Customizing the execution environment

Pipeline is designed to easily use Docker images as the execution environment for a single Stage or the entire Pipeline. Meaning that a user can define the tools required for their Pipeline, without having to manually configure agents. Practically any tool which can be packaged in a Docker container. can be used with ease by making only minor edits to a `Jenkinsfile`.

```
// Declarative //
pipeline {
    agent {
        docker { image 'node:7-alpine' }
    }
    stages {
        stage('Test') {
            steps {
                sh 'node --version'
            }
        }
    }
}
// Script //
node {
    /* Requires the Docker Pipeline plugin to be installed */
    docker.image('node:7-alpine').inside {
        stage('Test') {
            sh 'node --version'
        }
    }
}
```

When the Pipeline executes, Jenkins will automatically start the specified container and execute the defined steps within it:

```
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] sh
[guided-tour] Running shell script
+ node --version
v7.4.0
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

Caching data for containers

Many build tools will download external dependencies and cache them locally for future re-use. Since containers are initially created with "clean" file systems, this can result in slower Pipelines, as