# LoCo Release Notes

*Release 1.48*

info@scale.eu www.scale.eu

SCALE

IT-Solutions for CAE

Inhaltsverzeichnis


**1 Neue Features** **1**

1.1 Performance-Verbesserungen . . . . . 1
1.2 Anpassungen beim Taggen von Objekten . . . . . 2
1.3 Mehrstufiger Modellaufbau . . . . . 4
1.4 NASTRAN-Dateien können aus LoCo direkt mit ANSA geöffnet werden . . . . . 9
1.5 Partial Assembly . . . . . 11

**2 Weitere Verbesserungen** **13**

**3 Bugfixes** **16**

KAPITEL 1

# Neue Features

## 1.1 Performance-Verbesserungen

In umfangreichen Optimierungen wurde an der Performance von LoCo gearbeitet. In Folge der Optimierung verschiedener Teilaspekte wird mit dem aktuelen Release ein flüssigerer Workflow erreicht.

- Downloads von Includes oder Ergebnisdaten werden in eigenen Prozessen durchgeführt. Dadurch wird die GUI bei laufenden Downloads weniger träge. [#3180]
- Die Ermittlung der zu einem Run gehörenden Inhalte wurde beschleunigt. Aufgrund der gestiegenen Anzahl von Komponenten und Parametern, die zu einer Pool-Version gehören, kam es zuletzt zu erheblichen Verzögerungen, sobald in der GUI ein Run oder eine RunConfig ausgewählt wurde. Um das Problem zu addressieren, wurden verschiedene Aufgaben im Zusammenhang mit der Ermittlung des Inhalts vom Run individuell optimiert.
- Laden und Auswerten der Attribute einzelner Komponenten [#3242],[#3207]
- Ermittlung von in einer Pool-Version gemounteten Pools (z.B. Material, Dummies, ...) [#2488]
- Optimierung der Ermittlung der Sortierreihenfolge von Komponenten (An der Optimierung für das Laden der Sort-Spalte wird weiterhin gearbeitet. Bei Performance-Problemen sollte die Sort-Spalte ausgeblendet werden, sofern diese nicht benötigt wird.) [#3413]
- Verbesserungen beim Laden der Bilder aus der Image-Spalte (An der Optimierung für das Laden der Image-Spalte wird weiterhin gearbeitet. Bei Performance-Problemen sollte die Image-Spalte ausgeblendet werden, sofern diese nicht benötigt wird.) [#3263],[#3269]
- Verbesserung des Cachings beim Wechsel zwischen Runs oder RunConfigs [#3177]
- "Prefetch Downloads" werden mit niedrigster Priorität ausgeführt. Dadurch lässt sich der Inhalt einer noch nicht heruntergeladenen Komponente schneller öffnen.

  [#3205],[#3230]
- Umstellung des Verhaltens im *Jump to Version*-Dialog. Hier werden nur noch die Versionen der letzten 90 Tage geladen, wodurch der Dialog deutlich schneller verfügbar ist. Sollten Pool-Versionen älteren Datums benötigt werden, können diese separat geladen werden [#3181]
- Beschleunigtes Öffnen des Assembly-Dialogs [#3231]
- Verbesserung der Anzeige von Ladevorgängen im Browser. Diese werden nicht mehr in Textform auf den Unterebenen, sondern als Icon im Tree angezeigt.

  Dadurch ist dieser bei laufenden Aktualisierungen nicht mehr so stark in Bewegung und muss weniger häufig aktualisiert werden [#3226]
- Optimierungen für das Ableiten von Pool-Versionen [#3336]

Die Perfomance-Optimierungen sind mit dem Release 1.48 noch nicht abgeschlossen und werden in den kommenden Releases fortgesetzt. Darüber hinaus wird aktuell an automatisierten Performance-Tests gearbeitet, mit Hilfe derer eine Perfomance-Verschlechterung im Entwicklungsprozess rechtzeitig erkannt werden soll.

## 1.2 Anpassungen beim Taggen von Objekten

Komponenten, Pool-Versionen, Komponentengruppen, Parametergruppen und andere Objekte können in LoCo mit Tags versehen werden. Von besonderer Bedeutung sind dabei die Tags *obsolete*, *blocked*, *reference* und *status*. Diese Tags mussten bisher weitgehend manuell eingegeben werden.

Der Umgang mit Tags wurde so vereinheitlicht, dass ab sofort alle vordefinierten Tags im Kontextmenü der Komponente/Gruppe/Pool erreichbar sind (Abb. 1.1).

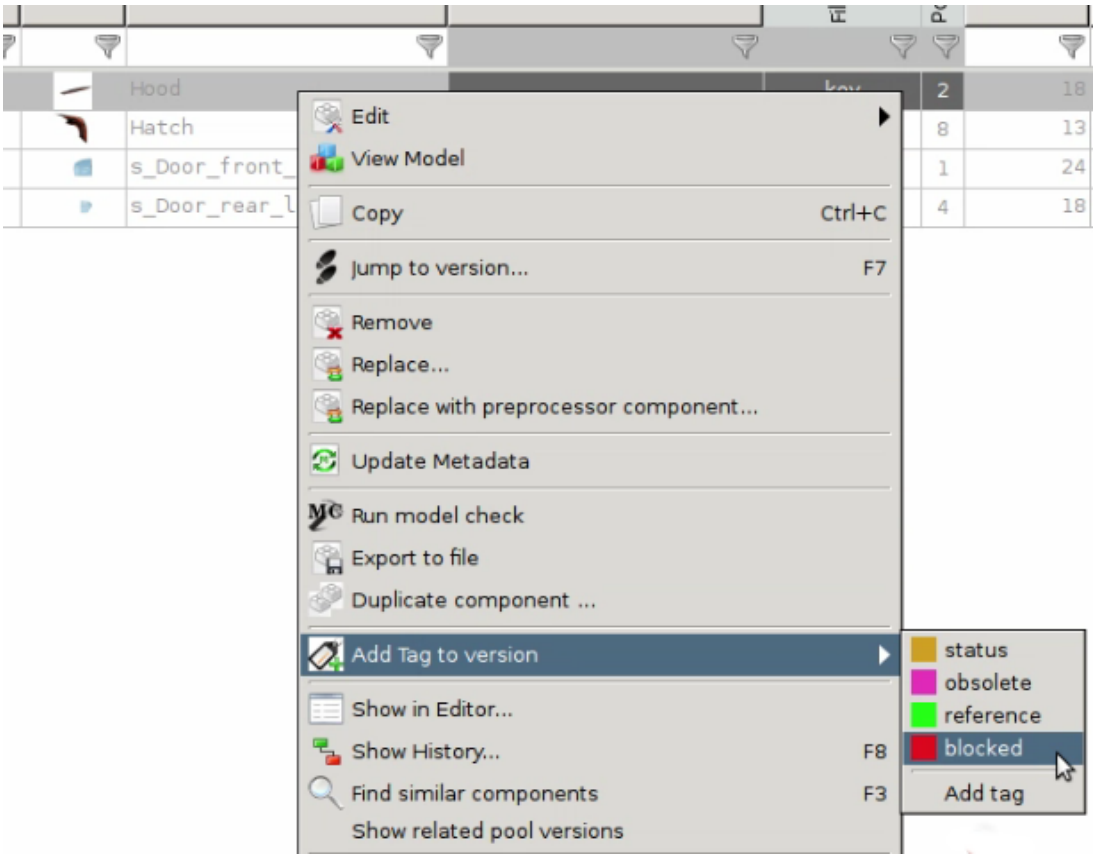


Abb. 1.1: *Add tag* im Kontextmenü

Wenn ein Objekt bereits einen Tag besitzt, erscheint im Kontextmenü zusätzlich die Option *Remove Tag from version*, um diese Tags wieder zu entfernen. (Abb. 1.2).

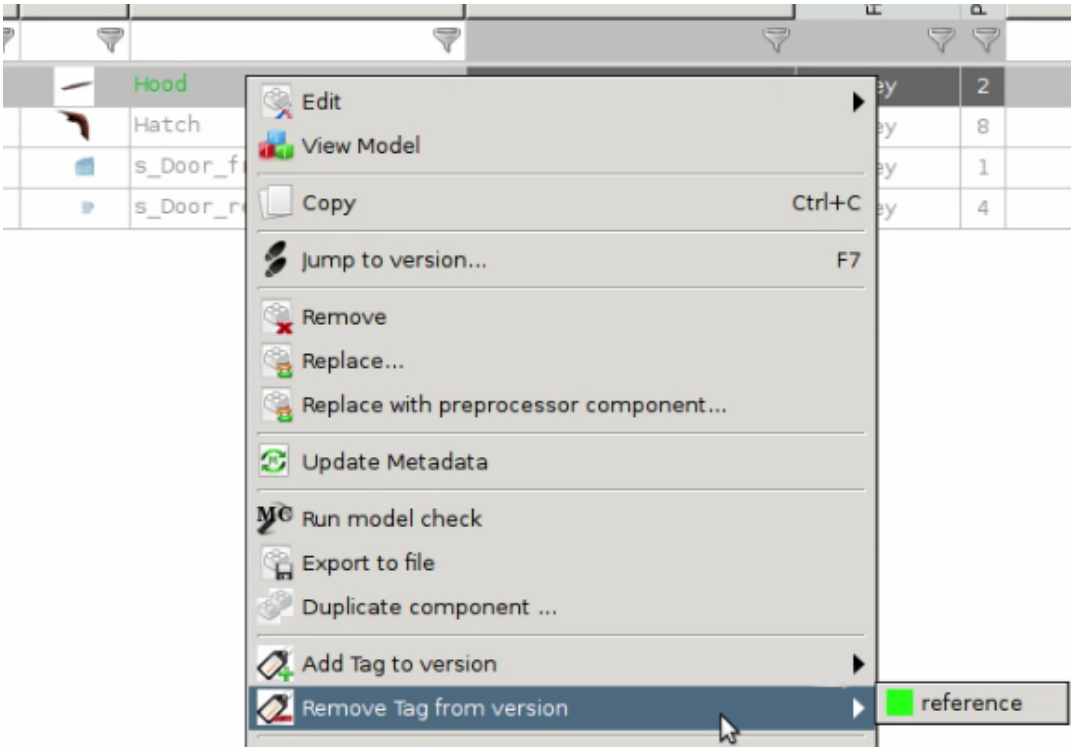


Abb. 1.2: *Remove Tag from version* im Kontextmenü

Um Tags zu definieren, werden KeyUser-Rechte benötigt. Der Tag-Editor ist über *View* > *Window* > *Administration* ereichbar (Abb. 1.3).

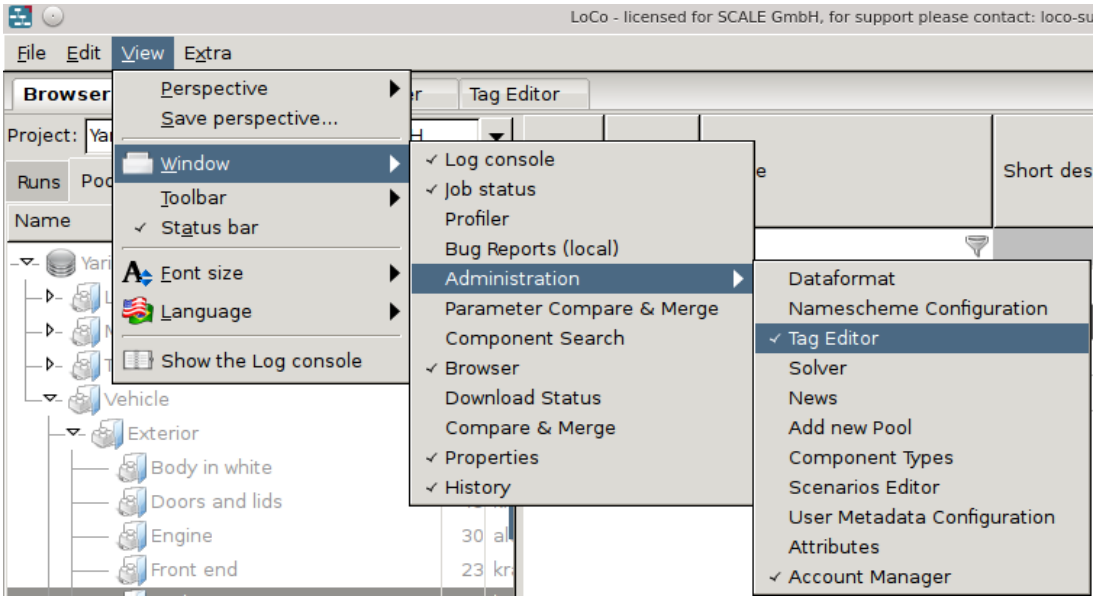


Abb. 1.3: Tag-Editor-Fenster öffnen

Beliebige neue Tags können mit *Add* definiert werden (Abb. 1.4). Die Änderungen im Tag-Editor müssen am Ende mit *Save* gespeichert werden. Für jeden Tag kann ein Name, eine Farbe und eine Beschreibung definiert werden. Die bereits sichtbare Option *Allow RegEx* wird erst in einer zukünftigen LoCo-Version umgesetzt und ist vorerst wirkungslos.

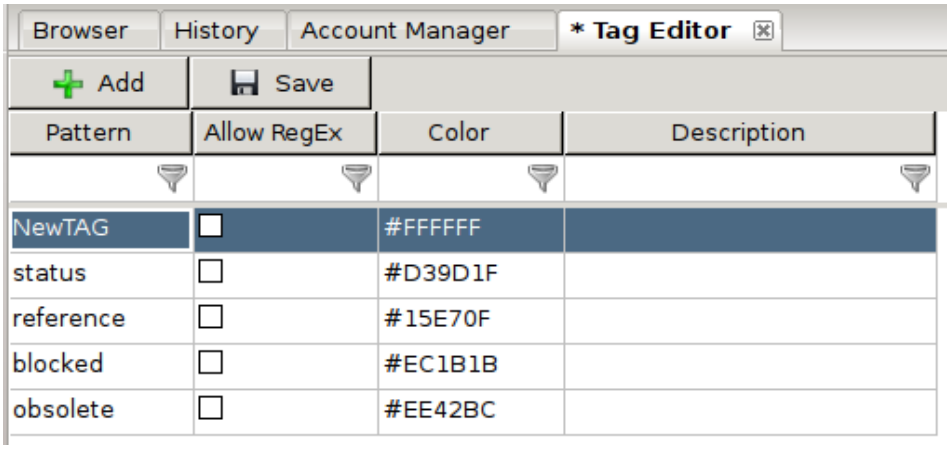


Abb. 1.4: Tag-Editor-Fenster

## 1.3 Mehrstufiger Modellaufbau

**Hinweis:** Die LoCo-Version 1.48.x enthält eine experimentelle Version des mehrstufigen Modellaufbaus!

Große Modelle können Teilmodelle enthalten, die selbst das Ergebnis einer Simulation oder eines anderen Prozesses sind. Beispielgebend ist eine Airbag-folding-Simulation, die ein Airbagmodell erzeugt, das als Teilmodell in einer Fahrzeugsimulation benutzt wird. Ein weiteres Beispiel ist ein Teilmodell *Rad*, das durch eine Aufblas-Simulation erzeugt wird. Um solche mehrstufigen Modelle effizient aufbauen zu können, wurde das neue Feature “Mehrstufiger Modellaufbau” entwickelt. Dabei wird ein SubordinatePool mit den Teilmodellen in den Projekt-Pool eingebunden. Das neue Feature ermöglicht es nun, die Ergebnisdateien der Teilmodelle im Projekt-Pool als Komponenten zu importieren. LoCo speichert und verwaltet die Abhängigkeiten zwischen den verschiedenen Assemblies. Die Abhängigkeiten zwischen den Assemblies können nicht nur eine Kettenstruktur, sondern auch eine Baumstruktur haben. Ein Beispiel für eine solche Baumstruktur ist in Abbildung (Abb. 1.5) dargestellt. Hier werden zunächst vier Räder mit Hilfe von vier Assemblies aufgeblasen. Anschließend werden die vier aufgeblasenen Räder mit Hilfe von vier weiteren Assemblies abgeplattet. Schließlich werden die abgeplatteten Räder in ein Fahrzeug-Crash-Assembly eingebunden. Die Abhängigkeiten zwischen den Assemblies haben eine Baumstruktur mit vier Zweigen und zwei Assemblies auf jedem Zweig.

Wenn eine neue Version des Projekt-Pools abgeleitet und ein Assembly durchgeführt wird, werden die erforderlichen Assembly-Prozesse der Teilmodelle automatisch gestartet. Wenn Subassembly-Parameter geändert werden, ist es vorgesehen, dass die entsprechenden Subassemblies beim Projekt-Pool-Assembly automatisch durchgeführt werden.

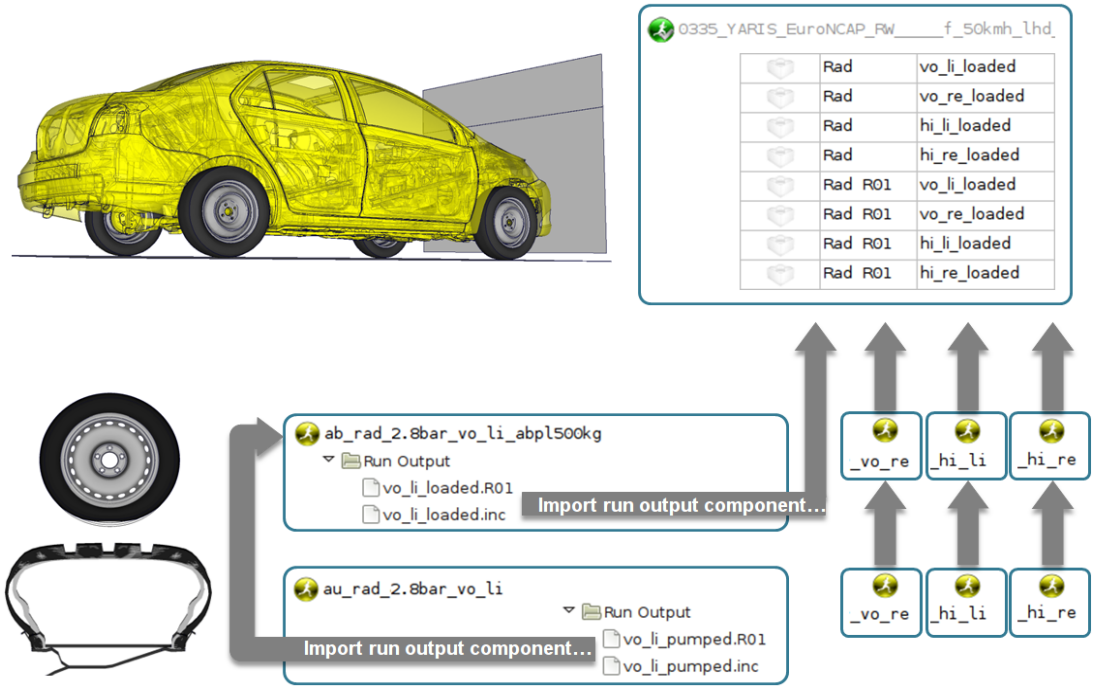


Abb. 1.5: Abhängigkeiten zwischen Assemblies (Beispiel)

Der Rest dieses Abschnitts beschreibt die notwendigen Einstellungen für den mehrstufigen Modellaufbau. Zunächst müssen im SubordinatePool Assemblies für die Teilmodelle definiert werden. Diese Assemblies müssen zusätzlich ein Python-Skript enthalten, welches die gewünschten Ergebnisdateien nach LoCo importiert (Abb. 1.6).

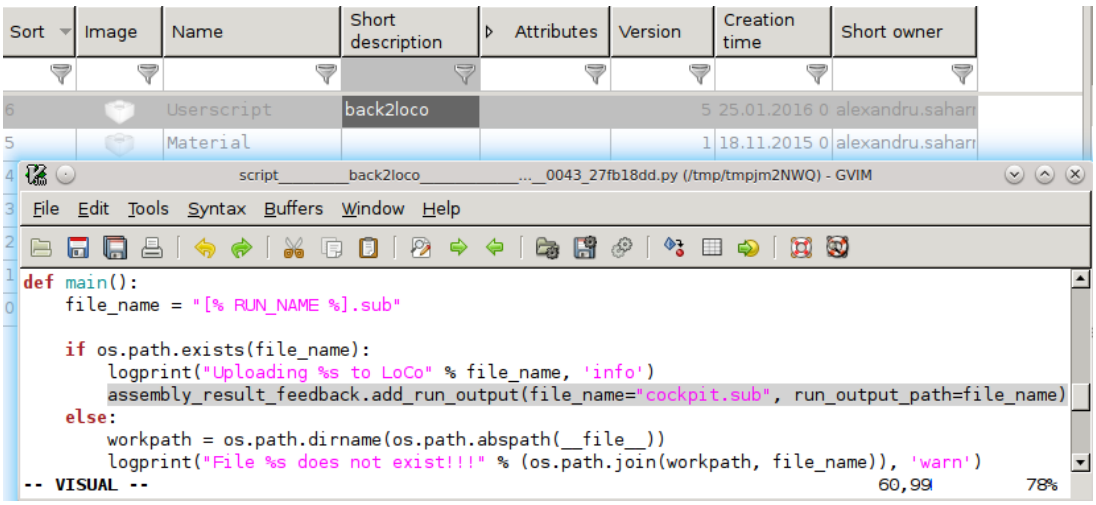


Abb. 1.6: Ergebnisdateien nach LoCo importieren

Die gewünschten Ergebnisdateien werden mit Hilfe der Methode `add_run_output` nach LoCo importiert:

Die Ergebnisdateien, welche nach einem Assembly zurück nach LoCo importiert worden sind, werden im *Properties*-Fenster unter *Run Details* angezeigt (Abb. 1.7).

Listing 1.1: Python Skript `back2loco`

```
# internal python
#
# -*- coding: utf-8 -*-
#! /usr/bin/python

import os
from loco2script import assembly_result_feedback

workpath = os.path.dirname(os.path.abspath(__file__))
fh = "cockpit.sub"
path_fh = os.path.join(workpath, fh)

# Import files back to LoCo
if os.path.exists(path_fh):
    print "Uploading %s to LoCo" % fh
    assembly_result_feedback.add_run_output(file_name=fh, run_output_path=path_fh)
else:
    print "File %s does not exist!!!" % path_fh
```

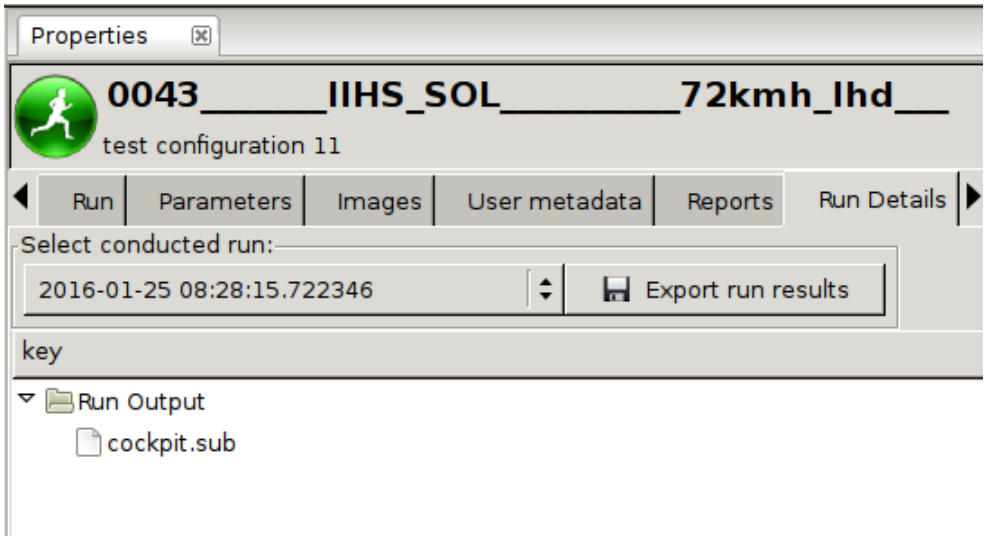


Abb. 1.7: Verfügbare Ergebnisdateien nach einem Assembly

Die Ergebnisdateien sind aber noch nicht im Projekt-Pool als Komponenten importiert. Dafür muss zunächst der SubordinatePool im Projekt-Pool gemountet werden. Anschließend können die Ergebnisdateien der Teilmodelle im Projekt-Pool mit *Import run output component...* als Komponenten importiert werden (Abb. 1.8).

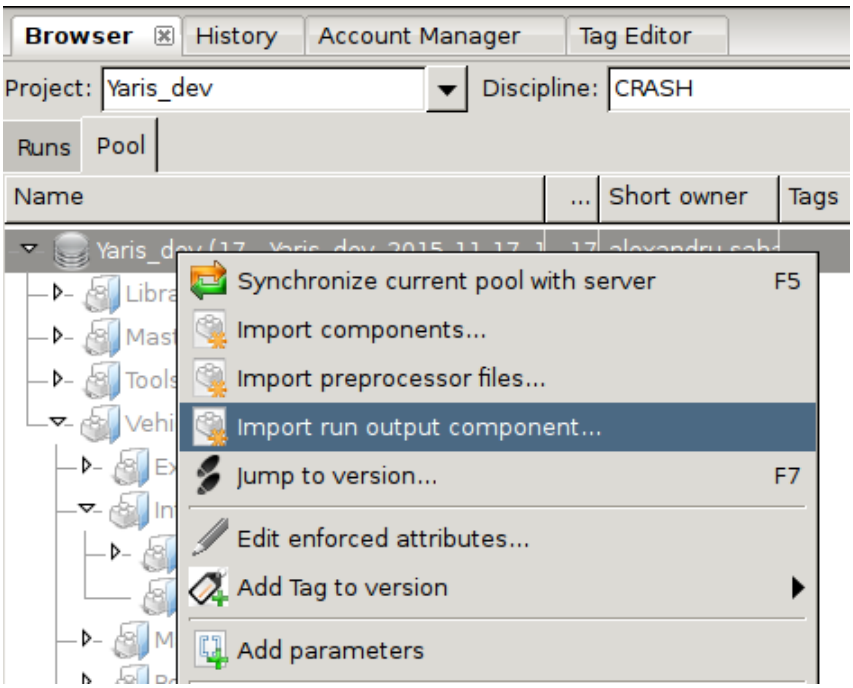


Abb. 1.8: Import run output component

Falls mehrere SubordinatePools im Projekt-Pool gemountet sind, erscheint nach der Auswahl von *Import run output component...* ein Fester, in dem der SubordinatePool ausgewählt wird. (Abb. 1.9).

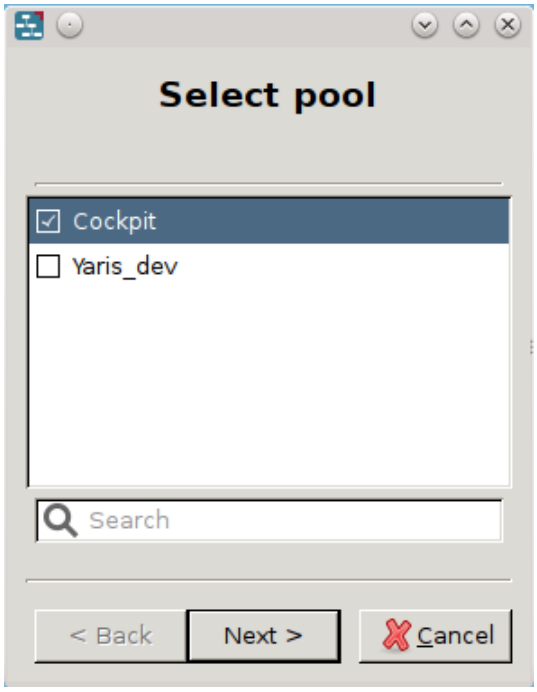


Abb. 1.9: SubordinatePool für Import der Ergebnissdateien selektieren

Falls der gewählte SubordinatePool mehrere Assemblies beinhaltet, erscheint als nächstes ein Fenster, in dem ein Assembly des SubordinatePools ausgewählt wird (Abb. 1.10).

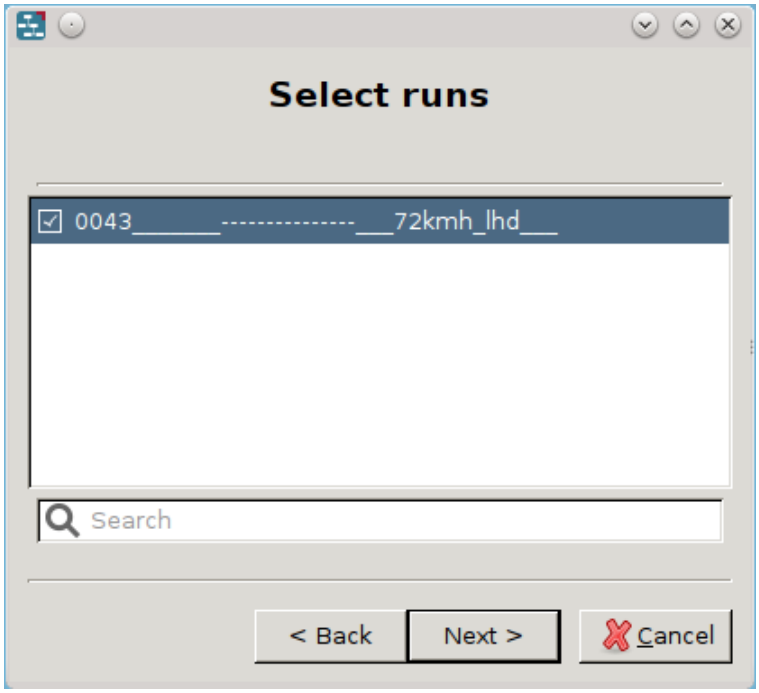


Abb. 1.10: Subassembly des SubordinatePools für Ergebnisdateien-Import selektieren

Falls im ausgewählten Assembly mehrere Ergebnisdateien verfügbar sind, erscheint als nächstes ein Fenster, in dem eine Ergebnisdatei ausgewählt wird (Abb. 1.11).

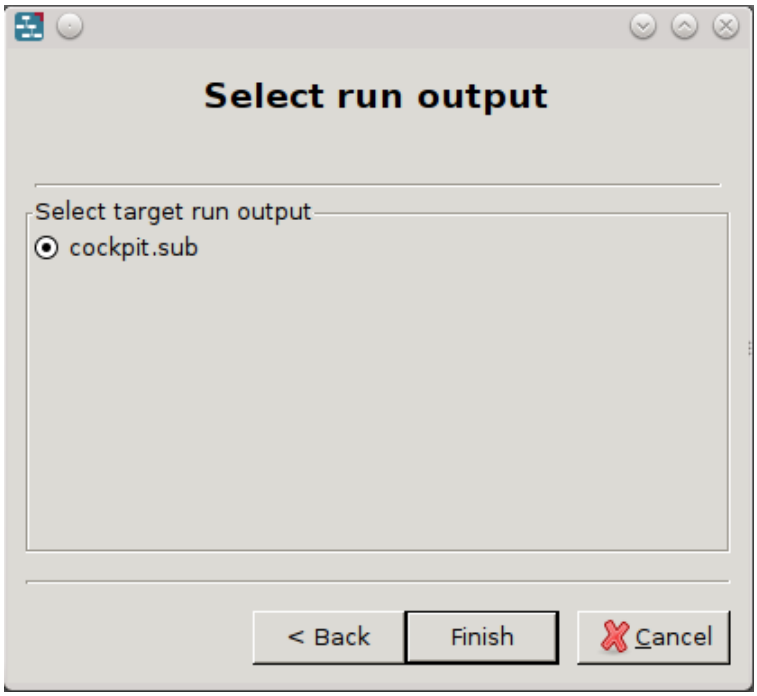


Abb. 1.11: Ergebnisdatei des Subassembly für Import der Ergebnisdateien selektieren

Schließlich erscheint ein normales Importfenster, in dem eine Zielgruppe und ein Komponententyp ausgewählt wird (Abb. 1.12).

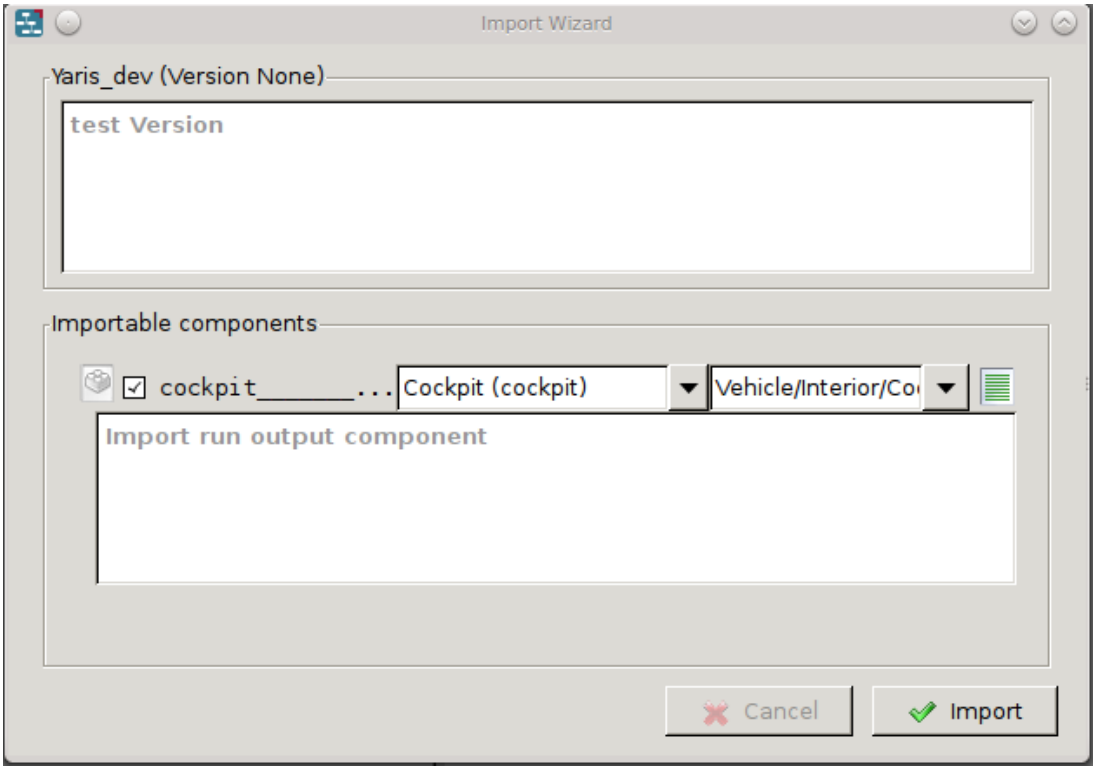


Abb. 1.12: Nomrales Importfenster für Import der Ergebnisdateien

Die neue Komponente liegt jetzt unter der gewählten Gruppe und kann wie eine normale Komponente attributiert und verwendet werden (Abb. 1.13). Im Properties-Fenster der Komponente gibt es einen neuen Reiter *Run Output Component*, in dem unter *Referenced run description* Informationen über das eingebundene Teilmodell angezeigt werden (Abb. 1.13).

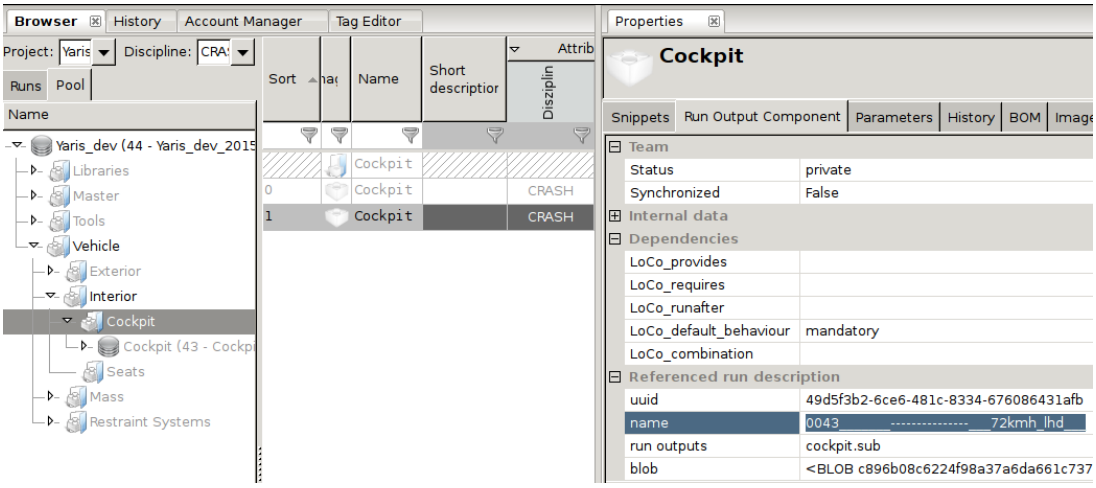


Abb. 1.13: Run-Output-Komponente im Projekt-Pool

## 1.4 NASTRAN-Dateien können aus LoCo direkt mit ANSA geöffnet werden

NASTRAN-Dateien können direkt aus LoCo in ANSA geöffnet werden (Abb. 1.14). Damit ANSA diese Dateien bzw. LoCo-Komponenten als NASTRAN-Input erkennt, müssen diese Komponenten in LoCo das #2778

Attribut FileType auf *nas* oder *bdf* gesetzt haben (Abb. 1.14). Dazu muss ANSA als externer Editor mit der Kategorie *Run-Script* in LoCo eingestellt werden (*Extra* > *Settings* > *External Applications* > *New Application*, Abb. 1.15).

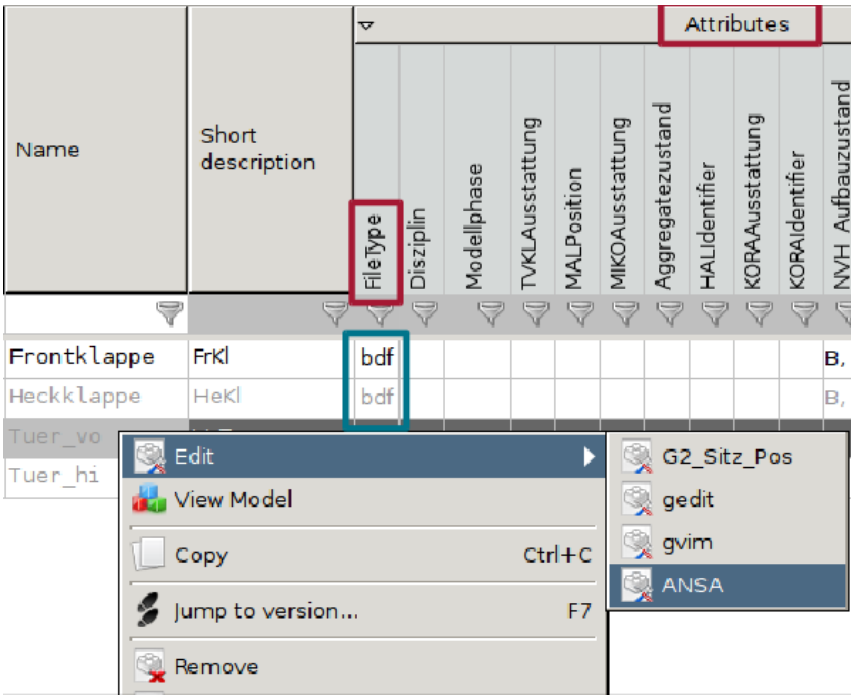


Abb. 1.14: NASTRAN-Datei (.bdf) mit ANSA öffnen

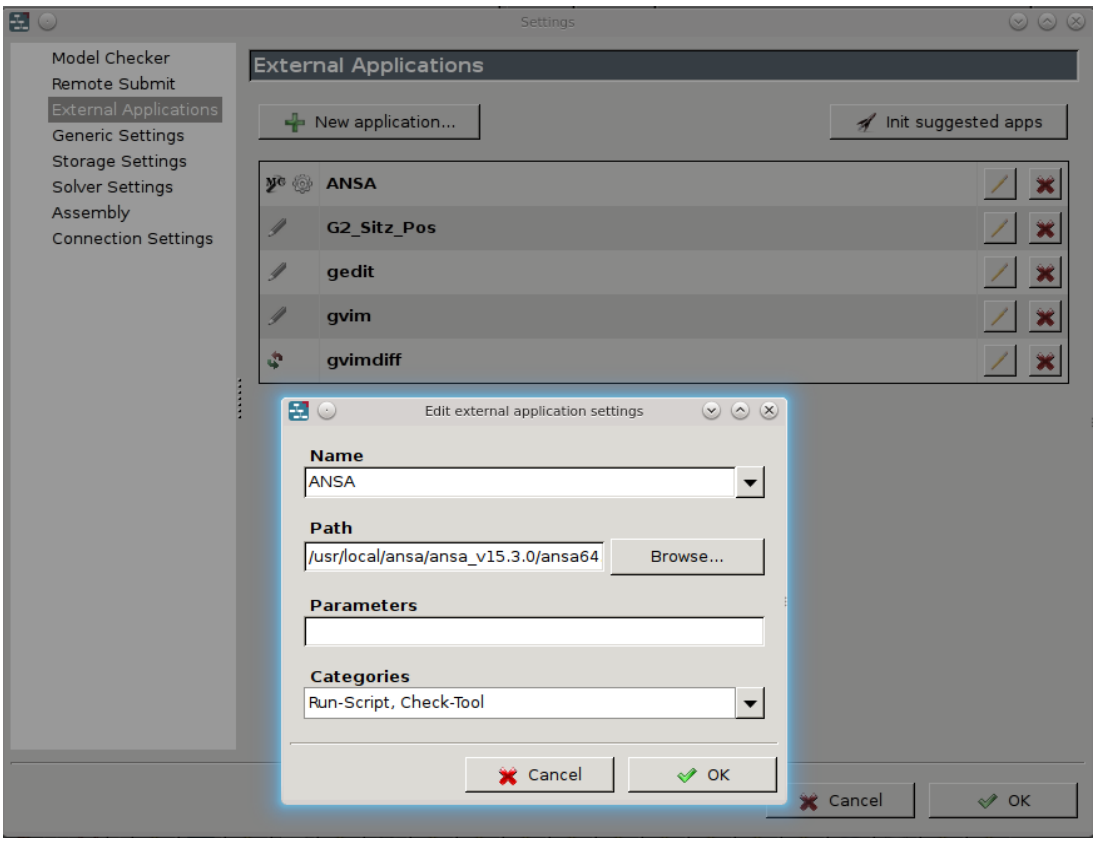


Abb. 1.15: ANSA als *External Application*

## 1.5 Partial Assembly

Es ist ab sofort möglich, einen Teilaufbau eines Modells vorzunehmen. Dafür werden eine oder mehrere Komponenten im Grid eines Runs/RunConfig selektiert. Mit Rechtsklick auf die selektierten Komponenten kann im Kontexmenü die Aktion *Partial assembly* (Abb. 1.16) gewählt werden. Nach Auswahl dieser Option erscheint der reguläre Assembly-Dialog. Beim Aufbau werden aus den normalerweise in diesem Run verfügbaren Komponenten nur die zuvor selektierten berücksichtigt. Zusätzlich werden die Komponenten berücksichtigt, die in der ModelChecker-Konfiguration für ein *mini assembly* vorgesehen sind. Der Ablauf des *Partial Assemblies* unterscheidet sich ansonsten nicht von einem normalen Assembly. #2779

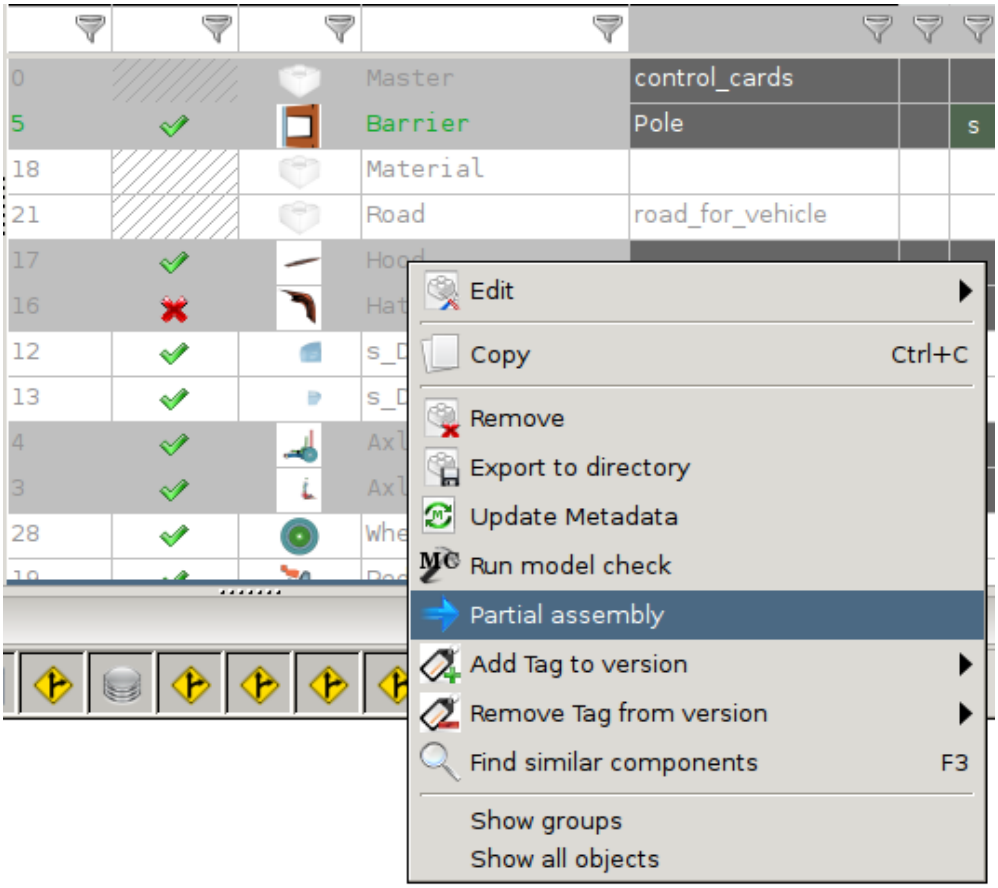


Abb. 1.16: Partial Assembly

KAPITEL 2

# Weitere Verbesserungen

- *Enforced Attributes* sind Attribute, die Pool-Version-übergreifend festgelegt werden. Im Zusammenhang mit enforced attributes wurden folgende Änderungen vorgenommen:
  - Attribute der Pool-Version sind automatisch in den RunConfigs gesetzt und nicht mehr änderbar [#2746];
  - Attributwerte sind im HistoryGraph farblich gekennzeichnet. Nur die Felder, in denen der Attributwert ausgewiesen wird, sind farblich hervorgehoben (Abb. 2.1) [#2410];
  - Um Attributwerte im HistoryGraph farblich kennzeichnen zu können, müssen im Attribut-Editor für diese Attributwerte Farben definiert sein (Abb. 2.2) [#2409].

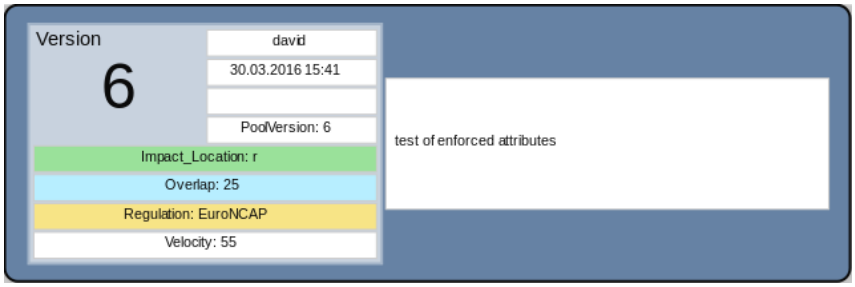


Abb. 2.1: Enforced Attributes HistoryGraph colour

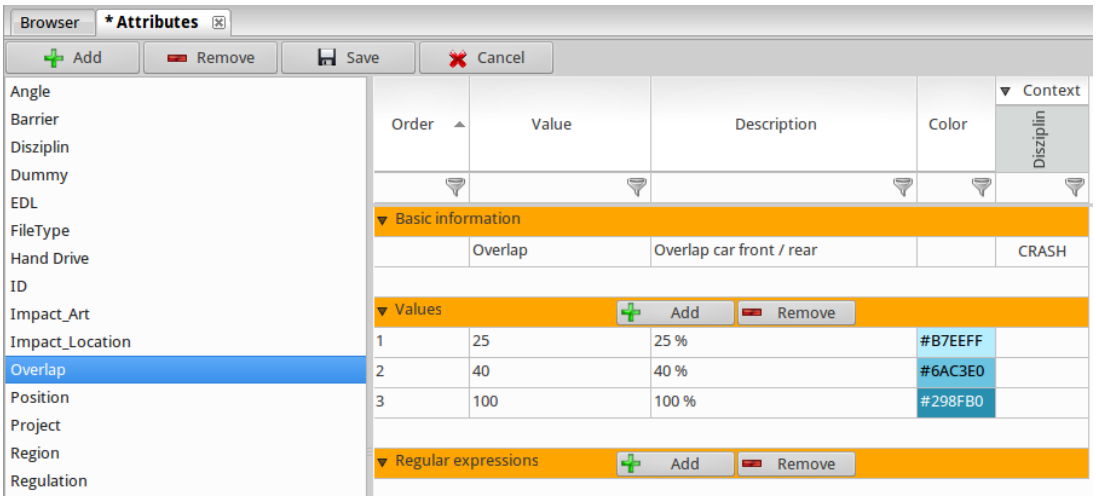


Abb. 2.2: Enforced Attributes set colour

- Assembly in Multistage-Pools triggert SubAssemblies (Mehrstufiger Modellaufbau)[#2390]:
  - Wenn in einem Pool ein SubordinatePool gemountet ist und die angezogene Pool-Version dieses SubordinatePool RunConfigs enthält, werden die Ergebnisse der SubAssemblies beim Assembly anstelle der Komponenten aus dem Pool verwendet;
  - Wurde die RunConfig, deren SubAssembly herangezogen werden soll, noch nicht aufgebaut, ist zuerst das Assembly aus dem Kontext des SubordinatePools ausgeführt (SubAssembly). Wurde die RunConfig bereits aufgebaut, kann auf das erzeugte Aufbau-Ergebnis zurückgegriffen werden und das SubAssembly wird nicht noch einmal ausgeführt;
  - Enthält das SubAssembly wiederum selbst SubAssemblies, von denen es abhängig ist, werden diese zuerst ausgeführt;
  - Die Assemblies sind immer komplett im Kontext des SubordinatePools ausgeführt (die ggf. dort eingehängten Materialien, Tools, etc... werden verwendet);
  - Die SubAssemblies werden jeweils im Assembly-Verzeichnis des jeweils übergeordneten Assemblies ausgeführt und legen ggf. selbst Verzeichnisse an.
- Abgeschickte Jobs werden nicht mehr abgebrochen, wenn nach dem Abschicken in den Offline-Mode gewechselt wird. Bei erneutem Wechsel in den Online-Mode werden die offenen Jobs abgeschickt.[#3316]
- Runs im RunStudyTree werden nur angezeigt, wenn die zugehörigen RunConfig noch unverändert vorhanden ist [#2570]. Bei Verzicht auf die Gruppierungsebene *Assembled status* werden ausschließlich Runs angezeigt, deren entsprechende RunConfig in der Form vorhanden ist, wie sie beim Assembly verwendet worden ist. Alle anderen Runs werden im RunStudyTree nicht mehr angezeigt. Bei Änderung an einer RunConfig verschwindet der Run. Nur die RunConfig wird angezeigt.
- Option *Show related Poolversions* ist auch im HistoryGraph verfügbar [#2317]. Bisher war die Option *Show related Poolversions* nur im Kontextemenü im Grid verfügbar. Die Aktion ist jetzt auch über das Kontextemenü einer Komponente, Komponenten-Gruppe, Parametertabelle und (Subordinate)Pool im HistoryGraph verfügbar.
- Pool, Disziplin und RunGroup-Editor-Auswahlmenüs haben eine Auto-Complete-Option erhalten [#2784 #2785]. Es ist jetzt möglich, in diesen Auswahlmenüs eine Selektion über eine Filtered-Autocomplete-Combobox vorzunehmen. Durch die Eingabe von Buchstaben können die Auswahloptionen gefiltert werden.
- Funktion zum Abbrechen des Assemblies für Skripte ist verfügbar [#2226]. Das Assembly kann jetzt aus einem Skript abbgebrochen werden. Dies ermöglicht den Abbruch bei fatalen Fehlern in Skripten.
- Bei der Bearbeitung von LoCo-Komponenten mit externen Editoren werden die Komponenten in ein temporäres Verzeichnis gespeichert. Der Pfad zu dieser Datei wird als Infomeldung in LoCo angezeigt [#2659].
- Mit Auxiliary Attributes können zwei oder mehrere Komponenten, die verschiedene Werte für das selbe Attribut haben, in einem Run verwendet werden. Die Parameterwerte zur Ersetzung in den durch die Auxiliary Attributes ausgewählten Komponenten werden durch die Auxiliary Attributes ausgewählt [#1839].
- Snippets vom Typ SubRun werden direkt verarbeitet und im PC-File direkt nach den Headern geschrieben [#2829].
- Es ist jetzt möglich, gleichzeitig eine ANSA-Datei und eine .key-Datei zu öffnen [#2688].
- Preprozessorkomponenten können in ANSA aus dem HistoryGraph geöffnet werden [#2431].

- Die LoCo-Performance beim Zugriff über Netzwerklaufwerk wurde verbessert. Dafür muss LoCo mit der Option "–database-guard=agent" gestartet werden [#2687].
- Durch einen Editor zwischengespeicherte Ergebnisse werden direkt in LoCo importiert, ohne den Editor zu schließen. Diese Option muss explizit beim Start von LoCo mit "–reimport-components-on-modification" aktiviert werden [#2585].
- Neues NASTRAN_EXT-Assembly-Template, welches das Assembly-Ergebnis als *tar* packt [#2055].
- Für die Anzeige im HistoryGraph wird die Masse aus den UserMetadaten der Komponente und nicht mehr aus der BOM verwendet [#2558].
- Der *Submit*-Button wird bei einem *Perform datacheck only*-Assembly auf *Datecheck* umbenannt [#1891].
- JobStatus-Liste ist wieder scrollbar [#3006].
- Die Größe des HistoryComments-Fenster wird automatisch angepasst [#2990].
- Szenarien können Kontextabhängig definiert werden. Dadurch wird es beispielsweise möglich für unterschiedliche Disziplinen und Projekte jeweils andere Szenarien anzubieten [#2597].
- Skripte werden jetzt ausgeführt, nachdem das Master-File vollständig erstellt wurde [#3062].
- *Duplicate Component* wird auch für PreProzessor-Komponenten in Kontextmenü angeboten [#3189].
- Komponenten hatten bislang "LoCo_default_behaviour: mandatory" als default gehabt. Jetzt ist es möglich, den gewünschten Wert für "LoCo_default_behaviour" zu wählen [#3056].
- Prefetch-Downloads werden bei niedrigster Priorität abgearbeitet [#3205].
- Auch bei Import von Preprozessorkomponenten wird *replace* angeboten [#3251]

KAPITEL 3

# Bugfixes

- Ausgeblendeter Komponententyp wird nicht mehr unter *Properties* zur Auswahl angeboten [#2329]
- Kein Fehler beim Einfügen von zusätzlichen Attributwerten [#2348]
- Beim Einfügen eines Snippets per Copy'n'Paste auf mehreren Komponenten wird der History-Kommentar nicht mehr für jede Komponenten einzeln abgefragt [#2798]
- *Blauer Pfeil*-Option für "–enable-deprecated-features" wurde entfernt [#2721]
- Falls Dateien, auf die in den .lol-Files verwiesen wird, lokal vorliegen, wird nur ein Link angelegt statt diese nochmals zu kopieren [#2737]
- Anwender, deren Zugriff auf Komponenten mit bestimmten Attributwerten eingeschränkt ist, können Komponenten mit einer Attributierung importieren, die für den Anwender freigegeben ist [#2861]
- Werte für neu erstellte Attribute sind wieder im Browser sichtbar, falls LoCo mit der "–enable-context-filtering"-Option gestartet wurde [#2877]
- ModelCheck bricht nicht mehr ab [#2884]
- Assembly bug 'NoneType' object has no attribute 'copy' gefixt [#2893]
- Multi-Edit von Komponenten funktioniert [#2727]
- Parameterkonflikt wird nicht mehr als solcher angezeigt, nachdem er aufgelöst wurde [#2573]
- Mit *Replace with preprocessor component* wird in ANSA die mit dem FilePicker ausgewählte ANSA-Datei geöffnet [#2985]
- ANSA-Dateien können jetzt auch unter Windows bearbeitet werden [#2953]
- IntegrityError beim initialen Checkout gefixt [#2992]
- Direkter Remote Submit von RunConfigs, die Run-Output-Komponente enthalten, funktioniert jetzt [#2995]
- *Edit with* Kontextmenü-Bug gefixt [#3012]
- Wenn keine Szenarien definiert sind, öffnet LoCo kein Error-Popup mehr [#3020]
- Einträge im JobStatus-Fenster werden auf die verfügbare Breite gekürzt [#1991]
- SSH Remote Submit Bug gefixt [#3016]
- History-Kommentar-Abfrage erfolgt auch nach BOM-Änderung [#371]
- Herunterladen von Ergebnisdaten nach SOAP Remote Submit unter Windows gefixt [#2782]
- Bei Derive New auf einer RunConfig werden im HistoryKommentar keine public-Komponenten mehr angezeigt [#3004]

- Gruppen-Editoren können wieder Gruppenmitgliedschaften bearbeiten [#3027]
- Autocomplete-Box + *STRG* + *A* in der Project-Selection funktioniert [#2842]
- Multi-Radio-Button in der Runconfig-Maske wird jetzt richtig dargestellt [#3032]
- Beim Kopieren von RunConfigs mittels *Duplicate Configuration* in einen neuen Pool bleibt das initiale Namenschema im RunConfigEdit-Fenster erhalten. Nur wenn das initiale Namenschema im neuen Pool nicht mehr verfügbar ist, wird das erste Namenschema selektiert. [#3031]
- Änderung an Komponente fragt HistoryComment vom richtigen Pool ab [#2416]
- Sanitizer-Bug beim Ableiten einer neuen Version von einer public-Komponente, deren Vorgängerversion einen History-Kommentar enthält, welcher durch den Sanitizer bearbeitet werden musste, gefixt [#3055]
- Sichtbare Attribute mit Kontext-Filtering-Bug gefixt [#3033]
- Im Schnellfilter vom *Jump to version*-Dialog ist das Datum explizit ausgeschlossen [#2896]
- Grantor bei Gruppenrecht wird wieder angezeigt [#3079]
- Creator der Gruppe wird im AccountManager wieder angezeigt [#3050]
- FatxmlDateien werden beim Editieren von Komponenten wieder angelegt [#3104]
- Bug beim Run Config > *Properties* > *show in parameter table* gefixt [#3053]
- Sync-Fehler nach Umbenennen eines Nutzers gefixt [#3147]
- Parameter [% RUN_CERTIFICATE %] wird wieder richtig ersetzt [#2580]
- Einfärbung von Tabellen bei AssociationRows korrigiert [#1730]
- Lokale Checks können auch auf Komponenten ausgeführt werden, die sich in eingehängten Pools befinden [#1824]
- Offline-Modus wird aktiviert, wenn Login abgebrochen wird [#3058]
- Importieren von PreProzessorkomponenten funktioniert wieder [#3199]
- Bug beim Bearbeiten (short description geändert) einer public-Komponente gefixt [#3195]
- Mehrfaches Erscheinen des *Edit History Comments*-Dialog beim Hinzufügen eines Snippets korrigiert [#3176]
- Fehler wegen *insufficient permissions* beim Submit einer kopierten RunConfig gefixt [#3204]
- HistoryGraph konnte nicht auf Parametergruppen geöffnet werden [#3155]
- Fehler bei Suche im HistoryGraph, wenn Sonderzeichen in den HistoryComments verwendet worden sind, gefixt [#3198]
- Pool-Versionen mit Subversionsnummern in *Compare&Merge*-Dialog werden wieder richtig angezeigt [#3259 #3267]
- Assembly-Bug beim Ausführen eines Skripts gefixt [#3273]
- Falsche/mehrfache Darstellung in Checks View nach mehrfacher Ausführung von Model Check korrigiert [#3237]
- Bug in *Edit History Comments*-Fenster gefixt [#3149]
- Fortschrittsbalken-Bug gefixt [#2968]
- Komponenten aus nicht-current Pool-Versionen können aus dem HistoryGraph geöffnet werden [#3246]
- Während des Downloads von Ergebnisdaten kamen bei Verwendung vom SOAP-Remote Submit keine anderen Anfragen mehr zum Server durch [#3095]

- Bug wegen Leseberechtigungen in HistoryGraph gefixt [#3340]