VILNIUS UNIVERSITY

VLADIMIRAS DOLGOPOLOVAS

SOFTWARE LEARNING OBJECTS FOR SCIENTIFIC COMPUTING EDUCATION: TEACHING SCIENTIFIC INQUIRY WITH RECURRENCE BASED STOCHASTIC MODELS

Doctoral Dissertation
Technological Sciences, Informatics Engineering (07 T)

Vilnius, 2018

The dissertation was written between 2013 and 2017 at Vilnius University.

Scientific Supervisor
Prof. Dr. Valentina Dagienė (Vilnius University, Technological Sciences, Informatics Engineering – 07 T).

VILNIAUS UNIVERSITETAS

VLADIMIRAS DOLGOPOLOVAS

PROGRAMAVIMO MOKYMOSI OBJEKTAI MOKSLINĖS KOMPIUTERIJOS MOKYMUI: MOKSLINIO TYRIMO STUDIJOS NAUDOJANT STOCHASTINIUS REKURENTINIUS MODELIUS

Daktaro disertacija
Technologijos mokslai, informatikos inžinerija (07 T)

Vilnius, 2018

Disertacija rengta 2013–2017 metais Vilniaus universitete.

Mokslinė vadovė
prof. dr. Valentina Dagienė (Vilniaus universitetas, technologijos mokslai, informatikos inžinerija – 07 T).

Acknowledgments

Firstly, I would like to express my sincere gratitude to the supervisor of the thesis, Professor Dr. Valentina Dagienė, for her considerable support. She undertook advising as well as organizational issues and all this had a considerable impact on the results and made it possible the thesis to be finished. I also would like to thank Professor Dr. Erkki Sutinen, who provided me with a pathway to systematize the thesis. His advice and support during my training period at the University of Turku was invaluable and allowed me to summarize and finish the research. He spent his time reviewing the thesis as well. Many thanks to Professor Dr. Saulius Gudas who spent a lot of time reviewing the thesis and who made many valuable comments and gave me important advice. I also would like to thank Professor Dr. Vitalij Denisov from Klaipėda University. He commented on the thesis and his remarks were very valuable finalizing the work. I would also like to thank the committee member, Professor Dr. Eduardas Bareiša, and the head of the committee Professor Dr. Julius Žilinskas for their time and personal contribution.

I especially thank Professor Dr. Inggriani Liem of the Bandung Institute of Technology and Assoc. Professor Dr. Mikko-Jussi Laakso from the University of Turku who spent much time reading and commenting on the thesis. Dr. Renata Burbaitė should be especially mentioned. Her comments and suggestions were invaluable summarizing the thesis. Thank you very much Renata for your time and tremendous contribution. Many thanks to Assoc. Professor Dr. Tatjana Jevsikova for her time and comments. I would like to express my sincere thanks to Professor Dr. Leonidas Sakalauskas for his important advice and positive attitude to my work and to Professor Dr. Albertas Čaplinskas, who gave me an understanding of the important topics that helped to systematize the results of the thesis.

I would especially like to thank all colleagues in the Educational Systems research unit at Vilnius University Institute of Data Science and Digital Technologies. All of you supported me a lot during my work on the thesis. I also thank the language editors Zuzana Šiušaitė and Audra Ivanauskienė for their contribution editing the thesis.

Abstract

Modern education requires innovative approaches that need to be implemented both at the highest levels of the educational environment, such as educational policy or curriculum requirements, and at practical levels such as instructional design and didactic aspects of particular subjects of university education. The reasons for this are: (1) obvious technological changes, including the progress of the world digital economy as a whole; (2) sufficient improvement of innovative digital technologies in the services and industry in particular. This progress requires that more and more knowledge and practical skills of students be taught and trained in university curricula.

One of the most important and at the same time difficult areas is computational science. Modern computational science has been transformed from emphasis in the early days to pure computing aspects, to the present-day focus on applications, scientific research, scientific inquiry and the digital design process, and this is of paramount importance for every area of modern science and technology. Another important aspect is innovation. Modern innovations are primarily interdisciplinary and require an interdisciplinary approach to the research process, which should be developed, implemented and taught in the framework of a very diverse curriculum of the university.

The aims of the research are: (1) to provide an integral view on various earlier described aspects of educational technology in general; (2) to provide a methodological constructionist framework for Scientific Inquiry (SI) based Scientific Computing Education (SCE); (3) to develop Design Principles and the Supportive Application and Integration Methodology (DPSAIM) for the development of Learning Resources for SCE; (4) to practically implement a set of sample learning resources including implementations for instructional design and didactics. The set of sample learning resources in the form of programming models and Software Learning Objects (SLOs) is aimed at such topics within the scope of scientific computing education as stochastics, including limit theorems, Monte Carlo methods, queueing theory, and big-data computation and visualization techniques. To implement the research task: (1) a comprehensive meta analysis of domain features which is based on the well known Technological Pedagogical and Content Knowledge (TPACK) model is provided; (2) a comprehensive study of a model of a general form of stochastic recurrence is done and the relevant computational model is provided; (3) a comprehensive study of a model of the system of queues in series, which is based on a general model of stochastic recurrence is done, the relevant computational model is provided; (4) a set of learning resources in the form of SLOs is developed and provided; (5) a constructionist framework for Scientific Computing (SC) education is provided; (6) a set of didactic tools and instructional design methods is implemented.

Keywords: scientific computing education, methodology of Scientific Computing education, Design Science Research, engineering education, university education, computer science education, information systems, interdisciplinary innovations, interdisciplinary university education, STEM university education, scientific inquiry, cognitive artifacts, model-centred instruction, model-based education, constructionism, constructionist education, computer simulations, computer science education, informatics education.

Contents

List of Figures

List of Tables

List of Algorithms

Glossary

Cognitive Artifact Cognitive artifacts may be defined as "those artificial devices that maintain, display, or operate upon information in order to serve a representational function and that affect human cognitive performance". vi, xiii, 9, 17, 48, 55, 56, 89, 90, 91, 96, 99, 117

Computational Model A computational model is a mathematical model in computational science that requires extensive computational resources to study the behavior of a complex system by computer simulation. 6

Constructionism Constructionist learning is when learners construct mental models to understand the world around them. Constructionism advocates student-centered, discovery learning where students use information they already know to acquire more knowledge. vi, 15, 43, 44, 50, 95

Design Science Research Design Science Research is the research methodology, which focuses on the development and performance of (designed) artifacts with the explicit intention of improving the functional performance of the artifact. Design science research is typically applied to categories of artifacts including algorithms, human/computer interfaces, design methodologies (including process models) and languages. vi, vii, ix, xiii, xviii, 2, 7, 9, 10, 17, 68, 70, 74, 85, 96, 99, 117, 169, 185

Educational Technology Educational technology is as the theory and practice of educational approaches to learning. xiii, 1, 7, 9, 13, 14, 39, 43, 46, 48, 58, 67, 68, 75, 85, 91, 117

Engineering Education Engineering education is the activity of teaching knowledge and principles to the professional practice of engineering. vi, 1, 37

Instructionism Instructionism refers to all of the educational theories based on the idea of the teacher teaching, usually according to a predetermined schedule, rather than on students learning from their own experiences at their own pace. This includes any form of rote learning, and most forms of book learning in actual use, as well as drill and practice. 95

Parallelization Parallelization is the act of designing a computer program or system to process data in parallel. xiv, 5, 7, 17, 18, 19, 23, 26, 28, 38, 40, 41, 42, 105, 107, 109, 111, 112, 123, 126, 130, 133, 134, 137, 177, 180, 181, 182, 187, 188, 203

Scientific Computing Scientific computing (also computational science or scientific computation) is a rapidly growing multidisciplinary field that uses advanced computing capabilities to understand and solve complex problems. It is an area of science which spans many disciplines, but at its core it involves the development of models and simulations to understand natural systems. vi, xiii, xix, 1, 3, 5, 6, 7, 13, 15, 17, 18, 19, 20, 34, 37, 39, 51, 77, 83, 84, 116, 119, 153

Scientific Inquiry Scientific inquiry generally aims to obtain knowledge in the form of testable explanations that scientists can use to predict the results of future experiments. This allows scientists to gain a better understanding of the topic under study, and later to use that understanding to intervene in its causal mechanisms. vi, xix, 1, 3, 5, 17, 18, 19, 83, 84, 119

Software Learning Object Software program as a learning object used to teach algorithms, programming, scientific computing topics. vi, xix, 5, 7, 14, 16, 17, 19, 137

Taxonomy Taxonomy is the practice and science of classification of things or concepts, including the principles that underlie such classification. xiv, 67, 120, 199, 201

University Education Higher education, post-secondary education, or third level education. vi, xiii, 1, 3, 10, 13, 15, 17, 24, 27, 43, 48, 70, 81

Acronyms

API Application Programming Interface. 41, 111, 126

BBIL Boundary Breaking for Interdisciplinary Learning. 33

CBT Competency based training. 193
CISC Complex Instruction Set Computing. 107
CPU Central Processing Unit. 120
CS Computer Science. xiii, 8, 25, 34, 36, 38, 42, 43, 48, 49, 60, 61, 62, 70, 71, 76, 77, 87, 104, 175
CS1 Computer Science One. 60, 61
CSE Computer Science Education. 15, 43, 44, 46, 50

DP Design Principles. 5, 7, 8, 10, 14, 18, 19, 25, 74, 72, 106, 111, 119, 195
DPSAIM Design Principles and the Supportive Application and Integration Methodology. vi, xvi, 10, 13, 14, 17, 188, 189, 191, 196, 198, 199, 202
DR Design Research. 69
DSR Design Science Research. *Glossary:* Design Science Research, vi, vii, ix, xiii, xiv, xvi, 2, 3, 4, 7, 8, 9, 10, 16, 17, 25, 48, 68, 69, 70, 71, 72, 74, 72, 74, 75, 76, 84, 85, 86, 96, 97, 99, 100, 104, 106, 116, 117, 119, 169, 185, 188

EC Educational Content. 13, 14, 17, 22, 27, 62, 79, 119

FIFO First-in First-out. 161, 162, 176
FODA Feature-oriented Domain Analysis. 65

GLO Generative Learning Objects. 64, 66
GPU Graphics Processing Unit. 123

HCI Human-Computer Interface. 71
HPC High Performance Computing. x, xi, 42, 123, 137, 146, 177, 180, 187, 196, 203
HPCC High Performance Computing Cluster. 26, 107
HTC High Throughput Computing. 175

IEEE Institute of Electrical and Electronics Engineers. 62, 191
IS Information System. 8, 9, 71
ITC Information and Communication Technologies. 56, 70, 71, 91, 95

KO Knowledge Objects. 8, 14

LIFO Last-in Last-out. 161
LIL Law of the Iterated Logarithm. 138, 141, 153
LO Learning Objects. xi, xii, 8, 9, 10, 14, 58, 60, 61, 62, 64, 65, 76, 188, 190, 191, 192, 193, 200, 201
LR Learning Resources. vi, xiv, 5, 7, 8, 10, 13, 14, 18, 19, 23, 107, 138, 169, 185, 188, 189, 190, 191, 192, 195, 196, 197, 200, 201, 202, 203

MC Monte Carlo. 6, 19, 26, 28, 106, 111, 128, 131, 132, 137, 138, 146, 153, 157, 162, 163, 167, 169, 176, 177, 179, 182, 187, 190, 192, 203

MIMD Multiple Instructions, Multiple Data. xiv, 26, 107, 120, 123, 177
MISD Multiple Instructions, Single Data. 107, 120
MPI Message Passing Interface. ix, x, xv, 26, 41, 111, 126, 130, 134, 163, 164, 169, 175, 177, 180, 181, 182, 187, 189

NUMA Non Uniform Memory Access. xiv, 107, 123

OpenMP Open Multi-Processing. ix, 26, 41, 111, 126, 130, 134, 177, 180, 181, 182, 183, 184, 187

PC Pedagogical Content. 22, 119
PDC Parallel and Distributed Computing. 42
PP Pedagogical Patterns. 58

R&D Research and Design. 28
RISC Reduced Instruction Set Computing. 146, 203
RTOS Real Time Operation System. 111, 164

SC Scientific Computing. *Glossary:* scientific computing, vi, xiii, 1, 3, 5, 6, 7, 14, 15, 17, 22, 23, 24, 25, 26, 27, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 46, 48, 60, 66, 68, 70, 71, 77, 81, 83, 84, 85, 103, 104, 109, 110, 111, 114, 116, 119, 137, 157, 175, 202, 249
SCE Scientific Computing Education. vi, 1, 5, 6, 7, 8, 10, 13, 14, 19, 22, 23, 25, 27, 34, 43, 66, 76, 119, 202
SCV Scope-Commonality-Variability. 65, 77
SI Scientific Inquiry. *Glossary:* scientific inquiry, vi, xiv, 1, 2, 3, 5, 6, 7, 8, 25, 27, 35, 43, 48, 55, 66, 67, 68, 71, 76, 82, 83, 84, 85, 95, 96, 97, 100, 104, 107, 116, 119, 141, 189, 202, 203
SIMD Single Instruction, Multiple Data. 26, 107, 120, 122
SISD Single Instruction, Single Data. 26, 107, 120
SLO Software Learning Object. *Glossary:* software learning object, vi, xii, 5, 7, 13, 16, 19, 22, 23, 24, 25, 26, 62, 63, 76, 137, 138, 188, 189, 190, 192, 193, 195, 196, 197, 202, 203, 205
SLOFTAP LR and SLOs for Teaching Advanced Parallelization. xvi, 189, 190, 192, 193, 195
SLOFTIS LR and SLOs for Teaching Introductory Stochastic. xvi, 189, 192, 193, 195
SMP Symmetric Multiprocessing. 126
STEM Science, Technology, Engineering and Mathematics. vi, 3, 5, 13, 14, 15, 17, 18, 19, 25, 31, 66, 87, 95, 99, 202
SWIG Simplified Wrapper and Interface Generator. 169, 175

TC Technological Content. 22, 28, 107, 119
TPACK Technological Pedagogical and Content Knowledge. vi, vii, viii, ix, xiii, 6, 22, 23, 24, 27, 28, 76, 202

UMA Uniform Memory Access. xiv, 107, 123

WIPO World Intellectual Property Organization. 28

Quickstart pathway

Table 1: A quickstart pathway

No	The relevance	Description	Reference to Figures	Reference to Sections
1	The research itself	The contribution of the author	4	1.5, 1.6
		DSR methods and implementation in the research	2, 3	
2	The case studies (experimental part) - samples of learning resources	Python based LR use case	99	6.4
		Python based LR instruction and didactics	45	4.5
		C based LR use case	111	6.5
		C based LR instruction and didactic	47	4.5
3	Educational solutions (theoretical part)	**The summary of the developed Design Principles**	42	4.3
		The Supportive Application & integration methodology:		
		Interdisciplinary approach	25	4.2
		DSR applications	28, 30	
		DSR didactic tool	38	
		Instruction and grounding	39	4.3
		The generalized feature model	48, 49	
4	Modeling solutions (theoretical part)	Explicit parallelization model		5.3
5	Modeling solutions (experimental part)	Theory and algorithms for queueing in series systems		6.3

1 Research context

1.1 Introduction

Nowadays research activities of university education in general and engineering education in particular are focused on a broad range of problems including philosophical and cognitive foundations of university and engineering education, topic-specific and general educational technology, organizational aspect of the educational process, and didactic research aimed to specific engineering disciplines. In spite of solid theoretical foundations and comprehensive practical input from the educational theorists and practitioners into each of these fields of study, it seems that there is a lack of interconnection between these specific research areas. As it could observed, many solid theoretical ideas and practical solutions in the field of didactic for engineering disciplines are mainly developed with minor or no attention to the educational technology. At the same time, the technology, considered as the universal one, is usually developed with serious attention to details and with little attention to the tools this technology should use in educational practice. This is especially true for such interdisciplinary and comprehensive educational field as Scientific Computing Education (SCE) within the university curriculum. SCE as a university discipline claims to be one of the most important within the scope of engineering education. The reason for such an attitude is following. As it is clear from the name of the topic, it deals with science, computing, and education. Going into details later in this study, it could be stated that science, scientific approach, scientific research, Scientific Inquiry (SI) and modern engineering educational technology are closely connected not only as stock-holding counter-agents of modern academic and scholar environments but also as important parts of the possible unifying holistic approach to modern engineering education.

One could hardly imagine a modern global world of engineering without non-stop questions and inquiries, experiments and solutions, innovations, pioneering engineering, and state-of-the-art technological approaches. All this, besides positive attitude to the participating parts, is based on scientific and technological inquiries and on modern computing as well. An obvious and not a trivial question arises: should one be taught Scientific Computing (SC)? The reasoning against could be like this: computing and informatics are taught in various separate courses and why are any additional models needed? Is there any reason to teach SI and should it be taught separately and (or) should it be taught within the scope of SC course? An affirmative answer is proposed – SC as an educational discipline should exist. Moreover, relevant educational technology should be developed, taking into account the comprehensive nature of the topic in the study. Plenty of environments, a variety of questions and inquiries are under the potential cover of this interdisciplinary in nature and wide-ranging in its scope educational topic. Unifying solutions, or at least some wide-ranging approaches, could serve a lot of scientists and educators, students, society and modern industry in general.

Innovations provide a challenging pathway for modern science and engineering. It is easy to declare, but not so easy to implement in practice and there are many obstacles on the way. Obviously, innovations are based on research, therefore, effective and innovations-oriented interdisciplinary research is needed. There is a need to merge different disciplines with different backgrounds, inner reasoning culture, and knowledge base. Interdisciplinary research team members should communicate and look for practical solutions. At the same time, research institutions should provide an effective interdisciplinary environment for support and management. Traditionally, to solve the described problems, some sort of managerial solutions are offered. New interdisciplinary educational programs are created, inner team communications are enhanced by better management solutions and similar "traditional" approaches based on directions and directives are promoted. Even if such solutions could be temporarily effective, the need for the new managerial decisions will arise with time, and such an approach will not solve the problem in general. The other possible solution – the self-directed environment should be designed. Universities should develop universal scientific culture including universal scientific approaches and unifying reasoning schemes. This will allow researchers from different disciplines communicate using the same scientific "Esperanto" and will provide the self-directed and self-organizing interdisciplinary environment for research and innovations.

Within this study, a SI centered model as a unifying solution for interdisciplinary education is proposed. SI is defined as an integrated and based on human-computer interaction activity of conducting (designing, developing and implementing) a set of simulations. Such simulation making activities are based on a set of conceptual models and are focused on conducting computer simulations for the topic of study. At the same time, this model is closely integrated with such cognitive processes as developing of the learner's understanding via simulative mental reasoning. Simulation as an artifact, allows using a set of well-developed Design Science Research (DSR) analytic techniques for its design and evaluation. Design Science Research and model-based simulation making skills could provide the required basis for SI centered pedagogy and become the universal solution for university interdisciplinary education. This practical and at the same time well theoretically grounded approach could satisfy both the industry and academia seeking for innovations and appropriate educational solutions. Therefore, the practical implementation of this model is a challenging task for the university and supporting educational systems. There is a need to rethink and redesign the present approaches to teaching and learning. However, this will provide a benefit and an advantage in the educational, research and global innovations markets.

1.2 Motivation

Diverse and interdisciplinary research is now a priority for high-ranked educational institutions. For example, the University of Turku (`www.utu.fi/en/`) stresses the importance of the interdisciplinary view on university education (`https://www.utu.fi/en/university/strategy-and-values/effective-research/Pages/home.aspx`); the important research question: is it possible to force interdisciplinarity without serious improvements of the classical approach to and style of university education? There are several practical reasons for a need of a new paradigm for university education, which enables teaching interdisciplinarity and its applications: Developing of interdisciplinarity enabling a unifying approach to university Science, Technology, Engineering and Mathematics (STEM) education; Focusing on and enabling of the learner-centered strategies in university education. At the same time, the next research questions are important: What is a unifying paradigm for interdisciplinary university education? What is the learner-centered methodology for SC education?

1.3 Problem statement

Scientific computing (after revision of the curriculum) could become a unifying discipline for the whole university (engineering, STEM) educational system. The possible methodology could be based on such well-known or well-developed approaches like SI, model-based simulations, DSR. Scientific Inquiry (in a broader sense) should be taught: how to design scientific model-based simulations (artifacts, which could be designed and evaluated using DSR methodology) to solve specific scientific problems. The unifying teaching paradigm for interdisciplinary university education could be formulated: SI (research) should be taught by means of making model-based computer simulations in various fields by implementing DSR methodology (as a teaching method for designing models and simulations) and using seamless approach for theoretical prerequisites. The meaning of SI is universal for all disciplines. At the same time, a general definition (a broader sense) could be provided: SI is an activity of conducting (making and using) scientific model-based simulations. The next important question to answer is – why models? The following argumentation could be provided:

(1) Theories of model-centered instruction and model-based education are well developed [1];
(2) Model as an artifact allows practical methods and analytic techniques of DSR and model-based system analysis to be used in research and education;
(3) There is a strong connection between cognitive activities (mental models) of students and activities of making (developing, programming and even using predeveloped) computer models [2];
(4) Model serves as a basis for model-based simulations.

Model-based simulations could be positioned as a basic tool within the provided methodology. There are several reasons for such an approach:

(1) There is a connection between mental simulations and computer simulations activities [3];
(2) In any case, simulations (of one or another type) are involved in the activity of any model development;
(3) Simulations is a kind of generalization of models;
(4) Simulations intersect with serious games and enable the constructionist environment and learner-centered education.

The research questions are formulated as follows:

(1) What is the generalized methodological framework for design and implementation of the relevant teaching and learning methods for SCE, which covers such major parts of the educational system like educational technology, instructional design, didactic tools?

(2) Focusing on general aspects of educational technology, what is the methodology for design of the university curriculum in general and STEM university curriculum in particular, focusing on interdisciplinary and research-based education? How could SC the discipline be integrated into the curriculum?

(3) Focusing on aspects of instructional design, how the teaching and learning process for SCE should be designed enabling SI and constructionist based education?

(4) Focusing on didactic aspects, how should the Learning Resources (LR) for SCE be designed?

(5) What are the appropriate computational models which are based on modelling of stochastic recurrences, which enable the relevant implementation of the LR in the form of Software Learning Objects (SLOs) for teaching introductory stochastics, basic probability distributions, limit theorems, queuing systems, hardware and software specific parallelization methodologies?

(6) How could educational tools which enable simulation-making and SI centered approach to a teaching and learning process, and promote practical knowledge of the relevant parallelization techniques, including hybrid computational platforms and big-data-related topics be designed? What are practical examples of implementations?

1.4 Research goal, objectives and tasks

(G) The goal of the research is to develop Design Principles (DP) for the design of the learning resources, including a set of Software Learning Objects for the effective teaching of Scientific Inquiry based Scientific Computing educational solutions. Achieving this goal, Scientific Computing Education and the relevant Scientific Computing teaching methods are positioned to be focused on teaching SI within a Science, Technology, Engineering and Mathematics interdisciplinary curriculum. The

Learning Resources (Software Learning Objects) are developed on the subjects of Introductory Stochastics and Parallelization. Making scientific model-based simulations is the key to model-centered and constructionist approaches to learning. This methodology should serve as a universal platform for designing of teaching methods aiming to enhance university interdisciplinary curricula by simulation-centred and problem-solving research-based educational activities.

(O) In achieving this goal, the research will address these objectives:

(1) To develop the supporting methodology for practical application and integration of the relevant learning resources;

(2) To develop the theoretical model for parallelization of Monte Carlo experiments for stochastic recurrences. Such theoretical model provides foundations for the development of appropriate computational models;

(3) To develop the relevant algorithms and software solutions for experimental validation of the law of the iterated logarithm for queues in series, which are based on the previously developed computational models for parallelization of stochastic recurrence.

For the research objectives:

(1) The supportive methodology should include a set of heuristics and feature models for teaching SI based Scientific Computing educational solutions. The process of the methodology development should be based on the Technological Pedagogical and Content Knowledge (TPACK) model including analysis the educational context, technological, content and pedagogical domains of the Scientific Computing Education domain and should be developed using a generalized Meta Analysis of Domain Features method, enabling to identify critical domain features, which are relevant to the goal of the research;

(2) Theoretical model for parallelization provides foundations for the development of appropriate computational models. Based on this, to develop the set of computational models for explicit parallelization of Monte Carlo experiment for stochastic recurrences. Such computational models provide foundations for computational experiments to be implemented during the experimental phase of the research;

    (3) The developed algorithms and software should provide a set of technological solutions for designing learning resources and for the implementation of sample learning resources.

(T) Tasks of the research are to provide sample educational solutions of implementation of the developed Design Principles and the supportive methodology in the form of a set of practical learning resources including Software Learning Objects within the scope of Scientific Computing Education. The aim of these sample Learning Resources is to present the practical examples of the Software Learning Objects for teaching SI based Scientific Computing:
    (1) teaching basics of stochastics;
    (2) teaching parallelization.

1.5 Research methods

1.5.1 Introduction

Design Science Research is used as the thesis research methodology for design and implementation of LR.

- The problems to be solved

(A) There is a need to specify Scientific Computing (SC) courses:
    (1) SC courses are too technical;
    (2) are mainly oriented in to mathematical and algorithmic foundations;
    (3) require many prerequisites;
    (4) hard to motivate students.

(B) to specify SC educational technology:
    (1) there is no vision of suitable educational technology (including the lack of solutions of how to develop the content and teach SC);
    (2) there is no vision on how to integrate SC to a broader curriculum.

- Motivation

  It is based on various opinions and literature review (presented in Section 3.3). The origin of problems with the present approaches to Scientific Computing Education (SCE):
  (A) improper didactic approaches and teaching techniques to SCE;
  (B) the lack of modern and focused on inter-disciplinary and research-based education approaches to SCE.

- Tentative solutions
  1. SCE should be based on teaching SI;
  2. SI is understood (in a broader sense) as a research activity based on scientific simulations (including and computer simulations);
  3. scientific simulations are based on computer models (artifacts);
  4. SCE is based on teaching of how to develop scientific simulations;
  5. simulation-based education is based on simulation-based cognitive reasoning processes;
  6. learner-centered educational technologies like co-mediated learning should be used;
  7. focus on DSR as a teaching technique.
- Artifact

  The aim of this study is to provide solutions (DP) of how to design and unify proper LR, Knowledge Objects (KO), and Learning Objects (LO) for simulation-based SCE. The general requirements for the methodology are:
  1. should provide a solution for SI and simulation-based SCE (as it was previously described);
  2. should be suitable for use within a learner-centered educational environment (support co-mediated learning technology, support the constructionist approach to learning).
- Evaluation

  It is based on formal DSR evaluation requirements including author related publications [4–8] in peer-review journals which are cited in Clarivate Analytics database and positive reviews related to these publications.

1.5.2 Implemented research methodology

The research by itself is based on well-known and well-described methodology – DSR [9–21]. 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 application is these are socio-technical domains. Not only technical aspects of the system but also social aspect and interactions should be considered. The relevance of the proposed methodology for the Computer Science (CS) educational domain is based on the next propositions. The domain:

(P1) Has a type of a socio-technical system;

(P2) Includes as participants (teachers, students, educational authorities, community, other stakeholders) as well as educational technology, instructional design methods, educational tools;

(P3) Educational tools are (mainly) artifacts including cognitive artifacts in the form of software-based LO (software as a learning object, educational software);

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. DSR is also known as methodology for doctoral research for IS and related topics [16, 18].

1.5.3 Design Science Research methodological formalization

Design Science Research methodological formalization are based on a number of heuristics [10], which are presented below. This set of heuristics is incorporated into the design framework (see Figure 1, reprinted from [9, p. 2]). This approach provides a systematic methodology for conducting of a research.

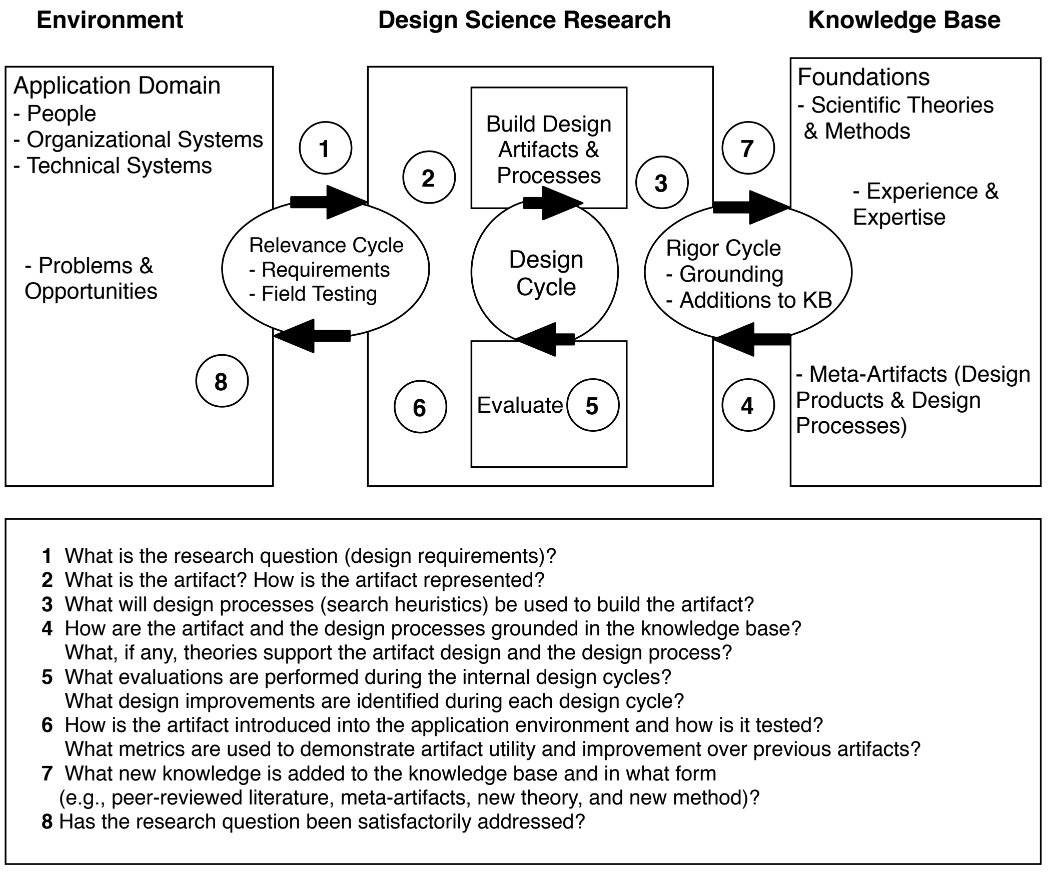
1. What is the research question (design requirements)?
2. What is the artifact? How is the artifact represented?
3. What will design processes (search heuristics) be used to build the artifact?
4. How are the artifact and the design processes grounded in the knowledge base?

4a. What, if any, theories support the artifact design and the design process?

5. What evaluations are performed during the internal design cycles? What design improvements are identified during each design cycle?
6. How is the artifact introduced into the application environment and how is it tested? What metrics are used to demonstrate artifact utility and improvement over previous artifacts?
7. What new knowledge is added to the knowledge base and in what form (e.g., peer-reviewed literature, meta-artifacts, new theory, and new method)?
8. Has the research question been satisfactorily addressed?

The inplemented research scheme and correspondence to DSR methodological formalization is presented in Fig. 2.

Conclusions: DSR as a methodology provides all the necessary formalization for the implementation of the research task – DP for development of LO for SCE. The presented design cycle provides the relevant formalization for utilization of inductive-abductive-deductive reasoning approach for conducting the research.

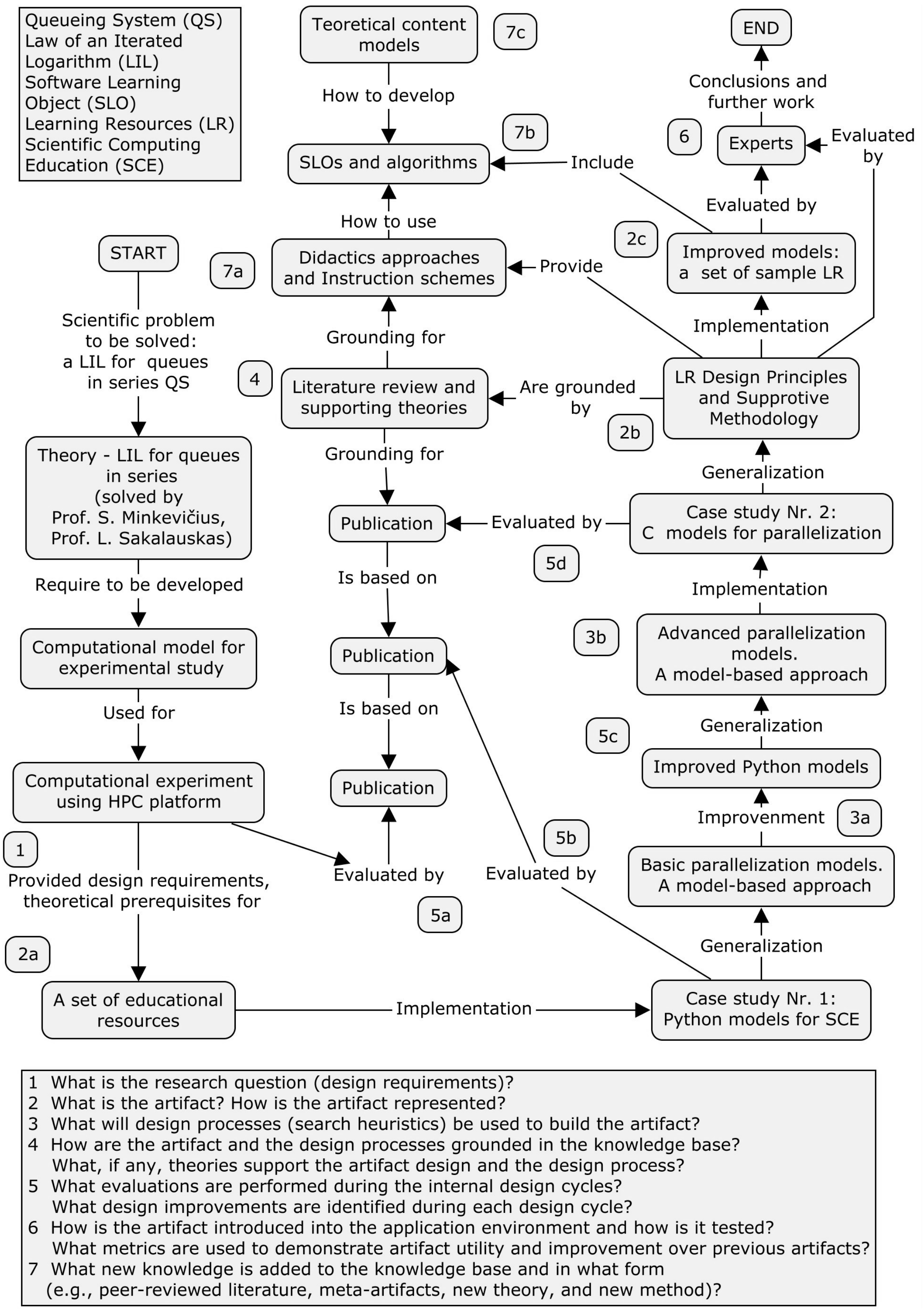


1 What is the research question (design requirements)?
2 What is the artifact? How is the artifact represented?
3 What will design processes (search heuristics) be used to build the artifact?
4 How are the artifact and the design processes grounded in the knowledge base? What, if any, theories support the artifact design and the design process?
5 What evaluations are performed during the internal design cycles? What design improvements are identified during each design cycle?
6 How is the artifact introduced into the application environment and how is it tested? What metrics are used to demonstrate artifact utility and improvement over previous artifacts?
7 What new knowledge is added to the knowledge base and in what form (e.g., peer-reviewed literature, meta-artifacts, new theory, and new method)?
8 Has the research question been satisfactorily addressed?

Figure 1: Design Science Research design cycle. Adapted from [9, p. 2]

1.5.4 Methodological requirements and implementation

The summary of characteristics, DSR formal requirements and the relevant implementation are presented in Table 2.


Queueing System (QS)
Law of an Iterated Logarithm (LIL)
Software Learning Object (SLO)
Learning Resources (LR)
Scientific Computing Education (SCE)
START
Scientific problem to be solved: a LIL for queues in series QS
Theory - LIL for queues in series (solved by Prof. S. Minkevičius, Prof. L. Sakalauskas)
Require to be developed
Computational model for experimental study
Used for
Computational experiment using HPC platform
1
Provided design requirements, theoretical prerequisites for
2a
A set of educational resources
Implementation
Case study Nr. 1: Python models for SCE
Generalization
Basic parallelization models. A model-based approach
Improvement
3a
Improved Python models
Generalization
5c
Advanced parallelization models. A model-based approach
3b
Implementation
Case study Nr. 2: C models for parallelization
Generalization
LR Design Principles and Supprotive Methodology
2b
Are grounded by
Implementation
Improved models: a set of sample LR
2c
Evaluated by
Experts
6
Evaluated by
Conclusions and further work
END
Literature review and supporting theories
4
Grounding for
Didactics approaches and Instruction schemes
7a
Provide
How to use
SLOs and algorithms
7b
Include
How to develop
Teoretical content models
7c
Grounding for
Publication
Evaluated by
5d
Is based on
Publication
Is based on
Publication
Evaluated by
5a
Evaluated by
5b

1 What is the research question (design requirements)?
2 What is the artifact? How is the artifact represented?
3 What will design processes (search heuristics) be used to build the artifact?
4 How are the artifact and the design processes grounded in the knowledge base?
What, if any, theories support the artifact design and the design process?
5 What evaluations are performed during the internal design cycles?
What design improvements are identified during each design cycle?
6 How is the artifact introduced into the application environment and how is it tested?
What metrics are used to demonstrate artifact utility and improvement over previous artifacts?
7 What new knowledge is added to the knowledge base and in what form
(e.g., peer-reviewed literature, meta-artifacts, new theory, and new method)?

Figure 2: Generalized scheme of the research and correspondence to DSR formalization

Table 2: Formal requirements and implementations in the research

Characteristics	**DSR Requirements**	**Implementation**
Objectives	Develop artifacts that enable satisfactory solutions to practical problems	Develop Design Principles and the Supportive Application and Integration Methodology (DPSAIM) for design of LR for SCE
Main activities	Define the problem; Suggest; Develop; Evaluate; Conclude	**Problems in SCE:** • There is no documented LR DPSAIM for SCE **Suggestion (research task):** • To develop LR DPSAIM for SCE based on constructionist paradigm • To develop practical examples of LR for SCE 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 DPSAIM for SCE is developed. The developed DPSAIM implements the constructionist paradigm • Ex-ante and ex-post evaluations are fulfilled • Example of LR and case studies are presented
Results	Artifacts (constructs, models, methods, DP, instantiations) and improvement of theories	Constructionist LR DPSAIM for SCE; Samples of LR for SCE and case studies.
Type of knowledge	Prescriptive	How to design LR enhancing interdisciplinarily and seamless approach to SCE

Table 2: Formal requirements and implementations in the research

Researcher's role	Builder and/or evaluator of the artifact	The scope of the research: analysis of the current state in the field of SCE; logical analysis, theoretical assertions, and confirmations of psychological, technological and instructional relevance and theoretical basics for the implemented LR DPSAIM; developing of the methodology; developing of sample LR; demonstrations via case studies
Empirical basis, Implementation	Not mandatory	Not implemented
Evaluation of results	Applications; Simulations; Experiments	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 for problems of developing of interdisciplinary curricula for inquiry-based university education

1.6 Summary of results and implementation

1.6.1 Application domain, research question

The application domain hierarchy consists of a set of sub related domains within the general domain of university educational system:

- University educational system;
- E-learning;
- STEM university curriculum;
- Interdisciplinary university curriculum;
- Scientific computing education (discipline);
- Simulation and modelling with computers.

The research question (design requirements) is DPSAIM for SLOs design, which support the relevant contructionistic methodology for SCE. The DPSAIM should provide possibility of the constructionist approach to interdisciplinary university education in general, and in particular, provide formalization for SCE:

- from the perspectives of educational technology;
- from the perspectives of instructional design;

- from the perspectives of Educational Content (EC) design: provide a methodology for design of LR for SCE.

1.6.2 Description of the research results

Formalization for the developed DPSAIM is provided. This formalization include implementation guidelines, recommendations, designing criteria, a set of heuristics, case studies and practical examples for SCE related domains:

- Educational technology domain;
- Instructional design techniques;
- Development of Learning Resources (LR) and Knowledge Objects (KO);
- Implementation of LOs for SCE.

From the perspectives of educational technology: the innovative methodology of designing of interdisciplinary STEM curriculum is proposed. The methodology is based on the proposition of the central place of SC within university STEM curricula. This provides the basis for the design of interdisciplinary and innovations oriented learning environment.

From the perspectives of instructional design:

(1) the revised definition of SCE is developed. The developed definition provide a basis for implementation of innovative instructional design techniques;

(2) a methodology for structuring of the EC is provided;

(3) an instructional approach and innovative epistemic educational tool based on deductive and circumscriptive reasoning techniques are implemented.

From the perspectives of the content design:

(1) the Design Principles (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 software learning objects are designed.

1.6.3 Grounding with the knowledge base

The proposed educational solutions are grounded in a systematic literature review. Table 3 provides a summary of topics, which were systematically observed for grounding purposes.

Table 3: Grounding with a literature review

Section	**Observed topics**	**Number of literature sources**
Motivation and development for interdisciplinary university curriculum (STEM, Scientific Computing)	Interdisciplinary innovations; Interdisciplinary research; Interdisciplinary education; Creating interdisciplinary environment	20
Scientific computing	Meaning and definitions; Existing approaches to SC education	29
Educational technologies	Constructivism and Constructionist approaches to learning; Constructivism and Constructionist approaches to Computer Science Education (CSE); Educational technologies for university education in general	22
Instructional approaches	Model-based approaches in education; Model-centered co-instruction and co-mediated constructionism; Model-based education; Simulations-centered approach; Scientific models and model-based scientific simulations; Simulation – the revised definition; Simulative Scientific reasoning	79
Formalization	Learning objects for SC and CSE; Classification of learning objects; Learning objects for CSE; Software program as a learning object; E-learning and the concept of Smart Learning Objects.	38

1.6.4 Grounding with the supporting theories

Systematic grounding with supporting theories is provided in Table 4.

Table 4: Grounding with supporting theories

Name of theory	Contributors	Literature
Pragmatism	C. Peirce	[17, 22]
Constuctivism	J. Piaget	[23]
Constructionism	S. Papert	[24, 25]
Model-centered instruction	A. S. Gibbons	[26]
Simulative reasoning theory	P. Johnson-Laird	[27]
Model based scientific reasoning theory	N. J. Nersessian, L. Magnani, P. Thagard	[28–32]
Theory of Smart Learning Objects	V.Štuikys	[33, 34]
Theory of Design Science Research	H.Simon, A.Hevner, V.Vaishnavi, W.Kuechler	[10, 9, 11, 35]

1.6.5 Evaluations

Evaluation of the research results is based on formal requirements to evaluation procedures under DSR methodology formal requirements [16, 36, 37]. 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 are based on demonstrations and case studies (see Figure 3, adapted from [37, p. 14]).

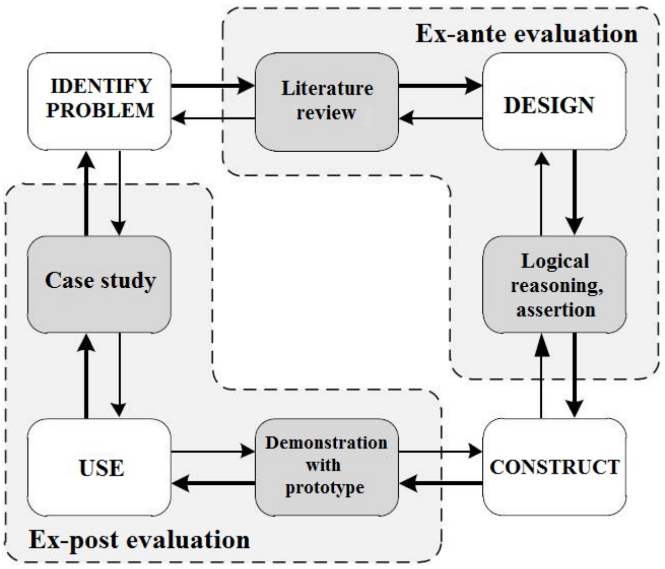


Figure 3: Evaluation of the research results. Adapted from [37, p. 14]

1.6.6 Example of the implementations

Examples of implementation the developed methodology in the form of Software Learning Objects are presented in Section 6.3.

1.7 Scientific contribution of the research

The research provides the Design Principles and the Supportive Application and Integration Methodology (DPSAIM) for application and integration of the educational resources in the form of software learning objects for teaching SC in general and teaching introductory stochastics and parallelization in particular.

(1) The educational solutions for teaching scientific inquiry based scientific computing education using Design Science Research analytical techniques are introduced. The presented educational solutions allow developing learner-oriented Educational Content (EC), which focuses on the constructionist approach to learning, thus improving the efficiency of educational process enhancing students' scientific inquiry, professional knowledge, and skills. Design Science Research provides a relevant set of analytical techniques that enable the creation of a unifying approach to such strongly interdisciplinary in its nature field like scientific computing education. Such unifying approach, in the form of the developed Design Principles and the Supportive Application and Integration Methodology, provides a base for modernization of existing and creation of innovative university educational programs, enhancing diversity and interdisciplinarity in research and modern university education. The presented educational solutions are aimed at university STEM education that is focused on enhancing interdisciplinary and innovations in the modern university curricula.

(2) The developed educational solutions use a scientific inquiry centered approach and provide a set of practical educational techniques and unifying teaching methods. Relying on Peirce pragmatism and principles of embodied cognition this study suggests a bridge from the cognitive theoretical constructions to practical educational techniques and methods, which are pragmatically useful and applicable for educational practitioners. The presented approach is based on such well theoretically grounded and practically effective solutions as the Design Science Re-

search methodology, a model-centered approach to instructional design, model-based teaching methods, problem-solving and constructionist didactic approaches. For the purpose of this study, scientific inquiry centered approach is understood as an educational process of designing (developing, testing, evaluating, and improving) model-based scientific computer simulations. Computer simulations, underlying software, computational and conceptual models as cognitive artifacts allow the Design Science Research methodology to be implemented in the form of a practical teaching tool. Accordingly, an appropriate educational environment that is based on pre-designed multifaceted models and a seamless approach to theoretical prerequisites is introduced.

(3) Several practical examples of the implementation of the developed educational solutions are provided – the Design Principles for designing Learning Resources for scientific computing education and the supportive application and integration methodology – in the form of a set of practical Learning Resources. The first one covers the topic of introductory statistics – Teaching introductory stochastics and queueing models with Python. The second one covers queueing in series systems and probability topics and could be positioned within the scientific computing or programming curricula – Teaching parallelization methods with C.

1.8 Scientific novelty

(1) Educational solutions – the Design Principles for the development of Learning Resources for teaching scientific inquiry based scientific computing education including the Supportive Application and Integration Methodology – are developed. A comprehensive analysis of the scientific computing educational domain from perspectives of constructionist approaches to education within STEM university curricula is provided;

(2) Model-centered instructional design methods for teaching scientific inquiry and scientific computing are introduced;

(3) A didactic model of introductory stochastics and parallelization using stochastic recurrence models is introduced.

1.9 Engineering novelty

(1) An innovative computational model for explicit parallelization of stochastic recurrences is developed;
(2) An innovative computational model for experimental confirmation of the limit theorem for the system of queues in series under heavy-traffic conditions is developed;
(3) An innovative model-centred framework for designing of learning resources for scientific computing education is developed;
(4) An innovative set of learning resources in the form of Software Learning Objects for teaching introductory stochastics is developed;
(5) An innovative set of Learning Resources in the form of Software Learning Objects for teaching parallelization is developed.

1.10 Practical significance of the achieved results

(1) The presented scientific results allow developing an innovative curriculum for SCE. Such a curriculum enhance interdisciplinarity and is focused on research-based and constructivists educational methods. The research provides a universal framework of the integral view at SCE with STEM university curricula covering educational technology, instructional design, and didactics of SCE.
(2) The presented engineering results allow the development of practically applicable learning resources for SCE in the form of SLOs under the constructionist paradigm.

1.11 Defending claims

(1) The developed Design Principles for design of educational resources allow design and development of the Learning Resources for teaching scientific inquiry based scientific computing education including the relevant Software Learning Objects which corresponds to design requirements – model-centred education methods and the constructionist education paradigm. The developed supportive methodology is appropriate for the application and integration of the relevant learning resources within interdisciplinary university curricula.
(2) Developed parallelization algorithms for Monte Carlo experiments for

stochastic recurrence models are relevant for designing learning resources in the form of Software Learning Objects for teaching scientific inquiry based introductory stochastics and parallelization.

The generalized view on the contribution of the thesis author is presented in Figure 4.

1.12 Approbation and publications

The results of the dissertation were presented and discussed at the following national and international conferences:

- 4th Doctoral Consorcium on Informatics Engineering Education Research, Druskininkai, Lithuania, 2013.12.03–07.
- 10th International Seminar on Informatics Contests, Druskininkai, Lithuania, 2014.06.03–06.
- 5th Doctoral Consorcium on Informatics Engineering Education Research, Druskininkai, Lithuania, 2014.11.26–30.
- ITiCSE 2014 19th Annual Conference on Innovation and Technology in Computer Science Education, June 23–25, 2014, Uppsala, Sweden.
- 8th International conference on E-learning, The University of La Laguna,Tenerife, Spain, 2014.09.06–14.
- 6th Doctoral Consorcium on Informatics Engineering Education Research, Druskininkai, Lithuania, 2015.12.08–12.
- ISSEP 2017, The 10th International Conference on Informatics in Schools, November 13–15, 2017, University of Helsinki, Helsinki, Finland.

The main results of the dissertation were published in the following papers:

- Dolgopolovas V, Dagienė V, Minkevičius S, Sakalauskas L. (2015) Teaching Scientific computing: a model-centered approach to pipeline and parallel programming with C. Scientific programming, 2015 Jan 1;2015:11.
- Dolgopolovas V, Dagienė V, Minkevičius S, Sakalauskas L. (2014). Python for Scientific computing Education: Modeling of Queueing Systems. Scientific Programming, 2014 Jan 1;22(1):37–51.
- Minkevičius S, Dolgopolovas V, Sakalauskas L (2014) A law of the iterated logarithm for the sojourn time process in queues in series Method-

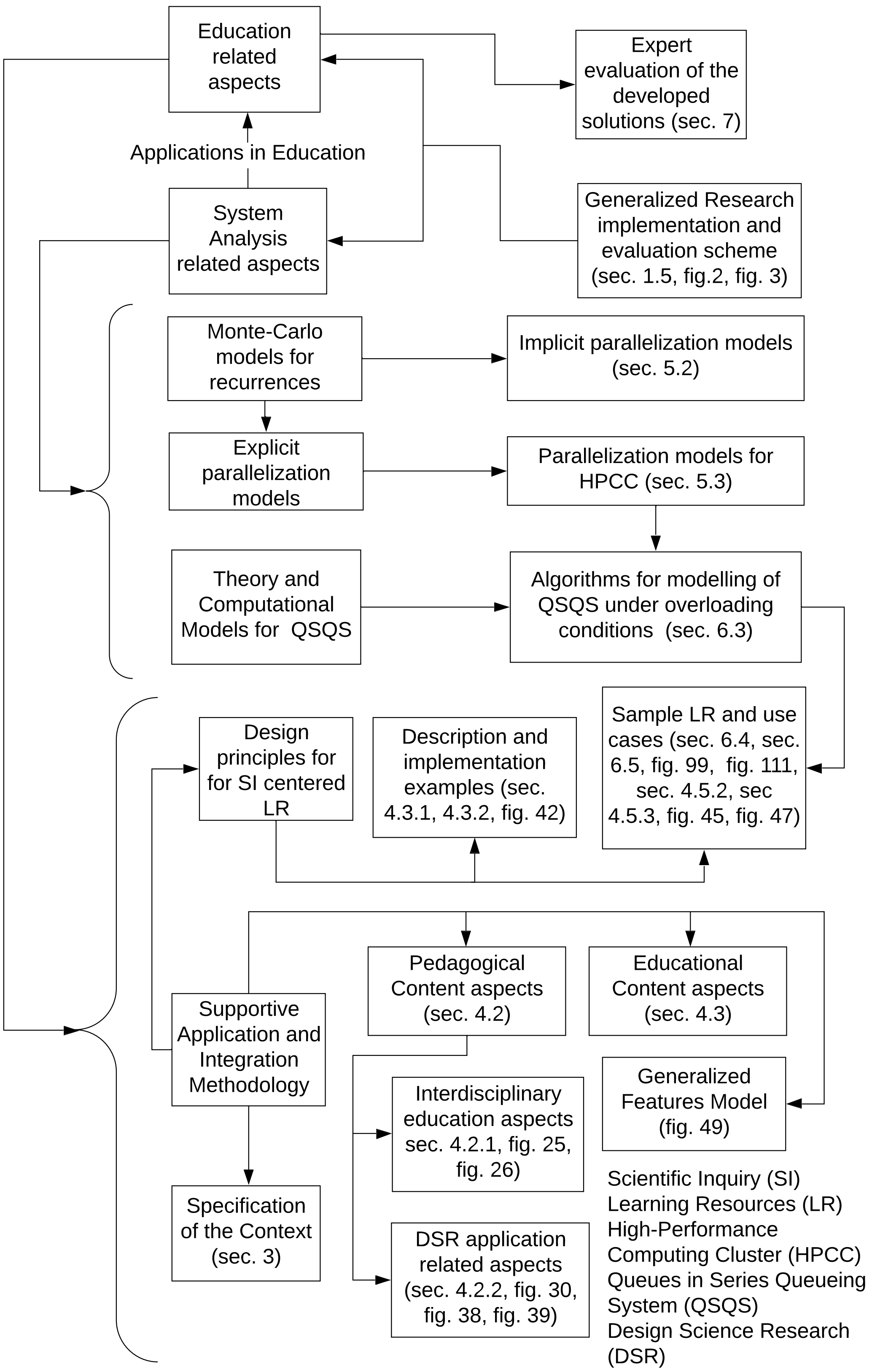

Education related aspects
Expert evaluation of the developed solutions (sec. 7)
Applications in Education
System Analysis related aspects
Generalized Research implementation and evaluation scheme (sec. 1.5, fig.2, fig. 3)
Monte-Carlo models for recurrences
Implicit parallelization models (sec. 5.2)
Explicit parallelization models
Parallelization models for HPCC (sec. 5.3)
Theory and Computational Models for QSQS
Algorithms for modelling of QSQS under overloading conditions (sec. 6.3)
Design principles for for SI centered LR
Description and implementation examples (sec. 4.3.1, 4.3.2, fig. 42)
Sample LR and use cases (sec. 6.4, sec. 6.5, fig. 99, fig. 111, sec. 4.5.2, sec 4.5.3, fig. 45, fig. 47)
Pedagogical Content aspects (sec. 4.2)
Educational Content aspects (sec. 4.3)
Supportive Application and Integration Methodology
Generalized Features Model (fig. 49)
Interdisciplinary education aspects sec. 4.2.1, fig. 25, fig. 26)
Specification of the Context (sec. 3)
DSR application related aspects (sec. 4.2.2, fig. 30, fig. 38, fig. 39)
Scientific Inquiry (SI)
Learning Resources (LR)
High-Performance Computing Cluster (HPCC)
Queues in Series Queueing System (QSQS)
Design Science Research (DSR)

Figure 4: Generalized view on the contribution of the thesis author

ology and Computing in Applied Probability, 2016 Mar 1;18(1):37–57.

- Dolgopolovas V, Jevsikova T, Dagienė V, Savulionienė L. Exploration of Computational Thinking of Software Engineering Novice Students Based on Solving Computer Science Tasks. International Journal of Engineering Education, 2016 Jan 1;32(3):1–10.
- Dolgopolovas V, Jevsikova T, Dagienė V. From Android games to coding in C – An approach to motivate novice engineering students to learn programming: A case study. Computer Applications in Engineering Education, 2018 Jan 1;26(1):75–90.

1.13 Outline

The dissertation consists of several major parts. The Research Context section (Section 1) presents introductory topics including an overview of the research methods, research findings, and results. The research methodology is covered in detail including formal requirements and implementation in the research.

Problematics of the implementation section (Section 2) describes the main topics covered in the research focusing on the requirements of the TPACK model. This section specifies the main parts of the model including technological, content, and pedagogical domains.

Section 3 describes the context topic of the TPACK model in detail. This section covers SC, SCE and related topics from the point of educational technologies, teaching methods, requirements for SLOs and didactic formalization combined with a systematic literature review. This part covers such important topics as motivation for the research; discussion on the SCE scope and definitions; discussion on educational technologies; approaches to instructional design; formalization techniques for design of SLOs; discussion on didactic approaches to SCE.

Section 4 provides detailed meta analysis of domain features of the SCE domain. The focus is on the Pedagogical Content (PC), Educational Content (EC), and Technological Content (TC) knowledge domains. The section provides a brief description of practical implementation, which is based on the provided analysis.

Section 5 covers theoretical aspects of the construction of SLOs for teaching parallelization based on stochastic recurrence models. The advantage of such an approach is that stochastic models allow the implementation of the process of multidimensional modeling during the construction of computational models and algorithms. At the same time, recurrences allow the implementation of additional dimensions for computational models under design. Two types of models are considered:

(I) implicit models, generally implemented by categorial data types within the functional programming paradigm. For such models parallelization is considered to be an automatic process, based on software sceletons. In the research such models are not studied in detail, and this study is positioned as a topic for future research;

(E) explicit models, based on explicit parallelization techniques. These models are studied in detail.

Section 6 covers the experimental part of the research. First, it studies and develops a big data computational model and algorithmic solution for experimental research of the limit behaviour of queues in series. Then the detailed results of the computational experiment are provided. Later, it describes two case studies of the implementation of LR for SCE and teaching parallelization, which are based on the developed computational model of queues in series. The first case study focuses on the topics of design and implementation of SLOs in an introductory SC course. The next one covers the topics of design and implementation of SLOs for teaching parallelization. Practical examples of SLOs are presented in the Appendices section.

Section 7 provides the author and expert evaluations and includes: specification of the developed sample educational resources, outline of the evaluation methodology, the author evaluations of the research results, outline of the expert evaluation.

Please note that figures, tables, formulas, theorems, algorithms, listings, statements, proofs without citations are made, formulated, proved by the author of the dissertation.

2 Problematics of the implementation of the TPACK model in the scientific computing educational domain

2.1 TPACK framework for scientific computing education

2.1.1 Introduction

The TPACK model for the systematization of the research content is implemented. The aim of this is to provide the description and specification of the content specific aspects, specifying definite feature, enabling the process of design of SLOs for SCE. The next remark should be made: a specific view on teaching with technology is implemented. In the context of this research, teaching with technology actually transforms into teaching technology, and the technological domain is presented in the content of teaching. Both technological and content knowledge domains are intersected forming an educational environment based on the common features of both domains.

The modeling methodology is based on the TPACK framework, based on which the generalized research content can be specified as:

(i) teaching theories for pedagogical knowledge;
(ii) educational environments for technological knowledge;
(iii) SC curriculum for content knowledge.

The TPACK model itself is presented in Figure 5 (reprinted from [38, p. 9]). One of the most important parts of the model, as well as one of the most important tasks to specify during the process of design of SLOs, is the proper understanding and specification of the features of the context. From the point of view of the pragmatist researcher (here the word "pragmatists" is understood as the description of a research paradigm [39, 22, 40–42], there exist some subsets of context features (sub-contexts), which are practically important during the design process of SLOs. Such sub-contexts include:

(1) interdisciplinarity university education in general and aspects of interdisciplinary university education from the point of view on interdisci-

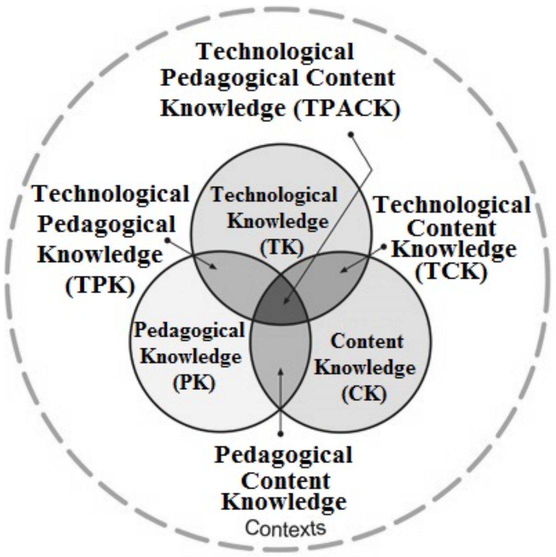

Technological Pedagogical Content Knowledge (TPACK)
Technological Pedagogical Knowledge (TPK)
Technological Knowledge (TK)
Technological Content Knowledge (TCK)
Pedagogical Knowledge (PK)
Content Knowledge (CK)
Pedagogical Content Knowledge
Contexts

Figure 5: TPACK model. Reprinted from [38, p. 9]

plinary innovations and research;

(2) aspects of SC education, including descriptions of the existing approaches to SC education in general;

(3) general aspects of educational technologies focusing on the constructionist approach including CS educational aspects;

(4) model-based approaches in education. These features provide the context for formulating DP of proper design of SLOs and proper didactic solutions;

(5) simulations in education. It is important to consider simulations not only as a technological or didactic tool but as an integral part of a learning/teaching context, including such cognitive processes as simulative reasoning and grounding;

(6) common features and intersection between CS and SC educational domains;

(7) aspects of SI and its place in STEM and engineering education;

(8) didactic aspects of using SLOs in SC education, including possible formalization using the DSR methodology.

2.1.2 Technological domain

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 SCE are:

(1) hardware architectures in general;
(2) SC and parallelization specific aspects of hardware architectures in particular;
(3) software engineering technologies in general; this 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; this technology could be promoted as a basic technology for design and implementation of SLOs.

Parallel computations are based on the next hardware architectures [43, 44]: Single Instruction, Single Data (SISD), Single Instruction, Multiple Data (SIMD), and Multiple Instructions, Multiple Data (MIMD). Multiple Instructions, Multiple Data machines are commonly divided into Shared-memory and Distributed-memory architectures. 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. The technological domain from the point of view on hardware features is studied in Subsection 5.1.2. Software developments (programming) methods include explicit parallelization tools for distributed memory – Message Passing Interface (MPI) and shared memory – Open Multi-Processing (OpenMP) implementations.

2.1.3 Content domain

In the general form, the content is based on the model of the stochastic recurrence. For example, the model is widely used in the queueing theory [45]. The most general model is the multi-variable, non-linear stochastic recurrence model. The solution for such a model will depend on the concrete form of non-linearity. For us it is the most interesting case is the model with linear and parametric non-linear parts. Depending on the value of the parameter, it could be possible to transform a non-linear part to its linear representation. Stochastic features of the model enable using such relevant modeling techniques as the Monte Carlo modeling method, which enables designing of models for studying parallelization. The content domain, focusing on content domain features, is studied in Subsection 4.3.

2.1.4 Pedagogical domain

The pedagogical domain is based on three major parts: educational technology, instructional design, and didactic aspects of SC education. Educational technology as related to SC education is focusing on the next questions:

(1) why is teaching SC important?
(2) could it be possible to use SCE as a part of a interdisciplinary curriculum?

The next important topic is the relevance of the instructional design. What are appropriate methods for SCE? Here the main focus could be made on studying methods of computer simulation with applications in education. How could computer simulations could be implemented and incorporated improving interdisciplinary and innovations focused educational methods?

Didactics aspects include the next questions: if there is a focus on SI and simulations, which didactic approaches are suitable for teaching SC and at the same time enabling an interdisciplinary approach to curricula design? The pedagogical domain is studied in Subsection 4.2.

2.2 Conclusions

The TPACK model provides a relevant framework for systematizing research context in the form of a definite structure. This could serve as a meta-structure for research context focusing on the three major areas: educational technology, instructional design and didactic aspects of teaching SC. As the result, the main directions for the research could be specified.

(E) Specifying an EC domain, the main focus should be on the next features:
    (E1) educational technology could be studied from the point of view on interdisciplinary university education as on the unifying paradigm within university curricula;
    (E2) instructional design could be focused on teaching/learning process which enables students' activities of designing simulations; the process of designing simulations could be positioned within the educational framework, which enables the relevant process of the SI;
    (E3) didactic approach could be focused on appropriate formalization which enable a suitable arrangement of the previously described teaching/learning process; the main focus could be on the cogni-

tive aspects of the educational process, including such important features as scaffolding and grounding.

(T) Specifying TC domain, the main focus could be done on:
- (T1) appropriate and parallelization enabling hardware architectures;
- (T2) relevant software engineering methods, enabling model-centered approaches for software development;
- (T3) relevant parallelization enabling software development/programming tools;
- (T4) relevant big data processing/programming/presentation tools;

(C) Specifying the content domain, the main focus could be on teaching/learning aspects of:
- (C1) stochastic and theoretical aspects of introductory stochastics and probability;
- (C2) theoretical and computational aspects of introductory recurrences and stochastic recurrences;
- (C3) distributions, limit theorems, and stochastic processes in the basics of probability theory;
- (C4) Monte Carlo modeling methods and the relevant computational aspects of Monte Carlo modeling;
- (C5) basic queueing theory, main parameters, and characteristics of queues, queues in series, queueing networks and computational aspects of queueing models.

3 Specification of the context for the scientific computing TPACK model

3.1 Enhancing interdisciplinarity

3.1.1 Interdisciplinary innovations

There is a need for interdisciplinary innovations. New areas and applications could be uncovered by crossing the boundaries including academics, business, government, society [46, p. 3]. Globalization and the knowledge economy requires more and more innovations in various fields, therefore interdisciplinary innovations could be by no means an advantage in the competition-driven economy. Generally, the importance of innovations and high quality Research and Design (R&D) activities is undoubted [47]. Multidisciplinary and Interdisciplinary innovations are mentioned among key factors for appropriate R&D policies for global innovations success. World Intellectual Property Organization (WIPO) reports [47, p. 13]: "Calls for proposals could, more often, be jointly issued by multiple countries, particularly when convening large-scale, multidisciplinary programs". The world leaders for global Innovations implement a strategy focusing on interdisciplinary innovations. For example, Singapore, one of the world global innovations leaders (world rank 6 in 2016 [47]), fosters the cooperation between leading universities and major industry players in R&D focusing on inter - and transdisciplinary. "Singapore recognizes that the greatest impact of innovation is often found at the convergence of different research fields and professions" [48, p. 137]. Finkel and Bell report about the Australian challenge to overcome: "…the growing complexity of science and technology, which requires greater international and inter-disciplinary cooperation" [49, p. 141].

3.1.2 Interdisciplinary research

Obviously, to move on with interdisciplinary innovations, interdisciplinary research needed to be implemented. The industry and academia are expecting increasing efforts for interdisciplinary research of all types. Leading universities announce the priority for interdisciplinary research in their development strategy. Stanford University [50] reports about 18 interdisciplinary "institutes span school boundaries, providing a physical and intellectual intersection between disciplines where new ideas emerge and innovative research across the humanities and sciences can happen". Cambridge University focus on interdisciplinarity within the research strategy: "Strategic Research Initiatives and Networks build on areas of existing research strength by bringing together a critical mass of expertise from across the Schools, with four key aims: to address large-scale multi-disciplinary research challenges; to strengthen research collaborations and knowledge transfer across disciplines" [51]. The University of Turku announces in the strategy for 2016 – 2020: "…Our strongest fields of research form the basis for innovative and interdisciplinary projects.… We allocate resources for creating versatile research communities and interdisciplinary intersections" [52]. Townsend et al. [53] report about a single case of a university in the

United Kingdom describing the university attitude and supporting organizational structure for interdisciplinary research. The above set of examples shows a widespread interest of universities in interdisciplinary research that is supported by practical managerial and organizational activities.

Another motivational factor is the high ranking of interdisciplinary publications (see Figure 6, reprinted from [54]). As an illustrative example, the Keck Futures Initiative [55] awarded 9 interdisciplinary grants (2004−2015) for overall 132 interdisciplinary research projects. What are the challenges? First, interdisciplinary innovation is based on teamwork. Team members bring a different background and different knowledge. Blackwell et al. report: "Different disciplines often have different core values, and have grown together as social groups precisely because of the shared values within each discipline. In order for a new interdisciplinary team to become effective, that team must develop shared values and culture" [46, p. 3].

Boundaries arise between various scientific approaches and disciplines like mathematics, law, medicine, engineering, history, biology, and many others [46, p. 15]. Interdisciplinary innovations and teamwork are closely connected; therefore, the problem is to combine disciplinary knowledge of the team members who came from different disciplines. The next remark should be made. There are several well-known models for the general organizational structure of the research process [46, p. 37]:

- **Multidisciplinarity**: "Researchers in different disciplines work in parallel and exchange knowledge in order to work on a shared goal. Each researcher's objectives are still determined by their discipline and results are reintegrated into this separate disciplinary context";
- **Interdisciplinarity**: "Researchers in different disciplines work towards a common goal in such a way that they cross subject boundaries and integrate knowledge from other disciplines. Disciplinary knowledge is transformed through this process such that new and independent theories and methods are created";
- **Transdisciplinarity**: "Involves academic researchers from different disciplines and non-academic participants who work together towards a common goal. Like interdisciplinarity "integration" is a key word in accounts of transdisciplinarity, but here it involves the breakdown of epistemological barriers not only at the level of disciplines but also at the level of institution."

Generally, in literature and among officials and scientists these approaches are usually considered as synonyms of the similar research activities based on more than one scientific fields, like in the case described by [53]. Different models require different approaches to teamwork, problem solving and collaborations, and that could become a reason for possible difficulties, which are model-dependent, and misunderstandings for the team members with different backgrounds. In spite of this, for the purpose of this study, the term "interdisciplinarity" is used as a general term describing all of the previous models, unless the definite model is not indicated in the text.

Traditionally, "classical" approach to research, focused on a research in a single area of knowledge, has been dominating for many years in the academy. "…is clear, then, that a move towards interdisciplinary collaboration requires a sea-change both in how researchers think about the relationships between individual disciplines and in how these relationships are supported" [56, p. 16]. Research institutions should "…provide suitable training to promote interdisciplinary understanding…" [57, p. 109]. Summarizing, two main obstacles are traditionally mentioned in literature:

(1) the problem of the inner-team communication;

(2) the lack of institutional support.

Various solutions are proposed. To overcome the problem of the inner-team communication, a suitable and interdisciplinary oriented training should be provided. The main tendency is to join several single disciplines into one "inter discipline" using the already existing scientific input (see for example [58]). Such "merging" approach could be effective, but this could introduce another set of problems. At the time, there will be a need to merge this newly merged discipline with another discipline and so on. It will not solve the problem in general. There is a need for the unifying approach that will enable a kind of a "seamless" and self-directed integration between researchers and research teams. Generally, all university STEM education should be transformed into teaching and learning of such interdisciplinary communication principles, and the language for this interdisciplinary communication, a kind of "interdisciplinary Esperanto" should be developed.

3.1.3 Interdisciplinary education

Is there any need for multidisciplinary, interdisciplinary education? What are the benefits? What are the challenges? Smith [59] reports about the

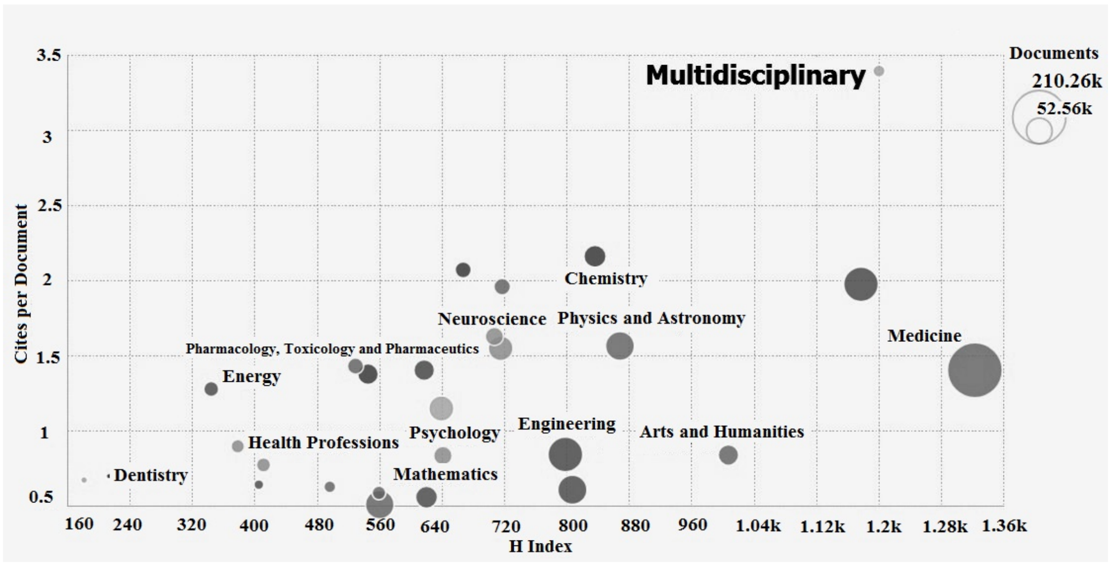


Figure 6: H-factor for US journals 2015. Reprinted from [54]

practice of curriculum redevelopment in the University of Southampton. The traditional, single discipline-based curriculum is constrained by accreditation requirements and generally, is focused on the content [59, p. 2]: "This focus on the delivery of content may lead to a perceived need to 'cover' a specified set of outcomes within the curriculum. This can then become a straitjacket that inhibits innovation. It may also lead to a preoccupation with modes of delivery and assessment, rather than on education in its broader sense. In turn, this may lead to a compartmentalization of learning and to emphasizing the need to 'get through' the necessary subject matter. Students become trapped in subject silos and may become preoccupied with outcomes, rather than being exposed to new and different ways of thinking". After some time and due to the increasing specialization of the "traditional" teachers, the content of the curriculum become more and more specialized as well. On the contrary, as was described earlier, the nature of the research becomes more and more interdisciplinary. Therefore, the redevelopment of the curriculum could serve as a possible solution [59]. Generally, many efforts should be used to develop an interdisciplinary curriculum in practice. But as a result, many advantages could be gained [60, p. 80]: "Interdisciplinary curricula are time-consuming and take collaborative teamwork to create, which can seem like a hard and exhausting disadvantage, but in the end, the interdisciplinary approached inhibits many favored skills that are sought by future colleges and employers. Students and their teachers will advance in critical thinking, communication, creativity, pedagogy, and essential academia with the use interdisciplinary techniques". The criticism, mostly concerned the lack of a deeper approach to foundations of constituent disciplines, should be mentioned [61, p. 480]. This is partially explainable by the fact that interdisciplinary teaching methods are focused on constructivist approaches, as for example [61, p. 480], [62].

3.1.4 Creating interdisciplinary environment

The most popular approach to the creation of an interdisciplinary educational environment could be named as "managerial". The general idea is to "correct" the existing "imperfect" (from the interdisciplinary point of view) educational environment through organizational mechanisms. As an example, the comprehensive approach offered by Kidron and Kali [63] could be mentioned. In order to promote interdisciplinary understanding, the authors develop the following model [63, p. 3]: "Boundary Breaking for Interdisciplinary Learning (BBIL) model, which harnesses technology to address the limitations described above regarding compartmentalization, traditional pedagogy, and organizational hierarchies". The model refers to the next perspectives: "From the curricula perspective, the model seeks to address the compartmentalization challenge by technology enhanced features, designed to promote interdisciplinary understanding and focusing on a crosscutting theme to help learners integrate knowledge from several disciplinary lenses; From the pedagogical perspective, the model seeks to address the traditional pedagogy challenge by adopting a learning community [64] in which a technological infrastructure is used for promoting a learning culture that enables participants to synthesize different views, solve problems and collaboratively advance knowledge using the wealth and diversity of ideas that community members contribute; From the organizational perspective, the model seeks to address the organizational hierarchy challenge by breaking the traditional boundaries between graduate and undergraduate students, while using technology enhanced features that implement a cognitive apprenticeship approach [65] to promote productive interactions". As it was mentioned earlier, in spite of presented profound study and comprehensive solutions, such "managerial" approach will not solve the problem in general. There is a need for new pedagogy, which will provide grounds for a self-regulated research environment. The outline for such pedagogical approach will be presented in the next sections.

3.2 Scientific computing

3.2.1 Introduction

The aim of this section is to provide existing definitions of the research and educational areas called Scientific Computing (SC) and existing approaches to Scientific Computing Education (SCE). Based on this, problems and drawbacks will be discovered and revisions and new teaching methods will be proposed in the next sections.

3.2.2 Meaning and definitions

We procced towards the definitions and meaning of SC and SCE. First, it should be discovered what is around or what is the world of SC [66]:

(1) The first part of this world is application. This could be from all possible fields of science like biology, ecology, chemistry, physics, astronomy, and engineerings like civil engineering, mechanical engineering, and aerospace;

(2) The next part of this world is mathematical solutions from the field of classical and applied mathematics. The technology for solving (1) problems is based on a definite type model, which is used for calculations or simulations answering question in study;

(3) The other part is computations and CS;

(4) Finally, the important part is the computer itself. The diagram of the SC is presented in Figure 7 (reprinted from [66, p. 3]).

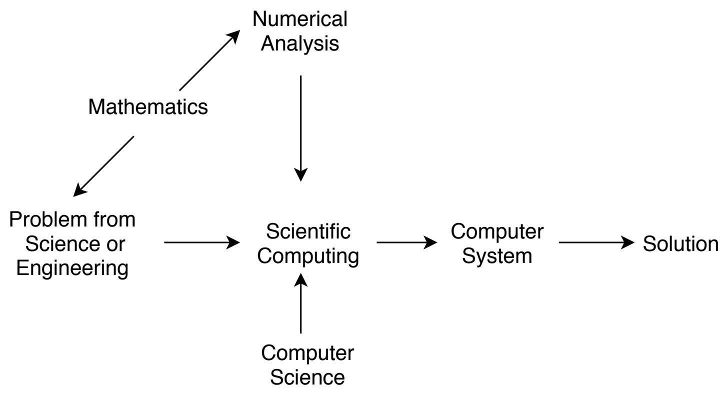


Figure 7: Scientific Computing and related areas. Reprinted from [66, p. 3]

The next important topic that should be taken into account is the common technology of SC [66]. First of all, the key to this technology is a model. Therefore, the process begins with the formulation of the model and continues with calculations, validations, and corrections of the model. A general view on the technological process is presented in Figure 8 (reprinted from [66, p. 6]).

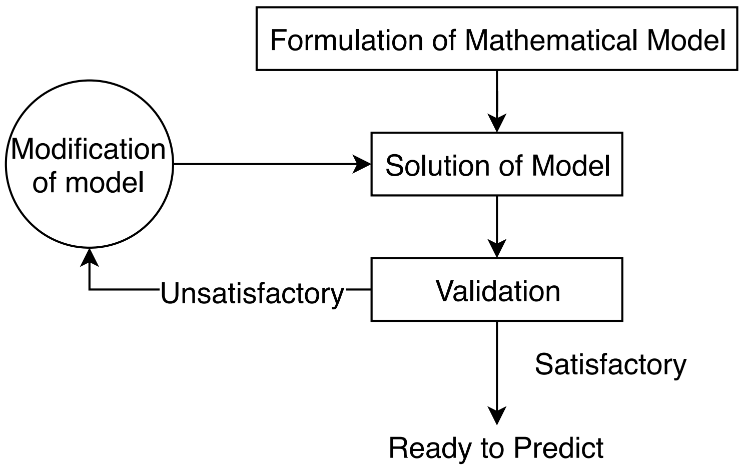


Figure 8: Mathematical modeling and solution process. Reprinted from [66, p. 6]

The third important part is methods or processes within the presented technology. This includes numerical solutions and related parameters like rounding and discretization errors, an efficiency of programming solutions and other topics like reliability, robustness, portability, and maintainability of the code [66]. The fourth player in this team called SC is a set of tools and equipment in the form of hardware and software as in the case of computing. Hardware could vary from a laptop on the table of the researcher to a high-performance computer cluster owned by a research institution or business. Software topics deal with operating systems and languages, data management problems, solutions for visualization and symbolic computations.

Summarizing all of the above the next working definition of SC [66] is accepted: "SC is the collection of tools, techniques, and theories required to solve on a computer mathematical models of problems in science and engineering".

3.2.3 Alternative approaches

Some alternative or similar approaches should be mentioned as well. These approaches arise due to greater emphasis on one or another part or the process, which is mentioned in the provided definition of the SC. An approach focusing on modeling of the specific types stresses the importance of simulations in the SI [67]. As there is no possibility by one or another reason to make physical experiments in situations, which are in the field of interest of scientists or engineers, simulations of the relevant computer model, could help. It is important to stress that an advantage of simulations is not only to provide the possibility of graphical representation of processes of interest in the real-time format but also the possibility to construct a virtual reality and to look beyond the horizons of the present possibilities. Such a simulation-centric approach describes SC as "the heart of simulation science" [67]. Figure 9 (reprinted from [67, p. 4]) presents the definition of SC as the intersection of numerical mathematics, computer science, and modeling.

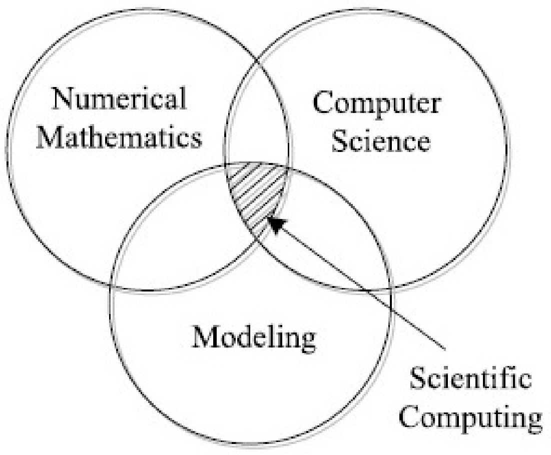


Figure 9: Definition of SC as the intersection of numerical mathematics, CS and modelling. Reprinted from [67, p. 4]

The overall problem solving process using simulations in SC includes [68, p. xv]:

"(1) Development of a mathematical model – often expressed as some type of equation – of a physical phenomenon or system of interest;

(2) Development of an algorithm to solve the equation numerically;

(3) Implementation of the algorithm in computer software;

(4) Numerical simulation of the physical phenomenon using the computer

software;

(5) Representation of the computed results in some comprehensible form such as graphical visualization;

(6) Interpretation and validation of the computed results, which may lead to correction or further renewal of the original mathematical model and repetition of the cycle, if necessary."

Focusing on numerical analysis, SC could be defined as a set of numerical methods and solutions, which are used for numerical computations in various fields of science. This is the historically traditional approach, and numerical methods are very important for the field of SC. These numerical methods include among others: solving systems of linear equations, eigenvalue problems, nonlinear equations, optimization, interpolation, numerical integration and differentiation, partial differential equations, fast Fourier transforms, random numbers, stochastic simulation, numerical approximations of linear and nonlinear differential equations, polynomial approximation, least squares approximations, numerical integration, finite element and spectral method, Swartz method and others [66–69].

Focusing on computations and hardware, SC could be defined as a set of tools and techniques which enables sufficient efficiency of computations [69–71]. The relevant theoretical basis, numerical methods, and algorithms, a computer model for calculations or imitations do not provide any guarantee of the final successful solution. The relevant computational resources should be employed for calculations, which should use a definite amount of the processor time. In the other case, even if the model is properly designed and coded but the computational efficiency of the model is low, the model should be redesigned or optimized, taking into account the relevant hardware resources.

3.3 Existing approaches to scientific computing education

3.3.1 Solving scientific problems

Scientific Computing plays an important role in science and engineering education. The leading world universities and organizations pay an increasing attention to the curriculum and educational methods. One of the tasks in

SC education is to provide a general understanding of solving scientific problems. For example, Allen et al. report on a new graduate course in SC that was taught at Louisiana State University [72]: The course was designed to provide students with a broad and practical introduction to SC which would provide them with the basic skills and experience to very quickly get involved in research projects involving modern cyberinfrastructure and complex real-world scientific problems.

Michael Heath [68] writes, "...try to convey a general understanding of the techniques available for solving problems in each major category, including proper problem formulation and interpretation of results...". Michael Heath offers a wide curriculum to be studied including a system of linear equations, eigenvalue problems, nonlinear equation, optimization, interpolation, numerical integration and differentiation, partial differential equations, fast Fourier transform, random numbers, and stochastic simulation. All these topics require a large number of computations and could require parallelization solutions to be solved.

3.3.2 Seamless approach to theoretical prerequisites

Studying SC is always a challenging task for a learner as well as for an educator. Such a studying process deals with plenty of technical and multidisciplinary issues and requires a synchronization of the learner's mathematical and CS competencies. One of the possible solutions, in order to overcome these difficulties, is to develop a set of learning objects and the relevant methodology. This should be based on the constructionist approach to learning, should provide a relevant framework for an educator and appropriate learning material for students. Such a framework should enable a learner to conduct series of computational experiments with computer models. Using this approach, related mathematical and programming learning material is provided on demand and in parallel to the main curriculum. This is especially true for an introductory SC course as possible application scope of this approach. For example, Karniadakis and Kirby II define [67]: "...a seamless approach to numerical algorithms, modern programming techniques, and parallel computing ... Often times such concepts and tools are taught serially across different courses and different textbooks, and hence the interconnection between them is not immediately apparent. The necessity of integrating concepts and tools usually comes after such courses are concluded, e.g. during a first job or a thesis project, thus forcing the student to synthesize what is perceived to be three independent subfields into one in order to produce a solution. Although this process is undoubtedly valuable, it is time-consuming and in many cases, it may not lead to an effective combination of concepts and tools. Moreover, from the pedagogical point of view, the integrated seamless approach can stimulate the student simultaneously through the eyes of multiple disciplines, thus leading to enhanced understanding of subjects in scientific computing."

3.3.3 Towards a learner-oriented and constructivist technology

It is obvious that, from the point of view on an educator, the presented topic is broad and complex in its nature, and to develop a proper and effective educational technology is a difficult and challenging task. Historically, SC started from applied mathematics, applied numerical computations and moved to computational and finally imitational modeling involving more and more topics to study including describing theories, scientific and engineering applications. The heritage of this is a variety of approaches to SC presented in the literature related to SC education. The main tendency is to move from traditional and less effective teacher-centered pedagogical approach like the one presented in [66, 73, 74] to more learner-oriented and constructivist technology. At the same time, the didactical aspects of the course are usually left as they traditionally were. This is especially true for traditional educational literature like textbooks. At the same time, only a small number of papers focused on SC education, has been already published.

To solve obvious problems with students' motivation, when students need years of pre-studying, educators try to adopt the classical teacher-centered educational methods and try to wrap these methods in a learner-centered wrapper. These techniques include various modernizations for the course syllabus including problem-based and other technologies. For example, a rather theoretical course for random differential equations called "Dynamic Systems and Scientific Computing – Introduction to the Theory and Simulation of Random Differential Equations" is designed in a student-centered manner as is presented in Figure 10 (adapted from [75, p. iii]):

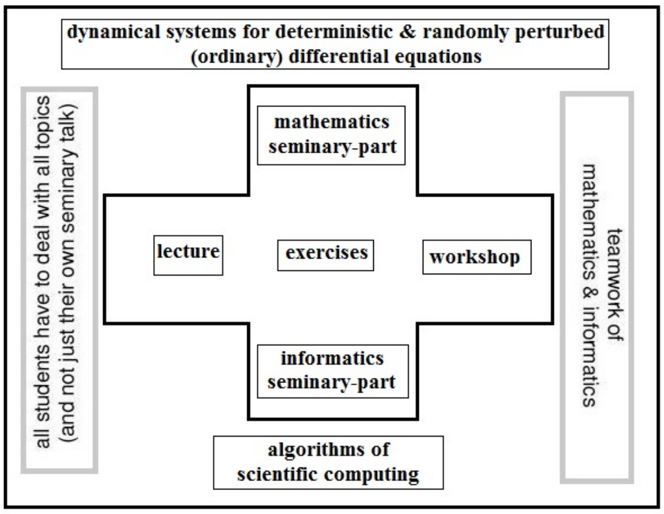

dynamical systems for deterministic & randomly perturbed (ordinary) differential equations
all students have to deal with all topics (and not just their own seminary talk)
mathematics seminary-part
lecture
exercises
workshop
informatics seminary-part
teamwork of mathematics & informatics
algorithms of scientific computing

Figure 10: The outline of the student-centered course. Adapted from [75, p. iii]

3.3.4 Project-based approach

Historically, software packages like packages for symbolic computations such as Matlab, Maple, Maxima or similar, made it possible to include additional student-oriented features into a course for SC. Educators, who initially were focused on numerical analysis, find it attractive to include Matlab-based projects in their course [69, 76]. The general motivation for this is various difficulties students have during learning. For example, Turner [76] reported about too much time students spend mastering the details often missing the point as to why there is a need for SC. Another advantage is to allow student teamwork, as project-based education enables this feature. And the next important feature is that such a computer-based project-oriented education allow users to continue their activities improving the model they already constructed in the previous steps [76]. Summarizing, the project-based approach to SC education helps to improve the course adding the next main features: computer models, possibility to develop earlier developed models, teamwork possibilities, making the course more student-oriented, and adding constructivist features.

Educators, who place emphasis on computations and hardware, are facing similar difficulties as colleagues based on numerical analyses approach to SC . One remark should be made in this context. If the focus is on computations, the efficiency of algorithms and coding solutions is of primary importance. If such topics are within the focus, usually such courses are developed under the names of “Scientific Programming” or “ High-performance computing". The curriculum of a course for scientific programming is mainly concentrated on algorithms efficiency, coding techniques and code efficiency, programming languages, and parallelization, where different programming techniques and parallelization are usually within the main focus [70, 77–84]. The problems with the efficiency arise during practical calculations within a limited number of computational resources involved as it is usually in practice. So even a well-developed model with correct algorithms could fail to provide the result if the computational resource is limited in time of performance. Therefore, two basic aspects are of considerable attention to educators: parallelization and software engineering.

3.3.5 Importance of parallelization methods

Modern technologies widely involve parallel computing, and plenty of scientific and industrial applications use parallel programming techniques. The teaching of parallel computing is one of the most important and challenging topics in SC and programming education. Properly developed constructionist educational methodology, for example as one which is based on a model-centered approach and uses the learning by comparison method, could be considered as an effective solution for this course. Different parallelization techniques could be implemented for programming of a model. Such an approach allows students to carry out a series of experiments with different programming models, compare the results, and investigate the effectiveness of parallelization and different parallelization methods. Such parallelization methods could include shared memory, distributed memory, and hybrid parallelization, which are implemented by MPI and OpenMP Application Programming Interfaces (APIs) [67, 85–88].

Karniadakis and Kirby II write [67]: "With the rapid and simultaneous advances in software and computer technology, especially commodity computing, the so-called supercomputing, every scientist and engineer will have on her desk an advanced simulation kit of tools consisting of a software library and multi-processor computers that will make analysis, product development, and design more optimal and cost-effective". The authors suggest the integration of teaching of MPI tools into the educational process. A large number of MPI implementations are currently available, each of which emphasizes different aspects of high-performance computing or is intended to solve a specific research problem. Other implementations deal with a grid, distributed, or cluster computing, solving more general research problems, but such applications are beyond the scope of this study. M. A. Heroux et al. [89] describe SC as "…a broad discipline focused on using computers as tools for scientific discovery". The authors claim: "The impact of parallel processing on SC varies greatly across disciplines, but it could be strongly argued, that it plays a vital role in most problem domains and has become essential in many".

NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing (PDC), Core Topics for Undergraduates, contain comprehensive research on the curriculum for parallel computing education [90]. The authors suggest including the teaching of PDC: "In addition to enabling undergraduates to understand the fundamentals of 'von Neumann computing', we must now prepare them for the very dynamic world of parallel and distributed computing". G. Zarza [91] et al. report: "High Performance Computing has turned into an important tool for modern societies, becoming the engine of an increasing number of applications and services. Along these years, the use of powerful computers has become widespread throughout many engineering disciplines. As a result, the study of parallel computer architectures is now one of the essential aspects of the academic formation of students in Computational Science, particularly in postgraduate programs". The authors notice significant gaps between theoretical concepts and practical experience: "In particular, postgraduate High Performance Computing (HPC) courses often present significant gaps between theoretical concepts and practical experience". B. Wilkinson et al. [92] offer "…an approach for teaching PDC at the undergraduate level using computational patterns. The goal is to promote higher-level structured design for parallel programming and make parallel programming easier and more scalable". J. Iparraguirre et al. [93] share their experience in a practical course of PDC for Argentina engineering students. One of the suggestions is: "Shared memory practices are easier to understand and should be taught first". The authors also suggest focusing on smartphones and tablets as on target platforms for learning parallelization.

Langtangen presents an approach, which focuses on software engineering [94]. He stresses the importance of software engineering techniques in SC: "Teaching material on SC has traditionally been very focused on mathematics and its applications, while details on how the computer is programmed to solve the problems have received little attention. Many end up writing as simple programs as possible, without being aware of much useful CS technology that would increase the fun, efficiency, and reliability of the their SC activities".

3.3.6 Conclusions

To conclude the material presented in this section, the importance of pedagogical ideas of integration and stemless approach to Scientific Computing Education (SCE) [67] could be stressed. As the discipline that could potentially cover different fields of science, with a variety of theoretical backgrounds and traditions, a unifying approach, implemented within an educational institution like a university, could provide complex foundations aimed, besides field-specific knowledge, at improving SI, computational literacy, and engineering skills. At the same time, such unifying approach could provide a solid background for interdisciplinary research activities, and this is of primary importance for modern science, technology, and education in these fields.

3.4 Educational technologies

3.4.1 Introduction

The term "educational technology" is understood here as the theory and practice of educational approaches to learning. This section presents educational technologies, which focuses on learner-centered approaches to university education in general, and CSE in particular. CS could be considered as a related field to SC (especially taking into account a computer simulations making approach to CSE). First, the description of constructionism and its applications to CSE will be provided. Next, the learner-centered approaches in general and the so-called "mediated" teaching style in particular will be examined. The learner-centered technologies are considered as important for SCE. Such technologies support the so-called "seamless" approach to students' theoretical prerequisites, provide a constructionist learning environment and support SI and simulation-making based education in the field of SC.

3.4.2 Constructivism and constructionist approaches to learning

First, the general approach to constructivism in education is studied. Constructivism is a theory of knowledge that argues that humans generate knowledge and meaning from an interaction between their experiences and their ideas [95]. Von Glasersfeld [96] describes Constructivism as "a theory of knowledge with roots in philosophy, psychology, and cybernetics. It asserts two main principles whose application has far-reaching consequences for the study of cognitive development and learning as well as for the practice of teaching, psychotherapy, and interpersonal management in general". The two principles are:

(1) knowledge is not passively received but actively built up by the cognizing subject;

(2) the function of cognition is adaptive and serves the organization of the experiential world, not the discovery of ontological reality.

These principles are of primary importance for the purpose of our study. To support an active buildup of knowledge, the appropriate learning objects needed to be constructed. Caine and Caine [97] in their fundamental research propose the main principles of constructivist learning. One of the most important for us is as follows: "The brain processes parts and wholes simultaneously". Therefore, a well-organized learning process provides details as well as underlying ideas. Using model-centered learning, the goal of the research is introduced first, after, the learner experiments with the model for simulation should take place. That allows us to observe the results and to draw relevant conclusions. Constructivism advocates student-centered and discovery learning where students use the information they already know to acquire more knowledge [98].

Constructionism [25] and constructionist learning is inspired by the constructivist theory specifying how individual learners construct mental models in order to understand the world around them. Constructionism provides us with a basic idea of an appropriate learning object. Such an object should support systematic understanding of the materials and concepts it represents, allowing the user to self-construct his or her knowledge.

3.4.3 Constructivism and constructionist approaches to computer science education

Ben-Ari [99] has developed a constructivist methodology for CSE. The author stresses an idiosyncratic version of knowledge each student constructs basing on the knowledge the students already have. Here it is very important to show the difference between classical and constructivist educational paradigm. The classical paradigm, among other theoretical statements, describes the student as "clear minds" and tries to fill it (his mind) with particular knowledge. The constructivist paradigm considers any student's previous experience (knowledge) as the main part of an educational process, stressing that the new knowledge could be only built "on the top" of the previous one and only by students themselves. Therefore, active learning must take place [99]: "Passive learning will likely fail, because each student brings a different knowledge framework to the classroom, and will construct new knowledge in a different manner. Learning must be active: the student must construct knowledge assisted by guidance from the teacher and feedback from other students". Therefore, the task of the educator is to develop the learning process in the manner that supports the self-construction of the student's knowledge with emphasis on the student's previous background (mental model) existence or non-existence.

Wulf [100] reviews "the application of constructivist pedagogical approaches to teaching computer programming in high school and undergraduate courses". The author stresses an importance of communicating with students explaining the principles of the constructivist approach to learning. Students, who are not familiar with the constructivist approach, usually complain about the lack of explanations from the teacher. Here it is important to motivate the student by involving him or her to the teaching process by providing an appropriately designed learning object.

Several additional remarks could be made here [99]. First, if the constructivist approach to programming education is accepted, the appropriate "level" of the basic mental model is essential for successful further education, so the model-centered approach could serve us in our didactical constructions. Second, the so-called "bricolage", a term coined by Claude Levi-Strauss and adapted for the needs of teaching programming by Turkle and Papert [101] should be avoided. The bricolage leads to the "endless debugging of the 'try-it-and-see-what-happens' variety" [99], therefore, an appropriately designed learning object must take care of this issue. In such a case, the model-centered approach could serve us as well. The basic model, which lies in the "center" of the learning object, will "protect" the student from the unlikely bricolage side effects.

Hadjerrouit [102] investigates the constructivist approach to practical software engineering lecturing and presents a case study of practical examples of the constructivist approach to teaching object programming and using web-based resources for the course. It is obvious that the constructivist approach should be supported by appropriate and properly designed learning objects, which support the constructivist paradigm.

The constructionist approach makes it possible to overcome difficulties and to raise motivation of the novice programmers. This could be considered as [8, p. 1]: "...an important factor for engineering education in general and for programming of embedded devices, as well as for calculations and modeling in the field of scientific computing in particular." Such an approach enables us to overcome some negative effects like the effect of the "bricolage" [99]. The negative influence of this side effect could be very important and could have a great influence on the teaching results especially within a simulation driven educational activity.

Another important topic is how to develop computational thinking skills for software engineering novice students. Such skills are of primary importance for school and university students [103]. The important problem here is the problem of the evaluation of the level of students' compuational thinking skills and abilities [7]. The solution could be to motivate students' participation in internationally based educational activities, like international challenge on informatics and computational thinking "Bebras" [104]. Such participation, besides the motivational factors, will provide a platform for a social learning thus enabling constructionist learning and even a further step of moving from computational thinking to computational participation [105].

3.4.4 Educational technologies for university education in general

As it could be seen from the previous section, educators pay little attention to the technology of SC education and its connection with the content of the course. In order to provide the pathway for improvements and innovations, this section will observe general tendencies in educational technology and applications in engineering and CSE. Generally, the main tendency in educational technology that could be observed is moving towards a learner or the so-called learner-centered approach. Educators would like to unify the technology, providing different unifying approaches and collecting the best practice in the field. Generally, the technology is based on two main principles:

(1) first, educators look for unifying and the most effective technological approach to pedagogy;
(2) and next, educators try to collect the best practices or case studies of applications of (1) in various fields of teaching and learning.

Several main questions concerning general principles of academic learning are considered to be of primary importance. Should learning be considered as imparting knowledge? Should learning be considered as situated cognition? Considering the arguments for and against, the conclusion could be drawn in favor of situated learning, so learning is more effective if it is situated in the domain of objective. As to academic education, situated learning is criticized for neglecting the role of abstraction, as an abstraction is essential for academic learning. Laurillard [106] proposes to consider teaching process as mediated learning, involving "constructing the environments which afford not only learning of the world but also learning of descriptions of the world. ... Thus teaching is a rhetorical activity: it mediates learning, allowing students to acquire knowledge of someone else's way of experiencing the world" and suggests the following teaching strategy [106, p. 77]:

"Discursive: teacher's and student's conceptions should each be continually accessible to the other; teacher and student must agree to learn goals for the topic; the teacher must provide a discussion environment for the topic goal, within which students can generate and receive feedback on descriptions appropriate to the topic goal;

Adaptive: the teacher has the responsibility to use the relationship between their own and the student's conception to determine the task focus of the continuing dialogue; the student has the responsibility to use the feedback from their work on the task and relate it to their conception;

Interactive: the teacher must provide a task environment within which stu-

dents can act on, generate and receive feedback on actions appropriate to the task goal; the students must act to achieve the task goal; the teacher must provide meaningful intrinsic feedback on their actions that relates to the nature of the task goal;
Reflective: the teacher must support the process in which students link the feedback on their actions to the topic goal for every level of description within the topic structure; the student must reflect on the task goal, their action on it, and the feedback they received, and link this to their description of their conception of the topic goal."

Moving towards learner-centered education inspired approaches, which unify educational methodology, like pedagogical design patterns [107] to strengthen their positions within the scope of university education. Why is this methodology important? The next remark should be made here. First, CS, contrary to physics or mathematics, as a scientific discipline is still on its way to clarify and develop its theoretical foundations [108], therefore sharing educational experience and the best practices is of primary importance for practitioners in this field; next, the accessible best practices could present different innovative ideas and approaches [109–112].

Discursive, adaptive, interactive and reflective ways of teaching, which are described in this section, give a pathway and inspiration for practical applications in engineering, CS, and SC education.

3.5 Model-based approaches in education

3.5.1 Introduction

Model-based approaches are the focus of our study. There are several reasons for such an attitude.

(1) First, models are considered as mental models, and mental models and their simulations provide theoretical foundations of the model-based approach to instructional technology;
(2) Next, computer models are the main elements of the model-based teaching methods. Such teaching methods are based on model-based simulations (cognitive artifacts) students should develop.
(3) Finally, scientific models are at the center of the educational technology that is based on teaching SI. Therefore, this complex nature of models allows us to use a universal approach for construction of an integrated

educational environment using DSR techniques (computer models and simulations are considered as cognitive artifacts).

First, the description of the model-centered approach to instruction is presented. Next, the model-based approach in CS education is covered, and finally, simulations and scientific simulation-based approach in general is discussed.

3.5.2 Model-centered co-instruction and co-mediated constructionism

The general description of models, as related to education, is based on Peirce pragmatism [39] and further neo-pragmatic studies [113, 3, 114] (from [3, p. 44]): "a model is a limited reproduction of reality, characterized by at least three features:

(1) Representation. Models are always "models of something", that is, images, representations of natural or artificial originals, which in turn can be models of something else;

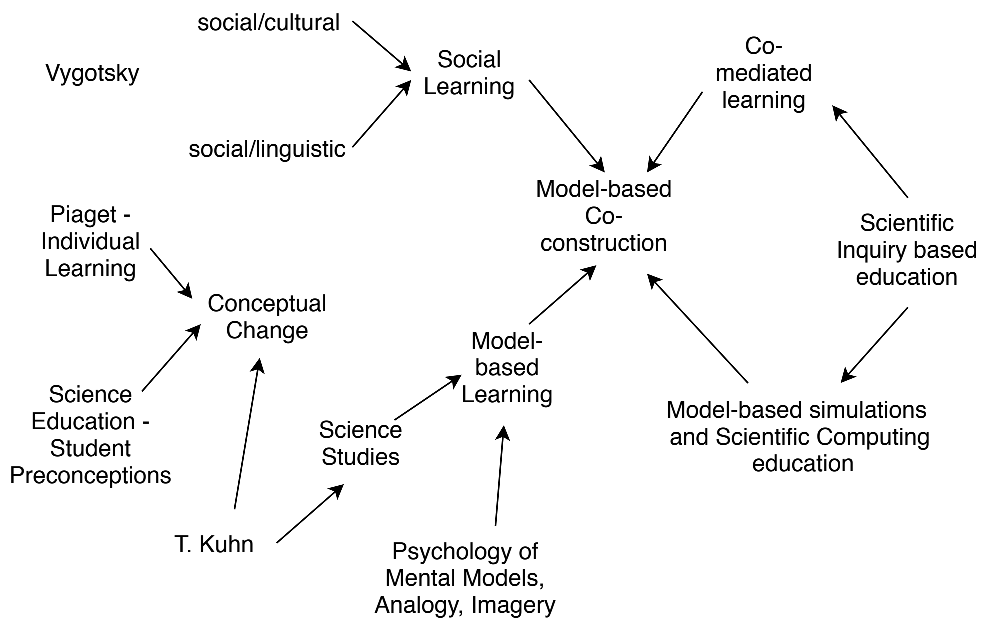
(2) Reduction. Models generally do not capture all the attributes of the original, but only attributes considered to be important by the model's creators and users.

(2) Pragmatism. Models are not copies of their originals. They have a substitution function for
   (a) specific individuals who must understand and/or act using the model, during
   (b) specific time intervals, and
   (c) within the limitations of specific ideal or real operations."

Gibbons introduced a model-centered instruction in 2001 [26]. The following main principles are important:

(1) Learner's experience is obtained by interacting with models;
(2) Learner solves scientific and engineering problems using simulation on models;
(3) Problems are presented in a constructed sequence;
(4) Specific instructional goals are specified;
(5) All necessary information within a solution environment is provided.

There is a sort of contradiction between such educational aims as achieving formal course requirements in the form of tests or exams and focusing on learner-centered educational methods [115]. Model-centered instruction could be a solution for this contradiction. At the same time, an important idea to be considered is a so-called process of "co-construction" of knowledge [115]. Such an attitude also influences educational approach to instruction, therefore, the term "co-mediated" instruction as a cognitive enhancement of a "mediated" teaching style [106] is proposed. This is very important from the model-based and simulation-based teaching perspectives, as an educational process based on model making requires an appropriate educational environment to be constructed and implemented in practice. Such educational environment is based not only on explicitly formulated knowledge, but also on expertise and tacit knowledge, which could be transferred only in the form of co-constructs using the proposed co-mediated learning technology (see Figure 11, adapted from [115, p. 25]).



Vygotsky
social/cultural
social/linguistic
Social Learning
Co-mediated learning
Model-based Co-construction
Piaget - Individual Learning
Conceptual Change
Scientific Inquiry based education
Science Education - Student Preconceptions
Model-based Learning
Science Studies
Model-based simulations and Scientific Computing education
T. Kuhn
Psychology of Mental Models, Analogy, Imagery

Figure 11: Relations between cognitive and technological approaches. Adapted from [115, p. 25]

Another important topic to discuss is the way students construct or improve their mental models. The fundamental approach proposed by Piaget [23] stresses the importance of persons' dissatisfaction with initial mental models. Such dissatisfaction will motivate individuals to improve or replace the existing model with a new one. Therefore, the main educational task is the so-called "conceptual change". This is a challenging task, as students come with their own perceptions, conceptions or misconceptions, which could resist the process of conceptual change of the models. While

Paget focused on constructivism, Vygotsky [116] stresses an importance of the social environment for the construction of knowledge. This leads, especially as related to CSE, to Papert's constructionism [24], and stresses the importance of the construction of an appropriate learning environment that enables teamwork and implements problem-solving educational methods.

The next point to study is the theory of mental modeling. The mental modeling theory could fill in some gaps of the described constructivist and social-constructivist approaches [115]. The most important fact here is that mental models, as opposed to classical deductive reasoning, are constructed by the process of "informal reasoning" [117, 118]. How are such models constructed? Johnson-Laird [27] proposes an explanation of reasoning by simulation and improvement of existing models. Such type of "simulation" reasoning states in some kind of opposition to classical deductive reasoning and positivistic approach in general [115]. Gentner and Clement [119] stress the importance of decompositions during the process of reasoning. Such decomposition could simplify the implication of so-called "target" model – the aim of the reasoning process. All these presented theories led to a model-based instructional theory. The theory focuses on the process of construction or modification of mental models. However, there is still a gap between theoretical constructions and practical applications of the theory [115, 120]. This is especially true if real educational environments with a close interconnection between learners and educators are to be encountered. To overcome these difficulties, as it was already mentioned, Clement proposed the so-called "model-based-co-construction" [115] which aims to integrate social and cognitive elements. The key idea is to incorporate the so-called "learning pathway" into the educational process (see Figure 12, reprinted from [115, p. 34]). This allows students to go through the process of constructing of models.

The concept of the evolution of models named "generation, evaluation, and modification cycles" or, in other words, the outline of the process of the improvement of the models generated during the learning process is proposed by Clement (see Figure 13, reprinted from [115, p. 35]).

3.5.3 Model-based education

Millard et al. [121] propose model facilitated learning using "interactive simulations". The authors present a modern computer technology powered by

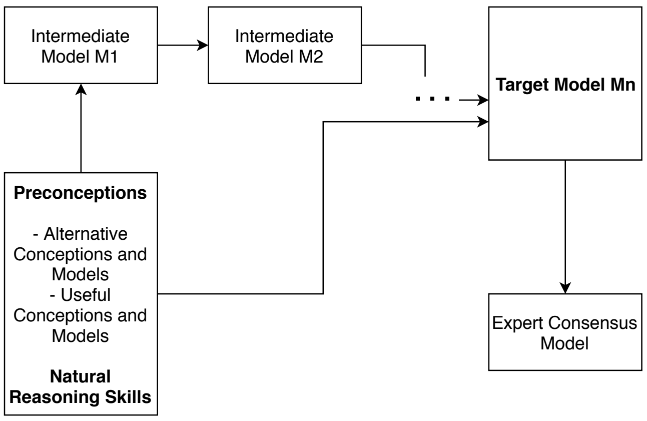


Figure 12: Learning pathway. Reprinted from [115, p. 34]

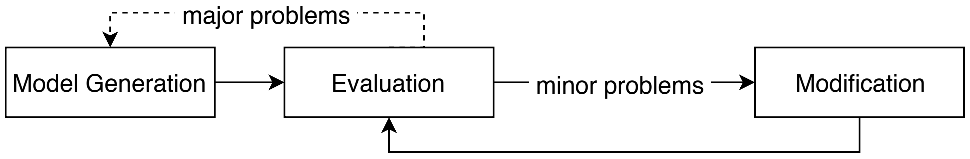


Figure 13: GEM (generation, evaluation, modification) cycle. Reprinted from [115, p. 35]

"promising methodology" based on "system dynamics". "Supportable experiences include the construction of interactive ... models as well as their use for hypothesis testing and experimentation". Lehrer and Schauble [122] refer to the experiments with different representations of the model: "Student learning is enhanced when students have multiple opportunities to invent and revise models and then to compare the explanatory adequacy of different models". L. Xue et al. [123] introduce "teaching reform ideas in the scientific computing education by means of modeling and simulation". The authors suggest "... the use of the modeling and simulation to deal with the actual problem of programming, simulating, data analyzing ... ". Model-centered learning is used in mathematics education. Plenty of models are constructed using "Geogebra" software [124]. Models play the central role in Science Education [115, 125]. The model-centered approach for simulations and learning is presented in [3]. For the purpose of this study, one historical view on models is very inspiring [126]: the models are presented in the form of relations between different states of the dynamic system S_i and its model M_i (see Figure 14 (reprinted from [3, p. 43]), where μ is a correspondence rule to translate the state to its model, μ^{-1} – the inverse rule, C and $C^{'}$ – the relevant maps.

$$\begin{tikzcd}
S_1 \arrow[r, "C"] \arrow[d, "\mu"'] & S_2 \\
M_1 \arrow[r, "C'"] & M_2 \arrow[u]
\end{tikzcd}$$

Figure 14: Relation between the dynamic system and its model. Reprinted from [3, p. 43]

Therefore, the model to be useful, the next composition should take place:

$$\mu^{-1} \cdot C^{'} \cdot \mu = C \tag{1}$$

The next interesting point is a general approach to education with models, which serve as a basis for imitational experiments. Figure 15 (adapted from [3, p. 53]) presents a general view on the process.

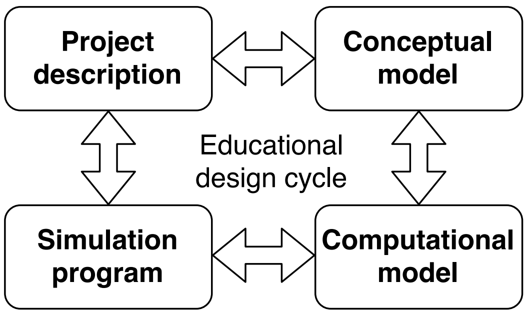


Figure 15: Main outputs of a modeling and simulation process. Adapted from [3, p. 53]

The process includes the next steps: the project description, conceptual model, computational model, simulation program. This concept provides us the pathway for developing of learning objects for a model-based education. First, the relevant conceptual model should be developed. This model should be a universal one and include different specification possibilities. Based on this conceptual model, a set of computational models should be developed and provided. Each of such computational models could be implemented for one or another computer architecture. As the last step, simulation programs could be used for the further improvement of the learner knowledge. Model-based simulations are widely used in engineering practice as well [127].

3.6 Simulations-centered approach

3.6.1 Scientific models and model-based scientific simulations

This century is going to be the "age of computer simulations" [128, p. 9]. Simulations play an extremely important role in all fields of engineering and science [3, p. 1]: "Science is, therefore, the realm that pushes the technological limits of simulation to their extremes. In fact, simulations of galaxy formations, molecular dynamics, protein unfolding, ocean currents, and aerodynamic design require the use of advanced numerical algorithms and parallel computers located in powerful data processing centers. Furthermore, simulation is not only transforming scientific practice, but it is also leading scientists and philosophers of science to re-examine relations between models, theories, and experiments". There is a strong belief of wider use of simulation as an educational method. The word "simulation" implies many different meanings and types of simulations, including so-called "experimental" simulations like a flight simulator. Within the scope of this research word "simulation" means the following types of simulations [3, p. 4]: model-based simulations, based on the construction of the theoretical model of a system (also known as "theoretical simulations").

Simulations and games overlap to some degree. This is even clearly expressed for serious games, which become more and more popular in education [129, 130]. Both model-based educational simulations and simulation-based serious games include such features as an explicitly formulated goal of simulation, rules of simulation and use a clearly stated educational task as a basis for the design. Figure 16 (adapted from [3, p. 6]) presents the relation between model-based educational simulations and serious games.

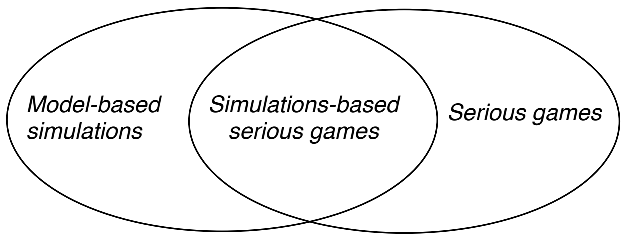


Figure 16: Serious games and simulations. Adapted from [3, p. 6]

How could scientific models be defined? Jadrich [131, p. 12] defines and describes the meaning of a scientific model as follows:

- “Scientific models are human creations meant to represent entities and phenomena in the physical world;
- Scientific models are not singular. Multiple models can exist to describe a single entity or phenomenon, and no single model ever fully and exactly represents reality;
- Scientific models must have a measure of consistency with existing data or evidence. They must be used for both explaining current observations and predict future ones;
- Scientific models are judged on both how simple they are and how well they can be used to explain and predict natural phenomena.”

The presented definition is important for the purpose of our study due to the following. First, scientific models are artifacts. It could be stated that scientific models are cognitive artifacts of a special type that serves as intermediates between the researcher’s cognitive activities and the subject of study. Next, these relations are universal for all fields of science; therefore, the presented definitions of models could be valid for any scientific subject of study. This universality could ground the possible solution for the unifying approach to the interdisciplinary education. The universal language or at least the universal alphabet enhancing SI in all fields of science needed to be developed. s

It is necessary to distinguish between static and dynamic models. Traditionally, the only dynamic model is associated with representing changes in time, while static model serves for representing relations and the time variable is not presented. For example, the model of a fully stochastic system is based on laws of probability. Obviously, such model does not include time variables in its descriptions. However, one could not describe such a model as static. Generally, the distinction between the static and dynamic models is not so obvious. If a person who interacts with a simulation is included, even “static” models could be mentally animated [3, p. 37].

3.6.2 Simulation: The definition

The definition for a model-based simulation is based on the following meanings [3].

(1) The first important part is the meaning of a system. A system is a collection of different elements whose combination yields results that are unobtainable by the elements alone. Therefore, the system is more

than a sum of its parts;

(2) The next important definition is the definition of a model. A model is a simplified representation of a real or imagined system;

(3) Finally, a simulation based on (1) and (2) could be defined: a simulation is an interactive representation of the system to be studied based on a model of the system.

This definition has a broader meaning than a traditional view on simulations as on a dynamic set of interactive representations. Fundamentally, model-based simulations are related to students' cognitive activities and to the process of constructing appropriate mental models. Landriscina [3, p. 6] provides the following explanation: "These definitions allow us to imagine a series of epistemic transitions, from a reality or an idea to a system, from the system to a model, and from the model to a simulation. Although these entities are conceptual in nature, during the construction process of a simulation, they become cognitive artifacts, such as physical models, data files, written descriptions, visual representations, mathematical formulas, formal specifications, and computer programs. Moreover, the above definition emphasis on the interactive nature of simulation distinguishes it from other forms of knowledge representation and focuses on its potential for creating a relation of interpenetration and synergy between a human mind and a computer."

This approach to simulations also enhances learning related cognitive processes, facilitates modification, construction or replacement of the relevant cognitive structures [132, 133].

These processes [3, p. 7] include enhancement of "...cognitive processes that are crucial to learning, such as:

- selecting key information;
- organizing this information into a cognitive structure;
- integrating this new information into previous knowledge;
- accessing and creating appropriate analogies and metaphors;
- generating inferences;
- reorganizing cognitive structures."

An alternative cybernetics approach is based on engineering traditions. It has its roots in the technique to present any system in the form of a "black box" with specified behavior. Generally, the purposes of a simulation are to observe the dynamic behavior of a model of the real system; thus a person, especially with an engineering background, could consider simulation as something surplus, sophisticated and not essential, especially for educational needs. A model could be considered as more important than its simulation and one could limit the educational task only to modeling. At the same time, it is usual to equal simulations with Information and Communication Technologies (ITC) tools, thus, the effectiveness of such the "traditionally understood" and based on simulations educational process could be doubtful. Within the scope of this study, the focus on simulation making and not on simulation using is made, although the role of simulation as an ITC tool is not rejected. Focusing on simulation making activities and simulation-using activities could become an alternative in some cases depending on the practical teaching environment.

Implementing the engineering point of view, modeling and simulation could be defined in a more general way as follows [134]: modeling as the relation between real systems and models, and simulation as the relation between models and computers. The heart of simulation is a model. From the engineering or pragmatic positions, a model could be defined as [134, 135]: "A model is a description of some system intended to predict what happens if certain actions are taken". Some concepts of modeling could be defined [134, 136]. First, a set of model components should be specified. Each component is described by the set of input, output and state variables. Then the experimental frame is defined as a set of all descriptive variables. There could be different experimental frames defined, depending on the chosen simplification level [136, 137]. The relations between models, experimental frames and simulation are presented in Figure 17 (reprinted from [134, p. 3]).

3.6.3 Simulative scientific reasoning

Nersessian [28, p. 128] proposes a model-based scientific reasoning based on simulations of mental models: "in certain problem-solving tasks, people reason by constructing an internal iconic model of the situations, events, and processes that in dynamic cases can be manipulated through simulation. Such a mental model is an organized unit of knowledge that embodies representations of spatiotemporal relations, representations of situations, entities, and processes, as well as representations of other pertinent information, such as causal structure. The reasoning is carried out by means

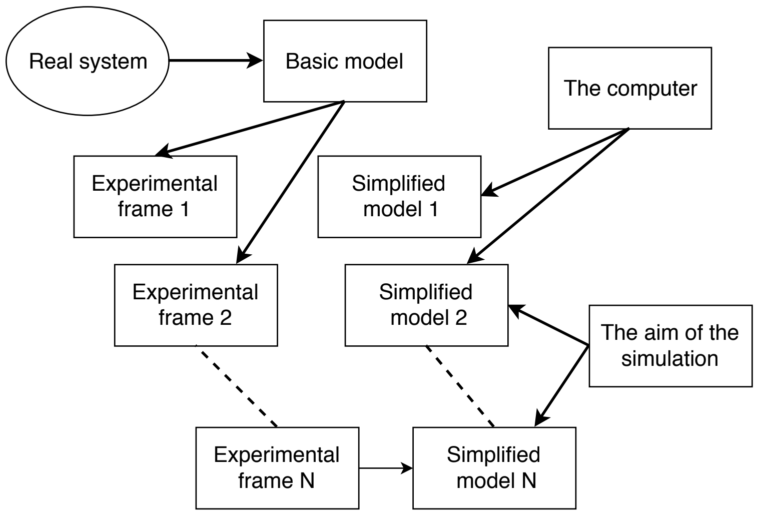


Figure 17: Relation between models, experimental frames and simulation. Reprinted from [134, p. 3]

of model construction and manipulation. In the processes of constructing, manipulating, and revising mental models, information in various formats, including linguistic, formulaic, visual, auditory, and kinesthetic, can be used to construct and animate the model."

Generally, the mental model corresponds to the conceptual model, which is constructed at the first stage implementing model-based simulation activities. The properly designed teaching process should implement a kind of mapping between model-based cognitive simulations and model-based computer simulations designed during an educational activity. This mapping could be provided by the teacher also in the form of co-mediated learning. The possible relation between computer simulation and model-based cognitive simulation is presented in Figure 18 (adapted from [2, p. 148]).

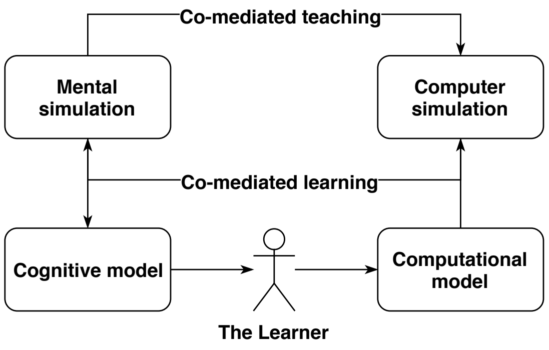


Figure 18: Relation between mental and computer-based simulations. Adapted from [2, p. 148]

3.7 Formalization: Learning objects for scientific computing and computer science education

3.7.1 Introduction

In this section, an observation of Learning Objects (LO) in general and software-based LO in particular is done. Why is it important for us? The concept of LO, as opposite to Pedagogical Patterns (PP), focus on teaching techniques and didactic activities, while PP are more related to educational technology and instruction. As the main result of this study is a teaching method based on an artifact – simulation (that is made by students) in the form of computer software, LO could serve as a wrapper for this or as a building block for an instructional unit. Generally, LO are associated with e-learning content, but for the purpose of our study LO are considered as more universal and covering simulation-based activities as well. At the same time, LO could provide universality for teachers as, depending on the specification of particular LO, teachers could also use them for the simulation-using type of activities as well.

3.7.2 Classification of learning objects

The definition of the learning object needs to be clarified, as, in spite of numerous literature on the topic, various interpretations of the concept still exist. D. Churchill [138] provides "a classification that potentially brings together various perspectives of what a learning object may be. Six unique types of learning objects are proposed and discussed: presentation, practice, simulation, conceptual models, information and contextual representation objects". The author proposes the next definition of the learning object [138]: "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. The first one is simulation learning objects [138]: "They allow a learner to explore, usually by trial and error, operational aspects of a system, carry on a task that the system supports, and develop a mental model of that system's functionalities. Although fidelity is often high in simulations, development of skills is hardly ever completed and learners must usually move to a real system to complete their practice to genuine competency level. 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. This is particularly effective when learning to use the real system requires an understanding beyond being able to operate it (e.g., understanding how a system works) and when the real system is expensive, unavailable or available in limited number, or learning to operate it is costly and possibly dangerous".

The next is a conceptual model [138]: "...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. An appropriate combination of these two models is considered as an appropriate solution for the construction of advanced learning objects, which represent SC and programming concepts and allow simulations and practical experiments with such conceptual models.

3.7.3 Learning objects for computer science education

Nugent et al. [139] describe an approach to design, develop, and validate learning objects for the Computer Science One (CS1) course. The authors focus on learning object for teaching classes and objects: "Each LO is self-contained and by design, the length of the content section is kept short to retain student interest. ...Each learning object covers a core CS topic addressed by four components:

(1) A brief tutorial or explanation including definitions, rules, and principles;
(2) A set of real-world examples illustrates key concepts and includes working examples and problems, models, and sample code;
(3) A set of practice exercises provides important active experiences to the student, with constructive feedback to student responses;
(4) A set of problems graded by the computer provides a final assessment."

As it could be seen from the description, the authors use models and sample code as a part of the learning object. In the model-centered approach the model would become the "center" of the learning object and the presented sample code provide a platform for the constructivist approach to the arrangement of the learning process.

Miller et al. [140] evaluate the use of leaning objects in CS1 education. The paper "...provide a high-level overview of our LO deployment". The authors' findings include:

- "students using LOs have significantly higher assessment scores than the control group;
- several student attributes are significant predictors of learning;
- active learning has a significant effect on student assessment scores;
- and feedback does not have a significant effect, but there are variables with significant moderating effects".

Here the authors stress the importance of the active learning and the relevantly designed educational process. Therefore, the properly constructed learning objects support the constructivist paradigm in education with better educational results. The modern CS education faces many challenges while trying to find the best and optimal way of providing the relevant teaching content. Educators discuss the benefits of one or another programming language to be the first one of study programming, the benefits, and drawbacks of various didactic approaches. The properly constructed learning object could be a solution in this case. Matthiasdotir [141] states that "learning objects with their visualization may be considered a feasible support". The author reports about the sufficiently increased motivation of novice programmers in studying programming, thus concentrating the efforts of the teaching staff in developing and deploying learning objects for programming education could cause a sufficient improvement of an educational process.

3.7.4 Software program as a learning object

For the purpose of our study, the concept of the software program as a learning object is one of the most important. Vytautas Štuikys describes the meaning in his fundamental research [33, p. 13] (CS – Computer Science, LO – Learning Objects): "...the CS content and its delivery as CS s are specific with respect to many attributes as follows:

- The large body of teaching content in CS is programs (algorithms) or their parts such as data structures.
- Program as an LO is abstract. The essence of the topic to be learned is

hidden and the cognition process requires a good understanding of other topics such as the computer architecture, LO and Internet. Therefore, students, especially novices, have difficulties in comprehending the essence to be taught.
- In contrast to the other type of LOs, a program is an executable specification with the well-formed internal structure. The program can produce not only data because of calculation but also the other program as a new LO.
- The program is a soft thing. There are practically unlimited opportunities for its change, modification, and adaptation or even for visualization of the algorithm behavior. Transferring to the different e-learning environments is easy.
- For the learning purposes, programs can be incorporated into other things (such as educational toys, robots, etc.) to enable them to perform the real-life processes (such as the physical items moving, carrying or finding by a robot, etc.).
- Teaching in CS (e.g. programming) can be seen as a problem solving (as it takes place, e.g. in mathematics) to enable the creation of a flexible means to the testing and self-testing of the acquired knowledge.
- LOs to teaching in CS 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."

Therefore, the roles of the programs are twofold. The programs are naturally presented in the scope of an EC. On the other hand, the programs are the learning objects by themselves. Therefore, one of the questions to study is: what is the difference between these two instances of the software program and how practically and efficiently to 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 constructivist and constructionist paradigms.

3.7.5 E-learning and the concept of software learning objects

For the purpose of our study, e-learning is understood as a universal paradigm of a modern learning using technological tools. The process-based view on e-learning is presented in Figure 19 (adapted from: [33, p. 7]).

The conception of LO has been developed by Štuikys introducing gener-

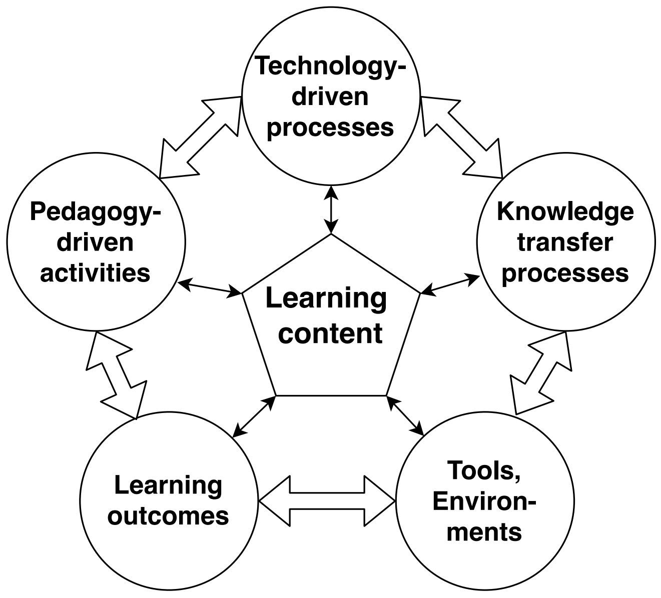

Technology-driven processes
Knowledge transfer processes
Pedagogy-driven activities
Learning content
Learning outcomes
Tools, Environ-ments

Figure 19: E-learning model. Adapted from: [33, p. 7]

ative LO and SLO [33]. Under the presented framework a learning content should be developed taking into account the surrounding socio-technical environment, thus in a universal form which allows such features of the learning content like flexibility, adaptability, reusability, and interoperability [33, p. 20]. The learning content is naturally based on LOs. Under the presented framework, LOs are considered as a part of the learning content. Therefore, the design of the SLOs should be aimed at the specification, which enables smart incorporation of the components into the learning content environment. Institute of Electrical and Electronics Engineers (IEEE) provides the most general definition of the LO stating that LO is any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning [33, p. 9]. Below, some theoretical aspects of SLOs are presented. This will serve us as a grounding theory for the development of simulation smart learning objects.

The following features of SLOs are of primary importance for the purpose of our study [33, p. 20]:

- “Learning context is implemented using a priority-based model;
- Priority based model enables learning variability, which is explicitly implemented via generative aspects of the learning content;
- Generative aspects enable refactoring, which is implemented via multi-stage models;

• Multi-teaching environments are also supported.”

3.7.6 Concepts of reusability and repurposing

One of the most important features of any LO is reusability. There are two general aspects of reusability [33, p. 15]: managerial (social) and technological. Within the scope of the research, the most technological aspects will be covered. Štuikys [33, p. 15] describes two technological approaches: component-based reuse and generative reuse. The next remark should be made. If the component based approach does not explicitly require automation process to be presented enabling reuse features, on the contrary, generative reuse implies automatic generation of instances. Consequently, generative LO should be designed with the process of some kind of automatic refactoring in mind. Therefore, learning context should allow the implementation of such procedures via learning variability issues and a possibility to distinguish generative aspects of the LO. The reasonable question arises within this aspect... is it always possible? Are there any requirements for the learning context to be generative friendly? Another topic possible for discussion is the topic of repurposing of LOs. If the problem of reusing mainly concentrates on LO features and their adaptation within the same learning context, repurposing [142], means an adaptation of LO to a different learning context or to a subcontext. This is another strategy. For example, if the idea of recurrent algorithms is considered to be introduced? How to construct an appropriate LO, enabling context repurposing? Why is this needed? Examples will be presented in the following sections, but the main idea is that using different contexts the understanding could be achieved better and at the same time this could enable us to use the constructionist approach to learning.

To resume, there are two general ideas: reuse and repurposing. Štuikys proposes “smartification” of the Generative Learning Objects (GLO) aimed to reuse (via meta-featuring) and the idea of repurposing of LO is left behind the scene. For the purpose of our study, the problem repurposing is of primary importance. For repurposing of LO, not only generative features for LO parameters generation should be provided, but also the generative features for generation of a LO structure. Several points should be mentioned in this regard. First, is it generally possible to design such a meta-structured LO? According to [142, p. 3], a properly designed LO should be designed as optimum cohesive and decoupled; therefore a meta-structured LO could possibly become too complex and unpractical. Next, how to consider a possible variation of learning contexts? Such variation could also imply changes in requirements, thus, again, such meta-structured LO will become too complex and possibly unpractical. The idea of solving the described problem is following: it should be moved from a priority-based context model to [33, p. 20] to a multi-context model as a higher level of abstraction. Such a multi-context model will include priority-based models as sub-models.

3.8 Modelling concepts and formalization rules

3.8.1 Modeling concepts of the computer science learning domain

Two main approaches could be mentioned: the model-driven engineering approach and product line engineering approach, which are based on object-oriented modeling and feature-based modeling accordingly [33, p. 77]. Meta-model represents domain concepts, while platform-specific meta-models are created using transformations of meta-models. Feature-based modeling could be practically implemented using feature diagrams. Feature diagrams could specify possible features of the generative meta analysis of domain features proceeding with further implementations of such LO using meta-programming techniques [33, p. 78]. The proposed feature-modeling method is based on the next main principles and specific requirements [33, p. 81]:

(1) “requirements for specification of domain models (scope, boundaries, transformation);
(2) general requirements for verification of models and for a technology of practical model;
(3) feature modeling should be started with the planning and formulating the objectives and role of the model;
(4) various requirements for the collection of models like priorities, hierarchies, intersections and others;
(5) internal structure of specific models could be clarified, for example, such sub-models as context (even implicitly implemented) and based models.”

Štuikys proposes two basic methods to construct the model: Feature-oriented Domain Analysis (FODA) and Scope-Commonality-Variability (SCV) methods. There are three basic types of features proposed: mandatory, optional and alternative. FODA principles apply to [33, p. 81]: " (1) domain boundaries and context identification; (2) modelling of the context by feature; (3) modelling of subdomains within the boundaries of features."

3.8.2 Formalization rules for feature modelling

The propositional logic for expression of feature logic is presented below. Let P be the parent feature and the sets $\{C_1, \ldots, C_n\}$ are children features of P. Then the feature relationships could be specified as follows [33, p. 84]:

$$\begin{aligned}(P \Leftrightarrow \vee_{1\leq i\leq n} C_i) \wedge_{i<j} (\neg C_i \vee \neg C_j)(XOR\ relationship)\\ (P \Leftrightarrow \vee_{1\leq i\leq n} C_i)(OR\ relationship)\\ \neg K \vee \neg F(constraint < mutex >)\\ \neg K \vee F(constraint < Require >)\end{aligned}$$

The formalization rules give us a way to develop generative user interfaces and GLOs. At present, the theory of GLOs is only at the starting point. Generative functions are often included using selective interfaces and use the corresponding "frozen" structure of the provided solutions.

Since we focus on simulation making activities, in our case the complexity of the internal structure of the provided learning objects is great, therefore the described modeling method can not fulfill the generating tasks in full. This approach requires further study and development for application to focused on simulation making learning processes, and it is positioned as a topic for further research.

3.9 Scientific inquiry and its place in engineering education

3.9.1 Introduction

An approach that is based on teaching SI is considered as important for the purpose of this study. This can become a kind of universal basis for the integration and unification of the university STEM curriculum. Such integration could be based on SCE. Scientific Computing Education can be viewed in a universal way and opposite to the approach to SC as a completely technical discipline with emphasis on computations. This universal method, in turn, is based on the approach to SI as on research activities based on the development of simulations [131].

3.9.2 Teaching scientific inquiry

First, the question should be answered: what is SI and why it is important for all levels of education. There is a strong opinion [131, 143, 144] that students should not only learn about science and scientific methods but should be able "...do science. This vision for science teaching stems directly from the educational imperative to develop scientifically literate students" [131, p. 3]. As it is clear, students cannot become scientifically literate if they are not involved in practical scientific activities. Moreover, this is definitely true for students of all levels of educational programs at school as well as universities. Implementing scientific inquiry based educational technology is a comprehensive and challenging task [131, 145]. Learning scientific content corresponds to the highest levels of Bloom's taxonomy. "Consequently, learning to think and act like a scientist is much more difficult to do than just learning about scientific content" [131]. How could SI be defined? First, what constitutes SI? Several approaches could be presented:

(1) SI begins with a scientific question;
(2) hands-on activity;
(3) a set of specific methods and practices used by scientists;
(4) a set of reasoning strategies or skills needed while driving a scientific process.

The main common feature of the presented approaches to the definition of the SI is that all these definitions are process-oriented as they attempt to define SI by describing the activities of scientists. Another solution is to define SI using the result-oriented approach [131, p. 9]

What is the primary goal of scientific activity? We share the opinion [131, p. 10], [28, 146] that the primary goal of science are scientific models and the aim of any scientific work is to develop, test, and modify the scientific model of the subject of study is shared. Generally, we could name the process of development, testing, evaluating and modification of a model as simulation. The reason for this is the following. In any case, such operations with the model should take place within the time so this could be described as simulative modeling process or simply simulation. Why do we focus on this? The reason will be clear after looking closer into the nature of the simulation. The scientific activity of designing (developing, testing, evaluating and modification) simulations as artifacts are closely connected to the cognitive activity of mental simulations and simulative reasoning [28]. Summarizing, SI could be defined as an activity of designing of scientific simulations and the aim of scientific work is to design the model-based scientific simulations.

3.10 Design Science Research

Design Science Research plays a very important role within this study.

(1) First, the DSR methodology is used as the research methodology for this research. The outline of DSR as the research methodology will be provided in further sections.

(2) Next, DSR could be considered as a part of educational technology (as one of the possible methodologies for instructional design). This will be described in section named "Example of applications of Design Science Research for education technology".

(3) And finally, and this is the main focus of this study, DSR could be considered as a part of a teaching technique for creation of artifacts – imitation models within the scope of students' activities of making model-based simulations for SI centered SC education.

The latter application is a kind of a truncated form of the "standard" DSR methodology, which is used in a situated educational environment under co-mediation of a teacher. Therefore, students here are considered as a kind of "quasi" researchers. They act within provided an artificial educational environment, use pre-provided slice (as result of seamless approach) of theoretical concepts, improve a faceted and predesigned model, make conclusions and propose (guided by a teacher) further improvements of their earlier created artifacts (in form of model-based simulations). Certainly, if there are such ambitions, an educator could design the teaching process in a manner that will try to reduce the difference between this adapted DSR methodology and the "real" one. However, this is a rather challenging task, it could be considered for advanced courses and more prepared students.

3.10.1 Design Science Research, its foundations and connections to the fields of science

Science, in general, could be divided into formal science like logic or classical mathematics and factual science, which describes, explains and predicts phenomena and is validated when provides empirical evidence. Factual science is divided into natural and social sciences. Natural science is interested in objects or phenomena and the main research activities are to analyze the nature of these and the reasons for them being so [20]. Social science describes and reflects the society and individuals. Research conducted in social science is usually question based and it is focused on the researchers' view on the problem in the study, so it is subjective in its nature [20, 147]. Social science could focus on descriptions with attention to a quantitative approach. Another focus, for example in management science, is on solutions to given problems or on artifacts creation [20].

The concept of Design Science as Science of Artificial was first introduced by Simon [35]. Table 5 presents the main characteristics of different type of science (from [20, p. 13]). As it could be seen from the presented synthesis, design science is focusing on practical solutions and artifacts. As it was described in the previous sections, the mediating role of a teacher could determine the view to the teaching process as to a managing activity [148–150]. Under this approach the DSR could be considered as a promising technique for managing of the teaching process [151].

The motivating reason for any research could vary from research oriented to solving theoretical problems and with no or minor concern to practical applications or applied research focused on practical solutions [152]. Generally, design means creation (or invention) of some new artifacts and its implementation into the area of application. This could be done under existing or non-existing (innovative design) theoretical backgrounds [11, p. 10]. If Design Research (DR) focuses on the question of how to design artifacts, DSR focuses on the problem of using design as a research method [11, p. 13]. Therefore, DSR could be positioned as a well-formalized teaching technique that implements learning through building of an educational paradigm. This is important for the purpose of this study and it will be discussed in more detail in the next subsection.

Table 5: Synthesis – natural sciences, social sciences, and design science. Reprinted from [20, p. 13]

Characteristic	**Natural Sciences**	**Social sciences**	**Design sciences**
Purpose	To understand complex phenomena. To discover how things are and to justify why they are this way	To describe, understand, and reflect on human beings and their actions	To design; to produce systems that do not yet exists; to modify existing situations to achieve better results. Focus on solutions.
Research goal	To explore, describe, explain, and predict	To explore, describe, explain, and predict	To prescribe. Research is oriented towards solving problems
Examples of areas tht usually employ each of these scientific paradigms	Physics, chemistry, biology	Anthropology, economics, politics, sociology, history	Medicine, engineering, management

3.10.2 Design Science Research and applications for information systems and computer science research

Design Science Research offers a practical methodology for the creation of innovative artifacts. This is important for the purposes of our study as the proper methodology provide a way for innovations in the technology of education. The key meanings in this are innovations and artifacts. As a methodology, DSR is of primary importance for CS, ITC and related education in the field. As it was mentioned earlier in this study, there is a strong demand for solid theoretical foundations for ITC and CS as it is related to university education. At the same time, ITC as well as CS are mostly practical activities dealing with various practical engineering solutions, thus DSR gives grounds for innovations and provide solutions for a technological breakthrough in these fields. Another advantage is that DSR provides a unifying approach for innovations and this is extremely important in such strongly interdisciplinary field like SC education, which as a discipline strongly overlaps with ITC and CS. Resuming the above-mentioned statements, SC education provides an interdisciplinary environment for innovations and DSR provides a methodology for these innovations to take place.

Another important feature of DSR should be stressed as well. The main focus of the DSR methodology is "to teach research" [11]. As it was discussed in the previous sections, the priority of SC education is to teach SI, to teach how to solve scientific problems or to teach research as well. Therefore, the educational task and an appropriate method strongly correlate. Below a brief description of the DSR as related to ITC and CS and research methods in these fields is provided.

Why is there a need for the methodology for the research in ITC and CS? For example, the traditional "pragmatic method" could serve as a solution, that is an important research questions could be formulated, inquiring the community for appropriate research methods, investigate the prior research in the field, check with our colleagues for the relevant knowledge, and look for acceptable information [11]. But such a traditional method will not properly work for ITC and CS. First of all the reason for this is in the interdisciplinary roots or "multi-paradigmatic" nature of ITC and CS.

Vaishnavi and Kuechler write ([11, p. 2]): "We believe researchers in ITC fields need a thorough grounding in each of the variety of research philosophies and techniques practiced in their field, and it simply is not practical for any student to undertake a multi-year apprenticeship in each of the major ITC research paradigms. Moreover, DSR as practiced in ITC fields is significantly different from the design-based research practiced in other fields (such as architecture or industrial design); the need for and manner of validation of research results, for example, is more emphasized in Information System (IS), Human-Computer Interface (HCI), and many branches of software engineering due to the grounding of those fields in management science, psychology, and other statistically based descriptive disciplines".

Therefore summarising, IS and CS are examples of "multi-paradigmatic" environments. To go further, an artifact is the main point of our interest providing us a target for a design process. Therefore, the design could be described as related to the artifact and its inner structure and an outer environment "crafting" process [11]. The next question is needed to be answered: can design be research? The answer is affirmative as both IS and CS are focusing on artifacts [153] as resulting entities of the research activities in the fields. A model for DSR process focuses on the contribution of new knowledge to be produced and consists of the following steps [11, p. 14]:

(1) First, is the awareness of a problem. This may come from different sources including scientific literature, reports, and projects;
(2) Second, is a suggestion or proposal based on beliefs and possible solutions of how to solve or improve the problem discovered in Phase 1;
(3) Next, is the development phase. Here the novelty should be involved. There are no requirements for a formal proof nor should innovative construction of an artifact be introduced. The novelty is in the process of the design. For example, some learning objects based on sample software could be developed. In such a case, the innovation in principles which found the process of the design is important under the DSR paradigm;
(4) Later, the evaluation phase follows. Evaluation criteria are not always explicitly specified, but the design process should lead to the solution of the problem formulated in Phase 1. If not, and this is almost the case, the process should be continued from one of the previous steps as is shown in Figure 20 (adapted from [11, p. 15]);
(5) Finally, a conclusion should be provided.

It is important that knowledge gained in the effort of the process is "firm" and could be used and applied as generalized knowledge. Therefore, communication possibly in the form of journal or conference papers is important at this stage [10].

To position itself as a methodology in general, DSR should be aware and specify universal reasoning techniques. This is also important for practicing teachers and instructional designers, especially if DSR is intend to use within the scope of teaching techniques. Figure 21 (adapted from [11, p. 17]) presents the cognitive aspects of the DSR process.

Generally, the output of DSR should be design science knowledge. Depending on the maturity of the problem and solution domains, the output could be classified into inventions, adaptations, improvements, and routine design; there the latter is not considered as possible knowledge contribution. This classification is important for the purpose of this study, as it should be adapted to the needs of an educational process. The contribution of DSR is presented in Figure 22 (reprinted from [11, p. 19]).

What are the outputs of DSR? These are [154]: (1) constructs; (2) mod-

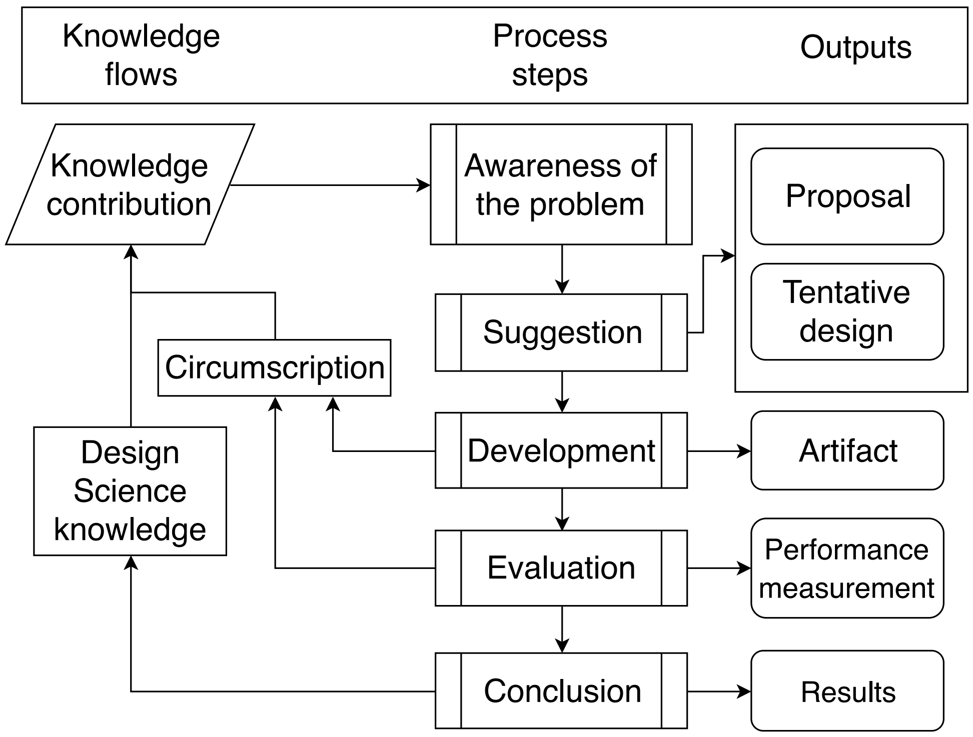


Figure 20: DSR process model. Adapted from [11, p. 15]

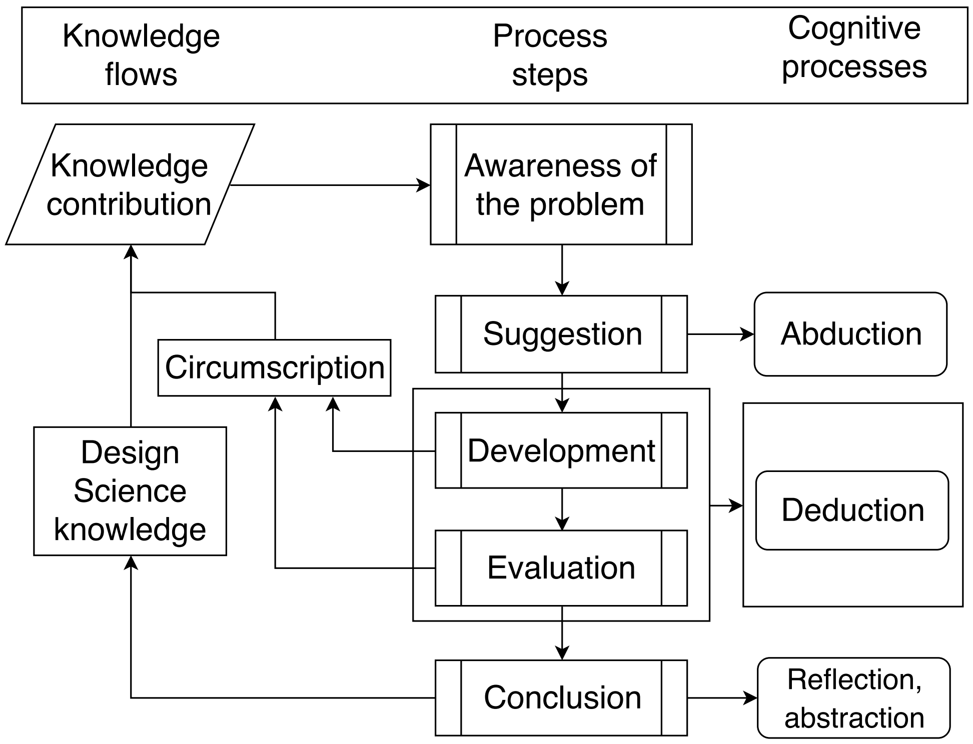


Figure 21: Cognitive aspects of DSR. Adapted from [11, p. 17]

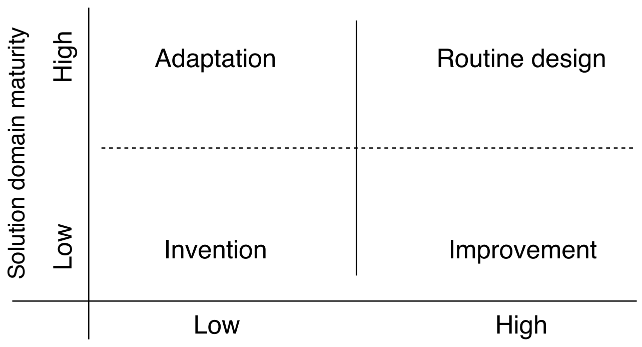


Figure 22: DSR knowledge contribution framework. Reprinted from [11, p. 19]

Table 6: DSR outputs. Reprinted from [11, p. 20]

	Output	**Description**
1	Constructs	The conceptual vocabulary of a domain
2	Models	Sets of propositions of statements expressing relationships between constructs
3	Frameworks	Real or conceptual guides to serve as support or guide
4	Architectures	High level structures of systems
5	Design Principles	Core principles and concepts to guide design
6	Methods	Set of steps used to perform tasks how_to knowledge
7	Instantiations	Situated implementations in certain environments that do or do not operationalize constructs, models, methods, and other abstract artifacts; in the latter case such knowledge remains tacit
8	Design Theories	A prescriptive set of statements on how to do something to achieve a certain objective. A theory usually includes other abstract artifacts such as constructs, models, frameworks, architectures, Design Principles, and design methods

els; (3) methods; (4) frameworks; (5) architectures; (6) Design Principles; (7) instantiations; and (8) better theories. Table 6 summarizes the outputs of DSR (reprinted from [11, p. 20]).

3.10.3 Example of the applications of Design Science Research for education technology

The educational research technology incorporates various methodologies and techniques. Some are better formalized, others are based on teacher expertize. Therefore, educational research technology is multi-paradigmatic in its nature [155]. At the same time, teaching technology becomes multi paradigmatic as well as modern teaching process incorporates various techniques and teaching methods and is always a combination of them that is used in practice [151, p. 3]. Another factor is that teaching is always a two-way process. Students come with preconceptions, and the role of the teacher is to detect and properly reflect existing situations. Teaching takes place in the social environment, and such social and cultural factors should be taken into account as well. All these factors make the modern teaching technology rather complex and challenging to develop. One of the possible recipes of how these problems could be solved is proposed by Laurillard [151, p. 211]. "Teacher should act as design scientists implementing the following strategy:

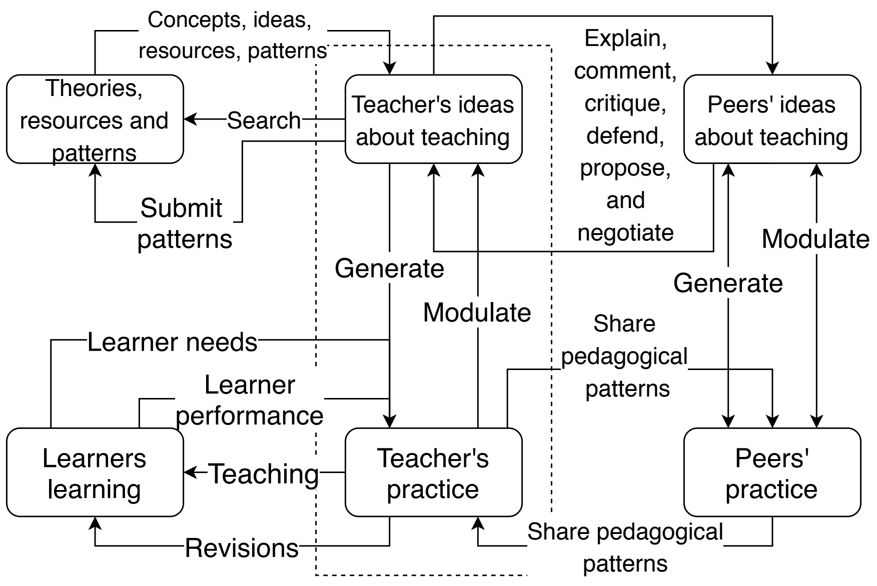
(1) keep improving their practice;
(2) have a principled way of designing and testing improvements in practice;
(3) build on the work of others;
(4) represent and share their pedagogic practice, the outcomes they achieved, and how these related to the elements of their design."

As it was already mentioned, communication and evaluation are important in DSR. At the same time, the achieved knowledge should be formalized and generalized. Laurillard [151] suggests the form of pedagogical patterns to be used for formalization. Pedagogical patterns as artifacts could form a basis for the implementation of methods of DSR to educational technology. Figure 23 (adapted from [151, p. 225]) presents a research cycle for the learner-centered approach to educational technology.

3.11 Conclusions

The context provides a clear set of its features, enabling further implementation of domain-specific models. These features could be specified as follows:

(1) interdisciplinarity enabling features; this is important for specifying the educational policy in general;
(2) constructionist educational approaches and the relevant features; these enable a proper direction for positioning of educational technologies in general, design and specification of instructional design approaches in particular;



Concepts, ideas, resources, patterns
Theories, resources and patterns
Search
Submit patterns
Teacher's ideas about teaching
Explain, comment, critique, defend, propose, and negotiate
Peers' ideas about teaching
Generate
Modulate
Generate
Modulate
Learner needs
Learner performance
Share pedagogical patterns
Learners learning
Teaching
Teacher's practice
Peers' practice
Revisions
Share pedagogical patterns

Figure 23: DSR cycle in the constructionist approach to educational technology. Adapted from [151, p. 225]

(3) model-based approaches; simulation-centered approaches; these features clarify the directions for development of constructionist educational methods;

(4) SLOs – clarify the description and specifications for LO as related to CS education in general and SLOs as related to SCE in particular;

(5) specification of SI related context provides a background for model-based and simulation-centred approaches;

(6) specification of DSR and its features within the context of the research; this enables further development and implementation of the eapplications of the DSR methodology in the research context and the domains in the study.

4 Meta analysis of the domain features of the scientific computing educational domain based on the TPACK model

4.1 Introduction

Štuikys proposes the next models for modelling a CS domain to specify: learning objectives, learning motivation, CS teaching content, technology used in CS e-learning. The set of properties for the presented model are provided [33, p. 93]:

- “properties of heterogeneity of CS domain, resulting to be represented by a number of a semantically identical and re-configurable feature models;
- two types of models: base and context model are presented with priority relation defined;
- SCV features specification is promoted;
- domain-based and feature-based semantic correctness are introduced;
- the list of characteristics to evaluate models is introduced”.

For the purpose of our research, it is important to stress that the CS and SC domains are closely related. Scientific Computing as the activity of making simulation relies on software and hardware solutions, or in other words, relay on the CS domain. Generally, it could be considered as a higher-level domain for the CS domain as it was presented in the above sections. The next idea seems to be very important for the purpose of our study as well. The proposed methodology [33, p. 97] could be considered as the mapping from of the problem domain (educational concept) to the solution domain (feature based concepts). Figure 24 (adapted from [38, p. 11]) presents a methodology for a feature model design.

4.2 Pedagogical content domain model

4.2.1 Educational technology for the scientific computing education

4.2.1.1 Interdisciplinary curricula: Existing approaches and perspectives for enhancements

First, we provide a general insight on the university curricula highlighting topics which are related to interdisciplinarity. The university curricula development is based on the next domains [156, 157]:

- educational philosophy, educational paradigms;
- didactics and epistemological perspectives;
- learning outcomes and course structure;
- assessment principles;
- organization of the learning process;
- instruction and teaching methods;
- educational technology and institutional organization;

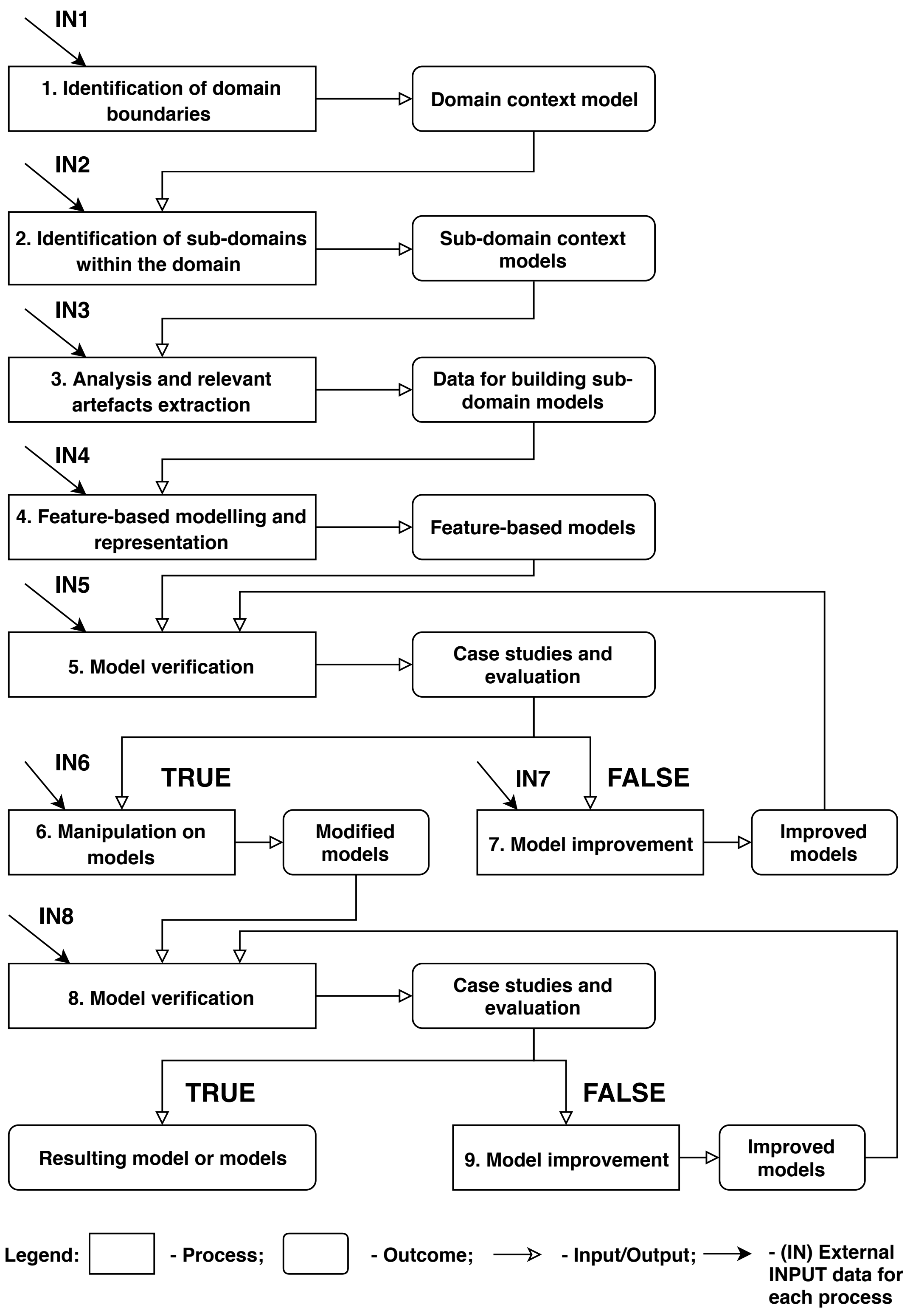

IN1
1. Identification of domain boundaries
Domain context model
IN2
2. Identification of sub-domains within the domain
Sub-domain context models
IN3
3. Analysis and relevant artefacts extraction
Data for building sub-domain models
IN4
4. Feature-based modelling and representation
Feature-based models
IN5
5. Model verification
Case studies and evaluation
IN6
TRUE
IN7
FALSE
6. Manipulation on models
Modified models
7. Model improvement
Improved models
IN8
8. Model verification
Case studies and evaluation
TRUE
FALSE
Resulting model or models
9. Model improvement
Improved models
Legend:
- Process;
- Outcome;
- Input/Output;
- (IN) External INPUT data for each process

Figure 24: Methodology for a feature model design. Adapted from [38, p. 11]

- limitations and perspectives.

Philosophy of education and educational paradigms can differ. It is important to indicate the assumptions for these aspects. We promote pragmatism [158] and postmodernism [159], especially given the obvious difficulties in developing interdisciplinary curricula. This attitude allows us to teach flexibility and learner-centered approaches to learning, and the importance of this will be seen in the following sections. In addition, it is important to emphasize that scientific research based on poststructuralism or postmodernism is universal and does not enter into a single discipline [160].

Didactics and epistemological perspectives are the following areas for study. Traditionally, methods of obtaining knowledge are linear, step by step or deeper methods of understanding. This is applicable to a discipline-oriented curriculum. Interdisciplinary knowledge is mainly based on social and institutional solutions and is not linear in nature. It is important to arrange the curricula in such a way that various epistemological ideas cross the boundaries of the various fields of knowledge [157]. In addition to expanding the epistemological perspectives, an interdisciplinary approach to curriculum development must take into account social, cultural and economic factors. The degrees obtained at such interdisciplinary courses could practically improve the income of students [161]. At the same time, this clearly corresponds to previously given philosophical assumptions. Another important task is to understand the epistemic nature of interdisciplinary knowledge. It is known [162] that cognitive processes which are involved into interdisciplinary based inquiry include, besides others, and a cognitive process of building integrations where such "...integrative devices include complex explanation and a focus on multiple causes for a multifaceted phenomenon" [157]. We will consider the described principle of designing "multifaceted" EC in the following sections.

Learning outcomes and the structure of the course should be based on the consideration of the interdisciplinary nature of knowledge. Knowledge develops as part of human contact and interaction [157]. Such interaction within the university community should be clearly indicated at the stage of curriculum development. Student learning outcomes include: flexible thinking, improved cognitive skills, tolerance for ambiguity, the ability to synthesize information, improve critical thinking skills [163]. The biggest problem is that interdisciplinary knowledge is usually not sufficiently structured. It is not always possible to determine the appropriate boundaries of knowledge. Disciplines develop in their own way. Understanding and processing knowledge related to another discipline is quite a challenge for the student and the teacher [164]. Therefore, there is a need for non-traditional approaches to structuring an interdisciplinary course.

Assessment principles should be developed taking into account the structure of interdisciplinary knowledge. A solution (and at the same time a challenge) might arise in view of the integrative nature of interdisciplinarity. Therefore, the main skill is the ability to integrate a different type of knowledge. Thus, the assessment can be based on an assessment of the improvement in the skills and cognitive abilities of students, including originality, non-traditional thinking skills, computational thinking skills [7], critical thinking skills, problem solving ability, the ability to synthesize and evaluate new information [164].

Organization of the learning process could be based on teamwork, the participation of students in discussions of interdisciplinary teams, design and research activities. This can happen through educational units that are integrated into the existing organizational structure of the university, such as autonomous colleges, cluster colleges, interdisciplinary departments, centers and institutes. At the same time, there may be non-traditional approaches, such as training communities, mass open online courses and multi-contact consortia [164].

Instruction and teaching methods for the interdisciplinary learning include the following processes [165, 157]: problem identification; definition of problem knowledge; clarification of relevant epistemological concepts; integrating interdisciplinary understanding. The nature of interdisciplinary knowledge contributes to team-oriented teaching technologies and project-based teaching methods. At the same time, online learning methods and computer enhancements can improve students' understanding and motivation. [166].

Educational technology and institutional organization should be adapted for the needs of interdisciplinarity. The solution could be: interdisciplinary learning groups; interdisciplinary learning environments; interdisciplinary doctoral programs; university center for interdisciplinary teaching and learning; training for e-technologies; alternative assessment techniques [157].

The reason for the limitations can be: institutional constraints; complexity of interdisciplinary integration; difficulties in developing the course. Motivation and prospects can include a holistic view of knowledge and integrative learning strategies [157].

4.2.1.2 A unifying approach for development of interdisciplinary curricula

One of the most important questions to answer is if it is possible to develop an interdisciplinary university curriculum using unifying approaches. Why is it so important? Educational institutions nowadays, for obvious reasons, struggle for the improvement of their scientific input, thus stressing the importance of the interdisciplinary research. This allows crossing the boundaries of the traditional science and increasing competitiveness [167, 168]. At the same time, there are various difficulties and obstacles in that way [169]. The need is to develop a unifying approach to the curriculum. Such approach could be based on SC education (see Figure 25).

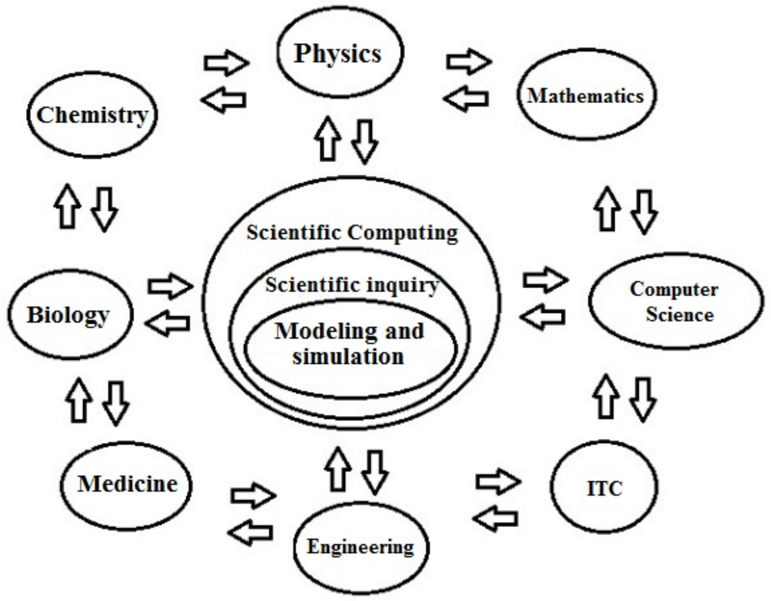


Figure 25: Interdisciplinary approach to university education

Generally, traditions in model-based simulations are based on different paradigms, which historically came from various different scientific schools and directions. There is no universal approach to modeling and simulations from the technical perspective. Every paradigm uses its own set of background theories and assumptions [3]. At the same time, such educational goals of simulation-based education as improvements of computational thinking, critical thinking, and problem-solving skills are quite universal [103, 170].

It could be possible to develop an interdisciplinary university curriculum based on these universal goals and focusing on SI using the model-based simulation approach. This could support a unification approach to teaching different sciences via looking for similarities in modeling and simulation paradigms. Such similarities could serve as bridges, which interconnect various disciplines. Another possibility for unification lies in the context of system modeling and simulation science. This approach, as opposed to traditional and based on theoretical backgrounds approach, is completely application oriented and focuses on practical solutions in various fields of applications. Experts in the field of system analysis could highlight the relevant expertise, which could be used by students. Students could incorporate this knowledge into their own simulation building activities. As to developing instructional strategies, the unifying approach could be found incorporating the model-centered approach to instructions [3].

4.2.1.3 Possible approaches to interdisciplinary instructional design

The main aspects to be considered in the process of instructional design for the interdisciplinary curriculum are:

(1) degree of interaction of people outside the same disciplinary community;
(2) degree of integration between knowledge bodies relevant to disciplines;
(3) the presence of a comprehensive problem, theme or topic, which stimulates interdisciplinary interaction [171].

The key point of the presented definition is the existence of a comprehensive problem, therefore the type of knowledge studied by the interdisciplinary curriculum is a priori by its nature. Interdisciplinarity examines issues and problems that do not exist within the disciplines [157], so it is possible that there are no available training resources. The model for instructional design should be project-oriented and research-oriented. The main task is to provide a formal instruction based on a formal approach and methods. The generalized view on interdisciplinary instruction is provided in Figure 26 (adapted from [172, p. 4]).

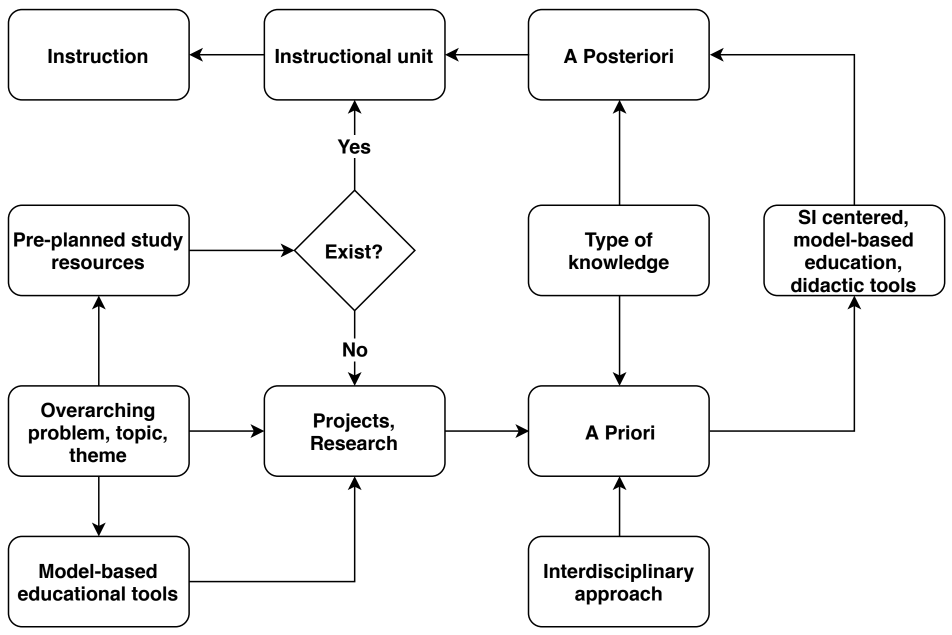

Instruction
Instructional unit
A Posteriori
Yes
Pre-planned study resources
Exist?
Type of knowledge
SI centered, model-based education, didactic tools
No
Overarching problem, topic, theme
Projects, Research
A Priori
Model-based educational tools
Interdisciplinary approach

Figure 26: The generalized view on interdisciplinary instruction. Adapted from [172, p. 4]

The presented approach to the instruction, focused on the scientific inquiry, contains only guidelines for further implementation in the process of practical instructional designing. This approach requires further investigation and is positioned as a topic for further research.

4.2.1.4 Scientific computing education: The scope and definitions revisited

The definitions of SC and SC education are going to be analyzed and revised, taking into account the previously presented descriptions. First, the following definition (first modification of definition provided by Golub [66]) is proposed: "Scientific computing is a scientific discipline of conducting (designing and implementing) scientific inquiry (in the field of interest) via computer simulations."

Two central meanings of the presented definition should be discussed:
(1) SI and
(2) computer simulations.

First, thre is a need to revise the meaning of the scientific inquiry that was described earlier in this study. It should be revised as a result oriented activity. Jadrich [131] suggests defining SI as an activity of creation, testing, and refinement of scientific models. Generally, this definition corresponds well with the purpose of our studies, although the next remark should be made here. Testing of models generally means in practice one or other types of simulations or computer simulations in particular. Therefore, it could be refered to the earlier presented definition of so-called "model-based" simulations [3]. The revised definition is:

"Scientific inquiry is an activity of conducting scientific model-based simulations".

Accordingly, SI in the field of computing could be defined as an activity of conducting model-based computer simulations. Therefore, the earlier proposed definition could be revised as follows:

"Scientific computing is a scientific discipline of conducting scientific inquiry using computers".

The following remark should be made here. The area of interest is not specified. Actually, the presented definition could be applied for any area in which simulations could take place including multidisciplinary areas. The focus in the presented definition is the model. The provided definition is rather broad. It could be applied to any field, including interdisciplinary, which accepts simulation. Consequently, SC education could be defined as an educational process in the field of SC.

The model becomes the central part under the revised definition of the SC, therefore the process of scientific activity converts here to the process of designing artifacts in general or designing computer simulations in particular. The research methodology that could be implemented is DSR, which allows unifying the process of research for SC thus providing new opportunities for scientists, educators, and policymakers. The DSR cycle [9], could be related to SC as following:

- Environment
  - organizational Systems supporting infrastructure and organizational structure for simulations;
  - technical systems hardware, software as related to simulations;
  - DSR design appropriate models including conceptual model, mathematical model, computational model, simulation solutions and process simulations
- Foundations

  – theories in the field of SI, theories in the field of computations and simulations.

Under the previous definitions, the main attention to teaching SC is shifted from the traditional approach like focusing on various internal parts as presented in Figure 27 to focusing on the overall process of DSR in general. The main teaching task could be formulated as how to teach students to conduct SI using computers. It is clear that this new paradigm also includes all steps of the previous definition but the main difference is that this new paradigm systematizes the field and provides the unified research method DSR. The aim of the educator is to develop an integrated educational environment as presented in Figure 27 (adapted from [3, p. 145]).

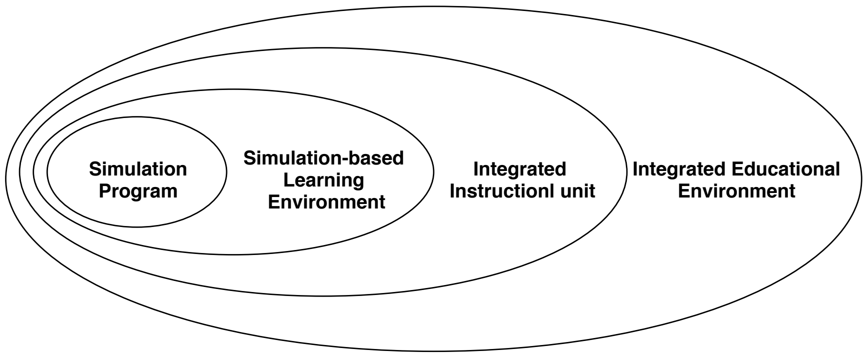

Simulation Program
Simulation-based Learning Environment
Integrated Instructionl unit
Integrated Educational Environment

Figure 27: Levels of instructional structure for SC education. Adapted from [3, p. 145]

4.2.1.5 Teaching Design Science Research for scientific computing education

The presented paradigm provides us a way for developing of educational technology for SC. The main teaching goal is now shifting from teaching students about how scientists do scientific research to teaching how to do scientific research using simulations and computers. Generally, this approach closely corresponds with the earlier described definition of SI as a model making activity [131, 146]. The classical view on how design is conducted in engineering is presented in Figure 28 (reprinted from [20, p. 75]). Attention should be paid to the following. Design Science Research as a methodology for creating innovative artifacts. Therefore, this should be the focus of teaching. In practice, it 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 imprinted into the learning environment as an integral part of it. Such learning environment could be named the DSR enabling learning environment. 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 become a process of "turning the model" enabling various planes of the model to be discovered.

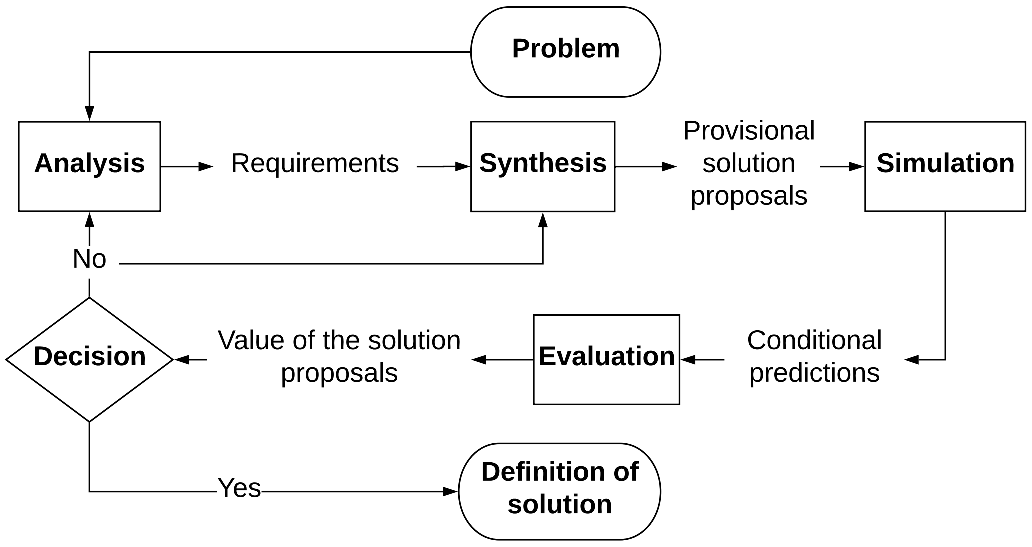


Figure 28: Design cycle by Eekels and Roozenburg. Reprinted from [20, p. 75]

An analog with floodlighting a multifaceted three-dimensional object with a stationary projector could be provided. Therefore, to understand what the flood-lighted object looks like, an activity as turning the object in space should be undertaken. One could achieve full understanding about the external structure of the object only after it is turned and observed all around. To construct such the DSR enabling learning environment is rather a complex and challenging task. First of all the problem should be transformed into a multifaceted problem. In addition, this is not decomposition, as decomposition is mainly related to the internal structure, but a process of faceting the initial problem. Such process is presented in Figure 29. Solving each of subproblems should allow, by "turning" the task, to enable further observations. This approach could be observed from epistemological perspectives as well. It allows including serious gaming elements into the learning process which simulates students' curiosity and motivates students for further studies.

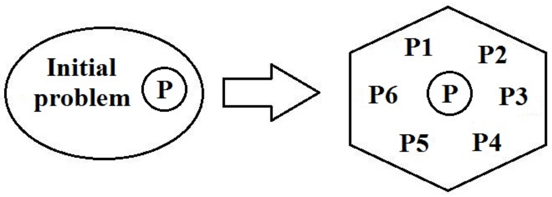


Figure 29: Faceting the initial problem

Taking into account the presented scheme the earlier presented design cycle should be adapted to the needs of the educational process (see Figure 30).

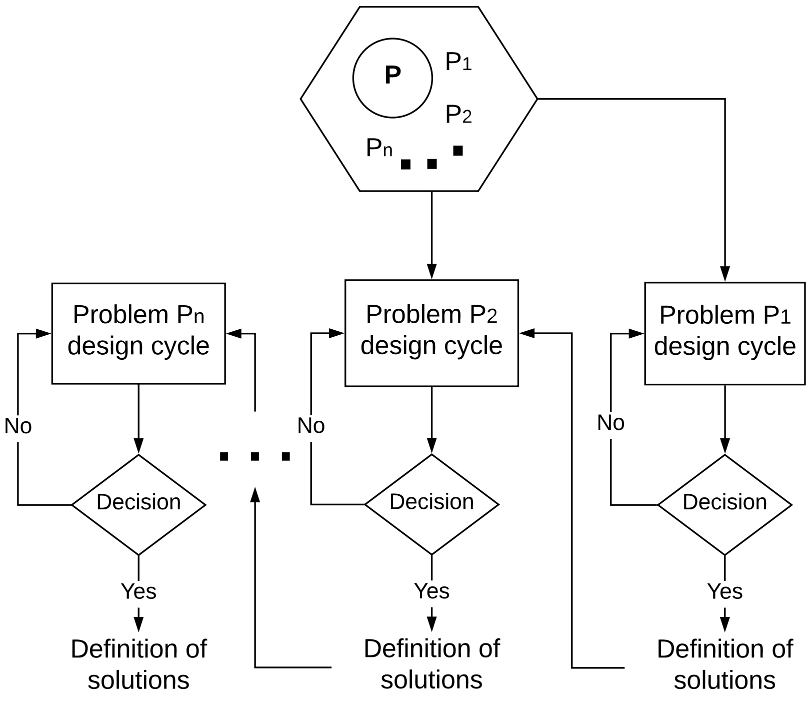


Figure 30: Applications of the Design Cycle model for Scientific Inquiry-Centered education

4.2.2 Didactic aspects of scientific computing education

4.2.2.1 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 [173–177]. This approach differs from the presented earlier organizational approach and focuses on didactic schemes, general computational thinking skills, and practical abilities to conduct simulations. This approach 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 (see Figure 31, reprinted from [174, p. 3]). Such an approach is considered to be demotivating for students, especially for topics where sufficient theoretical background and a large amount of pre-knowledge are required [174, 178].

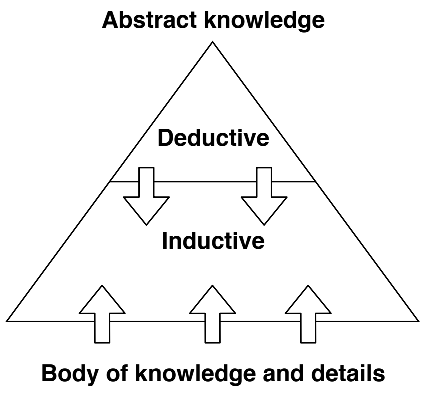


Figure 31: Direction of informational flow in deductive and inductive pedagogy. Adapted from [174, p. 3]

On the contrary, an inductive approach is promoted as such an alternative that could improve students' positive attitude to learning science. "The inductive approach to instruction, by contrast, first presents students with a problem, a case, or data from an experiment. Students are then guided to explore underlying facts, issues, and the like. As the culminating step, students are led to acquire on their own an understanding of the underlying concept or organizing principle [179]. Inquiry-guided learning, problem-based learning, and project based learning are all among forms of inductive instruction [180]. While empirical evidence suggests that the inductive approach to instruction is superior and that it fosters greater intellectual growth [179] prudent educators should take advantage of different approaches of teaching" (from [174, p. 4]).

To overcome the described problems of the deductive approach, "Modeling-and simulation-based computational pedagogy" [174, p. 4] is offered. Such pedagogy allows "cycle back and forth between the inductive and deductive approaches to learning" [174, p. 4] with the assistance of modeling and simulation tools. This is a very promising approach, especially for K-12 or starting university levels. At the same time, modeling and simulation approach is promoted as an example of the constructivist approach.

The theoretical foundations of Computational Pedagogy are based on the concept of cognitive retrieval presented by Brown et al. [181]. This so-called "interleaved retrieval" practice forms a cognitive foundation for the interdisciplinary computational pedagogical content knowledge. "Interleaving retrieval practices by weaving together multi-disciplinary features around a common topic (i.e., interdisciplinary education) have great advantages for gaining deep and lasting knowledge" [176, p. 2]. The process of a model-based reasoning is presented as an inductive/deductive cycle of modeling and retrieval of the models (see Figure 32, reprinted from [176, p. 3]). This process "is consistent with the dual deductive and inductive process of computational modeling and simulation" [182, p. 4]. By using modeling and simulation, the learner could receive a feedback from the simulation environment, thus promoting his or her construction of knowledge. "Creating a model through a step-wise process and running it at each stage of the development has the added advantage that learners get immediate feedback about their work. It may be used in situations when learning about the underlying theories and mathematical concepts that are important. Through this process, learners can be led to develop an understanding of scientific reductionism that studying a system or solving a complex problem requires breaking the system into its components or the complex problem into smaller chunks (decomposition). Using models and simulations, learners become actively engaged in'doing', rather than passively 'receiving' knowledge. In so doing, the learner becomes the center of the learning process, allowing self-interpretation of the problem and revise it if necessary, mediated by own biases, beliefs, preconceptions, prior knowledge and observations" [182, p. 6].

4.2.2.2 The problems of the presented Computational Pedagogy approach

First, from the educational perspective, there is a big difference between simulation making and simulation using activities [3]. Moreover, there are many doubts about the effectiveness of the simulation-using tools for enhancing learner's motivation and its conceptual understanding of scientific

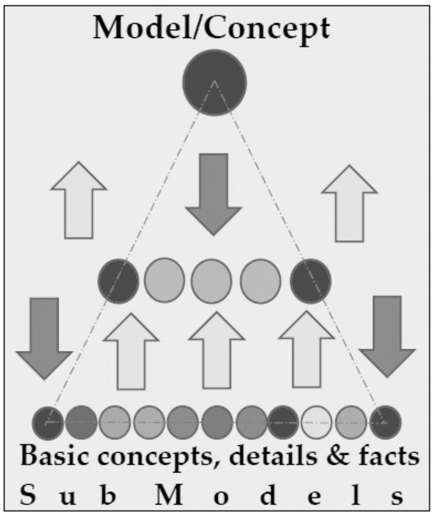


Figure 32: Scientific methodology of modeling-testing-remodeling process used in conducting research. Reprinted from [174, p. 3]

topics [183, p. 2]. 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 artifact. Such an artifact could be considered from different points of view. “When a person uses an artifact to accomplish some task, the outside observer sees the system view, the total structure of person plus artifact in accomplishing that task. The person, however, sees the personal view: how the artifact has affected the task to be performed. Under the system view on a cognitive artifact, we see the entire system composed of the person, the task, and the artifact. Seen from this perspective, the artifact enhances cognition, therefore, with the aid of the artifact, a system can accomplish more than without the artifact. Under the personal view on a cognitive artifact, that of the individual person who must use the artifact, the view of the task has changed: thus, the artifact does not enhance cognition it changes the task. New things have to be learned and old procedures and information may no longer be required: The person’s cognitive abilities are unchanged” (see Figure 33 and Figure 34, adapted from [184, p. 3]).

Norman provides the following illustrative example [184, p. 3]: consider a “todo” list. Such a checklist, for example, developed for aircraft pilots, from the system point of view, improve pilots’ cognitive abilities by enhancing the memory of pilots. Therefore, from the system perspectives, this is a memory enhancer. From the pilot perspectives or from the personal view, using a

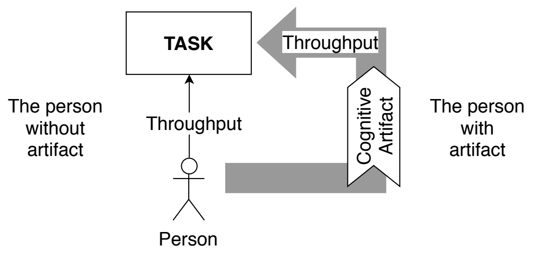


Figure 33: System view of a cognitive artifact. Adapted from [184, p. 3]

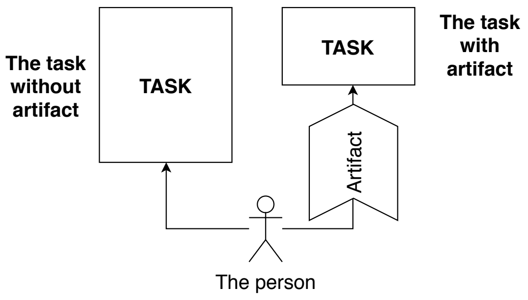


Figure 34: Personal view of a cognitive artifact. Adapted from [184, p. 3]

list is just another task requiring a different type of activity. Without a list, a person should remember all the "todo" tasks. For using the list, the next task should be performed:

(1) the construction of the list (in the described case is done beforehand by the third 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 personal (the learner's) view (if considering the person is involved in to only (2) and (3) types of activities) his cognitive abilities are degraded as instead of trying remembering 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 [185, 186]. As related to simulations as ITC tools, simulations, as cognitive artifacts from the user perspective, could be very harmful and demotivated. It is obvious that there is a kind of a shift in tasks and cognitive processes involved into the process of a simulation, therefore, educational simulations (the ones, which are intended to use in a simulation-using manner) should be carefully developed and tested [187].

4.2.2.3 Abductive reasoning as the key to discovery and innovations

From the point of view of educational technology, as related to academic teaching, the psychology of situated cognition provides recommendations of how the learning environment should be constructed. Obviously, there is a clear distinction "between natural environments which afford the learning of 'percepts' in everyday life, and unnatural environments" [106, p. 20]. Relying on the positions of Pragmatism, simulation could be considered as a tool for "grounding" the learner to such artificial environment in terms of grounded cognition [188, 189]. Using computer simulations could be considered as a kind of grounding via situated simulations [188, 190]. How this grounding could be practically achieved? Simulation based learning could be described from the positions of progression of mental models [3, p. 196]: "...beginning with a student's initial model of an examined system and developing into a target conceptual model-presumably the same one underlying the simulation's computational model. Moreover, to arrive at the target model, students must first develop their own intermediate conceptual models, which are mental models expressed as cognitive artifacts" (see Figure 35, adapted from [3, p. 197]).

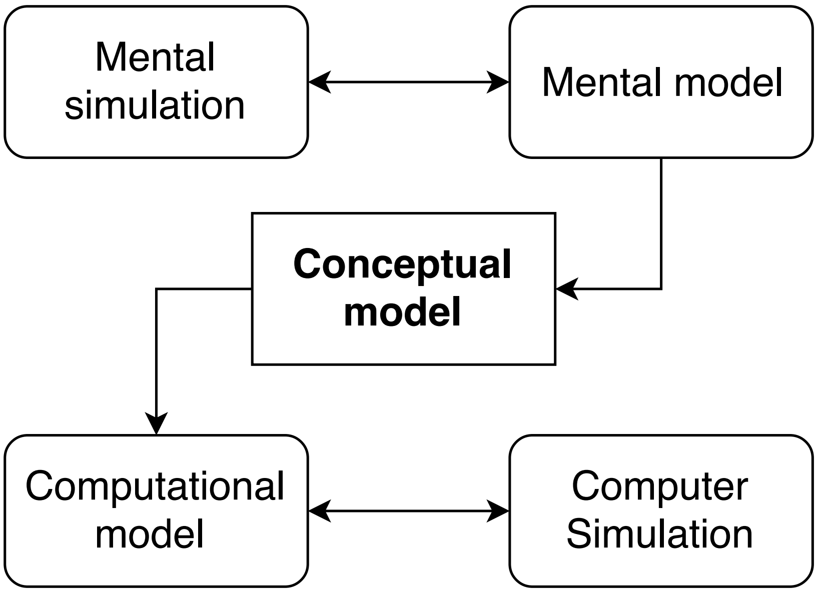


Figure 35: The mediating role of conceptual models. Adapted from [3, p. 197]

The presented approach is focusing on a conceptual model as on an intermediate for grounding to take place. How could such conceptual models be developed? Attention schould be paid to the following. Conceptual models are based on mental models, so first,the following corresponding question should be asked: how a mental model is developed by the human brain. The approach, presented by Thagard, considering "mental models as representations consisting of patterns of activation in populations of neurons" [31, p. 444] is refered to. This cognitive model-based approach allows to overcome the limitations of sentential models of theoretical abduction [29], [30, p. 31] and expand Peirce [39] ideas of how mental models can "contribute explanatory reasoning" going beyond verbal information and including "visual, olfactory, tactile, auditory, gustatory, and even kinesthetic representations" [31, p. 449]. Considering mental representations as patterns of firing in neural populations, the process of constructing of mental models could be presented as a chain of patterns developing by the process of causal correlations [31, p. 452]. Such an approach could be used to provide explanations of how abduction could generate new ideas. The neural model of abduction presented by Thagard and Stewart [32] implements a fully multimodal convolutional model of "creative conceptual combination" describing "many kinds of creativity and innovation, including scientific discovery, technological invention, social innovation, and artistic imagination" [31, p. 453]. The human brain is adapted to powerful learning mechanisms: "One of these learning mechanisms is an abductive inference, which leads people to respond to surprising observations with a search for hypotheses that can explain them. Like all cognitive processes, this search must be constrained by contextual factors such as triggering conditions that cut down the number of new conceptual combinations that are performed" [31, p. 458]. It is obvious that such triggering conditions use circumscription in the form of a previous experience [191] for eliminating the inapplicable transactions.

4.2.2.4 Enhancing modeling process by circumscription and abductive reasoning

To formalize the presented approach, the model of modeling diagram that describes how students produce their models as mental and as expressed ones [192], will be used. The presented approach considers modeling as a "non-linear creative process comprised of multiple and complexes stages mainly concerning with acquiring information about the entity that is being modelled (from empirical observations and/or from previous knowledge); producing a mental model of it; expressing that model in an adequate mode of representation, testing it (through mental and empirical experimentation) and evaluating its scope and limitations" [193, p. 32]. Figure 36 (reprinted from [193, p. 33]) presents the described diagram.

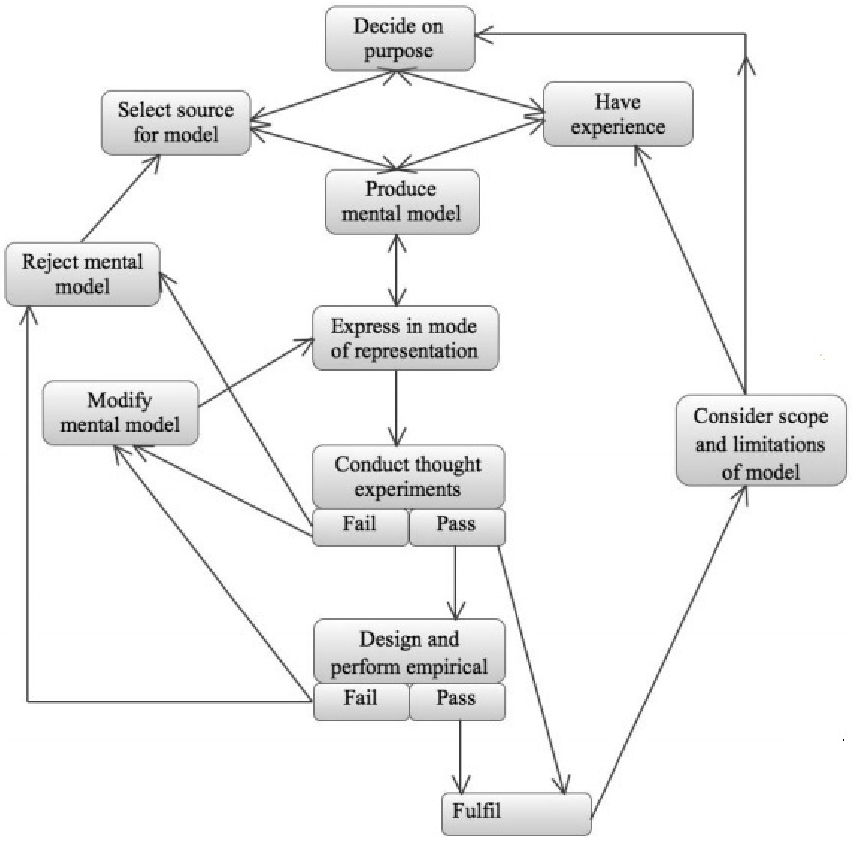


Figure 36: "Model of modelling" diagram. Reprinted from [193, p. 33]

From the perspectives of cognitive reasoning, the presented "model of modeling" could be aggregated as follows. First, the propositional phase is needed for the generalization of existing information by the inductive reasoning process. Next, the process of production of mental models that are based on the existing information requires a kind of hypothetical model-based reasoning to be involved, therefore, as it was described earlier, a kind of abduction (or grounded abduction in this particular case) needs to be implemented in such a case. The next steps are based on such classical reasoning method as deduction requiring some form of conceptualizations via an empirical design process. Finally, all these processes are based on existing knowledge and skills, which provide a sort of constraints in the form of circumscriptive reasoning. This practical model, as related to modeling and simulation in education, is generalized by Franco Landriscina and presented in Figure 37 (adapted from [3, p. 204]) in a modified version adding the described reasoning methods. The processes of abduction and circumscription are extremely important, as they provide a plane dividing simulation using educational methods from simulation making educational methods.

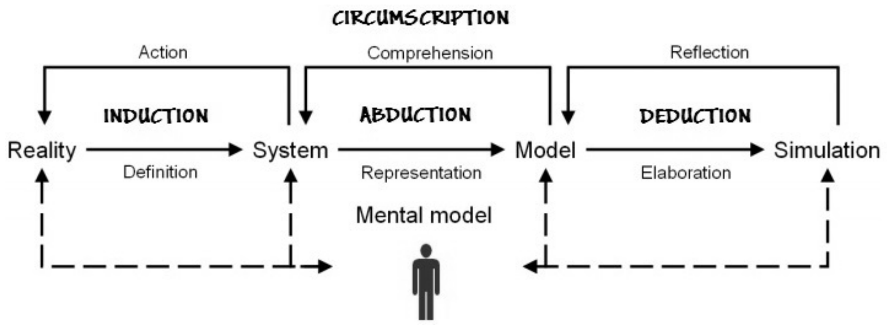


Figure 37: The epistemic cycle. Adapted from [3, p. 204]

The presented approach provides a clear picture of the common practice of eliminating abduction and circumscription from the educational process. This a kind of a usual practice that is based on complete reliance on ITC tools as on a sort of magic wand could totally defocus the educational goals from focusing on teaching SI and scientific reasoning to simply training additional skills of using ITC tools. Obviously, the practical implementation of the described approach is not a trivial task. This requires as additional efforts for pre- and post-training of STEM teachers as well as additional educational programs to be developed. Moreover, the lack of practical examples for such kind of activities stimulates a strong movement for education without computers like presented by Bell et al. [194], that is, behind-the-scenes, a movement for introducing abduction and circumscription into pedagogical practice. Another reason for such popular "anti-computational" movement could be a strong influence of "classical" instructional techniques, a sort of "Instructionism vs Constructionism" as it was defined by Papert and Harel [25, p. 9]: "Instructionism vs Constructionism looks like a split about strategies for education: two ways of thinking about the transmission of knowledge. But behind this, there is a split that goes beyond the acquisition of knowledge to touch on the nature of knowledge and the nature of knowing. There is a huge difference in status between these two splits.

The first is, in itself, a technical matter that belongs in an educational school course on 'methods'. The second is what ought properly to be called 'epistemological'."

4.2.2.5 Enhancing Computational pedagogy by Design Science Research

The methodology of a DSR gives a pathway for formalization of SI centered approach that is based on developing of model-based scientific simulations. This methodology provides a set of analytical techniques, which are based on circumscription and abduction reasoning. The structure of DSR (see Figure 20, Figure 21) very clearly corresponds with the presented educational models (see Figure 36, Figure 36). 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:

(1) First, to formalize the process of design model-based simulations as cognitive artifacts. 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 (see Figure 15). 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;

(2) Next, to introduce DSR as a practical design tool for the future students' scientific activity. For this purpose, an example of one or DSR formalism 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.

As an example, the methodological guidelines for instructional technology are presented in Table 7.

Table 7: Formalization for the methodology

Abbreviation	**Name**
(**R**)	**Introduction stage**
(**R** → **I**)	→ Information sources
(**R** → **P**)	→ The importance of the problem
(**R** → **A**)	→ Possible practical applications
(**R** → **O**)	→ Detailed formalization of the research question
(**A**)	**Analytical stage**
(**A** → **C**)	→ The causes of the problem
(**A** → **R**)	→ Functional requirements
(**A** → **P**)	→ Expected performance
(**A** → **O**)	→ Operational requirements
(**A** → **H**)	→ Formal review
(**A** → **F**)	→ Factorization
(**I**)	**Implementation stage**
(**I** → **P**)	→ Proposition
(**I** → **M**)	→ Design
(**I** → **I**)	→ Implementation and coding
(**I** → **E**)	→ Evaluation
(**I** → **C**)	→ Clarification
(**I** → **S**)	→ Summarizing and conclusions
(**I** → **G**)	→ Generalization
(**I** → **F**)	→ Final report

The process of implementation based on the provided guidelines is presented in Figure 38 (adapted from [20, p. 118]) which describes the application of DSR as an SI centered educational tool.

The research question **R** is provided by an educator in the form of a project or a problem to be studied in detail [195–197]. The learner should start with developing some definitions and generalizations and specify how the system to be modeled should be defined (from reality to a system, identification of the problem). This identification should be provided in the form of definite and clear answers to the next questions (adapted from [20, p. 118]):

(**RI**) - what are the sources of information?

(**RP**) - why does problem seem to be important?

(**RA**) - what are possible practical applications of the problem?

(**RO**) - the output of the step, should be a detailed formalization of the research question.

The next step (awareness of the problem) (**A**) is to systematize the problem in the form of relations with the outer structure and impacts of the

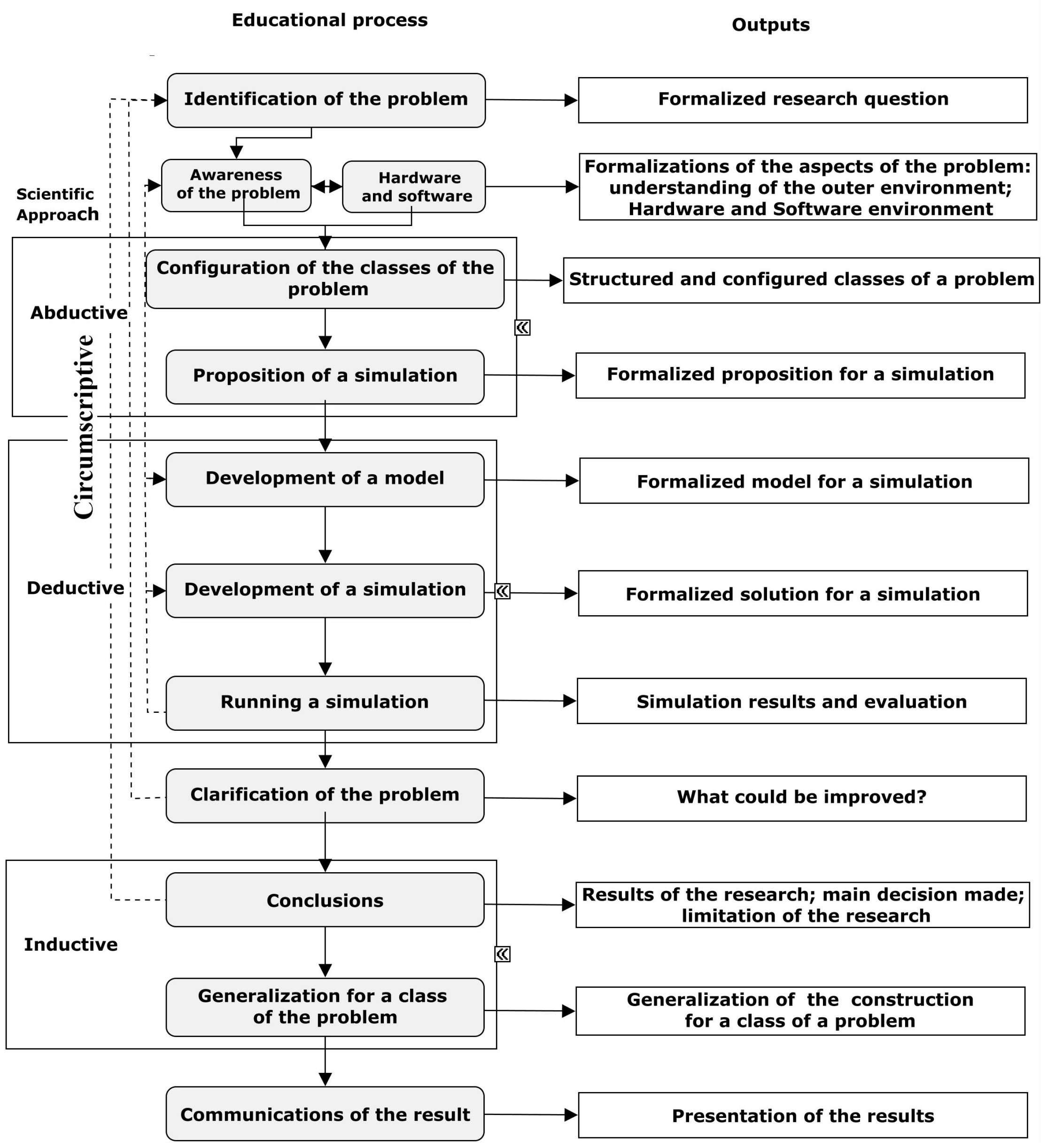

Educational process
Outputs
Identification of the problem
Formalized research question
Awareness of the problem
Hardware and software
Formalizations of the aspects of the problem: understanding of the outer environment; Hardware and Software environment
Scientific Approach
Configuration of the classes of the problem
Structured and configured classes of a problem
Abductive
Proposition of a simulation
Formalized proposition for a simulation
Circumscriptive
Development of a model
Formalized model for a simulation
Deductive
Development of a simulation
Formalized solution for a simulation
Running a simulation
Simulation results and evaluation
Clarification of the problem
What could be improved?
Conclusions
Results of the research; main decision made; limitation of the research
Inductive
Generalization for a class of the problem
Generalization of the construction for a class of a problem
Communications of the result
Presentation of the results

Figure 38: SI centered educational tool. Adapted from [20, p. 118]

environment:

(**AC**) what are the causes of the problem?

(**AR**) what are functional requirements?

(**AP**) what is the expected performance?

(**AO**) what are operational requirements? As related to simulations, the possible modelling and computing resources should be studied in detail.

(**AH**) Then, a formal review based on the previous study should be provided and approved by an educator.

(**AF**) The fourth step is to provide the generalization (factorization) of the previous studies in the next form: what class of similar problems could be named?

The final step (implementation) (**I**) is to propose a practically implementable solution for the problem:

(**IP**) The learner starts from proposition for the simulation solution (from system to model, the representation process): how to implement a simulation solution for the previously formulated problem

(**IM**) After, the design process for a specific cognitive artifact for the presented problem has to be solved: what are possible models of the system?

(**II**) Then, the implementation phase follows (from model to simulation, the exploration process): what is the practical solution for simulation?

(**IE**) Evaluation phase follows the next. In this phase, the learner runs the simulation on the computer and evaluates the results.

(**IC**) After, the clarification of the problem could be done and the steps are repeated if needed.

(**IS**) Finally, the learner summarizes the problem and

(**IG**) provides a kind of generalizations in the form of

(**IF**) the final report. All these steps provide a formal basis for project-based research in the form of developing of model-based scientific simulations.

The presented approach provides a practical educational methodology for the constructionist project-based learning. The aim of the presented methodology is to support the universal approach to the university STEM education, thus enabling a common basis for interdisciplinarity and innovations. Relying on Computational Pedagogy, abduction and circumscription could be next introduced into the practice of the educational process. This is important for a modern scientific environment as this enables creations and scientific innovations. At the same time, such well-developed approaches as a model-based approach to instruction and DSR are implemented in the form of scientific-inquiry centered pedagogy for university STEM education. The model-based approach provides a solid foundation for teaching via development of model-based scientific simulations, enhancing scientific-inquiry and project-based constructionist teaching methods. Design Science Research provides clear formalization in the form of universal educational techniques. Under these considerations, students could act within an interdisciplinary group as quasi-researchers generating hypotheses, designing simulations, evaluating the results and using DSR techniques. Teachers should provide a quasi-research environment in the form of predesigned multifaceted models, educational instructions, and seamless theoretical backgrounds. Figure 39 presents an outline of the methodology in the form of a feature model.

4.3 Domain model of the educational content

4.3.1 Designing content for Scientific Computing education

The simulation-based learning environment requires a systematic approach to design its content. The designer should enable a smooth pathway from the concept of the learning environment to practical solutions of problems, which students could simulate using computers. The key factor, which is important here, is to "understand the ways in which simulation can foster learning in so many different contexts" [3]. An appropriate way of learning is only the way of learning by building simulations as opposed to the way of learning by using simulations. Landriscina [3, p. 99] provides the following explanation: "…students must use either a programming language or the features of a given modeling and simulation software environment to build a simulation model on their own. To achieve this aim, they must:

(a) analyze a specific system;
(b) develop a conceptual model of it;
(c) create a computational model;
(d) implement the computational model as a simulation program;
(e) conduct numerical experiments on it to validate the computation model;
(f) Lastly, they can use the simulation program to solve a problem or

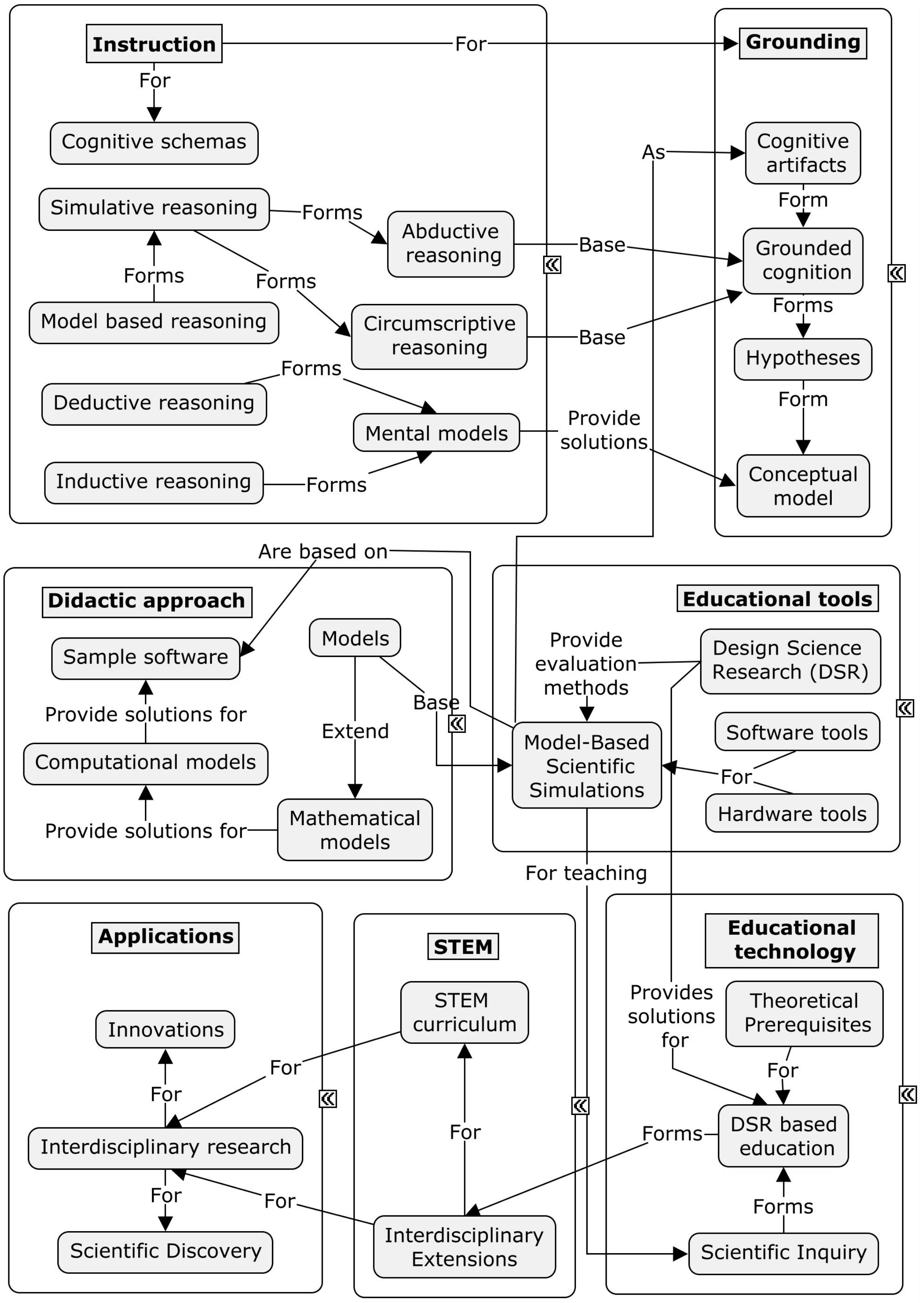


Figure 39: Feature model for DSR and SI centered pedagogy for grounding the learner's cognitive models in the form of a concept map

understand the causes of the phenomenon under study."

Each of these activities requires understanding, reasoning, and prediction abilities and the construction of mental models thereby. Building a simulation model is a challenging task in itself; also this is a challenging task in an instructional context. The pre-knowledge of students differs. In such a case, an educator could provide a preprogrammed model for simulations and shift the educational process from simulation-making to simulation-using, which is not so effective as the previous one, but supports a "mediated" style of teaching [106, p. 21] and fits the previously described teaching strategy. Generally, the role of instructional support is extremely important, as the main goal, besides training particular students' skills, is to develop students' understanding of the provided concepts. This instructional support includes [3, p. 100] background information, questions, hints, explanations, exploration guides, exercises, graphing tools, planning tools. However, an important observation is that this support should be provided "on the fly" and using a seamless approach, as previously described. Therefore, the "traditional" style, which is mainly oriented to requirements for prerequisites, should be transformed to the seamless style, with instructional support provided in parallel. At the same time, such transformations will improve students' motivational factors (Figure 40, adapted from [3, p. 101]).

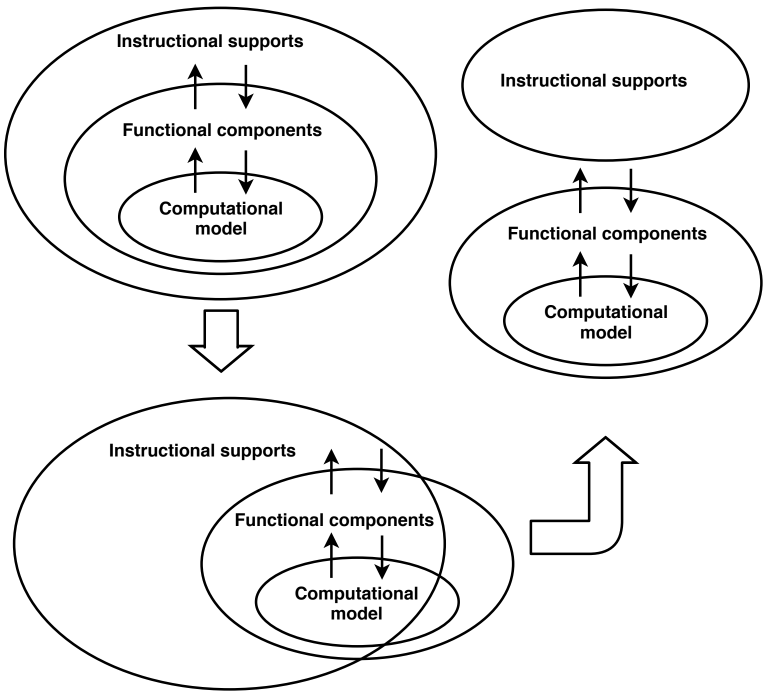


Figure 40: Seamless instructional support. Adapted from [3, p. 101]

This concept is closely related to the concept of “microworld” intoduced by Seymour Papert [24, 3] and to the similar concept of synthetic environments [3, 198–200]. The main difference is that simulation-based environments are artifact-centered and have a clear focus aimed at the central role of an artifact in the form of a model for simulations. It is also important to stress the difference between simulation using and simulation building activities. The difference is clearly seen if a well-known analogy or the so-called “black box” model will be applied. Franko Landrisina [3], as opposed to a “black box” as a model of learning by using simulations, suggests a “glass box” as a model describing learning activities of simulation-building.

Another important issue to be considered is the problem of the cognitive opacity of simulation models [3]. This problem, as refers to SC education, brings a new insight to the earlier discussed so-called “situated learning” approach [106, 201]. The situated environment should be designed in a manner which enables using a set of “rendering” the model features, as well as in the form of instructional design tools that brings such features into the environment. This is an absolutely challenging task, as it also requires teacher expertise to be transformed into a set of instructions. Moreover, this requirement is based on the prediction that such expertise is presented explicitely [202, 203]: “. . . these two are not sharply divided. While tacit knowledge can be possessed by itself, explicit knowledge must rely on being tacitly understood and applied. Hence all knowledge is either tacit or rooted in tacit knowledge”.

It is important to stress, that namely knowledge but not skills be considered as “tacit”. In spite of a seeming contradiction, namely, knowledge, related to cognitive concepts, involves such cognitive dimensions as schemata, paradigms, mental models, etc. [203]. Is it possible to make this knowledge explicitly available? It could be possible by practical experience or interaction with experts [203] and in the context of educational environment – teachers. Therefore, the mediating role of a teacher becomes even more important. A teacher should focus not only on mediating the situated learning environment but also on transferring his personal and best practice expertise in the field of model-based simulations. At the same time, simulation-based learning involves an epistemic interplay among different kinds of students’ mental models which are developed during simulation activities [3]. Therefore, a well-designed learning process should be aware of the process of constructing students' mental models (see Figure 41, adapted from [2, p. 29]).

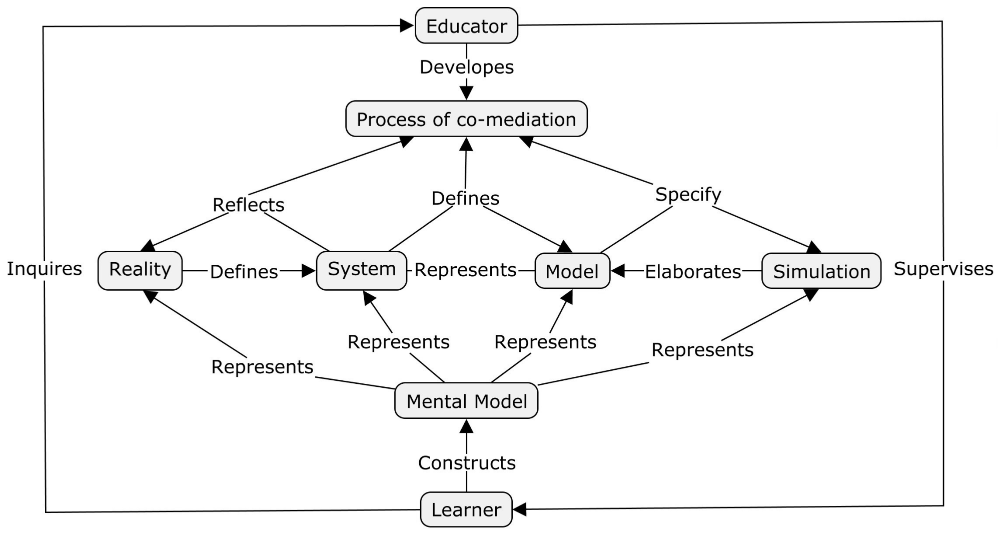


Figure 41: Epistemic cycle of a simulation and mental model. Adapted from [2, p. 29]

The focus of education is moved to the improvement of students' skills of building simulation models. These skills should provide universal prerequisites for the complete educational environment. At the same time, the content of CS, information, and communication technology courses could be revised focusing on modeling 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 DSR 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.

4.3.2 Main design principles for the model-based scientific inquiry centered Scientific Computing introductory content

First, accents of SC education should be refocused on teaching the basic principles of model-based simulations and training 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, brings possibilities of gamification into the learning process. Designing the content, an integrated approach considering all phases of SI should be undertaken. As an example it could be provided a class of models, which could be based on recurrent equations of the next form:

$$n + 1 = f(n, n - 1, \dots) \tag{2}$$

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

$$F(n + 1) = F(n - 1) + F(n - 2), n > 2; F(1) = F(2) = 1 \tag{3}$$

In spite of its simple form, these examples become a basis of the whole unit focused on modeling scientific problems, which are based on recurrences, including stochastic recurrences. First, a scientific problem, for example like analysis of the population growth problem (**P**) [204, p. 71], [205, p. 311], [206] 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 (**P1**) could be modified by asking about the time 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 suite 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):

$$F(n) = \frac{(1 + \sqrt{5})^n - (1 - \sqrt{5})^n}{2^n \sqrt{5}} \tag{4}$$

The next possible step is to introduce the Fibonacci primes. So the modified problem (**P3**) could be formulated as 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 second step is to involve optimization and to modify the problem into the next form: $k(t) \to max$ 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 [207] and analogs of Fibonacci series (**P5**) could be placed [205, 206]. The stochastic Fibonacci model for the evolution of populations whose members undergo an immaturity phase could be presented. Let $\xi_i(n), i \in N+, n \in N$ be independent identically distributed random variables of generating function:

$$f(z) = \sum_{k \in N} p_k z^k, 0 < m = f'(1) < \infty \tag{5}$$

Let $X(0)$ and $X(1)$ be independent random variables that are also independent of ξ. The stochastic Fibonacci model is defined as follows [205, p. 311]:

$$X(n+2) - X(n-1) = \begin{cases} \sum\limits_{l=1}^{X(n)} \xi_l(n), & \text{if } X(n) > 0, n \in N; \\ 0, & \text{if } X(n) = 0, n \in N. \end{cases} \tag{6}$$

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 modeling including stochastic processes and the Monte Carlo method [208, 209]. As an example, the next problem (**P6**) to prove ergodicity of the described stochastic process [207] could be formulated. The described process is considered as mean-square ergodic in the first moment if

$$\lim_{n \to \infty} \frac{\sum\limits_{0}^{n} X(n)}{n} = E[X(n)] \tag{7}$$

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

$$\lim_{n \to \infty} \frac{\sum\limits_{o}^{n}[(X(n+1) - E[X(n)])(X(n) - E[X(n)])]}{n} = \\ = E[(X(n+1) - E[X(n)])(X(n) - E[X(n)])] \tag{8}$$

Such modifications or “faceting” of the problem introduce gamification features into the learning process and facilitates the constructionist approach to learning. It is important to stress here, that this simply formulated example incorporates the main features of the model-centered scientific inquiry-based education as focus on model-based simulations and seamless approach to theoretical backgrounds. During the design of the artifact in the form of a simulation model, students are forced to use the 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 42 presents the summary of the presented DP.

4.4 Domain model of the technological content

Technological Content domain model is of primary importance. The reason for such attitude is twofold.

(1) First, technology plays an important role for providing a platform for experiments with computers, including modelling, simulation and parallelization topics;
(2) next, the technology itself is a topic to be included in to the curriculum, especially for studying parallelization.

Technological domain consists of several major parts like:

(1) hardware platforms for computations;
(2) software tools for implementation of algorithms;
(3) software engineering topics and technologies.

Hardware platforms for calculations include various computational platforms. Here the most important are the big data processing feature and parallelization enabling features:

(1) the type of the processing unit (usually Complex Instruction Set Computing (CISC));
(2) the type of the computational architecture (as related to the topic of parallelization);
(3) Details of various processing architectures: SISD, SIMD, MISD, MIMD;
(4) Details of MIMD architectures: Uniform Memory Access (UMA), NUMA solutions.
(5) Details of hybrid HPCC architecture.

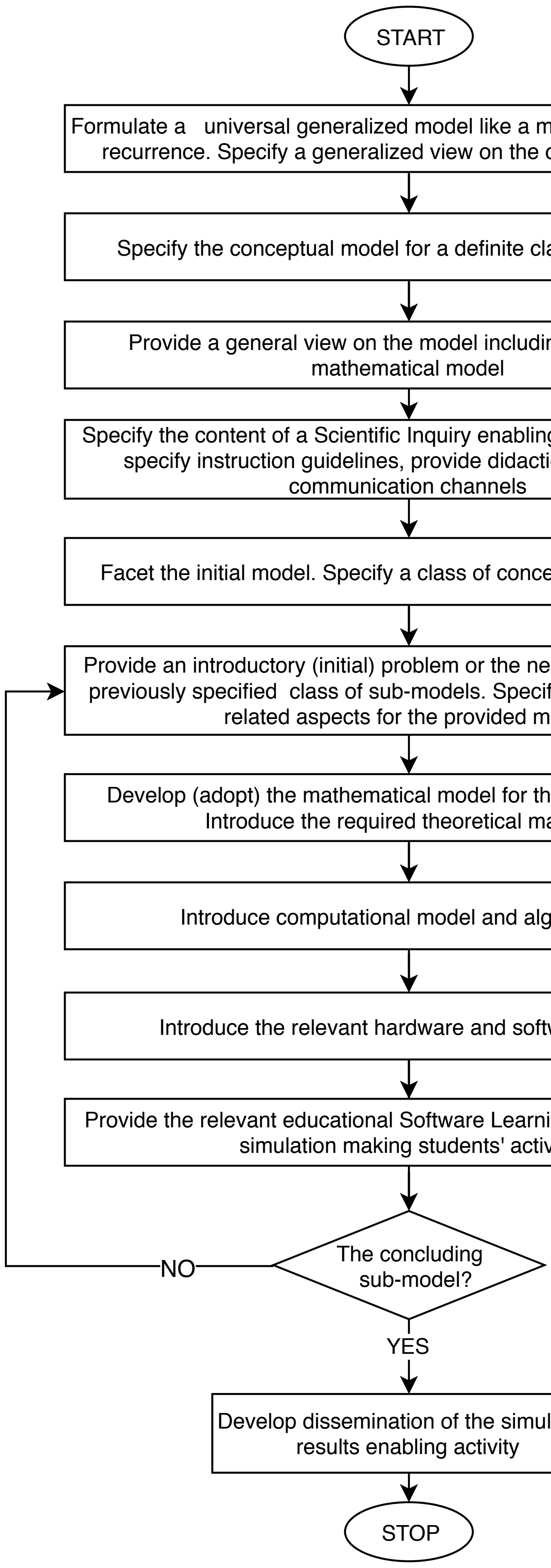

START
Formulate a universal generalized model like a model of a stochastic recurrence. Specify a generalized view on the class of problems
Specify the conceptual model for a definite class of problems
Provide a general view on the model including generalized mathematical model
Specify the content of a Scientific Inquiry enabling educational tools: specify instruction guidelines, provide didactic tools, specify communication channels
Facet the initial model. Specify a class of conceptual sub-models
Provide an introductory (initial) problem or the next model using the previously specified class of sub-models. Specify Scientific Inquiry related aspects for the provided model
Develop (adopt) the mathematical model for the initial problem. Introduce the required theoretical material.
Introduce computational model and algorithms
Introduce the relevant hardware and software topics
Provide the relevant educational Software Learning Object enabling simulation making students' activity
The concluding sub-model?
NO
YES
Develop dissemination of the simulation results enabling activity
STOP

Figure 42: Summary of the design principles for the model based SI centered LR

Software tools for implementation of algorithms include various parallelization specific software tools:
(1) shared memory parallelization tools;
(2) distributed memory parallelization tools;
(3) platforms specific operation systems;
(4) platforms specific command languages;
(5) SC specific tools: randomization tools, matrix and vector computations; big-data visualization tools;
(6) Platform (computational, visualization, simulation) specific programming languages and tools.

Software engineering topics and technologies include software development methods. This is especially relevant for implicit parallelization techniques and techniques based on the functional approach to programming. Such methods include:
(1) model-based approach to software engineering (parallelization);
(2) software engineering approaches to scientific computing methods and algorithms;
(3) software engineering approaches to big data processing including data storage and visualization aspects.

4.5 Implementation of the models

4.5.1 Introduction

The following examples are more complex and use queueing networks and stochastics. Basic principles of queueing networks [210–213] are easy to understand and could become a foundation for various problems and simulation experiments.

4.5.2 Python for model-based and simulation-centered education

The system of queues in series provides us with the kernel for the development of a relevant model-centered simulation framework, which could become as a basis for an integrated instructional unit. Such framework includes basic meanings, such as: probability and distributions, basic of queueing and other theoretical topics, as well as more complex theoretical results and methods [4]. The basic meanings include:
(R) Randomness:

    (1) random numbers;
    (2) random numbers' distributions;
    (3) generators of random numbers;
    (4) Central Limit Theorem.
(P) Python programming constructions:
    (1) decorators;
    (2) coroutines;
    (3) yield expressions.
(T) Results that are more complex include theoretical facts such as:
    (1) queues series specifications and parameters like the sojourn time of the customer;
    (2) recurrent equation for calculating the sojourn time;
    (3) stochastic simulation methods and multiprocessing techniques.

In Figure 43 (reprinted from [4, p. 45]) a general scheme of the described educational framework is provided. 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 experimentations 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.

That will give the whole picture of the scope of the general scientific research to the learner (see Figure 44, reprinted from [4, p. 45]). 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 SC.

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 modeling and simulation.

(1) 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 *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.
(2) The next model calculates the sojourn time of the customer and is based on stochastic simulations [6]. The model does not use multiprocessing directly. It emulates multiprocessing by using Python yield expressions.
(3) The last model presented here uses the Python MPI mpi4py module. From now, MPI techniques for statistical modeling could be used, and this could enhance Monte Carlo simulations by implementing additional trials [4].

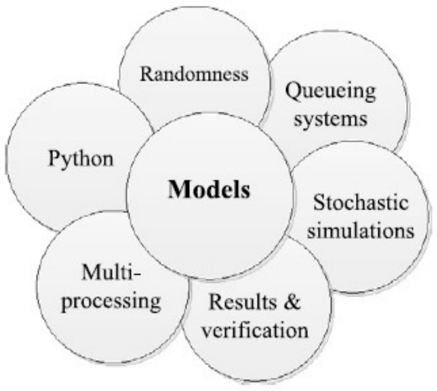


Figure 43: Integrated model-based framework. Reprinted from [4, p. 45]

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. An overview of the educational aspects is presented in Figure 45.

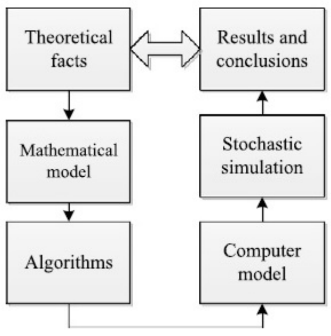


Figure 44: The model of the Educational Research. Reprinted from [4, p. 45]

4.5.3 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 [5]. The framework provides a set of programming models. 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. That 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. Figure 46 (reprinted from [5, p. 2]) presents the model-centered approach to the introduction into SC and parallel programming. 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. An overview of the educational aspects is presented in Figure 47.

4.5.4 Further improvements

The following possible improvements could be implemented.

(m) Models:

(m1) wider class of queueing networks could be involved in the modeling

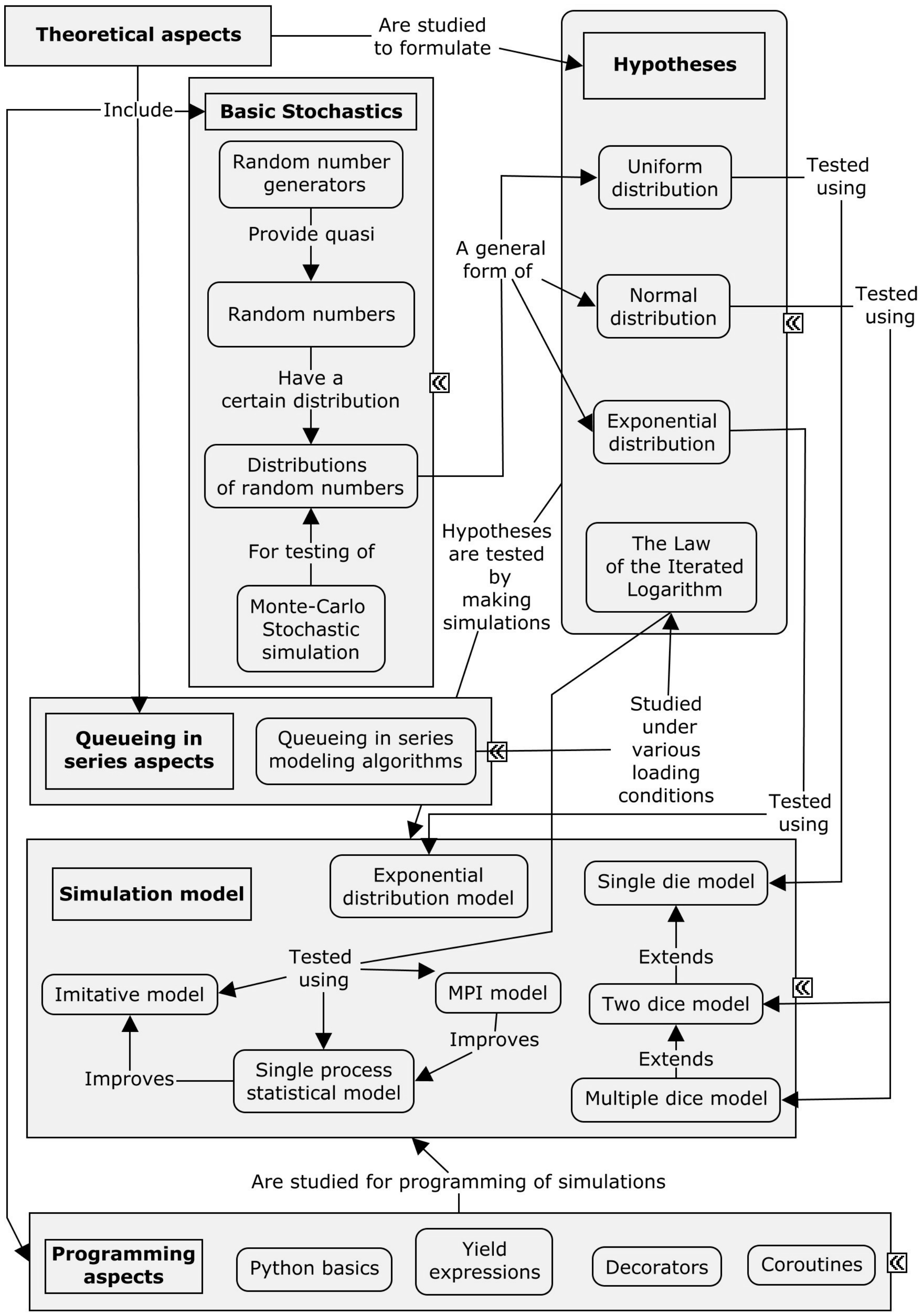


Figure 45: Educational aspects of the Python based Scientific Computing

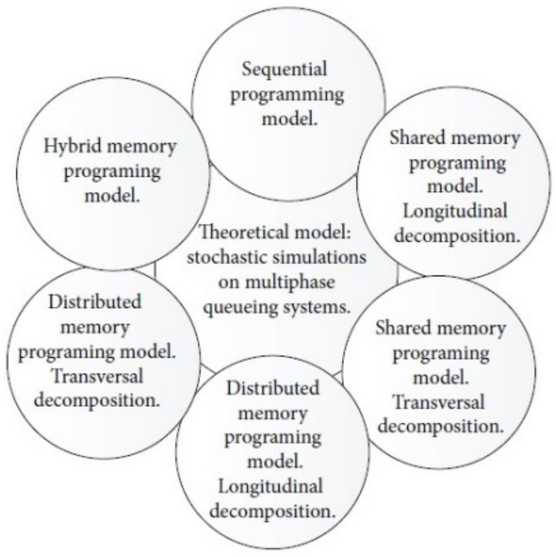

Sequential programming model.
Shared memory programing model. Longitudinal decomposition.
Shared memory programing model. Transversal decomposition.
Distributed memory programing model. Longitudinal decomposition.
Distributed memory programing model. Transversal decomposition.
Hybrid memory programing model.
Theoretical model: stochastic simulations on multiphase queueing systems.

Figure 46: Model-based framework for teaching parallelization. Reprinted from [5, p. 2]

process;

(m2) additional types of models, as imitational models could be developed;

(m3) additional types of computational features could be implemented;

(m4) theoretical backgrounds for computational techniques could be revised and improved.

(t) Technologies:

(t1) specifications for appropriate learning objects could be revised;

(t2) requirements for the integration and unification of an educational environment could be provided.

(i) Instruction:

(i1) theoretical foundations for instructional design should be adapted;

(i2) practical recommendations for instructional design should be provided.

(e) Educational policy:

(e1) additional studies for SC integration are needed;

(e2) additional studies for the implementation of multi-disciplinary curriculum are needed.

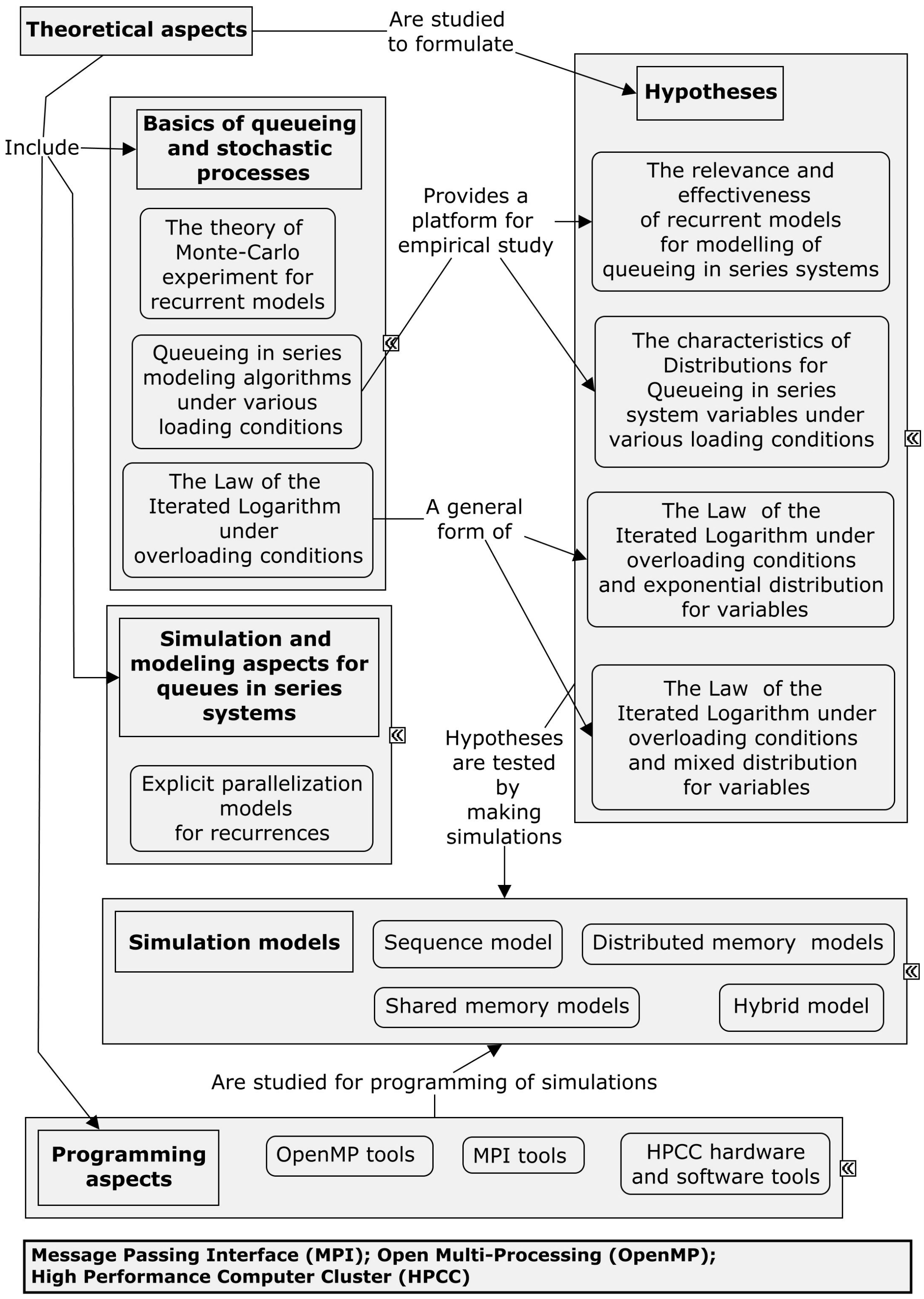

Theoretical aspects
Are studied to formulate
Hypotheses
Include
Basics of queueing and stochastic processes
The theory of Monte-Carlo experiment for recurrent models
Queueing in series modeling algorithms under various loading conditions
The Law of the Iterated Logarithm under overloading conditions
Provides a platform for empirical study
The relevance and effectiveness of recurrent models for modelling of queueing in series systems
The characteristics of Distributions for Queueing in series system variables under various loading conditions
A general form of
The Law of the Iterated Logarithm under overloading conditions and exponential distribution for variables
The Law of the Iterated Logarithm under overloading conditions and mixed distribution for variables
Simulation and modeling aspects for queues in series systems
Explicit parallelization models for recurrences
Hypotheses are tested by making simulations
Simulation models
Sequence model
Distributed memory models
Shared memory models
Hybrid model
Are studied for programming of simulations
Programming aspects
OpenMP tools
MPI tools
HPCC hardware and software tools
Message Passing Interface (MPI); Open Multi-Processing (OpenMP);
High Performance Computer Cluster (HPCC)

Figure 47: Educational aspects of the C and HPCC based Scientific Computing

4.6 Generalized feature model

Summarizing, the feature model could be implemented in the form of a conceptual map diagram. Implemented reasoning schemes provide an outline of concepts and meanings used during the assertions:

1. Scientific computing education:
   (a) Revision of definitions (the main focus is on SI and research, not on content);
   (b) Model centered approach;
   (c) From conceptual model to computer model (could be in the form of software – possible approach software as a learning object);
   (d) Teaching how to do research;
   (e) Teaching SI by how to make models and conduct experiments with models;
   (f) Teaching of how to make model-based simulations;
   (g) Seamless approach to theoretical prerequisites;
   (h) DSR for improvement of models (as a teaching method);
   (i) Educational task: improvements of the model-based simulation (with the aim to find a solution to the scientific problem);
   (j) Students as "quasi" researchers;
   (k) Educators provide an artificial environment for this "primary" research;
   (l) Main focus on models and simulations (on artifacts);
   (m) Inquiry-based, constructionist, and research-based education;
   (n) DSR-based education.
2. Interdisciplinary education:
   (a) Focus on research in the field of interest;
   (b) Doing research;
   (c) Solving scientific problems by SI;
   (d) Making scientific model-based simulation.
3. The research context:
   (a) Practical examples of implementations;
   (b) DSR formalisms;
   (c) Evaluation;
   (d) Dissemination;
   (e) Published papers.

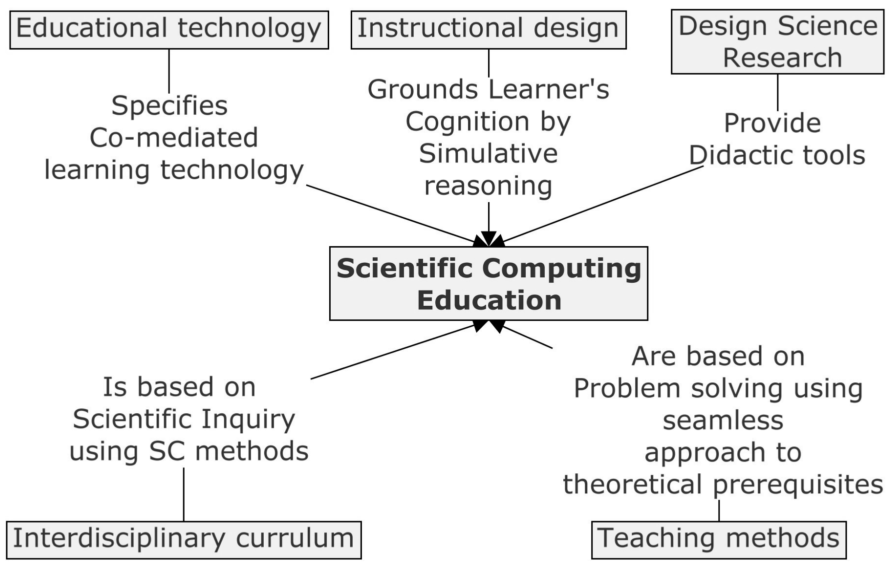
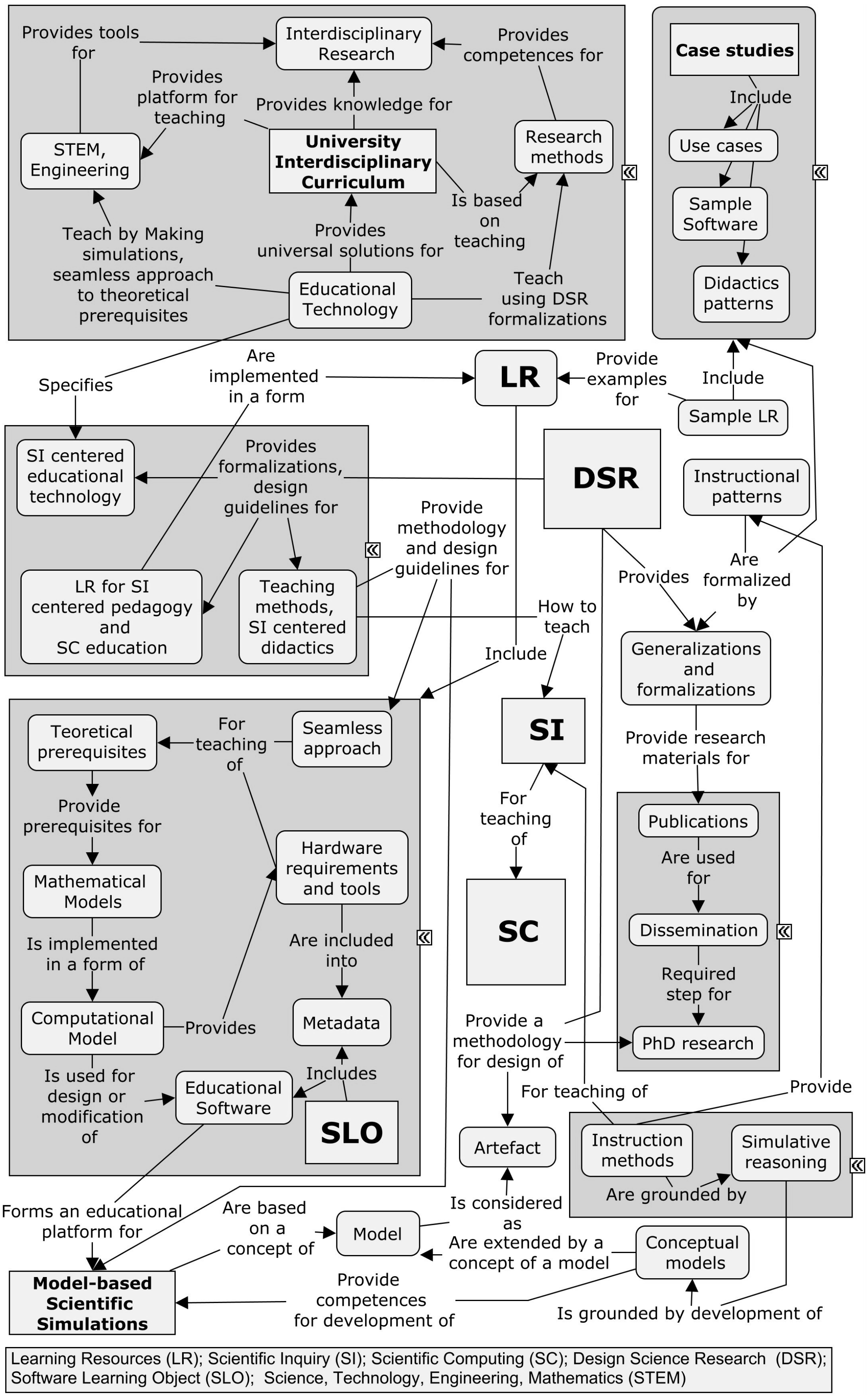

Educational technology
Instructional design
Design Science Research
Specifies Co-mediated learning technology
Grounds Learner's Cognition by Simulative reasoning
Provide Didactic tools
Scientific Computing Education
Is based on Scientific Inquiry using SC methods
Are based on Problem solving using seamless approach to theoretical prerequisites
Interdisciplinary currulum
Teaching methods

Figure 48: Generalized model of the Sceintific Computing Education

The generalized model of SI centered SC education is presented in Figure 48. The concept map of the related features is presented in Figure 49. The list of concepts covered in the research include:

1. Cognitive artifact;
2. Case Studies;
3. Computational Model;
4. Conceptual Model;
5. Design Science Research;
6. Dissemination;
7. Educational technology;
8. Evaluation of models;
9. Formalization;
10. Computer Hardware;
11. Instructional Design;
12. Instructional Patterns;
13. Interdisciplinary University Research;
14. Learning Object;
15. Mathematical Model;
16. Model (in general);
17. Model-based Scientific Simulations;
18. Pedagogical Patterns;
19. PhD Research;



Interdisciplinary Research
Provides tools for
Provides competences for
Provides platform for teaching
Provides knowledge for
STEM, Engineering
University Interdisciplinary Curriculum
Research methods
Is based on teaching
Teach by Making simulations, seamless approach to theoretical prerequisites
Provides universal solutions for
Educational Technology
Teach using DSR formalizations
Case studies
Include
Use cases
Sample Software
Didactics patterns
Specifies
Are implemented in a form
LR
Provide examples for
Include
Sample LR
SI centered educational technology
Provides formalizations, design guidelines for
DSR
Instructional patterns
LR for SI centered pedagogy and SC education
Teaching methods, SI centered didactics
Provide methodology and design guidelines for
Provides
Are formalized by
How to teach
Include
Generalizations and formalizations
Teoretical prerequisites
For teaching of
Seamless approach
SI
Provide research materials for
Provide prerequisites for
For teaching of
Publications
Mathematical Models
Hardware requirements and tools
Are used for
SC
Dissemination
Is implemented in a form of
Are included into
Required step for
Computational Model
Provides
Metadata
Provide a methodology for design of
PhD research
Is used for design or modification of
Educational Software
Includes
SLO
For teaching of
Provide
Artefact
Instruction methods
Simulative reasoning
Are grounded by
Forms an educational platform for
Are based on a concept of
Model
Is considered as
Are extended by a concept of a model
Conceptual models
Model-based Scientific Simulations
Provide competences for development of
Is grounded by development of
Learning Resources (LR); Scientific Inquiry (SI); Scientific Computing (SC); Design Science Research (DSR); Software Learning Object (SLO); Science, Technology, Engineering, Mathematics (STEM)

Figure 49: Generalized feature model in the form of a concept map

20. Publications;
21. Sample Software;
22. Scientific inquiry;
23. Simulational Reasoning;
24. Teaching Research methods;
25. Teaching Interdisciplinary Curriculum;
26. Teaching Methods;
27. Teaching Scientific Computing;
28. Teaching STEM, Engineering;
29. Theoretical prerequisites.

4.7 Conclusions

The section provides the detailed analysis of the SCE focusing on PC, EC, and TC domains. Analyzing PC, educational technology aspects, instructional design aspects and didactic aspects of PC domain were studied. As the result of the study DSR based educational methods are provided. Analyzing EC, the main DP of model-based SI centered introductory content are provided. As the summarizing result of this study – the generalized feature model for SI centered SCE was implemented in the form of a concept map.

5 Development of computational models for teaching of parallelization

5.1 Introduction

5.1.1 Parallelism and parallelization: Advantages and disadvantages

Why is parallelism needed and what are advantages and disadvantages of parallelism? First, many real-world domains are parallel by their nature. For example, many scientific problems, as the one which is mentioned above, business and numeric applications model a parallel world, although the classical approach to programming requires to sequentialise solutions to such problems. Next, parallel computers are more reliable, as even in case of failures, computations are continued, while in a less efficient manner. Further, clearly, there is no limit to computational power. Powerful enough uniprocessor architectures require expensive engineering solutions to be implemented and are limited by a size and physical constraints. Multiprocessor solutions could implement smaller computational devices, therefore determining more effective and at the same time cost-effective solutions. On the other hand, there are some problems, arising with the parallel computations. Skillicorn stresses that parallelism [44, p. 5]: "... introduces additional degrees of freedom into the space of programs, and into the space of architectures and machines." Solutions and algorithms become more difficult in the implementation as "... humans consciousness appears to be basically sequential ... The challenge is to provide suitable abstractions that either match our sequential style of thinking, or make use of other parts of our brain which are parallel. For example, the activity lights on the front panels of several commercial multiprocessors allow us to capture the behavior of the running system using the parallelism of the human object recognition system."

5.1.2 Computer architectures

The current state of parallelism could be described as follows [44]: "... architectures specific programming of parallel machines is rapidly maturing, and the tools used for it are becoming sophisticated; while architecture-independent programming of parallel machines is just beginning to develop as a plausible long-term direction". The two key elements of the conventional computational system are the processor and the memory [214]. Figure 50 (reprinted from [214, p. 1]) represents the memory-processor interconnection known as the Von Neumann model of computation. The classical taxonomy of parallel computers is presented in Fig-

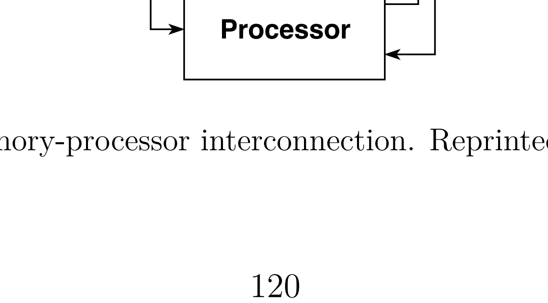


Figure 50: Memory-processor interconnection. Reprinted from [214, p. 1]

ure 51 (adapted from [215, p. 3]). The present-day taxonomy of parallel

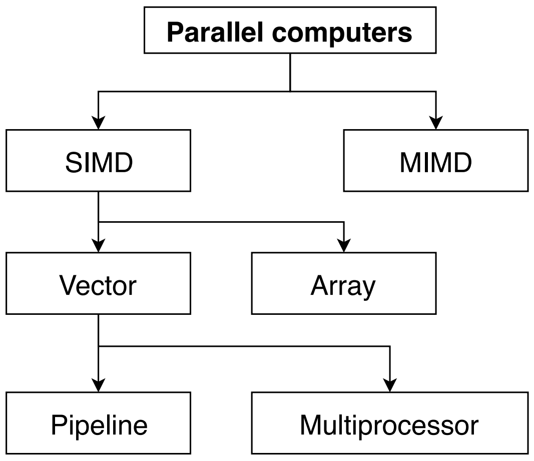


Figure 51: A simple taxonomy of parallel computers. Adapted from [215, p. 3]

architectures is presented in Figure 52 (reprinted from [214, p. 5]). This taxonomy also includes hybrid architectures as well. Modern classification

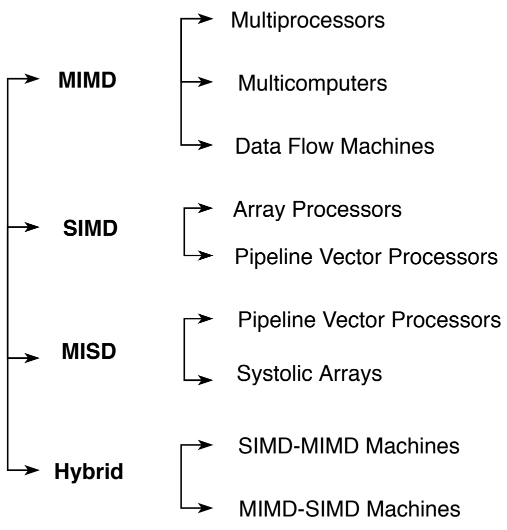


Figure 52: Taxonomy of parallel processing architectures. Reprinted from [214, p. 5]

of parallel computers include: SISD (Single Instruction, Single Data that is uniprocessors), SIMD (Single Instruction, Multiple Data), MISD (Multiple Instruction, Single Data) and MIMD (Multiple Instruction, Multiple Data), where MIMD is divided into two main classes, depending on the relationship of memory and processors: Shared-memory and Distributed-memory sub-classes. Shared-memory, or tightly-coupled, MIMD architectures allow any processor to access any memory module through a central switch.

Distributed-memory, or loosely-coupled, MIMD architectures connect individual processors with their own memory modules and implement access to remote memory by messages passed among processors [44]. SISD computers have one Central Processing Unit (CPU) that executes one instruction at a time (single instruction stream) and fetches or stores one item of data at a time (single data stream). The dominating concepts today are the SIMD and MIMD variants [216]. Figure 53 presents a general structure of an SISD architecture (reprinted from [214, p. 2]). SIMD machines have one Control Unit that executes a single instruction stream, but they have more than one Processing Element. The control unit generates the control signals for all of the processing elements, which execute the same operation on different data items (thus multiple data stream), meaning that they execute programs in a lock-step mode, in which each processing element has its own data stream. Figure 54 (reprinted from [214, p. 2]) presents a general view of an SIMD architecture [214]. MISD machines could execute several

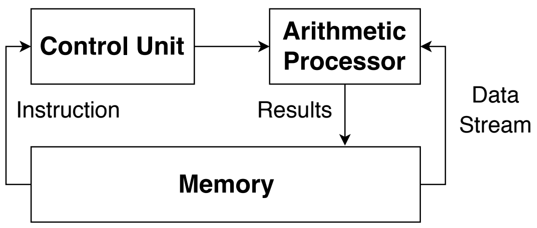


Figure 53: SISD architecture. Reprinted from [214, p. 2]

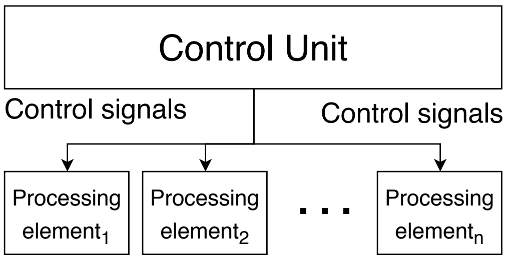


Figure 54: SIMD architecture. Reprinted from [214, p. 2]

different programs on the same data item. Figure 55 (reprinted from [214, p. 4]) represents the general structure of an MISD architecture and Figure 56 (reprinted from [214, p. 5]) represents the general structure of an MIMD architecture [214].

It is clear that each of architectures requires its own way for related soft-

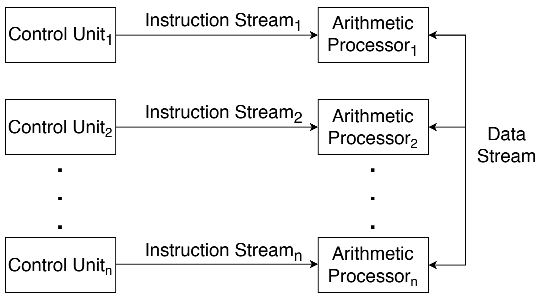


Figure 55: MISD architecture. Reprinted from [214, p. 4]

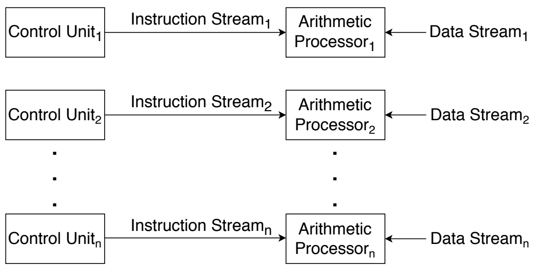


Figure 56: MIMD architecture. Reprinted from [214, p. 5]

ware development. Therefore, such an architecture-specific way of software development is dominating and programming models were successfully developed for each architecture class: lockstep execution for SIMD machines, test-and-set instructions for managing access in shared-memory machines, and message passing and channels for distributed-memory machines. Languages, algorithms, compiler technology, and in some cases whole application areas, have grown up around each architectural style [44].

MIMD architectures are of the main interest within the scope of our research. MIMD could be divided into two main groups: shared memory or uniform memory access system Uniform Memory Access (UMA) and distributed memory access systems or non-uniform memory access system Non Uniform Memory Access (NUMA) are presented in Figure 57 (reprinted from [214, p. 24]) and Figure 58 (reprinted from [214, p. 25]).

5.1.3 Hybrid parallel HPC Platforms

To improve performance, hybrid platforms which employ both shared and distributed memory architectures are implemented. The shared memory component usually is a shared memory multiprocessor socket or can be a shared memory machine and/or Graphics Processing Unit (GPU). The distributed memory component is the networking of multiple shared memory

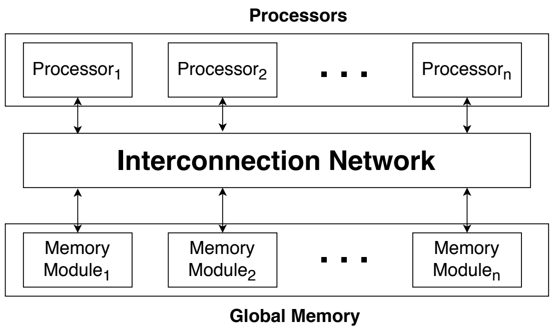


Figure 57: MIMD UMA architecture. Reprinted from [214, p. 24]

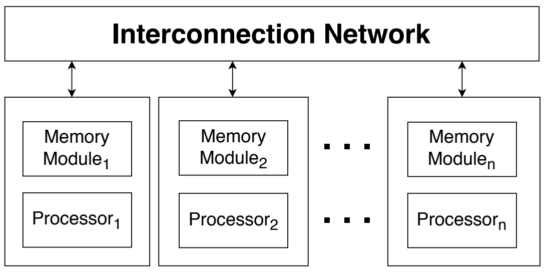


Figure 58: MIMD NUMA architecture. Reprinted from [214, p. 25]

units. Therefore, network communications of one or another type are required to move data from one unit to another. This could be a high-speed inter-computer network or internal inter-socket bus. “Current trends seem to indicate that this type of memory architecture will continue to prevail and increase at the high end of computing for the foreseeable future” [217]. Figure 59 (reprinted from [217, p. 17]) presents a schematic view of hybrid architecture. By using hybrid platforms it is possible to implement different explicit parallelization techniques from usual sequence programming to shared memory, distributed memory, and hybrid parallelization techniques.

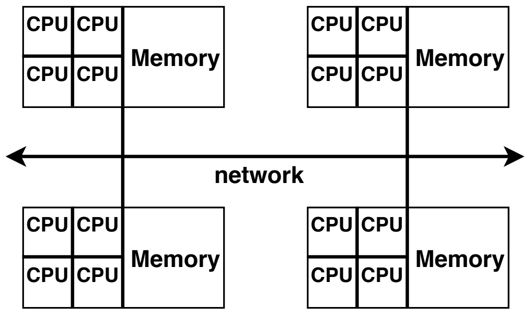


Figure 59: Hybrid distributed-shared memory platform. Adapted from [217, p. 17]

5.1.4 Model-centered approach to parallelization

From the point of view of an educator, this is also a challenge; such an approach also requires different and platform-specific learning objects to be developed, a different didactic approach to teaching to be implemented and different learning content to be prepared. Architecture-specific parallel computing is quite sophisticated by its engineering and used technology as well as teaching parallel computing is really a challenge for educators and learners [44, p. 7]: "The problem lies in the differences between the styles of parallel software that have grown up around each kind of architecture. ...At the heart of the problem is the tight connection between programming style and target architecture. As long as software contains embedded assumptions about properties of architectures, it is difficult to migrate it. This tight connection also makes software development difficult for programmers, since using a different style of architecture means learning a whole new style of writing programs and a new collection of programming idioms. The mismatch between parallel architectures and parallel software can be handled by the development of a model of parallel computation that is abstract enough to avoid this tight coupling. Such a model must conceal architectural details as they change while remaining sufficiently concrete that program efficiency is maintained. In essence, such a model describes an abstract machine, to which software development can be targeted, and which can be efficiently emulated on parallel architectures. A model then acts as the boundary between rapidly-changing architectures and long-lived software, decoupling software design issues from implementation issues." Figure 60 (adapted from [44, p. 8]) represents the role of a model of parallel computation.

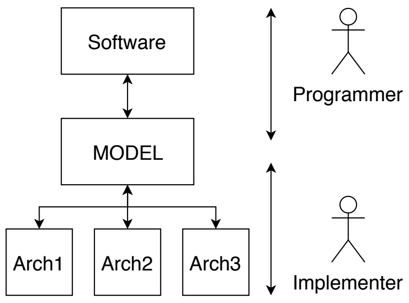


Figure 60: Role of a Model of Parallel Computation. Adapted from [44, p. 8]

As it could be seen from the presented figure, a suitable model should be independent of architecture; even the implemented solutions cannot be equally effective for all architectures. Another important feature is so-called “intellectual abstractness” [44]. Modern computer architectures are sophisticated enough, so appropriate programming solutions are complicated and too complex for the human mind. A suitable model must provide an abstraction for such details of parallelism as decomposition, mapping, communication, and synchronization. Other issues to be considered are a software development methodology, cost measures techniques, scale or granularity issues, and effectiveness of implementation issue.

5.1.5 Parallelization techniques

Software parallelization methods could be divided into two main groups: explicit parallelization and implicit parallelization methods. The main idea of implicit parallelization is to develop a universal and architecture independent model (mainly using functional programming techniques) and provide an automatic architecture dependent implementation for the developed model. That is why parallelization techniques are “hidden” in the model. Generally, for the implementation of implicit parallelization techniques, more sophisticated approaches for developing of the software models [44] are needed to be used. Explicit parallelization techniques use definite and software dependent implementation methods. To implement parallelization on shared memory systems, mainly OpenMP – Open Multi-Processing is used. OpenMP as an API supports multi-platform shared memory multiprocessing programming for some programming languages. MPI – Message Passing Interface is “de facto” standard for distributed memory systems.

There are several ways of implementing parallel programming using clusters of Symmetric Multiprocessing (SMP) nodes or other types of hybrid systems. Implementing models, one can use pure MPI or pure OMP techniques. Another solution is to use a hybrid approach combining MPI and OpenMP [218].

5.2 Implicit parallelization models

5.2.1 Implicit parallelization with skeletons

Skeletons give a possibility for the implementation of implicit and possibly automatic parallelization techniques [219, 220, 44, 221]. Map-reduce skeleton, for example, map-reduce skeleton in Haskell based Eden environment consists of Map, Parallel Map and Farm skeletons. Map functional is defined as follows:

$$\begin{aligned} &map :: (a \to b) \to [a] \to [b] \\ &mapf[\,] = [\,] \\ &mapf(x : xs) = (fx) : (mapfxs) \end{aligned}$$

Parallel Map (*parMap*) is defined next, where the Eden-specific type context (*Trans a*, *Trans b*) indicates that both types *a* and *b* must belong to the Eden Trans type class of transmissible values [221]:

$$parMap \;::\; (Trans\ a, Trans\ b) \;\Rightarrow\; (a \;\to\; b) \;\to\; [a] \;\to\; [b]$$

Parallel Map evaluation scheme is presented in Figure 61 (reprinted from [221, p. 5]).

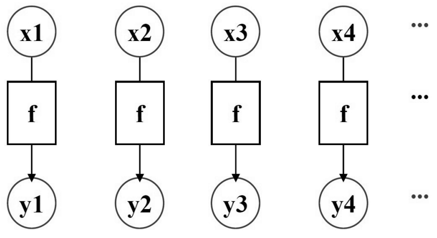


Figure 61: Basic map evaluation scheme. Reprinted from [221, p. 5]

The case when the number of list elements is higher than the number of available processors could be evaluated using Farm skeleton [221]:

$$\begin{aligned} farm \;&::\; (Trans\ a,\ Trans\ b) \;\Rightarrow \\ &([a] \;\to\; [[a]]) && \text{- distribute} \\ &\to ([[b]] \to [b]) && \text{- combine} \\ &\to (a \to b) \to [a] \to [b] && \text{- map interface} \\ &farm\ distribute\ combine\ f \\ &\quad = \; combine \;\circ\; (parMap(mapf)) \;\circ\; distribute \end{aligned}$$

Parallel Farm evaluation scheme is presented in Figure 62 (reprinted from [221, p. 6]).

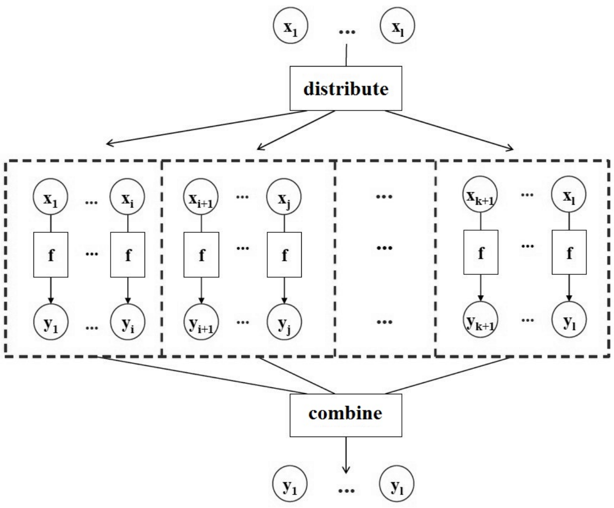


Figure 62: Parallel Farm evaluation scheme. Reprinted from [221, p. 6]

If a reduction is executed after the application of map Map-reduce skeleton could be used [221]:

$$
\begin{aligned}
&parMapRedr :: (Trans\ a,\ Trans\ b) \Rightarrow \\
&\qquad (b \rightarrow b \rightarrow b) \rightarrow b \rightarrow (a \rightarrow b) \rightarrow [a] \rightarrow b \\
&parMapRedr\ g\ e\ f \\
&\quad = if\ \ noPe\ == \ 1\ then\ mapRedr\ g\ e\ f\ xs\ else \\
&\qquad (foldr\ g\ e) \circ (parMap\ (mapRedr\ g\ e\ f)) \circ (splitIntoN\ noPe)
\end{aligned}
$$

Parallel Map-reduce evaluation scheme is presented in Figure 63 (reprinted from [221, p. 8]).

5.2.2 Implicit methods for the parallelization of linear stochastic recurrences

Generally, the linear stochastic recurrence could be written in the form:

$$
\begin{aligned}
x_0 &= B_0 \\
x_i &= (x_{i-1} \otimes A_i) \oplus B_i, 1 \leq i \leq n
\end{aligned}
$$

Here $A_i, 1 \leq i \leq n$ and $B_i, 0 \leq i \leq n$ are mutually independent and in pairs identically distributed random variables of general form.

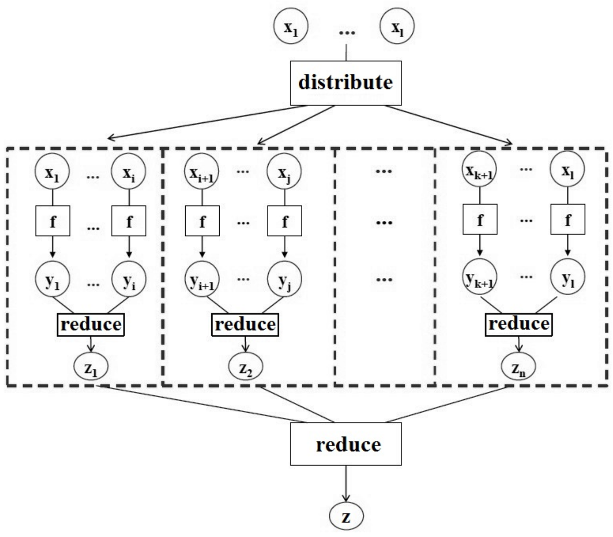


Figure 63: Parallel Map-reduce evaluation scheme. Reprinted from [221, p. 8]

There is no possibility to calculate the values of $x_i, 0 \leq i \leq n$ directly, as x_i has random values, which depend on the realization of random variables A and B. A possible solution is to use the Monte Carlo method to generate a set of sample values for every random variable and to use these the non-random values for calculations. Generally, if the underlying type for random variables A and B is defined as R_A and R_B accordingly, the underlying type for variable x is defined A, then using data type notation the solution could be written in the following form. First, a set of sample values for coefficients A and B should be calculated: $mc_A :: \tilde{R}_A \to R_A*$, where $(R_A)*$ is a notion for a list with underlying type R_A. Next, a list map functional could be used, enabling the recurrence formula of a form $\tau_R :: R_A \times R_A \to A$ to a set of realizations of random variables, and last, a list reduce functional $\backslash_\ominus$ could be applied (if needed) in order to obtain the result. Summarizing, there is a solution in the form of composition: $\backslash_\ominus \circ (\tau_R*) \circ (mc_A \nabla mc_A) :: \tilde{R}_A \times \tilde{R}_A \to A$, where we denote $(f*)$ as a list map functional for the function f, $\tilde{R}_A*$ and R_A* a list with underlying type $\tilde{R}_A$ and R_A, $(\circ)$ for composition of operations, and (∇) for polynomial compositions of functionals. As it is seen from section 5.2.1, such compositions allow us to use parallel map-reduce evaluation scheme.

At the next step, the τ_R operation for computing recurrences [44, p. 104] could be clarified. First, in the case of computing x_n value, $\tau_R = x \otimes /_{b0} \oplus y$

could be defined as:

$$x \otimes /_{b0} \oplus y = \begin{cases} b_0, & \text{if } \#x(= \#y) = 0; \\ b_0 \otimes \pi_1 A \oplus \pi_2 A, & \text{if } \#x(= \#y) \neq 0. \end{cases}$$

where

$$\begin{aligned} A &= \oslash/(x \curlyvee_{\otimes} y), \\ a \otimes b &= (a, b), \\ (a, b) \oslash (c, d) &= (a \otimes c, b \otimes c \oplus d), \\ \pi_1(a, b) &= a, \text{ and } \pi_2(a, b) = b \end{aligned}$$

Summarizing, the computational model could be presented as $\backslash_{\ominus} \ \circ \ \tau_R * \ \circ \ (mc_A \nabla mc_A) :: \tilde{R_A} \times \tilde{R_A} \to A$. The evaluation strategy for HPC clusters is the next:

(1) first, a parallel Map *parMap* for mc_A to distribute R_A* among evaluation nodes should be used:

$$\begin{aligned} (Trans\ \tilde{R}_A,\ Trans\ R_A) \ &\Rightarrow (\tilde{R}_A \to R_A*) \to (\tilde{R}_A \to R_A*) \\ &\to (\tilde{R}_A* \to R_A**) \nabla (\tilde{R}_A* \to R_A**) \end{aligned}$$

(2) next, the parallel MapRedr *parMapRedr* for $(\backslash_{\ominus}, e, \otimes/_{b0}\oplus)$, where we denote a unit for reduction functional as e, to evaluate recurrence in each node and make a reduction (if needed) could be used:

$$\begin{aligned} (Trans\ R_A*,\ Trans\ A) \ &\Rightarrow (R_A \to R_A \to R_A) \to (R_A \to R_A \to A) \\ &\to R_A* \to R_A* \to A \end{aligned}$$

Further development of the implicit model is out of the scope of this research, therefore explicit models will be considered in detail in the next section.

5.3 Explicit parallelization models

5.3.1 Introduction

Explicit parallelization models are based on distributed and shared parallelization tools such as MPI and OpenMP tools. First, the general description of non-linear models for stochastic recurrences will be provided.

(1) We define random vectors $\mathbf{a}_n = (a_n^1, a_n^2, \dots, a_n^p)$ and $\mathbf{b}_n = (b_n^1, b_n^2, \dots, b_n^p)$, $n = 1, 2, \dots$, where sequences of random variables a_n^i and b_n^i, $i = 1, 2, \dots, p$ are independent and consists of independent and identically distributed (within each sequence). We consider a_n^i, $i = 1, 2, \dots, p$; $n = 1, 2, \dots$, distributed identically as random variable a with distribution function $A(x) = \mathbf{P}(a < x)$ and b_n^i, $i = 1, 2, \dots, p$; $n = 1, 2, \dots$, distributed identically as random variable b with distribution function $B(x) = \mathbf{P}(b < x)$.
We define a first order non-linear stochastic recurrence relations for the random vector $\mathbf{x}_n = (x_n^1, x_n^2, \dots, x_n^p)$, $n = 1, 2, \dots$:

$$x_{j+1}^{i+1} = f(x_{j+1}^{i}, x_j^{i+1}, a_{j+1}^{i+1}, b_{j+1}^{i+1}),$$

$x_1^i = f'(a_1^i, b_1^i), x_j^1 = f''(a_j^1, b_j^1)$, $i = 1, 2, \dots, p$; $j = 1, 2, \dots$, where $f(x)$, $f'(x)$, $f''(x)$ define non-linear functions of random variables.

(2) The defined vector x_n is characterized by its components, and its length is p. Generally, it is not possible to find analytic solutions for random components of x_n vector. We could be interested in integral parameters of one or another form. For example, it could be interesting to find stochastic characteristics like distribution of $g(x_1^{i'}, x_2^{i'}, \dots, x_{j'}^{i'})$, where $g()$ is a non-linear function of random variables. To solve the problem indirectly, the Monte Carlo sampling method could be used.

(3) Monte Carlo sampling [222–224] is based on the next assumptions. Considering random variable a with distribution function $A(x) = \mathbf{P}(a < x)$ we define the $k-th$ moment (if exists) as $a_k = \mathbf{M}a^k = \int_0^\infty a^k dA(x)$. By $U_{[0,1]}$ we define the uniform distribution function, by A^- – the generalized inverse of A [223]:

$$A^-(u) = inf\{x; A(x) \geq u\}, u \in [0, 1].$$

The following lemma could be proved [223, p. 36]:

Lemma 5.1
If $U \sim U_{[0,1]}$, then random variable $A^-(U)$ has the distribution A.

Proof. For all $u \in [0, 1]$ and for all $x \in A^-([0, 1])$ the generalized inverse satisfies

$$A(A^-(u)) \geq u \texttt{ and } A^-(A(u)) \leq x.$$

Therefore,

$$\{(u,v) : A^{-}(u) \leq x\} = \{(u,v) : A(x) \geq u\}$$

and

$$\mathbf{P}(A^{-}(u) \leq x) = \mathbf{P}(U \leq A(x)) = A(x).$$

□

Using the method, in order to generate samples of random variable $a \sim A$, one should generate U according to $U_{[0,1]}$ and then make the transformation of the form $x = A^{-}(u)$. Using such transformation, a set of non-random samples of a random variable a: $(a', a'', \dots)$ could be generated.

(4) The next important point to consider is the strong law of large numbers [225]. The strong law of large numbers states that the sample average converges almost surely to the expected value [225]:

$$\bar{X}_n \xrightarrow{\text{a.s.}} \mu \qquad \text{when } n \to \infty.$$

That is,

$$\Pr\Big(\lim_{n\to\infty} \bar{X}_n = \mu\Big) = 1.$$

This means that as the number of trials n goes to infinity, the probability that the average of the observations is equal to the expected value will be equal to one.

(5) The previously considered Monte Carlo method and Strong Law of Large Numbers allow us to define a computational model for non-linear stochastic recurrences of a general form. Let us define random variables a and b with distribution functions $A(x)$ and $B(x)$ accordingly. We could define a non random matrix of Monte Carlo samples for variables a^i_j and b^i_j defined earlier, as $\mathbf{a}_n = (\mathbf{a}^1_n, \mathbf{a}^2_n, \dots, \mathbf{a}^p_n)$ and $\mathbf{b}_n = (\mathbf{b}^1_n, \mathbf{b}^2_n, \dots, \mathbf{b}^p_n)$, where $\mathbf{a}^i_j$ and $\mathbf{b}^i_j$ are by non-random sample vectors $\mathbf{a}^i_j = ((a^i_j)^1, (a^i_j)^2, \dots, (a^i_j)^m)$ and $\mathbf{b}^i_j = ((b^i_j)^1, (b^i_j)^2, \dots, (b^i_j)^m)$. Using these definitions the model for non-linear stochastic recurrences of a general form could be defined as follows. We define a

sample matrix $\mathbf{x}_n = (\mathbf{x}_n^1, \mathbf{x}_n^2, \ldots, \mathbf{x}_n^p)$, $n = 1, 2, \ldots$, where $\mathbf{x}_j^i = ((x_j^i)^1, (x_j^i)^2, \ldots, (x_j^i)^m)$ - sample vector, and

$$(x_{j+1}^{i+1})^k = f((x_{j+1}^i)^k, (x_j^{i+1})^k, (a_{j+1}^{i+1})^k, (b_{j+1}^{i+1})^k), \forall k \leq m \qquad (9)$$

where
$(x_1^i)^k = f'((a_1^i)^k, (b_1^i)^k)$; $(x_j^1)^k = f''((a_j^1)^k, (b_j^1)^k)$, $i = 1, 2, \ldots, p$; $j = 1, 2, \ldots, \forall k \leq m$, where $f(x)$, $f'(x)$, $f''(x)$ define non-linear component functions of non-random samples.

Consider the case, then $p \gg 0$, $n \gg 0$, $m \gg 0$. A big amount of computational resources are needed for modelling the above recurrences, thus parallelization techniques are needed to achieve sufficient values for p, n, m parameters.

5.3.2 Sequence model

The Sequence model is rather straightforward. No parallelization is implemented. Programming construction of inline for cycles is used. Algorithm (in ALGOL type pseudo-code) is presented in Algorithm 5.1.

Algorithm 5.1 Sequence programming model

```
global $p, n, m$

procedure CalculateSample($seed, \lambda$)
 $s \leftarrow randomSample \leftarrow (seed, \lambda)$
 return $(s)$

procedure CalculateRecurrence($i, j, k$)
 comment: Init random generator: Seed, DistributionParameter
 $a \leftarrow CalculateSample(Seed, DistributionParameter)$;
 $b \leftarrow CalculateSample(Seed, DistributionParameter)$;
 $(x_j^i)^k \leftarrow f((x_j^i)^k, a, b)$

main
 for $k \leftarrow 1$ to $m$
   do { $(x_j^i)^k \leftarrow initialization$
        for $j \leftarrow 1$ to $n$
          do for $i \leftarrow 1$ to $p$
        { CalculateRecurrence($i, j, k$)
 output $((x_j^i)^k)$
```

5.3.3 Distributed memory model

Distributed memory model uses MPI tools for parallelization. Algorithm in the form of pseudo-code is presented in Algorithm 5.2.

Algorithm 5.2 Distributed memory model

```
global $p, n, m$

procedure ProcessRecurrence($np, pid, seed, \lambda$)
 for $k \leftarrow 1$ to $m/np$
   do
     $(x_j^i)^k \leftarrow initialization$
     for $j \leftarrow 1$ to $n$
     for $i \leftarrow 1$ to $p$
       do
         $a \leftarrow randomSample \leftarrow (\lambda, seed)$
         $b \leftarrow randomSmple \leftarrow (\lambda, seed)$
         $(x_j^i)^k \leftarrow f((x_j^i)^k, a, b)$
 comment: Gathering results from each MPI
 $overallResultsArray \leftarrow (x_j^i)^k$
 if $pid == InitialProcessNr$
   then output ($overallResultsArray$)

main
 $MPI \leftarrow initMPI$
 $Random \leftarrow initRandomGenerator$
 $a, b, x \leftarrow initVariables$
 comment: Call ProcessRecurrence with parameters
 $PAR = \{NumberOfProcesses, ProcessId, Seed, StochasticsParameter\}$
   do ProcessRecurrence($PAR$)
 $MPI \leftarrow finalizeMPI$
```

5.3.4 Shared memory model

Shared memory model uses OpenMP tools for parallelization. Algorithm in the form of pseudo-code is presented in Algorithm 5.3.

5.3.5 Hybrid model

Hybrid programming model uses MPI tools as well as OpenMP tools for parallelization. The algorithm in the form of pseudo-code is presented in Algorithm 5.4.

Algorithm 5.3 Shared memory model

```
global $p, n, m$

main
  $a, b, x \leftarrow initVariables$
  $OpenMP \leftarrow initOpenMP(numOfThreads) \leftarrow privateVariables(thid, i, j, k)$
  $OpenMP \leftarrow sharedVaribales(overallResultsArray)$
  $Random \leftarrow initRandomGenerator$
  comment: Starting parallel region. Fork a team of threads
  {
    do for $k \leftarrow 1$ to $m/numOfThreads$
    do {
      $(x_j^i)^k \leftarrow initialization$
      for $j \leftarrow 1$ to $n$
      for $i \leftarrow 1$ to $p$
      do {
        $a \leftarrow randomSample \leftarrow (\lambda, seed)$
        $b \leftarrow randomSmple \leftarrow (\lambda, seed)$
        $(x_j^i)^k \leftarrow f((x_j^i)^k, a, b)$
      }
    }
  }
  comment: gather results from each Thread
  $overallResultsArray \leftarrow (x_j^i)^k$
  comment: Ending parallel region
  output $(overallResultsArray)$
```

Algorithm 5.4 Hybrid programming model

```
global $p, n, m$

procedure ProcessRecurrence($np, pid, seed, \lambda$)
 ${(x^i_j)}^k, a, b \leftarrow initValues$
 repeat
    do if $pid! = startPidID$
      then $MPI \leftarrow dataFromPreviousMPINode$
    $OpenMP \leftarrow initOpenMP(numOfThreads) \leftarrow privateVariables(thid, i, j, k)$
    $OpenMP \leftarrow sharedVaribales(overallResultsArray)$
    comment: Starting parallel region. Fork a team of threads
      do for $k \leftarrow 1$ to $m/(numOfThreads * np)$
      do
          ${(x^i_j)}^k \leftarrow initialization$
          $Random \leftarrow initRandomGenerator$
          for $j \leftarrow 1$ to $n$
          for $i \leftarrow 1$ to $p$
          do
              $a \leftarrow randomSample \leftarrow (\lambda, seed)$
              $b \leftarrow randomSmple \leftarrow (\lambda, seed)$
              ${(x^i_j)}^k \leftarrow f({(x^i_j)}^k, a, b)$
    comment: gather results from each Thread
    $overallResultsArray \leftarrow {(x^i_j)}^k$
    comment: Ending parallel region
 until $pid! = lastPidID$
 comment: Distributing MPI results
 if $pid! = lastPidID$
   then $distributrData \leftarrow MPI$
 if $pid == InitialProcessNr$
   then
      do $overallResultsArray \leftarrow {(x^i_j)}^k$
      output ($overallResultsArray$)

main
 $MPI \leftarrow initMPI$
 $Random \leftarrow initRandomGenerator$
 $a, b, x \leftarrow initVariables$
 comment: Call ProcessRecurrence with parameters
 $PAR = \{NumberOfProcesses, ProcessId, Seed, StochasticsParameter\}$
   do ProcessRecurrence($PAR$)
 $MPI \leftarrow finalizeMPI$
```

5.4 Conclusions

The meta-models and algorithms presented in this section provide a comprehensive background for developing of application-specific models, which will be presented in the next sections.

(1) First, such computational platforms as HPC cluster allow different algorithms and programming techniques to be implemented during parallelization. These include shared memory, distributed memory, and hybrid memory solutions. For application in the study, HPC 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' motivations and implementing the constructionist approach to learning.

(2) Next, stochastic recurrences of a general type provide a suitable model for computations. This could be used to model definite application areas. Incorporating stochastic into the model, one could test multidimensional parallelization techniques, using the Monte Carlo modeling method.

(3) Finally, the model-centered approach could be used for design and further development of SLOs for teaching parallelization. This approach is suitable both for implicit and explicit parallelization methods.

6 Experimental research: Educational implementations

6.1 Introduction

This section presents an approach to designing software learning objects based on the real-world scientific problem solved by the author. First, the description of the real scientific task and the method for the simulation is presented. Later, two case studies (practical applications), based on the described theoretical results are presented [4, 5]. Both applications require the same theoretical prerequisites from such fields as probability and operational research. The first application uses Python programming language for software development. It could be employed as an educational tool within a scientific inquiry-based introductory course for statistical methods or stochastic. 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 a SC course. These models were positively evaluated by reviewers and published in Scientific Programming Journal (the journal is cited in Clarivate Analytics Master journals list `http://mjl.clarivate.com/cgi-bin/jrnlst/jlresults.cgi?PC=D&ISSN=1058-9244`).

6.2 Experiment planning

The general view of the computational experiment is presented in Figure 64 and includes the System itself, Problem statement, Mathematical model of the system, Computational model, and Simulation experiment. In more detail:

(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 (see Figure 65) is hardware and software dependent. It provides a basis for designing SLOs in the following manner:

(CM) COMPUTATIONAL MODEL CM: Stochastic Recurrence model of Queues in Series

(SM) A SET OF SUB-MODELS: Model CM is divided into sub-models: {SCM1, SCM2, SCM3, ...}

(L) A SET OF LEARNING RESOURCES: A set of sub-models is projected to a set of LR in the form of SLOs.

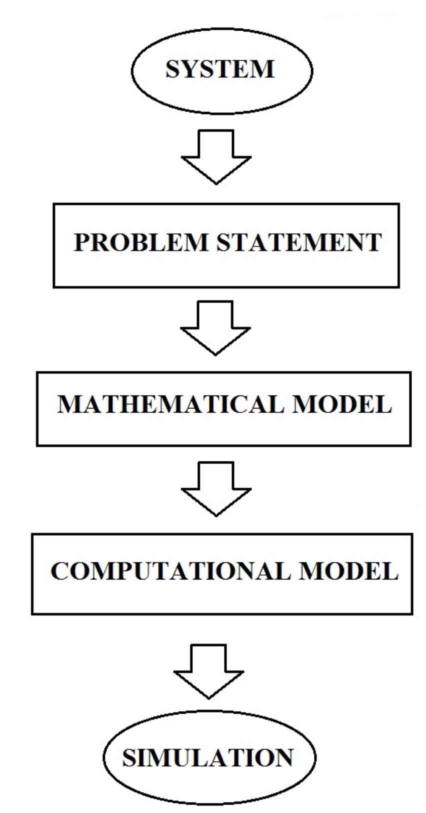


Figure 64: Computational experiment planning process

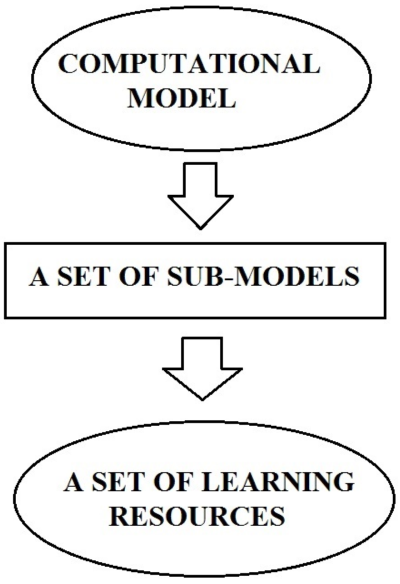


Figure 65: Learning resources design process

6.3 Experiment design

6.3.1 Statement of the problem

The object of this research in the sphere of queueing theory is the law of the iterated logarithm under the conditions of heavy traffic and with load factor more than *one* for queues in series. In the paper [6], 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. 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.

We investigate here a k-phase system of queues in series (i.e., after a customer has been served in the j-th phase of the queue, he goes to the $j+1$-st phase of the queue, and, after the customer has been served in the k-th phase of the queue, he leaves the queue). Let us denote by r_n^j the time of arrival of the n-th customer to the node number j; by $\tau_n^j = r_{n+1}^j - r_n^j$ – an interarrival time to the node number $j, j \geq 1$ (if $j = 1, \tau_n^1 = \tau_n$ means interarrival time to the system); by s_n^j – the service time of the n-th customer in the j-th phase; let interarrival times $\{\tau_n\}$ at queues in series and service times $\{s_n^j\}$ in each phase of the queue for $j = 1, 2, \cdots, k$ be mutually independent identically distributed random variables. Let denote E as the first moment for any random variable (if exists).

Next, denote by v_n^j the waiting time of the n-th customer in the j-th phase of the queue; $w_n^j = \sum_{i=1}^{j}(v_n^i + s_n^i)$ stands for the sojourn time of the n-th customer (time, which the n-th customer spent in the queueing system until the j-th phase), $j = 1, 2, \ldots, k$. By t_n^j we denote the sojourn time of the customer in the node number j, that is $t_n^j = v_n^j + s_n^j$. Vectors $\hat{t}_n = \{t_n^j\}$, $\hat{s}_n = \{s_n^j\}$, and $\hat{v}_n = \{v_n^j\}$ will denote the sojourn, servicing and waiting time for each node accordingly.

We consider such a modified system of queues in series in which $s_n^j = 0,\ j = 1, 2, \ldots, k,\ n \geq k$. Thus, further we can investigate only the modified system of queues in series and admit that $n \geq k$.

When $j = 1, 2, \ldots, k$, let

$$\delta_{j,n} = \begin{cases} s^j_{n-(j-1)} - \tau_n, & \text{if } n \geq k \\ 0, & \text{if } n < k. \end{cases}$$

Let us define $\alpha_j = E\delta_{j,n}$, $\alpha_0 \equiv 0$, $D\tau_n = \sigma_0^2$, $Ds^j_n = \sigma_j^2$, $\tilde{\sigma}_j^2 = \sigma_0^2 + \sigma_j^2$, $s^0_n = \tau_n$, $j = 1, 2, \ldots, k$, $\hat{\delta}_n = \max\limits_{1 \leq j \leq k} \max\limits_{0 \leq l \leq 2n} |\delta_{j,l}|$, $[x]$ as the integer part of number x.

We admit that the following conditions are fulfilled:

there exists a constant $\gamma > 0$ such that

$$\sup_{n \geq 1} E|s^j_n|^{4+\gamma} < \infty, \; j = 0, 1, 2, \ldots, k \tag{10}$$

and

$$\alpha_k > \alpha_{k-1} > \cdots > \alpha_1 > 0. \tag{11}$$

At first we investigate LIL for the sojourn time of a customer in the system of queues in series. The following theorem could be proved [6]:

Theorem 6.1 (The law of the iterated logarithm for the sojourn time of a customer [6])
If conditions (10) and (11) are fulfilled, then

$$\boldsymbol{P}\left(\overline{\lim_{n\to\infty}} \frac{w^j_n - \alpha_j \cdot n}{\tilde{\sigma}_j \cdot a(n)} = 1\right) = \boldsymbol{P}\left(\underline{\lim_{n\to\infty}} \frac{w^j_n - \alpha_j \cdot n}{\tilde{\sigma}_j \cdot a(n)} = -1\right) = 1,$$

$j = 1, 2, \ldots, k$ *and* $a(n) = \sqrt{2n \ln \ln n}$. □

6.3.2 Algorithmic solution

This section provides necessary theoretical background for creating of the computational model for modeling of the system of queueies in series. Such models allow us to study the behavior of the system of queueies in series in various (including overloading) conditions. The first result (Theorem 6.2) provides 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. The next result (Theorem 6.3) provides a solution for the development of more efficient (linear) computational model (in case of overloading). This result is generalized as Proposition 6.5.

Let us, as it was described earlier, define $n \in \mathbb{N}$ as the number of the arrival; independent random variables $s_n^j, j \geq 0$ as interrarival times (case $j = 0$) and service times, and be identically distributed with distribution functions identical to the distribution function of random variable s^j, that is $s^j(x) = P(s^j < x)$. Let us accordingly denote moments for variables s^j as

$$M(s^j)^k = \int_0^\infty (s^j)^k ds(x).$$

Let us denote as $\hat{t}(x)$ a vector of distributions of the sojourn time of the customer in each servicing node, that is $\hat{t}_n(x) = \{t_n^j(x)\}$, and $s_n^j(x) = P(s_n^j < x), j \geq 0$, $\hat{s}_n(x) = \{s_n^j(x)\}$, and $v_n^j(x) = P(v_n^j < x)$, $\hat{v}_n(x) = \{v_n^j(x)\}$ – distributions and distribution vectors of the service time and waiting time for the $n - th$ customer in a $j - th$ node.

Theorem 6.2 (On the distribution of the sojourn time for $GI/G/1$ queues in series)
If the series $s_n^j, j \geq 0$ fulfill the earlier provided conditions, and the condition

$$\alpha_k < \alpha_{k-1} < \cdots < \alpha_1 < 0. \tag{12}$$

holds, there exists a vector of cumulative limit distribution functions of the sojourn time for $GI/G/1$ queues in series, so that

$$\lim_{n\to\infty} \hat{t}_n(x) = \hat{t}(x).$$

Proof. 1. First, consider the first node of the system. It could be proved (see for example [45, p. 170]) that under theorem conditions and if $\alpha^1 < 0$ the limit distribution function of the waiting time for the node of type $GI/G/1$ exists:

$$\lim_{n\to\infty} v_n^1(x) = v^1(x), \forall x > 0.$$

As $t_n^1 = s_n^1 + v_n^1$, so $t_n^1(x) = P(t_n^1 < x) = P(s_n^1 + v_n^1 < x)$, where s_n^1 and

v_n^1 are independent random variables. In this case

$$t_n^1(x) = \int_{-\infty}^{\infty} v_n^1(x-y)ds^1(y)$$

and

$$t^1(x) = \lim_{n\to\infty} t_n^1(x) = \lim_{n\to\infty} \int_{-\infty}^{\infty} v_n^1(x-y)ds^1(y) = \int_{-\infty}^{\infty} v^1(x-y)ds^1(y).$$

From $x-y>0$ we have

$$t^1(x) = \int_{-\infty}^{\infty} v^1(x-y)ds^1(y) = \int_{-\infty}^{x} v^1(x-y)ds^1(y) = \int_{0}^{\infty} v^1(x)ds^1(x-y).$$

2. Consider $j=i$, such that $i>1$ and $i<k$. Assume that for $j=i$ the result holds, that is under conditions: $\tau_n^i, n=1,2,\dots$ and $s_n^i, n=1,2,\dots$ are independent random variables with distributions functions equal to the distribution function of variables τ^i, that is $\tau^i(x)=P(\tau^i<x)$ and s^i, that is $s^i(x)=P(s^i<x)$ and with moments (if exists) denoted as $a_k^i = M(\tau^i)^k$ and $s_k^i = M(s^i)^k$ accordingly – there exists a limit distribution $t^i(x) = \lim_{n\to\infty} t_n^i(x), \forall x>0$. Interarrival time for node $i+1$ is equal $\tau_n^{i+1} = s_{n+1}^i + max(0, \tau_n^i - t_n^i) = s_{n+1}^i + (\tau_n^i - t_n^i)^+$.
   (a) First, let us denote as $\eta_n^i = (\tau_n^i - t_n^i)^+$. In this case we have $\eta_n^i(x) = P((\tau_n^i - t_n^i)^+ < x) \overset{x\geq 0}{=} P(\tau_n^i - t_n^i < x) = -\int_0^\infty t_n^i(x+y)d\tau^i(y)$ and passing to the limit $\eta^i(x) = \lim_{n\to\infty}\eta_n^i(x) = -\lim_{n\to\infty}\int_0^\infty t_n^i(x+y)d\tau^i(y) = -\int_0^\infty t^i(x+y)d\tau^i(y), \forall x>0$ as $\lim_{n\to\infty} t_n^i(x+y) = t^i(x+y)$ exists according to preconditions.
   (b) Next, $\tau_n^{i+1} = s_{n+1}^i + \eta_n^i$. Variables s_{n+1}^i and η_n^i are independent random variables, continuing, $\tau_n^{i+1}(x) = P(\tau_n^{i+1} < x) = P(s_{n+1}^i + \eta_n^i < x) = \int_0^\infty s_n^i(x-y)d\eta^i(y)$, and passing to the limit $\tau^{i+1}(x) = \lim_{n\to\infty}\tau_n^{i+1}(x) = \lim_{n\to\infty}\int_0^\infty s_n^i(x+y)d\eta^i(y) = \int_0^\infty s^i(x+y)d\eta^i(y), \forall x>0$. Variables $\grave{\tau}_n = \tau_n^{i+1}$ and $\grave{s}_n = s_n^{i+1}$ are independent random variables with limit distribution functions $\tau^{i+1}(x)$ and $s^{i+1}(x)$ accordingly. From Preconditions (12) as $E\tau_n^i \geq Es_n^i > Es_n^{i+1}$ – the result follows from Part 1 of the theorem.
3. Combining the above cases, the theorem result follows by induction on j.

□

Theorem 6.3 (On the distribution of the sojourn time for $GI/G/1$ the system of queues in series under overloading conditions)
If series $s_n^j, j \geq 0$ fulfill the earlier provided conditions, and the condition

$$\alpha_k \geq \alpha_{k-1} \geq \cdots \geq \alpha_1 \geq 0. \tag{13}$$

holds, a vector of cumulative limit distribution functions of the sojourn time for $GI/G/1$ the system of queues in series do not exist, and

$$\lim_{n\to\infty} \hat{t}_n(x) = \hat{t}(x) \equiv 0, \forall x.$$

Proof. If the Condition (13) holds, then it follows, that for any node j the system of queues in series will be under overloading conditions, that is for any node j variables $\check{\tau}_n$ and $\grave{s}_n$ will be in relation, such that $E(\check{\tau}_n - \grave{s}_n) < 0$.

1. First, consider the first node of the system. It could be proved (see for example [45, p. 175]), that condition $\alpha^1 > 0$ defines that for any x, $\lim_{n\to\infty} v_n^1(x) = v^1(x) \equiv 0, \forall x$. From this, and from $t^1(x) = \int_0^\infty v^1(x) ds^1(x-y)$ we have $t^1(x) \equiv 0, \forall x$.
2. Consider conditions of Section 2 of Theorem 6.2. Under conditions (13), the limit value of the variable $(\tau_n^i - t_n^i)^+$ will be equal to 0, that is $\lim_{n\to\infty}(\tau_n^i - t_n^i)^+ = 0$. In such a case, $\tau^{i+1}(x) = \lim_{n\to\infty} \tau_n^{i+1}(x) = \lim_{n\to\infty}\{s_{n+1}^i(x) + (\tau_n^i(x) - t_n^i(x))^+\} = \lim_{n\to\infty} s_{n+1}^i(x) = s^i(x)$, and we could describe the node $i+1$ as the node of type $GI/G/1$ with limit interarrival rate $\check{\tau} = \tau^{i+1}(x) = \lim_{n\to\infty} \tau_n^{i+1} = \lim_{n\to\infty} s_{n+1}^i = s^i(x)$. As variables $\check{\tau}_n$ and $\grave{s}_n = s_n^{i+1}$ are independent random variables with limit distribution functions $s^i(x)$ and $s^{i+1}(x)$ accordingly, the result follows from precondition 13 ($\alpha_{i+1} \geq \alpha_i$) and Part 1 of the theorem.
3. Combining the above cases, the theorem result follows by induction on j.

□

Proposition 6.4 (The general form of recurrence equation for calculations of the sojourn time of a customer)
Under earlier described conditions, let us denote by t_n^j the time of arrival of the n-th customer; by s_n^j– the service time of the n-th customer in the j-th node; $\tau_n^j = t_n^j - t_{n-1}^j$; $j = 1, 2, \cdots, k$; $n = 1, 2, \cdots, N$; $\tau_n = \tau_n^1$. The

following recurrence equation for the sojourn time w_n^j of the n-th customer up to node j is valid [45, p. 180]:

$$\begin{aligned} w_n^j = w_n^{j-1} + s_n^j + \max\left(w_{n-1}^j - w_n^{j-1} - \tau_n, 0\right); \\ j = 1, 2, \cdots, k; n = 1, 2, \cdots, N; \\ w_0^j = 0, \forall j; w_n^0 = 0, \forall n. \end{aligned} \tag{14}$$

Proof. It is true that if the time $\tau_n + w_n^{j-1} \geq w_{n-1}^j$, the waiting time in the j-th phase of the n-th customer is 0. In the case of $\tau_n + w_n^{j-1} < w_{n-1}^j$, the waiting time in the j-th phase of the n-th customer is $v_j^n = w_{n-1}^j - w_n^{j-1} - \tau_n$ and $w_n^j = w_n^{j-1} + v_j^n + s_n^j$. Taking into account the above two cases, we finally have the proposition result. $\square$

The problem of the solution provided in Proposition 6.4 is that the solution is not linear and is complicated to implement in practice, taking into account the implementation on parallel machines. The earlier provided results of Theorem 6.2 and Theorem 6.3 give us a possibility to generalize the recurrence (14) linearizing its non linear part.

Proposition 6.5 (The parametric form of recurrence equation for calculations of the sojourn time of a customer under overloading conditions)
Let us define by $\hat{\alpha} = \{\alpha_1, \alpha_2, \dots \alpha_k\}$ – vector of parameters α. Under conditions of the Proposition 6.4 the recurrence equation (14) could be presented in the next parametric form:

$$w_n^j = f' * (1 - \mathbf{P}') + f'' * \mathbf{P}' \tag{15}$$

where

f' is a non linear function of a general form (14);
f'' is a linear function such as: $f''(\hat{w}, \hat{s}, \hat{\tau}) = w_n^{j-1} + s_n^j + \tau_n$,
$j = 1, 2, \cdots, k;\ n = 1, 2, \cdots, N;\ \ w_0^j = 0, \forall j;\ w_n^0 = 0, \forall n$
$\mathbf{P}' : \hat{\alpha} \to \{0, 1\}$ *– are vector $\hat{\alpha}$ predicate of the form (13).*

Proof. The proof is rather straightforward. It is clear that depending on the value of the predicate either linear or non-linear part is vanishing. Therefore we should analyze the value of $\mathbf{P}'$.

(1) The case $\mathbf{P}' \equiv false$ follows from the results of the Proposition 6.4;

(2) Finally, if $\mathbf{P}' \equiv true$, this means that the conditions of Theorem 6.3 hold. Under these conditions $lim_{n\to\infty}\hat{t}_n(x) = \hat{t}(x) \equiv 0, \forall x$, that means that a non linear part of $f' > 0, \forall n \to \infty$. The result of the proposition follows.

□

6.3.3 Benchmarking

The provided below benchmarking is based on counting of the number of assembler instructions for a reduced instruction set style processor. First, the number of instructions for non-linear model is calculated, next,the number of instructions for linear model is calculated. The results in the form of assembler pseudo-code are presented in Algorithms 6.1 and 6.2. As it is seen from the presented pseudo-codes the linear version is at least 2.6 times more effective as it is implemented on machines with Reduced Instruction Set Computing (RISC) processors architectures.

Algorithm 6.1 Benchmarking for the non-linear model

```
global $wjn, sjn, wjnmin, wjminn, taun, max$

main
 $max = wjnmin - wjminn - taun;$
 $\begin{cases} 00: BCF\ \ 03.5 \\ 01: MOVF\ \ 21,W \\ 02: SUBWF\ \ 23,W \\ 03: MOVWF\ \ 78 \\ 04: MOVF\ \ 24,W \\ 05: SUBWF\ \ 78,W \\ 06: MOVWF\ \ 25 \end{cases}$
 $if(max < 0)\ max = 0;$
 $\begin{cases} 07: BTFSS\ \ 25.7 \\ 08: GOTO\ \ 10 \\ 09: CLRF\ \ 25 \end{cases}$
 $wjn = wjminn + sjn + max;$
 $\begin{cases} 10: MOVF\ \ 22,W \\ 11: ADDWF\ \ 21,W \\ 12: ADDWF\ \ 25,W \\ 13: MOVWF\ \ 20 \end{cases}$
```

Algorithm 6.2 Benchmarking for the linear model

```
global $wjn, wjnmin, sjn, taun$

main
 $wjn = wjminn + sjn + taun;$
 $\begin{cases} 01: BCF \quad 03.5 \\ 02: MOVF \quad 22, W \\ 03: ADDWF \quad 21, W \\ 04: ADDWF \quad 24, W \\ 05: MOVWF \quad 20 \end{cases}$
```

6.3.4 Modeling results

The material presented in this section is based on the content of Section 7 of the publication [6]. In this section, the results of statistical modeling of the sojourn time of a customer in the system of queues in series under overloading conditions are presented. Specifically, we are interested in exploring the adequacy of the theoretical results presented earlier. We also pay special attention to Condition (2) as we consider it as the substantial condition for the relevant proof of the theoretical results such as the law of the iterated logarithm for the sojourn time of a customer. Here we also describe technical resources we used for modeling and brief analyses of modeling algorithms. For statistical modeling of a customer's interarrival and serving time, we use exponentially distributed random variables with the parameter $\lambda_z = E\tau_n$ for the customers interarrival time simulation, and with the parameter $\lambda_s^j = Es_n^j$ for the simulation of the serving time in the j-th phase of the system of queues in series. We integrate modeling results using the Monte Carlo simulation technique by selecting limit values of the simulated sojourn time. For implementing simulation algorithms, we use parallel computing techniques powered by the HPC cluster of Vilnius University `http://kedras.mif.vu.lt/itc/`. The cluster uses the Debian GNU/Linux 6.0 operating system(OS). The cluster resource management is implemented by SLURM – The Simple Linux Utility for Resource Management `https://computing.llnl.gov/linux/slurm/`. For implementing parallel programming techniques we use MPI interface Open MPI implementation `http://www.open-mpi.org/faq/?category=slurm`. Other available resources, used during the implementation of the modeling results, are gcc-4.4 4.4.5-8 the GNU C programming language compiler;

gsl-bin 1.14+dfsg-1 GNU Scientific Library (GSL) binary package; python 2.6.6-3+squeeze-7 interactive high-level object-oriented language (default version); python-numpy 1:1.4.1-5 – Numerical Python (adds a fast array facility to the Python language); python-matplotlib 0.99.3-1 – Python based plotting system in a style similar to Matlab.

The main algorithm is written in C using parallel programming techniques. For generating a random number sequence of interarrival and serving times, we use the GSL library provided second-generation ranlux generator gsl_rng_ranlxs2 with the rate of 253k doubles/sec `http://www.gnu.org/software/gsl/manual/html_node/Random-Number-Generator-Performance.html`. Software for data preparation and processing of the results is implemented in Python using numpy and matplotlib extensions. The possibility of using the cluster resources allowed us to obtain sufficient simulation parameters with the total number of customers up to 10^{10}, the total number of serving phases up to 2^{10} and the total number of sequences in simulations up to 100.

The main results of the simulation are presented in Figures 66–77. Here we consider the validity of Condition (2). First, we provide the results of statistical modelings for exponentially distributed interarrival and service times. We clarify the results for a different number of phases. Afterward, we provide the results for the mixed distributions. In this case, we consider the interarrival time as exponentially distributed and the service time as χ - squared distributed. The procedure of evaluation of the results is based on the visual evaluation of simulation plots. We consider the simulation results as valid if each line, which represents one simulation trial, is within the approximation lines, determined by the iterated algorithm law. In such a case we claim that the law of the iterated algorithm, as it is stated in Theorem 6.1, is valid.

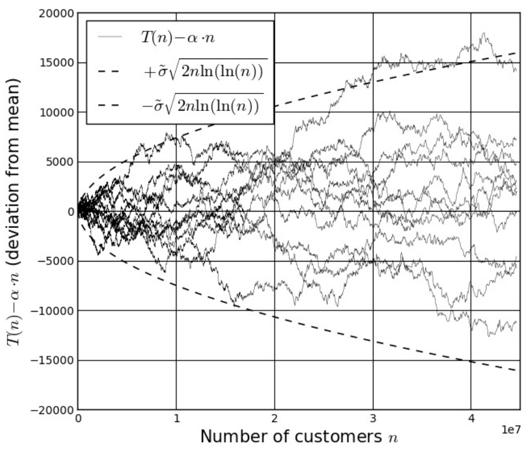


Figure 66: Simulation results for the system of queues in series with 2^{10} phases and 10 Monte Carlo trials. Exponential distribution

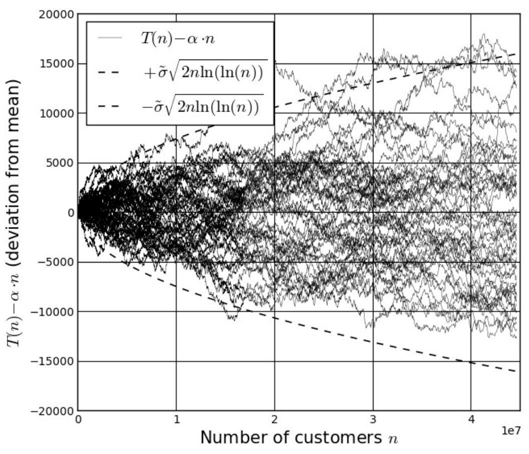


Figure 67: Simulation results for the system of queues in series with 2^{10} phases and 50 Monte Carlo trials. Exponential distribution

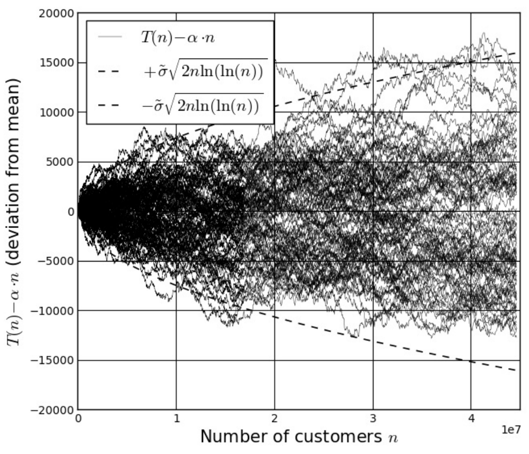


Figure 68: Simulation results for the system of queues in series with 2^{10} phases and 100 Monte Carlo trials. Exponential distribution. Reprinted from [6, p. 18]

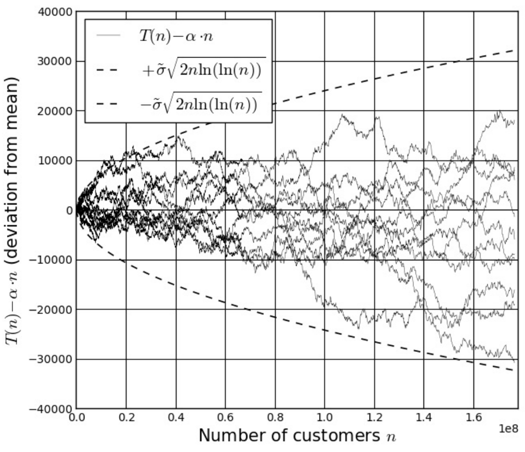


Figure 69: Simulation results for the system of queues in series with 2^8 phases and 10 Monte Carlo trials. Exponential distribution

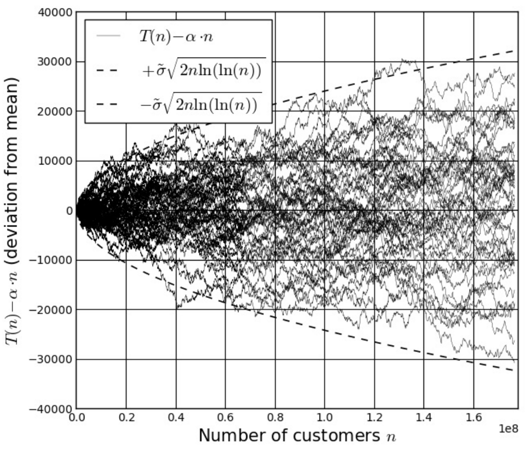


Figure 70: Simulation results for the system of queues in series with 2^8 phases and 50 Monte Carlo trials. Exponential distribution

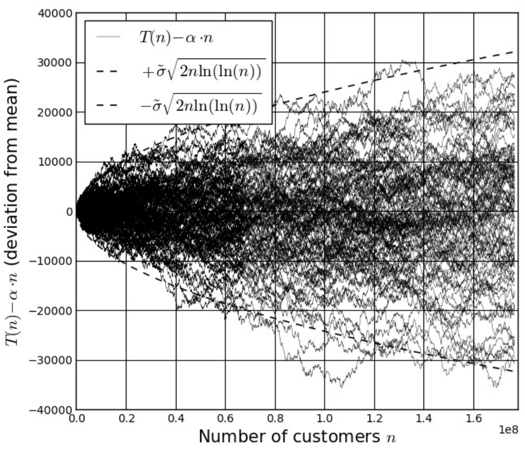


Figure 71: Simulation results for the system of queues in series with 2^8 phases and 100 Monte Carlo trials. Exponential distribution. Reprinted from [6, p. 18]

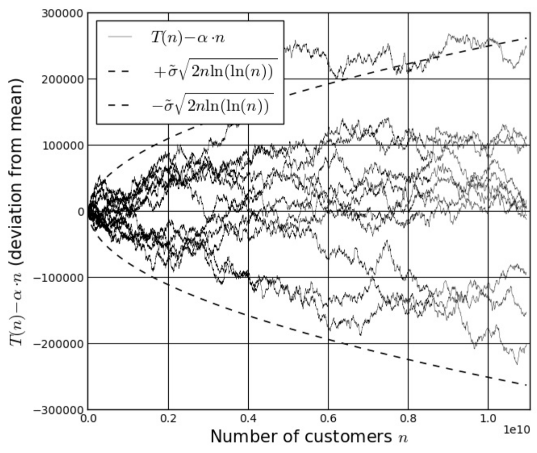


Figure 72: Simulation results for the system of queues in series with 2^3 phases and 10 Monte Carlo trials. Exponential distribution

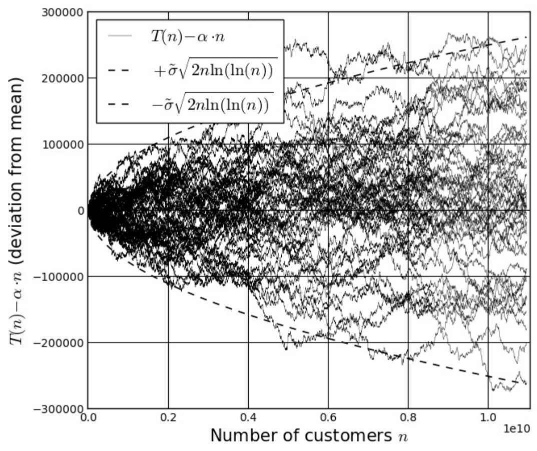


Figure 73: Simulation results for the system of queues in series with 2^3 phases and 50 Monte Carlo trials. Exponential distribution

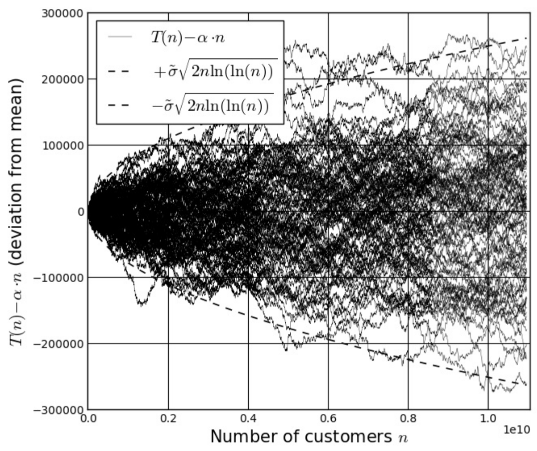


Figure 74: Simulation results for the system of queues in series with 2^3 phases and 100 Monte Carlo trials. Exponential distribution. Reprinted from [6, p. 19]

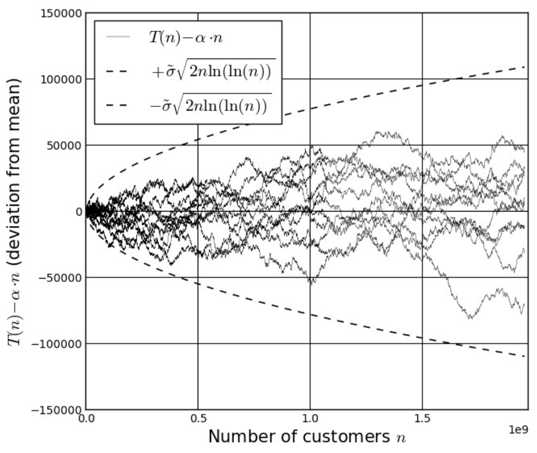


Figure 75: Simulation results for the system of queues in series with 2^2 phases and 10 Monte Carlo trials. Mixed distribution

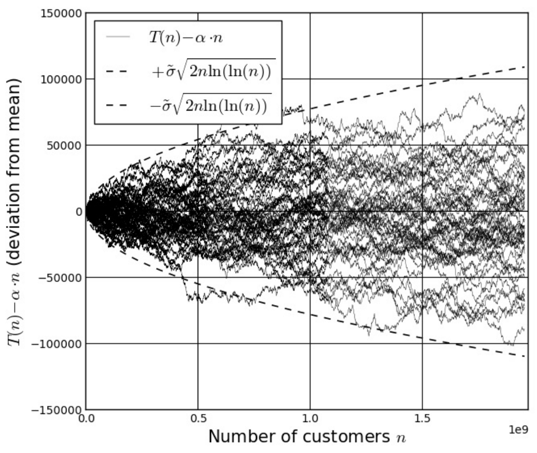


Figure 76: Simulation results for the system of queues in series with 2^2 phases and 50 Monte Carlo trials. Mixed distribution

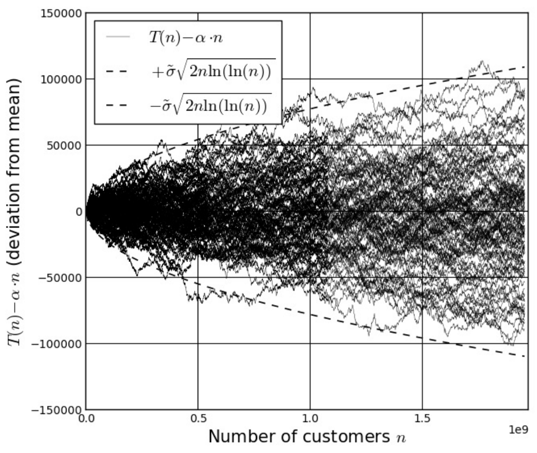


Figure 77: Simulation results for the system of queues in series with 2^2 phases and 100 Monte Carlo trials. Mixed distribution. Reprinted from [6, p. 19]

As we can see from the presented figures, in all the cases of simulation trials for randomly selected interrarival and service times, the relevant Monte Carlo simulation lines are inside the limits defined by the approximation results.

As a plan for future research, in spite of the sufficient values of the simulation parameters, we would like to proceed by increasing the number of trials and the number of phases in the computational model. In this particular research, the number of trials was limited by the number of available processors of the computer cluster. So we could improve the parallel algorithm to make it less dependent on the availability of the cluster resources. Another step could be the simulation of the probability distribution of the sojourn time on the exit of the queues in series. Here we could calculate the main distribution parameters and prepare plots for visual control. Finally, we could proceed with the simulation of a multi-server system of queues in series and investigate the validity of the relevant theoretical results. In the conclusion of this section, taking into account the validity of Condition (11) we state the correlation of simulation and approximation results of the LIL for the sojourn time of a customer in the system of queues in series under overloading conditions.

6.4 Case study of the model-based approaches to teaching interdisciplinary curricula: Python based simulation models

6.4.1 Theoretical basics

The material of this section is mainly based on the results of the publication [4]. We present a brief description of the key topics of an introductory curriculum in scientific computing. These topics include randomness with random numbers and distributions, stochastic simulations and multiprocessing. We use a simple model of throwing a die or a number of dice. The main task of these experiments is to 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. That could provide basic ideas for more advanced experiments with the model of queueing systems.

6.4.2 Random numbers and distributions

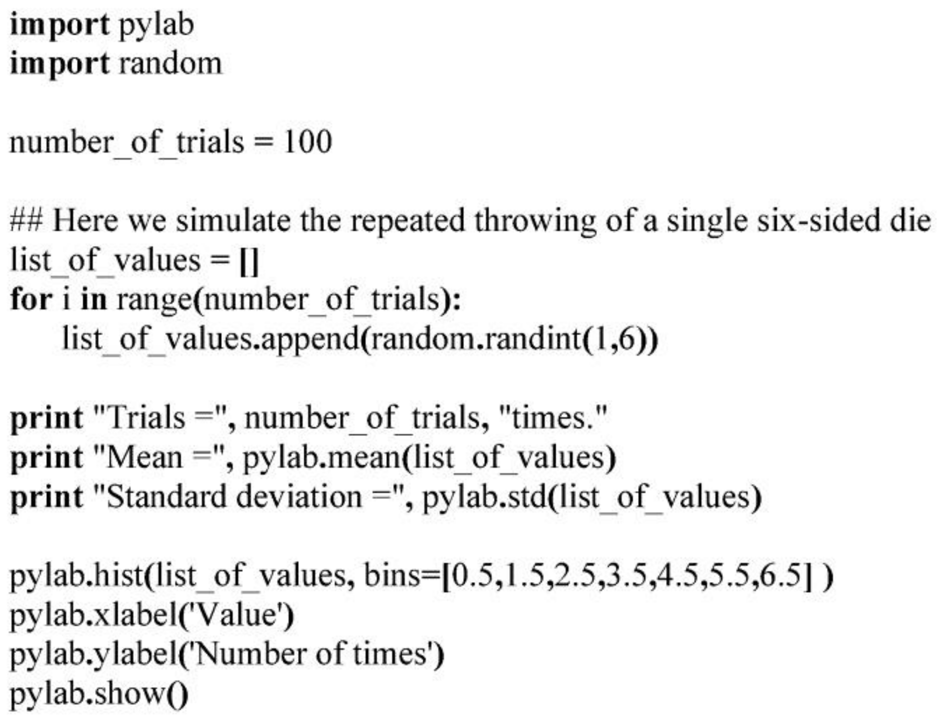
All probability topics could be traditionally considered difficult to understand and are always within the scope of interests of international education scholars [226]. At the same time, such topics are very important in scientific research [227]. The Model-Centered Approach makes it easier to understand the material. The model we are studying is a simple model of throwing a die or a specific number of dice. We start with one die and continue experimenting with more dice.

The aim of these introductory experiments is rather complex. We not only introduce probability and distributions but also we simultaneously introduce stochastic simulations and parallel computing. We also take one-step towards scientific research as we introduce the experimental proof of the Central Limit Theorem.

We begin with the introduction of random number generators leaving distributions behind the scene. We then explain uniform random numbers. Discussions about true randomness or quasi-randomness [228, 229] could follow. For advanced learners, the task to carry out a number of experiments with pseudo-randomness and the Python pseudo-random module could be presented. As an introductory step, the assignment for the learner is to increase the number of trials and supervise the results of simulations. In the next steps, we proceed to more sophisticated experiments and parallel calculations. We use the Python random module for simulations and the *mpi4py* for parallel programming. The Python random module implements pseudo-random number generators for various distributions. For example, $random.randint(a, b)$ returns a random integer N such that $a \leq N \leq b$ and $random.expovariate(lambd)$ returns exponentially distributed random numbers with the parameter 'lambda'. One should refer to Python documentation for specific details. The programming model of a single die is presented in Figure 78 (reprinted from [4, p. 39]). The results of a simulation in the case of a single die are presented in in Figure 79 (reprinted from [4, p. 39]).

Next, we proceed to the case of two dice. The main idea at this point is to explain the Central Limit Theorem by experimenting with different numbers of dice. Figure 80 (reprinted from [4, p. 40]) represents this idea.

The learner proceeds by modifying the two-dice code that enables him



```
import pylab
import random

number_of_trials = 100

## Here we simulate the repeated throwing of a single six-sided die
list_of_values = []
for i in range(number_of_trials):
    list_of_values.append(random.randint(1,6))

print "Trials =", number_of_trials, "times."
print "Mean =", pylab.mean(list_of_values)
print "Standard deviation =", pylab.std(list_of_values)

pylab.hist(list_of_values, bins=[0.5,1.5,2.5,3.5,4.5,5.5,6.5] )
pylab.xlabel('Value')
pylab.ylabel('Number of times')
pylab.show()
```

Figure 78: Python single die model. Reprinted from [4, p. 39]

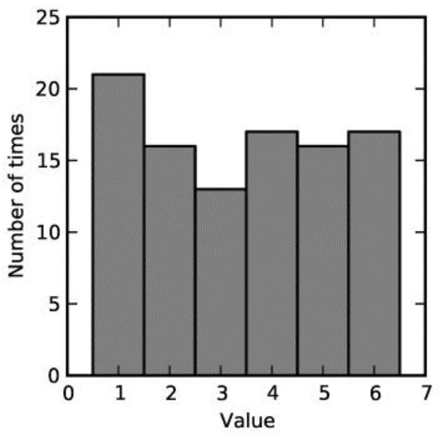


Figure 79: Simulation results for a single die. Reprinted from [4, p. 39]

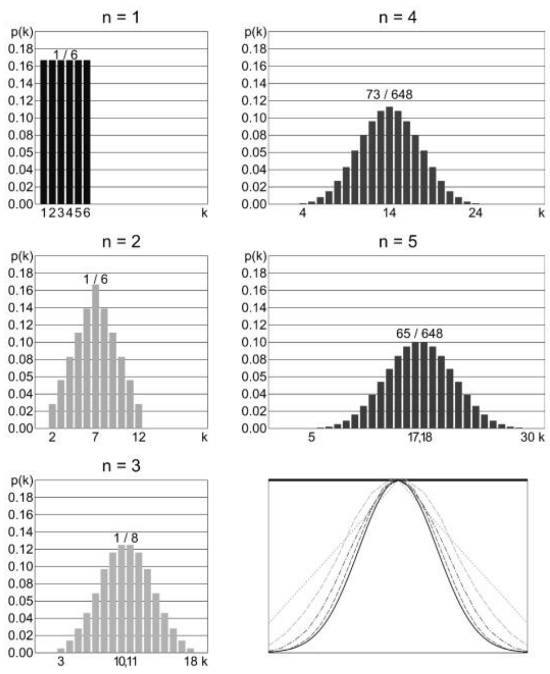


Figure 80: Comparison of the probability density functions. Reprinted from [4, p. 40]

to start a multi-dice case. The code is analogical to the one die code except for the two instructions presented below:

```
list_of_values.append(random.randint(1,6) + random.randint(1,6))
...
pylab.hist(list_of_values, pylab.arange(1.5,13.5,1.0) )
...
```

The results of a simulation in the two dice case are presented in Figure 81 (reprinted from [4, p. 40]).

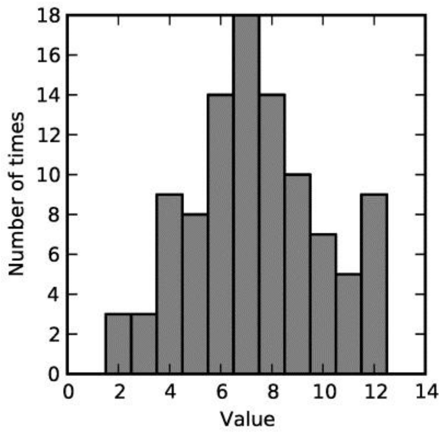


Figure 81: Two-dice case. Reprinted from [4, p. 40]

We can now proceed to normal distribution. The task here is to show how the above multi-dice case correlates with the normal distribution. Another task would be to introduce the mean and deviation. The code is similar to the one-die case except for the instruction used below:

```
...
list_of_values.append(random.normalvariate(7,2.4))
...
```

The results of a simulation for normal distribution are presented in Figure 82 (reprinted from [4, p. 40]).

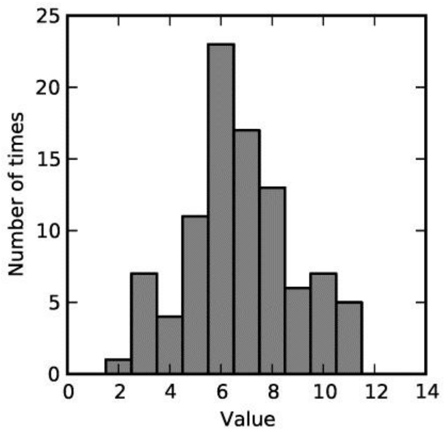


Figure 82: Simulation results for normal distribution. Reprinted from [4, p. 40]

The final step is to introduce the exponential distribution. One always uses the exponential distribution for simulating interarrival times of customers in queueing systems of various types. The model of the exponential distribution and the results of a simulation are presented in Figure 83 (reprinted from [4, p. 41]) and Figure 84 (reprinted from [4, p. 40]).

6.4.3 Stochastic simulation

Stochastic simulation is of primary importance in the field of SC . We focus on Monte Carlo methods [227, 230, 231]. After the model has been constructed, we could generate random variables and experiment with different parameters of the system. In the scope of this section, the point of the Monte Carlo experiments is to repeat the trials of our model many times with a view to accumulate and integrate the overall results. The simplest application was described in the previous subsection. If we increase the

```
import pylab
import random

number_of_trials = 1000
number_of_customer_per_hour=10

## Here we simulate the  interarrival time of the customers

list_of_values = []
for i in range(number_of_trials):
            list_of_values.append(random.expovariate(
            number_of_customer_per_hour))

mean=pylab.mean(list_of_values)
std=pylab.std(list_of_values)
print "Trials =", number_of_trials, "times"
print "Mean =", mean
print "Standard deviation =", std

pylab.hist(list_of_values,20)
pylab.xlabel('Value')
pylab.ylabel('Number of times')
pylab.show()
```

Figure 83: Python model for the exponential distribution. Reprinted from [4, p. 40]

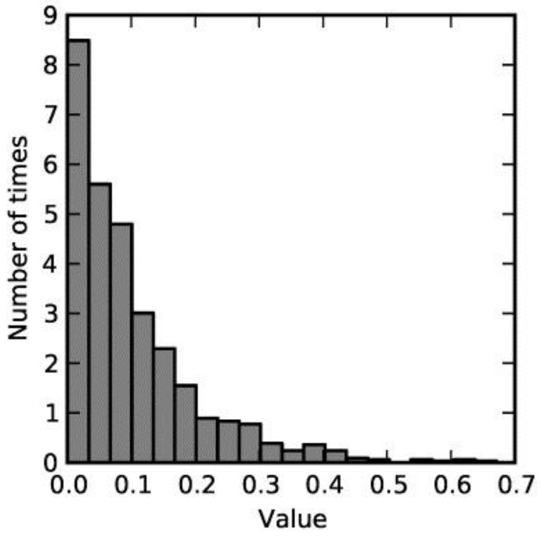


Figure 84: Simulation results for the exponential distribution. Reprinted from [4, p. 40]

number of trials, we increase the preciseness of simulation results. Here the learner should carry out a certain number of experiments using this simple model by increasing the number of trials. By increasing the number of dice and the number of trials, the learner will face relatively long calculation times. It could be a good motivation to use parallel calculations. The Python model of multiple dice is presented in Figure 85 and the result of a simulation is presented in Figure 86 (reprinted from [4, p. 41]).

```
import pylab
import random

number_of_trials = 150000
number_of_dice=200

## Here we simulate the repeated throwing
## of a number of single six-sided dice
list_of_values = []
for i in range(number_of_trials):
      sum=0
      for j in range(number_of_dice): sum+=random.randint(1,6)
      list_of_values.append(sum)

mean=pylab.mean(list_of_values)
std=pylab.std(list_of_values)
print "Trials =", number_of_trials, "times."
print "Mean =", mean
print "Standard deviation =", std

pylab.hist(list_of_values,20)
pylab.xlabel('Value')
pylab.ylabel('Number of times')
pylab.show()
```

Figure 85: Python model of multiple dice. Reprinted from [4, p. 41]

As a next step, a set of more comprehensive problems like modeling of various queueing systems could be introduced for the learner. A brief introduction to the classification of queueing systems is presented in the next section of this study. The learner begins with the modeling of $M/M/1$ system or a more complex queueing system. The basic meanings of stochastic processes might be introduced at this step as well. As a possible example, the problem of investigation of the output process could be offered. One can prove that for $M/M/1$ system the output is again the Poisson process. So the problem of gathering data and plotting the output empirical histogram might be presented.

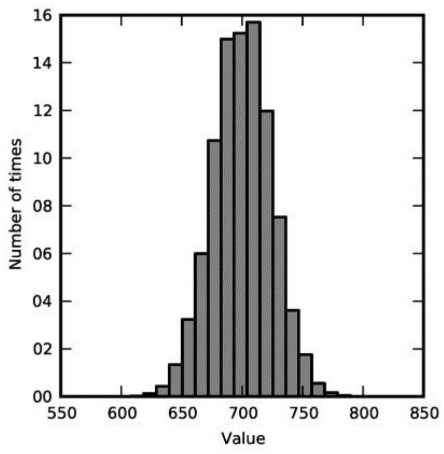


Figure 86: Simulation results of multiple dice. Reprinted from [4, p. 41]

6.4.4 Systems of queues in series and stochastic simulation

6.4.4.1 Queueing systems

Below we provide an introductory description of queueing systems that consider modeling and stochastic simulation positions. A simple queueing system consists of one server that provides service for arriving customers. The general scheme of the simple queueing system is presented in Figure 87 (reprinted from [4, p. 41]).

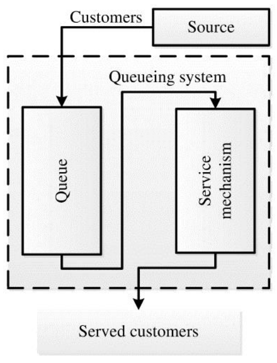


Figure 87: The simple queueing system. Reprinted from [4, p. 42]

In general, the queueing system consists of one or more servers that provide service to arriving customers. This could also include one or more servicing phases with one or more servers in each phase. Arriving customers who find all the servers busy join one or more queues in front of the servers. Many applications can be modeled as queueing systems, such as manufacturing systems, communication systems, maintenance systems, and so on.

An overall queueing system could be characterized by three main components: the Arrival process, the Service mechanism, and the Queue discipline. Arrivals may come from one or several limited or unlimited sources.

The arrival process describes how customers arrive at the system. We denote by α_i the interarrival time between the arrivals of the $(i-1)$ and $i-th$ customer, the expected inter-arrival time (or mean) by $E(\alpha)$ and the arrival frequency by

$$\lambda = \frac{1}{E(\alpha)}$$

We denote by s the number of servers in the queueing system. The service mechanism is specified by that number. Each server has its own queue as well as the probability distribution of customer service time. We denote by s_i the service time of the $i-th$ customer, by $E(s)$ the mean service time of a customer and by

$$\mu = \frac{1}{E(s)}$$

the service rate of a server.

The rule that any server uses to choose the next customer from the queue is called the queue discipline of a queueing system. The most used queue disciplines are: Priority – customers are served in the order of their importance; FIFO – customers are served on the First-in First-out basis; LIFO – customers are served on the Last-in Last-out basis. The extended Kendall classification of queuing systems uses 6 symbols: $A/B/s/q/c/p$ where A is the distribution of intervals between arrivals, B is the distribution of service duration, s is the number of servers, q is the queuing discipline (omitted for FIFO), c is the system capacity (omitted for unlimited queues), p is the number of possible customers (omitted for open systems) [232, 233]. For example, $M/M/1$ states for Poisson input, one exponential server, one unlimited FIFO queue, and unlimited customer population.

Queueing systems are used for modeling and research in various fields of engineering and science. For example, we could model and study manufacturing or transport systems using the queueing theory. Here the requests for service are considered as customers and the maintenance procedure as a service mechanism. The other examples are computer systems (terminal requests and server response accordingly), a computer multi-disk memory system (data writing/reading requests, shared disk controller), a trunked radio system (telephone signals, repeaters), local area computer network (requests, channel) [234]. In biology, one could employ a queueing theory to model an enzymatic system (proteins, common enzyme) [235]. In biochemistry, one could implement a queueing network model to study the regulatory circuit of the lac operon [236].

6.4.4.2 Why the system of queues in series

We consider a queueing system as a system of queues in series – it consists of more than one server, which is joined consequentially, and as unlimited – with an unlimited calling population. The interarrival time and the service time are both independent and exponentially distributed variables. The Queue discipline is endless FIFO. A multiphase queueing system naturally maps to a multicore computer topology. As we see in later sections of this report, such a model could be easily programmed, studied and modified. The model also allows a comparative study of different approaches to multiprocessing. The model of the system of queues in series is presented in Figure 100.

6.4.4.3 Theoretical framework

In the case of statistical modeling, we always face the problem of the computer code verification. Therefore, it is always an open question if there are any errors in our program or algorithm. The model is not fully analytical and each time we run the program, we have different inputs and outputs. So, to verify the correctness of the code or algorithm, a different (from that one we use in the case of fully deterministic input data) approach is needed. To solve this question, we could apply some theoretical results, which could be found in the scientific literature. Such results give us a base for output data analysis and verification as well as for solving the problem of the correctness of the modeling results [6]. We will investigate the sojourn time of the customer in the system of queues in series. The theoretical result of modeling is provided in 6.1 of this study.

6.4.4.4 Statistical modeling

After the model has been constructed, we could arrange a number of experiments with that model. This allows us to investigate certain parameters of the system we study. We could simulate random variables with an expected mean and calculate (using the recurrent equation presented below) the values needed to study. These values will be random as well (we have randomness in the input data of our model – interarrival times and serving times). Afterward, we can calculate some parameters of such random values (variables) as mean or probability distribution. We call this method as statistical modeling due to the randomness presented in the model. If more reliable results are needed, we must repeat the experiments with our model and then integrate the results i.e. calculate integral characteristics like a mean or a standard deviation. This is called the Monte Carlo method and it was described earlier in this study.

6.4.4.5 Recurrent equation

In order to design the modeling algorithm of the previously described queueing system, some additional mathematical constructions should be introduced. Our goal is to calculate and investigate the sojourn time of the customer number n in the system of queues in series of k phases. Recurrent equation (14) for calculation of the sojourn time could be provided [45]. This enables the implementation of the necessary algorithms since all the basic theoretical results have been introduced.

6.4.5 Python for multiprocessing

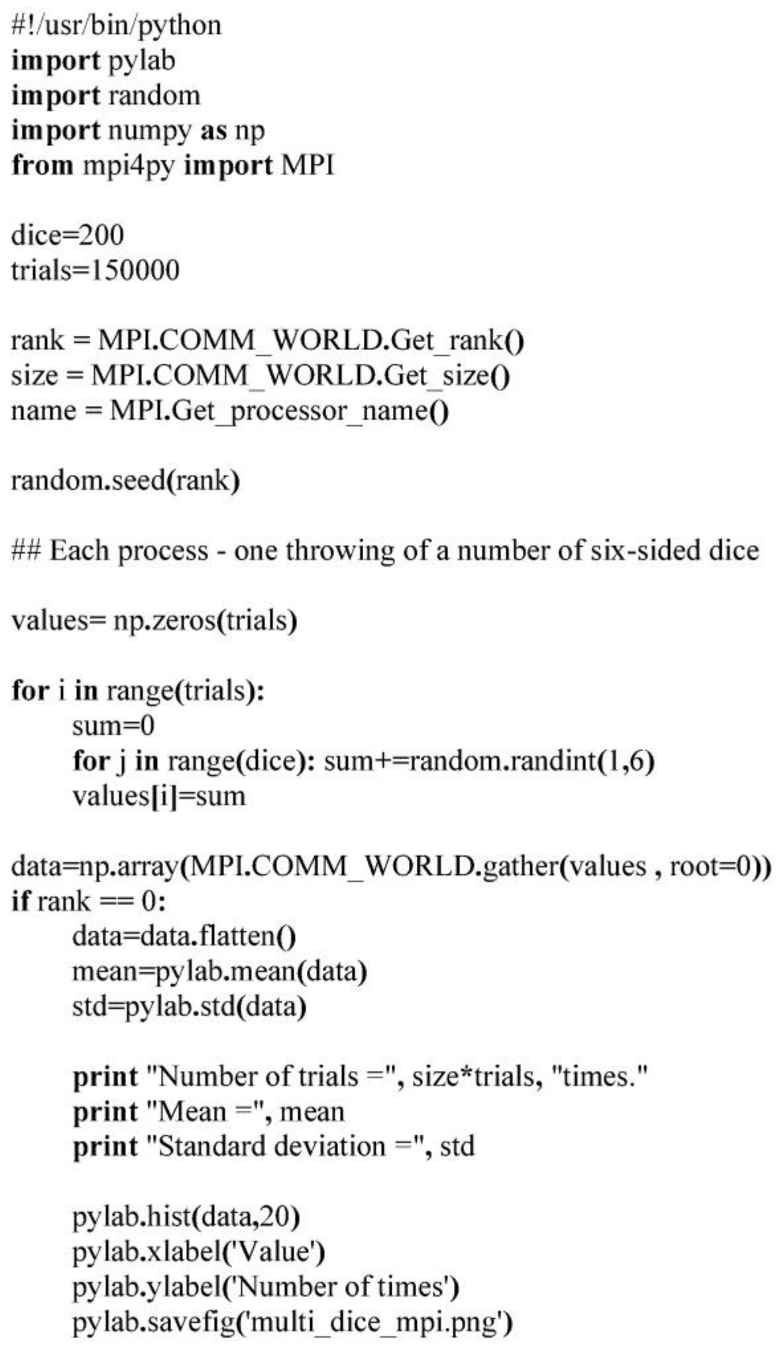
Python as a programming language is very popular among scientists and educators and could be an attractive solution for solving scientifically oriented tasks `http://www.linuxjournal.com/magazine/use-python-scientific-computing` [237]. Python provides a powerful platform for modeling and simulations including graphical options, a wide amount of mathematical and statistical packages as well as packages for multiprocessing. For time consumable solutions, Python and C codes could be combined. All that allows us to implement a powerful modeling platform for statistical modeling and processing of the results of the data. The key Python concepts which are important for modeling are decorators, coroutines, yield expressions, multiprocessing, and queues [238]. Although there are several ways of organizing the inter-process communication, we start with using queues, as it is very natural in the context of the queueing systems. The simple example of the advantage of using multiprocessing in order to increase the efficiency of the programming code is provided below. The learner could proceed with improvements to the provided model by using parallel calculations on super-computers or computer clusters [239, 240]. On the one hand, multiprocessing will allow us to map the model of the system of queues in series to the resources of a multicore computer and on the other hand, we could use multiprocessing to perform a number of Monte Carlo trials in parallel. We present these two approaches in the next sections. For motivated learners, a brief introduction to multiprocessing with Python presented below could be provided. We start by using the mpi4py module. It is important to show the learner the general idea of how MPI works. It simply copies the provided program to a number of the processor kernels, specified by the user, and integrates the results after using the gather() method. The sample Python code (see Figure 88, reprinted from [4, p. 41]) and simulation results (see Figure 89, reprinted from [4, p. 41]) are presented.

6.4.6 Experiments with the models

In this section, we provide 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 modeling and simulation.

The first model presented by us is based on the real-time recordings and we call it an imitative model. It uses the Python multiprocessing module. The precision of this model depends on the precision and resolution of the time() method. It could be rather low in the case of various general-purpose operating systems and rather high in the case of the 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. The model does not use multiprocessing directly. It emulates multiprocessing by using Python yield expressions. The last



```
#!/usr/bin/python
import pylab
import random
import numpy as np
from mpi4py import MPI

dice=200
trials=150000

rank = MPI.COMM_WORLD.Get_rank()
size = MPI.COMM_WORLD.Get_size()
name = MPI.Get_processor_name()

random.seed(rank)

## Each process - one throwing of a number of six-sided dice

values= np.zeros(trials)

for i in range(trials):
    sum=0
    for j in range(dice): sum+=random.randint(1,6)
    values[i]=sum

data=np.array(MPI.COMM_WORLD.gather(values , root=0))
if rank == 0:
    data=data.flatten()
    mean=pylab.mean(data)
    std=pylab.std(data)

    print "Number of trials =", size*trials, "times."
    print "Mean =", mean
    print "Standard deviation =", std

    pylab.hist(data,20)
    pylab.xlabel('Value')
    pylab.ylabel('Number of times')
    pylab.savefig('multi_dice_mpi.png')
```

Figure 88: Python model for the advanced normal distribution with MPI. Reprinted from [4, p. 44]

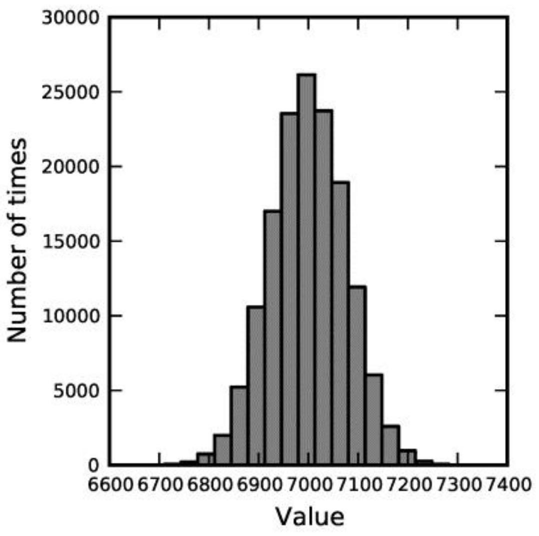


Figure 89: Normal distribution with MPI. Reprinted from [4, p. 44]

model presented here uses the Python MPI *mpi4py* module. Now we use real MPI techniques for statistical modeling and could enhance Monte Carlo simulations by 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.

6.4.7 The imitative model based on multiprocessing of services

Below we present the imitative model. The main issue to study is the difference between the imitative and statistical models. Another important question is the correctness or precision of the imitative model. It is also important to solve the question of verification of the presented model. The learner could study and compare modeling results depending on various modeling parameters such as interarrival and servicing frequencies, the numbers of customers and services. The general schema of the model is presented in Figure 90 (reprinted from [4, p. 46]).

The programming code (see Figure 91, reprinted from [4, p. 47]) consists of two main parts. The first one is directly intended for calculations and the next one is for plotting of the results. The module for calculations contains three main functions: *producer()* – for producing customers and putting them to the first queue; *server()* – for serving the customers; *consumer()* – for finalizing the results. This programming model is based on realistic

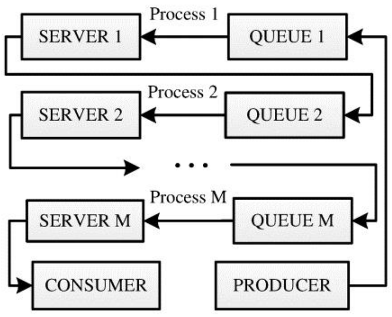


Figure 90: The Imitative Model. Reprinted from [4, p. 46]

simulations and uses no mathematical equations for calculations. Its precision depends on the precision of the Python timing module and generally varies depending on the operating system. Servers are distributed between various processes inside the multiprocessing system.

Questions to be studied:

- How are global variables shared between processes?
- How will the processes, associated with different servers terminate?
- How is the informational flow between various processes transferred?
- What about the correctness of the model?
- What about the efficiency of the model? How long does it take for different processes to exchange information?

Now we can print the results using the Python *matplotlib* module and we can visually analyze the results after the plot is prepared. We can see (see Figure 92, reprinted from [4, p. 47]) that the model needs further improvements. Therefore, we can proceed with a more powerful model.

6.4.8 The single process statistical model

The main features of the statistical model are as follows: now we use the recurrent equation for exact calculations of the customer's sojourn time; we process all the data in a single process using Python specific coroutine functions; we proceed with a definite number of Monte Carlo simulations for a better validity of the calculations. This model gives us "exact" calculations of the sojourn time. The general schema of the model is presented in Figure 93 (reprinted from [4, p. 47]). The learner could study the differences between the imitative and statistical models.

```
import multiprocessing
import time
import random
import numpy as np

def server(input_q,next_q,i):
  while True:
    item = input_q.get()
    if i==0:item.st=time.time()  ## start recording time
                                   ## (first phase)
    time.sleep(random.expovariate(glambda[i]))
##stop recording time (last phase)
    if i==M-1:item.st=time.time()-item.st
    next_q.put(item)
    input_q.task_done()
  print("Server%d stop" % i) ##will be never printed why'

def producer(sequence,output_q):
  for item in sequence:
    time.sleep(random.expovariate(glambda[0]))
    output_q.put(item)

def consumer(input_q):
  "Finalizing procedures"
   ## start recording processing time
  ptime=time.time()
  in_seq=[]
  while True:
    item = input_q.get()
    in_seq+=[item]
    input_q.task_done()
    if item.cid == N-1:
      break
  print_results(in_seq)
  print("END")
  print("Processing time sec. %d" %(time.time()-ptime))
  ## stop recording processing time
  print("CPU used %d" %(multiprocessing.cpu_count()))

def print_results(in_seq):
  "Output rezults"
  f=open("out.txt","w")
  f.write("%d\n" % N)
  for t in range(M):
    f.write("%d%s" % (glambda[t],","))
  f.write("%d\n" % glambda[M])

  for t in range(N-1):
    f.write("%f%s" % (in_seq[t].st,","))
  f.write("%f\n" % (in_seq[N-1].st))
  f.close()

class Client(object):
  "Class client"
  def __init__(self,cid,st):
    self.cid=cid  ## customer id
    self.st=st ## sojourn time of the customer

###GLOBALS
N=100  ## total number of customers arrived
M=5   ## number of servers
### glambda - arrival + servicing frequency
### = customers/per time unit
glambda=np.array([30000]+[i for i in
                    np.linspace(25000,5000,M)])

###START
if __name__ == "__main__":
  all_clients=[Client(num,0) for num in range(0,N)]
  q=[multiprocessing.JoinableQueue() for i in range(M+1)]

  for i in range(M):
    serv = multiprocessing.Process(target=server,args=
                                    (q[i],q[i+1],i))
    serv.daemon=True
    serv.start()

  cons = multiprocessing.Process(target=consumer,
                                  args=(q[M],))
  cons.start()

  ### start 'produsing' customers
  producer(all_clients,q[0])

  for i in q: i.join()
```

Figure 91: Python based solution for the imitative model based on multiprocessing of services. Reprinted from [4, p. 47]

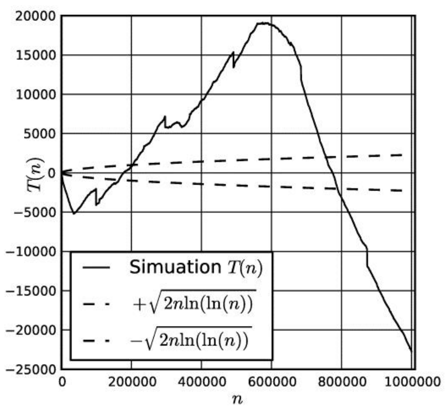


Figure 92: Simulation results for the imitative model. Reprinted from [4, p. 47]

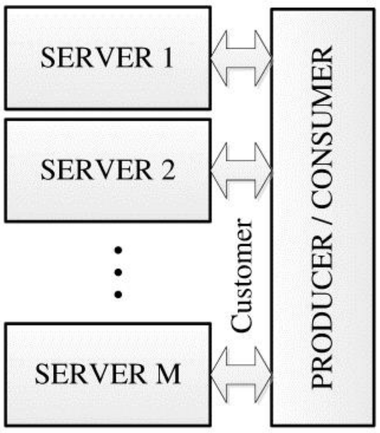


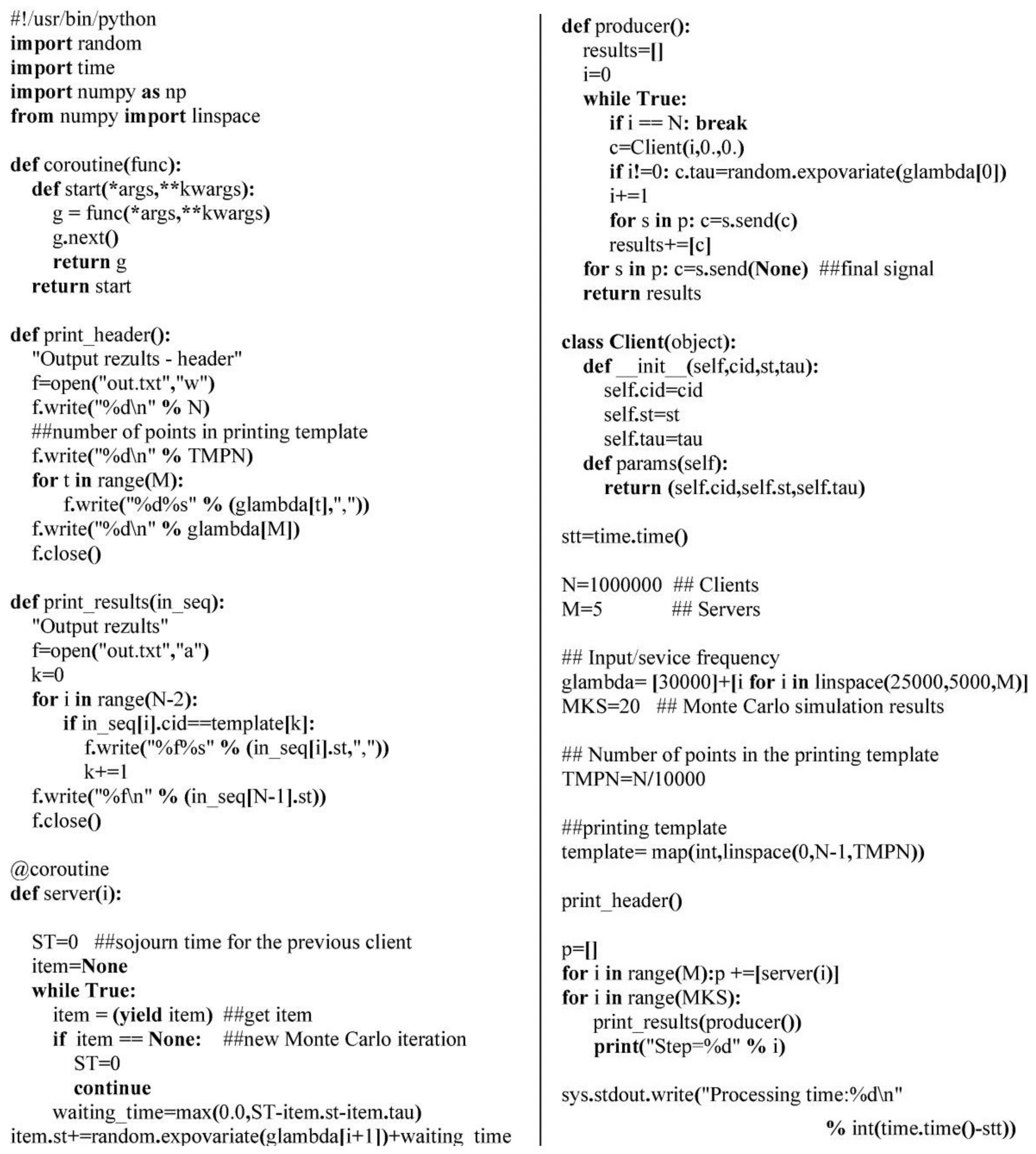
Figure 93: Simulation process for the imitative model. Reprinted from [4, p. 47]

The computer code for implementing of the above model is presented in Figure 94. The simulation results are presented in Figure 95 (reprinted from [4, p. 41]).

6.4.9 Statistical model strengthened by MPI

The next step is to strengthen our model using the Python MPI module – *mpi4py*. It allows us to proceed with more Monte Carlo simulations and using computer cluster for running and testing the model. The next step could be a further improvement of the model by using the C programming language, "real" MPI or Simplified Wrapper and Interface Generator (SWIG) technology for Python. This model is almost identical to the previous model with the only difference that it uses *mpi4py* for multiprocessing and integrating the results (see Figure 96, reprinted from [4, p. 41]).

In addition to the previous model, several additional modules need to be imported. The *print_results()* function also needs to be rewritten, because we now have more trials. We should also rewrite the main part of the program. In Figure 97 (reprinted from [4, p. 49]) we present only that part of the computer code which differs from the code of the previous model. The results of the simulation are presented in Figure 98 (reprinted from [4, p. 50]). The use case model of the presented LR in the form of a concept map is presented in Figure 99. Design Science Research considerations concerning the presented use case are discussed in detail in sections 4.2.1.4 and 4.2.2 of this thesis.



```
#!/usr/bin/python
import random
import time
import numpy as np
from numpy import linspace

def coroutine(func):
    def start(*args,**kwargs):
        g = func(*args,**kwargs)
        g.next()
        return g
    return start

def print_header():
    "Output rezults - header"
    f=open("out.txt","w")
    f.write("%d\n" % N)
    ##number of points in printing template
    f.write("%d\n" % TMPN)
    for t in range(M):
        f.write("%d%s" % (glambda[t],","))
    f.write("%d\n" % glambda[M])
    f.close()

def print_results(in_seq):
    "Output rezults"
    f=open("out.txt","a")
    k=0
    for i in range(N-2):
        if in_seq[i].cid==template[k]:
            f.write("%f%s" % (in_seq[i].st,","))
            k+=1
    f.write("%f\n" % (in_seq[N-1].st))
    f.close()

@coroutine
def server(i):

    ST=0  ##sojourn time for the previous client
    item=None
    while True:
        item = (yield item)  ##get item
        if  item == None:    ##new Monte Carlo iteration
            ST=0
            continue
        waiting_time=max(0.0,ST-item.st-item.tau)
item.st+=random.expovariate(glambda[i+1])+waiting_time
```

```
def producer():
    results=[]
    i=0
    while True:
        if i == N: break
        c=Client(i,0.,0.)
        if i!=0: c.tau=random.expovariate(glambda[0])
        i+=1
        for s in p: c=s.send(c)
        results+=[c]
    for s in p: c=s.send(None)  ##final signal
    return results

class Client(object):
    def __init__(self,cid,st,tau):
        self.cid=cid
        self.st=st
        self.tau=tau
    def params(self):
        return (self.cid,self.st,self.tau)

stt=time.time()

N=1000000  ## Clients
M=5          ## Servers

## Input/sevice frequency
glambda= [30000]+[i for i in linspace(25000,5000,M)]
MKS=20   ## Monte Carlo simulation results

## Number of points in the printing template
TMPN=N/10000

##printing template
template= map(int,linspace(0,N-1,TMPN))

print_header()

p=[]
for i in range(M):p +=[server(i)]
for i in range(MKS):
     print_results(producer())
     print("Step=%d" % i)

sys.stdout.write("Processing time:%d\n"
                                        % int(time.time()-stt))
```

Figure 94: Python based solution for the single process statistical model. Reprinted from [4, p. 48]

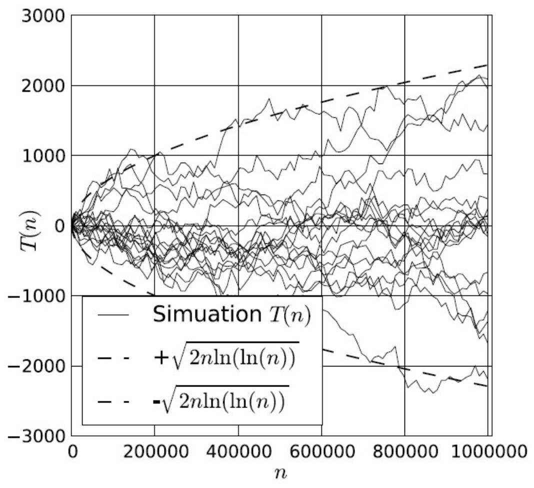


Figure 95: Simulation results for the single process statistical model. Reprinted from [4, p. 49]

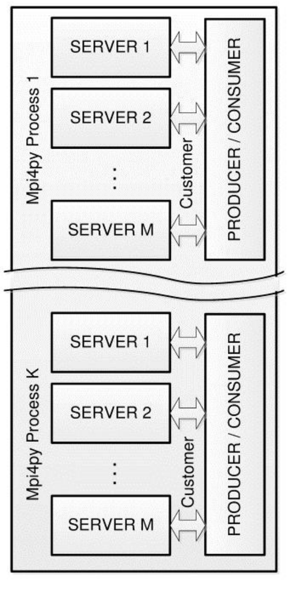
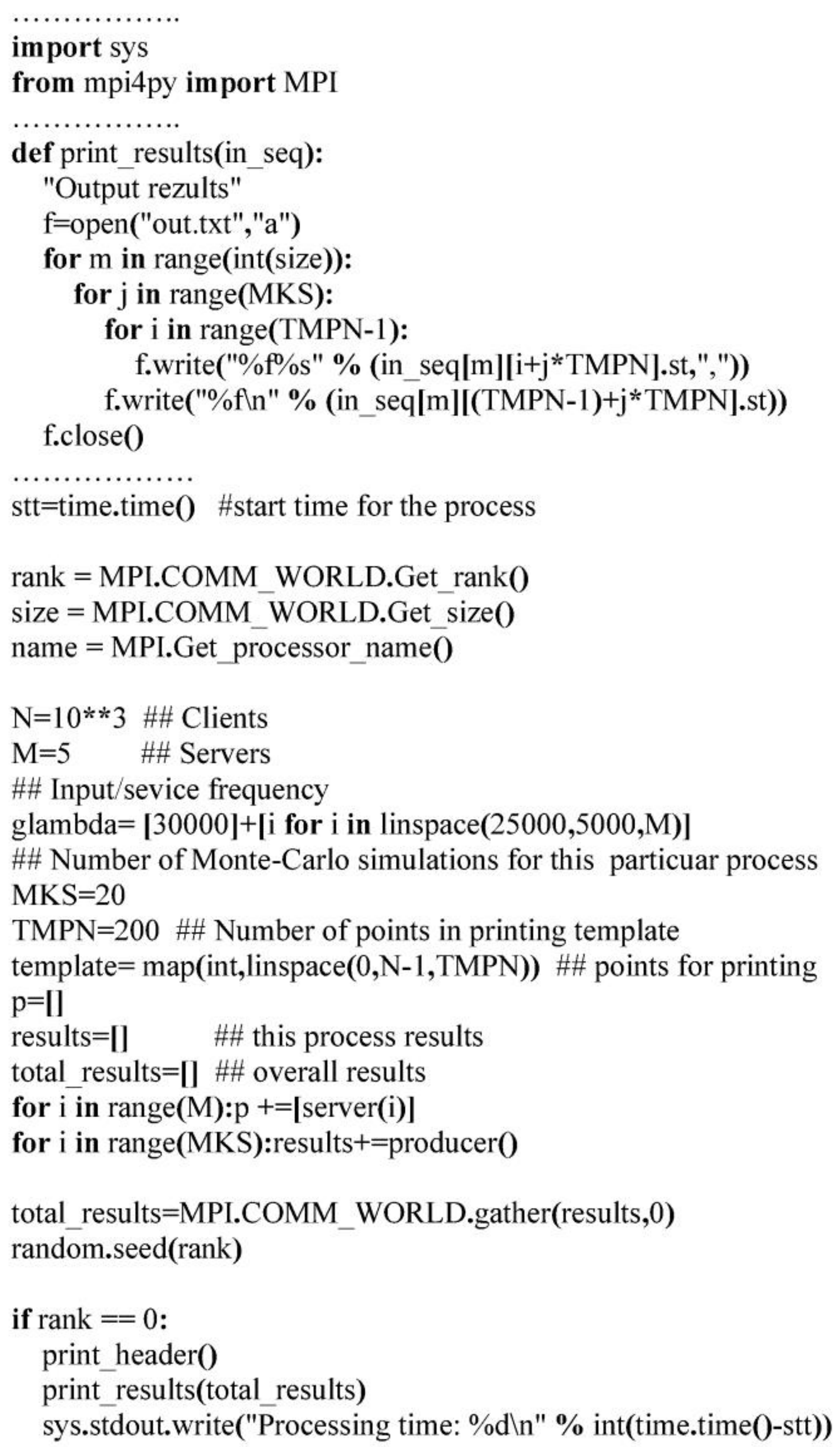


Figure 96: The MPI statistical model. Reprinted from [4, p. 49]



```
................
import sys
from mpi4py import MPI
................
def print_results(in_seq):
    "Output rezults"
    f=open("out.txt","a")
    for m in range(int(size)):
        for j in range(MKS):
            for i in range(TMPN-1):
                f.write("%f%s" % (in_seq[m][i+j*TMPN].st,","))
            f.write("%f\n" % (in_seq[m][(TMPN-1)+j*TMPN].st))
    f.close()
..................
stt=time.time()   #start time for the process

rank = MPI.COMM_WORLD.Get_rank()
size = MPI.COMM_WORLD.Get_size()
name = MPI.Get_processor_name()

N=10**3  ## Clients
M=5         ## Servers
## Input/sevice frequency
glambda= [30000]+[i for i in linspace(25000,5000,M)]
## Number of Monte-Carlo simulations for this  particuar process
MKS=20
TMPN=200  ## Number of points in printing template
template= map(int,linspace(0,N-1,TMPN))  ## points for printing
p=[]
results=[]          ## this process results
total_results=[]  ## overall results
for i in range(M):p +=[server(i)]
for i in range(MKS):results+=producer()

total_results=MPI.COMM_WORLD.gather(results,0)
random.seed(rank)

if rank == 0:
    print_header()
    print_results(total_results)
    sys.stdout.write("Processing time: %d\n" % int(time.time()-stt))
```

Figure 97: Python based solution for statistical model strengthened by MPI. Reprinted from [4, p. 49]

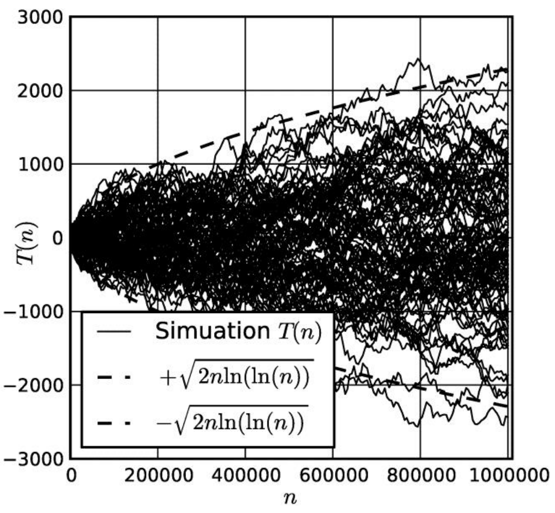


Figure 98: Simulation results for MPI statistical model. Reprinted from [4, p. 50]

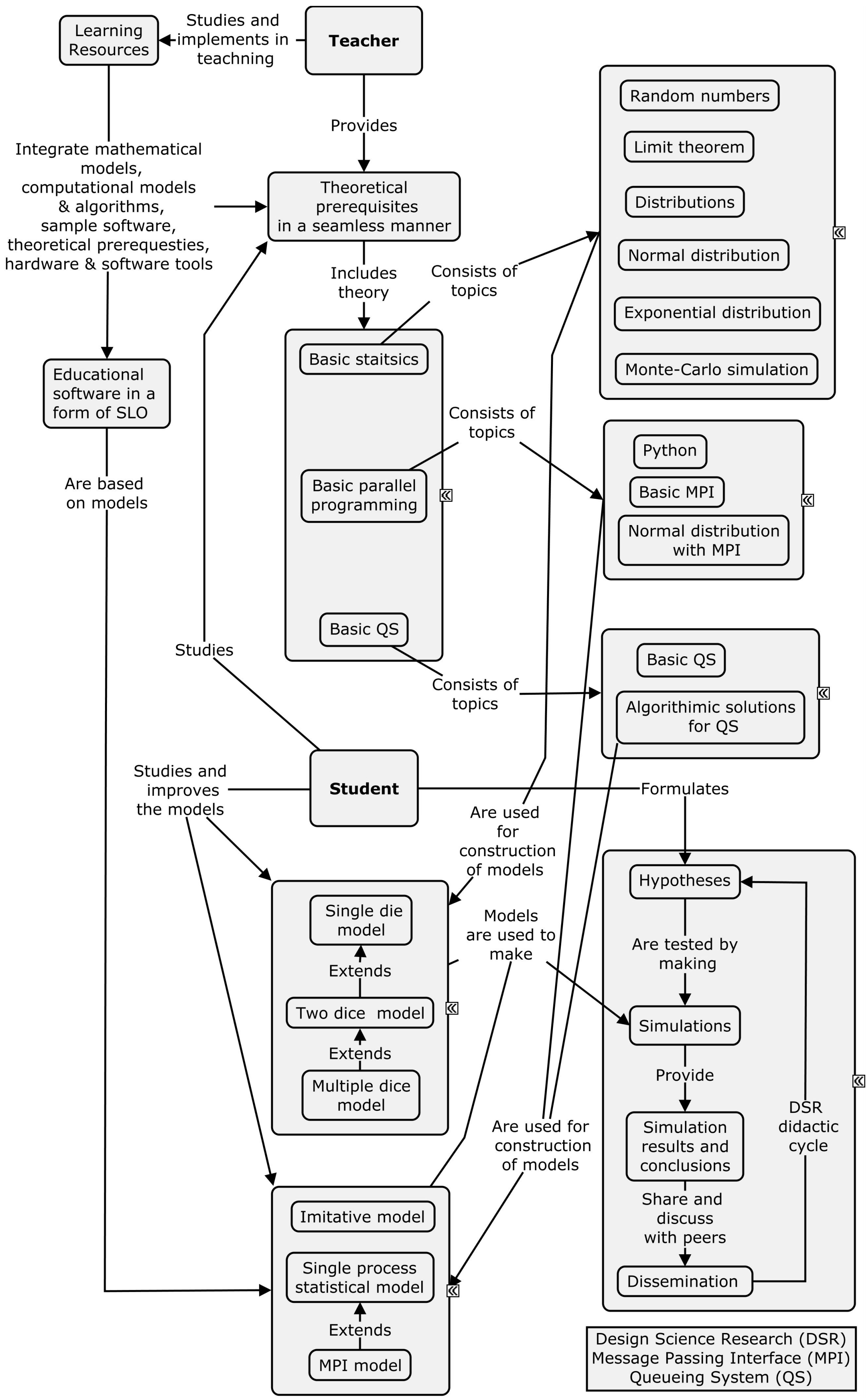
6.4.10 Conclusions

6.4.10.1 Introduction

In this section, a number of models for model-centered learning are provided. These models enable the learner to conduct a series of experiments and enhance the understanding of the discipline in the study. There are several difficulty levels of the presented models and experiments with such models. The first level is the basic one. It introduces randomness and enables primary understanding of the scope of the scientific research. The next one is more sophisticated and enables a deep understanding of parallel programming and stochastic simulations. The relevant theoretical knowledge is provided on demand and as supporting material for the learner's activities. This provides a constructivist framework for the model-centered introduction to the topic. Finally, we would like to provide recommendations for further study and improvements of the models.

6.4.10.2 Linearity of the model and statistical parameters of the queueing system

The model of the system of queues in series provided in this study is not linear [45]. It is obvious from the recurrent equation since it contains a nonlinear mathematical function *max*. If we want to obtain correct modeling results, especially in the case of calculating the statistical parameters of



Learning Resources
Studies and implements in teachning
Teacher
Provides
Integrate mathematical models, computational models & algorithms, sample software, theoretical prerequesties, hardware & software tools
Theoretical prerequisites in a seamless manner
Includes theory
Consists of topics
Random numbers
Limit theorem
Distributions
Normal distribution
Exponential distribution
Monte-Carlo simulation
Educational software in a form of SLO
Basic staitsics
Consists of topics
Basic parallel programming
Python
Basic MPI
Normal distribution with MPI
Are based on models
Basic QS
Studies
Consists of topics
Basic QS
Algorithimic solutions for QS
Studies and improves the models
Student
Formulates
Are used for construction of models
Hypotheses
Single die model
Models are used to make
Extends
Are tested by making
Two dice model
Simulations
Extends
Multiple dice model
Provide
DSR didactic cycle
Are used for construction of models
Simulation results and conclusions
Share and discuss with peers
Imitative model
Single process statistical model
Dissemination
Extends
MPI model
Design Science Research (DSR)
Message Passing Interface (MPI)
Queueing System (QS)

Figure 99: The use case model of the presented learning resources for teaching introductory stochastics

the queueing system, we must use a partially linear model for calculations. This is particularly important for non-heavy traffic systems as in this case we could make rather great mistakes in calculations.

6.4.10.3 Extensions of Python modules and parallel programming with C

For the skillful learners, it could be interesting to proceed with improvements in the efficiency of the programming code. That could be done by extending Python modules with C implemented functions using the SWIG technology. Learners could improve the code and speed-up calculations using Cython or C programming languages, "real" MPI technology and High Throughput Computing (HTC) cluster possibilities [239–241].

6.4.10.4 Efficiency of the programming solutions and further work

In this section, the learner could study the efficiency of various programming solutions. This topic is particularly important for any programming model, which is based on parallel calculations. In this case, learners could study the effectiveness of different programming models and could try to improve algorithms gradually. The key point here is to investigate the ratio of the amount of information flow and calculations for different programming processes. Such a ratio is important when constructing the most effective programming model for parallel calculations. Another interesting topic is to study possible mappings of the algorithm structure to the HTC cluster structure. As a further task for investigations, the authors consider studies of queueing networks to be introduced to the learner, modeled, and analyzed. The comparatively complex nature of queueing networks and a variety of applications requires more comprehensive programming techniques to be involved. This provides a good basic platform for the introduction of such general programming concepts like inheritance, encapsulation, and polymorphism. On the other hand, the basic theoretical CS constructions needed to be introduced as well. Besides all these, the modeling and statistics simulation of queueing networks requires more advanced probability topics to be presented, more computational resources to be occupied and provide the real SC environment and good motivation for the advanced learner.

6.5 Case study of the model-based approaches to teach scientific computing: C based simulation models for teaching programming and parallelization

6.5.1 System of queues in series and stochastic simulations

The material of this subsection is mainly based on the results of the publication [5]. A generalized system of queues in series consists of a number of servicing phases that provide service for arriving customers. The arriving customers move through the phases step-by-step from entrance to exit. If the servicing phase is busy with servicing the previous customer, the current customer waits in the queue in front of the servicing phase. The extended Kendall classification of queueing systems uses 6 symbols: $A/B/s/q/c/p$ where A is the distribution of intervals between arrivals, B is the distribution of service duration, s is the number of servers, q is the queueing discipline (omitted for FIFO – first in first out), c is the system capacity (omitted for unlimited queues), p is the number of possible customers (omitted for open systems) [233, 232]. For example, $M/M/1$ states for Poisson input, one exponential server, one unlimited FIFO queue, and unlimited customer population. The interarrival and servicing times both are independent random variables. We are interested in the sojourn time of the customer in the system and its distribution. The general schema of the system of queues in series is presented in Figure 100 (reprinted from [5, p. 3]).

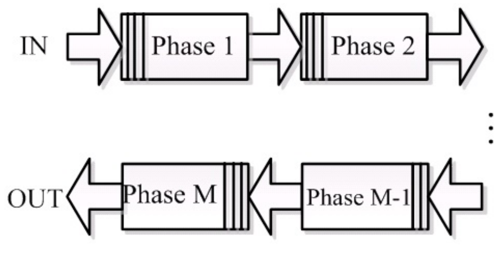


Figure 100: System of queues in series. Reprinted from [5, p. 3]

We consider both interarrival and servicing times as exponentially distributed random variables. Depending on the parameters of the exponential distributions, we distinguish different traffic conditions for the observed queueing system. That includes ordinary traffic, critical traffic, or heavy traffic conditions. We are interested to investigate the distribution of the sojourn time for these different cases [45, 242] and we will use Monte Carlo simulations for collecting the relevant data.

6.5.2 Recurrent equation for the calculation of sojourn time

In order to design the modeling algorithm of the previously described queueing system, some additional mathematical constructions should be introduced. Our goal is to calculate and investigate the sojourn time of the customer number n in the system of queues in series of k phases. We can prove the parametric recurrent equation (15) for calculation of the sojourn time. This enables the implementation of the necessary algorithms since all the basic theoretical results have been introduced.

6.5.3 Theoretical background: Parallel computing

In this research, we emphasize multiple instructions, multiple data MIMD parallel architecture and presume the HPC cluster as a target platform for calculations. Such a platform allows us to study different parallelization techniques and implement share-memory, distributed-memory as well as 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 message parsing MPI tools. For the hybrid decomposition, we use the shared memory (loop decomposition) for the longitudinal decomposition and MPI for the pipeline decomposition.

6.5.4 Parallelization for the systems of queues in series

Statistical sampling for modelling the sojourn time distribution (Figure 101, reprinted from [5, p. 4]) presents the general schema of the imitational experiment on the queueing system.

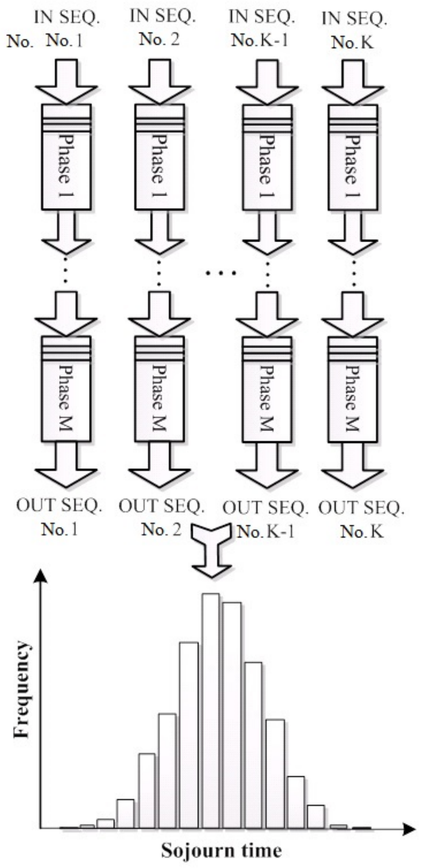


Figure 101: Statistical sampling for modelling of distribution of the sojourn time. Reprinted from [5, p. 4]

The programming model of the system of queues in series is based on the recurrent equation, presented in one of the upper sections. The Monte Carlo simulation method is used to obtain the statistical sampling for modeling the sojourn time distribution on the exit of the system. To introduce the topics as decomposition and granularity we present a space model of the simulation process of the queueing system. First, we start with a one-dimensional model. The one-dimensional model allows introducing a sequential programming model with no parallelism and could serve as a basic model for further improvements. Afterward, we proceed with a two-dimensional model. Such a model allows introducing the programming models for the longitudinal decomposition. As the last step, we introduce a three-dimensional space model. Such a model allows constructing the programming models for the transversal and hybrid decompositions.

6.5.5 Stochastic simulations and longitudinal decomposition

One of the solutions is to use the longitudinal decomposition (trials) and to parallelize the Monte Carlo trials. Thus, we can map each or a group of trials depending on the preferred granularity and the total number of desired trials. The schema of the longitudinal decomposition is presented in Figure 102 (reprinted from [5, p. 7]). A three dimensional model of the longitudinal decomposition is presented in Figure 103 (reprinted from [5, p. 7]).

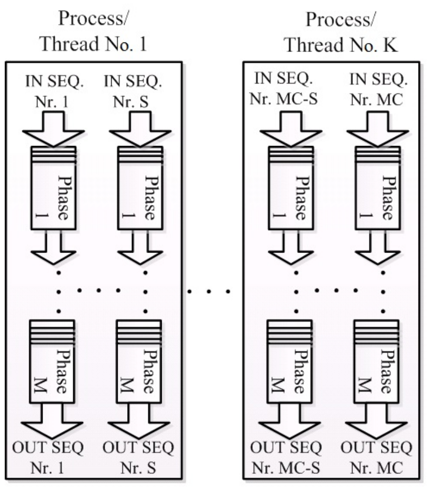


Figure 102: Longitudinal decomposition. Reprinted from [5, p. 7]

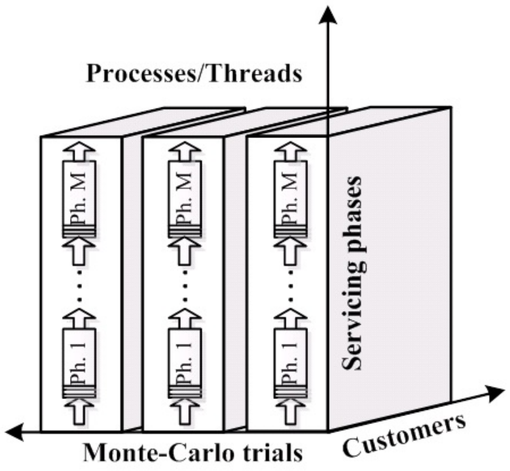


Figure 103: Three dimensional model of the longitudinal decomposition. Reprinted from [5, p. 7]

6.5.6 Pipelining and transversal decomposition

In another dimension (customers), the parallelization technique is not as straightforward as it was in the previous case of parallelization of the statistical trials dimension. There arises a difficulty as we have the pipelining structure of the algorithm. This is obvious: the customer moves through the system from the previous to current servicing phase; we need to have all the data from the previous stage for calculations at the current stage. We use the transversal decomposition and the number of customers in each stage depends on the preferred granularity and the total number of customers. Figure 104 (reprinted from [5, p. 7]) presents the decomposition in the case of the customer's dimension.

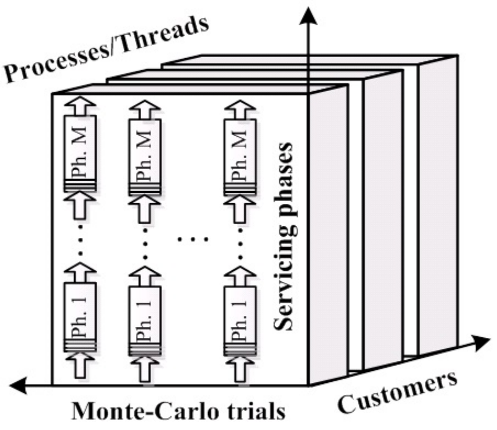


Figure 104: Transversal decomposition. Reprinted from [5, p. 7]

6.5.7 Shared memory implementation

The shared memory implementation is based on the OpenMP tools. The loop parallelization technique is used for the longitudinal decomposition. For the transversal decomposition, the OpenMP tasking model and dynamic scheduling are used.

6.5.8 Distributed memory implementation

The distributed-memory implementation is based on MPI tools. In both cases, i.e. longitudinal and transversal decompositions, the message-parsing interface provides synchronization tools and there is no need for additional programming constructions.

6.5.9 Hybrid models and HPC

Hybrid models provide a natural solution to computational platforms, based on high-performance computer clusters. It uses MPI tools to perform a decomposition and OpenMP tools for multithreading.

6.5.10 Dynamic and static scheduling

Pipelining requires dynamic scheduling since there is a connection between various nodes in the pipeline. In the shared-memory case, we must take care of scheduling. If we use the OpenMP tasking model, the relevant approach could be twofold. First, it is possible to obtain a dynamic scheduling by monitoring a critical shared-memory resource and using a task yielding construction. The other one is to use the dynamic task creation technique. In the case of MPI, synchronization is performed by the interface system, and then there is no need for additional programming constructions.

6.5.11 Sequential programming model

The flowchart of the sequential program model is presented in Figure 105 (reprinted from [5, p. 8]). The algorithm uses the recurrent equation and cycles for modeling the queueing system phases, customers, and statistical trials.

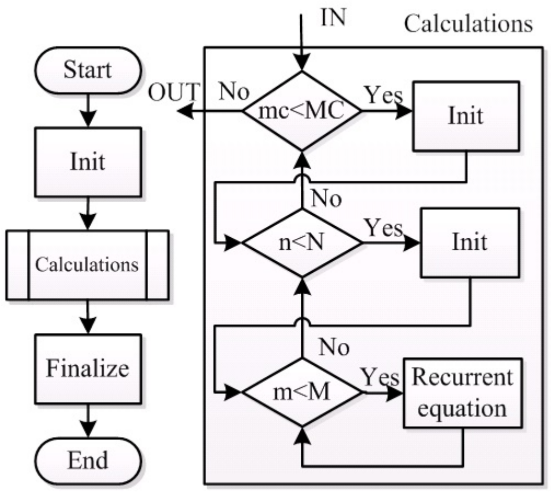


Figure 105: Sequential mode. Reprinted from [5, p. 8]

The program model of the sequential approach uses the programming language C and GSL (GNU scientific library) and it is presented in Appendix A.1 of this thesis including the comments.

6.5.12 Programming model for distributed-memory parallelization

6.5.12.1 Longitudinal decomposition

The programming model for the distributed memory parallelization is based on MPI tools and it is optimal for the multicore/multimode computer architecture. All the processes receive a full copy of the programming code and the rooting is made by using the number of the process. The flowchart of the longitudinal decomposition is presented in Figure 106 (reprinted from [5, p. 8]). The programming language C model with the comments is presented in Appendix A.2.

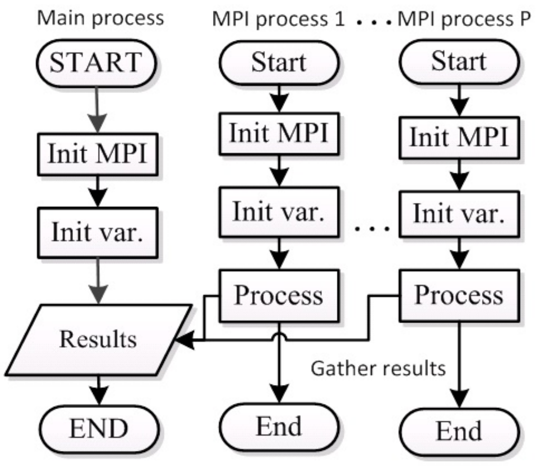


Figure 106: Distributed memory longitudinal decomposition. Reprinted from [5, p. 8]

6.5.12.2 Transversal decomposition

The flowchart for the transversal decomposition is presented in Figure 107 (reprinted from [5, p. 8]). Processes are attached to the customer's axis, which is divided into chunks. We use a mutual message parsing technique and MPI is responsible for scheduling. In order to provide the desired granularity, statistical trials are divided into the relevant chunks. The model coded in programming language C with comments is presented in Appendix A.3.

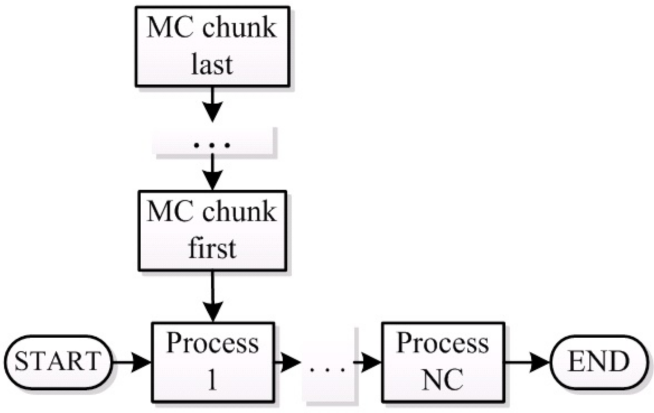


Figure 107: Distributed memory transversal decomposition. Reprinted from [5, p. 8]

6.5.13 Programming model for shared-memory parallelization

6.5.13.1 Longitudinal decomposition

In the case of the shared memory model, the OpenMP loop parallelization is the natural solution to the longitudinal decomposition. The scope of Monte Carlo trials is divided into chunks and each of such chunks is attached to the relevant thread. The flowchart of the shared memory model is presented in Figure 108 (reprinted from [5, p. 12]).

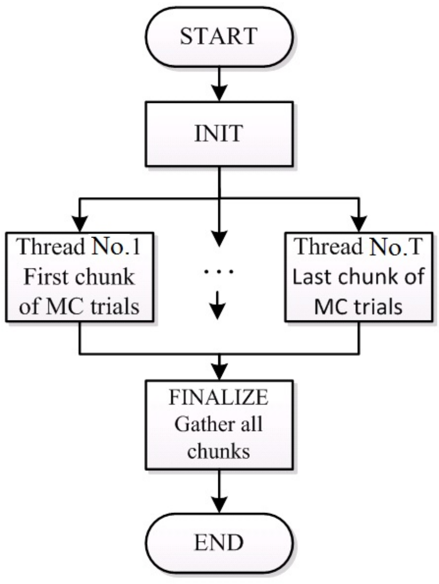

START
INIT
Thread No.1 First chunk of MC trials
. . .
Thread No.T Last chunk of MC trials
FINALIZE Gather all chunks
END

Figure 108: Shared memory longitudinal decomposition. Reprinted from [5, p. 12]

The programming model for the shared-memory longitudinal decomposition uses the programming language C, GSL (GNU scientific library), and

OpenMP libraries. The model and comments are presented in Appendix A.4 of this thesis.

6.5.13.2 Transversal decomposition

One of the most comprehensive programming models is the model of the shared-memory pipelining. We use the transversal decomposition to construct such a model. The model uses the OpenMP tasking technique and dynamic scheduling of tasks. The scheduler plays the central role in the model and it is responsible for creating new tasks and finishing the program, after completing all the tasks. Each task uses its own random generator, which allows avoiding time-consuming critical sections. The flowchart is presented in Figure 109 (reprinted from [5, p. 12]). The programming model for the shared memory transversal decomposition uses C programming language, GSL (GNU scientific library), and OpenMP libraries. The model and comments are presented in Appendix A.5 and Appendix A.6 of this thesis.

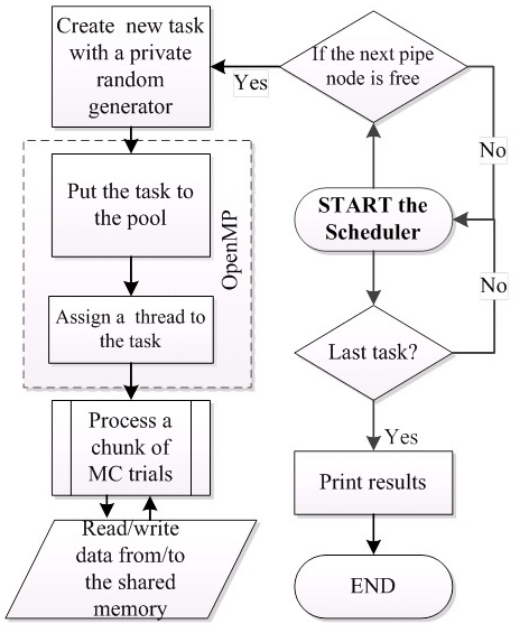

Create new task with a private random generator
If the next pipe node is free
Yes
No
Put the task to the pool
OpenMP
START the Scheduler
No
Assign a thread to the task
Last task?
Yes
Process a chunk of MC trials
Print results
Read/write data from/to the shared memory
END

Figure 109: Shared memory transversal decomposition. Reprinted from [5, p. 12]

6.5.14 Programming model for hybrid parallelization

The MPI transversal decomposition model could be transformed into a hybrid model by adding the OpenMP threads to the OpenMP trials axis. The flowchart of the hybrid model is presented in Figure 110 (reprinted from [5, p. 12]). The programming model with comments is presented in Appendix A.7.

6.5.15 Use case model

The use case model of the presented LR in the form of a concept map is presented in Figure 111. Design Science Research considerations concerning the presented use case model are discussed in detail in sections 4.2.1.4 and 4.2.2 of this thesis.

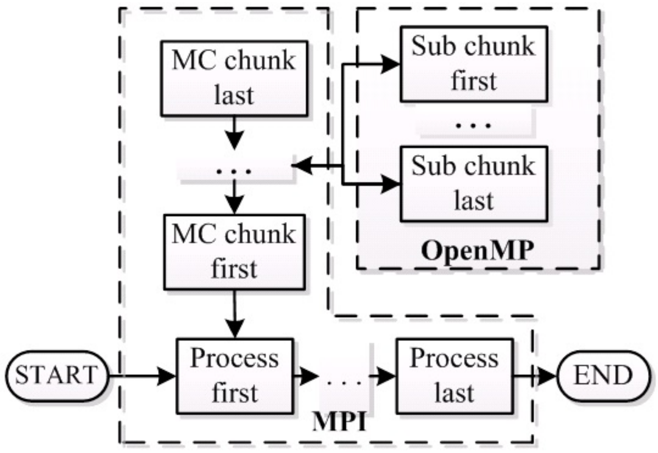


Figure 110: Hybrid decomposition. Reprinted from [5, p. 12]

6.5.16 Conclusions

6.5.16.1 Theoretical and programming models: The basis of the model-centered approach

The study provides a number of programming models for the introduction to scientific and parallel computing. All these programming models: sequential, distributed-memory, distributed-memory pipelining, shared-memory, shared-memory pipelining, and the hybrid model are based on statistical simulations of the theoretical model of the system of queues in series. After providing a theoretical background to the learner and explaining the main features of the theoretical model, we start experiments with programming models. The relevant problems could be provided to the learners. These could include the comparative investigation of the effectiveness of programming models, taking into account different computational platforms as well as different input parameters of the queueing system. For the advanced learner, the emphasis could be put on a variation of the parameters of

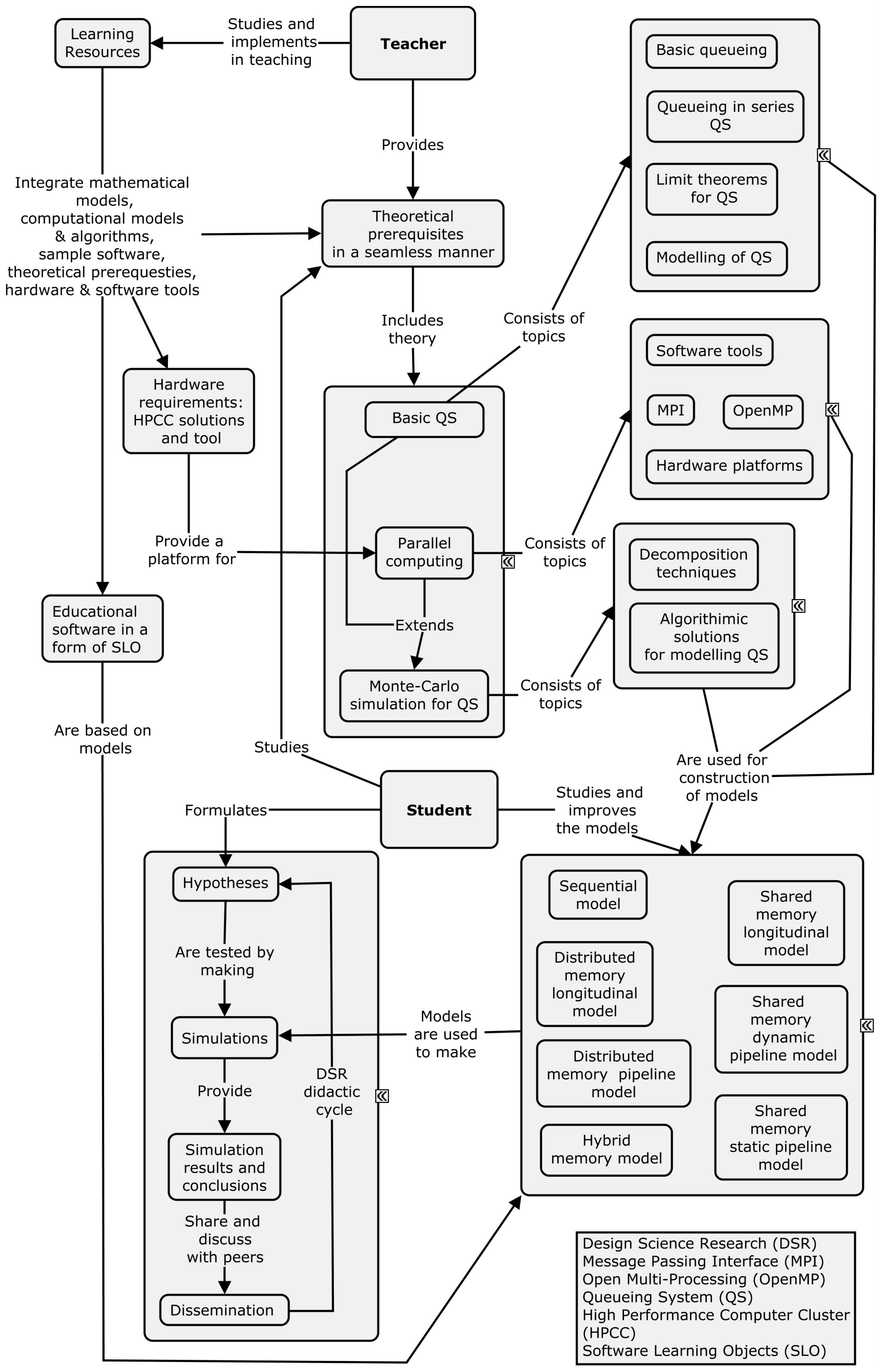

Learning Resources
Studies and implements in teaching
Teacher
Provides
Integrate mathematical models, computational models & algorithms, sample software, theoretical prerequesties, hardware & software tools
Theoretical prerequisites in a seamless manner
Includes theory
Consists of topics
Basic queueing
Queueing in series QS
Limit theorems for QS
Modelling of QS
Hardware requirements: HPCC solutions and tool
Basic QS
Software tools
MPI
OpenMP
Hardware platforms
Provide a platform for
Parallel computing
Consists of topics
Decomposition techniques
Algorithimic solutions for modelling QS
Extends
Educational software in a form of SLO
Monte-Carlo simulation for QS
Consists of topics
Are based on models
Studies
Student
Are used for construction of models
Formulates
Studies and improves the models
Hypotheses
Are tested by making
Simulations
Models are used to make
Provide
DSR didactic cycle
Simulation results and conclusions
Share and discuss with peers
Dissemination
Sequential model
Shared memory longitudinal model
Distributed memory longitudinal model
Shared memory dynamic pipeline model
Distributed memory pipeline model
Hybrid memory model
Shared memory static pipeline model
Design Science Research (DSR)
Message Passing Interface (MPI)
Open Multi-Processing (OpenMP)
Queueing System (QS)
High Performance Computer Cluster (HPCC)
Software Learning Objects (SLO)

Figure 111: The use case model of the presented learning resources for teaching programming and parallelization

inter-arrival and servicing time exponential distributions, moving from the heavy traffic to the non-heavy traffic case, since that could fundamentally change the distribution of the sojourn time of the customer.

6.5.16.2 Introduction to scientific computing: Research tasks, research methods

While investigating the theoretical model, studying the recurrent equation, and experimenting with the input parameters of the queueing system, we introduce the scientific research tasks and methods. It includes studying the distribution of the sojourn time of the customer, varying the parameters of inter-arrival and servicing time exponential distributions, comparing the results, analyzing the provided theoretical constructions as well as studying of the Monte Carlo method for statistical simulations, which is one of the basic methods in studying the topics related with probability.

6.5.16.3 Introduction to parallel computing: Terminology and methodology

Studying and experimenting with the programming models, the introduction to parallel computing terminology and methodology is provided. It includes the basic concepts such as shared and distributed-memory parallelization techniques, homogenous and heterogeneous computational platforms, HPC and multicore programming and it explains scheduling, mapping, and granularity. Using MPI and OpenMP tools, we introduce OpenMP tasking, MPI programming methods, synchronization, load balancing, decomposition techniques and other important topics of parallel computing.

6.5.16.4 Problems: Studying effectiveness, debugging and benchmarking

We could enhance the learner's understanding by providing a set of problems such as debugging, benchmarking, and comparative studying of the effectiveness of programming models. That includes the variation of different computing platforms for one of the models as well as testing different models for a definite platform. It could be done by applying a single processor, multicore, multiprocessor machines and computer clusters. As an example, the model with efficient results achieved by a single-processor machine could be inefficient on other platforms. When modifying the model, the relevant debugging tools and methods must be implemented. So the learner could proceed by modifying the model i.e. changing the distribution parameters and tuning the granularity.

6.5.16.5 Further studies: Queueing networks for constructing of learning objects

The theory of queueing systems renders wide possibilities for relevant theoretical constructions. The next obvious step could be studies of queueing networks of various types including open, closed or mixed networks, constructing the relevant learning objects, and investigating the respective theoretical and programming models.

6.6 Conclusions

This section provides a comprehensive review of experimental applications of stochastic recurrences, applied to modeling of the system of queues in series under various stochastic parameters. The developed theory for modeling of the system of queues in series allows experiments with big data applications to be implemented. At the same time, this real-live scientific application provides a suitable background for learning objects for teaching scientific computing to be designed and implemented. Such LO are implemented in the form of programming models which implement SLOs for topics in the study. The model-centered approach allows implementing a universal platform for teaching introductory statistics and parallelization.

7 Evaluation of the developed educational solutions

7.1 Specification of the developed sample learning resources

We propose expert evaluation of the developed Design Principles and the Supportive Application and Integration Methodology (DPSAIM). The evaluation based on DSR requirements was done in Sections 1.5 and 1.6 of the research. This additional evaluation is aimed at providing an expert view on the developed DPSAIM in the form of expertise from educational technology and teaching and LR design perspectives.

To evaluate the developed educational solutions we propose to implement the indirect method for evaluation – to evaluate the provided sample LR as cognitive artefacts – the output of the design process using the developed DPSAIM: LR and SLOs for Teaching Introductory Stochastic (SLOFTIS) and LR and SLOs for Teaching Advanced Parallelization (SLOFTAP). The provided LR could be considered as a set of teaching/learning material, including SLOs in the form of programming models and supportive material in the form of methodology including the relevant constructs, models, and frameworks. The supportive methodology has to be specified as:

(C) Content: SLOFTIS see Section 6.4; SLOFTAP see Section 6.5;
(I) Instruction: See Fig. 39;
(D) Didactics: See Fig. 38;
(T) Technology: See Fig. 39, Fig. 49.

Below we provide a more detailed specification of the designed sample resources (see Table 8 and Table 9).

Table 8: Description of SLOFTIS parameters

No	Description	Implementation
1	Constructs	(R) Randomness: random numbers; random number distributions; random numbers generators; Central Limit Theorem. (P) Python programming constructions: decorators; coroutines; yield expressions. (T) Results that are more complex include theoretical facts such as: queueing in series specifications and parameters like the sojourn time of the customer; the recurrent equation for calculating the sojourn time; stochastic simulation methods and multiprocessing techniques. (I) SI specific concepts: theoretical facts to be studied; conceptual model; mathematical model; algorithms and programming constructions; computational model; stochastic simulation and observation of simulation results.

Table 8: Description of SLOFTIS parameters

2	Models & SLO	Models and the relevant SLO: For introductory stochastics: Python single die model; Python multi-dice model; Python model for exponential distribution; For Python multiprocessing: Python model for the advanced normal distribution with MPI; For stochastic recurrences (the system of queues in series): Imitative model based on multiprocessing of services; Simulative model with Python yield expressions; Simulative model with Python MPI;

Table 9: Description of SLOFTAP parameters

No	Description	Implementation

1	Constructs	(Q) Queueing in series systems: Queueing basics; Queueing parameters; Monte Carlo experiment;
		(C) Parallel computing: hardware platforms; software tools;
		(S) Stochastics: stochastic recurrence, algorithmic solutions for queueing in series systems;
		(M) Modelling techniques: longitudinal system decomposition; transversal system decomposition;

2	Models & SLO	Sequential programming model; Distributed memory programming model; Distributed memory pipeline model; Shared memory programming model; Shared memory static pipeline programming model; Shared memory dynamic programming model; Hybrid programming model.

7.2 Evaluation scheme

The developed LR consist of an interactive part in the form of SLOs and supportive educational resources. The evaluation scheme could be designed based on the evaluation of the interactive part – SLOs and could be based on the presented general approaches to evaluation of interactive LO [243] with some extent as related to SLOs. We propose the next evaluation scheme as presented in Table 10.

Table 10: DPSAIM evaluation scheme

No	Name	Description	Literature
	To evaluate the developed LR from the perspectives of:		
1	Compatibility with the objectives of the LO design	Reusability; Interoperability; Durability; Accessibility	[244]
2	LO taxonomies	Redeker; Finlay; Churchill	[245], [246],[138]
3	LO standards	IEEE Standard for Learning Object Metadata (IEEE Std 1484.12.1 - 2002)	[247]
4	Application of quality measurement metrics	Defude & Farhat: **M1** – number of concepts in the LO content; **M2** – upper level concept - lower level concept (in content); **M3** – number of concepts in LO prerequisite; **M4** – upper level concept - lower level concept (in prerequisite); **M5** – number of concepts with educational state set to “high” in the LO prerequisites; **M6** – number of items set to null or default value in the educational characteristics; **M7** – number of nodes at the concrete level of the composition graph; **M8** – number of occurrences of the LO into existing complex LO; **M9** – number of levels between the abstract level and the concrete level of the composition graph; **M10** – number of delivering graphs of the LO; **M11** – number of ALT nodes and of query nodes (for intentional LO)	[248]
5	Possible disadvantages or limitations		
6	Possibility of further development		

7.3 Evaluation results

7.3.1 Evaluation from the perspectives of compatibility with the objectives of the LO design

The evaluation from the perspectives of compatibility with the objectives of th LO design is presented in Table 11

Table 11: Compatibility with the objectives of the LO design

No	Description	Requirements	Evaluation
1	Reusability	Learning content modularized into small units of instruction suitable for assembly and reassembly into a variety of courses	Could be used interchangeably or as the whole set depending on the grade and curriculum requirements. SLOFTIS include 3 programming models in the form of SLOs; SLOFTAP Include 6 programming models in the form of SLOs.
2	Interoperability	Instructional units that interoperate with each other regardless of developer or learning management system	Interoperation is set into the structure of the LR and is based on the model-centered approach. The central model is the model of the Monte Carlo experiment for the system of queues in series.
3	Durability	Units of instruction that withstand ever evolving delivery and presentation technologies without becoming unusable	SLOs could be used with different technologies by using the relevant system dependent translators or compilers.
4	Accessibility	Learning content that is available anywhere, any time-learning content that can be discovered and reused across networks	SLOs could be distributed via GitHub technology including the supportive material.

7.3.2 Evaluation from the perspectives of LO taxonomies

The evaluation from the perspectives of LO taxonomies is presented in Table 12.

Table 12: Compatibility with LO taxonomies

No	Description	Correspondence to the Requirements	Evaluation
1	Redeker	Internally interactive learning objects integrate the learner in the human-computer-interaction. The learner is either Competency based training (CBT) guided – albeit in a mini-CBT sequence – or is given the framework of his or her activities via simulation	SLOFTIS, SLOFTAP
2	Finlay	Application-based, which require the learner to respond with some reasoned action	SLOFTIS, SLOFTAP
3	Churchill	Simulation object: Representation of some real-life system or process; Conceptual model: Representation of a key concept or related concepts of subject matter; Information object: Display of information organized and represented with modalities	SLOFTIS, SLOFTAP

7.3.3 Evaluation from the perspectives of LO standards

The evaluation from the perspectives of LO standards is presented in Table 13.

Table 13: Compatibility with LO standards

No	Name	Explanation (as in standard)	Value Space (as in standard)	Explanation (as related to SLOs)
1.7	Structure	Underlying organizational structure of this learning object	1: collection – a set of objects with no specified relationship between them	SLOFTIS, SLOFTAP

Table 13: Compatibility with LO standards

1.8	Aggregation Level	The functional granularity of this learning object	2: a collection of level 1 learning objects, e.g., a lesson; 3: a collection of level 2 learning objects, e.g., a course.	2. If SLO is a lesson unit 3. If a set of SLO is a course unit
5.1	Interactivity Type	Predominant mode of learning supported by this learning project	active	Simulations making based active learning
5.3	Interactivity Level	The degree of interactivity characterizing this learning object. Interactivity	high very high	Simulations making based interactive learning
5.4	Semantic Density	The degree of conciseness of a learning object	high very high	Educational software with high level semantic density
5.5	Intended End User Role	Principal user(s) for which this learning object was designed, most dominant first	teacher learner	Could be used by a teacher (designer) of or by a learner
5.6	Context	The principal environment within which the learning and use of this learning object is intended to take place	school higher education	Advances secondary (SLOFTIS) and innovative university education (SLOFTAP)
5.7	Typical Age Range	Age of the typical intended user	18-	Advances secondary and innovative university education
5.8	Difficulty	How hard it is to work with or through this learning object for the typical intended target audience	easy medium difficult	SLOFTIS – easy, medium; SLOFTAP – difficult.

7.3.4 Evaluation from the perspectives of quality measurement metrics

The evaluation from the perspectives of quality measurement metrics is presented in Table 14.

Table 14: Compatibility with quality measurement metrics of the LO design

No	Description	Metrics	Evaluation
Q1	quality of the content	M1,M2	M1: number of concepts in the SLOFTIS, SLOFTAP content – high; M2: upper level concept – lower level concept (in content): middle
Q2	quality of the LO considering prerequisites	M3, M4, M5	M3: number of concepts in SLO prerequisites: SLOFTIS – low, SLOFTAP – middle; SLO corresponds to seamless approach to theoretical prerequisites; M4: upper level concept – lower level concept (in prerequisites): middle; M5: number of concepts with educational state set to "high" in the LO prerequisites: SLOFTIS – low; SLOFTAP – middle;
Q3	quality of the description	M6	Sample SLOs are presented as a demonstration of the developed DP. The supportive methodology of application and integration is provided. The number of default values is low
Q4	quality of the LO considering reusing	M1, M7, M8	A model-centered approach allow LR to be reused in a flexible sequence
Q5	quality of the LO considering structural complexity	M7, M9	Structural scheme for SLOFTIS – linear, SLOFTAP – star
Q6	quality of the LO considering adaptation	M10, M11	Both high

7.3.5 Evaluation from the perspectives of possible disadvantages or limitations

The next possible difficulties could be specified:

- Possible difficulties of practical adaptation and integration: although model-centered approach provides adaptive features for developed solutions, this practical adaptation could require a certain level of competences from the educator or designer of the curriculum topics;
- Possible difficulties of implementation: as explicit modeling solutions are platform dependent, there is a need for the relevant hardware platforms as, for example, HPC clusters to be involved into educational process.

7.3.6 Evaluation from the perspectives of possibility of furthe development

The possibility for further development include possible improvements for models, technologies, methods of instruction, educational technology as is specified in Section 4.5.4. In addition, further development could be as follows: there is no cost-calculus developed for the efficiency of provided programming models; more comprehensive models like programming models and the relevant SLOs for queuing networks could be developed; implicit prallelization models and the relevant LR could be studied in detail and developed.

7.4 Expert evaluation

7.4.1 Questionnaire

The evaluation provided by experts is based on the high-level expertise in the field of design, implementation, application and integration of educational resources. To provide expertise the next questions are to be answered:

(G) Evaluation of the developed educational solutions in general
   (1) What are the advantages and perspectives (if any) of the developed Design Principles and the Supportive Application and Integration Methodology (DPSAIM)?
   (2) Does the provided evaluation scheme correspond to the evaluation of the developed DPSAIM?
   (3) What is the general opinion or comments (if any) on the research?

(E) The evaluation of the developed sample resources (SLOs)
   (1) Are the developed LR – DPSAIM – compatible with the objectives of the DPSAIM design?

    (2) Does the developed SLOs meet LO taxonomies?
    (3) Does the developed SLOs meet the most important characteristics defined in the LO standards?
    (4) Can standard quality measurement metrics for the developed SLOs be used?
    (5) What are the disadvantages or limitations of the developed SLOs?
    (6) What could be further development of the solutions outlined in the research?

(A) Additional evaluation topics
    (1) What are (if any) additional evaluation criteria?

(S) Supplement expertise
    (1) What are (if any) supplement expert opinions (advantages, disadvantages) on the provided educational solutions in general and possible implementations in particular?
    (2) What are (if any) supplement expert opinions (advantages, disadvantages) on the provided sample LR?

The sample form of questionnaire for expert evaluation is provided in Appendices B.1.

7.4.2 Evaluation Summary

Evaluation summary is provided in Appendices B.2. The expert evaluation was provided by a number of experts. Experts indicated their expertise in various fields of informatics.

7.4.2.1 Evaluation of the expertise level

(1) As a field of the research interests experts indicated: Didactics, informatics education – 100%; Technology (software tools, engineering) – 80%; Programming (languages, solutions, tools) – 80%; Didactics, teachers' training – 60%; Educational solutions in general – 60%; Instruction (design, application) – 40%; Theory (algorithms, mathematical models) – 20%; Technology (hardware, platforms) – 20%.

(2) As an area of teaching, training: Programming (languages, solutions, tools) – 80%; Technology (software tools, engineering) – 80%; Programming (languages, solutions, tools) – 80%; Didactics, informatics education – 80%; Didactics, teachers' training – 80%; Educational so-

lutions in general – 60%; Instruction (design, application) – 40%; Theory (algorithms, mathematical models) – 20%; Technology (hardware, platforms) – 20%.

(3) As an area of educational material application, development, adaptation: Technology (software tools, engineering) – 100%; Didactics, informatics education – 100%; Programming (languages, solutions, tools) – 80%; Didactics, teachers' training – 80%; Educational solutions in general – 60%; Instruction (design, application) – 60%; Theory (algorithms, mathematical models) – 20%; Technology (hardware, platforms) – 20%.

(4) Experts indicated the experiences level in the research experience more than 10 years – 80% and more than 4 years – 20%. The indicated teaching experience level is more than 10 years for all experts. The number of research items for 100% of experts is more than 15 reviewed research publications.

The conclusion on the level of the expertise: all experts could provide a high level of expertise in the field of informatics and informatics engineering education. We could describe the generalized portrait of the experts as high competence experts in the filed of informatics, informatics engineering and programming education.

7.4.2.2 Evaluation of the developed educational solutions in general

(1) As an advantages and perspectives of the developed DPSAIM all the experts indicated:
Practically applicable for novice university level – 100%; Improves learner understanding – 100%; Could be included into teacher's toolkit – 100%; Provides constructionist educational toolkit – 100%; Covers important educational topics – 100%; The most of the experts experts indicated: Provides solutions for modern educational issues – 80%; Practically applicable for high secondary level – 80%; Could be further developed by the teacher – 80%.

(2) All experts indicated the correspondence of the provided evaluation scheme to the evaluation of the developed DPSAIM – 100%.

(3) All (100%) of the experts indicated a general opinion or comments

on the research as Agree or Strongly agree for the next items related to the research: Correspondence of the internal structure; The relevance of the research topic; The correspondence of the implemented research methodology; The novelty of the developed educational solutions; Correspondence of the theoretical grounding; The relevance and correspondence of the experimental research and case studies; Correspondence of the literature review. The major part of experts (80%) indicated: The novelty of the developed technological solutions; the correspondence of the volume of the research.

The conclusion on the evaluation of the developed educational solutions in general: experts provided highly positive evaluation of the developed educational solutions. The general level of positive evaluation is 94.4%.

7.4.2.3 Evaluation of the developed sample resources

(1) Experts indicated the compatibility of the developed DPSAIM with the objectives of the design: Reusability – 100%; Interoperability – 100%; Durability – 100%; Accessibility – 100%.

(2) Experts indicated the compatibility with Churchill's taxonomy as follows. Experts indicated: Simulation object – 100%; Contextual representation – 100%; Information object – 100%; Practice object – 80%; Presentation objects Practice object – 80%; Conceptual model Practice object – 80%.

(3) Experts indicated the compatibility with Redeker's taxonomy as follows. All experts indicated: Internally interactive – 80%. Some experts indicated: Receptive – 80%; Cooperative – 80%.

(4) Experts indicated the compatibility with Finlay's taxonomy as follows. All experts indicated: Application-based – 100%; Individualised – 100%. Some experts indicated: Theory-based – 80%; Cooperative – 40%.

(5) Experts provided the evaluation from the perspectives of quality measurement metrics as follows. All experts indicated: high level for quality of the content (Q1); from middle to high level for: quality of the LO considering prerequisites(Q2), quality of the description (Q3), quality of the LO considering reusing (Q4), quality of the LO considering structural complexity (Q5), quality of the LO considering adaptation (Q6).

The conclusion on the evaluation of the developed sample resources: the developed learning resources are compatible with LO taxonomies; quality measurement metrics is applicable for the developed LR, all experts evaluated the sample resources from middle to high.

7.4.2.4 Evaluation of the developed sample resources from requirements indicated in learning objects

(1) The experts evaluated the underlying organizational structure of this learning object (1.7) as atomic – 60%; collection – 40%; hierarchical – 60%. None of the experts evaluated as linear – 0%.
(2) The functional granularity of this learning object (1.8). The experts evaluated it as the smallest level of aggregation – 40%; a collection of level 1 learning objects – 80%; a collection of level 2 learning objects – 60%. Some experts evaluated as the largest level of granularity – 20%.
(3) Interactivity Type (5.1). The experts evaluated as active interactivity – 60%; mixed interactivity type – 80%. None of the experts evaluated as expositive interactivity.
(4) Interactivity Level (5.3). Most of the experts evaluated it as high or very high – 80%; Some experts evaluated as middle – 20%.
(5) Semantic Density (5.4). All experts evaluated as high – 100%.
(6) User Role (5.5). The experts indicated a user role as a teacher – 60%; an author – 20%; a learner – 80%; a manager – 20%.
(7) Context(5.6). The experts indicated: higher education – 60%; school level – 60%; training – 60%.
(8) Typical Age Range(5.7). Most experts indicated 18 and more years (novice university) – 60%; some experts indicated 15-16 years (primary school) – 40%; 16-18 years (secondary school) – 40%; 12-14 years (primary school) – 20%; 7-11 years (primary school) – 20%.
(9) Difficulty (5.8). Most experts indicated middle level – 80%. Some experts indicated high level of difficulty – 20%;

The conclusion on the evaluation of the developed sample resources from requirements indicated in LOs standards: the developed learning resources could be evaluated in accordance with the requirements of the LOs standards. It was possible for all experts to evaluate the provided educational solutions using the requirements of the standard.

7.4.2.5 Evaluation from the perspectives of possible disadvantages or limitations and possibility of further development

(1) Most experts indicated a level of possible possible disadvantages or limitations from middle to low – 80%.
    All experts indicated the high or very high level of possibility of further development – 100%.

Conclusion: there is a need of further possible improvements of the developed LR and additional research in the field of study.

7.4.2.6 Additional evaluation topics

(E) Some experts indicated the need of additional evaluation criteria:
   (1) Pedagogical experiment in real setting: students' engagement evaluation according to Bloom's (revised Bloom's) taxonomy, students' computational thinking skills evaluation.
   (2) Applicability and efficiency of the LO in the real context (teaching environment), learning outcomes measurement.
   (3) Extensive set of evaluation criteria has already been presented by the author.

(S) Some experts indicated a supplement expert opinion (advantages, disadvantages, opinion, recommendations, etc.) on the provided educational solutions and sample learning resources:
   (1) In my opinion, SLOs are very useful and effective in modeling real-world phenomena that are difficult to illustrate by real experiments; as a tool to develop computational thinking skills; as introduction to artificial intelligence concept. The use of SLOs can be started at secondary school level not only in computer science, but also in mathematics, social (economics) and natural (physics, biology, chemistry) sciences. The SLOs can play the important role in inquiry-based learning and enlarge the possibilities of STEM at schools. Potential difficulties are associated with the preparation of teachers to incorporate SLOs into education.
   (2) Valuable solutions, very well grounded and structured.

Conclusion: There is a very positive expert opinion on the developed LR.

7.4.3 Conclusions

The expert evaluation has been done on various aspects of the developed learning resources, such as formal approaches to evaluation like correspondence to standards, semi-formal approaches like correspondence to taxonomies and quality metrics and personal expert opinion on general quality of the research and quality of developed educational solutions. The level of the provided expertise which is based on scientific and educational experience of experts is high or very high. Concluding, it is possible to state that the provided research results in the form of the developed DPSAIM have very positive expert evaluation.

8 Conclusions

Concluding, as a result of the comprehensive study the next solutions and implementations are proposed:

(1) the constructionist approach for Scientific Computing Education (SCE) is studied in detail and adapted for the needs of the university curriculum in general and STEM university curriculum in particular, focusing on interdisciplinary and research-based education;

(2) the TPACK model, as related to the SCE within the university curriculum, is studied in detail and the relevant meta analysis of domain features is implemented in the form of feature models of various levels. Such study provides an appropriate background for implementation of the research tasks;

(3) the innovative Design Principles and the Supportive Application and Integration Methodology (DPSAIM) for teaching and learning SC, which covers such major parts of the educational system like educational technology, instructional design, didactic tools, and educational approaches is studied in detail and implemented. The proposed methodology allows enhancing university interdisciplinary curricula through problem-solving and research-based educational methods;

(4) the appropriate didactic approach for SC education has been studied in detail and implemented in the form of practical LR. Such LR include a set of programming models and educational tools in the form of SLOs.

The design approach and the structure of the relevant SLO is based on a model-based paradigm, enabling the implementation of the SI centered educational methods;

(5) the implemented comprehensive programming model for the computational study of stochastic recurrences provides a theoretical background for the relevant implementation of the LR. These LR (in the form of programming models and SLOs), besides their focus on such theoretical topics like introductory stochastics, basic probability distributions, limit theorems, queuing systems also provide practical knowledge of using and implementing the relevant hardware and software specific parallelization methodologies;

(6) the practical value of using proposed educational tools in the educational practice is enabling simulation-making and SI centered approach to teaching and learning processes. These tools promote practical knowledge of the relevant parallelization techniques, including HPC computational platforms and big-data-related topics, focusing on simulation making and SI enabling practical students' activities;

(7) the innovative computational model for experimental study of the law of the iterated logarithm for the system of queues in series under overloading conditions has been developed. It is shown that under certain conditions it is possible to use at least a 2.6 times more efficient parametric recurrent solution if it is implemented on processors with RISC architecture;

(8) the developed computational model enable to conduct a comprehensive study of the system of queues in series under overloading conditions. This study is based on the computer simulation of the system. The simulation results are obtained within the constraint on available computational resources for the following configurations of the system and modeling experiment: (1) M/M/1 system of queues in series with $\mathbf{2^{10}}$ servicing phases, 1E+7 number of customers, 1E+2 Monte Carlo trials; (2) M/M/1 system with $\mathbf{2^{8}}$ servicing phases, 1E+8 number of customers, 1E+2 Monte Carlo trials; (3) M/M/1 system with $\mathbf{2^{3}}$ servicing phases, 1E+10 number of customers, 1E+2 Monte Carlo trials; (4) M/χ^2/1 system with $\mathbf{2^{2}}$ servicing phases, 1E+9 number of customers, 1E+2 Monte Carlo trials. The results of the simulation confirm the theoretical assumptions with the significance level equal to **0.01**;

(9) the developed educational solutions and sample learning resources were evaluated by the high level experts in the field of informatics education and informatics engineering with a very positive outcome of the evaluation results. The level of the confidence interval is 90%.

Appendices

A Implementation of SLOs

A.1 Sequential programming model

```
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>

#define OUT_FILE  "out_seq.txt"


#define MC 100   //number of Monte-Carlo simulations
#define N 1000  // clients
#define M 5       // servicing phases

int lambda[M + 1] = {0}; //distribution parameter
double tau = 0; //interaarival time
double st = 0; //sojourn time
//sojorn time of the previous customer in each phase
double st_prev[M] = {0};
double results[MC] = {0};

int main(int argc, char *argv[]) {

    gsl_rng * ran; //random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);

    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;


    for (unsigned j = 0; j < MC; j++) {
        tau = gsl_ran_exponential(ran, 1.0 / (double) lambda[0]);
        st = 0;
        for (unsigned k = 0; k < M; k++) st_prev[k] = 0.;

        for (unsigned i = 0; i < N; i++) {
            for (unsigned t = 0; t < M; t++) {
                //recurrent equation
                st += gsl_ran_exponential(ran, 1.0 / lambda[t + 1])
                                     + fmax(0.0, st_prev[t] - st - tau);
                st_prev[t] = st;
            }
        }
        results [j] = st;
    }

    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp, "%d%s%d%s%d\n", N, ",", lambda[0], ",", lambda[M]);
    for (int j = 0; j < MC - 1; j++) fprintf(fp, "%f%s", results[j], ",");
    fprintf(fp, "%f\n", results[MC - 1]);
    fclose(fp);
    gsl_rng_free(ran);
}
```

A.2 Distributed memory programming model

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <unistd.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include "mpi.h"

#define OUT_FILE  "out_mpi.txt"
#define MC 100    //number of Monte-Carlo (MC) simulations in each process
#define N 10000   // number of clients
#define M 5       //number of phases
#define NP 10     //number of MPI processes

void print_results(double *results, double time, int *lambda) {

    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp, "%d%s%d%s%d%s%d\n", N, ",", M, ",", lambda[0], ",",
                                                          lambda[M]);
    time = MPI_Wtime() - time;
    fprintf(fp, "%f\n", time);
    for (int i = 0; i < MC * NP - 1; i++) fprintf(fp, "%f%s",
                                                    results[i], ",");
    fprintf(fp, "%f\n", results[MC * NP - 1]);
    fclose(fp);
}

void process(int numprocs, int myid, gsl_rng * ran, int * lambda) {
    double time = MPI_Wtime(); //start time
    double tau[MC] = {0}; //interarrival time
    double st[MC] = {0}; //sojourn time
    double st_prev[M][MC] = {
        {0}
    }; //sojourn time of the previous customer in each phase
    double results[MC * NP] = {0}; // overall results

    for (int j = 0; j < MC; j++) {
        //init each MC trial
        tau[j] = gsl_ran_exponential(ran, 1.0 / lambda[0]);
        st[j] = 0;
        for (int i = 0; i < M; i++)
            for (int j = 0; j < MC; j++) st_prev[i][j] = 0.;
        for (int i = 0; i < N; i++) {
            for (int t = 0; t < M; t++) {
                //recurrent equation
                st[j] += gsl_ran_exponential(ran, 1.0 / lambda[t + 1])
                        + fmax(0.0, st_prev[t][j] - st[j] - tau[j]);
                st_prev[t][j] = st[j];
            }
        }
    }
    MPI_Gather(&st, MC, MPI_DOUBLE, &results, MC, MPI_DOUBLE, 0,
                                                        MPI_COMM_WORLD);
    if (myid == 0) {
        print_results(&results[0], time, &lambda[0]);
    }
}

int main(int argc, char *argv[]) {
    int namelen;
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int numprocs;
    int myid;
    gsl_rng * ran; //random generator
    int lambda[M + 1] = {0}; //parameter of the exponential distribution

    //init mpi
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &(numprocs));
    MPI_Comm_rank(MPI_COMM_WORLD, &(myid));
    MPI_Get_processor_name(processor_name, &namelen);

    //init parameter for the exponential distribution
    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;

    fprintf(stdout, "Process %d of %d is on %s\n", myid, numprocs,
                                                         processor_name);
    fflush(stdout);

    //init random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);
    gsl_rng_set(ran, (long) (myid)*22);

    //process
    process(numprocs, myid, ran, lambda);

    //finish
    gsl_rng_free(ran);
    MPI_Finalize();
    return (0);
}
```

A.3 Distributed memory pipeline model

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <unistd.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include "mpi.h"

#define OUT_FILE  "out_mpi_pipe.txt"
#define PIPE_MSG 0 // next pipe node
#define END_MSG 1  // finish
#define MC 10      //number of Monte-Carlo (MC) simulations in each chunk
#define NP 10      //number of processes (client axis)
#define CMC 100    //number of MC chunks
#define N 1000     // number of clients
#define M 5        //number of phases

void print_results(double *results, double time, int *lambda) {

    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp, "%d%s%d%s%d%s%d\n", N, ",", M, ",", lambda[0], ",",
                                                              lambda[M]);
    time = MPI_Wtime() - time;
    fprintf(fp, "%f\n", time);
    for (int i = 0; i < MC * CMC - 1; i++) fprintf(fp, "%f%s",
                                                         results[i], ",");
    fprintf(fp, "%f\n", results[MC * CMC - 1]);
    fclose(fp);
}

void node(int numprocs, int myid, gsl_rng * ran, int * lambda) {
    int nmcb = 0;
    int nmcb_id = 0;
    int i, j, k, t, u, v;
    double time = MPI_Wtime(); //start time
    MPI_Status Status;

    double tau[MC] = {0}; //interarrival time
    double st[MC] = {0}; //sojourn time
    //sojourn time of the previous customer in each phase
    double st_prev[M][MC] = {{0}};
    double results[MC * CMC] = {0}; // overall results
```

```

while (1) {
    nmcb_id = CMC; // aux var. to omit the cycle
    if (myid != 0) { //receive data from the previous node
        MPI_Recv(&tau, MC, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                             MPI_COMM_WORLD, &Status);
        if (Status.MPI_TAG == END_MSG) break;
        MPI_Recv(&st, MC, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                             MPI_COMM_WORLD, &Status);
        MPI_Recv(&st_prev, MC*M, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                             MPI_COMM_WORLD, &Status);
        //eliminate below for for other than the main thread
        nmcb_id = 1;
    }

    //nmbc- Number of MC batches( for the main process)
    for (k = 0; k < nmcb_id; k++) {
        for (j = 0; j < MC; j++) {
            if (myid == 0) { //init each MC trial (main process)
                tau[j] = gsl_ran_exponential(ran, 1.0 / lambda[0]);
                st[j] = 0;
                for (u = 0; u < M; u++)
                    for (v = 0; v < MC; v++) st_prev[u][v] = 0.;
            }

            for (i = 0; i < N / numprocs; i++) {
                for (t = 0; t < M; t++) {
                    //recurrent equation
                    st[j] += gsl_ran_exponential(ran, 1.0 /
            lambda[t + 1]) + fmax(0.0,st_prev[t][j] - st[j] - tau[j]);
                    st_prev[t][j] = st[j];
                }
            }
            results[j + MC * nmcb] = st[j];
        }
        nmcb++;

        if (myid != numprocs - 1) {
            //if not the last process send data to the next process
            MPI_Send(&tau, MC, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                    MPI_COMM_WORLD);
            MPI_Send(&st, MC, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                    MPI_COMM_WORLD);
            MPI_Send(&st_prev, MC*M, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                    MPI_COMM_WORLD);
        }
    }
    //if the main process - go out of while cycle
    if (myid == 0)break;
}
//if finished - send the end msg. to the next pipe node
if (myid != numprocs - 1)
```

```
        MPI_Send(&tau, MC, MPI_DOUBLE, myid + 1, END_MSG, MPI_COMM_WORLD);
    //if last process - send results
    if (myid == numprocs - 1)
        MPI_Send(&results, MC * CMC, MPI_DOUBLE, 0, PIPE_MSG,
                                                          MPI_COMM_WORLD);
    //print results
    if (myid == 0) {
        MPI_Recv(&results, MC*CMC, MPI_DOUBLE, numprocs - 1, MPI_ANY_TAG,
                                                 MPI_COMM_WORLD, &Status);
        print_results(&results[0], time, &lambda[0]);
    }
}

int main(int argc, char *argv[]) {
    int namelen;
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int numprocs;
    int myid;
    gsl_rng * ran; //random generator
    int lambda[M + 1] = {0}; //parameter of the exponential distribution

    //init MPI
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &(numprocs));
    MPI_Comm_rank(MPI_COMM_WORLD, &(myid));
    MPI_Get_processor_name(processor_name, &namelen);

    //init parameter for the exponential distribution
    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;

    fprintf(stdout, "Process %d of %d is on %s\n", myid, numprocs,
                                                       processor_name);
    fflush(stdout);

    //init random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);
    gsl_rng_set(ran, (long) (myid)*22);

    //process
    node(numprocs, myid, ran, lambda);

    //finish
    gsl_rng_free(ran);
    MPI_Finalize();
    return (0);
}
```

A.4 Shared memory programming model

```
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>

#define OPENMP 12  //number of OpenMP threads
#define OUT_FILE  "out_openmp.txt"
#define MC 200  //number of Monte-Carlo simulations in one thread
#define N 10000  // number of clients
#define M 5       // number of phases


int lambda[M + 1] = {0}; //parameters of the exponential distribution
double tau = 0; //interarrival time
double st = 0; //sojourn time
double st_prev[M] = {0}; //sojourn time for the previous client
// results- sojourn time for all threads trials
double results[MC*OPENMP] = {0};
//results- sojourn time for each thread trial
double th_results[MC] = {0};
gsl_rng * ran; //random generator

int main(int argc, char *argv[]) {
    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;
    unsigned long int i, t, j;
    int th_id; //thread number

#pragma omp parallel num_threads(OPENMP) private(th_id,ran,j,i,t,tau,
                                                          st,st_prev)  \
firstprivate(th_results) shared(results,lambda)
    {
        th_id = omp_get_thread_num();

        //printf("Hello World from thread %d\n", th_id);

        gsl_rng_env_setup();
        ran = gsl_rng_alloc(gsl_rng_ranlxs2);
        gsl_rng_set(ran, (long) th_id * 22); //seed
        for (j = 0; j < MC; j++) {
            tau = gsl_ran_exponential(ran, 1.0 / lambda[0]);
            st = 0.;
            for (i = 0; i < M; i++) st_prev[i] = 0.;

            for (i = 0; i < N; i++) {
                for (t = 0; t < M; t++) {
                    //recurrent equation
                    st += gsl_ran_exponential(ran, 1.0 / lambda[t + 1])
```

```
                                    + fmax(0.0,st_prev[t] - st - tau);
                    st_prev[t] = st;
                }
            }
            th_results [j] = st;
        }
        for (i = 0; i < MC; i++) results[i + th_id * MC] = th_results [i];
        gsl_rng_free(ran);
    }
    //printing results
    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp, "%d%s%d%s%d\n", N, ",", lambda[0], ",", lambda[M]);
    for (i = 0; i < MC * OPENMP - 1; i++) fprintf(fp, "%f%s", results[i],
                                                                        ",");
    fprintf(fp, "%f\n", results[MC * OPENMP - 1]);
    fclose(fp);
    return EXIT_SUCCESS;
}
```

A.5 Shared memory static pipeline programming model

```
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>

#define OUT_FILE  "out_omp_pipe_static.txt"

#define MC 10000    //total number of Monte-Carlo (MC) simulations
#define CMC 1000  // chunks per MC axis
#define N 10000    // total number of clients
#define M 5     // total number of phases
#define CN 1000  // chunks per clients axis

#define OPENMP 12  //OMP threads

int lambda[M + 1] = {0}; // parameters of exponential distributions
double tau[MC] = {0}; // interarrival time for each MC trial
double st[MC] = {0}; // sojourn time for each MC trial
double st_prev[MC][M] = {{0}}; // sojourn time  of the previous client
// for each MC trial and each phase
int flag[CMC] = {0}; // aux variable - each MC chunk flag
int task_counter[CMC] = {0}; // aux variable - each MC chunk counter

int main(int argc, char *argv[]) {
    time_t t1, t2;
    t1 = time(NULL);

    //init exponential distribution parameters(heavy traffic case)
    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;

    gsl_rng * ran; //random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);
    gsl_rng_set(ran, (long) (CN * CMC + 10)*22.); //seed

    // set interarrival time for each MC trial
    for (int i = 0; i < MC; i++)
        tau[i] = gsl_ran_exponential(ran, 1.0 / lambda[0]);
    gsl_rng_free(ran);

    omp_set_num_threads(OPENMP);
#pragma omp parallel   //start threads
    {
#pragma omp single    //one thread to create tasks
        {
            int i, j, t, c; //aux variables
            int v = 0; // local variable - MC chunk number
            int sum = 0; // local variable - overall number of tasks
            //flag variable to stop external while cycle
            int while_flag = 1;

            // static task creation in each of MC chunks
            while (while_flag) { //create  many many tasks

#pragma omp task default(none)  private(i,j,t,c,ran) \
 firstprivate(tau,lambda,v,gsl_rng_ranlxs2,flag) shared(st,st_prev,
                                                          task_counter)

                if (!flag[v]) {
#pragma omp taskyield //suspend this task if previous task had not
                                                               finished
                }
                flag[v] = 1;

                gsl_rng * ran; //random generator for this task
                gsl_rng_env_setup();
                ran = gsl_rng_alloc(gsl_rng_ranlxs2);
                gsl_rng_set(ran, (long) (task_counter[v] + v * CN)*22.);
                                          /*seed with
                                                      the task number*/

                for (j = 0; j < MC / CMC; j++) {
                    c = j + v * (MC / CMC); // MC trial number
                    for (i = 0; i < N / CN; i++) {
                        for (t = 0; t < M; t++) {
                            //reccurrent equation
                            st[c] += gsl_ran_exponential(ran, 1.0 /
                                                        lambda[t + 1])
                            + fmax(0.0, st_prev[c][t] - st[c] - tau[c]);
                            st_prev[c][t] = st[c];
                        }
                    }
                }
                // end of the current task
                gsl_rng_free(ran);
                task_counter[v]++;
                if (task_counter[v] != CN) flag[v] = 0;
                v++;
                if (v == CMC) v = 0; // again a new loop
                sum = 0;
                for (i = 0; i < CMC; i++) sum += task_counter[i];
                if (sum == CN * CMC) while_flag = 0;
            }
        }
    }

    //print results
    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp,"%d%s%d%s%d%s%d\n",N,",",M,",",lambda[0],",",lambda[M]);
    t2 = time(NULL);
    fprintf(fp, "%f\n", difftime(t2, t1));
    for (int j = 0; j < MC - 1; j++) fprintf(fp, "%f%s", st[j], ",");
    fprintf(fp, "%f\n", st[MC - 1]);
    fclose(fp);
}
```

A.6 Shared memory dynamic programming model

```
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>

#define OUT_FILE  "out_omp_pipe.txt"
#define MC 10000  //total number of Monte-Carlo (MC) simulations
#define CMC 100  // chunks per MC axis
#define N 10000    // total number of clients
#define M 5     // total number of phases
#define CN 100  // chunks per clients axis
#define OPENMP 12  //OMP threads


int lambda[M + 1] = {0}; // parameters of exponential distributions
double tau[MC] = {0}; //interarrival time for each MC trial
double st[MC] = {0}; // sojourn time
```

```
// sojourn time  of the previous client for each MC trial and each phase
double st_prev[MC][M] = {{0}};
int flag[CMC] = {0}; // aux variable - each MC chunk flag
int task_counter[CMC] = {0}; // aux variable - each MC chunk counter

int main(int argc, char *argv[]) {
    time_t t1, t2;
    t1 = time(NULL);
    lambda[0] = 30000; // exponential distribution parameters
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;

    gsl_rng * ran; //random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);
    gsl_rng_set(ran, (long) (CN * CMC + 10)*22.); //seed

    // set interarrival time for each MC trial
    for (int i = 0; i < MC; i++)
        tau[i] = gsl_ran_exponential(ran, 1.0 / lambda[0]);
    gsl_rng_free(ran);

    omp_set_num_threads(OPENMP);
#pragma omp parallel   //start threads
    {
#pragma omp single    //one thread to create tasks
        {
            int i, j, t, c; //aux variables
            int v = 0; // local variable - MC chunk number
            int sum = 0; // local variable -  number of tasks
            int while_flag = 1; // var. to stop external while

            // dynamic task creation in each of MC chunks
            while (while_flag) {
                if (!flag[v]) {
                    flag[v] = 1;
                    //create a new task if previous task had finished
#pragma omp task default(none)  private(i,j,t,c,ran) \
firstprivate(tau,lambda,v,gsl_rng_ranlxs2) shared(st,st_prev,flag,
                                                            task_counter)
                    {
                        gsl_rng * ran; //random generator for this task
                        gsl_rng_env_setup();
                        ran = gsl_rng_alloc(gsl_rng_ranlxs2);
                        //seed with the task number
                        gsl_rng_set(ran,(long)(task_counter[v]+v*CN)*22.);

                        for (j = 0; j < MC / CMC; j++) {
                            c = j + v * (MC / CMC); // MC trial number
                            for (i = 0; i < N / CN; i++) {
                                for (t = 0; t < M; t++) {
                                    //recurrent equation
```

```c
                                    st[c] += gsl_ran_exponential(ran,
        1.0  / lambda[t + 1]) + fmax(0.0, st_prev[c][t] - st[c] - tau[c]);
                                    st_prev[c][t] = st[c];
                                }
                            }
                        }
                        // end of the current task
                        gsl_rng_free(ran);
                        task_counter[v]++;
                        flag[v] = 0;
                    }
                }

                // if all tasks of this chunk of Monte-Carlo trials
                //  had finished -then stop this chunk
                // v numbered MC chunk is over
                if (task_counter[v] == CN) flag[v] = 1;
                v++;
                if (v == CMC) v = 0; // again a new loop
                sum = 0; // variable to test all chunks
                for (i = 0; i < CMC; i++) sum += task_counter[i];
                // if all task had finished - exit while cycle
                if (sum == CN * CMC) while_flag = 0;
            }
        }
    }

    //print results
    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp,"%d%s%d%s%d%s%d\n",N,",",M,",",lambda[0],",",lambda[M]);
    t2 = time(NULL);
    fprintf(fp, "%f\n", difftime(t2, t1));
    for (int j = 0; j < MC - 1; j++) fprintf(fp, "%f%s", st[j], ",");
    fprintf(fp, "%f\n", st[MC - 1]);
    fclose(fp);
    return (0);
}
```

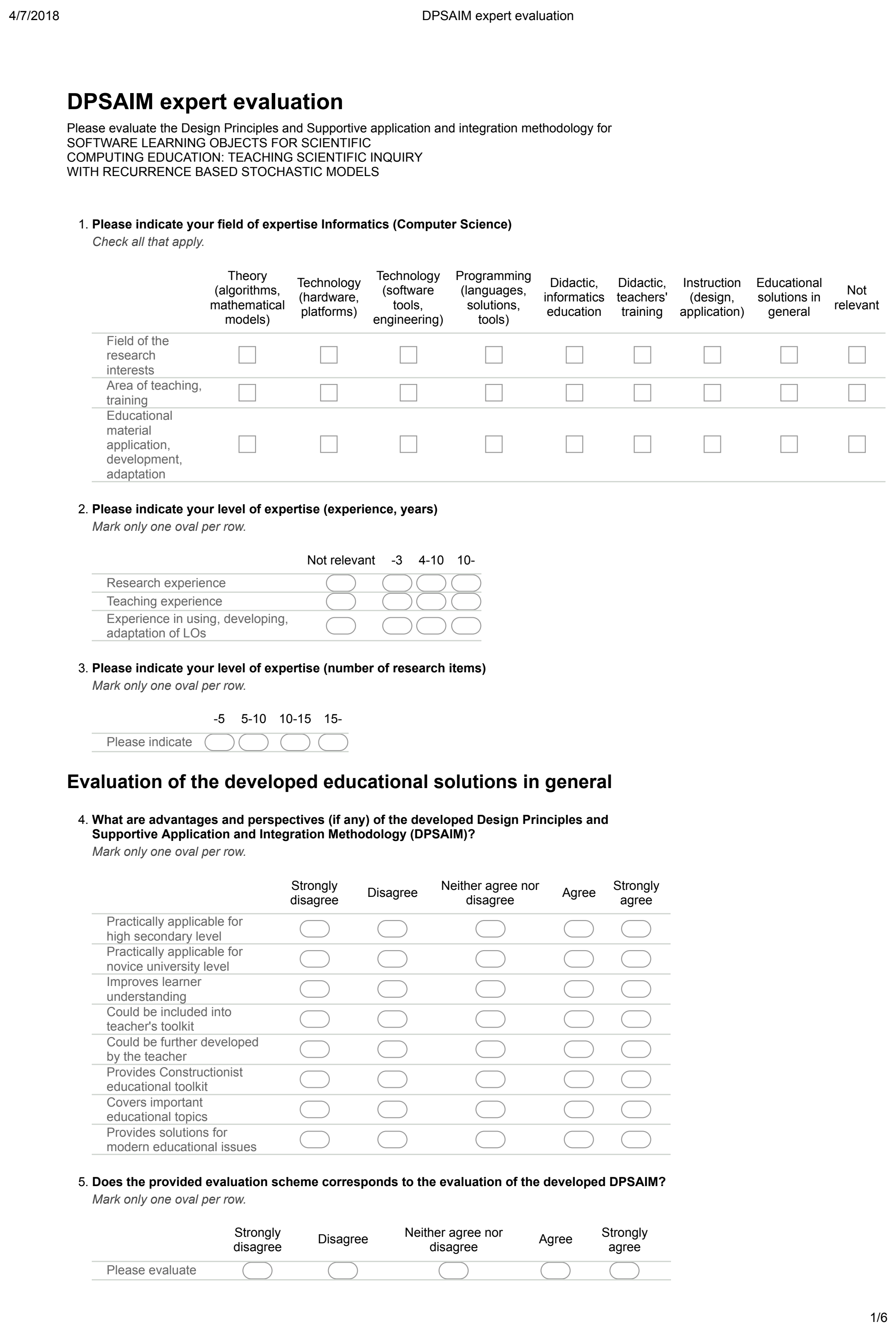
A.7 Hybrid programming model

```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <unistd.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>
#include "mpi.h"

#define OUT_FILE  "out_hybrid.txt"
```

```
#define PIPE_MSG 0  // next pipe node
#define END_MSG 1  // finish
#define OPENMP 12 //number of OMP threads

#define MC 100    //number of Monte-Carlo (MC) simulations in each chunk
#define NP 10     //number of processes (client axis)
#define CMC 100   //number of MC chunks
#define N 1000   // number of clients
#define M 5       //number of phases

void print_results(double *results, double time, int *lambda) {
    FILE *fp;
    const char DATA_FILE[] = OUT_FILE;
    fp = fopen(DATA_FILE, "w");
    fprintf(fp,"%d%s%d%s%d%s%d\n",N,",",M,",",lambda[0],",",lambda[M]);
    time = MPI_Wtime() - time;
    fprintf(fp, "%f\n", time);
    for (int i = 0; i < MC * CMC - 1; i++)
                                    fprintf(fp, "%f%s", results[i], ",");
    fprintf(fp, "%f\n", results[MC * CMC - 1]);
    fclose(fp);
}

void node(int numprocs, int myid, gsl_rng * ran, int * lambda) {
    int nmcb = 0; //nmbc - number of MC batches
    int nmcb_id = 0;
    int i, j, k, t, u, v;
    double time = MPI_Wtime(); //program start time
    MPI_Status Status;

    double tau[MC] = {0}; //interarrival time
    double st[MC] = {0}; //sojourn time
    //sojourn time of the previous customer in each phase
    double st_prev[M][MC] = {{0}};
    double results[MC * CMC] = {0}; // overall results
    double temp; // aux variable
    while (1) {
        nmcb_id = CMC; // aux var. to omit the cycle

        if (myid != 0) { //receive data from the previous node
            MPI_Recv(&tau, MC, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                               MPI_COMM_WORLD, &Status);
            if (Status.MPI_TAG == END_MSG) break;
            MPI_Recv(&st, MC, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                               MPI_COMM_WORLD, &Status);
            MPI_Recv(&st_prev, MC*M, MPI_DOUBLE, myid - 1, MPI_ANY_TAG,
                                               MPI_COMM_WORLD, &Status);
            //eliminate below for in case of not the main thread
            nmcb_id = 1;
        }


        for (k = 0; k < nmcb_id; k++) {
            omp_set_num_threads(OPENMP);
            {
#pragma omp  parallel for default(shared)         private(j,i,t,u,v)
                for (j = 0; j < MC; j++) {
                    if (myid == 0) { //init each MC trial (main process)
#pragma omp critical
                        {
                     tau[j] = gsl_ran_exponential(ran, 1.0 / lambda[0]);
                        }
                        st[j] = 0;
                        for (u = 0; u < M; u++)
                            for (v = 0; v < MC; v++) st_prev[u][v] = 0.;
                    }

                    for (i = 0; i < N / NP; i++) {
                        for (t = 0; t < M; t++) {
                            //recurrent equation
                      temp = gsl_ran_exponential(ran,1.0/lambda[t + 1]);
#pragma omp critical
                            {
                      temp = gsl_ran_exponential(ran,1.0/lambda[t + 1]);
                            }
                st[j] += temp + fmax(0.0, st_prev[t][j] - st[j] -tau[j]);
                            st_prev[t][j] = st[j];
                        }
                    }
                    results[j + MC * nmcb] = st[j];
                }
            }
            nmcb++;

            if (myid != numprocs - 1) {
                //if not the last process send data to the next process
                MPI_Send(&tau, MC, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                        MPI_COMM_WORLD);
                MPI_Send(&st, MC, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                        MPI_COMM_WORLD);
                MPI_Send(&st_prev, MC*M, MPI_DOUBLE, myid + 1, PIPE_MSG,
                                                        MPI_COMM_WORLD);
            }
        }
        //if the main process - go out of while cycle
        if (myid == 0)break;
    }
    //if finished - send the end message to the next pipe node
    if (myid != numprocs - 1)MPI_Send(&tau, MC, MPI_DOUBLE, myid + 1,
                                               END_MSG, MPI_COMM_WORLD);
    //if last process - send results
    if (myid == numprocs - 1) MPI_Send(&results,MC*CMC,MPI_DOUBLE,0,
                                              PIPE_MSG,  MPI_COMM_WORLD);
    //print results
    if (myid == 0) {
        MPI_Recv(&results,MC*CMC,MPI_DOUBLE,numprocs - 1,MPI_ANY_TAG,
                                                MPI_COMM_WORLD, &Status);
        print_results(&results[0], time, &lambda[0]);
    }
}

int main(int argc, char *argv[]) {
    int namelen;
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int numprocs;
    int myid;
    gsl_rng * ran; //random generator
    //parameter for the exponential distribution
    int lambda[M + 1] = {0};

    //init mpi
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &(numprocs));
    MPI_Comm_rank(MPI_COMM_WORLD, &(myid));
    MPI_Get_processor_name(processor_name, &namelen);

    //init parameter for the exponential distribution
    lambda[0] = 30000;
    for (int i = 1; i < M; i++)lambda[i] = lambda[i - 1] - 25000 / M;
    lambda[M] = 5000;

    fprintf(stdout,"Process %d of %d is on %s\n",myid,
                                             numprocs,processor_name);
    fflush(stdout);

    //init random generator
    gsl_rng_env_setup();
    ran = gsl_rng_alloc(gsl_rng_ranlxs2);
    gsl_rng_set(ran, (long) (myid)*22);

    //process
    node(numprocs, myid, ran, lambda);

    //finish
    gsl_rng_free(ran);
    MPI_Finalize();
    return (0);
}
```

B Expert evaluation

B.1 Questionnaire form



4/7/2018 DPSAIM expert evaluation

DPSAIM expert evaluation

Please evaluate the Design Principles and Supportive application and integration methodology for
SOFTWARE LEARNING OBJECTS FOR SCIENTIFIC
COMPUTING EDUCATION: TEACHING SCIENTIFIC INQUIRY
WITH RECURRENCE BASED STOCHASTIC MODELS

1. **Please indicate your field of expertise Informatics (Computer Science)**
Check all that apply.

	Theory (algorithms, mathematical models)	Technology (hardware, platforms)	Technology (software tools, engineering)	Programming (languages, solutions, tools)	Didactic, informatics education	Didactic, teachers' training	Instruction (design, application)	Educational solutions in general	Not relevant
Field of the research interests	☐	☐	☐	☐	☐	☐	☐	☐	☐
Area of teaching, training	☐	☐	☐	☐	☐	☐	☐	☐	☐
Educational material application, development, adaptation	☐	☐	☐	☐	☐	☐	☐	☐	☐

2. **Please indicate your level of expertise (experience, years)**
Mark only one oval per row.

	Not relevant	-3	4-10	10-
Research experience	◯	◯	◯	◯
Teaching experience	◯	◯	◯	◯
Experience in using, developing, adaptation of LOs	◯	◯	◯	◯

3. **Please indicate your level of expertise (number of research items)**
Mark only one oval per row.

	-5	5-10	10-15	15-
Please indicate	◯	◯	◯	◯

Evaluation of the developed educational solutions in general

4. **What are advantages and perspectives (if any) of the developed Design Principles and Supportive Application and Integration Methodology (DPSAIM)?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Practically applicable for high secondary level	◯	◯	◯	◯	◯
Practically applicable for novice university level	◯	◯	◯	◯	◯
Improves learner understanding	◯	◯	◯	◯	◯
Could be included into teacher's toolkit	◯	◯	◯	◯	◯
Could be further developed by the teacher	◯	◯	◯	◯	◯
Provides Constructionist educational toolkit	◯	◯	◯	◯	◯
Covers important educational topics	◯	◯	◯	◯	◯
Provides solutions for modern educational issues	◯	◯	◯	◯	◯

5. **Does the provided evaluation scheme corresponds to the evaluation of the developed DPSAIM?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Please evaluate	◯	◯	◯	◯	◯

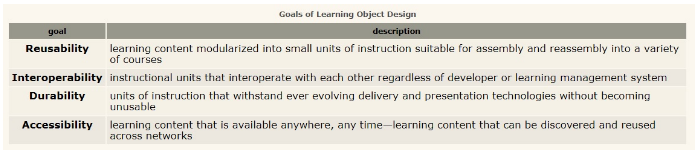
1/6

6. **What is the general opinion or comments (if any) on the Research?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Correspondence of the volume of the research					
Correspondence of the internal structure					
The relevance of the research topic					
The correspondence of the implemented research methodology					
The novelty of the developed educational solutions					
The novelty of the developed technological solutions					
Correspondence of the theoretical grounding					
The relevance and correspondence of the experimental research and case studies					
Correspondence of the literature review					

The evaluation of the developed sample resources (Software Learning Objects)

Goals of LOs Design



Goals of Learning Object Design

goal	description
Reusability	learning content modularized into small units of instruction suitable for assembly and reassembly into a variety of courses
Interoperability	instructional units that interoperate with each other regardless of developer or learning management system
Durability	units of instruction that withstand ever evolving delivery and presentation technologies without becoming unusable
Accessibility	learning content that is available anywhere, any time—learning content that can be discovered and reused across networks

7. **Are the developed learning resources - DPSAIM - compatible with the objectives of the DPSAIM design?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Reusability					
Interoperability					
Durability					
Accessibility					

Churchill LOs taxonomy

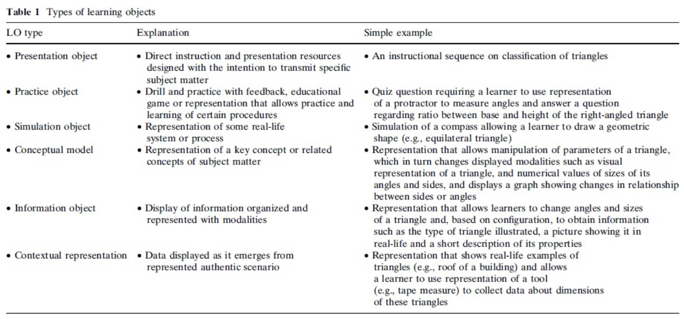


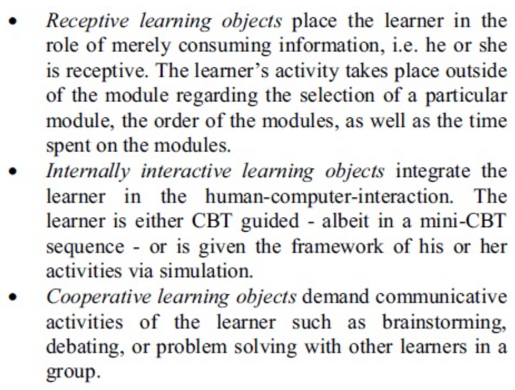
Table 1 Types of learning objects

LO type	Explanation	Simple example
• Presentation object	• Direct instruction and presentation resources designed with the intention to transmit specific subject matter	• An instructional sequence on classification of triangles
• Practice object	• Drill and practice with feedback, educational game or representation that allows practice and learning of certain procedures	• Quiz question requiring a learner to use representation of a protractor to measure angles and answer a question regarding ratio between base and height of the right-angled triangle
• Simulation object	• Representation of some real-life system or process	• Simulation of a compass allowing a learner to draw a geometric shape (e.g., equilateral triangle)
• Conceptual model	• Representation of a key concept or related concepts of subject matter	• Representation that allows manipulation of parameters of a triangle, which in turn changes displayed modalities such as visual representation of a triangle, and numerical values of sizes of its angles and sides, and displays a graph showing changes in relationship between sides or angles
• Information object	• Display of information organized and represented with modalities	• Representation that allows learners to change angles and sizes of a triangle and, based on configuration, to obtain information such as the type of triangle illustrated, a picture showing it in real-life and a short description of its properties
• Contextual representation	• Data displayed as it emerges from represented authentic scenario	• Representation that shows real-life examples of triangles (e.g., roof of a building) and allows a learner to use representation of a tool (e.g., tape measure) to collect data about dimensions of these triangles

8. **Does the developed SLOs meet LO Churchill taxonomy ?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Presentation objects	◯	◯	◯	◯	◯
Practice object	◯	◯	◯	◯	◯
Simulation object	◯	◯	◯	◯	◯
Conceptual model	◯	◯	◯	◯	◯
Information object	◯	◯	◯	◯	◯
Contextual representation	◯	◯	◯	◯	◯

Redeker LO taxonomy



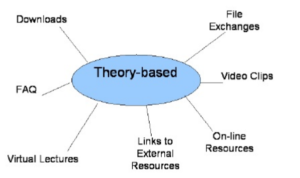
- *Receptive learning objects* place the learner in the role of merely consuming information, i.e. he or she is receptive. The learner's activity takes place outside of the module regarding the selection of a particular module, the order of the modules, as well as the time spent on the modules.
- *Internally interactive learning objects* integrate the learner in the human-computer-interaction. The learner is either CBT guided - albeit in a mini-CBT sequence - or is given the framework of his or her activities via simulation.
- *Cooperative learning objects* demand communicative activities of the learner such as brainstorming, debating, or problem solving with other learners in a group.

9. **Does the developed SLOs meet LO Redeker taxonomy ?**
Mark only one oval per row.

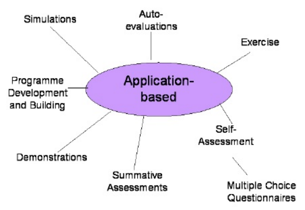
	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Receptive	◯	◯	◯	◯	◯
Internally interactive	◯	◯	◯	◯	◯
Cooperative	◯	◯	◯	◯	◯

Finlay LOs taxonomy

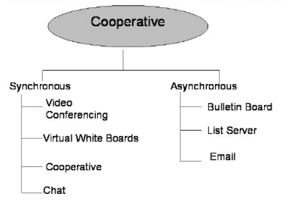
- Theory-based, providing conceptual and factual material that require the learner to read, listen or watch



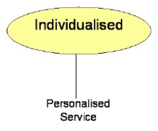
- Application-based, which require the learner to respond with some reasoned action



- Cooperative, which require learners to work togethe



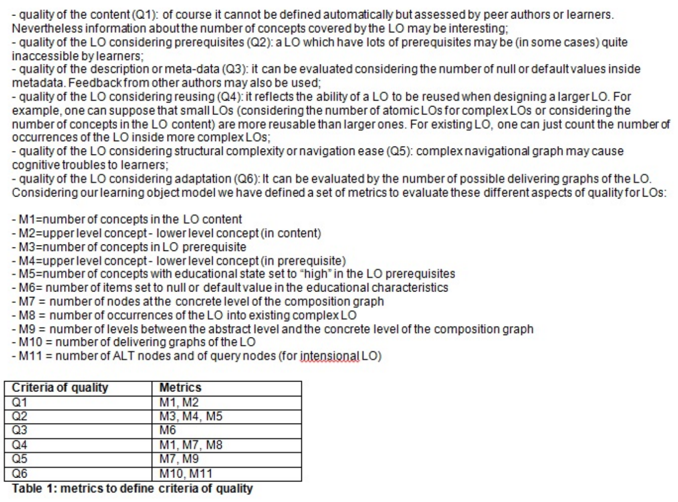
- Individualised, which are customised to particula learning needs



10. **Does the developed SLOs meet LO Finlay taxonomy?**
Mark only one oval per row.

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
Theory-based	◯	◯	◯	◯	◯
Cooperative	◯	◯	◯	◯	◯
Application-based	◯	◯	◯	◯	◯
Individualised	◯	◯	◯	◯	◯

LO quality measurement metrics


- quality of the content (Q1): of course it cannot be defined automatically but assessed by peer authors or learners. Nevertheless information about the number of concepts covered by the LO may be interesting;
- quality of the LO considering prerequisites (Q2): a LO which have lots of prerequisites may be (in some cases) quite inaccessible by learners;
- quality of the description or meta-data (Q3): it can be evaluated considering the number of null or default values inside metadata. Feedback from other authors may also be used;
- quality of the LO considering reusing (Q4): it reflects the ability of a LO to be reused when designing a larger LO. For example, one can suppose that small LOs (considering the number of atomic LOs for complex LOs or considering the number of concepts in the LO content) are more reusable than larger ones. For existing LO, one can just count the number of occurrences of the LO inside more complex LOs;
- quality of the LO considering structural complexity or navigation ease (Q5): complex navigational graph may cause cognitive troubles to learners;
- quality of the LO considering adaptation (Q6): It can be evaluated by the number of possible delivering graphs of the LO.
Considering our learning object model we have defined a set of metrics to evaluate these different aspects of quality for LOs:

- M1=number of concepts in the LO content
- M2=upper level concept - lower level concept (in content)
- M3=number of concepts in LO prerequisite
- M4=upper level concept - lower level concept (in prerequisite)
- M5=number of concepts with educational state set to "high" in the LO prerequisites
- M6= number of items set to null or default value in the educational characteristics
- M7 = number of nodes at the concrete level of the composition graph
- M8 = number of occurrences of the LO into existing complex LO
- M9 = number of levels between the abstract level and the concrete level of the composition graph
- M10 = number of delivering graphs of the LO
- M11 = number of ALT nodes and of query nodes (for intensional LO)

Criteria of quality	Metrics
Q1	M1, M2
Q2	M3, M4, M5
Q3	M6
Q4	M1, M7, M8
Q5	M7, M9
Q6	M10, M11

Table 1: metrics to define criteria of quality

11. **The evaluation from the perspectives of quality measurement metrics**
Mark only one oval per row.

	Not relevant	Low	Middle	High
Q1: quality of the content	◯	◯	◯	◯
Q2: quality of the LO considering prerequisites	◯	◯	◯	◯
Q3: quality of the description	◯	◯	◯	◯
Q4: quality of the LO considering reusing	◯	◯	◯	◯
Q5: quality of the LO considering structural complexity	◯	◯	◯	◯
Q6: quality of the LO considering adaptation	◯	◯	◯	◯

Does the developed SLOs meet the most important characteristics defined in the LO standards?

12. **Underlying organizational structure of this learning object (1.7)**
Check all that apply.

☐ atomic: an object that is indivisible (in this context)
☐ collection: a set of objects with no specified relationship between them
☐ hierarchical: a set of objects whose relationships can be represented by a tree structure.
☐ linear: a set of objects that are fully ordered. (Example: A set of objects that are connected by "previous" and "next" relationships.)

13. **The functional granularity of this learning object. (1.8)**
Check all that apply.

☐ 1: the smallest level of aggregation, e.g., raw media data or fragments.
☐ 2: a collection of level 1 learning objects, e.g., a lesson.
☐ 3: a collection of level 2 learning objects, e.g., a course.
☐ 4: the largest level of granularity, e.g., a set of courses that lead to a certificate.

14. **5.1. Interactivity Type. Predominant mode of learning supported by this learning project.**
Check all that apply.

☐ "Active" learning (e.g., learning by doing) is supported by content that directly induces productive action by the learner. An active learning object prompts the learner for semantically meaningful input or for some other kind of productive action or decision, not necessarily performed within the learning object's framework. Active documents include simulations, questionnaires, and exercises.
☐ "Expositive" learning (e.g., passive learning) occurs when the learner's job mainly consists of absorbing the content exposed to him (generally through text, mages or sound). An expositive learning object displays information but does not prompt the learner for any semantically meaningful input. Expositive documents include essays, video clips, all kinds of graphical material, and hypertext documents
☐ When a learning object blends the active and expositive interactivity types, then its interactivity type is "mixed."

15. **5.3. Interactivity Level. The degree of interactivity characterizing this learning object. The degree of interactivity characterizing this learning object. Interactivity in this context refers to the degree to which the learner can influence the aspect or behavior of the learning object.**
Mark only one oval per row.

	Very low	Low	Middle	High	Very high
Please indicate	◯	◯	◯	◯	◯

16. **5.4. Semantic Density. The degree of conciseness of a learning object. The semantic density of a learning object may be estimated in terms of its size, span, or—in the case of self-timed resources such as audio or video—duration. The semantic density of a learning object is independent of its difficulty. It is best illustrated with examples of expositive material, although it can be used with active resources as well.**
Mark only one oval per row.

	Very Low	Low	Middle	High	Very high
Please indicate	◯	◯	◯	◯	◯

17. **5.5. User Role. Principal user(s) for which this learning object was designed, most dominant first.**
Check all that apply.

☐ Teacher
☐ Author
☐ Learner
☐ Manager
☐ Other: ____________________

18. **5.6. Context The principal environment within which the learning and use of this learning object is intended to take place.**
Check all that apply.

☐ School
☐ Higher education
☐ Training
☐ Other

19. **5.7. Typical Age Range. Age of the typical intended user.This data element shall refer to developmental age, if that would be different from chronological age.**
Check all that apply.

- ☐ 7-11 primary school
- ☐ 12-14 primary school (5-8 school class)
- ☐ 15-16 primary school (9-10 school class)
- ☐ 16-18 secondary school (11-12 school class)
- ☐ 18- novice university

20. **5.8 Difficulty How hard it is to work with or through this learning object for the typical intended target audience.**
Mark only one oval per row.

	Very low	Low	Middle	High	Very High
Please indicate	◯	◯	◯	◯	◯

The evaluation from the perspectives of possible disadvantages or limitations

21. **Please evaluate possible influence of disadvantages and limitations on application and quality of provided solutions**
Mark only one oval per row.

	Very low	Low	Middle	High	Very hgh
Please indicate	◯	◯	◯	◯	◯

The evaluation from the perspectives of possibility of further development

22. **Please evaluate possibility of further development**
Mark only one oval per row.

	Very low	Low	Middle	High	Very high
Please indicate	◯	◯	◯	◯	◯

Additional evaluation topics.

23. **What are (if any) additional evaluation criteria?**

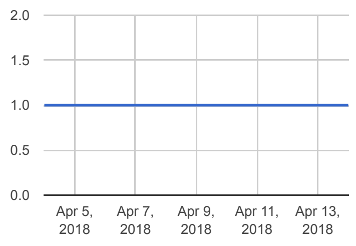
24. **What are (if any) a supplement expert opinion (advantages, disadvantages, opinion, recommendations, etc.) on the provided educational solutions and sample learning resources?**

Powered by Google Forms

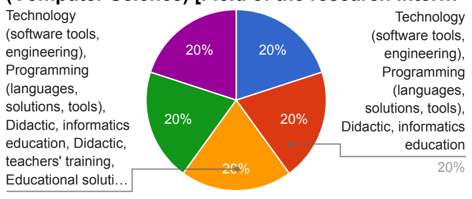
B.2 Summary of evaluation results

4/17/2018 [Addon Advanced summary] DPSAIM expert evaluation - Sat Apr 07 2018

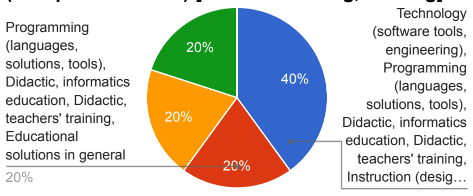
Timestamp 4/3/2018 16:01:24



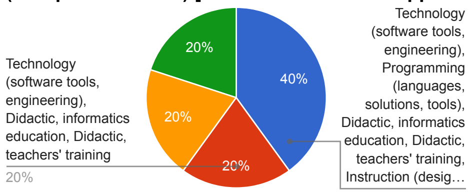
Please indicate your field of expertise Informatics (Computer Science) [Field of the research inter…



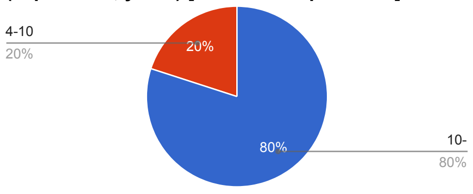
Please indicate your field of expertise Informatics (Computer Science) [Area of teaching, training]…



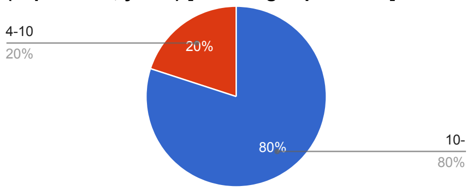
Please indicate your field of expertise Informatics (Computer Science) [Educational material appli…



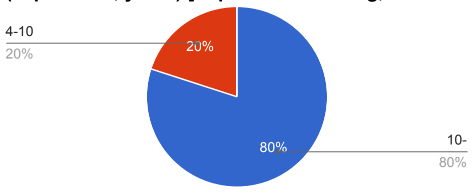
Please indicate your level of expertise (experience, years) [Research experience] 10-



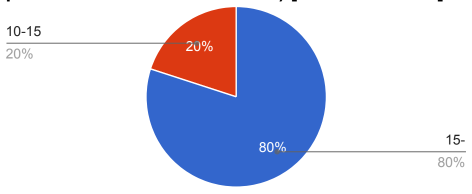
Please indicate your level of expertise (experience, years) [Teaching experience] 10-



Please indicate your level of expertise (experience, years) [Experience in using, devel…

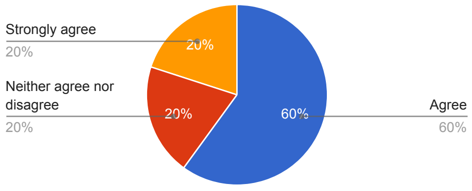


Please indicate your level of expertise (number of peer-reviewed research items) [Please indicate]…

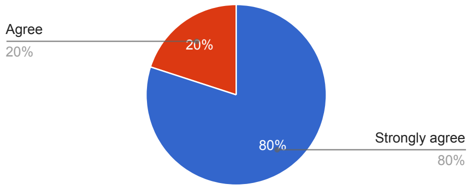


1/8

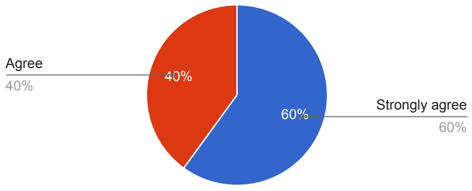
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



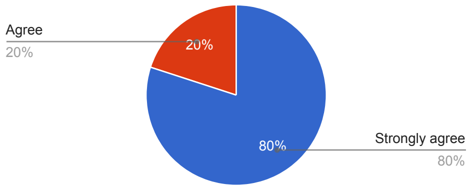
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



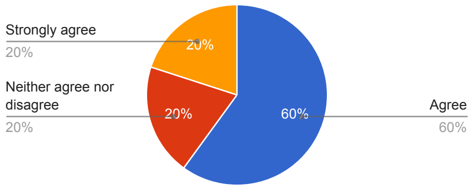
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



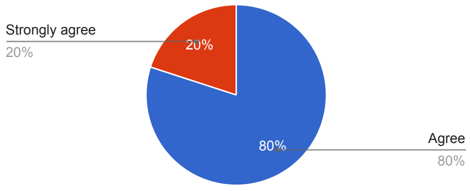
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



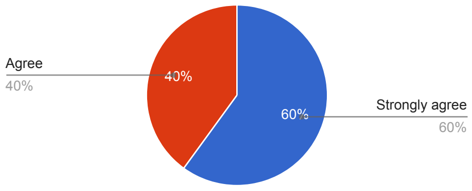
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



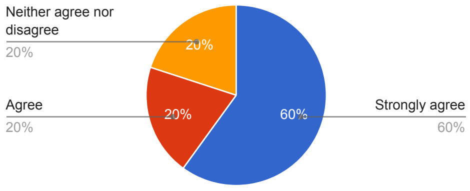
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



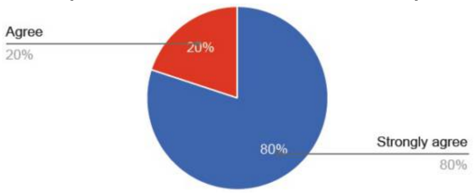
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



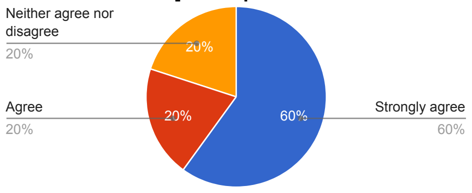
What are advantages and perspectives (if any) of the developed Design Principles and Supportiv…



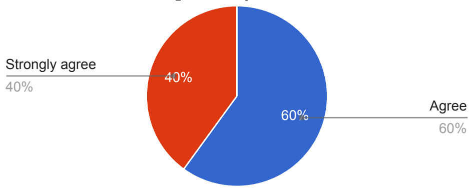
Does the provided evaluation scheme corresponds to the evaluation of the developed…



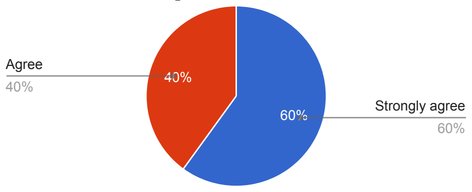
What is the general opinion or comments (if any) on the Research? [Correspondence of the volu…



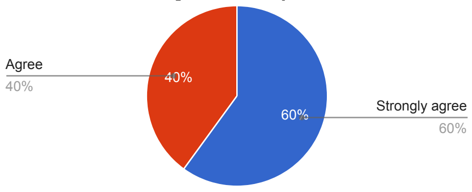
What is the general opinion or comments (if any) on the Research? [Correspondence of the inter…



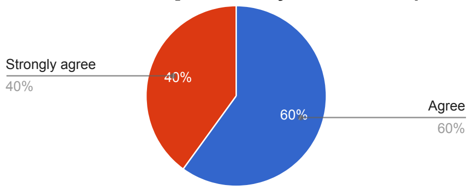
What is the general opinion or comments (if any) on the Research? [The relevance of the researc…



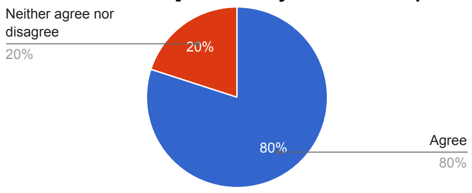
What is the general opinion or comments (if any) on the Research? [The correspondence of the i…



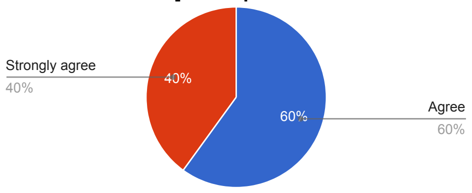
What is the general opinion or comments (if any) on the Research? [The novelty of the develope…



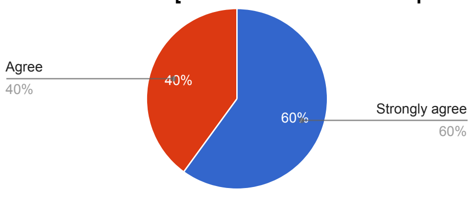
What is the general opinion or comments (if any) on the Research? [The novelty of the develope…



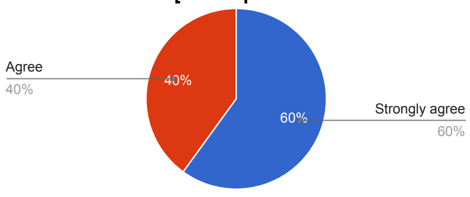
What is the general opinion or comments (if any) on the Research? [Correspondence of the theo…



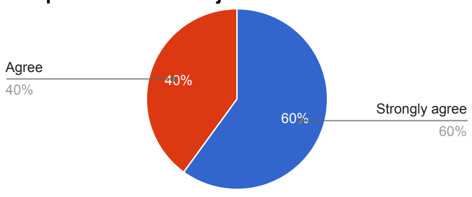
What is the general opinion or comments (if any) on the Research? [The relevance and correspo…



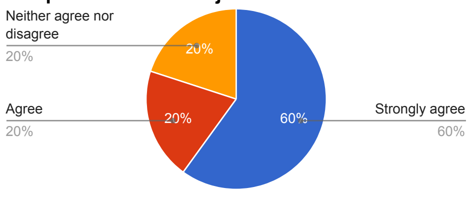
What is the general opinion or comments (if any) on the Research? [Correspondence of the litera…



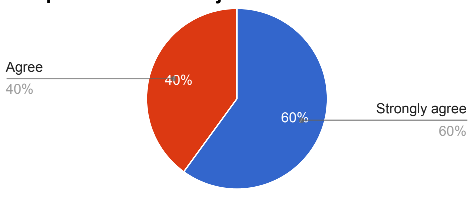
Are the developed learning resources - DPSAIM - compatible with the objectives of the DPSAIM d…



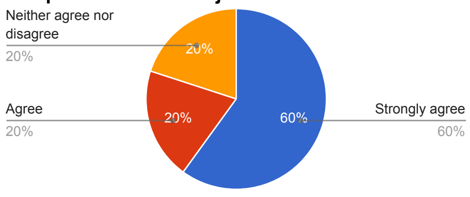
Are the developed learning resources - DPSAIM - compatible with the objectives of the DPSAIM d…



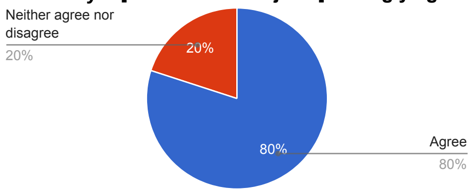
Are the developed learning resources - DPSAIM - compatible with the objectives of the DPSAIM d…



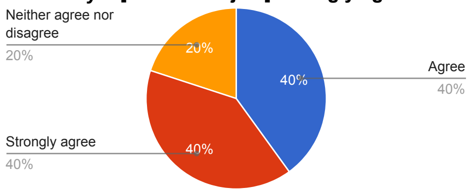
Are the developed learning resources - DPSAIM - compatible with the objectives of the DPSAIM d…



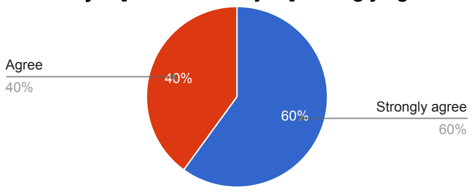
Does the developed SLOs meet LO Churchill taxonomy? [Presentation objects] Strongly agree



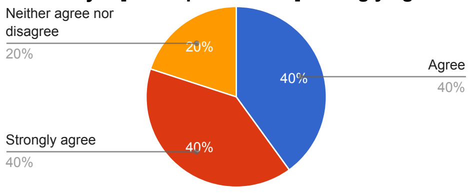
Does the developed SLOs meet LO Churchill taxonomy? [Practice object] Strongly agree

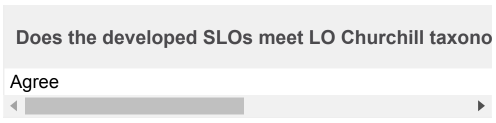


Does the developed SLOs meet LO Churchill taxonomy? [Simulation object] Strongly agree



Does the developed SLOs meet LO Churchill taxonomy? [Conceptual model] Strongly agree

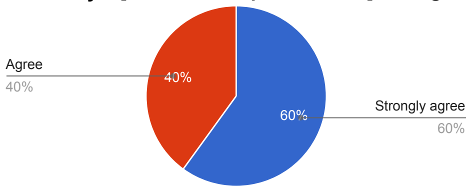




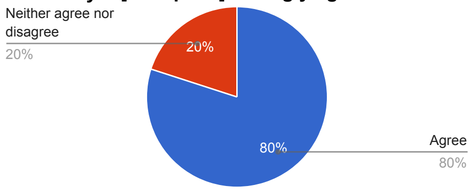
Does the developed SLOs meet LO Churchill taxono

Agree

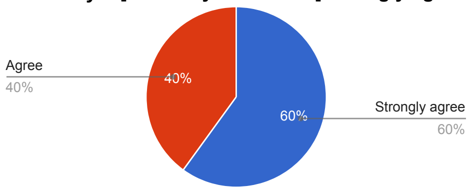
Does the developed SLOs meet LO Churchill taxonomy? [Contextual representation] Strong…



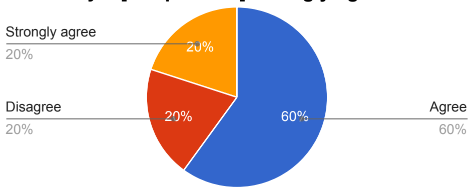
Does the developed SLOs meet LO Redeker taxonomy? [Receptive] Strongly agree



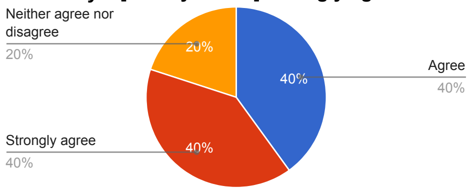
Does the developed SLOs meet LO Redeker taxonomy? [Internally interactive] Strongly agree



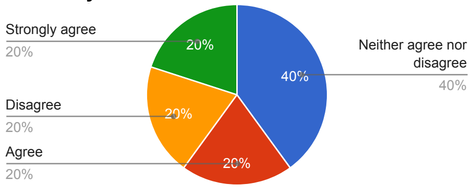
Does the developed SLOs meet LO Redeker taxonomy? [Cooperative] Strongly agree



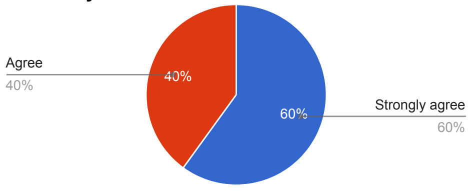
Does the developed SLOs meet LO Finlay taxonomy? [Theory-based] Strongly agree



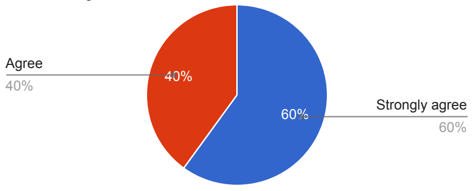
Does the developed SLOs meet LO Finlay taxonomy? [Cooperative] Strongly agree

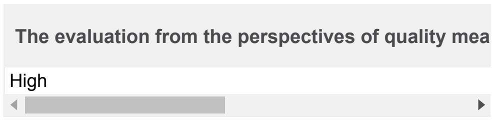


Does the developed SLOs meet LO Finlay taxonomy? [Application-based] Strongly agree



Does the developed SLOs meet LO Finlay taxonomy? [Individualised] Strongly agree

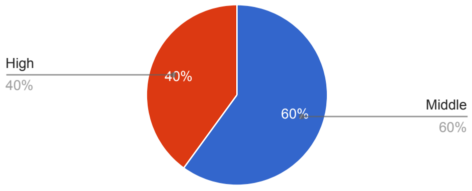




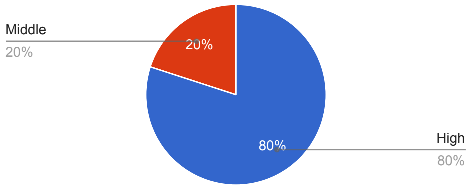
The evaluation from the perspectives of quality mea

High

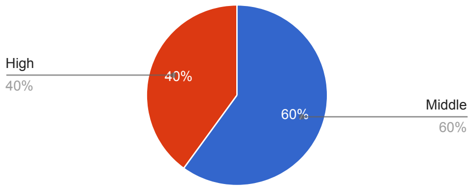
The evaluation from the perspectives of quality measurement metrics [Q2: quality of the LO co…



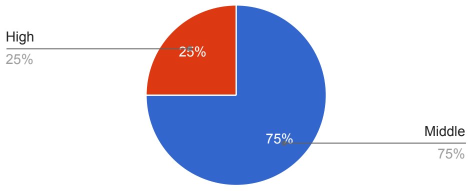
The evaluation from the perspectives of quality measurement metrics [Q3: quality of the descri…



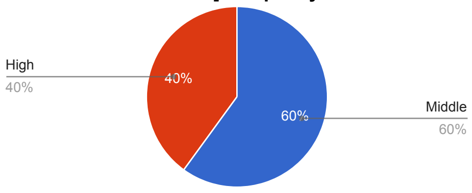
The evaluation from the perspectives of quality measurement metrics [Q4: quality of the LO co…



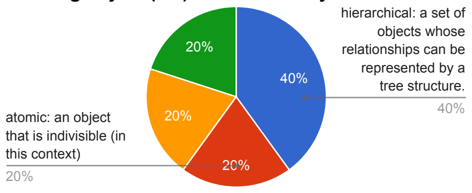
The evaluation from the perspectives of quality measurement metrics [Q5: quality of the LO co…



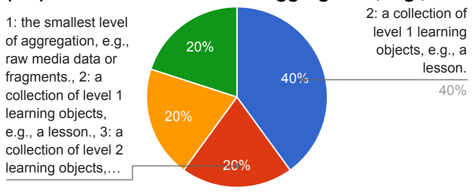
The evaluation from the perspectives of quality measurement metrics [Q6: quality of the LO co…



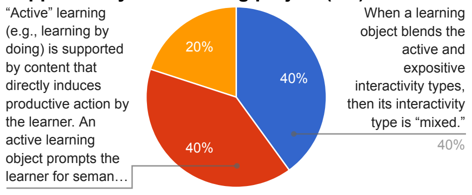
Underlying organizational structure of this learning object (1.7) atomic: an object that is in…



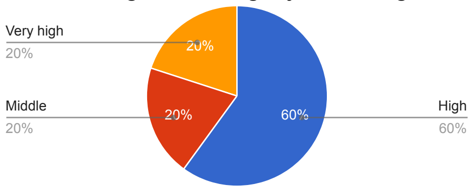
The functional granularity of this learning object (1.8) 1: the smallest level of aggregation, e.g., r…

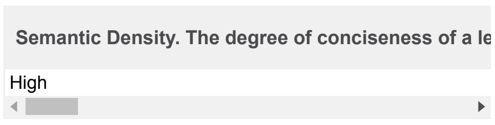


Interactivity Type. Predominant mode of learning supported by this learning project (5.1) “Active…



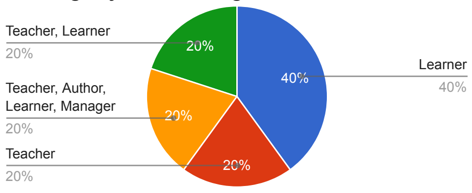
Interactivity Level. The degree of interactivity characterizing this learning object. The degree…



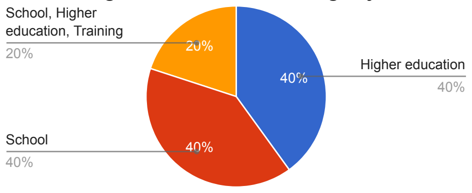

Semantic Density. The degree of conciseness of a le

High

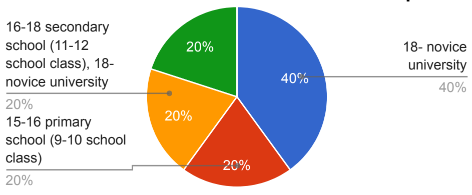
User Role. Principal user(s) for which this learning object was designed, most dominant fi…



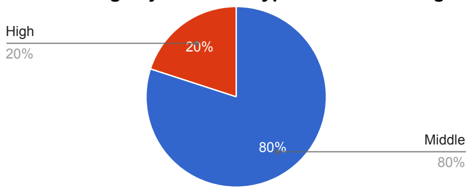
Context. The principal environment within which the learning and use of this learning object is in…



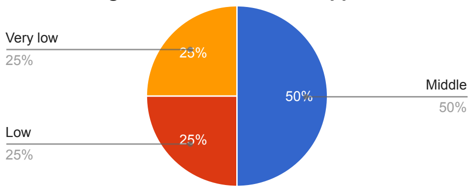
Typical Age Range. Age of the typical intended user.This data element shall refer to developme…



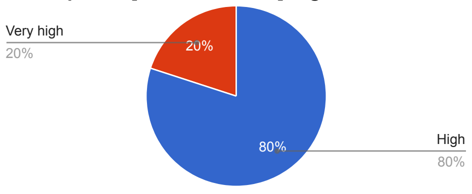
Difficulty. How hard it is to work with or through this learning object for the typical intended targ…



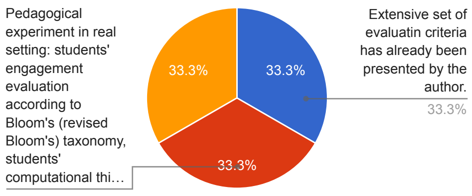
Please evaluate possible influence of disadvantages and limitations on application a…



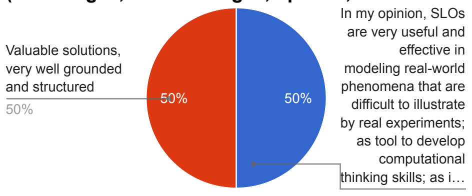
Please evaluate possibility of further development [Please indicate] High



What are (if any) additional evaluation criteria?



What are (if any) a supplement expert opinion (advantages, disadvantages, opinion, recomme…



References

[1] Charles M Reigeluth. *Instructional design theories and models: An overview of their current status*. Routledge, 2013. (page 4)

[2] Franco Landriscina. *Simulation and learning: the role of mental models*, pages 3072–3075 Springer, 2012. (page xiii, xiv, 4, 58, 104)

[3] Franco Landriscina. *Simulation and Learning*. Springer, 2013. (page xiii, xiv, 4, 49, 52, 53, 54, 55, 56, 81, 82, 84, 85, 89, 92, 95, 100, 102, 103)

[4] Vladimiras Dolgopolovas, Valentina Dagienė, Saulius Minkevičius, and Leonidas Sakalauskas. Python for scientific computing education: Modeling of queueing systems. *Scientific Programming*, 22(1):37–51, 2014. IOS Press. (page xiv, xv, 8, 109, 110, 111, 112, 137, 153, 154, 155, 156, 157, 158, 159, 160, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173)

[5] Vladimiras Dolgopolovas, Valentina Dagienė, Saulius Minkevičius, and Leonidas Sakalauskas. Teaching scientific computing: a model-centered approach to pipeline and parallel programming with c. *Scientific Programming*, 2015:18, 2015. Hindawi Publishing Corp. (page xiv, xv, 112, 114, 137, 176, 178, 179, 180, 181, 182, 183, 184, 185)

[6] Saulius Minkevičius, Vladimiras Dolgopolovas, and Leonidas L. Sakalauskas. A law of the iterated logarithm for the sojourn time process in queues in series. *Methodology and Computing in Applied Probability*, 18(1):37–57, 2016. Springer. (page xiv, xv, 111, 140, 141, 147, 149, 150, 151, 152, 162)

[7] Vladimiras Dolgopolovas, Tatjana Jevsikova, Valentina Dagienė, and Loreta Savulionienė. Exploration of computational thinking of software engineering novice students based on solving computer science tasks. *International Journal of Engineering Education*, 32(3):1–10, 2016. (page 46, 80)

[8] Vladimiras Dolgopolovas, Tatjana Jevsikova, and Valentina Dagienė. From android games to coding in c - an approach to motivate novice engineering students to learn programming: A case study. *Computer Applications in Engineering Education*, 26(1):75–90, 2018. (page 8, 46)

[9] Alan R. Hevner. A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4, 2007. (page xiii, 8, 9, 10, 16, 84)

[10] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, 28(1):75–105, 2004. Springer. (page 9, 16, 72)

[11] Vijay Vaishnavi and William Kuechler. Design research in information systems. *DSR journal*, 2004. (page xiii, xvi, 16, 69, 71, 72, 73, 74)

[12] M Arkus Helfert and Bri An Donnell An. Practical aspects of design science, 2012.

[13] Richard Baskerville, Jan Pries-Heje, and John Venable. Soft design science methodology. In *Proceedings of the 4th international conference on design science research in information systems and technology*, page 9. ACM, 2009.

[14] Romeo Botes and Roelien Goede. Bridging the gap for it students: Action research and design science research as research approaches for life-long learners. In *ISTE International Conference On Mathematics, Science Aand Technology Education*, page 330, 2014.

[15] The Design-Based Research Collective. Design-based research: An emerging paradigm for educational inquiry. *Educational Researcher*, pages 5–8, 2003.

[16] Shirley Gregor and Alan R Hevner. Positioning and presenting design science research for maximum impact. *MIS quarterly*, 37(2), 2013. (page 9, 16)

[17] D Hovorka. Design science research: A call for a pragmatic perspective. In *Proceedings of SIGPrag Workshop, Sprouts Working Papers on Information Systems*, 2009. (page 16)

[18] Paula Kotzé, Alta van der Merwe, and Aurona Gerber. Design science research as research approach in doctoral studies. *AMCIS 2015 Proceedings: Design Science Research*, pages 1–14, 2015. (page 9)

[19] Tobias Mettler, Markus Eurich, and Robert Winter. On the use of experiments in design science research: A proposition of an evaluation framework. *CAIS*, 34:10, 2014.

[20] Aline Dresch, Daniel Pacheco Lacerda, and José Antônio Valle Antunes Jr. *Design Science Research. A Method for Science and Technology Advancement.* Design Science Research. Springer, 2015. (page xiii, xiv, xvi, 69, 70, 85, 86, 97, 98)

[21] Sven Weber. Design science research: Paradigm or approach? In *AMCIS*, page 214, 2010. (page 8)

[22] Göran Goldkuhl. Pragmatism vs interpretivism in qualitative information systems research. *European Journal of Information Systems*, 21(2):135–146, 2012. (page 16, 24)

[23] Jean Piaget. *The construction of reality in the child*, volume 82 1136316949. Routledge, 2013. (page 16, 50)

[24] Seymour Papert. *Mindstorms: Children, computers, and powerful ideas.* Basic Books, Inc., 1980. (page 16, 51, 103)

[25] Seymour Papert and Idit Harel. Situating constructionism. *Constructionism*, 36:1–11, 1991. (page 16, 44, 95)

[26] Andrew S. Gibbons. Model-centered instruction. *Journal of Structural Learning and Intelligent Systems*, 14(4):511–540, 2001. (page 16, 49)

[27] Philip Nicholas Johnson-Laird. *Mental models: Towards a cognitive science of language, inference, and consciousness.* Harvard University Press, 1983. 6 (page 16, 51)

[28] Nancy J. Nersessian. *Creating scientific concepts.* MIT press, 2010. (page 16, 57, 67, 68)

[29] Lorenzo Magnani. *Model-based creative abduction*, pages 219–238. Springer, 1999. (page 93)

[30] Lorenzo Magnani, Walter Carnielli, and Claudio Pizzi. *Model-based reasoning in science and technology: Abduction, logic, and computational discovery*, volume 314 Springer, 2010. (page 93)

[31] Paul Thagard. *How brains make mental models*, pages 447–461. Springer, 2010. (page 93)

[32] Paul Thagard and Terrence C. Stewart. The aha! experience: Creativity through emergent binding in neural networks. *Cognitive science*, 35(1):1–33 1551–6709, 2011. Wiley Online Library. (page 16, 93)

[33] Vytautas Štuikys. *Smart Learning Objects for Smart Education in Computer Science: Theory, Methodology and Robot-Based Implementation.* Springer, 2015. (page xiii, 16, 61, 62, 63, 64, 65, 66, 77)

[34] Vytautas Štuikys and Robertas Damasevicius. Development of generative learning objects using feature diagrams and generative techniques. *Informat-*

ics in Education, 7(2):277–288, 2008. (page 16)

[35] Herbert A. Simon. *The sciences of the artificial.* MIT press, 1996. (page 16, 69)

[36] Christian Sonnenberg and Jan vom Brocke. Evaluation patterns for design science research artefacts. In *European Design Science Symposium*, pages 71–83. Springer, 2011. (page 16)

[37] Christian Sonnenberg and Jan vom Brocke. Evaluations in the science of the artificial-reconsidering the build-evaluate pattern in design science research. *Design Science Research in Information Systems. Advances in Theory and Practice*, pages 381–397, 2012. (page xiii, 16)

[38] Renata Burbaitė. *Advanced generative learning objects in informatics education: the concept, models, and implementation. Summary of doctoral dissertation, physical sciences, informatics (09P).* PhD thesis, Kaunas University of Technology, 2014. (page xiii, 24, 25, 77, 78)

[39] Charles Sanders Peirce and Nathan Houser. *The essential Peirce: selected philosophical writings*, volume 2 Indiana University Press, 1998. (page 24, 49, 93)

[40] Cleo H Cherryholmes. Notes on pragmatism and scientific realism. *Educational researcher*, 21(6):13–17, 1992. (page 24)

[41] Alan Malachowski. *The new pragmatism.* Routledge, 2014.

[42] Richard Rorty. *Philosophy and the Mirror of Nature.* Princeton University Press, 2009. (page 24)

[43] Michael J Flynn. Very high-speed computing systems. *Proceedings of the IEEE*, 54(12):1901–1909, 1966. (page 26)

[44] David B Skillicorn. *Foundations of parallel programming.* Cambridge University Press, 2005. (page xiv, 26, 120, 122, 123, 125, 126, 129)

[45] G .I. Ivcenko, V.A. Kastanov, and I.N. Kovalenko. *Queuing system theory.* Vishaja Skola, Moscow, 1982. (page 26, 142, 144, 145, 163, 173, 177)

[46] Alan F. Blackwell, Lee Wilson, Alice Street, Charles Boulton, and John Knell. Radical innovation: crossing knowledge boundaries with interdisciplinary teams. *University of Cambridge/NESTA Report. Cambridge, UK: University of Cambridge Computer Laboratory. See http://www-test. cl. cam. ac. uk/techreports/UCAMCL-TR-760. pdf*, 2009. Citeseer. (page 28, 30)

[47] Insead Cornell. Cornell university, insead, and wipo (2016): The global innovation index 2016: Winning with global innovation, 2016. (page 29)

[48] Lim Chuan Poh. From research to innovation to enterprise: The case of singapore, 2016. (page 29)

[49] Alan Finkel and John Bell. National innovation systems contributing to global innovation: The case of australia, 2016. (page 29)

[50] Stanford University. Stanford interdisciplinary, 2016. (page 29)

[51] University of Cambridge. Research, 2016. (page 29)

[52] University of Turku. Effective research, 2016. (page 29)

[53] Tony Townsend, John Pisapia, and Jamila Razzaq. Fostering interdisciplinary research in universities: A case study of leadership, alignment and support. *Studies in Higher Education*, 40(4):658–675 Taylor & Francis. (page 29, 31)

[54] Scimago Lab. Subjects bubble chart. SCImago, (n.d.). SJR – SCImago Journal & Country Rank [Portal]. Retrieved 21.05.2018, from http://www.scimagojr.com, 2016. (page xiii, 30, 32)

[55] National Academies Keck Future Initiative et al. *The National Academies Keck Future Initiative: The Informed Brain in a Digital World: Interdisciplinary Research Team Summaries.* National Academies Press, 2013. (page 30)

[56] Tom McLeish and V. Strang. *Leading interdisciplinary research: transforming the academic landscape. Stimulus paper.* Leadership Foundation for Higher Education, 2014. ST-28 (page 31)

[57] Pier Giuseppe Rossi and Laura Fedeli. Interdisciplinarity: rhetoric or the latestpromising new field? *Education Sciences & Society*, 5(1), 2014. (page 31)

[58] Stanford University. Interdisciplinary programs, 2016. (page 31)

[59] Peter Smith. From a multidisciplinary to an interdisciplinary curriculum: a case study of curriculum innovation. *Journal*, 2013. (page 31, 32)

[60] Casey Jones. Interdisciplinary approach-advantages, disadvantages, and the future benefits of interdisciplinary studies. *ESSAI*, 7(1):26, 2010. (page 32)

[61] Victoria Millar. Interdisciplinary curriculum reform in the changing university. *Teaching in Higher Education*, 21(4):471–483 Taylor & Francis. (page 33)

[62] Elisabeth Zimmermann, Markus F Peschl, and Brigitte Römmer-Nossek. Constructivist curriculum design for the interdisciplinary study programme mei: Cogsci–a case study. *Constructivist Foundations*, 5(3), 2010. (page 33)

[63] Adi Kidron and Yael Kali. Boundary breaking for interdisciplinary learning. *Research in Learning Technology*, 23 (page 33)

[64] Katerine Bielaczyc, Manu Kapur, and Allan Collins. Cultivating a community of learners in k-12 classrooms. *International handbook of collaborative learning*, pages 233–249, 2013. Routledge, New York. (page 33)

[65] Allan Collins. Cognitive apprenticeship: The cambridge handbook of the learning sciences, r. keith sawyer, 2006. (page 33)

[66] Gene H. Golub and James M. Ortega. *Scientific computing: an introduction with parallel computing.* Elsevier, 2014. (page xiii, 34, 35, 37, 39, 83)

[67] George Em Karniadakis and Robert M. Kirby Ii. *Parallel scientific computing in C++ and MPI: a seamless approach to parallel algorithms and their implementation*, volume 1 110749477X. Cambridge University Press, 2003. (page xiii, 36, 38, 41, 43)

[68] Michael T. X. Heath. *Scientific computing.* McGraw-Hill New York, 2002. (page 36, 38)

[69] Ionut Danaila, Pascal Joly, Sidi Mahmoud Kaber, and Marie Postel. *An introduction to scientific computing: Twelve computational projects solved with MATLAB.* Springer Science & Business Media, 2007. (page 37, 40)

[70] Sean Chester, Celina Gibbs, Fil Rossi, Andrew Brownsword, Poman So, and Aaron Gulliver. Insulating the scientific programmer from perilous parallel architecture. In *Proceedings of the 9th Workshop on Parallel/High-Performance Object-Oriented Scientific Computing*, page 3. ACM, 2010. (page 41)

[71] Elizabeth R Jessup and Henry M Tufo. Creating a sustainable high-performance scientific computing course. In *International Conference on Computational Science*, pages 1242–1248. Springer, 2004. (page 37)

[72] Gabrielle Allen, Werner Benger, Andrei Hutanu, Shantenu Jha, Frank Löffler, and Erik Schnetter. A practical and comprehensive graduate course prepar-

ing students for research involving scientific computing. *Procedia Computer Science*, 4:1927–1936 1877–0509, 2011. (page 38)

[73] Gene H. Golub and James M. Ortega. *Scientific computing and differential equations: an introduction to numerical methods.* Academic Press, 2014. (page 39)

[74] W. H. Steeb. *Problems & Solutions in Scientific Computing: With C++ and Java Simulations.* World Scientific, 2004. (page 39)

[75] Tobias Neckel and Florian Rupp. *Random Differential Equations in Scientific Computing.* Walter de Gruyter, 2013. (page xiii, 39, 40)

[76] Peter R Turner. Teaching scientific computing through projects. *Journal of Engineering Education*, 90(1):79–83, 2001. (page 40)

[77] Henry Neeman, Lloyd Lee, Julia Mullen, and Gerard Newman. Analogies for teaching parallel computing to inexperienced programmers. In *ACM SIGCSE Bulletin*, volume 38, pages 64–67. ACM, 2006. (page 41)

[78] Omar Abuzaghleh, Kathleen Goldschmidt, Yasser Elleithy, and Jeongkyu Lee. Implementing an affordable high-performance computing for teaching-oriented computer science curriculum. *ACM Transactions on Computing Education (TOCE)*, 13(1):3, 2013. ACM.

[79] Hans Petter Langtangen, Timothy J Barth, and Michael Griebel. *Python scripting for computational science*, volume 3. Springer, 2006.

[80] Hans Petter Langtangen. *A primer on scientific programming with Python*, volume 2. Springer, 2009.

[81] Svein Linge and Hans Petter Langtangen. *Programming for Computations-MATLAB/Octave: A Gentle Introduction to Numerical Simulations with MATLAB/Octave*, volume 14. Springer, 2016.

[82] John J Barton and Lee R Nackman. *Scientific and Engineering C++: an introduction with advanced techniques and examples.* Addison-Wesley Longman Publishing Co., Inc., 1994.

[83] Michael Kupferschmid. *Classical Fortran: Programming for engineering and scientific applications.* CRC Press, 2009.

[84] Joseph L. Zachary. *Introduction to scientific programming: computational problem solving using Maple and C.* Springer Science & Business Media, 2012. (page 41)

[85] Peter S. Pacheco. *Parallel programming with MPI.* Morgan Kaufmann, 1997. (page 41)

[86] William Gropp, Ewing Lusk, and Rajeev Thakur. *Using MPI-2: Advanced features of the message-passing interface.* MIT press, 1999.

[87] Rohit Chandra. *Parallel programming in OpenMP.* Morgan Kaufmann, 2001.

[88] Haoqiang Jin, Dennis Jespersen, Piyush Mehrotra, Rupak Biswas, Lei Huang, and Barbara Chapman. High performance computing using mpi and openmp on multi-core parallel systems. *Parallel Computing*, 37(9):562–575 0167–8191, 2011. Elsevier. (page 41)

[89] Michael A. Heroux, Andrew G. Salinger, and Laura J. D. Frink. Parallel segregated schur complement methods for fluid density functional theories. *SIAM Journal on Scientific Computing*, 29(5):2059–2077 (page 42)

[90] Sushil K Prasad, Almadena Yu Chtchelkanova, Sajal K Das, Frank Dehne, Mohamed G Gouda, Anshul Gupta, Joseph Jaja, Krishna Kant, Anita La Salle, Richard LeBlanc, et al. Nsf/ieee-tcpp curriculum initiative on par-

allel and distributed computing: core topics for undergraduates. In *SIGCSE*, volume 11, pages 617–618, 2011. (page 42)

[91] Gonzalo Zarza, Diego Lugones, Daniel Franco, and Emilio Luque. An innovative teaching strategy to understand high-performance systems through performance evaluation. *Procedia Computer Science*, 9:1733–1742 1877–0509, 2012. (page 42)

[92] Barry Wilkinson, Jeremy Villalobos, and Clayton Ferner. Pattern programming approach for teaching parallel and distributed computing. In *Proceeding of the 44th ACM technical symposium on Computer science education*, pages 409–414. ACM, 2013. (page 42)

[93] Javier Iparraguirre, Guillermo R Friedrich, and Ricardo J Coppo. Lessons learned after the introduction of parallel and distributed computing concepts into ece undergraduate curricula at utn-bahia blanca argentina. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 1317–1320. IEEE, 2012. (page 42)

[94] Hans Petter Langtangen. *Scientific Software Engineering*, pages 127–187. Springer, 2016. (page 42)

[95] MD Eddy. Fallible or inerrant? A belated review of the 'constructivist's bible' Jan Golinski, Making natural knowledge: Constructivism and the history of science. Cambridge history of science. Cambridge: Cambridge University Press, 1999. isbn 0-521-44913-8. *The British Journal for the History of Science*, 37(1):93–98, 2004. (page 44)

[96] E. von Glasersfeld. Constructivism in education. *The international encyclopedia of education*, 1:162–163, 1989. (page 44)

[97] Renate Nummela Caine and Geoffrey Caine. Making connections: teaching and the human brain. *Jrn*, 1991. (page 44)

[98] Kathryn Alesandrini and Linda Larson. Teachers bridge to constructivism. *The Clearing House*, 75(3):118–121 0009–8655, 2002. (page 44)

[99] Mordechai Ben-Ari. Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching*, 20(1):45–73 (page 45, 46)

[100] Tom Wulf. Constructivist approaches for teaching computer programming. In *Proceedings of the 6th conference on Information technology education*, pages 245–248. ACM, 2005. (page 45)

[101] Sherry Turkle and Seymour Papert. Epistemological pluralism: Styles and voices within the computer culture. *Signs*, pages 128–157 (page 45)

[102] Said Hadjerrouit. Constructivism as guiding philosophy for software engineering education. *ACM SIGCSE Bulletin*, 37(4):45–49 0097–8418, 2005. (page 46)

[103] Jeannette M. Wing. Computational thinking and thinking about computing. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences*, 366(1881):3717–3725 The Royal Society. (page 46, 82)

[104] Valentina Dagienė. Information technology contests-introduction to computer science in an attractive way. *Informatics in education*, 5(1):37, 2006. (page 46)

[105] Yasmin B Kafai and Quinn Burke. The social turn in k-12 programming: moving from computational thinking to computational participation. In *Pro-*

ceeding of the 44th ACM technical symposium on computer science education, pages 603–608. ACM, 2013. (page 46)

[106] Diana X. Laurillard. *Rethinking university teaching: A conversational framework for the effective use of learning technologies.* Routledge, 2013. (page 47, 50, 92, 102, 103)

[107] Yishay Mor, Harvey Mellar, Steven Warburton, and Niall Winters. *Practical design patterns for teaching and learning with technology.* Springer, 2014. (page 48)

[108] Paolo Rocchi. Questioning two myths in computer science education. In *IFIP Conference on Information Technology in Educational Management*, pages 34–41. Springer, 2014. (page 48)

[109] Eva Magnusson. Pedagogical patterns–a method to capture best practices in teaching and learning. *[Host publication title missing]*, 2006. (page 48)

[110] Antonio Miguel Seoane Pardo and Francisco José García-Peñalvo. Pedagogical patterns and online teaching. *Online tutor*, 2:298–316, 2014.

[111] Lori Lockyer, Sue Bennett, Shirley Agostinho, and Barry Harper. Handbook of research on learning design and learning objects: issues, applications, and technologies (2 volumes). *IGI Global, Hershey, PA*, 2009.

[112] Sally Fincher and Ian Utting. Pedagogical patterns: their place in the genre. In *ACM SIGCSE Bulletin*, volume 34, pages 199–202. ACM, 2002. (page 48)

[113] Norbert M. Seel. Model-centered learning and instruction. *Technology, Instruction, Cognition and Learning*, 1(1):59–85, 2003. (page 49)

[114] Herbert Stachowiak. Medicine and the paradigm of neo-pragmatism a contribution to medical decision theory. *Theory and decision*, 21(2):189–207, 1986. (page 49)

[115] John J. Clement and Mary Anne Rea-Ramirez. Model based learning and instruction in science. *Model based learning and instruction in science*, pages 1–9, 2008. Springer. (page xiii, 49, 50, 51, 52)

[116] Lev Semenovich Vygotsky. *Mind in society: The development of higher psychological processes.* Harvard university press, 1980. (page 51)

[117] James F. Voss, David N. Perkins, and Judith W. Segal. *Informal reasoning and education.* Routledge, 2012. (page 51)

[118] Fred M. Newmann, J. Voss, D. Perkins, and J. Segal. Higher order thinking in the teaching of social studies: Connections between theory and practice. *Informal reasoning and education*, pages 381–400, 1991. (page 51)

[119] Dedre Gentner and Catherine Clement. Evidence for relational selectivity in the interpretation of analogy and metaphor. *Psychology of Learning and Motivation*, 22:307–358 Elsevier. (page 51)

[120] John Clement. Model based learning as a key research area for science education. *International Journal of Science Education*, 22(9):1041–1053 Taylor & Francis. (page 51)

[121] Marcelo Milrad, J. Michael Spector, and Pal I. Davidsen. Model facilitated learning. *Learning and teaching with technology: Principles and practices*, pages 13–27, 2003. (page 51)

[122] Richard Lehrer and Leona Schauble. *Cultivating model-based reasoning in science education.* Cambridge University Press, 2006. (page 52)

[123] Lian Xue, Ming-hui Wu, Hui Zheng, Hui-zeng Zhang, and Wai-bin Huang. Modeling and simulation in scientific computing education. In *Scalable Com-*

puting and Communications; Eighth International Conference on Embedded Computing, 2009. SCALCOM-EMBEDDEDCOM'09. International Conference on, pages 577–580. IEEE, 2009. (page 52)

[124] Lingguo Bu and Robert Schoen. *Model-centered learning: Pathways to mathematical understanding using GeoGebra*, volume 6 Springer Science & Business Media, 2012. (page 52)

[125] Myint Swe Khine and Issa M. Saleh. *Models and modeling: Cognitive tools for scientific enquiry*, volume 6 Springer Science & Business Media, 2011. (page 52)

[126] W. Ross Ashby. Analysis of the system to be modeled. *The process of model building in the behavioral sciences*, pages 94–114, 1970. (page 52)

[127] Stewart Robinson. *Simulation: the practice of model development and use.* Palgrave Macmillan, 2014. (page 53)

[128] Eric Winsberg. *Science in the age of computer simulation.* University of Chicago Press, 2010. (page 54)

[129] Tarja Susi, Mikael Johannesson, and Per Backlund. Serious games: An overview, 2007. (page 54)

[130] David R. Michael and Sandra L. Chen. *Serious games: Games that educate, train, and inform.* Muska & Lipman/Premier-Trade, 2005. (page 54)

[131] James X. Jadrich. *Learning & teaching scientific inquiry: Research and applications.* NSTA press, 2011. (page 54, 67, 84, 85)

[132] Richard E Mayer. *The Cambridge handbook of multimedia learning.* Cambridge university press, 2005. (page 56)

[133] Richard E Mayer. Multimedia learning. In *Psychology of learning and motivation*, volume 41, pages 85–139. Elsevier, 2002. (page 56)

[134] Stanislaw Raczynski. *Modeling and simulation: the computer science of illusion.* John Wiley & Sons, 2014. (page xiii, 57, 58)

[135] Paul Bratley, Bennet L. Fox, and Linus E. X. Schrage. *A guide to simulation.* Springer Science & Business Media, 2011. (page 57)

[136] Bernard P Zeigler. Simulation based structural complexity of models. *International Journal of General Syatem*, 2(1):217–223, 1975. (page 57)

[137] Bernard P. Zeigler. *Object-oriented simulation with hierarchical, modular models: intelligent agents and endomorphic systems.* Academic press, 2014. (page 57)

[138] Daniel Churchill. Towards a useful classification of learning objects. *Educational Technology Research and Development*, 55(5):479–497 (page 59, 60, 191)

[139] Gwen Nugent, Leen-Kiat Soh, Ashok Samal, Suzette Person, and Jeff Lang. Design, development, and validation of a learning object for cs1. *ACM SIGCSE Bulletin*, 37(3):370–370, 2005. (page 60)

[140] Lee Dee Miller, Leen-Kiat Soh, Gwen Nugent, Kevin Kupzyk, Leyla Masmaliyeva, and Ashok Samal. Evaluating the use of learning objects in cs1. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 57–62. ACM, 2011. (page 61)

[141] A Matthiasdottir. Usefulness of learning objects in computer science learning. In *Proceedings of Codewitz Open Conference Methods, Materials and Tools for Programming Education*, 2006. (page 61)

[142] Tom Boyle. Design principles for authoring dynamic, reusable learning ob-

jects. *Australasian Journal of Educational Technology*, 19(1), 2003. (page 64)

[143] Science American Association for the Advancement of. *Benchmarks for science literacy.* Oxford University Press, 1994. (page 67)

[144] Steve Olson and Susan Loucks-Horsley. *Inquiry and the National Science Education Standards: A guide for teaching and learning.* National Academies Press, 2000. (page 67)

[145] L. Flick and N. G. Lederman. Scientific inquiry and nature of science. *Contemporary Trends and Issues in Science Education*, 2004. Springer. (page 67)

[146] Mark Windschitl, Jessica Thompson, and Melissa Braaten. Beyond the scientific method: Model-based inquiry as a new paradigm of preference for school science investigations. *Science education*, 92(5):941–967 1098–237X, 2008. Wiley Online Library. (page 67, 85)

[147] A. Georges L. Romme. Making a difference: Organization as design. *Organization science*, 14(5):558–573 1047–7039, 2003. INFORMS. (page 69)

[148] Richard Smith and David Lynch. *Rethinking teacher education.* Lulu. com, 2010. (page 69)

[149] N. Kaloyanova. Teacher-manager status and roles in the teaching profession. *Tracian Journal of Science*, 8(1):358–364, 2010.

[150] Judith Romonini and Joy Higgs. The teacher as manager in continuing and professional education. *Studies in Continuing Education*, 13(1):41–52 Taylor & Francis. (page 69)

[151] Diana Laurillard. *Teaching as a design science: Building pedagogical patterns for learning and technology.* Routledge, 2013. (page xiii, 69, 75, 76)

[152] Mark N.K. Saunders, Philip Lewis, and Adrian Thornhill. *Research methods for business students (7th ed.).* Pearson Education Limited, London, 2015. (page 69)

[153] Paul Johannesson. *Information Systems Engineering: From Data Analysis to Process Networks: From Data Analysis to Process Networks.* IGI Global, 2008. (page 71)

[154] Salvatore T. March and Gerald F. Smith. Design and natural science research on information technology. *Decision support systems*, 15(4):251–266 Elsevier. (page 72)

[155] Peter C. Taylor and Milton Norman D. Medina. Educational research paradigms: From positivism to multiparadigmatic. *The journal of Meaning-centered education*, 1(2):1–13, 2013. (page 74)

[156] Geraldine O'Neill. Curriculum design in higher education: Theory to practice, 2015. (page 77)

[157] Karri A Holley. *Understanding interdisciplinary challenges and opportunities in higher education.* Jossey-Bass, 2009. (page 77, 79, 80, 81, 82)

[158] R Pring. John dewey: Continuum library of educational thought. *London: Continuum*, 2007. (page 79)

[159] Marius Boboc. The postmodern curriculum in a modern classroom. *International Journal of Education*, 4(1):142, 2012. (page 79)

[160] Lisa R Lattuca. *Creating interdisciplinarity: Interdisciplinary research and teaching among college and university faculty.* Vanderbilt university press, 2001. (page 79)

[161] Julie Thompson Klein. *Mapping Interdisciplinary Studies. The Academy in Transition.* ERIC, 1999. (page 79)

[162] V Boix Mansilla. Learning to synthesize: the development of interdisciplinary understanding. *The Oxford Handbook of Interdisciplinarity. Oxford University Press, Oxford*, pages 288–306, 2010. (page 79)

[163] Elisabeth JH Spelt, Harm JA Biemans, Hilde Tobi, Pieternel A Luning, and Martin Mulder. Teaching and learning in interdisciplinary higher education: A systematic review. *Educational Psychology Review*, 21(4):365, 2009. (page 79)

[164] William H Newell, Jay Wentworth, and David Sebberson. A theory of interdisciplinary studies. *Issues in Interdisciplinary Studies*, 2001. (page 80)

[165] Julie Thompson Klein. *Interdisciplinarity: History, theory, and practice.* Wayne state university press, 1990. (page 80)

[166] Deborah DeZure. Interdisciplinary pedagogies in higher education. *The Oxford handbook of interdisciplinarity*, pages 372–386, 2010. (page 80)

[167] Martin Davies and Marcia T. Devlin. *Interdisciplinary higher education: Implications for teaching and learning.* Centre for the Study of Higher Education, 2007. (page 81)

[168] Gerald Dawe, Rolf Jucker, and Stephen Martin. Sustainable development in higher education: current practice and future developments. *A report for the Higher Education Academy*, page 87, 2005. (page 81)

[169] Lisa M. Campbell. Overcoming obstacles to interdisciplinary research. *Conservation biology*, 19(2):574–577 1523–1739, 2005. Wiley Online Library. (page 81)

[170] Lisa Gueldenzoph Snyder and Mark J. Snyder. Teaching critical thinking and problem solving skills. *The Journal of Research in Business Education*, 50 (2):90, 2008. Delta Pi Epsilon. (page 82)

[171] Karri A Holley. Special issue: Understanding interdisciplinary challenges and opportunities in higher education. *ASHE Higher Education Report*, 35(2): 1–131, 2009. (page 82)

[172] Alexander Joseph Romiszowski. *Designing instructional systems: Decision making in course planning and curriculum design.* Routledge, 2016. (page xiii, 83)

[173] Osman Yasar and Rubin H Landau. Elements of computational science and engineering education. *SIAM review*, 45(4):787–805, 2003. (page 87)

[174] Osman Yasar and Jose Maliekal. Computational pedagogy: A modeling and simulation approach. *Computing in Science & Engineering*, 16(3):78–88 AIP Publishing. (page xiii, 88, 90)

[175] Osman Yasar. A universal process: How mind and matter seem to work. *Science Discovery*, 3(6):79–87, 2015.

[176] Osman Yasar. Computational pedagogy: Fostering a new method of teaching, 2016. (page 89)

[177] Osman Yasar. Teaching science through computation. *International Journal of Science, Technology and Society*, 1(2):9–18, 2013. (page 87)

[178] Michael Prince and Richard Felder. The many faces of inductive teaching and learning. *Journal of college science teaching*, 36(5):14 Citeseer. (page 88)

[179] John D. Bransford, Ann L. Brown, and Rodney R. Cocking. *How people learn: Brain, mind, experience, and school.* National Academy Press, 1999.

(page 88)

[180] Michael J. Prince and Richard M. Felder. Inductive teaching and learning methods: Definitions, comparisons, and research bases. *Journal of engineering education*, 95(2):123–138 Wiley Online Library. (page 88)

[181] Peter C. Brown, Henry L. Roediger, and Mark A. McDaniel. *Make it stick*. Harvard University Press, 2014. (page 89)

[182] Osman Yasar. Cognitive aspects of computational modeling and simulation in teaching and learning. *Computational Science Education*, 7(1), 2016. (page 89)

[183] Margaret A. Honey and Margaret Hilton. *Learning science through computer games and simulations*. National Academies Press, 2011. (page 90)

[184] Donald A. Norman. *Cognitive artifacts*. Department of Cognitive Science, University of California, San Diego, 1990. (page xiii, 90, 91)

[185] Sean Coughlan. Computers 'do not improve' pupil results, says oecd. *BBC News, viewed*, 10, 2015. (page 91)

[186] Matt Richtel. A silicon valley school that doesn't compute. *The New York Times*, 22, 2011. (page 91)

[187] Carl E. Wieman, Wendy K. Adams, and Katherine K. Perkins. Phet: Simulations that enhance learning. *Science*, 322(5902):682–683 2008. American Association for the Advancement of Science. (page 91)

[188] Lawrence W. Barsalou. Grounded cognition. *Annu. Rev. Psychol.*, 59:617–645 2008. Annual Reviews. (page 92)

[189] Lawrence W. Barsalou. Grounded cognition: Past, present, and future. *Topics in cognitive science*, 2(4):716–724 Wiley Online Library. (page 92)

[190] Giovanni Pezzulo, Lawrence W. Barsalou, Angelo Cangelosi, Martin H. Fischer, Ken McRae, and Michael Spivey. Computational grounded cognition: a new alliance between grounded cognition and computational modeling. *Frontiers in psychology*, 3:612 Frontiers. (page 92)

[191] John McCarthy. Circumscription - a form of non-monotonic reasoning. In *Readings in Artificial Intelligence*, pages 466–472. Elsevier, 1981. (page 93)

[192] Rosária S. Justi and John K. Gilbert. Modelling, teachers' views on the nature of modelling, and implications for the education of modellers. *International Journal of Science Education*, 24(4):369–387 Taylor & Francis. (page 93)

[193] Rosária Justi. Learning how to model in science classroom: key teacher's role in supporting the development of students' modelling skills. *Educación química*, 20(1):32–40, 2009. (page xiii, 94)

[194] Tim Bell, Jason Alexander, Isaac Freeman, and Mick Grimley. Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13(1):20–29, 2009. (page 95)

[195] Stephanie Bell. Project-based learning for the 21st century: Skills for the future. *The Clearing House*, 83(2):39–43 0009–8655, 2010. Taylor & Francis. (page 97)

[196] Julie E. Mills and David F. Treagust. Engineering education - is problem-based or project-based learning the answer. *Australasian journal of engineering education*, 3(2):2–16, 2003.

[197] Gwen Solomon. Project-based learning: A primer. *TECHNOLOGY AND LEARNING-DAYTON-*, 23(6):20–20 PETER LI, INC. (page 97)

[198] Peter M. Senge and Robert M. Fulmer. Simulations, systems thinking and anticipatory learning. *Journal of Management Development*, 12(6):21–33 MCB UP Ltd. (page 103)

[199] J. Cannon-Bowers, C. Bowers, and Alicia Sanchez. Using synthetic learning environments to train teams. *Work group learning: Understanding, improving and assessing how groups learn in organizations*, pages 315–346, 2008.

[200] Norbert M. Seel and Patrick Blumschein. Modeling and simulation in learning and instruction: A theoretical perspective. *Model-based approaches to learning: Using systems models and simulations to improve understanding and problem solving in complex domains*, pages 3–15, 2009. (page 103)

[201] John Seely Brown, Allan Collins, and Paul Duguid. Situated cognition and the culture of learning. *Educational researcher*, 18(1):32–42 0013–189X, 1989. Sage Publications. (page 103)

[202] Michael Polanyi. The logic of tacit inference. *Philosophy*, 41(155):1–18 1966. Cambridge Univ Press. (page 103)

[203] Jacqueline Senker. Tacit knowledge and models of innovation. *Industrial and corporate change*, 4(2):425–447 Oxford Univ Press. (page 103)

[204] Wlodzimierz Bryc. *Applied Probability and Stochastic Processes. Lecture Notes.* University of Cincinnati, Cincinnati, OH, 1995. (page 105)

[205] Marius Iosifescu, Nikolaos Limnios, and Gheorghe Oprisan. *Introduction to stochastic models.* John Wiley & Sons, 2013. (page 105, 106)

[206] C. C. Heyde. On fibonacci (or lagged bienaymé-galton-watson) branching processes. *Journal of Applied Probability*, pages 583–591 1981. JSTOR. (page 105, 106)

[207] Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, random variables, and stochastic processes.* Tata McGraw-Hill Education, 2002. (page 106)

[208] Howard M. Taylor and Samuel Karlin. *An introduction to stochastic modeling.* Academic press, 2014. (page 106)

[209] Raúl Toral and Pere Colet. *Stochastic numerical methods: an introduction for students and scientists.* John Wiley & Sons, 2014. (page 106)

[210] Barry L. Nelson. *Stochastic modeling: analysis and simulation.* Courier Corporation, 2012. (page 109)

[211] C. Newell. *Applications of queueing theory*, volume 4 Springer Science & Business Media, 2013.

[212] Natarajan Gautam. *Analysis of queues: methods and applications.* CRC Press, 2012.

[213] Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications.* John Wiley & Sons, 2006. (page 109)

[214] Seyed H Roosta. *Parallel processing and parallel algorithms: theory and computation.* Springer Science & Business Media, 2012. (page xiv, 120, 121, 122, 123, 124)

[215] Garry Rodrigue. *Parallel computations*, volume 1. Elsevier, 2014. (page xiv, 121)

[216] Georg Hager and Gerhard Wellein. *Introduction to high performance computing for scientists and engineers.* CRC Press, 2010. (page 122)

[217] Blaise Barney et al. Introduction to parallel computing. *Lawrence Livermore National Laboratory*, 6(13):10, 2010. (page xiv, 124)

[218] Rolf Rabenseifner. Hybrid parallel programming on hpc platforms. In *proceedings of the Fifth European Workshop on OpenMP, EWOMP*, volume 3, pages 185–194, 2003. (page 126)

[219] Sergei Gorlatch and Murray Cole. Parallel skeletons. In *Encyclopedia of parallel computing*, pages 1417–1422. Springer, 2011. (page 126)

[220] Fethi Rabhi. *Patterns and skeletons for parallel and distributed computing.* Springer Science & Business Media, 2003. (page 126)

[221] Rita Loogen, Yolanda Ortega-Mallén, and Ricardo Peña-Marí. Parallel functional programming in eden. *Journal of Functional Programming*, 15(3):431–475, 2005. (page xiv, 126, 127, 128, 129)

[222] Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001. (page 131)

[223] Christian P Robert. *Monte carlo methods.* Wiley Online Library, 2004. (page 131)

[224] Ilya M Sobol. *A primer for the Monte Carlo method.* CRC press, 1994. (page 131)

[225] M Loève. Elementary probability theory. In *Probability Theory I*, pages 1–52. Springer, 1977. (page 132)

[226] MG Borovcnik and Ramesh Kapadia. Research and developments in probability education internationally. In *Proceedings of the British Congress for Mathematics Education*, pages 41–48, 2010. (page 154)

[227] Michael T. Heath. *Scientific Computing An Introductory Survey.* The McGraw-Hill Companies, New York, second edition, 1997. (page 154, 157)

[228] George Levy. An introduction to quasi-random numbers. *Numerical Algorithms Group Ltd., http://www. nag. co. uk/IndustryArticles/introduction_to_quasi_random_numbers. pdf (last accessed in April 10, 2012)*, page 143, 2002. (page 154)

[229] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM, 1992. (page 154)

[230] Jun S. Liu. *Monte Carlo Strategies in Scientific Computing.* Springer Series in Statistics. Harvard University, 2001. (page 157)

[231] Andreas Hellander. Stochastic simulation and monte carlo methods. *Available in: http://www. it. uu. se/edu/course/homepage/bervet2/MCkompendium/mc. pdf*, 2009. (page 157)

[232] David G Kendall. Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain. *The Annals of Mathematical Statistics*, pages 338–354, 1953. (page 161, 176)

[233] Rathindra P Sen. *Operations Research: Algorithms and Applications.* PHI Learning, 2010. (page 161, 176)

[234] János Sztrik. Finite-source queueing systems and their applications. *Formal Methods in Computing*, 2001. (page 162)

[235] Natalie A Cookson, William H Mather, Tal Danino, Octavio Mondragón-Palomino, Ruth J Williams, Lev S Tsimring, and Jeff Hasty. Queueing up for enzymatic processing: correlated signaling through coupled degradation. *Molecular systems biology*, 7(1), 2011. (page 162)

[236] Arnon Arazi, Eshel Ben-Jacob, and Uri Yechiali. Bridging genetic networks

and queueing theory. *Physica A: Statistical Mechanics and its Applications*, 332:585–616, 2004. (page 162)

[237] Joey Bernard. Use python for scientific computing. *Linux Journal*, (175), 2008. (page 163)

[238] David M. Beazley. *Python Essential Reference.* Addison-Wesley Professional, 4 edition, 2009. (page 163)

[239] V.E. Malishkin and V.D. Korneev. *Parallel programming of multicomputers.* Novosibirsk Technical University, Novosibirsk, 2006. (page 164, 175)

[240] Norm Matloff. *Programming on Parallel Machines: GPU, Multicore, Clusters and More.* University of California, 2012. (page 164)

[241] K.J. Bogacev. *Basics of parallel programming.* Binom, Moscow, 2003. (page 175)

[242] U. Narayan Bhat. *An Introduction to Queueing Theory Modeling and Analysis in applications.* Birkhauser, Boston, 2008. (page 177)

[243] Renata Burbaitė. *Išplėstiniai generatyviniai mokymosi objektai informatikos mokymuisi: koncepcija, modeliai ir realizacija.* PhD thesis, Kaunas University of Technology, 2014. (page 190)

[244] T Kilby. Learning objects. *Web Based Training Information Center*, 2002. URL `http://wbtic.com/trends_objects.aspx`. (page 191)

[245] Giselher HJ Redeker. An educational taxonomy for learning objects. In *Advanced Learning Technologies, 2003. Proceedings. The 3rd IEEE International Conference on*, pages 250–251. IEEE, 2003. (page 191)

[246] Janet Finlay. Context-neutral e-learning objects: a tale of two projects. *The 7th HCI Educators Workshop: Effective Teaching and Training in HCI, 1st April 2004*, 2004. (page 191)

[247] Learning Technology Standards Committee et al. Ieee standard for learning object metadata. *IEEE Standard*, 1484(1):2007–04, 2002. (page 191)

[248] Bruno Defude and Ramzi Farhat. A framework to design quality-based learning objects. In *Advanced Learning Technologies, 2005. ICALT 2005. Fifth IEEE International Conference on*, pages 23–27. IEEE, 2005. (page 191)

Index

Vladimiras Dolgopolovas

SOFTWARE LEARNING OBJECTS FOR SCIENTIFIC COMPUTING EDUCATION: TEACHING SCIENTIFIC INQUIRY WITH RECURRENCE BASED STOCHASTIC MODELS

Doctoral dissertation

Technological Sciences
Informatics Engineering (07 T)

Editor Zuzana Šiušaitė