



Enterprise versus Open Source Development Policy

Kostis Kapelonis

Athens Greece, December 2013

What is the best Methodology
for developing software
Applications?

most appropriate

What is the ~~best~~ Methodology
for developing software
Applications?

Enterprise Software

Sale Order

Voucher No. :

Customer Code :

Customer Name :

Address :

Sale Type :

Date :

Store name :

Decode Barcode :

S No	Item Code	Item Description	Qty.	Rate	Amount (a)
1	D11	D.Jing		3.00	3960.00
2	D2	TL Audio C-1		5.00	1250.00
3	D7	JBL T-45		6.00	2142.00
4	E1	Flip Chip		4.00	4360.00
5	E5	PreSonus		8.00	4800.00
6	D8	Phoxtex PH-5		4.00	800.00
7	E10	Plextor PX-01		3.00	1500.00

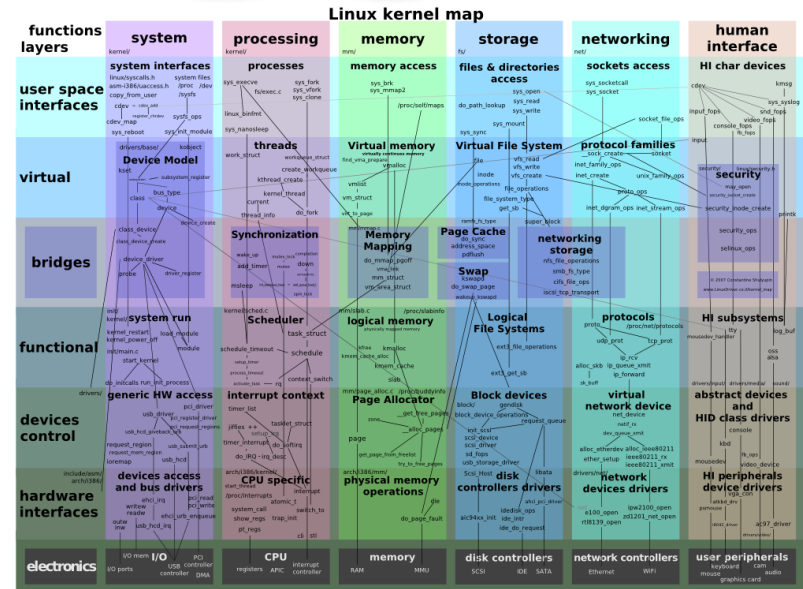
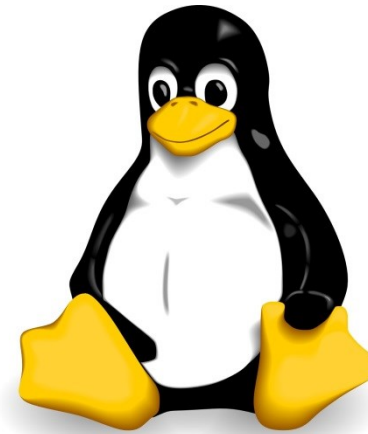
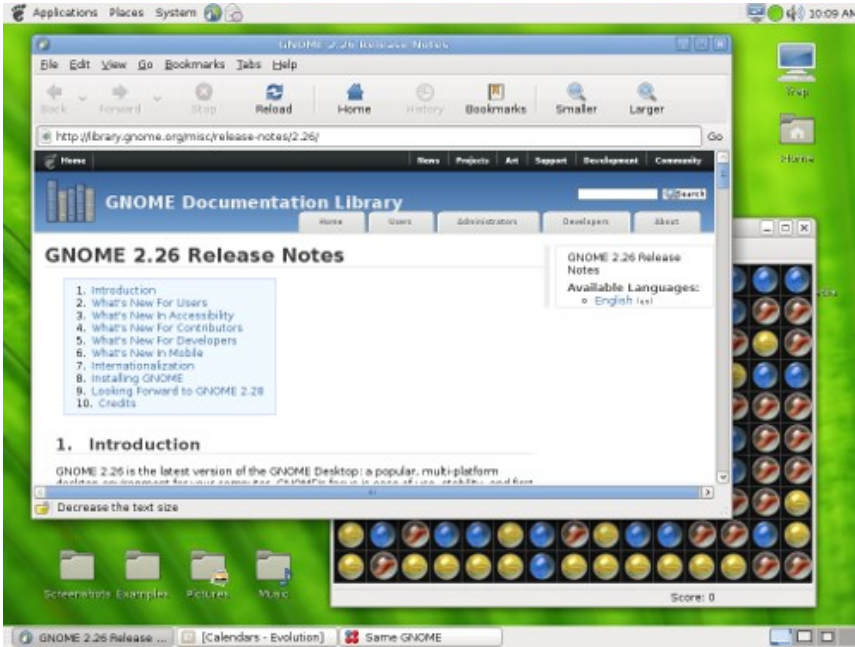
Qty 33 **Total :** 18812.00

Memo :

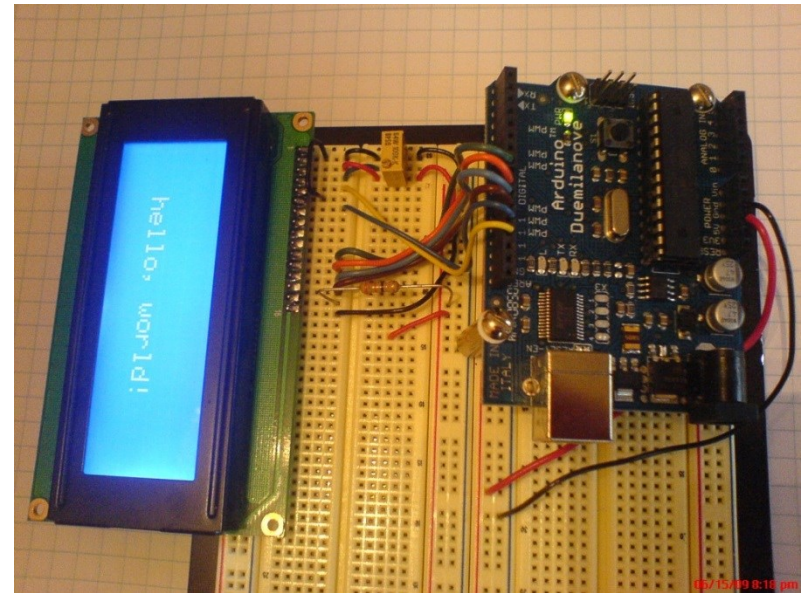
Tax Type	@	Per	Nature	Amount (a)
Central Sales Tax		50.00	Percentage	940.60

Grand Total : 19752.60

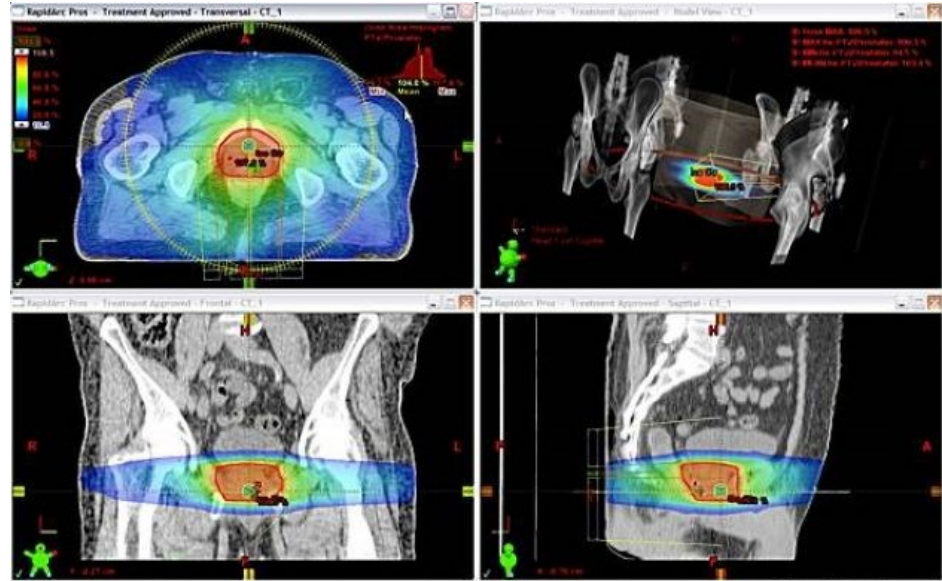
Open Source general purpose software



Embedded software



Medical Software



Liz Hafalia / The Chronicle



Military Software



QNX[®]
QNX SOFTWARE SYSTEMS

VxWorks

Nuclear Reactor Software



CHANNEL ID: ***** NORTH EAST WORKTABLE 17:31:52 2006-Nov-02

Worktable EF SPIRT PT Severing Platform Options User (Ctrl+L)

EF SPIRT OPERATION

HOMING POSITION
Offset X: 0.00 in. Y: 0.00 in.

PERMISSIVES
FULL PT SEVER
HOME TO EF
REMOVE SP
INSERT SP

AUTOMATIC SEQUENCE CONTROL
SEQUENCE SELECT: please Select One

AUTO SEQUENCE STATUS
Step: Sequence complete
Sequence: Shield Plug Removal
Series: PT Severing

SLEEVE POSITION
Z Position: 0.00 in.

SEMI-AUTO CONTROL

Drive	Permissive	Relative Position	Speed	Torque/Force	Brakes
SLEEVE	BACK FORWARD	***** in.	***** in./sec.	***** lbf	ON OFF
AXIAL	BACK FORWARD	***** in.	***** in./sec.	***** lbf	ON OFF
ROTARY	CCW CW	***** deg.	***** deg./sec.	***** lb-ft	
LOCKING LUG	BACK FORWARD	***** in.	***** in./sec.	***** lbf	ON OFF
TORQUE PIN	BACK FORWARD	***** in.	***** in./sec.	***** lbf	ON OFF

STATUS INFORMATION

Drive	Status	Home	Position	Speed	Current	Torque/Force
Sleeve	*****	○	0.00 in.	0.00 in./sec.	0.00 Amp.	0.00 lbf
Axial	*****	○	0.00 in.	0.00 in./sec.	0.00 Amp.	0.00 lbf
Rotary	*****	○	0.00 deg.	0.00 deg./sec.	0.00 Amp.	0.00 lb-ft
Locking Lug	*****	○	0.00 in.	0.00 in./sec.	0.00 Amp.	0.00 lbf
Torque Pin	*****	○	0.00 in.	0.00 in./sec.	0.00 Amp.	0.00 lbf

WORK TABLE STATUS

Position	Speed
X 0.00 in.	0.00 in./min.
Y 0.00 in.	0.00 in./min.

Worktable Indexed to: *****

PLC MODE: REMOTE SECURITY CLASS: MAINTENANCE

The Team

Roles and privileges

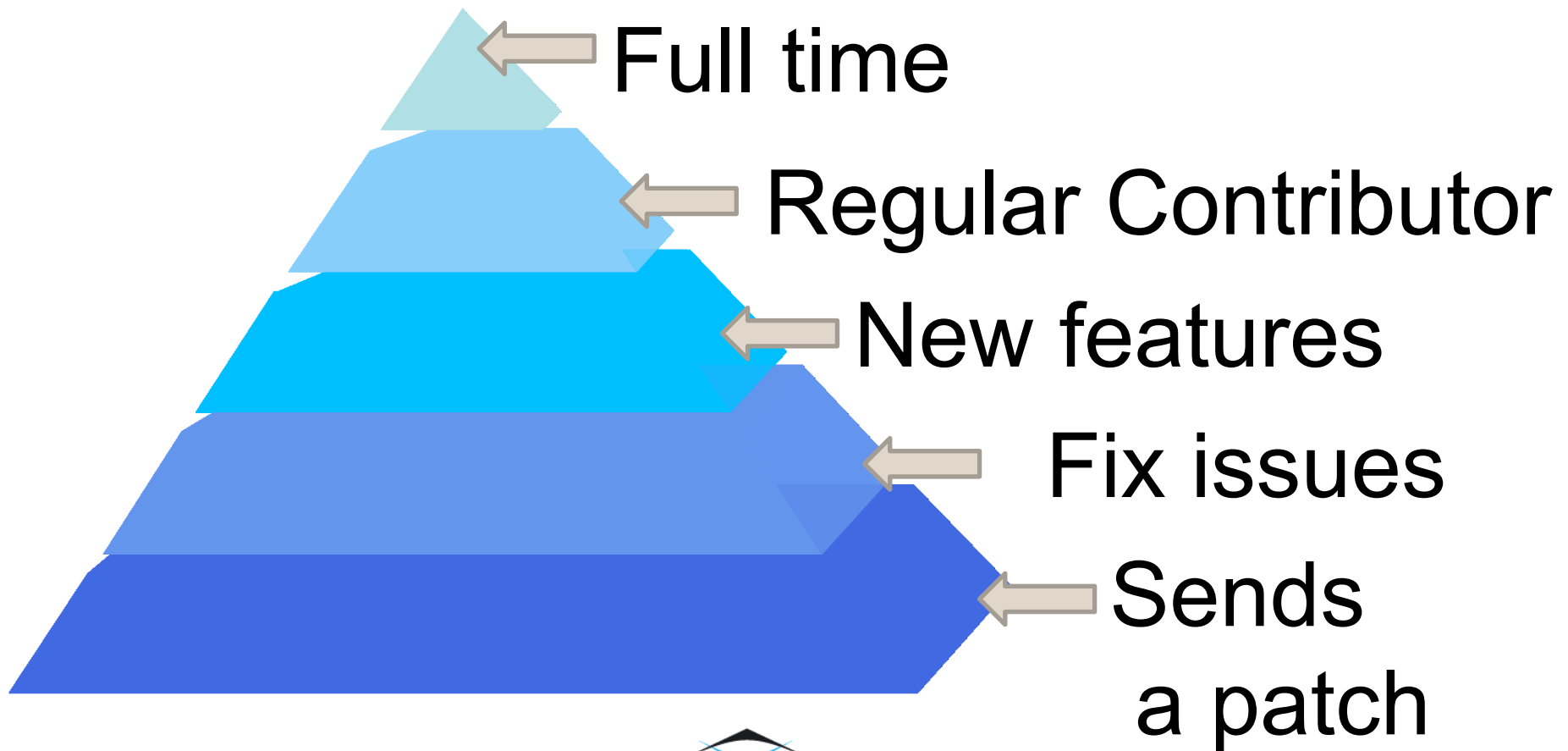
Open source roles (privileges)

- Client is everybody
- Developer could be anybody



Open source roles (involvement)

- Different level of involvement
- Author, Committer, Lieutenant, Contributor, Dictator



Enterprise Team



Unified Team

Features

Who approves what

Open source stakeholders

- There is no “hard” timetable
- There is no “hard” set of features

Features



“Client”



“Developer”

Features



Open source forks



MySQL vs MariaDB

Emacs vs XEmacs

Debian vs Ubuntu

WebKit vs KHTML

XFree vs X.org

LibreOffice vs OpenOffice

Xwiki vs FossWiki

Feature wars

Enterprise stakeholders

- Stakeholder is the client. Ultimate power on features
- Client proposes features, clients approves features (same entity)



Features



Client

Developer

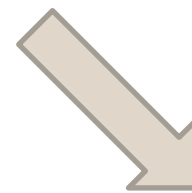
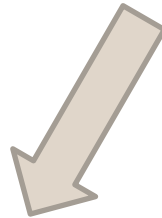
Changesets

Choosing The base line

Enterprise stakeholders

- Opensource loves the latest and greatest version (constant change)
- Enterprise loves backwards (and forwards) compatibility

A Patch comes
in for an older version



“Please verify on
latest version”

Regression
Testing

Other stuff

Different development
mindset

- Release early, release often
- It is done when it is done
- Show me the code
- If it compiles, ship it
- Users are lusers
- Cathedral versus Bazaar

Biggest difference

Trust on people

Enterprise stakeholders

- Opensource must deal with the crowd (also malicious commits)
- Enterprise has controlled teams

Open Source



Unknown

Enterprise



Handpicked

Border Control

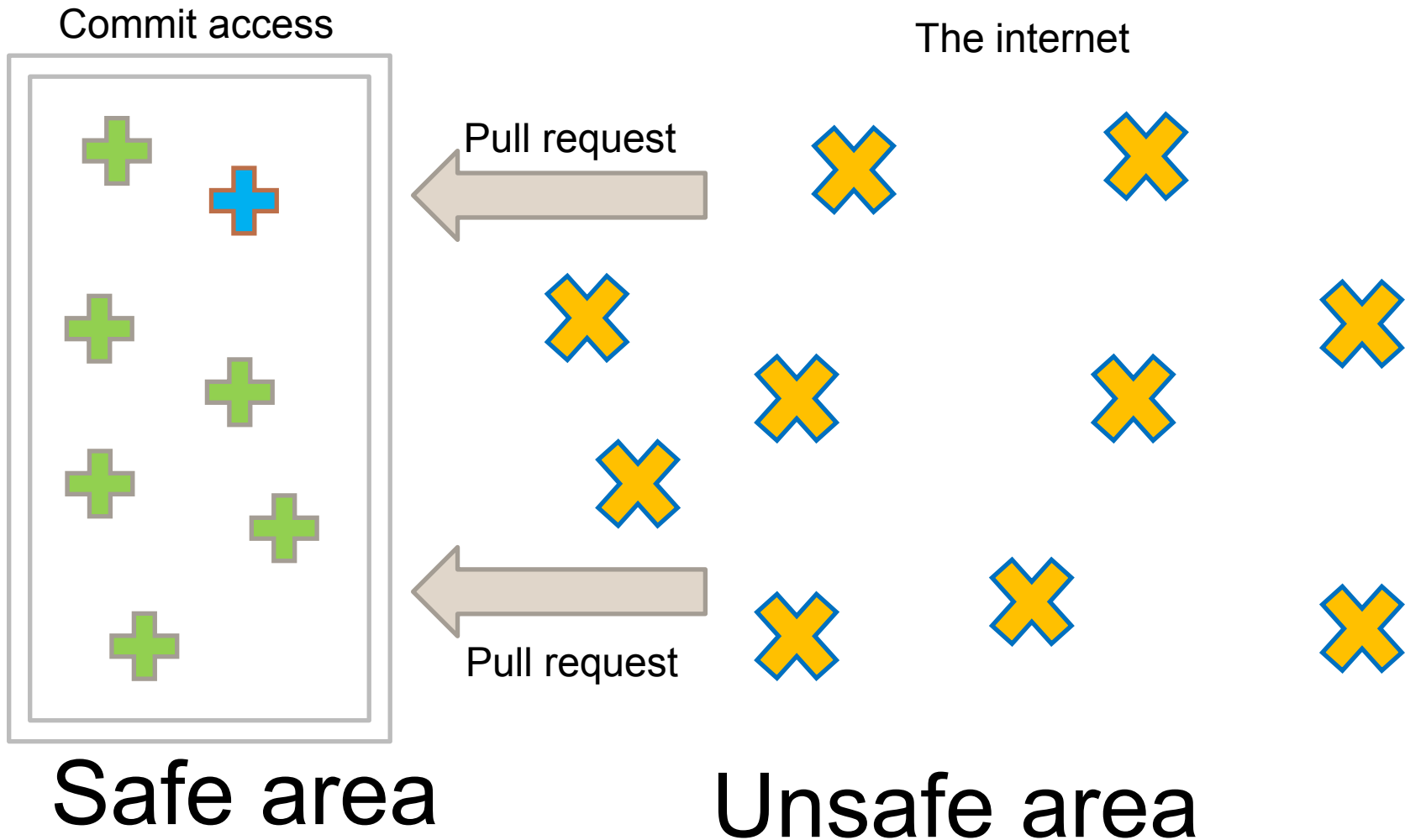
Trust no one (until they
get in)

Enterprise versus Open Source

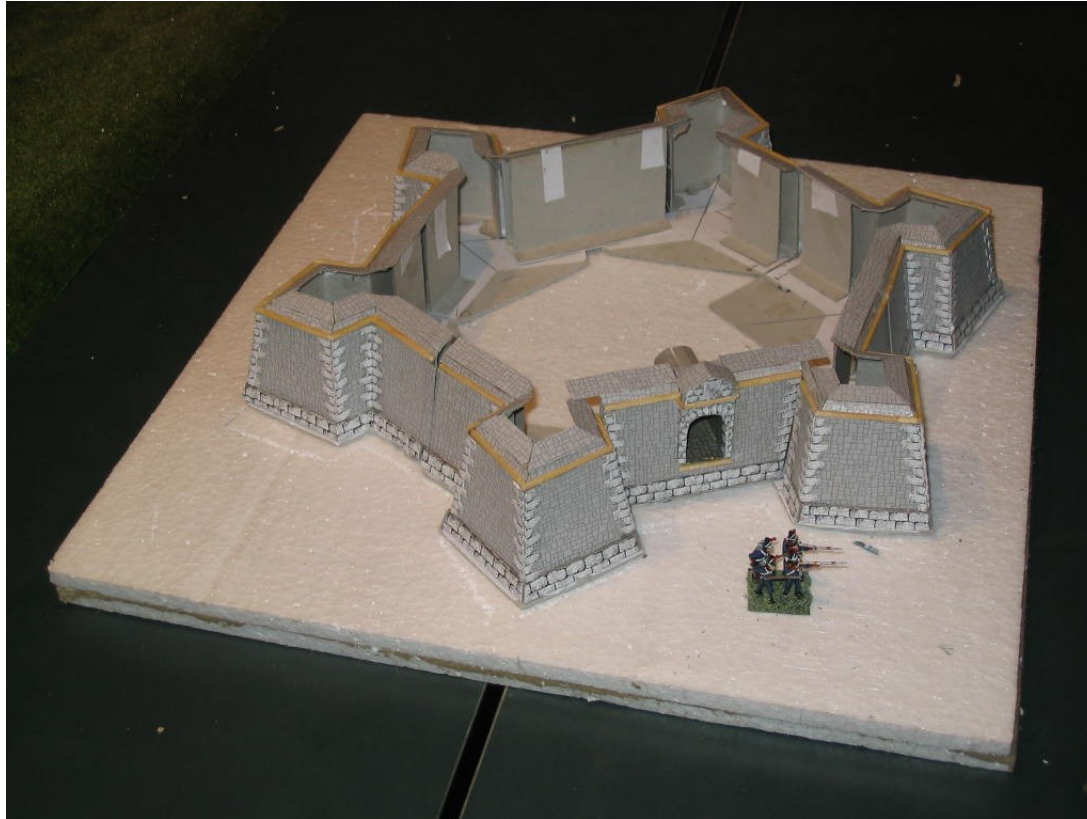


Either you are trusted or not

Enterprise versus Open Source



Enterprise versus Open Source



One big barrier (In or out)

A different approach

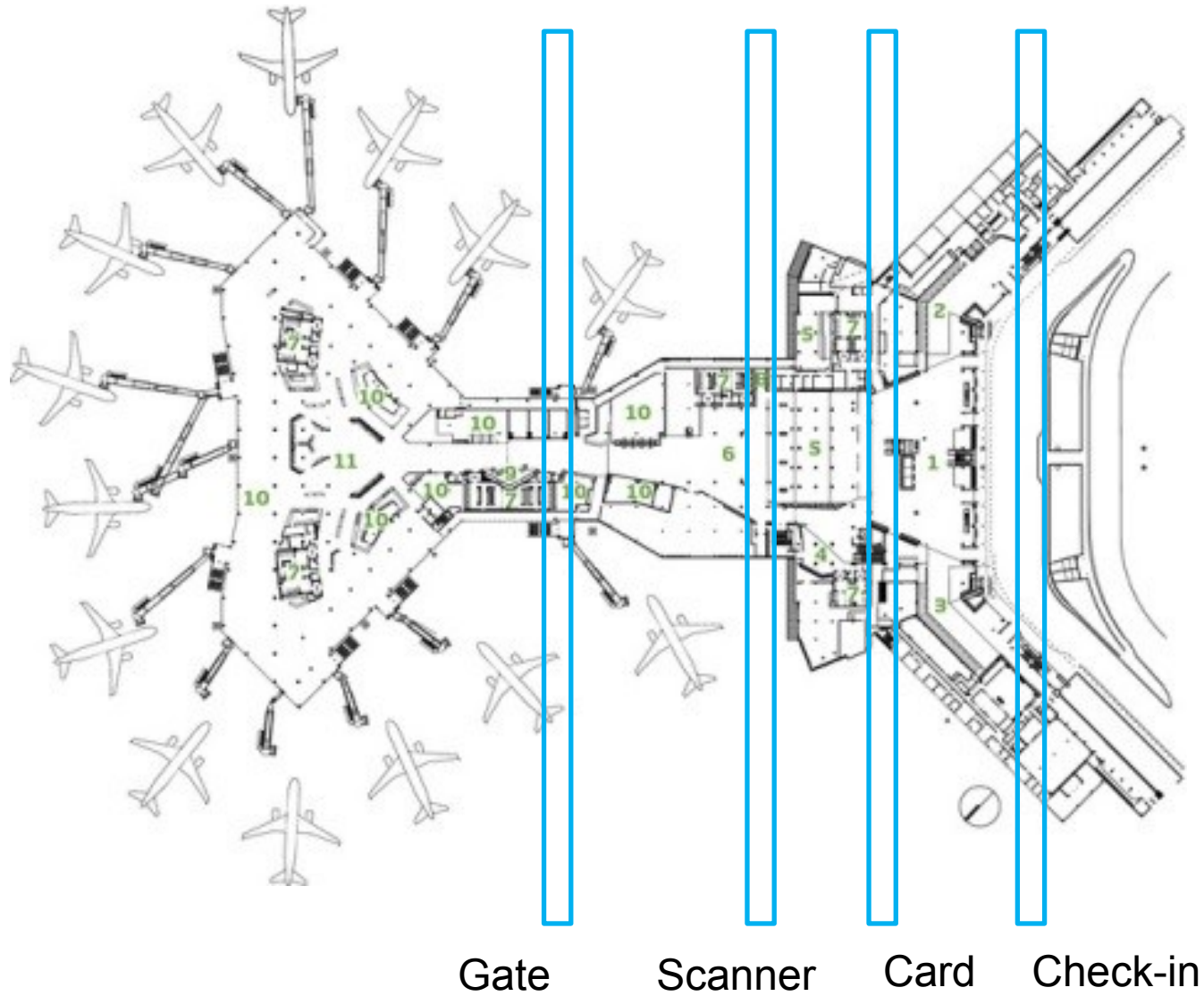
Layers of trust of code
(not people)

Enterprise versus Open Source



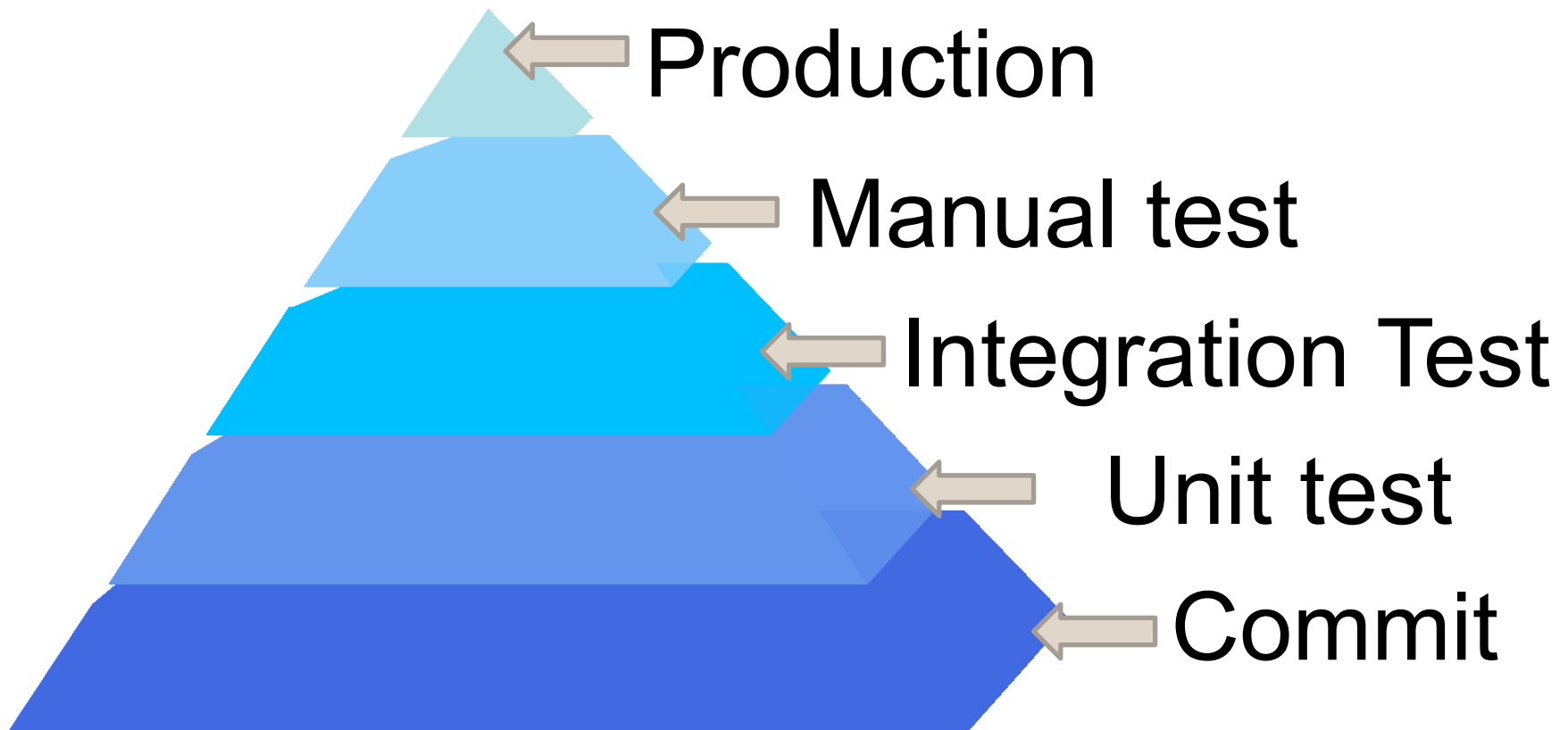
The airport approach

Enterprise versus Open Source



Code quality certainty

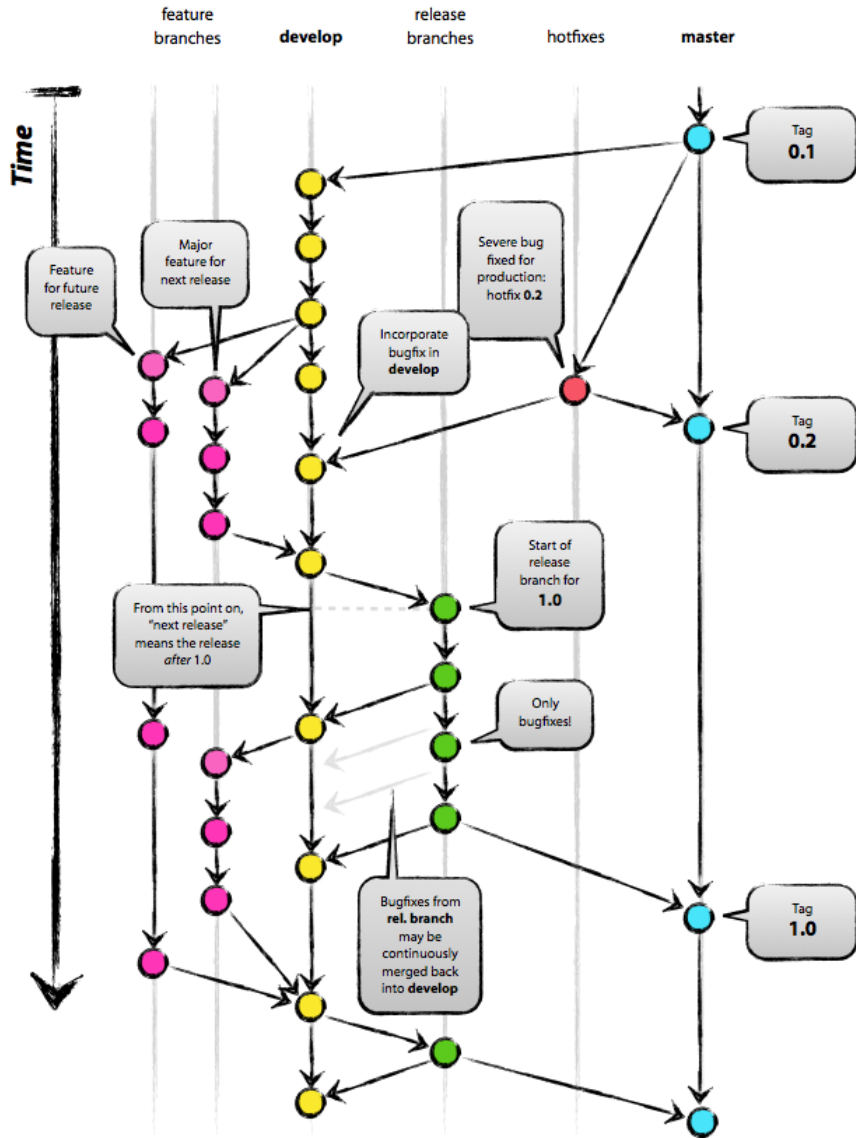
- Different level of code trust
- Code starts as unsafe and reaches production status



In action

Tools and support

Branch model



A successful Git model

Layers of trust

Development



Fully automated (everybody)

Staging



Semi- Automated (Testers)

Production



Manual (PM)

All commits go on dev. Should compile.

Test environments use staging.

Tags come from Production

Build jobs

- Developer build (almost every commit) – unit tests
- Integration tests (every 30 minutes)
- Sonar build (once a day)
- Promotion job (for testers)
- Staging environments (multiple)
- Release jobs
- Completely automated for day usage



Build server

Developer responsibilities

- Commit on “dev” branch. Should run unit tests first locally
- Pull/Merge as needed freely (but only on dev branch)
- Multiple developers can work on same feature branch
- Monitor build status and fix broken builds
- All branches are remote (no local code)
- “Code that is not committed does not exist”
- Code review (before merging with “dev”)
- Commit messages have JIRA number
- Feature branches can be long or short
- For long lived – pull from “dev” daily



Developer

QA responsibilities

- Runs tests on “Frozen” staging
- Promote a build from dev to staging (build job)
- Approve staging to PM (so that it can be released)
- During releases there is no promotion (hot fixes go on staging)
- Regression testing with maintenance branch



Tester

PM responsibilities

- Ranks features
- Approves features (by the client)
- Approves late release stages
- Approves Tags and releases from production branch



Project Manager

Code review of (merged) feature

#72 **MERGED** IUCVI-839/Readability → master

IUCVI-839/Readability

Overview Diff Commits

Changed files

- iuclid6-comm-common/src/test/java/eu/echa/iuclid6/...
 - SectionNodeDefinitionSerializerTest.java
- iuclid6-comm-server/src/test/java/eu/echa/iuclid6/con...
 - SectionTreeResourceIT.java
- iuclid6-core-definitions/src/main/java/eu/echa/iuclid6...
 - coredefinitions
 - document
 - endpoint
 - studyrecord
 - Acute ToxicityDermalDefinition.java
 - Acute ToxicityInhalationDefinition.java
 - Acute ToxicityOralDefinition.java
 - Acute ToxicityOtherRoutesDefinitio
 - AdditionalPhysicoChemicalDefinitio

iuclid6-comm-common / src / test / java / eu / echa / iuclid6 / c

SectionNodeDefinitionSerializerTest.java **MODIFIED**

142	142	.append("\\\" +
143	143	.append("\\\" +
144	144	.append("\\\" +
145	145	.append("\\\" +
146	146	.append("\\\" +
147	147	.append("\\\" +
148	148	.append("\\\" +
149	149	.append("}").app
150	150	.append("\\\" +
151	151	.append("\\\" +
- 152		.append("\\\" +
	+ 152	.append("\\\" +
153	153	.append("\\\" +
154	154	.append("}, {"
155	155	.append("\\\" +
- 156		.append("\\\" +
	+ 156	.append("\\\" +
157	157	.append("\\\" +
158	158	.append("}]")
159	159	.append("}");

JIRA - GIT



IUCLID6 / IUCVI-765

ST3-I3.2: Creation of an entity from Browse window

[Edit](#) [Assign](#) [Comment](#) [More Actions](#) [Submit \(Preparation\)](#) [Submit \(Work\)](#) [Workflow](#)

Should the 'New' option be available?
Should there be a validation that a user cannot create an entity unless he has at least one LE assigned to him?

Issue Links

duplicates	IUCVI-1229 Create new entity from browse window	
is blocked by	IUCVI-877 Validation errors should stop all exit/logout/change document requests	

Activity

[All](#) [Comments](#) [Work Log](#) [Worklogs +](#) [Time Spent](#) [History](#) [Activity](#) [Workflow Diagram](#) [Transitions Summary](#) [Subversion](#) [Source](#)



Konstantinos Kapelonis Today 19:05

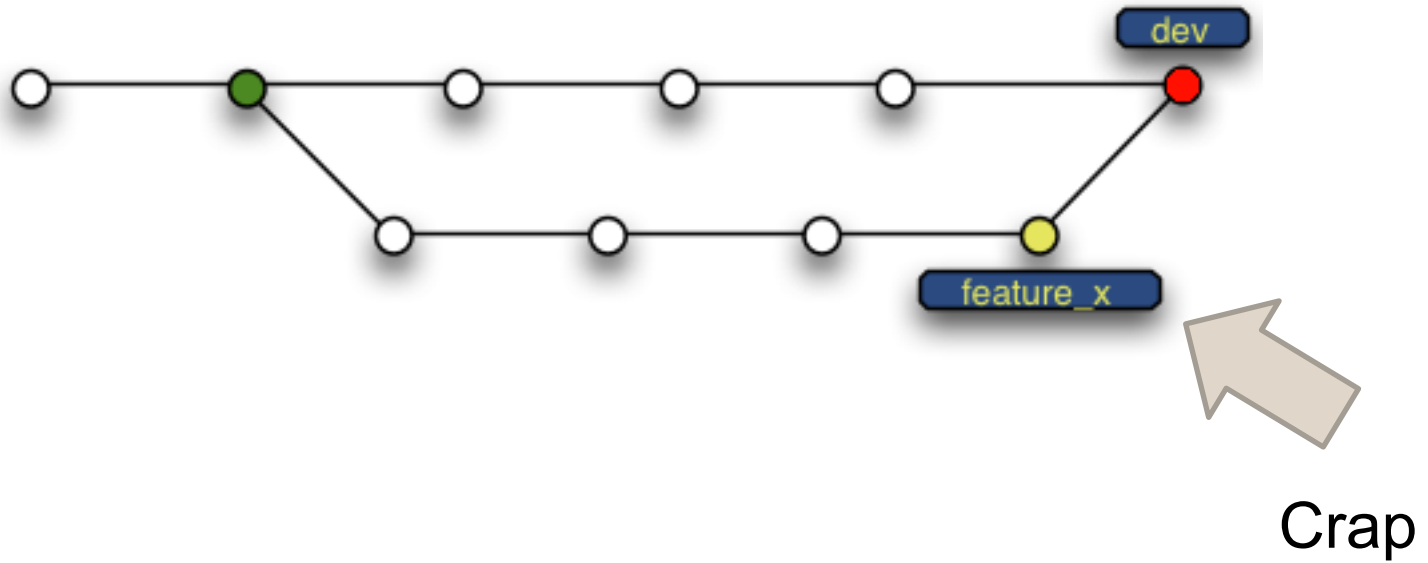
[IUCVI-765] New entity from the browse dialog

[View full commit](#)

ECHA IUCLID / iuclid6

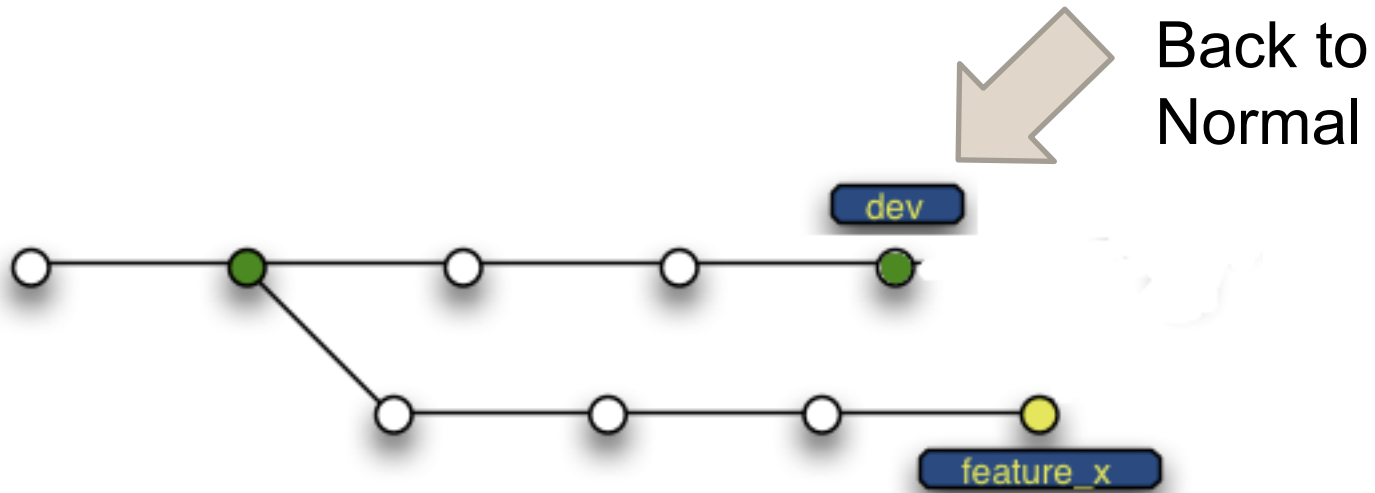
MODIFIED iuclid6-ui/src/main/java/eu/echa/iuclid6/ui/components/documentreference/DocumentBrowseDialog.java

Bad commits



Oops – bad commit

Bad commits



Git reset to previous commit

Enterprise versus Open Source



Comparison

Simplicity in day to day operations

Open Source



Pull, push,
merge,
commit,
squash,
rebase

Enterprise



Pull, push,
merge,
commit

Simplicity in Build server setup

Open Source



Custom
scripts, new
jobs for pull
requests

Enterprise



Usage out
of the box

Simplicity in Build server load

Open Source



Jobs $O(n)$
($n = \text{open PR}$)

Builds $O(n^2)$
($n = \text{commits}$)

Enterprise



Jobs $O(1)$

Builds $O(n)$
($n = \text{commits to dev}$)

How branches are treated

Open Source



1 Branch = 1
feature =
1 contributor = 1
pull request

Enterprise



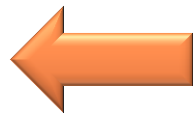
Only final result
matters (all
features
integrated)

Stability of workspace

Open Source



Your branch
may have
been rebased.
Checkout
again!



Dude, wtf?

Enterprise



You can
work on any
branch

Co-operation among developers

Open Source



External
regular
contributors
difficult
(unless
isolated work)

Enterprise



Multiple
external
developers
on same
branch

Conclusion

Use the open source approach
for open source software

Use the Enterprise approach
for Enterprise software

The future



Where we want to go

The future

Hudson search

Hudson » MPBuildPipeline ENABLE AUTO REFRESH

Maturity Profiler Application

[View/Hide Build Pipeline Icon Legend](#)

	mp-fast-test	mp-acceptance-tests	mp-preprod-backup	mp-preprod-deployment	mp-prod-backup	mp-prod-deployment
revision : 3262	mp-fast-test #627 Duration: 5 min 44 sec	mp-acceptance-tests #545 Duration: 1 min 33 sec	mp-preprod-backup #184 Duration: 0.55 sec	mp-preprod-deployment #249 Duration: 1 min 20 sec	mp-prod-backup #18 Duration: 3.4 sec	mp-prod-deployment #46 Duration: 50 sec
revision : 3262	mp-fast-test #626 Duration: 6 min 9 sec	mp-acceptance-tests #541 Duration: 1 min 38 sec				
revision : 3262	mp-fast-test #625 Duration: 5 min 50 sec	mp-acceptance-tests #539 Duration: 3 min 6 sec	mp-preprod-backup #182 Duration: 1.4 sec	mp-preprod-deployment #246 Duration: 35 sec		
revision : 3262	mp-fast-test #624 Duration: 8 min 51 sec					
revision : 3257	mp-fast-test #623 Duration: 5 min 50 sec	mp-acceptance-tests #538 Duration: 3 min 33 sec	mp-preprod-backup #181 Duration: 4.2 sec	mp-preprod-deployment #245 Duration: 1 min 14 sec	Manual Execution	
revision : 3256	mp-fast-test #622 Duration: 5 min 49 sec	mp-acceptance-tests #537 Duration: 1 min 30 sec	mp-preprod-backup #180 Duration: 2.3 sec	mp-preprod-deployment #244 Duration: 1 min 23 sec	Manual Execution	
revision : 3255	mp-fast-test #621 Duration: 5 min 47 sec	mp-acceptance-tests #536 Duration: 1 min 29 sec	mp-preprod-backup #179 Duration: 2.3 sec	mp-preprod-deployment #243 Duration: 1 min 22 sec	Manual Execution	
revision : 3254	mp-fast-test #620 Duration: 5 min 43 sec	mp-acceptance-tests #535 Duration: 1 min 34 sec	mp-preprod-backup #178 Duration: 2 sec	mp-preprod-deployment #242 Duration: 1 min 8 sec	Manual Execution	
revision : 3253	mp-fast-test #619 Duration: 5 min 52 sec	mp-acceptance-tests #534 Duration: 1 min 30 sec	mp-preprod-backup #177 Duration: 2.4 sec	mp-preprod-deployment #239 Duration: 1 min 18 sec	Manual Execution	

Software Pipelines

Thank you

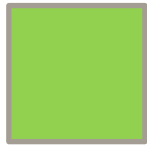


Enterprise versus Open Source

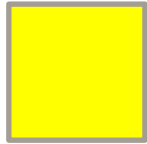


Backup Slides

Enterprise versus Open Source



Valid commit



Invalid commit



“fix build” commit



Build (success)



Build(fail)



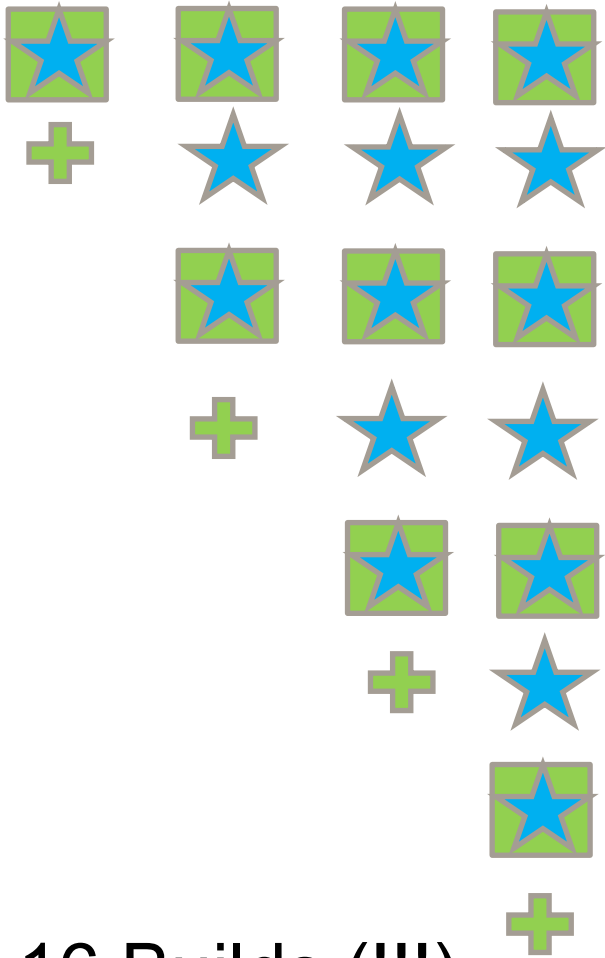
Feature finished (ok)



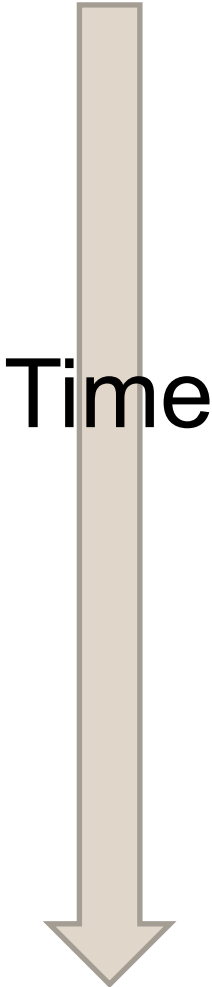
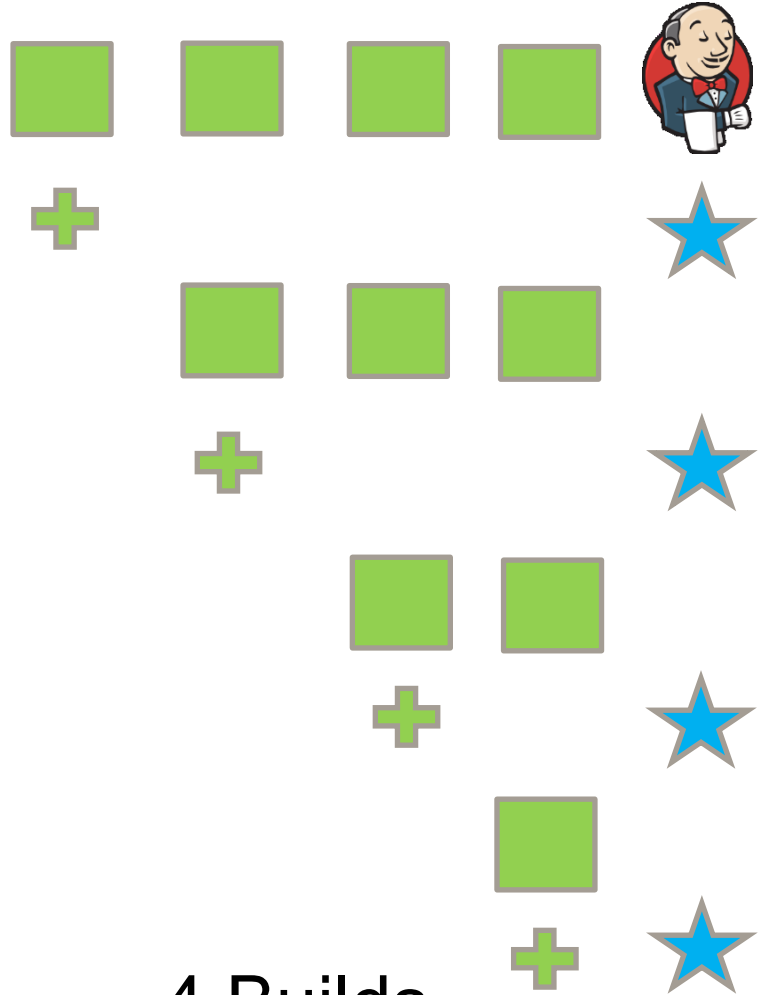
Feature finished (not ok)

Merging 4 open pull requests

Open Source (4 PRs)



Enterprise (4 PRs)



4 PRs = 16 builds

**If 6 open pull requests = 36
Builds???**

- 2 Regular committers on the same company
- 1 external contributor on different country
- 3 open pull requests (1 for each)
- 3 unrelated features that share code
- Scenario = the first committer changes the method signature of a module used by the other two.

Merging 3 open pull requests with errors

