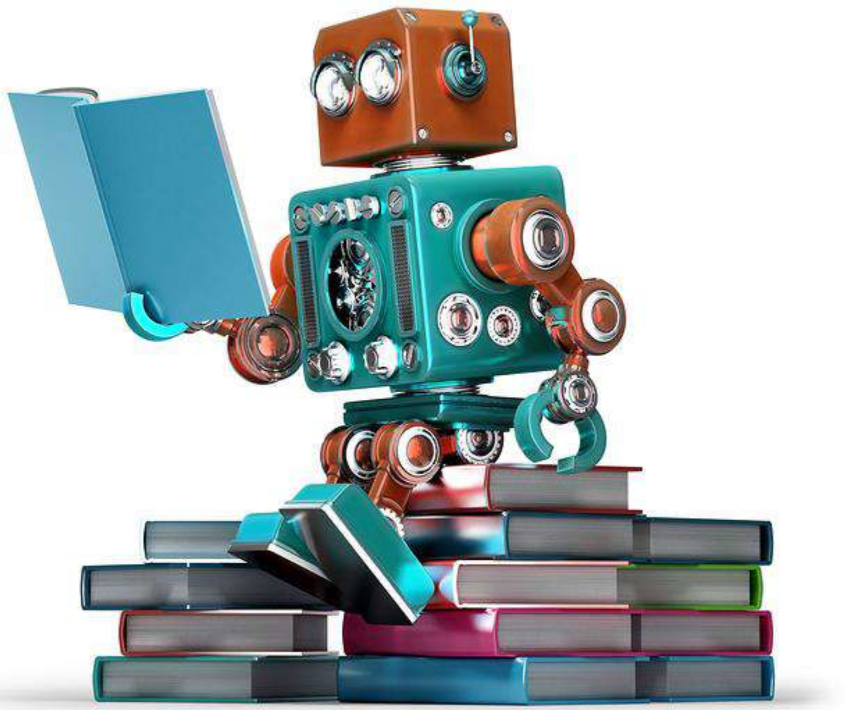
MACHINE REASONING
DAY 1


https://robohub.org/wp-content/uploads/2016/11/bigstock-Retro-Robot-Reading-A-Book-Is-110707406.jpg

DAY 1 AGENDA

1.1 Machine Reasoning Overview

1.2 Reasoning Types

1.3 Reasoning System Architectures

1.4 Knowledge Representation

1.5 Rule/Process Reasoning System **Workshop**

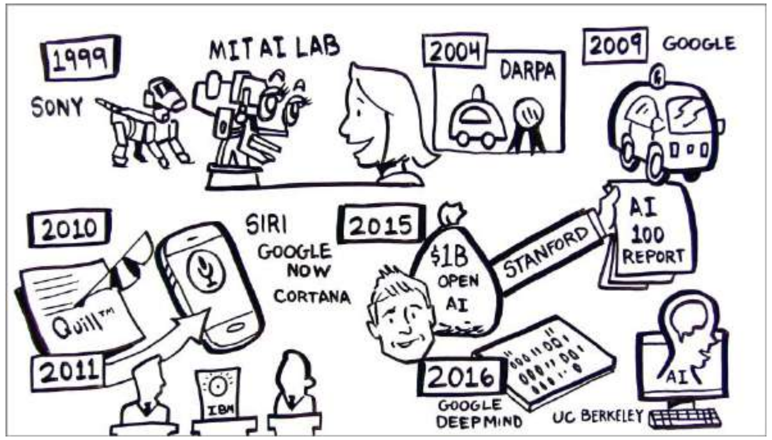
DAY 1 TIMETABLE

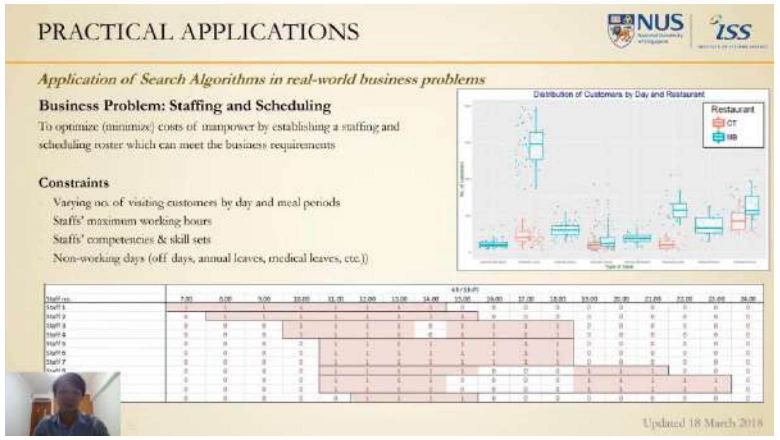
No	Time	Topic	By Whom	Where
1	9 am	Welcome and Introduction	GU Zhan (Sam)	Class
2	9.30 am	1.1 Machine Reasoning Overview	GU Zhan (Sam)	Class
3	10.10 am	Morning Break		
4	10.30 am	1.2 Reasoning Types 1.3 Reasoning System Architectures	GU Zhan (Sam)	Class
5	12.10 pm	Lunch Break		
6	1.30 pm	1.4 Knowledge Representation	GU Zhan (Sam)	Class
7	2.30 pm	1.5 Rule/Process Reasoning System Workshop Tutorial	GU Zhan (Sam) All	Class
8	3.10 pm	Afternoon Break		
9	3.30 pm	1.5 Rule/Process Reasoning System Workshop	All	Class
10	4.50 pm	Summary and Review	All	Class
11	5 pm	End		

1.1
MACHINE REASONING OVERVIEW

1.1 MACHINE REASONING OVERVIEW

- **AI History**




PRACTICAL APPLICATIONS

Application of Search Algorithms in real-world business problems

Business Problem: Staffing and Scheduling

To optimize (minimize) costs of manpower by establishing a staffing and scheduling roster which can meet the business requirements

Constraints

- Varying no. of visiting customers by day and meal periods
- Staffs' maximum working hours
- Staffs' competencies & skill sets
- Non-working days (off days, annual leaves, medical leaves, etc.)

Distribution of Customers by Day and Restaurant

Restaurant: CT, MB

Updated 18 March 2018

1.1 MACHINE REASONING OVERVIEW

- Reasoning
- Thinking
- Learning
- Artificial Intelligence

1.1 MACHINE REASONING OVERVIEW

- **Reasoning** Forward Chaining Inference

  is the capacity for consciously making sense of things, establishing and verifying facts, applying logic, and changing or justifying practices, institutions, and beliefs based on new or existing information.

- **Examples?**
  - Does this course look difficult to me?
  - Is the lecturer knowledgeable and competent?
  - Do my classmates appear to be smarter than I?
  - Am I likely to pass the course assessment?

1.1 MACHINE REASONING OVERVIEW

- **Thinking** **Backward Chaining Inference**

  encompasses a “goal oriented flow of ideas and associations that leads to a reality-oriented conclusion.”

- **Examples?**
  - I’d like to pass the course assessment…
  - What actions can I take to pass the course assessment?
  - I’d like to get NUS master degree…
  - What legitimate “optimization” can I do?

1.1 MACHINE REASONING OVERVIEW

- **Learning** **Knowledge Acquisition & Representation; Rule Extraction**

  is the process of acquiring new, or modifying existing, knowledge, behaviours, skills, values, or preferences.

- **Examples?**
  - Tell me, I shall hear.
  - Show me, I shall see.
  - But involve me, I shall learn.
  - Lecture, workshop, and further self study can enable me to build intelligent software to create business impact.

1.1 MACHINE REASONING OVERVIEW

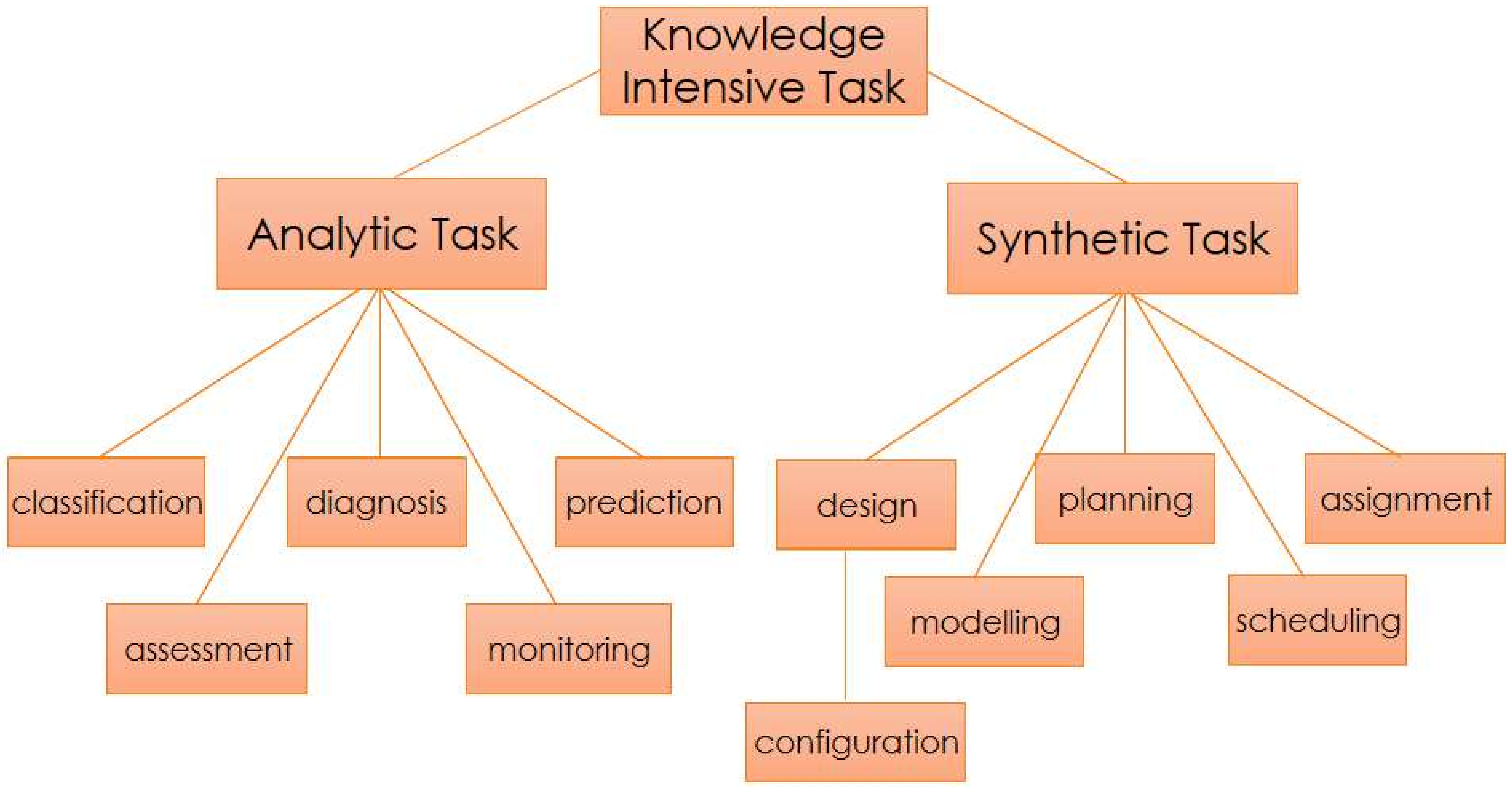
- **Artificial Intelligence** **used by machine to solve problem in fuzzy real world**

  is intelligence demonstrated by machines, which mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving".

- **Examples?**
  - Automated Cheque Recognition & Clearing
  - Algorithmic Trading System
  - High Frequency Trading System
  - DOTA2 Game Playing AI

1.1 MACHINE REASONING OVERVIEW

Problem Solving Task Hierarchy



1.1 MACHINE REASONING OVERVIEW

Problem Solving Task Types

- **Analytic Tasks**
  - System/Solution to be analysed pre-exists, but usually not completely "known“.
  - Input: some data to trigger the system (e.g. patient symptoms)
  - Output: some characterization or behaviours about the system (e.g. cause of illness)
- **Synthetic Tasks**
  - System/Solution does not yet exist.
  - Input: requirements about system to be constructed
  - Output: constructed system description

1.1 MACHINE REASONING OVERVIEW

Problem Solving of Analytic Tasks

- **Analytic Tasks**

  Identification, Classification, Prediction, Clustering/Grouping, …

- **Techniques (S-MR Machine Reasoning)**

  Heuristic Business Rules

  Decision Trees

  Case Based Reasoning

  Fuzzy Logic

  Rule Induction

  Machine Learning

  …

1.1 MACHINE REASONING OVERVIEW

Problem Solving of Synthetic Tasks

- **Synthetic Tasks**

  Planning, Scheduling, Optimisation, Design, …

- **Techniques (S-RS Reasoning Systems)**

  Uninformed (brute force / blind) Search

  Informed (heuristic) Search

  Simulations

  Genetic Algorithms

  Reinforcement Learning

  Data Mining

  …

1.2
REASONING TYPES

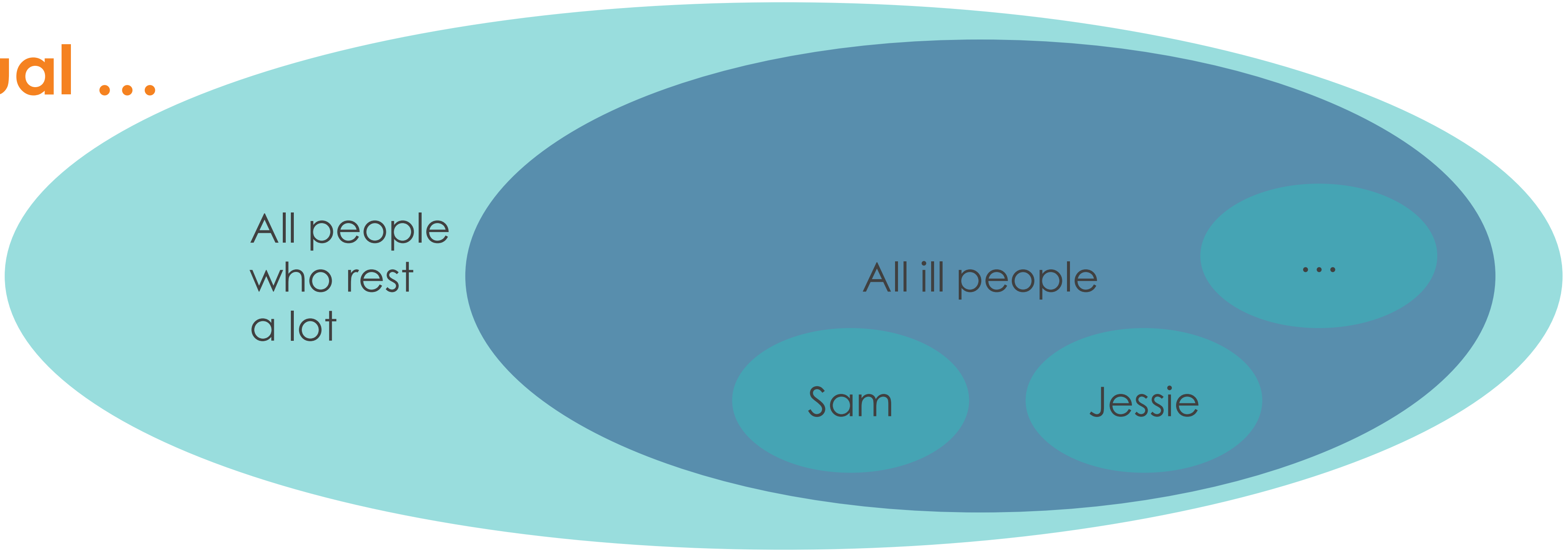
1.2 REASONING TYPES

- Deductive Reasoning
- Inductive Reasoning
- Analogical Reasoning
- Abductive Reasoning

1.2 REASONING TYPES

Deductive Reasoning

- **Knowledge/Rule** : **All ill people** need rest a lot.
- **Individual 1** : **Sam** is **ill**, therefore he need rest a lot.
- **Individual 2** : **Jessie** is **ill**, therefore she need rest a lot.
- **Individual …**

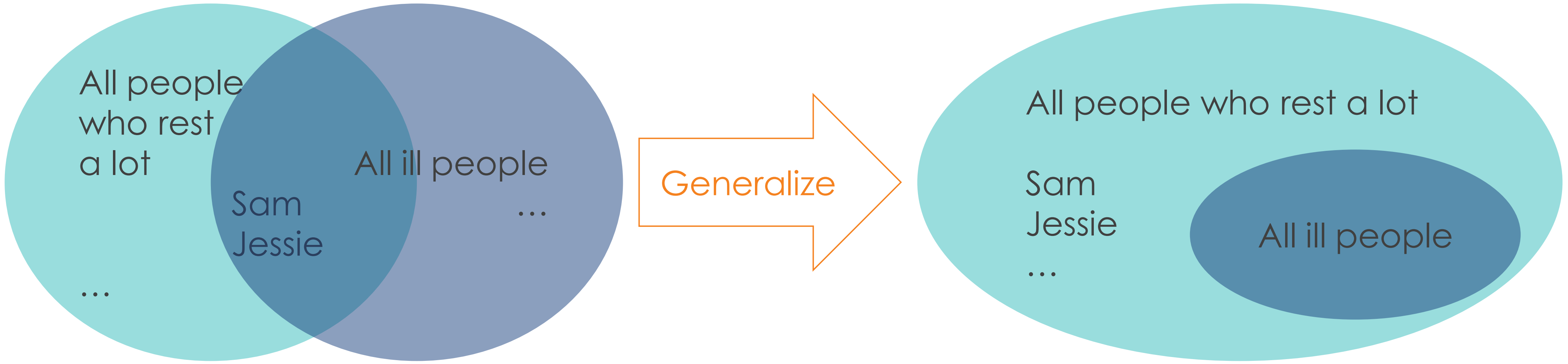


☺ Reasoning Rationality: Universal → Individual

1.2 REASONING TYPES

Inductive Reasoning

- **Individual 1** : When **Sam** is **ill**, he rests a lot.
- **Individual 2** : When **Jessie** is **ill**, she rests a lot.
- **Generalised Rule** : **All people** who are **ill**, they rest a lot.

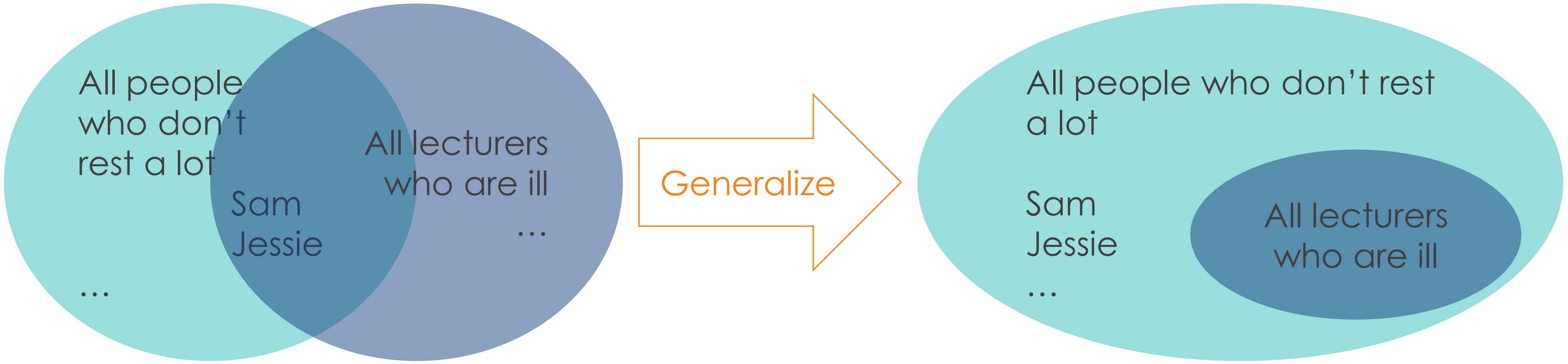


☺ **Reasoning Rationality: Individual → Universal** **(Machine Learning)**

1.2 REASONING TYPES

Inductive Reasoning

- **Individual 1** : When **lecturer Sam** is **ill**, he doesn't rest a lot.
- **Individual 2** : When **lecturer Jessie** is **ill**, she doesn't rest a lot.
- **Generalised Rule** : **All lecturers** who are **ill**, they don't rest a lot.

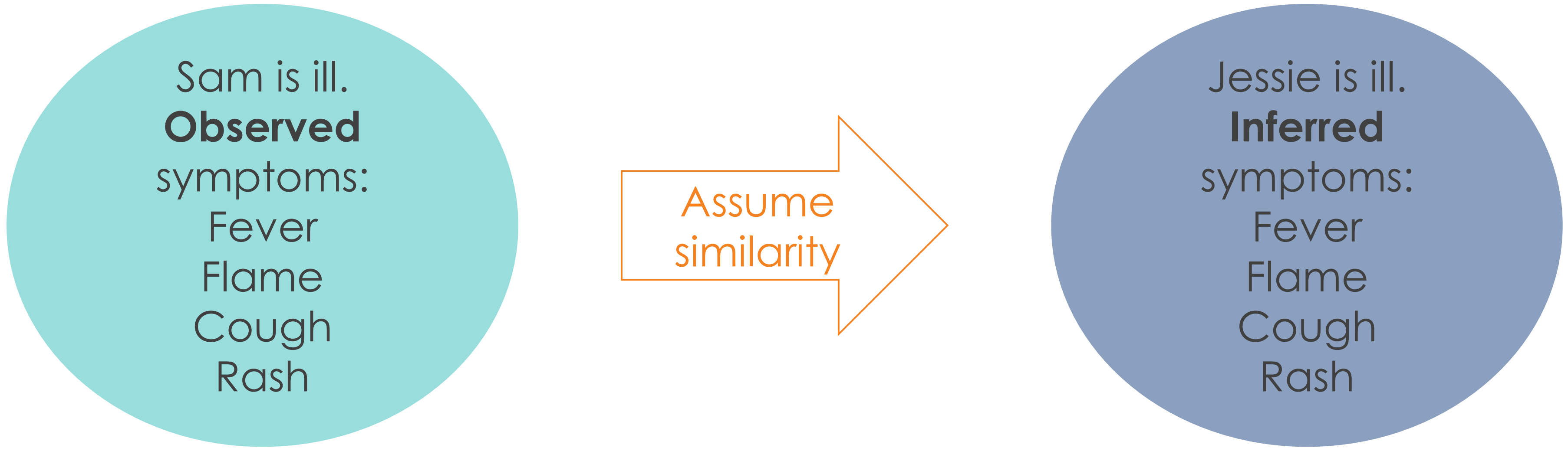

All people who don't rest a lot
...
Sam
Jessie
All lecturers who are ill
...
Generalize
All people who don't rest a lot
Sam
Jessie
...
All lecturers who are ill

☺ **Reasoning Rationality: Individual → Universal (Machine Learning)**

1.2 REASONING TYPES

Analogical Reasoning

- **Known case** : **Sam** is ill with his symptoms: fever, flame, cough, and rash.
- **Inferred case** : Jessie is ill too, therefore she **would** have **same** symptoms as **Sam**: fever, flame, cough, and rash.

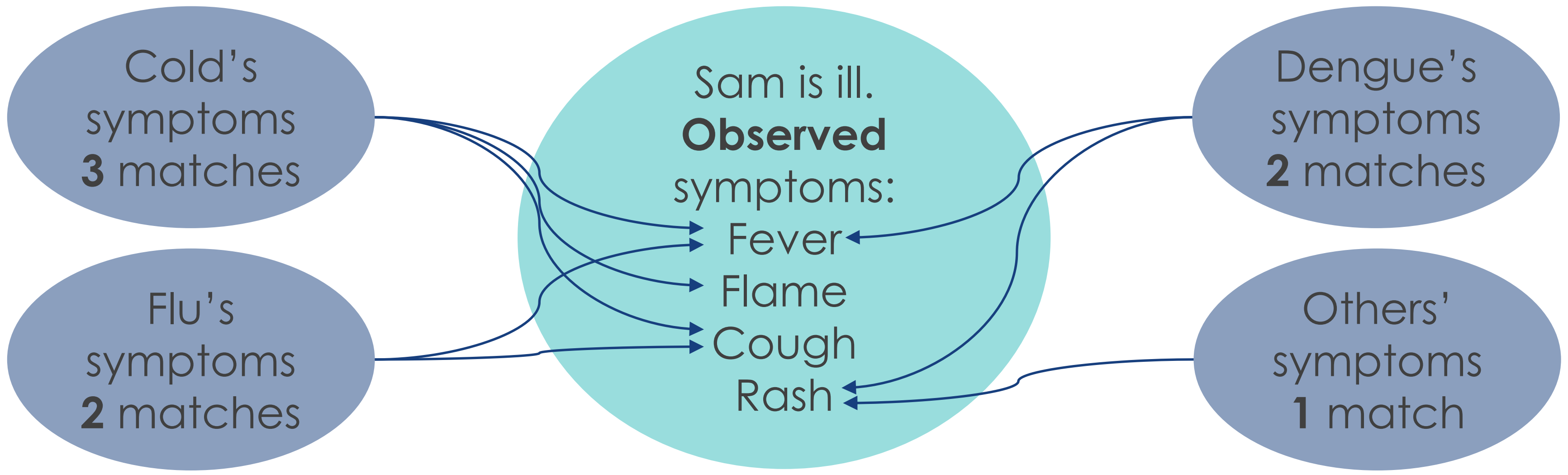


☺ Reasoning Rationality: Known case → Inferred case

1.2 REASONING TYPES

Abductive Reasoning

- **Known observations** : **Sam** is ill with his symptoms: fever, flame, cough, and rash.
- **Inferred root cause** : **Cold? Flu? Dengue? Others?**


Cold's symptoms 3 matches
Flu's symptoms 2 matches
Sam is ill. Observed symptoms:
Fever
Flame
Cough
Rash
Dengue's symptoms 2 matches
Others' symptoms 1 match

☺ **Reasoning Rationality: Observations → Causes likelihood**

1.2 REASONING TYPES

Fuzzy Reasoning

Long Hair Group ← Hair length ≥ 10 cm

Hair length < 10 cm → Short Hair Group

Long Hair Group ← Hair length is long

Hair length is short → Short Hair Group

What if the hair length is both long and short → Which Group?

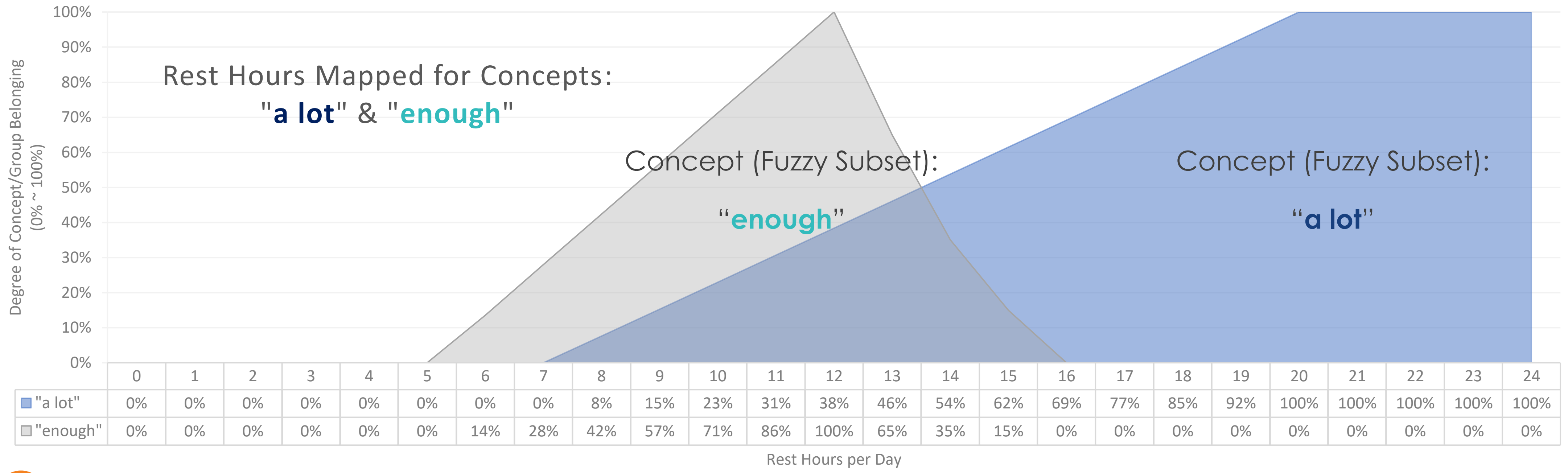
Source
https://www.google.com/search?q=nicolas+cage&rlz=1C1GCEU_enSG820SG820&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiOt7bhkqTjAhVcknAKHTSZBHgQ_AUIECgB&biw=1920&bih=896

1.2 REASONING TYPES

Fuzzy Reasoning

Linguistic Concept

- **Imprecise Knowledge** : All ill people need rests **a lot**.
- **Precise measurement** : How many hours/minutes per day is considered as "**a lot**"?



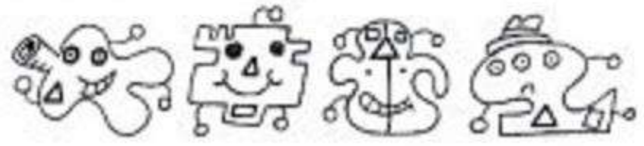
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
🟦 "a lot"	0%	0%	0%	0%	0%	0%	0%	0%	8%	15%	23%	31%	38%	46%	54%	62%	69%	77%	85%	92%	100%	100%	100%	100%	100%
⬜ "enough"	0%	0%	0%	0%	0%	0%	14%	28%	42%	57%	71%	86%	100%	65%	35%	15%	0%	0%	0%	0%	0%	0%	0%	0%	0%

☺ **Reasoning Rationality: Imprecise → Precise** (measures/actions)

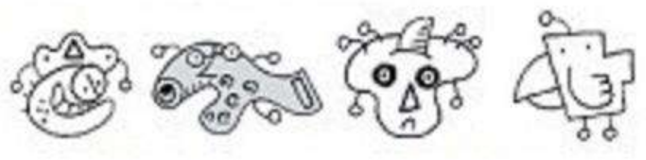
1.2 REASONING TYPES

Exercise 1.2

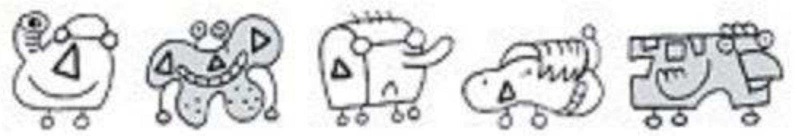
Aliens



Not aliens



Which one is alien?



A B C D E

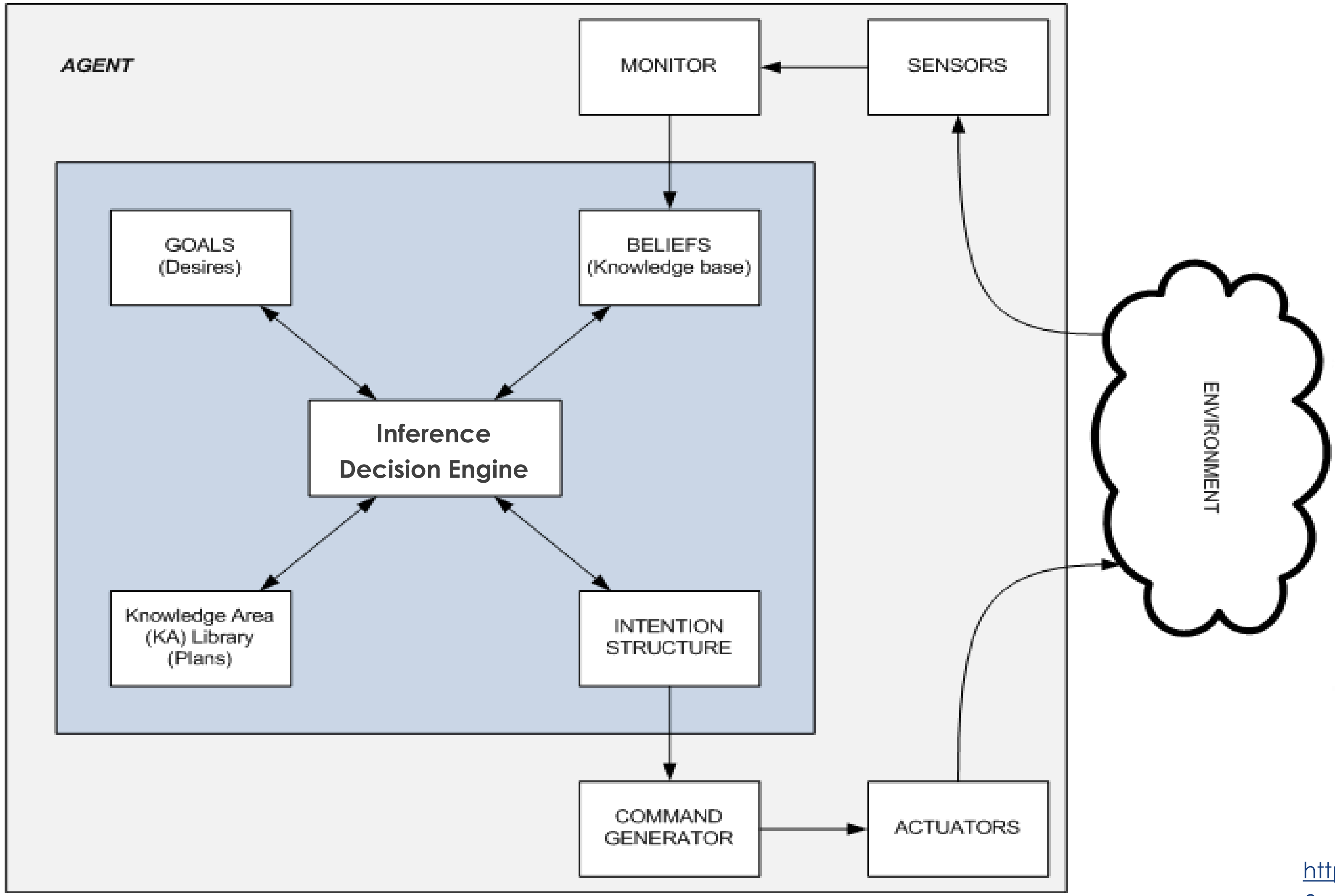
1.3
REASONING SYSTEM ARCHITECTURES

1.3 REASONING SYSTEM ARCHITECTURES

- **Proactive Reasoning Systems (Goal Driven)**
  - Autonomous Software System (Sales Chabot, Robotic Process Automation)
  - Multi Agent Cooperative System (Warehouse Robotic Swarm, Coordinated Robotic Cleaners)
  - Constrain Solver (Global Travel Planner)
- **Reactive Reasoning Systems (Data Driven)**
  - Business Rule Management System (BRMS)
  - Business Process Management System (BPMS)
  - Constrain Solver (Delivery Vehicles Scheduler)

1.3 REASONING SYSTEM ARCHITECTURES

Goal Driven Systems

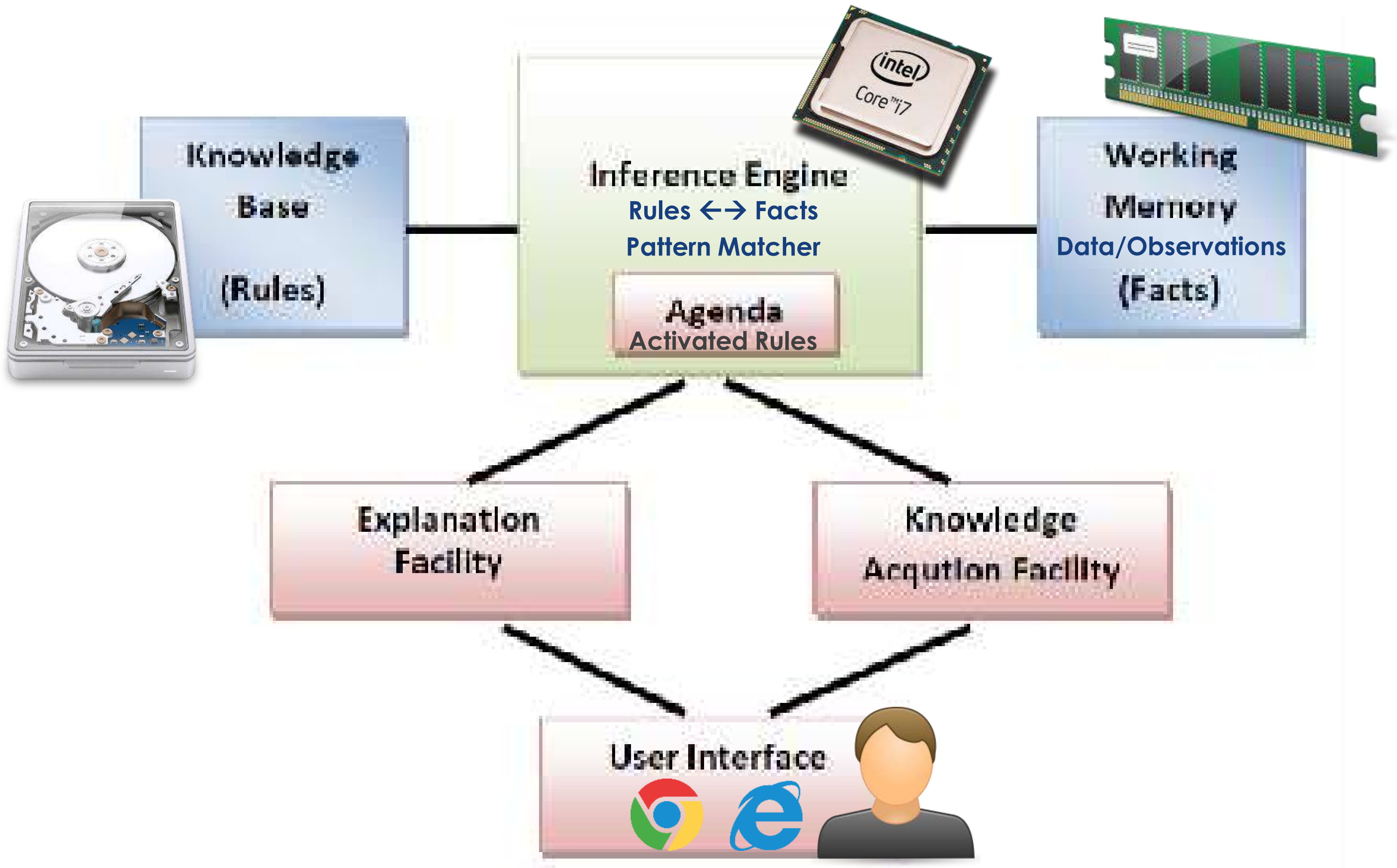


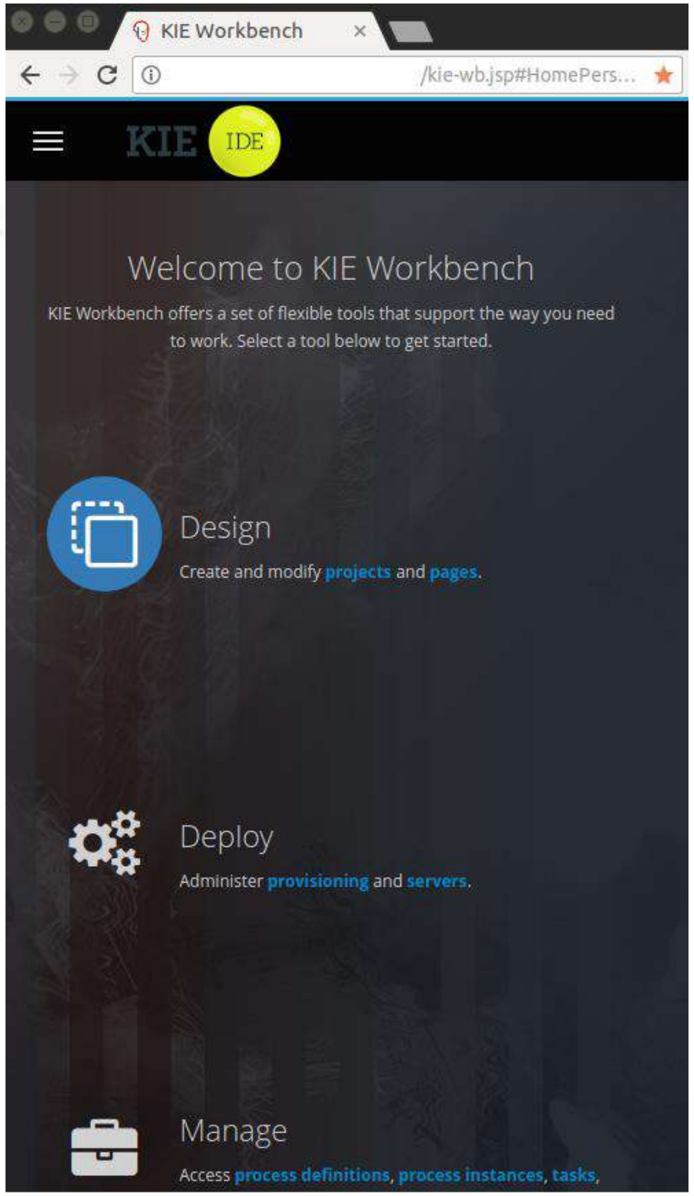
https://upload.wikimedia.org/wikipedia/commons/f/f5/PRS.gif

https://static1.squarespace.com/static/57c8a68a20099ef23fb19e90/t/5a32a9804192022be97f9d10/1513269668629/Atlas.png

1.3 REASONING SYSTEM ARCHITECTURES

Knowledge (Fact/Rule/Process) Driven Systems





Welcome to KIE Workbench

KIE Workbench offers a set of flexible tools that support the way you need to work. Select a tool below to get started.

Design

Create and modify projects and pages.

Deploy

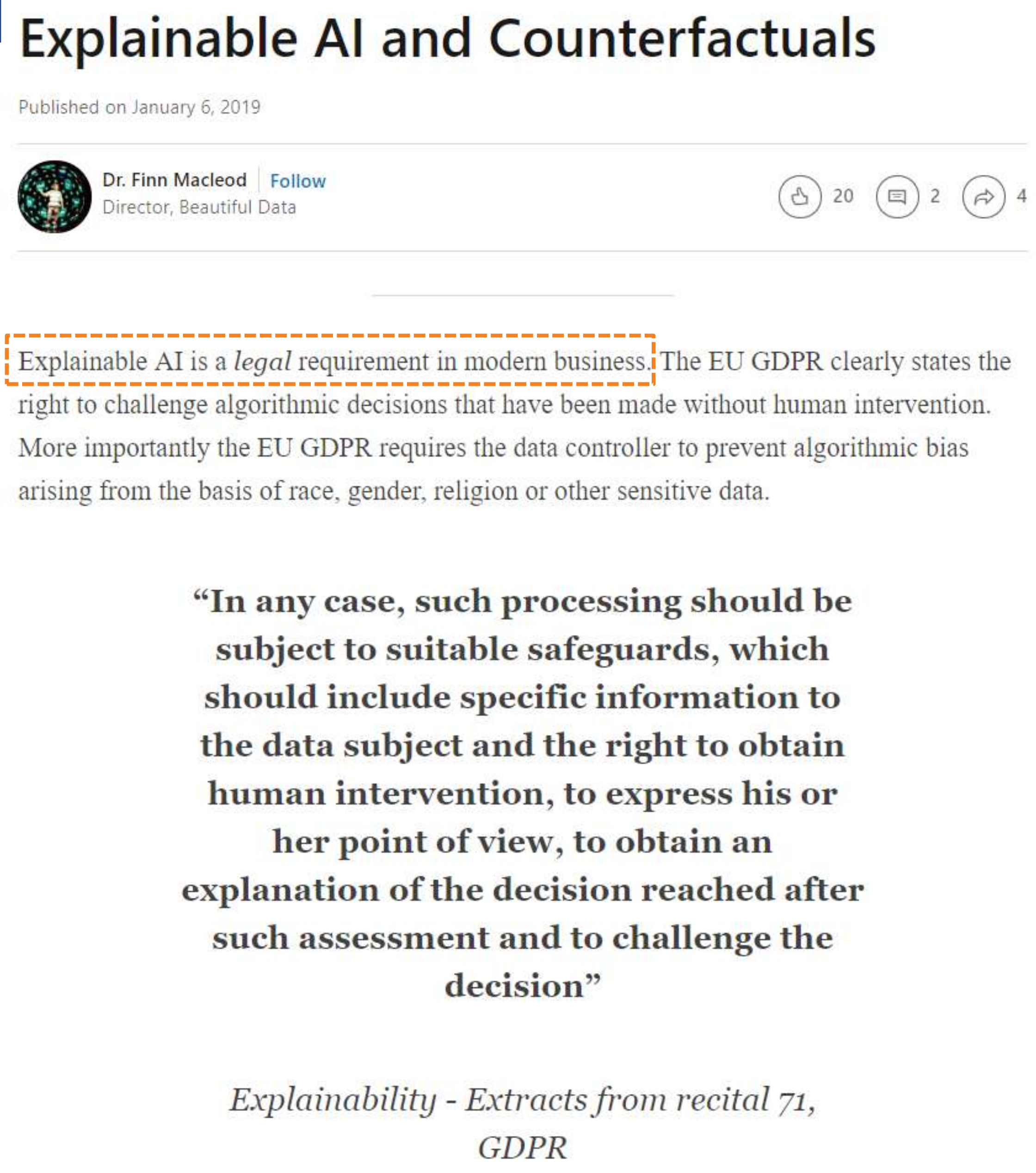
Administer provisioning and servers.

Manage

Access process definitions, process instances, tasks,

https://www.researchgate.net/profile/Bhavani_Panda/publication/232707515/figure/fig2/AS:339607343386657@1457980108974/Fig2-Structure-of-a-rule-based-expert-system.png

REGULATION



Explainable AI and Counterfactuals

Published on January 6, 2019

Dr. Finn Macleod | Follow
Director, Beautiful Data

20 2 4

Explainable AI is a *legal* requirement in modern business. The EU GDPR clearly states the right to challenge algorithmic decisions that have been made without human intervention. More importantly the EU GDPR requires the data controller to prevent algorithmic bias arising from the basis of race, gender, religion or other sensitive data.

"In any case, such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right to obtain human intervention, to express his or her point of view, to obtain an explanation of the decision reached after such assessment and to challenge the decision"

Explainability - Extracts from recital 71, GDPR

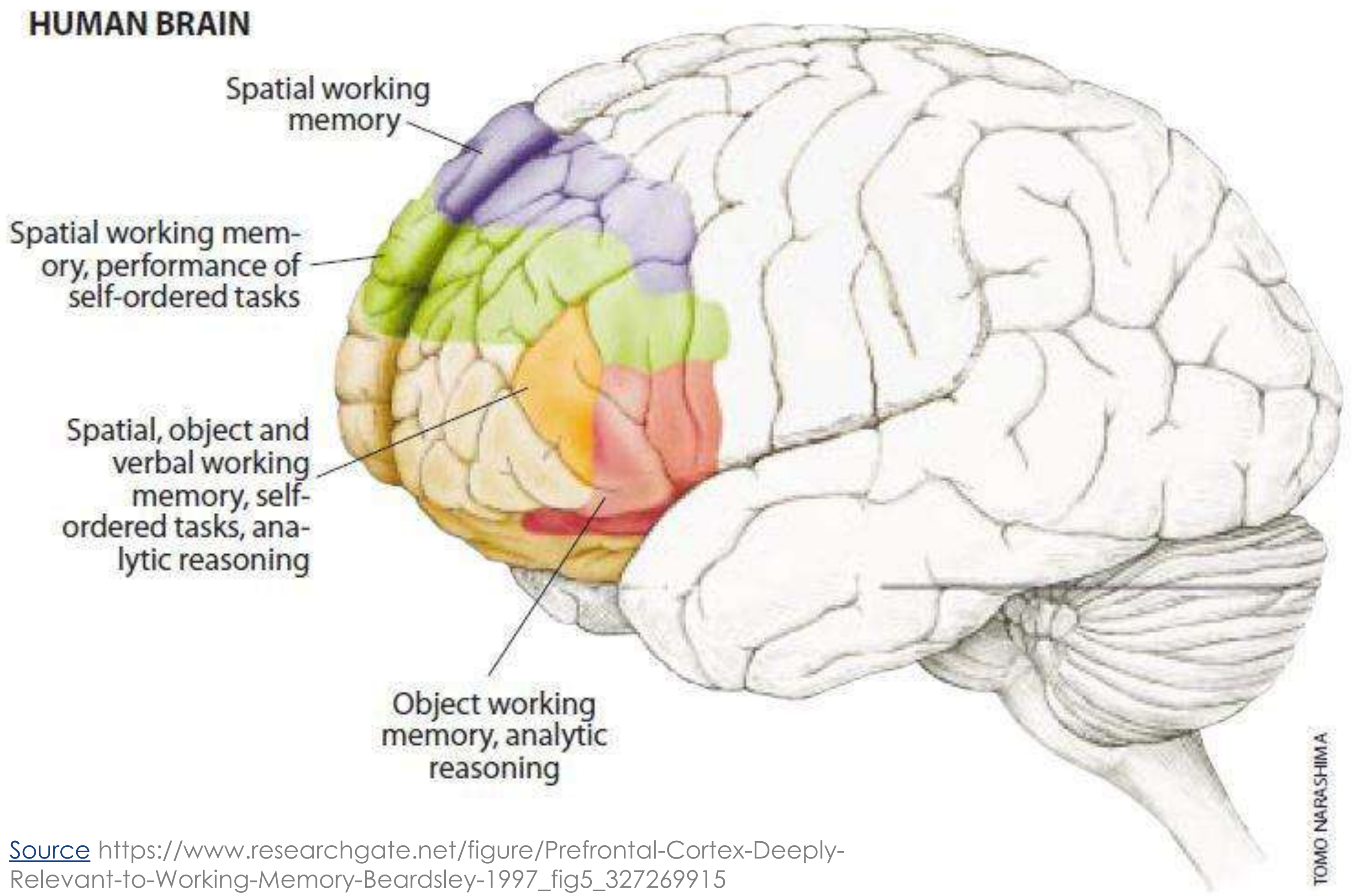
Source: https://www.linkedin.com/pulse/explainable-ai-counterfactuals-dr-finn-macleod

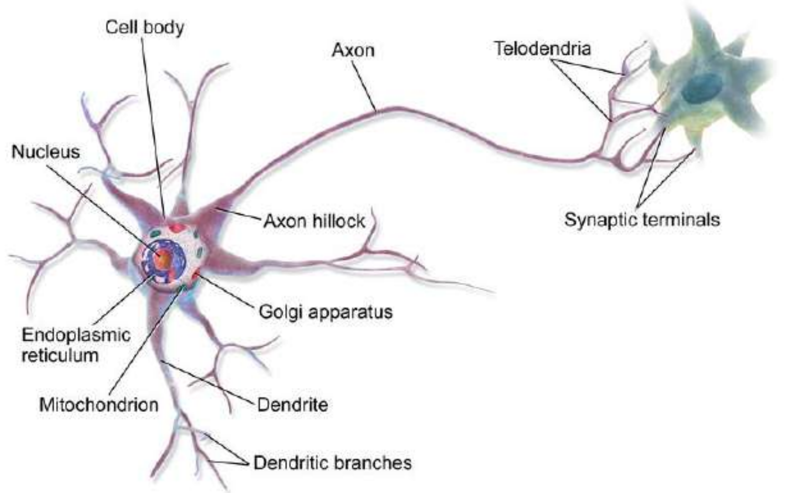
1.4

KNOWLEDGE REPRESENTATION

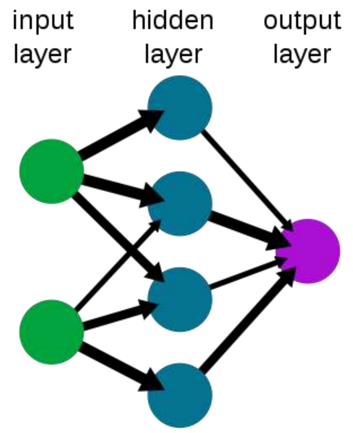
1.4 KNOWLEDGE REPRESENTATION

- **Knowledge representation in human brain** (black box)


HUMAN BRAIN
Spatial working memory
Spatial working memory, performance of self-ordered tasks
Spatial, object and verbal working memory, self-ordered tasks, analytic reasoning
Object working memory, analytic reasoning
TOMO NARASHIMA



A simple neural network



Source https://www.researchgate.net/figure/Prefrontal-Cortex-Deeply-Relevant-to-Working-Memory-Beardsley-1997_fig5_327269915

1.4 KNOWLEDGE REPRESENTATION

- **Knowledge representation in machine** (white box)**: A scheme /method that allows the computer system to use or manipulate it to reason and solve problems**
    - Unlike humans, knowledge must be 'transplanted/saved' into machine reasoning system/memory
    - Representation goes hand-in-hand with reasoning/inference mechanism (computer algorithm)
    - Large amount of knowledge is usually needed to solve complex problems

Data Structure

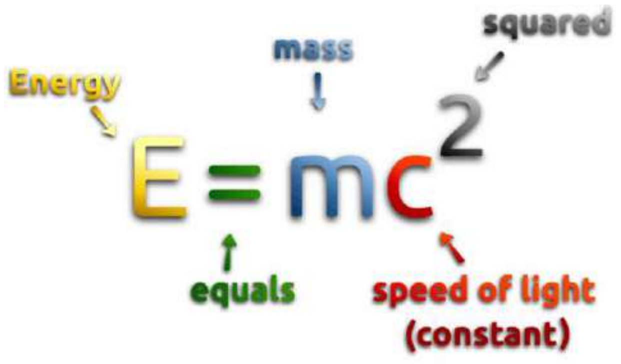
Processing Logic

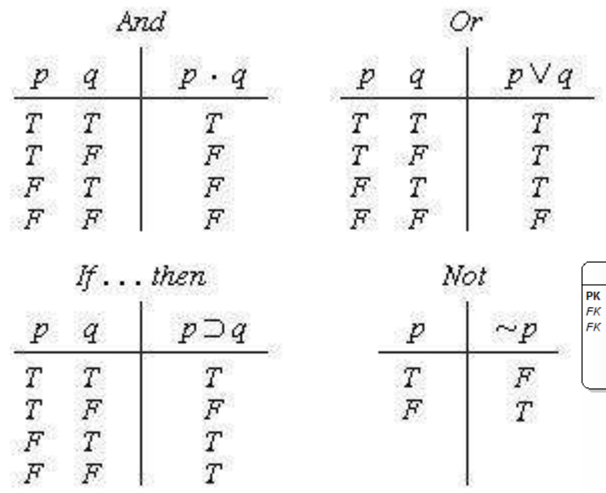
- **Explicit knowledge representation (think of documentation) enables business knowledge management and retention.**

1.4 KNOWLEDGE REPRESENTATION

Forms of Knowledge Representation

- Natural Language
- Formula
- Formal Logic
- Semantic Web
- Frames
- Ontology
- Knowledge Graph
- Database
- Rules
- And many other forms…





And

p	q	$p \cdot q$
T	T	T
T	F	F
F	T	F
F	F	F

Or

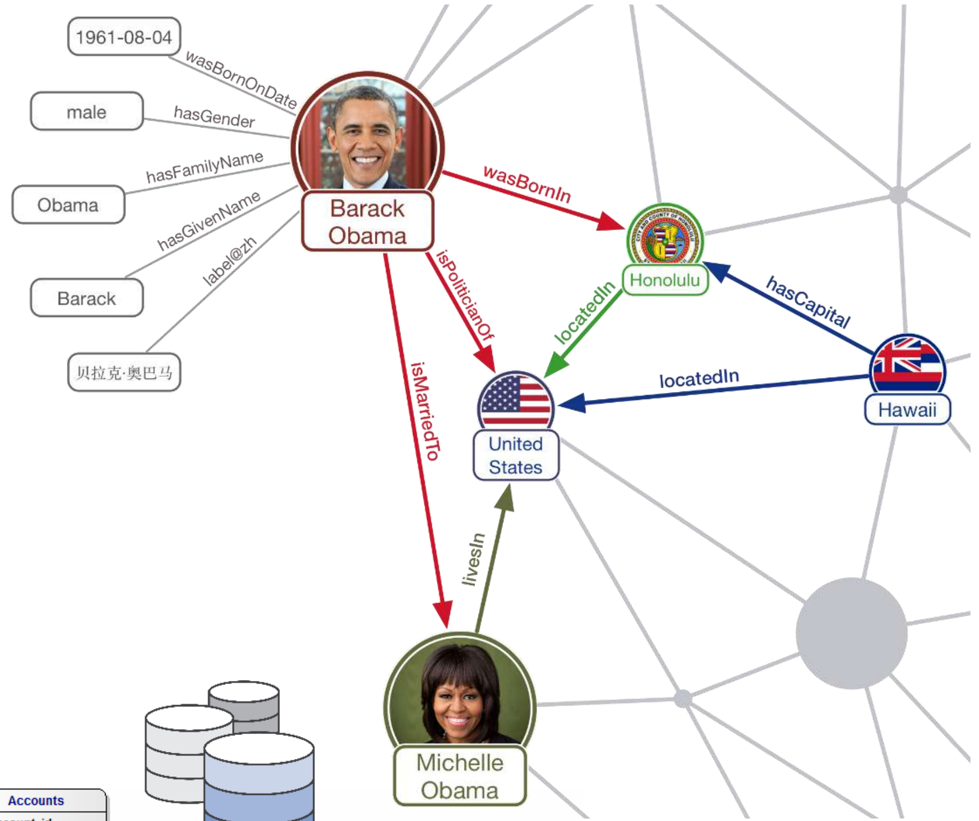
p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

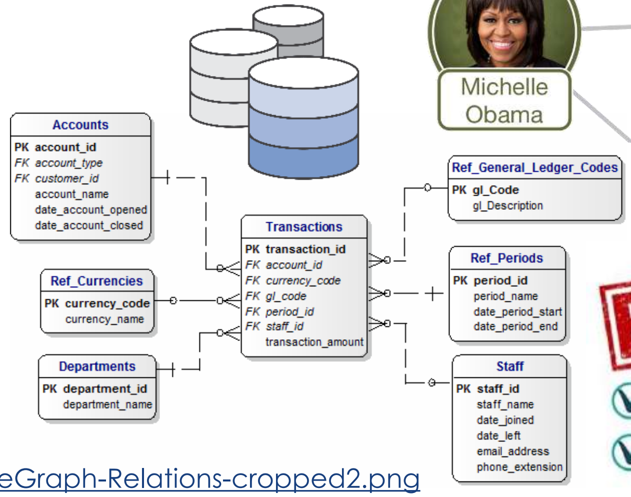
If . . . then

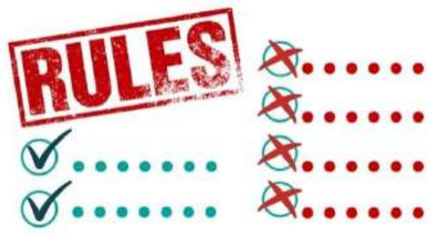
p	q	$p \supset q$
T	T	T
T	F	F
F	T	T
F	F	T

Not

p	$\sim p$
T	F
F	T







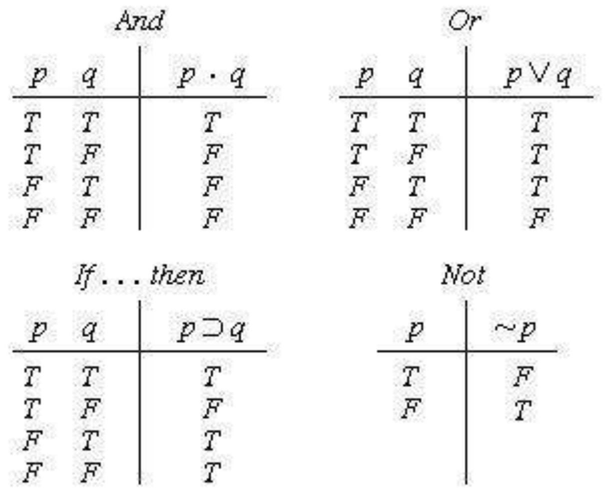
https://www.ambiverse.com/wp-content/uploads/2017/01/KnowledgeGraph-Relations-cropped2.png

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – Propositional Logic

- **Propositional Logic**
  - Examples of propositions: propositional sentence
    - ***p*** = "Sam has flu."
    - ***q*** = "Sam has cough."
  - What about sentence: ***s*** = "x + y = 5", where x and y are variables?

    ☺ Not a proposition, as its truth cannot be defined unless x and y are assigned specific values



And

p	q	$p \cdot q$
T	T	T
T	F	F
F	T	F
F	F	F

Or

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

If . . . then

p	q	$p \supset q$
T	T	T
T	F	F
F	T	T
F	F	T

Not

p	$\sim p$
T	F
F	T

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – Propositional Logic

- **The syntax of propositional logic expression is constructed using propositions and connectives**
    - All propositions must be either **true** or **false** (referred to as **truth value** of the proposition)
- **Connectives**

• $\neg$	negation	“not”
• $\vee$	disjunction	“or”
• $\wedge$	conjunction	“and”
• $\rightarrow$	implication	“if … then”
• $\leftrightarrow$	bi-conditional	“if and only if”

- **Complex expressions using connectives**
    - $\boldsymbol{p \wedge q}$ = “Sam has flu. **AND** Sam has cough.”
    - $\boldsymbol{p \wedge \neg q}$ = “Sam has flu. **AND** Sam has no cough.”
    - $\boldsymbol{p \rightarrow q}$ = “**IF** Sam has flu **THEN** Sam has cough.”

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

- **First Order Logic (Predicate Calculus)**
  - Propositional logic assumes the world contains: **facts**
  - First order logic (also called first-order predicate calculus, or predicate logic) assumes the world contains:
    - **Objects (Class)** : people, houses, numbers, colors, baseball games, …
    - **Constants (Instance)** : The White House, Sam GU Zhan, π, NUS,...
    - **Variables** : x, y, a, b, ...
    - **Relations (Predicate)** : is student, has leg, eats, is bigger than, is part of, is red, is round, prime to, come between, is one more than, …
    - **Functions (Predicate)** : father of, best friend, square root of, sum of, …
    - **Connectives** : $\neg \ \rightarrow \ \wedge \ \vee \ \leftrightarrow$
    - **Equality** : =
    - **Quantifiers** : $\forall \ \exists$

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

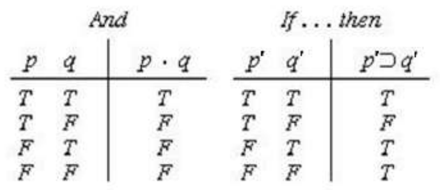
- **Sentences of First Order Logic**
  - **Term** (noun) is an expression that refers to an object.
    - FatherOf(DiDi) : "father of DiDi"
    - ¬ FatherOf(DiDi) : "not father of DiDi"
    - FatherOf(x) : "someone's father"
    - Sam, Jessie, DiDi, Machine Reasoning Course, PhD, …
    - Integer: x, y, z (variables of object: all integer numbers)
  - **Atomic Sentence** (semantics) is formed from **one** predicate symbol followed by **one** parenthesized list of terms
    - IsFriend(Jessie, Sam) : Jessie is friend to Sam.
    - IsFriend(Jessie, FatherOf(DiDi)) : Jessie is friend to DiDi's father.

relational predicate

functional predicate

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic



	And			If . . . then	
p	q	$p \cdot q$	p'	q'	$p' \supset q'$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	F	T	T
F	F	F	F	F	T

- **Sentences of First Order Logic**
  - **Complex Sentence** (semantics) is made from **Atomic Sentences** using logical connectives
    - IsClassmate(Jessie, Mary) ∧ IsFriend(Jessie, Sam) → IsFriend(Mary, Sam)

p'

q'

- **Establish <u>Truth</u> of Predicate**
  - Predicates need to be propositionalized for use in reasoning
  - **Method 1**: Assign specific value to predicate expressions (similar to instantiation)
    - StudyAt(x, NUS) → Smart(x)   What's the scope of x?
    - x = Sam
    - Conclusion: StudyAt (Sam, NUS) → **Smart(Sam)**

1.4 KNOWLEDGE REPRESENTATION
Formal Logic – First Order Logic

- **Establish Truth of Predicate**
  - **Method 2A**: Universal quantifier: ∀

    We want to express "Everyone studying at NUS is smart."

    √ ∀x StudyAt(x, NUS) → Smart(x) : "For everyone, if the person is studying at NUS then the (**same**) person is smart" (For those are not studying at NUS, we don't know.)

    × ∀x StudyAt(x, NUS) ∧ Smart(x) : "Everyone (all persons in Singapore) is studying at NUS and all (**these**) persons are smart." **incorrect semantic**
  - **Method 2B**: Existential quantifier: ∃

    We want to express "Someone studying at NUS is smart."

    √ ∃x StudyAt(x, NUS) ∧ Smart(x) : "There is someone studying at NUS and this (**same**) person is smart."

    × ∃x StudyAt(x, NUS) → Smart(x) : "There is someone, when he/she is studying at NUS then he/she is smart." (When this (**same**) person is having a rest, then he/she may not be smart.) **incorrect semantic**

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

An example:

- **Given the background information below:**
  - Sam loves all animals.
  - Anyone who loves all animals does not kill an animal.
  - Either Sam or Curiosity killed the cat, which is named HelloKitty.
- **Question: Did Curiosity kill HelloKitty?**
- **Use the following predicates:**

Animal(x)	x is an animal.
Cat(x)	x is a cat.
Loves(x, y)	x loves y.
Kills(x, y)	x kills y.

Source
https://www.pinterest.com/pin/355925176775527097/

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

Logical inference/calculation steps:

- Express the sentences/knowledge/facts/rules using first order logic formulas
- Convert first order logic formulas to clausal form
- Conduct resolution with unification and variable renaming

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

- **Knowledge Base (KB)**
  - ∀y Loves(Sam, y) ∧ Animal(y)
  - ∀x ∀y Loves(x, y) ∧ Animal(y) → ¬Kills(x, y)
  - Kills(Sam, Cat(HelloKitty)) ∨ Kills(Curiosity, Cat(HelloKitty))
- **Clouse form conversion: p → q ≡ ¬p ∨ q**
  - ∀x ∀y Loves(x, y) ∧ Animal(y) → ¬Kills(x, y) ≡ ∀x ∀y ¬[Loves(x, y) ∧ Animal(y)] ∨ ¬Kills(x, y)
- **Hypothesis to prove is:** **Curiosity killed HelloKitty.**
  - a = Kills(Curiosity, Cat(HelloKitty))
- **Refutation of hypothesis is:** **Curiosity didn't kill HelloKitty.**
  - ¬a = ¬Kills(Curiosity, Cat(HelloKitty))

1.4 KNOWLEDGE REPRESENTATION

Formal Logic – First Order Logic

- **Proof by refutation: KB ∧ ¬a**

  KB ∧ ¬a ≡ ∀y Loves(Sam, y) ∧ Animal(y) ∧ ∀x ∀y ¬[Loves(x, y) ∧ Animal(y)] ∨ ¬Kills(x, y) ∧ Kills(Sam, Cat(HelloKitty)) ∨ Kills(Curiosity , Cat(HelloKitty)) ∧ ¬Kills(Curiosity , Cat(HelloKitty))

  ≡ ∀y Loves(Sam, y) ∧ Animal(y) ∧ ∀x ∀y ¬[Loves(x, y) ∧ Animal(y)] ∨ ¬Kills(x, y) ∧ Kills(Sam, Cat(HelloKitty)) ∨ {}

  variable unification: (1) x = Sam; (2) y = Cat(HelloKitty)

  ≡ Loves(Sam, Cat(HelloKitty)) ∧ Animal(Cat(HelloKitty)) ∧ ¬[Loves(Sam, Cat(HelloKitty)) ∧ Animal(Cat(HelloKitty))] ∨ ¬Kills(Sam, Cat(HelloKitty)) ∧ Kills(Sam, Cat(HelloKitty)) ∨ {}

  ≡ Loves(Sam, Cat(HelloKitty)) ∧ ¬[Loves(Sam, Cat(HelloKitty)) ∧ Animal(Cat(HelloKitty))] ∧ Animal(Cat(HelloKitty)) ∨ {} ∨ {}

  ≡ Loves(Sam, Cat(HelloKitty)) ∧ ¬Loves(Sam, Cat(HelloKitty)) ∨ ¬Animal(Cat(HelloKitty)) ∧ Animal(Cat(HelloKitty)) ∨ {} ∨ {}

  ≡ {} ∨ {} ∨ {} ∨ {} ≡ {}

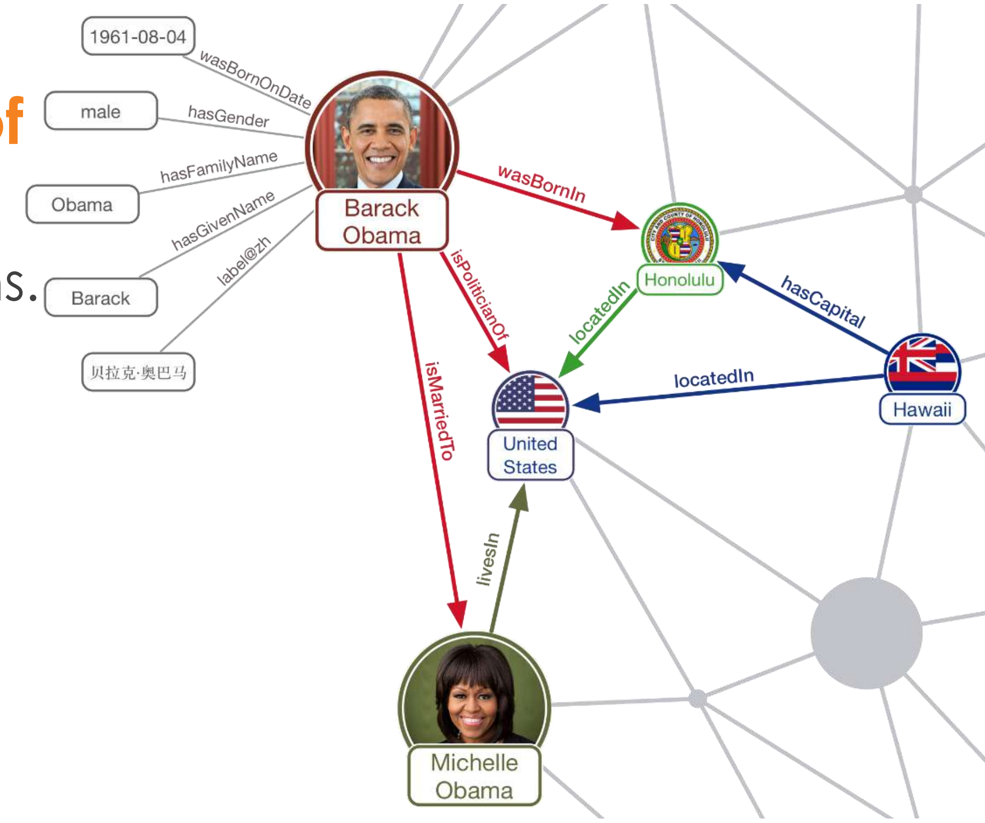
- **Conclusion:**

We reject	: ¬a = ¬Kills(Curiosity, Cat(HelloKitty))	Curiosity didn't kill HelloKitty.
But accept	: a = Kills(Curiosity, Cat(HelloKitty))	Curiosity killed HelloKitty.

1.4 KNOWLEDGE REPRESENTATION

Semantic Web, Knowledge Graph

- **Semantic web is a model for word concepts in human cognition, consisting of nodes, links and labels**
  - **Nodes** represent objects, concepts, or situations. They can be instances (individual objects as in Knowledge Graph) or classes (generic objects as in Semantic Web)
  - **Links** between nodes represent a relationship
  - **Labels**
    - Labels on nodes indicate the name of the object, concept, etc.
    - Labels on links describe the type of relationship between nodes
- **Reasoning question: What's the relationship between Barack and Michelle Obama?**

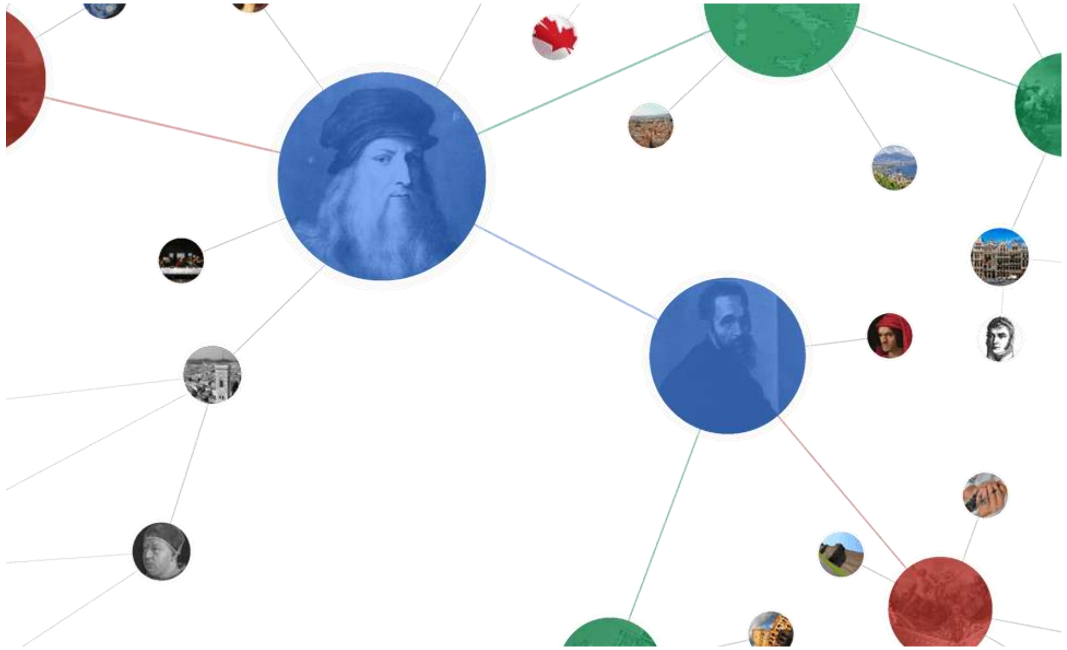


https://www.ambiverse.com/wp-content/uploads/2017/01/KnowledgeGraph-Relations-cropped2.png

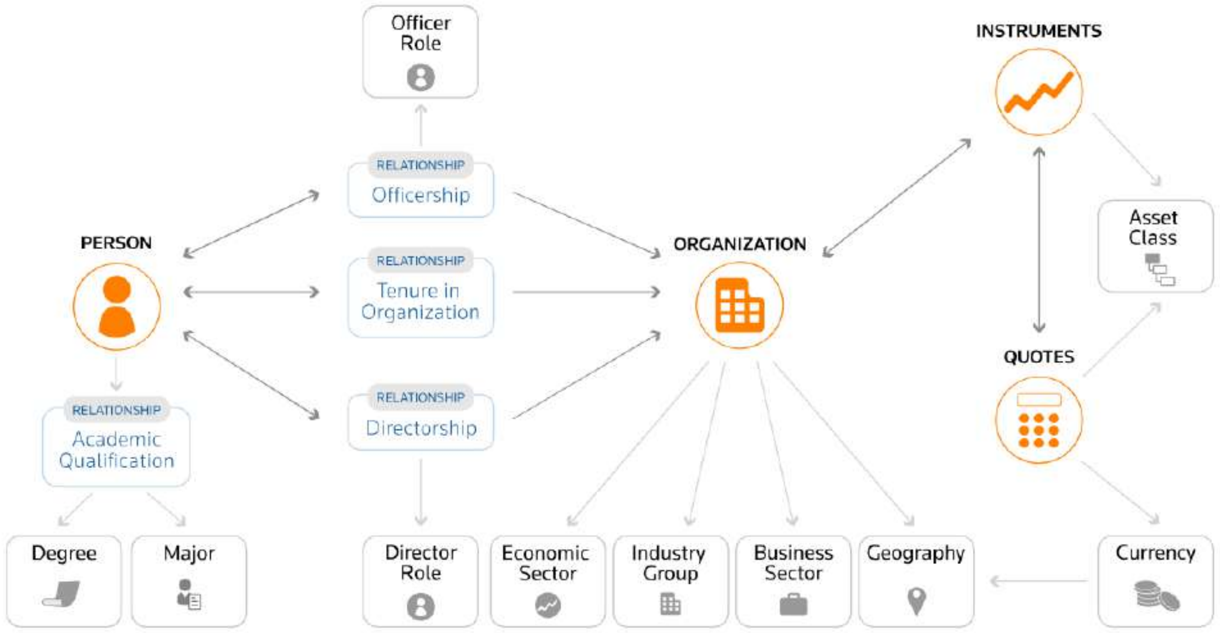
1.4 KNOWLEDGE REPRESENTATION

Semantic Web, Knowledge Graph

- Google Knowledge Graph
- Thomson Reuters Knowledge Graph product: Perm ID



https://www.tampa-seo.com/wp-content/uploads/static-graph.png

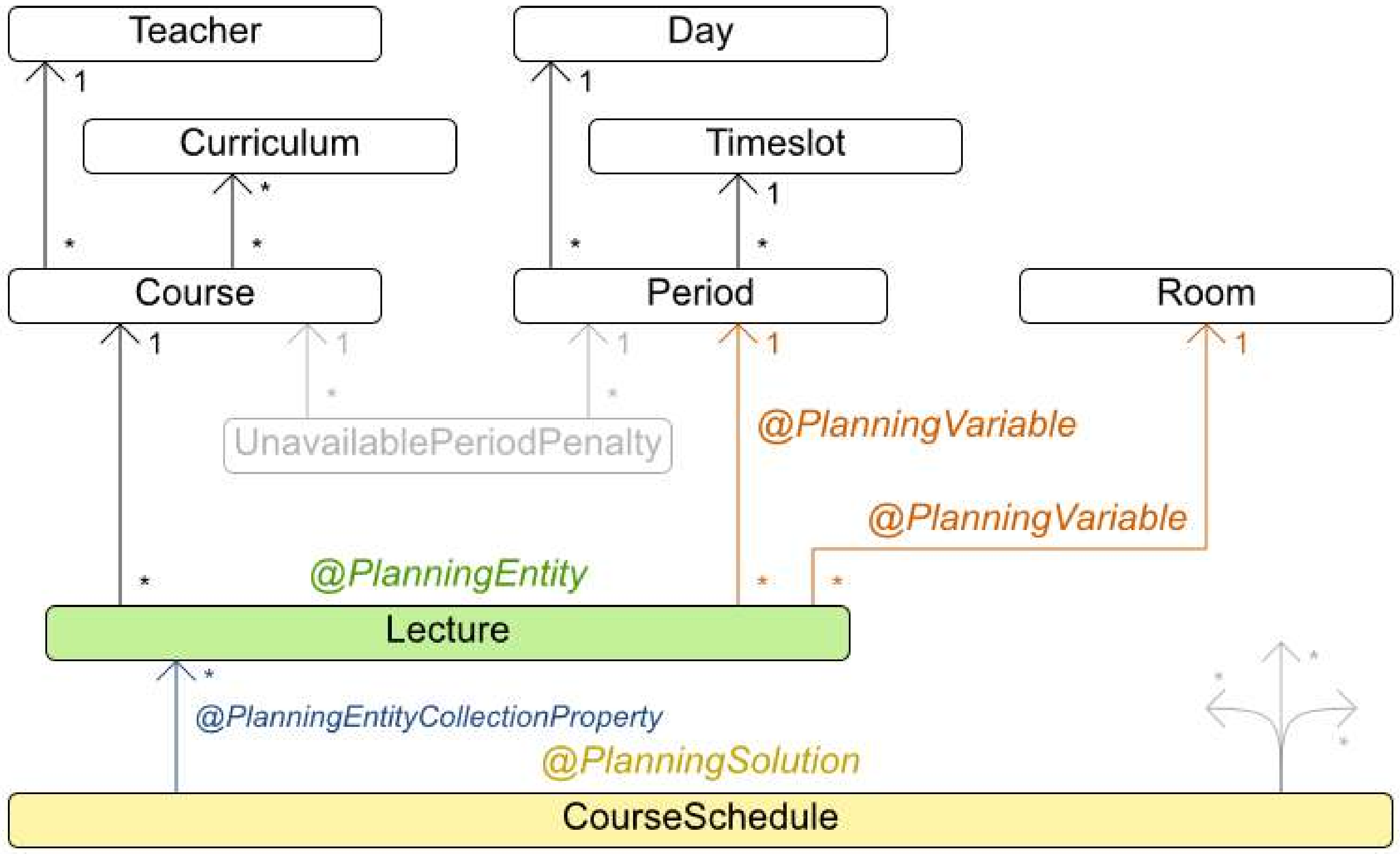


https://permid.org/

1.4 KNOWLEDGE REPRESENTATION

Domain Ontology / OO Classes / DB Schema

Curriculum course class diagram



1.4 KNOWLEDGE REPRESENTATION

Rules (A form of logic: propositional or first order)

- **Represent problem-solving knowledge as**

  “**IF/WHEN** … **THEN** …” rules

  - Is the classic technique for representing domain knowledge in an machine reasoning system
  - Is also a very natural way of human decision making
- **A rule consists of two parts:**
  - The **IF** part
    - called the **antecedent** or **premise** or **condition**
  - The **THEN** part
    - called the **consequent** or **conclusion** or **action**

1.4 KNOWLEDGE REPRESENTATION

Rules

- **Basic Rule Syntax**

IF	<antecedent>
THEN	<consequent>

IF	person X is ill
THEN	person X need rest a lot

1.4 KNOWLEDGE REPRESENTATION

Rules

- **Multi-antecedent Rule**

IF	<antecedent 1>
AND/OR	<antecedent 2>
	…
AND/OR	<antecedent n>
THEN	<consequent>

IF	person X is ill
AND	person X is a lecturer
THEN	person X cannot rest at home, but go to class

1.4 KNOWLEDGE REPRESENTATION

Rules

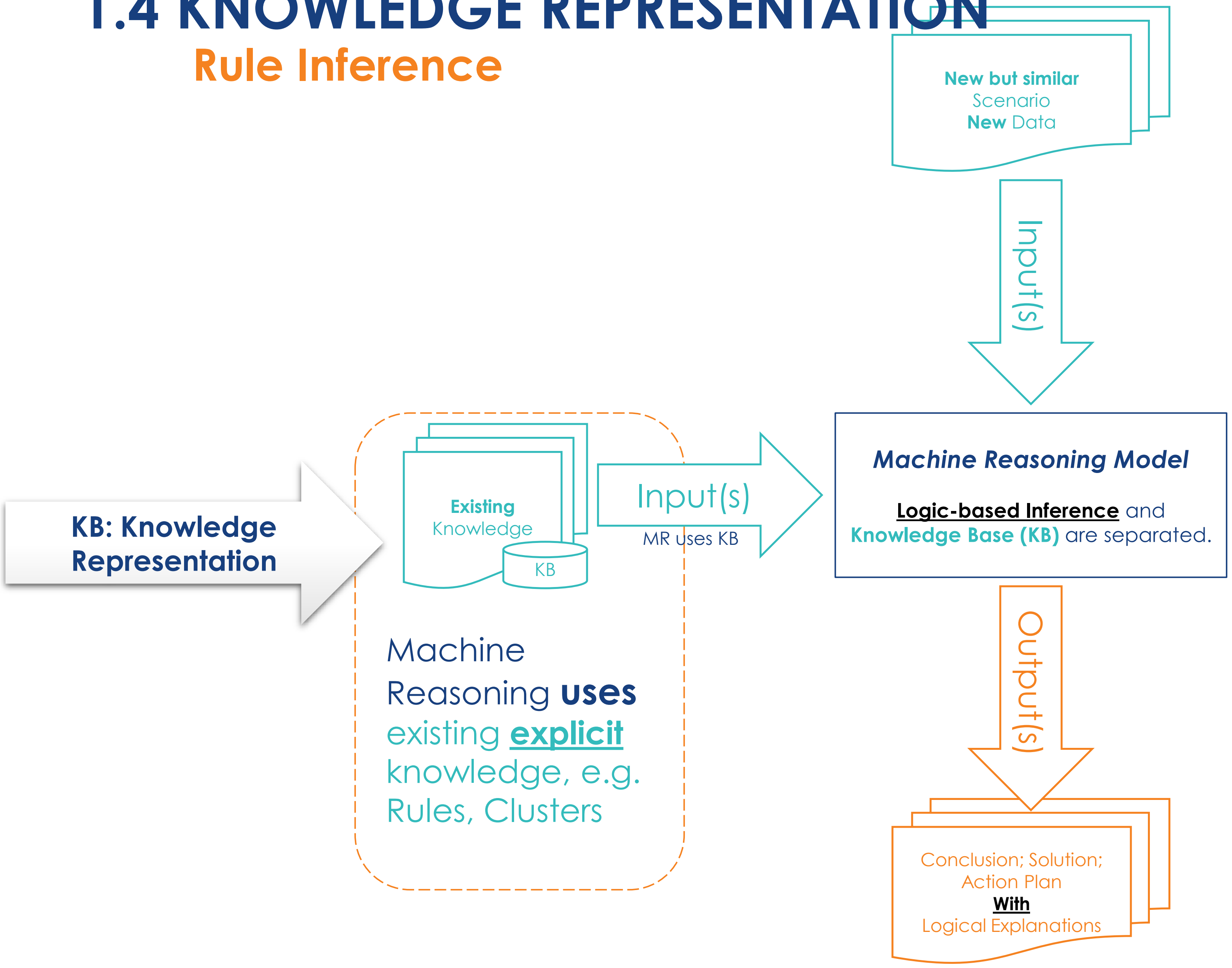
- **Multi-consequent Rule**

```
IF      <antecedent 1>
THEN    <consequent 1>
        <consequent 2>
              …
        <consequent m>
```

☺ The relationship between the multiple consequents is understood as AND, depending on the implementation of software for developing reasoning systems.

1.4 KNOWLEDGE REPRESENTATION

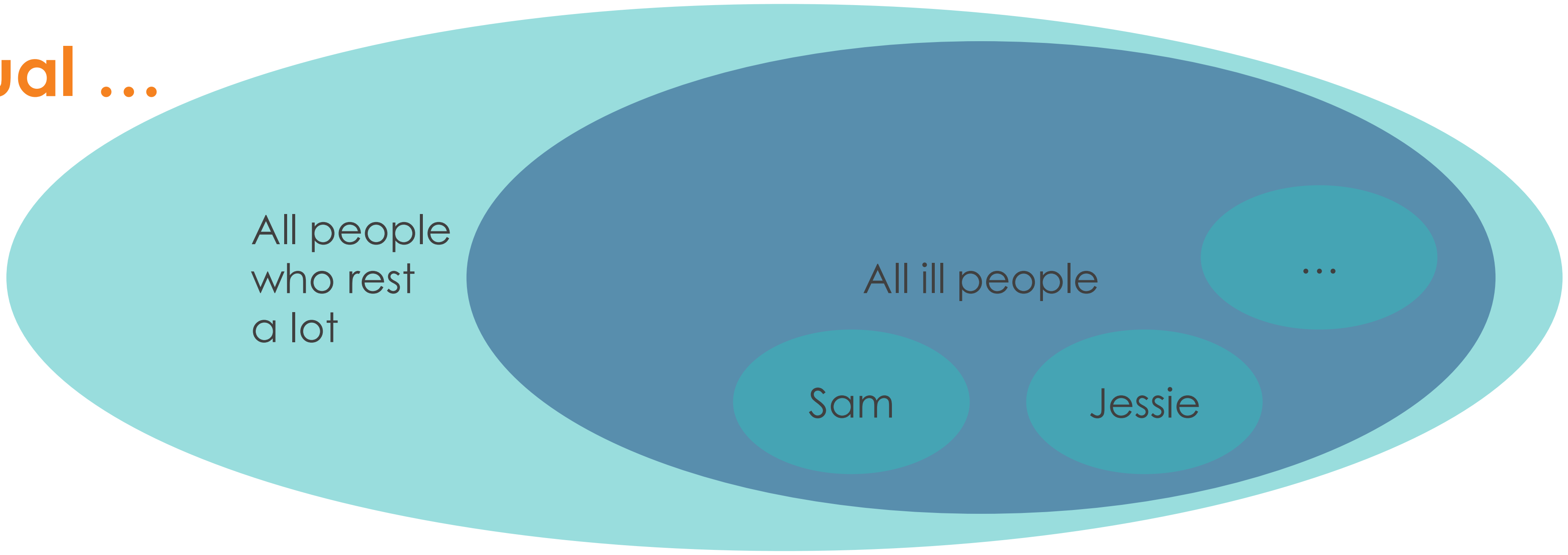
Rule Inference


New but similar
Scenario
New Data
Input(s)
KB: Knowledge Representation
Existing Knowledge
KB
Input(s)
MR uses KB
Machine Reasoning Model
Logic-based Inference and
Knowledge Base (KB) are separated.
Machine Reasoning uses existing explicit knowledge, e.g. Rules, Clusters
Output(s)
Conclusion; Solution; Action Plan
With
Logical Explanations

1.4 KNOWLEDGE REPRESENTATION

Rule Inference

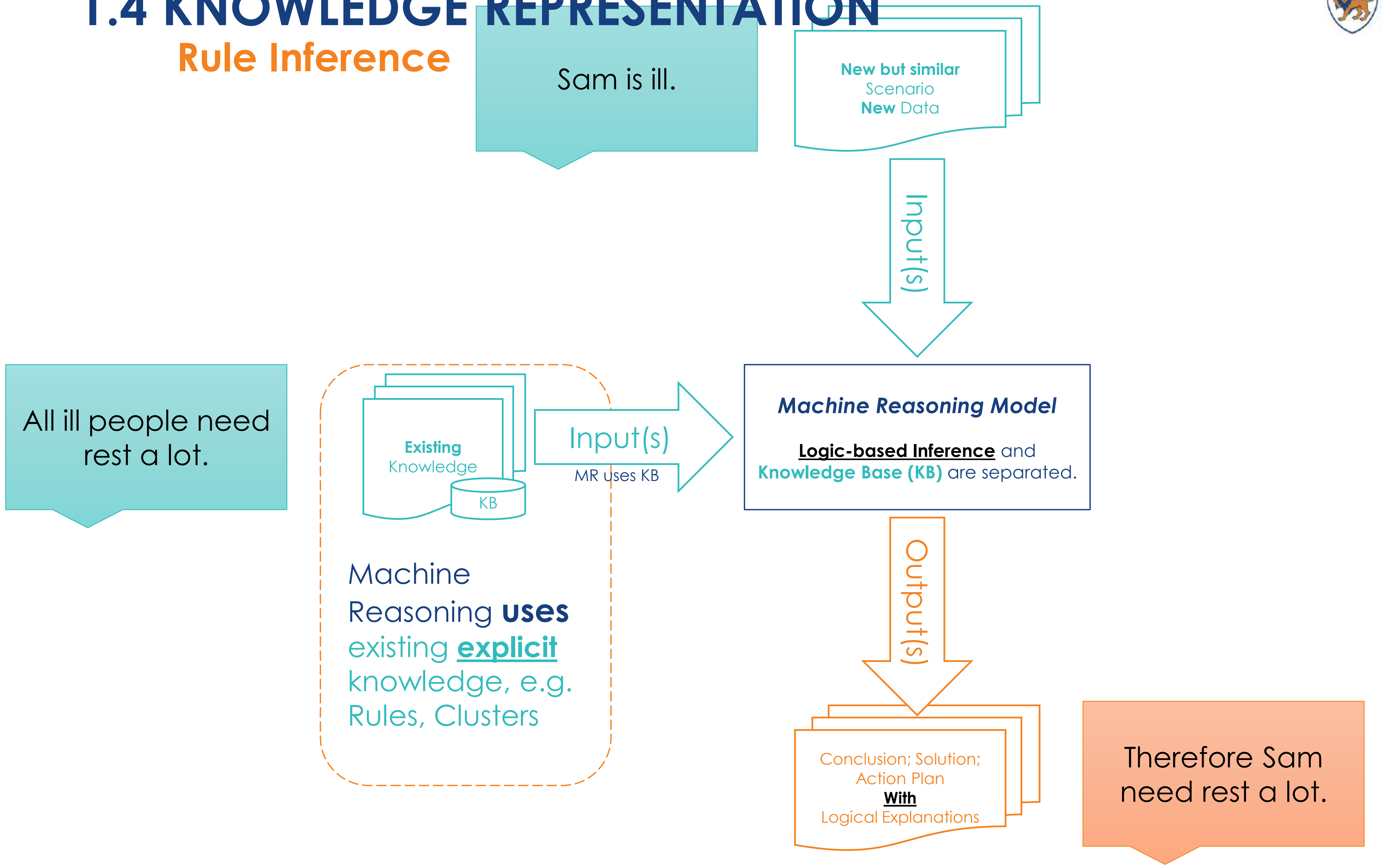
- **Knowledge/Rule** : **All ill people** need rest a lot.
- **Individual 1** : **Sam** is **ill**, therefore he need rest a lot.
- **Individual 2** : **Jessie** is **ill**, therefore she need rest a lot.
- **Individual ...**



☺ **Reasoning Rationality: Universal → Individual**

1.4 KNOWLEDGE REPRESENTATION

Rule Inference



Sam is ill.
New but similar
Scenario
New Data
Input(s)
All ill people need rest a lot.
Existing
Knowledge
KB
Input(s)
MR uses KB
Machine Reasoning Model
Logic-based Inference and
Knowledge Base (KB) are separated.
Machine Reasoning uses existing explicit knowledge, e.g. Rules, Clusters
Output(s)
Conclusion; Solution;
Action Plan
With
Logical Explanations
Therefore Sam need rest a lot.

1.4 KNOWLEDGE REPRESENTATION

Rules

- **Example rules in application**

IF	‘age of the customer’ < 18
AND	‘cash withdrawal’ > $1,000
THEN	‘signature of the parent’ is required

IF	‘taxable income’ > $16,238
THEN	‘Medicare levy’ = ‘taxable income’ * 1.5 %

1.4 KNOWLEDGE REPRESENTATION

Rules

- **Rule Types Example:**

  Rules can represent relations, recommendations/directives and heuristics

  - **Relation**

    IF The 'fuel tank' is empty
    THEN The engine will not start

  - **Recommendation**

    IF The season is autumn
    AND The sky is cloudy
    AND The forecast is drizzle
    THEN The advice is 'take an umbrella'

  - **Heuristic**

    IF PIE is jammed
    THEN Switch to AYE (or ask for working from home)

1.4 KNOWLEDGE REPRESENTATION

Rules

- **Rules (business knowledge) in Rule/Process Reasoning System are designed as mutually independent**

  Each rule represents a single chunk of knowledge

  - **IF** pet_size = medium **THEN** recommend = cats or small dogs

- **Rules are based on a priori knowledge or heuristics**

  Rules are derived from domain experts who uses experiential knowledge and "rules-of-thumb"

  - **IF** buyer = female **THEN** recommend = hamster

- **Rules can incorporate uncertainties**

  Real life business situations are plagued with uncertainties that make decision-making difficult (or flexible)

  - **IF** work = long_hours **THEN** recommend = dog (30% confidence in rule conclusion)

1.4 KNOWLEDGE REPRESENTATION

Exercise 2.1

- **Convert the following knowledge about animals into WHEN/THEN rules:**
  1. animals with hair as their body covering are mammals
  2. animals that feed their young with milk are mammals
  3. animals with feathers as their body covering are birds
  4. animals that fly and reproduce by eggs are birds
  5. mammals that eat meat are carnivores
  6. mammals with pointed teeth, claws on their feet, and eyes that point forward are carnivores
  7. mammals that eat grass are herbivores
  8. mammals with hooves on their feet are herbivores
  9. carnivores that have a tawny colour and dark spots as their marking are cheetahs
  10. carnivores that have a tawny colour and dark stripes as their marking are tigers
  11. herbivores that have a tawny colour and dark spots as their marking and long necks are giraffes
  12. herbivores that have a black and white colour are zebras
  13. birds that walk and are black and white and have a long neck are ostriches
  14. birds that swim and are black and white are penguins
  15. birds that fly and are black and white are albatrosses

1.4 KNOWLEDGE REPRESENTATION
Rules – KIE Drools

a driving license application.

```
public class Applicant {
    private String name;
    private int age;
    private boolean valid;
    // getter and setter methods here
}
```

Now that we have our data model we can write our first rule. We assume that the application uses rules to reject invalid applications. As this is a simple validation use case we will add a single rule to disqualify any applicant younger than 18.

```
package com.company.license

rule "Is of valid age"
when
    $a : Applicant( age < 18 )
then
    $a.setValid( false );
end
```

To make the Drools engine aware of data, so it can be processed against the rules, we have to ***insert*** the data, much like with a database. When the Applicant instance is inserted into the Drools engine it is evaluated against the constraints of the rules, in this case just two constraints for one rule. We say ***two*** because the type **Applicant** is the first object type constraint, and **age < 18** is the second field constraint. **An object type constraint plus its zero or more field constraints is referred to as a pattern**. When an inserted instance satisfies both the object type constraint and all the field constraints, it is said to be matched. The **$a** is a binding variable which permits us to reference the matched object in the consequence. There its properties can be updated. The dollar character ('$') is optional, but it helps to differentiate variable names from field names. The process of matching patterns against the inserted data is, not surprisingly, often referred to as ***pattern matching***.

Source https://docs.jboss.org/drools/release/latest/drools-docs/html_single/index.html#_methods_versus_rules

4.1.3. Methods versus Rules

People often confuse methods and rules, and new rule users often ask, "How do I call a rule?" After the last section, you are now feeling like a rule expert and the answer to that is obvious, but let's summarize the differences nonetheless.

```
public void helloWorld(Person person) {
    if ( person.getName().equals( "Chuck" ) ) {
        System.out.println( "Hello Chuck" );
    }
}
```

- Methods are called directly.
- Specific instances are passed.
- One call results in a single execution.

```
rule "Hello World" when
    Person( name == "Chuck" )
then
    System.out.println( "Hello Chuck" );
end
```

- Rules execute by matching against any data as long it is inserted into the Drools engine.
- Rules can never be called directly.
- Specific instances cannot be passed to a rule.
- Depending on the matches, a rule may fire once or several times, or not at all.

Source https://docs.jboss.org/drools/release/latest/drools-docs/html_single/index.html#_methods_versus_rules

4.1.4. Cross Products

Earlier the term "cross product" was mentioned, which is the result of a join. Imagine for a moment that the data from the fire alarm example were used in combination with the following rule where there are no field constraints:

```
rule "Show Sprinklers" when
    $room : Room()
    $sprinkler : Sprinkler()
then
    System.out.println( "room:" + $room.getName() +
                        " sprinkler:" + $sprinkler.getRoom().getName() );
end
```

In SQL terms this would be like doing `select * from Room, Sprinkler` and every row in the Room table would be joined with every row in the Sprinkler table resulting in the following

```
room:office sprinkler:office
room:office sprinkler:kitchen
room:office sprinkler:livingroom
room:office sprinkler:bedroom
room:kitchen sprinkler:office
room:kitchen sprinkler:kitchen
room:kitchen sprinkler:livingroom
room:kitchen sprinkler:bedroom
room:livingroom sprinkler:office
room:livingroom sprinkler:kitchen
room:livingroom sprinkler:livingroom
room:livingroom sprinkler:bedroom
room:bedroom sprinkler:office
room:bedroom sprinkler:kitchen
room:bedroom sprinkler:livingroom
room:bedroom sprinkler:bedroom
```

These cross products can obviously become huge, and they may very well contain spurious data. The size of cross products is often the source of performance problems for new rule authors. From this it can be seen that it's always desirable to constrain the cross products, which is done with the variable constraint.

```
rule
when
    $room : Room()
    $sprinkler : Sprinkler( room == $room )
then
    System.out.println( "room:" + $room.getName() +
                        " sprinkler:" + $sprinkler.getRoom().getName() );
end
```

This results in just four rows of data, with the correct Sprinkler for each Room. In SQL (actually HQL) the corresponding query would be `select * from Room, Sprinkler where Room == Sprinkler.room`.

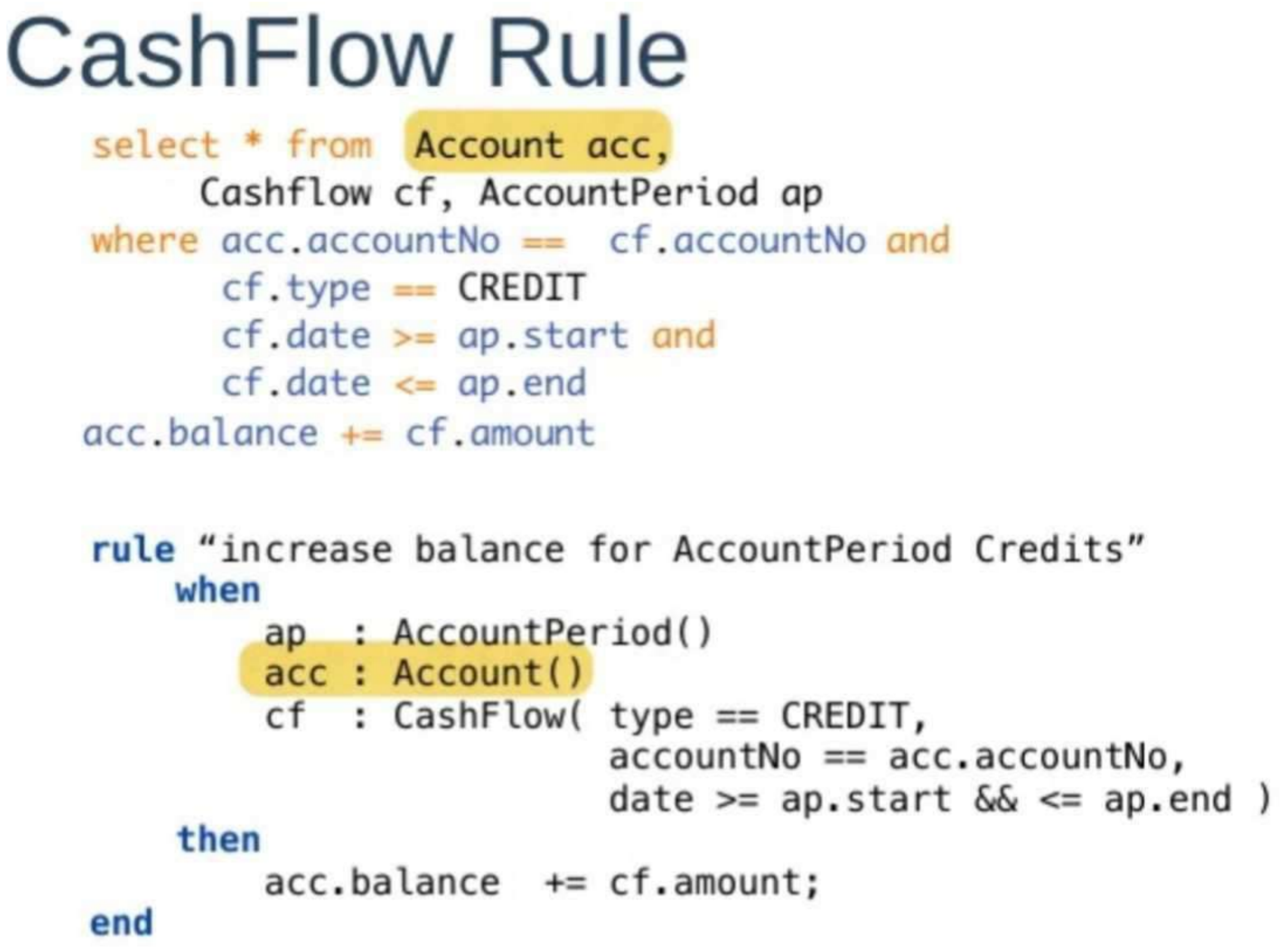
```
room:office sprinkler:office
room:kitchen sprinkler:kitchen
room:livingroom sprinkler:livingroom
room:bedroom sprinkler:bedroom
```

Source https://docs.jboss.org/drools/release/latest/drools-docs/html_single/index.html#_cross_products

1.4 KNOWLEDGE REPRESENTATION

Rules – KIE Drools

- KIE Drools rule is declarative language. It's functional similar to structured query language SQL.
- In logical reasoning context, When/Then rules (in knowledge base) are considered universally true, thus can be 'declared'.



CashFlow Rule

```
select * from  Account acc,
        Cashflow cf, AccountPeriod ap
where acc.accountNo ==  cf.accountNo and
        cf.type == CREDIT
        cf.date >= ap.start and
        cf.date <= ap.end
acc.balance += cf.amount


rule "increase balance for AccountPeriod Credits"
    when
        ap  : AccountPeriod()
        acc : Account()
        cf  : CashFlow( type == CREDIT,
                        accountNo == acc.accountNo,
                        date >= ap.start && <= ap.end )
    then
        acc.balance  += cf.amount;
end
```

Two rules can be used to determine the debit and credit for that quarter and update the Account balance. The two rules below constrain the cashflows for an account for a given time period. Notice the "&&" which use short cut syntax to avoid repeating the field name twice.

```
rule "increase balance for credits"
when
  ap : AccountPeriod()
  acc : Account( $accountNo : accountNo )
  CashFlow( type == CREDIT,
            accountNo == $accountNo,
            date >= ap.start && <= ap.end,
            $amount : amount )
then
  acc.balance  += $amount;
end
```

```
rule "decrease balance for debits"
when
  ap : AccountPeriod()
  acc : Account( $accountNo : accountNo )
  CashFlow( type == DEBIT,
            accountNo == $accountNo,
            date >= ap.start && <= ap.end,
            $amount : amount )
then
  acc.balance -= $amount;
end
```

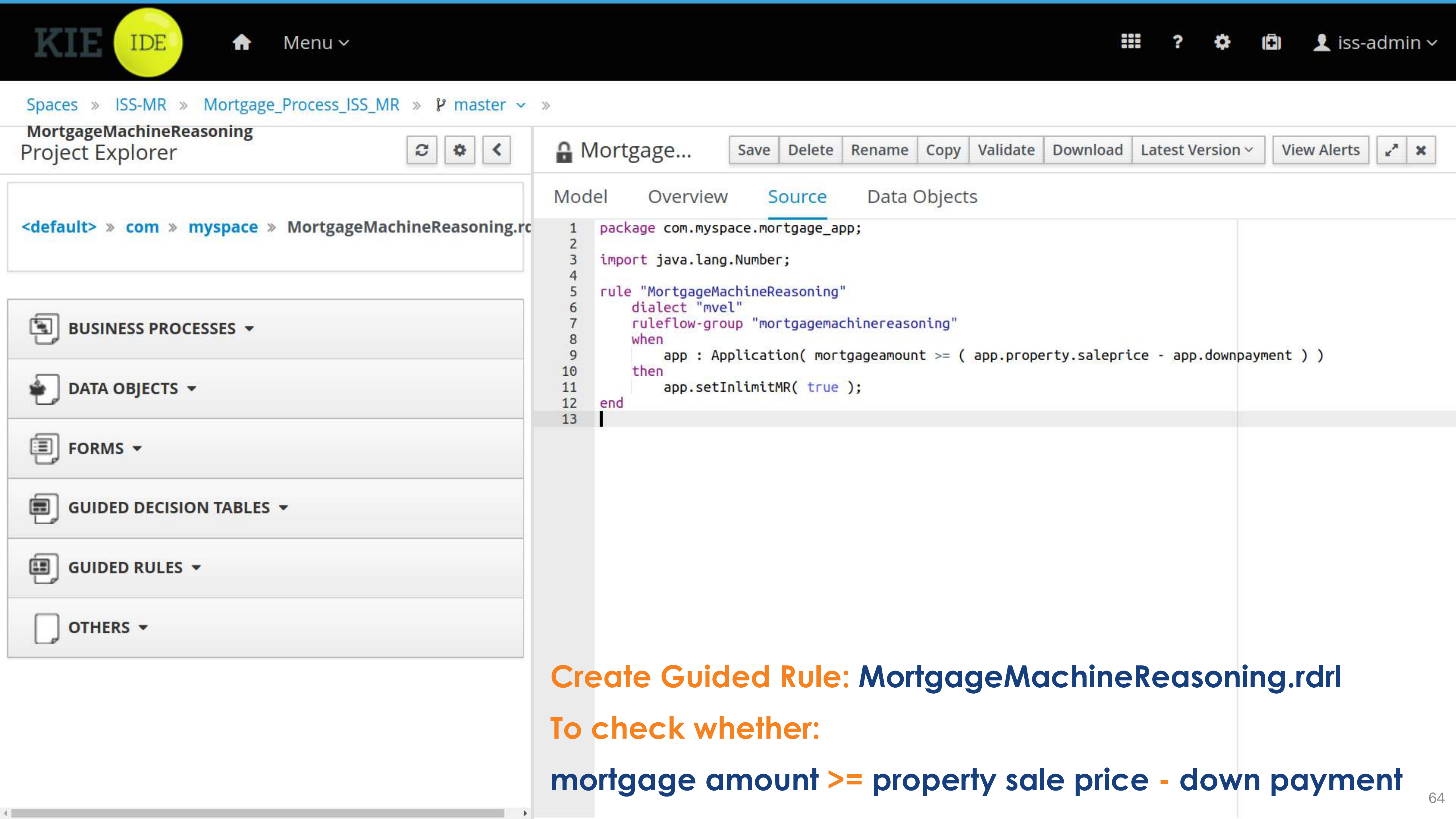
Earlier we showed how rules would equate to SQL, which can often help people with an SQL background to understand rules. The two rules above can be represented with two views and a trigger for each view, as below:

```
select * from Account acc,
              Cashflow cf,
              AccountPeriod ap
where acc.accountNo == cf.accountNo and
      cf.type == CREDIT and
      cf.date >= ap.start and
      cf.date <= ap.end
```

```
select * from Account acc,
              Cashflow cf,
              AccountPeriod ap
where acc.accountNo == cf.accountNo and
      cf.type == DEBIT and
      cf.date >= ap.start and
      cf.date <= ap.end
```

```
trigger : acc.balance += cf.amount
```

```
trigger : acc.balance -= cf.amount
```



```
package com.myspace.mortgage_app;

import java.lang.Number;

rule "MortgageMachineReasoning"
    dialect "mvel"
    ruleflow-group "mortgagemachinereasoning"
    when
        app : Application( mortgageamount >= ( app.property.saleprice - app.downpayment ) )
    then
        app.setInlimitMR( true );
end
```

Create Guided Rule: MortgageMachineReasoning.rdrl

To check whether:

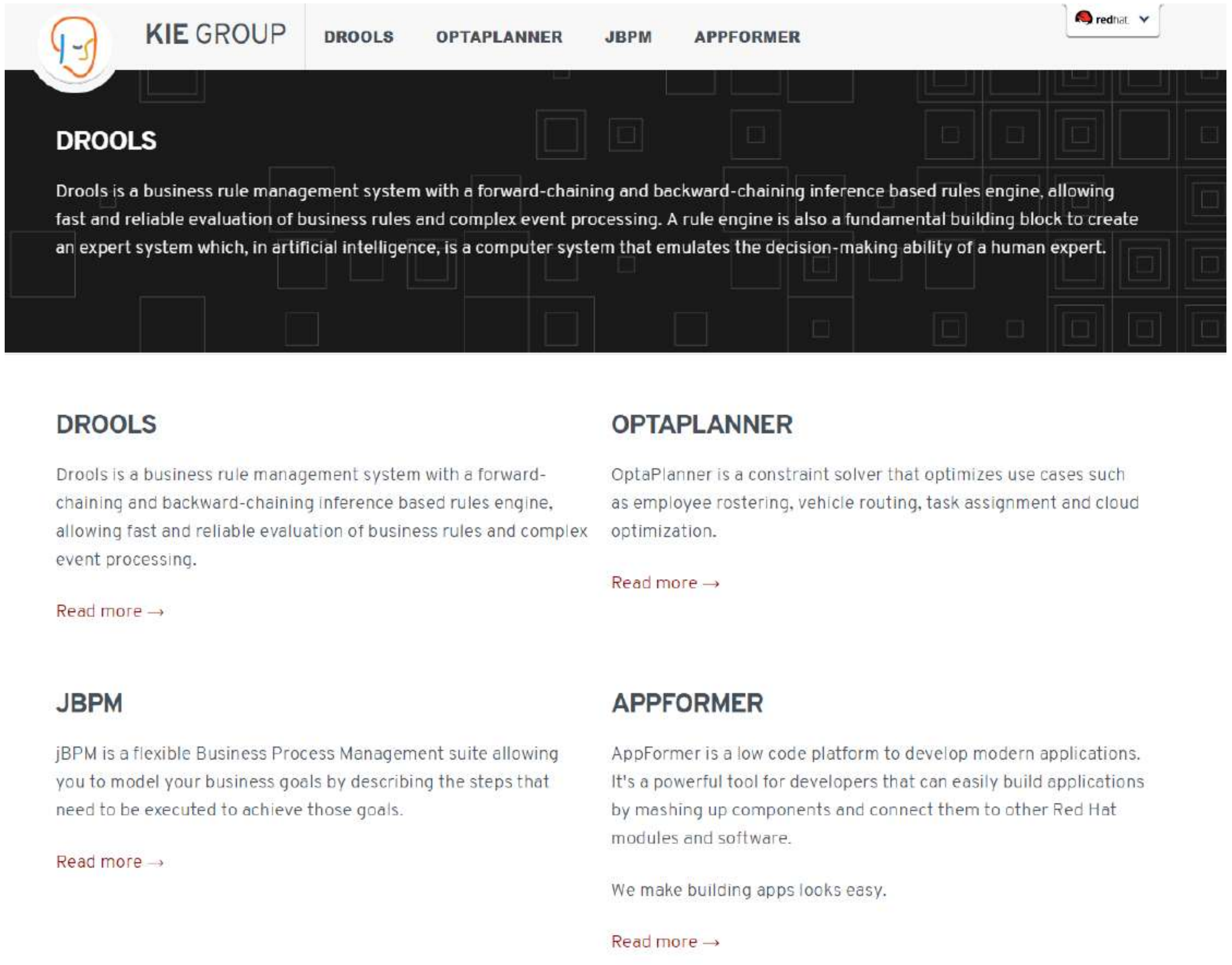
mortgage amount >= property sale price - down payment

1.5 WORKSHOP

RULE/PROCESS REASONING SYSTEM

1.5 WORKSHOP RULE/PROCESS REASONING SYSTEM

KIE BPMS/BRMS Suite – Workshop Tools



DROOLS

Drools is a business rule management system with a forward-chaining and backward-chaining inference based rules engine, allowing fast and reliable evaluation of business rules and complex event processing. A rule engine is also a fundamental building block to create an expert system which, in artificial intelligence, is a computer system that emulates the decision-making ability of a human expert.

DROOLS

Drools is a business rule management system with a forward-chaining and backward-chaining inference based rules engine, allowing fast and reliable evaluation of business rules and complex event processing.

Read more →

OPTAPLANNER

OptaPlanner is a constraint solver that optimizes use cases such as employee rostering, vehicle routing, task assignment and cloud optimization.

Read more →

JBPM

jBPM is a flexible Business Process Management suite allowing you to model your business goals by describing the steps that need to be executed to achieve those goals.

Read more →

APPFORMER

AppFormer is a low code platform to develop modern applications. It's a powerful tool for developers that can easily build applications by mashing up components and connect them to other Red Hat modules and software.

We make building apps looks easy.

Read more →

JBoss KIE

http://www.kiegroup.org/

JBoss KIE DROOLS

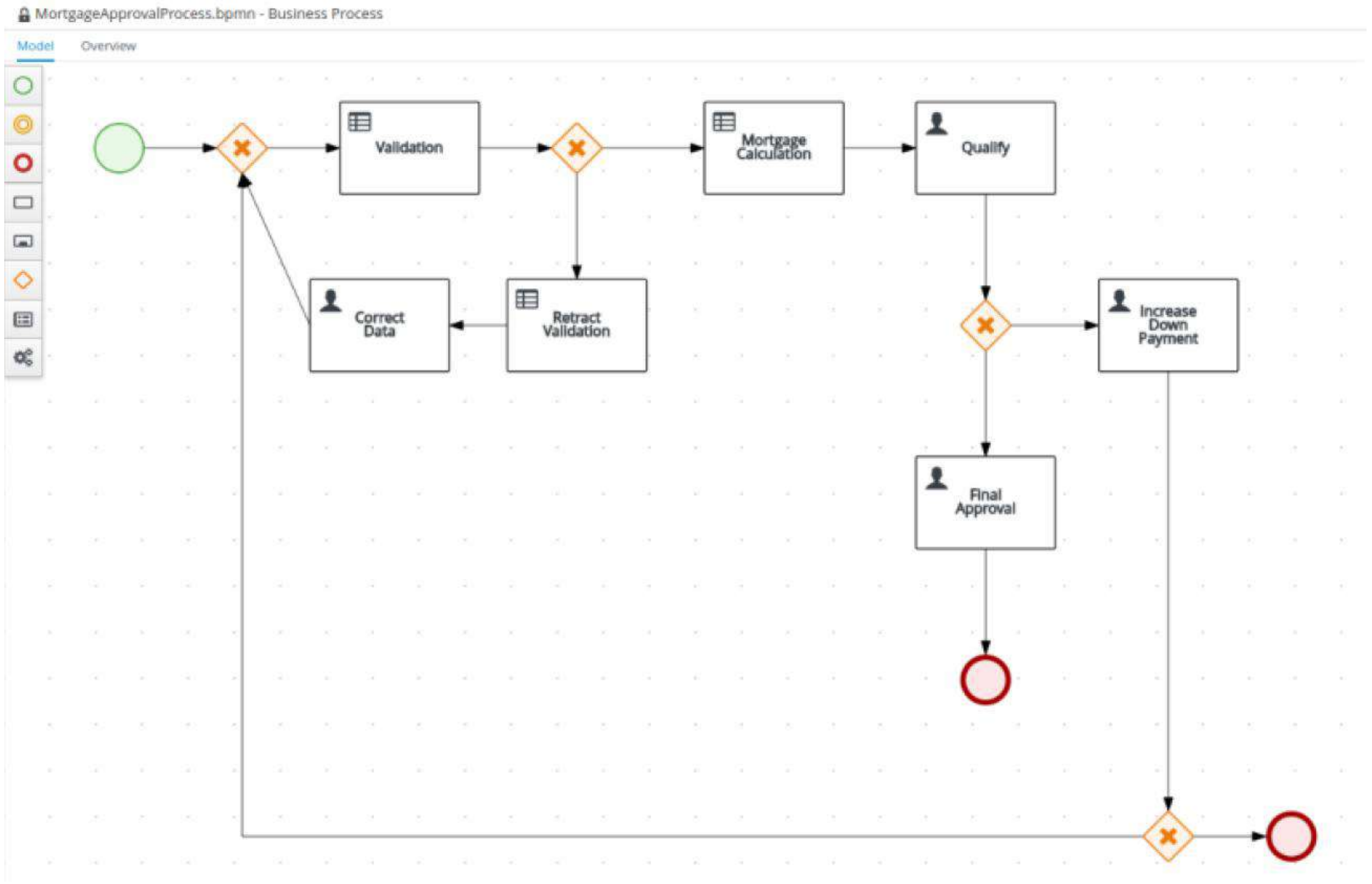
http://www.drools.org/

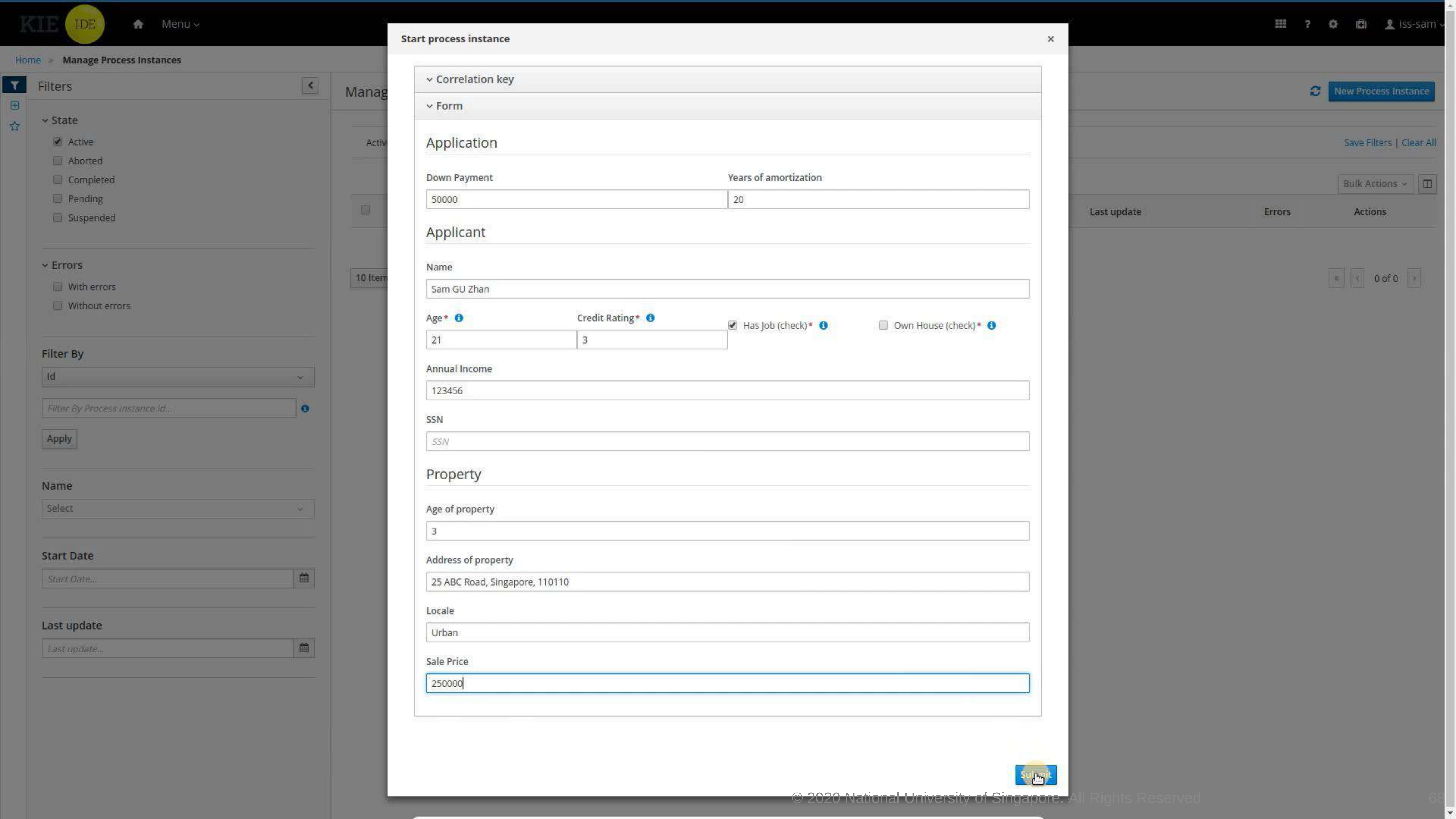
JBoss KIE JBPM

http://www.jbpm.org/

1.5 WORKSHOP RULE/PROCESS REASONING SYSTEM

KIE BPMS/BRMS Suite – Mortgage application systems





Start process instance

Correlation key

Form

Application

Down Payment	50000
Years of amortization	20

Applicant

Name	Sam GU Zhan
Age*	21
Credit Rating*	3
Has Job (check)*	✓
Own House (check)*	
Annual Income	123456
SSN	SSN

Property

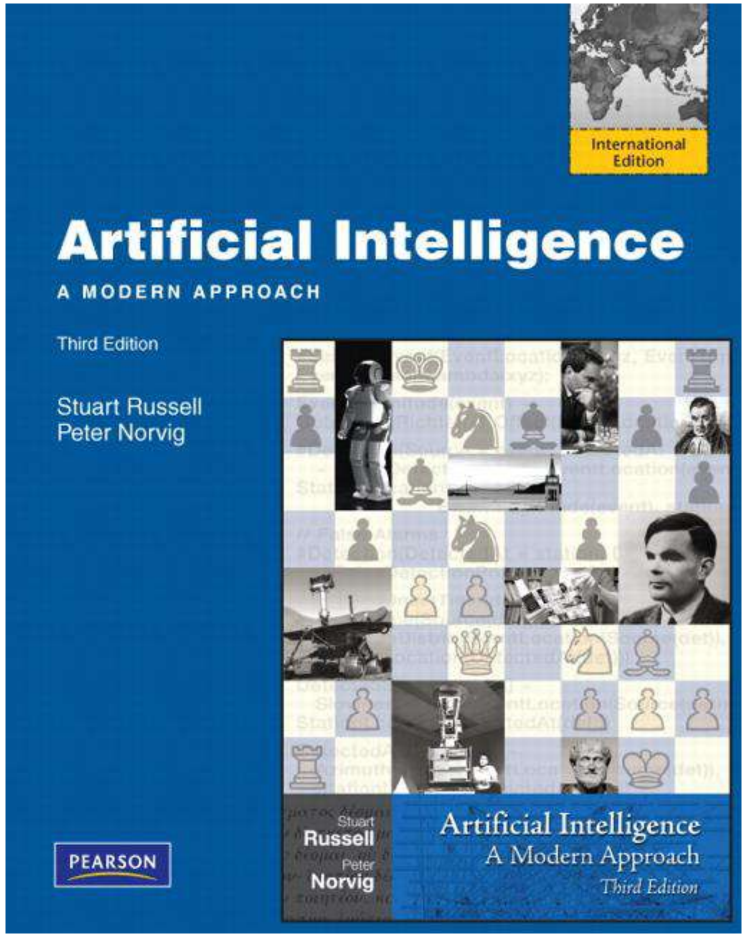
Age of property	3
Address of property	25 ABC Road, Singapore, 110110
Locale	Urban
Sale Price	250000

1.5 WORKSHOP RULE/PROCESS REASONING SYSTEM

- Refer to S-MR Workshop Guide.pdf

DAY 1 REFERENCE

Drools and jBPM tools	Eclipse plugins and support for Drools, jBPM and Guvnor functionality. Distribution zip contains binaries and sources.	Distribution ZIP



1. Getting Started With Business Processes (PAM / jBPM)

   https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.2/html-single/getting_started_with_business_processes/

2. Getting Started With Decision Services (DM / Drools)

   https://access.redhat.com/documentation/en-us/red_hat_process_automation_manager/7.2/html-single/getting_started_with_decision_services/

3. KIE Workbench Tutorial : Data Object, Form, Task and Process creation

   https://www.youtube.com/watch?v=xQqxhEcrFB0

   https://www.youtube.com/watch?v=US5tG4ZUPg0

4. KIE Drools Official Tutorial

   https://www.drools.org/learn/video.html

   https://www.drools.org/learn/slides.html

5. KIE Drools On Boarding Course (Java & Eclipse)

   https://nheron.gitbooks.io/droolsonboarding/content/

6. KIE Development Plugin for Eclipse IDE

   http://www.drools.org/download/download.html

DAY 1 SUMMARY

1.1 Machine Reasoning Overview

- Reasoning; Thinking; Learning; Cognition; Artificial Intelligence
- Problem Solving: Analytic Tasks vs. Synthetic Tasks

1.2 Reasoning Types

- Deductive; Inductive; Analogical; Adductive; Fuzzy

1.3 Reasoning System Architectures

- Proactive Goal Driven Systems vs. Reactive Data Driven Systems
- Knowledge Based System (Knowledge Base + Inference Engine)

1.4 Knowledge Representation

- Natural Language; Formula; Formal Logic; Semantic Web; Knowledge Graph; Ontology; Database
- Propositional Logic; First Order Logic; Rules

1.5 Rule/Process Reasoning System Workshop

END OF LECTURE NOTES

APPENDICES

WORKSHOP EXAMPLE SYSTEM: HDB BTO RECOMMENDER

1

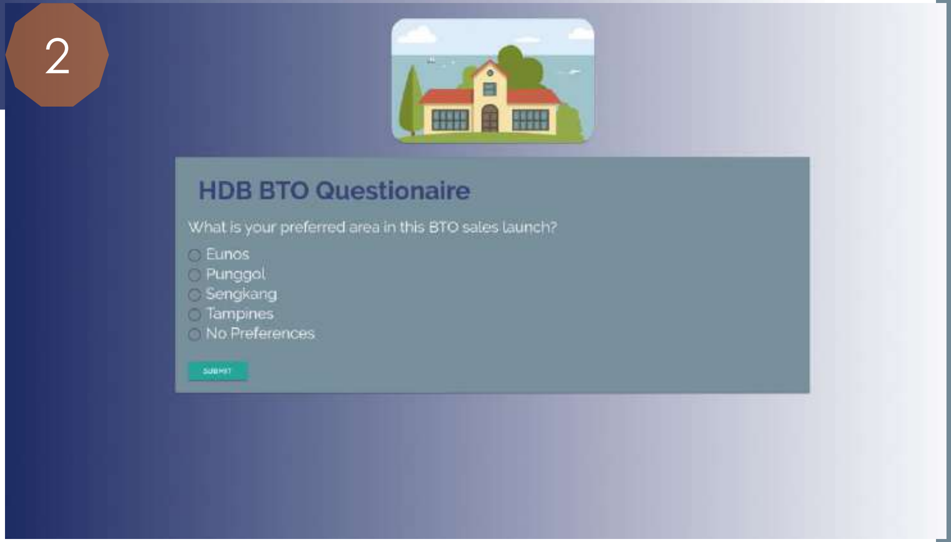
Welcome

HDB BTO Recommender System

Your Trusted Advisor To BTO Units

GET STARTED

2



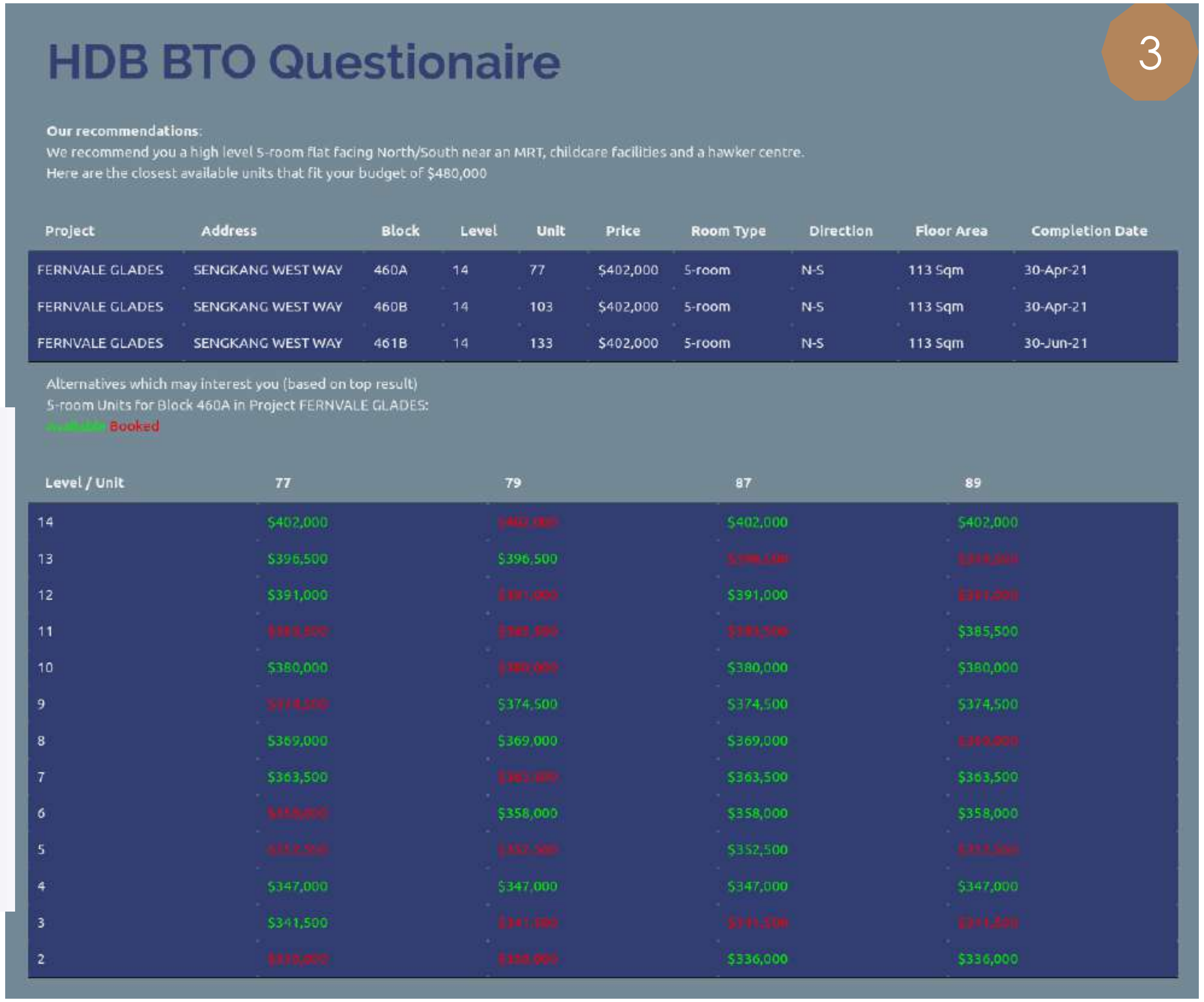
HDB BTO Questionaire

What is your preferred area in this BTO sales launch?

- Eunos
- Punggol
- Sengkang
- Tampines
- No Preferences

SUBMIT

3



HDB BTO Questionaire

Our recommendations:

We recommend you a high level 5-room flat facing North/South near an MRT, childcare facilities and a hawker centre.

Here are the closest available units that fit your budget of $480,000

Project	Address	Block	Level	Unit	Price	Room Type	Direction	Floor Area	Completion Date
FERNVALE GLADES	SENGKANG WEST WAY	460A	14	77	$402,000	5-room	N-S	113 Sqm	30-Apr-21
FERNVALE GLADES	SENGKANG WEST WAY	460B	14	103	$402,000	5-room	N-S	113 Sqm	30-Apr-21
FERNVALE GLADES	SENGKANG WEST WAY	461B	14	133	$402,000	5-room	N-S	113 Sqm	30-Jun-21

Alternatives which may interest you (based on top result)

5-room Units for Block 460A in Project FERNVALE GLADES:

Available Booked

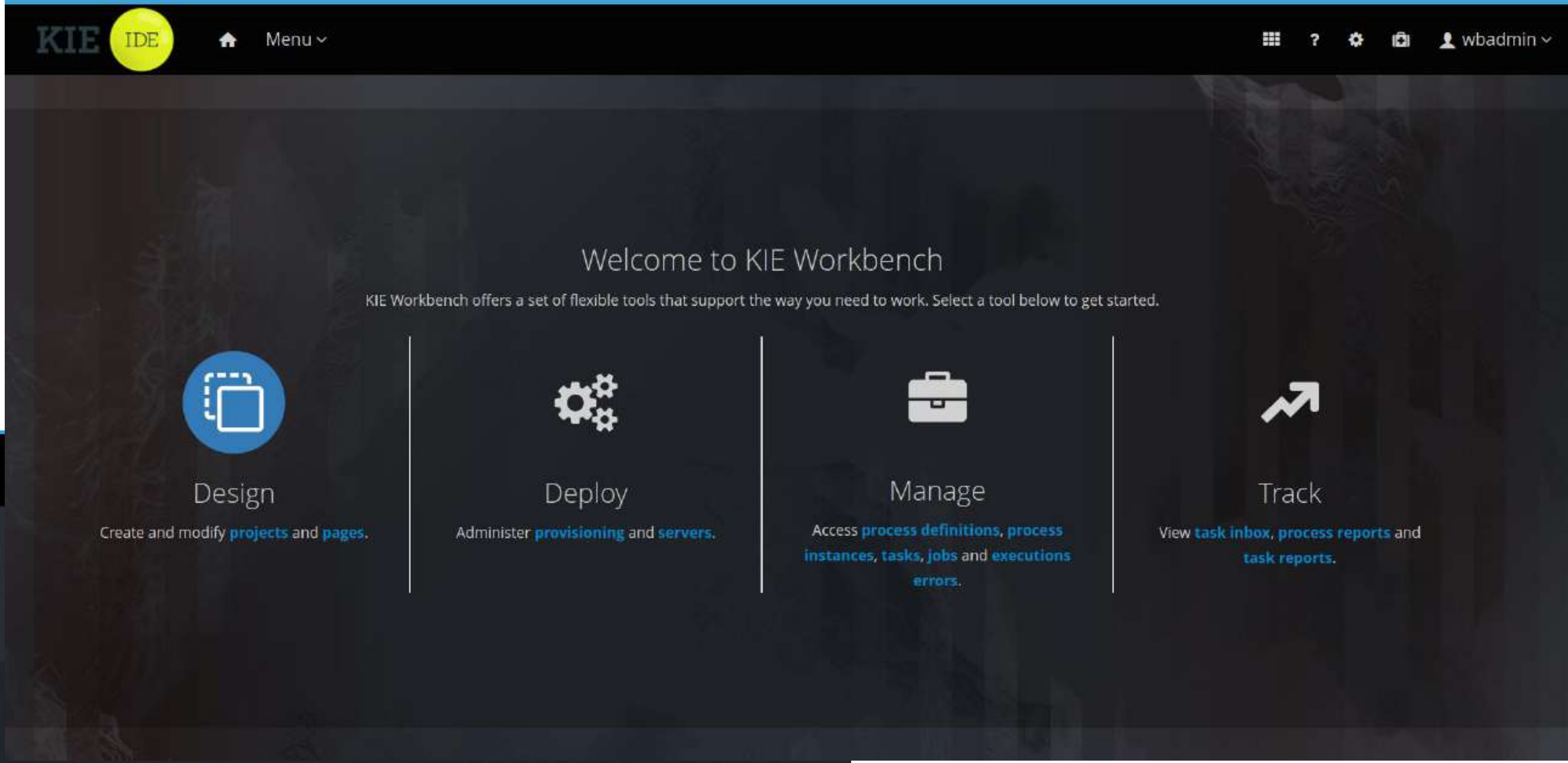
Level / Unit	77	79	87	89
14	$402,000	$402,000	$402,000	$402,000
13	$396,500	$396,500	$396,500	$396,500
12	$391,000	$391,000	$391,000	$391,000
11	$385,500	$385,500	$385,500	$385,500
10	$380,000	$380,000	$380,000	$380,000
9	$374,500	$374,500	$374,500	$374,500
8	$369,000	$369,000	$369,000	$369,000
7	$363,500	$363,500	$363,500	$363,500
6	$358,000	$358,000	$358,000	$358,000
5	$352,500	$352,500	$352,500	$352,500
4	$347,000	$347,000	$347,000	$347,000
3	$341,500	$341,500	$341,500	$341,500
2	$336,000	$336,000	$336,000	$336,000

Demo System : http://www.bit.ly/iss-vm

Source Code : https://github.com/telescopeuser/bto-recommender-system

PROCESS AUTOMATION MANAGER (PAM / JBPM) & DECISION MANAGER (DM / DROOLS)

Key Customer Case Studies



Welcome to KIE Workbench

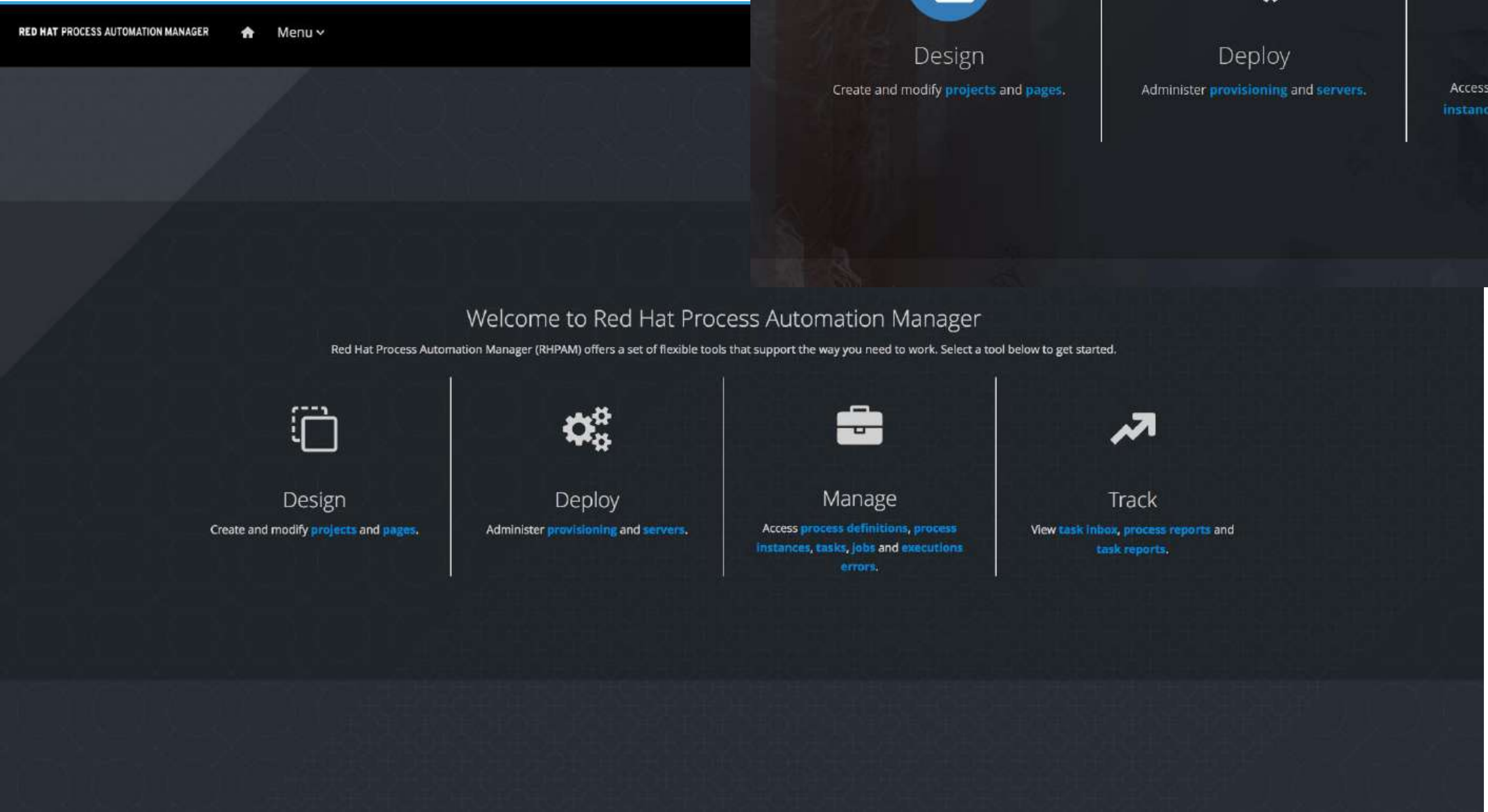
KIE Workbench offers a set of flexible tools that support the way you need to work. Select a tool below to get started.

Design
Create and modify projects and pages.

Deploy
Administer provisioning and servers.

Manage
Access process definitions, process instances, tasks, jobs and executions errors.

Track
View task inbox, process reports and task reports.



Welcome to Red Hat Process Automation Manager

Red Hat Process Automation Manager (RHPAM) offers a set of flexible tools that support the way you need to work. Select a tool below to get started.

Design
Create and modify projects and pages.

Deploy
Administer provisioning and servers.

Manage
Access process definitions, process instances, tasks, jobs and executions errors.

Track
View task inbox, process reports and task reports.

Aviva Achieves Faster Response Times - Courtesy: Red Hat PAM

AVIVA™

Aviva leveraged Fuse to manage additional service endpoints and establish a service gateway for routing and service integration with third-party vendor systems, such as Kofax and IBM FileNet.

CHALLENGE

The inflexible bond management and workflow systems of Aviva's new acquisition FPI, was hampering it's goal of bringing the new joint offering to the Asian market faster.

SOLUTION

- Aviva decided to migrate from FPI's existing AWD system to a new & faster imaging and workflow application based on **Red Hat Process Automation Manager (PAM)**.
- Standardized on a single process automation platform leveraging **PAM** to unify applications across users in Singapore, Hong Kong and Dubai, in 6 months.

New Services at a Faster Rate

Lower Overall Costs

Faster Response Times

Case Study

Jalisco State Government, Powered by Red Hat Middleware, Increases Service Rate by 900%

JALISCO
GOBIERNO DEL ESTADO

“With the Red Hat Solution, we can gradually scale up and grow in line with demand for our services.This capability is quite extraordinary, because we can add other solutions that permit further improvement of our servicesand building innovative applications ”

- MASTER MARIA ANGELINA ALARCON ROMERO
  DIRECTOR,TECHNOLOGY INNOVATION,
  GOVERNMENT OF THE STATE OF JALISCO

Case Study

CHALLENGE

With dispersed data sources and manual processes, Jalisco State was finding it challenging to cope with the increasing demands of its citizens

SOLUTION

With Red Hat Fuse connecting disparate systems and **BPM Suite** automating the complex processes, Jalisco State provided a wide range of services from mobility, public safety, and revenue collection to environmental issues. Citizens could pay road taxes or traffic fines, order birth certificates, or electronically sign receipts from one online location.

Traffic Infringement Notices Sent within 3 days rather than 120 days

Enhanced Security

Citizens Served Per Day Increased from 3,000 to 30,000

70 Million Seamless Ride Bookings Per Year Powered by Red Hat Decision Manager

LogistiCare®

“We anticipate moving from large software releases quarterly to functional releases monthly, with system refinements happening as often as weekly,” Our IT organization is now a responsive partner with a business focus.”

- MICHAEL QUINTERO
ENTERPRISE SOLUTIONS ARCHITECT
LOGISTICARE

Case Study

CHALLENGE

Business has grown over 60% and the core app was not able to handle the growing scale and complexity of business. The app contained custom business logic that made meeting the evolving needs of clients & partners, increasingly difficult.

SOLUTION

- Saved $6Mn in operations by using **DM** to define complex rules for regulatory compliance, routing, payments, and ride scheduling.
- Adopted the OpenShift container platform to manage, deploy and scale apps and APIs, built using Red Hat Middleware.

Easier Third Party Integration

Operational costs reduced by $6 Mn

Increased agent efficiency by 15%

What use cases do our MTech students apply?

How our learners apply the learnt from course:

Challenges

I work in a top **oil & gas** company as a **Pricing Tactics Advisor**. Among the big oil and gas firms and MNCs, our company is exceptionally known for their **rigor in business processes and controls**. For such a huge company, such traditional process and controls can hinder our speed to market, especially in the technology driven world. Many of our processes are **still very manual and excel based**, requiring many levels of endorsement and checks. These processes can be easily replaced with process automation tools.

What learnt is useful?

The **KIE tool** can easily replace many of the existing manual processes that we do. E.g. Email endorsements (sending to the right manager who has the right endorsement authority), request forms via email (many of these email request has missing info and we end up going back and forth. With a validation form, this will not happen).

Decision trees is also one of the relevant modules I have been using in my daily work as the Pricing Tactics Advisor when it comes to dynamic pricing.

How/Where to apply to workplace?

I have actually automated many of the existing **endorsement and email request** using Microsoft Sharepoint (paid software) workflow. It works in a similar concept as KIE.

I have been using JMP (paid software) to make numerous **dynamic/tactical pricing strategies**. Many of these strategies are first developed by exploring our historical competitors and transactional data using python in Jupyter notebook.

Business values

Our company has just begun on their **digital transformation** journey. I believe that the company of tomorrow is not one who has the most advanced technology, but the company who has the most amount of data. Many of the oil and gas firms are sleeping giants. We have a treasure trove of untapped data. It is a journey for us to move away from our oil and gas mind-set into the world of digital technology. I am leading the Asia Pacific Market Entry Strategy for China and we are **exploring new and lean ways** to do market entries without the baggage of traditional systems that the mothership is using.

How our learners apply the learnt from course:

Challenges

I build **credit risk** systems for the **bank**. The difficult part is:

1. There are huge number of credit policies, they **scattered around in many places**: within systems, excels, or in the human brains of those highly experienced account managers, they are **not synchronized**.

2. **Polices keep on changing** due to policies changing, regulatory requirements, etc. Some policies were built in the system and **logic were coded in programs**. It took very long time to adjust them as standard system development life cycle SDLC kicks in.

3. There is no centralized knowledge base which serve as the golden source of rules, there is also no centralized data taxonomy, which causes **conflicting results** and no one can tells which one is correct.

What learnt is useful?

The **KIE suite** is very useful in terms of **building and testing the policies** in the bank credit departments.

The **business users** could use the graphic tools to come up the flow, input their rules as guided decision table, auto generate the forms if input is required, and quickly start their testing on the new policies. (**rather than wait for tech team to finish the coding and test**)

For tech team, we could either code the tested policies into the legacy systems, or use the KIE suite to **expose the policies/rules into application programming interface API**, and simply call it.

This will significantly **shorten the turn around time** for new policies launch.

How/Where to apply to workplace?

Take the policies as example, we could easily **de-couple** the coding part and the logic part between development team and business team.

The business team could focus on the policies and come up the list of **mutually agreeable rules table in excel sheet**, track them in version management tools such git, to make all the rules traceable, and prevent multiple conflicting version.

Whenever there is need to change rules, the credit system will **simply load the excels and the polices are live**. **No software deployment** is required, this will avoid the software bugs which happens quite often for typical code deployment.

Business values

It **reduces the time** cost of launching new policies: as it will be shorter development time, shorter testing and deployment time.

It **decouples** the business logic and technical details, **easier for maintenance** and future system migration, which again saving cost.

How our learners apply the learnt from course:

Challenges

I am a **robotic process automation RPA** developer working for a major **bank**, and I am developing software robots to aid bank staff in automating tedious, but noncomplex tasks. There are few phases in the software development life cycle, and one of the difficulty we face is most likely **finalizing the business requirements**.

Another difficulty would be the business as usual BAU operations after development. Although we told the users to put in a specific file format (**RPA is not intelligent if we are looking at low cost solutions**), they will put in different formats causing the robot to fail. Hence we spend a bulk of time in development **writing exceptions** to prevent these.

What learnt is useful?

Firstly, this course has taught me how to use **KIE tool** (jBPM/Drools). This will definitely aid me in my development in the bank because I use a similar vendor product (Kofax Totalagility [KTA]). Although we use KTA to do basic BPM, we **do not integrate any intelligence** when using it, since most development time is short and they take man-hours budget into heavy consideration. So from this course I learnt to **integrate business process driven by a** (knowledge driven) **rule engine**. Also, by learning different techniques of reasoning, it will be helpful as a tool to help sketch out **multiple methods and models in deriving a smarter robot**. For example, since we deal with **lots of exceptions**, we can use a rule-based reasoning engine (for example, **guided decision table**) to resolve it to different scenarios, which would be **efficient** than having to write a line of code for every decision.

How/Where to apply to workplace?

We currently process company documents in one of our AML (Anti-Money Laundering) robots, which actively seeks for sanction words (example: nuclear) using OCR, which will be used to approve/reject loans by a customer. Right now, we have a process maker (operations staff), and a **checker (management)**. If a rule-based engine is well designed, it can even replace the checker. But a full replacement will not be recommended, and the checker should do double check. But this will **speed up document processing** in a day, improving efficiency.

Business values

Higher efficiency can be reached. In RPA we count profits by total hours saved. By introducing **more reliable and intelligent agents** in our robots, besides from human action assistance, we can replace human thinking cognition, thereby saving even more man hours.

How our learners apply the learnt from course:

Challenges

I am **test engineer** in **semiconductor** company that produce memory chips. I face difficulty in **identifying the failure mode of the memory chip** that I am testing. I will need to **forecast the yield for each product** and each product is having different attribute or properties. I am having hard time to process all the yield data and finally produce the yield forecast. I also need to generate test time forecast at the same time. As a result, I will need better forecast model.

What learnt is useful?

I can use Python **Orange tool** for **data mining**. I can key in the testing result in **excel form** then use the decision tree to find out which attribute/property of the testing parts **affects the testing results the most**.

KIE workbench tools can help me in creating **GUI**. Enable **multiple users** to key in their input data into the platform. They can also retrieve output that there are interested. They can also send their question using the KIE forms.

How/Where to apply to workplace?

I will use python orange for data mining. Finding out the **best decision tree then use it to create rules** that can **forecast the yield** of each product.

Then create KIE workbench application for **yield prediction**. **Multiple user** can key in the product yield value with properties they have. Then, the rule generated using data mining method can be applied for forecast.

Business values

I can help my company to **enhance** its yield forecast system and have a better production volume forecast. This will be chance to use system intelligence for cost saving.

I can also prepare myself in data mining and implementing rule based system in my working environment. Enhance my own skill set.

How our learners apply the learnt from course:

Challenges

I work in a **Systems Integration** company. Here we have multiple large-scale projects which require customization, development and implementation. It is difficult to manage scheduling and assignment of development effort to projects (due to skillsets, availability, etc.).

Also, as the company has been around for quite a while, much of the processes are still **manual and archaic**. As people come and go, the initial processes are no longer adhered to and there is **not much visibility** on them.

What learnt is useful?

KIE tool can encode the various **decision factors** for certain processes like (leave, scheduling of staff to projects). It also allows for **multiple users** to interact and utilize the system at once.

For example, in my company there are many projects and developers. KIE can help to automate the assignment, monitoring and scheduling of developers to projects.

How/Where to apply to workplace?

First, it makes sense to **map out the current given process** within my company. For example, I've recently done a Business Process Re-engineering to improve the Recruitment Process in my company. Using KIE, I can map out the process and include the business rules/ logic that is used to decide.

Secondly, **include the business logic** into the Recruitment Process. For example, the salary approval for certain staff rank must be routed and go through the Head of HR or to the Head of the Business Unit. This can be mapped within KIE and sent to them for approval.

Lastly, on-board users to utilize this automated process. KIE can be used as the system in which to **implement the automated process**.

Business values

One business value that can be derived is to **decrease** the required manual effort for writing out all the **physical forms**. Another is to **visualize** and provide awareness of the current **process**. A third is to ensure that the process is **fair** and not biased.

How our learners apply the learnt from course:

Challenges

I worked in a **charity** called Singapore Children's Society. My work involves **conducting research** on issues concerning the well-being of children, youth and family e.g., child abuse and bullying. One of the main work challenges in my field of work is **translating scientific findings into practices that social workers can then use to help their clients e.g., children and youths.**

What learnt is useful?

The most helpful thing that I had learnt from this course is **how to take the acquired knowledge that I had gained from my own research and from other's research findings and convert them into rules.** For example, I have converted these research findings into decision trees knowledge model and rules that I then give to social workers **to assist their decision-making**.

The **KIE tool** can support better to the social workers because it can be automated thus I do not need to sit down with social workers every time there is a change in the model. KIE can be scaled up to **support multiple users**, which help us a lot because **we are a small team with little manpower**. The other thing is that KIE can **generate reports easily**, which makes it easy to report to management on **how things are progressing** without having to spend time doing up reports. This leave us with more time to take on more higher value work.

How/Where to apply to workplace?

I used KIE to generate a small project based on the decision tree that ask the social workers to fill in the inputs in order to generate a set of recommendation on how they can work with their clients. For example, my research investigated whether certain demographic profile of children was more at risk of child abuse. I looked at variables such as race, age, gender, parental education level and examine the association with child abuse victimization. **I then use these findings to build a decision tree to derive simple rules to fit into the KIE project/system**.

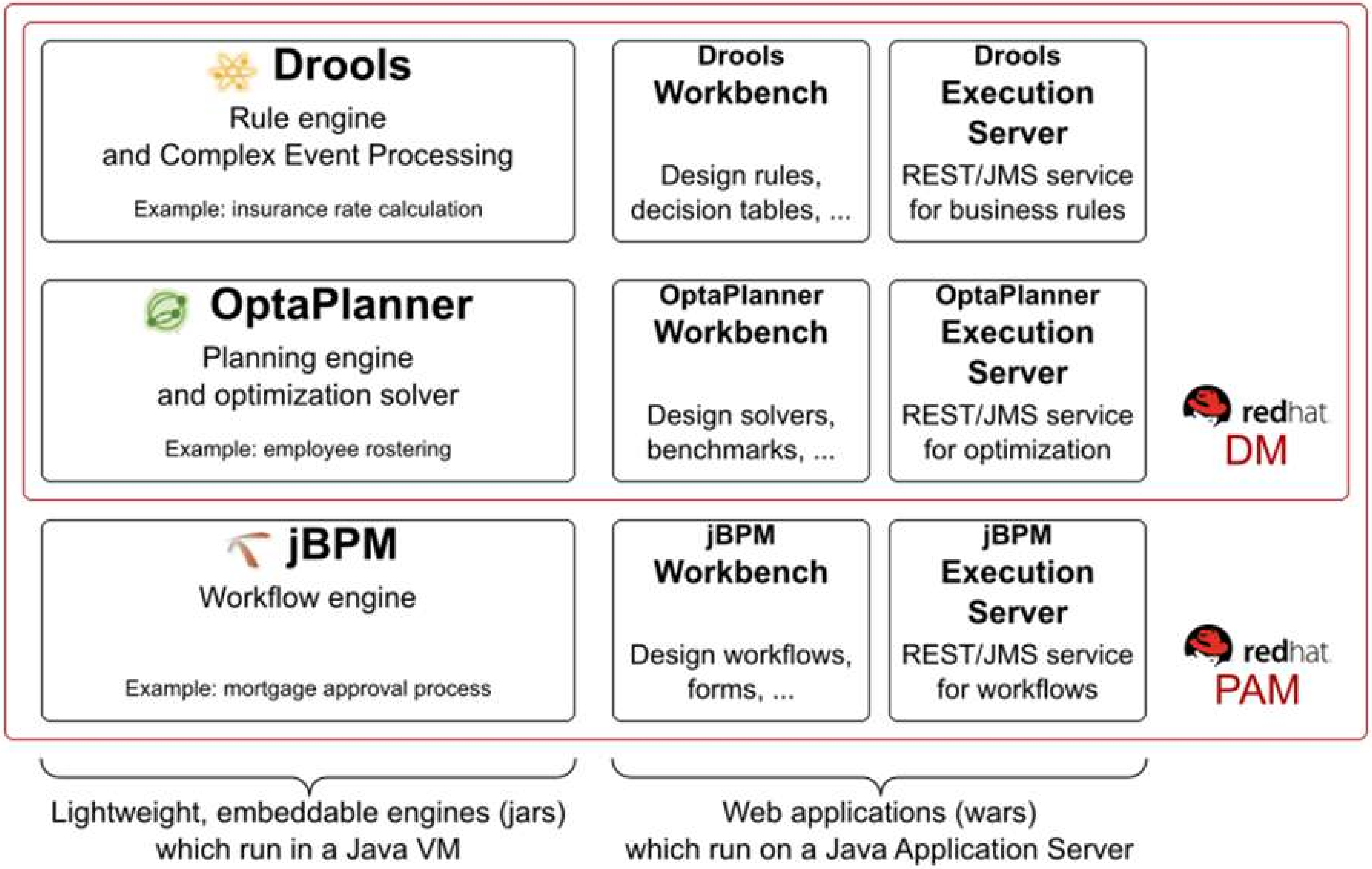
Business values

Because of the course, I had learnt to implement a **automated machine reasoning systems** that did part of my work for me. The business values derived from this is that my knowledge is now readily available to social workers, **knowledge can be updated easily**, social workers is now better able to use the knowledge because it has been translated into easily applied rules, and I can focus more on **higher values work** such as doing more research.

KIE System Architecture

KIE functionality overview

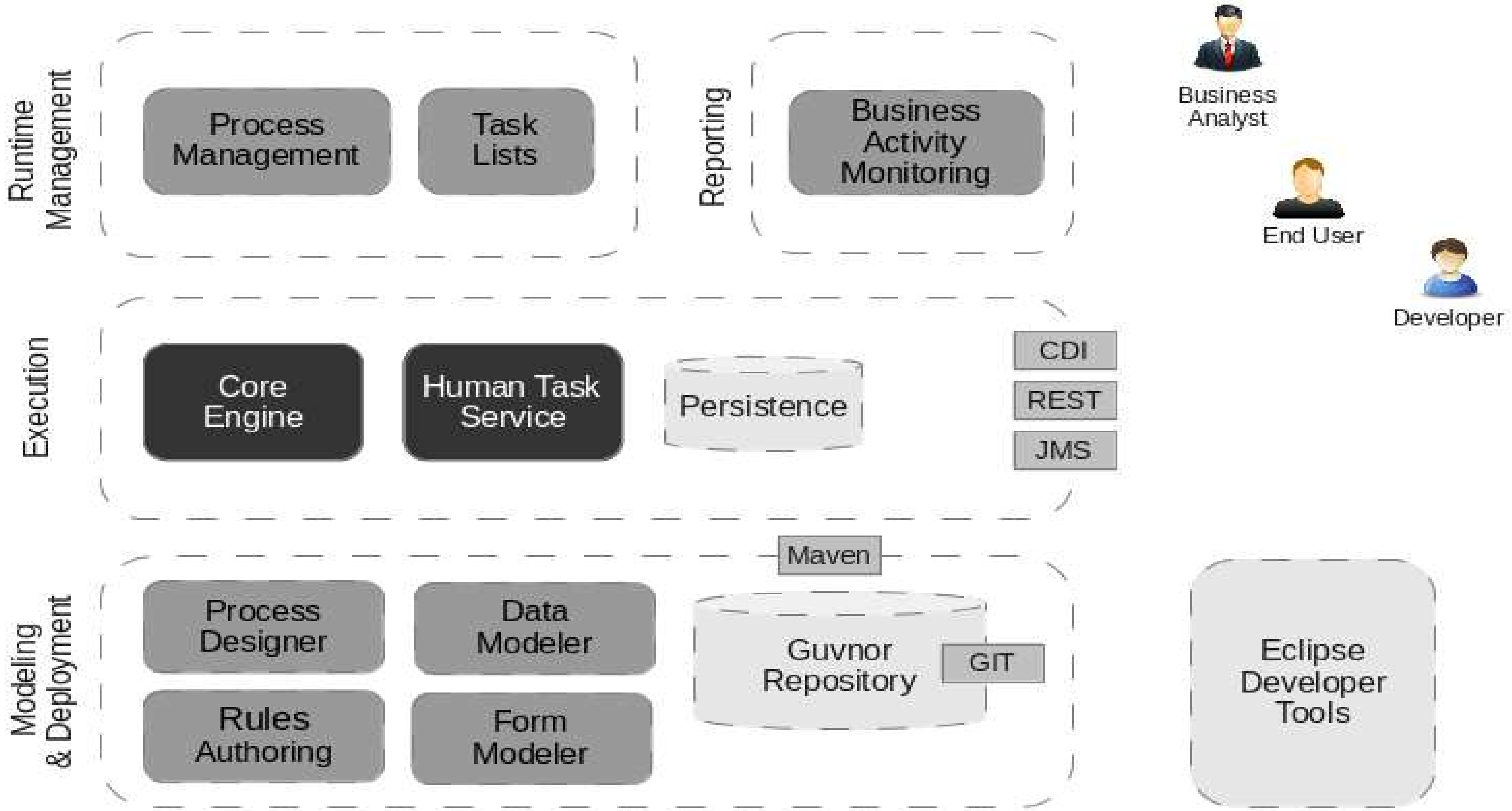
What are the KIE projects?

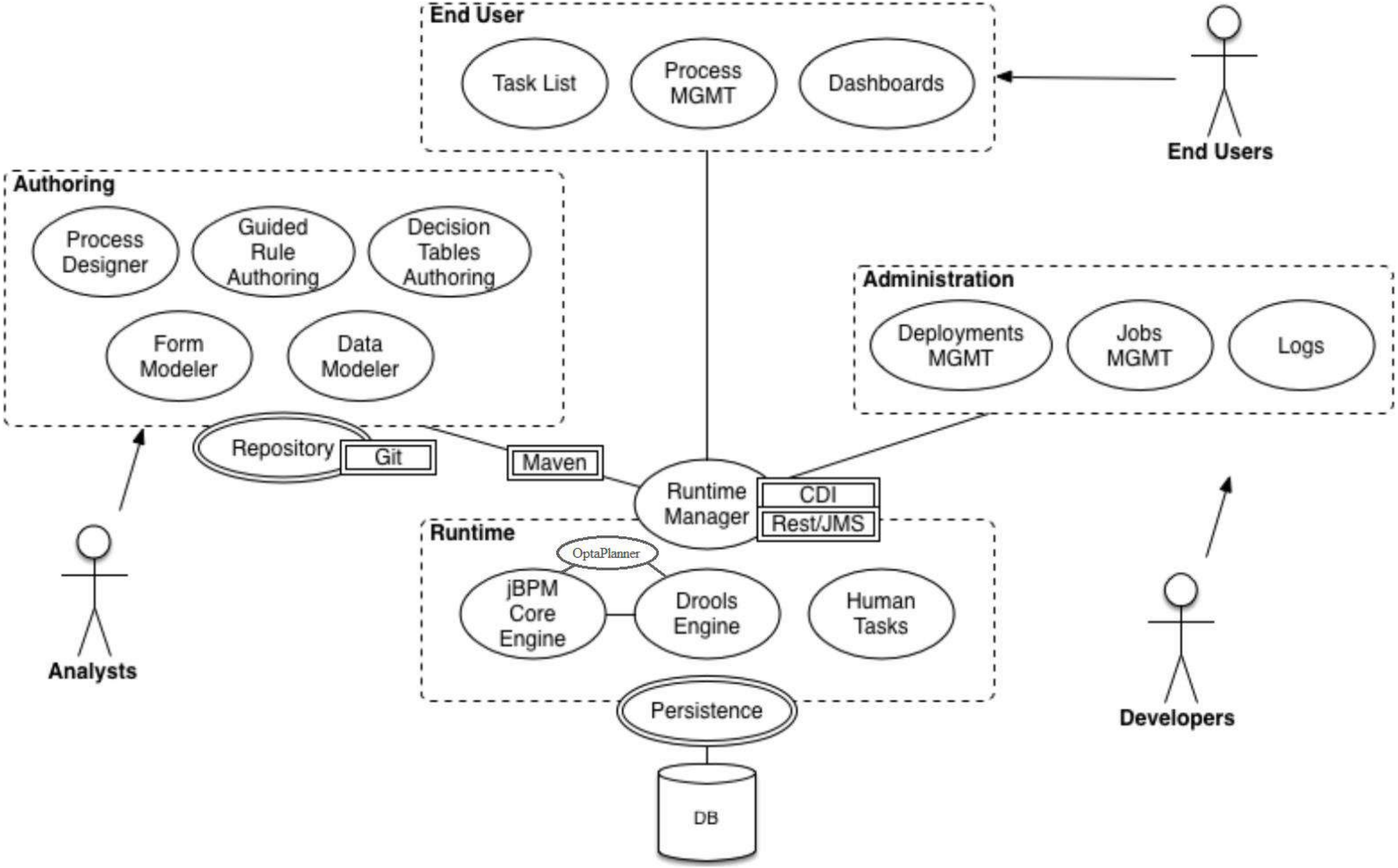


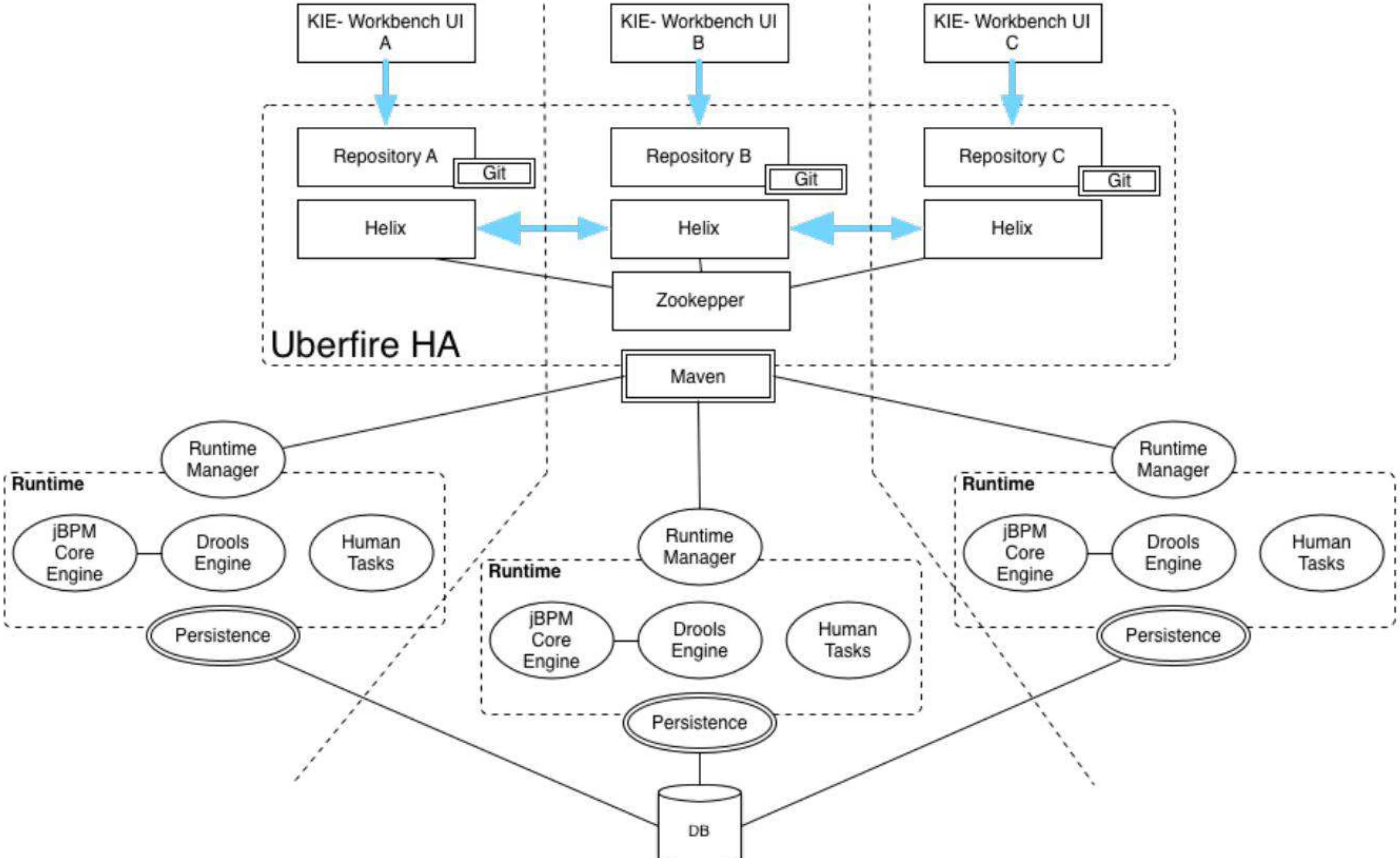
Drools Rule engine and Complex Event Processing. Example: insurance rate calculation	**Drools Workbench** Design rules, decision tables, ...	**Drools Execution Server** REST/JMS service for business rules	
OptaPlanner Planning engine and optimization solver. Example: employee rostering	**OptaPlanner Workbench** Design solvers, benchmarks, ...	**OptaPlanner Execution Server** REST/JMS service for optimization	redhat DM
jBPM Workflow engine. Example: mortgage approval process	**jBPM Workbench** Design workflows, forms, ...	**jBPM Execution Server** REST/JMS service for workflows	redhat PAM

Lightweight, embeddable engines (jars) which run in a Java VM

Web applications (wars) which run on a Java Application Server


Runtime Management
Process Management
Task Lists
Reporting
Business Activity Monitoring
Business Analyst
End User
Developer
Execution
Core Engine
Human Task Service
Persistence
CDI
REST
JMS
Maven
Modeling & Deployment
Process Designer
Data Modeler
Rules Authoring
Form Modeler
Guvnor Repository
GIT
Eclipse Developer Tools


End User
Task List
Process MGMT
Dashboards
End Users
Authoring
Process Designer
Guided Rule Authoring
Decision Tables Authoring
Form Modeler
Data Modeler
Repository
Git
Maven
Administration
Deployments MGMT
Jobs MGMT
Logs
Runtime Manager
CDI
Rest/JMS
Runtime
OptaPlanner
jBPM Core Engine
Drools Engine
Human Tasks
Persistence
DB
Analysts
Developers


KIE- Workbench UI A
KIE- Workbench UI B
KIE- Workbench UI C
Repository A
Git
Repository B
Git
Repository C
Git
Helix
Helix
Helix
Zookepper
Uberfire HA
Maven
Runtime Manager
Runtime Manager
Runtime Manager
Runtime
jBPM Core Engine
Drools Engine
Human Tasks
Persistence
Runtime
jBPM Core Engine
Drools Engine
Human Tasks
Persistence
Runtime
jBPM Core Engine
Drools Engine
Human Tasks
Persistence
DB

END OF APPENDICES