ArchitectNow

WEBINAR:

Modernizing Your Custom(ized) Apps

PRESENTED BY

Kevin Grossnicklaus

info@architectnow.net

www.architectnow.net

INTRODUCTIONS, EXPECTATIONS, AND AGENDA

CONTACT INFORMATION

Kevin Grossnicklaus
President
ArchitectNow
kvgros@architectnow.net
LinkedIn
@kvgros

www.ArchitectNow.net
@architectnow
LinkedIn

ArchitectNow

TRANSFORMATION THROUGH TECHNOLOGY

WELCOME TO ARCHITECTNOW

Whether launching new Cloud or mobile apps or modernizing your legacy platforms we can help you identify the best options and work with you on bringing those ideas to life. To get the ball rolling, reach out and tell us a bit about your needs and we can start identifying solutions. There is no risk and we can quickly get to the point of providing initial ideas along with rough estimates of the costs and implementation times required with various recommendations.

info@architectnow.net
www.architectnow.net

ArchitectNow

ArchitectNow

TERMINOLOGY

What do we consider
“***Legacy***”?

- Unsupported Frameworks
- Doesn’t fit growing business needs
- Dated security implementation
- Difficult to maintain and update

ArchitectNow

Rebuild vs. Update

Building a brand-new house

Hire an architect

Hire a designer

Multiple iterations of getting it just right

Approvals

Tear down old and undertake a rebuild

Updating a house

Typically, just cosmetic

New Paint

Maybe some new flooring

Selectively update what gets too far out of date

ArchitectNow

What does updating mean?

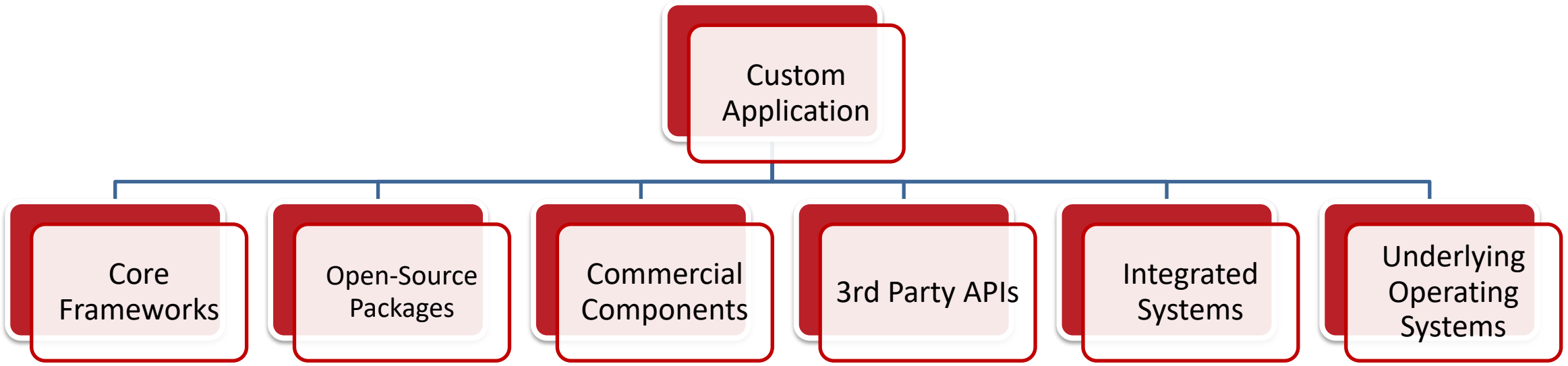
- Updating to fit business needs
- Taking advantage of integrating other critical software
- Keeping code and frameworks updated

- Latest Greatest
- New shiny tech
- Rewriting something that already exists

What is “DevOps”?

- Tools and processes for managing the software development process
- Versioning and branching source code
- Tracking requirements and bugs
- Automating deployments
- Managing test plans and results

HOW MODERN SOFTWARE IS WRITTEN



- Each component is dependent on multiple other components
- Every interconnected component is maintained and versioned independently.
- Each component is updated on their own schedule to:
  - Fix bugs
  - Increase performance
  - Patch security vulnerabilities
  - Add capabilities
  - Update their own dependencies

Why is software so interconnected?

Massive cost savings

More secure software

More capabilities

How do we determine when to build on top of something else?

Licensing

Cost (Open Source vs Commercial)

Support

Community

Documentation

What is DevOps?

- Where is your code managed?
- How is your code deployed?
- How is this process controlled or automated?
- Are their multiple environments for dev, QA, and production?
- Are things versioned and what is the impact of going back?
- How are environments monitored?

```
<ItemGroup>
  <PackageReference Include="Autofac" Version="5.0.0" />
  <PackageReference Include="Autofac.Extensions.DependencyInjection" Version="6.0.0" />
  <PackageReference Include="AutoMapper" Version="9.0.0" />
  <PackageReference Include="AutoMapper.Extensions.Microsoft.DependencyInjection" Version="7.0.0" />
  <PackageReference Include="Azure.Storage.Blobs" Version="12.7.0-preview.1" />
  <PackageReference Include="CsvHelper" Version="15.0.5" />
  <PackageReference Include="FluentEmail.Core" Version="2.7.0" />
  <PackageReference Include="FluentEmail.Mailgun" Version="2.7.0" />
  <PackageReference Include="FluentEmail.Smtp" Version="2.7.1" />
  <PackageReference Include="FluentValidation" Version="10.2.2" />
  <PackageReference Include="FluentValidation.AspNetCore" Version="10.2.2" />
  <PackageReference Include="Handlebars.Net" Version="1.10.1" />
  <PackageReference Include="Hangfire" Version="1.7.18" />
  <PackageReference Include="IronXL.Excel" Version="2021.9.0" />
  <PackageReference Include="MailChimp.Net.V3" Version="4.2.1" />
  <PackageReference Include="Microsoft.Azure.Cosmos" Version="3.13.0" />
  <PackageReference Include="Microsoft.Azure.CosmosDB.BulkExecutor" Version="2.4.1-preview" />
  <PackageReference Include="Microsoft.Extensions.Options" Version="3.1.1" />
  <PackageReference Include="Microsoft.Extensions.Options.ConfigurationExtensions" Version="3.1.1" />
  <PackageReference Include="Microsoft.Graph.Auth" Version="1.0.0-preview.5" />
  <PackageReference Include="Microsoft.Graph.Beta" Version="0.28.0-preview" />
  <PackageReference Include="Microsoft.Identity.Client" Version="4.20.0-wampreview1" />
  <PackageReference Include="Microsoft.IdentityModel.Clients.ActiveDirectory" Version="5.2.8" />
  <PackageReference Include="Microsoft.IdentityModel.Tokens" Version="5.6.0" />
  <PackageReference Include="Newtonsoft.Json" Version="12.0.3" />
  <PackageReference Include="NSwag.Annotations" Version="13.2.2" />
  <PackageReference Include="NSwag.Core" Version="13.2.2" />
  <PackageReference Include="Serilog" Version="2.9.1-dev-01154" />
  <PackageReference Include="Xrm.Tools.CRMWebAPI" Version="1.0.25" />
</ItemGroup>
```

RISKS OF OUTDATED SOFTWARE

In a perfect world you could custom develop a software solution and not touch it for many years while it works flawlessly!

Bugs

Flaw or fault in system

Browsers updates

OS updates

Cost more to fix

Security

Security moves fast, hackers move just as fast

You need to keep everything to update with the best security practices. The goal is to make your software a pain to break into.

Compatibility

Updates in underlying foundation

3rd party API updates

Compliance

Outdated software may not comply with regulations or industry standards.

ArchitectNow

Support

Vendors will only support for so long

EOL for various software

Harder to hire for

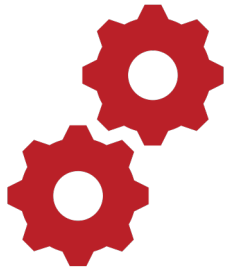
Monitoring and Tracking

Performance

Increased user demand can cause strain on older hardware.

Users across the globe expect faster load times today vs 10 years ago

ArchitectNow



Features

Adding new capabilities costs more than necessary

Much riskier to enhance older software

PLANNING AND BEST PRACTICES

Identify and Audit

Source Code and Other Assets

Development Environments

Hosting and Runtime Environments

Existing Documentation

Dependencies and Integrations

DevOps Platforms and Processes

Subject Matter Expertise (i.e. Power Users)

Development and Support Staff

Key Considerations

Risks if we do upgrade vs Risks if we don’t

Buy vs. Build

Upgrade vs. Rewrite (vs Retire)

Available Expertise

Cost and Timing

Dependencies

Cloud Native vs. On-Prem

Low Code/No Code Platforms

Some Tools of the Trade

Snyk
https://snyk.io/

Burp Scanner
https://portswigger.net/burp/vulnerability-scanner

Dotnet-outdated
https://github.com/dotnet-outdated/dotnet-outdated

Dependabot
https://github.com/dependabot

BrowserStack
https://www.browserstack.com/

***no "Silver Bullet" and we generally use specific tools to analyze specific platforms or environments**

ArchitectNow

CASE STUDIES

About Client

Their product/services consists of a large suite of data integration tooling that moves data between customers 3rd party systems. Their suite ranges from ingestion, managing, hygiene, processing, and more.

They already had a footprint in Azure but still had over half of their servers and storage on-prem.

Challenge

Insecure, dying, and outdated infrastructure forcing a move to cloud or a multi 7 figure bill coming their way

Solution

Lift and shift to the cloud. Then optimize and update

Impact

Reduced risk of day to day operations. Scale up/out as necessary (for less $). Improved monitoring and reduced cost.

ArchitectNow

About Client

Engineering, design, and manufacturing firm. Their products are installed in interior environments with no internet access. As an international distributor of devices their products are installed around in North and South America and Europe.

They utilize a proprietary mobile application (iOS and Android) to program and manage the products via BlueTooth.

Challenge

Mobile applications haven't been updated in many years
Older application is not compatible with newer devices and mobile OSs

Solution

Iterate updating application and then add new features.

Impact

Next time they want to add features they will have to spend a significant amount on updates and testing.

ArchitectNow

About Client

Large university with a significant footprint in medicine. 100+ year history and custom software needs going back decades.

They have internal applications and tools they custom developed that range from very legacy to very modern.

Challenge

50+ Applications that are running on unsupported windows OS.

Solution

The apps were either retired, replaced with SaaS, modernized, or lift and shifted.

Impact

Hard to maintain and leaves them open to security risks.

ArchitectNow

FINAL THOUGHTS

Recommendations

Take an honest assessment of what you have
Evaluate Risks
Gather Input and Recommendations
Justify Time and Cost

Ensure DevOps Setup, Support Process, and QA/Test Environments First
Plan and Execute Upgrade
Ensure monitoring or logging
DOCUMENT
TEST TEST TEST
Communicate Production Release and Process

Get an Extra Set of Eyes

Ask Around for Input
Research Options
External Resources for Guidance
Play to Your Strengths

CONTACT INFORMATION

Kevin Grossnicklaus
President
ArchitectNow
kvgros@architectnow.net
LinkedIn
@kvgros

www.ArchitectNow.net
@architectnow
LinkedIn

ArchitectNow

Thank you!

info@architectnow.net
www.architectnow.net

ArchitectNow