Real-world challenges in migration from traditional ILS to RDF

Rurik Thomas Greenall / @brinxmat

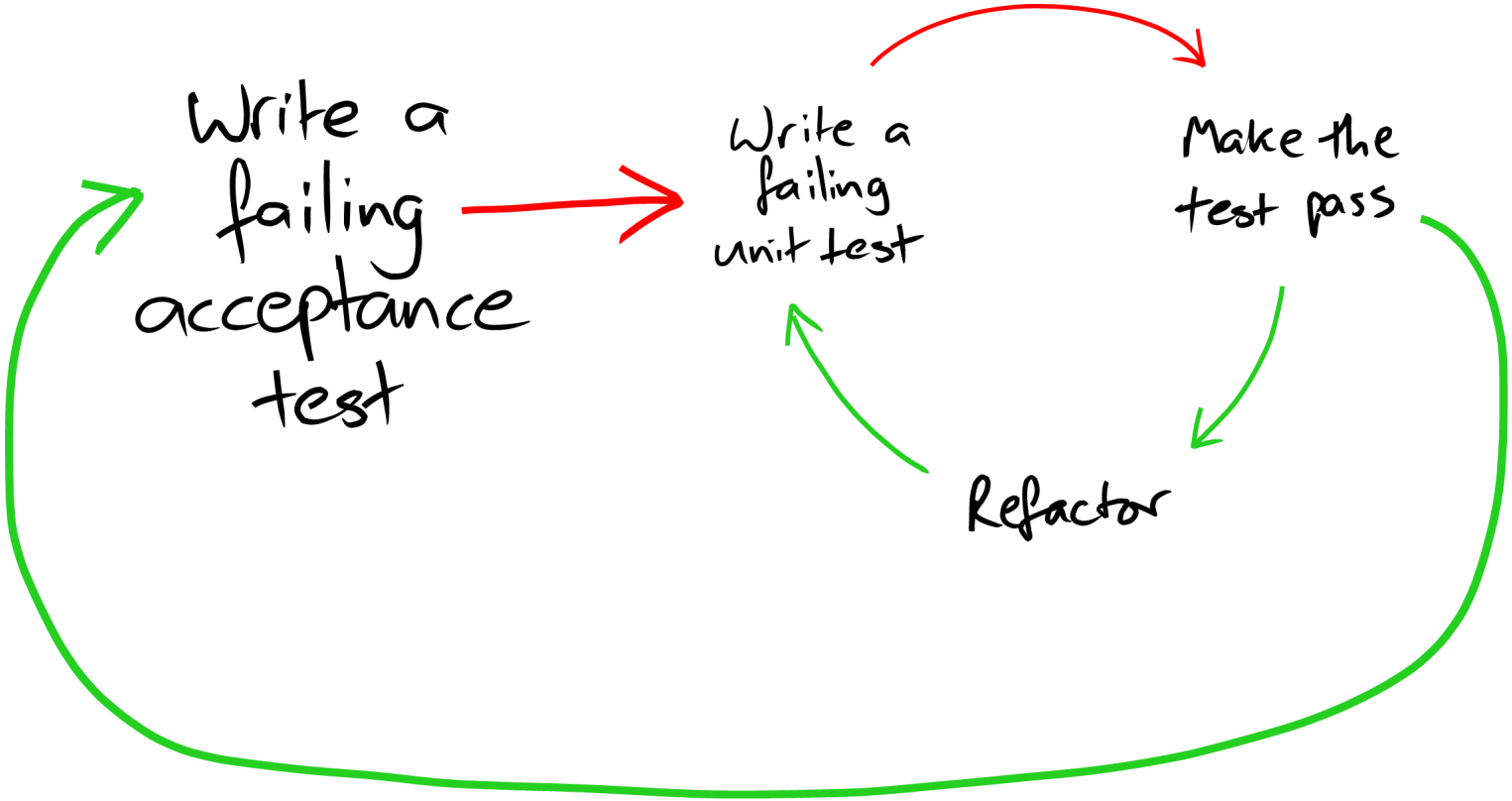
DEICHMANSKE BIBLIOTEK

computas

I will be irreverent/rude &
may make fun of system vendors

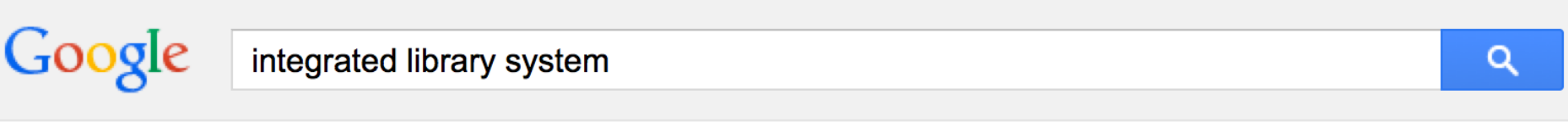
You want code?

https://github.com/digibib/ls.ext/


Write a failing acceptance test
Write a failing unit test
Make the test pass
Refactor

Focus your ire

nonsense cataloguing rules
navel-gazing staff
inept, in-the-rut vendors
hateful punctuation
fatuous domain leadership
ridiculous systems
pointless routines
things that just don't work



Web Images News Videos Maps More Search tools

About 14,800,000 results (0.69 seconds)

An **integrated library system** (ILS), also known as a **library** management **system** (LMS), is an enterprise resource planning **system** for a **library**, used to track items owned, orders made, bills paid, and patrons who have borrowed.

Integrated library system - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Integrated_library_system**

More about Integrated library system

Feedback

Integrated library system - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Integrated_library_system**
An **integrated library system** (ILS), also known as a **library** management **system** (LMS), is an enterprise resource planning **system** for a **library**, used to track items owned, orders made, bills paid, and patrons who have borrowed.
History - See also - References - Further reading

Koha - Open Source ILS - Integrated Library System
www.koha.org/
Koha is the first open-source **Integrated Library System** (ILS). In use worldwide, its development is steered by a growing community of libraries collaborating to ...

The Koha library

No.

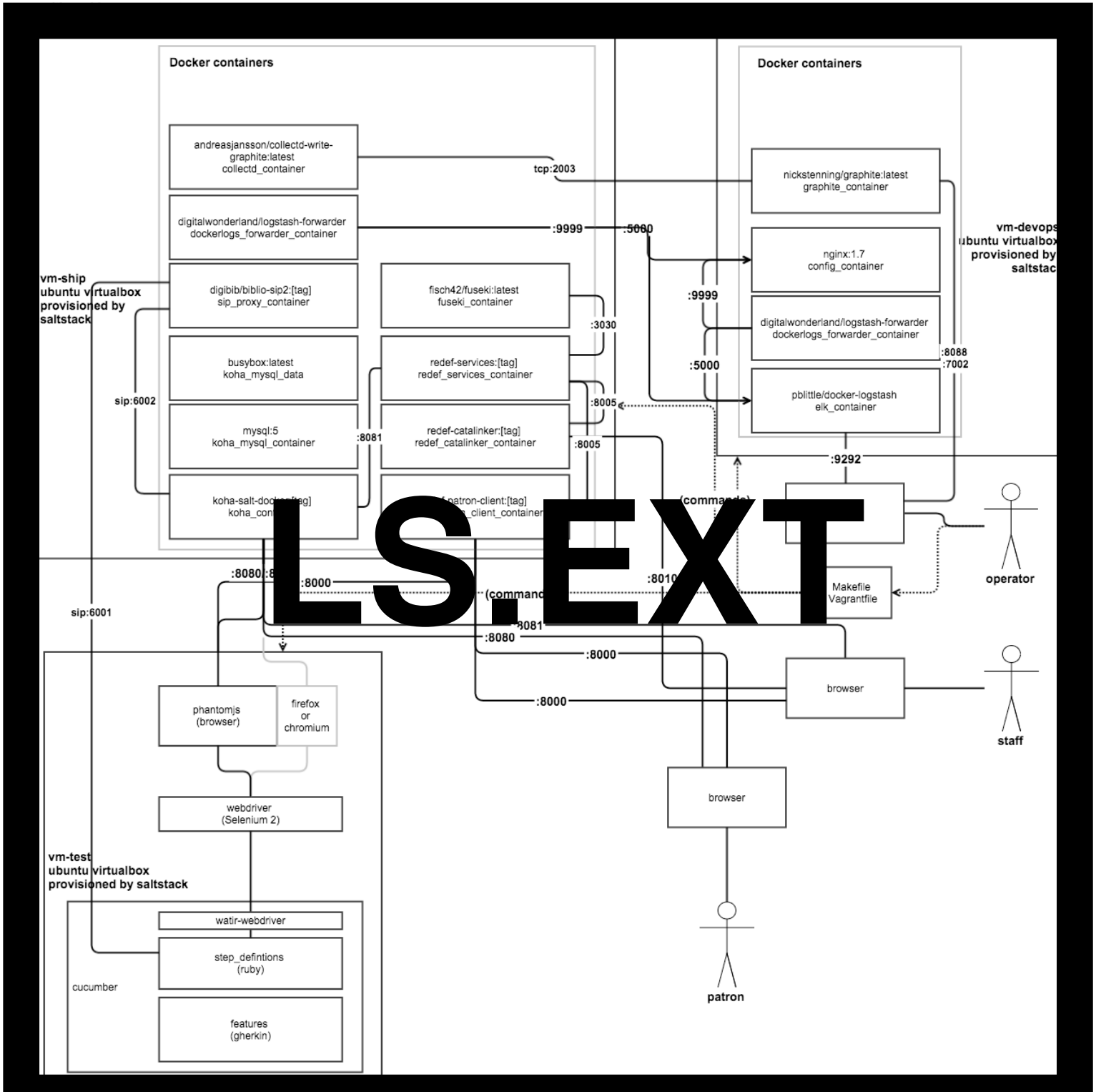
"Jack of all trades, master of none,
Certainly better than a master of one"

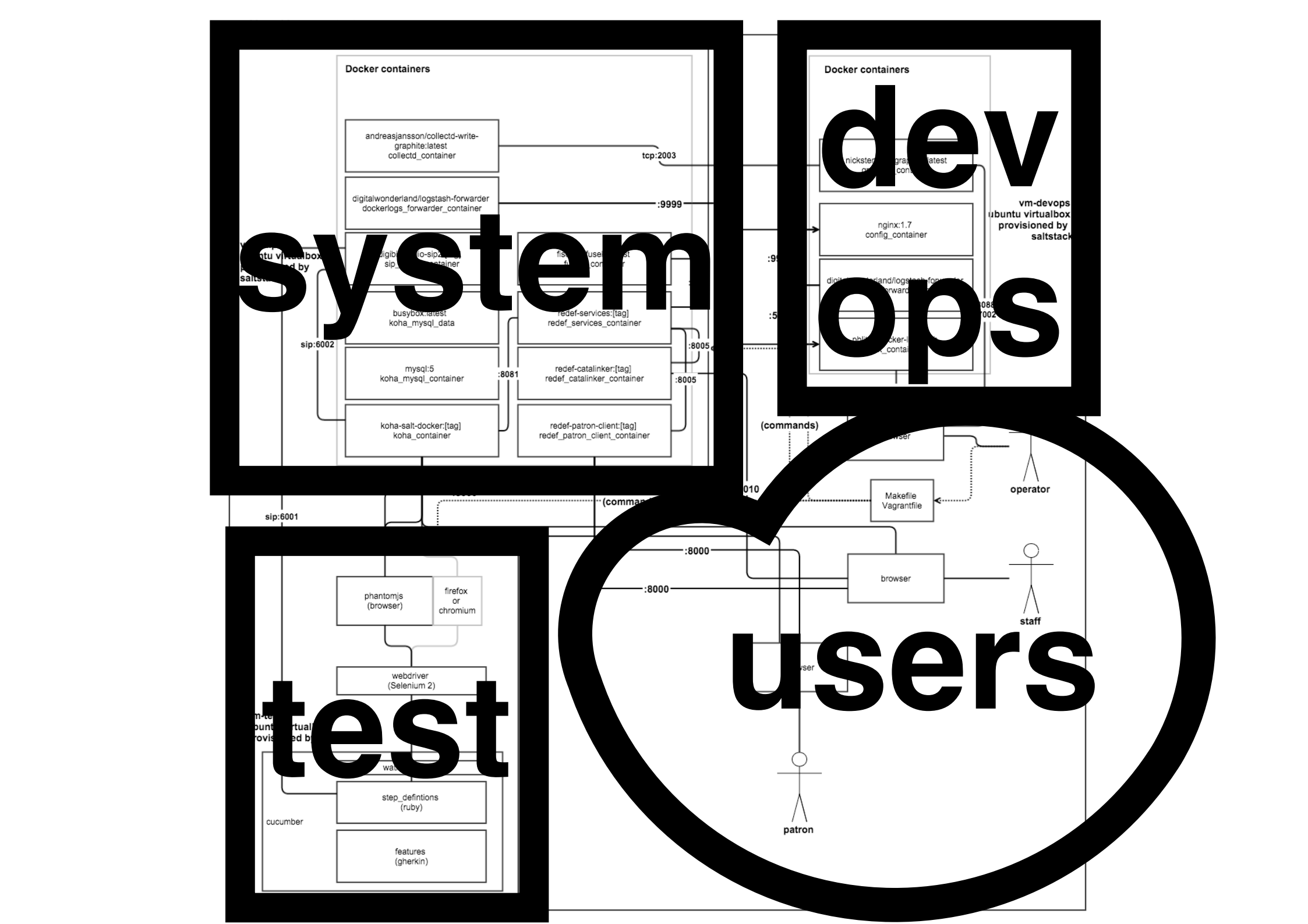
No.

"**This is the Unix philosophy:**
Write programs that do one thing and do it well.
Write programs to work together."

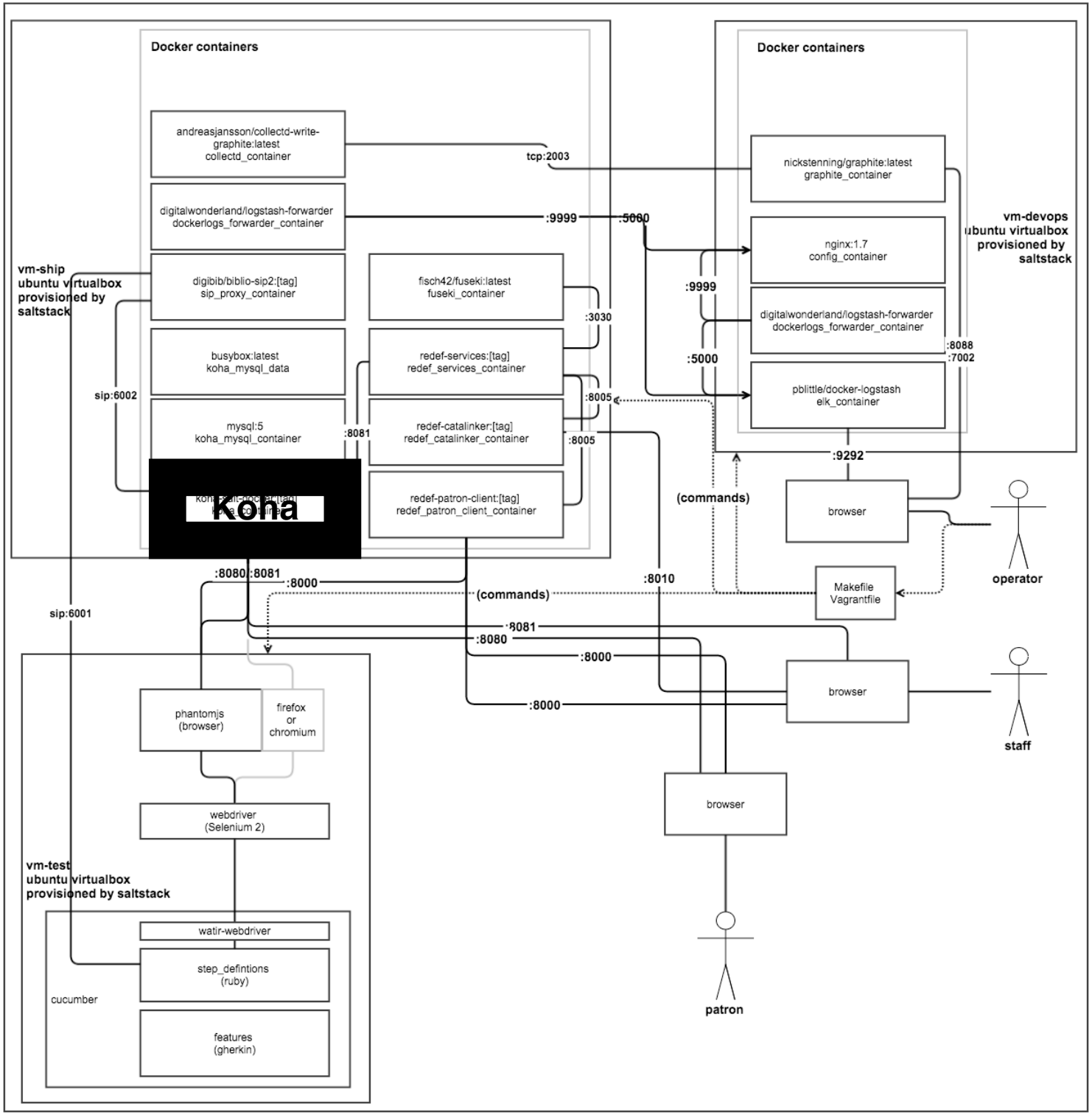
Douglas McIlroy

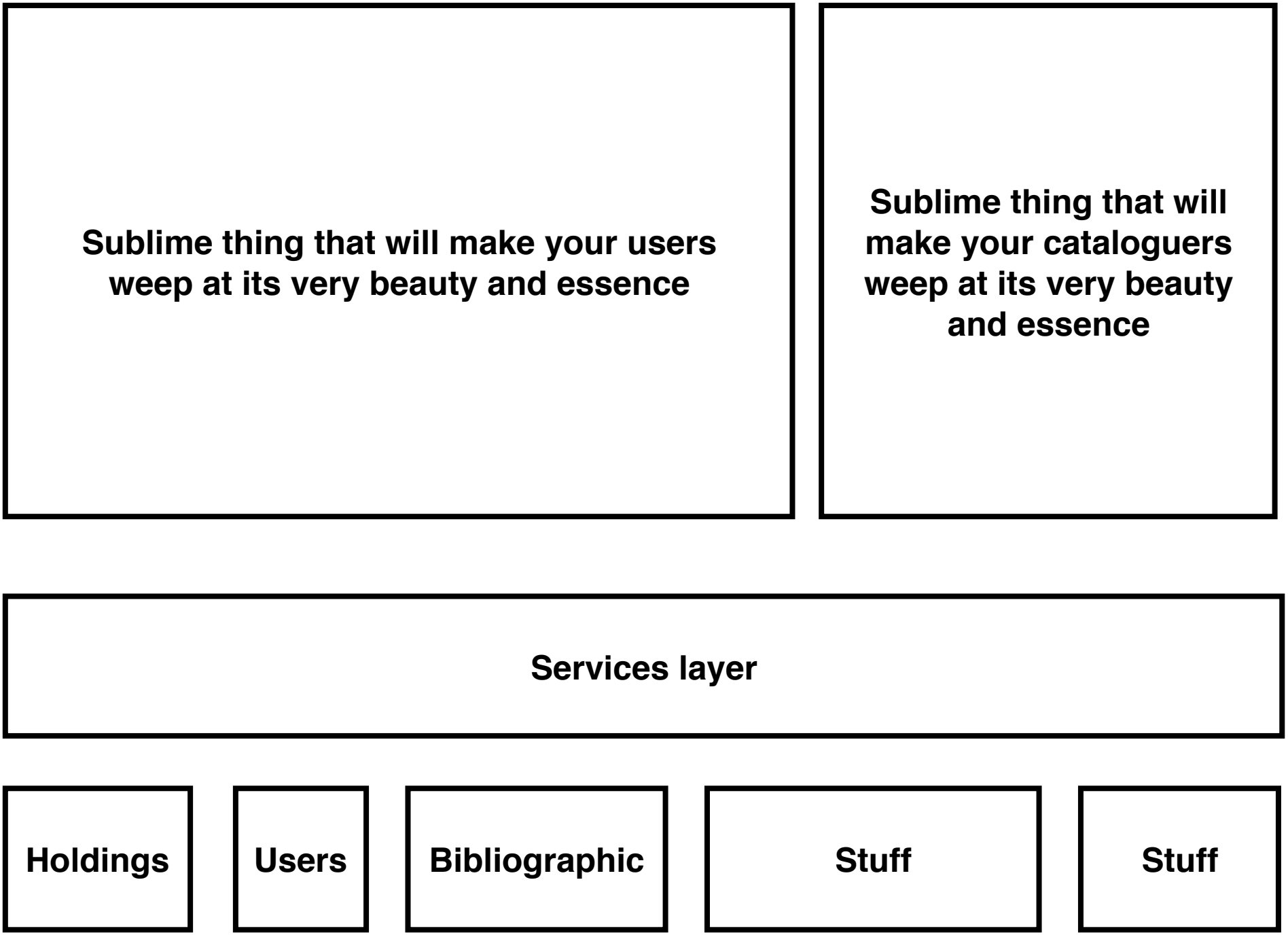
Peter H. Salus. A Quarter-Century of Unix. Addison-Wesley. 1994. ISBN 0-201-54777-5. [p. 52]

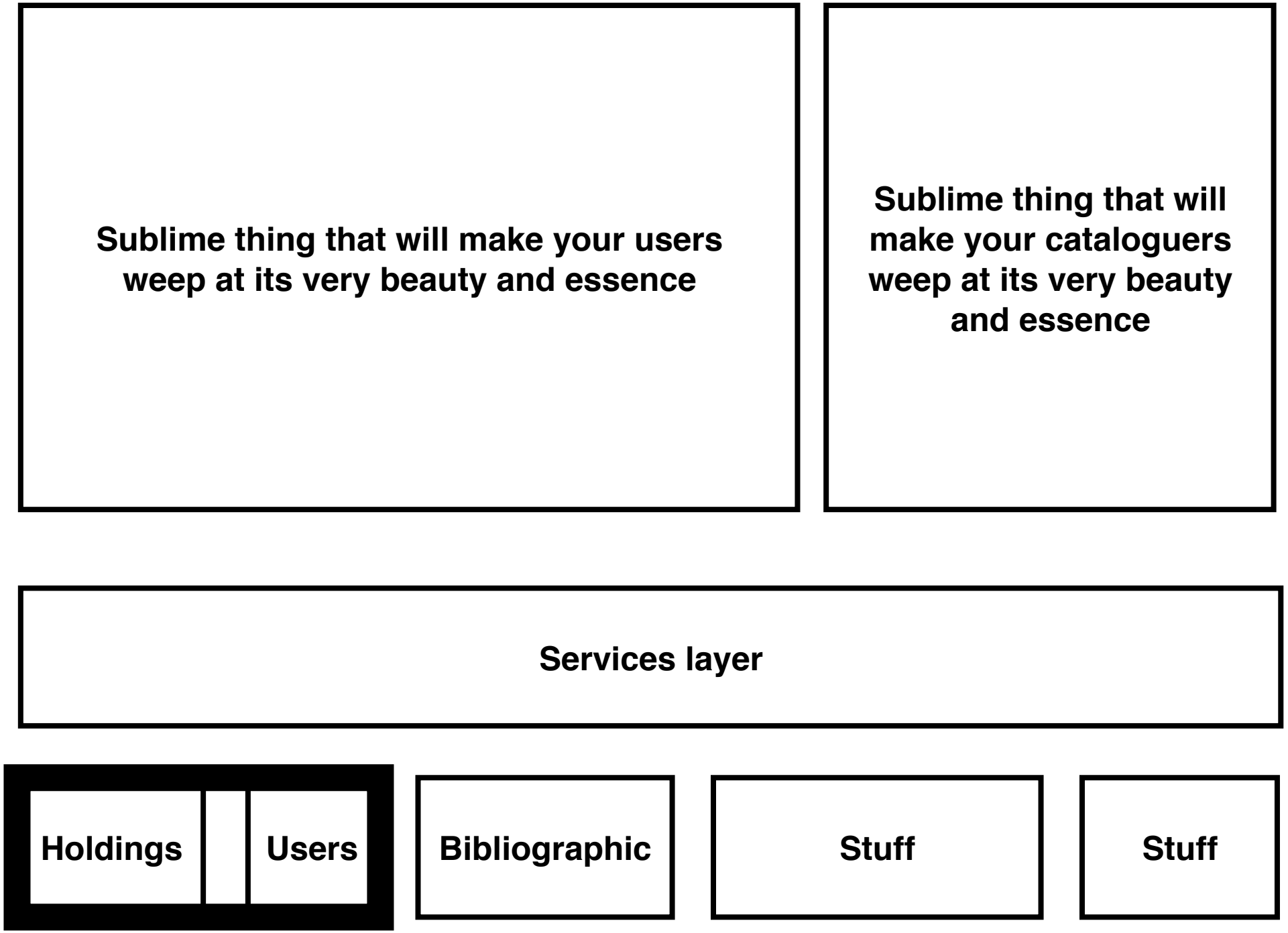

Docker containers
andreasjansson/collectd-write-graphite:latest
collectd_container
digitalwonderland/logstash-forwarder
dockerlogs_forwarder_container
digibib/biblio-sip2:[tag]
sip_proxy_container
busybox:latest
koha_mysql_data
mysql:5
koha_mysql_container
fisch42/fuseki:latest
fuseki_container
redef-services:[tag]
redef_services_container
redef-catalinker:[tag]
redef_catalinker_container
tcp:2003
:9999
:5000
:3030
:8005
:8081
sip:6002
vm-ship
ubuntu virtualbox
provisioned by
saltstack
Docker containers
nickstenning/graphite:latest
graphite_container
nginx:1.7
config_container
digitalwonderland/logstash-forwarder
dockerlogs_forwarder_container
pblittle/docker-logstash
elk_container
:8088
:7002
:9292
vm-devops
ubuntu virtualbox
provisioned by
saltstac
operator
Makefile
Vagrantfile
:8080
:8000
:8010
sip:6001
browser
staff
phantomjs
(browser)
firefox
or
chromium
webdriver
(Selenium 2)
vm-test
ubuntu virtualbox
provisioned by saltstack
watir-webdriver
step_defintions
(ruby)
cucumber
features
(gherkin)
patron


system
dev ops
users
test
Docker containers
andreasjansson/collectd-write-graphite:latest
collectd_container
tcp:2003
digitalwonderland/logstash-forwarder
dockerlogs_forwarder_container
:9999
busybox:latest
koha_mysql_data
redef-services:[tag]
redef_services_container
sip:6002
mysql:5
koha_mysql_container
:8081
redef-catalinker:[tag]
redef_catalinker_container
:8005
koha-salt-docker:[tag]
koha_container
redef-patron-client:[tag]
redef_patron_client_container
Docker containers
nginx:1.7
config_container
vm-devops
ubuntu virtualbox
provisioned by
saltstack
(commands)
Makefile
Vagrantfile
operator
sip:6001
:8000
:8000
browser
staff
patron
phantomjs
(browser)
firefox
or
chromium
webdriver
(Selenium 2)
step_defintions
(ruby)
cucumber
features
(gherkin)

multi-machine vagrant setup




Sublime thing that will make your users weep at its very beauty and essence
Sublime thing that will make your cataloguers weep at its very beauty and essence
Services layer
Holdings
Users
Bibliographic
Stuff
Stuff


Sublime thing that will make your users weep at its very beauty and essence
Sublime thing that will make your cataloguers weep at its very beauty and essence
Services layer
Holdings
Users
Bibliographic
Stuff
Stuff

Realization: Users & logistics are a closed world

(if you know some stuff that is known about items or users, then that is what is true)

Realization: Bibliographic data is an open world

(what you know about a resource is only what you know; other people know other relevant things)

RDF is an open world

(you want your bibliographic data in RDF, the other stuff is better stored in a RDBMS)

That isn't to say that other things can't be represented in RDF

(indeed...we do a lot of that)

How are we representing different data?

(easy answer: appropriately...for *our* usage)

Standardization

- Use MARC
- Use BIBFRAME
- Use Linked Data Platform
- Use Linked Data Fragments
- Use ISO XXXX

No.

Our data model

[this slide left intentionally blank]

Reality

- RDF: enough to support use cases like “I can see a work” and “I can see a list of items for a work”
- DB-schema: enough to support use cases like “I can borrow an item” and “I can see statistics report X”, “I can log in as user X with privileges Y”
- Index: enough to support use cases like “I can search for a work/item”

What about...?

- Internal MARC: doesn't really exist except as an abstract *model-of-sorts* for parts of Koha's DB-schema
- External MARC:
  - Input, yes.
  - Output: (no active use case as of yet)
- Internal linked-data model: what we use internally is an evolving, specific ontology for our business case
- External linked data: (no active use case as of yet)

What you do in your system should support your use cases,
not some abstract, idealised standard.

Interacting with others, if that is a use case, requires standards,
but don't let these infect your data model.

I have rambled a while, but here is what I want you to think about:

- The hardest part of creating a new system is knowing what you need; break that down into simple, owned use cases
- The second hardest part is keeping to that plan; you're bound to want to develop according to the standards — but which use case does that support?
- Keep doing things "according to the standards" and you have another FRBR: a nice, twenty year-old idea
- Do things that support your users, not things that support supposed, idealised users

...and finally

most of cataloguing is bollocks
that has nothing to do with reality

"With no data on the facts of catalog use,
the cataloging rules look a lot more like a religious text
than an application of science."

–Karen Coyle

http://listserv.loc.gov/cgi-bin/wa?A2=ind1505&L=bibframe&T=0&P=11707

...don't let that stuff mess up your data