Generischer Konfigurator

Gennadi Heimann

29.12.2017

1 Logik Beschreibung

1.1 Abhängigkeiten

Die Abhängigkeiten im Konfigurator bilden die Regel. Jede Komponente hat eine Ausgangsabhängigket (outDependency) und eine Eingangsabhängigket (inDependency). Die Ausgangsabhängigket bewirkt auf ihre Komponente mit der Regel die in der Abhängigkeit hintergelegt ist. Es gibt zwei Arten (dependencyType) von Abhängigkeiten $< exclude >$ und $< require >$. Die $< exclude >$ Abhängigkeit schließt die Komponente aus der Konfiguration aus. Die $< require >$ Abhängigkeit fordert wiederum, dass die Komponente in die KOnfiguration hinzugefügt werden muss. Das Verhalten der Abhügigkeiten wird von Parameter $< visualization >$ gesteuert. Der Benutzer des Konfigurators wählt die Komponente innerhalb eines Schrittes und die Logik das Konfigurators prüft die Abhängigkeiten und anhand der Parameter wird weitere Verlauf gesteuert. In der darauffolgende Beschreibung werden einzelne Parameter erklärt.

- visualization bei der $< exclude >$ Komponenten
  - auto
    - Die Komponente wird automatisch aus der Konfiguration entfernt.Wenn die ausgeschlossene Komponente weiter auf eine $< exclude >$ Komponente verweißt, werden diese Komponente recursive ausgeschlossen. Das bedeutet, dass bei der späterem Verlauf werden diese Komponente nicht gezeigt. Bei der $< require >$ Komponenten wird dieses Vorgehen nicht angewendet. Die Komponente wird nicht aus der Konfiguration entfernt.
  - marked + selectable
    - Die IN-Komponente wird visual merkiert, dass sie $< exclude >$ Komponente ist und der Benutzer hat die Möglichkeit diese Komponente auszuwählen. Wenn der Benutzer w¨hlt diese Komponente aus, wird ein Hinweis dem Benutzer vorgeschlagen, wie er die Abhängigkeiten in seiner Konfiguration lösen kann. Es wird eine andere OUT-Komponente aus dem gleichem Schritt vorgeschlagen. Wenn in diesem Schritt keinen Vorschlag gemacht werden kann wird weiter in dem bergeordneten Schritt weiter gesucht. Bei jedem vorgeschlagenen Komponente wird weiterhin nach Abhängigkeiten geprft.
  - marked + unselectable
    - Die Komponente wird visuel markiert aber kann nicht ausgewählt werden. Die Abhängigkeiten werden gleich wie bei der Visualizationparameter $< auto >$ behandelt.
- require
  - auto

    * Die Komponente wird automatisch in die Konfiguration hinzugefügt. Es wird geprüft, ob der IN-Komponente der Abhängigkeit weitere $< require >$ oder $< exclude >$ Komponente hat. Wenn ja, wird gemäß der Parameter in der Abhängigkeit die Konfiguration angepasst. Wenn die IN-Koponente ber mehrere Schritt in der Konfiguration entfernt ist. Der Konfigurator wird, dann in Laufe der Konfiguration bei jedem Schritt dem Benutzer einen Hinweis geben, welche Komponente der Benutzer auswählen kann/muss um zu dem Schritt mit der $< require >$ Komponente zu kommen. Die Komponente wird visual markiert, dass sie schon zu der Konfiguration hinzugefugt wurde. Diese Komponente ist auch $< unselectable >$. Bei der Abhängigkeit in gleichem Schritt wird die Komponente ohne jegliche Hinweise zu der Konfiguration hinzugefügt und dementsprechend markiert.

- selectable
    * Bevor die Komponente zu der Konfiguration hinzugefügt wird, wird dem Benutzer einen Hinweis mit dem Auswahl gegeben. Der Benutzer kann entscheiden ob er die required Komponente zu der Konfiguration hinzufügen will.

Die aktuelle Konfiguration behält die gesamte Information (Abhängigkeiten) von der hinzugefügten Komponente. Diese Information hilft dem Konfigurator die Prüfungen durchführen, wenn der Web-Client Fehler macht. Zum Beispiel, wenn der Web-Client dem Benutzer eine Komponente auszuwählen erlaubt, die von Server als $< unselecteble >$ oder $< remove >$ markiert wurde. In diesem Fall bekommt der Benutzer die Fehlermeldung.

1.2 Auswahlkriterium

Die Anzahl der Komponente die innerhalb eines Schrittes ausgewählt werden knnen, werden ber den Parameter $< SelectionCriterum >$ gesteuert. Dieses Kriterium definiert fr jeden Schritt die maximale und minimale Anzahl der Komponente, die von dem Benutzer ausgeählt werden können. Das Auswahlkriterium wird nach demPrüfung der Abhängigkeiten durchgefhrt. Es werden follgende Fälle unterschieden:

- RequireComponent
    - Wenn die Anzahl der ausgewählten Komponente kleiner als minimale Kriterium. Der Benutzer wird aufgefordert weiter Komponente auszuwählen, bis minimale Kriterium erreicht wird.
      min > countOfComponents
- RequireNextStep

  - Wenn alle Komponente in dem Schritt ausgewählt sind, wird der Konfigurator zu dem nächstem Schritt führen.
    max == countOfComponents
    Wenn der minimale Kriterium gleich Null und maximale Kriterium größer Eins ist, kann der Benutzer ohne den Auswahl der Komponente zu dem nächsten wechseln.
    min == 0 and max > 1
    min == 0 and max == 0 ist nicht erlaubt.

- AllowNextComponent
  - Wenn dem Benutzer noch weitere Komponente auszuwählen erlaubt. In diesem Fall kann der Benutzer weitere Komponente auszuwahlen oder zu dem nächstem Schritt wechseln.
    min <= countOfComponents and
    max > countOfComponents

- ExcludeComponent
  - Wenn der Benutzer mehr Komponente als erlaubt auswählen mchte.
    max < countOfComponents

<**countOfComponents**> = Anzahl der vorher ausgewählten Komponenten. Die Anzahl der Komponenten wird aus der aktuellen Konfiguraion ermittelt

Die Variable $< countOfComponents >$ wird vor der Prfung auf die Auswahlkriterium immmer angepasst. Die Ausgewählte Komponnete wird zuerst mit aktuellen Konfiguration verglichen.

Wenn die ausgewählte Komponente nicht in der bestehenden Konfiguration exestiert, wird $< countOfComponents >$ auf 1 inkrementiert. Somit wird temporär die ausgewhlte Komponnte zu der aktuellen Konfiguration hinzugefgt. Nach dem alle Prüfungen erfolgreich abgelaufen wird diese Komponnete zu der aktuellen Konfiguration hinzugefgt.

Wenn die ausgewählte Komponente in der aktuellen Konfiguration gefunden wird. Das bedeutet, dass die Komponnete abgewählt wird und uas der Konfiguration entfernt werden muss, deswegn wird $< countOfComponents >$ nicht verändert.

1.3 Hinweisfeld zu jeder Aktion

Der Server liefert bei jedem Aktion die Erklrungen zu diesen Aktion, deswegen bei dem Webclient sollte einen Bereich geben wo diese Information Prsentiert werden kann.

1.4 Auswahl der Komponente

Der Server bekommt von dem Web-Client die ausgewählte Komponente.

1. Die Komponente $< Vertex >$ aus der DB lesen. $< vComponent >$
2. Alle ausgehende und eingehende Abhängigkeiten $< Edge >$ dieser Komponente aus DB lesen.
3. Die Abhängigkeiten werden auf folgende Arten gefiltert:
   - Die $< exclude >$ ausgehende Abhängigkeiten $< excludeDependenciesOut >$
   - Die $< exclude >$ eingehende Abhängigkeiten $< excludeDependenciesIn >$
   - Die $< require >$ ausgehende Abhängigkeiten $< requireDependenciesOut >$
   - Die $< require >$ eingehende Abhängigkeiten $< requireDependenciesIn >$
4. Zu jede Abhängigkeit werden dazugehörige abhqangige Komponente ermittelt.
5. Der Vaterschritt der ausgewählten Komponente wird aus der DB abgefragt. $< vFatherStep >$
6. Aus diesem Vaterschritt wird das Auswahlkriterium ausgelesen. $< selectionCriterium >$
7. Der aktuelle Schritt wird aus der Aktuellen Konfiguration ebenso ausgelesen. $< currentStep >$
8. Im aktuellem Schritt aus der aktuellen Konfiguration wird geprüft, ob die ausgewählte Komponnte in vorherigen Aktionen ausgewählt war. Ausführende Funktion $< checkSelectedComponent(currentStep, componentId) >$
   - JA $- >$ Die Komponente aus der aktuellen Konfiguration wird gelöscht. Der Status ist $< RemovedComponent >$
   - NEIN $- >$ Der Status ist $< AddedComponent >$
9. Aus der aktuellen Konfiguration wird die Anzahl davor ausgewählten Komponente ermittelt und mit Auswahlkriterium verglichen. Die ermittelte Entscheidung wird unter Abhängigkeiten abgeklärt.
10.