OpenShift( provides a set of predefined runtime use cases, that are user configurable, and allow for the deployment of applications. These predefined runtimes are modeled as **OpenShift templates**.

A template describes a set of objects that can be parameterized and processed to produce a list of objects for creation by OpenShift. The objects to create can include anything that users have permission to create within a project, for example services, build configurations, and deployment configurations. A template may also define a set of labels to apply to every object defined in the template

This means that typically in a template we will have:

* A set of resources that will be created as part of “creating/deploying” the template
* A set of values for the parameters defined in the template
* A set of labels to describe the generated resources

A template will be defined in JSON or YAML format, and will be loaded into OpenShift for user instantiation, also known as application creation.

The templates can have global visibility scope (visible for every OpenShift project) or project visibility scope (visible only for a specific project).

Design our template

Designing the contents of template includes set of layers of resources with the following structure :

1. OpenShift Images: Base images we will be using for our containers.
2. Builds: Generate an image from source code (application source or Dockerfile source).
3. Images: Images produced by the builds.
4. Deployments: What images will be deployed and how.
5. Abstractions: Additional resources needed for our application, like networking, storage, security,…​

### Layer1: OpenShift images

In this first layer, we will need to define all the “base” images we will be using for our containers. These images typically will not be part of the template, but they need to be identified. These can be S2I images or plain Docker images.

In our sample application, we will be using 2 base images:

### Layer2: Builds

This layer defines all the builds we will require for our application. A build is the process of transforming input parameters into a resulting object. Most often, the process is used to transform source code into a runnable image.

**BuildConfig**

A **[BuildConfig](https://docs.openshift.com/enterprise/3.0/dev_guide/builds.html" \l "defining-a-buildconfig)** object is the definition of the entire build process.

A build configuration consists of the following key parts:

* A source description (**Where is your source code?**)
* A strategy for building (**How to build your image?**)
  + Source-To-Image: Transform your application into a runnable docker image, using a S2I image for building and running your application.
  + Docker: Your Dockerfile will be built into an image. This image will contain both, the runtime and the application already built.
  + Custom: You provide the building method in a Docker image.

### Layer3: Images

This layer defines all the images produced by the builds.

In our sample application we will be producing an image defined in a new ImageStream.

{

"kind": "ImageStream",

"apiVersion": "v1",

"metadata": {

"name": "svn-ifx",

"labels": {

"application": " svn-ifx "

}

},

"spec": {

"dockerImageRepository": "",

"tags": [

{

"name": "latest"

}

]

}

}

### Layer 4: Deployments:

This layer defines the core of our applications. It defines what will be running in OpenShift.

**DeploymentConfig**

A **[DeploymentConfig](https://docs.openshift.com/enterprise/3.0/architecture/core_concepts/deployments.html" \l "deployments-and-deployment-configurations)** is a definition of what will be deployed and running on OpenShift 3.

A deployment configuration consists of the following key parts:

* A replication controller template which describes the application to be deployed. (**What will be deployed?**)
* The default replica count for the deployment. (**How many instances will be deployed and running?**)
* A deployment strategy which will be used to execute the deployment. (**How it will be deployed?**)
* A set of triggers which cause deployments to be created automatically. (**When and Why will it be deployed?**)