8

Annotation of the Hidden Parts of Packages

Topics:

- modified type annotations;
- annotation of hidden type definitions;
- stability constraints, application to abstract types;
- Anna model of package states, annotation of package states, package state stability;
- annotation of hidden subprogram bodies;
- consistency of package implementations;
- packages as types.

Let us assume that at a particular stage in the design of the program, an Anna specification of an Ada package has been constructed. At some later stage it will be necessary to construct a hidden part (i.e., an Ada package private part and body) that satisfies this visible specification. When faced with this situation, a number of questions immediately come to mind.

- What kinds of annotations are useful in documenting a package hidden part?
- How are the hidden annotations related to the visible ones?
- Can the visible specification be used as an implementation plan to guide the construction of a hidden part that satisfies it?

These three questions are each the subject of a separate chapter. This chapter describes the special annotations for package hidden parts and the predefined structure of the Anna package state type. Chapters 9 and 10 describe a methodology for implementing package hidden parts so that they are consistent with their visible specifications. Chapter 9 describes how to define the relationship between visible specifications and the package hidden part. This is called interpreting the visible specifications. Interpretation is possible only if the package hidden part has adequate annotations, which are also described in Chapter 9. Chapter 10 outlines processes for implementing a package hidden part using a package specification as a guide.

Let us start by listing the special features in Anna for annotating package hidden parts. First of all, there is a new kind of type annotation, called a modified type annotation. Its main purpose is to express an important property of the hidden part of many packages called stability. Secondly,
the Anna package state type has a predefined hidden structure. It is defined by an implicit virtual full declaration of the package state type as a record within the hidden part of the package. This defines the semantics of Anna package states in a natural way that is compatible with Ada language rules. Type annotations, including modified ones, may be applied to this implicit state type declaration — even though the declaration is implicit and does not appear syntactically in the package. Thirdly, the hidden subprogram annotations of a subprogram body are required to be logically consistent with the visible annotations of the corresponding subprogram specification. Essentially, requiring logical consistency between visible and hidden subprogram annotations is an Anna rule that generalizes the Ada rule requiring syntactic conformance between visible and hidden subprogram specifications. We will see that requiring logical consistency has practical repercussions on how we annotate package bodies.

The reader should review Section 4.1 on annotations and package structure. Recall that annotations in the visible part of a package are visible both outside the package and in the hidden part. To the user, visible annotations specify how to call the package operations and what results to expect (Chapter 4). To the implementor of the hidden part, the visible annotations are constraints. For example, an annotation of a visible subprogram declaration constrains the hidden subprogram body. Figure 8.1 shows the various levels of annotations in a simple package — one without nested packages. Note that the Ada private part is pictured as part of the package body.

Annotation of a package hidden part may be undertaken with various goals in mind. The simplest goal is to document properties of the implementation that are not specified in the visible part — the so-called “implementation details.” Generally, a package specification will leave the implementor a great deal of freedom to choose data structures and algorithms. These choices are made not only to satisfy the visible specifications, but also to satisfy various aspects of efficient execution, such as timing and storage allocation, which are often not mentioned in the visible specification. The first goal, therefore, is to specify the additional properties with annotations in the hidden part of the package. Examples of such annotations are given in this chapter. Consistency between the hidden part and all of the annotations that apply to it — both visible and hidden — can be checked by runtime checking methods. This is discussed in Section 8.4.

A more ambitious goal is to give hidden annotations that are sufficient to enable consistency between specification and hidden part to be established by proof methods. Towards this goal, an important concept is interpretation of visible specifications. An interpretation of a visible specification is a set of constraints on the hidden types, objects, and subprogram bodies of a package that imply consistency. Since a visible specification cannot refer to hidden details, an interpretation of it is not usually immediately obvious but must be constructed.