南京大学
人工智能学院
School of Artificial Intelligence, Nanjing University

第三部分：通信与合作

章宗长

2023年3月21日

回顾

- 我们先前关注的内容——Agent的智能自治
  - 智能Agent
  - 演绎推理Agent
  - 实用推理Agent
  - 反应式和混合式Agent

- 第3~5部分关注：Agent的社会性设计
  - 如何构造能与其他Agent交互（通信、合作、协商等）的Agent？

内容安排

3.1 相互理解的Agent

3.2 通信

3.3 合作

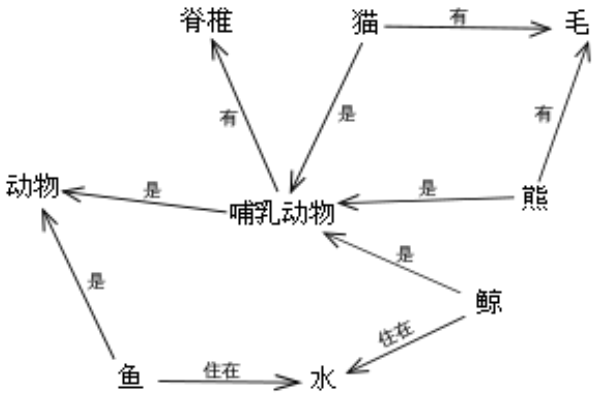
3.4 实践：使用Jason解释器的多Agent编程

相互理解的Agent

- **本体论（Ontology）基础**
- 本体描述语言
- 构建本体

什么是本体（Ontology）

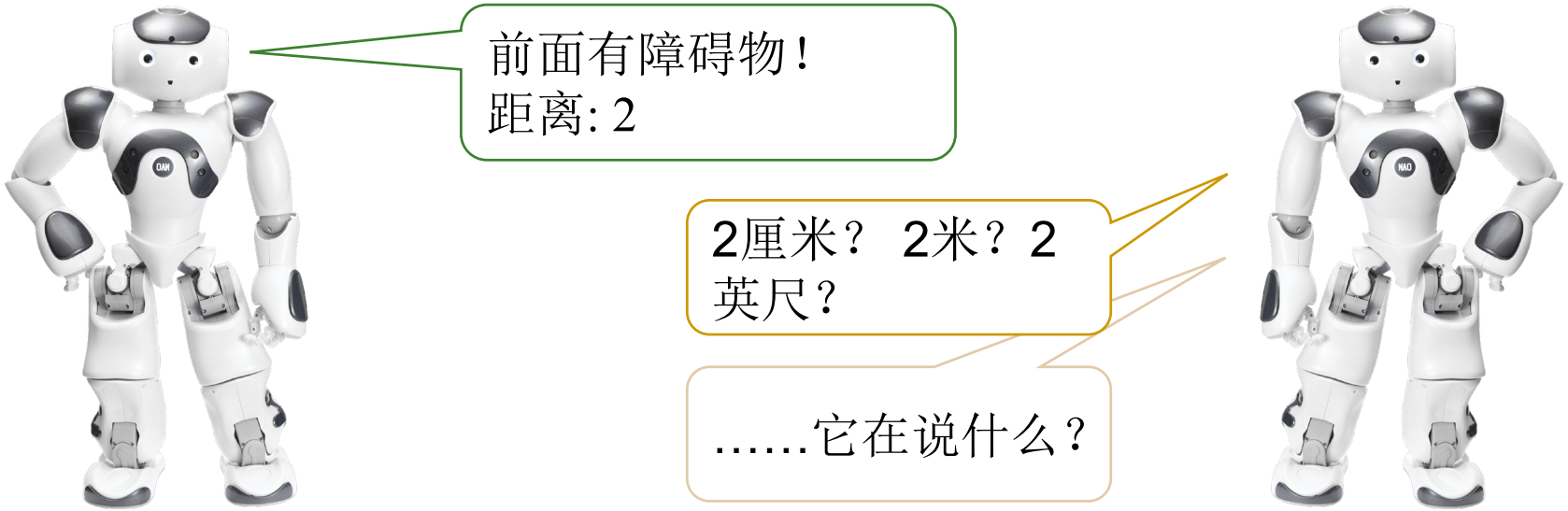
- 来源：哲学中的本体论
  - 探讨存在本身，即一切现实事物的基本特征
  - 是形而上学的基本分支
- 信息科学中：结构化的术语集
  - 特定领域中存在的对象类型或概念及其属性和相互关系
  - 应用领域：
    - 语义网
    - 软件工程
    - 图书馆学
    - ……



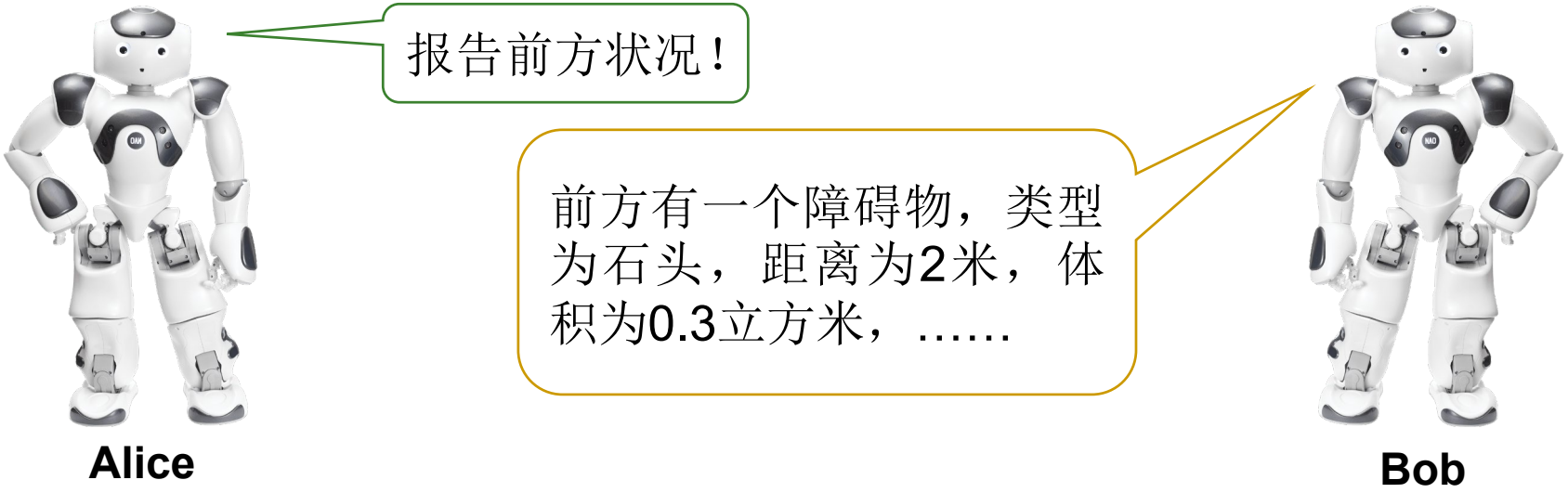
简单的本体示例

本体论与多Agent系统

- 多Agent系统为什么需要本体论？
  - 如果Agent需要通信，他们的通信内容就必须遵守相同的“术语”
  - 本体提供了一种特殊类型的术语集，具有结构化的特点

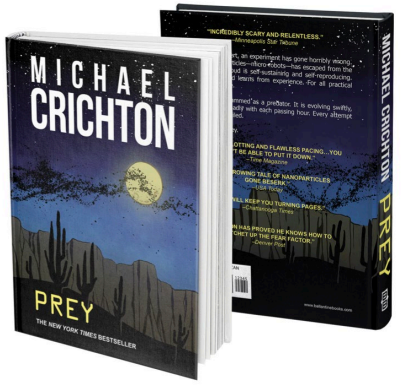


如何让Agent理解信息



- Alice可能得知的信息：
  - Bob前方有一块石头，石头是“障碍物”的一个实例
  - 石头有属性“距离”
  - 石头有属性“体积”

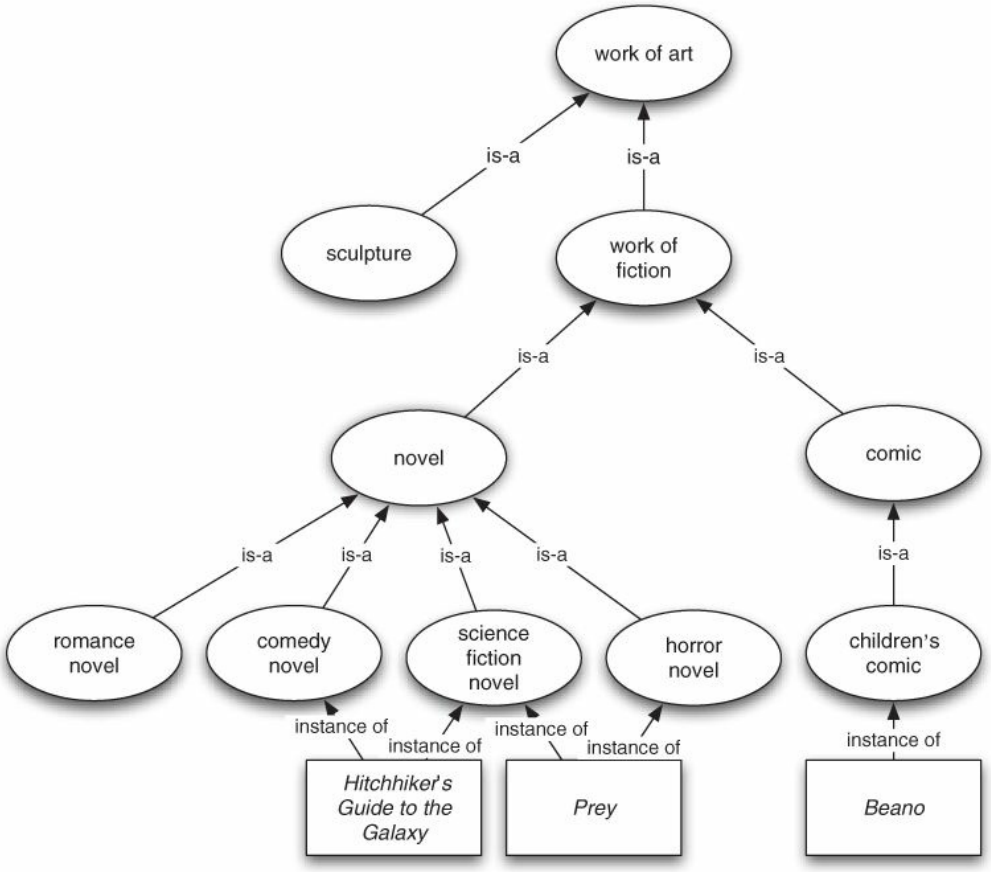
表示本体



- 假设有一个这样的对话：
  - Alice：你读过《Prey》吗？
  - Bob：没有。这是什么？
  - Alice：一本科幻小说，有点恐怖，它展现了多Agent系统不受控制的世界。
- Alice传递了关于《Prey》的什么？
  - 是什么：小说、科幻小说、恐怖小说
  - 有什么：内容关于多Agent系统
- Alice假设Bob知道什么是“小说”、“科幻小说”、“恐怖”、“Agent”……
- 换而言之，Alice在Bob已有知识下定义新的术语《Prey》

表示本体（续）

- 由于Bob具有相关的知识，所以这次交流成功了
- 不仅如此，Bob可能还知道……
  - 小说是某种文学作品
  - 文学作品不仅包含小说，还有喜剧
  - 文学是某种艺术形式，而雕塑也是某种艺术形式
  - 《银河系漫游指南》也是一本科幻小说
  - ……

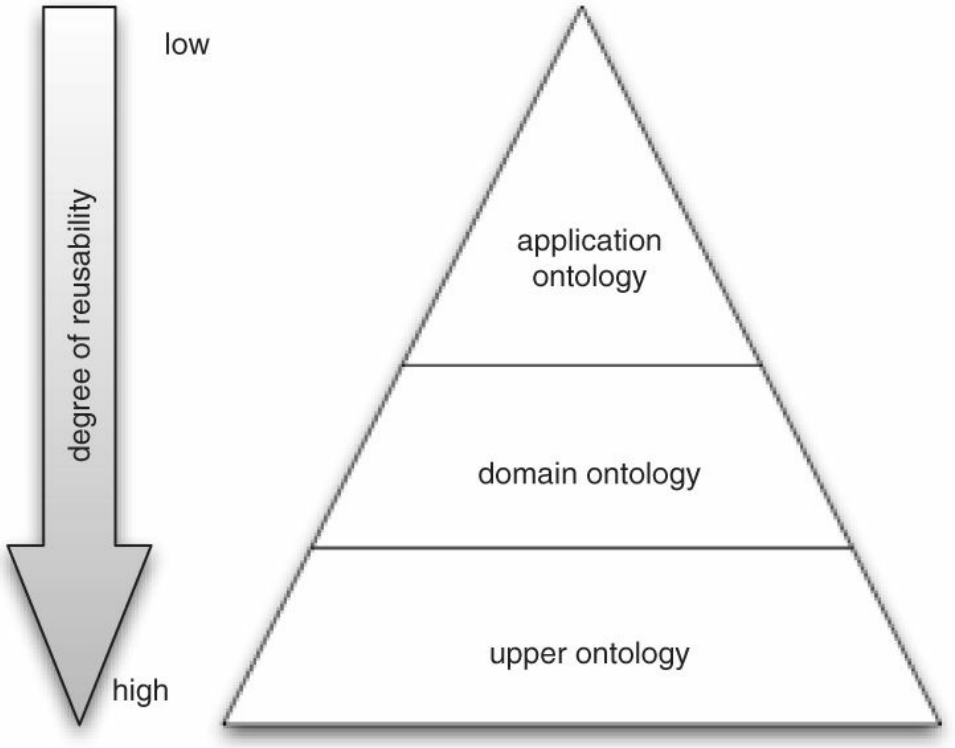


- Bob会将收到的关于《Prey》的信息加入自己的知识中

本体描述层次

- 按照可复用性从低到高将本体划分为三个层次
  - 应用本体
  - 领域本体
  - 上层本体

- 一个本体越具体，它的可复用性就越低



相互理解的Agent

- 本体论（Ontology）基础
- **本体描述语言**
- 构建本体

XML – 可扩展标记语言

- XML（eXtensible Markup Language）
  - 本身不是为本体而设计，但能够定义一些简单的本体
- XML的设计旨在扩展HTML语言格式
  - HTML：缺少语义标记
  - 能够定义新的标记标签，而并非一定需要遵循HTML中的固定格式

(a) Plain HTML

```
<ul>
   <li><em>Music</em>,
      <b>Madonna<b>,
      USD12<br><p>
   <li><em>Get Ready</em>,
      <b>New Order</b>,
      USD14<br><p>
</ul>
```

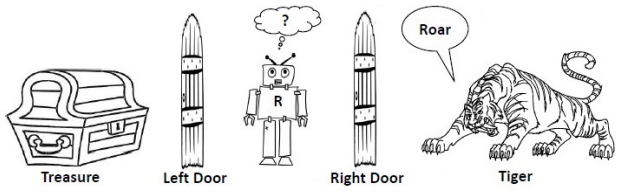
(b) XML

```
<catalogue>
   <product type="CD">
      <title>Music</title>
      <artist>Madonna</artist>
      <price currency="USD">12</price>
   </product>
   <product type="CD">
      <title>Get Ready</title>
      <artist>New Order</artist>
      <price currency="USD">14</price>
   </product>
</catalogue>
```

利用XML定义本体

Tiger.pomdpx

```
<?xml version='1.0' encoding='ISO-8859-1'?>


<pomdpx version='0.1' id='autogenerated'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xsi:noNamespaceSchemaLocation='pomdpx.xsd'>


<Description>This is an auto-generated POMDPX file</Description>
<Discount>0.95</Discount>

<Variable>

<StateVar vnamePrev="state_0" vnameCurr="state_1" fullyObs="false">
<ValueEnum>tiger-left tiger-right</ValueEnum>
</StateVar>

<ObsVar vname="obs_sensor">
<ValueEnum>obs-left obs-right</ValueEnum>
</ObsVar>

<ActionVar vname="action_agent">
<ValueEnum>listen open-left open-right</ValueEnum>
</ActionVar>

<RewardVar vname="reward_agent"/>
</Variable>


<InitialStateBelief>
<CondProb>
<Var>state_0</Var>
<Parent>null</Parent>
<Parameter type = "TBL">
<Entry>
<Instance>-</Instance>
<ProbTable>0.5 0.5</ProbTable>

</Entry>
</Parameter>
</CondProb>
</InitialStateBelief><StateTransitionFunction>

<CondProb>
<Var>state_1</Var>
<Parent>action_agent state_0</Parent>
<Parameter type = "TBL">
<Entry>
<Instance>listen - -</Instance>
<ProbTable>identity</ProbTable></Entry>
<Entry>
<Instance>open-left * *</Instance>
```



老虎问题——POMDP任务

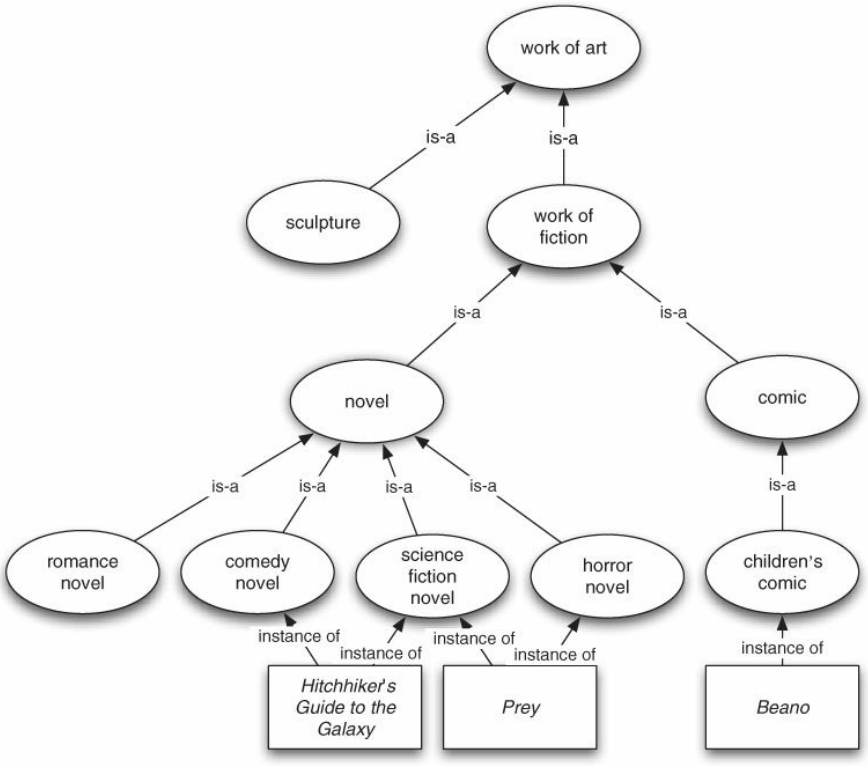
```
<ProbTable>0.5</ProbTable></Entry>
<Entry>
<Instance>open-right * *</Instance>
<ProbTable>0.5</ProbTable></Entry>
</Parameter>
</CondProb>
</StateTransitionFunction>

<ObsFunction>

<CondProb>
<Var>obs_sensor</Var>
<Parent>action_agent state_1</Parent>
<Parameter type = "TBL">
<Entry>
<Instance>listen - -</Instance>
<ProbTable>0.85 0.15 0.15 0.85</ProbTable></Entry>
<Entry>
<Instance>open-left * *</Instance>
<ProbTable>0.5</ProbTable></Entry>
<Entry>
<Instance>open-right * *</Instance>
<ProbTable>0.5</ProbTable></Entry>
</Parameter>
</CondProb>
</ObsFunction>

<RewardFunction>

<Func>
<Var>reward_agent</Var>
<Parent>action_agent state_0</Parent>
<Parameter type = "TBL">
<Entry>
<Instance>listen *</Instance>
<ValueTable>-1</ValueTable></Entry>
<Entry>
<Instance>open-left tiger-left</Instance>
<ValueTable>-100</ValueTable></Entry>
<Entry>
<Instance>open-left tiger-right</Instance>
<ValueTable>10</ValueTable></Entry>
<Entry>
<Instance>open-right tiger-left</Instance>
<ValueTable>10</ValueTable></Entry>
<Entry>
<Instance>open-right tiger-right</Instance>
<ValueTable>-100</ValueTable></Entry>
</Parameter>
</Func>
</RewardFunction></pomdpx>
```

利用XML定义本体（续）

Maven是Java的一个开源项目管理工具，其项目对象模型（POM, Project Object Model）利用名为 pom.xml 的XML文件定义项目的基本信息，包含项目的构建方法、依赖等等。

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <!-- The Basics -->
  <groupId>...</groupId>
  <artifactId>...</artifactId>
  <version>...</version>
  <packaging>...</packaging>
  <dependencies>...</dependencies>
  <parent>...</parent>
  <dependencyManagement>...</dependencyManagement>
  <modules>...</modules>
  <properties>...</properties>

  <!-- Build Settings -->
  <build>...</build>
  <reporting>...</reporting>

  <!-- More Project Information -->
  <name>...</name>
  <description>...</description>
  <url>...</url>
  <inceptionYear>...</inceptionYear>
  <licenses>...</licenses>
  <organization>...</organization>
  <developers>...</developers>
  <contributors>...</contributors>

  <!-- Environment Settings -->
  <issueManagement>...</issueManagement>
  <ciManagement>...</ciManagement>
  <mailingLists>...</mailingLists>
  <scm>...</scm>
  <prerequisites>...</prerequisites>
  <repositories>...</repositories>
  <pluginRepositories>...</pluginRepositories>
  <distributionManagement>...</distributionManagement>
  <profiles>...</profiles>
</project>
```

OWL – 网络本体语言（Web Ontology Language）



Bob的知识构成的本体

```
1. Ontology(
2.    Class(WorkOfArt partial owl:Thing)
3.    Class(Sculpture partial WorkOfArt)
4.    Class(WorkOfFiction partial WorkOfArt)
5.    Class(Novel partial WorkOfFiction)
6.    Class(Comic partial WorkOfFiction)
7.    Class(RomanceNovel partial Novel)
8.    Class(ComedyNovel partial Novel)
9.    Class(ScienceFictionNovel partial Novel)
10.   Class(HorrorNovel partial Novel)
11.   Class(ChildrensComic partial Comic)
12.   DisjointClasses(Sculpture WorkOfFiction)
13.   ObjectProperty(author
14.      domain(Novel) range(Person))
15.   ObjectProperty(content
16.      domain(Novel) range(String))
17.   Individual("Hitchhiker's Guide to the Galaxy"
18.      type(ScienceFictionNovel)
19.      value(author "Douglas Adams")
20.      value(content "Far out in the uncharted
21.      backwaters of the unfashionable end of
22.      the Western Spiral Arm of the Galaxy..."))
23.   Individual("Prey"
24.      type(intersectionOf(HorrorNovel ScienceFictionNovel))
25.      value(author "Michael Crichton")
26.      value(content "Things never turn out
27.         the way you think they will...."))
28.   Individual("Beano"
29.      type(ChildrensComic))
30.   DifferentIndividuals(
31.      "Hitchhiker's Guide to the Galaxy"
32.      "Prey")
33.)
```

KIF – 知识交换格式（Knowledge Interchange Format）语言

- KIF最初开发的目的是作为特定领域的公共语言
  - 表示消息的内容，而非是消息本身
- KIF严格基于一阶逻辑
- 使用KIF，Agent可以表示
  - 一个领域中事物的性质
    - 例：Michael是一个素食主义者
  - 一个领域中事物之间的关系
    - 例：Michael和Janine结婚了
  - 一个领域中的一般性质
    - 例：每个人都有母亲

KIF的一些例子

- 使用KIF表达式进行断言：

```
(= (temperature m1) (scalar 83 Celsius))
```

m1的温度是83摄氏度

- 使用KIF表达式定义概念：

```
(defrelation bachelor (?x) :=
  (and (man ?x)
    (not (married ?x))))
```

如果一个对象是男人并且没有结婚，则这个对象是单身

- 使用KIF表达式定义关系：

```
(defrelation person (?x) :=> (mammal ?x))
```

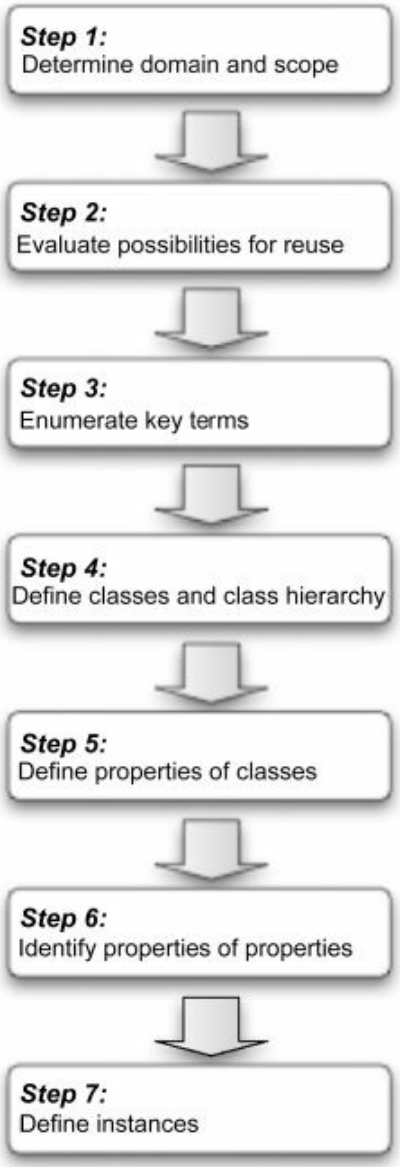
任何具有人的性质的个体也具有哺乳动物的性质

相互理解的Agent

- 本体论（Ontology）基础
- 本体描述语言
- **构建本体**

构建本体的步骤

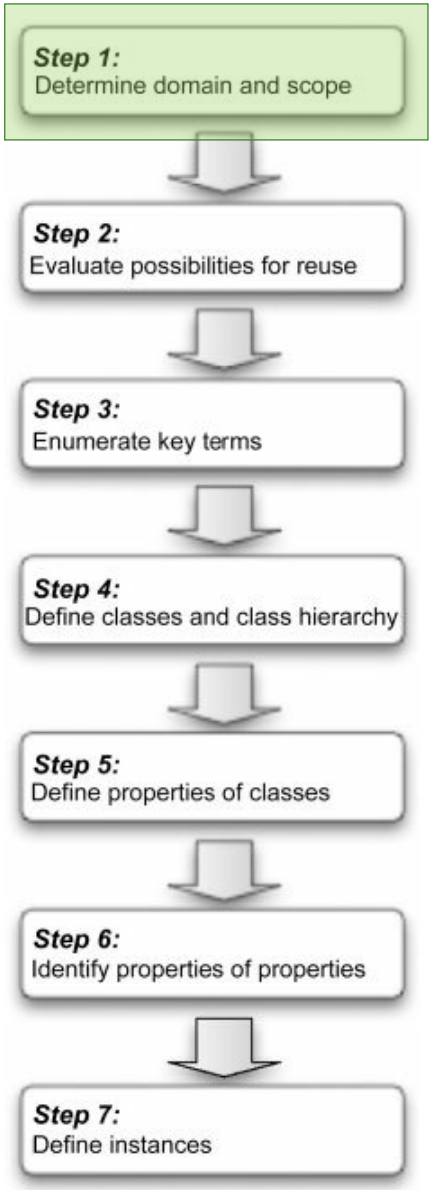
- 尽管我们可以用不同的描述语言来定义本体，但在构建一个本体时，我们可以遵守[Noy and McGuinness, 2004]提出的方法论，分为7个步骤：
  - 确定领域和范围
  - 考虑复用
  - 列举关键术语
  - 定义类和类的层次结构
  - 定义属性
  - 定义属性上的约束
  - 创建实例



确定领域和范围

- 在开发软件时，要理解用户有哪些需求

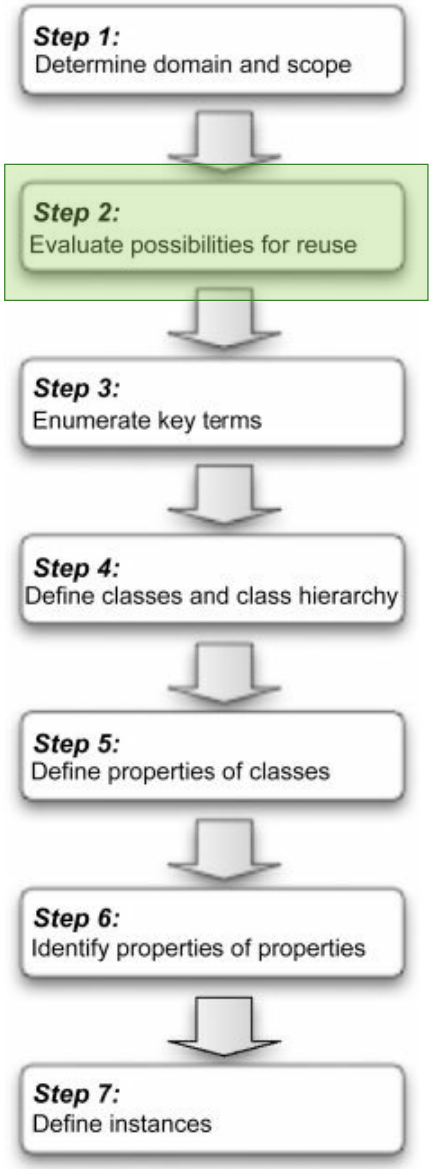
- 在开发本体时，我们也要回答类似的问题：
  - 我设计的本体涵盖哪个领域？
  - 我设计的本体用来回答什么问题？
  - 怎样使用这个本体？



考虑复用

- 复用在本体开发中是极其重要的，许多预定义好的本体能够简化我们的实现

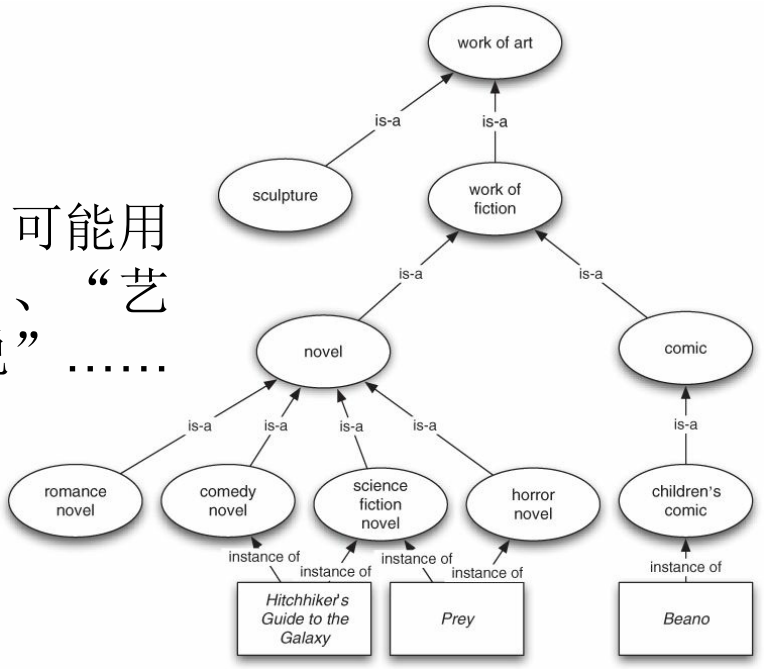
- 一些已公开的本体实例
  - 氨基酸本体：关于氨基酸及其性质的小型本体
  - 基本形式化本体：为科学技术研究工作提供支持的形式化上层本体
  - 细胞周期本体：用于表达细胞周期的应用本体
  - ……

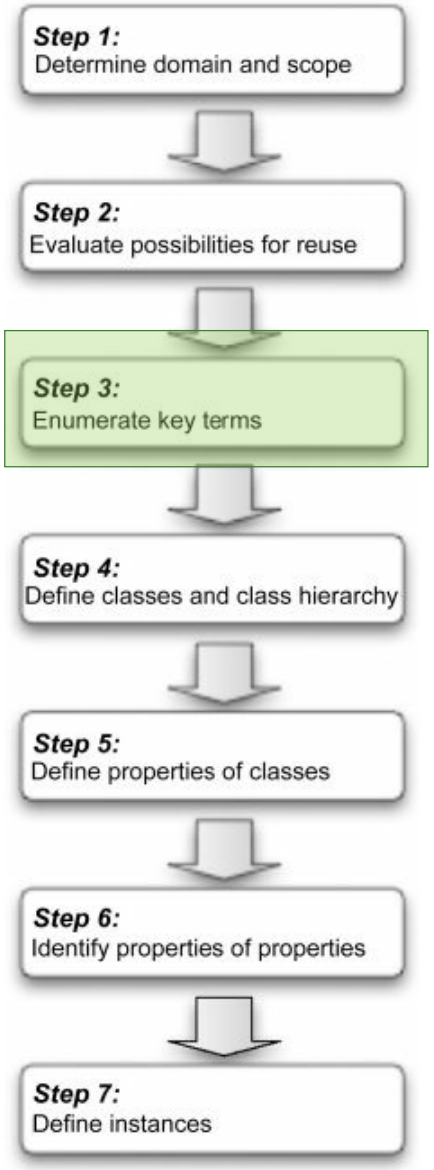


列举关键术语

- 可以通过“头脑风暴”，简单地将与所开发本体的领域关联的术语先列举出来，然后筛选出与需求相关的概念和名词

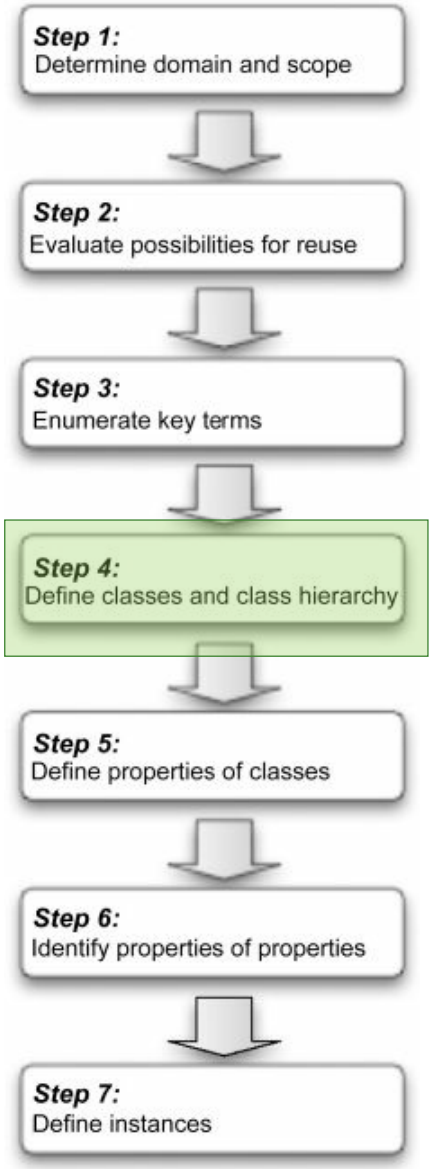
当领域是科幻小说时，可能用到“小说”、“文学”、“艺术作品”、“喜剧小说”……





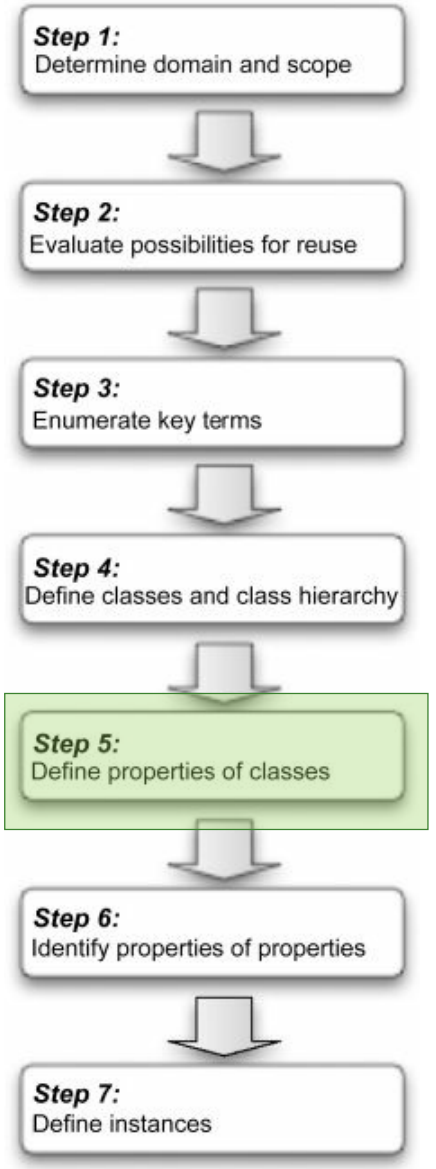
定义类和类的层次结构

- 根据对领域知识的理解，定义类和类的层次结构
- 在定义类时，要规避一些常见错误：
  - 不要混淆类和属性
    - 例：“人”和“年龄”应当分别是类和属性，而不是两个类
  - 不要定义琐碎的类
    - 例：定义一个类为“有黑色头发和蓝色眼睛的人”是不合理的
- 类的定义应当是有层次的，可遵循自顶向下或者自底向上的逻辑



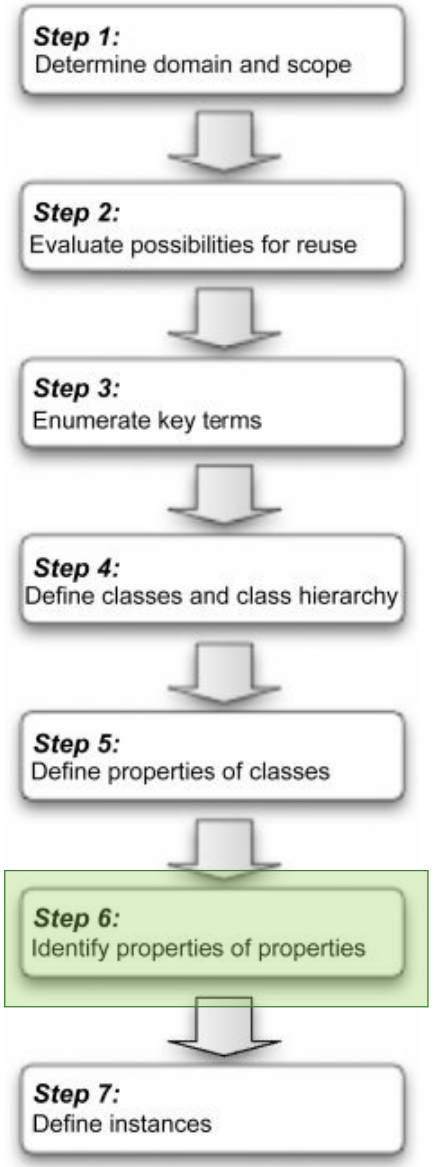
定义属性

- 一个类可以具有以下形式的属性
  - 固有属性
    - 与类的本质相关的属性，如人的身高、体重等
  - 外在属性
    - 能够关联到一个物体上的具体属性，如名字、身份号码等等
  - 组件
    - 如果一个物体是结构化的，那么就适于使用组件表示
  - 关系
    - 例：一个“作者”关系应当连接到两个类“作品”与“人”



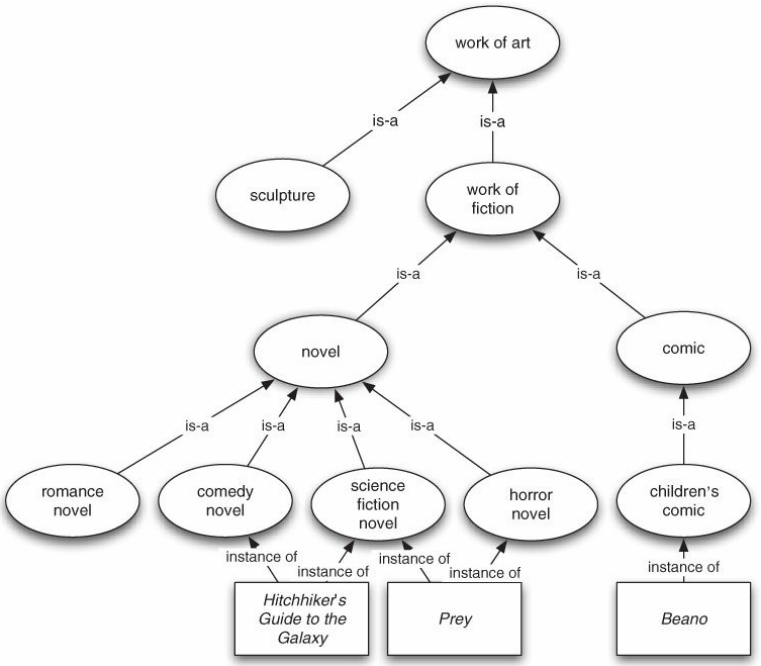
定义属性约束

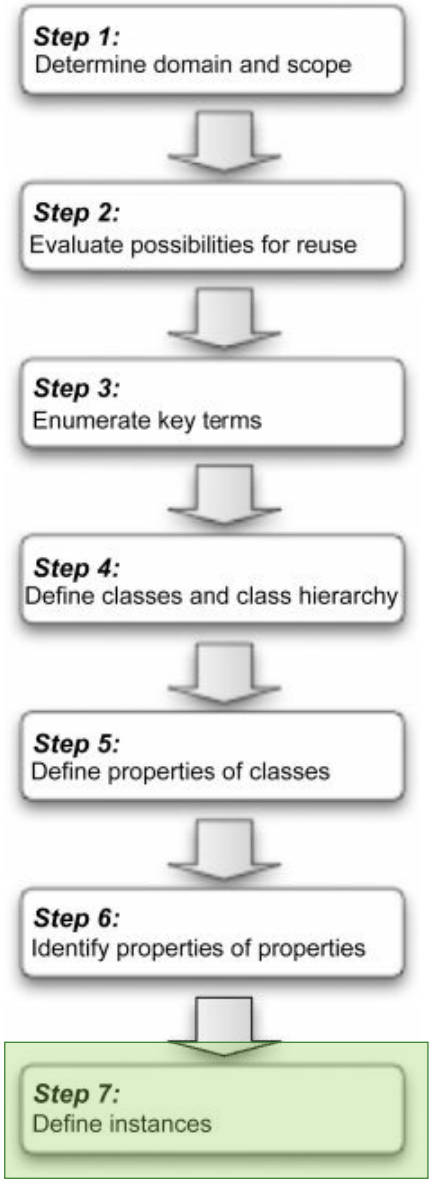
- 一个类的每个属性可能有不同的约束条件：
  - 势约束（cardinality constraints）
    - 如一个人只有一个出生日期
  - 类别约束（type constraints）
    - 如一个人的年龄是一个正整数
  - 范围约束（range constraints）
    - 如一个类“人”拥有“母亲”属性，那么该属性的类型不应当是“人”，而是范围更小的“女人”
  - 领域约束（domain constraints）
    - 如属性“作者”对于类“电影”而言是不合适的



创建实例

- 尽量选择具体的类
  - 例：《银河系漫游指南》是一本科幻小说，我们就应当令其为“科幻小说”类的实例，而非创建为“小说”的实例





小结

- 本体论基础
  - 本体：结构化的术语集
  - 对于多Agent系统，本体提供了Agent相互理解的基础
  - 本体表示方法
  - 本体描述层次
- 本体描述语言
  - XML、OWL、KIF
- 如何构建本体
  - 遵循构建本体的方法论（7个步骤）

内容安排

3.1 相互理解的Agent

3.2 通信

3.3 合作

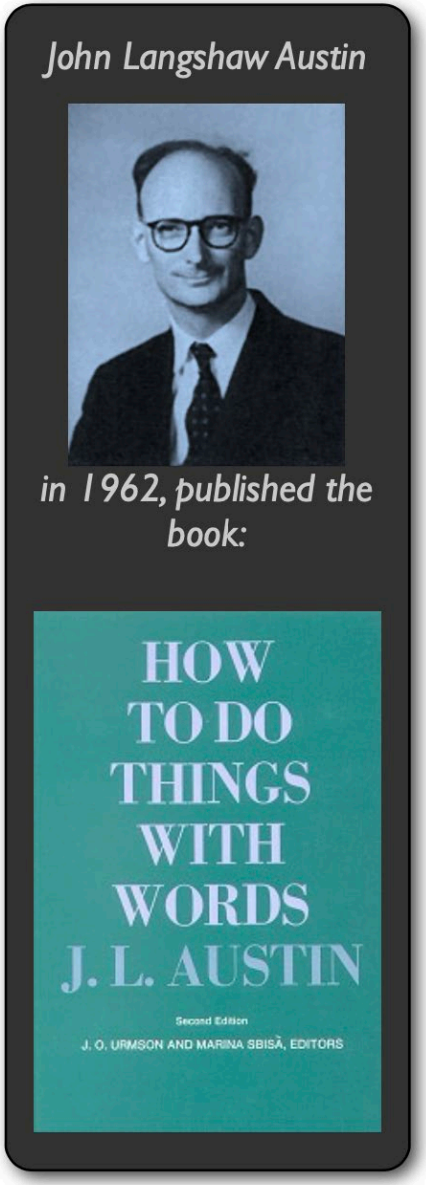
3.4 实践：使用Jason解释器的多Agent编程

通信

- 言语行为
- 通信方式
- 基于消息的Agent 通信语言：KQML
- 基于消息的Agent 通信语言：FIPA ACL

言语行为（Speech Act）

- 在多Agent系统研究领域，言语行为通常指Agent间的交互行为
- 言语行为的特点
  - 传达某种交互的意图和内容
  - 影响其他Agent的内部状态以及相应的行为实施
- 言语行为理论
  - 用于解释和分析人类通信和交互的高层理论框架
  - 研究言语行为的组成以及其语义和语用
  - 把通信建模成可以改变通信参与者的思维状态的动作



Austin的言语行为理论

- 言语行为不仅仅是为了描述状态，而是为了完成一定的行为，如：
  - 通知（Inform）：门被关了。
  - 请求（Require）：请关门！
  - 允诺（Promise）：我将关门。
  - 允许（Permit）：你可以关门。
  - 禁止（Prohibit）：你不准关门。
  - 声明（Declare）：我现在宣布你们为夫妻。
- Austin将成功完成动作所需要的条件称为合适措辞条件：
  - 准备条件（Preparatory condition）：必须有接受行为动词的常规过程，并且在这一过程中环境和人员必须是恰当的
  - 执行条件（Execution condition）：过程必须正确、完整地执行
  - 真诚条件（Sincerity condition）：行为必须是诚实的

言语行为理论三分说

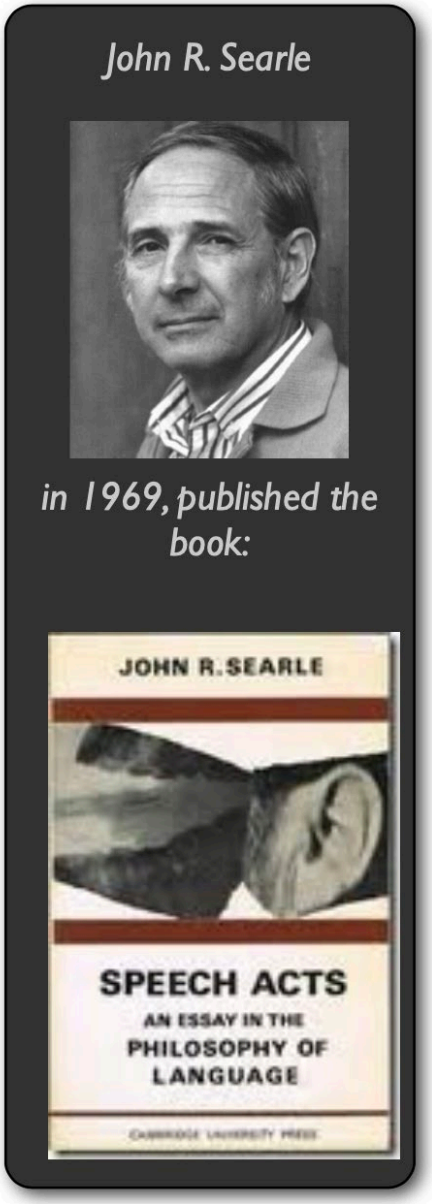
- Austin认为人与人之间的任何会话实际上产生了三个概念互有区别的活动：
  - 以言指事（Locutionary Act）：言语行为的物理动作
  - 以言行事（Illocutionary Act）：言者传递给听者的意图
  - 以言成事（Perlocutionary Act）：言语行为的结果
- 以言指事并不一定必然包含以言行事

  一个人自言自语时，有以言指事但没有传递言者的意图
- 以言行事并不一定必然包含以言成事

  一个Agent向另一个Agent发出服务请求，但接收方Agent可能由于某些原因拒绝为请求方Agent提供相应的服务

Searle的言语行为理论

- Searle进一步区分出说者和听者之间言语行为执行成功必须具有的几个特性：
  - 正常的I/O条件
    - 听者可以听到请求
    - 行为是在正常的环境下执行的
  - 准备条件
    - 为了使说者正确地选择行为，世界中什么必须为真
  - 诚实条件
    - 当说者并不真的希望执行ACTION时，动作会发生不诚实的行为



言语行为（续）

- Searle指出言语行为有5种主要类型：
  - 描写（Representatives）：表达说话者承认命题为真
    - 例：通知
  - 指导（Directives）：说话者试图使听者开始做某件事
    - 例：请求
  - 委托（Commissives）：向说话者承诺一系列动作
    - 例：保证
  - 表达（Expressives）：展示某个心理状态（态度）
    - 例：感谢
  - 宣布（Declartions）：使习惯的现状发生某种变化
    - 例：宣布战争

基于规划的言语行为理论

- 如果要求可以规划如何自动实现目标的系统与人或者其他自治Agent交互，则规划必须包括言语行为
- 如何表示言语行为特性，使得规划系统可以对其进行推理？

> 通过在规划系统中把言语行为建模为根据说者和听者的信念和目标定义的操作符。这样，以与物理动作相同的方式处理言语行为。
>
> (Cohen and Perrault, 1979)

- Cohen和Perrault选择的形式体系
  - STRIPS表示方法，其中动作特性通过前件和后件刻画

言语行为Request

- Request的目的：说者使听者执行某个动作
  - 两个前件
    - Cando.pr
    - Want.pr

Request(*S*, *H*, *α*)		
Preconditions	Cando.pr	(*S BELIEVE* (*H CANDO α*)) ∧ (*S BELIEVE* (*H BELIEVE* (*H CANDO α*)))
	Want.pr	(*S BELIEVE* (*S WANT requestInstance*))
Effect		(*H BELIEVE* (*S BELIEVE* (*S WANT α*)))

- Cando.pr：可以做的前提条件
  - 说者（S）必须相信听者（H）可以完成这个动作
  - 说者（S）必须相信听者（H）相信自己可以完成这个动作
- Want.pr：希望做的前提条件
  - 说者（S）必须相信他想要request的动作被执行
- Effect：听者相信说者相信他想要某个动作被执行

Cohen等的言语行为理论

- Cohen和Perrault在基于规划的言语行为理论中定义的其他一些言语行为

CauseToWant(A_1, A_2, α)		
Preconditions	Cando.pr	(A_1 *BELIEVE* (A_2 *BELIEVE* (A_2 *WANT* α)))
	Want.pr	$\times$
Effect		(A_1 *BELIEVE* (A_1 *WANT* α))

Inform(S, H, φ)		
Preconditions	Cando.pr	(S *BELIEVE* φ)
	Want.pr	(S *BELIEVE* (S *WANT informInstance*))
Effect		(H *BELIEVE* (S *BELIEVE* φ))

Convince(A_1, A_2, φ)		
Preconditions	Cando.pr	(A_1 *BELIEVE* (A_2 *BELIEVE* φ))
	Want.pr	$\times$
Effect		(A_1 *BELIEVE* φ)

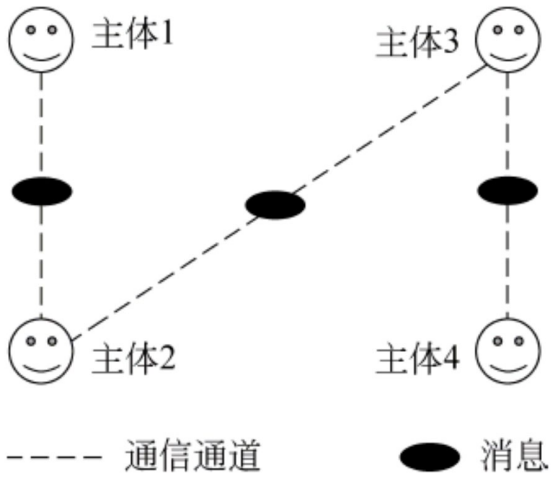
- Cohen和Levesque进一步提出了作为理性动作的言语行为
  - 把言语行为建模成理性Agent推进其意图所执行的动作

通信

- 言语行为
- **通信方式**
- 基于消息的Agent 通信语言：KQML
- 基于消息的Agent 通信语言：FIPA ACL

通信方式 – 消息传递

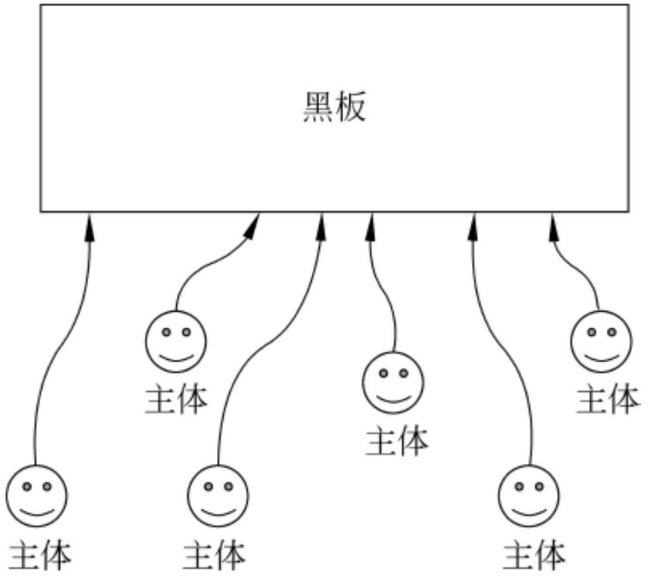
- 参与交互的Agent（**主体**）之间需要建立某种通信通道，并基于该通信通道进行消息传递
  - 例：基于TCP/IP协议的消息传递



- 一旦Agent间建立了通信通道，那么处于通信通道中的Agent就可以进行双向、对等的通信

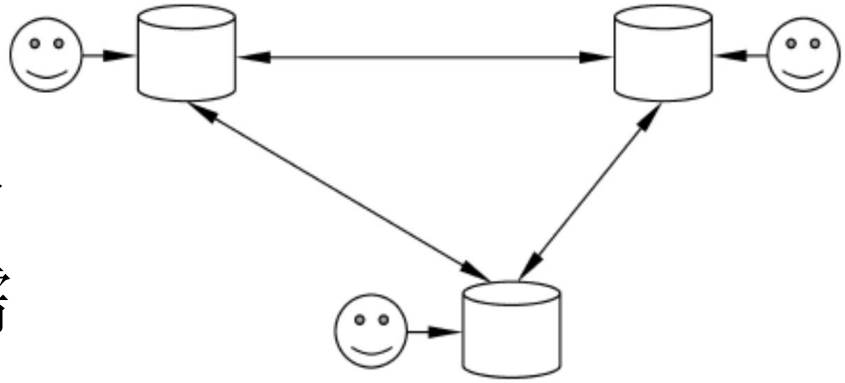
- 该通信方式具有较好的保密性和实用性
  - 通信通道归属于参与交互的双方Agent，不能为其他Agent共享

通信方式 – 黑板



- 在现实世界，人们通常借助于黑板通信方式进行各种各样的交流
- 参与通信的Agent共享一个公共的区域
  - 可以对该区域进行访问，包括写入信息、读取信息
- 无须建立通信通道，Agent之间的消息交互通过一个公共共享区域来完成
- 该通信方式的保密性较差，实时性不如消息传递

通信方式 – 邮箱



- 参与通信的Agent都有自己的邮箱并且它们之间需要建立起邮箱通道
- 一般情况下，这些邮箱通道可以为多个Agent之间的消息传输所共有，而不是由某些Agent独占
- 一个Agent欲向另一个Agent发送消息时，可将消息打包成邮件，并通过邮件通道发送到对方的邮箱中
- 该通信方式的保密性一般，实时性较差

几种通信方式的对比分析

通信方式	消息传递	黑板	邮箱
是否需通信通道	需要	不需要	需要
通道的持续性	持续或者暂时	持续或者暂时	暂时
接收确认	有	无	无
共享性	不共享	共享	共享
保密性	较好	差	一般
实时性	好	一般	差
示例	基于 TCP/IP 的消息传递，手机通信	基于黑板的教学，基于共分享区的通信	电子邮件系统

- 最常用的是基于消息传递的通信方式
  - 基于消息的Agent通信语言：KQML、FIPA ACL

课后作业3-1

- 使用课上所学的本体设计方法构建一个本体。使用类似课件中的**框图表示类别层次和实例关系**，并**描述相关的属性和至少2个可能的属性约束**（可用自然语言表达）。应满足的需求有：
  - 课程，可以分为必修课与选修课，其中必修课分为专业必修课和通修课，选修课分为专业选修课和公选课。课程的属性包括课程编号、授课老师、选课人数等等。
  - 《多智能体系统》是专业必修课
  - 《大学英语》是通修课
  - 《Java高级程序设计》是专业选修课
- 内容可根据上述需求进行补充。属性约束请自行设计，约束符合常理即可。