Architectural Views

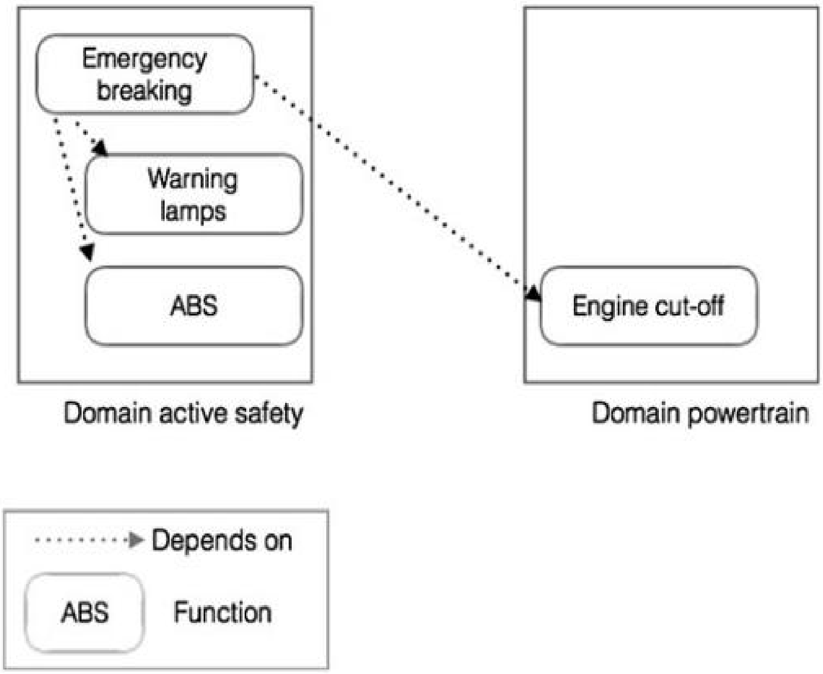
There are three architectural views.

1. Functional view
2. Physical view
3. Logical view

1. Function view:

- The functional view is a view which focuses on the functions of the vehicle and their dependencies on one another.
- The functional view consists of functions(plotted as round-edge rectangles), domain(plotted as sharp-edged rectangles) and dependency relations (plotted as dashed lines).
- The common domains are:

  Powertrain
  Active Safety
  Chassi and body
  Electronic systems
- Function view provides the architects with the possibility to cluster functions and distribute them to the right department to develop and to reason about these kinds of functionality.
- Example:



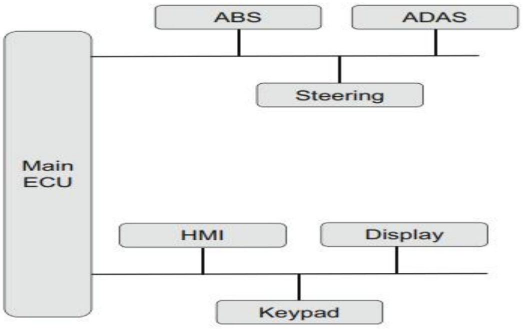
-

How to build?

- ✧ List all the functions and their dependencies.
- ✧ Group them into domains according to their functionalities.
- ✧ The organization of the functions is based on how they are dependent on each other with the principle that the number of dependencies that cross-cut the domains should be minimized.

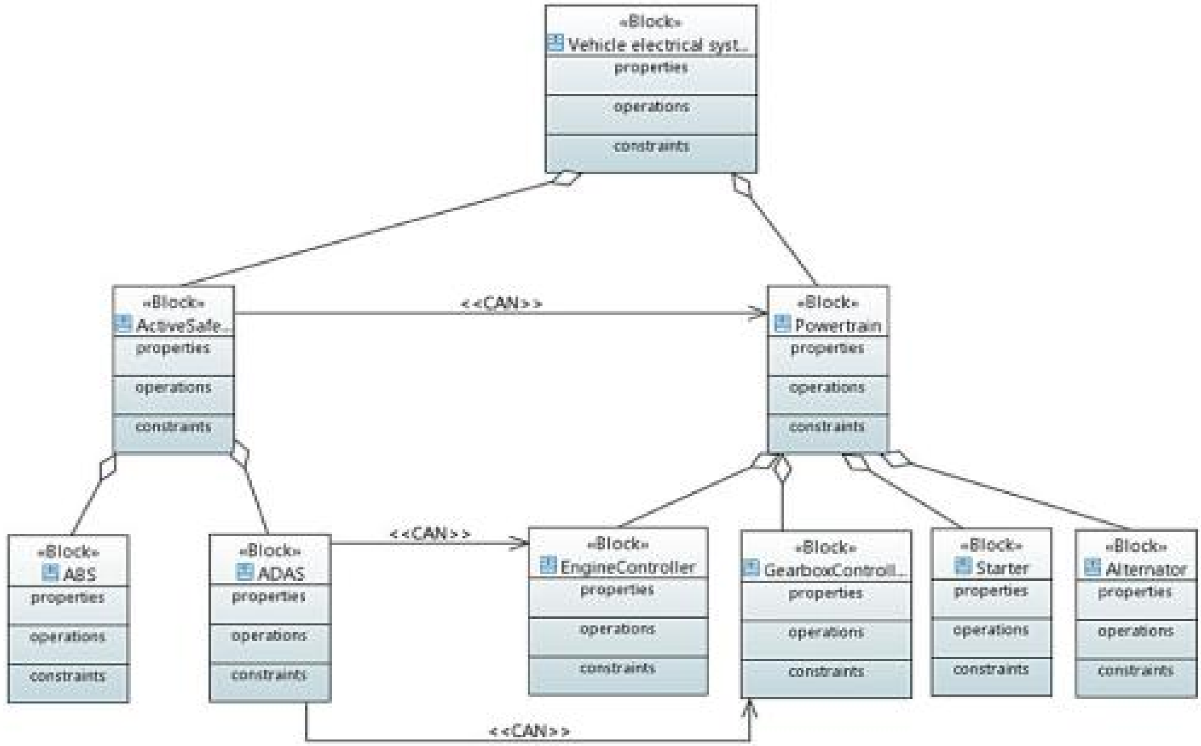
2. Physical View

- ✧ Physical view is the view of the entire electrical system at the top level accompanying with lower level diagrams.
- ✧ This view of the architecture provides the possibility to present the topology of the electrical system.
- ✧ This provides the architects with a way to reason about the placement of the ECU's on the communication buses.
- ✧ In the early it was very simple to represent physical view but due to the increasing no of ECU's, it become complex.
- ✧ The modern physical view on the topology also includes information about the processing power and operting system of each ECU.
- ✧ Example:



3. Logical View

- ✧ The logical view focuses on the software of the system.
- ✧ In the logical view we show which classes, modules and components are used in system and how they are related to each other.
- ✧ The notation used for this model is mostly UML or SysML.
- ✧ For the logical view, the architects uses different diagrams such as class diagrams, component diagrams to show various levels of abstraction of the software of the system.
- ✧ Example:



How to build?

- ✧ Identify all the components and model them as UML classes.
- ✧ Identify the relation between these components and add them in the form of associations.
- ✧ Direction of association should be correct as it indicates the how the communication takes place.