**Automated Release Notes**

Project Proposal

Contents

[1. Introduction 2](#_Toc441969784)

[2. Vision and Goals of the Project 2](#_Toc441969785)

[3. Users of the Project 3](#_Toc441969786)

[4. Scope and Features of the Project 3](#_Toc441969787)

[5. Solution Concept 3](#_Toc441969788)

[5.1. Architecture of the Project 3](#_Toc441969789)

[6. Acceptance Criteria 5](#_Toc441969790)

[7. Release Planning 6](#_Toc441969791)

# Introduction

Continuous Integration (CI) is a development practice that requires developers to integrate code into a shared repository even several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems as soon as possible. Releasing software with a Continuous Integration (CI) Pipeline can greatly increase the speed with which code gets released. CI Pipelines can largely replace manual processes of releasing. When it comes to the content of a release however, most tools focus on relating each build to a set of changes in source control. This is great for developers to determine the source of a bug in a particular build, but the source control does not hold much value when it comes to communicating the business content of the build. When a build gets deployed, the business wants to know what the content of the build is, so that they know what to test, as well as what can be released to production. This often means a manual step in the CI process is necessary: the developer must look up the work items in the work item tracking software, and enter them manually into the release tracking software. The goal of this project is to eliminate that manual step of entering release notes by integrating the build pipeline with Work Item Tracking software.

# Vision and Goals of the Project

“Automated Release Notes” involves the development of a Java plugin for Teamcity (Build Pipeline) which would integrate it with Work Item Tracking software(in this case Visual Studio) for generating a useful text file containing work item information which would be used by Business/Quality Analysts(QA) for testing purposes. This would eliminate the manual step of creating release notes by the developers while checking in their changes in source control software.

High-level goals of Automated Release Notes include:

* Providing an efficient tool to eliminate the manual step of entering release notes by integrating the build pipeline (Teamcity) with work item tracking software (Visual Studio). This implies that the Teamcity plugin should be able to parse commit messages from Source Control for work items and be able to get data from Work Item Tracking Software to output the work item information to a text file.
* Providing QA with a better understanding of the latest build and test the critical content of the build. This implies the generation of a text file with relevant work item information which would help the QA for testing the impacted functionality.
* Eliminating manual errors committed by developers while manually creating the release notes. It provides a way for a more effective team communication over the product.
* Enabling the improvement over the efficiency of releasing software with a continuous integration.

# Users of the Project

Automated Release Notes will mainly be used by software developers to create automated release notes notifying the Quality Analysts regarding description of the recently checked in work items for quality and testing purposes. Both – Developers and Quality Analysts will be the beneficial users of this product.

# Scope and Features of the Project

Described below are the features of the product:

* An efficient and user friendly interface within Teamcity for automatic release notes generation corresponding to each code check in.
* This approach should be configurable for multiple deployment servers such as Octopus Deploy.
* The end-product could be configurable for multiple tracking servers such as VSO
* Scalability: can scale to large number of users, projects and services

# Solution Concept

Described below is the high level outline of the solution:

## Architecture of the Project

Below is the description of the system components that are the building blocks of the architectural design:

**Build Pipeline (Team City):** Team City is an enterprise of JetBrains which is a Java based platform that implements Continuous Integration. It has the following features:

* The commits are [Gated](https://en.wikipedia.org/wiki/Gated_Commit)  (this feature prevents developers from breaking sources in a [version control system](https://en.wikipedia.org/wiki/Revision_control) done by running the build remotely for local changes prior to commit).
* Build [Grid](https://en.wikipedia.org/wiki/Grid_computing). Allows running multiple builds and tests under different platforms and environments simultaneously.
* Integrated [code coverage](https://en.wikipedia.org/wiki/Code_coverage), [inspections](https://en.wikipedia.org/wiki/Software_inspection) and [duplicates](https://en.wikipedia.org/wiki/Duplicate_code) search.
* Integration with [IDEs](https://en.wikipedia.org/wiki/Integrated_development_environment): [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software)), [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA" \o "IntelliJ IDEA), [Visual Studio](https://en.wikipedia.org/wiki/Visual_studio).
* Platforms supported: [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [.NET](https://en.wikipedia.org/wiki/.NET_Framework) and [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)).

**Source Control:** It is a software that constructs versions as arbitrary sets of revisions. The process of extracting an arbitrary version takes essentially the same speed and is thus more useful in environments that rely heavily on branching and merging with multiple "current" and identical versions.

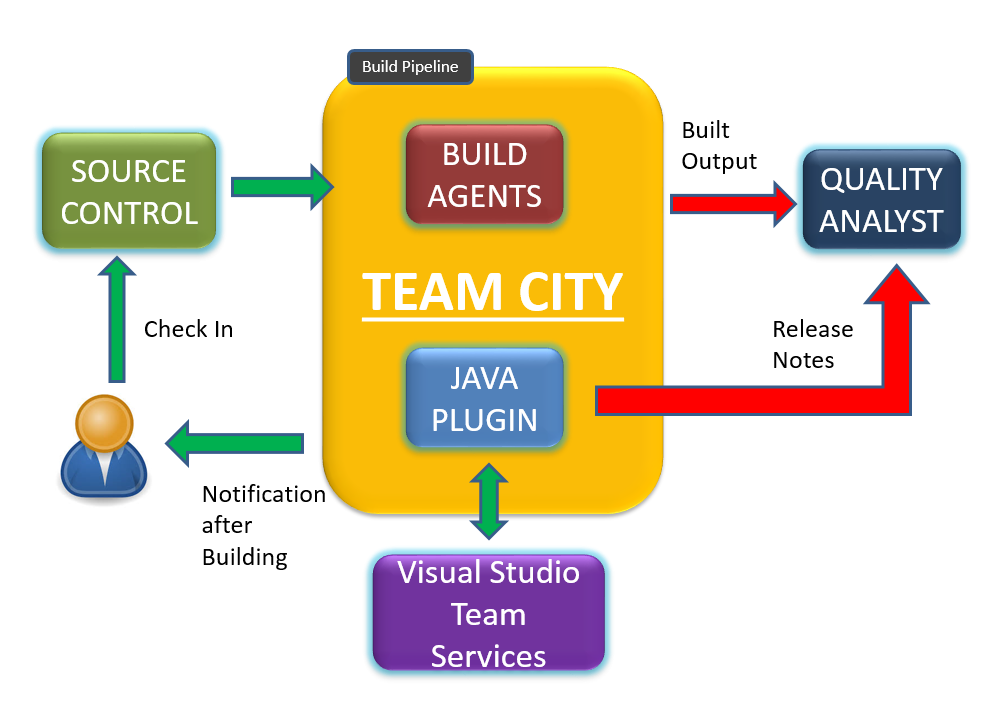
**Work Item Tracking Software (Visual Studio Team Services):** The work item tracking software which would be integrated with the Teamcity through Java API is Visual Studio Team Services (VSTS). Long term goal would be to generalize our software so that the end user can configure plugin for any work item tracking.

Visual Studio Team Services is service for teams to share code, track work and ship software for any language and all this is provided in a single package. VSTS has work item tracking features designed to assist enterprise software development teams to manage their work and software defect tracking. In VSTS, a work item tracks this work, and can be one of the individual tasks required to complete a project, a bug that needs to be fixed, a customer scenario that the project needs to address, or any other piece of work in a project that needs to be tracked in a database.

**Java Plugin for Teamcity:** The Java API which would be developed as a part of this project will have the following functionalities:

* It should be able to interact with the source control software and read and parse commit messages from it for work items.
* It should be able to interact with the Visual Studio Team Services and give output about the work item information in a text file.

Most of the Software used to develop the project would be the ones developed by Jet Brains. The IDE used to develop the Java API would be IntelliJ.



**Architectural Design**

# Acceptance Criteria

The minimum acceptance criteria for this project is defined below-

* A Teamcity plugin in the form of a Java API that could integrate the Teamcity and Visual Studio Team Services work item tracking module to generate a text file containing the work item description of the checked in code which could serve as the release notes for the Quality Analysts.
* A TeamCity based server would be established and a constant and stable API would be provided.
* The build of our product, which could be applied to a live environment.
* The end-product would be configurable for Visual Studio Team Services (VSTS) work item tracking software.

Stretch goals for this project are as follows-

* Enabling the update of work item in VSTS using the Teamcity plugin.
* Enabling the update work item status in VSTS based on the commit parsing.
* Generalize the product so that the end user can configure plugin for any work item tracking software such as JIRA, YouTrack, etc.

# Release Planning

Detailed user stories and plans are on the Trello board.

Below is the initial draft of the scheduled product development Milestones:

**Release #1:**User Stories- Environment Setup, Demo Plugin

Environment Setup: Setup the development environment and components required to develop the Teamcity Java plugin. This would include setup of IDE – IntelliJ, setting up Teamcity application and account, setting up VSTS.

Demo Plugin: Develop a fully functional demo plugin for Teamcity.

**Release #2:**Connecting a plugin to VSTS which can be used to read any work items and read some commits for a build.

**Release #3:**Parsing commits for work item numbers for a build.

**Release #4:**Download Work item information for work item numbers.

**Release #5:**Generating a final output to text file.