

Compiling Modules & Packages

What do we find in a module or package declaration?

- A name and components, like in a record
- Components declared in a new name space
- Components may include procedures and functions

The module/package declares an instance, not a template

Compilation challenges

- Incomplete declarations
 - Procedure headers
 - Opaque and private types
- Specification/body separation

Semantic Processing for Modules/Packages

The AST node for a package will include:

- The name
- A pointer to a component list
- In Ada, optionally a pointer to the *private part*, which is just another component list

Semantic processing for a package AST node:

- Put the name in the symbol table as a package name
- Create a new symbol table
- Declare all of the components in that symbol table
What kind of names are the components?
- Components in the private part must not be externally visible

If there is a separate package body AST node:

- Make sure the body is not already defined
- Add body components to the symbol table, making sure they are not visible externally
- Match procedures/functions in the body with headers declared as part of the interface

Code Generation for Modules/Packages

Compiling an instance makes it easier, but not trivial.

- The variables among the components make up a single, statically allocated data area (much like a file in C).
- Declarations (possibly) and a body, if one is present, generate executable initialization code.

How can you collect this possibly dispersed code?

Compiling References to Package Components

Syntax is identical to record components, generating the same AST structure

Look-up mechanism works just like for records

Minor extensions are needed to handle packages as an alternative wherever records are expected

Compiling Classes

A class looks much like a package, but it defines a template rather than a single instance

- (Instance) variables in the class cannot be statically allocated, thus act like record fields
- Operations must be able to access a particular instance of the class when they are executed, which must be provided as an implicit parameter
- Access to visible components may use record syntax or something slightly different; either can map to the record reference AST structure or something like it
- Component visibility may be determined by more complex rules:
 - public, protected, private in C++
 - visibility lists in Eiffel