Rep++, a low-code platform toward continuous delivery

Big desktop applications, designed from requirements that are as numerous as they are rigid, that take months if not years to develop, and that target only one output device are on steady decline. Instead, new lightweight, flexible, rapidly deployable, multi-device, cross-platform applications that work seamlessly together[1] are taking hold, becoming the new standard and driving the trend to an accelerated rate of delivery. To deliver better and faster in order to gain or retain a competitive edge, organizations must move toward a lighter, more versatile and more productive approach to application development. To that end, they need to:

- Accelerate the rate of application development, release and delivery.
- Create multi-platform, multi-device applications.

Low-code platforms

Low-code platforms are gaining popularity as a way to achieve the flexibility and rapid delivery that characterize this new shift in application development. Low-code platforms are environments that "...lower the barriers between requirements and delivery"[2] by providing:

- Minimal hand-coding.
- A single platform where requirements, design, development and deployment are shared .
- A centralized environment with increased automation for configuration, maintenance and delivery.

Low-code development shares core features of agile approaches such as the fast, iterative development cycles and frequent deliveries, but it goes beyond agile[3]. Low-code development encourages:

- Minimal hand-coding using "easily configured data models that eliminate data integration headaches", and tools for quick definition and assembly of applications[4].
- Flexibility in experimenting and testing ideas and designs. The iterative approach makes it easy to try out creative designs and new ideas and learn from them[5]. Experimentation not only can lead to unexpected innovations, but it also lowers the resistance to change that is so characteristic of big, rigid and over-planned projects.
- Continuous testing and feedback. In particular, close involvement and rapid feedback from the customer is an intrinsic part of the process, and earliest involvement is the best[6]. Fast iterations demonstrate progress, identify possible enhancements or pinpoint problematic issues, helping all stakeholders validate and clarify the requirements by producing functional prototypes incrementally. Since code is delivered with small enhancements each time, it can be corrected faster and more easily, reducing risks while providing high quality assurance.
- Continuous delivery through automation. Deployment can be a scary stage, especially with multi-platform applications: keeping them all synchronized requires exceptional coordination and is hard to achieve. By including deployment in the cycle, it becomes an inherent part of the process, and can be tested and automated like the rest of the application[7]; ultimately it becomes better understood and therefore becomes less of an issue[8]. Applications are delivered more rapidly and reliably, with minimal manual overhead[9].

[1] Forrester Consulting, p. 2.
[2] Richardson and Rymer, pp. 3-4.
[3] Richardson and Rymer, p. 12.
[4] Richardson and Rymer, p. 3.
[5] Forrester Consulting, p. 10.
[6] Forrester Consulting, pp. 6-7.
[7] Bittner and O'Donnell, p. 6.
[8] Bittner and O'Donnell, p. 12.
[9] Evans Research, p. 2.

- Responsive design for cross-platform, multi-device applications. Customers demand it, organizations that deliver benefit from it[10].

There are many benefits to low-code platforms. They enable rapid prototyping in cycles that produce smaller, more frequent deliverables. They help put in place more productive practices that include extended, cross-functional teams that can handle entire projects from design to delivery[11]. They encourage collaborative efforts as an intrinsic part of the process. They promote accelerated deployment. They centralize model, management, and configuration to facilitate development and delivery over multiple platforms. They enable better development practices such as decoupling layers of an application to enable independent changes, thereby implementing separation of concerns, and increasing automation to improve consistency and quality.[12,13]

To summarize, low-code platforms foster more productive development and delivery practices that can quickly demonstrate a project's value.

How Rep++ can lead you to low-code development

Rep++ possesses many characteristics of low-code platforms that can help organizations achieve continuous delivery.

Rep++ intrinsically supports minimal hand-coding through the Rep++ model and model execution mechanism.

- The Rep++ model is designed to define, group, and organize the metadata in a coherent whole. It is easily configurable and understandable by all authorized stakeholders, and contains no code. It is centralized and shared among all project members, and thus ensures consistency, from design to deployment.
- The Rep++ model reduces coding through reuse. For instance, Rep++ manages all occurrences of a field in different forms by reusing its definition wherever needed. A single definition in a single location maximizes reuse and reduces maintenance.
- The model execution mechanism used by Rep++ greatly reduces coding since a running application will use a meta-object to query the model directly for the information it needs, for instance to validate data, instead of using code. It also means that any change to the model is automatically applied to the applications, without additional coding, often without recompilation or other intervention.

Rep++ provides extended automation to the entire development process—from design to delivery.

- The Rep++ frameworks minimize hand-coding by automating a large portion of the applications for various architectures and technologies. Whether ASP/MVC, WPF, SPA or other technology, complex and repetitive functions such as navigation, creation, modification, transaction management, error management, and many more, are automatically taken care of. The developers only have to focus on the business logic of the application.
- The Rep++ wizards enable you to create applications for different architectures, from simple two-tier applications with business objects in the same assembly, to multi-tier applications or services where business objects reside in different tiers and data transits using POCO entities between the different tiers. The wizards define all the necessary objects for each tier.
- Rep++ automatically optimizes and takes care of the data access layer, supporting the most popular relational databases, whether in client-server or multitier architecture.
- Rep++ automatically creates the presentation layer and most of the business and services layers, for the chosen technology and architecture.
- Deployment is a matter of updating the repository that contains the model. In most cases a simple restart of the application is sufficient.
- The same Rep++ model is used to create the same application for various architectures and technologies, whether desktop, web, or responsive mobile.

Rep++ makes it easy to try out new ideas and designs. Modifying the model or integrating new specifications during a project is not a source of tension, but instead a source of innovation.

Rep++ promotes continuous testing and feedback from customers, employees and partners. Because the model can be shared easily with all stakeholders, including the customer, rapid feedback is easy to achieve, even remotely.

[10] Richardson and Rymer, p. 3.
[11] Bittner, p. 17.
[12] Bittner, p. 25.
[13] Bittner and O'Donnell, p. 9.

Rep++ helps put in place better development practices, such as:

- Separation of concerns. The presentation and business objects are cleverly designed for their respective layer.
- Elimination of code replication. A single definition for a single piece of information reduces ambiguities and facilitates maintenance. Rep++ inherently applies this principle through its Rep++ model, where the metadata is centralized and reused instead of being replicated. Other features of Rep++ objects, such as custom code methods and Rep++ events, also prevent code replication.

Summary

To deliver better and faster in order to gain or retain a competitive edge, organizations must move to a lighter and more productive development approach. The Rep++ low-code development platform is an enabling tool for restructuring a development strategy toward continuous delivery. Because it provides extended automation, minimal hand-coding, lower resistance to changes, and other benefits of low-code platforms, Rep++ is one of the best low-code platforms available today.

References

Bittner, K. *The Business of Continuous Delivery*, Forrester Research, Keynote address notes, CD Summit, New York, NY, June 19, 2014.

Bittner K. and G. O'Donnell. *The Eight Tenets of Faster Application Delivery*, Forrester Research, April 15, 2014.

Evans Research. *Continuous Delivery: The New Normal for Software Development*, Findings from Survey of Software Development Professionals Commissioned by Perforce Software, 2014.

Richardson, C. and J.R. Rymer. *New Development Platforms Emerge For Customer-Facing Applications*, Forrester Research, June 9, 2014.

Forrester Consulting. *The State of Modern Product Delivery; Challenges and Trends Making Product Delivery Responsible, Iterative, and Collaborative In The Age Of The Customer*, Thought Leadership Paper Commission by Jama Software, November, 2013.