Beyond Low-Code

A critical look at the category

By Gastón Milano | October 2020

GeneXus™

The year 2020 will stay in everyone's memory for obvious reasons: COVID-19.

There is no doubt that humanity has experienced an almost unprecedented event this year.

In addition to people's health around the world, COVID-19 has put governments, industries, and businesses to the test.

It has also accelerated the digital transformation processes and revealed some traits that are necessary to survive these changes and that must be taken into account in the future.

In all sectors, the need to quickly adopt a **real, profound digital transformation** was laid bare. All of a sudden, medicine had to use telemedicine, education had to be offered remotely, shopping obviously shifted dramatically to online channels, contactless payments flourished, invisible voice interfaces were used increasingly more often, and all sorts of traditionally physical things became virtual and remote. Thus, countless industries accelerated a digital transformation that was already taking place. We all knew it was going to happen, but before COVID-19 it was advancing at a very slow pace.

The software industry now faces a special challenge and opportunity to support and empower the industries that have succeeded and **provide solutions and alternatives to those industries lagging behind**. In particular, there is a relatively new category of platforms and tools capable of quickly bringing about major changes in companies: they are called "Low-Code Platforms" and GeneXus competes in this category.

The **"Low-Code"** platforms have emerged as an alternative to traditional software development, a set of tools that allow creating systems using various editors, languages in general, graphics and some type of code generation, as part of a quest for a better, more effective, and more efficient way to develop software.

·The debate about Low-Code

Although in this circumstance **GeneXus** belongs to this category (and it is always good to belong to some kind of classification), I think there are some concepts in the name and in what the category itself means that I disagree with and that lead me to write this article and take part in the discussion.

Regarding the meaning of “Low-Code” (“low level of coding required”), I think it should be pointed out that its objectives of 1) building business software systems faster than with traditional methods, and 2) that they can be built by people with different levels of technical expertise, is a far-reaching and ambitious goal that I share completely.

However, there are some aspects that I disagree about with those who often talk about “Low-Code,” and I’d like to elaborate on these issues:

1) Its name

The name **“Low-Code”** focuses on a feature that is not necessarily a benefit: Writing a little or a lot of code doesn’t mean anything good or bad per se. The fact that a project requires little code does not necessarily imply that it will be easy to maintain, understand, or extend after completion.

The terms “Low-Code” highlight the way in which systems are built with these platforms and leaves aside the most important thing of all: the benefit of creating software with this type of solution.

Call me quixotic, but in these times of crisis I would like to belong to a category whose name indicates why we believe that **tools like GeneXus are relevant to the world.**

Because in this new world it doesn’t matter if I write little code or not, it matters much more if I accomplish my goals in the required timeframe. Can I build 3 applications in 1 week to solve a real problem? Can I extend those applications or are they rigid or limited? Can I create and interconnect systems in a few days? Can I quickly update the developed system if the reality changes? Can I easily switch technologies if I have to leave a particular provider?

One thing we do know is that today reality is changing more than ever before, and it will continue to change increasingly fast, perhaps even faster than technology itself. Therefore, the focus on how we build something new very fast is important, but it is far more important to focus on how we manage to adapt and evolve systems as rapidly as the world changes while we are working.

Writing little code is not our goal; our goal is to build tools that will enable useful transformations in the various industries that can **benefit from our platform and expertise.**

As long as there is no other category, we will keep talking about “Low-Code” and even **“Multi-Experience”,** but the truth is that I would feel more comfortable in other categories, such as **High Performance Platforms, Evolutionary Development Platforms, or Knowledge-driven Platforms.**

2) Graphic languages

Another thing that intrigues me is that having graphic languages seems to be an essential element in this category.

It is true that many times an image is worth a thousand words, but it is even more so that detailed and l**ong-lasting knowledge is written on words and formulas.** Suggesting that the graphic language by itself will allow for the evolution and storage of the business knowledge of a system is naive and overly optimistic, not to say completely wrong. The wide range of existing realities, situations, needs, and people, and the various forms of knowledge cannot be simplified or reduced in such a way.

It is true that a graphic language can often, and easily, make a tool more attractive. It is also true that a graphical interface can reduce the level of knowledge required to carry out some types of more standard solutions, but it is not clear if it is the best option when it comes to creating, understanding, and preserving knowledge. There are studies and analyses from both sides, and certainly there is no consensus today that something completely graphic is the ideal path, particularly for the creation of complex systems that evolve over time.

We only know that if something is not going to last over time, surely it is the code. Whether it is a little or a lot, the code will disappear, it will change, and we are convinced that knowledge cannot be stored either in lines of code or in drawings. The knowledge must be kept in some kind of abstract knowledge base, which is both **flexible and independent of technology**, in order to facilitate the evolution of the software solutions to be generated.

· The good things about Low-Code

Nevertheless, I do think the “Low-Code” category is helping us realize what we need in the software industry.

Why is the category necessary?

Because in this world that is full of all kinds of uncertainties, we must evolve towards new ways of building software. Below are my reasons why this category is necessary (regardless of its name)

Time

The time required by today’s companies to build solutions, regardless of their industry, cannot be met with traditional software development methodologies.

People

Building multiple experiences for systems implies involving an endless number of specialists, which is often impossible for cost reasons.

The time required for training in all the new technologies that need to be taken into account for a complex system is very long.

The Future

Evolving from manually written code is very time-consuming and costly, and often difficult for companies to afford.

Most likely, the people who build the system today will not be the people who will maintain it in the future. Can I really leave the knowledge of my company’s system written in code that nobody will understand in a few years?

Today’s complexity

Because making a system almost always implies creating a complex information system that involves the integration and understanding of various technologies. Even when high-performance experts are present, the lack of task automation leads to an endless number of costly errors.

As Richard Feynman said, **“There’s a big difference between knowing the name of something and knowing something.”**

I understand that the name of the category is not the best. I think that even what some people think that name means is not the best either, but it is also true that in essence what this category brings to the market is something we have long been waiting for. We may have agreements and disagreements about this new category, but what’s most important is that it exists and is getting stronger. As Breogán Gonda, one of GeneXus’ Co-founders usually says: **“We welcome the competition.”**

GeneXus™

MONTEVIDEO - URUGUAY	Av. Italia 6201- Edif. Los Pinos, P1	(598) 2601 2082
CIUDAD DE MÉXICO - MÉXICO	Hegel N° 221, Piso 2, Polanco V Secc.	(52) 55 5255 4733
MIAMI - USA	8950 SW 74th Ct, Suite 1406	(1) 201 603 2022
SÃO PAULO - BRASIL	Rua Samuel Morse 120 Conj. 141	(55) 11 4858 0300
TOKYO - JAPAN	2-27-3, Nishi-Gotanda	(81) 3 6303 9381
	Shinagawa-ku, Tokyo, 141-0031	(81) 3 6303 9980