An *Empirical Study* on the Developers’ Perception of Software Coupling
“the measure of the \textit{strength} of association established by a connection from one module to another” [Stevens et al., 1974]
Coupling

Module 1

Module 2
Coupling

Measure coupling…

Module 1  Module 2
Coupling

Measure coupling…

Use coupling…
- predicting fault proneness
- change impact analysis
- software remodularization
- software reuse
- change propagation
- etc.
What are the Typical Types of Information used for Coupling?
What are the Typical Types of Information used for Coupling?

[Briand et al., TSE’99]: overview of structural metrics
What are the Typical Types of Information used for Coupling?

[Briand et al., TSE’99]: overview of *structural* metrics

Typical types of information:
- Structural
- Dynamic
- Semantic
- Logical
Structural
<table>
<thead>
<tr>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute_A</td>
</tr>
<tr>
<td>getAttributeA()</td>
</tr>
<tr>
<td>setAttributeA(...)</td>
</tr>
<tr>
<td>methodA1()</td>
</tr>
<tr>
<td>methodA2()</td>
</tr>
</tbody>
</table>
Methods sharing attributes are more likely to implement similar responsibility.
Methods grouped together by developers are more likely to implement similar responsibility.
Classes having inheritance relations are more likely to implement similar responsibility.
Calls between methods (Classes)
Structural

Dynamic
A

attribute_A
getAttributeA()
setAttributeA(…)
methodA1()
methodA2()

B

D

attribute_D
methodD1()
methodD2()
methodD3()

C

methodC1()
methodC2()

E

methodE1()
Execution Trace
A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
**Execution Trace**

A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
The calls that are not in the trace will not be used for computing dynamic coupling

**Execution Trace**
A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
The calls that are not in the trace will not be used for computing dynamic coupling.

Dynamic coupling is more precise than structural coupling.

Execution Trace:
A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
The calls that are not in the trace will not be used for computing dynamic coupling.

**Dynamic coupling** is more precise than **structural coupling**.

**Dynamic coupling** is harder to generate than **structural coupling**.

**Execution Trace**

A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
Structural
Dynamic
Semantic
/ * Insert a new user in the system.
  * @param pUser: the user to insert. */
public void insert(User pUser) {

    connect = DBConnection.getConnection();

    String sql = "INSERT INTO USER "
        + "(login,first_name,last_name,password" + ",email,cell,id_parent) " + "VALUES ("
        + pUser.getLogin() + ","
        + pUser.getFirstName() + ","
        + pUser.getLastName() + ","
        + pUser.getPassword() + ","
        + pUser.getEmail() + ","
        + pUser.getCell() + ","
        + pUser.getIdParent() + ")";

    executeOperation(connect, sql);
}

/* Delete an user from the system.  
  * @param pUser: the user to delete. */
public void delete(User pUser) {

    connect = DBConnection.getConnection();

    String sql = "DELETE FROM USER "
        + "WHERE id_user = "
        + pUser.getId();

    executeOperation(connect, sql);
}
/* Insert a new user in the system. */
* @param pUser: the user to insert */
public void insert(User pUser) {

    connect = DBCConnection.getConnect();

    String sql = "INSERT INTO USER"
            + "(login,first_name,last_name,password"
            + " ,email,cell,id_parent) " + "VALUES ("
            + pUser.getLogin() + ","
            + pUser.getFirstName() + ","
            + pUser.getLastName() + ","
            + pUser.getPassword() + ","
            + pUser.getEMail() + ","
            + pUser.getCell() + ","
            + pUser.getIdParent() + ")";

    executeOperation(connect, sql);
}

/* Delete an user from the system. */
* @param pUser: the user to delete. */
public void delete(User pUser) {

    connect = DBCConnection.getConnect();

    String sql = "DELETE FROM USER "
            + "WHERE id_user = "
            + pUser.getId();

    executeOperation(connect, sql);
}
/* Insert a new user in the system. 
 * @param pUser: the user to insert */
public void insert(User pUser) {
    connect = DBCConnection.getConnection();
    String sql = "INSERT INTO user " + "(login,first_name,last_name,password, " + "email,cell,id.parent) " + "VALUES " + pUser.getLogin() + "," + pUser.firstName() + "," + pUser.lastName() + "," + pUser.getPassword() + "," + pUser.getEmail() + "," + pUser.getCell() + "," + pUser.getIdParent() + ");";
    executeOperation(connect, sql);
}

/* Delete an user from the system. 
 * @param pUser: the user to delete. */
public void delete(User pUser) {
    connect = DBCConnection.getConnection();
    String sql = "DELETE FROM USER " + "WHERE id_user = " + pUser.getId();
    executeOperation(connect, sql);
}
/* Insert a new user in the system. */
public void insert(User pUser) {
    connect = DBCConnection.getConnection();
    String sql = "INSERT INTO user " + " (login,first_name,last_name,password, email,cell.id,parent) " + " VALUES (" + pUser.getLogin() + ",
+ pUser.getFirstName() + ",
+ pUser.getLastName() + ",
+ pUser.getPassword() + ",
+ pUser.getEmail() + ",
+ pUser.getCell() + ",
+ pUser.getIdParent() + ");"
    executeOperation(connect, sql);}

/* Delete an user from the system. */
/* @param pUser: the user to delete. */
public void delete(User pUser) {
    connect = DBCConnection.getConnection();
    String sql = "DELETE FROM user " + " WHERE id_user = " + pUser.getId();
    executeOperation(connect, sql);}
Compute textual similarity between documents "Textual" coupling...
Structural
Dynamic
Semantic
Logical (Evolutionary)
Changes to file A
A, D & E changed together often
B & C changed together often
Classes that often change together are likely to implement similar responsibility.
Which type of **coupling** aligns better with developers’ perception?
Which type of *coupling* aligns better with developers’ perception?

Empirical study with 76 developers.
Choosing Software Systems
<table>
<thead>
<tr>
<th>Size</th>
<th>200 KLOC</th>
<th>29 KLOC</th>
<th>28 KLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td># of classes</td>
<td>1,889</td>
<td>289</td>
<td>245</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>7,961</td>
<td>2,834</td>
<td>2,418</td>
</tr>
</tbody>
</table>

**JHotDraw**
<table>
<thead>
<tr>
<th></th>
<th>200 KLOC</th>
<th>29 KLOC</th>
<th>28 KLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>200</td>
<td>29</td>
<td>28</td>
</tr>
<tr>
<td># of classes</td>
<td>1,889</td>
<td>289</td>
<td>245</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>7,961</td>
<td>2,834</td>
<td>2,418</td>
</tr>
<tr>
<td></td>
<td>200 KLOC</td>
<td>29 KLOC</td>
<td>28 KLOC</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>Size</td>
<td>200 KLOC</td>
<td>29 KLOC</td>
<td>28 KLOC</td>
</tr>
<tr>
<td># of classes</td>
<td>1,889</td>
<td>289</td>
<td>245</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>7,961</td>
<td>2,834</td>
<td>2,418</td>
</tr>
<tr>
<td>Historical</td>
<td>8 years</td>
<td>5 years</td>
<td>12 years</td>
</tr>
<tr>
<td></td>
<td>200 KLOC</td>
<td>29 KLOC</td>
<td>28 KLOC</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>200 KLOC</td>
<td>29 KLOC</td>
<td>28 KLOC</td>
</tr>
<tr>
<td><strong># of classes</strong></td>
<td>1,889</td>
<td>289</td>
<td>245</td>
</tr>
<tr>
<td><strong>Vocabulary</strong></td>
<td>7,961</td>
<td>2,834</td>
<td>2,418</td>
</tr>
<tr>
<td><strong>Historical</strong></td>
<td>8 years</td>
<td>5 years</td>
<td>12 years</td>
</tr>
<tr>
<td><strong>Execution Traces</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Coverage</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
No test cases available…
No test cases available…
No test cases available...
No test cases available…

ArgoUML

Trace 1

Trace 2

Trace i
<table>
<thead>
<tr>
<th></th>
<th>Size 200 KLOC</th>
<th>Size 29 KLOC</th>
<th>Size 28 KLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td># of classes</td>
<td>1,889</td>
<td>289</td>
<td>245</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>7,961</td>
<td>2,834</td>
<td>2,418</td>
</tr>
<tr>
<td>Historical</td>
<td>8 years</td>
<td>5 years</td>
<td>12 years</td>
</tr>
<tr>
<td>Execution Traces Coverage</td>
<td>66%</td>
<td>67%</td>
<td>86%</td>
</tr>
</tbody>
</table>
Choosing Software Systems

Choosing Participants
Original Developers

JHotDraw
Original Developers

6 3 3

JHotDraw
Original Developers:
- 6 developers
- 3 faculty
- 33 MS students
- 3 undergraduates

External Developers:
- 64 developers
- 5 with industrial experience
- 7 faculty
- 16 PhD students
- 33 MS students
- 3 undergraduates
Original developers evaluated their own system
External developers evaluated all systems
Choosing Software Systems

Choosing Participants

Coupling Metrics
/* Insert a new user in the system. */
public void insert(User user) {
    connect = DBCConnection.getConnection();
    String sql = "INSERT INTO USER"
    + " (login, first_name, last_name, password"
    + " email, cell, id parent) " VALUES (" +
    + user.getLogin() + ", " +
    + user.getFirstName() + " , " +
    + user.getLastName() + " , " +
    + user.getPassword() + " , " +
    + user.getCell() + " , " +
    + user.getIdParent() + ");"
    executeOperation(connect, sql);
}

/* Delete an user from the system. */
public void delete(User user) {
    connect = DBCConnection.getConnection();
    String sql = "DELETE FROM USER"
    + " WHERE id_user = " +
    + user.getId();
    executeOperation(connect, sql);
}

Execution Trace
A.methodA2() → C.methodC2()
C.methodC2() → E.methodE1()
**Structural Coupling:**
ICP: Information flow-based coupling [Lee et al.'95]

**Dynamic Coupling:**
IC_CD: Import Coupling Class Dynamic Message [Arisholm et al.'04]

**Semantic Coupling:**
CCBC: Conceptual Coupling Between Classes [Poshyvanyk et al.'08]

**Logical Coupling:**
Association rule-based Change Coupling [Ying et al.'04] [Zimmerman et al.'04]
Choosing Software Systems

Choosing Participants

Coupling Metrics

Data Collection
Structural
<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.88</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>
**High Structural Coupling:**
- Top 2 pairs

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.88</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>
### High Structural Coupling:
- Top 2 pairs

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.88</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>

### Low Structural Coupling:
- Bottom 2 pairs
package org.jhotdraw.standard;

import java.awt.Cursor;

/**
 * Default implementation of the (link org.jhotdraw
 * for AWT/Swing.
 *<p>
 * @version 1.3 @
 * @author Ricardo Padilha</a>
 * @see org.jhotdraw.framework.Cursor
 */

public class AWTCursor extends Cursor implements org.jhotdraw.framework.Cursor {

/**
 * Constructor for <code>AWTCursor</code>.
 * @param type
 * @see Cursor#Cursor(int)
 */

public AWTCursor(int type) {
    super(type);
}

package org.jhotdraw.framework;

import org.jhotdraw.util.Storable;

/**
 * Locators can be used to locate a position on a figure.
 *<p>
 * @version Design Patterns</b></p>
 * <img src="images/red-ball-small.gif" width=6 height=
 * @param type
 * @see Locator#Locator(int)
 */

public interface Locator extends Storable, Serializable{

/**
 * Locates a position on the passed figure.
 * @return a point on the figure.
 */

public Point locate(Figure owner);
```java
package org.jhotdraw.standard;

import java.awt.Cursor;

/**
 * Default implementation of the [link org.jhotdraw framework for AWT/Swing.
 * 
 * @version $Revision: 1.3 $
 * @author <a href="mailto:ricardo_padilha@users.sourceforge.net">Sangoi Padilha</a>
 * @see org.jhotdraw.framework.Cursor
 */

class AWTCursor extends Cursor implements org.jhotdraw.framework.Scrollable {

    /**
     * Constructor for <code>AWTCursor</code>.
     * @param type
     * @see Cursor#Cursor(int)
     */
    public AWTCursor(int type) {
        super(type);
    }
}
```
### Structural

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.86</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>V</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>

### Dynamic

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.86</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>V</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>

### Semantic

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.86</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>V</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>

### Logical

<table>
<thead>
<tr>
<th>Class1</th>
<th>Class2</th>
<th>Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.95</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
<td>0.90</td>
</tr>
<tr>
<td>M</td>
<td>P</td>
<td>0.86</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>V</td>
<td>T</td>
<td>0.25</td>
</tr>
<tr>
<td>S</td>
<td>U</td>
<td>0.20</td>
</tr>
<tr>
<td>V</td>
<td>B</td>
<td>0.18</td>
</tr>
</tbody>
</table>
JHotDraw

Structural

Dynamic

Semantic

Logical

16 pairs of classes / system
Choosing Software Systems

Choosing Participants

Coupling Metrics

Data Collection

Results
Likert scale for coupling perceived by developers
Types of high coupling
Types of high coupling

If developers agree with *high* coupling, results should be here...
If developers agree with *low* coupling, results should be here...

<table>
<thead>
<tr>
<th>High Semantic</th>
<th>High Structural</th>
<th>High Dynamic</th>
<th>High Logical</th>
<th>Low Semantic</th>
<th>Low Structural</th>
<th>Low Dynamic</th>
<th>Low Logical</th>
</tr>
</thead>
</table>

Types of *low* coupling
Statistical analysis
Classes with high semantic measure are perceived as **strong coupled**
Semantic outperforms others \([p-value<0.05]\)
Structural and dynamic align well with developers’ perception.
Logical is outperformed by others.
Qualitative example...
Original developer:

“these classes have a very high coupling even if there is no cooperation between them”
Semantic measure is the only one capturing coupling between these classes

Original developer:

“these classes have a very high coupling even if there is no cooperation between them”
Classes with low semantic measure are perceived as **loose coupled**
Semantic outperforms others
[p-value<0.05]
Wider distribution of values...why?
Classes with low dynamic coupling, but ranked as strong coupled.
“even if indirectly, the two classes exhibit some form of coupling since they implement similar jobs and changes to the GUI are likely to involve both classes”
JHotDraw
Classes with high semantic measure are perceived as **strong coupled**.
Semantic outperforms others \( [p-value<0.05] \)
Structural is most unsuited to capture coupling.
Dynamic and semantic are the only ones that properly capture low coupling.
Structural produces strange results… Why?
ZoomTool

```java
public class ZoomTool extends AbstractTool {

    private Tool child;

    public ZoomTool(DrawingEditor editor) {
        super(editor);
    }

    public void mouseDown(MouseEvent e, int x, int y) {
        super.mouseDown(e, x, y);
        // Added handling for SHIFTed and CTRLed BUTTON3 MASK so that normal
        // BUTTON3 MASK does zoomOut, SHIFTed BUTTON3_MASK does zoomIn
        // and CTRLed BUTTON3 MASK does nothing.
        child = new ZoomAreaTracker(editor);
        child.mouseDown(e, x, y);
    }
}
```

ZoomUpdateStrategy

```java
public class ZoomUpdateStrategy implements Painter {

    public void paint(Graphics g, int d, int d2) {
        // The offscreen image
        // transient private Image fOffscreen;
        private int fImagewidth = -1;
        private int fImageheight = -1;

        /*
        * Draw the view contents.
        */
        fImagewidth = d.width;
        fImageheight = d.height;
    }
}
```

Classes with low coupling => perceived as coupled
Classes with low coupling => perceived as coupled

Original developer:

“these classes are peer features participating in the same context”
Classes with low coupling => perceived as coupled

Original developer:

"these classes are peer features participating in the same context"

Semantic measure captures this coupling (CCBC=0.52)
Semantic measure is not a silver bullet...

```
public class AWTCursor extends Cursor implements org.jhotdraw.framework.Cursor {
    /** Constructor for <code>AWTCursor</code>.
     * @param type
     * @param Cursor#Cursor(int)
     */
    public AWTCursor(int type) {
        super(type);
    }
}
```

```
public interface Locator extends Storable, Serializable, Cloneable {
    /**
     * Locate a position on the passed figure.
     * @param Figure owner
     */
    public Point locate(Figure owner);
}
```

Not coupled, but semantic measure is high (CCBC=0.41)
Same trends as other systems...
How can these types of coupling affect the remodularization of a software system?
Original software structure

Package 1
A
B
C

Package 2
D
E
F
G

Package n
H
I

Structural dependencies

A → C
B → G
D → E
I → D

Bunch
[Mitchell and Mancoridis’06]
Original software structure

Package 1

A

B

C

Package 2

D

E

F

G

Package n

H

I

Structural dependencies

A → C
B → G
D → E
I → D

Package i

A

B

D

H

Package j

I

E

F

C

Bunch

[Mitchell and Mancoridis’06]
Original software structure

Package 1
A
B
C

Package 2
D
E
F
G

Package n
H
I

Structural dependencies

Package i
A
B
D
H

Package j
I
E
F
C

Bunch
[Mitchell and Mancoridis’06]
Original software structure

Package 1
A
B
C

Package 2
D
E
F
G

Package n
H
I

MoJoFM
[Wen and Tzerpos’04]

Package i
A
B
D
H

Package j
I
E
F
C

Bunch
[Mitchell and Mancoridis’06]

Structural dependencies
How many *move* and *join* operations are required to transform the produced modularization to the original one?
Median MoJoFM for 30 runs of Bunch (to account for randomness)

<table>
<thead>
<tr>
<th></th>
<th>ArgoUML</th>
<th>JE</th>
<th>JHotDraw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structural</td>
<td>62%</td>
<td>43%</td>
<td>59%</td>
</tr>
<tr>
<td>Semantic</td>
<td>69%</td>
<td>49%</td>
<td>73%</td>
</tr>
<tr>
<td>Logical</td>
<td>3%</td>
<td>3%</td>
<td>6%</td>
</tr>
<tr>
<td>Dynamic</td>
<td>29%</td>
<td>26%</td>
<td>59%</td>
</tr>
</tbody>
</table>
Median MoJoFM for 30 runs of Bunch (to account for randomness)

<table>
<thead>
<tr>
<th></th>
<th>ArgoUML</th>
<th>JE</th>
<th>JHotDraw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structural</td>
<td>62%</td>
<td>43%</td>
<td>59%</td>
</tr>
<tr>
<td>Semantic</td>
<td>69%</td>
<td>49%</td>
<td>73%</td>
</tr>
<tr>
<td>Logical</td>
<td>3%</td>
<td>3%</td>
<td>6%</td>
</tr>
<tr>
<td>Dynamic</td>
<td>29%</td>
<td>26%</td>
<td>59%</td>
</tr>
</tbody>
</table>

Structural and semantic coupling produce better remodularity
Median MoJoFM for 30 runs of Bunch (to account for randomness)

Maybe not enough data?
Threats to Validity

• Metrics frequently used in the literature
• Performance of *dynamic* and *logical* metrics may have been impacted by insufficient data
• Pairs of classes may not be representative
• Remodularization task: use of current modularization of the system as *oracle*
• Choice of coupling measures
Conclusions

• Empirical study to assess the developer perception on software coupling
  – 76 developers
• Investigate which type of information is more suitable for remodularization
• We need more studies to generalize the results
Future Work

• Consider systems with more dynamic and logical information
• Use different coupling measures
• Consider combinations of coupling metrics
Thank you! Questions?

http://www.distat.unimol.it/reports/coupling
References

# Overlap

<table>
<thead>
<tr>
<th>#Links</th>
<th>#Exclusive links</th>
<th>ArgoUML</th>
<th>JHotDraw</th>
<th>jEdit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Structural</td>
<td>Semantic</td>
<td>Logical</td>
</tr>
<tr>
<td>124,346</td>
<td>48,471</td>
<td>-</td>
<td>61%</td>
<td>1%</td>
</tr>
<tr>
<td>1,078,663</td>
<td>1,002,358</td>
<td>7%</td>
<td>-</td>
<td>1%</td>
</tr>
<tr>
<td>180</td>
<td>0</td>
<td>24%</td>
<td>88%</td>
<td>-</td>
</tr>
<tr>
<td>3,138</td>
<td>89</td>
<td>89%</td>
<td>81%</td>
<td>1%</td>
</tr>
</tbody>
</table>

|        |                  | Structural | Semantic | Logical | Dynamic |
| 5,029  | 3,042            | -         | 37%     | 1%     | 5%     |
| 9,613  | 7,665            | 20%       | -       | 1%     | 2%     |
| 52     | 6                | 37%       | 39%     | -      | 29%    |
| 358    | 31               | 85%       | 64%     | 4%     | -      |

|        |                  | Structural | Semantic | Logical | Dynamic |
| 1,924  | 822              | -         | 54%     | 1%     | 13%    |
| 21,152 | 19,953           | 5%        | -       | 1%     | 2%     |
| 87     | 5                | 52%       | 40%     | -      | 14%    |
| 453    | 50               | 84%       | 75%     | 2%     | -      |