



IBM Bluemix
www.bluemix.net



Bluemix DevOps Services概要

Version 1.1



2017年 7月 5日

日本アイ・ビー・エム 株式会社
IBMクラウド事業本部
第二クラウド・テクニカル・セールス
樽澤広亨 HIROT@jp.ibm.com

免責事項

ワークショップ、セッション、および資料は、IBMまたはセッション発表者によって準備され、それぞれ独自の見解を反映したものです。それらは情報提供の目的のみで提供されており、いかなる参加者に対しても法律的またはその他の指導や助言を意図したものではなく、またそのような結果を生むものでもありません。本講演資料に含まれている情報については、完全性と正確性を期するよう努力しましたが、「現状のまま」提供され、明示または暗示にかかわらずいかなる保証も伴わないものとします。本講演資料またはその他の資料の使用によって、あるいはその他の関連によって、いかなる損害が生じた場合も、IBMは責任を負わないものとします。本講演資料に含まれている内容は、IBMまたはそのサプライヤーやライセンス交付者からいかなる保証または表明を引きだすことを意図したものでも、IBMソフトウェアの使用を規定する適用ライセンス契約の条項を変更することを意図したものでもなく、またそのような結果を生むものでもありません。

本講演資料でIBM製品、プログラム、またはサービスに言及していても、IBMが営業活動を行っているすべての国でそれらが使用可能であることを暗示するものではありません。本講演資料で言及している製品リリース日付や製品機能は、市場機会またはその他の要因に基づいてIBM独自の決定権をもっていつでも変更できるものとし、いかなる方法においても将来の製品または機能が使用可能になると確約することを意図したものではありません。本講演資料に含まれている内容は、参加者が開始する活動によって特定の販売、売上高の向上、またはその他の結果が生じると述べる、または暗示することを意図したものでも、またそのような結果を生むものでもありません。パフォーマンスは、管理された環境において標準的なIBMベンチマークを使用した測定と予測に基づいています。ユーザーが経験する実際のスループットやパフォーマンスは、ユーザーのジョブ・ストリームにおけるマルチプログラミングの量、入出力構成、ストレージ構成、および処理されるワークロードなどの考慮事項を含む、数多くの要因に応じて変化します。したがって、個々のユーザーがここで述べられているものと同様の結果を得られると確約するものではありません。

記述されているすべてのお客様事例は、それらのお客様がどのようにIBM製品を使用したか、またそれらのお客様が達成した結果の実例として示されたものです。実際の環境コストおよびパフォーマンス特性は、お客様ごとに異なる場合があります。

IBM、IBM ロゴ、ibm.com、Bluemixは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれIBMまたは各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtmlをご覧ください。

Windowsは Microsoft Corporationの米国およびその他の国における商標です。
JavaおよびすべてのJava関連の商標は Oracleやその関連会社の米国およびその他の国における商標または登録商標です。

目次

- ❑ SoEを具現化するためのアプローチ
～クラウド・ネイティブな開発スタイルとは何か～
- ❑ Microservices概要
- ❑ クラウド・ネイティブなアプリケーション・ライフサイクル管理
～継続的デリバリーとOpen Toolchains～
- ❑ Bluemix Garageのご紹介

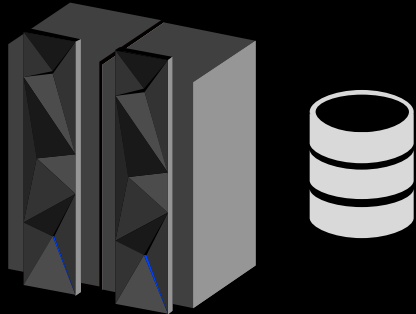
SoEを具現化するためのアプローチ ~クラウド・ネイティブな開発スタイルとは何か~



マルチ・スピード化するITシステム

クラウド・テクノロジーの適用ドメイン

Systems of Record



オペレーションの確実性

ミッション・クリティカルな
アプリケーションの管理, 運用

**安全・確実な開発運用
変わらぬ普遍性・信頼性**

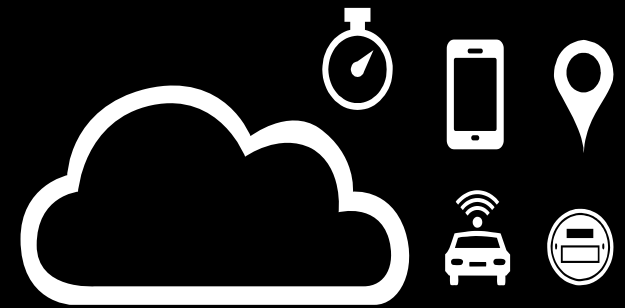
Systems of Insight



ビジネス機会の創造

ビジネス成長のための
消費者動向の洞察

Systems of Engagement



イノベーションのスピード

マーケット開発のための
探索, 開発, デプロイ

**素早いスピード
継続的な変化**

ギャップ

Systems of Engagementに求められる要件

スピード

~素早いITシステム開発~

柔軟性

~容易なシステム変更・保守~

スピードと柔軟性を兼ね備えた

- IT基盤を実現するのがクラウド・ソリューション
- 業務を実現するのがクラウド・ネイティブ・アプリケーション

クラウド・ネイティブなユースケース：スモール・スタート

リスクを最小化しつつ、投資を確実にビジネス成長につなげるには・・・

ビジネスの視点

マーケットに受け入れられているか、逐次フィードバックを受けながら、段階的にビジネスを拡張する

ITの視点

エンドユーザーのニーズを把握し、本当に使ってもらえるという確証を得るまでは、ITシステム開発への投資を最低限に抑える



リスクを抑え、投資対効果を最大化するには、
ビジネスのみならずITシステム開発も
インクリメンタル開発が有効なアプローチです

モノづくりの様々な手法

ダム



Waterfall

- ダムはOne off (1回だけの作業)の建造物
- 自動車は, その基本構造含め, 機能改善を繰り返す

自動車

クラウド・ネイティブ的



Incremental

クラウド・ネイティブな開発スタイル：インクリメンタル開発

□インクリメンタル開発

- **短期間・高頻度**で変わるビジネス・ニーズに見合ったITシステム構築を段階的に実現

□インクリメンタル開発に必要な要素

- **素早い構築・開発**
- **容易な変更や追加**

□インクリメンタル開発を支える環境

○クラウド

- インクリメンタル開発を支える基盤技術

○マイクロサービス

- インクリメンタル開発を支えるアプリケーション開発手法

○継続的デリバリー

- インクリメンタル開発を支えるプラクティスとプリンシプル

- ・ インクリメンタル開発することで、クラウドの価値が享受できる
- ・ 特に投資対効果の観点で、One Offの開発では、十分な効果は得られない

Microservices概要



Microservices

□ クラウド上で稼働するITシステム開発プロジェクトからのフィードバックを集めたもの

- James Lewis & Martin Fowler
- Martin FowlerのWebサイト(Bliki)で発表
 - <http://martinfowler.com/articles/microservices.html>

□ マイクロサービス・アーキテクチャ・スタイル

- **小さなサービスを組み合わせ**て, 一つのアプリケーションを開発する
- 各サービスは, **それぞれ独立したプロセス**で動作する
- 各サービスは, **RESTのような軽量な仕組みで通信**する
- 各サービスは, **完全に自動化された仕組みで, それぞれ個別にデプロイ(配備)**される
- サービスは, それぞれ**異なるプログラミング言語**で実装することができるし, **異なるデータ・ストレージ**を利用できる

□ 動機

- **チーム開発, 各アプリケーション個別の保守, 繰り返し開発**を, 促進するためのモジュラー構造(マイクロサービス)を導入
 - 一枚岩アプリケーション(非マイクロサービス・アプリケーション, モノリス)は, 保守性に劣り, 繰り返し開発を阻害する

Microservices : 9つの特徴

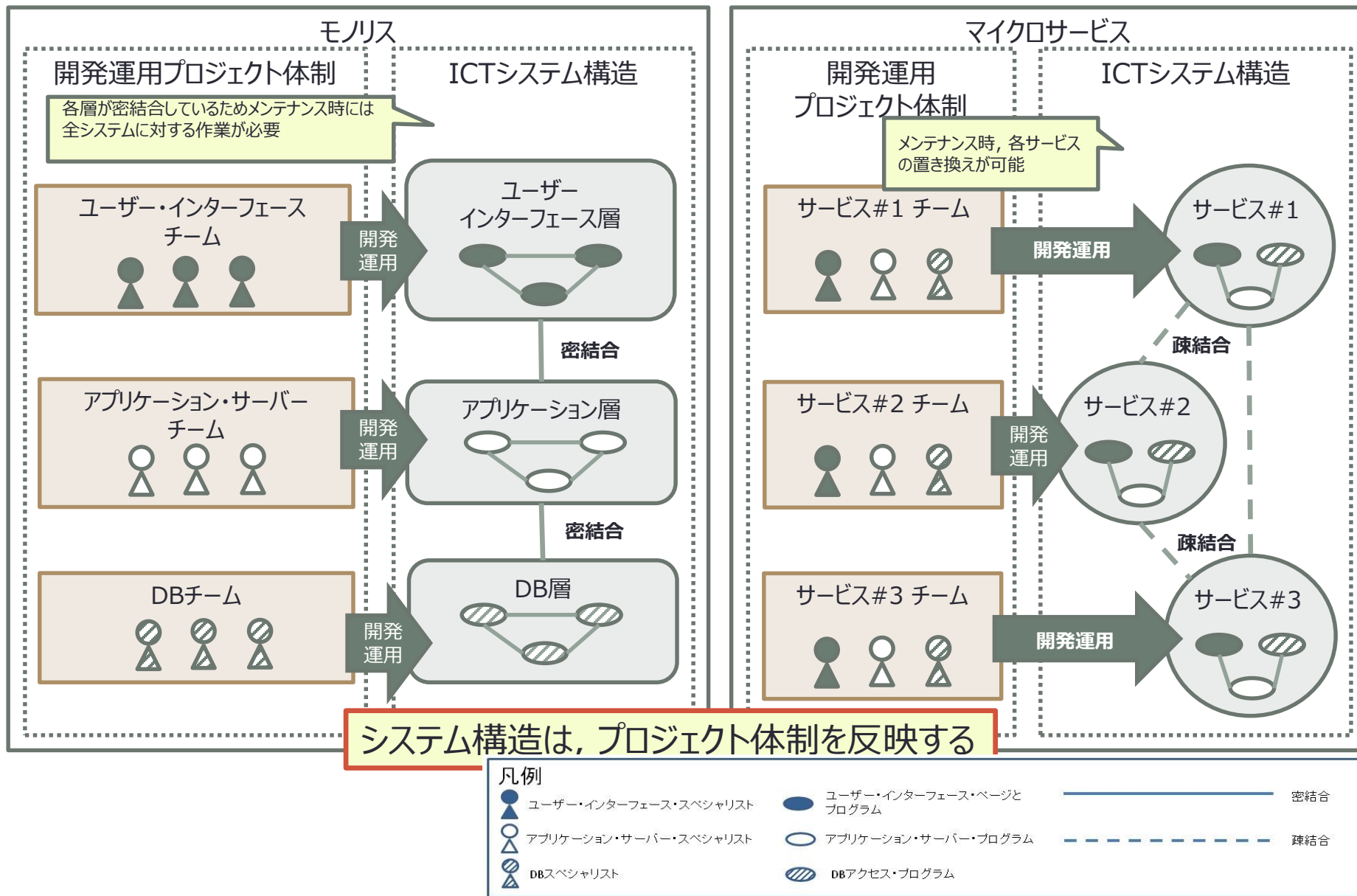
Componentization via Service

サービスによるコンポーネント化

Organized by Business
Capability

ビジネス機能に基づいたチーム編成

Microservices : マイクロサービスの開発体制とサービスの関係



Microservices : 9つの特徴

Componentization via Service
サービスによるコンポーネント化

Products Not Projects
プロジェクトではなく製品として捉え開発運用する

Organized by Business
Capability
ビジネス機能に基づいたチーム編成

Microservices : プロジェクトではなく製品として捉え開発運用する

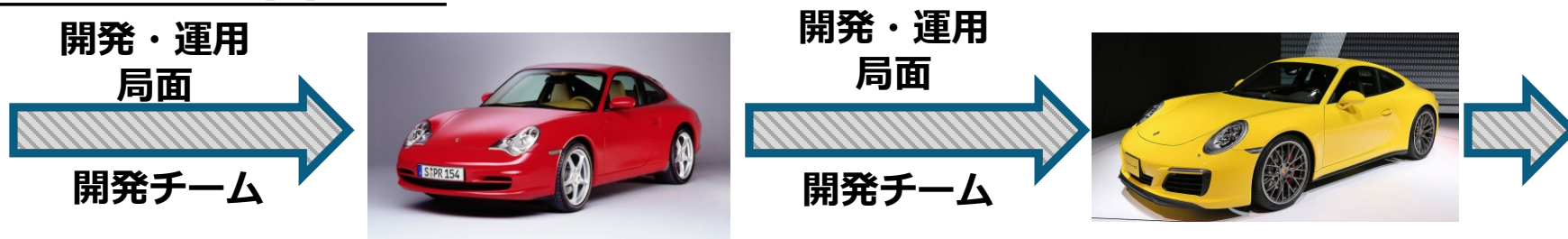
フィードバックを受けながら, One Teamが継続的に**“開発・運用”**する

プロジェクトの例：ダム



Microservicesの開発・運用スタイル

製品の例：自動車



Microservices : 9つの特徴

Componentization via Service

サービスによるコンポーネント化

Products Not Projects

プロジェクトではなく製品として捉え開発運用する

Decentralized Governance

非中央集権的な言語やツールの選択

Infrastructure Automation

基盤の自動化

Organized by Business Capability

ビジネス機能に基づいたチーム編成

Smart endpoints and dumb pipes

インテリジェントなエンドポイントとシンプルなパイプ

Decentralized Data Management

非中央集権的なデータ管理

Design Failure

障害, エラーを前提とした設計

Evolutionary Design

先進的な設計

Microservicesの懸念点

□ サービスの粒度

- チームの粒度でもある

- 2 Pizza Team

- ドメイン・スペシャリストであっても一回の分析・設計でサービス粒度を確定することは難しい

□ 分散システムであるが故の複雑さ

- 設計/開発/テスト

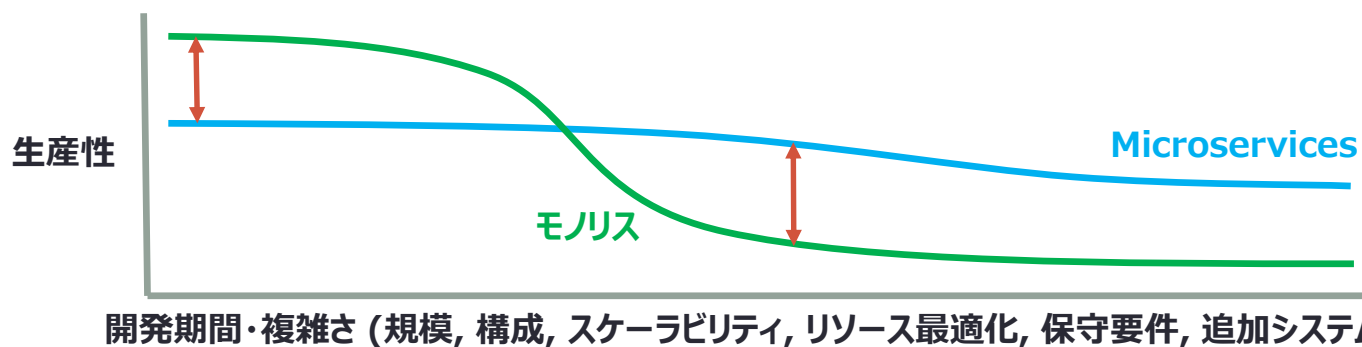
- データ整合性

- 運用オペレーション

□ パフォーマンス

□ One Off (一回だけの作業) では投資対効果が得られない

- MicroservicePremium



Microservicesの懸念点

□技術の多様性

- 各サービス毎に、プログラミング言語、ツール、データストアの選択を許容
- 課題
 - チーム間でのエンジニアの異動が困難になる可能性あり
 - 採用技術が一般的では無い場合、エンジニアの新規雇用が困難になる可能性あり
- 現実事例での対応例
 - Netflixでは採用技術の標準化をすすめている
 - プログラミング言語 : Java
 - データストア : Cassandra
 - 適切な理由があれば、例外を許容する

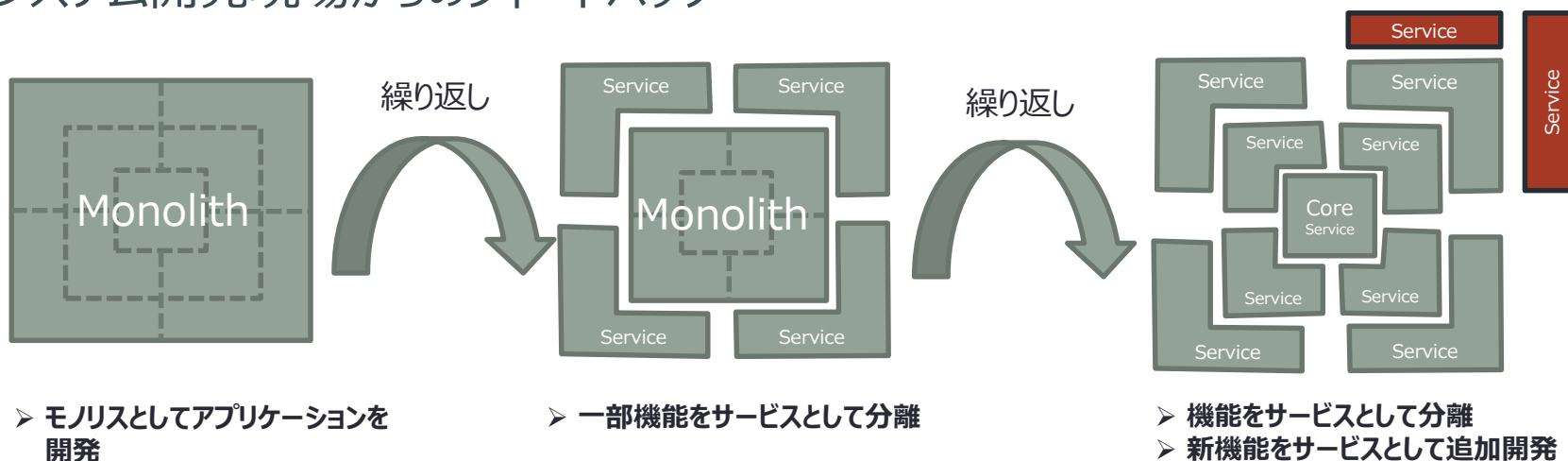
モノリス・ファースト

□モノリス・スタイルのアプリケーションを、段階的にマイクロサービス・スタイルに置き換えるプラクティス

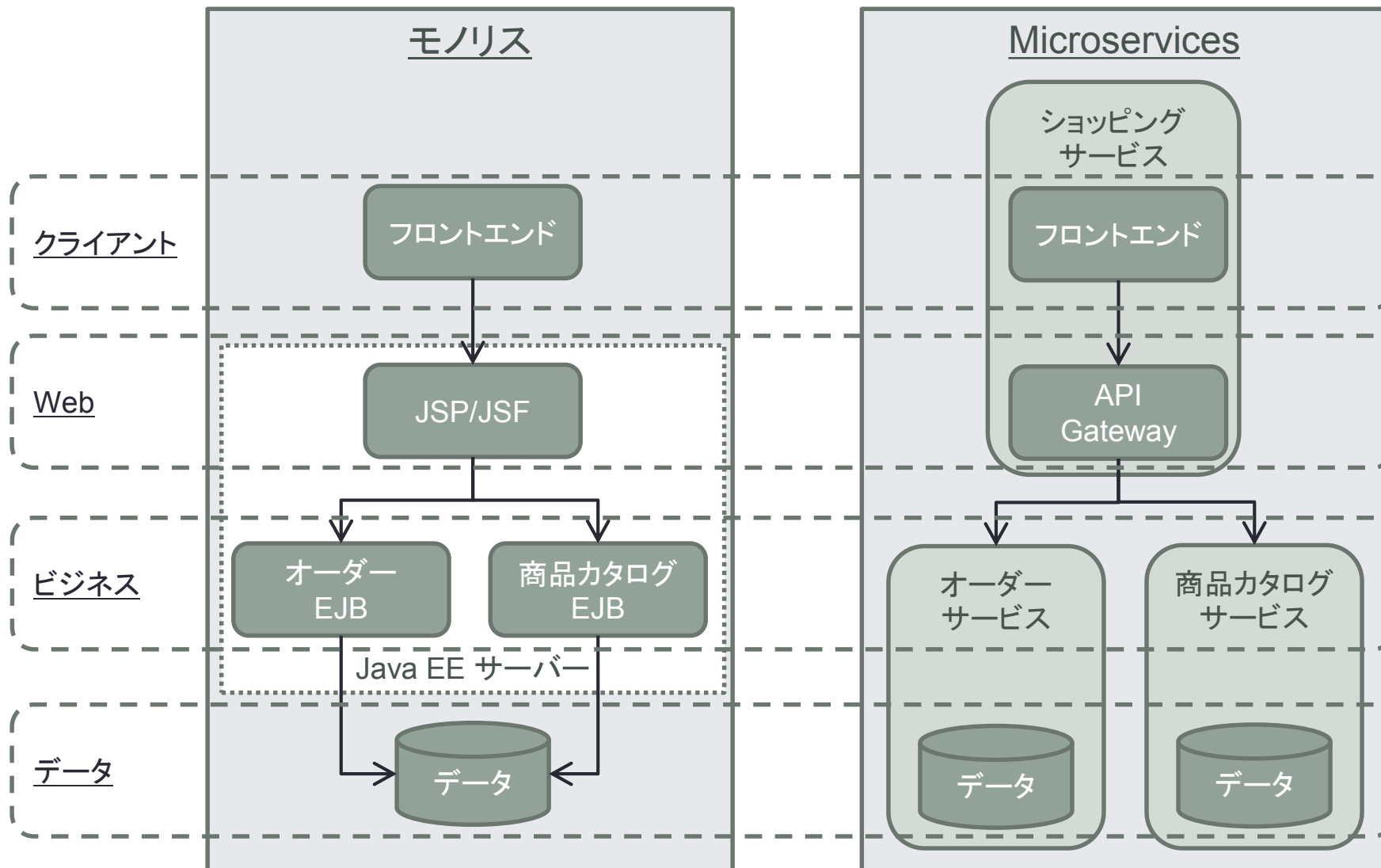
- 初回のアプリケーション開発時：従来通りモノリス・スタイルで開発
- 繰り返し開発/保守時：マイクロサービス・スタイルで開発
- <http://martinfowler.com/bliki/MonolithFirst.html>

□動機

- マイクロサービスの懸念点の解決
- システム開発現場からのフィードバック



Microservicesとモノリス ~Java Webアプリケーションにおける構造の比較~



註:

Java EE : Javaによるサーバー・サイド・アプリケーション開発のための技術仕様。

JSP/JSF : それぞれJavaによるWebページのテンプレート・ライブラリー、フレームワーク。Java Server Pages, Java Server Facesの略。

EJB : Javaによる、分散型コンポーネント・モデル。Enterprise JavaBeansの略。

MicroservicesとSOA

□ トップ・ダウン or ボトム・アップの違いがある

○ SOAは、トップ・ダウン・アプローチのアーキテクチャー

- ITベンダー、アーキテクトによる概念先行型の“アーキテクチャー”
- W3C策定のWebサービス仕様に深く依存

○ Microservicesは、ボトム・アップ・アプローチのアーキテクチャー・スタイル

- 実際のITシステム開発プロジェクトからのフィードバックを基にした“アーキテクチャー・スタイル”
- Technology-Agnostic
 - 製品・技術への依存性は最低限

□ Microservicesは、SOAを実現するための1つのアプローチ

○ 共に、“サービス”主体でITシステムを構成する考え方

○ Microservicesは、アジャイル・ソフトウェア開発、リーン・スタートアップ、モダンなクラウド・テクノロジーを積極的に活用したサービス指向開発スタイルの派生形

[参考] Service Oriented Architecture (SOA)

□ ITシステムを構築するためのアーキテクチャーの一つ

- 独立し、相互接続可能な“サービス”単位に、アプリケーションを設計・実装
- 各サービスはオープンな手段で相互接続し、業務処理を遂行
- **本来は、実装技術に依存しない上流レベルのコンセプト**

□ SOAの実際

- 実装技術としてWebサービスに深く依存
 - 通信プロトコル：SOAP/HTTP
 - インターフェース定義：WSDL
 - その他、“WS-”で始まる多くの技術
- 数々の課題より日本では普及したとはいえない
 - SOAの本格的適用にあたっては、ユーザー企業・法人の組織体制見直しが求められる
 - 技術的課題：サービスの抽出手法, パフォーマンス, Webサービスのインターオペラビリティ, 他

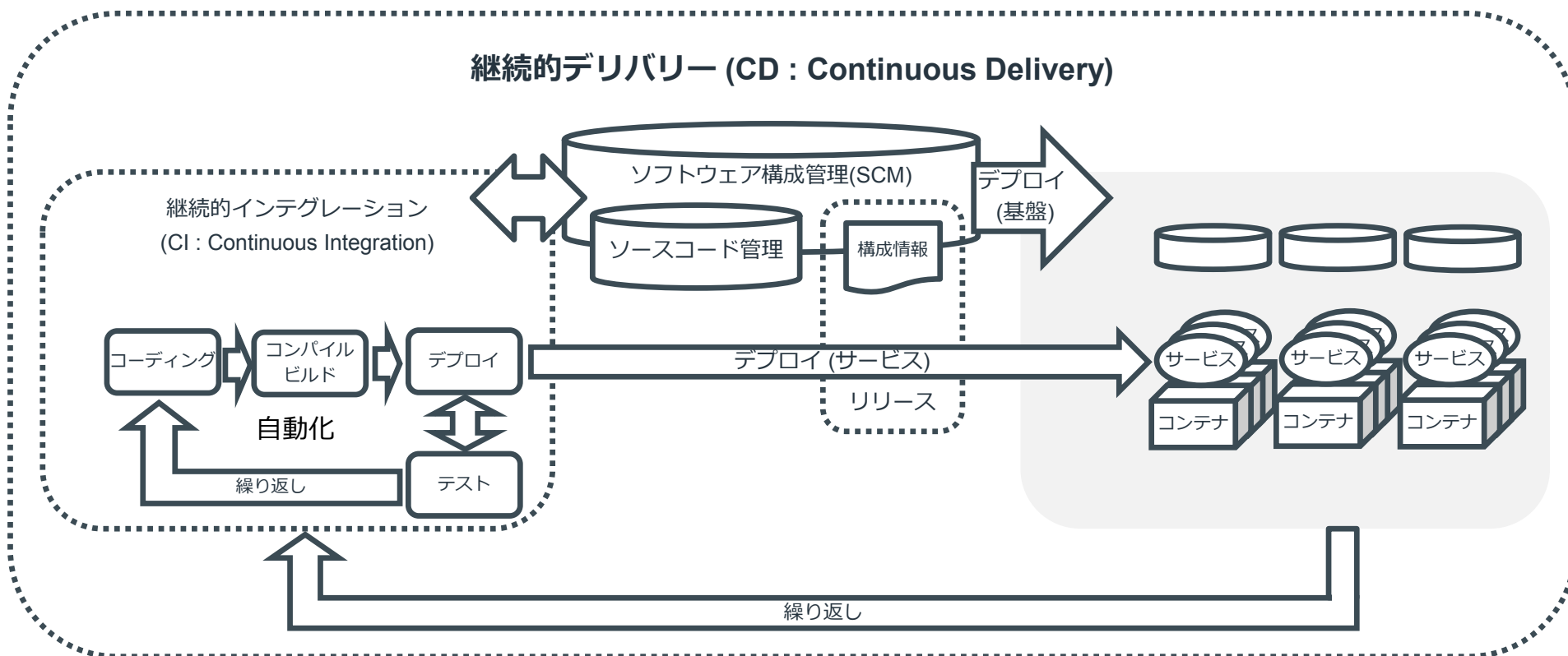
□ 主な標準

- ISO/IEC 17998 : Information technology -- SOA Governance Framework
- ISO/IEC 18384-1 : Information technology -- Reference Architecture for Service Oriented Architecture (SOA RA) -- Part 1: Terminology and concepts for SOA
- ISO/IEC 18384-2 : Information technology -- Reference Architecture for Service Oriented Architecture (SOA RA) -- Part 2: Reference Architecture for SOA Solutions
- ISO/IEC 18384-3 : Information technology -- Reference Architecture for Service Oriented Architecture (SOA RA) -- Part 3: Service Oriented Architecture ontology

クラウド・ネイティブな アプリケーション・ライフサイクル管理 ～継続的デリバリーとOpen Toolchains～

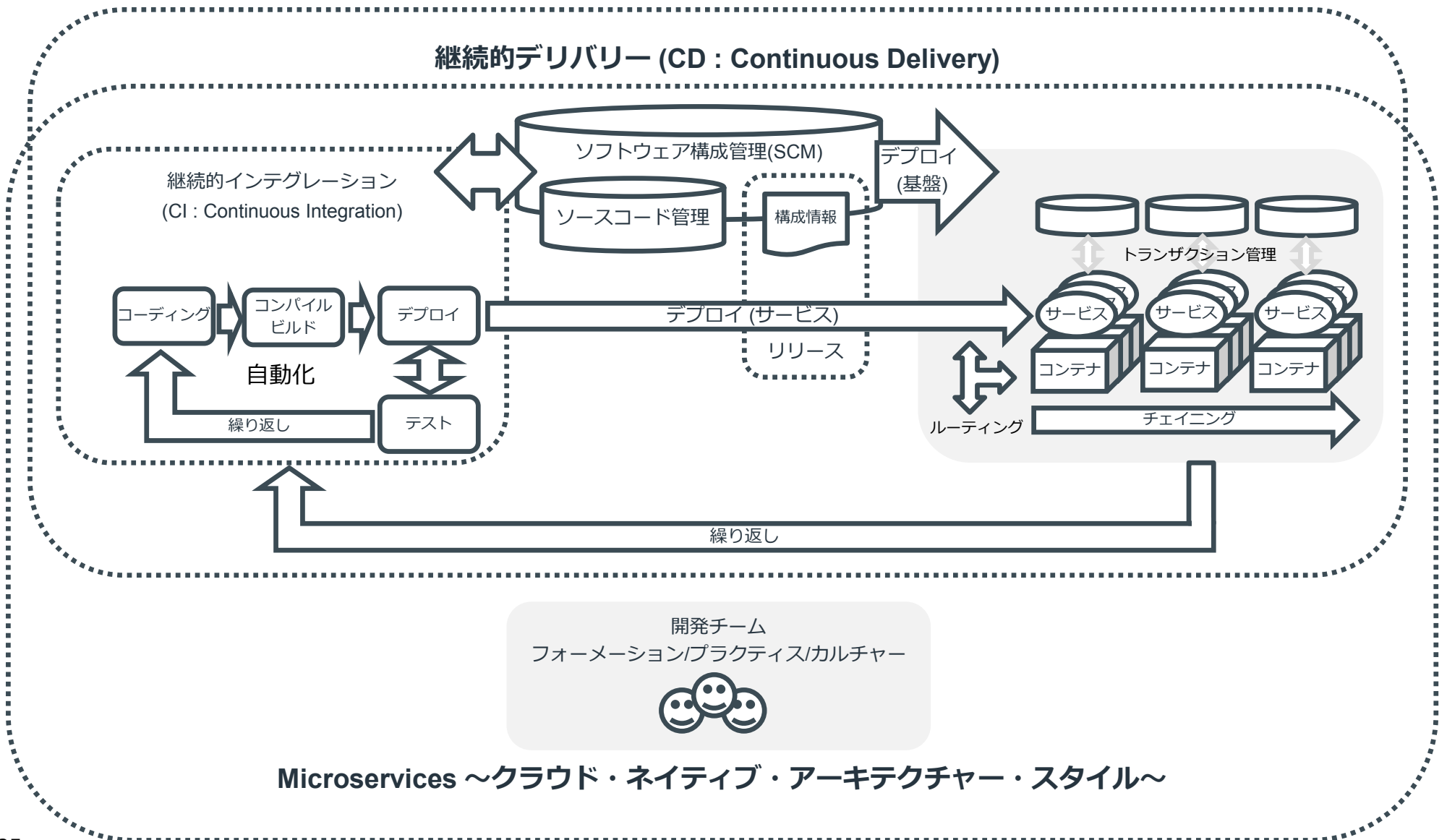


継続的デリバリー

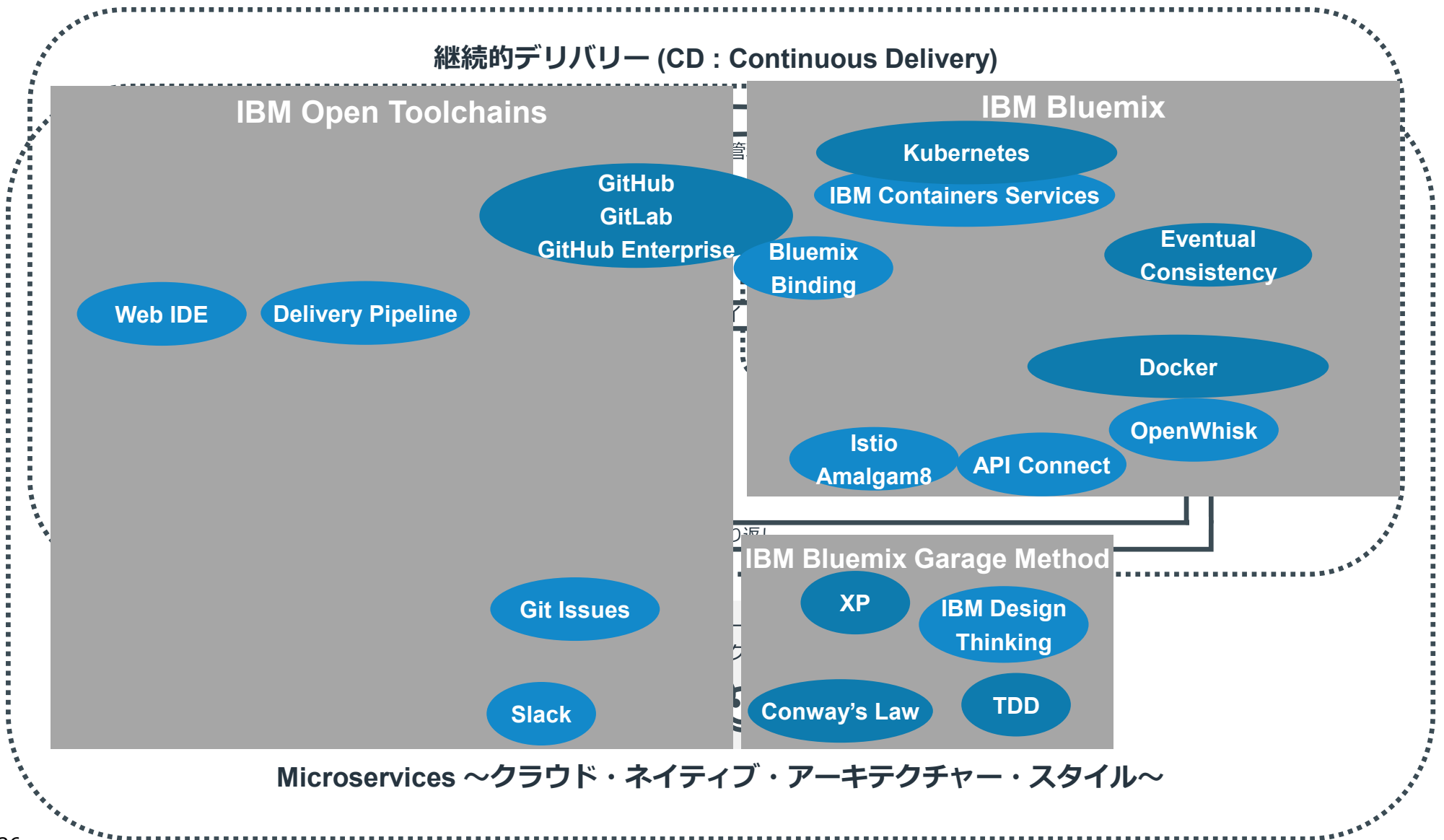


- アプリケーションの開発プロセスとデプロイ、基盤のデプロイを自動化
- システム・リリースのスピード・アップ、コスト抑制、品質向上を実現

クラウド・ネイティブ・アプリケーション・ライフサイクル全体像



クラウド・ネイティブ・アプリケーション・ライフサイクル全体像



Bluemix DevOps Servicesとは

□クラウド・ネイティブ・システムを開発するために必要なフル・カバレッジのIBMソリューション

□構成要素

○Bluemix Continuous Delivery : Bluemix上でSaaSとして提供される Collaborative Lifecycle Management (CLM) ソリューション

■Open Toolchains : カスタマイズ可能なオープンなToolchain

■Open Toolchainsを構成する主要なツール :

- WebIDE (Eclipse Orion based) : 統合開発環境
- IBM Git Repos and Issue Tracking : SCM & 課題管理
- Deliver Pipeline : ビルド/テスト/デプロイの自動化
- DevOps Insights : 品質管理

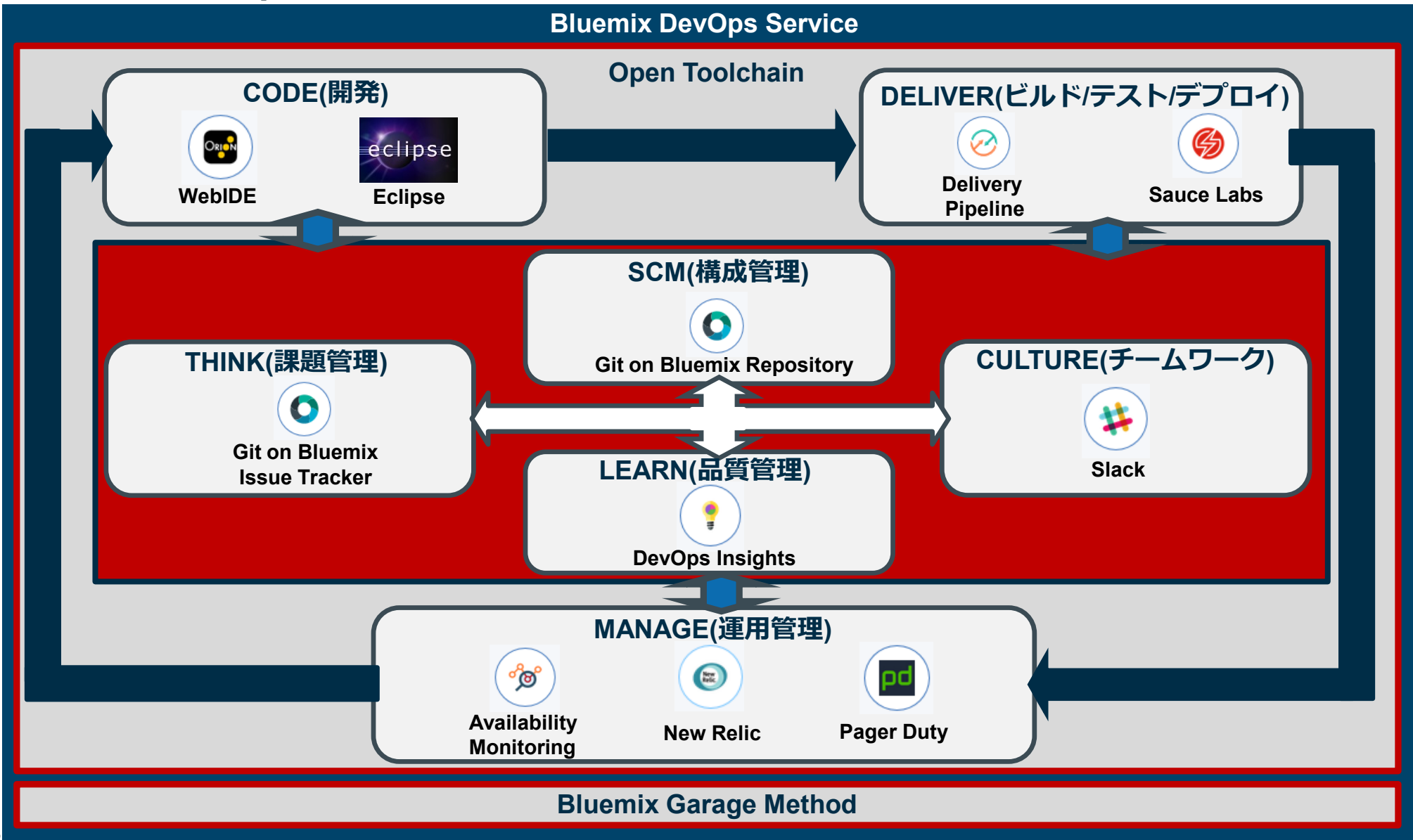
■他多数Open Toolchainsへの組み込みが可能

- GitHub, Slack, PagerDuty, Sauce Labs, Jenkins, JIRA, ...

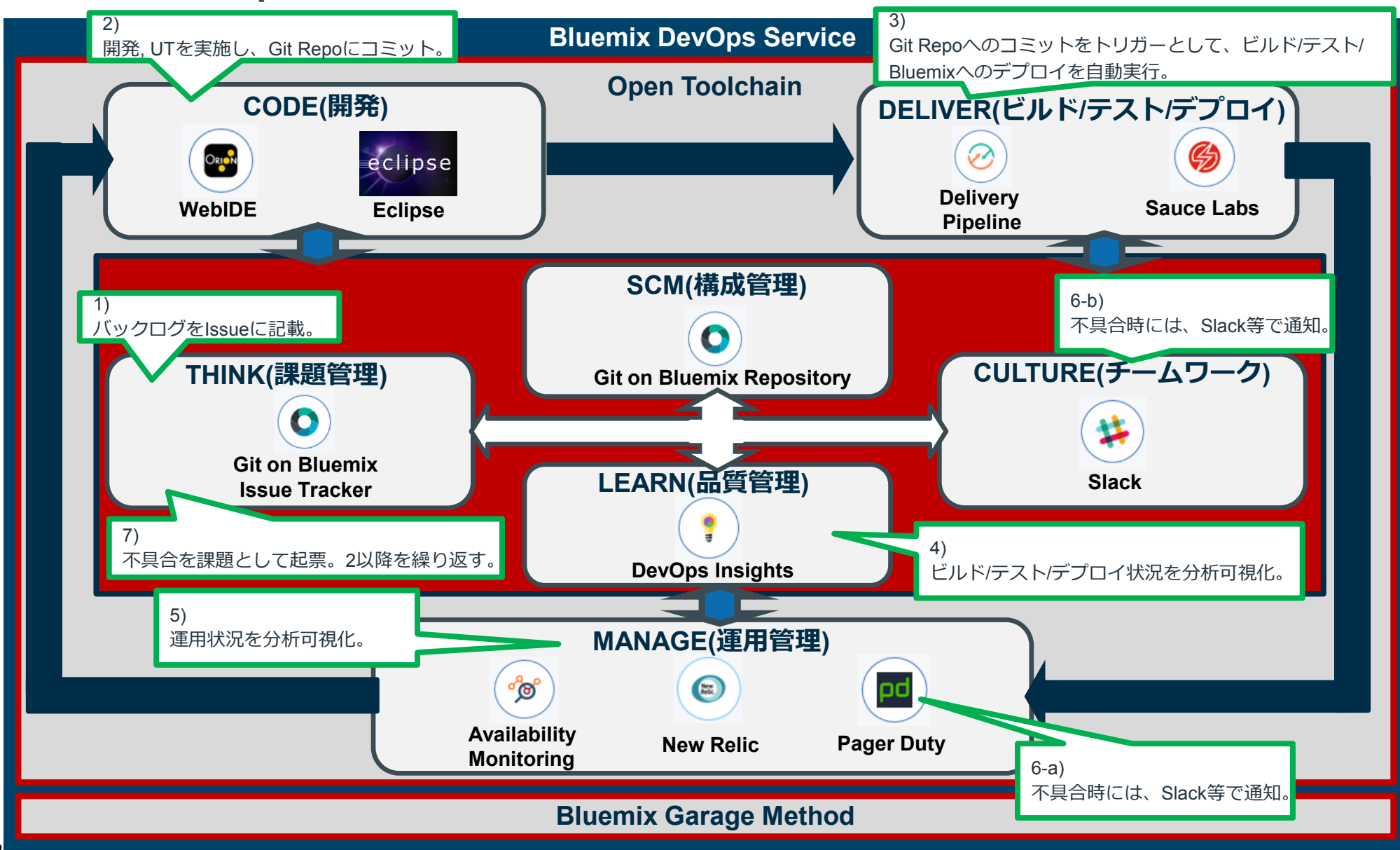
○Bluemix Garage Method :

コンサルテーション&デリバリー・サービス

Bluemix DevOps Servicesによるクラウド・ネイティブ・アプリケーション開発全体像



Bluemix DevOps Servicesによるクラウド・ネイティブ・アプリケーション開発シナリオ



Open Toolchains

オープンかつフルカバレッジのDevOpsソリューション

- 開発, デプロイ, 運用タスクを支えるツールを統合するプラットフォーム
 - 一連のタスクを支えるツール群を一括管理
 - ツール間の管理データ連携
- メリット～**開發生産性**～
 - DevOpsの一連の作業にオープンなツールを組み込み可能
 - デファクト・スタンダード, 使い慣れたツールを活用してDevOpsを実践
 - オープンなツールをシームレスに統合
 - 開発者の手作業による各ツール間の個別の連携を排除
- 参照
 - <https://www.ibm.com/devops/method/category/tools>

Open Toolchains : ダッシュボードに見るコンセプト

- 各ツールを統合したビューをダッシュボードとして提供
- Open Toolchainsが各ツール間にて、管理データを連携 (連携のレベルは各ツールに依存)

アプリケーション開発
に利用するツール群

運用管理に利用する
ツール

新たなツールの追加

アイデアや課題等を
管理するツール

THINK



Issues
ui-microservices-tool...

✓ 構成済み

CODE



Git
ui-microservices-tool...

✓ 構成済み

DELIVER



Delivery Pipeline
ui-microservices-tool...

✓ 構成済み

MANAGE



PagerDuty
microservices-toolchai...

✓ 構成済み

LEARN



DevOps Insights

✓ 構成済み

CULTURE



Slack
#general

✓ 構成済み

ツールの追加

コミュニケーション等
チームのカルチャー
形成に利用するツール

状況を可視化する
ツール

デリバリーに利用する
ツール群

Git

クリック

Web IDE

クリック

クリック

Delivery Pipeline

Open Toolchains : 始め方 1

□アプリケーションからToolchainを作成する

○Bluemixコンソール上 > アプリケーションの概要

- 継続的デリバリー・カードの“有効化”ボタンをクリック

○ツールチェーン画面

- 組織名とツールチェーン名を入力
- 必要に応じてツールを構成
- “作成”ボタンをクリック



継続的デリバリー

このアプリ用の継続的デリバリーは有効になっていません。
継続的デリバリーを有効にすると、Delivery Pipeline、GitHub などを使用したビルド、テスト、およびデプロイメントが自動化されます。

有効化

ツールチェーン画面

思考 コード 配信 実行

Issue トラッカー リポジトリ PIPELINE BLUEMIX

WEB IDE

組織名: hirot@jp.ibm.com / ツールチェーン名: hirot20170701

ツール統合

Git Repos and Issue Tracking Eclipse Orion Web IDE Delivery Pipeline

GitLab Community Edition 上に構築されて IBMi によってホストされる Git リポジトリおよび Issue のトラッキング

作成

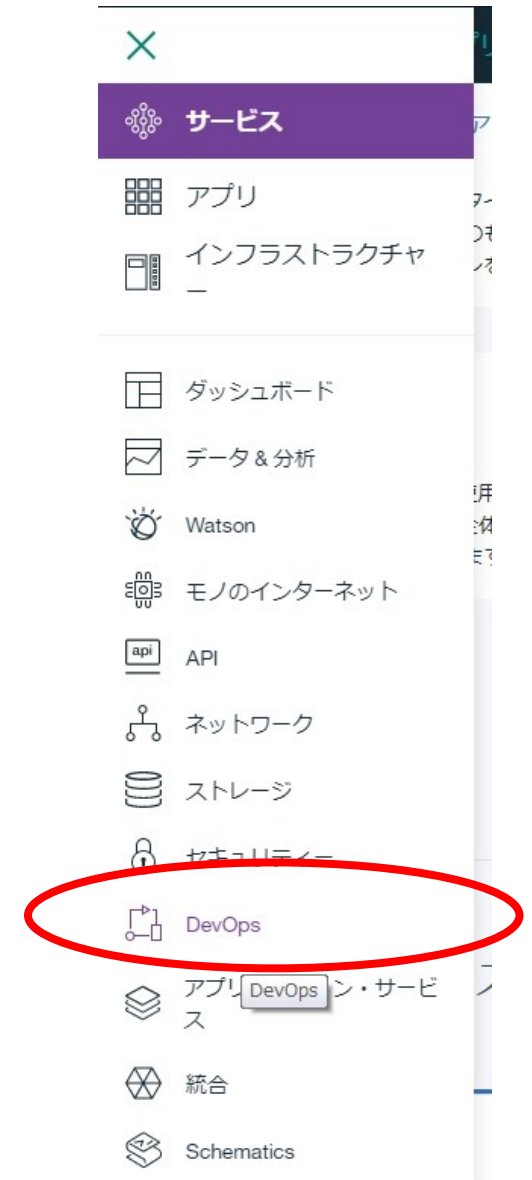
Open Toolchains : 始め方 2 (1/2)

□テンプレートからToolchainを作成する (1/2)

○ Bluemixコンソール上 > ハンバーガー
> DevOps

○ ツールチェーン画面

■ "ツールチェーンの作成" ボタン
をクリック



Open Toolchains : 始め方 2 (2/2)

□テンプレートからToolchainを作成する (2/2)

○テンプレート画面

- 適切なテンプレートをクリックする

○ツールチェーンの作成画面

- 組織名とツールチェーン名を入力
- 必要に応じてツールを構成
- “作成”ボタンをクリック



Open Toolchains : サポート対象ツール 1/3

局面	ソリューション	概要	提供元	種別
SCM 構成管理	Git on Bluemix (Repository)	IBMがホスティングするGitリポジトリでソース・コードを管理します。	IBM	Open Toolchain
	GitHub Enterprise	Bluemix Dedicated上にホスティングされるGitリポジトリでソース・コードを管理します。	IBM	サービス
	GitHub	GitHub.comがホスティングするGitリポジトリでソース・コードを管理します。	3rd Party	Open Toolchain
THINK 課題管理	Git on Bluemix (Issue Tracker)	プロジェクトの課題を管理します。	IBM	Open Toolchain
	JIRA	プロジェクトの課題を管理します。	3rd Party	Open Toolchain
	Rational Team Concert	ビルドやデプロイメント等の自動化に加え、プロジェクト計画、変更管理、課題管理、ソースコード管理、レポート等、チーム開発に必要な統合開発機能を提供します。	IBM	Open Toolchain
LEARN 品質管理	DevOps Insights ベータ	コードの品質、チーム開発状況、デプロイメント状況等プロジェクト全般の状況や品質を可視化します。	IBM	サービス
CULTURE チームワーク	Slack	プロジェクト・チーム・メンバー間のコミュニケーション機能を提供します。	3rd Party	Open Toolchain
CODE 開発	WebIDE	SaaS形式の統合開発環境です。ブラウザーをインターフェースとしてプログラミング、テストを行えます。	IBM	Open Toolchain
	IBM Eclipse Tools for Bluemix	Bluemixとの連携機能を付加するEclipseプラグインです。EclipseによるBluemixアプリケーション開発を効率化します。	IBM	Eclipse Plugin

Open Toolchains : サポート対象ツール 2/3

局面	ソリューション	概要	提供元	種別
DELIVER				
ビルド デプロイ 自動化	Delivery Pipeline	ビルドやデプロイメント等を自動化します。	IBM	Open Toolchain
	Jenkins	ビルドやデプロイメント等を自動化します。	3 rd Party	Open Toolchain
デプロイ	UrbanCode Deploy	ビルドやデプロイメント等を自動化します。	IBM	Open Toolchain
ビルド 管理	Artifactory	リポジトリやビルドを管理します。	3 rd Party	Open Toolchain
	Nexus	リポジトリやビルドを管理します。	3 rd Party	Open Toolchain
テスト	Hiptest	リアルタイム・テスト管理環境を提供します。	3 rd Party	サービス
	Sauce Labs	プロジェクトの継続的統合テストを自動化します。	3 rd Party	Open Toolchain
	BlazeMeter	パフォーマンス・テスト機能を提供します。	3 rd Party	サービス
	Load Impact	パフォーマンス・テスト機能を提供します。	3 rd Party	サービス
	SonarQube	ソース・コードの品質をチェックします。	3 rd Party	Open Toolchain

Open Toolchains : サポート対象ツール 3/3

局面	ソリューション	概要	提供元	種別
MANAGE				
監視	Availability Monitoring	Bluemix上のアプリケーションの状況を監視し、Bluemixコンソールにて可視化することに加え、Eメール、Slack、SMS、プッシュ通知にてアラートを送信します。	IBM	サービス
	Monitoring and Analytics	アプリケーションのパフォーマンスを監視し分析するApplication Performance Monitoring (APM) です。	IBM	サービス
	New Relic	アプリケーションのパフォーマンス監視、ログ分析、イベント監視機能を提供します。	3rd Party	サービス
	jKool	アプリケーション間でやりとりされるデータの分析と可視化を行うSaaSです。	3rd Party	サービス
通知	Alert Notification 試験中	DevOpsツール処理中の問題に関する通知を、Eメール、SMS、ボイスで送信します。	IBM	Open Toolchain
	Cloud Event Management 試験中	イベントや障害などを統合管理し、速やかな問題の分析や解決を支援します。	IBM	サービス
	PagerDuty	障害などの重要イベントをEメールやSlack等で通知します。	3rd Party	Open Toolchain

Open Toolchains : 情報源

❑ IBM Cloud Garage Method DevOps toolchainsページ

○ <https://www.ibm.com/devops/method/category/tools>

❑ Bluemix 資料

○ https://console.bluemix.net/docs/services/ContinuousDelivery/toolchains_working.html#toolchains_getting_started

❑ GitHub

○ <https://github.com/open-toolchain>

❑ Wiki (GitHub)

○ <https://github.com/open-toolchain/sdk/wiki>

IBM Bluemix Continuous Delivery : Web IDE

□ ブラウザーをインターフェースとする統合開発環境

○ Eclipse Foundation “Orion Project”をベースに実装

- Webブラウザー上で稼動するオープンなツールの統合基盤
- “Developing for the web, in the web”



□ 機能

○ コード・アシスト, コード・コンプリーション, エラー・チェック

- JavaScript, HTML, CSS

○ シンタックス・ハイライティング

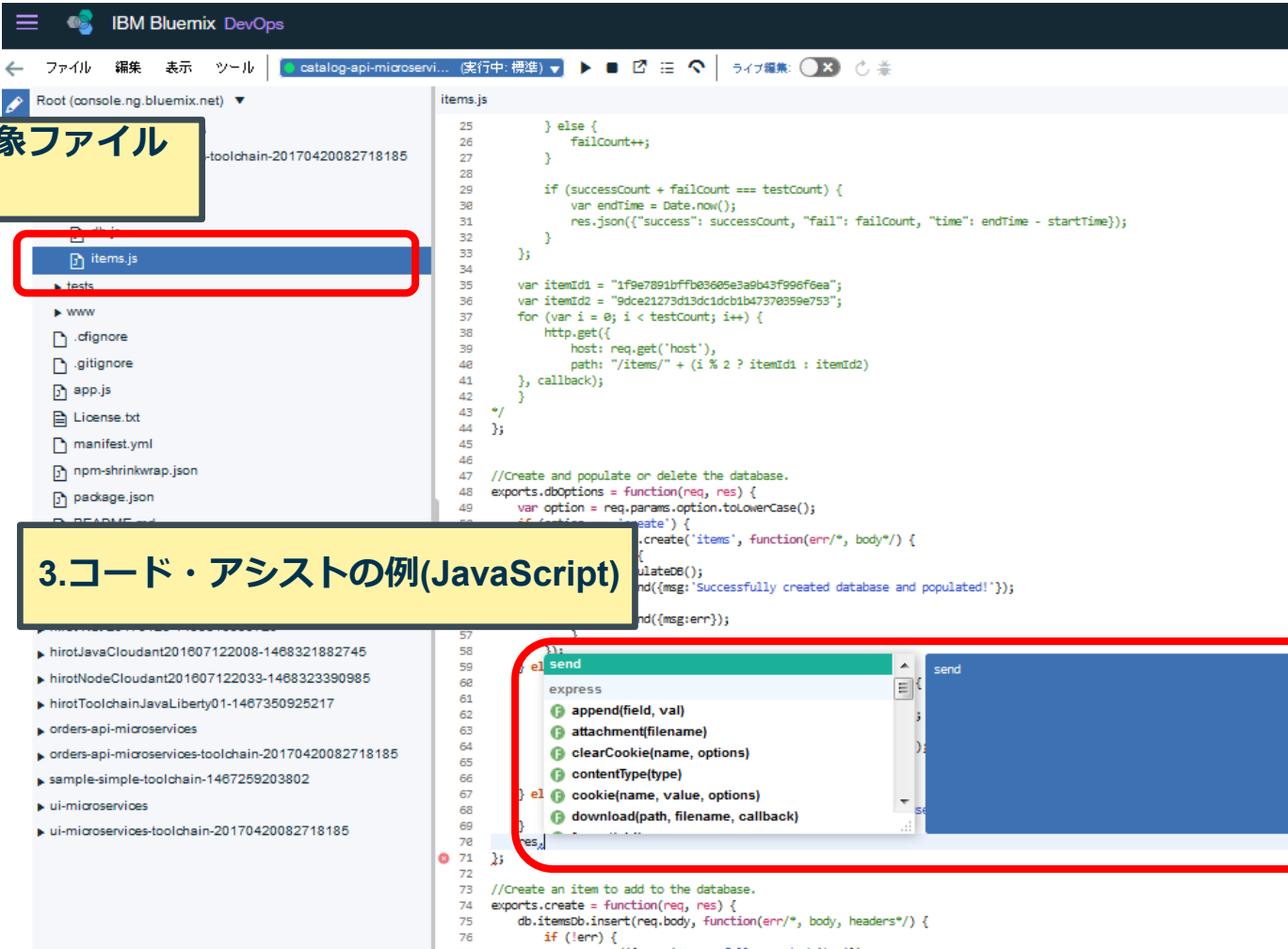
- C, C++, Go, Java, JSON, PHP, Python, Ruby, Swift, VB, XML, 他

○ デバッグ (Bluemix Live Sync)

○ SCMとのシームレスな統合

IBM Bluemix Continuous Delivery : Web IDE

□編集画面



The screenshot shows the IBM Bluemix DevOps Web IDE interface. The top bar displays the IBM Bluemix DevOps logo and navigation tabs for 'ファイル' (Files), '編集' (Edit), '表示' (View), and 'ツール' (Tools). The main area is divided into three sections:

- 1. 編集対象ファイルの選択** (Select the file to be edited): A file explorer on the left shows a list of files and folders. The file 'items.js' is highlighted with a red box.
- 2. 編集対象ファイルの表示・編集領域** (Display and edit the file to be edited): The code editor on the right displays the contents of 'items.js'. The code is written in JavaScript and includes database operations and API endpoints. A red box highlights the code editor area.
- 3. コード・アシストの例(JavaScript)** (Example of code assistance (JavaScript)): A dropdown menu is shown below the code editor, displaying a list of JavaScript methods such as 'append(field, val)', 'attachment(filename)', 'clearCookie(name, options)', 'contentType(type)', 'cookie(name, value, options)', and 'download(path, filename, callback)'. A red box highlights this dropdown menu.

IBM Bluemix Continuous Delivery : Delivery Pipeline

□ 継続的デリバリー (Continuous Delivery:CD)を実現するフレームワーク

- ソフトウェアのビルド、デプロイ、テストの自動化を実現

□ 機能

- CDを構成する3つのステージを提供

■ビルド

- コンパイルが必要な場合にはコンパイルを実施
- プログラム・ランタイムに適したパッケージを作成

■デプロイ

- ビルド成果物を指定したクラウド・プラットフォームにデプロイ

■テスト

- プログラムのテストを実施

- 各ステージを必要な数だけ作成可能

- 各ステージの順番を制御可能

- 各ステージの実施条件を指定可能

- ステージ構成例)

■ “ビルド” → “テスト環境へのデプロイ & テスト” → “本番環境へのデプロイ”

- SCMにソース・コードをコミットしたことをトリガーに一連のステージ処理を開始
- 前ステージ処理が成功した場合のみ、次ステージに遷移
- ステージ失敗時には、ログを出力して、処理を中断

IBM Bluemix Continuous Delivery : Delivery Pipeline

□ ステージの構成例



IBM Cloud DevOps Insights

- クラウド環境におけるアプリケーション開発の品質管理ソリューション
- 4つのサブシステムを提供
 - Deployment Risk : ビルド品質の可視化、次工程遷移のポリシー管理
 - Developer Insights : 開発アクティビティ/インシデント履歴統計の可視化と予測される方向性の定時
 - Team Dynamics : ソーシャル・コーディングの分析と可視化
 - Delivery Insights : 過去のデリバリー履歴の分析とレポート

IBM Cloud DevOps Insights : Deployment Risk

ビルド内各アクティビティの達成状況

IBM Bluemix DevOps

microservices-toolchain-20170420082718185

Getting Started

Deployment Risk

Developer Insights

Team Dynamics

Delivery Insights

Settings

ベータ

実動ゲート・ダッシュボード

ステージングから実動用にプロモートする準備ができています

アプリケーション	ビルドはステージング中	実動ゲート・ポリシー	最新の評価	ポリシー名	ビルドは作成中
catalog- api-microservices- toolchain- 201704200827181...	master:6	✓	Apr 20, 11:40 PM	PRODUCTION D...	master:6
orders- api-microservices- toolchain- 201704200827181...	master:2	✓	Apr 20, 11:37 PM	PRODUCTION D...	master:2
ui-microservices- toolchain- 201704200827181...	master:1	✓	Apr 20, 11:21 PM	PRODUCTION D...	master:1

環境	コードカバレッジ	ユニットテスト	機能テスト	動的スキャン	静的スキャン
ステージング	✓	✓	✓	●	●
実動	✓	✓	✓	●	●

ビルド情報

アプリケーション名: catalog-api-microservices-toolchain-20170420082718185

ビルド ID: master:6

組み込み staging

最終コミット ID 24d05be

デプロイ済み: 6 days ago

ポリシー名 PRODUCTION Deployment Checks

決定 パス

ビルド内各テストのカバレッジ状況

テストの要約:

84%

コード・カバレッジ
ステートメント: 84.51
機能: 84.21

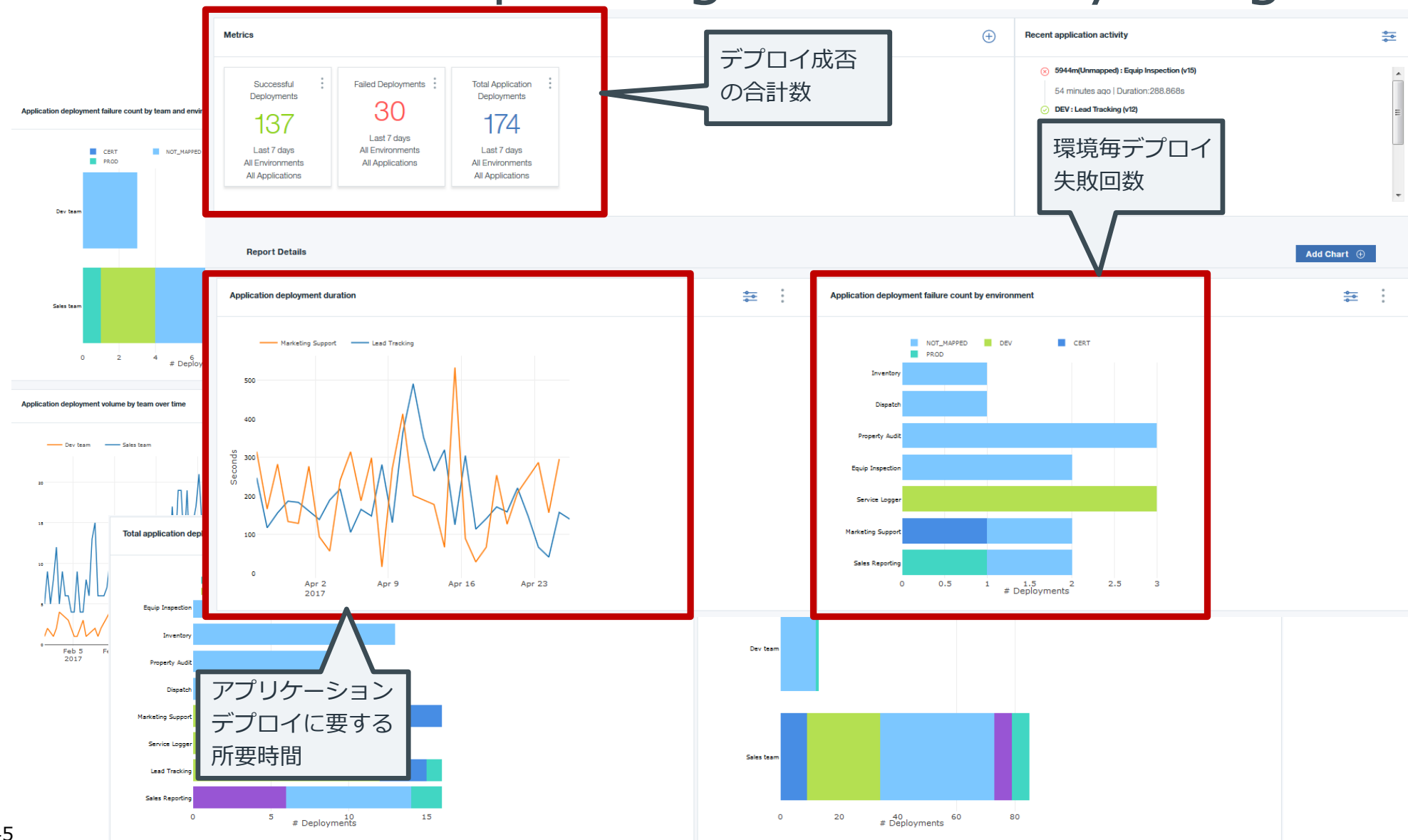
100%

単体テスト
合計: 16
パス: 16
失敗: 0

100%

機能テスト
合計: 1
パス: 1
失敗: 0

IBM Cloud DevOps Insights : Delivery Insights



Open Toolchainsの拡張

□ Open Toolchainsがサポートしないツールも統合可能

□ 統合のレベル

- ツールに依存

- データ連携のための構成は、ユーザーの責任で実装

 - 統合対象ツールがデータ連携のための仕組みを提供しない場合

 - エンドユーザーの手動運用にて対応する必要有

□ ユースケース

- Open Toolchainsがサポートしない

 - 使い親しんだツールを統合したい

 - 機能を提供するツールを統合したい

Open Toolchainsの拡張例:要件のトレーサビリティ機能の追加

□要件定義成果物のトレーサビリティ

○要件定義の成果物に関する対応状況をトラッキングすること

■要件定義成果物

- ユーザー・ストーリー、機能要件、システム要件等

■トラッキング例

- カバレッジ分析、影響分析、実装状況、テスト状況等

□IBM要件トレーサビリティ・ソリューション

○Rational DOORS Next Generation (DNG)

■要件定義、要求管理のためのソリューション

■Rational Team Concert(構成・変更管理)やRational Quality Manager (品質管理)と連携し、要件とレーサビリティを実現する (次ページ参照)

■オンプレミス製品、SaaSオフアリングの2形態として提供

- Bluemix上のサービスとしては提供していない

○Bluemix Open Toolchainsでは当該ソリューションを提供していない

■Git Issuesは課題管理(チケット管理)ソリューションであり、要求管理機能は提供しない

□Open Toolchainsに要件のトレーサビリティ機能を追加するには・・・**“Open ToolchainsにDNGを追加”**

[ご参考] IBM Rational Collaborative Lifecycle Management (CLM) ソリューション

役割、情報、成果物の結び付け、協調し合うことで ライフサイクルの 運営力を強化
要件～設計&実装～テストの情報を一元管理するリポジトリを持つ



Open Toolchainsの拡張例:要件のトレーサビリティ機能の追加

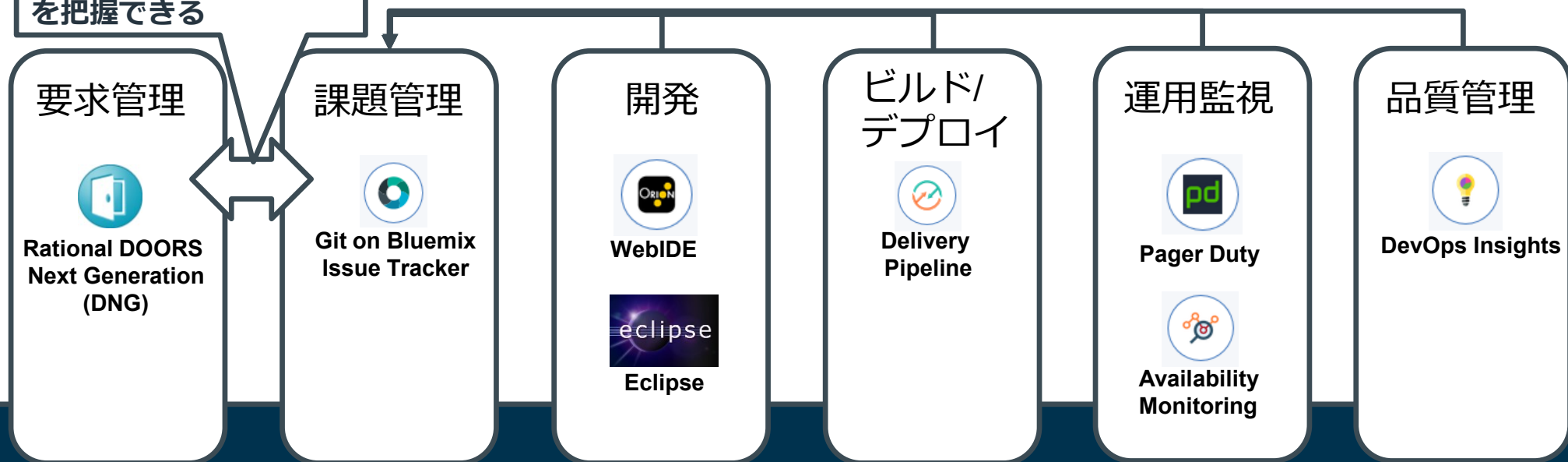
□ Open ToolchainsにDNGを統合すれば・・・

- 要求管理/課題(チケット)管理/開発/ビルド/デプロイ/運用監視の一連のプロセスをフルカバレッジで管理できる

要求と課題の関連付け

課題を通して要求の進捗を把握できる

進捗やインシデントは課題として管理される



Open Toolchains

Open Toolchainsの拡張例:要件のトレーサビリティ機能の追加

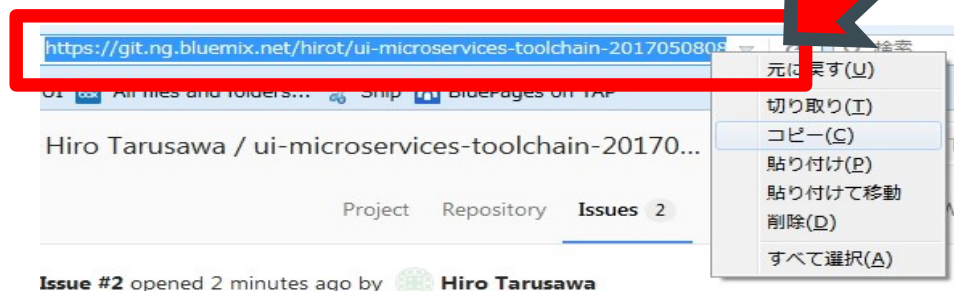
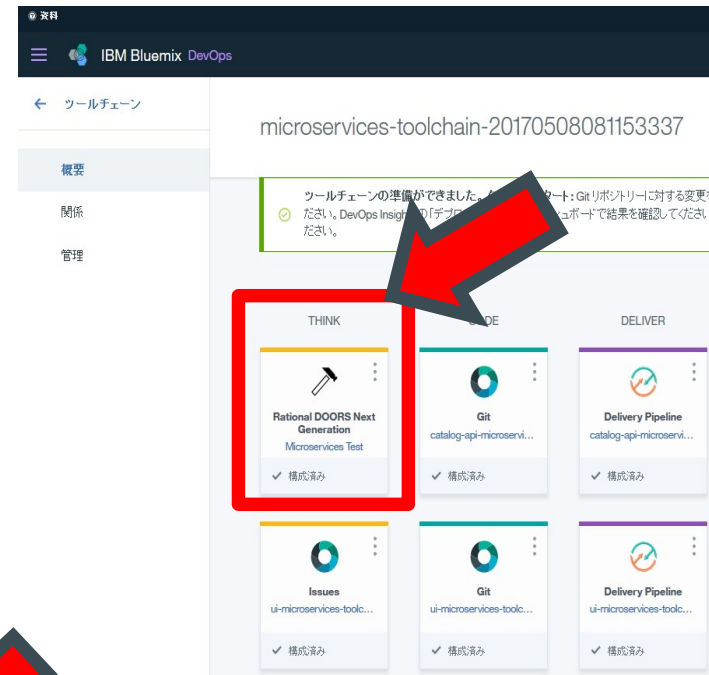
□注意事項：DNGはOpen Toolchainsのサポート対象ツールではない

○Open Toolchainsができること

- Open Toolchainsダッシュボード上に、DNGへのアイコンを追加

○Open Toolchainsができないこと

- Open Toolchains上の他ツールとDNGの自動データ連携
 - エンドユーザーの運用でカバー
 - または
 - 何らかの作りこみで対応



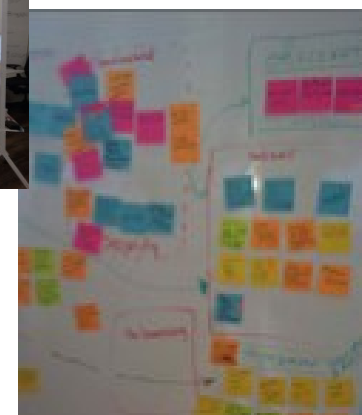
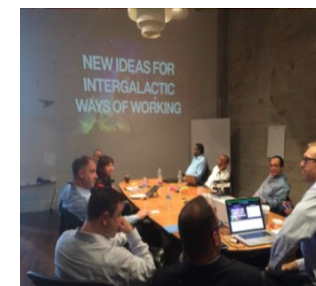
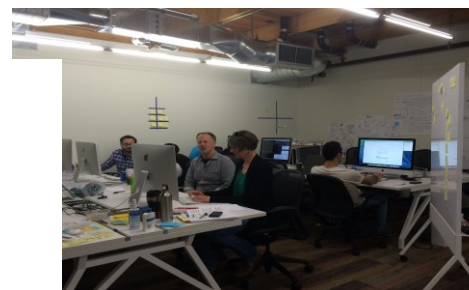
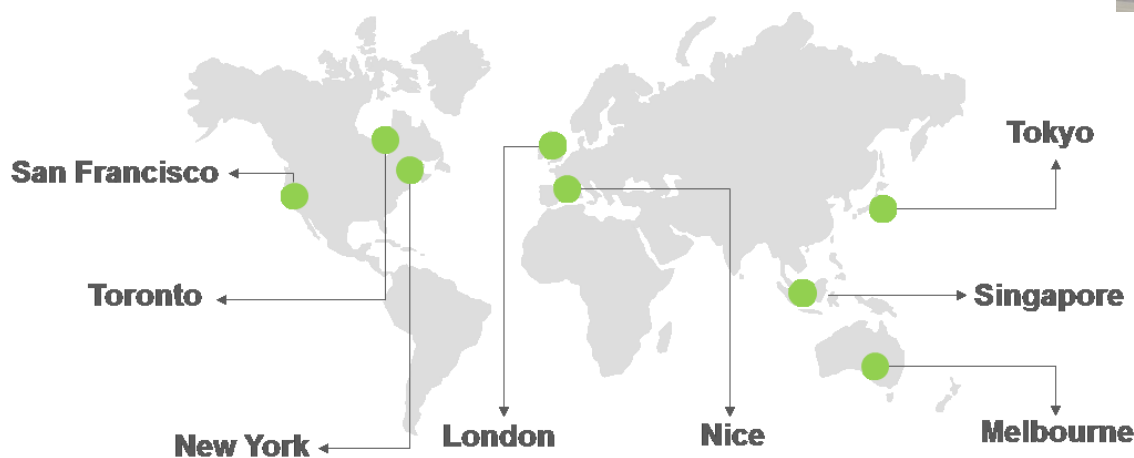
Bluemix Garageのご紹介



Bluemix Garage

Bluemix Garageは、カスタマー・ペインから導出したMVPを実装、検証した上で、お客様ビジネスのトランスフォーメーションを促進するコンサルティング・サービス

- **Disruptiveなトランスフォーメーション**を促進するコンサルティング・サービス
- MVP : Minimum Viable Product
 - 短期間で、開発と稼動が可能で、且つ、仮説を検証できる、最小の成果物
 - エンドユーザーに使ってもらえるという確証を得るまでは、開発への投資を最小限に抑える
- Bluemix Garage Method
 - リーン, IBM Design Thinking, eXtreme Programmingを活用
 - 素早いアイデア創出, 設計, 開発, デプロイ, 評価, 本番移行を促進



Bluemix Garage オファリング



イノベーティブな アプリケーション開発

- Garage への訪問
- Design Thinking
- Minimum Viable Product (MVP) ビルドアップ
- Garage トランスフォーメーション



ハイブリッド・クラウド アーキテクチャーへの移行

- アーキテクチャー
ワークショップ
- アプリケーション移行
アセスメント
- DevOpsパイプライン
- サービス・ビルドアウト
- アプリケーション移行パイロット

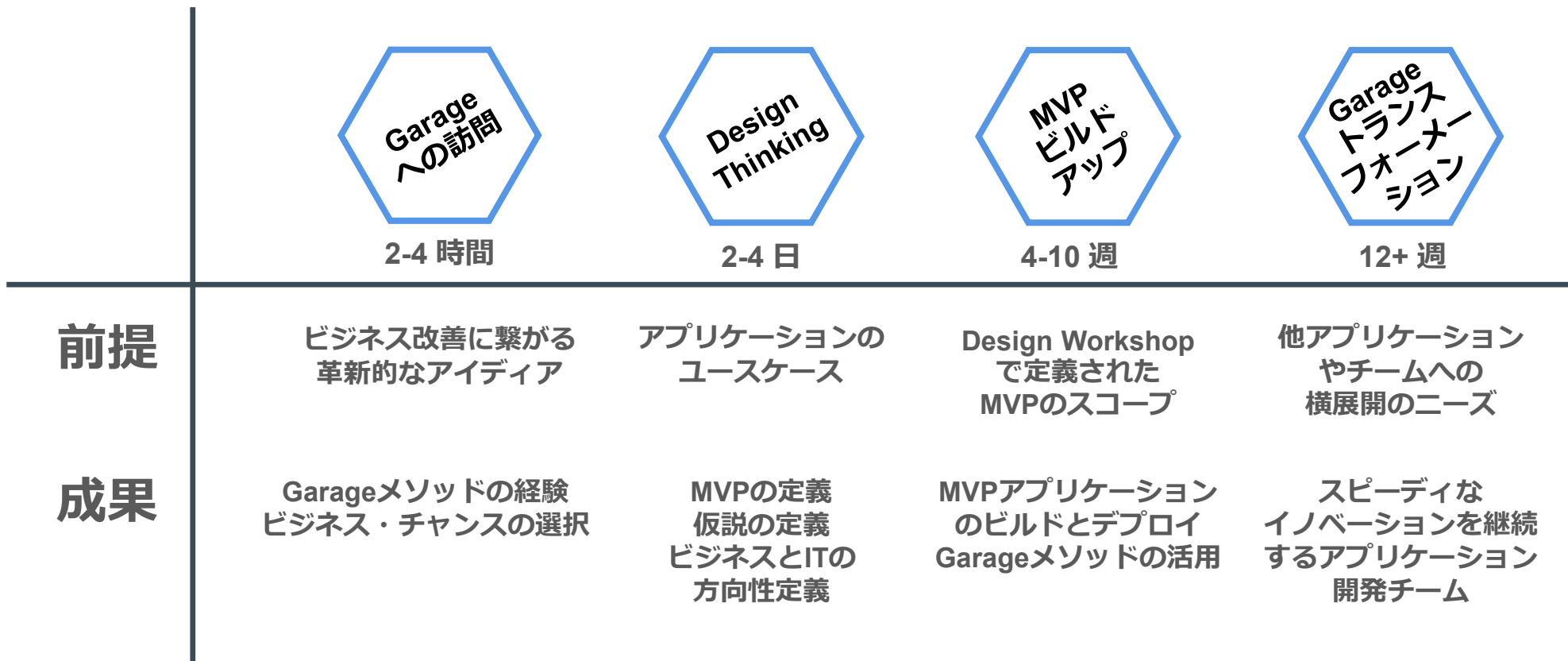


Garage アカデミー スキル・トランスファー

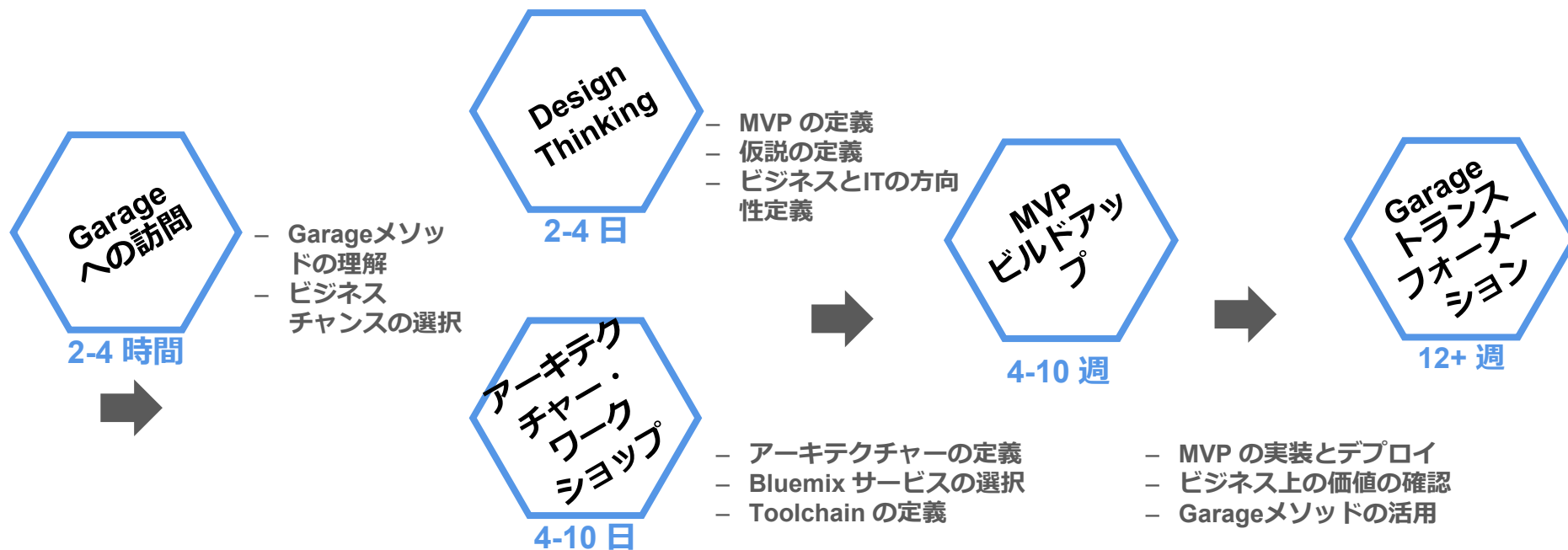
- Bluemix エンタープライズ
デベロッパー
- Bluemix ソリューション
アーキテクト・ブートキャンプ
- Bluemix Garage
デベロッパー・ブートキャンプ
- ガルバナイズ Bluemix
アカデミー

ニーズに応じて適切なオファリングを選択・組み合わせ, 実行する

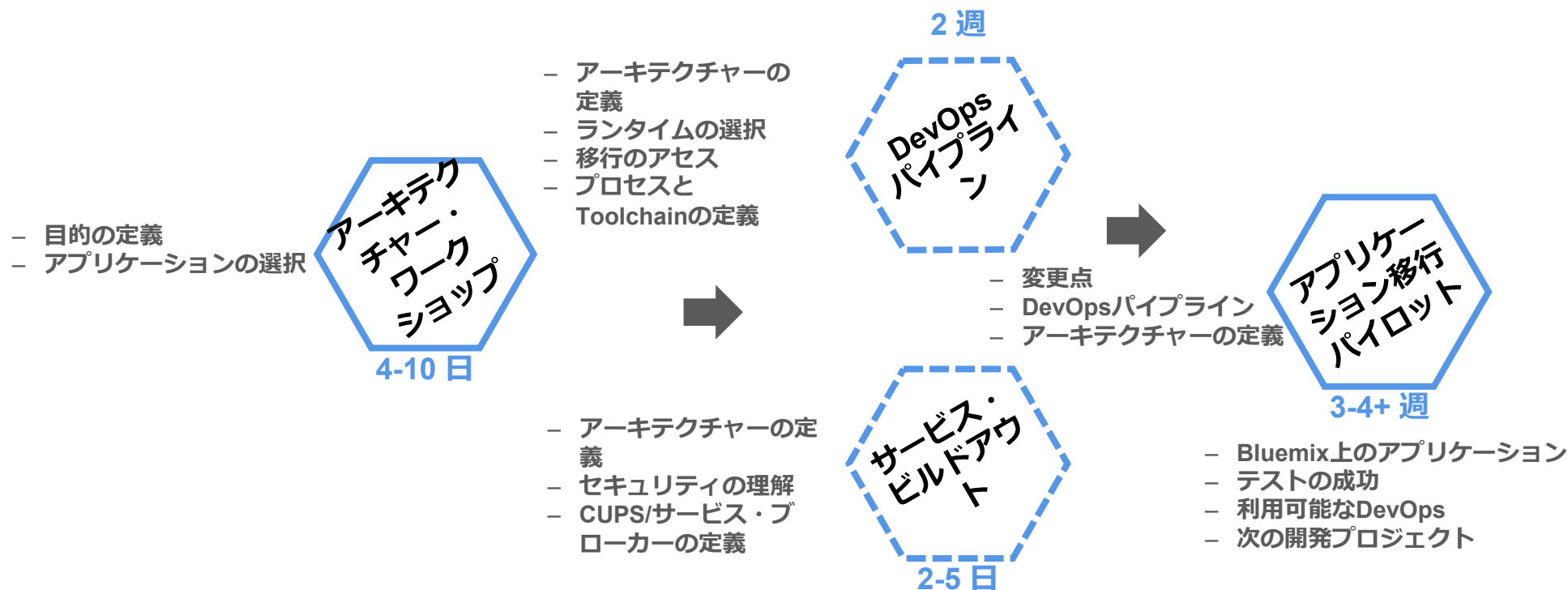
Bluemix Garage オフアリング例：基本的流れ



Bluemix Garage オフアリング例：新規アプリケーション開発



Bluemix Garage オフアリング例：既存アプリケーションの移行



*Cloud Foundry User Provided Services (CUPS)

Ready for Cloud
Ready for Business

