Pietro Liguori

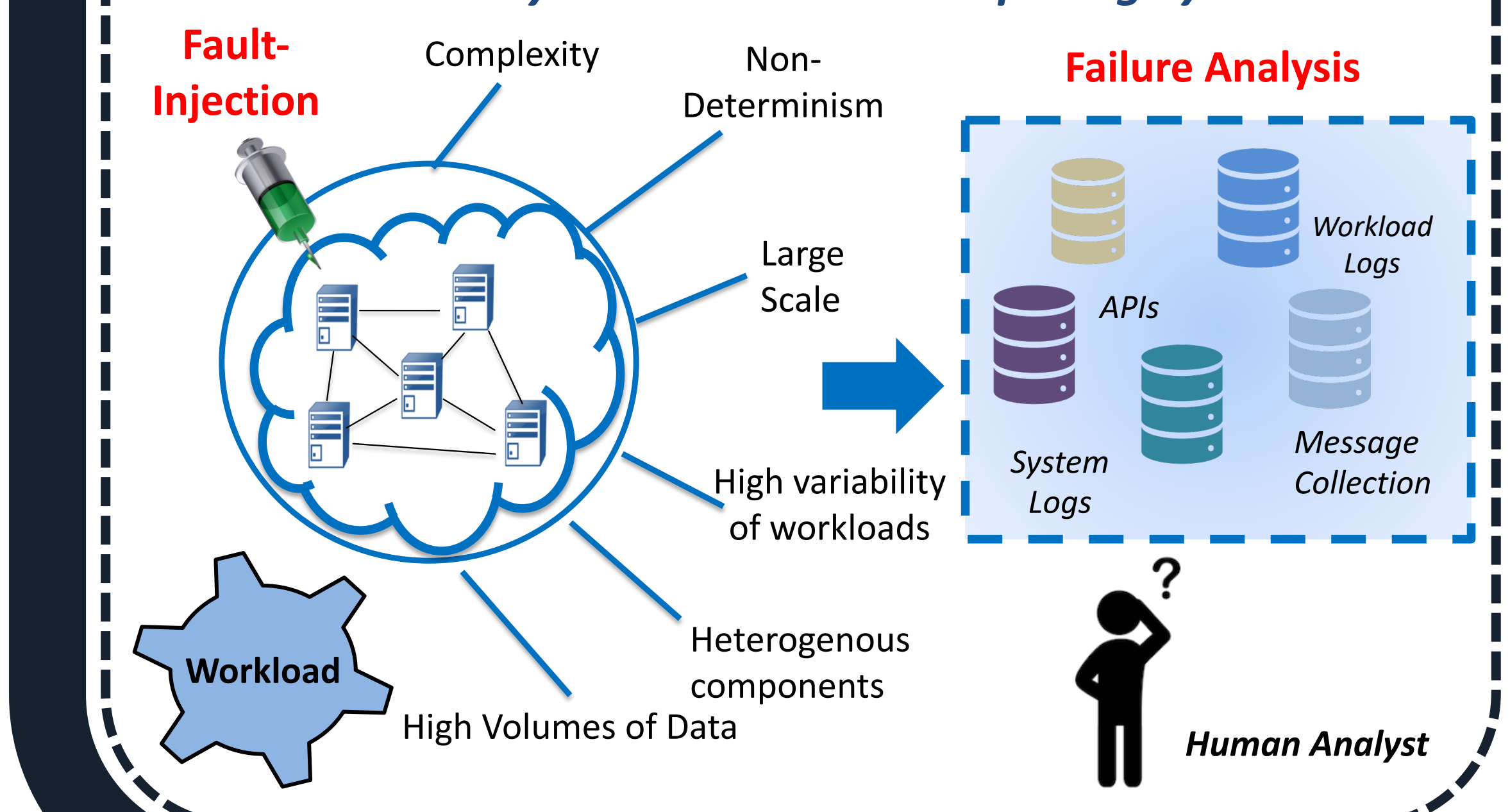
Tutor: Prof. D. Cotroneo – co-Tutor: Prof. R. Natella

XXXIV Cycle - II year presentation

Failure Mode Analysis in Cloud Computing Infrastructures

Research Context

Failure Analysis in the Cloud Computing Systems



Motivation

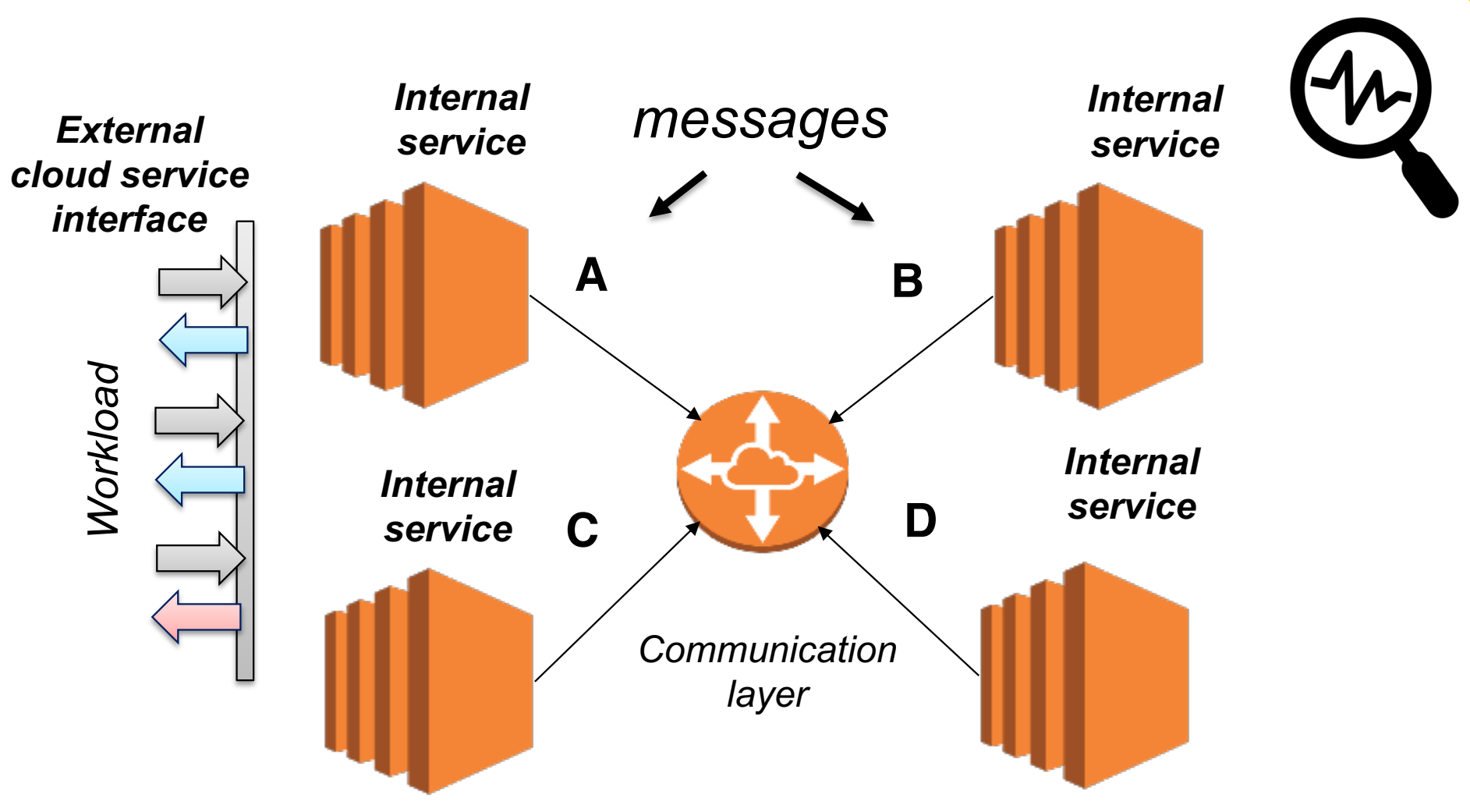
- **Distributed systems often fail in complex and unexpected ways**
  - *Combinations of **events** and **interactions** between component can not be anticipated by the system designers*
- **Fault-Injection is an effective means to analyze failures of distributed systems**
  - *Failure specifications are **written before the experiments***
  - *Specifications are not meant for discovering **unknown failure modes***
  - *Writing failure specifications is a **time-consuming** and **cumbersome task***

Needs

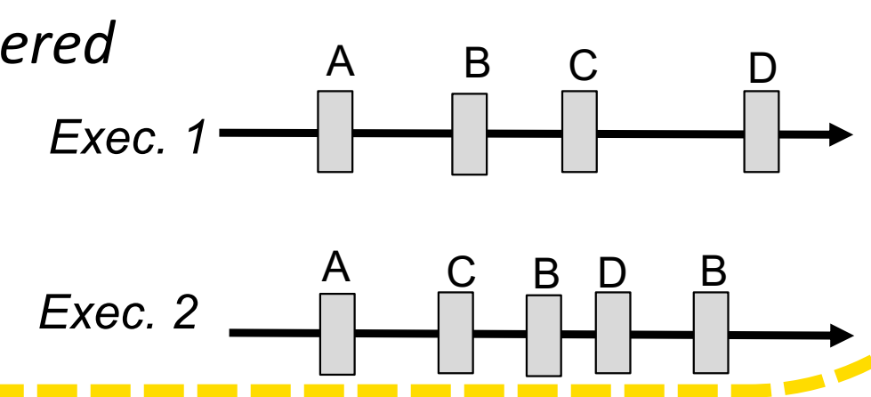
- **Failure Mode Analysis:** Getting insights on **how the distributed system can fail** is valuable for system designers to plan informed strategies for failure detection, diagnosis, and recovery.
- **Fault-Injection** needs to be supported by **techniques for discovering new, unknown failure modes** from the experimental data, which could otherwise be missed by a manual approach.

We need to develop **tools** and **methodologies** to **analyze** and **classify fault-injection experiments**

Runtime Monitoring

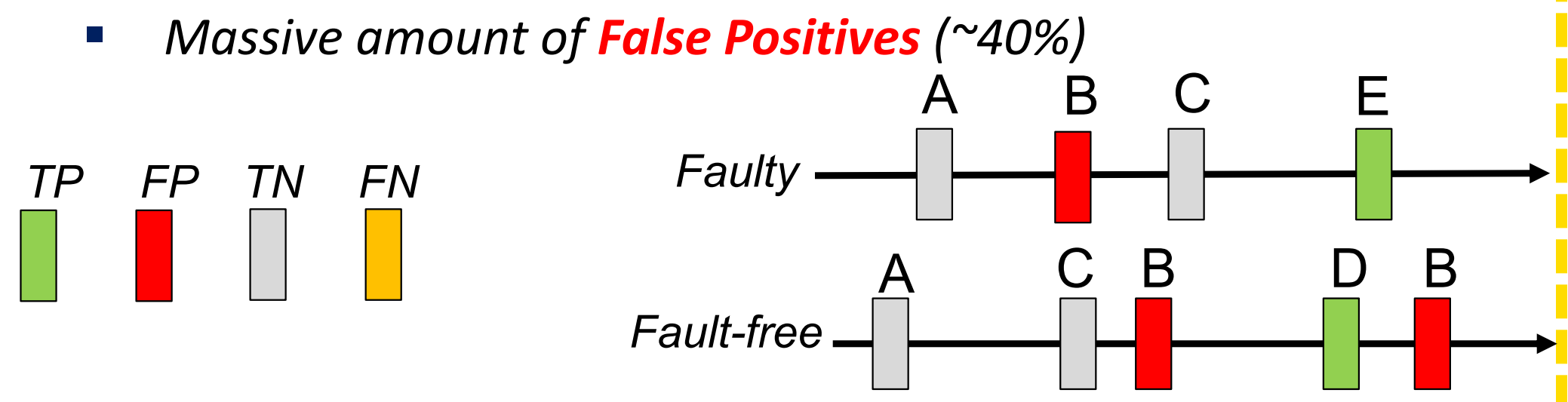


- **Black box Tracing**
  - *Low intrusiveness*
  - *API calls instrumentation (REST, Message queueing)*
  - *Essential information collected (message sender, service invoked, duration, timestamp, etc.)*
- **System Execution → Sequence of messages**
  - *Messages temporally ordered*

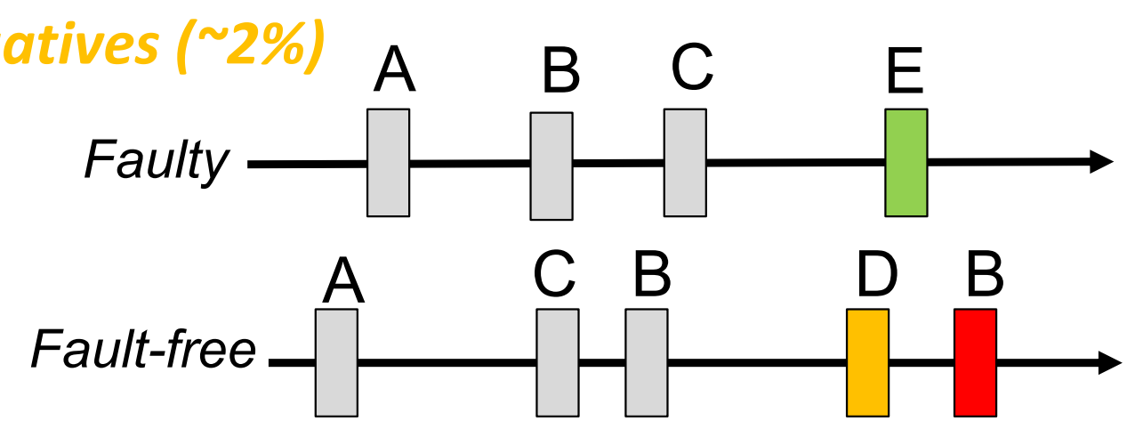


Detection of Anomalies

- **Modeling the normal (fault-free) behavior of the system**
- **Compare faulty traces with the normal behavior of the system to identify the anomalies**
  - *Sequence comparison (e.g., LCS)*
  - *Massive amount of **False Positives** (~40%)*



- **Probabilistic modeling on the anomalies**
  - *Reducing **False Positives (~22%)***
  - *Limiting **False Negatives (~2%)***



D. Cotroneo, L. De Simone, P. Liguori, R. Natella and N. Bidokhti "**Enhancing Failure Propagation Analysis in Cloud Computing Systems,**» *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, Berlin, Germany, 2019, pp. 139-150

Failure Mode Analysis

- **Applying unsupervised machine learning to group the experiments by their failure modes**
  - ***Experiments** represented as vectors of features*
  - ***Features** represent message (event) types*
  - ***Values** represent the number of anomalies for event type*
  - ***Clustering** applied on all the experiments*

Ground Truth

Failure Modes	# of Exp.
Instance Failure	224
Volume Failure	151
Network Failure	52
SSH Failure	41
Cleanup Failure	69
No Failure	539

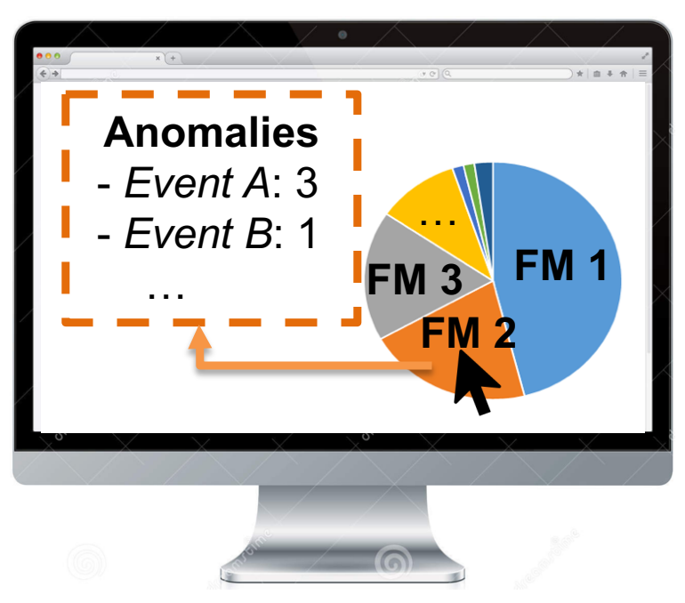
***Internal Evaluation**: How many failure modes?* Suggested number of cluster similar or equal to the failure classes of the ground truth (*Silhouette Index*)

***External Evaluation**: How accurate is the classification? Purity* is ~0.94

D. Cotroneo, L. De Simone, P. Liguori and R. Natella, "**Fault Injection Analytics: A Novel Approach to Discover Failure Modes in Cloud-Computing Systems,**" in *IEEE Transactions on Dependable and Secure Computing,* September 2020

Visualization

- **Visualization tool to help the human analyst**
  - *A **tool** for visualizing fault injection experiments, which points out relevant events for interpreting the failures*
  - *A **dashboard** showing the failure modes for one of the fault injection campaigns*



D. Cotroneo, L. De Simone, P. Liguori, R. Natella and N. Bidokhti, "**FailViz: A Tool for Visualizing Fault Injection Experiments in Distributed Systems,**" *2019 15th European Dependable Computing Conference (EDCC)*, Naples, Italy, 2019, pp. 145-148

Experimental Evaluation through Fault-Injection

- **Empirical analysis of cloud computing infrastructures by using the software fault-injection**
  - *Fail-stop behaviour*
  - *Failure Logging*
  - *Failure Propagation*

openstack

D. Cotroneo, L. De Simone, P. Liguori, R. Natella and N. Bidokhti, "**How bad can a bug get? an empirical analysis of software failures in the OpenStack cloud computing platform** ,". In *Proceedings of the 2019 27th ACM Joint Meeting on ESEC/FSE , Tallinn, Estonia, 2019* (pp. 200-211).

D. Cotroneo, L. De Simone, P. Liguori and R. Natella, "**ProFIPy: Programmable Software Fault Injection as-a-Service,**" *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Valencia, Spain, 2020, pp. 364-372

My Research group

I am a member of the **Dependable and Secure Software Engineering Real-Time Systems (DESSERT)**, at DIETI – UNINA.

http://www.dessert.unina.it

University Cooperation

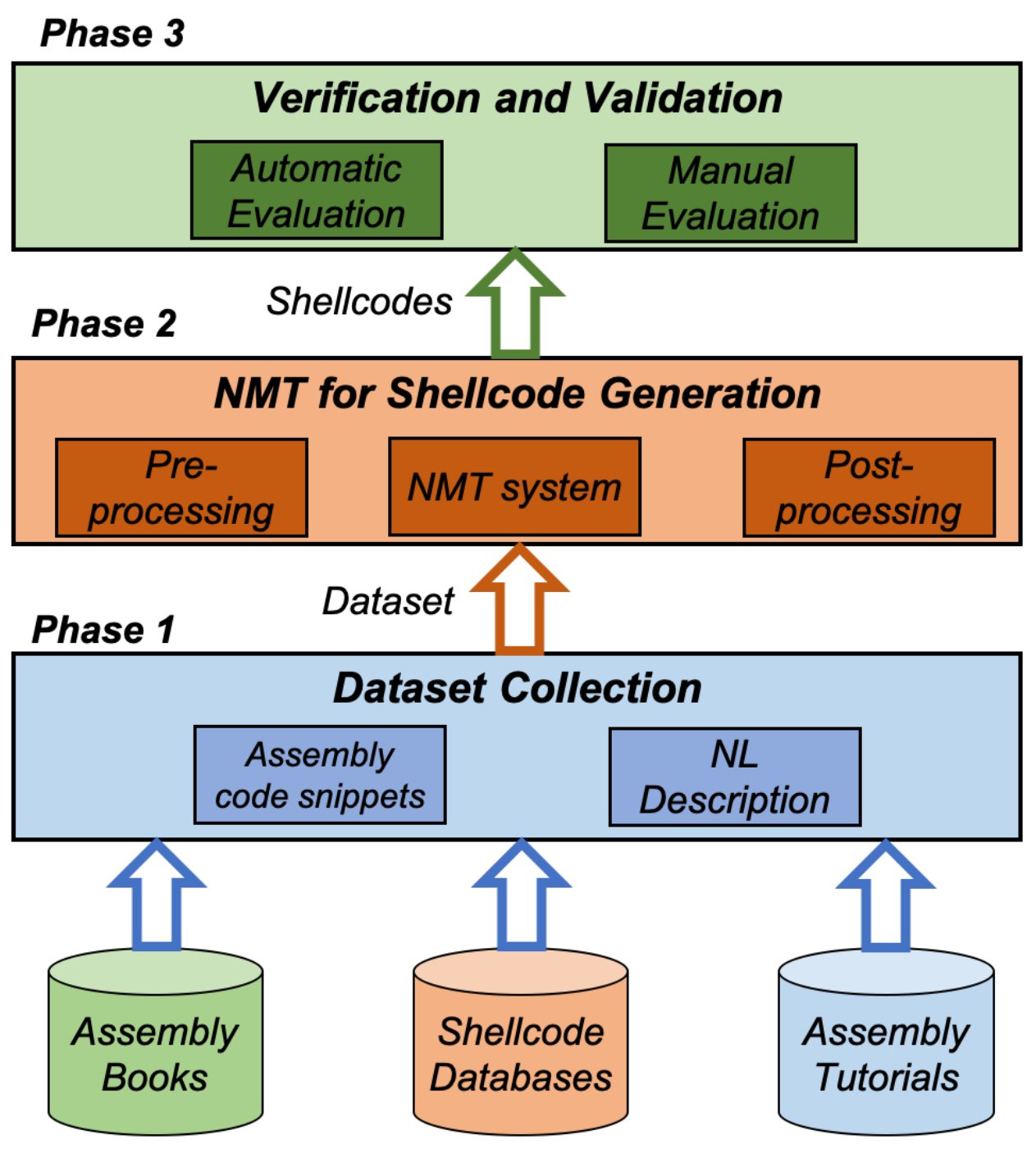
I am collaborating with the **University of North Carolina at Charlotte** (**UNCC**), North Carolina, USA, on a new research topic. This research focuses on the automatic generation of software exploits by using the Neural Machine Translation (NMT) techniques.

Ongoing and Future Activities

- **Runtime Verification (RV) via stream processing in the cloud computing infrastructures**
  - *Building a set of lightweight **monitoring rules** from correct executions of the system in order to specify the desired system behavior*
  - *Synthesis of the rules in a **runtime monitor** that verifies whether the system's behavior follows the desired one*

D. Cotroneo, L. De Simone, P. Liguori, R. Natella and A. Scibelli "**Towards Runtime Verification via Event Stream Processing in Cloud Computing Infrastructures**", *International Workshop on Artificial Intelligence for IT Operations*, December 2020

- **Automatic Software Exploit Generation**
  - *Kickstarting a new research topic in collaboration with the University of North Carolina at Charlotte (UNCC)*
  - *Generation of **software exploits** (e.g., **shellcodes)** from natural language*
  - *Leveraging the **Neural Machine Translation** techniques*



Phase 3
Verification and Validation
Automatic Evaluation
Manual Evaluation
Shellcodes
Phase 2
NMT for Shellcode Generation
Pre-processing
NMT system
Post-processing
Dataset
Phase 1
Dataset Collection
Assembly code snippets
NL Description
Assembly Books
Shellcode Databases
Assembly Tutorials