Software Engineering Lab
Spring 87

COUPLING
DEFINITION & CONSEQUENCES

- the degree to which each program module relies on each one of the other modules.

- With low coupling, a change in one module will not require a change in the implementation of another module.

- Low Coupling + High Cohesion = sign of well-structured computer system.
SYSTEMS WITH HIGH COUPLING

- Change in one module forces a ripple of changes in other modules.
- Modules are difficult to understand in isolation.
- Modules are difficult to reuse or test because dependent modules must be included.
HOW TWO CLASSES COUPLED TOGETHER

- A has an attribute that refers to (is of type) B.
- A calls on services of an object B.
- A has a method which references B (via return type or parameter).
- A is a subclass of (or implements) class B.
TYPES OF COUPLING

- Content coupling (high)
- Common coupling
- External coupling
- Control coupling
- Stamp coupling (Data-structured coupling)
- Data coupling
- Message coupling (low)
CONTENT COUPLING (HIGH)

- Two (or more) modules exhibit content coupling if one refers to the “inside” - the “internal” or “private” part - of the other in some way
  - Module A “branches” or “falls through” into Module B (by containing a GOTO statement that transfers control somewhere into the middle of Module B);
  - Module A refers to, or changes, Module B's internal (and, again, “private”) data
  - Module A changes one of the statements in Module B's object code.
- Good for optimization if it is done automatically.
Two or more modules exhibit common coupling if they refer to the same global data area.

- Changing the shared resource implies changing all the modules using it.
Two or more modules exhibit external coupling if they share direct access to the same I/O device or are “tied to the same part of the environment external to software” in some other way.

- two modules share an externally imposed data format, communication protocol, or device interface.
Two modules exhibit control coupling if one ("module A") passes to the other ("module B") a piece of information that is intended to control the internal logic of the other. This will often be a value used in a test for a case statement, if-then statement, or while loop, in module B's source code.

- one module controlling the logic of another, by passing it information on what to do (e.g. passing a what-to-do flag).
Two modules ("A" and "B") exhibit stamp coupling if one passes directly to the other a "composite" piece of data such as record, array, tree, list, ...

- modules share a composite data structure and use only a part of it, possibly a different part.
- This may lead to changing the way a module reads a record because a field, which the module doesn't need, has been modified.
Two modules exhibit data coupling if one calls the other directly and they communicate using “parameters”.
- each parameter is an elementary piece.
- parameters are the only data which are shared.
MESSAGE COUPLING (LOW)

- This is the loosest type of coupling. Modules are not dependent on each other, instead they use a public interface to exchange parameter-less messages (or events).