# SMV Configuration

## Structure

The configuration file essentially specifies the steps that are run against the module. The schema for the XML configuration file is defined in Config.xsd in the SmvLibrary project. Figure 1 below shows the basic structure of the schema and Figure 2 shows the expanded Action node specified in Figure 1:

Figure : Configuration structure

Figure : Action node

* **SMVConfig -** Root element of the schema  
  + **Variables** – Contains 0 or more SetVar nodes
    - **SetVar** – Specifies a set of key/value pair which are loaded during initialization and is available to all the action nodes in Build & Analysis
  + **Build** – Executed after initialization. Contains 0 or mode Action nodes
    - **Action** – Refer to the bullet on Action node below
  + **Analysis** – Executed after build. Contains 0 or mode Action nodes
    - **Action** – Refer to the bullet on Action node below
* **Action** – Action nodes are the building blocks of SMV in which most of the commands and configurations is defined. It has the following attributes:  
  + **Attributes**:
    - **Name (required)** – Name of the action
    - **NextAction –** Name of the action to be run after the current action
    - **BreakOnError –** Whether to break the execution if the current action fails
    - **ExecuteOn –** Whether to run locally or on the cloud (IF ENABLED).
  + **Child nodes:**
    - **Path –** Absolute path or path relative to directory where SMV is run, where the action is executed.
    - **Env –** Specifies a set of key/value pair which are set on the environment before executing the commands
    - **CopyArtifact –** This can be used to copy artifacts from a SMV database as specified in the connection string of SMV.EXE. The allowed artifacts are specified in the schema file. The type is usually specified in smv2sql.exe (look at sources).
    - **Command –** Specifies the command to be executed and the arguments to be passed to the command. An action can contain multiple commands and all commands within an action are always run **sequentially**.

## Branching in SMV

It is possible for an action to have dependencies in SMV. An action can have a dependency on another action within its parent Build or Analysis node. It is specified using the **NextAction** attribute and it ensures that the action is executed only after all the dependencies have been executed. Using this approach, it is possible to create a control flow which resembles a **linear chain**. Each root action inherits a dictionary of the SMV global properties and any changes to it by an action are inherited by the subsequent actions in that chain. This allows us to **maintain state** between the actions in a chain.

Before executing the build or the analysis nodes, SMV extracts all the root actions (actions without dependencies) and enqueue them into the master queue. Once an action has been executed then its child node will be pushed into the master queue. SMV executes the chains **sequentially** or **parallely** in an **interleaved** manner based on the number of threads (refer Parallelization below).

## Parallelization in SMV (locally)

SMV supports multithreading and can be configured to run any number of actions parallely. The number of threads can be specified in the variables node with key as “**localThreads**”. If the number of threads is 1 (**default is 5**), then there will be sequential execution. Otherwise, multiple branches will be executed parallely.