Abstract

This White Paper outlines the data requirements and functional definitions and actions that are required to undertake an import of Asset Records in to HighStone from an external source. Typically such data is sourced from flat spreadsheet files with multiple columns defining the attributes associated with the source asset records.

HighStone Asset Records are held in a single definition accommodating all Asset Types required in the contract database. Attribute definitions are defined and held as child records to the main assets records, with different classes of asset attributes defined for each class of asset.

This document is intended as a technical briefing on HighStone database definitions and usage - it is not written as a User Manual for the day to day operation of HighStone.

Document Summary

Title of Document	HighStone Import of Assets Records
Date of Issue	December 2011
Document Version	1.2

Claremont Controls Limited
Suite 4, Wansbeck Business Centre
Rotary Parkway
Ashington
Northumberland, United Kingdom
NE63 8QZ

UK Tel: (01670) 819000
Int Tel: +44 1670 819000

Web: www.claremontcontrols.co.uk
Email: info@claremontcontrols.co.uk

HighStone Asset Record Definitions

HighStone holds definitions for Asset Classes in a series of master data tables within the main contract database. There is a master definition for each class of Asset Record held in the system, with child Asset Attributes defined in a linked data table. Live asset records and associated child attributes are held in a set of data tables, with links back to the Master definitions of both Asset Class and Asset Attribute. Full details on the definition of the Asset Records are given under a separate White Paper title - a summary outline is given here.

Master Assets Class Definition

These definitions are held in [tblRoomComponents] - with a separate record entry for each class of asset. The main definitions in the table are given as follows:

RComponentId	Internal class numeric identifier - used internally by HighStone to identify the record.
RCItemCode	A unique tag or code that identifies each class of asset.
RCName	A text title / description for the asset class.
RCAutoGUID	A GUID definition for each asset class. This unique tag can be used as an alternative to the numeric identifier - and is particularly useful when identifying records across different contract database installations.

Master Assets Attribute Definition

This definition table holds definitions for all potential Asset Attributes that are associated with each Asset Class defined in [tblRoomComponents]. The table is [tblComponentAttributes] and each entry defines the data class of the defined attribute. Each master Asset Class may hold one or more Attribute Definitions, and the total number and type of Attributes will vary considerably between the different master classes.

CAttribId	Internal class numeric identifier - used internally by HighStone to identify the record.
CAComponentId	An identifier link to the parent Master Asset Class as defined in [tblRoomComponents].
CAName	A text title / description for the asset attribute definition.
CASortNo	Each record contains a sequential count entry that can be used to ensure attributes are always displayed in the required order (over-riding a display sequence based on the alphabetic sequence by title / description).
CAMandatory	A Boolean setting that determines if the asset attribute *must* be defined for each created Asset Record (generally all Asset Attributes will be classed as Mandatory).
CAPrimary	A Boolean setting that determines which of the attributes is held as the *primary* attribute - typically this will be an attribute holding an Asset Identification Number or similar.

CADataType	This classification defines the basic asset attribute data type - defining text, integer numeric, decimal numeric, date/time holding data types. Asset Attributes that are defined through look-up lists (pick lists) will be declared as an Integer Numeric data class. The available data types are defined in [ltbAttribDataTypes].
CAUnits	This classification defines the underlying Units that are associated with the Asset Attribute. Some entries will have no units (e.g. date style attributes), others will be simple No (Number/Quantity of Units), or a dimension in metres, temperature in degrees C, etc. The available data types are defined in [ltbAttribUnitTypes]. Asset Attributes that are defined through look-up lists (pick lists) will have a Units Type entry set to '17'.
CAPromptSelect	This entry is set for Asset Attributes that are defined through look-up lists (pick lists) and the value specified will identify the target prompt list definition to be applied as held in data table [tblAttribPrompts].
CAAutoGUID	A GUID identifier for each master asset attribute record. This unique tag can be used as an alternative to the numeric identifier - and is particularly useful when identifying records across different contract database installations.

Asset Attribute Lookup Prompts (Pick Lists)

The Asset Attribute lookups are defined in a pair of data tables: [tblAttribPrompts] lists all available prompt lists defined on the system, [tblAttribPromptItems] lists the individual prompt items associated with each list.

Usually a separate prompt list is defined for each Asset Attribute definition - although the same prompt list can be applied to more than one target record. Each record in [tblAttribPromptItems] includes:

APItemsId	Integer Record Identifier
APIAPId	Link identifier to parent table [tblAttribPrompts].
APIValue	Integer value that is associated to the prompt item. It is this value that is stored as the selected value in the Component Attribute record.
APITag	A text caption / tag that represents the prompt item.
APIDescrip	Full text description of prompt item. HighStone posts this text value as the 'general' value for the selected item in the Component Attribute record in order to simplify reporting.
APISortTag	A numeric sort that can be applied to define the display sequence of prompt list items - overriding alphabetic and value sorting.
APIAutoGUID	A GUID identifier for each prompt list item record.

The links between the data tables is given as follows:

[tblComponentAttributes].CAPromptSelect On [tblAttribPromps].AttribPromptId
[tblAttribPromps].AttribPromptId on [tblAttribPromptItems].APIAPId

Import of Asset Classes

The Asset Import function checks to see if the target Master Asset Class has already been declared and held in the target contract database before any import of Asset Records is activated. If the parent Asset Class record is not present, then it is created automatically taking the definition from the import definition file. This creation includes all required Asset Component Attributes, and associated prompt item pick lists.

Asset Record Definitions

Asset records are held in two live data tables:

[tblRoomsRC]	Header record for each Asset Record held on the contract database.
[tblRoomsCA]	The Component Attribute Records that are associated with each Asset Record held in [tblRoomsRC].

As indicated by the naming convention, each Asset must be classified against the location or geographic section within the network model at the time each record is created. In a Buildings / Floors / Rooms scenario, the reference will be linked to a Room or Area definition. In a highways scenario this link will be based on County / Route / Road Section definitions. This parent location definition is held in the live data table [tblRooms] with the link being made to [tblRooms].RoomId.

When Asset Records are imported asset records are added to these two data tables - parent Asset Records written to [tblRoomsRC] and all 'mandatory' / imported Component Attributes are created in the child table [tblRoomsCA].

Source Import Data

The nature of asset data import is that the source data is provided in a 'flat file' format - typically with a single line or record for each asset item, and the required asset definition values and attributes listed across the fields or columns of each entry. As each asset type will carry a different range of attributes, the record field layout will vary for each source asset type that is to be processed. The import action must be able to recognise the different asset values, and cater for the custom asset attribute definitions. The action must also convert the source 'flat record' definition of the source in to the strict parent record / child record structure that HighStone uses for the asset record and the associated attributes.

The first step in processing the source data is to bring the records in to a matching temporary data table that carries all record fields, and holds all source records. The precise structure of this temporary table is not crucial and will usually be a straight representation of the source data. It may be necessary to add extra data fields to this table definition to carry temporary data conversions and lookup values. The following requirements should be considered:

- The temporary table may hold all source data field values - even if they are not to be used by the subsequent import process.

- Each data field will require a name that is consistent with the host database - typically a field name should be a single word only (use the '_' character to replace spaces) and should not include other punctuation and special characters.
- The field data type created to hold the target values in the temporary table would normally match the values held (e.g. text fields, integer and floating point values, date values) but this is not essential. Provided the source data is held in a format that can be interpreted / converted within the subsequent data processing the actual field data type can be set to the most convenient for the import action.
- It is essential that the source data values are consistent across all data columns and through all records. If the source data is from a validated database table then this is unlikely to be an issue. However, where data is taken from a manually managed spreadsheet data inconsistencies can be expected. Mistyped entries such as characters within numeric values such as '123$56', '678c' will result in data conversion failures. Ad hoc entries such as '???', and 'Not known' against dimension fields will cause similar conversion failures. Date / time and unique identification GUID entries also need to be held in a valid format that can be recognised consistently by both the import and data conversion actions.

The means of creating a suitable temporary data table and uploading the source records may be completed by a number of different routes and users should use a method that best suits the facilities they have available. Options include:

- Manually create host data table and import the data records using SQL scripts.
- Use a 'data table' import function taking the data directly from a structured definition (e.g. import from an XML definition, or a data table held in a compatible application).
- Use source data import functions that can take data from spreadsheet or CSV data formats.

At the conclusion of this step the source data should be held in a temporary data table within the HighStone contract database, with all data records validated for consistency and true to the holding format.

Asset Data Value Conversion

In order to convert the source asset data from the temporary table created above a set of conversion rules must be created so that the import functions can process the data. This definition is held in a text file (.txt extension) and may be created manually using Notepad or similar. The file format is similar to the .ini file definitions with a series of [Section] entries created with surrounding square brackets, and definition values or full SQL scripts specified under each Section heading.

When an asset import is run HighStone prompts for the source definition file that is to be used, and then reads the data definitions and conversions from the file before processing the data. Each definition is limited to handling just one asset component type - if the source data table holds details for more than one asset type a separate definition type must be created for each type and repeat import actions undertaken.

The source definition must provide the following information:

- Source Data Definition - including the name of the asset source temporary data table, a SQL script to extract the source records, and the target asset component type that is being processed.
- Any data processing on the source temporary table that is needed to prepare data values *before* the import action. Ideally such conversions will have been made as part of the temporary table import above, but HighStone can apply additional processing at this stage if required.

- Define those source values that are to be cast to the parent Asset Component record. These values are written directly to table [tblRoomsRC].
- Define the Asset Component Attributes that relate to the target asset type. This will usually be a series of lookup definitions, but may also include characteristics such as Reference Numbers, Dimensions and Count attributes. These values are written to table [tblRoomsCA].
- Finally any data processing on the source temporary table that is needed to set data values *after* the import action. This can provide any tidy-up actions, or file processing such as handling associated photographs etc.

The following description gives an outline of these definitions by specific example. In setting these asset values in place it is important to apply the values to the appropriate target location. Particular reference should be made to:

- Data values that are held against the parent Asset Record (in [tblRoomsRC]) should *not* be written to Component Attribute child table. Such values include location details (cross-section location, grid reference details, etc.), date asset record created, note entry against asset. This ensures source data values are not held in two separate places within the Contract Database (leading to confusion and data maintenance of two sets of values).
- File references held in the source temporary table (e.g. file name of photograph images held against assert records) should not be imported as Component Attribute entries against asset records. HighStone handles images in a more general manner and holding a (single) filename against the asset record is not appropriate.
- Transient asset details such as 'Date of Survey', and 'Condition', should not be imported as Component Attribute entries against asset records. HighStone supports only a single data value against each Asset Component Attribute [not strictly true, but should be considered as such] and hence cannot hold a history of actions / conditions within this data structure. Generally this class of data should be held under the Asset Surveys data definitions and thereby form the first of a potential series of cyclic survey actions recorded against the asset records.

Import Definition

```
[ImportDefn]
ImportTableName = SignsSurvey2011
ImportSQL = SELECT * FROM SignsSurvey2011 ORDER BY ID;
Component = SSG
```

ImportTableName	The name of the temporary import table within the contract database holding the source records.
ImportSQL	The SQL Select script that is used to extract the target records from the temporary data table. Here *all* records are selected for import, but it may be necessary to apply a record filter if the source table holds records for more than one asset type.
Component	The target Asset Type component that is being imported - identified by the short asset type code. If the target asset code type does not already exist in the contract database (in [tblRoomComponents]) then a new record will be created automatically - including the Asset Component Attributes.

Pre Process SQL Scripts

```
[PRESQL01]
Update A19SignsSurvey2011 Set Asset_GUID = SubString(Asset_Identifier, 1, 13) +
'-' + SubString(Asset_Identifier, 14, 9) + '-' + SubString(Asset_Identifier, 23,
4) + SubString(Asset_Identifier, 28, 8);

[PRESQL02]
BLOCK RECAST SELECT ID, LOWER(Asset_GUID) As ITEMGUID From A19SignsSurvey2011
Apply As Update A19SignsSurvey2011 Set Asset_True_GUID = <ITEMGUID> Where ID =
<ID>;
```

[PRESQLnn]	The section heading must conform to the style [PRESQL01], [PRESQL02], [PRESQL03] and so on. HighStone takes the scripts in order, and stops as soon as a break in the sequential numeric suffix is met.
(SQL Script)	The SQL Select script that is to be applied. This may be entered on one or more lines within the section of the import definition.

The example above is processing the import data to cater for a mismatch in the format supplied for the Asset GUID identifier. The first script creates a new version of the source text value (putting the values and '-' characters in the correct place), the second scripts then converts the value to a true GUID value within the target database table. This action is applied to support record cross-referencing under the Post-SQL processing.

Cast Asset Characteristic Values

```
[CompAttrib01]
CAImportName = Easting
RCImportName = RoomsRCGridRefP1E

[CompAttrib02]
CAImportName = Northing
RCImportName = RoomsRCGridRefP1N

[CompAttrib03]
CAImportName = Date
RCImportName = RoomsRCCreated

[CompAttrib04]
CAImportName = Asset_GUID
RCImportName = RoomsRCAutoGUID

[CompAttrib05]
CAImportName = Asset_Room_Id
RCImportName = RoomsRCRoomId
```

[CompAttribnn]	The section heading must conform to the style [CompAttrib01], [CompAttrib02], [CompAttrib03] and so on. Each section processes a single source value and stores the value against the asset record. HighStone takes the sections in order, and stops as soon as a break in the sequential numeric suffix is met.

CAImportName	This is the field or column name in *source* temporary data table that is to be processed.
RCImportName	This is the field or column name in the HighStone table [tblRoomsRC] where the value is to be written and saved. The source values must be held in a format that is compatible with the target field type.

Asset Component Attribute Values (Basic)

```
[CompAttrib01]
CAImportName = Identity_Code_SG
CAName = Identity Code (SG)
CAShortName = Identity Code
CASortNo = 2
CAMandatory = True
CAPrimary = True
CADataType = 0 (Text)
CAUnitsType = 0 (None)
CAPromptSelect = 0
CADataFieldType =
CAClientOptional = 4 (Signs)

...

[CompAttrib18]
CAImportName = Width_SG
CAName = Width (SG)
CAShortName = Width
CASortNo = 16
CAMandatory = True
CAPrimary = False
CADataType = 1 (Integer)
CAUnitsType = 16 (mm)
CAPromptSelect = 0
CADataFieldType =
CAClientOptional = 4 (Signs)

[CompAttrib19]
CAImportName = Heigh_SG
CAName = Height(SG)
CAShortName = Height
CASortNo = 17
CAMandatory = True
CAPrimary = False
CADataType = 1 (Integer)
CAUnitsType = 16 (mm)
CAPromptSelect = 0
CADataFieldType =
CAClientOptional = 4 (Signs)
```

[CompAttribnn]	The section heading must conform to the style [CompAttrib01], [CompAttrib02], [CompAttrib03] and so on. Each section processes a single source value and stores the value against the asset record. HighStone takes the sections in order, and stops as soon as a break in the sequential numeric suffix is met.

CAImportName	This is the field or column name in *source* temporary data table that is to be processed.
CAName	This is the Component Attribute name as defined in [tblRoomComponents] and is used by HighStone to identify the target attribute type to be referenced.
CAShortName *	The alternative shorter name for the Component Attribute definition.
CASortNo *	A sequential numeric count that defines the order the Component Attributes should be displayed in (this overrides an alphabetic sorting on the Attribute Name).
CAMandatory *	Defines in the Attribute Entry is either mandatory (True) or optional (False). It is recommended the mandatory option is applied for all entries.
CAPrimary *	Identifies that Component Attribute that is the primary attribute (True). Generally this will be set against a Reference Number of Identification Code attribute. Only *one* attribute may be classed as Primary (True), all other attributes must be set as (False).
CADataType *	Defines the Component Attribute data class - this is set as: 0 = Text entry 1 = Integer Numeric Entry 2 = Float / Decimal Numeric Entry 3 = Date / Time Entry 4 = True / False (Boolean) Value 5 = GUID Data Entry The data type options are listed in database table [ltbAttribDataTypes].
CAUnitsType *	Defines the type of units that are applied for Integer Numeric and Float Numeric attribute types. Entries must conform to the options listed in database table [ltbAttribUnitTypes].
CAPromptSelect *	Set as zero (0) for all Basic attribute definitions.
CADataFieldType *	
CAClientOptional *	

The items marked as * are applied only when HighStone has to create a new Asset Attribute definition in the contract database. If the Attribute Definition already exists in the host database, these values are not used as part of the import definition.

The examples above take the basic dimensions of a sign asset (in millimetres) and saves the values as two separate Component Attribute entries in table [tblRoomsRC]. Note the misspelling of the source name as 'Heigh_SG' - this is set to cater for an error in the prompt definitions used when the data was collected in the field as it should really have read 'Height_SG'. Such transposition in source data is not uncommon in this sort of data processing and demonstrates just how important it is that all definitions do correspond at all stages of the import process.

Asset Component Attribute Values (Pick Lists)

```
[CompAttrib04]
CAImportName = Sign_VRS_Type
CAName = VRS Type (SG)
CAShortName = VRS Type
CASortNo = 4
CAMandatory = True
CAPrimary = False
CADataType = 1 (Integer)
CAUnitsType = 17 (Option)
CAPromptSelect = 0 (Number)
CADataFieldType =
CAClientOptional = 4 (Signs)
CAPromptItem01 = Not Primary Sign
CAPromptItem02 = None
CAPromptItem03 = Steel Barrier
CAPromptItem04 = Concrete Barrier

[CompAttrib05]
CAImportName = Ownership
CAName = Ownership (SG)
CAShortName = Ownership
CASortNo = 6
CAMandatory = True
CAPrimary = False
CADataType = 1 (Integer)
CAUnitsType = 17 (Option)
CAPromptSelect = 0 (Number)
CADataFieldType =
CAClientOptional = 4 (Signs)
CAPromptItem01 = Highways Agency
CAPromptItem02 = County
CAPromptItem03 = District
CAPromptItem04 = Unknown
```

[CompAttrib*nn*]	As above
CAImportName	As above
CAName	As above
CAShortName *	As above
CASortNo *	As above
CAMandatory *	As above
CAPrimary *	As above
CADataType *	This must be set to the value of '1' (Integer Numeric values). This reflects the lookup classification of holding a data value of 1 for the first list item, 2 for the second, 3 for the third, etc.
CAUnitsType *	Must be set to a value of '17' (Option). This value setting triggers the use of a prompt or pick list of valid entries.
CAPromptSelect *	As above
CADataFieldType *	As above
CAClientOptional *	As above

CAPromptItem*nn* *	This series of entries must be numbered in sequence with the suffix 01, 02, 03, 04 etc. The associated text caption is the pick list prompt text and *must* match exactly the corresponding entries in the source data table. The lookup is not case sensitive but use of intermediate space characters and punctuation must match the entries. HighStone stores lookup entries with integer numeric values equal to the sequential count number - 1, 2, 3, 4 etc.

Asset Component Attribute look-up definitions are held in database tables [tblAttribPrompts] and [tblAttribPromptItems].

The above examples show just two of a long list of lookup class asset component attribute values (actually around 35 in the full definition). Note how the lookup entries include default options of 'Not Primary Sign' (indicates 'No Applicable'), 'None' and 'Unknown'. These prompts provide an option to the user at all times - so leaving a selection blank or overtyping with '?' is not allowed and full integrity of the collected data is preserved.

Post Process SQL Scripts

```
[POSTSQL01]
BLOCK RECAST Select Asset_GUID As ItemGUID, XSPId As ENTRYID, XSPTagCode As
ENTRYTAG From A19SignsSurvey2011 Inner Join ltbXSP On
A19SignsSurvey2011.Asset_XSP_SG = ltbXSP.XSPTagCode ORDER BY ID APPLY AS Update
tblRoomsRC Set RoomsRCXRefCode = <ENTRYTAG>, RoomsRCXRefCodeId = <ENTRYID> Where
RoomsRCAutoGUID = <ITEMGUID>;

[POSTSQL02]
Select 10271 As OBJECTCLASS, RoomsRCId As OBJECTID, 'Z:\Client Data\Signs Survey
Nov 2011\Signs Inventory 2011\11 Photos\' As SOURCEPATH, Photograph As SOURCEFILE
From SignsSurvey2011 Inner Join tblRoomsRC On SignsSurvey2011.Asset_True_GUID =
tblRoomsRC.RoomsRCAutoGUID Order By ID;
```

[POSTSQLnn]	The section heading must conform to the style [POSTSQL01], [POSTSQL02], [POSTSQL03] and so on. HighStone takes the scripts in order, and stops as soon as a break in the sequential numeric suffix is met.
(SQL Script)	The SQL Select script that is to be applied. This may be entered on one or more lines within the section of the import definition.

The example above is processing the import data to cater for data values that cannot be imported directly through the Attribute Definitions. In the first script the asset location fields are set on [tblRoomsRC] from the survey data - in this instance being referenced through the entries held in the contract database in table [ltbXSP].

The second SQL script processes a series of photograph images taken for each asset record - and relocates the image files by moving them from the holding folder on a server and placing in the associated Asset Object Folder for each asset record.

In both cases it is necessary to link the source data records in the temporary table with the asset record as saved by the import process - which is where the GUID Identifier held on the data records is used. This reference is the only way to create a link between the two sets of data as the Asset

Identifier raised on [tblRoomsRC] is created by the contract database and not HighStone and cannot be used for this purpose.

Note about Asset Definition and Import

As part of the Asset Import action HighStone will check to see if the Asset Type being imported is already defined within the Contract Database. To simplify the import process, HighStone supports the action of creating a missing asset type definition automatically from the import definition. This includes creating the parent asset type entry in [tblRoomComponents], and the attribute definitions in the child table [tblComponentAttributes] (including the prompt list definitions).

In applying this facility the following aspects must be remembered and followed in order to preserve data integrity of the asset records:

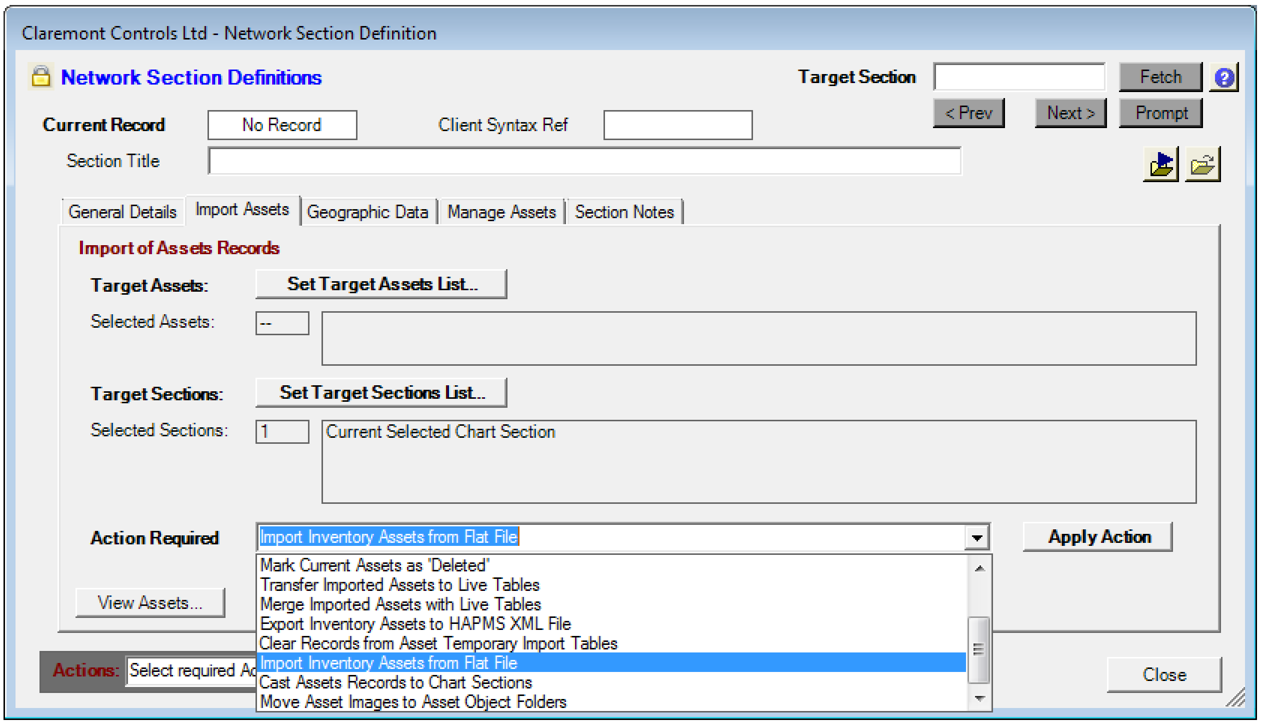
- If the import does find a matching entry for the target Asset Type, then that existing definition is adopted as defined in the contract database. HighStone *does not* check the validity of the definitions against those given in import definition.
- If the import definition reflects any changes to existing asset characteristic records, then the changes will not be picked up by the import and the data conversion may not match the values as supplied.
- If the asset type import definition *is changed* then it will be necessary to clear out the old asset type definitions manually before re-running the asset import action again. **This definition change cannot be performed if the contract database already holds asset records of the given type previously imported under an original definition - significant data corruption will result if definitions are changed when existing data is held.**
- If asset attribute characteristics change for subsequent imports, then the new definitions must be added to existing asset definitions in a consistent manner before any asset record import is undertaken.
- Asset Attribute prompt lists (or pick lists) are more likely to change over time but care is needed in reflecting changes to definitions in an existing contract database. It is usually possible to append new options to the pick lists without too much difficulty. However re-ordering existing prompt lists, or inserting new entries part way down the list of options, should be avoided as existing records in the database may be left referenced to the wrong definition record.
- This import asset action is best suited to the import of asset records from scratch - importing records that are not already held in the contract database. This import action is not suited to the merging of new survey information with existing asset records held in the contract database - alternative asset management functions should be used for this task.
- Users may consider using this import action as a 'clear out and replace' facility - removing all existing asset records from the database before undertaking the import to re-populate the asset model. This is strongly discouraged for two reasons:
    - This approach destroys any ability to record the maintenance history against individual asset records as the asset base is constantly changing.
    - The contract database will become un-necessarily large as each asset import will *add* the new records to those previously held (HighStone cannot actually *delete* old asset records at any time - they are simply flagged as 'no longer active').

Asset Import via Temporary Data Tables

The original implementation of Asset Import records being posted directly to the live data tables ([tblRoomsRC] and [tblRoomsCA]) has been discontinued in live HighStone systems and the use of temporary intermediate asset record tables is now used ([tbsImportComponents] and [tbsImportCompAttribs]).

This change of approach was instigated because of the generally poor level of data accuracy that is found in source data records - requiring a number of attempts and iterations on the source data before it is brought to a required level of accuracy and integrity. Undertaking such tasks on 'live' data tables carries risks of data corruption whilst the use of temporary data tables allows source data to be imported, processed fully and checked before the records are moved across in to the 'live' data tables. The user is free to clear and re-run base imports as often as is required to achieve a valid data preparation. The mechanism used for this is the same process as that used for importing Asset Records form HAPMS Systems using XML definitions.

The HighStone asset import utility is held under the Network Chart Section display panel, on the Import Assets panel tab, select the **Import Inventory Assets from Flat File** action and click on **Apply Action**. The facility is only offered to Administrator Class users as the use of the function is not recommended for general use. The Import Action will prompt for the source Asset Data Value Conversion definition file as outlined above.


Claremont Controls Ltd - Network Section Definition

Network Section Definitions

Target Section:
Current Record: No Record
Client Syntax Ref:
Section Title:

Import of Assets Records

Target Assets:
Selected Assets: --

Target Sections:
Selected Sections: 1 — Current Selected Chart Section

Action Required: Import Inventory Assets from Flat File

- Mark Current Assets as 'Deleted'
- Transfer Imported Assets to Live Tables
- Merge Imported Assets with Live Tables
- Export Inventory Assets to HAPMS XML File
- Clear Records from Asset Temporary Import Tables
- Import Inventory Assets from Flat File
- Cast Assets Records to Chart Sections
- Move Asset Images to Asset Object Folders

Actions: Select required Ac

The target data table [tbsImportComponents] may require additional configuration if it is to carry additional data record values for the target file [tblRoomsRC]. Such fields are not automatically defined in the temporary data table but may be added as required. It is recommended that additional fields (for example, to hold Grid Reference details, or Notes) are added to the data table [] using the same field name and characteristics as in the target live table. HighStone will take these values across automatically when asset records are 'moved' from the temporary data table to the 'live' tables.

Note that whilst the Asset Records are held in the temporary data tables - the Asset Type Definition is *always* created in the 'live' table definitions [tblRoomComponents] and [tblComponentAttributes].

Users should be aware of this, and ensure the Asset Data Value Conversion definition file is correct and valid before it is run as part of the import process.

Highways Asset Location Classification

On Highways Contracts, geographic classification for asset data is made under a network model of Chart Sections, with each section representing a length on the network road. Asset location within each section is specified by the length along the section from the start point (usually known as 'chainage'), and the cross-sectional position at the given point. Modern data capture techniques are moving to the use of GPS tools and holding positional information as OS Grid Coordinates (Eastings and Northings). Whilst the use of GPS location information is useful classification under the Chart Section definitions is usually required:

- Client master database definitions are still based on the Chart Section model and where data records have to be maintained in this external system, **classification under this model is essential** - not optional.
- Survey management is usually based around driven routes and the derivation of surveys and routine maintenance is much easier through the Chart Section models, compared to base location co-ordinates.

Therefore, it is usual that asset records must be located within the Chart Section model as part of the data import action. The most reliable method of classification is to include this as part of the base survey data, make the classification in the field and import the values directly. Where the survey data has been prepared *without* the Chart Section classification it will be necessary to apply a method of conversion to determine the location details from the grid reference information.

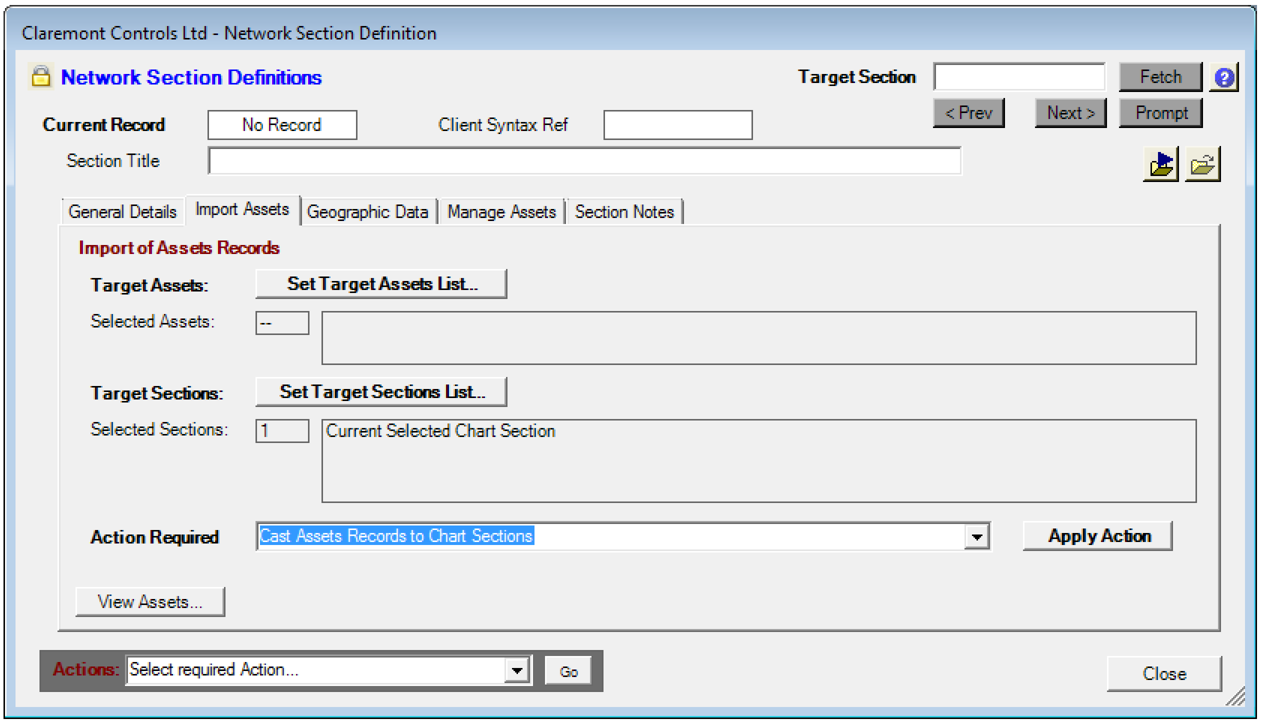
HighStone includes a facility that will reverse reference Grid Coordinates back to Chart Sections and this may be applied to source data values. Due to complexities in the underlying Network Model this conversion process rarely provides a 100% success rate and a certain level of iteration and manual intervention can be expected. Locations around roundabouts, and interchanges with multiple slip lanes do not lend themselves to automatic classification and results at these points should always be checked.

In order to apply the geographic conversion the following information is required:

- Chart Sections must be classified by Route Number (road number), carriageway type (main, slip, roundabout or layby), and carriageway direction.
- Chart Sections must hold grid reference co-ordinates at both the start and end of each road length (as a minimum).
- If Chart Sections hold more detailed co-ordinate details with points defined along the length of each section, then HighStone will use this information in making the conversion.
- Each asset record must be held including the grid reference co-ordinate details.
- The position of each asset record in terms of Route Number, carriageway type and carriageway direction must be held. This is essential for classification at interchanges where a basic grid co-ordinate could relate to a number of adjacent or parallel Chart Sections.
- On dual carriageway routes, assets within the central reserve are classified against Chart Sections running in a 'nominated' direction - even when the asset faces the opposite carriageway direction.

As noted above, this conversion is likely to prove adequate over sections of the network where the carriageway is straight and does not involve interchanges and roundabouts. For the more complex sections of the network a more careful study may be required.

The HighStone conversion utility is held under the Network Chart Section display panel, on the Import Assets panel tab, select the **Cast Assets Records to Chart Section** action and click on **Apply Action**. The facility is only offered to Administrator Class users as the use of the function is not recommended for general use. Although the facility is described here for use on Network Assets, the conversion may be applied to the records of other Object Records in HighStone.



Current Record: No Record
Client Syntax Ref:
Section Title:

Import of Assets Records

Target Assets:
Selected Assets: --

Target Sections:
Selected Sections: 1 — Current Selected Chart Section

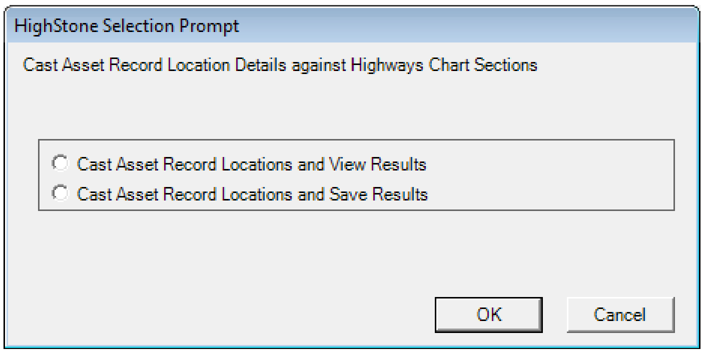
Action Required: Cast Assets Records to Chart Sections

Actions: Select required Action...

The basic steps of operation are as follows:

- Select the Convert option on the pull down options
- HighStone will prompt for a *SELECT* SQL script that is applied to extract a list of Asset Records that are to be submitted to the conversion process. This SQL script *must* return a series of data fields giving the necessary source values in defined field names.
- A prompt for a second *UPDATE* SQL script is given that is used by HighStone to save the converted values back to the Contract Database. This script is optional and if it is not given then HighStone will display the results of the coordinate conversion in a grid panel display for manual checking.

HighStone gives a prompt as to the action required:



Cast Asset Record Location Details against Highways Chart Sections

- Cast Asset Record Locations and View Results
- Cast Asset Record Locations and Save Results

If necessary, the conversion of positional details may be applied in a series of individual steps rather than a single pass across all source records. This is particularly relevant for areas where the network layout is complex, or the source data is lacking in precise definitions.

Asset Location Conversion Source SQL Script

This is a *SELECT* SQL script that extracts a list of Asset Records that are to be submitted to the conversion process. The following is an example:

```
Select ID As ItemID, Asset_True_GUID As ItemGUID, SignsSurvey2011.Route,
SignsSurvey2011.Route_Type_SG, SignsSurvey2011.Route_Dir_SG,
ltbCwayDirection.DirectionId As ItemDirnId, ltbCwaySlip.CSIdentifier As
ItemCwayTypeId, ltbRoadNumbers.RoadNo As ItemRoadId, Easting As ItemEast,
Northing As ItemNorth, 0 As ItemSectionId, 'Section' As ItemSectionRef, 0 As
ItemChainageSet, 0 As ItemChainage, 0 As ItemXSPLocn From SignsSurvey2011 Inner
Join ltbRoadNumbers On SignsSurvey2011.Route = ltbRoadNumbers.RoadNo Left Join
ltbCwaySlip On SUBSTRING(SignsSurvey2011.Route_Type_SG,1,1) =
SUBSTRING(ltbCwaySlip.CSTagCaption,1,1) Left Join ltbCwayDirection On
SUBSTRING(SignsSurvey2011.Route_Dir_SG,1,1) =
SUBSTRING(ltbCwayDirection.DirectionDesc,1,1) Order By ID;
```

The fields included in the SQL Script may include any values that are useful in a validation display (given when no *UPDATE* SQL script is specified), but the following values are mandatory:

ItemEast	Easting Grid Reference for the target record.
ItemNorth	Northing Grid Reference for the target record.
ItemRoadId	Identifier for the Route Number (road number) on which the asset is located. This uses the same identifier as that adopted for the network chart sections - usually data table [ltbRoadNumbers].
ItemDirnId	Identifier for the direction of the carriageway on which the asset is located. This uses the same identifier as that adopted for the direction on network chart sections - usually data table [ltbCwayDirection].
ItemCwayTypeId	Identifier for the carriageway type on which the asset is located. This uses the same identifier as that adopted for the carriageway type on network chart sections - usually data table [ltbCwaySlip].
ItemId	The numeric Identifier for the record that is to be used to hold the results of the conversion.
ItemGUID	The GUID Unique Identifier for the record that is to be used to hold the results of the conversion - this may be used as an alternative to the numeric identifier above.

In the SQL example given above, the source definitions for carriageway direction, and carriageway type, do not match the tag captions for the entries in the defining data tables. Consequently the inter-table joins are defined using just the initial letters of each entry - rather than simple joins based on the full word captions.

The following data field names are used by HighStone to post the resulting conversion values back in to the source SQL values. These result values can are then reviewed if the results set is displayed on a grid display.

ItemSectionId	Identifier for the Chart Section to which the source record location is allocated.

ItemSectionRef	Gives the identification code for the Chart Section to which the source record location is allocated.
ItemChainageSet	This value is set to 'True' if the conversion gives a valid Chainage value to the location.
ItemChainage	The calculated intermediate point within the chart section where the location is found - the Chainage value in metres.
ItemXSPLocn	The position of the location point in respect to the carriageway direction - either Left or Right. The value is referenced to the data table [ltbXSPLocn].

Asset Location Conversion Update SQL Script

When specified, this SQL Script is used by HighStone to store the results of the location conversions back to the holding data table. An example is:

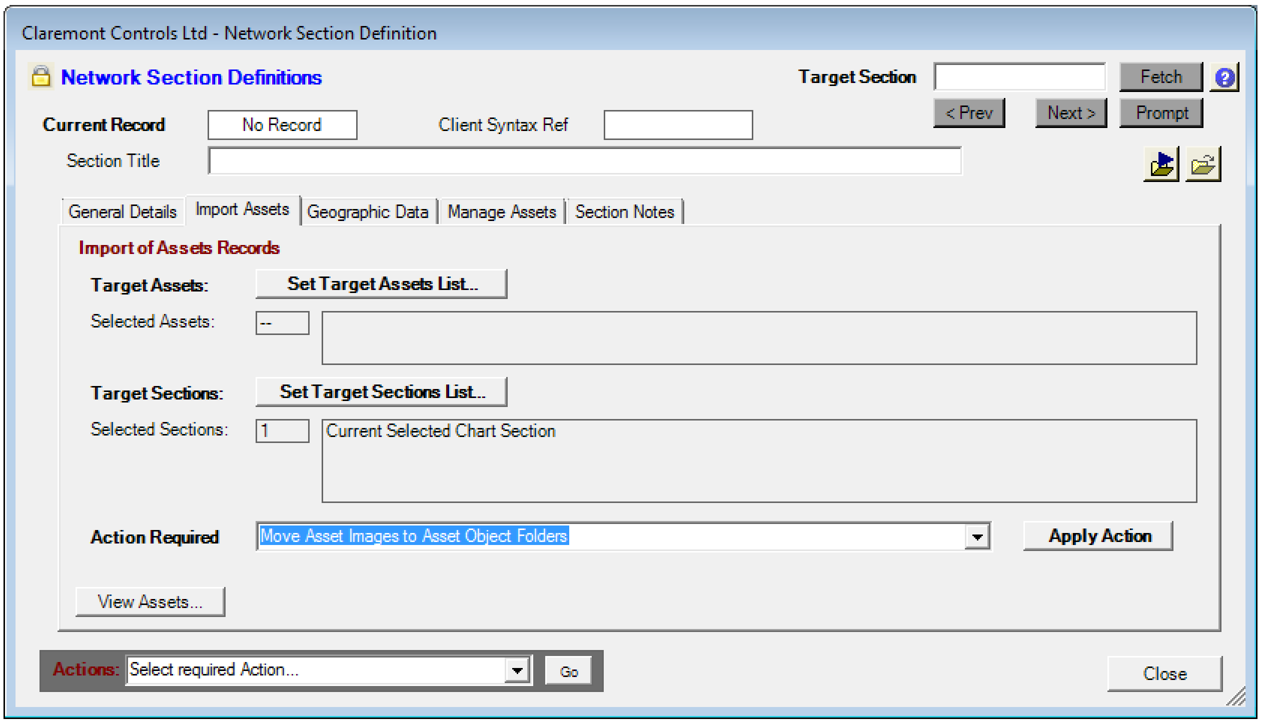
```
Update SignsSurvey2011 Set Asset_Section_Id = <ITEMSECTIONID>, Asset_Section_Ref
= <ITEMSECTIONREF>, Asset_Chainage = <ITEMCHAINAGE> Where ID = <ITEMID>;
```

The substitution of the results in to the script is made through the field names in the initial SELECT SQL script (character case is not important). The UPDATE SQL script is applied for each data record where a successful conversion has been made.

Survey Images Upload

Many surveys will collect photographic images of the assets surveyed, collecting the images as .jpg files on the local device. Survey data will hold the name of the image file as part of the assert record and this data can be used by HighStone to transfer the taken images to the Object Folders under the imported asset records.

The HighStone image import utility is held under the Network Chart Section display panel, on the Import Assets panel tab, select the **Move Asset Images to Asset Object Folders** action and click on **Apply Action**. The facility is only offered to Administrator Class users as the use of the function is not recommended for general use. Although the facility is described here for use on Network Assets, the conversion may be applied to the records of other Object Records in HighStone.



Claremont Controls Ltd - Network Section Definition

Network Section Definitions

Target Section:

Current Record: No Record

Client Syntax Ref:

Section Title:

Import of Assets Records

Target Assets: Set Target Assets List...

Selected Assets: --

Target Sections: Set Target Sections List...

Selected Sections: 1 | Current Selected Chart Section

Action Required: Move Asset Images to Asset Object Folders

Actions: Select required Action...

This function can only be run *after* the Asset Records have been imported and moved from the temporary in to the live data tables. The reference to the Object Record is made through the record numeric Identifier that is used by HighStone to provide the user reference to each record. This reference does not exist within the Contract Database until the record import process is completed.

HighStone will prompt for a *SELECT* SQL script that extracts a list of Asset Record Identifiers and filenames that are to be submitted to the file re-location process. The following is an example:

```
Select 10271 As OBJECTCLASS, RoomsRCId As OBJECTID, 'Z:\Client Data\Signs Survey
Nov 2011\Signs Inventory 2011\11 Photos\' As SOURCEPATH, Photograph As SOURCEFILE
From SignsSurvey2011 Inner Join tblRoomsRC On SignsSurvey2011.Asset_True_GUID =
tblRoomsRC.RoomsRCAutoGUID Order By ID;
```

The SQL Script must include the following field values. Some of these are specified by direct reference rather than extraction from a source data table.

OBJECTCLASS	Identifies the Object Class that is being processed. The numeric entry 10271 references the Asset Record Object Class.
OBJECTID	Gives the numeric identification for the target Object Record. For Asset Records this is taken as the internal numeric identifier held on [tblRoomsRC].
OBJECTTAG	Gives the non-numeric identification for the target Object Record. For Asset Records this value is not used.
SOURCEPATH	The source path where the image files are held. Usually this will be located on a local server and all image files from a single survey held in a single location.
SOURCEFILE	The filename for the target image file - including file extension (usually .jpg).

The above example is applied to the loading of photographs for Assets Records, but the same routine may be used to upload any document types for any Object Class records. Change the OBJECTCLASS value to reflect the required HighStone internal object class, and set the OBJECTID to the individual record identifier, or use the OBJECTTAG value where an object is identified by a non-numeric tag. Note that HighStone uses the *user facing* object record identifier for all Object Folder manipulation which does not always correspond to the relevant record identifier.

On all Live HighStone configurations this action will *move* source image files from their original location and place them in the target folder areas. If HighStone is run in Debug mode (often used in HighStone Development) the file action applied is just a *copy* action - the source files are not removed after a successful file move action.