The program structure should be partitioned both horizontally and vertically.   
  
(1) Horizontal partitioning defines separate branches of the modular hierarchy for each major program function.  
  
Simplest way is to partition a system into: input, data transformation (processing), and output  
  
Advantages of horizontal partition:  
- easy to test, maintain, and extend  
- fewer side effects in change propagation or error propagation  
  
Disadvantage: more data to be passed across module interfaces  
- complicate the overall control of program flow  
  
(2) Vertical partitioning suggests the control and work should be distributed top-down in program structure.  
  
Advantages:   
- good at dealing with changes:  
- easy to maintain the changes  
- reduce the change impact and and propagation

Structural Partitioning

The program structure should be partitioned both vertically and horizontally. As describe in Figure 18.4 a horizontal partitioning describe separate branches of the modular hierarchy of reach major program function. The Control modules, represented in a darker shade are used to coordinate communication among and execution of program functions. The easiest approach to horizontal partitioning describes 3 partitions - input, data transformation often called processing and output. The Partitioning the architecture horizontally gives a number of distinct advantages:

Results in software which is easier to test

leads to software which is easier to maintain

Results in propagation of less side effects

Results in software which is easier to extend

Because huge functions are decoupled from one another change tends to be less extensions and complex to the system a general occurrence tend to be easier to accomplish without side effects. The negative side horizontal partitioning often causes more data to be passed across module interfaces and can complicate the overall control of program flow if processing needs rapid movement from one function to another.

The Vertical partitioning often called factoring suggests in which control and work should be distributed top- down in the program architecture. The Top-level modules should perform control functions and do little original processing work. Modules which reside low in the architecture should be the workers performing all computational, input, and output tasks.

The nature of modification in program architectures justifies the requirement for vertical partitioning. The change in a control module high in the architecture will have a higher possibility of propagating side effects to modules which are subordinate to it. The change to a worker module given its low level in the structure is less likely to cause the propagation of side effects. In common changes to computer programs revolve around changes to input computation or transformation and output. Overall control structure of the program example for its basic behavior is far less such as to change. For this reason vertically partitioned architectures are less likely to be susceptible to side effects when changes are made and will thus be more maintainable - a key quality factor.