Syntax/Semantic Language

**S/SL - Syntax/Semantic Language**

- Special purpose language (DSL) specifically designed for implementing compilers
- Designed for specifying multi-pass compilers
- Main language (other than PT Pascal) used in this course - you will have to write S/SL programs in quizzes and on the exam
- **S/SL** itself is a dataless executable specification language - programs can define only constants (no variables or assignment!)
- All real data manipulation done indirectly in black boxes called semantic mechanisms (implemented elsewhere)
The S/SL Computational Model

Three Components

• The S/SL program (a finite-state control)
• An integer pushdown return stack implementing S/SL rule calls
• Semantic mechanisms - abstract computation modules
The S/SL Processing Model

Pipe-and-Filter Chain

- Input/output stream model designed for joining S/SL programs end-to-end pipe-and-filter style to form multi-pass solutions
- Well suited to compiler implementation
The S/SL Processing Model with Error Streams

Pipe-and-Filter Chain

- Error stream normally does not go on to next pass, rather translated into error messages for user
Semantic Mechanisms

Enforced Information Hiding

- **S/SL** itself is, for all intents, a *dataless* programming language.
- No variables, assignments, or data operations - all data manipulation is *indirectly* carried out in semantic mechanisms.
- A small number of named *constants* may be referred to, and the value of the current *input/output/error token*, but *no operations* can be performed other than those provided by *semantic mechanisms*.
- Semantic mechanisms augment this dataless world with a set of *black boxes* with buttons (operations) for indirect data manipulation.

![Diagram of S/SL system](image)
Semantic Mechanisms

• Opaque abstract data types (like static class interfaces)
  • organize semantic operations
  • indirect manipulation of data
  • interface defined in S/SL, but implementation is hidden from the S/SL program
Semantic Mechanisms

Example

• A Symbol Table Mechanism

• From the point of view of the S/SL program, looks like an opaque magic black box

• Internally, has data structures to implement a scoped table of identifiers and their attributes
Summary

The S/SL Computational Model

• **S/SL** is a dataless push-down transducer language well suited to implementing multi-pass compilers

• **Semantic mechanisms** augment **S/SL** with black-box data manipulation modules that can be specified, invoked and queried from **S/SL** programs, but whose implementation is strictly hidden (**enforced information hiding**)

Next

• **S/SL** syntax, program structure and operations