
Any2Any Integration

While web-based technologies have become the de facto standard for new applications development, the fact remains that large organizations have significant investments into “legacy” platforms. Core business functions continue to run on mainframes, midrange machines and older servers that are stable and effective. It is therefore essential that any new applications built in these multi-platform environments have the ability to integrate seamlessly into the network and access data from all these systems easily. This has been a frustratingly-difficult aspect of applications development – till zeroCode came along. Today, zeroCode is unarguably the fastest way to generate enterprise-class applications.

Enabling factors

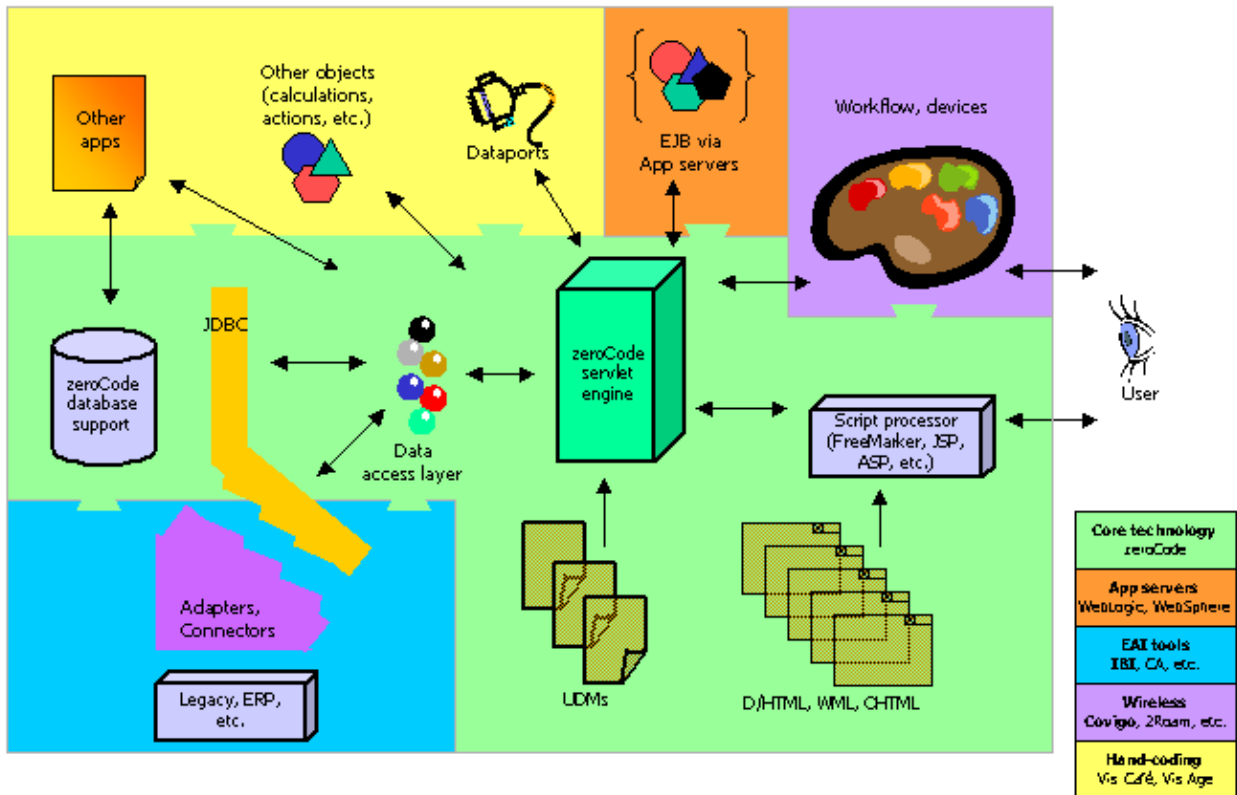
Applications built with zeroCode have the ability to integrate into and with any kind of computing environment, thanks to the following factors:

- 1 **The core technology is based on server-side Java.** The Java language gives zeroCode-built applications the ability to run on any platform that is JVM-enabled, which includes pretty much anything out there. And since the Java used is all on the server side, throughput to the client is very fast, leveraging all the advantages of a “thin” client.
- 2 **XML is the key structural mechanism.** An internal XML substrate gives zeroCode the ability to address any kind of user-interface, with cell-phone/WML access being integrated as easily as browser/HTML access. Client-side components like Java clients, while useful if present, are not a requirement. XML is also used to record server-side logic and that further exploits the abilities of XML.
- 3 **The data access layer is very well abstracted.** The manner in which zeroCode-built applications are generated, they have the unique capability of run-time re-targeting, where a designer can take a zeroCode application built for Oracle, say, and change a few configuration parameters to make it work as effectively as before against a database running in SQLServer, DB2, PointBase, PostgreSQL, etc. The application then automatically reconfigures itself to generate the appropriate SQL dialect.
- 4 **The reliance on meta-data enables platform-independent integration.** Applications built with zeroCode are generated beginning with the schema of the target database. This provides for some interesting capabilities. For example, zeroCode can generate an application with a schema that includes foreign-key references, although the target database may not have these defined (for historical or technical reasons). This ensures that new applications automatically enforce referential integrity on old databases that do not. This also provides opportunities where one zeroCode-built application accesses multiple databases in the same network, using server-side adaptors and connectors from vendors like IBM, IBI, BEA and Oracle. Such connectors include meta-data about their targets that zeroCode can use to build a common application that, say, runs a transaction needing and updating data in a DB2 database, an Oracle database and a VSAM structure, all at one shot.
- 5 **Dataports give zeroCode built applications infinite extensibility.** Applications built with zeroCode include an architectural interface whereby hand-coded components can provide multi-way access to non-conventional data

sources and sinks. These interfaces, called dataports, can be used for simple things like rendering charts with data from a database, using a commercially-available charting environment through complex interfaces to third-party environments like ERP and CRM systems. As flexible, simple-to-implement interfaces, dataports give applications designers unlimited capabilities to extend their sites, using hand-coding.

Technology

Below is a diagram that depicts the architecture that zeroCode-built applications run on. Around the zeroCode-generated site components are a number of options



that applications designers can use to extend and integrate such applications into other frameworks, platforms and technologies. Beginning with data-side interfacing to databases and file systems, zeroCode-built applications can work with transactions in other applications, user-interfaces built in various technologies and objects built for specific business functions. Beans available in app servers can be retrieved, manipulated and integrated into the zeroCode UDM (user data model) structure. Third-party user-interaction frameworks like workflow engines and navigation modeling toolsets can be used to build platform-specific UI for zeroCode-built applications. Native changes can be made to the generated code, to make them render in WML, cHTML and a host of other mark-up languages. Most importantly, at a data level, adapters and connectors can be used to build applications that access multiple data sources, even in one transaction.

A significant contributor in data access is the JDBC layer. A number of enterprise applications integration (EAI) situations depend on this layer for switching among various databases. Products from IBM, IBI, BEA, etc. all provide very good capabilities in this regard, allowing the applications layer to completely release itself from low-level database primitives. An application built with zeroCode relies on this

structure and extends these assumptions by generating the specific SQL dialect required for a given database at run-time, thereby using the database very efficiently. Designers thus do not have to worry about database-specific SQL issues in building an application.

Incidentally, zeroCode can generate applications in C# and ASP, instead of the Java servlet model. Applications can thus leverage the native functions and throughput capabilities of NT-based platforms as well as they can use the multi-platform portability of Java.

Benefits

In a computing environment that is reasonably standardized and includes a minimum number of platforms to work with, the ability of zeroCode-built applications to address a large number of platforms equally well may be irrelevant. But in reality, most organizations have a need to access data from a variety of platforms. This is especially true of larger organizations - and older organizations. Fortune 1000 companies continue to rely on proven systems running on older hardware, since they provide the best bang-for-buck even today. In that kind of an environment, zeroCode-built applications are ideal mechanisms to truly leverage the information assets of a company. Some of the specific benefits are listed below.

Integrability. Applications built with zeroCode integrate into any kind of information systems environment, as we have seen above. They are particularly well-suited for large, multi-platform organizations that have developed systems over the years and would like to leverage them with today's technology, for tomorrow's user demands.

Scalability. The code generated by zeroCode is tight, effective and well-organized. It includes techniques like connection-pooling, for example, where the generated application scales to a large number of users in a linear manner. A large number of factors can be configured in the application, increasing its scalability to a growing number of users. A multi-tiered architecture further reduces the sensitivity of the applications to the number of transactions handled. Business logic functions can be shared across many machines using commercially available load-balancing hardware and software.

Reliability. The mean time between failures of the generated code is very high and is controllable. The generated code is based on mathematical, patent-pending algorithms that are very robust in deployment environments. These objects can be integrated into app server environments from vendors like IBM and BEA, for further execution-time scalability.

Customizability. The look and feel of a generated application can be modified or completely replaced using a standard HTML editor such as FrontPage. Designers can use other editors that they are comfortable with. Also, zeroCode can generate applications in other user-interface languages like WML, further enhancing their usability.

Accessibility. zeroCode uses industry-standard technologies like Java and XML. Designers don't need to learn new languages to be able to use the framework. While the completed applications are Java-based, their creation and maintenance do not require Java programming skills. The basic skills of a web-master or business analyst are normally sufficient.

* * * *