Experience in Using a Process Language to Define Scientific Workflow and Generate Dataset Provenance

Leon J. Osterweil[1], Lori A. Clarke[1], Aaron Ellison[2], Rodion Podorozhny[3], Alexander Wise[1], Emery Boose[2], and Julian Hadley[2]

[1]Lab. for Advanced SW Engineering Research (LASER), Dept of Computer Science Univ. of Massachusetts Amherst, MA 01003

[2]Harvard Forest, Harvard University, Petersham, MA

[3]Dept. of Computer Science, Texas State Univ., San Marcos, TX

Or:
Why software engineers should be interested in scientific data and how it is processed

Or:
Why software engineers should be interested in scientific data and how it is processed

Scientific data management raises problems in:

- Configuration Management
- Abstraction/modularity
- Exception management
- Concurrency control
- Static Analysis
  - --and more

Or:
Why software engineers should be interested in scientific data and how it is processed

Scientific data management raises problems in:

- **Configuration Management**
- **Abstraction/modularity**
- **Exception management**
- **Concurrency control**
- **Static Analysis**
- **--and more**

Addressing these Problems can shed light on fundamental issues in software engineering

Our approach

- Evaluate applying the principles of
  - Modern programming languages
  - Software engineering
  - Other CS domains
- In the form of the *Analytic Web* concept
  - Two interrelated types of graphs
- To see how this helps address the problems of scientists
  - And what it tells us about the CS domains

Our approach

- Evaluate applying the principles of
  - Modern programming languages
  - Software engineering
  - Other CS domains
- In the form of the *Analytic Web* concept
  - Two interrelated types of graphs
- To see how this helps address the problems of *scientists*
  - And what it tells us about the CS domains

"You're not a real scientist, you're a Computer Scientist"

--A Physicist/Astronomer to Lee

"Computer Scientists do real science, but it is a science of abstract, non-tangible things. And this is a science that you need our help with"

Lee, back to Physicist/Astronomer

So, What do real scientists do (in the abstract)?: The Traditional Structure of Science--The Scientific Method

- Hypothesis
- Experimental Design
  - Define experimentation process
- Experimentation
  - Execute the process to produce data
- Evaluation
  - Analyze datasets, producing more datasets
- Reproduction by others
  - Reapply processes to compare results and increase confidence in the conclusions

Focus on Datasets

- Datasets
  - Their production
  - Their reproduction by others
  - Their use by consumers
    - E.g., In producing subsequent datasets
- Dataset Provenance
  - Document where datasets come from
  - Who, What, When, Where (Typical Metadata)
  - How
    - What computer programs
    - What cleansing and interpolation algorithms
    - Process Provenance Metadata

Real Datasets: Real Complications

- Is there really any "raw data"?
  - Mostly recorded automatically
  - Often preprocessed by computers
- Considerable "cleaning" of datasets
  - Some by humans
  - Some by automation
- Documentation is usually scant or absent
- Reproduction is hard or impossible
  - Processes are complex and hard to document
- Real threat to the foundations of science
  - Reproducibility is the bedrock of science

Perils in Inadequate Dataset Documentation

- Some datasets are wrong
  - Incorrectly cleaned
  - Inappropriate algorithms applied incorrectly
  - Incompletely processed
- Statistical data is often misunderstood/misused
  - Used under assumptions that are unwarranted
- Incorrect/incorrectly used data can be the basis for gravely important decisions
  - Clear cut forests
  - Release of unsafe pharmaceuticals

How to know which datasets to trust?

The Internet Aggravates These Problems

- Scientific datasets are published
  - In the literature
  - ON THE WEB (NSF mandate)
- What might be published?
  - All the data
  - Analyzed datasets
  - With all of the "fudge factors"?
- What does get published
  - Highly processed datasets
  - With important details omitted

➢THE DETAILS MATTER A GREAT DEAL

Key Problems to be Addressed

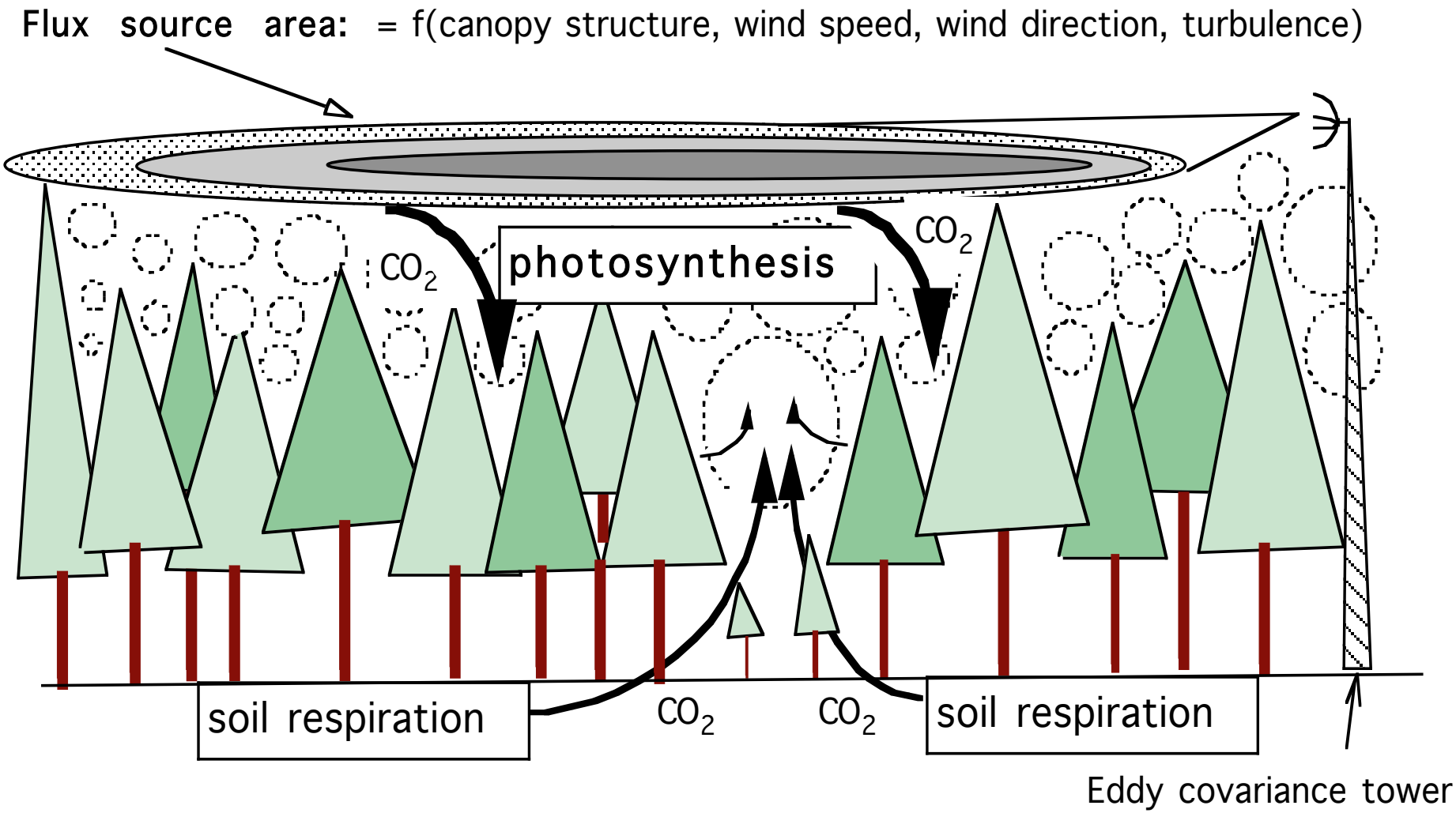
- Producers need automation help for
  - Generation
  - Distribution
  - Provenance documentation
- Consumers need help for
  - Understanding
  - Reproduction
  - Regeneration
  - Reuse

These are primarily process issues

A Real Ecological Research Problem: The Water Budget Problem

- **Measure and analyze the flow of water through a small forested watershed**
- **dS = P - ET- Q**
- **Precipitation (*P*)**
  - Two gauges, P1, P2 used to minimize gaps in data.
- **Surface Discharge (*Q*)**
  - Measured at a stream gauge
  - Gaps caused by sensor failure, ice build-up, etc.
  - Fill as a function of preceding *P* and *Q*.
- **Evapotranspiration (*ET*)**
  - measured at an eddy-flux tower.
- **Photosynthetically active radiation (*PAR*)**
  - *PAR* can be used to estimate *ET*

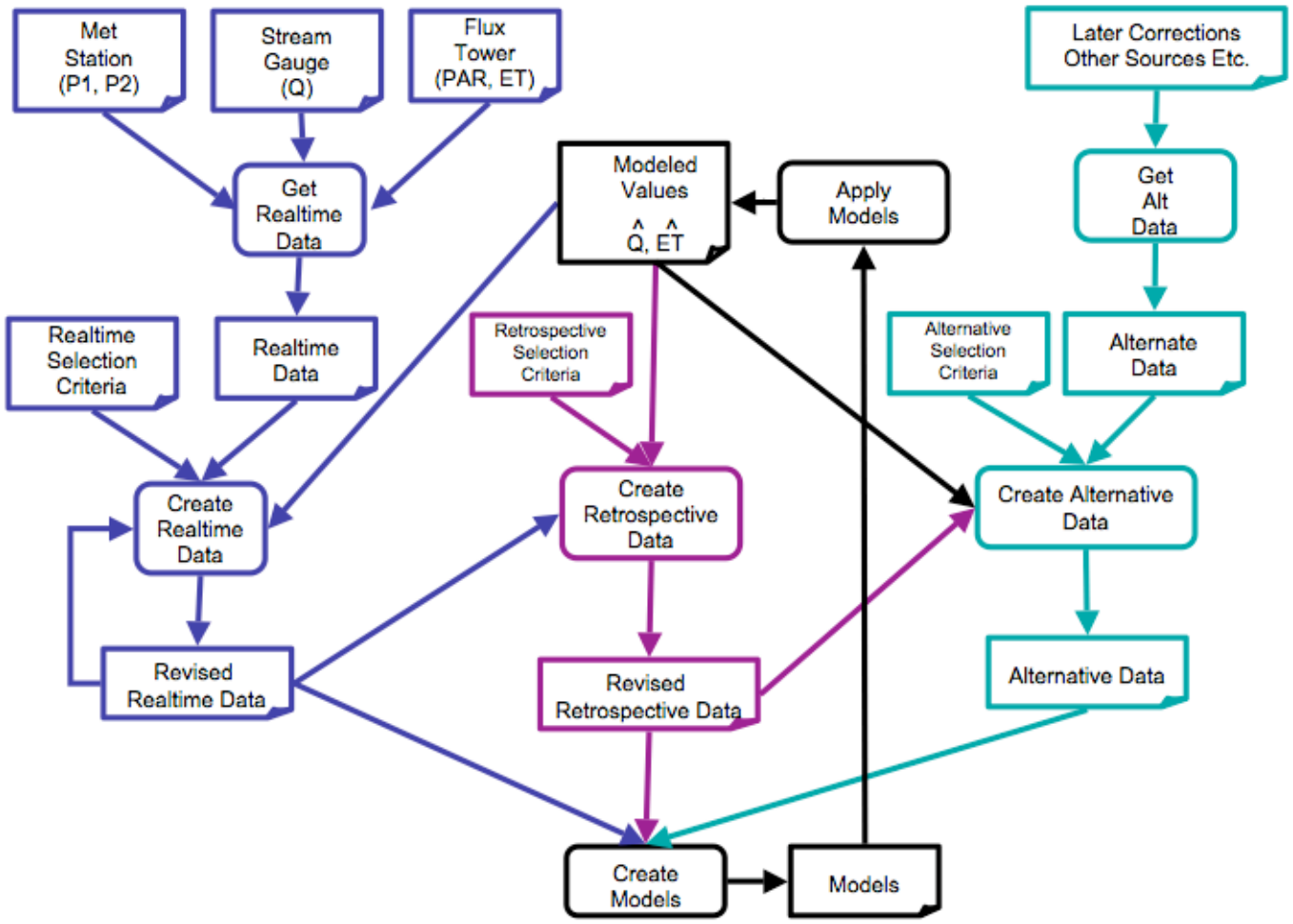
An Eddy Covariance (Flux) Tower



Key Features of the Water Budget Process

- Frequent measurements taken in real-time
  - Some wirelessly
  - Considerable manipulation in real-time
- Datasets published in real-time(?)
- 30-day retrospective revision of dataset
  - Maybe revisions at other (more immediate?) intervals
- Other revisions done at other times
  - Adjustments for sensor drift
  - Alternative model evaluation
  - Use of data from other sites (?)
- Different scientists want to do different things
- ????

DFG of Water Budget Process



Some key issues to be dealt with

- Processes are complex
  - Intertwined iterations
  - Exceptions (not shown) interrupt nominal flow
  - Concurrency
- Create profusions of artifact instances
  - How to keep track of multiple instances of a single dataset type?
  - Need an artifact structure
  - And configuration management
- Still other issues
  - Mixed human and automated agents

DFG does not do well with some of these issues

- Producers generate datasets by executing paths
  - DFG has some spurious unexecutable paths
- Consumers need provenance documentation
  - How datasets were generated
  - How data items were created
  - But DFG nodes are types. There are no instances
- Both need reproducibility
  - Sufficient details are needed
  - And can be hard to produce
- Both need analyses
  - Which requires rigor and precision
  - As well as details

Our Proposed Approach: Analytic Web

- Represent scientific datasets as a web structure specifying:
  - What data were used to derive each new dataset
  - The process by which each dataset was derived
- Use two complementary graphs
  - Process Definition Graph (PDG) (a visual program)
    - Defines process in sufficient detail to support
      - Understanding, Execution, Reproducibility, Scrutiny
  - Data Derivation Graph (DDGs) (program traces)
    - Documents how each dataset instance was actually derived
      - In terms of actual process steps applied and choices made

The Two Graphs

- Process Derivation Graph (PDG)
  - Defines the process
  - Types of tasks that are applied to types of datasets
- Data Derivation Graph (DDG)
  - Which instances derived by which others
  - By which activities
- PDG execution leads to the creation of the DDG
- Dataset provenance metadata created from DDG nodes

The Process Definition Challenges

- How to represent processes
  - Accurately,completely,clearly,reproducibly
- Real processes are large and complex
  - Even "easy" ones
- How to support execution of these processes
- How to communicate the processes to working scientists
- How to validate these processes
  - Scientific scrutiny
  - Analysis to assure that composed processes are not anomalous

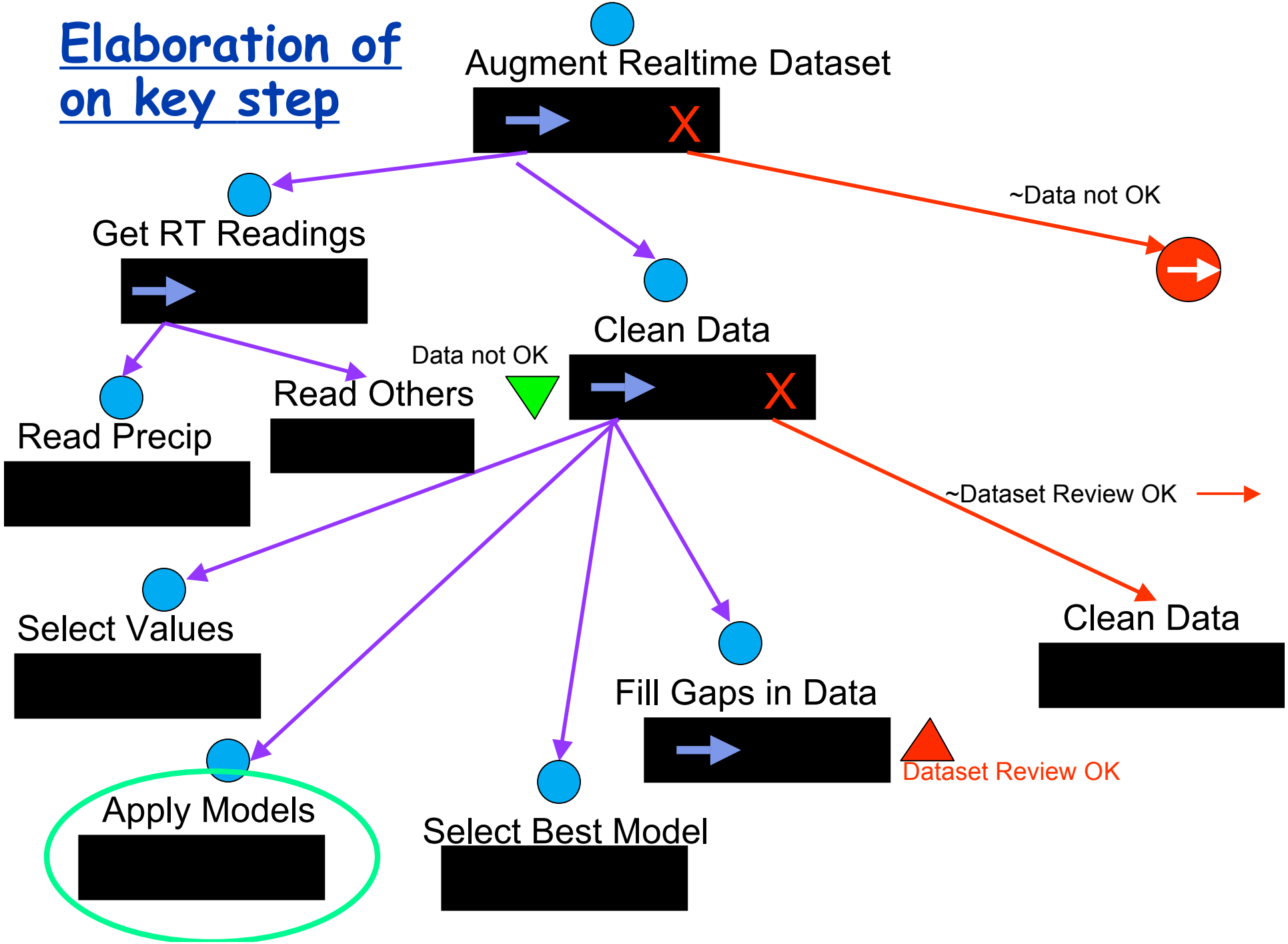
The Little-JIL Process Language

- Executable,visual process language emphasizing
  - Use of abstraction
  - Exception management
  - Concurrency control
  - Resource specification and late-binding
- Use it to define PDGs to determine the characteristics and features needed
- Execute the PDGs to generate DDGs
  - That define dataset provenance

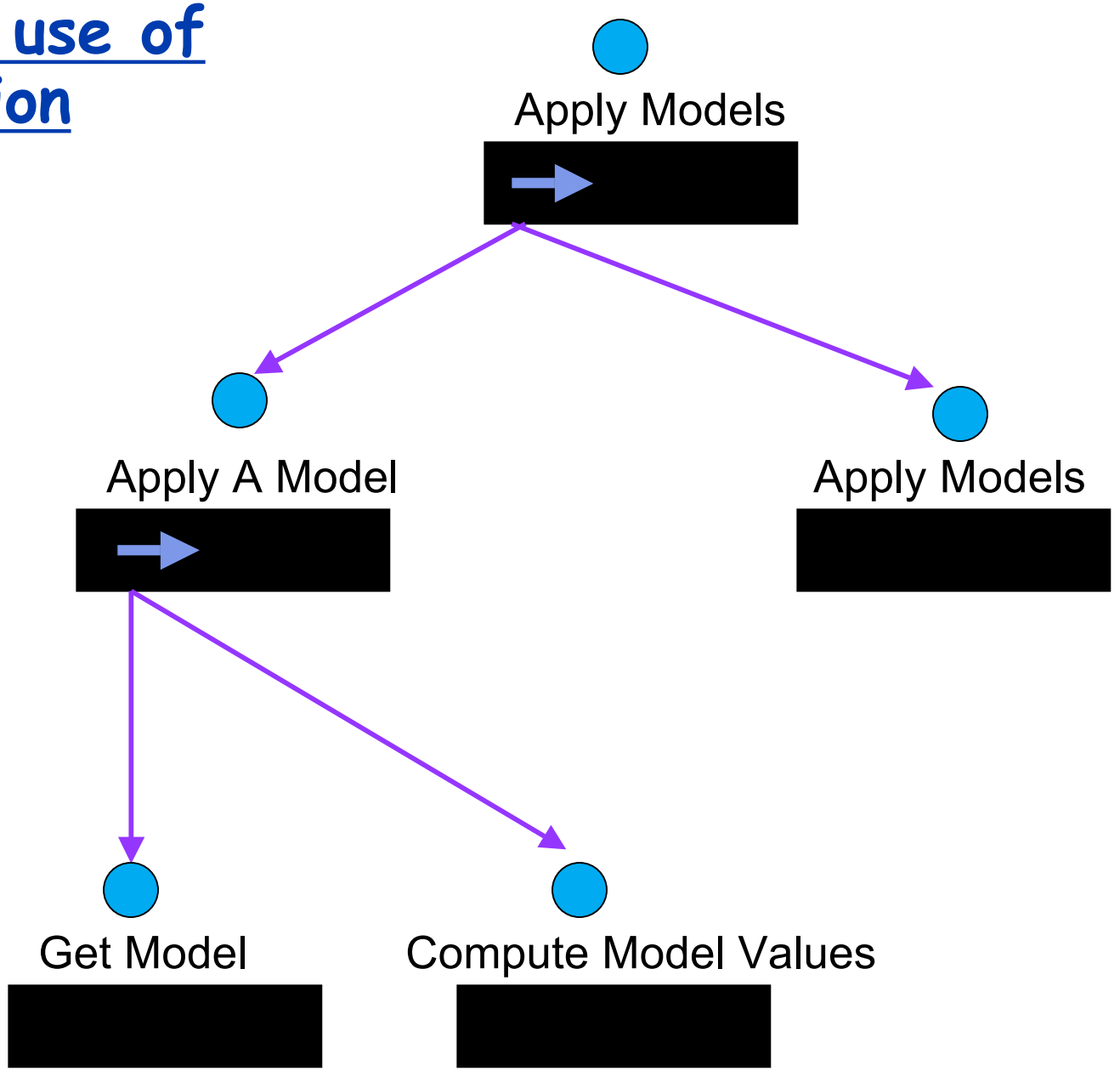
Value of Hierarchy, Scoping, and Abstraction

- Process definition is a hierarchical decomposition
- Think of steps as procedure invocations
  - They define scopes
  - Copy and restore argument semantics
  - Creation of scopes
- Encourages use of abstraction
  - Eg. process fragment reuse
- Use of recursion helps define, propagate, exploit context information to elucidate iteration

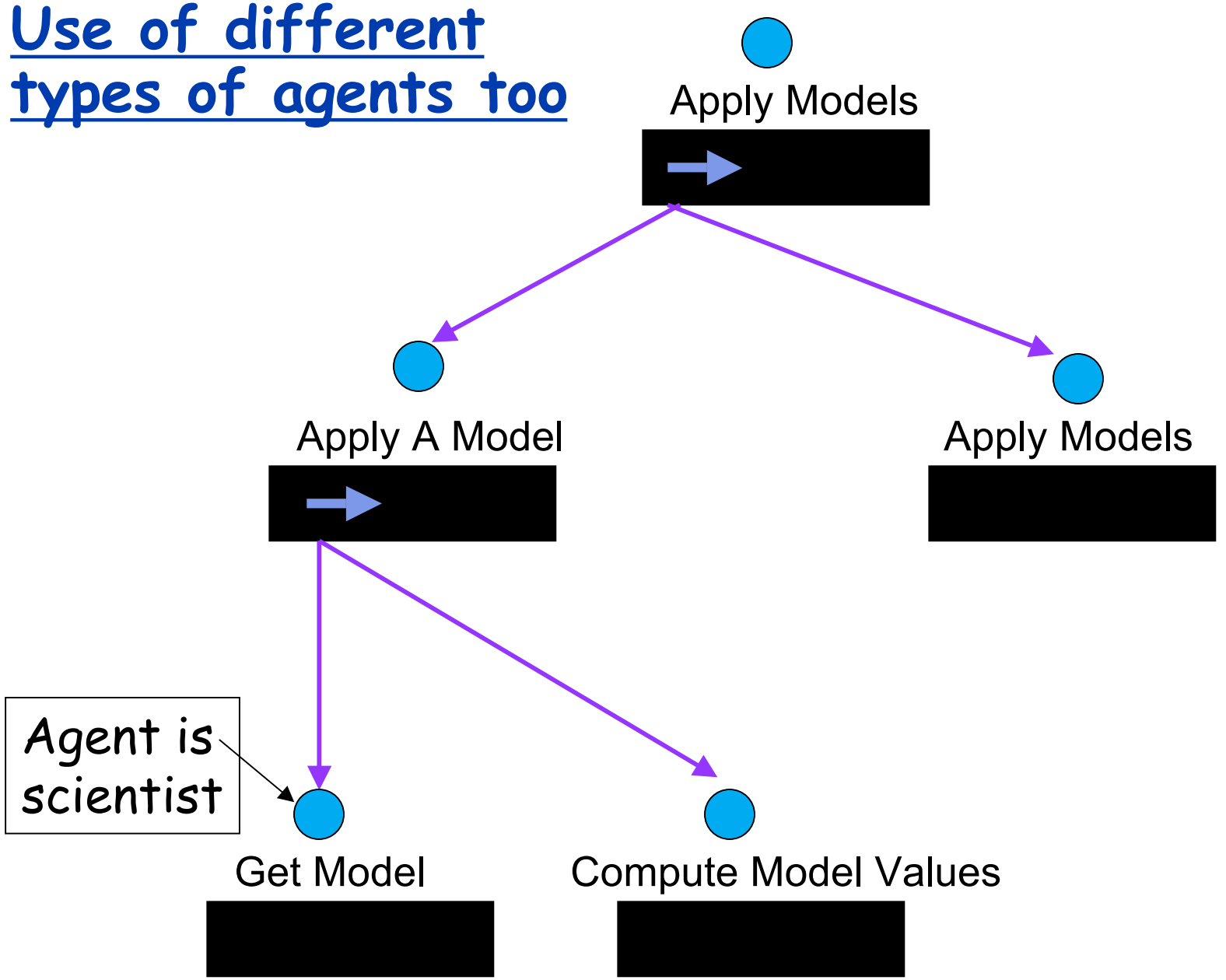
Elaboration of on key step



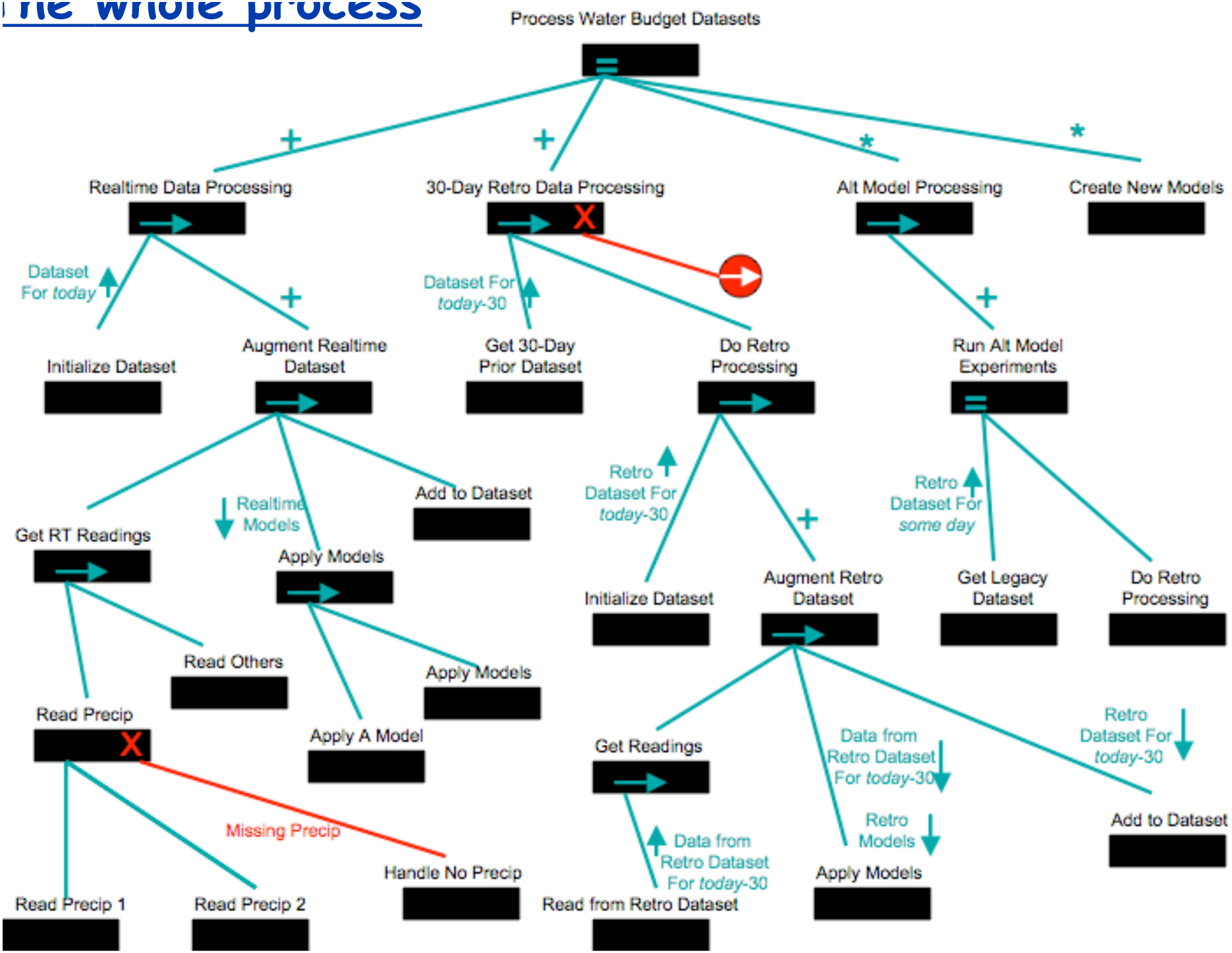
Makes use of recursion

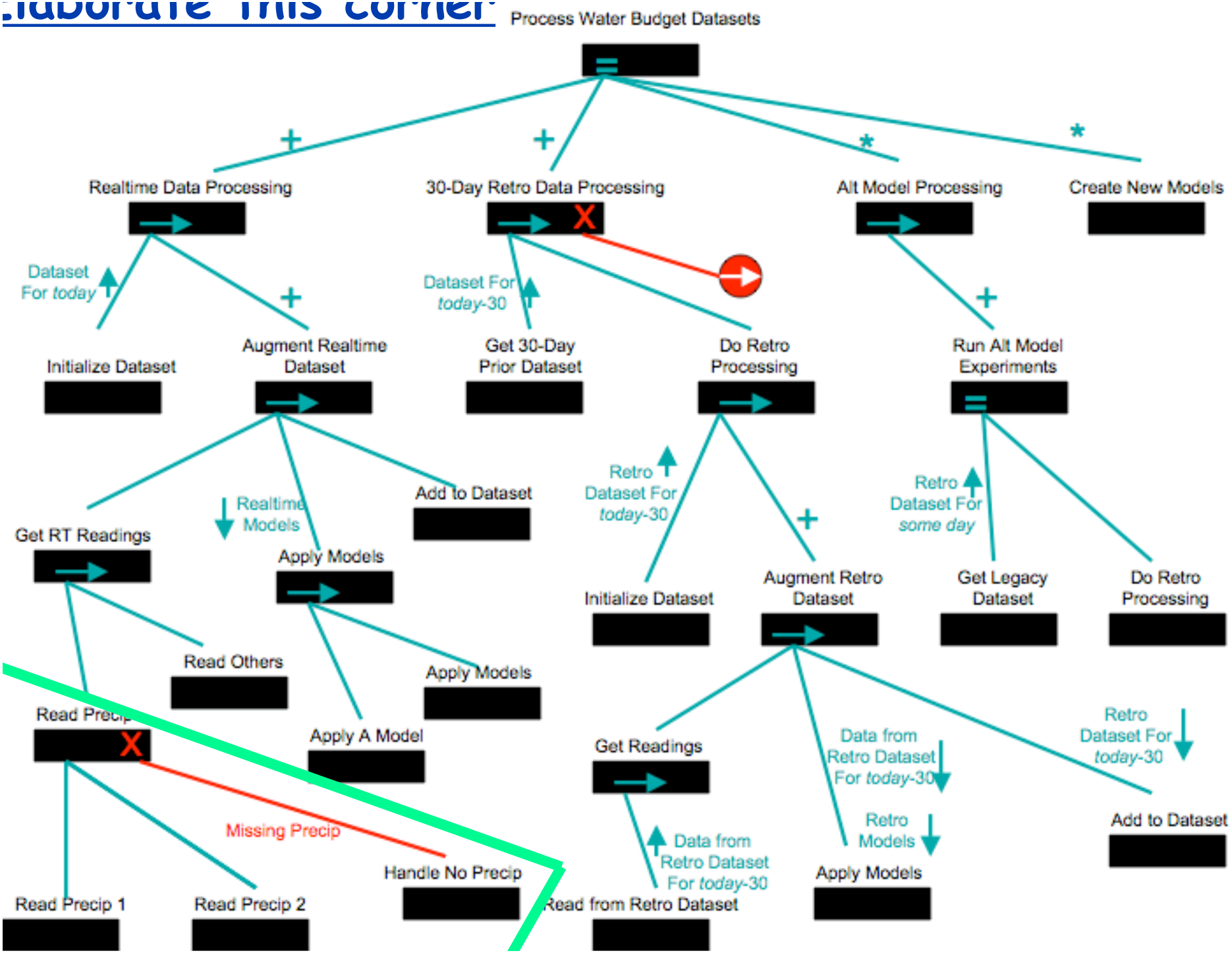


Use of different types of agents too

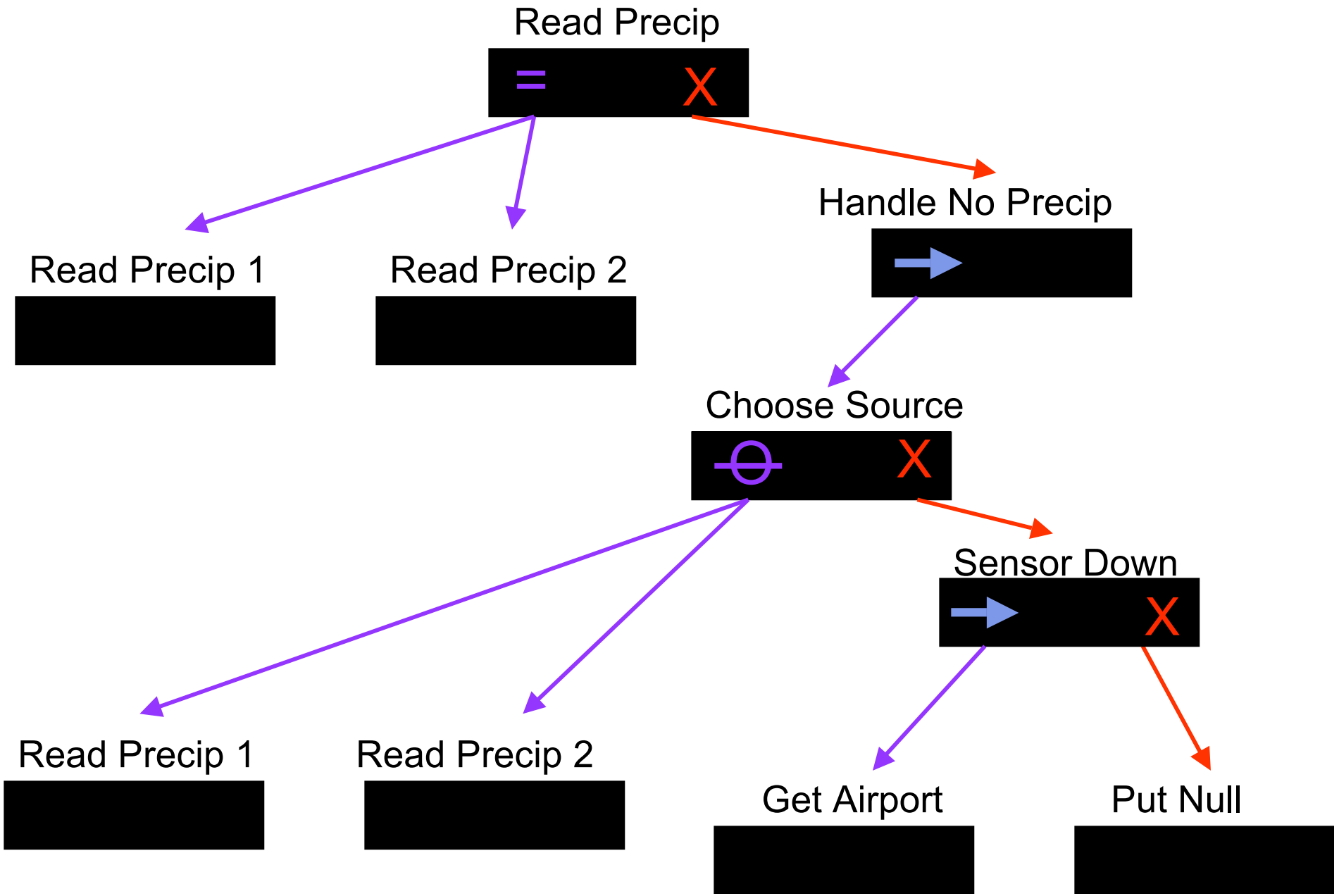


The whole process

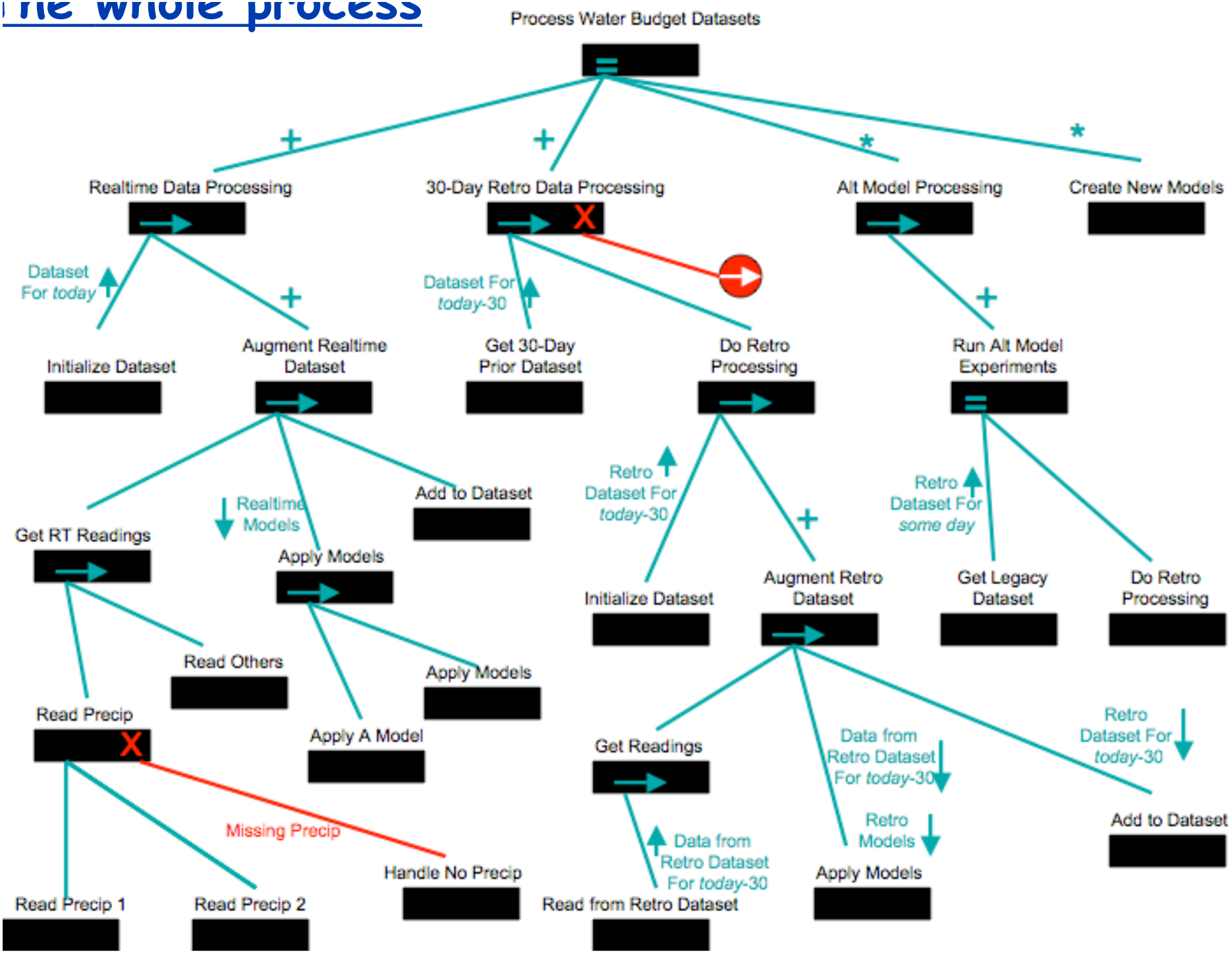



Process Water Budget Datasets
Realtime Data Processing
30-Day Retro Data Processing
Alt Model Processing
Create New Models
Dataset For today
Dataset For today-30
Initialize Dataset
Augment Realtime Dataset
Get 30-Day Prior Dataset
Do Retro Processing
Run Alt Model Experiments
Get RT Readings
Realtime Models
Add to Dataset
Apply Models
Retro Dataset For today-30
Retro Dataset For some day
Initialize Dataset
Augment Retro Dataset
Get Legacy Dataset
Do Retro Processing
Read Others
Apply Models
Apply A Model
Get Readings
Data from Retro Dataset For today-30
Retro Dataset For today-30
Add to Dataset
Missing Precip
Retro Models
Data from Retro Dataset For today-30
Handle No Precip
Apply Models
Read Precip 1
Read Precip 2
Read from Retro Dataset

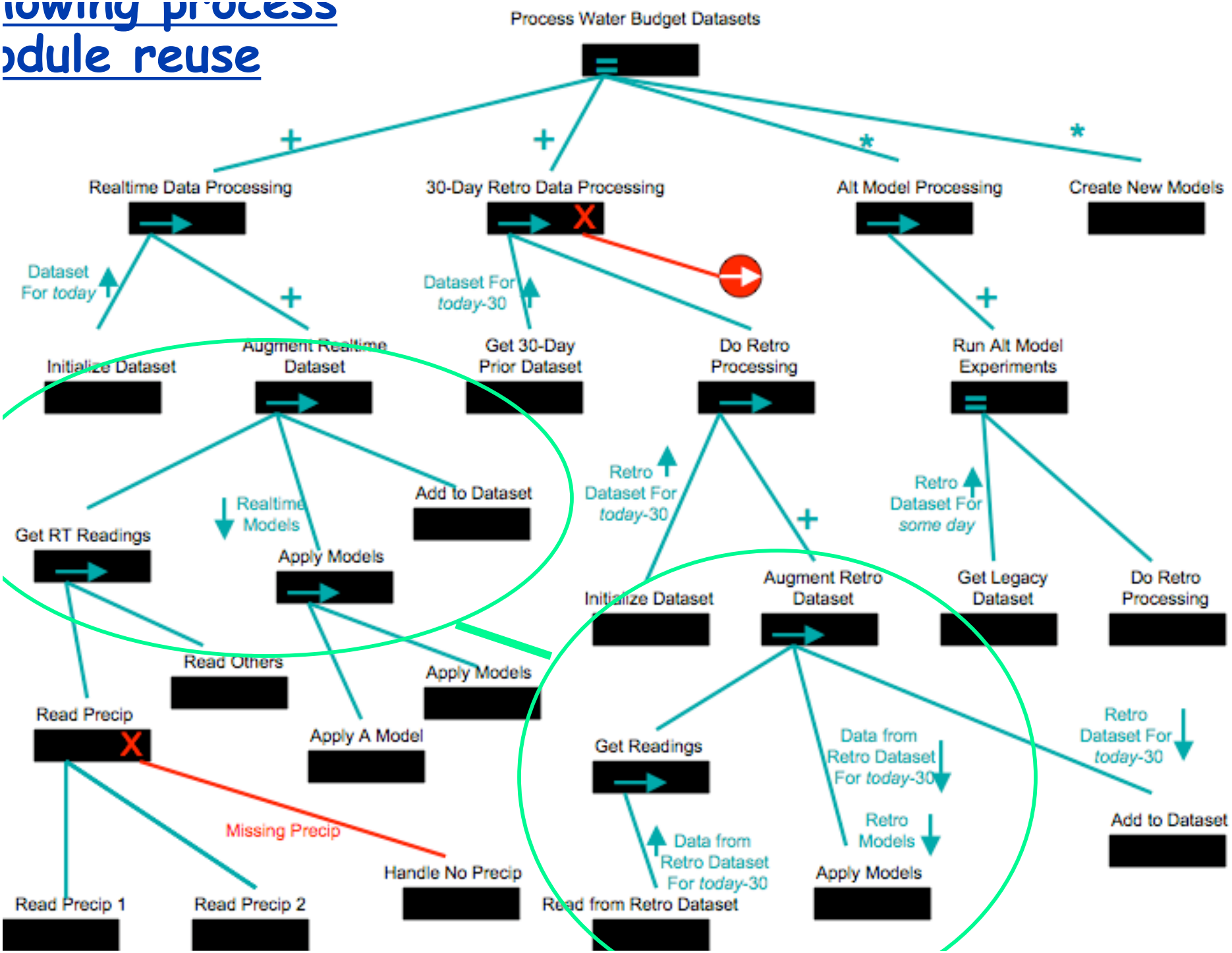
Need for complex exception handling



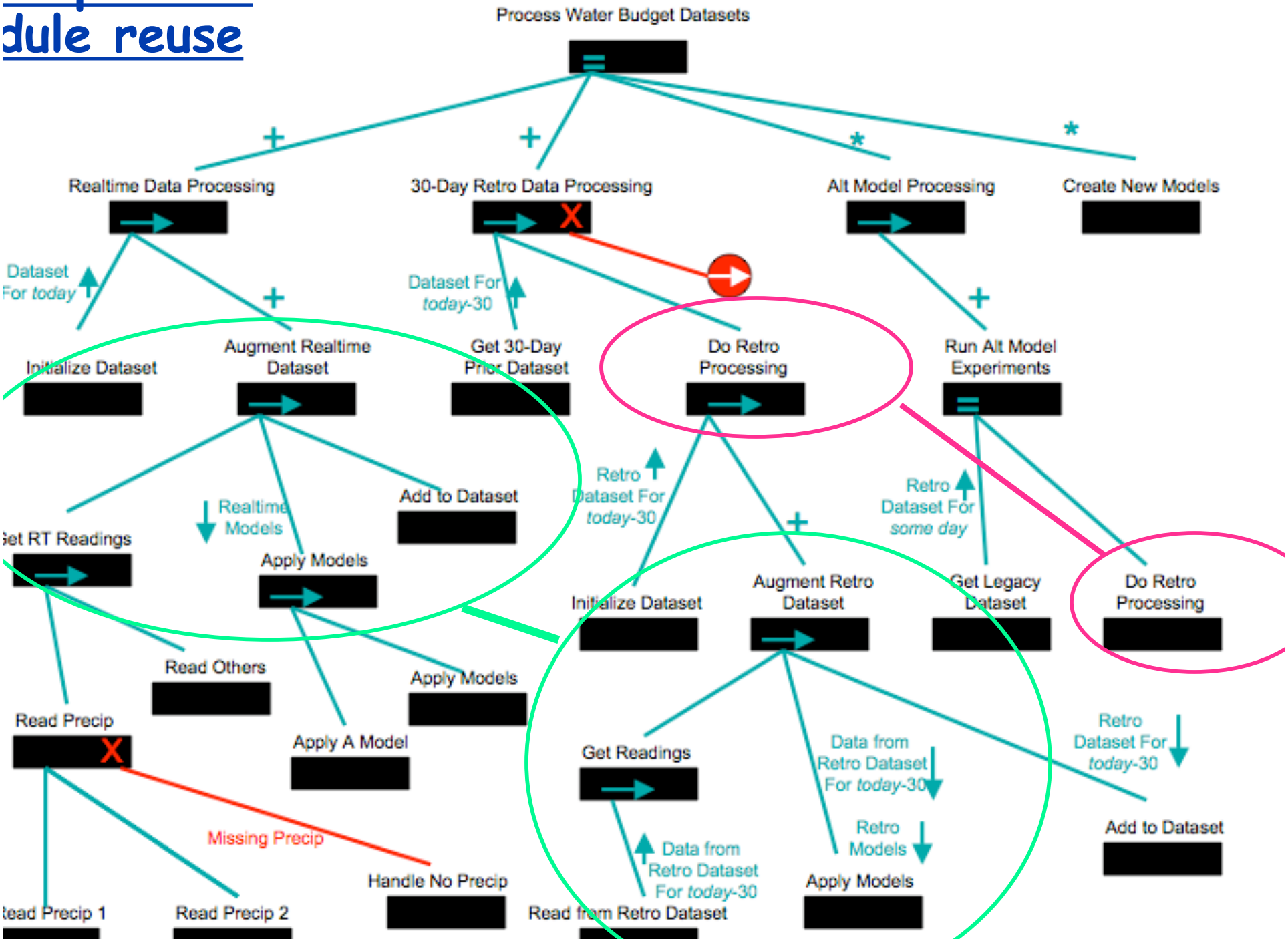
The whole process



lowing process
odule reuse



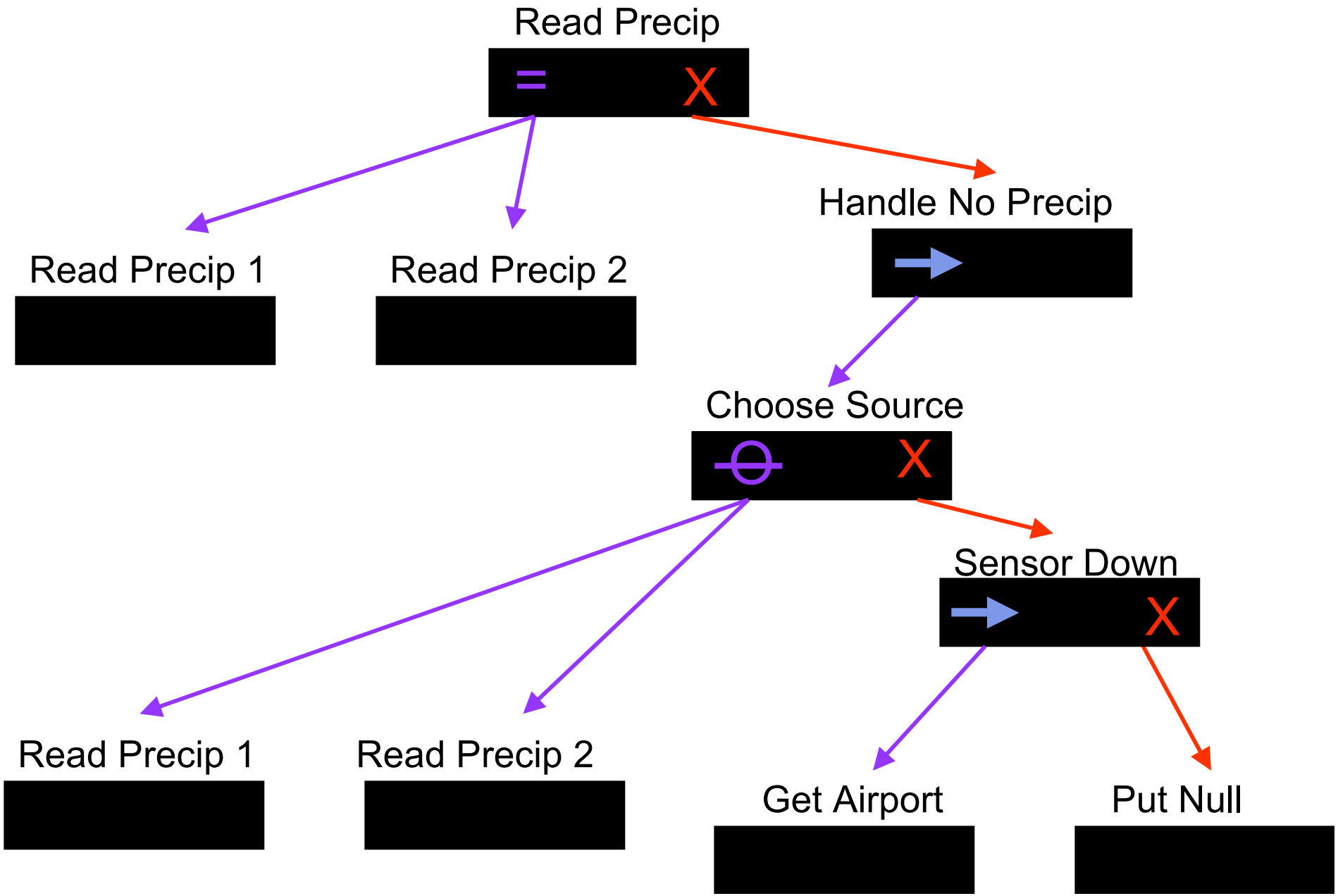
re process
dule reuse



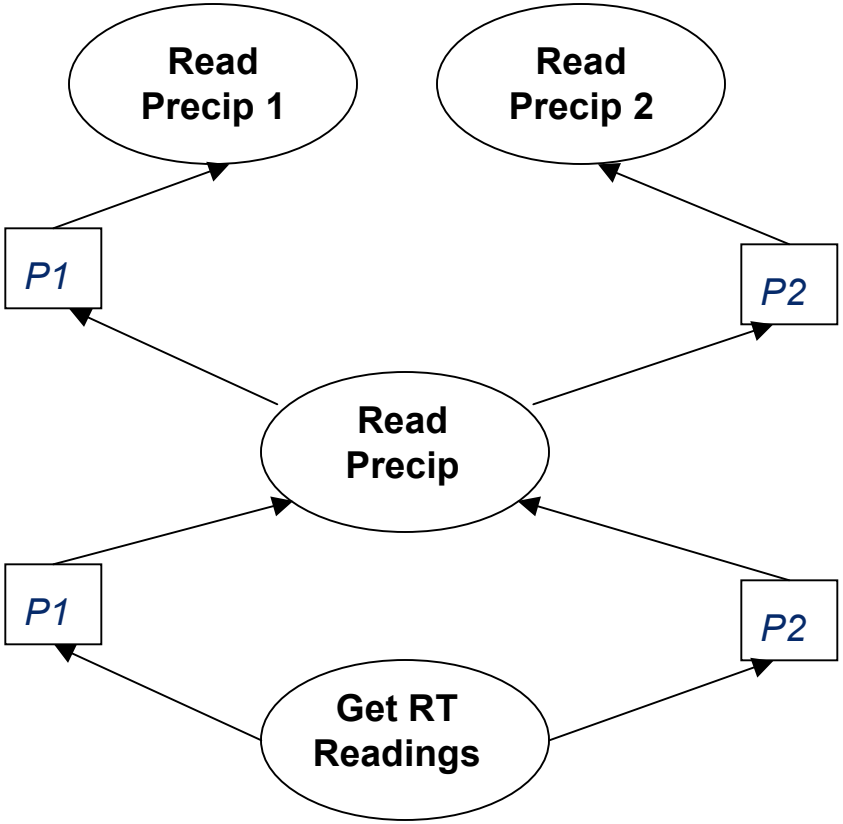
The DDG

- A DAG of instances
  - Dataset instances
  - Process step instances (and their agents)
- Edges indicate which datasets are
  - Derived from executing which step instance
  - Using which dataset instances as inputs

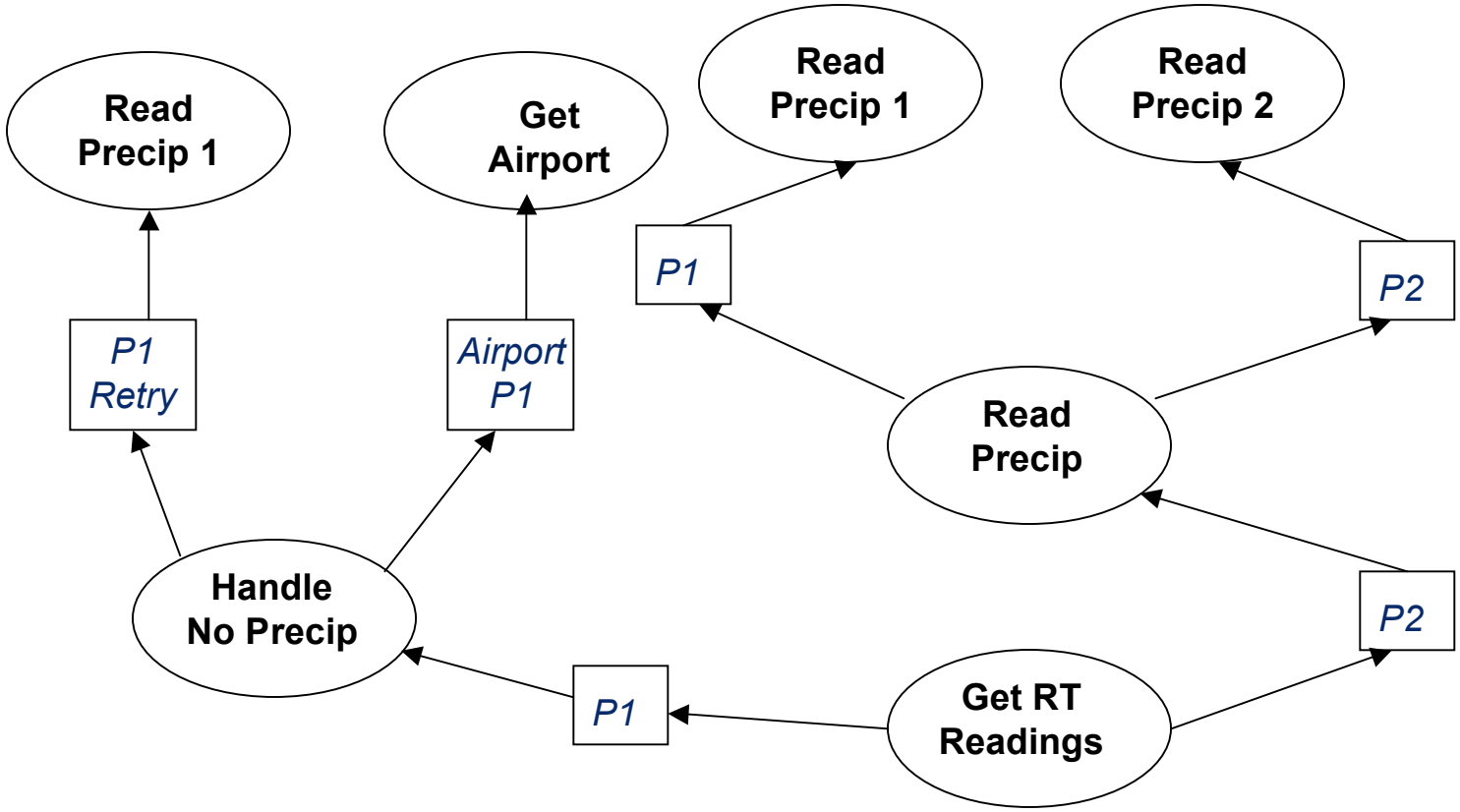
Different executions lead to different DDGs



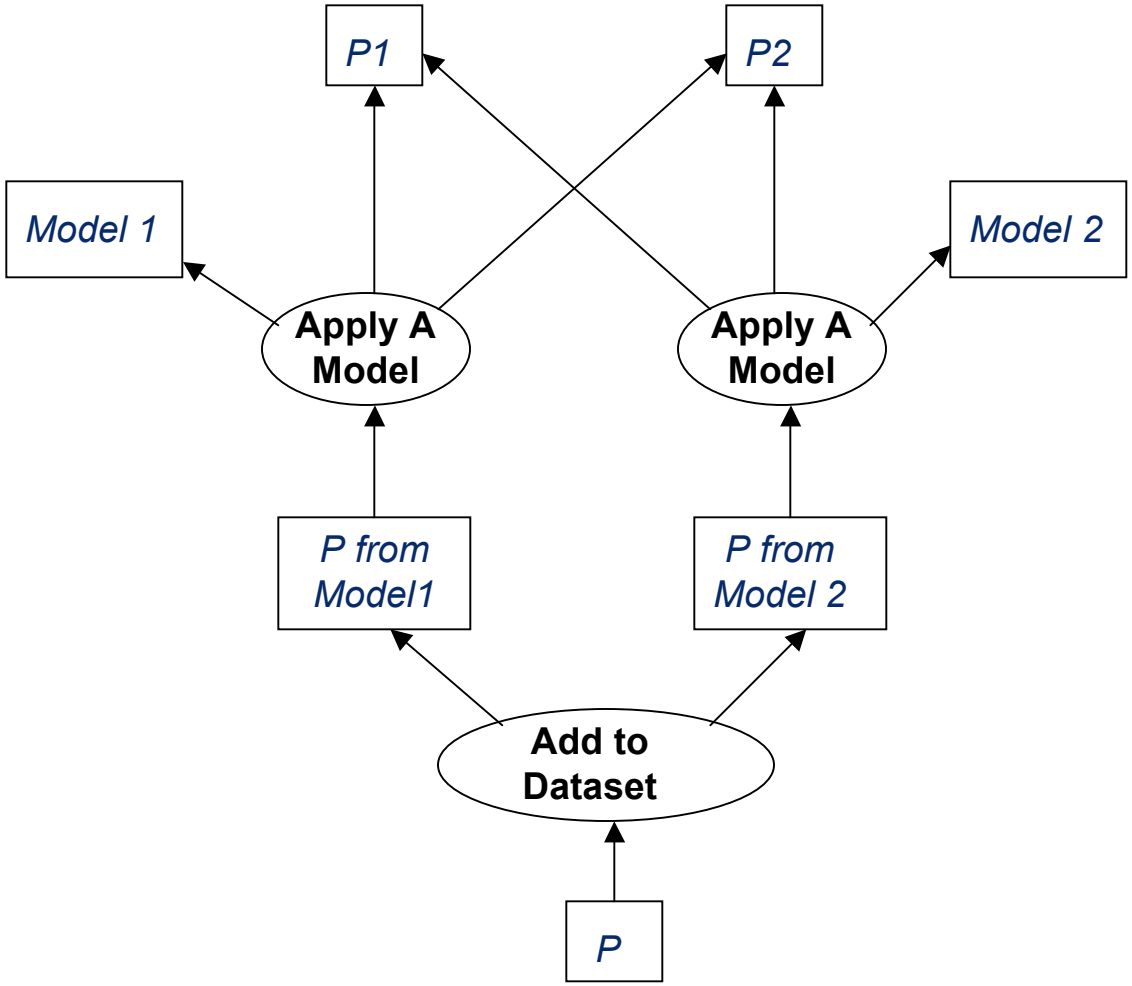
DDG if both sensors are read successfully



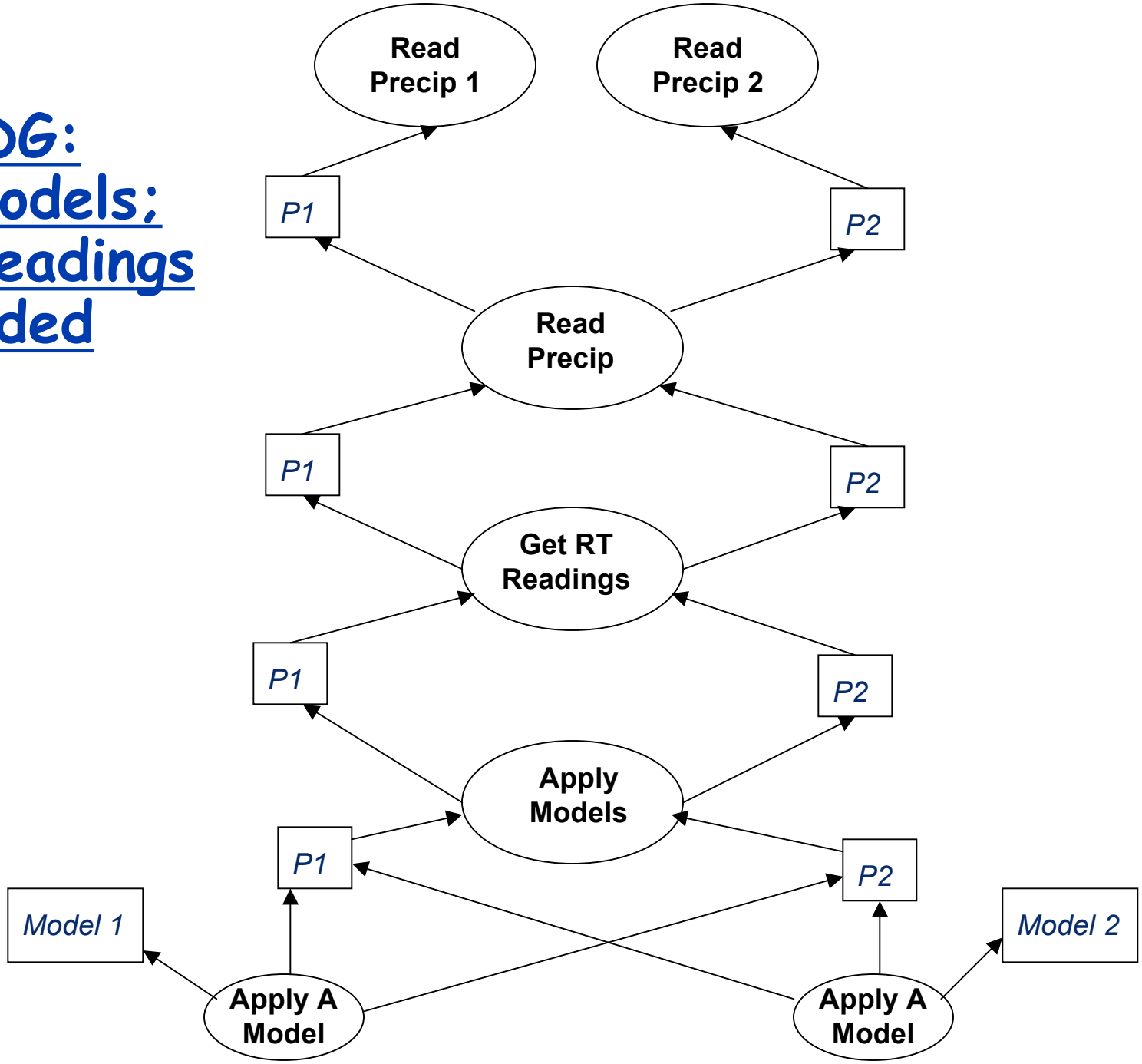
DDG if Read of Precip 1 Fails twice, then gets value from alternative source (airport)

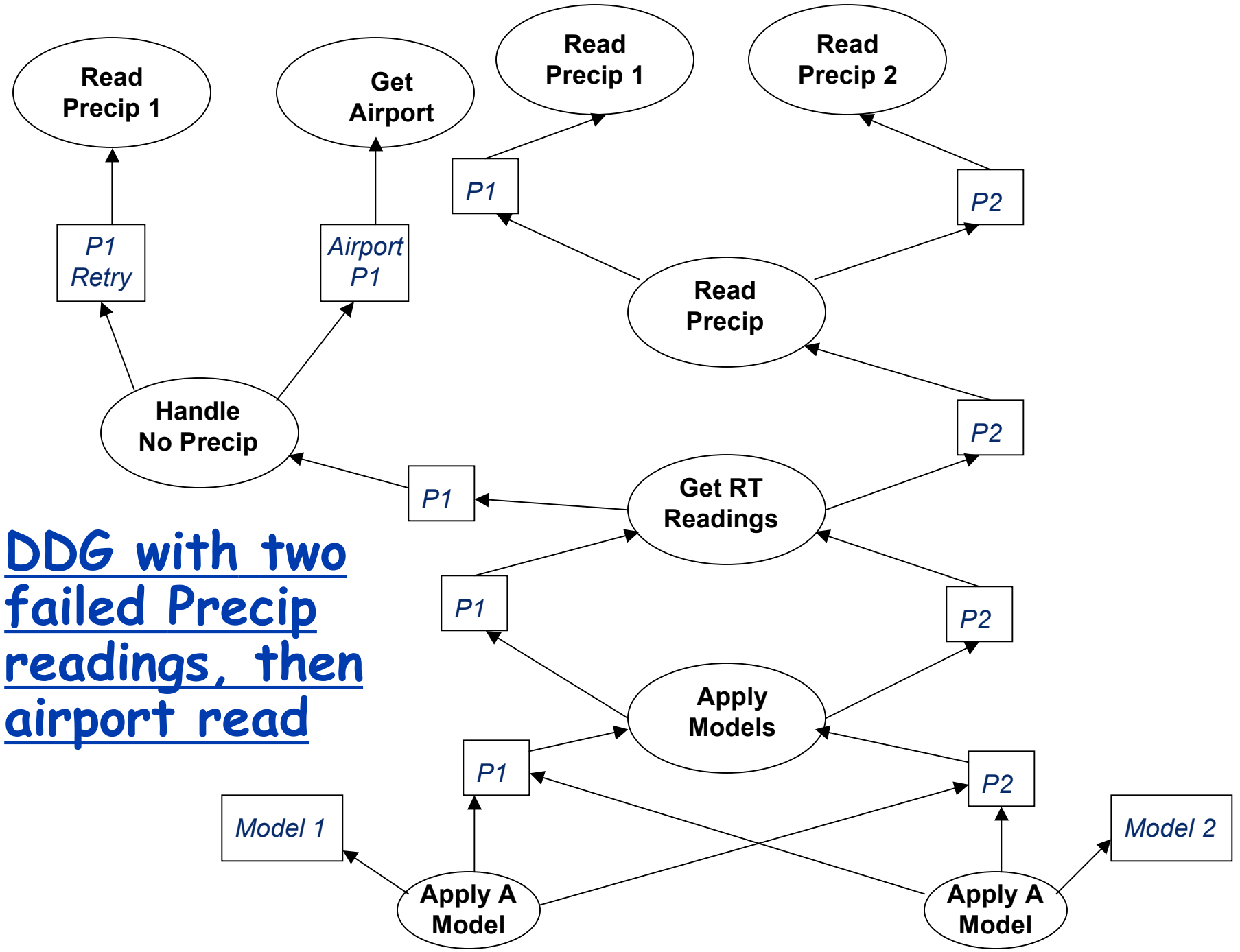


DDG showing consideration of two alternative models

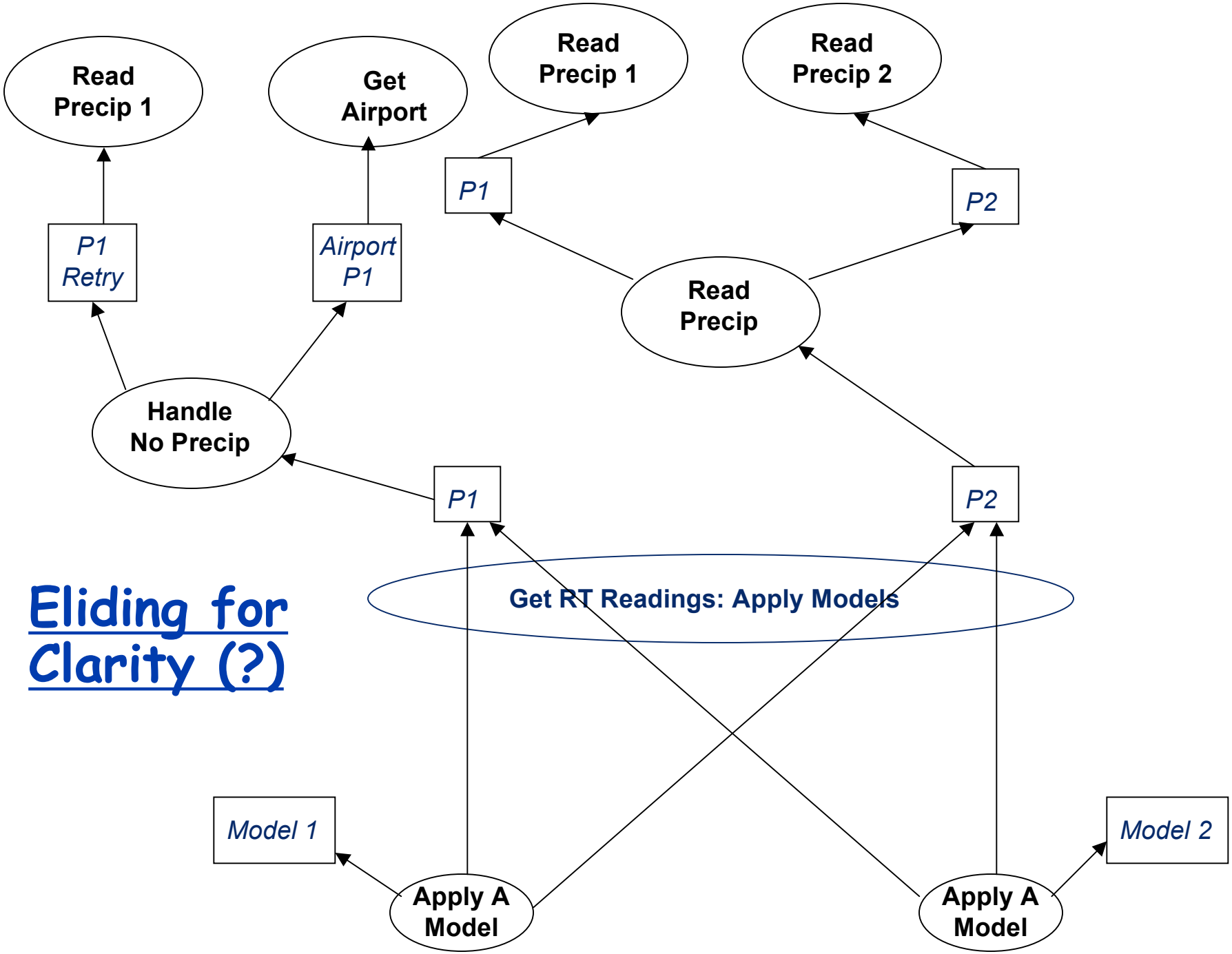


Full DDG: Two Models; Both readings succeeded

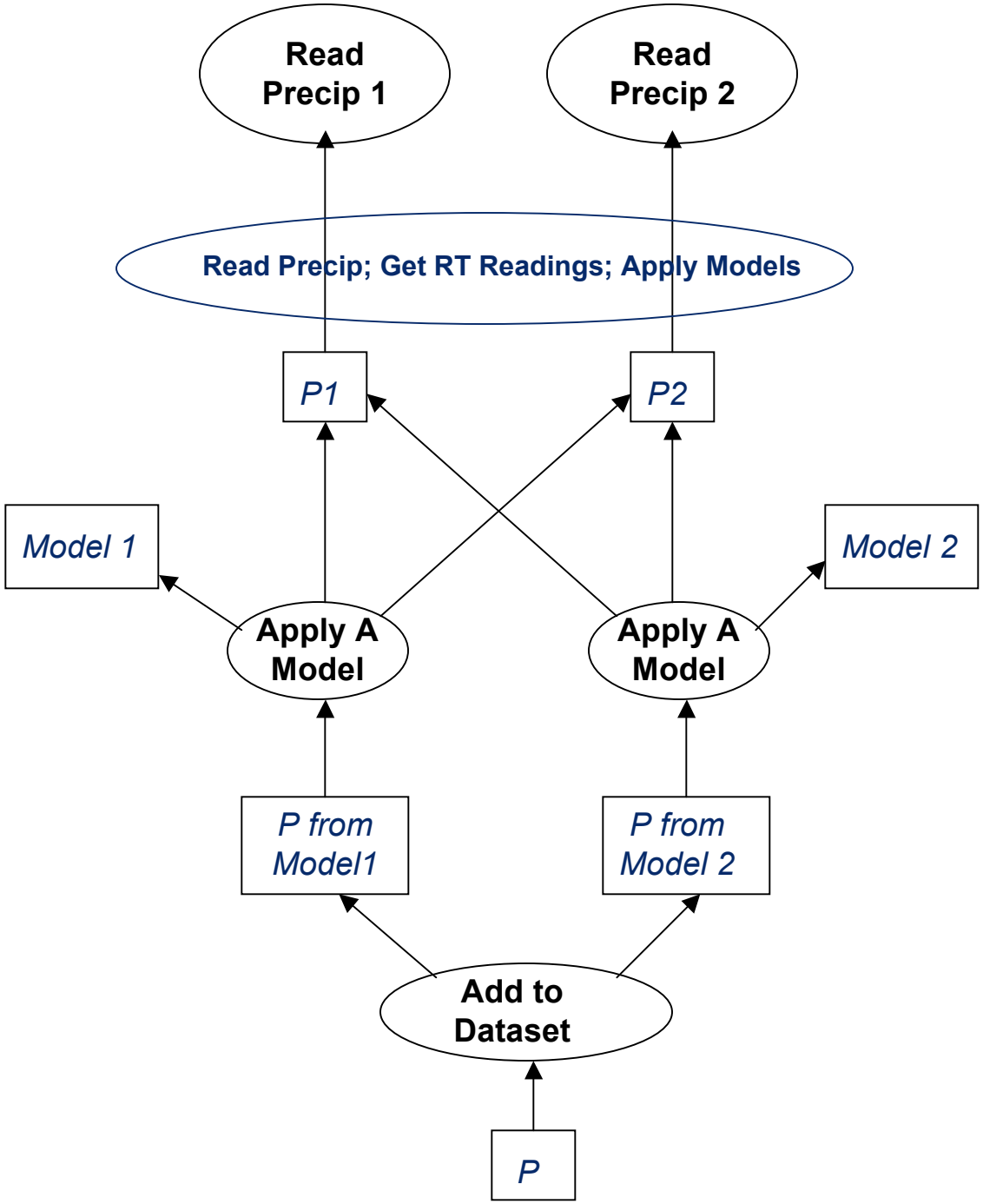




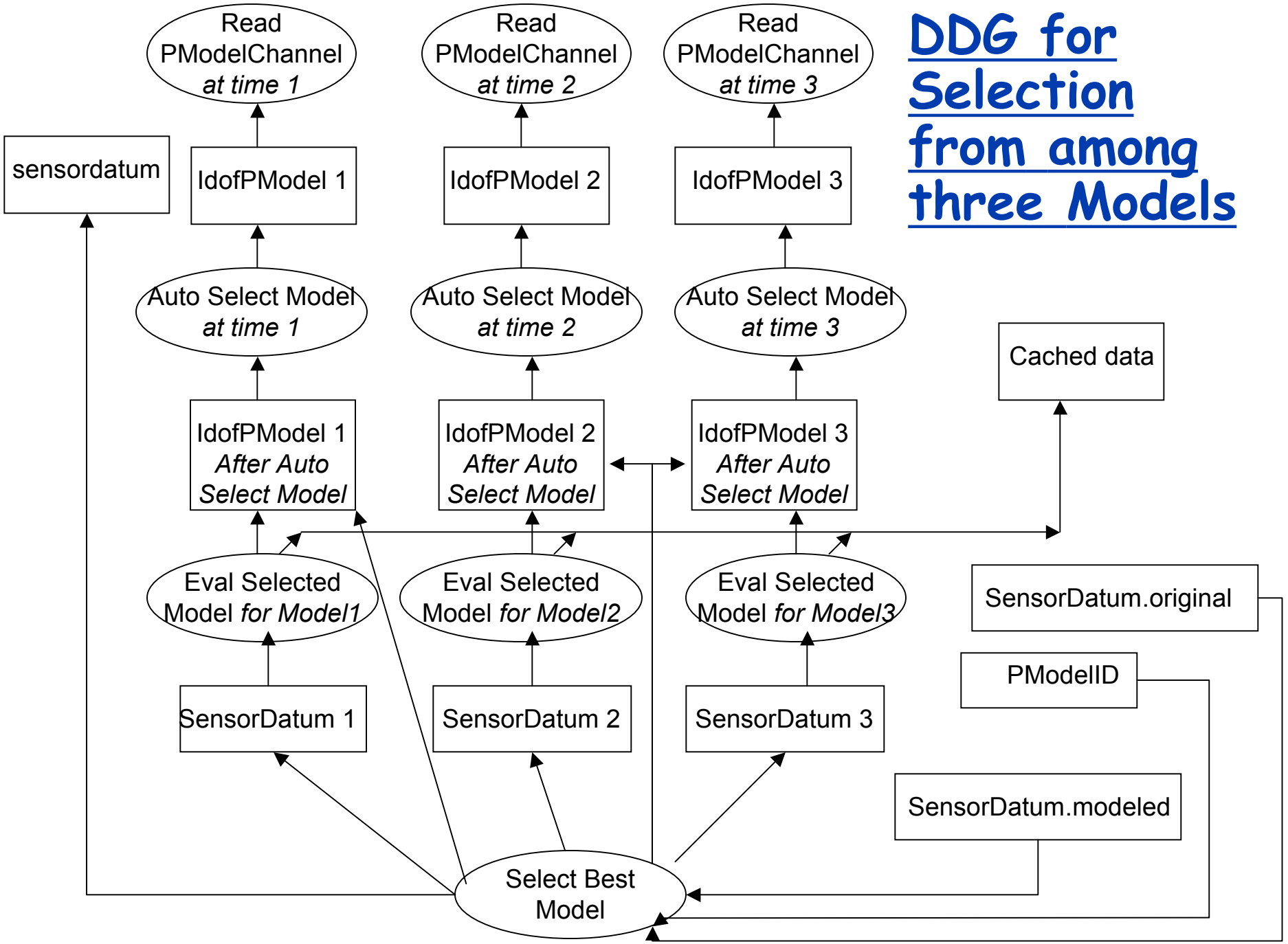
DDG with two failed Precip readings, then airport read

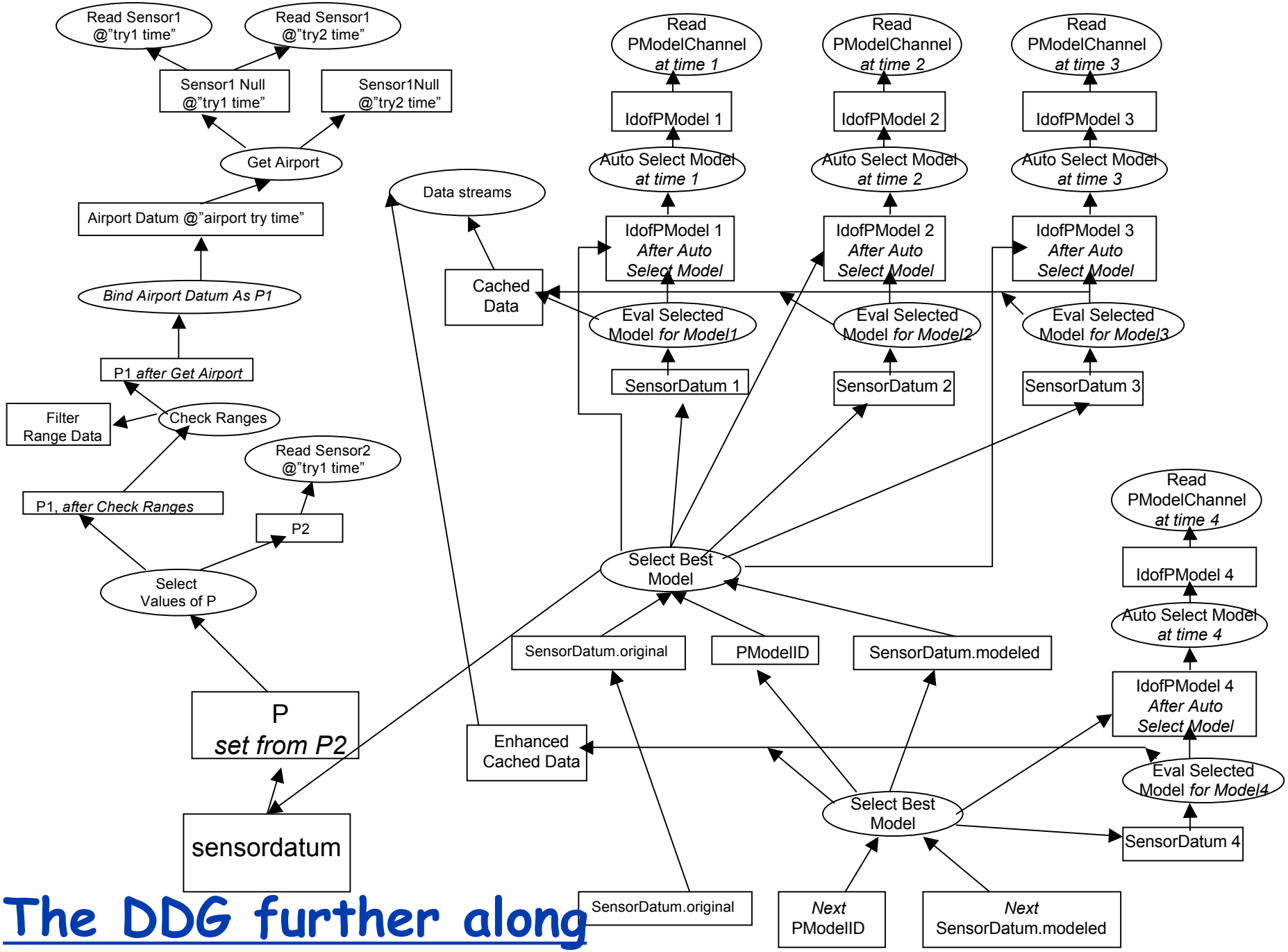

Read Precip 1
Get Airport
Read Precip 1
Read Precip 2
P1
P2
P1 Retry
Airport P1
Read Precip
Handle No Precip
P1
P2
Get RT Readings: Apply Models
Eliding for Clarity (?)
Model 1
Model 2
Apply A Model
Apply A Model

A Further DDG



DDG for Selection from among three Models





The DDG further along

Process Provenance Metadata

- Derived from the DDG
  - Root of the appropriate DDG subDAG
  - Efficiencies by storing one DDG
    - Provenance metadata stored as pointer
- Tradeoffs possible
  - Cache some intermediate values
  - Rederive the rest

Some Related Work in Scientific Workflow

- The Kepler Project
  - Largest, best funded
  - Based on Ptolemy II Data Flow Graph Approach
  - Oodles of tools
- Taverna, Teuta, Chimera, JOpera
  - Different Data Flow Graph projects
- Provenance Projects
  - Database approaches
  - Process history logging approaches

Problems with Current Scientific Workflow

- Languages tend to be semantically weak
  - Based on "box and arrow" charts
- Principal semantic weaknesses
  - Lack of abstraction facilities
  - Weak exception management
  - Non-trivial iteration can be hard to understand
- Other problems (with some)
  - Lack of focus on data and data flow
  - Provenance viewed as a separate issue
  - Poorly defined semantics
    - Restricts the rigor of analyses
- Kepler is better than most

Kepler

- Large toolset to support users
- Highly visual
- Box and arrow charts are defined by Directors, which provide semantics
  - But different levels of hierarchy may have different Directors
  - No standard usage of Directors
    - E.g. for loops and exceptions
  - Hard to tell what a diagram is saying without looking at
    - Its Directors
    - Director of its parent
- Programming language experience can inform work in this area

Important Clarification

- DFG-based scientific workflow systems can do (most or) all of the above
- BUT: Process definitions tend to be ungainly, opaque
- The "Turing Tarpit" for process definition
- Focus here:
  - What language constructs should be used to make these processes clear, evolvable, analyzable, etc.
  - Software engineering has much to say about this

Observations

- Creating a PDG makes scientists think carefully
- The PDG helps scientists create flexible and scalable processes
- The PDG supports re-execution using different input data and/or different tools.
- The PDG provides a useful and efficient framework for structuring and generating the DDG.
- PDGs can be rigorously evaluated for logical, statistical and propagation of measurement errors.

Summary

- Defined PDGs for ecological research processes
  - Carbon Flux
  - Water Budget
- Key semantic features needed
  - Exception management
  - Abstraction/modularity/reuse
  - Agent definition
- Not favorable for use of DFGs as PDGs
- Little-JIL interpreter executed simple processes only
- DDGs done only manually
  - Automation needed
  - Application to other process areas and problems?

Future work

- SciWalker 2
  - Environment for building and maintaining PDGs and DDGs
- Focus on DDGs
  - How to build, maintain, optimize, display them
  - What else are they good for?
    - (Software) rework?
    - Running benchmarks?
    - ???
- Integrate with Kepler
  - Use Kepler for low-level tasks
  - Little-JIL for high-level process definition

Questions and Discussion?