Sandman 2

Sandbox

A Sandbox can contains one or more components.

- May be a different sandbox type
  - Sandbox types are determined at sandbox creation (sm init component.branch.sandboxtype)
  - Sandbox types can modify script behavior by using different environment variables
  - Sandbox types can run different scripts as part of the build / test / publish cycle.
  - Sandbox types can contain different build steps.
    - Build
    - Test
    - Publish
    - User Defined
  - Sandbox type names can include wildcards. If there is more than one sandboxtype that matches, the first match is the type.

Components

Components are the basic building block in Sandbox Manager 2.

- Components consist of aspects.
- Components are typically a deliverable, an independently built block or a library.
- Components may contain different instructions on how to build.
- Components can override aspect definitions, but cannot add new aspects.
- Components contain a list of dependencies.
- Components may contain attributes for external tools, such as buildbot.
  - An example would be this is a non-builtbot component, so don’t build this independently of another component.

Aspects

Aspects are what are traditionally thought of as a source control repository.

A sandbox will typically contain at least a code and built aspect. Some aspects may not be checked out depending upon architecture or Operating System.

Example: built.linux_x86-64 or built.debian_arm64 or report.windows_32

A script file in the configuration repo should return the current build type in reference to the sandbox type. (The sandbox type may provide additional built configuration options to the script.) If the script file doesn’t return, built aspects will be just be built.

Aspects (Continued)

- A sandbox can have an arbitrary number of aspects.
- If there are more than one built aspect included in each component, a publish will occur depending upon the sandbox type. (Sandboxtype configures which aspects will be published, this may be zero or more.)
- Aspects define the VCS system for that aspect. This can be overridden in a component.

Sandman (sm?)

Init (Get sandbox with aspects accord to file)

Remove (Delete it from my environment)

update(Git pull, bzr pull, svn up, get new dependencies)

status(git status, bzr status, svn status)

missing[what is the git term for this?] (git fetch --dry-run branchname? , bzr missing, svn status --show-updates)

Example remote configuration file

```
{
“aspects”: [
        {“name”: “built”, “type”: “built”, “vcsrepo” : {“provider”: “git-annex”, “source”:
“{user.gitannex.name}@10.10.10.100:/gitroot/${compoment}/built”, “revision”: “${branch}-${built}”}},
        {“name”: “memcheck”, “type”: “built”, “vcsrep”: {“provider”: “git”, “source”: “{user.git.name}@10.10.10.100:/gitroot/${compoment}/memcheck”,
“revision”: “${branch}”}},
        {“name”: “code”, “type”: “code”, “vcsrepo”: {“provider”: “git”, “source”: “{user.git.name}@10.10.10.100:/gitroot/${compoment}/code”, “revision”:
“${branch}”}},
        {“name”: “test”, “type”: “code”, “vcsrepo”: {“provider”: “svn”, “source”: “{user.svn.name}@10.10.10.100:/svnroot/${branch}/${component}/test”,
“revision”: “HEAD”}}, ],

“commands”: [
        {“name”: “clean”, “type”: “script”, “command”: “./clean.py”},
        {“name”: “build”, “type”: “script”, “command”: “./build.sh”, “env”: {“xyz”: “zy” } },
        {“name”: “test”, “type”: “script”, “command”: “../buildscripts/test.py”},
        {“name”: “copy”, “type”: “script”, “command”: “./copytobuilt.py”},
        {“name”: “assemble_run”, “type”: “script”, “command”: “./.if_top/assemble_run.py”},
        {“name”: “publish_prep”, “type”: “builtin”, “command”: “”] },
        {“name”: “publish”, “type”: “builtin”, “command”:“” }],
```

Example remote configuration file (Continued)

```
“sandbox_types”: [
            {“name”: “dev”, “commands”: [ “clean”, “build”, “test”  ], “env”: {“buildtype”: “dev”}, “default”: true, “aspects”: [“built”, “code”, “test”],
“dependency_types”: {“code”: [“code”, “test”], “built”: [“built”, “test”] } }, // Default means if it doesn’t match any other sandbox type, it is this type.
            {“name”: “continuous.*”, “commands”: [ “clean”, “build”, “test”  ], “env”: {“buildtype”: “dev”}, “aspects”: [“built”, “code”, “test”],
“dependency_types”: {“code”: [“code”, “test”], “built”: [“built”, “test”] } },
            {“name”: “official.*”, “commands”: [ “clean”, “publish_prep”, “build”, “test”, “publish” ], “env”: {“buildtype”: “official”}, “aspects”: [“built”, “code”,
“test”], “dependency_types”: {“code”: [“code”, “test”], “built”: [“built”, “test”] } },
            {“name”: “test.*”, “commands”: [ “clean”, “publish_prep”, “build”, “test”, “publish” ], “env”: {“buildtype”: “official”}, “aspects”: [“built”, “test”],
“dependency_types”: {“code”: [“built”, “test”], “built”: [“built”, “test”] } },
            {“name”: “memcheck.*”, “commands”: [ “clean”, “publish_prep_memcheck”, “build”, “test”, “publish_memcheck” ], “env”: {“buildtype”: “dev”,
“memcheck”: “yesplease”}, “aspects”: [“built”, “code”, “test”, “memcheck” ], “dependency_types”: {“code”: [“code”, “test”], “built”: [“built”, “test”,
“memcheck”] }  }  ]

“components”: [
            {“name”: “sandman2”, “dependencies”: [], “attributes”: {“variable”: True } },
            {“name”: “xyzzy”, “aspects”: [{“name”: “code”, “type”: “code”, “vcsrepo”: {“provider”: “git”, “source”: “github:/xyzzy/xyzzy”, “revision”:
“3b9e9aa094cd3c3ca0e1f8d756eb37ff0ae3135f”}}, {“name”: “test”, “type”: “none”} ]},
            {“name”: “asdf”, “dependencies”: [{“component”:“xyzzy”, “type” : “code” }] },
            {“name”: “agasdj”, “dependencies”: [{“component”: “xyzzy”, “type”: “built” }] },
            {“name”: “ss”, “commands”, [ {“name”: “build”, “type”: “script”, “command”: “./ss_build.sh”} ], “aspects”: [{“name”: “code”, “type”: “code”,
{“vcsrepo”:“provider”: “bzr”, “source”: “{user.bzr.name}@10.10.10.100:/reporoot/${branch}/${compoment}/code”, “revision”: “-1”}} ] }
            ]
}
```

Local Configuration

```
{“vcsrepo”: {
        “path” : “~/.sandman_conf”,
        “provider”: “git”,
        “source”: “http://10.10.10.100:/reporoot/${branch}/${compoment}/code”,
        “revision”: “master”,
},
“user”: {
      “git”: {“name”: “kim”},
      “bzr”: {“name”: “kim.ebert”}
}}
```

What we should start with.

1. Git only repos supporting all the stuff in the config to start with?
2. Integrate with buildbot (bu2, and individual builders)
3. Add BZR and Git-Annex support
4. Integrate with our current BZR repos (Replace sadm)
5. Add tools to migrate large number of repos
6. Migrate components to Git and Git-Annex
7. Eliminate buildscripts