TOWARDS MODEL CHECKING OF THE INTERNET OF THINGS SOLUTIONS INTEROPERABILITY

Presented by Ahmed Jagmagji

Supervised by Dr. Istvan Majzik

Budapest University of Technology and Economics

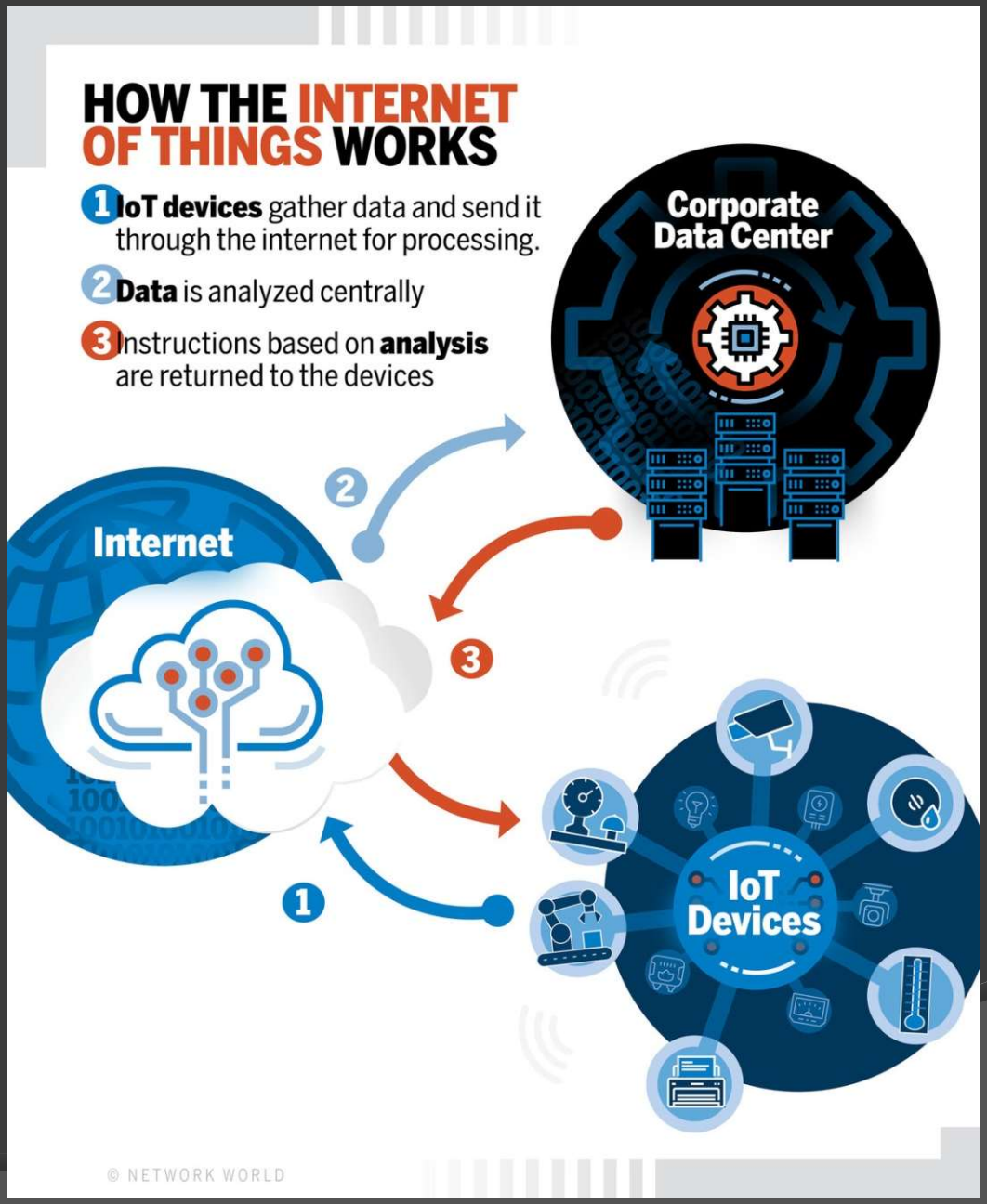
Outline

- What is Internet of Things (IoT) and why it is important?
- How the Internet of Things Works?
- Model Checking of the Internet of Things Solutions Interoperability : Introduction
- Problem Statement
- What is the TLA+
- The Approach and Case study
- TLA+ Tools (TLC Model Checker)
- PlusCal Code Example
- Results Analysis
- Conclusion and Future Work

What is the Internet of Things (IoT) and why it is important?

- The Internet of Things (IoT) describes the network of physical objects—"things"—that are embedded with sensors, software, for the purpose of connecting and exchanging data with other devices over the internet.
- We can connect everyday objects—kitchen appliances, cars, thermostats, baby monitors—to the internet via embedded devices.
- Seamless communication is possible between people, processes, and things.
- In this hyper-connected world, digital systems can record, monitor, and adjust each interaction between connected things.
- The physical world meets the digital world—and they cooperate [1].

How the Internet of Things Works?



https://www.networkworld.com/article/3207535/what-is-iot-the-internet-of-things-explained.html [2]

Model Checking of the Internet of Things Solutions Interoperability

Introduction

- IoT Interoperability : The ability for multiple IoT platforms from different vendors to work together without critical issues such as vendor lock-in, impossibility to develop IoT application etc [3].
- Both the diversity of "smart" devices and the difference of Internet protocols make it a challenging task to achieve the true interoperability.
- Because of the IoT Interoperability challenges, the approach to the Internet of Things solutions consistency checking has been proposed
- The IoT devices are considered to be highly heterogeneous in terms of communication protocols, data formats and technologies.

Problem Statement

- The IoT system is considered on Service Enablement Layer– as a compound of communicating web services.
- These services are treated as the components of system.
- The components are represented with a set of state variables.
- The authors have considered two approaches to TLC method usage: the breadth-first search (BFS) of transition system states and the depth-first search (DFS).

$$V = \{v_i\}_{i=1}^{m \in N}$$

- The values of $v_i \in V$ are represented with $d_1 = 0,\ d_2 = 1$

The $d_1 \in D$ is interpreted as Boolean "False", $d_2 \in D$ is "True"

Problem Statement

- The elements of AP set are classified to two groups:
- $(v_i, 0) \in AP$ – i -th component hasn't yet been invoked – involved in computational process (communication between smart devices).
- $(v_i, 1) \in AP$ – i -th component has already been invoked.
- The Kripke structure has been chosen as a transition system model:

$$M = \langle S, \{s_0\}, R, L \rangle,$$

where S – finite set of states, $s_0 \in S$ – initial state, $R \subseteq S^2$ – set of transitions,

$L : S \to 2^{AP}$ – states labeling function, AP – atomic prepositions set.

Problem Statement

$$AP = V \times D$$

$$V = \{v_i | i = 1,2,\ldots,n\} - \text{state variables set}$$

- $n = |V|$ is the amount of atomic web services
- $D = \{0,1,2\}$ set of state variables values [4].
- Each state variable represents corresponding atomic web service. To prove that the components of system are consistent, the following expression should be satisfied $\forall s \in S$:

$$M, s \models \varphi,$$

- Where φ is the resulting temporal formula to be checked $\forall s \in S$ Checking liveness properties, if no deadlocks have been reached, the IoT solution can be characterized as the consistent one.

What is the TLA+

- A language for modeling software above the code level and hardware above the circuit level.
- IDE (Integrated Development Environment) for writing models and running tools then check them.
- TLA+ is based on mathematics and does not resemble any programming language.
- TLA+ models are usually called specifications.
- An event is represented by a pair of consecutive states.
- A sequence of states called a behavior.
- A pair of consecutive states called a step rather than an event.

- TLA+ describes a set of behaviors with two things:

1- An initial condition that specifies the possible starting states.

2- A next-state relation that specifies the possible steps (pairs of successive states) [5].

The Approach and Case Study

- Temporal Logic of Actions (TLA) and the appropriate TLA+ formalism have been used to create the formal model of IoT system.

- To describe the proposed approach, the smart home scenario has been planned from the web services viewpoint:

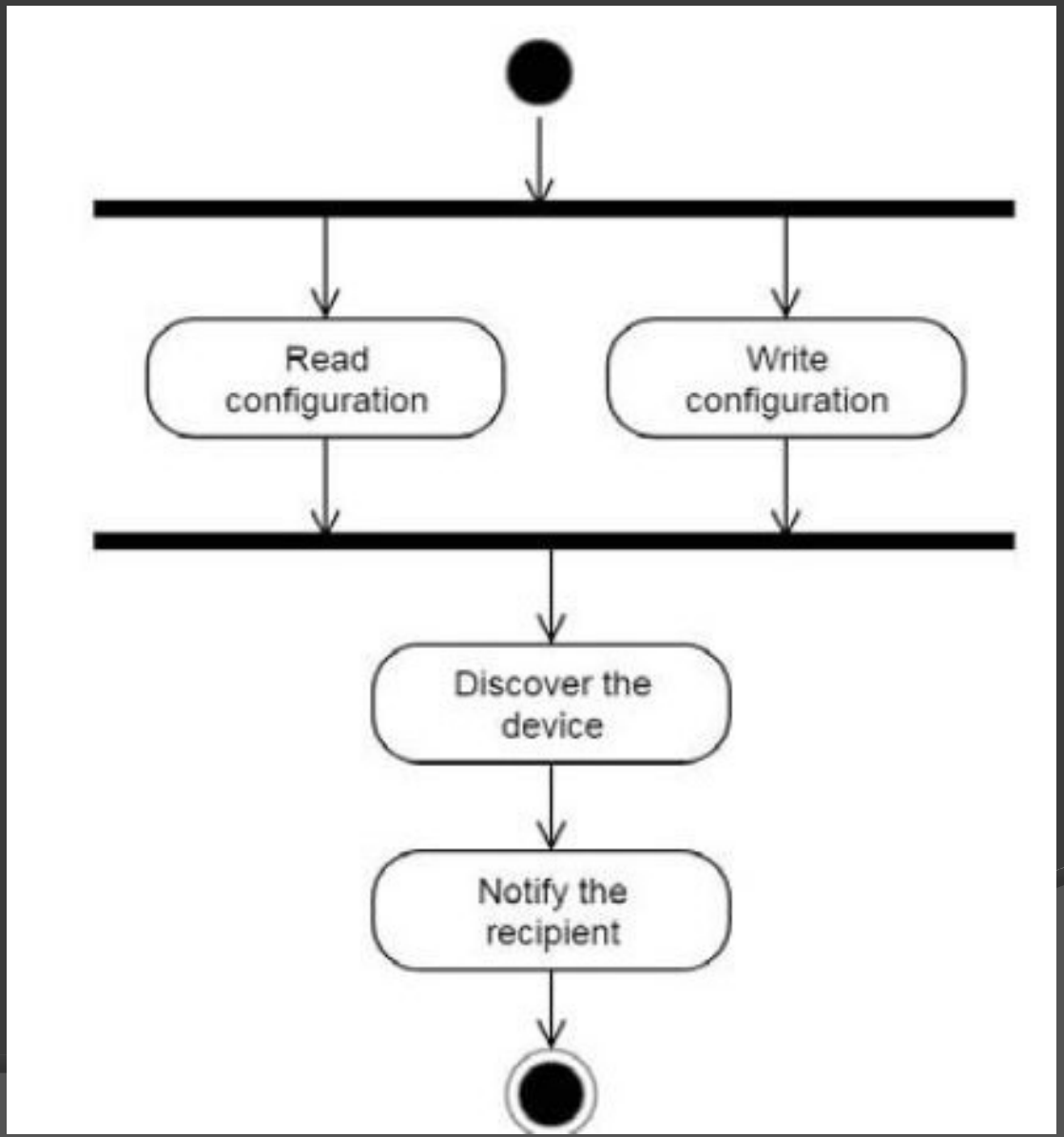
1. **First step** – conduct the management of system configuration with respect to certain task (e.g., automatically open garage door on time). The initial step, either the "read configuration" or "write configuration" web service should be invoked.

2. **Second ste**p – conduct the discovery of required devices components.

3. **Third (final) step** – notify the recipient about the results of discovery to make the decision about the scenario.

The Approach and Case Study

- The activity diagram for the described sequence of steps is given in Fig. 1.



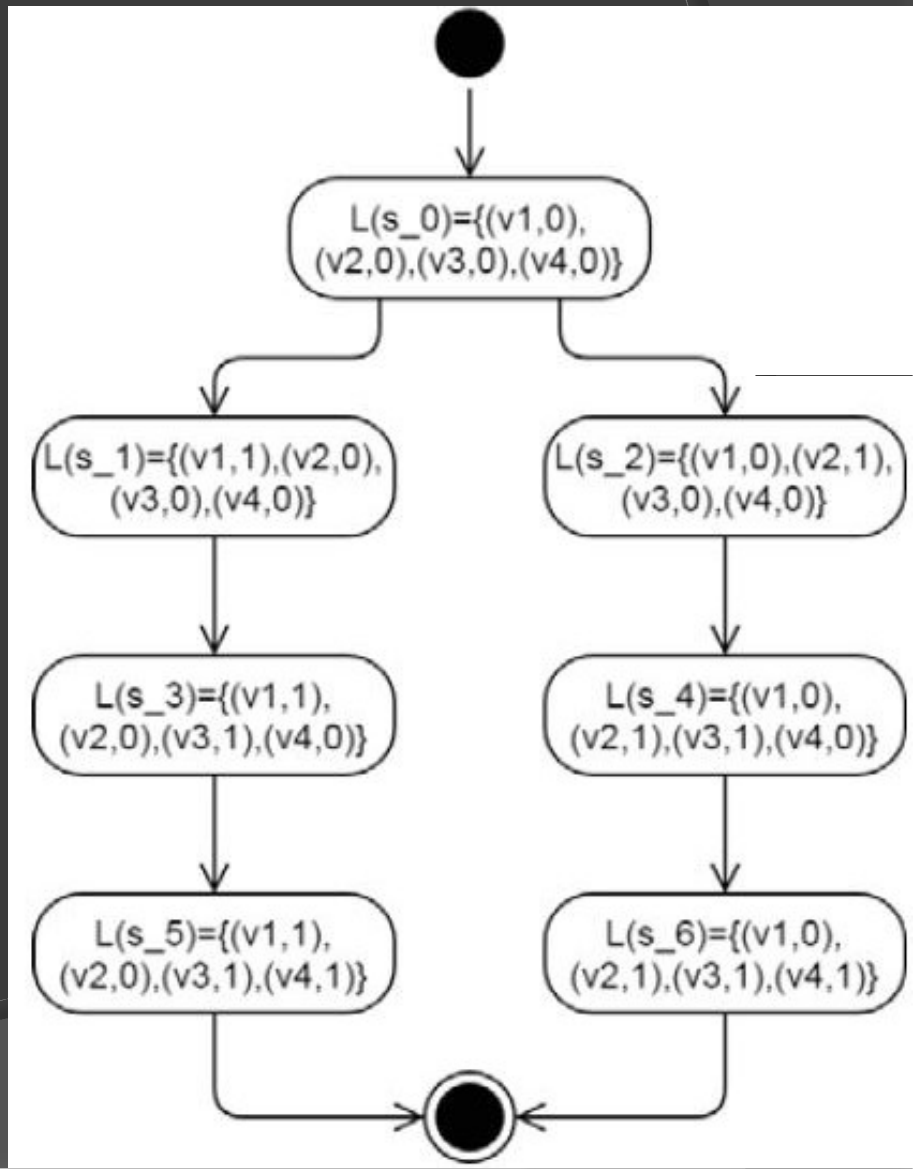
The Approach and Case Study

- Kripke structure (2) has been created to represent formal model for the activity diagram, given in Fig. 1.
- The V set has been filled first:
- $V = \{v_1, v_2, v_3, v_4\}$, where $v_1 \in V$ represents the web service conducting the "Read configuration" activity, $v_2 \in V$ "Write configuration", $v_3 \in V$ "Discover the device", $v_4 \in V$ "Notify the recipient". Then the AP set (1) is as follows:

$$AP = \{(v_1,0),(v_1,1),\ldots,(v_4,0),(v_4,1)\}$$

The Approach and Case Study

- There are 7 distinct states that should be reached during the verification (Fig. 2).



- As it can be seen in Fig. 2, the depth of state space search is 4. This value should be assigned as a parameter to start the DFS-driven TLC verification.
- To get started with TLA+ specification creation, the initial state of formal model has to be specified first. For this purpose the elements $L(S_0)$ set have to be used:

$$L(s_0) = \{(v_1,0),\ldots,(v_4,0)\}.$$

The Approach and Case Study

- The resulting successfully verified formal model in TLA+ syntax is as follows:

- VARIABLES v1, v2, v3, v4 * state variables representing the web services (Fig. 1).
- Invariant == v1 \in BOOLEAN /\ v2 \in BOOLEAN /\ v3 \in BOOLEAN /\ v4 \in BOOLEAN * setting up the allowable values of state variables.
- S_0 == v1 = FALSE /\ v2 = FALSE /\ v3 = FALSE /\ v4 = FALSE * specify $L(s_0)$.
- S_1 == v1 = TRUE /\ v2 = FALSE /\ v3 = FALSE /\ v4 = FALSE * specify $L(s_1)$.
- S_2 == v1 = FALSE /\ v2 = TRUE /\ v3 = FALSE /\ v4 = FALSE * specify $L(s_2)$.
- S_3 == v1 = TRUE /\ v2 = FALSE /\ v3 = TRUE /\ v4 = FALSE * specify $L(s_3)$.
- S_4 == v1 = FALSE /\ v2 = TRUE /\ v3 = TRUE /\ v4 = FALSE * specify $L(s_4)$.

TLA+ Tools (TLC Model Checker)

- **SANY Syntactic Analyzer:**

  A parser and syntax checker for TLA+ specifications (catches parsing errors and some semantic errors).
- **TLC Model Checker:**

  A model checker and simulator for executable TLA+ specifications.
- **PlusCal Translator:**

  A translator from the PlusCal algorithm language to TLA+.
- **TLAPS Proof System**

  A system for mechanically checking proofs written in TLA+.
- **TLATeX Pretty-Printer**

  A program for typesetting TLA+ specifications [5].

PlusCal Code Example

- PlusCal is a simple programming language, except that an expression can be any TLA+ expression (any mathematical formula) [5].

```
--algorithm Peterson {
   variables flag = [i \in {0, 1} |-> FALSE],  turn = 0;
      \* Declares the global variables flag and turn and their initial values;
      \* flag is a 2-element array with initially flag[0] = flag[1] = FALSE.
   fair process (proc \in {0,1}) {
     \* Declares two processes with identifier self equal to 0 and 1.
     \* The keyword fair means that no process can stop forever if it can
     \* always take a step.
     a1: while (TRUE) {
           skip ;  \* the noncritical section
     a2:  flag[self] := TRUE ;
     a3:  turn := 1 - self ;
     a4:  await (flag[1-self] = FALSE) \/ (turn = self); \* \/ is written || in C.
     cs:  skip ;  \* the critical section
     a5:  flag[self] := FALSE                  }     }     }
```

PlusCal Code Example

- The TLC model checker can verify that the algorithm satisfies two important properties: mutual exclusion (two processes are never executing their critical section at the same time); and starvation freedom (each process keeps executing its critical section).

- The TLA+ translation introduces an array variable (**pc**) whose value indicates where each process is in its algorithm.

- pc[i] = "cs" means that process i is in its critical section (pc[0] and pc[1] are never both equal to "cs“)

- This condition can be written as:
  (pc[0] /= "cs") \/ (pc[1] /= "cs") [5].

Results Analysis

- The created formal model can be characterized as transparent, scalable and reconfigurable solution.
- TLA Toolbox utility implementing the TLC (TLC2 Version 2.05) has been used.
- No deadlocks have been reached.
- The formal model has been proved with TLC (7 distinct states found).
- The depth of state space is 4.
- The average value of BFS time costs is 1195 ms, while the average value of DFS time costs is 572 ms.
- The drawback though DFS-driven TLC verification which is that it can’t be conducted if the depth of state space search is unknown.
- BFS driven verification should be applied first to overcome the DFS drawback.

Conclusion and Future Work

- The approach to model checking of the IoT solutions interoperability on the basis of Temporal Logic of Actions has been proposed.
- Formal model of the IoT system with respect to smart home scenario has been created and successfully verified with TLC model checker – by way of DFS and BFS.
- Future work is aimed at combining the advantages of both BFS and DFS approaches.

Thanks for Your Listening

References:

1. Internet website https://www.oracle.com/internet-of-things/what-is-iot/
2. https://www.networkworld.com/article/3207535/what-is-iot-the-internet-of-things-explained.html
3. Noura, M., Atiquzzaman, M. & Gaedke, M. Interoperability in Internet of Things: Taxonomies and Open Challenges. Mobile Netw Appl 24, 796–809 (2019).
4. V. Viktorovych Shkarupylo, I. Tomicic, and K. Mykolaiovych Kasian, “The Investigation of TLC Model Checker Properties”, JIOS, vol. 40, no. 1, Jun. 2016.
5. https://lamport.azurewebsites.net/tla/tla.html
6. V. Shkarupylo, R. Kudermetov, T. Golub, O. Polska and M. Tiahunova, "Towards Model Checking of the Internet of Things Solutions Interoperability," 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 465-468. **(The Original Paper for Homework)**

Notice : The whole presentation slides information were taken from the reference numbered 1-6.