**CI/CD** — одна из практик DevOps, подразумевающая непрерывную интеграцию и доставку. Этот набор принципов предназначен для повышения удобства, частоты и надежности развертывания изменений программного обеспечения или продукта. CI/CD относится к agile-практикам и позволяет разработчикам уделять внимание реализации бизнес-требований, качеству кода и безопасности продукта.

**Цели CI/CD:**

* обеспечение последовательного и автоматизированного способа сборки, упаковки и тестирования продуктов или приложений;
* автоматизация развертывания в разных окружениях;
* сведение к минимуму ошибок и проблем.

**Для CI/CD существуют четыре руководящих принципа:**

* **Разделение ответственности**. Каждый из участников процесса делит ответственность за те или иные этапы жизненного цикла продукта. Проектируется бизнес-логистика, внедряются сквозные функции, проводятся приемочные тесты и организуется логистика кода.
* **Снижение рисков**. Каждая команда, участвующая в разработке продукта, стремится к снижению рисков — контролируется корректность бизнес-логистики, проверяется пользовательский опыт, улучшается хранение и обработка данных и прочее.
* **Сокращение цикла обратной связи**. Разработчик и клиент должны стремиться к увеличению скорости внесения изменений и согласования правок. Сборку и тестирование кода можно автоматизировать. А для ситуаций, когда требуется участие человека, можно минимизировать число информационных посредников.
* **Реализация среды**. У разработчиков должно быть общее рабочее пространство с основной и вспомогательными ветками для контроля версий и качества, приемлемости, отказоустойчивости и других критериев.

**Методология CI/CD подразумевает разделение процесса разработки на семь этапов:**

1. **Написание кода**. Разработчики пишут код своего модуля и проводят тестирование в ручном режиме. После этого результат работы соединяется в главной ветке с текущей версией проекта. После того, как в главной ветке публикуются все коды модулей, начинается второй этап.
2. **Сборка**. Выбранная система контроля версий инициирует автоматическую сборку и последующее тестирование проекта. Триггеры для активации сборки могут быть настроены самостоятельно. Для автоматизации сборки применяется Jenkins или другой инструмент.
3. **Ручное тестирование**. После проверки CI-системой работоспособности тестовой версии код передается для ручного исследования.
4. **Релиз**. После ручного тестирования в сборку вносятся исправления. Следом проходит релиз версии кода для клиентов.
5. **Развертывание**. На этом этапе текущая (рабочая) версия кода размещается на production-серверах разработчика. Клиент может взаимодействовать с программой и изучать ее функции.
6. **Поддержка и мониторинг**. Продукт начинает использоваться конечными пользователями. При этом разработчики продолжают его поддерживать и проводят анализ пользовательского опыта.
7. **Планирование**. Исходя из пользовательского опыта разрабатывается новый функционал и готовится план доработок. После этого разработчик начинает написание кода — и цикл замыкается.

**Инструменты для CI/CD**

В ПО-разработке могут применяться разные инструменты для автоматизации процессов тестирования и доставки кода конечным пользователям. Вот некоторые из них:

* **GitLab**. Среда дает возможность управления репозиториями проекта, документирования результатов тестов (доработок) и функциональности, а также отслеживания ошибок.
* **Docker**. Система автоматического развертывания проектов, поддерживающая контейнеризацию, которая дает возможность упаковки проекта со всем окружением и зависимостями.
* **Travis-CI**. Инструмент, способный подключаться к репозиториям в GitHub с наименьшими настройками. Travis-CI — облачный инструмент, поэтому его установка не потребуется.
* **Jenkins**. Востребованный DevOps-инструмент, позволяющий работать с разными плагинами для гибкой настройки процессов под конкретные задачи разрабатываемого продукта.
* **PHP Censor**. Сервер непрерывной интеграции, предназначенный для автоматизации сборки PHP-проектов. Работает с репозиториями GitLab, GitHub, Mercurial и прочими, а также с библиотеками тестирования Atoum, PHP Spec, Behat. Инструмент документирован, но требует самостоятельной настройки и хостинг.
* **SberCloud K8S (SberCloud Managed Kubernetes).** Сервис автоматического развертывания, масштабирования и управления софтом на основе Kubernetes. Позволяет создавать кластеры последних версий, управлять автообновлением, автоматически распределять нагрузку, отслеживать статистику потребления и т.д.