A Smarter, Faster Way to Build Embedded Touch Devices and Empower Your Team

GreenHouse
by ICS

Table of Contents

INTRODUCTION

A New Approach to Product Development

Successful touch devices appear deceptively simple to the user. In reality, they are highly complex and challenging to build because they require so many different skills. Each device requires custom hardware, touchscreen, operating system and web connectivity, as well as custom software. Getting it right is the difference between winning or losing before you ever get to market.

That's where *GreenHouse by ICS* comes in. GreenHouse is a smarter way to build embedded devices. In fact, it's a sophisticated solution to a complex problem. And using it for your development projects benefits not only the project overall, but every member of your team. In the next four chapters, we'll explain how.

CHAPTER 1

Less Complexity Means Better Results for Your Team

Built on the popular Qt framework, GreenHouse eliminates much of the complexity associated with product development — delivering better results in substantially less time. How? We provide the elements that are common to all devices, allowing you to focus sharply on the things that make your device unique.

Your team has the freedom to concentrate on creating features your customers most want without getting bogged down with software infrastructure.

GreenHouse Overview

Auto generates a touch UI with many common features

Creates pixel-perfect UI that flows directly from UX tools

Provides early working prototype for market testing and stakeholder feedback

Includes built-in device simulator for software and usability testing on target or non-target hardware

Defines a clear API where device control and business logic can be injected, reducing project risk

Voice interface and mobile interface can be added easily

With each sprint, the working prototype evolves into finished product

Projects Move Swiftly from Concept to Working Prototype with Minimal Risk

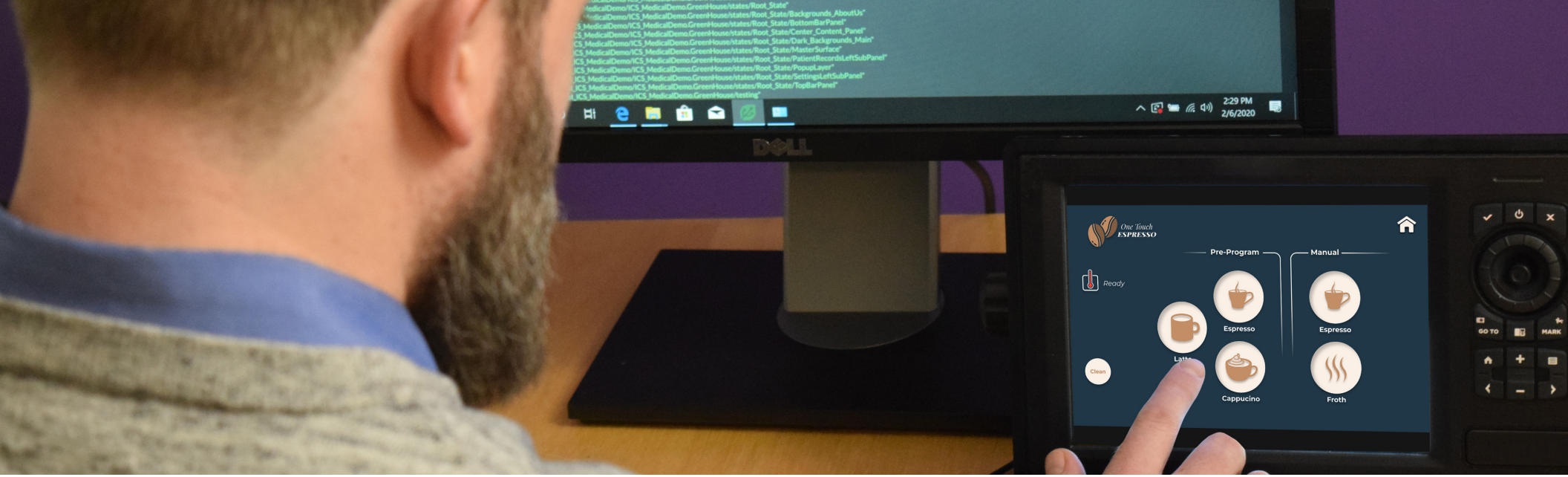
GreenHouse creates a pixel-perfect user interface (UI) by flowing the UI directly from best-in-class user experience (UX) design tools like Figma. Initially, your device is simulated using tooling from GreenHouse, and enhanced over time with custom plug-ins via agile and test-driven development.

The prototype that works on your desktop moves seamlessly to your device.

The result is profound: the prototype that works on your desktop moves seamlessly to your device. With each sprint, your working prototype running on target hardware evolves into a high-quality finished product.

Built on Standards

Your completed application is source code that will work on a wide range of platforms, including Intel and ARM, and it will run on most operating systems (Windows, Linux, QNX). It's based upon open standards, such as C++, Qt and the Qt development tools. And it works with popular CI tools like Git and many others.



GreenHouse forges close collaboration among product owners, UX designers, developers, and QA experts.

GreenHouse Benefits Your Entire Team

GreenHouse supports a highly productive iterative, end-to-end development process. How? It forges close collaboration among product owners, UX designers, software engineers and QA experts while allowing product development team members to focus on their individual areas of expertise.

With the freedom to concentrate on creating features your customers most want without getting bogged down with software infrastructure, **your team can quickly deliver a superior product.** Read more about how GreenHouse benefits specific team members, including your product owner, lead developer and UX designer, in the following chapters.

CHAPTER 2

Early Working Prototypes Help **Product Owners** Create Success

Product owners are responsible for identifying market needs, writing the business plan, convincing senior executives to invest in the product, managing the budget and timeline, and making hard tradeoffs when needed. As such, they're constantly worried about whether the product will be accepted in the market.

Market acceptance is such a big concern for a product owner because the potential for failure is so high. The Harvard Business School estimates that more than 30,000 new products are introduced annually, and 95% fail.

Ninety-five percent!

So product owners are right to worry. Think about it. They're typically asking shareholders for an investment of $10 million, $20 million or even more to develop a product with a miniscule chance of winning. That's why it's up to the product owner to do everything (legally) possible to swing the odds in his or her company's favor.

And that's where GreenHouse by ICS comes in.

GreenHouse is an ambitious undertaking that leverages our experience and knowledge to create a rapid development environment for embedded touch devices. In addition to helping address key concerns, including time to market, cost of bill of materials, product quality, etc.

GreenHouse offers two things guaranteed to make a product owner cheer:

- Early **availability of a working prototype** that is continuously extended through the development cycle, providing for in-parallel market testing
- The ability to **adjust the user experience (UX) at minimal cost even late in the development cycle,** allowing owners to react to competition or take advantage of new market opportunities

GreenHouse works by allowing UX designers to create using their usual workflow. Designers then feed their work directly into GreenHouse, which in turn generates the application structure that implements this UX. Through the use of a rules engine, environmental or user input can cause actions to be taken. Initially, these actions are just "stubs" and do nothing. However, with each development sprint they evolve until the whole application comes into existence.

The important point here is that very early in the product life cycle, a working UX can be shown and discussed with critical customers to ensure market acceptance. And this process of soliciting market input can occur throughout the product development cycle.

At no time is the product owner without a working demo that can be shown to critical customers.

No More Throwaway Prototypes

The value of prototypes when gathering market information is unmatched as watching a presentation or reviewing a document does not have the same impact as seeing the burgeoning product in action. **Traditionally, the problem has been that the prototype is only created once.** As the product evolves, it evolves further and further from the prototype until the prototype is no longer of value in gathering market input.

This means that in the typical product development cycle, there is a dangerous "blind period" where the prototype is no longer relevant yet engineering can't provide the product owner with a demonstration version to use to gather market input.

But GreenHouse changes this paradigm.

GreenHouse not only helps to create the prototype, but it actually becomes the real product though continual refinement. That means at no time is the product owner without a working demo that can be shown to critical customers.

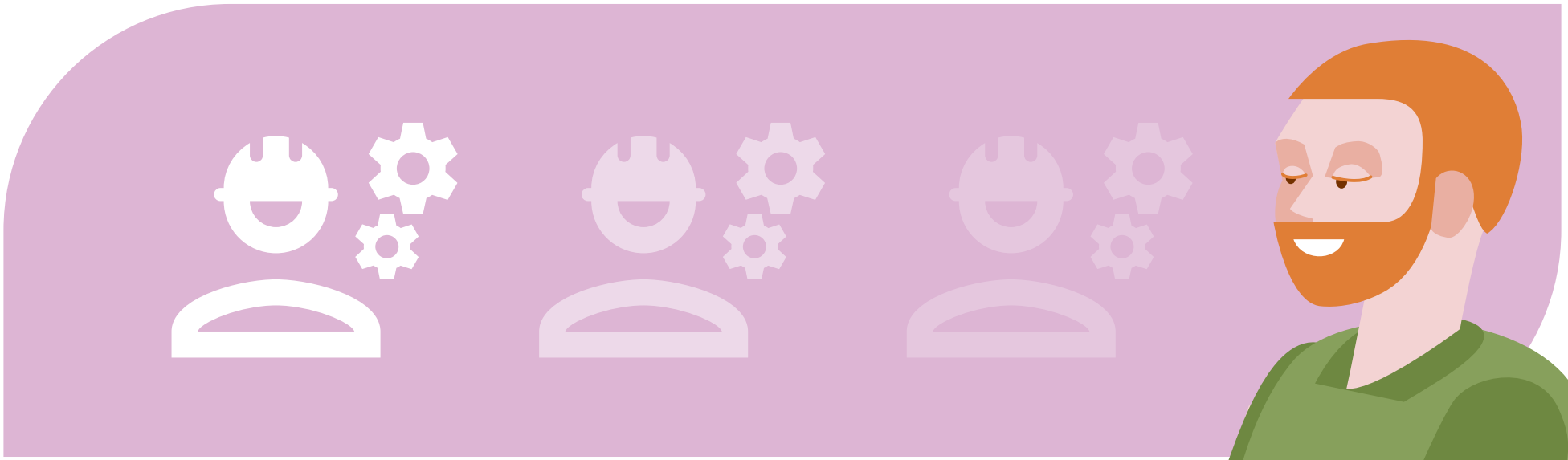
By providing a prototype early, one that continues to evolve into the actual product, **GreenHouse eliminates the blind period.** This allows market adjustments and corrections to occur early and continuously throughout development, giving product owners the ability to evolve as business realities change.

Late Changes are Easily Accommodated

GreenHouse also provides product owners with another invaluable benefit: the ability to make late-stage changes without busting the budget. This allows owners the flexibility to respond to competition or market opportunities for minimal cost. This is possible because with GreenHouse, the UX is architecturally isolated from the application. The implementation of either user touch actions or external stimulus (e.g. alarm sensors, etc.) is effectively a plugin.

And it's all driven by a rules engine. So revising or adding a feature, even late in the game, is a matter of changing the UX (if necessary), altering some rules, adding or modifying the business plugins, and then regenerating the application. The architecture generated by GreenHouse ensures that modifying the application, even at the eleventh hour, is a clean process.

Taken together, using GreenHouse dramatically improves the odds of product development success.



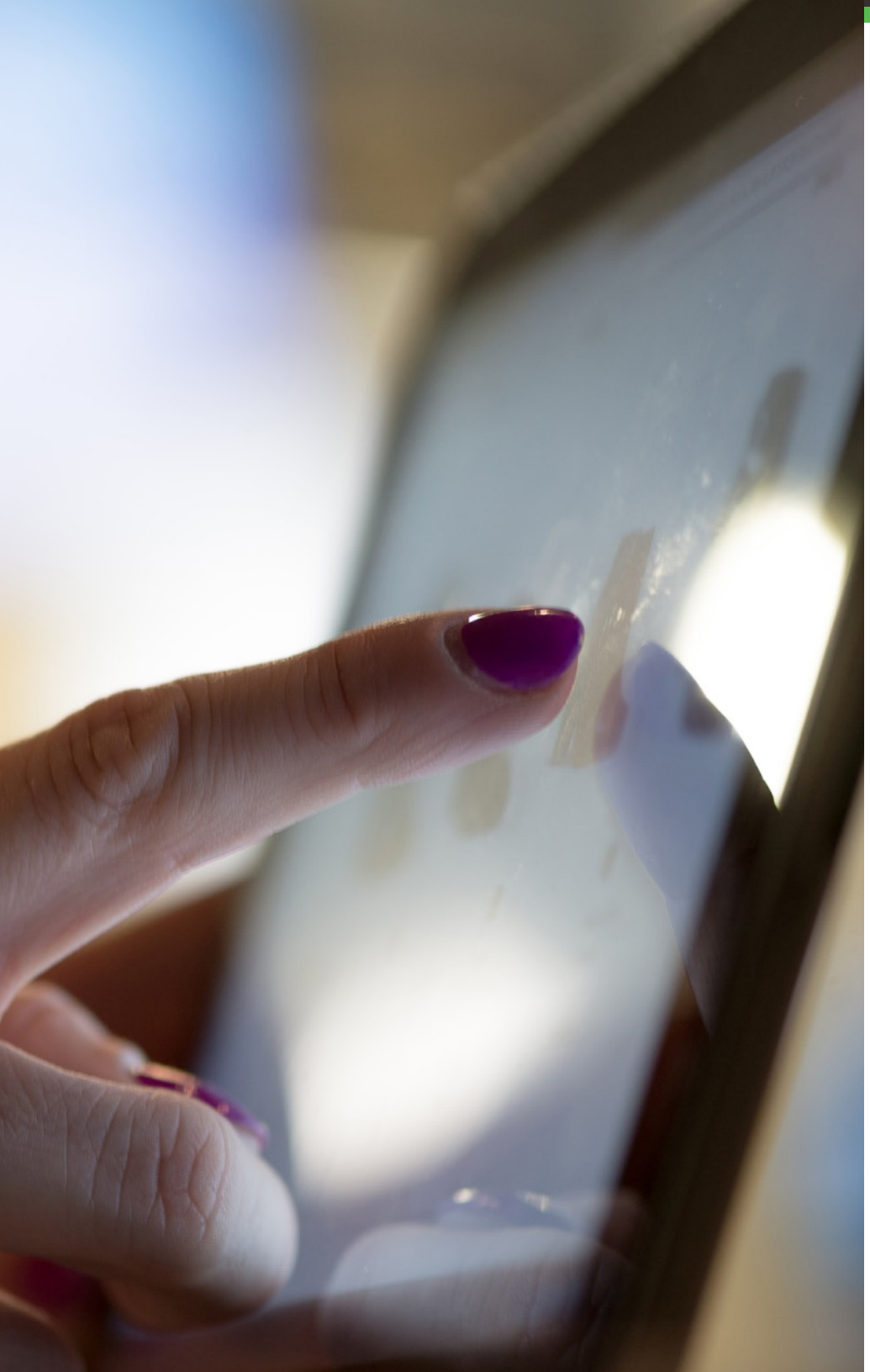
CHAPTER 3

Helping Software Engineers Turn UX into Reality

Building a new device challenges the skills of a software engineering team to the extreme. Although code reuse has been the motto of software development for years, in reality very little code is re-used (aside from personal "tricks").

In fact, most code isn't written to facilitate reuse because of pressures from deadlines and hidden dependencies.

And, in the cases where the code is incorporated, it often threatens the success of the project by forcing the use of old concepts and discarded approaches that may not be appropriate in a new device. This is something developers know all too well.



If you ask a developer to itemize what makes his or her job difficult, three common themes will emerge:

- **Translating** the vision of the UX team into the reality of the software and hardware environment
- **Gaining consensus** within the engineering team on an architecture that makes sense today and will work through future generations
- **Testing the software** when the hardware is late or has limited availability

At Integrated Computer Solutions (ICS), we develop more than 50 high-impact devices annually. We know that projects can succeed only if these types of concerns are effectively addressed.

For that reason, our team of engineers devised GreenHouse to solve these problems and smooth the way for faster product development.

Translating the UX Vision into Reality

Around ICS, we recall a story that illustrates reality colliding with the UX. Around a dozen years ago, we worked on a project for a major appliance manufacturer that had created the UX portion in house. Their vision: while the appliance was operating, an animation would appear on the touchscreen to indicate to users that the wash cycle was active. The designer who created this was understandably proud of his innovative UX, which worked perfectly in "flash" on a quad core Intel processor packed with lots of memory.

Unfortunately, that wasn't the platform the actual machine was being built on.

Sadly, we had to inform the designer, who had expended great effort and time on the nifty animation, that the processor they had chosen lacked a GPU capable of making that level of animation possible. Needless to say, people did not leave the meeting with smiles on their faces.

To ensure this type of problem would never again arise, we took a **UX-centric approach with GreenHouse by putting the UX designer in charge of the UX from the beginning** of the project.

Designers can create using their preferred tools (e.g. Figma, Qt Design Studio, etc.) and then, through "bridge" software supplied by GreenHouse, import it into the development environments that engineers are using. Almost immediately, the design is concretely implemented — if available, even on the target hardware — and the UX engineer can confirm that the design is capable of meeting the reality of the device's display and processor.

GreenHouse eliminates this battle by delivering a four-layer architecture, driven by a sophisticated state machine, that has proven itself repeatedly in all the devices that ICS delivers.

Architecture for Today and Tomorrow

Some developers love to work on the architecture of a software application, tweaking and re-writing until it is perfect. Others believe in the "good enough" principle and want to get on with the tasks of developing the actual software. The battle between these positions (and perhaps other subtle variations) makes it difficult for a typical manager to make the best decision.

GreenHouse eliminates this battle by delivering a four-layer architecture, driven by a sophisticated state machine, that has proven itself repeatedly in all the devices that ICS delivers. Furthermore, it isolates changes because the layers are loosely coupled. This makes it relatively simple to address even last-minute changes — you know, the ones that always seem to be "must haves" but arrive just before the team wraps the project up.

For the same reason, it's future proof. By expanding the state machine or modifying selected states, v2, v3 and future versions can be delivered. GreenHouse also provides a number of large-grain components, such as Wi-Fi and bluetooth, that can be immediately dropped into your application with minor tailoring.

Hardware is Unavailable. What Now?

One of the very frustrating aspects of developing software for a new device is that often the hardware is late, or because of cost and size, only available in limited quantities (and perhaps in limited locations). When this happens, it can be a disaster for the productivity of the software team.

Often, key members can't do anything more on the project and are "loaned" out to other projects never to return. If they do return, they'll take days and weeks to return to the point of productivity where they left the project.

Believe me, it happens all the time.

GreenHouse was specifically designed to address this issue by building a device simulator into the application. When the application needs data from the device, the developer can provide it through the simulator workbench. This data also can be saved and "replayed" later to test for regressions.

The simulator console doesn't even have to be on the device itself. Through a built in RPC mechanism, the simulator console can be on your desktop while the application is running on the target hardware. This makes testing and debugging much more straightforward on devices that only have a touchscreen and might not have the facility to enter data to trigger states or explore the state of the machine.

Even after the hardware becomes available, the simulator console is still very useful in triggering error conditions that might be difficult — even impossible without doing damage to the machine — and seeing how the software responds. In other words, GreenHouse's simulation console is one of its most powerful features and well worth exploring.

No Hardware, No Problem

The concept of GreenHouse originated in the challenges we continuously saw each year as we developed devices for our customers. We frequently had to be the messenger of bad news when the UX outran the capability of the hardware. And our engineering culture of getting things done now didn't allow us the luxury of debating which architecture was best. **So we developed an architecture that works well in 99% of applications.**

GreenHouse solves the issue of code reuse by generating all of the boilerplate interface code and ensuring all dependencies are properly oriented. This means implementations are truly reusable.

Regarding hardware, prior to the development of GreenHouse, if we couldn't get the hardware, our engineers couldn't work. And, if they don't work, we don't get paid. There goes the business plan.

With GreenHouse, the simulator provides an elegant solution that benefits our customers and allows our team to move ahead even when hardware availability is an issue.

CHAPTER 4

Empowering UX Designers to Focus on User Needs

UX designers work on a wide variety of projects, ranging from web and mobile apps to desktop and embedded devices to name a few. No matter the application, there is a common thread of persistent obstacles that designers on integrated teams encounter most — if not all — of the time.

As the team gains knowledge through research and user testing, designers come up with new ideas. But at some point, innovation starts to bump against the constraints of budget and scope.

In addition, there are delicate transfer points between the UX design and engineering that have to be well-handled and documented, and one set of control artifacts (e.g. information architecture, wireframe or prototype) often gets replaced by another. Maintenance of previous control artifacts is tedious so it often falls by the wayside, mucking up definition and allowing for misinterpretation and mistakes.

For those reasons, UX designers need workable solutions to avoid the potential for:

- **Miscommunication** between the design team and the software development team during hand-off, which can set a project back days or weeks
- **UX deliverables that are not updated** downstream and disposable prototypes that lose relevance in the iterative design process because the code isn't reusable
- **Expending resources less than optimally** because designers have no way of knowing whether their ideas can be implemented within the client's budget until it is too far into the product development timeline

Our own pain in constantly navigating these obstacles has driven us to devise a solution that allows UX designers to focus even more closely on user needs while still accommodating business goals. GreenHouse bulldozes obstacles to project success and allows UX designers to have a greater influence throughout the life of a project.

No More Disposable Prototypes

UX designers create using their usual workflow and then feed it directly into GreenHouse, which in turn generates the application structure that implements the UX. Through the use of a rules engine, environmental or user input can cause actions to be taken. With each development sprint the functional prototype evolves — meaning the code stays current with the design — until the whole application comes into existence.

That means no more disposable prototypes. This is significant because GreenHouse allows you to avoid the "blind period" — the portion of the development lifecycle when the prototype is no longer relevant yet a demonstration version to use to gather market input and conduct user testing is not available — that is a hallmark of traditional product development processes.

GreenHouse helps create the prototype, which actually becomes the real product though continual refinement.

GreenHouse provides project transparency so all stakeholders can see exactly what is being built. There's no risk that the design effort gets in front of the engineering effort — no need to worry that the designers will create something that can't be built — because the UX and the UI are the same development effort. There is not a separate set of UX deliverables that get tossed along the way.

GreenHouse affords a more active and dynamic collaboration with engineering throughout the life of the project, in design and especially in development, which leads to tighter QA and ultimately a better product.

Benefits to Designers

GreenHouse affords designers a number of additional benefits, including greater control over designs and the opportunity to integrate more deeply into other phases of the project. Additional benefits include:

- **Closer collaboration** with developers
- Ability to select from an ever-growing **library of UX interactions** as code-ready components. (Using reusable code accelerates creation of prototypes.)
- **Easy access to documentation** that evolves organically with the design and development environment
- **More integrated QA processes** and stronger support for usability testing and validation

Using GreenHouse means the UX designs are automatically converted into code. It is the designer who is driving and QA-ing this work as it happens. There is little if anything lost in translation between designer and developer because, in fact, there's less translation. The process is automated.

"By using GreenHouse you don't lose control over your design when it travels downstream into engineering," explained Boris Savic, Associate Director of User Experience at Boston UX. Rather, GreenHouse affords a more active and dynamic collaboration with engineering throughout the life of the project, in design and especially in development, which leads to tighter QA and ultimately a better product.

"It also provides for greater repeatability and reusable code for prototypes, which allows me to focus on communicating ideas," Savic said. "UX is an iterative process, and many of our best ideas come at a point when in a traditional environment it would be too late to integrate them. GreenHouse extends our ability to keep doing that without putting undue pressure on the engineers or creating overages."

Working with GreenHouse

Designers use Axure for wireframes and prototypes, and Figma for designs and design components, and then use GreenHouse's Bridge tool to directly import the Figma designs. (Since Figma can import from Photoshop, Sketch and other tools, the GreenHouse Bridge effectively spans the most popular design tools.)

Once the design is imported, the UX designer (or, in some cases the software engineer) can use GreenHouse Architect to finetune the design for the chosen platform, for instance tweaking color or fonts, as well as use GreenHouse's rules engine to determine the flow of the application.

At this point, GreenHouse generates the application's framework.

With GreenHouse, there's no opportunity for confusion or miscommunication between designer and developer. There's no blind spot — the point in traditional development when the UX lead and product owner know what's been designed but have no clue on what's actually been built. The functional prototype GreenHouse creates is perfectly suited for usability testing and stakeholder buy-in.

GreenHouse empowers designers by streamlining the process with and strengthening communication between UX and engineering — both essential to the success of any development project.

About ICS

Integrated Computer Solutions (ICS) delivers excellence in both user experience (UX) design and custom user interface (UI) software development for IoT and embedded devices, and companion mobile and desktop applications.

If your product is driven by touchscreen or voice, it's in our wheelhouse. We can design the intuitive UX and the UI software, as well as integrate with your platform. That means we help you build better products, with lower risk and cost. We've worked on everything from sensitive medical devices to business-critical industrial equipment to automotive digital cockpits. And we've helped some of the world's most-renowned brands, including Abbott, Boeing, GE, Intel, MilliporeSigma and Thermo Fisher.

We develop on a variety of platforms, including Qt/QML, HTML5, QNX, Linux, Android, iOS and are the largest independent source of Qt expertise in the U.S. Our successful track record stretches back to 1987.

Author Bios

Mark Hatch

Chief Operating Officer

Mark, who joined ICS in 1995, oversees the company's Qt business. He has more than 30 years experience managing the development and business aspects of software tools. Previously, Mark was VP of marketing for a networking middleware company. Before that, he was manager of software marketing for Apollo Computers. He helped form key industry groups, including the X Consortium and the Open Software Foundation. Mark has a Masters in electrical engineering and computer science from the University of California, Berkeley, and an MBA from Boston University.

Stephanie Van Ness

Associate Director of Marketing Communications & Chief Storyteller

Head of content development, Stephanie is an experienced copywriter who covers software development, user experience design and innovations in technology, from self-driving vehicles to gesture-controlled medical devices. Her work has appeared in a variety of industry publications, includ-ing *Medical Design & Outsourcing, Mass Device, Connected World, Medical Device + Diagnostics, UX Collective* and *Prototypr.* She has a B.S. in journalism and biology from Boston University.

Krzysztof Krzewniak

Senior Software Architect

Krzysztof has extensive experience developing complex solutions for global companies, including Tier 1 automakers and avionics firms, as well as deep user interface knowledge and implementation skills using Qt Quick. He also developed ICS' rapid-development solution called GreenHouse. He holds an MSc in software engineering and IT systems from Poland's Czestochowa University of Technology.


GreenHouse
by ICS