Modellgetriebene Berechnung

existenzsichernder Zusatzleistungen zur AHV/IV.
Projekt ZLPro im Amt für Zusatzleistungen zur AHV/IV (AZL), Stadt Zürich.

Björn Reber / Bruno Böni
Zürich / 23.10.2017

Consulting
IT Solutions

emineo

1. Thematische Einführung
2. Fachliche und organisatorische Herausforderungen
3. Lösungsfindung
4. Modellieren der Berechnungslogik über ein Regelwerk

Was sind Zusatzleistungen?

Die Zusatzleistungen sind bedarfsorientierte Sozialversicherungsleistungen und gehören zusammen mit der **AHV** und **IV** zum sozialen Fundament unseres Staates. Das System der Zusatzleistungen wurde für AHV- und IV-Rentenberechtigte geschaffen, die in **finanziell bescheidenen** Verhältnissen leben oder **hohe Heimkosten** zu bezahlen haben. Ziel ist, ihnen ein Leben ohne materielle Existenzsorgen zu ermöglichen.

Zusatzleistungen zur AHV/IV bestehen aus

- Ergänzungsleistungen (Bund)
- Beihilfen und Zuschüsse (Kanton)
- Gemeindezuschüsse (Gemeinde)
- Krankheits- und Behinderungskorsten

Zusatzleistungen zur AHV/IV		
Vom **Bund** verordnete Leistungen	Vom **Kanton Zürich** verordnete Leistungen	Von der **Stadt Zürich** verordnete Leistungen
Ergänzungsleistungen	Beihilfen	Gemeindezuschüsse
Krankheits- und Behinderungskosten	Kantonale Zuschüsse	Pflegekosten-zuschüsse
		Ausserordentlicher Gemeindezuschuss

Wie berechnen sich diese Leistungen

- Einkommen und Vermögen wird Ausgaben gegenübergestellt.
- Differenz zu einem Mindestbetrag wird als Ergänzungsleistung (Rechtsanspruch) bzw. Beihilfe und/oder Gemeindezuschuss festgelegt.
- Eigentlich eine "umgekehrte" Steuererklärung.

Schwierigkeiten in der Praxis

- Viele Einflussfaktoren
  - Karenzfristen
  - Freibeträge
  - Alterskategorien
  - Regionale Unterschiede (Land/Agglo/Stadt)
  - Prämienverbilligung Krankenkasse (Systemwechsel)
- Teilwiese komplexe Berechnungen von
  - Vermögensverzehr
  - Liegenschaften
  - Anrechenbarem Einkommen

Fachliche Herausforderung

- Es besteht ein gesetzlicher Anspruch auf Zusatzleistungen, umfangreiche Rechtsgrundlagen und Praxen.
- Ändernde Gesetzesgrundlagen auf Ebene Bund, Kanton und Gemeinde(n).
- Unterschiedliche Anwendung in den einzelnen Ausführungsstellen (sprich Gemeinden).
- Rechtzeitige Ausschüttung der monatlichen Betreffnisse, genaue Berechnung von fälligen Rückforderungen und/oder Nachzahlungen.
- Verfügung der Rechtsansprüche, Dokumentation.
- Steigende Anforderungen an Schnittstellen-Integration in Umsysteme auf Ebene Bund, Kanton, Gemeinde(n).

- Generell eine komplexe Fachmaterie.

Weshalb braucht es eine neue Lösung

- Ende des Lifecycles des benutzten Legacy-Systemes ist erreicht, keine strategische Plattform mehr.
- Anpassungen im Legacy-System sind mit grösseren Aufwänden verbunden
- Erweiterbarkeit des Legacy-Systems ist generell beschränkt.
- User Experience des Legacy-Systems antiquiert.
- Letzte Know-How-Träger beenden ihr Berufsleben – Brain Drain.

Fachliche und organisatorische Herausforderungen

Aufgabe an das Interdisziplinäre Team

Aufgabe:

Realisierung der Berechnungsengine innerhalb einer Frist von 6 Monaten, Berücksichtigung der bekannten Eingabe- und Ausgabewerte (Black-Box-Betrachtung).

Eingabe: Fallinformationen (aus Legacy-System), Rechtsgrundlagen/Parameter/Regeln, angewandte Praxen.
Ausgabe: Berechnete Rechtsansprüche (Zusatzleistungen) zum Vergleich.

Ziele:
- 3 Monate: 80% der laufenden Fälle richtig berechnet.
- 6 Monate: 100% der laufenden Fälle und alle Fälle rückwirkend bis Jahr 2004 richtig berechnet.

Interdisziplinäres Team bestehend aus:

- Organisation und Informatik der Stadt Zürich: Projektleitung, Vorgaben, Betriebliche Aspekte, Betrieb und Unterhalt.
- Amt für Zusatzleistungen der Stadt Zürich: Projektleitung, Fachanforderung und Branchenwissen, Spezifikation der Regeln.
- emineo: Projektleitung, Umsetzung Fachliche- und Betriebliche Anforderungen.

Lösungsfindung

Gemeinsame Sprache finden

- Interdisziplinäres Team muss ein gemeinsames Ziel haben
  - Schulung des Teams (alle werden «kleine» Sachbearbeiter)
  - Spezifikation der Berechnungsregeln
  - Formulieren der Eingabedaten
  - Vision des «Resultats»
- Interdisziplinäres Team muss sich austauschen können
  - Jedes «Ding» hat einen Namen
  - Jedes «Ding» hat eine Definition und eine Abgrenzung
    - wird es berechnet, eingegeben oder abgeleitet
    - unter welchen Bedingungen ist es relevant für die Ergänzungsleitungen

Lernen aus der Vergangenheit

- Vorgaben durch OIZ
  - Erfahrungen aus ähnlichen Lösungen
  - welches sind strategische, bewährte oder neue Systeme
- Lessions Learned mit Legacy-System
  - Was gefällt Anwender
  - Was sind die grössten Schwächen
- Qualitätssicherung durch rückwirkendes Berechnen
  - ist Fluch und Segen…

Prototyping

- Konzeptioneller Weg für Umsetzung ist offen
  - Wunsch «Interdisziplinäres Team»
    -> alle können erklären wie es funktioniert
  - Wunsch «hochflexibel»
    -> Berechnungslogik ist noch nicht formalisiert, wird während der Umsetzung aus Rechtsgrundlagen und Praxen abgeleitet und laufend verfeinert
  - Wunsch «Verständlichkeit»
    -> wird ein Fehler gefunden, ist für alle klar wieso falsch gerechnet wird, wo die Korrektur anzubringen ist und wie sie sich auswirkt
- Ausprobieren von Lösungsansätzen
  - Umsetzung als «konventionelles» Domänenmodell
    - wirkt schwerfällig und in notwendigem Detailierungsgrad zu technisch
  - Umsetzung mittels «Business Rule Engine»
    - Produkte haben zu breiten Anwendungsfokus, sind zu komplex

«Domain Driven Design»-Ansatz

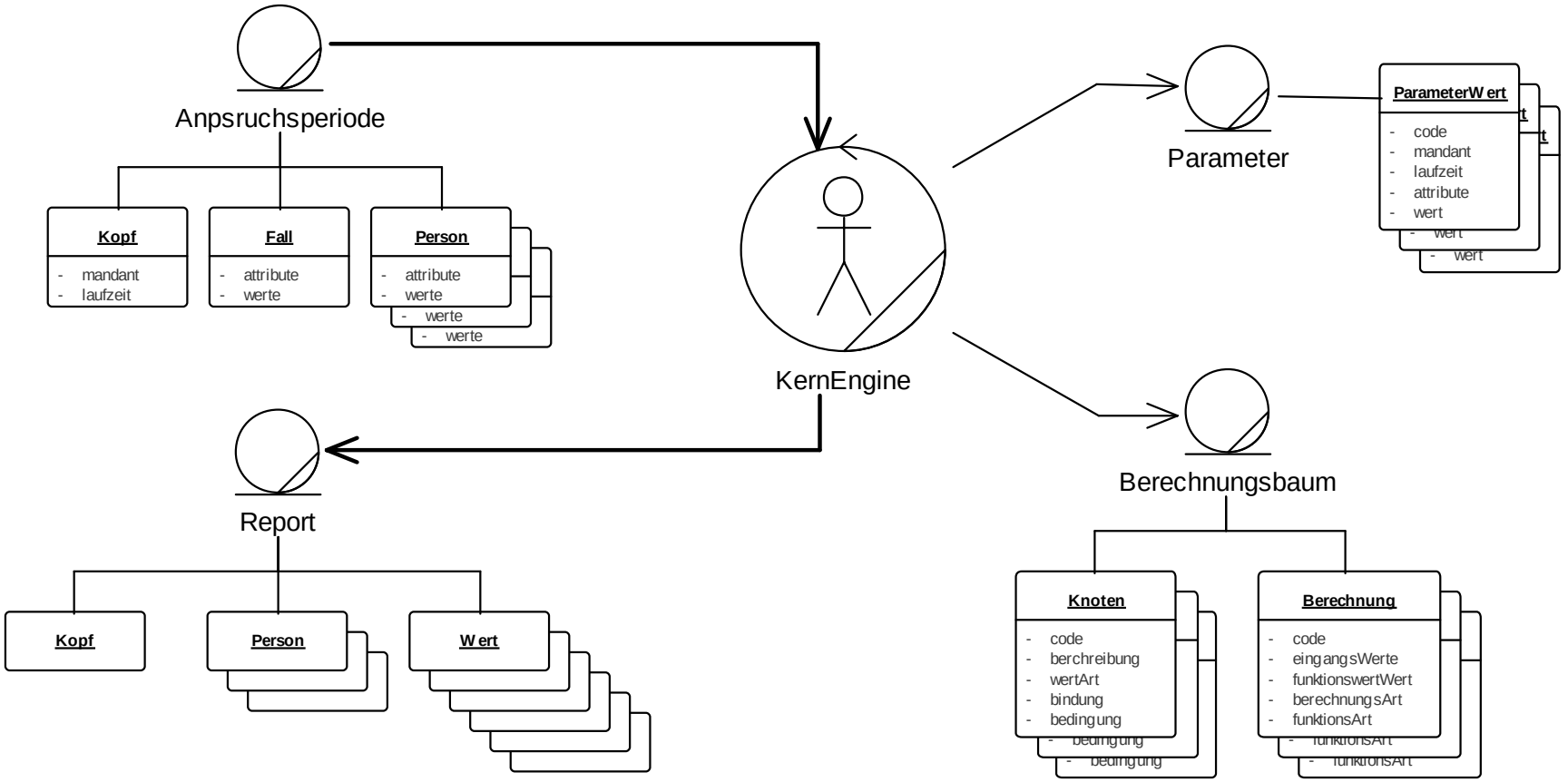
- Bestandteile des Domänenmodells
  - Entitäten
    Personen im Fall mit ihren spezifischen Eigenschaften
  - Werteobjekte
    Einnahmen, Ausgaben, Parameter (Freibeträge, Grenzbeträge etc.)
  - Aggregate
    Rechtsansprüche (inkl. Zwischenresultate) und Regeln wie sie zu Stande kommen
- Modell kann im Team erarbeitet werden
  - Visualisierung und Dokumentation
- «Domain Specific Language» kann abgeleitet werden
  - Deklarative Beschreibung eines Sachverhalts
  - Validierbarkeit
  - Leichte Erlenbarkeit

Mengengerüste

- Domänenmodell
  - Entitäten
    1 – n Personen im Fall (Fallträger, Ehepartner, Kinder)
  - Werteobjekte
    ca. 80 berechnungsrelevante Wertearten (Einnahmen und Ausgaben)
    ca. 70 Parameter
  - Aggregate (Anspruchsherleitung)
    ca. 670 Wertearten (Ansprüche, Zwischenresultate, Parameter und Eingabewerte)
    ca. 20 Attributtypen zur Fallsteuerung mit Total 100 Attributen
- Qualitätssicherung
  - ca. 1/2 Million Testfälle (Anspruchsperioden) über einen Zeitraum von 15 Jahren
  - ca. 150 konstruierte Testfälle

Modellieren der Berechnungslogik über ein Regelwerk

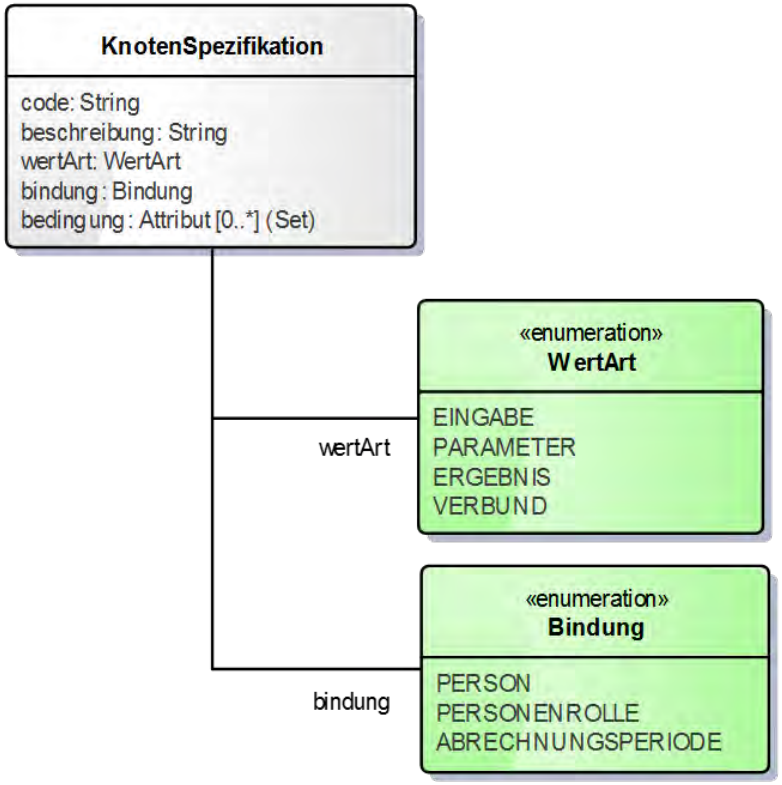
Ausführen einer Berechnung



Elemente

- Anspruchsperiode
  - «Input» für Berechnung von Zusatzleistungen
- Report
  - «Ergebnis» der Berechnung von Zusatzleistungen
- Parameter
  - «Einstellwerte» zur Berechnung von Zusatzleistungen
- Berechnungsbaum
  - «Knoten» beschreiben jede Art von Werten die in der Berechnung von Zusatzleistungen vorkommen können (Eingaben, Parameter, Ergebnisse)
  - «Berechnungen» beschreiben wie Werte für berechnet werden (Summen aus Werten, Auswahl grösster Wert etc.)
  - Hierarchischer Aufbau, an der «Wurzel» gibt es nur **ein** Ergebnis
  - Die Berechnungen erfolgt von «unten» nach «oben»

Eigenschaften eines Knotens

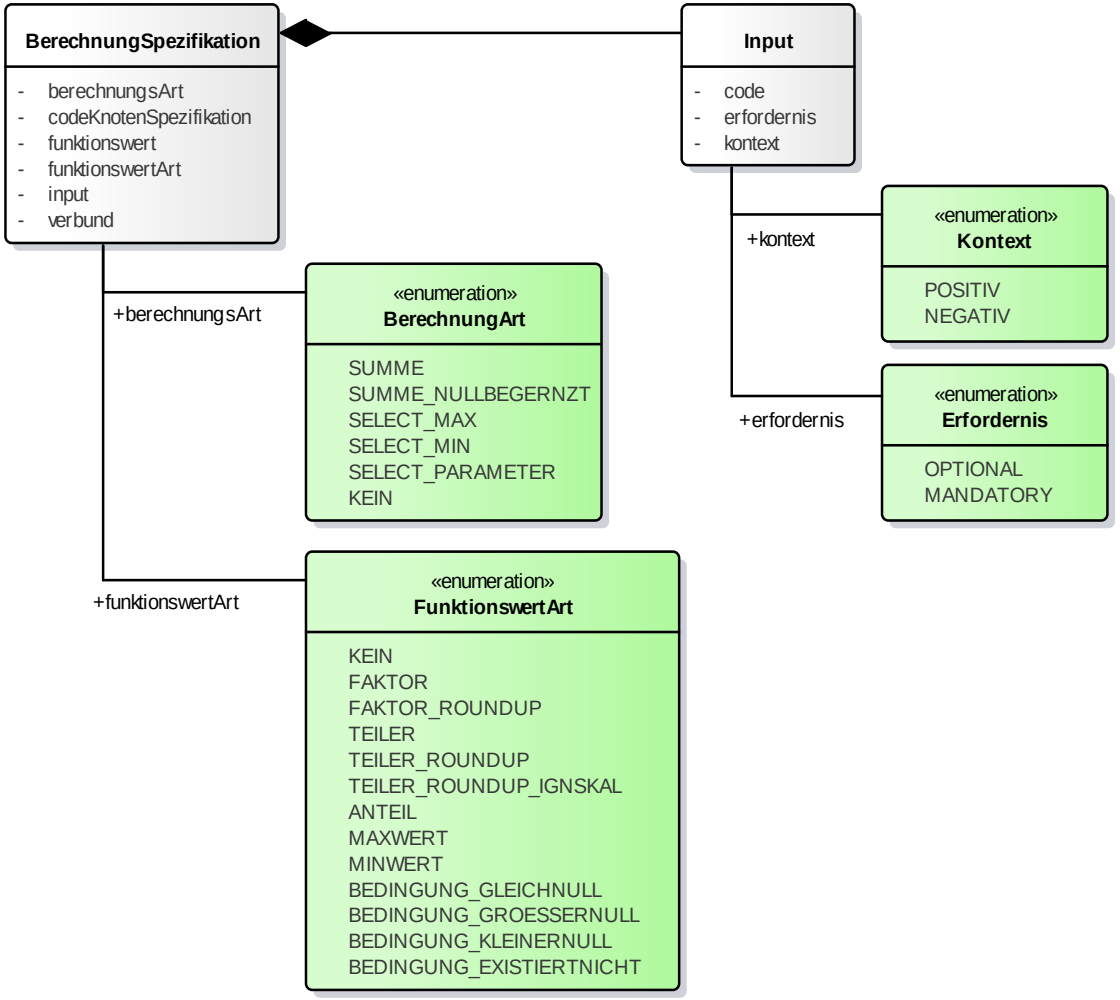


- Code
  Eindeutiger Bezeichner
- Beschreibung
  Erklärungstext
- WertArt
  Definiert die Art des Werts
- Bindung
  Definiert zu wem ein Wert gehört
- Bedingung
  Definiert wann ein Wert existieren darf
  (Attribute = Eigenschaften)

Wenige Regeln

- Code
  - Darf im ganzen Berechnungsbaum nur einmal verwendet werden
- Werte
  - Werte in einer Berechnung werden mit Code der KnotenSpezifikation erfasst
- Bindung: Parameter und Ergebnisse existieren im Kontext nur einmal:
  - «Anspruchsperiode»: ein Wert mit Code innerhalb einer Berechnung
  - «Personenrolle»: pro Personenrolle ein Wert mit Code
  - «Person»: pro Person ein Wert mit Code
  - Ausnahme Eingabewerte: es können beliebig viele Wert mit dem gleichen Code, unabhängig vom Kontext, erfasst werden
- Bedingung
  - Ein Wert wird nur dann in der Berechnung berücksichtigt, wenn die Fall- und Personenattribute im Input (Anspruchsperiode) enthalten sind

Wie ein Knoten vom Typ Ergebnis berechnet wird

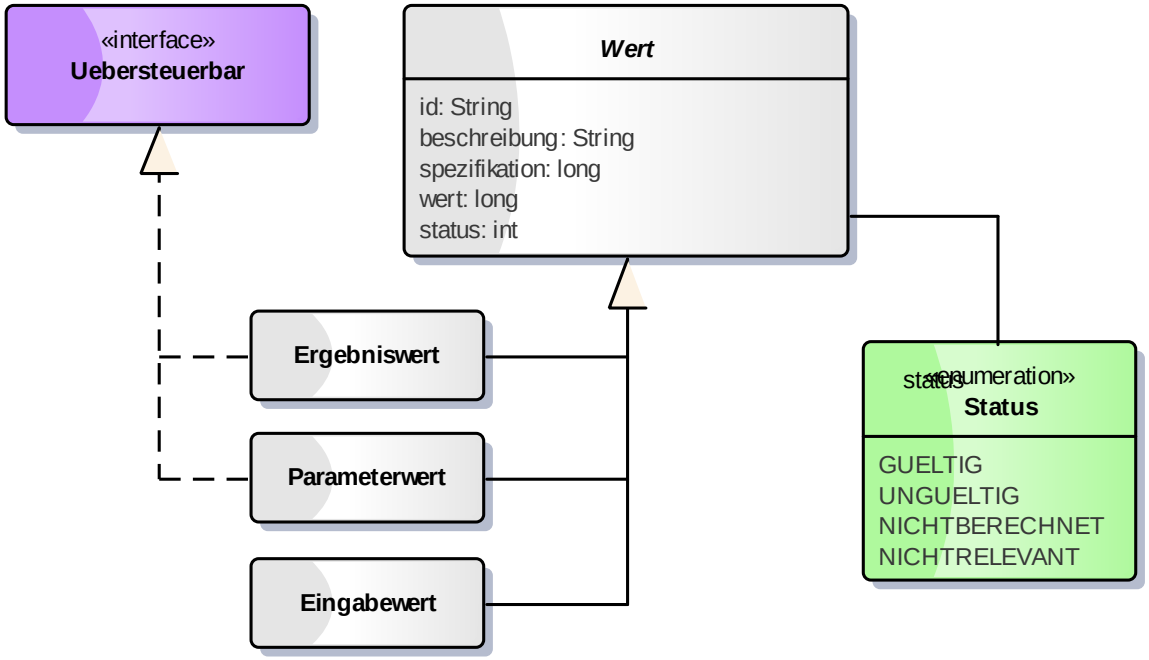


- Input
  Für die Berechnung verwendete Knoten
- BerechnungsArt
  Was passiert mit dem Input
- FunktionswertArt
  Was steuert der Funktionswert

Wenige Regeln

- Input
  - Eine Reihe von Werten die für Summierung oder Auswahl von Max./Min.-Wert verwendet werden
  - «Kontext» steuert ob ein Knoten addiert / subtrahiert oder ignoriert wird
  - «Erfordernis» steuert ob Berechnungsbaum ein Knoten immer anliegen muss oder nur wenn eine entsprechende Eingabe erfolgt ist
- Funktionswert
  - Ist optional
  - Steuert ob ein Ergebnis aus dem Input berechnet werden soll oder eine weitere Berechnung mit dem Ergebnis aus dem Input stattfindet
- Verbund
  - Definiert welche Inputwerte aus einem Verbund entnommen werden müssen

Wie Werte im Berechnungsbaum abgebildet werden

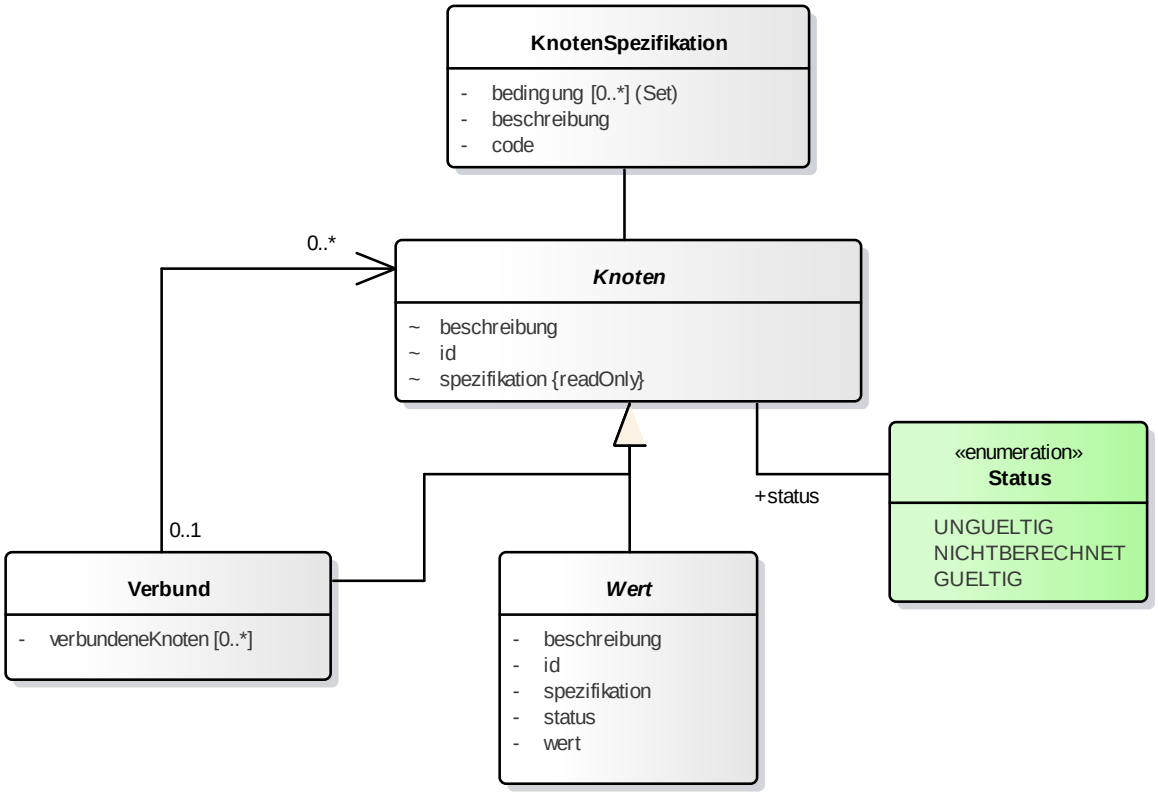


- ID
  Werte werden unterscheidbar
- Beschreibung
  Wird im Report zurückgeliefert
- Status
  Gilt der Wert für Berechnung?
- Übersteuert

Wenige Regeln

- ID
  - Jeder Wert hat eine ID, sie wird entweder im Input mitgeliefert oder durch die Engine generiert
- Status
  - «Gültig»: der Wert ist in der Berechnung verwendet worden
  - «Nicht Berechnet»: Das Ergebnis ist nicht berechnet worden
  - «Nicht Relevant»: Eingabewerte die in der Berechnung nicht verwendet wurden
  - «Ungültig»: Fehler bei der Berechnung
- Übersteuert
  - Für Ergebnisse oder Parameter können im Input Werte mitgeliefert werden, die Engine verwendet den mitgelieferten Wert zur Berechnung, zeigt im Report zusätzlich das berechnete Ergebnis / den ermittelten Parameter

Zusammenfassen von Knoten

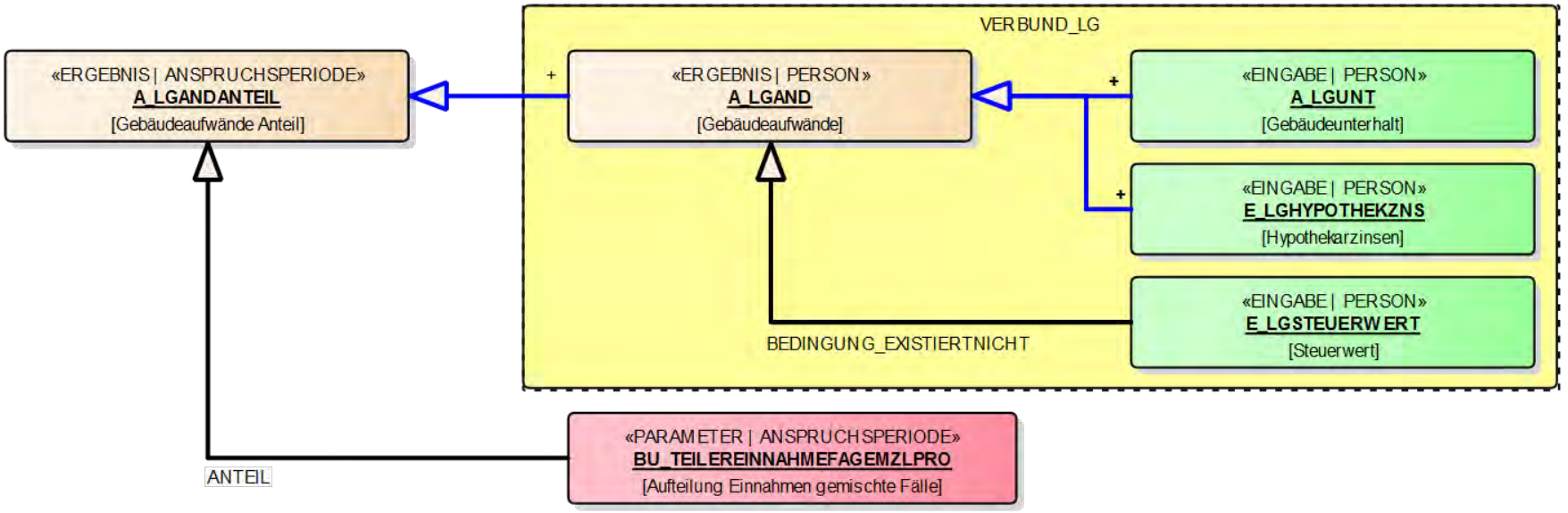


- Verbund ist auch ein Knoten
- fasst logisch Knoten zusammen
- ist berechnungsrelevant
- ID
  Verbunde werden unterscheidbar
- Beschreibung
  Wird im Report zurückgeliefert

Wenige Regeln

- ID
  - Jeder Verbund hat eine ID, sie wird entweder im Input mitgeliefert oder durch die Engine generiert
- Verbundene Knoten
  - Werden über den Code erfasst
  - Ein Knoten darf nur zu einem einzigen Verbund gehören
- Einfluss auf die Anwendung
  - Verbunde werden typischerweise auf Eingabewerte angewendet
    - Beispiel «Hypothek» und «Hypothekarzinssatz»
  - Verbunde können einen Ergebniswert enthalten
    - Beispiel «Hypothekarzins»
  - Engine kann den Hypothekarzins aus den entsprechenden Eingabewerten berechnen
- Einfluss auf den Report
  - Verbundene Knoten können im Report identifiziert werden
  - Zusammengehörende Werte müssen nicht «extern» verwaltet werden

Grafische Darstellung und Umsetzung in Java-Code




```
bvAp.addBerechnugsSpezifikationen(bsf.createBerechnungsSpezifikation("A_LGANDANTEIL", Arrays.asList(
                                  bsf.createInput("A_LGAND")),
                                  "BU_TEILEREINNAHMEFAGEMZLPRO", "ANTEIL"));
bvAp.addKnotenSpezifikation(ksf.createKnotenSpezifikation("A_LGAND", "Gebäudeaufwände", "ERGEBNIS", "PERSON"));

bvAp.addBerechnugsSpezifikationen(bsf.createBerechnungsSpezifikation("A_LGAND", Arrays.asList(
                                  bsf.createInput("A_LGUNT"),
                                  bsf.createInput("E_LGHYPOTHEKZNS")),
                                  "SUMME", "E_LGSTEUERWERT", "BEDINGUNG_EXISTIERTNICHT", "VERBUND_LG"));
bvAp.addKnotenSpezifikation(ksf.createKnotenSpezifikation("A_LGUNT", "Gebäudeunterhalt", "EINGABE", "PERSON"));
bvAp.addKnotenSpezifikation(ksf.createKnotenSpezifikation("E_LGHYPOTHEKZNS", "Hypothekarzinsen", "EINGABE", "PERSON"));
```

Generell

- Ein Attribut beschreibt eine Eigenschaft einer Person oder einer Anspruchsperiode
- Attribute definieren Bedingungen von Knoten und steuern die Parametrierung
- Attribute haben einen AttributTyp
  - Definiert Verwendung für Person oder Anspruchsperiode
  - Definiert ob pro Person / Anspruchsperiode ein oder mehrere Attribute dieses Typs verwendet werden dürfen
- Attribute haben einen Code
  - Codes sind innerhalb der Berechnungsbaums eindeutig
- Attribute habe eine Reihenfolge
  - Attribute mit dem gleichen AttributTyp sind geordnet
- ca. 20 Attribut-Typen zur Fallsteuerung

Beispiel

- AttributTyp «ZLART»
  - Definiert Arten von Zusatzleistungen
  - Gebunden an Anspruchsperiode
  - Kann innerhalb einer Anspruchsperiode mehrfach verwendet werden
- Attribute vom Typ «ZLART»
  - «EL»
    - EL-Karenzfrist erfüllt, kein Verzicht oder Verweigerung
  - «BH»
    - BH-Karenzfrist erfüllt, kein Verzicht oder Verweigerung
  - «GZ»
    - GZ-Karenzfrist erfüllt, kein Verzicht oder Verweigerung

Beispiel

- AttributTyp «ROL»
  - Definiert die Rolle einer Person innerhalb der Berechnung von Zusatzleistungen
  - Gebunden an Person
  - Kann pro Person einmal verwendet werden
- Attribute vom Typ «ROL»
  - «ROLBEG»
    - Begünstigte Person (Fallträger)
  - «ROLPAR»
    - Partner der begünstigten Person
  - «ROLKND»
    - Kind innerhalb der Berechnung
    - Achtung: ist ein Kind der Fallträger hat es die Rolle «ROLBEG»

Fragen?

we never stop thinking! *for you.*

emineo AG
Pfingstweidstrasse 106
CH-8005 Zürich

T +41 44 578 68 00
Info@emineo.ch
www.emineo.ch

Bern | Vevey | Zug | Zürich

Consulting
IT Solutions

emineo