НОРНИКЕЛЬ

Как low-code и no-code влияют на сроки и стоимость проекта

Максим Тарасов
Руководитель практики систем управления бизнес-процессами

СОДЕРЖАНИЕ

- **Pro-Code / High-Code** – классическое написание кода на обычных или специализированных языках программирования (JS, C#, Java, Python и др.) *
- **Low-Code** – это методика создания информационных систем, сервисов и программного обеспечения, с минимальным использованием рукописного кода. *
- **No-Code / Zero-Code** – производная от Low Code концепция, предполагающая полное отсутствие рукописного кода в программном обеспечении. *

* - из открытых источников

НОРНИКЕЛЬ

Структура затрат ИТ-проекта

Затраты проекта*

- Наличие методик расчета вычислительных мощностей и количества лицензий программного обеспечения у вендоров.
- Замена суммы статей «Мощности» и «Лицензии» общей стоимостью подписки для «облачных» решений.
- Работы могут быть классифицированы различным образом:
  - консалтинг, разработка, поддержка и др.
  - внешний подряд и внутренние ресурсы.

На первый взгляд, кажется, что согласно определениям методик low-code и no-code, их применение может существенно снизить трудозатраты на внедрение решений, а как следствие, и стоимость реализации проекта.

* Размеры секторов диаграммы приведены абстрактно

НОРНИКЕЛЬ

Стереотипы и факты Low/No-code (1\4)

Стереотипы Low / No-code			**Факты** Low/No-code
• Low-code платформа поможет закрыть любую потребность • Легкий путь реализации любого проекта	01	**Универсальность концепции:**	01	• Зависит от масштаба и сложности проекта • Работает в связке с Pro-code
• Гарантирует высокую скорость внесения изменений в ходе проекта\эксплуатации	02	**Стоимость разработки**	02	• Зависит от архитектуры решения и количества кастомного кода
• Большую часть задач закрывают бизнес-аналитики • Разработка может быть выполнена сотрудниками без навыков программирования • Low-code/No-code платформы практически не требует обучения	03	**Сотрудники**	03	• В Low-Code командах в основном программисты • Аналитики редко непосредственно участвуют в разработке • Требуются базовые знания ИТ по ряду направлений, в т.ч. алгоритмизации и программировании

Стереотипы и факты Low/No-code (2\4)

Универсальность концепции

No-code

- Разработка прототипа
- Разработка MVP (Minimum Viable Product)
- Визуальный интерфейс

Low-code

- Расширение логики работы No-Code инструментов скриптами

Pro-code

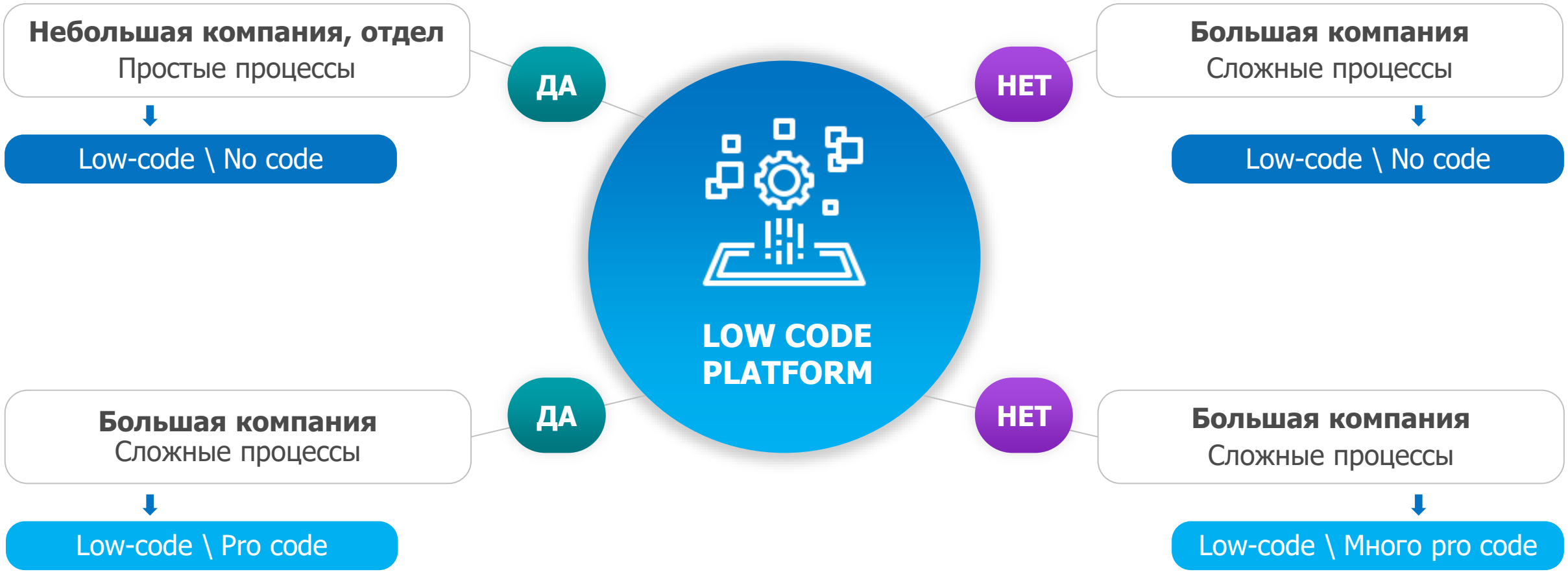
- Реализация сложной логики поведения системы
- Интеграционные коннекторы, не заложенные в базовом пакете

Использование готовых партнерских решений вендоров на Low-Code платформах: CRM, ITSM и т.д. (слайды 7-8)

НОРНИКЕЛЬ

Стереотипы и факты Low/No-code (3\4)

Скорость разработки



НОРНИКЕЛЬ

Стереотипы и факты Low/No-code (4\4)

Сотрудники. Быстрота освоения

	No-code	Low-code	Pro-code
Определение методики	концепция, предполагающая полное отсутствие рукописного кода в программном обеспечении.	методика создания информационных систем, сервисов и программного обеспечения, с минимальным использованием рукописного кода	классическое написание кода на обычных или специализированных языках программирования (JS, C#, Java, Python и др.)
Уровень требований методики к сотрудникам	Базовые знания ИТ	Начинающий разработчик (Junior)	Разработчик среднего уровня (Middle), старший разработчик (Senior)
Время на освоение	Недели	Месяцы	От года, годы
Вид обучения	Видеокурсы, вебинары, самостоятельное обучение	+ онлайн\оффлайн курсы, техническая документация	+ специальное образование, много практики
Результат	Создание прототипов, проверка гипотез, проведение экспериментов	Реализация процессов с несложной логикой	Сложные группы бизнес-процессов, интеграции, высоконагруженные системы

Использование готовых партнерских решений (1\2)

Плюсы и минусы

Необходимость отслеживания версионности:

- Учет совместимости версий базового продукта и партнерского решения
- При повышении версий, установке обновлений, строгое одновременное изменение версий

Зависимость не только от вендора, но и от партнера:

- Дополнительное лицензирование (базовый + партнерский продукты)
- Сужение спектра потенциальных подрядчиков в части партнерского продукта

Сохранение рисков усложнения комплексного решения:

- Необходимость балансировки соотношения Low\No-Code и Pro-code в комплексном решении.
- Учет особенностей как базового, так и партнерского функционала в ходе собственной разработки.

Повышение скорости внедрения:

- Пакет методологически и технически готовых решений.
- Экспертиза партнера-разработчика в ходе внедрения.

Сбалансированное использование Low-code и Pro-code:

- Сохранение Low\No-Code преимуществ.
- Возможность использования в рамках крупных предприятий при приемлемой скорости внесения изменений.

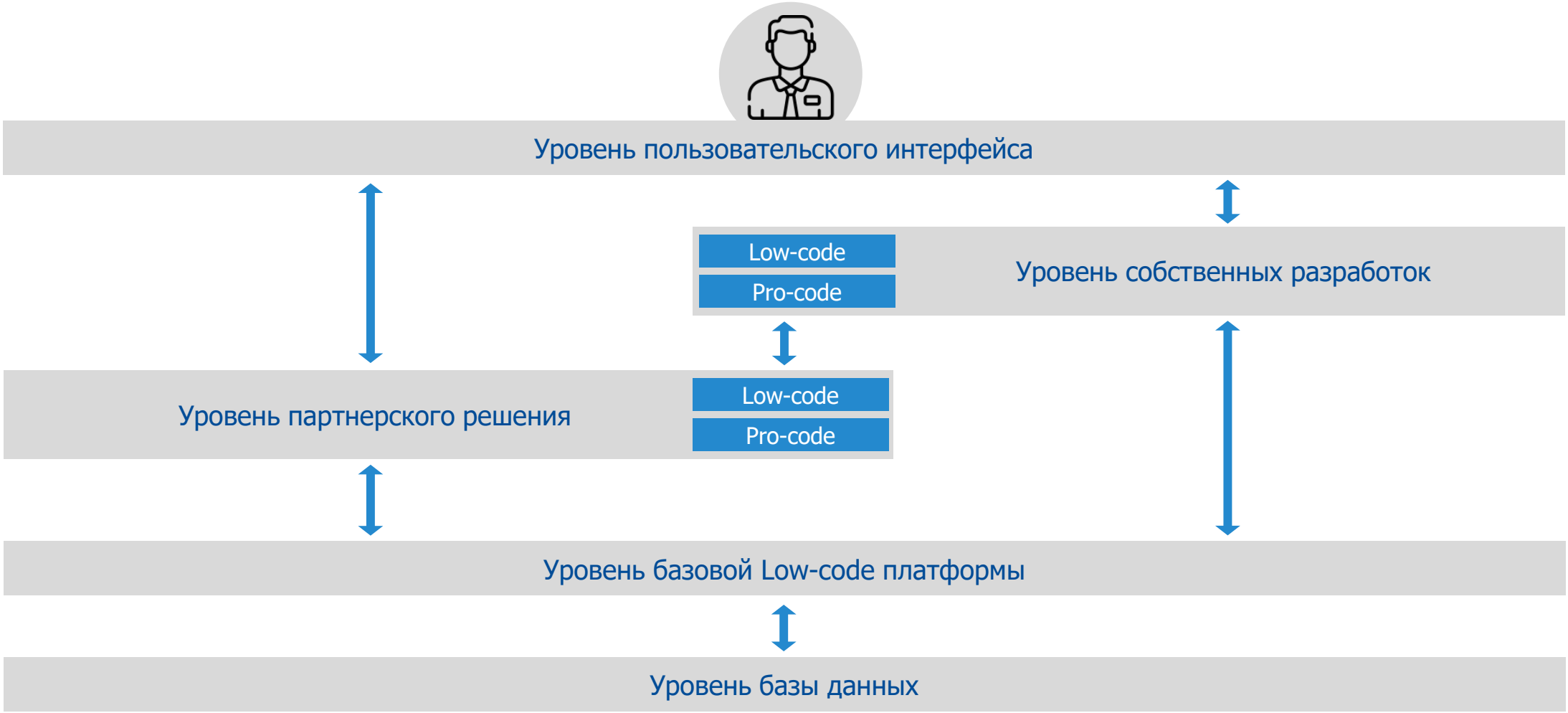
Скорость формирования команды внедрения:

- Возможность привлечения специалистов партнера-разработчика.
- Параллельный набор и повышение внутренней экспертизы.

НОРНИКЕЛЬ

Использование готовых партнерских решений (2\2)

Уровни «гибридного» Low-code решения



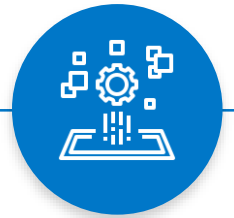
НОРНИКЕЛЬ

Выводы

Затраты на программное и аппаратное обеспечение при внедрении Low-code проектов принципиально не отличаются от затрат при внедрении ИТ-систем других классов.

Low-code платформы отлично подходят для прототипирования, сокращают разрыв между бизнес-пользователями и IT для формирования видения будущей системы. На этом этапе можно обеспечить экономию бюджета и сокращение сроков.

Оптимизировать сроки и стоимость внедрения, а также восполнить недостаток внутренней экспертизы можно с использованием готовых партнерских продуктов (CRM, ITSM и т.п.).



Эффект от внедрения увеличивается, когда Low\No-code являются «слоями» гибридной платформы совместно с Pro-code.

Автоматизация сложных и высоконагруженных бизнес-процессов требует кастомизации, которой, однако, не стоит увлекаться, чтобы не потерять преимущества Low-code и экономии, полученной на проекте, в ходе эксплуатации.

Методики Low-code и No-code дают возможность компаниям значительно снизить порог входа для сотрудников, желающих принимать участие в разработке программного обеспечения, что позволяет сократить затраты в ходе эксплуатации.

НОРНИКЕЛЬ

Играйте по правилам BPMS.
Не так плох Low-code, если умеешь его правильно "готовить”!

Максим Тарасов
Руководитель практики систем управления бизнес-процессами