Using Educational IoT System

Alexander Provotar, Maksym Veres and Maksym Samoilenko

Taras Shevchenko National University of Kyiv, Glushkov ave., 4g, 03022, Kyiv, Ukraine.

Abstract
This paper investigates using educational IoT in group of students. There is formal investigation of IoT system approach presented as transition systems and their composition which presented as Petri net with next liveness checking. Paper describes architecture of IoT system which is implementation of Petri net.

Keywords [1]
educational IoT, web – server, transition system, Petri net

1. Introduction

Nowadays IoT devices is popular and can be used as IoT for mobile phones, smart homes, automotive technologies. There is a good opportunity for using IoT in education curriculum. This paper exposes using IoT technologies in education, especially model of interaction between IoT and end-users, and realization of model by using system architecture.

This paper has a contribution: model of interaction between IoT and end-users which exposed as Petri net with liveness checking which is result of multiplication transition systems of each aspect of interaction and system architecture of implementation of Petri net.

Section II provides a literature review oof using educational IoT. Section III introduces using IoT in curriculum and described initial information about IoT system. Section IV provides transition system and Petri net definitions, investigates liveness of Petri net presented as multiplication of transition systems. Section V presents system architecture of Petri net implementation. Section VI describes conclusions and Section VII – references.

2. Literature review

There are plenty examples of using IoT in education curriculum with Lego Mindstorm. There is Robolab programming tool for Lego Mindstorm analysis of using IoT for learning programming languages by students which take a part in Lego IoT building blocks, decomposing and investigation of educational task given, running program [1]. Working with education IoT described step by step in system engineering students by experiencing software engineering lifecycle from requirements and design to system implementations by means of Lego Mindstorm [2]. There is an example of using maker made IoT in educational process described where this IoT can be integrated in school curriculum with strong acceptance by the school community [3]. According to papers above, educational IoT has a good possibility to be used in curriculum. Also, there is paper with interaction of group of IoT with different relationship types and with system architecture of connection protocols discussion and analysis presented [4].

Different interaction between system architecture aspects can be presented by Petri net. Current paper describes model interaction by using transition systems of each aspect by example of model of working with graphic processing (GPU) systems where working GPU cycle is decomposed by parts and presented as transition systems with further transforming to Petri net with liveness checking[5].

IT&I-2020 Information Technology and Interactions, December 02–03, 2020, KNU Taras Shevchenko, Kyiv, Ukraine
EMAIL aprowata1@bigmir.net (A. Provotar); veres@ukr.net (M. Veres); maxsam125@gmail.com (M. Samoilenko)
ORCID: 0000−0002−6556−3264 (A. Provotar); 0000-0002-8512-5560 (M. Veres); 0000−0002−2237−7138 (M. Samoilenko)

3. Using IoT device in curriculum

Let's describe overall view of using IoT in education curriculum. Let's assume that there is theoretical educational IoT device which is controlled by student's command in moment of solving educational task. This IoT device illustrates task invocation, which helps student understand correctness of command according to task. Task for student is generated by task service and student solves this task by means of user interface service which handle user task's input and user interface service interacts with IoT. Also, IoT can be used in group of student which can share common task generated from task service – task generates accordingly to data collected for group of IoT devices paralelly. There are three aspect of interaction:

- user service – gets task from task service, interacts with user input, transform input to task command and sends to IoT device,
- task service – generates task for students, collects information about each IoT device and generates task according to information collected. Task service controls parallel invocation of task of all IoT devices assigned
- IoT – device which performs each command obtained by user interface assigned.

In conclusion, user service, task service and IoT can be presented as transition system in the next chapter.

4. Petri net of IoT system

Let's introduce transition systems and Petri net [6]. Transition system is a system $A = (S, T, \alpha, \beta,\ s_0)$ where:

- S is finite or infinite set of states;
- T is finite or infinite set of transitions
- α, β are two relations from T to S where each transition t of T set results in two states $\alpha(t), \beta(t)$ which are start and end of transition t correspondingly.
- s_0 is the start state of transition system.

Let's $A_1, \dots, A_n$ are transition systems where $A_i = (S_i, T_i, \alpha_i, \beta_i, s_0),\ i = 1, \dots, n$. Constraint of synchronization is subset T of set $(T_1 \cup \{\varepsilon\}) \times \dots \times (T_n \cup \varepsilon) \backslash \{(\varepsilon, \dots, \varepsilon)\}$ where ε is identical transition which is absence of any action in transition system. Elements of set T are global transitions. If $t = (t_1 \dots t_n) \in T$ and $t_i \neq \varepsilon$ then transition system A_i takes a part in the transition t. Tuple $A = (A_1 \dots A_n, T)$ is multiplication of transition systems $A_1 \dots A_n$, which are multiplication components of A. Global transition $t = (t_1, \dots, t_n)$ is modeling possible transition to $A_1 \dots A_n$. If transition $t_i = \varepsilon$ then transition system A_i doesn't take a part in the global transition t .

Petri net (P, T, M, F_0) describes multiplication $A = (A_1, \dots, A_n, T)$ of transition systems $A_i = (S_i, T_i, \alpha_i, \beta_i, s_0)$ where $A_i \cap A_j = \emptyset$ with $i \neq j. i, j = 1,2, \dots, n$ if $P = S_1 \cup S_2 \cup \dots \cup S_n$, $T = T$, $F = \{(s,t) \,|\, t_i \neq \varepsilon\ \&s = \alpha_i(t_i)\} \cup \{(t,s) \,|\, t_i \neq \varepsilon\ \&s = \beta_i(t_i)\}$ for some $i = \{1,2, \dots, n\}$ where t_i defines i-th component $t \in T$, $M_0 = (S_0^1, S_0^2, \dots, S_0^n)$. Semantic of multiplication of transition systems and semantic of Petri net which is illustration of this multiplication, accords in sense of sequence of global transitions $t_1, \dots, t_k$ is global history of multiplication of transition systems only in case of using allowable sequence of results of transitions for Petri net.

Let's abbreviate transition system as TS and there are:

- TS_1 – IoT
- TS_2 – user interface service
- TS_3 – task service

Interpretations of states and transitions from fig.1 for TS_1 are next:

- a_0 – waiting command from TS_2
- p_1 – fetching command from TS_2
- a_1 – command fetched is stated and is ready to use data from command

- p_2 – send command to controller to start
- a_2 – controller started command invocation
- p_3 – get command invocation result
- a_3 – result is fetched and prepared to return to TS_2
- p_4 – result of invocation is sending to TS_2
- a_4 – result of invocation is sent and there is nothing to do.
- p_5 – transition to a_0

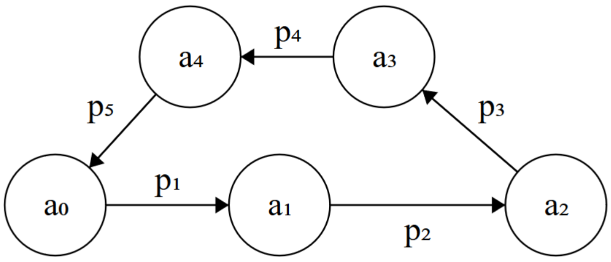


Figure 1: TS_1 IoT

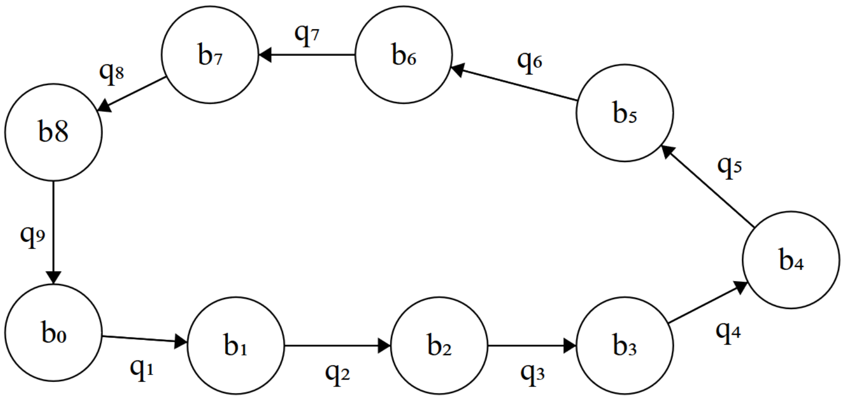


Figure 2: TS_2 user interface service

Interpretations of states and transitions from fig. 2 for TS_2 are next:

- b_0 – waiting for user input
- q_1 – fetching command from user input
- b_1 – fixation command from user input
- q_2 – send command to TS_3
- b_2 – command is sent to TS_3, waiting for result
- q_3 – getting result from TS_3
- b_3 – fixation result command from TS_3
- q_4 – sending command to TS_1
- b_4 - command is sent to TS_1, waiting for result
- q_5 – getting result from TS_1
- b_5 – fixation result command from TS_1
- q_6 – send result command from TS_1 to TS_3
- b_6 - command is sent to TS_3, waiting for result
- q_7 – getting result command from TS_3
- b_7 - fixation result command from TS_3
- q_8 – show result of command invocation
- b_8 – result is shown

- q_9 – go to initial state: wait for user input

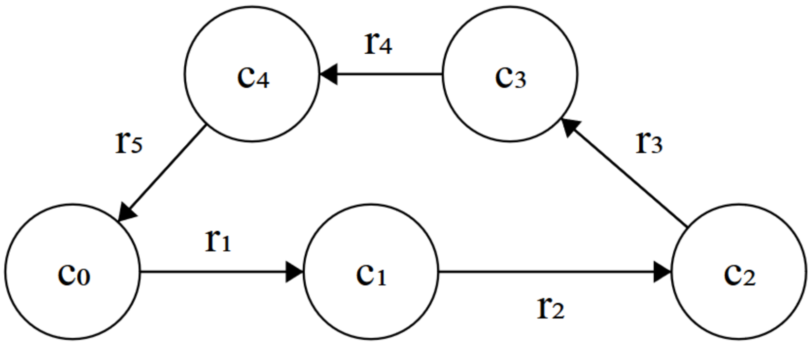


Figure 3: TS_3 task service

Interpretations of states and transitions from fig. 3 for TS_3 are next:

- c_0 – waiting command from TS_2
- r_1 – fetching command from TS_2
- c_1 – command fetched is stated and is ready to use data from command
- r_2 – send command to data source
- c_2 – waiting for data source response
- r_3 – get command invocation result
- c_3 – result is fetched and prepared to return to TS_2
- r_4 – result of invocation is sending to TS_2
- c_4 – result of invocation is sent and there is nothing to do.
- r_5 – transition to c_0

By available transition systems there is a possibility to build IoT system as Petri net in synchronous multiplications of transition systems TS_1, TS_2, TS_3 with global transitions

$$T_1 = \{(\varepsilon, q_1, \varepsilon), (\varepsilon, q_2, \varepsilon), (\varepsilon, q_2, r_1), (\varepsilon, q_2, r_2), (\varepsilon, q_2, r_3), (\varepsilon, q_2, r_4), (\varepsilon, q_3, \varepsilon),$$
$$(\varepsilon, q_4, \varepsilon), (p_1, q_4, \varepsilon), (p_2, q_4, \varepsilon), (p_3, q_4, \varepsilon), (p_4, q_4, \varepsilon), (\varepsilon, q_5, \varepsilon),$$
$$(\varepsilon, q_6, \varepsilon), (\varepsilon, q_6, r_1), (\varepsilon, q_6, r_1), (\varepsilon, q_6, r_2), (\varepsilon, q_6, r_3),$$
$$(\varepsilon, q_6, r_4), (\varepsilon, q_7, \varepsilon), (\varepsilon, q_8, \varepsilon), (\varepsilon, q_9, \varepsilon), (\varepsilon, \varepsilon, r_5), (p_5, \varepsilon, \varepsilon)\}$$

Petri net which models all transition systems is built by using set of constraints of synchronous multiplications T_1 as showed fig. 4

Petri net states and transitions references to TS

- $s_0(a_0 b_0 c_0)$
- $t_1 - q_1$
- $s_1(a_0 b_1 c_0)$
- $t_2 - q_2$
- $s_2\ (a_0 b_2 c_0)$
- $t_3 - r_1$
- $s_3(a_0 b_2 c_1)$
- $t_4 - r_2$
- $s_4(a_0 b_2 c_2)$
- $t_5 - r_3$
- $s_5(a_0 b_2 c_3)$
- $t_6 - r_4$
- $s_6(a_0 b_2 c_4)$
- $t_7 - q_3$
- $s_7(a_0 b_3 c_0)$
- $t_8 - q_4$
- $s_8(a_0 b_4 c_0)$

- $t_9 - p_1$
- $s_9(a_1b_4c_0)$
- $t_{10} - p_2$
- $s_{10}(a_2b_4c_0)$
- $t_{11} - p_3$
- $s_{11}(a_3b_4c_0)$
- $t_{12} - p_4$
- $s_{12}(a_4b_4c_0)$
- $t_{13} - q_5$
- $s_{13}(a_0b_5c_0)$
- $t_{14} - q_6$
- $s_{14}(a_0b_6c_0)$
- $t_{15} - r_1$
- $s_{15}(a_0b_6c_1)$
- $t_{16} - r_2$
- $s_{16}(a_0b_6c_2)$
- $t_{17} - r_3$
- $s_{17}(a_0b_6c_3)$
- $t_{18} - r_4$
- $s_{18}(a_0b_6c_4)$
- $t_{19} - q_7$
- $s_{19}(a_0b_7c_0)$
- $t_{20} - q_8$
- $s_{20}(a_0b_8c_0)$
- $t_{21} - q_9$

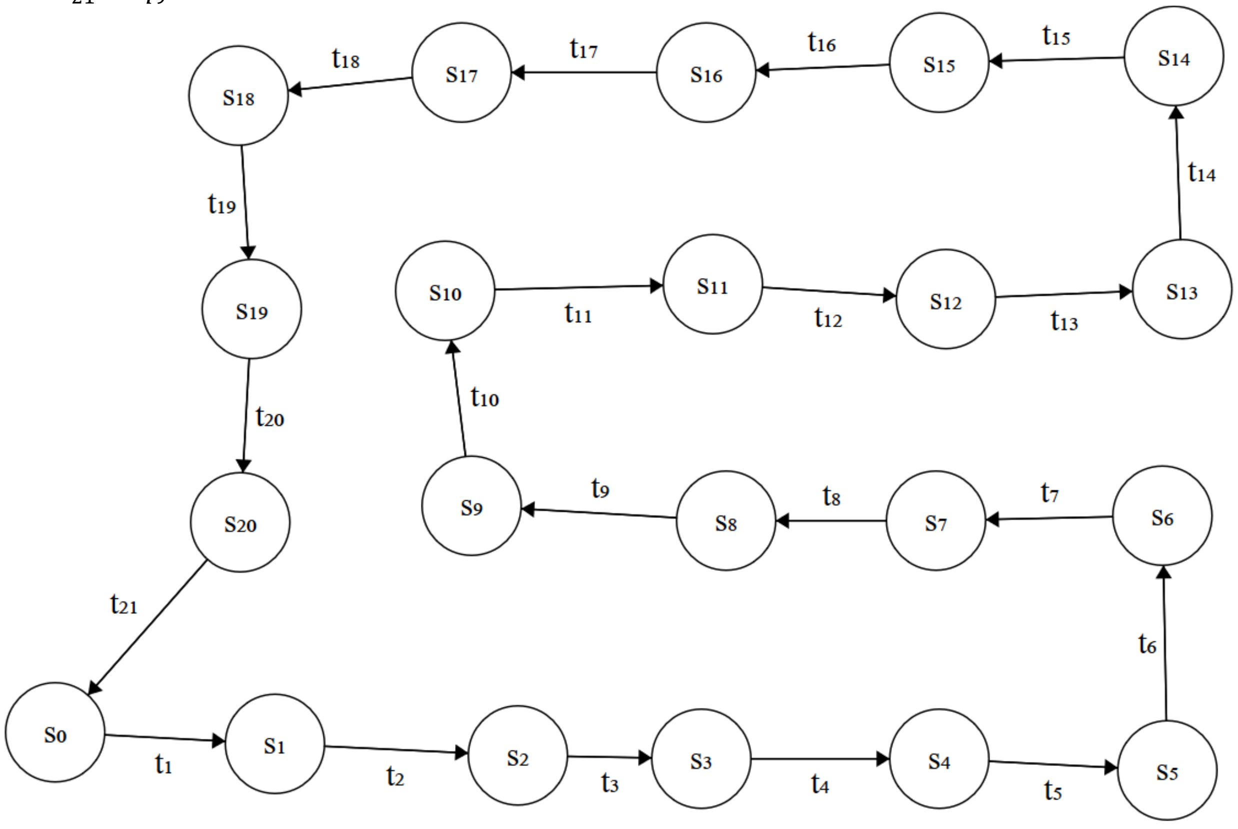


Figure 4: Petri net of multiplication TS_1, TS_2, TS_3 with global transitions

After building Petri net there is a task of checking correctness of model – let's check liveness of Petri net. For liveness checking it needs to find solution of equation $A \cdot x = 0$ where A is incidence matrix for this Petri net on fig. 5.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}	t_{17}	t_{18}	t_{19}	t_{20}	t_{21}
s_0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
s_1	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_2	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_3	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_4	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_5	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_6	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s_7	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0	0
s_8	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0
s_9	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0
s_{10}	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0
s_{11}	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0
s_{12}	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0
s_{13}	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0
s_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0
s_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0
s_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0
s_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0
s_{18}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	0
s_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0
s_{20}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1

Figure 5: incidence matrix of Petri net A

According to TSS [7] one solution is on fig. 6.

t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}	t_{17}	t_{18}	t_{19}	t_{20}	t_{21}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 6: solution of $A \cdot x = 0$

All transitions of Petri net are covered by positive invariant values so Petri net is viable.

5. System architecture of Petri net model implementation

Here Petri net can be implemented as system on fig. 6.

Task service generates one common shared task parallelly for group of IoT devices. Common task means one common task with specific context of each IoT device. Specific context can have task difficulty, IoT configuration, actual state of IoT.

According to transition systems above there is system implementation:

- Task service transition system is implemented as task service
- User interface service and IoT transition systems are implemented into education IoT system.

There is the scheme for implementation on fig. 6:

Let's view education IoT implementation: it can consist of educational robot as IoT, as Wi – Fi web server and mobile phone, as user interface service, as Wi – Fi client. Mobile phone is used as illustrative example of device with user interface. They have JSON restful communication on fig.7. Via mobile phone student can send command to robot to change location and task service generate tasks for students and can check command from mobile phone before sending to robot.

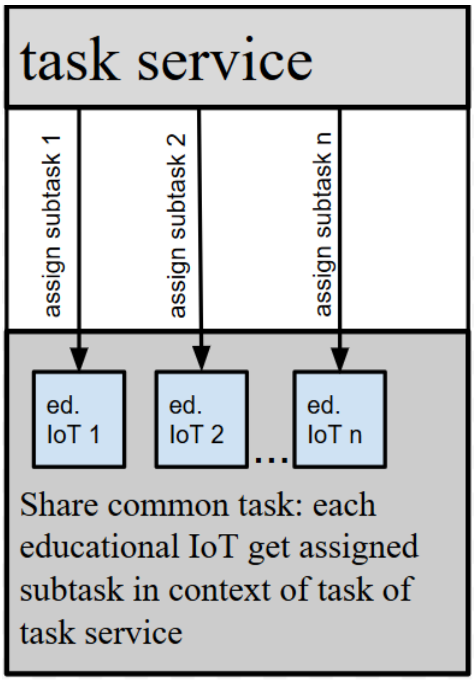

task service
assign subtask 1
assign subtask 2
assign subtask n
ed. IoT 1
ed. IoT 2
...
ed. IoT n
Share common task: each educational IoT get assigned subtask in context of task of task service

Figure 6: Petri net system architecture scheme

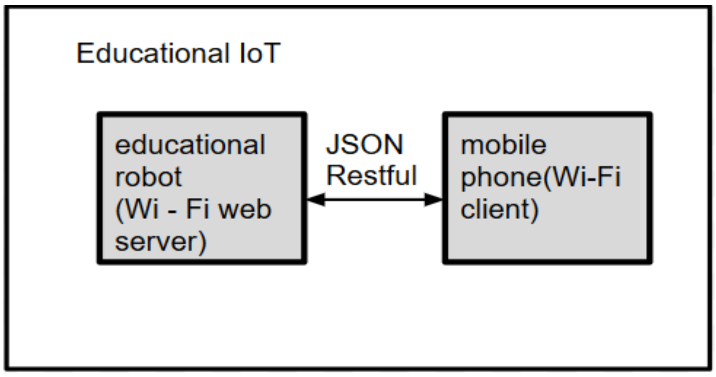

Educational IoT
educational robot (Wi - Fi web server)
JSON Restful
mobile phone(Wi-Fi client)

Figure 7: Interaction between ed. robot and mobile phone

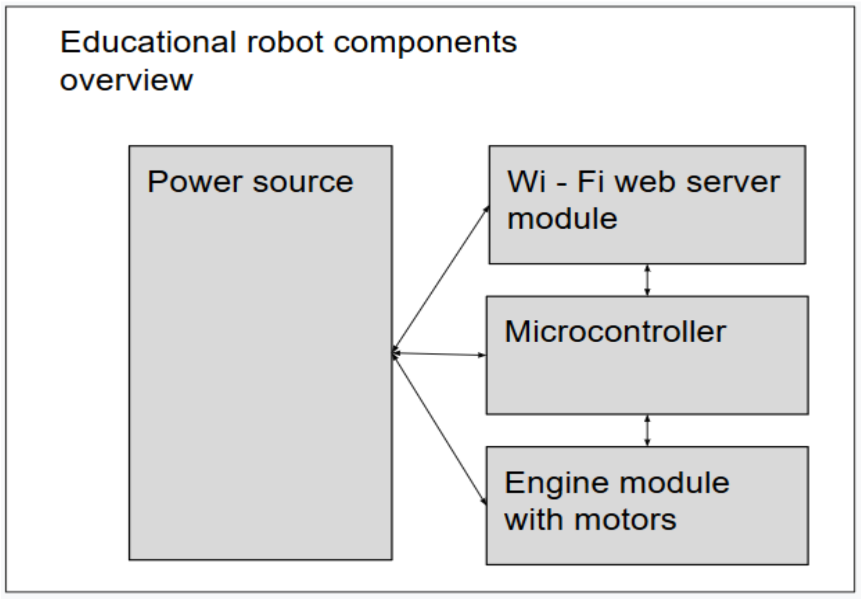
Let's see IoT robot inner components: there is a power source, which gives a possibility to use robot IoT autonomously, Wi – Fi web server module which communicates with mobile phone as user interface service via JSON protocol, microcontroller which controls command invocation, engine module with motors which can move robot to other location on fig. 8. Robot can have functionality of changing location. Group of students solves common changing location task with purpose to impove programming skills. Task service controls all IoT devices assigned to common task and generates new task to each IoT device accordingly.

Task service can be implemented as web – server which generates, checks task for mobile phones for students. Web service interacts with mobile phone via JSON restful protocol and web – service can support parallel connections which allow to do common task in group of students as showed of fig. 9.

Web service uses database as data source for keeping all actual information data about all IoT assigned to task service.

Let's see implementation of Petri net as interaction of systems.

- User interface (mobile phone) waits for user input, which is command programmed for IoT robot.
- After user input user command in fetched and going to be sent to task service via JSON protocol
- User command is sent to task service web server via JSON protocol and wait for task service response



Educational robot components overview
Power source
Wi - Fi web server module
Microcontroller
Engine module with motors

Figure 8: Overview of IoT robot components

- Task service web server obtains the command
- Task service web server sends the command to database to get actual info about other IoT
- Task service web server got database result about the command invocation
- Task service web server return result with the command to user service (mobile phone)
- User service sends the command to IoT robot
- Iot service (IoT robot) obtains a command via JSON protocol by means of Wi – Fi controller
- IoT service does the command by using IoT robot’s components
- IoT service sends to User interface the command invocation result
- User interface service obtains a result from IoT service
- User command is sent to task service server and wait for task service response
- Task service obtains the command
- Task service sends the command to database to get actual info about other IoT and generates new tasks
- Task service got database result about the command invocation and new task
- Task service return result with the command to user service (mobile phone)
- User interface service shows result to end users and waits for next input
- Implementation of IoT system is similar to Petri net transitions sequence.

6. Conclusions

There is education system for possibility to solve common task in group of students which can be decomposed into three transition systems IoT, user interface service and task service. After Petri net, as multiplication of three systems, is used for checking liveness of IoT educational system model. This model can be implemented as IoT robot as IoT transition system, mobile phone as user interface service and task server as task service. This system supports parallel work which allows generate one common task for group of IoT devices.

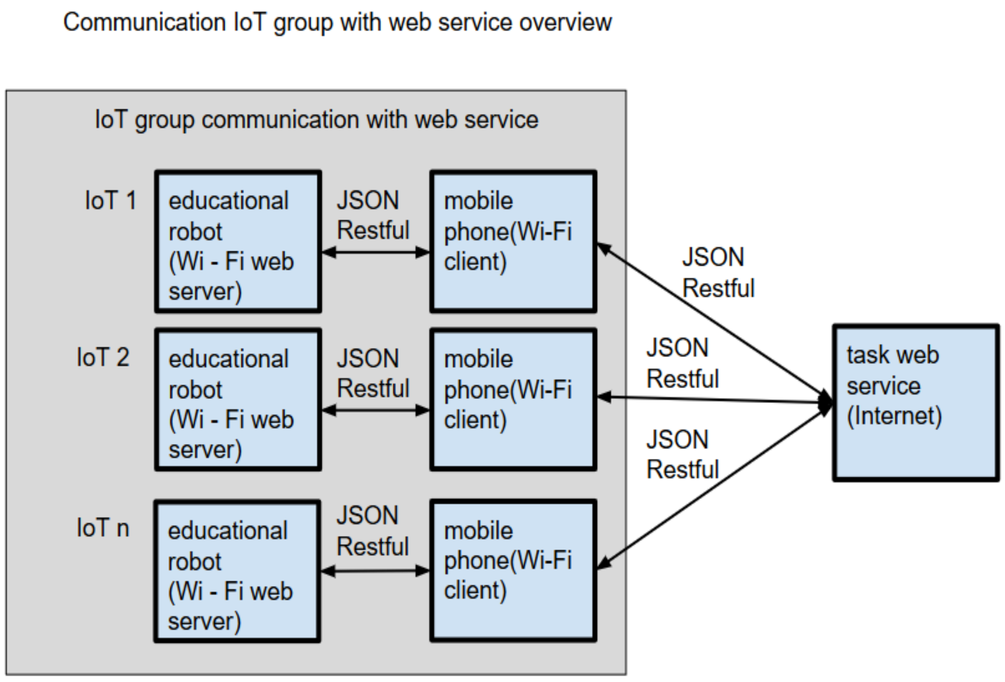

Communication IoT group with web service overview
IoT group communication with web service
IoT 1
educational robot (Wi - Fi web server)
JSON Restful
mobile phone(Wi-Fi client)
IoT 2
educational robot (Wi - Fi web server)
JSON Restful
mobile phone(Wi-Fi client)
IoT n
educational robot (Wi - Fi web server)
JSON Restful
mobile phone(Wi-Fi client)
JSON Restful
JSON Restful
JSON Restful
task web service (Internet)

Figure 9: Overview of IoT education system

7. References

[1] Karatrantou, Anthi & Panagiotakopoulos, Christos. (2008). Algorithm, Pseudo-Code and Lego Mindstorms Programming, Workshop Proceedings of SIMPAR 2008 Intl. Conf. on Simulation, modeling and programming for autonomous robots. Venice(Italy) 2008 November, 3-4, ISBN 978-88-95872-01-8, pp. 70-79

[2] Nielsen, Claus & Adams, Paul. (2015). Active learning via LEGO MINDSTORMS in Systems Engineering education. 489-495. 10.1109/SysEng.2015.7302802, 2015 IEEE International Symposium on Systems Engineering (ISSE).

[3] Mylonas, Georgios & Amaxilatis, Dimitrios & Pocero, Lidia & Markelis, Iraklis & Hofstaetter, Joerg & Koulouris, Pavlos. (2019). Using an Educational IoT Lab Kit and Gamification for Energy Awareness in European Schools. This is a preprint version of a paper submitted toFabLearn Europe'18, Proceedings of the Conference on Creativity and Making in Education. DOI: 10.1145/3213818.3213823

[4] Lloret, Jaime & Sendra, Sandra & Gonzalez Ramirez, Pedro & Parra, Lorena. (2019). An IoT Group-Based Protocol for Smart City Interconnection. 10.1007/978-3-030-12804-3_13. Springer Nature Switzerland AG 2019S. Nesmachnow and L. Hernández Callejo (Eds.): ICSC-CITIES 2018, CCIS 978, pp. 164–178, 2019. https://doi.org/10.1007/978-3-030-12804-3_13

[5] S. L. Kryvyi, S. D. Pogorilyy, M. S. Slinko , MODEL JUSTIFICATION OF GPU-BASED APPLICATIONS, DOI https://doi.org/10.15407/usim.2018.04.0046 , ISSN 0130-5395, Control systems and computers, 2018, № 4

[6] Kryvyi S. L., Boyko Y. V., Pogorilyy S. D., Boretskyi O. F., Glybovets M. M. Design of Grid Structures on the Basis of Transition Systems with the Substantiation of the Correctness of Their Operation. Cybernetics and Systems Analysis. January 2017, Volume 53, Issue 1, pp 105—114. Springer Science+Business Media New York 2017.

[7] Kryvyi S.L. 2015. Linear Diophantine constraints and their application. Chernivtsi: “Bukrek” Publishing House.