Recorder6 Parallel/Successor System - Feasibility Project

Concept

Develop a prototype system to prove the feasibility of developing a new user interface system for the existing Recorder6 SQL Server database that will ultimately become a replacement or parallel system to Recorder6.

Requirements

If possible the prototype would meet the all of following requirements:

- Represent a commonly-used and clearly-defined area of user functionality (e.g. data entry via a species recording card).
- All source components to be open source.
- Developed without any dependency on proprietary software.
- Run independent of any proprietary software for users.
- Continue to use currently supported and future versions of SQL Server or SQL Server Express for the database.
- Secure from unauthorised access or malicious attack.
- Will run on a standalone PC with a single user.
- Will support multiple users using standalone PCs, remote desktop servers or other virtual environments (e.g. Citrix).
- Run via any common web browser interface (e.g. Internet Explorer, Edge, Chrome, Firefox) or similar thin client.
- Support multiple operating systems (e.g. Microsoft Windows, Microsoft Server, Linux, Apple OS).
- Will interface with a SQL Server (or Express) executing on the same PC, on the local network, or on a remote network/cloud server.
- Be able to support different languages and regions (e.g. for users in Luxembourg and Wales) through internationalisation and localisation.
- Will be based on the existing R6 data model so that it can be connected to existing live Recorder6 installations and run alongside Recorder6 as an alternative interface.

Deliverables

The project will produce the following deliverables:

- A prototype system/installer.
- Instructions for installing and using the prototype system.
- All code and associated components uploaded to an online, publicly accessible management system (e.g. GitHub) with a suitable open-source licence (e.g. GPL v3).
- Accompanying documentation detailing the options, recommendations and feedback (as outline below).

Documentation

The documentation accompanying the prototype is just as important as the prototype system itself as it provides the R6 steering group and wider community with vital information, such as:

- What requirements were considered important for the development?
- What options were considered and selected/rejected?
- How the development of the prototype went and any lessons learnt?
- What are the recommendations for continuing with the development (or starting again)?

Development language and framework/platform

It is possible/likely that further development of a parallel/successor system will be undertaken by volunteer developers, so it is important to understand which programming language and development frameworks/platforms were considered and chosen to ensure that they are robust, readily available, free and likely to supported long into the future. The following should therefore be documented:

- The requirements and considerations appropriate for choosing a programming language and framework/platform suitable for developing this prototype.
- The options that were chosen and the reasons for their selection.
- The alternative options considered for development and the reasons for their rejection.

Target installation platform/method

It would be ideal if any parallel/successor system is available for multiple operating systems and remote/virtual environments so that it can reach the widest set of users. It is also crucial that it can be installed very easily so that (a) non-technical users are not dissuaded and (b) it will meet the typical standards and expectations of any host ICT departments. The following should therefore be documented:

- The requirements and considerations for choosing the target installation method(s) for the prototype system.
- The options that were chosen and the reasons for their selection.
- The alternative installation options considered and the reasons for their rejection.

Limitations and dependencies

It is crucial that any limitations or dependencies of the prototype system and/or the chosen development framework/platform are clearly documented, including (but not restricted to):

- Any requirements that are not met with the prototype system, or cannot be met in a full parallel/successor system.
- Any software, systems, configuration or infrastructure settings that the prototype system is dependent on for users.
- Any limitations or dependencies of the development framework/platform.

Lessons learnt

Any lessons learnt from developing the prototype will be very valuable for both assisting the steering group to make a decision on further development, and for any potential developers interested in getting involved. Relevant lessons might include:

- Problems and pitfalls experienced/resolved.
- Software requirements/recommendations.
- Developer knowledge requirements/recommendations.
- Other knowledge requirements/recommendations (e.g. use of GitHub).

- The ease of installation, configuration of the development framework/platform.
- The ease of development and testing.
- The actual cost (time or money) that it took to complete the prototype and installation/user instructions.

Recommendations

Following completion of the prototype system what are the recommendations for further development, e.g.:

- Is the choice of language and framework/platform still appropriate/recommended?
- Any recommendations for how to approach future design, coding, testing, source management, issue logging/resolution, etc.?
- Are there any areas of R6 functionality that are likely to be easier than others?
- Are there any areas of R6 functionality that are likely to be very difficult/impossible because of the complexity of the functionality or because of the complexity or limitations of the chosen language or framework/platform?