Goals of software design:

--------------------------

* Correctness – what is needed is coded - This is extreme programming in which design to known requirements but not more.
* Robustness – what is coded is working as coded
* Flexible - changes are easy to make
* Efficiency - Algorithm are efficient
* Reusability - OO concepts have been used
* Reliability - Testing is good
* Usability - user friendly

Design principles:

------------------

Achieving correctness:

\* Modules and Interfaces make it easy to understand (part of correctness)

\* Classes: Domain (specific to application like Teller), non-Domain (generic, abstract and utility like File, Database)

\* Refactoring: promoting primitive attribute to class and introducing abstract base classes or interface

Achieving Robustness:

\* verify inputs (ensuring environmental robustness)

\* initializing to improve robustness

\* enforcing our intentions by protecting against design and implementation errors

Achieving Flexibility:

\* Use inheritance (for adding same functionality) and Delegation (to add different functionality)

\* Use appropriate Design Patterns

 - create objects in variable configurations determined at runtime

 - represent variable trees or objects or other structures at runtime

 - change, recombine, or otherwise capture the mutual behavior of a set of objects

 - create and store a possible complex object of a class

 - configure objects of predefined complex classes, or sets of classes, so as to interact in many ways

Achieving Reusability:

\* use OO principle (Dependency, Aggregation, and Inheritance) along with design patterns and component tech.

Achieving Efficiency:

\* from start identity crucial spots or just later do as needed

\* speed efficiency bottlenecks: loops, eliminate remote calls, certify nested calls

\* storage efficiency (time - space tradeoffs): store only data needed (tradeoff storage efficiency vs. time to extract and reintegrate), Compress the data (tradeoff storage efficiency vs. time to extract and decompress), Store in order of relative frequency (tradeoff storage efficiency vs. time to determine location)