

# GeneXus を使用した DevOps

## 概要



DevOps は、アプリケーションの開発と、それに伴う作業、たとえば製品の移行、メンテナンス、ソフトウェアの進化を、終わりのないサイクルの中に統合する方法論です。この方法論が GeneXus にどう当てはまるのかを説明します。

## DevOps の定義



DevOps は、一連のアジャイル開発手法から構成されるもので、システム開発のライフサイクルを短縮しつつ継続的にソフトウェアを提供するために、ソフトウェア開発 (Dev) と情報テクノロジーの運用 (Ops) を組み合わせます。

「Dev」(開発) のサイクルには、アプリケーションの設計、開発、コーディング、ビルド、テストが含まれます。「Ops」(運用) には、アプリケーションのリリース、製品の移行、運用、モニターが含まれます。このモニターによって、ソフトウェアに必要な将来の変更点が決められます。たとえば、問題を修正するメンテナンスや、アプリケーションを便利にしたり有効性を維持したりする新機能の追加による進化などがあります。

継続的なサイクル内のすべての作業を統合することで、システム開発に携わる企業は、市場にあるチャンスを一ち早く活用したり、顧客の声を迅速に取り入れてソフトウェアのエクスペリエンスを強化したりできます。

それでは、アプリケーションのライフサイクルで発生する主な問題を見てみましょう。

## 変更の統合



特殊な場合を除いて、ソフトウェア開発は常にチーム活動です。チームでのソフトウェア開発における主な問題の 1 つに、各開発者がアプリケーションに加えた変更の統合があります。

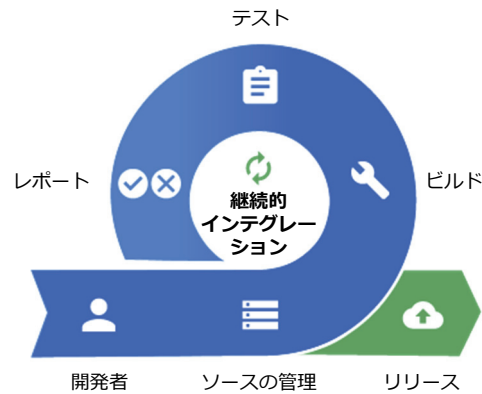
各開発者は、開発サイクルのなかで目的が達成されるまでコーディングとテストを繰り返し、機能の開発に取り組みます。

開発者の作業を統合して、開発された機能がアプリケーションの次のリリースに含まれるようにするには、どうすればいいでしょうか。

開発者のそれぞれが変更を GeneXus Server にアップロードしようとするすると、開発者が個別に作業している期間が長い場合は特に、多くの競合が発生します。複数の開発者が同じ名前のオブジェクトを定義していても、それぞれ目的が異なるかもしれません。また、誰かが既存のオブジェクトを変更して、ほかの開発者が使っている機能を削除してしまうかもしれません。

## 継続的インテグレーション

- 少なくとも 1 日に 1 回以上のペースで、変更を頻繁に統合する。
- 統合を自動的に検証する。



継続的インテグレーションでは、競合を避けるか、簡単に解決できるような小規模な競合に留めるために、可能であれば 1 日に 1 回以上のペースで、作業を頻繁に統合します。

統合が行われた後には、アプリケーションを自動的にセットアップおよびテストして、統合のエラーをできるだけ早期に検出する必要があります。既に機能していたものが機能しなくなることがないように確認する (リグレッションテスト) 場合も、自動テストが有効です。

## 継続的インテグレーションのツール



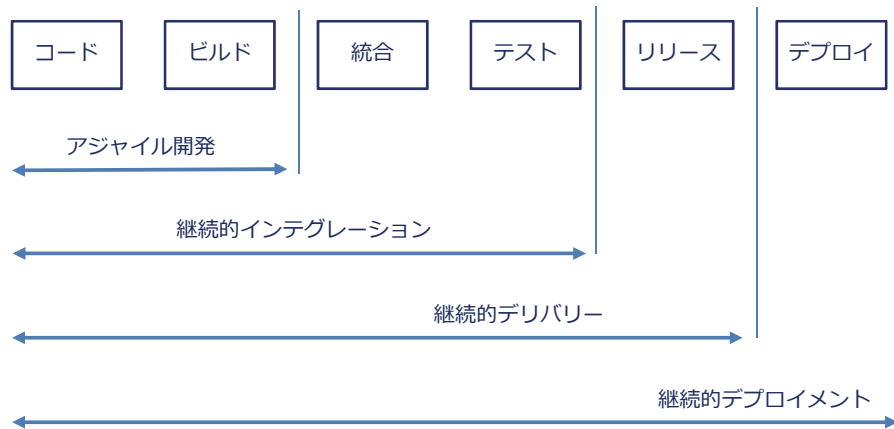
そのためには、GeneXus Server と Jenkins を使います。Jenkins は継続的インテグレーションのエンジンです。このツールにより、プロセス全体を実装できます。



開発者がコミットするたびに、少なくとも次の自動プロセスが実行されます: ナレッジベースの更新、すべてビルド、テストの実行。

すべてが問題なく実行されれば、ビルドは成功し、次に進むことができます。問題があった場合は、プロセスを停止して、競合を解決する必要があります。

## アプリケーションのライフサイクルにおける段階



アプリケーションのライフサイクルは、スライドに示す手順で構成されます。

アジャイル開発では、開発とビルドのプロセスのスピードを向上させます。

継続的インテグレーションでは、プロセスに統合と検証の自動化を追加します。

継続的デリバリーでは、リリースのセットアップを自動化し、1回クリックするだけで本番環境に公開できるようにします。

継続的デプロイメントでは、プロセスが自動化されたスムーズなものになり、製品の移行を自動化できます。

## DevOps プロセス



さらに、プロセスに本番環境でのアプリケーションの運用とモニターが含まれ、今後どの機能を開発するかを計画するための情報をフィードバックとして生成するようになれば、そこでようやく DevOps プロセスを実践していると言えます。



## 継続的インテグレーションのプロセスの作成とモニター

The screenshot shows the GeneXus IDE interface for configuring and monitoring Continuous Integration (CI). The main window displays the 'Continuous Integration' tab, which contains two tables:

**Integration Pipelines**

| Status  | Name           | Version     | Environment      | Run | Last Run        | Next Run            |
|---------|----------------|-------------|------------------|-----|-----------------|---------------------|
| Failure | JUnitStable    | JUnitStable | Java Environment | 69  | 9/10/2020 17... | 12/10/2020 10:31:51 |
| Success | JUnitStableNew | JUnitStable | Java Environment | 15  | 9/10/2020 17... | 12/10/2020 17:26:48 |

**Pipeline Activity**

| Run | Status   | Run Date        | Duration |
|-----|----------|-----------------|----------|
| 15  | Success  | 9/10/2020 17:26 | 00:06:05 |
| 14  | Unstable | 6/10/2020 20:19 | 00:01:01 |
| 13  | Failure  | 6/10/2020 20:18 | 00:00:00 |
| 12  | Failure  | 6/10/2020 19:38 | 00:00:00 |
| 11  | Success  | 6/10/2020 18:11 | 00:00:48 |
| 10  | Failure  | 6/10/2020 18:08 | 00:00:48 |

これまで見たとおり、DevOps の方法論を取り入れるには、継続的インテグレーションを実現している必要があります。つまり、アプリケーションの開発後に変更を統合して検証するプロセスを自動化します。

GeneXus IDE および GeneXus Server コンソールから、継続的インテグレーションのプロセス (パイプライン) を作成してモニターすることができます。

ナレッジベースを GXserver にアップロードした後、[ナレッジマネージャ] の [チーム開発] に移動し、[継続的インテグレーション] タブでパイプラインを作成できます。また、いつパイプラインが実行されたか、それぞれの実行時に行われたコミット、その結果、それぞれの実行の完全なログを確認することもできます。

詳細情報:

<http://wiki.genexus.jp/hwikibypageid.aspx?40706>

GeneXus を使用した DevOps の詳細情報については、こちらの Wiki を参照してください: <http://wiki.genexus.jp/hwikibypageid.aspx?40706>