风洞设备PHM模型规范化管理研究

2.存储使用运维规范

随着现代软件行业的高速发展，软件的复杂性越来越高，对开发者的综合素质要求也越来越高，不仅是编程知识点，其它维度的知识点也会影响到软件的最终交付质量。为了保障开发出软件的综合质量，需要对开发流程、编程规范、异常处理、软件测试、安全规约、数据存储、工程结构、设计规约等方面做出相应的规范进行约束。

2.1开发规范

在应做出的各个方面规范中，最重要的就是要遵循规范的软件工程开发方法。从60年代末以来，出现了许多软件工程方法，其中最具影响的是面向对象方法、结构化方法和形式化方法，前两种方法是使用最广的两种方法，本部分主要介绍面向对象和结构化两种软件工程方法的特点及过程，对两种方法做出对比，并对模块化的概念做出介绍。

2.1.1面向对象方法

面向对象方法将面向对象的思想应用于软件开发过程中，指导开发活动，是建立在“对象”概念基础上的方法学，简称OO(Object-Oriented)方法。面向对象方法起源于面向对象的编程语言，对象指的就是面向对象编程语言中类（class）的实例，对象由属性和操作组成，可按其属性进行分类，对象之间的联系通过传递消息来实现，对象具有封装性、继承性和多态性。它将对象作为程序的基本单元，将程序和数据封装其中，以提高软件的重用性、灵活性和扩展性，对象里的程序可以访问及经常修改与对象相关联的数据。在面向对象程序设计里，计算机程序会被设计成彼此相关的对象。面向对象方法主要分为面向对象的需求分析和面向对象的系统设计，接下来将对它们做出解释。

（1）面向对象的需求分析

面向对象的需求分析的核心是利用面向对象的概念和方法为软件系统的需求搭建分析模型。面向对象方法使用统一建模语言（UML）表示系统的数据模型、功能模型和行为模型，这些模型共同组成了系统的分析模型，它们之间的关系如下表x所示。

表x 面向对象的分析模型构成及表示

|  |  |
| --- | --- |
| 分析模型 | 表示方式 |
| 数据模型 | 类图 |
| 功能模型 | 用例图 |
| 行为模型 | 顺序图、协作图、活动图、状态图 |

其中类图用来表示数据模型。类图中用矩形框来表示系统中的具体类，矩形框分为三层：第一层是类的名字。第二层是类的成员变量即类的属性；第三层是类的方法。成员变量以及方法前的访问修饰符用下列符号来表示：

“+”表示 public；

“-”表示 private；

“#”表示 protected；

如果不带符号则表示 default。

图x所示即为一个简单的学生类的类图表示。



图x 简单学生类的类图表示

类和类、类和接口、接口和接口之间存在一定关系，UML类图中一般会有连线指明它们之间的关系。关系共有六种类型，分别是实现关系、泛化关系、关联关系、依赖关系、聚合关系、组合关系。

用例图用来表示功能模型，即通过用例图表现出软件系统所需要提供的功能。用例图能够描述用例、参与者以及它们之间的关系，从用户的角度描述用户对软件系统的需求。其中参与者是指在系统之外，但与系统直接交互的对象，即actor，也叫执行者、活动者。参与者用人形符号表示，在人形符号下面标出参与者的角色名。参与者不止是人员，也有可能是信息系统、设备。用例是用户期望系统具备的功能，每一个用例说明一个系统提供给它的使用者的一种服务或功能。用例的目标是要定义系统的一个行为，但并不显示系统的内部结构。用例名一般为动宾短语，符号是椭圆加用例名。一个简单的用例图示例如下图x所示。



图x 一个简单的用例图示例

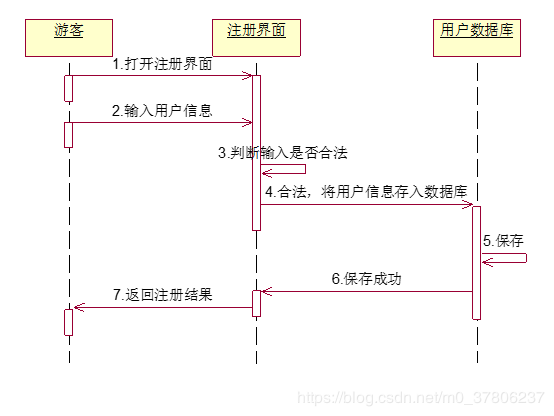
在面向对象的需求分析阶段，主要是构建出系统中各个参与者的用例图，根据用例图即可得到系统的功能性需求。在后续的系统设计过程中，根据得到的需求进行设计。

（2）面向对象的系统设计

面向对象的系统设计主要是对分析模型中的内容进行整理，其中内容包括架构设计、用例设计、子系统设计、类设计等。架构设计的侧重点在于系统的体系框架的合理性，保证系统架构在系统的各个非功能性需求中保持一种平衡；子系统设计一般是采用纵向切割，关注的是系统的功能划分；类设计是设计各个类的属性、方法、关联关系等。面向对象设计是为了解决软件问题而设计一个交互对象系统的过程，是一种软件设计的方法。

在面向对象的系统设计过程中，要根据软件系统的需求选用合适的体系结构即架构，并根据用例图中的用例构建出顺序图、协作图等。

顺序图（Sequence Diagram）又叫做时序图，是一种能够显示对象之间交互的图。在顺序图中对象之间的交互按照时间顺序排列，能够方便地看出对象之间信息传递过程。顺序图中有角色（Actor）、对象（Object）、生命线（LifeLine）、控制焦点（Activation）和消息（Message）。其中角色可以是人或其他系统、子系统，类似于用例图中的参与者。对象位于顺序图的顶部，以一个矩形表示。生命线是顺序图中从角色和每一个对象处垂直画下的虚线，表示角色和对象的存活情况。控制焦点表示顺序图中在对象的生命线上某段时期内执行的操作，以一个很窄的矩形表示。消息表示对象之间相互传递的信息，有简单消息、同步消息、异步消息和返回消息四种。一个简单的顺序图示例如下图x所示。

.

图x 一个简单的顺序图示例

协作图与顺序图有相同的作用，同样表示系统中对象之间的交互。区别在于顺序图能够清晰地表达出交互中的先后顺序，而协作图不具备此优点。根据得到的顺序图可以对交互进一步细化，详细设计出交互过程，进行后续的程序实现。

2.1.2结构化方法

结构化方法又叫做面向过程方法、生命周期法，与面向对象方法相似，都属于软件工程方法。结构化方法是一种传统的软件工程方法，用系统工程的思想和工程化的方法，按用户至上的原则，结构化、模块化、自顶向下地对系统进行分析和设计，使得每个阶段处理的问题都控制在人们容易理解和处理的范围内，是迄今为止最传统、应用最广泛的一种信息系统开发方法。其中，自顶向下指的是将复杂的大问题分解为相对简单的小问题，找出每个问题的关键、重点所在，然后用精确的思维定性、定量地去描述问题；逐步求精指的是将现实问题经过几次抽象（细化）处理，最后只需解决一些简单的算法描述和算法实现问题。

结构化方法主要包括结构化分析和结构化设计两部分。

（1）结构化分析

结构化分析（Structured Analysis，简称SA）类似于面向对象方法中的面向对象的需求分析，利用图形表达用户需求，强调开发方法的结构合理性以及所开发软件的结构合理性。结构化分析方法使用数据字典和E-R图表示数据模型，使用数据流图（DFD）表示功能模型，使用状态转换图表示行为模型。

结构化分析的步骤如下：

①分析当前的情况，做出反映当前物理模型的DFD；

②推导出等价的逻辑模型的DFD；

③设计新的逻辑系统，生成数据字典和基元描述；

④建立人机接口，提出可供选择的目标系统物理模型的DFD；

⑤确定各种方案的成本和风险等级，据此对各种方案进行分析；

⑥选择一种方案；

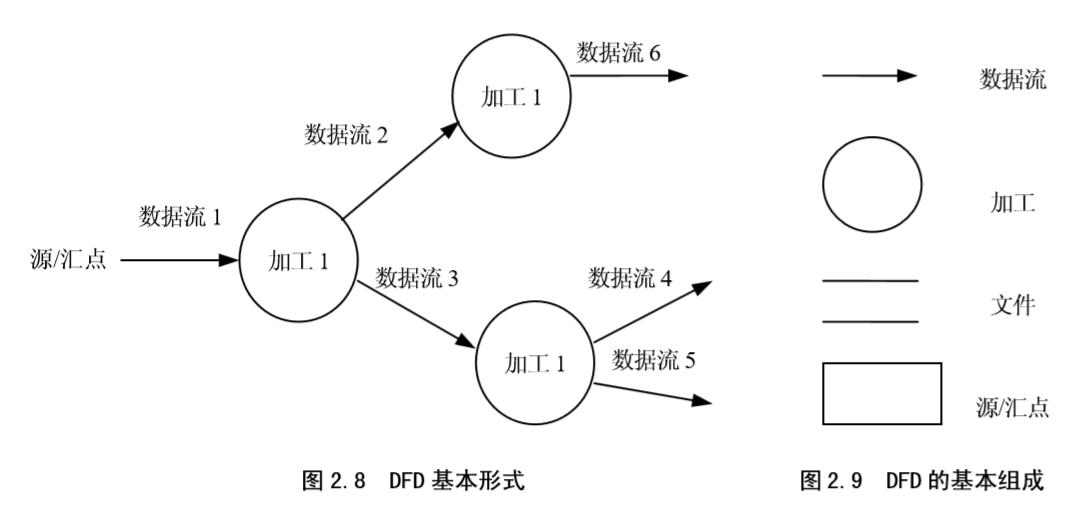
⑦建立完整的需求规约。

其分析模型如表x所示。

表x 面向对象的分析模型构成及表示

|  |  |
| --- | --- |
| 分析模型 | 表示方式 |
| 数据模型 | 数据字典、E-R图 |
| 功能模型 | 数据流图 |
| 行为模型 | 状态转换图 |

数据流图用于表示业务信息系统中的数据流，能够形象地表示出系统内部数据的流动传输情况。数据流图中有外部实体、处理过程（加工）、数据流、数据存储四个组成部分。外部实体指系统以外又和系统有联系的人或事物，它说明了系统中数据的来源和去处，通常外部实体在数据流图中用矩形框表示。处理过程指对数据的处理，也就是数据变换，在数据流图中可以用圆形或带圆角的长方形表示。数据流指处理功能的输入和输出，用来表示中间数据流值，但不能用来改变数据值，通常用一个水平箭头或垂直箭头表示。数据存储表示数据保存的地方，用来存储数据，在数据流图中可以使用右侧开口的矩形或上下平行的直线表示。一个数据流图基本形式示例如图x所示。



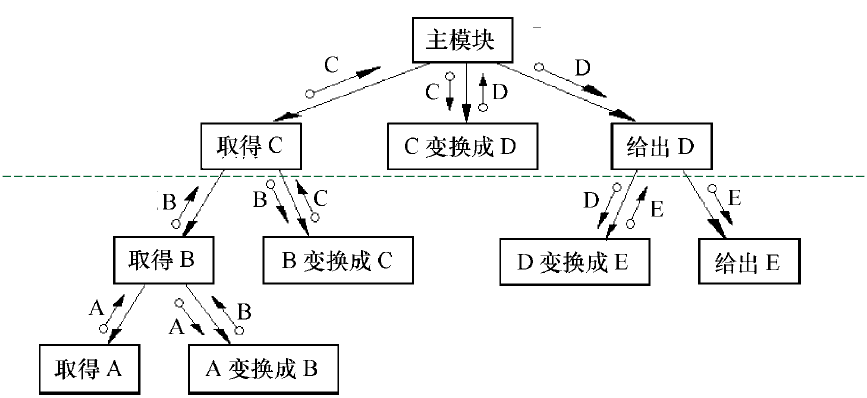
图x 数据流图基本形式

在构建系统的数据流图时，采取自顶向下、逐层分解的方式，首先画出系统的顶层数据流图，再慢慢画出顶层数据流图中的一个个小的部分，这样就可以构建出系统的整体数据流图，在后续的设计过程中以此为基础进行设计。

（2）结构化设计

结构化设计把系统作为一系列数据流的转换，输入数据被转换为期望的输出值，通过模块化来完成自顶而下实现的文档化，并作为一种评价标准在软件设计中起指导性作用，通常与结构化分析方法衔接起来使用，以数据流图为基础得到软件的系统结构图。

系统结构图又称为软件结构图，是一种能够反映软件系统中组件之间相互关系和约束的体系结构设计图，主要分为变换型和事务型两种。变换型系统结构图中信息一般是以外部形式进入系统，经过系统处理后离开系统，事务型系统结构图中将输入流分离成许多发散的数据流，根据输入的值选择其中一条路径执行。一个变换型系统结构图如下图x所示。



图x 变换型系统结构图示例

为了开发出高质量的软件，在软件开发过程中必须遵守抽象、信息隐藏、模块化、一致性等原则。

抽象指抽取事物最基本的特性和行为，忽略非基本的细节。采用分层次抽象的办法可以控制软件开发过程的复杂性，有利于软件的可理解性和开发过程的管理。

信息隐藏是指采用封装技术，将程序模块的实现细节（过程或数据）隐藏起来，对于不需要这些信息的其它模块来说是不能访问的，使模块接口尽量简单。

按照信息隐藏的原则，系统中的模块应设计成“黑箱”，模块外部只能使用模块接口说明中给出的信息，如操作、数据类型等等。

一致性是指整个软件系统（包括文档和程序）的各个模块均应使用一致的概念、符号和术语；程序内部接口应保持一致；软件与硬件接口应保持一致；系统规格说明与系统行为应保持一致；实现一致性需要良好的软件设计工具（如数据字典、数据库、文档自动生成与一致性检查工具等等）、设计方法和编码风格的支持。

面向对象方法个结构化方法二者的对比见下表x：

表x 面向对象方法和结构化方法对比

|  |  |  |  |
| --- | --- | --- | --- |
| 软件工程方法 | 特点 | 优点 | 缺点 |
| 面向对象方法 | 将面向对象的思想应用于软件开发过程中，将对象作为程序的基本单元 | 程序易理解，可维护性、灵活性和扩展性高，代码可重用性好，数据和操作封装在一起 | 类复杂性高，会出现设计人员之间的认识差异，编程便捷性较差 |
| 结构化方法 | 用户至上，结构化、模块化、自顶向下、逐步细化 | 程序结构清晰、开发有序，目标明确，代码可读性高 | 不太适合需求模糊或随着时间变化的系统，数据与程序分离，调试难度增较大 |

2.1.3模块化

模块化思想是一种软件工程中的编程思想，强调把模块化编程方法应用在编程开发的过程中，下面对模块化编程进行介绍。

模块化强调将计算机程序的功能分离成独立的、可相互改变的“模块”（module），它使得每个模块都包含着执行预期功能的一个唯一方面（aspect）所必需的所有东西。

模块化用来分割，组织和打包软件。每个模块完成一个特定的子功能，所有的模块按某种方法组装起来，成为一个整体，完成整个系统所要求的功能。

模块具有以下几种基本属性：接口、功能、逻辑、状态，功能、状态与接口反映模块的外部特性，逻辑反映它的内部特性。

在系统的结构中，模块是可组合、分解和更换的单元。模块化是一种把复杂系统分解成为更好的可管理模块的方式。它可以通过在不同组件设定不同的功能，把一个问题分解成多个小的独立、互相作用的组件，来处理复杂、大型的软件。

模块化能够让系统更加接近高内聚低耦合的状态，也使得软件系统方便维护，其所具备的模块可重用、功能独立等特性使得模块化成为当前软件工程开发的主要特性。

在模块化编程时，应该将软件结构划分为若干个模块分别进行开发。在对每个模块进行开发完成后，应该进行单元测试检测模块的功能和性能。在所有模块都开发完成之后，要对所有模块进行集成并测试，以验证整个系统的功能和性能。

2.2存储规范

《基于文档型非关系型数据库的档案数据存储规范》中对不同类型、不同格式的档案数据存储制定了规范。要求模型库中字段类型应包括字符串、数值、日期、时间、文本、二进制等，并对各种类型做出解释。要求模型库中的存储对象包括内容数据等非结构化数据、XML文件等半结构化数据以及元数据等。对于数据的存储方式，指出了以下四种方式：

①将内容数据和元数据全部装入模型库,按照与元数据的匹配关联关系,内容数据存储在文档型模型库的二进制字段中;

②将元数据装入模型库,将内容数据映射到模型库;

③同一模型库可存储多种格式的内容数据,不同记录(行)的内容数据的格式可不同;

④同一条记录可存储一个或多个内容数据,同一条记录(行)中多个内容数据的格式可不同。

除此之外，《基于文档型非关系型数据库的档案数据存储规范》中还对导入数据的方法、数据的质量控制、维护等进行规范。

在计算机系统中，由于计算机只能够处理数字，如果要处理文本就必须先把文本转换为数字才能进行处理，因此要对文本字符串进行编码。在字符的编码和校验过程中，为了能够支持中文，应该使用utf-8编码，使用utf-8-cgi校验。除了对数据进行编码之外，还需要为数据库本身以及其内部的表和字段进行合适的命名，并且要对数据进行备份处理，命名规范和数据备份将在下面进行说明。

2.2.1命名规范

数据库命名要采用26个英文字母（区分大小写）和0-9的自然数加上下划线’\_’组成，命名简洁明确，多个单词用下划线’\_’分隔，一个项目一个数据库，多个项目慎用同一个数据库。

在对数据库表命名时，遵循以下规范：

①采用26个英文字母（区分大小写）和0-9的自然数（经常不需要）加上下划线’\_’组成，命名简洁明确，多个单词用下划线’\_’分隔。

②全部小写命名，禁止出现大写。

③禁止使用数据库关键字，如：name,time, datetime,password等。

④表名称不应该取得太长（一般不超过三个英文单词）。

⑤表的名称一般使用名词或者动宾短语。

⑥用单数形式表示名称，例如，使用employee，而不是employees。

⑦表必须填写描述信息（使用SQL语句建表时）。

对数据库中字段进行命名时，应该遵循以下规范：

①采用26个英文字母（区分大小写）和0-9个自然数（经常不需要）加下划线’\_’组成，命名简洁明确，多个单词用下划线’\_’分隔。

②全部小写命名，禁止出现大写。

③字段必须填写描述信息。

④禁止使用数据库关键字，如：name、time、datetime、password等。

⑤字段名称一般采用名词或动宾短语。

⑥采用字段的名称必须是易于理解，一般不超过三个英文单词。

⑦在命名表的列时，不要重复表的名称。

⑧不要在列的名称中包含数据类型。

⑨字段命名使用完整命名，禁止缩写。

2.2.2数据备份与恢复

在使用数据库存储数据文件时，可能会出现以下场景导致数据丢失或损坏：

①人为操作失误造成某些数据被误操作；

②软件 BUG 造成部分数据或全部数据丢失；

③硬件故障造成数据库部分数据或全部数据丢失；

④全漏洞被入侵数据恶意破坏。

还可能会由于特殊应用场景下基于时间点的数据恢复、开发测试环境数据库搭建、相同数据库的新环境搭建、数据库或者数据迁移等情况需要对数据进行迁移处理，因此应该对数据库本身及其数据进行备份，以便各种事件发生时能够迅速恢复数据。数据库备份的内容应该包括数据库数据、数据库结构和数据库定义文件。应按照数据库结构(字段)备份数据库数据。数据的备份方式可以有整体备份、拆分备份、增量与差异备份、在线备份和离线备份以及自动备份几种方式。

整体备份是指当文档型数据库的容量小于备份介质的容量时，无需对文档型数据库做任何处理，直接对数据库整体进行复制备份。拆分备份是指当文档型数据库的容量大于备份介质的容量时，应将数据库拆分成容量小于备份介质容量的若干个子数据库，然后将各子数据库分别备份到备份介质上，保留原有的访问控制策略，并保证原数据库的完整性。增量与差异备份是指对文档型数据库中新增的档案数据进行增量备份，对被修改的档案数据进行差异备份。在线备份指将数据库数据、数据库结构和数据库定义文件备份到在线存储介质上。离线备份指是将数据库数据、数据库结构和数据库定义文件备份到离线存储介质上。自动备份通过软件的控制方式将数据库数据、数据库结构和数据库定义文件进行备份。

在风洞一体化平台数据的备份时可以采取增量与差异备份，这种方法不会产生重复的备份数据，备份数据需要时间较短且相较于其它方法更能节省磁盘空间。

在对数据进行还原时，又有整体还原和合并还原几种方式。整体还原将整体备份的数据库数据还原到原数据库系统。合并还原将拆分备份的多个子数据库数据还原到原数据库系统，采用的方法包括：

（1）在新建数据库系统合并全部子数据库数据，然后将合并形成的数据库数据还原到原数据库系统；

（2）在原数据库系统中合并还原全部子数据库数据。

在意外发生时的恢复策略也有正常恢复和异常恢复两种。正常恢复指使用备份的数据库数据直接覆盖原数据库数据。异常恢复指在异常情况下，用备份的数据库数据、数据库结构、数据库定义文件和日志文件进行恢复，并进行数据完整性校验，以确保数据的完整性。