**Monolithickernel** mengintegrasikan seluruh fungsi OS ke dalam satu blok kode tunggal yang berjalan di ruang kernel (kernel space). Semua layanan, termasuk driver perangkat, scheduler proses, dan protokol jaringan, dijalankan sebagai bagian dari kernel yang sama. Komunikasi antar-komponen dilakukan melalui panggilan fungsi langsung, yang membuatnya sangat efisien dan cepat karena minim overhead. Keuntungannya adalah performa tinggi untuk operasi intensif, seperti multitasking di server. Namun, kekurangannya signifikan: kurangnya isolasi berarti bug atau kesalahan di satu modul (misalnya, driver rusak) dapat menyebabkan crash seluruh sistem (kernel panic). Struktur ini juga menyulitkan pengembangan dan debugging karena ketergantungan ketat antar-komponen, membuatnya kurang modular dan rentan terhadap serangan keamanan.

**Microkernel** dirancang minimalis, di mana kernel hanya menangani fungsi esensial seperti komunikasi antar-proses (IPC), manajemen thread dasar, dan penjadwalan sederhana. Layanan lain—seperti file system, driver, dan server jaringan—dijalankan sebagai proses terpisah di ruang pengguna (user space). Komunikasi dilakukan melalui mekanisme pesan IPC, yang memerlukan konteks switching lebih sering. Perbedaan utama dengan monolithic adalah modularitas tinggi: kegagalan satu layanan tidak memengaruhi kernel inti, memungkinkan restart modular dan peningkatan keamanan melalui isolasi. Ini mendukung reliabilitas dan portabilitas, ideal untuk sistem kritis. Namun, overhead IPC menyebabkan performa lebih rendah, terutama untuk tugas yang memerlukan interaksi cepat, membuatnya kurang cocok untuk aplikasi berat.

**Layered Architecture** membagi OS menjadi hierarki lapisan abstrak, di mana setiap lapisan menyediakan layanan kepada lapisan di atasnya dan bergantung pada lapisan di bawahnya. Lapisan terbawah berinteraksi langsung dengan hardware (misalnya, manajemen interrupt dan I/O dasar), sementara lapisan atas menangani abstraksi tinggi seperti antarmuka pengguna atau aplikasi. Komunikasi bersifat vertikal: permintaan dari lapisan atas diteruskan ke bawah melalui protokol antar-lapisan, dan hasil dikembalikan secara bertahap. Berbeda dengan monolithic yang datar dan microkernel yang terdistribusi, model ini menekankan abstraksi bertingkat, memudahkan pemeliharaan karena perubahan di satu lapisan jarang memengaruhi yang lain. Keuntungannya adalah struktur jelas dan modularitas relatif, yang mendukung pengujian terpisah. Kekurangannya adalah overhead performa akibat traversal multiple lapisan (bottleneck), serta ketergantungan linier yang bisa menyulitkan optimasi paralel. Secara keseluruhan, perbedaan mendasar terletak pada tingkat integrasi: monolithic prioritas efisiensi tapi rentan; microkernel fokus isolasi tapi lambat; layered menawarkan keseimbangan abstraksi tapi berpotensi inefisien. Pilihan model tergantung kebutuhan—performa vs. keamanan—dan sering diadopsi secara hybrid di OS modern untuk menggabungkan kelebihan masing-masing.

1. **Monolithickernel**

**Kelebihan:** Performa tinggi karena komunikasi antar-komponen cepat (langsung di memori kernel), tanpa overhead konteks switching.

**Kekurangan:** Kurang modular; kesalahan di satu komponen bisa meruntuhkan seluruh kernel. Sulit dikembangkan dan debug karena ukurannya besar.

**Fokus:** Efisiensi dan kesederhanaan implementasi.

1. **Microkernel**

**Kelebihan:** Modularitas tinggi; lebih aman dan andal karena kegagalan satu layanan tidak memengaruhi kernel inti. Mudah dikustomisasi dan portabel.

**Kekurangan:** Performa lebih rendah karena overhead komunikasi (pesan passing) antar-modul di user space.

**Fokus:** Keamanan, keandalan, dan fleksibilitas untuk sistem real-time atau embedded.

1. **Layered Architecture**

**Kelebihan:** Modularitas sedang; memudahkan pengembangan karena perubahan di lapisan bawah tidak langsung memengaruhi atas. Lebih terstruktur daripada monolithic.

**Kekurangan**: Performa bisa menurun karena ketergantungan lapisan (overhead traversal); kesalahan di lapisan bawah bisa memengaruhi seluruh sistem. Kurang fleksibel untuk perubahan dinamis.

**Fokus:** Struktur dan abstraksi bertahap, sering digunakan untuk sistem besar dan kompleks.

**Contoh nyata yang menggunakan masing-masing model**

1. **Monolothickernel: Linux, Unix, FreeBSD, OpenBSD, NetBSD**
2. **Microkernel: MINIX3, QNX, GNUHurd, Mach**
3. **Layered architecture: The Operating System, Multics, Windows NT**