M-Track: A Metric Tool Framework for Monitoring the Evolution of OO Systems

Fraunhofer

Institut
Experimentelles
Software Engineering

Dharmalingam Ganesan
Jean-François Girard
Sauerwiesen 6
D-67661 Kaiserslautern
Germany

Outline

- Introduction
- Metrics for Evolution
- Tool Framework
- Conclusion and Future Work

What is Evolution ?

- Changes made on software systems by introducing features

Why Monitor Evolution ?

- To know where the changes happened between releases
- To identify the places unchanged between releases
- To control the changes

Low coupling and high cohesion will make evolution easier !

Metrics for Evolution

Why Use Metrics To Monitor The Evolution ?

- Metrics can identify artefacts with unusual measurement values
- Metrics can be computed directly from the source code – source code is up-to-date

Monitor evolution using coupling, cohesion and size metrics

- **Coupling through**
  - Method invocation
  - Data dependencies (For e.g., Attributes of class)
  - Inheritance
- **Cohesion**
  - To measure the dependency within a class
- **Size**
  - Number of methods implemented in a class

Metrics for Evolution

Selecting subset of metrics

- Practioners can‘t monitor all the existing metrics

Use Principal component analysis (PCA) to select subset of metrics:

- Select those metrics which can explain the rest of the metrics

Requirements of the Tool Framework

Language and Fact Extractor Independent

- Evolution analysis for more than one OO language (Java, Delphi, C++)
- Change the fact extractor tool – if better one is available !

Extendable

- New metrics can be easily introduced (e.g. metrics at package level)
- Multiple criterias for analysing the evolution results

Inter-operable with other tools

- Export the evolution results into visualization, statistical analysis tools

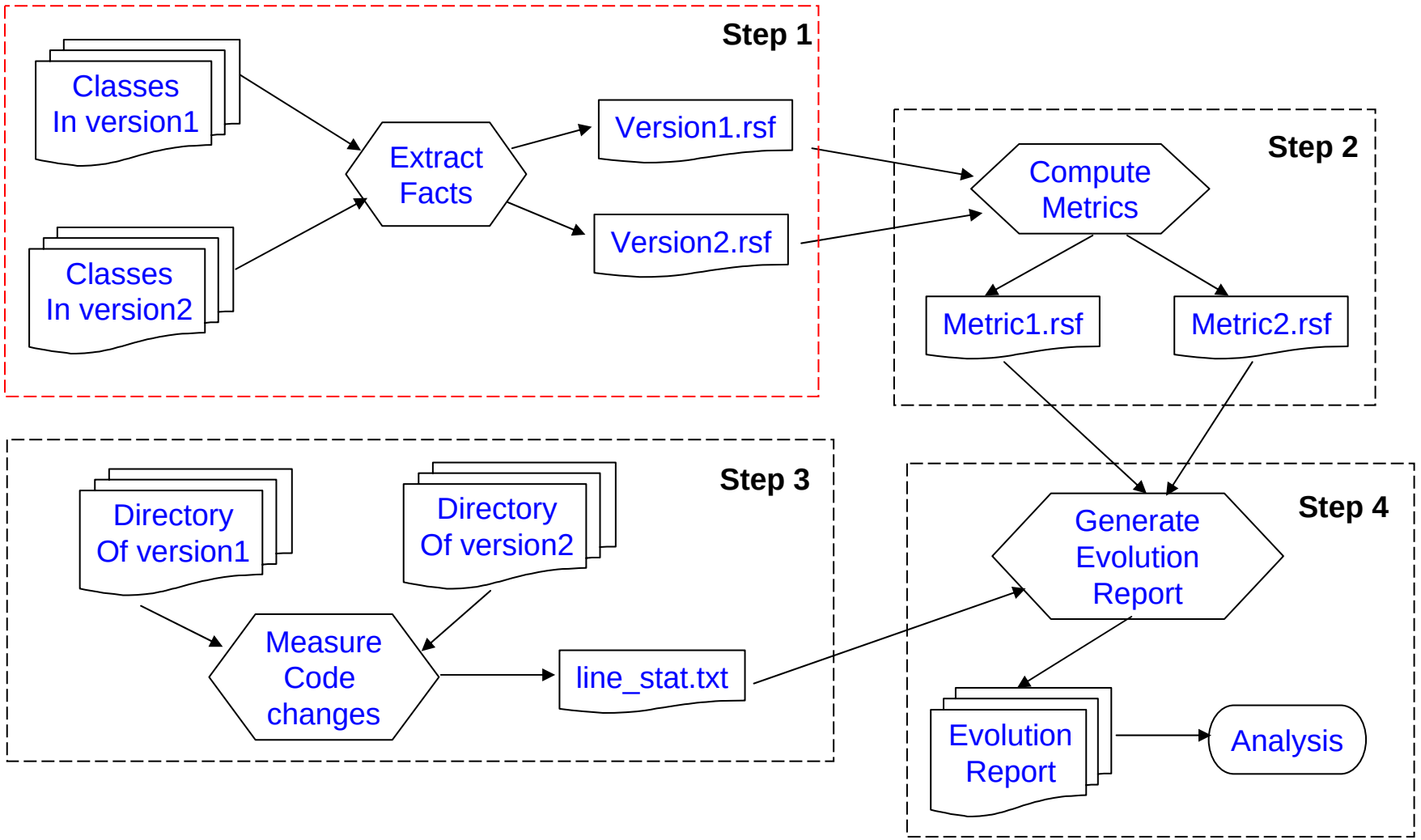
Scalable

- Able to handle large industrial systems

Portable

- Able to run under in different platforms at various customers site

M-Track:Tool Framework



M-Track for Java: Extraction Phase

Fact Extraction using Chava (AT &T)

- Chava can extract facts direcly from the class files generated by javac
- Chava is fast and scalable
- Necessary facts for computing the defined metrics is extracted

Problems in Chava

- Chava has problem with calls to base class methods
- Chava doesn‘t distinguish 2 classes with same name in different packages

We fixed these Chava problems by directly changing the intermediate files generated by the Chava

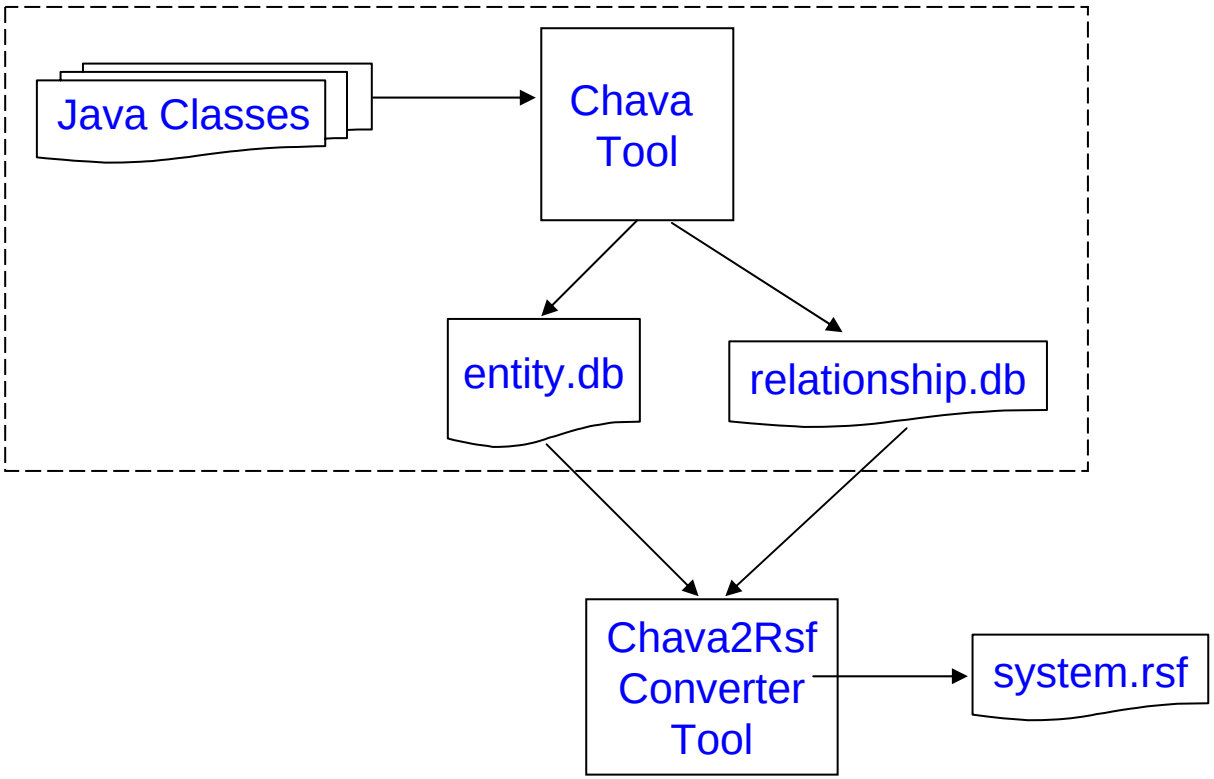
M-Track for Java: Extraction Phase

Converting to RSF Format

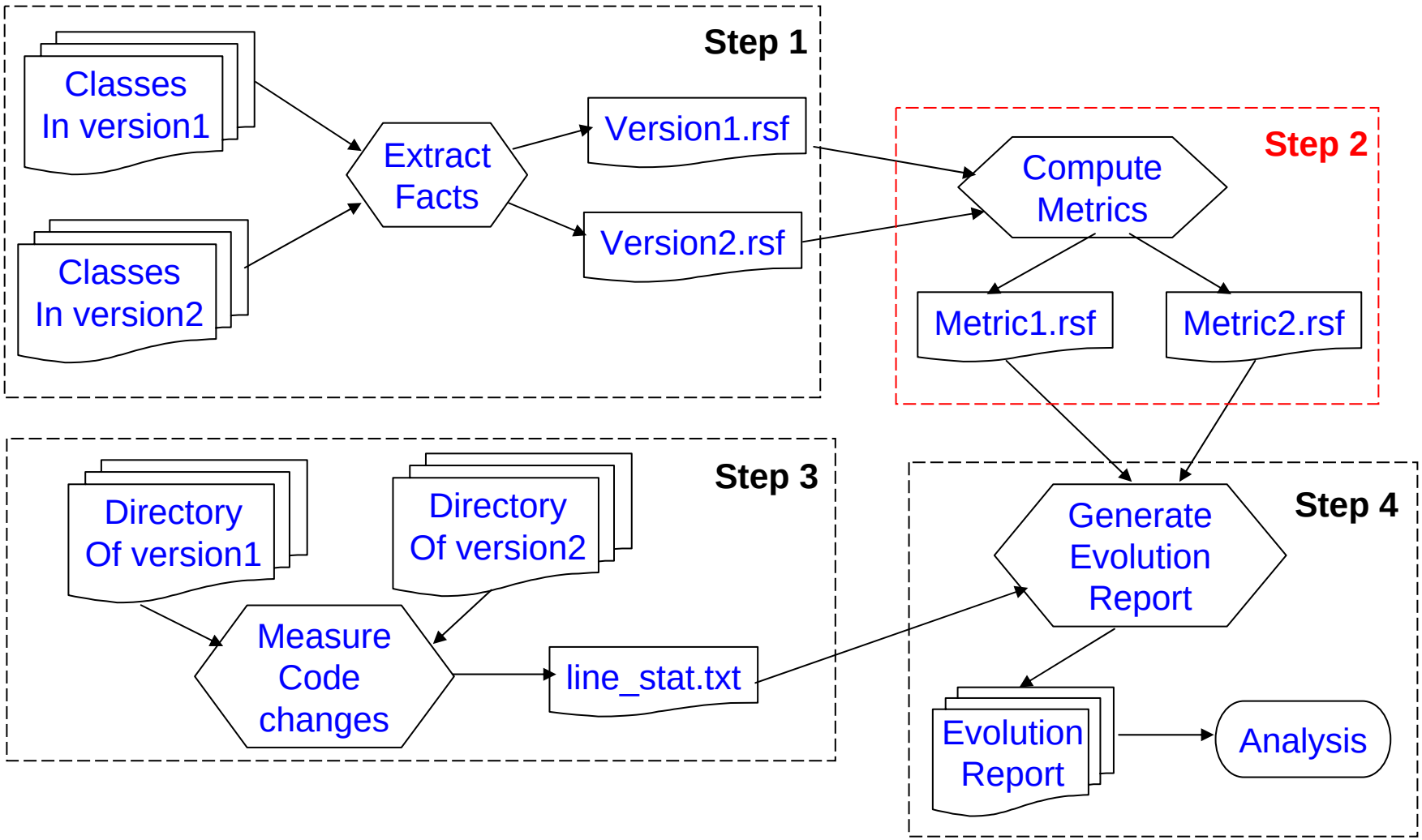
- RSF format is understandable by many reverse engineering tools
- RSF format is generated by many parsers of different languages
- To compute metrics from a common format than a specific one

M-Track for Java: Extraction Phase

Converting To RSF Format



M-Track:Tool Framework



M-Track: Metrics Computation Phase

- Each metric is implemented as a Perl subroutine by using our Perl package for querying the RSF

- Metric for each version is stored again in RSF format for further analysis

An Example

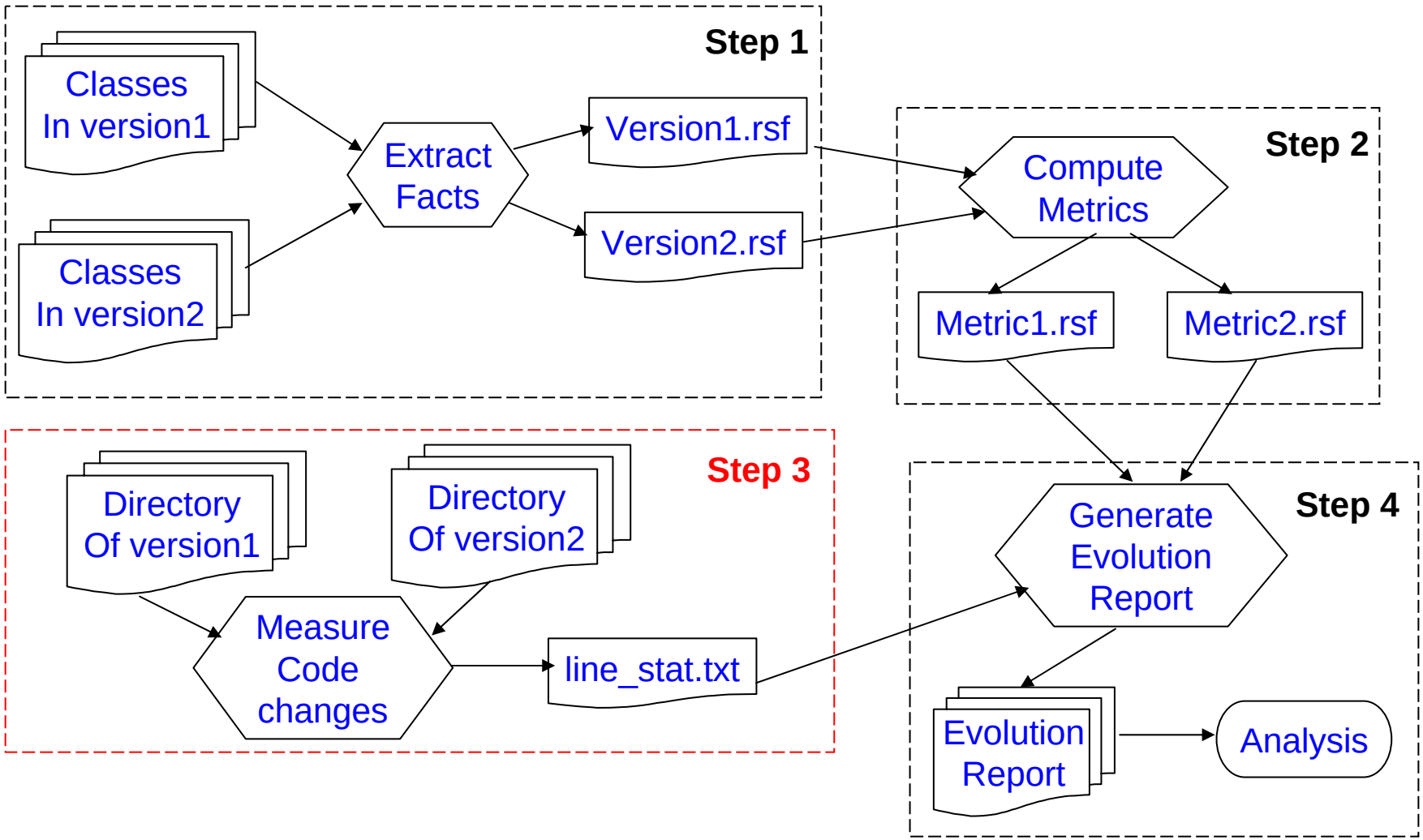
cbo de.marketmaker.utils.ByteString 45
nmimp de.marketmaker.utils.ByteString 15
nminh de.marketmaker.utils.ByteString 10

M-Track: Metrics Computation Phase

Performance in Metrics Computation Phase

System	Number of library classes	Number of application classes	Total Number of classes	Time (in sec)
Fusion-03-09	312	420	712	59
Merger-03-11	426	1320	1746	339

M-Track:Tool Framework



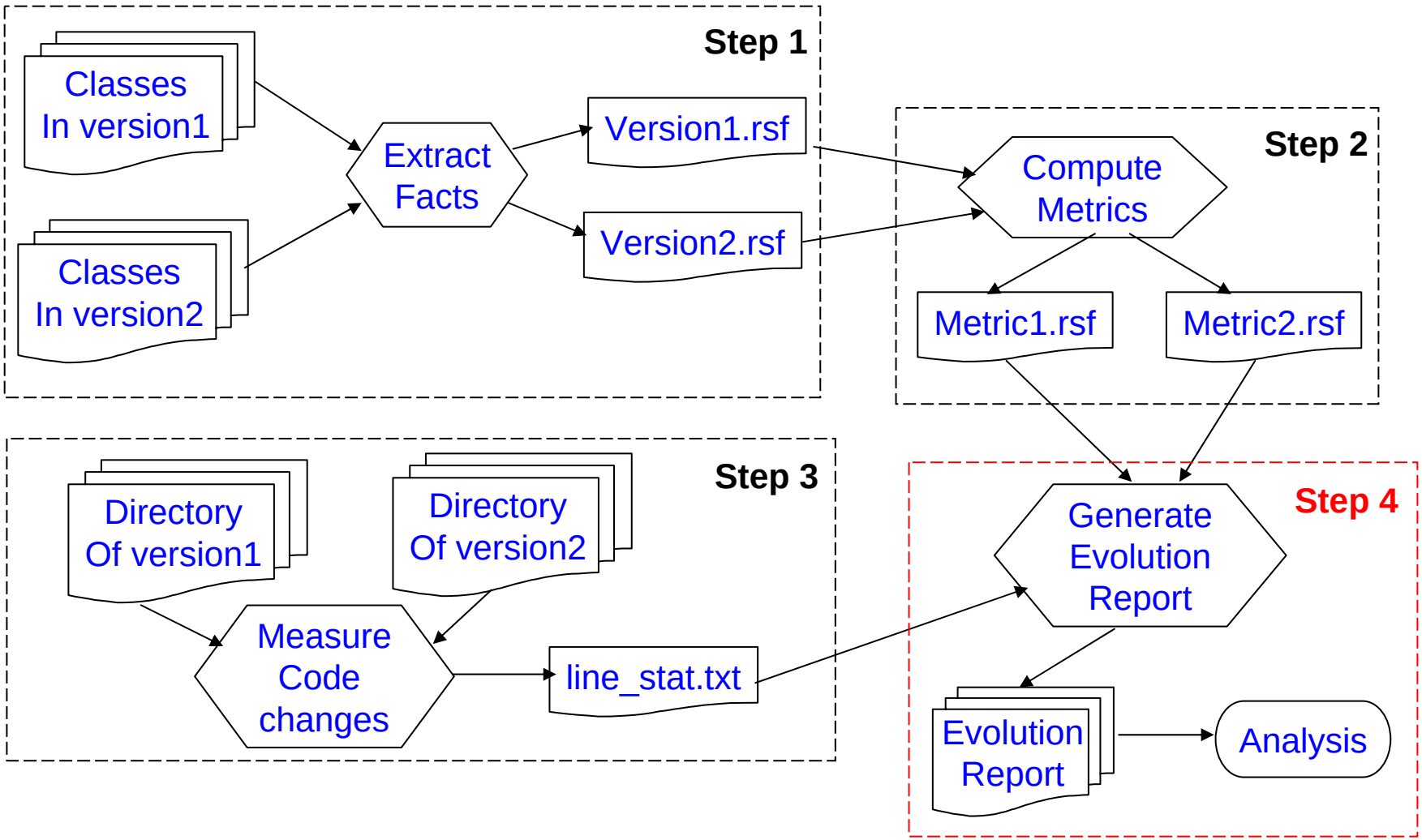
M-Track for Java:Extraction Phase

Measure the difference between two versions

- For every file, calculate number of lines added, changed, deleted, modified

Java_file	Tot_modified	Added	Changed	Deleted	Tot_Loc	%modif	Package
Employee	11	0	11	0	236	4.7	iese.spl
Student	50	50	0	0	50	100	iese.spl.student

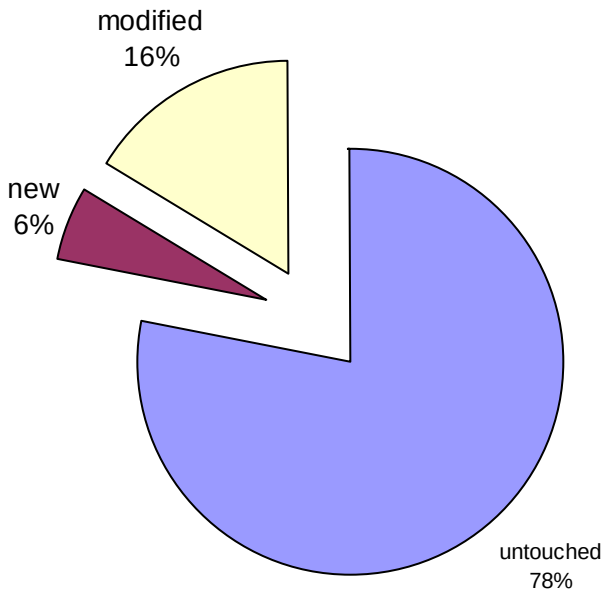
M-Track:Tool Framework



M-Track: Presentation Phase

- For every metric, we presented top candidate classes based on metric value
- For every metric, we presented top candidate classes based on high change in metric value
- We presented number of classes which are changed, unchanged, newly introduced between releases
- Microsoft Excel and R is used in this phase

M-Track: Presentation Phase



M-Track: Lessons Learned

- Know the problems of fact extraction tools to avoid surprises in the end !
- Compute metrics from a common format then a specific one
- Store the intermediate results for analysis (For e.g., Testing)
- Keep presentation phase flexible to introduce new criteria's for analysing the evolution

M-Track: Conclusion and Future Work

- Generic tool framework is presented for monitoring the evolution of OO systems
- Our tool framework is scalable till now
- In future:
  - Extending the tool to support dynamic metrics
  - Studying the relationship between static and dynamic measures
  - Identifying good visualization mechanism