Автономная некоммерческая организация образовательная организация дополнительного профессионального образования «Учебный Центр Информационные Бизнес Системы»
(АНО ДПО «УЦ ИБС»)

Утверждаю
Директор АНО ДПО «УЦ ИБС»
Гернер В.А.
_____
(подпись)
«09» января 2024 г.

ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА

«Архитектор ПО. Путь к мастерству в проектировании систем»

ДЛЯ ОБУЧЕНИЯ В РАМКАХ ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
(ПОВЫШЕНИЕ КВАЛИФИКАЦИИ)

Москва 2024 г.

1. ОБЩИЕ ПОЛОЖЕНИЯ

Дополнительная профессиональная программа повышения квалификации «Архитектор ПО. Путь к мастерству в проектировании систем» разработана на основе следующих нормативных документов:

- Федеральный закон от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Министерства образования и науки Российской Федерации от 1 июля 2013 г. № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам»;
- Профессиональный стандарт «Архитектор программного обеспечения» №67 утвержден приказом Министерства труда и социальной защиты Российской Федерации от 30.08.2021 № 579н.

2. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

1.1 Целью преподавания дисциплины является

Повышение квалификации по образовательной программе «Архитектор ПО. Путь к мастерству в проектировании систем», в т.ч. формирование у слушателей набор знаний и навыков для проектирования архитектуры программного обеспечения.

1.2 Задачи изучения дисциплины

- Научить планировать будущее развитие сложных систем с нуля: выбирать инструменты и средства моделирования архитектуры, формат документирования архитектурных решений, выбирать архитектурный стиль под конкретную бизнес-задачу, планировать будущее масштабирование, гибкость.
- Ознакомить с лучшими практиками построения архитектуры программного обеспечения.

1.3 Связь дисциплины с другими учебными дисциплинами

Изучение дисциплины предполагает наличие у слушателей специальных знаний и умений использования UML, владение основами объектно-ориентированного проектирования. Желательно знать язык Java, основы архитектуры компьютеров, архитектуры сетей передачи данных и GOF паттернов. Слушатель должен знать и понимать принципы объектно-ориентированного программирования, как устроен процесс разработки ПО.

3. СОДЕРЖАНИЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Категория слушателей программы:

- middle- и senior- разработчики;
- системные и бизнес аналитики;
- начинающие архитекторы ПО, системные архитекторы;
- менеджеры с техническим бэкграундом.

К освоению учебной программы допускаются: лица, имеющие среднее профессиональное и (или) высшее образование; лица, получающие среднее профессиональное и (или) высшее образование.

Базовые компетенции, которыми должен владеть слушатель программы:

- Знать методы и инструменты анализа и проектирования, владеть интегрированными средами разработки, инструментарием управления проектом, системами контроля версий;
- Проводить оценку осуществимости требований, вырабатывать требования к программному обеспечению;
- Использовать методы и технологии верификации формальных спецификаций;
- Уметь писать код и оценивать соответствие программного кода архитектуре компьютерной системы;
- Осуществлять объектно-ориентированное проектирование, взаимодействовать с представителями заказчика или специалистами в предметной области.

4. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

В ходе изучения программы слушатель должен приобрести знания и навыки, необходимые для проектирования архитектуры программного обеспечения, для работы в проектах с различной предметной областью и технологической базой.

После завершения обучения слушатели смогут:

1. проектировать системы с использованием производительности приемов;
2. оптимизировать системы с повышенными требованиями к производительности;
3. проектировать интерфейс распределённого доступа к объектам;
4. работать с требованиями и определять иерархию требований;
5. использовать атрибуты требований и связи между требованиями для оценки трудоемкости их реализации или изменения.

5. МОДУЛЬНЫЙ ПЛАН ПРОГРАММЫ

Формат дистанционный индивидуальный

№	Наименование модулей	Количество часов	Форма контроля
1	ARC-I-001 Архитектурные стили программного обеспечения: формирование базовой архитектуры	4	Выходное тестирование по каждому модулю
2	ARC-I-002 Работа с требованиями при проектировании архитектуры	6	
3	ARC-I-003 Создание устойчивых решений: принципы проектирования	8	
4	ARC-I-004 Технологии интеграции и взаимодействия микросервисов	18	
5	ARC-I-005 Технологии хранения и управления информацией	18	
6	ARC-I-006 Системная архитектура: дизайн и оптимизация инфраструктуры	18	
7	ARC-I-007 Документирование архитектурных решений: BPMN, UML, нотация 4+1 и ADR	12	
8	ARC-I-008 Микросервисная архитектура: проектирование, внедрение, поддержка	20	
9	ARC-I-009 Инструменты командного игрока: Soft Skills	1	
	Итого	105	

Формат дистанционный групповой

№	Наименование модулей	Количество часов	Форма контроля
1	ARC-I-001 Архитектурные стили программного обеспечения: формирование базовой архитектуры	10	Выходное тестирование по каждому модулю. Практические задания по каждому модулю. Проверка практических заданий преподавателем.
2	ARC-I-002 Работа с требованиями при проектировании архитектуры	18	
3	ARC-I-003 Создание устойчивых решений: принципы проектирования	24	
4	ARC-I-004 Технологии интеграции и взаимодействия микросервисов	44	
5	ARC-I-005 Технологии хранения и управления информацией	44	
6	ARC-I-006 Системная архитектура: дизайн и оптимизация инфраструктуры	44	
7	ARC-I-007 Документирование архитектурных решений: BPMN, UML, нотация 4+1 и ADR	34	
8	ARC-I-008 Микросервисная архитектура: проектирование, внедрение, поддержка	48	
9	ARC-I-009 Инструменты командного игрока: Soft Skills	2	
	Итого	268	

6. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН МОДУЛЕЙ ПРОГРАММЫ

1. ARC-I-001 Архитектурные стили программного обеспечения: формирование базовой архитектуры

№	Тема	Форма контроля
1	Понятия архитектуры: основные термины и определения	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Понятия архитектуры: критерии качества архитектуры	
3	Роли и виды архитекторов	
4	Архитектурные стили: монолит	

5	Архитектурные стили: микросервисы	
6	Принципы предметно-ориентированного проектирования (DDD)	
7	Clean Architecture: правила создания архитектур	
8	Архитектурные стили: событийно-управляемая архитектура	
9	Архитектурные стили: классический ETL/DWH/BI	
10	Позиция архитектора в Agile проекте	

2. ARC-I-002 Работа с требованиями при проектировании архитектуры

№	Тема	Форма контроля
1	Управление заинтересованными сторонами (стейкхолдерами) проекта	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Виды требований и атрибуты качества	
3	Влияние нефункциональных требований на архитектуру	
4	Сбор и работа с функциональными требованиями	
5	Дополнительные виды требований: ограничения	
6	Управление критериями надежности (SLA, SLO, SLI)	
7	Как определить и улучшить критерии качества требований	
8	Работа с архитектурными изменениями	

3. ARC-I-003 Создание устойчивых решений: принципы проектирования

№	Тема	Форма контроля
1	Проектирование от атрибутов качества	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Тактики и паттерны проектирования	

4. ARC-I-004 Технологии интеграции и взаимодействие микросервисов

№	Тема	Форма контроля
1	Задача коммуникации приложений	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Уровни (методы) интеграционных решений	
3	Шаблоны коммуникации в контексте приложений и микросервисной архитектуры: применение и практические сценарии	
4	Синхронное взаимодействие и REST: основы, генерация API и управление версиями	
5	REST API: рекомендации по дизайну, блокировкам, кешированию и версионированию	
6	OpenAPI Spec: интеграция и API First подход в создании удобных и гибких интерфейсов	
7	Архитектура сообщений: ключевые паттерны и методы разработки	
8	AsyncAPI: организация асинхронного взаимодействия	

9	Технологии асинхронного взаимодействия: RabbitMQ, Kafka в сравнении	
10	Интеграция баз данных и ETL: сильные стороны, метрики и методы оптимизации	
11	Capture Data Changes: анализ Debezium и его использование в различных сценариях	
12	Интеграция через файлы: особенности, преимущества и правила наименования	
13	Использование вспомогательных протоколов и средств интеграции API	

5. ARC-I-005 Технологии хранения и управления информацией

№	Тема	Форма контроля
1	Реляционные и нереляционные базы данных: обзор, особенности и гарантии доступности	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Как выбрать базу данных на основании требований и контекста	
3	Организация конкурентного доступа: стратегии блокировок, изоляции транзакций и обеспечение согласованности; детали реализации транзакций и блокировок: обзор PostgreSQL, MySQL и их возможностей. CAP-теорема	
4	Современные тактики производительности реляционной базы данных	
5	Виды нереляционных баз данных: обзор, гарантии; отличия и детали реализации нереляционных баз данных	
6	Проектирование моделей данных	

6. ARC-I-006 Системная архитектура: дизайн и оптимизация инфраструктуры

№	Тема	Форма контроля
1	Технологии виртуализации	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	Основы построения информационных сетей	
3	Архитектура Linux	
4	Основы CI/CD: построение процессов с использованием Jenkins, GitLab, Ansible	
5	Мониторинг и наблюдаемость системы: паттерны, инструменты и протоколы	
6	Проектирование облачной архитектуры: провайдеры, возможности, экономика	
7	Проектирование облачной архитектуры: паттерны	
8	Разработка 12-Факторного приложения	
9	Контрактные обязательства сервиса: метрики надежности, SLA, RTO, RPO и паттерны реализации	
10	Понимание Kubernetes: типы сущностей, Helm, CI/CD и балансировка трафика	

7. ARC-I-007 Документирование архитектурных решений: BPMN, UML, нотация 4+1 и ADR

№	Тема	Форма контроля
1	UML для моделирования и анализа систем	• тестовые вопросы • задачи для самостоятельной практики • живая практика (лабораторные) с экспертом-преподавателем и кейс-проект (для группового формата)
2	BPMN для моделирования бизнес-процессов	
3	Нотация 4+1 для анализа и проектирования архитектуры	
4	Нотация C4 для наглядного и эффективного проектирования архитектуры	
5	Изучение подхода ADR для документации архитектурных решений	
6	Язык Archimate	

8. ARC-I-008 Микросервисная архитектура: проектирование, внедрение, поддержка

№	Тема	Форма контроля
1	Введение	• тестовые вопросы • задачи для самостоятельной практики
2	Стратегические паттерны Event Storming и DDD	
3	Проектирование микросервисов, тактические паттерны	
4	Инфраструктура и данные в микросервисах	

5	Жизненный цикл микросервисов	• живая практика (лабораторные) с экспертом-преподавателем • сквозной кейс-проект
6	Тестирование	
7	Observability и поддержка	
8	Распил монолита	

9. ARC-I-009 Инструменты командного игрока: Soft Skills

№	Тема	Форма контроля
1	Сдавать задачи в срок: принципы личной эффективности	• тестовые вопросы
2	Как развивать команду и выстраивать коммуникации	
3	Навыки презентации	

7. ФОРМА ОБУЧЕНИЯ

Формат дистанционный индивидуальный

Обучение проходит в системе дистанционного обучения, самостоятельно:

- Материалы курсов изучаются в системе дистанционного обучения (СДО). По окончании теоретической части каждого курса программы предусмотрен тест для проверки уровня знаний.

Формат дистанционный групповой

Обучение проходит в системе дистанционного обучения в смешанном формате:

- Материалы курсов изучаются в системе дистанционного обучения (СДО). По окончании теоретической части каждого курса программы предусмотрен тест для проверки уровня знаний.
- Практические групповые сессии с преподавателем проходят в режиме вебинара.

8. ФОРМА КОНТРОЛЯ

Контроль усвоения учебной программы проводится в различных формах:

1. Решение практических задач;
2. Промежуточный контроль (тестовый контроль);
3. Тестовый итоговый контроль.

Курс считается успешно пройденным при достижении следующих результатов:

1. В системе дистанционного обучения прогресс по изучению материалов не менее 80%;
2. Тестовый итоговый контроль сдан не менее чем на 80%.

При успешном прохождении итоговой аттестации или получении на итоговой аттестации удовлетворительных результатов, выдается Удостоверение о повышении квалификации установленного образца.
Лицам, не прошедшим итоговой аттестации или получившим на итоговой аттестации неудовлетворительные результаты выдаётся Справка об обучении по программам дополнительного профессионального образования в соответствии с Положением о порядке выдаче и форме данной справки.

Примеры тестовых вопросов:

1. Какие шаблоны/стили могут быть использованы внутри монолита?

- Layers
- Event Sourcing
- Enterprise Service Bus
- Active Record
- Service Discovery

2. Для каких бизнес-операций оптимально применить кэширование в архитектуре коммуникации REST?

- Запрос значений справочника валют
- Перевод баллов лояльности другому пользователю
- Создание учетной записи пользователя
- Запрос координат любимых мест пользователя

3. Какие атрибуты качества характерны для микросервисной архитектуры?

- Микросервисная архитектура более производительна, чем монолитные решения
- Микросервисная архитектура обладает высокой согласованностью данных между сервисами
- Микросервисная архитектура более проста в сопровождении, чем монолит
- Микросервисная архитектура обычно обеспечивает бОльшую масштабируемость и гибкость, чем архитектура на основе монолита

4. В какой последовательности выполняется алгоритм моделирования диаграммы действий:

- Формируем диаграмму состояний основного объекта процесса. Выделяем то, что можно автоматизировать Если много шагов, декомпозируем диаграмму на отдельные элементы. Рисуем Use-case диаграмму.
- Выделяем то, что можно автоматизировать Если много шагов, декомпозируем диаграмму на отдельные элементы Рисуем Use-case диаграмму. Формируем диаграмму состояний основного объекта процесса.
- Если много шагов, декомпозируем диаграмму на отдельные элементы Выделяем то, что можно автоматизировать Рисуем Use-case диаграмму. Формируем диаграмму состояний основного объекта процесса.

5. Какое значение RTO должно быть у критичного для компании приложения?

- Максимальное
- Среднее
- Минимальное

9. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ПРОГРАММЕ

1. Майкл Нейгард, Release it! Проектирование и дизайн ПО для тех, кому не все равно, 2006
2. Л. Басс, П. Клементс, Р. Кацман. Архитектура программного обеспечения на практике, 2-е издание, 2006
3. Michael Keeling, Design It!: From Programmer to Software Architect (2017)
4. Мартин Клеппман, Высоконагруженные приложения. Программирование, масштабирование, поддержка, 2017
5. Paul Clements, et al. Documenting Software Architectures. Views and Beyond. 2nd Edition, 2010
6. Марк Ричардс, Паттерны архитектуры программного обеспечения, 2-е издание
7. Paul Clements, et al. Evaluating Software Architectures Methods and Case Studies. 2001
8. Мартин Роберт,, 2023Чистая архитектура. Искусство разработки программного обеспечения, 2023
9. Соммервилл Иан. Инженерия программного обеспечения = Software Engineering. – 6-е изд. – М.: «Вильямс», 2002

Учебная программа разработана: Отделом разработки и методологии АНО ДПО «УЦ ИБС»