

NEW PLAN

软考高级架构师

• — 段 新 征 程 •





大纲介绍





软件架构复用



软件产品线是指**一组软件密集型系统，它们共享一个公共的、可管理的特性集**，满足某个特定市场或任务的具体需要，是以规定的方式用**公共的核心资产**集成开发出来的。即围绕核心资产库进行管理、复用、集成新的系统。

- 举例：假设一家汽车导航系统制造公司想要开发多款汽车导航产品，以满足不同市场和客户的需求。他们可以采用软件产品线方法：
- 核心功能：公司首先开发一个通用的核心导航引擎，包括地图数据处理、路线规划、导航指令生成等功能。然后，他们可以基于这个核心引擎，创建不同变种的导航产品。例如：
 - 汽车导航器A：针对高端汽车市场，具有高级语音识别、实时交通信息和豪华界面。
 - 汽车导航器B：针对经济型车型，功能较少，但价格更实惠。
 - 卡车导航系统：针对卡车司机，包括特殊的货物规划和路线优化功能。
 - 定制特性：每个产品变种可以根据特定客户的需求进行定制，添加或移除特定功能或界面元素。



软件架构复用根据复用的时机包括**机会复用和系统复用**。机会复用是指开发过程中，只要发现有可复用的资产，就对其进行复用。系统复用是指在开发之前，就要进行规划，以决定哪些需要复用。



可复用的资产包括：需求、架构设计、元素、建模与分析、测试、项目规划、过程方法和工具、人员、样本系统、缺陷消除。



复用的基本过程主要包括**3个阶段**：首先构造/获取可复用的软件资产，其次管理这些资产(把它们放入到构件库)，最后针对特定的需求，从这些资产中选择可复用的部分，以开发满足需求的应用系统。



特定领域软件架构



DSSA就是专用于一类特定类型的任务(领域)的、在整个领域中能有效地使用的、为成功构造应用系统限定了标准的组合结构的软件构件的集合。它旨在满足该领域的独特需求和约束。这种架构通常通过针对特定问题领域的专业知识和最佳实践来优化软件系统的设计，以提供更高的性能、可维护性和可扩展性。

- 例如：在医疗保健领域，电子病历系统是一个常见的应用，用于管理患者的医疗记录。为了满足医疗保健行业的特殊需求，可以使用DSSA来设计和构建这样的系统。通过采用DSSA，这家医疗保健软件公司能够开发出适用于医疗保健领域的高度定制化且符合行业标准的电子病历系统。这就是特定领域的软件架构在医疗保健领域的一个示例。不同领域的DSSA可以根据其特殊需求进行定制化。



DSSA就是一个特定的问题领域中支持一组应用的领域模型、参考需求、参考架构等组成的开发基础，其目标就是支持在一个特定领域中多个应用的生成。



垂直域：在一个特定领域中的通用的软件架构，是一个完整的架构。例如电子病历系统、医院信息系统或医学影像分析系统



水平域：在多个不同的特定领域之间的相同的部分的小工具(如购物和教育都有收费系统，收费系统即是水平域)。例如网络安全通用架构等



特定领域软件架构



DSSA的三个基本活动：领域分析、领域设计和领域实现



领域分析：这个阶段的主要目标是获得领域模型(领域需求)。识别信息源(需求)，即整个领域工程过程中信息的来源，可能的信息源包括现存系统、技术文献、问题域和系统开发的专家、用户调查和市场分析、领域演化的历史记录等，在此基础上就可以分析领域中系统的需求，确定哪些需求是领域中的系统广泛共享的，从而建立领域模型。

- 比如在医疗系统中，领域分析阶段，团队收集了与医院管理相关的信息源，包括医疗领域的法规、患者需求、医院流程和现有系统。通过与医院管理员、医生和护士的讨论以及研究医疗保健法规，他们确定了系统的需求，如患者信息记录、医生排班、药物管理等。这些需求构成了领域模型，也就是医院信息管理领域的需求。



领域设计：这个阶段的目标是获得DSSA。DSSA描述在领域模型中表示的需求的解决方案，它不是单个系统的表示，而是能够适应领域中多个系统的需求的一个高层次的设计。建立了领域模型之后，就可以派生出满足这些被建模的领域需求DSSA。

- 举例：在领域设计阶段，基于领域模型，团队提供了医院信息管理系统的高层次设计。这个设计不是一个具体的应用程序，而是一个通用的架构。它包括模块化组件，如患者信息管理模块、医生排班模块、药物管理模块等。这些模块设计成可扩展和可重用的，以便满足不同医院的需求。这个领域设计能够适应医院信息管理领域中多个系统的需求。



特定领域软件架构



领域实现：这个阶段的主要目标是依据领域模型和**DSSA**开发和组织可重用信息。这些可重用信息可能是从现有系统中提取得到，也可能需要通过新的开发得到。

- 举例：在领域实现阶段，团队根据领域模型和领域设计来开发具体的医院信息管理系统。他们实现了患者信息管理模块，包括患者信息录入、查看和编辑功能。同时，他们也开发了医生排班模块，以及药物管理模块，确保这些模块符合领域设计的要求。这些模块的开发是基于领域模型和DSSA的指导原则，以确保系统的可维护性和可重用性。



领域分析用于确定需求，**领域设计**用于提供通用架构，而**领域实现**用于将该架构转化为具体的应用程序模块。这个过程有助于确保系统能够满足特定领域的需求，并具备可维护和可重用的特性。

特定领域软件架构



参与DSSA的四种角色人员：领域专家、领域分析人员、领域设计人员和领域实现人员



领域专家：包括该领域中系统的有经验的用户、从事该领域中系统的需求分析、设计、实现以及项目管理的有经验的软件工程师等。提供关于领域中系统的需求规约和实现的知识，帮助组织规范的、一致的领域字典，帮助选择样本系统作为领域工程的依据，复审领域模型、DSSA等领域工程产品，等等。



领域分析人员：由具有知识工程背景的经验丰富的系统分析员来担任。控制整个领域分析过程，进行知识获取，将获取的知识组织到领域模型中。



领域设计人员：由有经验的软件设计人员来担任。根据领域模型和现有系统开发出DSSA，并对DSSA的准确性和一致性进行验证。



领域实现人员：由有经验的程序设计人员来担任。根据领域模型和DSSA，开发构件。



特定领域软件架构



建立DSSA的过程(实现的角度):

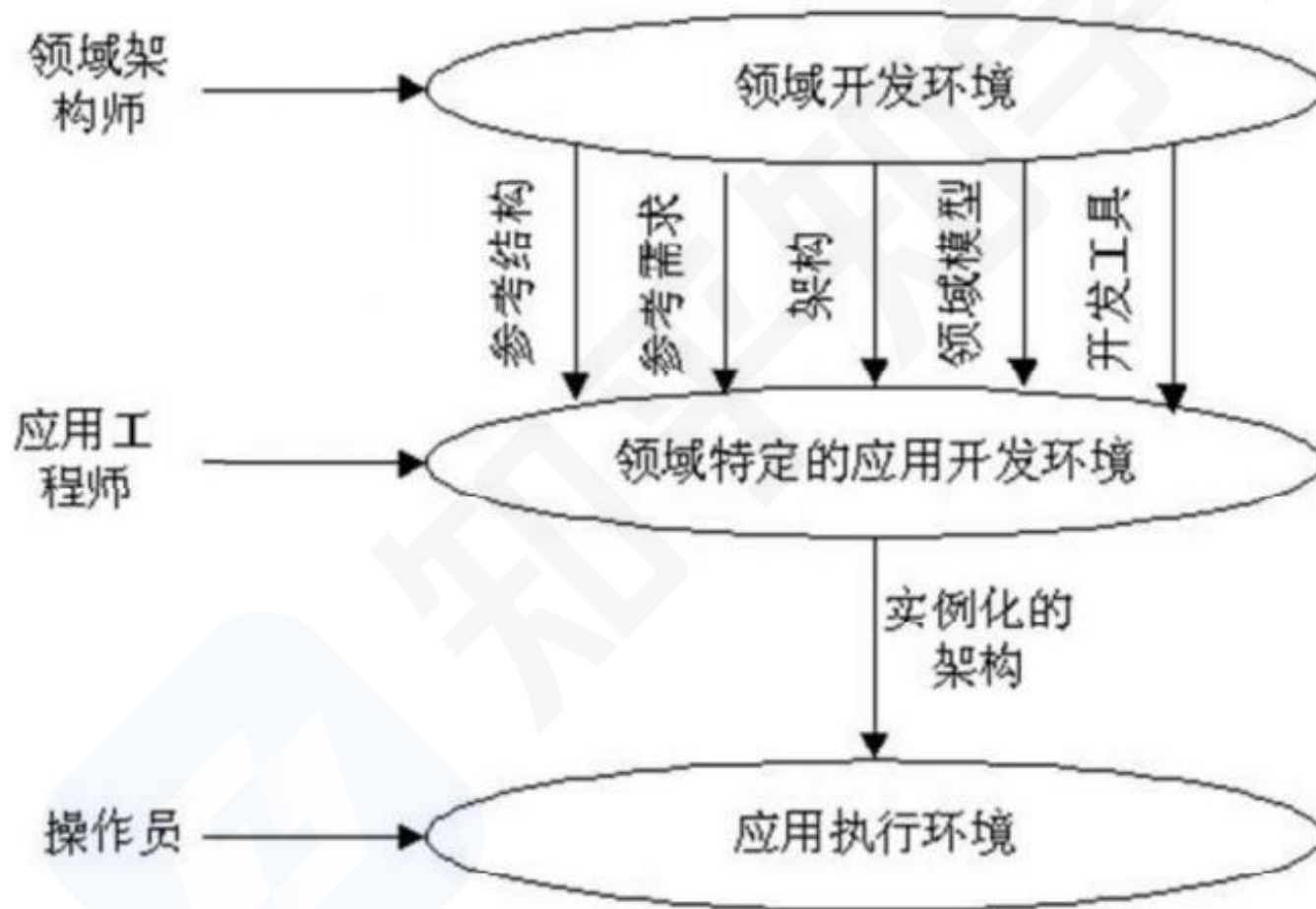
- **定义领域范围**: 领域中的应用要满足用户一系列的需求。
- **定义领域特定的元素**: 建立领域的字典, 归纳领域中的术语, 识别出领域中相同和不相同的元素。
- **定义领域特定的设计和实现需求的约束**: 识别领域中的所有约束, 这些约束对领域的设计和实现会造成什么后果。
- **定义领域模型和架构**: 产生一般的架构, 并描述其构件说明。
- **产生、搜集可复用的产品单元**: 为DSSA增加复用构件, 使可用于新的系统。

以上过程是**并发的、递归的、反复的、螺旋型**的。



特定领域软件架构

三层次模型：领域开发环境、领域特定的应用开发环境、应用执行环境





特定领域软件架构



领域开发环境：在领域开发环境中，领域架构师负责制定医院信息管理系统核心架构，以及定义了系统的参考结构、参考需求、架构、领域模型和开发工具。这个环境的任务是为特定领域(医院信息管理)创建通用的框架，以满足多个医院信息系统的需求。

- 例如，领域架构师可能决定使用分布式数据库系统以确保数据的可扩展性和安全性，还可能定义患者信息管理、医生排班和药物管理等领域需求。



领域特定的应用开发环境：在领域特定的应用开发环境中，应用工程师根据具体的医院信息管理系统需求，将核心架构实例化为一个具体的应用程序。这个环境会根据领域开发环境提供的参考结构和模型，来创建医院A、医院B等具体医院信息管理系统的实例。

- 例如，对于医院A，应用工程师会根据核心架构和领域模型来定制患者信息管理模块，以适应医院A的需求。同样，对于医院B，应用工程师也会根据同一核心架构，但可能进行不同的定制以满足医院B的特定需求。



应用执行环境：应用执行环境是医院信息管理系统最终运行的地方，由操作员负责实际使用和维护。在这个环境中，已经实例化的医院信息管理系统被部署和运行。

- 例如，医院A的操作员使用医院A的实例化系统来记录和管理患者信息，医院B的操作员使用医院B的实例化系统来执行相同的任务。这个层次关注的是系统的实际运行和操作。



DSSA是在一个特定应用领域中为一组应用提供组织结构参考的软件体系结构，参与DSSA的人员可以划分为4种角色，包括领域专家、领域设计人员、领域实现人员和()，其基本活动包括领域分析、领域设计和()。

- | | | | |
|-----------|--------|---------|--------|
| A. 领域测试人员 | B.领域顾问 | C.领域分析师 | D.领域经理 |
| A. 领域建模 | B.架构设计 | C.领域实现 | D.领域评估 |



特定领域软件架构(Domain Specific Software Architecture, DSSA)以一个特定问题领域为对象，形成由领域参考模型，参考需求，()等组成的开发基础架构，支持一个特定领域中多个应用的生成。DSSA的基本活动包括领域分析、领域设计和领域实现。其中领域分析的主要目的是获得()，从而描述领域中系统之间共同的需求，即领域需求;领域设计的主要目标是获得()，从而描述领域模型中表示需求的解决方案;领域实现的主要目标是开发和组织可重用信息，并实现基础软件架构。

- | | | | |
|-------------|------------|--------------|--------------|
| A. 参考设计 | B.参考规约 | C.参考架构 | D.参考实现 |
| A. 领域边界 | B.领域信息 | C.领域对象 | D.领域模型 |
| A. 特点领域软件需求 | B.特定领域软件架构 | C.特定领域软件设计模型 | D.特定领域软件重用模型 |



基于架构的软件开发



基于架构的软件开发(Architecturally Based Software Development, ABSD)是一种软件开发方法，**强调在开发过程中首先定义系统的体系结构，然后根据这个体系结构来实现系统。它有助于确保系统的结构和设计与业务需求保持一致**



假设一家电子商务公司决定开发一个全新的在线购物网站，他们采用基于架构的软件开发方法：

- 定义系统架构：首先，开发团队会定义系统的体系结构，包括用户界面层、业务逻辑层和数据存储层。这些层次将被明确定义，以确保开发过程中每个组件的职责清晰明确。
- ABSD的关键决策：在系统架构阶段，团队会做出一些关键的架构决策，例如选择使用哪种技术堆栈、如何处理用户身份验证、如何处理库存管理、如何处理支付等。这些决策是ABSD的核心，它们会在整个开发过程中起到指导作用。
- 系统实施：一旦系统架构被定义和核心决策被制定，开发团队会开始实施系统。他们会根据架构中的不同层次来编写代码，确保各个组件按照系统设计进行开发。
- 持续维护和演化：随着时间的推移，业务需求可能会发生变化，但由于系统的基本架构已经定义，团队可以相对容易地进行扩展和修改，而不必重新设计整个系统。



基于架构的软件开发方法强调了在软件开发过程中先关注系统的结构和核心决策，以确保最终的系统能够满足业务需求并具有良好的扩展性和维护性。这个方法有助于降低项目失败的风险，因为它强调了在开发之前做出关键决策的重要性。



基于架构的软件开发



ABSD方法是架构驱动，强调由业务、质量和功能需求的组合驱动架构设计。它强调采用视角和视图来描述软件架构，采用用例和质量属性场景来描述需求。进一步来说，用例描述的是功能需求，质量属性场景描述的是质量需求(或侧重于非功能需求)。



使用ABSD方法，设计活动可以从项目总体功能框架明确就开始，这意味着需求获取和分析还没有完成，就开始了软件设计。



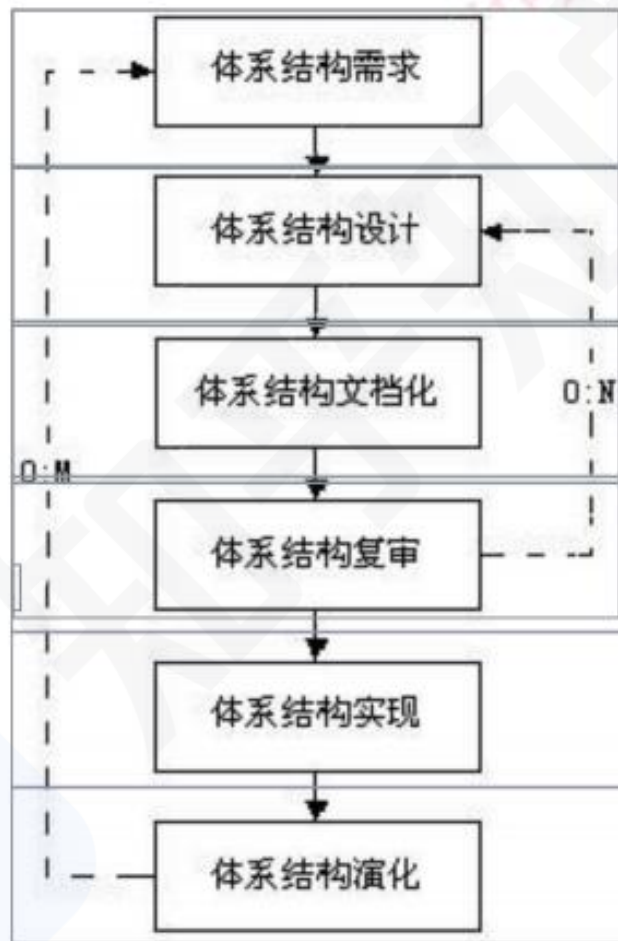
ABSD方法有三个基础。第一个基础是功能的分解，使用已有的基于模块的内聚和耦合技术；第二个基础是通过选择架构风格来实现质量和业务需求；第三个基础是软件模板的使用，软件模板利用了一些软件系统的结构进行复用。



ABSD方法是递归的，且迭代的每一个步骤都是清晰定义的。因此，不管设计是否完成，架构总是清晰的，有助于降低架构设计的随意性。

基于架构的软件开发

基于架构的软件开发过程可分为下列六步：





基于架构的软件开发



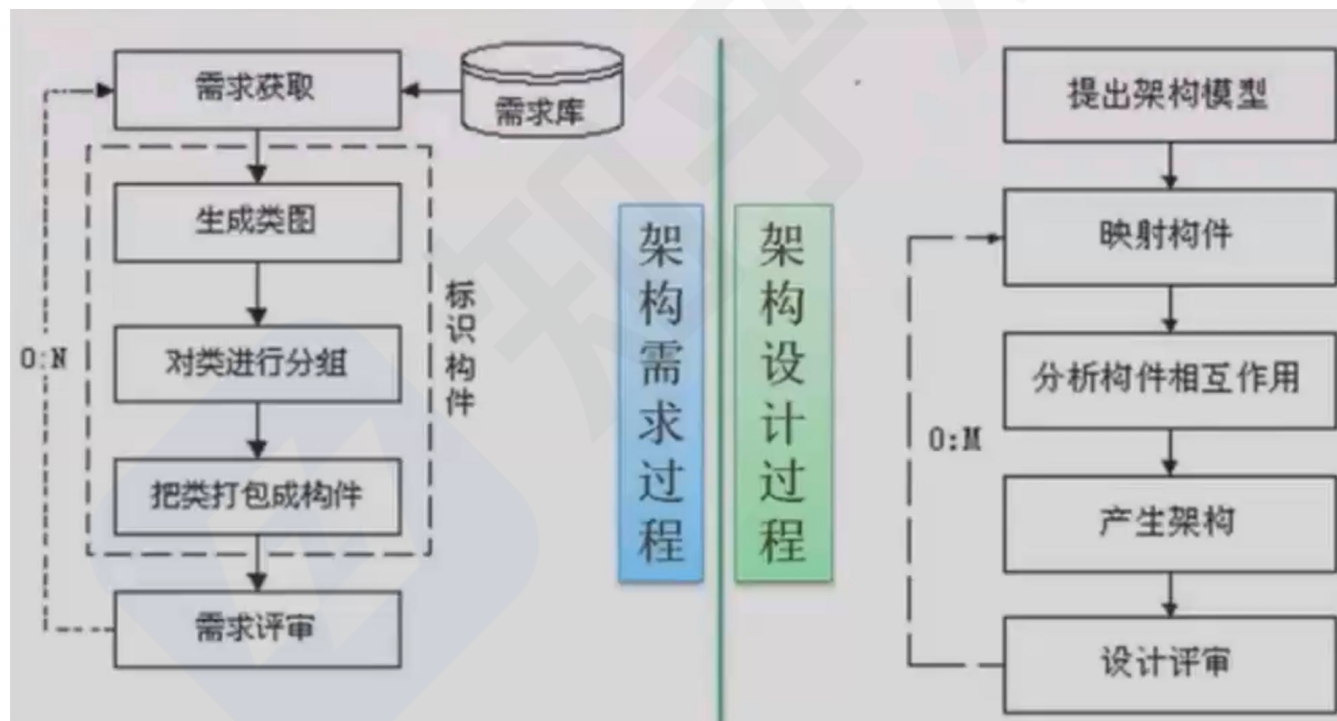
1. 架构需求：重在掌握标识构件的三步，如下左图。



2. 架构设计：将需求阶段的标识构件映射成构件，进行分析，如下右图。



3. 架构(体系结构)文档化：主要产出两种文档，即架构(体系结构)规格说明，测试架构(体系结构)需求的质量设计说明书。文档是至关重要的，是所有人员通信的手段，关系开发的成败。



基于架构的软件开发



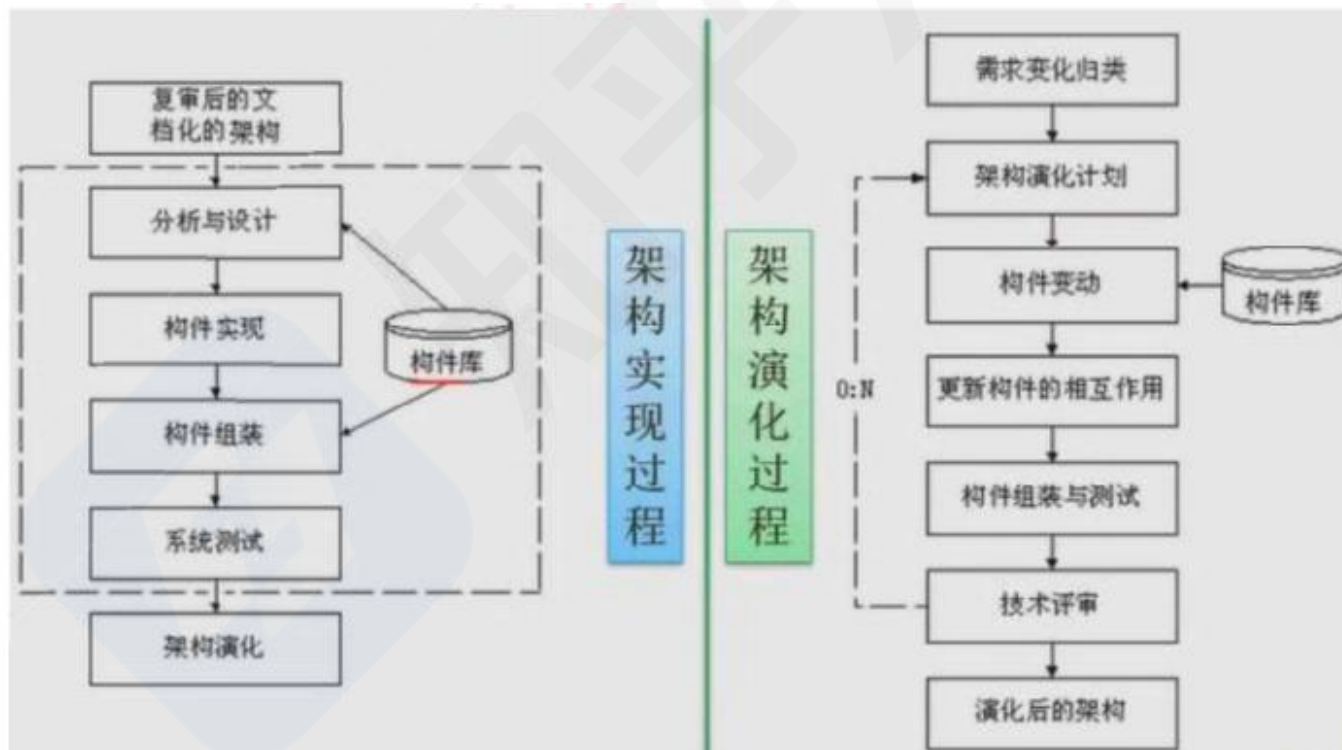
4. 架构复审：由外部人员(独立于开发组织之外的人，如用户代表和领域专家等)参加的复审，复审架构是否满足需求，质量问题，构件划分合理性等。若复审不过，则返回架构设计阶段进行重新设计、文档化，再复审。



5. 架构实现：用实体来显示出架构。实现构件，构件组装成系统，如下左图：



6. 架构演化：对架构进行改变，按需求增删构件，使架构可复用，如下右图：





在基于体系结构的软件设计方法中，采用()来描述软件架构，采用()来描述功能需求，采用()来描述质量需求。

- | | | | |
|----------|---------|---------|---------|
| A.类图和序列图 | B.视角与视图 | C.构件和类图 | D.构件与功能 |
| B.类图 | B.视角 | C.用例 | D.质量场景 |
| C.连接件 | B.用例 | C.质量场景 | D.质量属性 |



体系结构文档化有助于辅助系统分析人员和程序员去实现体系结构。体系结构文档化过程的主要输出包括()。

- A.结构规格说明、测试体系结构需求的质量设计说明书
- B.属性说明书、体系结构描述
- C.结构规格说明、软件功能需求说明
- D.图体系结构模型、体系结构验证说明

T H E E N D

功不唐捐，玉汝于成！

• 开 启 新 征 程 •

