Performance Evaluation of Knowledge-based Kitting via Simulation

Thomas Kramer[1], Zeid Kootbally[2], Stephen Balakirsky[3], Craig Schlenoff[4], Anthony Pietromartire[4], and Satyandra Gupta[2]

Presented by Zeid Kootbally

1: Department of Mechanical Engineering, Catholic University of America, Washington, DC, USA
2: Department of Mechanical Engineering, University of Maryland, College Park, MD, USA
3: Georgia Tech Research Institute, Atlanta, GA
4: Intelligent Systems Division, National Institute of Standards and Technology, Gaithersburg, MD, USA

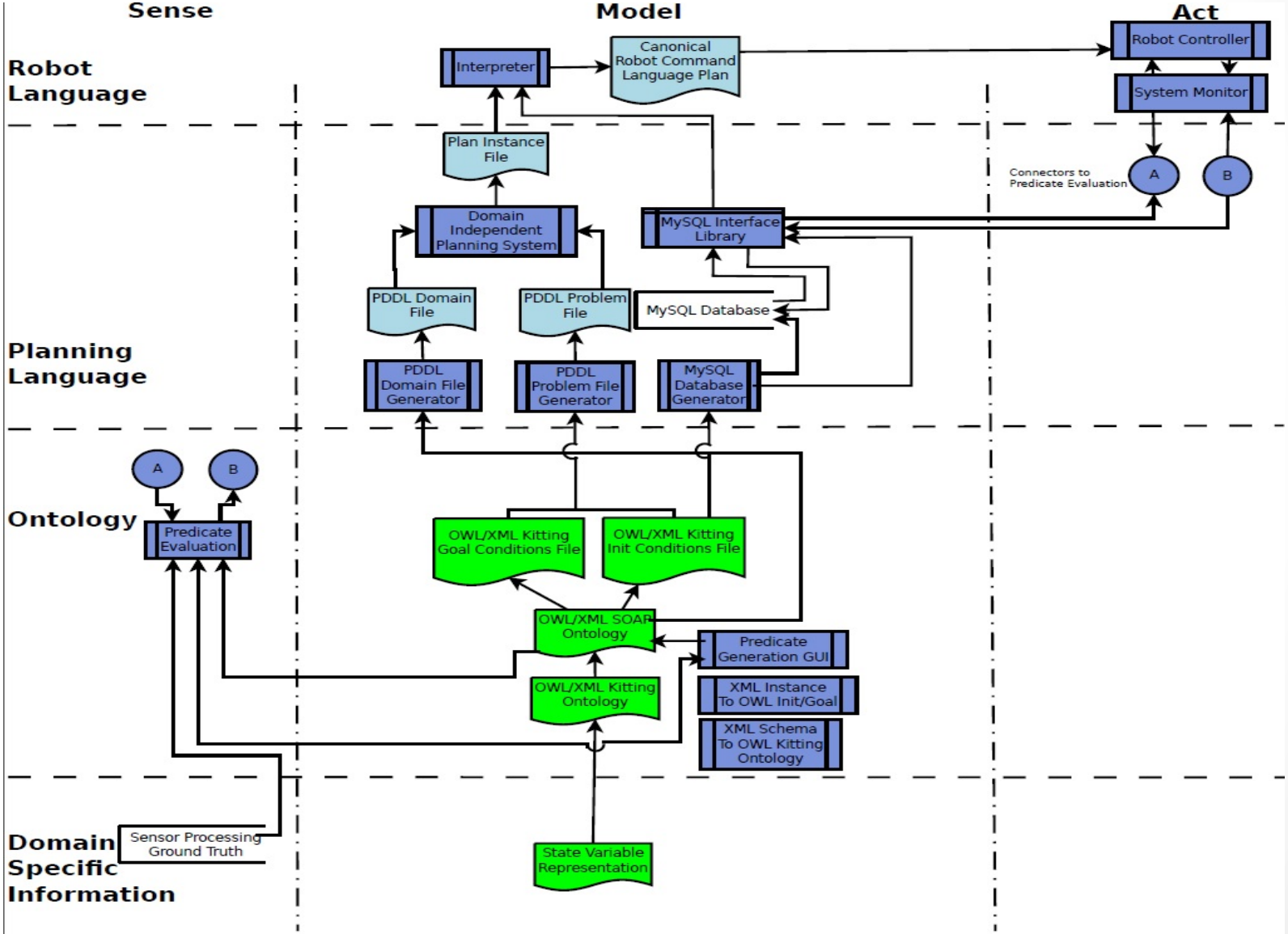
Kitting (Kit Building)

- Process in which individually separate but related items are grouped, packaged, and supplied together as one unit
- Utilized in many manufacturing assembly lines

Kitting (Kit Building)

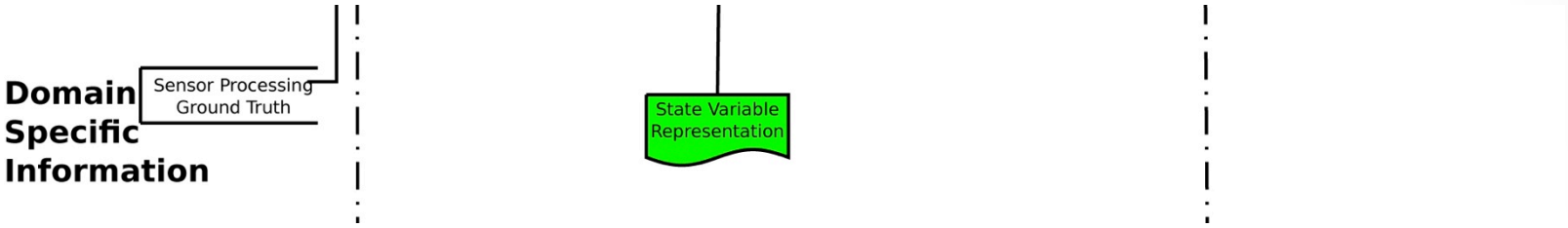
- Process in which individually separate but related items are grouped, packaged, and supplied together as one unit
- Utilized in many manufacturing assembly lines
- Robotic solutions require
  - Flexibility - Many parts with varying characteristics, part-to-part inconsistences
  - Agility - Changes in part supply locations, lack of fixturing
  - Rapid re-tasking – Ability to quickly build new varieties of kits

Knowledge Driven Methodology



Knowledge Driven Methodology

- Domain Specific Information

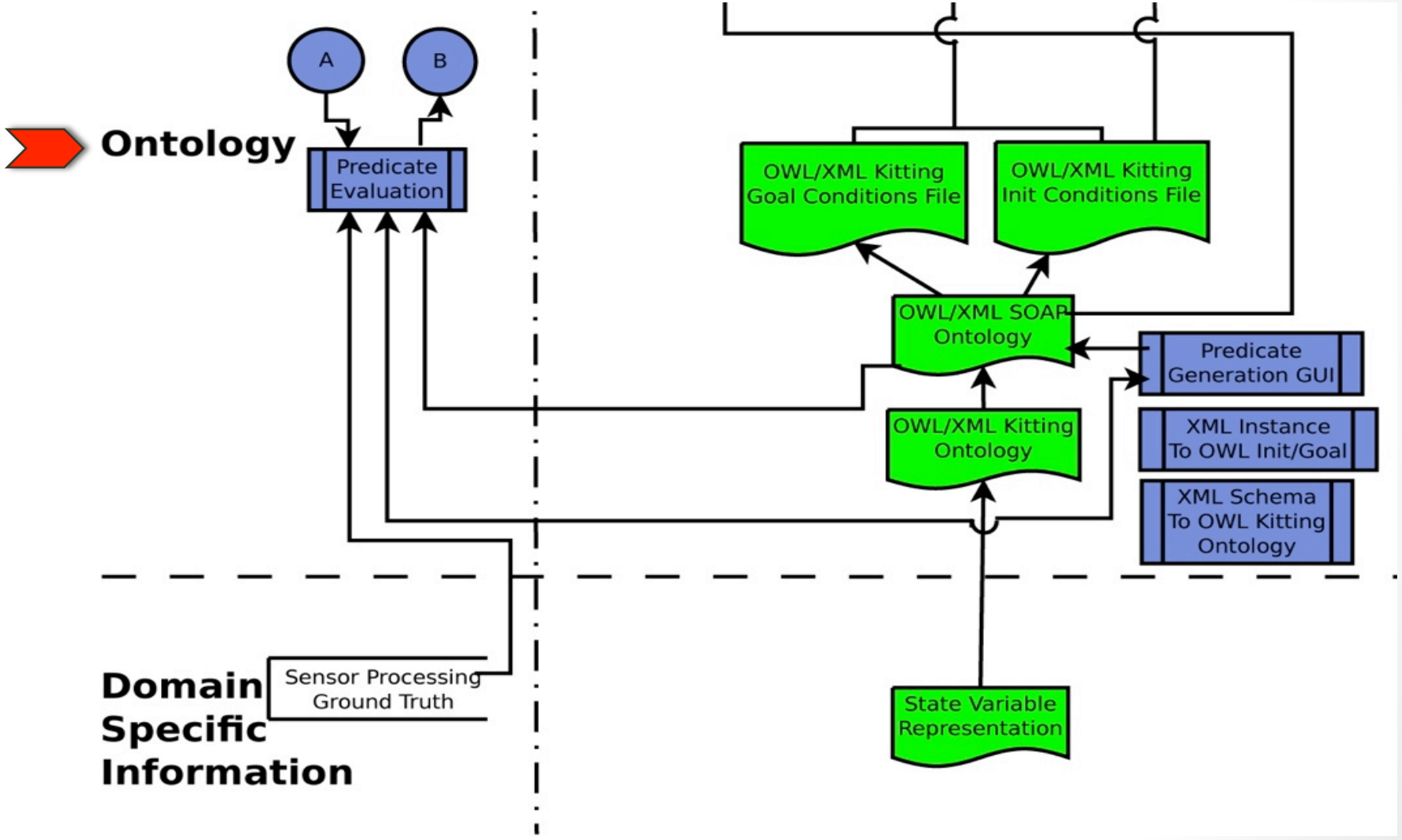


- Captures operational knowledge that is necessary to be successful in the particular domain in which the system is designed to operate
  - Information on items ranging from
    - From what actions and attributes are relevant,
    - To what the necessary conditions are for an action to occur and what the likely results of the action are
  - Formalism known as a state variable representation [1]

[1] D. Nau, M. Ghallab, and P. Traverso, *Automated Planning: Theory & Practice.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

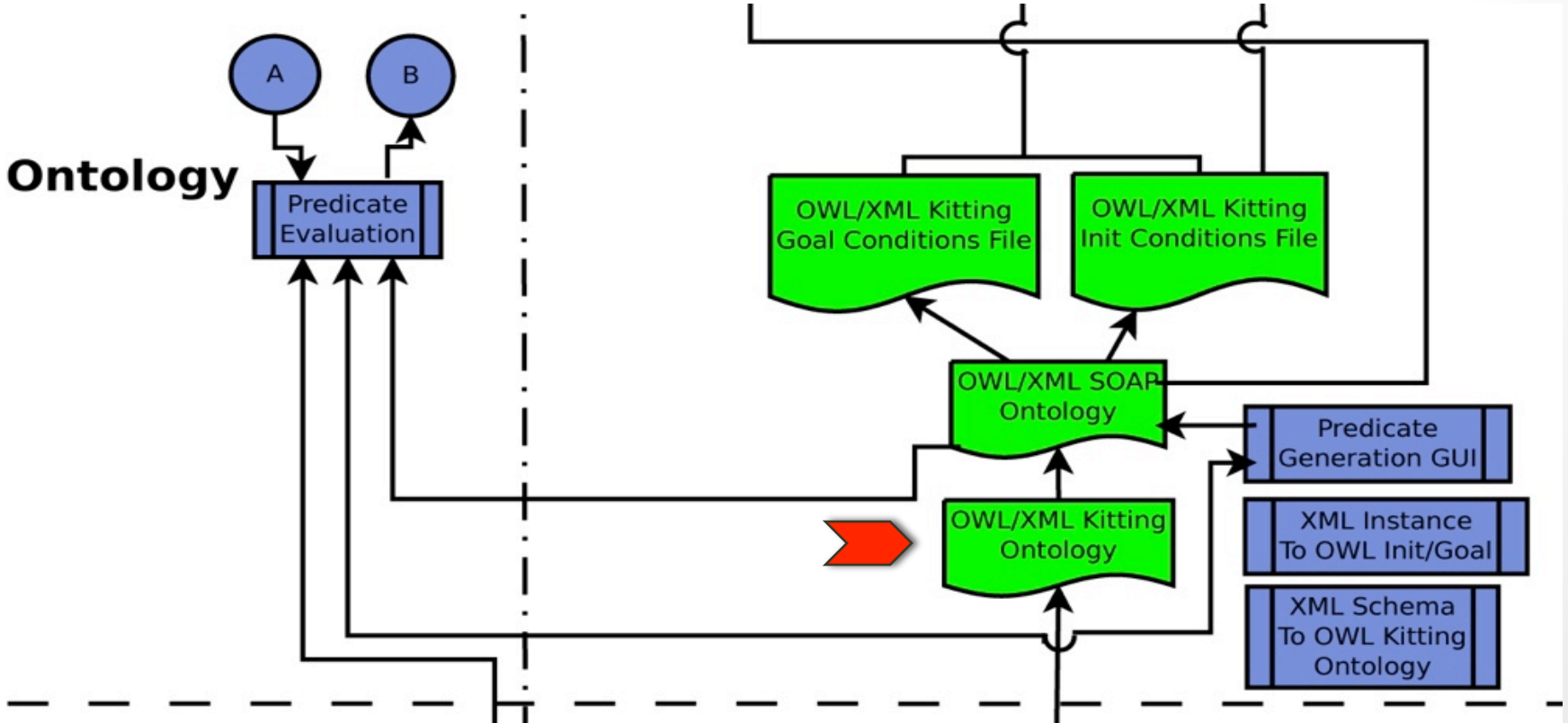
Knowledge Driven Methodology

- Ontology



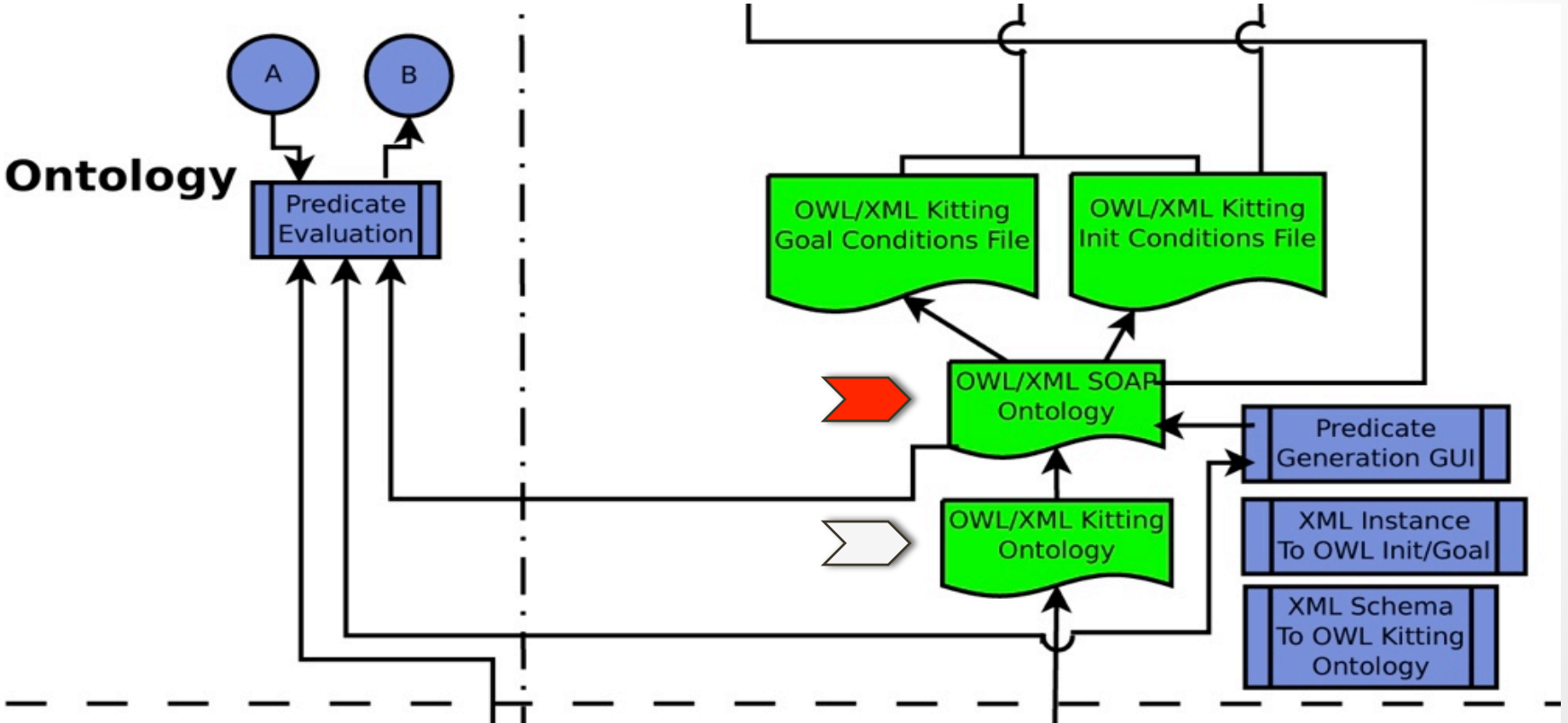
Knowledge Driven Methodology

- Ontology
  - A base ontology (OWL/XML Kitting) contains all of the basic information that was determined to be needed during the evaluation of the use cases and scenarios



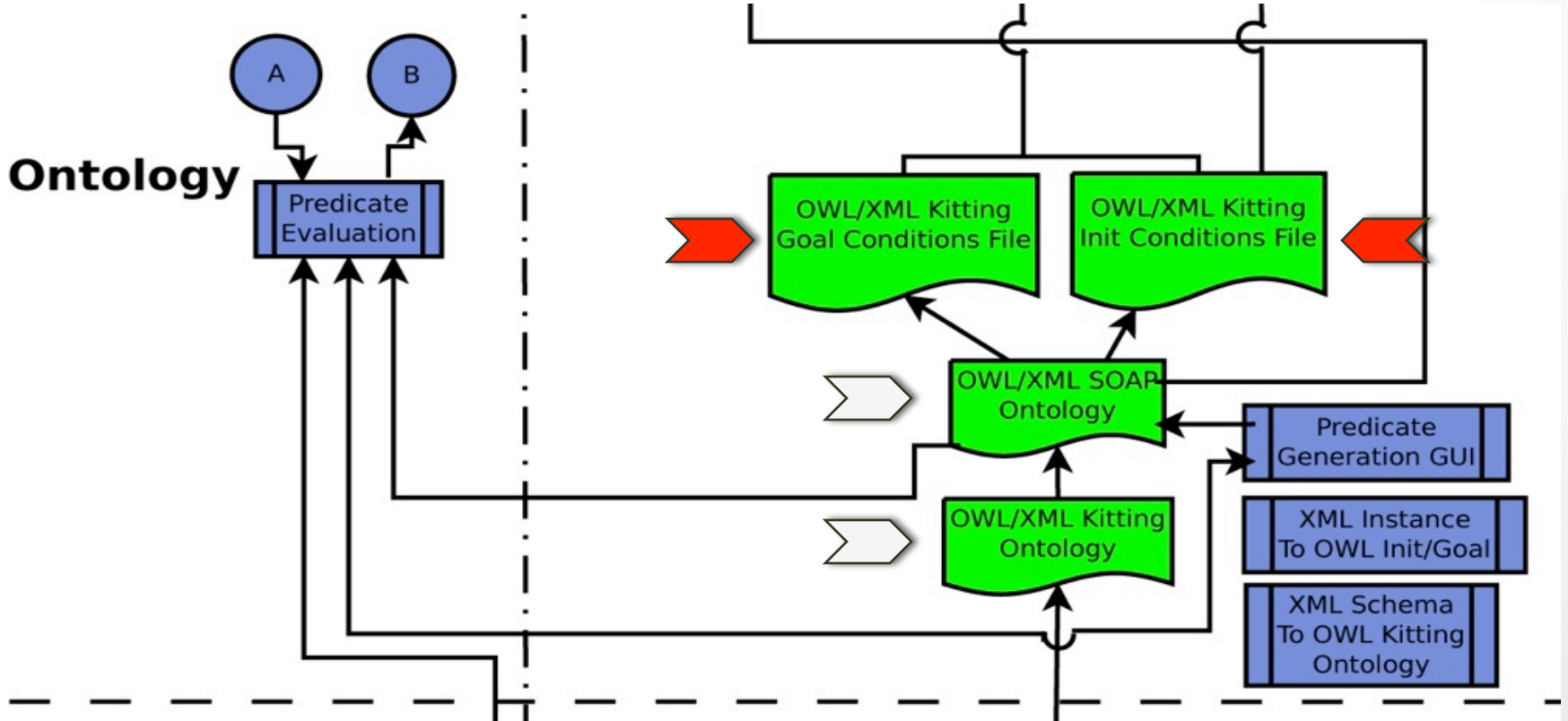
Knowledge Driven Methodology

- Ontology
  - The OWL/XML SOAP ontology describes aspects of individual actions and predicates that are necessary for the domain under study



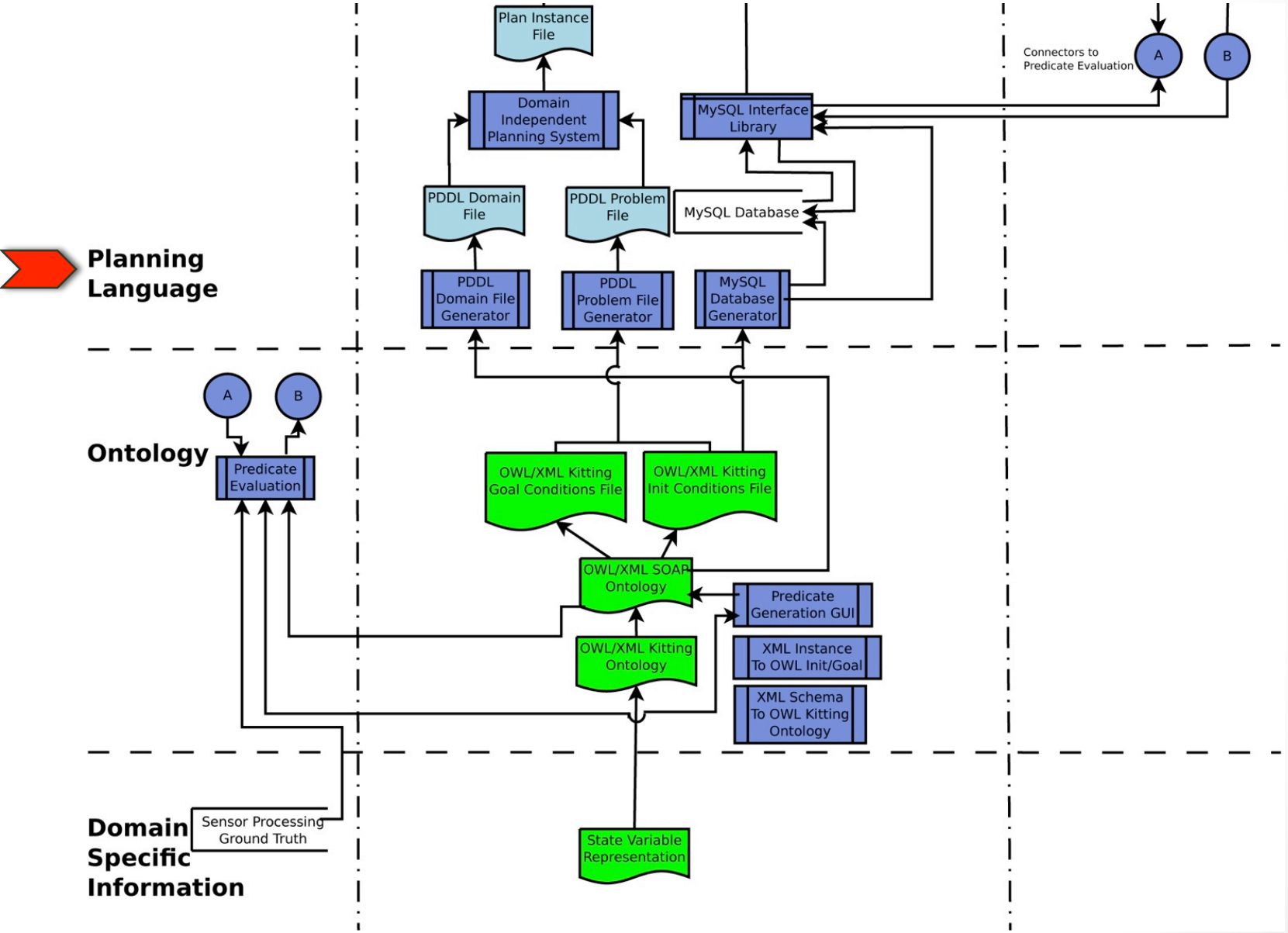
Knowledge Driven Methodology

- Ontology
  - The instance files describe the initial and goal states for the system
    - Initial state file must contain a description of the environment
    - Goal state file only needs to contain information that is relevant to the end goal of the system



Knowledge Driven Methodology

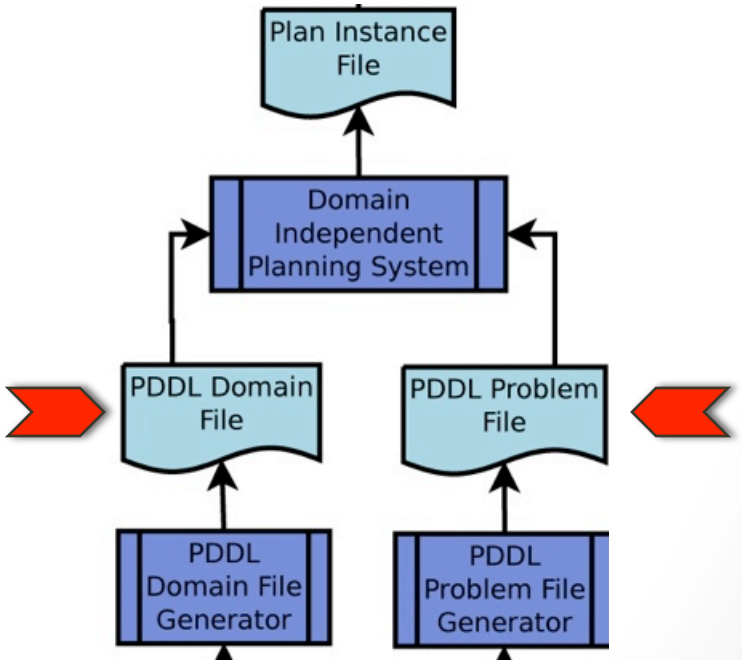
- Planning Language



Knowledge Driven Methodology

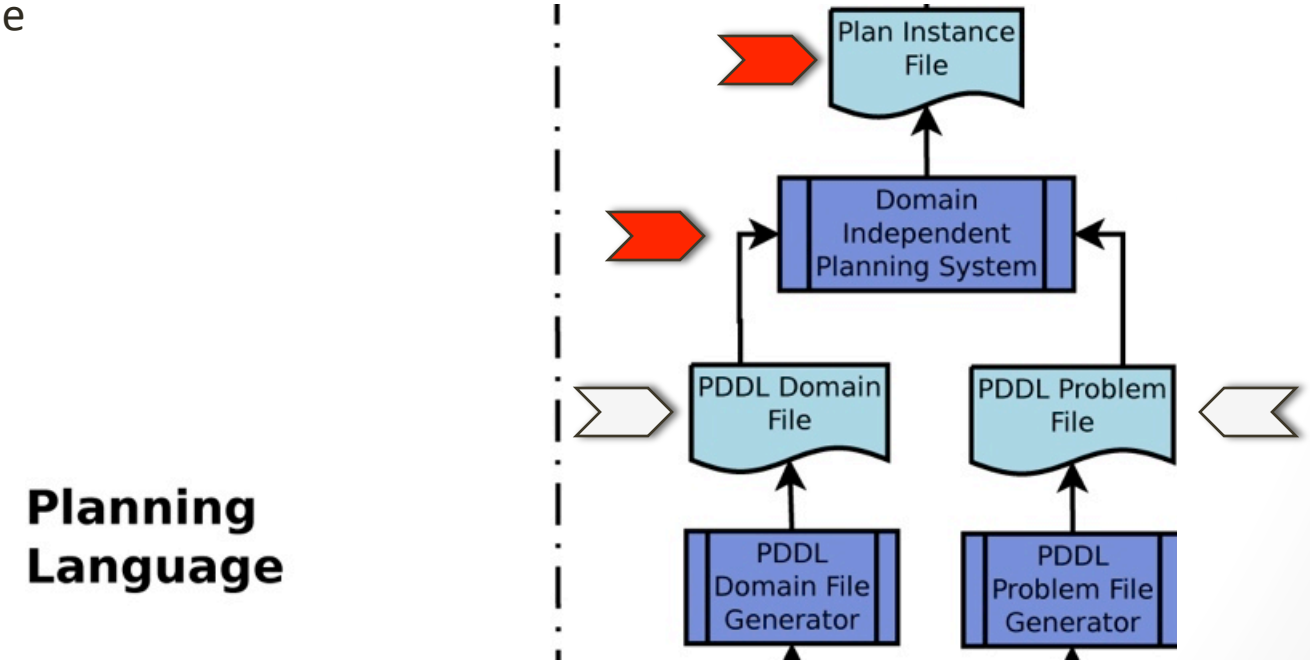
- Planning Language
  - Aspects of this knowledge are automatically extracted and encoded in a form that is optimized for a planning system to utilize (the Planning Language)
    - Planning language used in the knowledge driven system is expressed with the Planning Domain Definition Language (PDDL) (version 3.0)
    - The PDDL input format consists of two files that specify the domain and the problem
    -

Planning Language



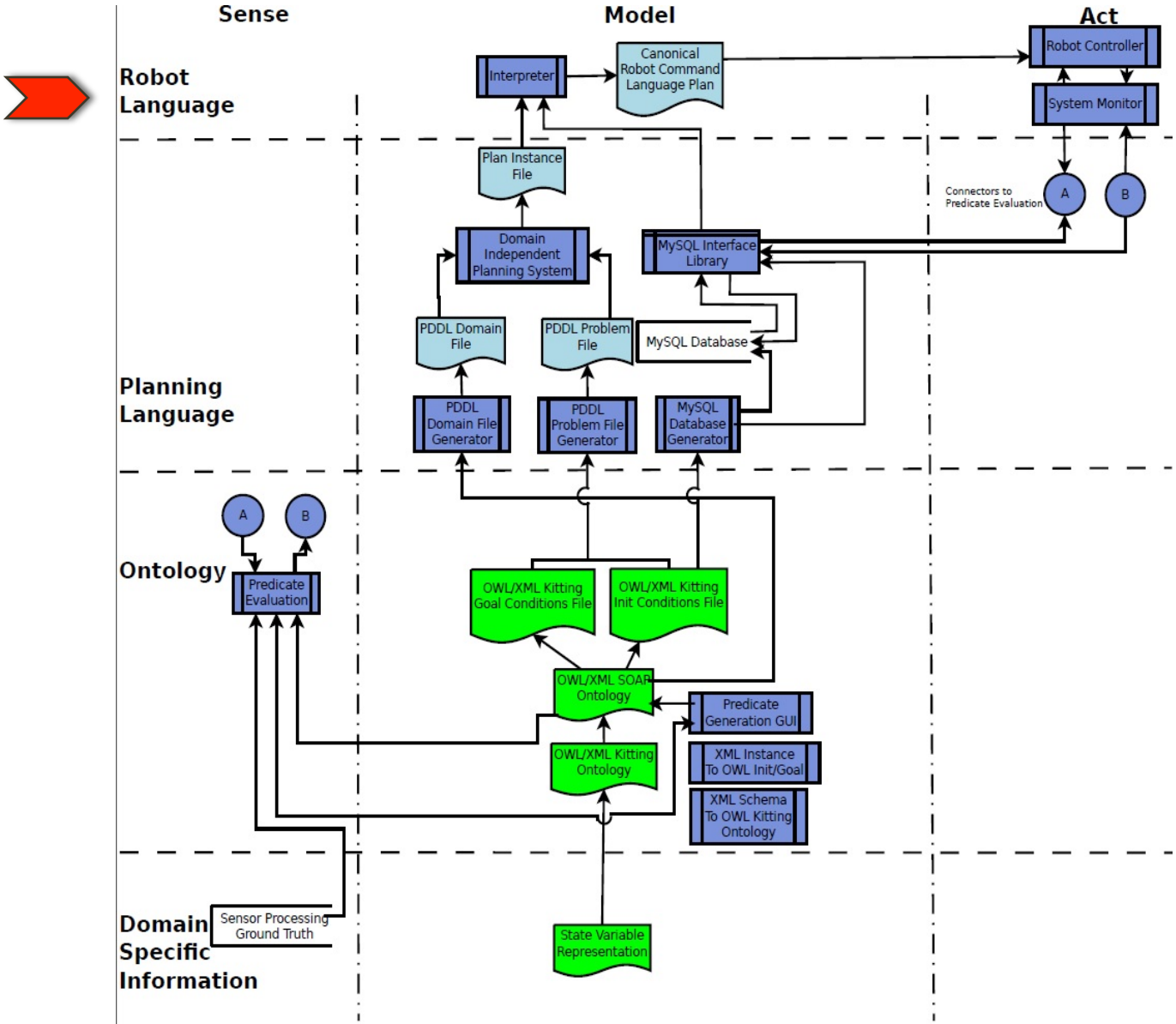
Knowledge Driven Methodology

- Planning Language
  - Aspects of this knowledge are automatically extracted and encoded in a form that is optimized for a planning system to utilize (the Planning Language)
    - Planning language used in the knowledge driven system is expressed with the Planning Domain Definition Language (PDDL) (version 3.0)
    - The PDDL input format consists of two files that specify the domain and the problem
    - Domain independent planning system used to produce a static Plan Instance File



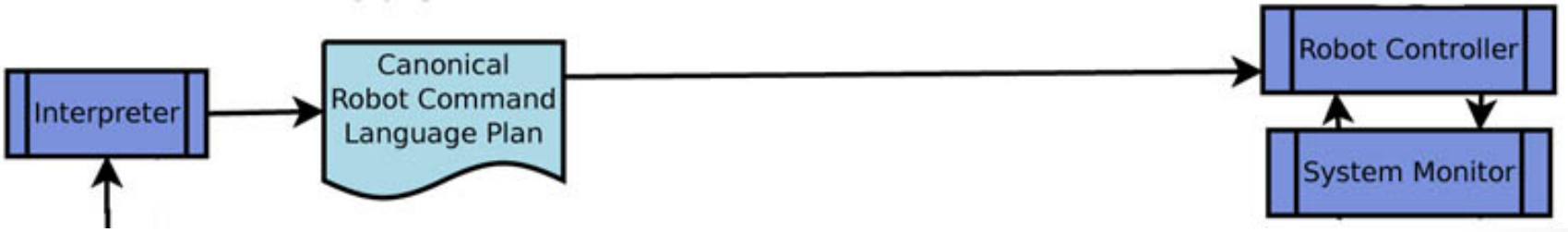
Knowledge Driven Methodology

- Robot Language



Knowledge Driven Methodology

- Robot Language



- The interpreter combines knowledge from the plan to form a set of sequential actions that the robot controller is able to execute
  - Set written in a canonical robot command language (CRCL)
  - CRCL provides generic commands that implement the functionality of typical industrial robots without being specific to
    - The language of the planning system that makes a plan
    - The language used by a robot controller that executes a plan

Knowledge Driven Methodology

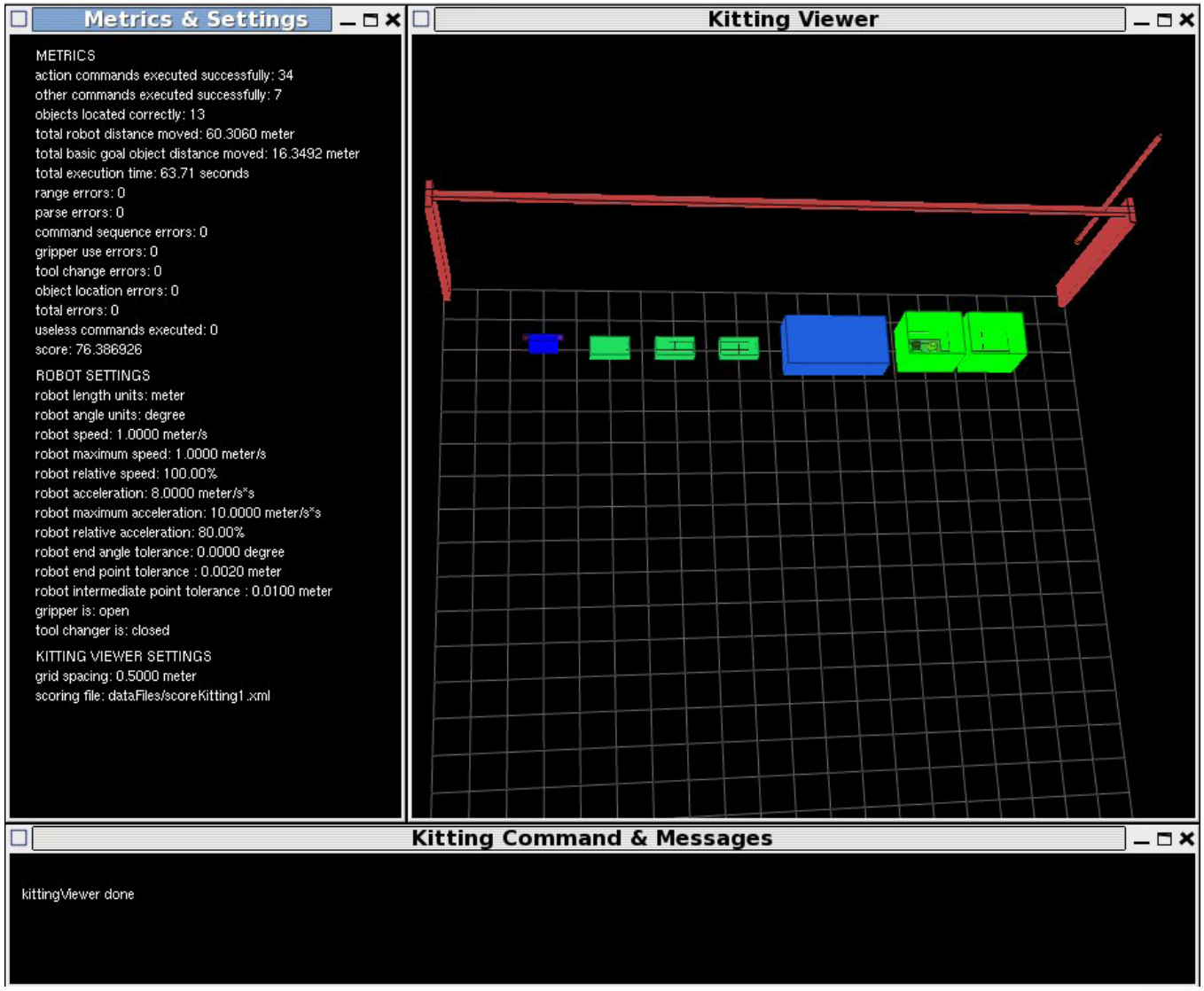
- Robot Language

- Example of a CRCL file:
  - InitCanon()
  - SetLengthUnits("meter")
  - SetAbsoluteSpeed(1.0)
  - SetRelativeAcceleration(80.0)
  - SetEndPointTolerance(0.002)
  - SetIntermediatePointTolerance(0.01)
  - OpenGripper()
  - MoveThroughTo({ {{1.5,1,1}, {0,0,-1}, {1,0,0}},{{1.5,1,0.0001}, {0,0,-1}, {1,0,0}} }, 2)
  - CloseGripper()
  - EndCanon(2)

Kitting Viewer

- Simulates the execution of a plan (CRCL command file) for changing a kitting workstation from an initial state to a goal state
  - Initial and goal states correspond to XML kitting schema
- Built in C++ using OpenGL graphics
- User controllable animated 3D view (pan, rotate, zoom)
- Uses three windows: graphics, metrics, command & messages

Kitting Viewer



Kitting Viewer

- Simulates the execution of a plan (CRCL command file) for changing a kitting workstation from an initial state to a goal state
  - Initial and goal states correspond to XML kitting schema
- Built in C++ using OpenGL graphics
- User controllable animated 3D view (pan, rotate, zoom)
- Uses three windows: graphics, metrics, command & messages
- Two phases:
  - (1) Execute commands
    - Each push of ‘e’ key executes another command
  - (2) Check final location of movable objects against goal state
    - Each push of ‘e’ key checks another object
- calculates execution metrics and a total score

Test Method

- The Kitting Viewer embodies a test method for comparing the performance of different kitting planning systems
- It may be used to compare different plans for building kits
  - Each plan to be compared has the same initial and goal states

Metrics – Execution Phase

- Action Commands Executed Successfully
  - Number of action commands that have been executed successfully so far
- Other Commands Executed Successfully
  - Number of commands that are not action commands that have been executed successfully so far – mostly setting commands
- Total Robot Distance Moved (from command points)
  - Total distance that the tool tip has moved so far
- Total Execution Time (from speed and distance, not clock)
- Range Errors (e.g., percentage > 100)
  - Number of times a command tries to set a parameter to a value that is out of the allowed range of the parameter
- Parse Errors
  - Number of lines in the CRCL command file that cause an error in the command file parser
- Gripper Use Errors (open or close when robot has no gripper)
- Tool Change Errors (must be at tool changer)
- Total Errors
- Useless commands (e.g., open gripper when already open)

Metrics – Check Phase

- Objects Located Correctly
  - Number of movable goal objects checked so far that were moved from their initial location to the correct goal location
- Object Location Errors
  - Number of movable goal objects checked so far whose final position is not the one given in the goal file
- Total Basic Goal Object Distance Moved
  - Total net distance moved from initial location to final location by movable basic goal objects checked so far
- Total Errors
  - Number of Total Errors from the first phase plus the number of Object Location Errors for movable goal objects checked so far
- Score (next slide)

Scoring

- Scoring is configurable using a scoring file to specify weights and (optionally) valuation functions for 5 factors
- Five factors are:
  - Right Stuff: a measure of achieving goal positions
  - Command Execution: fraction of commands executed correctly
  - Distance: a measure of robot efficiency in terms of distance
  - Time: a measure of robot efficiency in terms of time
  - Useless Commands: from 1st phase
- Scoring file is XML instance file corresponding to XML schema for scoring kitting

Conclusion

- Metrics for industrial kit building have been developed based on a knowledge model
- A configurable method of computing a net score from the metrics has been developed
- A Kitting Viewer has been developed that simulates the execution of a plan in the form of a robot command file intended to take a kitting workstation from an initial state to a goal state
- The Kitting Viewer computes the metrics and a final score for the plan

Future Work

- Allow identical parts in the goal file to be interchangeable
  - If parts A and B are identical, and the goal file has part A in position 1 and part B in position 2, it will be equally valid to put part B in position 1 and part A in position 2
- Other desirable improvements
  - Using OWL instance files as input
  - Implementing animation of gripper rotation
  - Using other types of grippers
  - Implementing kinematics of commercial robots
    - Move towards ROS/USARSim framework

Scoring

- Additive Score

$$S_a = \frac{(V_1 \times W_1) + (V_2 \times W_2) + \ldots + (Vn \times Wn)}{W_1 + W_2 + \ldots + W_n}$$

V_i : Additive factor value

W_i : Non-negative weight for additive factor

- Total Score

$$S = (100 \times S_a \times U_1 \times U_2 \times \ldots \times U_m)$$

U_i : Multiplicative factor