# Fundamentals of Data Synchronization

Installations of DDMS can share data with each other using the data synchronization feature. No connection to the Internet or other network is required. Data is exchanged by the sharing of data files between installations. These data files can be distributed on a compact disk, a flash drive, or any other kind of data media. Each installation can produce a data file that contains all records that were created or updated on that installation. Any other installation that imports this data file will then contain these same records.

Each installation of DDMS tracks all of its transactions, which can then be sent to other installations using the synchronization feature. Examples of tracked transactions include data entry through the web interface, excel imports, and synchronization data imported from other installations. Each transaction record is a self-contained unit, containing all information required to duplicate itself on other computers. Since records are created locally when importing transaction data from other installation, multiple updates can be easily combined into a single export file. For example, if computer A exports data to computer B, and computer B then exports its data to computer C, C will get the data of both A and B.

For example: A village exists within a district, which exists within a province, which exists within a country. The village installation exports its data to the district installation. The district installation contains its data plus the data from the village. The district then exports its data to the province installation. The province installation now contains its data plus the data from the district and the village, even thought the province did not import the data file from the village. The data from the village was contained within the export file from the district. Likewise when the province installation exports its data to the country installation, the country installation will contain all of the data from the province, district, and village. Any installation that the country installation exports its data to will also contain the data from the village, district, and province. Eventually, data from each installation will migrate to every other installation in the country.

**Master and Slave Installations**  
All DDMS installations fall into one of two categories: master or slave. In order to avoid unrecoverable synchronization conflicts, certain actions, such as modification of the Geo Hierarchy, are permitted only on master installations. Additionally, data cannot be shared between two master installations, so it is critical that each deployment contains only one master. No more than one master installation may exist within any given country. Otherwise, non-recoverable errors can occur during the synchronization process. Such errors will permanently prevent an installation from participating in synchronization again.

**Export Sequence Numbers**

A transaction is an atomic set of creates, updates, and deletes on one or more objects that collectively implement an action in DDMS. Data is exported in segments of transactions. When a transaction is exported it is assigned an export sequence number. This number is used to ensure that data is imported in the correct order and complete. That is to say, the import node has all of the required data from the export node for import data to not be corrupt. When a node is importing data it cannot have any gaps in the export sequence number. If there is a gap in the sequence number then the import node must contact the export node and retrieve the transactions for its missing sequence numbers.

Looking at the previous example where a village exists within a district, which exists within a province, which exists within a country. The village installation exports its data to the district installation. This export contains the transactions one, two, and three. Later the district installation gets another export from the village. However, this new export contains the transactions five, six, and seven. The district installation will not be able to import the village's second export until the district gets the village's export data for transaction four.

Site Masters

During the install process each install is provided a unique identifier. This identifier is used to associate all data with the install which was responsible for creating it. It is extremely important that duplicates of the install identifier do not exist. If duplicates exist then DDMS will treat data coming from the duplicate nodes as if it was produced by the same node. In such a situation synchronization of data will behave incorrectly and likely become impossible. Additionally, in order to prevent a large number of data conflicts DDMS restricts modification of data to its original install. For example, a spray team created at the village install cannot be modified by the district install, it can only be edited at the village install. However, the synchronization resolver allows the user to by-pass this restriction. As such, in a worst case scenario using the synchronization resolver the village install would be able to modify the district install. Note that when an install modifies data from a different install it can cause a divergence of data when the data is propagated to other installs.

**Data Export**

**Data Import**

# Basic CRUD operations

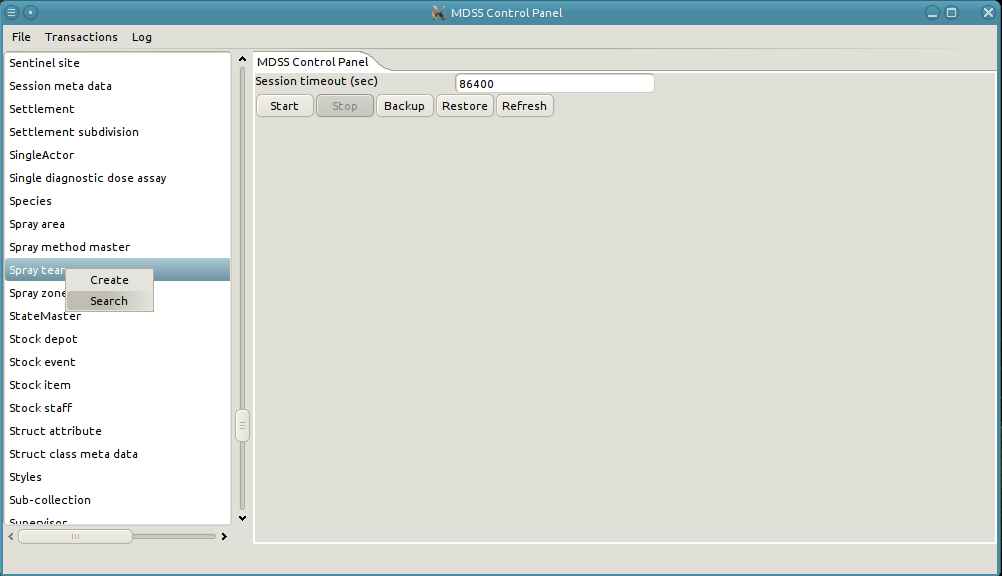
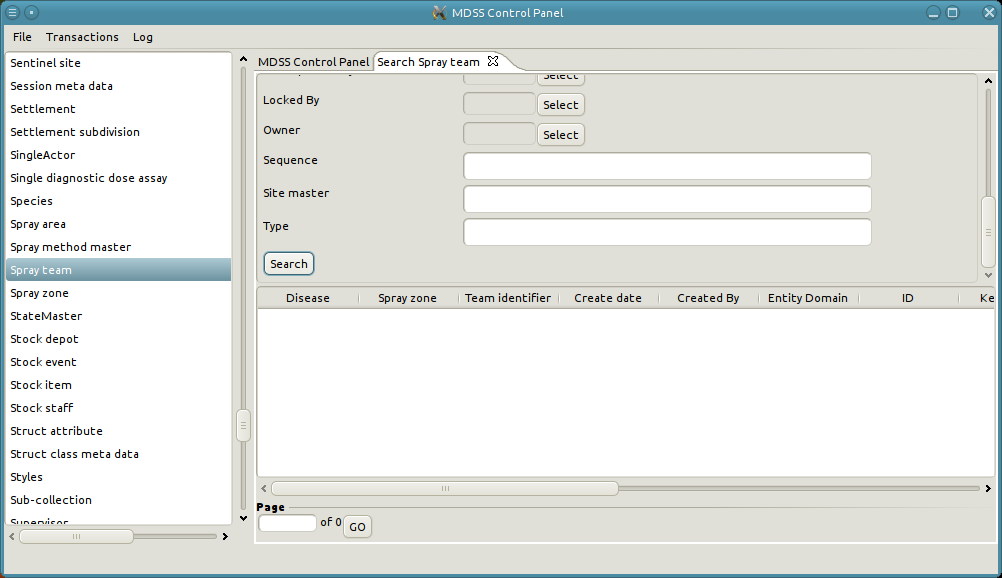
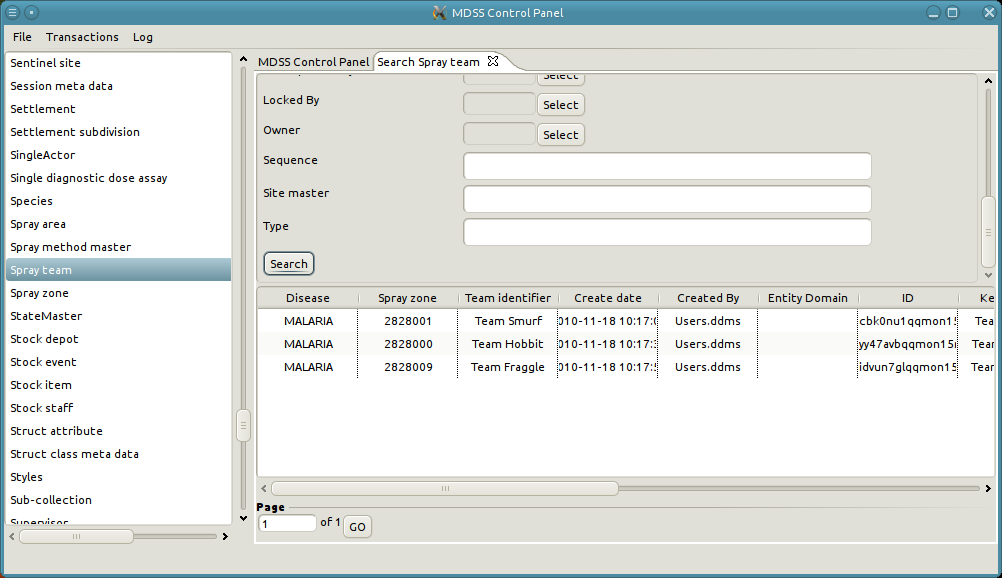
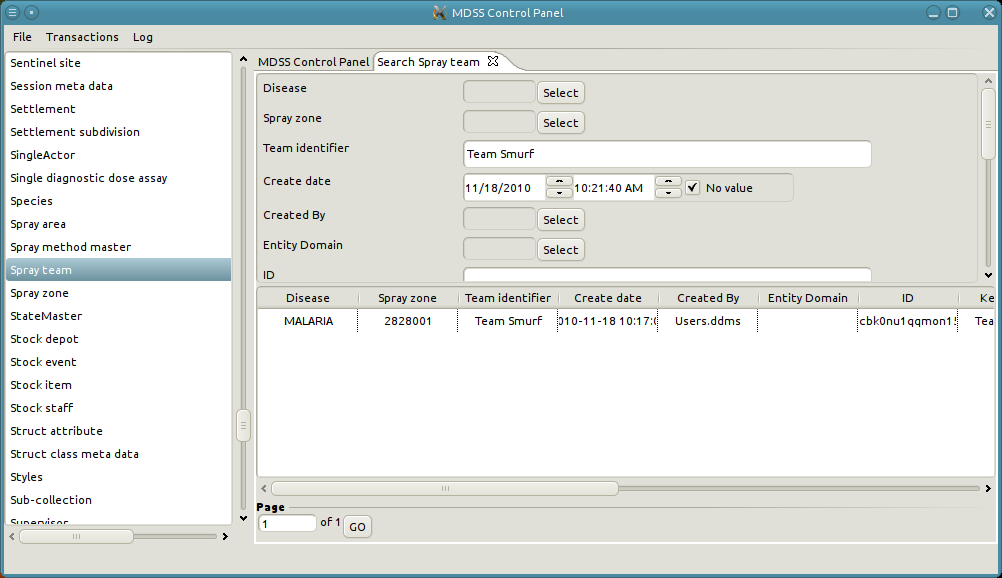
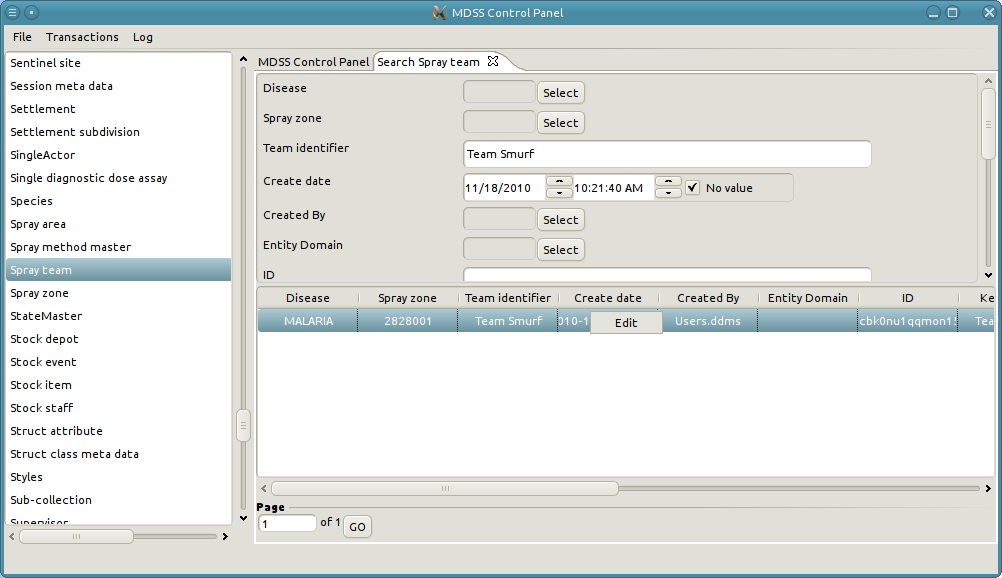
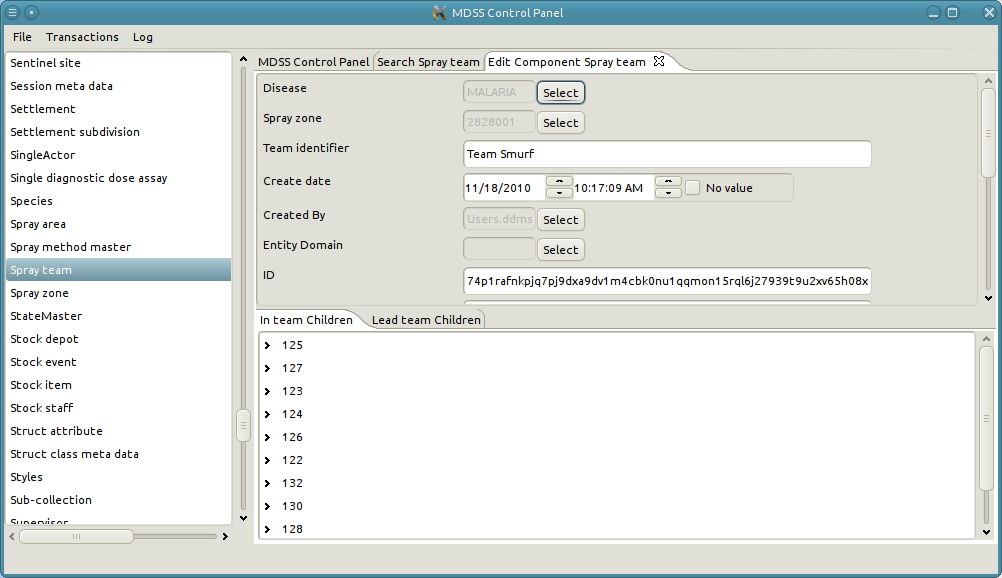
The synchronization resolver allows the user to view, create, and delete data in the system. These three operations enable the resolution of any conflicts raised during synchronization. However, since the synchronization resolver directly modifies objects in the database, it presents views of the database model, which differ from views seen through the web interface. As such, the resolver requires users with a high degree of technical proficiency and familiarity with DDMS.

## View existing data

The basic use case for viewing data in the system is as follows:

* The left panel lists all data types in the system. Note the list of data types does not include relationship types. Viewing relationship types is covered later in the manual. Double click the data type to search for records of that type. Additionally, it is possible to or right-click on the data type and select "Search". A new tab opens with a search form for the selected type
* Fill in search criteria as needed
* Click "Search" at the bottom of the form. The results do not auto-update. Thus as new criteria is added to the form, you must re-click on "Search" before new results will appear.
* Search results appear in the paginated table below the form. The results can be sorted on non-reference attributes by clicking the column header.
* [Optional] Access additional pages of search results with the input field below the results table
* Double click a row of the result table to view the object

## Example of searching for and viewing a spray team

1. Select Spray Team from the list of types to search. 
2. A new tab is opened with the search form for spray teams.
3. Click on search with an empty form to see all of the spray teams in the system. If the results have more than a single page of results more results can be displayed by changing the page number and clicking on "Go".
4. We can restrict the results by adding additional criteria to the search form. For instance, restricting the Team identifier to "Team Smurf"
5. We can view the details of "Team Smurf" by right-clicking on its row and select edit or by double-clicking on its row.
6. This will bring up a new tab with the details of the spray team.

## Relationships

A relationship is an associate between two pieces of data. For instance, a Spray team has many Spray operators. In an abstract sense, all relationships have a direction associated with them. As such the ends of a relationship can be thought of as source and sink nodes, or parent and child nodes. In the example where a Spray team has many Spray operators, the source/parent node would be the Spray team and the sink/child node would be the spray operator.

Similar to objects, relationships can have their own attributes. As an example, we may want to track when a Spray operator was added to a Spray team. This can be accomplished by adding an attribute to persist the creation date of the relationship object.

## Viewing relationships

When looking at the details of an object the relationships that the object participates in are displayed. The use model for viewing the relationships is as follows:

* Search for the object in the relationship [See “View existing data”]
* The bottom panel contains tabs for each type of relationship the object participates in. The tab also indicates the direction (parent or child) of the relationship.
* Select the tab with the correction relationship and direction
* The panel contains a tree structure representing the objects on the other end of the relationship. Each object is represented by their key.
* If a relationship is cyclical such that the parent and child are of the same type you can expand a node to see any objects the node is related to. Repeat as desired.
* To view the details of the participating object right-click on the node and select "Edit"
* To view the details of the relationship right-click on the node and select "Edit Relationship"

## Example of viewing the relationships

## In the previous example the Spray team "Team Smurf" has many Spray operators. This is modeled through the "In team" relationship. Notice under the "In team Children" tab is a list of numbers. These number are actually the key for the Spray operator objects.

## We can examine the participating object by right-clicking on its node and selecting "Edit".sprayTeamView6.png

## This will bring up a new tab with the details of the Team member object. sprayTeamView7.png

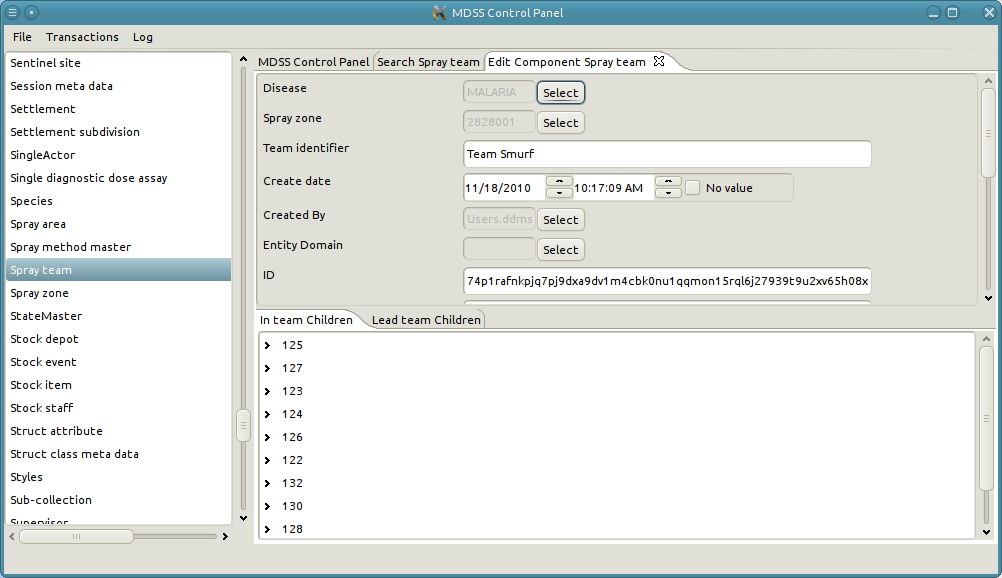
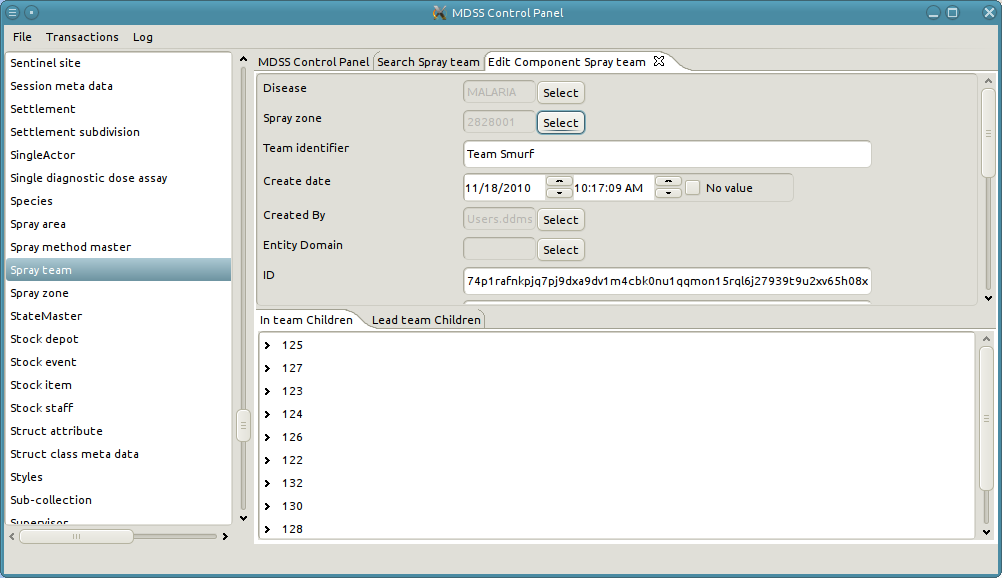
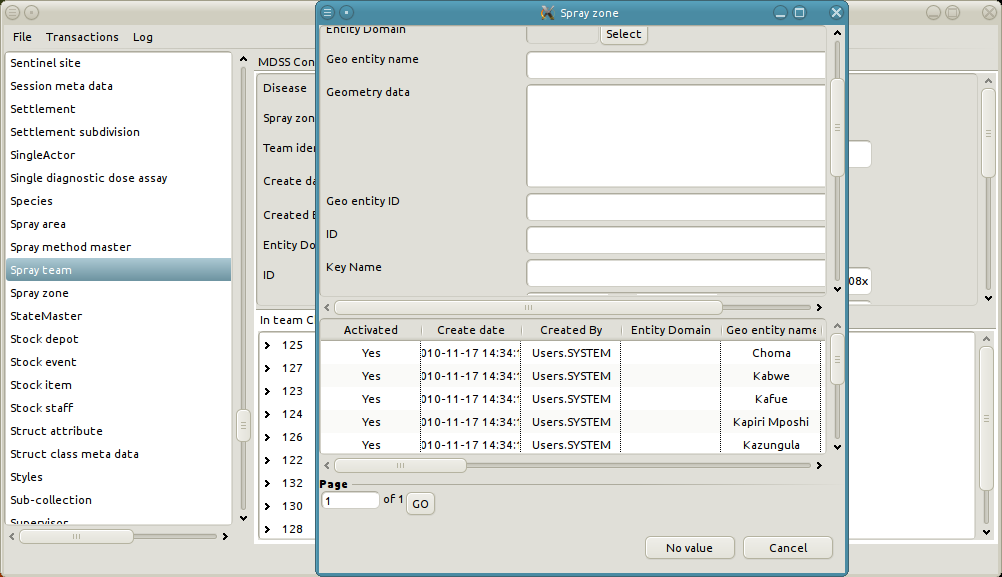
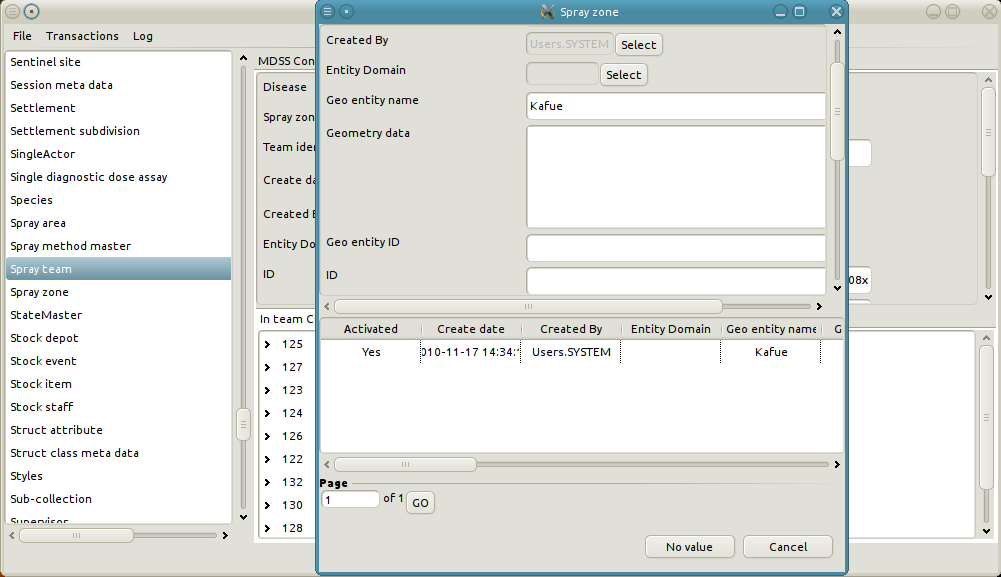
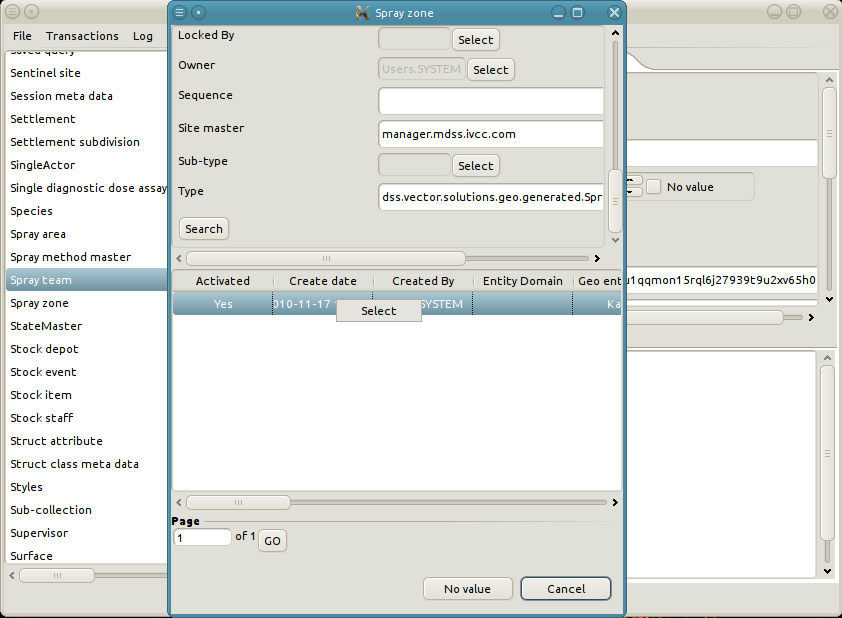
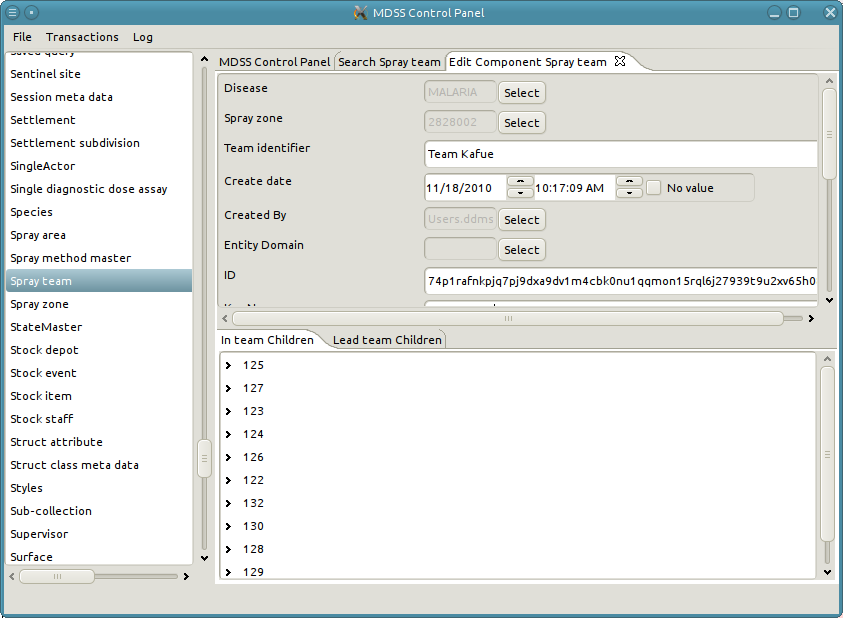
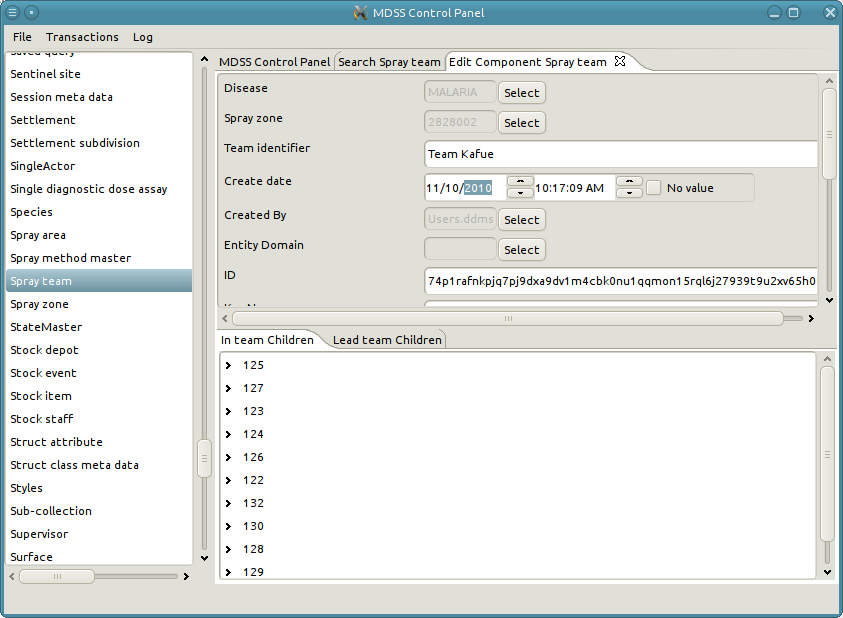
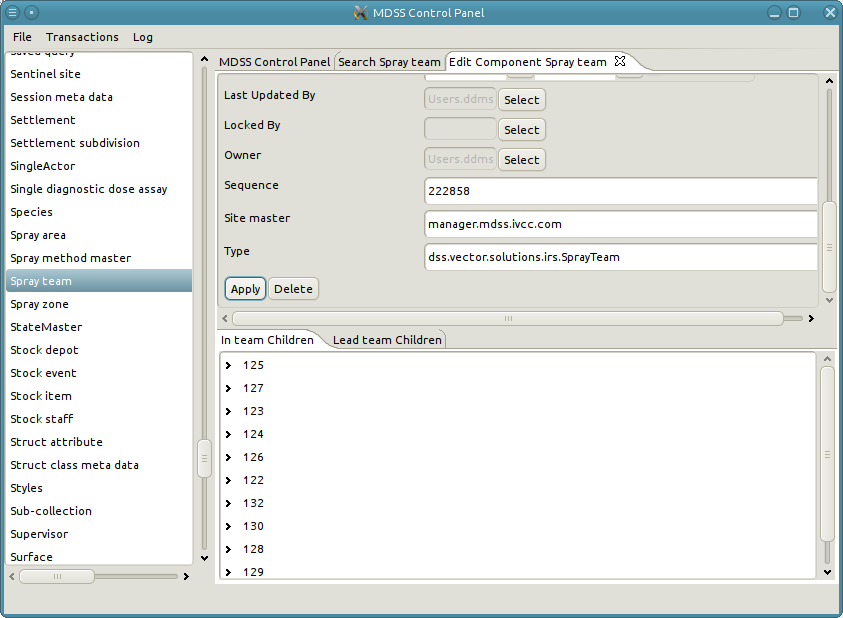
## We can also examine the relationship object itself. To do this click on the "Edit relationship" option. sprayTeamView8.png

## This will bring up a new tab with the details of the relationship object. sprayTeamView9.png

## Edit existing data

* From the data viewing screen it is possible to edit the existing data.
* See [Attributes Section] to get a run down on how to interact with the form attributes.
* The user modifies the data on the form as needed and then clicks on "Apply" to persist the changes to the database.

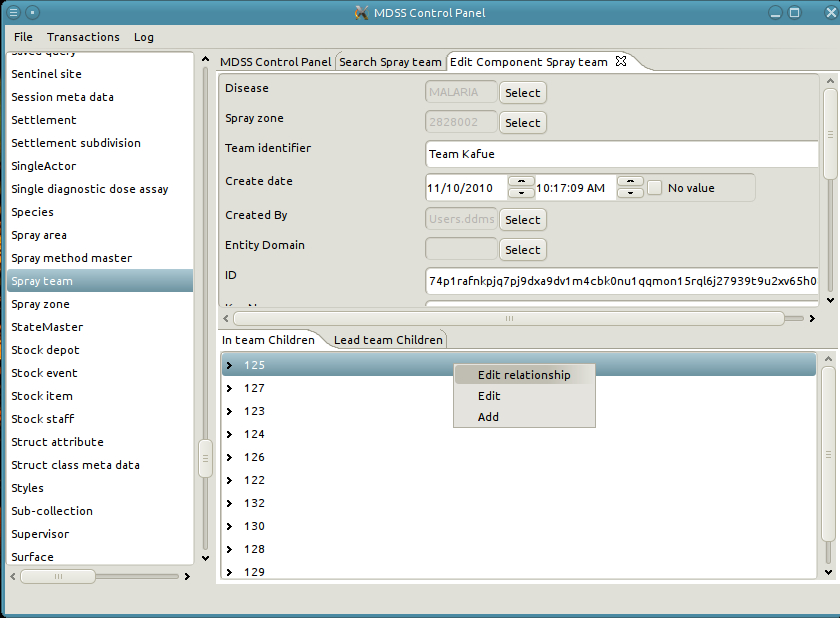
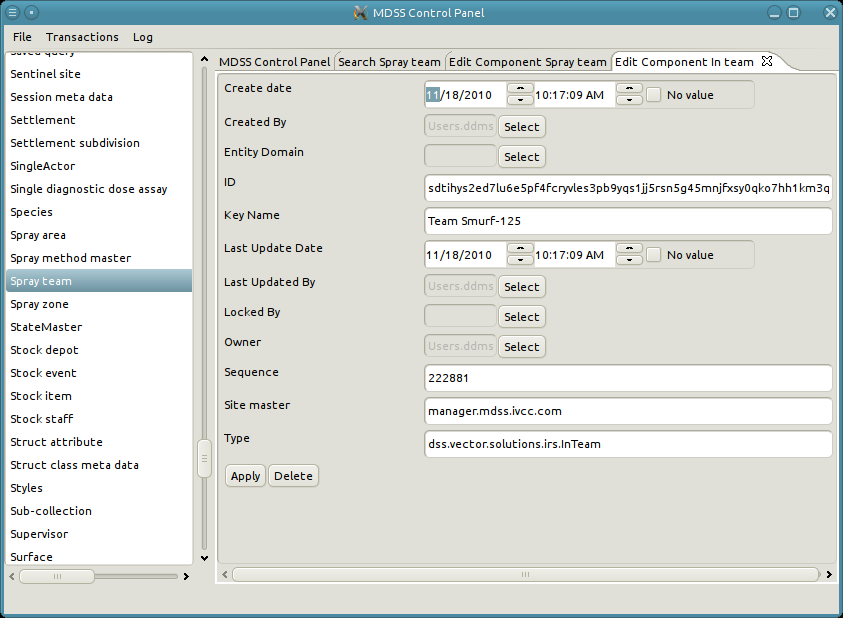
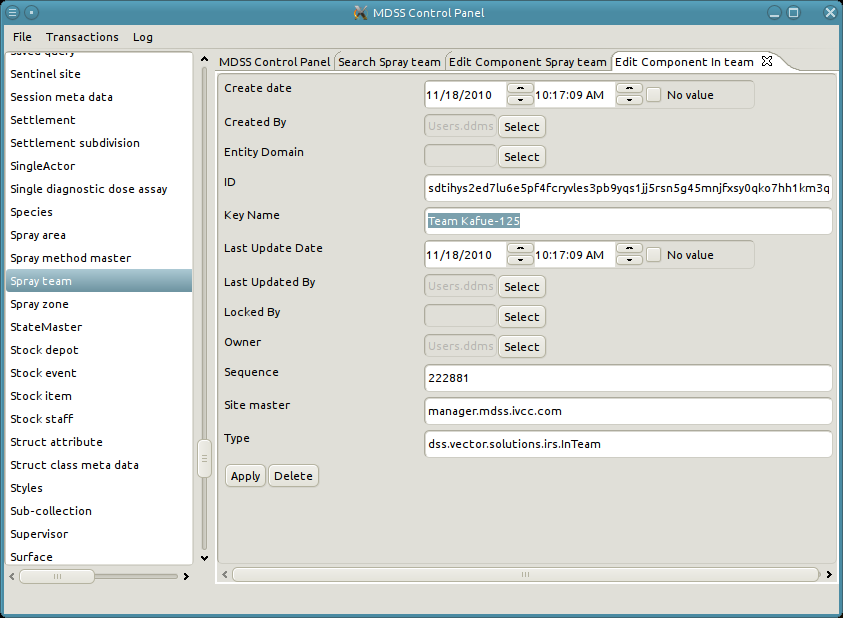
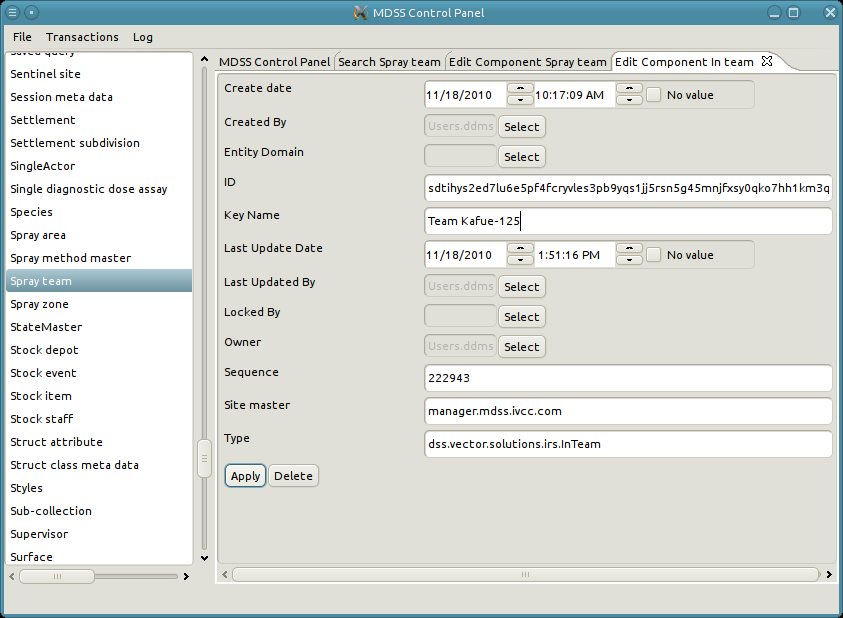
## Example of modifying existing data

1. Continuing from the previous example of viewing the spray team "Team Smurf"
2. Lets modify the spray zone of the spray team. To start we click on the "Select" button next to the Spray zone input field. 
3. This will bring up a pop-up in which the user can search for and select a spray zone. If the Spray zone field already has a value then the pop-up will initially appear with data from the selected spray zone. Note that the user can decide to not change the value by selecting "Cancel" or the user can decided to remove the current value by selecting "No value".
4. Remove some of the criteria in the form to get a broader result set. This can also be accomplished by selecting "No Value" and then hitting the Spray zone "Select" button again.
5. Similar to a search tab we can refine the list of possible spray zones by adding criteria to the form. For instance, we want to restrict the spray zones to ones which have a geo entity name of "Kafue".
6. To select the "Kafue" spray zone we can right click on the row and hit "Select" or simply double click the row.
7. The Spray zone of the Spray team has now been changed to the "Kafue" spray zone. Similarly, let's change the Team identifier of the Spray Team. In order to change the Team identifier simply type in the desired value in the Team identifier text field. 
8. Finally, lets modify the Create date of the Spray Team to 11/10/2010. 
9. Now that we have finished modifying all of our desired values we simply click on apply to persist the changes to the database. 
10. If we search the spray teams we can see the new changes to "Team Smurf"

### Edit an existing relationship

Editing a relationship is basically the same as editing a object. The major difference is finding the relationship object in which to edit. See "Viewing a relationship" to learn how to view the relationship objects. Once the user has a tab with the relationship object up, then editing the relationship is the same. One important fact to note is that it is impossible to change the parent or child or a relationship. If a relationship is created with the wrong parent or child the only recourse is to delete the relationship object, create a new one with the correct parent and child, and copy the values from the deleted relationship.

## An example of editing a relationship

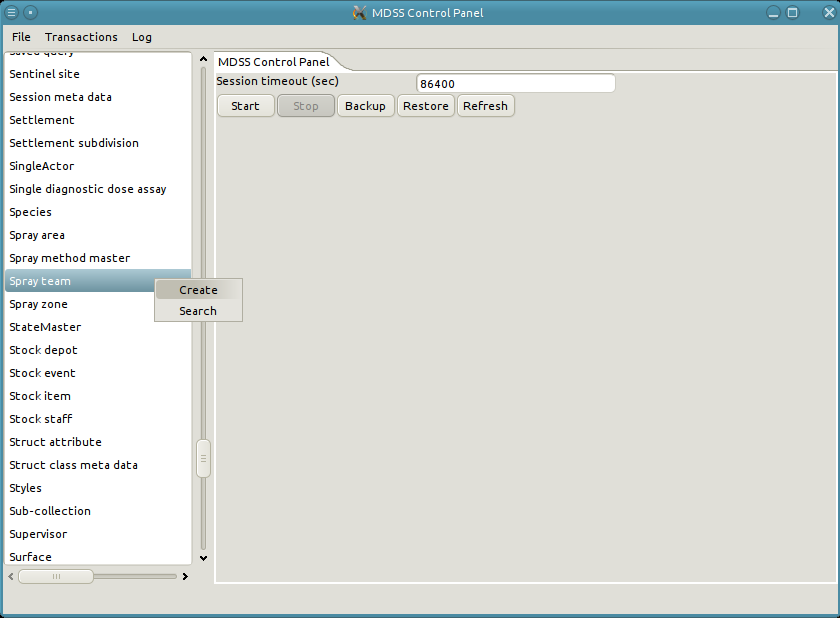
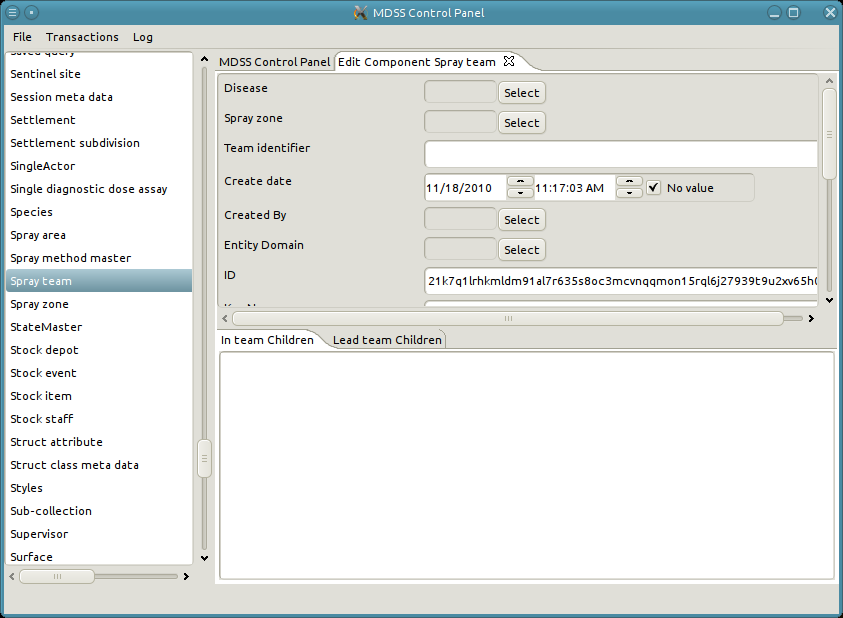
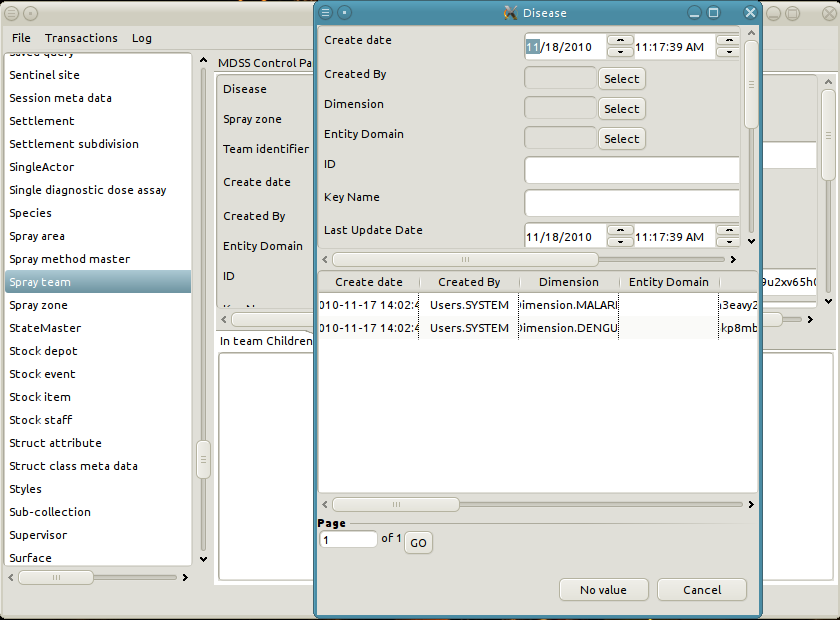
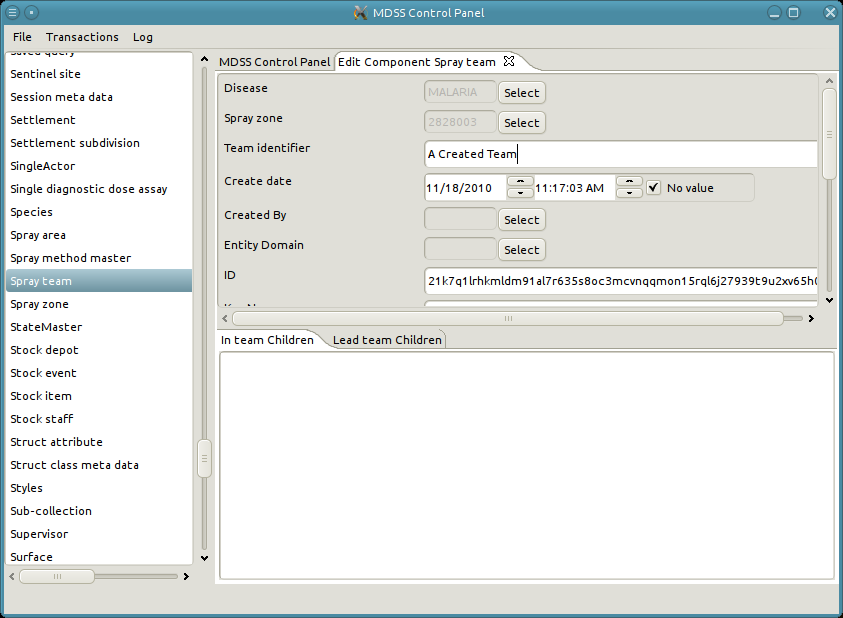
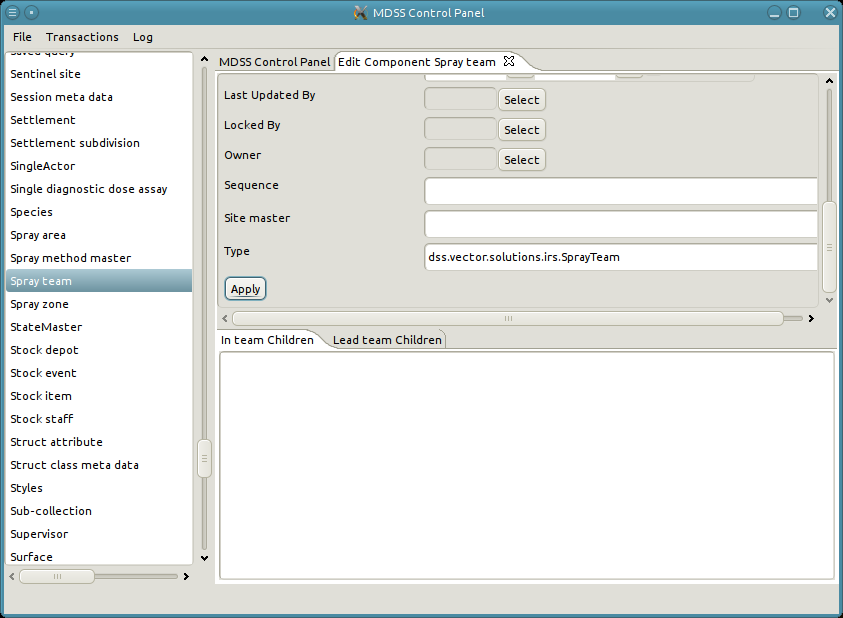
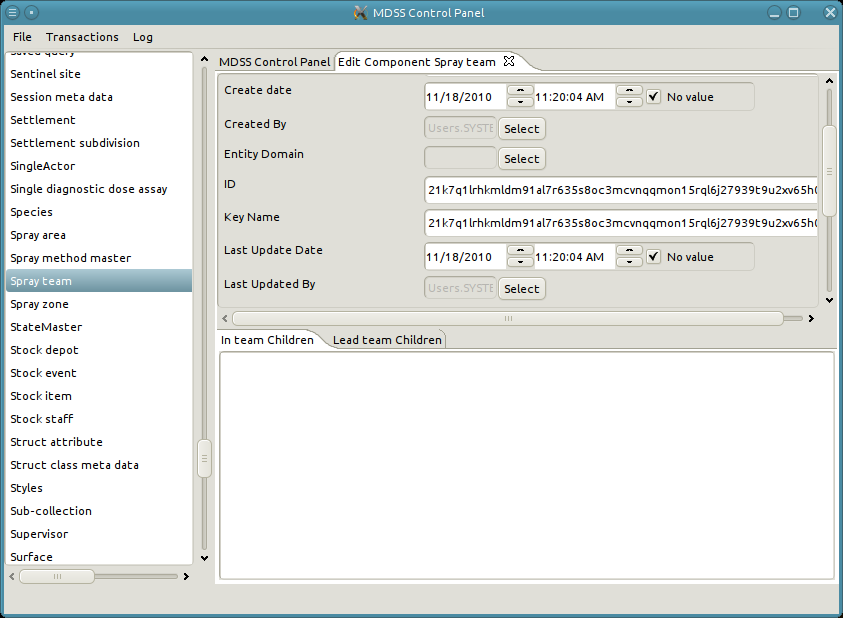
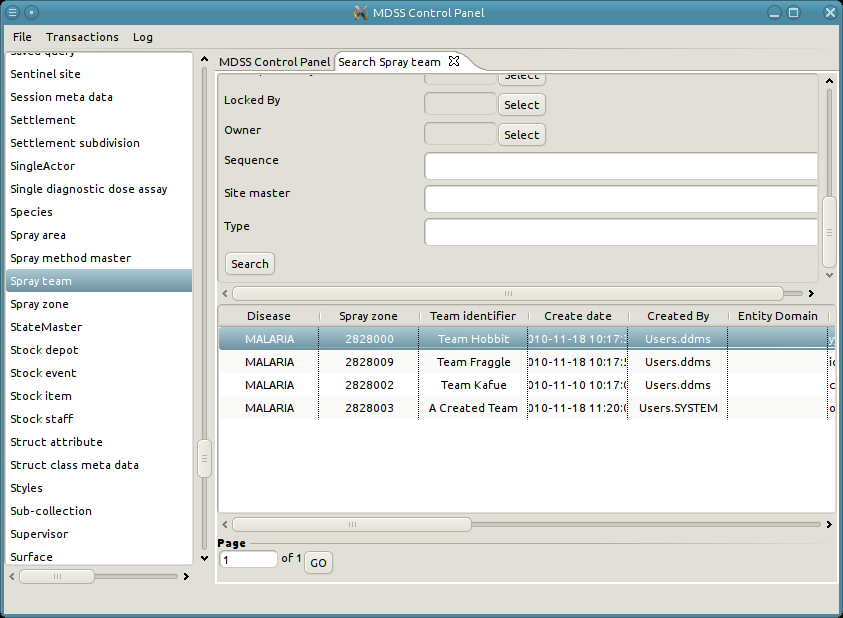
1. Find the relationship the you want to edit and click "Edit relationship". In this example we want to modify the relationship for Team member "125" of the Spray team "Team Kafue".
2. This will bring up a new tab with the relationship object.
3. Edit the object. In this case we want to change the Key Name to be "Team Kafue-125".
4. Apply the object to persist the changes. 

## Create new data

It is doubtful that the user will even need to create new data through the manager. However, the option has been included as a last resort. The basic use case for creating data is as follows:

* The left panel lists all data types in the system
* Right-click the desired type from the list and select "Create"
* A new tab opens with the create form for the selected data type
* Fill in the form
* Click “Apply”

**An Example of creating a new Spray team**

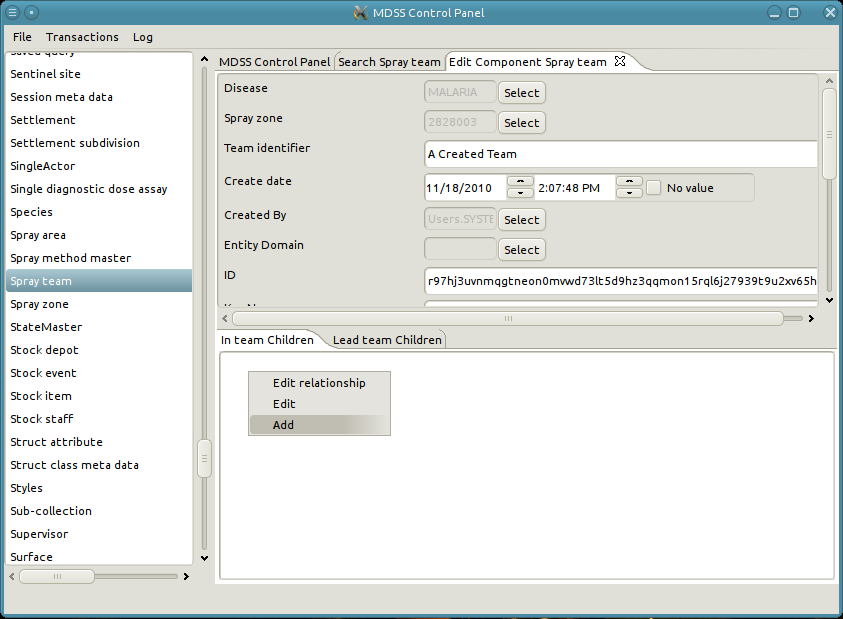
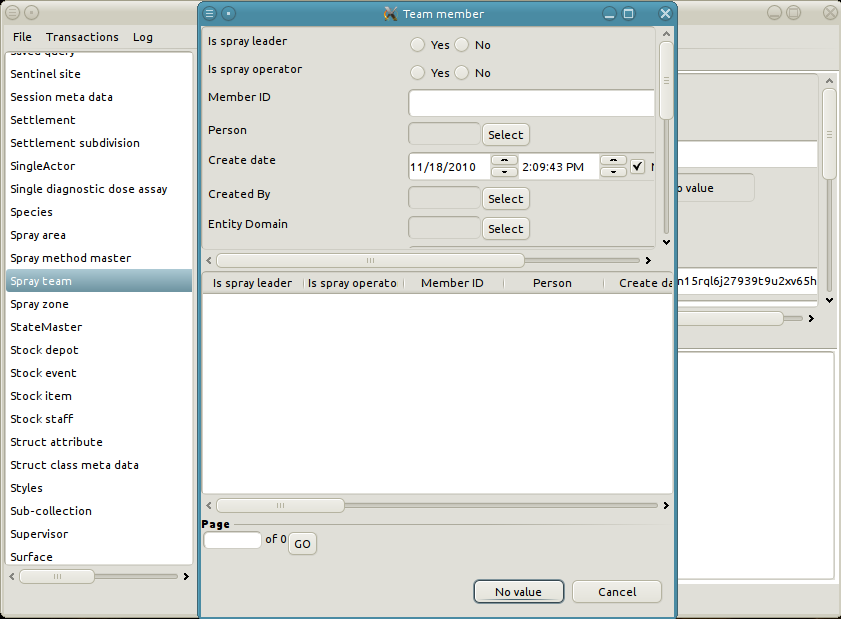
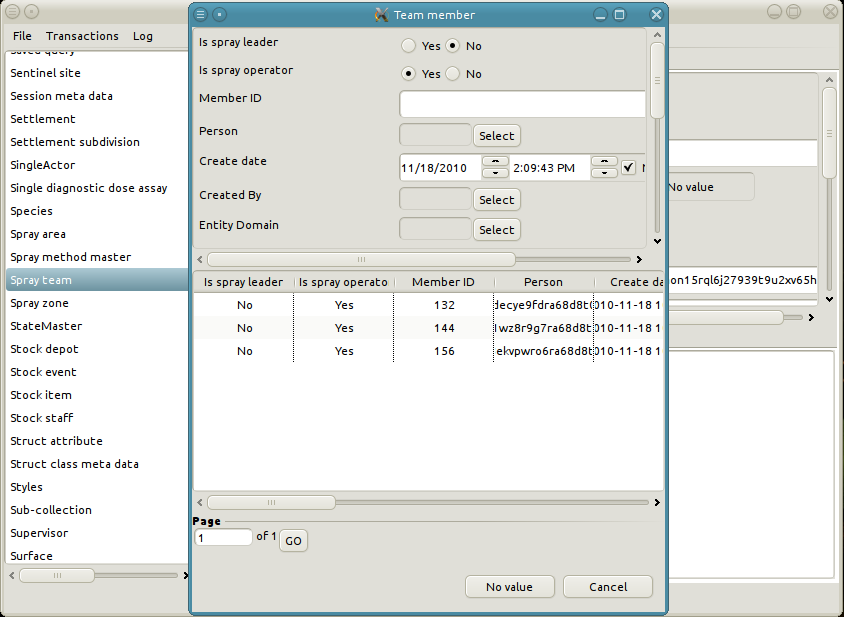
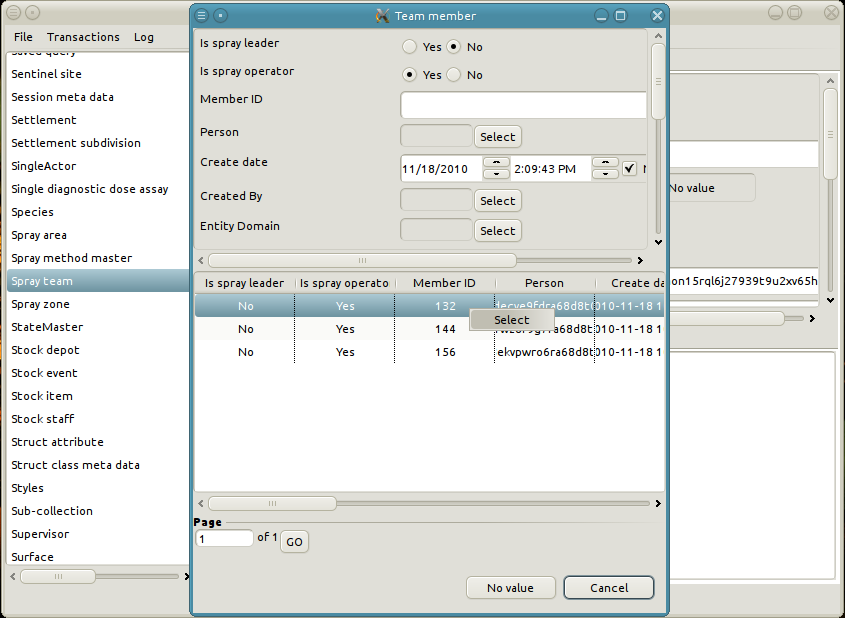
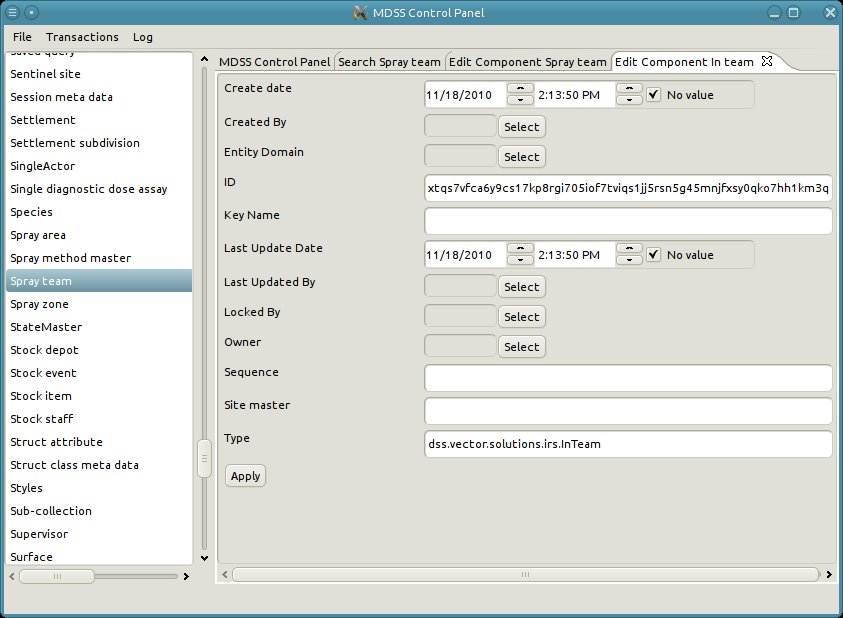
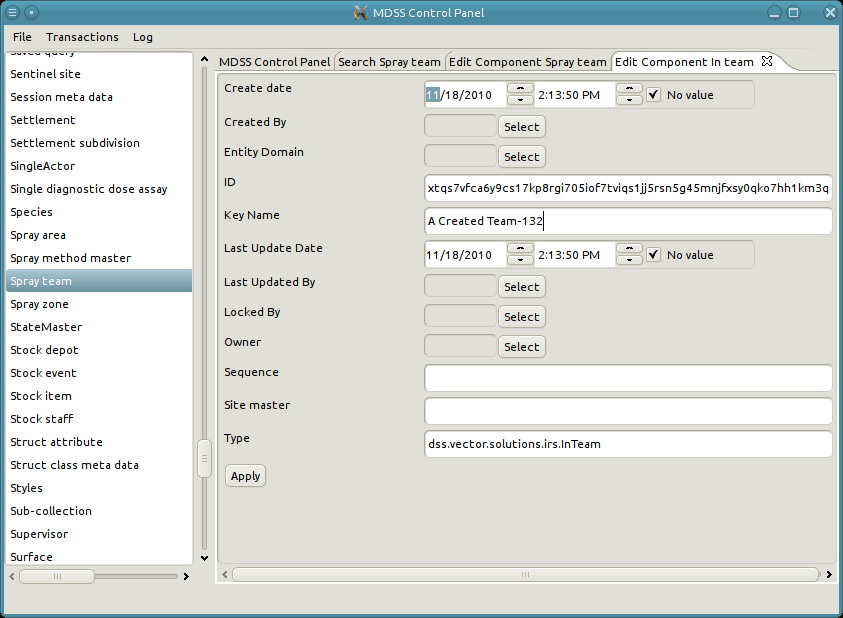
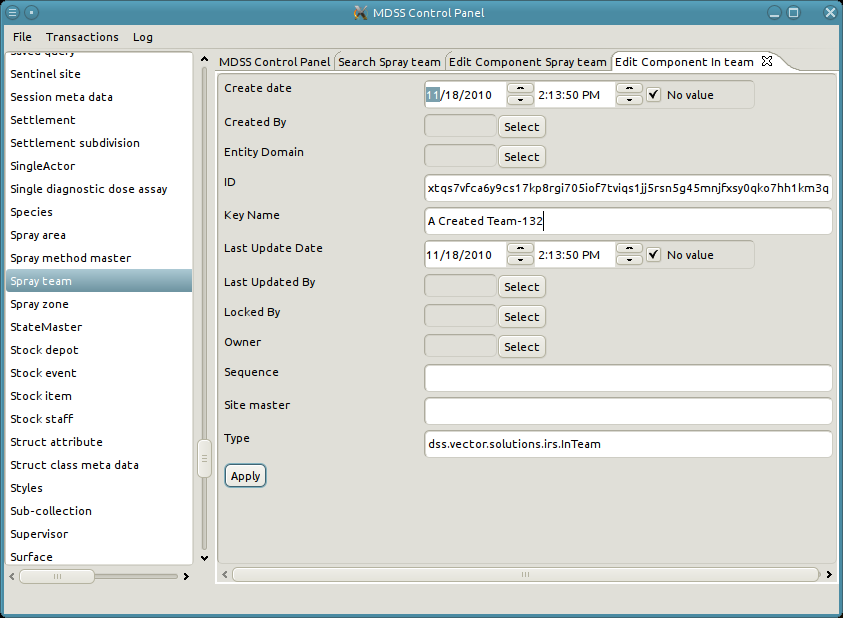
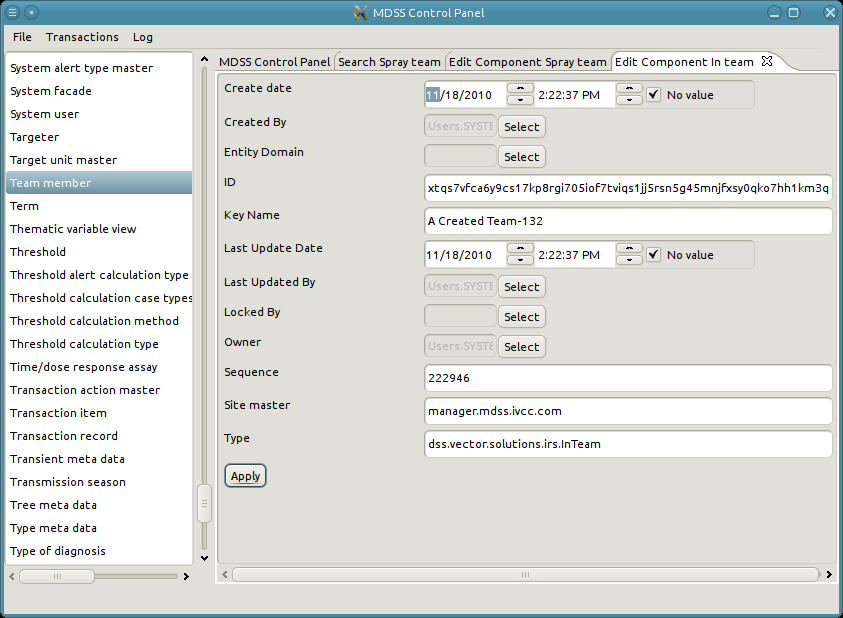
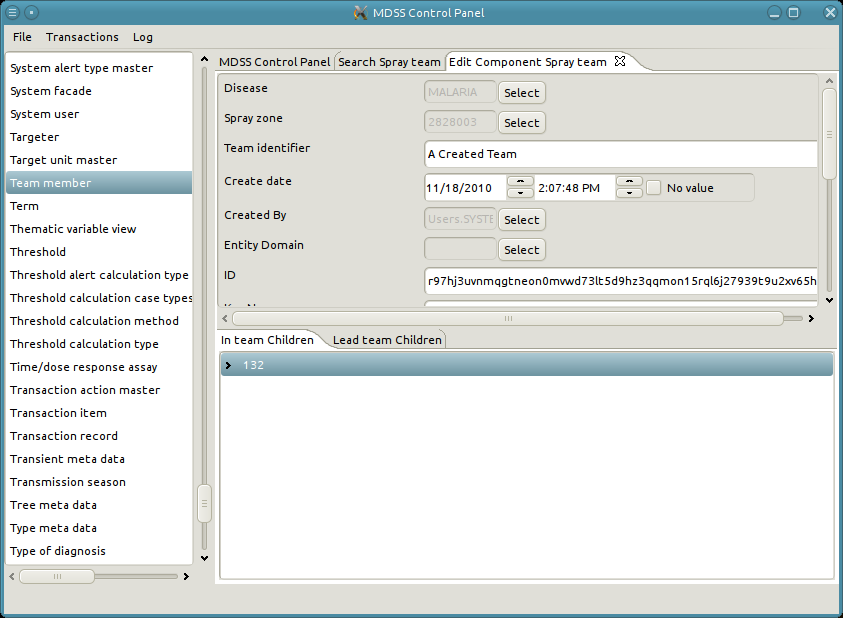
1. Right click on Spray team in the left panel and select "Create".
2. A new tab will open up with the form to create the new Spray team.
3. In order to create a Spray team we need to fill out the Disease, Spray zone, and Team identifier fields. First, lets set the Disease to malaria. 
4. Second, we will set the Spray zone to "Kapiri Mposhi".
5. Lastly, we will set the Team identifier to "A Created Team".
6. Finally we can persist the new Spray team to the database.
7. Notice that some of the system attributes were automatically updated once the Spray team was applied. These attributes are used by the system for book keeping and to ensure data integrity. 
8. Once again if we search for all of the Spray teams in the system then we will see the newly created Spray team.

### Add a new relationship

Adding a relationship is accomplished in the same panel as viewing relationships. See "Viewing a relationship" to learn how to view the relationships. The use model for defining a new relationship is as follows:

1. Anywhere in the relationship panel the user right-clicks and selects "Add a relationship"
2. A pop-up appears to search for and select the other participating object in the relationship.
3. Once the other participating object has been selected a new tab appears for editing the relationship.
4. Modify the relationship form as needed.
5. Click on "Apply" to persist the relationship to the database.

## An example of adding a new relationship

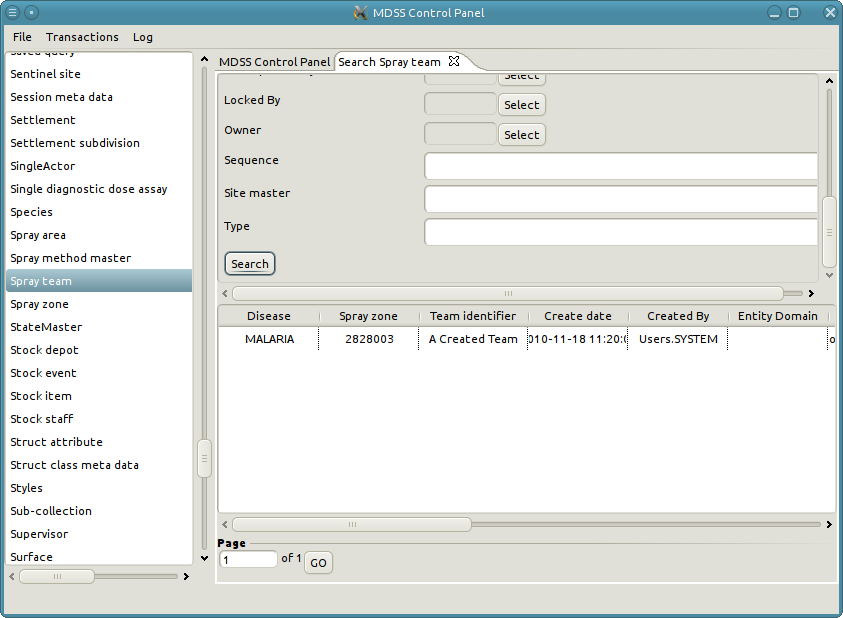
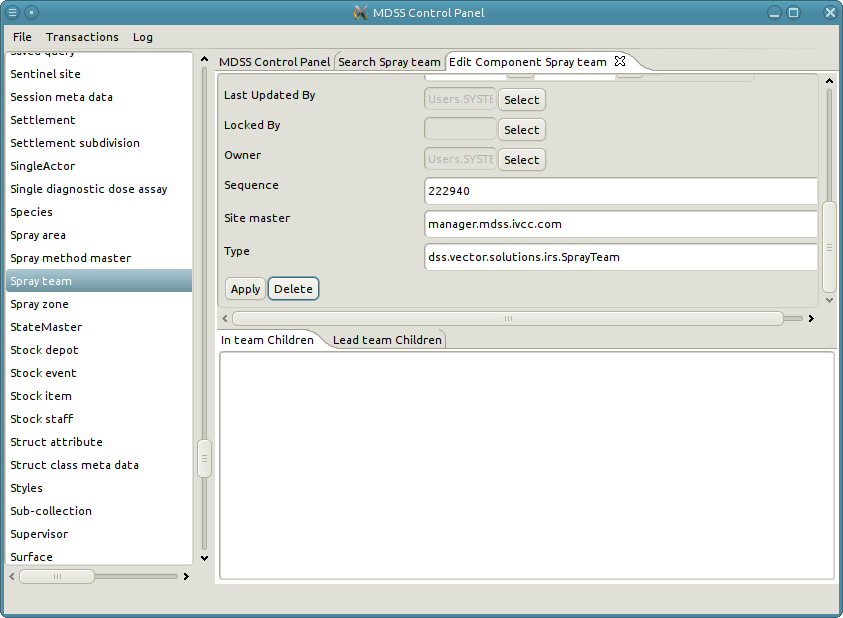
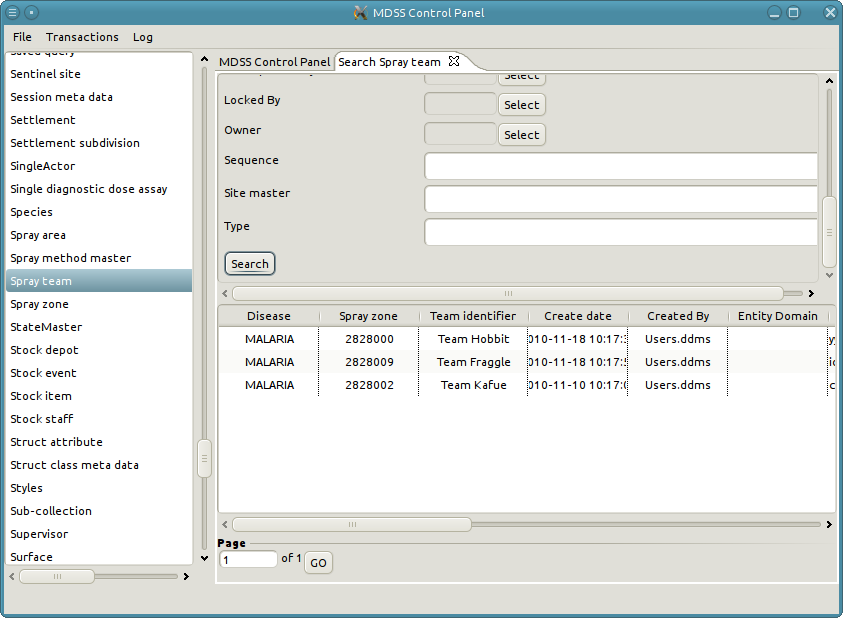
1. Select "Add a relationship" from the relationship panel. In this example we want to assign a new Spray operator to the newly create Spray team "A Created Team".
2. A pop-up in which to search and select a Team Member appears.
3. Refine the search criteria as desired. In this case we only want Team members which are Spray operators and not Spray leaders.
4. Select the appropriate participant. For example we want Team member "132".
5. A new tab is opened with the form for editing the relationship object.
6. Edit the values as needed. We want to change the key of the relationship to be "A Created Team - 132".
7. Click apply to persist the object to the database.
8. Some fields will be auto-populated when the relationship is applied
9. View the Spray team "A Created Team" to see the new relationship with Team member "132"

## Delete existing object

It is doubtful that the user will even need to delete existing data through the manager. However, the option has been included as a last resort. The basic use case for creating data is as follows:

* The left panel lists all data types in the system
* Double click the data type to search for records of that type. Alternatively, right-click and select "Search"
* A new tab opens with a search form for the selected type
* Fill in search criteria as needed
* Click "Search" at the bottom of the form
* Search results appear in the paginated table below the form
* [Optional] Access additional pages of search results with the input field below the results table
* Double click a row of the result table to view the object. The user can also right-click and select "Edit".
* Click on the "Delete" button next to "Apply" to delete the object.

**Example of deleting a Spray team**

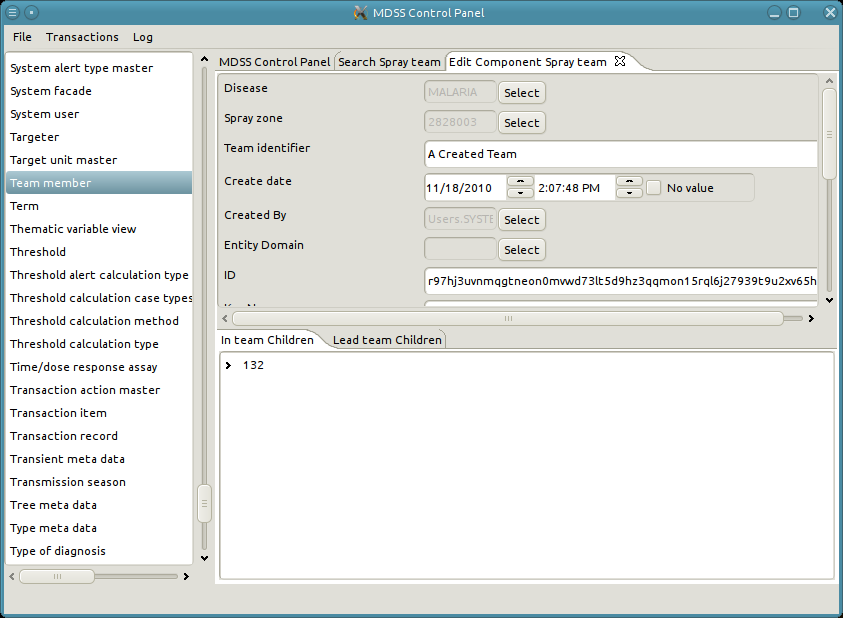
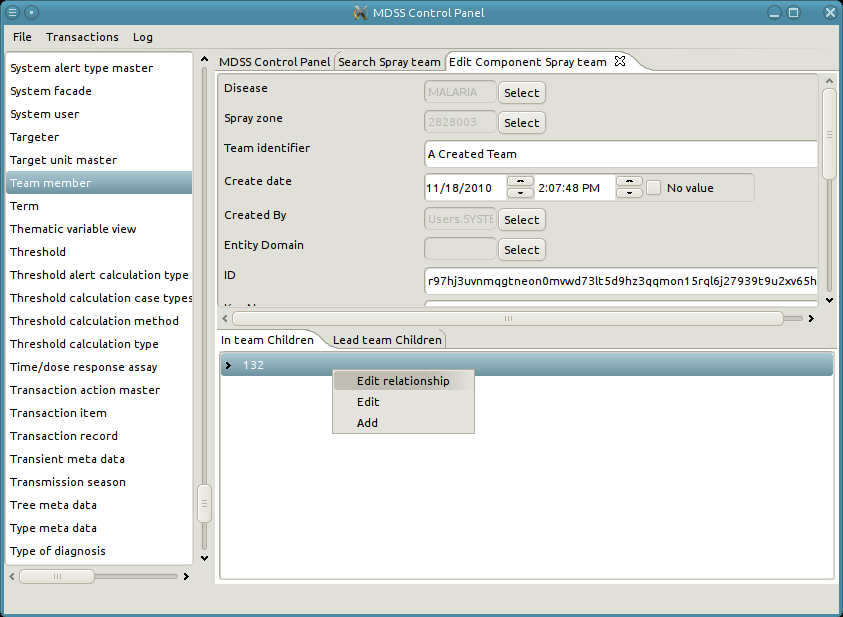
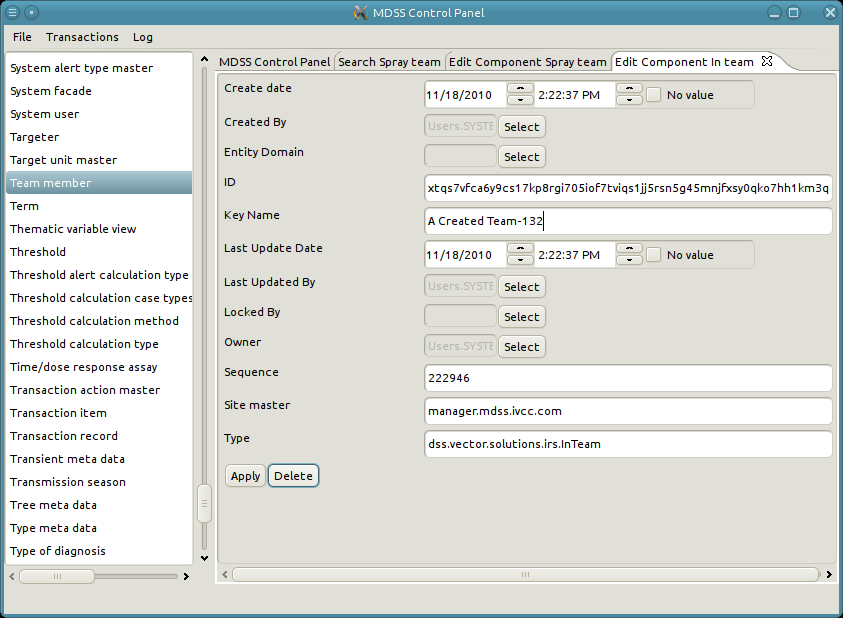
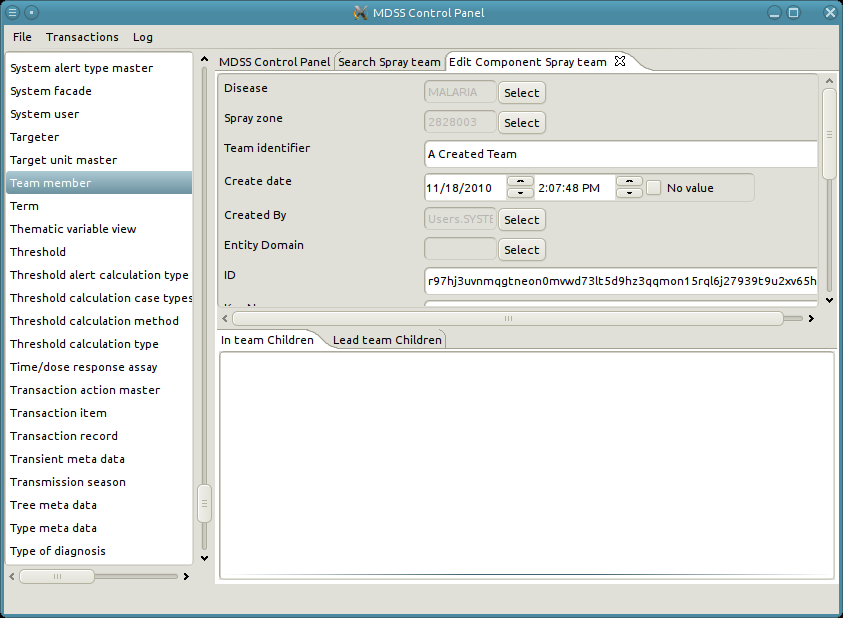
1. In this instance we want to delete the Spray team "A Created Team". As such, we first search the Spray teams for "A Created Team".
2. Double click on the row to open up a new tab with the details of the Spray team. 
3. To delete the Spray team click on the "Delete" button.
4. Note when deleting the tab of the deleted Spray team automatically closes. Finally, if we search all of the Spray teams in the system the "A Created Team" will be gone.

### Deleting relationships

Similar to deleting an object, deleting a relationship is done when viewing the relationship. See "Viewing a relationship" to learn how to view the relationships. The use model for defining a new relationship is as follows:

1. Select the relationship object to edit.
2. A new tab with the details of the relationship is opened.
3. Scroll down to the bottom of the tab and click on the "Delete" button.

## An example of deleting a relationship

1. We want to delete the relationship we created in the example for creating relationships. As such we first need to view the Spray team "A Created Team".
2. Next, right-click and select "Edit Relationship" on node "132".
3. Now, click on the "Delete" button to delete the relationship from the system.
4. Once again look at Spray team "A Created Team" to ensure that the relationship was deleted.

**Edit a Reference Attribute**

* Click “Search” next to the reference attribute
* A search dialog pops up
* Fill in criteria and click “Search.” Results appear in the bottom panel.
* Double-click the desired row or Right-click and click “Select”
* The pop-up closes, and the reference field contains the key of the selected object.

### 

# Transactions

As previously stated a transaction is an atomic set of creates, updates, and deletes on one or more objects that collectively implement an action in DDMS. For instance, the creation of a Spray team and assignment of its respective Spray operators is a single transaction. A transaction can be broken down into many transaction items.

A transaction item is a single action which occurs during a transaction. The creation of a Spray team has many transaction items. There is a transaction item for the creating the spray team itself. Additionally, there is a transaction item for every assignment of a spray operator to the spray team. A transaction is comprised of all of its transaction items.

At the database level committing a transaction is an all or nothing operation. That is to say either all of the transaction items in the a transaction are committed to the database or none of the transaction items are committed. During the commit process a database will create a save point before committing any of the transaction items. If any of the transaction items fail, the database rollbacks its state to that of the save point.

DDMS creates transaction records for every transaction which is successfully committed to the database. A transaction records record the transaction items required to reproduce a transaction. By sharing transactions a transaction can be reproduced on a different install of DDMS. This is the mechanism for sharing and synchronizing data in DDMS. The synchronization manager primarily facilitates data transfer among multiple installations through use of the viewing, importing, and exporting transaction records.

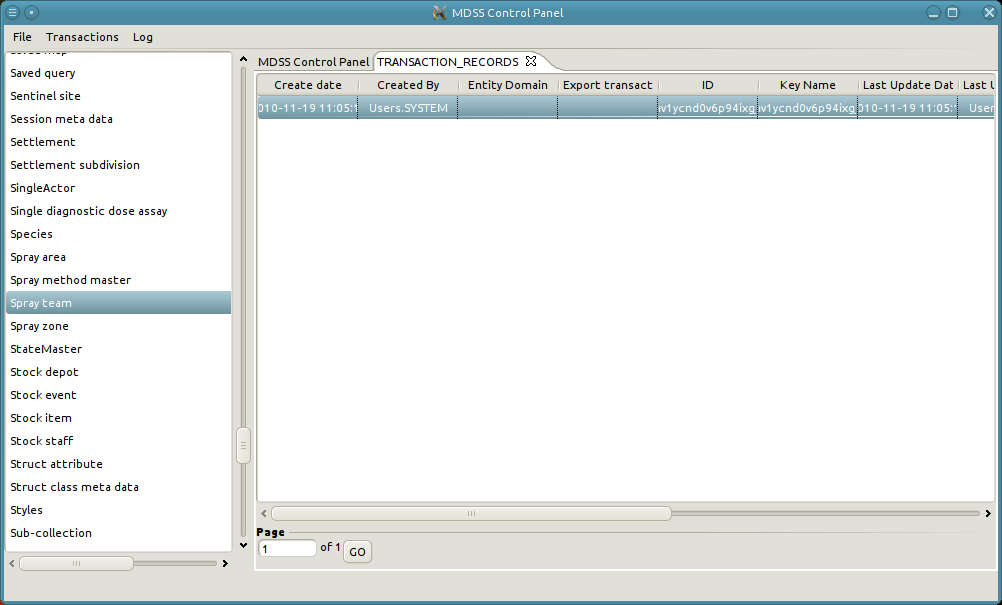
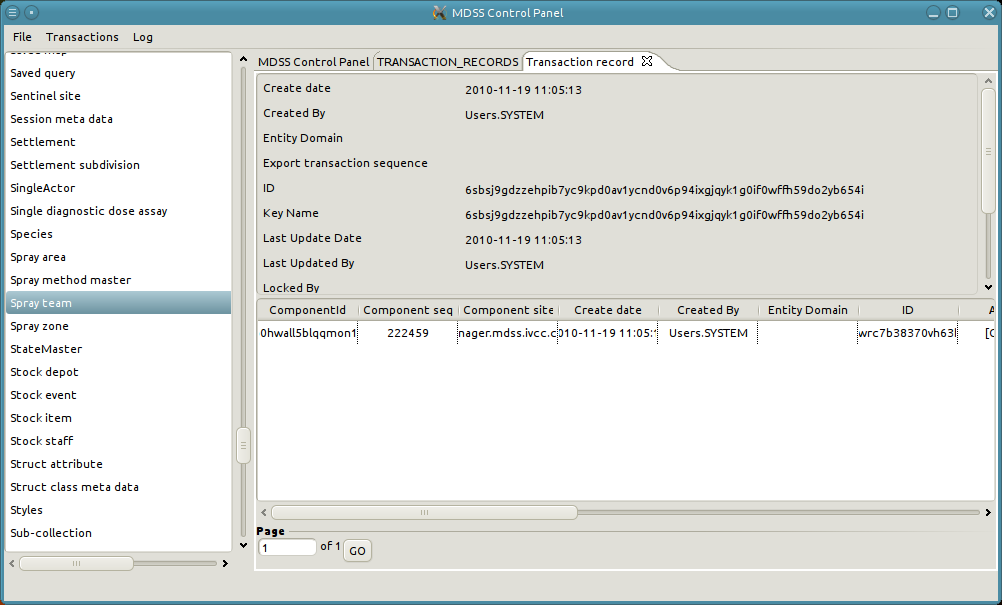
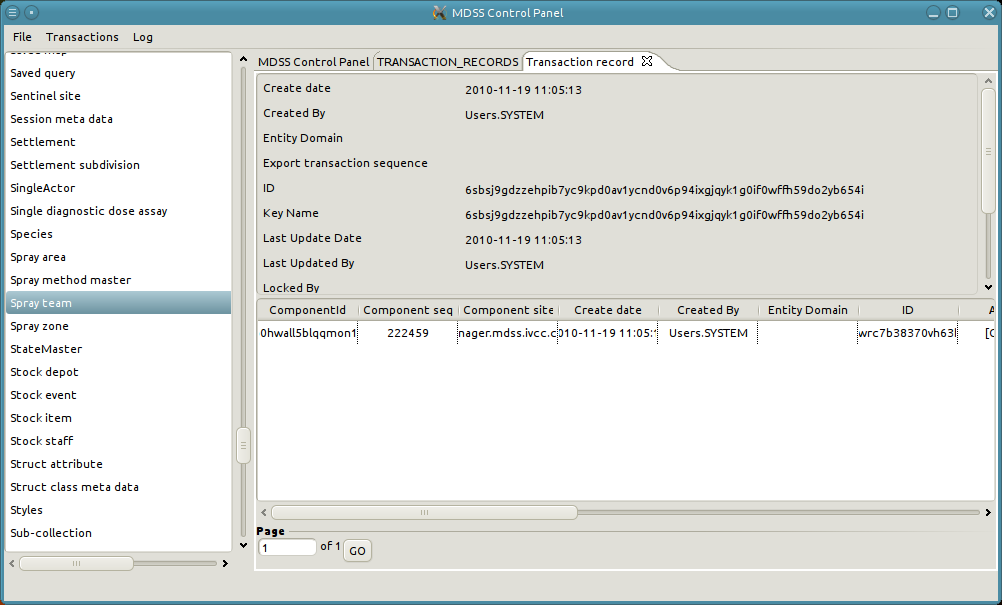
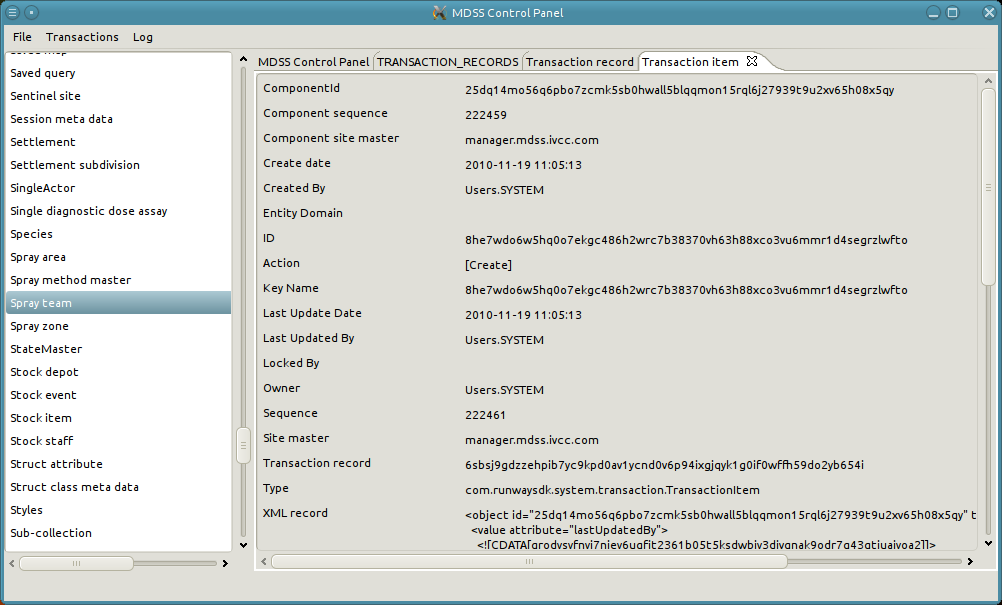
## View Transaction Records

The user may want to view the transaction records in the system to find which transaction record is responsible for creating some data. Additionally, it is possible to determine which records have been exported.

* Open the "Transaction" menu and click "View Transaction Records"
* The main panel opens a paginated table listing all transactions in the system
* Double click a row to open the details of the transaction record in a new tab
* The bottom panel of the transaction record tab contains a list of transaction items in the record
* Double click a transaction item to open its view in a new tab

## 

## An example of viewing transaction items

1. Open the "Transaction" menu and click "View Transaction Records".
2. A new tab will open with all of the transaction records in the system. In this case there is only one transaction record. 
3. Double click on the row of the transaction record to see its details. Note that this transaction record has yet to be exported. As such it does not have an export sequence number. 
4. The table at the bottom contains all of the transaction items in a transaction record. This transaction record has a single transaction item. Double click on the row of the transaction item to examine its details. 
5. A new tab is opened with the details of the transaction item.

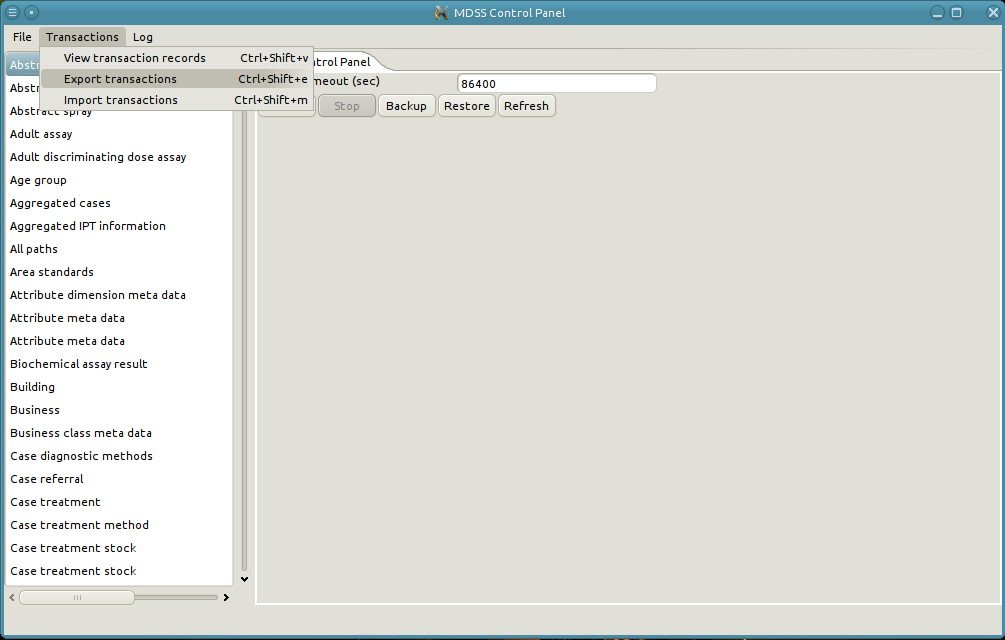
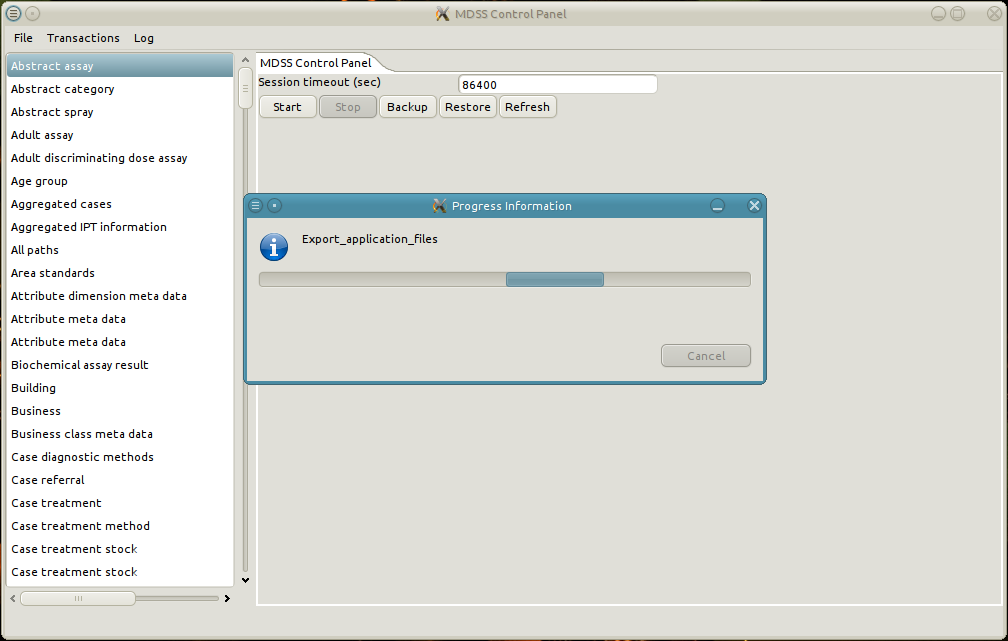
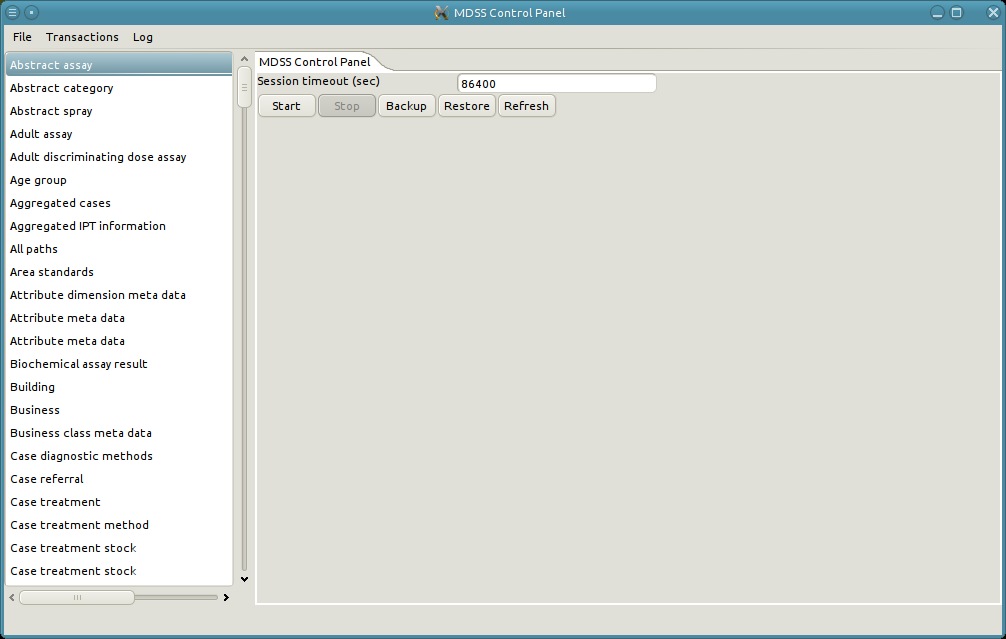
## Export Transaction

In order to propagate an install's data to another install it most export it's transactions. In general the use model for exporting transactions is as follows:

* Ensure that the web server is shut down
* Open the "Transaction" menu and click "Export Transaction"
* A pop-up opens with three options: All, Range, and Not Exported
  + All: Exports all transactions from the node regardless of the fact that they might have already been exported. An import node will skip all transactions which it has previously imported. As such, when exporting there is no need to worry about exporting the same transaction multiple times. However, exporting all transaction will create much larger files.
  + Range: Exports transactions between the specified start and end sequence numbers. This option should only be used if an install is requesting the transaction of a specific export sequence number.
  + Not Exported: Exports all transactions which have not been previously exported. In general this option should be used when generating export files.
* Click “Choose File” and select a destination for the export file
* Click “OK” to being the export
* A pop-up appears with status information concerning the export. The pop-up closes when then the export completes.

## 

## Example of exporting transactions

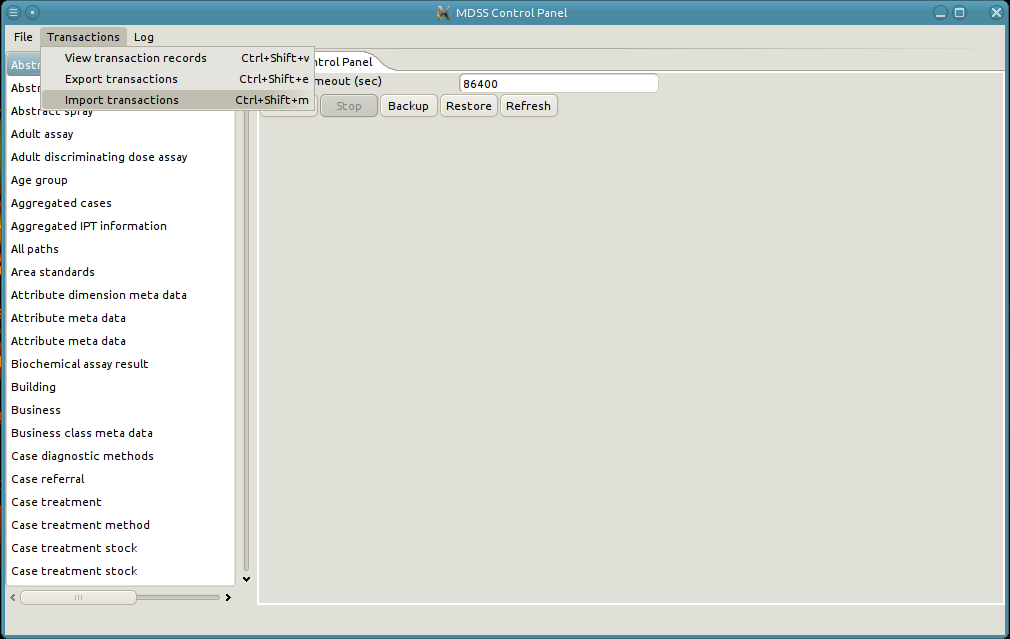
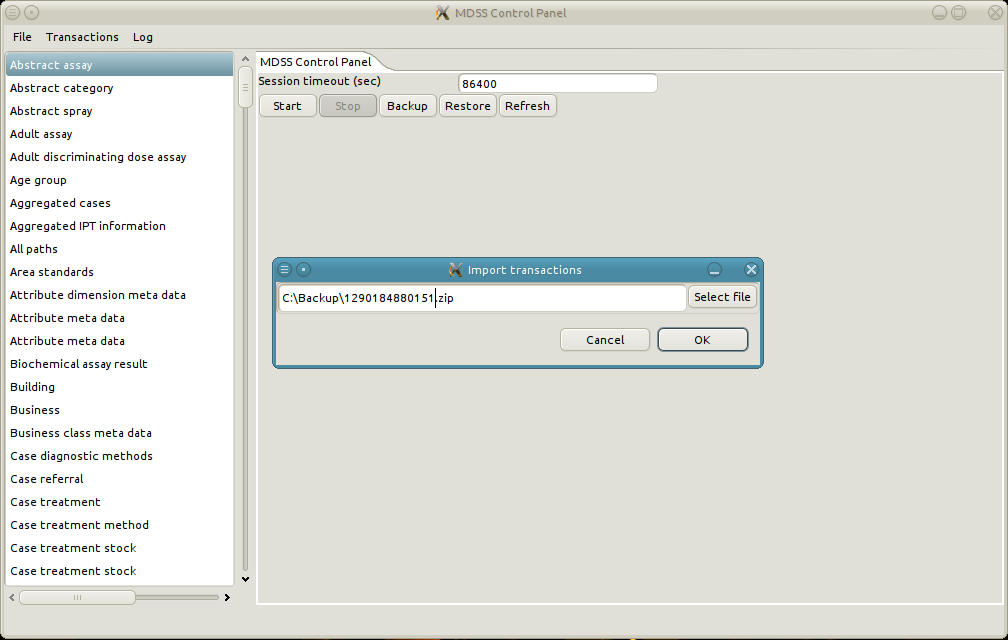
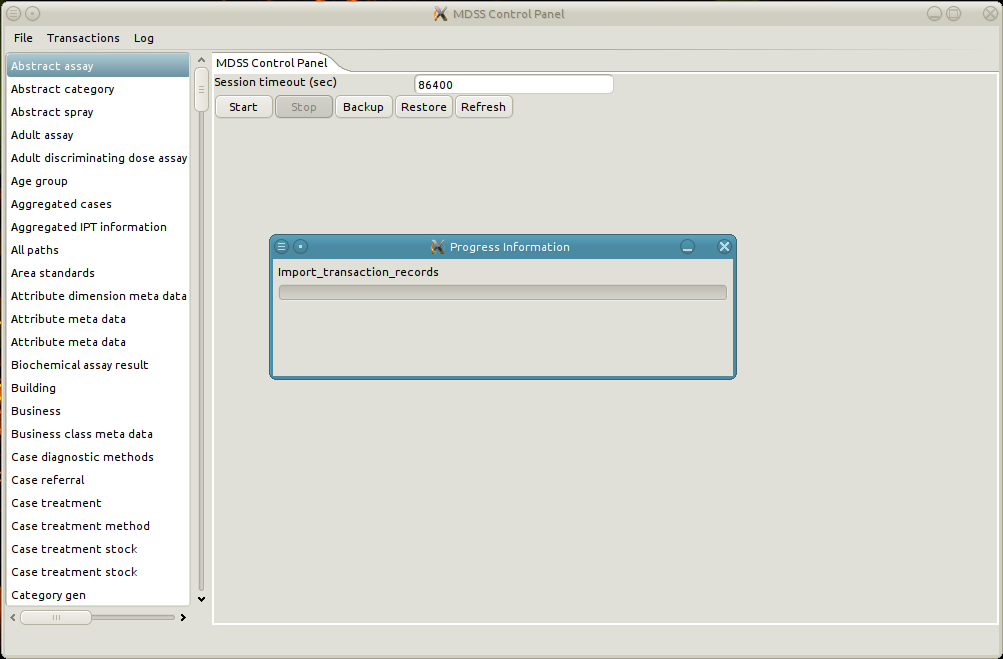
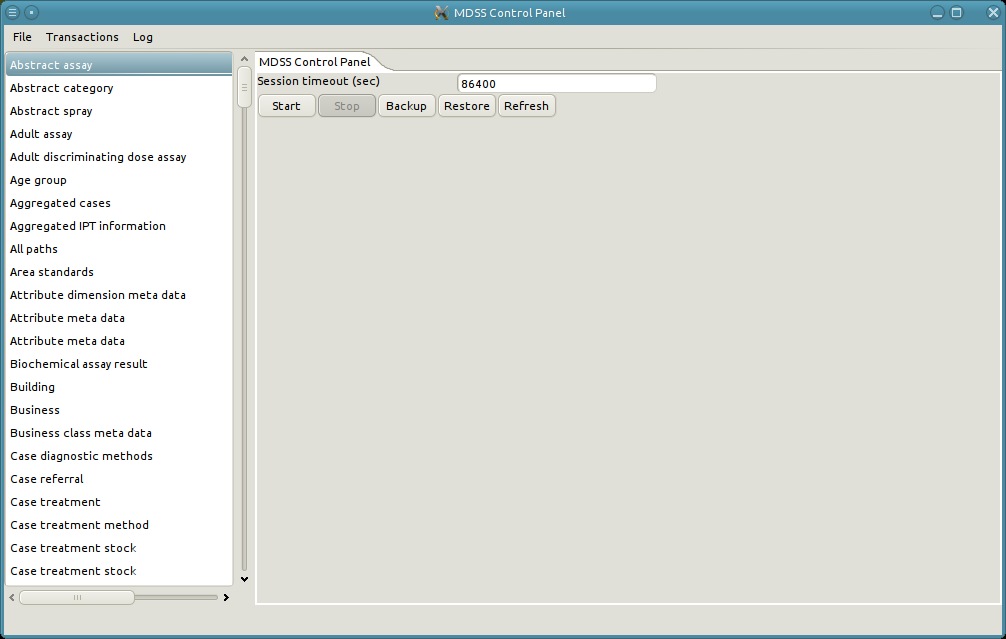
1. Open the "Transaction" menu and click "Export Transaction". Note that the server is shutdown.
2. Select desired location of the export file and the sequences to export. In this example we are just going to export everything. Finally, click "Ok".
3. A pop-up appears with the status of the export.
4. The pop-up closes when the export is complete.

## Import Transaction

In order to retrieve data from a different install that install's transactions must be imported. The use model for importing transactions is as follows:

* Ensure that the web server is shut down
* Open the "Transaction" menu and click "Import Transaction"
* A file selection dialog opens
* Select the import zip file and click "Import"
* A pop-up appears with status information concerning the import. The pop-up closes when the import completes.
* Conflicts can arise during the import process. These must be resolved manually. See [Conflict Resolution] for in an depth look at resolving conflicts.

**An example of importing transactions.**

1. Open the "Transaction" menu and click "Import Transaction". Note that the server is shutdown.****
2. Select the file to import and click ok.****
3. A pop-up appears with the status of the import.****
4. The pop-up automatically closes when the import is finished.****

**Conflicts**

**Conflict resolution control flow**

The control flow for resolving a conflict is relatively straight forward. When a conflict occurs during an import, the import process pauses, and waits for the conflict to be resolved. A new tab will open with the cause and the error which caused of the conflict. The user then manually resolves the conflict. Once the conflict has been resolved, the import process continues. Due to the fact that an import may contain more than one conflict this control flow may be repeated many times during the import. Note that it is impossible to abort an import once the process has started.. In addition, it is critical that the manager is not closed nor the computer is shutdown while an import is still in progress. Doing so can result in corrupt and incomplete data.

**Conflicts on relationships**

In addition to all the conflicts which can occur on normal objects, relationships can have conflicts occur when the parent or child object is missing. These conflicts are especially tricky to resolve because it is impossible to modify the parent or child object of a relationship. In general the best way to resolve a conflict when the parent or child is missing from the relationship object is to delete the imported relationship. However, if it is imperative that the relationship is persisted, then it possible to create a new relationship object with a different parent or child, and copy the values of values of the failed import relationship. Once the values have been copied, apply the new relationship and delete the import relationship.

**Resolving a Conflict**

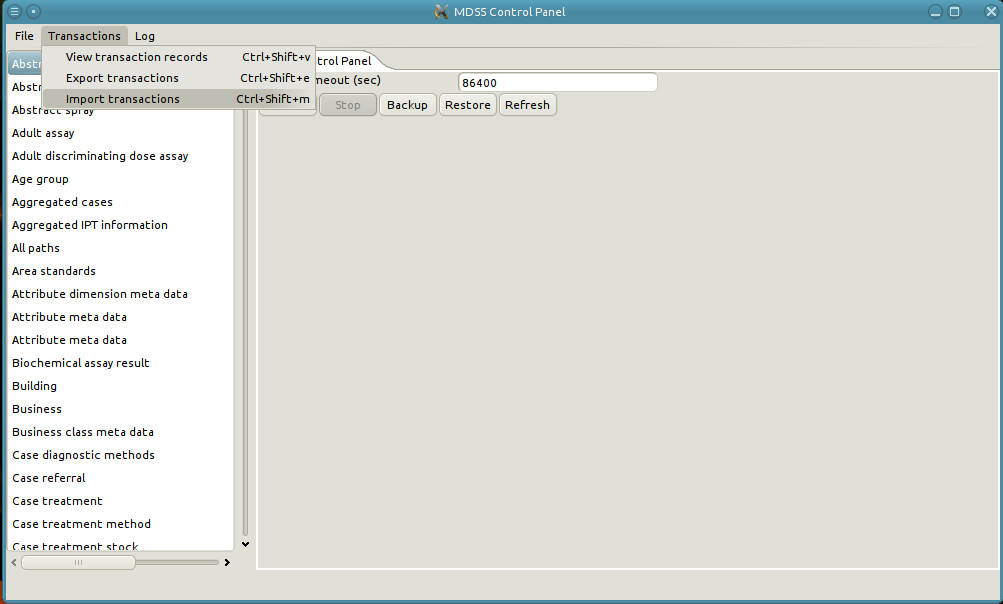
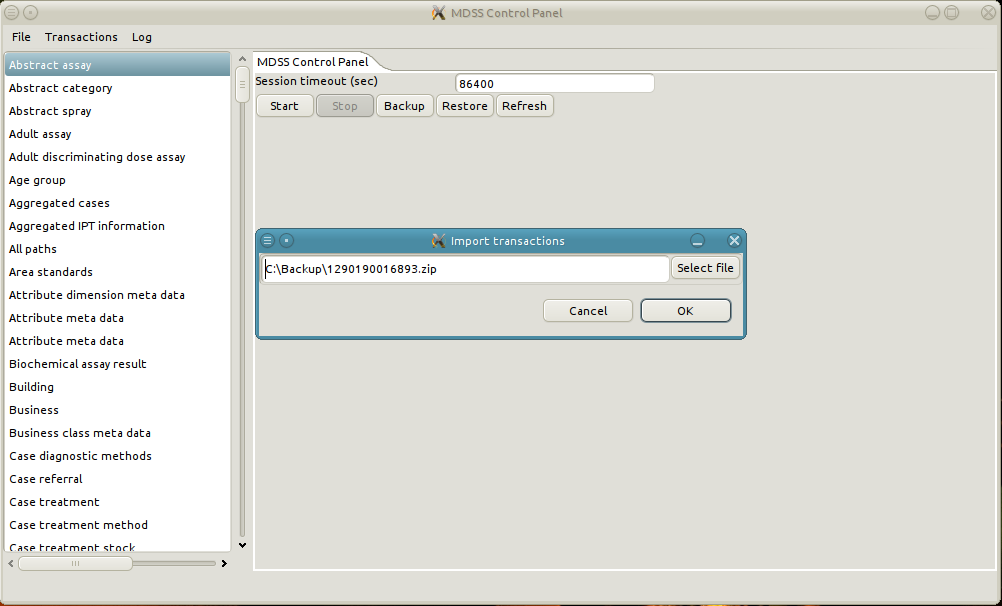
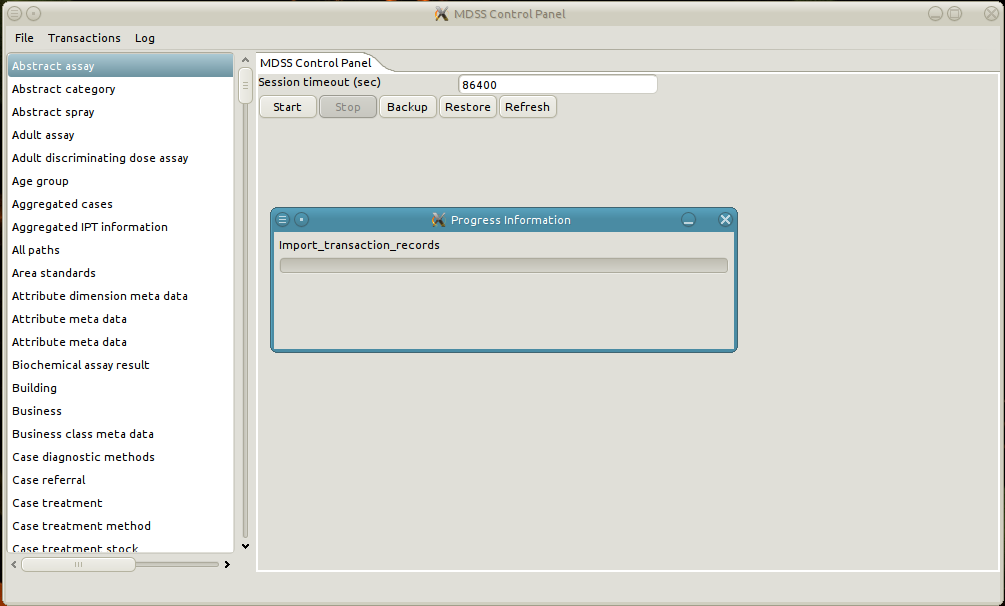
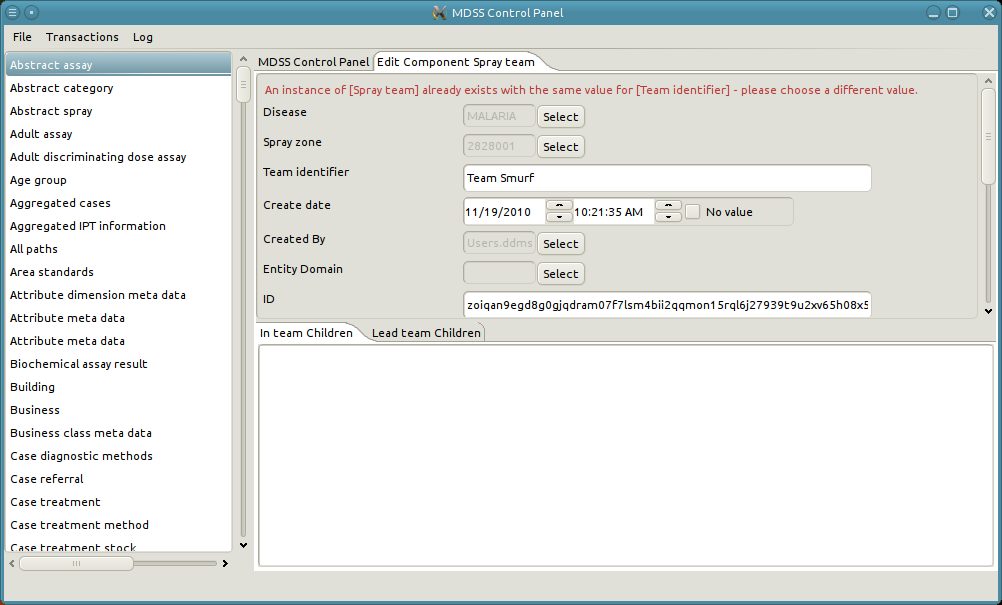
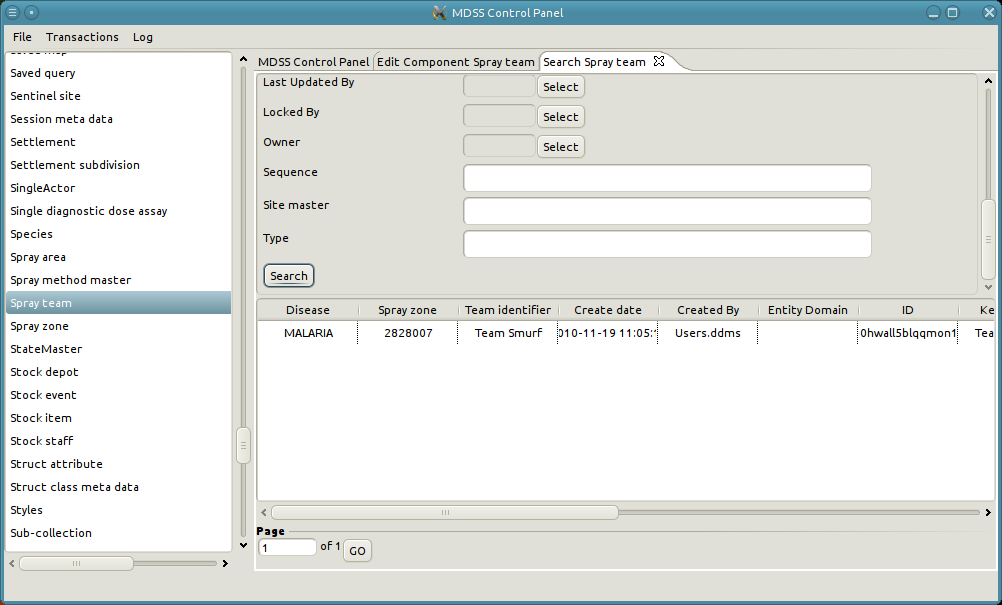
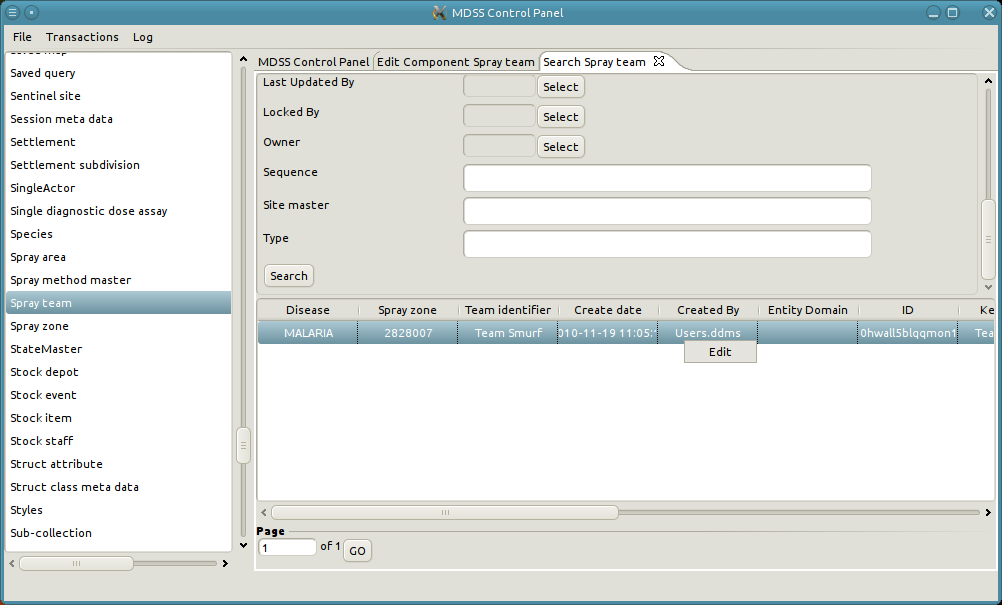
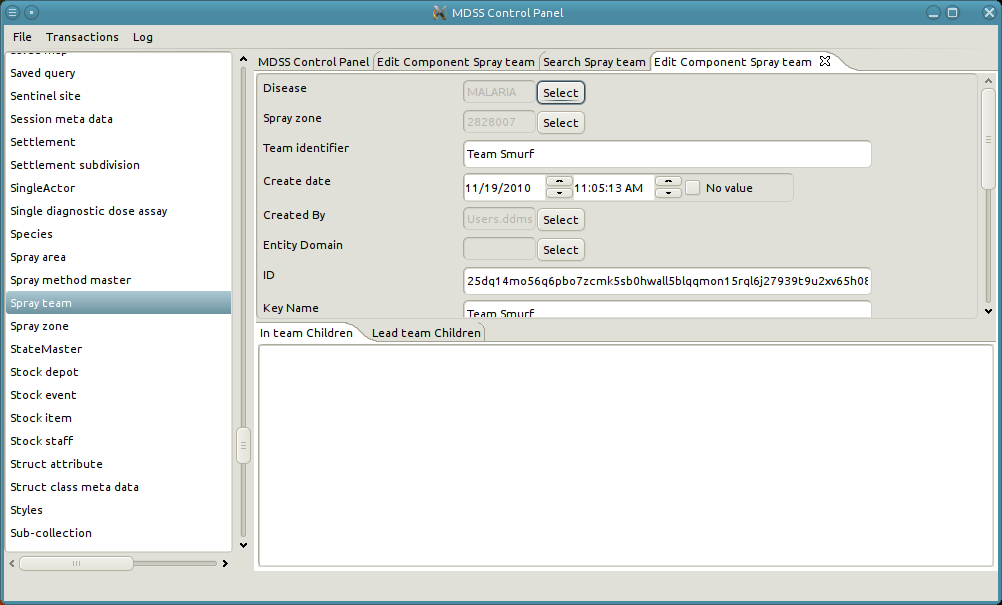
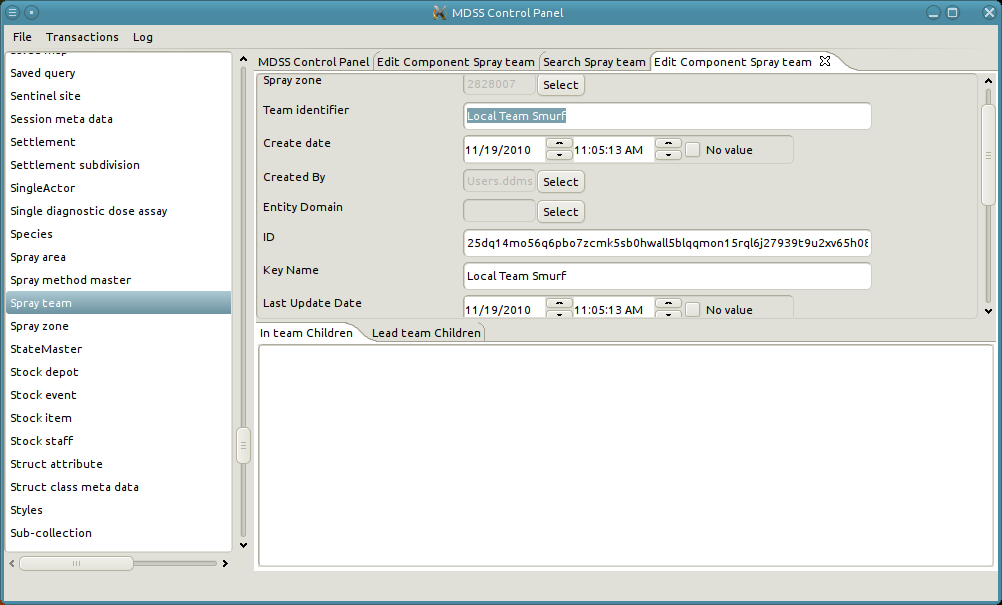
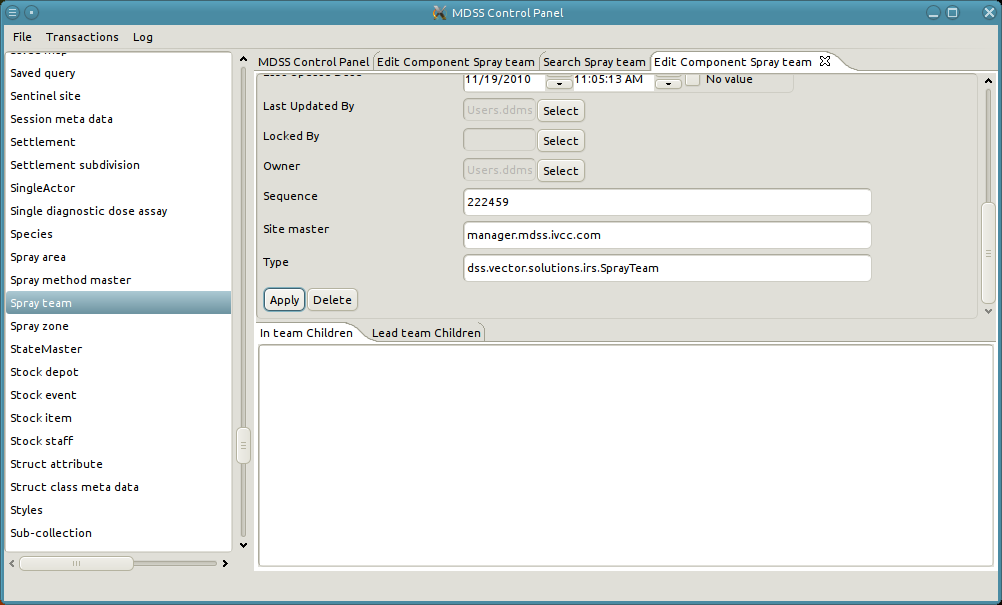
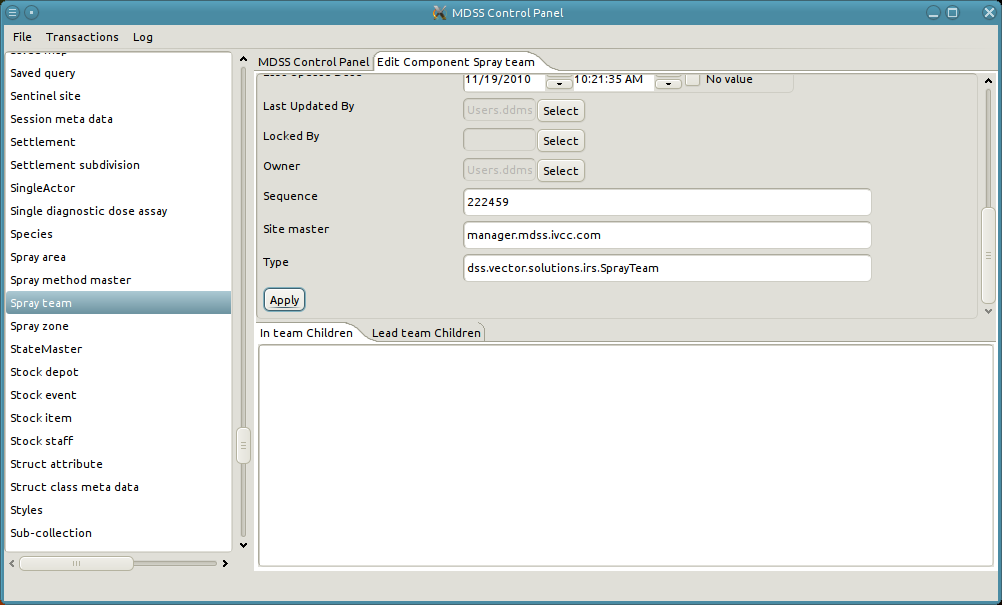
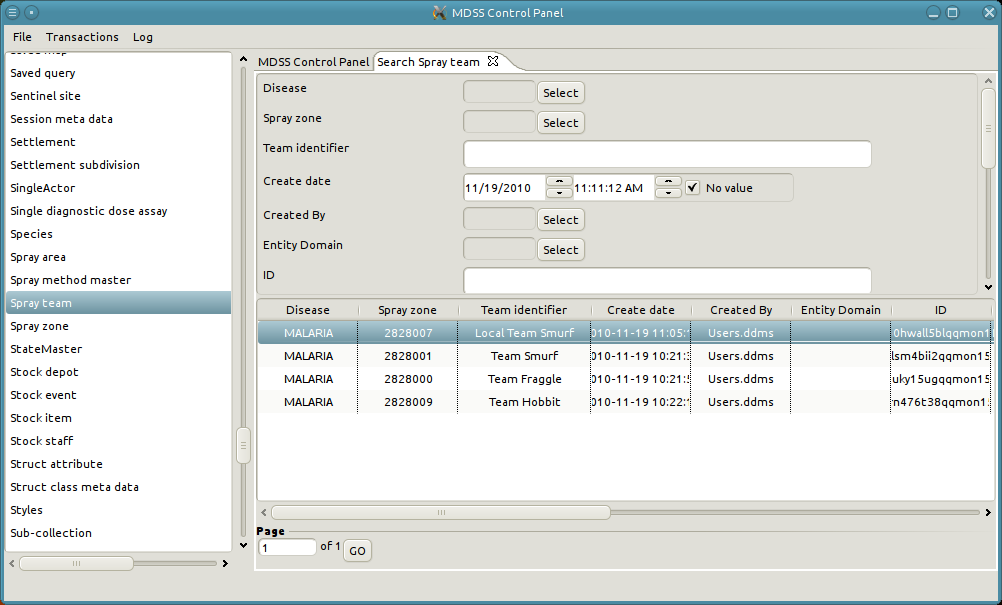
Many conflicts may occur during while importing transactions. When a conflict occurs the import halts until the conflict is resolved. It is impossible to continue the import without resolving the conflict. As such, the user will need to manually resolve each conflict. The basic use model for resolving a single conflict is as follows:

* The import pauses on conflicts
* A new tab opens viewing the conflicting object, with the error message at the top of the tab.
* Resolve the conflict with the techniques described in the “Basic CRUD Operations” section. The specific steps necessary for resolution differ on a case-by-case basis, and may involve modifying several objects or relationships.
* Assuming that the conflict has been resolved, clicking "Apply" on the conflicting object will automatically resume the import process. If the conflict has not been resolved then an error message will pop-up and the import will continue to wait until resolution.

## 

## An example of resolving a conflict

Let's examine the steps required to resolve a conflict when importing Spray team data from a different install.

1. Open the "Transaction" menu and click "Import Transaction"
2. Select the import file and click "Ok".
3. A pop-up occurs with the status of the import.
4. There is a conflict, the pop-up closes and a new tab is opened describing the conflict. In the example the conflict arises from a naming collision on the team identifier of the Spray team. Specifically, it is trying to import a Spray team called "Team Smurf", but there is already a Spray team of the same name on the local box.
5. To resolve this conflict we are going to change the name of the local "Team Smurf" to a new name. As such we need to search the spray teams on the local box.
6. Next, we select to edit the local "Team Smurf".
7. A new tab will appear with the details of the local spray team. 
8. We change the Team identifier and the Key Name of the local spray team to "Local Team Smurf"
9. Click apply to persist those changes. 
10. Finally, we click apply on the conflicting object to persists its changes and continue the import. The pop-up box with the status of the import will re-appear. 
11. When the import is finished the pop-up automatically closes. Finally if we re-search on the Spray teams in the system we should see the newly imported Spray teams.

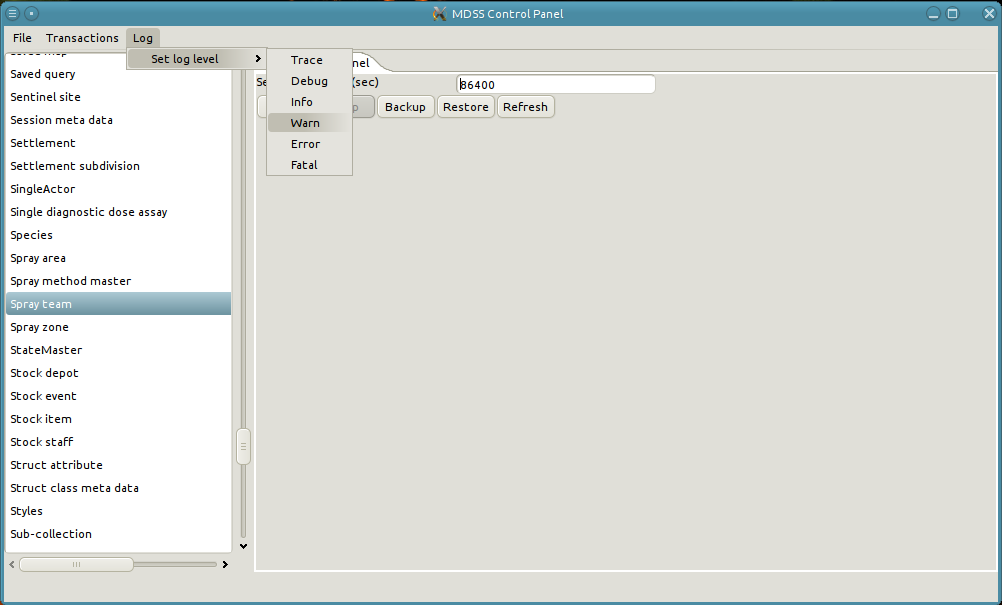
## Modify Log Level

DDMS features a logging system that records data about application usage and saves the data to files on the computer called logs. The log files include data ranging from normal usage to serious errors and are recorded as one log file per day. Technical support can then use the log files to troubleshoot problems that have occurred within DDMS.

There are six different log levels, each one representing a different level of verbosity and seriousness. At the lowest level is Trace, which records virtually all usage but has the tendency to make the system slow and consume large amounts of disk space. At the highest level is Fatal, which records only the most serious errors but provides the least amount of contextual information. The levels between Trace and Fatal, in order of least to most serious, are Debug, Info, Warn, and Error. The right level can be chosen to provide the most meaningful compromise between verbosity and seriousness for a given problem. By default the log level is Error.

When a problem occurs in DDMS that requires technical support, a user may want to recreate the problem but under a more verbose log level to provide extra contextual information. The user can change the log level with the following steps:

* Open the “Log” menu item
* Mouse over “Set Log Level”
* Click on the desired log level
* Restart the server
* Recreate the error
* Repeat the process to reset the log level back to Error.



The logs are located in the directory C:/MDSS/logs/ with the current log file as log.xml and older log files named relative to their date. A user may select the current log file, representing today's DDMS usage, or any other number of log files and send them to technical support\* for analysis.

\* This process has not been completely defined.