COMP3029J
Software Architecture
Software Architecture Applications

DENG, YONGJIAN

Faculty of Computer Science, BJUT

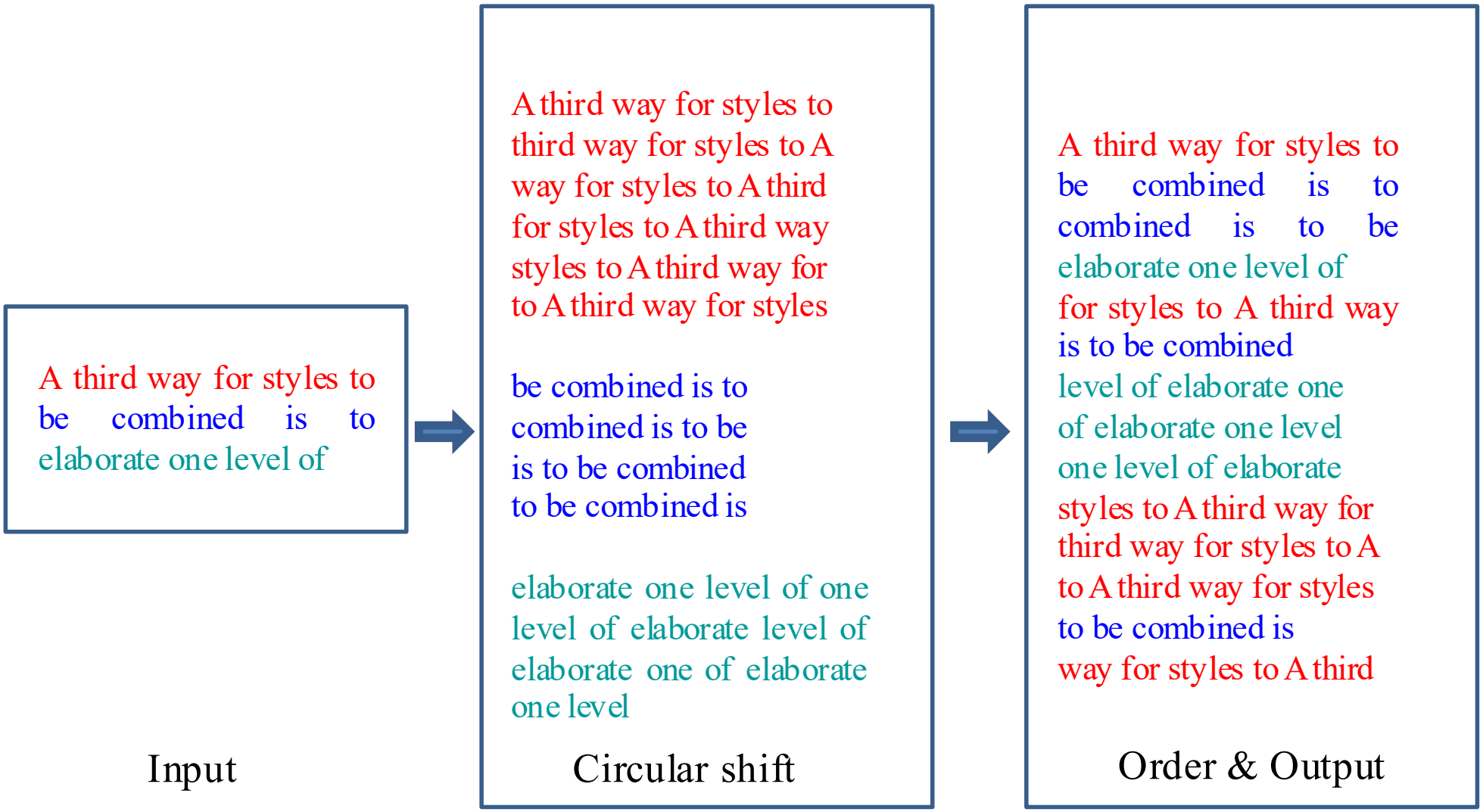
Data Mining & Security Lab (DMS Lab)

KWIC-Functional Requirements

In 1974, Parnas proposed the following problem, The KWIC (keyword- in-context) index system (KWIC索引系统)

1. **accepts an ordered set of lines (接受一些行)**
2. **each line is an ordered set of words (每行有若干词)**
3. **each word is an ordered set of characters (每个词由若干字符组成)**
4. **any line may be "circularly shifted" by repeatedly removing the first word and appending it at the end of the line (每行都可以循环移位：重复地把第一个词删除，然后接到行末)**
5. **The KWIC index system outputs a listing of all circular shifts of all lines in alphabetical order. (KWIC把所有行的各种移位情况按照字母表顺序输出)**

KWIC-A Case for KWIC



KWIC-Non-Functional Requirements

- **Modifiability (可修改性)**
  - ✓ **Change in Algorithm (算法的变化): batch vs. incremental**
  - ✓ **Change in Data Representation (数据表示方式的变化)**
    - ✓ **line storage, explicit vs. implicit shifts**
  - ✓ **Enhancement to system function (系统功能的可扩展性)**
    - ✓ **e.g., eliminate lines starting with trivial words**
    - ✓ **e.g., input lines from a database or from UI**
    - ✓ **e.g., delete, modify or add to lines from the original shifted lines**

KWIC-Non-Functional Requirements

- **Performance (性能): Both space and time (时空复杂性)**
- **Reusability (系统构件的可复用性)**
  - ✓ **To what extent can the components serve as reusable entities.**

KWIC-Main Program/Subroutine with Shared Data

- 采用“主程序-子过程”风格、对系统进行功能分解，是最自然的想法，也是“面向过程的编程”的主要思路。
  - **Decomposes the problem according to the four basic functions performed: input, shift, alphabetize, and output.**
  - **These computational components are coordinated as subroutines by a main program that sequences through them in turn.**
  - **Data is communicated between the components through shared storage with an unconstrained read-write protocol.**

KWIC-Main Program/Subroutine with Shared Data

Advantages

- Data can be represented efficiently, since computations can share the same storage. (模块之间的数据共享)
- Distinct computational aspects are isolated in different modules (不同的计算功能被隔离在不同的模块中)

Disadvantages

- A change in data storage format will affect almost all of the modules.
- Changes in the overall processing algorithm and enhancements to system function are not easily accommodated.
- This decomposition is not particularly supportive of reuse.

KWIC-Abstract Data Types

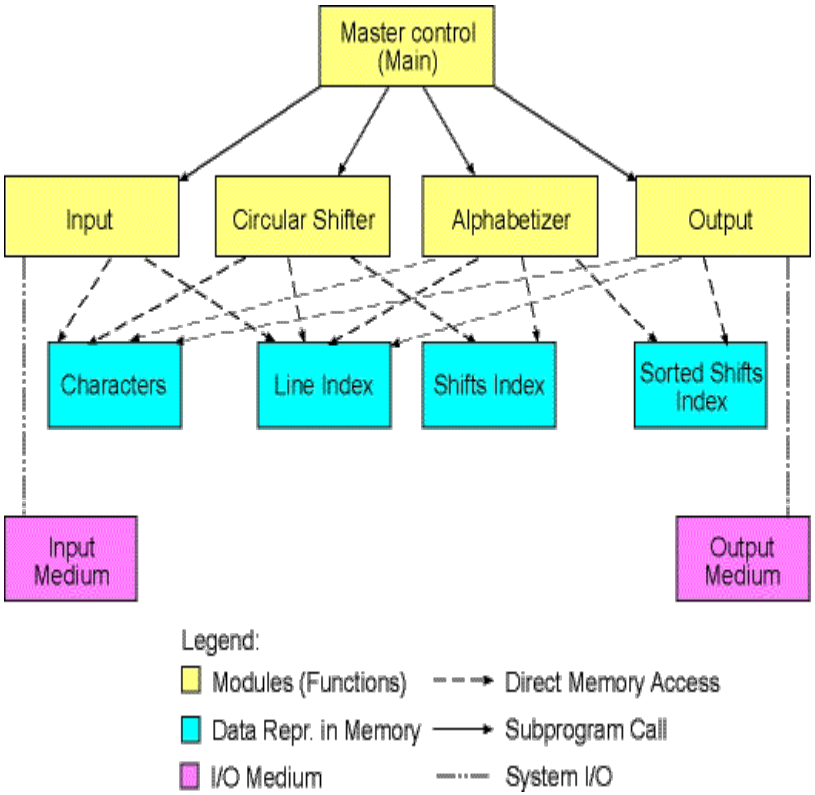
Advantages

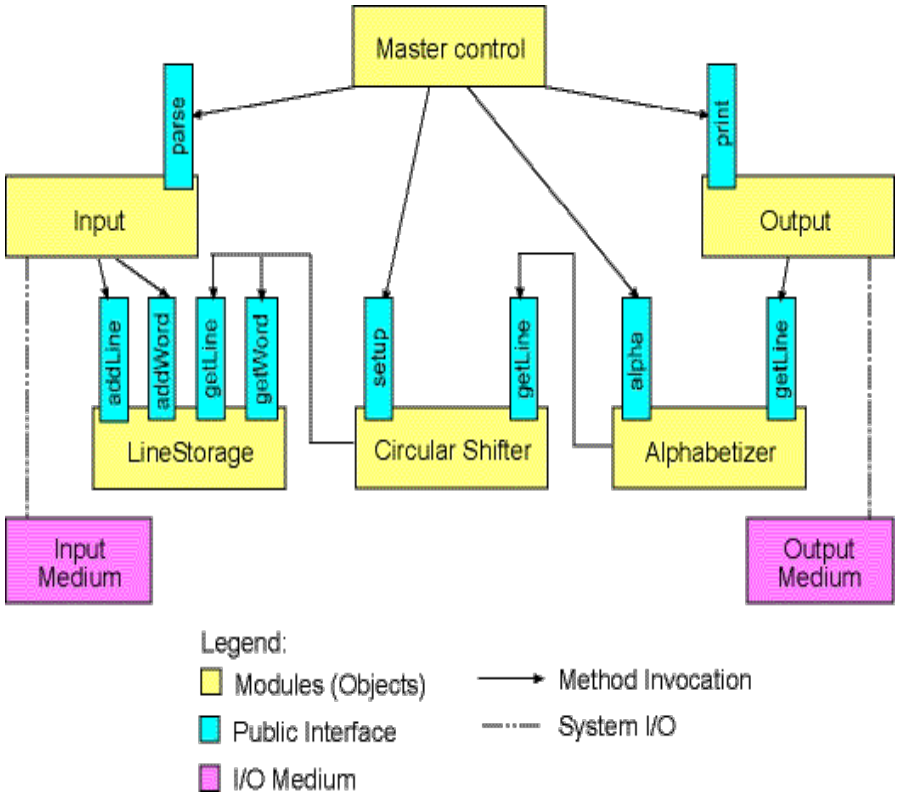
- ✓ Both algorithms and data representations can be changed in individual modules without affecting others.
- ✓ Reuse is better supported than in the first solution because modules make fewer assumptions about the others with which they interact.

Disadvantages

- ✓ The solution is not particularly well-suited to enhancements.
- ✓ For adding new functions to the system, the implementor must either modify the existing modules or add new modules that lead to performance penalties.

KWIC-Abstract Data Types





从架构观点看，OO优于Main Program/Subroutine

依赖减少 易于理解 易于维护 易于复用

KWIC-Pipes and Filters

Uses a pipeline solution. (使用管道-过滤器风格)

- ✓ **There are four filters: input, shift, alphabetize, and output.**
- ✓ **Each filter processes the data and sends it to the next filter.**
- ✓ **Control is distributed: each filter can run whenever it has data on which to compute.**
- ✓ **Data sharing between filters is strictly limited to that transmitted on pipes.**

KWIC-Pipes and Filters

Advantages of Pipes and Filters

- It maintains the intuitive flow of processing. (过程流非常直观)
- It supports reuse (支持复用!!)
- It supports ease of modification, since filters are logically independent of other filters. (容易修改!!)
  - ✓ Each filter can function in isolation (provided upstream filters produce data in the form it expects).
  - ✓ New functions are easily added to the system by inserting filters at the appropriate point in the processing sequence.

北京工業大學
BEIJING UNIVERSITY OF TECHNOLOGY

KWIC-Pipes and Filters

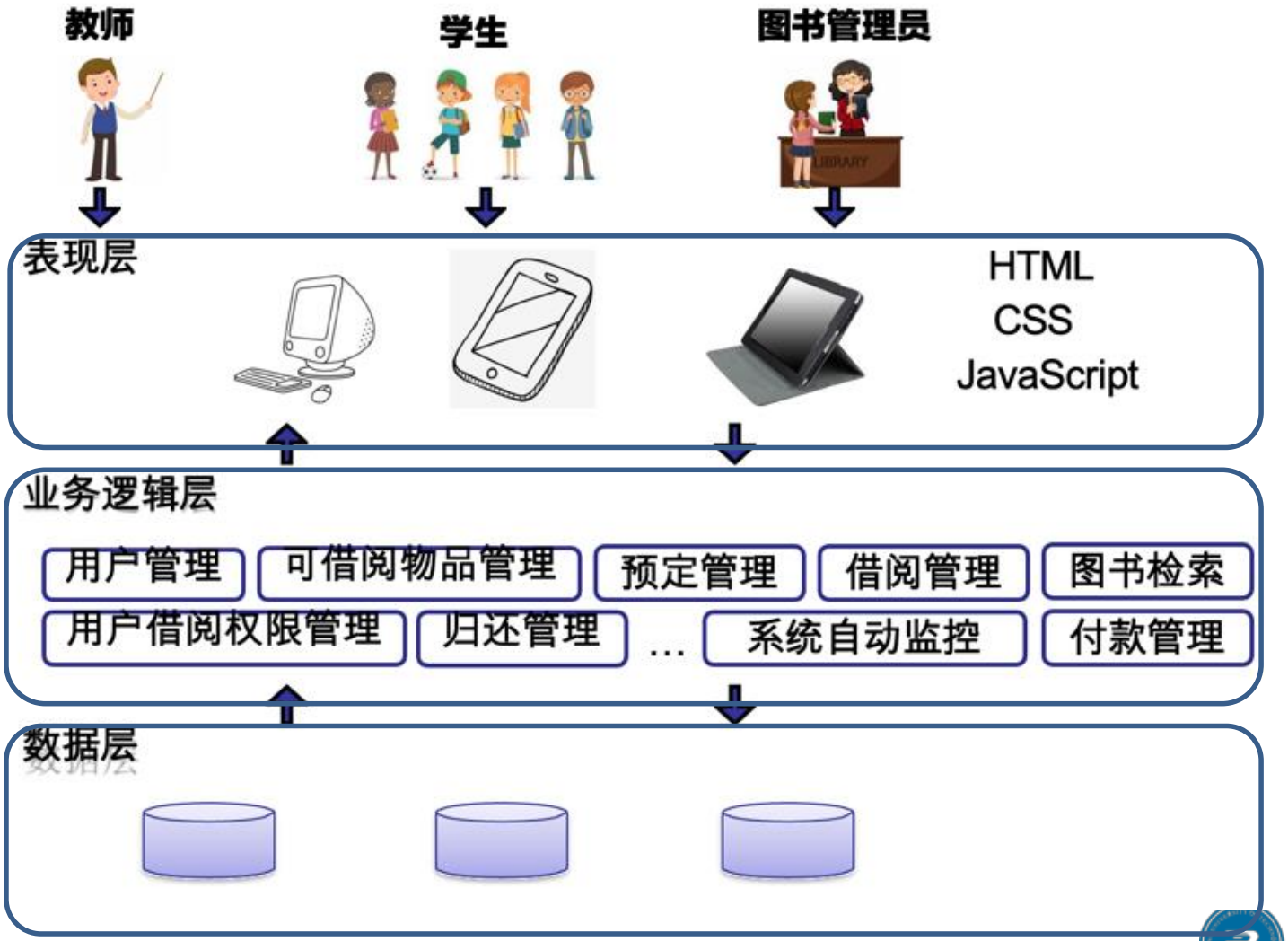
Disadvantages of Pipes and Filters

- It is virtually impossible to modify the design to support an interactive system. (无法支持交互式系统，局限性较大!)
  - ✓ For example, in order to delete a line, there would have to be some persistent shared storage, violating a basic tenet of this approach.
- The solution is inefficient in terms of its use of space, since each filter must copy all of the data to its output ports. (空间复杂性高)

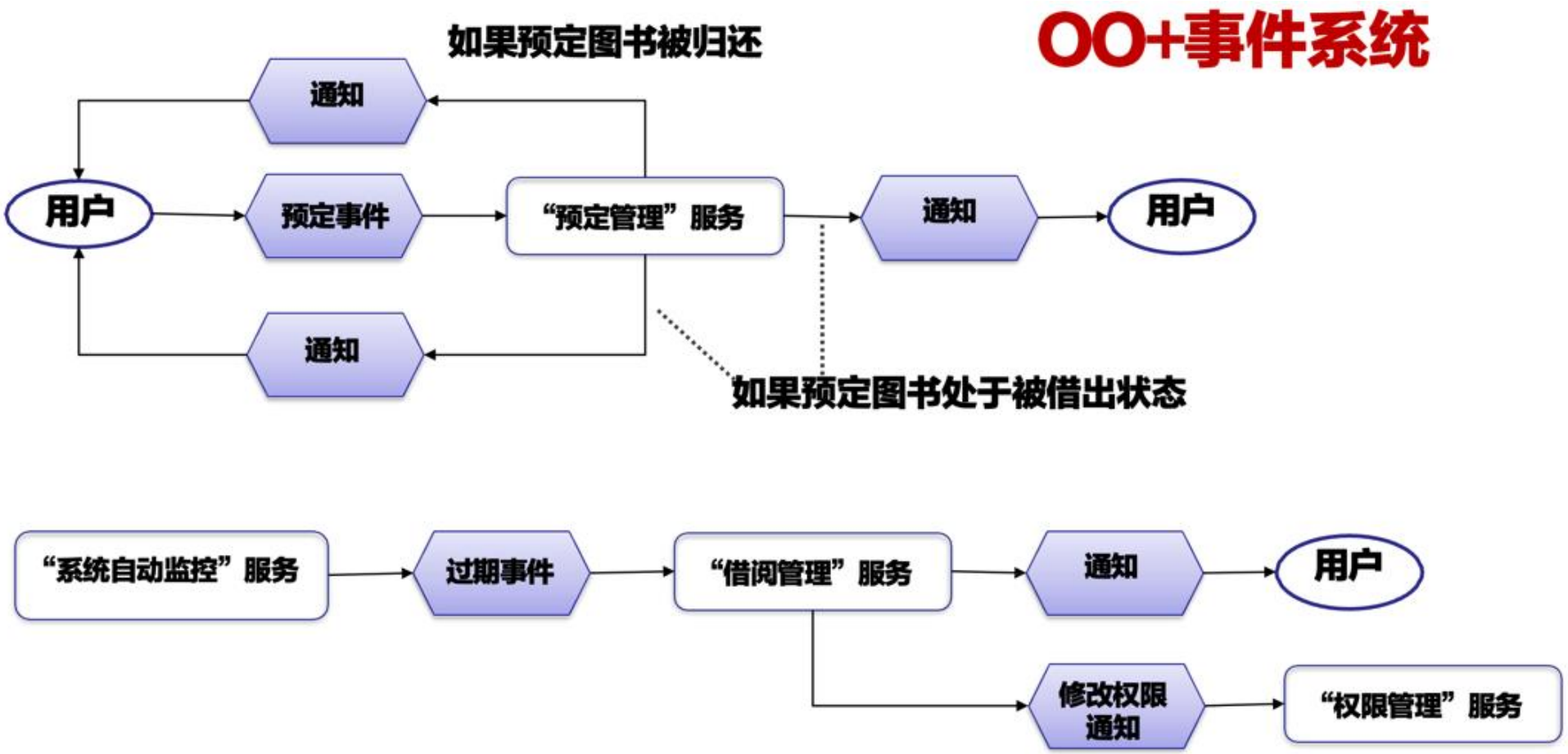
Library-Requirements

该系统向师生提供图书借阅服务；有两个图书存储地点(即2个分馆)，图书索引应能查询所有的分馆的信息；分馆之间不支持馆际借阅；学生一次可借阅16本书，1个月内归还；教师一次可借阅任何数量的书，2个月内归还；学生和教师均可以预定图书，如预定图书处于可借阅状态，预定有效期为1天；如果预定的图书正处于被借出状态，系统需要向借阅者发出一封 email进行提醒；如果借阅的图书过期2周、6周、10周，则向借阅者发出email进行提醒；当过期未还时，该借阅者无法再借阅其他图书，而且当过期超过1个月后，每本书每天罚款1元；如果有未付清的罚款，借阅者也无法再借阅其他图书；除了图书之外，还提供杂志的借阅服务，借阅规则同图书相同。

Library-Architecture Design



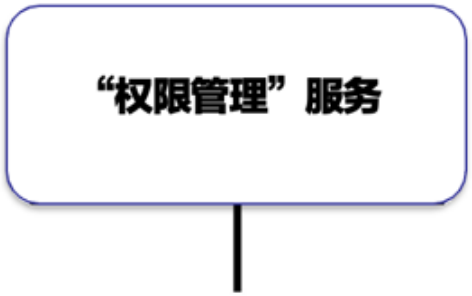
Library-Architecture Design



Library-Architecture Design

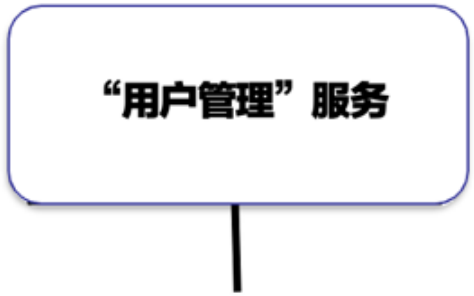
该系统为某连锁书店的图书借阅系统，向书店会员提供图书借阅服务；该市共有分布在不同地点的5家连锁书店，图书索引应能查询所有分店的信息；分店之间不支持跨店借阅与归还；普通会员一次可借阅10本书，3个月内归还；高级会员一次可借阅任何数量的书，6个月内归还；所有会员均可以预定图书，如预定图书处于可借阅状态，预定有效期为1周；如果预定的图书正处于被借出状态，系统会向借阅者发出一封email进行提醒；如果借阅的图书过期，系统将从过期之日起，每天向借阅者发出email进行提醒；当过期未还时，该借阅者无法再从任何分店借阅其他图书，而且当过期超过1周后，每本书每天罚款1元，从会员押金中自动扣除；如果有未付清的罚款，借阅者也无法再借阅其他图书；此外，还为高级会员提供杂志的借阅服务。

“基于规则的系统”风格



规则定义：

IF 用户类别=“…” AND 项目类别=“…”
THEN
每次借阅的最大数目=“…”
最多借阅天数=“…”
超期后是否提醒=“Y/N”
超期后是否可再借阅=“Y/N”
超期是否罚款=“Y/N”
超期罚款标准=“…”
是否可预定=“Y/N”
预定请求的过期期限=“…”
预定是否付费=“Y/N”
预定的付费标准=“…”



规则定义：

IF 用户类别=“…”
THEN
需要交费成为会员=“Y/N”
费用标准=“…”
会员有效期=“…”

Library-Product Line

A software product line is a set of software-intensive systems that share a common, managed feature set satisfying a particular market segment's specific needs or mission and that are developed from a common set of core assets in a prescribed way. (软件产品线是一组软件系统，共享一组通用的特征集合，通过使用一组预先开发的/通用的核心资产来满足不同产品的研发需求)