Different components in this system:

1. APL (Application provisioning library): a dll. It is a library which provides all the interfaces for the provisioning and licensing related functionality. It also included the functionality for serial number validations. This dll will be used at the time of installation and at the time of package launch every time.

It should be a static lib so that the calling exe need not check its signature every time.

1. DMM (database management module) : a static lib having all the sqlite databases interaction interfaces and implementations. It will provide interfaces for all different kinds of databases.
2. EDMM (encrypted database management module): a static library. A layer above DMM to provide read and write functionality in encrypted format. This is as a derived class of DMM.
3. Installer engine (IE): a dll, which will handle all the responsibilities of installing and uninstalling the packages on the machine.
4. SLE (secure licensing engine): a static lib which deals with the licensing related thing. It directly works on the machine id level and validates any license. It defines the format of license and the algorithm for validating it. APL uses it internally.
5. LATE (License activation terminal engine): a component on the server side which will activate the license and maintain the licenses related to serial numbers on the server. It will use EDMM also
6. SMM (serials management module): a component which maintains all the serial numbers which are valid. This doesn’t care about which serial number has been activated or sold to whom. That work is done by LATE.
7. PBS (package builder system): a tool having multiple scripts which constitutes all the content and make a package. A package is made up of protected content databases and the assets.
8. PA (Package assembler): a tool having multiple scripts which assembles all the packages and makes a media out of that. This makes the whole family of packages which can then be burnt on the DVD and can be sold out.
9. Executable: There will be multiple executables in the system which will be using different dlls and static libs and serve as the module which can be used in different processes.
10. Home window: a flex component, a home window UI application which will detect the installed packages on the machine and then show users the options. After selecting a package, it would validate the package licensing.
11. IDApp (interactive display application): the application which will display videos, pdf and other interactive swf inside it. It is a flex application.
12. Cleanup script: the script which will clean up all the components and objects from the machine.
13. IDAppNative: This is the concept of an executable which is used for the communication over a socket from any module and it then loads a dll and executes its functions according to the arguments given to it over the socket in form of xml messaging. It also computes the signing of every dll which it will use. The communication over the socket will use session keys for every request which a module makes to it. BE creates thread for every request and then gets the result back and send it back to the requester with the session key. (??) (It can be a complicated design).
14. Bridge interface: a communication library between flex and native communication using stdin and stdout interfaces. It supports Unicode also. This is on native side.

**Different classes and functionalities in the code:**

1. Encryption and decryption functionality – symmetric as well as asymmetric
2. Sqlite db access functionality
3. Xml parser functionality
4. Socket programming and data communication
5. Named pipe communication
6. Other utilities functions like file/folder copying etc. etc.
   1. Random string generation
7. Logging functionality – multiple log files.
   1. Package installer logs
   2. package provisioning logs
   3. Application launch and interaction related logs