ModernStats World Workshop 2019

Documenting Health and Population Research in DDI 4

Monday, June 17, 2019

Jay Greenfield
Chifundo Kanjala
Arofan Gregory

Introduction

- The purpose of this presentation is to revisit and question some of the explicit and implicit assumptions that drove the development of the early DDI 4 process model with its two parts – Workflows and BusinessWorkflow
- The goal of this review is to consider alternative scopes for the process model based on some issues that have come to light with the prototype and some ”field work” that has been undertaken
- In this field work an augmented DDI 4 process model is used to first describe data management on an ETL platform called Pentaho and then the process model is used to inform the development of an interactive provenance browser

The Principal Components

- In line with the GSBPM (Generic Statistical Business Process Model) and GSIM (Generic Statistical Information Model), the DDI 4 process model has three principal components:
  - Business tasks
  - Process steps
  - Services
- ***Business tasks*** are domain specific
  - A group of business tasks forms a pipeline through which we might construct a statistic, produce an analysis dataset and/or construct and submit a DNA sequence to the GenBank
  - Business Process Models like the GSBPM and the Generic Longitudinal Business Process Model (GLBPM) describe these data pipeline components

The Principal Components (continued)

- Each business task in a data pipeline is achieved using a collection of ***process steps***
- These collections follow a ***design pattern*** that may be further specified with an ***algorithm***
- Each process step in turn may be implemented by one or more ***services*** that are either vendor and platform specific or not
  - A SAS PROC, an SPSS or Stata command, an R statement or an ETL transformation are examples of platform specific services
  - Web services and microservices are examples of vendor and platform independent services
  - Note that services break down workflow steps. Services are ***granular***. Services are ***atomic***. They are the antidote to massive applications.

03_Core_ETL_Raw_6.1_Dataset_Quality_Metrics Step

Overview

Calculate Data Quality Metrics

Business Task or Subtask

ProcessStep Collection

quality of the data in the raw specification created in business process, 02 Core ETL for Raw 6.1, on the basis of a set

Each step is implemented by one or more atomic services

Algorithm

[step: 3.1] Compile a list of Quality Metrics: Compile a list of quality metrics relevant to the data specification
This algorithm step references the following study concepts:

[step: 3.2] Create events consistency matrix: Create events consistency matrix showing the logical ordering of event sequences
This algorithm step references the following study concepts:

[step: 3.3] Compile residency starting events: Identify in the data, events that start a residency episode (birth, external-immigration, enumeration, becoming eligible for a study, found after being lost to follow-up)
This algorithm step references the following study concepts: residency birth migration

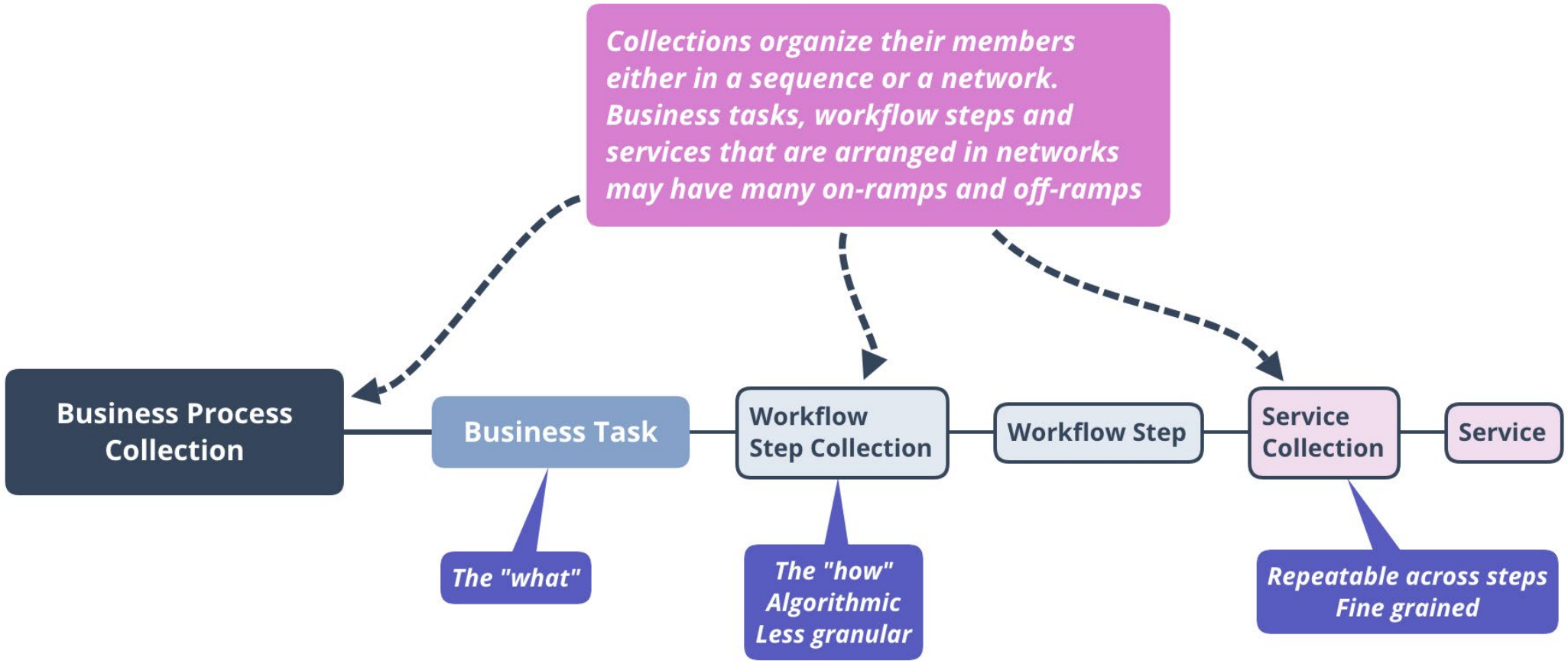
[step: 3.4] Compile residency ending events: Identify in the data, events that end a residency episode (external-outmigration, death, became ineligible for study, lost to follow-up, internal-outmigration, present in the study (right censored))
This algorithm step references the following study concepts: residency death migration

[step: 3.5] Compile legal and illegal start events: Review the identified start events and distinguish between legal and illegal ones
This algorithm step references the following study concepts:

[step: 3.6] Compile legal and illegal end events: Review the identified end events and distinguish between legal and illegal ones
This algorithm step references the following study concepts:

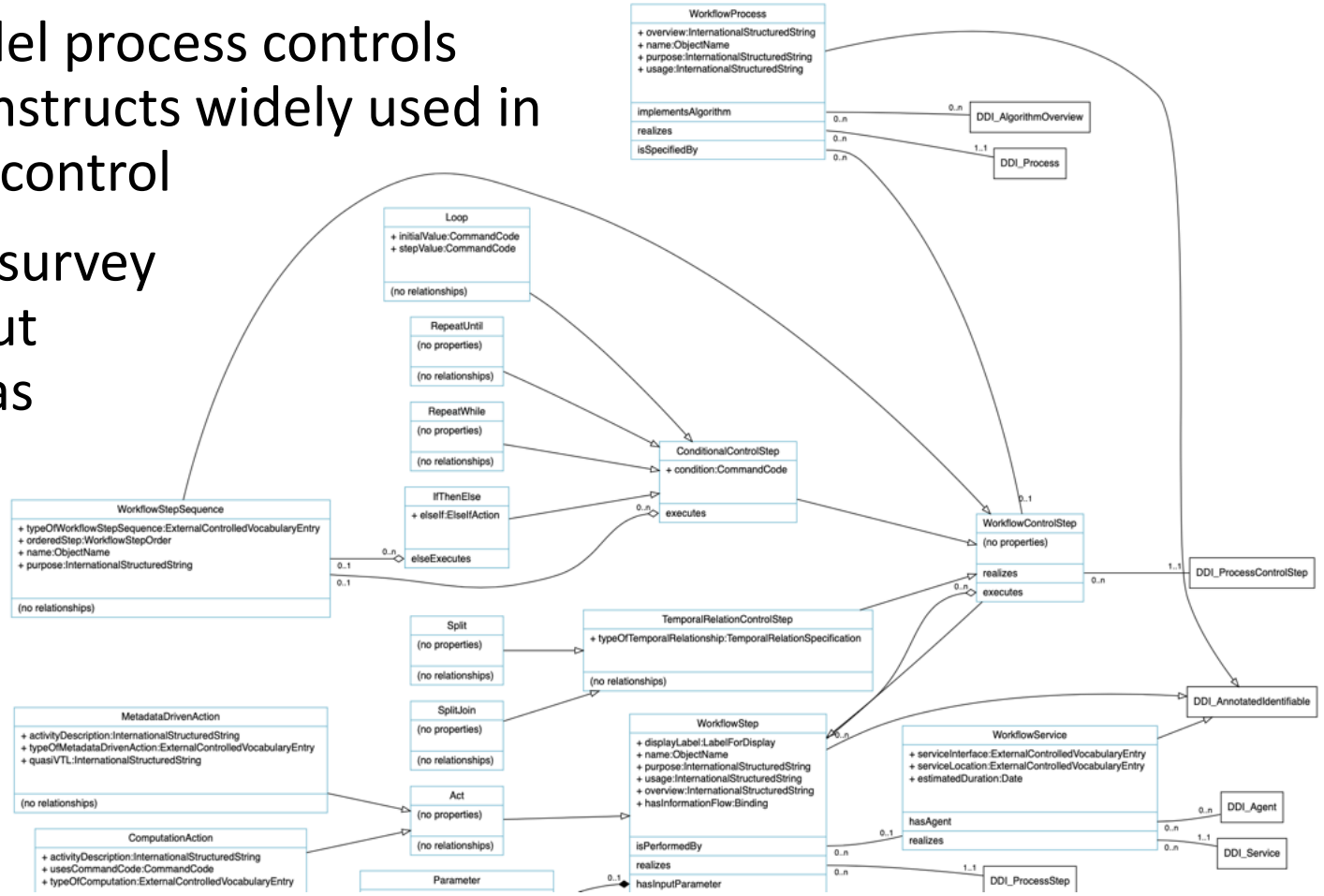
[step: 3.7] Compile legal and illegal transitions: Review all transitions between two events and distinguish between legal and illegal ones
This algorithm step references the following study concepts:

Towards a UML Model…

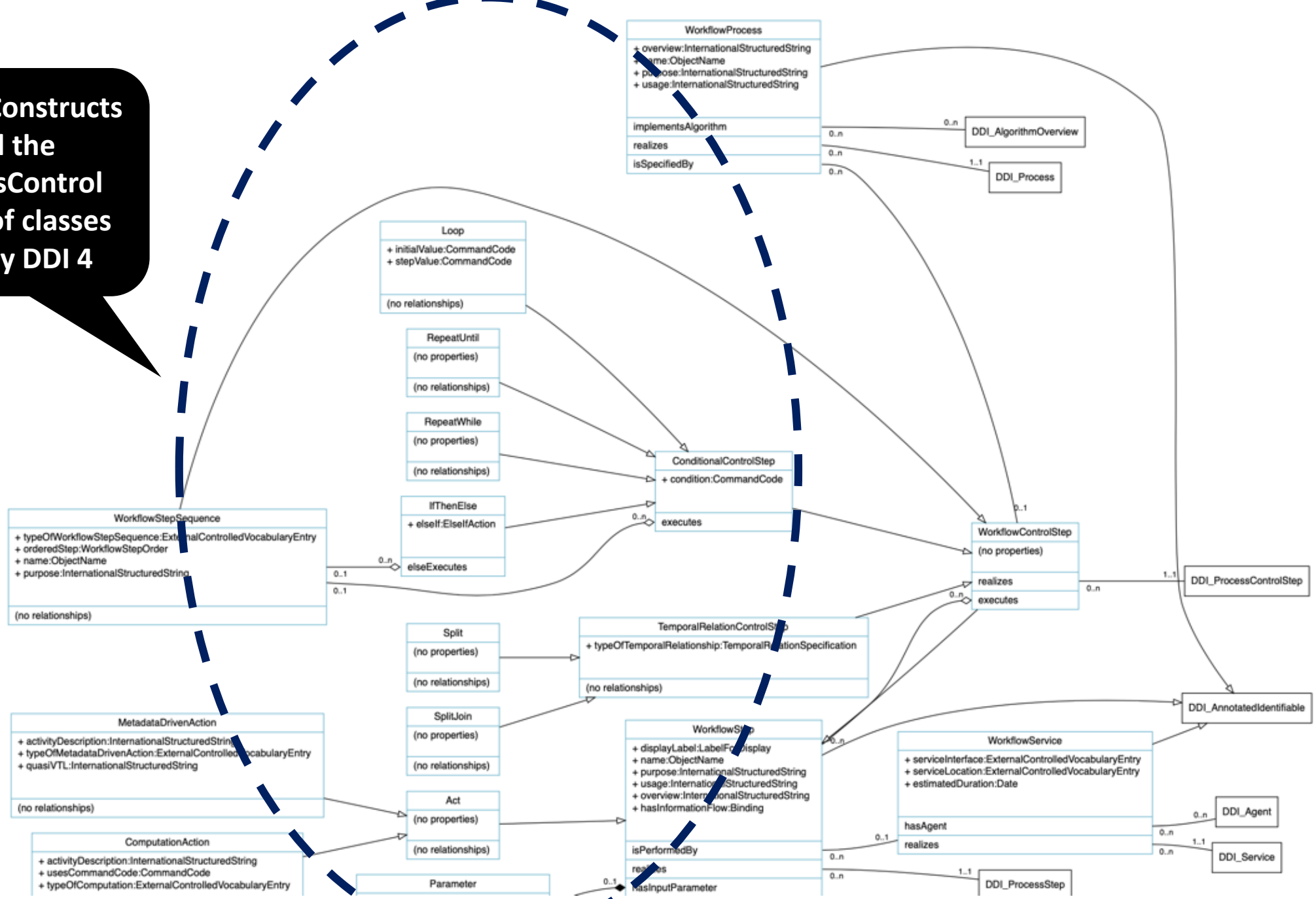


Towards a UML Model (continued)

- In the early DDI 4 process model process controls consisted mainly of control constructs widely used in support of questionnaire flow control
- The weight the model gave to survey research raised questions about whether data management was being given its due
- With this in mind we began to entertain other forms of process control in support of data management



ControlConstructs and the ProcessControl family of classes in early DDI 4



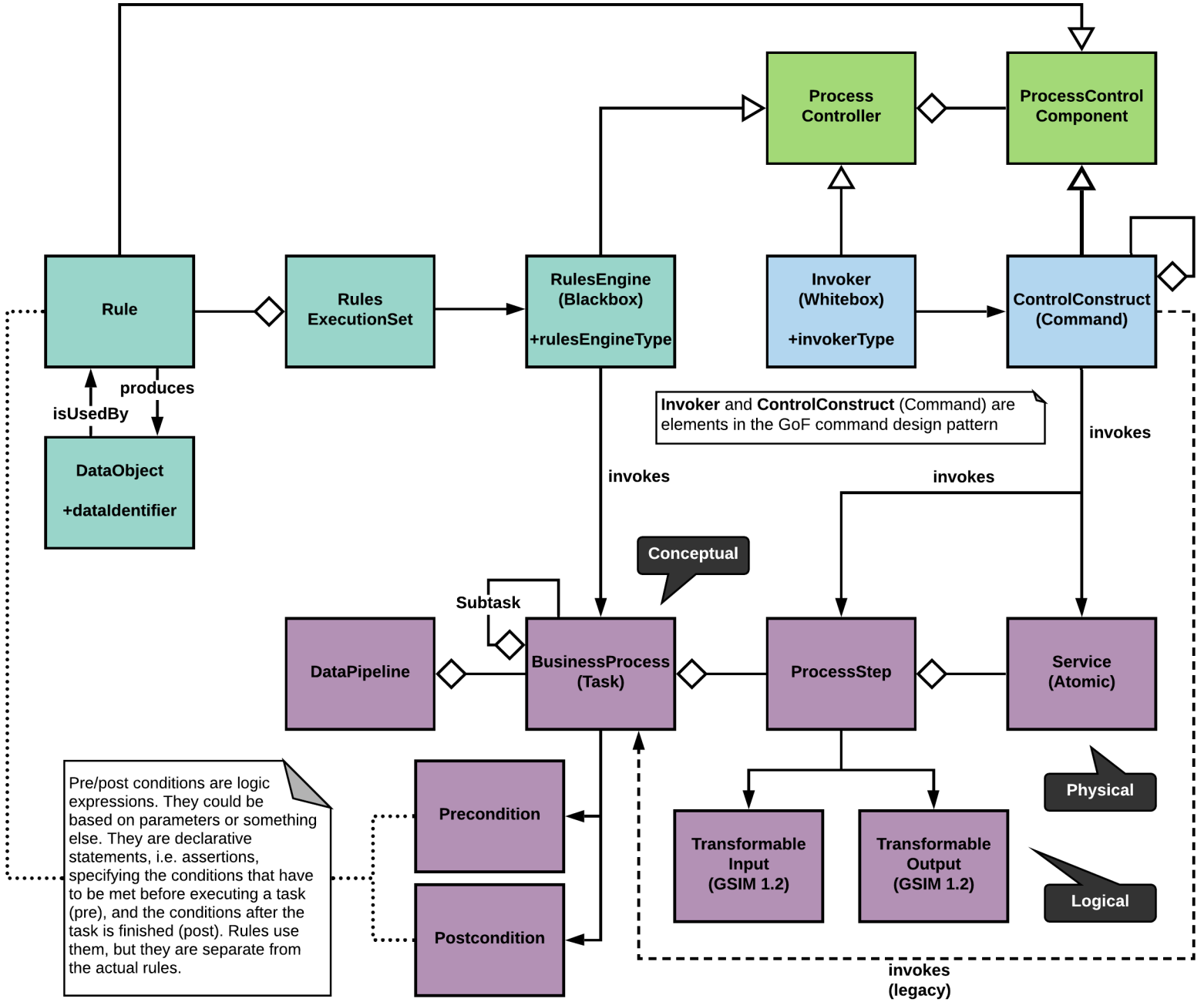
Process Control Requirements for Survey Research and Data Management

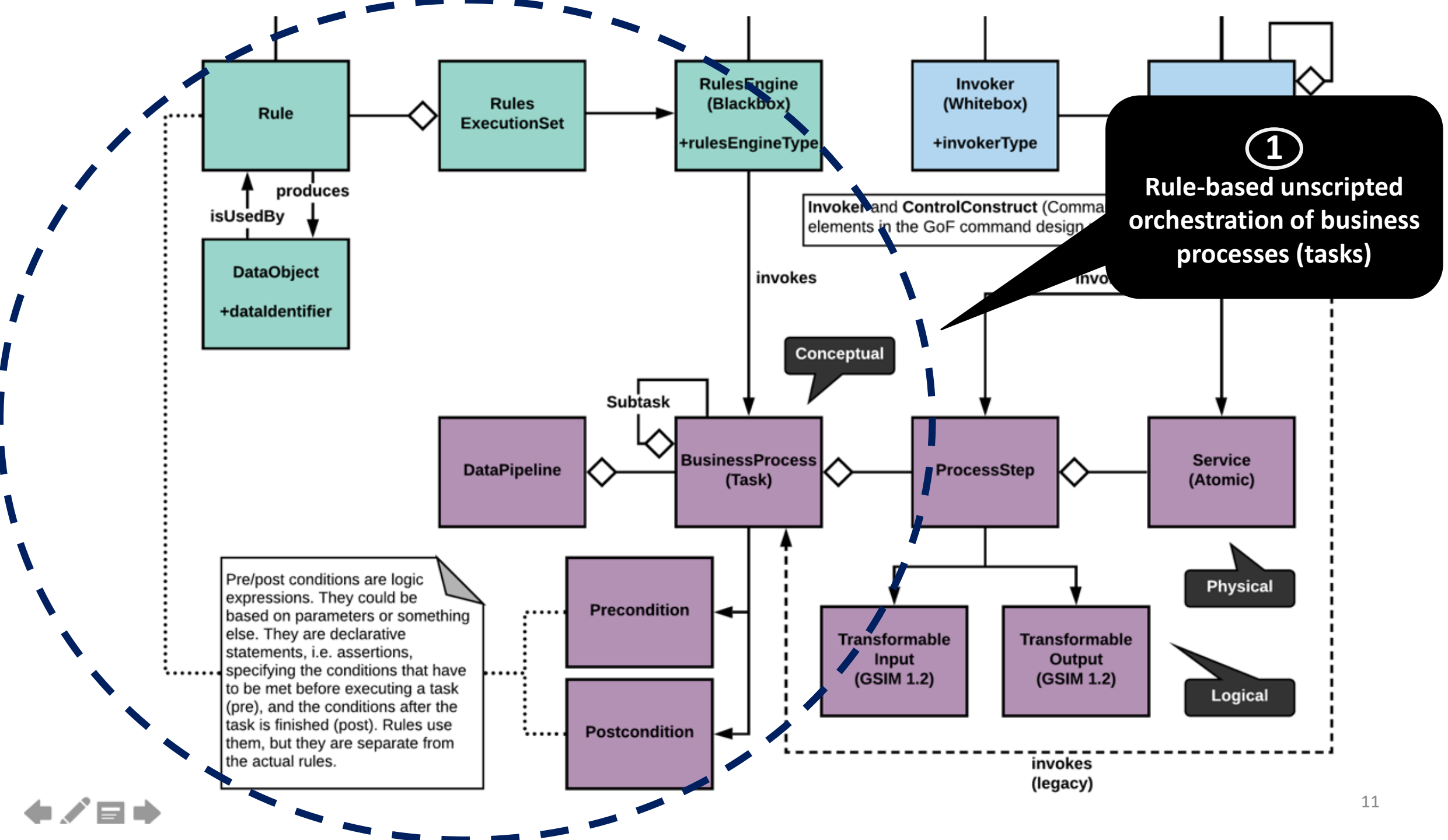
Characteristic	Survey Research	Data Management
Flow Unit	Question, Question Block, Preload, Questionnaire, Procedure, Variable	Dataset, Database, Files, Sensor-driven data streams, Key/Value Stores, Variables and Variable Groups
Flow Patterns	Sequential with Loops and Skips	Directed Acyclic Graph Multiple On-ramps and Off-ramps
Flow Logic	Control Constructs (If Then Else, Loop, Repeat Until, etc.)	Preconditions and Postconditions (consistent with PROV-O where "*Activities* may be started or ended by *Entities*") Data Parallelism Task Parallelism

- By and large data management platforms and survey research platforms have ***distinct*** flow characteristics

A Change in the Requirements

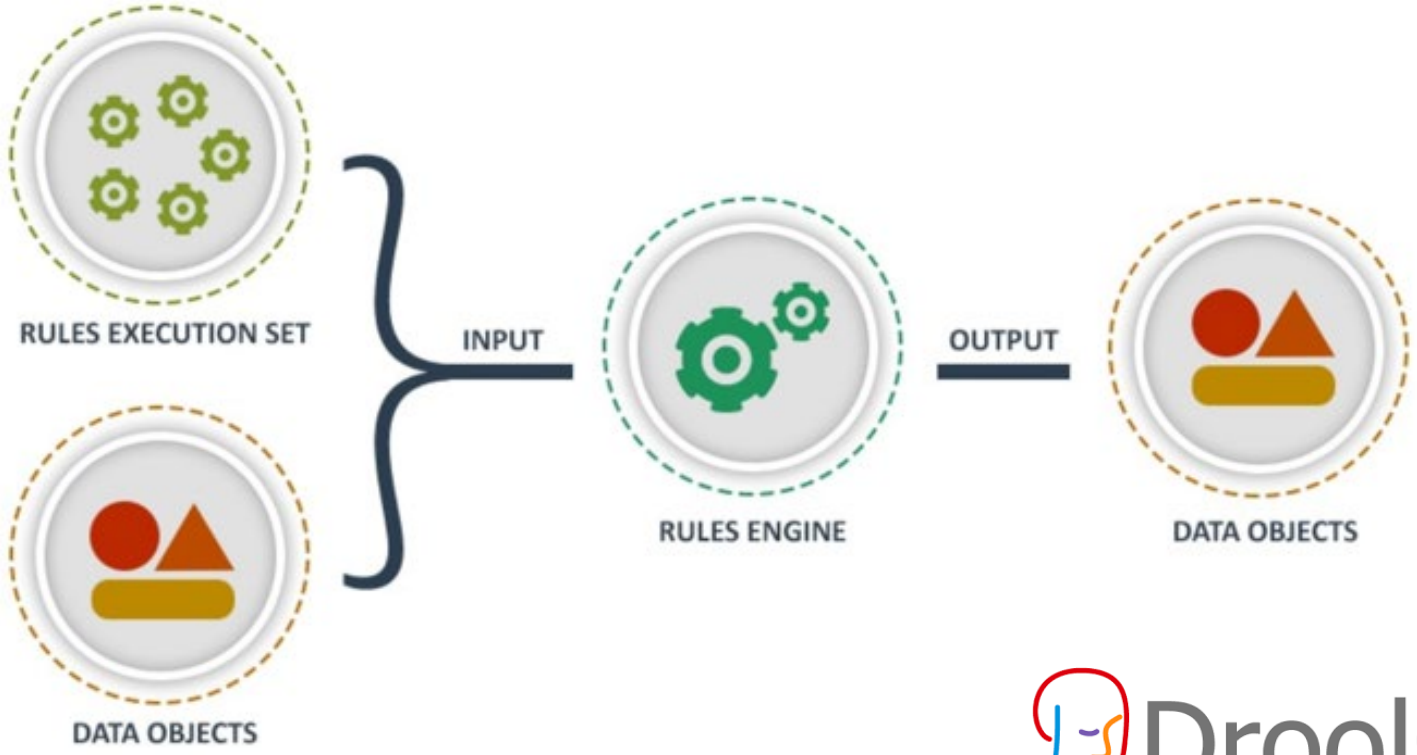
- Depending on the platform, questionnaire control constructs may not be needed to support data management
- And a new cast of information objects besides question and variable units are needed including datasets, databases, files, key/value stores, data cubes and sensor-driven data streams
- These new data objects don’t figure into DDI 3 and early DDI 4 inputs and outputs
- Those inputs and outputs take only a single value or an array


Process Controller
ProcessControl Component
Rule
Rules ExecutionSet
RulesEngine (Blackbox)
+rulesEngineType
Invoker (Whitebox)
+invokerType
ControlConstruct (Command)
produces
isUsedBy
DataObject
+dataIdentifier
Invoker and ControlConstruct (Command) are elements in the GoF command design pattern
invokes
invokes
invokes
Conceptual
Subtask
DataPipeline
BusinessProcess (Task)
ProcessStep
Service (Atomic)
Physical
Pre/post conditions are logic expressions. They could be based on parameters or something else. They are declarative statements, i.e. assertions, specifying the conditions that have to be met before executing a task (pre), and the conditions after the task is finished (post). Rules use them, but they are separate from the actual rules.
Precondition
Postcondition
Transformable Input (GSIM 1.2)
Transformable Output (GSIM 1.2)
Logical
invokes (legacy)

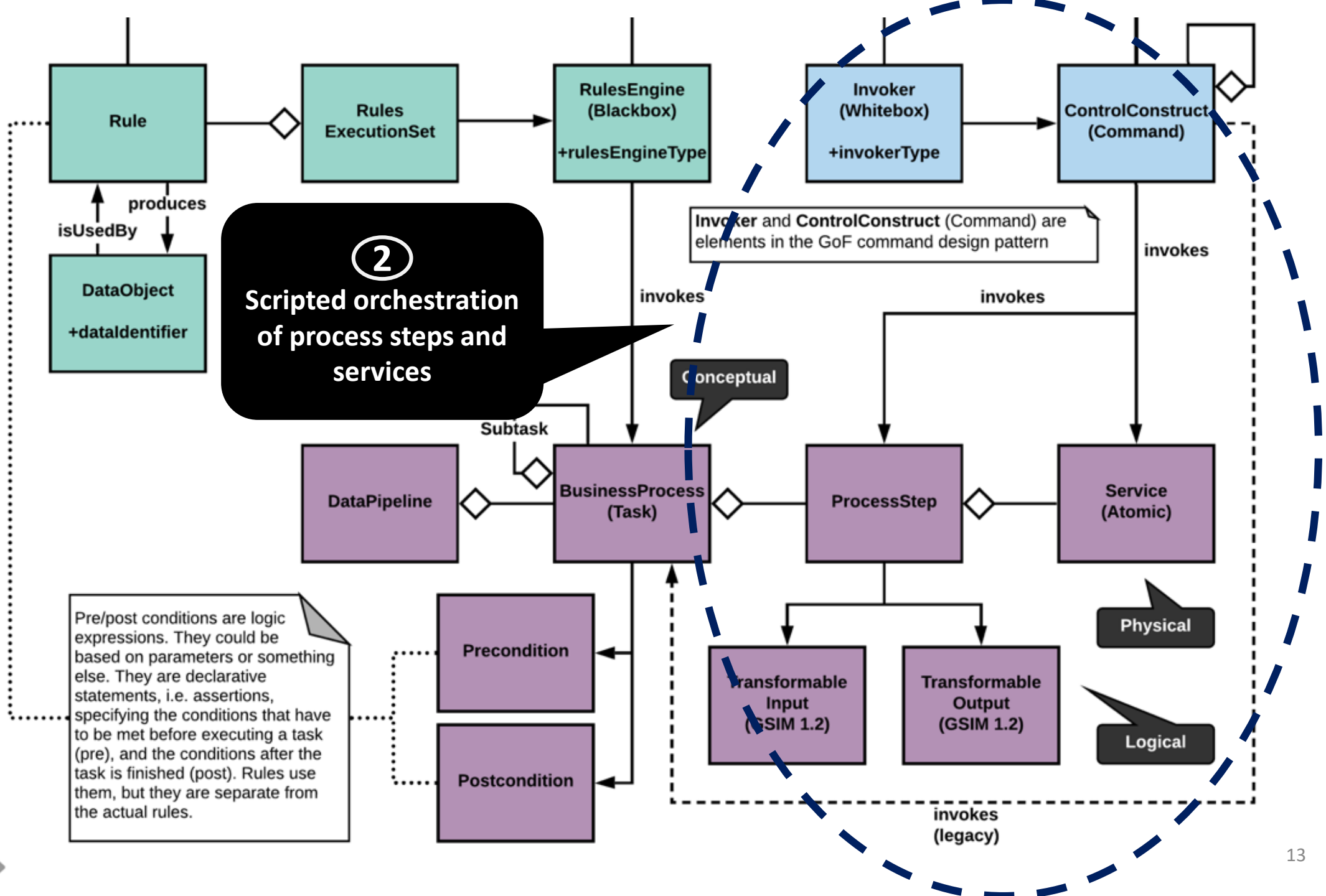

Rule
Rules ExecutionSet
RulesEngine (Blackbox)
+rulesEngineType
Invoker (Whitebox)
+invokerType
produces
isUsedBy
DataObject
+dataIdentifier
invokes
Conceptual
Subtask
DataPipeline
BusinessProcess (Task)
ProcessStep
Service (Atomic)
Physical
Logical
Precondition
Postcondition
Transformable Input (GSIM 1.2)
Transformable Output (GSIM 1.2)
invokes (legacy)
Pre/post conditions are logic expressions. They could be based on parameters or something else. They are declarative statements, i.e. assertions, specifying the conditions that have to be met before executing a task (pre), and the conditions after the task is finished (post). Rules use them, but they are separate from the actual rules.
1
Rule-based unscripted orchestration of business processes (tasks)

Drools: A Business Rules Management System

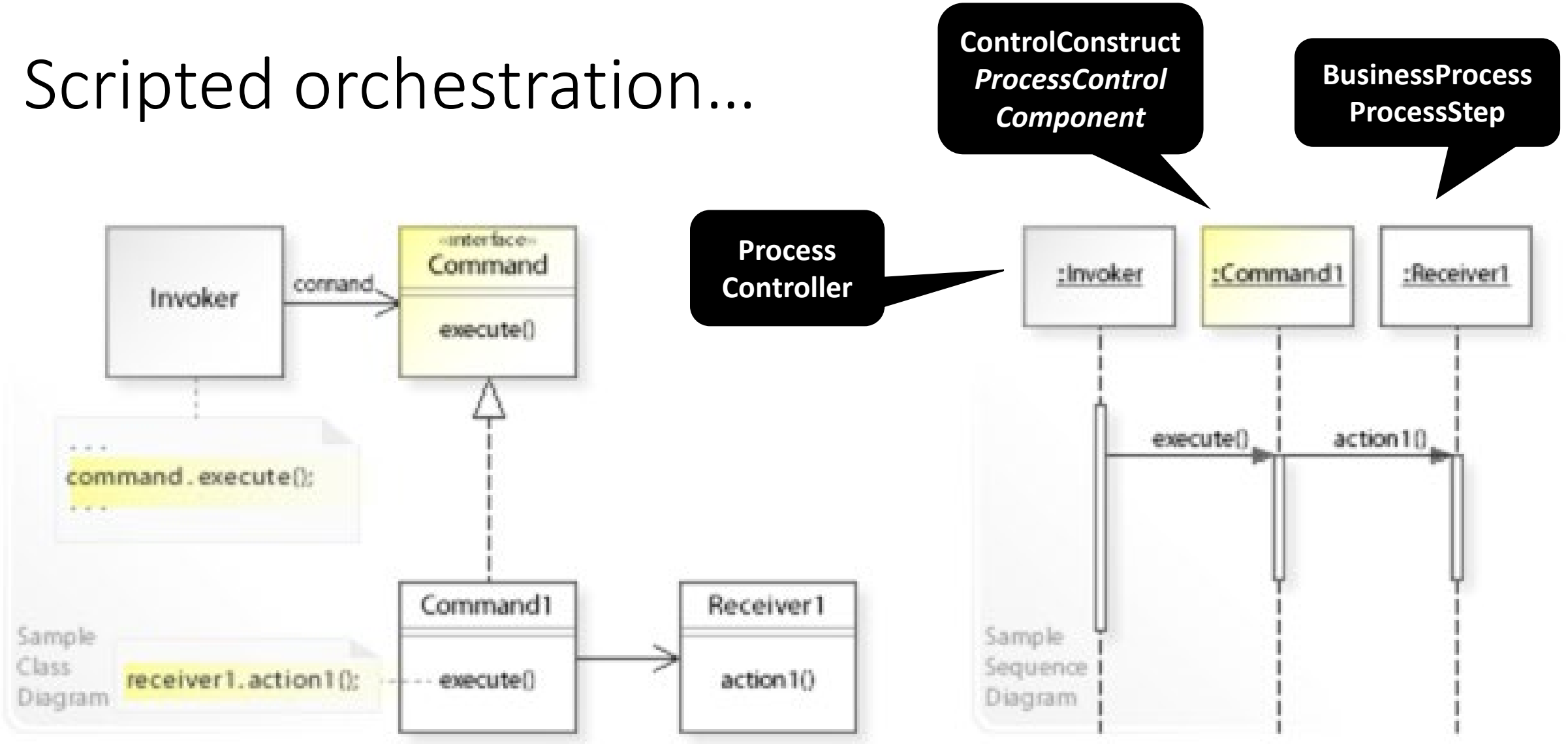
Most programmers are familiar with if / then statements. A rule engine can be viewed as a sophisticated interpreter for if / then statements, where the statements themselves are known as rules. So, the 'if' portions of rules contain conditions and the 'then' portions contain actions. The inputs to a rule engine are a rule execution set and some data objects. The outputs from a rule engine may include the original input data objects with possible modifications, new data objects, and side effects.

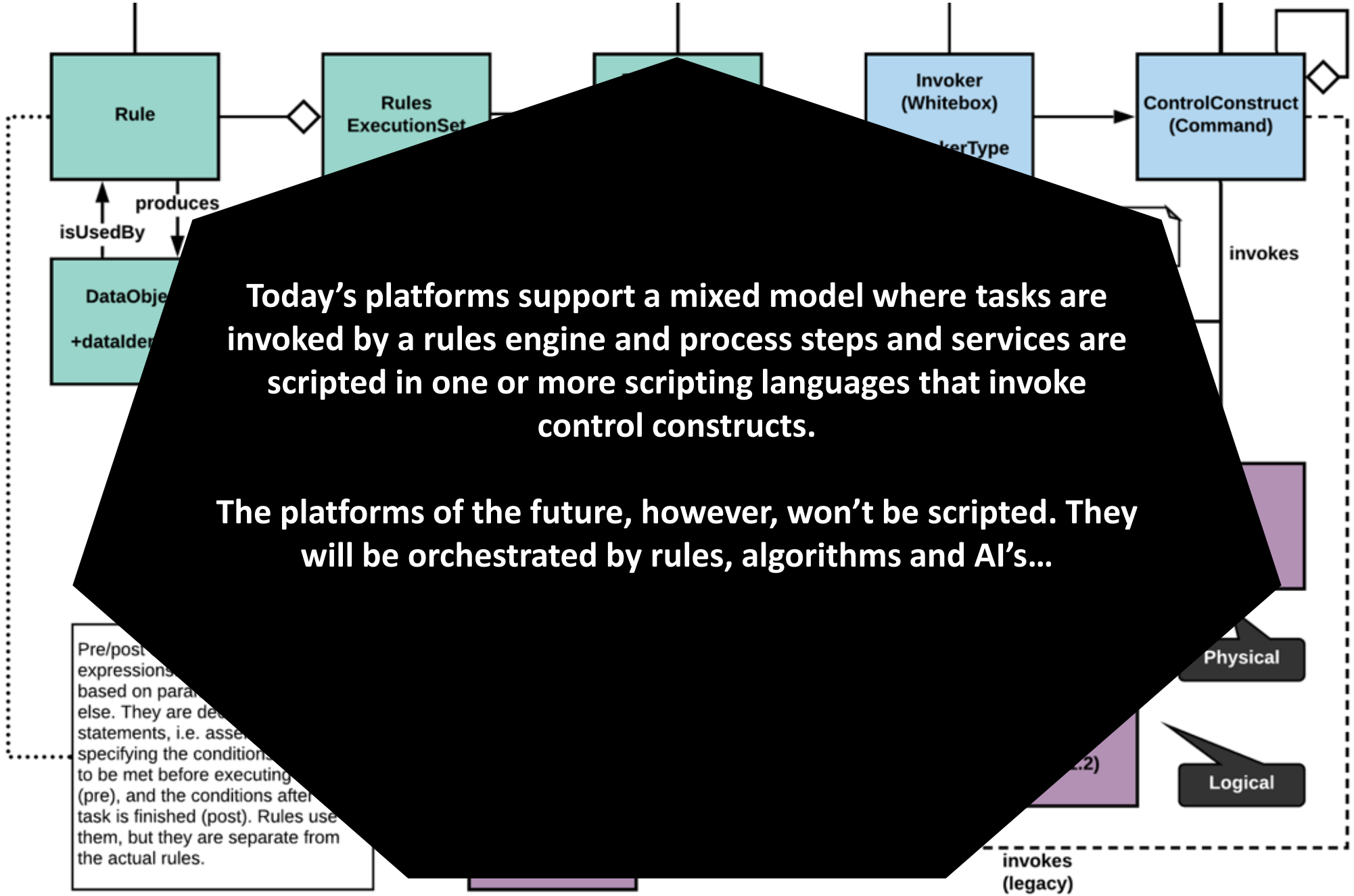


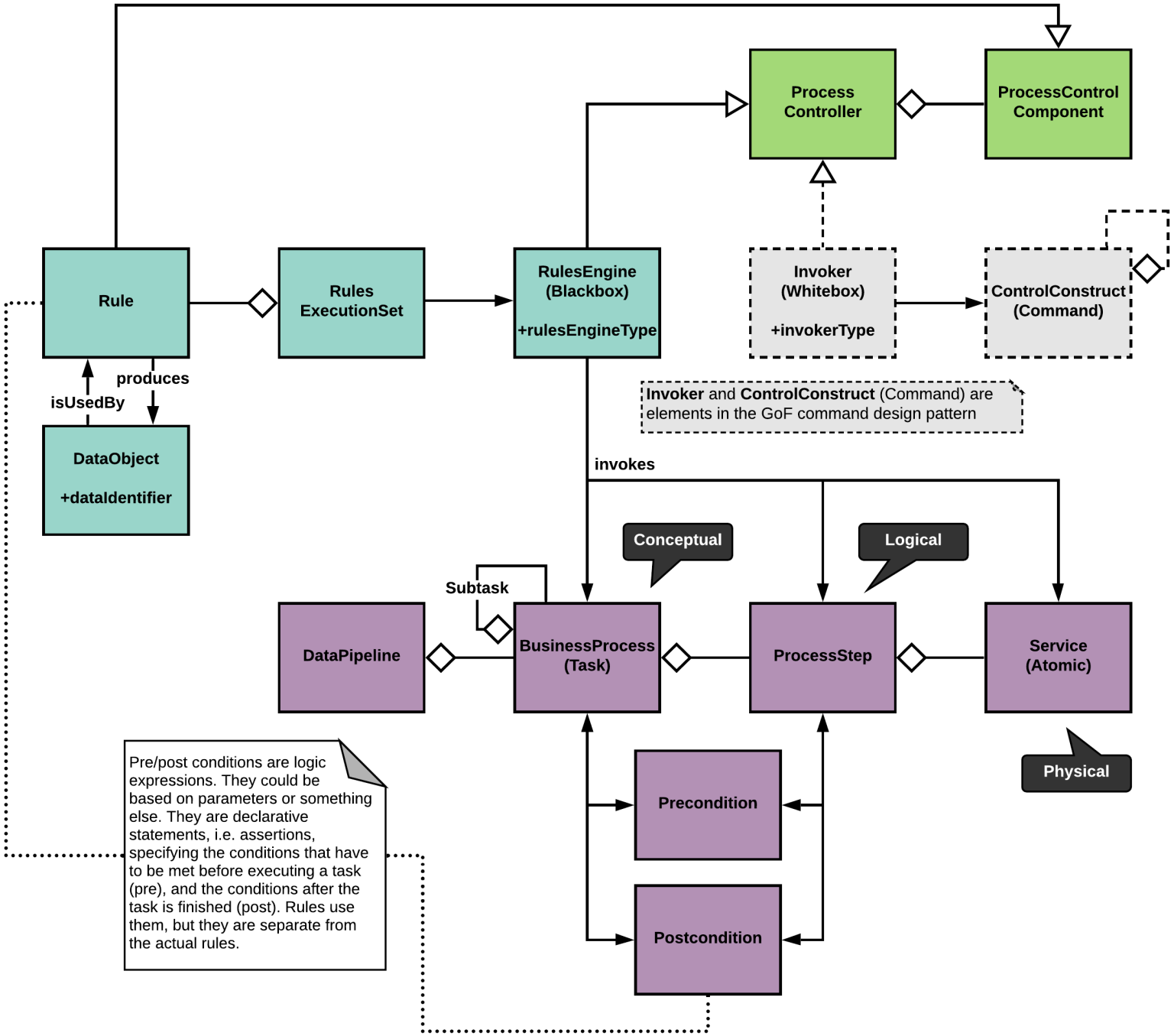
Drools

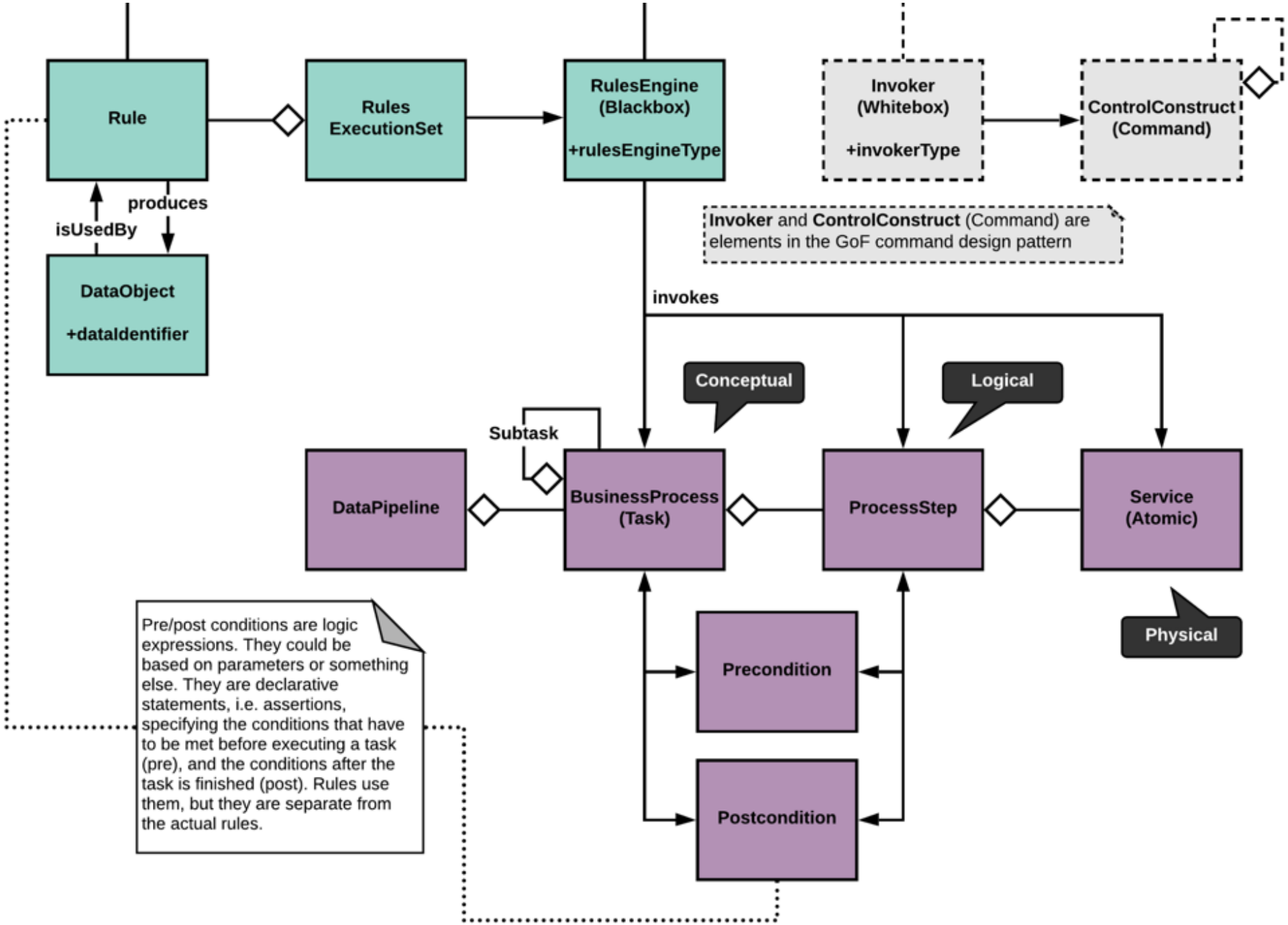

Rule
Rules ExecutionSet
RulesEngine (Blackbox)
+rulesEngineType
Invoker (Whitebox)
+invokerType
ControlConstruct (Command)
produces
isUsedBy
DataObject
+dataIdentifier
2
Scripted orchestration of process steps and services
Invoker and ControlConstruct (Command) are elements in the GoF command design pattern
invokes
invokes
invokes
Conceptual
Subtask
DataPipeline
BusinessProcess (Task)
ProcessStep
Service (Atomic)
Physical
Logical
Pre/post conditions are logic expressions. They could be based on parameters or something else. They are declarative statements, i.e. assertions, specifying the conditions that have to be met before executing a task (pre), and the conditions after the task is finished (post). Rules use them, but they are separate from the actual rules.
Precondition
Postcondition
Transformable Input (GSIM 1.2)
Transformable Output (GSIM 1.2)
invokes (legacy)

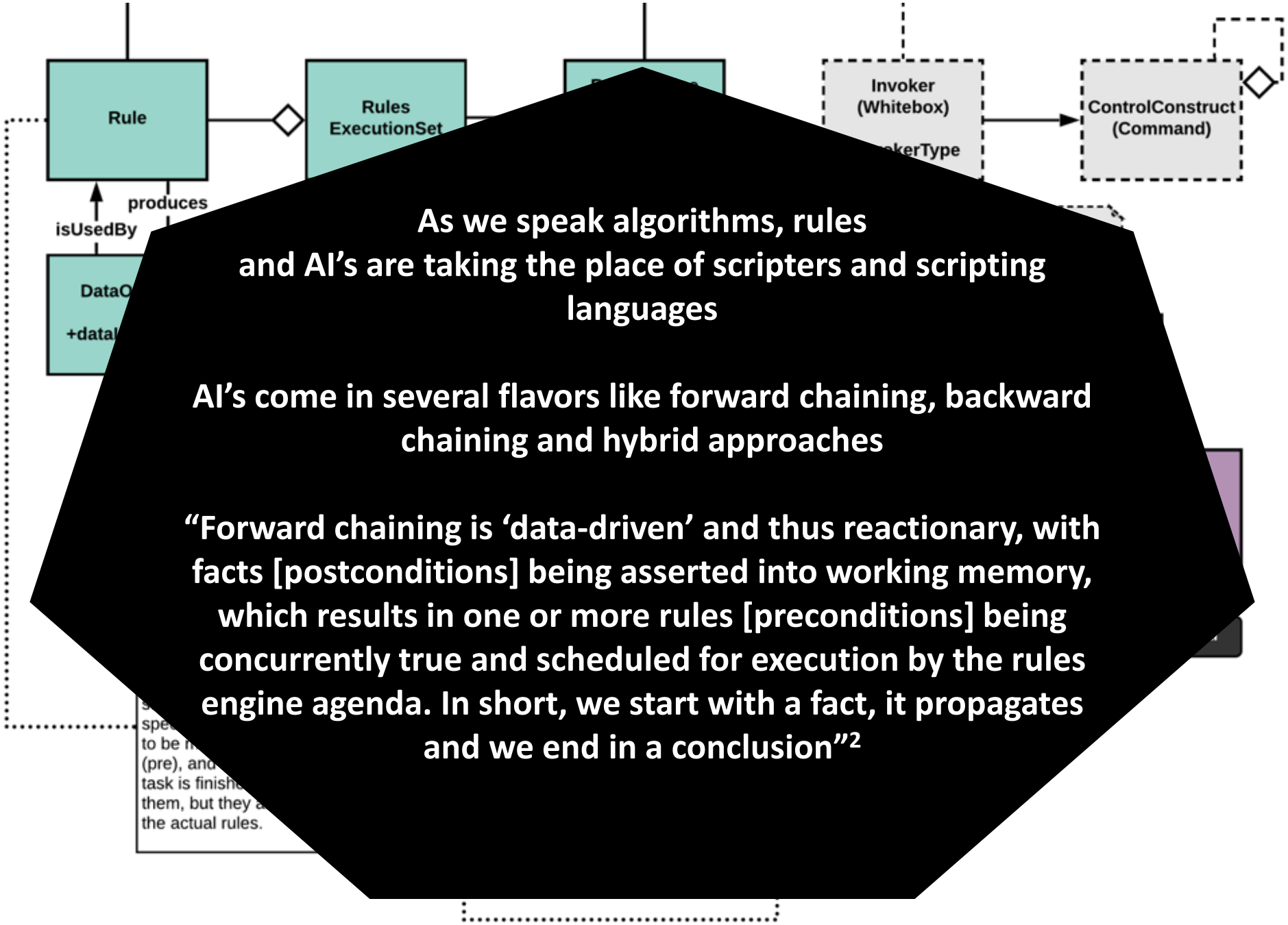
Scripted orchestration…




Rule
Rules ExecutionSet
Invoker (Whitebox)
ControlConstruct (Command)
produces
isUsedBy
invokes
Physical
Logical
invokes (legacy)
Today's platforms support a mixed model where tasks are invoked by a rules engine and process steps and services are scripted in one or more scripting languages that invoke control constructs.
The platforms of the future, however, won't be scripted. They will be orchestrated by rules, algorithms and AI's…


Process Controller
ProcessControl Component
Rule
Rules ExecutionSet
RulesEngine (Blackbox)
+rulesEngineType
Invoker (Whitebox)
+invokerType
ControlConstruct (Command)
produces
isUsedBy
DataObject
+dataIdentifier
Invoker and ControlConstruct (Command) are elements in the GoF command design pattern
invokes
Conceptual
Logical
Subtask
DataPipeline
BusinessProcess (Task)
ProcessStep
Service (Atomic)
Physical
Precondition
Postcondition
Pre/post conditions are logic expressions. They could be based on parameters or something else. They are declarative statements, i.e. assertions, specifying the conditions that have to be met before executing a task (pre), and the conditions after the task is finished (post). Rules use them, but they are separate from the actual rules.

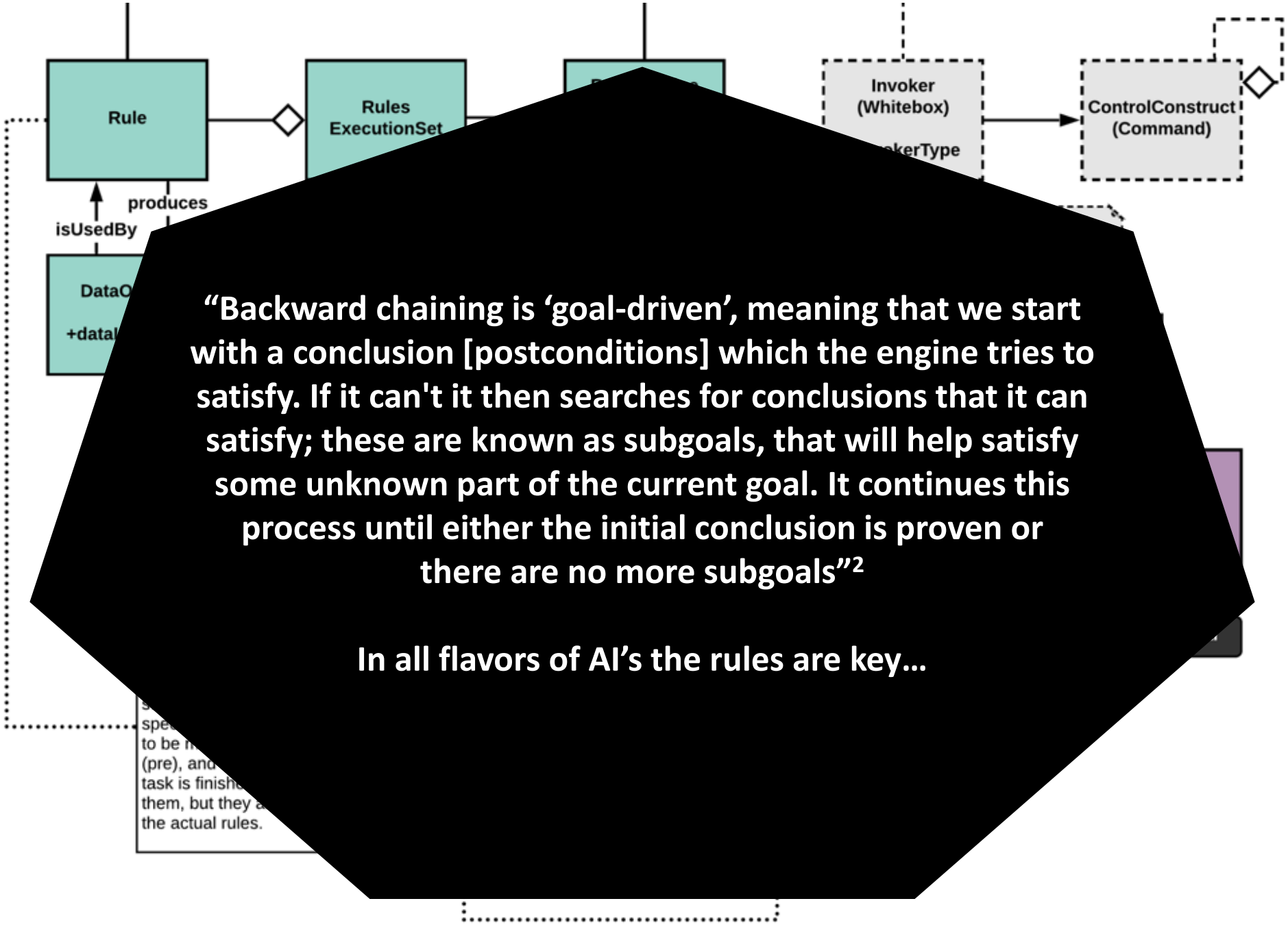

Rule
Rules ExecutionSet
RulesEngine (Blackbox)
+rulesEngineType
Invoker (Whitebox)
+invokerType
ControlConstruct (Command)
produces
isUsedBy
DataObject
+dataIdentifier
Invoker and ControlConstruct (Command) are elements in the GoF command design pattern
invokes
Conceptual
Logical
Subtask
DataPipeline
BusinessProcess (Task)
ProcessStep
Service (Atomic)
Physical
Precondition
Postcondition
Pre/post conditions are logic expressions. They could be based on parameters or something else. They are declarative statements, i.e. assertions, specifying the conditions that have to be met before executing a task (pre), and the conditions after the task is finished (post). Rules use them, but they are separate from the actual rules.



As we speak algorithms, rules and AI's are taking the place of scripters and scripting languages

AI's come in several flavors like forward chaining, backward chaining and hybrid approaches

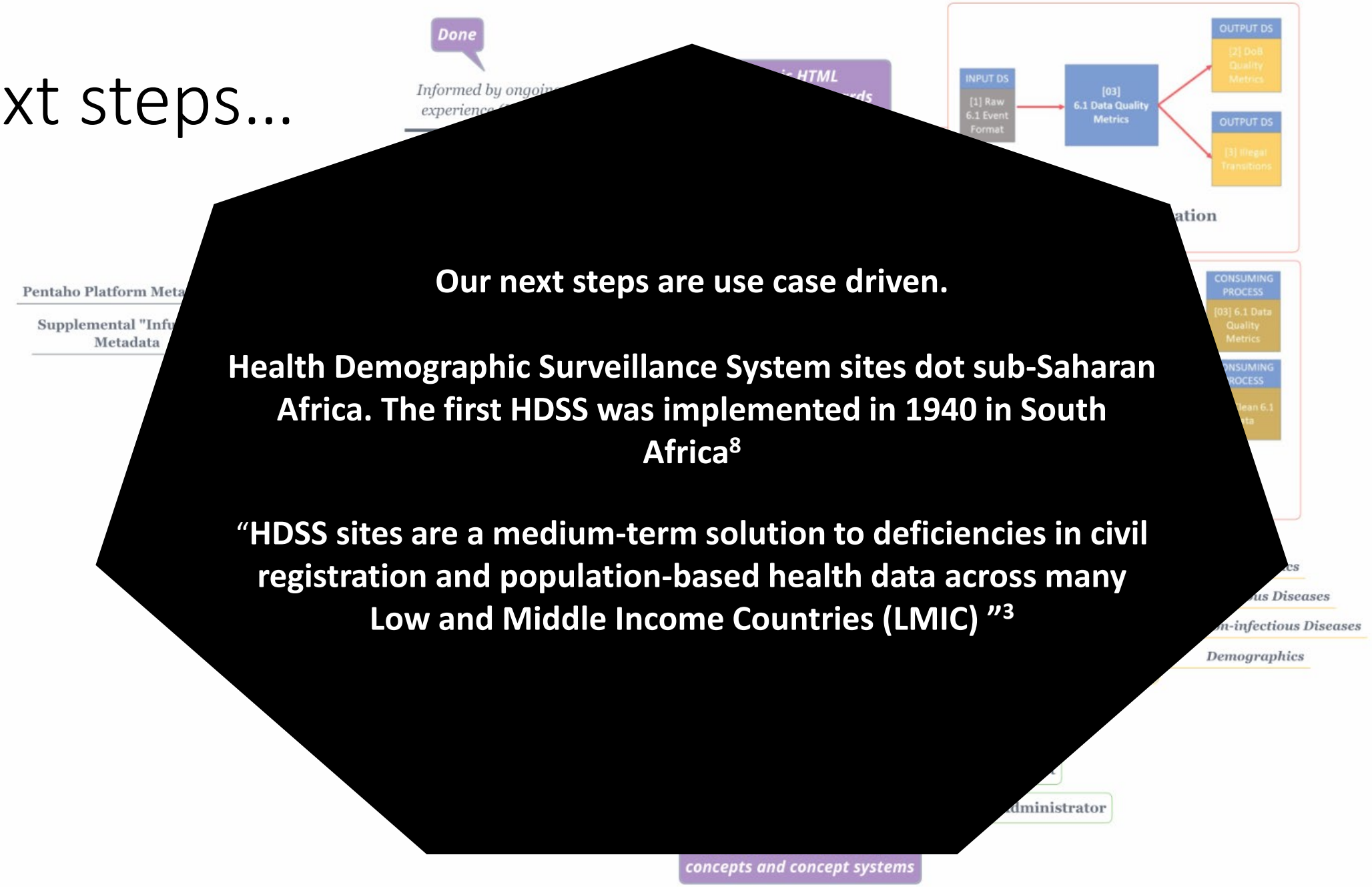
"Forward chaining is 'data-driven' and thus reactionary, with facts [postconditions] being asserted into working memory, which results in one or more rules [preconditions] being concurrently true and scheduled for execution by the rules engine agenda. In short, we start with a fact, it propagates and we end in a conclusion"[2]



“Backward chaining is ‘goal-driven’, meaning that we start with a conclusion [postconditions] which the engine tries to satisfy. If it can't it then searches for conclusions that it can satisfy; these are known as subgoals, that will help satisfy some unknown part of the current goal. It continues this process until either the initial conclusion is proven or there are no more subgoals”[2]

In all flavors of AI’s the rules are key…

Next steps…

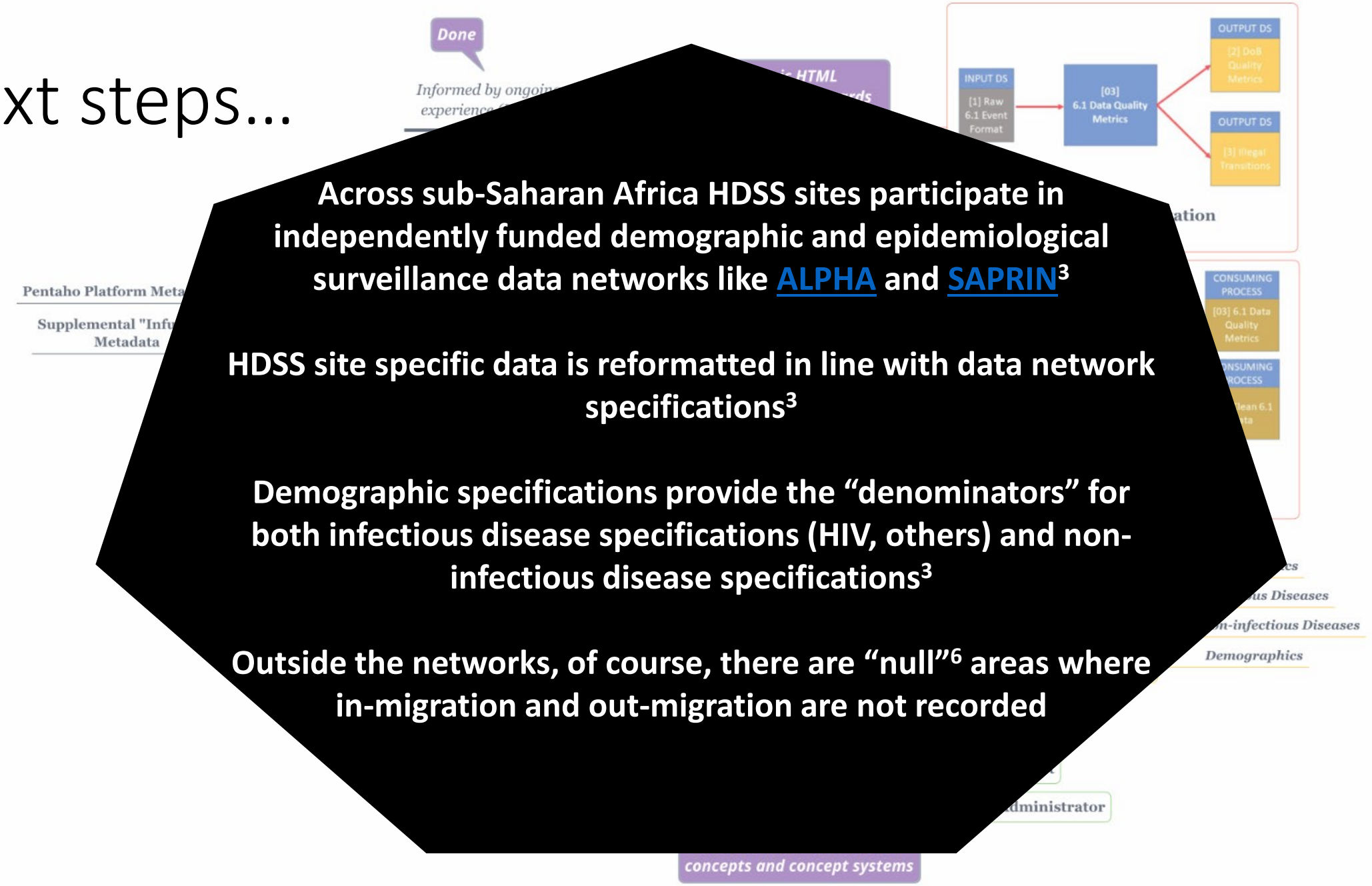


Our next steps are use case driven.

Health Demographic Surveillance System sites dot sub-Saharan Africa. The first HDSS was implemented in 1940 in South Africa[8]

"HDSS sites are a medium-term solution to deficiencies in civil registration and population-based health data across many Low and Middle Income Countries (LMIC) "[3]

Next steps…



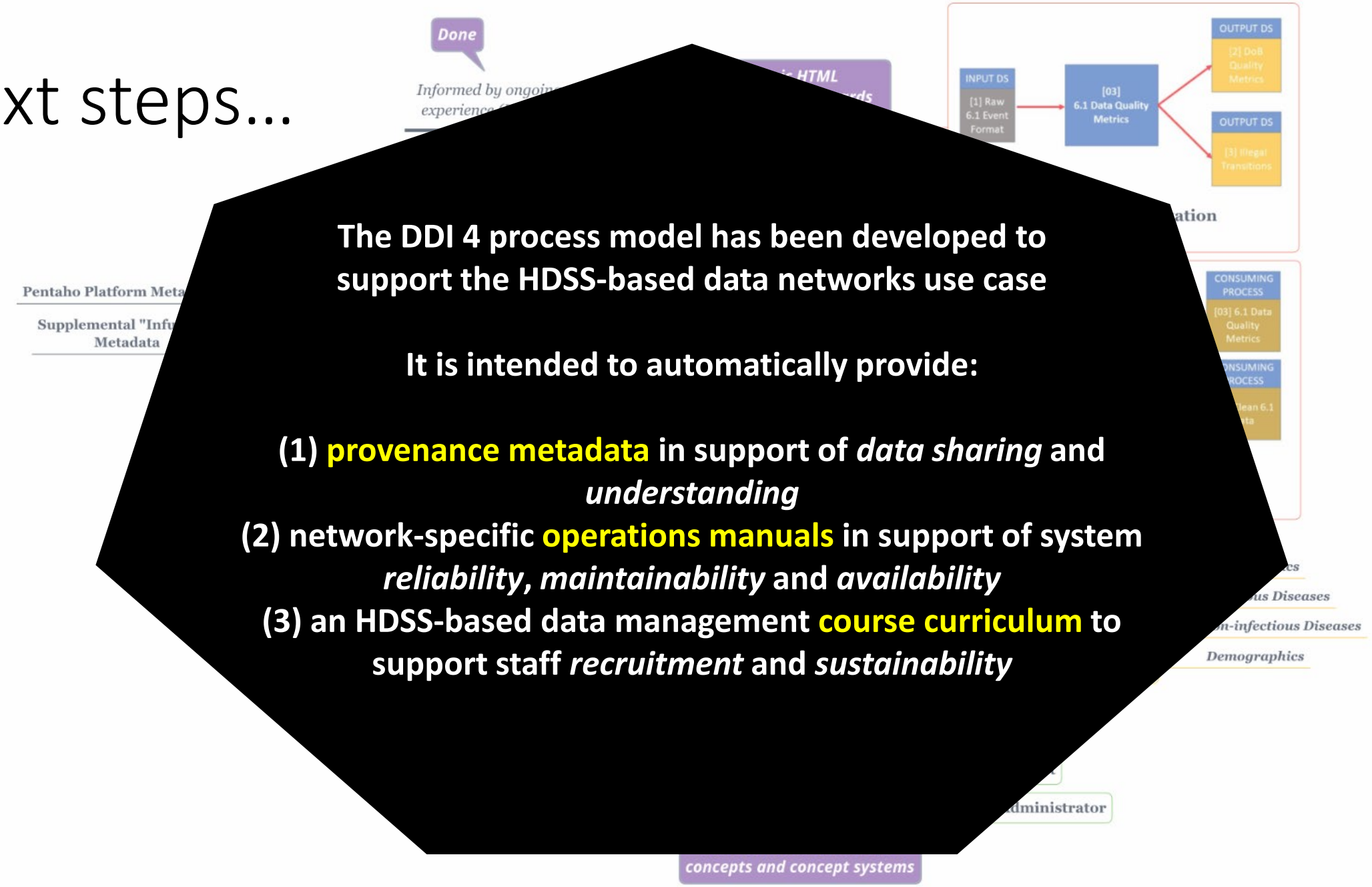
Across sub-Saharan Africa HDSS sites participate in independently funded demographic and epidemiological surveillance data networks like ALPHA and SAPRIN[3]

HDSS site specific data is reformatted in line with data network specifications[3]

Demographic specifications provide the "denominators" for both infectious disease specifications (HIV, others) and non-infectious disease specifications[3]

Outside the networks, of course, there are "null"[6] areas where in-migration and out-migration are not recorded

Next steps…

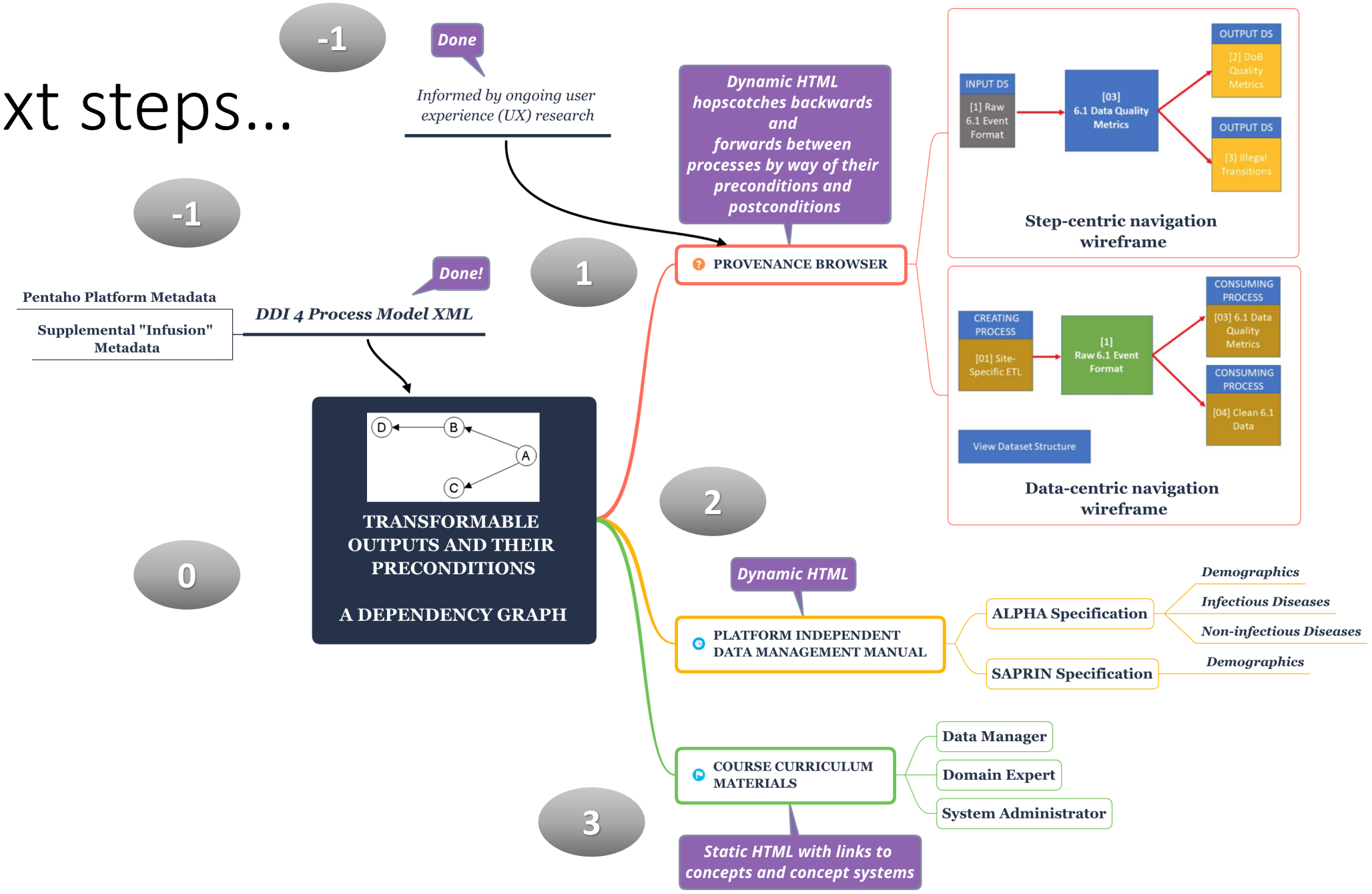


The DDI 4 process model has been developed to support the HDSS-based data networks use case

It is intended to automatically provide:

(1) provenance metadata in support of *data sharing* and *understanding*
(2) network-specific operations manuals in support of system *reliability*, *maintainability* and *availability*
(3) an HDSS-based data management course curriculum to support staff *recruitment* and *sustainability*

Next steps…


-1
Done
Informed by ongoing user experience (UX) research
Dynamic HTML hopscotches backwards and forwards between processes by way of their preconditions and postconditions
-1
Done!
Pentaho Platform Metadata
Supplemental "Infusion" Metadata
DDI 4 Process Model XML
1
PROVENANCE BROWSER
INPUT DS
[1] Raw 6.1 Event Format
[03] 6.1 Data Quality Metrics
OUTPUT DS
[2] DoB Quality Metrics
OUTPUT DS
[3] Illegal Transitions
Step-centric navigation wireframe
CREATING PROCESS
[01] Site-Specific ETL
[1] Raw 6.1 Event Format
CONSUMING PROCESS
[03] 6.1 Data Quality Metrics
CONSUMING PROCESS
[04] Clean 6.1 Data
View Dataset Structure
Data-centric navigation wireframe
D
B
A
C
TRANSFORMABLE OUTPUTS AND THEIR PRECONDITIONS
A DEPENDENCY GRAPH
2
0
Dynamic HTML
PLATFORM INDEPENDENT DATA MANAGEMENT MANUAL
ALPHA Specification
Demographics
Infectious Diseases
Non-infectious Diseases
SAPRIN Specification
Demographics
COURSE CURRICULUM MATERIALS
Data Manager
Domain Expert
System Administrator
3
Static HTML with links to concepts and concept systems

References

1. Gryaznov, Egor: Business Rules on Hadoop
2. jboss.org: Chapter 1. The Rules Engine
3. Kanjala, Chifundo: Provenance of “after the fact” harmonised community-based demographic and HIV surveillance data from ALPHA cohorts. London School of Hygiene and Tropical Medicine. Ph.D. Thesis. 2019
4. Mastertheboss: What is a rules engine?
5. Mastertheboss: Drools – Introduction
6. Null States is Book 2 of the Centenal Cycle by Malka Older. In the Centenal Cycle the decline of nation states created during the colonial period is chronicled together with the rise of alternative forms of social organization based on Information. Null states are informationally opaque
7. Rizzolo, Flavio. Note on Pre/post conditions. Personal communication
8. Rosario VN, Costa D, Francisco D, Brito M. HDSS Profile: The Dande Health and Demographic Surveillance System (Dande HDSS, Angola). Int J Epidemio. 2017 May 3; 46: 1094
9. Wikipedia: Business rules engine
10. Wikipedia: Command pattern