定义软件开发新模式

中国企业级无代码开发白皮书

2021年

摘要

定义

企业级无代码核心包括：①无代码：改变以往业务需求与编程逻辑的对接方式，以无代码（不生产代码）的方式对接业务需求，从根本上改变了生产方式，节省中间过渡环节；②企业级：能够满足逻辑、表现形式复杂度较高，容量要求、安全可靠性要求高的企业诉求；③数据驱动：与表单/模型驱动不同，企业级无代码完成了从单纯的形式层抽象到数据层抽象的转变，以柔性的数据为抓手打造业务与数据闭环，使得应用程序能够自动适应业务需求的变化，并于用户侧表现出完全无码状态

溯源

随着数字化发展不断深入，客户需求与底层技术发生翻天覆地的变化。传统的软件开发方式显然已无法快速响应由产业环境变化而导致的企业业务诉求变化；其次，我国软件产业迅速发展，旺盛的IT人才需求与当前人才供给能力不匹配，人才贵、流失率高等问题成为限制企业发展的重要因素；下游企业传统信息化建设效用低下，对其自身建设需求的认知模糊会进一步传递至乙方（IT服务厂商及软件企业），导致双方“降本”、“增效”、“提质”诉求均无法达成。稳健经营、创新发展，打造差异化竞争力，寻求第二增长曲线成为当下厂商的共同诉求

变革

并非日光之下无新事，而是革命性的改变一旦发生，我们便习以为常。纵观软件工程发展史，无论是从面向过程到面向对象的工具语言变化，还是Scratch、 Axure以及我们最熟悉不过的Excel软件出现，都是人类追求无码化，降低开发应用门槛、提升效率的大胆实践。企业级无代码通过对产业分工、商业模式、开发模式、开发流程、开发者角色的变革，推动软件工程向前跨越了一大步

展望

我国无代码起步较晚，但近两年在市场、疫情、资本多重驱动下迈进了加速发展期，市场需求体量庞大。随着厂商纷纷入场，无代码开发技术的打磨与沉淀，开发平台生态体系及行业标准的逐渐完善，AI、数字孪生等前沿技术的深度应用，传统开发“思维的枷锁”会被进一步打破，实现全民创新开发、全流程全域的无码化指日可待

溯源：低/无代码演进

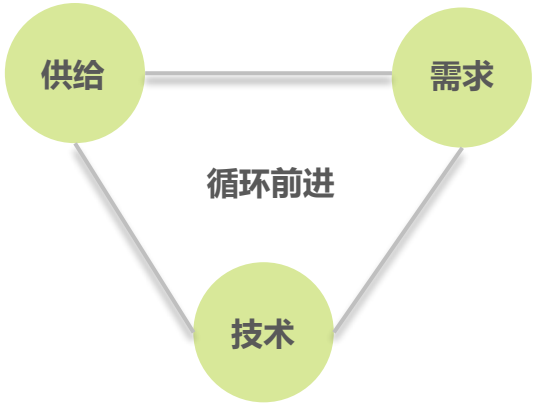
甲乙方诉求和技术进步共同促进低代码不断演进

低代码由Forrester于2014年提出，但其理念并不新鲜，实际上从纸带打孔、到汇编语言、到高级语言，再到各种IDE、各种框架，人们始终在试图屏蔽底层的复杂性与难以理解性，通过归纳、抽象、封装，进而通过点拉拖拽及少量代码来快速完成应用程序的开发。早期的Access数据库、QuickBase、PowerBuilder等，其实都可以看作低代码的雏形。因客户需求和底层技术的不断变化，低代码形态也随之变化，例如Dreamweaver等所见即所得的网页三剑客，作为低代码的一种形式，风光一时，但随着B/S架构的兴起、前后端的分离以及网页程序化，其风光不再。当下，低代码应当是指云原生的、可水平扩展的应用程序开发平台（APaaS）。无代码是指不需任何代码的应用程序构建方式，既可看作低代码的子集，也可看作低代码的高级形式。

三个视角下的低/无代码发展

易复用、易理解、易分工

降本、增效、扩张是乙方企业（软件开发者和集成商）的追求。然而，软件业早期是个“手艺活儿”，离不开少数人的经验与创造力。之后，随着前后端分离、技术栈的分化、框架的不断丰富和DevOps平台的普及，软件产业摆脱了手工作坊，成为了集团化作业。但是如何提高复用程度、摆脱高级技术人员限制，从而降低风险、降低成本、提高效率，并实现企业的规模化扩张，是乙方企业永恒考虑的问题，正是在这种源动力下，企业不断以不同的路径尝试易复用、易理解、易分工的低/无代码。



贴近市场、贴近客户、贴近员工

市场迅速且加速变化，没有任何企业可以以一成不变的策略做到基业长青，“敏捷”成为众多管理者的共识。如何快速响应客户、如何快速调整组织架构、如何快速试错、快速迭代，成为企业管理者越来越关注的问题。采买成熟软件系统、采用传统方式外包开发或自主开发，共同面临一个问题：系统上线不久甚至还未上线就已不再适用。另外，开发中，业务人员参与度低的软件系统，往往不具有良好的C端体验。敏捷、快速、业务人员可参与的低/无代码，成为甲方企业感兴趣的方向。

前端技术，使得点拖拉拽有更强大的性能和更良好的体验；Restful API和Graph API使得内外数据互通；软件定义及云原生技术，使得业务负载的水平扩展接近于无限；大数据技术，打通了从收集到实时分析的价值闭环；人工智能技术，使得数据作用再次放大，从辅助分析，到辅助决策，到自动决策；物联网，连接了物理世界与虚拟世界……这些技术，共同为低/无代码铺平道路。

来源：艾瑞咨询研究院自主研究及绘制。

演进：常见低/无代码应用形态

无码程度和复杂场景适应性是主要区分依据

根据开发中的无码程度、应用场景、使用者，以及复杂场景适应性，可以把低/无代码分为传统软件开发、轻量级无代码、企业级低代码和企业级无代码等几种形态。几种形态并非是完全的取代关系，它们将在较长时间内，在不同类型、不同规模企业的不同业务场景下，具有自己的生存空间，但企业级无代码，作为“量变引起质变”的软件开发模式，因重塑了软件开发模式、企业业务增长模式、社会生产关系等，将具有越来越高的占比。

低/无代码的不同形态-开发模式视角

典型产品形态	传统软件开发	轻量级无代码	企业级低代码	企业级无代码
开发模式	**代码驱动**；早期主要通过编程IDE及工具包实现软件开发；后期，可基于丰富的前后端开发框架、组件，框架自生成部分基础代码	**表单驱动**；预先设定的规则，应用开发（业务人员）拖拉拽配置生成表单、可视化看板等	**模型驱动**；通过建立模型来定义数据关系及流程逻辑；低代码平台可自动生成软件框架代码和基础代码	**数据驱动**；通过对具备柔性的内外部数据采集、存储、加工、使用，打造数据价值链，驱动并支撑应用智能生成
无码率	完全代码开发	简单场景可无码开发 复杂场景需大量代码	平台是无代码，但函数与系统是解耦的，支持自写代码开发	大部分场景不需要代码; 限制代码的反向输出
目标客群	**程序员**需同时关注业务逻辑和底层技术的实现	**业务人员**通过简单拖拉拽配置生成表单、可视化看板等	**程序员关注核心业务逻辑的实现**	**针对企业通用场景，业务人员配置生成**；其余场景可通过少量代码、插件满足
落地场景	所有应用场景	部分特定、简单轻量场景集中在管理&分析领域	企业级复杂应用	企业级复杂应用

注释：仅展示部分企业级应用软件无码化应用形态；四大应用形态并不是相互替代的关系，从左到右的顺序仅代表形态产生的先后。
来源：专家访谈，公开资料，艾瑞咨询研究院自主研究及绘制。

演进：驱动模式

围绕着样式、逻辑、数据，低/无代码产品不断朝无码化演进

低/无代码平台是软件开发生产力工具，其产品形态、驱动模式的变革都是在社会经济飞速发展、生产力不断提升需求下的产物，反映出大众对于“从何处入手”才能赋予工具软件更高价值的诉求；是**围绕着样式、逻辑、数据**，在追求极致无码的进程中，将业务逻辑转变为编程语言，再抽象成为面向大众，普适易用、图形化、可视化产品的一次次尝试。三种驱动方式特点鲜明，与其所承载的低/无代码产品形态一样，将在较长时间内具备一定的市场空间。值得一提的是，**数据驱动的企业级无代码改变以往业务需求与编程逻辑的对接方式，使得生产方式本身发生变化**：从原先先生产代码，再对接业务需求，转变为以不生产代码的方式直接对接需求；无需对业务逻辑做过多抽象来适应编程语言，而是让程序主动适应业务诉求，省下中间过渡成本，大幅降低学习及技术门槛，让普通人也能“所想即所得”。

驱动模式定义及特点

	定义	特点
数据驱动	数据驱动着眼数据，重视解决数据“从哪儿来”，“给谁用的问题”，不依赖于数据结构，也不依赖于业务流程本身；数据驱动可被认为是模型驱动、表单驱动的集合	■ 切入角度：以数据资产的挖掘、应用及管理为核心，重视业务与数据闭环打造 ■ 数据具备柔性，形式具备多样性：数据是软件开发的基础，业务/其他因素的变化首先会改变数据属性，其次才是数据结构。只有以可积累、柔性的数据带动并支撑着形式的多样性，才能实时响应复杂多变的市场需求；不需要在开始设计数据模型时就做到“精益求精”，只要数据资产利用得当，后续可持续、敏捷修改
模型驱动	核心在于数据结构/表结构，即通过数据结构的定义来描述业务的核心能力；具体而言，模型驱动在构建软件时，会先构建数据结构，再构建业务场景	■ 复杂场景满足度高，形式丰富：模型驱动对大型软件系统构建友好，形式丰富；支持复杂业务逻辑转化，应用场景的局限性低。 ■ 平台灵活性/业务响应度较低：围绕数据结构来实现业务场景的构建，业务场景发生变化，数据结构就需要改动，软件开发工作量加大；在需要进行数据资产深度使用、或被其它系统调用时，通常需要数据中台的支撑，在一定程度上限制了数据价值的发挥
表单驱动	表单驱动是业务/流程驱动，通过将业务抽象成普适易用的样式，供只懂业务逻辑的人员使用	■ 适用性：可视化适用于管理型软件，对不懂代码逻辑的业务人员友好； ■ 局限性：无法实现超复杂、容量较大的软件构建

来源：艾瑞咨询研究院自主研究及绘制。

定义：企业级无代码

兼具企业级和无代码双重属性

在传统IT开发思维下，低代码尤其是无代码只能适应较轻的场景，难以担当企业级软件开发的重任。这是因为，不管是表单驱动还是模型驱动，尽管都是具体问题的一定抽象，但抽象仍然不彻底。当抽象不彻底时，其可迁移性和普适性便不足，从而使得在软件开发中捉襟见肘、四处碰壁。只有在表单、工作流、权限等基础上再度抽象，将其“无差别”看作数据时，才可使得适应性进一步增强。基于数据驱动的无代码，可满足大多场景下企业级IT软件开发需求，因此具有企业级和无代码的双重属性。

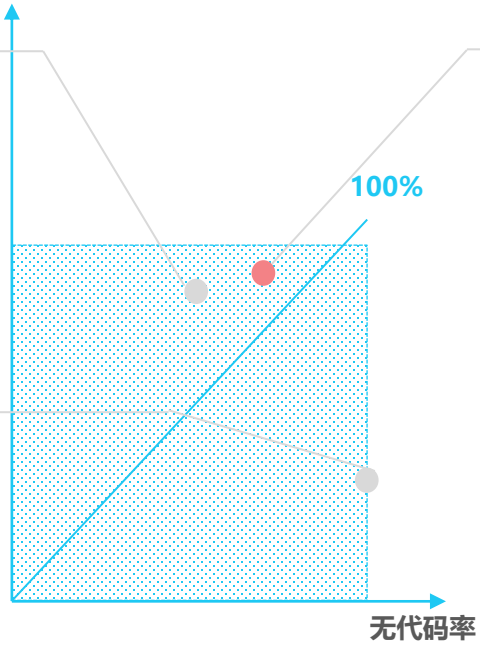
低/无代码/企业级无代码概念界定

低代码

- 低代码没有脱离软件工程的思路，而是软件开发技术持续迭代，在进一步降低技术门槛、提高开发效率需求推动下产生的生产力工具。
- 可视化、组件化以及框架化是其发展的主要方向，旨在通过为开发者提供可视化的应用开发环境，降低或去除应用开发对原生代码编写的需求量， 进而实现便捷构建应用程序的一种解决方案。

无代码

- 无代码开发平台属于低代码平台的一种，不提供或者仅支持非常有限的编程扩展能力，在提升易用性的同时“牺牲“了可扩展性、对复杂业务场景满足度。



企业级无代码

- **“企业级”软件开发：**①资源规模：站在企业整体视角来审视软件开发，内外部资源投入庞大； ②技术难度：应用结构复杂，涉及内外部资源配置，事务密集、数据量大、用户数多，对于不同应用之间的连接性、交互性，安全隐私的要求高；③需求升级：对软件质量要求高，包括对业务响应度、能力可复用性、升级和维护的平滑无感。此外，应具备推动企业创新的附加能力。
- **“无码化”：**①不等同于零代码，是数据、分析及管理域的绝对无码化，运营域仍需开源技术、少量代码作业来提升平台的复杂场景满足度、可扩展性、易用性等；②不是将软件开发应用的难度前置，即无代码平台“拖拉拽”，也不是将难度后置，即开发人员承担繁杂代码编写工作，而是机器与人力分离：能让机器做的事情绝对不让人动手； ③业务与数据闭环的形成使得应用程序能够自动适应业务需求的变化，于用户侧表现出完全无码状态。

来源：专家访谈，艾瑞咨询研究院自主研究及绘制。

核心能力：技术指标

大数据量、高并发，完整覆盖软件开发全周期

企业级无代码开发技术指标

数据视角

① 每日数据产生量达到TB/PB级
② 总数据量达到PB/EB级
③ 数据具有结构化、半结构化和非结构化等不同类型
④ 数据源具有内部、外部等两种，且支持通过自定义API和Webhook的双向操作，支持IoT数据的采集
⑤ 支持多种数据库

开发视角

① 完整覆盖软件的全生命周期，包括设计、开发、测试、运维等
② 具有CMDB概念，可进行统一配置
③ 在②的基础上支持版本管理，及相应的回滚、AB测试、灰度发布等
④ 支持物理机、虚拟机、容器、公共云等多种部署方

业务视角

① 除操作系统、数据库、中间件、安全、游戏、交易等特殊领域外，零代码覆盖客户90%以上业务场景
② 以Python、JavaScript等少量胶水性脚本语言进行低代码扩展，可覆盖98%以上业务场景
③ 完整覆盖客户表单、流程、搜索、集成、分析等企业

安全视角

① 代码安全，平台经过安全加固、风险扫描等，确保基础设施的安全性
② 数据安全，支持数据的加密处理和隐私保护，数据的权限控制和溯源跟踪等
③ 应用安全，对开发的应用分层分级权限控制、访问日志跟踪、安全隔