GUIDE: Code Generation

Luis Da Costa and Marc Schoenauer

April 15, 2008

Chapter 1

Code Generation in GUIDE

The idea GUIDE can be used to generate an EA to be run by itself (”*standalone*”) or it can also be used to produce an *evolution engine* to solve a problem as part of another application (see Fig. 1.1). As such, we need a centralized code generation scheme offering this flexibility.

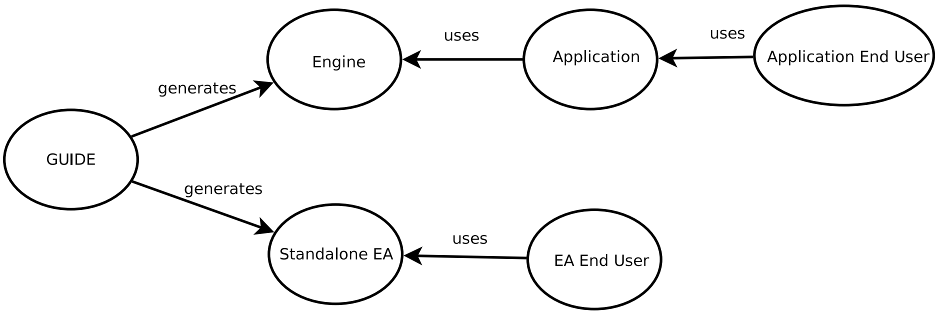

Figure 1.1: GUIDE: two ways of generating an EA

The more straightforward way of reaching this goal is by considering a standalone EA as a special type of engine: one that is run by a special application, *i.e.*, the end-user itself. GUIDE always generates an *engine*, that needs to be ”controled” (*i.e.*, started, stopped, paused/resumed or given feedback) by another program, the *controler* (see Fig. 1.2).

Under this scheme, the engine provides 2 things: (1) all the logic and important code to execute the generated EA, and (2) a way of creating itself. The controler has to create the engine, and to start it. On demand, it provides the fitness for the individuals (equivalently, population) and sends

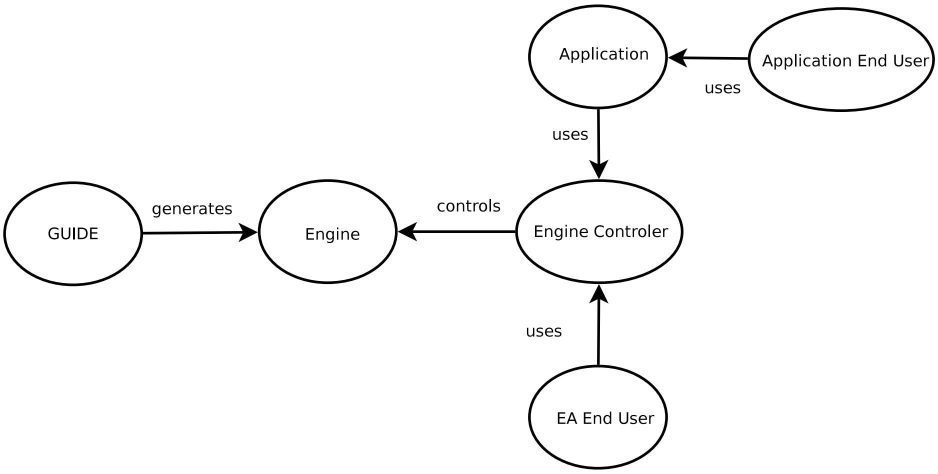

Figure 1.2: The engine and the controler idea

messages to the engine concerning the continuation or pausing of the execution. The important point here is that the engine gives total flexibility to the controler to perform those tasks depending on the logic of the application. A summarizing scheme of this idea is shown in Fig. 1.2

Practical implementation details. Implementing this idea is achieved by the following basic steps:

1. The code representing an engine is located on a directory called **src**, directly under the directory of the experiment. The directory of the experiment himslef will be where the specification file for the generation of the algorithm is located. For example, if the specification for the current experiment is at

   ```
   /myDir/myExperiment.xml
   ```

   then the code corresponding to the EA will be located at the following directory:

   ```
   /myDir/myExperiment/
   ```

   and the code corresponding to the generated EA will be at:

   ```
   /myDir/myExperiment/src/
   ```

2. GUIDE's distribution provides two interfaces ("header files"), called **IEOEngine.h IEOEngineControl.h**. These two files are provided in the distribution of GUIDE, and must exist in a directory specified in the settings file, under the tag DIR on INCLUDES[1]

3. The controler (see Fig. 1.2 for a quick reminder of what is a Controler):

   (a) A general, abstract implementation of a Controler is generated by GUIDE at code-generation time: it is on the file **GUIDEGenericEngineControl.h**, on the source directory of the EA.

   (b) The user (equivalently, application) must provide a specific implementation of a Controler. An example of a Controler is provided with GUIDE's distribution: an user can modify it to make it suitable for their needs. It must be kept in the directory specified in the settings file, under the tag CONTROLERDIR (see footnote 1)

4. Any other classes needed by the engine should be specified in the directory of the settings file identified by the tag DIR on EXTRACODE (see footnote 1)

An example Here I put the example that is in GUIDEKernel (for Evotest)

[1] If this last sentence is confusing to you, or you would like more details, please refer to the Installation Manual, found in `http://gforge.inria.fr/projects/guide/`, Tab Docs, and here `http://www.lri.fr/~ldacosta/guide/installation/GUIDEInstallationManual.html`