1 Work

The first component of the work is a ruby module designed specially to act as a back-end for the entire solution. A module is a collection of programming functions and libraries that are bundled together to achieve a particular task. In ruby, modules are called gems. For writing my first gem, the official ruby guide was used as guide [**?**].

The file `lib/deeploy.rb` is what sets up the database connection and bootstraps the whole module by auto-loading every component. To ease integration, the file loads configuration properties from the file `lib/config/application.yml`. `Application.yml` contains configuration for production, development and testing, this is to allow separation and configuration of different parameters. The file specifies location of the database file when in development, or it has the database drivers in production with information about connection parameters. The file also describes the network interfaces to be used in each environment.

`lib/deeploy.rb` has a helper method that returns the current network interface and network mask, this function is responsible for generating IP addresss and also for verifying that the virtual network is up and running. It does so by issuing commands for bringing up the virtual networks `vboxnet` up, then using network sockets to extract relevant information. This function is the programatic equivalent to the command `ifconfig or ipconfig`.

The file also contains a static function that returns a map of all supported distributions and the associated Vagrant machine. The function is helpful when validating creation, and will error out if the distribution name is not in the list. When extending the supported distribution, this function needs updating.

Similarly to the supported distributions function, there is a static function to return the supported packages list, used for displaying the options and validating unsupported packages.

The module provides a function slugify, its purpose is to convert string with special characters and spaces to a set of words and dashes, non-ASCII characters are stripped.

IP address helper function checks if the supplied is part of the current network by using ruby's `ipaddr` build in module.

The file `lib/vm.rb` contains the code for managing the virtual machine instances, the class is called `Deeploy::Configurable::VM`. The class inherits from `Configurable`. Calling `VM.new` contains validators that raise errors upon initialising the class inappropriately. The method is responsible for specifying the distribution, available resources, configuration destination, name of the instance, packages, open ports and prepares it for creation. A design decision was made to allow a machine to be configured with the `new` operator and then created by issuing `machine_instance.build()`.

The method `VM.alive()` verifies if the machine is alive by trying to listen on ssh port, if it fails within a time-out, the machine is updated to state of not alive.

`VM.get()` is used when performing power up, power down and destroy virtual machines. It queries the database and discovers everything about the instance, from packages, to RAM and location of configuration files.

The virtual machine is built only when an instance is initialised properly by calling `VM.new` with the correct arguments and then invoke `VM.build()`. The design decision to have a separate functionality of bootstrapping instances was mainly for cleaner testing by compartmentalising the components into smaller elements. The `build` method also accepts a boolean argument. This argument tells the current build if it is in "dry-run" mode. The mode, the machine will create directories and configuration without actually bringing the instance up and running. This is again, done for testing reasons - they verify that different distributions have appropriate configurations.