1. 什么是软件架构？

答：程序或计算系统的软件架构是该系统的结构，包括软件组件、那些组件的外部可见的属性，以及那些组件之间的关系。

1. 什么是SAD？

答：SAD即软件架构文档（Software Architecture Document）。

软件体系结构文档的内容应该提供整个软件体系结构文档的概述。

它应该包括软件体系结构文档的目的、范围、定义、缩略语、缩写、引用和概述。

三．为什么需要编写SAD？

答：

a.从软件架构实现本身来看：

软件架构可以递归地使用。考虑一个属于某个系统的软件架构组成部分的软件组件 (C1)。软件架构师将该组件及其应该公开的属性、功能和非功能特性及其与其他软件组件的关系交给系统设计人员。设计人员在分析软件组件 C1 之后，决定将该组件分解为更细粒度的组件（C11、C12 和 C13），其中每个组件提供可重用的功能，这些功能将用于实现 C1 的要求属性。设计人员详细设计了 C11、C12、C13 及其接口。

此时，对设计人员来说，C11、C12 和 C13 是体系结构构造（或组件）；其中每个构造具有显式定义的外部接口。对设计人员来说，C11、C12 和 C13 是软件组件 C1 的体系结构，并且这些构造需要进一步的改进和设计，以处理它们的内部实现。通过将大型、复杂的系统划分为小型的构成部分并集中于每个部分，可以递归地使用体系结构。

体系结构使用共同满足行为和质量目标的体系结构构建块将系统绑定在一起。参与者必须能够理解体系结构。因此必须为体系结构编写足够的文档说明，包括SAD。

b.从软件架构的参与者来看：

软件架构的参与者包括：体系结构的下游设计和实现用户。为体系结构的定义、维护和增强功能进行投资的人。

而向参与者传达正在构建的系统蓝图的关键是为系统体系结构编写文档说明。软件架构通过不同的视图进行表示——功能、操作、决策等等。没有任何单一视图能够表示整个体系结构。并非所有视图都需要表示特定企业或问题领域的系统体系结构。架构师将确定足以表示所需软件架构范畴的视图集。

综上，SAD提供了系统的全面架构概述，使用许多不同的架构视图来描述系统的不同方面。其目的是捕获和传递在系统上做出的重要架构决策，且同时确定文档的特定受众，并指出各类使用者将如何使用文档。

1. SAD的实用性？

答：

1. 可以向开发团队和业务及 IT 参与者传达有关该不断发展的系统的信息;
2. 有利于架构师弥补用白板描述解决方案（使用框线图方法）与以对下游设计和实现团队有意义的方式表示解决方案之间众所周知的差距；
3. 促进创建切合实际并且可以系统开发（例如遵循标准模板）的体系结构构件；
4. 帮助新成员快速熟悉工程内容，更快成为高效参与者。
5. SAD的基本构成？

答：

1. 介绍

[软件体系结构文档的介绍提供整个软件体系结构文档的概述。其包括软件体系结构文档的目的、范围、定义、缩略语、缩写、引用和概述。]

1.1目的

本文档提供系统的全面架构概述，使用许多不同的架构视图来描述系统的不同方面。其目的是捕获和传递在系统上做出的重要架构决策。同时应该确定文档的特定受众，并指出他们将如何使用文档。

1.2范围

a.软件体系结构文档应用于什么的简要描述;

b.受本文档影响的内容。

1.3定义、缩写和缩略语

提供正确解释软件体系结构文档所需的所有术语、缩写词和缩写词的定义。这些信息可以通过参考术语表项目来提供。

1.4参考资料

提供软件体系结构文档中其他地方引用的所有文档的完整列表。每个文件都应该通过标题、报告编号(如果适用)、日期和出版机构来标识。指定可以获得引用的来源。该信息可以通过参考附录或其他文件来提供。

1.5概述

描述软件体系结构文档的其余部分包含什么，并解释软件体系结构文档是如何组织的。

2. 架构表示

描述当前系统的软件架构是什么，以及它是如何表示的。在用例视图、逻辑视图、流程视图、部署视图和实现视图中，枚举必要的视图，并且对于每个视图，解释它包含的模型元素的类型。

3.架构目标和约束

描述对体系结构有重大影响的软件需求和目标，例如，安全性、安全性、隐私性、成品产品的使用、可移植性、分发和重用。捕获可能应用的特殊约束、设计和实现策略、开发工具、团队结构、进度、遗留代码，等等。

4. 用例视图

列出用例模型中的用例或场景，如果它们代表最终系统的一些重要的、中心的功能，或者如果它们有一个很大的体系结构覆盖范围——它们执行许多体系结构元素，或者如果它们强调或说明体系结构的一个特定的、微妙的点。

4.1用例实现

通过给出一些选定的用例(或场景)实现来说明软件实际上是如何工作的，同时解释各种设计模型元素是如何贡献它们的功能的。

5. 逻辑视图

描述设计模型在体系结构上的重要部分，例如分解为子系统和包。而对于每个重要的包，将其分解为类和类实用程序。引入体系结构上重要的类，并描述它们的职责，以及一些非常重要的关系、操作和属性。

5.1概述

根据包的层次结构和层描述了设计模型的总体分解。

5.2架构上重要的设计包

对于每个重要的包，包括一个带有它的名称的分段，它的简要描述，以及包含在包中的所有重要的类和包的图表。

对于包中的每个重要类，包括它的名称、简要描述，以及一些主要职责、操作和属性的描述(可选)。

6. 流程视图

描述系统分解为轻量级进程(单线程控制)和重量级进程(轻量级进程分组)。通过通信或交互的过程组来组织部分。描述进程之间通信的主要模式，如消息传递、中断和集合。

7. 部署视图

介绍一种或多种部署并运行软件的物理网络(硬件)配置。它是部署模型的视图。对于每个配置，它至少应该指出执行软件的物理节点(计算机、cpu)和它们的互连(总线、局域网、点对点等等)。还包括一个流程视图到物理节点的流程映射。

8. 实现视图

描述实现模型的整体结构，将软件分解为实现模型中的层和子系统，以及任何体系结构上重要的组件。

8.1概述

命名并定义各个层及其内容，管理包含到给定层的规则，以及层之间的边界。包括一个显示层之间关系的组件图。

8.2层

对于每一层，包括一个带有其名称的分段，位于该层中的子系统的枚举，以及一个组件图。

9. 数据视图(可选)

系统的持久化数据存储视图的描述。如果只有很少或没有持久数据，或者设计模型和数据模型之间的转换很简单，则此部分是可选的。

10. 尺寸和性能

对影响体系结构的软件的主要尺寸特征的描述，以及目标性能约束。

11. 质量

对软件体系结构如何对系统的所有功能(除功能外)作出贡献的描述:可扩展性、可靠性、可移植性，等等。如果这些特征具有特殊意义，例如安全、安全或隐私影响，则应明确描述。