# SOFTWARE DESIGN

A software design document (SDD) is a written description of a [software](http://en.wikipedia.org/wiki/Software) product, that a software designer writes in order to give a [software development](http://en.wikipedia.org/wiki/Software_development) team overall guidance to the architecture of the software project. An SDD usually accompanies an architecture diagram with pointers to detailed feature specifications of smaller pieces of the design.

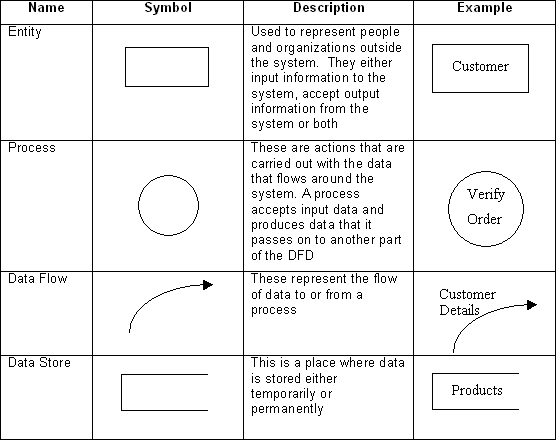
Practically, a design document is required to coordinate a large team under a single vision. A design document needs to be a stable reference, outlining all parts of the software and how they will work. The document is commanded to give a fairly complete description, while maintaining a high-level view of the software.

There are two kinds of design documents called HLDD (high-level design document) and LLDD (low-level design document).

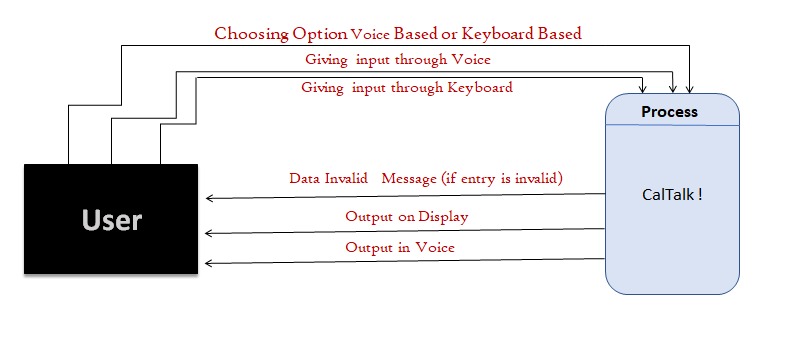
The SDD contains the following documents:

* 1. The [**data design**](http://en.wikipedia.org/wiki/Data-driven_design) describes structures that reside within the software. Attributes and relationships between [data objects](http://en.wikipedia.org/wiki/Data_object) dictate the choice of [data structures](http://en.wikipedia.org/wiki/Data_structures).
  2. The [**architecture design**](http://en.wikipedia.org/wiki/Software_architecture)uses information flowing characteristics, and maps them into the program structure. The transformation mapping method is applied to exhibit distinct boundaries between incoming and outgoing data. The data flow diagrams allocate control input, processing and output along three separate modules.
  3. The [**interface design**](http://en.wikipedia.org/wiki/Interface_design) describes internal and external program interfaces, as well as the design of human interface. Internal and external interface designs are based on the information obtained from the analysis model.

The [**procedural design**](http://en.wikipedia.org/wiki/Procedural_design)describes structured programming concepts using graphical, tabular and textual notations. These design mediums enable the designer to represent procedural detail that facilitates translation to code. This blueprint for implementation forms the basis for all subsequent software engineering worked

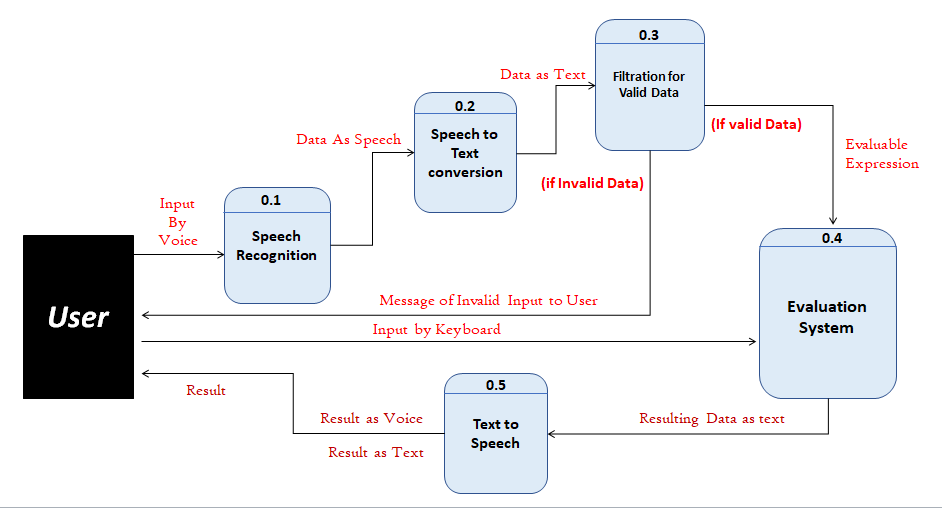


**3.1.0 DFD LEVEL 0**

****

**Fig 3.2: 0 Level DFD**

* + 1. **DFD LEVEL 1**

****

**Fig 3.3: 1 Level DFD**