Desgrabación de la Presentación en el DrupalCon Latin America

Poner en pie una fábrica de DurableDrupal (Drupal Duradero) en base de un proceso Lean reutilizable

Ya que el sonido de la grabación de mi presentación quedó muy bajo en volumen, quiero presentar la desgrabación con la esperanza de que mi mensaje llegue a la mayor cantidad de personas posible.

El video de la presentación en sí se encuentra aquí: https://www.youtube.com/watch?v=bNbkBvtQ8Z0

Los diapositivas: http://awebfactory.com/drupalcon2015lean/#/

Para cada diapositiva, incluyo a continuación el link al slide y el texto correspondiente del video (algo desarrollado y corregido).

En algunos casos hay correcciones inevitables o he extendido el texto para su mejor comprensión. También he traducido diapositivas importantes y agregado algunos comentarios para incluir puntos de suma importancia que no fueron mencionados en la presentación por falta de tiempo, como el Kanban y sus características.

Presentación

http://awebfactory.com/drupalcon2015lean/#/

La base material para el proceso Lean y para cualquier desarrollo de software es por un lado la disciplina de diseño [[Lean UX]] y por otro lado el DevOps. Esto es medio raro que estamos hablando de proceso en la pista de DevOps pero la razón es... vamos a ver bien qué es Lean. En esta conferencia hubo bastantes charlas excelentes sobre Lean:

- Drupal 8 CMI on Managed Workflow - https://latinamerica2015.drupal.org/session/drupal-8-cmi-managed-workflow
- DevOps, por donde comenzar? - https://latinamerica2015.drupal.org/session/devops-por-donde-comenzar
- Best practices for Continuous Deployment with Drupal. - https://latinamerica2015.drupal.org/session/best-practices-continuous-deployment-drupal
- I Want it All and I Want it Now: Configuration Management and CI - https://latinamerica2015.drupal.org/session/i-want-it-all-and-i-want-it-now-configuration-management-and-ci

Yo quisiera compartir algunos conceptos que nos puede salvar del fracaso. Todos los éxitos vienen de haber previamente fracasado. He fracasado yo suficientemente como para ser experto en el proceso. Así que... vamos a ver de qué se trata.

Lo primero que quiero hacer es... necesitamos un feeling de qué es el proceso Lean, que

vamos a apreciar desde una lectura. Pero primero me presento y veamos unos conceptos básicos del Lean.

whoami

quien soy https://www.drupal.org/user/36006

Yo soy [Victor Kane (@victorkane)](https://www.drupal.org/user/36006), usuario 36006, en Drupal.org están todos mis datos. Como director y duenño de AWebFactory.com por [más de ocho años](http://awebfactory.com/node/96) he estado desarrollando sitios en Drupal desde 2006. Participé en mi primer DrupalCon en 2007 en Barcelona. Trabajo en la construcción de sitios y aplicaciones web, y como mentor para formar equipos internos en las empresas y organizaciones para que puedan producir su producto y dedicarse a la mejora continua de su proceso interno.

DurableDrupal Lean Process Factory (Introducción)

Fábrica de proceso Lean para el DrupalDuradero
http://awebfactory.com/drupalcon2015lean/#/2

La fábrica de proceso Lean DurableDrupal (Drupal perdurable) se trata de herramientas y procesos reutilizables, adaptados a la organización que los usa, en evolución, con el fin de derrotar a la metodología cascada (waterfall) definitivamente y de garantizar la entrega de valor al cliente.

Bueno, es una frase bastante lleno de significado. Si lo comprendieramos y supieramos como obrar así, ya podríamos terminar la sesión ahora mismo y ir a tomar una cerveza.

That's a mighty phrase up there. If we understood it and knew how to get it done we could end this session right now and go have a beer. Pero la cuestión es, con la Fábrica de Proceso Lean, ¿como se salva de los rockstars, de las divas y de la metodología cascada (waterfall) para lograr que todo el equipo esté trabajando en paralelo? ¿Cómo se logra trabajar codo a codo con el cliente sin parar hasta que se verifica el hecho de que se haya entregado valor verdadero?

Vamos a ver, lo más rápido que podamos porque la verdad que hy mucho material para ver acá, entonces va a ser medio escueto. Voy a dejar el link para los slides y también para un video que forma parte de la presentación, cualquier pregunta, si necesitan interrumpir porque queda muy borroso algún concepto, interrumpenme. Si no, esperamos hasta el final para ver las preguntas.

Vamos a hablar sobre Kanban, hay que entender algunas características que tiene. Suele ser utilizado por el Scrum, pero tiene algunas características aparte (que lo distingue del scrum como metodología).

También vamos a hablar sobre la Visión y la Incepción del proyecto; sobre el Team Kickoff, o sea, la patada inicial de los proyectos, la primera reunión, el inicio del proyecto; sobre el flujo de trabajo y el pradigma de todo en código; después sobre el provisionamiento y el DevOps, no ya para los miembros del equipo, sino del servidor de desarrollo e integración, y el servidor de testeo (staging) y el de producción, es decir, sobre la automatización del deployment (despliegue) como se habló en los otros talleres

(sobre DevOps); y la cuestión de la validación de los usuarios que es tan importante.

Process Overview: Waterfall -> Agile

Vista de conjunto del proceso: De cascada al ágil
http://awebfactory.com/drupalcon2015lean/#/2/1

(Tomado del artículo de 2007 de Desirée Sy: Adapting Usability Investigations for Agile User-centered Design http://uxpajournal.org/wp-content/uploads/pdf/agile-ucd.pdf)

El artículo explica como Cascada y aun Agil conduce a la creación de silos aislados que separan las distintas disciplinas, donde los entregables se elaboran y se pasan a otros sin que existiese la más mínima colaboración entre sí o retroalimentación. Agil es una mejora en comparación con Cascada ya que el abismo que separa la colección de los requirimientos del testeo final es mucho más corto.

Esto fue... son conceptos. El proceso de Waterfall (Cáscada) es el enemigo. Yo supongo que si soy ingeniero civil o estructural voy a repetir recetas probadas de A a Z, no voy a inventar la rueda, no me voy a preguntar desde cero cómo se construye una puente y por ahí (aunque no necesariamente sea así) voy a usar la metodología cáscada. Pero como éste tiene características especiales el método de cáscada no funciona. Sin embargo es el método que todo el mundo utiliza por defecto, a veces inconcientemente cuando algunas personas crean que está utilizando algo mucho más ágil, están usando waterfall. Y vamos a ver exactamente por qué. Eso es lo que yo quiero compartir.

El waterfall divide las disciplinas en varias: Captura de requerimientos, análisis y diseño, la implementación en código, el testing y control de calidad, y la liberación y entrega como despliegue en el destino final. Y hay otras disciplinas también. Entonces, qué es lo que se hace? Primero se efectúan la captura de los requerimientos, después se prioritizan los requerimientos y se concreta la planificación, se trabaja en la gestión de la configuración (Configuration Management) y los distintos entornos de desarrollo, testeo y producción y el proceso de entrega, se va definiendo la arquitectura candidata, se implementa, y luego se testea. Y luego se entrega y luego se lanza. Entre cada paso en que especialistas de la disciplina determinada ejecuta su labor, se recibe una entrada (handoff) y luego al completar las tareas se entrega la salida a la próxima (handoff). Este proceso de waterfall, una veritable cáscada o serie de cascádas en fila de entregables hecho cada uno en turno, es el enemigo, conduce al fracaso, y vamos a ver por qué.

El proceso ágil trata de mejorar esto. ¿Cuál es el problema acá? En los años 80 un estudio de la Fuerza Aerea norteamericana (Development of an Environment for Software Reliability - http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA256609) descubrió que el uso del método cascada (waterfall) resultaba en un alto porcentaje de fallas. Primero en cualquier proyecto hay un 40% mínimo de cambios en requerimientos, y si se hace todo el analisis y diseño y luego la implementación, entonces el resultado está muy lejos de los requirimientos reales. Primero, Waterfall no toma en cuenta los cambios y no es flexible. Y segundo hay un abismo entre las entrevistas y la captura de los requirimientos (superado por la participación directa del cliente en el equipo en Agile y aun más en Lean), por un lado, y los resultados que el cliente termina recibiendo en las "entregas" (superado por las entregas frecuentes y integración continua de Lean DevOps). "Yo no

pedí eso", es lo que se escucha por parte del cliente, que no ha podido tener oportunidad de ofrecer retroalimentación durante el largo período de construcción, que a veces puede durar 6 meses o un año con esta metodología.

Y es más frecuente de lo que parece. Hay muchos equipos con "Scrum Masters" que terminan haciendo exactamente lo mismo, marginando a DevOps y a UX y diseño y el cliente en la práctica durante largos períodos.

El problema es que cada disciplina corresponde a toda una fase aislada de los demás, y no hay comunicación entre las distintas disciplinas. Por ejemplo, muchas veces se hace captura de requirimientos y la diseñadora gráfica "hace el diseño" y como resultado toma muchas decisiones de UX, funcionalidad y aun de arquitectura, toda en forma aislada y sin discusión con nadie, finalmente ofreciendo dos o tres "alternativas" monolíticas. Cuando lo que queremos y recibimos con Lean es que todos participan colectiva y conscientemente en paralelo, compartiendo conocimientos y ofreciendo retroalimentación constantemente. Ayer vimos en la presentación de Larry Garfield en la que dijo ("Diseñar Sistemas, no Páginas" - https://latinamerica2015.drupal.org/session/design-systems-and-drupal), como los diseñadores participan en el análisis de los contenidos estructurados, y en el diseño de componentes que refleja esa estructura y que son reutilizables una y otra vez; y que pueden ser utilizados en distintas maneras.

Frente a las críticas en numerosos informes sobre el alto índice de fracaso que como resultado de utilizar la metodología Waterfall y la práctica de exigir burocráticamente estimaciones de tiempo y esfuerzo sin conocer lo suficiente sobre un proyecto determinado, surgen las metodologías incremental e iterativas.

El Rational Unified Process (http://en.wikipedia.org/wiki/Rational_Unified_Process) (más tarde comprado por IBM) primero, y el Agile (http://en.wikipedia.org/wiki/Agile_software_development) después, como se ve en el diagrama de Desirée Sy de 2007, reemplazan a la cáscada con una secuencia de cascaditas, que por lo menos ofrecen más retroalimentación porque puede haber retroalimentación en cada entrega. Está fuera del alcance de esta presentación hacer una definición pormenorizado del agil en este momento, es importante tomar en cuenta los principios valiosos que se expresan en el Manifiesto Agil (http://www.agilemanifesto.org/) que este autor firmó [en mediados de 2006] (http://www.agilemanifesto.org/sign/display.cgi?ms=000000067); lo importante que tenemos que marcar en este momento es su falta de idoneidad todavía para las aplicaciones web.

Porque el Agil es la nueva Cáscada. Agile is the new Waterfall.

Porque encierra una serie de problemas que es importante ver: dentro de la cascadita, o secuencia de actuación de disciplinas en forma aislada, se sigue practicando "handoff", o sea, un flujo de trabajo secuencial y no en paralelo, encerrando entradas y salidas entre estados aislados que no retroalimentan nada entre sí. El enemigo en el proceso sigue siendo no trabajar en conjunto.

Si se trabaja en conjunto y todo el mundo comparte una conciencia de cómo se está llevando adelante el trabajo en su conjunto, entonces nos va bien, porque hay retroalimentación contínua y mutual, y un intercambio constante de información.

Ver el corto extracto del libro de Gothelf (http://awebfactory.com/drupalcon2015lean/files/lavidaenlean.pdf), mencionado hacia abajo en The goal of cross-collaboration (El meta de colaboración multi-disciplinaria (http://awebfactory.com/drupalcon2015lean/#/5/15) to sample the atmosphere of what this actually looks and fells like in actual practice.

Un punto interesante es, dado la firme conciencia del asunto que se comparte, tiene toda la confianza del mundo de que en dos días tendrán un prototipo hecho.

En parte es gracias de la Guía de Estilos Viviente que permite que el desarrollador empiece a construir sin tener que esperar un diseño completo de la "página". Todos pueden trabajar en conjunto, el desarrollo, el diseño, el test de aceptación y la documentación, por ejemplo. Y gracias a la conciencia del DevOps del trabajo en curso, no va a haber ninguna confusión sobre como todo el mundo tenga acceso a la guía o cualquier otra cosa, que versión era, si es lo mismo en mi máquina que en la tuya. Entendemos esto, ¿no es cierto? Todo el mundo sabe qué está pasando. They all get it. Y cuentan con las herramientas, en el conjunto de las disciplinas, que necesitan.

Better Process (2007)

El mejor proceso (2007) http://awebfactory.com/drupalcon2015lean/#/2/2

En el artículo Sy y Miller continúan describiendo en detalle su idea de una integración productiva del Agil con el diseño cenrado en el usuario por medio de una técnica llamada Ciclo Cero (a veces llamada Sprint 0 o Sprints Escalonados), en lo cual la actividad del diseño tiene lugar un sprint adelantado del desarrollo. El trabajo se diseña y se valida durante el sprint de diseño y luego el diseño se entrega a la pista de implementación. A pesar de que Sy y Miller siempre abogaban por una fuerte colaboración entre los diseñadores y los desarrolladores, muchos equipos se no comprendieron este punto crítico y han hecho flujos de trabajo en que los diseñadores y los desarrolladores solo se comunican por medio de las entregas, creando un especie de proceso mini-cascada. Pero por más que se aliente la colaboración multi-disciplinaria, con los sprints escalonados se da demasiado facilmente el caso en que el equipo entero nunca se enfoca sobre lo mismo en ningún momento. Entonces el equipo no se beneficia de la colaboración porque los distintos disciplinas se enfocan siempre en asuntos distintos.

Esta sugerencia, que se remonta a 2007, explica cómo se puede mejorar el ágil para las aplicaciones y sitios web, para que no sea la nueva cascada en el desarrollo de sitios y aplicaciones web. Si bien, lamentablemente, sigue con la idea de entrega, de entradas y salidas entre una secuencia de silos aislados: Yo llevo a cabo un trabajo y lo entrego a vos, vos hacés un trabajo y lo entregás a otro especialista, lo cual ocasiona mucho deroche. Lo que se hace es, tomando en cuenta que el ágil es una cascadita, lo que propone Desirée Sy en 2007 es que se da comienzo al proyecto con una reunión entre todos --Se acaba eso de "no hace falta contratar diseñadores hasta más tarde" o "solo al principio del proyecto, por una semana nada más, que hagan el diseño (sic) y listo" (no tanto diseñadores como DevOps tienen que formar parte del equipo)-- en que se planifica y se comprende en conjunto todo el trabajo. Y después, en el primer sprint, mientras los desarrolladores trabajan sobre asuntos de alto riesgo en el proyecto, que tienen que ver con el back-end y no tiene nada que ver con la funcionalidad ni con la usabilidad ni con

el diseño del interfaz del usuario, los diseñadores están haciendo el diseño para el sprint siguiente. Mientras se está haciendo la implementación de este diseño, se testea lo que está implementado en el ciclo anterior y se va diseñando el ciclo o sprint siguiente, y se captura información del cliente para otro ciclo después de éste. Y así succesivamente. Son los ciclos escalonados: es mejor, hay menos deroche. Pero no es la mejor solución de todos porque la gente igual está haciendo "handoff". No está trabajando de conjunto en paralelo. Unos están en una cosa, y viene alguien de la otra pista e interrumpe. Una de las reglas del Lean es no hacer varias tareas a la vez, saltando de una a otra.

Still better process: Design + Dev

Proceso aun mejor: Diseño + Desarrollo http://awebfactory.com/drupalcon2015lean/#/2/3

Vale subrayar la cita de Gothelf (Gothelf, Jeff (2013-02-22). Lean UX: Applying Lean Principles to Improve User Experience (p. 97). O'Reilly Media.):

> Muchos equipos han interpretado mal el modelo de sprints escalonados de Sy y Miller, quienes siempre abogaban por una fuerte colaboración entre los diseñadores y los desarroladores, tanto durante los sprints de diseño como los de desarrollo. Muchos equipos, sin embargo, no han captado este punto crítico, y en su lugar han creado flujos de trabajos con handoffs (entradas y salidas, entregas), creando un especie de proceso mini-cascada.

También ver Beyond Staggered Sprints: How TheLadders.com Integrated UX into Agil (http://johnnyholland.org/2010/10/beyond-staggered-sprints-how-theladders-com-integrated-ux-into-agile/)

Por eso, el mejor proceso es este.

En primer lugar los “themes” (temas) unifican una serie de sprints alrededor de uno de los features (rasgos de alto nivel, como “página principal”, “landing page” – página de aterrizaje publicitado para lograr conversiones de marketing, captando por ejemplo el correo electrónico de los visitantes; o “comentarios utilizando disqus”. De este modo, los diseñadores tienen más de un solo sprint para organizarse, y pueden gozar del tiempo necesario para ser creativos mientras estén en la misma página que los desarrolladores de front-end y back-end.

Luego, se hace la reunión al principio de todo, como el ciclo 0, con todos presentes, para ver la visión del proyecto, para hacer brainstorming (tormenta de cerebros), esbozar algunos diseños, que se necesita en la página principal, y aquí y allá, y después en base de esto, vamos a ver en detalle cuales son los artefactos que se necesitan para esto. Este primer reunión de conjunto puede durar unas horas o una semana ya que es posible que, para llegar a fondo para validar, por ejemplo, el dominio del problema en el proyecto (porque vamos a hablar de problema, mercado y producto) tengamos que "salir del edificio" para validar el problema con el supuesto mercado. Sin escribir siquiera una sola línea de código. Porque el Lean trae toda la experiencia de los 90% o más de los startups (emprendimientos) que fracasan. Los Angry Birds que nunca vimos. La lección es que no hay que escribir nada de código hasta que se haya validado el problema y el mercado. Y aun el producto que construimos en forma iterativo e incremental, tiene que ser sujeto a pruebas no solo de funcionalidad,, sino también de si otorga el valor esperado.

Cada proyecto debe considerarse como un startup, y no solo un proyecto. Entonces el método desde el principio es, luego de tratar de definir el problema, directamente salir a validar esa definición, con los mismos usuarios que se imaginan pidiendo a gritos una solución. ¿Cómo se va a saber? Bueno, saliendo a investigar a ver si es cierto.

Una vez que se tiene un hipótesis, y una visión de qué producto mínimo podría ser capaz de validar ese hipótesis, se puede hacer una reunión enseguida para efectuar la planificación. Y allí se baja todo a una serie de user stories. El theme son 3 corridas o sprints si se usa scrum, y la idea es que el theme sea un rasgo general de funcionalidad, como ya explicamos: la página principal; o un shopping cart. ¿Se entiende?

Y lo genial de este esquema es que tiene que haber validación por los usuarios por lo menos dos veces por cada sprint, es decir nunca se va más de cinco días sin contar con la retroalimentación del usuario y del cliente. En este enfoque no hay nada de handoff sino entregas constantes de todos en paralelo. En estos primeros dos reuniones de cada corrida está trabajando todo el mundo. Después como vimos en el extracto del libro de Gothelf que leimos, todo el mundo trabaja en paralelo; se colaboran entre sí cuando haga falta, juntos, y se van trabajando cada uno con su parte de un mismo trabajo (el tema) sobre el cual todos comparten conciencia. Y al final de cada sprint se va liberando una versión del MVP (producto mínimo viable capaz de llevar a cabo una prueba para ver si el hipótesis de mercado, problema y producto sea cierto). No va a ser el MVP con lo cual se considera completada el proyecto, sino una serie de MVPs, uno para cada sprint.

Best Process: Design + Dev + DevOps

El mejor proceso: Diseño + Dev + DevOps
http://awebfactory.com/drupalcon2015lean/#/2/4

Lo que nosotros queremos agregar aquí hoy a este conjunto es el concepto de que así como es crítico la colaboración multi-disciplinaria entre diseño y desarrollo, es tan crítico también que haya colaboración entre éstos y DevOps también: entre Diseño, Desarrollo y DevOps. Esto es cierto para el aprovisionamiento para que que cada miembro cuente con su entorno para poder trabajar en paralelo, y es cierto para la entrega e integración continua, para el testeo y sus entornos, para las liberaciones y el deployment (despliegue?). Todo debe fluir siempre sin interrupciones.

Waste Avoided by DurableDrupal Lean

El deroche evitado por el DurableDrupal Lean
http://awebfactory.com/drupalcon2015lean/#/2/5

1. Trabajo sin terminar dejado pudriendose en silos aislados
2. El crecimiento del alcance sin validar su impacto o valor real
3. Silos aislados que sigue cada uno inventando la rueda una y otra vez
4. Entradas y salidas que no se entienden y sobre lo cual no hay retroalimentación, que cronicamente interrumpen el flujo de trabajo
5. Demoras debido al aislamiento de cada silo (area de trabajo de un especialista determinado) y la falta de determinados especialistas en el equipo, o la falta de

comunicación si están presentes.

6. Constantes cambios de tarea impiden que se pueda enfocar sobre cualquier tarea, y la rotura consecuente del flujo de trabajo
7. Muchos defectos debido a esta falta de concentración y debido a la falta de intercambio de conocimientos

Artículo de fondo: How to Manage the "7 Wastes" of Agile Software Development https://www.scrumalliance.org/community/articles/2013/september/how-to-manage-the-7-wastes%E2%80%9D-of-agile-software-deve#sthash.AQrw03H3.dpuf

Esto es una lista de los tipos de deroche que se evitan utilizando el proceso Lean. Si no, está comprobado que quedan un montón de trabajo pudriéndose en los silos aislados. Si yo estoy trabajando sobre tres cosas, no puedo concentrarme en ninguno bien, entonces siempre se va a quedar una parte considerable que no se terminó y que se va a descartar, tirar al cesto. ¿Por qué? Porque yo hice tres cosas a la vez, algo se tiene que postergar, y es muy posible que al pasar un poquito de tiempo nada más, se llega a cambiar el requirimiento o hay una modificación por más ligera que sea, entonces se descarta el trabajo hecho para comenzar todo de vuelta. Si hubiera terminado una cosa, y se está testeando mientras sigo con otro, no hay tanto derroche. Otro derroche se da con un crecimiento de los features (rasgos funcionales) y el alcance sin haber medido su impacto o haberse validado su valor.

También los silos aislados, compartamentos estancados, digamos por ejemplo dos user stories que se hacen al mismo tiempo por separado sin beneficio de conciencia colectiva sobre los trabajos a hacer, es muy posible que se invente la rueda más de una vez. Por ejemplo en el caso de un proyecto hecho en Drupal, que dos implementaciones de user stories hecho con sub-equipos que no están hablando, o conectados entre sí, utilicen dos módulos distintos para la misma cosa, y solo se descubre esto después, entonces alguien tiene que tirar trabajo, o sufre la calidad de conjunto. Eso es el tema del derroche inevitable si no se usa Lean UX y DevOps con las aplicaciones web.

Los handoffs (entradas y salidas de un silo a otro) cronicamente interrumpe el flujo de trabajo. Hay demoras por todo esto. Todos se sienten que están a la espera de lo que les van a entregar y cuándo. Todos se sienten que alguien les está esperando por algo. Y también por las mezquindades que muchas veces se dan en la planificación que no contemplaba un equipo que posee todos los conocimientos realmente necesario. Si no está presente, por ejemplo, el experto del dominio de negocios, si se hace mucho trabajo sin su presencia, sin que tenga la oportunidad de opinar sobre la marcha de la implementación, vamos a tener problemas y demoras por tener un equipo incompleto.

Después "task switching", o cambios frecuentes de tarea, es terrible, esto es tu enemigo. Porque generan un clima de sobrecarga en el trabajo y se dificulta el flujo de trabajo constante, tranquilo y en paralelo. Es un ambiente no creativo. Es un ambiente que no llega a ningún lado. Todos lo hemos padecido en algún momento. Algunos siempre lo están padeciendo.

Por este mismo motivo surgen muchos defectos, y también muchos defectos por la falta de compartir los conocimientos entre todos los miembros del equipo. Tiene que haber una onda, como vimos en el extracto que leímos, de compartir. Y no una jerarquía, con divas

y rockstars, un clima sobre-competitivo, sino un equipo, a quienes les encanta compartir entre sí lo que aprende sobre la marcha, por eso por ejemplo no se va a utilizar dos módulos distintos que hacen lo mismo.

Pregunta: ¿Cómo se van a manejar las dependencias. Un diseño puede ser dependencia de una implementación. Una solución de comercio electrónico puede ser una dependencia para una aplicación?

Respuesta: El secreto es la reunión colectiva inicial de cada sprint y una vez por semana por lo menos la validación del usuario. Porque se planificó entre todos. No era planificado todo por un guru que se metió en la oficina de la esquina, para luego decir: "Tu hacés esto, tu hacés aquello, tu hacés el otro..."; sino que todo el mundo planificó todo y entonces pueden trabajar en paralelo por más que existan dependencias, ya que se puede trabajar sobre una guía de estilos mientras se haga un desarrollo de back-end, o se puede esbozar un diseño, diseñar la versión final mientras el desarrollador siga adelante utilizando la guía de estilos para llegar a un prototipo que luego se pueda mejorar con afinaciones del diseño, por ejemplo. Y solamente se hace la cantidad de trabajo, justo a tiempo, que pueda tomar el equipo, así dificilmente uno se adelante o se atrase mucho de lo demás.

La otra respuesta es que hay alguien que va a definir, en paralelo con todo el trabajo de todo el equipo, una arquitectura candidata, que también se da a conocer entre todos. Allí justamente se evite el problema expresado en la pregunta porque el grupo tiene una conciencia social, en común, colectivamente, que permite que se entiende y se pueda preveer las cosas, todo el mundo entiende lo que está pasando.

Kanban

Kanban (not Scrum) (I)

Kanban (no Scrum) http://awebfactory.com/drupalcon2015lean/#/3/1

Las restricciones sobre el WIP (Work in Progress – Trabajo en progreso) es lo que hace el Kanban como metodología única.

Lo más importante para comprender sobre el Kanban es que no se basa en corridas de duración prefijada con el fin de entregar una mini-liberación en una fecha determinada, como es el Scrum. Más bien se trata de un proceso de flujo continuo de trabajo colectivo. “Themes” (temas), es decir, features (rasgos funcionales de alto nivel) que buscan resultados determinados de valor se implementan de a uno en un contexto de colaboración multidisciplinaria (cross-collaboration) y los trabajos y user stories (tarjetas) asociadas pasan a través de varios estados (representados por columnas) en el proceso. Al comienzo todos los trabajos (tarjetas) se colocan en la columna “New – Nuevo”. Estos se prioritizan y una cantidad determinada se colocan en la columna “To Do – por hacer”.

Es importante notar que la columna Por Hacer está limitada, no es lo mismo que Nuevo. Para que la atención del equipo no se dispersa.

Kanban (not Scrum) (II)

Kanban (no Scrum) http://awebfactory.com/drupalcon2015lean/#/3/2

Luego, cuando los miembros del equipo llevan a cabo un trabajo, su tarjeta se coloca en la columna “In Progres – en progreso”. Esta columna también está limitada en la cantidad de trabajos que pueda contener. La razón es que estas restricciones reemplaza la longitud de las corridas y las estimaciones de tiempo que son características del Scrum. En vez de colocar una fecha de estimación de entrega de una unidad de integración arbitraria, simplemente se deja trabajar al equipo en una cantidad de trabajos considerada óptima para su mejor rendimiento, que se calcula según la reputación de productividad que el equipo historicamente ha mostrado. Por eso, Kanban es diferente: Defiende la concentración y prohibe a la gente de trabajar en demasiados asuntos al mismo tiempo. Esto se concreta en el límite del WIP (Work in Progress – Trabajo en Progreso), y reemplaza la restricción de la duración de una corrida (sprint) en Scrum. De esta manera Kanban nos libera de las “estimaciones” y mini-liberaciones que convierte el proceso en mini-cascada. De esta manera, no hay sprints o períodos de trabajo con duración pre-determinada, sino un flujo de trabajos continuamente entregados para las pruebas de resultados de valor; y el límite de trabajo en progreso, visto como un número en paréntesis en los encabezamientos de algunas columnas, busca optimizar la productividad sin sobrecargar a nadie. El número exacto se calcula al tomar en cuenta el tamaño del equipo y su velocidad, y se ajusta para máximo rendimiento sin sobrecargar a medida que se va madurando el equipo. Al calcular en base del tamaño del equipo, se toma en cuenta que cada trabajo va a tener que contar con un mini-equipo que también contempla el paired (peer) programming (programación de pares, de a dos; ver http://en.wikipedia.org/wiki/Pair_programming ; y ver https://tctechcrunch2011.files.wordpress.com/2012/03/pair-programming.jpg?w=400).

Project Inception and Vision

Incepción del proyecto y la visión http://awebfactory.com/drupalcon2015lean/#/4

La Incepción del proyecto y la Visión signfica que tiene que haber algún punto de partida. ¿Por dónde empieza un proyecto? Siempre existen algunas entradas, que podría haber sido una conversación telefónica con el cliente, hasta todo un procedimiento que empezara con un RFP (pedido de propuesta). Es necesario juntar estas entradas para hacer la preparación necesaria para el Kickoff del equipo en la forma de un texto Visión.

Online Literary Workshop - Inception and Vision

Proyecto Taller Literario en línea – Incepción y Visión
http://awebfactory.com/drupalcon2015lean/#/4/1

Todo lo que hace falta para llevar a cabo el Kickoff (patada inicial) del Proyecto, material reunido a través de colaboración colectiva por la dirección inicial del equipo.

Se reunen todas las entradas: la documentación del cliente, las conversaciones, el sitio anterior si existe, y el contexto de negocio.

Se crea la versión inicial del texto Visión del proyecto para dar un punto de partida para el equipo. Debe incluir todos los “puntos de dolor” detectados: quienes los padecen, y cuál sería la solución según ellos. También se debe listar las alternativas de arquitectura para que puedan ser discutidas: los frameworks (entornos), distribuciones de Drupal reutilizables y otras soluciones probadas. Y también se deben listar las restricciones

(constraints) que se aplican.

Se debe elegir el equipo y reunirlos para el Kickoff.

Se debe considerar cómo se va a aprovisionar al equipo para que todo el mundo tenga su entorno de desarrollo y testeo.

Se debe decidir se se va a hacer o no un prototipo para discutir durante el Kickoff.

Inputs for Online Literary Workshop Vision

Entradas para la Vision del Proyecto Taller Literario en línea
http://awebfactory.com/drupalcon2015lean/#/4/2

Si bien el teto es la responsabilidad del Dueño del Producto (Product Owner) podrá ser elaborado en colaboración colectiva con el equipo incial pre-Kickoff.

Initial Online Workshop Vision

Texto Visión inicial para el Taller en línea
http://awebfactory.com/drupalcon2015lean/#/4/3

Hay una plantilla que ayuda a redactar el texto Visión aquí:
http://awebfactory.com/drupalcon2015lean/files/VisionTemplate.pdf

Y un ejemplo aquí:
http://awebfactory.com/drupalcon2015lean/files/OnlineLiteraryWorkshopVision.pdf

La Visión representará un primer paso de procesamiento de todas las entradas disponibles para proveer un punto de partida en el Kickoff para el abordaje de validación del problema, mercado y producto del proyecto.

Team Selection

La selección del equipo http://awebfactory.com/drupalcon2015lean/#/4/4

Como mínimo los siguientes disciplinas deben estar representadas en el equipo:

Product Owner – Dueño del Producto (más bien alguien del cliente)

DevOps – ver http://es.wikipedia.org/wiki/DevOps

UX – Experiencia de usuario http://es.wikipedia.org/wiki/Experiencia_de_usuario

Back-end Ver http://es.wikipedia.org/wiki/Front-end_y_back-end

Front-end *Idem.*

Project Coach

Graphic Design

El front-end y back-end se refiere a especialistas en cada uno de las capas involucradas en la “separation of concerns” o división del sistema en capas separadas para no mezclar la lógica y persistencia de datos con la capa de presentación en el interfaz del usuario. Ver http://en.wikipedia.org/wiki/Separation_of_concerns.

La cuestón es que un programador front-end se concierna más con lo que se ejecuta en el

explorador de internet (browser) mientras el programador back-end se concierna más con lo que se ejecuta en el servidor.

Obviamente el texto Visión determinará si se debería incluir más especialistas. Pero es importante destacar que para DurableDrupal Lean es mucho más importante el reclutamiento de un equipo que se va a funcionar como tal, y no el reclutamiento de llamados "rockstars" (sic) o aun peor los desarrolladores "10x" (sic) (ver el artículo que empezó el término en base del nombre de una agencia http://www.newyorker.com/magazine/2014/11/24/programmers-price y otro artículo mucho más sensato con lo cual estamos totalmente de acuerdo: https://codequalified.com/blog/2014/11/20/nonsense-of-10x-developers-and-github-as-a-cv/

> "There is no such thing as a 10X developer (without a 10X environment)"

"No existe ningún desarrollador 10X (sin un entorno 10X)", habría que decir, sin un equipo 10X.

Idealmente el Product Owner (Dueño del Proyecto) proviene de la organización del cliente y no es visitante sino que forma una parte integral y muy importante del equipo. Si no hay nadie disponible... entonces es mejor no aceptar hacer el proyecto: La probabilidad del fracaso es alta. Si se quiere seguir con el proyecto igual, entonces se debe utilizar un proxy, que representa los intereses del cliente vocalmente y con mucho coraje.

Resources for Team Provisioning

Recursos para el aprovisionamiento del equipo
http://awebfactory.com/drupalcon2015lean/#/4/5

- Ansible playbook para bajar e instalar la distribución de Drupal DurableDrupalDistro o bien localmente en un laptop o estación de trabajo, o en un servidor, en base a Vagrant y VirtualBox. Ver el video demo que lo demuestra: https://www.youtube.com/watch?v=ZVSGrID3g_s
- El playbook en GitHub: https://github.com/DurableDrupal/ansible-vagrant-durable-drupal-distro
- La distribución DurableDrupalDistro en GitHub: https://github.com/DurableDrupal/durable-drupal-distro
- Artículo: Create a Drupal project on Pantheon and pull it down to your laptop on Kalabox http://awebfactory.com/node/521
- Descripción del entorno servidor Pantheon: https://pantheon.io/how-it-works y descripción del flujo de trabajo en el entorno Pantheon: Workflow Overview https://pantheon.io/docs/articles/sites/code/using-the-pantheon-workflow/
- Cómo crear un entorno local de desarrollo para un proyecto Drupal en Platform.sh: https://docs.platform.sh/use-platform/getting-started-for-the-impatient/#how-to-set-up-your-local-drupal-development otro link en más detalle: https://docs.platform.sh/use-platform/set-up-local-development/

- Descripción general del Platform.sh: https://docs.platform.sh/use-platform/getting-started-for-the-impatient/ con un video: https://platform.sh/videos/2014/07/31/development-workflow/, y sobre el entorno: https://docs.platform.sh/overview/platform-environments/

Team Kickoff

Patada inicial del equipo http://awebfactory.com/drupalcon2015lean/#/5/1

- Cada sprint tiene una reunión Kickoff, no solo el inicial
- Participa todo el mundo
- Expertos del dominio se invitan según necesidad
- Puede durar unas horas o una semana, lo que haga falta

What goes on at the Kickoff

Qué pasa durante la reunión de patada inicial
http://awebfactory.com/drupalcon2015lean/#/5/2

No es la reunión de planificación, que lo sigue. El Kickoff es para tormenta de mentes y ejercisios de validación capaz de permitir al equipo definir un MVP (Minimum Viable Product – Mínimo Producto Viable capaz de validar los hipoteses) que luego en la reunión de planificación puede conducir a la planificación de los trabajos a efectuar en la forma de user stories.

Se tratan los problemas, los presupuestos y se generan los hipóteses que deben ser validados por los MVP (ver abajo). Una de las herramientas colectivas es el Estudio de Diseño que permite que todo el equipo trabaja juntos para visualizar cómo va a ser el producto.

Problems, Assumptions, Hypotheses

Problemas, Presupuestos, Hipoteses http://awebfactory.com/drupalcon2015lean/#/5/3

- Clarificar los planteamientos de los problemas principales
- Listar los supuestos (assumptions)
- Prioritizar los supuestos para crear los hipoteses necesarios para testearlos
- Elaborar y validar personas (sujetos)
- Tormenta de ideas para listar los features (rasgos funcionales de alto nivel)

  “Nuestro meta no es la elaboración de un entregable, sino cambiar algo en el mundo real – dar pie a un resultado. Empezamos con supuestos en vez de requirimientos. Creamos y validamos hipoteses. Medimos los resultados para ver si se ha alcanzado los resultados esperados.” (Gothelf)

El MVP será el mínimo producto que funciona capaz de testear los hipoteses. Iteraciones sucesivas desmenuzarán a los hipoteses en sub-hipoteses (trabajos, tarjetas), acercándose

al MVP final.

Problem Statement Template

Plantilla del Planteamiento del Problema
http://awebfactory.com/drupalcon2015lean/#/5/4

[Nuestro servicio/producto] fue diseñado para alcanzar [estos objetivos]. Hemos observado que el producto/servicio no está cumpliendo con [estos objetivos], los cual está causando [este efecto negativo] a nuestro negocio. ¿Cómo podríamos mejorar el [servicio/producto] para que nuestros clientes sean más exitosos, en base de [estos criterios medibles]?

Problem Statement Example

Ejemplo de Planteamiento del Problema http://awebfactory.com/drupalcon2015lean/#/5/4

Nuestro servicio ofrece un conducto entre los solicitantes de empleo y los empleadores que tratan de contratarlos. A través de nuestro servicio, los empleadores pueden llegar a los solicitantes de empleo en nuestro ecosistema con las oportunidades de empleo. Hemos observado que un factor crítico que afecta a la satisfacción del cliente es la frecuencia con que los solicitantes de empleo responden a los mensajes de los empleadores. Actualmente, los solicitantes de empleo está respondiendo a estas comunicaciones a una velocidad muy baja. ¿Cómo podemos mejorar la eficacia de nuestros productos de comunicación, así haciendo los empresarios más exitosos en sus búsquedas de solicitantes para llenar los puestos de trabajo y los solicitantes de empleo más satisfechos con nuestro servicio?

Hypothesis Template

Plantilla de Hipótesis http://awebfactory.com/drupalcon2015lean/#/5/6

Creemos que [esta afirmación es cierta]. Sabremos que estamos en lo [correcto / incorrecto] cuando vemos la siguiente retroalimentación del mercado: [retroalimentación cualitativa] y / o [comentarios cuantitativa] y / o [cambio indicador clave de rendimiento].

Hypothesis Example

Ejemplo de Hipótesis http://awebfactory.com/drupalcon2015lean/#/5/7

Creemos que la creación de un sistema de comunicación eficaz dentro de la experiencia del producto TheLadders' para los reclutadores y para los empleadores logrará una mayor tasa de éxito de contacto y mayor satisfacción con el producto. Nosotros sabremos que esto es cierto cuando vemos un aumento en el número de respuestas de los solicitantes de empleo a los contactos del reclutador y un aumento en el número de mensajes iniciados por los reclutadores en nuestro sistema.

Subhypothesis Template

Plantilla de sub-hipótesis http://awebfactory.com/drupalcon2015lean/#/5/8

Creemos que [hacer tal cosa / construir este rasgo de funcionalidad / crear esta

experiencia] para [estas personas] logrará [este resultado]. Nosotros sabremos que esto es cierto cuando vemos [esta retroalimentación del mercado, medida cuantitativa o percepción profunda].

La prioritization de la lista de sub-hipóteses constituye uno de los resultados más importantes de la reunión Kickoff del equipo.

Personas

Personas http://awebfactory.com/drupalcon2015lean/#/5/9

En un papel doblado en cuatro cuadrantes:

1. Atavar o croquis, nombre, edad, ciudad, ocupación, etc.
2. Información demográfica de conducta; estado civil, número de hijos, sus edades, condiciones de trabajo, el tiempo de ocio, estilo de vida, etc.
3. Puntos de dolor y las necesidades
4. Posibles soluciones

Primero crear Proto-personas, y luego investigar esto hablando con gente en el mundo real para validarlas y seguir desarrolándolas. Para realizar esta investigación tenemos que "salir del edificio" (Steve Blank: http://steveblank.com/2010/03/11/teaching-entrepreneurship-%E2%80%93-by-getting-out-of-the-building/) para poner a prueba el planteamiento del problema y los supuestos asociados. ¿Por qué tomar el consejo de un entrenador entreupeneur?

Debido a que cada proyecto es una start-up!

Feature brainstorming

Tormenta de cerebros sobre rasgos funcionales de alto nivel
http://awebfactory.com/drupalcon2015lean/#/5/10

Una vez que contamos con personas con problemas reales que han sido, idealmente, investigados y validados, podemos llevar a cabo una tormenta de cerebros, o lluvia de ideas (brainstorming) sobre los rasgos funcionales de alto nivel (features) capaces de motorizar el comportamiento de los clientes hacia los resultados deseados. Con la lluvia de ideas creamos una lista de rasgos que pueden ser visualizados y prioritizados.

Estos rasgos funcionales de alto nivel (features) serán los temas (themes), los conjuntos de sprints relacionados sobre la base de otorgar suficiente tiempo para la creatividad de los diseñadores y también para colaboración colectiva (cross-collaboration).

Collaborative Design and Prototype – Towards an Initial MVP

El diseño colaborativo y el prototipo – Hacia un MVP inicial
http://awebfactory.com/drupalcon2015lean/#/5/11

- Todo el mundo tiene la oportunidad de diseñar en forma colectiva
- Bocetos de baja fidelidad y artefactos aumentan la colaboración
- Métodos como el estudio de diseño grupal construyen una comprensión y una

base compartida de los rasgos funcionales por todo el equipo

- Las guías de estilo y librerías de modelos permiten que los desarrolladores trabajen en forma paralela con los diseñadores
- Se utilizan también técnicas especiales para los equipos remotos y distribuidos

Iteration Planning Meeting

Reunión de planificación para la iteración
http://awebfactory.com/drupalcon2015lean/#/5/12

La reunión clásica que toma todos los elementos elaborados hasta ahora para juntos escribir user stories (narrativas de los usuarios) y especificar los trabajos a implementar.

User Story Template

Plantilla de user story http://awebfactory.com/drupalcon2015lean/#/5/13

(Con influencia de A framework for modern User Stories by @jonatisokon https://medium.com/@jonatisokon/a-framework-for-user-stories-bc3dc323eca9)

Formato clásico

Como [persona, usuario] quiero [realizar una acción] para que [meta alcanzable].

Agregado para preuba de validación del usuario (testeo de aceptación)

Dado que estoy [realizando una acción] que estoy ... **Cuando** [punto de observación] **Entonces** [resultado observable]

En la tarjeta sobre el tablero Kanban, se coloca este user story. Se agregará la conversación entre miembros del equipo que acompaña la implementación, y al final se ejecutará la prueba de aceptación por parte del cliente.

User Story Example

Ejemplo de user story http://awebfactory.com/drupalcon2015lean/#/5/14

Como escritor quiero guardar mi texto para que pueda recibir una crítica.

Dado que he guardado mi texto **Cuando** hago click o toco el ícono guardar para guardar el texto **Entonces** se puede comprobar que esté presente en la base de datos y que es visible a los compañeros miembros del taller.

Es importante resaltar que la prueba de aceptación se debería escribir por el cliente y también antes de que se escriba una línea de código.

The goal of cross-collaboration

El meta de la colaboración colectiva http://awebfactory.com/drupalcon2015lean/#/5/15

Cuando se haya terminado de escribir los user stories que se van a colocar en el tablero Kanban , todo el equipo ha sido "iniciado" en cada uno de ellos. Un breve extracto (http://awebfactory.com/drupalcon2015lean/files/lavidaenlean.pdf) de Gothelf demuestra cómo es el desarrollo colectivo al trabajar en conjunto todo el equipo para completar el

trabajo planificado.

> "Este es el ritmo cotidiano del Lean UX: un equipo que trabaja en colaboración constante, de manera iterativa y en paralelo, con pocas entradas y salida, con mínimas entregas, y enfocados en software que funciona y en la retroalimentación del mercado."

Development Workflow with Everything in Code

El flujo de trabajo del desarrollo con todo en código

Everything in code example

Ejemplo de todo en código http://awebfactory.com/drupalcon2015lean/#/6/1

Hay un video con un demo en el minuto 14:35 del video:
https://www.youtube.com/watch?v=ZVSGrID3g_s

Drupal por defecto conserva toda su configuración lograda en forma interactiva en las páginas de administración como valores persistidos en la misma base de datos dónde se guardan los contenidos. Pero para poder trabajar en equipo de desarrollo, es decir, para poder “hacer commit” de modificaciones o de trabajos completados, en forma asincrónico sin destruir el trabajo de otros o los contenidos que van entrando los encargados del contenido, es necesario utilizar un repositorio de código con control de versiones, como exigen las mejoras prácticas del desarrollo del software.

Ofrecemos un ejemplo sencillo en el video, y llevamos a cabo los siguientes pasos:

- Cambiamos el tema corriente habilitado (default theme, con plantilla con la lógica de la capa de presentación y que genera el HTML, y los estilos CSS) y cambiamos el nombre del sitio
- Creamos un feature, o conjunto de características de configuración utilizando el módulo Features – https://www.drupal.org/project/features
- Especificamos los elementos para incluir en el feature.
- Generamos el código del feature y lo guardamos en el sistema de archivos
- Habilitamos la función, efectivamente es este momento mudando la configuración de los elementos seleccionados al código en vez de la base de datos.
- Pasamos a cambiar el tema corriente habilitado y el nombre del sitio interactivamente por las páginas de administración.
- Veamos en la página administrativa de Features que la feature se encuentra sobreescrito (overriden): ahora la base de datos está anulando el código debido a los cambios interactivos.
- "Revertimos" el feature, forzando la configuración a adaptarse al código, es decir, refrescamos la base de datos con lo que está fijado en el código.
- Comprobamos los cambios.

DevOps, Server Provisioning and Deployment

DevOps, el aprovisionamiento del servidor y el despliegue (habilitación de las instancias de testeo y de producción)

Resources for Server Provisioning and Deployment

Recursos para el aprovisionamiento del servidor y el despliegue (habilitación de las instancias de testeo y de producción) http://awebfactory.com/drupalcon2015lean/#/7/1

En esta breve presentación en realidad queríamos hacer hincapié en la utilización práctica y en conjunto de Lean, Lean UX y Lean DevOps. A continuación presentamos una lista corta de los muchos recursos disponibles para la cuestión tan importante del aprovisionamiento de servidores y la automatización del despliegue para los abordajes de la construcción frecuente y contínuo. Sin poder contar con ello, la parálisis del equipo, los retrasos y las interrupciones en el flujo de trabajo nos perseguirá mientras el monstruo de las entradas y salidas asoma su fea cabeza de cascada por defecto.

Lista muy corta de recursos http://awebfactory.com/drupalcon2015lean/#/7/2

- El libro de Jeff Geerling Ansible for DevOps: Server Configuration Management for humans
  - Repo en GitHub para el libro (actualizado frecuentemente)
  - La presentación de Jeff en el DrupalCon de Austin DevOps for Humans - Ansible for Drupal Deployment Victory!
  - Las diapositivas
- Tutorial sobre despliegue automatizado por Ansible en la documentación de DigitalOcean How To Create an Ansible Playbook To Automate Drupal Installation on Ubuntu 14.04