Universidade Federal Fluminense

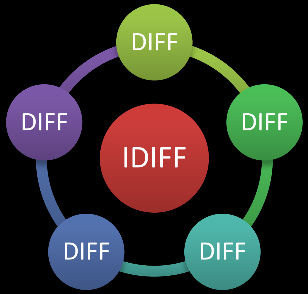
INTRODUCTION

MOTIVATING EXAMPLE

ITERATIVE DIFF (IDIFF)

EVALUATION

CONCLUSIONS



The Brazilian Symposium on
Software Engineering (SBES)

Instituto de Computação

Towards a Difference Detection Algorithm aware of Refactoring-related Changes

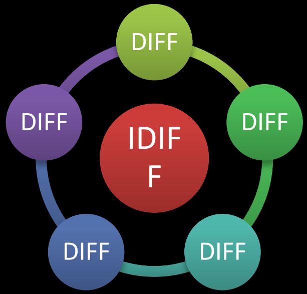
Fernanda Silva[1] Eraldo Borel[1] Evandro Lopes[2] **Leonardo Murta**[1]

[1] Computing Institute
Fluminense Federal University (UFF)
Niterói, Rio de Janeiro, Brazil
e-mail: {ffloriano,leomurta}@ic.uff.br
eraldoborel@id.uff.br

[2] Department of Statistics
Fluminense Federal University (UFF)
Niterói, Rio de Janeiro, Brazil
e-mail: evandro_dalbem@id.uff.br

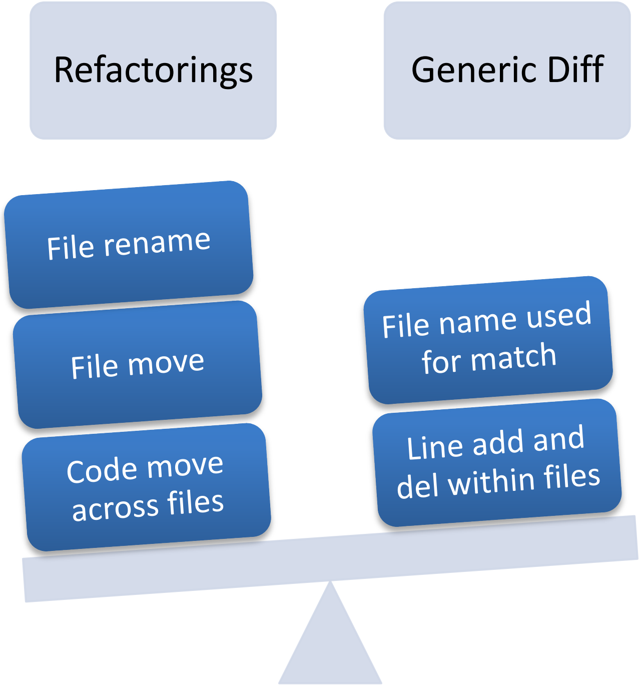
GEMS
Software Maintenance and Evolution Group

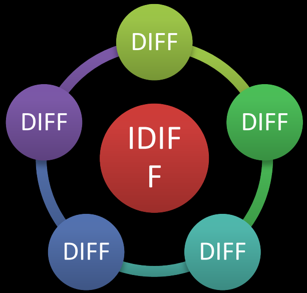
CBSoft2014
Congresso Brasileiro de Software: Teoria e Prática
28 de Setembro a 03 de Outubro de 2014
Maceió - Alagoas



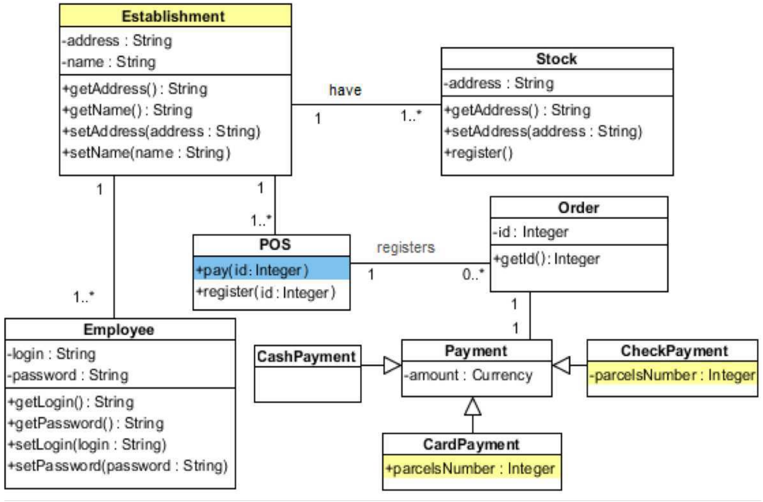
MOTIVATION

- Refactorings are a usual practice during software development
- At the physical level, **refactorings** imply **file renames and moves** and **code snippets moves across files**
- However, current **generic diff tools** detect **lines additions and deletions within files**



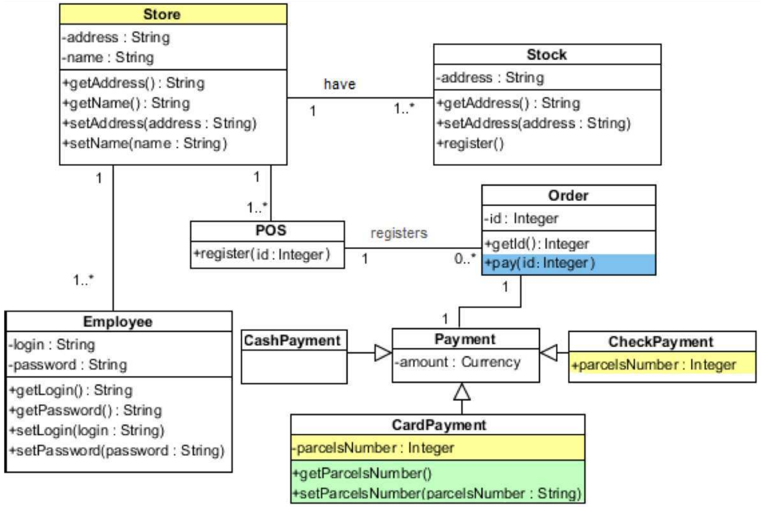


POS (point-of-sale) system, implemented in Java

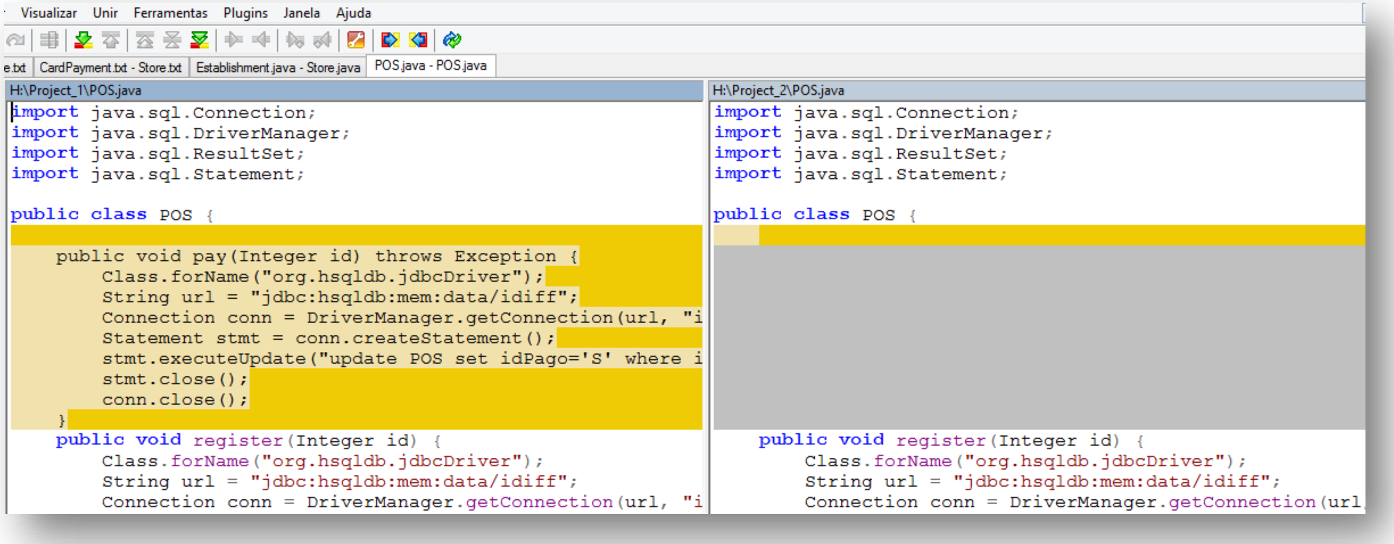


Refactoring applied over the base version:

Move Method
Rename Method
Encapsulate Field



POS (point-of-sale) system, implemented in Java

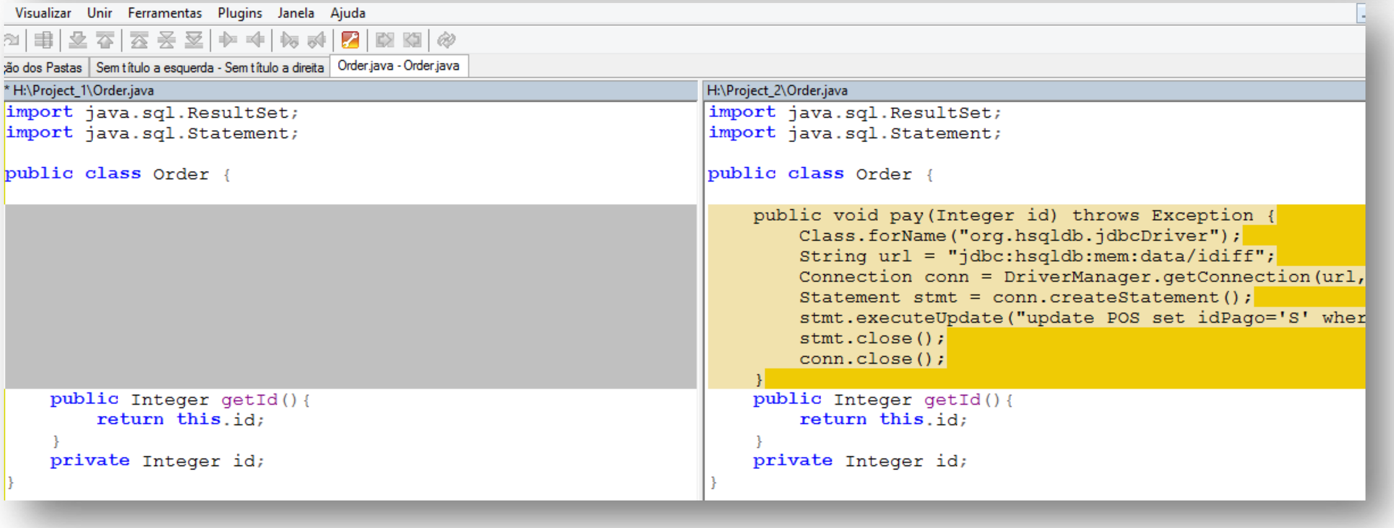


H:\Project_1\POS.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class POS {

    public void pay(Integer id) throws Exception {
        Class.forName("org.hsqldb.jdbcDriver");
        String url = "jdbc:hsqldb:mem:data/idiff";
        Connection conn = DriverManager.getConnection(url, "i
        Statement stmt = conn.createStatement();
        stmt.executeUpdate("update POS set idPago='S' where i
        stmt.close();
        conn.close();
    }
    public void register(Integer id) {
        Class.forName("org.hsqldb.jdbcDriver");
        String url = "jdbc:hsqldb:mem:data/idiff";
        Connection conn = DriverManager.getConnection(url, "i
```

H:\Project_2\POS.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class POS {

    public void register(Integer id) {
        Class.forName("org.hsqldb.jdbcDriver");
        String url = "jdbc:hsqldb:mem:data/idiff";
        Connection conn = DriverManager.getConnection(url
```

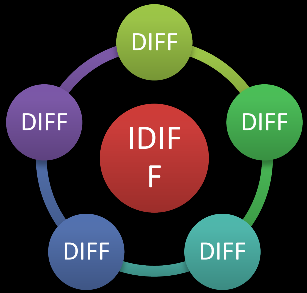


* H:\Project_1\Order.java

```
import java.sql.ResultSet;
import java.sql.Statement;

public class Order {

    public Integer getId(){
        return this.id;
    }
    private Integer id;
}
```

H:\Project_2\Order.java

```
import java.sql.ResultSet;
import java.sql.Statement;

public class Order {

    public void pay(Integer id) throws Exception {
        Class.forName("org.hsqldb.jdbcDriver");
        String url = "jdbc:hsqldb:mem:data/idiff";
        Connection conn = DriverManager.getConnection(url,
        Statement stmt = conn.createStatement();
        stmt.executeUpdate("update POS set idPago='S' wher
        stmt.close();
        conn.close();
    }
    public Integer getId(){
        return this.id;
    }
    private Integer id;
}
```



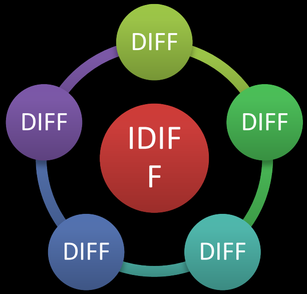
INTRODUCTION

MOTIVATING EXAMPLE

ITERATIVE DIFF (IDIFF)

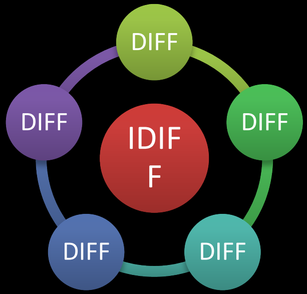
EVALUATION

CONCLUSIONS



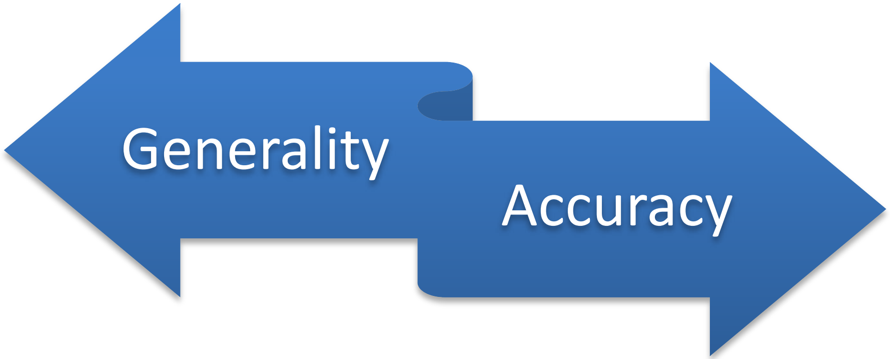
GOAL

Conceive a generic diff algorithm that precisely identify refactoring-related changes

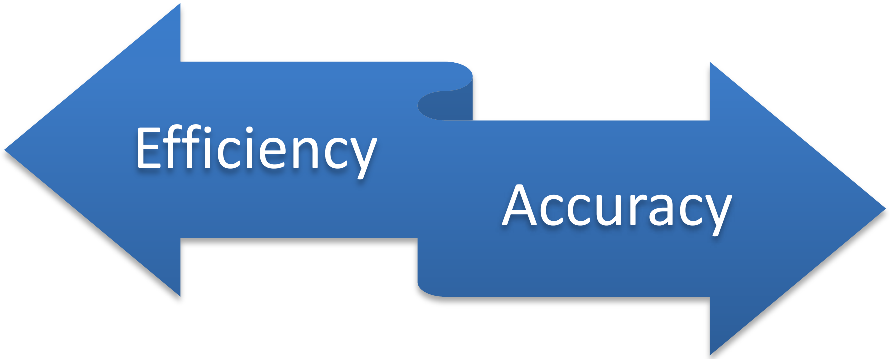


TRADEOFFS

Data model



Granularity



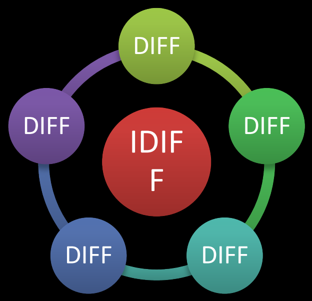
INTRODUCTION

MOTIVATING EXAMPLE

ITERATIVE DIFF (IDIFF)

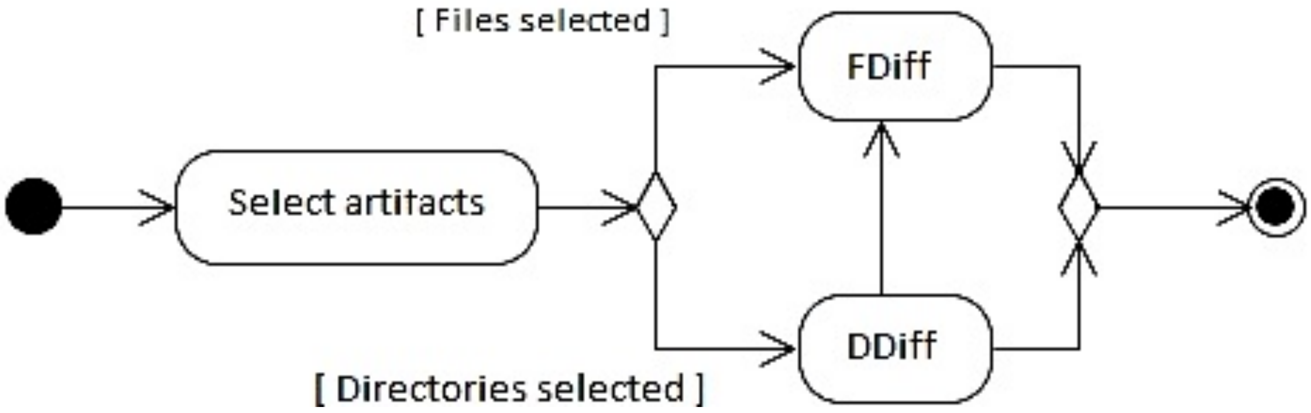
EVALUATION

CONCLUSIONS

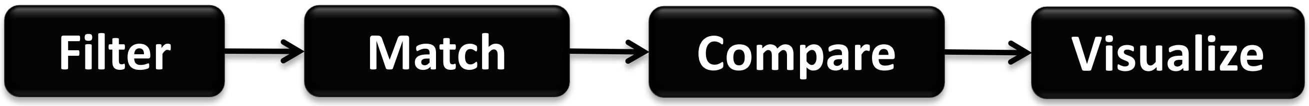


IDIFF

- **Approach overview:**



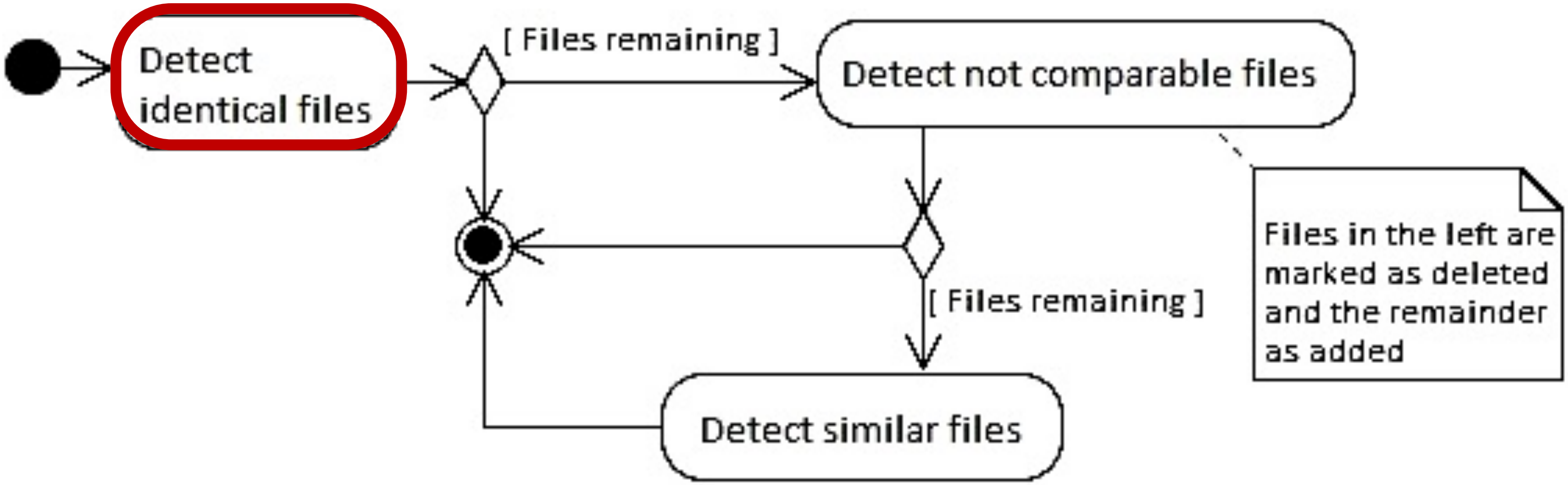
- **Approach steps:**



Filter → **Match** → **Compare** → **Visualize**

Filter → Match → Compare → Visualize

DDIFF



Detect identical files

[Files remaining]

Detect not comparable files

Files in the left are marked as deleted and the remainder as added

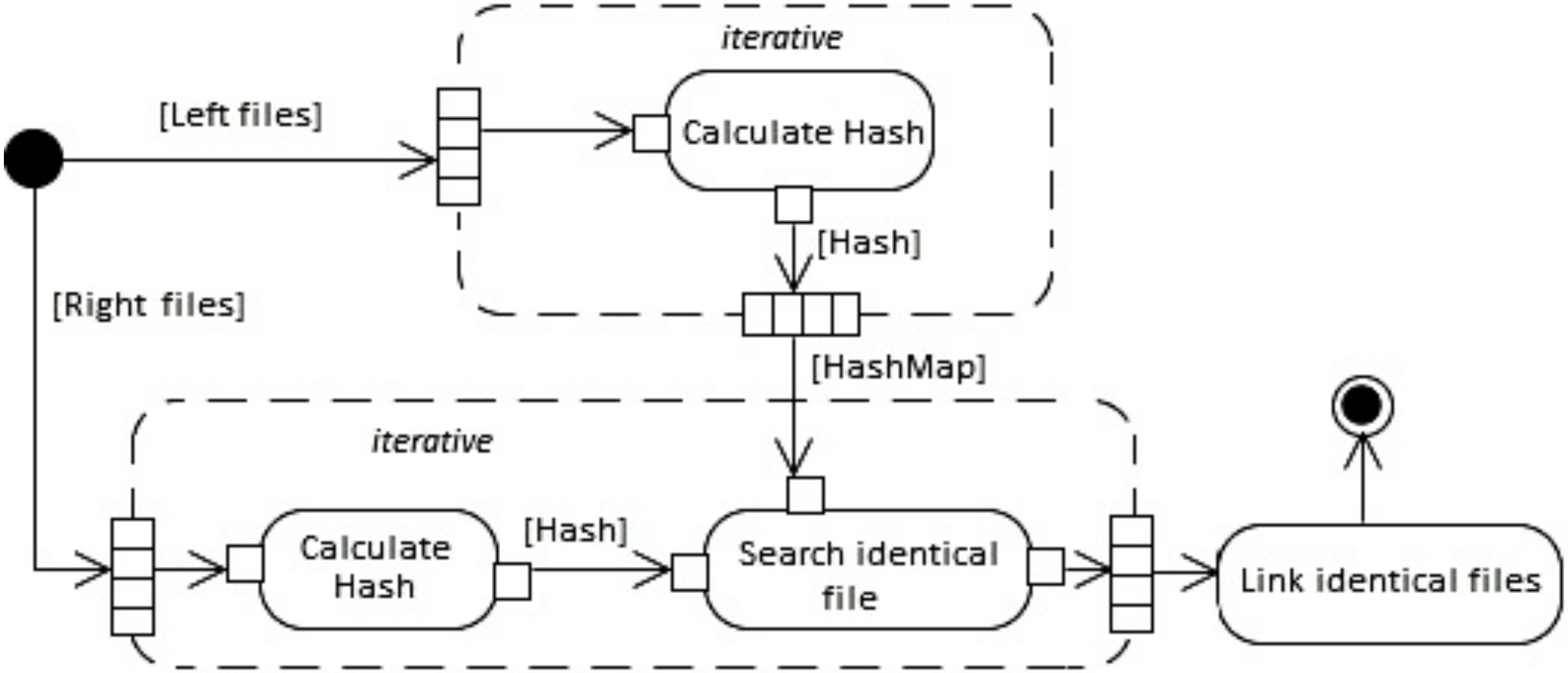
[Files remaining]

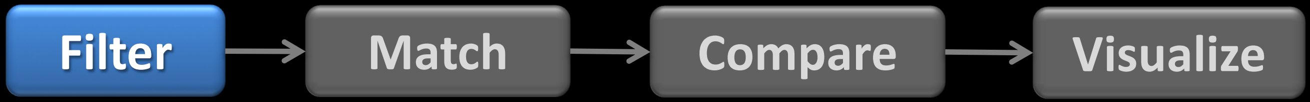
Detect similar files

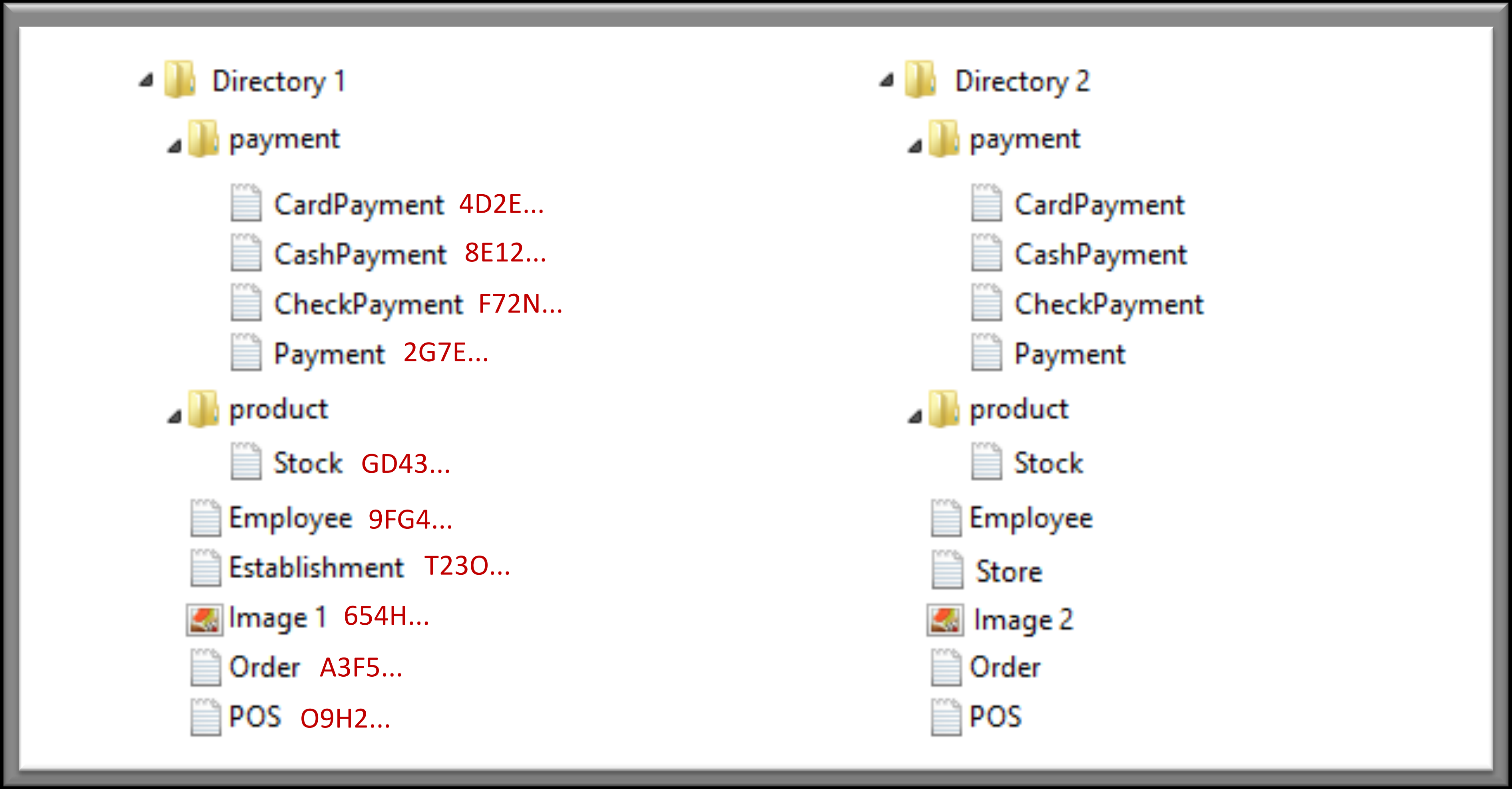
“two successive revisions are often very similar (98% similar in average)”

Jacky Estublier

DDIFF - Detect identical files

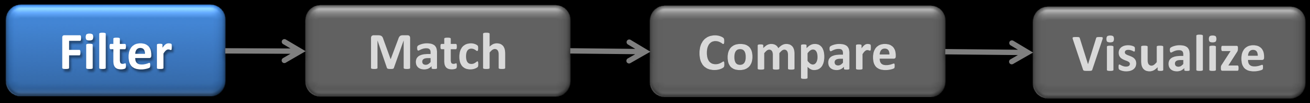


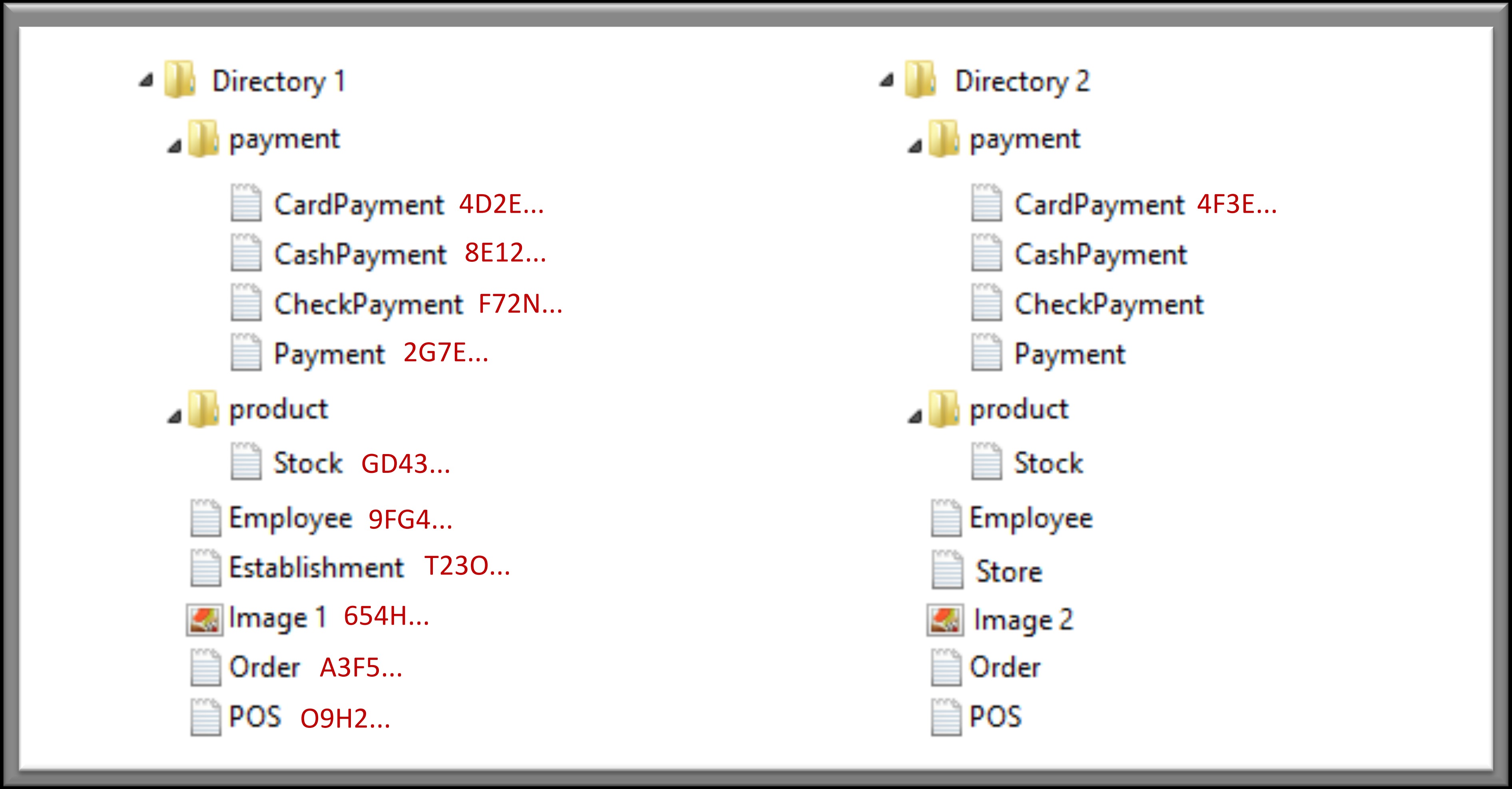




- Directory 1
  - payment
    - CardPayment 4D2E...
    - CashPayment 8E12...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

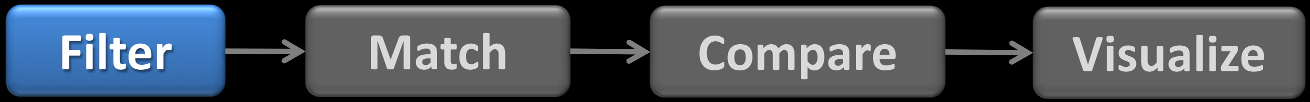
- Directory 2
  - payment
    - CardPayment
    - CashPayment
    - CheckPayment
    - Payment
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

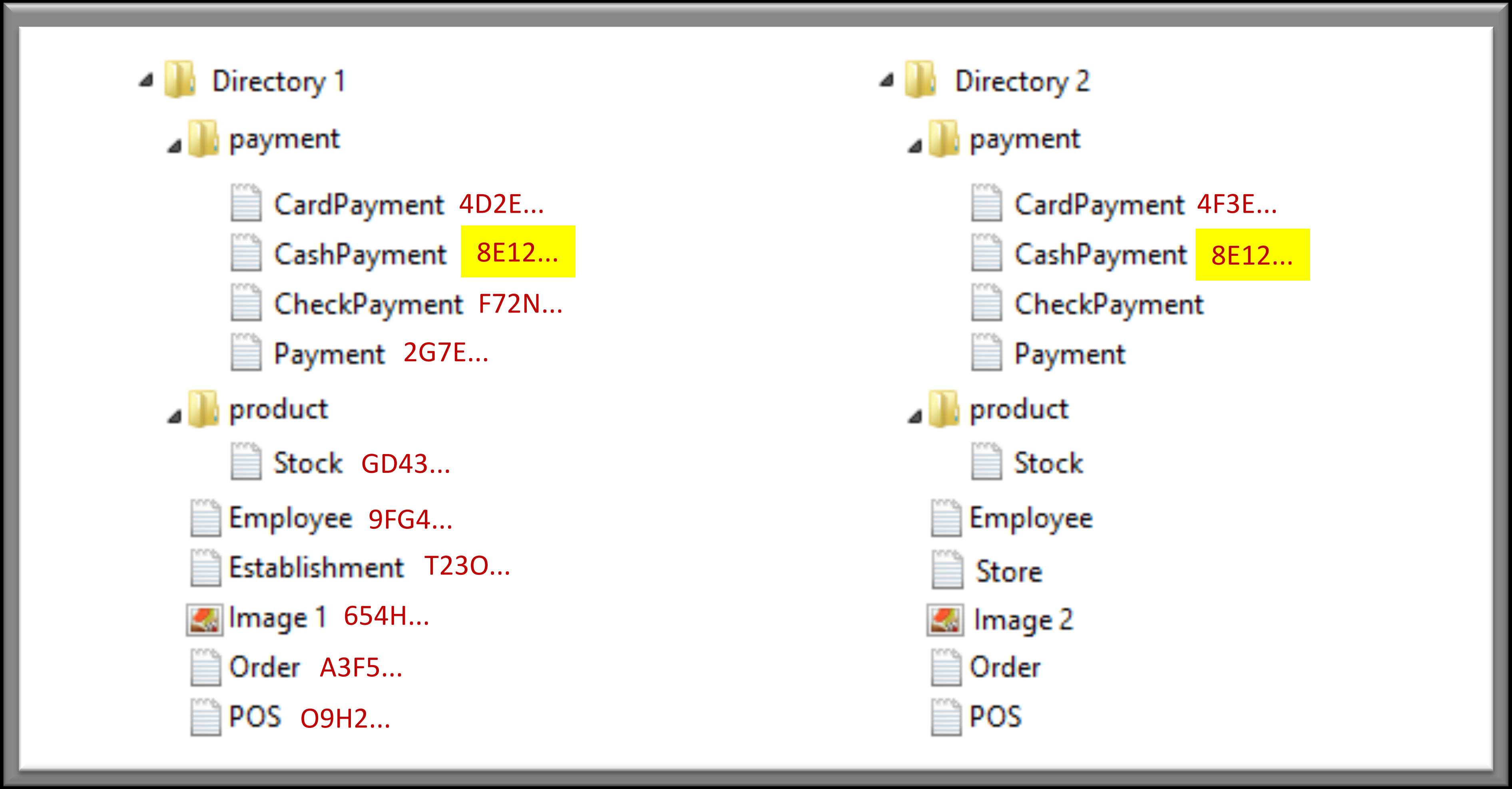




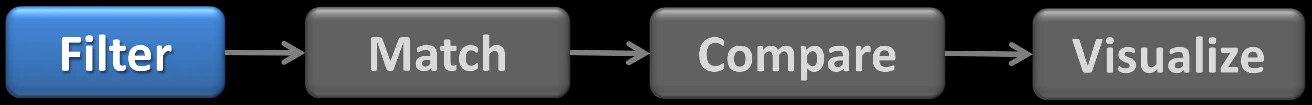
- Directory 1
  - payment
    - CardPayment 4D2E...
    - CashPayment 8E12...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CashPayment
    - CheckPayment
    - Payment
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

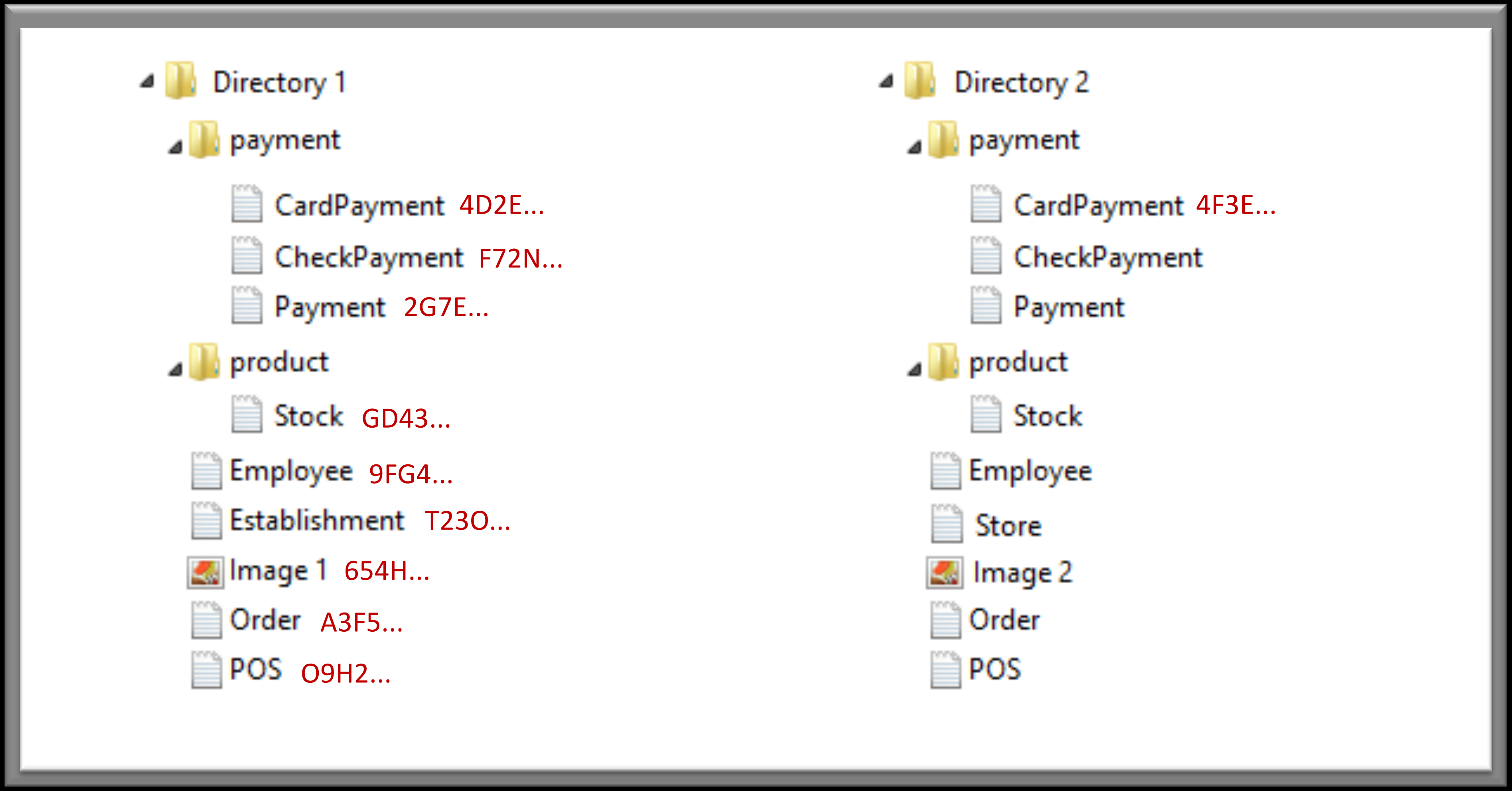

Filter → Match → Compare → Visualize


- Directory 1
  - payment
    - CardPayment 4D2E...
    - CashPayment 8E12...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CashPayment 8E12...
    - CheckPayment
    - Payment
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

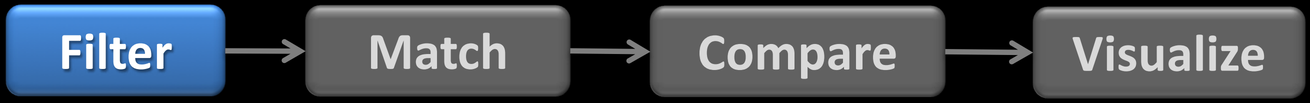


Filter → Match → Compare → Visualize

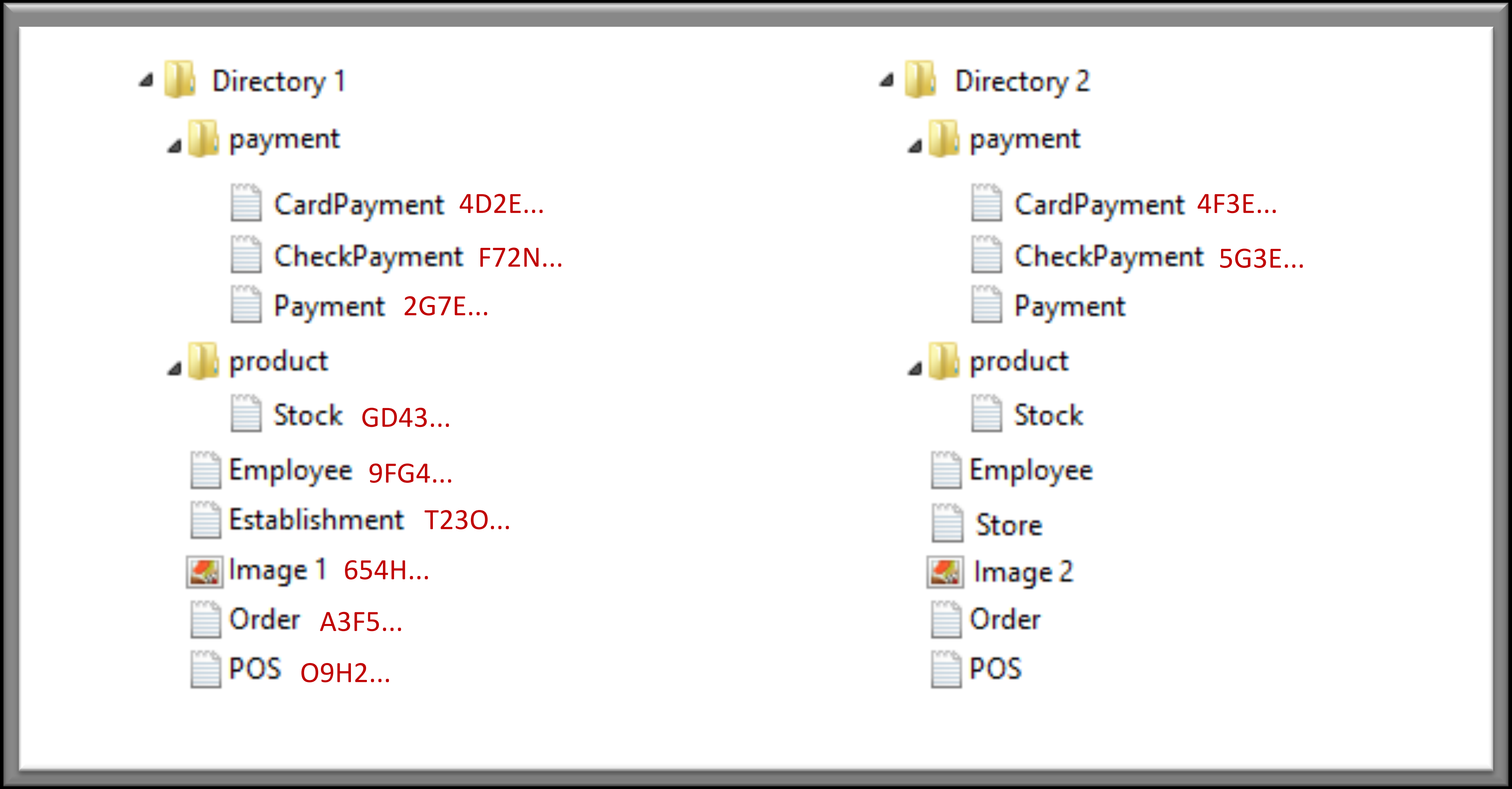


- Directory 1
  - payment
    - CardPayment 4D2E...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CheckPayment
    - Payment
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

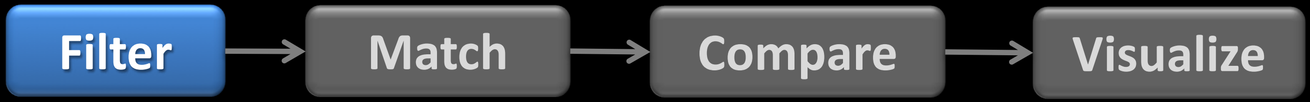


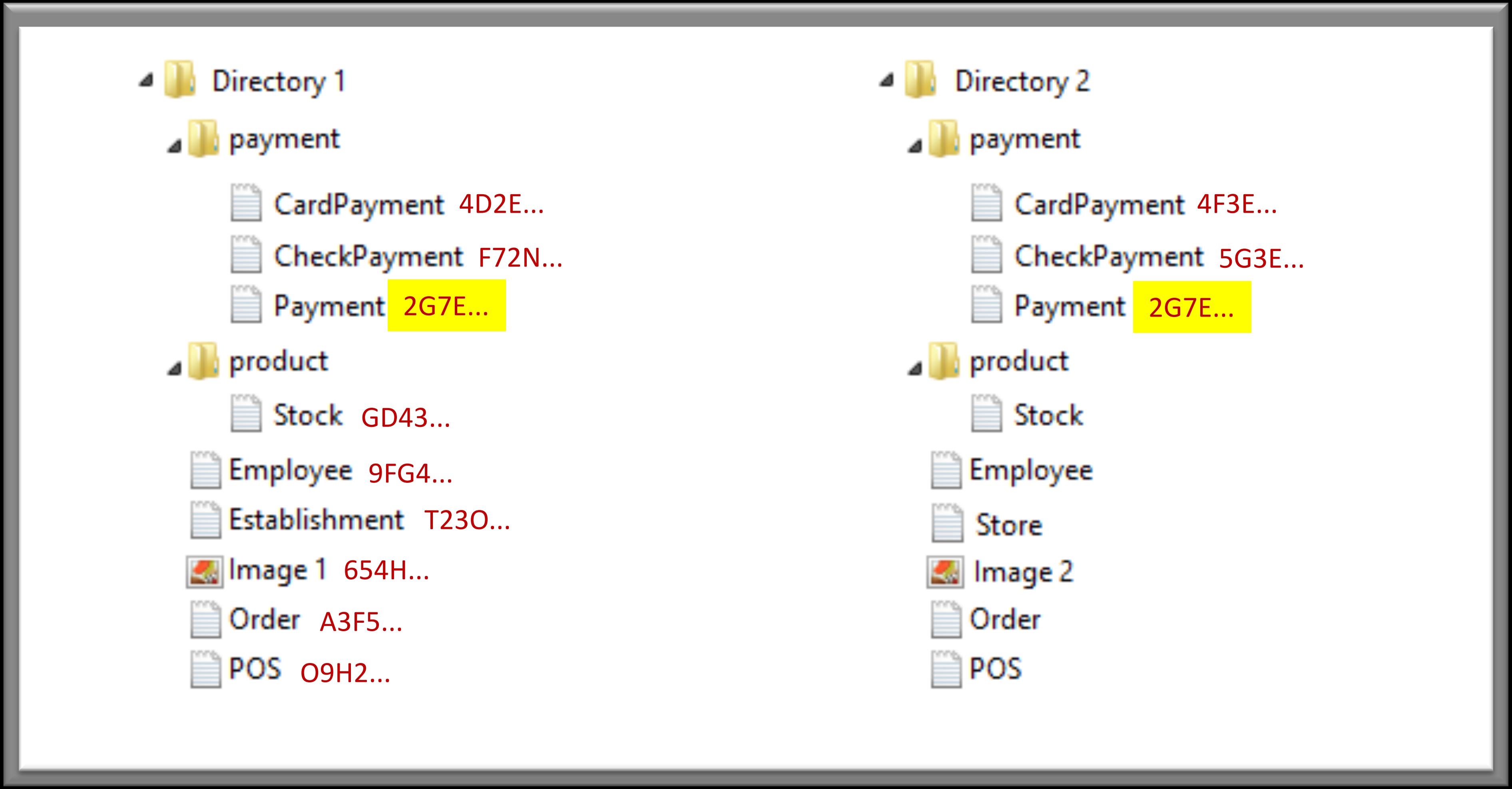
Filter → Match → Compare → Visualize



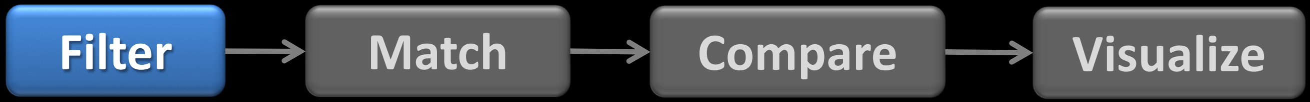
- Directory 1
  - payment
    - CardPayment 4D2E...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CheckPayment 5G3E...
    - Payment
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

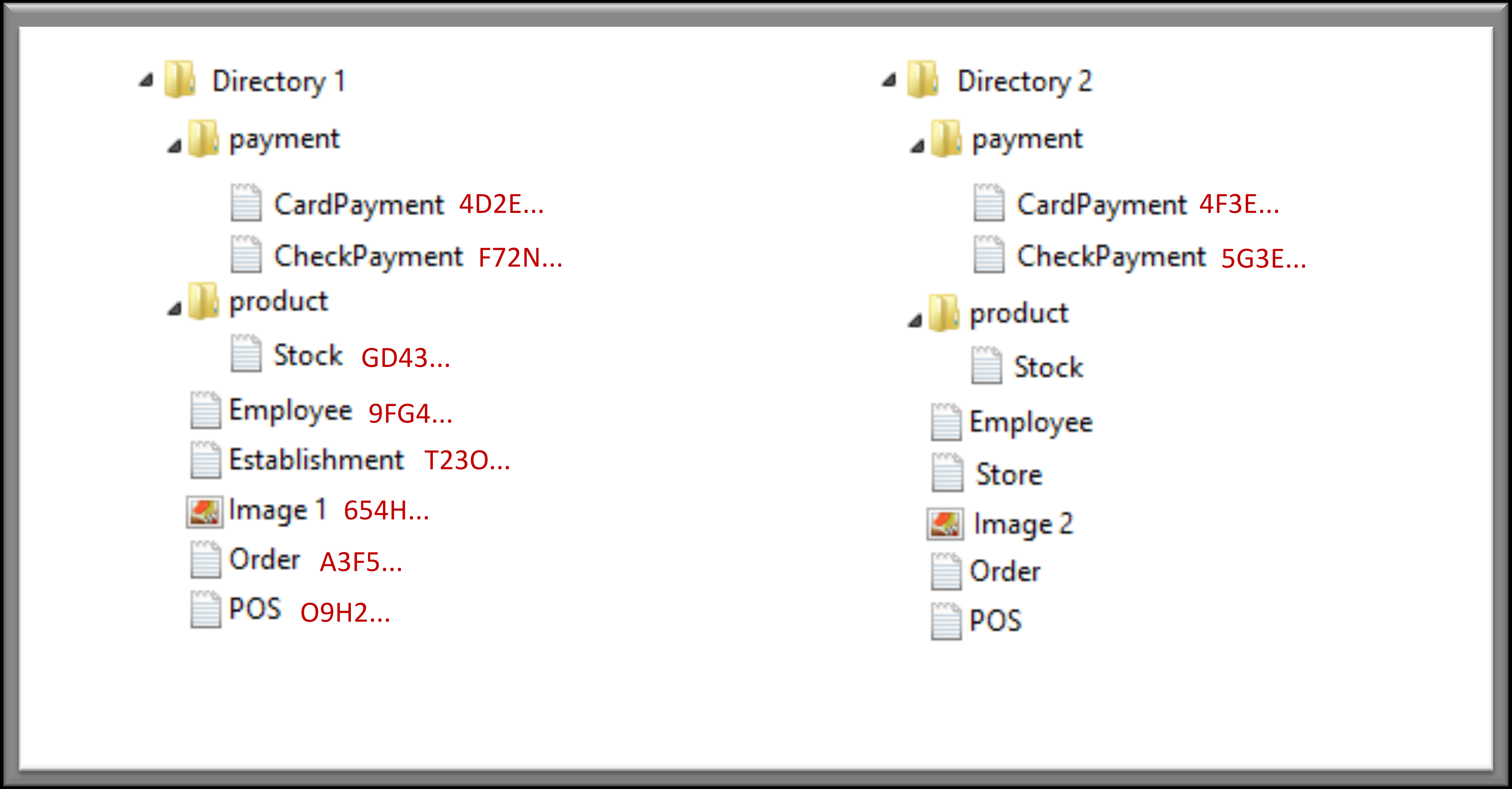

Filter → Match → Compare → Visualize


- Directory 1
  - payment
    - CardPayment 4D2E...
    - CheckPayment F72N...
    - Payment 2G7E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CheckPayment 5G3E...
    - Payment 2G7E...
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

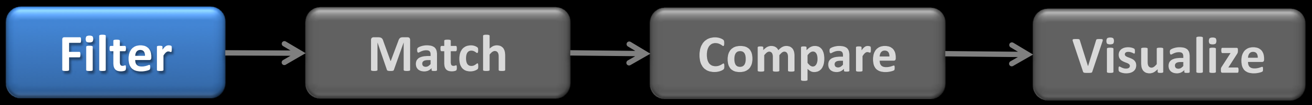


Filter → Match → Compare → Visualize

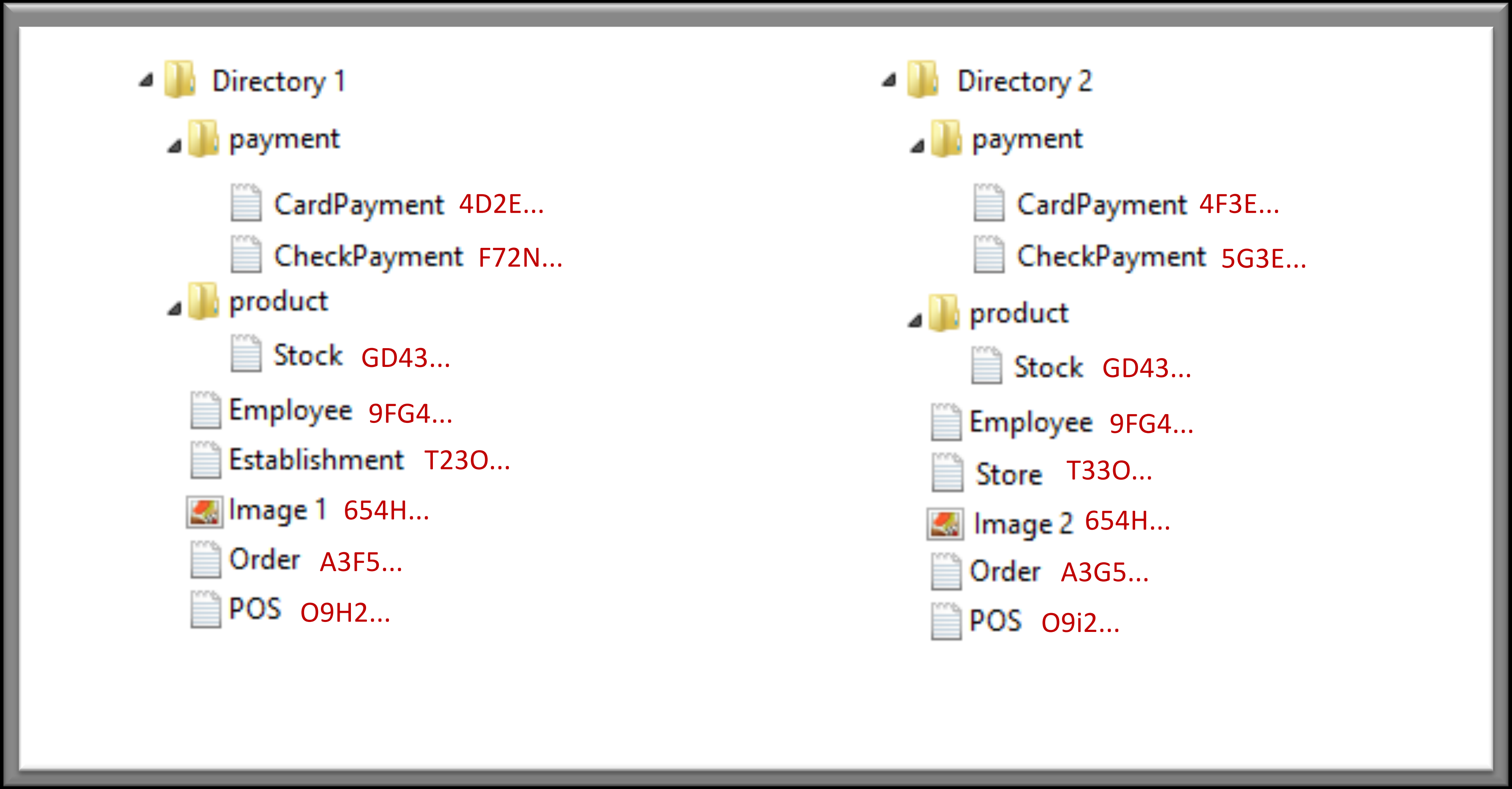


- Directory 1
  - payment
    - CardPayment 4D2E...
    - CheckPayment F72N...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CheckPayment 5G3E...
  - product
    - Stock
  - Employee
  - Store
  - Image 2
  - Order
  - POS

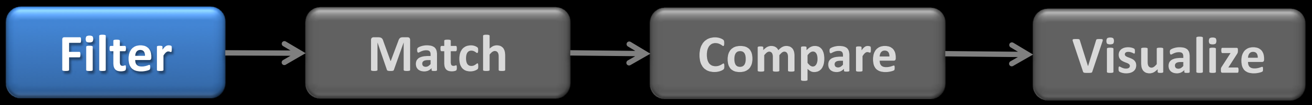


Filter → Match → Compare → Visualize

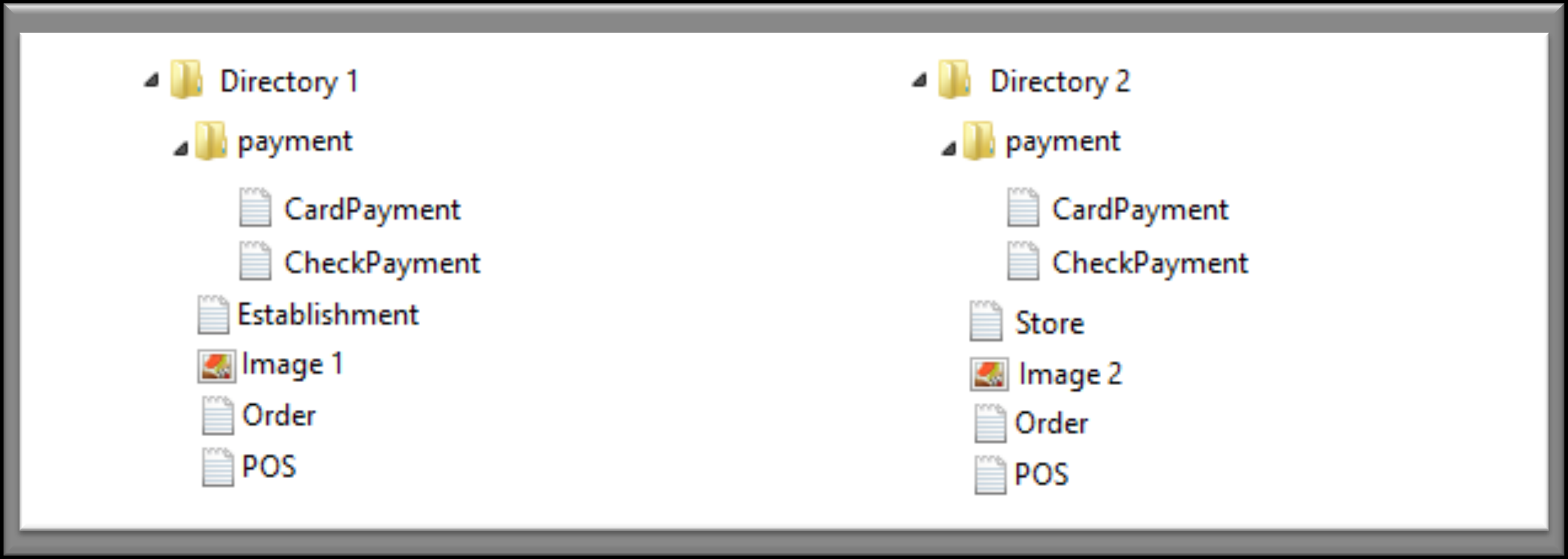


- Directory 1
  - payment
    - CardPayment 4D2E...
    - CheckPayment F72N...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Establishment T23O...
  - Image 1 654H...
  - Order A3F5...
  - POS O9H2...

- Directory 2
  - payment
    - CardPayment 4F3E...
    - CheckPayment 5G3E...
  - product
    - Stock GD43...
  - Employee 9FG4...
  - Store T33O...
  - Image 2 654H...
  - Order A3G5...
  - POS O9i2...



Filter → Match → Compare → Visualize

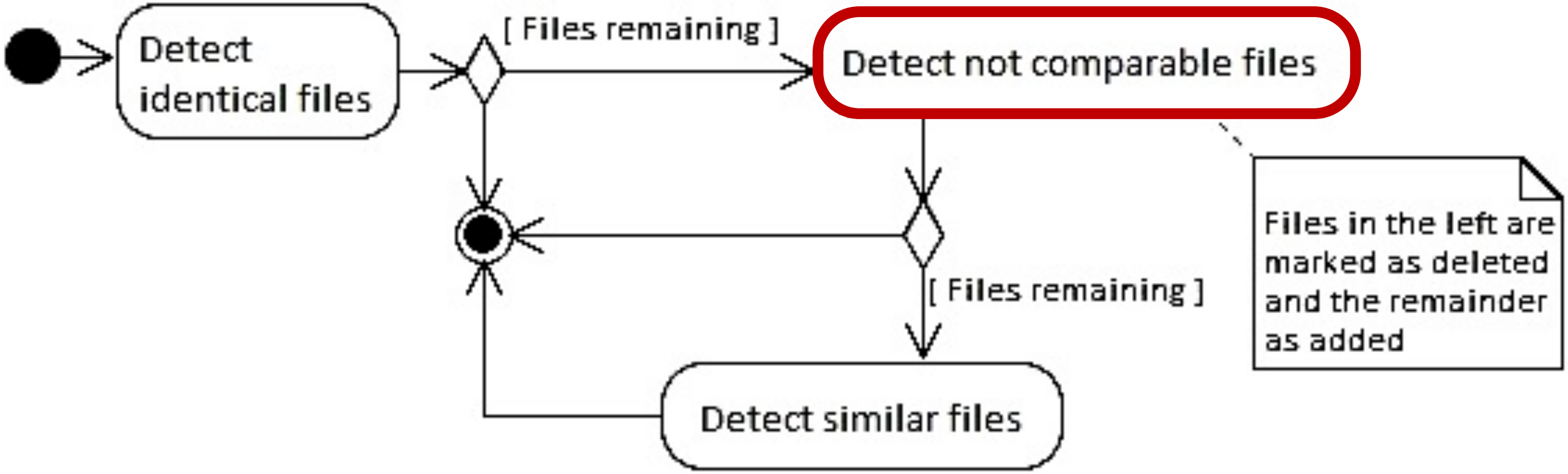


```
Directory 1
    payment
        CardPayment
        CheckPayment
    Establishment
    Image 1
    Order
    POS
```

```
Directory 2
    payment
        CardPayment
        CheckPayment
    Store
    Image 2
    Order
    POS
```

Filter → Match → Compare → Visualize

DDIFF



Detect identical files

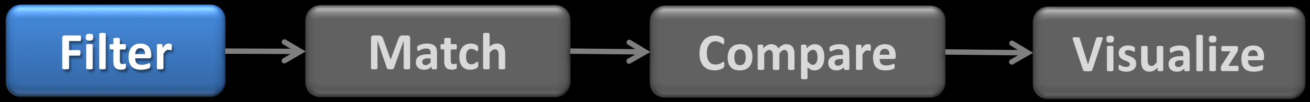
[Files remaining]

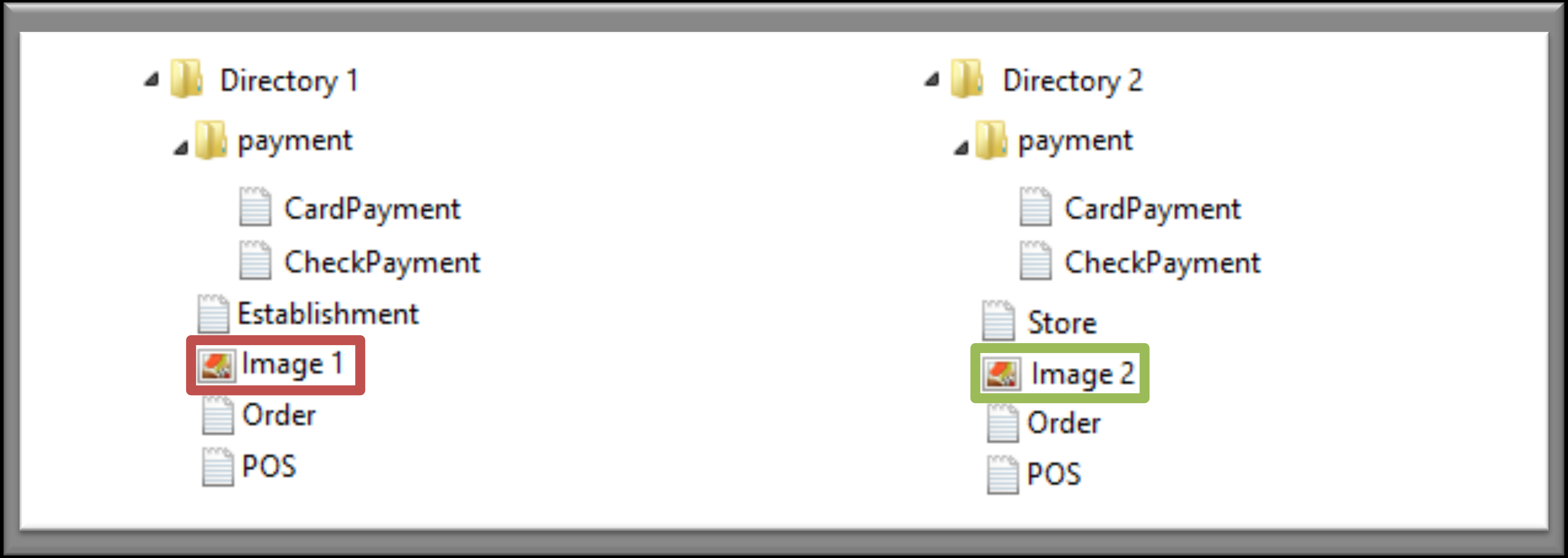
Detect not comparable files

Files in the left are marked as deleted and the remainder as added

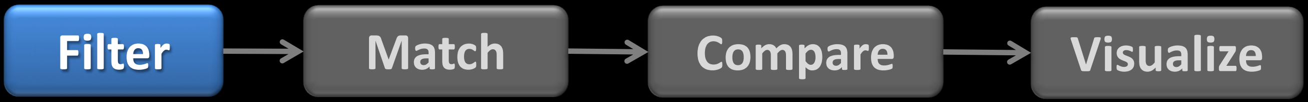
[Files remaining]

Detect similar files

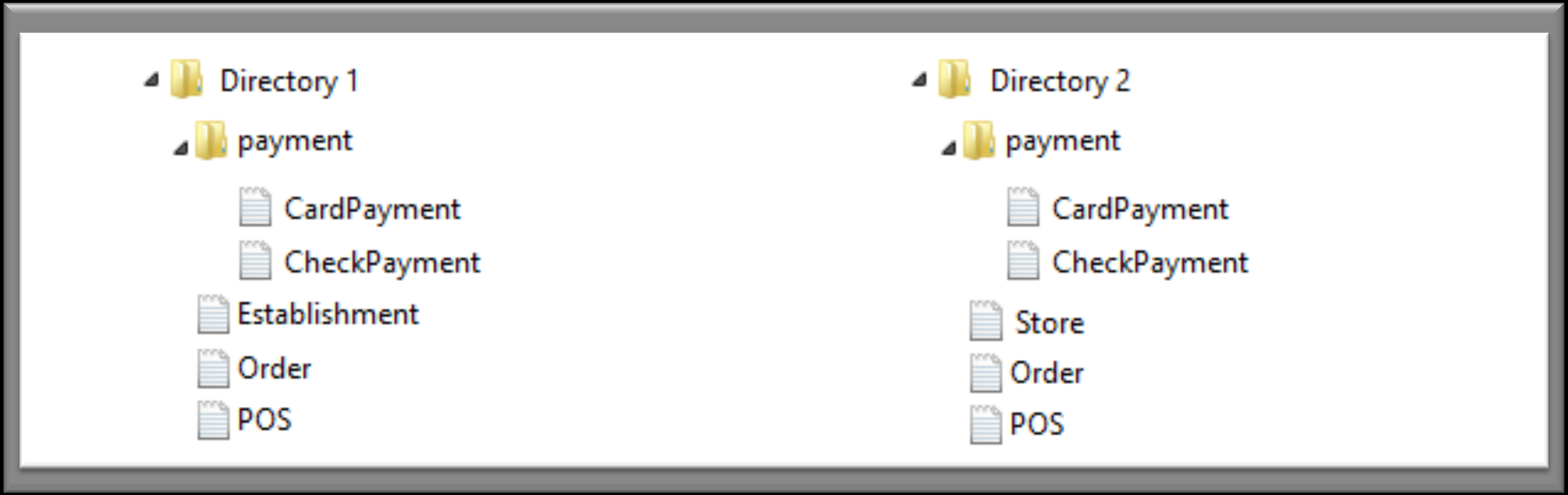

Filter → Match → Compare → Visualize


Directory 1
- payment
  - CardPayment
  - CheckPayment
- Establishment
- Image 1
- Order
- POS

Directory 2
- payment
  - CardPayment
  - CheckPayment
- Store
- Image 2
- Order
- POS



Filter → Match → Compare → Visualize

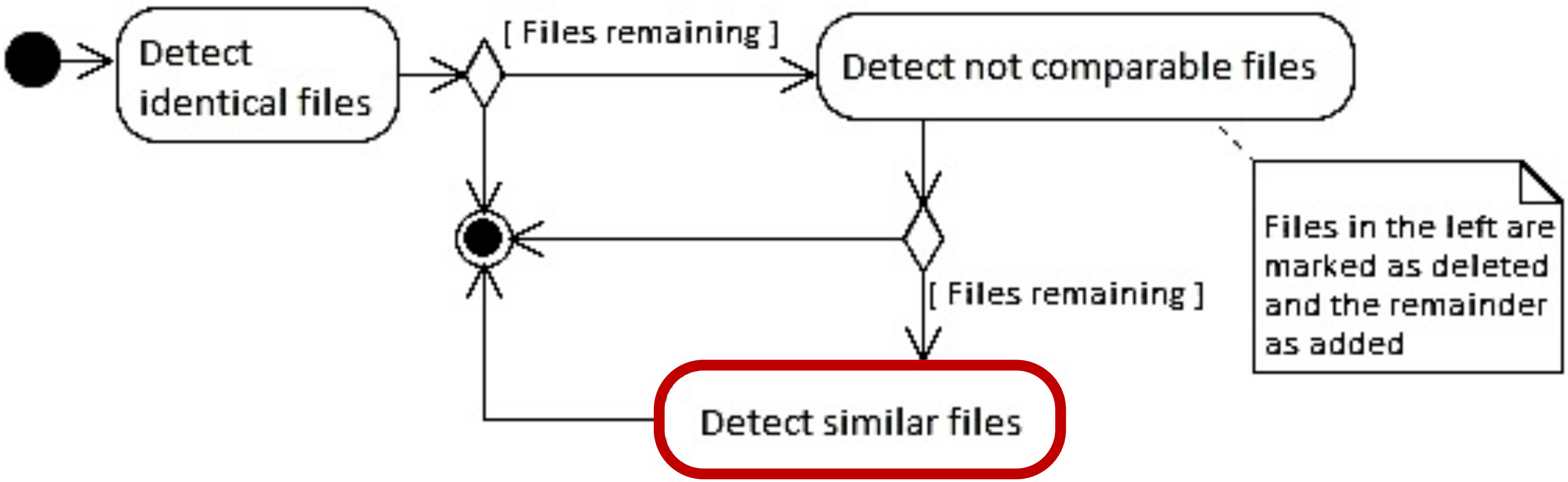


- Directory 1
  - payment
    - CardPayment
    - CheckPayment
  - Establishment
  - Order
  - POS

- Directory 2
  - payment
    - CardPayment
    - CheckPayment
  - Store
  - Order
  - POS

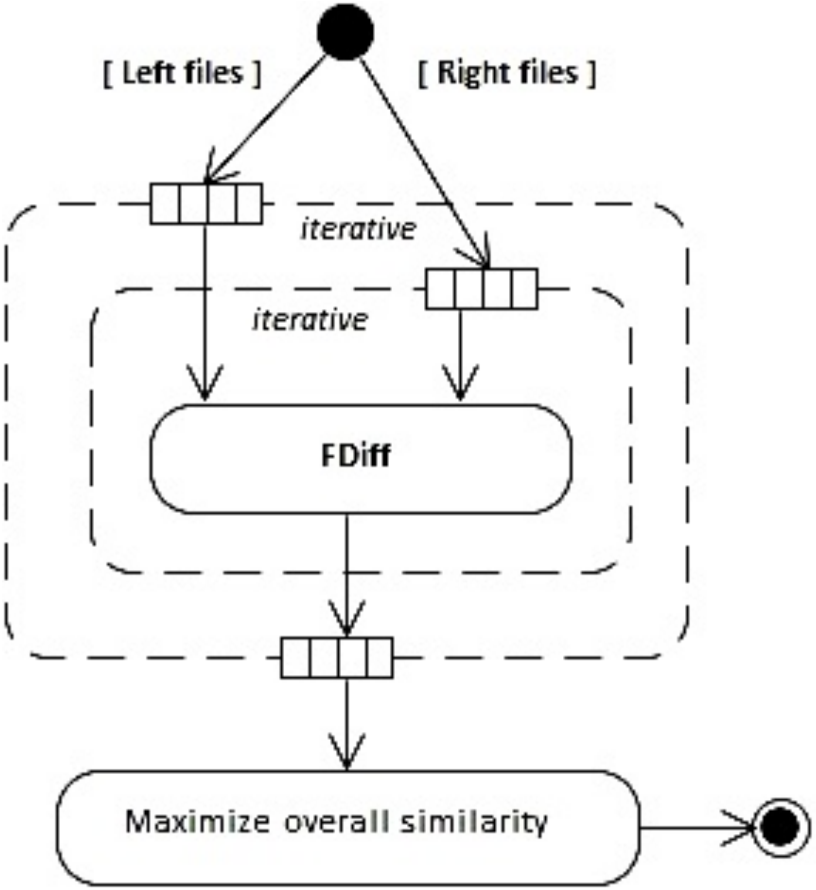
Filter → Match → Compare → Visualize

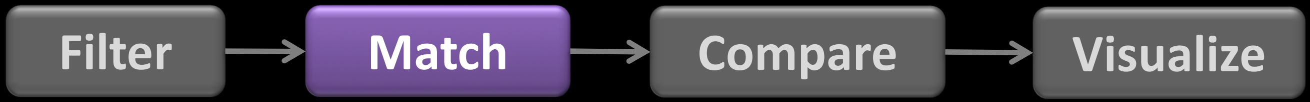
DDIFF


Detect identical files
[Files remaining]
Detect not comparable files
[Files remaining]
Detect similar files
Files in the left are marked as deleted and the remainder as added

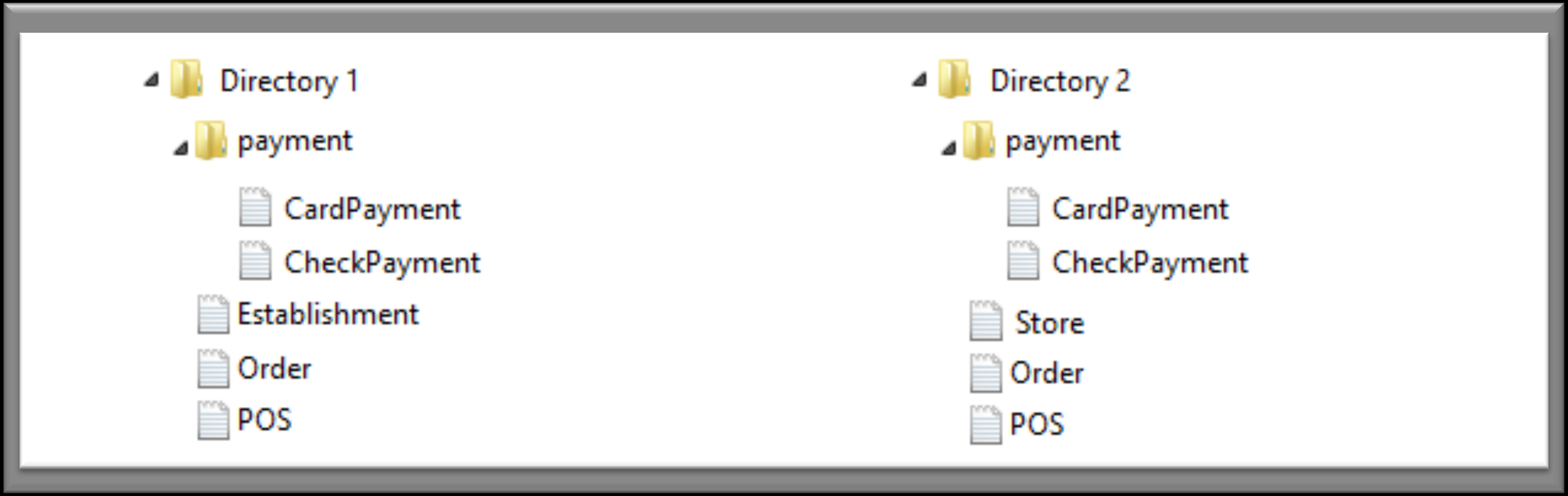
Filter → Match → Compare → Visualize

DDIFF – Detect similar files



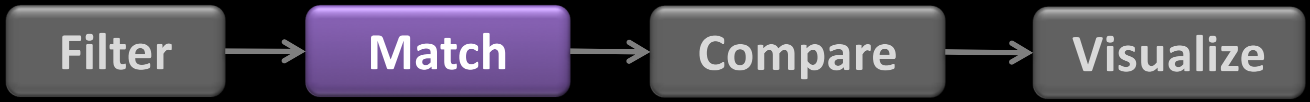


Filter → Match → Compare → Visualize

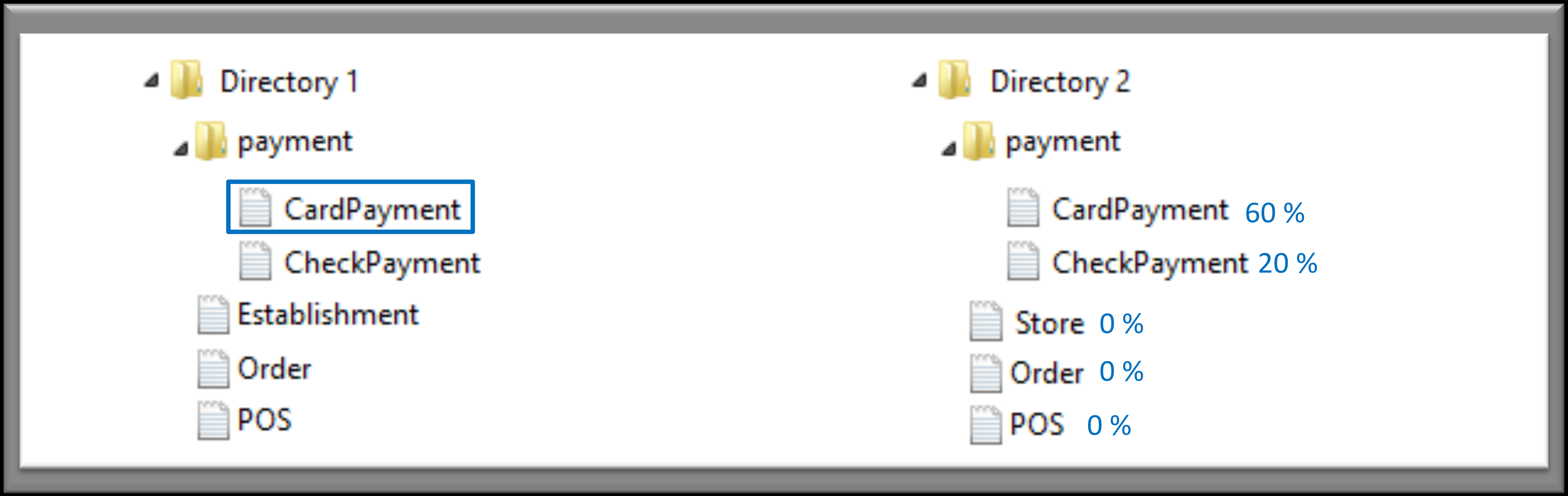


```
Directory 1
  payment
    CardPayment
    CheckPayment
  Establishment
  Order
  POS
```

```
Directory 2
  payment
    CardPayment
    CheckPayment
  Store
  Order
  POS
```



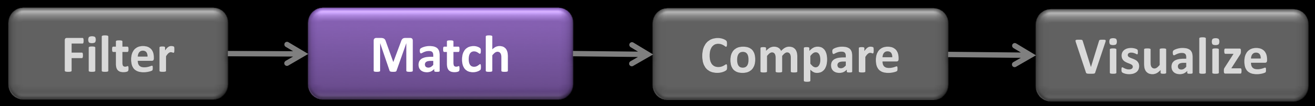
Filter → Match → Compare → Visualize

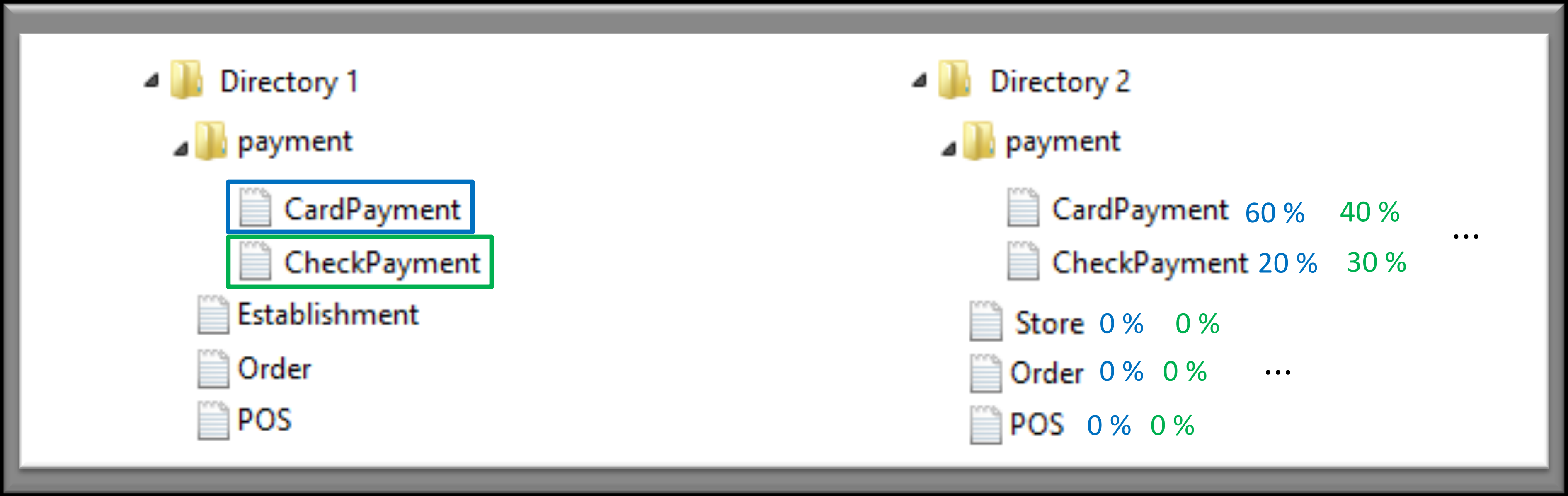


- Directory 1
  - payment
    - CardPayment
    - CheckPayment
  - Establishment
  - Order
  - POS

- Directory 2
  - payment
    - CardPayment 60 %
    - CheckPayment 20 %
  - Store 0 %
  - Order 0 %
  - POS 0 %

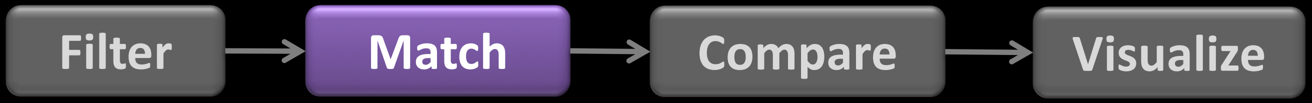
$$Similarity = \frac{2 \times LCS(F_1, F_2)}{Size(F_1) + Size(F_2)}$$

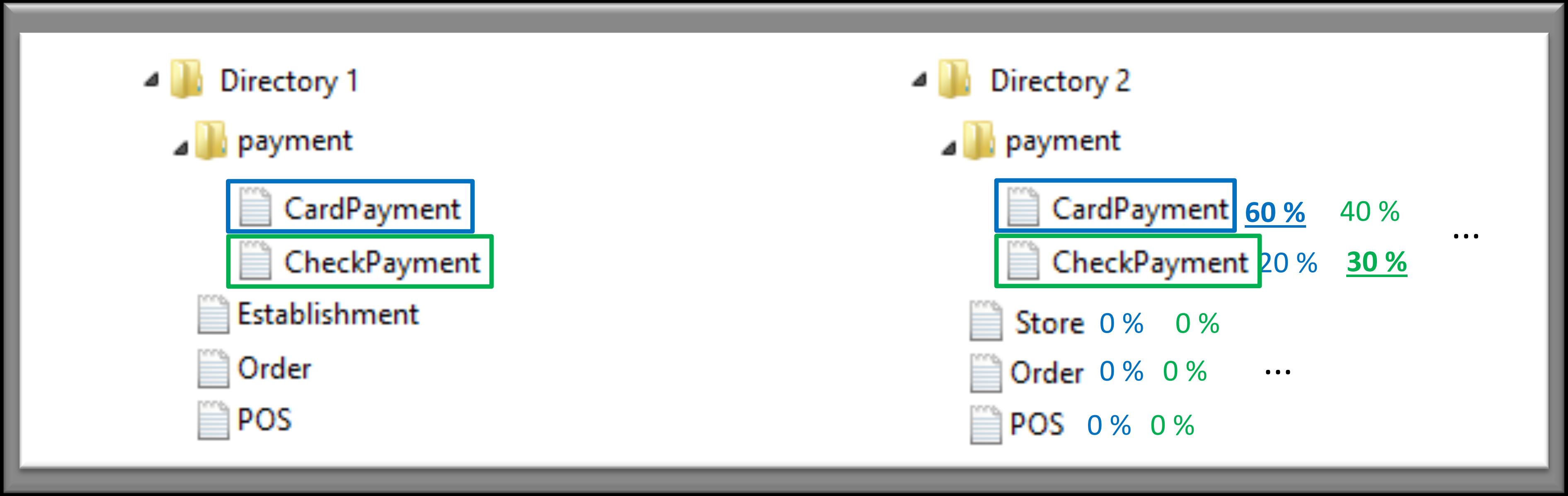

Filter → Match → Compare → Visualize


- Directory 1
  - payment
    - CardPayment
    - CheckPayment
  - Establishment
  - Order
  - POS

- Directory 2
  - payment
    - CardPayment 60 % 40 % …
    - CheckPayment 20 % 30 %
  - Store 0 % 0 %
  - Order 0 % 0 % …
  - POS 0 % 0 %

$$Similarity = \frac{2 \times LCS(F_1, F_2)}{Size(F_1) + Size(F_2)}$$



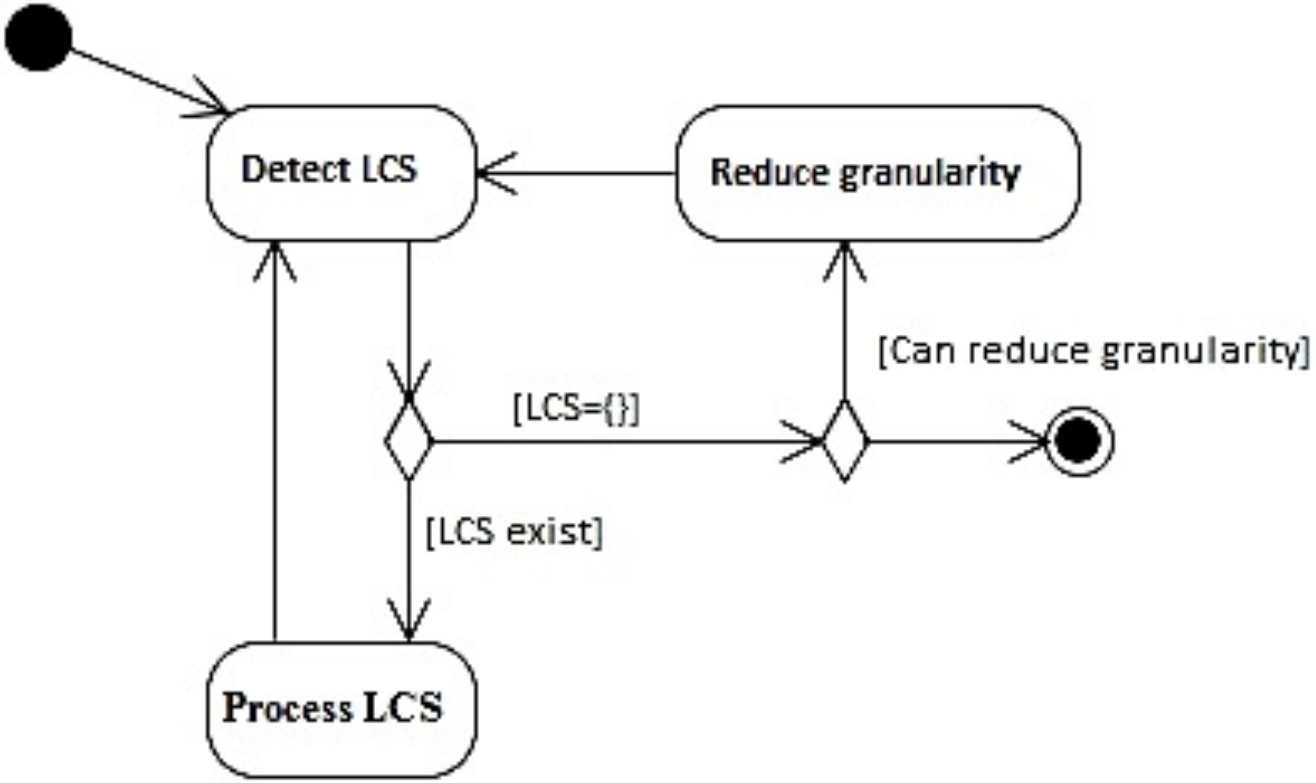


Directory 1
payment
CardPayment
CheckPayment
Establishment
Order
POS

Directory 2
payment
CardPayment 60 % 40 % …
CheckPayment 20 % 30 %
Store 0 % 0 %
Order 0 % 0 % …
POS 0 % 0 %

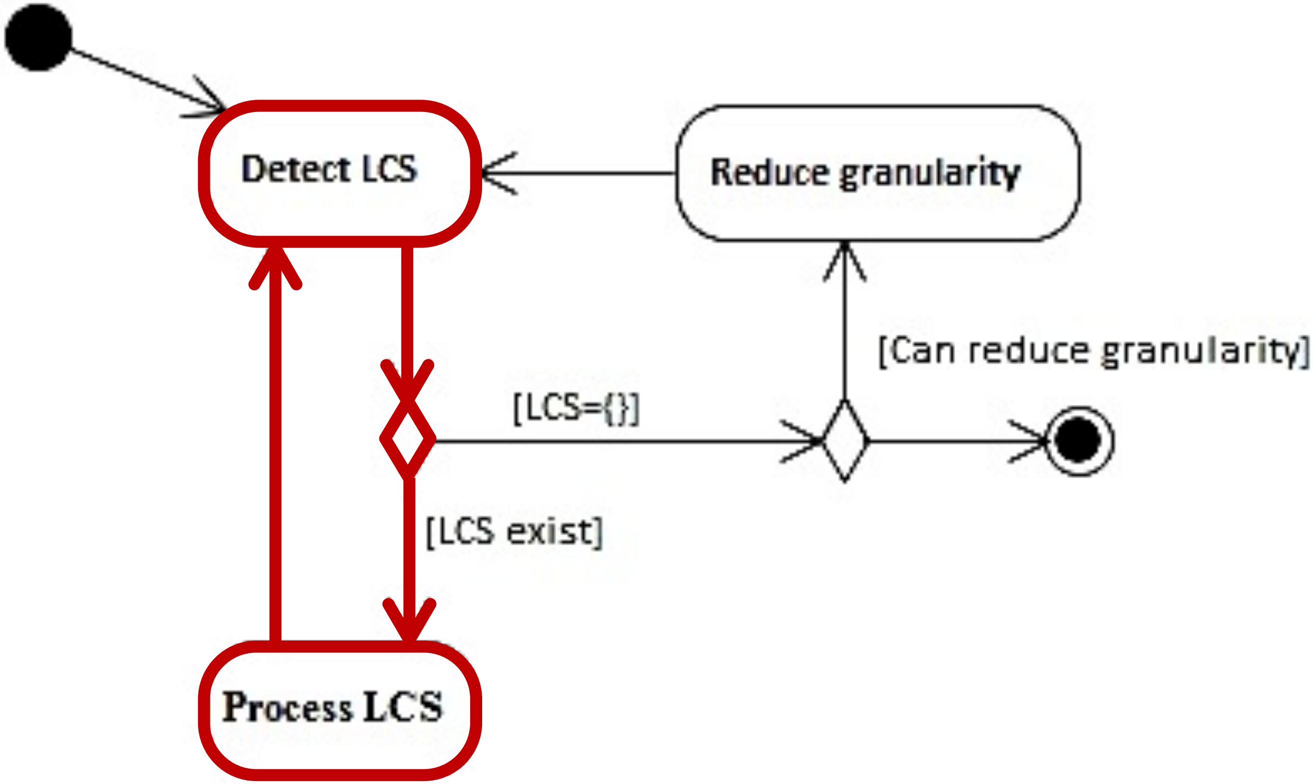
Hungarian Algorithm

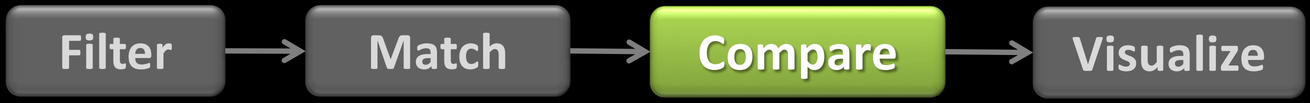
FDIFF


Detect LCS
Reduce granularity
[Can reduce granularity]
[LCS={}]
[LCS exist]
Process LCS

Filter → Match → Compare → Visualize

FDIFF





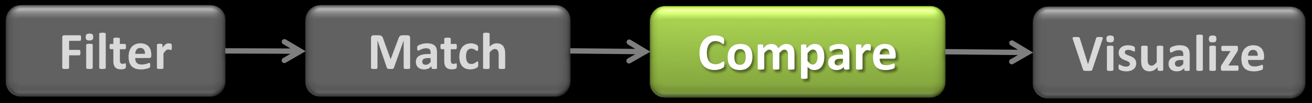
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
import java.util.List;
import java.util.Map;
public class CardPayment extends Payment {
  private Integer installments;
}
```

C:\ Directory 2\payment\CardPayment.java

```
import java.util.Map;
import java.util.List;
public class CardPayment extends Payment {
  public Integer installments;
}
```


Filter → Match → Compare → Visualize

FDIFF

C:\ Directory 1\payment\CardPayment.java

```
import java.util.List;
import java.util.Map;
public class CardPayment extends Payment {
  private Integer installments;
}
```

C:\ Directory 2\payment\CardPayment.java

```
import java.util.Map;
import java.util.List;
public class CardPayment extends Payment {
  public Integer installments;
}
```

Filter → Match → Compare → Visualize

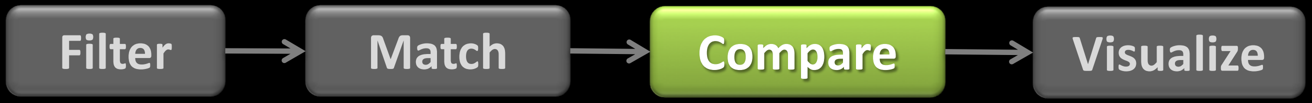
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
import java.util.List;
  private Integer installments;
```

C:\ Directory 2\payment\CardPayment.java

```
import java.util.List;
  public Integer installments;
```


Filter → Match → Compare → Visualize

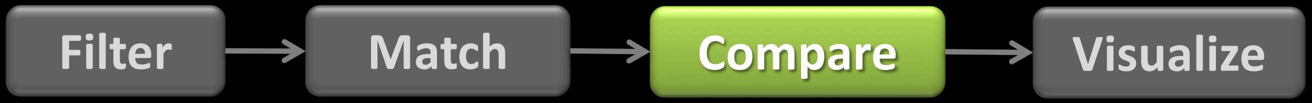
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
import java.util.List;
  private Integer installments;
```

C:\ Directory 2\payment\CardPayment.java

```
import java.util.List;
  public Integer installments;
```



Filter → Match → Compare → Visualize

FDIFF

C:\ Directory 1\payment\CardPayment.java

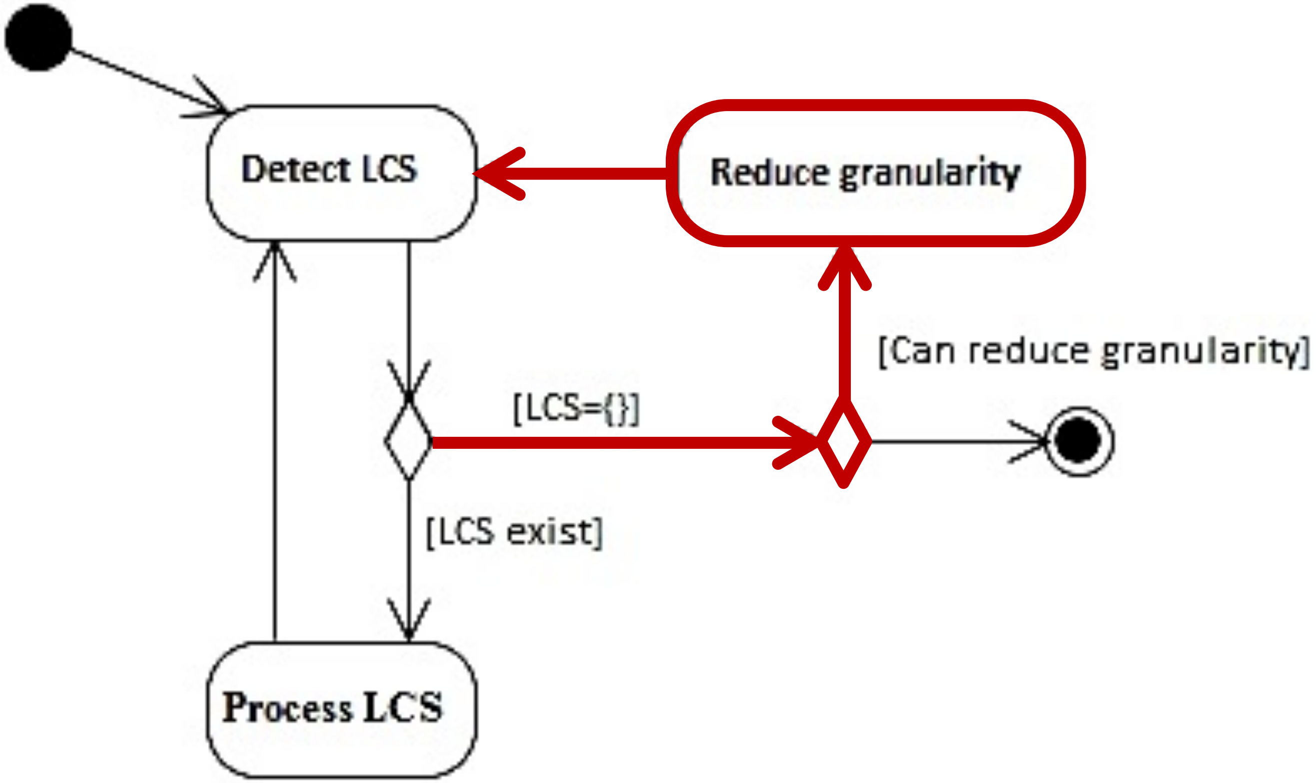
```
private Integer installments;
```

C:\ Directory 2\payment\CardPayment.java

```
public Integer installments;
```

Filter → Match → Compare → Visualize

FDIFF



Detect LCS

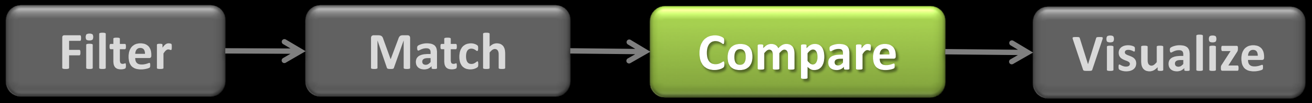
Reduce granularity

[Can reduce granularity]

[LCS={}]

[LCS exist]

Process LCS



Filter → Match → Compare → Visualize

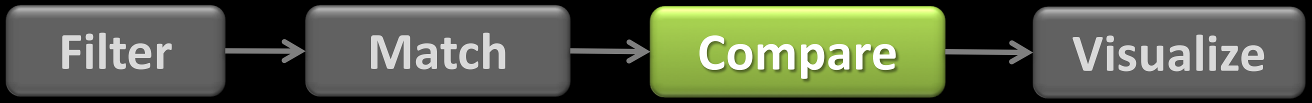
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
private
Integer
installments;
```

C:\ Directory 2\payment\CardPayment.java

```
public
Integer
installments;
```


Filter → Match → Compare → Visualize

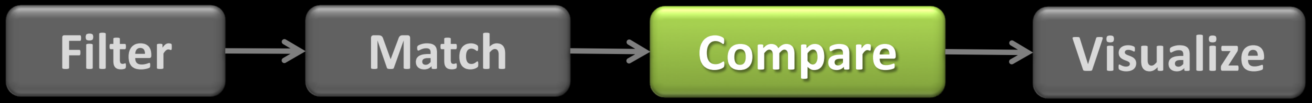
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
private
Integer
installments;
```

C:\ Directory 2\payment\CardPayment.java

```
public
Integer
installments;
```



Filter → Match → Compare → Visualize

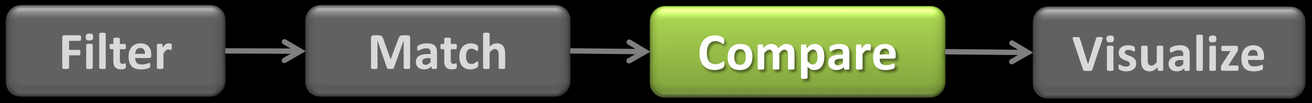
FDIFF

C:\ Directory 1\payment\CardPayment.java

```
private
```

C:\ Directory 2\payment\CardPayment.java

```
public
```



FDIFF

C:\ Directory 1\payment\CardPayment.java

```
import java.util.List;
import java.util.Map;
public class CardPayment extends Payment {
  private Integer installments;
}
```

C:\ Directory 2\payment\CardPayment.java

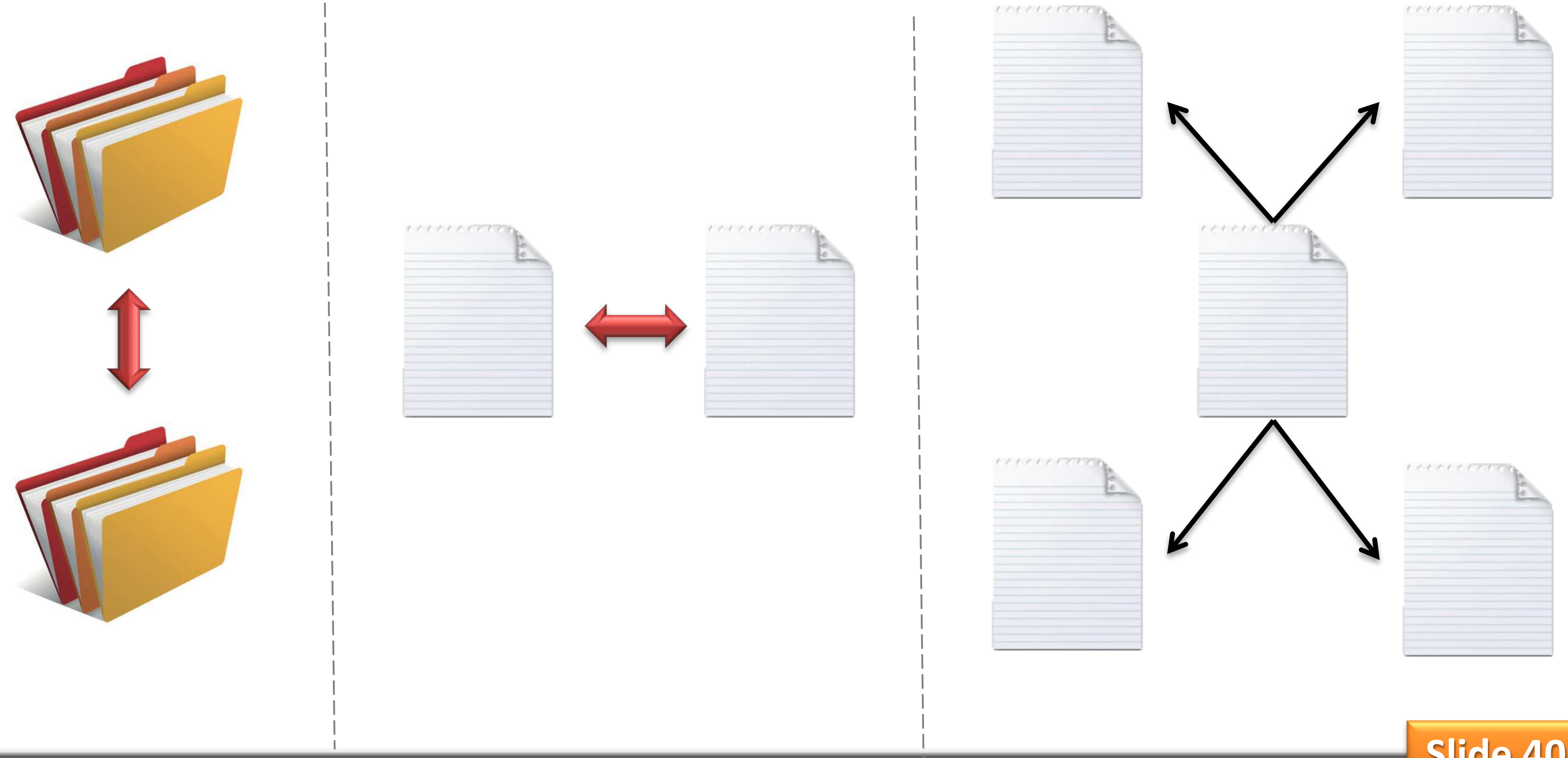
```
import java.util.Map;
import java.util.List;
public class CardPayment extends Payment {
  public Integer installments;
}
```

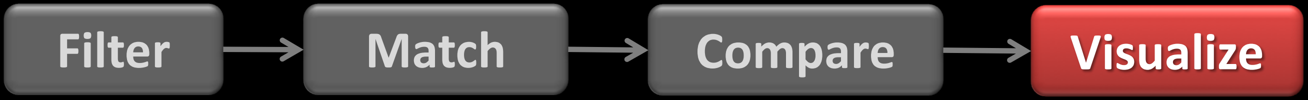
Filter → Match → Compare → **Visualize**

DDIFF

FDIFF

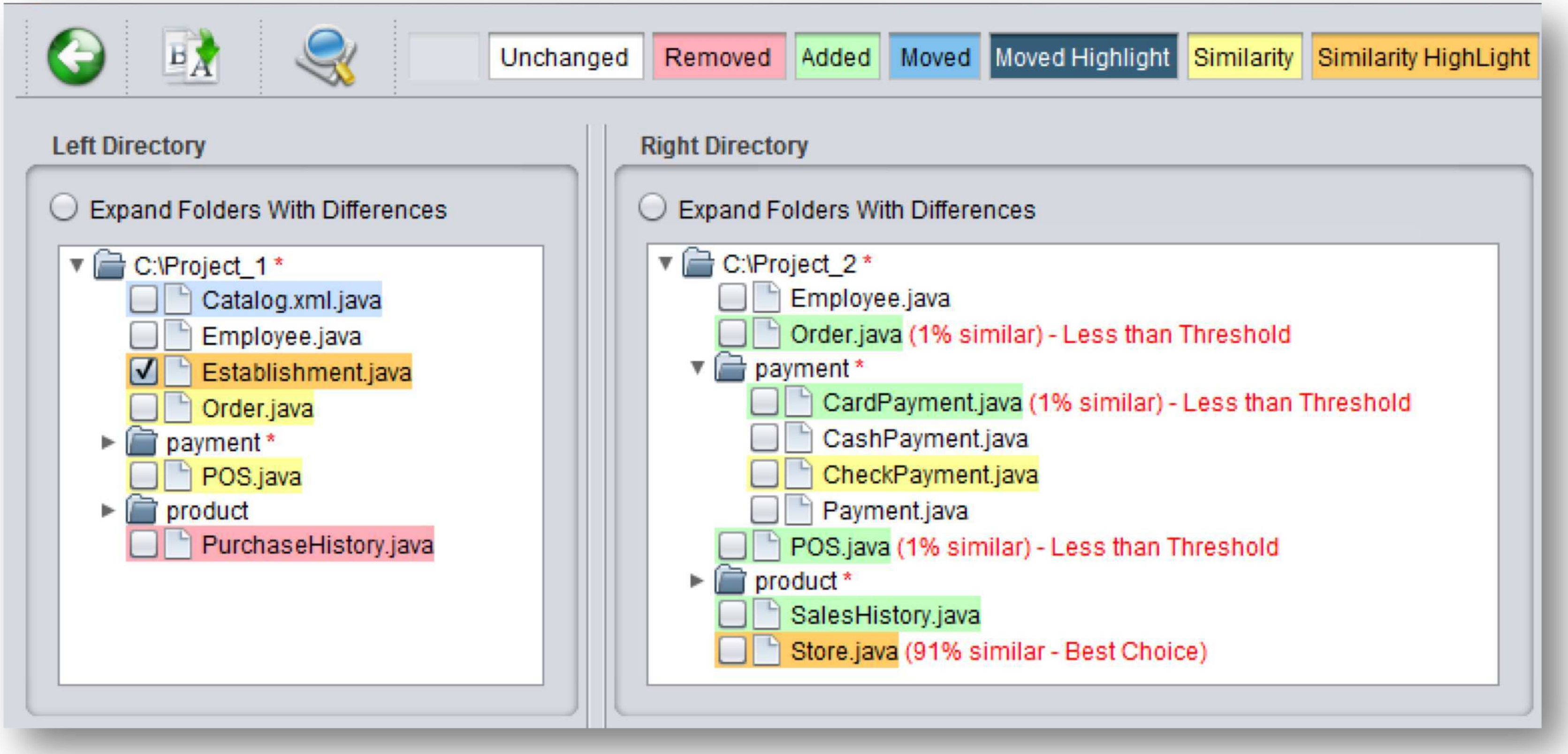
Overview vs. Pairwise Comparison vs. Multiple Comparison





Filter → Match → Compare → Visualize

DDIFF – Overview



Left Directory

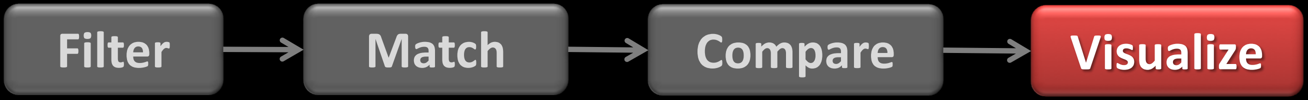
○ Expand Folders With Differences

- ▼ C:\Project_1 *
  - □ Catalog.xml.java
  - □ Employee.java
  - ☑ Establishment.java
  - □ Order.java
  - ▶ payment *
  - □ POS.java
  - ▶ product
  - □ PurchaseHistory.java

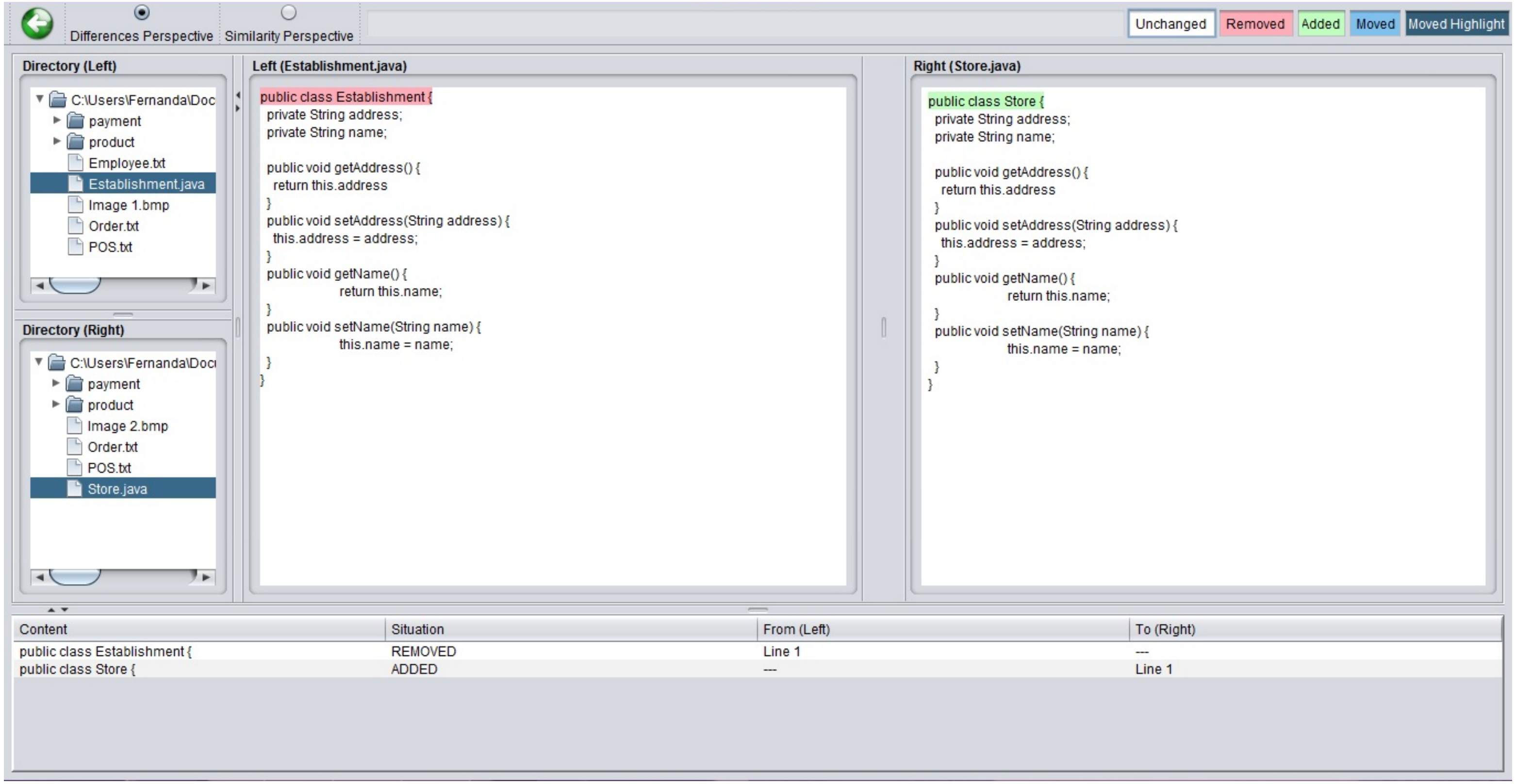
Right Directory

○ Expand Folders With Differences

- ▼ C:\Project_2 *
  - □ Employee.java
  - □ Order.java (1% similar) - Less than Threshold
  - ▼ payment *
    - □ CardPayment.java (1% similar) - Less than Threshold
    - □ CashPayment.java
    - □ CheckPayment.java
    - □ Payment.java
  - □ POS.java (1% similar) - Less than Threshold
  - ▶ product *
  - □ SalesHistory.java
  - □ Store.java (91% similar - Best Choice)



FDIFF – Pairwise Comparison, Differences Perspective (comparing the same file)



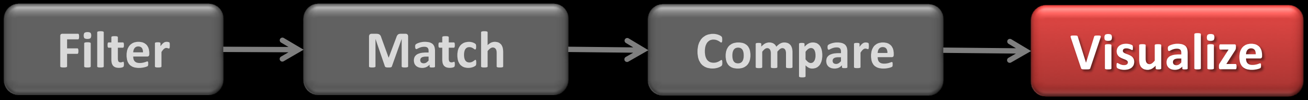
Left (Establishment.java)

```
public class Establishment {
 private String address;
 private String name;

 public void getAddress() {
  return this.address
 }
 public void setAddress(String address) {
  this.address = address;
 }
 public void getName() {
            return this.name;
 }
 public void setName(String name) {
            this.name = name;
 }
}
```

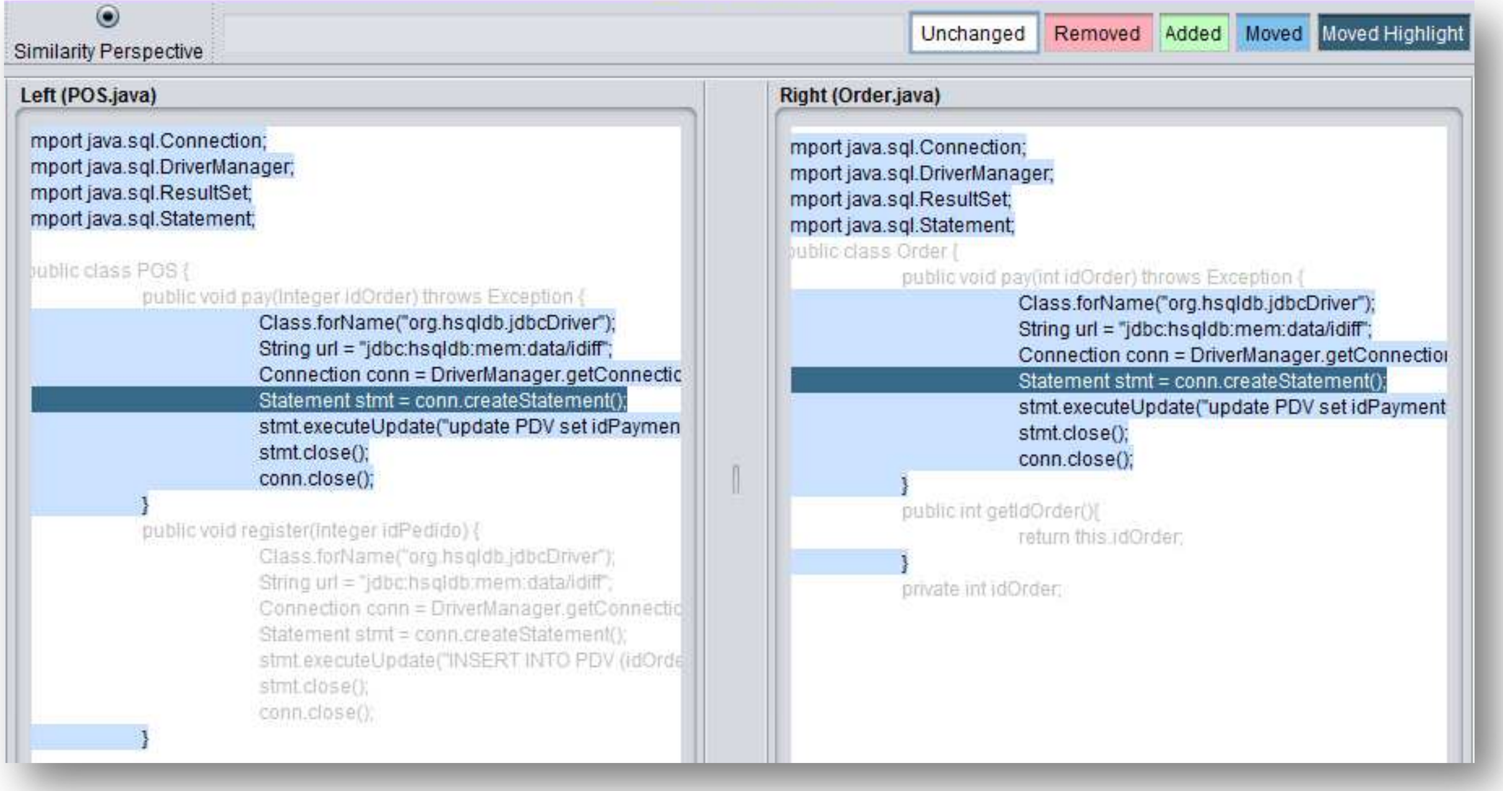
Right (Store.java)

```
public class Store {
 private String address;
 private String name;

 public void getAddress() {
  return this.address
 }
 public void setAddress(String address) {
  this.address = address;
 }
 public void getName() {
            return this.name;
 }
 public void setName(String name) {
            this.name = name;
 }
}
```

Content	Situation	From (Left)	To (Right)
public class Establishment {	REMOVED	Line 1	---
public class Store {	ADDED	---	Line 1



FDIFF – Pairwise Comparison, Similarity perspective (comparing different files)

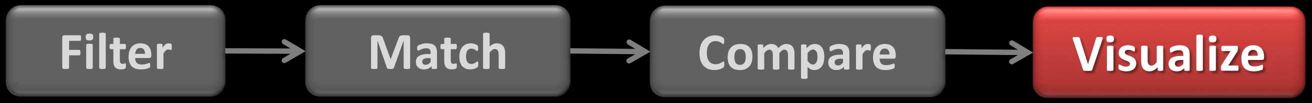


Left (POS.java)

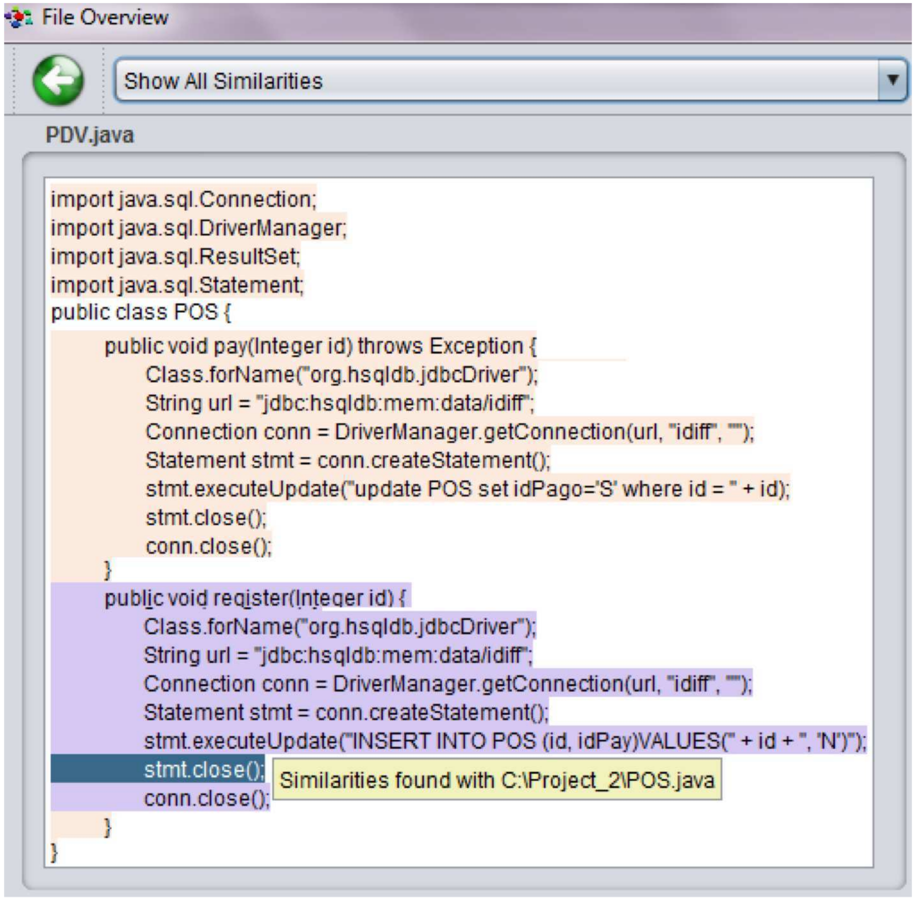
```
mport java.sql.Connection;
mport java.sql.DriverManager;
mport java.sql.ResultSet;
mport java.sql.Statement;

ublic class POS {
        public void pay(Integer idOrder) throws Exception {
                Class.forName("org.hsqldb.jdbcDriver");
                String url = "jdbc:hsqldb:mem:data/idiff";
                Connection conn = DriverManager.getConnectic
                Statement stmt = conn.createStatement();
                stmt.executeUpdate("update PDV set idPaymen
                stmt.close();
                conn.close();
        }
        public void register(Integer idPedido) {
                Class.forName("org.hsqldb.jdbcDriver");
                String url = "jdbc:hsqldb:mem:data/idiff";
                Connection conn = DriverManager.getConnectic
                Statement stmt = conn.createStatement();
                stmt.executeUpdate("INSERT INTO PDV (idOrde
                stmt.close();
                conn.close();
        }
```

Right (Order.java)

```
mport java.sql.Connection;
mport java.sql.DriverManager;
mport java.sql.ResultSet;
mport java.sql.Statement;
ublic class Order {
        public void pay(int idOrder) throws Exception {
                Class.forName("org.hsqldb.jdbcDriver");
                String url = "jdbc:hsqldb:mem:data/idiff";
                Connection conn = DriverManager.getConnectio
                Statement stmt = conn.createStatement();
                stmt.executeUpdate("update PDV set idPayment
                stmt.close();
                conn.close();
        }
        public int getIdOrder(){
                return this.idOrder;
        }
        private int idOrder;
```


Filter → Match → Compare → Visualize

FDIFF – Multiple Comparison



Show All Similarities

PDV.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
public class POS {
        public void pay(Integer id) throws Exception {
                Class.forName("org.hsqldb.jdbcDriver");
                String url = "jdbc:hsqldb:mem:data/idiff";
                Connection conn = DriverManager.getConnection(url, "idiff", "");
                Statement stmt = conn.createStatement();
                stmt.executeUpdate("update POS set idPago='S' where id = " + id);
                stmt.close();
                conn.close();
        }
        public void register(Integer id) {
                Class.forName("org.hsqldb.jdbcDriver");
                String url = "jdbc:hsqldb:mem:data/idiff";
                Connection conn = DriverManager.getConnection(url, "idiff", "");
                Statement stmt = conn.createStatement();
                stmt.executeUpdate("INSERT INTO POS (id, idPay)VALUES(" + id + ", 'N')");
                stmt.close();
                conn.close();
        }
}
```

Similarities found with C:\Project_2\POS.java

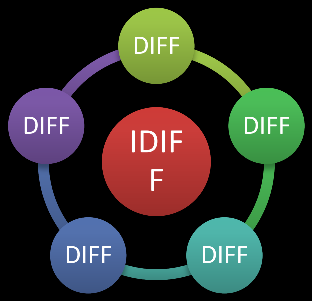
INTRODUCTION

MOTIVATING EXAMPLE

ITERATIVE DIFF (IDIFF)

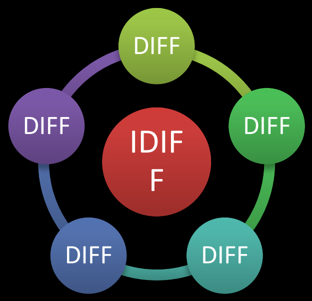
EVALUATION

CONCLUSIONS



PLANNING AND EXECUTION

- Research questions:
  - Which is **the best granularity** configuration for IDiff?
  - Does IDiff **increase the precision** (correctness) when compared to a generic Diff tool?
  - Does IDiff **increase the recall** (completeness) when compared to a generic Diff tool?
  - In which situations (refactoring types) **IDiff performs better than a generic Diff tool?**

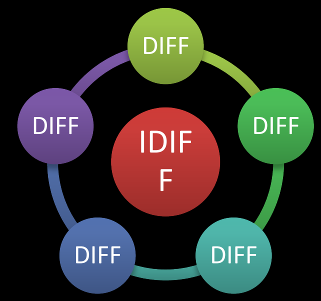


PLANNING AND EXECUTION

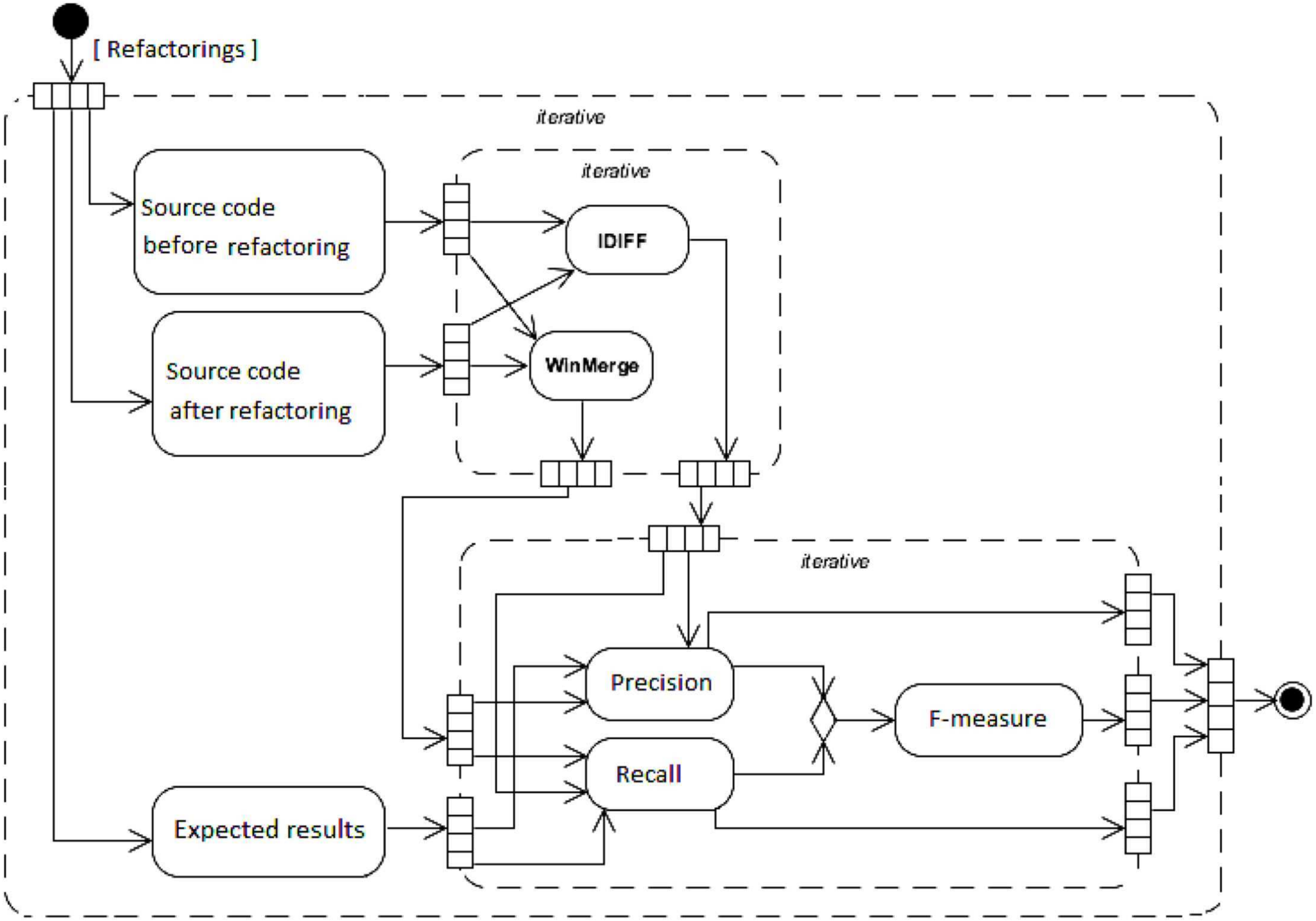
- Execution of **76 refactorings** from the **Fowler's book**

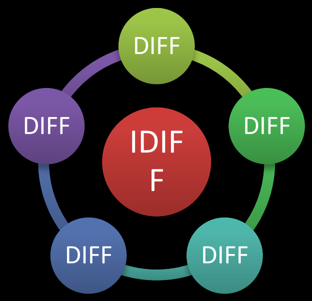
- **Comparison** of the expected **results** with the results provided by IDiff and WinMerge

  - **WinMerge** selected as baseline out of a survey with 63 developers



EVALUATION PROCESS



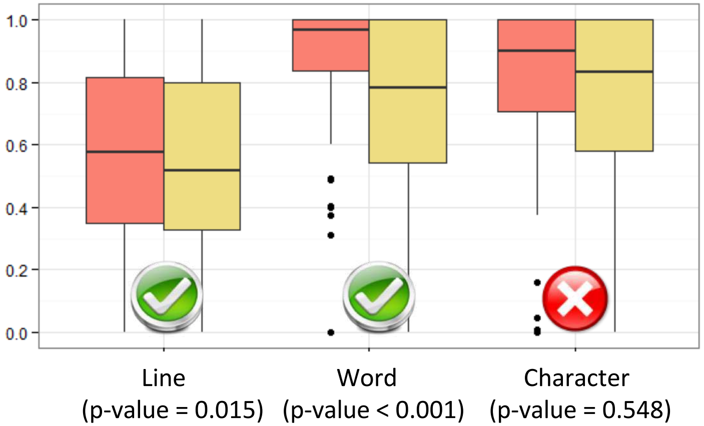


PRECISION/RECALL ANALYSIS

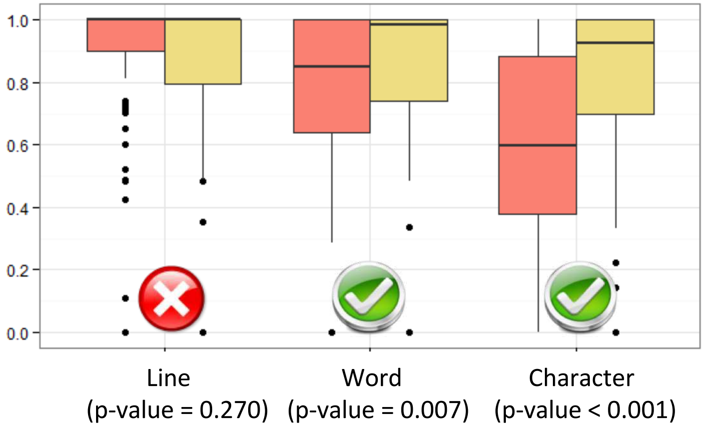
COMPARISON OF MEAN – WILCOXON TEST

🟥 IDIFF
🟨 WinMerge

Precision

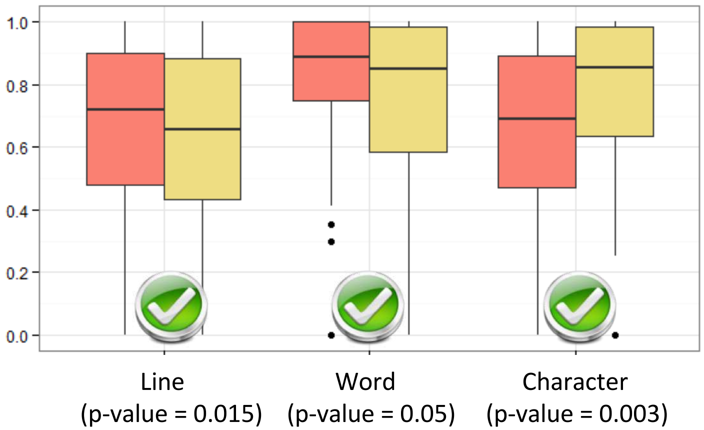


Recall



Summary distributions for IDIFF and WinMerge

F-measure



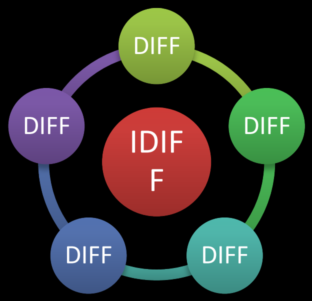
INTRODUCTION

MOTIVATING EXAMPLE

ITERATIVE DIFF (IDIFF)

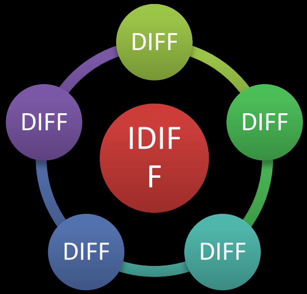
EVALUATION

CONCLUSIONS



MAIN THREATS TO VALIDITY

- Reliability of measurements
- The use of 76 refactorings described in the Fowler's book
- The absence of experience with large projects leaves doubt whether the result will be satisfactory in these scenarios
- WinMerge as baseline



CONTRIBUTIONS

- IDiff provides **results with higher precision** if compared to a generic Diff tool, without drastic reduction of recall

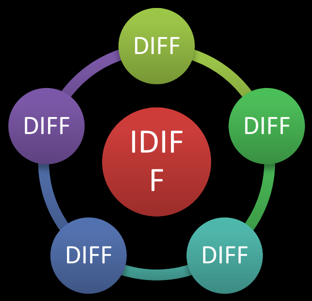
- IDiff employs **efficient algorithms** for detecting the optimal content-based similarity amongst files

- **Different visualizations** (pairwise and multiple) and according to **different perspectives** (similarities and differences)

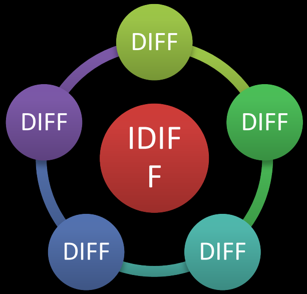
- **The use of iterative granularity reduction to conciliate precision and efficiency**

FUTURE WORK

- Consider programming language **grammars**
- Develop a **merge tool** supported by the foundations of this work
- Exploit **parallel processing** of ever-common multi-core computers and GPU
- **Combine with refactoring detection techniques** by using regular expressions over the diff results to index a refactoring catalog



Universidade Federal Fluminense



The Brazilian Symposium on Software Engineering (SBES)

Instituto de Computação

Towards a Difference Detection Algorithm aware of Refactoring-related Changes

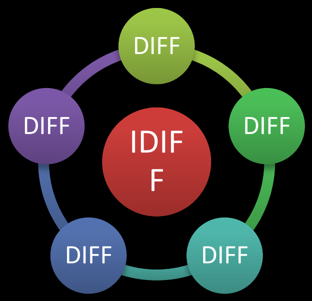
Fernanda Silva[1] Eraldo Borel[1] Evandro Lopes[2] **Leonardo Murta**[1]

[1] Computing Institute
Fluminense Federal University (UFF)
Niterói, Rio de Janeiro, Brazil
e-mail: {ffloriano,leomurta}@ic.uff.br
eraldoborel@id.uff.br

[2] Department of Statistics
Fluminense Federal University (UFF)
Niterói, Rio de Janeiro, Brazil
e-mail: evandro_dalbem@id.uff.br

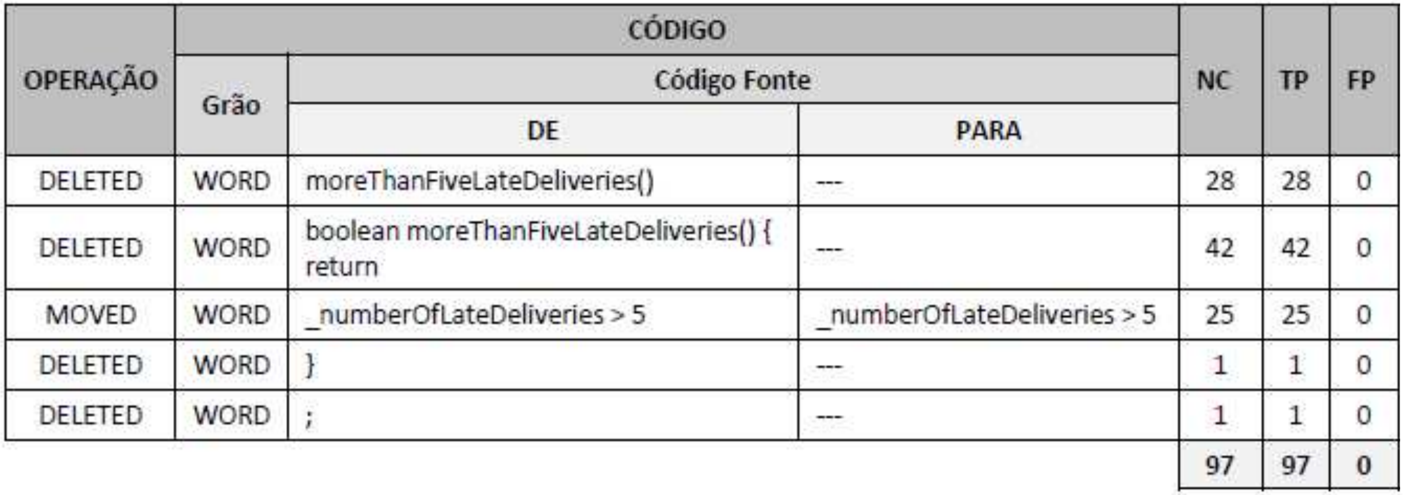
GEMS
Software Maintenance and Evolution Group

CBSoft2014
Congresso Brasileiro de Software: Teoria e Prática
28 de Setembro a 03 de Outubro de 2014
Maceió - Alagoas

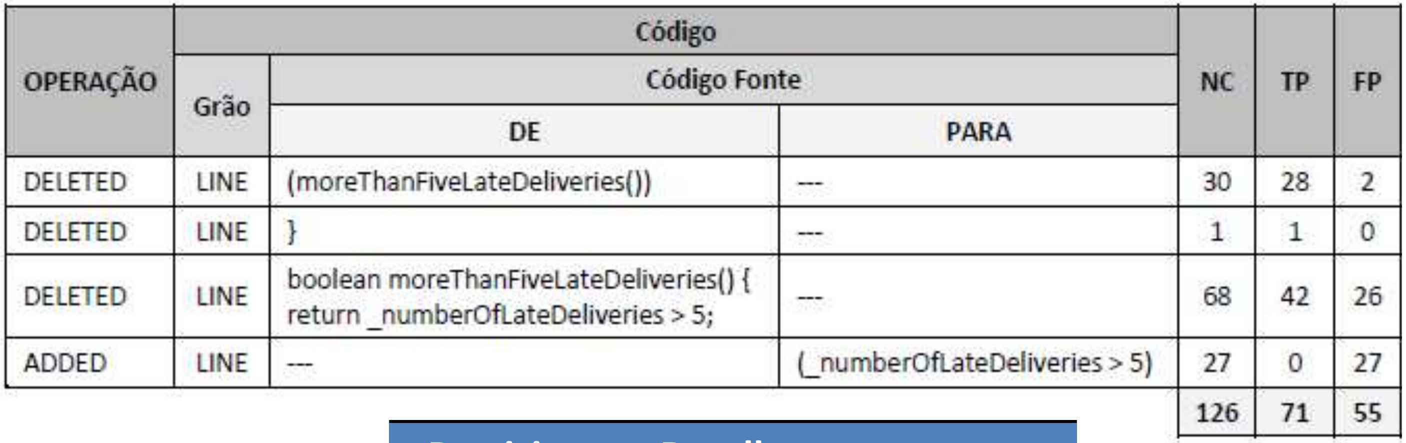


PLANNING AND EXECUTION

Inline Method

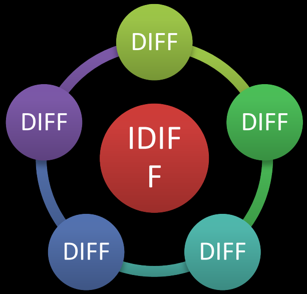


OPERAÇÃO	CÓDIGO			NC	TP	FP
	Grão	Código Fonte				
		DE	PARA			
DELETED	WORD	moreThanFiveLateDeliveries()	---	28	28	0
DELETED	WORD	boolean moreThanFiveLateDeliveries() { return	---	42	42	0
MOVED	WORD	_numberOfLateDeliveries > 5	_numberOfLateDeliveries > 5	25	25	0
DELETED	WORD	}	---	1	1	0
DELETED	WORD	;	---	1	1	0
				97	97	0



OPERAÇÃO	Código			NC	TP	FP
	Grão	Código Fonte				
		DE	PARA			
DELETED	LINE	(moreThanFiveLateDeliveries())	---	30	28	2
DELETED	LINE	}	---	1	1	0
DELETED	LINE	boolean moreThanFiveLateDeliveries() { return _numberOfLateDeliveries > 5;	---	68	42	26
ADDED	LINE	---	(_numberOfLateDeliveries > 5)	27	0	27
				126	71	55

	Precision	Recall	F-measure
IDIFF	1	1	1
WinMerge	0,56	0,73	0,63



Which is **the best granularity** configuration for IDIFF?

Precision					
Line (0.58)	-	Word (0.84)	p-value	<	0.001
Line (0.58)	-	Character (0.75)	p-value	=	0.002
Word (0.84)	-	Character (0.75)	p-value	=	0.002
		Word > Character > Line			
Recall					
Line (0.87)	-	Word (0.78)	p-value	=	0.002
Line (0.87)	-	Character (0.58)	p-value	<	0.001
Word (0.78)	-	Character (0.58)	p-value	<	0.001
		Line > Word > Character			
Harmonic Mean					
Line (0.67)	-	Word (0.8)	p-value	<	0.001
Line (0.67)	-	Character (0.63)	p-value	=	0.487
Word (0.8)	-	Character (0.63)	p-value	<	0.001
		Word > (Line, Character)			

Answer: Word

(Friedman test → Bonferroni corr. → Wilcoxon test)

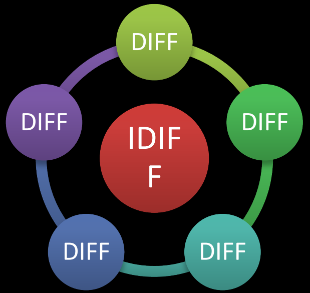
INTRODUCTION

MOTIVATING EXAMPLE

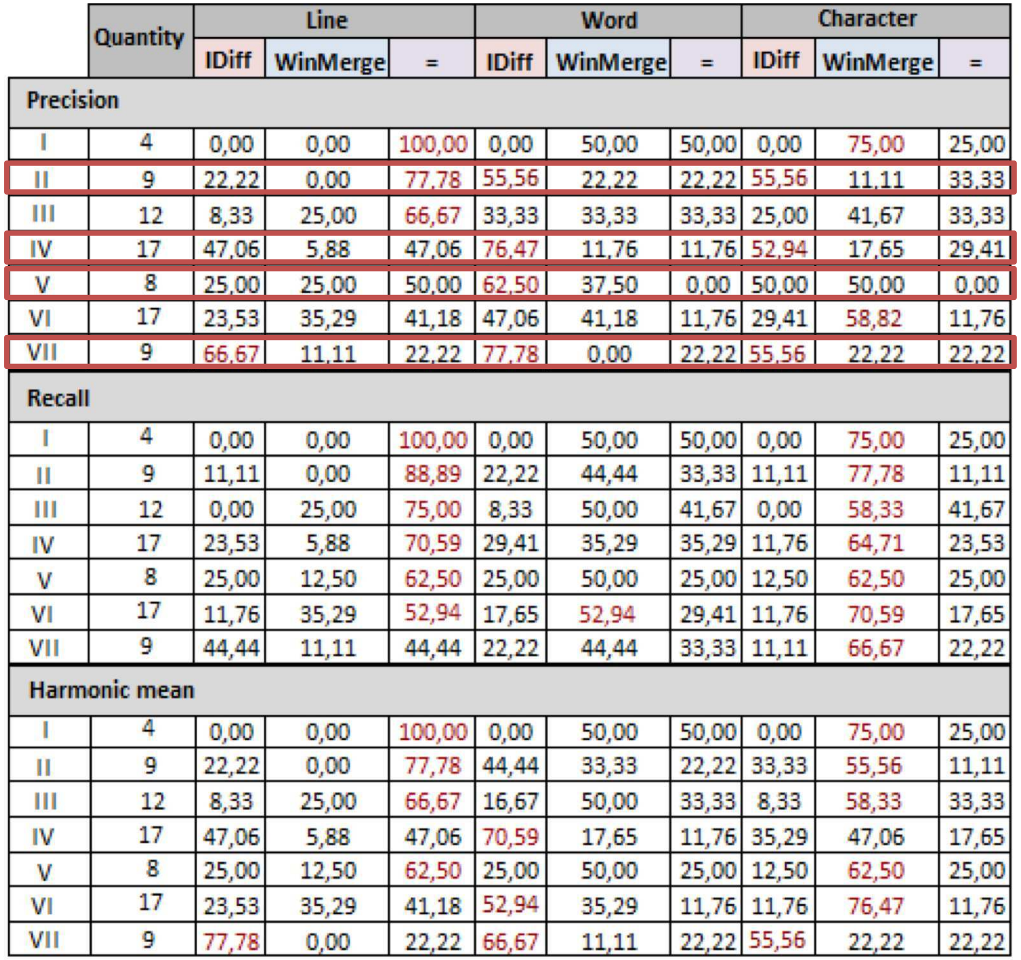
ITERATIVE DIFF (IDIFF)

EVALUATION

CONCLUSIONS



In which situations (refactoring types) IDIFF **is more precise** than a generic Diff tool?



	Quantity	Line			Word			Character		
		IDiff	WinMerge	=	IDiff	WinMerge	=	IDiff	WinMerge	=
Precision										
I	4	0,00	0,00	100,00	0,00	50,00	50,00	0,00	75,00	25,00
II	9	22,22	0,00	77,78	55,56	22,22	22,22	55,56	11,11	33,33
III	12	8,33	25,00	66,67	33,33	33,33	33,33	25,00	41,67	33,33
IV	17	47,06	5,88	47,06	76,47	11,76	11,76	52,94	17,65	29,41
V	8	25,00	25,00	50,00	62,50	37,50	0,00	50,00	50,00	0,00
VI	17	23,53	35,29	41,18	47,06	41,18	11,76	29,41	58,82	11,76
VII	9	66,67	11,11	22,22	77,78	0,00	22,22	55,56	22,22	22,22
Recall										
I	4	0,00	0,00	100,00	0,00	50,00	50,00	0,00	75,00	25,00
II	9	11,11	0,00	88,89	22,22	44,44	33,33	11,11	77,78	11,11
III	12	0,00	25,00	75,00	8,33	50,00	41,67	0,00	58,33	41,67
IV	17	23,53	5,88	70,59	29,41	35,29	35,29	11,76	64,71	23,53
V	8	25,00	12,50	62,50	25,00	50,00	25,00	12,50	62,50	25,00
VI	17	11,76	35,29	52,94	17,65	52,94	29,41	11,76	70,59	17,65
VII	9	44,44	11,11	44,44	22,22	44,44	33,33	11,11	66,67	22,22
Harmonic mean										
I	4	0,00	0,00	100,00	0,00	50,00	50,00	0,00	75,00	25,00
II	9	22,22	0,00	77,78	44,44	33,33	22,22	33,33	55,56	11,11
III	12	8,33	25,00	66,67	16,67	50,00	33,33	8,33	58,33	33,33
IV	17	47,06	5,88	47,06	70,59	17,65	11,76	35,29	47,06	17,65
V	8	25,00	12,50	62,50	25,00	50,00	25,00	12,50	62,50	25,00
VI	17	23,53	35,29	41,18	52,94	35,29	11,76	11,76	76,47	11,76
VII	9	77,78	0,00	22,22	66,67	11,11	22,22	55,56	22,22	22,22

Answer:

II – Composing methods

IV – Making method calls simpler

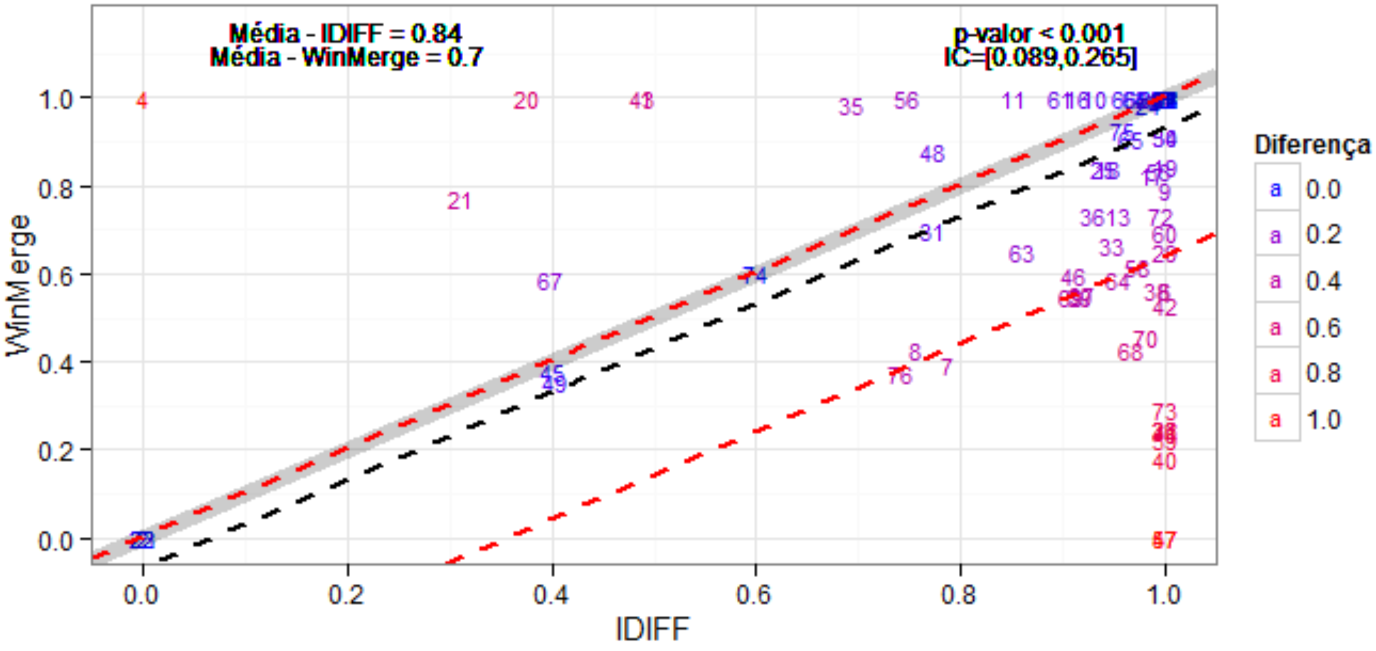
V – Moving features between objects

VII – Simplifying conditional expression

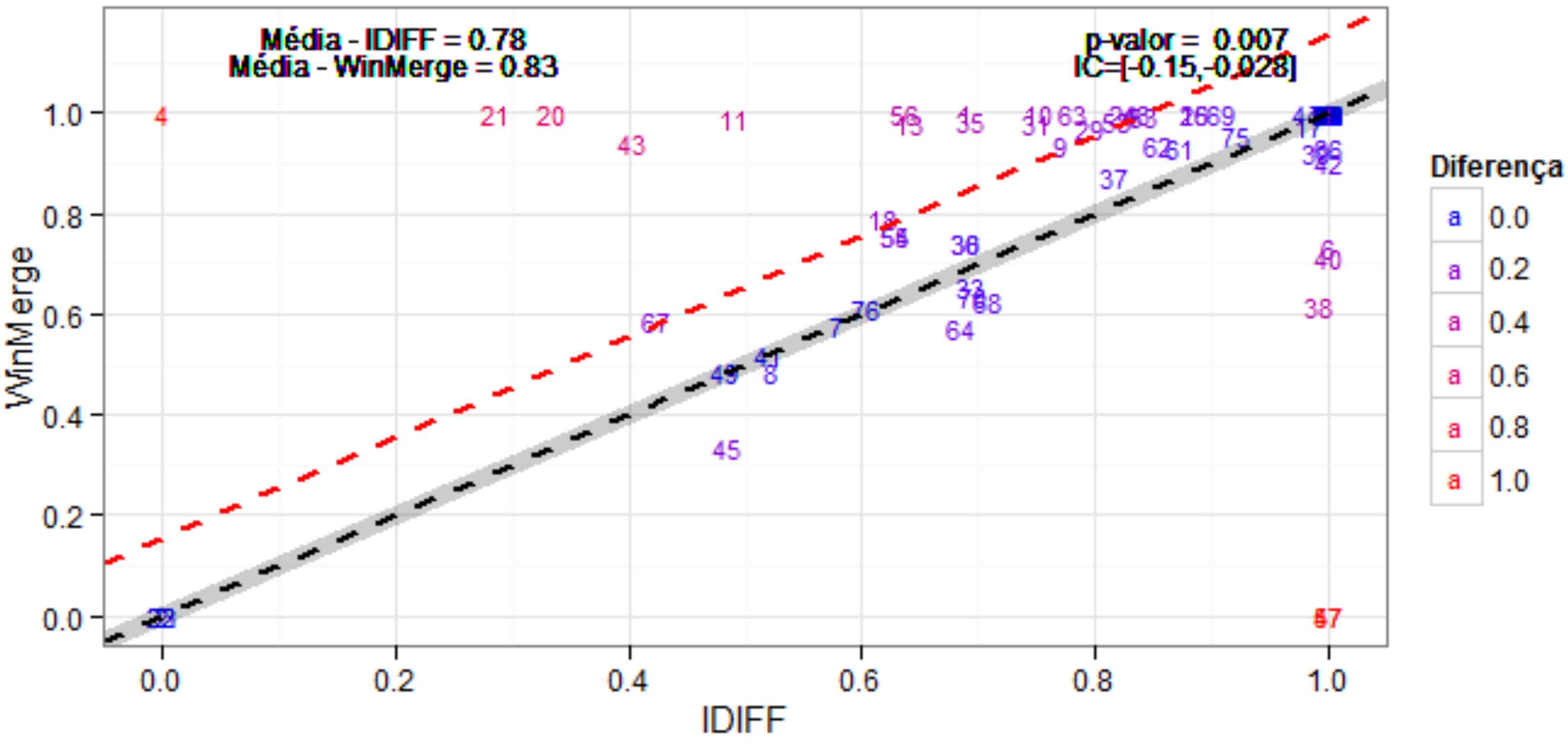
Legend:

Big Refactoring	I
Composing Methods	II
Dealing with generalization	III
Making method calls simpler	IV
Moving Features between objects	V
Organizing data	VI
Simplifying conditional expression	VII

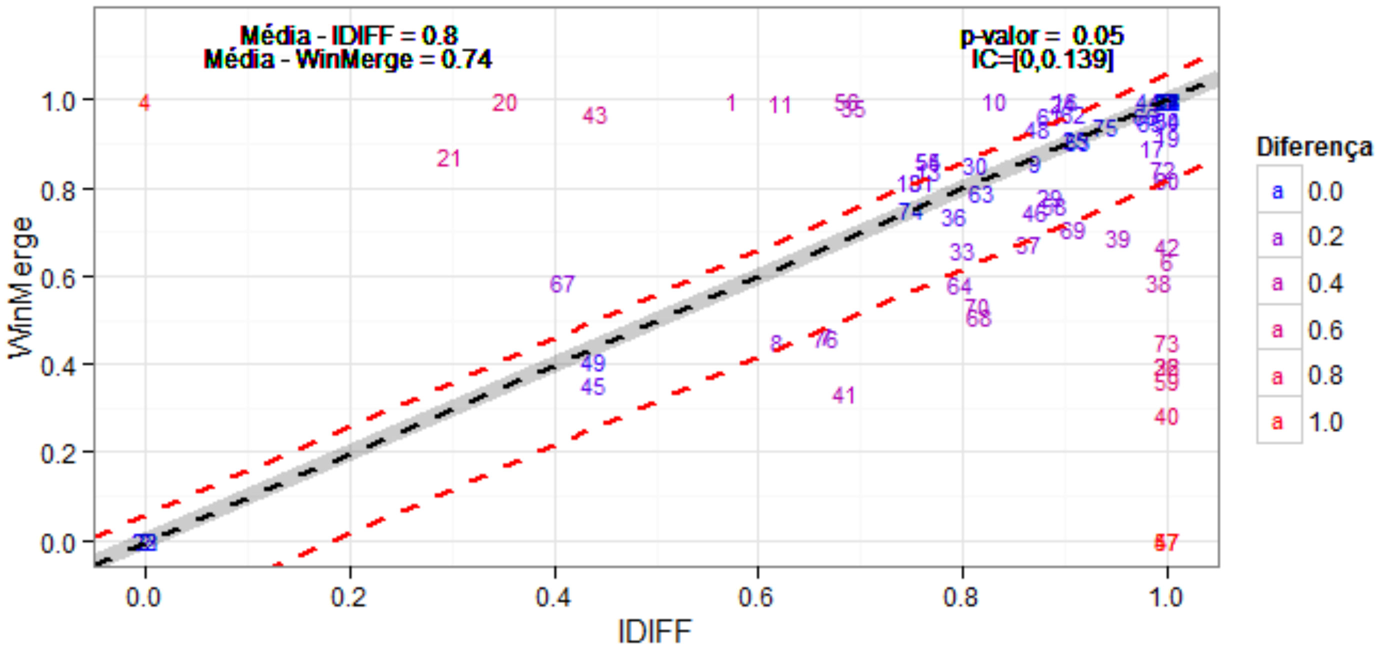
Precision – word grain



Recall – word grain



F-measure – word grain



Related Work

- Clone detection
- Refactoring detection
- Diff
  - Malpohl (2003): rename detection, language specific
  - Canfora et al. (2009): improvements over Unix Diff, line grain
  - Antoniol et al. (2004): evolution discontinuities, language specific