



User's Guide of the CROSSMINER Advanced Integrated Development Environments Plug-in

FrontEndART Szoftver Kft.
Version 1.0.0.rev0 • October 17, 2018

Chapter 1

Introduction

The general layout of the development and testing process are shown on Figure 1.1

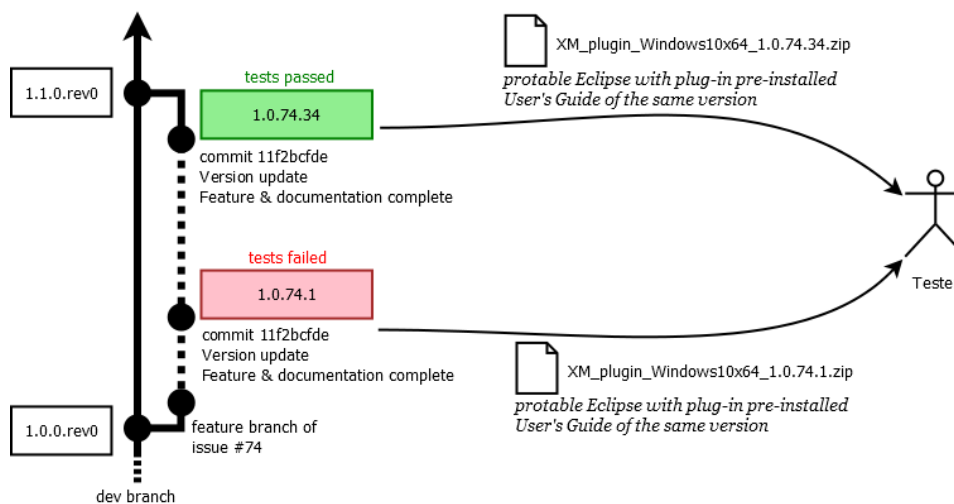


Figure 1.1: Overview of the testing and development process

1.1 Versions

The version number consists of three parts separated by period. It follows the layout $\langle main \rangle . \langle sub \rangle . \langle issue \rangle . rev \langle revision \rangle$. To understand when and how to change these parts please consider the following guide.

Increasing main version Request rights of lead developer or above. It will be increased when the current version are shipped and demonstrated to the (whole) consortium.

Increasing sub version Could be changed by any participant. Increase it by one respecting to the current version of the `dev` branch when a new feature are finished, passed all required tests and merged into the current development version (on `dev` branch). Reset it to 0 every time main version number increases.

Changing issue number You have to set this part to be equal the identification number of the current issue associated to the feature branch you are on. Set it 0 on non-feature branches.

Incrementing the revision Could be changed by any participant. Increment it by one, before you send a new version for testing. It could be the first version containing a testable new feature or any fixes applied due to testers concerns. Reset it to 0 at the beginning of each new feature branch.

There are some additional rules need to consider, partially implicated by the these statements. You have to follow these as well the previous ones.

Development versions Each version on the `dev` branch should follow this scheme: $\langle main \rangle . \langle sub \rangle . 0 . rev0$, for example *1.12.0.rev0*.

Testable versions Each version sent for testing should follow this scheme: $\langle main \rangle . \langle sub \rangle . \langle current-issue \rangle . rev \langle revision \rangle$, for example *2.23.42.rev11*.

Version tagging Each version should be explicitly tagged in the Git repository using the following commands (if your HEAD points to the relevant commit):

```
git tag <version>
git push origin <version>
```

Issue-branch-feature tracibility Each feature branch has to have a single and unique issue. Do not change (create new) version off the plugin after merging a branch contains only refactoring without any new features or fixes.

Tracability of User's Guide The User's Guide (this document) has to follow the version any version changes. You have to update the guide before you change the version number.

1.2 Structure of Releases

After chancing the version you have to assemble a new release package. By doing so please follow these rules.

Naming Each release package should be named by the following scheme:

XM_plugin-⟨operation-system⟩-v⟨version⟩.zip,

for example `XM_plugin.Windows10x64_2.23.0.rev0`

Content Each release should consist of a single, self-contained compressed folder, containing a portable Eclipse with the plug-in pre-installed.¹

Placing Documentation The relevant version of this document should be placed in the root folder of the compressed package in PDF format.

1.3 Testing

Each feature and function should be tested against expected functionality described in this document. Any discrepancies should be reported.

Reporting failed tests All failed test should be reported using a dedicated issue recorded in the project. <https://git.sed.hu/geryxyz/crossminer/issues/new>

Content of the report Each report should contain a detailed description of the action and condition of test to be repeatable. You have to specify the concrete version in which the divergence occur and the exact place in this document where the feature was described.

Goal of testing As a tester you have to ensure that each features are working as described in this document with reasonable reliability.

¹Except in the cases, when the goal of testing is installation.

Chapter 2

General features

In this chapter we present several functions which can not be linked to any single feature. These embodied an overview about the general principles which interwove the whole plug-in.

Requirement U12 SHALL Status: **done**
Able to obtain the API results in JSON format



Requirement U13 SHALL Status: **done**
Able to use the API over REST



Requirement U18 SHALL Status: **partially done**
API is utilised by all UIs (dashboard, IDE plugin)



Requirement U154 SHALL Status: **to do**
Eclipse users can invoke code recommender via an easy shortcut



Requirement U156 SHALL Status: **to do**
Eclipse IDE proposes a dedicated view or perspective for recommendations



Requirement U158 SHALL Status: **to do**
Eclipse IDE plugin can be installed via the Marketplace



Requirement U159 SHALL Status: **done**
Eclipse IDE plugin can be installed via an update site



Requirement U220 SHALL Status: **to do**
User and admin documentation is embedded into the UI



Requirement U225 SHALL Status: **done**

Plugin supports the latest supported release of Eclipse

**Requirement D138** SHALL Status: **done**

The CROSSMINER REST API shall use UTF-8 encoding for all kind of data sent or received in text mode.

up to date

Chapter 3

Acquire Recommendations

There are several kind of suggestions and recommendation which could be helpful for the developer during their daily tasks. In this chapter we introduce those features which result some kind of recommendation. One of the main properties is the type of entity which are the subject of the suggestion. To simplify any further discussion our tool only permit recommendation with a single subject type. You can find more details in the following sections.

Requirement U1 **SHALL** **Status:** waiting for server side implementation
Able to extract the development dependencies of a project

Nothing to do directly on IDE side?

Requirement U2 **SHALL** **Status:** waiting for server side implementation
Able to extract the test dependencies of a project

Nothing to do directly on IDE side?

Requirement U3 **SHALL** **Status:** waiting for server side implementation
Able to extract the runtime dependencies of a project

Nothing to do directly on IDE side?

Requirement U157 **SHALL** **Status:** to do
Eclipse IDE proposes filtering options for recommendations



Requirement U223 **MAY** **Status:** waiting for server side implementation
Provides recommendations to improve test coverage

Howto: e.g. label recommendation as TEST

Requirement D75 **SHALL** **Status:** partially done
Recommendation query shall be initiated from the menu, from context menus, and from the toolbar in the IDE. So the user can easily start working with the recommendations.



Requirement D76 **SHALL** **Status:** to do waiting for server side implementation



Some recommendation types needs to be defined in details (with examples).

The IDE shall use a view to list the code recommendations for the examined project. The view shall provide hierarchical grouping of listed items, filtering by meta-properties, and navigation.

API needs to be defined. According to L'Aquila the server will receive only a string for a prepared set of options.



Requirement U167 **SHALL** **Status:** waiting for server side implementation
CROSSMINER IDE provides the ability for the developer to give feedback on the usefulness of the advices in the given situation

Not IDE! Needed for U167 and D79

Requirement D105 **SHOULD** **Status:** waiting for server side implementation
The CROSSMINER REST API should provide an interface that allows the user to report if the answer of query previously returned by the API was inappropriate answer for the query. So the knowledge base gets feedback and can be improved.

3.1 Library Based Recommendations

Recommendation which subject are libraries are called *library based recommendations*.

Not IDE! Howto: server detects, IDE provides query?

Requirement U4 **SHOULD** **Status:** waiting for server side implementation
Able to warn about superfluous dependencies

Not IDE! Howto: server detects, IDE provides query?

Requirement U5 **SHOULD** **Status:** waiting for server side implementation
Able to warn about overly precise dependencies

On IDE side: extract from maven or query from server



Requirement U81 **SHALL** **Status:** done
Able to identify and list the third-party components used in a project.

Enough to identify maven libs on IDE side



Requirement D78 **SHALL** **Status:** partially done
The IDE shall provide an interface to the user, where libraries that match some recommendations are listed with their selected set of details in filterable and sortable form. So the developer can easily compare the libraries and choose those best fit to their expectations.



Requirement D79 **SHOULD** **Status:** waiting for server side implementation
The IDE should provide the ability to the user to mark a library proposed for a query as “not appropriate”. This information can help in the improvement of the knowledge base.



Requirement D80 SHALL Status: **done**

The IDE shall provide an interface where known information from a single library is shown. This interface can help the user to check details of the library, and reach out for more information through links (e.g. to project home page).

Requirement D100 SHALL Status: **waiting for server side implementation**

The CROSSMINER REST API shall provide an interface that accepts a name and version description and returns the library that is described by them.

Not IDE! Needed for D78

Requirement D101 SHALL Status: **done**

The CROSSMINER REST API shall provide an interface that accepts a library and returns detailed information about it (including description, metric values, URLs).

Not IDE! Needed for D78

3.1.1 Searching Additional Libraries

One sub-type of library based recommendation when the goal of the developer is to find new libraries based on some user specified search criteria.

Requirement U172 SHALL Status: **waiting for server side implementation**

CROSSMINER IDE provides developers code templates and example of codes related to the usage of the API of a specific project

Source code recommendation on the IDE side, ready.

Requirement D77 SHALL Status: **to do waiting for server side implementation**

The IDE shall provide an interface, where recommendations against a new (to-be-used-in-the-project) library can be given. So the user can describe features and functionalities they wants to perform using an external library. The user can also give some constraints, e.g. minimal age or users of the library.



On the IDE side, it is a free text search (with some pre-defined constraints, if needed). Free text search is okay as far as it satisfy all the use cases.

Requirement D99 SHALL Status: **partially done waiting for server side implementation**

The CROSSMINER REST API shall provide an interface that accepts a description and some filtering constraints and returns a list of libraries that match with the description and fulfil the constraints.



Not IDE! Needed for D77.

3.1.2 Searching Similar Libraries

There are usually some pre-existing selection of libraries which serve as a base to find more relevant API-s. We split these kind of recommendation into two major sub-category: when the pre-existing set of libraries are already installed or not.

Not IDE! Needed
for U174

Requirement D102 **SHALL** **Status:** waiting for server side implementation

The CROSSMINER REST API shall provide an interface that accepts a library, a description and some filtering constraints and returns a list of libraries that match with the description, fulfil the constraints, and can be used as a replacement to the given library.

Suggestion Based on Non-installed Libraries

Suggestion Based on Already Installed Libraries

Not IDE! Needed
for D81



Requirement U174 **SHOULD** **Status:** waiting for server side implementation

CROSSMINER IDE provides suggested alternatives to the usage of third-party jar which offer the same range of services as a jar used in current project



Requirement D81 **SHALL** **Status:** to do waiting for server side implementation

The IDE shall provide an interface to the user, where recommendations for replacing a library used in the current project with some alternative libraries can be given. The user can select the library currently used in the project to be replaced and can give further recommendations against the alternatives.

3.1.3 Handling Changed and Deprecated APIs

Third party libraries are prone to change and evolution. To help the developers adapt their project to these changes we defined an other sub-category of recommendations, namely when their goal is to provide information about modified or deprecated interfaces.

Not IDE! Needed
for U160-162, U164,
U168, U169

Requirement U70 **SHALL** **Status:** waiting for server side implementation

Able to identify the list of changed third-party API methods from the source code of the third-party API

Not IDE! Needed
for U160-162, U164,
U168, U169

Requirement U71 **SHALL** **Status:** waiting for server side implementation

Able to identify the list of deprecated third-party API methods from the source code of the third-party API

Not IDE! Needed
for U160-162, U164,
U168, U169, U173,
U175

Requirement U72 SHALL Status: waiting for server side implementation

Able to determine migration pattern from two (or more) code snippets when one of them uses the old third-party API and the other uses the new third-party API



Requirement U80 SHALL Status: waiting for server side implementation

Able to detect from the configuration settings if a new version of a used third-party library is available

Not IDE! Needed
for U160-162, U164,
U168, D83-84

Requirement U130 SHALL Status: waiting for server side implementation

Able to identify if the developer is not using the most recent version of a library and provide notification

Not IDE! Needed
for U160-162, U164,
U168, D82-84

Requirement U160 SHALL Status: to do waiting for server side implementation

CROSSMINER IDE notifies if a new version of a third-party API used by the project on which the developer is working is available



Requirement U161 SHALL Status: to do waiting for server side implementation

CROSSMINER IDE notifies if a new version of a third-party API used by the project on which the developer is working breaks backward compatibility



Requirement U162 MAY Status: to do waiting for server side implementation

CROSSMINER IDE is able to offer the use of the newest version of a third-party API utilised in the project



Requirement U163 SHALL Status: partially done waiting for server side implementation

CROSSMINER IDE is able to identify and navigate to those places that became suspicious for changing behaviour after the third-party API version used in the project has changed



Requirement U164 SHALL Status: partially done waiting for server side implementation

CROSSMINER IDE is able to mark the usage of deprecated third-party APIs in the source code the developer is working on.



Requirement U168 MAY Status: to do waiting for server side implementation



CROSSMINER IDE is able to notify the developer about third-party API changes that are in design or development phase



Summary on IDE side

Requirement U169 **MAY** **Status:** to do **waiting for server side implementation**

CROSSMINER IDE is able to provide an overview of the impact of a third-party API change on the project the developer is working on



Howto: simply insert the received snippet

Requirement U173 **SHALL** **Status:** partially done **waiting for server side implementation**

CROSSMINER IDE assists developers to migrate the current version of a third-party jar to the new version by providing a list of required changes, advices and code templates



Howto: simply insert the received snippet

Requirement U175 **SHOULD** **Status:** partially done **waiting for server side implementation**

CROSSMINER IDE assists developers to address a deprecated API by proposing an alternative for obtaining the same behaviours of the code



Requirement D82 **SHALL** **Status:** to do **waiting for server side implementation**

The IDE shall provide the user the ability to initiate library version check. So the user is notified if some libraries used in their project have new versions.



Requirement D83 **SHALL** **Status:** to do **waiting for server side implementation**

The IDE shall perform library version checks on the libraries used in the current project. The check is performed on project load, and only marks the upgradeable libraries.



Requirement D84 **SHALL** **Status:** to do **waiting for server side implementation**

The IDE shall be able to show if a library used in the current project has a new version that satisfies some pre-determined criteria set globally for library upgrades by the user. So the user can see the relevant results of the library upgrade search.

ready, up to date



Requirement D85 **SHALL** **Status:** partially done

The IDE shall provide an interface where the details of an available library upgrade can be shown. These may include the new version number, release date, number of users, number of bugs, estimated impact of the upgrade, etc.

dy, up to date

Requirement D86 SHOULD Status: partially done

The IDE should provide the ability to initiate a library upgrade that marks all those places in the source code that needs rework due to the change of the library version.

Requirement D88 MAY Status: partially done

ready, up to date



The IDE may perform the steps of a library upgrade autonomously (if the user requested the upgrade).

Requirement D103 SHALL Status: partially done waiting for server side implementation

Not IDE!

The CROSSMINER REST API shall provide an interface that accepts a library and filters, and returns a list of versions available for that library and match the filters.

Requirement D104 SHALL Status: waiting for server side implementation

Not IDE! Needed for D86, D88

The CROSSMINER REST API shall provide an interface that accepts a library and two versions, and returns the differences between these library versions. Differences include removed, deprecated, changed, new API elements, and API elements with changed behaviour.

3.2 Source Code Based Recommendations

In this section we elaborate features related to those recommendation which subject entities are present in the source code of the project under development. They usually retrieve some code chunk, which could be annotated to ease further understanding.

Requirement U73 SHALL Status: partially done

Not IDE! Needed for U170-171...

Able to identify the part of the API that the developer is currently using to provide code snippets in relation with current development activity

Requirement U170 SHOULD Status: partially done waiting for server side implementation

no data for “as suspicious”

CROSSMINER IDE provides the ability for the developer to ask recommendations for a code chunk previously marked by the CROSSMINER IDE as suspicious

Requirement U171 SHOULD Status: done

CROSSMINER IDE provides the ability for the developer to ask recommendation for an arbitrary code chunk or code element

**Requirement D92** SHOULD Status: **partially done**

The IDE should provide the ability to the user to select a code snippet or a file and ask for code recommendations for it. So the user can check whether there is a better practice to solve her problem.

**Requirement D93** SHOULD Status: **partially done**

The IDE should be able to show recommendations assigned with code elements. So the user can see what the knowledge base suggested for a code snippet.

Not IDE! Needed
for U170-171, D91-
94

Requirement D107 MAY Status: **waiting for server side implementation**

The CROSSMINER REST API may provide an interface that accepts code snippets and library context, and returns recommendations on how to improve that part of the code.

3.2.1 Retrieving Suggested Code Snippets

Based on the current development context, our plug-in is able to yield a set of source code snippets (code chunks), which could be useful to implement or to understand various features.

done, up to date

**Requirement D91** MAY Status: **done**

The IDE may insert a recommendation in the code if the user accepts it and requested it on a code position.

**Requirement D94** MAY Status: **partially done waiting for server side implementation**

The IDE may be able to process recommendations and perform code migration autonomously. So the user can point to a recommendation and the IDE replaces the old code to the new one.

“IDE replaces ...”
will probably not be
implemented, un-
less the server gives
proper recommen-
dations

**Requirement D106** MAY Status: **waiting for server side implementation**

The CROSSMINER REST API may provide an interface that accepts a list of libraries and a description and returns recommendations about how to implement the described functionality using the given libraries.

Not IDE! Needed
for U170-171, D91-
94

3.3 Text Based Recommendations

Finally there are tons of documentation and discussion available for various topics, which could be useful for the developers. Those recommendations which yield some natural language documents (or reference to them) are called *text based recommendations*.

3.3.1 API Usage

Developers could use these recommendation to get more information about the features and their usage of a 3rd part API, for example official pages, documentations, and examples.

Requirement D89 MAY Status: to do waiting for server side implementation

The IDE may provide an interface where description of a feature to be implemented and libraries that should be used to implement it can be given. So the user can ask for recommendations how to implement features using specific libraries.

API is prepared for sending context-data

Requirement D90 MAY Status: to do waiting for server side implementation

The IDE may provide an interface to show recommendations on how to implement a feature using some specific libraries.

API is prepared for sending context-data

3.3.2 Handling API changes

There several forum threads and change reports, which could ease the migration between different versions of the same library.

Requirement U165 SHOULD Status: to do waiting for server side implementation

CROSSMINER IDE offers a list of community discussion forums concerning the use of a changed third-party API element



Requirement U166 SHOULD Status: to do waiting for server side implementation

CROSSMINER IDE offers code examples for deprecated third-party API usage points



Requirement D87 SHALL Status: to do waiting for server side implementation

The IDE shall provide an interface that explains the steps of how to upgrade a library used in the project to a new version.



3.3.3 Inspecting Code Chunk Related Q&A Posts

Also there are some discussions about how to implement a function using a specific features or a set of libraries. The recommendations which present these documents (posts and blog entries) are represent an other subclass of text based recommendations.

Chapter 4

User Activity Monitoring

Requirement U176 SHALL Status: to do <i>There is a strict and public strategy regarding privacy and data</i>	 Not specific to IDE
Requirement U177 SHALL Status: to do waiting for server side implementation <i>Users cannot be identified from monitoring data</i>	 Not specific to IDE
Requirement U178 SHOULD Status: partially done <i>Monitoring is able to identify testing activities</i>	
Requirement U179 SHOULD Status: done <i>Monitoring is able to identify development activities</i>	
Requirement U180 SHOULD Status: ??? <i>Monitoring is able to identify errors in IDE</i>	 This needs to be clarified
Requirement U181 SHOULD Status: partially done <i>Monitoring records the time the developer works on a given file/code element/line</i>	
Requirement U218 SHALL Status: partially done <i>All metrics are documented, motivated, possibly with references</i>	
Requirement D96 SHALL Status: to do <i>The IDE shall recognize, compute, and extract the following user activities, metrics, or information: frequent search expression.</i>	 Details needs to be defined. Search expressions come only from searches
Requirement D97 SHOULD Status: partially done	 plug-in. Search patterns come only from searches in the plug-in.

The IDE should recognize, compute, and extract the following user activities, metrics, or information: project or file open, manipulation, close, program execution, test execution, user search patterns, working time on a file.

Not IDE! Needed
for U176-181, D96-
98

Requirement D108 **SHALL** **Status:** waiting for server side implementation
The CROSSMINER REST API shall provide an interface that accepts developer activity data. So the CROSSMINER can build this information in the knowledge base.

4.1 Retrieving Process Metrics via crossminer Web-based Dashboard

You are able to inspect any computed process metrics for your project by using the relevant features of CROSSMINER Web-based Dashboard.

4.2 Plug-in Side Debugging Features

In the case of unexpected errors during the user activity monitoring, you are able to check the value of the process metrics and some relevant meta-data about the underlying database on the client side. To do this please activate some of the plug-in side debugging features. Please note that these are only available in the debug version of the plug-in.

4.3 Server Side Debugging Features

There are ways to access the raw data, stored at the server side. To do this you have to execute the following REST API calls. You could use your preferred REST client, but for illustration purposes we will use Postman¹



Requirement D98 **SHALL** **Status:** to do

The IDE shall be able to send developer activity data (as controlled by the user settings) to the CROSSMINER server.

¹<https://www.getpostman.com/>

Chapter 5

Settings and Customization

Requirement U155 **SHALL** **Status:** to do
Eclipse users can easily deactivate the analysis

Not IDE!

5.1 Integration Related Settings

Requirement D74 **SHALL** **Status:** partially done

The IDE shall provide a settings interface to the user, where the different properties of the CROSSMINER IDE plugin (like server address and port, global settings for recommendation queries, etc.) can be checked and changed. So the user can configure the plugin.



5.2 Process Metric Related Settings

Requirement U182 **SHALL** **Status:** to do

Monitoring of developer activity can be disabled by the developer



Requirement U183 **SHALL** **Status:** to do

Types of data collected from monitoring are transparent to the developer



Requirement D95 **SHALL** **Status:** to do

The IDE shall provide full control over the collected and transferred user activity monitoring data. So the user can allow or deny the collection and/or anonymised transfer of the activity data collected from their session.

