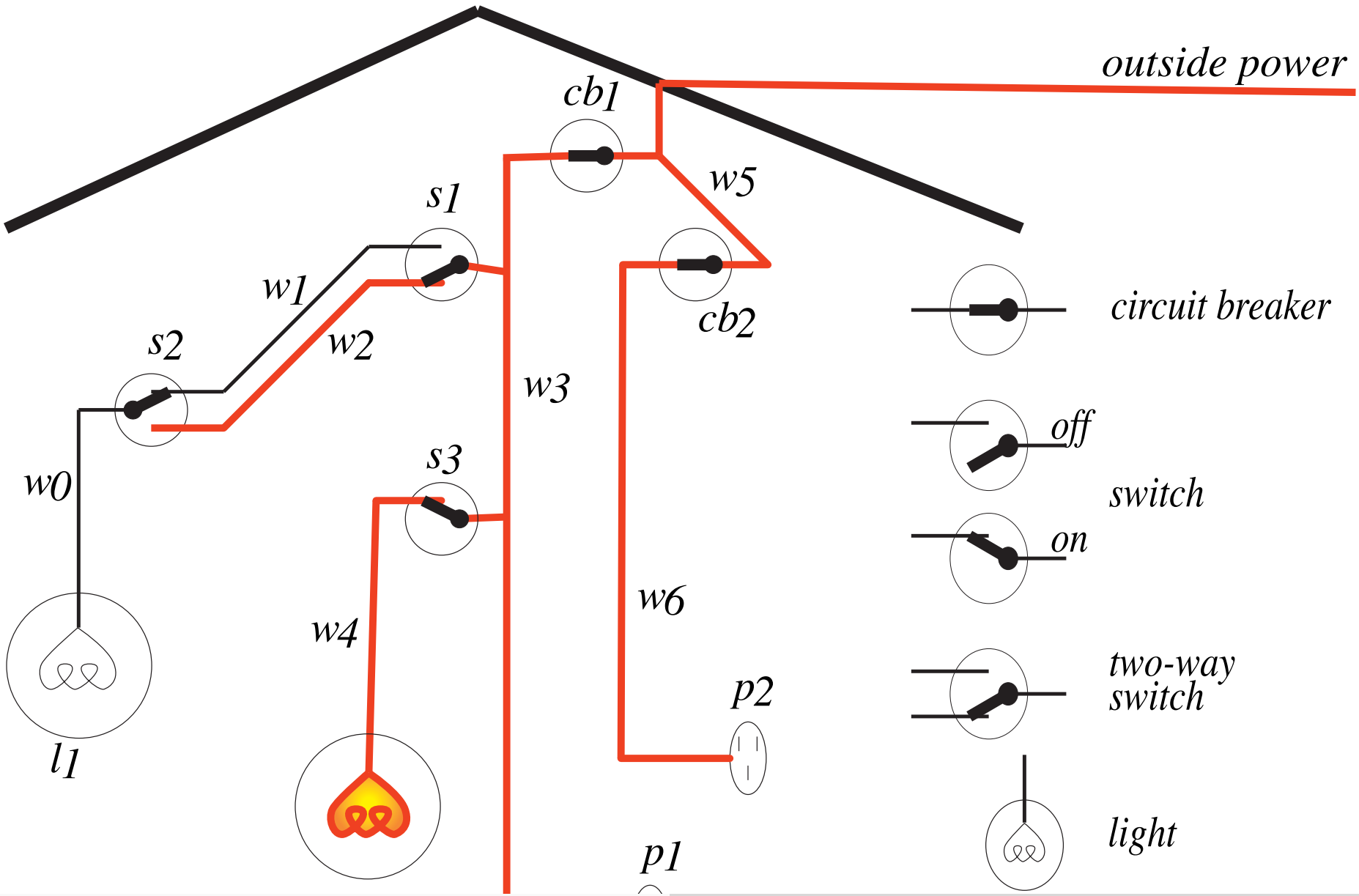
Electrical Domain



Users

- In the electrical domain, what should the house builder know?
- What should an occupant know?

Users

- In the electrical domain, what should the house builder know?
- What should an occupant know?
- Users can't be expected to volunteer knowledge:
  - They don't know what information is needed.
  - They don't know what vocabulary to use.

Ask-the-user

- Users can provide observations to the system. They can answer specific queries.
- Askable atoms are those that a user should be able to observe.
- There are 3 sorts of goals in the top-down proof procedure:
  - Goals for which the user isn't expected to know the answer.
  - Askable atoms that may be useful in the proof.
  - Askable atoms that the user has already provided information about.

Ask-the-user

- Users can provide observations to the system. They can answer specific queries.
- Askable atoms are those that a user should be able to observe.
- There are 3 sorts of goals in the top-down proof procedure:
  - Goals for which the user isn't expected to know the answer.
  - Askable atoms that may be useful in the proof.
  - Askable atoms that the user has already provided information about.
- The top-down proof procedure can be modified to ask users about askable atoms they have not already provided answers for.

Knowledge-Level Explanation

- HOW questions can be used to ask how an atom was proved.
  It gives the rule used to prove the atom.
  You can the ask HOW an element of the body of that rules was proved.
  This lets the user explore the proof.
- WHY questions can be used to ask why a question was asked.
  It provides the rule with the asked atom in the body.
  You can ask WHY the rule in the head was asked.

ailog: **ask** lit_l_1.

Is up_s_1 true? **why**.

up_s_1 is used in the rule $live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1$: **why**.

$live\_w_1$ is used in the rule $live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2$: **why**.

$live\_w_0$ is used in the rule $live\_l_1 \leftarrow live\_w_0$: **why**.

$live\_l_1$ is used in the rule $lit\_l_1 \leftarrow light\_l_1 \wedge live\_l_1 \wedge ok\_l_1$: **why**.

Because that is what you asked me!

Knowledge-Level Debugging

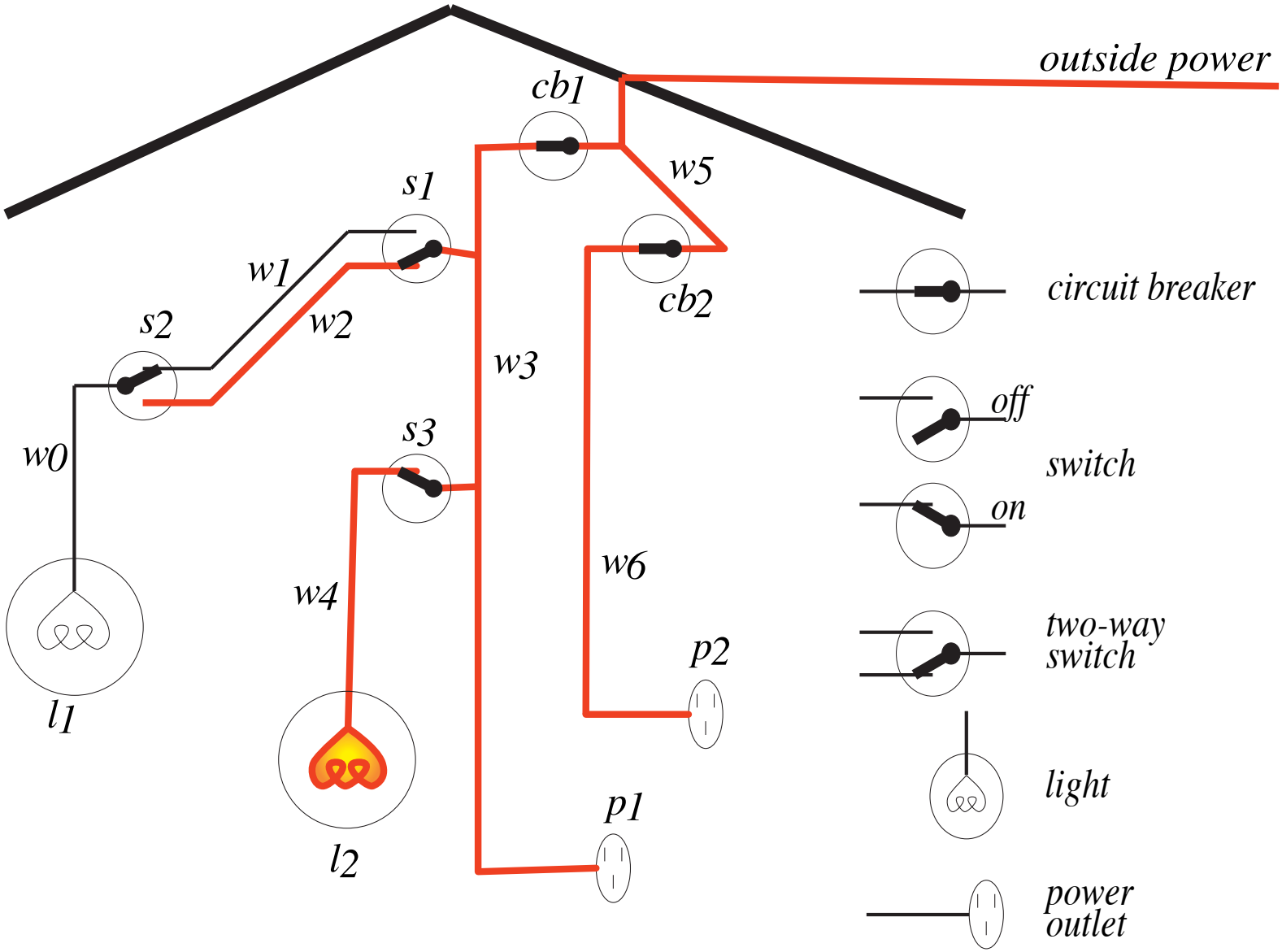
There are four types of non-syntactic errors that can arise in rule-based systems:

- An incorrect answer is produced: an atom that is false in the intended interpretation was derived.
- Some answer wasn't produced: the proof failed when it should have succeeded. Some particular true atom wasn't derived.
- The program gets into an infinite loop.
- The system asks irrelevant questions.

Debugging incorrect answers

- Suppose atom g was proved but is false in the intended interpretation.
- There must be a rule $g \leftarrow a_1 \wedge \ldots \wedge a_k$ in the knowledge base that was used to prove g.
- Either:
  - one of the a_i is false in the intended interpretation or
  - all of the a_i are true in the intended interpretation.
- Incorrect answers can be debugged by only answering yes/no questions.

Electrical Environment



Missing Answers

If atom g is true in the intended interpretation, but could not be proved, either:

- There is no appropriate rule for g.
- There is a rule $g \leftarrow a_1 \wedge \ldots \wedge a_k$ that should have succeeded.

Missing Answers

If atom g is true in the intended interpretation, but could not be proved, either:

- There is no appropriate rule for g.
- There is a rule $g \leftarrow a_1 \wedge \ldots \wedge a_k$ that should have succeeded.
  - One of the a_i is true in the interpretation and could not be proved.