Software Corpora

Ekaterina Pek
Uni Koblenz

Definitions

- *Corpus* is a set of projects (systems) for the research task at hand
- *Corpus engineering* is a sum of all efforts spent on making the corpus usable

Usage

Year	Conf.	Papers	"No"	%	"Yes"	%	"Gray"	
							(i)	(ii)
2012	CSMR	30	9	30	17	56.7	3	1
2012	ICPC	23	11	47.8	8	34.8	1	3
2012	ICSE	87	25	28.7	39	44.8	6	17
2011	ICSM	36	6	16.7	25	69.4	1	4
2011	SCAM	16	4	25	6	37.5	4	2
2011	WCRE	47	10	21.3	21	44.7	5	11
	Total	239	65	27.2	116	48.5	20	38

Used corpora

- Almost always home-grown
- Consist of source code
- Most popular languages: Java, C/C++
- Rather small: Median=2, mode=1

Established corpus: Qualitas

- > 100 systems
- Source code and binary form
- Metadata:
  - prefixes for *system* types
  - LOC, NCLOC, etc.

Qualitas 20101126r

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max
# files	77	500	1,102	2,794	2,845	66,550
# extensions	1	13	21	29.2	35	346
# Java files	44	200	564	1,453	1,192	32,550
# sysJava files	38	198	540	1,341	1,192	29,180
# roots	1	2	5	29.79	15	1,499
# sysRoots	1	1	3	21.27	8	1,130
# packages	2	21	37	130.9	92	3,620
# prefixes	1	1	1	1.648	2	20
# JARs	1	1	7	38.22	20	1,822
# ANT scripts	1	1	1	14.43	2.75	1,020
# Maven 1.x scripts	1	1	1	12.08	1	1,035
# Maven 2.x scripts	1	1	1	8.811	1	341
# Makefile scripts	1	1	1	2.679	1	135

Survey vs. Qualitas

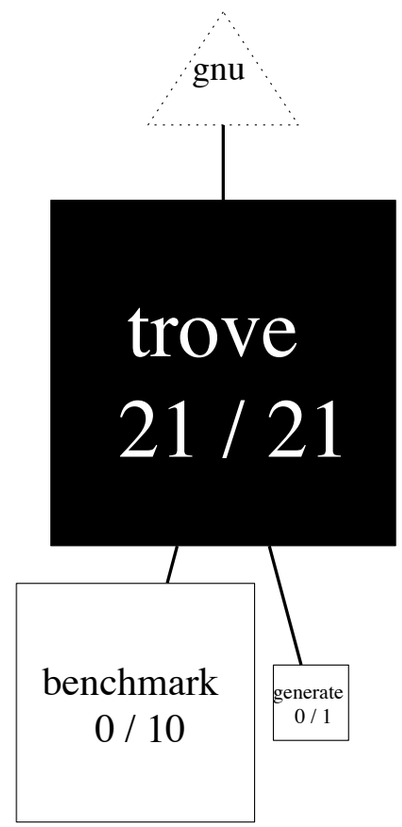
Freq	# Sys	# In Q.	In Qualitas	Not in Qualitas
8	1	1	jedit	—
7	2	2	argouml, eclipse	—
5	3	3	ant, jhotdraw, pmd	—
4	3	2	lucene, xalan	rhino
3	8	4	aspectj, hibernate, hsqldb, jfreechart	fop, jabref, jface, jython
2	20	10	antlr, htmlunit, itext, jmeter, pooka, rssowl, tomcat, vuze, weka, xerces	bloat, chart, compare, debug.core, exvantage, freemind, jdt.core, lexi, quickuml, zxing

Qualitas: Drawbacks

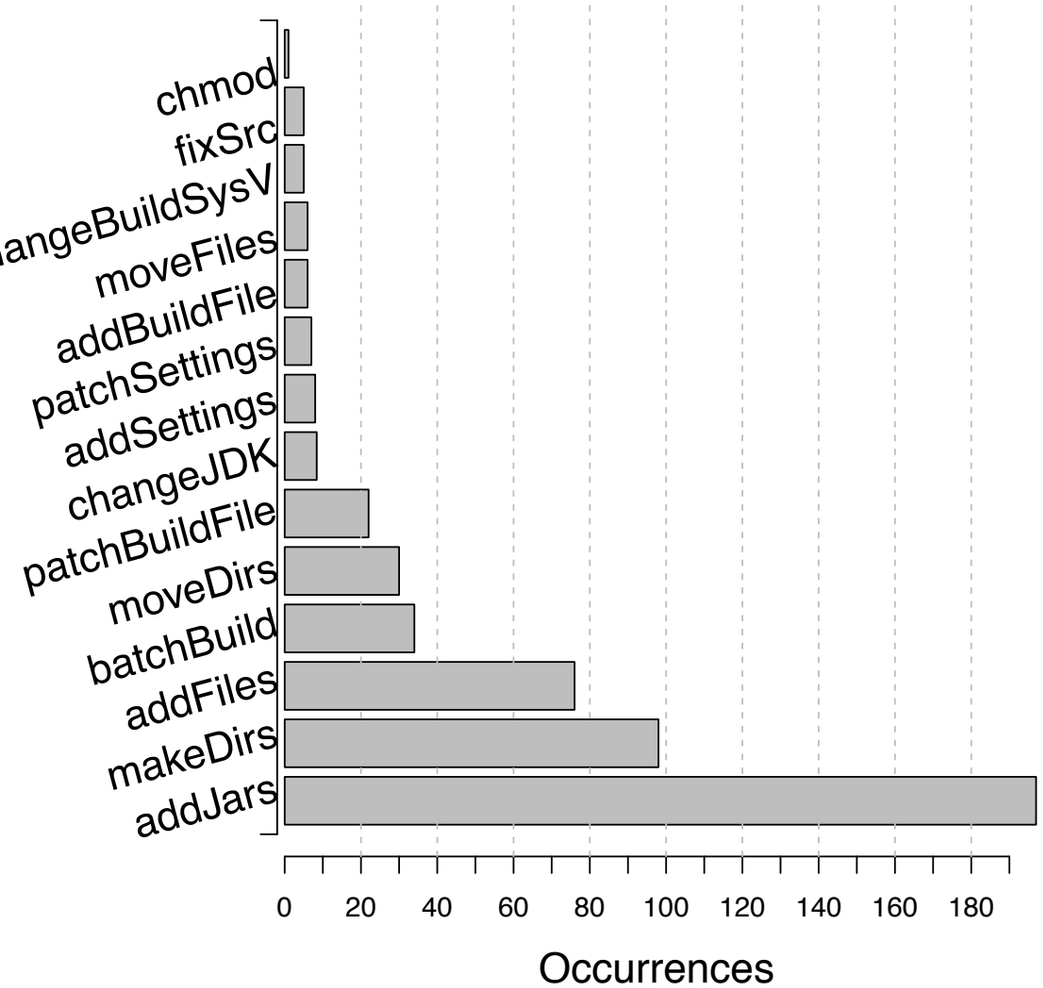
- For our kind of research (API usage):
  - libraries missing
  - no automation of build
- Not only our problem:
  - more than third of corpora in survey need resolved dependencies
  - any points-to analysis

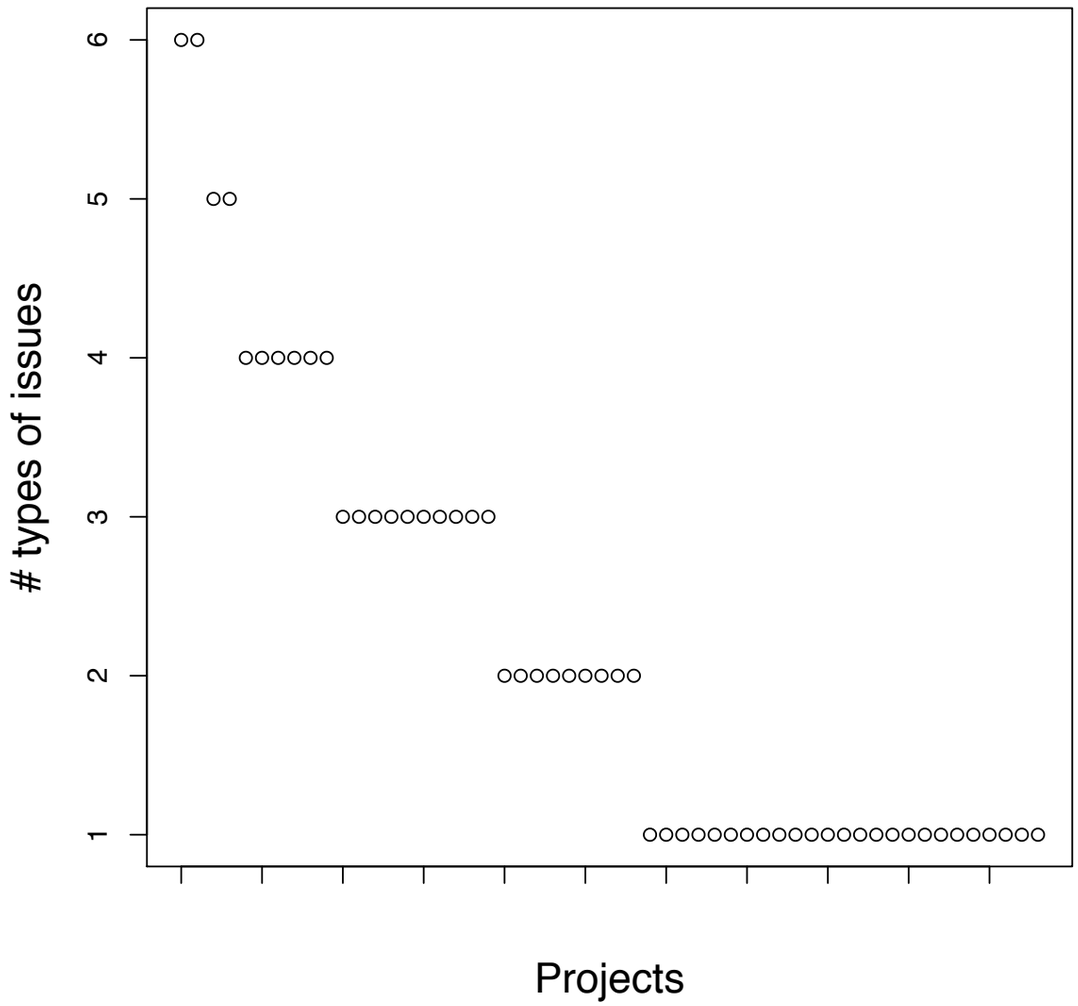
Our method

1. input : $corpus, systemCandidateList$
2. output : $corpus$
3. for each $name$ in $systemCandidateList$:
4. $\quad (p_{src}, p_{bin}) = obtainSystem(name);$ — once
5. $\quad patches = exploratoryBuild(p_{src}, p_{bin});$ — few times
6. $\quad timestamp = build(p_{src}, patches);$ — corpus build
7. $\quad (java, classes, jars) = collectStats(p_{src});$
8. $\quad java' = filter(java);$
9. $\quad (jars_{built}, jars_{lib}) =$
   $\quad\quad detectJars(timestamp, java', jars);$
10. $\quad java'_{compiled} =$
    $\quad\quad detectJava(timestamp, java', classes, jars_{built});$
11. $\quad p'_{src} = (java'_{compiled}, jars_{lib});$
12. $\quad p'_{bin} = jars_{built};$
13. $\quad p' = (p'_{src}, p'_{bin});$
14. $\quad$ if $validate(p')$:
15. $\quad\quad corpus = corpus + p';$
16. $\quad\quad factExtraction(p');$



Build patches





Code classifications

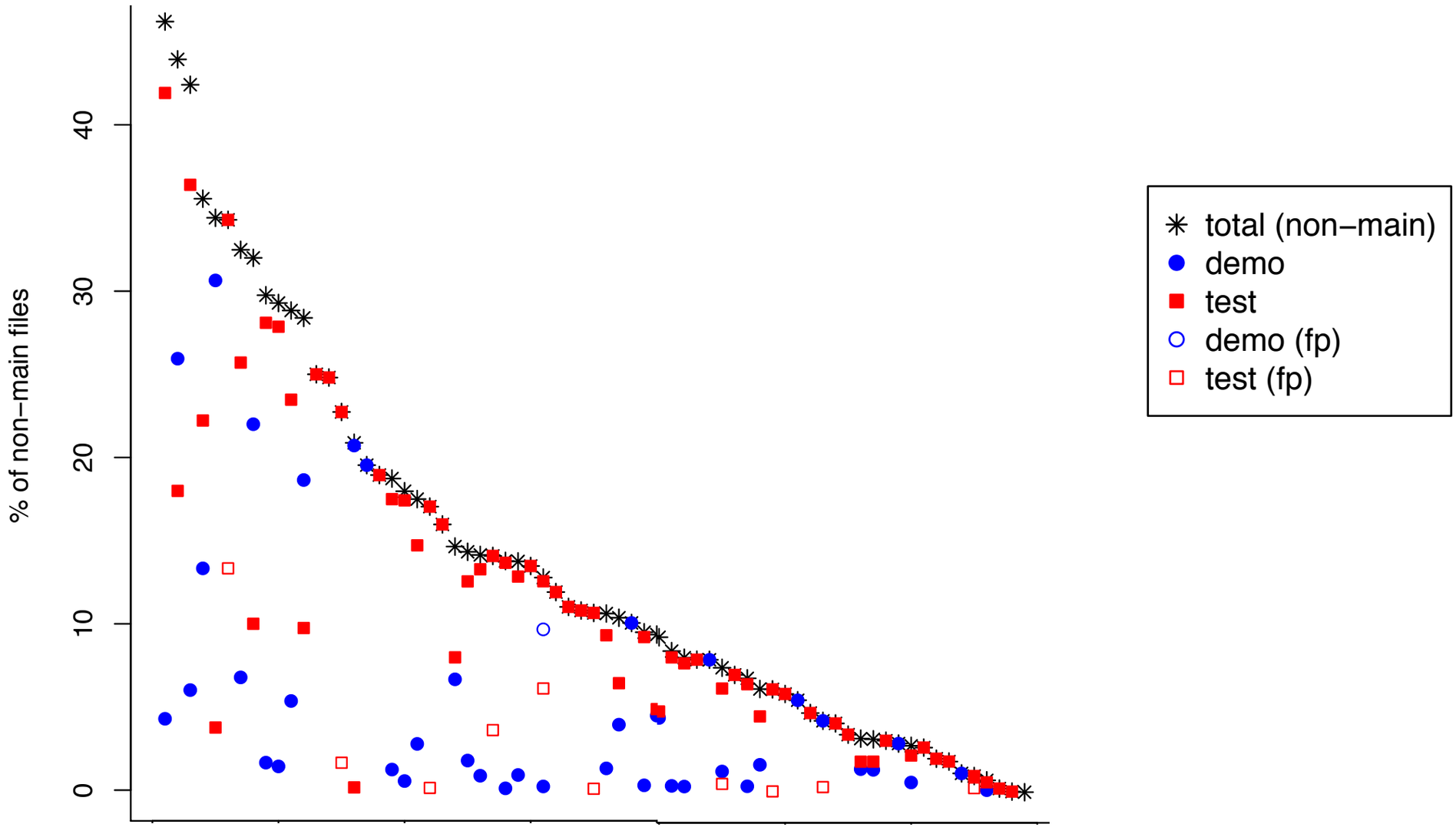
- Qualitas metadata:
  - system vs. non-system
- Our heuristics:
  - core vs. test vs. demo

System code

System	Prefix	# files	% of all	% in	covered by		isIn	Comment
				built JARs	built JARs	binaries		
fitjava	eg.*	21	30.88	100.00	21	21		Example(s)
mvnforum	net.myvietnam.*	174	24.96	100.00	174	174		Example(s)
webmail	*<default>*	24	21.62	0.00	0	0		Default package
jsXe	gnu.regexp	27	20.61	100.00	27	0		3rd party sources
jext	*<default>*	101	18.17	0.00	0	0		Default package
jsXe	treeview.*	20	15.27	100.00	20	0	✓	Feature(s)
gt2	net.opengis.*	801	14.95	100.00	801	801	?	Extension(s)
jFin_DateMath	*<default>*	8	12.70	0.00	0	0		Default package
azureus	org.bouncycastle.*	384	11.83	100.00	384	384		3rd party sources
jext	org.gjt.sp.jedit	61	10.97	100.00	61	61		3rd party sources
mvnforum	org.mvnforum.*	75	10.76	0.00	0	0	?	Contribution(s)
itext	com.itextpdf.rups.*	46	10.29	0.00	0	0	?	Extension(s)
jsXe	org.syntax.*	13	9.92	100.00	13	0		3rd party sources
jrefactory	*<default>*	127	9.58	0.00	0	0		Default package
fitlibraryforfitnesse	fitbook.*	64	8.66	100.00	64	64		Example(s)
jsXe	sourceview.*	11	8.40	100.00	11	0	✓	Feature(s)
fitlibraryforfitnesse	fit.*	62	8.39	100.00	62	62		3rd party sources
mvnforum	com.mvnsoft.*	54	7.75	98.15	53	53	?	Contribution(s)
roller	org.apache.jsp.*	35	5.71	0.00	0	0		Internals of app.server
jhotdraw	net.n3.nanoxml	23	5.10	100.00	23	23		3rd party sources
compiere	org.apache.ecs.*	126	5.06	100.00	126	126		3rd party sources

5% cut: the rest contains 65 prefixes from 32 projects with median “% of all” 0.55%

Non-core code



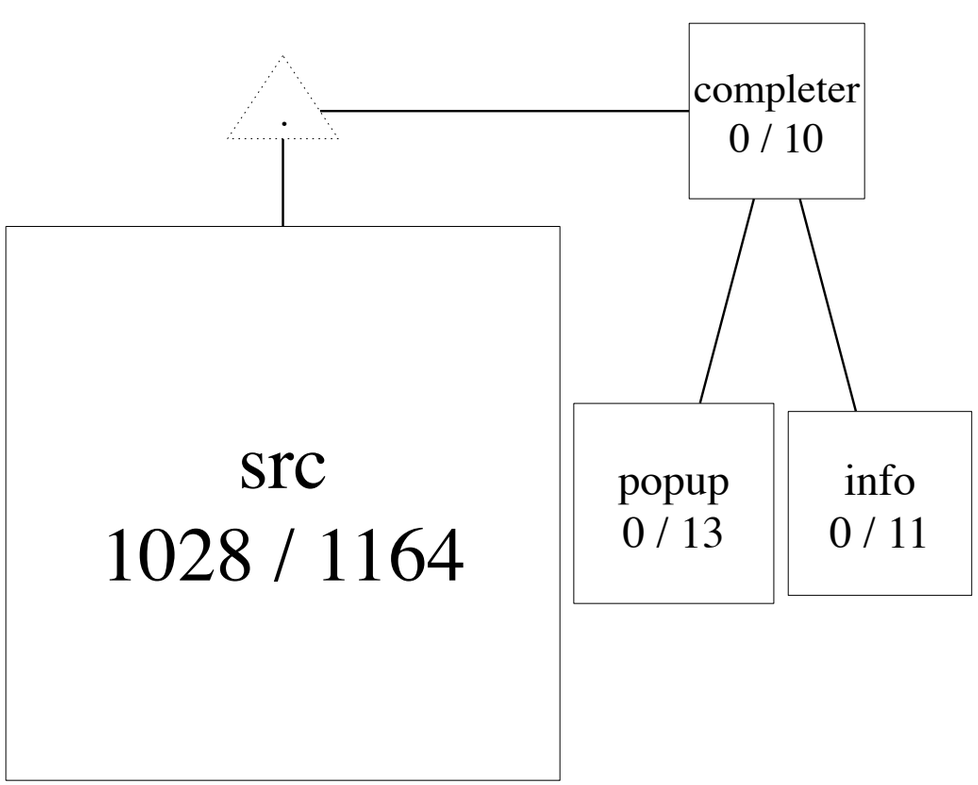
%	Min.	1st Qu.	Median	Mean	3rd Qu.	Max
total	0.04	5.36	10.73	14.14	19.09	46.2
test	0.04	4.57	9.75	11.81	17.04	41.91
demo	0.11	0.91	1.78	5.88	6.66	30.65

Build results

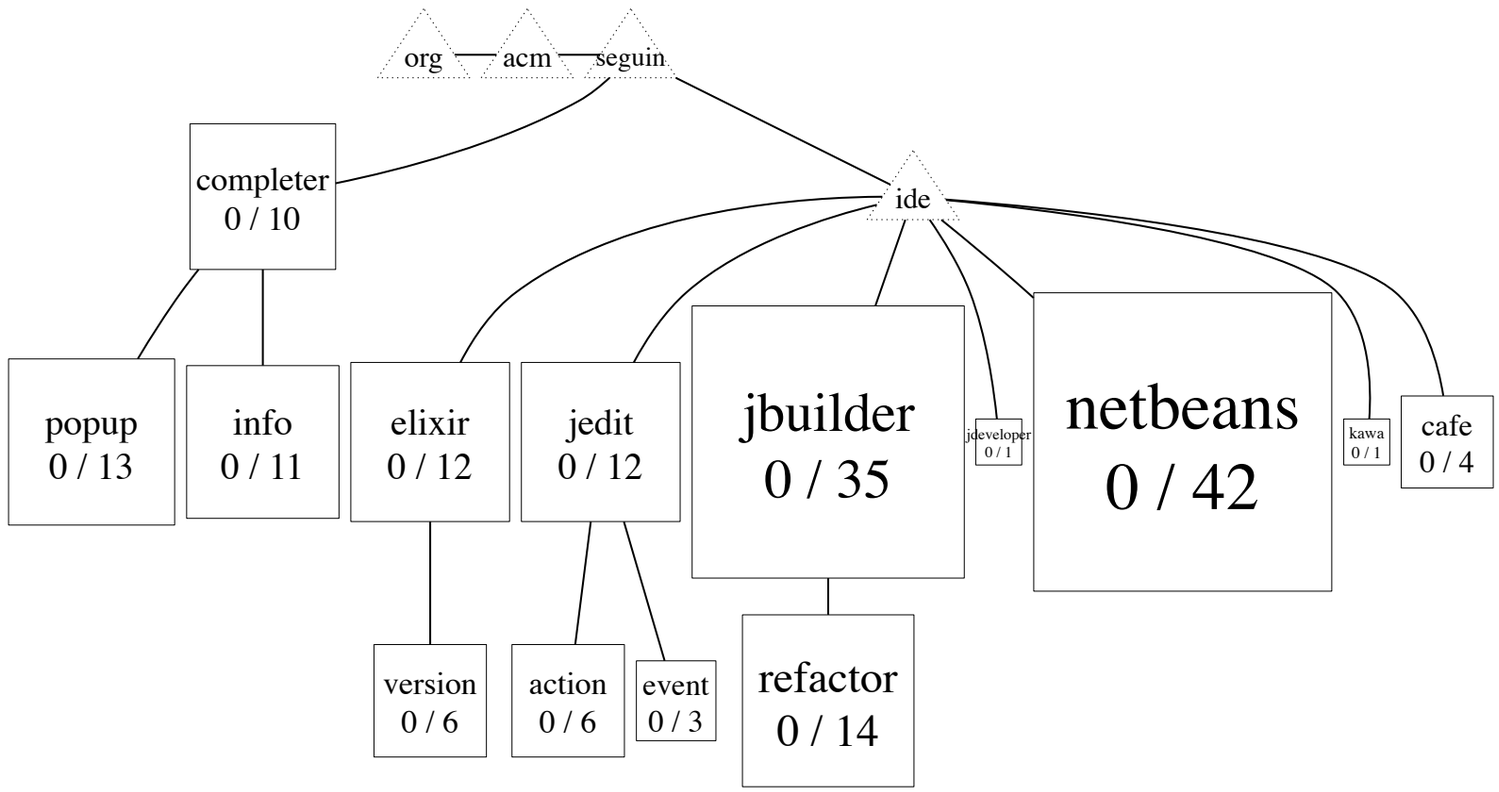
System	types			Compare to Q	# non-packed roots		# non-packed packages		Reason
	# total	% non-compiled	% non-packed		completely	partially	completely	partially	
mvnforum	510	36.08	36.08	=	3	0	11	0	Feature
trove*	32	0.00	34.38	=	0	1	2	0	Feature
james	419	34.13	34.13	=	1	4	4	3	???
checkstyle	359	25.63	25.63	=	2	0	6	0	Feature
log4j*	242	11.16	23.14	>	8	1	4	4	???
gt2	5234	17.90	17.90	>	34	8	128	25	???
jrefactory	1199	14.18	14.18	>	3	1	14	0	Feature
itext	447	10.29	10.29	=	1	0	10	0	Feature
jung	358	9.78	9.78	>	2	0	6	0	Feature
hsqldb*	428	3.27	9.58	>	0	1	3	1	???
nakedobjects	2176	8.96	8.96	=	9	0	33	3	Feature
jext	365	8.77	8.77	>	1	0	3	0	Feature
derby*	1779	1.41	7.31	=	3	5	17	27	???
jag	130	6.15	6.15	=	1	0	1	1	Feature
proguard	562	5.87	5.87	>	0	2	0	10	???

5% cut: the rest contains 21 project with median “% non-packed” 1.05%

Non-built code: Why?



(a) Root forest



(b) Package forest

Summary

- Refined approach to corpus engineering (In fact, the only one detailed/practical)
- As means of validation, projects are exported to Eclipse
- Providing different “views” for different fact extractors

Questions?

pek@uni-koblenz.de