POLYTIPE

Proyecto de Grado

Mejoramiento de procesos de producción en empresas creadoras de aplicaciones web mediante el uso de metodologías ágiles de desarrollo

Autores
Andrea Castañeda
Alejandro Sanclemente

Tutor
José Moncada

Asesor
Javier Aguirre

UNIVERSIDAD ICESI

dmi diseño de medios interactivos

Tabla de contenido

Planteamiento de la temática

En Colombia, según un estudio realizado en el año 2012 por la Federación Colombiana de la Industria del Software y Tecnologías de la Información-FEDESOFT (Fedesoft, 2012), de una muestra de 1120 empresas, el 48.57% se dedica al desarrollo de aplicaciones web, teniendo como principales líneas de negocio a nivel nacional el desarrollo a la medida, desarrollo de aplicaciones web y soporte y mantenimiento de software. En el informe, se observa cómo la industria de software y servicios relacionados tiene un grado de especialización bajo, en donde las firmas compiten en diferentes segmentos de mercado y en las que priman las líneas de negocio tradicionales del sector, por encima de las nuevas oportunidades de negocios que han surgido a nivel mundial en los últimos años (Fedesoft, 2012).

En la actualidad, existen diversos acercamientos al proceso de desarrollo de aplicaciones web. La CMS define este proceso como un marco utilizado para estructurar, planificar y controlar fases de desarrollo de sistemas de información (CMS, 2008). Una gran cantidad de estos marcos han evolucionado a lo largo de los años, cada uno con diferentes consideraciones técnicas y organizacionales que dependen del proyecto y del equipo de trabajo disponible. Es por esto que no existe una solución única que se aplique a todo tipo de proyectos; las metodologías tienen sus fortalezas y debilidades, siendo unas más eficaces que otras en ciertos escenarios.

Pero la industria ha conservado una metodología tradicional. Según un estudio publicado en la revista IEEE Software, el modelo en cascada es la metodología más utilizada en la industria (Neill & Laplante, 2003) y consiste en varias iteraciones entre el desarrollo de wireframes y simulaciones de interacción, hasta el proceso de diseño visual de la aplicación y etapa de desarrollo (Royce, 1970). Esto supone una inversión de tiempo elevada en la fase de planeación del proyecto, a pesar que en muchas ocasiones los requerimientos iniciales no son bien formulados y el proceso debe reiniciarse perdiendo tiempo de trabajo. En el ámbito del diseño esto implica diseñar varias veces la misma aplicación, en etapas iniciales desde un entorno de diseño gráfico y, en fases posteriores, desde un entorno basado en código.

La principal razón por la que se presenta esta situación se debe al uso del proceso secuencial conocido como modelo en cascada, originado en las industrias manufactureras y de construcción. Las primeras aproximaciones en el ámbito de la ingeniería del software datan del año 1956 y no es sino hasta la década de 1970 que se desarrolla una definición formal en el Lockheed Software Technology Center de Texas. Este modelo se establece entonces como una serie de etapas secuenciales que van desde la definición de requerimientos hasta la instalación y mantenimiento de sistemas completos (Royce, 1970) y se ha mantenido a día de hoy como la metodología más utilizada en el mundo en cuanto al desarrollo de software (Neill & Laplante, 2003).

Debido a su antigüedad y a la evolución de sistemas informáticos, el modelo ha sido criticado por su ineficacia en ciertos escenarios. De acuerdo con un paper publicado por el MIT en el año 1995 acerca del desarrollo de software en Microsoft, en casos donde los requisitos de la aplicación no están del todo claros, si los diseñadores tratan de crear especificaciones detalladas del producto demasiado temprano en el ciclo de desarrollo, el equipo del proyecto va a construir un producto que no cumple con las necesidades del cliente o que está desactualizado, [...] teniendo que deshacerse de muchas de sus partes (Cusumano, Smith, 1995).

Es por esto, que a partir de la década de 1990, junto con la aparición de nuevas tecnologías como cliente/servidor, programación orientada a objetos, y una lista cada vez mayor de herramientas de desarrollo de aplicaciones, surge el modelo de desarrollo rápido de aplicaciones (DRA). Este consiste en un conjunto integrado de técnicas, directrices y herramientas que facilitan el despliegue de las necesidades de software de un cliente en un corto plazo. Los proyectos no terminan súbitamente al final del ciclo de desarrollo, sino que se van elaborando durante el proceso basado en una continua retroalimentación del cliente. Además, todo el producto de software no se entrega al mismo tiempo, sino que siguen un orden de importancia comercial. De esta manera, el producto se implementa incrementalmente a medida que pasa el tiempo (Gottesdiener, 1995). Su objetivo es dedicar menos tiempo a las etapas de planeación y entrar a diseñar el sistema directamente, lo que ofrece un vistazo rápido a los estados dinámicos del sistema antes de ser construidos. (Hirschberg, 1998).

Otras metodologías que empezaron a aparecer incluyendo rapid throwaway prototypes, incremental development, evolutionary prototypes, reusable software, y automated software synthesis. Los desarrolladores de estas técnicas tuvieron distintas motivaciones, la principal de ellas fue buscar maneras de rescatar la industria del software de lo que parece ser un dilema: el software es casi siempre más costoso y se tarda en entregar más de lo que se espera sin mencionar, que muchas veces es poco fiable y falla en encontrar las necesidades del usuario. Los jefes del proyecto a menudo pasan por alto etapas o toman atajos con el fin de resolver el problema, pero estas alteraciones al ciclo de vida sólo hacen al software más costoso, tardío y poco fiable (Davis, Bersoff, Comer, 1988).

Esto se debe a que por cada aplicación, las necesidades del usuario están en constante evolución y el objetivo del proyecto cambia constantemente. Es por esto que se retrasa el cronograma del proyecto, cuando intentan cumplir un requisito que no estaba establecido inicialmente, y el resultado final no cumple con las expectativas del usuario, porque el desarrollador no entendió la necesidad del usuario (Davis, Bersoff, Comer, 1988).

Posteriormente, siguiendo esta filosofía de desarrollo que se adapta rápidamente a los cambios, surge el modelo de desarrollo ágil de software conocido como Agile, que sigue los lineamientos establecidos en el Agile Manifesto. Entre estos se destaca un enfoque en producir software funcional antes que en presentar documentos de especificaciones a los clientes y en una relación más directa con ellos para responder a los cambios de manera eficaz (Agile Manifesto, 2001).

Uno de los problemas que se presentan con esta metodología está relacionado con la calidad gráfica del producto. Esto es debido a la naturaleza ágil del modelo, que no presta demasiada atención a la apariencia de la aplicación que se construye y descuida un poco la interfaz de usuario, pudiendo entrar en errores de usabilidad. Las pruebas, maquetaciones, alternativas de diseño y la revisión de los lineamientos de una marca, son fases que tienen que ocurrir antes de construir un producto y no pueden dejarse de lado. El proceso funciona muy bien una vez que un diseño está en su lugar, pero no si el desarrollo de la apariencia de un sitio está destinado a ser parte del proyecto (Lillington, 2015).

Otro de los inconvenientes es la plataforma que se utiliza para desarrollar los proyectos, pues es necesario que puedan hacerse pruebas por múltiples usuarios con una variedad de dispositivos cada vez más alta. Actualmente, la web está evolucionando y se está cerrando la brecha que durante muchos años se llevaban las aplicaciones nativas y las basadas en un navegador web (Hazaël-Massieux, 2013). Google tiene esto muy presente, y es por eso que se encuentra desarrollando nuevas tecnologías como Polymer, una librería para la implementación de las últimas especificaciones de la W3C, que facilita el desarrollo de aplicaciones web con énfasis en un mayor rendimiento (Bleigh, 2014). Uno de los principales factores a considerar, en cuanto a las características del mercado objetivo, corresponde al uso exponencial de dispositivos móviles en los últimos años, que según un informe de la firma estadounidense comScore, en el 2014 el tiempo total que pasan las personas frente a dispositivos conectados a internet, diferentes al computador personal, alcanzó el 60% y continúa en aumento (comScore, 2014). Por último, es importante resaltar que el salto al uso de plataformas web para aplicaciones cada vez más complejas es inminente y que existe una muy buena oportunidad para desarrollar productos que faciliten el desarrollo de prototipos en esta plataforma, optimizando procesos mediante metodologías de desarrollo ágil y beneficiando a la industria de desarrollo de software, que sobrepasa las 5000 empresas (registradas en FEDESOFT) en el país (Fedesoft, 2012).

Pregunta de investigación

¿Cómo mejorar los procesos de producción en empresas creadoras de aplicaciones web mediante el uso de metodologías ágiles de desarrollo?

Objetivos

Objetivo General

- Mejorar los procesos de producción en empresas creadoras de aplicaciones web mediante el uso de metodologías ágiles de desarrollo

Objetivos Específicos

- Conocer lo que dice la teoría acerca de los modelos de desarrollo de aplicaciones.
- Identificar modelos de desarrollo ágil.
- Conocer los procesos que llevan a cabo empresas locales para el desarrollo de aplicaciones web.
- Proponer una solución que implemente metodologías ágiles para el desarrollo web.

Justificación

Existe una buena cantidad de empresas en el sector de las TICs que trabajan en desarrollo web en Colombia, pero que utilizan prácticas tradicionales. Es por esto, que el proyecto aporta un gran valor a las empresas en cuanto a la aplicación de diferentes modelos de desarrollo de aplicaciones enfocados en la optimización de procesos. De igual manera, aporta a la disminución en el tiempo de desarrollo de proyectos y en un mayor grado de fidelidad entre el diseño gráfico de la propuesta inicial y el producto final.

Se ha considerado que el proceso de prototipado es parte fundamental del desarrollo de aplicaciones, y la forma de elaborarlos no se ha optimizado completamente. Actualmente, no se encuentran herramientas exclusivamente enfocadas en la creación de prototipos que puedan ser escalados a un producto final sin tener que iterar por las fases de desarrollo. Se ha visto una muy buena oportunidad en realizar un aporte que facilite prototipar aplicaciones sin tener conocimientos avanzados de código y que se pueda llevar a un producto final fácilmente.

Marco de referencia

Categorías Conceptuales

- Modelos de desarrollo de aplicaciones
- Procesos de diseño web y UX

Modelos de desarrollo de aplicaciones

Modelo cascada: este modelo se originó en la década de 1950, donde compañías como la defensa de Estados Unidos, la industria espacial u otras que requerían aplicaciones comerciales, necesitaban sistemas de software de gran escala que tomaban varios años de desarrollo, suponían altos costos y evolucionaban lentamente. Fue modelado pensando en proyectos de diseño de hardware donde podía predecirse más fácilmente cómo las piezas del sistema iban a interactuar entre sí. En la actualidad, debido a que la mayoría de proyectos requieren iteraciones entre la especificación, el desarrollo y las pruebas; algunas compañías se han alejado del modelo en cascada en la práctica (Cusumano, Smith, 1995).

Modelo incremental: consiste en una serie de cascadas más pequeñas en cada paso del modelo en cascada enfocadas en dar solución a problemas específicos antes de pasar al siguiente incremento, buscando lograr una mayor adaptación a los cambios que puedan presentarse. La fase inicial de planteamiento de requerimientos se da igual al modelo en cascada, seguido de un proceso iterativo de prototipado hasta llegar al prototipo final (CMS, 2008). Una de sus debilidades es que los requerimientos generales del sistema deben estar claros desde un principio, lo cual pocas veces sucede y causa que se deba retroceder fases del desarrollo cuando algo no fue planeado completamente.

Modelo en espiral: es una combinación entre un proceso lineal y uno iterativo. Consiste en un ciclo de 4 etapas: determinar objetivos, alternativas y restricciones; evaluar alternativas, identificar y resolver riesgos; desarrollar y verificar entregables; y planear la siguiente iteración. Cada ciclo involucra una progresión a través de la misma secuencia de pasos, para cada nivel de elaboración del proyecto desde una etapa conceptual hasta la elaboración de código. Con esto se busca minimizar los riesgos facilitando realizar cambios a lo largo de todo el proyecto, pero su mayor debilidad consiste en un tiempo elevado en cada etapa del desarrollo, aumentando los costos y los tiempos de entrega (CMS, 2008).

Metodologías ágiles de desarrollo: Las metodologías ágiles de desarrollo son una alternativa a las mencionadas anteriormente que ayudan a responder de manera más rápida a cambios durante el proceso de desarrollo.

Entre las metodologías ágiles se encuentran:

Scrum: En esta metodología el dueño del producto trabaja con el equipo para identificar y priorizar la funcionalidad del sistema. Se usa una lista de prioridades que consiste en características, corrección de errores y requerimientos no funcionales que deben desarrollarse en orden de entregar un sistema de software exitoso. Una vez un punto ha sido resuelto, se vuelve a analizar la lista y el siguiente punto con mayor prioridad entra en desarrollo.(McLaughlin, 2015).

Lean and Kanban Software Development: En esta metodología seleccionan la característica que genera más valor al sistema y les da mayor prioridad en el desarrollo, se enfatiza en la eficiencia del flujo de trabajo y se concentra en el uso de recursos procurando que todo el equipo sea lo más productivo posible. Aquí se recomienda fuertemente que la unidad de pruebas se desarrolle al mismo tiempo que el código. Kanban está basado en tres principios básicos (McLaughlin, 2015):

- Visualizar lo que harás hoy (flujo de trabajo)
- Limitar la cantidad de trabajo en proceso: Esto ayuda a balancear el flujo basado en el objetivo inmediato que se debe alcanzar, así los equipos no trabajan en muchas cosas al mismo tiempo.
- Mejorar el flujo: cuando algo es terminado, lo siguiente más importante de la lista se pone en marcha.

Extreme Programming (XP): El objetivo de esta metodología es entregar software de alta calidad de manera rápida y continua. Promueve un alto involucramiento con el cliente, rápidos circuitos de retroalimentación, pruebas y planeación constante además de un gran trabajo en equipo que pueda entregar software terminado en intervalos muy frecuentes. En XP el cliente trabaja de cerca al equipo de desarrollo para definir prioridades y se refieren a ellas como “Historias de usuario”. El equipo de desarrollo estima, planea y entrega la mayor prioridad de la historia de usuario en orden de maximizar la productividad, estas prácticas proveen un marco que soporto y guía al equipo a garantizar un software de alta calidad.(McLaughlin, 2015).

Crystal: Este método es uno de los más ligeros y adaptables acercamientos al desarrollo de software. Comprime una familia de metodologías ágiles como Crystal Yellow, Crystal Orange y otras donde sus características únicas están basadas en factores como el tamaño del equipo, criticalidad del sistema y las prioridades del proyecto. Crystal promueve entrega temprana y frecuente de software terminado, alto involucramiento del usuario, adaptabilidad y la eliminación de distracciones (McLaughlin, 2015).

Dynamic Systems Development Method (DSDM): Está basado en nueve principios clave: Negocios, necesidades, valor, participación activa del usuario, empoderamiento de equipos, entregas frecuentes, pruebas integradas y colaboración de las partes interesadas. (McLaughlin, 2015).

Feature-Driven Development (FDD): Este modelo empieza empieza estableciendo una figura general del modelo, luego con una serie de iteraciones diseñando y construyendo por funciones. Las funciones son pequeños resultados que son “útiles para los ojos del cliente” . FDD recomienda prácticas específicas de programación como “Pruebas regulares” y “Propiedades Clase/Componente”. Los autores de este método aseguran que FDD se escala de manera más directa que otros acercamientos y funciona mejor con grandes equipos. (McLaughlin, 2015).

Luego de examinar las diferentes metodologías de desarrollo de aplicaciones más comunes y de identificar las fortalezas y debilidades de cada una de ellas, se evidencia como las metodologías de desarrollo ágiles son las que mejor se adaptan a la problemática en el entorno nacional. Es el modelo ideal para beneficiar a empresas de desarrollo web locales al manejar factores que involucran un tamaño de equipo pequeño, una carga de trabajo moderada y una duración de proyectos corta similar a lo manejado en el país. Este tipo de metodologías ágiles son las indicadas porque permiten la adaptabilidad a cualquier tipo de cambio que surja durante el proceso del proyecto sin afectar su desarrollo garantizando eficacia, calidad y entrega del producto en el tiempo justo.

Tenemos también, que Material Design es un lenguaje visual que sintetiza los principios del buen diseño con la innovación y posibilidades de la tecnología y la ciencia, creando así, un sistema que permite unificar la experiencia del usuario a través de distintas plataformas y dispositivos de diferentes tamaños ("Introduction - Material design - Google design guidelines", 2016).

Material Design está diseñado en un espacio tridimensional donde cada elemento ocupa su propio lugar en el eje z, dando al usuario la impresión de que cada elemento existe en un lugar dentro de la pantalla. Igualmente, Material Design tiene en cuenta el tiempo ya que el usuario no sólo experimentan los estados de los elementos sino las transiciones entre ellos. Es por ésto que también le dedican gran tiempo a las animaciones de cada componente, haciendo la experiencia más real para el usuario ("Introduction - Material design - Google design guidelines", 2016).

Procesos de diseño web y UX

El Diseño de experiencias de usuario (UX) es una disciplina que se enfoca en diseñar de principio a fin la experiencia de cierto producto, en este caso específicamente en el ámbito web. Diseñar experiencias significa planear reacciones a una serie de eventos por parte de un público objetivo, lo cual genera un cambio en su comportamiento al interactuar con un producto en específico. UX no trata solamente temas de prototipado, diseño visual e investigación, sino que va mucho más allá, abarcando temas acerca de la creación de productos eficientes y agradables basados en el conocimiento sobre el comportamiento humano y las emociones. Treder (2015).

Fases del Diseño de Experiencias de Usuario

Para el profesor Guiseppe Getto existen cuatro fases fundamentales en el diseño de experiencias de usuario, que son a su vez etapas recursivas dentro del proceso de desarrollo de aplicaciones (2015). Estas se encuentran descritas en el manual de Getting Started With UX Design Process & Documentation publicado por UXPin (2015).

Investigación preliminar
Comprende las acciones que se llevan a cabo para entender lo que el usuario quiere y necesita en el contexto en que se usaría el producto final. En esta fase es importante definir:

Personas: Son perfiles o arquetipos de usuarios, representativos de una población demográfica para la investigación.

Historias de usuario: Se refiere a las acciones que deberán realizar los usuarios teniendo en cuenta sus motivaciones.

Escenarios de usuario: Usuario + tarea + contexto = escenario.

User flows: Para cada escenario, es el camino que toman los usuarios para desarrollar una tarea específica. Consiste en una representación gráfica de la navegación que lleva a cabo un usuario.

Prototipado
Luego de conocer los usuarios y sus necesidades, es tiempo de hacer algo con ellos. El principal propósito de construir prototipos es evaluar si el flujo de interacción de un producto es consistente. Para lograrlo se debe tener en cuenta la relación entre prototipos, interacción, y elementos de la interfaz, que son los componentes visuales que el usuario verá.

Pruebas de usuario
El proceso de diseño es recursivo, ya que pocas veces avanza de manera lineal. Las pruebas de usuario pueden llevar a descubrir fallas en la planeación o variables no pensadas hasta este momento. Éstas se encuentran diseñadas para probar la usabilidad de una aplicación, evaluando a través de métricas cuán efectiva es ayudando a los usuarios a alcanzar sus objetivos, que deben estar claros desde la etapa de investigación.

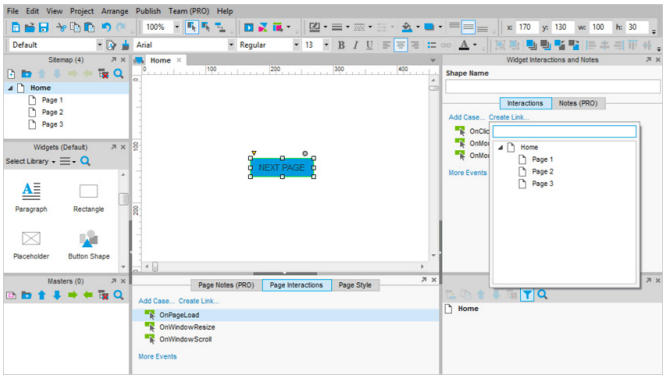
Mantenimiento
Las pruebas de usuario buscan mejorar los prototipos. Es complicado predecir cómo los usuarios reales interactuarán con él, es por esto que es probable y entendible que haya problemas de usabilidad en este punto. Lo que se debe hacer es tomar la retroalimentación que arrojan dichas pruebas y realizar cambios y volver a probar para garantizar que se mejoren los diferentes aspectos de la aplicación.

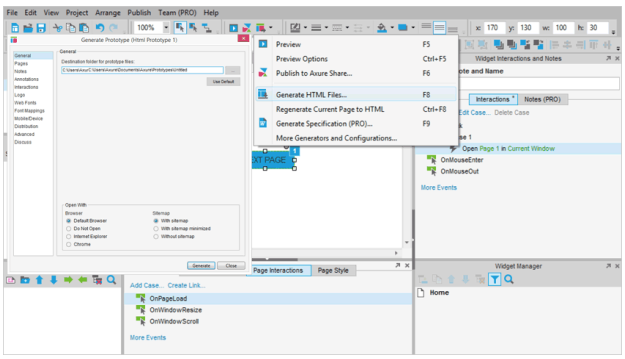
Estado del arte

Referencias

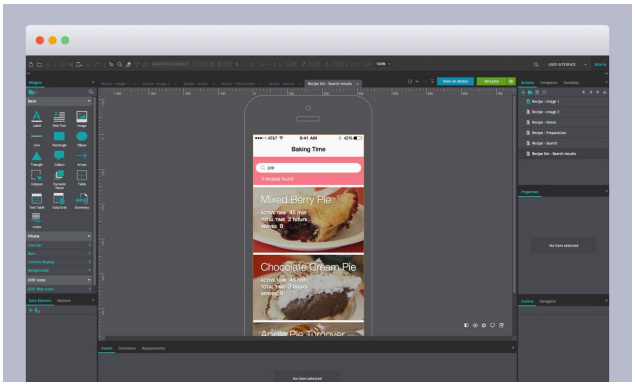
Diversas herramientas como WebFlow y Macaw se han creado para producir código mediante una interfaz gráfica. Su objetivo es el de construir sitios web a partir de componentes pre-establecidos ofrecidos en la aplicación. Por otra parte, existen herramientas de prototipado como Balsamiq, que dejan de lado el código para enfocarse en una solución meramente gráfica. También existen otras como InVision que, a partir de imágenes crea secuencias de interacción pre-establecidas. Las herramientas de prototipado analizadas no cumplen con el objetivo de este proyecto de grado de optimizar procesos y lograr mayor eficacia en el desarrollo de aplicaciones web, pues unas descuidan completamente la fase de prototipado y otras, al contrario, descuidan las fases de desarrollo del producto final.

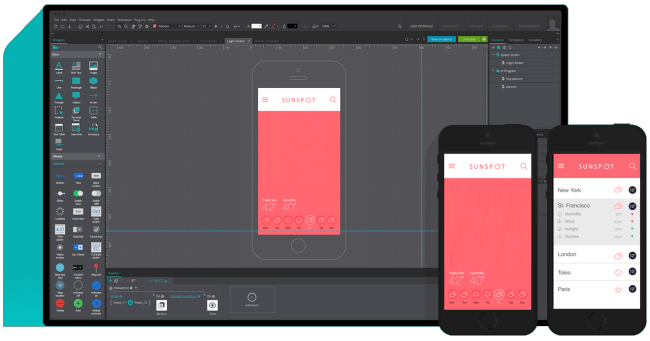
Nombre del Proyecto: Axure
Fecha: Enero 2003
Autores: Victor Hsu y Martin Smith
Lugar: Estados Unidos
Descripción: Es ideal para prototipos de aplicaciones web o software de escritorio que incluyen interacciones complejas y múltiples estados.
¿Cómo aporta al proyecto?
La funcionalidad de exportar a código permite generar HTML a partir de las interacciones con los elementos en la aplicación, lo cual es similar a lo que buscamos. También, permite visualizar la interacción entre pantallas.



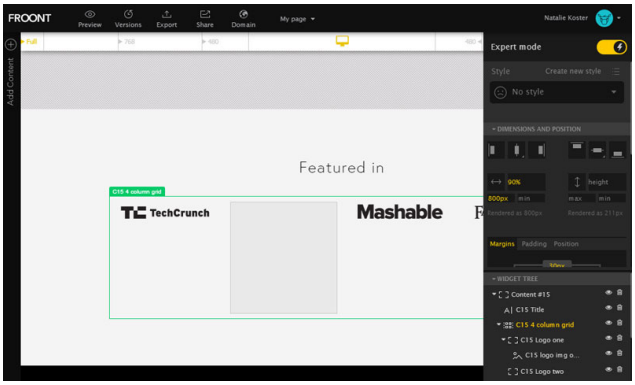


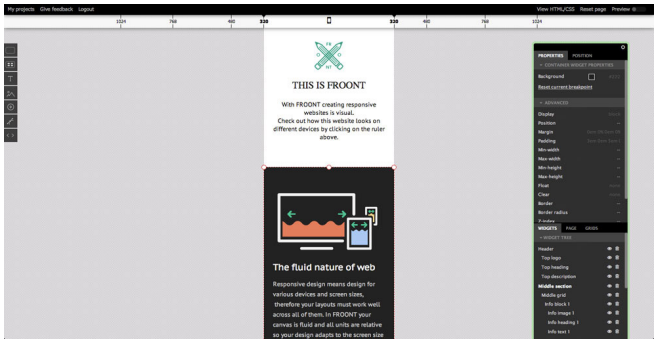
Nombre del Proyecto: Justinmind
Fecha: Octubre, 2007
Descripción: Herramienta de creación de prototipos flexibles que soporta varios dispositivos y funciona bien para prototipos de interacciones simples o complejas. Se pueden crear desde plantillas o mediante la construcción de pantallas utilizando bibliotecas estándar.
¿Cómo aporta al proyecto?
Su interfaz es amigable al usuario y cuenta con diferentes tamaños para visualizar la aplicación. También tiene elementos de la interfaz de iOS.



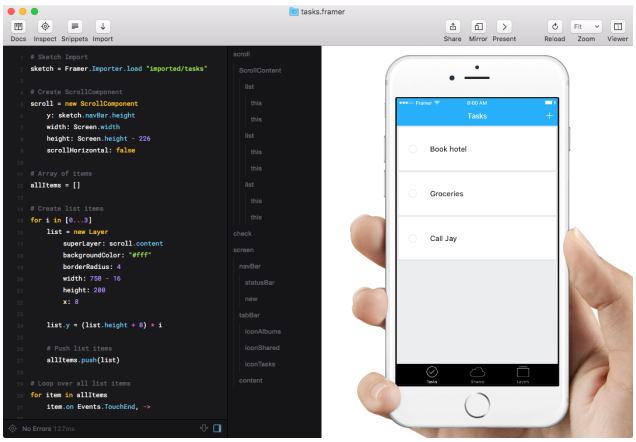


Nombre del Proyecto: Froont
Fecha: Septiembre, 2014
Autor: Sandijs Ruluks
Lugar: Estados Unidos
Descripción: Herramienta flexible para crear y colaborar en el diseño web sin código.
¿Cómo aporta al proyecto?
Utiliza componentes y tiene una interfaz muy amigable, además permite exportar en código HTML.

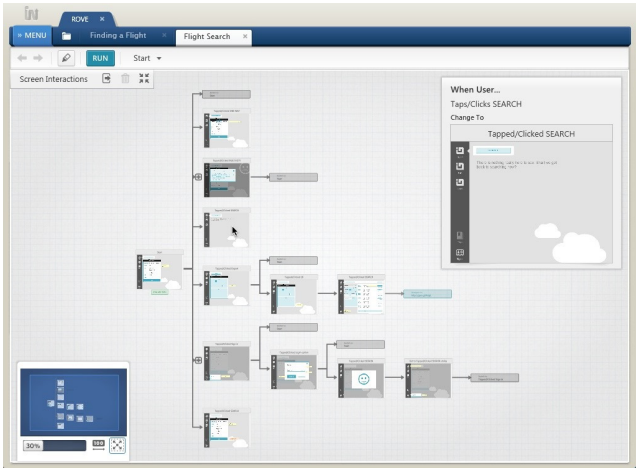


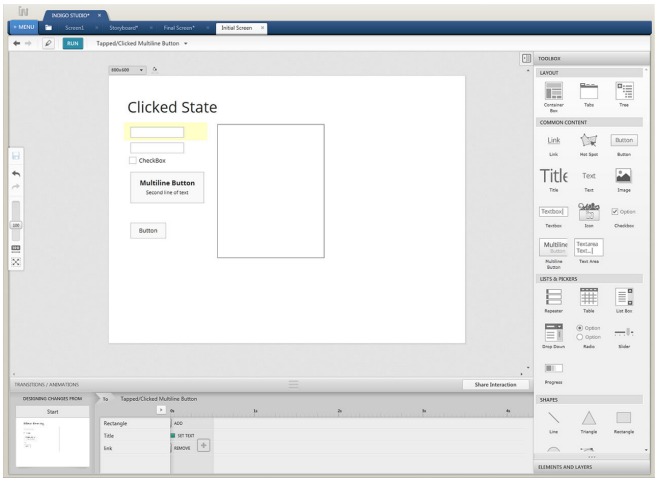


Nombre del Proyecto: Framer
Fecha: Mayo, 2014
Lugar: Países Bajos
Descripción: Es ideal para la creación de prototipos de animaciones de alta fidelidad o interacciones complejas de escritorio y aplicaciones móviles.
¿Cómo aporta al proyecto?
Fácil manejo de la edición de código permitiendo el desarrollo de animaciones e interacciones.

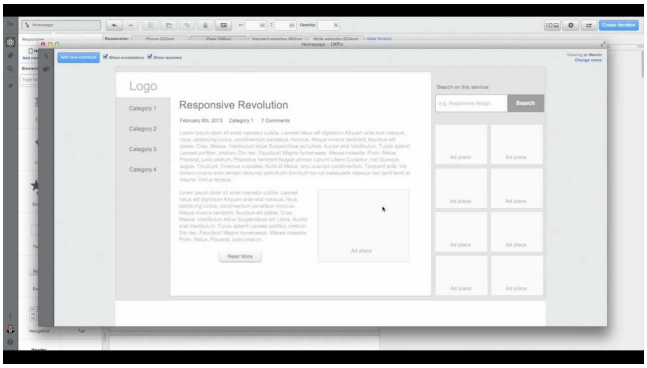


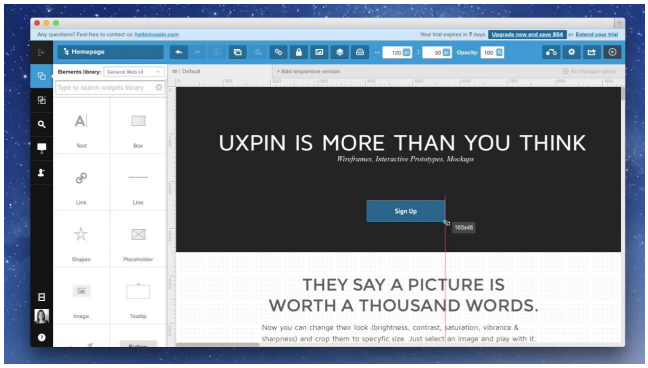
Nombre del Proyecto: Indigo Studio
Fecha: Septiembre, 2014
Autor: Dean Guida
Lugar: Estados Unidos
Descripción: Creación de prototipos utilizando componentes preestablecidos.
¿Cómo aporta al proyecto?
Es una herramienta de desarrollo de wireframes que nos aporta en la forma de la visualización del flujo de interacción de la aplicación.



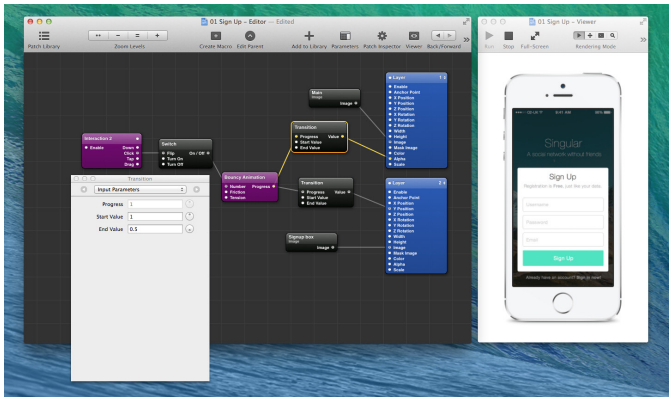


Nombre del Proyecto: UXPin
Fecha: 2010
Autor: Marcin Trender
Lugar: Polonia
Descripción: Ideal para la creación de prototipos web o de aplicaciones de que requieren interacciones ágiles. Incluye bibliotecas que ayudan al desarrollo.
¿Cómo aporta al proyecto?
Tiene una interfaz completa y maneja una amplia biblioteca de componentes propios.

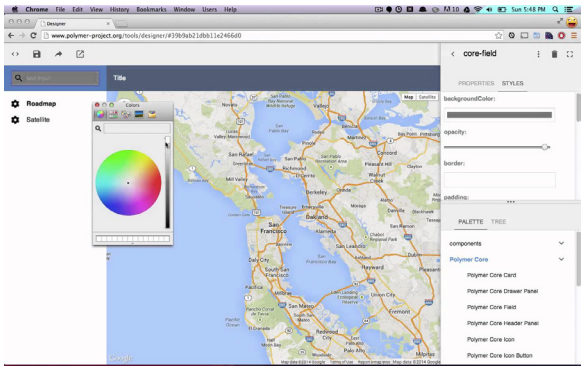


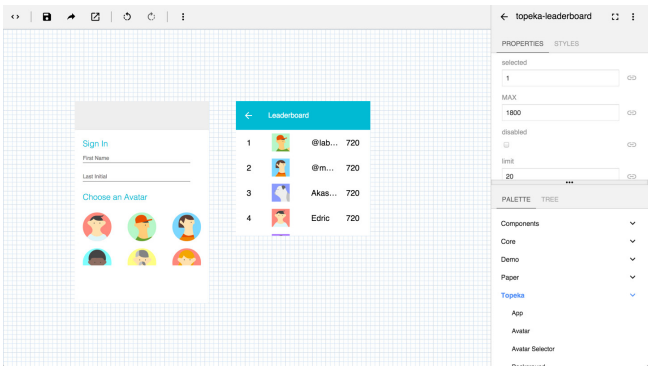


Nombre del Proyecto: Origami
Fecha: Diciembre, 2013
Autor: Facebook
Lugar: Estados Unidos
Descripción: Herramienta gratuita para el diseño de interfaces de usuario modernas. Rápidamente se puede armar un prototipo, correrlo en iOS, iterar, y exportar fragmentos de código que ingenieros pueden usar.
¿Cómo aporta al proyecto?
La manera de simular las interacciones entre elementos y pantallas es bastante avanzada, lo cual puede servir de guía. La aplicación sólo funciona en Mac, lo cual es limitante para los usuarios; sería ideal que fuera multiplataforma.

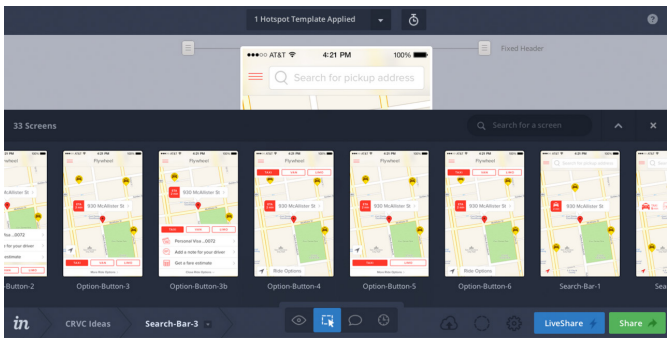


Nombre del Proyecto: Polymer Designer
Fecha: Diciembre, 2013
Autor: Google - Polymer Team
Lugar: Estados Unidos
Descripción: Herramienta de código abierto para la creación de interfaces utilizando componentes. Permite crear interacciones básicas entre elementos en aplicaciones de una sola pantalla.
¿Cómo aporta al proyecto?
Es basado en web, cuenta con componentes prediseñados de Material Design. Tiene una opción de controlar el versionamiento a través de git y permite exportar código HTML. Puede ser una base para nuestro proyecto.





Nombre del Proyecto: InVision
Fecha: Diciembre, 2012
Autor: Andy Orsow
Lugar: Estados Unidos
Descripción: Es una herramienta de prototipado que permite cargar los archivos de capturas de la aplicación y transformarlas en prototipos interactivos, agregando áreas sensibles, gestos y transiciones.
¿Cómo aporta al proyecto?
Permite el desarrollo colaborativo el cual queremos implementar por medio de git.



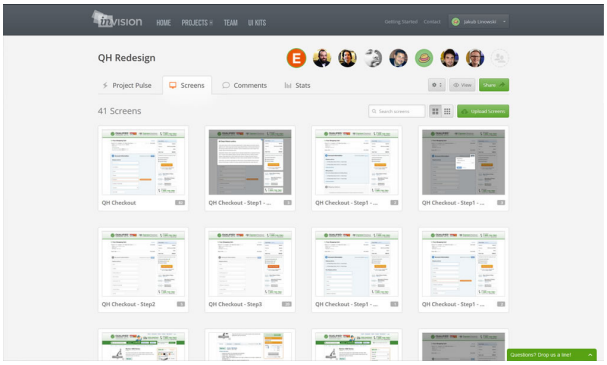


Tabla comparativa de características

ESTADO DEL ARTE - HERRAMIENTAS DE PROTOTIPADO					
TOOL	DISEÑO	INTERACCIÓN	COMPONENTES	EXPORTAR CÓD. HTML	OPEN SOURCE
Axure	■	■		■	
Justinmind		■	■		
Froont	■	■		■	
Framer			■	■	
Indigo Studio	■	■			
UXPin	■				
Origami	■			■	■
Polymer Designer	■		■	■	■
InVision		■			

■ Función disponible ■ Tiene limitaciones

Diseño
Enfocado en crear el diseño final de la aplicación y no solamente mockups. Siguen un estilo gráfico predefinido para sus elementos, ya sea propio, de iOS o Material Design.

Flujo de interacción
Es posible crear y visualizar el flujo general de interacción de la aplicación entre pantallas.

Componentes
Enfocado en la creación de componentes para su reutilización en otros proyectos.

Exportar a código HTML
Permite exportar a código lo creado en la aplicación.

Open Source
Herramienta gratuita de software libre que permite ser utilizada y modificada.

Conclusiones estado del arte

Las herramientas aquí exploradas ofrecen ideas que pueden adaptarse a nuestro proyecto. Vemos como ninguna tiene las mismas funciones que lo que nosotros buscamos realizar, pero que en su conjunto se pueden tomar partes de aquí y de allá para formular una solución completa. Siguiendo esto, nuestra intención es la de crear una herramienta enfocada en crear el diseño final de la aplicación desde un principio del desarrollo, lograr visualizar y simular el flujo general de interacción utilizando componentes que puedan ser reutilizables en diferentes proyectos, que pueda ser exportar como código para que luego de servir como prototipo pueda utilizarse para el producto final. También debe tener una curva de aprendizaje baja para que sea fácil utilizarlo, debe seguir los lineamientos de diseño ya sea de iOS o Android para lograr una buena fidelidad y facilitar la obtención de métricas a la hora de hacer pruebas de usuario.

Trabajo de campo

El objetivo del trabajo de campo fue el de recopilar información acerca de las metodologías de desarrollo de aplicaciones web empleadas por empresas locales, cómo están conformados los equipos de trabajo y cuál es el tiempo promedio de duración de un proyecto. Estos datos son parte importante para la validación de las observaciones hechas durante la fase de investigación bibliográfica y orientan la solución que se plantea más adelante.

Grupo objetivo

Empresas pequeñas y startups que ofrecen productos y servicios en el ámbito del desarrollo de aplicaciones web de la ciudad de Cali, que cuentan con un equipo de trabajo menor a 8 personas.

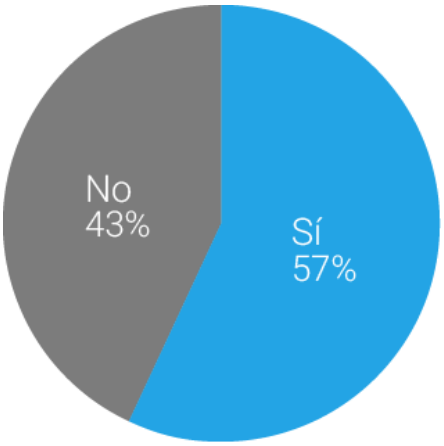
Metodología

Cuantitativa. Se realizó una encuesta donde se recopiló información de 7 empresas de desarrollo web locales para conocer más a fondo acerca de las metodologías de desarrollo empleadas por dichas empresas, cómo están conformados los equipos de trabajo y cuál es el tiempo promedio de duración de un proyecto. Con estos datos se pudo validar las observaciones hechas durante la fase de investigación bibliográfica y son parte fundamental para orientar de una mejor manera la solución a plantear a esta problemática.

Resultados

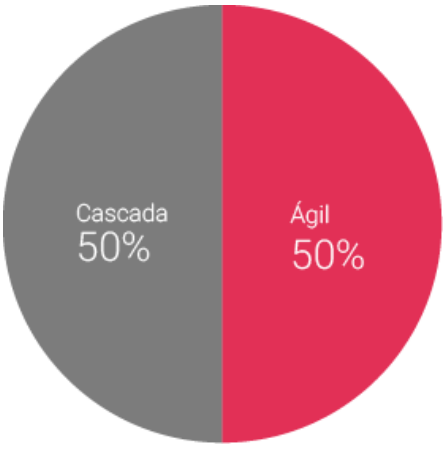
• Uno de los puntos descubiertos en la encuesta es que cerca de la mitad de las empresas encuestadas (43%) no utilizan ninguna metodología estándar de desarrollo.

¿Siguen algún proceso estándar de desarrollo?



• Las que sí lo hacen, siguen en un 50% el modelo en cascada y el otro 50% utilizan metodologías ágiles.

Modelo de desarrollo empleado

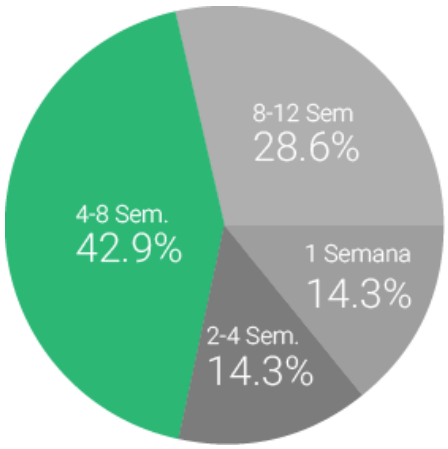


• Se identificaron unas etapas claras en el proceso de desarrollo en todas las empresas:

a. Identificación de requerimientos y usuarios
b. Maquetación y Diseño
c. Desarrollo e implementación
d. Pruebas

• El 71% de los proyectos tardan entre 4 y 12 semanas de desarrollo, siendo de 4 a 8 semanas la duración más común.

¿En promedio cuánto se tarda el desarrollo de un proyecto?



• El tiempo empleado en el planteamiento de requerimientos corresponde al 30% del tiempo total de desarrollo.

• En cuanto al desarrollo de la interfaz se tarda el 28% del tiempo total en promedio.

• Para la programación se tarda alrededor de un 52% del tiempo total de desarrollo.

• Pruebas de usuario toman en promedio el 32% del tiempo total de desarrollo.

• Las empresas realizan tanto soluciones de diseño a la medida como hacen uso de herramientas de contenido a través de plantillas como Wordpress, Joomla, entre otras.

• Los equipos de trabajo son inferiores a 8 personas y se comprenden principalmente de ingenieros y diseñadores.

• La gran mayoría (85%) lleva un control de versionamiento a través de Git.

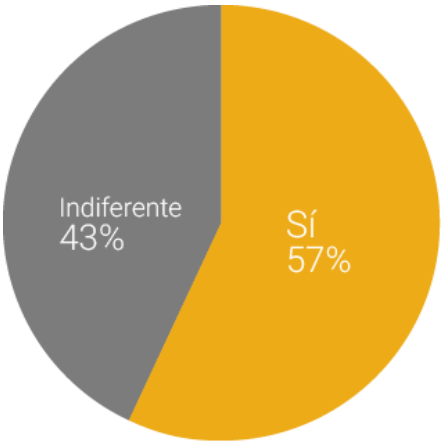
• Los problemas más comunes son:

a. El cliente cambia los requerimientos constantemente
b. El cliente pide cosas demasiado complejas
c. Integración de la interfaz en los distintos módulos de código

• Se mantiene el grado de participación del cliente en todos los casos, pues participa a lo largo del desarrollo. Una reunión de requerimientos inicial, reuniones intermedias para seguir el progreso y una reunión final de entrega.

• El usuario es prioridad en el desarrollo de la aplicación en el 57% de los casos, y en el 43% restante en ocasiones es indiferente el usuario.

Cuando hacen la validación de la aplicación,
¿tienen en cuenta al usuario?



Determinantes de diseño

Multiplataforma
La aplicación debe funcionar en diferentes sistemas operativos. Por ello, será realizada mediante tecnologías y estándares web actuales, buscando abarcar la mayor cantidad de dispositivos posible.

Material Design
La aplicación debe seguir los lineamientos de Material Design de Google pues permite diferentes ventajas de la mano de Polymer y componentes web, además de ser de carácter open source.

Metodologías Ágiles
La aplicación debe estar enfocada a funcionar en entornos de desarrollo ágil correspondiente al panorama actual de las empresas desarrolladoras locales, que son equipos de trabajo pequeños y tiempos cortos en los proyectos.

Versionamiento Git
La aplicación debe adaptarse a la forma de trabajar de las empresas integrándose con git para controlar su versionamiento y progreso de desarrollo.

Componentes Web
La aplicación deberá contar con componentes prediseñados basados en Material Design utilizando la librería de Polymer y permitiendo al usuario agregar tanto elementos propios como creados por otros usuarios descargados a través de internet.

Últimas versiones de Chrome, Firefox, Safari, Opera
Para que los últimos estándares de la W3C funcionen completamente, la aplicación deberá desempeñarse correctamente en las más recientes versiones de los principales navegadores: Chrome, Firefox, Safari y Opera, dejando de lado a Internet Explorer y Edge por el momento por motivos de compatibilidad.

Propuesta de desarrollo

Nuestra propuesta es Polytipe, una plataforma web de prototipado que permite que el diseñador, principiante o avanzado, desarrolle sus propios proyectos en menor tiempo y con la posibilidad de reutilizar el prototipo para el producto final optimizando la creación de piezas gráficas y código. Esta plataforma será capaz de implementar el diseño del usuario a través de una interfaz gráfica y volverlo código, así como facilitar el proceso de creación de su interacción empleando metodologías de desarrollo ágil. Para garantizar esta unidad de diseño y código, se tendrán en cuenta los lineamientos de Material Design para los componentes de la interfaz a través de la librería de Polymer.

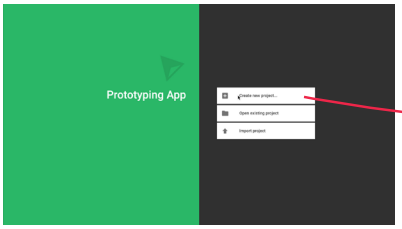
La estructura general de la aplicación consiste en la creación de proyectos independientes, cada uno con la posibilidad de contar con múltiples pantallas. Dentro de cada una hay una variedad de componentes prediseñados de Material Design que van desde botones y menús hasta barras de herramientas y diálogos que pueden ser utilizados para la construcción de las interfaces del prototipo. Asimismo, podrán crearse las interacciones entre estas pantallas y probar el flujo de navegación de la aplicación mediante pruebas de usuario y obtención de métricas.

Respondiendo a la necesidad de las empresas locales de llevar a cabo un control del versionamiento, la aplicación deberá estar integrada con git para guardar el progreso en la nube y facilitar la colaboración entre equipos de trabajo en los proyectos.

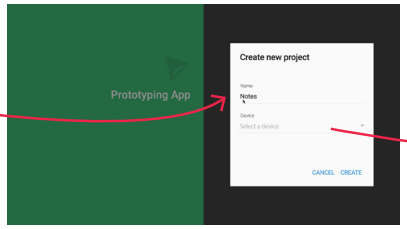
La web ha sido principalmente desarrollada de manera libre desde sus inicios. La especificación de la W3C respecto a los componentes web no es la excepción y herramientas como Polymer tampoco lo son. Es por esto que buscamos que la aplicación sea open source permitiendo lograr un mayor crecimiento con la ayuda de la comunidad global, así como llenando el vacío en el estado del arte de una herramienta de prototipado gratuita y libre.

Actualmente, la comunidad ha desarrollado cientos de componentes web que están disponibles al público a través de la página customelements.io. Al trabajar de manera open source podemos aprovecharlos e incluirlos en nuestra aplicación, permitiendo que elementos como sliders de fotos, mapas, selectores de fecha, etc. que no hacen parte inicial de nuestra propuesta, puedan ser utilizados por nuestros usuarios.

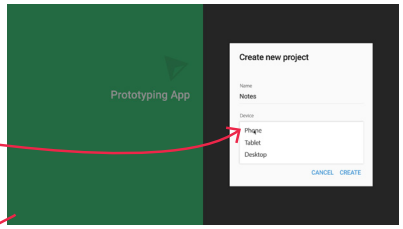
Por último, para que el prototipo no se quede en el olvido luego de realizar las pruebas de usuario, la aplicación permitirá que se pueda acceder al código HTML para utilizarlo en el desarrollo del producto final, optimizando tiempo de desarrollo y garantizando una unidad del diseño entre la propuesta y lo que se entrega al cliente.



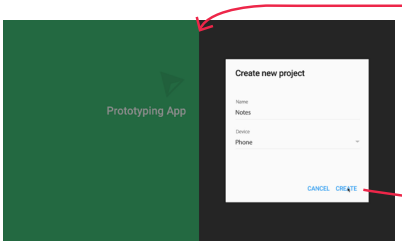
Pantalla de inicio



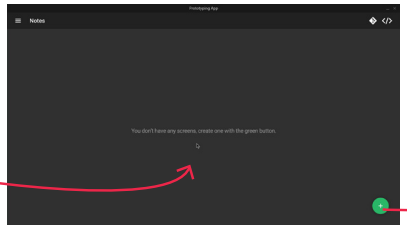
Crear nuevo proyecto



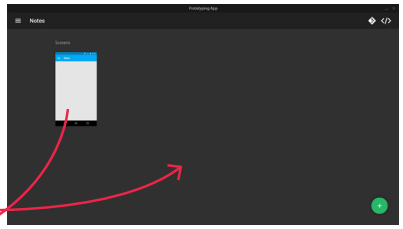
Crear nuevo proyecto / Selección de dispositivo



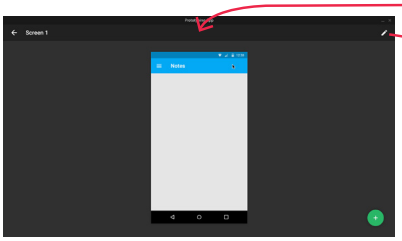
Dispositivo seleccionado



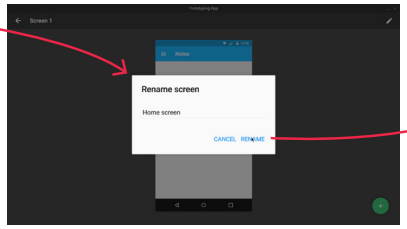
Vista general del proyecto / Sin pantallas



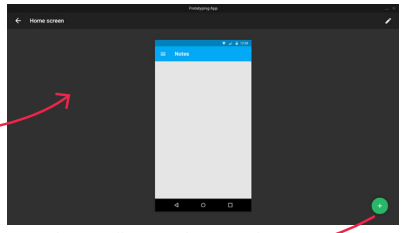
Vista general del proyecto / Una pantalla



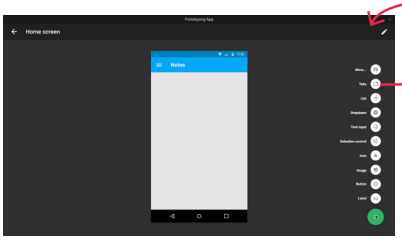
Vista de pantalla



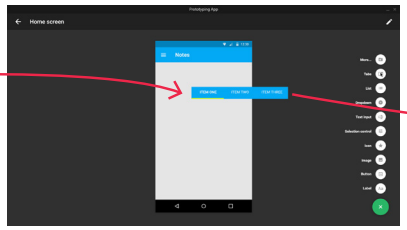
Vista de pantalla / Cambiar nombre



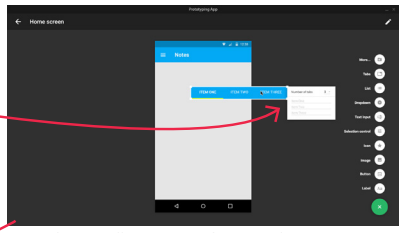
Vista de pantalla / Cambio completo



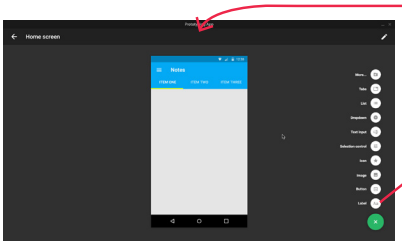
Vista de pantalla / Componentes desplegados



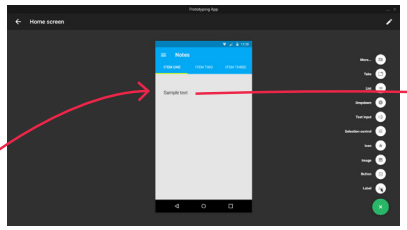
Vista de pantalla / Pestañas agregadas



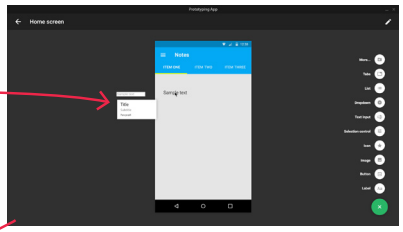
Vista de pantalla / Personalización de pestañas



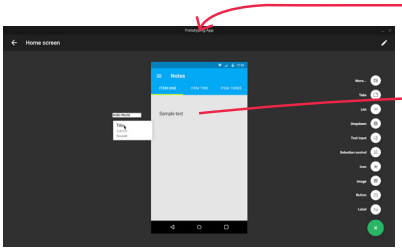
Vista de pantalla / Pestañas agregadas



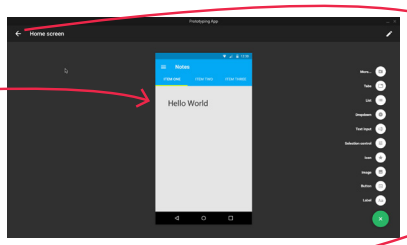
Vista de pantalla / Etiqueta agregada



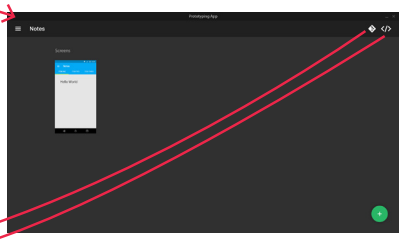
Vista de pantalla / Personalización de etiqueta



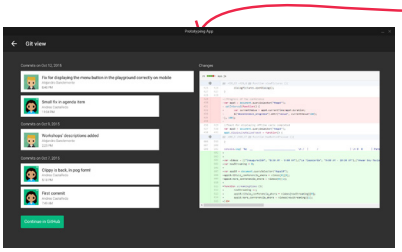
Vista de pantalla / Estilos de etiqueta



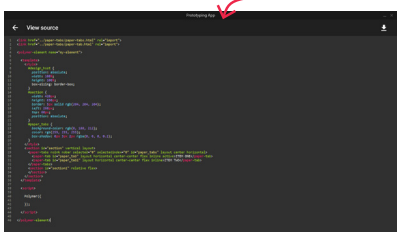
Vista de pantalla / Etiqueta agregada



Vista general del proyecto

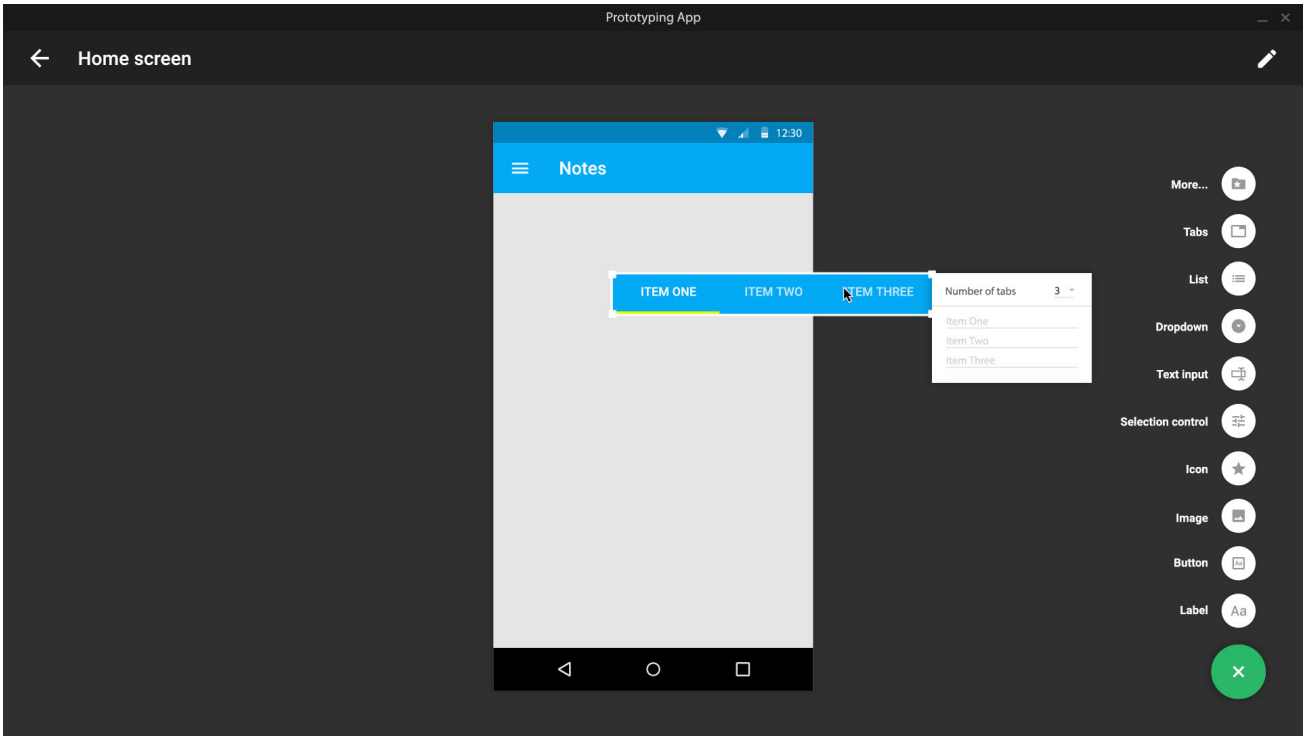


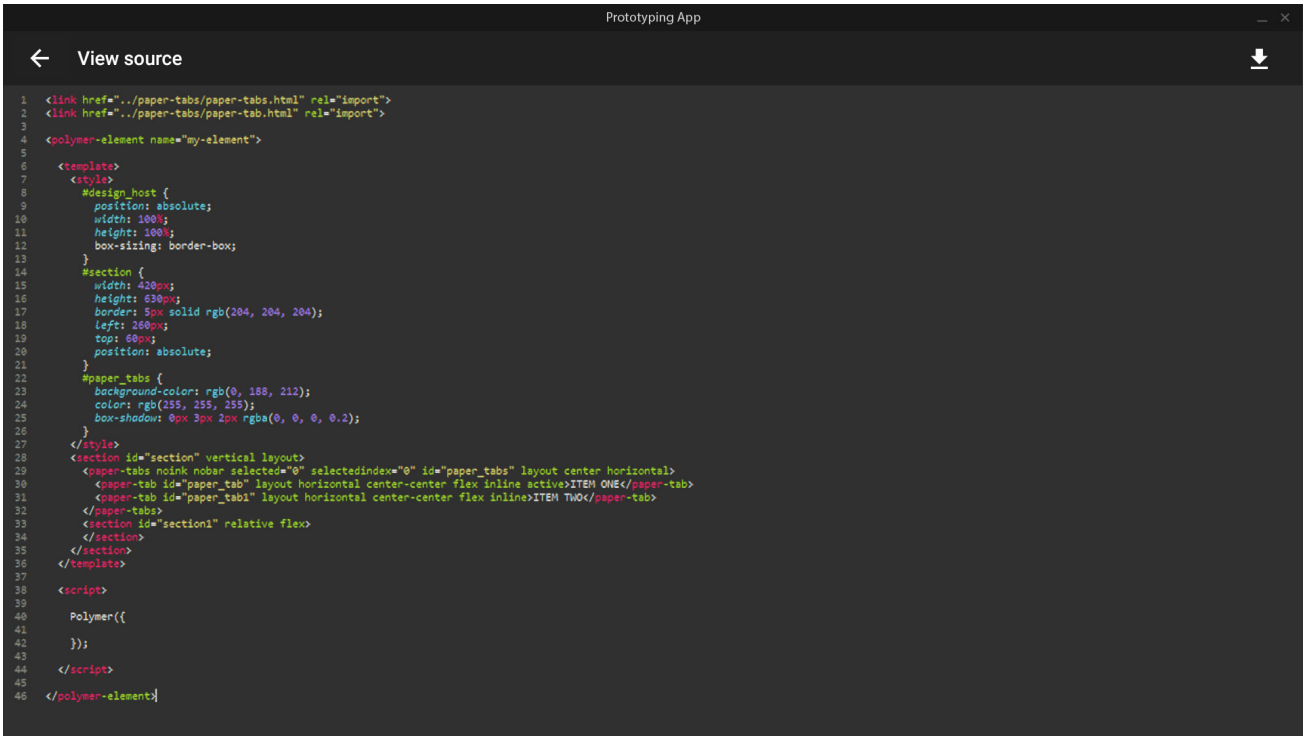
Versionamiento Git



Vista de código

Flujo de interacción

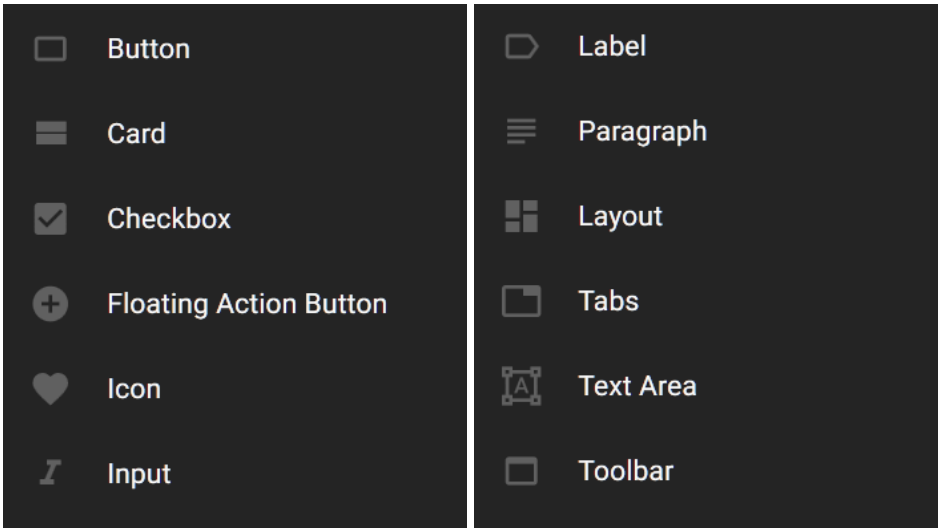

Prototyping App
Home screen
12:30
Notes
ITEM ONE
ITEM TWO
ITEM THREE
Number of tabs
3
Item One
Item Two
Item Three
More...
Tabs
List
Dropdown
Text input
Selection control
Icon
Image
Button
Label
Aa


Prototyping App
View source
<link href="../paper-tabs/paper-tabs.html" rel="import">
<link href="../paper-tabs/paper-tab.html" rel="import">
<polymer-element name="my-element">
<template>
<style>
#design_host {
position: absolute;
width: 100%;
height: 100%;
box-sizing: border-box;
}
#section {
width: 420px;
height: 630px;
border: 5px solid rgb(204, 204, 204);
left: 260px;
top: 60px;
position: absolute;
}
#paper_tabs {
background-color: rgb(0, 188, 212);
color: rgb(255, 255, 255);
box-shadow: 0px 3px 2px rgba(0, 0, 0, 0.2);
}
</style>
<section id="section" vertical layout>
<paper-tabs noink nobar selected="0" selectedindex="0" id="paper_tabs" layout center horizontal>
<paper-tab id="paper_tab" layout horizontal center-center flex inline active>ITEM ONE</paper-tab>
<paper-tab id="paper_tab1" layout horizontal center-center flex inline>ITEM TWO</paper-tab>
</paper-tabs>
<section id="section1" relative flex>
</section>
</section>
</template>
<script>
Polymer({
});
</script>
</polymer-element>

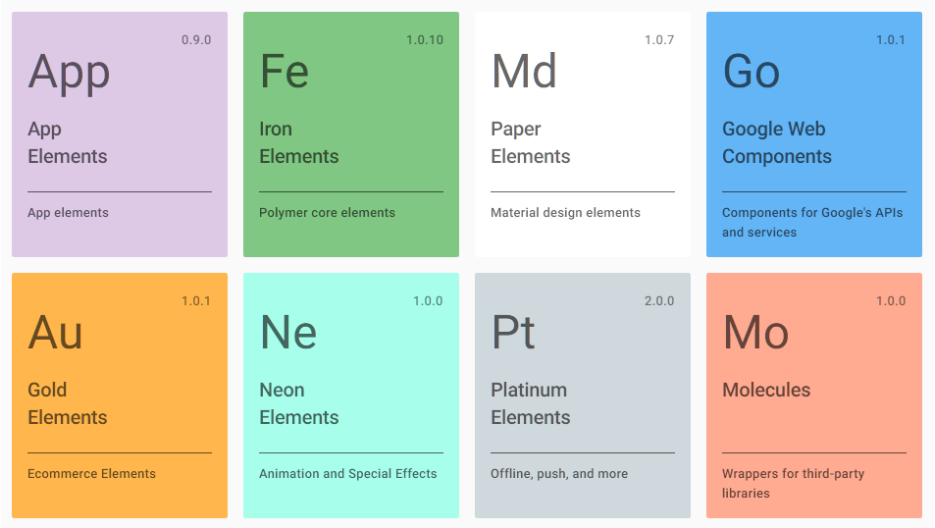
Capturas

Componentes Web

Polymer cuenta con un catálogo de elementos que permiten crear aplicaciones con patrones de diseño basados en Material Design. Comprende un set de elementos numeroso, del cual hemos extraído 12 con algunas de sus propiedades para nuestra propuesta.

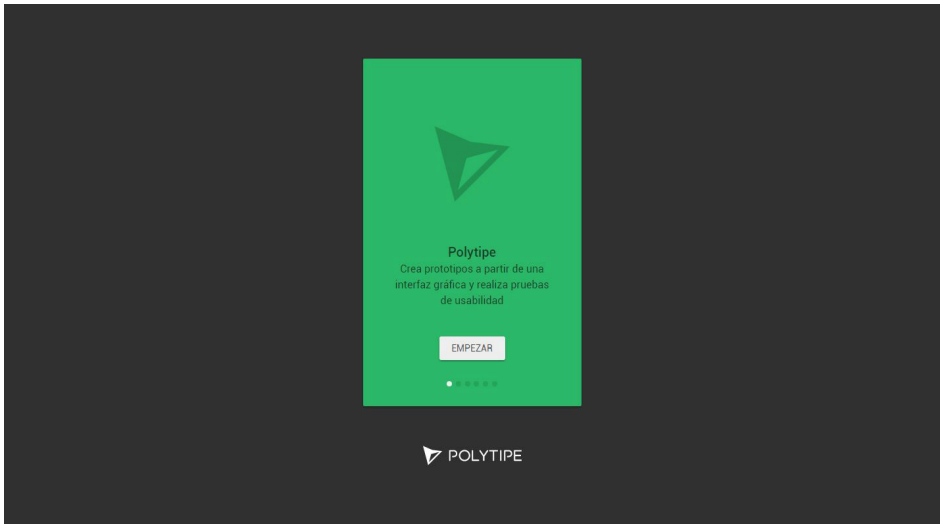


Para ver la lista completa de elementos, visita https://elements.polymer-project.org



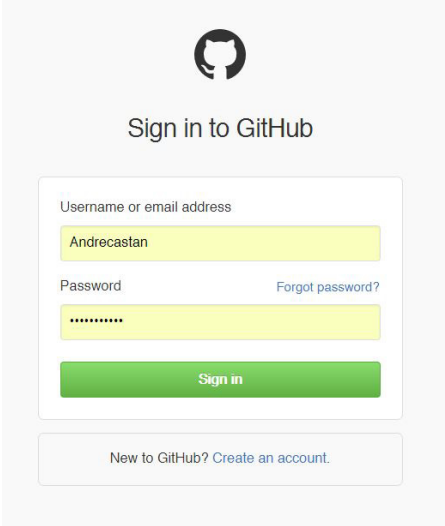
Desarrollo de Polytipe

El desarrollo de Polytipe ha evolucionado en relación con la propuesta que se dio inicialmente. Para poder cumplir con las determinantes y mejorar la experiencia de los usuarios, se han implementado secciones, cambios en en la interfaz y el flujo de interacción.



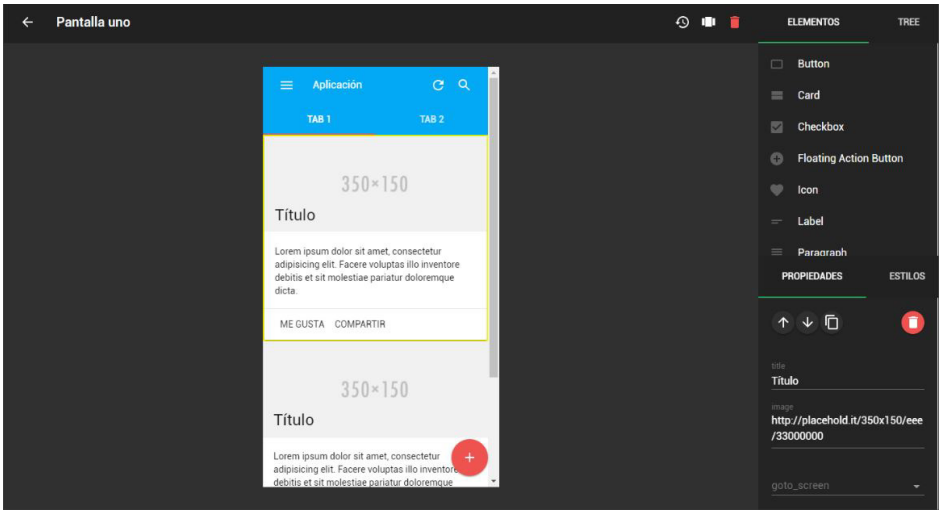
Inicio

En la propuesta inicial, el usuario ingresaba y tenía la opción de crear un proyecto nuevo o de abrir un archivo guardado localmente para continuar el proyecto. Sin embargo, esta propuesta presentaba algunas dificultades técnicas en el desarrollo y decidimos implementar Firebase para que el usuario inicie sesión con su cuenta de GitHub teniendo en cuenta que una de nuestras determinantes es el versionamiento en git, ahora el usuario no guarda sus archivos localmente sino que al ingresar a Polytipe, crea un repositorio nuevo en su cuenta donde puede manejar todos sus proyectos.



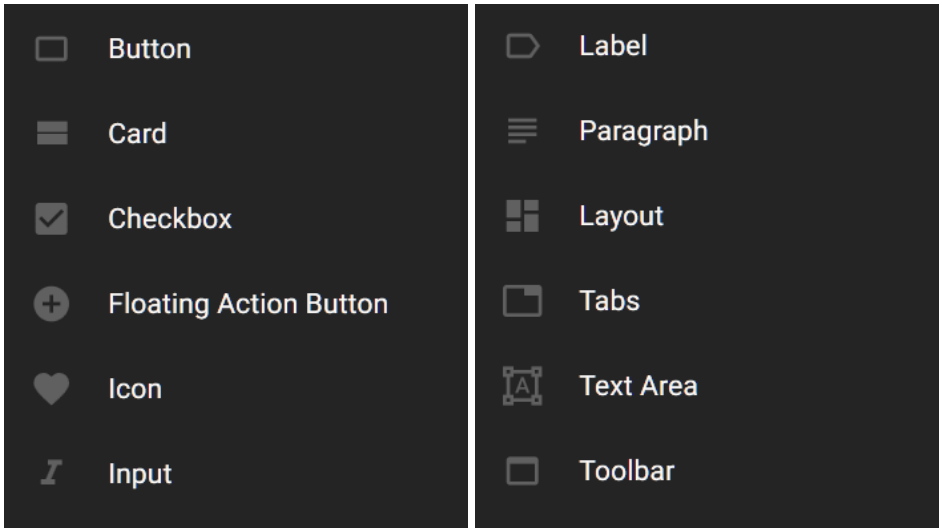
Área de trabajo

Anteriormente, el área de edición de una pantalla constaba de un canvas y un botón de acción flotante del cual se desprendía los elementos, el usuario debía usar drag and drop para acomodarlos y al hacer click sobre ellos aparecían las opciones de edición. Sin embargo ésta dinámica nos generó problemas técnicos en el desarrollo del código y decidimos implementar el panel de herramientas.



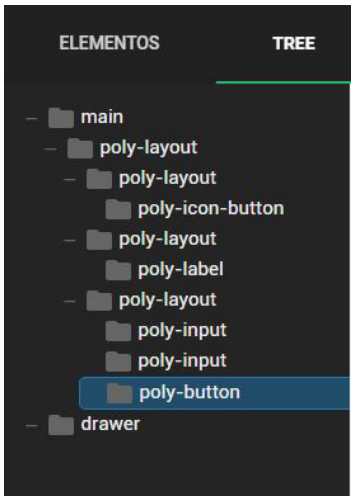
Ahora, una vez el usuario cree una pantalla nueva, esto es lo que verá en el navegador. El área de trabajo consta con un canvas donde se creará el prototipo de la aplicación que estén desarrollando. A la derecha se encuentra el panel de herramientas donde se encuentran los componentes junto con sus propiedades y estilos.

Elementos



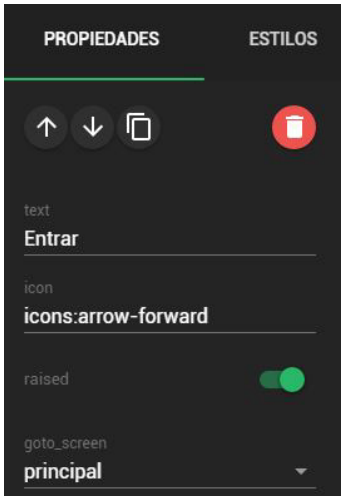
Ordenamiento

A diferencia de la propuesta anterior, los elementos no se organizan en el canvas usando drag and drop, en cambio implementamos el uso de layouts, lo que permite distribuir los elementos de una mejor manera y evitar que sean ubicados al azar. La sección del ordenamiento (Tree), permite ver y reorganizar de una manera sencilla los elementos del prototipo y garantiza que el usuario pueda seleccionar el elemento que desea para poderlo editar sin generar confusiones.



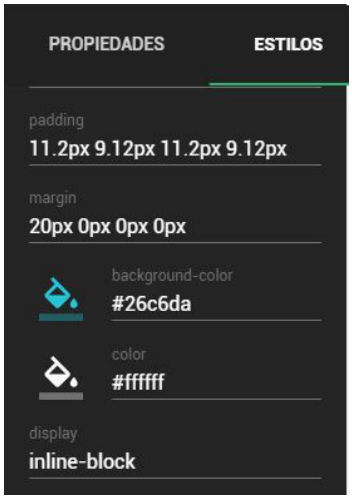
Propiedades

La sección de propiedades permite cambiar como su nombre lo indica, las propiedades de cada elemento (número de pestañas, tipo de icono, etiquetas…). En la propuesta anterior estos cambios se generaban al hacer click al elemento dentro del canvas y ahí mismo salían las herramientas de edición. Sin embargo, cuando un elemento estaba dentro de otro como es el caso de los layouts se generaba una confusión en la selección.



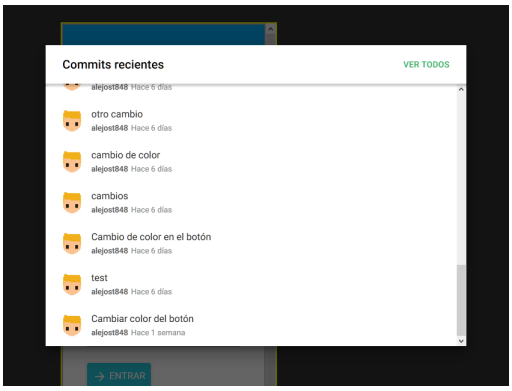
Estilos

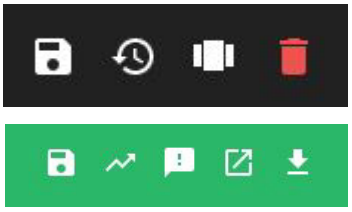
La sección de estilos permite agregar estilos CSS a los elementos como background-color, margin, padding etc. Teniendo en cuenta una de las determinantes que es el uso de Material Design, los colores que puede seleccionar el usuario para los elementos del prototipo, son aquellos que se encuentran en la paleta de color de Material Design.



Guardar el código

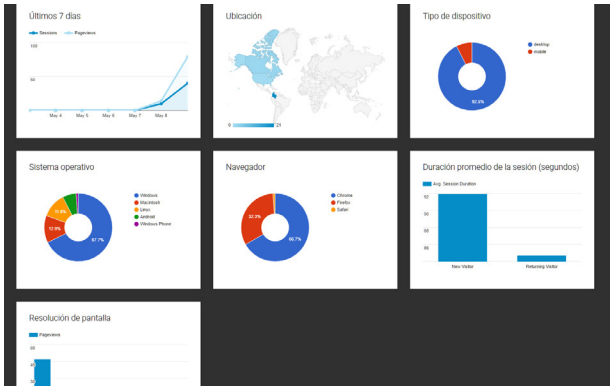
Usando el API de GitHub, cada usuario al ingresar a Polytipe genera un repositorio llamado polytipe-projects, en este repositorio se guardan los distintos proyectos del usuario. Cada vez que se hace un cambio en el proyecto, el usuario puede hacer commit y el código se guarda en el repositorio.





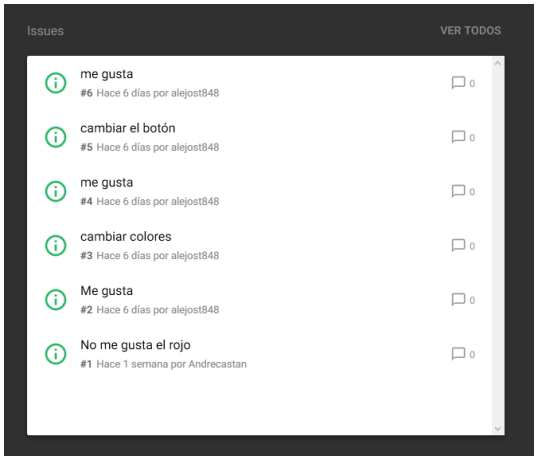
Obtener métricas

Utilizando Google Analytics, Polytipe obtiene datos de los usuarios que están visitando los prototipos y que están siendo parte de las pruebas. Estos datos corresponden a métricas de usabilidad que incluye los dispositivos utilizados, navegadores y sistemas operativos, así como el lugar de su visita y el tiempo que emplea en el prototipo, entre otros.



Recibir feedback

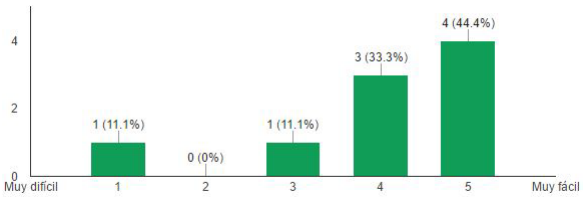
Los usuarios pueden obtener feedback directamente de los usuarios a través de la integración con Github Issues y recibir así información valiosa acerca de su experiencia con los prototipos que se generen.



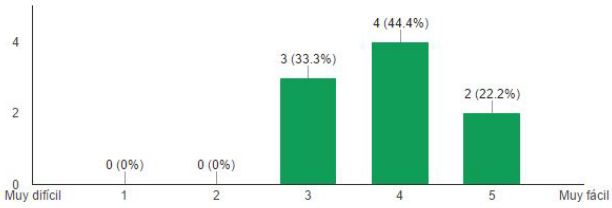
Pruebas de usuario

Se realizaron pruebas a nueve usuarios con el objetivo de validar el flujo de interacción de la aplicación. Para esto, se les pidió a los usuarios que navegaran a través de polytipe.me y crearan un repositorio con un proyecto, añadir un colaborador y exportar el código.

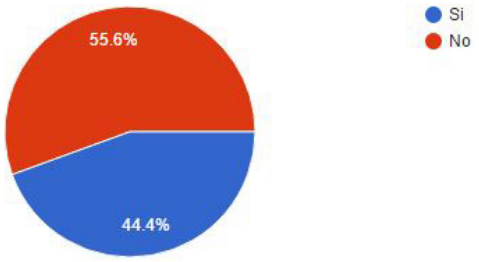
¿Qué tan fácil fue crear un proyecto nuevo?



¿Qué tan fácil fue cambiar propiedades y/o estilos de los elementos?

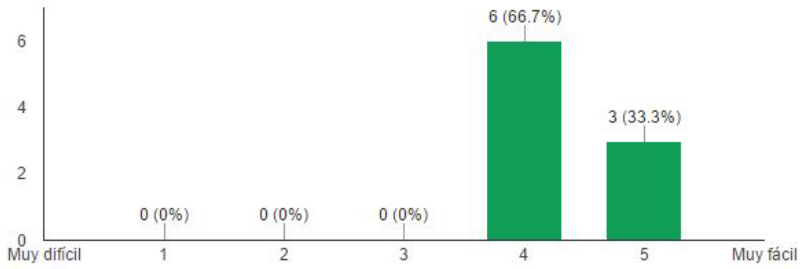


¿Fue posible agregar a un colaborador?



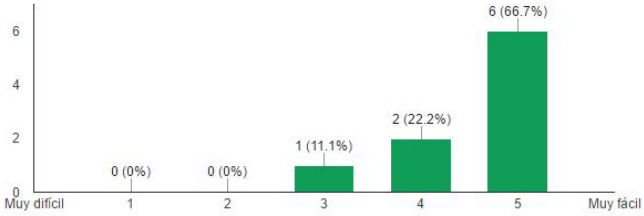
Nos dimos cuenta que agregar colaboradores fue una tarea que se les dificultó realizar debido a la ubicación del botón.

¿Qué tan fácil fue agregar elementos a la pantalla?

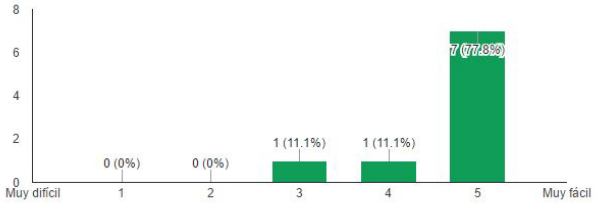


Sin embargo, lograron realizar tareas clave como la creación de proyectos, edición de elementos y exportación de código con éxito.

¿Qué tan fácil fue guardar los cambios en la pantalla?

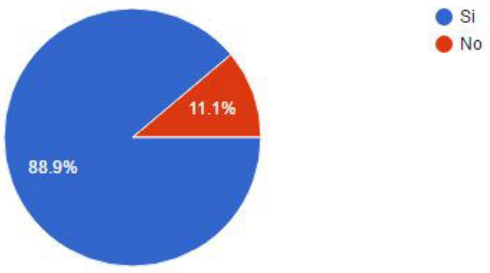


¿Qué tan fácil fue encontrar el icono de previsualización?



¿Fue posible realizar la previsualización de su proyecto?

¿Fue posible descargar el código?



Modelo de negocio

<table>
<tr><th>Socios clave</th><th>Actividades clave</th><th>Propuesta de valor</th><th>Relación clientes</th><th>Segmentos Cliente</th></tr>
<tr><td rowspan="3">GitHub
Polymer
Comunidades de desarrolladores
Firebase</td><td>Diseño
Programación
Mantenimiento de la página web</td><td rowspan="3">Plataforma de prototipado web que une al diseñador y al programador con el fin de lograr un resultado más coherente con la propuesta inicial.

Optimiza el tiempo de programación haciendo uso del prototipado y obteniendo métricas de usabilidad.</td><td>Foros
Reference
Issues tracking</td><td rowspan="3">Desarrolladores web
Diseñadores web
Compañías de desarrollo web</td></tr>
<tr><th>Recursos clave</th><th>Canales</th></tr>
<tr><td>Dominio
- polytipe.me
Base de datos
- Firebase</td><td>Foros
Redes sociales
Página web
Showcase</td></tr>
<tr><th colspan="2">Costo de la estructura</th><th colspan="3">Ingresos</th></tr>
<tr><td>Dominio</td><td>10 USD/año</td><td colspan="3" rowspan="2">Donaciones
Planes corporativos / Capacitaciones / Talleres</td></tr>
<tr><td>Hosting</td><td>60 USD/año</td></tr>
</table>

Viabilidad Técnica

Polytipe hace uso de herramientas como Polymer que cuenta con un catálogo de elementos que permiten seguir los lineamientos de Material Design y facilita la programación de la aplicación.

Actualmente, la comunidad ha desarrollado cientos de componentes web que están disponibles al público a través de la página customelements.io.

Así mismo, Polytipe ha sido desarrollada haciendo uso del API de GitHub que permite crear y administrar repositorios donde se generan los proyectos creados por el usuario.

Adicionalmente, la aplicación hace uso de Firebase que permite vincular Polytipe a la cuenta de GitHub del usuario, evitando la implementación de bases de datos que representan un gasto en el manejo de hosting.

Viabilidad Económica

Calculamos que en nuestro primer año Polytipe tendrá un promedio de 3000 USD como ingreso y contará con aproximadamente 2000 usuarios contando con las empresas de desarrollo web en Colombia y algunos usuarios independientes.

Tamaño de mercado
En Colombia, según un estudio realizado en el año 2012 por la Federación Colombiana de la Industria del Software y Tecnologías de la Información-FEDESOFT, de una muestra de 1120 empresas, el 48.57% se dedica al desarrollo de aplicaciones web, teniendo como principales líneas de negocio a nivel nacional el desarrollo a la medida, desarrollo de aplicaciones web y soporte y mantenimiento de software (Fedesoft, 2012).

Modelo de monetización
Polymer es una la aplicación open source que permite lograr un mayor crecimiento con la ayuda de la comunidad a nivel global. Por ésta razón, la plataforma es gratuita y los usuarios tienen la posibilidad de donar si desean, la cantidad que crean pertinente para apoyar el proyecto.

Sin embargo, Polytipe ofrece los servicios de asesoría y capacitación, para aquellos usuarios que lo requieran. Estos servicios tienen un costo dependiendo de los paquetes que adquieran los usuarios y de si es un individuo o una empresa.

Referencias Bibliográficas

Centers for Medicare & Medicaid Services (CMS) Office of Information Service (2008). Selecting a development approach. United States Department of Health and Human Services (HHS). Retrieved August 30, 2015, from www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf

Neill, Colin J. & Laplante, Phillip A. (November 2003), Requirements Engineering: The State of the Practice, IEEE Computer Society. Retrieved August 30, 2015 from www.panda.sys.t.u-tokyo.ac.jp/kushiro/ReferencePaper/Requirements%20Elicitation/01241365.pdf

United States, Navy Mathematical Computing Advisory Panel (29 June 1956), Symposium on advanced programming methods for digital computers, [Washington, D.C.]: Office of Naval Research, Dept. of the Navy, OCLC 10794738

Royce, Winston (1970), Managing the Development of Large Software Systems (PDF), Proceedings of IEEE WESCON 26 (August): 1–9

Cusumano, Michael A. & Smith, Stanley (1995) Beyond the Waterfall: Software Development at Microsoft, p. 3 Retrieved August 30, 2015 from dspace.mit.edu/bitstream/handle/1721.1/2593/SWP-3844-33836288.pdf?sequence=1

Gottesdiener, Ellen (August 1995), Rad Realities: Beyond The Hype To How Rad Really Works, Application Development Trends. Retrieved August 30, 2015 from www.ebgconsulting.-com/Pubs/Articles/RAD_Realities_Beyond_the_Hype_Gottesdiener.pdf

Hirschberg, Morton A. (1998) Rapid Application Development (RAD): A Brief Overview, U.S. Army Research Laboratory. Retrieved August 30, 2015 from www.csiac.org/sites/default/files/journal_files/1998_03_01_RapidApplicationDevelopment.pdf

Davis, Bersoff, Comer (1988), A Strategy for Comparing Alternative Software Development Life Cycle Models (PDF), Proceedings of IEEE TRANSACTION ON SOFTWARE ENGINEERING 14 (October): 1–2

Agile Manifesto (2001) Manifesto for Agile Software Development. Website. Retrieved August 30, 2015 from www.agilemanifesto.org/

Lillington, Marcus (2015) Does Agile methodology lead to poor interface design? Web article. Retrieved August 30, 2015 from headscape.co.uk/design/agile-methodology-design/

Hazaël-Massieux, Dominique (2013) Closing The Gap With Native Apps, W3C, Web article. Retrieved August 30, 2015 from www.w3.org/blog/2013/03/closing-the-gap-with-native-ap/

Bleigh, Michael (2014) The Future of the Web (According to Google), Divshot, Web article. Retrieved August 30, 2015 from
divshot.com/blog/opinion/the-future-of-the-web-according-to-google/

comScore, Inc (2014) The U.S. Mobile App Report Retrieved August 30, 2015 from
www.comscore.com/Insights/Presentations-and-Whitepapers/2014/The-US-Mobile-App-Report

Fedesoft (2012) Estudio de la caracterización de productos y servicios de la industria de software y servicios asociados. Ministerio de Tecnologías de Información y Telecomunicaciones. Colombia. Tomado en Septiembre 1, 2015 de
www.fiti.gov.co/Images/Recursos/estudiocifrassectorsw2012.pdf

McLaughlin (2015) Agile Methodologies. Website. Retrieved September 28, 2015 from https://www.versionone.com/agile-101/agile-methodologies/
Getto, Guiseppe (2015) Cover Letters. User Experience Design. Website. Retrieved October 6, 2015 from http://www.guiseppegetto.com/engl7766ux/cover-letters/
Treder (2015) The User Experience Guide Book For Product Managers. Page 24. Retrieved October 6, 2015 from
http://studio.uxpin.com/ebooks/user-experience-for-product-managers/

UXPin (2015) Getting Started With UX Design Process & Documentation. Page 10. Retrieved October 6, 2015 from
http://studio.uxpin.com/ebooks/primer-ux-design-process-documentation

Introduction - Material design - Google design guidelines. (2016). Google design guidelines. Retrieved 9 May 2016, from
https://www.google.com/design/spec/material-design/introduction.html

Proyecto de Grado

Mejoramiento de procesos de producción en empresas creadoras de aplicaciones web mediante el uso de metodologías ágiles de desarrollo

Andrea Castañeda
Alejandro Sanclemente