北京航空航天大学 计算机学院
School of Computer Science and Engineering,Beihang University

软件工程综合实验示例

-- 软件需求规格说明

北京航空航天大学软件工程研究所
刘超

2016.3

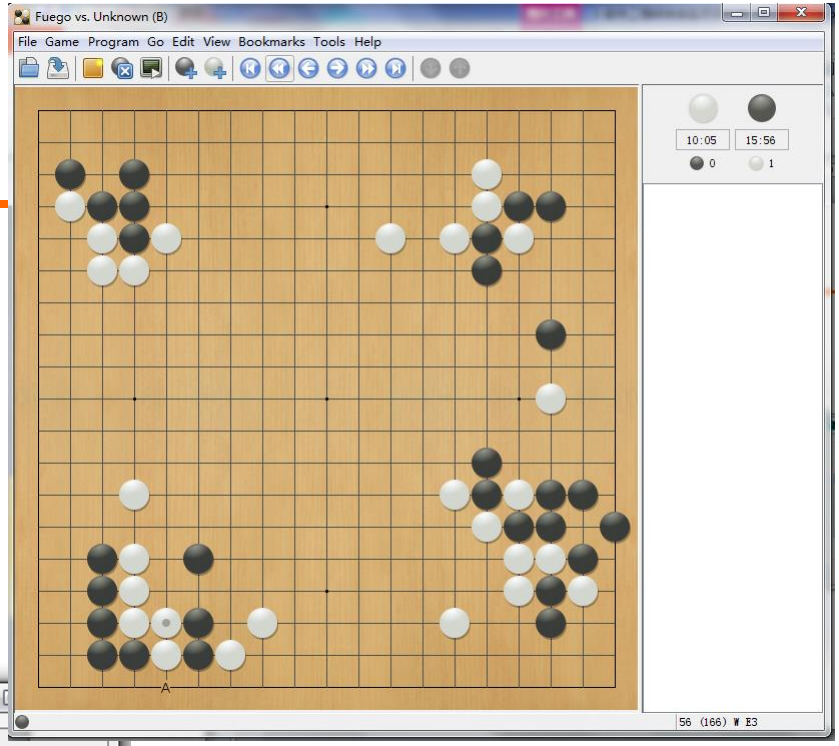
刘超：liuchao@buaa.edu.cn
电话： (10)82317641

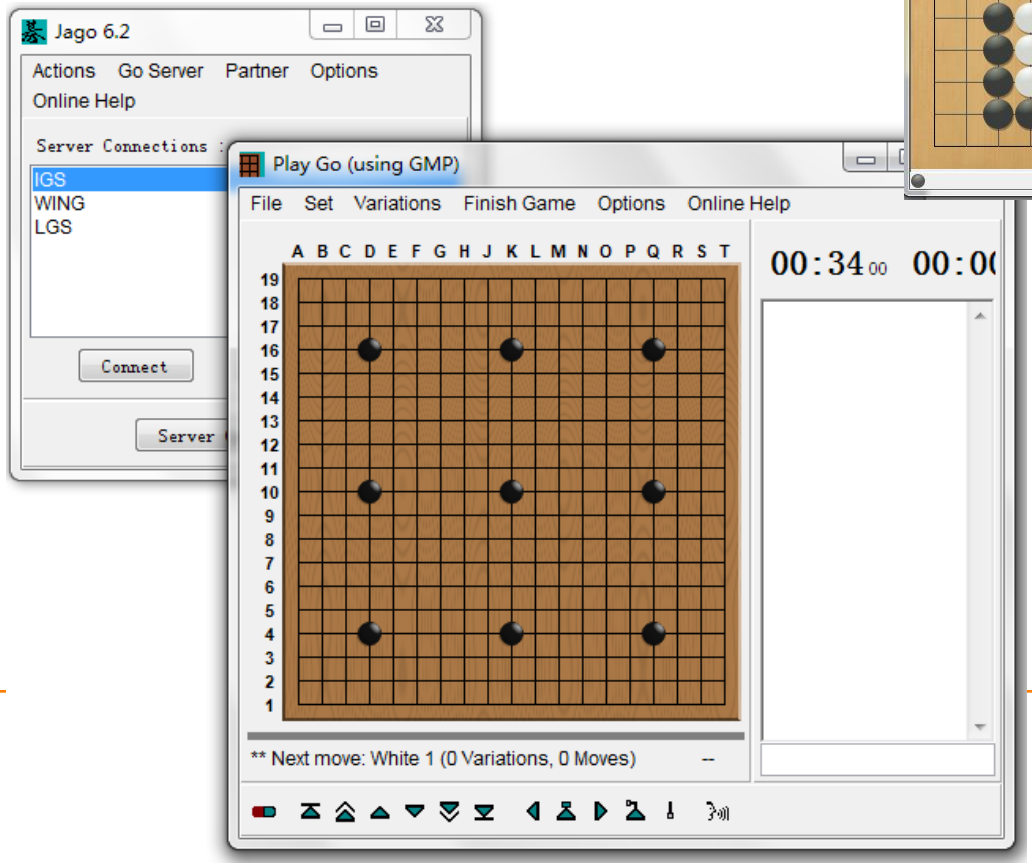
内容提要

- 示例
  - 围棋
- 2014
  - Apache Nutch
  - Linux进程通信
  - Node.js
  - VxWorks5.5内核模块
- 2015
  - Redis
  - Hadoop-MapReduce
  - Lua
  - CompCert C

示例：围棋

- 改进与展示？
  - GoGui-1.4.9
  - GNU Go





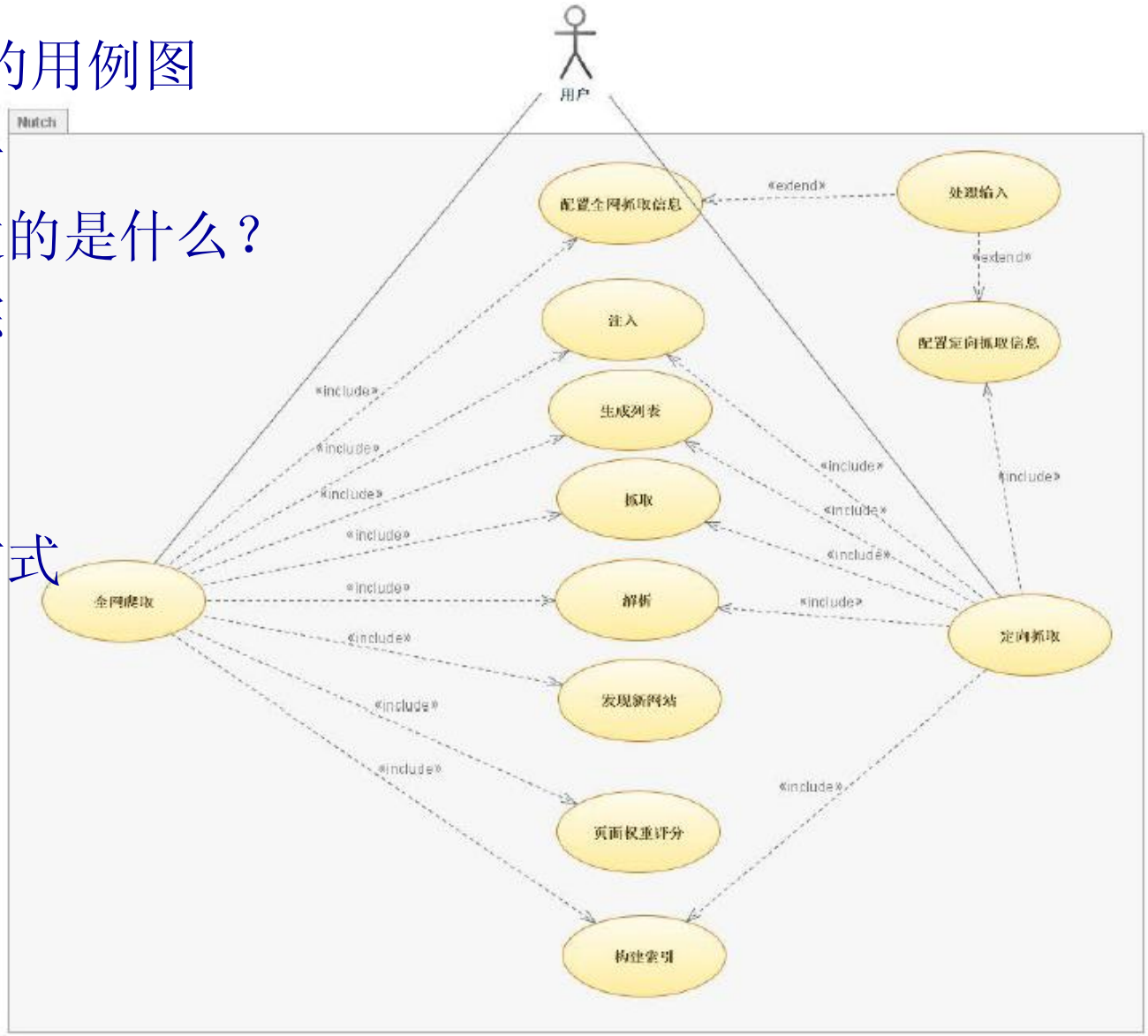
北航软件所

项目1：Apache Nutch

- Apache 基金会的一个开源项目
  - 开源文件索引框架Lucene 项目的一个子项目，后来渐渐发展成长为一个独立的开源项目
- 基于Java 开发，基于Lucene 框架
- 提供Web 网页爬虫功能
  - 提供了一种插件框架，支持各种网页内容的解析、各种数据的采集、查询、集群、过滤等功能的扩展

- Apache Nutch软件系统需求及规格说明
  - 以用例图、状态图的形式给出Nutch系统功能需求的分解结构，并对用例模型中的参与者和用例进行详细的描述，其中主要包括软件系统的用例模型、系统的核心流程等；
  - 使用RUCM模型对功能需求进行建模；
  - 描述了与此次系统实施相关的硬件环境的一些要求；
  - 描述了与此系统实施相关的软件环境的要求；

- Apache Nutch的用例图
  - 命名：统一
  - 说明：描述的是什么？
  - 编号：追踪
- 用户：
  - 两种使用方式
  - 两个用例？



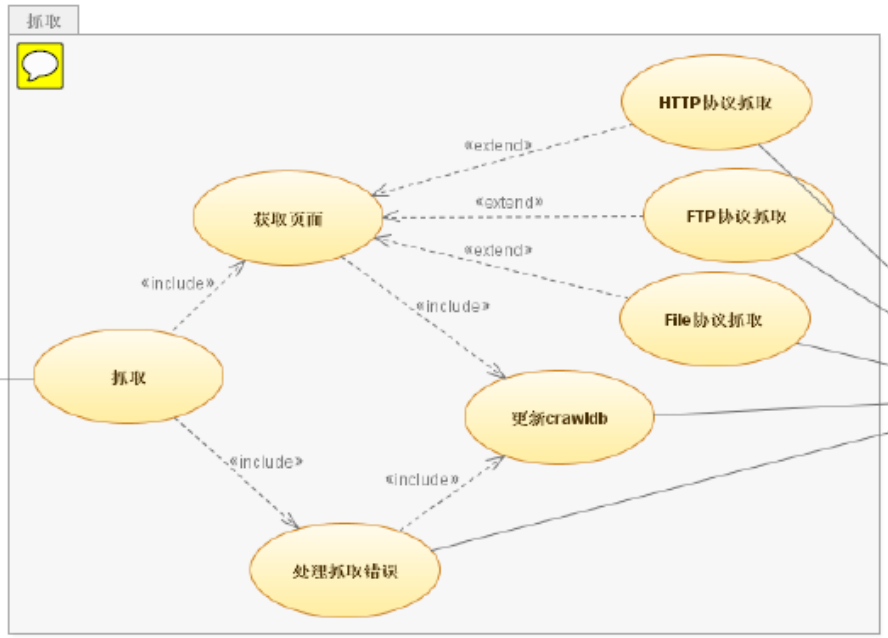
- 对“抓取”用例的细化？

liuchao
2015/3/18 14:41:12

与图1中的抓取用例的关系？

liuchao
2015/2/24 12:13:25

中文？抓取线程？



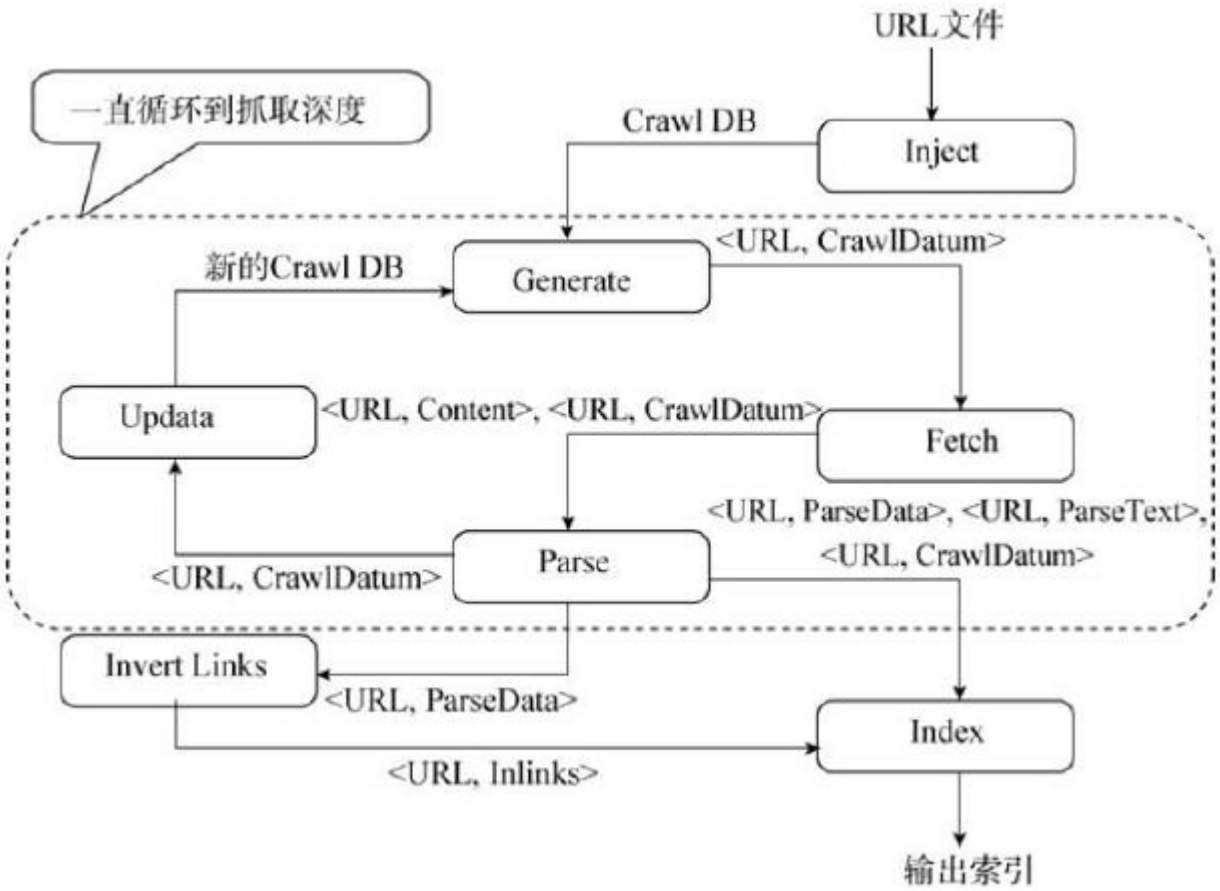
QueueFeeder

FetcherThread

liuchao
2015/3/18 14:40:38

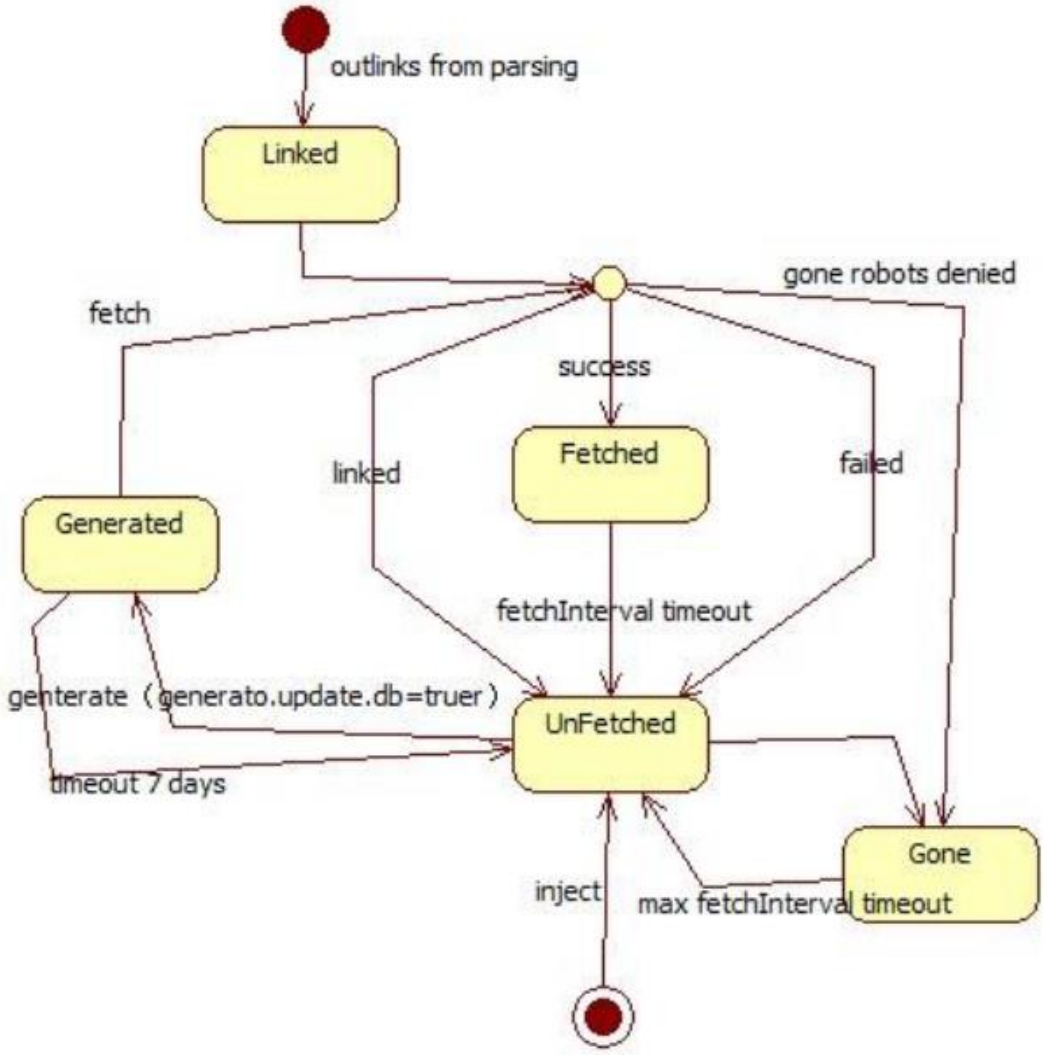
中文？

- Nutch的抓取过程工作流
  - 解释？



北航软件所

- Apache Nutch抓取过程的状态图
  - 解释
  - 布线规则



项目2：

- 全网爬取
  - 解释
  - 编号
  - 追踪

Use Case Specification

Use Case Name	全网爬取
Brief Description	系统根据用户的配置，对全网进行抓取
Precondition	系统初始化完成
Primary Actor	用户
Secondary Actors	None
Dependency	INCLUDE USE CASE Nutch::配置全网抓取信息, INCLUDE USE CASE Nutch::生成列表, INCLUDE USE CASE Nutch::解析, INCLUDE USE CASE Nutch::页面权重评分, INCLUDE USE CASE Nutch::构建索引, INCLUDE USE CASE Nutch::注入, INCLUDE USE CASE Nutch::抓取, INCLUDE USE CASE Nutch::发现新网站
Generalization	None

Basic Flow	Steps	
(Untitled) ▼	1	INCLUDE USE CASE Nutch::配置全网抓取信息
	2	INCLUDE USE CASE Nutch::注入
	3	DO
	4	INCLUDE USE CASE Nutch::生成列表
	5	INCLUDE USE CASE Nutch::抓取
	6	INCLUDE USE CASE Nutch::解析
	7	INCLUDE USE CASE Nutch::发现新网站
	8	INCLUDE USE CASE Nutch::页面权重评分
	9	UNTIL 系统满足用户配置的抓取内容要求
	10	INCLUDE USE CASE Nutch::构建索引
	Postcondition	系统保存合并之后的页面数据以及页面索引

- 配置全网抓取信息

与用例图不一致？

Use Case Specification

Use Case Name	配置全网抓取信息
Brief Description	用户设定待抓取URL地址列表、抓取深度和层次范围、抓取使用的线程或进程情况、时间间隔和更新周期。 系统进行验证
Precondition	系统初始化完成
Primary Actor	用户
Secondary Actors	None
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	**Steps**
	1 系统读取用户输入的待抓取URL地址列表
	2 系统读取用户输入的待抓取深度
	3 系统读取用户输入的待层次范围
	4 系统读取用户输入的抓取使用的线程数量
	5 系统读取用户输入的抓取使用的时间间隔
	6 系统读取用户输入的抓取使用的更新周期
	7 系统读取用户输入的DNS服务器地址
	8 系统 **VALIDATES THAT** 用户输入信息正确
	9 系统生成初始抓取列表
	Postcondition 系统成功保存初始抓取列表

Bounded Alternative Flow (Untitled) ▼	**RFS** Basic Flow 7-8
	1 系统显示错误信息
	2 **ABORT**
	Postcondition 用户指定的URLs未被抓取，系统停止

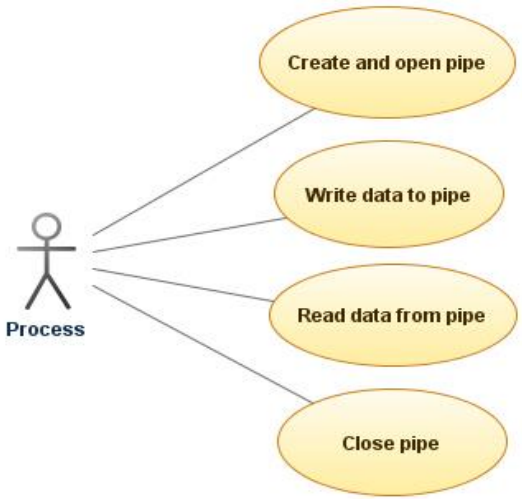
项目2：Linux进程通信

- Linux使用的进程间通信方式：
  - **管道(pipe)与有名管道 (named pipe):** 管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。
  - **信号量(semaphore):** 信号量是一个计数器，可以用来控制多个进程对共享资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源时，其他进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。

- **消息队列(message queue):** 消息队列是由消息的链表，存放在内核中并由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。
- **信号 (signal):**信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。
- **共享内存(shared memory):**共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号两，配合使用，来实现进程间的同步和通信。

- 典型应用场景分析
  - **管道和有名管道**
    - 管道主要用于父进程与子进程之间，或者两个兄弟进程之间。首先，由父进程创建一个管道后通过系统调用产生子进程。接着，父进程关闭这个通道的读出端，子进程关闭同一通道的写入端。这就在父子进程间提供了一个单向数据流。管道是在内核中的数据结构，所以子进程不会复制管道。但管道的文件描述符在进程空间中，所以子进程获得了管道的文件描述符副本
    - 有名管道是为了解决无名管道只能用于近亲进程之间通信的缺陷而设计的。有名管道是建立在实际的磁盘介质或文件系统（而不是只存在于内存中）上有自己名字的文件，任何进程可以在任何时间通过文件名或路径名与该文件建立联系。有名管道常被用在shell命令行将数据从一条命令传送到另一条命令，而不需要创建中间的临时文件，以及在客户--服务器结构中，使用有名管道在客户和服务器之间交换数据

- 功能需求概述
  - 管道



Use Case Specification	
Use Case Name	Create and open pipe
Brief Description	Create a pipe for unidirectional communication.
Precondition	The kernel is running. The processes have relatives.
Primary Actor	Parent process, child process
Secondary Actors	Kernel
Dependency	None
Generalization	None

Basic Flow (Untitled) ▼	Steps	
	1	The system VALIDATES THAT the number of file descriptors the parent process uses is under the OS limit.
	2	The system VALIDATES THAT the OS limit on the total number of open files is not reached.
	3	The parent process creates a duplex communication channel.
	4	The parent process creates its child process.
	5	IF the pipe is used for parent-child communication THEN
	6	The parent process and the child process close one ends of the communication channel respectively to make the channel unidirectional.
	7	ELSEIF the pipe is used for brother processes' communication THEN
	8	The child process A and the child process B close one ends of the communication channel respectively to make the channel unidirectional.
	9	ENDIF
	Postcondition	The pipe is created and opened.

Specific Alternative Flow (Untitled) ▼	RFS 1	
	1	The kernel returns an error code to the parent process.
	2	ABORT
	Postcondition	The pipe is not created.

Specific Alternative Flow (Untitled) ▼	RFS 2	
	1	The kernel returns an error code to the parent process.
	2	ABORT
	Postcondition	The pipe is not created.

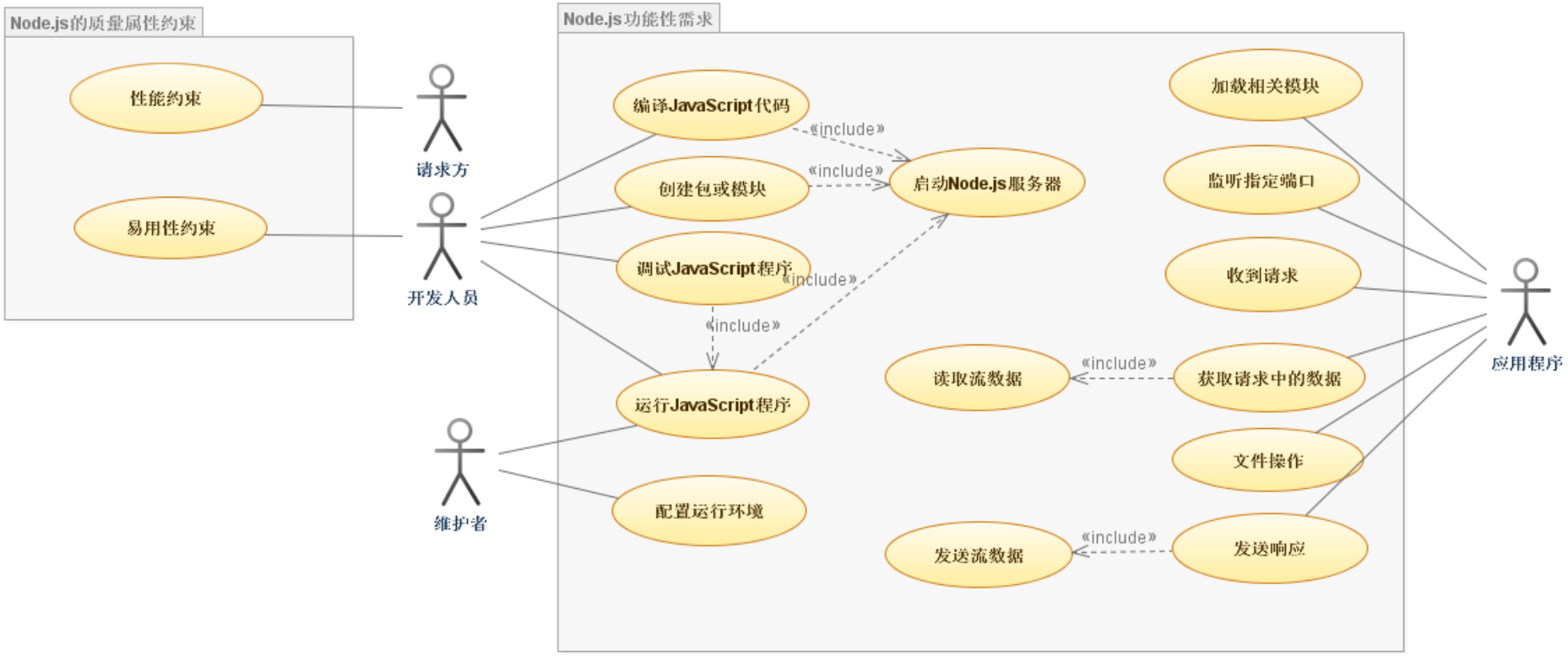
北航软件所

项目3：Node.js

- 支持JavaScript 运行在服务器端的平台
  - 让JavaScript 脱离浏览器的束缚运行在一般的服务器环境下，就像运行 Python、Perl、PHP、Ruby程序一样
  - Node.js 内建了 HTTP 服务器支持，这和 PHP、Perl 不一样，因为在使用 PHP 的时候，必须先搭建一个 Apache 之类的HTTP 服务器

- 使用 Node.js，可以开发：
  1. 具有复杂逻辑的网站；
  2. 基于社交网络的大规模 Web 应用；
  3. Web Socket 服务器；
  4. TCP/UDP 套接字应用程序；
  5. 命令行工具；
  6. 交互式终端程序；
  7. 带有图形用户界面的本地应用程序；
  8. 单元测试工具；
  9. 客户端 JavaScript 编译器

- 用户和相应的需求用例


Node.js的质量属性约束
性能约束
易用性约束
请求方
开发人员
维护者
Node.js功能性需求
编译JavaScript代码
创建包或模块
调试JavaScript程序
运行JavaScript程序
配置运行环境
启动Node.js服务器
«include»
«include»
«include»
«include»
加载相关模块
监听指定端口
收到请求
读取流数据
获取请求中的数据
文件操作
发送流数据
发送响应
«include»
«include»
应用程序

项目3：VxWorks5.5内核模块

- **VxWorks 5.5内核模块——项目说明**

目录

北航软件所

1.2.1. VxWorks简介

VxWorks是美国风河（WindRiver）公司于1983年设计开发的一种高模块化、高性能的嵌入式实时操作系统（RTOS），是嵌入式开发环境的关键组成部分。它具有很好的安全性、可靠性（如冗余性[多CPU]、容错性[检测、隔离、恢复]）以及系统灵活性（如高可剪裁、超过1800个应用程序接口）。

1.2.2.项目简介

VxWorks核心是一个实时微内核，主要包括基于优先级的任务调度、任务同步和通信、中断处理、定时器和内存管理。

- 内核的基本功能可以分为如下几大类：
- 任务管理；
- 事件机制和异步信号服务；
- 信号量服务；
- 消息队列服务；
- 内存管理；
- 中断服务程序；
- 时钟管理和定时器服务；
- 出错处理。

本项目着重研究VxWorks内核的部分模块（包括内存管理，任务管理，中断处理），分析并获取其需求，采用规范的软件需求定义方法（RUCM）制定相应的软件需求规格说明书，并进一步定义软件测试需求，设计并实现软件测试用例，生成相关文档。

- Hadoop-MapReduce
- Lua
- CompCert C

Redis：远程字典服务(REmote Dictionary Server)

- NoSQL 系列中的一种key-value 类型的轻量级内存数据库
  - 数据类型：string(字符串)、lists(链表)、sets (集合)、zsets(有序集合)和hash 等
  - 数据操作：push、pop、add、remove，取交集、并集和差集，各种排序
  - 数据都保存在内存中,但是Redis 会周期性地把更新的数据写入磁盘或把修改操作写入追加的记录文件,并且在此基础上实现了 master-slave 同步
    - 因为是纯内存操作，Redis 的性能非常出色,每秒可以处理超10 万次读写操作，是已知性能最快的Key-Value 数据库

Hadoop-MapReduce

- MapReduce 是一种编程模型，用于大规模数据集（大于1TB）的并行运算
- 基本思想来源于函数式编程语言和矢量编程语言
- 极大地方便了编程人员在不会分布式并行编程的情况下，将自己的程序运行在分布式系统上
- 概念"Map（映射）"和"Reduce（归约）“
  - 指定一个Map（映射）函数，用来把一组键值对映射成一组新的键值对
  - 指定并发的Reduce（归约）函数，用来保证所有映射的键值对中的每一个共享相同的键组

Lua

- 一种小巧的脚本语言
  - 嵌入到应用程序中，为其提供灵活的扩展和定制功能
  - 由标准C 编写而成，几乎在所有操作系统和平台上都可以编译运行
  - 并没有提供强大的库，所以不适合作为开发独立应用程序的语言
  - 有一个同时进行的JIT 项目，提供在特定平台上的即时编译功能。
- 巴西里约热内卢天主教大学（PontificalCatholic University of Rio de Janeiro ）的一个研究小组于1993年开发

CompCert C

- C compiler
  - for life-critical and mission-critical software meeting high levels of assurance
  - Performance of the generated code is decent but not outstanding.
  - **formal verification**, using machine-assisted mathematical proofs, to be exempt from miscompilation issues. In other words, the executable code it produces is proved to behave exactly as specified by the semantics of the source C program.