

# Документация SCL-machine

# **Программный вариант реализации логической машины интерпретации логических sc-моделей компьютерных систем**

**:= [SCL-machine]**

**:= [ostis-inference]**

**⇒ декомпозиция программной системы\*:**

- База знаний *SCL-machine*
- Решатель задач *SCL-machine*
- Интерфейс *SCL-machine*

**⇒ реализованные логические связи\*:**

- импликация\*
- дизъюнкция\*
- конъюнкция\*
- отрицание\*

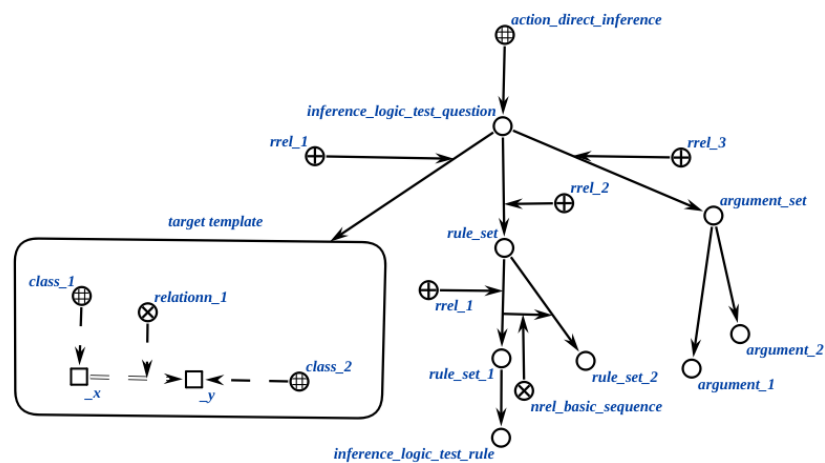
**⇒ не реализованные логические связи\*:**

- эквиваленция\*
- строгая дизъюнкция\*

## **Решатель задач *SCL-machine***

**⇒ обобщённая декомпозиция\*:**

- Агент прямого логического вывода
- ⇒ пример входной конструкции\*:**
- [



**⇒ параметры агента\*:**

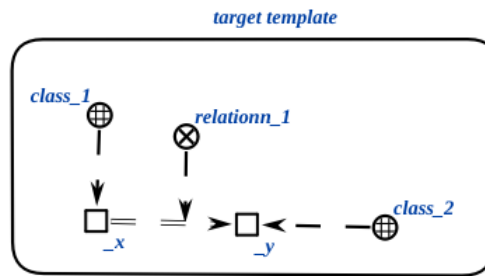
- шаблон цели
- := [target template]**

**⇒ пояснение\*:**

[Шаблон, успешный поиск которого показывает, что цель логического вывода достигнута и применение правил можно прекратить.]

**⇒ описание примера\*:**

[



- множество правил

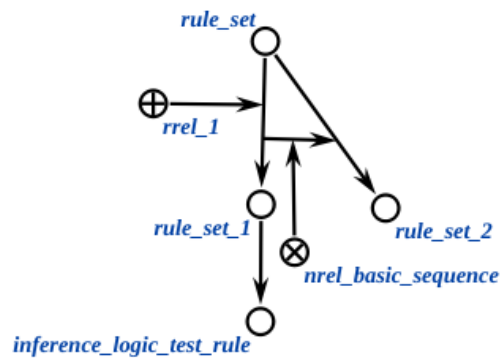
**:=** [rule set]

**⇒** пояснение\*:

[Ориентированное множество, первым элементом которого является множество правил, которые применяются в первую очередь, а каждое следующее множество правил применяется после предыдущего. Таким образом указываются приоритеты множеств правил.]

**⇒** описание примера\*:

[



- множество аргументов

**:=** [argument set]

**⇒** пояснение\*:

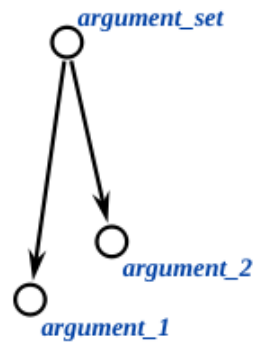
[Множество тех элементов, которые должны быть подставлены как значение переменных шаблона цели.]

**⇒** пояснение\*:

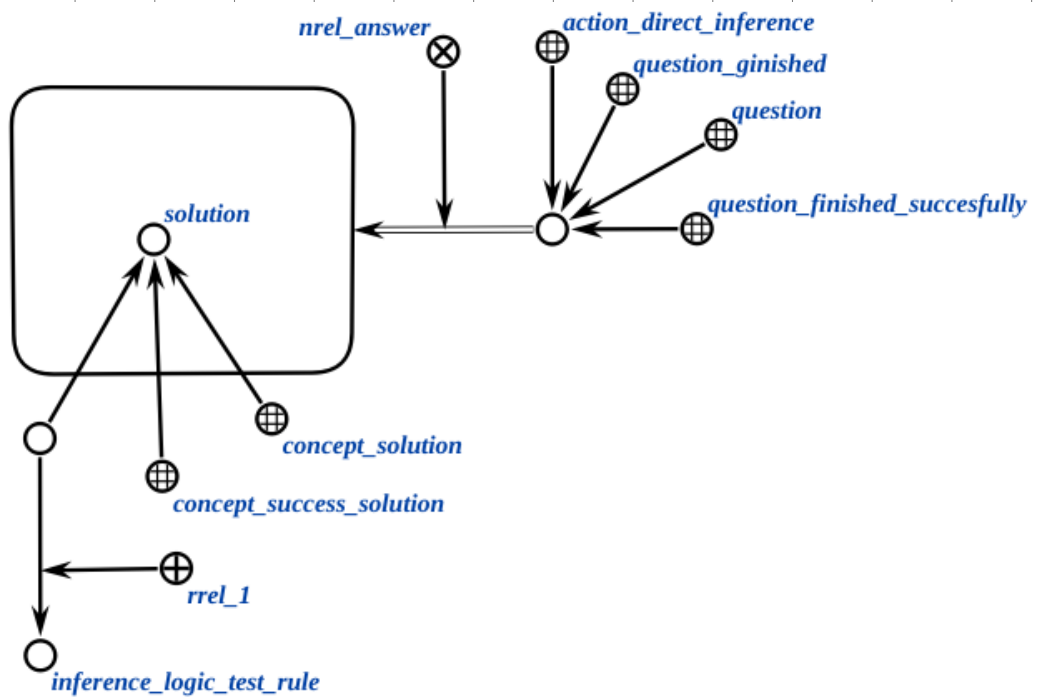
[В данном примере значением переменной  $\_x$  в шаблоне цели может быть sc-узел argument\_1, тогда значением  $\_y$  будет argument\_2 или наоборот: значением  $\_x$  sc-узел argument\_2, а значением  $\_y$  – argument\_1.]

**⇒** описание примера\*:

[



⇒ пример выходной конструкции\*:



⇒ обобщённый алгоритм\*:

- [Получение параметров агента, проверка их валидности. Вызов агента;]
  - [Проверка, достигнута ли уже цель в базе знаний;]
  - [Построение вектора очереди правил на основе множества правил. Цикл по всем правилам и по не достигнута цель;]
    - ⇒ циклические операции\*:
      - [Получение посылки логического правила;]
      - [Определение типа посылки (связка конъюнкции, дизъюнкции, отрицания или атомарная логическая формула);]
      - [Проверка истинности посылки в зависимости от её типа;]
      - [Генерация по шаблону следствия;]
      - [Добавление в дерево решений узла правила.]
  - [Возврат дерева применённых правил.]
- Агент обратного логического вывода
- ⇒ примечание\*:
- [Не реализовано.]