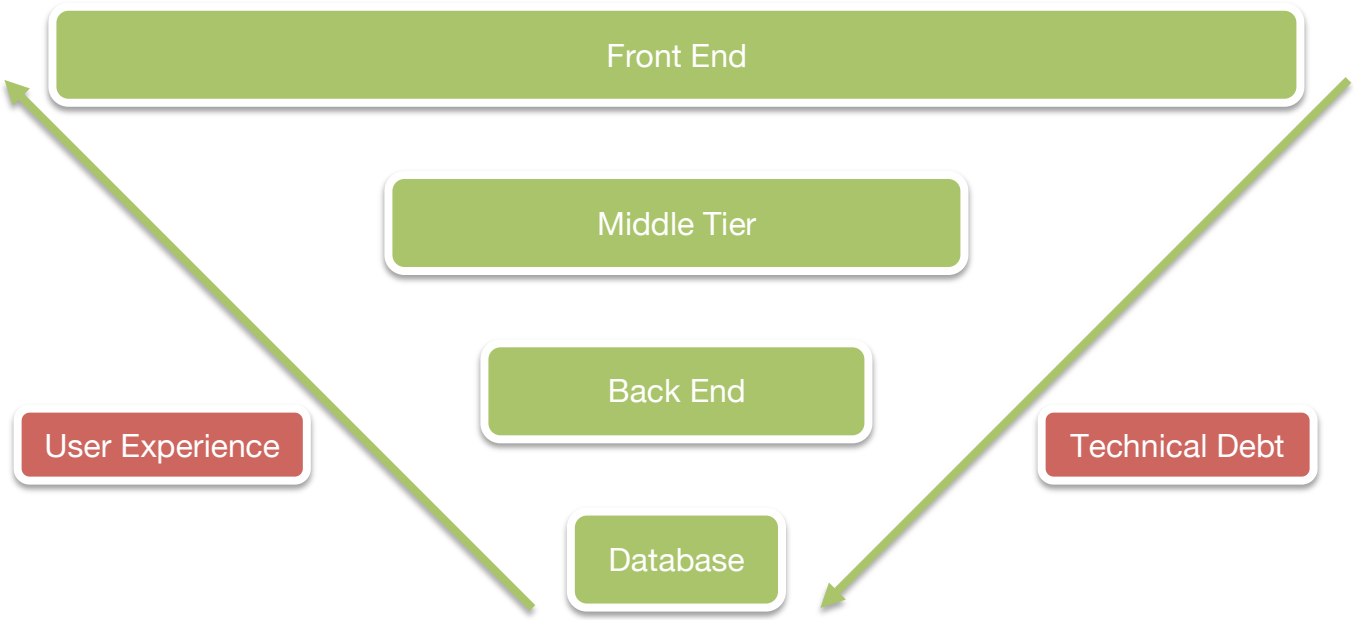
EQengineered



Rationalizing the Tension Between User Experience & Technology

A Front End Enterprise Application Green Paper for CMOs and CIOs
by Julian Flaks and Mark Hewitt

P.O. Box 211 Hampton Falls, NH 03844 | info@eqengineered.com | eqengineered.com | 857.770.1019

EQengineered

Green Paper versus White Paper

The term white paper originated with the British government, and many point to the Churchill White Paper of 1922 as the earliest well-known example under this name.

White papers are a way the government can present policy preferences before it introduces legislation. Publishing a white paper tests public opinion on controversial policy issues and helps the government gauge its probable impact.

By contrast, green papers, which are issued much more frequently, are more open-ended. Also known as consultation documents, green papers may merely propose a strategy to implement in the details of other legislation, or they may set out proposals on which the government wishes to obtain public views and opinion.

EQengineered

Table of Contents

EQengineered

Why Read This Green Paper

The demand by your customers to ensure that interactions with your brand are effortless is complicated by the ever changing and accelerating speed of new technology. As this trend continues to evolve, how can companies remain viable?

In this green paper, get more insight into how CMOs and CIOs will embrace collaboration to ensure that their organizations remain competitive in today's customer-driven and technology-led economy.

Key questions addressed include:

- What challenges keep CMOs and CIOs up at night?
- How can user experience be improved to meet the needs of your intended audience, in context?
- What is the current state of front end enterprise technologies and frameworks?
- Why does your organization need to have a "blueprint" for future friendly connected experiences?

EQengineered

Executive Summary

Creating a value system that equates the demand for superior user experience to the requirement for a more future friendly technical architecture is a core challenge for organizations and a matter of survival.

The tension inherent in the relationship between speed, user experience, and technical currency is palpable. Organizations are witnessing conflict between the advocacy by the CMO to meet the needs of an empowered and demanding customer and the urging by the CIO to adhere to a sound technical roadmap to avoid unnecessary friction, rework, and investment.

Key Findings

- Business and technology leadership must be transparent and collaborate “in the sunlight”. Technical direction must permeate and become part of business decision-making.
- User experience should be contextual, performant, and accessible.
- Technical architecture serves to not only deliver the user experience, but to also establish a platform for cost effective development that mitigates risk and technical debt, an often foreseen implication of business strategy.

Recommendations

- Foster open and transparent communication between the CMO and CIO to understand the tradeoffs between pace, user experience, and technical architecture adherence and integrity.
- Design should be holistic to your projects rather than a discrete phase. Anticipate change, reduce iteration friction, solicit continuous customer feedback and embrace design thinking to establish a culture of innovation.
- Develop a technology “blue print” informed by your business strategy and the input of the CMO and CIO that will guide the organization and become a foundational document of architecture, patterns and development processes.
- Create a company-wide digital strategy with an assigned executive owner, clear milestones and meaningful metrics.

Aggressive forward progress is imperative as leading firms are using digital as a differentiator. Maintaining the status quo is a non-option and the time for “watch and learn” is over. Inaction by fast followers and laggards will leave them vulnerable to disruption by more innovative entrants into their markets. The CMO and CIO bound together by shared values, goals, and a clear digital strategy is your competitive advantage.

EQengineered

User Experience

Today's CMO is challenged to meet or exceed customer demands while delivering on the brand promise. The challenges of building for the future and delighting customers are many. At the forefront of this is the user experience of application front ends. User experience challenges include:

- Understanding customers, their motivations, priorities, and goals
- Creating value for customers that is free of friction, contextual, and if possible, automated
- Creating experiences in an ever changing and hard to understand technology landscape
- Staying on the cutting edge of user experience trends, interaction patterns, and modern application interface design

User experience, while focused on usability, information architecture and interaction pattern design across application interfaces is part of a much more holistic tapestry of experiences, the overall customer experience (CX).

As customer experience becomes a stronger differentiator for organizations, the user experience of web applications becomes more important as it often represents the front line when it comes to customer interactions. It is this continuous drive to improve user experience that has led to the shift in technical strategy favoring heavier front end applications. There are several important questions that you must ask to evolve the practice of user experience design:

- How should you approach the problem solving necessary to evolve and improve your user experience design?
- How do you instill a user experience oriented mindset in your technology teams?

- How do you learn more about your customers in order to optimize the user experience of your applications?

Similar to the technology landscape, there are several tools and strategies that can be employed to confront the challenge and answer these questions.

Design Thinking

Design is often thought of as the creative work conceiving the look and feel of interfaces as a discrete phase at the beginning of a project. It is informed by a set of requirements developed by a product manager and has a distinct output that then feeds development. *But how do you adapt to change? How do you evolve your technology teams to think more creatively?*

One answer is to employ design thinking. Design thinking prescribes utilizing creativity and design principles at a business level to influence strategy and innovation. The tools of design thinking are a pencil and paper, white boards, mind mapping applications, customer interviews and rapid prototyping to name a few. At its very core, design thinking is a human centric approach. It is about empathizing with your customer and experimenting to arrive at solutions based on what customers actually want rather than metrics or historical data analysis alone. By empathizing and finding a path that accounts for their emotions in your solution, you can truly connect with your customer.

Approaches to Design Thinking

- **Create a design framework that allows you to fail fast.** Define your methods of customer research, definition, ideation, prototyping/experimentation, testing, and critical thinking to facilitate the needed idea generation, collaboration and problem-solving.
- **Problem-solve through experimentation.** Create experiences - be that low fidelity sketches or models or high fidelity HTML prototypes - that enable you to quickly and aesthetically collaborate and know whether your approach meets the needs of the

customer. Iterate and improve your solution to the point of exhaustion before considering next steps.

- **Utilize patterns, methods and constructs that inspire the team creatively.** Ideation and collaboration across a multi-disciplinary team that is broader than just designers necessitates the identification of new ways to serve and support team communication and collaboration to uncover customer needs, desires and behaviors.

Design Thinking Considerations

- **Define and employ designer tools and skill sets that work for your team and needs.** Be bold and creative in selecting the constructs and approaches used in your design thinking methodology.
- **Insights and data alone will not transform anything.** Converting findings into actionable ideas demands sustained innovation practices and eventually implementation and build competencies.
- **Speed should not trump the journey and process.** Stay centered on your customers' needs, desires and motivations and you will remain on the right track.

By leveraging these approaches you can bring a broader set of tools and strategies to bear when problem solving throughout your organization. Do not be afraid to engage your technology teams throughout this process. By bringing your technology teams into the process of solving customer experience problems, you can ensure that your technical architecture will be developed in context. Exposing your technology teams to the tools and disposition of design thinking will challenge them to think differently about the applications they are developing, considering the broader needs and motivations of the customer and impacts on the business.

EQengineered

Technology

Today's CIO is challenged in ways unforeseen in the past.

Key concerns include:

- Leading the company's digital transformation, not just facilitating it
- Staying customer centric by prioritizing user experience
- Balancing sometimes competing priorities of reducing cost, improving security and analytics and embracing cloud technologies all while eliminating data centers
- Continually moving faster while maximizing flexibility
- Anticipating change and being forward friendly
- All of the above, while on a flat budget prioritizing support over innovation

A Brief History of Modern Web Applications

Front end experiences entered a new age of challenges with the introduction and proliferation of mobile devices. The nature of what defined "user experience" changed as native apps and browser-based interfaces took off. Customers began to demand rich and interactive experiences that, regardless of platform, provided a native-like feel. At the same time, designers grew a more empathic approach to design user journeys in their own right, rather than simply as data entry steps in a system.

As the nature of these interfaces became more complex and the need for rapid iteration increased, the long-used strategy of server side rendering interfaces became inadequate. Where views had been rendered on the server side and merely decorated by client interfaces with dynamic interactions, the client side now took over a lot of the flow of the application.

Not only views, but now validation, routing, data persistence, and core business logic found their way to the front end. This trend continued with the advent of new browser based platform integrations and capabilities that further improved the native feel of applications.

As more functionality was written on the front end it gave rise to new frameworks that defined better patterns, abstracted browser specific differences, and facilitated leveraging community developed software. The front end stack sprouted the rich complexity and infrastructure that had grown up around the server side.

Smooth empowered interactions on the front end have meant a focus on asynchronous patterns, where the greatest complexity lies in how to coordinate responses to delayed requests. Scalability and performance have both driven the same direction on the server side, and users have come to expect capabilities that have big data implications.

The Challenges of the Present

Front end applications provide richer experiences than they ever have across a wide range of devices and browsers. This has led to several outcomes:

- Web applications require serving large assets, to the detriment of page load times
- Front end code complexity has increased substantially, requiring more specialized expertise to maintain and develop
- Interface design trends continue to evolve, often outpacing widespread best practices on how to most robustly implement
- The device landscape continues to evolve requiring entirely new interfaces and interaction paradigms (VR, AR)
- Performance suffers for applications running overtop large data sets
- The rapid evolution of front end frameworks and difficulty in upgrading leads to fragmented application ecosystems
- Business logic has tended to leak into client device code, a problem made worse by there being multiple devices to support

The Future of Front End Experiences

The reality is that the need for customers to access experiences in new ways on more devices is going to increase. We are on the verge of entirely new modes of user experience with the rise in virtual reality, augmented reality, and virtual assistants. These new platforms will demand new interfaces, interaction patterns, and native integrations. The bottom line is that your teams and applications will always need to evolve to keep pace.

Key Technologies & Strategies

You must find ways to mitigate risk, maximize flexibility and maintain productivity. You need to maintain a focus on using the right tool for the right job, separating concerns, and creating intuitive usable abstractions that progressively simplify and enable your application architectures. In pursuit of this, you should:

- Remember the front end's first priority: user experience. Unify business logic implementation across platforms by consolidating it to a middle tier or back end
- Leverage the back end or database for data intensive tasks slimming down the data sent to the front end. Reduce front end data flow and state management, where possible
- Minimize use of view-centric back end services and favor front end agnosticism

Angular & React – Frameworks and Patterns

Modern frameworks such as Angular and React have grown to meet some of these challenges. Both Angular and React embrace the idea of dividing your front end interfaces into components. This results in cleanly isolated markup, styling, and interaction logic. In addition, components can also offer a vehicle for more organic state flow throughout your application in ways more relevant to the interface being presented. In some cases, where more complicated state management is required, libraries like Redux can be leveraged, but this should be the exception rather than the rule. Components cleanly separate the concepts in your interface more clearly leading to a maintainable and extensible front end application.

The fit will be different for different organizations. Angular provides a larger framework to build with, and React begins with a smaller building block to build upon more freely. The most essential aspect is that both present patterns which should be adhered to where possible for a clean extensible platform. Deviating will be necessary at times, but should be done with great discipline and with appreciation for the technical impact.

Componentized CSS / Sass / LESS

Whether using straight CSS or a preprocessor, styles need to be organized as components. CSS has constraints, often arising from what is efficient for the browser to interpret and apply. The cascading nature often causes applications to become tightly coupled and hard to unravel in their styling. When not properly isolated or namespaced, an application's CSS can become brittle and inflexible to change.

CSS preprocessors have become extremely popular to help proper organization of styles and to re-use design elements simply. The reduced constraints compared to CSS can actually tend towards large chaotic rule sets which become hard to prune and organize. Establishing a direction and strategy for organizing styles will pay off greatly.

Minimal Semantic HTML

In either case, the goal should be to write the absolute minimum amount of HTML necessary to serve the needs of an interface, and ideally in components. That is often best served by leveraging semantic HTML, or HTML that represents expressively the data or interface that it is presenting. This serves to not only make styling simpler and more intuitive but aids in future development by improving maintainability and extensibility.

Modern JavaScript and TypeScript

While functional and flexible, JavaScript has previously been a fairly anemic language feature wise. It has lacked the classical object-oriented programming paradigms and type checking that other back end languages have afforded. The good news is that the newer JavaScript standards that support these features, while not yet completely supported by all browsers,

are accessible by leveraging transpiling via libraries like Babel or superset languages like TypeScript.

By leveraging tools like these to write the most modern front end applications possible, you can mitigate the risk of runtime errors, improve readability, maintainability and extensibility of your code. In addition, you can maximize reuse since the code written can be simultaneously leveraged server side via Node.js, offering the prospect of unifying our front ends and back ends in a single stack.

Maintain A Performant Single Business Source of Truth

Business logic must exist in the back end, if only for security and data integrity reasons. The more it exists solely in the back end, the better protected an architecture is from overinflation of front ends. Where different front end client applications exist, having a single business API unifies the application and minimizes the logic necessary in the front end to focus on user experience.

Not only does this single business layer implementation facilitate thinner clients that are more focused and use less code, it also allows for optimizing data services closer to where the data resides. As front ends have grown, data concerns have been second class citizens leading to performance compromises. A new evolved business layer can specialize in consolidation and performance, and help to address choke points involving large data sets or algorithms.

Microservices and Embracing a Truly N-Tiered Architecture

Microservices have become a successful strategy to mitigate challenges with performance, scalability and maintainability of application code. By breaking the application problem up into smaller pieces, the complexity and risk of each piece can be kept down. Microservices give you the ability to compartmentalize different bodies of functionality and to engineer and scale them individually.

However, the messaging domain of microservices is a complex problem that demands a documented strategy and adherence. The lowering of risk in the big picture is dependent

upon the boundaries of responsibility between services being identified carefully, as these divisions become resistant to change.

It is important not to allow the wider messaging domain of microservices to overcomplicate the decoupling between server and client. Good decoupling as well as security considerations point toward tightly controlling where client interfaces will communicate with the larger system. With the increased popularity of Node.js, this connection is sometimes becoming a complex layer in its own right. This requires careful and disciplined thinking rather than becoming a nebulous zone where architectural misses are papered over.

Adhere To a Blueprint

From the top down it is imperative to document the technical strategy formed by the collaboration between the CMO and CIO. Documentation of technology architecture should pervade the organization so that known problems are tackled in accordance with existing answers, and new conversations begin at the right point.

As client side applications have matured into true user experiences, the flows through them have become increasingly unique. It is important to resist the temptation to see these novel situations as exceptions to an overarching strategy. New challenges represent the time to evolve the technical strategy; novel situations are likely to repeat, and these situations are where it is most important to follow a coherent approach.

Let the strategic blueprint help guide those responsible for more tactical decisions and encourage engineering team documentation to evolve over time and not become stagnant.

Technology Takeaways

Despite the exponential increase in expertise of web application creation, the challenges faced make it a harder time than ever to make the right choices related to a clean architecture that lends itself to cost effective development. With increased overhead due to more devices consuming your products, the costs involved in making the wrong choices are higher.
With this raising of the stakes, you must balance the following priorities:

- Always consider the technical architecture in a holistic way; all parts are interlinked and can impact your own organization and the end user experience.
- Create and maintain a blueprint, and make sure it becomes part of development culture.
- Understand deviations from the desired approach as potentially valid tradeoffs with associated costs and consequences.
- Find maximum alignment between frameworks, patterns and best practices and your own unique challenges.

Meeting Two Goals: A Unified Approach To User Experience and Technical Architecture

The ever-changing technology landscape requires your teams to move faster than before. Simultaneously, customers are demanding new user experiences on a broadening range of platforms and devices. Keeping up with both in isolation is no longer a viable strategy.

You cannot prioritize the development of user experiences at the expense of application technical architecture. In order to satisfy the needs of evolving user experiences, you need to better architect applications anticipating the need for change, solving business problems where they make sense, and designing front ends to cater exclusively to delivering the best user experience possible.

Today more than ever, customers are demanding their needs are met in the context they choose - at the right time, in the right channel on the right device. The proliferation of front end experiences, be it desktop, mobile, OTT, car, home or otherwise is ever expanding, and there is no end in sight.

Enabling the customer experience while balancing the constraints of time, resources (technology/personnel) and money is the collaborative responsibility of the CMO and the CIO.

EQengineered

Recommendations

- Foster open and transparent communication between the CMO and CIO to understand the tradeoffs between pace, user experience, and technical architecture adherence and integrity.
- Enable the needed collaboration between the CMO and CIO to future proof your organization against disruption, costly technical debt, and unhappy customers.
- Design should be holistic to your projects rather than a discrete phase. Anticipate change, reduce iteration friction, solicit continuous customer feedback and embrace design thinking to establish a culture of innovation.
- Develop a technology “blue print” informed by your business strategy and the input of the CMO and CIO that will guide the organization and become a foundational document of architecture, patterns and development processes.

Closing Thoughts

- Digital disruption requires a disciplined approach to achieve results.
- Digital transformation may be thought as the third stage of embracing digital technologies: digital competence → digital literacy→ digital transformation.
- The pace of innovation and competition will lead to more personalized interactions.

EQengineered

Authors

Julian Flaks – Partner & CTO

Julian is a relentless problem solver and hoarder of full stack expertise. Having thrown himself headlong into Internet technology when best practices had barely begun to emerge, Julian is happiest putting his experience to use unlocking business value. Julian holds a Bachelor's of Laws from The University of Wolverhampton, England and a Master of Science in Software Engineering from The University of Westminster.

Mark Hewitt – Partner & Chief Revenue Officer

Mark Hewitt is responsible for the development, execution and leadership of sales, marketing and revenue growth strategies and initiatives to achieve EQengineered's short and long term results. Prior to joining EQengineered, Mark worked in various direct sales and sales management capacities at companies including Cantina, Molecular, Collaborative Consulting, and Forrester Research. Mark is a graduate of the United States Military Academy and served in the US Army.