Научно-методическая статья
УДК 378.147: 004.584+004.588+4.853
DOI: 10.24412/2072-9014-2026-276-7-18

КОНЦЕПЦИЯ ОБУЧЕНИЯ БУДУЩИХ ВЕБ-ИНЖЕНЕРОВ АСИНХРОННОМУ ПРОГРАММИРОВАНИЮ С ИСПОЛЬЗОВАНИЕМ ТРЕНАЖЕРА-СИМУЛЯТОРА

Илья Борисович Государев

Национальный исследовательский университет ИТМО,
Санкт-Петербург, Россия
goss@itmo.ru, https://orcid.org/0000-0003-4236-5991

Аннотация. В статье предложена и теоретически обоснована концепция обучения будущих веб-инженеров асинхронному программированию на основе использования тренажера-симулятора. Обучение асинхронному программированию базируется на сочетании терминологической проработки ключевых понятий (таких как асинхронность) и моделировании с применением сетей Петри. Базовым в концепции обучения является тренажер-симулятор, позволяющий визуализировать динамику асинхронных взаимодействий. Концепция обучения интегрирует проектную деятельность, исследовательские задания, направленные на формирование у студентов устойчивых ментальных моделей исполнения программного кода, актуальных для решения профессиональных задач в области веб-инженерии.

***Ключевые слова*:** веб-инженерия; асинхронное программирование; JavaScript; сети Петри; визуализация; моделирование; тренажер-симулятор.

Для цитирования: Государев И. Б. Концепция обучения будущих веб-инженеров асинхронному программированию с использованием тренажера-симулятора / И. Б. Государев // Вестник МГПУ. Серия «Информатика и информатизация образования». 2026. № 2 (76). С. 7–18. https://doi.org/10.24412/2072-9014-2026-276-7-18

Original article
UDC 378.147: 004.584+004.588+4.853
DOI: 10.24412/2072-9014-2026-276-7-18

CONCEPT FOR TEACHING FUTURE WEB ENGINEERS ASYNCHRONOUS PROGRAMMING USING A SIMULATOR

Ilya B. Gosudarev

ITMO University,
Saint-Petersburg, Russia
goss@itmo.ru, https://orcid.org/0000-0003-4236-5991

Abstract. This article proposes and theoretically substantiates a concept for teaching asynchronous programming to future web engineers using a simulator. Teaching asynchronous programming is based on a combination of terminological elaboration of key concepts (such as asynchrony) and modeling using Petri nets. The simulator, which allows visualization of the dynamics of asynchronous interactions, is at the core of the training concept. The training concept integrates project activities and research assignments aimed at developing students' stable mental models of program code execution, relevant for solving professional problems in the field of web engineering.

Keywords: web engineering; asynchronous programming; JavaScript; Petri nets; visualization; simulationbased learning; projectbased learning.

For citation: Gosudarev I. B. Concept for teaching future web engineers asynchronous programming using a simulator? // MCU Journal of Informatics and Informatization of Education. 2026. № 2 (76). P. 7–18. https://doi.org/10.24412/2072-9014-2026-276-7-18

Введение

Рост глобального рынка веб-разработки в условиях цифровизации социальных процессов и увеличения востребованности разработчиков на веб-языках (таких как JavaScript) определяет новые вызовы для системы образования. В частности, инициируют совершенствование дистанционного и смешанного обучения с использованием веб-платформы, где особое место отводится эмергентному обучению, по мнению С. Г. Григорьева и О. В. Андрюшковой [1; 2].

Длительное время при обучении программированию доминировали понятия и подходы классических языков программирования (в частности, императивной парадигмы), в которых закреплено последовательное выполнение (или, напротив, акцентируется истинный параллелизм). На этом фоне веб-платформа за последнее десятилетие приобрела, как показано нами в [3], устойчивые черты одного из важнейших трендов образовательного процесса. При этом языки, определяющие веб-платформу (JavaScript и др.), реализуют специфичные подходы, не укладывающиеся в традиционные противопоставления парадигм: например,

в событийно ориентированном программировании центральным понятием выступает асинхронность.

Изучение аспектов понятия асинхронности мотивировало к обращению за поиском его смыслового и прагматического значения к различным предметным областям. Так, например, в специальной теории относительности с точки зрения современной физики не существует наблюдательно определимой абсолютной одновременности удаленных событий, что имеет определенный смысл для асинхронности. Если понимать под синхронностью одновременность или сводящуюся к ней согласованность, то события или процессы могут считаться синхронными лишь условно, в некоторых ограниченных рамках. Асинхронность как противоположная характеристика оказывается в таком рассмотрении фундаментальным свойством сред, состоящих из удаленных сущностей. Веб-платформы и используемые для разработки на них языки программирования отражают эту структуру распределенной реальности.

На философском уровне темпоральная природа веб-платформы может быть концептуально описана через идею диахронии, обсуждаемую Э. Левина [4] в полемике с различными философскими концепциями времени. На техническом уровне эта диахрония проявляется как необходимость описывать взаимодействия компонентов без предположения о едином такте и без возможности естественного глобального упорядочивания событий. В классической работе Л. Лэмпорта [5] вводится такая характеристика, как выражение «вне синхронности» (out of synchrony), для описания событий в распределенных системах, в которых отсутствует единое системное время, а также отношение «произошло раньше» (happened before) как частичный порядок событий и раскрывается идея конкурентности как отношений между событиями, не связанными этим частичным порядком. Данная трактовка принципиальна для обучения программированию, поскольку многие ошибки в асинхронном программировании возникают как следствие неявной подмены частичного порядка линейным временем.

Веб-платформа является диахронной распределенной системой, компоненты которой взаимодействуют посредством передачи сообщений и обработки ответов на запросы, которые С. Парастатидис с соавторами [6] называет асинхронными. На уровне инженерной реализации ключевым механизмом асинхронности в JavaScript-средах выступает так называемый цикл событий (алгоритм event loop) и система очередей задач. В среде Node.js цикл событий обеспечивает неблокирующий ввод/вывод, разгружая операции в сторону ядра ОС и возвращая управление JavaScript-коду через функции обратного вызова (известные также как *коллбэки*). Программирование приложений осуществляется на основе парадигмы событий: часть кода исполняется при любых условиях, а другая часть — альтернативно (что можно интерпретировать как своего рода распределенное недетерминированное ветвление).

Обобщая результаты в области теории распределенных систем, Р. Шуп, А. Стрельцофт [7], П. Шик и У. Гольц [8] формально исследуют, в каких случаях синхронное взаимодействие может быть реализовано в асинхронной

среде без изменения поведения системы, привлекая формализм сетей Петри. До введения в эксплуатацию веб-платформы ученые предлагали структуры данных, позволяющие инкапсулировать идею ожидания ответа на запрос или реакции на сообщения. Одна из таких структур (Promises, или обещания, которые в специальной литературе упоминаются также как *промисы*) описана коллективом под руководством Б. Лисков [9].

Если говорить о психолого-педагогическом аспекте, то асинхронность и конкурентность относятся к числу тем, которые обучающиеся нередко воспринимают как концептуально трудные, поскольку модель конкурентного исполнения оставляет больше неопределенное поведение по сравнению с последовательным, что мешает формированию корректной ментальной модели. Эта проблематика также нашла отражение в исследованиях как за рубежом, так и в России. В одной из первых зарубежных работ по данной теме, предложенной Л. Рурк [10], асинхронность рассматривается как характеристика или форма коммуникации, которая устраняет барьеры одновременности, расширяя категорию социального присутствия в контексте опосредованных сетевых взаимодействий.

Таким образом, в философской и методологической литературе накоплена теоретическая база для постановки вопроса о методике обучения асинхронности как на макроуровне, так и на более частном уровне, который относится к конкретным языкам веб-программирования.

Методы исследования

Методология исследования, направленного на разработку концепции обучения асинхронному программированию будущих веб-инженеров для ее использования в профессиональной подготовке данных специалистов на уровне высшего образования с целью обеспечения решения ключевых задач цифровой трансформации образования с использованием веб-платформы в условиях формирования технологического суверенитета, базируется на:

а) положениях национальной программы «Цифровая экономика Российской Федерации», а также направлениях развития Российской Федерации на период до 2030 года и на перспективу до 2036 года, обозначенных в Указе Президента РФ от 7 мая 2024 г. № 309;

б) трактовке методологической основы концепции обучения как совокупности взаимодополняющих востребованных в российском образовании подходов, обеспечивающих целостность учебного процесса в вузе (компетентностного, системно-деятельностного, проектного, а также подходов, связанных с обучением через моделирование и визуализацию);

в) трактовке концепции обучения как совокупности теоретико-методологических положений, основных идей обучения, раскрывающих в нашем случае сущность обучения асинхронному программированию посредством описания целей, содержания, принципов, методов, форм и средств обучения.

Результаты исследования

Основную идею концепции обучения асинхронному программированию составляет подход «обучение через моделирование», реализуемый посредством использования фундаментальных понятий программирования: сети Петри и моделирование. Сети Петри исторически применяются как средство описания и анализа информационных систем, где важны конкурентность, частичный порядок событий и асинхронность. В процессе исследования этой проблематики были изучены существующие научно-методические публикации и специальная литература по информатике и программированию, на которые в дальнейшем опирались при отборе образовательного контента, методов, форм и средств обучения асинхронному программированию. Остановимся на кратком обзоре и характеристике некоторых из них.

В отечественной научной и научно-методической литературе к проблематике обучения параллельному и конкурентному программированию неоднократно обращались А. Г. Марчук, Л. В. Городняя, А. Ф. Дедков, М. В. Швецкий и др. В работах советского периода раскрывались перспективы параллельного, распределенного, конкурентного и асинхронного программирования, а в более современных трудах описывается развивающаяся парадигма параллельного программирования и подчеркивается сложность обучения программированию в рамках этой парадигмы. В работах указанных авторов (например, [11]) вводится в числе прочих понятие «псевдопараллельное программирование», призванное акцентировать внимание на том, как выполнение процессов воспринимается пользователями таких систем, как веб-платформа. Заметим, что при реализации этих процессов на таких веб-языках, как JavaScript, для моделирования привлекаются также традиционные формализмы (конечные автоматы для описания состояний, лямбда-исчисление для описания взаимодействий между функциями, которые, как правило, представляют собой основной строительный блок веб-приложения). Некоторые из этих аспектов в той или иной мере были рассмотрены, например, в работе С. Д. Каракозова и М. В. Худжиной [12] в контексте профессиональной подготовки IT-специалистов в области веб-программирования.

Таким образом, при разработке веб-приложений широко используются методы и параллельного, и псевдопараллельного программирования. Однако анализ употребления этих терминов в научно-методической и учебно-методической литературе выявляет их неоднозначность и актуальность методического переосмысления в реалиях веб-программирования.

Понятие асинхронности широко распространилось в предметной области веб-инженерии в связи с рядом обстоятельств, главным образом связанных с внедрением фоновых запросов к интернет-ресурсам (AJAX) и дальнейшей реализацией операций с внешними ресурсами на платформе Node.js. Акцент в этих ситуациях делался на таком качестве исполнения кода, как блокирование. Поэтому с методической точки зрения целесообразно опираться на концепцию блокирования. Для ее рассмотрения достаточно ситуации вызова функции,

исполнение которой требует действий, внешних по отношению к коду, который осуществляет вызов. Блокирование вызова имеет место, если внешний по отношению к исполнителю ресурс удерживает вызывающего в ожидании любого исхода запроса, причем в это время тот не может продолжать полезную работу. Асинхронность рассматривается как темпоральная «развязка» начала запроса, то есть его инициализирующей части, и завершения или последующей обработки результата, которая исключает проблему блокирования.

Практическая реализация основных идей концепции обучения будущих веб-инженеров асинхронному программированию, организованного нами в рамках программы магистратуры «Веб-технологии» (Университет ИТМО, Санкт-Петербург), опирается на классическую модель методики обучения (МСО), «представляющую собой взаимосвязанную совокупность пяти базовых компонентов образовательного процесса — целей (внешних и внутренних), содержания (предметные линии, учебные элементы), методов, форм и средств обучения» [13, с. 193]. Для конкретизации каждого из этих компонентов и учитывая современные тенденции дидактики, предлагаем следующее описание.

Целевой компонент определяется социальным заказом веб-индустрии и спецификой веб-платформы: выпускник должен уметь проектировать неблокирующие сценарии взаимодействия с внешними ресурсами, правильно использовать композиции промисов и async/await, понимать приоритеты микрозадач, оценивать риски зависания цикла событий и корректно организовывать конкурентные процессы в рамках однопоточного исполнения JavaScript.

Содержательный компонент методики опирается на две взаимосвязанные предметные линии. *Первая линия* — собственно средства языка и платформы: промисы, async/await, таймеры, событийная модель, I/O-операции Node.js, а также устройство цикла событий и очередей. Здесь содержание опирается на документационную базу: например, различение очередей задач и микрозадач в HTML-стандарте и правила обработки микрозадач. *Вторая линия* — формальные модели и визуализация: введение сетей Петри и их применение для моделирования очередей, событий, переходов между состояниями и зависимостей. Сеть Петри используется не как лишь наглядное изображение (аналог UML-диаграммы), а как исполняемая модель, где динамика выражается перемещением токенов и срабатыванием переходов. Основное содержание изложено в формате серии *видеоуроков* курса «Асинхронное программирование в экосистемах веб-инженерии», доступных в рамках LMS и выполняемых последовательно (рис. 1).

Отметим, что данный курс, в основе которого лежит предложенная концепция, является составляющей магистерской программы «Веб-технологии», действующей в университете ИТМО[1]; структура содержания данной программы выполнена в модульной структуре, а последовательность изучения курсов определяется обходом графа, представляющего логическую структуру содержания обучения в рамках программы [14, с. 108]. Учет отзывов и пожеланий выпускников

[1] См. перечень курсов на сайте: https://abit.itmo.ru/program/master/web_tech#passport

Рис. 1. Темы (названия видеоуроков) и учебные элементы содержания обучения асинхронному программированию веб-инженеров (визуализация выполнена автором)

и работодателей позволил определить базу для обновления учебного плана и уточнения перечня компетенций. В частности, были конкретизированы компетенции в области асинхронности (табл. 1).

Таблица 1

Составляющие компетенций, формируемые посредством освоения содержания обучения асинхронному программированию

Компетенция (способность)	Понятия содержания обучения асинхронному программированию — учебные элементы как составляющие компетенции	Кол-во учебных часов
Распознавать правильность синтаксиса, устанавливать смысловое значение и прагматику терминов асинхронного программирования	Понимание асинхронности, различение синхронного и асинхронного выполнения, базовые представления о цикле событий, задачах, очередях, событиях, промисах и await как элементах единой концептуальной системы	4
Использовать базовые механизмы асинхронного взаимодействия в JavaScript	Коллбэки, слушатели событий, подписка и отписка, порядок вызовов	4
Применять подход на основе промисов и синтаксис async/await для разработки асинхронного кода	Состояния промиса, then/catch/finally, комбинаторы, await, try/catch, последовательное и параллельное выполнение асинхронных операций	4
Проектировать и реализовывать асинхронные сценарии в веб-приложениях	Выбор адекватного механизма под задачу, организация пользовательских событий, запросов к серверу, обработки результатов, ошибок, состояний загрузки и координации нескольких операций	4
Анализировать, отлаживать и оценивать корректность асинхронного кода	Поиск ошибок, связанных с порядком выполнения, await, необработанными ошибками, лишними подписками, логикой параллелизма и взаимодействием компонентов	6

Обратим внимание, что для формирования указанных выше профессиональных компетенций у будущих специалистов в области веб-инженерии и освоения ими составляющих содержательного компонента методики обучения деятельность студентов (рис. 2) сводится к выполнению мини-сценариев. Эта деятельность реализуется в учебном процессе посредством использования активных методов обучения, в частности метода проектов и базового дидактического средства — тренажера-симулятора, которые являются составляющими процессуального компонента методики обучения асинхронному программированию.

Рис. 2. Базовый сценарий учебной деятельности студентов по освоению содержания обучения асинхронному программированию (визуализация выполнена автором)

Процессуальный компонент — это деятельностная основа методики обучения, формируемая использованием в учебном процессе совокупности традиционных и инновационных методов, форм и средств обучения. Наиболее эффективными методами при обучении асинхронному программированию являются метод демонстрационных примеров, метод целесообразно подобранных задач (примеров, кейсов) и обучения через задачи, а также метод моделирования [15, с. 96].

В числе используемых традиционных цифровых инноваций и решений в качестве средств обучения, на наш взгляд, актуальное значение имеют ИИ-агенты в составе сред разработки, а также тренажер-симулятор «Сети Петри для асинхронных операций», разрабатываемый студентами-магистрантами в формате группового проекта при изучении «обещаний» (промисов) и структур псевдосинхронного кода (стартовая страница тренажера-симулятора представлена на рис. 3).

Тренажер-симулятор, одна из основ концепции обучения, представляет собой изолированную оболочку, использующую виртуальную машину JavaScript для выполнения или имитации выполнения асинхронных функций. Тренажер-симулятор реализует в учебном процессе следующие дидактические функции.

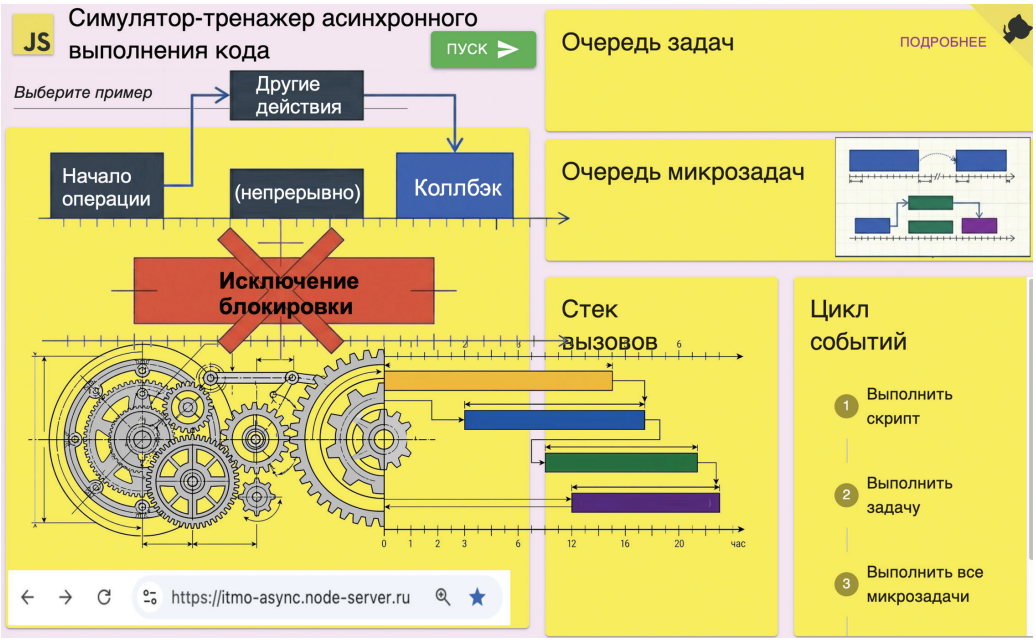

Рис. 3. Тренажер-симулятор цикла событий для визуализации исполнения кода на языке JavaScript (см.: https://itmo-async.node-server.ru/)

1. Обеспечивает визуализацию очередей и отношения причинности. Например, узлы (места) сети интерпретируются как состояния и очереди, переходы — как события постановки / извлечения коллбэков, токены — как единицы работы (коллбэки, фрагменты с передачей продолжения выполнения).

2. Поддерживает режим управляемой симуляции (по шагам) и режим сценарной симуляции (автоисполнение кода), что позволяет создавать исследовательские задания: студент формулирует гипотезу о порядке событий, затем проверяет ее на модели, после чего сопоставляет результат с поведением реального кода в Node.js / браузере.

3. Выступает как средство для перехода от «интуитивных» объяснений к формальным: вводится понятие разрешенности перехода, конфликтов, конкуренции, ограниченности очередей, достижимости и др.

Оценочно-результативный компонент включает систему критериев и заданий, отражающих уровни сформированности компетенций в области асинхронности. Предлагается использовать *комбинацию задач*:

1) диагностических (на предсказание порядка исполнения и объяснение причин);

2) на отладку асинхронных дефектов;

3) на оценивание проектных артефактов.

В качестве примера приведем формулировку задачи на предсказание состояния промиса: «*Если создать экземпляр промиса, передав конструктору функцию-исполнитель* () => {}, *в каком состоянии он будет находиться*?». На этот вопрос в рамках тестового задания можно дать однозначный ответ

в виде одного слова (а именно pending) или проследить жизненный цикл и раскрыть это состояние как результат последовательности шагов, через который проходит промис (согласно спецификации ECMA-262). Во втором случае визуализацию указанной последовательности можно наглядно представить с помощью тренажера-симулятора. Проверка правильности выполнения осуществляется путем обращения по адресу https://node-server.ru/r?id=student1#Ivanov|67 и набора решения в соответствующем поле веб-редактора.

Заключение

Теоретическая значимость описанной концепции состоит в том, что обучение асинхронному программированию предложено трактовать как обучение особому типу рассуждений, где акцентируется внимание на такой математической модели, как частичный порядок событий, а также на правилах планирования исполнения программ и различении синхронного кода и асинхронного исполнения. Практическая значимость заключается в разработке инструментария для эффективного обучения асинхронному программированию посредством тренажера-симулятора, реализуемого в условиях магистерской подготовки будущих веб-инженеров на базе университета ИТМО (Петербург). Таким образом, тренажер-симулятор на основе сетей Петри позволяет реализовать обучение асинхронности в режиме управляемого эксперимента с исполняемой моделью, наблюдением динамики выполнения кода и проверкой гипотез о порядке событий.

Список источников

1. *Андрюшкова О. В.* Эмергентное обучение в информационно-образовательной среде / О. В. Андрюшкова, С. Г. Григорьев. М.: Образование и Информатика, 2018. 103 с.

2. *Григорьев С. Г.* Генезис инженерной мысли: учеб. пособие / С. Г. Григорьев. М.: МГПУ, 2022. 95 с.

3. *Государев И. Б.* Веб-платформа как современный тренд развития образовательного процесса // Человек и образование. 2024. № 1 (78). С. 149–156.

4. *Levinas E.* Le temps et l'autre. Paris: Presses universitaires de France, 1979. 91 p.

5. *Lamport L.* Time, Clocks, and the Ordering of Events in a Distributed System // Communications of the ACM. 1978. Vol. 21. No. 7. P. 558–565.

6. Asynchronous Messaging between Web Services Using SSDL / S. Parastatidis [et al.] // IEEE Internet Computing. 2006. Vol. 10. No. 1. P. 26–39.

7. *Schoop R.* Asynchronous and synchronous approaches for programming distributed control systems based on standards / R. Schoop, A. Strelzoft // Control Engineering Practice. 1996. Vol. 4. No. 6. P. 855–861.

8. *Schicke J.-W.* Synchrony vs. Causality in Asynchronous Petri Nets / J.-W. Schicke, K. Peters, U. Goltz / arXiv:1108.4471. URL: https://arxiv.org/abs/1108.4471 (дата обращения: 24.04.2026).

9. *Liskov B.* Promises: linguistic support for efficient asynchronous procedure calls in distributed systems / B. Liskov, L. Shrira // Proceedings of the ACM SIGPLAN 1988 conference on Programming language design and implementation (PLDI ‘88). N. Y.: ACM, 1988. P. 260–267.

10. Assessing social presence in asynchronous Text-based computer conferencing / L. Rourke [et al.] // Journal of Distance Education. 1999. Vol. 14. No. 2. P. 50–71.

11. *Шилов Н. В.* Параллельное программирование среди других парадигм программирования / Н. В. Шилов, Л. В. Городняя, А. Г. Марчук // Прикладная информатика. 2011. № 1 (31). С. 120–129.

12. *Каракозов С. Д.* Обучение webпрограммированию бакалавров IT-направлений в региональных вузах: актуальность, проблемы и подходы / С. Д. Каракозов, М. В. Худжина // Проблемы современного образования. 2021. № 5. С. 182–195.

13. Использование цифрового образовательного контента в школе: модель методики и принципы / Н. И. Рыжова [и др.] // Вестник Томского государственного университета. 2025. № 511. С. 191–199.

14. *Государев И. Б.* Модель содержания профессиональной подготовки IT-специалиста в области веб-инженерии / И. Б. Государев // Сибирский педагогический журнал. 2026. № 2. С. 108–122.

15. *Лаптев В. В.* Специальные методы обучения информатике / В. В Лаптев, Н. И. Рыжова, М. В. Швецкий // Вопросы теории и практики обучения информатике: сб. науч. тр. Вып. 3. СПб.: РГПУ им. А. И. Герцена, 1998. С. 95–113.

References

1. *Grigoriev S. G.* Emergent learning in the information-educational environment / S. G. Grigoriev, O. V. Andryushkova. M.: Education and Informatics. 103 p.

2. *Grigoriev S. G.* The genesis of the engineering thought. M.: MCU, 2022. 95 p.

3. *Gosudarev I. B.* Web platform as a new trend of the educational process evolution // Man and Education. 2024. No. 1 (78). P. 149–156.

4. *Levinas E.* Le temps et l’autre. Paris: Presses universitaires de France, 1979. 91 p.

5. *Lamport L.* Time, Clocks, and the Ordering of Events in a Distributed System // Communications of the ACM. 1978. Vol. 21. No. 7. P. 558–565.

6. Asynchronous Messaging between Web Services Using SSDL / S. Parastatidis [et al.] // IEEE Internet Computing. 2006. Vol. 10. No. 1. P. 26–39.

7. *Schoop R.* Asynchronous and synchronous approaches for programming distributed control systems based on standards / R. Schoop, A. Strelzoft // Control Engineering Practice. 1996. Vol. 4. No. 6. P. 855–861.

8. *Schicke J.-W.* Synchrony vs. Causality in Asynchronous Petri Nets / J.-W. Schicke, K. Peters, U. Goltz / arXiv:1108.4471. URL: https://arxiv.org/abs/1108.4471 (accessed: 24.04.2026).

9. *Liskov B.* Promises: linguistic support for efficient asynchronous procedure calls in distributed systems / B. Liskov, L. Shrira // Proceedings of the ACM SIGPLAN 1988 conference on Programming language design and implementation (PLDI ‘88). N. Y.: ACM, 1988. P. 260–267.

10. Assessing social presence in asynchronous Text-based computer conferencing / L. Rourke [et al.] // Journal of Distance Education. 1999. Vol. 14. No. 2. P. 50–71.

11. *Marchuk A. G.* Parallel Programming as Programming Paradigm / A. G. Marchuk, L. V. Gorodnyaya, N. V. Shilov // Journal of Applied Informatics, No. 1 (31). P. 120–129.

12. *Karakozov S. D.* Teaching web development to IT bachelors in regional universities: relevance, challenges and approaches / S. D. Karakozov, M. V. Khudzhina // Problems of Modern Education. No. 5. P. 182–195.

13. Digital educational content in school: Teaching methods model and usage principles / N. I. Ryzhova [et al.] // Vestnik Tomskogo gosudarstvennogo universiteta // Tomsk State University Journal. 2025. No. 511. P. 191–199.

14. *Gosudarev I. B.* Model of the content of professional training for an IT specialist in the field of web engineering / I. B. Gosudarev // Siberian Pedagogical Journal. No. 2. P. 108–122.

15. *Laptev V. V.* Special methods for teaching computer science / V. V. Laptev, N. I. Ryzhova, M. V. Shvetsky // Questions of Theory and Practice of Teaching Computer Science: Collection of Scientific Works. Is. 3. SPb.: Herzen University, 1998. P. 95–113.

Статья поступила в редакцию: 02.04.2026;
одобрена после рецензирования: 25.04.2026;
принята к публикации: 25.04.2026.

The article was submitted: 02.04.2026;
approved after reviewing: 25.04.2026;
accepted for publication: 25.04.2026.

Информация об авторе / Information about the author

Илья Борисович Государев — кандидат педагогических наук, доцент факультета программной инженерии и компьютерной техники, Национальный исследовательский университет ИТМО, Санкт-Петербург, Россия.

Ilya B. Gosudarev — Candidate of Pedagogical Sciences, Associate Professor, Faculty of Software Engineering and Computer Technology, ITMO University, Saint-Petersburg, Russia.

goss@itmo.ru, https://orcid.org/0000-0003-4236-5991