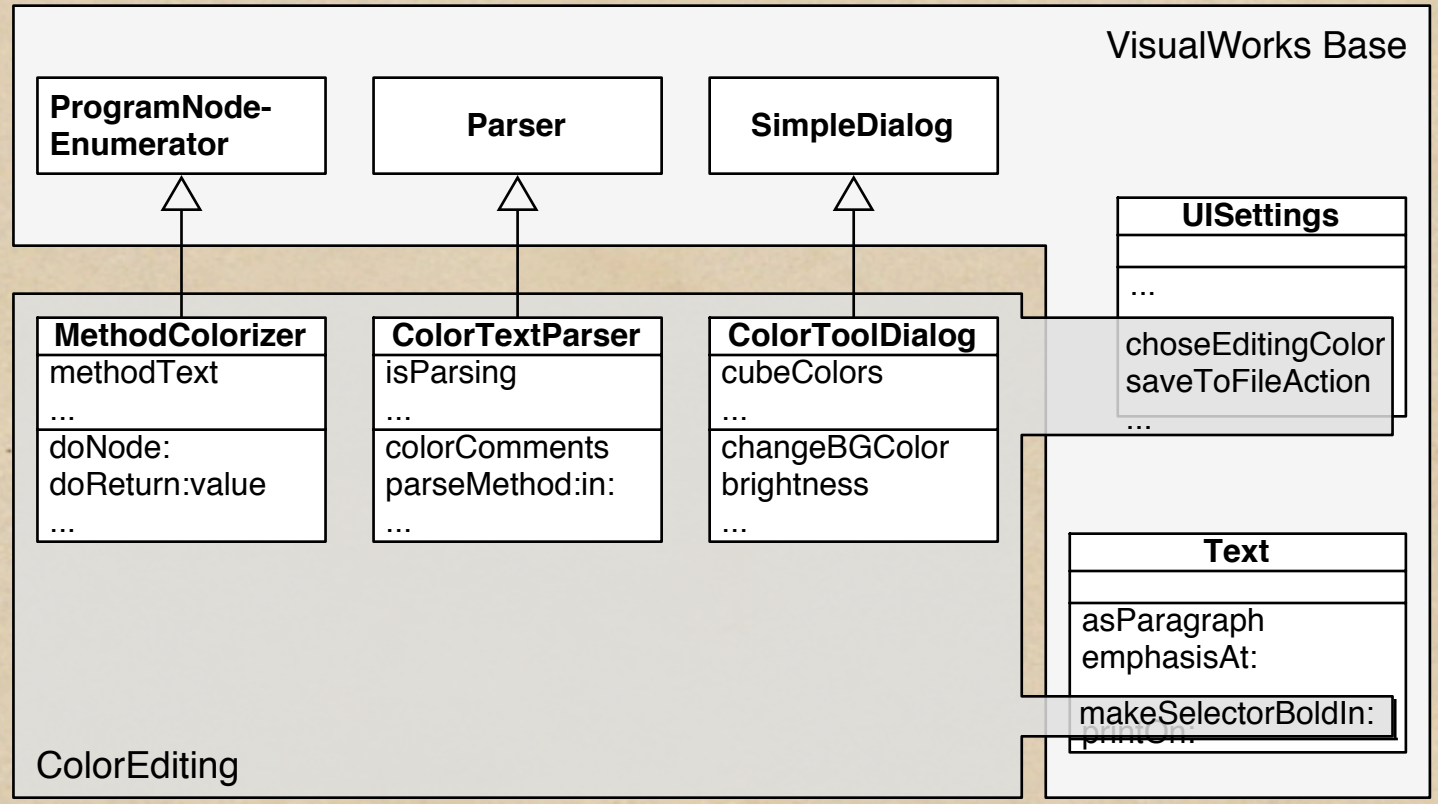
Parcels: a Fast and Feature-Rich Binary Deployment Technology

Eliot Miranda
David Leibs
Roel Wuyts

Deployment

- Writing software is easy
  - just use the right language :-)
- But what about deployment?
  - How to package your code
    - Do you bend your design?

Deployment driving design



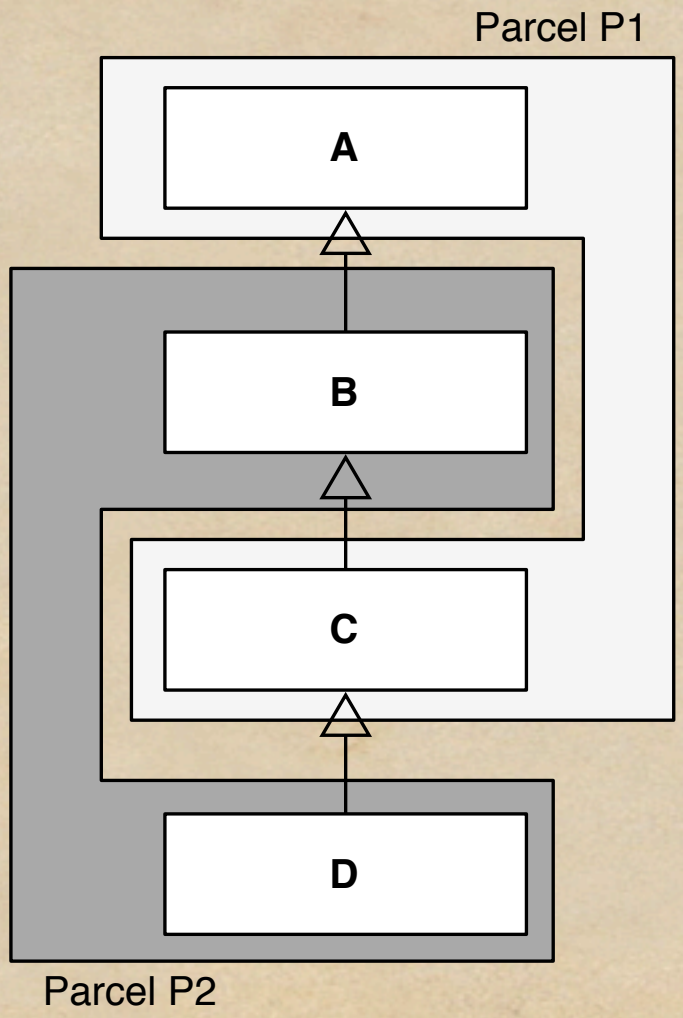
Parcels in a nutshell

- binary code deployment technology
- can be loaded and unloaded
- loading is fast
- support class extensions
- have meta-information (version, prereqs)

What's with (un)loading?

- Loading adds code to the system
  - what is not loaded is remembered
  - supports shape changing of classes
- Unloading removes it...
  - Interacts with method replacement

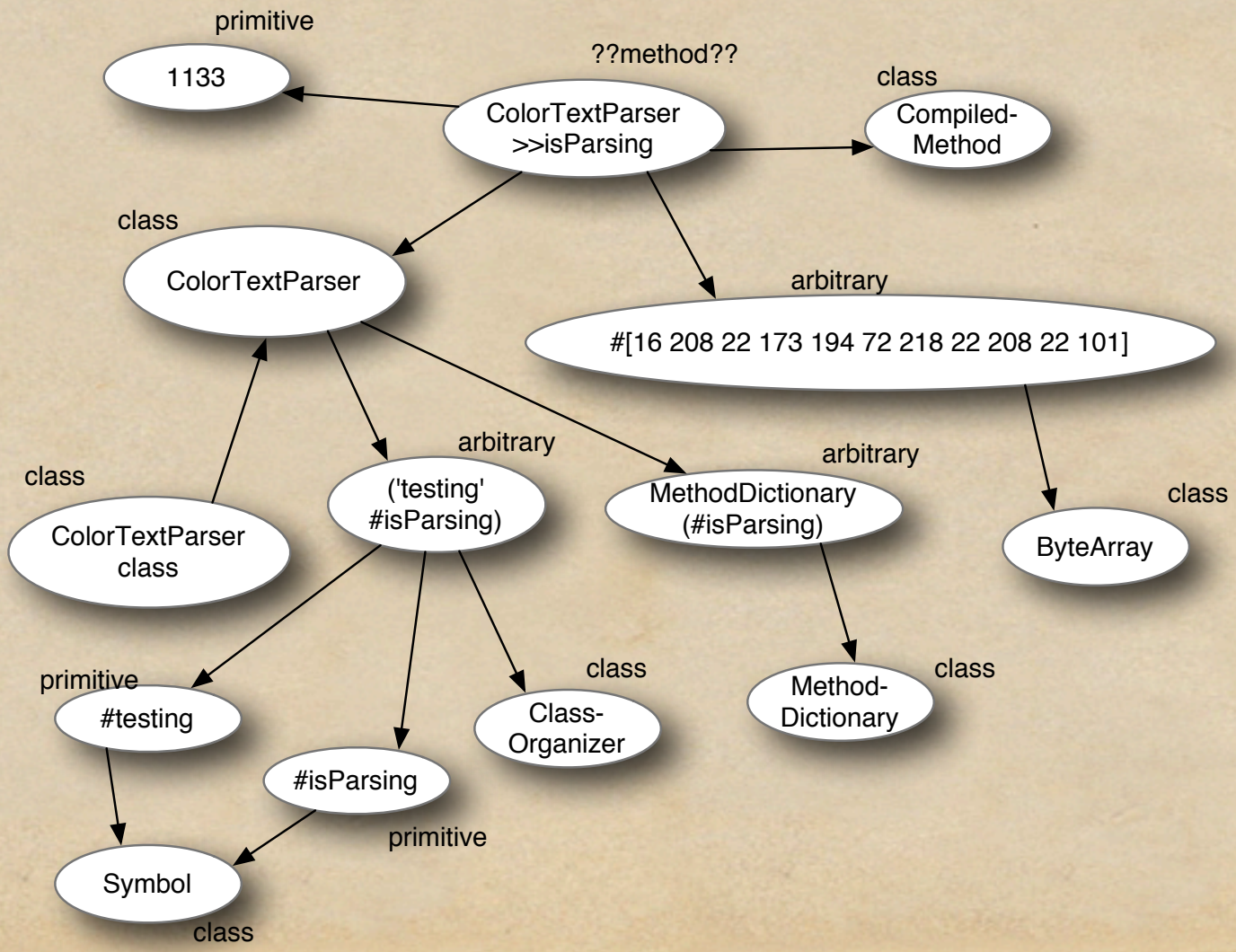
Loading recursive things



Why are they fast?

- Pickling: store object graph so that it can be loaded later on (unpickled)
- typically need recursive descent parser
- Parcels: separate and order the node descriptions from the arc descriptions
  - two sections: objects & references

Parcel example



Pickled file

"refs to dependent classes"
'Kernel.Parser'
16406
22
'source'
'mark'
'prevEnd'
'hereChar'
'token'
'tokenType'
...
"strings, bytearrays, floats, ..."
'methodDict'
'Kernel'
'testing'
'isParsing:'
...
#[16 208 28 173 ... 208 28 101]
#[16 208 23 173 ... 208 23 101]
...
0.86
0.8
...
1291792260
1292108356
...

"Classes defined in parcel"
Kernel
Parser
'ColorTextParser'
16393
0
9
'superclass'
'methodDict'
'format'
'subclasses'
'instanceVariables'
'organization'
'name'
'classPool'
'environment'
16413
7
29
'source'
'mark'
'prevEnd'
'hereChar'
'token'
'tokenType'
...

"arcs"
1
2
Kernel.ColorTextParser class
Kernel
#UISettings
...
"Instances of indexed objects"
OrderedCollection
1
5
CompiledMethod
141
2
1
1
...
"compiled methods"
ColorTextParser>>isParsing:
...

VW/Monticello/Dolphin

Soul (v. 3.2)	146 / 2,081 / 61	Store	17,586	16,690
	(131 / 1,834 / 51; no init)	mzc	6,799	5,000
	(129 / 1812 / 51; no init)	PAC	4,684	455
		parcel	1,514	1,677
Swazoo	101 / 6,646 / 4	Store	61,721	43,501
	(no initialization)	mcz	3,420	2,100
	(no initialization)	PAC	1,667	734
		parcel	1,764	49,479
SmallWiki	119 / 1,613 / 13	Store	12,959	8,664
	(no initialization)	mcz	5,776	4,000
	(117 / 1,539 / 8; no init)	PAC	3,016	1,095
		parcel	4,229	1,015
	(117 / 1,539 / 8; no init)	parcel	1,903	983
SIXX	37 / 271 / 100	Store	3,352	2,822
	(no initialization)	mcz	1,741	900
	(55 / 573 / 112)	PAC	1,485	4,031
		parcel	2,331	353

VW/ENVY comparisons

Application	classes/methods/extensions	format	load	write
VW-XML	48 / 543 / 0	chunk	2,615	244
		envy	2,000	900
		parcel	85	211
Soul (v. 2.3)	117 / 1,923 / 18	chunk	7,906	800
		envy	4,000	2,000
		parcel	393	676
RB	187 / 3,327 / 53	chunk	11,554	1,352
		envy	4,000	5,000
		parcel	630	1,920
Jun (v. 4.99.08)	757 / 25,212 / 0	chunk	110,729	8,694
		envy	11,000	14,000
		parcel	2,008	21,979

Conclusion

- Binary code deployment mechanism
  - support loading and unloading
    - with advanced features
  - are very fast to load
  - have meta-information
- Enjoyed by thousands of people