En un esquema de traducción escrito con ANTLR, los errores semánticos se emiten en el interior de las acciones semánticas y, por lo tanto, son completamente controlables por el programador. Sin embargo, los errores léxicos y sintácticos son, de entrada, gestionados por el código generado automáticamente por ANTLR, que implementa una política de detección y recuperación de errores. Cualquier política de este tipo pretende maximizar el número de errores detectados a la vez que se minimiza el número de veces que se emite equivocadamente un error inexistente. Si deseamos que el análisis se detenga tras el primer error, es necesario modificar el comportamiento por defecto.

En el capítulo 10 del libro de ANTLR de Terence Parr se explica cómo personalizar la política de gestión de errores. La idea básica es cambiar el código que por defecto genera ANTLR al final del método asociado a cada no terminal de la gramática; este código implementa los manejadores de las excepciones lanzadas cuando se detecta un error léxico o sintáctico y, por defecto, es como sigue:

```
catch (RecognitionException re) {
  reportError(re);
  recover(input,re);
}
```

Este código se puede sobrescribir mediante una acción rulecatch; por ejemplo, con las siguientes líneas se consigue que el análisis se detenga tras el primer error:

```
@rulecatch {
  catch (RecognitionException re) {
    reportError(re);
    System.exit(1);
  }
}
```

Sin embargo, la acción anterior no parece funcionar sobre los errores del analizador léxico. Por ello, hay que utilizar una estrategia diferente, que, sin embargo, podría ser dependiente de la versión de ANTLR utilizada. Bajo este enfoque, hay que reescribir algunos de los métodos creados por ANTLR para gestionar los errores; en particular el método emitErrorMessage, que es el que en última instancia emite el mensaje de error, se puede definir como sigue para que el traductor se detenga ante el primer error léxico o sintáctico:

```
@members {
  public void emitErrorMessage(String msg) {
    System.err.println(msg);
    System.exit(1);
  }
}

@lexer::members {
  public void emitErrorMessage(String msg) {
    System.err.println(msg);
    System.exit(1);
  }
}
```

Lo anterior funciona, al menos, con la versión 3.0.1 de ANTLR. Esta versión, curiosamente, imprime una línea adicional con la cadena "BR.recoverFromMismatchedToken" junto a cada mensaje de error sintáctico; es, probablemente, consecuencia de una línea de código no comentada en el código de ANTLR, usada para depuración.

Es importante, además, que tengas en cuenta que los siguientes aspectos de los errores emitidos por ANTLR:

- Las columnas comienzan a contarse desde cero.
- El código generado por ANTLR realiza un análisis léxico completo antes de pasar al análisis sintáctico; por ello, los errores léxicos se emiten aunque sean posteriores a un error sintáctico o semántico en el fichero de entrada.
- El analizador sintáctico generado por ANTLR no comprueba que la entrada se haya acabado una vez que finaliza la construcción del árbol; por ello, en algunos casos no emitirá ningún error en el caso de caracteres sobrantes al final del fichero. Para que actuará de otro modo habría que introducir una regla inicial explícita que asegurara la aparición del final del fichero, pero no es necesario que lo hagas.