Tietohippu Oy

Collecting user-defined data from harvesting operations

Outcomes and actions in the development of prototype application

Timo Komulainen
30.10.2018

Author's address:
Tietohippu Oy, Teknologiapuisto PL 107, 87400 Kajaani, Finland
Email: timo.komulainen@tietohippu.fi

Northern Periphery and Arctic Programme 2014–2020

EUROPEAN UNION
Investing in your future
European Regional Development Fund

Contents

This report has been written as part of Tietohippu Ltd's work in the FOBIA project funded by the Northern Periphery and Arctic programme. The report consists of the summary of the UDI/UDD protocol defined in the StanForD (standard for forest machine data and communication) and documentation how to apply it to collecting of user-defined data for business management purposes. In addition to the Finnish wood harvesting industry, the method can be used in other countries as well. The UDI/UDD application developed in the Fobia project is based on the protocol above. This report includes a description of the protocol contents and a summary of the implementation of the UDI/UDD protocol in the FOBIA project (Deliverable T2.1.2).

1. Background

There is need to collect additional work and material information from the harvesting operations for business management purposes (e.g. invoicing). Discussion about the method for data collection started already in 2011, when the Trade Association of Finnish Forestry and Earth Moving Contractors established a working group composing of the representatives of the wood harvesting entrepreneurs and the major forest machine manufacturers. The UDI/UDD (user-defined data instructions / user-defined data) protocol defined in the StanForD 2010 standard enables the collection of such data in a uniform way.

Uniform message and data structures for the transfer of free-form user-specific data to machines and from machines were needed due to the problems in managing of differing ratings and coding in the standard. The problems were caused by various practices in different countries. There are also company- and entrepreneur-specific information needs (the types of work and material to be invoiced, billing basis and unit, customer lists, operator's working time tracking codes). Table format was considered a flexible way of transmitting and reporting such information. An earlier proposal for Invoice Specification message was withdrawn in 2011, and therefore a new user-defined data message and its data structures were decided to add to the draft version 2.1 of the StanForD 2010 standard schemas. Finally, the Invoice Specification message was included in the schema version 2.1 in November 2012. Consequently, the timetable of the national implementation of the UDI/UDD protocol in Finland was dependent on the use of the StanForD 2010 standard. The application of the UDI/UDD protocol has been possible since 2017 when WoodForce operation planning and monitoring system (https://www.woodforce.fi/), based on StanForD 2010, was published in production use for forest harvesting entrepreneurs.

2. Introduction to the UDI/UDD messages

The User Defined Data structure enables questions with pre-defined response options to the machine (e.g. for self-monitoring of harvesting quality). It includes several options for data saving:

- There is no separate independent reply message, but the User Defined Data is written together with other message returning from the measuring device of a forest harvesting machine.
- The information about production per operator and object can be saved hpr, thp and fpr files.
- Machine monitoring information is stored for operator and object in mom files.
- MonitoringShift information is stored for shift, operator and object in mom files.

A summary of the phases of use

Office application / entrepreneur, manager		**Measuring device / operator**
Defines the query instruction in the UDI message - One or more tables of the data to be surveyed (DataTableGroup, DataTable) - Columns (DataColumn) and rows (DataRow) - Cell Data Types: (DataCategory) - Optional Selection Lists: (eNumList) - mandatory or optional fields - Specifying a message returning the driver input data (OutputDataLocation) - Question Time on the Machine: (TableTriggerPoint)		
Generates an UDI message that is sent to the measuring device	->	Reads the UDI message: - The machine can have multiple UDI messages received from different sources that are managed on the system
		Entering data - The system opens a table-based query view to the driver in accordance with the trigger points defined in UDI ("StartObject", "EndObject", "EndShift", "UnloadingComplete", "ChangeSubObject", "StartShift", "ChangeObject" - The user fills fixed rows, or may add rows manually (if UDI permits) - The system validates the entered data - Aligns data to an object according to UDI (ProductionObject, MonitoringObject, MonitoringShift, ProductionLocation)
Receives UDD message	<-	Returns the driver's information in a message specified by the UDI
Manages the further processing of the received data		

More detailed introduction of UDI and UDD messages and the recommendations

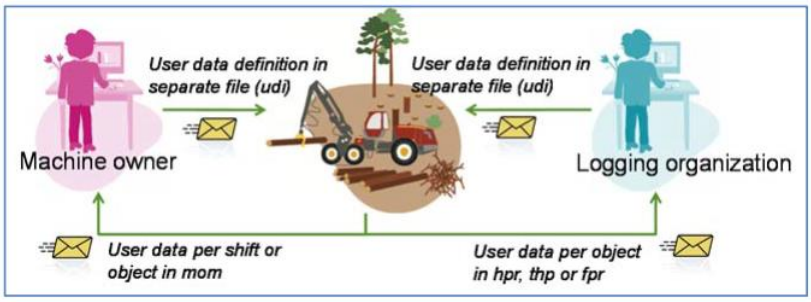
The contents of this paragraph are mostly quoted from the Skogforsk document ”Structural descriptions and implementation recommendations” (https://www.skogforsk.se/contentassets/1a68cdce4af1462ead048b7a5ef1cc06/stanford-2010-introduction-150826.pdf)

User Defined Data

StanForD 2010 includes a flexible solution for sending company-specific forms for e.g. follow-up in digital format (from version 2.1). The instruction defines user-specific tables and questionnaires to be manually filled in by the operator. The manually registered data are returned from the machine as part of messages for either production reporting or operational monitoring. User-specific data for follow-up could be cleaning of understory, oil consumption, number of high stumps for nature conservation, information regarding landings etc. This means that we hopefully can get rid of the operator writing additional company specific information on a piece of paper or in an excel-sheet.

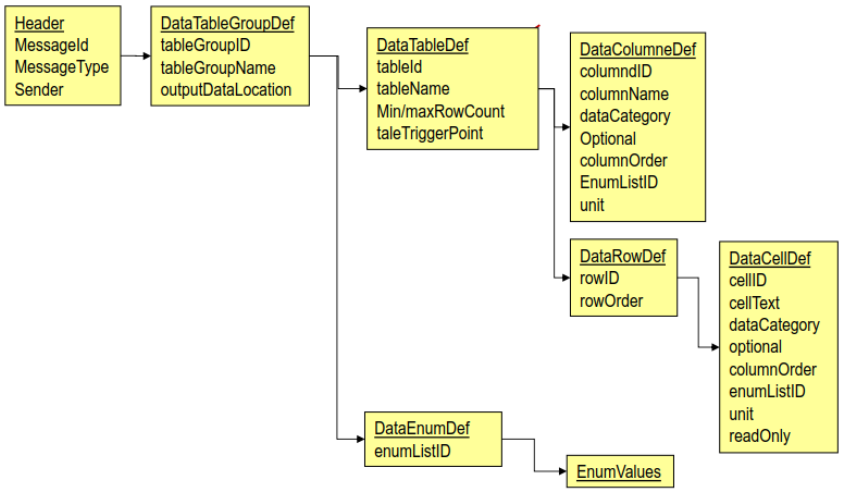
The figure below illustrates the solution for how instructions are sent to a machine from either forest machine owner or logging organization and then returned within a StanForD message.

The basic idea is to make it possible to define tables that the operator can use in the machine to manually register data relevant for logging organization or machine owner.



Input / Instruction

Below is an illustration describing the structure of the User defined data instruction (UDI) looks like:



Observe that the DataRowDefinition was added to make it possible to pre-define a number of questions with different enumeration lists for different rows.

DataTableGroupDefinition

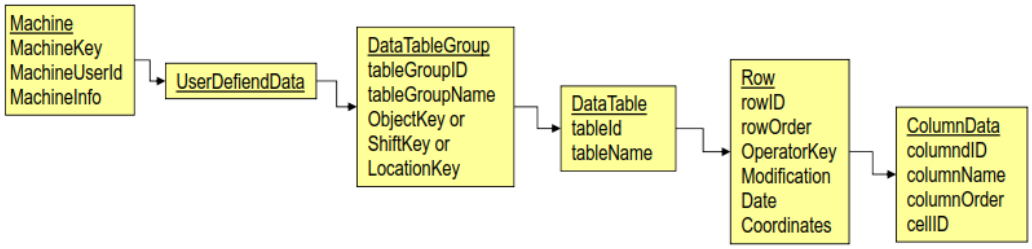
This structure is comparable with a database or an “excel work book” to be reported in a single way.

Attributes for DataTableGroupDefinition:
- tableGroupId (mandatory, string)
- tableGroupName (mandatory, string) = Name of user defined data table, to be used in the UI
- outputDataLocation (mandatory, enumeration list)

DataTableGroupDefinition is repeatable. The machine owner might need information both per shift and per harvesting object. This requires that there are possibilities to add many DataTableGroupDefinitions. The idea is that for every DataTableGroupDefinition there will be a new "tab" in the dialog.

Attribute **outputDataLocation** points to where the user defined data should be registered and then reported. The following enumerations are included:
- ProductionObject, information is stored per object in hpr, thp and fpr files (ObjectKey in illustration below is used).
- MonitoringObject, information is stored per object in mom files (ObjectKey in illustration below is used).
- MonitoringShift, information is stored per shift in mom files (ShiftKey in illustration below is used)). This means that the element ShiftKey must be used.
- ProductionLocation, information is stored per location in fpr files (LocationKey in illustration below is used). A recommendation could be to assume that the LocationKey references the last location where any volumes have been unloaded if several different locations are used at the same object.

The output data is stored using the following structure in hpr, thp, fpr and mom.



Tables with ProductionObject and MonitoringObject are object specific tables that are to be reset with a new object. If outputLocation is equal to MonitoringShift the table is shift specific, meaning it is to be reset when starting on new shift.

DataTableDefinition

This structure is comparable with a database table or an “excel work sheet”.

Attributes for DataTableDefinition:
- tableId (mandatory, string)
- tableName (mandatory, string) = Name of user defined data table, to be used in the UI
- maxRowCount (mandatory, ”1” or “maxint”) = Maximum number of rows in user defined data table.
- minRowCount (mandatory, ”0” or “1”) = Minimum number of rows in user defined data table.
- tableTriggerPoint introduced in version3.0 (mandatory, enumerations are "StartObject", "EndObject", "EndShift", "UnloadingComplete", "ChangeSubObject", "StartShift", "ChangeObject"). Attribute indicating when a user defined table must be presented to the operator (in the GUI of the machine) in order to make it simple to fill in the table at the right time.

DataColumnDefinition

This structure is comparable with fields in a database table or columns in an “excel-sheet”.

Attributes for DataColumnDefinition:
- columnId (mandatory, string) = Identity of column
- columnName (mandatory, string) = Name of column. Presented in column header.

- dataCategory (mandatory, enumerations = ”enum”, “string”, “integer”, “decimal”, “date”, “boolean” or “sequentialInteger”) = Defines data type of column.
- optional (mandatory, boolean) = Indicates if column must be filled in by operator.
- columnOrder (mandatory, integer) = Presentation order of column in GUI.
- enumListId (optional, string) = Identity of enumeration list. E.g. "UnitList”, refers to a DataEnumDefinition (see also next section)
- unit (optional, string) = Defines unit of column, eg mm, kg, m3, cm etc. Can be used in GUI to inform operator about unit.

DataRowDefinition

This new suggested structure is used in order to define individual cells within a row if there e.g. is a need for defining different enumerations on different rows. All included setting must override settings in DataColumnDefinition.

Attributes for DataRowDefinition:

- rowId (mandatory, string) = Identity of row
- rowOrder (mandatory, integer) = Presentation order of row in GUI.

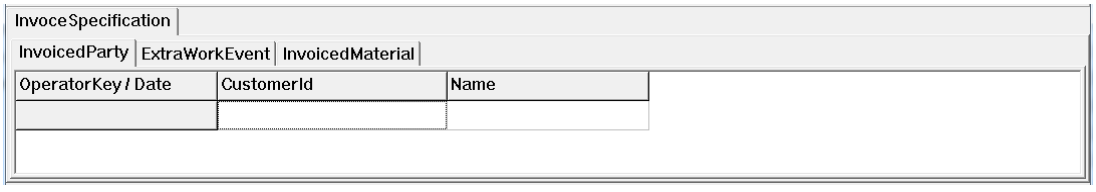
Attributes for DataCellDefinition:

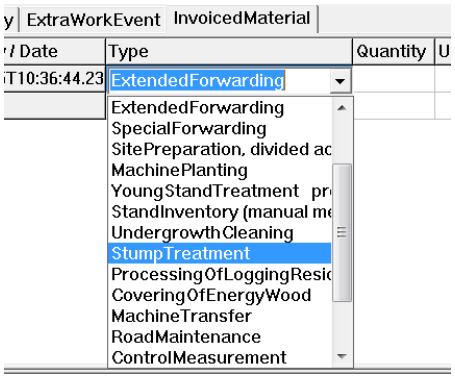
- cellId (mandatory, string) = Identity of cell
- cellText (optional, string) = Text in cell. E.g. a question to be answered by operator.
- dataCategory (optional, enumerations =”enum”, “string”, “integer”, “decimal”, “date”, “boolean” or “sequentialInteger”) = Defines data type of current cell, overrides DataColumnDefinition. Allowed values are: enum, integer, string, decimal, date, boolean, sequential no per row.
- optional (optional, boolean): Indicates if cell must be filled in by operator.
- columnOrder (mandatory, integer) = Column order for current cell in GUI.
- enumListId (optional, string) = Identity of enumeration list. E.g. "UnitList”, refers to a DataEnumDefinition (see also next section)
- unit (optional, string) = Defines unit of column, eg mm, kg, m3, cm etc. Can be used in GUI to inform operator about unit.
- readOnly (optional, Boolean) = Indicates if cell can be edited by operator.

DataEnumDefinition

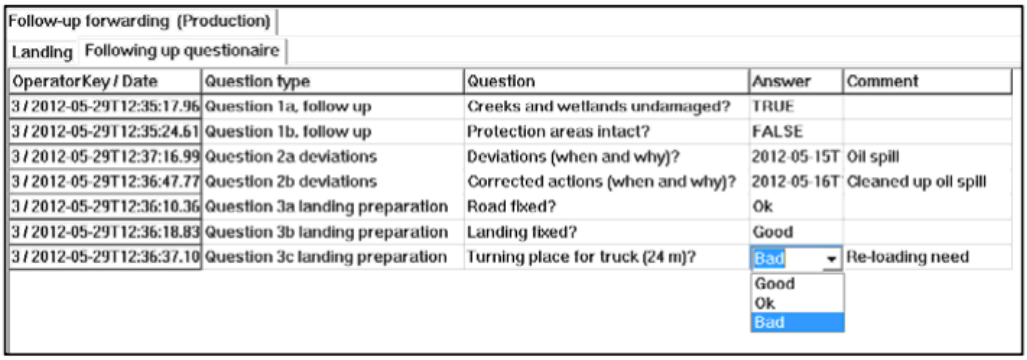
This structure includes enumeration lists to be used in the machine. Attribute enumListId identifies the enumeration list to be used.

User interface example of table
The instruction could render the following GUI.





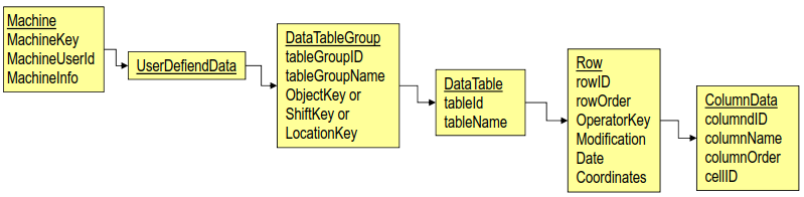
An instruction could also render the following GUI (example is filled in by operator):



Output / Report

The data is validated according to the definition before the file is created. This means that all mandatory fields have to be filled in.

Below is the structure for reporting user defined data from the machine, always included under Machine in mom, fpr, hpr or thp:



User interface example

The output data (XML) described above could be illustrated through the following GUI example.

InvoceSpecification

InvoicedParty | ExtraWorkEvent | InvoicedMaterial

OperatorKey / Date	CustomerId	Name
3 / 2001-12-17T09:30:47Z	35643	BigForest Lmtd

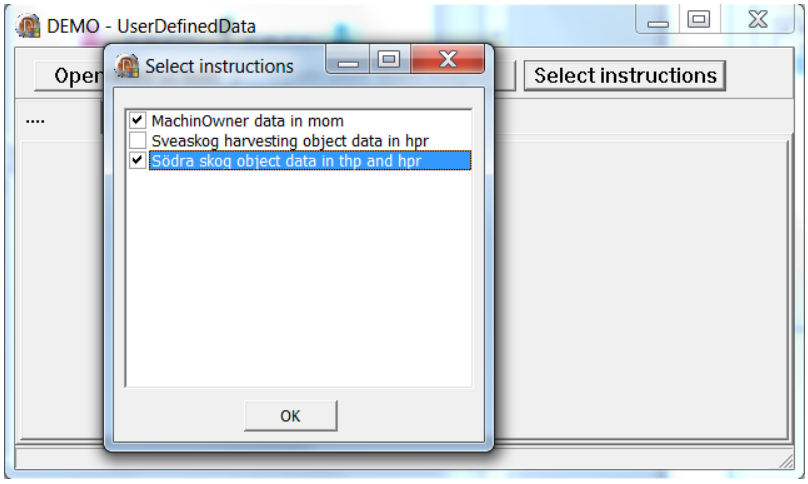
InvoceSpecification

InvoicedParty | ExtraWorkEvent | InvoicedMaterial

OperatorKey / Date	ExtraWorkCategory	BasicsForInvoicingTheWork	Quantity	Unit	Comment
3 / 2001-12-17T09:30:47Z	YoungStandTreatment	Ordered by forest owner	26	hours	This is a test of invoice extrawork
3 / 2001-12-17T09:30:47Z	StandInventory	Ordered by logging company	4	hours	
3 / 2001-12-17T09:30:47Z	StandInventory	Ordered by logging company	4.5	ha	

Managing different instructions in machines

A machine may very well receive instructions from several different organizations. A simple solution for handling different instructions could be that the operator manually selects what instructions to use when starting at a harvesting object:



Operator manually deletes old instructions that are not to be used anymore.

3. Implementation of the UDI/UDD protocol

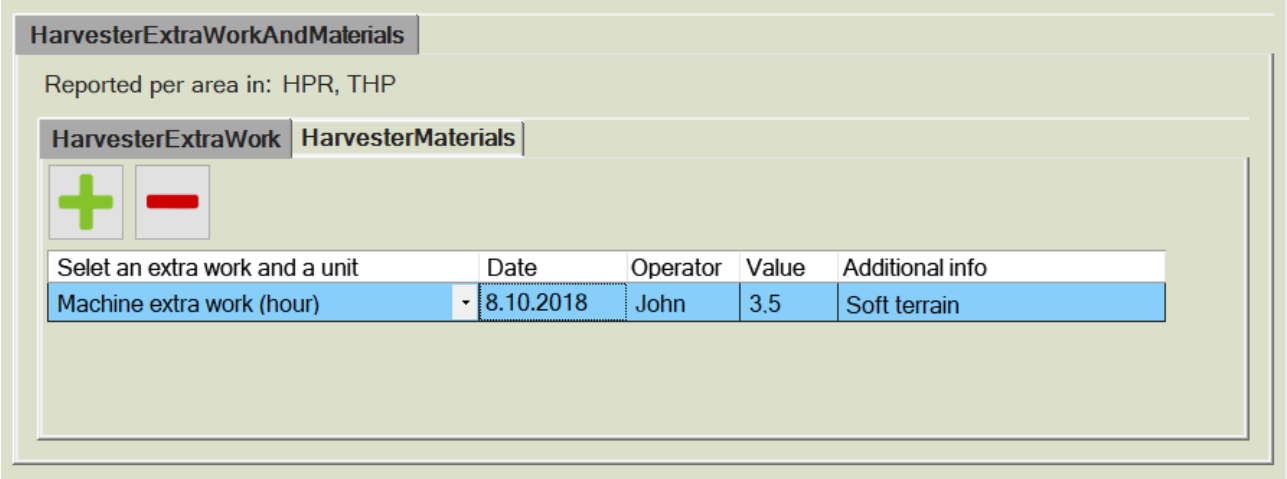
The schema allows quite a wide variety of different use cases. Tietohippu Ltd has implemented two alternative approaches that can be used as guidance when implementing and testing the system. In practice, these alternatives do not exclude each other, but the same UDI message may contain features of both modes simultaneously.

Standardized use based on a homogeneous table structure

The standardized use contains the following features:

- An example UDI message was generated in 2015 by Metsäteho Ltd on the basis of an Excel table compiled by the Trade Association of Finnish Forestry and Earth Moving Contractors
- Consists of a table defined in the UDI, where the structure is determined by column
  - Includes separate questionnaires for additional work and materials
  - The following improvements has been done when utilizing this use case
    - Header texts in the first row of the table shall be moved to header definitions
    - The enumlist attribute is better to be defined per column instead of individual cells
    - Two possible ways of defining data rows:
      - The table can be fixed with a fixed number of lines, as long as the cells are set to optional (the user does not have to fill in the rows)
      - Alternatively, rows can be added by the operator as needed

Here is an example, how the standardized row structure may look like in the measuring device software. The extra work type and unit are selected in the first column, and the rest of the fields are filled up accordingly. The drop list content is the same in each row. The user can add or remove rows in the window, when the table structure is homogeneous.



Alternative method with row-specific table structure

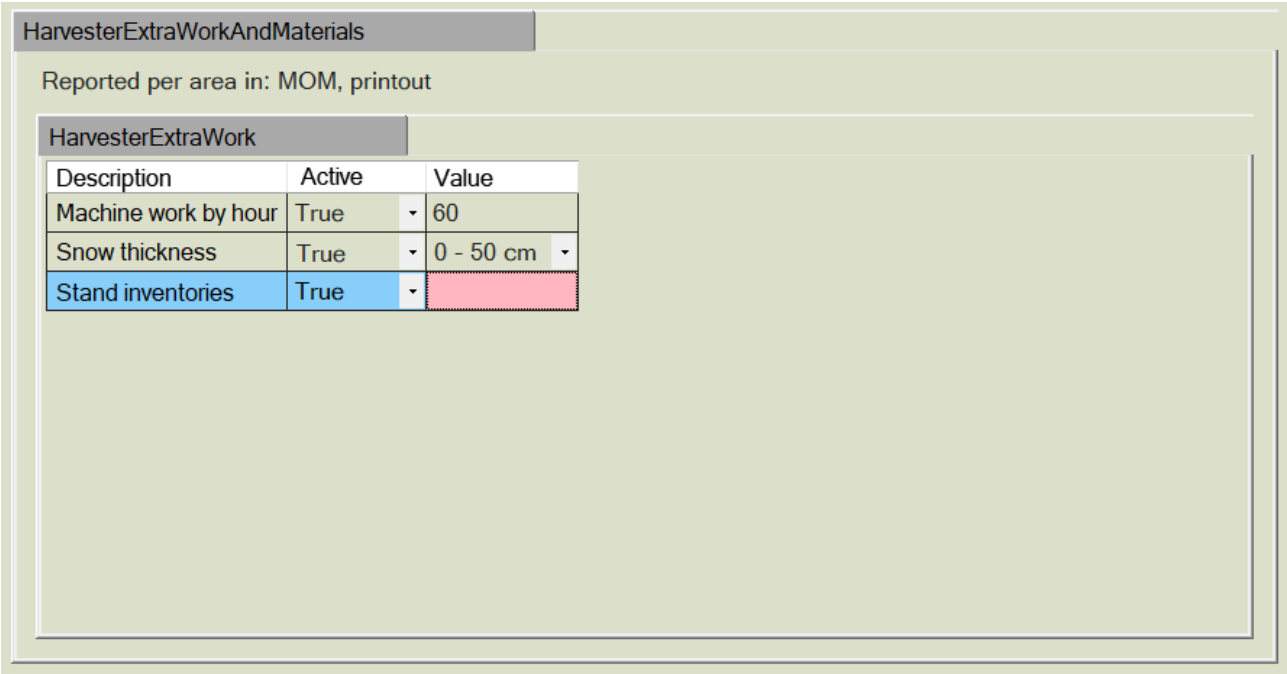
The row-specific use utilizes following features:

- The schema allows for line and cell-specific configuration
- The table is implemented as a fixed predefined row and cell-specific content, and the different rows don't need to be uniform
- Column Definitions apply by default, but they can be overridden by line item definitions
- Rows cannot be added in the machine system, because the structure of the row to be added is not uniquely defined
- Additional guidance is needed for user interface application methods in order to be flexible in utilizing various controls in graphical user interface
- The following features are applied for the measuring device system interface:
  - Use a row / cell ID to identify the context of a data in an office application

- Normal input field with the appropriate data type (integer, string, decimal, date)
- Fixed readOnly string for standard content (displaying standard text in the row)
- The unit can be determined by the unit attribute of the DataCellDefinitionType
- The cellText attribute can be used to determine e.g. a question to the user
- Cell-specific enumListID can be used to present a selection list in a cell (e.g., pre-defined snow depth classes)

- Proposals to clarify the scheme or scope of application:
  - Using CheckBox control in the graphical user interface, when the DataCellDefinition DataCategory attribute is set to Boolean
  - Is it possible to defile a new attribute, so that boolean data type and checkbox control to activate / hide all of the controls in selected row, if needed? (as default, row is hidden with boolean value is false and displayed when value is true)?

Here is an example, how the row-specific structure may look like in the measuring device software. The amount of the rows and the structure of each row are defined in the UDI message, and the machine operator can't add or delete rows. The structure of each row may vary. Separate drop lists may be defined to use in each individual cell.



The example messages described above were tested with a Komatsu Forest measuring device in October 2018. Both practices were proven to be applicable. However, some minor needs for improving the graphical user interface were identified during the tests. The most essential results of Tietohippu Ltd's work in Fobia project concerning UDI/UDD protocol were introduced in October 2018 in StanForD Finland meeting, where most of the measuring device manufacturers were represented. In the near future, Tietohippu Ltd will continue fine tuning in cooperation with Komatsu Forest Ltd. Currently the measuring device systems of the other machine brands do not support the UDI/UDD messages. Tietohippu Ltd will implement the UDI/UDD messages in production use after the other forest machine manufacturers have included the support into their systems.