Hindawi
Scientific Programming
Volume 2020, Article ID 6807515, 19 pages
https://doi.org/10.1155/2020/6807515

Research Article

Methodological Guidelines for the Design and Integration of Software Learning Objects for Scientific Programming Education

Vladimiras Dolgopolovas, Valentina Dagienė, and Tatjana Jevsikova

Vilnius University Institute of Data Science and Digital Technologies, Akademijos 4, Vilnius 04812, Lithuania

Correspondence should be addressed to Vladimiras Dolgopolovas; vladimiras.dolgopolovas@mif.vu.lt

Received 11 January 2020; Revised 20 May 2020; Accepted 11 June 2020; Published 1 July 2020

Academic Editor: Autilia Vitiello

The motivation for the research is the need to develop an integrated and holistic approach to fostering students' scientific inquiry based on scientific programming education by conducting computational experiments and simulations. At the same time, the implementation of the learner-centred approaches to scientific programming education and the related development of science, technology, engineering, and mathematics (STEM) learner-centred educational environment are of primary importance for K-16 education. The key interest is how to design and integrate learning resources which include software learning objects for making simulations. The research investigates educational aspects of the technological, pedagogical, and content knowledge (TPACK) framework applied to scientific computing and scientific programming educational domain and provides methodological guidelines and design principles of practical implementation of educational resources. These include design principles for the development of the model-based scientific inquiry-centred learning resources, generic design templates for designing educational aspects of scientific programming education, generic use case models for learning resources for scientific programming education, and supportive methodological considerations.

1. Introduction and Problem Statement

Diverse and interdisciplinary research is now a priority for highly ranked educational institutions. An important question is whether it is possible to force interdisciplinarity without serious improvements of the classical approach to and style of university education. Obviously, there is a need for new approaches which enable teaching interdisciplinarity and its applications in an effective and learner-centred way. Therefore, it is important to answer the following questions. What is a unifying paradigm for interdisciplinary university education? What is the learner-centred methodology for scientific programming (SP) education? One of the solutions is that such new approaches—and this is especially important for science, technology, engineering, and mathematics (STEM) educational environment—could be based on the positioning of SP as a unifying discipline for STEM education. This is an important and at the same time a challenging task, as it requires the proper organization of educational settings and design of educational materials in such a way that enables students' motivation, seamless approaches to instruction, simulation-making-based pedagogy, and fostering student's scientific inquiry educational methods.

An overview of the general research problems related to SP and STEM education can be formulated as follows. What is the generalized methodological framework for the design and implementation of the relevant teaching and learning methods for scientific programming education (SPE), which covers such major parts of the educational system like educational technology, instructional design, and didactic tools? Focusing on general aspects of educational technology, what is the methodology for designing a university curriculum in general and a STEM university curriculum in particular, focusing on interdisciplinary and research-based education? How can SP as a discipline be integrated into the curriculum? Focusing on aspects of instructional design, how the teaching and learning process for SPE should be designed, enabling scientific inquiry- (SI-) and constructionist-based education? Focusing on didactic aspects, how should the learning resources (LR) for SPE be designed?

What are the appropriate computational models and algorithms that enable the relevant implementation of LR for students' simulation-making activities including the principles of design of software learning objects (SLOs) for making simulations? How can educational tools be developed in a way that enables simulation-making and SI-centred approach to the process of teaching and learning and promotes practical knowledge of the relevant parallelization methods, including hybrid computational platforms related to big data? What are the practical examples of implementations? The article presents an approach that covers some of these problems, providing examples of solutions and implementation of relevant educational resources.

This study continues the previous works of the authors [1, 2] and presents methodological generalizations that enable the creation of an SI-centred educational environment for scientific computing and scientific programming education.

2. Research Methods

Design science research (DSR) [3–5] is used as a research methodology for design and implementation of learning resources (LR) presented in this research. There are several major problems to be solved: (A) There is a need to specify SP courses: SP courses (1) are too technical, (2) are mainly oriented to mathematical and algorithmic foundations, (3) require many prerequisites, (4) cannot easily motivate students. (B) There is a need to specify SP educational technology: (1) there is no vision of suitable educational technology (including the lack of solutions of how to develop the content and teach SP); (2) there is no vision of how to integrate SP into a broader curriculum.

Motivation is based on various opinions and literature review. The origins of problems with the present approaches to SPE are (A) improper didactic approaches and teaching techniques to SPE and (B) the lack of modern, interdisciplinary, and student-centred research-based education approaches to SPE.

> Tentative solutions: (1) SPE should be based on teaching SI; (2) SI is understood (in a broader sense) as a research activity based on scientific simulations (including computer simulations); (3) scientific simulations are based on computer models (artefacts); (4) SPE is based on teaching of how to develop scientific simulations; (5) simulation-based education is based on simulation-based cognitive reasoning processes; (6) learner-centred educational technologies like co-mediated learning should be used.
>
> Artefact to design: the aim of this study is to provide methodological guidelines (MG) and design principles (DP) of how to design and integrate proper LR and learning objects (LO) for simulation-based SPE. The general requirements for the methodology are (1) providing solutions for SI and simulation-based SPE (as it was previously described) and (2) being suitable for use within a learner-centred educational environment (supporting co-mediated learning technology and the constructionist approach to learning).

2.1. Implemented Research Methodology. The research by itself is based on well-known and well-described methodology, DSR. Although the primary application of DSR is information system (IS) design, the methodology has migrated into various fields like education technology and business management. The common feature of the described applications is the following sociotechnical domains. Not only technical aspects of the system but also social aspects and interactions should be considered. The relevance of the proposed methodology to the computer science (CS) educational domain is based on the next propositions. The domain: (1) has a type of a sociotechnical system; (2) includes participants (teachers, students, educational authorities, community, other stakeholders) as well as educational technology, instructional design methods, and educational tools; (3) includes educational tools, mainly artefacts including cognitive artefacts in the form of software-based LO (software as a learning object, educational software [6]). The research methodology specifies the general approach to the research procedures, as specific methods are defined by the topic and the scope of the research.

2.2. Design Science Research Methodological Formalization. DSR methodological formalization is based on a number of heuristics [7]. This set of heuristics is incorporated into the design framework. This approach provides a systematic methodology for conducting research. To summarize, we could conclude that DSR as a methodology provides all the necessary formalization for the implementation of the research task—DP for development of SLO for SPE. The DSR design cycle [3] provides the relevant formalization for utilization of inductive-abductive-deductive reasoning approach for conduction of research.

2.3. Methodological Requirements and Implementation. The summary of the characteristics, the DSR formal requirements, and the relevant implementation is presented in Table 1.

3. Summary of Results and Implementation

3.1. Application Domain. The application domain hierarchy consists of a set of related subdomains within the general domain of university educational system: university educational system; e-learning; STEM university curriculum; interdisciplinary university curriculum; scientific computing education (discipline); scientific programming education (discipline); simulation and modelling with computers. The research question (design requirements) is MG for SLO design, which supports the relevant constructionist methodology for SPE. The MG should provide possibility of the constructionist approach to interdisciplinary university education in general and provide formalization for SPE in particular: from the perspectives of educational technology; from the perspectives of instructional design; from the perspectives of educational content (EC) design.

Table 1: Formal requirements and implementation in the research.

Characteristics	DSR requirements	Implementation
Research objectives	Developing artefacts that enable satisfactory solutions to practical problems	Developing methodological guidelines for design and integration of LR for SPE
Main activities	Defining the problem; suggesting; developing; evaluating; concluding	Problems in SPE: There is no documented LR MG for SPE Suggestion (research task): To develop LR MG for SPE based on constructionist paradigm To develop practical examples of LR for SPE using the provided methodology Evaluation: Ex-ante evaluation (comprehensive literature review, logical reasoning, assertion) Ex-post evaluation (demonstrations with prototypes, case studies) Conclusions: The LR MG is developed. The developed MG implements the constructionist paradigm Ex-ante and ex-post evaluations are fulfilled Examples of LR and case studies are presented
Results	Artefacts (constructs, models, methods, DP, instantiations) and improvement of theories	Constructionist LR MG for SPE Samples of LR for SPE and case studies
Type of knowledge	Prescriptive	How to design LR enhancing interdisciplinarily and seamless approach to SPE
Researcher's role	Builder and/or evaluator of the artefact	The scope of the research: analysis of the current state in the field of SPE; logical analysis, theoretical assertions, and confirmations of psychological, technological, and instructional relevance and theoretical basics for the implemented LR MG; developing the methodology; developing sample LR; demonstrations via case studies
Empirical basis	Not mandatory	Not implemented
Evaluation of results	Applications; simulations; experiment	Evaluation is based on ex-ante and ex-post evaluations methodology implementing DSR requirements
Approach	Qualitative and/or quantitative	Qualitative approach is used, including theoretical analysis and literature review
Specificity	Generalizable to a certain class of problems	Is generalizable to problems of developing interdisciplinary curricula for inquiry-based university education

3.2. Description of the Research Results. Formalization for the developed methodological guidelines is provided. This formalization includes implementation guidelines, recommendations, designing criteria, a set of heuristics, case studies, and practical examples for scientific programming education related domains: (1) educational technology domain; (2) instructional design techniques; (3) development of LR; (4) implementation of SLOs for SPE.

From pedagogical perspectives, the innovative methodology of designing interdisciplinary STEM curriculum is proposed. The methodology is based on the proposition of the central place of SP within university STEM curricula. This provides the basis for the design of interdisciplinary and innovation- oriented learning environment.

From the perspectives of the educational content design, (1) the DP of EC design are provided; the DP are based on the presented approaches and requirements enabling development of the learning content providing constructionist educational environment and co-mediated learning technology; (2) practical examples of LR including SLOs are designed.

3.3. Grounding with the Supporting Theories. Systematic grounding with supporting theories is presented in Table 2.

3.4. Evaluations and Examples of the Implementations. Evaluation of the research results is based on formal requirements for evaluation procedures under DSR methodology [22–24]. The practical evaluation consists of two major parts: ex-ante evaluation and ex-post evaluation. Ex-ante evaluation is based on a systematic literature review, logical reasoning, and assertion evaluation patterns; ex-post evaluation is based on demonstrations and case studies. Examples of implementation of the developed methodology including SLOs are presented in [1, 2].

3.5. Structure of the Article. The article consists of four main parts: (A) *introduction part* including methodological considerations and research design; (B) *theoretical results part* including sections on a brief on formalization for SLO, a brief on TPACK model for SP educational domain, TPACK pedagogical domain model in detail, and principles for designing introductory content for scientific computing education; (C) *case studies* on practical

Table 2: Grounding with supporting theories.

Name of theory	Contributors	Literature
Pragmatism	Peirce	[8, 9]
Constructivism	Piaget	[10]
Constructionism	Papert	[11, 12]
Model-centred instruction	Gibbons	[13]
Simulative reasoning theory	Johnson-Laird	[14]
Model-based scientific reasoning theory	Nersessian, Magnani, Thagard	[15–19]
Theory of smart learning objects	Štuikys, Damaševičius, Burbaitė	[6, 20]
Theory of design science research	Simon, Hevner, Vaishnavi, Kuechler	[3, 4, 7, 21]

implementation of presented theoretical approach with generalizations, examples, and references to previous research; (D) *summary and conclusions* section and references section.

Section 5 briefly presents the main aspects focusing on the requirements of TPACK model. This section describes requirements for the main parts of the model including technological, content, and pedagogical domains.

Section 6 describes pedagogical aspects of the scientific programming education. DSR is positioned as an instructional tool based on circumscription and abduction reasoning and enabling students' simulation-making activities.

Section 7 provides some practical examples of the design process for model-based and SI-centred LR.

Section 8 presents a methodological approach to the design and integration of educational resources and software learning objects based on real-world scientific problems and related research. This section provides an introduction to the further presented case studies of the implementation of LR for scientific programming education and teaching parallelization which are based on the computational model of queues in series. The first case focuses on the topics of design and implementation of SLOs in the introductory SP course. The next one covers the topics of design and implementation of SLOs for teaching parallelization.

Section 9 systematically develops a content domain model for presented case studies and gives methodological generalizations on aspects of educational content in the form of self-explanatory conceptual maps. As a result, two content models of the presented educational solutions for teaching programming and introductory stochastics are introduced.

Section 10 systematically develops a pedagogical domain model for presented case studies and explores methodological generalizations on pedagogical content aspects in a form of use case models displayed as self-explanatory concept maps. As a result, the two use case models of the presented learning resources for teaching programming and introductory stochastics are introduced.

Summary and conclusions are presented in Section 11 of the article.

4. Research Questions

The main research question could be formulated as follows: how to design and integrate learning resources which include software learning objects aimed at teaching scientific programming focusing on fostering students' scientific inquiry and activities of making simulations.

The goal of the research is to develop methodological guidelines and design principles for design and integration of the relevant learning resources which include software learning objects for scientific inquiry driven scientific programming. For achieving this goal, the research will provide a theoretical case study and practically develop and evaluate a set of sample learning resources, including implementations for instructional design and didactics. The study implements the design science research methodology and aims to develop a prescriptive type of scientific knowledge related to design methods in general.

4.1. Formalization: Software Learning Objects for Scientific Programming Education. This section provides a brief overview of learning objects (LO) in general and software-based LO (SLO) in particular. Why is this important for us? The concept of LO [25], as opposite to pedagogical patterns (PP) [26], focuses on teaching techniques and didactic activities, while PP are more related to educational technology and instruction. Since the focus of this study is on an artefact-based teaching method—simulation (that is made by the students) in the form of computer software—LO can serve as a wrapper for this or as a building block for an instructional unit. As a rule, LO are associated with the content of e-learning, but for the purpose of our study, LO are considered more universal and cover simulation-based activities as well. At the same time, LO can provide versatility for teachers as, depending on the specification of a particular LO, teachers can also use them for the simulation-using type of activities.

4.2. On Classification of Learning Objects. The definition of the learning objects needs to be clarified, as, in spite of extensive literature on the topic, various interpretations of the concept still exist. Churchill [27] proposes the next classification: presentation, practice, simulation, conceptual models, information, and contextual representation objects. We share the next definition of the learning objects by Churchill [27]: "a learning object is a representation designed to afford uses in different educational contexts". For the purpose of this study, two types of the learning objects are of primary importance: simulation learning objects and a conceptual model.

Simulation learning objects are focused on operational aspects, exploration of system functionality, and, what is important to emphasize, development of an appropriate mental model. Another aspect is the interconnection between the simulation and real systems. Even though the accuracy of simulations is high, to gain a complete understanding, the learner should practise with the real system; however, "by the time a learner shifts to the real system, he or she would already have constructed a mental model of the system's functionalities and operational possibilities" [27]. This is especially important in cases when practical interactions with the real system go beyond the mere ability to operate and require a broader understanding of behaviour of the system, or when physical connection to the real system is difficult to ensure for educational purposes.

The next is a conceptual model [27]. It "is a type of a learning object that represents one or more related concepts or ideas, usually in an interactive and visual way. It might be appropriate to think of a conceptual model as a representation of a cognitive resource existing in the mind of a subject matter expert, as a useful conceptual knowledge that aids decision-making, disciplinary problem-solving and discipline-specific thinking." These two models are of primary importance for the purpose of our study. The corresponding combination of these two models is considered as the appropriate solution for construction of the advanced learning objects which represent concepts of scientific computing and scientific programming and allow the implementation of simulations and the practical experiments based on such conceptual models.

4.3. Software Program as a Learning Object. For the purpose of our study, the concept of software as a learning object is important. The main aspects to consider are as follows [20]: the content of computer science and programming education is mainly focused on programming; the abstraction is of high level, hence the difficulty in understanding the program as a SLO; SLO is a highly structured executable specification; SLO has almost unlimited possibilities for change, adaptation, modification, and transfer; SLO could be incorporated into the containing artefacts; the use of SLO in programming teaching allows for flexible creation and self-testing of the acquired knowledge; SLO in programming education "can be also viewed as a tool to provide researching with the nearly unlimited possibility for experimentation in various domains such as design, automation, gamification and many more" [20].

Thus, the roles of the software are twofold. Naturally, the programs are presented within the framework of educational computing. On the other hand, programs could be positioned as learning objects by themselves. Therefore, one of the issues to be studies is as follows: what the difference between these two instances of the software program is and how to practically and efficiently design the piece of a program when considering it as a learning object. Another point of interest is how to design an appropriate learning object within the framework of constructivist and constructionist paradigms.

5. TPACK Model for Scientific Programming Educational Domain

Within the context of this article, TPACK is positioned as a framework for teachers' education focusing on providing a general knowledge and a systematic overview of scientific programming education within the SI-centred educational environment. Programming education is traditionally focused on technical and corresponding technological aspects; however, this should be related to (and interconnected with) pedagogical and instructional aspects, and the manuscript presents this consistently. At the same, it is important to highlight the context including scientific computing and related SI aspects. All together (context, pedagogy and instruction, content, technology) will provide a systematic discourse and a conceptual model for teachers' education within a TPACK framework.

Traditionally, the TPACK model focuses on three major parts: technological, content, and pedagogical domains, moving the context out of the boundaries of the model. However, understanding the importance of including the context in the model itself is arising [28]. A systematic study of the context could provide a set of features for implementation of the design principles for LR. At the same time, the model of context should provide conditions for practical implementation of constructivist students' activities and could be considered as a platform for pragmatist [29–31] approaches to education. Such a platform will provide opportunities for students to conduct a research and test the relevant hypotheses and research questions.

The pedagogical domain model provides a set of universal settings for the educational environment irrespective of the educational content and technology implemented. Thus, the pedagogical domain is within the focus of our theoretical research. As for technological and content domains, these domains are tightly connected to specific educational settings and could be studied using examples and particular implementation cases. In our study, we use a combination of both approaches, ensuring consistent study of the pedagogical domain and providing generalizations and generic templates of the technological and content domains based on the presented case studies.

5.1. Technological Domain Model. The specification of the technological domain is aimed at practically enabling students to perform simulation-making activities by providing the appropriate hardware and software tools. Such simulation-making computations are based on parallelization techniques; therefore, the corresponding specification should provide requirements for the relevant hardware architectures [32, 33]. The next remark should be done: a specific view on teaching with technology is implemented. In the context of this research, teaching with technology is actually transformed into teaching technology, and the technological domain is presented in the content of teaching. Both technological and content knowledge domains intersect forming an educational environment based on the common features of both domains.

The proper specification of a technological domain is one of the most important parts of the specification process. The main aspects of the technological domain for scientific computing education (SCE) are (1) hardware architectures in general; (2) SC and parallelization specific aspects of hardware architectures in particular; (3) software engineering technologies in general, which is important from the point of view of the possible implementation of software development methods in the design process of creating of SLOs; (4) model-based methods of software development, a technology that could be promoted as a basic technology for design and implementation of SLOs.

Parallel computations are based on the next hardware architectures [32, 33]: single instruction, single data (SISD); single instruction, multiple data (SIMD); and multiple instruction, multiple data (MIMD). Multiple instruction, multiple data machines are commonly divided into shared memory and distributed memory architectures. In implementing calculations, high performance computing cluster (HPCC) could be considered as a target platform. Such a platform allows us to study different parallelization techniques and implement shared memory, distributed memory, and hybrid memory solutions. Software development (programming) methods include explicit parallelization tools for distributed memory (Message Passing Interface, MPI) and shared memory (Open Multiprocessing, OpenMP) implementations.

5.2. Content Domain Model. The specification of content domain should be focused on theoretical models, including hypotheses and research problems, as well as appropriate and simulation-making enabling computational models. All these should be transformed into educational resources for students' scientific activities and educational research. As a case study, a model of stochastic recurrence and its applications [1] are presented.

5.3. Pedagogical Domain Model. The specification for pedagogical domain is based on three main parts: educational approaches, instructional design, and didactic aspects of SP education. Within the scope of this article, the main focus is made on practical aspects of instructional design, including the relevant educational tools and design principles for educational resources. How can we introduce computer modelling and simulations and integrate advanced interdisciplinary and innovation-oriented teaching methods? If the focus is on SI and simulations, what are the kinds of didactic approaches that are suitable for teaching SP and at the same time enable an interdisciplinary approach to curriculum design?

6. Developing a TPACK Model: On the Problematic of the Pedagogical Domain

6.1. Teaching Design Science Research for Scientific Programming Education. The presented pedagogical approaches provide us with a way for developing educational technology for SP. The main teaching goal is now shifting from teaching students how scientists do scientific research to teaching them how to do scientific research using simulations and computers. Generally, this approach closely corresponds with the definition of SI as a model-making activity [34, 35]. Attention should be paid to design science research as a methodology for creating innovative artefacts. Therefore, this should be the focus of teaching. In practice, this means a continuing process of improving the simulation first provided to the student. Therefore, the suggested teaching scheme is the following. The model improvement cycle should be merged into the learning environment as an integral part of it. Such learning environment could be named the DSR enabling learning environment [36]. How is it possible to construct such environment? The method of a multifaceted simulation is proposed. Such simulation should be based on multifaceted models, and the teaching process actually becomes a process of "turning the model," enabling various planes of the model to be discovered.

An analogy with floodlighting a multifaceted three-dimensional object with a stationary projector could be provided. Therefore, to understand what the floodlighted object looks like, an activity as turning the object in space should be undertaken. One could achieve full understanding of the external structure of the object only after it is turned and observed all around. To construct such a DSR enabling learning environment is a rather complex and challenging task. First of all, the problem should be transformed into a multifaceted problem. In addition, this is not decomposition, since the decomposition is mainly related to the internal structure, but a process of faceting the initial problem. Solving each of the subproblems should enable, by "turning" the task, further observations. This approach could be observed from an epistemological perspective as well. It allows including serious gaming elements into the learning process, which simulates students' curiosity and motivates students for further studies.

6.2. Outline of the Computational Pedagogy Approach. Computational pedagogy could be mentioned as an example of one possible approach to STEM education, especially for the introductory level [37]. This approach focuses on didactic schemes, general computational thinking skills, and practical abilities to conduct simulations. It could be considered as a kind of a unifying approach for CS and STEM education. Traditionally, a deductive approach is considered as the most applicable for teaching scientific topics. The problem of deductive scheme is that it is considered to be demotivating for students, especially for topics where sufficient theoretical background and a large amount of prior knowledge are required [37].

On the contrary, an inductive approach is promoted as such an alternative that improves students' positive attitude to learning science and "first presents students with a problem, a case, or data from an experiment" [38]. As the culminating step, students are led to acquire on their own an understanding of the underlying concept or organizing principle [39]. Inquiry-guided learning, problem-based learning, and project-based learning are all among the forms of inductive instruction [40]. While empirical evidence suggests that the inductive approach to instruction is

superior and that it fosters greater intellectual growth [39], educators should take advantage of different approaches of teaching [38]. To overcome the described problems of the deductive approach, "modelling and simulation-based computational pedagogy" [38] is offered.

6.3. Problems of the Presented Computational Pedagogy Approach. First, from the educational perspective, there is a big difference between simulation-making and simulation-using activities [41]. Moreover, there are many doubts about the effectiveness of the simulation-using tools in enhancing learner's motivation and their conceptual understanding of scientific topics [42]. The described problems will be covered in more detail. Generally, the practical implementation of a simulation in the form of software could be considered as an example of a cognitive artefact. Such an artefact could be considered from different points of view: a person view and a system view. Norman provides the following illustrative example [43]. Consider a "to do" list. Such a checklist, for example, developed for aircraft pilots, from the system point of view, improves pilots' cognitive abilities by enhancing the memory of pilots. Therefore, from the system perspective, this is a memory enhancer. From the pilot perspective or from the person view, using a list is just another task requiring a different type of activity. Without a list, a person should remember all the "to do" tasks. For using the list, the next tasks should be performed: (1) the construction of the list (in the described case is done beforehand by another person); (2) remembering to consult the list; (3) reading and interpreting the items of the list.

To resume, if from the system (read educator) view the cognitive abilities are enhanced, from the person (the learner's) view (if considering the person is involved in only (2) and (3) types of activities) his/her cognitive abilities are degraded, as, instead of trying to remember the list items, the person now should remember only to consult the list. Such type of memory degradation is clearly seen in the daily practices of education. Maybe this is one of the reasons for the present popular movement towards educational technology without a computer [44, 45]. With regard to simulations considered as information and telecommunications tools, simulations, as cognitive artefacts from the user perspective, could be harmful and demotivating. It is obvious that there is a kind of shift in tasks and cognitive processes involved in the process of simulation; therefore, educational simulations (the ones that are intended to be used in a simulation-using manner) should be carefully developed and tested [46].

6.4. Enhancing Computational Pedagogy by Design Science Research. The methodology of a DSR gives a pathway for formalization of SI-centred approach that is based on developing model-based scientific simulations. This methodology provides a set of analytical techniques, which are based on circumscription and abduction reasoning. The structure of DSR very clearly corresponds with the presented educational models. This mapping could provide a set of formalisms for practical implementations of DSR as a set of educational tools. The purpose of DSR as an educational tool is twofold.

First, the process of designing model-based simulations as cognitive artefacts is formalized. For this educational purpose, the set of DSR techniques is truncated and adapted for the educational needs. The learner is immersed in a kind of "quasi"-scientific research environment, satisfying SI by designing a set of simulations based on provided models of one or another type. Therefore, a practical algorithm is provided for such a case. The most important remark is the next: the learner should gain a clear understanding of the meaning "rigor" (as applied to this educational scientific research environment and future "real" scientific research). In this educational case, the word "rigor" means rigorous following of algorithm steps and teacher instructions.

Next, DSR as a practical design tool [36] for the future students' scientific activity is introduced. For this purpose, an example of one of the DSR formalisms could be introduced to the students. Such an introduction will provide a clear understanding of the method and its possible future (research) and present (educational tool) applications.

The focus of education is moved to the improvement of students' skills of building simulation models. These skills should provide universal prerequisites for the holistic educational environment. There is strong evidence of the tight relation of students' computational thinking and other simulation-making related skills with student's digital competences [47]. At the same time, the content of SP courses could be revised, focusing on modelling and simulation solutions. The model by itself should be designed as a multifaceted model with emphasis on teaching students and using didactic approach which is based on design science research methodology requirements. The multifaceted feature of the model allows students to construct their knowledge in a constructionist manner, enabling students' group work and collaborations and bringing gaming elements into learning.

7. Design Principles for Scientific Computing Introductory Content

7.1. Model-Based SI-Centred LR Design: Practical Example. The main points of SCE should be refocused on teaching the basic principles of model-based simulations and training on the skills enabling students to develop model-based simulations. This means not narrowing but, on the contrary, widening the scope of the curricula. At the same time, this unifies and integrates the complete educational environment, enables the constructionist approach to learning, and brings possibilities of gamification into the learning process. In designing the content, an integrated approach considering all phases of SI should be undertaken.

As an example, we provide a class of models, based on recurrent equations of the following form:

$$n + 1 = f(n, n - 1, \ldots). \quad (1)$$

The simplest example is Fibonacci numbers. The recurrence definition of Fibonacci numbers is

$$F(n+1) = F(n-1) + F(n-2), N > 2; F(1) = F(2) = 1. \tag{2}$$

In spite of their simple form, these examples become a basis of the whole unit focused on modelling scientific problems, which are based on recurrences, including stochastic recurrences. First, a scientific problem, for example, analysis of the population growth problem (P) [48–50], should be formulated. The recursive programming model could solve it. Therefore, to find a solution, students could develop a simple simulation in the form of a recursive program that calculates Fibonacci numbers. To facet the model, some additional feature should be incorporated. For example, the problem could be modified (P1) by asking about the time at which the population should be k times bigger than the population of humans. Now, students should think about the improvement of the previous model and make a horizontal step in the educational design cycle. It could be so that recursive solution will not be suitable due to the lack of computational resources, and this will force students to look for another solution as an explicit form of Fibonacci (P2) (Binet's formula): the next possible step is to introduce the Fibonacci primes. Thus, the modified problem (P3) could be formulated as

$$F(n) = \frac{(1+\sqrt{5})^n - (1-\sqrt{5})^n}{2^n\sqrt{5}}, \tag{3}$$

with the modification of (P2) with constraints on the population number allowing only prime Fibonacci values for the number of population. In such a case, students will be forced to implement parallelization techniques into their previously developed models.

The next step is to involve optimization and to modify the problem into the next form: $k(t) \longrightarrow \max$ within provided computational resources (P4). Students could be asked to represent the dependence $k(t)$ graphically. For better-prepared students, the emphasis on probability, stochastics [51], and analogues of Fibonacci series (P5) could be placed [48, 50]. The stochastic model of the evolution of populations, in which members undergo a phase of immaturity based on Fibonacci series, could be presented [48]. Let $\xi_i(n), i \in N+, n \in N$ be independent identically distributed random variables of generating function:

$$f(z) = \sum_{k \in N} p_k z^k, 0 < m = f'(1) < \infty. \tag{4}$$

Let $X(0)$ and $X(1)$ be independent random variables that are also independent of ξ. The stochastic Fibonacci model is defined as follows [48]:

$$X(n+2) - X(n-1) = \begin{cases} \sum_{l=1}^{X(n)} \xi_l(n), & \text{if } X(n) > 0, \quad n \in N; \\ 0 & \text{if } X(n) = 0, \quad n \in N. \end{cases} \tag{5}$$

Obviously, the particular case $\xi_i(n) = X(0) = X(1) = 1$ *a.s.*, $i \in N+, n \in N$, is nothing but the Fibonacci sequence. This allows introduction to stochastic experiments and stochastic modelling including stochastic processes and the MC method [52, 53]. As an example, the next problem (P6) to prove ergodicity of the described stochastic process [51] could be formulated. The described process is considered as mean-square ergodic in the first moment if

$$\lim_{n \longrightarrow \infty} \frac{\sum_0^n X(n)}{n} = E[X(n)], \tag{6}$$

where $E[X(n)]$ is the constant mean. The ergodicity in the second moment could be studied as well:

$$\lim_{n \longrightarrow \infty} \frac{\sum_0^n [(X(n+1) - E[X(n)])(X(n) - E[X(n)])]}{n} = E[(X(n+1) - E[X(n)])(X(n) - E[X(n)])]. \tag{7}$$

Such modifications or "faceting" of the problem introduces gamification features into the learning process and facilitates the constructionist approach to learning. It is important to stress that this simply formulated example incorporates the main features of the model-centred scientific inquiry-based education as focus on model-based simulations and seamless approach to theoretical backgrounds. In the process of designing a simulation model, students are forced to practise using design cycles and other DSR analytic techniques starting from formulating the hypothesis and providing working solutions for simulations. At the same time, the multifaceted feature of the model allows students to construct their knowledge in the constructionist manner, enabling students' group work and collaborations and bringing gaming elements into learning. Figure 1 presents the corresponding algorithm of the LR design.

7.2. Learning Resources for SP Education: A Generic Model for Design and Integration. This section provides generalizations on the development and integration of learning resources in the form of self-explanatory concept maps. Figure 2 provides a generic template for the development of educational content for SI-centred scientific programming education within an SI-centred educational environment. Figure 3 provides a generic template for a use case model for the integration of SLOs.

8. Conducting Educational Experiments: Case Study on Stochastics and Queueing

This section presents a methodological approach to designing software learning objects based on real-world scientific problems. Below, we present two case studies of designing SLO for student's simulation-making and scientific inquiry fostering activities. The following examples for developing educational content are based on the applications of queueing networks and stochastics. Basic principles of queueing networks [54–56] are easy to understand and could become a foundation for various problems and simulation experiments. Both applications require the same theoretical prerequisites from fields such as probability and operational research. The first application uses Python

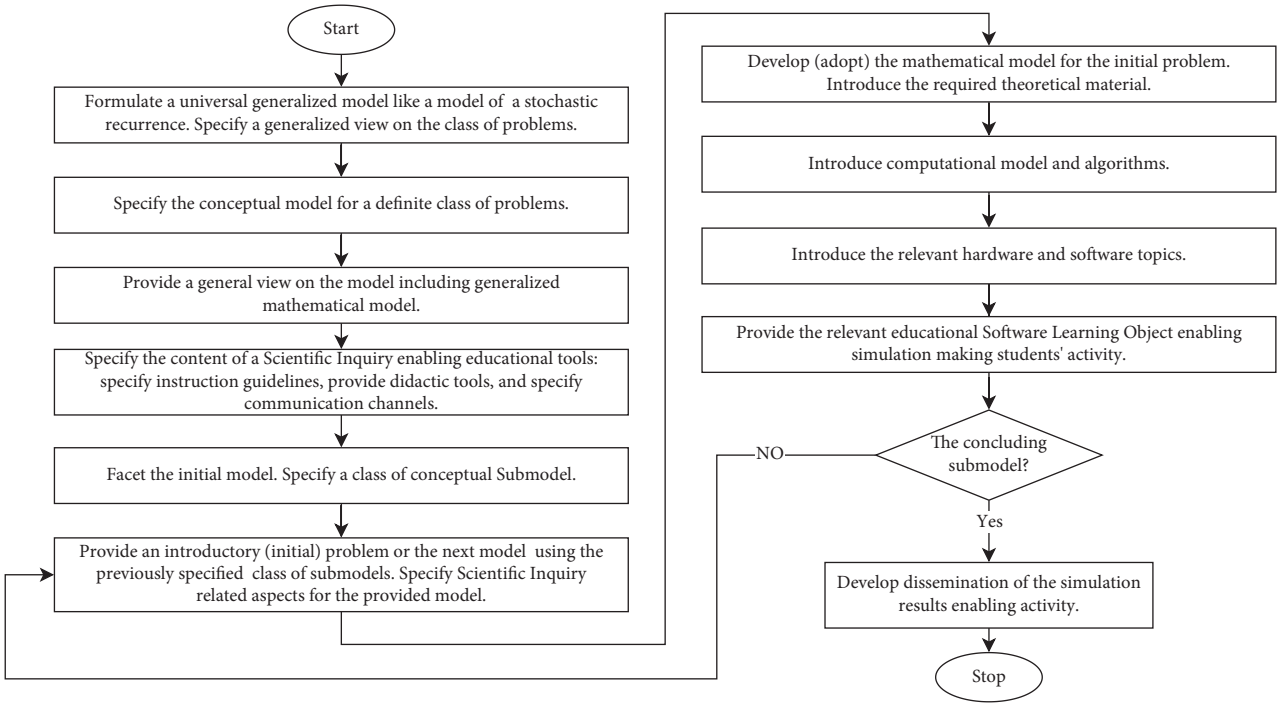

Figure 1: Model-based SI-centred LR design algorithm.

programming language for software development. It could be employed as an educational tool within a scientific inquiry-based introductory course for statistical methods or stochastics. The second application is based on the same theoretical foundations as first presented. It uses advanced computational techniques and C programming language for programming of models. This application could be employed for the teaching of scientific computations within an SP course.

8.1. Educational Experiment Planning. The general view of the computational experiment is presented in Figure 4 and includes the system itself, problem statement, mathematical model of the system, computational model, and simulation experiment. The details of the experiment are as follows: (S) system: queueing in series system; (P) problem statement: limit theorem of the system of queues in series under over-traffic condition; (M) mathematical model: Monte Carlo modelling; stochastic recurrence-based algorithmic solution; (C) computational model: hardware, software tools dependent computational model CM; (SI) simulation: computational experiment based on CM for HPCC.

The computational model is hardware and software dependent. It provides a basis for designing SLOs in the following manner: (CM) computational model: stochastic recurrence model of queues in series; (SM) a set of submodels: CM is divided into submodels: {SCM1, SCM2, SCM3,....}; (L) a set of learning resources: a set of submodels is projected to a set of LR in the form of SLOs.

8.2. Educational Experiment Design. The law of the iterated logarithm is proved for the values of important probabilistic characteristics of the queueing system, like the sojourn time of a customer and the maximum of the sojourn time of a customer [57]. It is also proved that the sojourn time of a customer can be approximated by some recurrent functional. The results of statistical simulations for various system parameters and distributions are provided as well. The laws of the iterated logarithm (LIL) are considered for investigating the sojourn time of a customer for the system of queues in series. The queue in series is a queueing system which consists of the series of single servicing nodes and in which a customer does not visit the same queueing node twice.

8.3. Algorithmic Solution, Benchmarking, and Modelling Results. Necessary theoretical background for creating the computational model for modelling the system of queues in series is provided in [57]. Such models allow us to study the behaviour of the system of queues in series in various (including overloading) conditions. Theoretical results provide a general insight and example of application of the mathematical method for stochastic analysis of the system of queues in series. This could be used as an aim for SI focused teaching within the practical study of the system of queues in series using simulation-making activities. At the same time, this provides a solution for the development of more efficient (linear) computational model (in case of overloading). A benchmarking could be based on counting the number of assembler instructions for a reduced instruction set style processor. First, the number of instructions for nonlinear model is calculated; next, the number of instructions for linear model is calculated. As the results of statistical modelling, the modelling results of the sojourn time of a customer in the system of queues in series under overloading conditions should be presented.

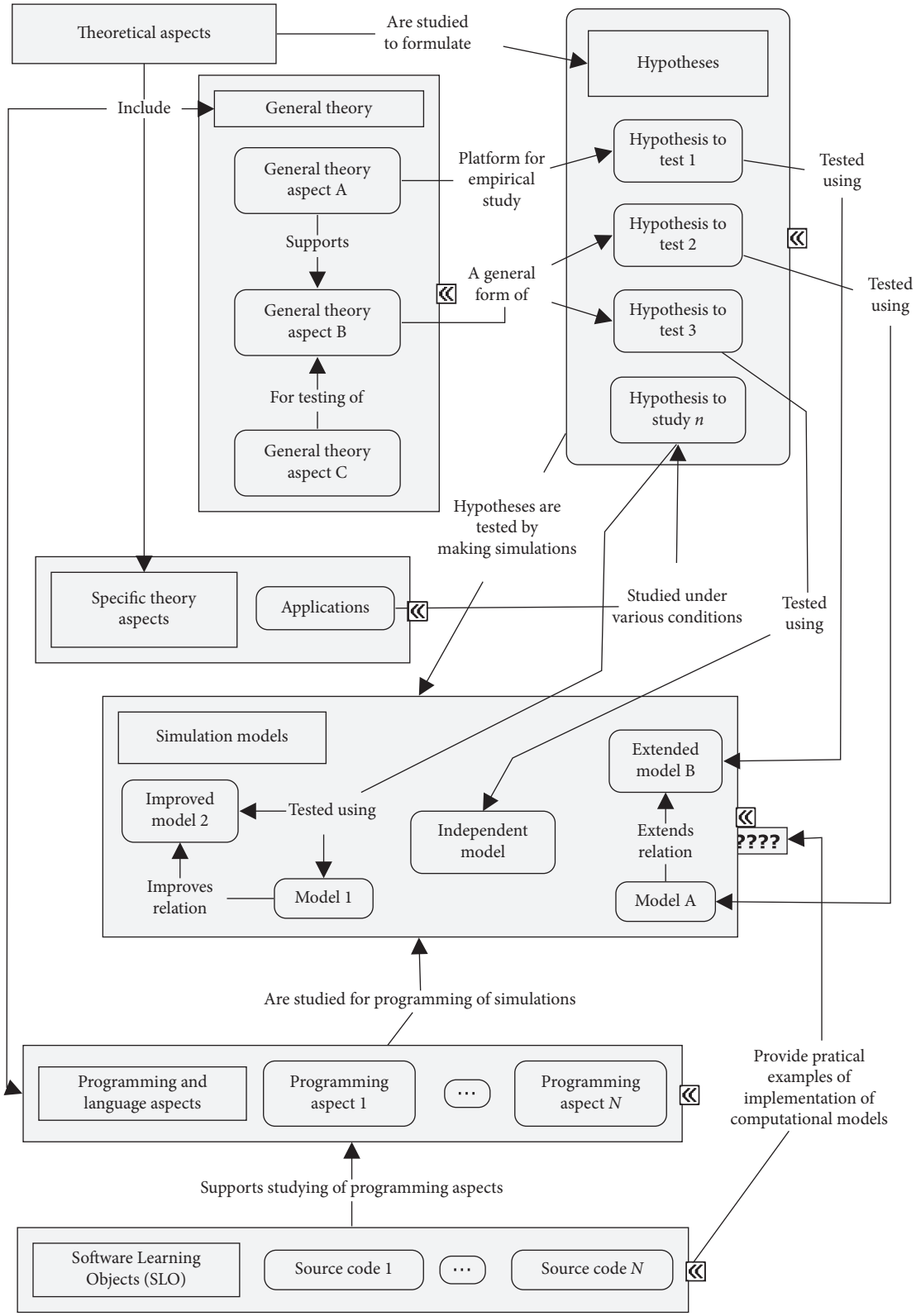

Figure 2: Generic template for educational content of SI-centred scientific programming education.

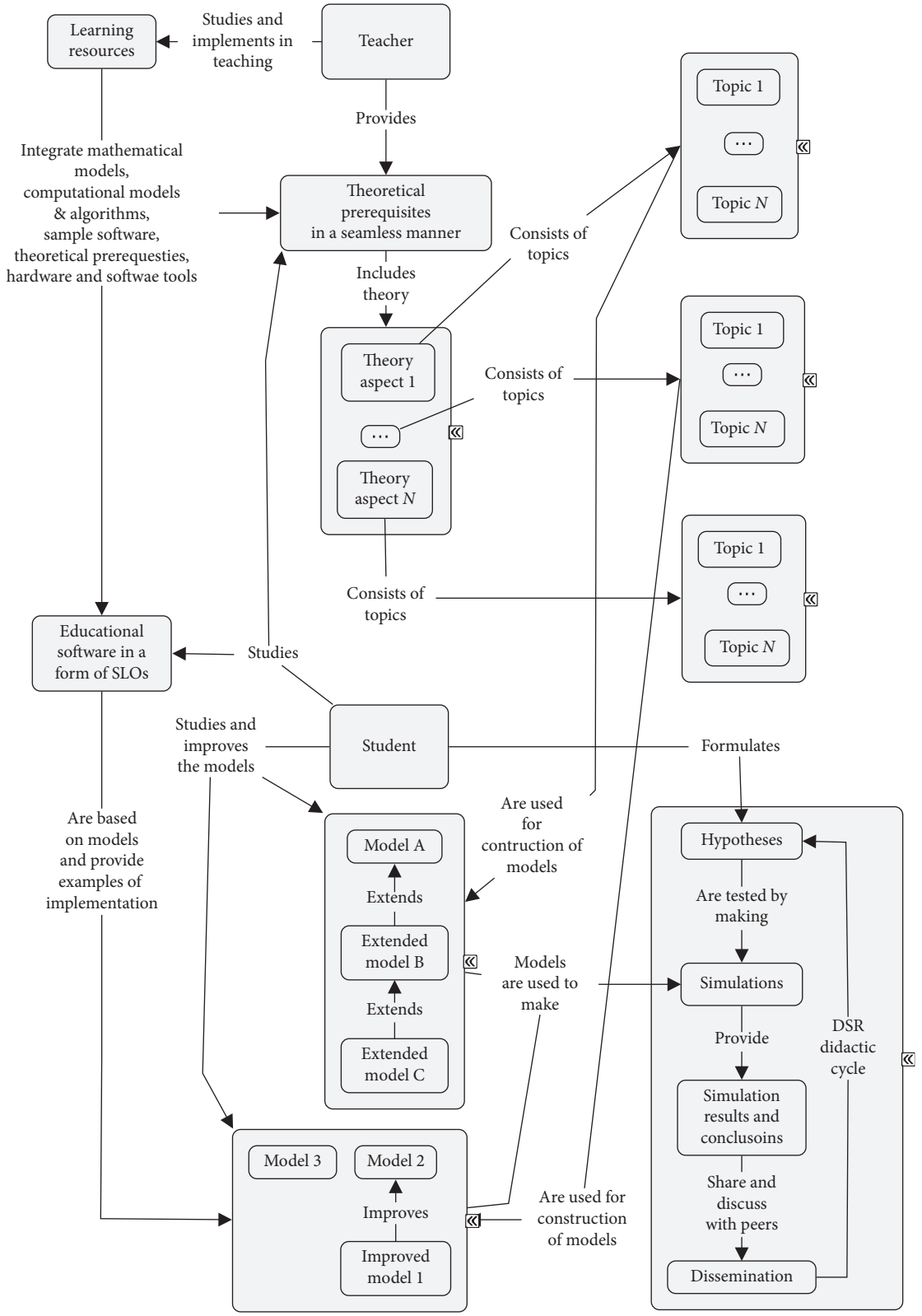

Figure 3: Generic template for a use case model for SLO integration.

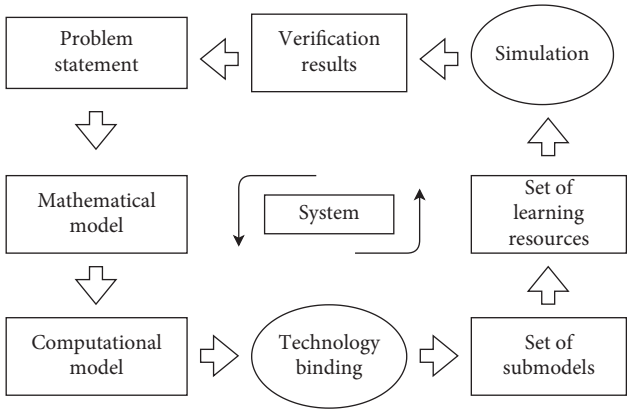

Figure 4: Educational computational experiment.

9. Case Study on Stochastics and Queueing: Content Domain Model

9.1. Python for Model-Based and Simulation-Centred Education. The system of queues in series provides us with the core for the development of a relevant model-centred simulation framework, which could become a basis for an integrated instructional unit. Such framework includes basic meanings, such as probability and distributions, basics of queueing and other theoretical topics, as well as more complex theoretical results and methods [1]. The basic meanings include randomness (random numbers, random numbers' distributions, generators of random numbers, Central Limit Theorem) and Python programming constructions (decorators, coroutines, yield expressions). Results that are more complex include theoretical facts such as queues series specifications and parameters like the sojourn time of the customer; recurrent equation for calculating the sojourn time; stochastic simulation methods; and multiprocessing techniques.

All these theoretical and programming structures allow the learner to carry out experiments with different models of the system of queues in series. The aim of such experimentation is twofold. First, it enables the learner to understand the next sequence, which is important in any scientific research: (1) theoretical facts to be studied; (2) conceptual model; (3) mathematical model; (4) algorithms and programming constructions; (5) computational model; (6) stochastic simulation and observation of simulation results. This will give the whole picture of the scope of the general scientific research to the learner. Then, it forces a deep understanding of stochastic simulations and basic programming constructions like multiprocessing and parallel programming. Such competencies are of primary importance in the field of SP.

The presented framework provides three computer models of the system of queues in series. Each of these models is rather different from its philosophy and key features. Although the aim of each of these models is to statistically model and investigate the main parameters of the system of queues in series, the ideas which stay behind the scene of these models are completely different. A comparison of these basic ideas will help the learner to understand the main fundamentals that lie behind the parallel calculations, multiprocessing statistical modelling, and simulation. The first model is based on real-time recordings and it is defined as an imitative model. It uses the Python multiprocessing module. The precision of this model depends on the precision and resolution of the Python time() method. It could be rather low in the case of various general-purpose operating systems and rather high in the case of the real-time operation system (RTOS). The learner could modify this model using the earlier presented recurrent equation (for the sojourn time calculations) and compare the results in both cases. The next model calculates the sojourn time of the customer and is based on stochastic simulations [57]. The model does not use multiprocessing directly. It emulates multiprocessing by using Python yield expressions. The last model presented here uses the Python MPI mpi4py module. From now, MPI techniques for statistical modelling could be used, and this could enhance Monte Carlo simulations by implementing additional trials.

In general, the task for the learner is to provide a series of experiments with the presented models and to obtain the experimental proof of the law of the iterated logarithm for the sojourn time of the customer in the case of the system of queues in series. A generic template for educational content design is presented in Figure 5.

9.2. Stochastic Simulations of Queueing Systems Using the C, MPI, and OpenMP Tools. The next approach, which is based on the same theoretical foundations as the previously presented one, is a framework for learning parallelization [2]. The framework provides a set of programming models and implements a model-centred approach to the introduction into SP and parallel programming. It is based on the task of stochastic simulations of the provided theoretical model of the system of queues in series. The system of queues in series is selected due to the simplicity of the primary definitions and wide possibilities for parallelization. Different parallelization techniques are implemented for programming the model. This allows carrying out a series of experiments with different programming models, comparing the results, and investigating the effectiveness of parallelization and different parallelization methods. Such parallelization methods include shared memory, distributed memory, and hybrid parallelization. These methods are implemented by MPI and OpenMP APIs. The other important task, to be solved within the scope of this research, is to develop DP for constructing of learning objects for learning parallelization. Such DP should be based on the relevant theoretical constructions and provide a basis for practical implementations. A generic template for designing educational aspects is presented in Figure 6.

10. Case Study on Stochastics and Queueing: Pedagogical Domain Model

10.1. Python-Based Simulation Models. A practical set of learning resources for the case study of the model-based approaches for teaching scientific programming and

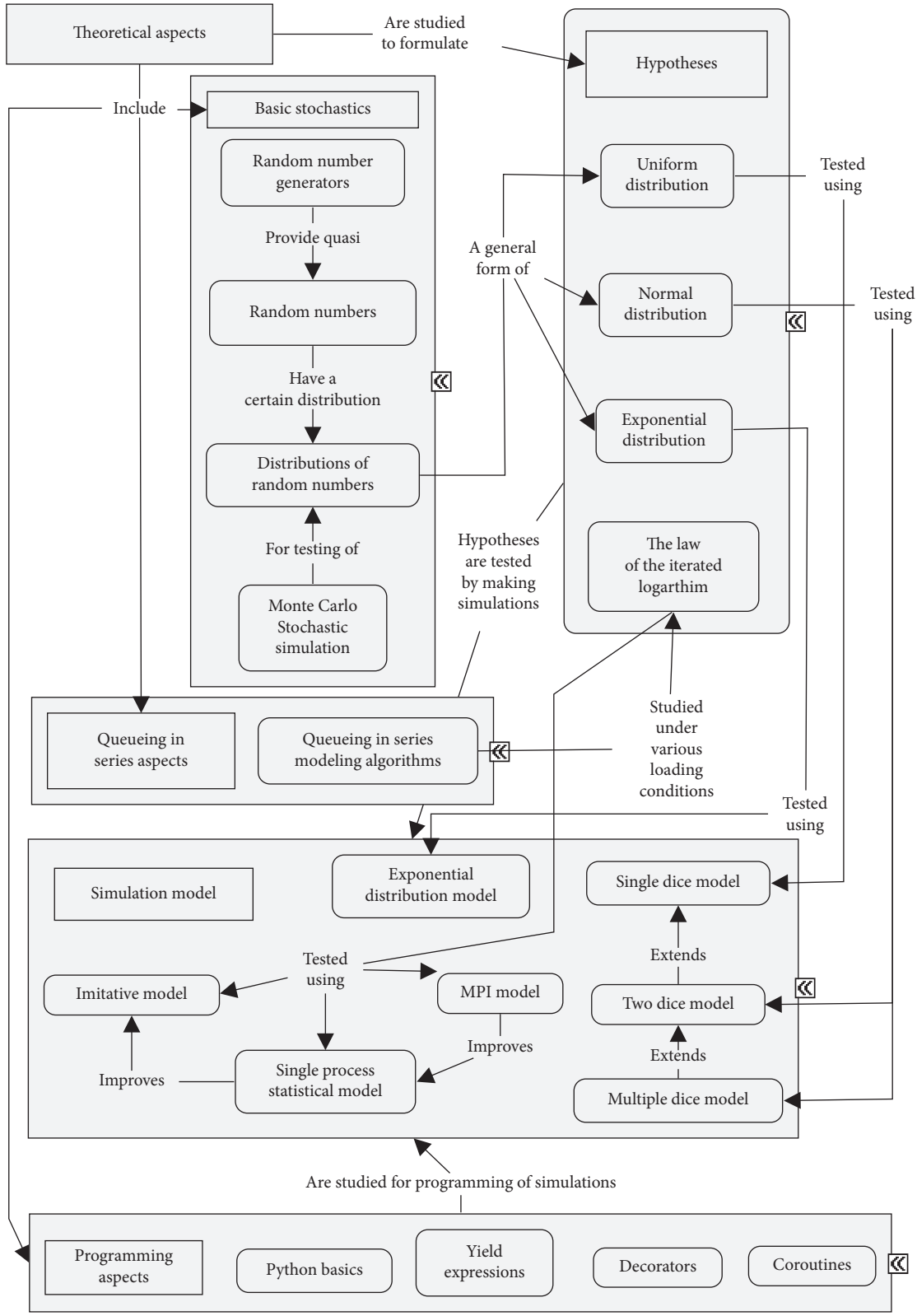

Figure 5: Educational content aspects of the Python-based scientific programming.

parallelization could be developed using Python programming language [1]. Key topics of an introductory curriculum in scientific programming include randomness with random numbers and distributions, stochastic simulations, and multiprocessing. We use a simple model of throwing a dice or a number of dice. The main task of these experiments is to

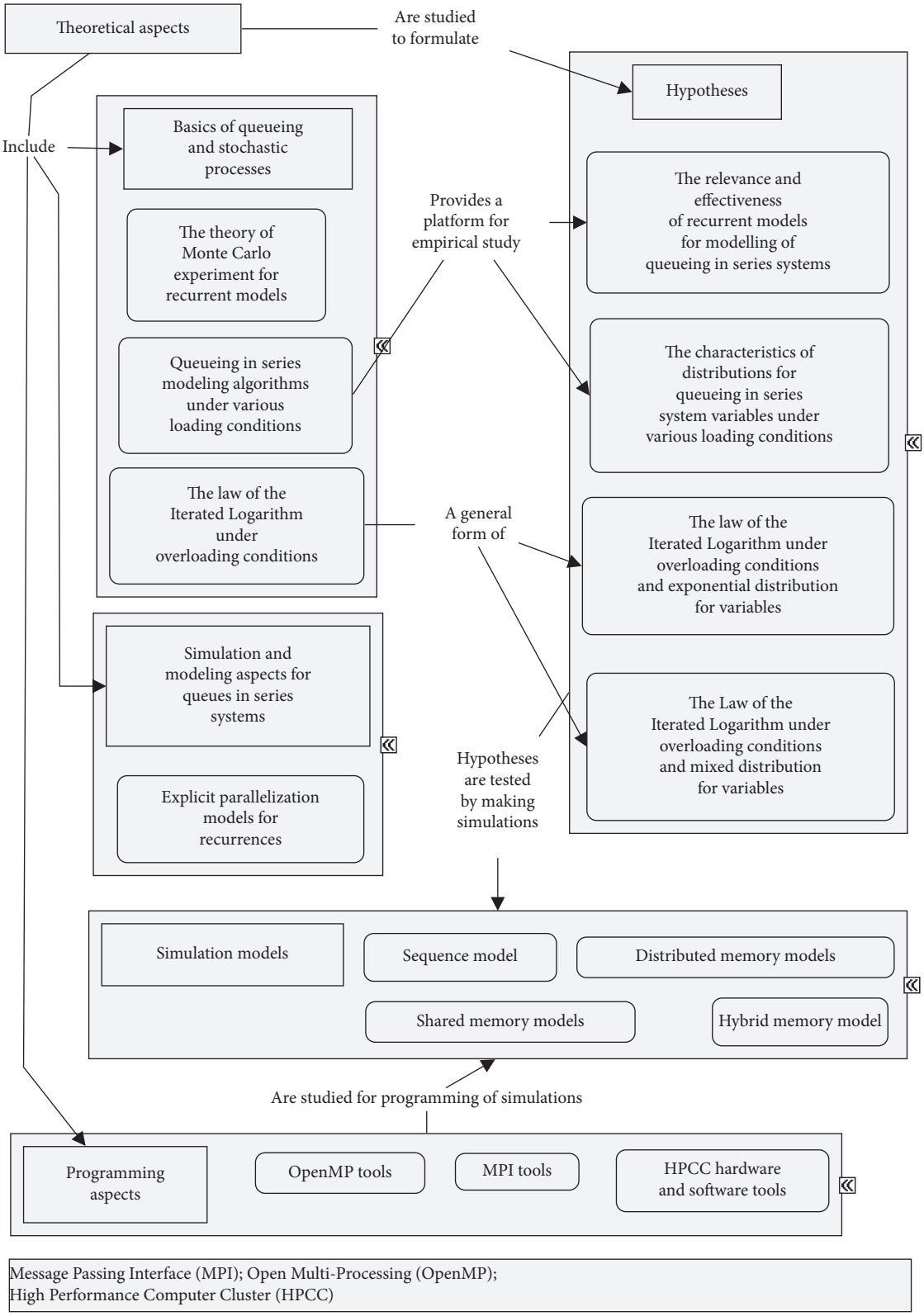

Figure 6: Educational content aspects of the C- and HPCC-based scientific programming.

provide an experimental proof of the Central Limit Theorem. These models and experiments with such models also enhance the learner's understanding of pseudo- and quasi-random number generators and the exponential distribution. This could provide basic ideas for more advanced experiments with the model of queueing systems. The use case model of the presented LR in the form of a concept map is presented in Figure 7.

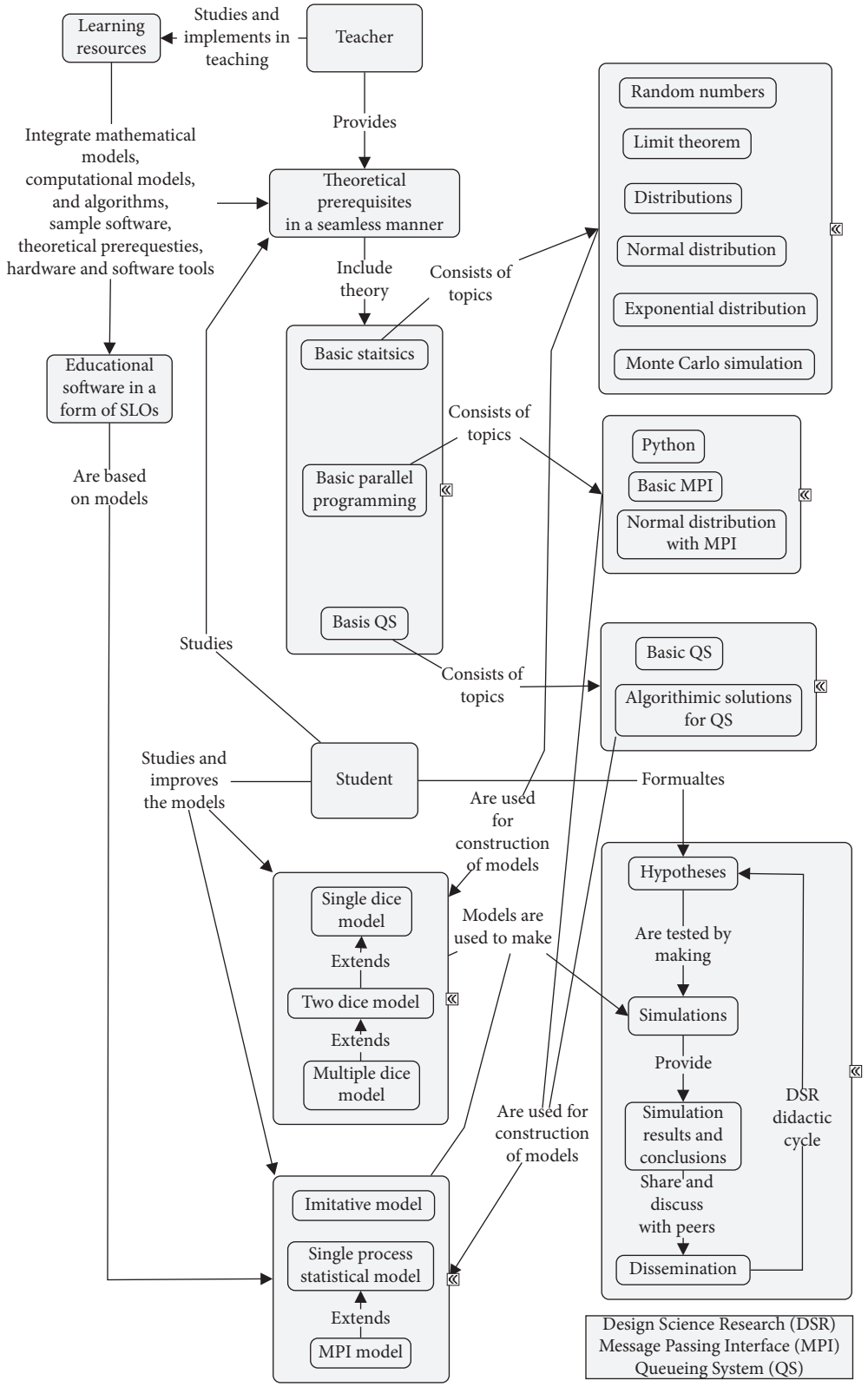

Figure 7: A use case model of the presented learning resources for teaching introductory stochastics

10.2. C-Based Simulation Model. A practical set of learning resources for the case study of the model-based approaches for teaching scientific programming and parallelization could be developed using C programming language [2]. In this research, we emphasize multiple instruction, multiple data (MIMD) parallel architecture and presume the HPCC

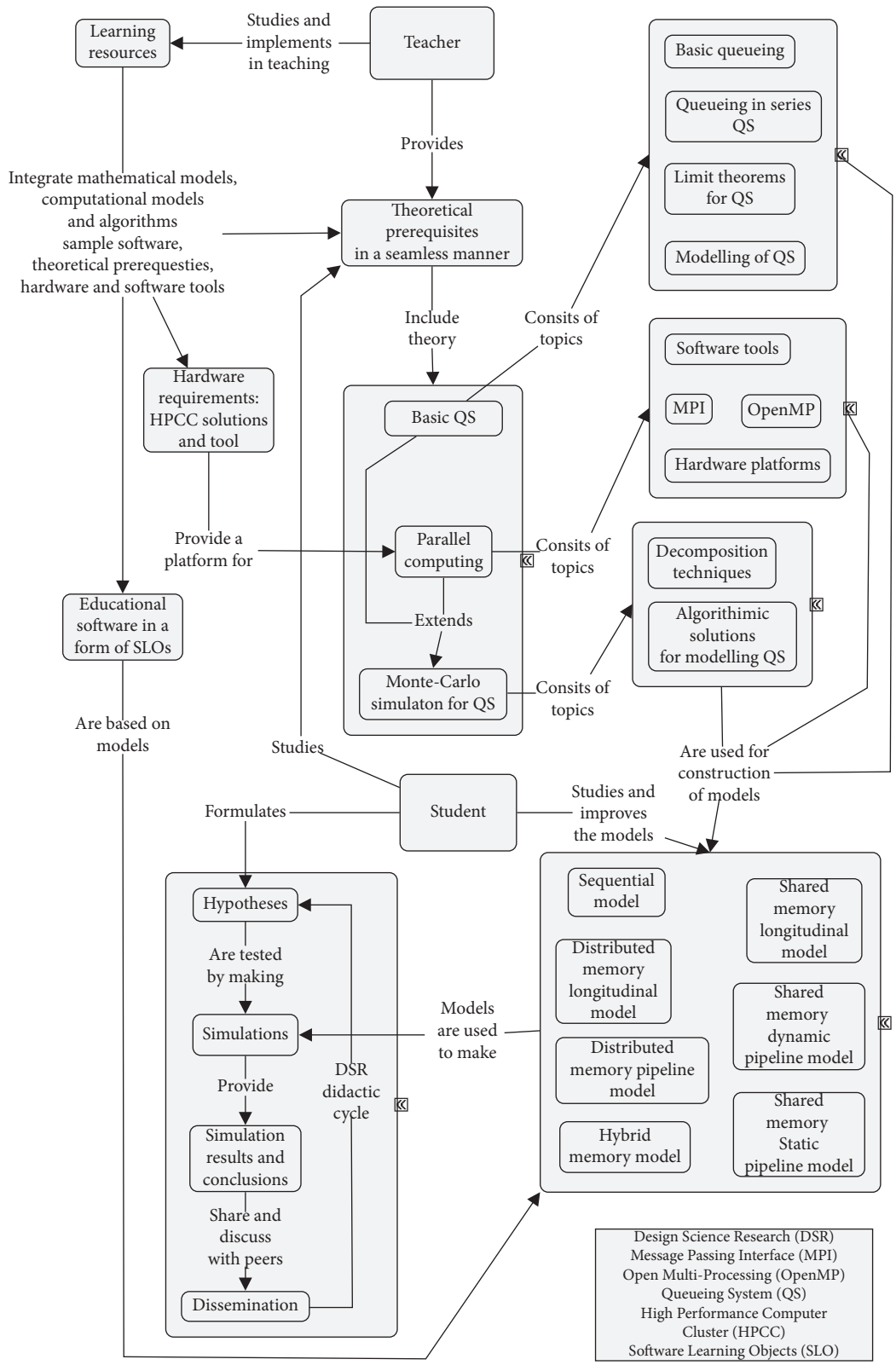

Figure 8: A use case model of the presented learning resources for teaching programming and parallelization.

as a target platform for calculations. Such a platform allows us to study different parallelization techniques and implement shared memory, distributed memory, and hybrid memory solutions. The main goal of parallelization is to reduce the program execution time by using the multiprocessor cluster architecture. It also enables us to increase

the number of Monte Carlo simulation trials during the statistical simulation of the queueing system and to achieve more accurate results in an experimental construction of the sojourn time distribution.

To implement parallelization, we use OpenMP tools for the shared memory model, MPI tools for the distributed memory model, and the hybrid technique for the hybrid memory model. For the shared memory decomposition, we use tasks and the dynamic decomposition technique in the case of the pipeline (transversal) decomposition, and the loop decomposition technique in the case of threads (longitudinal) decomposition. For the distributed memory decomposition, we use the standard MPI tools. For the hybrid decomposition, we use the shared memory (loop decomposition) for the longitudinal decomposition and MPI for the pipeline decomposition. The use case model of the presented LR in the form of a concept map is presented in Figure 8.

11. Summary and Conclusions

Methodological guidelines for the design and integration of learning resources that include software learning objects for scientific programming education have been developed. These include design principles for the development of the model-based scientific inquiry-centred learning resources, generic design templates for designing educational aspects of scientific programming education, generic use case models for learning resources for scientific programming education, and supportive methodological considerations. Several aspects should be highlighted. First, we try to avoid a reductionist approach and consider the educational process holistically, as it usually happens in a practical educational environment. Such a holistic approach considers all parts of the educational environment as an integral unit, and it is not possible to reduce the entire system to the number of its parts without losing the essence of the system as a whole. In our case, the scientific programming related to education must be seen as part of the entire educational environment, closely interwoven with all its constitutional components, such as educational context, technology, pedagogy, and teaching and learning content. Further, we should consider teachers and students, as well as the educational environment, in terms of integration rather than opposing any human or nonhuman participant in such educational processes. Furthermore, we encourage the view of educational sciences in general and scientific programming education in particular as applied and pragmatic activities that should focus on the development of practical solutions within the framework of properly designed SI and modelling focused educational settings. At the same time, we promote a set of practical tools, such as DSR and its application, to ensure such pragmatic approach to teaching and learning activities.

11.1. Conclusions on Methodological Guidelines. In response to the main question of the study, we could summarize the methodological guidelines as follows: (1) SLOs should be seen as part of a holistic educational content focused on SI-oriented education, based on the establishment of educational environments for developing educational simulations and conducting digital experiments. (2) SLOs should be developed as a set of educational resources that ultimately provide the technological linking of mathematical and computational models developed within a general educational discourse. This set of resources should reinforce the learner (based on a general model of the system to be studied) to develop a range of software solutions with different software and technology focus, allowing for the study of complex programming topics seamlessly and within a complex digital research environment. (3) SLOs should focus on providing practical examples of the process of creating practical implementations for developing simulation models: (3a) an independent simulation model; (3b) a consistent process of improving or expanding the model, requiring an increase in the number of computing resources involved in the educational process. (4) The SLO should focus on exploring an expanded number of programming aspects that are important for the technological binding of the model. (5) A model-centred approach to the development and design of SLOs should be applied, allowing for the development of a set of training resources based on a universal model. (6) Learning resources should be designed in such a way as to ensure the integration of mathematical models, computational models and algorithms, a seamless approach to theoretical prerequisites, and the operation of the corresponding hardware and software, as well as appropriate SLOs. (7) SLO are based on models and provide relevant implementation examples; an appropriate TPACK model should be developed for training teachers and instructional designers.

11.2. Conclusions on Pedagogical Domain of the TPACK Model. Appropriate implementation of the TPACK model provides an applicable framework for SP education focusing on three main areas: pedagogy, content, and technology of teaching SC and SP. Regarding the pedagogical domain, we could consider such a model as rather universal and independent of specific educational settings. For the technological and content domains, we could provide a sample specification based on the case studies presented.

Specifying the pedagogical domain, the main focus should be on the next features: (1) educational technology could be studied in terms of interdisciplinary university education as well as in terms of a unifying paradigm within university curricula; (2) curriculum development and instructional design could focus on the teaching/learning process enabling students to develop simulation models; (3) the process of designing simulations could be positioned within the educational framework enabling the relevant process of SI; (4) the didactic approach could be focused on appropriate formalization to ensure that the previously described teaching/learning process is well structured and properly arranged; (5) the focus could be on cognitive aspects of the educational process, including important features such as scaffolding and grounding.

11.3. Conclusions on Presented Case Studies. The article presents generalizations for the case studies which are focused on experimental applications of stochastic recurrences, applied to modelling of the system of queues in series under various stochastic parameters. The developed theory for modelling of the system of queues in series allows educational experiments with big data to be implemented. At the same time, this real-life scientific application provides a suitable background for learning objects for teaching scientific programming to be designed and implemented. Such learning objects are designed in the form of software models that implement software learning objects on topics in the study. The model-centred approach allows us to implement a universal platform for teaching in introductory statistics and parallelization.

The presented universal templates provide a comprehensive background for developing application-specific models. First, computational platforms such as HPCC allow different algorithms and programming techniques to be employed for parallelization. These include shared memory, distributed memory, and hybrid memory solutions. For application in the study, HPCC serves as a universal platform, allowing different parallelization techniques to be tested using this definite platform for computations. This allows implementing benchmarking for different programming solutions, thus increasing students' motivation and implementing the constructionist approach to learning. Next, stochastic recurrences of a general type provide a suitable model for computations. This could be used to model definite application areas. Incorporating stochastics into the model, one could test multidimensional parallelization techniques, using the Monte Carlo method. Finally, the model-centred approach could be used for the design and further development of software learning objects for teaching parallelization. This approach is suitable for both implicit and explicit parallelization methods.

For the technological domain, the main focus could be on (1) appropriate and parallelization enabling hardware architectures; (2) relevant software engineering methods, enabling model-centred approaches for software development; (3) relevant parallelization enabling software development/programming tools; relevant big data processing/programming/presentation tools.

For the content domain, the main focus could be on teaching/learning aspects of (1) stochastic and theoretical aspects of introductory stochastics and probability; (2) theoretical and computational aspects of introductory recurrences and stochastic recurrences; (3) distributions, limit theorems, and stochastic processes in the basics of probability theory; (4) Monte Carlo modelling methods and the relevant computational aspects of Monte Carlo modelling; (5) basic queueing theory, main parameters, and characteristics of queues, queues in series, queueing networks, and computational aspects of queueing models.

Data Availability

The case studies data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] V. Dolgopolovas, V. Dagienė, S. Minkevičius, and L. Sakalauskas, "Python for scientific computing education: modeling of queueing systems," *Scientific Programming*, vol. 22, no. 1, pp. 37–51, 2014.

[2] V. Dolgopolovas, V. Dagienė, S. Minkevičius, and L. Sakalauskas, "Teaching scientific computing: a model-centered approach to pipeline and parallel programming with C," *Scientific Programming*, vol. 2015, Article ID 820803, 18 pages, 2015.

[3] A. R. Hevner, "A three cycle view of design science research A three cycle view of design science research," *Scandinavian Journal of Information Systems*, vol. 19, pp. 87–92, 2004.

[4] C. Science, "Design research in information systems overview of design research can design Be research," *Information Systems Journal*, vol. 1–17, 2006.

[5] A. Dresch, D. P. Lacerda, and J. A. V. Antunes, *Design Science Research: A Method for Science and Technology Advancement*, Springer, Berlin, Germany, 2015.

[6] V. Štuikys, *Smart Learning Objects for Smart Education in Computer Science: Theory, Methodology and Robot-Based Implementation*, Springer, Berlin, Germany, 2015.

[7] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[8] G. Goldkuhl, "Pragmatism vs interpretivism in qualitative information systems research," *European Journal of Information Systems*, vol. 21, no. 2, pp. 135–146, 2012.

[9] D. S. Hovorka, "Design science research: a call for A pragmatic perspective,"May 2009, https://aisel.aisnet.org/sprouts_all/322.

[10] J. Piaget, *The Construction of Reality in the Child*, Routledge, Abingdon, UK, 2013.

[11] J. S. Wooster and S. Papert, "Mindstorms: children, computers, and powerful ideas," *The English Journal*, vol. 71, no. 8, p. 60, 1982.

[12] S. Papert and I. Harel, "Situating constructionism," *Constructionism*, vol. 36, pp. 1–12, 1991.

[13] A. S. Gibbons, "Model-centered instruction," *Journal of Intelligent Materials Systems and Structures*, vol. 14, pp. 511–540, 2001.

[14] M. Ford and P. N. Johnson-Laird, *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*, Harvard University Press, Cambridge, MA, USA, 1985.

[15] N. J. Nersessian, *Creating Scientific Concepts*, MIT press, Cambridge, MA, USA, 2018.

[16] L. Magnani, "Model-based creative abduction," in *Model-Based Reasoning in Scientific Discovery*, pp. 219–238, Springer, Berlin, Germany, 1999.

[17] G. D. Crnkovic, *Constructive Research and Info-Computational Knowledge Generation*, Springer, Berlin, Germany, 2010.

[18] P. Thagard, "How brains make mental models," in *Studies in Computational Intelligence*, pp. 447–461, Springer, Berlin, Germany, 2010.

[19] P. Thagard and T. C. Stewart, "The AHA! experience: creativity through emergent binding in neural networks," *Cognitive Science*, vol. 35, no. 1, pp. 1–33, 2011.

[20] V. Štuikys and R. Damaševičius, "Development of generative learning objects using feature diagrams and generative techniques," *Informatics Education*, vol. 7, pp. 277–288, 2008.
[21] A. C. Michalos and H. A. Simon, *The Sciences of the Artificial*, MIT press, Cambridge, MA, USA, 1970.
[22] C. Sonnenberg and J. Vom Brocke, "Evaluation patterns for design science research artefacts," in *Communications in Computer and Information Science*, pp. 71–83, Springer, Berlin, Germany, 2012.
[23] C. Sonnenberg and J. vom Brocke, "Evaluations in the science of the artificial - reconsidering the build-evaluate pattern in design science research," *Lecture Notes in Computer Science*, Springer, Berlin,Germany, pp. 381–397, 2012.
[24] S. Gregor and A. R. Hevner, "Positioning and presenting design science research for maximum impact," *MIS Quarterly*, vol. 37, no. 2, pp. 337–355, 2013.
[25] L. Lockyer, S. Bennett, S. Agostinho et al., *Handbook of Research on Learning Design and Learning Objects: Issues, Applications and Technologies*, IGI Global, Hershey, PA, USA, 2008.
[26] A. M. Seoane-Pardo and F. J. García-Peñalvo, "Pedagogical patterns and online teaching," in *Online Tutor 2.0: Methodologies and Case Studies for Successful Learning*, pp. 298–316, IGI Global, Hershey, PA, USA, 2014.
[27] D. Churchill, "Towards a useful classification of learning objects," *Educational Technology Research and Development*, vol. 55, no. 5, pp. 479–497, 2007.
[28] P. Mishra, "Considering contextual knowledge: the TPACK diagram gets an upgrade," *Journal of Digital Learning in Teacher Education*, vol. 35, no. 2, pp. 76–78, 2019.
[29] C. S. Peirce and N. Houser, *The Essential Peirce: Selected Philosophical Writings*, vol. 1, pp. 1867–1893, Indiana University Press, Bloomington, IN, USA, 1993.
[30] E. R. House, "Response to "notes on pragmatism and scientific realism"" *Educational Researcher*, vol. 21, no. 6, pp. 18-19, 1992.
[31] P. Collier, *The New Pragmatism*, Routledge, Abingdon, UK, 2017.
[32] M. J. Flynn, "Very high-speed computing systems," *Proceedings of the IEEE*, vol. 54, no. 12, pp. 1901–1909, 1966.
[33] D. B. Skillicorn, *Foundations of Parallel Programming*, Cambridge University Press, Cambridge, UK, 1994.
[34] J. X. Jadrich and C. Bruxvoor, *Learning and Teaching Scientific Inquiry: Research and Applications*, NSTA Press, Arlington, VA, USA, 2015.
[35] M. Windschitl, J. Thompson, and M. Braaten, "Beyond the scientific method: model-based inquiry as a new paradigm of preference for school science investigations," *Science Education*, vol. 92, no. 5, pp. 941–967, 2008.
[36] V. Dolgopolovas, V. Dagienė, E. Jasutė, and T. Jevsikova, "Design science research for computational thinking in constructionist education: a pragmatist perspective," *Problemos*, vol. 95, pp. 144–159, 2019.
[37] O. Yasar, P. Veronesi, J. Maliekal, L. J. Little, S. E. Vattana, and I. H. Yeter, "Computational pedagogy: fostering a new method of teaching," in *Proceedings of the 2016 ASEE Annual Conference & Exposition Proceedings*, New Orleans, LA, USA, June 2016.
[38] O. Yasar and J. Maliekal, "Computational pedagogy: a modeling and simulation approach," *Computing in Science & Engineering*, vol. 16, no. 3, pp. 78–88, 2014.
[39] J. d. Bransford, *How People Learn - Barin, Mind, Experience and School*, National Academy Press, Washington, DC, USA, 2004.
[40] M. J. Prince and R. M. Felder, "Inductive teaching and learning methods: definitions, comparisons, and research bases," *Journal of Engineering Education*, vol. 95, no. 2, pp. 123–138, 2006.
[41] S. Meyn and S. Meyn, *Simulation and Learning*, Springer, Berlin, Germany, 2011.
[42] Y. Samur and M. A. Evans, *Learning Science through Computer Games and Simulations*, M. A. Honey and M. Hilton, Eds., National Academies Press, Washington, DC, USA, 2011.
[43] D. Lomas, *Cognitive Artifacts: An Art-Science Engagement*, Department of Cognitive Science, University of California, San Diego, CA, USA, 2007.
[44] S. Coughlan, "Computers "do not improve" pupil results," *Says OECD. BBC News, Viewed*.vol. 10, 2015.
[45] M. Ritchel, "A silicon valley school that doesn't compute," *New York Times*, vol. 22, p. A1, 2011.
[46] C. E. Wieman, W. K. Adams, and K. K. Perkins, "Physics: PhET: simulations that enhance learning," *Science*, vol. 322, no. 5902, pp. 682-683, 2008.
[47] A. Juškevičiene and V. Dagienė, "Computational thinking relationship with digital competence," *Informatics in Education*, vol. 17, no. 2, pp. 265–284, 2018.
[48] M. Iosifescu, N. Limnios, and G. Oprişan, "Branching models," in *Introduction to Stochastic Models*, pp. 227–313, Wiley, Hoboken, NJ, USA, 2013.
[49] W. Bryc, *Applied Probability and Stochastic Processes*, University of Cincinnati, Cincinnati, OH, USA, 1996.
[50] C. C. Heyde, "On Fibonacci (or lagged Bienaymé-Galton-Watson) branching processes," *Journal of Applied Probability*, vol. 18, no. 3, pp. 583–591, 1981.
[51] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, Tata McGraw-Hill Education, New York, NY, USA, 2002.
[52] W. A. Thompson, H. M. Taylor, and S. Karlin, *An Introduction to Stochastic Modeling*, Academic Press, Cambridge, MA, USA, 1985.
[53] R. Toral and P. Colet, *Stochastic Numerical Methods: An Introduction for Students and Scientists*, Wiley, Hoboken, NJ, USA, 2014.
[54] B. L. Nelson, *Stochastic Modeling: Analysis and Simulation*, Courier Corporation, Chelmsford, MA, USA, 2012.
[55] A. Mercer and G. F. Newell, *Applications of Queueing Theory*, Springer Science & Business Media, Berlin, Germany, 1972.
[56] N. Gautam, *Analysis of Queues: Methods and Applications*, CRC Press, Boca Raton, FL, USA, 2012.
[57] S. Minkevičius, V. Dolgopolovas, and L. L. Sakalauskas, "A law of the iterated logarithm for the sojourn time process in queues in series," *Methodology and Computing in Applied Probability*, vol. 18, no. 1, pp. 37–57, 2016.