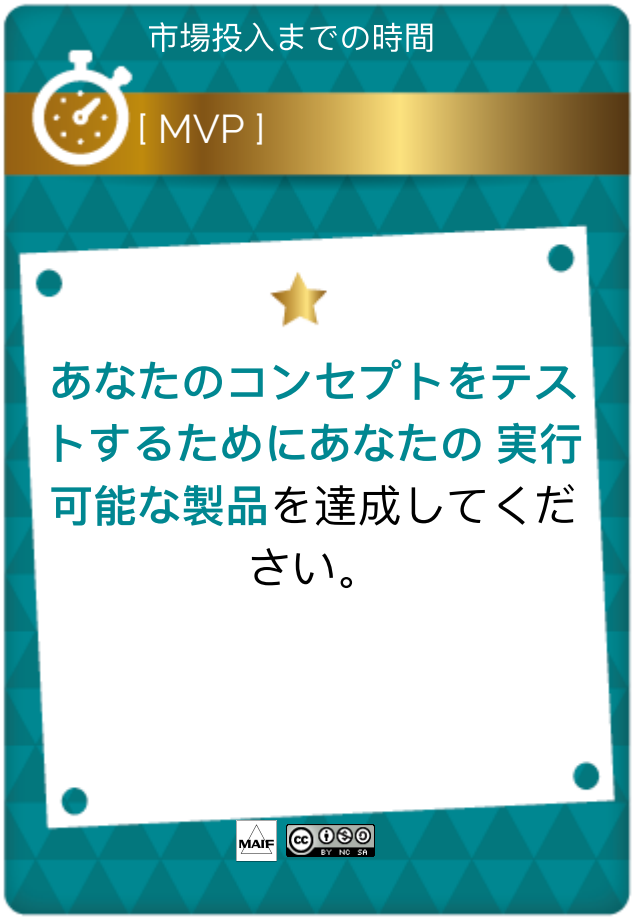

市場投入までの時間
MAIF


市場投入までの時間
MAIF


市場投入までの時間
[MVP]
あなたのコンセプトをテストするためにあなたの 実行可能な製品を達成してください。
MAIF

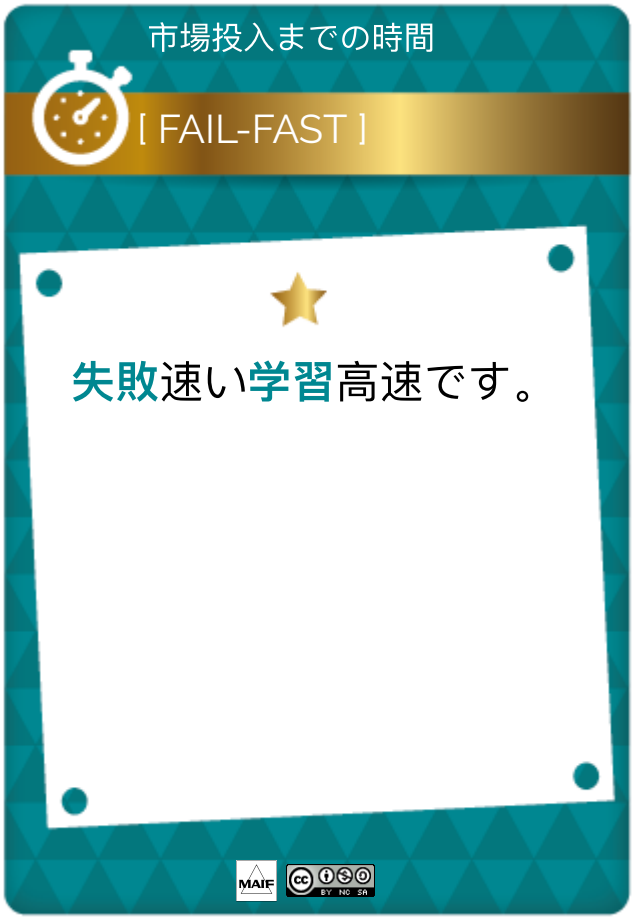
市場投入までの時間

[MVP]

MVPの**周辺**は**あなたの製品を販売することを可能にしながら**減らされなければなりません**。

早期導入者に賭けて**最大限のフィードバック**を得る

あなたのMVPは、**プロダクション**に導入され、使用可能です。


市場投入までの時間
[FAIL-FAST]
失敗速い学習高速です。
MAIF

市場投入までの時間

[FAIL-FAST]

すぐに解決策（数週間）を経験し、ユーザーからフィードバック**を収集し、あなたの間違いから学ぶ。

すべてを変えることを恐れないでください**。

あなたが失敗することを忘れないでください!

市場投入までの時間

[キッス]

** K ** eep ** I ** t ** S **
imple** S ** tupid。

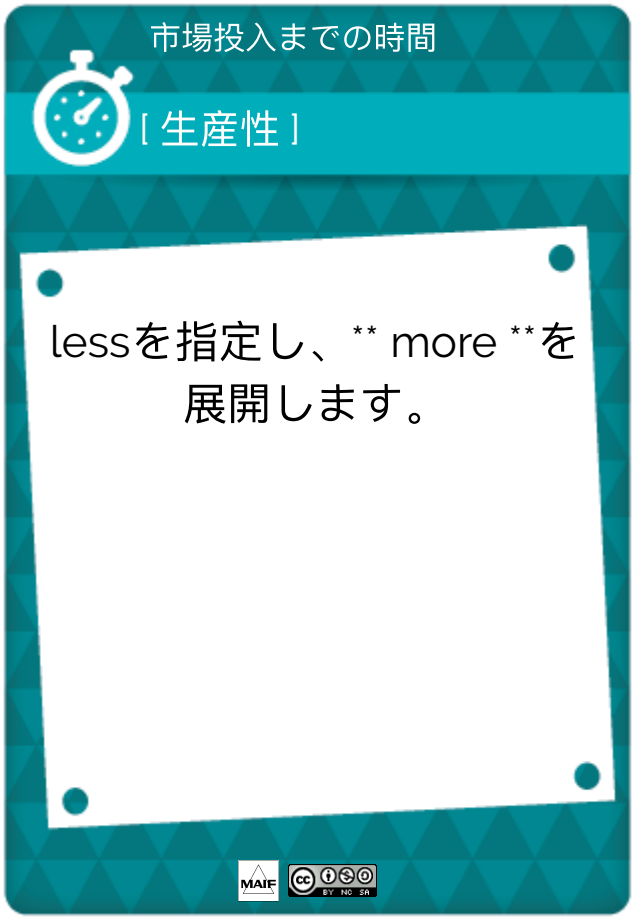
シンプルになると複雑になるのはなぜですか?

市場投入までの時間

[キッス]

オーバーエンジニアリングを避けてください。"ペーパー"モデルやGoogleフォームがコンセプトをテストするのに十分であれば、さらに進まないでください。

シンプルに！技術的にも機能的にも。


市場投入までの時間
[生産性]
lessを指定し、** more **を
展開します。
MAIF

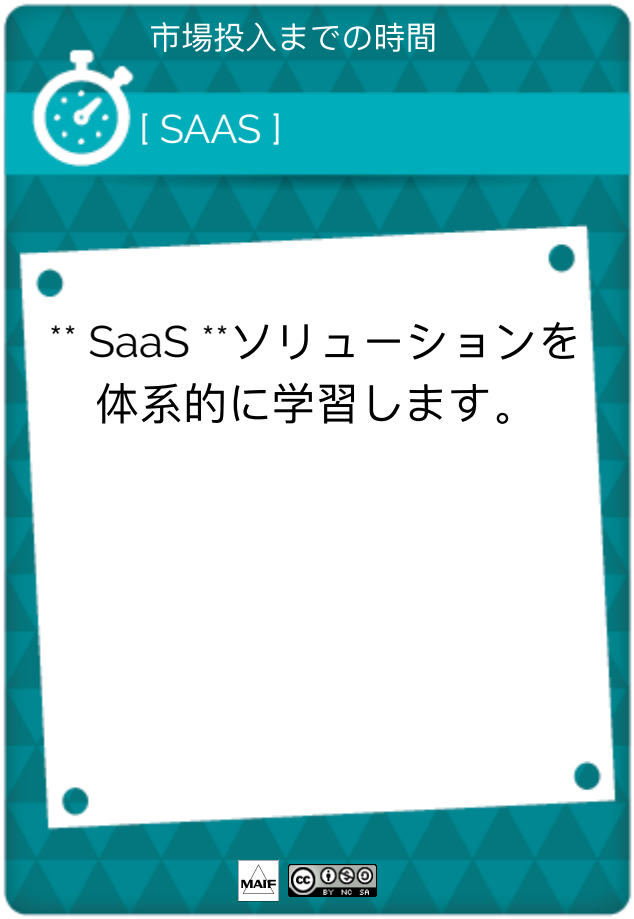
市場投入までの時間

[生産性]

あなたの仕様を裸の必需品に限定し、**どのように "どのように" **に焦点を合わせます。

製品は、**ほとんどの場合、自己文書化されたものでなければなりません。

ドキュメンテーションは、コードと同じ方法でバージョン管理する必要があります。


市場投入までの時間
[SAAS]
** SaaS **ソリューションを
体系的に学習します。
MAIF

市場投入までの時間

[SAAS]

** SaaS **ソリューションは**持続可能でコスト効率に優れています**。

場合によっては、** SaaS **は** MVP **の実装をスピードアップすることができます。**

経済的ビジョンを、**総費用**（** TCO ： T ** otal ** C ** ost ** ** O ** wnership ）だけでなく、ライセンスコストの面で。

市場投入までの時間

[ビジネスの中心]

コアビジネスは、新しいサービスやアプリケーションの構築を妨げるものではありません。

市場投入までの時間

[ビジネスの中心]

コアビジネスの進化と配信のペースは、それを消費するサービスの俊敏性**と互換性がなければなりません。

コアビジネスは**サービス**を公開する必要があります。

コアビジネスは** Event-Driven **の原則を採用しなければならず、イベントの形で管理アクションを報告します。

市場投入までの時間

[継続的な導入]

プロダクションでの展開は
非イベントです。

市場投入までの時間

[継続的な導入]

継続的な展開を活用して、**ビジネス**の要件に** **プロダクション**を適応させます。

環境全体の配備は、**生産**まで**自動**および**頻繁**です。

市場投入までの時間

[PERPETUAL BETA]

永続的なベータアプローチにより、開発プロセスにユーザーを巻き込むことができます。

市場投入までの時間

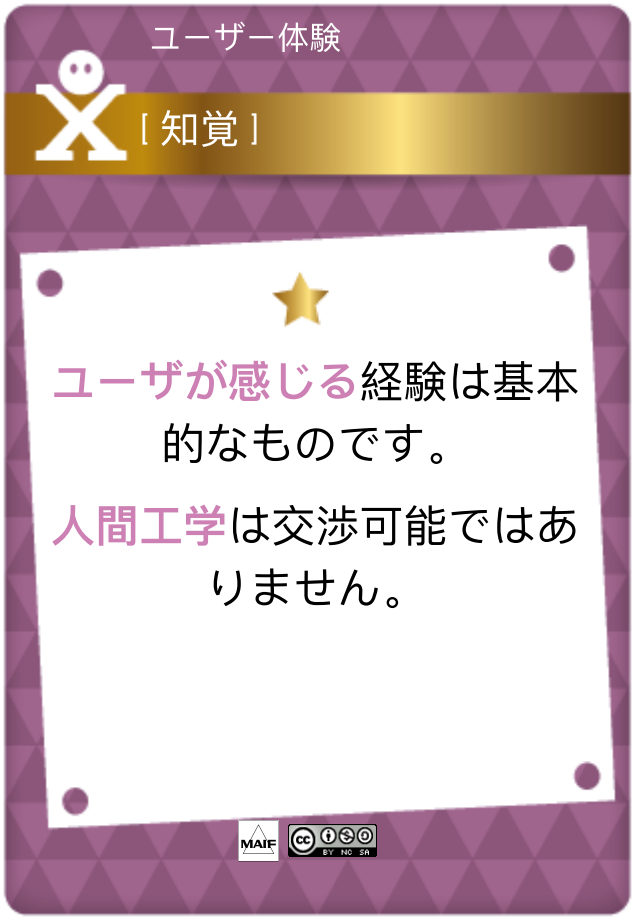
[PERPETUAL BETA]

ユーザが開発に参加する永久ベータの原則を自由に使用してください**。

永久ベータという用語は、ジャストインタイムで開発されたアプリケーションを指します。**常に変化しています**。不完全な製品ではありません。


ユーザー体験
MAIF


ユーザー体験
MAIF

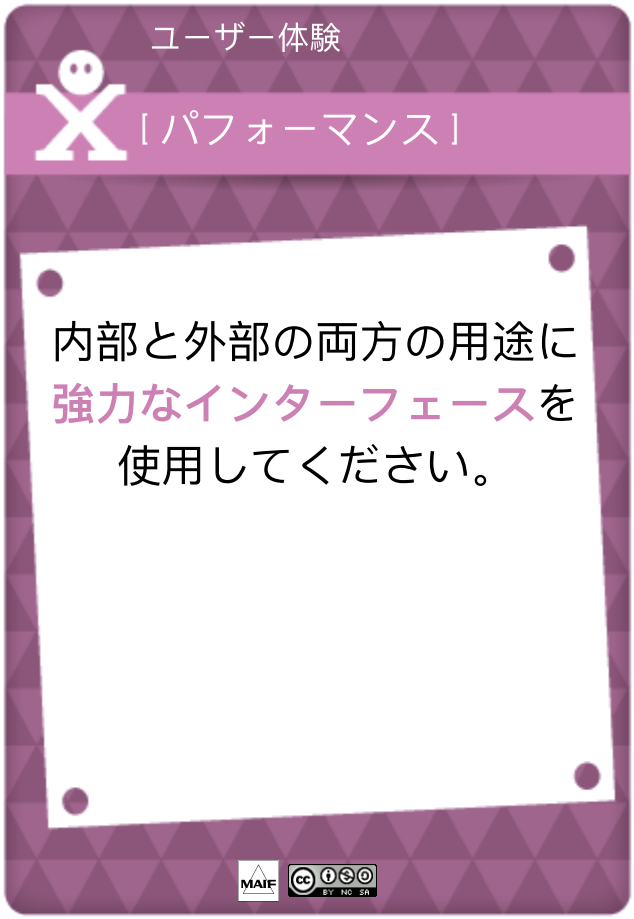

ユーザー体験
[知覚]
ユーザが感じる経験は基本的なものです。
人間工学は交渉可能ではありません。
MAIF

ユーザー体験

[知覚]

** UXデザイナーの仕事を怠ってはいけません**、それはアプリケーション開発の基本です。

あなたのユーザーの**フィードバック**を統合しなさい、これは本質的です。

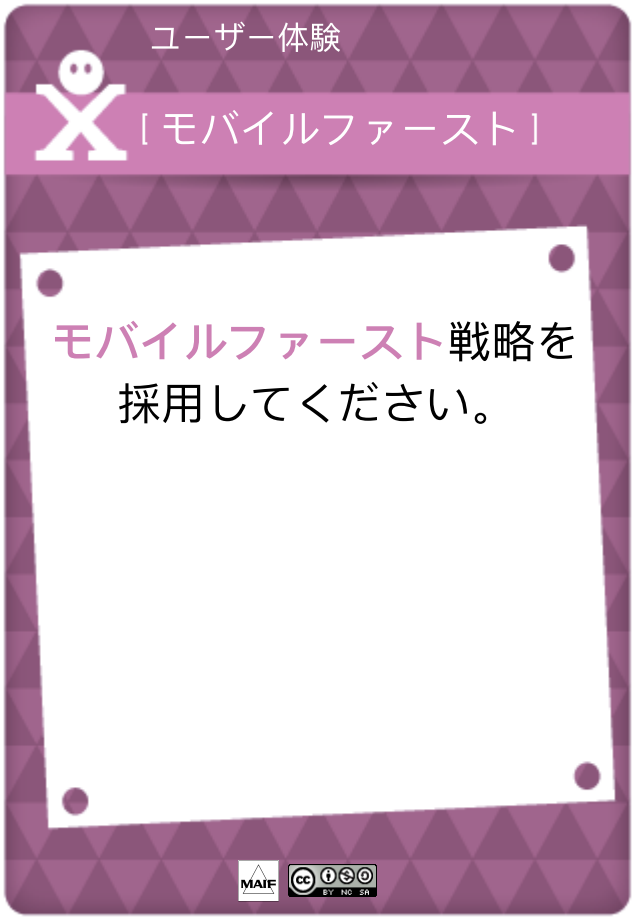

ユーザー体験
[パフォーマンス]
内部と外部の両方の用途に
強力なインターフェースを
使用してください。
MAIF

ユーザー体験

[パフォーマンス]

インターフェイスは**効率**に対応しています。

インターフェイスの**パフォーマンス**は時間を節約し、**ユーザーの満足度**を高めます。したがって、**は欲求不満を救います**。


ユーザー体験
[モバイルファースト]
モバイルファースト戦略を
採用してください。
MAIF

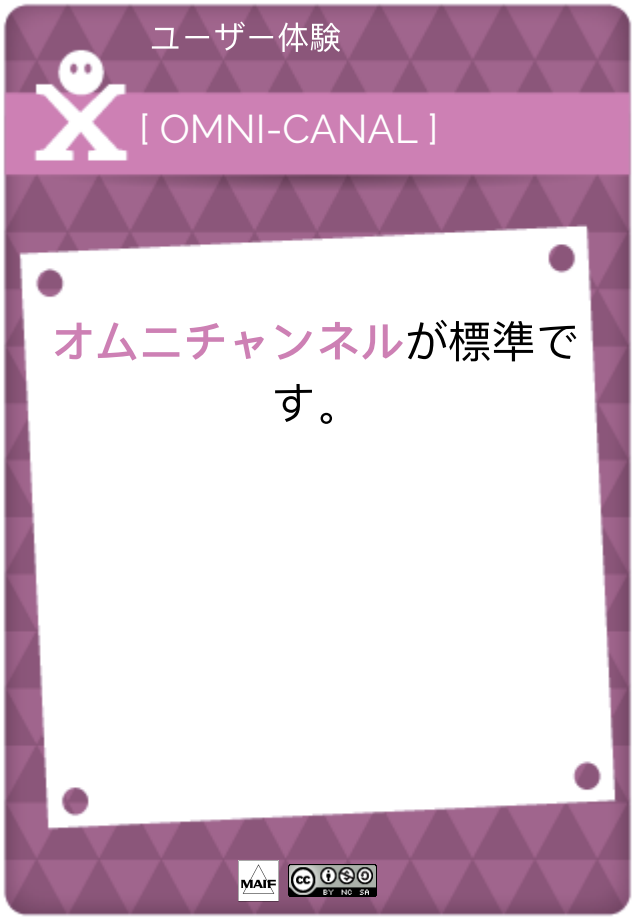
ユーザー体験

[モバイルファースト]

モバイルデバイスは、**市場**の最も重要な**部分です。

Thinking Mobileは**必須**を考えています。

レスポンシブルデザインは標準であり、貯蓄源（** MVP **）です。


ユーザー体験
[OMNI-CANAL]
オムニチャンネルが標準です。
MAIF

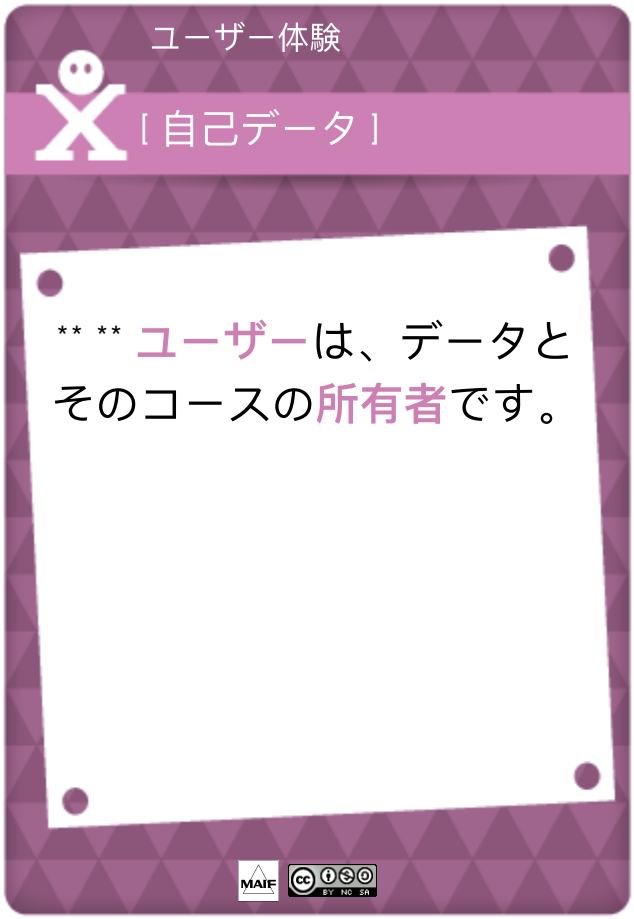
ユーザー体験

[OMNI-CANAL]

オムニチャンネルのアプローチは、ユーザーに**統一されたエクスペリエンス**（例：Netflix）を提供します。

異なる**チャンネル**は**同期**と**コヒーレント**です（バッチ処理とは異なります）。

すべてのアクター（クライアント、アドバイザー）は同じ情報にアクセスします。



ユーザー体験
[自己データ]
** ** ユーザーは、データとそのコースの所有者です。
MAIF

ユーザー体験

[自己データ]

- **個人**データはいつでも**コントロール**にしておいてください。

ユーザーにトレーサビリティと制御をリアルタイムで許可することで**信頼**を確立します。

サブシステムは同じ要件を満たさなければなりません。


ユーザー体験
[CRM / SFA]
顧客との関係は、柔軟で統一されたイベント主導の CRM / SFA によって統一され、文脈化される必要があります。
MAIF

ユーザー体験

[CRM / SFA]

顧客関係とセールスフォースの両方のリーダーシップを管理する** CRM **を選択してください**（SFA： S ** ales ** F ** orce ** A ** utomation ）。

** CRM **は新しい機会に**開かれている**必要があります。

** CRM **はプラットフォームの**イベントドリブン**ロジックに適合する管理アクションに対応する**イベント**を生成します。


ユーザー体験

[ビッグデータ]

ビッグデータプラットフォームではを一元化し、ユーザーデータを処理してあなたの旅に最適に対応できます**。

ユーザー体験

[ビッグデータ]

マイフグループ、パートナーおよび**ベンダー**データを**経路**ロジックに集中します。

"データの準備"と処理** **データを統合することができます。**

ビッグデータチームはフィーチャーチームと協力してデータ**ガバナンスを確保します。


ユーザー体験

[ワークステーション]

ワークステーションは、**との最新のチャンネル**を使用するように適応されています。

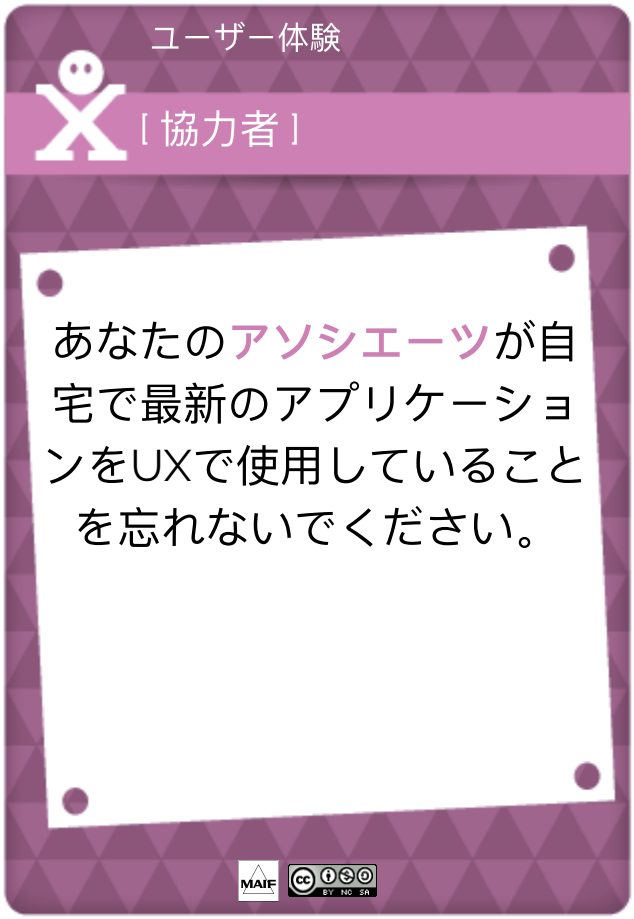
ユーザー体験

[ワークステーション]

統一された経験のために** IDフェデレーション**を採用してください。

ポータルでは**概要**を提供できますが、アプリケーションを置き換えるものではありません。

ワークステーションは、**拡張エンタープライズ**内で開くために**モバイル**、**マルチチャネル**および**標準**である必要があります。

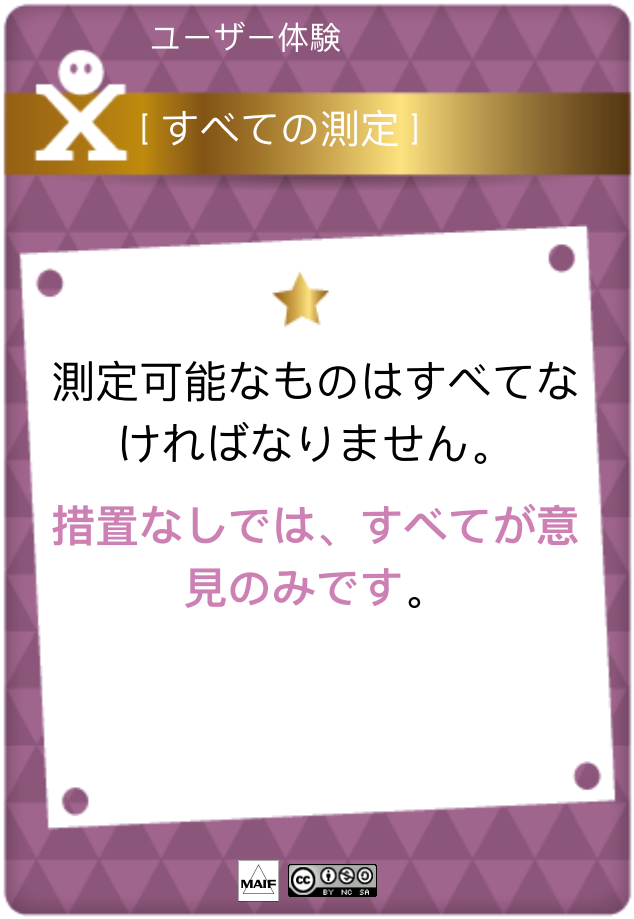

ユーザー体験
[協力者]
あなたのアソシエーツが自宅で最新のアプリケーションをUXで使用していることを忘れないでください。
MAIF

ユーザー体験

[協力者]

すべてのユーザーを「顧客」として扱う**: インターネットユーザー、マネージャー、運用、開発者など

内部使用管理アプリケーション用に実装する** UXの努力**を過小評価しないでください。



ユーザー体験
[すべての測定]
測定可能なものはすべてなければなりません。
措置なしでは、すべてが意見のみです。
MAIF

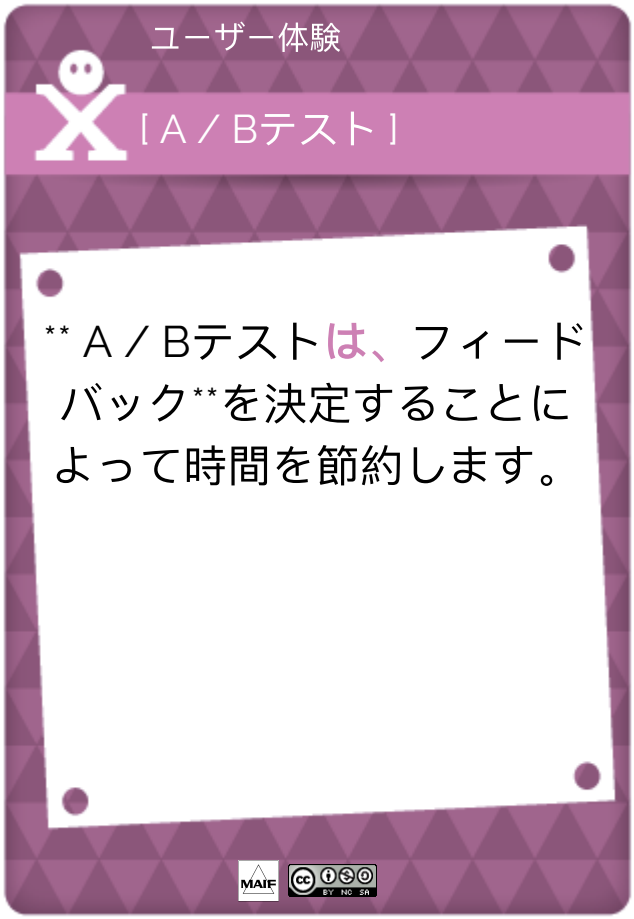
ユーザー体験

[すべての測定]

アプリケーションの**開発中の**メトリックを考えてみてください。**ログ**には**ビジネスだけでなく、技術的な**ディメンションが必要です。

パフォーマンス指標を無視しないでください。これらは基本的なものです。

フィーチャーチームは**操作**を提供します：**アプリケーションを**使用可能にする責任があります。


ユーザー体験
[A / Bテスト]
** A / Bテストは、フィードバック**を決定することによって時間を節約します。
MAIF

ユーザー体験

[A / Bテスト]

2つのソリューションの間で任意に決定するのではなく、** A / Bテストを設定することを躊躇しないでください**。

このパターンは、**同一のアプリケーションの** 2つの異なるバージョン**を提示し**、ユーザ活動の**客観的尺度**に基づいてそれらの1つを選択することからなる。

ユーザー体験

[変質や劣化]

障害が発生した場合のサービス中断ではなく、劣化を考慮してください。

ユーザー体験

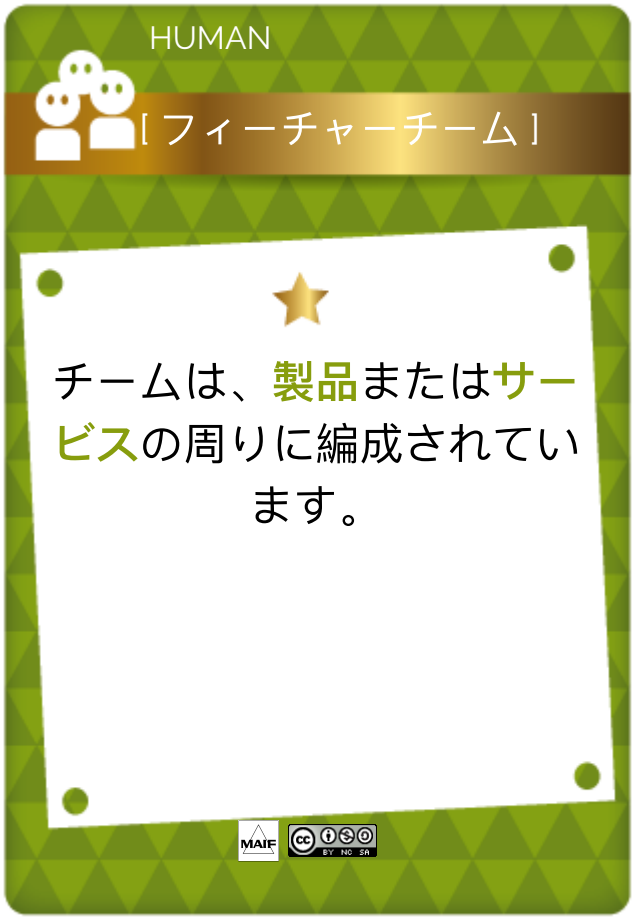
[変質や劣化]

サブシステムのうちの1つの**失敗**では、**サービスの劣化**バージョンは**中断ではなく**最初に考慮されなければなりません**。

サーキットブレーカを使用すると、** **システム**全体に**影響**と**スプレッド**を回避するために**故障を分離できます**。


HUMAN
MAIF


HUMAN
MAIF


HUMAN
[フィーチャーチーム]
チームは、製品またはサービスの周りに編成されています。
MAIF

HUMAN

[フィーチャーチーム]

チームはコヒーレントな機能セットを中心に構成された** Feature Teams **であり、このセットに必要なすべての**スキル**で構成されています。

たとえば、Business Expert + Web Developer + Java Developer + Architect + DBA + Operationalなどです。

責任は**集団**です。機能チームはこの責任のために必要な力を持っています。

HUMAN

[2-PIZZA TEAM]

フィーチャーチーム**のサイズを制限する（5人から12人まで）。

HUMAN

[2-PIZZA TEAM]

フィーチャーチームのサイズを制限する：** 5〜12人**。

5歳未満では、彼女は外部の出来事にはあまりにも敏感で、創造性に欠けています。 12を超えると、生産性が失われます。

「** 2ピザチーム」という用語**は、フィーチャーチームのサイズが2ピザを摂取できる人数を超えてはならないことを示しています。


HUMAN

[アーティザンソフトウェア]

やり方を知っているや**誰がやりたいのか多才な人にベットしてください。

HUMAN

[アーティザンソフトウェア]

最も重要なのは、開発**の**文化、**スケーラビリティ**、**適応性**です。

ソフトウェア職人とフルスタック開発者を募集することで、彼らはノウハウと全体的なビジョンを通じて真の付加価値をもたらします。

それにもかかわらず、モバイル開発者は通常、**専門開発者**です。

HUMAN

[募集]

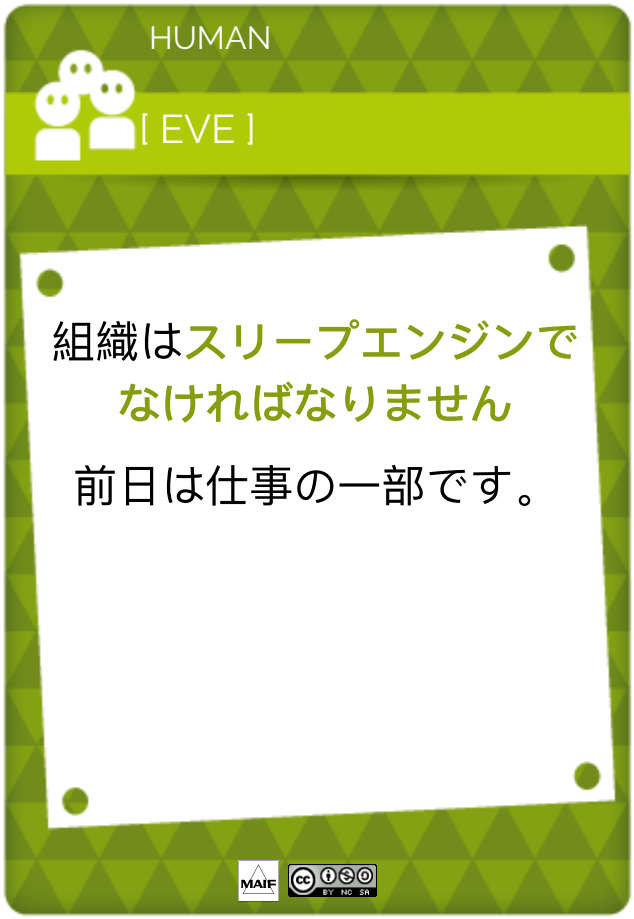
魅力的 最高のを募集する。

HUMAN

[募集]

モビリティ、家庭作業、CYOD
(**C**hoose **Y**our **O**wn **D**evice) 。

実験に時間を費やし、**作業時間にする**。


HUMAN
[EVE]
組織はスリープエンジンで
なければなりません
前日は仕事の一部です。
MAIF

HUMAN

[EVE]

組織は**継続教育**または**ビジネス大学**のようなシステムをセットアップすることによって**デイケア**エンジンでなければなりません。

コーディングDojos *、 * Brown Bag Lunchs 、 External ** Conferencesなどの他のより多くの非公式な**方法と自由に組み合わせてください。

HUMAN

[CO-CONSTRUCTION]

コンバージェンスの目標に賭けて、取引の間の障壁を取り除く。

HUMAN

[CO-CONSTRUCTION]

トレード間の障壁を解消するために、一般的な製品を共通の場所でグループ化するだけでは不十分です。

アジャイルアプローチは、目的のコンバージェンスを確実にするためにこれらの障壁を排除します**。

これらのプラクティスは成功への鍵の不可欠な部分です。組織は保証人です。

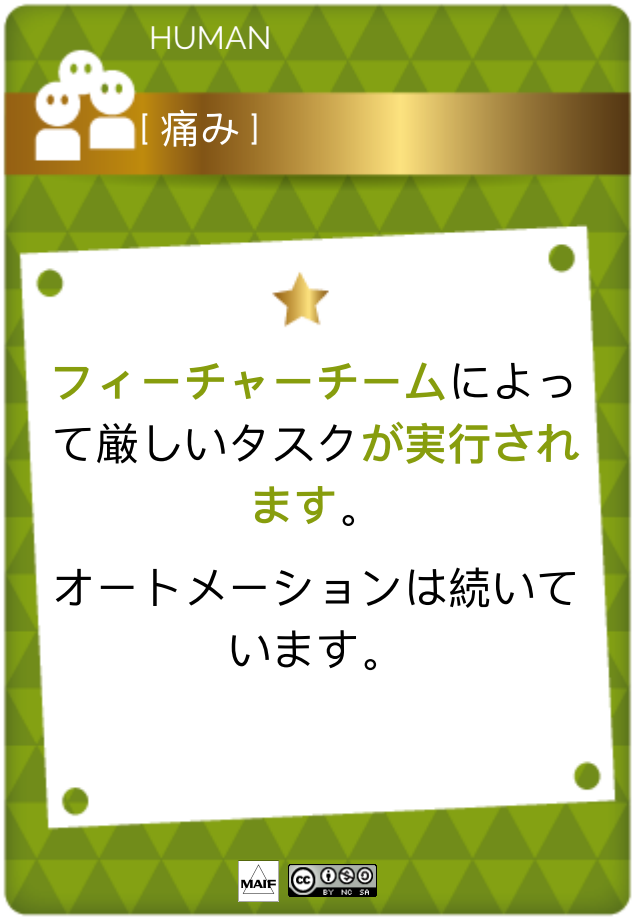

HUMAN
[DEVOPS]
** DevOps **プラクティスでは、壁がBuildとRunの間に入ることができます。
MAIF

HUMAN

[DEVOPS]

共通の目標に向けて** Dev **と** Ops **を統合するには、** DevOps **を採用してください。

取引は変わりません！ DevOpsは、同じ人がDevとOpsのタスクを実行することを意味しません。 **開発者と運用は**** ** **スキルから恩恵を受けるために**共同作業**を要求され**、共感**を改善する必要があります。


HUMAN
[痛み]
フィーチャーチームによって厳しいタスクが実行されます。
オートメーションは続いています。
MAIF

HUMAN

[痛み]

伝統的な組織では、**チーム間の理解の欠如**は、通常、距離やコミュニケーションの欠如**に関連しています。

フィーチャーチームの**メンバーは**すべてのタスクに対して**共同責任**と**固い**です。

痛みは、**継続的改善**の重要な要素です。

HUMAN

[CDS]

サービスセンターは集団コミットメントとは調和しません。

HUMAN

[CDS]

フィーチャーチームは、**コラボレーション**と**集団エンゲージメント**に大きく依存する原則に基づいて構築されています。

サービスセンターは、企業によるITの合理化と統合に向かっており、これは集合的コミットメントのこの概念に反しています。

HUMAN

[検証]

組織は独断的ではなく、検証**の役割を担っています。

HUMAN

[検証]

組織が**ツールと用途で検証の役割を保持していることを確認してください。特に遺産に影響を与えるツール**（例：ソースコードの管理）。

** **の機能チームは、その選択肢をサポートする**を意味します。

独断的ではない 実験を励ますことを忘れないでください**。

HUMAN

[横断]

フィーチャーチームは**コミュニケーション**を期待し、**経験**と**スキル**を共有することが期待されます。

HUMAN

[横断]

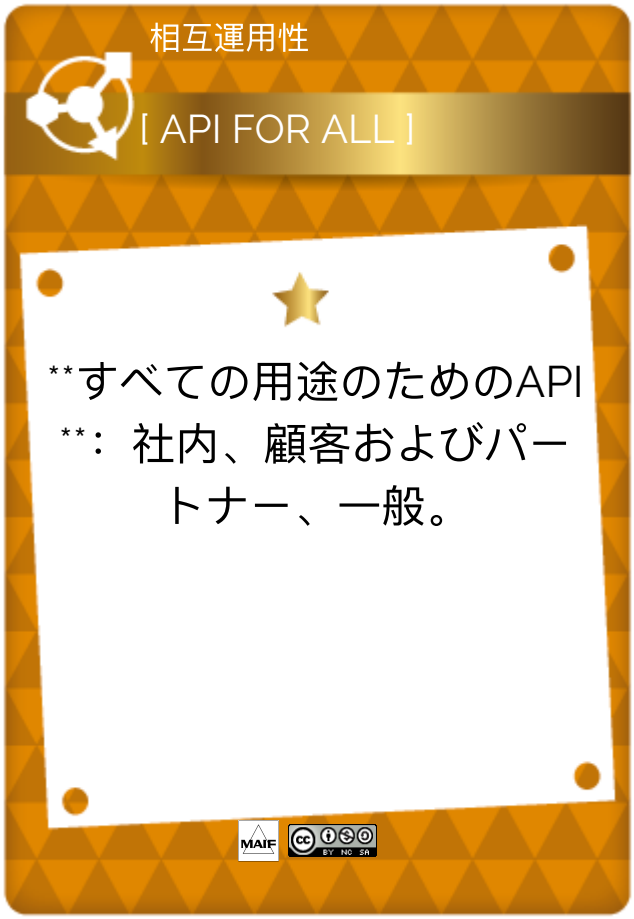
** Feature Teams **の間に障壁を作りません。

フィーチャーチームが互いにコミュニケーションを取り、スキルと経験を共有するために必要な**組織**と**アジリティ**を設定します。

** Spotify **（部族、章および組合）における横断の組織は、雄弁な例です。**


相互運用性
MAIF


相互運用性
MAIF


相互運用性
[API FOR ALL]
すべての用途のためのAPI：社内、顧客およびパートナー、一般。
MAIF

相互運用性

[API FOR ALL]

**公開API **を使用して、組織を新しい用途や新しい顧客に開放します。

商用パートナーシップ、顧客** **プロバイダー**では、APIは標準的な交換フォーマットです。

** API **は、組織の内部使用**にも使用されることを意図しています。

相互運用性

[セルフサービス]

APIを使用するには、**シンプル**と**高速**を使用する必要があります。

相互運用性

[セルフサービス]

APIの使用は可能な限り単純でなければなりません。 **開発者の経験を考えてください。**

必要性の妥当性を検証する最善の解決策は** APIを素早くテストすることです**: 数分で十分です!

プラットフォームはAPIを簡単にテストするための**グラフィカルインタフェース**を提供する必要があります。

相互運用性

[API管理]

APIの使用は、**制御**および**制御**でなければなりません。

相互運用性

[API管理]

クォータ、**スロットル**、**認証**、**ロギング**を管理するAPI管理ソリューションを実装します。

監視、**フィルタリング**、**レポート**を管理するためのメトリックを収集します。

相互運用性

[要件]

プラットフォームに組み込まれた**外部システムおよびサービス**の**要件**を設定します。

相互運用性

[要件]

内部システムと同じ**要件**を満たすために**外部システム**を必要とします。

外部システムは**イベント**を公開し、**技術**モニタリングを許可する必要があります。

外部システムデータを統合する必要がある場合、**合計**同期は**可能でなければなりません。**

相互運用性

[多人数テナント]

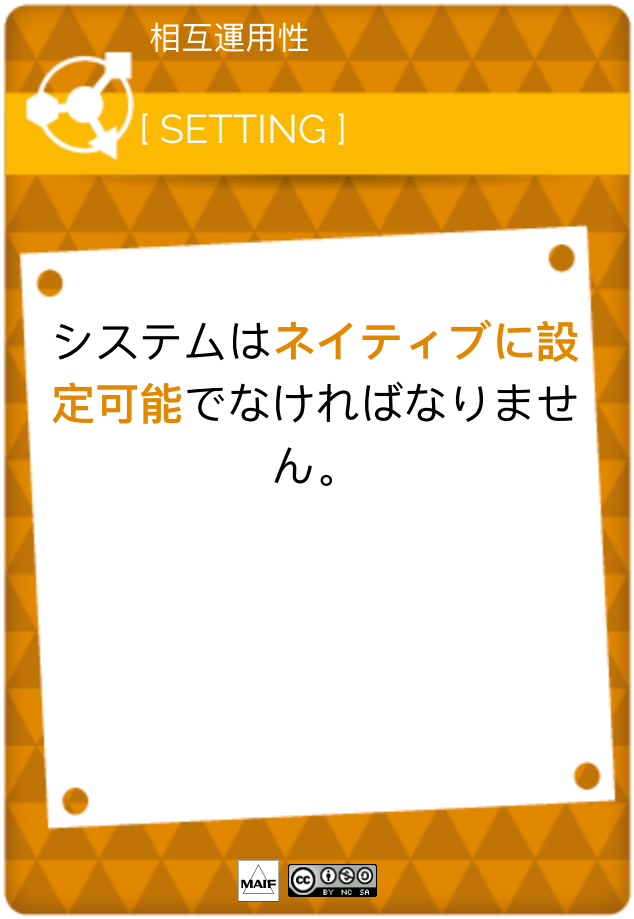
アーキテクチャは**マルチテナント**と考える必要があります。

相互運用性

[多人数テナント]

ホワイトマークがベースで考慮されていない場合でも、マルチテナントアーキテクチャを設定します。あなたの**最初の**アプリケーションは、最初の**保持**です。

システムの**多機能インスタンシエーション**を最初から考えてください。


相互運用性
[SETTING]
システムはネイティブに設定可能でなければなりません。
MAIF

相互運用性

[SETTING]

言語、通貨、ビジネスルール、セキュリティプロファイルは設定が簡単でなければなりません。

ハイパージェネリックに注意してください、それはしばしば役に立たず、**コストソース**です。

セットアップは**スケーラブル**で、必要に応じて高速にする必要があります。

相互運用性

[フィーチャー・フリッピング]

機能の反転を使用して柔軟性の高いシステムを作成します。

相互運用性

[フィーチャー・フリッピング]

機能の反転は、アプリを**機能**のセットとしてデザインすることです** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **使用可能にすることができます

マルチテナントアプリケーションでは、フィーチャーを反転させると**サポーターをカスタマイズできます。**

簡単なA / Bテストの特徴を反転します。


ゲームのルール
MAIF


ゲームのルール
MAIF

ゲームのルール

[技術的選択]

技術的な選択肢は、フィーチャーチームによって** ** ** ** ** **は想定されています。

ゲームのルール

[技術的選択]

フィーチャーチームは、**責任ある行動**を行い、排他的に影響を与える選択肢と組織に影響を与える選択肢を特定する必要があります。

フィーチャーチームの範囲**を超える**選択肢**（ライセンス、まれなプログラミング言語など）は、組織またはピアコンバージェンスプロセスによって**検証されなければなりません**。

ゲームのルール

[良い使用]

正しい使い方のための正しいツール**は節約の源です。

ゲームのルール

[良い使用]

すべての人に課された悪いツールは**リスクです。良いツールの**誤用**は**非常に**有害な**結果**を持つことができます。たとえば、あまり使用されていないアジャイルメソッドは危険です。

ツールは**質問**する必要があります。

** Excel **はしばしば合理的な選択肢**ですが、**すべてを行うための**
** （CRM、ERP、Datamart、…）

ゲームのルール

[BUILD VS.購入]

特権コアビジネスのためのビルド

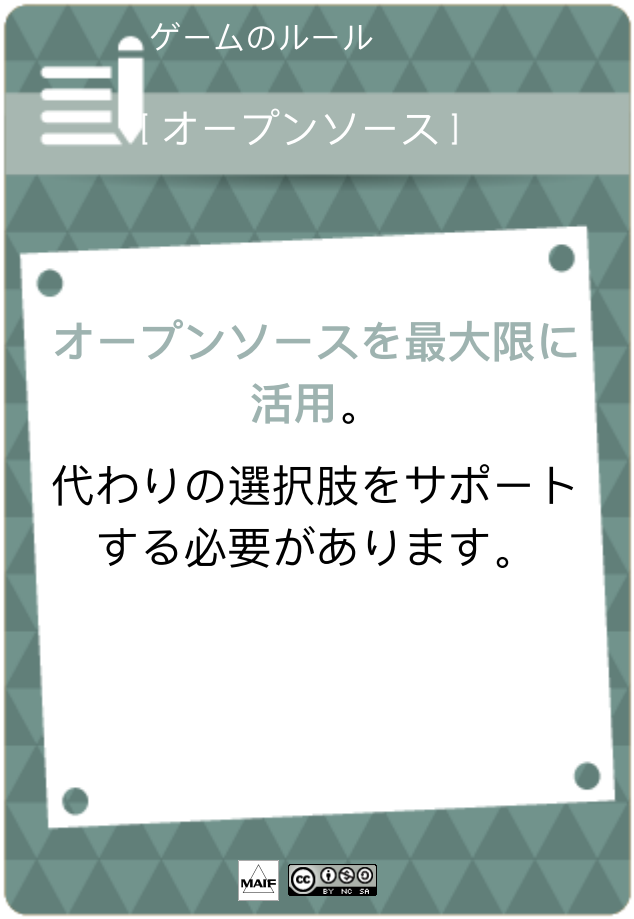
その他の場合は、購入を考慮してください。

ゲームのルール

[BUILD VS.購入]

ツールが組織の**機能を差別化する**機能を備えているほど、**構築される方が多くなります**。コアビジネスは**特異性**を可能にする必要があり、**は迅速かつ頻繁に適応**しなければなりません。いくつかの**ソフトウェアパッケージ**は、**このニーズに合わせて**適応されることがあります。

** **その他の場合**：SaaS、オープンソース、ビルド、オーナーは**ケースバイケース**で学びます。


ゲームのルール
[オープンソース]
オープンソースを最大限に活用。
代わりの選択肢をサポートする必要があります。
MAIF

ゲームのルール

[オープンソース]

プロプライエタリなソリューションは、必要に応じてメンテナンスを再開できる必要がある組織の**リスク**です。

オープンソースの代替手段を持たないプロプライエタリなツールはほとんどありません。

組織は**オープンソースコミュニティーのメリット**を提供し、**は貢献することができます。**

ゲームのルール

[マイクロサービス]

スタンドアロンおよび弱く結合したサービスを開発する。

ゲームのルール

[マイクロサービス]

弱いカップリングは標準でなければなりません。

各マイクロサービスは**明確に定義されたインターフェース**を持っています。

この**インターフェース**は**マイクロサービス**間の**リンク**を決定します。

ドメイン駆動設計では、特に**有界コンテキスト**でこの問題を予期することができます。

ゲームのルール

[DATA]

各サービスには独自の**データストレージシステムがあります。

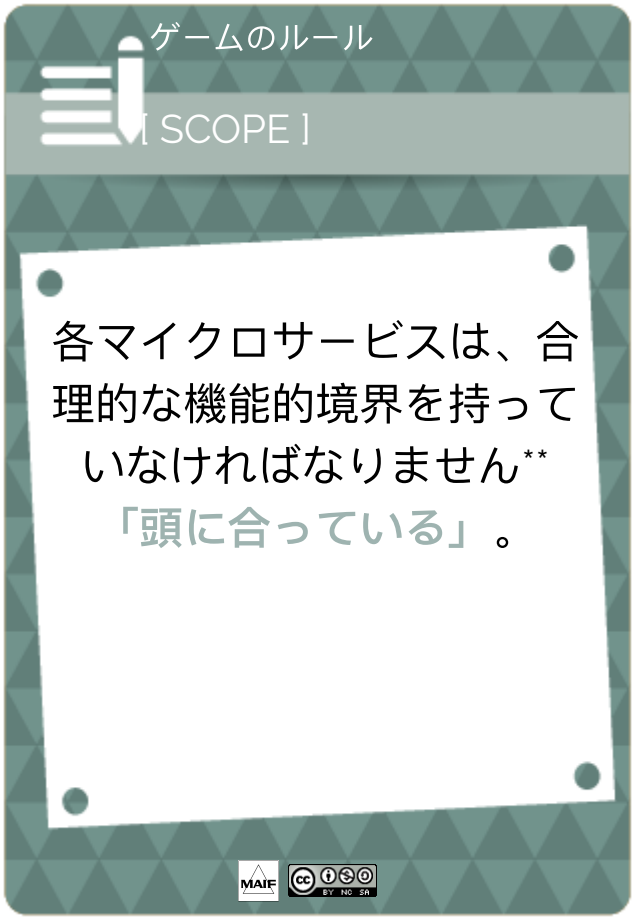
ゲームのルール

[DATA]

データストアは、**単一のマイクロサービス**とのみ**結合**されることを意図しています。

あるマイクロサービスから別のマイクロサービスへの**データへのアクセス**は、そのインターフェース**を介して排他的に**行われます。

この設計は**プラットフォーム全体の時間の経過と共に**一貫性を意味します。 UXを含む**すべてのレベルで**認識されなければなりません。


ゲームのルール
[SCOPE]
各マイクロサービスは、合理的な機能的境界を持っていなければなりません**
「頭に合っている」。
MAIF

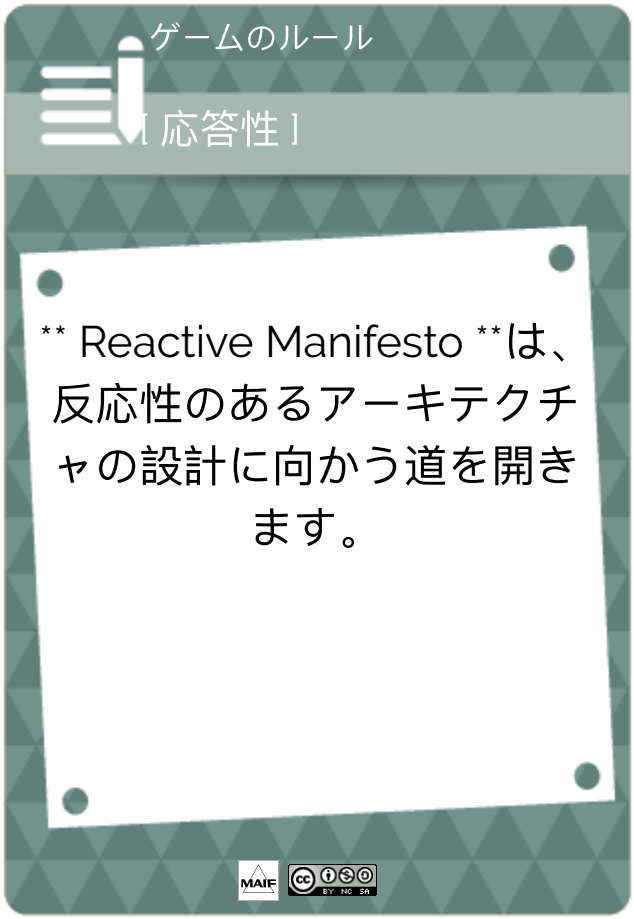
ゲームのルール

[SCOPE]

マイクロサービスは**合理的な数の機能**を提供します。

成長が始まると、マイクロサービスを切断するのをためらってください。

リーズナブルなサイズのサービスは、**必要に応じて**書き換え**を穏やかに検討することを可能にします。

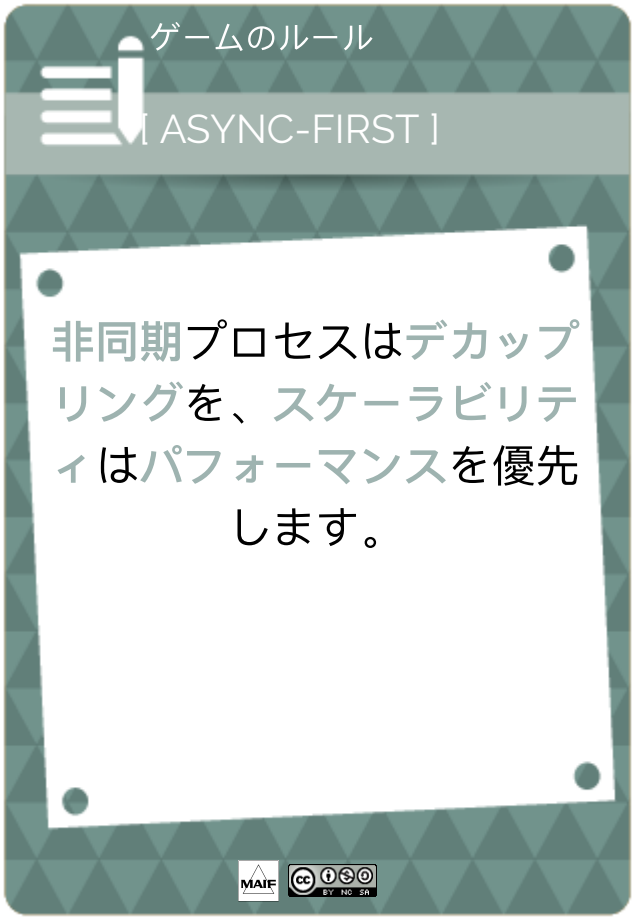

ゲームのルール
[応答性]
** Reactive Manifesto **は、反応性のあるアーキテクチャの設計に向かう道を開きます。
MAIF

ゲームのルール

[応答性]

レスポンシブなプログラミングは、データの流れと変化の伝播に焦点を当てています。これは、より伝統的なアプローチ "**反復子**"に反するパターン "** Observer **"に基づいています。

リアクティブマニフェストは基本的な軸を設定します：**可用性**とスピード、**反発力**、柔軟性、弾力性、メッセージオリエンテーション**。


ゲームのルール
[ASYNC-FIRST]
非同期プロセスはデカップリングを、スケーラビリティはパフォーマンスを優先します。
MAIF

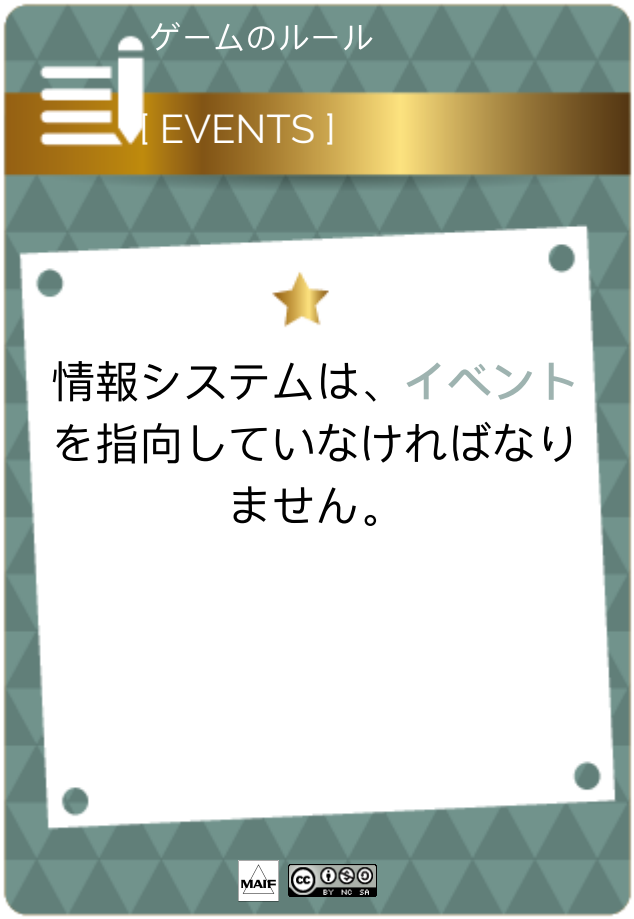
ゲームのルール

[ASYNC-FIRST]

アプリケーション間の交換は**非同期**でなければなりません。

非同期交換は自然に**弱い**カップリング、**分離**、**フロー制御（背圧）**を可能にします。

同期通信は、アクションがそれを必要とするときのみ**考慮されるべきです。

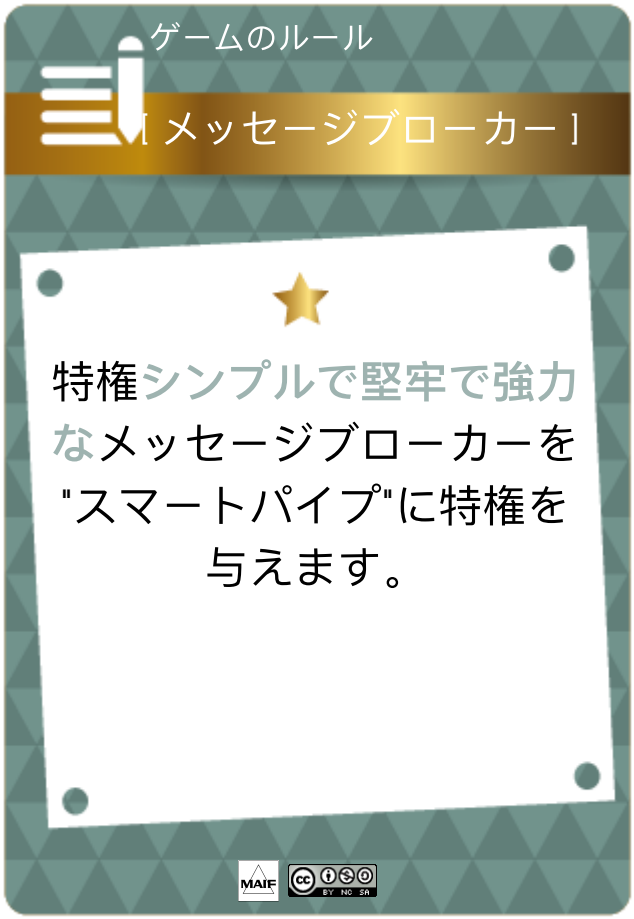

ゲームのルール
[EVENTS]
情報システムは、イベントを指向していなければなりません。
MAIF

ゲームのルール

[EVENTS]

** **イベント駆動型**機能**プロセス**は**自然に 非同期に**実装されています。

イベントオリエンテーションは、** C と Q ** uery ** R **の可能性** S ** egregation（** CQRS ** ）と** Event Sourcing **です。


ゲームのルール
[メッセージブローカー]
特権シンプルで堅牢で強力なメッセージブローカーを"スマートパイプ"に特権を与えます。
MAIF

ゲームのルール

[メッセージブローカー]

** ESB **は限界を示しました：**スケーラブルなメンテナンス**は技術と**組織**の観点の両方から**重要**です。

** **カフカ**のようなブローカー**メッセージは、シンプルな、耐久、弾力のある**ソリューションを提供します。

スマートエンドポイントと**シンプルパイプ**は規模で動作するアーキテクチャです。**インターネット**です。

ゲームのルール

[TIMING]

システムの**完全な同期**は、
** **設計されるとすぐに考えられるべきです**。

ゲームのルール

[TIMING]

イベントフローによって2つのシステム間の**同期**が保証されている場合、これらのシステムの合計**再同期**は**設計時**に計画する必要があります。

自動** ** **同期監査**（例：サンプル別）は、**測定**および**可能な**同期エラー**の検出**を可能にします。

ゲームのルール

[一元化]

サービスの構成は集中、発見はディレクトリによって保証されています。

ゲームのルール

[一元化]

マイクロサービスの構成はすべての環境のための集中です。

中央ディレクトリはマイクロサービスの動的検索を保証します。

** **グローバルスケーラビリティはこのディレクトリ**に依存します。**

ゲームのルール

[サンドトレイ]

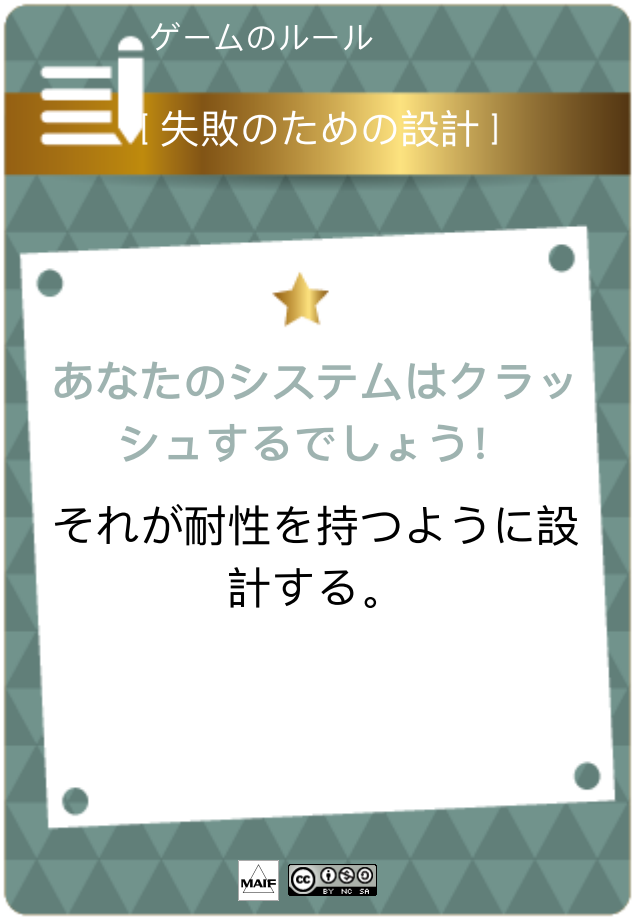
フィーチャーチームはサンドボックス環境を提供します。

ゲームのルール

[サンドトレイ]

フィーチャーチームは**サンドボックス**環境（現在のバージョンと今後のリリース）を維持して、他の**チームを拡大することができます。**

いくつかの非名義のケースでは、**機能**は**開発**環境で**無効**になることがあります。


ゲームのルール
[失敗のための設計]
あなたのシステムはクラッシュするでしょう!
それが耐性を持つように設計する。
MAIF

ゲームのルール

[失敗のための設計]

あなたの**システムは失敗します**、それは避けられません。このために設計されていなければなりません（** Design For Failure **）。

ハードウェア（ネットワーク、ディスクなど）、**アプリケーション**（複数のアプリケーションのインスタンス）、**地理的なゾーン**プロバイダ**（例：AWS + OVH）。

ゲームのルール

[ツールキット]

** toolkits **を提供し、厳格な枠組みを課してはいけません。

ゲームのルール

[ツールキット]

テクニカルコンポーネントの住宅および横断への注意！それらは制限的で、費用がかかり、維持するのが難しい。

アクセラレータ、**ツールキット**、**テクニカルスタック プール**、**無料**フィーチャーチーム、独断的なアプローチを避ける。

ゲームのルール

[雲]

パブリック、プライベートまたはハイブリッドのクラウド（** IaaS または PaaS **）は、制作の標準です。

ゲームのルール

[雲]

** PaaS **サービスは**優先、簡単**、および規模をすばやく調整できます。

** IaaS **サービスでは**、柔軟性**を高める必要があるケースに対処できますが、より多くの運用作業が必要になります。

プライベートクラウドは従来の仮想化環境ではなく、**コモディティハードウェア**に依存しています。

ゲームのルール

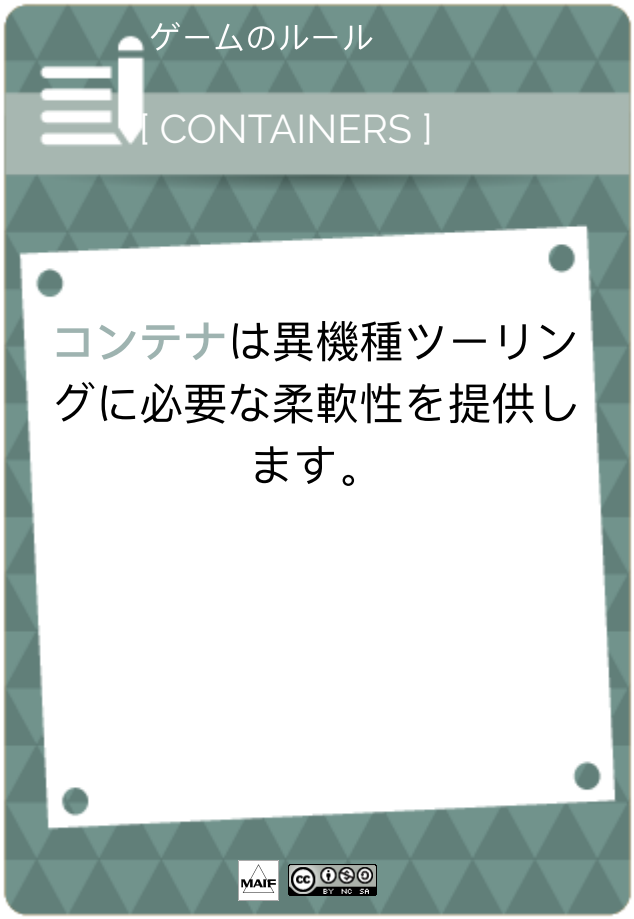
[インフラ]

フィーチャーチームはインフラストラクチャを管理していませんが、組織によって提供され管理されています。

ゲームのルール

[インフラ]

インフラストラクチャの問題は** Feature Teams **にありません。インフラストラクチャは**機能×サービス**によって**提供され**維持**されなければなりません。



ゲームのルール
[CONTAINERS]
コンテナは異機種ツーリングに必要な柔軟性を提供します。
MAIF

ゲームのルール

[CONTAINERS]

コンテナは**異質なツール**を**同種のコンテキスト**で可能にするためにフィーチャーチームが必要とする**柔軟性**を提供します。

ゲームのルール

[ENVIRONMENTS]

容器の使用は、技術環境の問題を克服することを可能にする。

ゲームのルール

[ENVIRONMENTS]

コンテナ（例：** Docker ）**は、環境の違いを** **解放することを可能にします。

配備プロセスは環境に対して**不可知論的**でなければなりません。

データベースなどの一部のコンポーネントはコンテナに配置しないでください。彼らの展開はまだ自動化されています。

ゲームのルール

[METRIC]

対策はすべてに中央および
アクセス**する必要があり
ます。

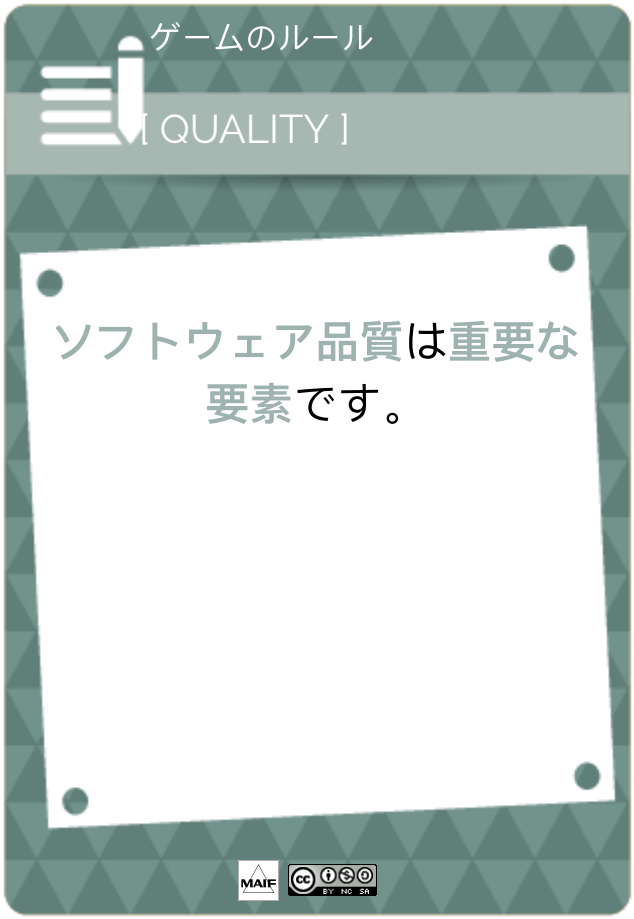
ゲームのルール

[METRIC]

指標は、異なるレベルの細かさを持つすべての人に**アクセス可能**です。関連するチームフィーチャーの詳細ビュー、組織の他のメンバーの集計。

指標へのアクセスはユニットデータへのアクセスを意味するものではなく、機密性を維持するように制御されなければなりません。

すべての環境が影響を受けます。


ゲームのルール
[QUALITY]
ソフトウェア品質は重要な要素です。
MAIF

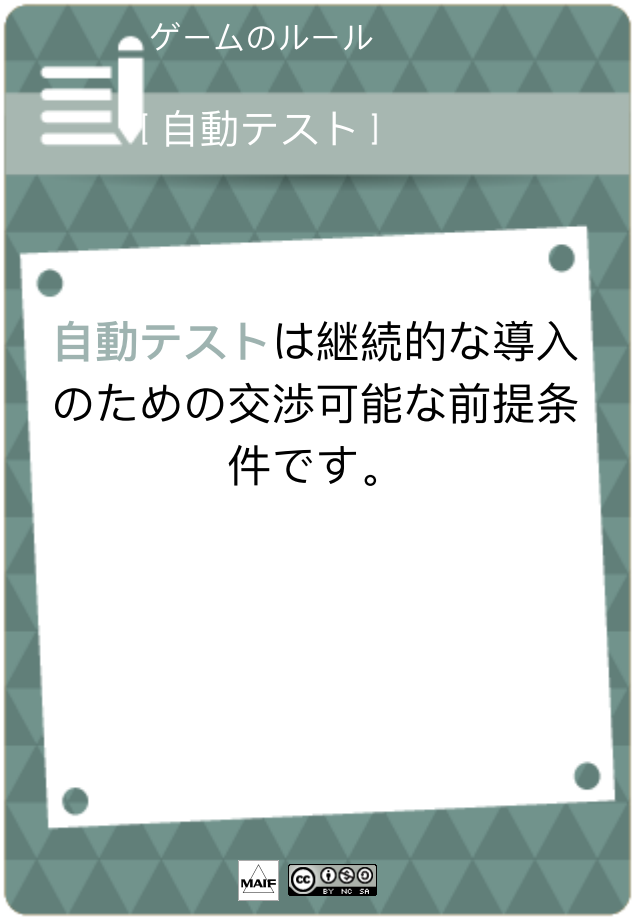
ゲームのルール

[QUALITY]

コードレビューは**体系的**です。 **継続的改善**の一環として、フィーチャーチームのメンバーまたは組織の他のメンバーが実施します。

あなたが監査されているのではなく、あなたのコード: "あなたはあなたのコードではありません！"

光度測定は部分的に自動化できますが、** "新しい目" **に勝るものはありません。**


ゲームのルール
[自動テスト]
自動テストは継続的な導入のための交渉可能な前提条件です。
MAIF

ゲームのルール

[自動テスト]

自動**テスト** **時間の経過とともに**製品**の**品質**を保証します。

継続的な導入には**前提条件**があります。** **変更**および**頻繁な展開**が可能です。

プロダクションの公開は**逸話**イベントになります!

ゲームのルール

[テストレベル]

すべてのレベルでのテスト：ユニット、統合、機能性、反発力、パフォーマンス

ゲームのルール

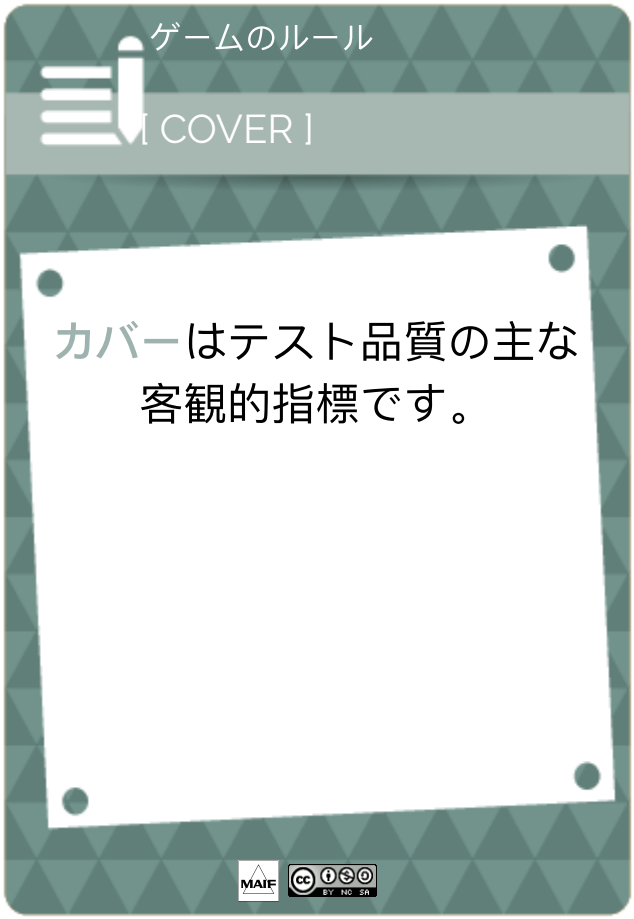
[テストレベル]

統合および**機能**テストは最も重要です** **保証** ** **効果的な**運用**

ユニットテストは**開発**に適しています。

パフォーマンスは時間の経過とともにパフォーマンスを測定します**。

弾力性テストは**失敗を予測するのに役立ちます。**


ゲームのルール
[COVER]
カバーはテスト品質の主な
客観的指標です。
MAIF

ゲームのルール

[COVER]

テストによる**コードカバレッジ**は、コード品質の**良い**メトリックです。

これは、**必要な条件です**しかし、**十分ではありません**、コードの良質を保証することなく、**悪い**テスト戦略の適用範囲が高くなります。

ゲームのルール

[SAFETY]

セキュリティはプロセスです。問題に応じて対処すべきではありません。

ゲームのルール

[SAFETY]

セキュリティ専門家はフィーチャーチームに直接統合することができます** **必要に応じて。**

セキュリティ専門家は、**監査**、**認識**、**フォワード**のために組織内で利用可能です。