

OpenSPG-KAG

KAG: 知识增强生成的垂域知识库知识服务框架

目录

contents

01

大模型垂直领域应用 关键问题

领域知识注入、复杂决策执行、幻觉等问题

02

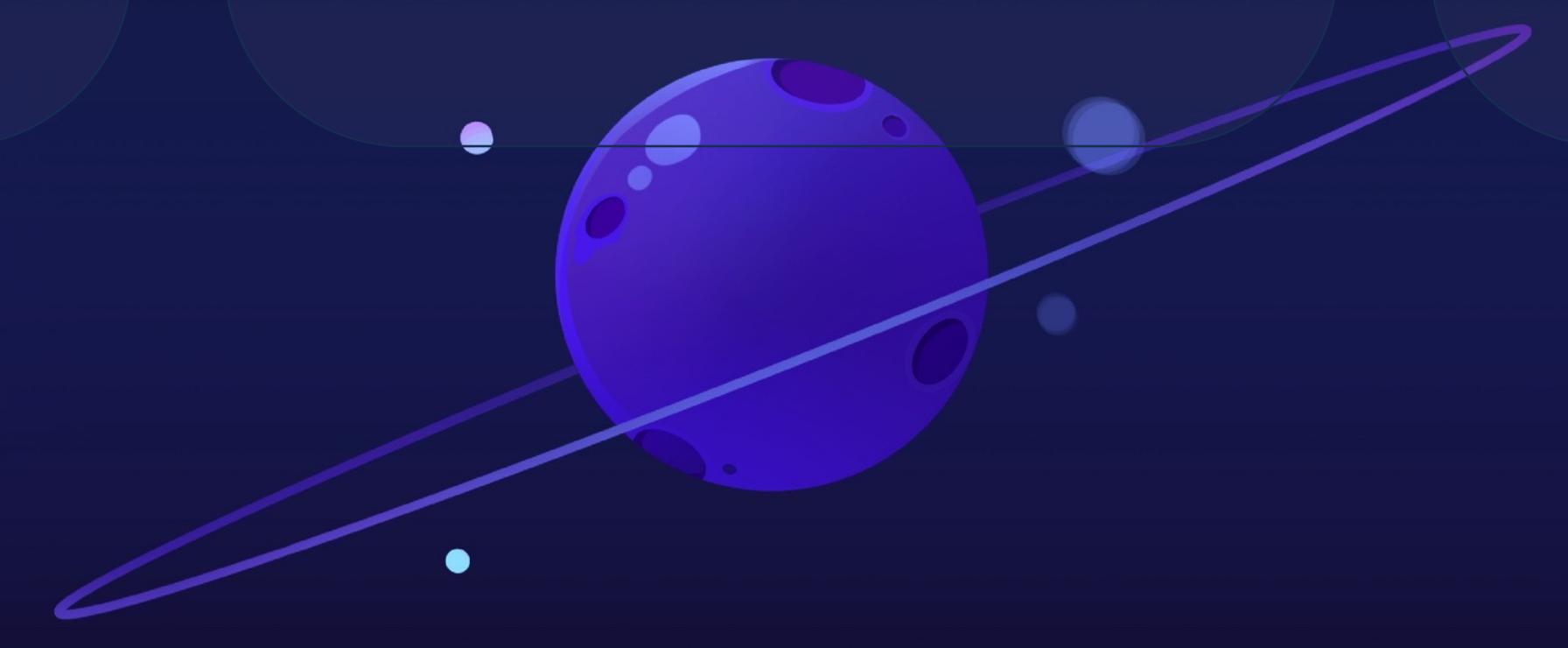
知识增强生成服务框架 (KAG) 介绍

技术架构、kag-builder、kag-solver、自定义扩展、部署&使用

03

KAG 典型应用

风险挖掘、多跳问答、医疗问诊、事理图谱等

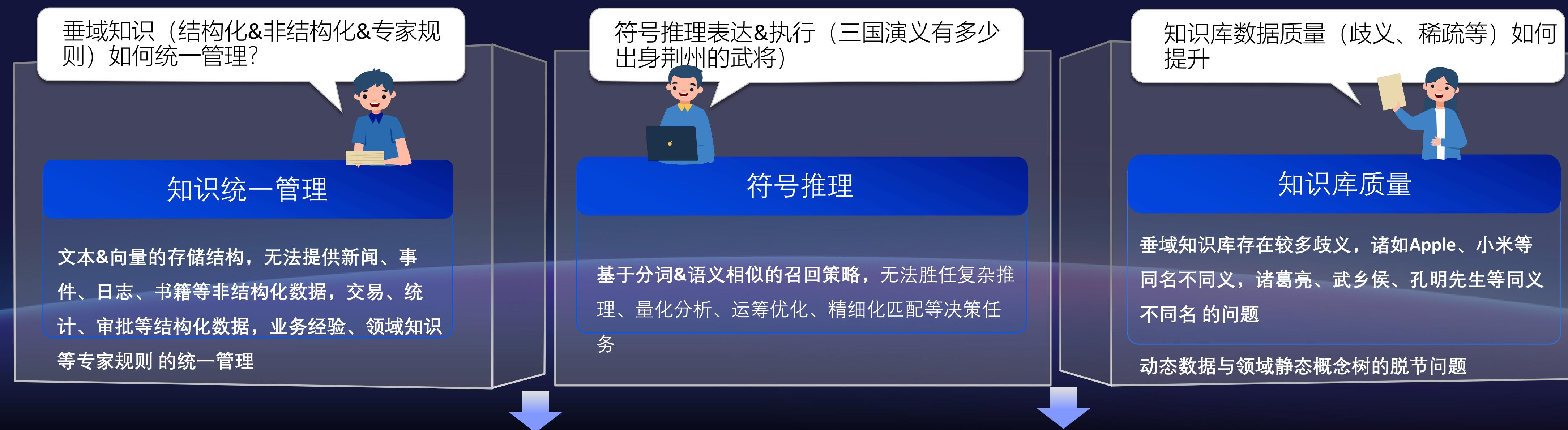


大模型垂域应用(RAG+LLM)关键问题

LLM 垂域应用一般配备私域知识库，以解决：

- 垂域隐私数据难以成为 开源&商业 大模型预训练语料
- 大模型 sft 对人员能力&资源配置 要求较高
- 大模型 sft 耗时长，难以与知识库更新保持同步

RAG+LLM 模式落地遇到的挑战



知识图谱所倡导技术路线（语义、逻辑、符号等）可为大模型垂域应用提供更好支持

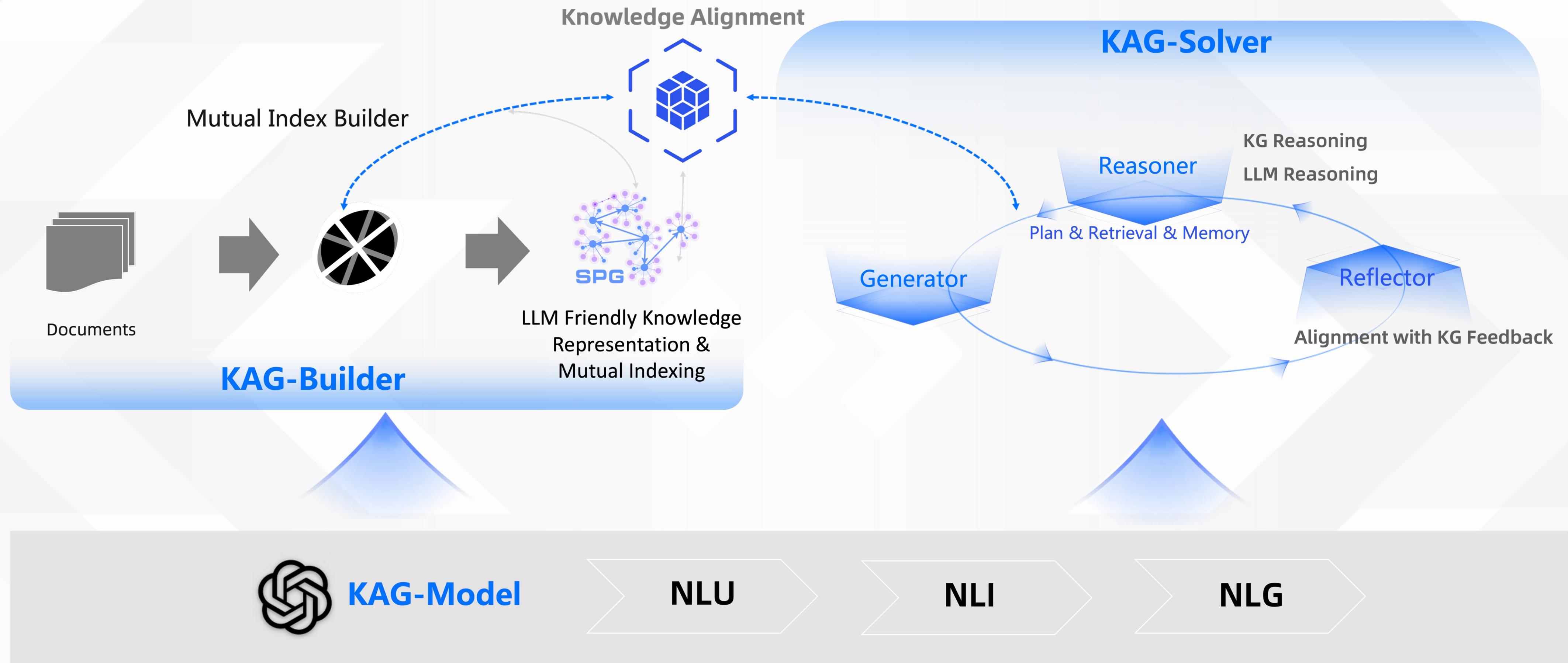
图谱/图+大模型 典型技术路线

框架	适用场景	特点
GraphRAG(MS)	摘要生成类任务(评估方式:可理解性、完整性、多视角)	<ul style="list-style-type: none"> 通过层次聚类实现段落摘要的逐级生成，更关注答案生成的可理解性、完整性、多视角 多跳问答等评测集量化指标较差，未提供逻辑符号推理的能力
HippoRAG	事实问答类任务(评估方式:em、f1)	<ul style="list-style-type: none"> 通过rdf 抽取 + 语义相似拉边，完成图谱构建 问答阶段，通过dpr + ppr 实现Chunk 召回 未利用语义、逻辑、符号等图谱技术栈
LightRAG	摘要生成类任务(评估方式:可理解性、完整性、多视角)	<ul style="list-style-type: none"> 通过rdf 五元组（带类型）抽取完成图谱构建 问答阶段，通过对query 中所包含实体、实体归属的概念实现Chunk 召回 未利用语义、逻辑、符号等图谱技术栈
OpenSPG-KAG (V0.5)	事实问答类任务+逻辑推理类任务(评估方式:em、f1)	<ul style="list-style-type: none"> 基于知识抽取、语义对齐、文本&图互索引等完成图谱知识库构建 基于逻辑符号引导的混合推理, 实现事实问答&逻辑推理类任务。 摘要生成、对话问答类任务，待开源… kag-model 小模型媲美大模型效果，待开源…

KAG 介绍

V0.5 版本

KAG - 技术架构

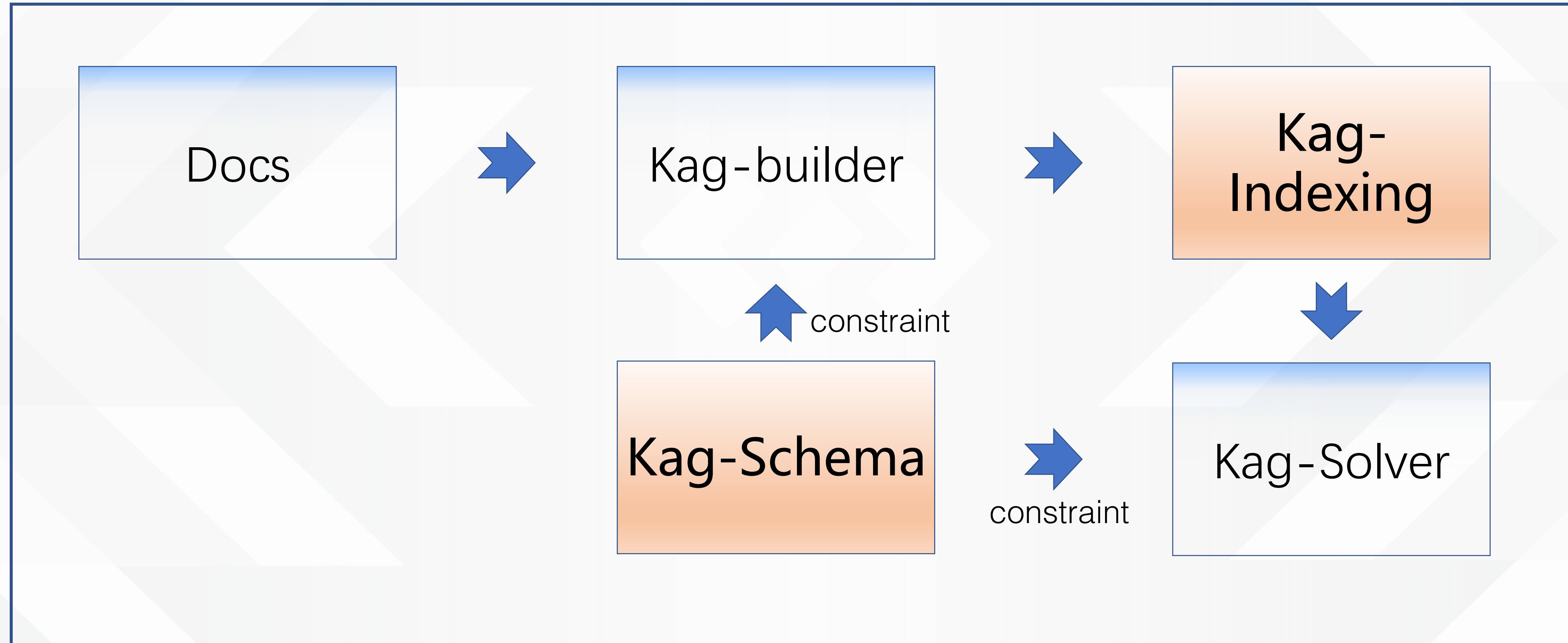


Kag-builder: 大模型友好的知识表示

Kag-solver: 逻辑符号引导的混合推理引擎

Kag-Model: 10B sft 模型媲美 72B 模型效果(NLU, NLI, NLG 任务), 资源消耗显著降低

KAG-Schema & Indexing



Kag-Schema(升级SPG为面向大模型友好)

medicine.schema(医疗)

```

namespace Medical

Disease(疾病): EntityType
properties:
    desc(描述): Text
    index: Text
semanticType(语义类型): Text
index: Text
complication(并发症): Disease
constraint: MultiValue
commonSymptom(常见症状): Symptom
constraint: MultiValue
applicableMedicine(适用药品): Medicine
constraint: MultiValue
diseaseSite(发病部位): HumanBodyPart
constraint: MultiValue

HospitalDepartment(科室): ConceptType
hypernymPredicate: isa

HumanBodyPart(人体部位): ConceptType
hypernymPredicate: isa

HypertensionLevel(高血压分级): ConceptType

SCAD(稳定性冠心病): ConceptType

Drug(药品): ConceptType
.....

```

Hypertension.rule (高血压诊断&用药规则)

```

namespace Medical

(HypertensionLevel/`高血压`):
rule: [[
    收缩压>=140 or 舒张压>=90
]]

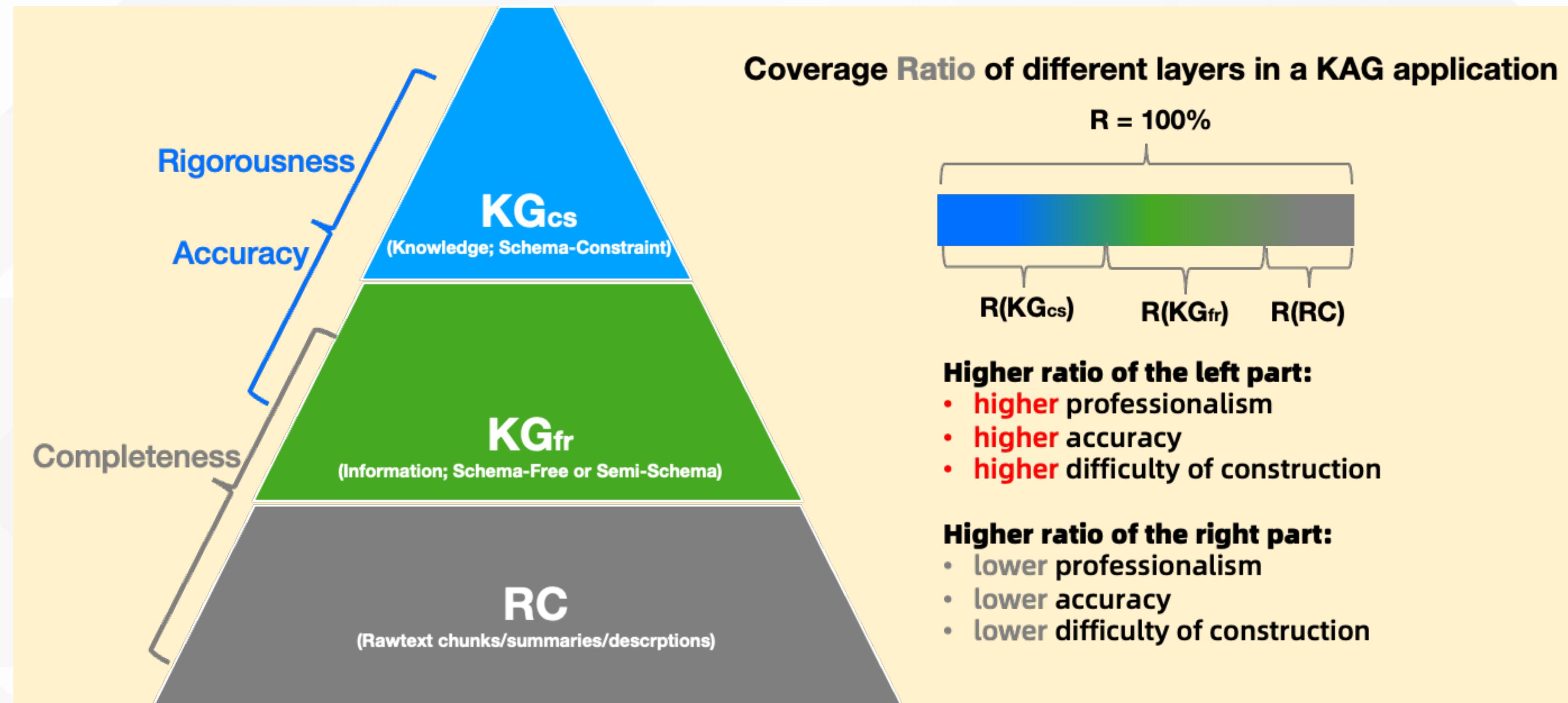
[联合用药方案]->(Drug/`ACEI`+`β受体阻滞剂`):
rule: [[
    ()-[:conclude]->(:Medical.SCAD) and ()-[:适用药品]->(:Medical.Drug/`β受体阻滞剂`) or ()-[:推荐药品]->(:Medical.Drug/`β受体阻滞剂`)) and ()-[:适用药品]->(:Medical.Drug/`ACEI`) or ()-[:推荐药品]->(:Medical.Drug/`ACEI`)
]]

[适用药品]->(Drug/`β受体阻滞剂`):
rule: [[
    ()-[:conclude]->(:Medical.SCAD) and HypertensionLevel/`高血压`]
]]

[推荐药品]->(Drug/`β受体阻滞剂`):
rule: [[
    HypertensionLevel/`高血压` and ()-[:conclude]->(:Medical.SCAD) and 心率 > 60
]]

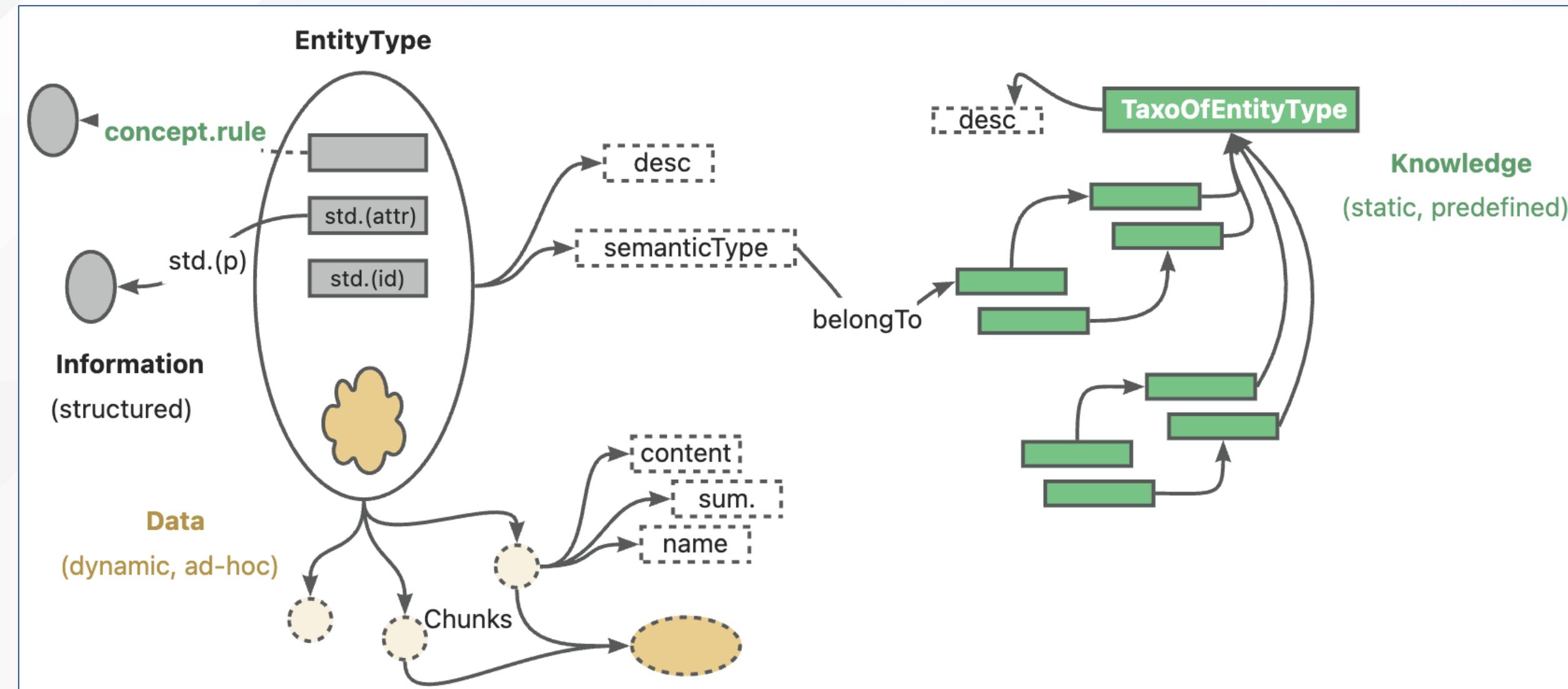
```

Kag-Indexing知识分层

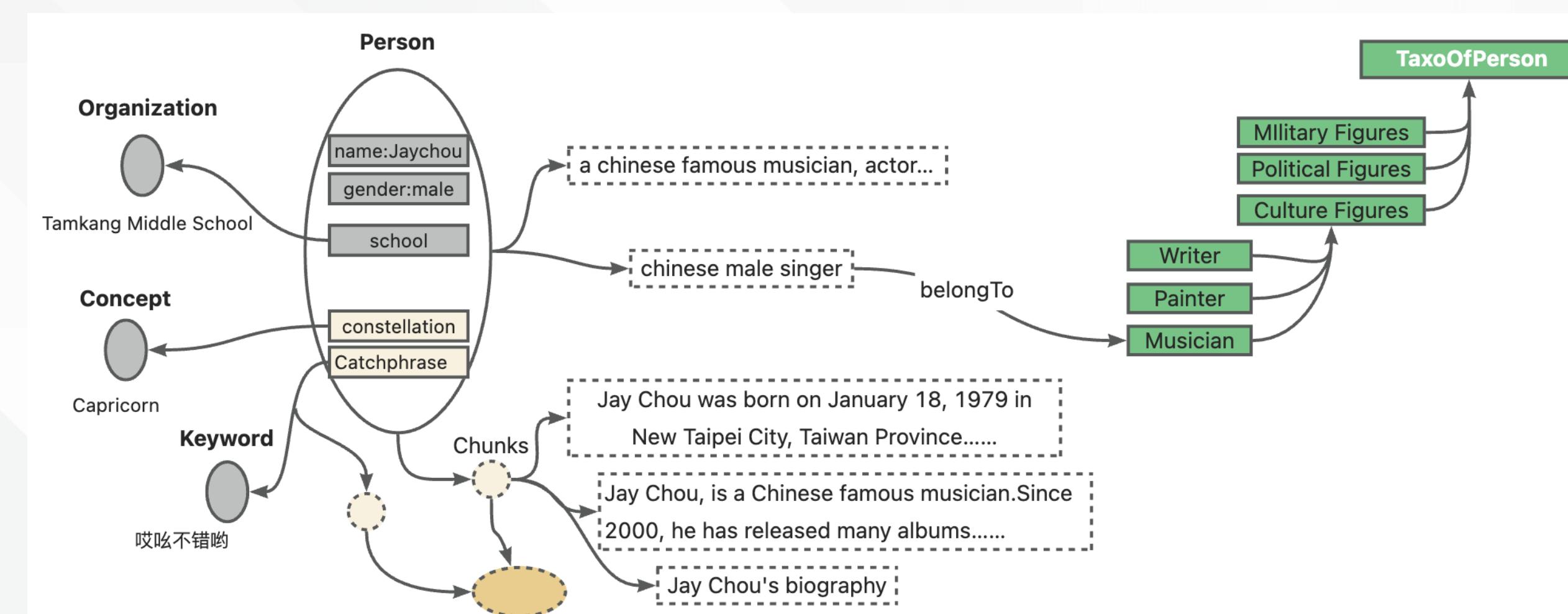


- 升级SPG为面向大模型友好的知识表示LLMFriSPG
- 兼容强Schema**专业知识**和弱Schema**开放信息**
- 图结构知识与文本知识的**互索引结构**
- 专业领域可平滑调节的**专业决策与信息检索，丰富知识完备性**

Kag – Indexing Structure



Kag – Indexing instance of Jay Chou



default.schema

Organization(组织机构): EntityType
properties:
 id(主键): Text
 index: TextAndVector
 name(机构名): Text
 index: TextAndVector
 desc(描述): Text
 index: TextAndVector
 semanticType(语义类型): Text

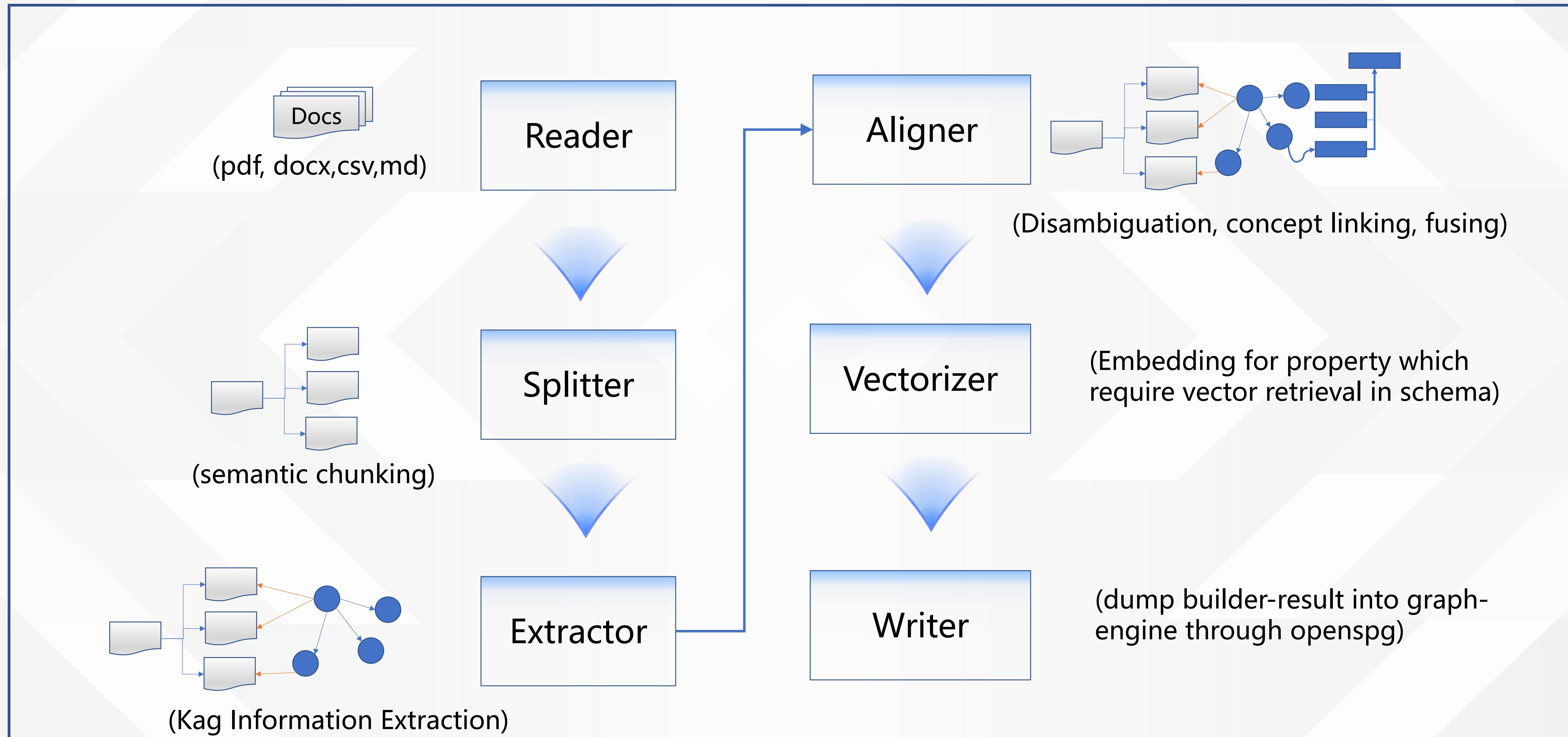
Person(人物): EntityType
properties:
 id(主键): Text
 index: TextAndVector
 name(姓名): Text
 index: TextAndVector
 desc(描述): Text
 index: TextAndVector
 school (毕业院校): Organization
 gender ('性别'): Text
 semanticType(语义类型): Text

Works(作品): EntityType
Concept(概念): EntityType
GeoLocation(地理位置): EntityType

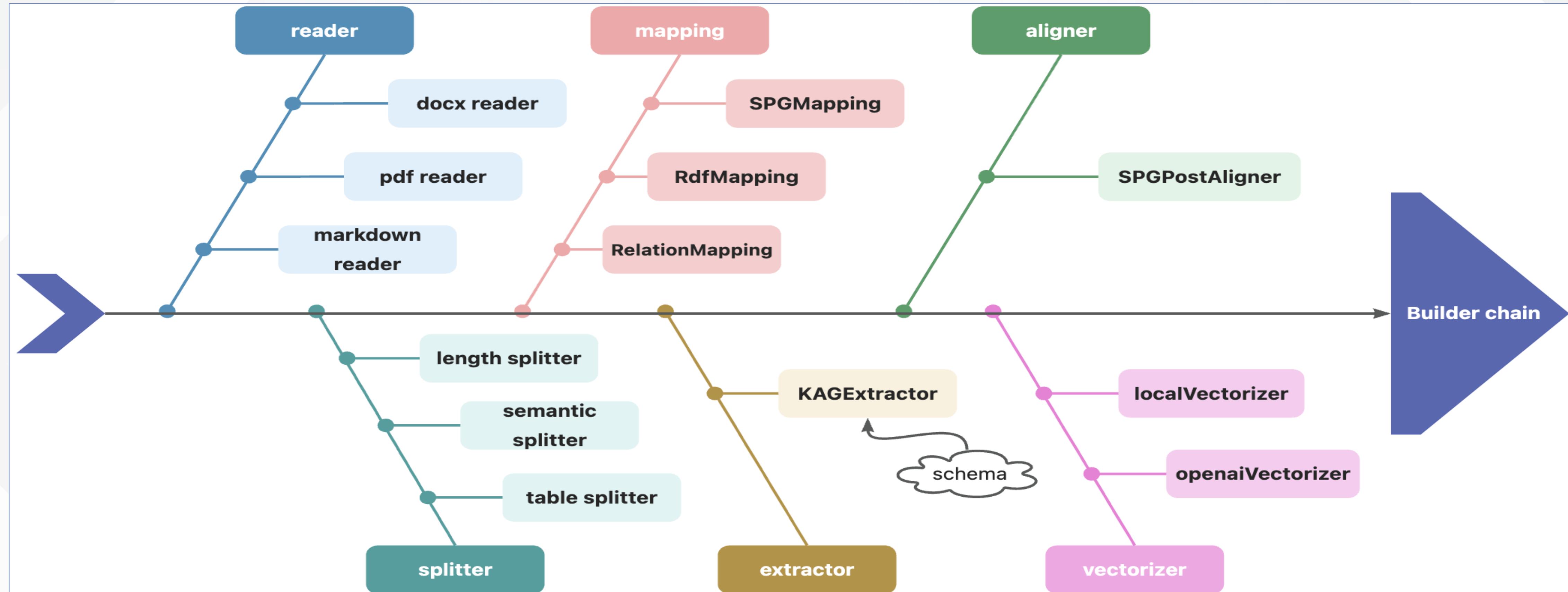
Chunks(文章段落): EntityType

Others(其它): EntityType

Kag-builder

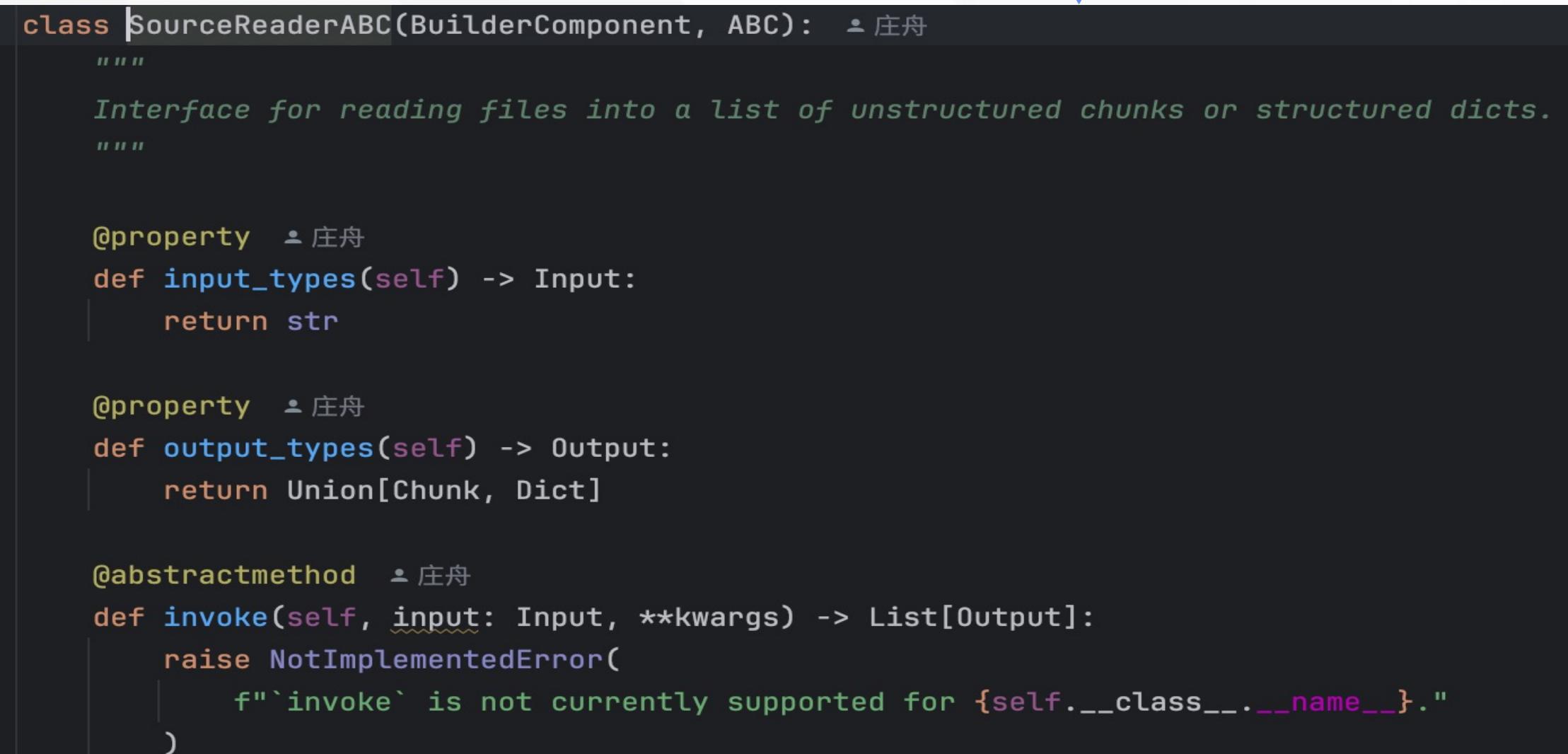
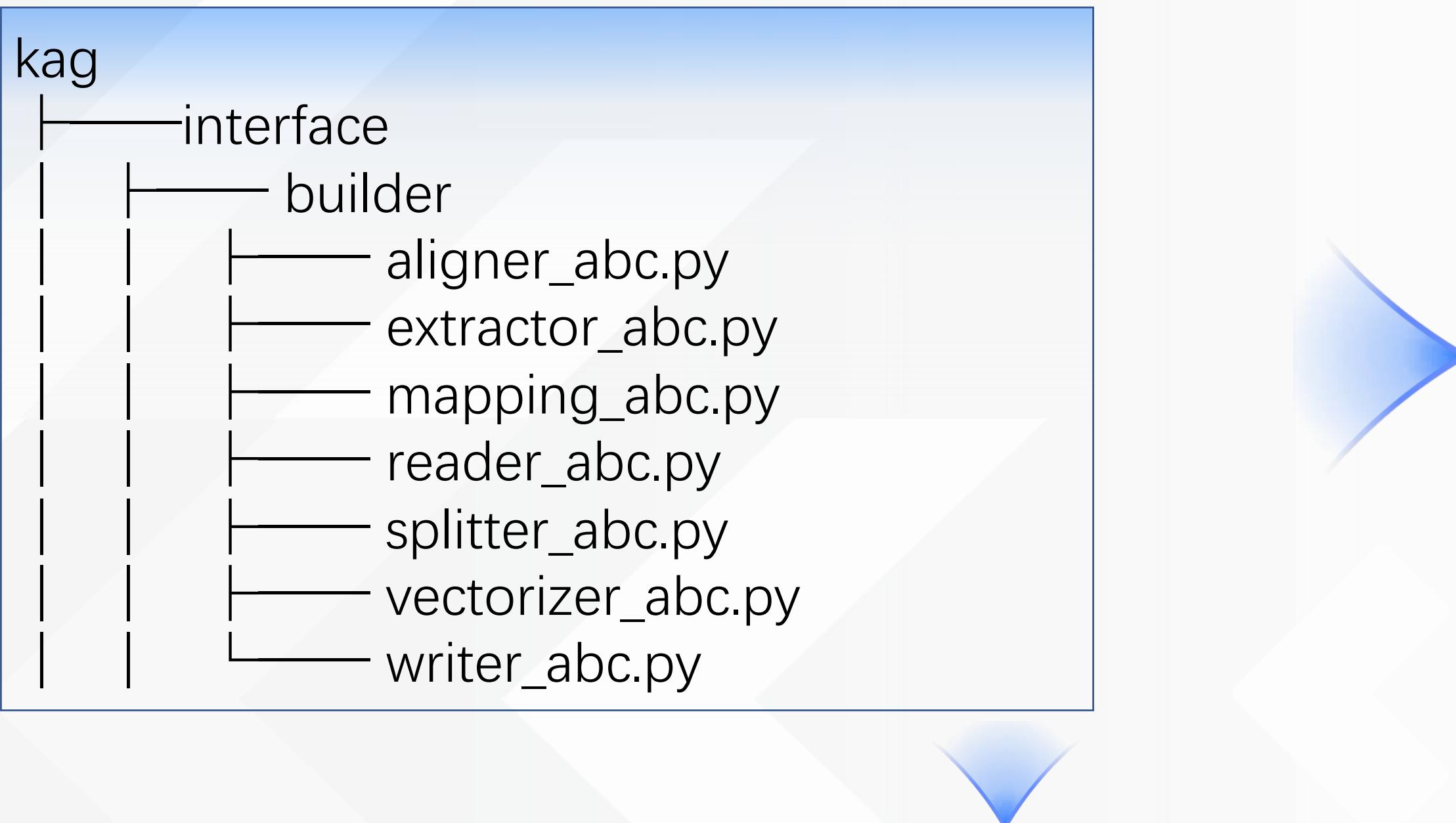


Kag-builder (built-in components)



- Kag 内置了reader, splitter, mapping, extractor, aligner, vectorizer 的多种实现
- Kag-builder chain 前一个节点的输出格式与后一个节点的输入格式对齐
- Schema (默认、自定义) 引导大模型完成图谱知识抽取&语义对齐
- Kag-builder 生成的subgraph 子图，调用openspg-writer接口写入图存引擎

Kag-builder (Interface & example)

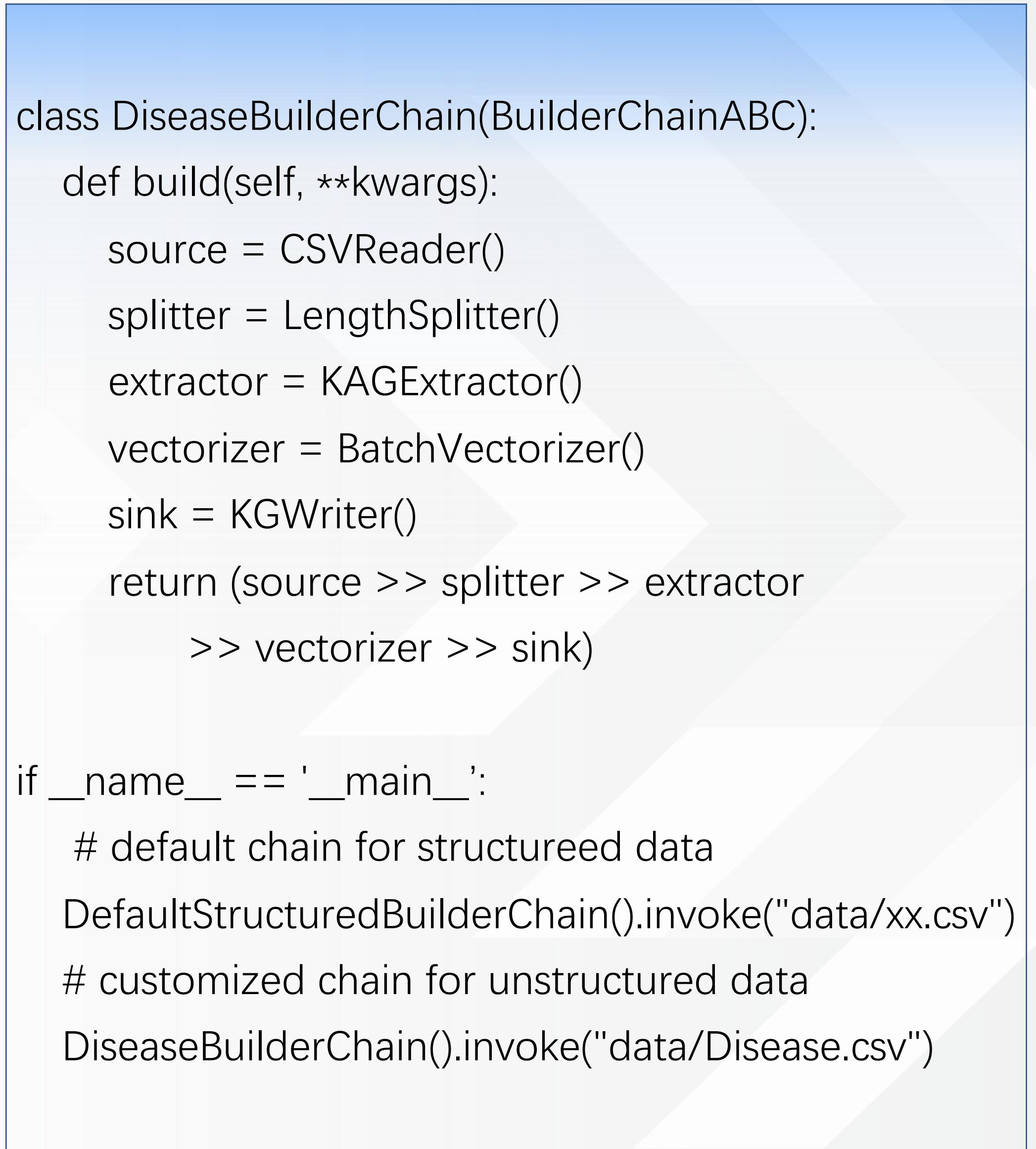


```
class SourceReaderABC(BuilderComponent, ABC): # 庄舟
    """
    Interface for reading files into a list of unstructured chunks or structured dicts.
    """

    @property # 庄舟
    def input_types(self) -> Input:
        return str

    @property # 庄舟
    def output_types(self) -> Output:
        return Union[Chunk, Dict]

    @abstractmethod # 庄舟
    def invoke(self, input: Input, **kwargs) -> List[Output]:
        raise NotImplementedError(
            f"`invoke` is not currently supported for {self.__class__.__name__}."
```

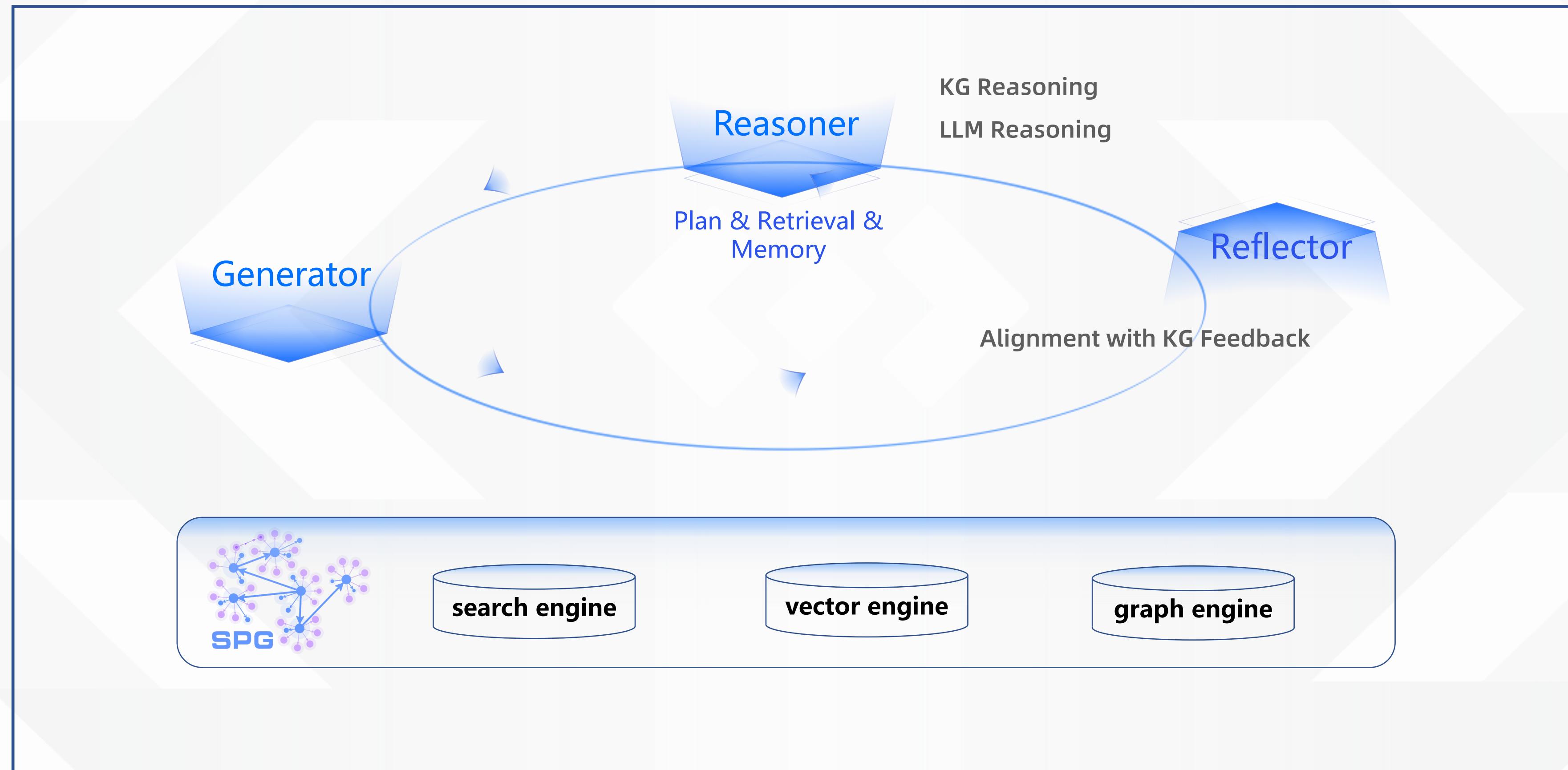


```
class DiseaseBuilderChain(BuilderChainABC):
    def build(self, **kwargs):
        source = CSVReader()
        splitter = LengthSplitter()
        extractor = KAGEExtractor()
        vectorizer = BatchVectorizer()
        sink = KGWriter()

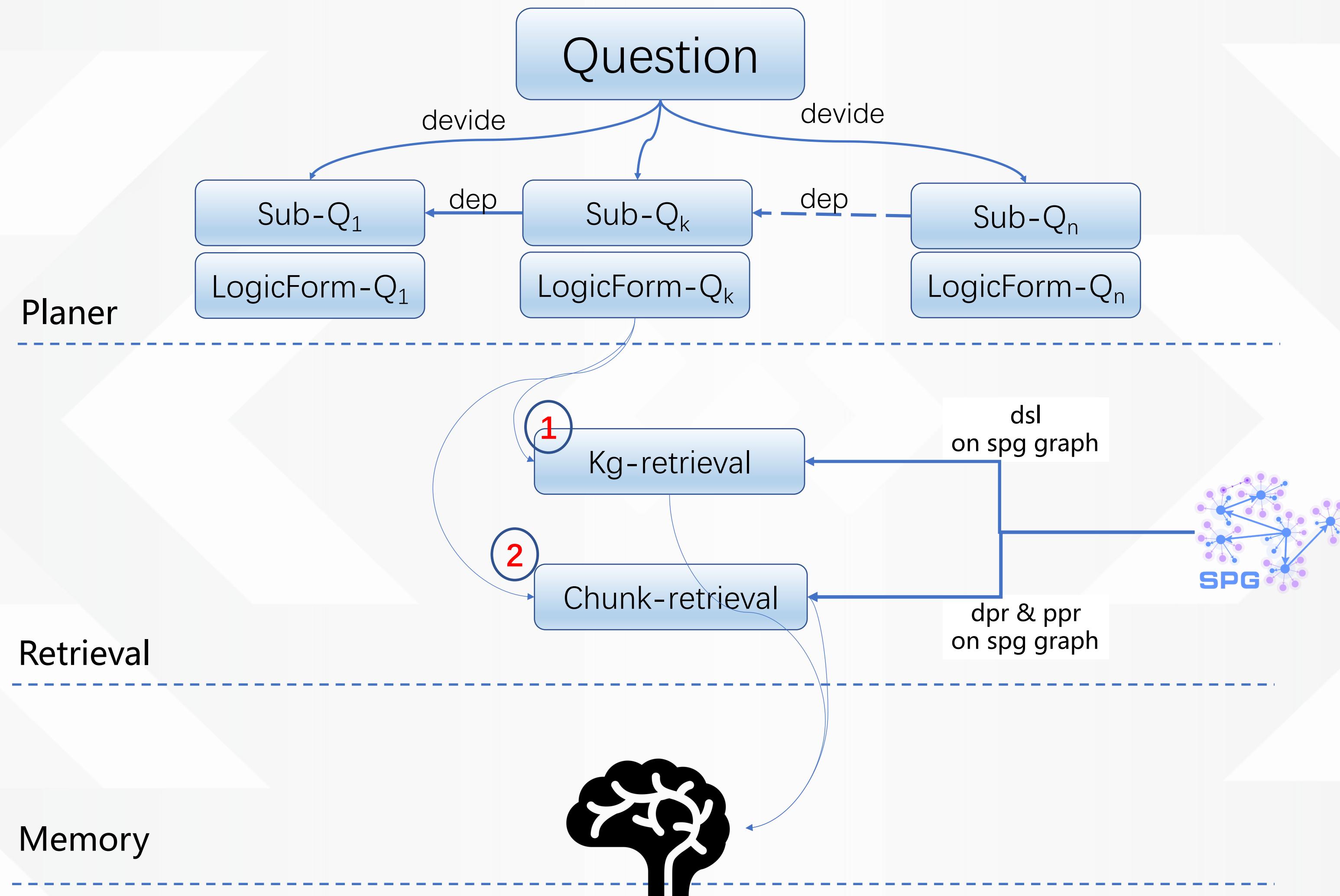
        return (source >> splitter >> extractor
               >> vectorizer >> sink)

if __name__ == '__main__':
    # default chain for structureed data
    DefaultStructuredBuilderChain().invoke("data/xx.csv")
    # customized chain for unstructured data
    DiseaseBuilderChain().invoke("data/Disease.csv")
```

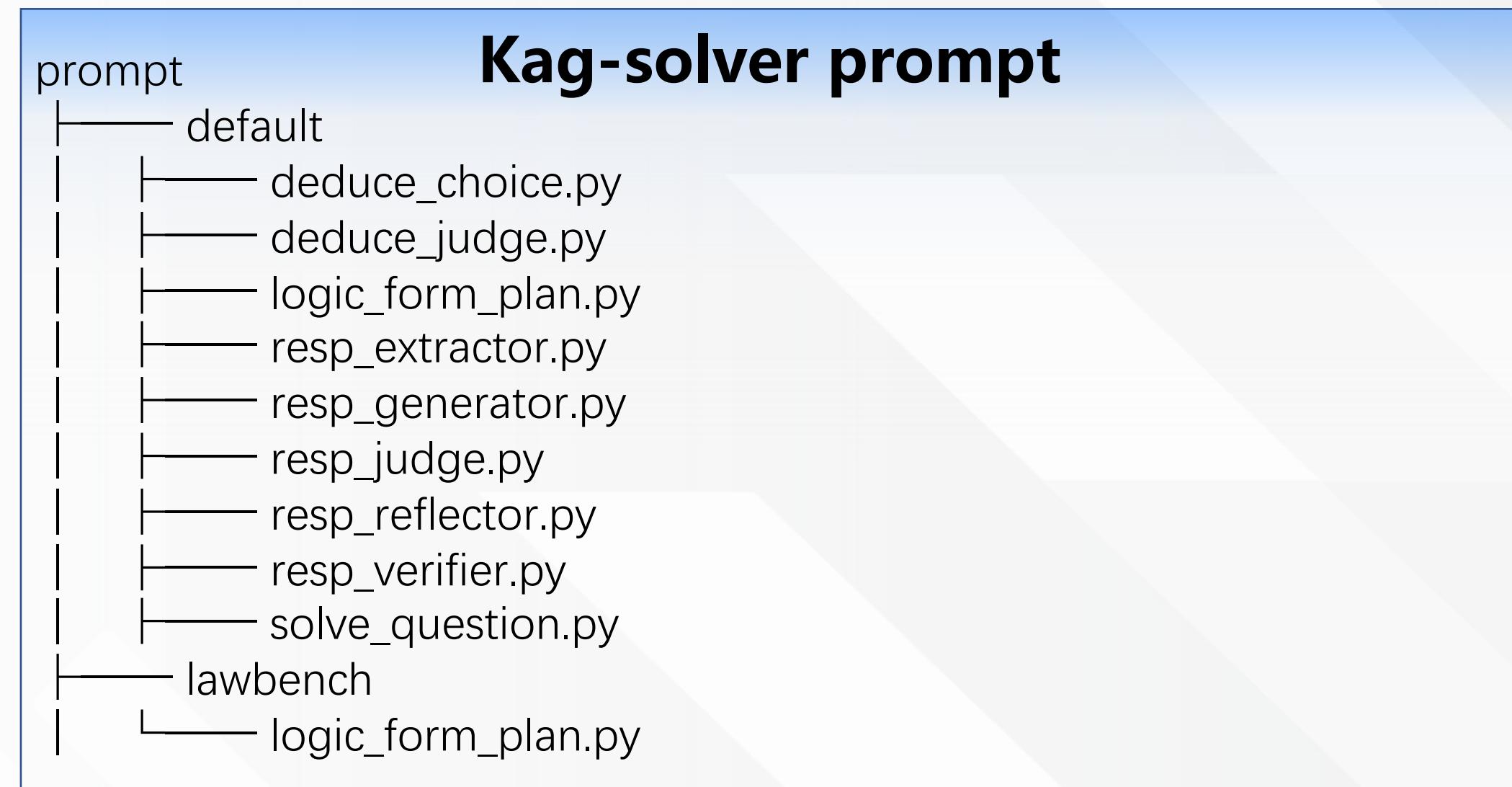
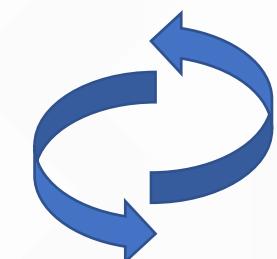
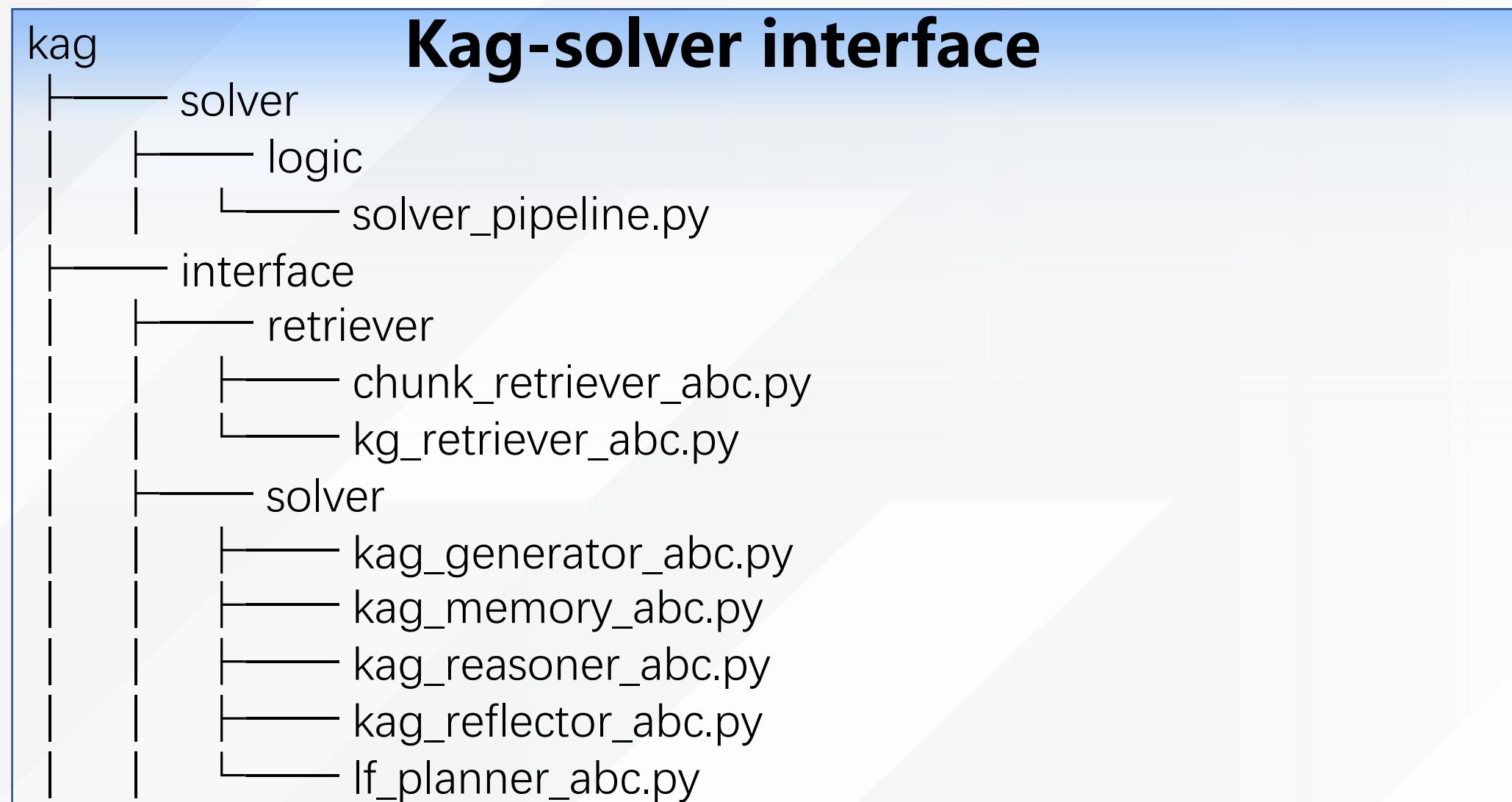
KAG-Solver



Kag-Solver (reasoner)



Kag-Solver (自定义扩展)



LogicForm Plan interface

```

class LFPlannerABC(KagBaseModule, ABC):
    """
        LFPlanResult is a LogicForm expression.
        kag provides Retrieval(kg-retrieval, chunk-retrieval) function for LogicForm execution
        kag provides Deduce (judgement|entailment|rule|choice|multiChoice) function for LogicForm execution
        kag provides Output(refer output of former function) function for LogicForm execution
    """

    @abstractmethod
    def lf_planing(self, question, llm_output=None) -> List[LFPlanResult]:
        """
            developer can invoke llm guided by prompt to generate LFPlanResult"""
        pass

```

LogicForm Solver interface

```

class LFSolver:
    """
        kg_retriever recall subgraph through graph pattern match on kgdsl
        chunk_retriever recall Chunk by combining ppr & dpr
    """

    def __init__(self, kg_retriever: KGRetrieverABC = None,
                 chunk_retriever: ChunkRetrieverABC = None,
                 report_tool=None, **kwargs):
        pass

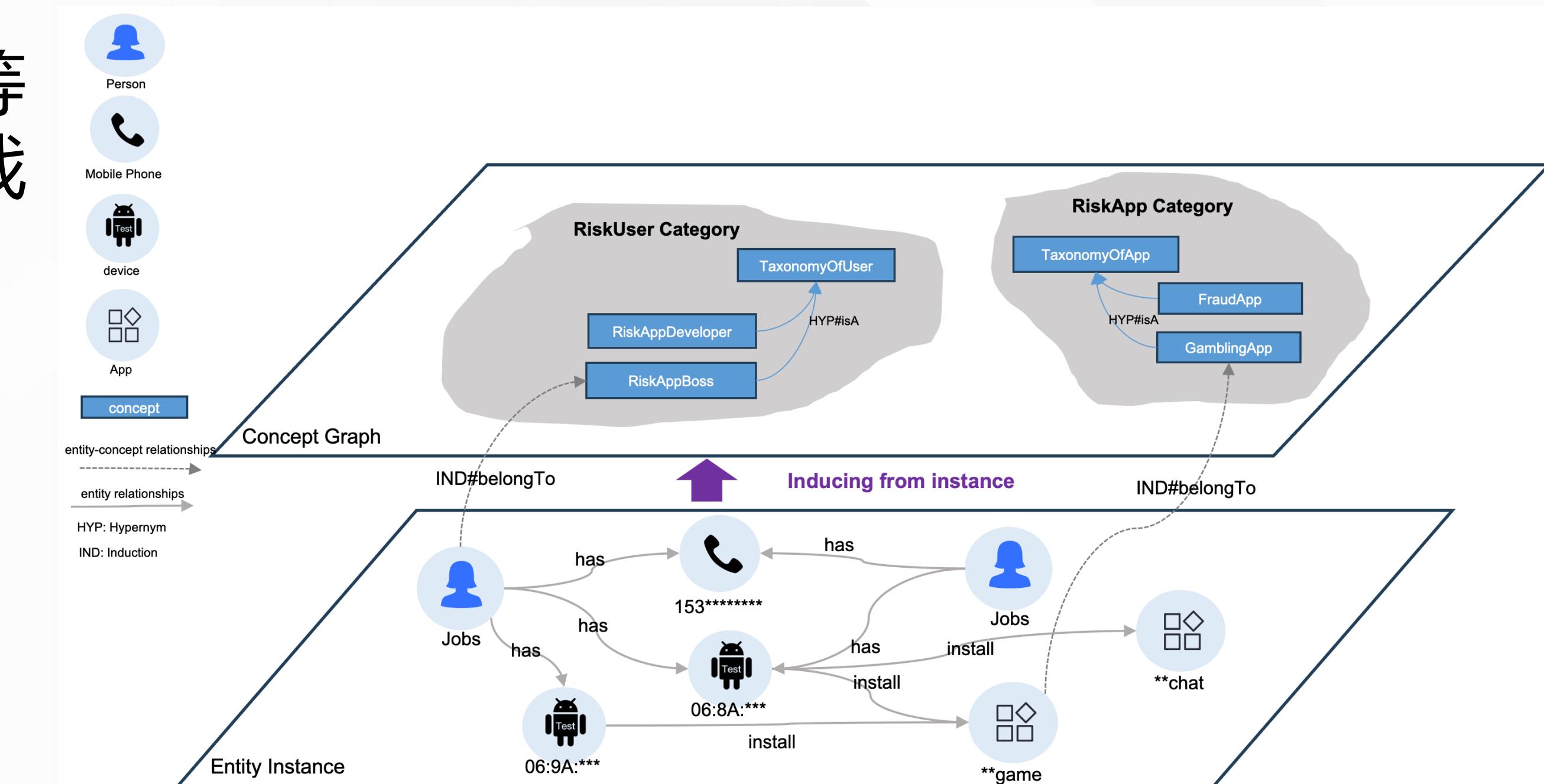
    def solve(self, query, lf_nodes: List[LFPlanResult]):
        """
        Returns:
            tuple: A tuple containing the final answer, sub-query-answer pairs,
            relevant documents, and history.
        """

```

基于KAG自定义严谨决策（示例-黑产图谱）

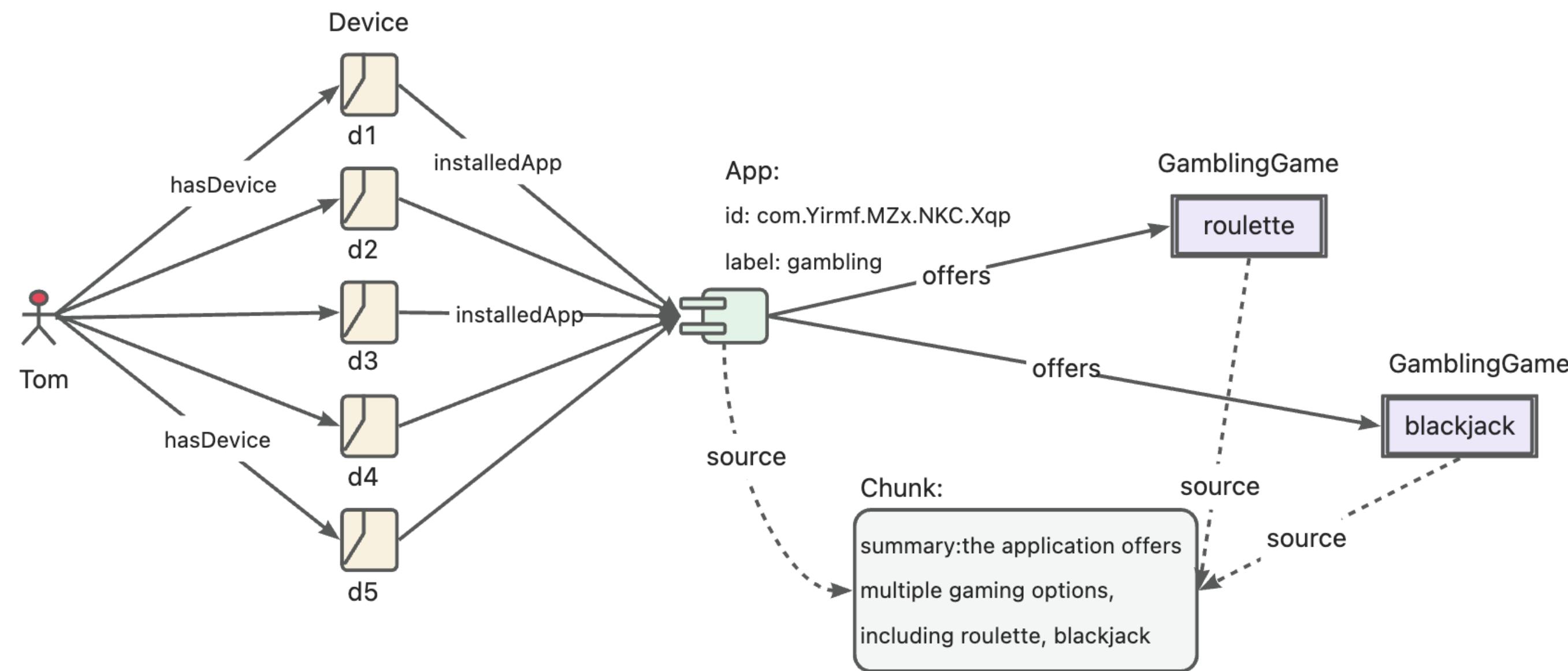
黑产图谱，通过提升黑产特征挖掘、完善黑产团伙刻画、推送线索助力司法部门线下打击等手段，降低体系内赌博电诈风险。其主要挑战在于：

- 领域专家规则与业务数据的统一建模
- 灵活的知识架构，以匹配适应多风险场景



基于SPG语义表示自定义专家规则

RC & KG_{fr} & KG_{cs}



define a RiskUser of gambling app rule

③ 定义"赌博App开发者" 规则

```

Define (s:Person)-[p:belongTo]->(o:`Tax0fRiskUser`/`Developer0fGamblingApp`) {
    Structure {
        (s)-[:developed]->(app:`Tax0fRiskApp`/`GamblingApp`)
    }
    Constraint {
    }
}

```

① 定义"赌博App" 规则

```

define riskAppTaxo rule
Define (s:App)-[p:belongTo]->(o:`Tax0fRiskApp`/`GamblingApp`) {
    Structure {
        (s)
    }
    Constraint {
        R1("risk label marked as gambling") s.riskMark like "%Gambling%"
    }
}

```

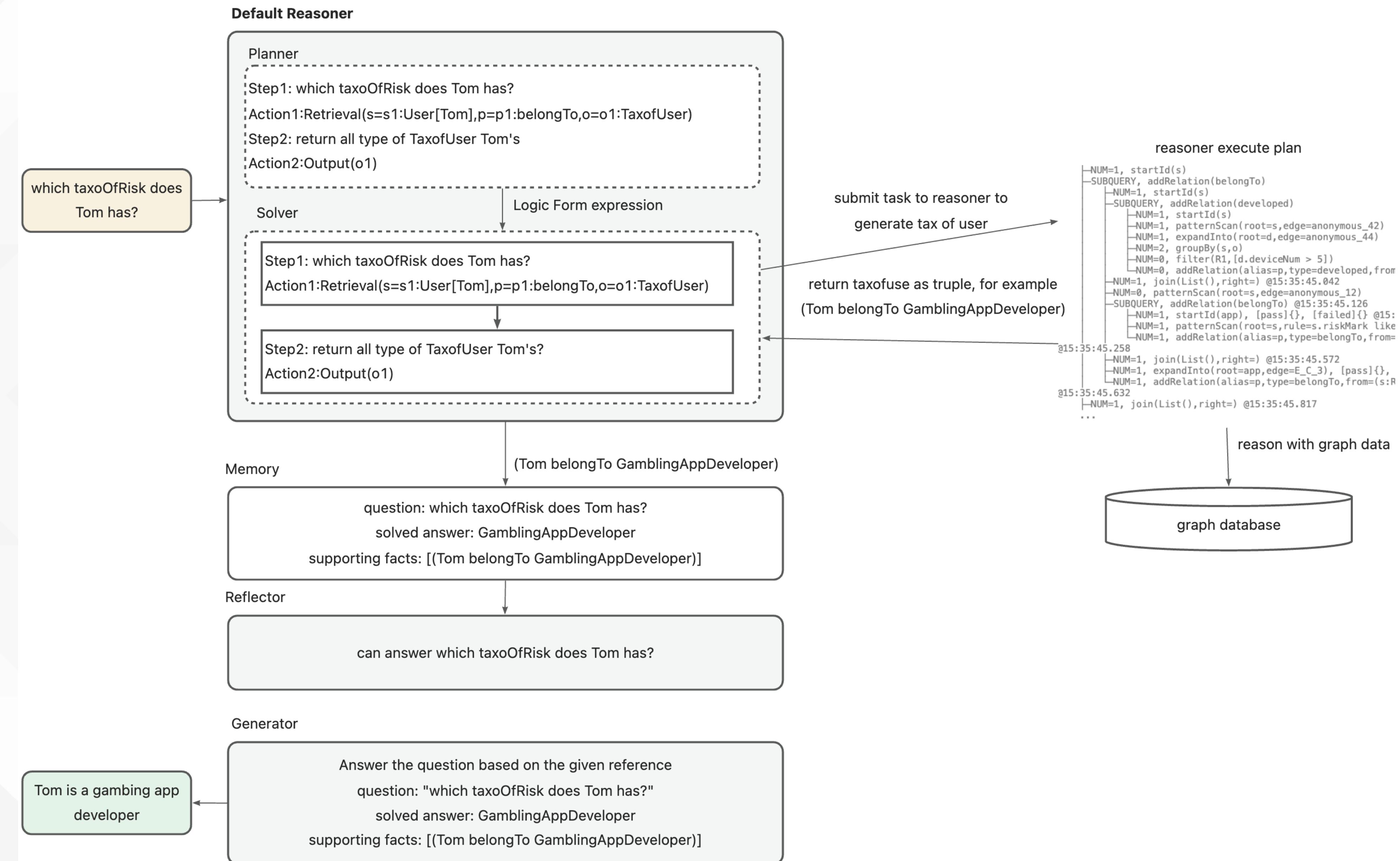
② 定义"App开发者" 规则

```

define app developper rule
Define (s:Person)-[p:developed]->(o:App) {
    Structure {
        (s)-[:hasDevice]->(d:Device)-[:install]->(o)
    }
    Constraint {
        deviceNum = group(s,o).count(d)
        R1("device installed same app"): deviceNum > 5
    }
}

```

Kag-Solver决策流程 & 结果



部署 & 使用

基于产品

开发者模式

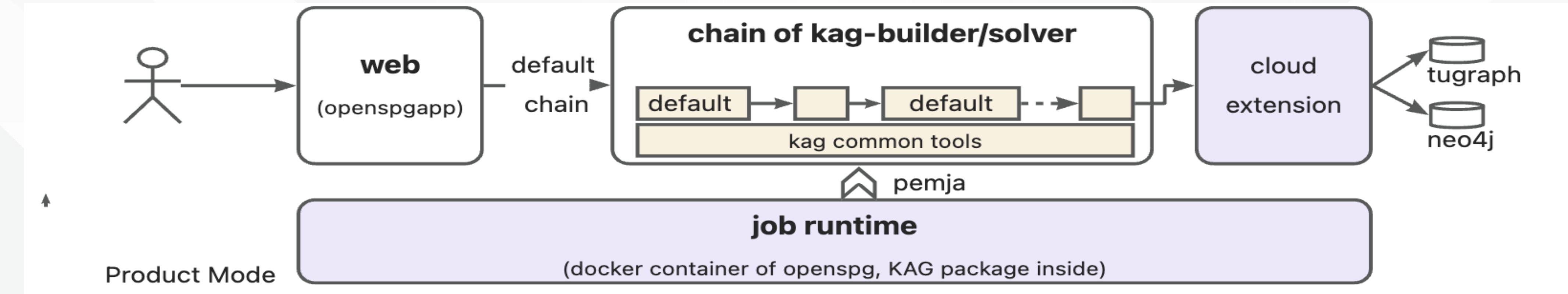
扩展&产品结合

部署&使用（产品模式）

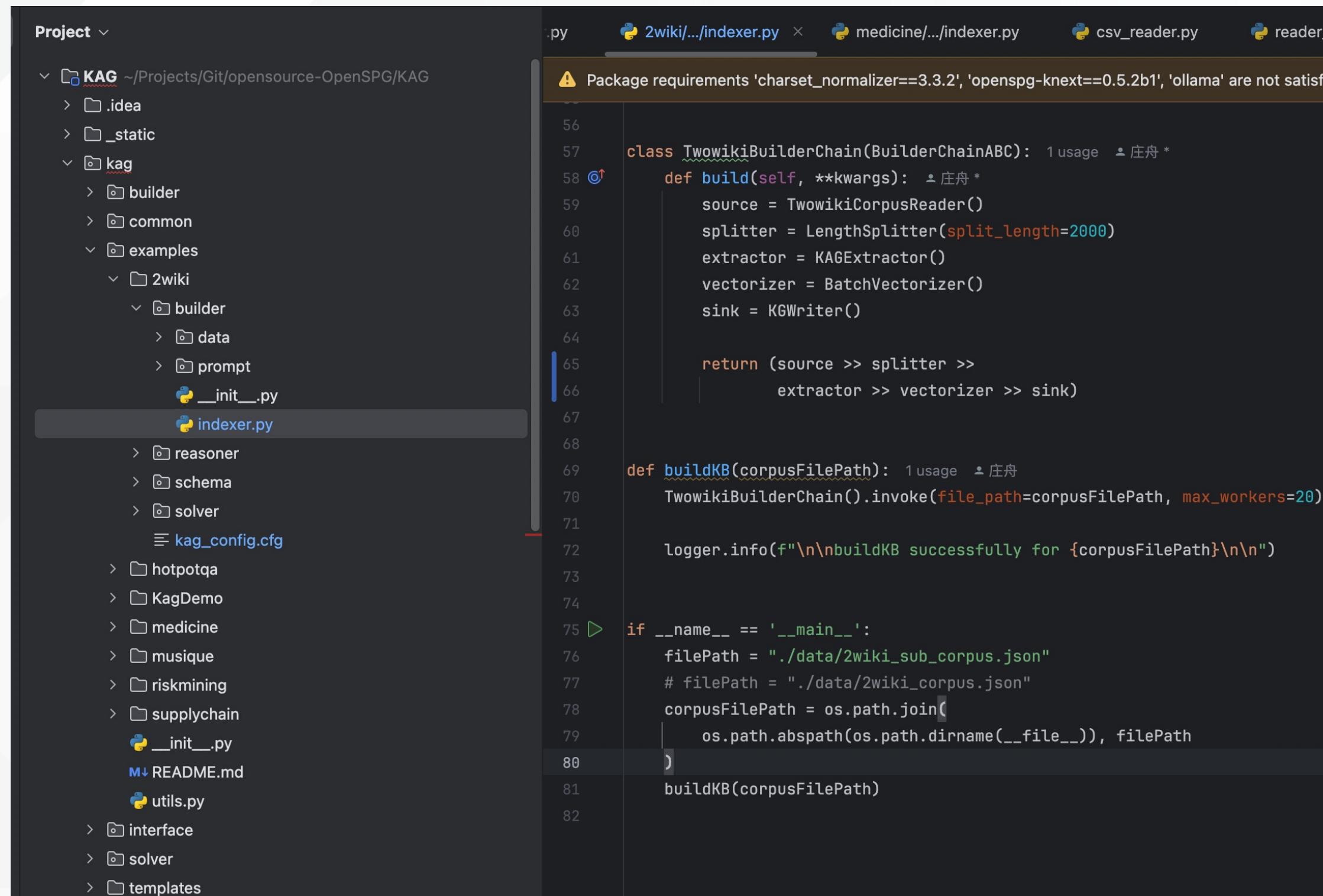


产品模式: readme

- kag default schema + default builder chain
- Openspg provides runtime env for Kag-builder chain through pemja
- kag-builder call API of openspg server for dumping extraction result to graph-engine



部署&使用（开发者模式）



```

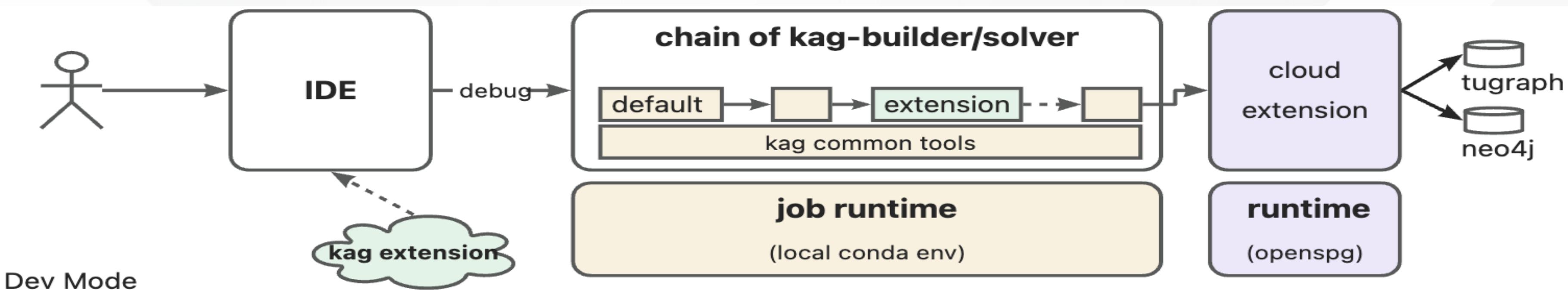
Project ▾
  KAG ~/Projects/Git/opensource-OpenSPG/KAG
    .idea
    _static
    kag
      builder
      common
      examples
        2wiki
          builder
            data
            prompt
            __init__.py
            indexer.py
        reasoner
        schema
        solver
        kag_config.cfg
      hotpotqa
      KagDemo
      medicine
      musique
      riskmining
      supplychain
        __init__.py
      README.md
      utils.py
    interface
    solver
    templates

indexer.py
  Package requirements 'charset_normalizer==3.3.2', 'openpgp-knext==0.5.2b1', 'ollama' are not satisfied
  ...
  56
  57   class TwoWikiBuilderChain(BuilderChainABC):
  58     def build(self, **kwargs):
  59       source = TwoWikiCorpusReader()
  60       splitter = LengthSplitter(split_length=2000)
  61       extractor = KAGExtractor()
  62       vectorizer = BatchVectorizer()
  63       sink = KGWriter()
  64
  65       return (source >> splitter >>
  66             extractor >> vectorizer >> sink)
  ...
  69   def buildKB(corpusFilePath):
  70     TwoWikiBuilderChain().invoke(file_path=corpusFilePath, max_workers=20)
  71
  72     logger.info(f"\n\nbuildKB successfully for {corpusFilePath}\n\n")
  73
  74
  75 if __name__ == '__main__':
  76   filePath = "./data/2wiki_sub_corpus.json"
  77   # filePath = "./data/2wiki_corpus.json"
  78   corpusFilePath = os.path.join(
  79     os.path.abspath(os.path.dirname(__file__)), filePath
  80   )
  81
  82   buildKB(corpusFilePath)

```

开发者模式: readme

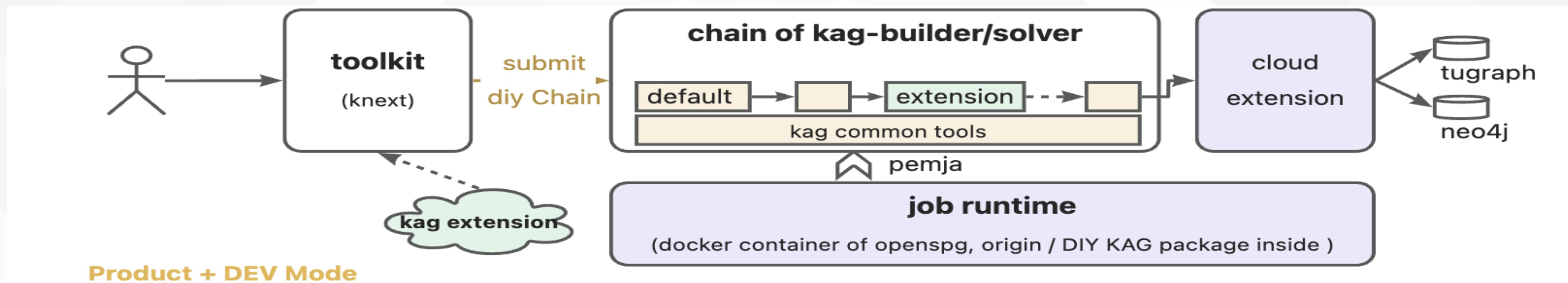
- developer customize schema & builder chain
- Local python IDE provides runtime env for Kag-builder chain
- kag-builder call API of openpgp for dumping extraction result to graph-engine



部署&使用（产品能力定制化）

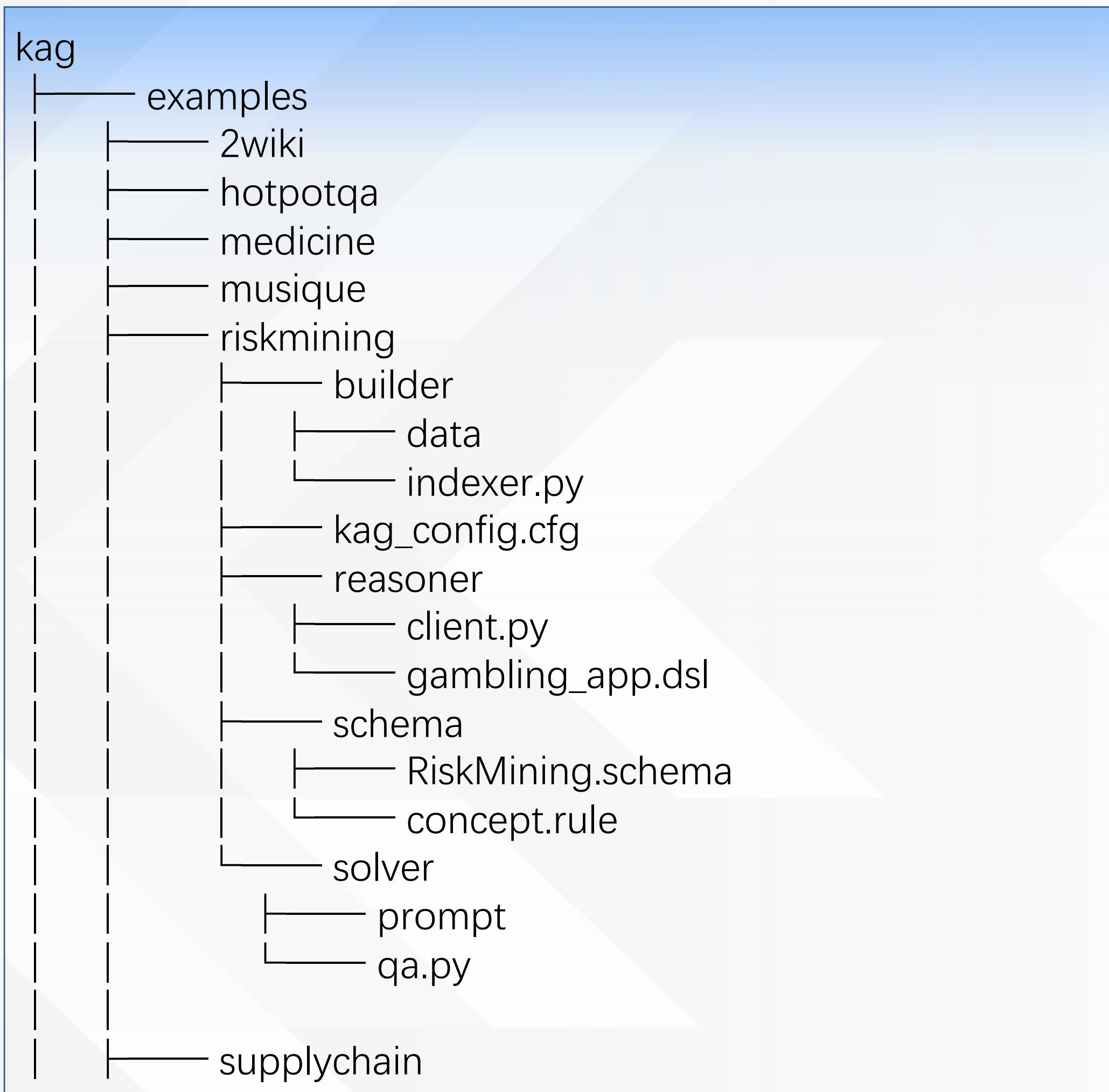
产品能力定制化: release soon

- 开发者模式完成自定义扩展的能力校验
- 任务提交到 openspg-server 执行



KAG 典型应用

Kag 内置案例



Framework	Model	HotpotQA		2WikiMultiHopQA		MuSiQue	
		EM	F1	EM	F1	EM	F1
NativeRAG [24, 23]	ChatGPT-3.5	43.4	57.7	33.4	43.3	15.5	26.4
HippoRAG [6, 23]	ChatGPT-3.5	41.8	55.0	46.6	59.2	19.2	29.8
IRCoT+NativeRAG	ChatGPT-3.5	45.5	58.4	35.4	45.1	19.1	30.5
IRCoT+HippoRAG	ChatGPT-3.5	45.7	59.2	47.7	62.7	21.9	33.3
IRCoT+HippoRAG	DeepSeek-V2	51.0	63.7	48.0	57.1	26.2	36.5
KAG (ours)	DeepSeek-V2	62.5	76.2	67.8	76.7	36.7	48.7

Table 9: The end-to-end generation performance of different RAG models on three multi-hop question answering datasets. Bold text indicates that the same base model performs best. NativeRAG and HippoRAG use single-step retrieval, while other models employ multi-step retrieval.

	Retriever	HotpotQA		2WikiMultiHopQA		MuSiQue	
		Recall@2	Recall@5	Recall@2	Recall@5	Recall@2	Recall@5
Single-step	BM25 [25]	55.4	72.2	51.8	61.9	32.3	41.2
	Contriever [26]	57.2	75.5	46.6	57.5	34.8	46.6
	GTR [27]	59.4	73.3	60.2	67.9	37.4	49.1
	RAPTOR [28]	58.1	71.2	46.3	53.8	35.7	45.3
	Proposition [29]	58.7	71.1	56.4	63.1	37.6	49.3
	NativeRAG [24, 23]	64.7	79.3	59.2	68.2	37.9	49.2
	HippoRAG [6, 23]	60.5	77.7	70.7	89.1	40.9	51.9
Multi-step	IRCoT + BM25	65.6	79.0	61.2	75.6	34.2	44.7
	IRCoT + Contriever	65.9	81.6	51.6	63.8	39.1	52.2
	IRCoT + NativeRAG	67.9	82.0	64.1	74.4	41.7	53.7
	IRCoT + HippoRAG	67.0	83.0	75.8	93.9	45.3	57.6
	KAG (ours)	72.8	88.8	65.4	91.9	48.5	65.7

Table 10: The performance of different retrieval models on three multi-hop question-answering datasets.

- Kag 示例的复现，可参考kag 用户手册: [kag user manual](#)
- 更多领域(教育、维修、工业、医疗、事理图谱)案例后续按计划发布

Kag蚂蚁应用

知识问答

分析写作

生活管家热点小报

12:51 ●
用过 助理 探索 ...
LPR下调，一年期五年期降10基点

7月22日，中国1年期和5年期LPR均下调10个基点，广州四大银行首套房贷利率降至3.2%。部分银行对特定购房者提供最低3.05%利率。中国人民银行行长潘功胜暗示，LPR可能在三季度进一步下调，以刺激房地产市场和降低融资成本。

深度解读

以下内容基于AI模型生成，请注意甄别

1 LPR调整详情
2024年7月22日，LPR迎来年内第二次调整，1年...

2 政策目的与影响
为稳增长促发展，调整LPR发布时间与逆回购利率...

3 经济背景与政策考量
全年经济增长目标设为5.0%左右，政策将适度加...

4 LPR调整机制分析
...

银行风险分析



银行股份有限公司 民营企业
行业：境内银行金融机构/城商行
业务余额：财富[代销合作机构]71.89亿元,消金事业群

LPR下调，一年期五年期降10基点

7月22日，中国1年期和5年期LPR均下调10个基点，广州四大银行首套房贷利率降至3.2%。部分银行对特定购房者提供最低3.05%利率。中国人民银行行长潘功胜暗示，LPR可能在三季度进一步下调，以刺激房地产市场和降低融资成本。

深度解读

以下内容基于AI模型生成，请注意甄别

1 LPR调整详情
2024年7月22日，LPR迎来年内第二次调整，1年...

2 政策目的与影响
为稳增长促发展，调整LPR发布时间与逆回购利率...

3 经济背景与政策考量
全年经济增长目标设为5.0%左右，政策将适度加...

4 LPR调整机制分析
...

保险事件市况



12:53 ●
用过 助理 探索 ...
助理

孕期口腔问题影响宝宝健康

北京社保转移地的相关信息如下：

城乡居民基本养老保险关系转移接续申请办理地点：

- 北京市朝阳区政务服务中心人力社保分中心：北京市朝阳区管庄周家井世通国际大厦E座三层、四层社会保险“一窗通办”综合业务经办大厅窗口。
- 北京市朝阳区政务服务中心：北京市朝阳区霄云路霄云里1号二层综合窗口。

工伤职工工伤保险关系转移办理地点：

- 北京市朝阳区政务服务中心人力社保分中心。

其他相关事项：

北京市—朝阳区军地养老保险关系转移接续申请—未就业随军配偶、失业保险关系跨省转移

政务办事问答

12:53 ●
用过 助理 探索 ...
助理

孕期口腔问题影响宝宝健康

分享 评论 赞

健康创意坊 · 7月23日

激光笔致5岁男孩失明，家长需警惕

分享 评论 赞

健康创意坊 · 7月22日

泰州市柳园晨练点：丰富多彩的晨练活动

分享 评论 赞



后续迭代计划



OpenSPG-KAG 后续迭代计划

模块	能力项	发布节奏
产品交互优化	<ul style="list-style-type: none">1、自定义schema2、图分析查询3、前端开源	
Kag-builder 优化	<ul style="list-style-type: none">1、增量更新 完善2、性能优化3、垂直领域对齐及多种文档结构4、分布式	参考OpenSPG官网发布计划
Kag-solver 优化	<ul style="list-style-type: none">1、logical-Form 算子完备性2、多任务问答3、类o1	
Kag-model	<ul style="list-style-type: none">1、kag model 发布	

联系我们

KAG: <https://github.com/OpenSPG/KAG>

使用文档: [ReadMe](#)

(欢迎关注star)



(技术交流群)

Thanks & QA