requisis

RECONF 2018

DOORS Next Generation – Requirements-Engineering reinvented

> Nikolai Stein, REQUISIS GmbH

requisis

Nikolai Stein

Solution Architect, Requirements Experte & Geschäftsführer

- Seit 2002 im Bereich des RE&M mit DOORS im Automobilsektor tätig
- Seit 2007 Senior Consultant & Geschäftsführer der REQUISIS GmbH
- RE&M-Methoden und –Konzepte
- Daten-Austausch mit ReqIF und OSLC
- DNG-Schnittstellen, -Automatisierung und Migrations-Szenarien
- Global Configuration mit DNG
- Sicherheitsanalysen

requisis

Agenda

- Kurzeinführung Konfigurationsmanagement in DNG
- Global Configuration Management in DNG
- Baureihenentwicklung vs. konfiguriertes Produkt
- Handhabung von Varianten
- Konsequentes Vorgehen und effiziente Nutzung
- Benötigte Rollen

requisis

REQUISIS GmbH

- Bundesweit agierendes Unternehmen mit Standorten in Berlin und Stuttgart
- Spezialist für toolunterstütztes Anforderungsmanagement und individuelle pragmatische Lösungen
- Viele Projektkunden, z.B. im Automobilsektor
- Diverse Addons für DOORS
- Aktuelle Mission
  - Entwicklung eines Tools für eine sanfte Migration von DOORS nach DNG

4

requisis

Vorwort

- **Erster Eindruck**
  - Sehr kompliziert, sehr komplex, unhandlich.
- **Nach längerer Beschäftigung mit dem Thema**
  - Es bietet unglaublich viele Möglichkeiten!
  - Das Prinzip „Teile und Herrsche“ wurde durchgängig umgesetzt.
  - Die Komplexität lässt sich daher gut aufteilen, kapseln und beherrschbar machen!
- **Ich beleuchte im Vortrag die Möglichkeiten, welche Out-of-The-Box funktionieren.**
  - Keine Lösungen durch Dritthersteller
- **Ich gehe nicht auf Schwächen des Tools selbst ein! :-)**

requisis

Einführung in Konfigurations-Management in DNG

- **In DNG ist es möglich**
  - Stände einzufrieren (Baseline, Referenzversion)
  - An verschiedenen parallelen Entwicklungszweigen zu arbeiten (Stream)
  - Änderungen in Änderungsmengen (Change-Sets) zu sammeln, bevor diese anderen zur Verfügung gestellt werden. Diese können an Change-Management-Elemente oder Tasks geknüpft sein.
  - Einzelne Änderungsmengen auch anderen Entwicklungszweigen bereitzustellen (selektiver Merge)
- **Das ermöglicht**
  - Rückverfolgbarkeit
  - Abbildung von Varianten und Versionen
  - Merging & Branching
  - Paralleles, entkoppeltes Arbeiten
- **Konfigurationsmanagement muss im jeweiligen Projekt erst aktiviert werden!**

requisis

Lokales Konfigurationsmanagement

- Mehrere Components in einem Projekt möglich
- Jede Component kann beliebig viele Streams (=Branches, Entwicklungszweige) haben.
- Ein Stream kann vom aktuellen Stand eines Streams oder von einer Baseline ausgehend erzeugt werden – also auch nachträglich!
- Stände können als Baseline eingefroren werden!

requisis

Change-Sets

- Jede Änderung erzeugt automatisch ein Change-Set (=Änderungsmenge).
- Es ist jedoch ratsam, selbst ein Change-Set zu starten und zu benennen und nach Fertigstellung bereitzustellen.
  - So lange bleiben die Änderungen unsichtbar und können ggf. auch verworfen werden.
  - Change-Sets können ausgewählt und anderen Streams bereitgestellt werden (Realisierung von gemeinsamen Anforderungen in Varianten oder Versionen).
- Change-Sets können dem aktuellen Stream, aber auch anderen Streams bereit gestellt werden.
  - Dadurch können z.B. Anforderungen, die für mehrere Varianten gelten, nach dem Verfassen gleich auf alle Streams verteilt werden.
  - Gleiches gilt natürlich auch, wenn man einen Fix für eine ältere Version erstellt, der aber auch für die aktuelle Version relevant ist.
  - Achtung: Change-Sets können nur vollständig bereitgestellt werden!

requisis

Global Configuration Management

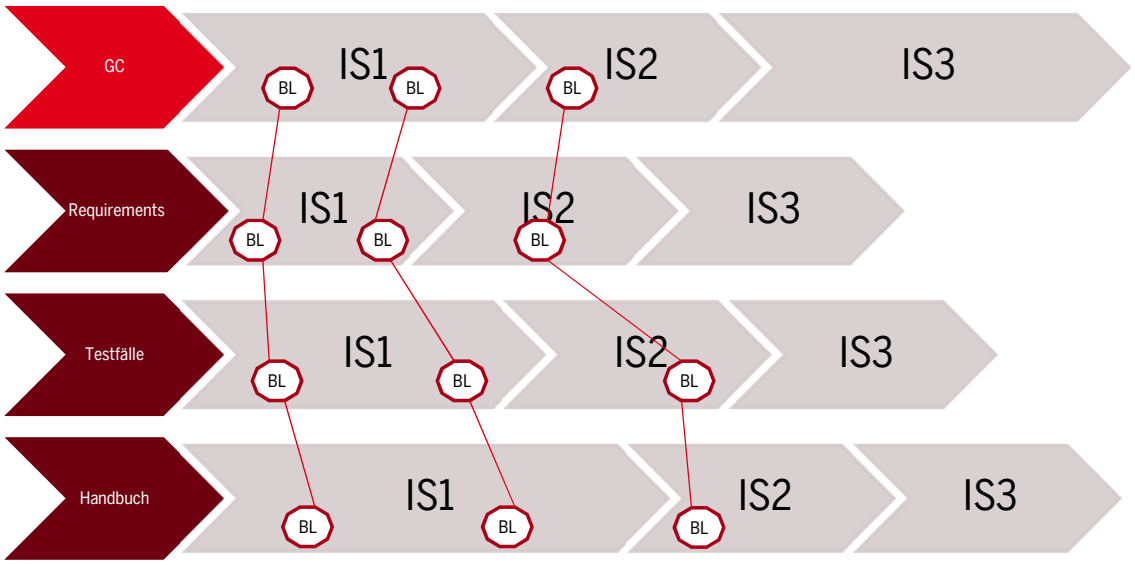
requisis

Global Configuration Management

- **Vereinheitlicht das Versions- und Variantenmanagement über verschiedene Jazz-Applikationen und verschiedene Zeitschienen!**
- **Beispiel**
  - Anforderungen, Usecases, Testfälle entstehen naturgemäß zu verschiedenen Zeiten.
  - Es gibt jedoch eine Beziehung zwischen einer Version der Anforderungen und des Testfalls.
- **Diese zusammengehörigen Entwicklungszweige (Streams) können in einer Global Config zu einem GC-Stream zusammengefasst werden.**
- **Es wird dabei eine Beziehung zwischen einem Globalen Stream und einem Lokalen Stream hergestellt und in allen Applikationen der globale Stream angezeigt.**
- **Auch mit Baselines ist das möglich.**
  - Auch nachträglich können Baselines einem GC-Bereitstellungsstream zugeordnet werden.

requisis

Beispiel zeitlich verzahnte Entwicklung



requisis

Global Configuration Management als Strukturmodell

- **Neben der Bereitstellung von übergreifenden Streams und Baselines, dienen die GC-Component-Streams aber auch der Strukturierung des Produkts!**
- **Dazu verweist ein Component-Stream auf andere Streams, die er baumartig unter sich aufhängt.**
- **Jeder Stream einer Component kann auf Streams von anderen Global Configuration Components verweisen.**
  - Entstehung eines Konfigurations-Baum (z.B. PKW->Innenraum->Innenlicht->Dachbedieneinheit)
  - Abbildung der Produktstruktur oder Architektur
  - Verteilung der Verantwortung durch „Teile und Herrsche“
  - Jeder Knoten im Baum ist ein Stream einer GC-Component
    - Die Unterknoten sind im Stream des oberen Knoten referenziert -> mögliche Struktur-Varianz
- **Jede Component kann auf Streams von anderen Applikationen (DNG, DM, QM, CCM) verweisen**
  - Einbringung der Entwicklungsartefakte in den Produktbaum (z.B. in Configuration Innenlicht -> RM Dachbedieneinheit)

requisis

GC-Struktur mit Components

requisis

Global Configuration Management Component Streams

- Eine Global Configuration enthält wie andere Applikationen auch: Components.
- Bei der Erstellung entsteht automatisch 1 Entwicklungszweig (Stream).
- Es können aber auch beliebig viele weitere Streams erzeugt werden.
  - Jede Component kann also in verschiedenen Varianten existieren.
  - Abbildung von Baureihen durch Baureihenstreams
  - Abbildung von Produktvarianten durch Varianten-Streams
- In jedem Stream ist festgelegt, welcher Stream welcher Component in der nächsten Ebene zu finden ist.
  - Dort ist dann wiederum festgelegt, auf welche Streams in der Ebene darunter verwiesen ist.

requisis

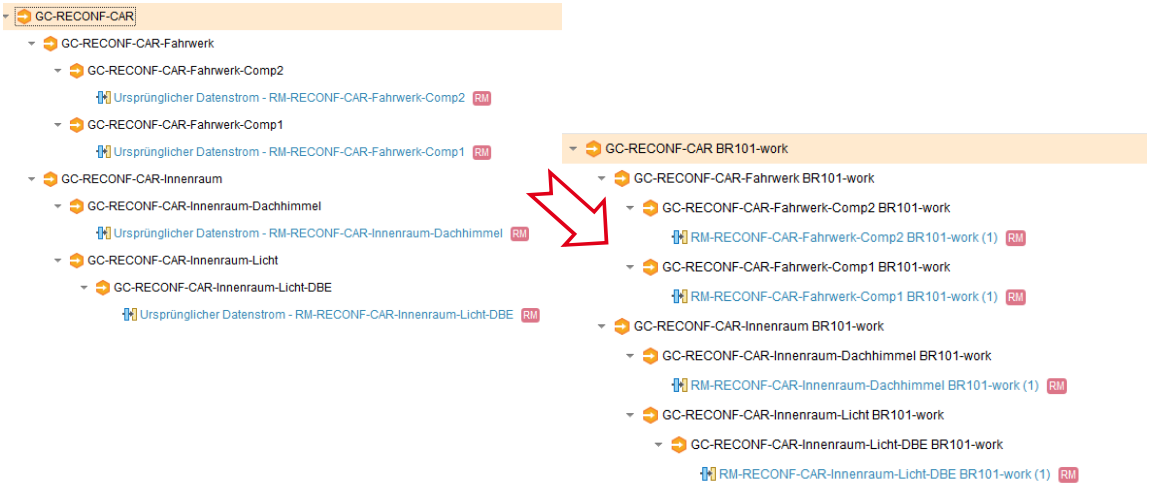
Baureihenentwicklung vs. Konfiguriertes Produkt

- **Baureihenentwicklung**
  - An fast allem wird gearbeitet.
  - Meist mit Übernahme einer Vor-Version vom Vorgänger-Produkt
  - Es wird zu Beginn auf jeder Ebene ein neuer Stream erstellt.
  - GC hilft hier bei dem Zusammenhalten von allen Streams einer Baureihe und dem Anlegen von neuen Streams.

- **Konfiguriertes Produkt**
  - Ein Produkt wird aus verschiedenen „Standard-Komponenten“ und deren Varianten zusammengebaut.
  - Großteil bleibt unverändert, es werden Standard-Komponenten-Anforderungen übernommen.
  - Zu Beginn gibt es nur einen neuen Stream auf oberster Ebene und auf allen Ebenen, unter denen Anpassungen notwendig sind, d.h. wo Standard-Konfigurationen durch Varianten ausgetauscht werden oder Anpassungen gewünscht sind.
  - Verwendung von festen Referenzversionen statt Streams, wo keine Anpassung notwendig ist.
  - **Enormer Vorteil von DNG durch die Fähigkeit von DNG „Struktur-Varianz“ (Wiederverwendung von ganzen Teilstrukturen) abzubilden.**

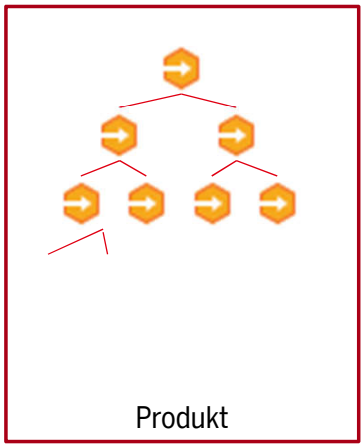
requisis

Erstellung neuer Stream über ganze Struktur

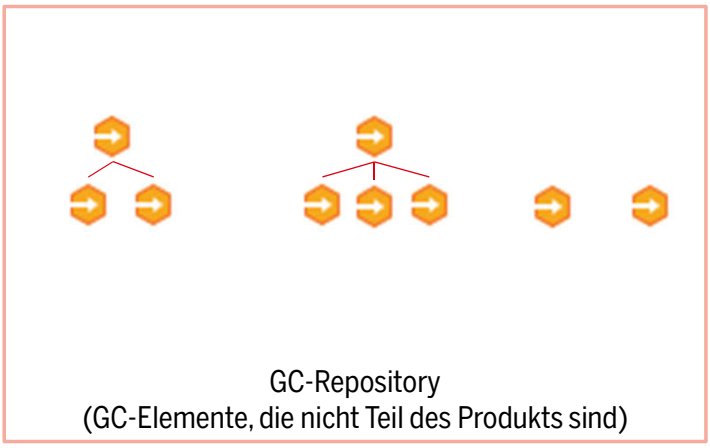


requisis

Folie: Konfiguriertes Produkt



Produkt



GC-Repository
(GC-Elemente, die nicht Teil des Produkts sind)

requisis

Exkurs Variantenmanagement (1)

- **Verschiede Arten von Varianzen**
  - Zeitlich: Baureihen, Versionen
  - Inhaltlich: „Varianten“
- **Varianzen im Automobilbereich:**
  - Baureihen: weitgehend entkoppelte Entwicklung, meist Übernahme aus „vorheriger Baureihe“ des „Vorgänger-Produkts“
  - Varianten (ein kleiner Ausschnitt):
    - Unterschiedlicher Funktionsumfang einer ansonsten vergleichbaren Komponente (z.B. klassisches Kombiinstrument oder ein volldigitales Kombi)
    - Fahrzeugvarianten (Kombi, Limousine, Cabrio)
    - Motorvarianten (Diesel oder Benziner)
- **Abbildung durch:**
  - Klassisch: Verschiedene Dokumente oder Dokument mit „Variantenspalte“
  - Mit DNG: Streams, verschiedene Module oder Components oder Variantenattribut

requisis

Exkurs Variantenmanagement (2)
Möglichkeiten der Variantenabbildung

- **Streams**
  - Bei große Unterschieden
  - Gewünschte entkoppelte Entwicklung
  - Höherer Pflegeaufwand für Gemeinsamkeiten ist akzeptabel bzw. es werden nur wenige Anforderungen dem anderen Stream bereitgestellt
  - Feste Abweichungen zum „Standard-Stream“ und Übernahme aller Änderungen aus dem Stream (z.B. länderspezifische Anpassung)
  - Unpraktisch, wenn Übergabe der Anforderungen an einen Zulieferer geplant, der mehrere Varianten umsetzen soll!
  - Königsweg bei „Konfiguration“ eines Produkts aus Standardkomponenten mit Anpassungen

- **Variantenattribut**
  - Bei geringen Unterschieden der Varianten oder rein additiv, also mehr Funktionen im Vergleich zur anderen Variante
  - Keine entkoppelte Entwicklung gewünscht
  - Alle/mehrere Varianten werden vom gleichen Zulieferer hergestellt
  - Höherer Pflegeaufwand für Gemeinsamkeiten ist nicht akzeptabel
  - Widersprüche in den Varianten durch Dopplung und variantenspezifische Anpassung einer Anforderung gut transparent darstellbar

requisis

Exkurs Variantenmanagement (3)
Möglichkeiten der Variantenabbildung

- **Verschiedene Module**
  - Gekoppelte Entwicklung
  - Gut teilbare Umfänge: z.B. in gemeinsame Anforderungen, Anforderungen High-Line, Anforderungen Low-Line

- **Verschiedene Components (Königsweg für Wiederverwendung und Einbau im GC-Baum)**
  - Gekoppelte oder entkoppelte Entwicklung
  - Gut teilbare Umfänge: z.B. in gemeinsame Anforderungen, Anforderungen High-Line, Anforderungen Low-Line

requisis

Exkurs Variantenmanagement (4)

- **Varianten sind notwendig, diese müssen dokumentiert werden**
  - Marketing / Kostenreduktion: low, mid, high Varianten
  - Lokalisierung (Länderunterschiede)
- **Abbildung von Abhängigkeiten**
  - Bestimmte Funktionen benötigen andere Funktionen oder Eigenschaften
  - Verschiedene Möglichkeiten: Links, Boolsche Ausdrücke, Einsatz eines Variantenmanagement-Werkzeugs hilfreich
  - Prüfung

requisis

Lokale Varianten als Stream im GC-Modell (1)

- **Lokale Varianten sind Varianten, die nicht für das ganze Produkt gelten, sondern nur für eine oder wenige Components!**
- **Das GC-Modell ermöglicht im Prinzip 2 Dimensionen:**
  - Stream-Baum (durch Erstellen von Streams auf Basis von anderen Streams)
  - Struktur-Baum (durch Einbindung von anderen Global Configurations Streams und Streams von Components)
- **Lokale Varianten passen daher nicht so recht in den GC-Baum**
  - Produktvarianten können im Stream-Baum abgebildet werden.
    - Funktioniert aber bei Varianten, die für alle beteiligten Components gelten.
  - Es kann nur eine Variante einer Component in das GC-Modell einbezogen werden, da es sonst Komponentenüberschneidungen gibt.

requisis

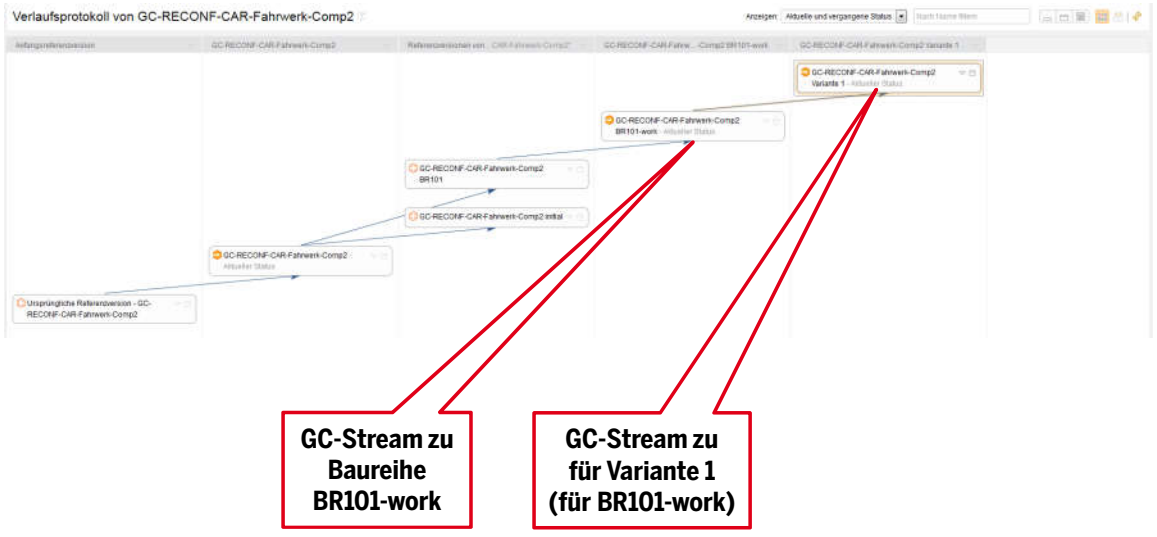
Lokale Varianten als Stream im GC-Modell (2)

- **Mehrere Varianten einer Component im GC-Modell unterbringen**
  - Eigene GC-Component, in der alle Varianten-Streams angelegt sind.
    - In der übergeordneten GC-Schicht nur auf den „allgemeinen“ Stream der Variante verweisen.
    - Nicht im GC-Baum verwiesene Streams (und von diesem verwiesene Streams) sind nicht im Baum sichtbar, aber in der Stream-Liste der GC-Komponente aufrufbar. Sind sozusagen losgelöst/detached.

- **Alternativer Königsweg:**
  Components kleiner und disjunkt halten:
  - z.B. Comp1-Common, Comp1-Hi, Comp1-Low
  - Alle Components können in den GC-Baum eingehangen werden
  - Sie haben keine Überschneidungen und sind noch besser wiederverwendbar!

requisis

Darstellung eins GC-Stream „Stammbaums“



requisis

Konsequentes Vorgehen und effiziente Nutzung

requisis

Global Configuration konsequent einsetzen

- Abbildung der Produktstruktur erfolgt nicht im Verzeichnis-Baum sondern in der Global Configuration Struktur!
- Abbildung von Produkt-/Baureihen durch Ausrollen von Streams auf allen Ebenen des GC-Modells und der Components der Applikationen
- Navigation zum richtigen Stream der eigentlichen Komponente erfolgt über die GC-Struktur, welche Produkt- und Baureihenstruktur abbildet. (Navigation wird derzeit in diese Richtung umgestellt!)
- Abbildung von Varianten durch Teilung von Components oder Baureihenattributen
  - Aber: Streams für Varianten bei Produktvarianten oder bei konfigurierten Produkten benutzen

requisis

Vorschläge zur Arbeitserleichterung

- **Produktstruktur mit sauberem Ursprungsstream der Components selbst in eigenem Stream pflegen.**
  - Neue Baureihen davon ausgehend erzeugen.
  - Dadurch lassen sich Streams für neue Baureihen einfach über alle beteiligten Elemente erstellen.
  - Nachträglich kann dann jede Komponente durch Bereitstellung von Änderungen auf die gewünschte „Start-Konfiguration“ aktualisiert werden.
- **Für jede Component eine Ur-Baseline erzeugen.**
  - Damit später davon ausgehend Streams erzeugt werden können.
- **Arbeit nur in Änderungsmengen (Change-Sets) zulassen und Thematisch benennen.**
  - Folgen von Fehlern (wie Löschen von Artefakten) haben begrenzte Auswirkung, da eine Änderungsmenge verworfen werden kann.
  - Änderungsmengen können selektiv anderen Varianten-Streams bereitgestellt werden.
- **Lokale Varianten Streams nur bei Entkopplung einsetzen.**
  - Reduktion der Komplexität für die Bearbeiter!

requisis

Benötigte neue Rollen bei Baureihenentwicklung

- **Konfigurations-Administrator / Konfigurations-Leiter**
  - Kann auf allen Ebenen des GC-Baumes unterschiedlich sein.
  - Aufgaben:
    - Struktur anlegen
    - Streams anlegen
    - Konsistenz prüfen
    - Bereitstellung von Baselines zu Globalen Referenzversionen (z.B. Integrationsstufen) prüfen
- **Varianten-Manager**
  - Lokal pro Component oder Funktionsbereich
  - Aufgaben:
    - Legt Varianten an
    - Prüft den „Fluss“ von Anforderungen zwischen den Varianten (Bereitstellen von relevanten Änderungsmengen)
    - Sorgt für die Bereitstellung der Varianten-Streams in die jeweiligen Baureihen-Streams

requisis

Fazit

- Configuration Management in DOORS Next ist ein flexibles, umfangreiches und komplexes Werkzeug
  - Jedoch: Nicht für jeden Anwendungsfall ist es das beste Mittel!
- Das Verständnis fällt einfacher, wenn man Versions- und Configuration-Management aus der Software-Entwicklung kennt (z.B. mit GIT)
- Man benötigt ein klares Verständnis, um Global Configuration sinnvoll einzusetzen und eine passgenaue Vorgehensweise zu entwickeln!

requisis

Vielen Dank für Ihre Aufmerksamkeit.

- Besuchen Sie auch unseren Messestand!
- Fragen Sie uns nach unserem Migrations-Tool!
- Für Fragen und Anregungen stehe ich gerne zur Verfügung!
- Nikolai Stein-Cieslak
nikolai.stein@requisis.com
+49 (30) 536506-711

30