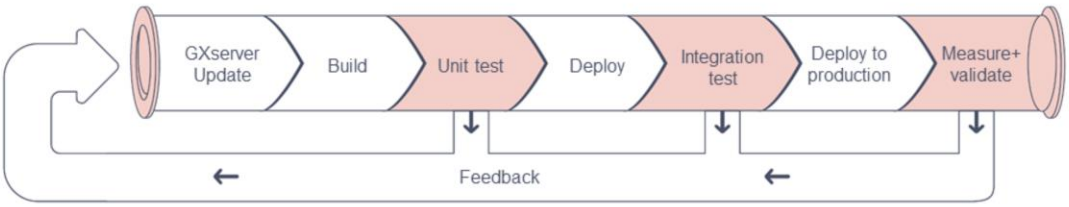

GeneXus™
by Globant

パイプラインの概要

GeneXus by Globant

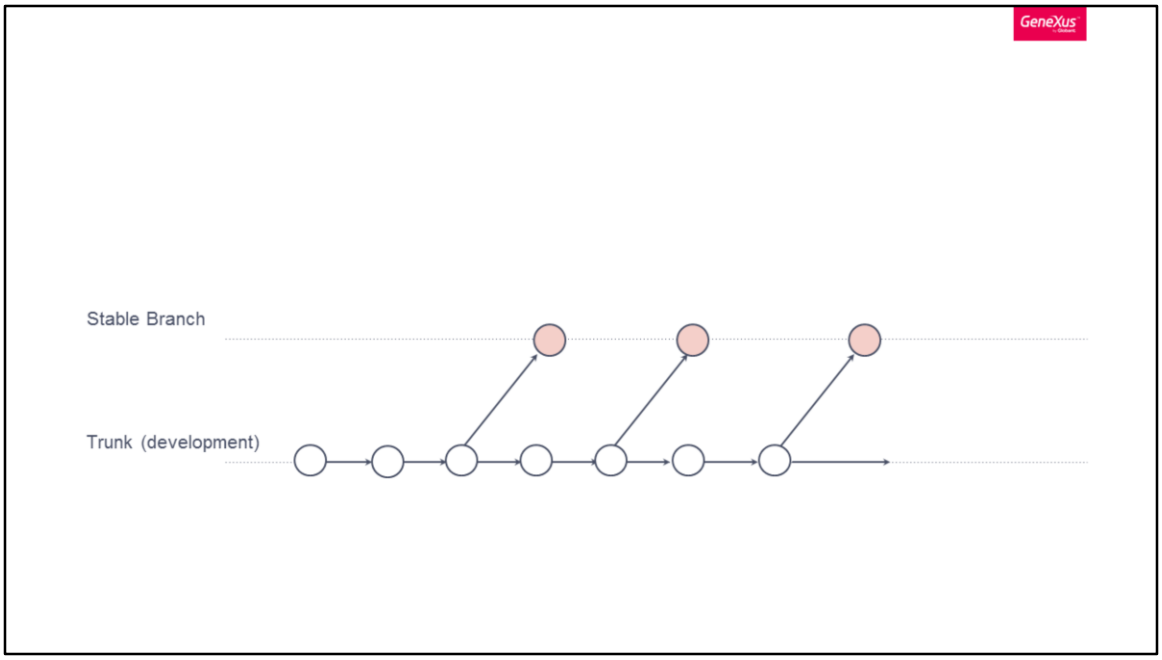


パイプラインの目標は、開発プロセスを自動化し、直近のアプリケーションのバージョンを頻繁に検証することです。
自動テストで問題が検出された場合、プロセスの各段階で開発チームへの迅速なフィードバックが行われます。

パイプラインを作成することで、生産性が向上し、より高い品質でより頻繁にソフトウェアをリリースすることができます。

トランクベース開発

GeneXus by Globant

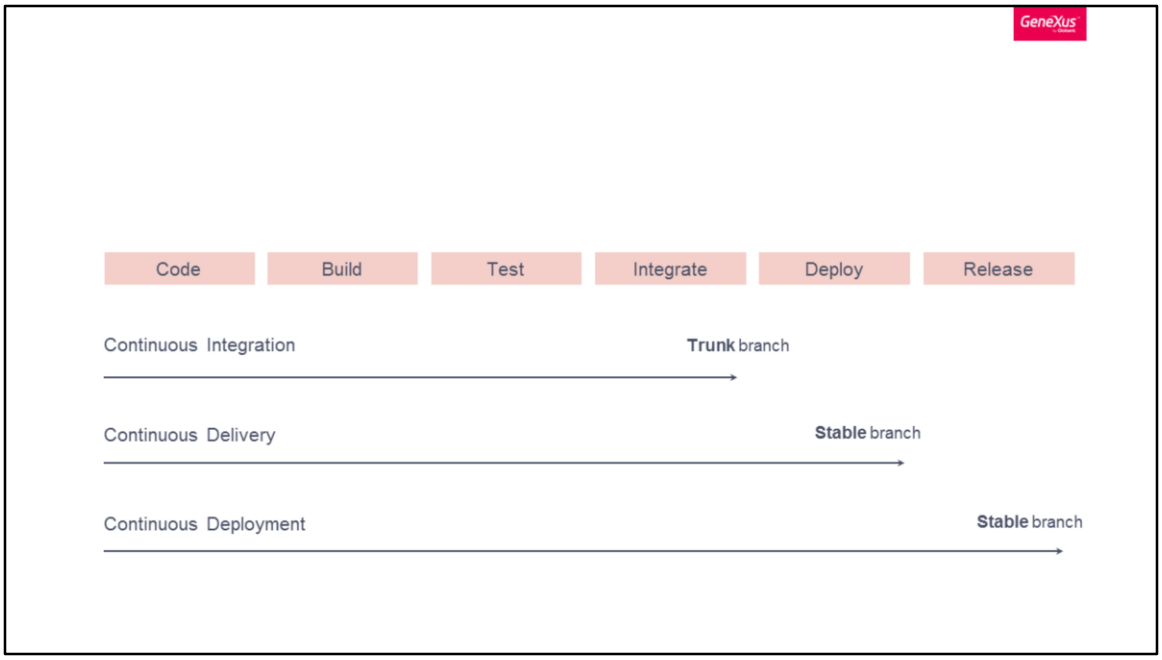


トランクベース開発（TBD）とは、ソースコントロール内でプロジェクトを管理するプロセスのことで、
プロジェクトで作業するすべての開発者がコードの変更をトランク（プライマリ/マスター）ブランチに直接コミットすることを指します。

Abstracta社としては、二つの異なる開発ブランチを使い始めることを推奨します。それは、トランク（開発版）と安定版です。
安定版は、あなたがリリースの一部にすることを決めた、すでに開発済みである機能の全てを含む統合ナレッジベースです。
一方、トランク（開発版）はいわゆる「ベータ」ブランチで、開発者は毎日新機能やバグフィックスに取り組んでいます。

安定版では通常、パイプライン全体が各ステージで自動化されているので、トランクからマージ（変更点を持ってくること）した後、
新しいバージョンの品質をチェックするために、異なるステージでテストを自動的に実行し始めるようにします。
バージョンの名前と量は、各チーム内のコンセンサスの問題に過ぎません。
ここで重要なのは、ブランチの種類ごとに定義された名前（ブランチ名）を持ち、チームがそれらのブランチモデルを正確に認識していることです。

http://wiki.genexus.jp/hwikibypageid.aspx?38329



パイプラインは、各要素の出力が次の要素の入力になるように配置された処理要素の連鎖を表します。
これは、パイプラインがソフトウェアの配信をいくつかの段階に分解し、新機能の品質を検証し、
バグによってユーザーに影響を与えることを防ぐことを意味します。

自動テストによるパイプラインは、継続的インテグレーション、継続的デリバリー、継続的デプロイメントを実現することができます。

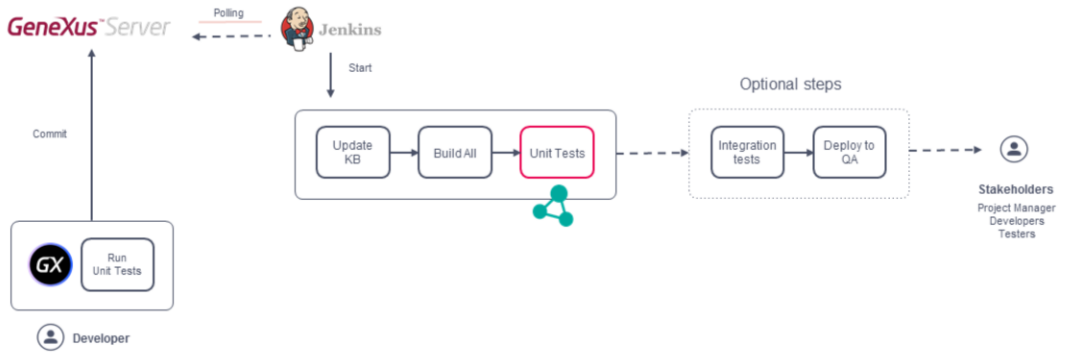
継続的インテグレーションとは、各開発者のコードを統合し、ユニットテストによって検証することです。
継続的インテグレーションは、通常Trunkブランチで設定されます。

リリースを自動化し、ワンクリックでアプリケーションを本番稼動させることができるようになれば、これは継続的デリバリーとなります。
さらにプロセス全体が自動化され、コードが自動的に本番環境に投入される

と、そこで継続的デプロイが実現されます。
通常、最後の二つのスキームは、Stableブランチに対して設定されます。

Source:
https://www.genexus.com/es/productos/genexus/versiones/genexus17/videos/cultura-devops-con-genexus (min 5)

Trunk/Dev branch パイプライン



トランク版は、開発者が新機能のコーディングやバグの修正を行っている場所です。
これらの変更は多くの開発者によって異なるオブジェクトに対して行われるため、
このブランチに頻繁にコミットされることになります。

このバージョンで作業する最善の方法は、すべての開発者の変更を頻繁に統合し、
システムを常に「ビルド可能な状態」、
つまりナレッジベースのコードに問題がない状態を開発習慣とすることです。
開発者がGeneXusServerに変更をコミットしたときに自動タスクが起動するように、
トランク版で継続的インテグレーションを設定し、毎日実行することをお勧めします。
そうすることで、最近の変更がビルドを壊さないように、また、品質基準を尊重するようにプロセス内で自動的にチェックすることができます。

Jenkinsのような継続的インテグレーションサーバーがある場合、最初に行うべきことは新しい変更を待機するリスナーを追加することです。

この場合、パイプラインは少なくとも次のステップを実行する必要があります。

1. GeneXusServer（トランク版）から最新の変更点を「更新」する。
   これにより、ビルドする資材（KB）を最新のコミットに更新することができます。

2. すべてビルド
   これにより、KBオブジェクトの指定、生成、コンパイルを行う「すべてビルド」が実行されます。

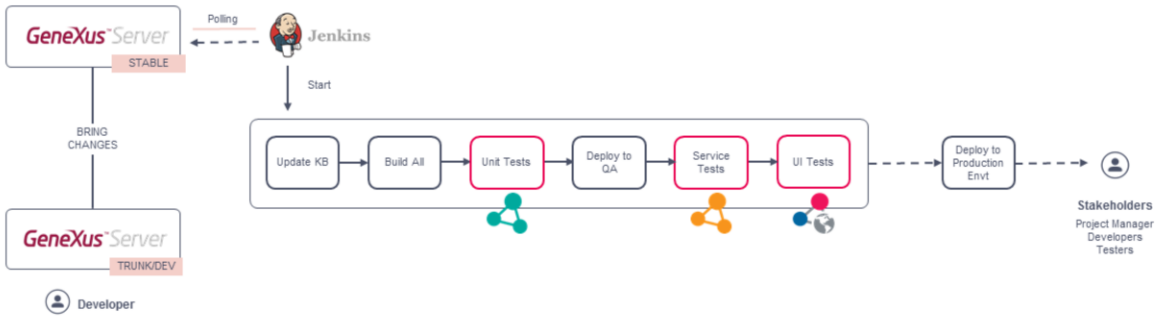
3. ユニットテストの実行

さらに推奨するステップとして、グッドプラクティスに準拠しているかどうかを確認するための静的コードチェック（KB Doctor ツールなど）の実行や、

いくつかの基本的な自動統合テストの実行、そして、手動テストのための QA 環境へのデプロイを行うことが可能です。

このトレーニングでは、このようなパイプラインを設定する方法を学びます。

Stable branch パイプライン



このバージョンは、新しい統合バージョン（新しいリリースの候補）の一部とすることを決定した主な変更をホストします。
これらの変更は、通常、他のバージョンからのコミットのセットで、たとえば、一人または複数の開発者が変更するトランク版のものがあります。

安定版では、KBを自動的にビルドして品質基準をチェックするだけでなく、さまざまなテスト（APIテストとUIテスト）を実行します。
その結果、このバージョンに変更を加えるたびに、パイプラインはそれが「デプロイ可能」な状態であること、そしてそれが機能的であることをチェックします。

パイプラインのジョブは、トランク版で定義されたものと同じです。
さらに、アプリケーションをQA環境にデプロイし、これらの後にサービスとUIテストを実行するステップがいくつか追加されています。
画像では、継続的デリバリーアプローチとして、更新、ビルド、ユニットテストの実行、QA環境へのデプロイ、サービステストの実行、
UIテストの実行という自動タスクがあることがわかります。

このアプローチでは、本番環境へのデプロイは手動であることに注意してください。
このステップを自動化することは可能であり、このアプローチは「継続的デプロイメント」と呼ばれます。

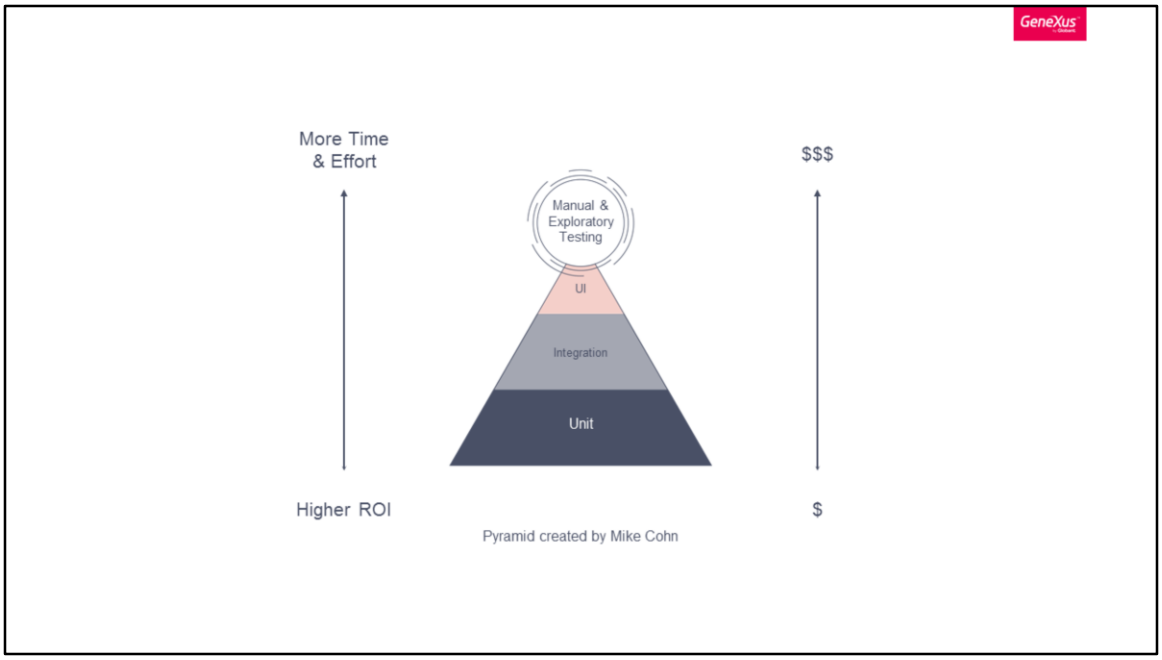
このアプローチを実現するためには、新しいバージョンが顧客にデプロイされることを確認するための、堅牢なテストが必要です。
自動化のコスト、テストの実行頻度と速度、テストの実行環境、チェックの価値と効果などの要因によって、パイプラインに付加価値を与えるテストの自動化にはいくつかの層があります。

最大のリスクはどこにあるのかを特定し、さまざまなレベルのテストでそれを軽減する方法と、自動化ピラミッドに基づいて、
各組織はどのタイプのテストをどれだけ実装するかを決定します。

自動化されたテストは、パイプラインに付加価値を与え、迅速な検証を行い、手動での検証の必要性を減らして顧客に価値を提供することができます。
これにより、より速く、継続的に、より高い品質で製品を提供することが可能になります。

テスト自動化ピラミッド

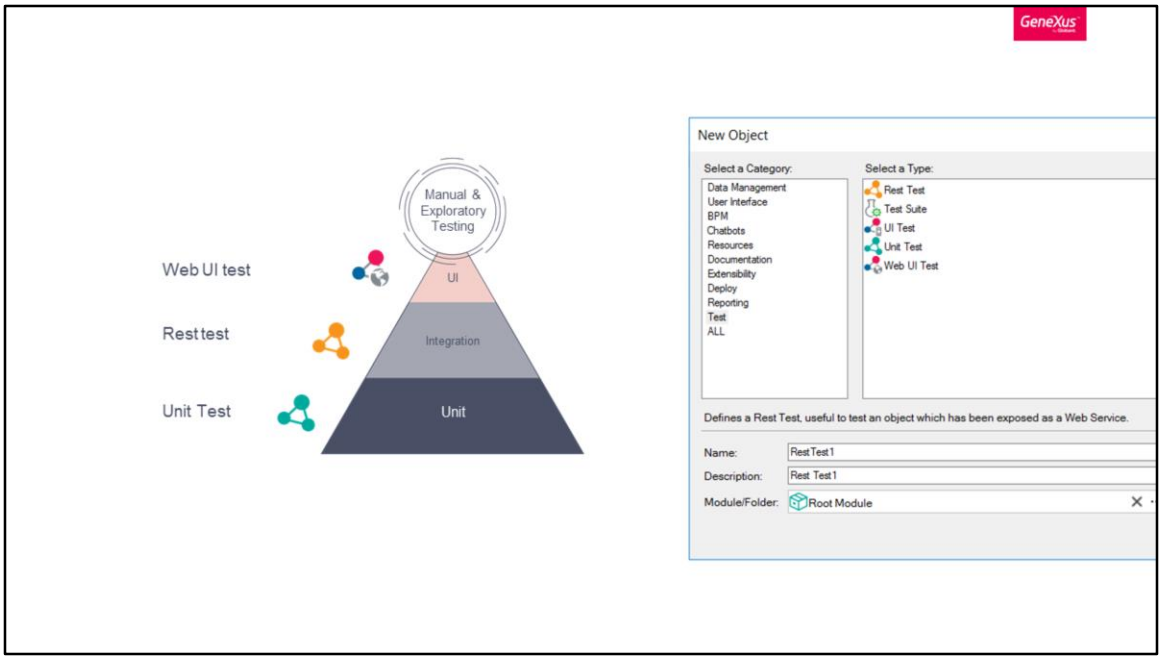
GeneXus by Globant



アプリケーションを自動化する前に、マイク・コーンによる理想的なテスト自動化ピラミッドで表される、異なるテストの層を理解する必要があります。

ピラミッドを下から見ていきます。
最もテストの比重が大きい1番目のテスト層は、ユニットテスト層です。この層は、最も速く実行でき、最も安く維持することができます。
2番目のテスト層では、通常、アプリケーションの異なるコンポーネントをテストする統合テスト、すなわち、サービステストがあります。
3番目の層では、エンドツーエンドテストのようなUIテストがあります。
このテストは、実行速度が遅く、維持するために多くの費用がかかります。
また、回帰テストやユーザー受入インターフェースのテストに有用ですが、アプリケーションをデプロイする必要があり、テスト環境はより高価になります。

最後に、手動テストと探索的テストは、自動化できない機能性のために確保されます。
テストの正確な組み合わせは、各チームによって異なりますが、各特定のテストアプローチに投資する努力は、このピラミッドに従うべきです。



GeneXusでは、どのタイプのテストから始めるかは、KBの構造やプログラム、パイプラインやアプリケーションのニーズによって決まります。

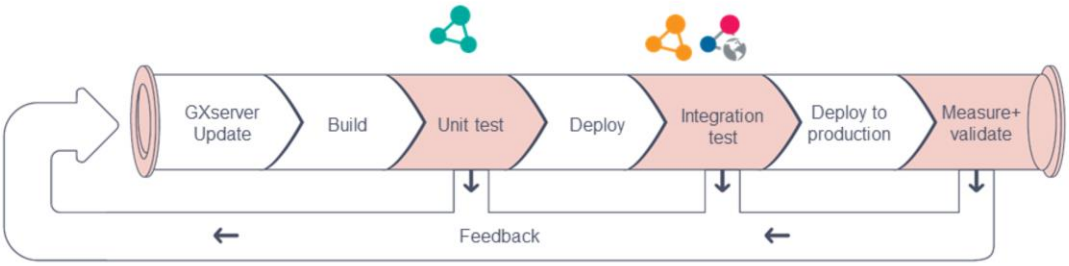
すべての種類のテストは、GeneXus IDEから作成されます。

ユニットテストは、Unit Testオブジェクトで作成されます。
このオブジェクトを使用すると、プロシージャ、データプロバイダー、およびビジネスコンポーネントのような他のオブジェクトにカプセル化されたビジネスロジックをテストすることが可能です。

Rest Testオブジェクトを使用すると、RESTサービスとして公開されたオブジェクト（プロシージャ、データプロバイダ、およびビジネスコンポーネント）を HTTP経由で呼び出してテストすることができます。

GeneXusのUIテストは、Web UI TestオブジェクトまたはUI Testオブジェクト（モバイル）で実装されます。

また、Test Suiteオブジェクトも用意されており、異なる種類のテストをグループ化することが可能です。



自動テストはパイプラインに追加され、異なる環境での各変更後のアプリケーションの動作をチェックします。


GeneXus™
by Globant
training.genexus.com
wiki.genexus.com