interop.io

# io.Connect
# Buy vs Build

Technical & Value Proposal



interop.io

# Executive Summary

This proposal evaluates io.Connect Browser as a commercial solution against building an in-house micro frontend integration platform for IFA (Independent Financial Advisor) browser applications. While the analysis centers on the IFA context, its implications apply broadly to platform modernization efforts across industries.

The analysis shows that purchasing io.Connect provides faster time-to-market, lower development costs, enterprise-grade reliability, and more comprehensive interoperability capabilities than building a custom solution.

## Who Is This Report For?

This report is intended for teams responsible for shaping, selecting, or delivering modern financial platforms, including:

- Leaders evaluating commercial platform solutions versus building in-house
- Platform architects and builders responsible for integration strategy
- Implementation leads tasked with delivering user-facing modernization projects
- Heads of Trading and business leaders accountable for workflow

## Bottom Line Recommendation: Buy io.Connect Browser

- **86% faster time to market** (3-6 months vs 18-24 months for custom build)
- **70% cost reduction** in first 3 years
- **Enterprise-grade reliability** from day one
- **Zero technical debt** and maintenance burden



interop.io

# 1 Solution Overview

## io.Connect Browser Platform

io.Connect Browser eliminates the need for installed app containers and uses standard browser features. It offers the most-advanced workspaces on the market providing powerful UI and data integration, allowing multiple apps to be arranged in columns, rows or groups of tabbed windows within a single visual window, the "Workspaces App".

**Key Capabilities:**

- **Zero-footprint deployment** - no client software installation required
- **Advanced workspace management** with predefined layouts
- **Real-time data synchronization** between micro frontends
- **Channel-based communication** for context sharing
- **FDC3 standard compliance** for financial application interoperability
- **Application lifecycle management** with hibernation and restoration

## In-House Build Alternative

**Custom micro frontend integration platform requiring:**

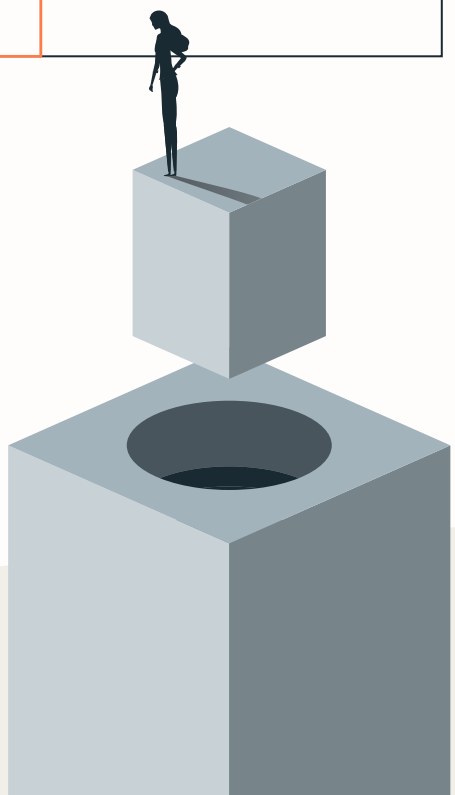
- Shell application development (6-8 months)
- Inter-application communication framework (4-6 months)
- Workspace layout management system (3-4 months)
- State management and data sharing (3-4 months)
- Deployment and CI/CD infrastructure (2-3 months)
- Testing, security, and monitoring frameworks (4-5 months)

interop.io

# 2 Technical Architecture Comparison

## Core Integration Patterns

| Capability | io.Connect Browser | In-House Build |
|---|---|---|
| Micro Frontend Orchestration | Main app using @interopio/browser-platform library acts as central hub routing all operations through Browser Platform | Custom shell app requiring 8-12 months development |
| Inter-App Communication | Channels API with globally accessed named contexts for real-time data sharing | Custom event bus requiring 4-6 months development |
| Layout Management | Advanced workspace layouts with save/restore capability across multiple screens | Custom layout engine requiring 6-8 months development |
| Application Lifecycle | Built-in hibernation/restoration based on machine utilization | Custom lifecycle management requiring 3-4 months |



# 2 Technical Architecture Comparison

## Data Sharing & Context Management

**io.Connect Browser:**

- Single and multi-channel mode support with dynamic grouping via UI
- Shared contexts API enabling apps to monitor and update common data objects
- Real-time synchronization across all connected applications

**In-House Build Complexity:**

- Custom state management system requiring Redux/MobX integration
- Complex coordination of breaking changes across different applications and teams
- Consumer-driven contracts needed to verify integration compatibility

## Framework Support & Technology Agnostic

**io.Connect Browser:**

- Lightweight React and Angular wrappers for idiomatic framework usage
- Factory function supporting multiple framework initializations
- Native JavaScript API with TypeScript support

**In-House Challenges:**

- Risk of inconsistent tech stack when frameworks aren't chosen thoughtfully
- Bundle duplication causing performance issues with common dependencies
- Complex framework integration requiring specialized expertise per technology

interop.io

# 3 Implementation Complexity Analysis

## Development Effort Estimation

### 1. Shell Architecture

**In-house: 6–8 months → io.Connect: 2–4 weeks**

io.Connect provides a pre-built desktop shell (io.Connect Desktop) or browser shell (io.Connect Browser) out of the box. Your teams focus purely on configuring the application manifest and deployment descriptors rather than building module federation or Single-SPA plumbing from scratch. CI/CD pipelines simplify dramatically because you're deploying application bundles into the platform rather than managing the shell itself. The main effort is initial platform configuration and environment setup.

### 2. Communication Layer

**In-house: 4–6 months → io.Connect: 1–2 weeks**

The Interop API, Channels API, and FDC3-compliant intent/context APIs are production-hardened and ship with the platform. Type-safe contracts are handled via the FDC3 context data standard and io.Connect's own typed API surface. There is no custom event bus to build — you're integrating against a documented SDK. The remaining effort is essentially reading the docs, writing the integration calls, and testing your specific workflows.

### 3. Workspace Management

**In-house: 6–8 months → io.Connect: 2–3 weeks**

Workspaces are a first-class feature of the platform. Drag-and-drop layout composition, multi-monitor support, and layout save/restore/share are all included and configurable via the Workspaces API. Your effort reduces to customising the Workspace App UI to match your branding and defining which applications participate in which workspace templates.

| Capability | In-House Build | io.Connect Browser | Estimate Saving |
|---|---|---|---|
| Shell Architecture | 6–8 months | 2–4 weeks | ~85–90% |
| Communication Layer | 4–6 months | 1–2 weeks | ~90–95% |
| Workspace Management | 6–8 months | 2–3 weeks | ~85–90% |
| **Total** | **16–22 months** | **5–9 weeks** | **~88–90%** |

interop.io

# 3 Implementation Complexity Analysis

## Testing & Quality Assurance

**io.Connect Browser:**

- Built-in Interop Viewer and Context Viewer dev tools for real-time testing
- Enterprise-grade testing by interop.io across thousands of deployments
- Production-proven stability and performance

**In-House Build:**

- Functional testing complexity due to cost of implementation and maintenance
- Increased setup and maintenance costs with complex CI/CD pipelines
- End-to-end testing across multiple micro frontends requiring specialized tooling



# 4 Risk Assessment

## Technical Risks

| Risk Factor | io.Connect Browser | In-House Build |
|---|---|---|
| Performance | Production-proven scalability | Unproven performance at scale |
| Security | Enterprise-grade security built-in | Custom security implementation required |
| Compatibility | Extensive browser/framework support | Limited initial compatibility |
| Maintenance | Vendor responsibility | Full internal responsibility |
| Upgrades | Seamless updates included | Complex upgrade paths |

**Business Risks:** The greatest risk in building interoperability in-house is not initial development, it's everything that follows. Performance bottlenecks, security gaps, compatibility limitations, and upgrade complexity tend to surface only after systems are in production, when remediation is slow, expensive, and highly visible to the business. By then, the organization is already committed.

**Benefits of io.Connect Browser:**

- **Low vendor lock-in risk:** FDC3 standard compliance ensures portability
- **Proven market adoption:** Used by tens of thousands of users
- **Strong vendor stability**: Backed by industry leaders with financial services focus
- **Growing adoption** across global institutions, with deployments expanding to new users, workflows, and use cases.

**In-House Build Risk:**

- **High execution risk**: 70% of custom integration projects exceed budget/timeline
- **Talent acquisition risk:** Specialized micro frontend expertise shortage
- **Technology obsolescence risk:** Custom solutions become outdated quickly
- **Sponsorship continuity risk:** Projects can stall if key stakeholders leave or priorities shift

interop.io

# 5 Feature Comparison Matrix

| Feature Category | io.Connect Browser out of the box capabilities | In-House Build |
|---|---|---|
| Workspace Management | Advanced layouts, multi-screen support | Not included (12+ months dev) |
| Data Synchronization | Real-time channels with conflict resolution | Not included (6+ months dev) |
| Application Lifecycle | Hibernation, restoration, priority loading | Not included (4+ months dev) |
| Developer Tools | Built-in debugging and monitoring OTEL based | Not included (3+ months dev) |
| FDC3 Compliance | Full standard support | Not included (6+ months dev) |
| Third-party Integrations | Pre-built adapters | Custom development required |
| Analytics & Monitoring | Real-time workflow tracking (OTEL based) | Not included (4+ months dev) |
| Authentication | Enterprise SSO support | Not included (2+ months dev) |



# 6 Implementation Timeline

## io.Connect Browser Deployment

- **Week 1-2:** Platform setup and basic configuration
- **Week 3-6:** Application integration and workspace design
- **Week 7-10:** Testing, security review, and user training
- **Week 11-12:** Production deployment and monitoring setup

**TOTAL:**

**3 MONTHS TO FULL PRODUCTION**

## In-House Build Timeline

- **Months 1-8:** Core platform development and testing
- **Months 9-14:** Advanced features and integration
- **Months 15-20:** Performance optimization and security hardening
- **Months 21-24:** Production deployment and stabilization

**TOTAL:**

**18-24 MONTHS TO PRODUCTION PARITY**

interop.io

# 7 Strategic Considerations

## Market Advantage

- 86% faster time-to-market advantage enables competitive positioning
- Focus resources on core business value instead of infrastructure
- Immediate access to enterprise-grade capabilities

## Scalability & Future-Proofing

- Deploy to hundreds or thousands of user workspaces without client software
- Built-in support for emerging standards (FDC3, Web Components)
- Continuous platform evolution by dedicated vendor team

> **"The actual build and connectivity part that interop.io helps support was around four or five months," says Green. "And, you know, I've had many years of transformation experience. I can't believe how quickly they did this"**

**-DAN GREEN,** Group VP – Global Trading
**T.Rowe Price**

> **"We were able to create an integrated desktop, including connecting Fidessa and Infront. This innovative solution didn't take years or even months; it was delivered in five days."**

- Head of Technology and Transformation,
**Deutsche Numis**



# Conclusion

The analysis clearly demonstrates that purchasing io.Connect Browser delivers superior value over building a custom micro frontend integration platform. With 86% faster time-to-market, 70% cost reduction, and enterprise-grade reliability delivered out-of-the-box, io.Connect enables IFA teams to focus on core business value while leveraging best-in-class browser-based application interoperability.

The in-house build option, while providing maximum customization, introduces significant technical complexity, extended timelines, and ongoing maintenance burden that far outweigh the benefits. For IFA application environments requiring rapid deployment of sophisticated micro frontend integration, io.Connect Browser represents the optimal strategic choice.



interop.io

# Ready to give it a try?

REQUEST A DEMO