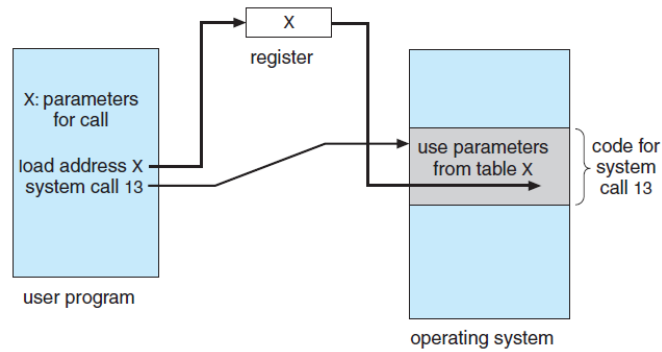
- **İşletim sistemi fonksiyonları, kernel'daki sistem çağrılarını başlatır.**
- Windows'daki CreateProcess() fonksiyonu, Windows kernel'daki NtCreateProcess() sistem çağrısını çalıştırır.
- Programlama dilleri, compiler tarafından sağlanan **kütüphaneler ile sistem çağrılarına erişmeyi kolaylaştıran arayüz sağlarlar.**



14

## Sistem çağrıları

- Sistem çağrılarına veri girişi gerekebilir (dosya adı, giriş cihazı, ...).
- Üç farklı yöntemle işletim sistemine parametre gönderilebilir:
  - Register'lar kullanılabilir.
  - Hafızada blok veya tablo kullanılabilir.
  - Stack kullanılabilir.




15

## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

16






## Sistem çağrı türleri

- Sistem çağrıları 6 grup altında sınıflandırılabilir:
  - Process control
  - File management
  - Device management
  - Information maintenance
  - Communications
  - Protection
- İşletim sistemlerinde desteklenen sistem çağrılarında farklılık olabilir, ancak gerçekleştirilen işlevler aynıdır.

17



## Sistem çağrı türleri

### Process control

- Bir program çalışmasını **normal bir şekilde** (end()) veya **beklenmeyen şekilde** (abort()) bitirebilir.
- Eğer bir **sistem çağrısı** programın çalışmasını sonlandırırsa, bir **hata mesajı** üretilir ve **hafızanın dökümü diske kaydedilir**.
- Hafıza dökümü **debugger** programı tarafından incelenir ve hataya neden olan **bug** düzeltilir.
- Bir programın çalıştırılması için **load()** ve **execute()** işlemleri yapılır.

18

## Sistem çağrı türleri

- Sistem çağrı türleri ve gerçekleştirilen işlevler aşağıda verilmiştir.

<ul style="list-style-type: none"> <li>• Process control <ul style="list-style-type: none"> <li>◦ end, abort</li> <li>◦ load, execute</li> <li>◦ create process, terminate process</li> <li>◦ get process attributes, set process attributes</li> <li>◦ wait for time</li> <li>◦ wait event, signal event</li> <li>◦ allocate and free memory</li> </ul> </li> <li>• File management <ul style="list-style-type: none"> <li>◦ create file, delete file</li> <li>◦ open, close</li> <li>◦ read, write, reposition</li> <li>◦ get file attributes, set file attributes</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Device management <ul style="list-style-type: none"> <li>◦ request device, release device</li> <li>◦ read, write, reposition</li> <li>◦ get device attributes, set device attributes</li> <li>◦ logically attach or detach devices</li> </ul> </li> <li>• Information maintenance <ul style="list-style-type: none"> <li>◦ get time or date, set time or date</li> <li>◦ get system data, set system data</li> <li>◦ get process, file, or device attributes</li> <li>◦ set process, file, or device attributes</li> </ul> </li> <li>• Communications <ul style="list-style-type: none"> <li>◦ create, delete communication connection</li> <li>◦ send, receive messages</li> <li>◦ transfer status information</li> <li>◦ attach or detach remote devices</li> </ul> </li> </ul>
---	--

## Sistem çağrı türleri

### Process control

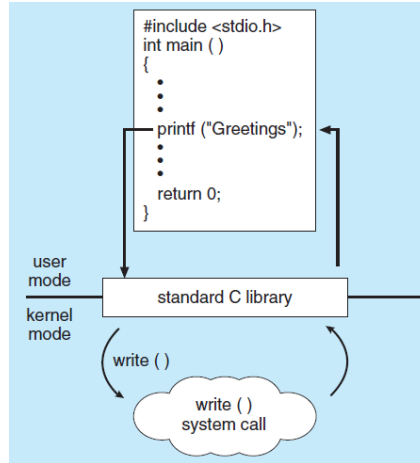
- Windows ve UNIX için örnek sistem çağrıları tabloda verilmiştir.

	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shm_open() mmap()
Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

## Sistem çağrı türleri

### Process control

- Bir C programının **printf()** deyimini standart C kütüphanesi ile çalıştırması için **sistem çağrısını kullanması gereklidir.**



21

## Sistem çağrı türleri

### Process control

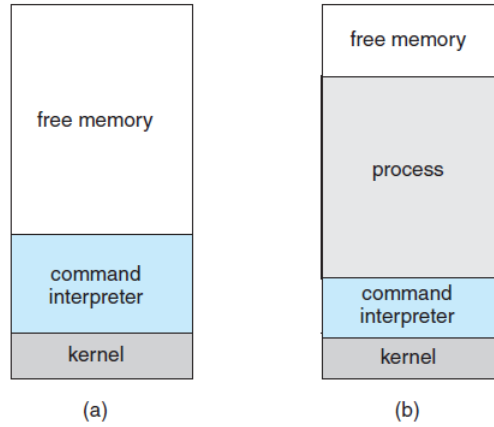
- Birden fazla program aynı veriyi paylaşabilir. Bu durumda kullanmakta olan **process veriyi kilitler (lock)** ve **diğer process'ler erişemez.**
- Kitleme için **acquire lock()**, serbest bırakmak için **release lock()** sistem çağrıları kullanılır.
- **Process kontrol işlemleri**, tek görevli (**single-tasking**) ve çok görevli (**multiple-tasking**) sistemlerde farklı gerçekleştirilir.
- **MS-DOS (Microsoft-Disk Operating System)**, single-tasking işletim sistemidir ve **bir process çalışırken yeni process başlatılamaz.**
- **MS-DOS'da bir program** hafızaya yerleştirilir ve instruction pointer (program counter register) ile **ilk komut gösterilerek çalıştırılır.**

22

## Sistem çağrı türleri

### Process control

- MS-DOS ile (a) başlangıç durumu ve (b) programın çalıştırılması şekilde görülmektedir.

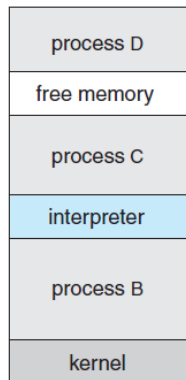


23

## Sistem çağrı türleri

### Process control

- [FreeBSD \(Berkeley Software Distribution\)](#), multi-tasking işletim sistemidir ve bir process çalışırken yeni process başlatılabilir.
- **Çok görevli işletim sistemlerinde, command interpreter** bir program çalışırken de **sürekli çalışmaktadır** ve yeni process'ler başlatabilir.



24

## Sistem çağrı türleri

### File management

- Dosya işlemleri de sistem çağrıları tarafından gerçekleştirilir.
- Dosya oluşturmak ve silmek için **create()** ve **delete()** sistem çağrıları kullanılır.
- Var olan dosyayı açmak için **open()**, dosyayı kapatmak için **close()** sistem çağrıları kullanılır.
- Dosyadan okuma yapmak veya dosyaya yazmak için **read()** ve **write()** sistem çağrıları kullanılır.
- Dosyaların özellikleri ve erişim haklarına erişmek ve değiştirmek için **get file attributes()** ve **set file attributes()** sistem çağrıları kullanılır.
- Dosyaların taşınması veya kopyalanması için **copy()** ve **move()** sistem çağrıları kullanılır.

25

## Sistem çağrı türleri

### Device management

- Bir **process**, çalışması sırasında **farklı kaynaklara ihtiyaç duyabilir**.
- İşletim sisteminin kontrol ettiği **tüm kaynaklar cihaz olarak düşünülebilir**. Bunlar **fiziksel** kaynaklar (disk sürücüleri) veya **sanal** cihazlar (dosya) olabilir.
- Bir kaynağa erişim isteği **request()** ile işinin tamamlandığı ise **release()** sistem çağrısı ile bildirilir.
- Bir kaynağa erişim hakkı alındığında, **read()** veya **write()** sistem çağrıları ile okuma ve yazma işlemleri gerçekleştirilir.

26

## Sistem çağrı türleri

### Information maintenance

- Birçok sistem çağrısı **kullanıcı programı ile sistem çağrıları arasında veri transferi** yapmak için kullanılır.
- Örneğin **time ()** ve **date ()** sistem çağrıları kullanıcı programına anlık saat ve tarih bilgilerini aktarır.
- Diğer sistem çağrıları, **anlık kullanıcı sayısı, işletim sistemi versiyonu, boş hafıza veya disk alanı** gibi sisteme ait bilgileri aktarır.
- **dump ()** gibi bazı sistem çağrıları ise bir programın debug aşamasında faydalıdır.
- Çoğu işletim sistemi, çalışan programları periyodik aralıklarla izler ve durumunu saklar. **Bunun için timer interrupt'ları kullanılır.**

27

## Sistem çağrı türleri

### Communication

- Process'ler arasında iletişim için iki yöntem kullanılır:
  - **Message-passing model**
    - Mesajlar process'ler arasında doğrudan veya dolaylı (mesaj kutusu) gönderilebilir.
    - **Her process'in bir process adı ve ID değeri vardır.**
    - **open connection ()** and **close connection ()** sistem çağrıları kullanılır.
    - Her process gelen mesajları kabul etmek için **accept connection ()** sistem çağrısını kullanır.
    - **Kaynak process istemci (client), hedef process sunucu (server)** olarak adlandırılır.
  - **Shared-memory model**
    - Hafıza alanı oluşturmak için **shared memory create ()** ve erişmek için **shared memory attach ()** sistem çağrıları kullanılır.
  - **İşletim sistemi, paylaşılan alanın yönetimini yapar, veri içeriğini kontrol etmez.**

28

## Sistem çağrı türleri

### Protection

- **Protection, bilgisayar sistem kaynaklarına erişimi kontrol eden mekanizmaları sağlar.**
- **Protection, çok kullanıcı ve multiprogramming sistemler için geliştirilmiştir.**
- Günümüzde, mobil cihazlar da dahil tüm cihazlar için protection gereklidir.
- Sistemdeki kaynaklara erişim yetkilendirmesi için **set permission()** ve **get permission()** sistem çağrıları kullanılır.
- İzin vermek veya izni kaldırmak için **allow user()** ve **deny user()** sistem çağrıları kullanılır.

29

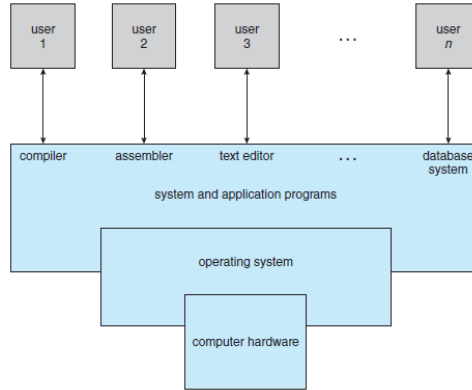
## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- **Sistem programları**
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

30

## Sistem programları

- Modern bilgisayar sistemlerinde **en alt seviyede donanım vardır.**
- Donanımın üzerinde **işletim sistemi** çalışır.
- **İşletim sisteminin üzerinde ise sistem programları vardır.**
- **En üstte ise uygulama programları vardır.**



31

## Sistem programları

- Sistem programları (**system utilities**), **program geliştirme ve çalıştırma için kolay bir platform sağlar.**
  - **File management**
    - Dosya ve dizinlerde, **create, delete, copy, rename, print, list, ...** gibi işlemleri gerçekleştirir.
  - **Status information**
    - Bazı programlar sistemden **tarih, saat, hafıza ve disk boş alanı, kullanıcı sayısı** gibi bilgileri ister.
    - Daha karmaşık programlar ise, **performans, log, debug** gibi bilgileri ister.
    - Bazı sistemler sistem konfigürasyonunu saklayan **registry** yapısına sahiptir.
  - **File modification**
    - Disk veya diğer saklama birimlerindeki **dosyaların içeriğini görüntüleme ve değiştirme** amacıyla kullanılan **metin editörleri** vardır.
  - **Programming-language support**
    - Programlama dilleri (C, C++, Java, Perl) için **compiler, assembler, debugger, interpreter** sağlarlar veya ayrıca download edilerek kullanılmasına izin verirler.

32



## Sistem programları

- **Program loading / execution**
  - Bir programı derledikten sonra **hafızaya yükleyip çalıştırmak için** kullanılırlar.
  - Makine dilinde veya yüksek seviyeli dilde **debug amacıyla için programlar kullanılabilir.**
- **Communications**
  - Bu sistem programları, **process'ler, kullanıcılar ve bilgisayarlar** arasında **bağlantı** kurulması sağlarlar.
  - Kullanıcılar arasında **mesajla gönderme, Web sayfalarında gezinti, e-posta gönderme, uzak bağlantı** veya **dosya aktarımları** yapmayı sağlarlar.
- **Background services**
  - Tüm genel amaçlı **sistemler boot sırasında bazı sistem programlarını başlatırlar.**
  - Bu programların **bazıları** sistem boot edildikten sonra **sonlandırılır, bazıları** ise sistem çalıştığı sürece **çalışmasını sürdürür.**
  - Bu tür sistem programları, **service, subsystem veya daemon** olarak adlandırılır.
  - **Ağ bağlantısı, yazıcı işlemleri** veya **sistem hata izleme** servislerini sistem programları sağlarlar.

33

## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- **İşletim sistemi tasarımı ve gerçekleştirimi**
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

34

## İşletim sistemi tasarımı ve gerçekleştirimi

### Tasarım hedefleri

- En üst düzeyde **bir işletim sisteminin tasarım hedefleri, donanım özellikleri ile sistem türünü** (single user, multiuser, gerçek zamanlı, dağıtık, özel amaçlı, genel amaçlı, ...) **belirler.**
- Gereksinimler, **kullanıcı hedefleri (user goals)** ve **sistem hedefleri (system goals)** olarak iki gruba ayrılır.
- Kullanıcı, sistemin kolay ve rahat kullanılmasını, kolay ve hızlı öğrenilmesini, güvenilir, güvenli ve hızlı olmasını ister.
- **Kullanıcı gereksinimlerine göre sistem gereksinimlerinin belirlenmesi ve bu hedeflerin gerçekleştirilmesi gereklidir.**
- Bir işletim sisteminin tasarımı ile ilgili genel prensipler, **software engineering** alanındaki çalışmalarla belirlenmiştir.

35

## İşletim sistemi tasarımı ve gerçekleştirimi

### Mekanizmalar ve kurallar

- Kurallar (**policies**) **ne yapılacağını**, mekanizmalar (**mechanisms**) **nasıl yapılacağını** belirler.
- Örneğin, **CPU'nun korunması için timer kullanımı mekanizmadır**, bir kullanıcı için **timer süresinin ne kadar olacağına karar vermek kuraldır.**
- Kurallar, kaynak ayrılmasına yönelik kararlarda oldukça önemlidir.

36

## İşletim sistemi tasarımı ve gerçekleştirimi

### Implementation

- Bir işletim sistemi tasarlandıktan sonra gerçekleştirilme aşamasına geçilir.
- İlk işletim sistemleri **assembly dili** ile yazılmıştır.
- Günümüzde işletim sistemleri C ve C++ gibi **yüksek seviyeli dillerle yazılmaktadır**.
- **Kernel kısmı assembly, yüksek seviyeli işlemler C, sistem programları C, C++, Perl, Python** gibi dillerle yazılmaktadır.
- Linux işletim sisteminde tüm dillerle yazılan kısımlar bulunmaktadır.

37

## İşletim sistemi tasarımı ve gerçekleştirimi

### Implementation

- **Bir işletim sisteminin yüksek seviyeli dillerle yazılması, farklı mikroişlemcilerde çalışabilmesini kolaylaştırır.**
- MS-DOS, 8088 assembly diliyle yazılmıştır ve sadece Intel X86 işlemcilerde çalışabilir.
- Linux, C ile yazılmıştır ve Intel X86, Oracle SPARC ve IBM PowerPC işlemcilerde çalışabilir.
- **Yüksek seviyeli dillerde yazılan işletim sistemleri daha yavaştır ve daha fazla depolama alanına ihtiyaç duyarlar.**
- Günümüzdeki **compiler'lar** derleme sırasında **optimizasyon** yapar ve **mikroişlemciler pipelining** gibi teknolojilerle çalışma performansı artırır.

38



## Konular

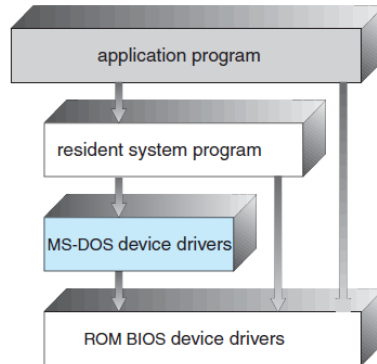
- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- **İşletim sistemi yapısı**
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

39

## İşletim sistemi yapısı

### Basit yapı

- **Çoğu işletim sistemi, küçük, basit ve sınırlı bir şekilde geliştirilmeye başlar, daha sonra gelişir.** MS-DOS bu şekilde bir işletim sistemidir.
- Başlangıçta az sayıda kişi tarafından geliştirilmiş ve **modüler yapı dikkatli bir şekilde oluşturulmamıştır.**



40

## İşletim sistemi yapısı

### Basit yapı

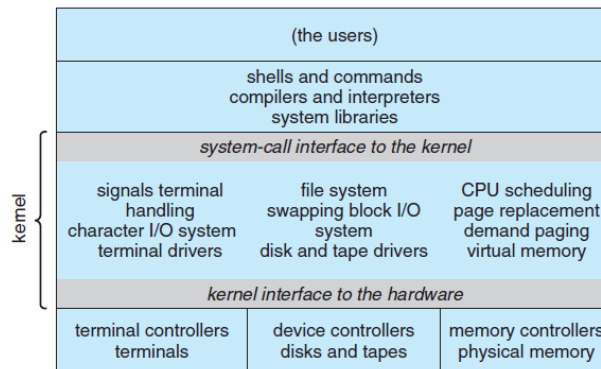
- **MS-DOS işletim sisteminde, arayüzler ve işlev seviyeleri iyi bir şekilde oluşturulmamıştır.**
- MS-DOS'ta bir **uygulama programı** temel **I/O rutinlerine erişip display veya disk sürücülerini** kontrol edebilir.
- MS-DOS'un bu özelliği, kötücül programların sisteme zarar vermesine ve tümüyle çalışmaz hale gelmesine neden olabilmektedir.

41

## İşletim sistemi yapısı

### Basit yapı

- **UNIX işletim sistemi, ilk geliştirildiğinde kernel ve sistem programları şeklinde iki parçadan oluşmaktaydı.**
- Kernel kısmı arayüzler ve cihaz sürücüler şeklinde alt parçalara bölünerek gelişmesine devam etmiştir.

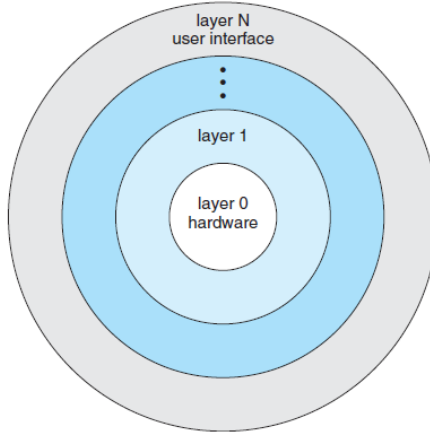


42

## İşletim sistemi yapısı

### Katmanlı yapı

- İşletim sistemi, modüler yapıda birden fazla katman halinde geliştirilir.
- En alt katman donanım, en üst katman ise kullanıcı arayüzüdür.



43

## İşletim sistemi yapısı

### Katmanlı yapı

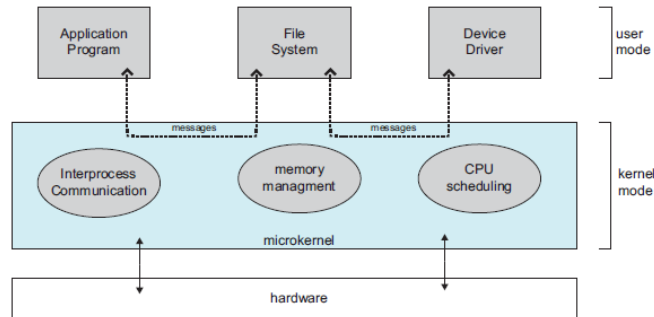
- Katmanlı yapıda, **tasarım ve implementation kolay yapılır.**
- **Her katman ayrı ayrı debug edilip doğrulama yapılabilir.**
- Her katman alt katman tarafından sağlanan işlemler kullanılarak oluşturulur.
- **Katmanlı yapıda her katmanda yapılacak işlevlerin çok iyi planlanması gereklidir.**
- Katmanlı yapıda, her katman alt katmandaki fonksiyonu çalıştıracağından dolayı performans düşme eğilimindedir.
- Katman sayısını olabildiği kadar az olacak şekilde tasarım yapmak performans açısından gereklidir.

44

## İşletim sistemi yapısı

### Mikrokernel

- UNIX geliştikçe, kernel kısmı büyümüş ve yönetimi zorlaşmıştır.
- 1980 yılında Carnegie Mellon Üniversitesinde mikrokernel yaklaşımıyla Mach adında işletim sistemini geliştirmişlerdir.
- Bu yaklaşımda, kernel içerisindeki gereksiz tüm bileşenler sistem programlarına aktarılmıştır.



45

## İşletim sistemi yapısı

### Mikrokernel

- Mikrokernel kısmında, iletişim bileşenleri, hafıza yönetimi ve küçük process'ler kalmıştır.
- Process'ler arasındaki iletişim mesaj gönderimi ile yapılmıştır.
- Mikrokernel yapısında çoğu servis kullanıcı olarak çalıştığından (kernel servisi olmadığından) güvenlik daha iyi sağlanabilmektedir.
- Mac OS X kernel'ı kısmen mikrokernel yapısını kullanmaktadır.
- Windows NT 4.0 kernel yapısına sahiptir.
- Windows XP ile birlikte ağırlıklı olarak monolithic yapıya (tek parça) dönmüştür.

46

## İşletim sistemi yapısı

### Modül

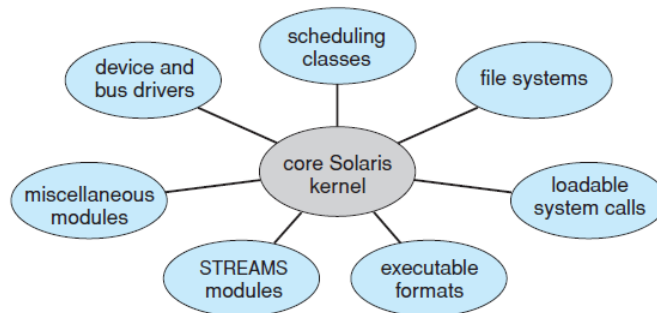
- İşletim sistemi tasarımında günümüzdeki **en iyi teknoloji yüklenebilir kernel modülleri kullanmaktır.**
- Kernel, **boot sırasında** veya **sistemin çalışması devam ederken yüklenen bileşenlere sahiptir.**
- Modern işletim sistemlerinde, (UNIX, Linux, Mac OS X, Windows) bu tür çalışma yaygındır.
- **CPU yönetimi ve hafıza yönetimi doğrudan kernel kısmındadır**, farklı dosya sistemleri gibi destek bileşenleri yüklenebilir kernel modülleri ile sağlanmaktadır.

47

## İşletim sistemi yapısı

### Modül

- Bir modül kendisi yüklendikten sonra başka modülleri çağırabilir veya iletişime geçebilir.
- Linux, **cihaz sürücülere ve dosya sistemleri için** yüklenebilir kernel modüllerini kullanır.



48



## İşletim sistemi yapısı

### Hibrit

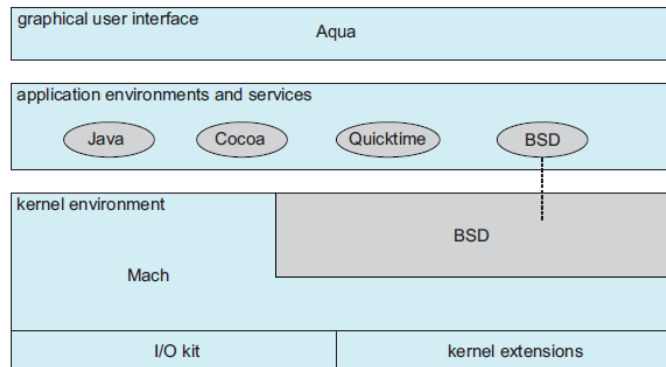
- Günümüzde işletim sistemleri, tek yapıya bağlı bir şekilde geliştirilmemektedir.
- **Farklı yapıları birleştirerek kullanıp; performans, güvenlik ve kullanılabilirliği artırmak amaçlanmıştır.**
- **Linux ve Solaris monolithic yapıdadır**, ancak kernel kısmına yeni fonksiyonlar dinamik olarak eklenebilmektedir.
- **Windows monolithic yapıdadır**, ancak bazı temel işlevleri mikrokernel yapısında destekler.

49

## İşletim sistemi yapısı

### Mac OS X

- Apple **Mac OS X** işletim sistemi **hibrit yapıya sahiptir.**
- En üst katmanda Aqua kullanıcı arayüzü vardır.
- İkinci katmanda, uygulama geliştirme platformları ve servisler vardır.



50

## İşletim sistemi yapısı

### Mac OS X

- **Cocoa, Objective C** programlama diline API sağlar.
- **Kernel**, BSD kernel ile Mach kernel'ına sahiptir.
- **Mach**, hafıza yönetimi, process'ler arası iletişim, çağrı başlatma, thread yönetimi gibi işlevleri sağlar.
- **BSD**, command-line, ağ ve dosya sistemleri gibi işlevleri sağlar.
- **I/O kit** cihaz sürücülerini sağlar.
- **Kernel extensions**, yüklenebilir modülleri sağlar.

51

## İşletim sistemi yapısı

### iOS

- **iOS**, Mac OS X üzerine yapılandırılmıştır ve Apple iPhone, tablet ve iPad ürünlerinde çalışmak üzere tasarlanmıştır.
- **Cocoa Touch**, Objective C programlama dili için API sağlar.
- **Media services**, grafik, video ve ses servislerini sağlar.
- **Core services**, cloud computing ve veritabanı servislerini sağlar.
- **Core OS**, en alt katmandır ve kernel işlevlerini sağlar.

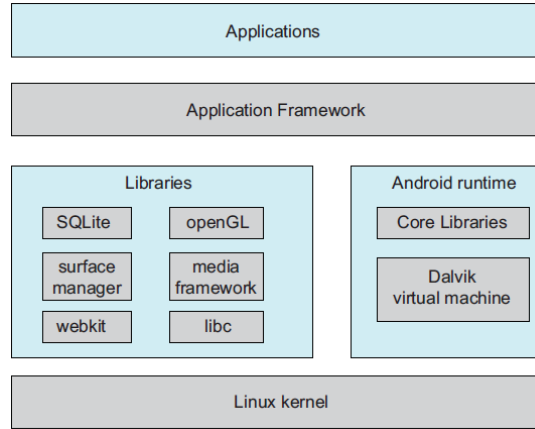


52

## İşletim sistemi yapısı

### Android

- **Google Android**, akıllı telefonlar ve tablet bilgisayarlar için tasarlanmıştır.
- Android işletim sistemi açık kaynaklıdır ve farklı platformlarda çalışabilir.



53

## İşletim sistemi yapısı

### Android

- En alt katmanda, **Linux kernel** çalışır. **Hafıza yönetimi, process yönetimi, cihaz sürücüler ve power management işlemleri için kullanılır.**
- **Anroid runtime**, Java programlama diliyle uygulama geliştirilmesi için gerekli kütüphaneleri sağlar.
- **Dalvik virtual machine**, Java executable dosyalarını çalıştırır.
- Kütüphaneler ise farklı uygulamaların geliştirilmesi için gerekli olabilecek bileşenleri bulundurur.
  - Web browser için **webkit**
  - veritabanı işlemleri için **SQLite**
  - multimedia ve oyunlar için **media framework** ve **OpenGL**
  - standart C kütüphanesine benzeyen **libc**
  - ekran erişimlerinin yönetimi için **surface manager**

54

## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

55

## İşletim sistemi debugging

- **Debugging**, genel olarak sistemdeki hataların bulunması ve giderilmesi işlemlerini ifade eder.
- **Debugging**, bug'lar ile ortaya çıkan performans sorunlarını gidermeyi amaçlar.
- Bu yüzden, **performans ayarı (performance tuning)**, konusunu da içerisine almaktadır.

56

## İşletim sistemi debugging

### *Hata analizi*

- Bir process hata verdiğinde, işletim sistemleri **log** dosyasına hata bilgisini kaydeder.
- İşletim sistemi aynı zamanda hafızanın anlık durumunu da (**memory dump**) kaydedebilir.
- Kernel'da ortaya çıkan hata **crash** olarak adlandırılır.
- **Debugging, bir kodu yazmaktan çok daha zordur.**

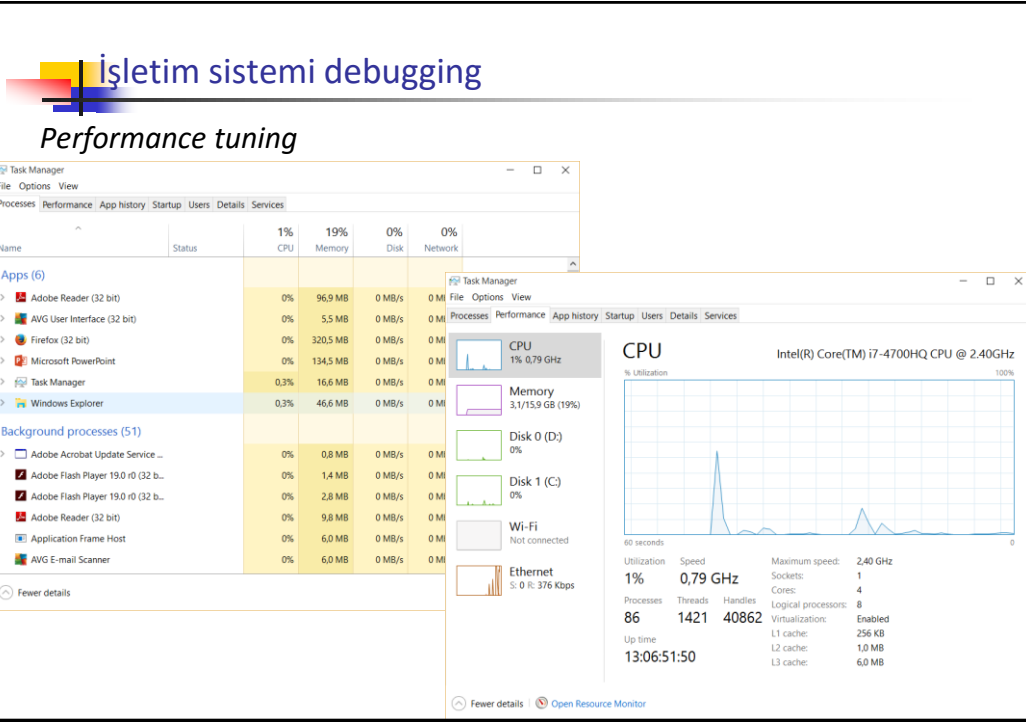
57

## İşletim sistemi debugging

### *Performance tuning*

- Sistem performansını artırmak için yapılan işlemlerdir.
- Sistemdeki tıkanıklıkların monitör edilerek belirlenmesi gereklidir.
- **Windows Task Manager**, sistemin izlenmesi ve performans için kullanıcı etkileşiminin sağlanması amacıyla kullanılmaktadır.

58



- ## Konular
- İşletim sistemi servisleri
  - Kullanıcı ve işletim sistemi arayüzü
  - Sistem çağrıları
  - Sistem çağrı türleri
  - Sistem programları
  - İşletim sistemi tasarımı ve gerçekleştirimi
  - İşletim sistemi yapısı
  - İşletim sistemi debugging
  - İşletim sistemi üretilmesi
  - Sistem boot

## İşletim sistemi üretilmesi

- İşletim sistemleri, farklı makinelerde ve farklı konfigürasyonlarda çalışmak üzere tasarlanırlar.
- İşletim sistemleri, **disklerde, CDROM'larda, DVD-ROM'larda** veya **ISO image** olarak dağıtılırlar.
- İşletim sisteminin yüklenmesi için ayrı bir yazılım çalıştırılır ve **aşağıdaki bilgileri kullanıcının sağlaması gereklidir:**
  - Kullanılan CPU
  - Disk formatı ve partition'ları
  - Kullanılacak hafıza miktarı (Bazı sistemler hafıza erişimini denetler.)
  - Cihazların interrupt bilgileri (Günümüzdeki çoğu sistem bu bilgileri kendisi oluşturur.)
  - Hangi işletim sisteminin tercih edildiği (donanım kapasitesine göre belirlenebilir.)

61

## Konular

- İşletim sistemi servisleri
- Kullanıcı ve işletim sistemi arayüzü
- Sistem çağrıları
- Sistem çağrı türleri
- Sistem programları
- İşletim sistemi tasarımı ve gerçekleştirimi
- İşletim sistemi yapısı
- İşletim sistemi debugging
- İşletim sistemi üretilmesi
- Sistem boot

62

## Sistem boot

- Kernel'ın yüklenerek bilgisayarın başlatılmasına **booting** denilmektedir.
- **Bootstrap program**, **ROM (Read Only Memory)** üzerindedir ve işletim sisteminin kernel kısmını yükler.
- Bilgisayar reset yapıldığında veya yeni açıldığında **instruction register (program counter - PC)** önceden tanımlı noktaya geçer ve buradan itibaren çalışma başlar.
- **Mobil cihazlarda** işletim sisteminin tamamı **EPROM (erasable programmable read-only memory)** üzerinde saklanır.
- Bu tür **ROM'lar firmware** olarak adlandırılır.
- **Büyük işletim sistemlerinde**, **bootstrap** programı **ROM'da**, **işletim sistemi** ise **disk** üzerinde saklanır.
- Boot kısmına sahip olan disk, **boot disk** veya **system disk** olarak adlandırılır.

63

## Ödev

- İşletim sistemlerinde kernel debugging hakkında araştırma ödevi hazırlayınız.

64