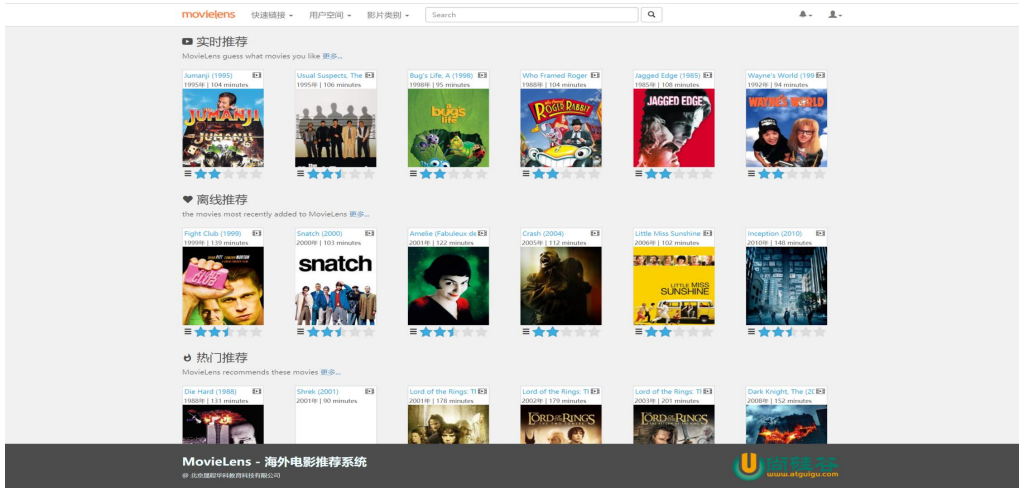
电影推荐系统设计

讲师：武晟然

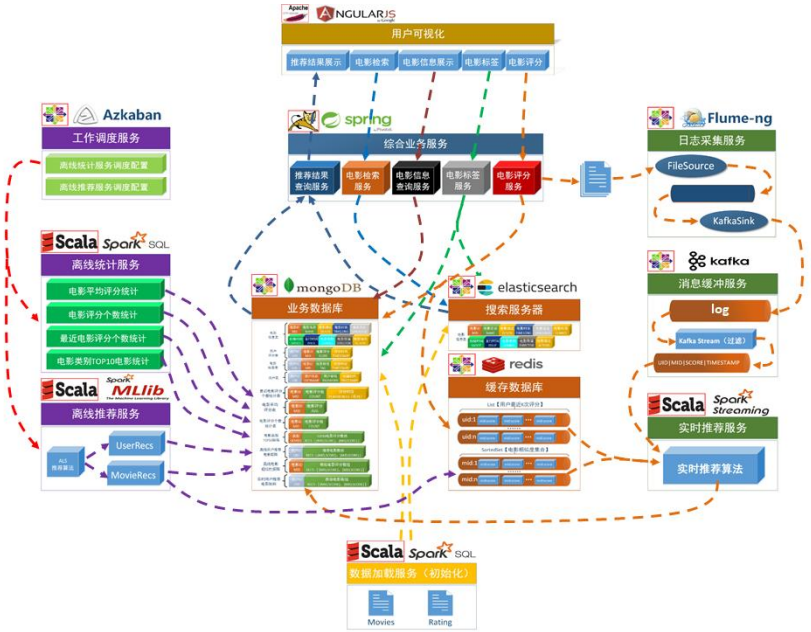
主要内容

- 项目框架
- 数据源解析
- 统计推荐模块
- 离线推荐模块
- 实时推荐模块
- 基于内容的推荐模块



项目框架

- 大数据处理流程
- 系统模块设计
- 项目系统架构
- 项目数据流图

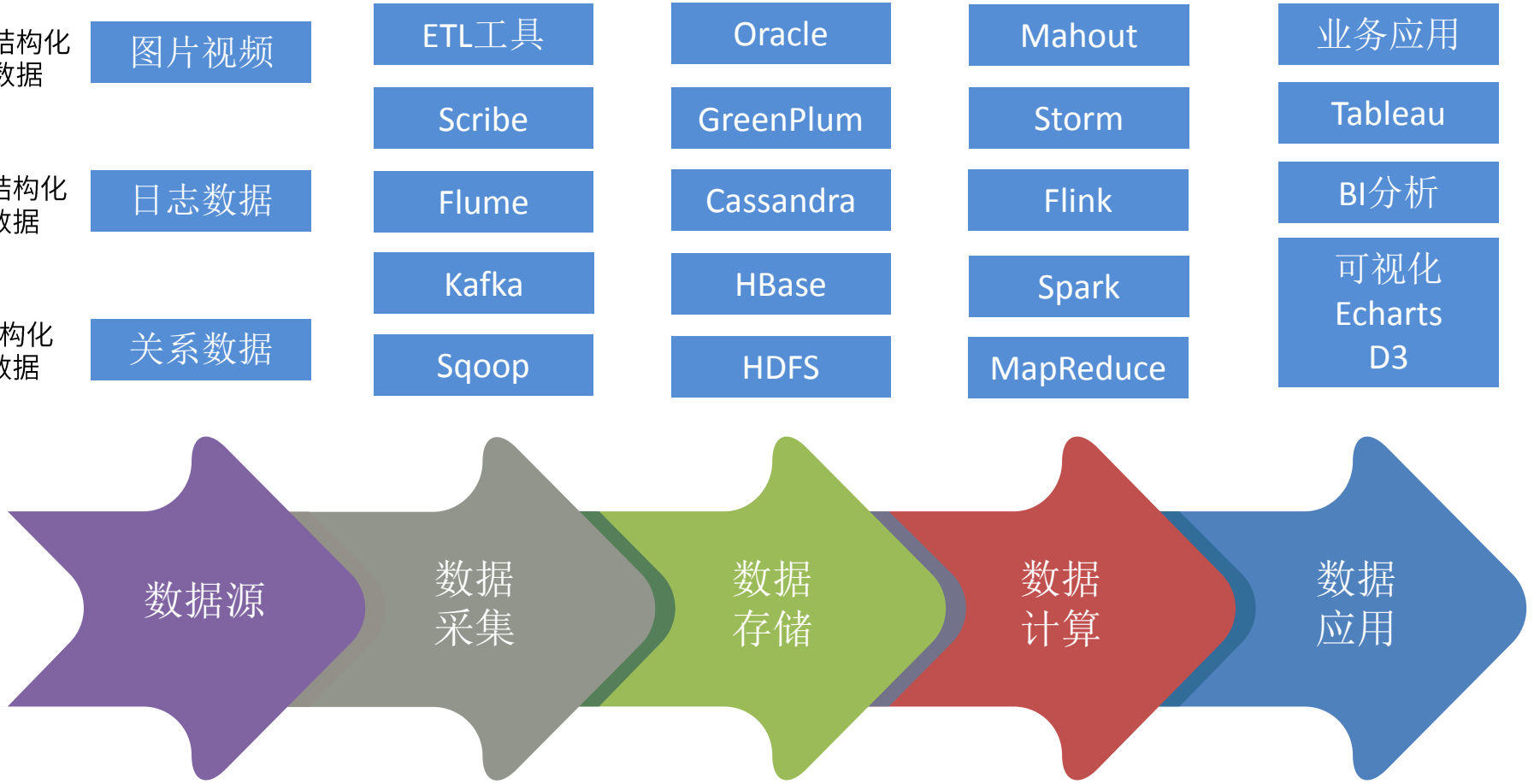


数据生命周期

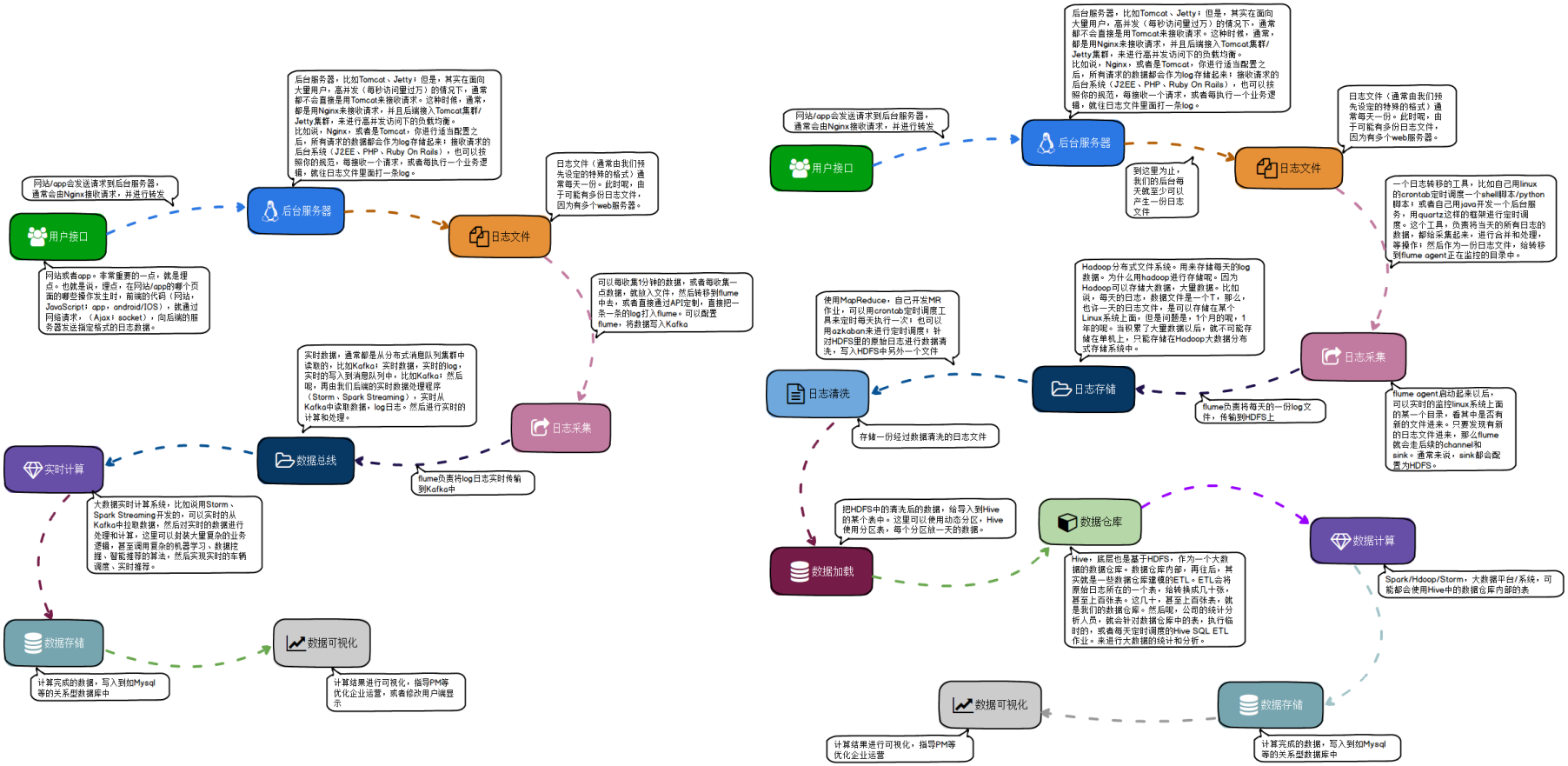
非结构化数据

半结构化数据

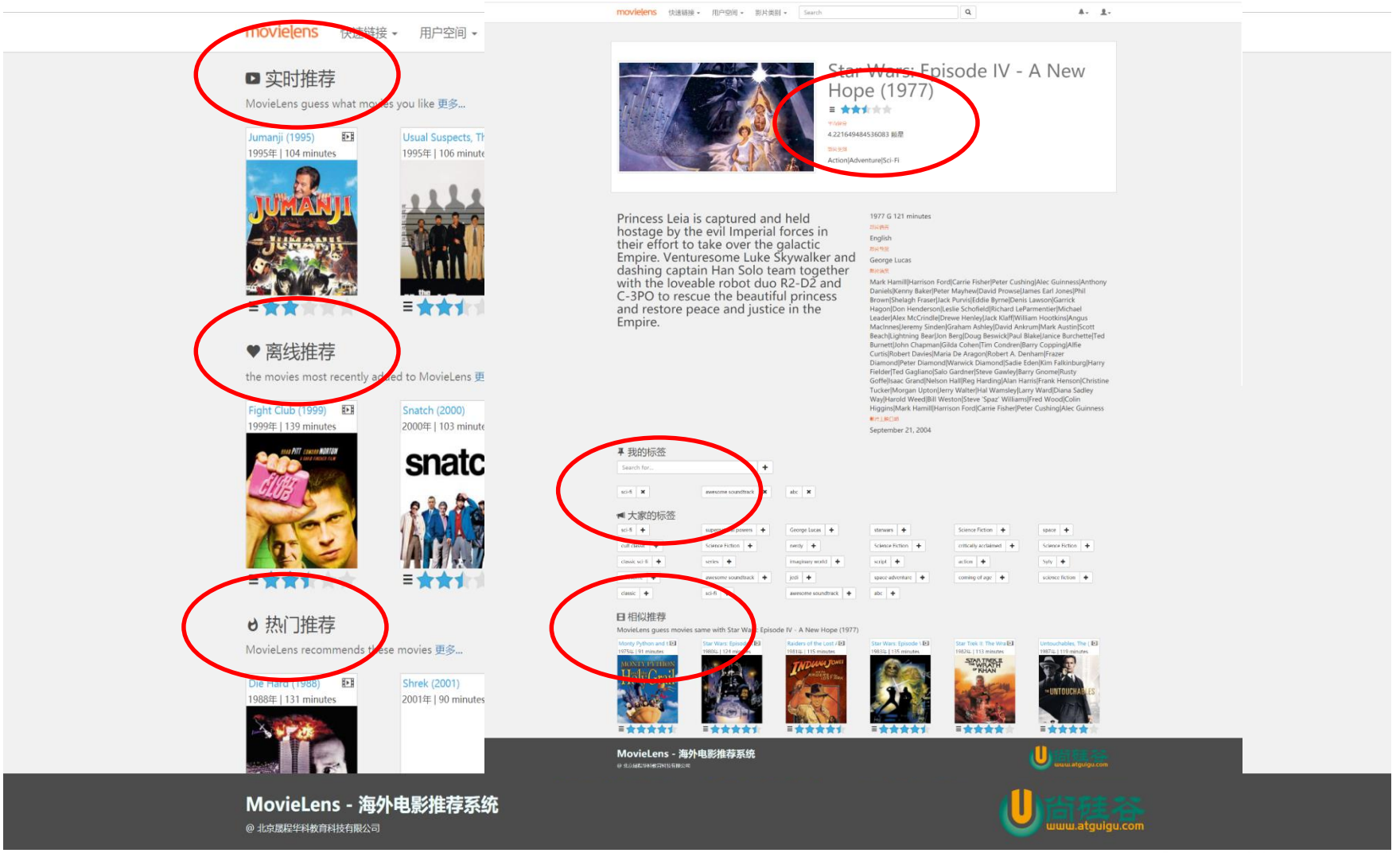
结构化数据



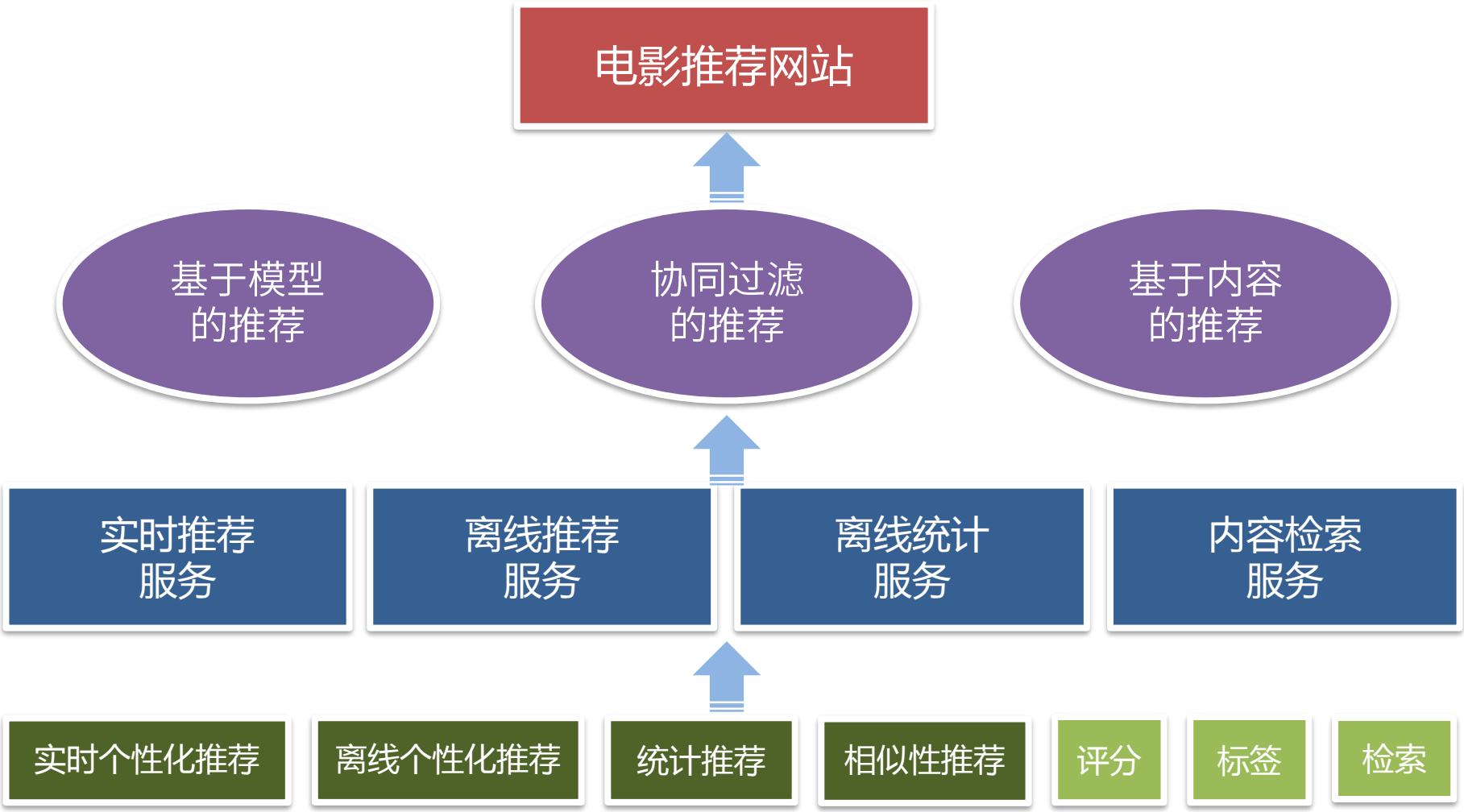
大数据处理流程



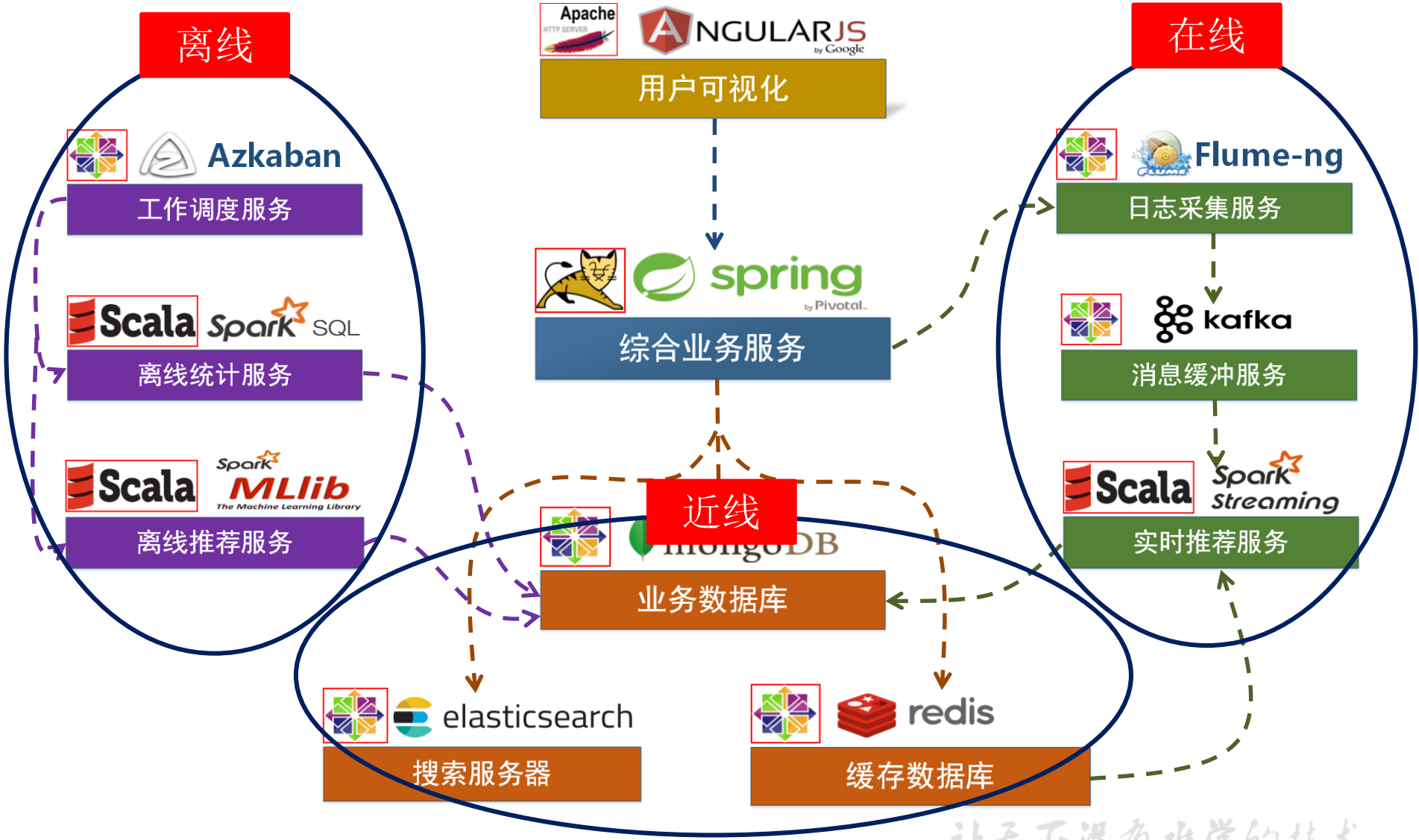
我们的目标

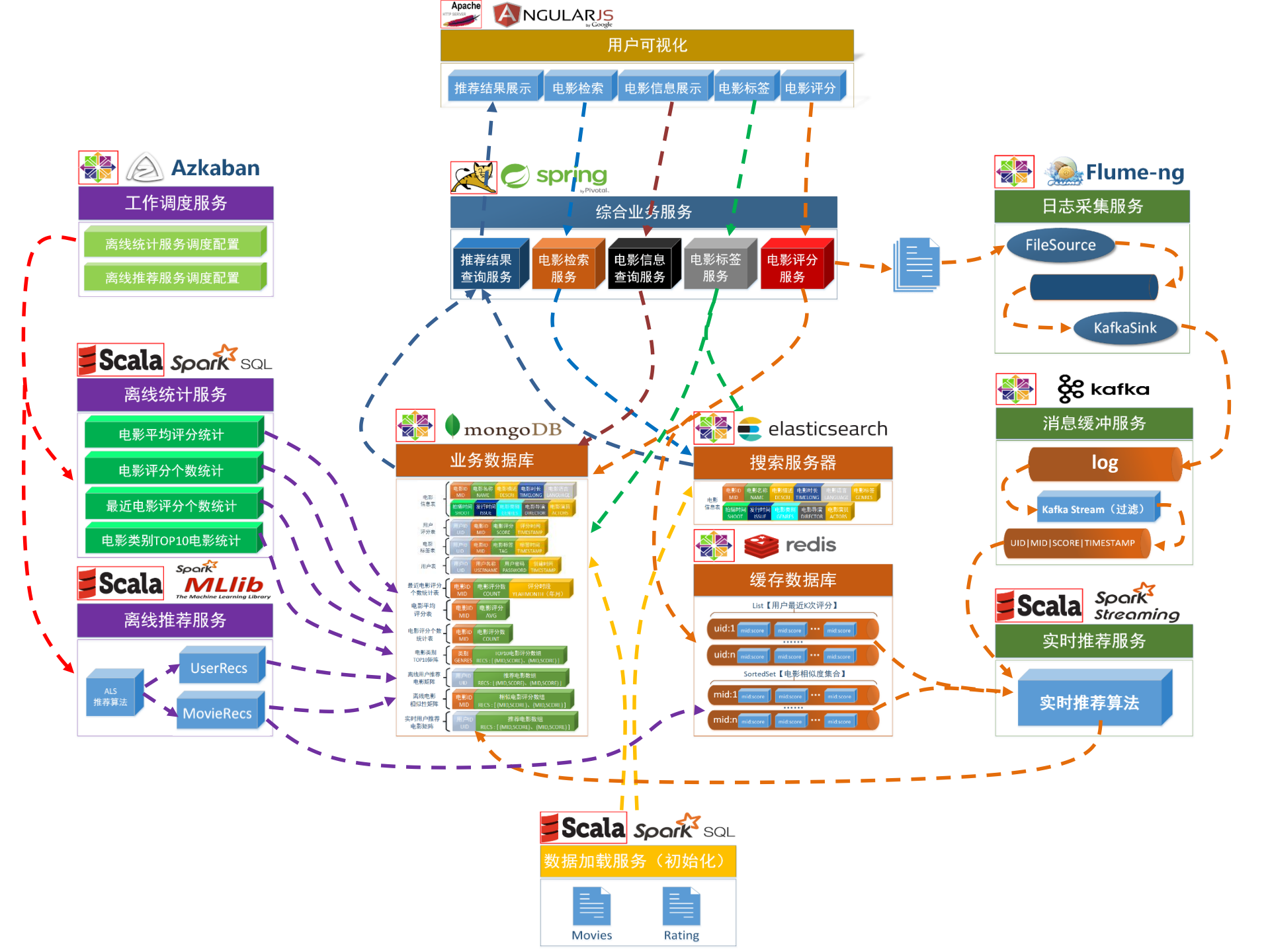


系统模块设计

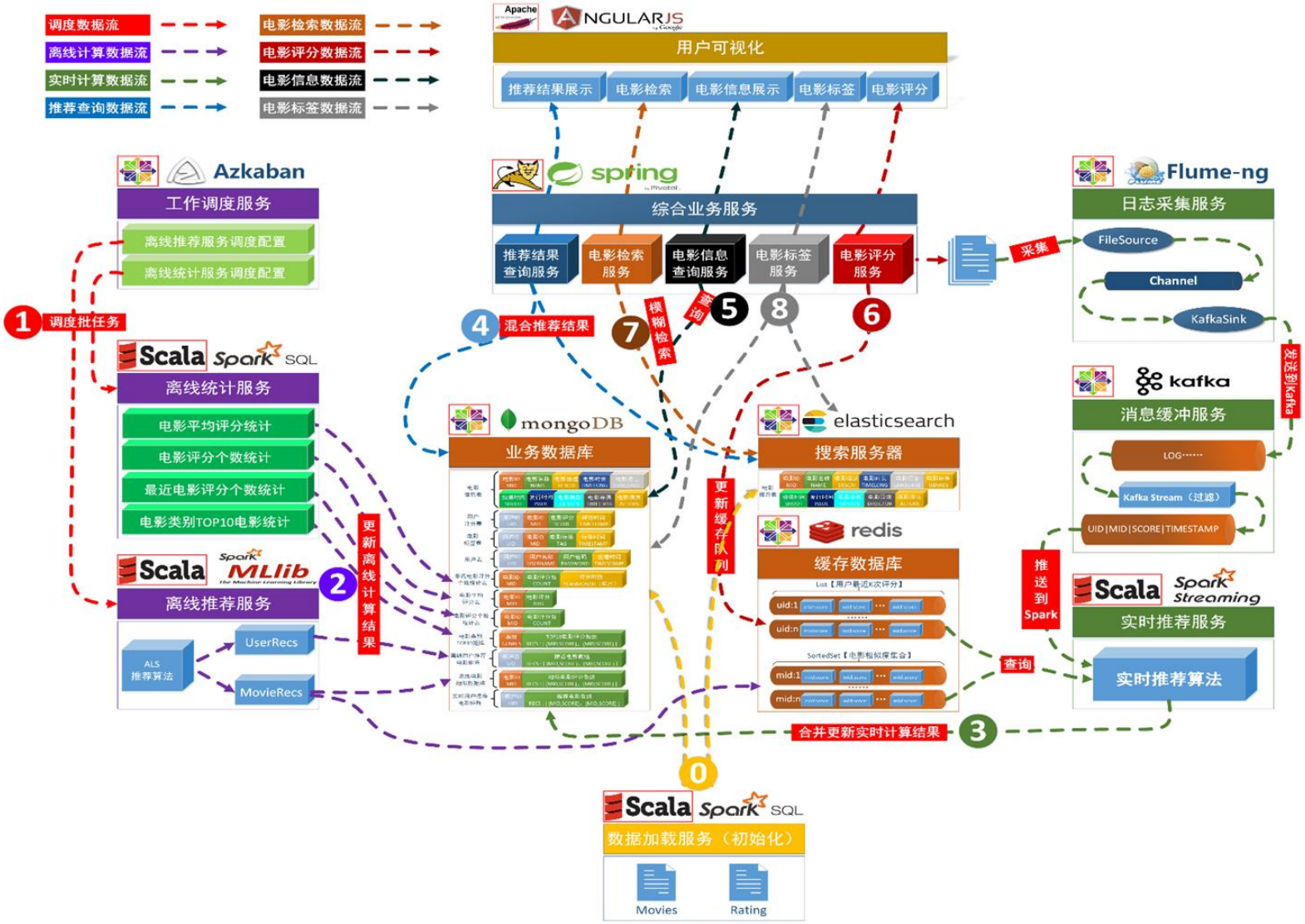


项目系统架构




Apache
ANGULARJS
用户可视化
推荐结果展示
电影检索
电影信息展示
电影标签
电影评分
Azkaban
工作调度服务
离线统计服务调度配置
离线推荐服务调度配置
spring
综合业务服务
推荐结果查询服务
电影检索服务
电影信息查询服务
电影标签服务
电影评分服务
Flume-ng
日志采集服务
FileSource
KafkaSink
Scala
Spark SQL
离线统计服务
电影平均评分统计
电影评分个数统计
最近电影评分个数统计
电影类别TOP10电影统计
mongoDB
业务数据库
elasticsearch
搜索服务器
kafka
消息缓冲服务
log
Kafka Stream（过滤）
UID|MID|SCORE|TIMESTAMP
redis
缓存数据库
List【用户最近K次评分】
uid:1
uid:n
SortedSet【电影相似度集合】
mid:1
mid:n
Scala
Spark MLlib
The Machine Learning Library
离线推荐服务
ALS 推荐算法
UserRecs
MovieRecs
Scala
Spark Streaming
实时推荐服务
实时推荐算法
Scala
Spark SQL
数据加载服务（初始化）
Movies
Rating

系统数据流图



数据源解析

- 电影信息
- 用户评分信息
- 电影标签信息

电影信息

电影ID (MID)	电影名称 (NAME)	电影描述 (DESCRI)	电影时长 (TIMELONG)	发行时间 (ISSUE)	拍摄时间 (SHOOT)	电影语言 (LANGUAGE)	电影类别 (DIRECTOR)	电影演员 (ACTORS)	电影导演 (DIRECTOR)
1	Toy Story	-	81minutes	March 20 2001	1995	English	Adventure|Animation|…|Fantasy	Tom Hanks|…|Jim	John Lasseter
…	…	…	…	…	…	…	…	…	…
30	Shanghai Triad	-	108minutes	December 12 2000	1995	Chinese	Crime|Drama	Gong Li|…|Li Bao-Tian	Zhang Yimou

用户评分信息

用户ID (UID)	电影ID (MID)	电影评分 (SCORE)	评分时间 (TIMESTAMP)
671	5816	4	1065111963
671	5902	3.5	1064245507
…	…	…	…
671	5952	5	1063502716

电影标签信息

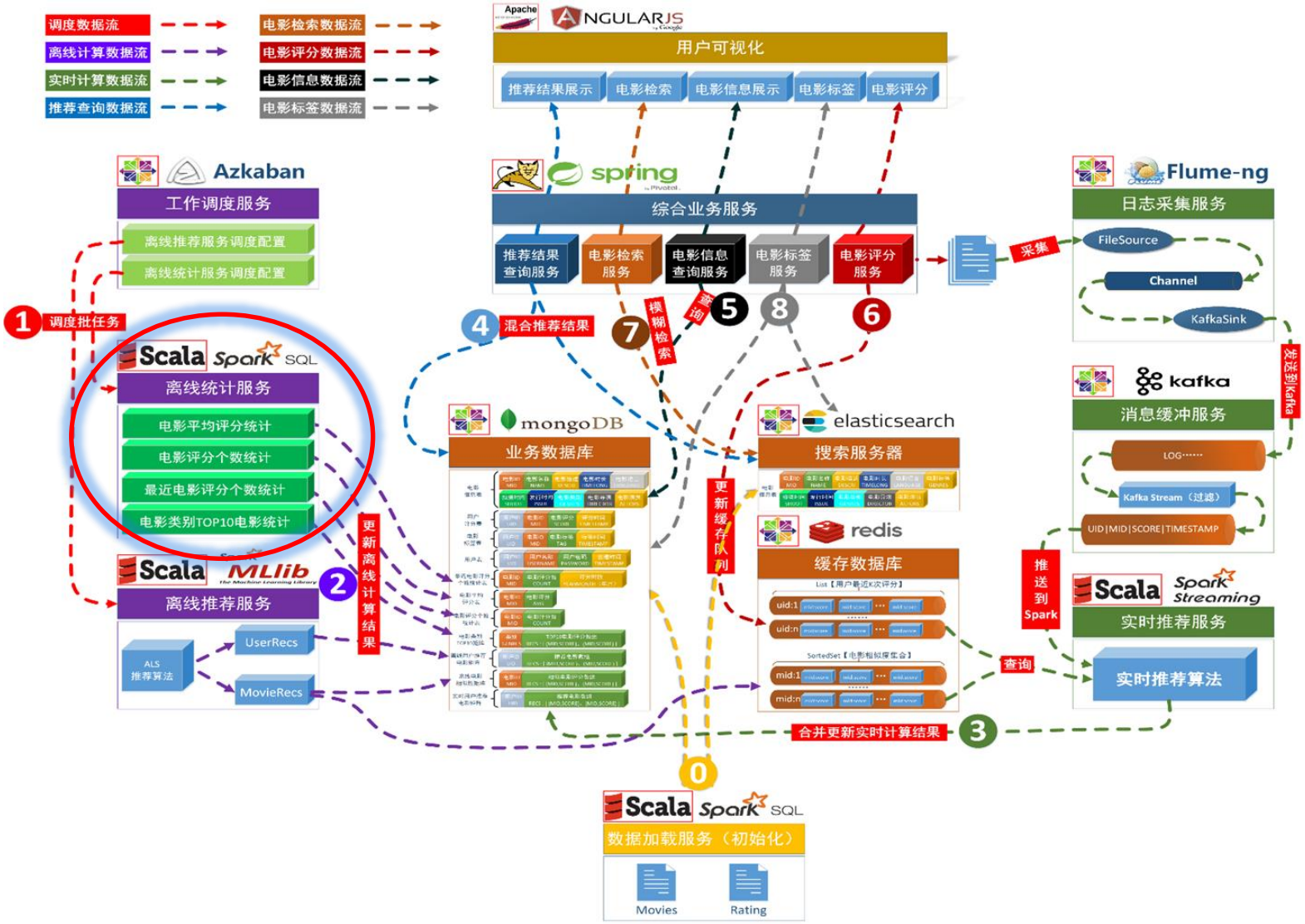
用户ID (UID)	电影ID (MID)	电影标签 (TAG)	标签时间 (TIMESTAMP)
15	339	sandra 'boring' bullock	1138537770
15	1955	dentist	1193435061
…	…	…	…
15	100365	uganda	1425876220

主要数据模型

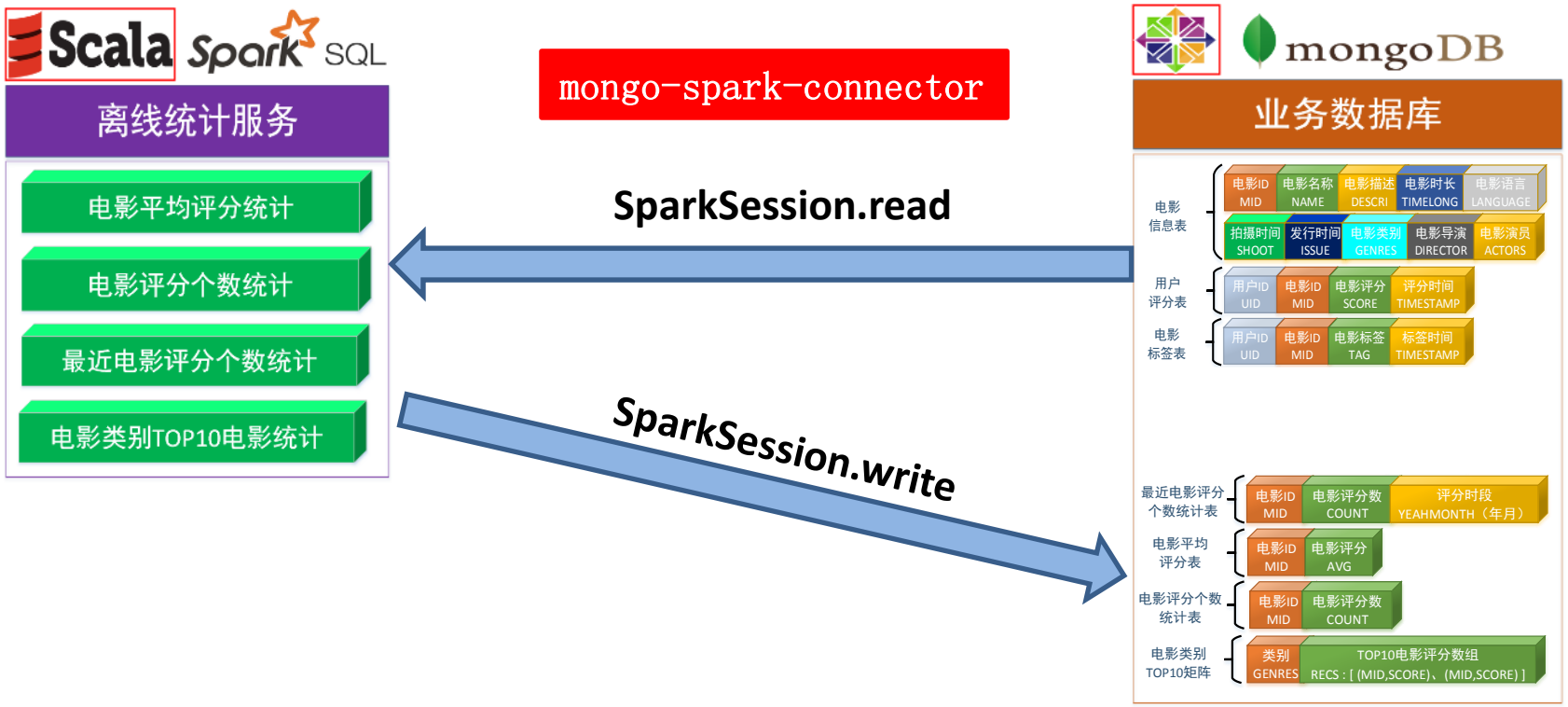
统计推荐模块

- 历史热门电影统计
- 近期热门电影统计
- 电影平均评分统计
- 各类别 Top10 优质电影统计

统计推荐模块



统计推荐模块



历史热门电影统计

- 统计所有历史数据中每个电影的评分数
- select mid, count(mid) as count from ratings group by mid

  → RateMoreMovies

- RateMoreMovies 数据结构：mid，count

近期热门电影统计

- 统计每月的电影评分个数，就代表了电影近期的热门度
- select mid, score, changeDate(timestamp) as yearmonth from ratings

  ➡ ratingOfMonth
- select mid, count(mid) as count ,yearmonth from ratingOfMonth group by yearmonth,mid order by yearmonth desc,count desc

  ➡ RateMoreRecentlyMovies
- changDate：UDF函数，使用 SimpleDateFormat 对 Date 进行格式转化，转化格式为“yyyyMM”
- RateMoreRecentlyMovies 数据结构：mid，count，yearmonth

电影平均评分统计

- select mid, avg(score) as avg from ratings group by mid

AverageMovies

- AverageMovies 数据结构：mid，avg

各类别 Top10 评分电影统计

- select a.mid, genres, if(isnull(b.avg),0,b.avg) score from movies a left join averageMovies b on a.mid = b.mid

  → movieWithScore

- spark.sql("select * from (select " +

  "mid," +

  "gen," +

  "score, " +

  "row_number() over(partition by gen order by score desc) rank " +

  "from " +

  "(select mid,score,explode(splitGe(genres)) gen from movieWithScore)
  genresMovies) rankGenresMovies " +

  "where rank <= 10")

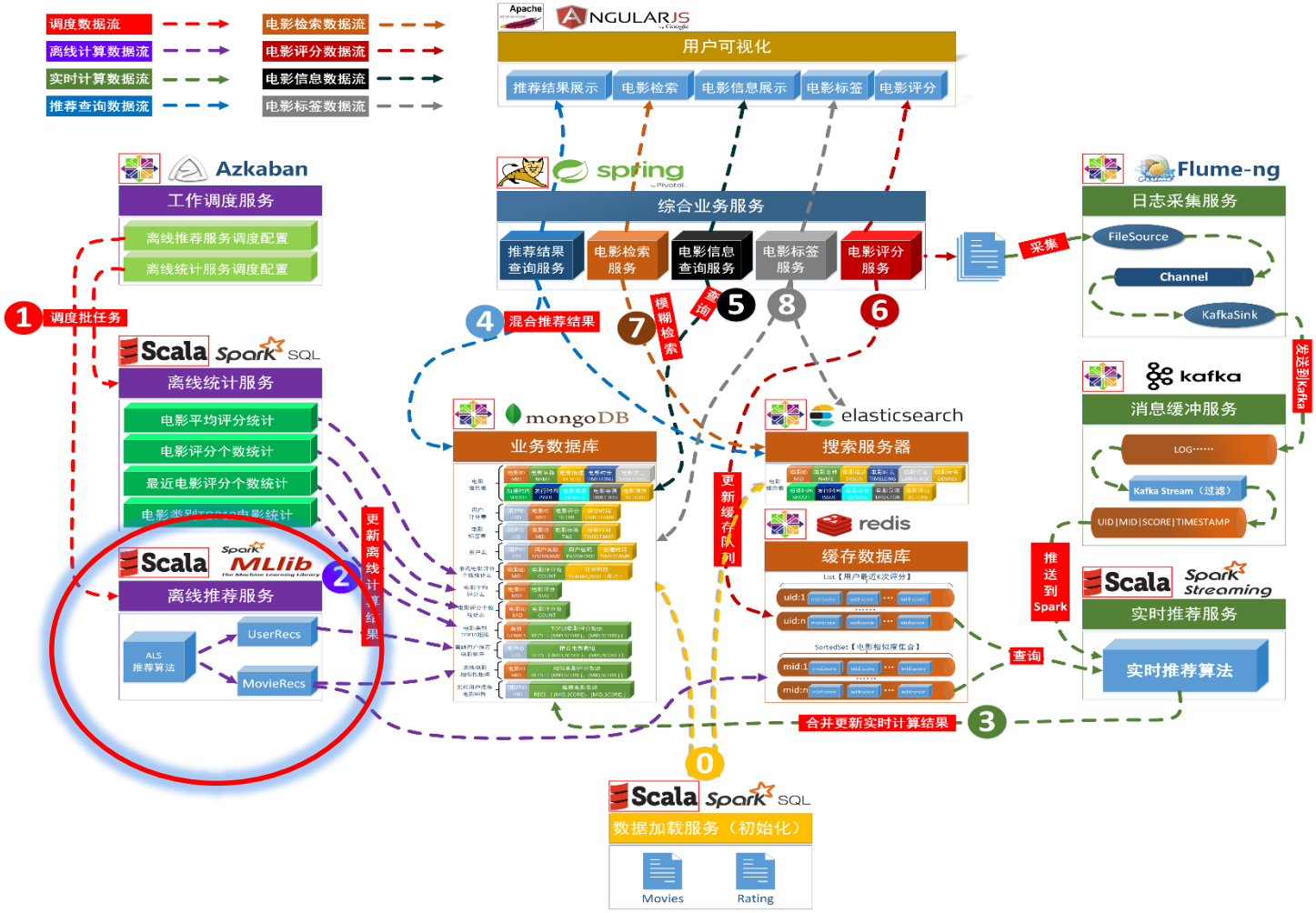
- splitGe：UDF函数，按照‘|’字符对字符串进行切分

离线推荐模块

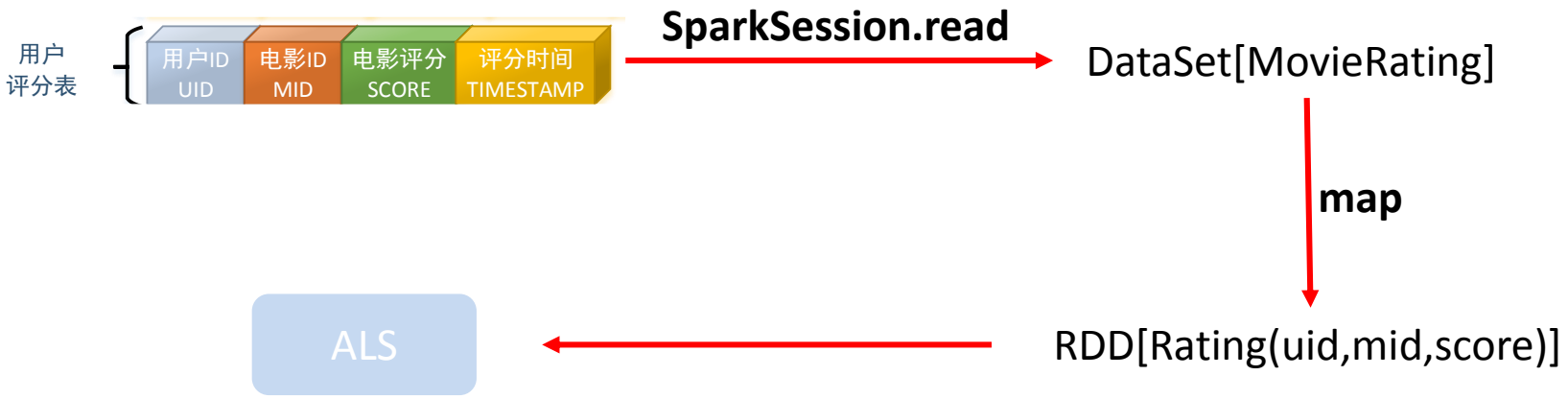
- 用ALS算法训练隐语义模型
- 计算用户推荐矩阵
- 计算电影相似度矩阵

离线推荐模块

电影推荐系统数据流图



ALS推荐模型训练



val model = ALS.train(trainData,rank,iterations,lambda)

- RMSE

  均方根误差：均方误差的算术平方根，预测值与真实值之间的误差

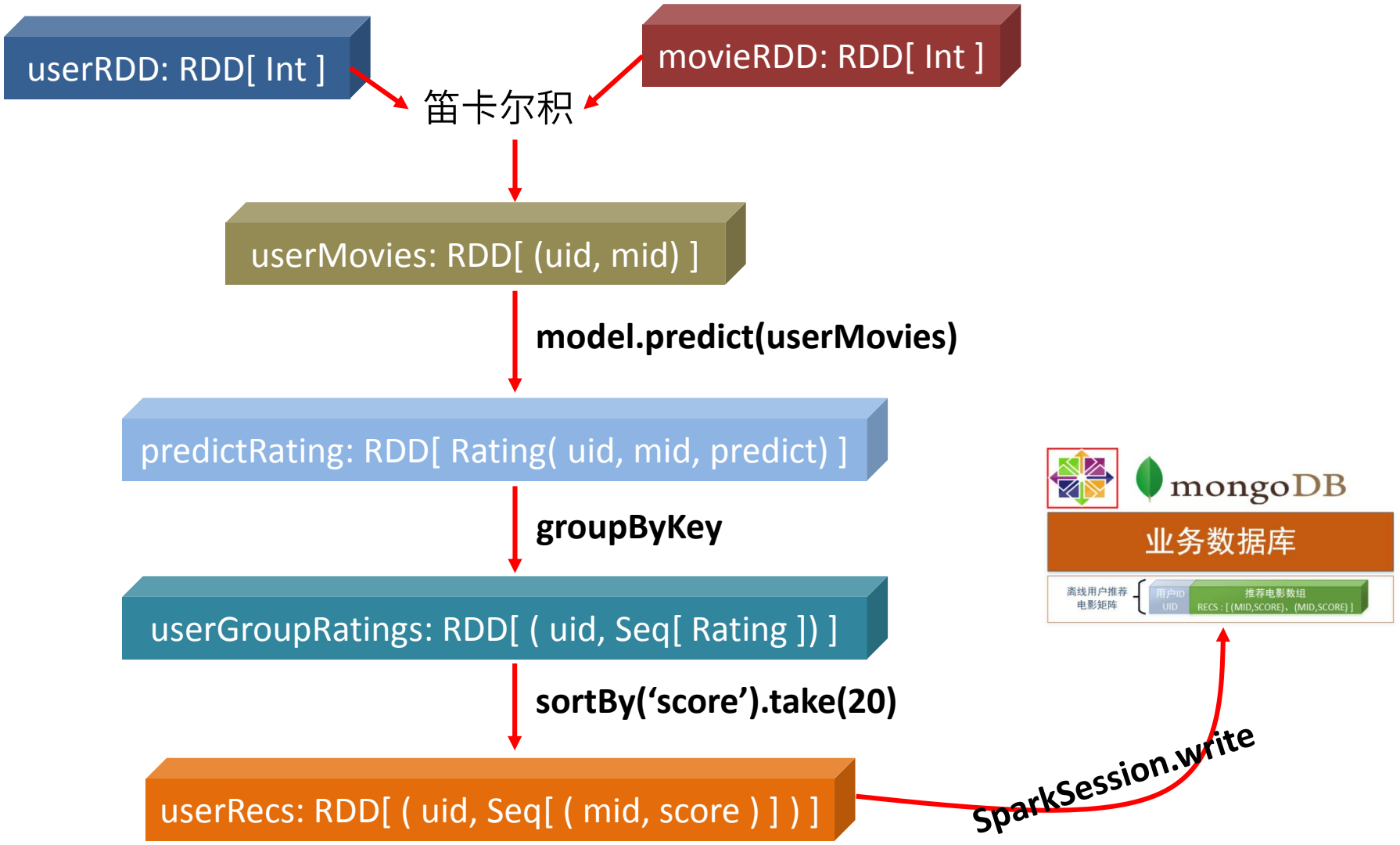
$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(observed_t - predicted_t)^2}$$

- **参数调整**

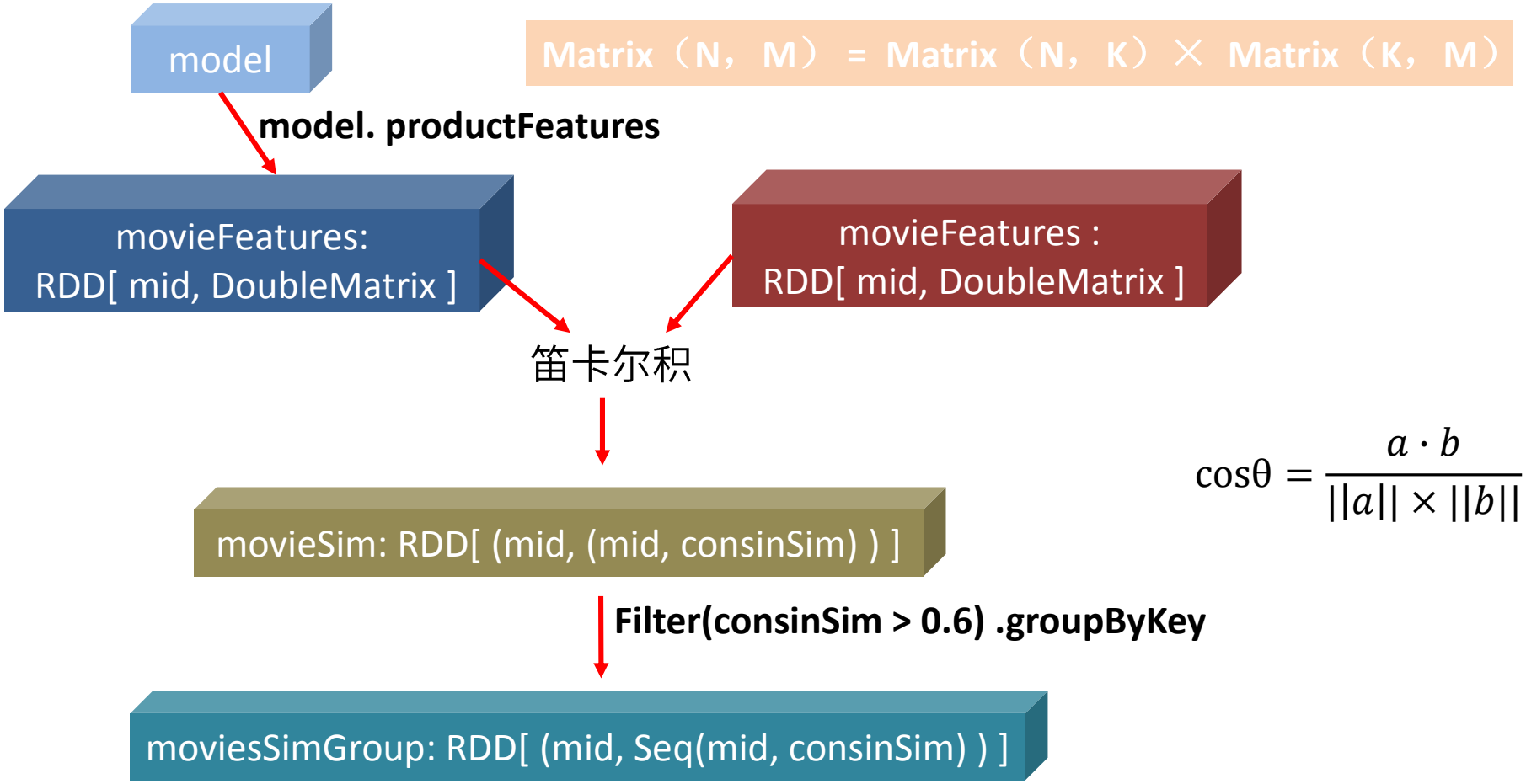
  可以通过均方根误差，来多次调整参数值，选择RMSE最小的一组参数值

- rank，iterations，lambda

计算用户推荐矩阵



计算电影相似度矩阵



$$\cos\theta = \frac{a \cdot b}{||a|| \times ||b||}$$

存储电影相似度矩阵

movieRecs: RDD[(uid, Seq[(mid, score)])]

spark-redis

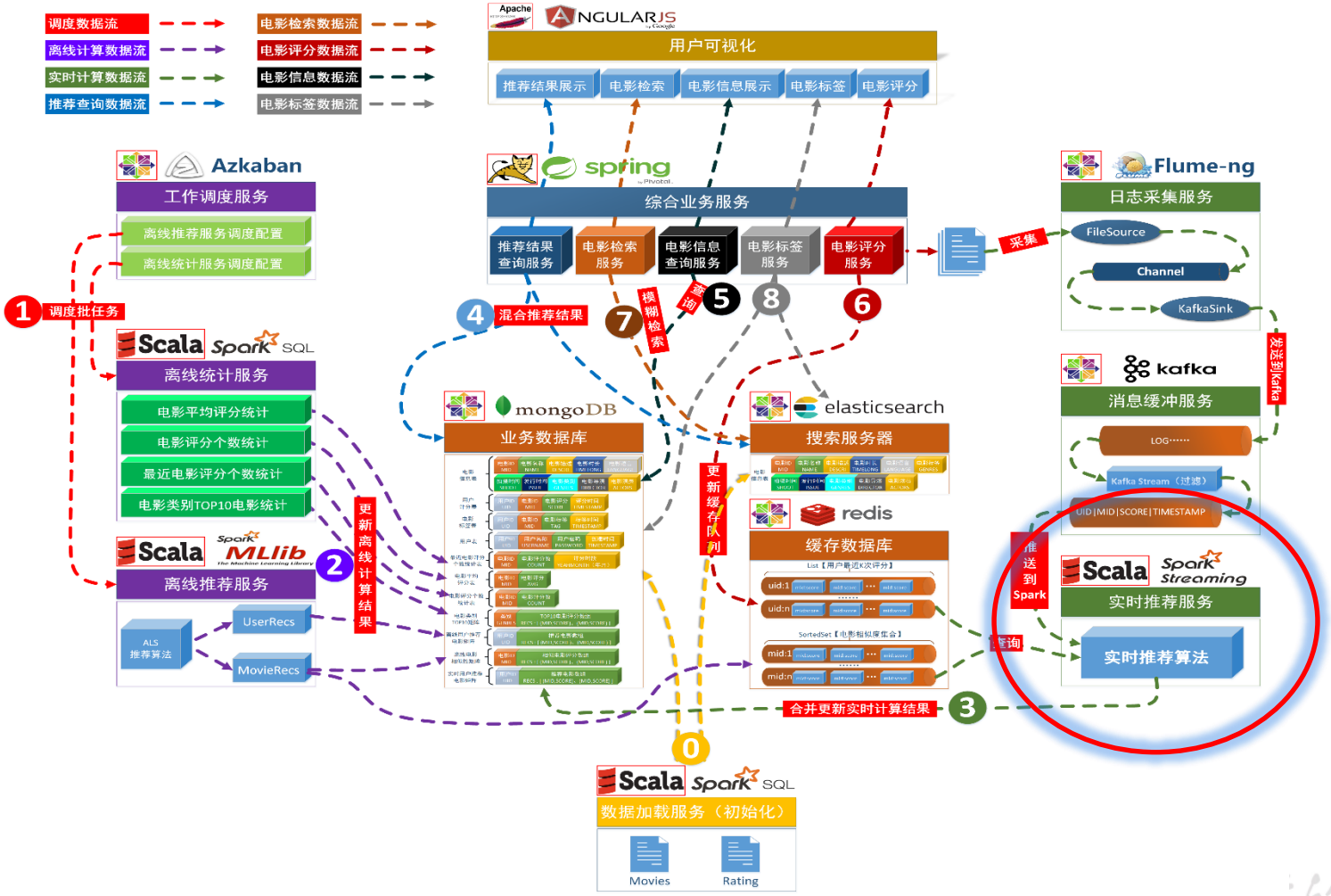
SparkSession.write

实时推荐模块

- 实时推荐架构
- 实时推荐优先级计算

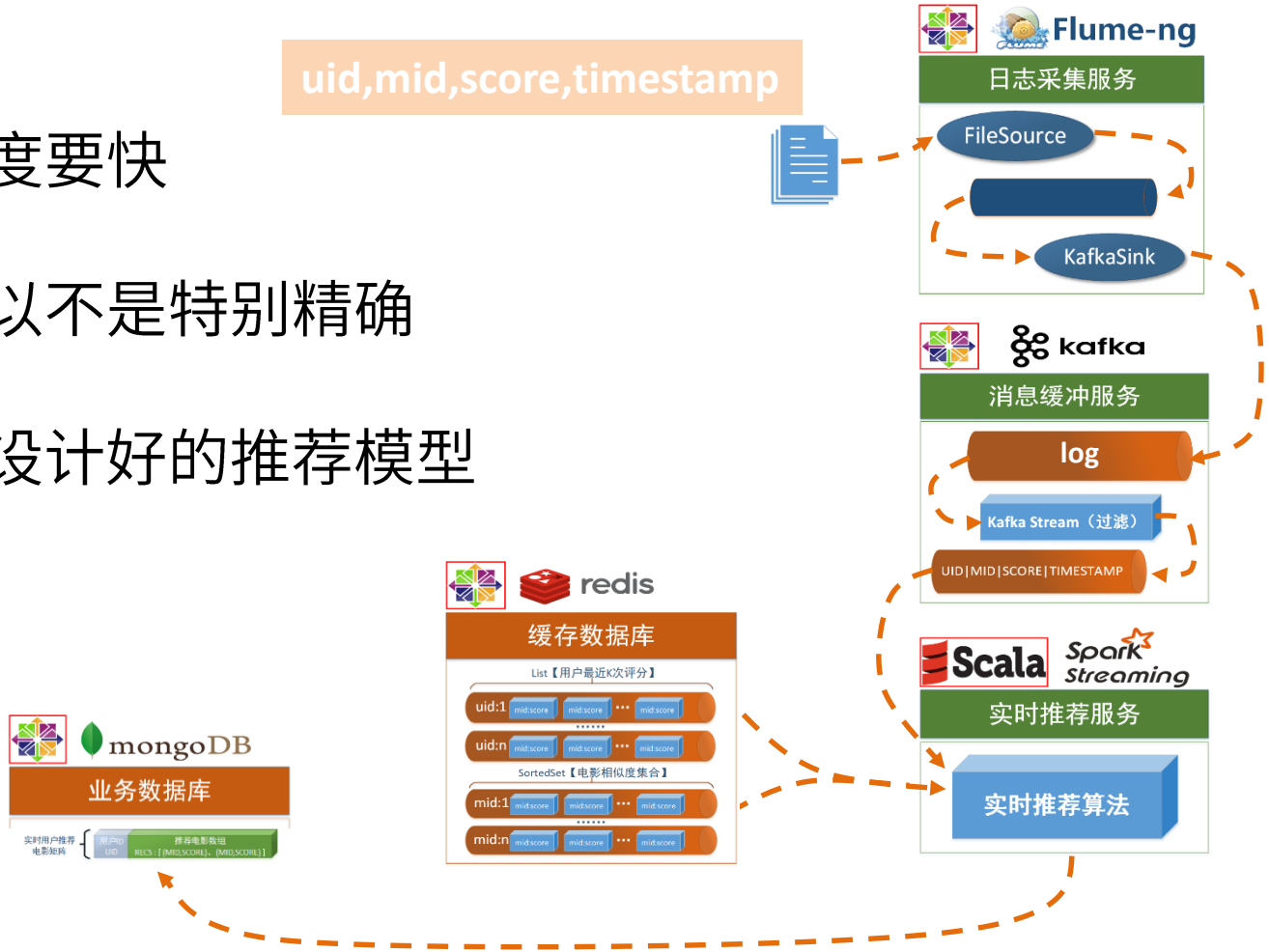
基于模型的实时推荐模块

电影推荐系统数据流图



基于模型的实时推荐模块

- 计算速度要快
- 结果可以不是特别精确
- 有预先设计好的推荐模型



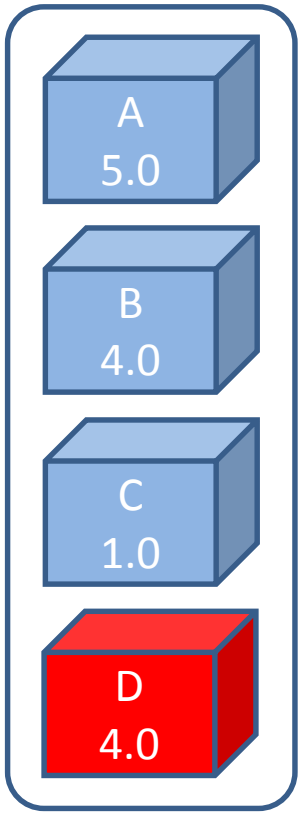
推荐优先级计算

基本原理：用户最近一段时间的口味是相似的

备选电影推荐优先级:

$$E_{uq} = \frac{\sum_{r \in RK} sim(q,r) \times R_r}{sim\_sum} + \lg \max\{incount,1\} - \lg \max\{recount,1\}$$

用户最近k次评分



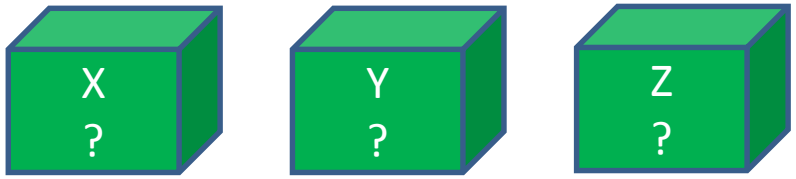
Sim(A, X)

Sim(B, X)

Sim(C, X)

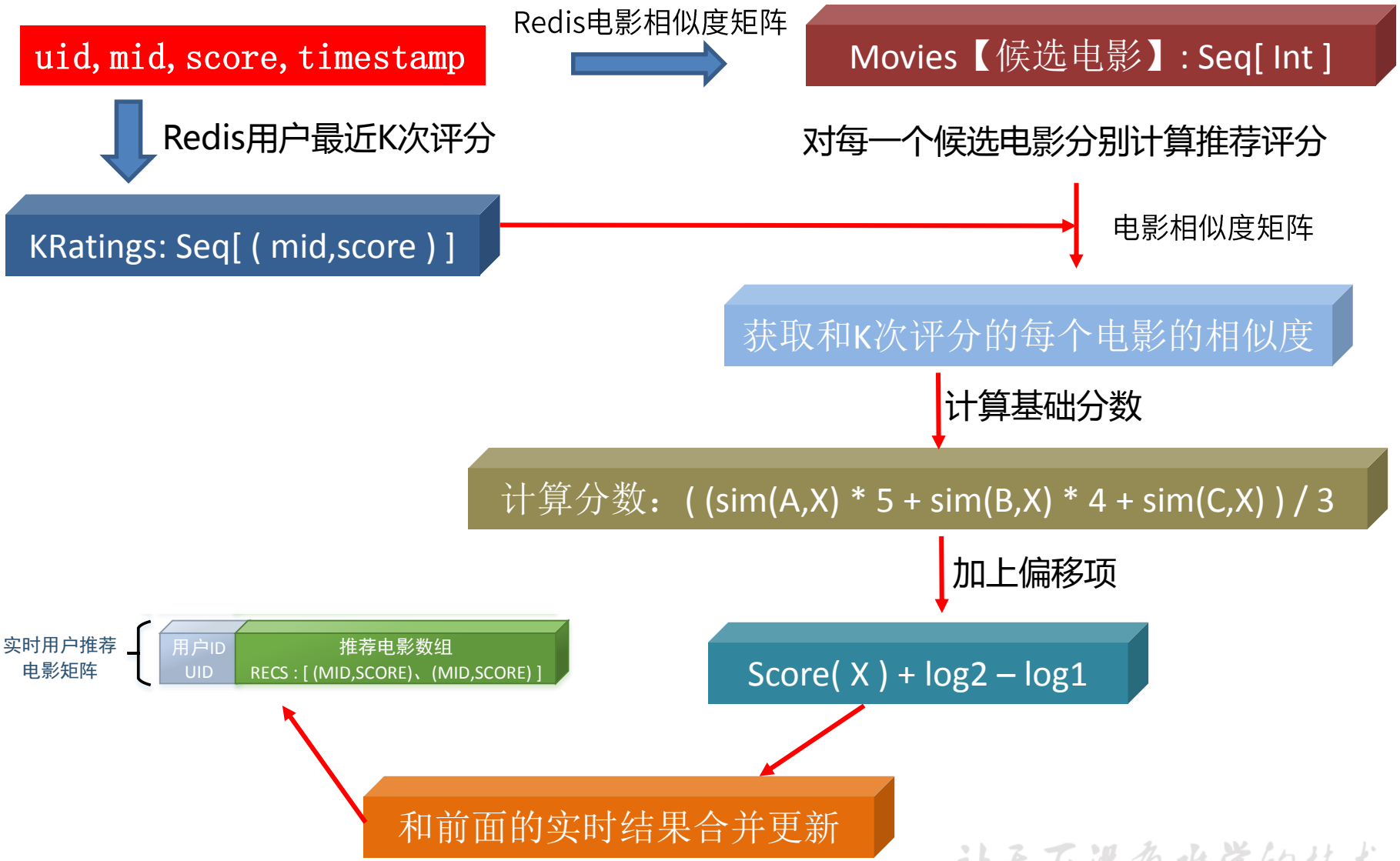
X的推荐优先级分数为：

? =(sim(A,X) * 5 + sim(B,X) * 4 + sim(C,X) * 1) / 3 + lg2 - lg1



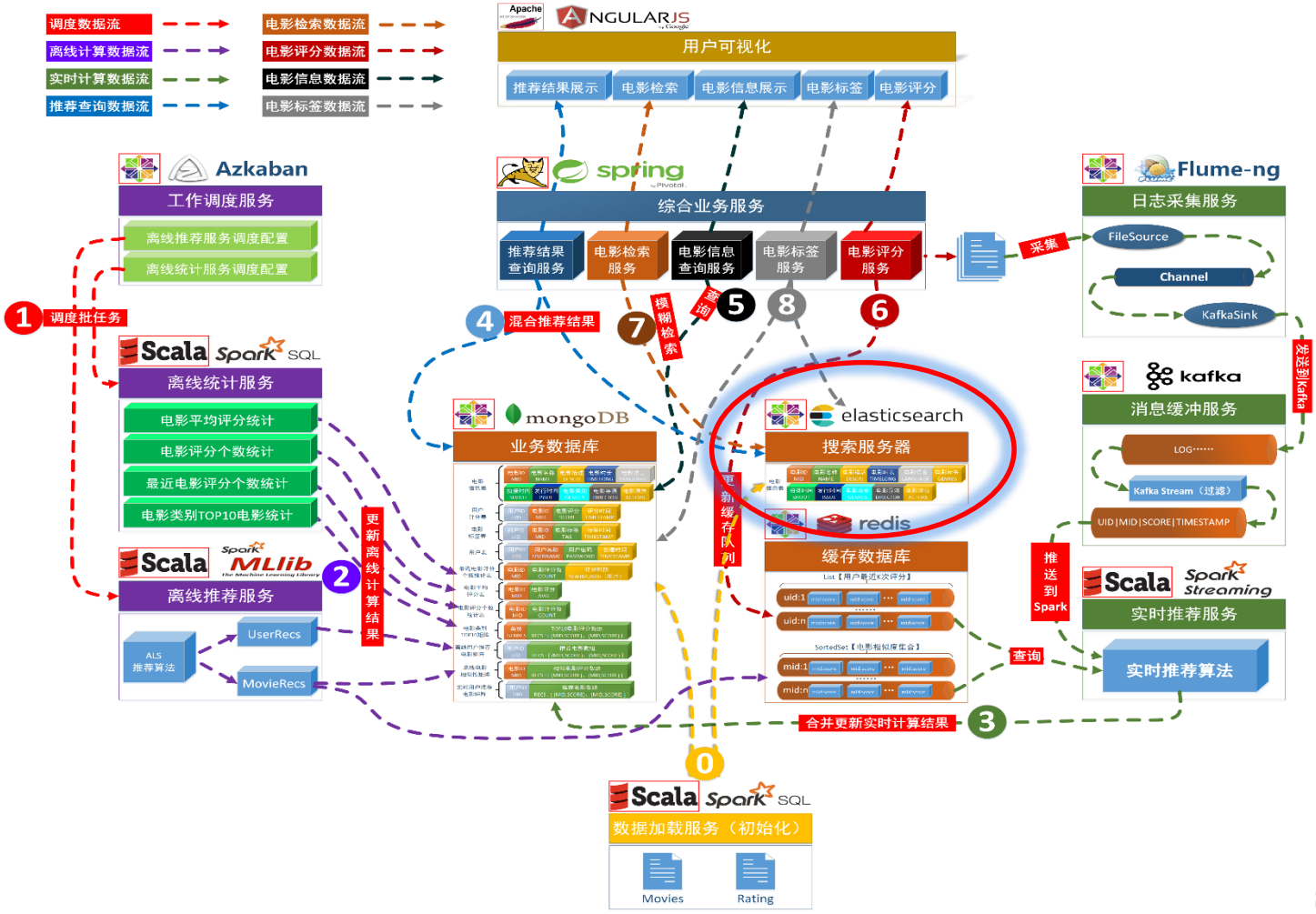
备选电影

推荐优先级计算

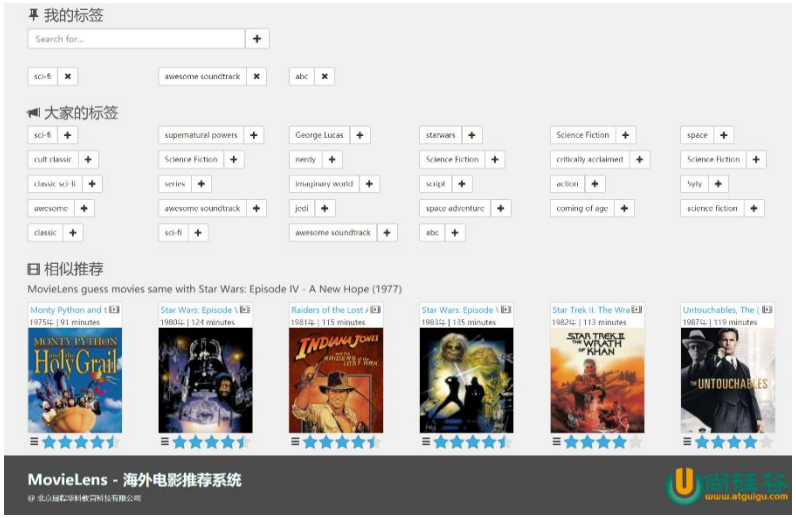


基于内容的推荐

电影推荐系统数据流图

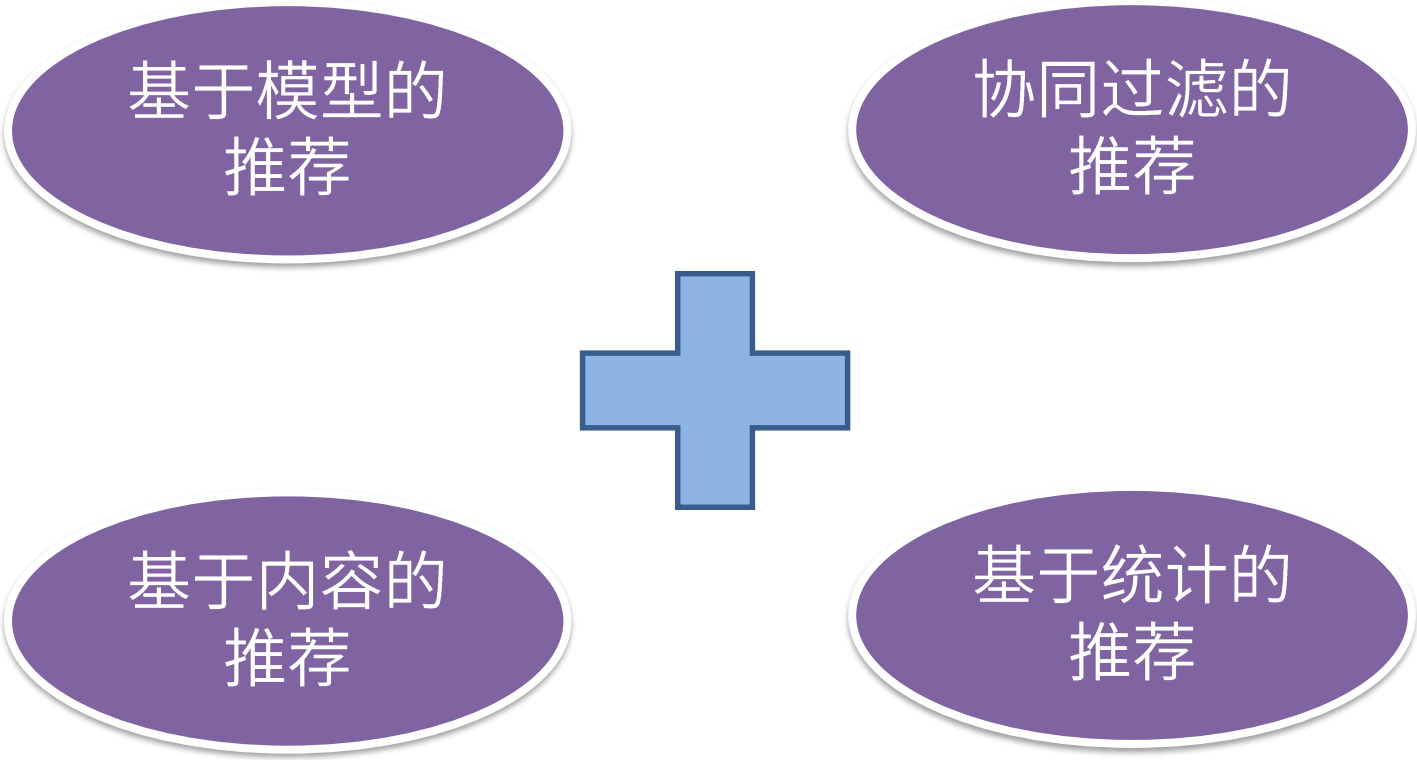


基于内容的推荐



- 电影 A 的相似电影？—— 有相同标签的电影
- Item-CF：根据标签提取电影 A 的内容特征，选取与 A 特征相似的电影
- 根据 UGC 的特征提取 —— TF-IDF

混合推荐 —— 分区混合



Q & A