```
1
2   % Programm für die Kostenberechnung auf Komponentenebene
3   function [costCompAnnuities,compInvest, errorInLoopsFlag] = ...
4       BgaPtg2CalcCompCost(handles,paramChain,paramComp,techSimuCompVec,
        techSimuPreCalcChainVec)
5
6
7   % Beinhaltet die Konstanten der aktuellen Prozesskette
8   global const;
9
10  % Damit ohne GUI gearbeitet werden kann.
11  global startGuiFlag;
12
13  % Interne Parameter
14  global inParam;
15
16
17  % Outputs definieren
18  costCompAnnuities   = ones(inParam.nrOfCostAnnuities,1)*-1;
19  compInvest          = -1;
20  errorInLoopsFlag    = false;
21  %return;
22
23  % Konstante
24  LHV_kg_H2 = const.LHV_kg_H2;
25  HHV_kg_H2 = const.HHV_kg_H2;
26
27  % Zuweisung technischer Berechnungen
28  % Leistung der Komponente
29  % Volllaststunden der Komponente in einem Jahr [h]
30  % Strombedarf der Komponente [kWh]
31  % Aktivkohlebedarf der Komponente [kg]
32  % Nährmedienbedarf [m3]
33  % Wärmebedarf; Platzhalter da hauptsächlich über PK bestimmt werden soll
34  % Wärmeüberschuss [kWh th]
35
36  compPower           = techSimuCompVec(1);
37  techCompFLH         = techSimuCompVec(2);
38  techCompElecDemand  = techSimuCompVec(3);
39  techCompActCDemand  = techSimuCompVec(4);
40  compNutrientDemand  = techSimuCompVec(5);
41  techCompHeatDemand  = techSimuCompVec(6);
42  compHeatSurplus     = techSimuCompVec(7);
43  compWaterDemand     = techSimuCompVec(8);
44  techCompOtherDemand = techSimuCompVec(9);
45
46
47  % Volllaststunden auf PK-Ebene [h]
48  % Energiemenge SNG [kWh th]
49  % Wärmebedarf auf PK-Ebene [kWh th]
50  % Wärme Überschuss auf PK-Ebene [kWh th]
51  % Biogas Bedarf auf PK-Ebene [kWh th]
52  % BioMethan Überschuss auf PK-Ebene [kWh th]
53  % Strombedarf für übergeordnete Etechnik [kWh el]
54  % Leistung der Methanisierung wird auf Komp.Ebene zur Berechnung benötigt [kW th]
55  % chainWater4H2
56  % chainElyPers
57
58
59  % techChainVolFlowBG = techSimuPreCalcChainVec(1);
60  % chainVolFlowCO2    = techSimuPreCalcChainVec(2);
61  % chainVolFlowH2     = techSimuPreCalcChainVec(3);
62  % chainVolFlowH2kg   = techSimuPreCalcChainVec(4);
63  techChainMethPower = techSimuPreCalcChainVec(5);
64  % techChainElyPower  = techSimuPreCalcChainVec(6);
65  % techChainpSystem   = techSimuPreCalcChainVec(7);
66
67
68  %% Parameter (Anfang) ************************************************
69
70  % Inflationsrate
71  % Hier werden für jede Komponente einzelne Inflationsraten bestimmmt [%]
72  compInflRate = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
```

```
    'compInflRate')));
73  if isempty(compInflRate)
74      msg = 'Parameter "compInflRate" bzw. dessen Wert ist nicht definiert.';
75      if startGuiFlag == true
76          BgaPtg2UpdateMessageBox(handles,'apend',msg);
77      else
78          disp(msg);
79      end
80      errordlg(msg);
81  end
82
83  % Investitionskosten der Komponente, leistungsabhängig [€/kW]
84  compInvest1 = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compInvest1')));
85  if isempty(compInvest1)
86      msg = 'Parameter "compInvest1" bzw. dessen Wert ist nicht definiert.';
87      if startGuiFlag == true
88          BgaPtg2UpdateMessageBox(handles,'apend',msg);
89      else
90          disp(msg);
91      end
92      errordlg(msg);
93  end
94
95  % Investitionsnebenkosten zb. Montage, Transport, Einbindung, etc...
96  % Fixwert [€]
97  compInvest2 = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compInvest2')));
98  if isempty(compInvest2)
99      msg = 'Parameter "compInvest2" bzw. dessen Wert ist nicht definiert.';
100     if startGuiFlag == true
101         BgaPtg2UpdateMessageBox(handles,'apend',msg);
102     else
103         disp(msg);
104     end
105     errordlg(msg);
106 end
107
108 % Nutzungsdauer der Komponente über die gesamte Lebensdauer [h]
109 compLifetimeHours = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,
    1),'compLifetimeHours')));
110 if isempty(compLifetimeHours)
111     msg = 'Parameter "compLifetimeHours" bzw. dessen Wert ist nicht definiert.';
112     if startGuiFlag == true
113         BgaPtg2UpdateMessageBox(handles,'apend',msg);
114     else
115         disp(msg);
116     end
117     errordlg(msg);
118 end
119
120 % Wartungsfaktor der Komponente
121 % Faktor wird mit Capex multipliziert
122 % z.b. 1% der Capex [%]
123 compMaintFactor = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1
    ),'compMaintFactor')));
124 if isempty(compMaintFactor)
125     msg = 'Parameter "compMaintFactor" bzw. dessen Wert ist nicht definiert.';
126     if startGuiFlag == true
127         BgaPtg2UpdateMessageBox(handles,'apend',msg);
128     else
129         disp(msg);
130     end
131     errordlg(msg);
132 end
133
134 % Instandhaltungsfaktor der Komponente [%]
135 compUpkeepFactor = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1
    ),'compUpkeepFactor')));
136 if isempty(compUpkeepFactor)
137     msg = 'Parameter "compUpkeepFactor" bzw. dessen Wert ist nicht definiert.';
138     if startGuiFlag == true
139         BgaPtg2UpdateMessageBox(handles,'apend',msg);
```

```
140         else
141             disp(msg);
142         end
143         errordlg(msg);
144     end
145 
146     % Sonstige Kostenfaktoren [€]
147     compOtherCost = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compOtherCost')));
148     if isempty(compOtherCost)
149         msg = 'Parameter "compOtherCost" bzw. dessen Wert ist nicht definiert.';
150         if startGuiFlag == true
151             BgaPtg2UpdateMessageBox(handles,'apend',msg);
152         else
153             disp(msg);
154         end
155         errordlg(msg);
156     end
157 
158     % Strompreis für Komponente
159     % compElecPrice = CompElecPrice [ct/kWh]
160     compElecPrice = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compElecPrice')));
161     if isempty(compElecPrice)
162         msg = 'Parameter "compElecPrice" bzw. dessen Wert ist nicht definiert.';
163         if startGuiFlag == true
164             BgaPtg2UpdateMessageBox(handles,'apend',msg);
165         else
166             disp(msg);
167         end
168         errordlg(msg);
169     end
170 
171     % Wasserstoffpreis [€/kg H2]
172     compH2Price = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compH2Price')));
173     if isempty(compH2Price)
174         msg = 'Parameter "compH2Price" bzw. dessen Wert ist nicht definiert.';
175         if startGuiFlag == true
176             BgaPtg2UpdateMessageBox(handles,'apend',msg);
177         else
178             disp(msg);
179         end
180         errordlg(msg);
181     end
182 
183 
184     % compOtherPrice [???]
185     compOtherPrice = paramComp.valueCompParam(find(strcmp(paramComp.txtCompParamInfo(:,1),
    'compOtherPrice')));
186     if isempty(compOtherPrice)
187         msg = 'Parameter "compOtherPrice" bzw. dessen Wert ist nicht definiert.';
188         if startGuiFlag == true
189             BgaPtg2UpdateMessageBox(handles,'apend',msg);
190         else
191             disp(msg);
192         end
193         errordlg(msg);
194     end
195 
196     % Parameter auf chainebene
197     % Projektlaufzeit [a]
198     chainPeriod = paramChain.valueChainParam(find(strcmp(paramChain.txtChainParamInfo(:,1
    ),'chainPeriod')));
199     if isempty(chainPeriod)
200         msg = 'Parameter "chainPeriod" bzw. dessen Wert ist nicht definiert.';
201         if startGuiFlag == true
202             BgaPtg2UpdateMessageBox(handles,'apend',msg);
203         else
204             disp(msg);
205         end
206         errordlg(msg);
207     end
```

```
208
209 % Kalkulationszinsatz, discount rate [%]
210 chainDiscRate = paramChain.valueChainParam(find(strcmp(paramChain.txtChainParamInfo(:,
    1),'chainDiscRate')));
211 if isempty(chainDiscRate)
212     msg = 'Parameter "chainDiscRate" bzw. dessen Wert ist nicht definiert.';
213     if startGuiFlag == true
214         BgaPtg2UpdateMessageBox(handles,'apend',msg);
215     else
216         disp(msg);
217     end
218     errordlg(msg);
219 end
220
221 % Aktivkohlepreis [€/kg] Aktivkohle
222 chainActCPrice = paramChain.valueChainParam(find(strcmp(paramChain.txtChainParamInfo
    (:,1),'chainActCPrice')));
223 if isempty(chainActCPrice)
224     msg = 'Parameter "chainActCPrice" bzw. dessen Wert ist nicht definiert.';
225     if startGuiFlag == true
226         BgaPtg2UpdateMessageBox(handles,'apend',msg);
227     else
228         disp(msg);
229     end
230     errordlg(msg);
231 end
232
233 % Wärme Verkaufspreis in €/kWh [ct/kWh]
234 chainHeatSellingPrice = paramChain.valueChainParam(find(strcmp(paramChain.
    txtChainParamInfo(:,1),'chainHeatSellingPrice')));
235 if isempty(chainHeatSellingPrice)
236     msg = 'Parameter "chainHeatSellingPrice" bzw. dessen Wert ist nicht definiert.';

237     if startGuiFlag == true
238         BgaPtg2UpdateMessageBox(handles,'apend',msg);
239     else
240         disp(msg);
241     end
242     errordlg(msg);
243 end
244
245 % Wärme Einkaufsspreis in €/kWh [ct/kWh]
246 chainHeatPurchasePrice = paramChain.valueChainParam(find(strcmp(paramChain.
    txtChainParamInfo(:,1),'chainHeatPurchasePrice')));
247 if isempty(chainHeatPurchasePrice)
248     msg = 'Parameter "chainHeatPurchasePrice" bzw. dessen Wert ist nicht definiert.';

249     if startGuiFlag == true
250         BgaPtg2UpdateMessageBox(handles,'apend',msg);
251     else
252         disp(msg);
253     end
254     errordlg(msg);
255 end
256
257 % Wasserpreis
258 chainWaterPrice = paramChain.valueChainParam(find(strcmp(paramChain.txtChainParamInfo
    (:,1),'chainWaterPrice')));
259 if isempty(chainWaterPrice)
260     msg = 'Parameter "chainWaterPrice" bzw. dessen Wert ist nicht definiert.';
261     if startGuiFlag == true
262         BgaPtg2UpdateMessageBox(handles,'apend',msg);
263     else
264         disp(msg);
265     end
266     errordlg(msg);
267 end
268
269 % chainNutrientPrice
270 chainNutrientPrice = paramChain.valueChainParam(find(strcmp(paramChain.
    txtChainParamInfo(:,1),'chainNutrientPrice')));
271 if isempty(chainNutrientPrice)
272     msg = 'Parameter "chainNutrientPrice" bzw. dessen Wert ist nicht definiert.';
```

```
273         if startGuiFlag == true
274             BgaPtg2UpdateMessageBox(handles,'apend',msg);
275         else
276             disp(msg);
277         end
278         errordlg(msg);
279     end
280
281     %% Parameter (Ende) ****************************************************
282
283
284     compInflRate                = compInflRate/100;
285     compMaintFactor             = compMaintFactor/100;
286     compUpkeepFactor            = compUpkeepFactor/100;
287     compElecPrice               = compElecPrice/100;
288     % compp                       = compp*(10^5);
289     % compDeltap                  = compDeltap * (10^5);
290     % compT                       = compT + 273.15;
291     % compConcBioM                = compConcBioM/100;
292     % compConcSNG                 = compConcSNG/100;
293     % compConcCO2                 = compConcCO2/100;
294     % compConcH2                  = compConcH2/100;
295     % compConcH2S                 = compConcH2S/(10^6);
296     % compConcH2O                 = compConcH2O/100;
297     % compConcTraceGas            = compConcTraceGas/(10^6);
298     % compConcOther               = compConcOther/100;
299     % compDeltaVolFlowBioM        = compDeltaVolFlowBioM/100;
300     % compDeltaVolFlowSNG         = compDeltaVolFlowSNG/100;
301     % compDeltaVolFlowCO2         = compDeltaVolFlowCO2/100;
302     % compDeltaVolFlowH2          = compDeltaVolFlowH2/100;
303     % compDeltaVolFlowH2S         = compDeltaVolFlowH2S/100;
304     % compDeltaVolFlowH2O         = compDeltaVolFlowH2O/100;
305     % compDeltaVolFlowTraceGas    = compDeltaVolFlowTraceGas/100;
306     % compDeltaVolFlowOther       = compDeltaVolFlowOther/100;
307     % compLeakage                 = compLeakage/100;
308     % compOtherFactor             = compOtherFactor/100;
309     % compNutrientVolFlowFactor   = compNutrientVolFlowFactor/100;
310     % compActCH2S                 = compActCH2S/100;
311     % compConvCO2Ideal            = compConvCO2Ideal/100;
312     % compConvCO2Real             = compConvCO2Real/100;
313
314     %chain Parameter
315     chainDiscRate               = chainDiscRate/100;
316     % chainElecPrice              = chainElecPrice/100;
317     % chainOperationFactor        = chainOperationFactor/100;
318     % chainInsurFactor            = chainInsurFactor/100;
319     % chainRenatFactor            = chainRenatFactor/100;
320     % chainTechInflRate           = chainTechInflRate/100;
321     % chainBuildInflRate          = chainBuildInflRate/100;
322     % chainInflRate               = chainInflRate/100;
323     chainWaterPrice             = chainWaterPrice;
324     % chainBGPurchasePrice        = chainBGPurchasePrice/100;
325     % chainBMSellingPrice         = chainBMSellingPrice/100;
326     chainHeatPurchasePrice      = chainHeatPurchasePrice/100;
327     chainHeatSellingPrice       = chainHeatSellingPrice/100;
328     % chainO2SellingPrice         = chainO2SellingPrice/100;
329     % chainComprEff               = chainComprEff/100;
330     % chainEffMeth                = chainEffMeth/100;
331     % chainEffCHP                 = chainEffCHP/100;
332     % chainEffEly                 = chainEffEly/100;
333
334     %% Berechnungen
335
336     if (1+chainDiscRate) == 0 || (chainDiscRate-compInflRate) == 0
337         msg = 'Variable "compNPVFactor" kann nicht berechnet werden. Division durch null.'
            ;
338         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
339         if startGuiFlag == true
340             BgaPtg2UpdateMessageBox(handles,'apend',msg);
341         else
342             disp(msg);
343         end
```

```
344         errorInLoopsFlag = true;     return;
345
346     else
347         % Barwertfaktor, english net present value factor (NPV)
348         compNPVFactor = (1-((1+compInflRate) / (1+chainDiscRate))^chainPeriod) ...
349             / (chainDiscRate-compInflRate);
350     end;
351
352
353     % Annuitätsfaktor wird für Berechnung in den Komponenten benötigt, ist für alle
        Komponenten gleich
354     if (((1+chainDiscRate)^chainPeriod)-1) == 0
355         msg = 'Variable "compAnnuityFactor" kann nicht berechnet werden. Division durch
            null.';
356         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
357         if startGuiFlag == true
358             BgaPtg2UpdateMessageBox(handles,'apend',msg);
359         else
360             disp(msg);
361         end
362         errorInLoopsFlag = true;     return;
363
364     else
365         compAnnuityFactor = (((1+chainDiscRate)^chainPeriod)*chainDiscRate) ...
366             / (((1+chainDiscRate)^chainPeriod)-1);
367
368     end;
369
370     % Nutzungsdauer in Jahren
371     if techCompFLH == 0
372         msg = 'Variable "compLifetimeYears" kann nicht berechnet werden. Division durch
            null.';
373         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
374         if startGuiFlag == true
375             BgaPtg2UpdateMessageBox(handles,'apend',msg);
376         else
377             disp(msg);
378         end
379         errorInLoopsFlag = true;     return;
380
381     else
382         compLifetimeYears = compLifetimeHours/techCompFLH;
383     end;
384
385     % Ersatzhäufigkeit
386     %compReplacement = math.ceil(chainPeriod/(compLifetimeYears)-1);
387     if compLifetimeYears == 0
388         msg = 'Variable "compReplacement" kann nicht berechnet werden. Division durch
            null.';
389         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
390         if startGuiFlag == true
391             BgaPtg2UpdateMessageBox(handles,'apend',msg);
392         else
393             disp(msg);
394         end
395         errorInLoopsFlag = true;     return;
396
397     else
398         compReplacement =ceil(chainPeriod/(compLifetimeYears)-1);
399     end;
400
401
402     % Investitionskosten in Abhängigkeit der Leistung
403     compCapex1 = compPower * compInvest1;
404
405     % Investitionskosten als Fixwert, z.B. für Montage, Transport etc,
406     % Übernahme von Parameter
407     compCapex2 = compInvest2;
408
409     % Anfangsinvestition der Komponente
```

```
410   compInvest = compCapex1 + compCapex2;
411
412   % Ohne Ersatzbeschaffung ist der Barwert gleich Null
413   if compReplacement == 0
414       compReplNPV1 = 0;
415       compReplNPV2 = 0;
416
417   % Bei 1 Ersatzbeschaffung muss für eine fiktive 2. der Wert weiter Null sein
418   elseif compReplacement == 1
419
420       if ((1+chainDiscRate)^(1*compLifetimeYears)) == 0
421           msg = 'Variable "compReplNPV1" kann nicht berechnet werden. Division durch
              null.';
422           msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
              Prozesskette werden unterbrochen.'];
423           if startGuiFlag == true
424               BgaPtg2UpdateMessageBox(handles,'apend',msg);
425           else
426               disp(msg);
427           end
428           errorInLoopsFlag = true;     return;
429
430       else
431           compReplNPV1 = compInvest * ((1+compInflRate)^(1*compLifetimeYears)) ...
432           /((1+chainDiscRate)^(1*compLifetimeYears));
433       end;
434
435       compReplNPV2 = 0;
436
437   % Bei 2 Ersatzbeschaffungen wurde die 1. Beschaffung zwar bereits ersetzt, hat aber
      weiterhin einen Barwert
438   % bei noch mehr Ersatzbeschaffungen müsste der Code erweitert werden
439   elseif compReplacement == 2
440
441       if ((1+chainDiscRate)^(1*compLifetimeYears)) == 0
442           msg = 'Variable "compReplNPV1" kann nicht berechnet werden. Division durch
              null.';
443           msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
              Prozesskette werden unterbrochen.'];
444           if startGuiFlag == true
445               BgaPtg2UpdateMessageBox(handles,'apend',msg);
446           else
447               disp(msg);
448           end
449           errorInLoopsFlag = true;     return;
450
451       else
452           compReplNPV1 = compInvest * ((1+compInflRate)^(1*compLifetimeYears)) ...
453           /((1+chainDiscRate)^(1*compLifetimeYears));
454       end;
455
456       if ((1+chainDiscRate)^(2*compLifetimeYears)) == 0
457           msg = 'Variable "compReplNPV2" kann nicht berechnet werden. Division durch
              null.';
458           msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
              Prozesskette werden unterbrochen.'];
459           if startGuiFlag == true
460               BgaPtg2UpdateMessageBox(handles,'apend',msg);
461           else
462               disp(msg);
463           end
464           errorInLoopsFlag = true;     return;
465
466       else
467           compReplNPV2 = compInvest * ((1+compInflRate)^(2*compLifetimeYears)) ...
468           /((1+chainDiscRate)^(2*compLifetimeYears));
469       end;
470
471   else
472       msg = 'Fehler! Es wird von max. 2 Ersatzbeschaffungen ausgegangen. Überprüfe
          Lebensdauer und FLH der Komponenten.';
473       msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
          Prozesskette werden unterbrochen.'];
```

```
474         if startGuiFlag == true
475             BgaPtg2UpdateMessageBox(handles,'apend',msg);
476         else
477             disp(msg);
478         end
479         errorInLoopsFlag = true;     return;
480     end;
481 
482     % Restwert, residual or salvage value
483     if compLifetimeYears == 0 || (1+chainDiscRate) == 0
484         msg = 'Variable "compResidual" kann nicht berechnet werden. Division durch null.';
485         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
486         if startGuiFlag == true
487             BgaPtg2UpdateMessageBox(handles,'apend',msg);
488         else
489             disp(msg);
490         end
491         errorInLoopsFlag = true;     return;
492 
493     else
494         compResidual = compInvest * ((1+compInflRate)^(compReplacement*compLifetimeYears))
             ...
495         * (((compReplacement+1)*compLifetimeYears - chainPeriod)/compLifetimeYears) ...
496             *(1/(1+chainDiscRate)^chainPeriod);
497     end;
498 
499 
500     % Annuität der kapitalgebundenen Kosten
501     % Gesamt-CAPEX + Barwerte der Ersatzbeschaffungen abzüglich Restwert, * AnFaktor
502     compAnnuityCapital = (compInvest + compReplNPV1 + compReplNPV2 - compResidual) ...
503         * compAnnuityFactor;
504 
505     % Stromkosten der Komponente
506     compCostElec = techCompElecDemand * compElecPrice;
507 
508     % Annuität Strom
509     compAnnuityElec = compCostElec * compAnnuityFactor * compNPVFactor;
510 
511     % Annuität der Wärme
512     compAnnuityHeat = (techCompHeatDemand * chainHeatPurchasePrice ...
513                       - compHeatSurplus * chainHeatSellingPrice) ...
514                       * compAnnuityFactor * compNPVFactor;
515 
516     % Wasserkosten
517     % Es geht weniger darum, ob eine H2-Komponente enthalten ist, sondern mehr darum,
518     % dass die Abfrage nur bei der H2 Komponente erfolgt und ob darin ein H2Preis>0
        gegeben ist
519     % if isempty(strfind(paramComp.txtCompName,'Wasserstoffinput')) == 0 && compH2Price
        == 0
520     %     compCostWater = techChainMethPower * techChainFLH/LHV_kg_H2 * chainWater4H2 *
        chainWaterPrice;
521     % else
522     %     compCostWater = 0;
523     % end;
524 
525     compCostWater = compWaterDemand * chainWaterPrice;
526 
527     % Aktivkohle wird nur für Entschwefelung benötigt
528     compCostActCar = techCompActCDemand * chainActCPrice;
529 
530     % Annuität bei LCOH- Preis für Wasserstoff
531     if HHV_kg_H2 == 0
532         msg = 'Variable "compAnnuityH2" kann nicht berechnet werden. Division durch null.'
            ;
533         msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
            Prozesskette werden unterbrochen.'];
534         if startGuiFlag == true
535             BgaPtg2UpdateMessageBox(handles,'apend',msg);
536         else
537             disp(msg);
538         end
539         errorInLoopsFlag = true;     return;
```

```
540
541   else
542       compAnnuityH2 = compH2Price/HHV_kg_H2 * techCompFLH * techChainMethPower * 1000;
543   end;
544
545
546   % costCompNutrient
547   costCompNutrient = compNutrientDemand * chainNutrientPrice;
548
549   % Annuität der bedarfsgebundenen Kosten
550   compAnnuityDemand = compAnnuityH2 + (compCostWater + compCostActCar + costCompNutrient
      ) ...
551       * compAnnuityFactor * compNPVFactor;
552
553   % Wartungskosten
554   compCostMaint = compMaintFactor * compInvest;
555
556   % Instandhaltungskosten
557   compCostUpkeep = compUpkeepFactor * compInvest;
558
559   % Annuität der betriebsgebundenen Kosten
560   compAnnuityOperation = (compCostMaint + compCostUpkeep) ...
561       * compAnnuityFactor * compNPVFactor;
562
563   % Kosten für sonstige Verbräuche, die sich aus TSimu ergeben, Platzhalter
564   compCostOther = techCompOtherDemand * compOtherPrice;
565
566   % Annuität sonstiger Kosten (Sonstiges)
567   compAnnuityOther = compOtherCost * compAnnuityFactor * compNPVFactor;
568
569
570   % Zusammenfassung der Annuitäten
571   costCompAnnuities(1) = compAnnuityCapital;
572   costCompAnnuities(2) = compAnnuityElec;
573   costCompAnnuities(3) = compAnnuityHeat;
574   costCompAnnuities(4) = compAnnuityDemand;
575   costCompAnnuities(5) = compAnnuityOperation;
576   costCompAnnuities(6) = compAnnuityOther;
577
578
579
```