Vanessa UD10 Importation Service

2020.04.13.ET

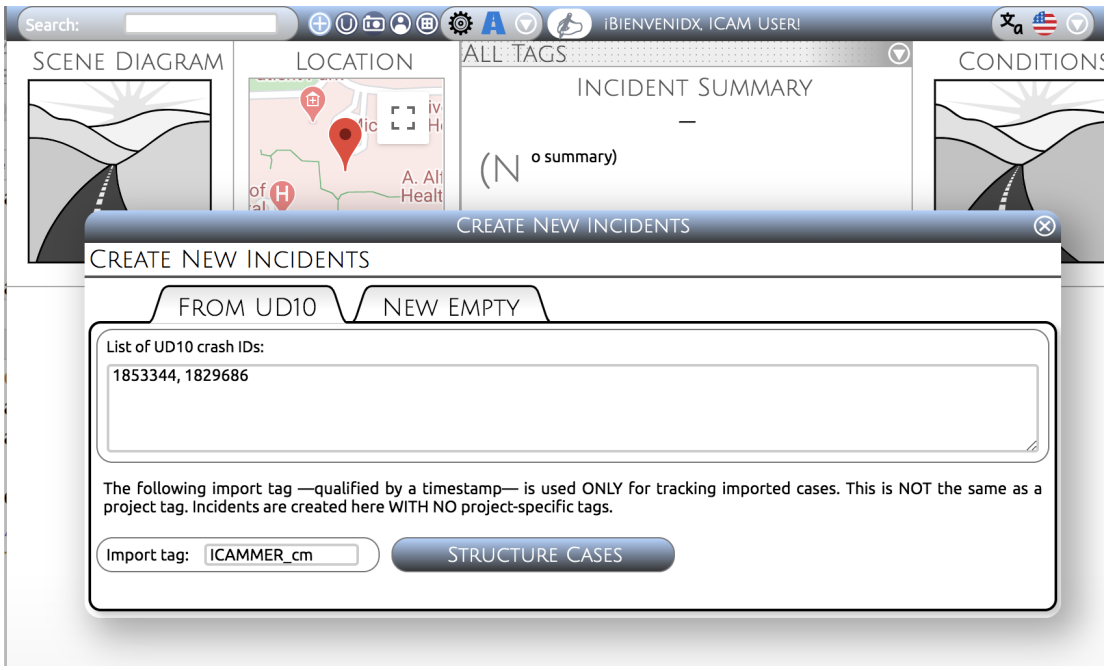
Introduction

The new UD10 importation service runs on a dedicated Node server. The service is still under active development. As a result, certain details of the service mentioned in this document may change in the near future. Nevertheless, the higher-level aspects of the service mentioned below are not likely to change significantly.

Using the service from within Vanessa

To access the service from within Vanessa, click on the circled plus sign next to the incident ID box:

A window will appear with two tabs. The first tab allows you to import and structure cases directly from UD10. The second tab preserves the legacy functionality which allows you to create a completely empty case. In this document, we will focus only on the first tab, *From UD10*:



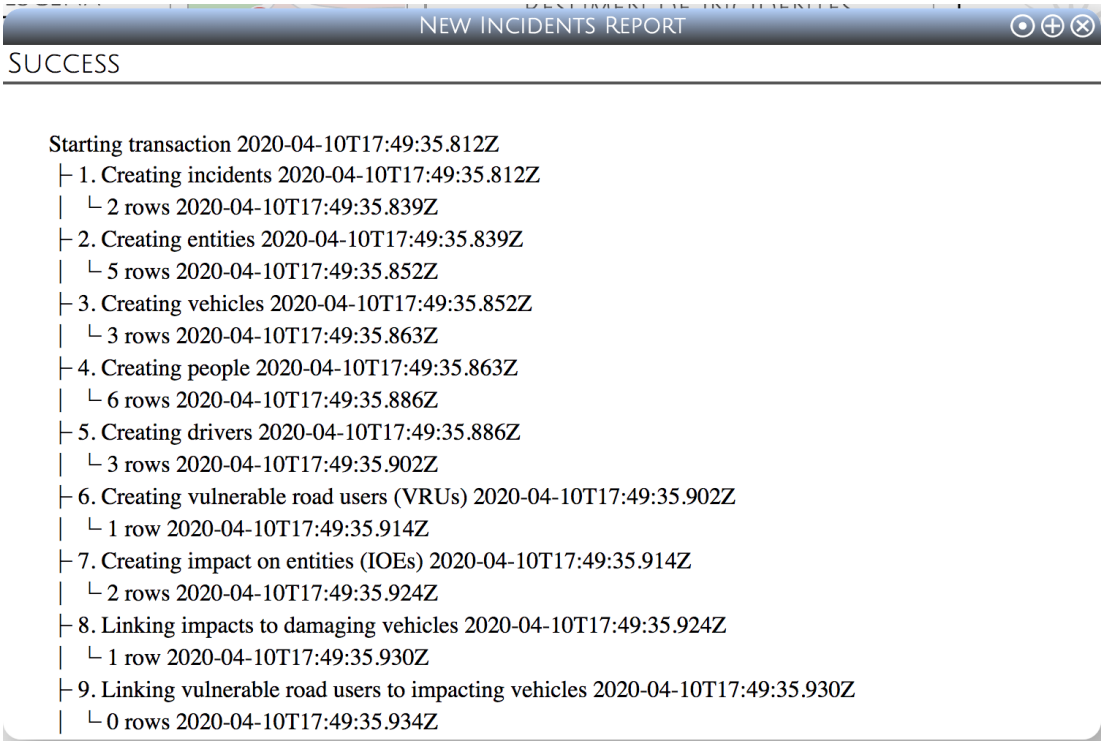
In the designated box, you may enter one or more UD10 crash IDs. If you enter more than one ID, be sure to separate the IDs with commas between them.

There also appears here a box labeled *Import tag* which will be filled in automatically with a tag constructed from your user ID along with a couple of random letters. In general, there is no reason to change this auto-generated tag. Any cases that you create will be tagged internally in the system with this tag so that we can trace the origin of imported incidents. Note that this tag is ***not*** **related to the project tags. The importation service has no knowledge of project tags.** ***Therefore, you will need to add any project tags (ICAM, ICAMB, ICAMP, PRDM, etc.) manually after you have imported incidents.***

Currently there is a limit which allows you to import ***up to 5 cases at a time*****.**

Results Report

Once you press the ***Structure Cases*** **button, you should expect to wait a few seconds for the import service to complete. If all goes well, you will next see a report similar to this:**

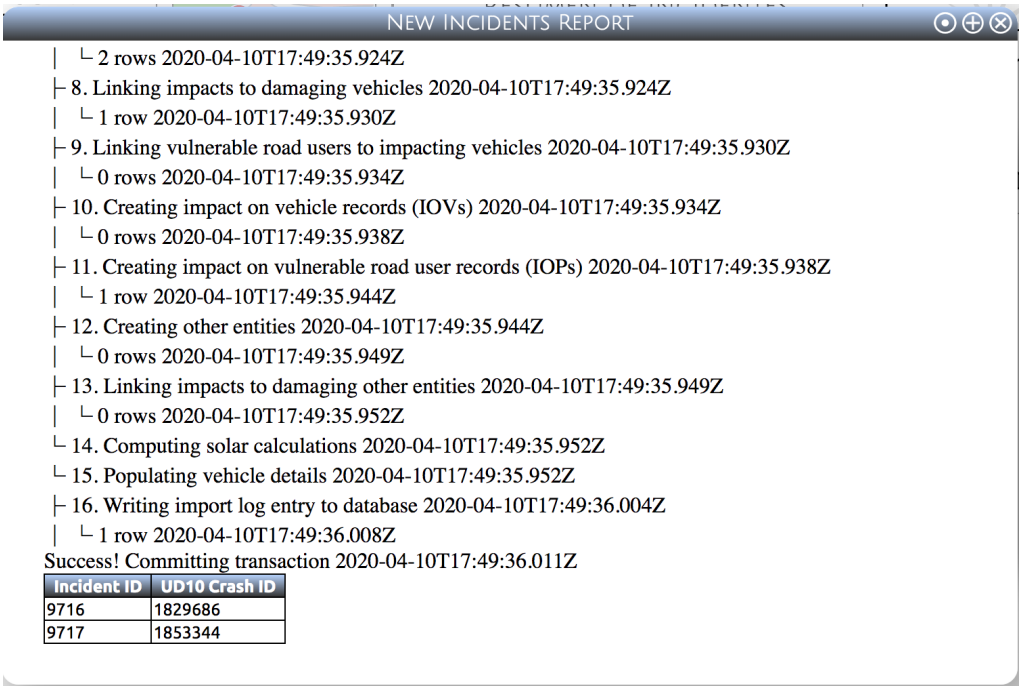


The report details all of the steps that the importer went through to structure the incidents for you, as well as how many records were created at each step. In the event of an error, an error message will appear after the step in which the error occurred.

In rare cases, the importer will succeed but some information may still be missing from the incident. In particular, it is possible to see incomplete vehicle information in the event that the NHTSA VIN decoder service is down or running slowly. As this is an external service, we have no control over it.

Incident assignments are at the end of the report

It is essential to scroll down to the end of the report, because here a table is produced showing the assignments of the new incident IDs to the UD10 cases that you requested:



Since the importer has the ability to import multiple incidents at one time, it does not automatically take you to the imported cases. Therefore it is critical to take note of the incident assignments in this table.

Technical details & features

(NOTE: Portions of the remaining documentation have been written with technical staff readers in mind).

The new Vanessa UD10 server runs as a standalone Node.js server. It currently is set to run on port 17443:

`https://icamlinux.surg.med.umich.edu:17443/`

While limited access to the service is provided in a user-friendly manner from directly within Vanessa, unlimited access for the purpose of importing scores or hundreds of incidents is only available to technical staff.

Parameters

Certain parameters must be provided to the service by the user for it to perform its function. These parameters are discussed below.

Importation Tag

Each importation event needs to be named using an importation tag. The server will further qualify the name that the user provides with a timestamp. For example, if the importation tag supplied by the user is "`test`", the server will append a timestamp such as "`2020-03-13T15:20:24.669Z`". In this case, the final tag will be "`test 2020-03-13T15:20:24.669Z`". When accessed from within Vanessa, the import tags are constructed from the current username.

This importation tag is written into a new *import_tracking_tag* on every new incident imported into the Vanessa database, making it trivial to find and do follow-up work on the set of imported cases.

List of UD10 IDs

The server also needs to be provided with a list of one or more UD10 crash IDs to import. A comma-separated list of UD10 crash IDs is required.

Process

The server runs in an automated, non-interactive, step-by-step manner. The entire set of UD10 crash IDs are processed as part of a single transaction on the Postgres database. Failure at any step of the process will result in a rollback of the entire transaction.

Progress Report

The server generates a report as it runs. Line items are added to the report at each step of the process. Currently, there are 16 steps in the process. If all goes well, at the end the user will see a report similar to the following (*fig. 1*):

```
Starting transaction                                          2020-03-13T15:20:24.669Z
 ├ 1. Creating incidents                                      2020-03-13T15:20:24.670Z
 |  └ 4 rows                                                  2020-03-13T15:20:24.830Z
 ├ 2. Creating entities                                       2020-03-13T15:20:24.830Z
 |  └ 7 rows                                                  2020-03-13T15:20:24.875Z
 ├ 3. Creating vehicles                                       2020-03-13T15:20:24.875Z
 |  └ 6 rows                                                  2020-03-13T15:20:24.890Z
 ├ 4. Creating people                                         2020-03-13T15:20:24.890Z
 |  └ 19 rows                                                 2020-03-13T15:20:25.009Z
 ├ 5. Creating drivers                                        2020-03-13T15:20:25.009Z
 |  └ 6 rows                                                  2020-03-13T15:20:25.141Z
 ├ 6. Creating vulnerable road users (VRUs)                   2020-03-13T15:20:25.141Z
 |  └ 0 rows                                                  2020-03-13T15:20:25.151Z
 ├ 7. Creating impact on entities (IOEs)                      2020-03-13T15:20:25.151Z
 |  └ 7 rows                                                  2020-03-13T15:20:25.205Z
 ├ 8. Linking impacts to damaging vehicles                    2020-03-13T15:20:25.205Z
 |  └ 5 rows                                                  2020-03-13T15:20:25.214Z
 ├ 9. Linking vulnerable road users to impacting vehicles     2020-03-13T15:20:25.214Z
 |  └ 0 rows                                                  2020-03-13T15:20:25.219Z
 ├ 10. Creating impact on vehicle records (IOVs)              2020-03-13T15:20:25.219Z
 |  └ 4 rows                                                  2020-03-13T15:20:25.232Z
 ├ 11. Creating impacts on vulnerable road users (IOPs)       2020-03-13T15:20:25.232Z
 |  └ 1 row                                                   2020-03-13T15:20:25.244Z
 ├ 12. Creating other entities                                2020-03-13T15:20:25.244Z
 |  └ 1 row                                                   2020-03-13T15:20:25.250Z
 ├ 13. Linking impacts to damaging other entities             2020-03-13T15:20:25.250Z
 |  └ 1 row                                                   2020-03-13T15:20:25.256Z
 ├ 14. Computing solar calculations                           2020-03-13T15:20:25.256Z
 ├ 15. Populating vehicle details                             2020-03-13T15:20:25.257Z
 ├ 16. Writing import log entry to database                   2020-03-13T15:20:25.331Z
 |  └ 1 row                                                   2020-03-13T15:20:25.331Z
Success! Committing transaction                              2020-03-13T15:20:25.367Z
```

Figure 1. Importation transaction report

In this case, all steps of the importation process were successful and the transaction was commited. The fact that "`0 rows`" were generated at a given step (*e.g, step 9 in the example shown here*) does not necessarily represent an error, although it might be an item worthy of manual review. For example, if we are importing a set of ICAM cases, we do not expect to have any VRUs in the importation set. The server however will always try running every step of its fixed program.

Calls to External Services

The server calls two external services, as follows:

In step 14, "`Computing solar calculations`", the importation server calls the external solar calculations service. This means that the external solar calculations service must also be running (*normally it is*).

In step 15, "`Populating vehicle details`", the importation server calls the external NHTSA VIN lookup service in order to populate the vehicles' *year, make, model, vehicle type, etc.* This step also fills in *vehicle_class* based on our own accumulated set of data.

Errors rollback the entire transaction

If an error occurs at any step, then the entire transaction will be aborted and rolled back. The report will appear truncated just after the step where the error occurred, and the report will conclude with a line telling you that an error occurred and that the transaction has been rolled back.

Mapping Table

The final section of the report consists of a table showing the mapping of the UD10 crash IDs to the new Vanessa incident IDs (*fig. 2*):

Incident ID	UD10 Crash ID
9673	1498083
9674	1671065
9675	8094073
9676	9770248

Figure 2. Table of incident IDs and UD10 crash IDs.

Example Results

The importation service brings as much information as possible from the UD10 database and structures it as completely as possible within Vanessa. Here are a few examples (*figs. 3,4,5*):

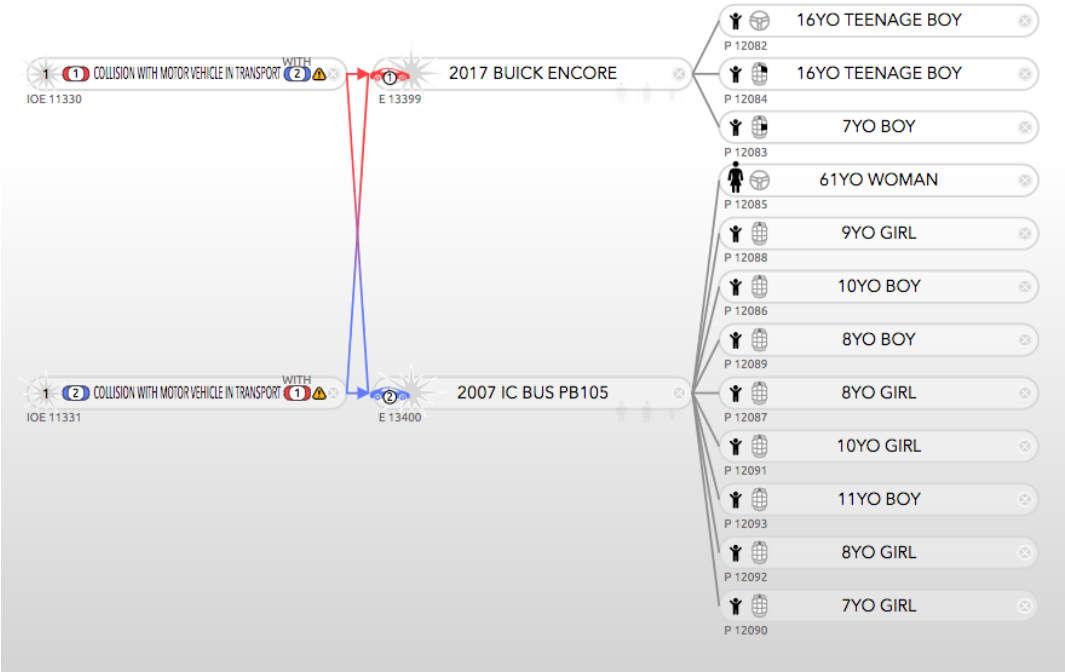


Figure 3. Example ICAM case.

In *fig. 3*, we see that in addition to importing all drivers and passengers, the service has also structured the collisions between the two vehicles.

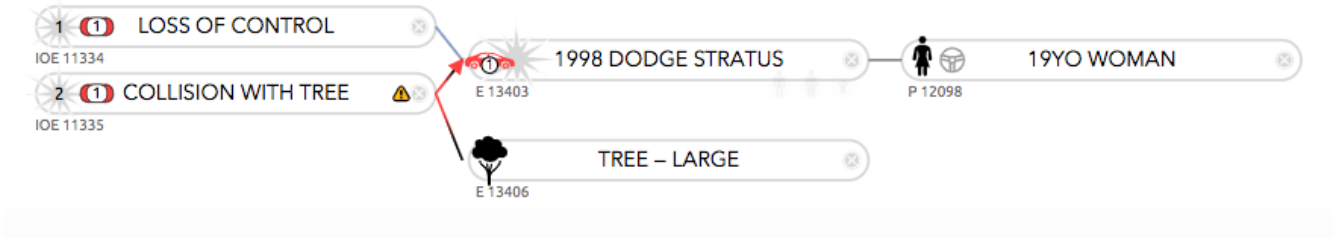


Figure 4. Vehicle hits a tree.

In *fig. 4*, we see that the importer properly structures both *non-collision* and *collision* events and, in the case of impacts with "other" objects, also properly creates and links the other objects (*the tree in this case*).

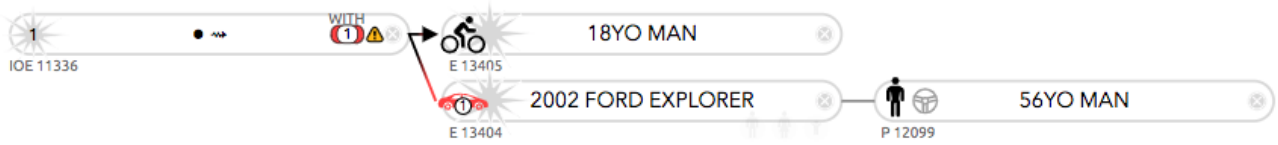


Figure 5. A vulnerable road user (VRU VIPA) case.

In *fig. 5*, we see that the importer has correctly structured a vulnerable road user (VRU VIPA) case.

Time Savings

The importation service is quite capable and should prove very helpful in reducing the amount of time that staff spend on structuring the basic aspects of incidents, allowing more time to concentrate on the detailed data that are not available from UD10.

Limitations on Case Structuring

However, the importer does have certain limitations. For the most part, these limitations result directly from limitations within the UD10 database itself.

The primary issue is that the UD10 database does not *explicitly* record what vehicle hit what other vehicle or vulnerable road user. Despite this limitation, we can *unambiguously infer* the interactions in a significant subset of cases. These subsets are treated below:

Two Vehicles

In the case of an incident with exactly two vehicles, it is clear that one vehicle hit the other vehicle, and so we can structure the reciprocal impact interactions between the two vehicles.

Three or more vehicles

However, if there are three or more vehicles in an incident, it is not possible to make assumptions about which vehicle hit what other vehicle. Therefore in these cases, the importer will not link the vehicle impacts. Users will see “hanging” red dotted-line arrows with question marks. The interactions in these cases need to be structured manually.

One vehicle and one or more Vulnerable Road User

In incidents with exactly one vehicle and one or more vulnerable road users (VRUs), it is clear that the vehicle hit the VRUs and so the importer can unambiguously structure the case.

Incidents with more than one vehicle and one or more Vulnerable Road User

However, if there is more than one vehicle in a VRU case, then again it is not possible to automatically structure the interactions. Therefore in these cases, users will likewise see “hanging” red dotted-line arrows with question marks indicating unresolved impacts. The interactions will need to be structured manually.

**** end of document ****