Idioms to Implement Flexible Binding Times for Features

Rodrigo Andrade
Advisor: Paulo Borba
CIn - UFPE

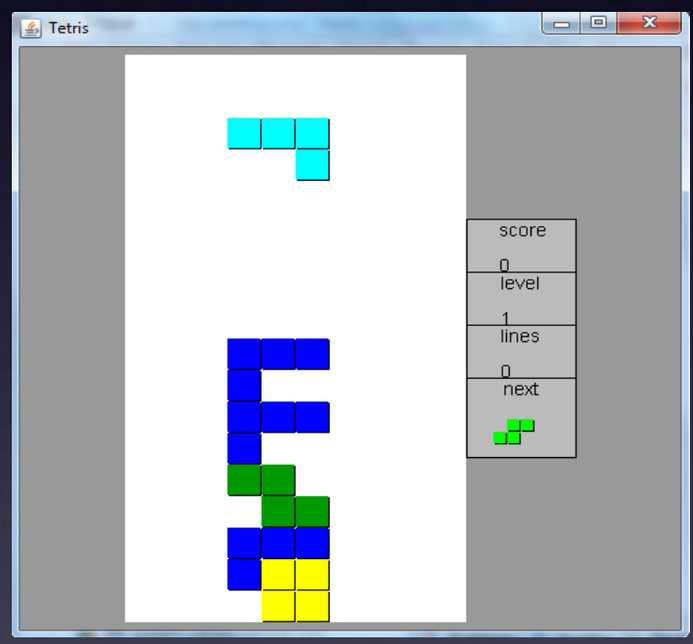
Software Product Lines

Reusable assets

Different products

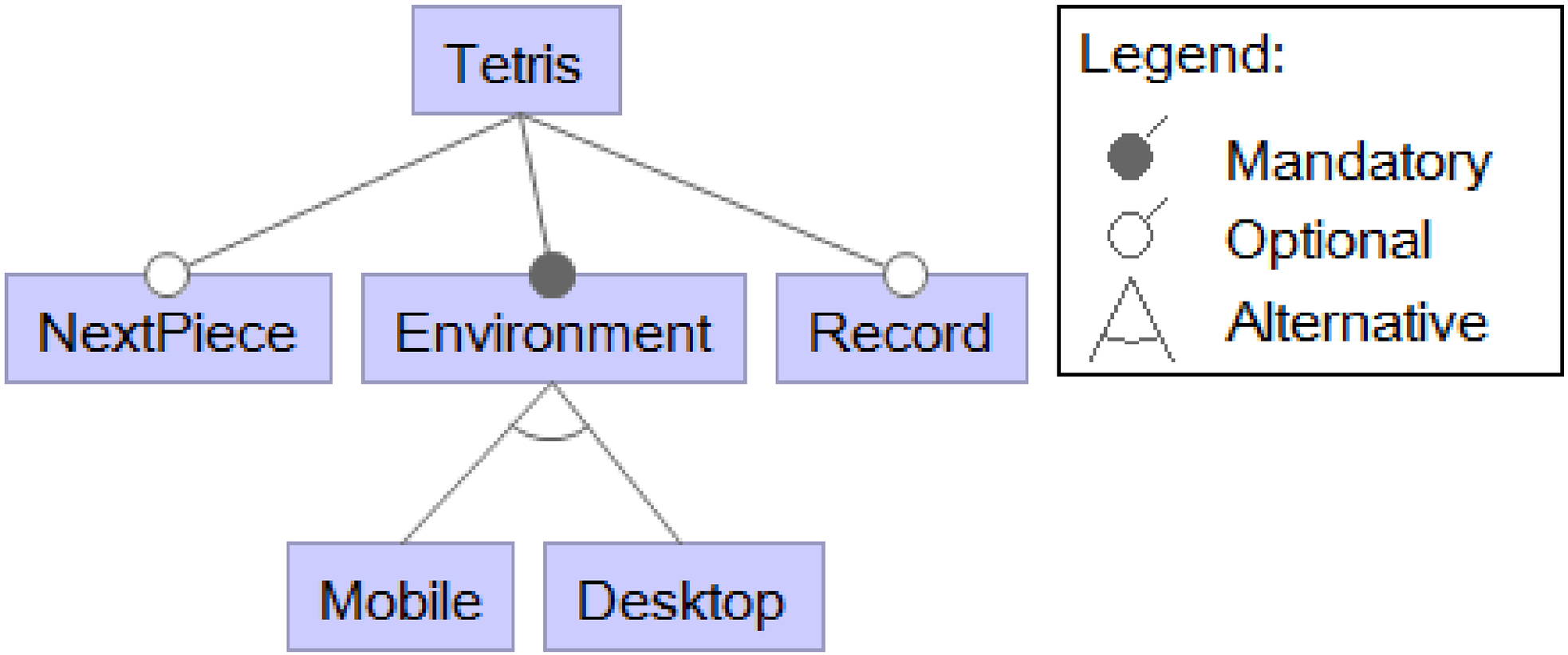
.aj

.java



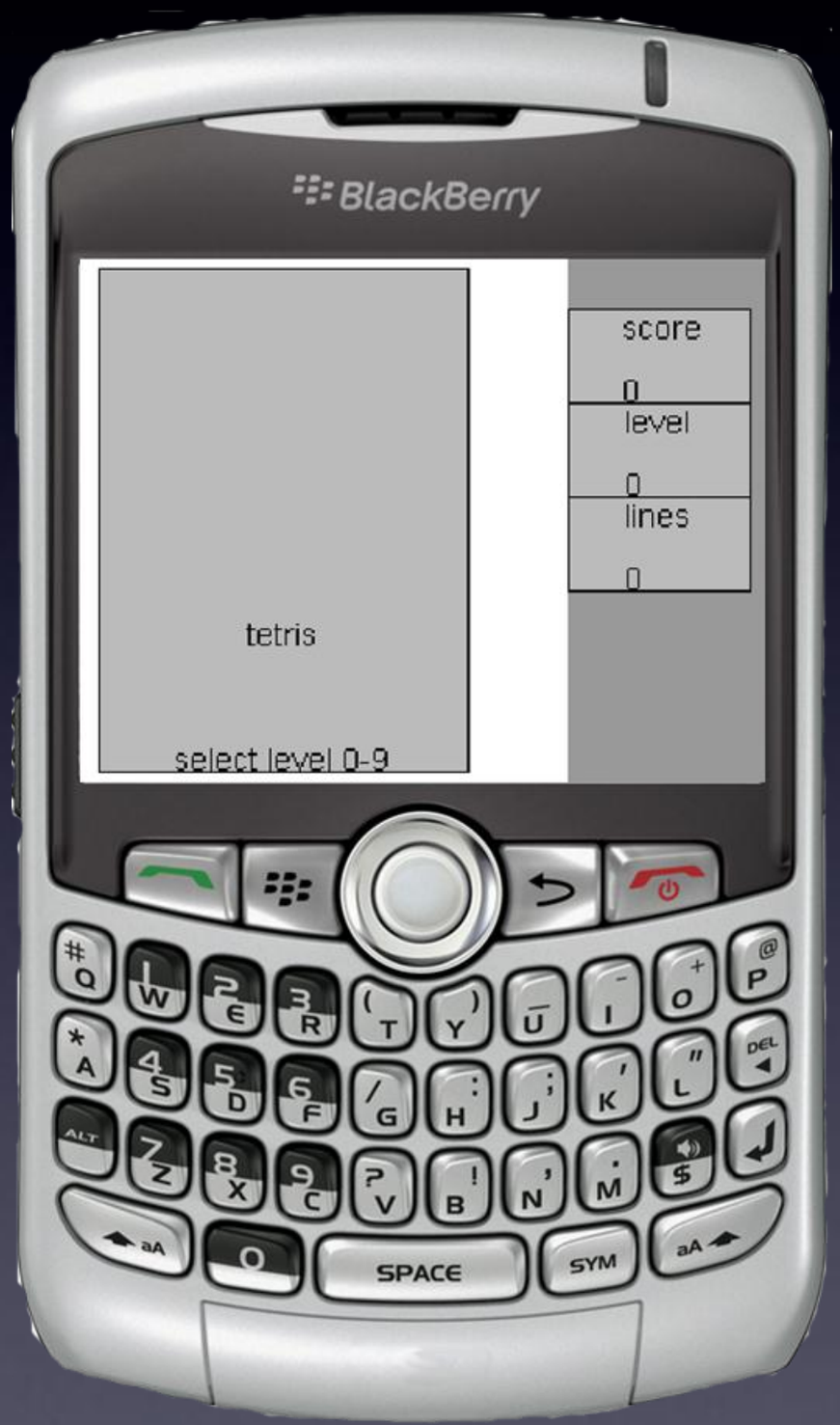
Cost Time to market Quality

Feature Model

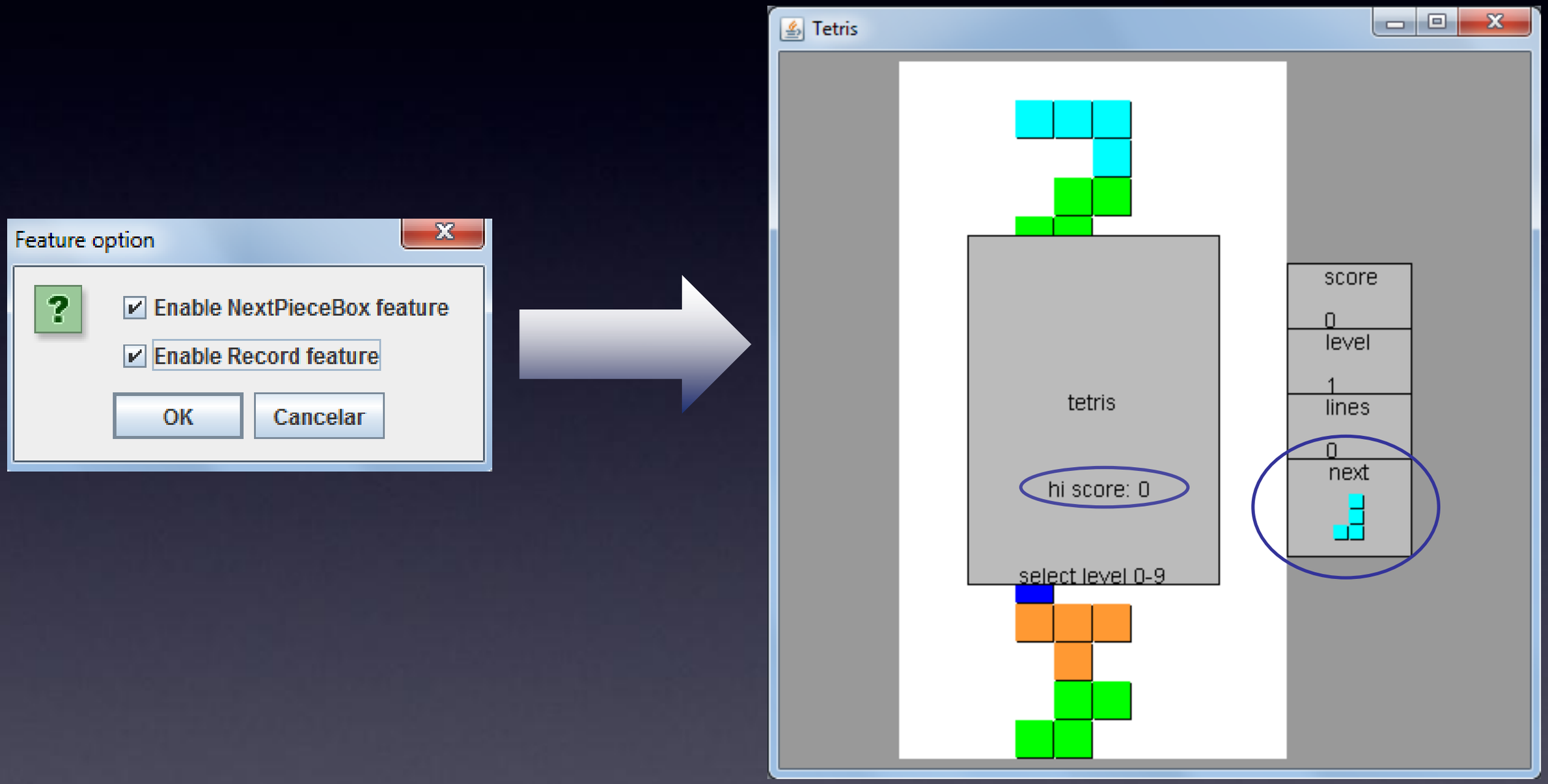


What if we need to change these compositions dynamically?

Static Binding Time



Dynamic Binding Time



Motivation

- Why applying flexible binding time?
- Solutions to implement flexible binding time
- Assessment of these solutions

How to implement flexible binding time?

Edicts Idiom

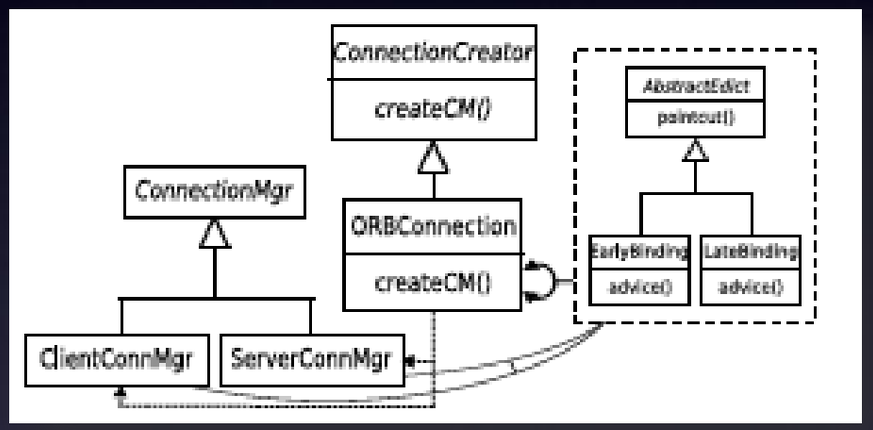
Design Patterns

+

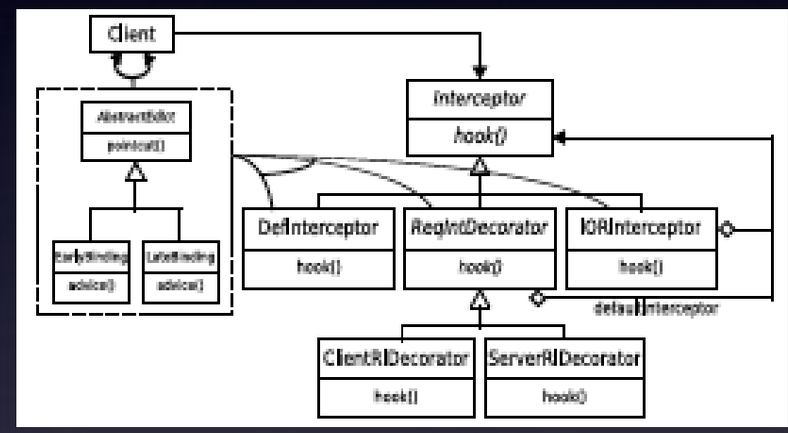
AspectJ

Design Patterns

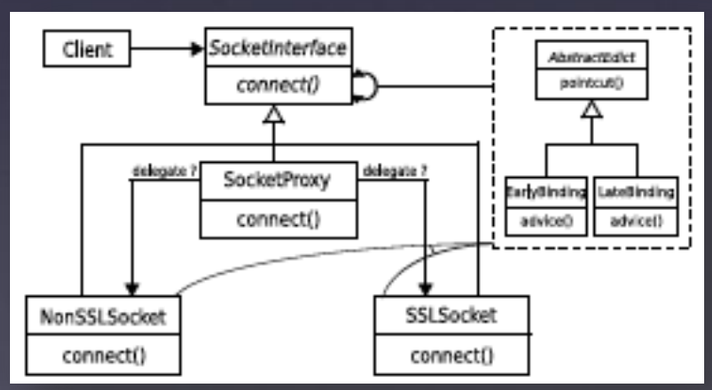
Factory Method



Decorator



Proxy



AspectJ

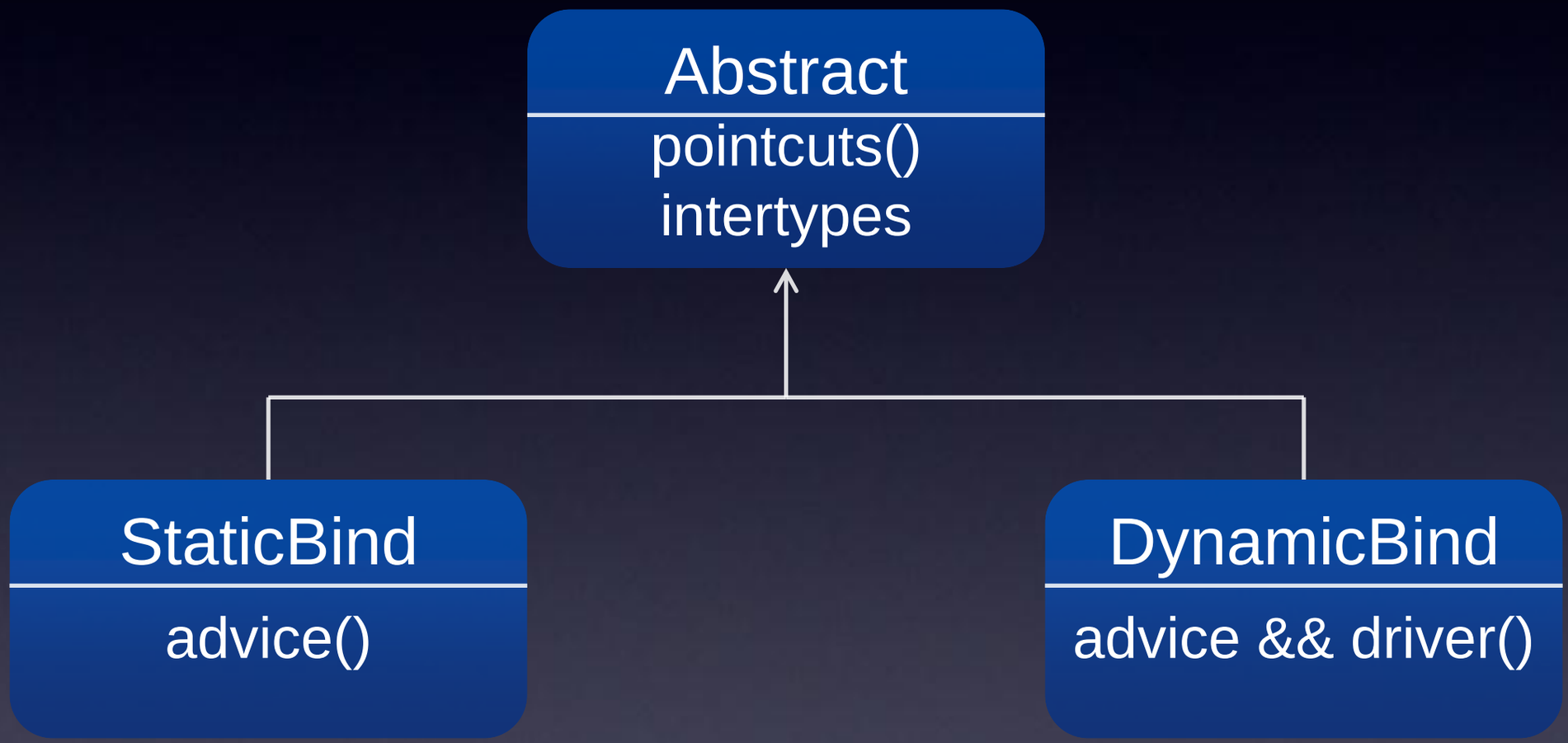
Intertype

```
NextPieceBox TetrisCanvas.nextPieceBox;

pointcut pcnextpiecebox() : execution(*
TetrisCanvas.createNextPieceBoxHook());


after() : createNextPieceBoxHook() {
    nextPieceBox = new NextPieceBox();
}
```

Pointcut

Advice

Hierarchy of Edicts



Abstract

```
abstract aspect ChecksumAbstract {

    pointcut readHeader() : call(void
            EntryHeader.readHeader());

    void FileReader.validateChecksum() {
            . . .
    }
}
```

StaticBind

```
aspect ChecksumStatic extends
    ChecksumAbstract {

    after() throws DatabaseException :
    readHeader() {
        if (lm.doChecksumOnRead) {
            validator = new ChecksumValidator();
        }
    }
}
```

Feature code scattering

DynamicBind

```
aspect ChecksumDynamic extends ChecksumAbstract {

    after() throws DatabaseException : readHeader() {
        if (driver.isActivated("checksum“))
            if (lm.doChecksumOnRead)
                validator = new ChecksumValidator();
    }
}
```

Feature code cloning

Feature and driver code tangling

Edicts problems

- Feature code cloning
- Driver and feature code scattering
- Tangling between feature and driver code

That was our motivation to define new idioms

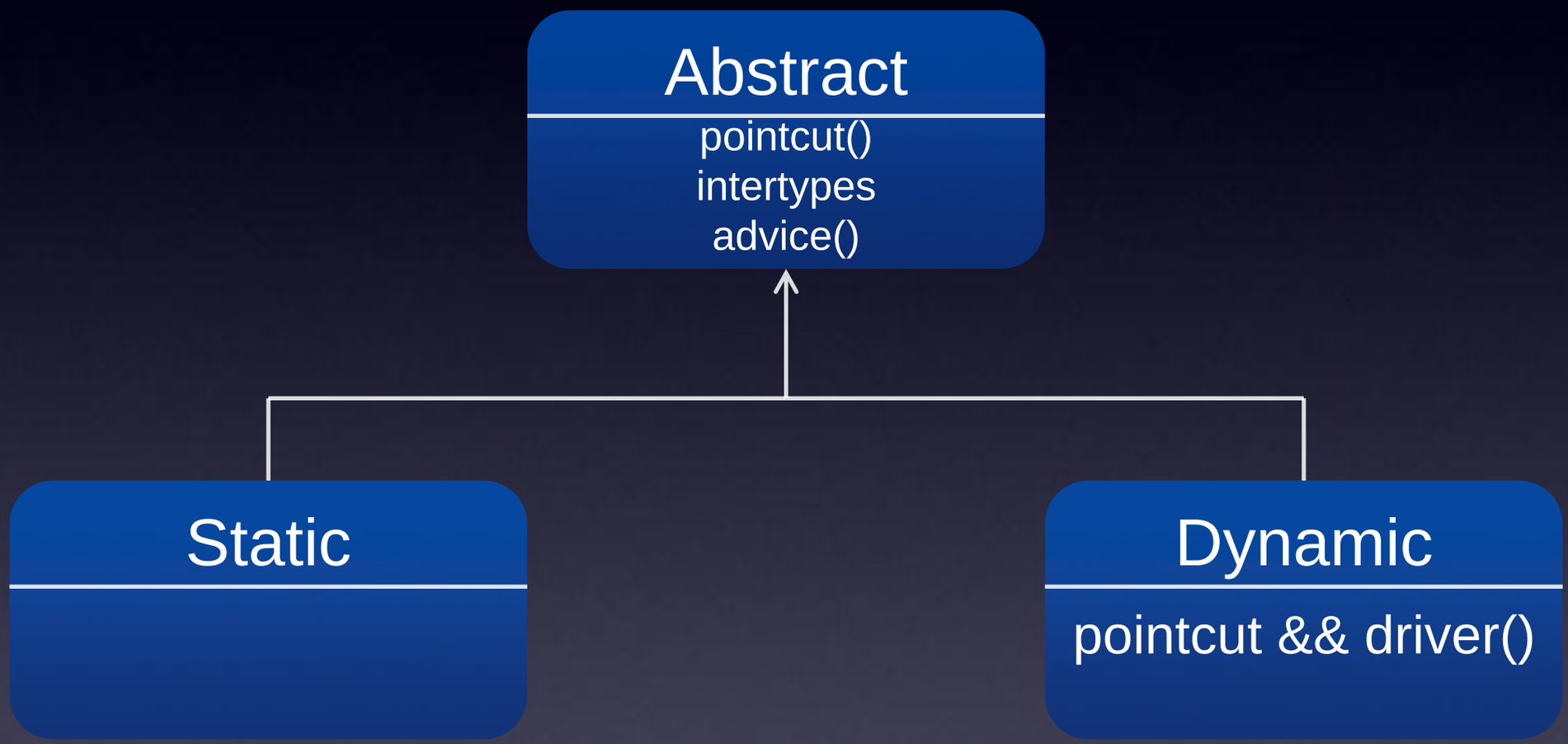
Pointcut Redefinition

AspectJ

+

Redefined pointcuts

Hierarchy of Pointcut Redefinition



Abstract

```
abstract aspect ChecksumAbstract {
    pointcut readHeader() : call(void
            EntryHeader.readHeader());

    after() throws DatabaseException : readHeader() {
            if (doChecksumOnRead ) {
                    validator = new ChecksumValidator() ;
            }
    }

    void FileReader.validateChecksum() {
    }
}
```

StaticBind

```
aspect ChecksumStatic extends ChecksumAbstract {

}
```

DynamicBind

```
aspect ChecksumDynamic extends ChecksumAbstract {

    pointcut driver() : if (new Driver()
            .isActivated("checksum"));

    pointcut readHeader() : ChecksumAbstract
            .readHeader() && driver();
}
```

Implementation size

Driver scattering

We reduce some problems of Edicts, but we still have some

Layered Aspects

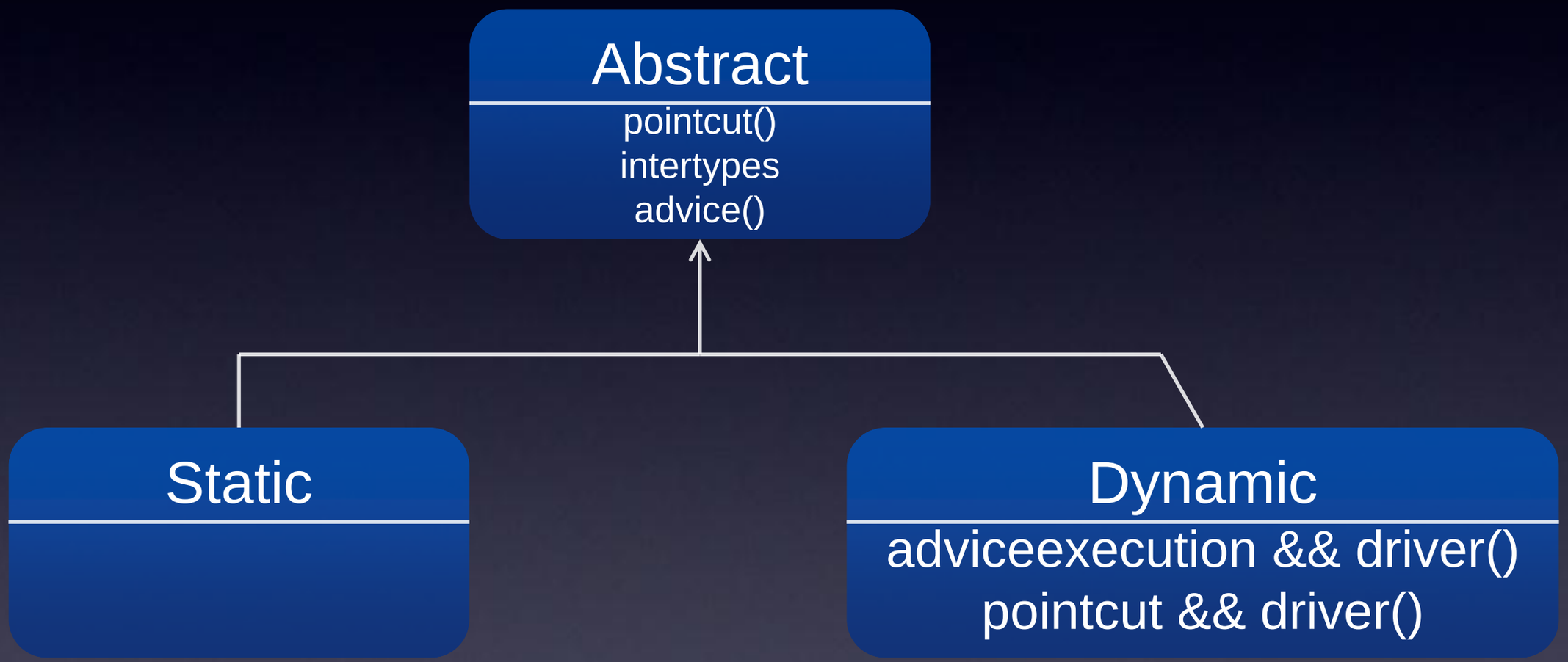
AspectJ

Redefined pointcuts

+

adviceexecution pointcut

Hierarchy of Layered Aspects



DynamicBind

```
aspect ChecksumDynamic extends ChecksumAbstract {

    pointcut driver() : if (new Driver().isActivated(“checksum”));

    pointcut addPrevOffset(int entrySize) :
     ChecksumAbstract.addPrevOffset(entrySize) && driver();

    Object around() : adviceexecution() &&
    within(com.checksum.ChecksumAbstract) && ! driver() {
            return null;
    }
}
```

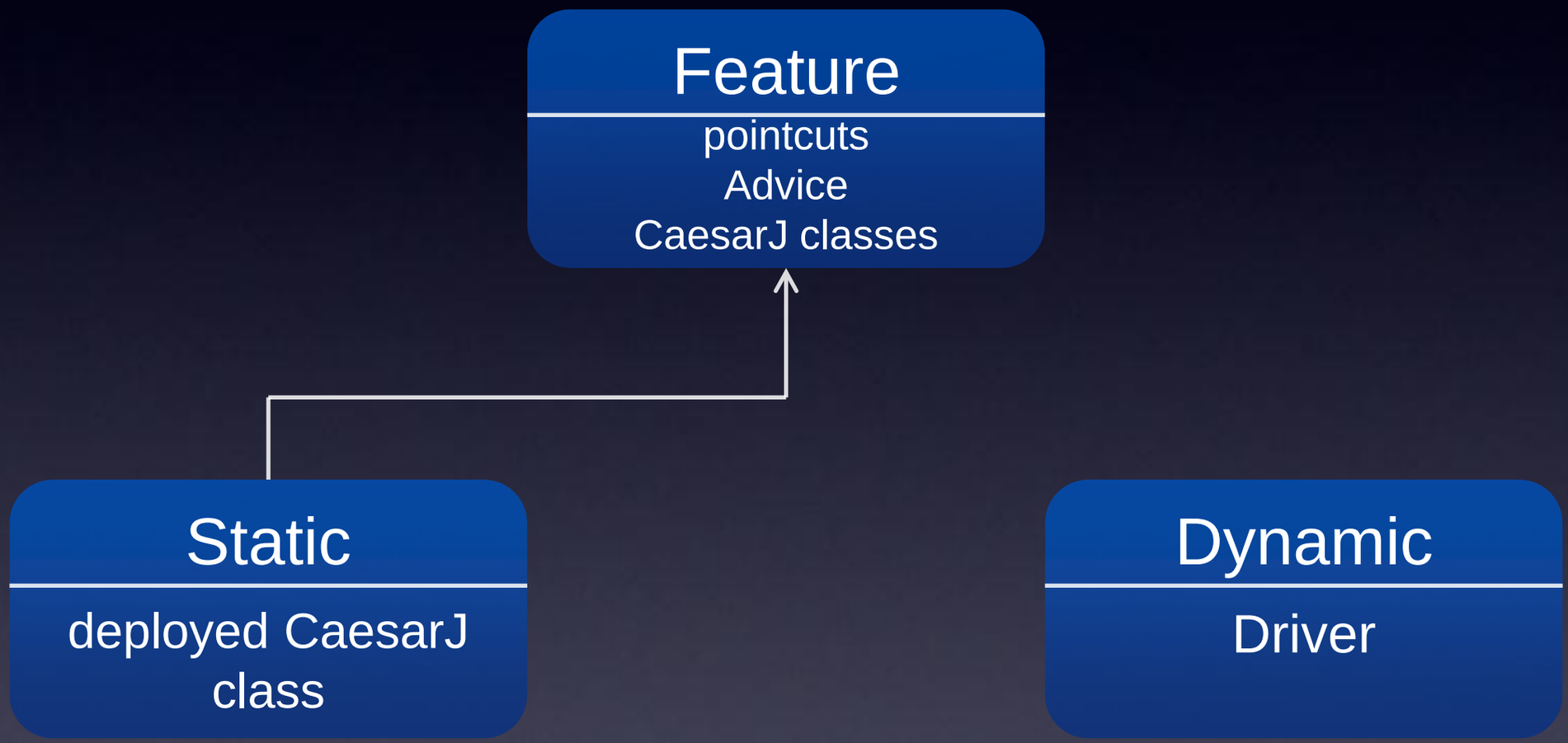
Flexible Deployment

CaesarJ

+

Dynamic deployment

Hierarchy of Flexible Deployment



Feature

```
cclass ChecksumCaesarJClass {
        pointcut readHeader() : call(void EntryHeader.readHeader());

        pointcut addPrevOffset(int entrySize) : execution(ByteBuffer
        LogManager.addPrevOffset()) && args(entrySize);

        after() throws DatabaseException : readHeader() {
                if (doChecksumOnRead ) {
                        validator = new ChecksumValidator() ;
                }
        }

        ByteBuffer around(int entrySize) : addPrevOffset(entrySize) {
                return proceed(entrySize);
        }

        cclass FileReaderCaesarJ wraps FileReader {
                void validateChecksum() {
                }
        }
}
```

StaticBind

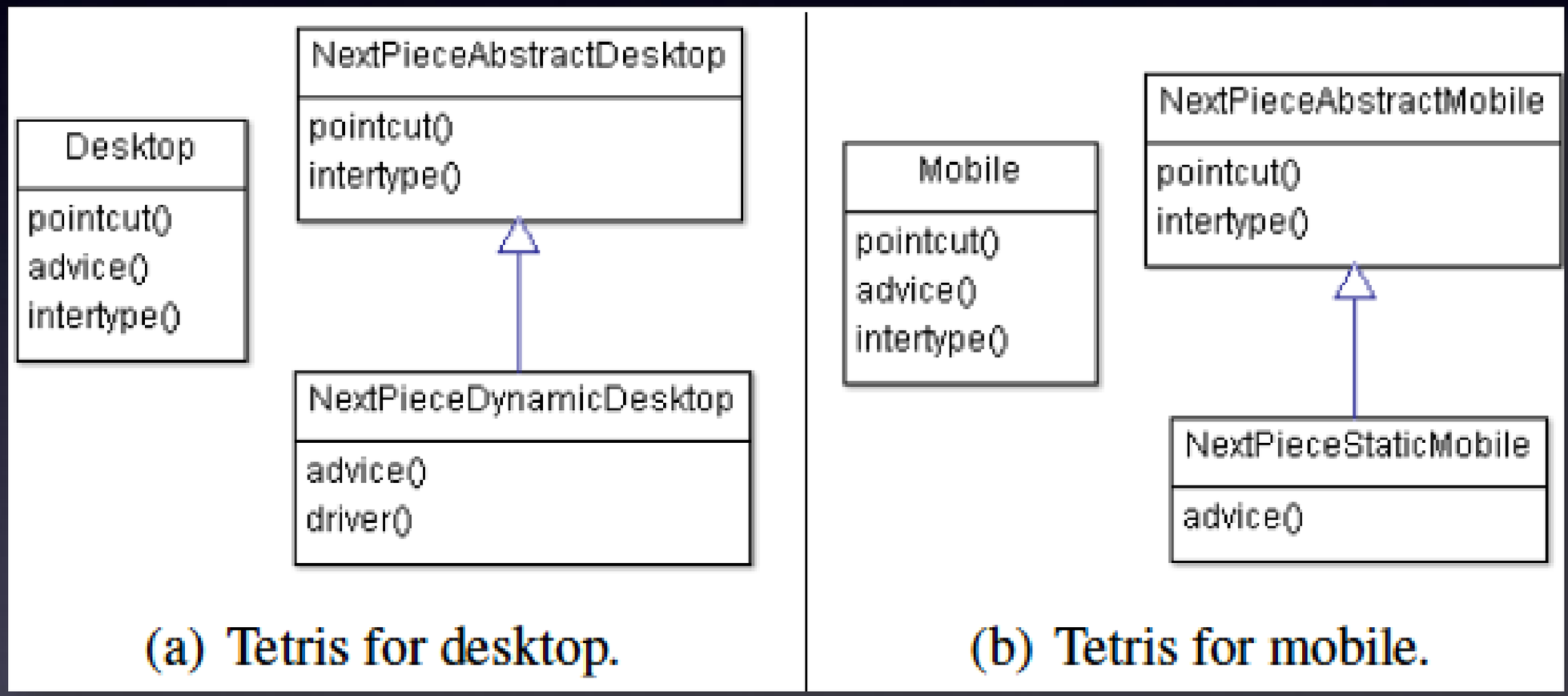
```
deployed cclass ChecksumStatic extends ChecksumCaesarJClass {

}
```

DynamicBind

```
deployed cclass ChecksumDynamic {
    pointcut pc_jarmain() : execution(* JarMain.main(..));

    before() : pc_jarmain() {
        if (new Driver().isActivated("checksum")) {
            ChecksumCaesarJClass checksumCclass = new
              ChecksumCaesarJClass();
         deploy checksumCclass;
        }
    }
}
```

Feature Interaction

Feature Interaction Example



(a) Tetris for desktop.

(b) Tetris for mobile.

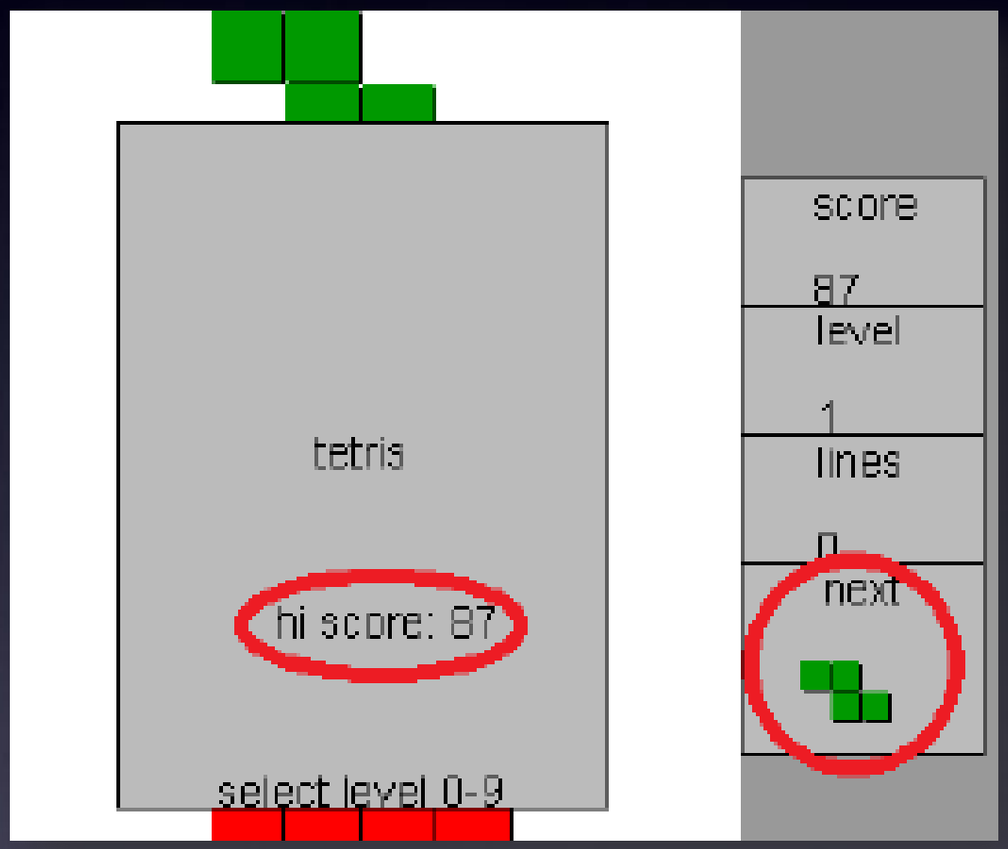
Study Settings

Case Studies

- 18 features
- Tetris | Freemind | ArgoUML | BerkeleyDB

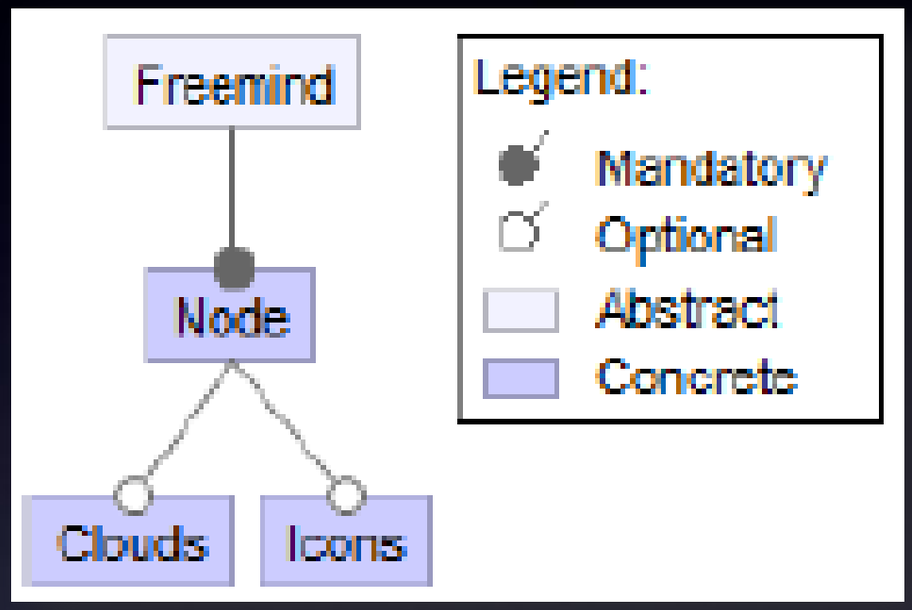
Tetris

- 1500 SLOC
- 400 features SLOC

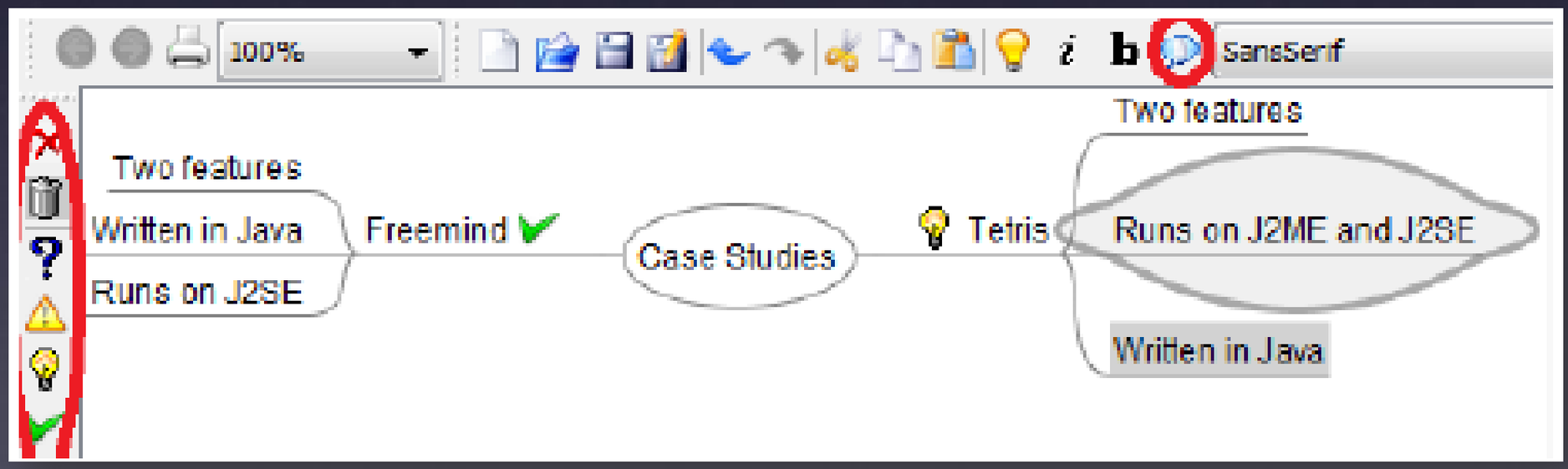


Freemind

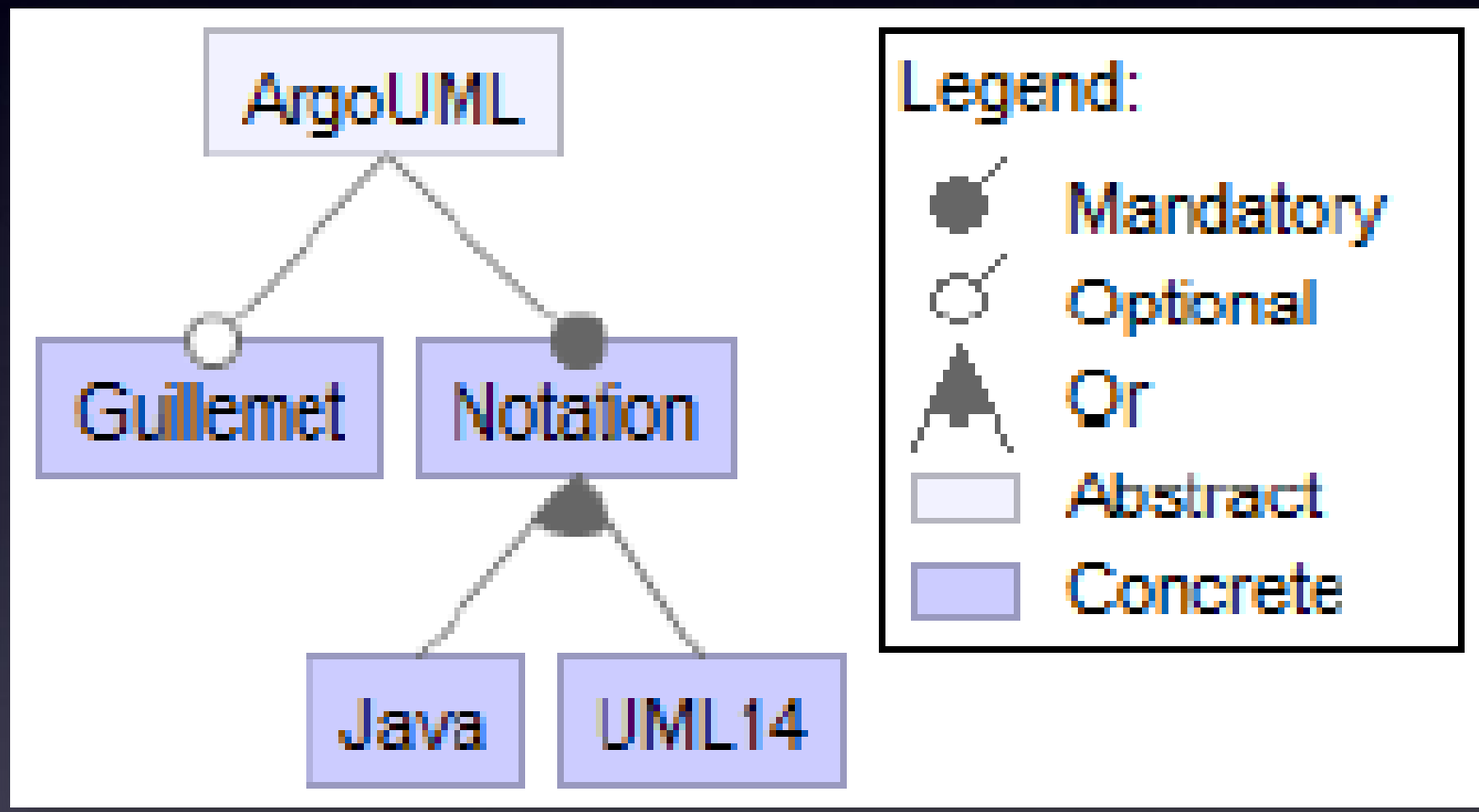
67000 SLOC



4000 feature SLOC

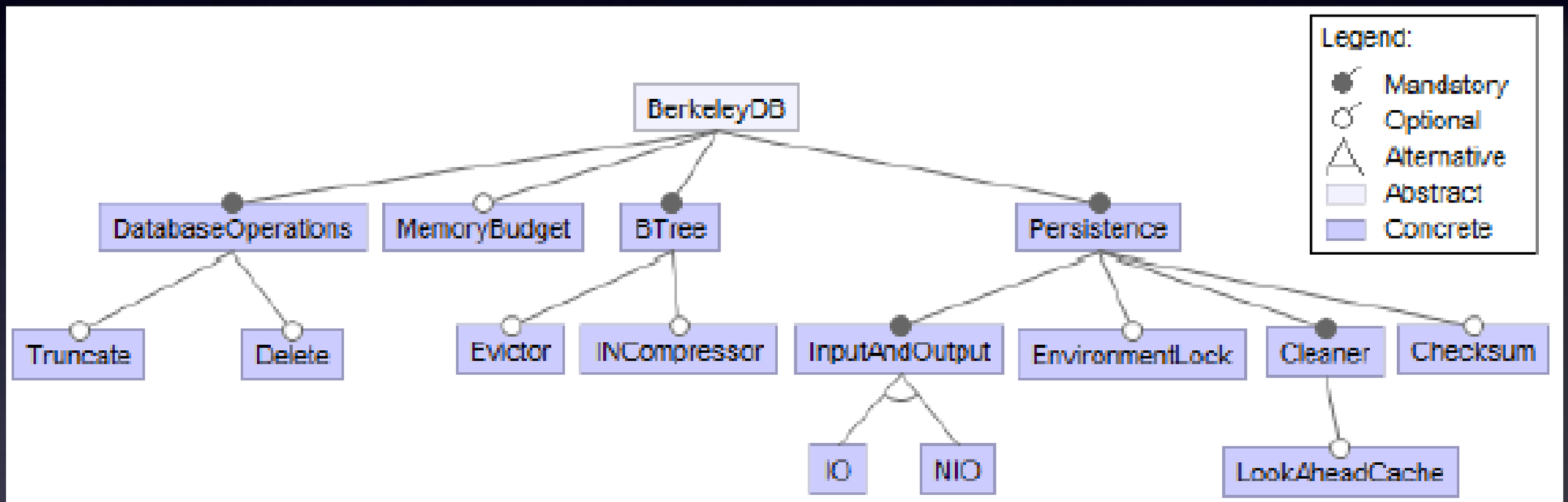


ArgoUML


ArgoUML
Gullemet
Notation
Java
UML14
Legend:
Mandatory
Optional
Or
Abstract
Concrete

BerkeleyDB

32000 SLOC



4000 feature SLOC

GQM

Goal

Questions

Metrics

Goal

Evaluate idioms to implement flexible binding time with respect to code quality factors and behavioral changes

Questions

Which idiom reduces:

1. Code duplication?
2. Driver and feature code scattering?
3. Tangling between driver and feature code?
4. Lines of code and number of components?

Cloning metric

- Pairs of Cloned Code (PCC)
  - Q1

Modularity metrics

- Degree of Scattering across Components (DOSC)
- Degree of Scattering across Operations (DOSO)
- Concern Diffusion over Components (CDC)
  - Q2

- Degree of Tangling within Components (DOTC)
- Degree of Tangling within Operations (DOTO)
  - Q3

Size metrics

- Source Lines of Code (SLOC)
- Vocabulary Size (VS)
  - Q4

Assessment procedures

1. Selection of case studies
2. Feature code identification and assignment
3. Code modularization
4. Flexible binding time implementation
5. Evaluate idioms

Evaluation

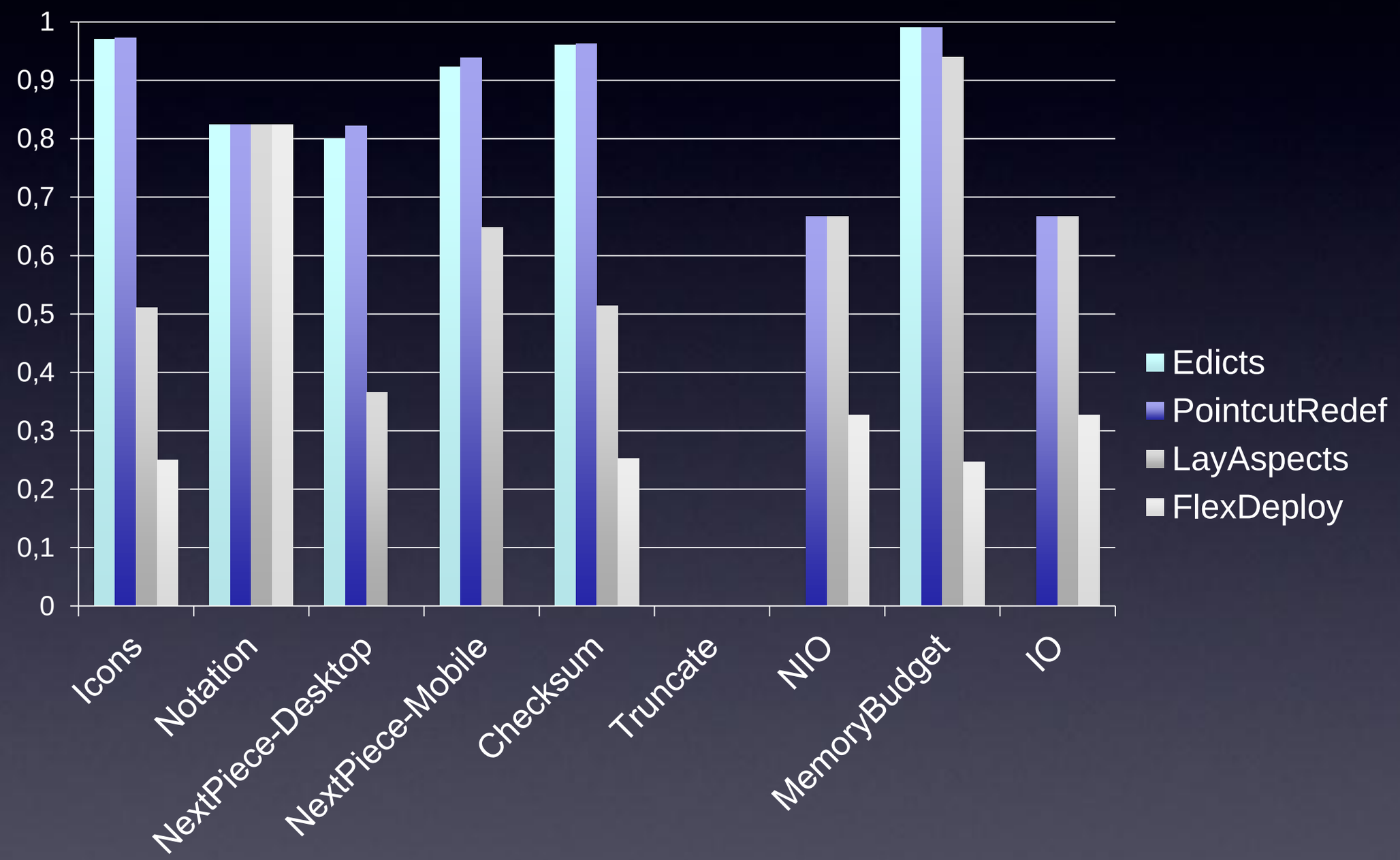
Clone

PCC

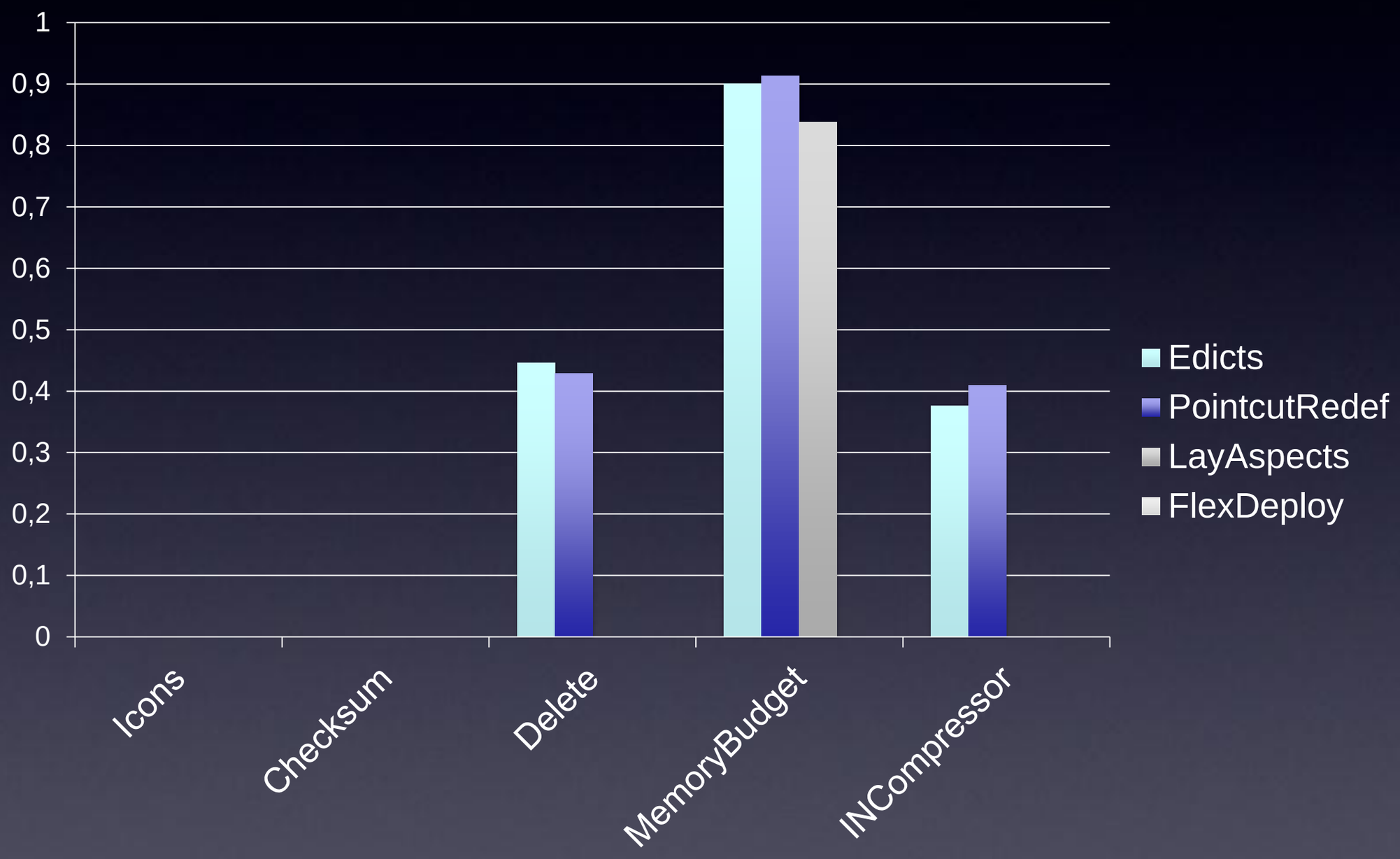
Application	Feature	Edicts	PointcutRedef	Layered Aspects	FlexDeploy
Freemind	Icons	19	4	5	4
	Clouds	5	1	1	1
ArgoUML	Notation	9	9	9	8
BerkeleyDB	Checksum	5	0	0	0
	Delete	9	3	2	0
	Memory Budget	34	17	7	1

Modularity

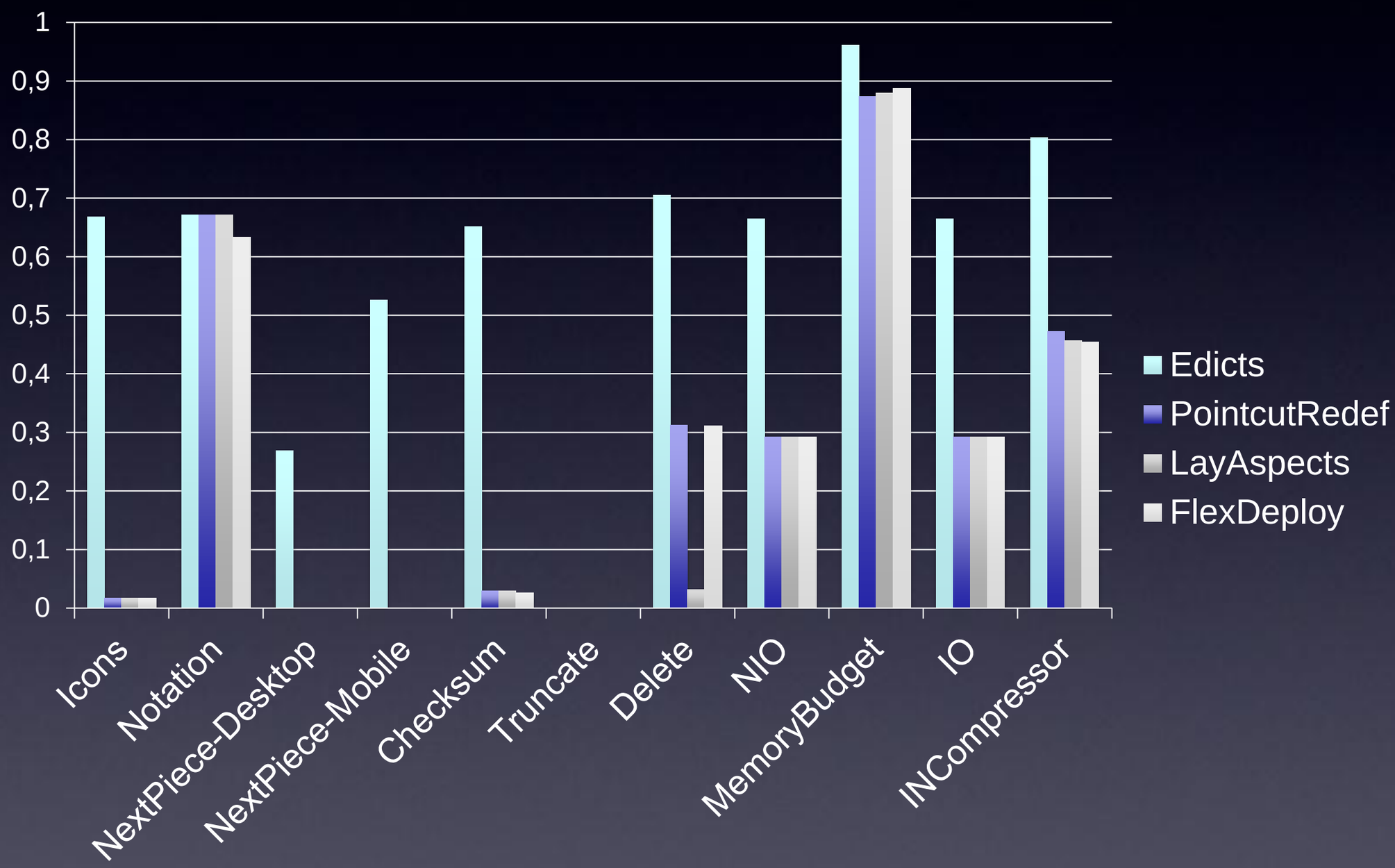
DOSO - Driver


1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
Icons
Notation
NextPiece-Desktop
NextPiece-Mobile
Checksum
Truncate
NIO
MemoryBudget
IO
Edicts
PointcutRedef
LayAspects
FlexDeploy

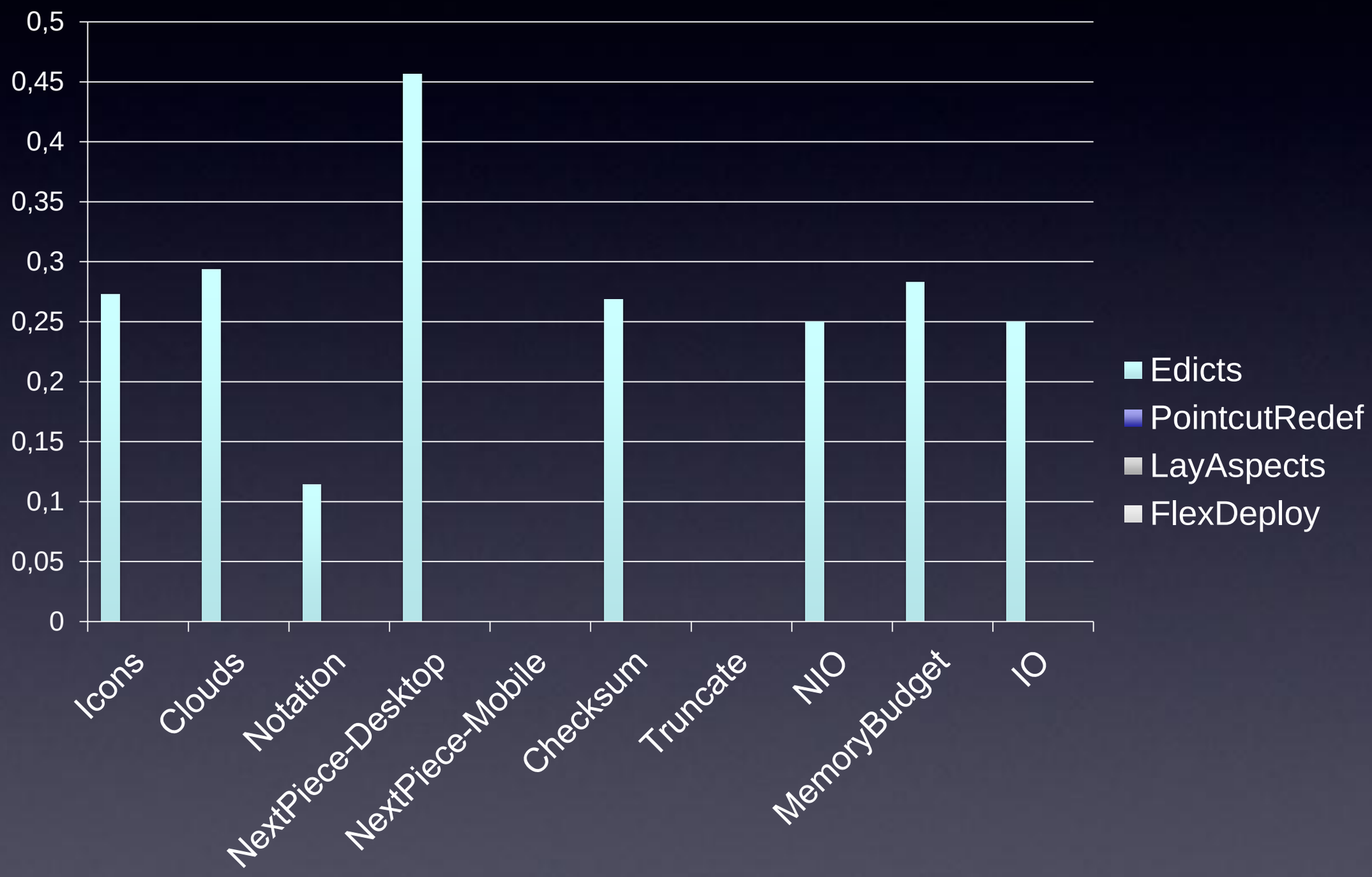
DOSC - Driver


1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
Icons
Checksum
Delete
MemoryBudget
INCompressor
Edicts
PointcutRedef
LayAspects
FlexDeploy

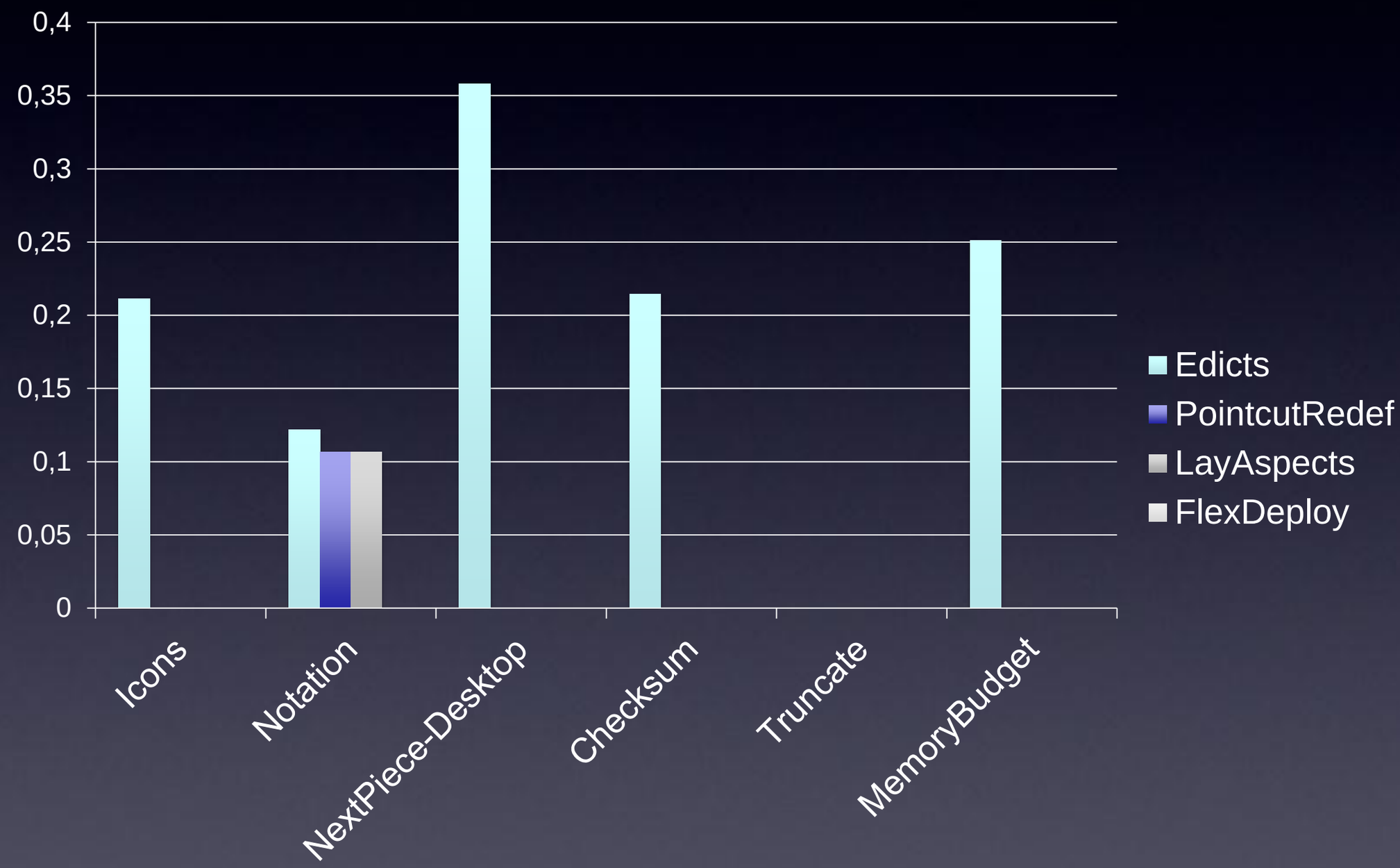
DOSC - Feature



DOTO


0,5
0,45
0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
Icons
Clouds
Notation
NextPiece-Desktop
NextPiece-Mobile
Checksum
Truncate
NIO
MemoryBudget
IO
Edicts
PointcutRedef
LayAspects
FlexDeploy

DOTC


0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
Icons
Notation
NextPiece-Desktop
Checksum
Truncate
MemoryBudget
Edicts
PointcutRedef
LayAspects
FlexDeploy

Size

SLOC

Application	Feature	Edicts	PointcutRedef	Layered Aspects	FlexDeploy
Freemind	Icons	2198	2180	2127	2031
	Clouds	2015	1919	1897	1833
ArgoUML	Notation	172	172	172	153
BerkeleyDB	Checksum	476	469	441	456
	Delete	460	358	338	358
	Memory Budget	1891	1397	1397	1285

VS

Application	Edicts	PointcutRedef	Layered Aspects	FlexDeploy
Freemind	550	551	549	557
ArgoUML	1622	1621	1621	1629
Tetris	21	21	21	-
Berkeley	345	338	345	350

Behavior

Safe Refactor tests

Application	Edicts and PCRedef	Edicts and LayAspects	PcRedef and LayAspects
Freemind	2005	2072	2072
ArgoUML	1443	1443	1443
Tetris	2394	2380	2384
BerkeleyDB	3362	3141	3141

Threats to validity

Case studies

Selected Features

Application refactoring

Multiple drivers

Metrics

Cloning results

Related work

Code Generation to Support Static and Dynamic Composition of Software Product Lines (Rosenmuller et al.)

Dynamic Software Product Line Approach using Aspect Models at Runtime (Dinkelaker et al.)

Future work

Memory Consumption

Performance

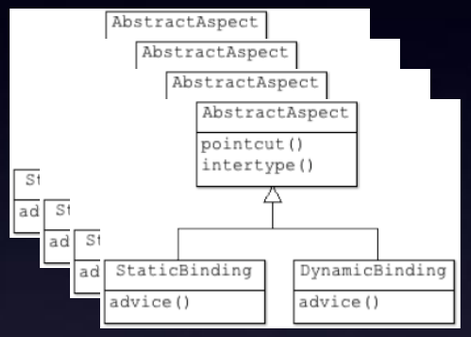
Idiom non AOP-based

More metrics

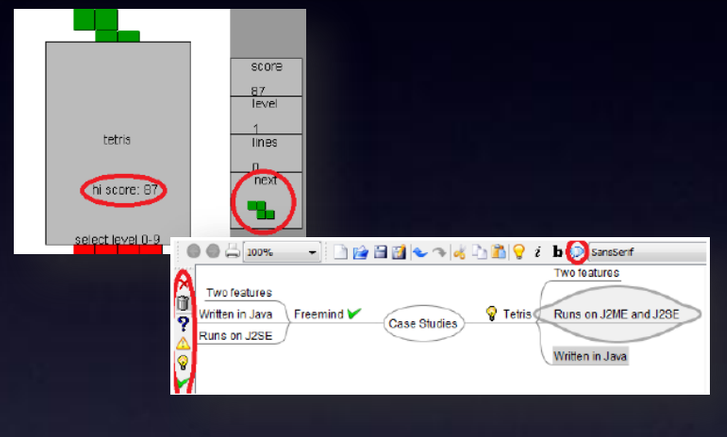
Multiple drivers

Conclusions

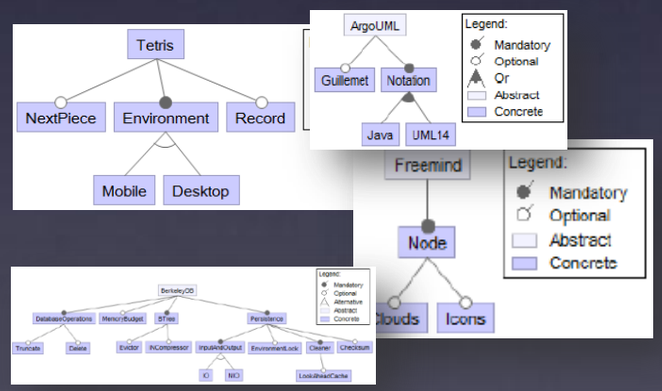
4 idioms



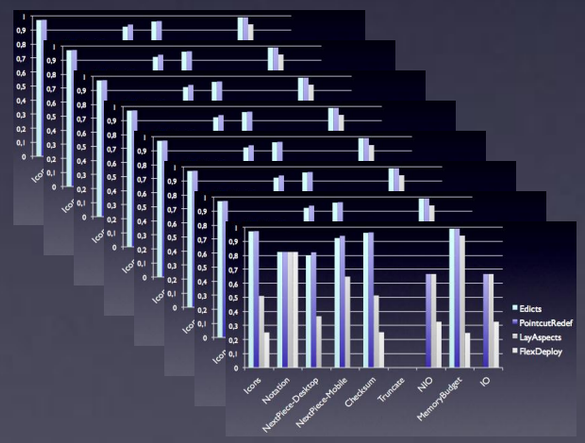
18 features



4 case studies



8 metrics



Idioms to Implement Flexible Binding Times for Features

Rodrigo Andrade
Advisor: Paulo Borba
CIn - UFPE