A Container description

Glyn Normington Steve Powell

April 15, 2014

This document attempts to describe *container*s (see in, for example, [2]) in a precise fashion.

Contents

1 Introduction

This is a document that records the deliberations of Glyn and Steve as they come to grips with "what containers really are"[1].

2 Overview of this document

This document is a rag-bag of concepts and ideas (at the moment). The intention is to find the right decomposition of ideas to simply describe the state, and state transitions, of *Containers* and the *Jobs* that they *Run*.

3 Containers

MultiSS

$$dummy : \mathbb{N}$$

$$[FILESYSTEM, NETWORK, PID, TASK]$$

Job

$$t : TASK$$

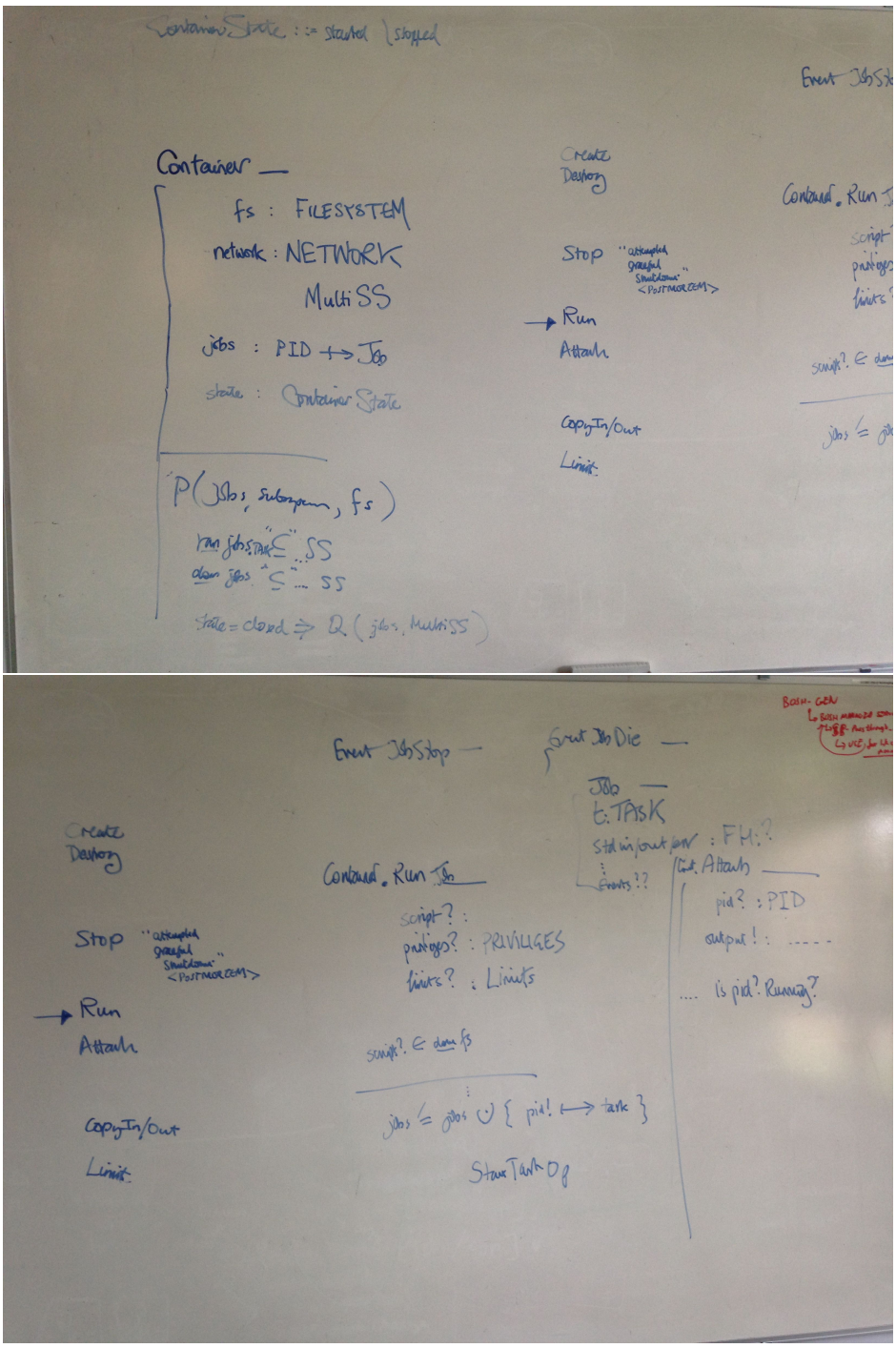
$$ContainerState ::= STARTED \mid STOPPED$$

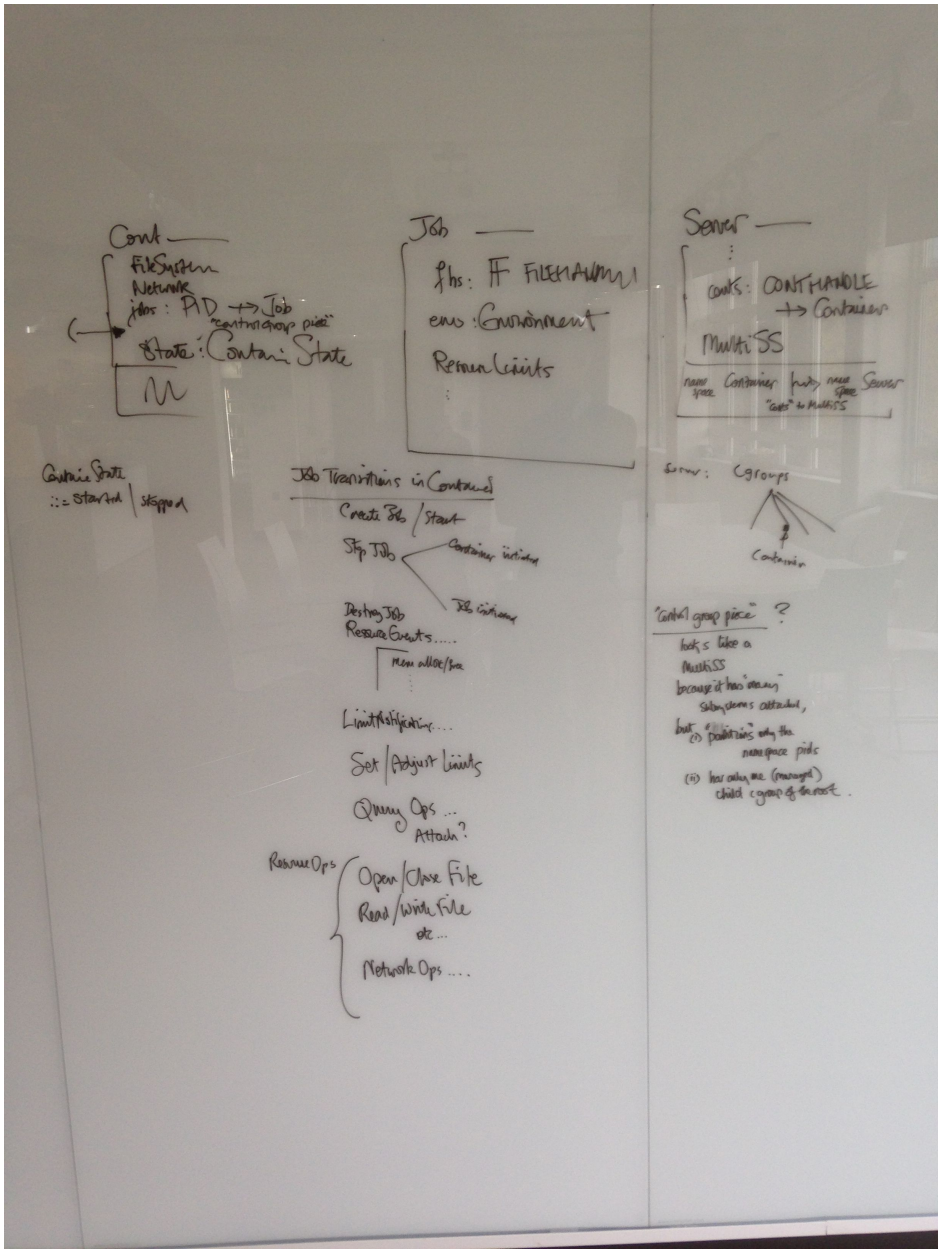
Container

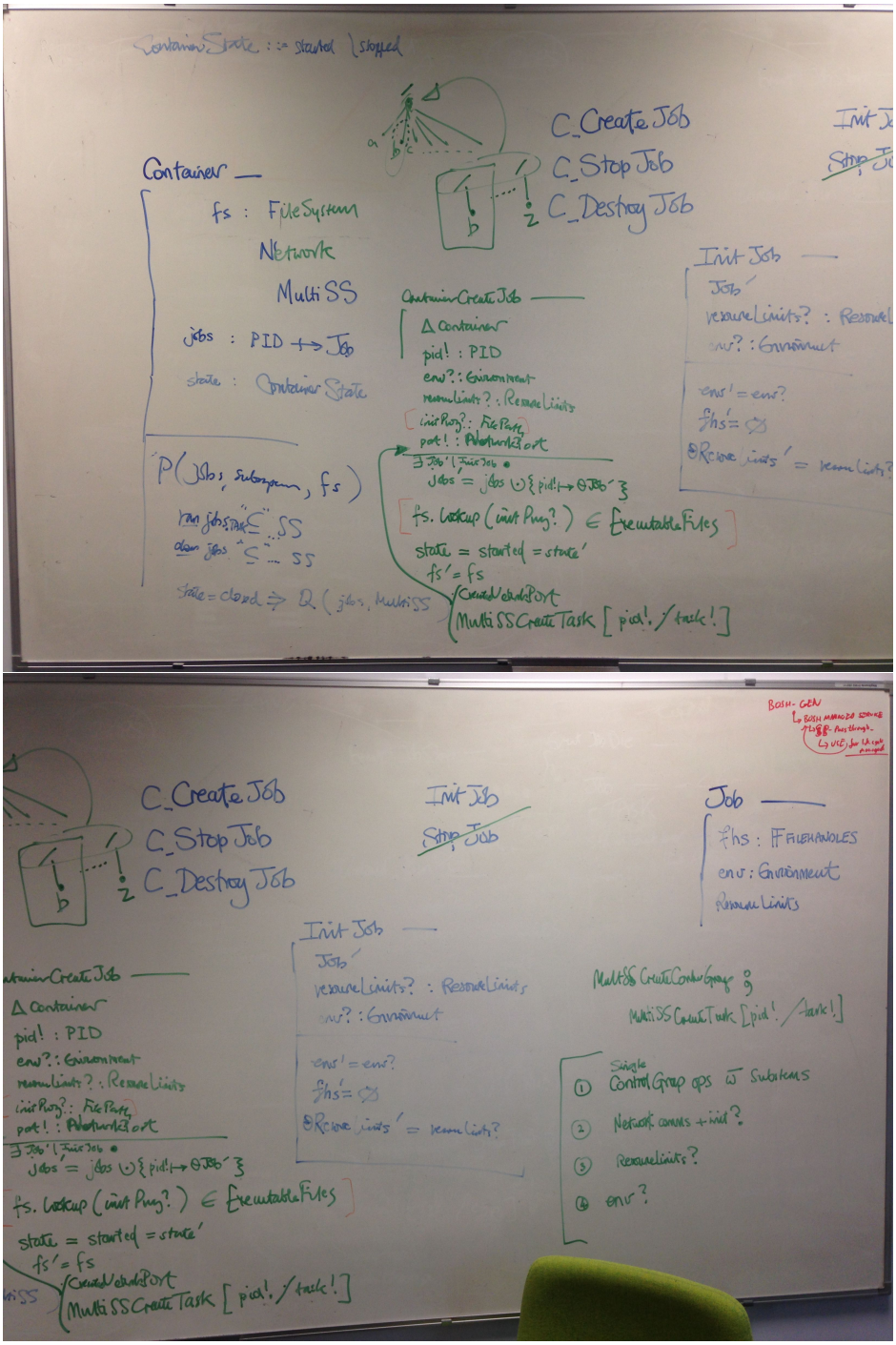
$$fs : FILESYSTEM$$
$$network : NETWORK$$
$$MultiSS$$
$$jobs : PID \pfun Job$$
$$state : ContainerState$$

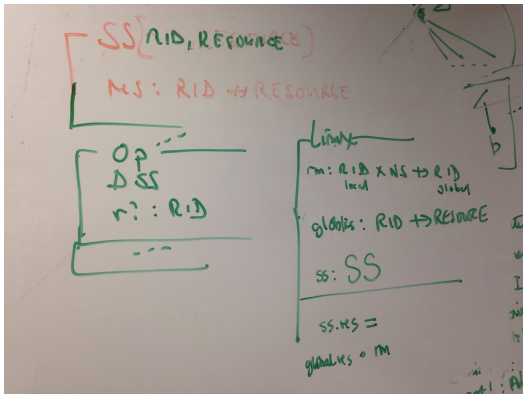
[1] "What *are* containers?" *Jerzy Czaykowski* (adapted)

4 Initial whiteboard stuff




Cont
FileSystem
Network
jobs : PID → Job
"cgroup piece"
State : Contain State
Job
fhs : FF FILEHANDLE
envs : Environment
Resource Limits
Server
conts : CONTHANDLE
→ Container
MultiSS
Container State
::= started | stopped
Job Transitions in Container
Create Job / Start
Stop Job
Destroy Job
Resource Events ....
Limit Modifications ...
Set / Adjust Limits
Query Ops ...
Attach?
Resource Ops
Open / Close File
Read / Write File
etc ...
Network Ops ....
Cgroups
"Control group piece" ?
looks like a
MultiSS
because it has many
subsystems attached,
but (i) "partitions" only the
namespace pids
(ii) has only one (managed)
child cgroup of the root.


ContainerState ::= started | stopped
Container
fs : FileSystem
Network
MultiSS
jobs : PID ⇸ Job
state : ContainerState
C_CreateJob
C_StopJob
C_DestroyJob
InitJob
StopJob
ContainerCreateJob
ΔContainer
pid! : PID
env? : Environment
resourceLimits? : ResourceLimits
initProg? : FilePath
port! : NetworkPort
jobs = jobs ∪ {pid! ↦ θJob'}
fs.Lookup(initProg?) ∈ ExecutableFiles
state = started = state'
fs' = fs
MultiSSCreateTask [pid!, task!]
InitJob
Job'
resourceLimits? : ResourceLimits
env? : Environment
env' = env?
fs' = ∅
θResourceLimits' = resourceLimits?
Job
fs : FileHandles
env : Environment
ResourceLimits
MultiSSCreateControlGroup ;
MultiSSCreateTask [pid!, task!]
Single
1 ControlGroup ops ⊆ Subsystems
2 Network consts + init?
3 ResourceLimits?
4 env?


SS(RID, RESOURCE)
MS : RID ⇀ RESOURCE
Op
Δ SS
r? : RID
Library
rm : RID × NS ⇀ RID
local
global
globals : RID ⇀ RESOURCE
ss : SS
ss.MS =
globals ∘ rm

A Z Notation

Numbers:

$\mathbb{N}$ `Natural numbers {0,1,...}`

Propositional logic and the schema calculus:

$\ldots \land \ldots$	`And`	$\langle\!\langle \ldots \rangle\!\rangle$	`Free type injection`
$\ldots \lor \ldots$	`Or`	$[\ldots]$	`Given sets`
$\ldots \Rightarrow \ldots$	`Implies`	$', ?, !_{0 \ldots 9}$	`Schema decorations`
$\forall .. \mid .. \bullet ..$	`For all`	$\ldots \vdash \ldots$	`theorem`
$\exists .. \mid .. \bullet ..$	`There exists`	$\theta \ldots$	`Binding formation`
$\ldots \setminus \ldots$	`Hiding`	$\lambda \ldots$	`Function definition`
$\ldots \mathrel{\widehat=} \ldots$	`Schema definition`	$\mu \ldots$	`Mu-expression`
$\ldots == \ldots$	`Abbreviation`	$\Delta \ldots$	`State change`
$\ldots ::= \ldots \mid \ldots$	`Free type definition`	$\Xi \ldots$	`Invariant state change`

Sets and sequences:

$\{\ldots\}$	`Set`	$\ldots \setminus \ldots$	`Set difference`
$\{.. \mid .. \bullet ..\}$	`Set comprehension`	$\bigcup \ldots$	`Distributed union`
$\mathbb{P} \ldots$	`Set of subsets of`	$\# \ldots$	`Cardinality`
$\varnothing$	`Empty set`	$\ldots \subseteq \ldots$	`Subset`
$\ldots \times \ldots$	`Cartesian product`	$\ldots \subset \ldots$	`Proper subset`
$\ldots \in \ldots$	`Set membership`	$\ldots$ partition $\ldots$	`Set partition`
$\ldots \notin \ldots$	`Set non-membership`	seq	`Sequences`
$\ldots \cup \ldots$	`Union`	$\langle \ldots \rangle$	`Sequence`
$\ldots \cap \ldots$	`Intersection`	disjoint $\ldots$	`Disjoint sequence of sets`

Functions and relations:

$\ldots \leftrightarrow \ldots$	`Relation`	$\ldots^*$	`Reflexive-transitive`
$\ldots \nrightarrow \ldots$	`Partial function`		`closure`
$\ldots \rightarrow \ldots$	`Total function`	$\ldots (\!\mid \ldots \mid\!)$	`Relational image`
$\ldots \rightarrowtail \ldots$	`Partial injection`	$\ldots \oplus \ldots$	`Functional overriding`
$\ldots \rightarrowtail \ldots$	`Injection`	$\ldots \lhd \ldots$	`Domain restriction`
dom $\ldots$	`Domain`	$\ldots \rhd \ldots$	`Range restriction`
ran $\ldots$	`Range`	$\ldots \lhd \ldots$	`Domain subtraction`
$\ldots \mapsto \ldots$	`maplet`	$\ldots \rhd \ldots$	`Range subtraction`
$\ldots^\sim$	`Relational inverse`		

Axiomatic descriptions:

Declarations
Predicates

Schema definitions:

SchemaName

Declaration
Predicates

B References

[1] Various authors, *Warden github repository*, `https://github.com/cloudfoundry/warden`.

[2] Various authors, *Garden github repository*, `https://github.com/pivotal-cf-experimental/garden`.

[3] Various authors *libcgroup*, `http://libcg.sourceforge.net/html/index.html`.

[4] Paul Menage, Paul Jackson and Christoph Lameter, *CGROUPS*, `https://www.kernel.org/doc/Documentation/cgroups/cgroups.txt`, 2004-2006.

[5] Linus Torvalds, *et al*, *Linux kernel source tree*, `https://github.com/torvalds/linux`.

[6] Various authors, *Memory Resource Controller*, `https://www.kernel.org/doc/Documentation/cgroups/memory.txt`.

[7] Kamezawa Hiroyuki, *et al*, *NOOP cgroup subsystem*, `http://thread.gmane.org/gmane.linux.kernel/777763`.

[8] Martin Prpič, Rüdiger Landmann, and Douglas Silas, *Red Hat Enterprise Linux 6.5 GA: Resource Management Guide*, `https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/pdf/Resource_Management_Guide/Red_Hat_Enterprise_Linux-6-Resource_Management_Guide-en-US.pdf`.

[9] Bertrand Meyer, *On Formalism In Specifications*, IEEE Software, Vol. 2(1), 1985.