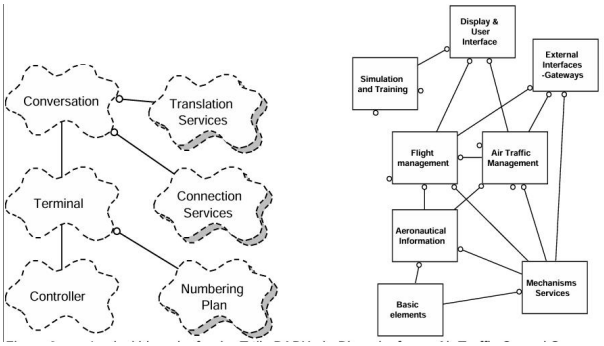
Homework：阅读《Software Architecture4+1》，试给出 SA 中 4+1 视图的描述。

答：

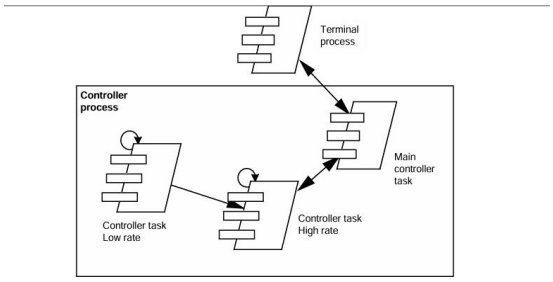
1. **逻辑视图**

当使用面向对象设计方法时，体现系统的对象模型。它将系统分解为一组关键的抽象，通常表现为对象或对象类，这些对象或对象类大多数情况下来自问题域。逻辑视图使用类图和类模板来表现，其中类图显示了一组类及其逻辑关系（如关联、使用、组合、继承等），而类模板则专注于每个类的主要操作和关键对象特性。



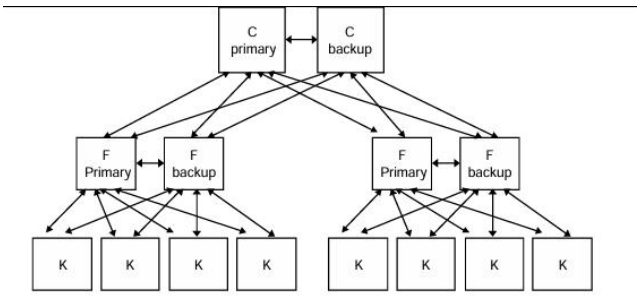
2. **过程视图**

捕获设计的并发和同步方面，关注一些非功能性需求，如性能和可用性。该视图通过独立执行的逻辑网络（“进程”）来表示，这些进程分布在通过局域网或广域网连接的一组硬件资源上。过程视图描述了系统中各种任务的组织和它们之间的通信方式，如同步和异步消息传递、远程过程调用等。此视图展示了系统如何利用并发来满足性能和可靠性要求，以及如何通过分布式计算资源来增强系统的容错能力。



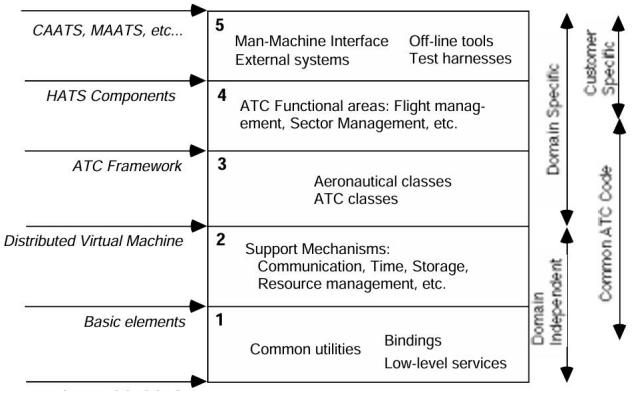
3. **物理视图**

物理视图考虑了系统的非功能性需求，如可用性、可靠性和性能。它描述了软件在计算机或处理节点网络上的执行情况，以及如何将各种元素（如网络、进程、任务和对象）映射到这些节点上。物理视图对于理解软件如何在实际的物理环境中部署和运行至关重要，它帮助系统工程师确定如何最有效地利用硬件资源来满足性能和可靠性的需求。



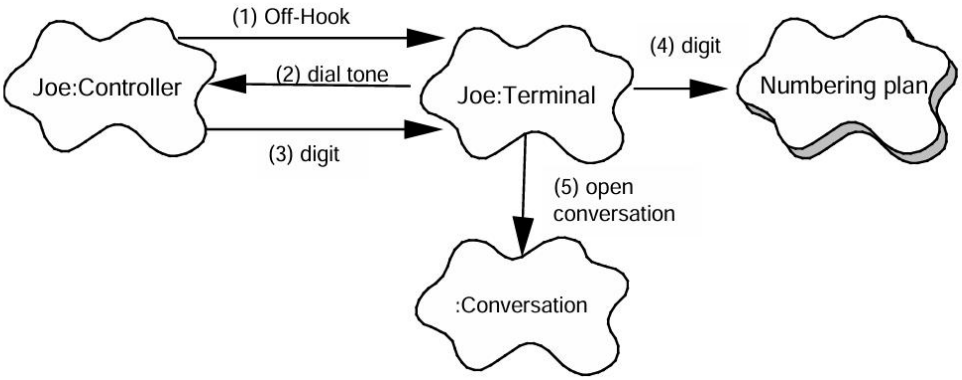
4. 开发视图

开发视图关注软件的实际模块组织和软件开发环境。它通过程序库或子系统的形式将软件拆解成小块，这些小块可以由一个或少数几个开发人员开发。子系统在层次结构中组织，每个层次提供了一个接口给它上面的层次。开发视图是为了解软件如何在开发过程中被组织和管理而设计的，涵盖了分配工作给团队、规划成本评估、项目进度监控、以及考虑软件复用的需求。



+1. 场景视图

场景视图通过一小组重要场景（或用例实例）展示其他四个视图中的元素如何协同工作。场景或用例是从系统最关键的功能性需求中提炼出来的，它们通常代表了系统的核心价值、最频繁使用的功能，或者需要缓解的技术风险。通过场景视图，可以展现和验证架构设计如何在实际操作中满足这些关键需求。



4+1 视图模型通过以上这些互补的视图提供了一个全面的软件架构描述方法，旨在确保软件架构的设计能够全面应对各种功能性和非功能性需求，同时促进软件项目中不同利益相关者之间的沟通和理解。