

# Model for Introducing STEM<sup>1</sup> into High School Computer Science Education\*

VYTAUTAS ŠTUIKYS, RENATA BURBAITĖ, TOMAS BLAŽAUSKAS, DOMINYKAS BARISAS  
and MIKAS BINKIS

Software Engineering Department, Kaunas University of Technology, Studentu 50, Kaunas, Lithuania.

E-mail: vytautas.stuikys@ktu.edu, renata.burbaite@ktu.lt, tomas.blazauskas@ktu.lt, dominykas.barisas@ktu.lt, mikas.binkis@ktu.lt

In this paper, we discuss a vision and the model on how it is possible to introduce STEM\*\* into a single Computer Science (CS) course to educate students at the High School level. We present our approach at four levels: (i) as a conceptual model; (ii) as a framework that gives more information about the conceptual model, (iii) as a process-based vision that is close to the implementation level; and (iv) as two Case Studies that outline the implementation and use aspects. The essence and novelty of our approach is the seamless integration of the essential attributes of STEM-driven and CS-oriented content with the STEM pedagogy features. The use of meta-programming as the implementation technology enables to achieve that though the pre-designed Smart Generative Learning Objects, though we discuss those aspects only fragmentally. We also present an evaluation of the approach from the methodological and pedagogical perspectives, by describing both advantages and limitations.

**Keywords:** STEM-driven CS education; robotics; process-based model

## 1. Introduction

The rapid development of science and technology raises new challenges for labor market and education. Perhaps the most important one is to educate STEM-literate students who, after graduating, could be able to join the modern labor market as fluently and easily as possible. Typically, the acronym STEM means *Science, Technology, Engineering, and Mathematics*. The STEM-based education is defined as “an interdisciplinary approach to learning where rigorous academic concepts are coupled with real world lessons as students apply science, technology, engineering, and mathematics in contexts that make connections between school, community, work, and the global enterprise enabling the development of STEM literacy and with it the ability to compete in the new economy” [1].

Many reports predict that the demand for the STEM-based workforce in the 21st century will be growing continuously. Therefore, STEM-oriented education becomes well timed and extremely significant, though there are many challenges and issues that require of better understanding to manage them adequately. Among others, they include: (i) motivating and engaging students to participate in STEM-oriented learning [2–4]; (ii) integrating STEM-oriented aspects into the school curriculum [5, 6]. The others include: (iii) selecting adequate technological tools, pedagogical methods and activities for the paradigm [5, 7]; (iv) providing

students’ research and introducing real problem solving so that to enforce critical and computational thinking, to develop collaborative learning skill for modern workforce market [4–13].

As science, technology, engineering and mathematics are highly broad and heterogeneous fields, there are quite different views and approaches to deal with STEM education. One specific view relates to the role of Computer Science (CS) in STEM-based education. Gander [14] observes that CS is the leading science of the 21st century. Similarly to mathematics, practically all sciences use CS approaches. According to the author, it has to be a part of general knowledge in education. On the other hand, *Informatics Europe* and *ACM Europe* convincingly state that CS education in the school must consist of two parts. The first should focus on learning to make good use of ICT and its devices, also called as *Digital Literacy*. These skills are short living knowledge, because they are changing with technology. The second should focus on learning the fundamentals of CS that are essential to understand our digital world. This is *Informatics*. The latter brings “long living knowledge which lasts forever and does not change with technology”.

Though there are a variety of approaches, we need still to improve CS education on the STEM paradigm at High School because of the inadequacy between the currently available technological capabilities and needs of the interdisciplinary knowledge for the 21st century labor market. Our research objectives therefore are to achieve a higher efficiency in dealing with integrative aspects of the advanced technology and STEM pedagogy into CS and engineering education through systematization and automation.

<sup>1</sup> Science, Technology, Engineering, and Mathematics (STEM). See list of acronyms and an extended explanation at the end of this paper.

The aim of this paper is to discuss the approach on how to introduce the STEM paradigm into CS education at school. We present the approach as a vision and STEM-driven model. Two basic attributes predefine the essence of our approach. The first is the use of educational robots and other smart devices for CS education at the secondary school. The second is the use of the robot-oriented teaching content that we represent in a specific way. One type represents the so-called smart learning objects [15] adapted for the STEM needs (meaning meta-programming-based generative learning objects (GLOs) with extended features and possibilities). Another type represents component-based LOs such as tutorials, quizzes, etc. Both attributes, when implemented, require a wide range of interdisciplinary knowledge to define and integrate STEM components (S, T, E, and M) into CS education. In our model, STEM components represent the following items. The component S covers CS and partially physics topics, i.e. those topics that relate to understanding of robotics functionality. The component T covers a variety of technologies, including the Internet, educational software tools, communication, etc. The component E covers constructing of an educational robot system from available parts, designing of the educational environment and testing it and robot's functionality through modeling. We are also able to introduce the engineering aspects through dealing with real-world tasks or their prototypes. The component M is implicit in our model. Either it appears in the task dedicated for the robot, or it appears within the algorithm to specify the robot's functionality. In our model, the component S stands for the root while the remaining ones are supplementary to define the STEM paradigm within the CS teaching curriculum. Therefore, we consider such a paradigm that integrates STEM within the single teaching course (i.e. CS course also known as Informatics).

Our approach has multiple aspects that we treat as a novelty. (1) It focuses on the wide-scale analysis as a context to build the robot-based environment for STEM-driven CS education at school. (2) It covers the full life-cycle processes ranging from constructing/testing of the robot itself to the development and use of robot control programs for the CS education. (3) The STEM knowledge is explicit and integrated within the processes and content and, therefore, it is easy to extract and to present the content for learning. (4) When implemented, our approach exploits the advanced technologies such as feature-based modeling and meta-programming, enabling to introduce systemization and generalization and to achieve a higher extent of reuse through automation, though we present those aspects as a by-product only.

The structure of the paper is as follows. In Section 2, we discuss the related work and motivate this research. In Section 3, we describe the basic idea of our approach that contains a conceptual model and framework providing more details on the model. In Section 4, we provide a process-based vision of our approach with the focus on the development of STEM-driven content and its use in CS education. In Section 5, we analyze two Case Studies to demonstrate a practice of the approach. In Section 6, we deliver a summarizing evaluation from the pedagogical perspective. In Section 7, we summarize the capabilities of the approach and identify some difficulties. Finally, in Section 8, we provide the conclusion and outline the future work.

## 2. Related work

With regard to the aim of this paper, we categorize the related work into three parts as follows: (i) STEM education challenges; (ii) the role of CS in STEM education; (iii) educational robotics in STEM education. Before starting a discussion, one should know that so far in the literature, there is no uniform definition of the term STEM. We have presented one in the introduction; however, the following definition is more relevant to our context. According to [11], STEM is often defined as “learning and/or work in the fields of Science, Technology, Engineering and Mathematics, including preliminary learning at school prior to entry into the specific disciplines”.

*STEM education challenges.* We have already enlisted some challenges in the introduction. Here we extend a discussion on the topics. Many researchers identify the *learner's motivation and engagement* as the most important issue in STEM education. Therefore, there are many suggestions and approaches on how to deal with the issue. We classify those approaches as follows: (i) using technological toys (i.e. Lego, Knex) [2, 3, 9]; (ii) increasing the role of the teacher as a mentor in STEM classes [2, 4, 6, 16, 17]; (iii) introducing after-school activities related to STEM [6, 17, 18]; and (iv) focusing on active learning based courses as well as Engineering Based Learning [19].

The next challenge is the *integration STEM-oriented aspects into school curriculum*. That covers: (i) availability of advanced STEM courses [4, 8, 18]; (ii) IT/STEM oriented learning (e.g. GPS, GIS, robotics programming) [7]; (iii) curriculum reform agendas and programmes [5, 6, 10, 11]; (iv) conceptualizing of STEM education [10]. The other issues largely relate to technology. The choice of *appropriate technological tools* is important too: (i) in creating adequate technological infrastructures to support collaboration [4]; (ii) for monitoring and

programming easiness [20]; (iii) using the fully configurable user interface [20]; and (iv) making improvements in STEM learning using robotics [3, 9].

Finally, a large body of reviewed papers focuses on the selection of *suitable pedagogical methods, activities and resources*. Those issues include: (i) providing learners' research, problem solving, critical computational thinking, collaborative learning skills [4, 7, 8, 21]; (ii) making and tinkering activities [16], including robotics activities [3]; (iii) implementing consequential, side-by-side [16], inquiry-based [6, 7, 16, 19, 20], design-based [7], game-based [21], project-based [17] learning; and (iv) using socio-cultural approaches to the design of learning environments [16].

*The role of CS in STEM-oriented education.* The following extract from [14], perhaps, is the best to characterize the role of CS in STEM education.

“Computer Science is the leading science of the 21st century. It is used like mathematics in all sciences. It has to become part of general knowledge in education. Informatics Europe and ACM Europe convincingly state that computer science in the school must consist of two parts:

1. Learn to make good use of IT and its devices. This is called Digital Literacy, often also ICT. These skills are short living knowledge, they change with technology.
2. Learn the fundamentals of computer science, which are essential to understand our digital world. This is called Informatics. It is long living knowledge, which lasts forever and does not change with technology”.

The papers [7, 8] also emphasize the role of CS in STEM-oriented education. According to the papers, CS within the context of STEM means the advanced use of ICT and its devices to perform data processing in different domains (e.g. biology, physics, and chemistry).

In the context of STEM-oriented education, the development of computational thinking and problem solving skills [22] is one of the most important targets. The concepts and approaches taken from CS, such as structured task decomposition, abstractions and patterns generalizations, iterative and parallel thinking, conditional logic, debugging and systematic error detection, etc., are directly influencing the development of computational thinking. One another source [15] summarizes the challenges to teach CS fundamentals such as: usage of adequate learning models; teaching context and learning personalization; learners' motivation improvement; cognitive problems related to high-level abstractions, theoretical knowledge and practice; content adaptation to learner's context, etc. The listed statements suggest that CS education

requires to be changed and is an important part of STEM-oriented education.

*The role of educational robotics in STEM-driven CS education.* The paper [23] provides a systematic review by exploring the educational potential of robotics in schools. The paper introduces three groups of the essential problems: (a) using robots as educational tools; (b) testing of effectiveness of robots; (c) providing future perspectives of use robotics. The paper also concludes that robotics improve learning achievements in STEM-related areas such as construction, mechatronics, and programming. The other works [24, 25] highlight the importance of educational robotics for promoting 21st century core skills in creativity and innovation, critical thinking, problem solving, communication and collaboration, technological literacy, personal and social responsibility. The papers [26–28] provide discussions on how educational robotics help to develop computational thinking skills. The following papers evaluate educational robotics as the most effective supporting tools and learning methods in CS education [29–34]. The next portion of works focuses on robot-based learning environments to support modern learning methods and on the involvement of students in knowledge construction processes [34–38]. The following works [29, 39–44] distinguish CS topics that relate to the use of robot-based learning environments.

In summary, it is possible to state the following. One of the biggest challenges in STEM-driven CS education is the implementation the most effective learning methods, resources and tools to achieve learning goals. Though there are many possible solutions, the use of educational robotics in schools should be the focus in this regard. Though the provided analysis covers multiple topics and a variety of approaches, there is still a big gap between the current technological capabilities and needs for the improvement of CS education on the STEM paradigm at school. This is especially true in terms of integrating the advanced technology with STEM pedagogy aiming at achieving a higher efficiency through systematization, integration and automation. We hope that our approach is able to bring the relevant contribution in this respect.

### 3. A general description of our approach

This description gives a general understanding of our approach and includes two parts: (1) a conceptual model of STEM-driven CS education (Section 3.1) and (2) a framework presenting more details on the model (Section 3.2). The conceptual model deals with the pedagogical approach (learning-by-doing), the scenarios and focuses on the STEM-driven knowledge introduced through activ-

ities of the scenarios. The framework introduces STEM-driven components, their interactions and attribute-based vision of two basic components, i.e. STEM pedagogy and STEM-related content.

### 3.1 A conceptual model of STEM-driven CS education

In our approach, educational robotics is a primary concept that leads to the development of the STEM-driven CS curriculum to educate students at the secondary school level. Fig. 1(a) presents the conceptual model to understand our approach. As we stated in Section 2, educational robots are extremely useful instruments to provide interdisciplinary knowledge regarding all STEM components. Here, therefore we accept the educational robot (ER) as a Physical Learning Object (PLO) [45] to provide STEM-driven learning and achieve education objectives. By notions S, T, E, M within the circles, we mean the adequate pieces of knowledge obtained using the ER as PLO. The darkened area is to be thought as the integrated STEM knowledge.

The main pedagogical approach we use within the introduced model is *learning-by-doing*. A motivated involvement of students in this process is a primary concern. We split the process into two phases treated as scenarios here: (1) *activities of a preparatory work* to construct and test the robot itself and (2) the robot's use in full functionality (in the *mode use-as-is*) to solve real world tasks (sub-tasks) or their prototypes. Scenario 1 includes a set of mini-tasks such as researching of robot's components (motors, sensors) characteristics. As solving of mini-tasks is a short-time process (students may enjoy immediately by the achieved result) and the sequence of solving mini-task is logically dependent

(i.e. the next mini-task requires the previous knowledge of the previous mini-tasks), these factors highly self-motivate students in the involvement into the process. The next motivating factor is the possibility of the student to make an independent decision (teacher acts as a mentor). A successful pass through Scenario 1 (i.e. a student sees a working thing and its usefulness in practice) is also a motivating factor for the Scenario 2. We also motivate students by introducing the real world tasks that robots are able to solve, or by showing a film or visual slides on how this is happening in reality. The analysis of the robot as PLO in Scenario 1 as well as in Scenario 2 leads to the following pieces of knowledge obtained in the process 'learning-by-doing'.

*Engineering knowledge (E)*. Robots may function using the only *motors* to realize such tasks as the *straight-line movement* and *rotary motion*. Both are indeed sub-tasks of more complex tasks such as carrying objects in manufacturing by the robot from location A to location B, etc. However, sensors enable to extend the robot's functionality and capabilities largely. There are different kinds of sensors used in robots (ultrasonic, light, touch, color, sound, and infrared, etc.) [46]. The list of tasks that require the use of sensors is much wider (Obstacle detection, Line following, Light following, Service activities using touch and sound sensor, etc.). Note that all these are prototypes of real world tasks. In general, it is possible to interpret sensors as either engineering or technology products. As in using Scenario 2, our main focus is taken to the Robot Control Program (RCP) development to teach fundamentals of programming, students have to study characteristics of different types of sensors before developing RCPs, so in this way

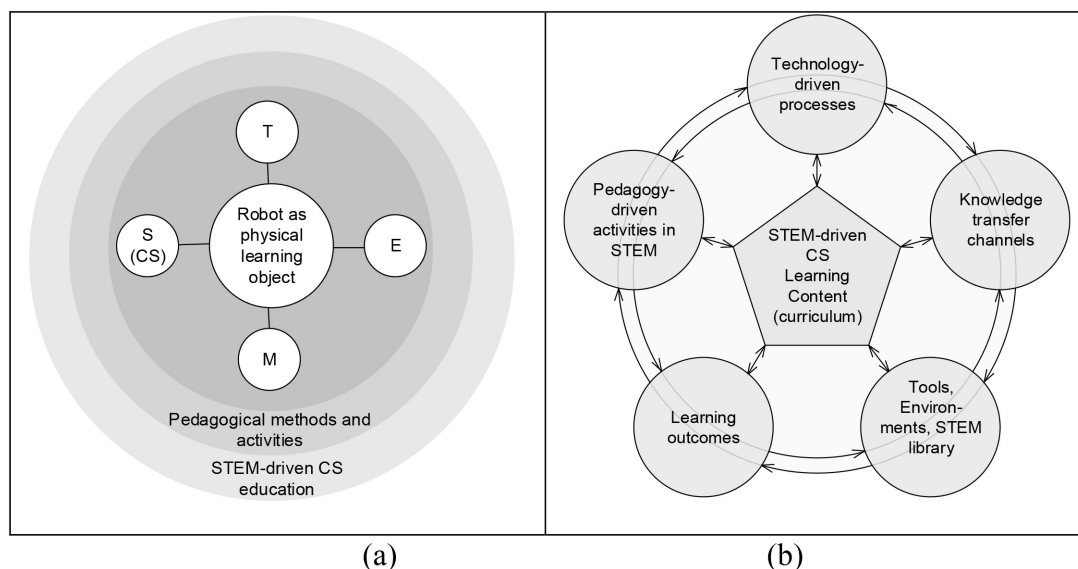


Fig. 1. A conceptual model (a) and framework (b) of STEM-driven CS education at high school.

obtaining the knowledge in engineering. Next, a construction of educational robots from scratch, i.e. assembling the robot from parts, is indeed an engineering activity. Students provide this activity working in groups.

*Technology-related knowledge (T).* This knowledge comes through analyzing robots components such as mentioned before (Motors, LED—Light Emission Diodes, LCD—Liquid Cristal Display, etc.). The robot itself, i.e. the kinds of robots such as LEGO, ARDUINO, RASPBERRY Pi, or others [46] can be viewed as products of engineering or technology. Software tools used in e-learning and those used in our approach (they will appear later) are also examples of technology. As both kinds of knowledge (engineering and technology) are highly underpinned among themselves, sometimes it is difficult to consider them as separate items. Often, therefore, we consider them in combination.

*Science knowledge (S).* A typical example of gaining scientific knowledge in using robots as PLOs is the scientific *inquiry method*. Say, we have two types of sensors: color sensor and light sensor. As they are programmable units, it is possible to change color sensor functionality capabilities, i.e. by changing the intensity of light, we are able to achieve the behavior of the color sensor similar to the light sensor through means of soft programming. The reason of using color sensors in the role of light sensors may be pure practical: the number of available color sensors is insufficient in the classroom to demonstrate a particular task for all students. In this case, students are encountering with physical laws of light combined with scientific experimentation. Furthermore, an analysis of sensor properties and capabilities takes another portion of scientific knowledge. We are able to see those capabilities explicitly through sensor testing experiments. For example, the ultrasonic sensor is able to recognize obstacles being in the vertical position much better in comparison to those obstacles that are in a shifted position. However, to know the limits of shifting, we need to provide a scientific experiment.

*Knowledge in mathematics (M).* This knowledge appears in multiple cases (problem statement, analysis and representation of experimental results, task modeling, etc.). To illustrate that, we present the following examples. In formulating the robot's movement task, we have dependencies (distance/speed). This is a functional relationship in the mathematical sense, i.e.  $s = f(v)$  [38]. To ensure a stable construction of the robot, we need to perform its mathematical modeling. Furthermore, each task requires some calculations specific to a particular task. For example, the *ornament drawing* task to be performed by the robot needs calculation to ensure

a required shape of ornaments. Another example is Boolean algebra concepts in a traffic light problem statement. Note also that in most cases there is an integrated knowledge as we discuss below.

*Integrated STEM knowledge.* So far, we have discussed pieces of knowledge for each STEM component separately due to methodological reasons, though perhaps in the case of using robots, there is no such a type of knowledge at all. Instead, there is the integrated knowledge. For example, this knowledge arrives through modeling and creating of robot's construction, investigating characteristics of components, testing capabilities of components as separate activities. However, a large body of integrated knowledge arrives through the full cycle of analysis, which covers modeling, constructing and testing, including the development of RCPs for the robot testing needs. Note also that the testing phase gives some fragments of RCPs. It is possible to use them in describing a real task prototype to ensure STEM-based CS education. This, in fact, is the use of a previous knowledge in learning also contributing to the integration.

In the next sub-section, we present more details regarding our conceptual model.

### 3.2 A framework to implement the proposed conceptual model

The conceptual model brings the basic idea on CS STEM-driven education and motivates the paradigm only. Below therefore we present a framework that explains the way we need to pass to achieve the implementation phase that we discuss in Section 4.

The framework (see Fig. 1(b)) has been adopted from the more general one described in [15]. The adopted version comprises a set of components, the hidden processes within components, and two-sided external processes among components. The set of components includes *pedagogy-driven activities*, *technology-driven processes*, *knowledge transfer channels with the actors* involved, a set of *tools* used (including robots), *STEM-oriented teaching/learning content* and the *pedagogical/learning outcome*. In fact, Fig. 1(b) specifies the whole problem domain, though very abstractly, which we call STEM e-learning domain (further domain). To understand the domain, we need to look into the inside of each component, to discover their properties, their internal and external interaction. This can be done systematically using, for example, SWE approaches such as FODA [47] or other domain analysis methods.

This framework, of course, is still general enough. Nevertheless, it highlights two important issues: an extremely high *heterogeneity* within the domain and *diversity of the interplay* among components when they are oriented for using in the STEM-based

paradigm. Indeed, the pedagogy-driven activities induce, for example, the interplay among e-learning, STEM pedagogy and educational theories. The technology-driven processes, on the other hand, indicate on how the information induced by components we are able to transform and process using educational tools (computers, robots and other devices, educational software tools) in order to achieve the pre-specified STEM objectives. The tools are for ensuring the functionality and efficiency of the whole system. The knowledge transfer channels connect the main actors (students and teachers) at different ends of the channels. The latter is the core of the education process as a pure social activity.

The STEM-oriented teaching/learning *content* (we treat that as a set of real world tasks in the conceptual model) plays an exclusive role. From the viewpoint of functionality, content stands for *data-base* to enriching other components with the *information* that enables to start the processes, to initiate and support the functioning of the components and the whole system. Here we use the term database as a generic concept. In our case, when the database is implemented, it becomes the STEM library, containing within the structured content being represented in different forms (LO, RCP instances, or Smart Generative LOs that represent a set of related RCP instances). As a result, content stands for an intermediate link to connect and integrate the different in nature domains — social and technological. Finally, in a social sense, the pedagogical (teaching/learning) outcome needs to be thought of as a measure to reason about on how the component interplay was relevant to prescribed objectives, what difficulties could be identified within components and what improvements could be done in the future.

In this context, there are some observations important to state as follows.

1. The interplay among components specifies the functionality of a learning/teaching process. We

can model this functionality through component attributes. Though those attributes differ in semantics, when specified for modeling purposes, we are able to evaluate them using the adequate measures specific to each component, and then, to *express uniformly* (we will show that later in our Case Studies).

2. We need to harmonize the interplay between components with respect to the prescribed learning and teaching objectives. From the pure technological perspective, the harmonization should be correct, meaning that the interaction model is correct and we take into account the pre-specified constraints.
3. It is possible to enlarge the *space* for modeling functionality (the interplay between components) significantly if we take into account the possible values of different attributes for each component. As it is possible to express those values uniformly, we are able to integrate and specify that as a single content-based specification.
4. Using this framework, it is possible through analysis to extract main attributes related to S, T, E, and M aspects so that it would be possible to integrate them and represent as an attribute-based model (Fig. 2).

Here, by the term *domain*, we mean objects (such as teaching materials, including literature sources), approaches, processes, standards defined by the terminology taken from the STEM-driven CS curriculum. Note that Fig. 1(b) partially outlines our domain at a higher level of abstraction. By the term *domain analysis*, we mean activities, typically performed by a teacher or course designer, such as reading of teaching materials, extracting and representing the relevant information explicitly, using the previous knowledge and cognitive processes with or without the use of systematic domain analysis approaches (such as FODA [48, 49]).

By the *attribute-based model*, we mean the integrated characteristics (i.e. attributes, properties and

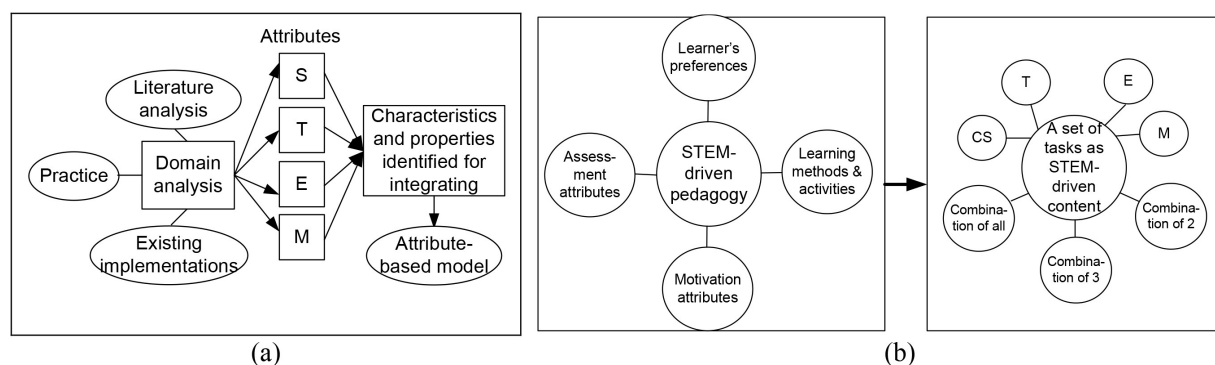


Fig. 2. A vision (a) for integrating STEM-driven pedagogy and content attributes (b).

**Table 1.** STEM-driven pedagogy attributes

Learner's preferences	Learning methods & activities	Motivation attributes	Assessment attributes
<b>Learner's level:</b> beginner, intermediate, advanced. <b>Learning style:</b> audial, visual, kinesthetic. <b>Learning pace:</b> slow, medium, fast. <b>Learning preference:</b> conceptual, example-oriented, case study, simulation, demonstration. <b>Learning objective:</b> research, survey, quick reference, basic introduction, project, assessment, and seminar.	<b>Learning-by-doing methods:</b> consequential, side-by-side, inquiry-based, design-based, game-based, project-based, problem-based. <b>Activities:</b> making and tinkering, robotics-related.	Technological devices, teacher as mentor, active learning, after-school STEM-related activities, short-time mini-task solving	<b>Bloom's taxonomy-based model by levels:</b> knowledge, comprehension, application, analysis, synthesis, evaluation. <b>SOLO taxonomy-based model:</b> <i>surface learning:</i> pre-structural, uni-structural, multi-structural; <i>deep learning:</i> relational, extended abstract. <b>Concepts' map model.</b>

their relationships) obtained within the domain objects and represented explicitly by texts, tables, pictures, etc. More formally, in terms of the feature-based notion [49], those characteristics are also known as *features* and their *variants*. Fig. 2 presents a vision (a) of the attribute-based model that includes attributes of two main components, the STEM-driven pedagogy and STEM-oriented content (b). Table 1 provides variants of the STEM-driven pedagogy attributes. Note that attributes related to the motivation, learning methods and activities specifically extended to be relevant for the STEM paradigm, while the remaining attributes (assessment and learner's preferences) are generic.

Here (see Fig. 2(b)) by the *STEM-driven content*, we mean the STEM library components, i.e. smart GLOs (further SGLOs) and component-based LOs (CB LOs, i.e. tutorials, quizzes, models, etc.). Note that SGLO is organized so that it contains within the GLO for generating RCPs and CB LO as a parameter of the SGLO (that will be explained in more detail in our Case Study 2). Now we are able to describe the way of how we implemented the proposed model. The next two sections are about that. In Section 4, we present a process-based vision to implement the model.

#### 4. Basis for implementing our approach: A process-based vision

The central task of the STEM paradigm is on how to integrate aspects of different disciplines into a coherent system so that it would be possible to provide the so-called integrated STEM education [50]. The aim of this section therefore is to highlight those aspects of our model on which basis we are able to achieve an overall integration and implement it in practice to provide CS teaching on the STEM paradigm. The core property of our approach is that we are able to recognize and extract the social, pedagogical, content and technology aspects related to STEM and represent them uni-

formly by features and their variants. Note that this property is universal and does not depend on the educational paradigm used. We have exploited this property, for example, in designing smart LO [15]. This property results in explicit representation of essential features. This, in fact, means that we are able to create the formal model that integrates the prescribed domain aspects through the features, their relationships and constraints. One can learn more about feature models either from the original sources taken from SWE literature (we recommend the one [48]), or from [15] (here feature models are directly connected to the e-learning domain).

The model we describe in this paper differs from those we considered in our previous research [15] by the following aspects.

1. The STEM-oriented feature model contains a wider spectrum of features.
2. The variants of features correspond to specific attributes related to STEM components; therefore, the model itself becomes more complex and more specific in terms of features, their variants, relationships and constraints.
3. The STEM model focuses on smart generative learning objects (SGLO) that essentially differ from the ones described in [15]. The difference is not only in an extended semantics, but also in a hierarchical structure, meaning that at the top there are pedagogy-oriented parameters and at the bottom – content-based parameters (i.e. they are separated explicitly).
4. Smart GLOs (SGLOs) in the STEM model are defined as the learning resource as follows:

$$\text{SGLOs} = \text{GLOs} + \text{CB LOs}. \quad (1)$$

Here GLO (i.e. generative LO) is a generic specification represented by a family of the related RCPs for a given real task, which are coded by means of meta-programming [15]; CB LO is a digital entity (text fragments, movies, models, guides, quizzes, tutorials) to be used and reused either for a single task or multiple tasks.

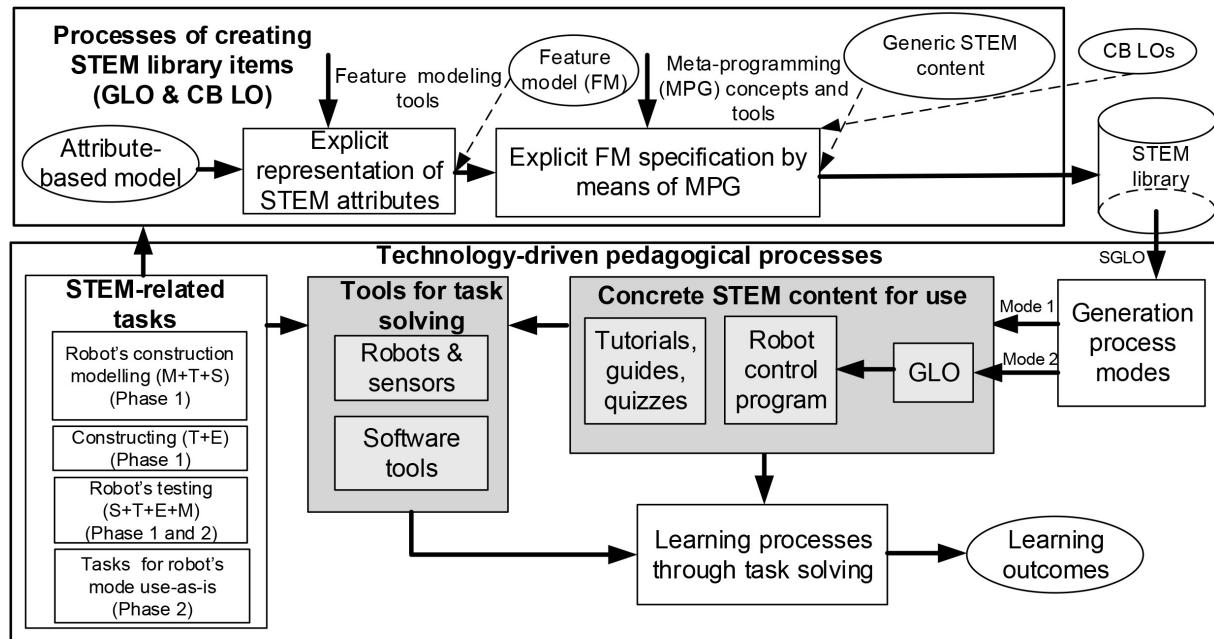


Fig. 3. A process-based view of the STEM-driven model at the implementation level.

- The STEM model has an explicit relation to the learning resources of distinct forms, if not to say more—they are deeply integrated in our model (see Fig. 3). There are hardware resources (robot as a PLO) and software-oriented resources (including SGLOs and SW tools: either of general use and domain-specific, such as modelling tools, robot programming environments).

In Fig. 3, we outline our approach as a process-based vision that is relevant to the STEM paradigm and reveals on how we have implemented it. Here we grouped the processes into two large groups: (1) processes of creating STEM library items; (2) technology-driven pedagogical processes. The first group includes two transformations: (a) the domain attribute-based model into the feature model and (b) the feature model into the meta-programming-based SLO specification [51]. This specification enables the two-level generation process represented as Mode 1 and Mode 2:

$$\begin{aligned}
 &\text{Mode 1} \\
 &\text{SGLO} \Rightarrow \text{GLOs} + \text{CB LOs}; \\
 &\text{Mode 2} \\
 &\text{GLO} \Rightarrow \text{RCP}.
 \end{aligned}
 \tag{2}$$

Why we use the STEM-based RCP not as a single control program, but as a generative one (GLO)? It is so because RCPs, specified as GLO, have many advantages. GLO is a pre-defined family of the RCP instances. They may cover a variety of use cases in designing individual learning paths. It is possible to derive a particular instance from the GLO specifica-

tion automatically on demand. Therefore, we have a great flexibility in time saving and efficiency in use (either in the testing phase or learning process phase in which CS education objectives we are able to fulfill in the full range). We present more details regarding the implementation in our Case Studies in Section 5.

## 5. Case Study 1. Line following: testing of robot's functionality

The aim of this case study is to demonstrate an experiment related to the accuracy evaluation of line following. Students use the robot called “Line follower” and test four line following algorithms such as: *One Inside*, *One Bounce*, *Straddle*, *Two Inside* [52]. The first two algorithms are based on the use of one light sensor in front of the robot that detects the edge of the line. The second two algorithms use two light sensors that are positioned side-by-side inside in case of the *Two Inside* algorithm, or on the either side of the line in case of the *Straddle* algorithm.

By carrying out this research, students use the GLO “Line following algorithms”. Firstly, students choose line following algorithm (see Fig. 4(a)). At the next step, the light sensors’ input and motors’ output ports in the LEGO NXT Intelligent Brick should be defined and velocity of motors should be chosen (Fig. 4(b)). When students select values of all parameters, the robot control program (RCP) is generated automatically, and young researchers transfer it to the RobotC environment (Fig. 4(c)). Then RCP should be compiled and transferred to

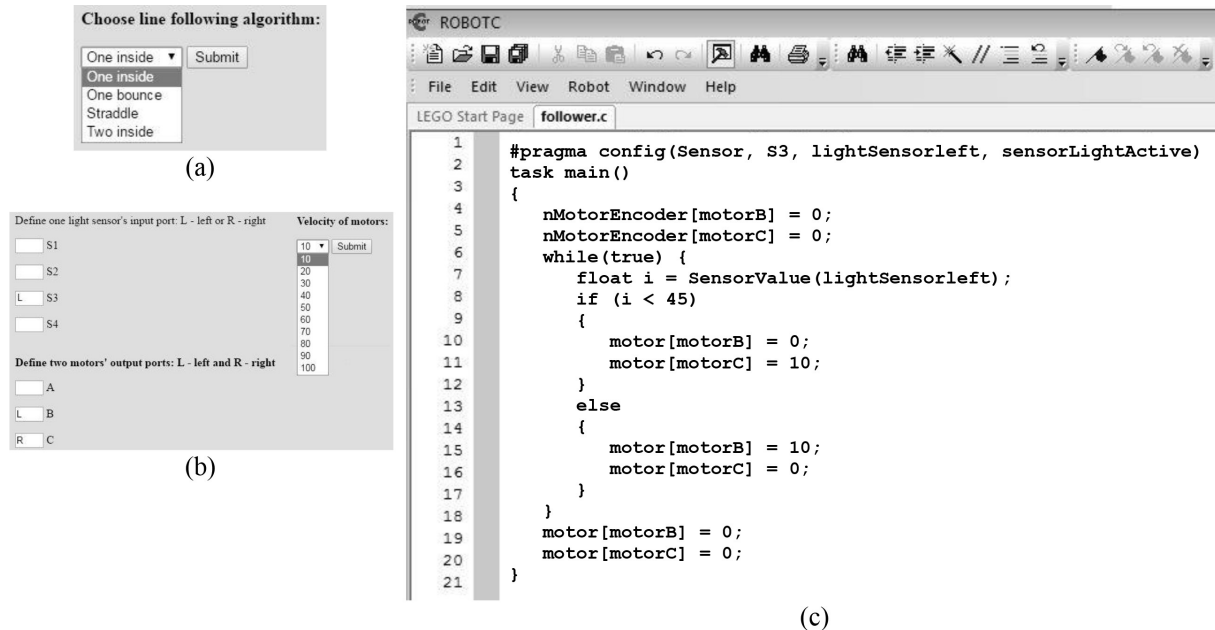


Fig. 4. GLO “Line following algorithms”: top-level interface (a), technological parameter interface (b), generated instance in RobotC (c).

the robot (i.e. to flash memory within the LEGO NXT Intelligent Brick).

Students test the accuracy of line following using the elliptical route (its radii are 21 and 32 cm) for robot movement (see Fig. 5(a)). The dotted line shows a real path of the robot’s movement. Accuracy is calculated by estimating what part of the path the robot overcomes without leaving the black line driving at different speed [53]. Fig. 5(b) presents the experimental results of accuracy testing.

This case study therefore demonstrates solving of the task taken from of robot testing tasks and covers both processes of STEM-driven CS education, i.e. processes of creating the content and technology-driven pedagogical processes (see Fig. 3).

#### Case Study 2. Line following with obstacle

The aim of this case study is to demonstrate the implementation of the robot-based learning enviro-

nement to provide STEM-driven aspects in the course “Programming Basics” for the 10th grade secondary school students. Below we consider the topic *Conditional statements and loops* and present it by SGLO.

In the first step, the user can select a topic and objective from the interface (see Fig. 6(a)). In the next step, the user selects a knowledge level, learning activity and learning content (learning object) (see Fig. 6(b)). Those activities result in representing CB LO presented in Fig. 6(c).

If the user (i.e. learner) selects the knowledge level *Intermediate*, the learning activity *Practice* and GLO *Following the line with an obstacle* (see Fig. 7(a)), the interface based on the multi-menu opens (see Fig. 7(b)). Note that one part of the menu is the same as in Fig. 4(b). Therefore, we do not show it in this case study. Then the student inserts the parameter values according to the task as Fig. 4(b)

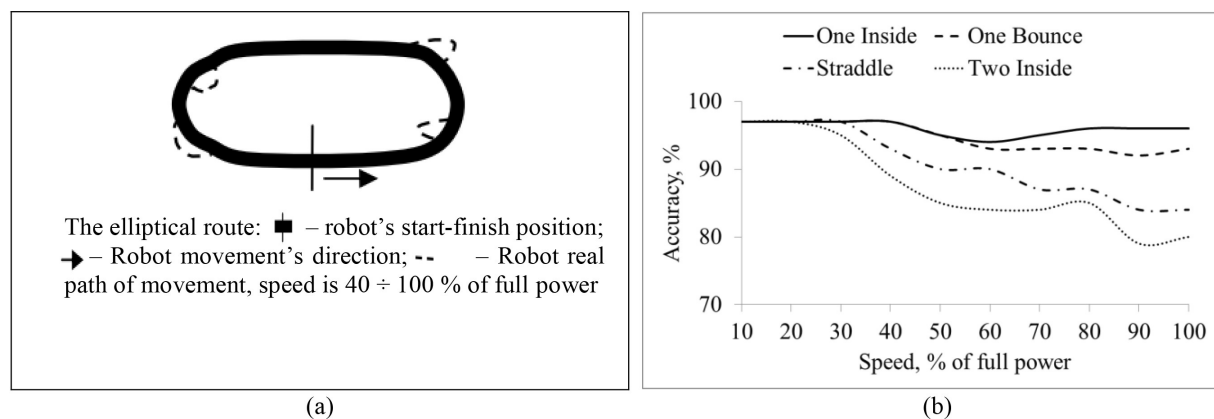


Fig. 5. The elliptical route of line following (a) and accuracy experiment using different algorithms (b).

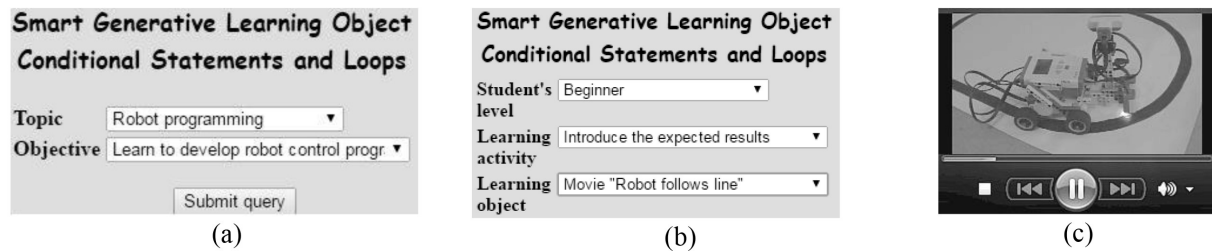


Fig. 6. SGLO “Conditional Statements and Loops”: top interface (a), learning content interface (b), CB LO (c).

shows. The result of the previous activities is the RCP with conditional statements and loops in RobotC (Fig. 7(c) gives its fragment). Then the generated RCP is to be loaded into the robot’s memory and the student is able to monitor its execution in real time. In addition, this result might be quite different from the “motivating movie” depending on the task.

As the main objective is obtaining knowledge in programming, the student modifies the RCP code so that to cover different variants of the task. For example, the student is able to change a minimal distance between an obstacle and the robot, to use another line following algorithm, to change the line’s shape by reconstructing the previously generated RCP. In this way, the student is able to learn more efficiently.

## 6. Pedagogical evaluation of the approach

Note that we have used this approach for four years (2011–2014) implicitly, i.e. without the explicit representation of STEM features. We have introduced those features in the years 2014–2016. Below

we present the evaluation of our approach from the students’ and teacher’s viewpoints.

We have created a questionnaire to assess the students’ opinion on using STEM-driven content, methods and activities in the learning processes. The teacher has developed the questions. The respondents (in total: 80 who used STEM-driven approach during 2014–2015 and 38 who used the same approach during 2015–2016) were secondary school students of the 10th grade (15–16 years old). Table 2 presents the results. For percentage calculating, we take into account the total number of respondents (in the case of multiple answers).

The pedagogical effectiveness of the STEM-driven CS education approach from the teacher’s perspective was evaluated by “engagement levels”, using the methodology proposed by [54]. This methodology covers five levels:

1. *Viewing*. Students are viewing the programs, different tutorials and animations given by the teacher.
2. *Responding*. Students are observing the run of programs (i.e. the robot’s action), and the

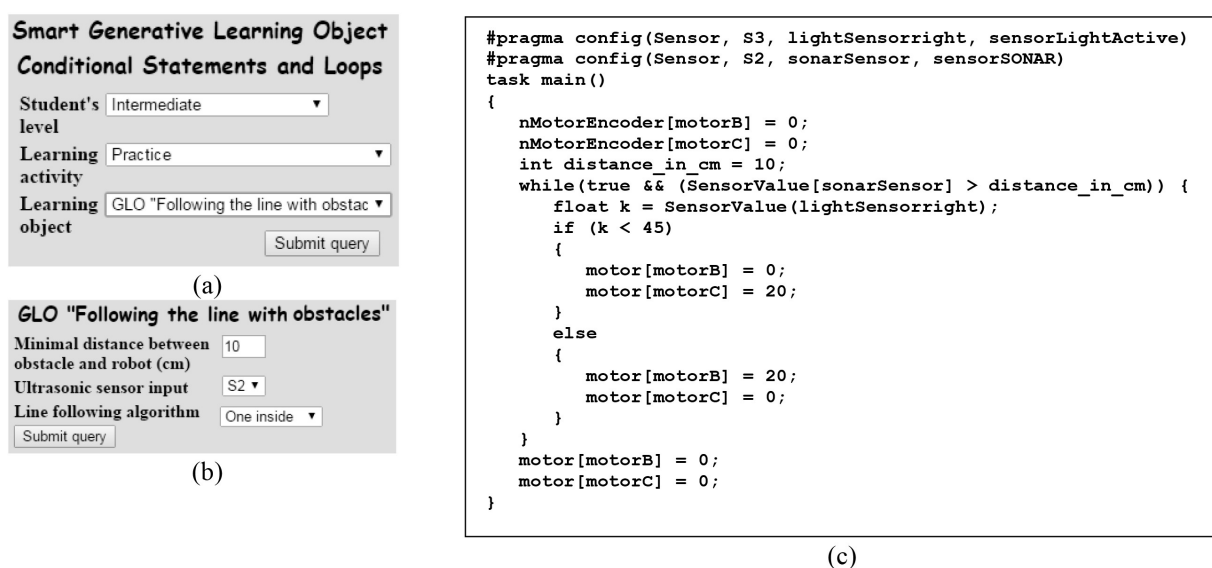


Fig. 7. SGLO “Conditional Statements and Loops”: top interface (a), technological parameter interface (b), generated instance in RobotC (c)

**Table 2.** Results of the students' evaluation of STEM paradigm.*At which extent the methodology was useful? (only one answer possible)*

Answer choice	2014–2015		2015–2016	
	#	%	#	%
Very useful	28	36%	17	45%
Useful	32	40%	21	55%
More useful than non-useful	16	20%	0	0%
More non-useful than useful	4	6%	0	0%
Non-useful	0	0%	0	0%
Totally non-useful	0	0%	0	0%

*What was the most interesting within the methodology? (multiple answers possible)*

Answer choice	2014–2015		2015–2016	
	#	%	#	%
Interesting tasks	48	60%	24	63%
New learning way	39	49%	24	63%
Learning is easier and faster	22	28%	10	26%
Fault-tolerance	16	20%	10	26%
Stimulate thinking	31	39%	17	45%

*What knowledge and competence you were able to improve using STEM-driven approach? (multiple answers)*

Answer choice	2014–2015		2015–2016	
	#	%	#	%
Programming	58	73%	31	82%
Mathematics	27	34%	24	63%
Logic thinking and cognition	36	45%	33	87%
The practical evidence on tasks solving	8	10%	4	11%

*Where and when the use of STEM-driven approach should be targeted? (multiple answers possible)*

Answer choice	2014–2015		2015–2016	
	#	%	#	%
Always in each lesson on programming	30	38%	10	26%
In other courses (mathematics, physics)	21	26%	14	37%
Sometimes for lesson variation	51	64%	24	63%
For generalizing the topic	9	11%	4	11%

*For what student's abilities the use of STEM-driven approach fits best? (multiple answers possible)*

Answer choice	2014–2015		2015–2016	
	#	%	#	%
Low abilities	12	15%	11	29%
Adequate abilities	46	58%	21	55%
High abilities	56	70%	35	92%
Very high abilities	34	43%	17	45%

process is the resource for taking and answering questions given by the teacher.

3. *Changing*. Students themselves are modifying programs by changing the parameter values of the GLO; therefore, they are acting as designers of LOs.
4. *Constructing*. Students are constructing their own programs, so they are becoming the LO co-designers and testers.
5. *Presenting*. Students are presenting their results to the audience for discussion.

We calculated this evaluation based on our two-year experience in using educational robotics in CS education. Table 3 summarizes results of this evaluation.

We can conclude that students treat the metho-

dology as a useful means for their active learning, because the methodology supports to some extent the interdisciplinary aspects. The results showed that, from the teacher's perspective, the STEM-driven CS education approach is most effective: (1) in constructing and presenting levels for all students and boys; (2) in all levels for girls. The increase in performance was 10–17 % for girls (especially in the constructing level).

## 7. Discussion

In this paper, we have presented a vision on how it is possible to introduce STEM into Computer Science (CS) education at high school. At the centre of our approach is the robot-based educational environment and STEM-driven resources for this environ-

**Table 3.** Results of the “engagement levels” evaluation during 2014–2016 (students of the 10th grade (15–16 years old): 118 students, 90 boys, 28 girls).

Engagement level	All students		Boys		Girls	
	CS	STEM-driven CS	CS	STEM-driven CS	CS	STEM-driven CS
<i>Viewing</i>	90% (106)	100% (118)	91% (81)	100% (90)	88% (25)	100% (28)
<i>Responding</i>	82% (97)	86% (101)	82% (74)	84% (75)	82% (23)	92% (26)
<i>Changing</i>	63% (74)	67% (79)	67% (60)	69% (62)	51% (14)	62% (17)
<i>Constructing</i>	24% (28)	35% (41)	26% (23)	34% (31)	18% (5)	37% (10)
<i>Presenting</i>	15% (18)	22% (26)	15% (14)	21% (19)	15% (4)	25% (7)

ment. The resources include items that integrate the CS-oriented content (an essential part of it is the robot control programs in a generic format) along with pedagogical approaches adapted for STEM. We have represented this integration in a specific form called Smart Generative Learning Object. This format is an essential modification of Smart Learning Objects (SLOs) described in [15] in two aspects: (1) we have enriched semantics by introducing STEM features for both the content and pedagogy; (2) we have changed the internal structure of SLO by separating pedagogical parameters from content parameters and representing them at different levels aiming at achieving a higher flexibility. We have presented our vision from the perspective of an external viewer, i.e. we have discussed a process-based model (as a derivative product from the conceptual model) and outlined its implementation by analysing two Case Studies. We have also discussed students’ evaluation by a questionnaire and teacher’s evaluation using the *engagement levels-based* methodology [54]. We obtained that the use of our approach was more effective for girls as compared to pure CS education.

Our approach has multiple aspects that we consider as a novelty. (1) It focuses on the wide-scale analysis as a context to build the robot-based environment for STEM-driven CS education at high school. (2) It covers the full life-cycle processes ranging from constructing/testing of the robot itself to the development and use of robot’s control programs for the CS education. (3) The STEM knowledge is explicitly integrated within the processes and content and therefore can be easily extracted to present for learning. (4) When implemented, our approach exploits the advanced technologies such as feature-based modeling and meta-programming, enabling to introduce systemization and generalization and to achieve a higher extent of reuse through automation, though those aspects in the paper are presented as a by-product only.

Therefore, here we are able to evaluate our approach from methodological and pedagogical perspectives, assuming that the technological aspects are highly influential for both. Indeed, the technology (i.e. meta-programming combined with

feature-based modelling) we use for implementing the approach enables: (i) to generalize the content and represent it as a family of related variants to support reusability, modifiability and adaptability; (ii) to pre-design the generic content in advance, to test it and load it to the STEM library for multiple use and re-use; (iii) to systemize the content design activities.

From the methodological perspective, educational robotics and STEM are two sides of the same coin. However, in order to achieve a full integration in creating the coherent STEM-driven educational environment, one needs to go a long way. Firstly, we need to clearly understand what separate constituents (i.e. robot, STEM content for CS and STEM pedagogy) are from the much broader perspective. It is not enough to use the robot in the mode *use-as-is*. There is to be created the possibility to pass by student the full life-cycle of the robot functionality (from researching its components and units to modeling, constructing, testing and solving prototypes of the real-world tasks). On the other hand, the STEM content is to be well-defined (in terms of objectives and topics) before being transformed and integrated into the environment. The content designer and/or teacher is responsible for that. The relevant topics are to be associated with STEM features explicitly to make easier the integration and use later. All those activities require a great deal of analysis. In our model, we have considered that as a context or as a by-product only. Therefore, we have generalized the methodological issues of that part through the concept of the STEM library and its entities Smart Generative Learning Objects (SGLOs) in our model.

The remaining part of the model relates to technology-driven pedagogical processes that appear in the real educational setting. They include a series of activities such as task solving, outcomes evaluating on-the-fly, task modifying and repeating its solving. The model was implemented and used first without introducing the STEM paradigm explicitly. Later STEM features were integrated and exploited in the real setting to provide CS education. As the STEM pedagogy (i.e. pedagogical

approaches relevant to STEM) is seamlessly integrated with STEM content, the teacher is able to reveal the possibilities of this paradigm to students in multiple contexts. Students are able to select and use the content so that it would be possible to respond to pre-defined objectives (i.e. one representation of the content more reflects the CS knowledge, such as algorithms, while the other better suits to provide the engineering knowledge, such as constructing of the robot). The flexibility of choice is embodied into the menu-driven content structure we call the SGLOs. Therefore, learners are able to create different learning paths predefined by those technological capabilities, pedagogical models used (project-based, problem-based, inquiry-based, game-based, etc.) and individual preferences.

So far, we have discussed the properties and capabilities of the proposed approach that basically reflect advantages. What are disadvantages of our approach? From the teacher's perspective, the approach is complex if one focuses on design processes. It is convenient to evaluate the approach from two perspectives (the design of the educational environment and its use). The design requires the knowledge and competence that far more exceed the level outlined in the CS school curriculum. Not each CS teacher, therefore can stand in two roles at once, i.e. act as the designer and as the service provider. The environment is indeed the intellectual property of a high value. It should not be shared for free. Therefore, at the moment the experience of using the approach is restricted by one high school and by one teacher, though the approach was used for more than 4 years without the explicit STEM and about two years with the STEM paradigm. What disadvantages may see students? At the very beginning, for many students, it is difficult to transform the previous knowledge in the new context in terms of use, representation, and application. Some activities, such as modeling of the robot's behavior, are not achievable for all students. The same is with scientific experiments. The only students with a higher self-motivation make choices to do that. Therefore, it is very difficult to maintain the equity for all due to the previous knowledge level, communication skills, self-motivation, etc. Therefore, some students may feel a dissatisfaction with the learning outcomes.

## 8. Conclusion and future work

As the educational robots are programmable entities having both the mechanical and computational capabilities, they fit well to provide STEM-driven education in CS, though the introduction of that in reality requires many efforts, resources and adequate approaches. One approach we have discussed

in this paper. The proposed approach covers both a vision and process-based model that outlines on how it is possible to implement the approach in practice and provide the adequate evaluation. The essence of our approach is the *seamless integration* of the essential attributes of the STEM-driven and CS-oriented content with the STEM pedagogy features. We argue that the use of meta-programming as the implementation technology enables to achieve this integration though the pre-designed Smart Generative Learning Objects. Though our focus was mainly on CS education, we argue that our approach is partially applicable for other school courses, such as physics and mathematics. The collected data, taken from the educational practice at one high school over 5 years, shows effectiveness of the proposed approach. We have obtained that our approach was more influential for girls as compared to pure CS education. We hope also that our approach is applicable for college and university education, especially in engineering education.

The future work includes providing more experiments, collecting of more data, especially from the pedagogical perspective, for more extensive evaluation of the approach. In addition, we plan to introduce the remote mode of our environment that students could be able to work at home or other places.

## References

1. Southwest Pennsylvania STEM network long range plan (2009–2018): Plan summary, <http://business-leadership-coaching.com/wp-content/uploads/2013/08/SWP-STEM-STRATEGY-Final-Report-Summary-July-2009.pdf>, Accessed 31 October 2016.
2. J. Ardies, S. De Maeyer, D. Gijbels and H. van Keulen, Students attitudes towards technology, *International Journal of Technology and Design Education*, **25**(1), 2015, pp. 43–65.
3. K. Ohkuma, M. Osogami, N. Shiori and K. Sugihara, Motivation Effects of Using Actual Robots Controlled by the Scratch Programming Language in Introductory Programming Courses, *International Journal of Engineering Education*, **33**(2A), 2017, pp. 575–587.
4. N. Arshavsky, J. Edmunds, K. Mooney, B. Thrift, L. Wynn, S. Center, K. Samonte and L. Janda, Race to the Top STEM Affinity Network, 2014.
5. C. Robertson, Restructuring High School Science Curriculum: A Program Evaluation, 2015.
6. B. C. Gamse, A. Martinez, L. Bozzi and H. Didriksen, Defining a research agenda for STEM Corps: Working white paper: Cambridge, MA: Abt Associates, 2014.
7. M. Duran, M. Höft, D. B. Lawson, B. Medjahed and E. A. Orady, Urban high school students' IT/STEM learning: Findings from a collaborative inquiry-and design-based afterschool program, *Journal of Science Education and Technology*, **23**(1), 2014, pp. 116–137.
8. J. Dutta-Moscato, V. Gopalakrishnan, M. T. Lotze and M. J. Becich, Creating a pipeline of talent for informatics: STEM initiative for high school students in computer science, biology, and biomedical informatics, *Journal of pathology informatics*, **5**, 2014.
9. S. Holmquist, A multi-case study of student interactions with

- educational robots and impact on Science, Technology, Engineering, and Math (STEM) learning and attitudes, 2014.
10. T. H. Nelson, K. Lesseig and D. Slavit, Making Sense of “STEM Education” in K-12 Context.
  11. B. Freeman, S. Marginson and R. Tytler, 1 Widening and deepening the STEM effect, *The Age of STEM: Educational Policy and Practice Across the World in Science, Technology, Engineering and Mathematics*, 2014, p. 1.
  12. N. W. Sochacka, K. Guyotte and J. Walther, Learning Together: A Collaborative Autoethnographic Exploration of STEAM (STEM+ the Arts) Education, *Journal of Engineering Education*, **105**(1), 2016, pp. 15–42.
  13. N. Mentzer, K. Becker and M. Sutton, Engineering design thinking: High school students’ performance and knowledge, *Journal of Engineering Education*, **104**(4), 2015, pp. 417–432.
  14. W. Gander, Informatics—New Basic Subject, *Bulletin of EATCS*, **2**(116), 2015.
  15. V. Štūkys, *Smart Learning Objects for Smart Education in Computer Science: Theory, Methodology and Robot-Based Implementation*: Springer, 2015.
  16. D. K. DiGiacomo and K. D. Gutiérrez, Relational equity as a design tool within making and tinkering activities, *Mind, Culture, and Activity*, **23**(2), 2016, pp. 141–153.
  17. P. Edgert, C. Gallagher, K. Huang and M. Joseph Van Matre, STEM Learning Opportunities Providing Equity: An Investing in Innovation (i3) Grant Final Evaluation Report, 2015.
  18. S. Chachashvili-Bolotin, M. Milner-Bolotin and S. Lissitsa, Examination of factors predicting secondary students’ interest in tertiary STEM education, *International Journal of Science Education*, **38**(3), 2016, pp. 366–390.
  19. I. Zeid, J. Chin, C. Duggan and S. Kamarthi, Engineering based learning: a paradigm shift for high school STEM teaching, *International Journal of Engineering Education*, **30**(4), 2014, pp. 867–887.
  20. V. Bimpikas, G. Hloupis, I. Stavarakas, K. Moutzouris, C. Stergiopoulos and D. Triantis, Developing open source dataloggers for inquiry learning, 2015.
  21. J. Earp, Game making for learning: A systematic review of the research literature, *Proceedings of 8th International Conference of Education, Research and Innovation (ICERI2015)*, 2015, pp. 6426–6435.
  22. J. Jensen and M. Droumeva, Exploring Media Literacy and Computational Thinking: A Game Maker Curriculum Study, *Electronic Journal of e-Learning*, **14**(2), 2016.
  23. F. B.V. Benitti, Exploring the educational potential of robotics in schools: A systematic review, *Computers & Education*, **58**(3), 2012, pp. 978–988.
  24. A. Eguchi, Robotics as a learning tool for educational transformation, *Proceeding of 4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education Padova (Italy)*, 2014.
  25. A. Eguchi, Educational robotics for promoting 21st century skills, *Journal of Automation Mobile Robotics and Intelligent Systems*, **8**(1), 2014, pp. 5–11.
  26. S. Atmatzidou and S. Demetriadis, How to Support Students’ Computational Thinking Skills in Educational Robotics Activities.
  27. F. R. Sullivan and J. Heffernan, Robotic Construction Kits as Computational Manipulatives for Learning in the STEM Disciplines, *Journal of Research on Technology in Education*, **48**(2), 2016, pp. 105–128.
  28. A. Eguchi, Computational Thinking with Educational Robotics, *Society for Information Technology & Teacher Education International Conference*, 2016, pp. 79–84.
  29. B. S. Fagin and L. Merkle, Quantitative analysis of the effects of robots on introductory Computer Science education, *Journal on Educational Resources in Computing (JERIC)*, **2**(4), 2002, p. 2.
  30. D. Alimisis, M. Moro, J. Arlegui, A. Pina, S. Frangou and K. Papanikolaou, Robotics & constructivism in education: The TERECoP project, *EuroLogo*, 2007, pp. 19–24.
  31. S. Frangou, K. Papanikolaou, L. Aravecchia, L. Montel, S. Ionita, J. Arlegui, A. Pina, E. Menegatti, M. Moro and N. Fava, Representative examples of implementing educational robotics in school based on the constructivist approach, *SIMPAP Workshop on Teaching with robotics: didactic approaches and experiences, Venice, Italy*, 2008.
  32. S. Kurebayashi, S. Kanemune, T. Kamada and Y. Kuno, The effect of learning programming with autonomous robots for elementary school students, *11th European Logo Conference*, 2007, p. 46.
  33. J. Zalewski, Creating Research Opportunities with Robotics across the Undergraduate STEM Curricula, *age*, **24**, p. 1.
  34. A. Eguchi, Educational Robotics as a Learning Tool for Promoting Rich Environments for Active Learning (REALs), *Handbook of Research on Educational Technology Integration and Active Learning*, 2015, p. 19.
  35. R. Gerndt and J. Lüssem, Mixed-reality robotics—a coherent teaching framework, *Proceedings of 2nd international conference on robotics in education (RiE)*, 2011, pp. 193–200.
  36. L. M. Grabowski and P. Brazier, Robots, recruitment, and retention: Broadening participation through CS0, *2011 Frontiers in Education Conference (FIE)*, 2011, pp. F4H-1–F4H-5.
  37. D. Catlin, A. P. Csizmadia, J. G. OMeara and S. Younie, Using Educational Robotics Research to Transform the Classroom, *RiE 2015: 6th International Conference on Robotics in Education*, 2015.
  38. R. Burbaitė, V. Štūkys and R. Marcinkevičius, The LEGO NXT robot-based e-learning environment to teach computer science topics, *Elektronika ir Elektrotechnika*, **18**(9), 2012, pp. 113–116.
  39. J. D. Weingarten, D. E. Koditschek, H. Komsuoglu and C. Massey, Robotics as the delivery vehicle: A contextualized, social, self paced, engineering education for life-long learners, 2007.
  40. S. H. Kim and J. W. Jeon, Introduction for Freshmen to Embedded Systems Using LEGO Mindstorms, *IEEE Transactions on Education*, **52**(1), 2009, pp. 99–108.
  41. J. S. Kay, J. G. Moss, S. Engelman and T. McKlin, Sneaking in through the back door: introducing K-12 teachers to robot programming, *Proceedings of the 45th ACM technical symposium on Computer science education*, 2014, pp. 499–504.
  42. E. Sklar, S. Parsons and M. Azhar, Robotics Across the Curriculum, *AAAI Spring Symposium: Semantic Scientific Knowledge Integration*, 2007, pp. 142–147.
  43. A. Misirli and V. Komis, Robotics and programming concepts in early childhood education: A conceptual framework for designing educational scenarios, *Research on e-Learning and ICT in Education*: Springer, 2014, pp. 99–118.
  44. K. Highfield, Stepping Into STEM With Young Children: Simple Robotics and Programming as Catalysts for Early Learning, *Technology and Digital Media in the Early Years: Tools for Teaching and Learning*, **150**, 2014.
  45. R. Burbaitė, R. Damaševičius and V. Štūkys, Using robots as learning objects for teaching computer science, *X world conference on computers in education (WCCE’13)*, Torun, Poland, 2013, pp. 103–111.
  46. M. E. Karim and F. Mondada, A review: Can robots reshape K-12 STEM education?, *2015 IEEE International Workshop on Advanced Robotics and its Social Impacts (ARSO)*, 2015, pp. 1–8.
  47. K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak and A. S. Peterson, Feature-oriented domain analysis (FODA) feasibility study: DTIC Document, 1990.
  48. K. Czarnecki and S. Helsen, Feature-based survey of model transformation approaches, *IBM Systems Journal*, **45**(3), 2006, pp. 621–645.
  49. D. Batory, Feature models, grammars, and propositional formulas, *International Conference on Software Product Lines*, 2005, pp. 7–20.
  50. T. R. Kelley and J. G. Knowles, A conceptual framework for integrated STEM education, *International Journal of STEM Education*, **3**(1), 2016, pp. 1–11.
  51. V. Štūkys, R. Burbaitė, K. Bespalova and G. Ziberkas, Model-driven processes and tools to design robot-based generative learning objects for computer science education, *Science of Computer Programming*, 2016.
  52. J. Gray, Toeing the Line: Experiments with Linefollowing Algorithms: Technical Report, 2003.

53. R. Burbaite, V. Štuikys and R. Damasevicius, Educational robots as collaborative learning objects for teaching Computer Science, *System Science and Engineering (ICSSE)*, 2013 International Conference on, 2013, pp. 211–216.
54. J. Urquiza-Fuentes and J. A. Velázquez-Iturbide, Pedagogical effectiveness of engagement levels—a survey of successful experiences, *Electronic Notes in Theoretical Computer Science*, **224**, 2009, pp. 169–178.

**Table of acronyms used in this paper**

Acronym	Meaning	Extended Explanation
CS	Computer Science	Widely used abbreviation
STEM	Science (S) Technology (T) Engineering (E) Mathematics (M)	Commonly accepted term
LO	Learning Object	Common term to define the teaching/learning content
PLO	Physical Learning Object	A particular LO such as robot itself
LED	Light Emission Diodes	Electronics component
LCD	Liquid Cristal Display	Electronics component
RCP	Robot Control Program	Entity to control robot's actions
FODA	Feature-Oriented Domain Analysis	A method to analyse domains
GLO	Generative learning object	Accepted term introduced by Boyle et al. It generates LO instances
SGLO	Smart GLO	GLO with extended functionality
CB LO	Component-Based Learning Objects	LO instances such as separate components: tutorials, guides, quizzes, movies, slides, etc.
SLO	Smart Learning Object, i.e. a short name of SGLO	Entity having generative and other important properties
MPG	Meta-programming	A kind of generative technology to implement GLO and SLO

**Prof. Vytautas Štuikys** currently is a researcher at Software Engineering Department of Kaunas University of Technology. He is a PhD holder in Software Engineering since 1970 and doctor *habilitatus* since 2003. He is a co-author of the monograph “Meta-programming and Model-Driven Meta-Program Development” (2013) and author of the monograph “Smart Learning Objects for Smart Education in Computer Science” (2015) published by Springer. His research interests include model-driven system development, including those for STEM education.

**Dr. Renata Burbaite** is currently a lecturer at Software Engineering Department of Kaunas University of Technology and Informatics teacher at the High School. She holds a PhD degree in Computer Science since 2014. Her research interests include the CS educational domain modeling and STEM-oriented robot-based CS education methodologies. She is a co-author of two Chapters of the monograph “Smart Learning Objects for Smart Education in Computer Science” (2015).

**Dr. Tomas Blažauskas** is an Associate Professor and head of Software Engineering Department at Kaunas University of Technology. He holds PhD in Software Engineering from Kaunas University of Technology since 2003. His current research interests include e-learning systems, software engineering systems, natural user interfaces. He is an author of two e-learning platforms, which are available for public use.

**Dr. Dominykas Barisas** holds a PhD in Software Engineering from Kaunas University of Technology since 2012 and presently is a lecturer. Latest research is related to domain-specific languages, e-learning systems, gamification, user behavior evaluation, data collection and analytics.

**Dr. Mikas Binkis** holds a PhD in Software Engineering from Kaunas University of Technology since 2012. Currently he is a lecturer. His research interests include gamification, e-learning systems, domain independent modeling, domain-specific languages, and script based software design methods.