Monolithic Kernel adalah model di mana seluruh komponen kernel—termasuk driver perangkat, sistem file, jaringan, dan scheduler—berjalan dalam satu ruang memori yang sama (kernel space). Semua layanan diintegrasikan secara ketat, sehingga komunikasi antar-komponen sangat cepat melalui panggilan fungsi langsung tanpa overhead konteks switching. Keuntungannya adalah efisiensi tinggi dan latensi rendah, ideal untuk aplikasi yang memerlukan performa maksimal. Namun, kekurangannya adalah kurangnya modularitas: jika satu driver crash, seluruh kernel bisa gagal, menyebabkan sistem tidak stabil. Model ini juga sulit untuk dikembangkan karena kode yang besar dan saling bergantung, meskipun dukungan untuk modul loadable membantu mengurangi masalah ini.

Contoh OS yang menerapkan monolithic kernel adalah Linux dan Unix tradisional, termasuk varian seperti FreeBSD. Linux, misalnya, menggunakan kernel monolithic dengan modul yang dapat dimuat secara dinamis untuk menambah fleksibilitas, tetapi tetap mempertahankan integrasi ketat untuk kecepatan, mendukung ekosistem Android hingga superkomputer.

Microkernel, sebalaniknya, meminimalkan ukuran kernel dengan hanya menyertakan fungsi dasar seperti inter-process communication (IPC), manajemen thread, dan penjadwalan dasar di kernel space. Komponen lain seperti driver, sistem file, dan server jaringan dijalankan sebagai proses user space terpisah. Komunikasi dilakukan melalui pesan IPC, yang menambah overhead tapi meningkatkan modularitas dan keamanan. Jika satu layanan gagal, yang lain tetap berfungsi, memudahkan isolasi kesalahan dan pemeliharaan. Kekurangannya adalah performa yang lebih lambat karena switching konteks yang sering, meskipun kemajuan seperti virtualisasi telah mengurangi gap ini, dan bahasa pemrograman modern seperti Rust mulai digunakan untuk mengurangi bug memori.

Contohnya termasuk Minix (dipopulerkan oleh Andrew Tanenbaum sebagai prototipe pengajaran), QNX (untuk sistem real-time seperti otomotif dan medis), L4 microkernel family (digunakan di seL4 untuk keamanan tingkat tinggi), serta elemen Mach di macOS yang hybrid.

Layered Architecture membagi OS menjadi lapisan-lapisan hierarkis, di mana setiap lapisan bergantung pada lapisan di bawahnya dan menyediakan abstraksi untuk lapisan atas. Lapisan terbawah berinteraksi langsung dengan hardware, sementara lapisan atas menangani antarmuka pengguna. Ini mirip model OSI untuk jaringan, dengan prinsip enkapsulasi yang membuat sistem lebih terstruktur dan mudah di-debug. Keuntungannya adalah modularitas sedang dan kemudahan pemeliharaan, tapi bisa menimbulkan bottleneck karena alur data harus melewati banyak lapisan, mengurangi efisiensi. Selain itu, perubahan di lapisan bawah memerlukan penyesuaian berantai.

Contoh klasik adalah THE OS (dibuat oleh Edsger Dijkstra pada 1960-an untuk komputer elektro) dan Multics (pendahulu Unix). Saat ini, elemen layered terlihat di beberapa OS embedded atau hybrid seperti IBM OS/360, serta bagian dari Windows NT yang menggunakan lapisan untuk manajemen I/O.

Dalam analisis relevansi untuk sistem modern, monolithic kernel tetap paling dominan dan relevan, terutama untuk desktop, server, dan cloud computing. Linux, sebagai monolithic, mendukung miliaran perangkat karena keseimbangan performa dan skalabilitasnya—modul dinamis memungkinkan adaptasi tanpa mengorbankan kecepatan. Menurut survei Stack Overflow 2023, Linux digunakan di 80% server global, menunjukkan ketahanannya di era AI dan big data di mana latensi rendah krusial.

Microkernel lebih relevan untuk domain khusus seperti IoT, real-time systems (misalnya, autonomous vehicles di QNX), dan keamanan tinggi (seperti di militer dengan seL4, yang diverifikasi secara formal). Dengan tren virtualisasi dan containerization (Docker, Kubernetes), microkernel mendukung isolasi yang lebih baik, mengurangi serangan seperti Spectre/Meltdown. Namun, overhead IPC masih menjadi hambatan untuk aplikasi umum, meskipun unikernel (seperti unikernel berbasis microkernel) sedang naik daun untuk cloud-native apps.

Layered architecture kurang relevan hari ini karena kurang fleksibel dibanding hybrid models. Banyak OS modern seperti Windows NT menggunakan pendekatan hybrid: monolithic untuk performa inti, tapi dengan elemen microkernel untuk driver (via Windows Driver Model) dan layered untuk abstraksi. Tren masa depan menuju microkernel atau hybrid untuk edge computing dan 5G, di mana keamanan dan reliabilitas lebih penting daripada kecepatan mentah. Secara keseluruhan, monolithic unggul untuk mayoritas kasus, tapi microkernel akan tumbuh seiring tuntutan keamanan siber yang meningkat, didorong oleh regulasi seperti GDPR dan kemajuan hardware seperti ARM untuk embedded systems.