



S-Square - LowCode/NoCode (LC/NC) Enabling Technology Presentation

Jeff Friedman,
VP, Sales & Customer Success

Version - 20221215_V1



Current Challenges in Traditional Application Development

Long Development Timelines

- Custom development with standard SDLC processes
- Long incubation period before seeing a MVP
- Minor changes require long turn around time for design, build and testing.

High Capital Expenditure and Operating Costs

- Investment in Software platforms and Infrastructure for custom development
- Higher support costs due to diverse support requirements

Disparate Technology Landscape

- Multiple small projects using disparate technologies
- No uniform platform to manage small developments

Developer Shortages

- Developer shortages and skill-set challenges
- Multiple small productivity projects get deprioritized

6 Generations of Programming Languages

First generation (1GL) - machine-level programming language used to program first-generation computers

Examples: machine-level programming languages

Second generation (2GL) - assembly languages. Examples: Assembly

Third generation (3GL) - more machine-independent (portable) and more abstract therefore more programmer-friendly than previous generations of languages

Examples: Fortran, COBOL, BASIC, Pascal, C, C++, Perl, Python, Java, JavaScript, Ruby, PHP, C#

Fourth generation (4GL) - include support for database management, report generation, mathematical optimization, GUI development, or web development. Examples: ABAP, Unix Shell, SQL, PL/SQL, Oracle Reports, R

Fifth generation (5GL) - any programming language based on problem-solving using constraints given to the program to make the computer solve a given problem without the programmer, rather than using an algorithm written by a programmer. Examples: Prolog, OPS5, Mercury

Sixth generation (6GL) - programming language based on visual development. The overall umbrella term for these is "NoCode". Examples: Appian, WEM.io, Bubble.io

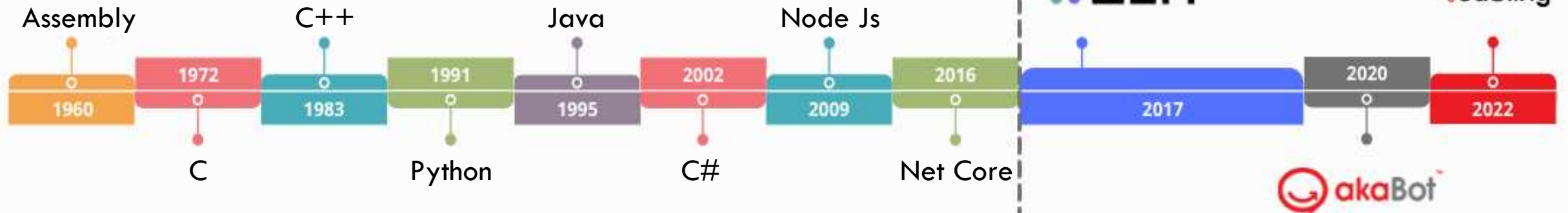
Reinventing Software Development

Traditional Coding

Requiring expensive, hard to retain code-linguists

No-Code

Empowering transforming support to employ business-knowledgeable techno-functional resources



Traditional computer languages require programmers to translate their thinking process into code built for the CPU and memory

Optimized for how we humans think. Converting natural thinking process into working software

Digital Transformation.
Legacy Modernization.
Business Velocity.

80%

COST REDUCTION

Empowers employing business knowledgeable (techno-functional) resources instead of costly, hard to retain code-linguists to build, deploy and maintain secure scalable enterprise-grade software.

10%

FASTER TIME-TO-MARKET

View app development in real-time. Deploy and update applications with a single click. Deliver software 10 times faster than traditional programming methods.

100%

ALIGNED TO BUSINESS

Translate innovative business ideas to custom software built with no code app builder at the speed of, and fully aligned with, business requirements.



Banks,
Financial
Services and
Insurance >



Healthcare >



Telecommunication
>



Education &
Training >



Manufacturing
>



Public Sector
>



Automotive
>

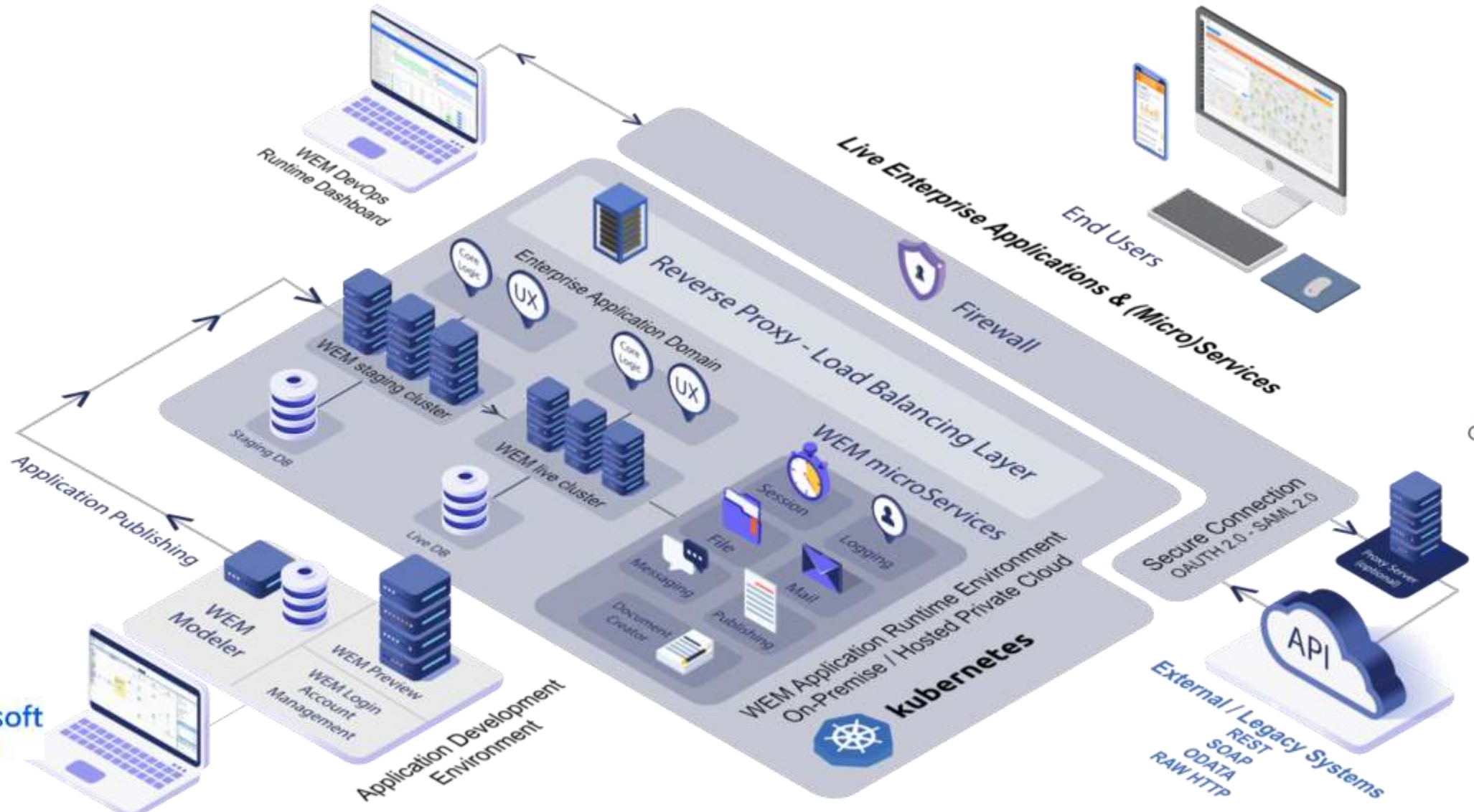


Real Estate
>

SCALABLE, SECURE CLOUD ARCHITECTURE



Google Cloud Platform



FLEXIBLE DEPLOYMENT OPTIONS FOR SHARED HOSTING, ON-PREMISE APPLIANCE AND PRIVATE APPLICATION CLOUD

3rd Party LCNC Marketplace Product Evaluation



Criteria	WEM	Betty Blocks	Power Apps	OutSystems	Mendix
Category	No Code	Low code	Low code	Medium to high code	Low code
Platforms	Web, native apps	Web apps	Web, native apps	Web, native apps	Web, native apps
Data Model	Drag & Drop	Visual Editor	Tables	Visual Editor	Visual editor
Visual Editor	Web-based	For backend apps	Web-based	Many designer	Web-based, desktop-based
Workflows	Drag & Drop	Action Modeler	MS Flow	Visual modeler	Visual modeler
Look & Feel	Custom templates	Custom js/css/html	Customizable	Custom js/css	Custom js/css
Environment	Public, private cloud, on premise	Public cloud, on premise	Public, private cloud, on premise	Public, private cloud, on premise	Public, private cloud, on premise
Release Management	Fully	Fully	Partially	Fully	Fully
Integration	All API standards	JSON, SOAP/REST	Office365, REST	SOAP/REST	SOAP/REST

Use Case – Card Reader Processing for Diplomat Tax-free Fuel Cards



This organization is the first service provider that helps diplomats in Europe to directly claim their VAT-free and Tax-free privileges. In cooperation with in-country authorities, the Diplomatic Card Company has developed a Card Issuing and Authorization platform. This innovative and excellent service, allows diplomats to buy VAT-free safe, simple, and swift. They provide fuel cards for Diplomats working in the Netherlands, Belgium, and the UK.

CUSTOMER CHALLENGES

- The existing years old application built in Java and SQL needed improvement.
- Automated invoicing, billing, and government tax-audit data submission.
- Betterment of fuel tax calculation algorithm that was faster and easier to maintain than their legacy system.
- Extensive logging system to keep track of all mission-critical application data changes.

WEM ADVANTAGES

- **From an old legacy system to an easy to use and no-code web environment to be maintained by non-IT skilled employees**
- **Agile development, week to week results, short time to market (live in 3 months)**
- **Use of existing data from legacy systems/integration with legacy systems**
- **Cloud solution offers flexible workspaces (not tied to a location)**
- **Easy to extend the application**
- **Fast return on investment**

PROBLEM

The organization needed to build a Tax-free Fuel Card registration system. They wanted to replace the previous legacy Java-based system built in Java and SQL. The main requirement of this company was to add and update all the new features in the system like updating of fuel usage from fuel card, integrated invoicing and billing system, data submission. The transition required from their years old legacy system built in Java and SQL, completely to a WEM system in 3 months in step by step roll out.

SOLUTION

They used WEM's platform to transition their years-old legacy system to a no-code environment. The invoicing and billing systems were fully integrated into the application and complex mathematical fuel tax calculation algorithms were developed and integrated. The platform so developed has a built-in automatic fuel quota tracker to accurately calculate tax-free and taxed deductions and send out quota limit notifications. The new application has an automated invoicing, billing, and government tax-audit data submission system. There's a feature that enables the uploading of fuel usage from fuel card registrations provided by Gas stations from text files.

Representative WEM Enterprise Customers



Thank You

Jeff Friedman,
VP, Sales & Customer Success

S-Square Systems, Inc.

4225 Executive Square Suite 600

La Jolla, CA 92037

+1 858-213-7063, +1 858-764-4441



S-Square

TRUSTED . TESTED . COMMITTED