MESSAGE-ABSTRACTOR

Sergey Panov
xpanov00@stud.fit.vutbr.cz
Faculty of Information Technology, Brno University of Technology

Motivation

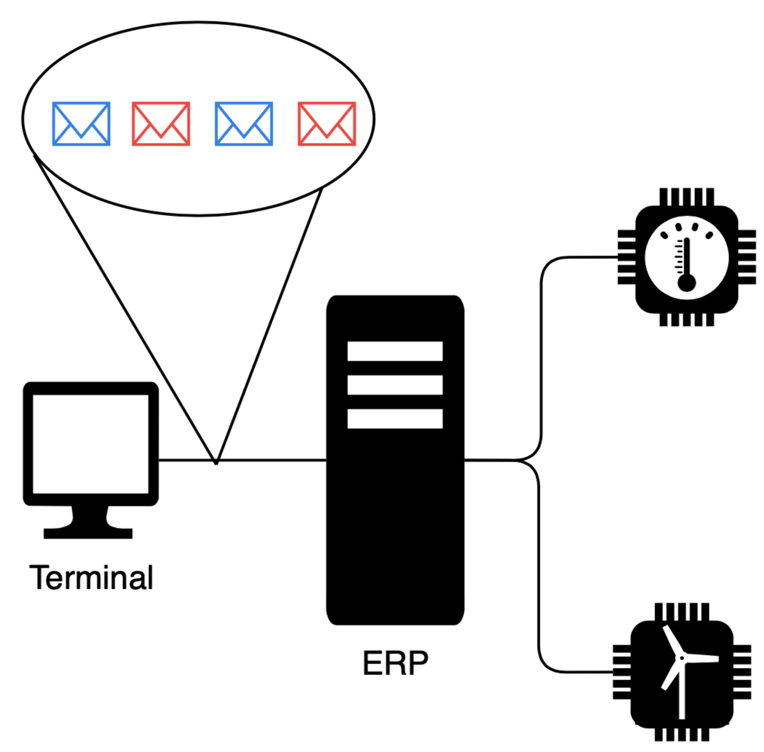
The world is full of distributed **multi-component** computer **systems** used in many different industries, e.g., power stations, pumping stations, and others. These systems are **hard to test**, but assuring their quality is crucial since one bug (e.g., in IoT systems used in the industry) can lead to high losses. The testing is difficult because

- systems are distributed, asynchronous, and uses complicated communication protocols with different formats of messages,
- components in these systems are often expensive machines used in factories, and it is impossible to use them for testing purposes.

Therefore this work **proposes** a method of **testing** the **systems** in a virtual environment by **creating a digital twin** of the tested system. The digital twin simulates the behavior of the real system, so we can stress it by generating different kinds of inputs or test a new version of one of the system components to verify that no bug was introduced during the upgrade.

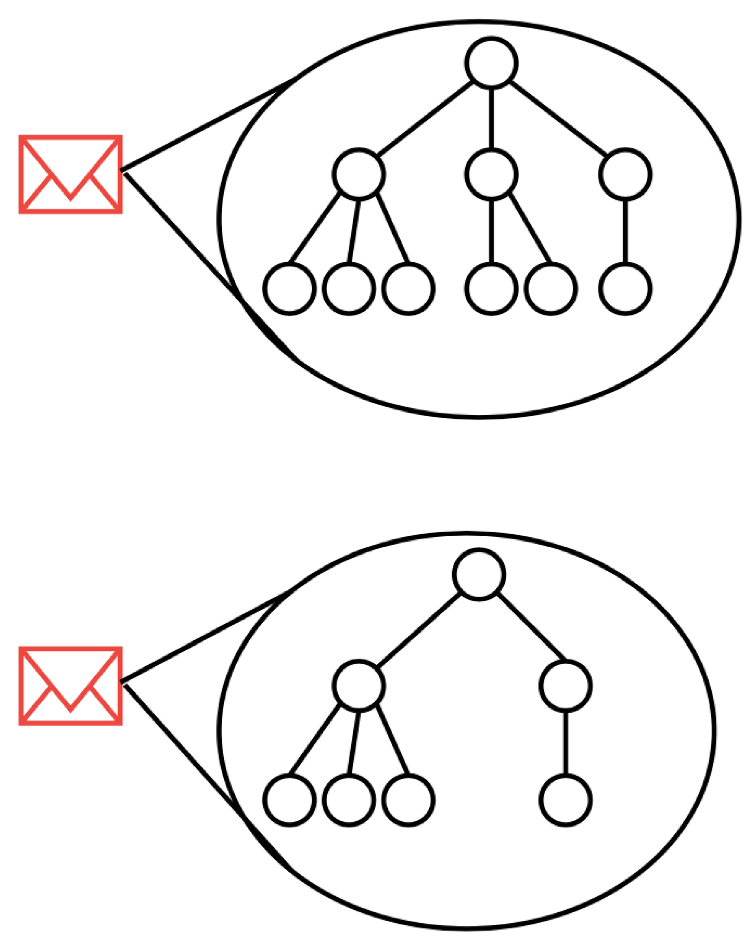
For creating the digital twin, it needs to **automatically learn** the **protocols** and the **messages** used by the real system. In my work, I focused on learning and representing messages.

Simple Network



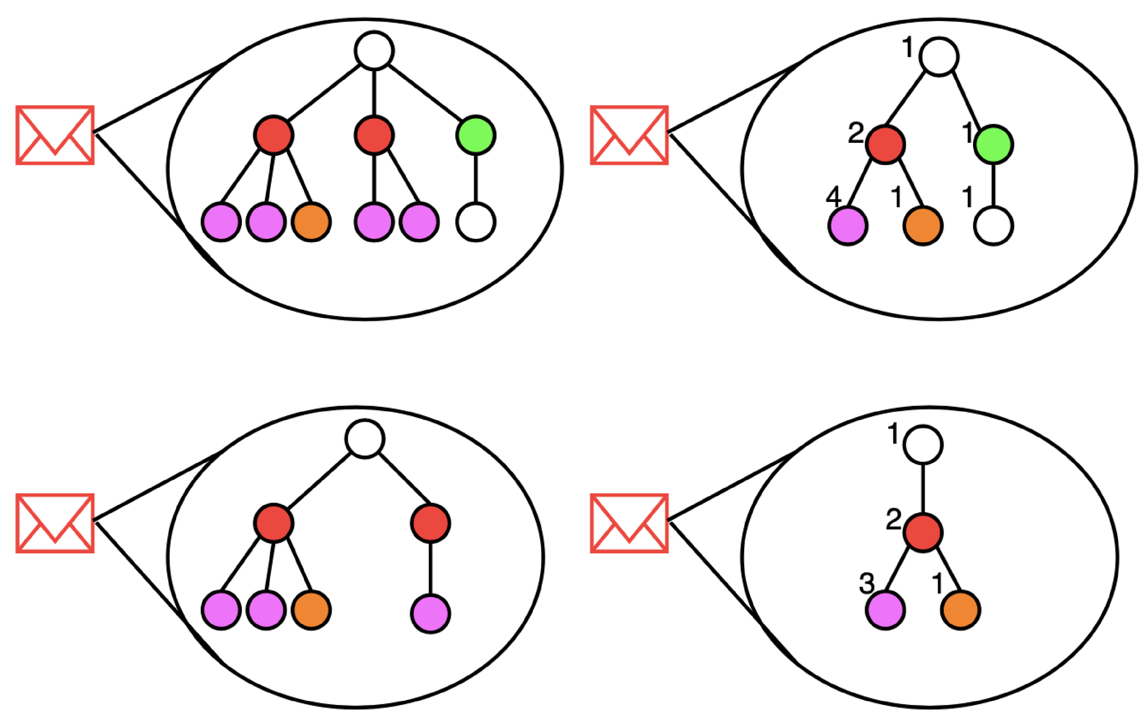
Let there be a **terminal** that **communicates with ERP** (Enterprise Resource Planning) system. The ERP interacts with multiple sensors distributed in a factory. To provide information about different sensors, the ERP sends the **messages** of **different types** (the message reporting pressure differs from the message reporting temperature). The messages can be in **different formats**, e.g., XML, JSON, that can be logically represented as tree structures.

Messages Representation



The messages of the **same type** report different states of sensors, so they are represented by **trees with different shapes**. We take recorded **messages** and create the **model of them**. This model is used to represent messages in a digital twin.

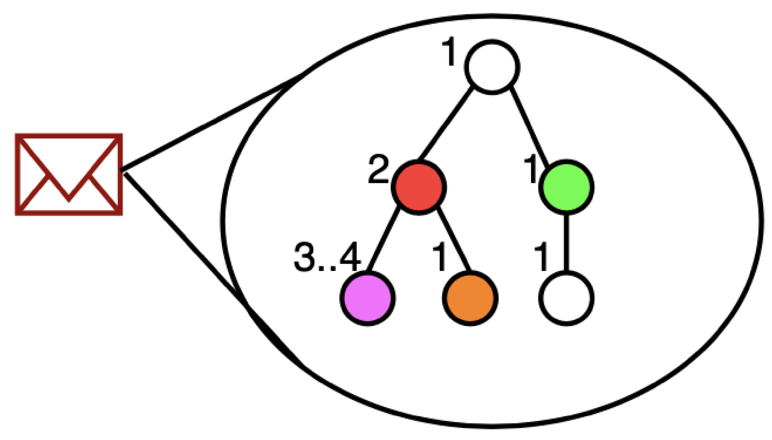
Messages Processing



To make the **model compact**, we **abstract** some **nodes** without losing the structure of the message nor the content.

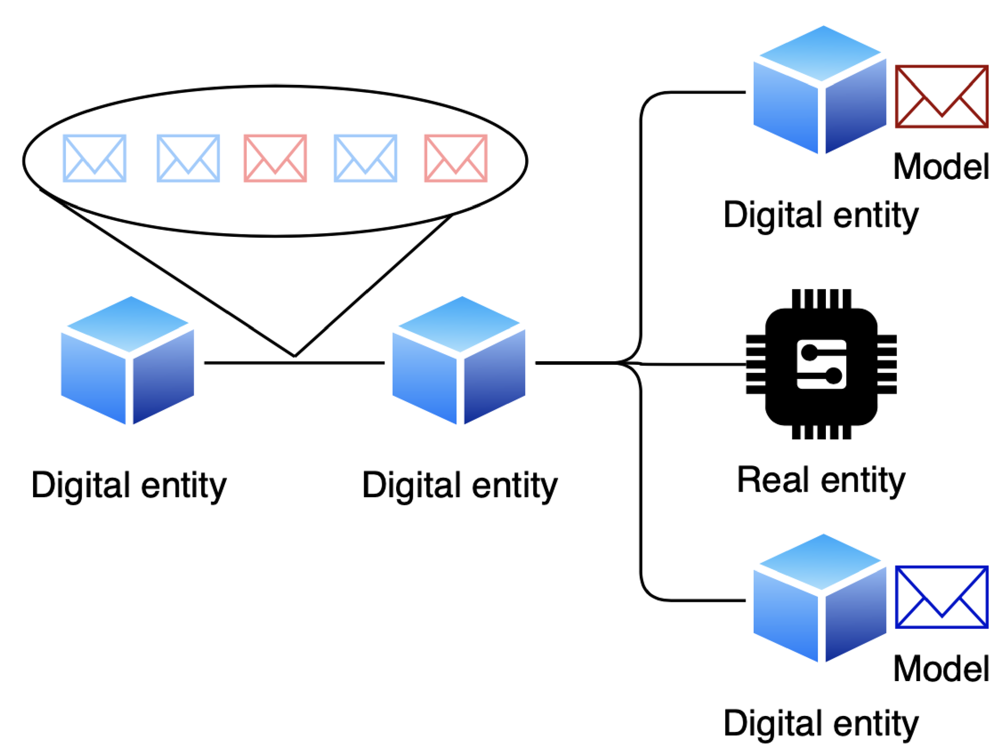
Particularly, we **group** the **nodes** according to a given **equivalence criterion** and **reduce** them to a **single node** representing the group. **While reducing**, we **store information** about the number of reduced nodes and other meta-information.

Model Creation



Having the compact **representations of messages**, we **merge** them into one **abstract-tree** that contains all relevant meta-information about each tree-like message. This structure we **use as a model** and **create messages similar to real ones** from it.

Result Network



Once we have the compact **representations of messages**, we **merge** the messages of the same type into one **abstract-tree** containing all relevant meta-information about each tree-like message. The **abstract-tree** is later **used** in digital twin as some kind of **template** to **generate** different concrete **messages** of the given type to stress the system under testing.
Placing the **real component** into the **network**, we **observe** its **behavior** to make sure it works correctly. Moreover, we can **extrapolate** the **behavior** the **digital components** and **observe** the behavior of the **real component** in **extreme conditions**.

Experiments With the Model

Message	1_432	2_425	3_432	4_432
The nodes of the model been used	83	0	13	54
Number of nodes of the tree-like message	673	29	119	798
Covered nodes of the tree-like message	673	0	99	798
Accuracy	$\frac{673}{673} = 1$	0	$\frac{99}{119} = 0.83$	$\frac{798}{798} = 1$
Structure-overhead	$1 - \frac{83}{225} \approx 0.63$	1	$1 - \frac{13}{225} \approx 0.94$	$1 - \frac{54}{225} = 0.76$
Expressiveness	$1 - \frac{83}{673} \approx 0.88$	0	$1 - \frac{13}{99} \approx 0.87$	$1 - \frac{54}{798} \approx 0.93$

Fig. 6: The experiments with the model with the 225 nodes.

Acknowledgment

This work was supported by National Cybersecurity Competence Centre, by TAČR project no. TN01000077/06. I would like to thank my supervisor Aleš Smrčka and colleagues Tomáš Fiedor and Martin Hruška from VeriFIT research group for weekly consultations and discussions.