

EUROMNIS 2023

---

**DEVOPS WITH OMNIS STUDIO**

## AGENDA

- ▶ My journey to a better way of building Studio apps and why does this matter to you?
- ▶ What is DevOps and why I dislike that term
- ▶ How to DevOps with Studio

## MY JOURNEY TO A BETTER WAY OF BUILDING STUDIO APPS

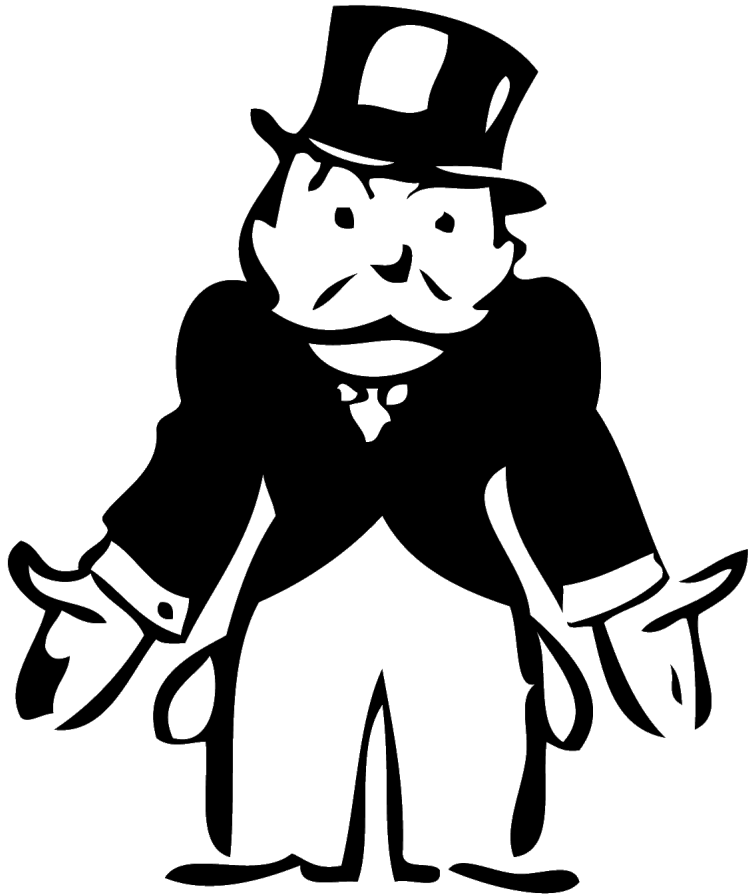
- ▶ CEO at Suran Systems, Inc., previously Software Development Manager
- ▶ Family-owned business turns 36 years old in December
- ▶ Our products used Omnis Classic in the 80's and 90's
- ▶ Moved to Studio in early 2000's
- ▶ Completed our Classic to Studio port by the late 2000's
- ▶ Hit a wall

## CHALLENGES IN 2009

- ▶ Developers were spending time helping our support team track down bugs instead of coding
- ▶ When we did code, we were spending more time fixing bugs than adding new features
- ▶ Bug fixes often caused regressions which was a double blow to user confidence
- ▶ Needed to switch off OpenBase to PostgreSQL
- ▶ Couldn't afford a shaky transition like the Classic to Studio re-write

## THE ROOT CAUSE OF THESE CHALLENGES

- ▶ As we learned and adopted new techniques, we rarely went back to apply them to existing code
- ▶ It was dangerous to work with the existing code because you could break something and not know it
- ▶ And yet, needing to add new features and fix bugs required us to work with existing code
- ▶ This lead to copied and pasted code, duplicated functionality, and increased complexity



WE FOUND  
OURSELVES DEEP IN

---

**TECHNICAL  
DEBT**

## WHAT IS TECHNICAL DEBT

- ▶ As a change is started on a codebase, there is often the need to make other coordinated changes
- ▶ Required changes that are not completed are considered debt that must be paid at some point in the future
- ▶ If technical debt is not repaid it can accumulate 'interest', making it harder to implement changes later on
- ▶ Unaddressed technical debt increases software entropy

## HOW DID WE GET INTO TECHNICAL DEBT

- ▶ We pushed to get the classic version ported
- ▶ At the same time, we need to build new features to serve our customers
- ▶ We didn't devote time to improving the code itself
- ▶ We were afraid to improve our code because it could cause new bugs (regressions)
- ▶ So we leaned on minimal changes at the expense of the right fix



## EFFECTS ON OUR BUSINESS

- ▶ Users had a negative experience because of bugs; they hated updating
- ▶ We spent more and more time fixing bugs than developing new features
- ▶ When we did fix a bug, there was low confidence it would stay fixed
- ▶ We needed to move away from OpenBase and develop new features to stay competitive, but how could we possibly do that without making things worse?



IT ALL CAME DOWN  
TO ONE ELEMENT:

---

**QUALITY**

## HOW COULD FOCUSING ON QUALITY HELP ADDRESS OUR CHALLENGES?

- ▶ Improve our user's experience
- ▶ User will stay with the product and invest more
- ▶ Reduce support time tracing down problems
- ▶ Free up development time to build new products
- ▶ Improve developer confidence in our codebase
- ▶ Learn to become better developers to build better products

## WHAT IS QUALITY?

- ▶ Quality is matching the product's **behavior** to its **expectations**
- ▶ If you don't understand the stakeholders' **expectations** you waste time and money solving the wrong problems
- ▶ If the product doesn't **behave** as expected, you waste time understanding why and changing its behavior

## HOW DO YOU UNDERSTAND PRODUCT EXPECTATIONS

- ▶ You can ask what users want, or they can tell you what they want, but...
- ▶ People aren't good at describing new features
- ▶ They're much better at describing how to change existing software or explaining how a new feature is "like" an existing feature
- ▶ Fortunately, Omnis is good at prototyping and building solutions quickly

## THE SOLUTION TO UNDERSTANDING PRODUCT EXPECTATIONS

- ▶ Ship a basic version of the feature as soon as possible
- ▶ Gather feedback
- ▶ Improve the design
- ▶ Ship the improved product
- ▶ Gather feedback
- ▶ Repeat

## ENSURING THE PRODUCT BEHAVES AS EXPECTED

- ▶ Build it to match expectations
- ▶ Test it to ensure it matches those expectations, ideally every time you change your code
- ▶ Invest in your code to handle boundaries, edge cases, and adopt best practices

## PUTTING IT TOGETHER

- ▶ We needed to ship early and ship often
- ▶ We needed to be able to change our code base without fear of breaking existing features
- ▶ We needed to focus on writing better code and applying new techniques and solutions to existing code
- ▶ These changes would pay off our technical debt and increase quality



**HOW DID WE  
DO IT?**

## SOLUTIONS

- ▶ Automated testing
- ▶ Continuous integration
- ▶ Agile development

## AUTOMATED TESTING

- ▶ Software is about matching behavior to expectations through code
- ▶ Your application codes the behavior
- ▶ The test codes the expectations

## CONTINUOUS INTEGRATION

- ▶ Occurs after committing changes to your code and before shipping a new version of your application
- ▶ Combines all the work that's been done into a release candidate
- ▶ Runs automated tests to ensure everything passes
- ▶ ONLY when all tests are passing is your new release made available
- ▶ New releases are ONLY provided after going through automated testing

## AGILE DEVELOPMENT

- ▶ Organized work in 2 week sprints
- ▶ Focused us on the most important work
- ▶ Retrospectives give a framework for adopting testing and continuous integration
- ▶ Enabled stakeholders to plan around releases
- ▶ A consistent development and release schedule lets us build in time to pay down technical debt

## AUTOMATED TESTING IN OMNIS

- ▶ OmnisTAP
- ▶ Initial built in 2010
- ▶ Open-sourced in September 2017 (for EurOmnis!) and made freely available
- ▶ Brief introduction in this session; the second session will explore OmnisTAP in depth

## CONTINUOUS INTEGRATION WITH OMNIS

- ▶ OmnisCLI allows calling Omnis from the command line and hence used in a continuous integration tool to:
  - ▶ Build the latest changes into a library
  - ▶ Run OmnisTAP on that library
  - ▶ Update databases for testing and deployment
- ▶ We use a customized version of the Omnis VCS...but the JSON export is MUCH better 😊



WHAT ABOUT

---

DEVOPS?



## DEVELOPMENT + OPERATIONS = DEVOPS

- ▶ Automated testing, continuous integration, and agile development started gaining traction in the early-mid 2000's
- ▶ However, the proliferation of the Internet and cloud services meant companies needed to be able to deploy changes rapidly to stay competitive
- ▶ Traditional models of one team developing an application, another running QA on it, and yet another deploying it became a constraint



## DEVOPS IS A PHILOSOPHY

- ▶ Developers needed to be involved in solving deployment problem to get the best fixes as quickly as possible
- ▶ Cloud infrastructure needs code to manage it, and the applications running on them needed to be designed to scale with demand
- ▶ Operations and development needed to merge because each had a stake in the other succeeding

## WHAT DEVOPS HAS BECOME

- ▶ A buzzword
- ▶ A great way to get people to look at your LinkedIn profile
- ▶ A focus on tools that promise to solve all your problems overnight

**WE DO DEVOPS ALREADY**



**JUST WITHOUT AGILE,  
STANDUPS, CI, CODE OR ANY OF  
THAT OTHER HIPSTER STUFF**

memes.com

## DEVOPS AND OMNIS

- ▶ Omnis excels at letting a small team or single developer build usable solutions
- ▶ Omnis isn't likely being developed by one team, tested by another, and deployed by a third
- ▶ DevOps came from:
  - ▶ Operations needing to adopt existing best practices in the development world
  - ▶ Development needing to care about how the app was deployed and working
- ▶ I suspect everyone here cares about how well their app works

## OMNIS IS CATCHING UP TO THE CURVE

- ▶ The rest of our industry adopted agile practices, automated testing, and continuous integration years ago
- ▶ Omnis missed this boat, partially because the tools didn't exist
- ▶ Omnis is already good for rapidly developing solutions (very DevOps) and simultaneously testing your code in a user environment (very, very DevOps)
- ▶ But eventually, you'll fall deep into technical debt like we did without automated testing and continuous integration

**HOW DO WE “DEVOPS”  
WITH OMNIS STUDIO?**



## THE SOLUTION

- ▶ Add automated testing and continuous integration to the rapid development world of Omnis
- ▶ Improve the quality of your application for fun and profit
- ▶ Start focusing on being a better coder and building great software
- ▶ And enjoy a world of confidence and continuous improvement

**DEMO**