API Design Implications of Boilerplate Client Code

Daye Nam

Carnegie Mellon University

institute for SOFTWARE RESEARCH

Carnegie Mellon University

Code to write an XML document to a specified output stream?

Code to write an XML document to a specified output stream?

Expectation

```
writeXMLDoc(Document doc, OutputStream out);
```

Code to write an XML document to a specified output stream?

Expectation

```
writeXMLDoc(Document doc, OutputStream out);
```

Reality

```
static final void writeDoc(Document doc, OutputStream out) throws IOException {
    try {
        Transformer t = TransformerFactory.newInstance().newTransformer();
        t.setOutputProperty(OutputKeys.DOCTYPE_SYSTEM, doc.getDoctype().getSystemId());
        t.transform(new DOMSource(doc), new StreamResult(out));
    } catch(TransformerException e) {
        throw new AssertionError(e);    //Can't happen!
    }
}
```

```
static final void writeDoc(Document doc, OutputStream out) throws IOException {
    try {
        Transformer t = TransformerFactory.newInstance().newTransformer();
        t.setOutputProperty(OutputKeys.DOCTYPE_SYSTEM, doc.getDoctype().getSystemId());
        t.transform(new DOMSource(doc), new StreamResult(out));
    } catch(TransformerException e) {
        throw new AssertionError(e);    //Can't happen!
    }
}
```

Boilerplate Code

```
static final void writeDoc(Document doc, OutputStream out) throws IOException {
    try {
        Transformer t = TransformerFactory.newInstance().newTransformer();
        t.setOutputProperty(OutputKeys.DOCTYPE_SYSTEM, doc.getDoctype().getSystemId());
        t.transform(new DOMSource(doc), new StreamResult(out));
    } catch(TransformerException e) {
        throw new AssertionError(e);    //Can't happen!
    }
}
```

Boilerplate Code

Hard to understand

Verbose

Error-Prone

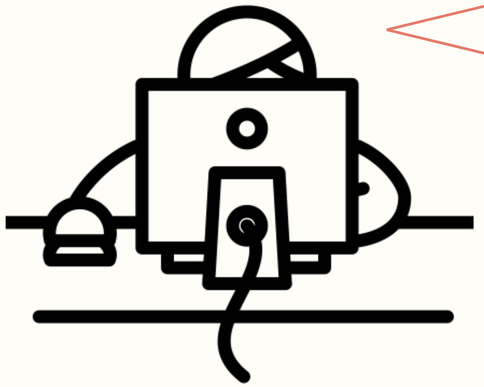
API Design Guidelines suggest

to reduce the need for boilerplate code.

[Mosqueira-Rey et al. 2018, Reddy 2011]

The existence of boilerplate client code may serve as an indicator of poor API design.

API Designer

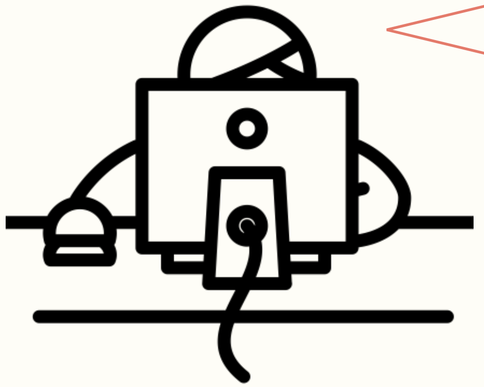


I thought users will need the flexibility,

but most users do not...

```
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                                                != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(
        this,
        new String[]{ Manifest.permission.ACCESS_FINE_LOCATION },
        LOCATION_PERMISSION_REQUEST);
    return;
}
```

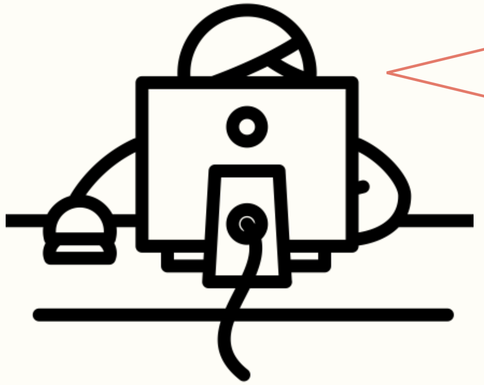
API Designer



My API does not directly provide the methods that programmers need...

```
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                                                != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(
        this,
        new String[]{ Manifest.permission.ACCESS_FINE_LOCATION },
        LOCATION_PERMISSION_REQUEST);
    return;
}
```

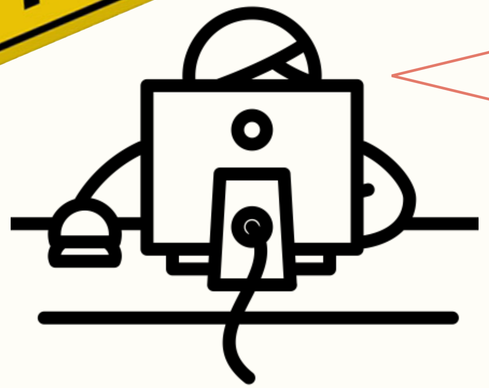
API Boilerplate Code Miner



My API does not directly provide the methods that programmers need…

```
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                                                != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(
        this,
        new String[]{ Manifest.permission.ACCESS_FINE_LOCATION },
        LOCATION_PERMISSION_REQUEST);
    return;
}
```

API Boilerplate Code Miner



My API does not directly provide the methods that programmers need...

```
if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                                                != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(
        this,
        new String[]{ Manifest.permission.ACCESS_FINE_LOCATION },
        LOCATION_PERMISSION_REQUEST);
    return;
}
```

Define Boilerplate Code

Common Properties of Boilerplate

P1 **Annoying!!!**

Common Properties of Boilerplate

P1 Annoying!!!

P2 Frequently Occurs in Client Code

Common Properties of Boilerplate

P1 Annoying!!!

P2 Frequently Occurs in Client Code

P3 Occurs Within a Relatively Condensed Area

Common Properties of Boilerplate

P1	Annoying!!!
P2	Frequently Occurs in Client Code
P3	Occurs Within a Relatively Condensed Area
P4	Used in Similar Forms Without Significant Variations

Common Properties of Boilerplate

Subjective

P1 Annoying!!!

Automatable

P2 Frequently Occurs in Client Code

P3 Occurs Within a Relatively Condensed Area

P4 Used in Similar Forms Without Significant Variations

Mining Boilerplate Code

Overview of Mining Process

High Frequency

Target API & **Client Code Files**

`javax.xml.transform`

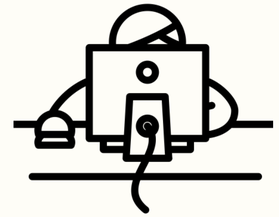
Identifying patterns occurring frequently among the client code

Condensed Area

Identifying patterns occurring within condensed area

Annoying

API Designer

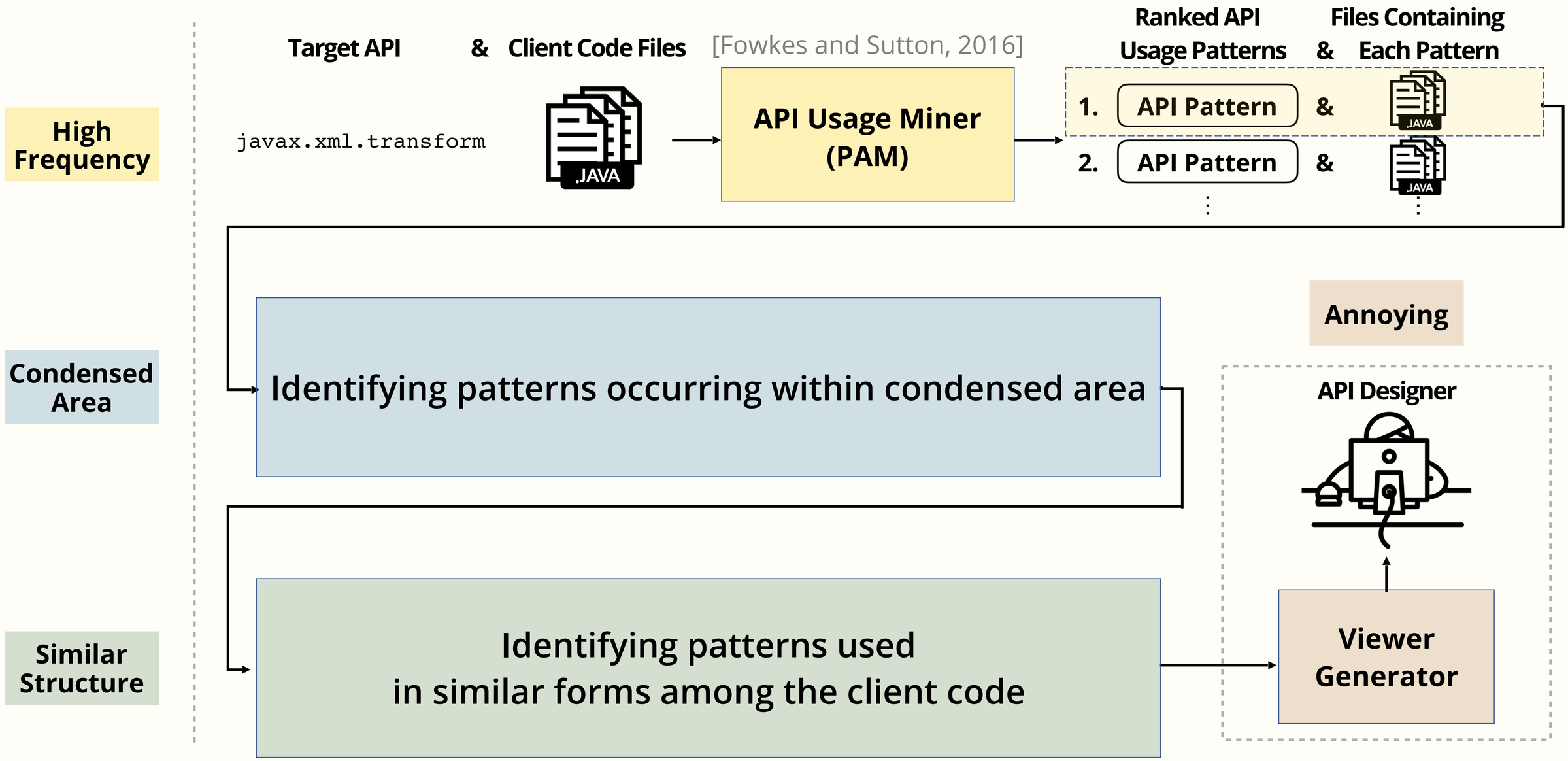


Similar Structure

Identifying patterns used in similar forms among the client code

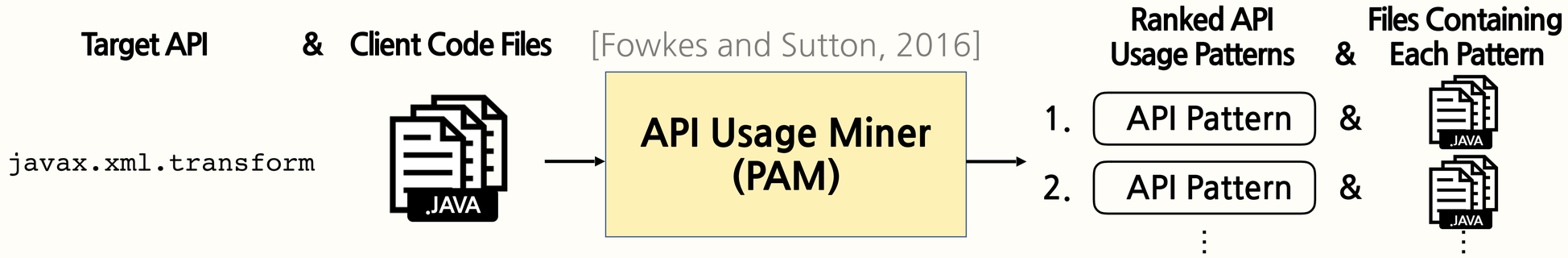
Viewer Generator

Overview of Mining Process



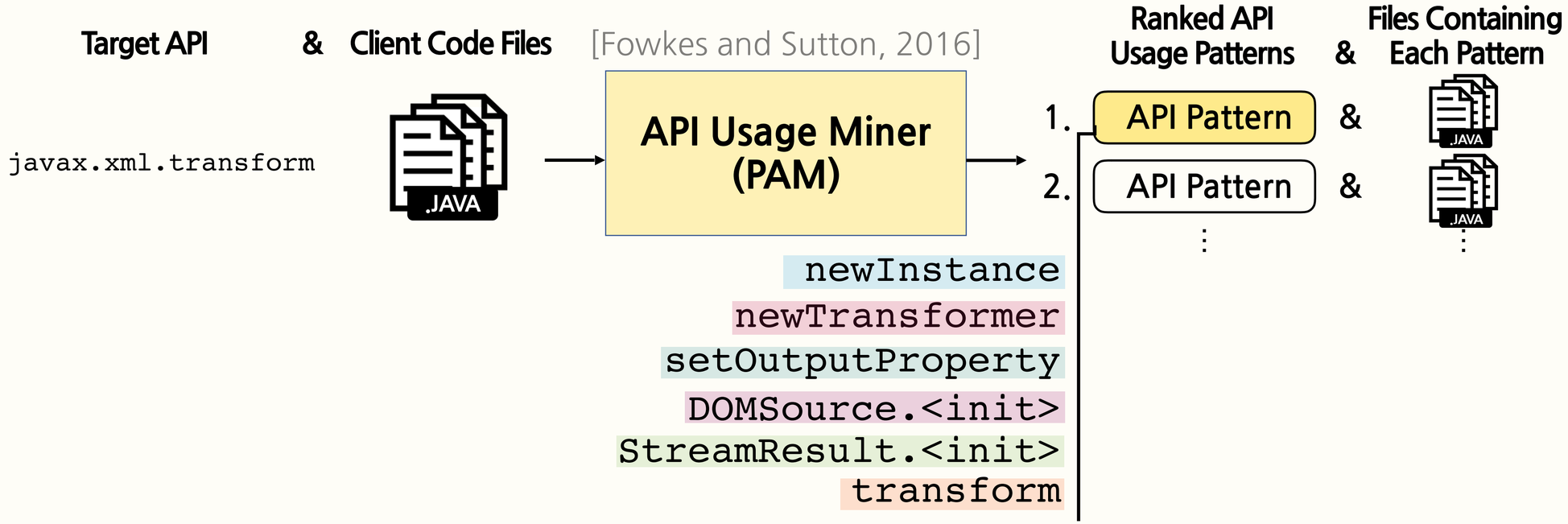
API Usage Mining

High Frequency



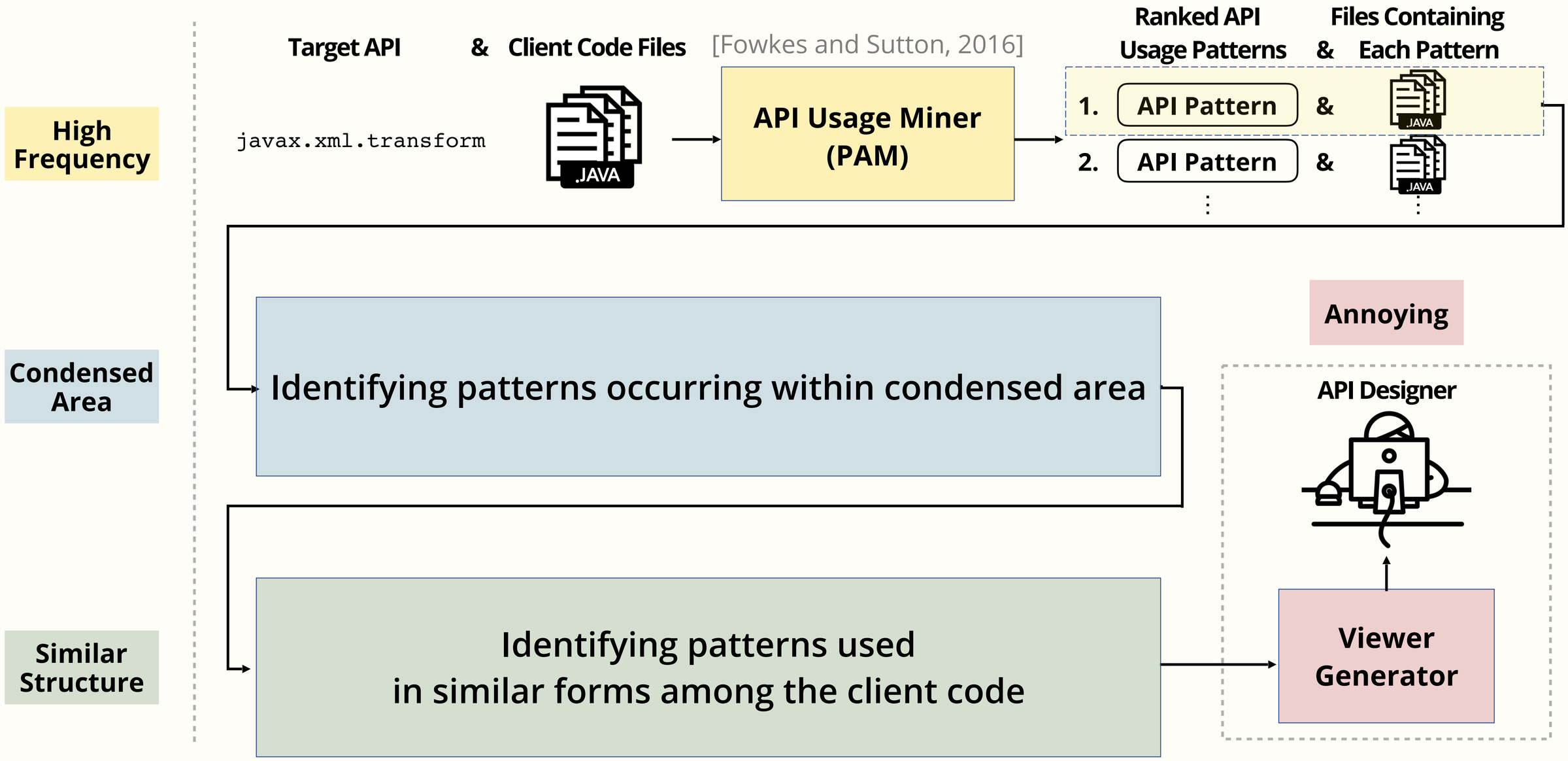
API Usage Mining

High Frequency

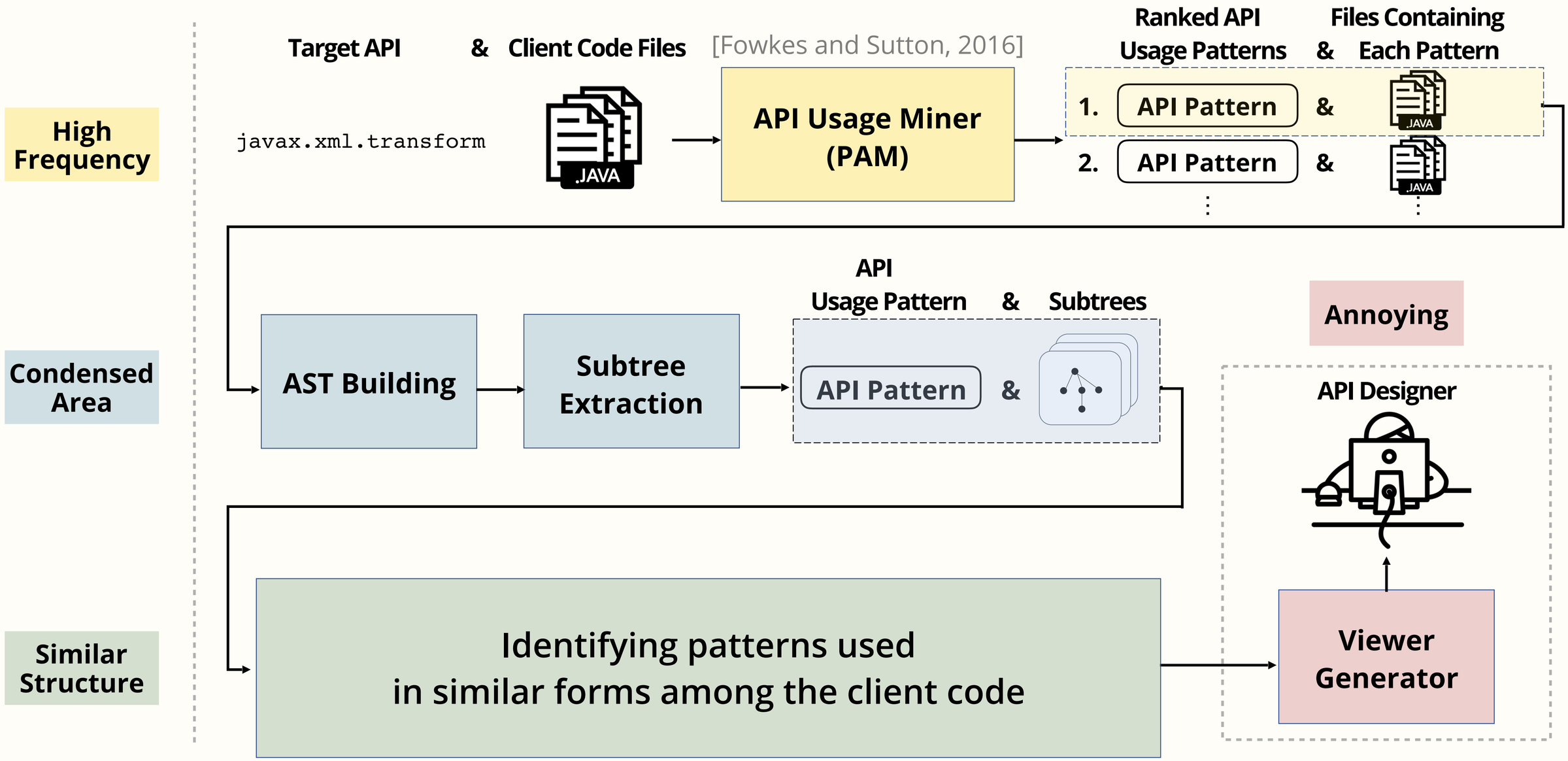



```
static final void writeDoc(Document doc, OutputStream out) throws IOException {
    try {
        Transformer t = TransformerFactory.newInstance().newTransformer();
        t.setOutputProperty(OutputKeys.DOCTYPE_SYSTEM, doc.getDoctype().getSystemId());
        t.transform(new DOMSource(doc), new StreamResult(out));
    } catch(TransformerException e) {
        throw new AssertionError(e);    //Can't happen!
    }
```

Overview of Mining Process

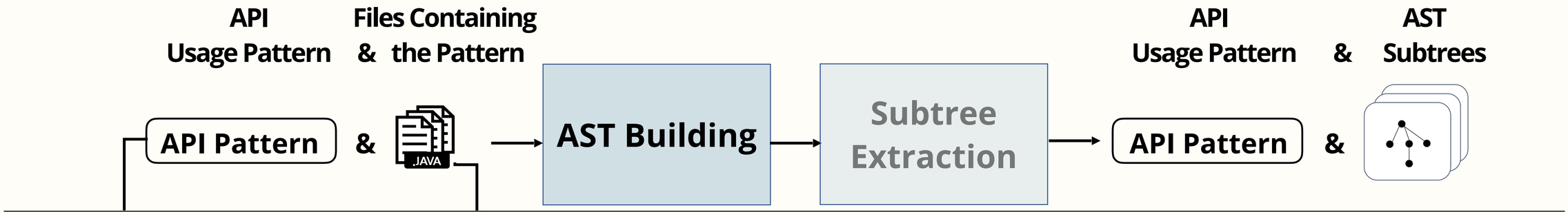


Overview of Mining Process



Subtree Extraction

Condensed Area



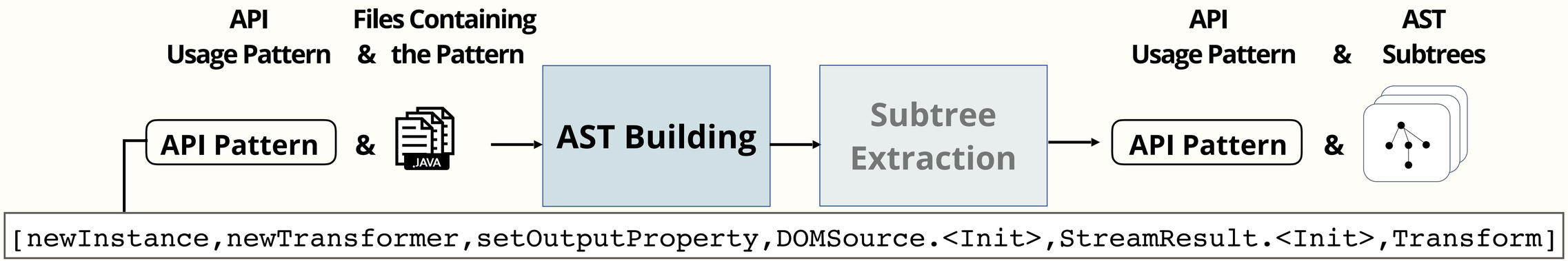

```
[newInstance,newTransformer,setOutputProperty,DOMSource.<Init>,StreamResult.<Init>,Transform]
```

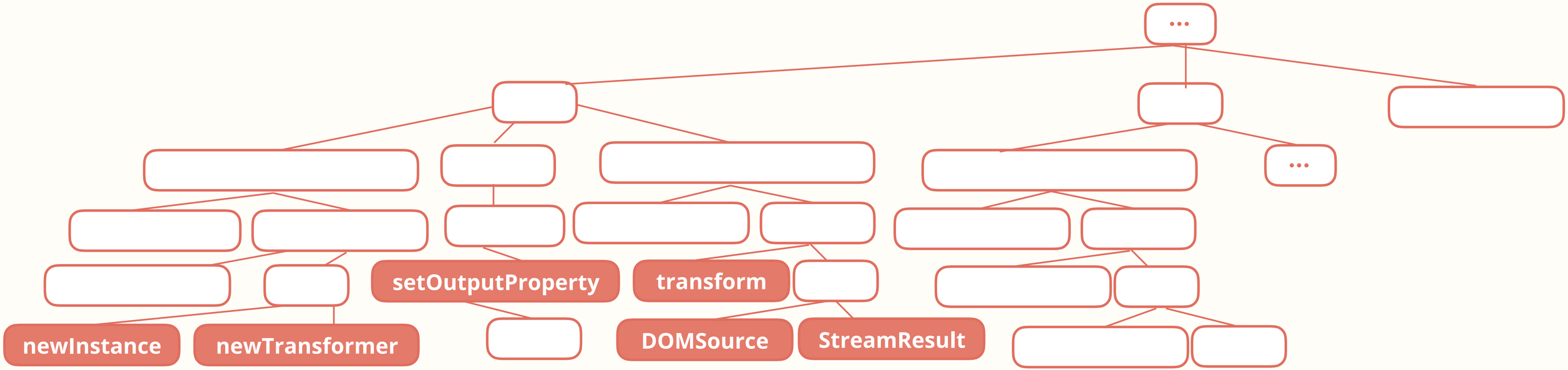
alibaba/fastjson/MiscCodec.java

```
try {
  TransformerFactory transFactory = TransformerFactory.newInstance();
  Transformer transformer = transFactory.newTransformer();
  DOMSource domSource = new DOMSource(node);
  …
```

Subtree Extraction

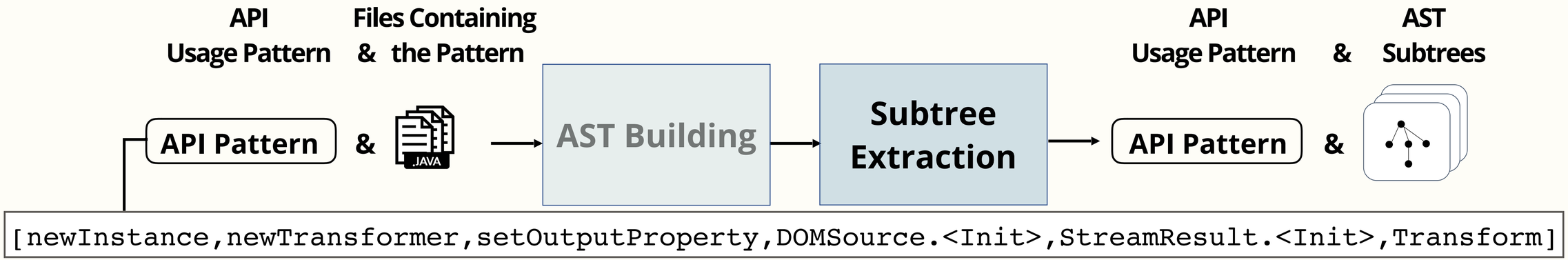
Condensed Area

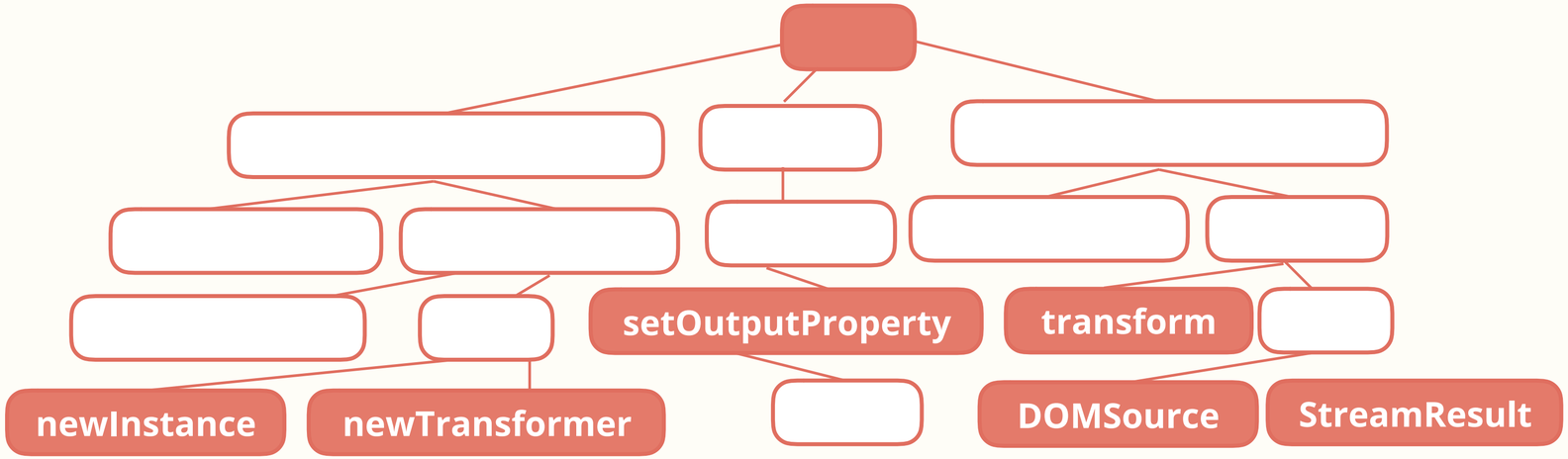




Subtree Extraction

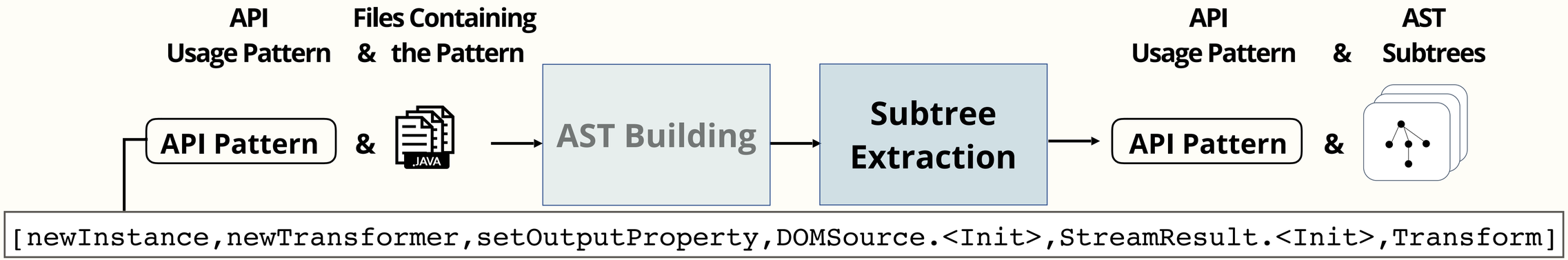
Condensed Area

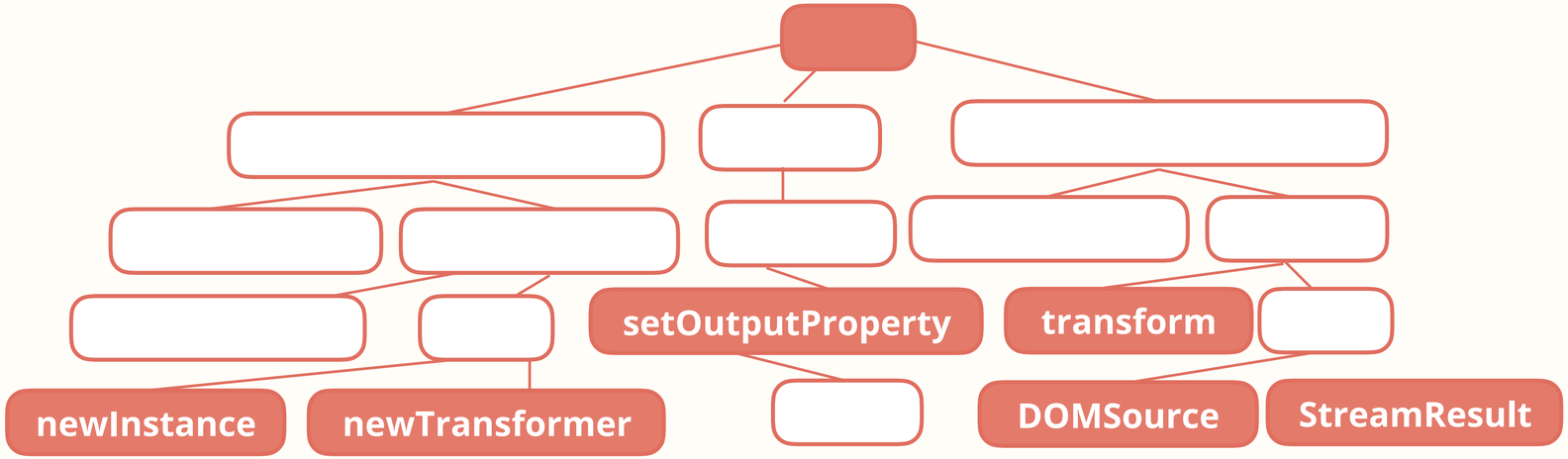




More Likely To Contain Boilerplate

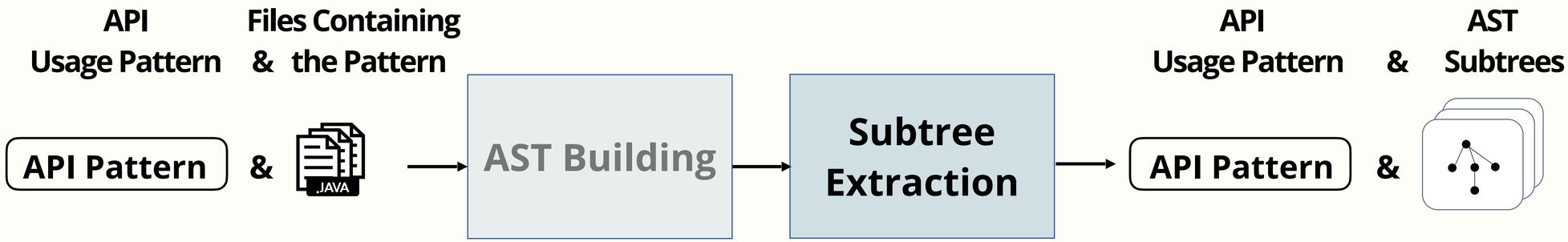
Condensed Area

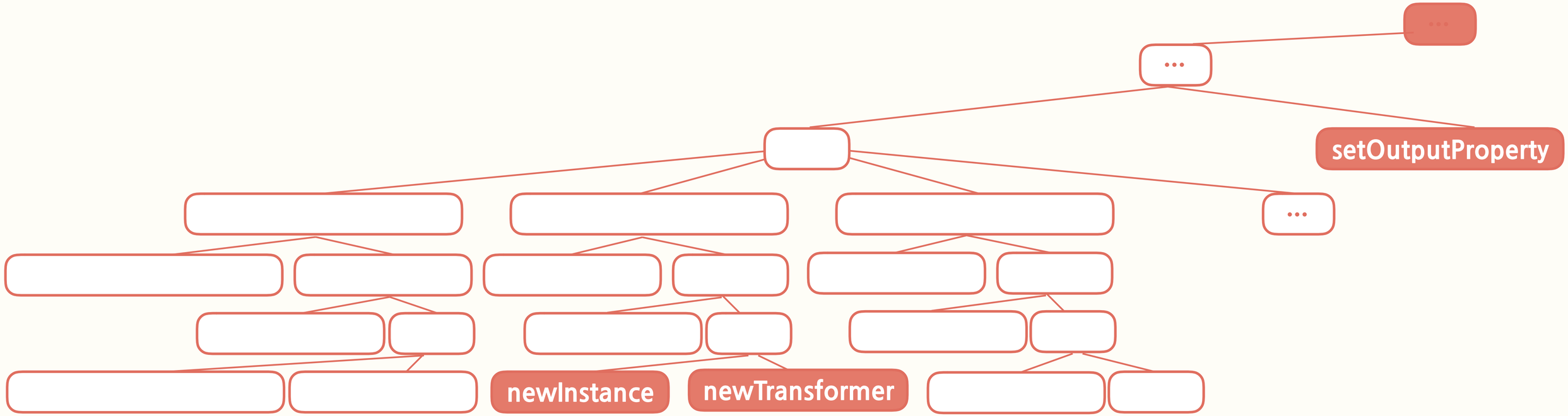




Less Likely To Contain Boilerplate

Condensed Area





Overview of Mining Process

High Frequency

Condensed Area

Similar Structure

Target API & Client Code Files [Fowkes and Sutton, 2016]

javax.xml.transform

API Usage Miner (PAM)

Ranked API Usage Patterns & Files Containing Each Pattern

1. API Pattern &

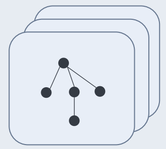
2. API Pattern &

AST Building

Subtree Extraction

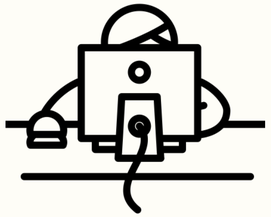
API Usage Pattern & Subtrees

API Pattern &



Annoying

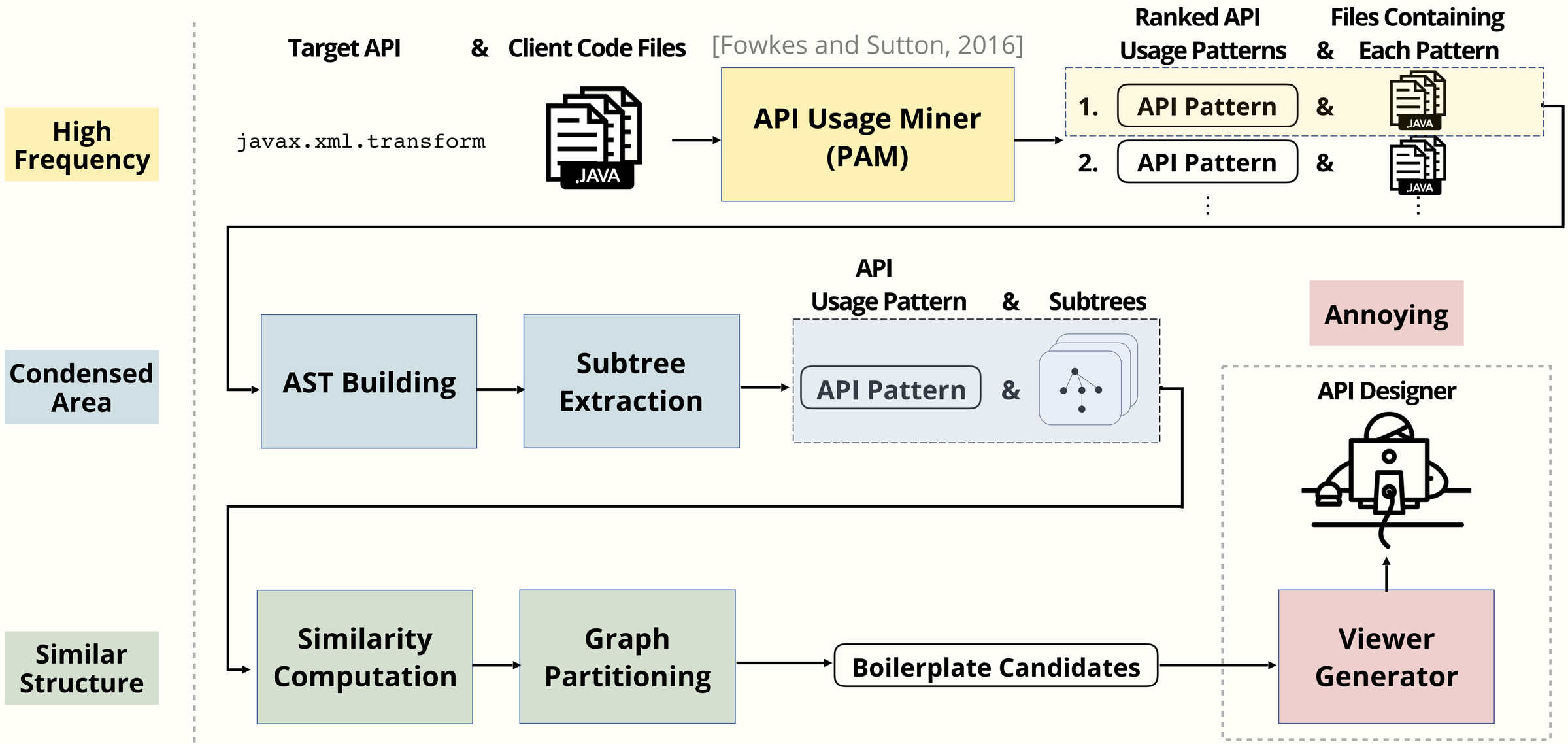
API Designer



Viewer Generator

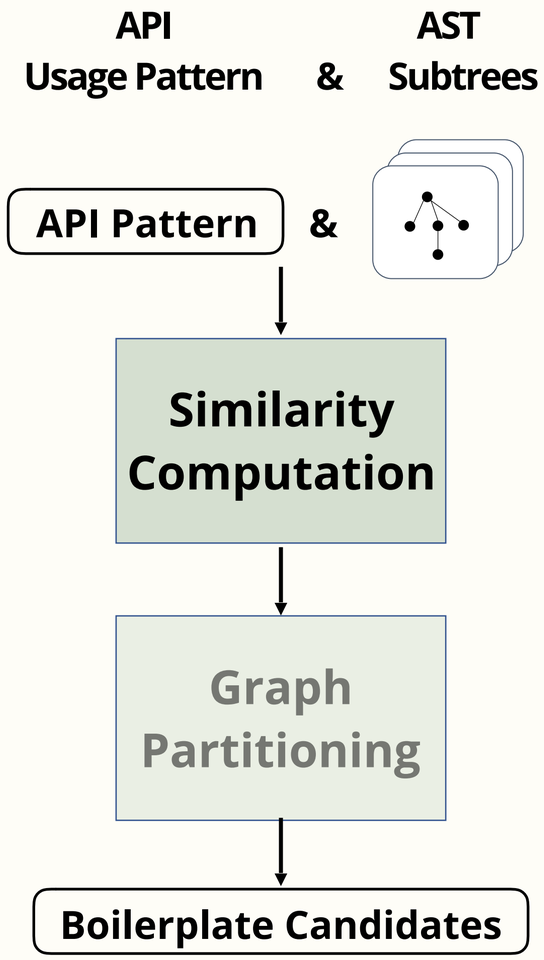
Identifying patterns used
in similar forms among the client code

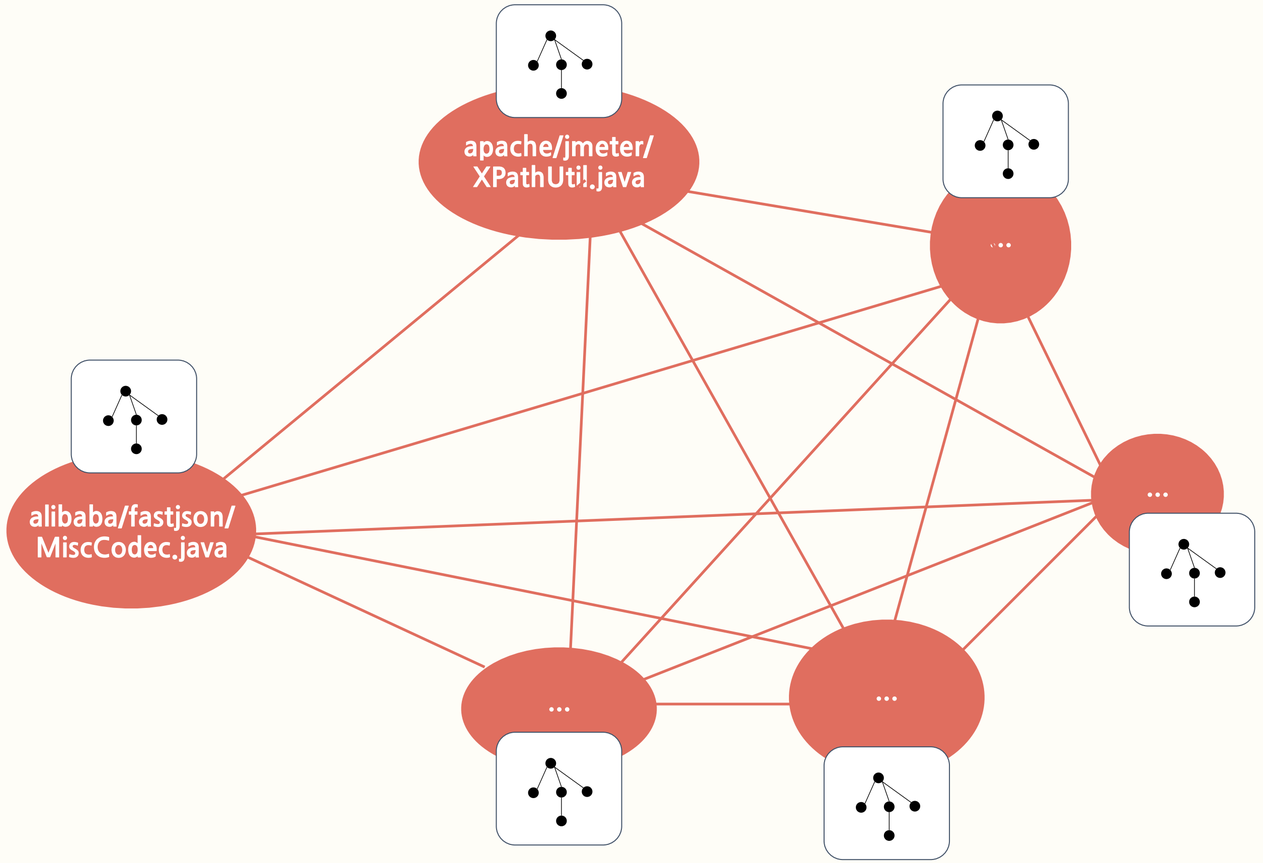
Overview of Mining Process



Similarity Computation

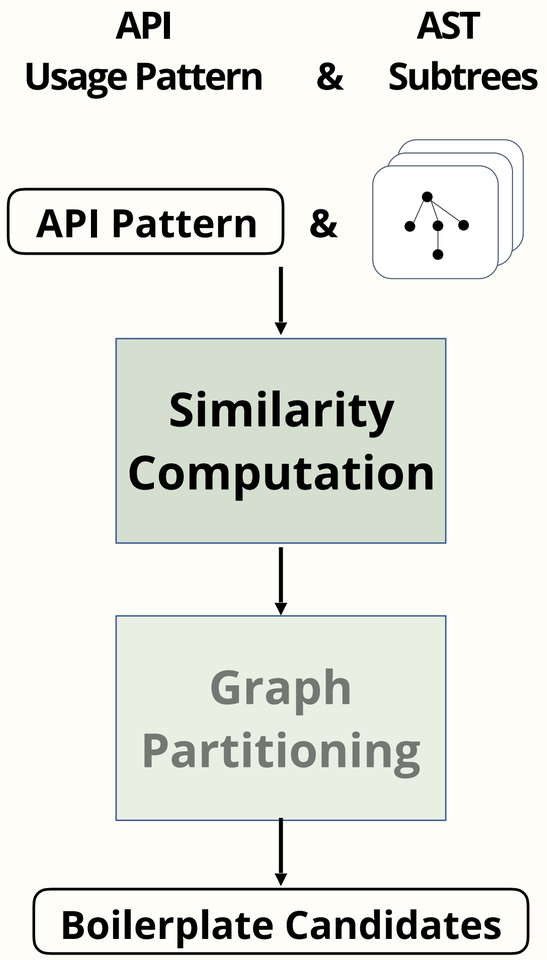
Similar Structure



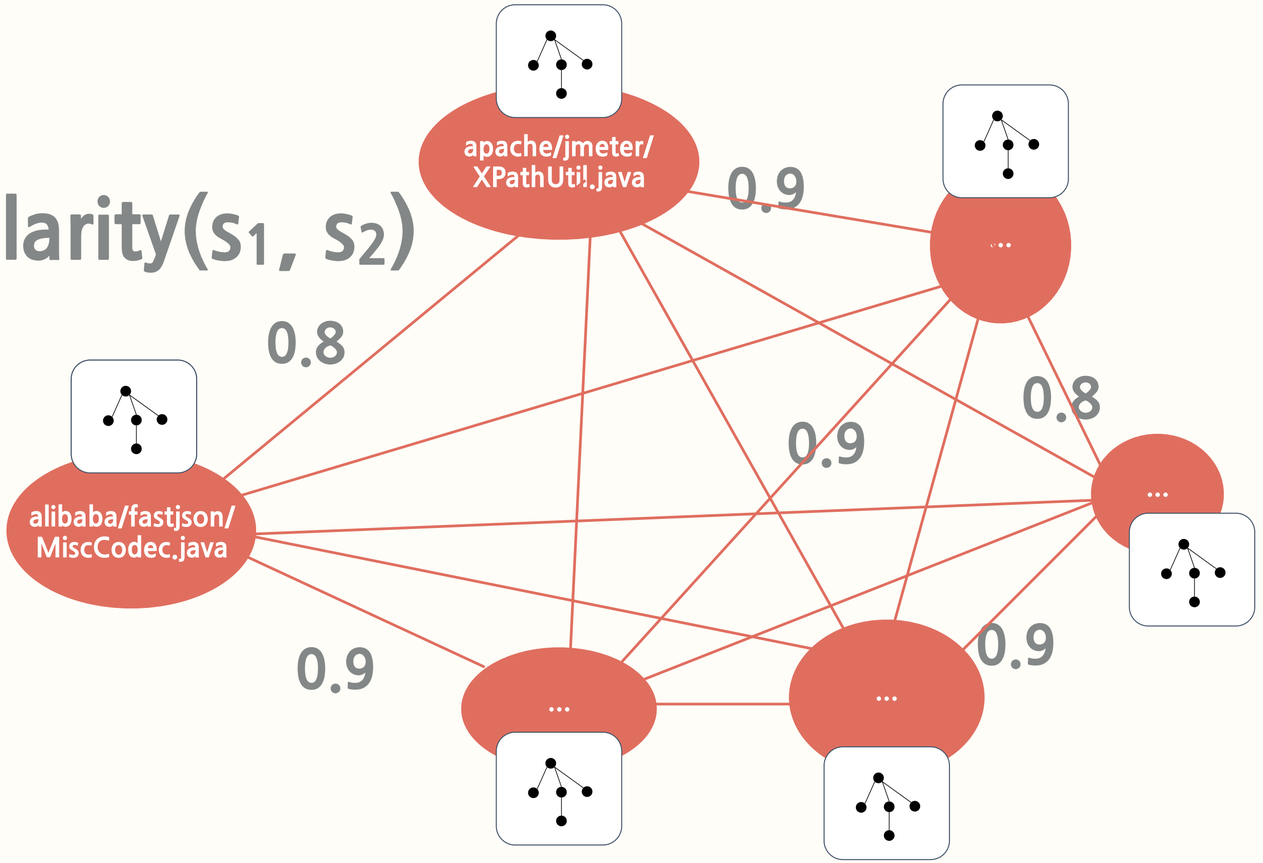


Similarity Computation

Similar Structure

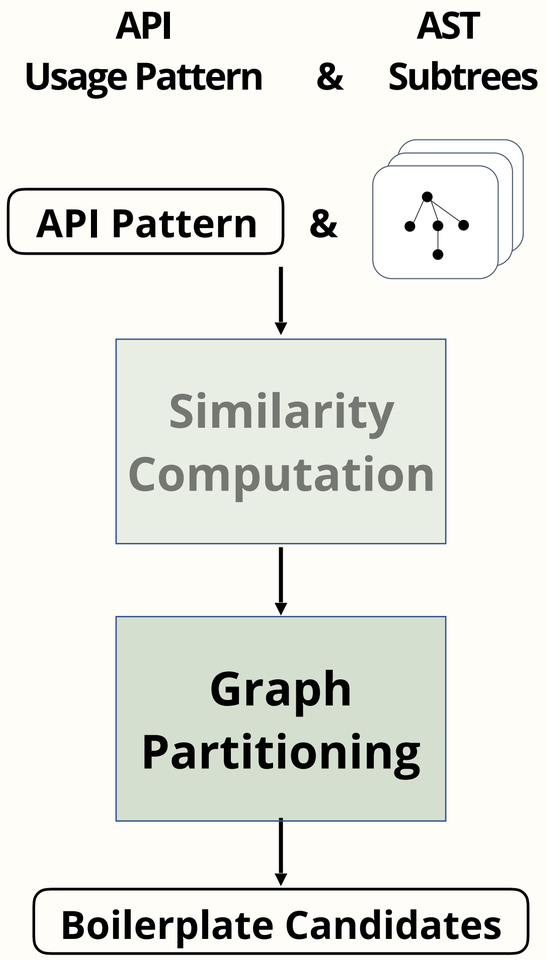


Similarity(s_1, s_2)

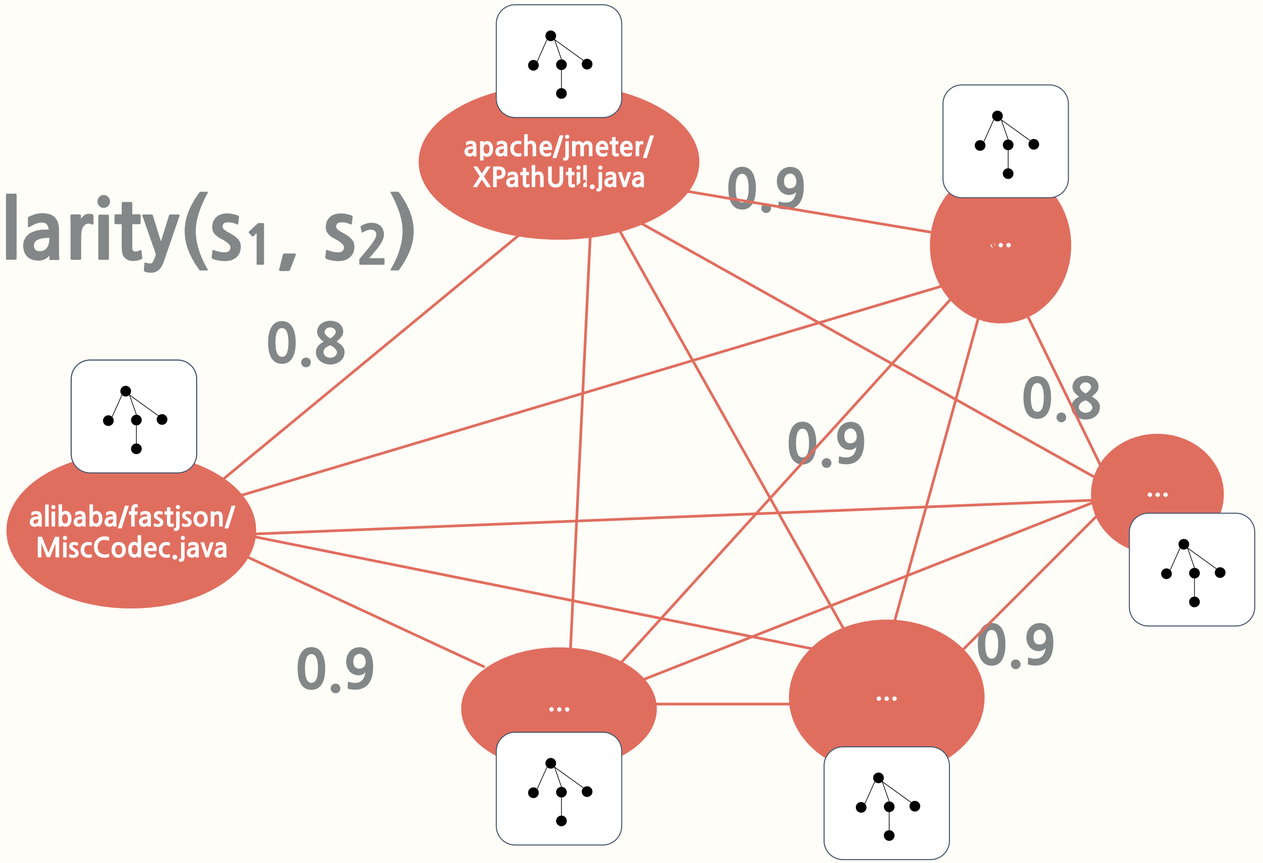


Similarity Computation

Similar Structure

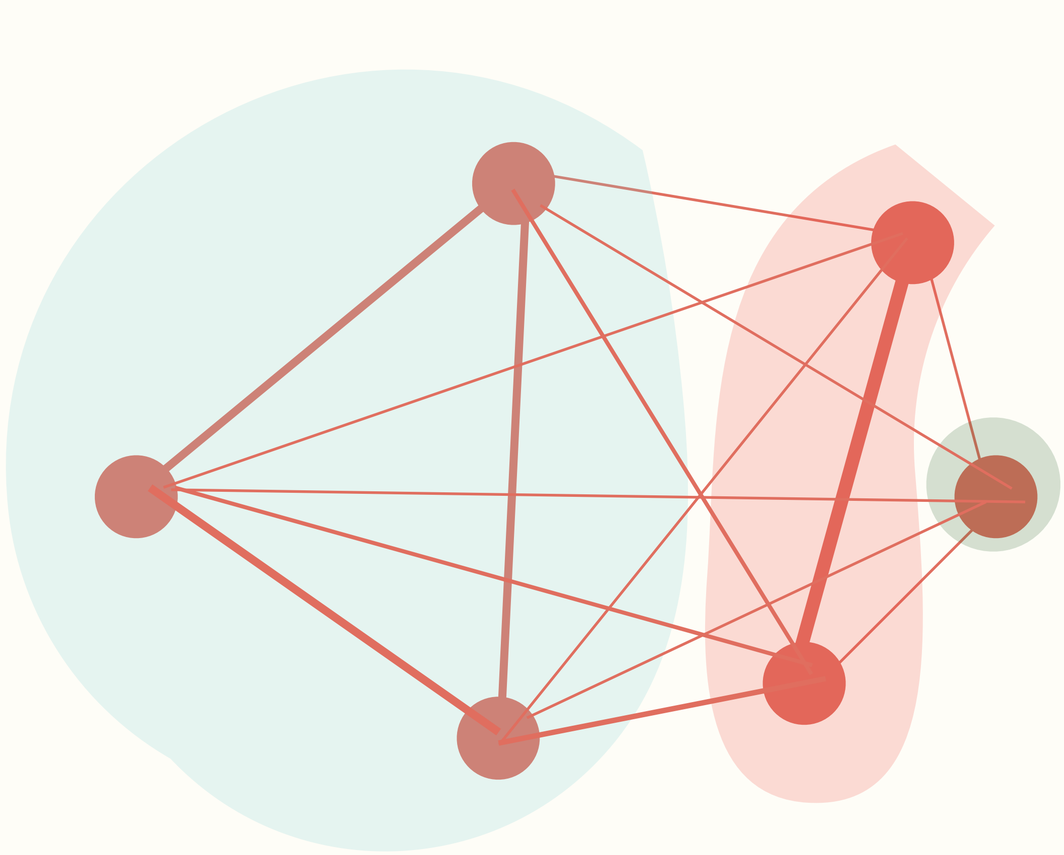


Similarity(s_1, s_2)



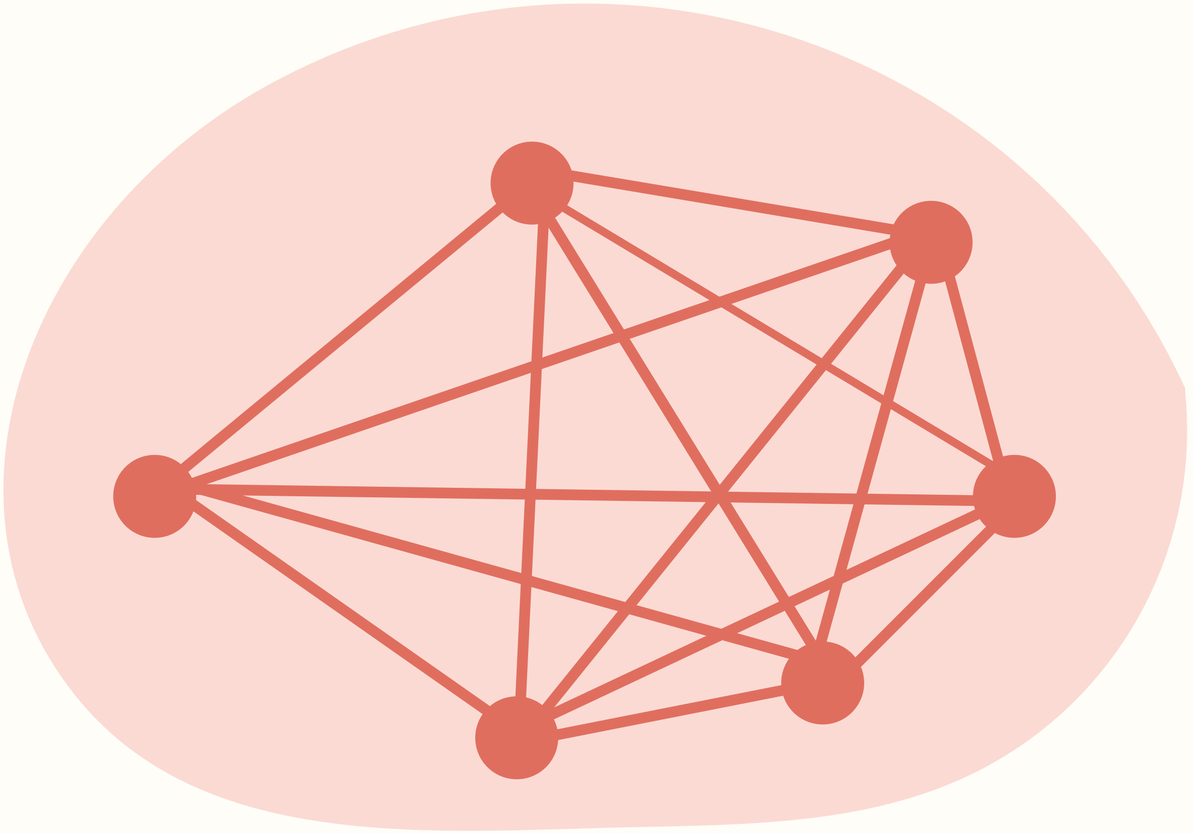
Graph Partitioning

Similar Structure



Less likely Boilerplate

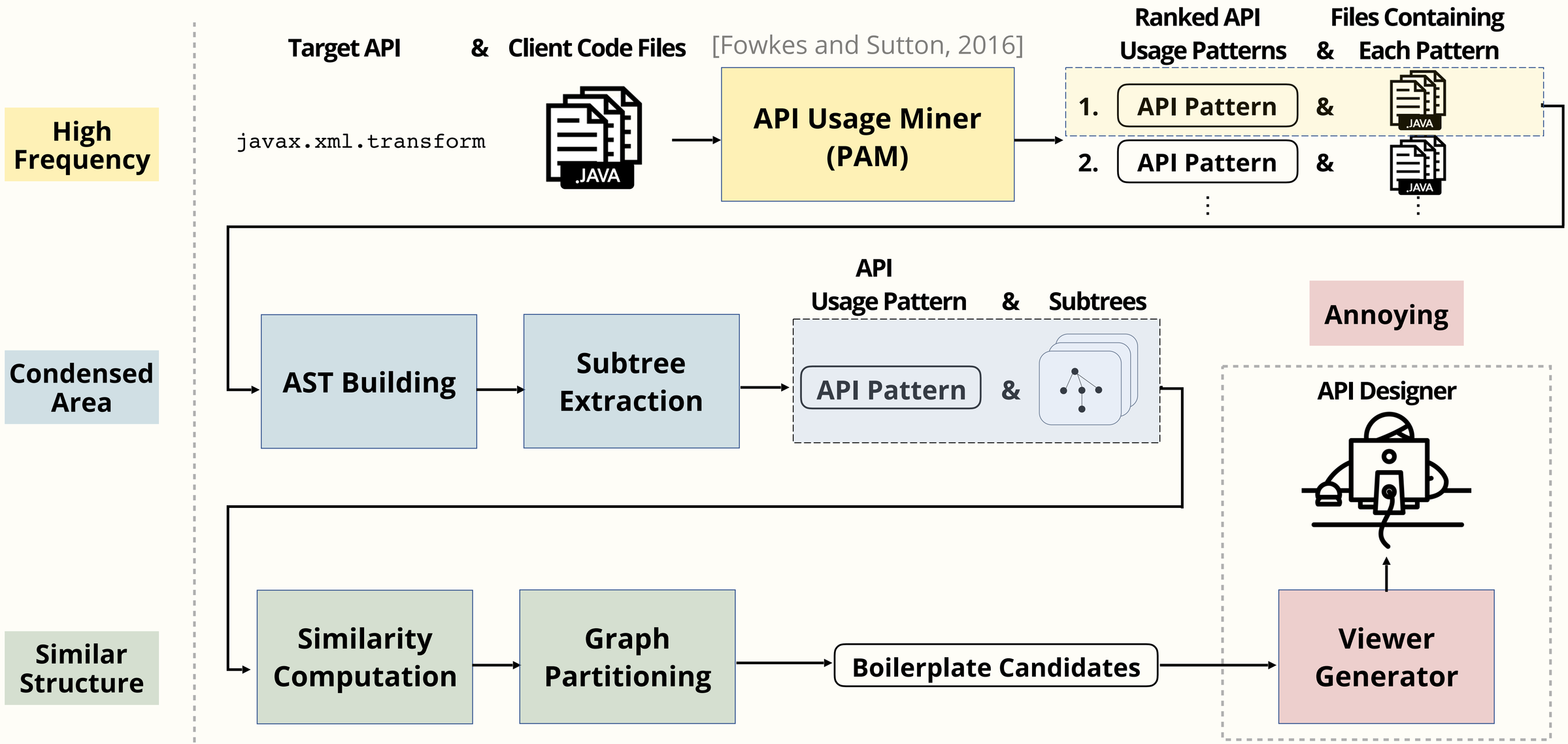
of Clusters > Threshold



More likely Boilerplate

of Clusters ≤ Threshold

Overview of Mining Process



Candidate Viewer

Annoying

javax_xml_transform

API Usage Pattern

pattern_15 (3 partitions, 43 files)

javax.xml.transform.TransformerFactory.newInstance, javax.xml.transform.Transformer.setOutputProperty, javax.xml.transform.dom.DOMSource.<init>, javax.xml.transform.stream.StreamResult.<init>, javax.xml.transform.Transformer.transform

Cluster 0 (14 files, similarity: 0.40290943956043934)

mozilla-mobile____focus-android____SearchEngineManager

```
        try {
            final Transformer tf = TransformerFactory.newIns
            tf.setOutputProperty(OutputKeys.ENCODING, "UTF-8
            tf.transform(new DOMSource(doc), new StreamResul
        } catch (TransformerConfigurationException e) {
            return null;
        } catch (TransformerException e) {
            return null;
        }
```

geoserver____geoserver____CatalogWriter

```
    public void write(File file) throws IOException {
        try (FileOutputStream os = new FileOutputStream(file
            Transformer tx = TransformerFactory.newInstance(
            tx.setOutputProperty(OutputKeys.INDENT, "yes");
            DOMSource source = new DOMSource(document);
            StreamResult result = new StreamResult(os);

            tx.transform(source, result);
        } catch (Exception e) {
            String msg = "Could not write catalog to " + fil
            throw (IOException) new IOException(msg).initCau
        }
    }
```

jOOQ____jOOX____Util

```
    static final String toString(Element element) {
        try {
            ByteArrayOutputStream out = new ByteArrayOutputS
            Transformer transformer = TransformerFactory.new
            transformer.setOutputProperty(OutputKeys.OMIT_XM
            Source source = new DOMSource(element);
            Result target = new StreamResult(out);
            transformer.transform(source, target);
            return out.toString("UTF-8");
        }
        catch (Exception e) {
            return "[ ERROR IN toString() : " + e.getMessage
        }
    }
```

Representative Boilerplate Client Code

Evaluation

Evaluation Dataset

13 Java APIs Client code from 10,000 Github Java repositories

1	android.app.ProgressDialog
2	android.database.sqlite
3	android.support.v4.app.ActivityCompat
4	android.view.View
5	com.squareup.picasso
6	java.beans.ProperrtyChangeSupport
7	java.beans.PropertyChangeEvent
8	java.io.BufferedReader
9	java.sql.DriverManager
10	java.swing.JFrame
11	Javax.swing.SwingUtilities
12	java.xml.parsers
13	java.xml.transform

Evaluation Dataset

My approach returned 59 boilerplate candidates

1	android.app.ProgressDialog	12
2	android.database.sqlite	7
3	android.support.v4.app.ActivityCompat	5
4	android.view.View	11
5	com.squareup.picasso	0
6	java.beans.ProperrtyChangeSupport	8
7	java.beans.PropertyChangeEvent	5
8	java.io.BufferedReader	3
9	java.sql.DriverManager	0
10	java.swing.JFrame	0
11	Javax.swing.SwingUtilities	2
12	java.xml.parsers	3
13	java.xml.transform	3

Precision

1	android.app.ProgressDialog	12
2	android.database.sqlite	7
3	android.support.v4.app.ActivityCompat	5
4	android.view.View	11
5	com.squareup.picasso	0
6	java.beans.ProperrtyChangeSupport	8
7	java.beans.PropertyChangeEvent	5
8	java.io.BufferedReader	3
9	java.sql.DriverManager	0
10	java.swing.JFrame	0
11	Javax.swing.SwingUtilities	2
12	java.xml.parsers	3
13	java.xml.transform	3

56%

Out of 59 boilerplate candidates, 33 judged to be boilerplate

More than 1 out of 2 results are worth looking

Validation

1	android.app.ProgressDialog	12
2	android.database.sqlite	7
3	android.support.v4.app.ActivityCompat	5
4	android.view.View	11
5	com.squareup.picasso	0
6	java.beans.ProperrtyChangeSupport	8
7	java.beans.PropertyChangeEvent	5
8	java.io.BufferedReader	3
9	java.sql.DriverManager	0
10	java.swing.JFrame	0
11	Javax.swing.SwingUtilities	2
12	java.xml.parsers	3
13	java.xml.transform	3

9 out of 13

Out of 13 known Boilerplate Instances (one for each API)

My approach identified 9

Boilerplate Review Example

API `android.database.sqlite`

Pattern `[execSQL, onCreate]`

Boilerplate Review Example

API android.database.sqlite

Pattern [execSQL, onCreate]

Client Code

```
@Override
public void onUpgrade(
            SQLiteDatabase db, int oldVersion, int currentVersion) {
    Log.w(TAG, ”Upgrading test database from version ”
                + oldVersion + ” to ” + currentVersion
                + ”, which will destroy all old data”);
    db.execSQL(”DROP TABLE IF EXISTS data”);
    onCreate(db);
}
```

Boilerplate Review Example

API

`android.database.sqlite`

Pattern

`[execSQL, onCreate]`

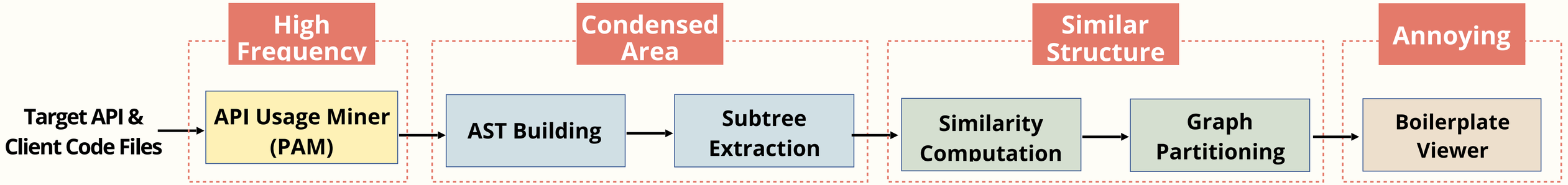
Client Code

```
@Override
public void onUpgrade(
            SQLiteDatabase db, int oldVersion, int currentVersion) {
    Log.w(TAG, "Upgrading test database from version "
                + oldVersion + " to " + currentVersion
                + ", which will destroy all old data");
    db.execSQL("DROP TABLE IF EXISTS data");
    onCreate(db);
}
```

Potential Improvement

To make the common usage as the default functionality of `onUpgrade`.

API Design Implications of Boilerplate Client Code



- The existence of boilerplate code may serve as an indicator of poor API usability.
- My approach can identify known and new boilerplate instances for the manual review of API design.

Source code and the result are available at https://dayenam.com/MARBLE

Daye Nam: dayen@andrew.cmu.edu