Simulator based research involves the participation of human subjects whose behavior is tracked and measured in the course of several exposures to driving situations. The number of research participants can vary widely, but it is not uncommon to utilize a number of participants large enough to make manual accounting and tracking of the associated data problematic. Typically, information that must be tracked for each participant involves electronic data files, video files, questionnaires, but may also involve physical artifacts such as signed consent forms, video tapes etc. In addition, it is necessary to manage the software configurations that must be used for exposing each participant to the appropriate simulation scenarios. In organizations that conduct multiple studies, this problem is exasperated because of the need to maintain and track data for numerous subjects from multiple studies. This paper overviews a comprehensive approach for managing the problem of subject accounting, proper configuration management and result tracking across an organization conducting multiple studies on multiple simulator devices. The key aspect of the solution is using a centralized database maintaining all information about research participants and their linkages to all pre-study and post-study artifacts. Linking all operations through this database ensures appropriate accounting and minimizes the likelihood of human error.
INTRODUCTION

By definition, simulator-based research involves the participation of human subjects in a closed loop virtual driving environment. Maintaining control of certain aspects of the virtual driving environment allows investigation of the effects of other parameters on the driver or the driver’s behavior. The number of research participants can vary widely, but it is not uncommon to utilize a number of participants large enough to make manual accounting and tracking of the associated data burdensome. Typically, information that must be tracked for each participant involves electronic data files, video files, questionnaires, but may also involve physical artifacts such as signed consent forms, video tapes etc. In addition, it is necessary to manage the software configurations that must be used for exposing each participant to the appropriate simulation scenarios. In organizations that conduct multiple studies, this problem is exacerbated because of the need to simultaneously maintain and track data for numerous subjects from multiple studies.

The problem is similar to traditional software configuration management, where repeatability of builds is important not only for maintaining an effective workflow, but also for being able to re-create a sequence of events when necessary to address questions as to the origin of unexpected data or results. As in most disciplines, avoiding human error is critical when dealing with complex and voluminous information. We believe that an effective solution must involve centralized maintenance of all information associated with a participant, combined with software that consults the centralized database for obtaining the necessary associations. Automation is also necessary to avoid human error. It is human nature to create redundant representations of the same information, primarily for convenience but also to prevent loss. Whereas redundant storage of information may be necessary to avoid data loss, redundancy in the sources of authoritative answers about data associations can create confusions and errors when the information becomes uncorrelated.

This paper overviews a comprehensive approach for managing the problem of subject accounting, configuration management and result tracking across an organization conducting multiple studies on multiple simulator devices. The key aspect of the solution is using a centralized database maintaining all information about research participants and their linkages to all pre-study and post-study artifacts, combined with software that automates any aspect of handling subject thus minimizing the possibility of human error. The database acts as the sole authoritative source of information about the associations and the data, thus eliminating the possibility of divergent sources of information.

The following describes some terms that are used in this paper. An experiment is the conceptual construct that encapsulates all information about a research effort that involves a simulator data collection. In the database, an experiment is comprised of one or more participants. The experiment is usually designed to expose each participant to one more scenarios. Each of these unique scenarios is associated with one or more runs that expose the participant to a particular treatment. When the participant drives the scenario, the instance of that run is referred to as a run.
instance. Data can be produced during a run, but there is other data associated with a given scenario or with a participant, independent of any simulator exposure.

![Figure 1 - NADS experiment structure.](image)

**MOTIVATION**

When conducting human based research, it is critical to associate all data with their source. In the NADS, like in most high fidelity simulators, this data is represented by numerous files each representing a different aspect of a research participant’s performance during exposure to multiple simulated driving conditions. More specifically, participating in a NADS study involves experiencing one or more simulator drives, each of which produces a variety of artifacts that must be maintained and linked to the participant. The following is a typical example of data associated with a single drive that a research participant experiences during their involvement in a study:

- The date and time of the drive
- The scene and scenario used for the experience
- The configuration of the software that was running during the drive
- Binary data files containing the real-time data collected by the simulator
- One or more questionnaires filled before, during or immediately after the drive
- One or more videos, each providing different views of the drive
- Additional files with data produced outside the simulator’s software (i.e., eye-tracking data, manually scored performance values etc.)

In addition, associated with each participant are a set of properties that are independent of how many times they drive, or their performance during the drive. These vary according to the study design but typically include:

- Gender
- Age
- Experience
We realize that not all simulators are alike, and the specific examples may either exceed or lack what is typical of other systems. However, we believe that whereas the details may vary, the need to provide multiple views to simulator data is universal to all systems. Furthermore, we believe that the approach presented here could be beneficial to others as long as the specific requirements are identified and addressed in similar ways.

Specifically at NADS, the methods and requirements for accessing the data following an experiment vary greatly depending on the reasons for the access. The following provide a list of reasons and the associated requirements imposed on the structure of the data.

**Software configuration management**

This information includes the confirmation management versions for both the hardware and software of the simulator. This information is necessary for multiple reasons. The most obvious, of course, is the need to recreate the same simulator environment if it becomes necessary to re-run the same, or a similar experiment in the future. Another reason is having access to the data files in order to better understand the behavior of the participants during the drive. Whereas software tools exist for managing software versions, hardware configuration and version control is admittedly harder to implement. We address this issue by base-lining, as much as necessary, the core performance characteristics of the system and maintain these associated with each study.

**Subject Classification and Disposition**

It is obvious that basic biographical information about the research participants is required during data analysis. However, we have found that it is critically important to have up-to-date access about the biographical data of participants that have successfully completed their participation at any given time during a study. This is necessary in order to assess the progress of the study and take action if the study goals are not met. For example, consider a study requiring that 60 novice, young, and older participants complete the study. As often happens, participants may not show up, may withdraw consent, or complete the drive but be excluded for other reasons. Unless an up-to-date status of completion is available, it is impossible to ensure that 60 participants in each group have completed. Finishing the study with 70 young but 50 older participants is not an acceptable outcome.

**Data Reduction and Processing**

Due to the automatic nature of producing binary data files, it is possible to end-up with excessive or redundant files. For example consider the example of a participant who completes the drives but is excluded from the study because they didn’t follow the protocol. Their data files will exist, but clearly should not be included in the analysis. The disposition information is necessary to allow the software or individual performing the data reduction to decide if a particular data file should be included or excluded. Similar decisions have to be made about all the data artifacts associated with each participant or their drives. Even in situations where a participant was not excluded, it may be necessary to analyze one a subset of the data files produced; maybe one of the drives was a practice and the practice data is not to be included in the study, or a particular analysis may require looking at the third drive for all participants. There is an almost infinite set of views one requires of the data.
Data Tracking

Another problem with having numerous sensitive data files is that becomes necessary to be able track where the original data and any copies are located. This functionality provides a single location for a user to track the location of any data file generated by the simulator. This information is critical when it becomes necessary to delete duplicate copies of data files.

Administrative

Another reason for needing detailed simulator accountability is in keeping track of hours that the simulator is operating. This is necessary for knowing when to perform periodic maintenance, account of simulator usage and improving organizational productivity.

SOLUTION APPROACH

Not surprisingly, our approach to addressing the aforementioned problems is to utilize a centralized database that maintains all information about studies, research participants, software & hardware configuration, and resultant data files. Using a database provides flexibility in that one can construct queries and obtain an arbitrary view of the data in order to meet any requirements that involve data relationships. However, the utilization of a database alone would not address the problems unless significant effort is placed in ensuring that the database is updated consistently. Thus, a key component of our solution is a software framework that provides a uniform means of accessing the central database and the adaptation of simulator tools into this framework.

Experiment Database

Figure 2 shows the design of the simulator experiment database. The blocks represent tables in the database and the arrows represent relationships between tables. Each table block specifies the fields that belong to that table. Fields with a \textbf{“PK”} next to them are primary keys. Fields with a \textbf{“FK#”} are linked keys to another table. Fields in bold are required and may not contain NULL values. The \textbf{“[List]”} marker indicates a field which contains an ordered list of unique auto-number values.
The experiments, scenarios, participants and runs tables contain information about the design of the experiment. The subsystems, cells, elements and cec_elem contain information about the data streams available for collection on the simulator. The run_instances and comments tables contain information about what and when was actually run on the simulator during an experiment. The eif and data_coll_elem_file tables specify initialization values for data streams and account for which data streams were actually collected during the experiment respectively. Finally, the sim_data_file, sim_data_file_activity, sim_data_file_activity_codes and sim_data_file_type provide data tracking capabilities. The following sub-sections provide examples of how various tools interface to the database and maintain a consistent, yet updateable, view of the experiment.
To facilitate tracking of data files, the “runinst_id” field, in the run_instances table, is used as a universal naming component of all data files generated during a run. Standard prefixes differentiate the files based on their type. This provides an implicit link between a run_instance and all data files produced during that run. The run_instance is generated automatically and is a timestamp that, in addition to being unique, also provides a reference to when the collection took place.

**Experiment Builder Tool**

The first software component is responsible for updating the database when adding new data that acts as the root of upcoming dependencies. Such data includes an experiment, a research participant, or demographic data about a participant. This data, whose origin is the experimental design associated with a given study, takes advantage of general patterns that are inherently present in experimental designs thus allowing the software to somewhat automate their generation.

Figure 3 shows which tables the Experiment Builder tool reads and writes information from in the database. The tool assumes that somebody has already added an entry for the experiment itself. The tool checks to see if any scenarios, participants, run and run instances have already been to the experiment and reads those into the interface. From that point, the user is free to add more scenarios, participants and runs to the experiment as shown by the interface in Figure 4. The user may remove runs as long as they don’t have any run instances associated with them. Participants may be removed if they don’t have runs associated with them.

![Figure 3 - Experiment Builder tool and its interaction with the database.](image-url)
Figure 4 displays the interface for adding subjects and runs. This dialog is a part of the Experiment Builder presented in Figure 5. This particular snapshot carries forward the sample experiment presented in Figure 1. This dialog window is divided into 3 sections: a section for defining and adding subjects, a section for defining and adding runs, and a section for associating a scenario with a run.

Each subject name may be entered manually or a set of subject names may be automatically generated by defining a pattern. A pattern may consist of up to 5 different kinds of permutations which gives the user a lot of flexibility in creating names for subjects. Each permutation may either be a set or a series. This feature becomes more useful as this tool is used in experiments with a large number of subjects. The test permutation feature provides a user a snapshot of what the subject names will look like and how many subjects will be created as result of the pattern.
and permutations specified. The run definition is very similar to subject definition. The user may either enter runs manually or by specifying a pattern which may consists of up to 3 different permutations. The “%s” represents the subject’s name in the run name pattern. Finally, each run has to be associated with a scenario that will be executed during the simulation.

Thus a pattern of “Subject%1_%2” with permutations of “1…3” and “N,A” results in 6 subject names being generated as shown in Figure 4 and Figure 5.

**Simulation Operation (Simop) Console**

The second piece of software is responsible for automatically updating the database when dependent data becomes available. Automation ensures that no human is responsible for entering that information, and furthermore ensures the elimination of clerical errors. Note that achieving that level of functionality required that all software involved in the production of data files, or management of research artifacts had to be modified to interface with the centralized database. However, the result is that every time a data file is created, a corresponding entry is also created and appropriately referenced in the database.

The Simop Console is the primary tool used to control the simulator. This tool features a live connection to the experiments database. In order to run a simulation, an existing experiment and an associated run must be selected on the Simop Console. Using the live connection, the Simop Console is able to extract any needed information. The SimopConsole produces a run instance and associates with the selected run.
Figure 6 demonstrates an example where the run “Subj1_N_BRAKE” has been selected from the sample experiment that we’ve shown earlier in this paper. As the simulator is run, the Simop Console produces the run instance “Sub1_N_BRAKE-20060210130125”. The name of this run instance implies that the run was made on February 10th, 2006 at 1:01:25pm.

The software framework presented in this paper has tremendous potential for automation. The fact that data is stored in the form of a relational database automatically guarantees efficient data storage and quick access to that data. Simple queries can be constructed to return information about common day-to-day operation of the simulator such as current study completion status and data reduction status. As any organization changes or grows, so does the need for information. New queries can be easily constructed to provide new information and metrics as they become needed.

**CONCLUSION**

The daily operation of a typical simulator involves a lot of data that used both to configure and run the simulator along with data that is generated as a result of running the simulator. Such is the case at the NADS and the database presented in this paper was designed to address these concerns. This centralized database encapsulates a comprehensive approach for managing the problem of subject accounting, proper configuration management and result.