Zackary T. Morelli

zackmorelli@gmail.com

7/9/2021

The MD CarePath Schedule Conflict Check program is not an Eclipse Script or a standalone program that calls Eclipse, it is just a executable Windows program. It has a small WinForms GUI that is just a progress bar so that the user can see that it is running, but it doesn’t take any input from the user and its output comes in the form of a message box at the end of the program.

The purpose of this program is very specific; it finds conflicts between the Doctor’s schedules and Carepath tasks that they have. Specifically, we are interested in situations where a doctor has a carepath task assigned to them (a task like Contours Needed, or Plan Review) that is due on a day they are not available, AND they are not available the day before as well. These situations in particular gum-up the department’s workflow and potentially delay patients from starting on time.

There is a need for a program that can identify these conflicts, because even though the department’s secretaries spend a lot of time creating coverage schedules (for both locations in the department, Burlington and Peabody) for the doctor’s each month, they don’t reflect when the doctors have treatment planning related tasks due. In addition, even though we go to the trouble of creating appointments in Aria to reflect the doctors schedules, Aria is not capable of alerting users when there is a conflict between the doctors schedules and their assigned tasks. Aria is not designed to do that. Aria’s appointment scheduling is designed to schedule patients. We make appointments to reflect the doctors schedules so we can look them up easily in Aria, but Aria’s appointment scheduling is not meant to manage staff schedules.

I don’t know why, but ESAPI does not contain anything related to Aria’s scheduling software, either appointment scheduling or carepaths. I guess this is because ESAPI is really just for external beam planning with Eclipse, not the entire Aria suite. Anyway, although it is very convoluted and spread across a number of different tables, all the appointment information in Aria is stored in its database. Aria’s database is a standard Microsoft SQL Server database, so in order to get information out of it, it is just a matter of using .NET’s native SQL libraries and making the connection string. I did try using Entity Framework to make an entity out of the database, but unfortunately it simply refused to include some very important, apparently unique, view-tables that are absolutely necessary, whatever I did. So, Entity Framework doesn’t work with Varian’s database. They are also complex databases built by someone else, so using Entity Framework with it isn’t really y the best idea in the first place.

Anyway, in order to make a program that finds conflicts between the doctors schedules and their assigned carepath tasks, it is necessary to make SQL queries of the database to get the scheduling information. I’m not going to explain how to make SQL queries here. You can look it up if you want to know more, but my code should help you. The important thing to be aware of is, because this program uses a .NET library to make SQL queries, the actual SQL code is written as a string in the C# code that is then given to some object of the .NET SQL library so it can use it to make a query. So, if a mistake is made in the SQL syntax (because SQL is essentially a distinct language made for database querying that has its own syntax), Visual Studio won’t know, because it is being written as a string. This can make it pretty difficult.

I originally made this program back when I had direct access to the Aria database to make SQL queries. Specifically, we had a SQL user which I hardcoded the credentials for into the program so that it could work for everyone. Unfortunately, we are no longer able to have a SQL user account like that, so it doesn’t work for everyone. I have database access through my personal login, and I made modifications to this program later so that it works for people who have access, through Active Directory. So, if you want this program to work for you, you’ll have to get Varian to make a SQL account for you on the Aria database, since Varian is the System administrator, not Lahey IT.

The program basically works like this. A bunch of SQL queries are done to construct a list of all the doctor appointments. Another series of SQL queries is made to construct a list of all the doctor-related carepath tasks. We then loop through them to find date conflicts. When I say “doctor appointments”, I mean the appointments that the secretaries have made in Eclipse to reflect the schedules they have made for the doctors. These are all-day long appointments that have a doctor attached to the appointment as a resource. They are just meant as a way for people to know where the doctors are by looking at the Aria schedule, but we can use them for the purposes of this program. These “doctor appointments” come in three categories: Contract Off, Not at this site, and Vacation. Contract Off is a day that the doctor isn’t working at all and is not available, similar to vacation where the doctor is on vacation and not available. “Not at this site” means the doctor is working at some clinic somewhere that day, but at a non-Lahey location.

One last important thing. It is very useful to have Microsoft SQL Server management studio (SSMS) so that you can peruse the database and experiment with SQL. That is how I figured out a lot of the stuff for this program.