



HORA DE COMPRAR





HORA DE COMPRAR



HORA DE COMPRAR



[MVP]



Logre su **Producto Mínimo Viable** para probar su concepto.





HORA DE COMPRAR

[MVP]

El **perímetro** de su MVP debe **reducirse** mientras le permite comercializar su producto.

Apuesta a **Early Adopters** y recibe un máximo de **feedback**.

Su MVP está implementado y puede utilizarse en **producción**.



HORA DE COMPRAR



[FALLAR RAPIDO]



Fail fast es **learn** fast.





HORA DE COMPRAR

[FALLAR RAPIDO]

Experimente rápidamente la solución (unas semanas), reúna **comentarios** de sus usuarios y aprenda de sus errores.

No tengas miedo de cambiar todo

**

No lo olvides, **ifallará!**



HORA DE COMPRAR



[BESO]



** Mantenlo simple y estúpido.

- ¿Por qué complicarse cuando puede ser simple? *





HORA DE COMPRAR

[BESO]

Evite la sobreingeniería , si un modelo de "papel" o un Formulario de Google es suficiente para probar su concepto, no vaya más allá.

¡Mantente simple! Tanto técnica como funcionalmente.



HORA DE COMPRAR



[PRODUCTIVIDAD]

Especifique menos,
expanda más.





HORA DE COMPRAR

[PRODUCTIVIDAD]

Limite sus especificaciones a lo esencial, **concéntrese en el "qué"** en lugar del "cómo".

El producto debe ser **lo más** auto-documentado posible.

La documentación debe estar versionada de la misma manera que el código.



HORA DE COMPRAR



[SAAS]

Estudiar sistemáticamente
las soluciones **SaaS**.





HORA DE COMPRAR

[SAAS]

Las soluciones de SaaS son sostenibles y rentables.

En algunos casos, **SaaS** puede **acelerar** la implementación de **MVP**.

Piense en la visión económica **con respecto a las alternativas en términos de** costo total (TCO : **T**otal **C**ost de **O**wnership) y no solo en términos de costo de la licencia.



HORA DE COMPRAR



[CORAZÓN DE
NEGOCIOS]

El **negocio principal** no
debe ser un obstáculo
para la construcción de
nuevos servicios y
aplicaciones.





HORA DE COMPRAR

[CORAZÓN DE NEGOCIOS]

El ritmo de evolución y entrega del negocio central debe ser **compatible con la agilidad** de los servicios que lo consumen.

El negocio principal debe **exponer servicios**.

El negocio principal debe adoptar un principio **orientado al evento**, informa acciones de gestión en forma de eventos.



HORA DE COMPRAR



[DESPLIEGUE CONTINUO]

**Despliegue en
producción** no es un
evento.





HORA DE COMPRAR

[DESPLIEGUE CONTINUO
]

Aproveche **la implementación continua** para adaptar ****
producción **a los requisitos**
comerciales ** y no al revés.

Las implementaciones en todos
los entornos, hasta **producción** ,
deben ser **automáticas** y
frecuentes.



HORA DE COMPRAR



[BETA PERPETUA]

El enfoque **perpetuo beta** le permite involucrar a sus usuarios en el proceso de desarrollo.





HORA DE COMPRAR

[BETA PERPETUA]

Siéntase libre de utilizar el principio beta perpetuo en el que **los usuarios participan en el desarrollo.**

El término beta perpetuo se refiere a una aplicación desarrollada en just-in-time, **en constante evolución**, y no en un producto incompleto.









EXPERIENCIA DE
USUARIO

[PERCEPCIÓN]



La **experiencia percibida**
por el usuario es
fundamental.

ergonomía no es
negociable.





EXPERIENCIA DE
USUARIO

[PERCEPCIÓN]

No descuides el trabajo de los **diseñadores de UX** , es fundamental en el desarrollo de una aplicación.

Integre el **feedback** de sus usuarios, este es esencial.





EXPERIENCIA DE
USUARIO

[ACTUACIÓN]

Utilice **potentes interfaces**
para usos internos y
externos.





EXPERIENCIA DE
USUARIO

[ACTUACIÓN]

Las interfaces están orientadas a **eficiencia**.

El **rendimiento** de una interfaz ahorra tiempo , **aumenta la** satisfacción de los usuarios **y por lo tanto** guarda su frustración **.





EXPERIENCIA DE
USUARIO

[MÓVIL PRIMERO]

Adopte una estrategia
Mobile First.





EXPERIENCIA DE
USUARIO

[MÓVIL PRIMERO]

Los dispositivos móviles son la
parte más importante del
mercado.

Pensar en dispositivos móviles es
pensar en lo **esencial**.

Diseño receptivo es la norma, es
una fuente de ahorro (**MVP**).





EXPERIENCIA DE
USUARIO

[OMNI-CANAL]

Adáptate a los usos, el
omni-canal es la norma.





EXPERIENCIA DE
USUARIO

[OMNI-CANAL]

El enfoque omnicanal proporciona al usuario una **experiencia unificada** (ejemplo: Netflix).

Los diferentes **canales** están **sincronizados** y **coherentes** (a diferencia de los procesos por lotes).

Todos los actores (clientes, asesores) tienen acceso a la misma información.





EXPERIENCIA DE
USUARIO

[AUTODATOS]

**** usuarios *son*
propietarios ** de sus datos
y su curso.





EXPERIENCIA DE
USUARIO

[AUTODATOS]

Deje **personas** , en cualquier momento, **control** en sus datos **personales**.

Establezca una **confianza** al permitir a los usuarios la trazabilidad y el control en tiempo real.

subsistemas deben cumplir los mismos requisitos.





EXPERIENCIA DE
USUARIO

[CRM / SFA]



La relación con el cliente
debe unificarse y
contextualizarse con un
CRM / SFA flexible,
unificador y orientado a
eventos **.





EXPERIENCIA DE
USUARIO

[CRM / SFA]

Opte por **CRM** que gestione tanto la relación con el cliente como el liderazgo de la fuerza de ventas (**SFA** : **S**ales **F**orce **A**utomation).

CRM debe estar **abierto** para nuevas oportunidades.

CRM produce **eventos** correspondientes a las acciones de gestión para ajustarse a la lógica **de la plataforma orientada a eventos**.





EXPERIENCIA DE
USUARIO

[BIG DATA]

La plataforma Big Data le
permite centralizar **y**
procesar datos de usuario
para servir mejor su viaje

**
,





EXPERIENCIA DE
USUARIO

[BIG DATA]

Centralice los datos **** Maif Group
, Partner y Vendor **en una lógica**
pathway **.

La "preparación de datos" y el
procesamiento pueden **consolidar**
los datos.

Los equipos de Big Data
colaboran con los equipos de
características para garantizar el
gobierno de **datos**.





EXPERIENCIA DE
USUARIO

[PUESTO DE TRABAJO]

La estación de trabajo está
adaptada y se puede
adaptar a **usos** y **canales**
modernos.





EXPERIENCIA DE
USUARIO

[PUESTO DE TRABAJO]

Adopte la **federación de identidades** para una experiencia unificada.

Un **portal** permite ofrecer **una descripción general** , no reemplaza las aplicaciones.

La estación de trabajo debe ser **móvil** , **multicanal** y **estándar** para permitir la apertura dentro de **empresa extendida**.





EXPERIENCIA DE
USUARIO

[COLABORADORES]

No olvide que sus
asociados están utilizando
aplicaciones modernas en
su hogar en UX.





EXPERIENCIA DE
USUARIO

[COLABORADORES]

Trate a **todos sus usuarios como "clientes"** : usuarios de Internet, gerentes, operadores, desarrolladores, etc.

No subestime el **esfuerzo de UX** para implementar aplicaciones de administración de uso interno.





EXPERIENCIA DE
USUARIO

[TODA LA MEDIDA]



Todo lo que se puede
medir debe ser.

**Sin medida, todo es solo
opinión.**





EXPERIENCIA DE
USUARIO

[TODA LA MEDIDA]

Piense en las métricas durante el **desarrollo** de la aplicación. **logs** debe tener una dimensión **comercial así como técnica**.

No descuides las **métricas de rendimiento** , son fundamentales.

El equipo de características proporciona **operación** : es responsable de hacer **la aplicación utilizable**.





EXPERIENCIA DE
USUARIO

[Prueba A / B]

A / B Testing le ahorra
tiempo al permitir que se
decida **feedback**.





EXPERIENCIA DE USUARIO

[Prueba A / B]

En lugar de decidir arbitrariamente entre dos soluciones, no dude en configurar **pruebas A / B**.

Este patrón consiste en presentar **dos versiones diferentes** de la misma aplicación y elegir una de ellas en función de **medidas objetivas** de la actividad del usuario.





EXPERIENCIA DE
USUARIO

[DETERIORO]



Considere la degradación
en lugar de la interrupción
del servicio en caso de
falla.





EXPERIENCIA DE
USUARIO

[DETERIORO]

En **falla** de uno de los subsistemas, **una versión degradada** del servicio **debe considerarse** en primer lugar en lugar de una interrupción.

Con **Disyuntores** , **aisle un desglose** para **evitar** su **impacto** y **propagación** en todo el **sistema**.







HUMANO



[EQUIPO PRINCIPAL]



El equipo está organizado
en torno a **productos** o
servicios prestados.





HUMANO

[EQUIPO PRINCIPAL]

Los equipos son **Equipos destacados**, organizados en torno a un conjunto funcional coherente, y compuestos por todas las **habilidades** necesarias para este conjunto.

Por ejemplo: Business Expert + Web Developer + Java Developer + Architect + DBA + Operational.

La **responsabilidad** es **colectiva**, el Equipo de funciones tiene el poder necesario para esta responsabilidad.



HUMANO



[EQUIPO 2-PIZZA]

Limite el **tamaño de los equipos especiales** (de 5 a 12 personas).





HUMANO

[EQUIPO 2-PIZZA]

Limite el tamaño de un equipo de características: **entre 5 y 12 personas.**

Por debajo de 5, ella es demasiado sensible a los eventos externos y carece de creatividad. Por encima de 12, pierde productividad.

El término "**Equipo de 2 pizzas**" indica que el tamaño del equipo de características no debe exceder el número de personas que pueden ser alimentadas con dos pizzas.



HUMANO



[SOFTWARE ARTESANAL
]



Apuesta a personas
versátiles que **saben cómo
hacer** y a quienes **les
gusta hacer**.





HUMANO

[SOFTWARE ARTESANAL]

Lo más importante es la **cultura de desarrollo, escalabilidad y adaptabilidad.**

Reclutando **artesanos de software y desarrolladores de software completo**, aportan un valor agregado real a través de su conocimiento y su visión general.

Sin embargo, los desarrolladores móviles, por ejemplo, suelen ser **desarrolladores especializados.**



HUMANO



[RECLUTAMIENTO]

Sé **atractivo** para reclutar
el **mejor**.





HUMANO

[RECLUTAMIENTO]

Proponer modos de trabajo adaptados a los empleados:
movilidad, trabajo a domicilio, CYOD CYOD (Choose Your Own Device.

Deje tiempo para la experimentación y hágalo realidad
en tiempo de trabajo.



HUMANO



[EVE]

La organización debe ser
un **motor de sueño**

El día anterior es parte del
trabajo.





HUMANO

[EVE]

La organización debe ser un motor **de cuidado diurno** mediante el establecimiento de sistemas como **educación continua** o **universidades empresariales**.

Siéntase libre de combinarlos con otras formas **más informales**
como: Codificación de Dojos ,
Almuerzos de Brown Bag ,
Conferencias ** Externas.



HUMANO



[CO-CONSTRUCCIÓN]

Rompe las barreras entre
los intercambios, apuesta
por los objetivos de
convergencia.





HUMANO

[CO-CONSTRUCCIÓN]

Para romper las barreras entre los intercambios, no es suficiente agrupar a las personas en torno a un producto común en un lugar común.

Los **Enfoques ágiles** eliminan estas barreras para asegurar **la convergencia de objetivos**.

Estas prácticas son una parte integral de las claves del éxito, la organización es el garante.



HUMANO



[DEVOPS]

Las prácticas **DevOps**
permiten que las paredes
caigan entre Build y Run.





HUMANO

[DEVOPS]

Adoptar **DevOps** para converger **Dev** y **Ops** hacia un objetivo común: **servir a la organización**.

¡Los intercambios siguen siendo diferentes ! DevOps no significa que la misma persona realiza las tareas de Dev y Ops. **Los desarrolladores y Operativo** deben **colaborar** para **beneficiarse** de ^{****} habilidades ^{*}y *para mejorar*^{*} empatía ^{**}.



HUMANO



[DOLOR]



Las tareas difíciles se
realizan **por el equipo de**
características.

La automatización sigue.





HUMANO

[DOLOR]

En una organización tradicional, **la falta de comprensión** entre los equipos suele estar relacionada con la distancia y **la falta de comunicación**.

Los **miembros de un equipo de características** son **co-responsables** y **solidarios** para todas las tareas.

Dolor es un factor clave en **Mejora continua**.



HUMANO



[CDS]

Los centros de servicio son
difíciles de conciliar con el
compromiso colectivo.





HUMANO

[CDS]

Los equipos de características se basan en principios que se basan en gran medida en **colaboración** y **participación colectiva**.

Los centros de servicios están avanzando hacia la racionalización y consolidación de TI por parte de las empresas, lo que es contrario a esta noción de compromiso colectivo.



HUMANO



[VALIDACIÓN]

La organización tiene **un rol de validación**, sin ser dogmático.





HUMANO

[VALIDACIÓN]

Asegúrese de que la organización conserve **su función de validación en las herramientas y usos. En particular sobre las** herramientas que afectan el patrimonio ** (ejemplo: gestión del código fuente).

Proporcione Equipos destacados con **medios** para respaldar sus elecciones.

No seas **dogmático** y asegúrate de **estimular la experimentación.**



HUMANO



[TRANSVERSALIDAD]



Se espera que los equipos
especiales **se
comuniquen** y compartan
sus **experiencias** y
habilidades.





HUMANO

[TRANSVERSALIDAD]

No cree barreras entre **Equipos destacados**.

Configure una **organización** y la **agilidad** necesaria para que los equipos de características se comuniquen entre sí y compartan sus habilidades y experiencias.

La organización de la transversalidad en **Spotify** (Tribus, Capítulos y Gremios) es un ** ejemplo elocuente.





INTEROPERABILIDAD





INTEROPERABILIDAD



INTEROPERABILIDAD



[API PARA TODOS]



API para todos los usos :
interno, clientes y socios,
público.





INTEROPERABILIDAD

[API PARA TODOS]

Abra su organización para nuevos usos y nuevos clientes con **Public APIs**.

En **sociedades** comerciales , clientes **como** proveedores **, las API son el formato de intercambio estándar.

Las API también están destinadas a **usos internos** de la organización.



INTEROPERABILIDAD



[AUTOSERVICIO]



El uso de una API debe ser
simple y rápido.





INTEROPERABILIDAD

[AUTOSERVICIO]

El uso de API debe ser lo más simple posible. Piensa en **la experiencia del desarrollador**.

La mejor solución para validar la adecuación con la necesidad es **probar la API rápidamente** : ¡unos pocos minutos deben bastar!

La plataforma debe ofrecer una **interfaz gráfica** para probar la API simplemente.



INTEROPERABILIDAD



[GESTIÓN DE API]

Los usos de las API deben
ser **controlados** y
controlados.





INTEROPERABILIDAD

[GESTIÓN DE API]

Implemente una solución de administración de API para administrar **cuotas, aceleración, autenticación y registro**.

Recoja métricas para administrar **monitoreo, filtrado y informes**.





Establezca **requisitos** para
sistemas y servicios
externos integrados en la
plataforma.



INTEROPERABILIDAD

[REQUISITOS]

Requerir que **sistemas externos** cumplan los mismos **requisitos** que **sistemas internos**.

Los sistemas externos deben publicar **eventos** y permitir la supervisión **técnica**.

En el caso donde los datos del sistema externo deben integrarse, la sincronización **total** debe ser **posible**.



INTEROPERABILIDAD



[MULTI-TENANT]

La arquitectura debe ser
pensada **multi-tenant**.





INTEROPERABILIDAD

[MULTI-TENANT]

Incluso si la marca blanca no se considera en la base, configure una arquitectura multi-tenant. Su aplicación **inicial** es la primera **celebración**.

Piense en la **instanciación multifuncional** del sistema desde el principio.



INTEROPERABILIDAD



[AJUSTE]

Los sistemas deben ser
**nativamente
configurables.**





INTEROPERABILIDAD

[AJUSTE]

Los idiomas, las monedas, las reglas comerciales, los perfiles de seguridad deben ser fáciles de configurar.

Tenga cuidado con **hipergenericidad**, a menudo es inútil y **fuentes de costo**.

la configuración debe ser **escalable** y rápida según sea necesario.



INTEROPERABILIDAD



[FLIPPING DE
CARACTERÍSTICAS]



Cree sistemas flexibles y
genéricos usando **función
voltear**.





INTEROPERABILIDAD

[FLIPPING DE CARACTERÍSTICAS]

la función voltear se trata de diseñar una aplicación como un conjunto de **funciones** que se pueden **habilitar** o **desactivar** caliente, **producción**.

En una aplicación **multi-tenant** , la función flipping le permite **personalizar** a los seguidores.

Le función voltear **simplifie l'A / B testing**.









Las **opciones técnicas**
están **realizadas y**
asumidas por **Equipo de**
características.





REGLAS DEL JUEGO

[OPCIONES TÉCNICAS]

El Equipo de características debe actuar **responsablemente** para identificar las opciones que lo afectan exclusivamente y las elecciones que impactan a la organización.

Las **opciones** que **exceden el alcance** del Equipo de características (por ejemplo, licencia, lenguaje de programación infrecuente) deben **validarse** por la organización o por el proceso de convergencia entre pares .





La **herramienta correcta**
para **buen uso** es una
fuente de ahorro.






REGLAS DEL JUEGO

[BUEN USO]

Una **mala herramienta** impuesta a todos es un **riesgo**. El **uso indebido** de una buena herramienta puede tener consecuencias **muy** dañinas **. Por ejemplo, los métodos ágiles mal utilizados son peligrosos.

herramientas debe ser **cuestionado**.

Excel a menudo es una opción racional **pero no es** una herramienta para hacer todo **. (CRM, ERP, DATA MAINTENANCE ...)

REGLAS DEL JUEGO



[CONSTRUIR VS.
COMPRAR]

Privilege **Build** para el
negocio principal.

Considere **Comprar** para
el resto, caso por caso.





REGLAS DEL JUEGO

[CONSTRUIR VS. COMPRAR]

Cuanto más una herramienta tenga una función de **característica diferenciadora** para la organización, más se pretende **que se cree**. El negocio principal debe permitir **la especificidad** y **adaptarse rápidamente ya menudo**. Algunos **paquetes de software a veces se adaptan** a esta necesidad.

Para **el resto** : SaaS, Open Source, Build o Owner se estudiarán **caso por caso**.



REGLAS DEL JUEGO



[FUENTE ABIERTA]

**Aproveche al máximo el
código abierto.**

Las opciones alternativas
deben ser compatibles.





REGLAS DEL JUEGO

[FUENTE ABIERTA]

Las **soluciones propietarias** son un **riesgo** para la organización que debe poder reanudar el mantenimiento si es necesario.

Hay pocas herramientas propietarias que no tienen **alternativas de código abierto**.

La organización **se beneficia** de **la Comunidad de Código Abierto** y puede **devolver sus contribuciones**.



REGLAS DEL JUEGO



[MICRO-SERVICIOS]



Desarrollar servicios
independientes y
débilmente acoplados.





REGLAS DEL JUEGO

[MICRO-SERVICIOS]

acoplamiento débil debe ser la norma.

Cada micro-servicio tiene **una interfaz claramente definida**.

Esta **interfaz** determina el **enlace** entre los **micro servicios**.

Domain Driven Design permite, especialmente con **Contextos acotados**, anticipar este problema.



REGLAS DEL JUEGO



[DATOS]

Cada servicio tiene su
propio ** sistema de
almacenamiento de datos.





REGLAS DEL JUEGO

[DATOS]

Un **Data Store** está destinado a ser **acoplado** solo con **un solo micro-servicio**.

El **acceso a los datos** de un micro-servicio a otro se hace **exclusivamente a través de su interfaz**.

Este diseño implica **consistencia en el tiempo** en toda la plataforma. Debe ser **aprehendido en todos los niveles**, incluido UX.





Cada micro-servicio debe tener un perímetro funcional razonable, que **"cabe en la cabeza"**.



REGLAS DEL JUEGO

[ÁMBITO]

Un micro-servicio ofrece **un número razonable de funciones**.

No dude en cortar un micro-servicio cuando este empiece a crecer.

Un servicio de tamaño razonable permite **considerar** serenamente **la reescritura** , si surge la necesidad.



REGLAS DEL JUEGO



[CAPACIDAD DE
RESPUESTA]

El **Manifiesto reactivo** abre
el camino hacia el diseño
de arquitecturas reactivas.





REGLAS DEL JUEGO

[CAPACIDAD DE RESPUESTA]

La programación sensible se centra en el flujo de datos y la propagación del cambio. Se basa en el patrón "**Observer**" al contrario del enfoque "**Iterator**", más tradicional.

El Manifiesto reactivo establece los ejes fundamentales:

disponibilidad y velocidad,
resiliencia a las interrupciones,
flexibilidad, **elasticidad** y
orientación del mensaje.



REGLAS DEL JUEGO



[ASYNC-FIRST]

los procesos asincrónicos
favorecen
desacoplamiento y
escalabilidad **a favor de**
rendimiento **.





REGLAS DEL JUEGO

[ASYNC-FIRST]

El intercambio entre aplicaciones debe ser **asincrónico** primero.

Los intercambios asíncronos naturalmente **permiten un acoplamiento** débil , aislamiento **y** control de flujo (contrapresión **).

la comunicación síncrona solo debe considerarse **cuando la acción lo requiera**.





El sistema de información
debe estar orientado
eventos.





REGLAS DEL JUEGO

[EVENTOS]

Los procesos **** *de* procesos impulsados por eventos **están** naturalmente **implementados de forma asincrónica**.

La **orientación del evento** permite favorecer la implementación de enfoques tales como **C**ommand **Q**uery **R**esponsibility **S**egregation (**CQRS**) y **Event Sourcing**.





Privilege a **agente de mensajería** simple, robusto y poderoso a un "conducto inteligente".





REGLAS DEL JUEGO

[AGENTE DE MENSAJES]

ESB mostró **límites** : el **mantenimiento escalable** es **crítico** , tanto desde el punto de vista **técnico** como **organizacional**.

Los mensajes de Broker como Kafka ofrecen una solución simple , **duradera** y **flexible**.

Puntos finales inteligentes y **Tuberías simples** es una arquitectura que funciona a escala: es **Internet**.





La **sincronización completa** del sistema debe pensarse tan pronto **esté diseñado.**





REGLAS DEL JUEGO

[TIEMPO]

Si la **sincronización** entre dos sistemas está garantizada por un **flujo de eventos** , la **resincronización** total de estos sistemas debe **planificarse en el momento del diseño**.

Una **** **auditoría de sincronización** automática (ejemplo: por ejemplos) permite **medir** y **detectar** posibles **errores de sincronización**.





La **configuración** de los servicios está **centralizada**, su **descubrimiento** está garantizada por un **directorio**.



REGLAS DEL JUEGO

[CENTRALIZACION]

La **configuración** de los **micro-servicios** está **centralizada** para todos los **entornos**.

Un **directorio** central garantiza **descubrimiento dinámico** de **micro-servicios**.

La **escalabilidad global** depende **de este** directorio **.





Los equipos especiales
proporcionan un entorno **
de espacio aislado.





REGLAS DEL JUEGO

[CAJÓN DE ARENA]

Los equipos especiales mantienen un entorno de **recinto de seguridad** (versión actual y próximo lanzamiento) para permitir que otros **equipos amplíen**.

En **algunos casos** no nominales, **funciones** pueden **estar inhabilitados** en el entorno **de desarrollo**.



REGLAS DEL JUEGO



[DISEÑO PARA FALLAS]



¡Tu sistema se bloqueará!

Diseñarlo para que sea
tolerante.





REGLAS DEL JUEGO

[DISEÑO PARA FALLAS]

Tu **sistema fallará**, es inevitable.
Debe estar diseñado para esto
(**Design For Failure**).

Predecir **redundancia** en todos los niveles: **hardware** (red, disco, etc.), **aplicaciones** (instancias múltiples de aplicaciones), **zonas** geográficas, **proveedores** (ejemplo: AWS + OVH).



REGLAS DEL JUEGO



[KITS DE HERRAMIENTAS]

Proporcione **kits de herramientas** , no imponga marcos estrictos.





REGLAS DEL JUEGO

[KITS DE HERRAMIENTAS
]

Atención a los componentes técnicos casas y transversal ! Son restrictivos, caros y difíciles de mantener.

aceleradores , juegos de herramientas , acumulaciones técnicas pueden ser **agrupados , gratuitos** Equipos destacados, evitando un enfoque dogmático.





Público, privado o híbrido,
la nube (IaaS o PaaS) es el
estándar para la
producción.



REGLAS DEL JUEGO

[NUBE]

Los servicios de **PaaS** son **preferidos, simples** y se escalan rápidamente.

Los servicios **IaaS** le permiten abordar casos que requieren una mayor **flexibilidad** pero requieren más trabajo operativo.

Una nube privada no es un entorno de virtualización tradicional, sino que se basa en **hardware básico**.





Los equipos especiales no administran la infraestructura, **la organiza y la mantiene.**





REGLAS DEL JUEGO

[INFRAESTRUCTURA]

Los problemas de infraestructura no están dentro de **Equipos destacados**. La infraestructura debe **proporcionarse** y **mantenerse** mediante un servicio ** con funciones cruzadas.





Los contenedores brindan
la flexibilidad necesaria
para herramientas
heterogéneas.



REGLAS DEL JUEGO

[ENVASES]

Los contenedores proporcionan la **flexibilidad** que necesitan los equipos de características para habilitar **herramientas heterogéneas** en **un contexto homogéneo**.





El uso de **contenedores**
hace posible superar los
problemas de **entornos**
técnicos.





REGLAS DEL JUEGO

[ENTORNOS]

Los contenedores (ejemplo: **Docker**) hacen posible **ser liberado** de las diferencias de entorno.

El proceso de **implementación** debe ser **agnóstico** para el entorno.

Algunos componentes como las bases de datos no se deben implementar en contenedores. Su implementación aún está automatizada.





Las medidas deben estar
centralizadas y
accesibles para todos.





REGLAS DEL JUEGO

[METRICO]

Las **métricas** son **accesibles** para todas las personas con diferentes niveles de granularidad: vista detallada de la función del equipo relevante, agregaciones para otros miembros de la organización.

El acceso a **las métricas no implica el acceso a los datos de la unidad**, debe controlarse para mantener la confidencialidad.

Todos los entornos están afectados.



REGLAS DEL JUEGO



[CALIDAD]

la calidad del software es
un factor clave.





REGLAS DEL JUEGO

[CALIDAD]

las revisiones de los códigos son **sistemáticas**. Son dirigidos por miembros del Equipo de funciones u otros miembros de la organización, como parte de **Continuous Improvement**.

Eso **no está siendo auditado sino su código** : "¡Usted no es su código!".

El **qualimetry** puede ser parcialmente automatizado, pero nada supera al **"nuevo ojo"**.





Pruebas automatizadas
es un requisito previo no
negociable para la
implementación continua.





REGLAS DEL JUEGO

[PRUEBAS AUTOMATIZADAS]

Pruebas automatizadas asegura **calidad** del producto **a lo largo del tiempo**.

Es **un prerequisite** para la implementación continua, permite ^{***} cambios ^{*}y^{*} implementaciones frecuentes ^{**}.

¡El **lanzamiento de la producción** se convierte en un evento **anecdótico** !





**Pruebas en todos los
niveles** : unidad,
integración, funcional,
resiliencia, rendimiento.





REGLAS DEL JUEGO

[NIVELES DE PRUEBA]

Las pruebas de **integración** y **funcional** son las más importantes **garantizan** la **operación** efectiva **.

Las pruebas de **unidad** son adecuadas para **desarrollo**.

las pruebas de rendimiento **miden el rendimiento** a lo largo del tiempo **.

Las pruebas de resiliencia ayudan a anticipar **fallas**.





Cover es el principal
indicador objetivo de la
calidad de la prueba.





REGLAS DEL JUEGO

[CUBIERTA]

La **cobertura del código** según las pruebas es una **buena** métrica de la calidad del código.

Esta es una **condición necesaria** pero **no suficiente**, la cobertura de una estrategia de prueba **incorrecta** puede ser alta sin garantizar la buena calidad del código.





La **seguridad** es un **proceso** , no debe tratarse en respuesta a los problemas.



REGLAS DEL JUEGO

[SEGURIDAD]

expertos en seguridad pueden **integrarse** directamente en los equipos de características **si es necesario**.

expertos en seguridad están disponibles en la organización para **auditoría** , **conocimiento** y **reenvío**.

