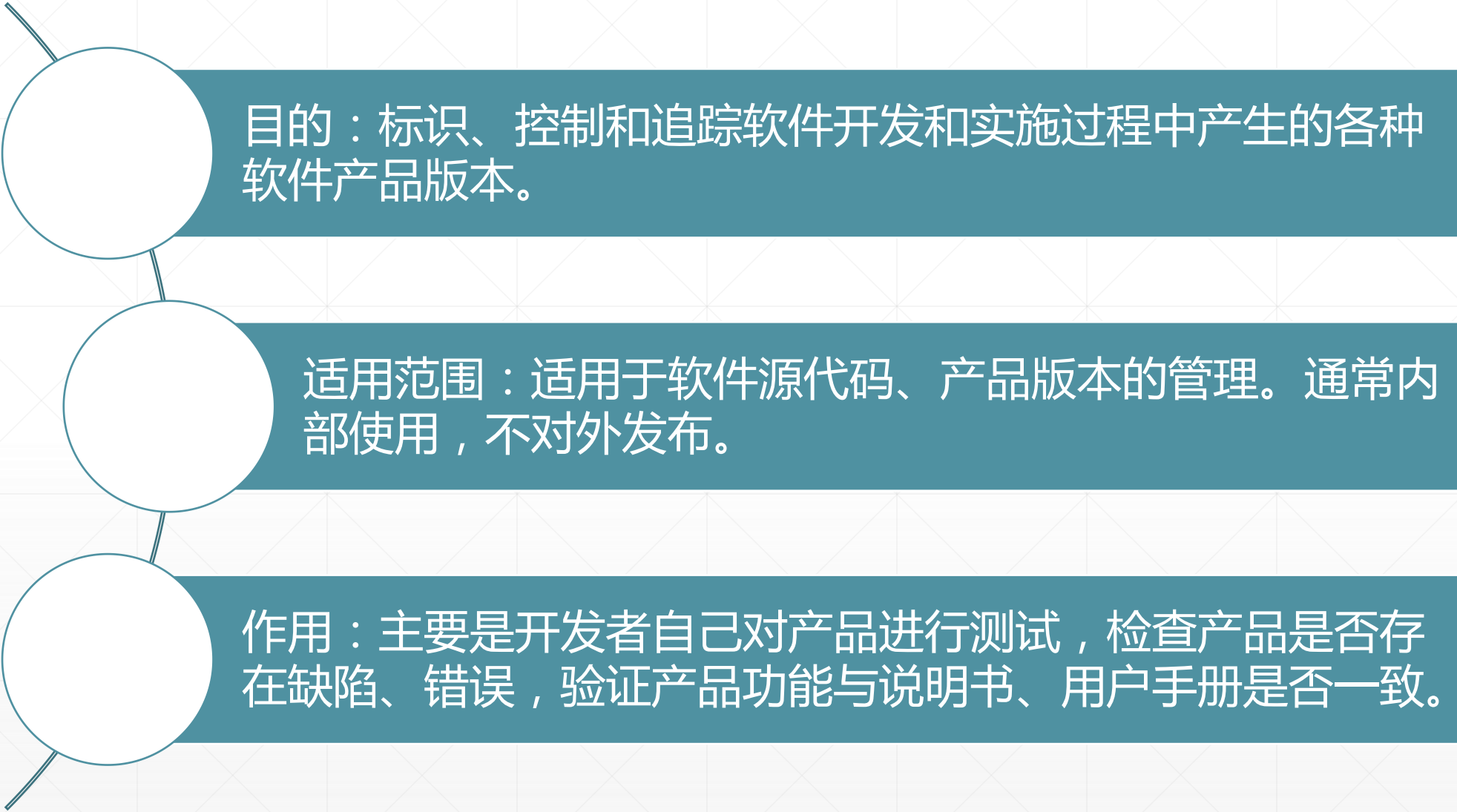




1. 编码管理

- 版本管理的意义
- 常见软件版本
- 常见版本命名规则
- 面向对象程序设计
- 编码策略
- 编码规范

版本管理的意义



目的：标识、控制和追踪软件开发和实施过程中产生的各种软件产品版本。

适用范围：适用于软件源代码、产品版本的管理。通常内部使用，不对外发布。

作用：主要是开发者自己对产品进行测试，检查产品是否存在缺陷、错误，验证产品功能与说明书、用户手册是否一致。

常见软件版本

Alpha版

内部测试版，用作内部测试

Beta版

外部测试版，典型性用户的外部测试

Demo版

演示版，演示部分功能，为发售造势

Enhanced版

增强版或加强版，增加了新功能、新游戏场景的正式发售版本。

Free版

自由版，没有版权，免费给大家使用的版本。

Full Version版

完全版，最终正式发售的版本。

Shareware版

共享版，为了吸引客户，带有限制的版本。

Release版等。

发行版，可从Internet上免费下载

常见版本命名规则

GNU版

主版本号 . 子版本号 [. 修正版本号 [. 编译版本号]]

示例 : 1.2.1,
2.0, 5.0.0
build-13124

Windows版

主版本号 . 子版本号 [修正版本号 [. 编译版本号]]

示例: 1.21,
2.0

Net.Framework 版

主版本号.子版本号[.编译版本号[.修正版本号]]

示例 :
4,4.5,4.6,4.7

目前主要有三种版本控制工具 : CVS、SVN、Git.

面向对象程序设计

主要概念

对象、类、数据抽象、继承、动态绑定、数据封装、多态性、消息传递。

02

03

方法

选择程序设计语言、类的实现、方法的实现、用户接口的实现等

01

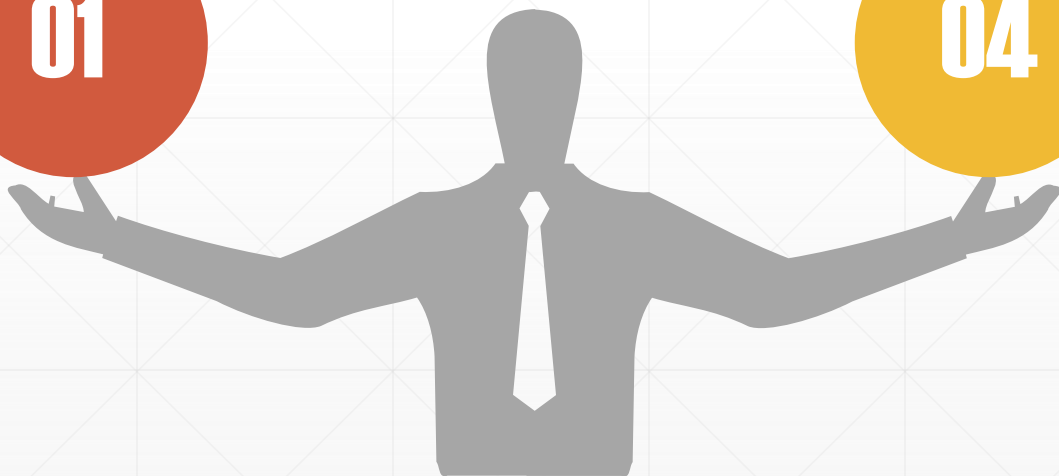
04

含义

面向对象程序设计是以建立模型体现出来的抽象思维过程和面向对象的方法。

语言

C++、JAVA、VB、VC++、C#



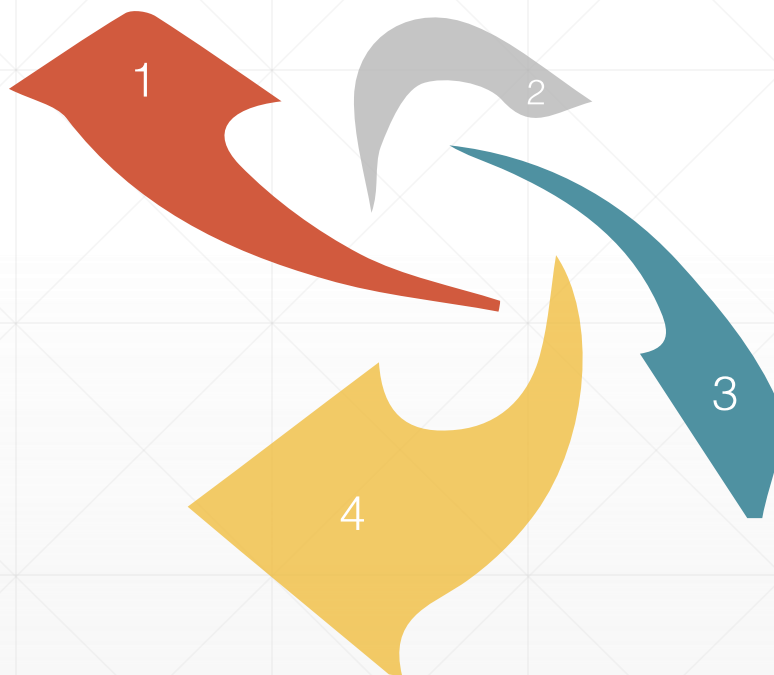
编码策略

自顶向下

从模块的最高层次逐步向下编码

自底向上

从类继承的底层开始向上编码



复合模式

自顶向下与自底向上相结合
可以先总体界面、交互，到基础模块

线程模式

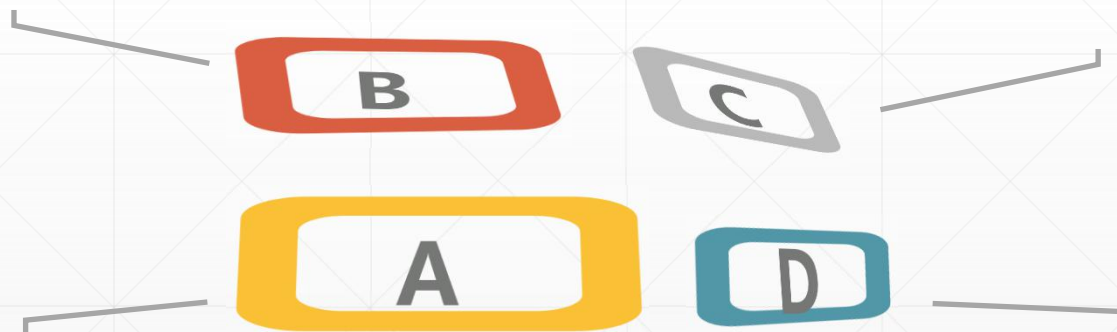
线程：执行关键功能的最小模块集合
先线程，级关键构建，后其他模块

编码规范

编码规范：一个团队、企业给出的内部开发最佳做法的建议、最好方式的推荐和经验总结。

使开发人员有据可依

代码易读



易于测试和维护

易于定位错误、变更管理



授课教师：吴祖峰

电子邮箱：wuzufeng@uestc.edu.cn