Rigshospitalet

PET & Cyklotronenheden

Technical documentation and risk assessment for Tracershop

Christoffer Vilstrup Jensen

Resume

This document describes the current Tracershop ecosystem, a bookkeeping system for the production of radioactive tracers for clinical procedures produced at Rigshospitalet. The system was designed in 2004 and in the field of Computer Science such a system is considered ancient. This document presents a replacement system with the aims to improve the user experience, functionality, and improve IT-security, namely a react-django new web server and makes a risk assessment of the current and the new system.

Tracershop - Current system

The original Tracershop was originally produced in 2004. Various contributions to the ecosystem have been made over the years, however development the system stopped around 2012. The main functionality of Tracershop is fulfilling the documentations requirements in Rigshospitalets production of radioactive tracers specified by GMP. Rigshospitalet delivers radioactive tracers to many hospitals and scientific institutions, that would not be able to perform various PET and SPECT scans without the tracer. Therefore Tracershop should be considered a important piece of software.
The system is centered around a MySQL 5.1 database running on a openSUSE 11.2 distribution, which contains all the records and logs about the production of tracers. The user interface is a Zope 2-2.13 web interface allowing the users interface with the main database. It is normal that software packages deprecated after a period of support, and keeping a software product up to date is part of the maintenance of a software products life cycle, however that have not been possible due to insufficient resources.
An overview of the system can be seen in figure 1. The dispenser ensures the amount of product in vials and

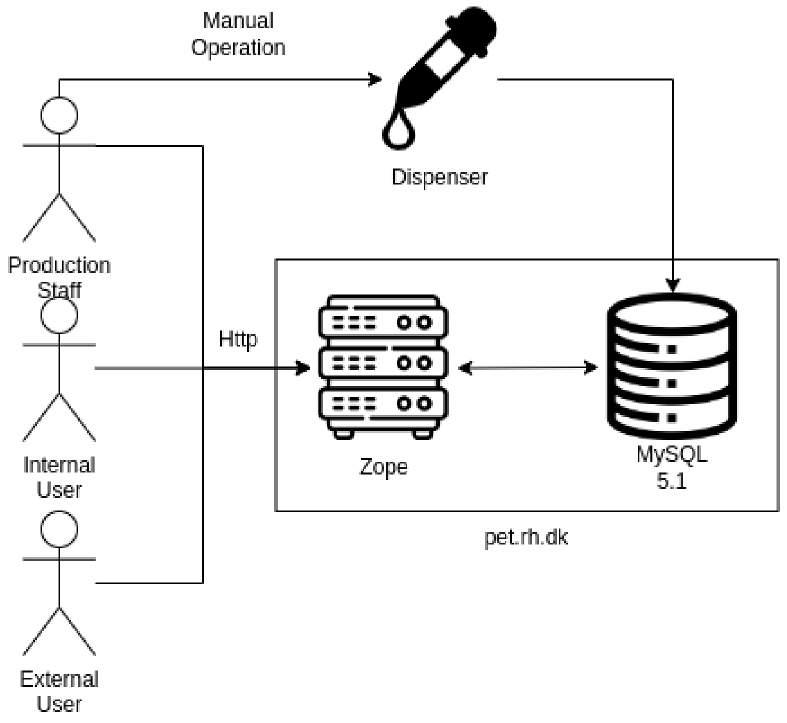

Figure 1: Tracershop system overview

the digital record of the product are equal. The dispenser writes these records to it own database, which does not support multiple concurrent connections, making it impossible to retrieve them the conventional way. A script circumvents this obstacle by opening the database as a file, and by using a byte offset retrieves the data of the dispensed vial. This solution is highly fragile, and an update to the database could change the file format of the database. This would render the script inoperable as the byte offset would no longer be correct.
Should the offset be incorrect the script would read garbage, so it could not construct a valid entry and thus would mostly likely fail. In that event the production staff would have to manually read the data from the dispenser and then write the data into Tracershops database, which introduces another human error.
It's therefore recommended to replace the dispenser, with one fulfils the requirements of either:

- Writes dispensed tracer data to a database that allows multiple connections.

- Writes dispensed tracer data to a file.

Currently it's the The Department of clinical physiology nuclear medicine which hosts the server. Which is not desired because hosting server is a core task CIMT and not the clinical departments.

Abstractions

Tracershop handles two different types of orders. Activity based orders and Injection based orders.
An **activity order** is defined as an order, where the user orders an amount of MBq radioactive tracer at a predetermined time slot known as a "deliver time". It's the user responsibility to account for radioactive decay between injection time and delivery time.
An **injection order** is an order with a number of injections with a predefined amount of activity, and it's Tracershops responsibility to account for any decay between production time and injection time. Users er limited per user basis which injection tracers they can order.

Requirements

- A user can order radioactive tracer from the internet while not connected to the Capital Region intranet. A user can be limited in their selection of tracers and limited time window where such tracers are available for ordering.
- A user can review any order they have made, and view batch number of tracer that they have ordered. They cannot view or alter order, which doesn't belong to them. They cannot alter delivered orders.
- A released order has a batch number and a record of who released it.
- Non authenticated users cannot alter or view information in tracershop.

Database layout

The Tables in the Tracershop main database, note the tables casing does not conform to the snake casing that standard for SQL databases:

1. Log - Likely Zope related. No longer in use as the last entry was in 2010-03-18.
2. MiscData - Likely Zope related. Likely a temporary value container.
3. Roles - Zope related, defines different user roles in the Tracershop program.
4. Sessions - Likely Zope related, defines active user sessions.
5. Tokens - Likely Zope related, Likely defines the length of active sessions.
6. TracerCustomer - Defines which user have access to order which injection Tracer.
7. Tracers - Catalog of tracer available in tracershop.
8. UserRoles - Relates user to a Role from the Role table.
9. Users - All users able to be authenticated in tracershop
10. VAL - Record of a vial with tracer, produced by the dispenser.
11. VAL2 - Not in use.
12. blockDeliverDate - Extraordinary dates where tracershop is closed, such a holydays.
13. deliverTimes - Weekly points in time specifying when a customer can place an activity order.
14. isotopes - Catalog of radioactive isotope used in the tracers.
15. orders - List of activity orders.
16. productionTimes - Weekly points in time, where a production should happen. Entries are also referred as a run.
17. productions - Production of tracers.
18. storage - Old mails, assumed not in use.

19. t_orders - List of injection orders.

The database are not utilizing the foreign key restriction, meaning that the relations are ensured at application level and not the database level. It's recommend to utilize the these functionality to ensure that the database stays in a valid state. For instance if a tracer entry is dropped or updated, it may not be possible to determine the tracer of any historic order, that might have used this tracer.
Despite Activity orders having a tracer field, this is ignored by the Zope application and assumed to be FDG.

New web server

Tracershop is a system of individual components, so a simple switch of system is not advised, as any of new components would likely contain early adopter errors. While these errors likely are easy to resolve, they would still take the system down, which is unacceptable. The best plan of action would be replace each individual component one at a time. If possible the old component should be running as a backup in the event that a new component has an error and is unavailable.
The largest but easiest to replace component is the web server, as it can connect to the old MySQL database and therefore can co-exists with the old system, as long as the old and the new web server follows the same assumptions. Once the new tracershop is considered stable, the Zope web service can be discarded.
This report suggest a django web server with a react front end. The project is open source found at:
https://github.com/demiguard/Tracershop
The new system uses common software packages for web servers:

- React - Frontend Javascript library developed by facebook. https://reactjs.org
- Django - Backend Python web server library https://djangoproject.com
- Channels - Django extension for using websockets. https://channels.readthedocs.io./en/stable

These packages are well supported and all in active development, so their developers will continue to provide security updates.
The new Tracershop utilize websockets instead of http communication, because websockets allows the server to push updates to the users unprompted, because the protocol uses a persistance connection, while http does not use a persistent connection. An overview of the communication can be found in figure This means that user

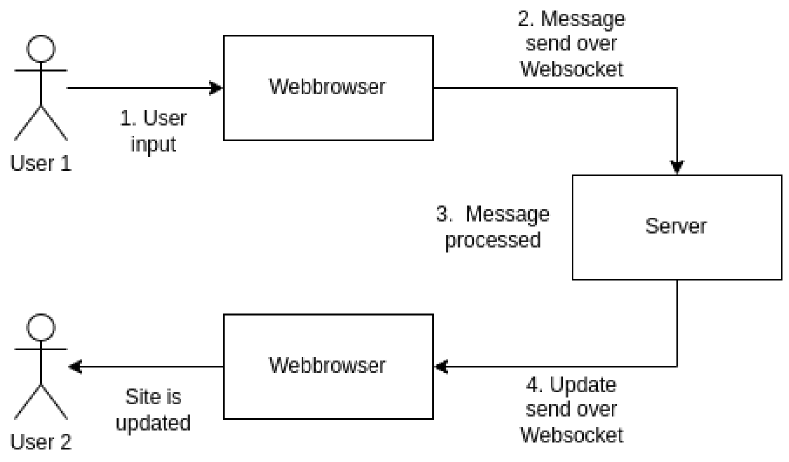

Figure 2: Overview of the message handling.

always view up to date data, while in old system users might operate on out of date data, if another user have altered that data.

The new system should be hosted hosted on CIMT servers, which would be in line with Rigshospitalet policies regarding software.

To ensure that the new tracershop handles message correctly behavior and follows requirements of GMP. The message is p in a three step plan seen below:

1. Validate the message - ensure that the message is correctly formatted and therefore would not cause an error if processed by the web server.

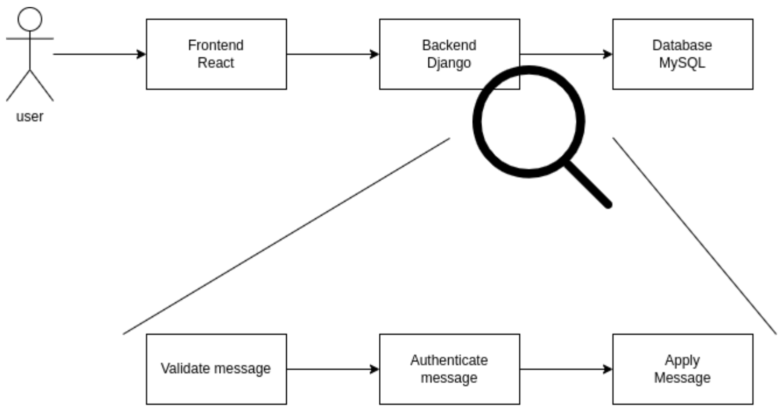

Figure 3: Message handling by the backend

2. Authenticate the message - Now the server can assume, the message is of a valid structure and determine it type. Then it can check if the authenticated user is allowed to complete such a message.
3. Apply the message - Now the message is both valid and authorized, so the server will handle the message.

Since the frontend doesn't have any contact with the database, it is sufficient to ensure that the backend handles the users messages correctly. This is done by a series of unit tests and end to end tests, which have a 92% code coverage. Which means that there's a program, which creates a test environment and run a number of tests where 92% of all statement in the program is executed at least once. Having above 90% code coverage is considered good coverage as not all code is easily testable, nor does a code coverage of 100% ensure that the program have no errors. The testing proves the web server with high confidence handles messages correctly.

Conclusion

The old tracershop fulfils the requirement set forwards by GMP, but is running on modules that have reached end of life.

A new system is proposed, that fulfils GMP, no longer uses, and improves various aspects of the tracershop system. An extensive test sweep ensure that the django web server handles messages correctly.

Glossary

CIMT Center for IT and Medico technologies. Responsible department for IT in the Capital Region of Denmark. 2, 3

FDG [^{18}F]Fluorodeoxyglucose, a common radioactive tracer used for PET images. 3

GMP Good manufacturing practice, created by the EU and Danish medicines agency.. 1, 3, 4