**Protokoll Workshop HarnessAnalyzer Features 2019 (24.01.2019)**

## Issues Version 2018:

## Datenvergleich (WVPROTICS-270)

Best Practice bei Daimler ist heute die neue HCV als Referenz für den Vergleich zu setzen. Die Anwenderdenke ist hier unterschiedlich zu unserer Implementierung bezüglich gelöschter und hinzugefügter Teile.

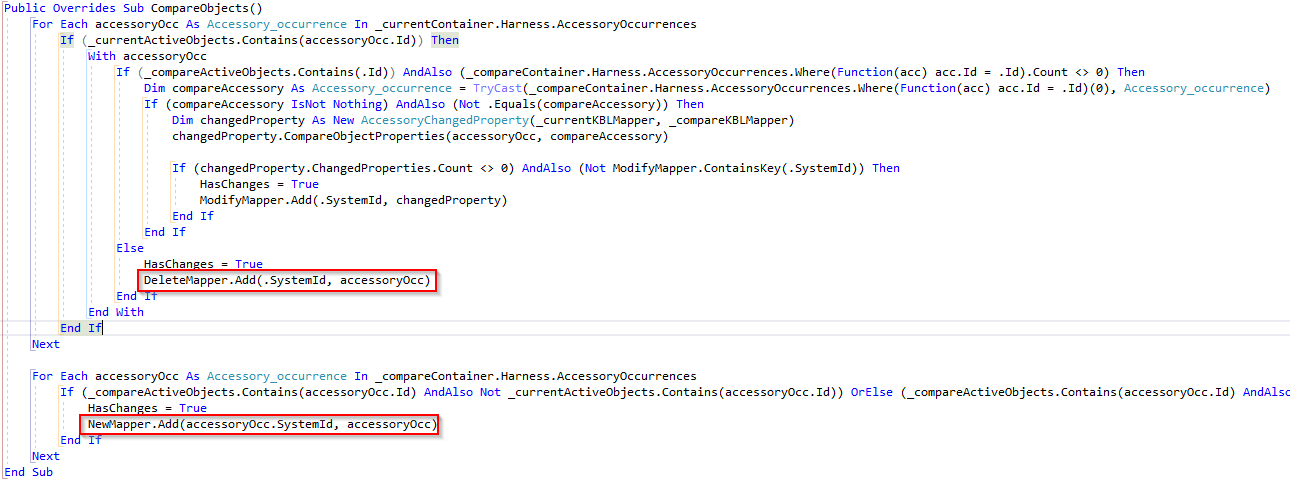
Da der Issue immer wieder hochkommt, daher die Frage, ob die Anzeige hier invertierbar wäre.

Wir werden das analysieren und falls möglich konfigurierbar machen.

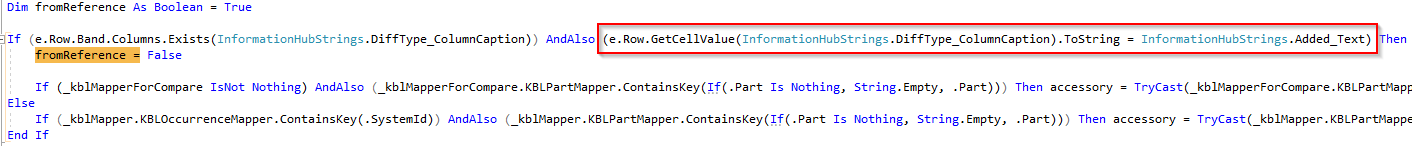
Möglicher Lösungsansatz:

Es muss ein globales Flag in der GeneralSettings-Konfig hinzugefügt werden, in der die Invers-Steuerung der Hinzugefügt/Gelöscht-Anzeige bei den beiden Vergleichsfunktionen gesteuert werden kann (nicht über UI steuerbar).

Hierzu muss im Code für den Datenvergleich an voraussichtlich zwei Stellen eingegriffen werden.



Zum einen müssen alle ComparisonMapper der Objekttypen an markierten Stellen per Flag gesteuert werden, sodass die Mapper ggf. invers befüllt werden.



Zum anderen muss in allen Grids bei entsprechenden Events wie z.B. CellDataRequested die Steuerung des fromReference-Flags richtiggestellt werden und ebenfalls über das Flag der Invertierung der Anzeige des Datenvergleichs korrekt gesetzt werden.

Beim grafischen Vergleich ist der Aufwand überschaubar: Hier muss nur in der Methode „CompareDocuments“ beim Aufruf der Methode „AddResultRow“ die Invertierung der Anzeige für Hinzugefügt/Gelöscht über das Flag gesteuert werden. Mehr sollte nicht notwendig sein.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 1 MW**

**(Daimler specific-enhancement)**

## Gemeinsames Öffnen von xHcv und hcv:

Daimler hat ein Werkzeug zur Erstellung von xHCVs gebaut. Jetzt ist (natürlich) im Rahmen der Masterfreigabe aufgefallen, dass es zusätzliche Anteile (nicht KSK) gibt, die auch dazu müssen. In der langfristigen Planung sollen diese auch in den xHCV. Heute geht das aus technischen Gründen nicht und daher würde man gerne diese erstmal manuell dazu laden.

Wir können analysieren was alles betroffen ist, wenn man das so machen wollte. Heute ist das gezielt verriegelt, da Redlining und Changes usw. in den xHcv / Hcv gespeichert werden und dass mit der jetzigen Struktur nicht auseinanderhaltbar ist. Außerdem werden Modulsettings mittransportiert, und die passen dann möglicherweise auch nicht.

Möglicher Lösungsansatz:

Unter dieser Maßgabe heißt es, dass der einzige Weg darin besteht, die dazu ladbaren KBL/HCV-Dateien nach der Wahl einer bestimmten xHCV, welche geöffnet werden soll, mit anzugeben und zusammen als erweiterte xHCV im HarnessAnalyzer zu laden. Dazu ist bei dem Öffnungs-Prozess der xHCV eine Auswahl von bestimmten HCV/KBL zu treffen (max. 10 Dateien), welche mit der xHCV zusammen geladen werden sollen. Intern im System soll dabei kein Impakt entstehen, welcher zur Folge hätte, an den jetzigen Implementierungen beim Laden der xHCV etwas zu verändern (mögliche Probleme hierbei: Inliner-Mapping, Overall connectivity-Sicht und Speicherung von Redlining und weiteren User-Settings etc.)! Es muss lediglich beim Extrakt der Container in das UserTemp-Verzeichnis eine Überstruktur geschaffen werden, in welcher das xHCV sowie die HCVs entpackt bzw. dann im Falle eines Speicherns wieder gepackt werden.

Die Auswahl, ob Zusatz-Dateien mitgeladen werden sollen, muss wiederum per Flag in den GeneralSettings gesteuert werden (UI-setzbar).

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,5 MW**

**(Daimler specific –enhancement)**

## Neue Features:

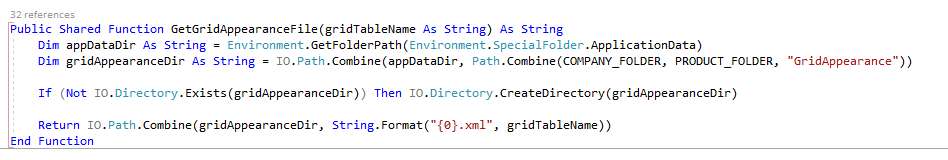
## Datenvergleich auf kbl Basis:1,2/3

Der Datenvergleich heute nimmt alle Attribute an den Objekten mit. Es gibt Anwederrollen, die bestimmte Attributänderungen nicht interessieren, und daher wäre eine Funktion wünschenswert, mit der man den Vergleich feiner steuern kann. Entgegen der Folien ist in der Diskussion klargeworden, dass eine Steuerung wie bei der Suchbarkeit sinnvoll ist. Allerdings muss die Möglichkeit gegeben sein, den Default dieser zu vergleichenden Attribute vorgeben zu können, analog wie es bei den Weight Settings gemacht wird. Die heutige xml Struktur könnte erweitert werden, aber das default Thema ist dafür noch nicht vorhanden.

Eine Anzeige, dass die vom Anwender eingestellten Properties von dem vorgegebenen Default Setting abweichen ist notwendig, damit der Anwender gewarnt ist, dass der Vergleich möglicherweise nicht vollständig ist.

Möglicher Lösungsansatz:

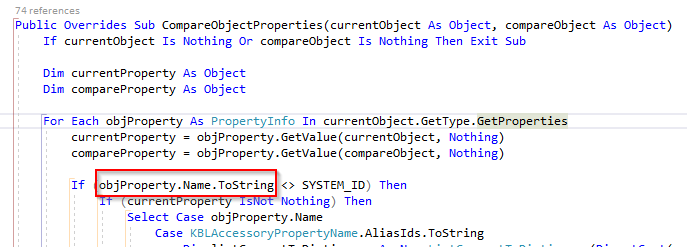
Hier muss, identisch zu den WeightSettings, beim Laden eine Vorranglogik eingebaut werden, welche im Installationsverzeichnis nach einem „GridAppearance“-Ordner samt der entsprechenden XML-Dateien sucht, um daraus die entsprechenden Einstellungen zu laden. Hierzu muss im Code nur an einer Stelle (in der Basisklasse „GridAppearance“) etwas verändert werden.

****

Voraussetzung ist, dass vorab der Ordner „GridAppearance“ im AppData-Verzeichnis des Users gelöscht wird, sonst kann der Ansatz nicht greifen.

Im Tabelleneinstellungs-Dialog muss im Kontext-Menü ein weiterer Button „Vergleichbar“ sowie das Flag innerhalb der Config-Datei erweitert werden (hat zur Folge, dass die Version hochgezählt werden muss).

Bei den ChangedProperty-Klassen muss beim Loop über die Properties des Objects (per Reflection) der Filter ausgewertet werden.

****

Zudem muss eine Warn-Anzeige im Vergleichs-Dialog hinzugefügt werden, wenn eine Abweichung der Comparable-Settings der Grids zu den Default-Settings im Installationsverzeichnis aufgetreten sind.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,5 MW**

**(Update)**

## Der Punkt Datenvergleich 3/3:

Der heutige Vergleich basiert auf der Teilenummer. Es gab eine Diskussion, den Vergleich auf dem Modulnamen aufzubauen, der aber wieder verworfen wurden. Wir müssen nochmals auf beiden Seiten nachsehen, was der Grund war- die Featureeingabe kam vermutlich von verschiedenen Anwendern.

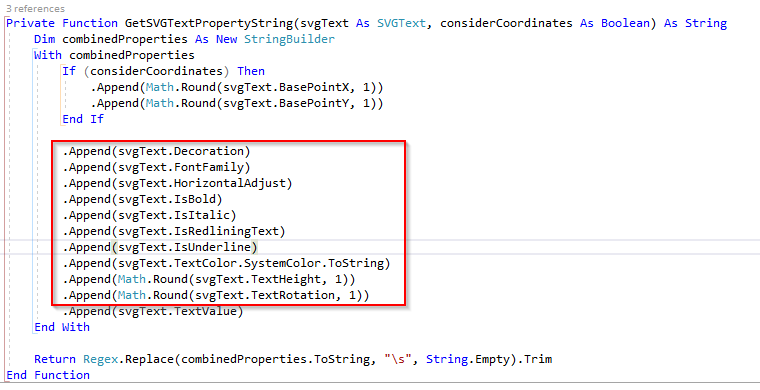
## Grafischer Vergleich 1/5: (WVPROTICS-267)

Es besteht der Wunsch, Änderungen von Schriften und Schrittgrößen zu erkennen (da in der Norm festgelegt), aber diese auch gezielt aus dem Vergleich nehmen zu können. Heute sind diese in den Vergleich einbezogen.

Technisch ist das möglich, aber es müsste eine Parametrierung erfolgen können. Diese Prüfung muss getrennt von der Basis SVG Prüfung steuerbar sein, da die Performance sonst zu schlecht wird.

Möglicher Lösungsansatz:

Es muss ein Flag setzbar im grafischen Vergleichs-Dialog hinzugefügt werden, welcher die Steuerung übernimmt, ob Textstyle-Eigenschaften mit berücksichtigt werden beim Vergleich oder nicht. Das Ganze wird erst einmal on block umgesetzt, später kann hier evtl. noch feingranularer die Steuerung der Vergleichs-Properties an den Elementen eingestellt werden.



Hier muss im Code nur an einer Stelle das Flag ausgewertet werden.

**Zeitaufwandsschätzung: Implementierung 🡪 0,5 MW, Test 🡪 0,25 MW**

**(Update)**

## Grafischer Vergleich 2/5: (WVPROTICS-269)

Der grafische Vergleich sollte auch auf andere Tabellen mit der Pfeil Markerlogik funktionieren. Heute ist er auf Stecker- und Fixingtabellen beschränkt. Wir prüfen, ob das Verlinken und das Row Tagging heute in der SVG sauber rausgeschrieben wird. Wenn das der Fall ist, wäre eine solche Funktion realisierbar.

Möglicher Lösungsansatz:

Zuerst muss die SVG-Struktur für diese Art Tabellen geändert werden: Diese sind heute nicht getaggt, sondern reine Grafik ohne Struktur, also nicht erkennbar und die heutige Logik kann dabei nicht greifen. Diese Art vom Modul-Tabellen muss sich strikt an die Struktur der Steckertabellen anlehnen, um diese Funktion umzusetzen.

Falls dies im entsprechenden Autoren-System implementiert wird, ist der Aufwand auf unserer Seite recht überschaubar, da nur der Filter der „Connector“-Bezeichnung erweitert werden muss.

**Zeitaufwandsschätzung: Implementierung 🡪 0,5 MW, Test 🡪 0,5 MW**

**(Update)**

## Grafischer Vergleich 3/5: (WVPROTICS-268)

Der Sprung vom Pfeilmarker auf einer Tabelle zum Eintrag in der Liste ist gewünscht.

Diese Funktionalität ist sehr aufwändig und die heutigen Strukturen geben das nicht her.

Ist aus technischen Gründen so nicht machbar.

## Grafischer Vergleich 4,5/5: (WVPROTICS-276)

Referenzmaß und Dimensions sind jetzt in den KBLs enthalten. Diese werden heute nicht in den Vergleichen mit einbezogen. Dies muss nachgezogen werden. Was die grafische Darstellung anbelangt, ist die Spezifikation und die notwendige Verlinkung mit zu erweitern.

Möglicher Lösungsansatz:

Für die beiden Objekttypen „Default\_dimension\_specification“ und „Dimension\_specification“ sind sämtliche Implementierungen notwendig, die bei Neuanlage eines Objekttyps notwendig sind:

* KBL-Mapper erweitern
* Grids samt Konfig/Appearance anlegen + Properties inkl. Exports
* Vergleichsfunktionen erweitern (auch grafisch falls Link zur SVG vorhanden)
* Redlining/Memolist etc. erweitern

**Zeitaufwandsschätzung: Implementierung 🡪 2 MW, Test 🡪 1 MW**

**(Update)**

## Datenvergleich allgemein:

Das Thema Datenvergleich auf KBL Basis zerfällt eigentlich in zwei Teile. Da wir die Vergleichspunkte im grafischen Vergleich abhaken und auch speichern können, wird es als inkonsistent angesehen, dass diese Funktion im Datenvergleich nicht vorhanden ist. Diese wäre für die Benutzer alleine schon sehr hilfreich.

Von der technischen Seite sollte dies kein größeres Problem sein, einzig die Frage, ob diese Informationen in die gleiche Struktur kommen, die im grafischen Vergleich verwendet wird, oder ob hier eine getrennte Lösung sinnvoller ist.

Final wünscht man sich eine Kopplung dieser beiden Vergleiche, damit man Punkte, die schon im technischen Vergleich abgehakt worden sind nicht nochmal im grafischen Vergleich bearbeiten muss.

Es wurde von uns darauf hingewiesen, dass diese Abbildung des technischen Objektes gegen die grafische Repräsentanz nicht 1:1 ist und daher eine hohe Komplexität in das Thema kommt.

(abgehakter Stecker versus Steckertabelle, die auch Leitungen usw. anzeigt.)

Der Punkt ist verstanden worden und man ist auch im ersten Schritt mit einer Lösung des reinen Abhakens zufrieden.

Möglicher Lösungsansatz:

Es wäre aus Sicht der Umsetzung ein kompletter Blue-Print der Funktionalitäten des grafischen Vergleichs samt Config-Struktur zur Speicherung der User-Inputs. Diese soll aber in einer separaten Datei gespeichert werden.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,5 MW**

**(enhancement)**

## Grafische Darstellung 1/5:

Es soll eine Möglichkeit geschaffen werden, bei der Objekte, die keine Modulzuweisung haben, ausgeblendet werden können. Entgegen der Folien ist hier ein Schalter in der GUI vorgeschlagen, mit dem die Funktion gesteuert werden kann. Der Modulbaum bleibt unverändert.

Möglicher Lösungsansatz:

Hier ist ein zusätzlicher State-Button notwendig, der die Visible-Steuerung aller grafischen Elemente, welche keine Modul- oder Occurrence-Zuordnung haben (außer Blattrahmen), übernimmt.

**Zeitaufwandsschätzung: Implementierung 🡪 0,5 MW, Test 🡪 0,25 MW**

**(update)**

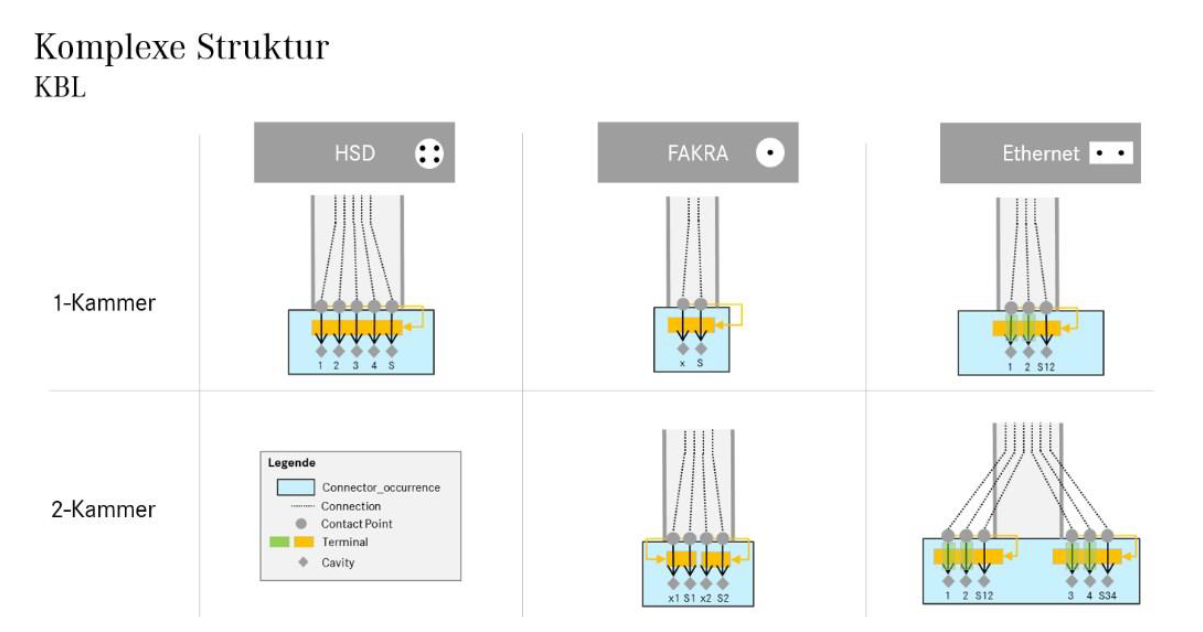
## Grafische Darstellung 3/5:

Es werden immer mehr komplexe Stecker mit besonderer Kontaktierung im HF Bereich verbaut. Eine Ansicht ist gewünscht, die aus der KBL Kontaktpunkt Modellierung bestimmte Muster erkennt und eine generische Ansicht des Steckers generiert. Die Vorgaben dazu sind bei uns vorhanden. Modulabhängigkeiten müssen nicht berücksichtig werden. Wir können diese Funktion abschätzen.

Möglicher Lösungsansatz:

Ein neuer Eintrag im Stecker-Kontext-Menü ist notwendig „Zeige Prinzip-Schaubild“, welcher einen neuen View (Dialog) öffnet, mit einer VectorDraw-Komponente, welche das Schaubild visualisieren kann. Dabei muss die Innenkontaktierung des Steckers samt Kontaktpunkten, Steckerkammern sowie Terminals berücksichtigt (analysiert) werden und im generischen Ansatz visualisiert werden.

Wichtig ist der Test der Funktion, sodass die Modellierung in der KBL hier keinen Impakt auf die Visualisierung des Schaubilds hat (ggf. bei Daimler nach sämtlichen möglichen Szenarien der Kontaktpunkt-Modellierung nachfragen).



**Zeitaufwandsschätzung: Implementierung 🡪 4 MW, Test 🡪 1,5 MW**

**(enhancement)**

## Grafische Darstellung 4/5:

Dieser Punkt zerfällt in zwei Teile. Der erste Wunsch kommt wieder aus der nicht richtig funktionierenden Masterfreigabe. Das Übereinanderlegen von Zeichnungen wurde von uns als nicht möglich gesetzt. (Weder technisch mit vertretbarem Aufwand, noch von der Datenseite aus).

Das Thema JT wird jetzt für kleine Umfänge gelebt. Das gesamte ID-Handling ist von DC aus geklärt und es wird in einer Aliasid mit entsprechendem scope in der KBL zur Verfügung stehen.

Das JT würde mit im hcv enthalten sein. Einen 1:1 Abbildung (ein JT Stecker ist mit einer ID Verlinkt) ist laut Daimler gegeben.

Möglicher Lösungsansatz:

Voraussetzung ist ein Satz von Demo-Ständen, in welchen die 3D-Verknüpfung im JT-File angegeben ist und auch eine dazu passende KBL. Es wird kein neuer MIME-Type definiert, sondern es gilt die Voraussetzung, dass innerhalb der HCV ein JT-File gegen ein KBL-Dokument gelinkt wird.

Hier ist auch bzgl. Deployment noch ein geringer Aufwand notwendig, da wir hier die kernel\_io.dll der CoreTech-Jungs mit ausliefern müssen, und diese muss für das Deploy gegen unsere interne DLL ausgetauscht werden.

Die Implementierung ist eine Erweiterung des PoC-Ansatzes, welcher schon zu großen Teilen die Funktionalität trägt. Hier muss nur auf die neuen Model3D-Library sowie die neue Kernel3D-Library referenziert werden, sowie die internen Strukturen bzgl. Mesh-Generierung überarbeitet werden.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 1 MW**

**(update)**

## Grafische Darstellung 5/5:

Diese Funktion der Erweiterung der Verbindungsansicht mit Sprung zur Gegentrennstelle im xHCV ist technisch möglich, wird von uns im Zusatzschaltplanmodul realisiert und wird daher hier nicht vorgesehen.

## Visualisierung Datagrids:

Die Filterfunktion wird ausgiebig genutzt und der Custom Filter ist für Standardaufgaben zu umständlich. Eine Funktion wie in Excel ist gewünscht, bei der man direkt den Wert eingeben kann.

Wir werden die Möglichkeiten analysieren.

Möglicher Lösungsansatz:

Ist zu großen Teilen schon gelöst! Muss noch in HarnessAnalyzer eingebaut werden.

**Zeitaufwandsschätzung: Implementierung 🡪 0,5 MW, Test 🡪 0,25 MW**

**(update-fertig)**

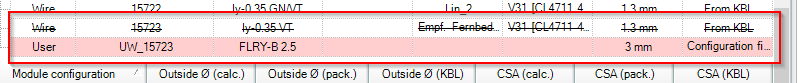
## Bündelview und Änderung des Leitungstyps: (WVPROTICS-181)

Die Änderung des Leitungstyps für Leitungen aus der KBL ist hier verriegelt. Um einen anderen Typ zu nehmen, muss man ein Userwire anlegen, den entsprechenden Typ wählen, und das Alte dann deaktivieren.

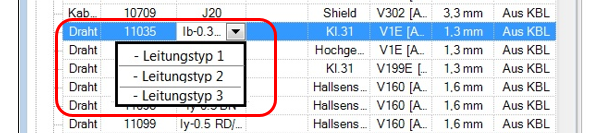
Der Weg ist etwas umständlich und daher würde man gerne den Typ direkt auf der Leitung ändern. Wir müssen analysieren, welche Details an der Verriegelung hängen oder evtl. eine automatische Funktion bauen, die bei Änderung des Leitungstyps diese obigen Schritte automatisch durchführt.

Möglicher Lösungsansatz:

Die Combo-Box im Cables/Wires-Grid wird für alle realen Rows aufgemacht, sodass der Benutzer immer in der Lage ist, eine Drahttyp-Auswahl zu treffen. Wird dies gemacht, wird automatisch der selektierte Draht auf gelöscht gesetzt und ein neuer User-Draht mit entsprechender Voreinstellung des Drahttyps angelegt.



Ein weiterer Ansatz wäre, gar kein UserWire zu generieren, und den Typ direkt für entsprechende Wire/Core-Row zu ersetzen. Hier muss nur darauf geachtet werden, dass für die Kalkulation sowie Anzeige im Grid der Link zum GeneralWire separat mitgehalten werden muss (kann hidden column sein etc.).



**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,5 MW**

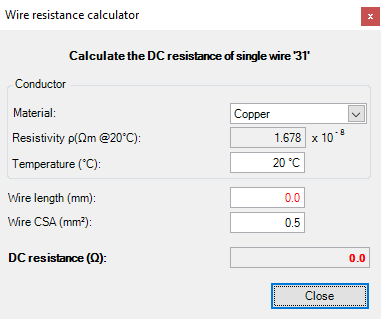
**(enhancement)**

## Widerstandberechnung(WVPROTICS-243)

Eine Widerstandsberechnung des gesamten Pfades über die Trennstellen ist gewünscht. Kontakte werden erstmal nicht mit einbezogen. Diese Funktion kann von uns abgeschätzt werden.

Möglicher Lösungsansatz:

Wir verwenden hierfür die schon existierende Funktionalität sowie UI der Widerstandberechnung, erweitern diese um eine CheckBox, mit welcher Trennstellenübergänge des selektierten Drahtes berücksichtigt werden sollen.



Dazu muss eine xHCV geladen sein und die Übergänge sauber erkannt worden sein. Darüber hinaus muss in diesem kleinen Dialog noch eine Liste der gefundenen Drähte in anderen Leitungssätzen angezeigt werden.

Möglicherweise ist ein weiterer Einstiegspunkt für dieses Feature der OverallConnectivity-View, wo schon heute die Gesamtstruktur ausgehend von einem selektierten Draht angezeigt wird.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,5 MW**

**(update-fertig)**

## Leitungsgewichtsberechnung(WVPROTICS-271)

Es taucht häufig die Fragestellung auf, Alu gegen Kupfer zu bewerten. Nach der Diskussion wie das funktionieren kann (z.B. fester Querschnittsfaktor usw.) ist vereinbart worden, dass Daimler hier erst nochmal in die Diskussion mit den Anwendern geht.

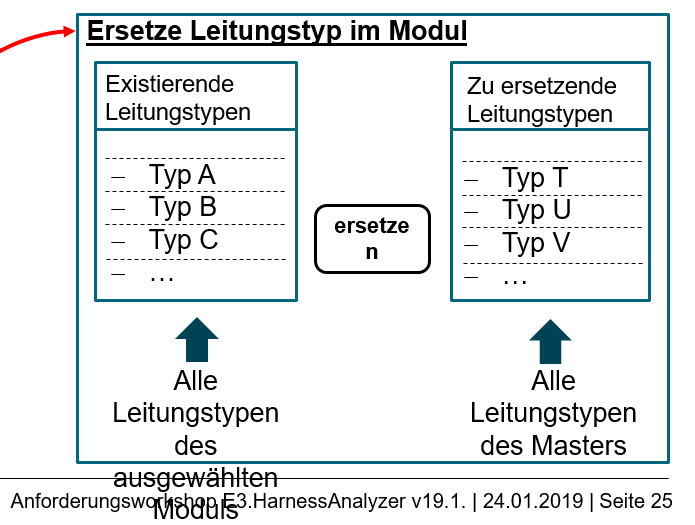
## Leitungstyp für Module oder global änderbar machen (WVPROTICS-256):

Für viele Analysen gibt es den Wunsch, den Leitungstyp umsetzen zu können. Das sollte sowohl global als auch auf Modulbasis möglich sein. Von der technischen Seite ist das nicht ganz einfache, da wir dann neue Teileinstanzen in die Datenstruktur bauen müssten und die betroffenen Leitungen dann entsprechend umhängen müssen. Es muss intern gewährleistet sein, dass alle weitern Algorithmen nicht betroffen sind und wie vorher funktionieren. Diese Funktion bringt höhere Komplexität und ist vermutlich so nicht sinnvoll darstellbar.

Möglicher Lösungsansatz:

Hier haben wir mehrere Aktivitäten, welche bei der Umsetzung berücksichtigt werden müssen:

* Ein neuer Dialog ist anzulegen, wo das Mapping der Typen konfiguriert werden kann
* Es kann nur auf vorhandene Drahttypen innerhalb der KBL verwiesen werden
* Globales Setting pro KBL/HCV überschreibt Setting der Zuordnung pro Module ohne Rückfrage!
* Pro Modul kann dies per neuen Kontext-Menu-Button konfiguriert werden
* Global kann dies per Menü-Button im Hauptmenü konfiguriert werden
* Es wird nichts persistent gehalten, ein Mapping ist nicht speicherbar und auch nicht revertbar innerhalb einer Session
* Wir muss hierzu das KBL-Property „Part“ am General\_wire\_occurrece-Objekt manipulieren; gilt nicht für Special\_wire\_occurrence-Objekte
* Es darf nicht das KBL gespeichert werden! (Check im Code, ob dies tatsächlich so ist!)
* Der Einstellungs-Dialog darf nur exklusiv geöffnet werden, alle anderen Views müssen geschlossen sein
* Reload benötigt das Cable/Wire/Core-Grid, der 3D-View sowie die Schematics, etc.



**Zeitaufwandsschätzung: Implementierung 🡪 4 MW, Test 🡪 1 MW**

**(enhancement)**

## Redlining:

Bei Klick auf das Redlining Symbol soll ein Sprung auf die Anmerkung möglich sein. Das geht heute nicht, da das Symbol nur ein Overlay ist und der link zur Anmerkung fehlt. Wir müssen die Struktur analysieren, um den Aufwand bewerten zu können.

Möglicher Lösungsansatz:

Hier muss im Code beim Click-Event auf dem Canvas geprüft werden, wie man den Stift vom eigentlichen Occurrence-Objekt separieren können. Der Cross-Highlight ist hier möglicherweise um ein Flag zu erweitern, wo geprüft wird, ob das Redlining in den Tabellen gewählt werden soll oder nicht.

**Zeitaufwandsschätzung: Implementierung 🡪 1 MW, Test 🡪 0,25 MW**

**(update)**

## Changes:

Die KBL Changes werden jetzt in der KBL mittransportiert. Der Crosslink in die Zeichnung geht noch nicht. Das soll möglich sein und die Svg Spezifikation muss erweitert werden.

Möglicher Lösungsansatz:

Ist nur eine Stelle im Code (Select case…), welche erweitert werden muss.

**Zeitaufwandsschätzung: Implementierung 🡪 0,25 MW, Test 🡪 0,25 MW**

**(update)**

## SVG Validierung:

Der Punkt ist ähnlich zu dem schon weiter oben genannten Punkt. Zusätzlich müssen wir noch die momentan abgeprüften SVG Punkte dokumentieren. Die Überprüfung der normgerechten Schriften usw. ist zwar möglich, aber ein absolut Daimler spezifischen Thema. Zusätzlich müsste man über eine Plugin / Konfigurations Möglichkeit nachdenken. Die Funktion ist außerdem performance-kritisch.

Möglicher Lösungsansatz:

Eigenschaften der einzelnen Grafik-Elemente in der SVG können mit ihren Attributen nur gegen die dazugehörigen Gruppen bzw. ihrer Occurrence-Objekte gemappt werden. Die Spezifikation hierfür muss von Daimler vorgegeben werden (auf Basis ihres Dokuments). Um diese Sonderfunktionalität isoliert anbinden zu können, muss eine Plugin-Struktur geschaffen werden (eigene DLL neben EXE), in welcher die SVG-Validierungslogik ausgeführt wird. Falls das Plugin gefunden wird, muss ein Menü-Button angezeigt werden als Einstiegspunkt der Funktionalität.

Die gefundenen Validierungs-Warnings/Errors müssen als ConsoleItems wieder im Log-Window angezeigt werden.

**Zeitaufwandsschätzung: Implementierung 🡪 2 MW, Test 🡪 0,5 MW (Daimler specific)**

**(enhancement)**

## Virtuelle Trennstellen:

Mit den virtuellen Trennstellen sind hier Trennstellen gemeint, bei den mehrere einzelne Stecker auf einen Stecker der Gegenseite gesteckt werden. Diese Mapping lässt sich nicht sauber aus der Namenskonvention auflösen, ist aber in Connect sauber dokumentiert. Um Trennstellenübergänge richtig hinzubekommen, ist diese Mapping eigentlich nötig. Zusätzlich gibt es noch einen weiteren Sonderfall: bestimmte Leitungssätze wie Sitz werden nur einmal dokumentiert, aber für beide Sitze eingesetzt. Auch hier greift das momentane Mapping über Namen nicht mehr, da zwei Stecker im Innenraum vorhanden sind, aber nur einer auf der Sitzseite vorhanden ist.

Daimler hat einen Vorschlag gemacht, dieses Mapping als individuelle KBL mit den xHCV zu stellen. (nur für solche Trennstellen, nicht für triviale Fälle). In diesen KBLs wird das Mapping über eine Componentbox mit Innenverschaltung realisiert. Der alte Mapping Algorithmus über die Namen würde dann wieder funktionieren können, nur die Auswertung und Bildung der Inlinermap muss mit entsprechender Priorisierung zuerst alle virtuellen Trennstallen erkennen.

Die Funktion ist komplexer, aber sie tritt prinzipiell bei allen Anwendern auf. Wir müssen den Aufwand analysieren.

Möglicher Lösungsansatz:

Ansatz ist, die entsprechenden virtuellen Trennstellen, welche eigenständige KBLs sind, mit einer spezifischen File-Extension (z. B. .vt) im xHCV zu hinterlegen, um dies beim Prozessing besser separieren zu können. Diese Strukturen der Component\_box\_occurrence-Objekte müssen beim Laden in eine eigenständige Struktur überführt werden. Die Inliner-Paare als Objektstruktur müssen erweitert werden: Hier muss neben den eigentlichen Steckerpaar auch eine Definition der relevanten Kammer-Paare angelegt werden.

Die virtuellen Trennstellen müssen vorrangig ausgewertet werden, schon gefundene Paare dürfen beim Prozess der Findung über die Identifikationskriterien nicht mehr berücksichtigt werden.

Der Consistency-Checker der Inliner-Paare muss mit entsprechender Anpassung der Funktionalität überarbeitet werden.

**Zeitaufwandsschätzung: Implementierung 🡪 6 MW, Test 🡪 2 MW**

**(enhancement)**

## Spezifikation:

Wir haben zugesagt, die Svg Spezifikation durchzusehen und bisher aufgetauchte Punkte zu verfeinern oder zu erweitern. Der aktuelle Stand von Daimler liegt uns vor.

**Zeitaufwandsschätzung: Implementierung 🡪 2 MW**

**(update)**

## KBL:

KBL2.4 Innenverschaltung kann zurzeit von Daimler nicht genutzt werden, da der generische Export von E3 nicht brauchbar ist.

KBL 2.5 wird in den nächsten Ausschreibungen gefordert und in die AV eingebaut.

## Weiteres Vorgehen:

Daimler stellt uns einen Querschnitt aus aktuellen hcvs verschiedener Anbieter zu Datenanalyse speziell SVG / Modultabellen zur Verfügung.

Wir haben am 18.02.2019 ein Feedback zu den Punkten versprochen, was wie realisiert oder nicht realisiert werden kann.

**GESAMTAUFWAND GESCHÄTZT: ~43 MW + Aufwendungen für Help/Deploy (~2-4 MW)**