软件架构4+1视图模型

1. 逻辑视图：主要支持系统的功能需求，描述系统应该提供的服务，系统被分解为一组关键抽象（通常是对象或类），使用面向对象的设计方法，展示类、类之间的关系（如关联、继承、组合等），通常使用类图和类模板来描述。

2. 进程视图：处理系统的并发性、同步性、分布性、完整性、容错性等非功能性需求，描述系统中的进程、任务及其通信机制，展示如何将逻辑视图中的对象映射到进程和任务中，使用进程图和任务图来描述。

3. 开发视图：关注软件在开发环境中的静态组织结构，支持软件的管理、复用、可移植性等，描述软件的模块和子系统的组织结构，通常采用分层架构，展示模块之间的依赖关系，使用模块图和子系统图来描述。

4. 物理视图：处理系统的非功能性需求，如可用性、可靠性、性能和可扩展性，描述软件如何映射到硬件上，展示进程、任务和对象在物理节点（如计算机、服务器）上的分布，使用物理架构图来描述。

5. 场景视图：通过一些关键场景（用例）来展示其他四个视图中的元素如何协同工作，验证架构的正确性，描述系统的关键用例或场景，展示对象之间的交互和进程之间的通信，使用对象交互图和场景图来描述。

总的来说，逻辑视图关注系统的功能结构和对象模型，进程视图关注系统的并发性和任务调度，开发视图关注软件的模块化组织和开发环境，物理视图关注软件在硬件上的部署和分布，场景视图通过关键场景验证和展示其他视图的协同工作。