Perbedaan mendasar antara Monolithic Kernel, Microkernel, dan Layered Architecture terletak pada bagaimana fungsionalitas inti sistem operasi (OS) diorganisasi dan diimplementasikan.

Pada Monolithic Kernel, semua layanan OS esensial mulai dari manajemen proses, manajemen memori, hingga device driver dimuat dan berjalan bersama dalam satu ruang alamat kernel yang besar. Keuntungan utamanya adalah kecepatan tinggi karena komunikasi antar komponen sangat langsung dan efisien. Namun, kelemahannya adalah kerentanan; kegagalan pada satu komponen, seperti driver, dapat menyebabkan crash seluruh sistem. Contoh populer dari arsitektur ini adalah Linux.

Sebaliknya, Microkernel mengadopsi pendekatan minimalis. Kernelnya sengaja dibuat sangat kecil dan hanya menangani fungsi paling dasar seperti IPC (Inter-Process Communication) dan manajemen memori dasar. Mayoritas layanan OS lainnya dipindahkan ke ruang alamat pengguna dan berjalan sebagai proses terpisah (server). Hal ini memberikan stabilitas dan modularitas yang jauh lebih baik, karena kegagalan satu layanan tidak akan merusak kernel utama. Namun, kinerjanya cenderung lebih rendah daripada Monolithic Kernel karena adanya overhead yang signifikan dari pertukaran pesan (IPC) yang konstan antar proses. QNX adalah salah satu contoh Microkernel yang sering digunakan.

Sementara itu, Layered Architecture adalah model struktural yang mengatur OS menjadi lapisan-lapisan hierarkis. Setiap lapisan hanya boleh berinteraksi dengan lapisan di bawahnya dan/atau di atasnya, menciptakan pemisahan fungsionalitas yang bersih. Keuntungan utama dari arsitektur ini adalah kemudahan dalam pengembangan dan debugging, karena setiap lapisan dapat dimodifikasi secara independen. Akan tetapi, kinerja bisa terganggu karena setiap permintaan sistem harus melewati beberapa lapisan secara berurutan, menimbulkan overhead yang tidak ada pada kernel monolitik. Model ini sering digunakan sebagai panduan desain internal, bukan sebagai jenis kernel yang berdiri sendiri. Beberapa contoh sistem operasi (OS) di dunia nyata yang mengimplementasikan arsitektur kernel ini adalah sebagai berikut: Monolithic Kernel secara dominan diwakili oleh Linux dan kernel UNIX tradisional, di mana seluruh layanan sistem beroperasi dalam satu ruang alamat yang cepat dan efisien. Di sisi lain, Microkernel yang menekankan stabilitas dan modularitas dicontohkan dengan baik oleh MINIX (sering digunakan untuk pendidikan dan sistem kecil) dan QNX (yang banyak digunakan dalam sistem real-time dan otomotif karena keandalannya yang tinggi). Sementara itu, Layered Architecture sebagai model struktural, tidak banyak digunakan untuk kernel secara utuh di OS modern karena masalah kinerja; contoh historis yang paling terkenal adalah Multics, dan saat ini, model ini lebih sering diadopsi secara internal untuk menata komponen dalam kernel yang lebih kompleks seperti dalam sistem UNIX yang membagi tugas menjadi lapisan-lapisan logis (meskipun kernel itu sendiri bersifat monolitik). Model arsitektur kernel yang paling relevan dan dominan dalam sistem modern bukanlah salah satu dari ketiganya secara murni, melainkan Hybrid Kernel, yang mengambil aspek terbaik dari Monolithic dan Microkernel. Arsitektur Hybrid ini, yang digunakan oleh sistem operasi populer seperti Microsoft Windows dan macOS (Darwin), menempatkan layanan penting seperti networking stack di dalam ruang kernel untuk kinerja tinggi (seperti Monolithic), tetapi menjaga modularitas dengan menjalankan driver dan layanan tertentu di luar kernel untuk stabilitas dan isolasi kesalahan (seperti Microkernel). Di sisi lain, Microkernel memiliki relevansi yang sangat tinggi di sektor yang menuntut keandalan ekstrem dan keamanan, seperti sistem embedded, otomotif (QNX), dan perangkat IoT, di mana kernel minimalis dan isolasi fault lebih penting daripada kecepatan mentah. Sementara itu, Monolithic Kernel (seperti Linux) masih sangat relevan di lingkungan yang mengutamakan kinerja throughput tertinggi, seperti server high-performance dan supercomputing. Adapun Layered Architecture telah kehilangan relevansinya sebagai arsitektur kernel mandiri karena overhead kinerja, tetapi prinsip modularitasnya tetap diadopsi secara internal oleh semua kernel modern. Secara keseluruhan, tren sistem modern mengarah pada solusi yang menawarkan keseimbangan antara kinerja dan modularitas, menjadikan Hybrid Kernel sebagai model yang paling serbaguna dan dominan.