隐私计算线上慕课【第16讲】

隐语框架概览及设计思想

王磊 | 蚂蚁集团隐私智能计算部总经理、隐语框架负责人

目 录

Contents

动机

01

国家政策趋势

国家政策

2020.03
国务院《关于构建更加完善的要素市场化配置体制机制的意见》

建立健全城乡统一的建设用地市场；
深化户籍制度改革；
加快培育数据要素市场

2020.05
国务院《关于新时代加快完善社会主义市场经济体制的意见》

建立健全统一开放的要素市场；
推进要素价格市场化改革；
创新要素市场化配置方式

2022.01
国务院《要素市场化配置综合改革试点总体方案》

建立健全数据流通交易规则；
探索“原始数据不出域、数据可用不可见”的交易范式

工作目标

2021年
启动要素市场化配置综合改革试点工作

2022年
完成试点地区布局、实施方案编制报批工作

2023年
试点工作取得阶段性成效，力争在土地、劳动力、资本、技术等要素市场化配置关键环节上实现重要突破，在数据要素市场化配置基础制度建设探索上取得积极进展

2025年
基本完成试点任务，要素市场化配置改革取得标志性成果

隐私计算面临问题

问题	架构需求	
技术路线众多，架构完全不同，切换成本高，多技术组合困难	**完备性**	在一套架构下支持主流隐私计算方案且未来可扩展
底层技术与上层应用耦合严重，底层技术升级，牵一发动全身	**透明性**	底层技术和上层应用解耦，底层技术迭代对上层应用透明
开发者需要同时了解AI算法和安全协议，开发门槛很高	**开放性**	良好的架构分层，不同专业开发者只需要专注自身领域的研发
业务集成成本高，框架二开能力弱	**易用性**	良好的接口设计，合理的模块划分，方便集成和二开
规模化生产需要的各种高可用能力缺失	**可用性**	具备灰度、回滚、弹性扩容等规模化生产能力

整体架构和设计思路

02

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备
  - MPC设备
  - HE设备
  - TEE设备
  - TECC设备
- 明文计算设备
  - Python 解释器
  - SQL 执行环境
- 隐私保护原语
  - 差分隐私
  - 脱敏

→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

隐语整体架构


5
用户界面
可视化操作界面
开放编程接口
业务研发使用友好
平台开发接入成本低
4
AI & BI
隐私算法
多方安全计算
联邦学习
可信执行环境
隐私保护算法使用友好
提升算法开发效率
3
明密文
混合调度
设备计算图
分布式调度引擎
调度/编译器开放合作
共建明密文混合编程能力
1
明密文
计算设备与原语
密文计算设备
MPC设备
HE设备
TEE设备
TECC设备
明文计算设备
Python
解释器
SQL
执行环境
2
隐私保护原语
差分隐私
脱敏
密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语
6
资源管理
数据管理
计算管理
网络管理
业务交付运维友好
大规模高可用，部署运维成本低

密文计算设备

03

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备
  - MPC设备
  - HE设备
  - TEE设备
  - TECC设备
- 明文计算设备
  - Python 解释器
  - SQL 执行环境
- 隐私保护原语
  - 差分隐私
  - 脱敏

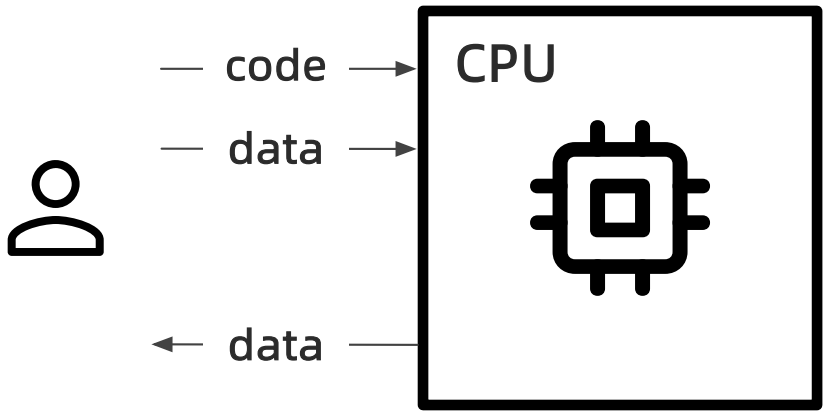
→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

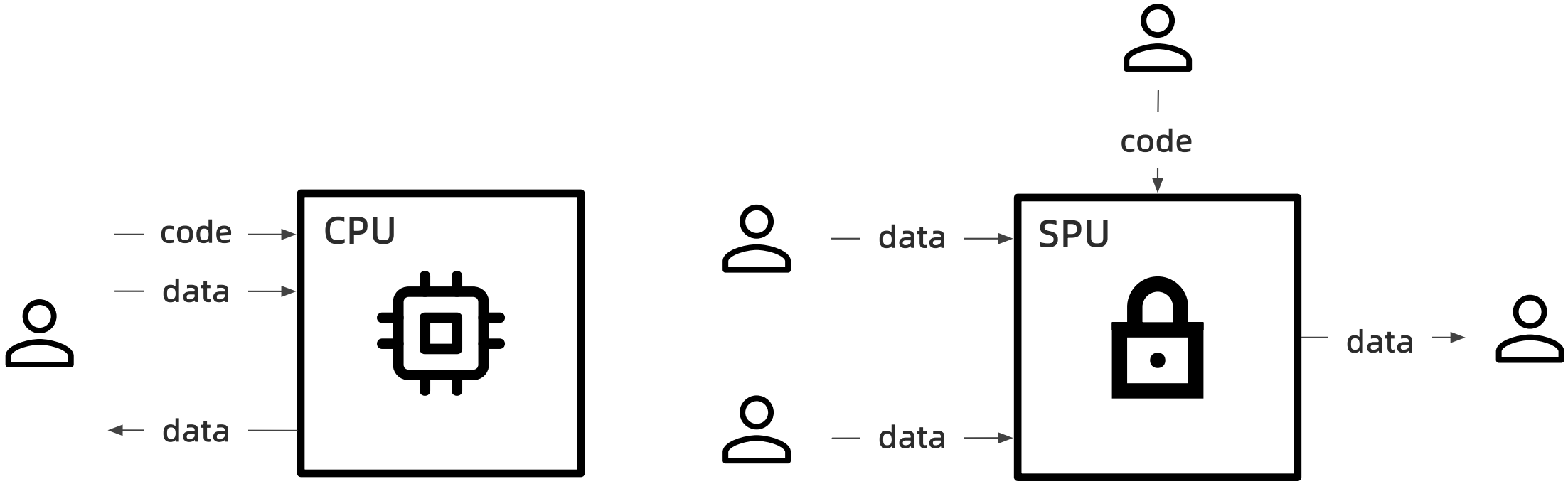
- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

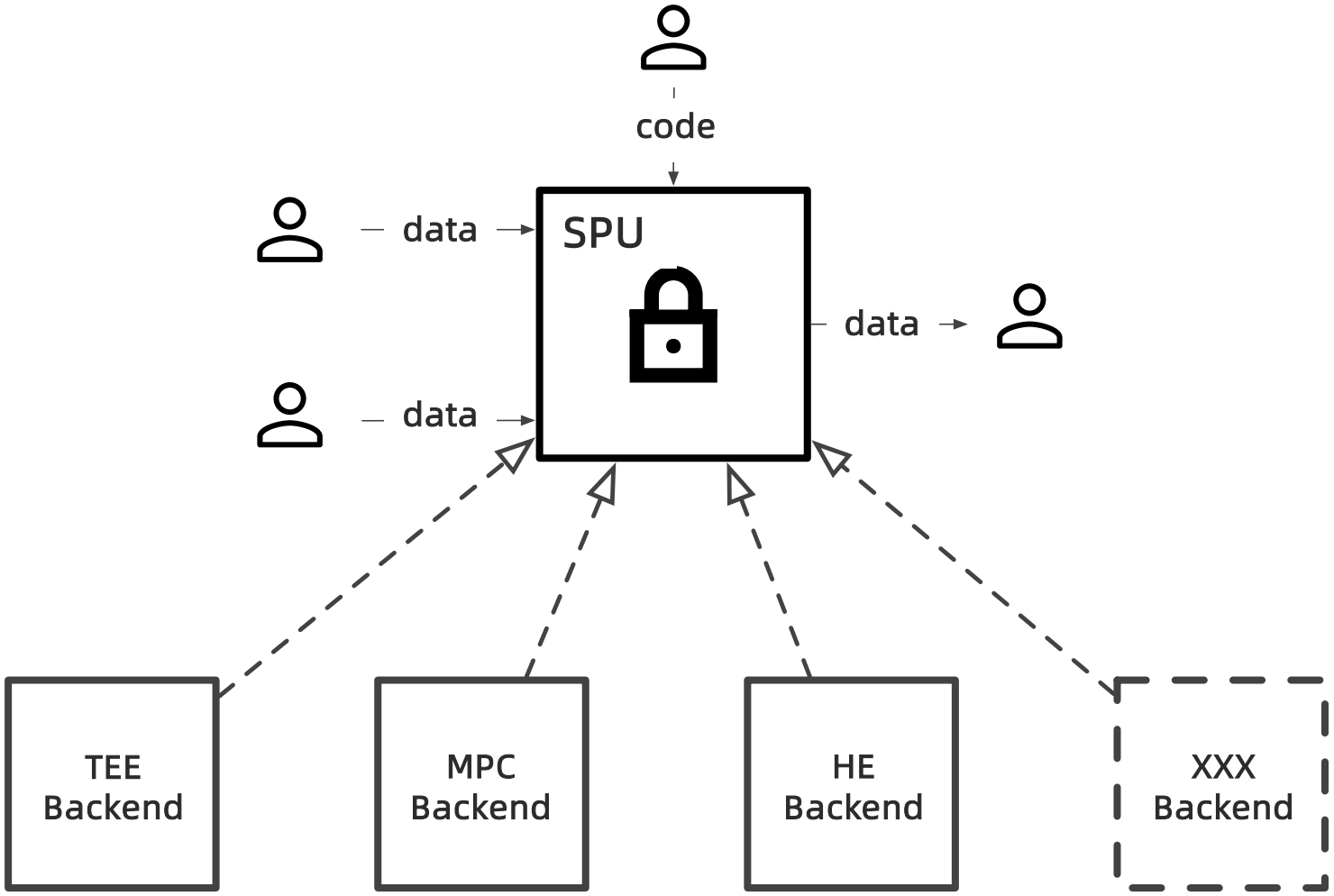
什么是密文计算设备-SPU



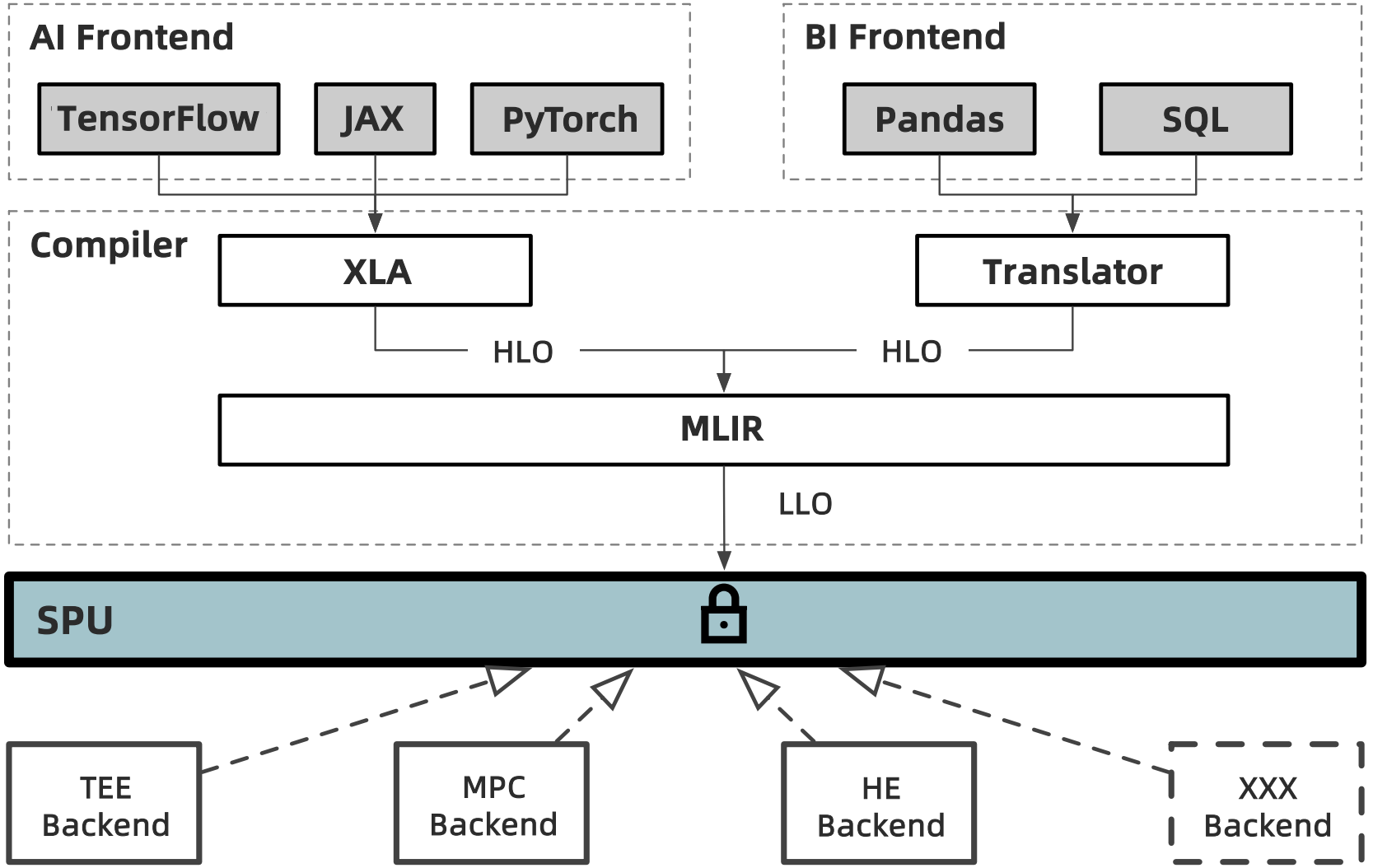
什么是密文计算设备-SPU



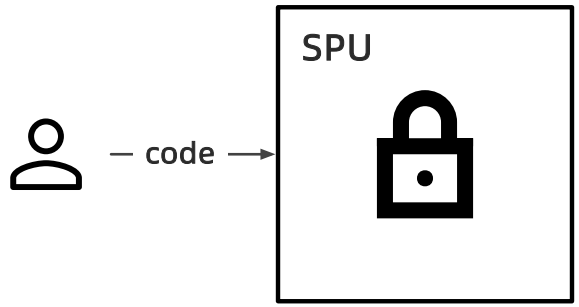
SPU的多后端


code
data
data
SPU
data
TEE
Backend
MPC
Backend
HE
Backend
XXX
Backend

SPU抽象的优势-复用经典技术

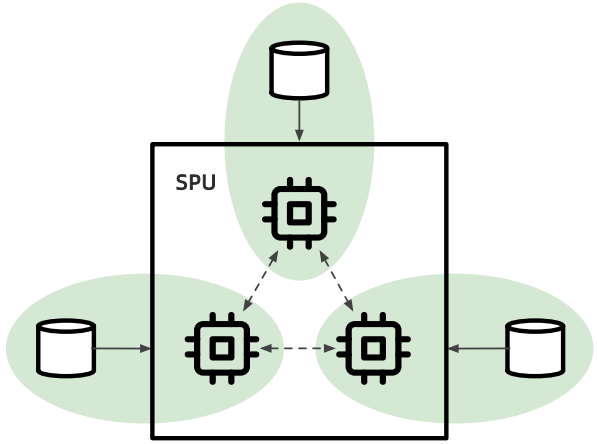


SPU抽象的优势-开发部署分离

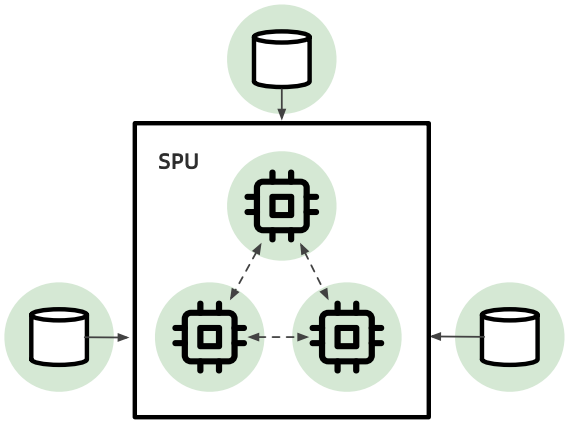


Development

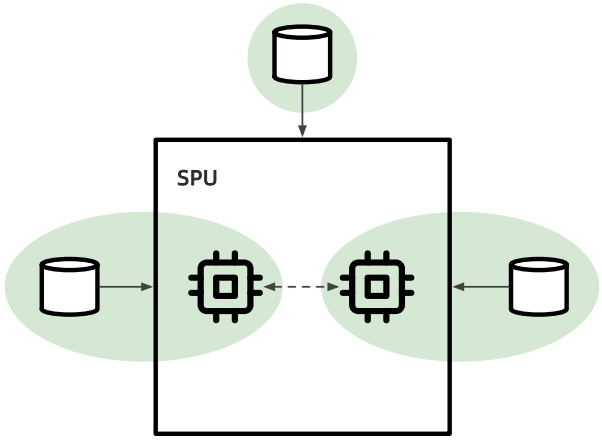
Deployment



Colocated



Outsourcing



Hybird

隐私保护原语

04

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备
  - MPC设备
  - HE设备
  - TEE设备
  - TECC设备
- 明文计算设备
  - Python 解释器
  - SQL 执行环境
- 隐私保护原语
  - 差分隐私
  - 脱敏

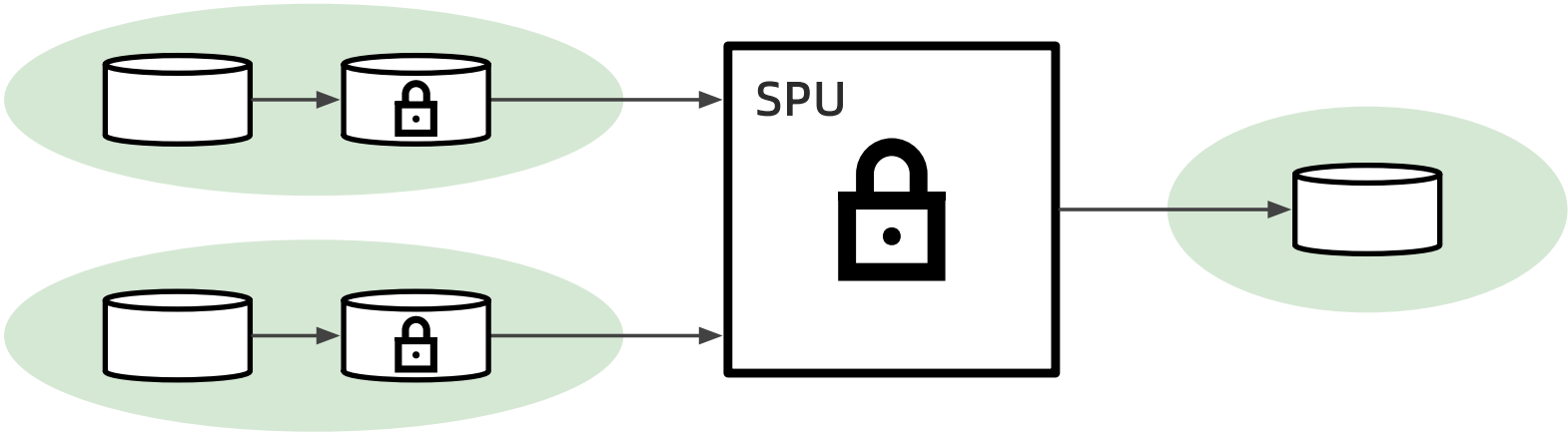
→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

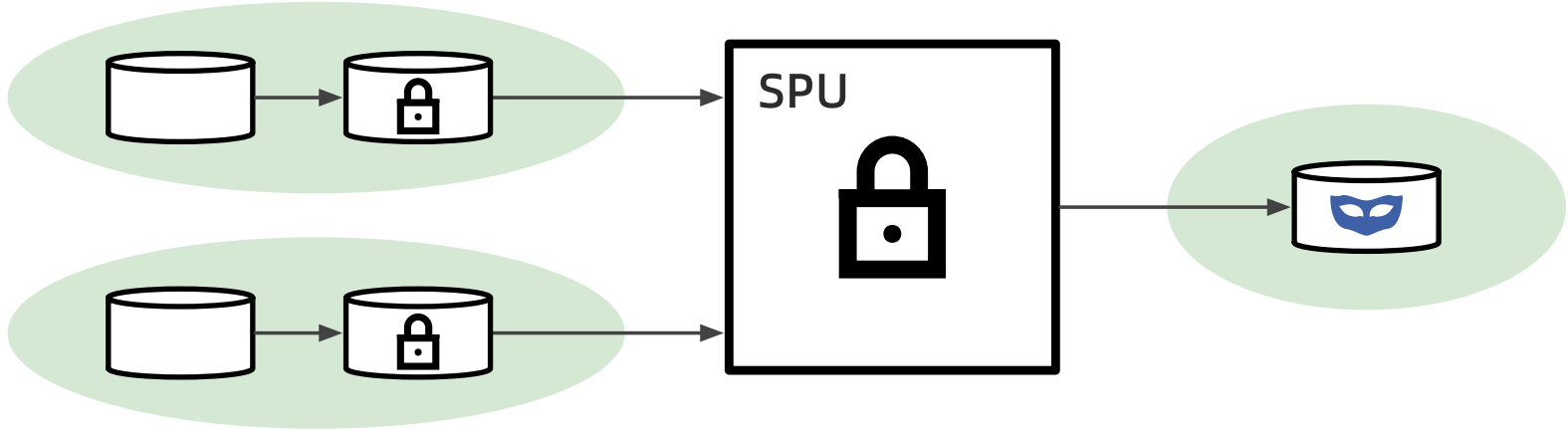
- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

隐私保护和密文计算的区别

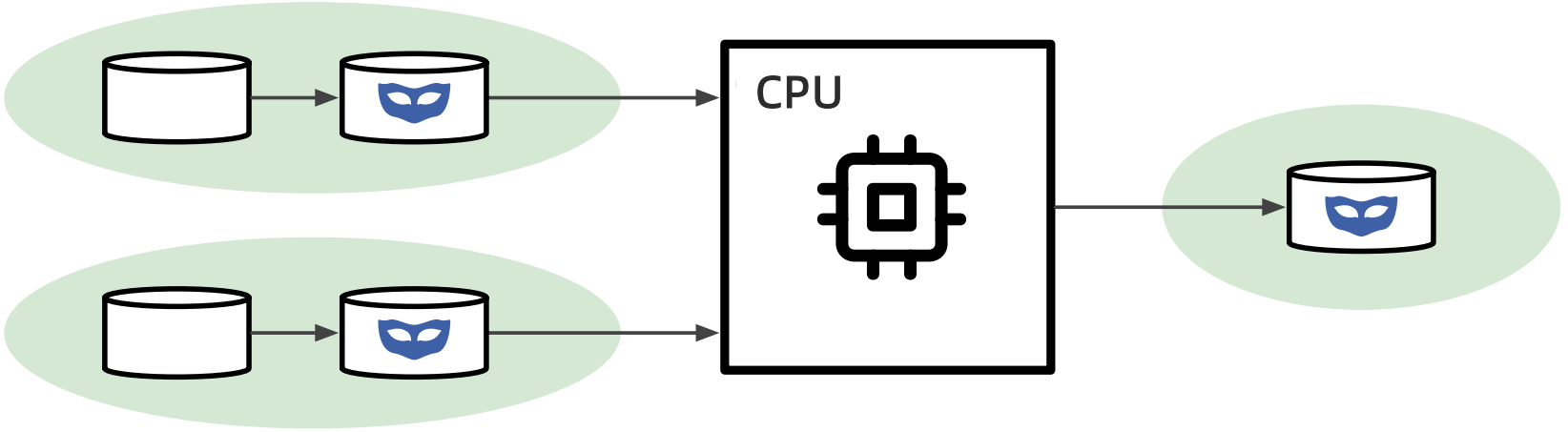


隐私保护和密文计算的区别



差分隐私

脱敏算法



明密文混合计算

05

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备
  - MPC设备
  - HE设备
  - TEE设备
  - TECC设备
- 明文计算设备
  - Python 解释器
  - SQL 执行环境
- 隐私保护原语
  - 差分隐私
  - 脱敏

→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

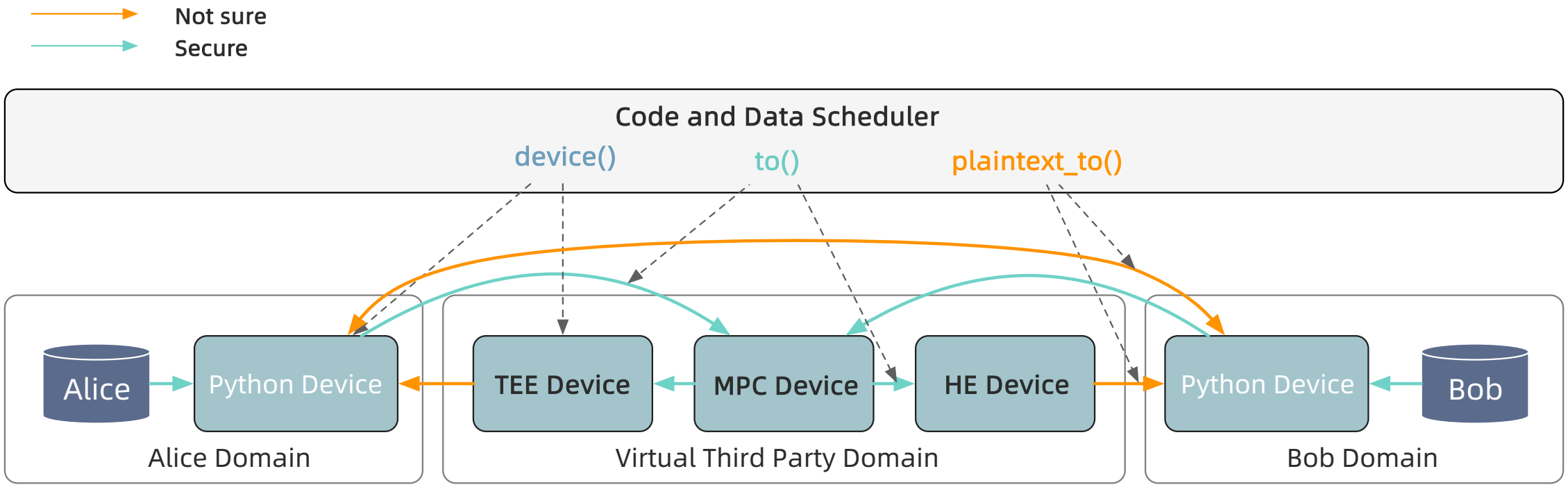
- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

计算抽象

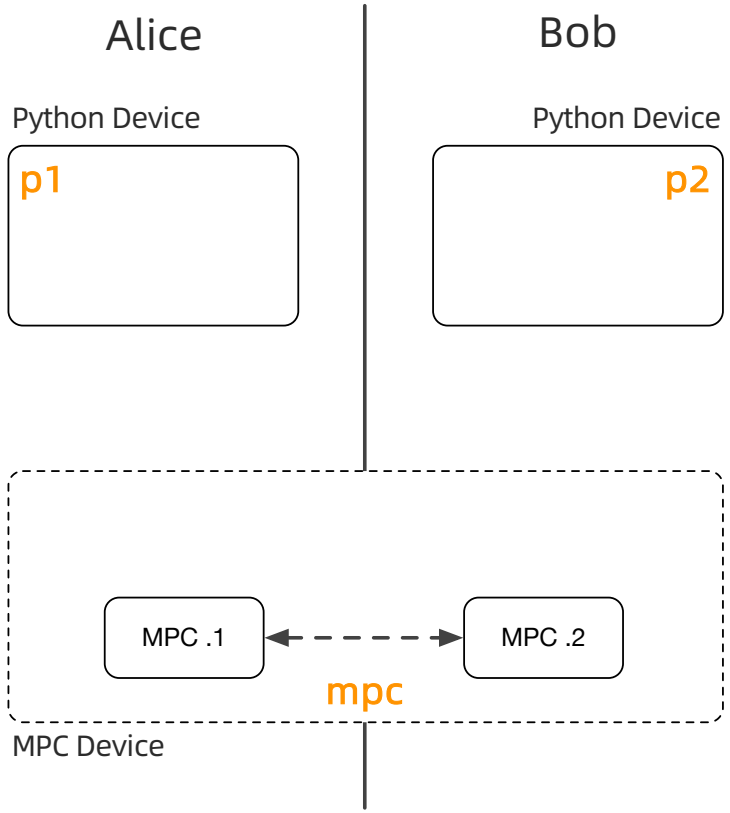
Computation = Device + Data Flow

SecretFlow



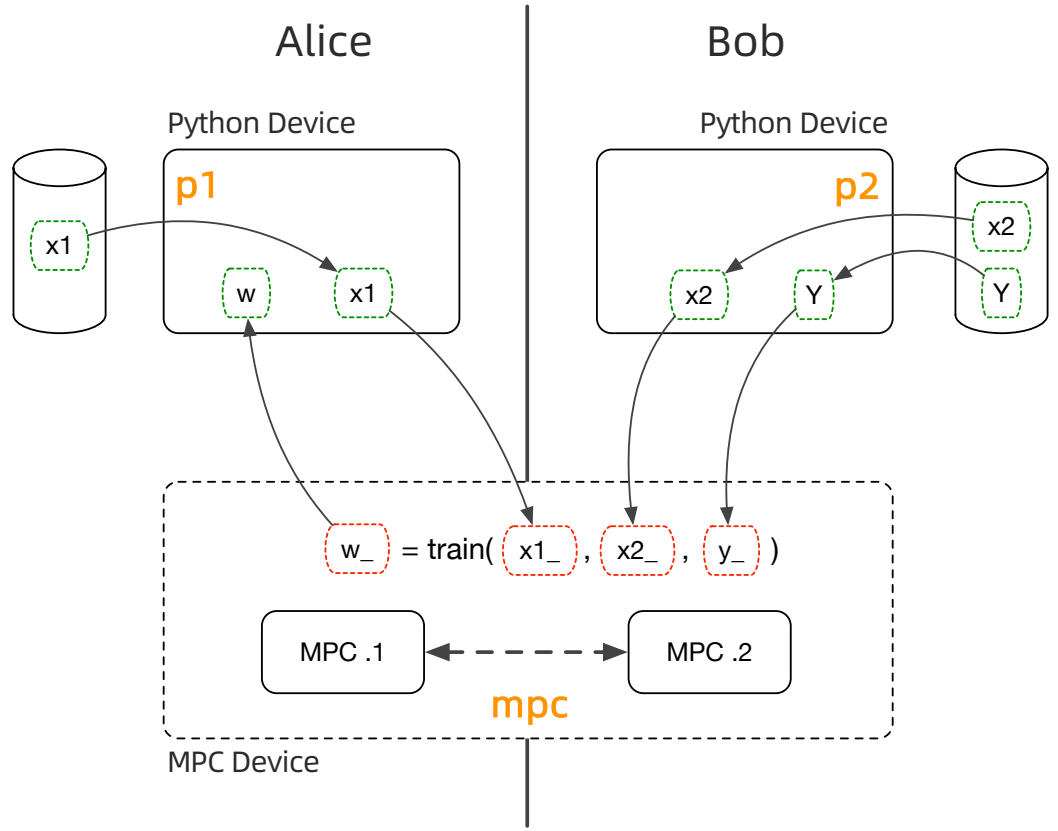
明密文混合编程-MPC DEMO

```
domain: [“Alice”, “Bob”]

device:
  - p1:
    type: PYU
    domain: “Alice”
  - p2:
    type: PYU
    domain: “Bob”
  - mpc:
    type: MPC
    domain: [“Alice”, “Bob”]
```



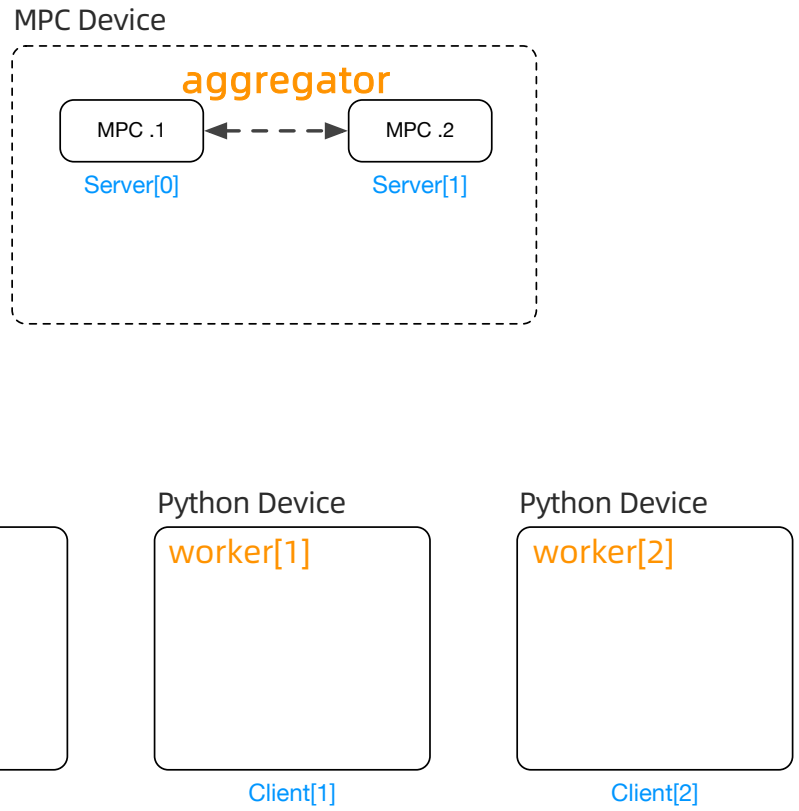
明密文混合编程-MPC DEMO

```
def load_data(path):
    XXXX # load data from local path
    return data

def train(x1, x2, y):
    x = jax.numpy.concatenate((x1, x2), axis=1)
    lr = jax_utils.logistic_regression()
    return lr.fit_auto_grad(x, y)

# initialize device
init_device(device_config) #p1, p2, mpc

# load data
x1 = device(p1)(load_data)("/data/x1.csv")
x2 = device(p2)(load_data)("/data/x2.csv")
y = device(p2)(load_data)("/data/y.csv")

# train
x1_, x2_, y_ = to(x1, mpc), to(x2, mpc), to(y, mpc)
w_ = device(mpc)(train)(x1_, x2_, y_)

# reveal
w = plaintext_to(w_, p1)
```



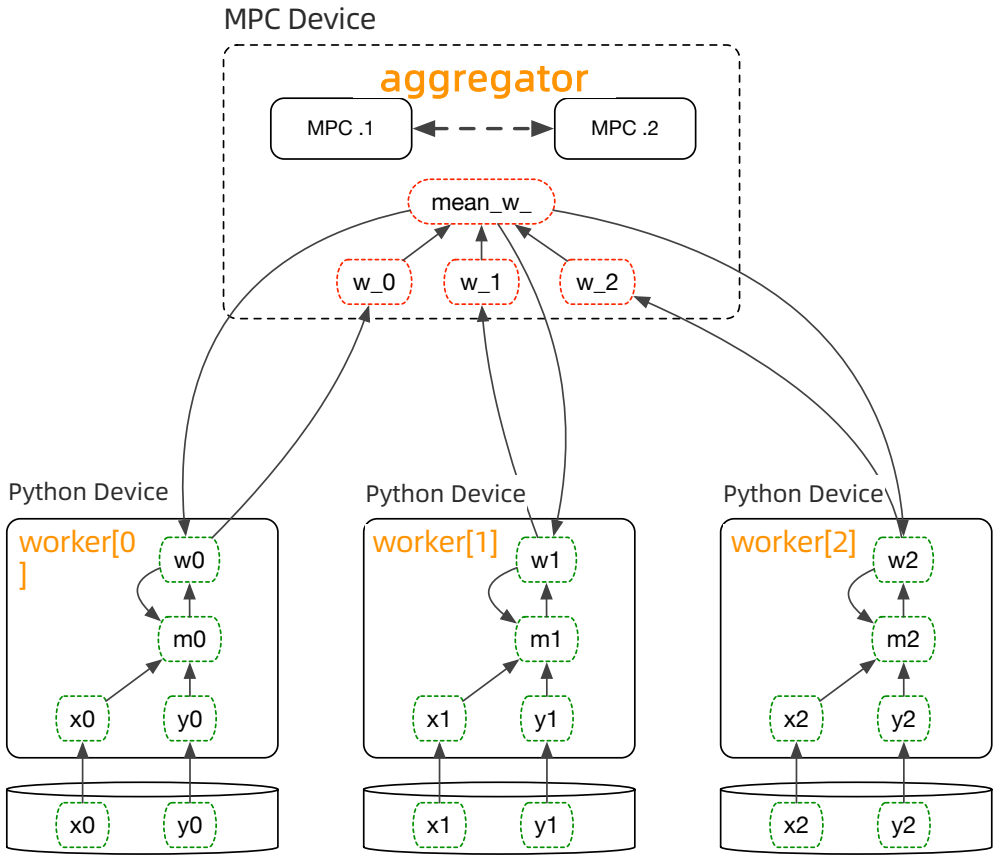
明密文混合编程-FL DEMO

```
domain: ["Client[#i, range(0,3)]", "Server[#j, range(0,2)]"]

device:
  - worker[#i, range(0,3)]:
    type: PYU
    domain: "Client[#i]"
  - aggregator:
    type: MPC
    domain: ["Server[0]", "Server[1]"]
```



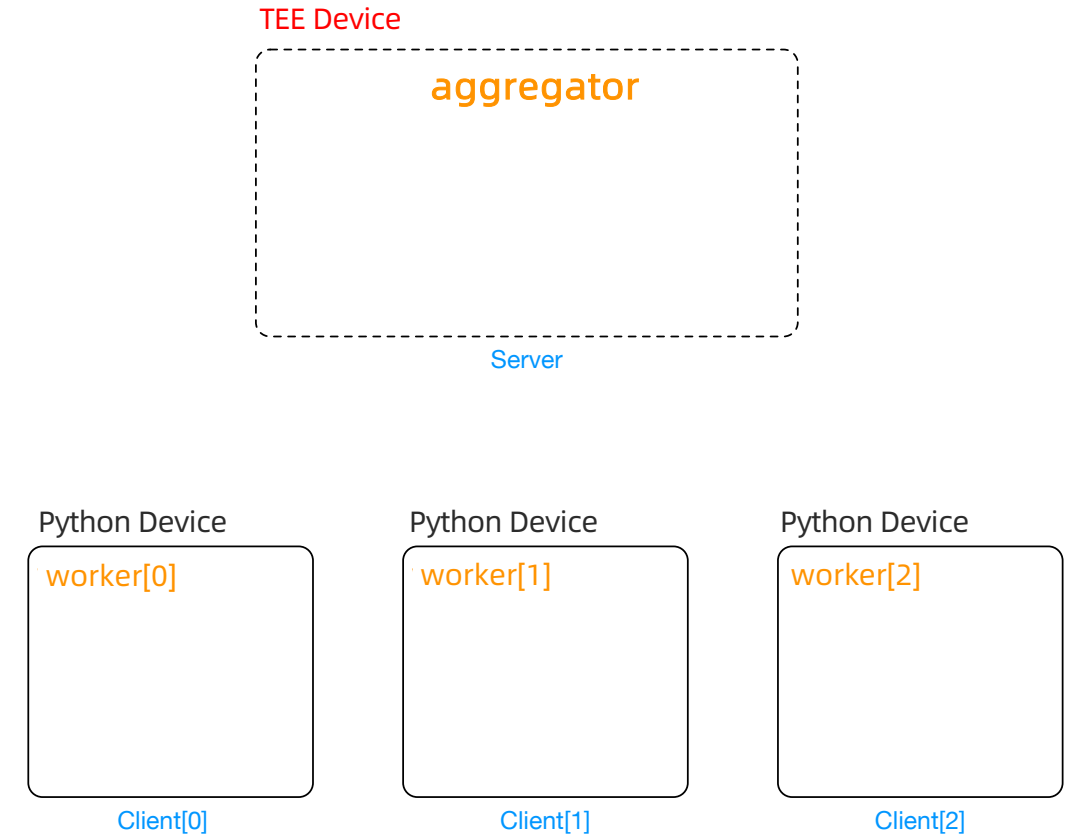
明密文混合编程-FL DEMO

```
def load_data(path):
    XXXX # load data from local path
    return x, y

# initialize device
init_device(device_config) # worker[], aggregator

# load data and build  model
for i in range(len(worker)):
    x[i], y[i] = device(worker[i])(load_data)("/data/x_y.csv")
    m[i] = device(worker[i])(tf.keras.Sequential)(...)

for j in range(10):
    # local train
    for i in range(len(worker)):
        m[i].fit(x[i], y[i])
        w[i] = m[i].get_weights();
        w_[i] = to(w[i], aggregator)
    # secure aggregation
    mean_w_ = device(aggregator)(np.mean)(w_[i], axis = 1)
    # local update
    for i in range(len(worker)):
        w[i] = plaintext_to(mean_w_, worker[i])
        m[i].apply(w[i])
```



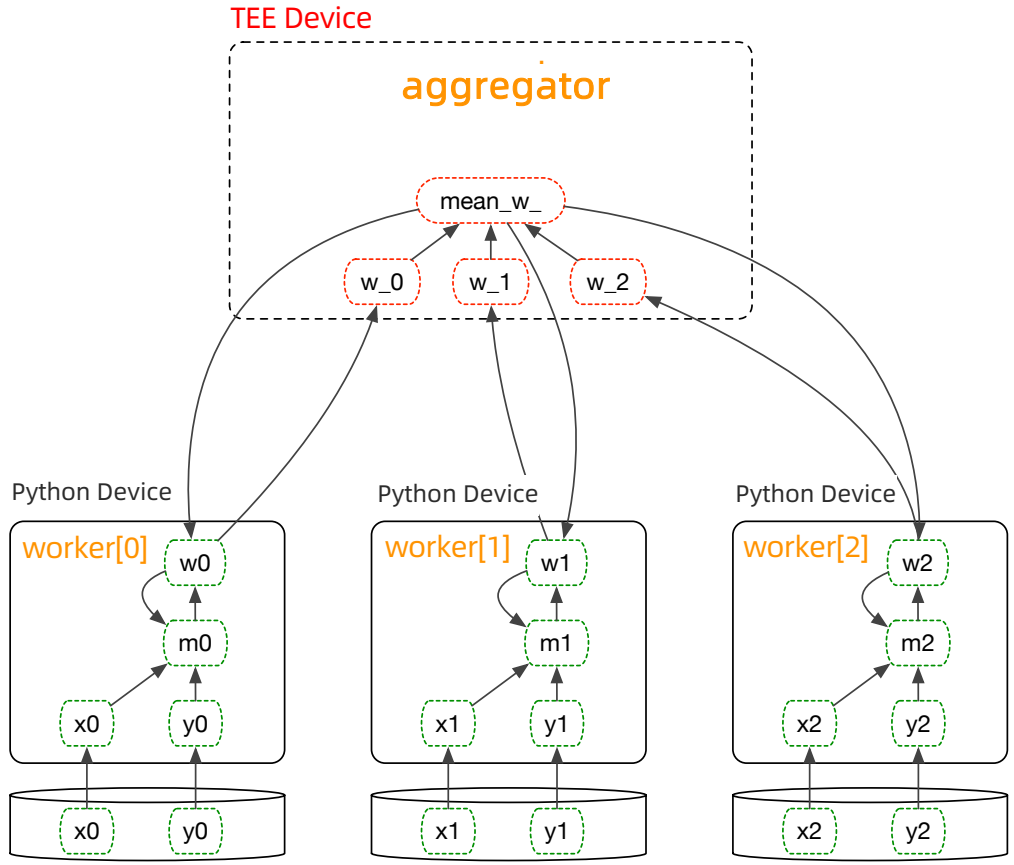
明密文混合编程-FL DEMO

```
domain: ["Client[#i, range(0,3)]", "Server"]

device:
  - worker[#i, range(0,3)]:
    type: PYU
    domain: "Client[#i]"
  - aggregator:
    type: TEE
    domain: ["Server"]
```



明密文混合编程-FL DEMO

```
def load_data(path):
    XXXX # load data from local path
    return x, y

# initialize device
init_device(device_config) # worker[], aggregator

# load data and build  model
for i in range(len(worker)):
    x[i], y[i] = device(worker[i])(load_data)("/data/x_y.csv")
    m[i] = device(worker[i])(tf.keras.Sequential)(…)

for j in range(10):
    # local train
    for i in range(len(worker)):
        m[i].fit(x[i], y[i])
        w[i] = m[i].get_weights();
        w_[i] = to(w[i], aggregator)
    # secure aggregation
    mean_w_ = device(aggregator)(np.mean)(w_[i], axis = 1)
    # local update
    for i in range(len(worker)):
        w[i] = plaintext_to(mean_w_, worker[i])
        m[i].apply(w[i])
```



全链路数据处理能力

06

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备：MPC设备 / HE设备 / TEE设备 / TECC设备
- 明文计算设备：Python 解释器 / SQL 执行环境
- 隐私保护原语：差分隐私 / 脱敏

→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

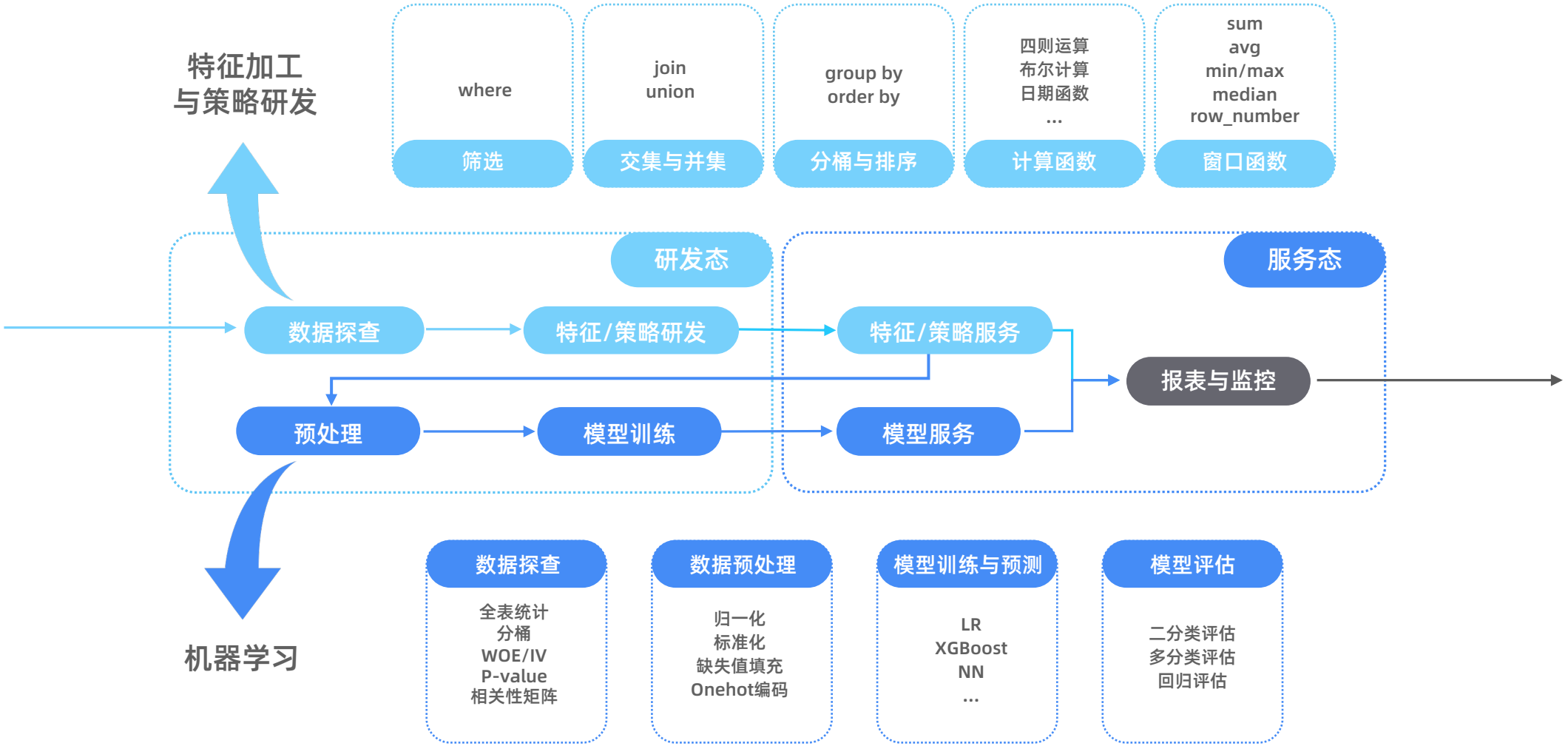
每种解决方案需要具备全链路的数据处理能力

业务全链路的隐私安全级别取决于链路中最薄弱环节的安全级别

目前业界的主要工作集中在对隐私保护机器学习能力的支持

对数据分析、模型服务等环节的重视程度和难度认识不足

隐语提供全链路的数据处理能力



产品化和快速集成

07

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备：MPC设备 / HE设备 / TEE设备 / TECC设备
- 明文计算设备：Python 解释器 / SQL 执行环境
- 隐私保护原语：差分隐私 / 脱敏

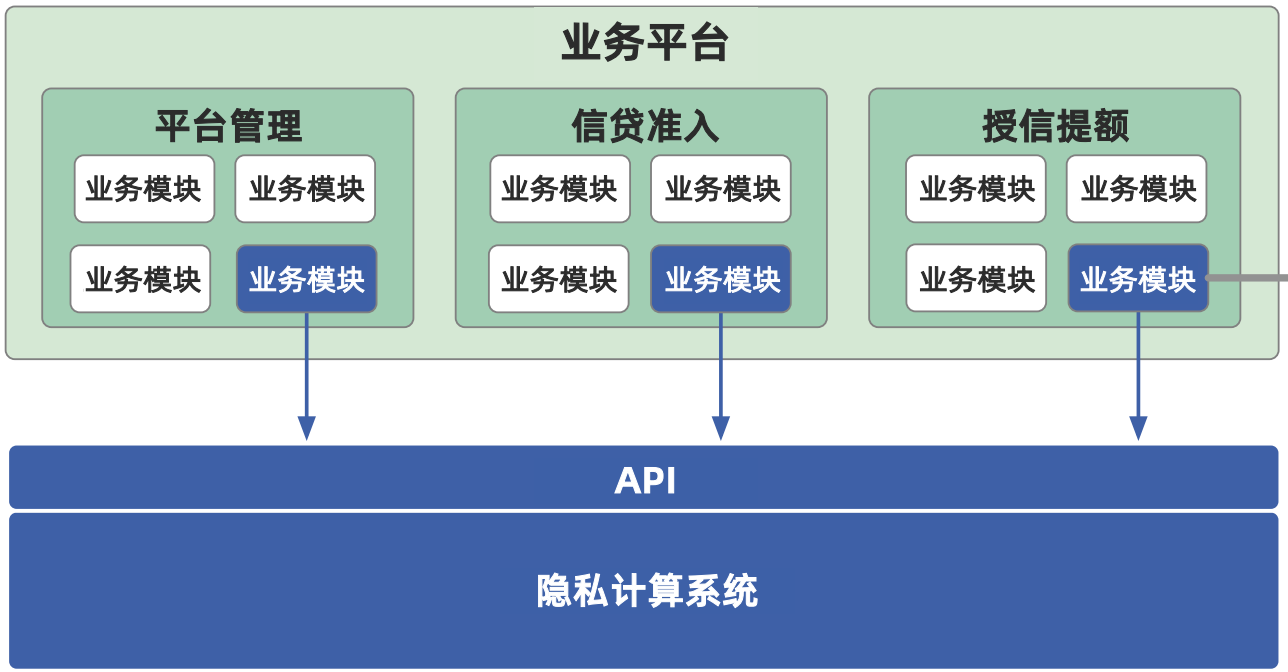
→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

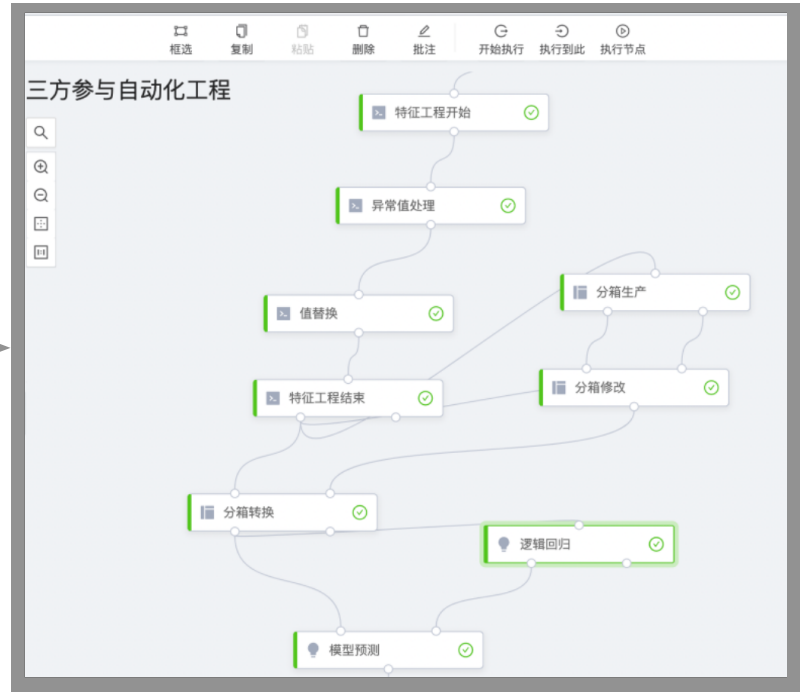
资源管理

- 数据管理
- 计算管理
- 网络管理

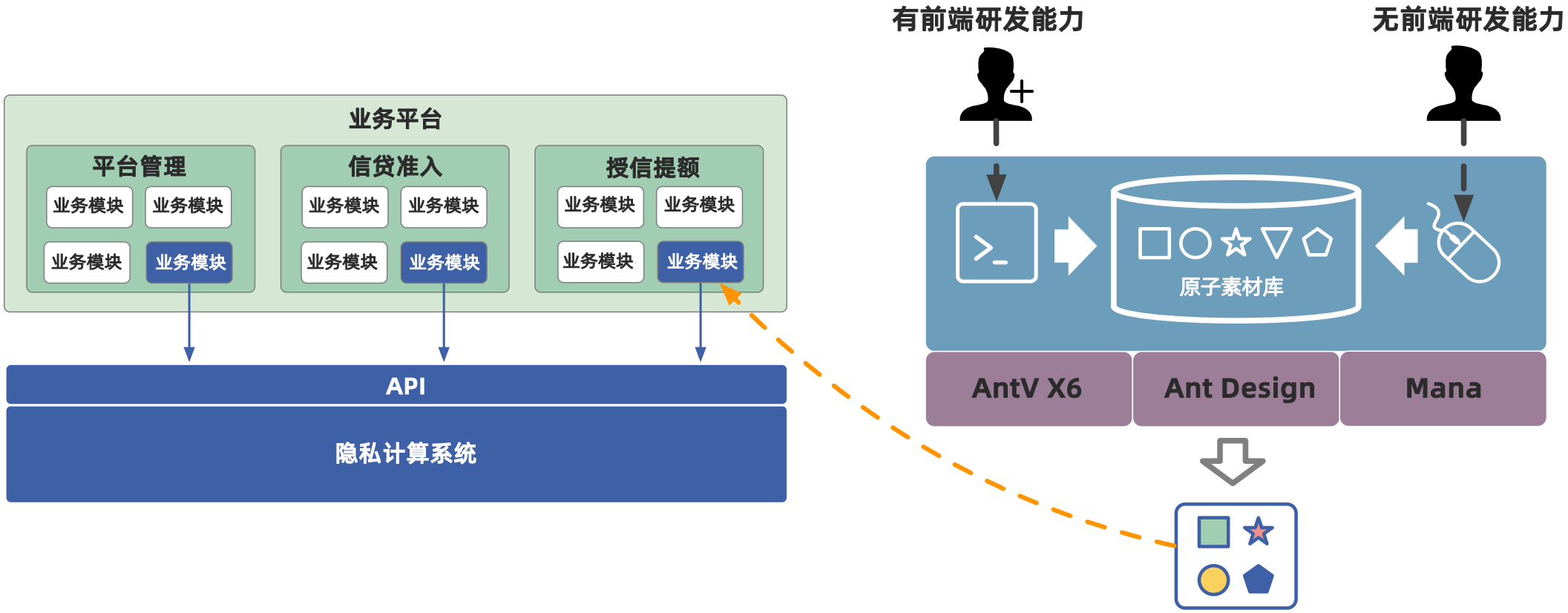
→ 业务交付运维友好
大规模高可用，部署运维成本低

应用系统开发者的痛点


业务平台
平台管理
业务模块
业务模块
业务模块
业务模块
信贷准入
业务模块
业务模块
业务模块
业务模块
授信提额
业务模块
业务模块
业务模块
业务模块
API
隐私计算系统


框选
复制
粘贴
删除
批注
开始执行
执行到此
执行节点
三方参与自动化工程
特征工程开始
异常值处理
值替换
分箱生产
特征工程结束
分箱修改
分箱转换
逻辑回归
模型预测

隐语提供前端原子化集成的能力



从PoC到规模化生产

08

隐语整体架构

用户界面

- 可视化操作界面
- 开放编程接口

→ 业务研发使用友好
平台开发接入成本低

AI & BI 隐私算法

- 多方安全计算
- 联邦学习
- 可信执行环境

→ 隐私保护算法使用友好
提升算法开发效率

明密文混合调度

- 设备计算图
- 分布式调度引擎

→ 调度/编译器开放合作
共建明密文混合编程能力

明密文计算设备与原语

- 密文计算设备：MPC设备 / HE设备 / TEE设备 / TECC设备
- 明文计算设备：Python 解释器 / SQL 执行环境
- 隐私保护原语：差分隐私 / 脱敏

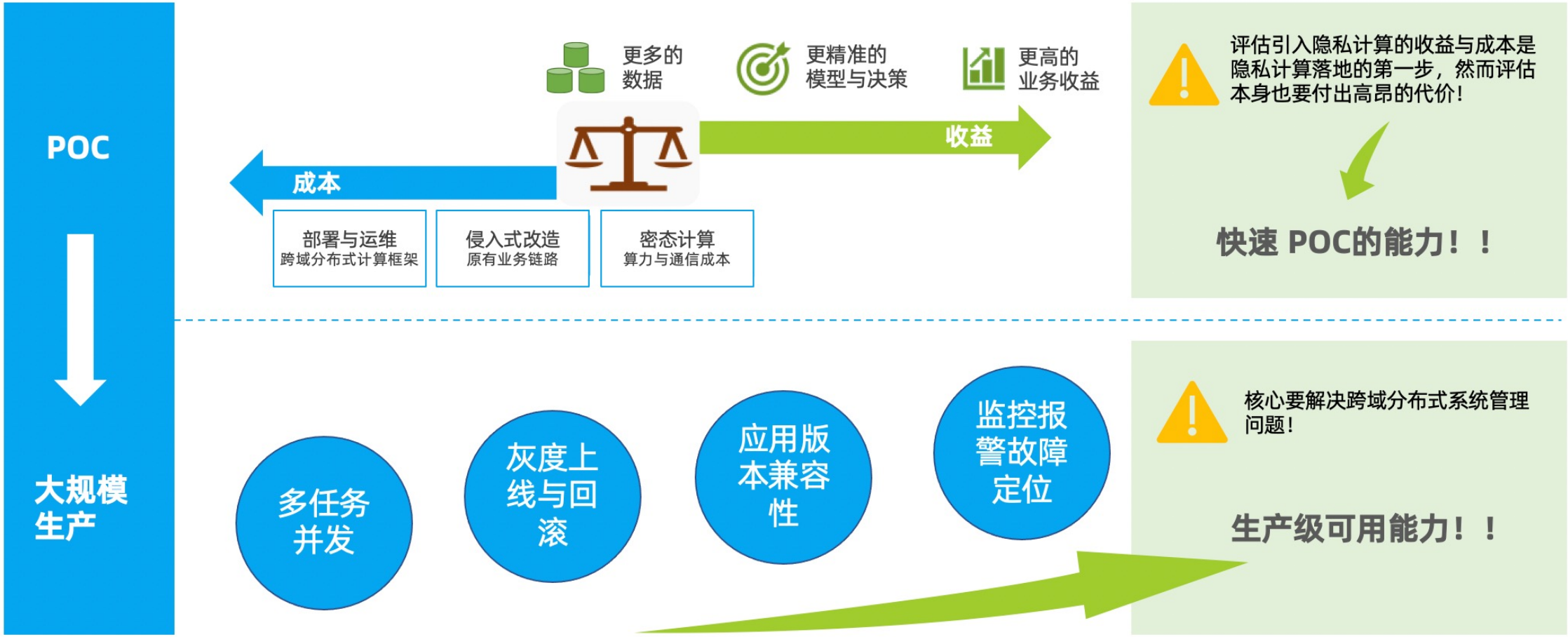
→ 密码/TEE/硬件/AI开放合作
共建密文计算能力和隐私保护原语

资源管理

- 数据管理
- 计算管理
- 网络管理

→ 业务交付运维友好
大规模高可用，部署运维成本低

不同业务孵化阶段的能力诉求不同



隐语提供快速POC和大规模生产的能力

问题

跨机构的在线服务升级时，如何做灰度验证和快速回滚？

不同机构升级窗口不同，导致机构间应用版本不同，如何兼容？

高并发任务，对跨机构的资源需求可能产生死锁，怎么解决？

思路

屏蔽掉不同机构基础设施差异

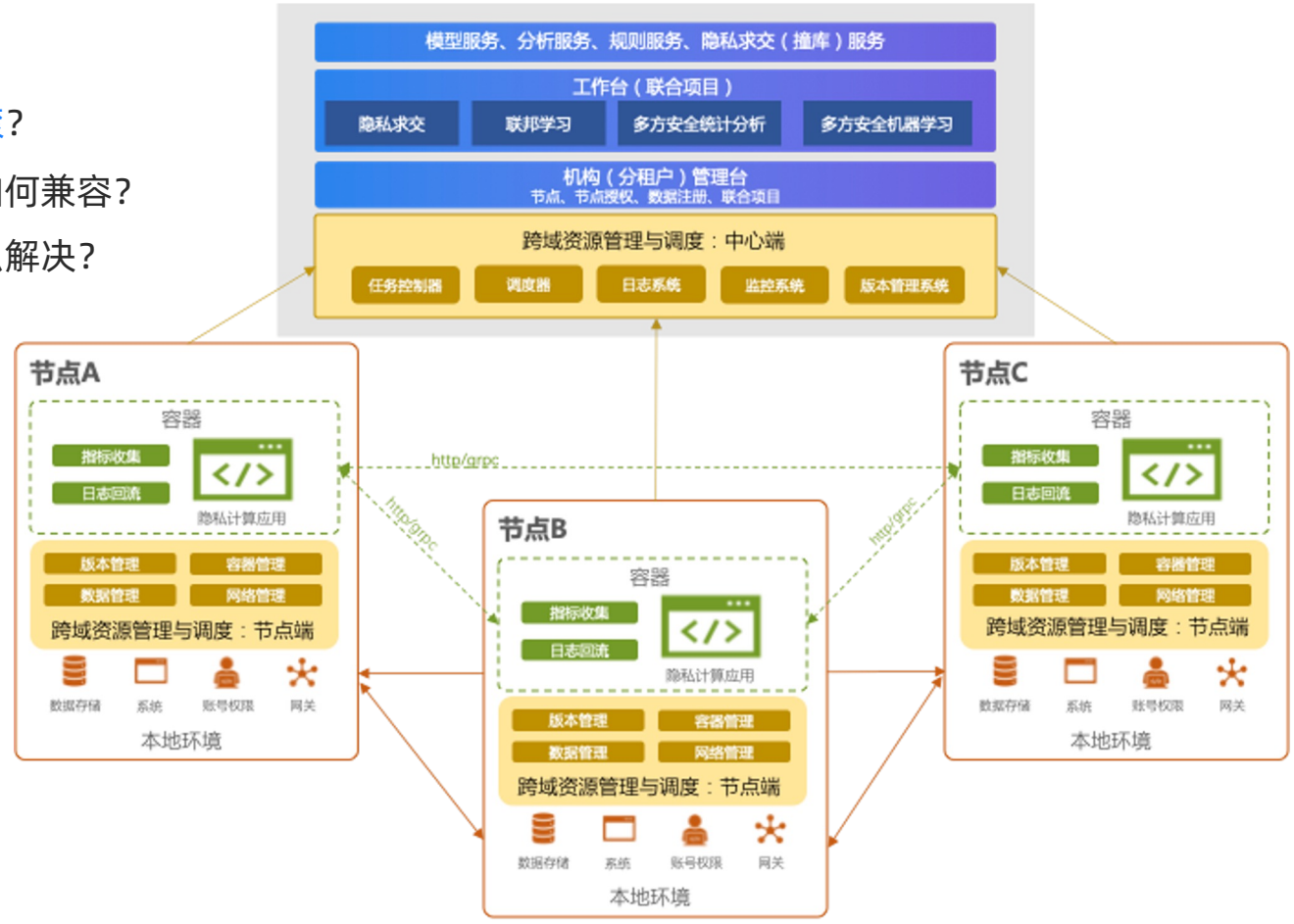
将跨架构的资源组织成统一的资源管理和调度网络

复用现有的规模化生产的经验和能力

难点

机构间的基础设施千差万别

机构的数据和系统安全性如何保证



SECRET FLOW隐语

https://github.com/secretflow

https://gitee.com/secretflow

期待您的反馈和共建

THANKS