

Studienarbeit

Benutzerstudie zur Usability einer Break The Glass Annotationssprache für Business-Processes

Tutorial zum Erlernen der
BTG-Annotationssprache

Mit Übungsaufgaben

Herzlich Willkommen

- Danke für Ihr Interesse an der heutigen Nutzerstudie
- Das heutige Experiment dient der Überprüfung der Usability einer Workflow-Annotationssprache
- Diese Sprache wurde am IPD entwickelt und soll nun hinsichtlich ihrer Praxistauglichkeit optimiert werden

Zum Ablauf der Studie

- Zuerst: Einführung/Wiederholung von Grundlagen
Motivation der Thematik
- Dann: Aktives Erlernen der Annotationssprache
anhand eines Beispiels
Kleine Aufgabe am Ende jedes Kapitels
- Danach: Einsatz des Gelernten an drei praktischen
Beispielen
- Zuletzt: Feedback mittels eines Fragebogens

Hinweise zu den Aufgaben

- Es gibt kein „Richtig“ oder „Falsch“
 - Wir wollen die Sprache überprüfen
 - Jede Antwort hilft uns weiter!
- Wenn Sie eine Aufgabe gar nicht lösen können ist das ebenfalls ein Ergebnis, das uns hilft!
- Die von uns gezeigten Musterlösungen können von Ihren Lösungen abweichen. Das heißt allerdings nicht, dass Ihre Lösungen falsch sind!
- Fragen dürfen Sie gerne stellen, ich werde diese allerdings nicht an jeder Stelle der Studie beantworten

Hinweise zu den Aufgaben (2)

- Bitte konzentrieren Sie sich auf das Wesentliche!
 - Die zusätzlichen Materialien sind so einfach wie möglich gehalten
 - Die gezeigten Diagramme sind vereinfachte Ausschnitte
 - Bitte nicht auf dadurch entstandene Formalismus-Fehler versteifen 😊

- Bitte seien Sie ehrlich!
 - Beschönigungen helfen uns nicht und sind nicht nötig, da niemandes Karriere von diesem Test abhängt
 - Wenn Sie krampfhaft Fehler suchen kann es passieren, dass wir an den falschen Stellen optimieren
 - Sagen Sie also einfach, was Sie denken

Hinweise zu den heutigen Tests (3)

- Und nochmal:
Es gibt kein „Richtig“ oder „Falsch“



AGENDA

- 1. Einführung und Motivation**
2. BTG-Konzept
3. Rollenverteilung
4. BTG-Annotationen
5. Obligations
6. BPCC
7. Weitere Aufgaben
8. Fragebogen

1. Einführung und Motivation

Grundidee von Workflows

- Kontrollfluss eines Programmes von dessen Ausführung trennen
- Dazu: Abbildung dieses Kontrollflusses in Diagrammen
- Allgemeiner: Abbildung von Prozessen in Diagrammen
- Diagramme werden kompiliert und ausgeführt

Business Process Model and Notation (BPMN)

- Darstellung von Geschäftsprozessen in Diagramme
- Grundlage für heutige Thematik
- Für die Studie beschränken wir uns hierbei auf das Wesentliche

1. Einführung und Motivation (2)

Personen und Rollen

- Personen sind Individuen, die Aufgaben in einem Prozess ausführen, z.B. „Prof. K. Böhm“
- Personen können Rollen innehaben; so hat das Individuum „K. Böhm“ die Rolle „Professor“ inne

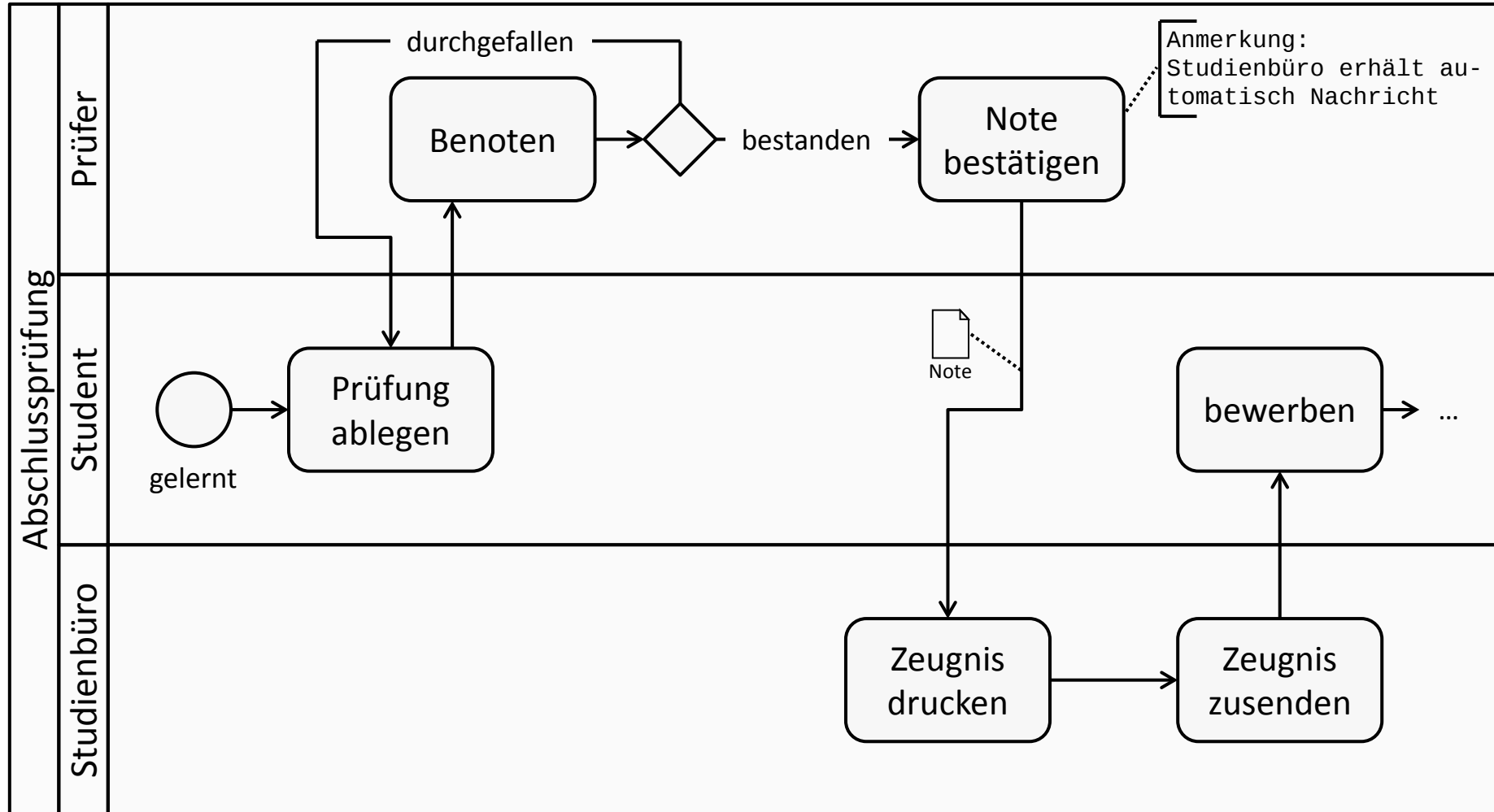
Attribute

- Personen haben Attribute (z.B. Alter)
- Wir nutzen eine vereinfachte Sicht auf die Daten
- Syntax: Person.attribut
 - z.B. Person.age für das Alter
- Für die Aufgaben nötige Attribute werden vorgegeben
- Im Zweifelsfall einfach etwas Passendes ausdenken

1. Einführung und Motivation

Beispiel eines Business Processes

Abschlussprüfung an der Uni (mit Zeugnisvergabe)



1. Einführung und Motivation (3)

Datenobjekte

- Normalerweise haben Personen, die Datenobjekte benötigen, entsprechende Zugriffsrechte
 - z.B. Hausarzt darf auf Patientenakte zugreifen
 - z.B. Betreuer bei Versicherung darf Kundendaten einsehen
- Datenzugriffe sind im Prozess-Diagramm ausmodelliert und an Aufgaben gekoppelt
- Es gibt allerdings auch Ausnahmesituationen, in denen Personen, die normalerweise keine Rechte dazu haben, auf die Datenobjekte zugreifen müssen
 - z.B. Arzt in der Notaufnahme braucht Patientenakte
 - z.B. Urlaubsvertretung darf Kundendaten einsehen

1. Einführung und Motivation (4)

Ausnahme-Datenzugriffe

- Außerplanmäßige Datenzugriffe sind i.d.R. zum Modellierungszeitpunkt bekannt und müssen vom Prozessmodellierer umgesetzt werden
 - Die direkte Modellierung dieser Ausnahmen in BPMN-Diagrammen wäre zwar möglich, allerdings auch
 - aufwändig
 - fehleranfällig
 - schnell ausufernd
- ⇒ Somit Suche nach einem eleganteren Weg zur Ausnahmebehandlung
- ⇒ BTG: Break the Glass als Lösung

1. Einführung und Motivation (5)

Break the Glass (veranschaulicht)

- Datenobjekte befinden sich hinter Glasscheibe
 - Nutzer weiß, dass sie da sind
 - Nutzer darf allerdings nicht darauf zugreifen
 - Glasscheibe zerbrechen, um an Daten zu gelangen
 1. Wunsch, an Datenobjekt zu gelangen
 2. Genehmigung nötig
 3. Randbedingungen (Hammer dabei? Alarmanlage aus?)
 4. Die Konsequenzen auf sich nehmen
- ⇒ Diese Schritte zum Zerbrechen der Glasscheibe müssen modelliert werden

1. Einführung und Motivation (6)

Modellierung

- Modellierung mit Hilfe einer dafür entwickelten Annotationssprache zur Umsetzung von BTG
- Kommentar-Funktion von BPMN wird genutzt, um Annotationen zu „coden“
- Für jeden außerplanmäßigen Datenzugriff ist also eine entsprechende BTG-Annotation zu modellieren
 - Dazu wird ein Kommentar an die Aufgabe, nach der das Brechen des Glases eingeleitet wird, angefügt
 - **<<BTG: >>** kennzeichnet eine solche Annotation

⇒ Prozessmodellierer „coden“ also die Ausnahmen

1. Einführung und Motivation (7)

Umsetzung

- Ein BPMN-Tool transformiert die BTG-Annotationen in sogenannte Prozess-Fragmente
 - Prozess-Fragmente sind die „ausmodellerte“ BPMN-Version der BTG-Annotationen
 - Die Fragmente werden vor der Kompilierung an die entsprechenden Stellen des BPMN-Diagramms eingefügt
-
- ⇒ Spätere Nutzung der Sprache im Rahmen eines BPMN-Tools
 - ⇒ Die Umwandlung der Annotationen erfolgt für den Endanwender unsichtbar

Beispiel: Abschlussprüfung



- Nach der Anmeldung zur Abschlussprüfung erbringt ein Student seine Prüfungsleistung. Diese wird vom Prüfer benotet. Ist der Student durchgefallen wird die Prüfung wiederholt, hat er bestanden trägt der Prüfer die Note ein. Das Studienbüro wird automatisch benachrichtigt, druckt ein Zeugnis und schickt dieses an den Studenten.
- *Wir betrachten nun folgende Ausnahmesituation:*
- Ein Student möchte sich möglichst sofort nach Erbringen seiner Prüfleistung bewerben. Dafür braucht er eine Vorab-Version seiner Prüfungsnote. Diese kann vom Prüfer ausgestellt und dem Studenten gegeben werden. Nach Aushändigung der Vorab-Note muss der Prüfer eine E-Mail ans Studienbüro senden.

Aufgabe 1

BP um BTG-Annotation ergänzen

- *Angenommen, der Student möchte sich unmittelbar nach Abschluss der Prüfungen bewerben*
- *Er braucht seine Noten für diese Bewerbung*
- *Er möchte also eine inoffizielle Vorab-Note seiner Prüfung für seine Bewerbung haben*

❖ **Wo wäre hier das Glas zu brechen?**

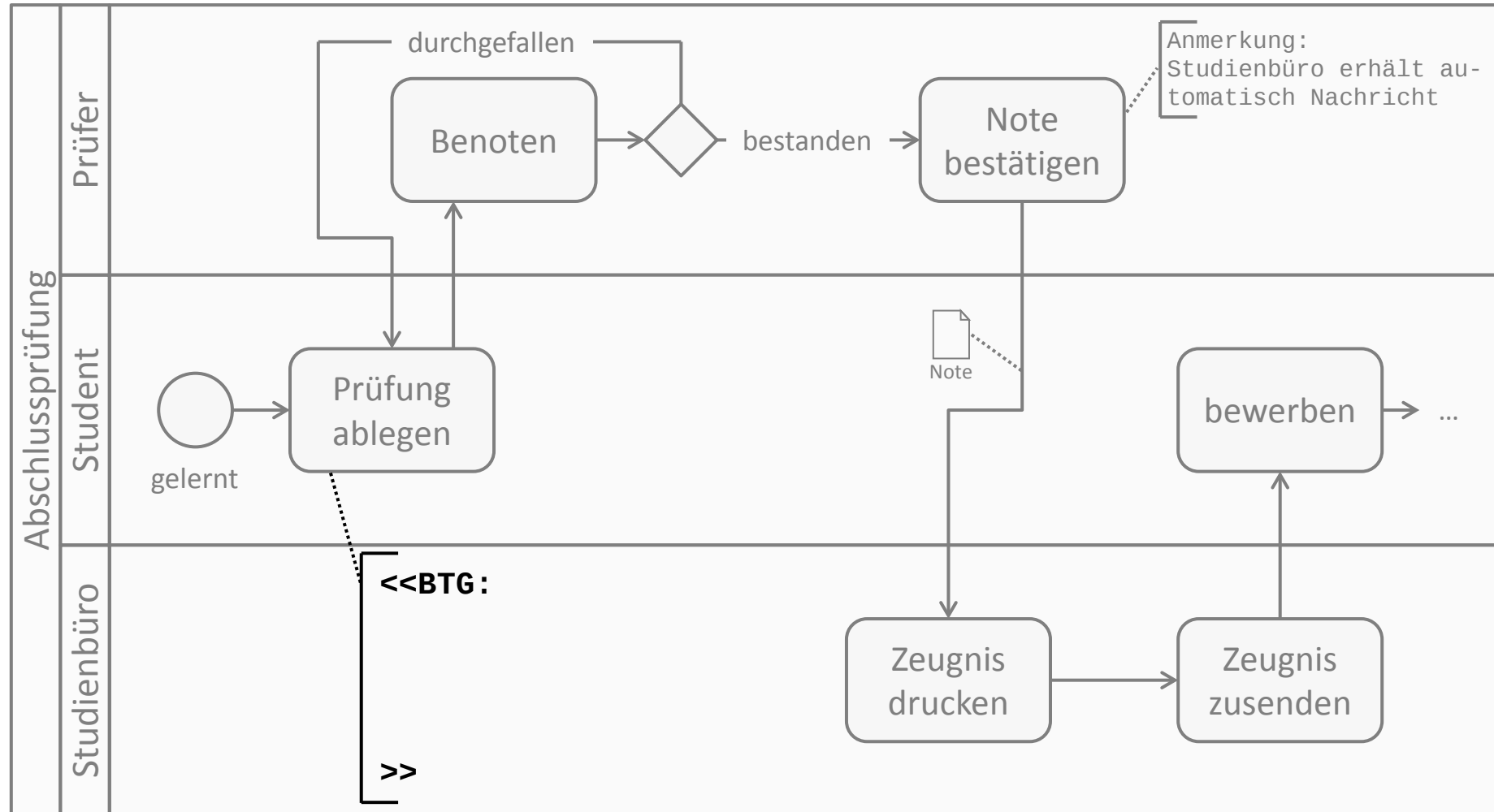
- ✓ Fügen Sie eine Annotation an der entsprechenden Stelle des vorliegenden Geschäftsprozesses ein
- ✓ Machen Sie deutlich, dass es sich dabei um eine BTG-Annotation handelt

1. Einführung und Motivation (2)

Beispiel eines Business Process



Abschlussprüfung an der Uni (mit Zeugnisvergabe)



AGENDA

1. Einführung und Motivation
- 2. BTG-Konzept**
3. Rollenverteilung
4. BTG-Annotationen
5. Obligations
6. BPCC
7. Weitere Aufgaben
8. Fragebogen

2. Gesamtmodell

- Die BTG-Annotationssprache besteht im Wesentlichen aus drei Elementen
 - Break The Glass Annotationen
 - Einbeziehung des Kontext, z.B. Bedingungen an Ausführung
 - Zugehörige Folgeaufgaben, d.h. Konsequenzen
- Genauere Vorstellung dieser Elemente im Folgenden

2. Gesamtmodell (2)

Break The Glass (BTG)

- In BTG-Annotationen wird definiert
 - Wer greift auf die Daten zu?
 - Wer gibt den Zugriff auf die Daten frei?
 - Authentifizierung dieser Individuen

 - Die Datenobjekte, auf die zugegriffen wird
 - Rechte zur Nutzung der Daten (R/W)

 - Randbedingungen
 - Auflistung zugehöriger Folgeaufgaben: Obligations

2. Gesamtmodell (3)

Obligations (OG)

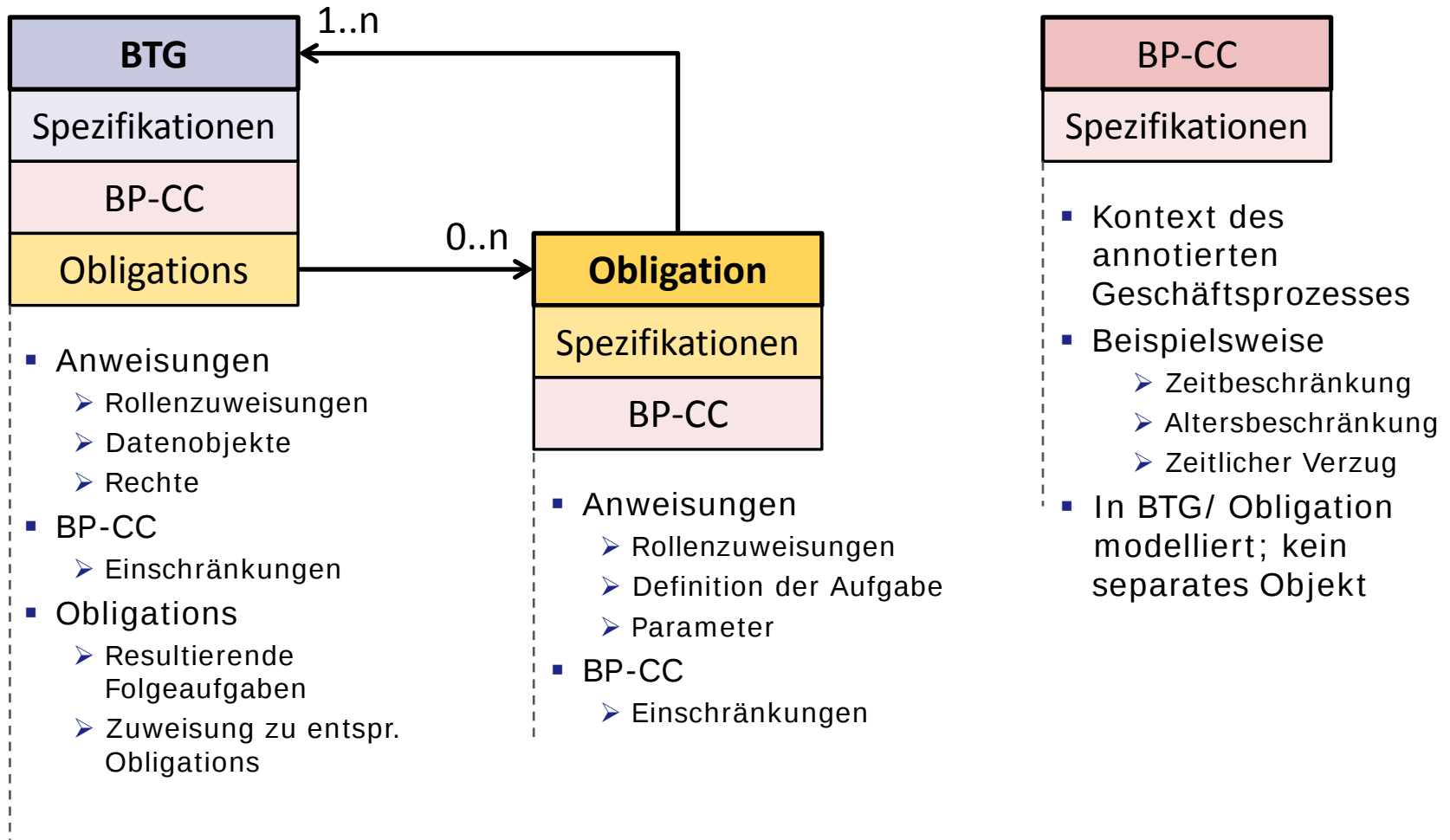
- Folgeaufgaben, die aus BTG resultieren (Verpflichtungen)
- Anschaulich: Beseitigung der Schäden, die durch Brechen des Glases entstanden sind
 - z.B. Scherben wegfeegen
 - z.B. neue Glasscheibe kaufen
 - z.B. Hausmeister informieren
- Legen (in vorgegebenen Mustern) fest, was als Folge des Glas-Brechens zu tun ist (und wer es tun muss)
- Separate Modellierung (in eigenen Annotationen)
- Referenzierung von BTG aus

2. Gesamtmodell (4)

Business Process Context Constraints (BPCC)

- Bezug zum zugrundeliegenden Geschäftsprozesses
- Bedingungen an die Ausführung von BTG
 - Temporal
 - Konditional
- Klären, ob BTG überhaupt möglich ist
 - z.B. Altersbeschränkung
 - z.B. Zeitverzug bis zum Start
- Definition von Ausdrücken mit Bezug zum Geschäftsprozess
- BPCC Finden sich sowohl innerhalb von BTG-Anweisungen als auch innerhalb von Obligations wieder
- Keine separate Modellierung in eigenen Annotationen

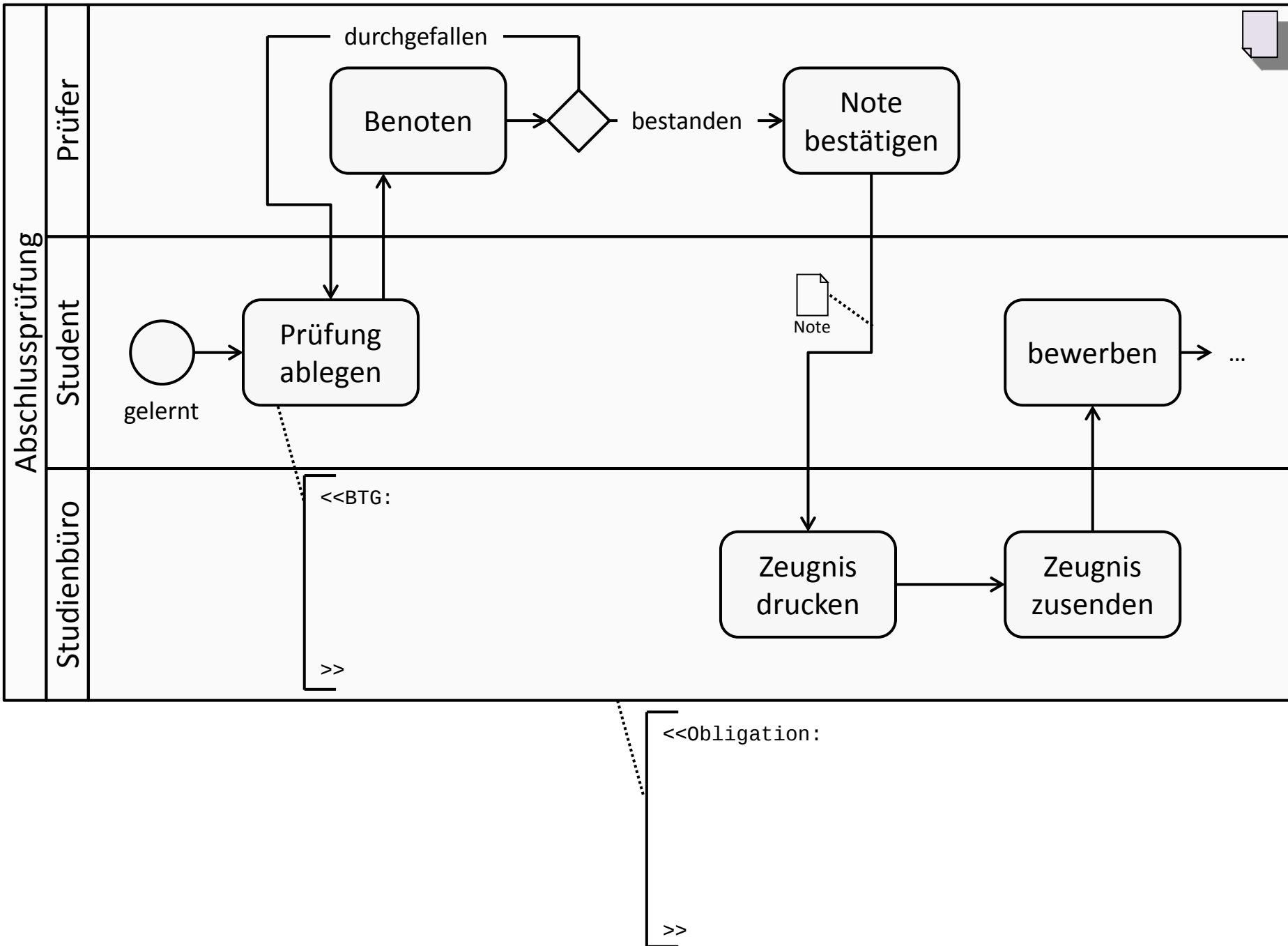
2. Gesamtmodell (5) Übersicht



Aufgabe 2

Obligation und BPCC beschreiben

- ❖ **Gegeben das vorliegende Szenario und die entsprechende BTG-Annotation, beschreiben Sie verbal die nötige Obligation.**
- ❖ **Geben Sie außerdem verbal sinnvolle Rahmenbedingungen (BPCC) an**
 - ✓ Für die BTG-Annotation
 - ✓ Für die Obligation
- ❖ *Schreiben Sie Ihre Anmerkungen einfach in die entsprechenden Annotationen*





AGENDA

1. Einführung und Motivation
2. BTG-Konzept
- 3. Rollenverteilung**
4. BTG-Annotationen
5. Obligations
6. BPCC
7. Weitere Aufgaben
8. Fragebogen

3. Rollen Konzept

- Bei BTG gibt es drei verschiedene Rollen
- Beschreiben die Personen, die an BTG beteiligt sein können
- Es müssen nicht immer alle Rollen definiert werden
 - Teils werden Aufgaben automatisch ausgeführt
- Eine Person kann mehrere Rollen belegen
- Die BTG-Rollen werden in BTG und OG vergeben

3. Rollen

Definition

■ **Accessor**

- Person, die auf Daten zugreifen will
- fordert BTG an, damit also zugleich Initiator
- In BTG-Annotation spezifiziert

■ **Activator**

- Person, die Ausnahme-Zugriff genehmigt oder verweigert
- i.d.R. nicht Person, die Ausnahme-Zugriff anfordert
- In BTG-Annotation spezifiziert

■ **Compensator**

- Zuständig für Folgen der Ausnahme (Reparatur)
- In Obligation definiert

3. Rollen

Notwendigkeit / Default

- Accessor
 - Default: Ausführer der BTG-annotierten Aktivität

- Activator
 - Default: Activator == Accessor

- Compensator
 - Default: Automatisierte, nicht-menschliche Ausführung
 - Gehören zu einer BTG-Annotation mehrere Obligations, so kann es für jede Obligation einen eigenen Compensator geben

Aufgabe 3

Rollen vergeben



- ❖ Bitte benennen Sie im vorliegenden Beispiel der Abschlussprüfung die drei involvierten Rollen

✓ Accessor:

✓ Activator:

✓ Compensator:

Aufgabe 3

Rollen vergeben



- ❖ Bitte benennen Sie im vorliegenden Beispiel der Abschlussprüfung die drei involvierten Rollen

✓ <u>Accessor:</u>	Student <i>ODER</i> <i>weglassen (default)</i>
--------------------	---

✓ <u>Activator:</u>	Prüfer
---------------------	---------------

✓ <u>Compensator:</u>	Prüfer <i>(versendet E-Mail)</i>
-----------------------	--

AGENDA

1. Einführung und Motivation
2. BTG-Konzept
3. Rollenverteilung
- 4. BTG-Annotationen**
5. Obligations
6. BPCC
7. Weitere Aufgaben
8. Fragebogen

4. Break The Glass Aufbau

Pflichtelemente

- Beginn und Ende mit doppelten spitzen Klammern
- Am Anfang zusätzlich BTG:
- Datenobjekte und Rechte

```
<<BTG:  
  {Roles}  
  objects = "..."  
  rights  = "..."  
  {BPCC}  
  {Obligations}  
  {Insert}  
>>
```

Optionale Elemente

- Definition von Rollen (Accessor, Activator)
- Angabe möglicher Einschränkungen (BPCC)
- Auflistung zugehöriger Obligations
- Einfügemodus

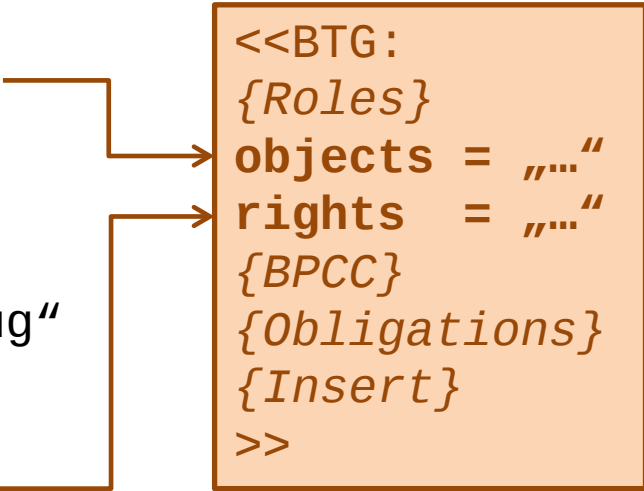
4. Break The Glass

Einschub: Terminologie

- `Something = list(x)`
auf den Folien bedeutet
`Something = „a, b, c, ...“`
- `Something = list((x,y))`
auf den Folien bedeutet
`Something = „(a,b), (c,d), ...“`

4. Break The Glass Elemente

- Datenobjekt(e) hinter dem Glas
 - Ein Objekt oder mehrere (a,b,...)
 - In Anführungszeichen
 - z.B.: `objects = „$student.notenauszug“`
- Rechte auf dem Datenobjekt
 - „read“ ODER „write“ ODER „update“
 - read und write sind disjunkt!
 - „update“ $\leq \Rightarrow$ „read,write“ (beides möglich)
 - z.B.: `rights = „update“`
- Diese Rechte gelten, nachdem das Glas gebrochen wurde



The diagram consists of an orange box with a brown border containing BTG syntax. Two brown arrows originate from the list items: one from 'objects = „\$student.notenauszug“' pointing to the 'objects' line in the box, and another from 'rights = „update“' pointing to the 'rights' line in the box.

```
<<BTG:  
{Roles}  
objects = „...“  
rights = „...“  
{BPCC}  
{Obligations}  
{Insert}  
>>
```

4. Break The Glass Elemente (2)

■ Zuweisung der Rollen

- Accessor
- Activator
- **Compensator** (erst mit Obligations)

```
<<BTG:  
{Roles}  
object = "..."  
right = "..."  
{BPCC}  
{Obligations}  
{Insert}  
>>
```

■ BTGActivator = „Arzt“

■ In der Praxis meist Authentifizierung gewünscht

- Auflistung aller Attribut-Wert-Paare
AuthnBTGActivator-attr = list((attribute,value))
 - z.B.: AuthnBTGActivator-attr = „(age,12), (eyes,brown)“
- Optional Angabe des Identity-Providers:
idp = „\$idp-address“

**BTGAccessor
analog!**

4. Break The Glass Elemente (3)

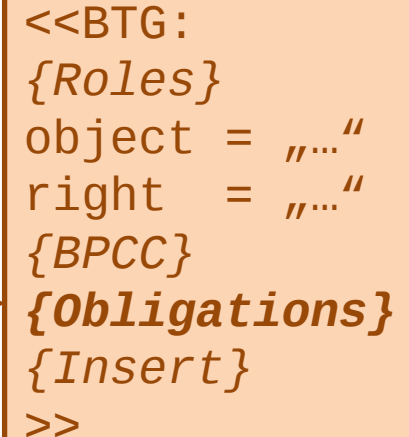
- Business Process Context Constraints
 - Einschränkungen bzgl. Prozesskontext
 - Bezug zum zugrundeliegenden BP
 - Syntax und Semantik später
- Zwei mögliche Bedingungen
 - Temporale Anforderungen:
start = „[BPCC]“
 - Konditionale Anforderungen:
exec = „[BPCC]“
 - Syntax und Semantik innerhalb der Anführungszeichen
mittels BPCC (später)



```
<<BTG:  
  {Roles}  
  object = "..."  
  right  = "..."  
  {BPCC}  
  {Obligations}  
  {Insert}  
>>
```

4. Break The Glass Elemente (4)

- Folgeaufgaben zur Reparatur des gebrochenen Glases
- Separate Modellierung
- In BTG-Annotation nur Auflistung aller Obligations



```
<<BTG:  
{Roles}  
object = "..."  
right = "..."  
{BPCC}  
{Obligations}  
{Insert}  
>>
```

- Obligations = list(\$obligation-ID)
 - z.B.: Obligations=„01,05,06“

4. Break The Glass Elemente (5)

- Ausführungsmodus
 - sequentiell: Insert = „seq“
ODER
 - parallel: Insert = „par“
- Angabe ist optional
- Standard: sequentiell

```
<<BTG:  
{Roles}  
object = „...“  
right = „...“  
{BPCC}  
{obligations}  
{Insert}  
>>
```

Heute außen vor 😊

4. Break The Glass

Zusammenfassung

Aufbau

- <<BTG: XXXXXXXXXX >>
- Datenobjekte und Rechte sind Pflichtangaben
- Rollenspezifikationen optional
- Obligations werden nur referenziert
- BPCC werden innerhalb von BTG-Annotation ausformuliert (temporal und kausal)

Syntax allgemein

- Something = „\$whatever“
- Auflistungen mittels Kommata
- Paare in Klammern
- Kein Semikolon

Aufgabe 4.1

BTG – Grundlagen

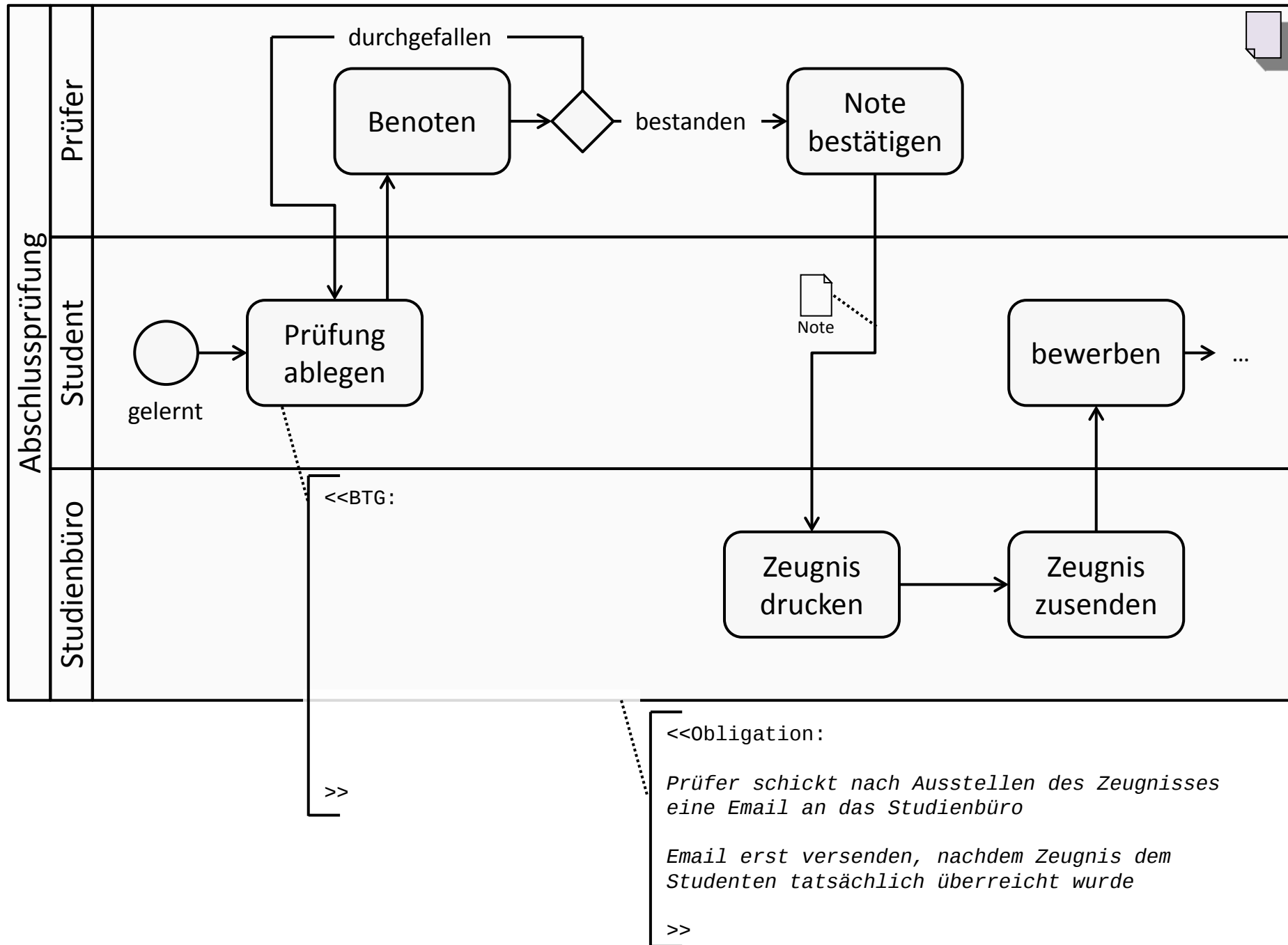


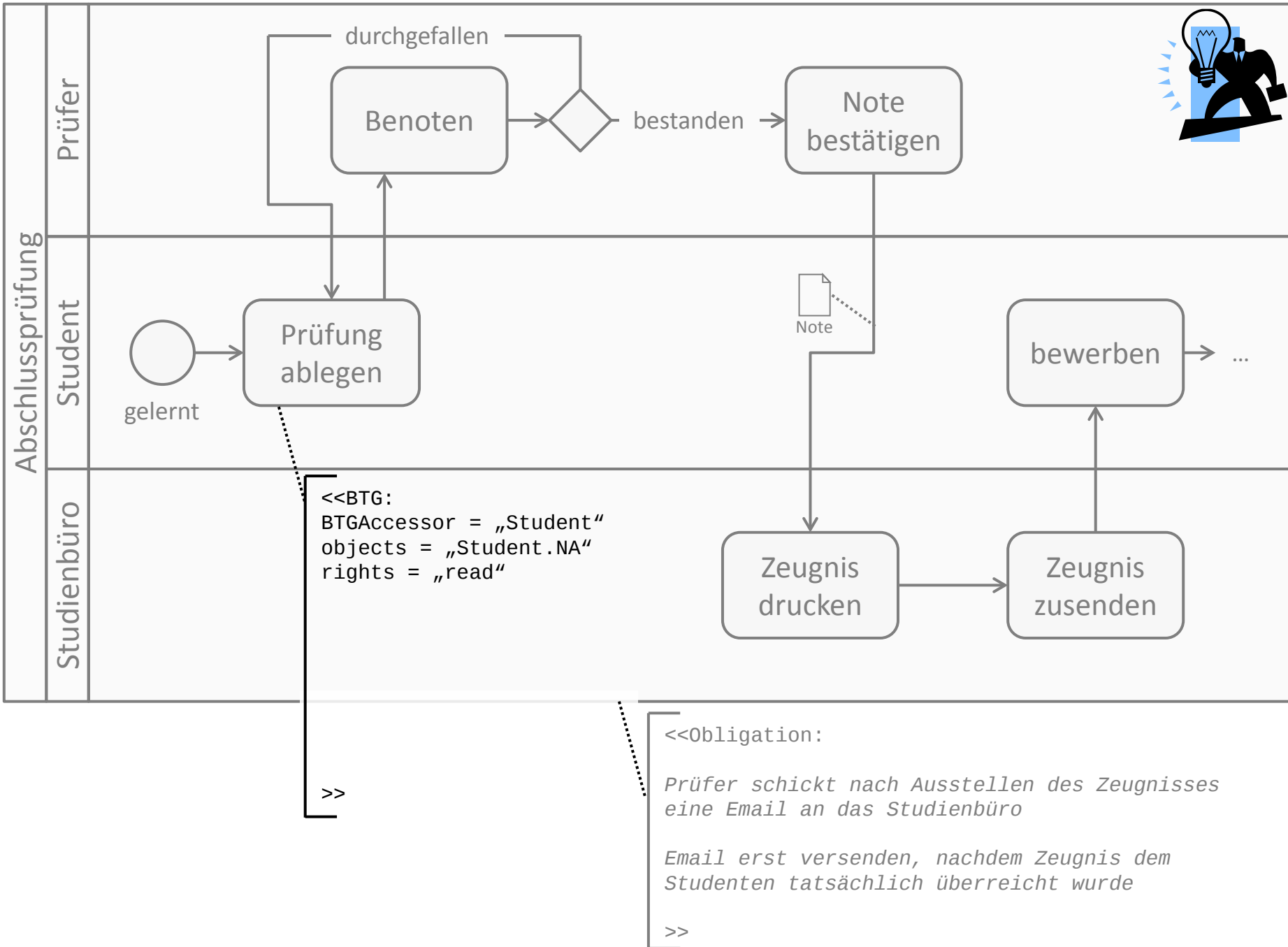
❖ Modellieren Sie die BTG-Annotation im vorliegenden Notenbeispiel. Modellieren Sie dabei Folgendes:

- ✓ Den Studenten als Accessor ohne Authentifizierung
- ✓ Das Datenobjekt, nämlich den Notenauszug des Studenten („Student.NA“)
- ✓ Die darauf verfügbaren Leserechte des Studenten

❖ Vorerst nicht zu modellieren sind:

- ✓ BPCC
- ✓ Obligations
- ✓ Activator





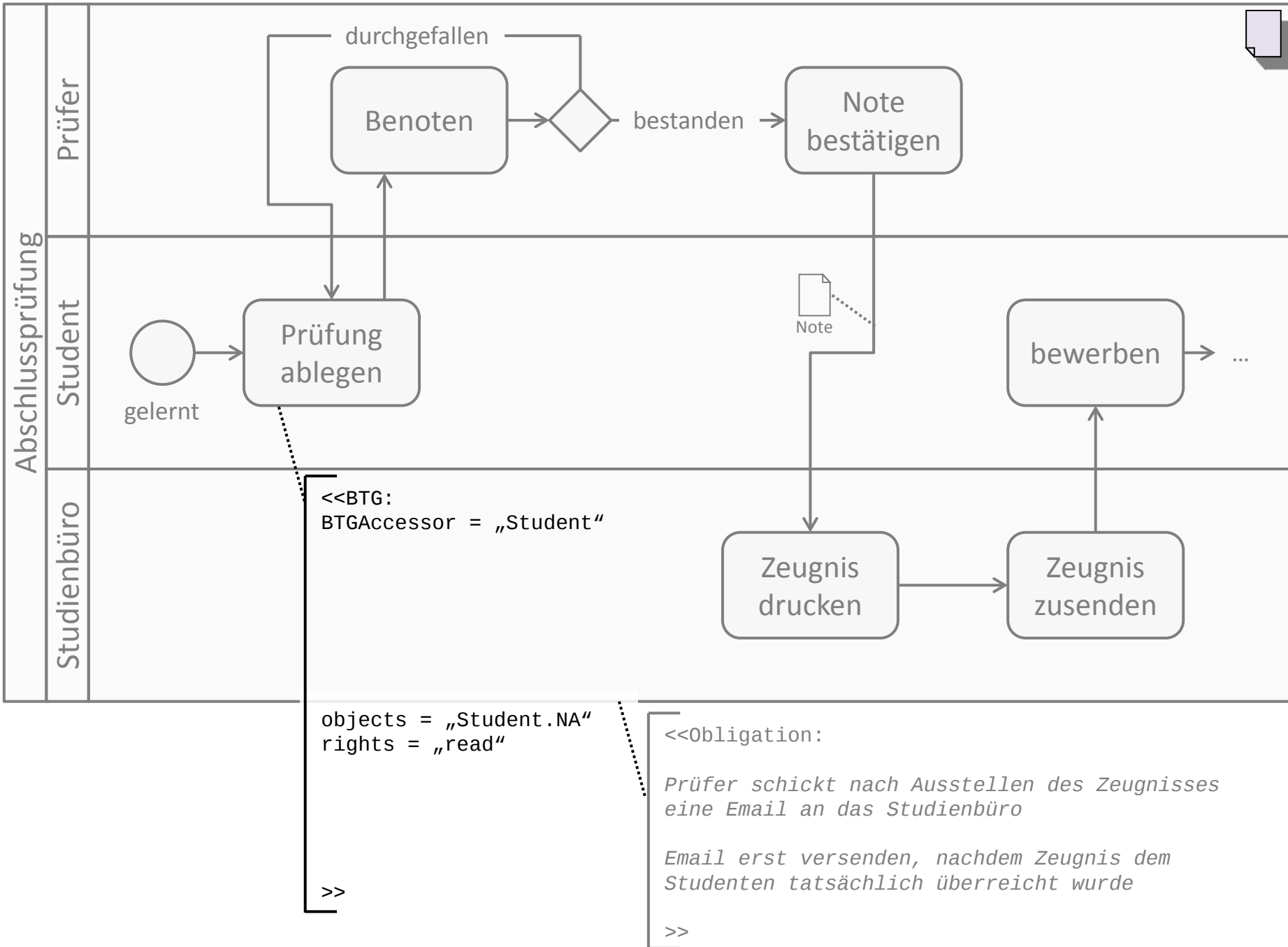
Aufgabe 4.2

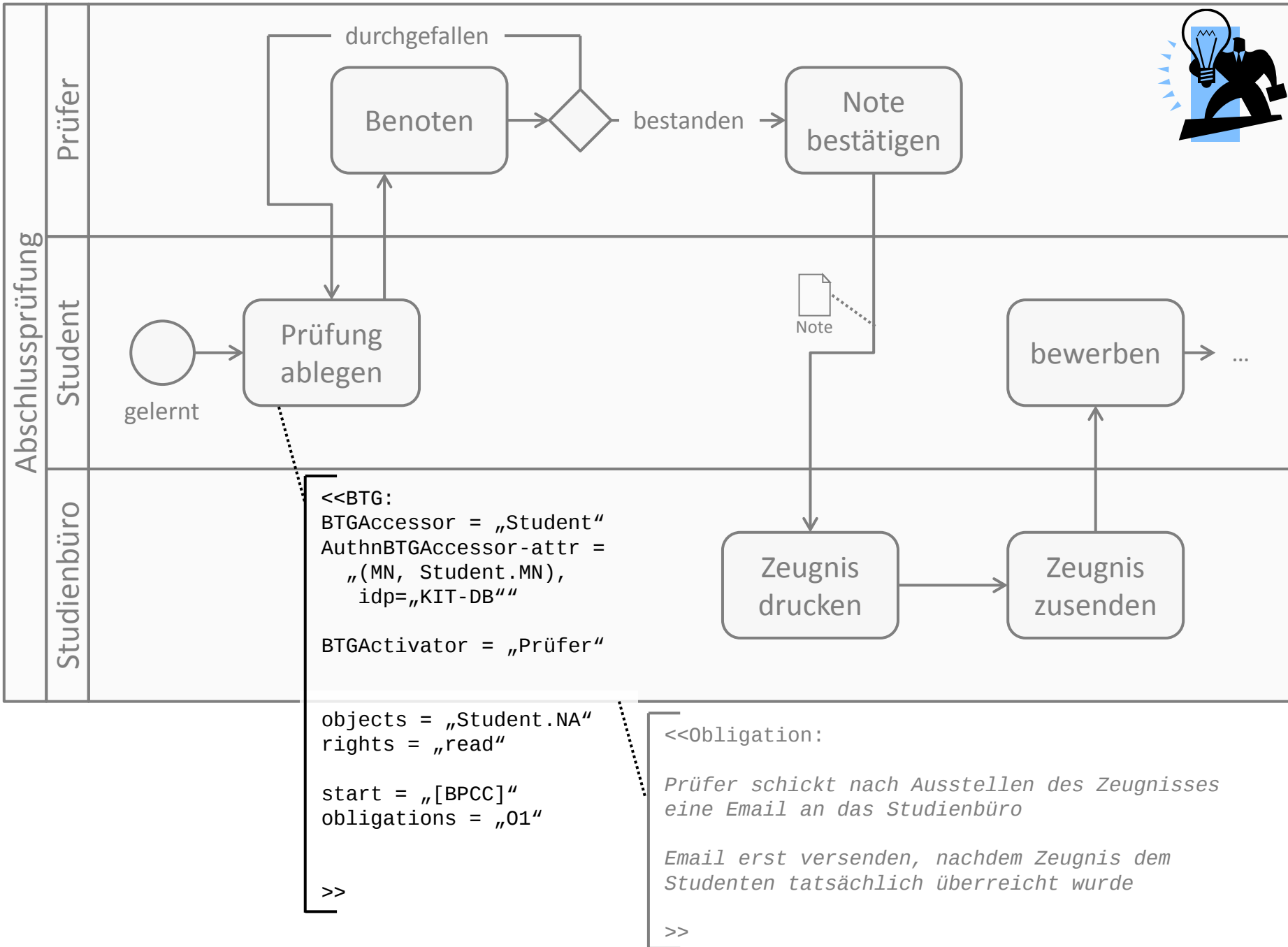
BTG für Fortgeschrittene



❖ **Erweitern Sie die Modellierung aus dem ersten Aufgabenteil um die bisher fehlenden Aspekte:**

- ✓ Den entsprechenden Activator
- ✓ Der Student muss sich authentifizieren
 - *Die Matrikelnummer (MN) muss seiner eigenen (Student.MN) entsprechen*
 - *Authentifizierungsdaten werden bereitgestellt von KIT-DB*
- ✓ Temporale Bedingungen an den Beginn der Ausführung
 - *BTG soll möglich sein, sobald eine BPCC erfüllt ist*
 - *Verwenden Sie für jegliche BPCC bitte „[BPCC]“ als Platzhalter*
- ✓ Die Verknüpfung zur zugehörigen Obligation
 - *zum Versenden einer E-Mail*
 - *mit der ID O1*





AGENDA

1. Einführung und Motivation
2. BTG-Konzept
3. Rollenverteilung
4. BTG-Annotationen
- 5. Obligationen**
6. BPCC
7. Weitere Aufgaben
8. Fragebogen

5. Obligationen

Einführung

- Syntaktischer und semantischer Aufbau äquivalent zu Break The Glass Annotationen
- Annotationen analog gekennzeichnet
- BPCC analog verwendet
- Formulierung der Obligation an sich im Rahmen eines Musters (pattern)
- Muster definieren die allgemeine Obligations-Aufgabe und die dafür festzulegenden Parameter
 - z.B.: E-Mail versenden, Parameter wie Sender/Empfänger

5. Obligationen

Aufbau

Pflichtelemente

- Beginn und Ende mit doppelten spitzen Klammern
- Am Anfang zusätzlich `Obligation:`
- Vergabe einer `id` (numerisch)
- Festlegung auf ein Muster (Aufgabentyp)

```
<<Obligation:  
id = "..."  
{Role}  
pattern = "..."  
{Parameter}  
{BPCC}  
>>
```

Optionale Elemente

- Definition der Compensator-Rolle
- Auflistung der Parameter für das Pattern (verpflichtend, falls Pattern Parameter verlangt)
- Angabe möglicher Einschränkungen (BPCC)

5. Obligationen Elemente

- Vergabe einer eindeutigen ID
 - Ox, wobei x eine Zahl ist
 - id = „043“
- Referenzierung der Obligations aus BTG-Annotationen heraus über diese Identifikationsnummer



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

5. Obligationen Elemente (2)

- Definition der Compensator-Rolle
 - optional
 - nur nötig, falls Obligation von Menschen (und nicht automatisiert) erfolgt



```
<<Obligation:  
  id = „...“  
  {Role}  
  pattern = „...“  
  {Parameter}  
  {BPCC}  
>>
```

- OGCompensator = „Arzt“
- In der Praxis meist Authentifizierung gewünscht
 - Auflistung aller Attribut-Wert-Paare
AuthnOGCompensator-attr = list((attribute,value))
 - z.B.: AuthnOGCompensator-attr = „(age,12), (eyes,brown)“
 - Optional Angabe des Identity-Providers:
idp = „\$idp-address“

**Accessor und Activator
existieren hier nicht!**

5. Obligationen Elemente (3)

- Festlegung der in der Obligation durchzuführenden Aufgabe
- Dazu: Festlegung eines Musters
- Muster sind vorgegeben, aktuell:
 - SendEmail
 - AduitAccess
- Festlegung auf eines dieser Muster pro Obligation
- Gegebenenfalls sind Parameter zu definieren
- Zusätzliche Muster können vom Prozessmodellierer in BPMN einmalig modelliert und dann im Rahmen von Obligations immer wieder verwendet werden



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

5. Obligationen Elemente (4)

- pattern = „SendEmail“
- Dient dem Versenden einer E-Mail



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

Parameter

- from Sender
- to Empfänger
- subject Betreffzeile
- body Inhalt der E-Mail
- attachment Dateianhang

5. Obligationen Elemente (6)

- pattern = „AuditAccess“
- Nachvollziehen eines Datenzugriffs
- Als Folge eines BTG-Zugriffs

```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

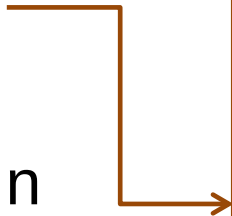
Parameter

- auditpolicy Definition einer Richtlinie für der
- start Zeitpunkt, zu dem Audit beginnt
- end Zeitpunkt, zu dem Audit endet

Heute außen vor 😊

5. Obligationen Elemente (7)

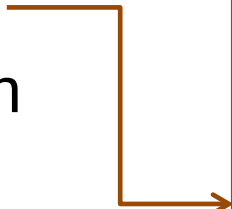
- Definition der Parameter für das gewählte Muster
- zu definierende Parameter hängen vom Muster ab
- Falls für das gewählte Muster keine Parameter nötig sind entfällt OGParameter
- OGParameter = list((name,value))
 - OGParameter=„(from,Arzt),(to,Patient),(subject,“Ihr Arztbesuch”),(body,“Hallo...”)”
- Falls ein einzelner Parameter nicht nötig ist wird dieser weggelassen



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

5. Obligationen Elemente (8)

- Verwendung von BPCC zur Festlegung der Werte von Parametern
- Es sei beispielsweise der Empfänger-Parameter des „SendEmail“-Patterns gegeben
- Empfänger könnte nun die Person sein, die eine bestimmte Aufgabe ausgeführt hat
- Die Referenzierung dieser Person wäre hier über einen BPCC möglich
 - (from, [BPCC])



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

5. Obligationen Elemente (9)

■ Business Process Context Constraints

- Einschränkungen bzgl. Kontext
- Bezug zum zugrundeliegenden BP
- Syntax und Semantik später



```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

■ Zwei mögliche Bedingungen

- Temporale Anforderungen:
start = „...“
- Konditionale Anforderungen:
exec = „...“
- Syntax und Semantik innerhalb der Anführungszeichen
mittels BPCC
- Default: Obligations werden automatisch direkt nach BTG
ausgeführt; hierfür wäre also keine BPCC nötig

5. Obligationen

Zusammenfassung

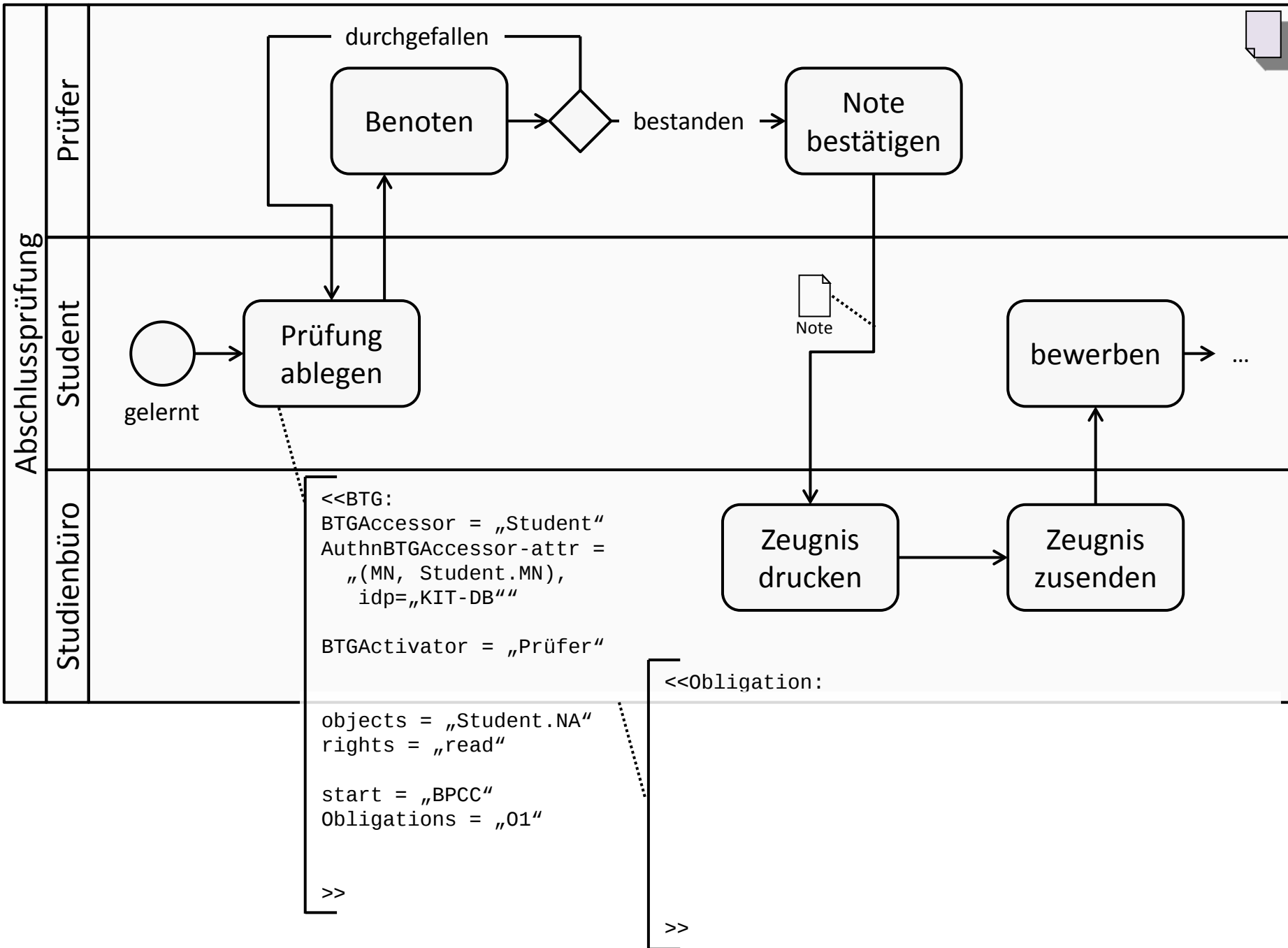
- Aufbau ähnlich zu BTG-Annotationen
- Festlegung einer ID
- ggf. Compensator-Rolle spezifizieren
- Festlegung auf eine Aufgabe: pattern
- Definition der Parameter für gewähltes Muster
- Rahmenbedingungen mittels BPCC festlegen

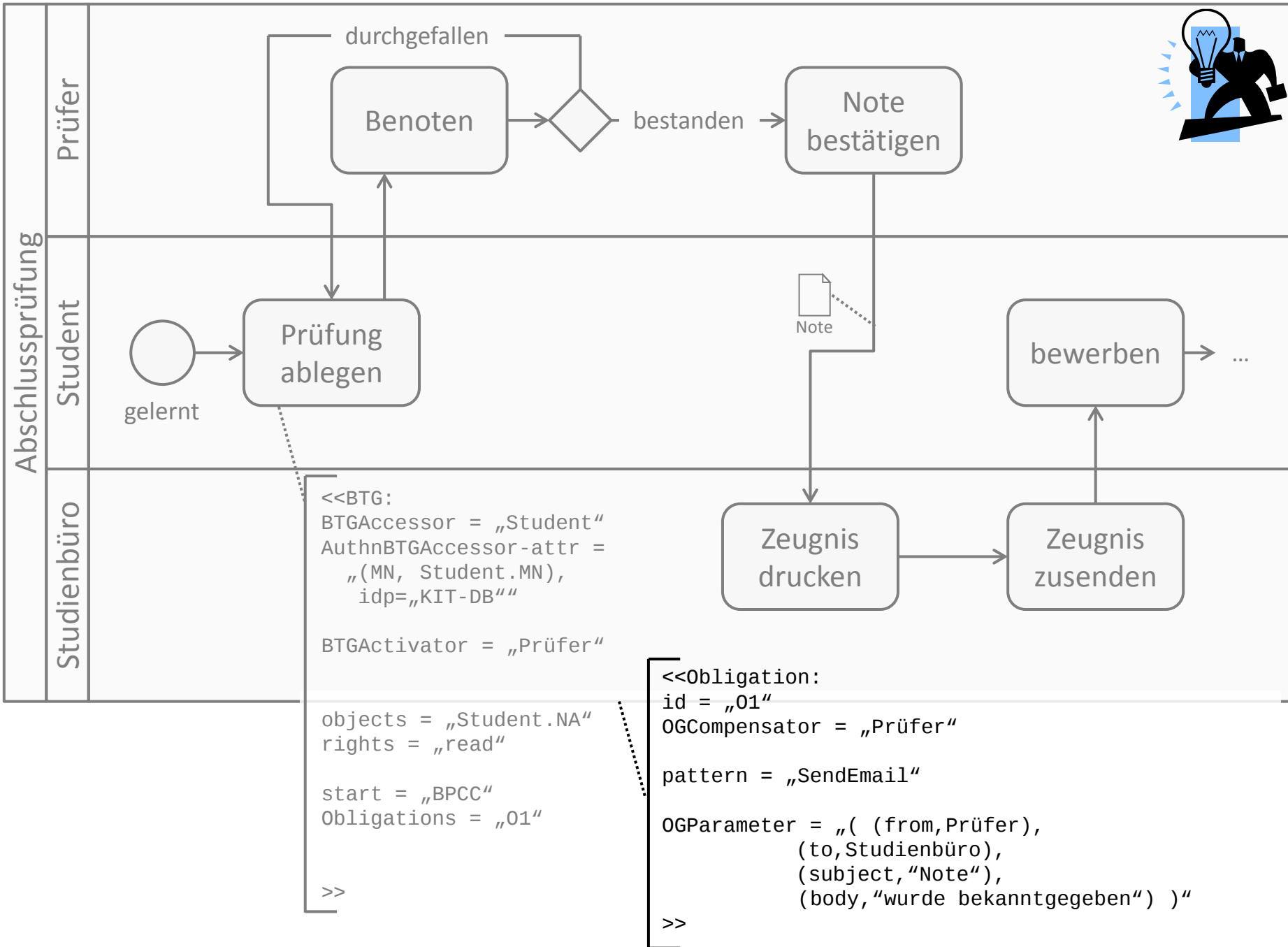
Aufgabe 5

Modellierung einer Obligation



- ❖ **Modellieren Sie die Obligation O1 zum Versenden einer E-Mail (nach der BTG-Ausführung) durch den Prüfer für das Notenvergabebeispiel**
- ❖ **Verwenden Sie für die Parameter folgende Werte:**
 - ✓ Absender: Prüfer
 - ✓ Empfänger: Studienbüro
 - ✓ Betreff: „Note“
 - ✓ Inhalt: „wurde bekanntgegeben“
 - ✓ Anhang: *keiner*





AGENDA

1. Einführung und Motivation
2. BTG-Konzept
3. Rollenverteilung
4. BTG-Annotationen
5. Obligations
- 6. BPCC**
7. Weitere Aufgaben
8. Fragebogen

6. BPCC

Einführung

- Werden innerhalb von BTG-Annotationen oder Obligations modelliert
- Modellierung von durch den zugrundeliegenden BP gegebenen Bedingungen
 - Temporal: start = „[BPCC]“
 - Konditional: exec = „[BPCC]“
- Durch Einbettung in BTG/OG keine syntaktische Kennzeichnung von Anfang und Ende nötig
- Finden ebenfalls Verwendung in OGParameter

6. BPCC Konzept

- Funktionen über
 - Datenobjekte
 - Aktivitäten
 - Akteure
 - Sonstige BPMN-Elemente
- werden mittels logischer Operatoren abgeglichen mit
 - Werten/Mengen
 - Anderen Funktionen

6. BPCC

Funktionen

- Funktionen über
 - Datenobjekte, Aktivitäten, Akteure und andere Elemente
 - Optional: Angabe einer Zahl für
 - Anzahl der zu betrachtenden Datenobjekte
 - Anzahl der zu betrachtenden Ausführungen der Aktivität
 - Anzahl der zu betrachtenden Akteure
 - Schachtelungen von Funktionen sind zulässig, wenn der Rückgabewert der inneren Funktion zum Argument der äußeren passt
-
- | | |
|---|---|
| ■ <code>owner(\$student.notenauszug)</code> | Betroffener des Notenauszugs |
| ■ <code>end-time (\$lesezugriff,3)</code> | Ende der Ausführung von insgesamt 3 Lesezugriffen |

6. BPCC

Einschub

- Rollennamen (z.B. Student, Prüfer, Professor) dürfen in BPCC nicht zur Referenzierung verwendet werden
- Daher sind Funktionen zum Zugriff auf Personen und Rollen nötig

6. BPCC Funktionen (2)

Funktionsname	Argumenttyp	Rückgabewert
data-user	Datenobjekt{, Recht}	Individuum, das auf dieses Datenobjekt zugreift
owner	Datenobjekt	Betroffener, dessen persönliche Daten im Datenobjekt stehen
performer	Aktivität	Individuum, das diese Aufgabe (Aktivität) ausführt
start-time	Datenobjekt{, Recht}	Zeitpunkt, zu dem Zugriff auf dieses Datenobjekt beginnt
	Aktivität	Zeitpunkt, zu dem Ausführung dieser Aktivität beginnt
end-time	Datenobjekt{, Recht}	Zeitpunkt, zu dem Zugriff auf dieses Datenobjekt beendet ist
	Aktivität	Zeitpunkt, zu dem Ausführung dieser Aktivität beendet ist
data-object	Aktivität{, Recht}	Alle Datenobjekte, auf die während dieser Aktivität zugegriffen wird
tasks	Rolle	Aktivitäten, die von dieser Rolle bereits ausgeführt wurden
	Individuum	Aktivitäten, die von diesem Individuum bereits ausgeführt wurden

6. BPCC

Funktionen (3)

Funktionsname	Argumenttyp	Rückgabewert
duration	Aktivität	Zeit, die eine Aktivität bereits aktiv ist bzw. gedauert hatte
frequency	Datenobjekt{, Recht}	Anzahl der Nutzungen des Datenobjekts
	{, Group}	zusätzliche Gruppierung der betrachteten Aktivitäten
fulfilled	Gateway, Bedingung	„true“, falls die genannte Bedingung bei Gateway eingetroffen ist
	Ereignis	„true“, falls ein Ereignis eingetreten ist
	Nachricht	„true“, falls Nachrichtenfluss ankommt
executed	list(Aktivität)	„true“, falls Aktivitäten ausgeführt wurden
owned-objects	Individuum	Datenobjekte, die persönliche Daten des Betroffenen enthalten
used-objects	Individuum{, Recht}	Datenobjekte, auf die das Individuum zugegriffen hat
role	Aktivität	Rolle, die die Aktivität ausführt

6. BPCC

Funktionsoperatoren

- Abgleich von Funktionsrückgabewerten
- Abgleich mit einem Objekt/Wert oder einer Menge davon

Operator	Bedeutung
>	Größer
<	Kleiner
>=	Größer oder gleich
<=	Kleiner oder gleich
==	Gleichheit
≠	Ungleichheit
∈	Element der Menge
∉	Kein Element der Menge

- Beispiele:
 - `owner($student.notenauszug) == „Diplomer“`
 - `owner($student.notenauszug) ∈ „Studenten“`
 - `start-time („lesen“).hours > 50`
 - `start-time („lesen“) ∉ „abends“`

6. BPCC

Komplexe Operatoren

- Logische Verknüpfung von zwei Funktionen bzw. Funktionen mit Funktionsoperatoren

ODER

- Vergleich der Rückgabewerte zweier Funktionen

- Beispiele:

- `owner($student.notenauszug) == „Diplomer“
 ^ start-time („lesen“) > „50“`
- `owner($student.notenauszug) == performer („lesen“)`

Operator	Bedeutung
$\wedge$	UND
$\vee$	ODER
<code>==</code>	Gleichheit
<code>≠</code>	Ungleichheit

6. BPCC Einbettung

- Nutzung in BTG/OG
- Bedingungen an die Ausführung
 - Temporal: start = „[BPCC]“
 - Konditional: exec = „[BPCC]“

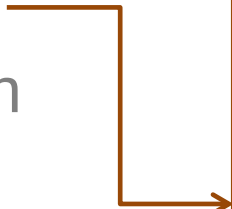
```
<<BTG:
... ..
start = „ end-time („Lesen“) < start-time („Schreiben“) “
exec  = „ owner($student.notenauszug)==„Diplomer“
        ^ tasks(„Diplomer“) == „Lesen“ “
... ..
>>
```

- „Lesen“ muss vor Beginn von „Schreiben“ abgeschlossen sein
- Es handelt sich um den Notenauszug des Diplomiers
- Der Diplomer hat die Aufgabe „Lesen“ und nur diese ausgeführt

6. BPCC

Verwendung in Obligations (Wdh.)

- Verwendung von BPCC zur Festlegung der Werte von Parametern
- Es sei beispielsweise der Empfänger-Parameter des „SendEmail“-Patterns gegeben
- Empfänger könnte nun die Person sein, die eine bestimmte Aufgabe ausgeführt hat
- Die Referenzierung dieser Person wäre hier über einen BPCC möglich
 - **(from, performer („AufgabeX“))**



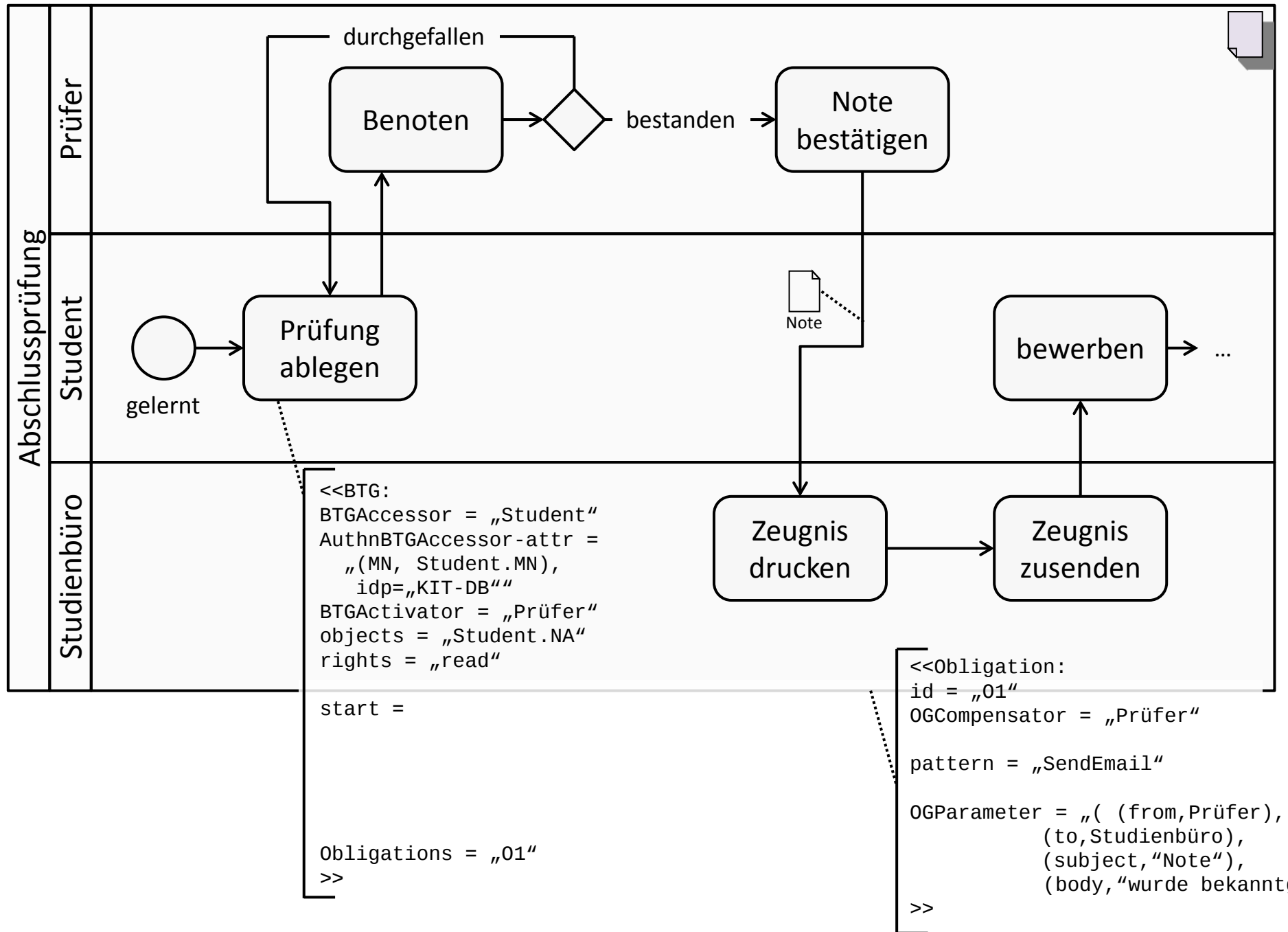
```
<<Obligation:  
id = „...“  
{Role}  
pattern = „...“  
{Parameter}  
{BPCC}  
>>
```

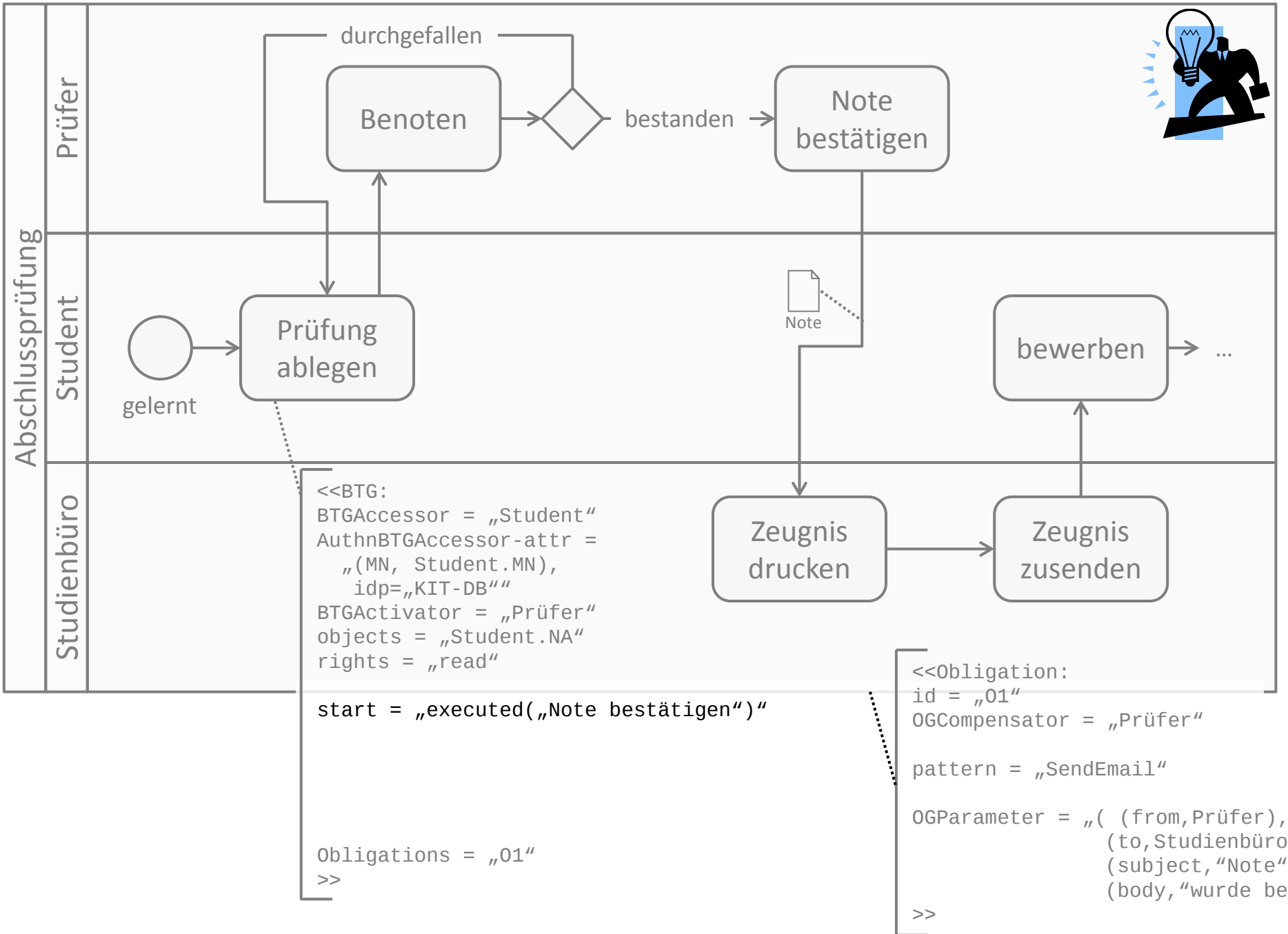
Aufgabe 6.1

Simpler BPCC



- ❖ **Gegeben sei die BTG-Annotation aus Aufgabe 4. Modellieren Sie nun die dort fehlenden Rahmenbedingungen zur Ausführung**
- ❖ **Verwenden Sie dabei BPCC**
- ❖ **Beginnen Sie mit der Modellierung folgender Bedingung:**
 - ✓ Der Student erhält sein Vorab-Zeugnis erst (sprich das Glas darf erst dann gebrochen werden), wenn die Note bestätigt wurde



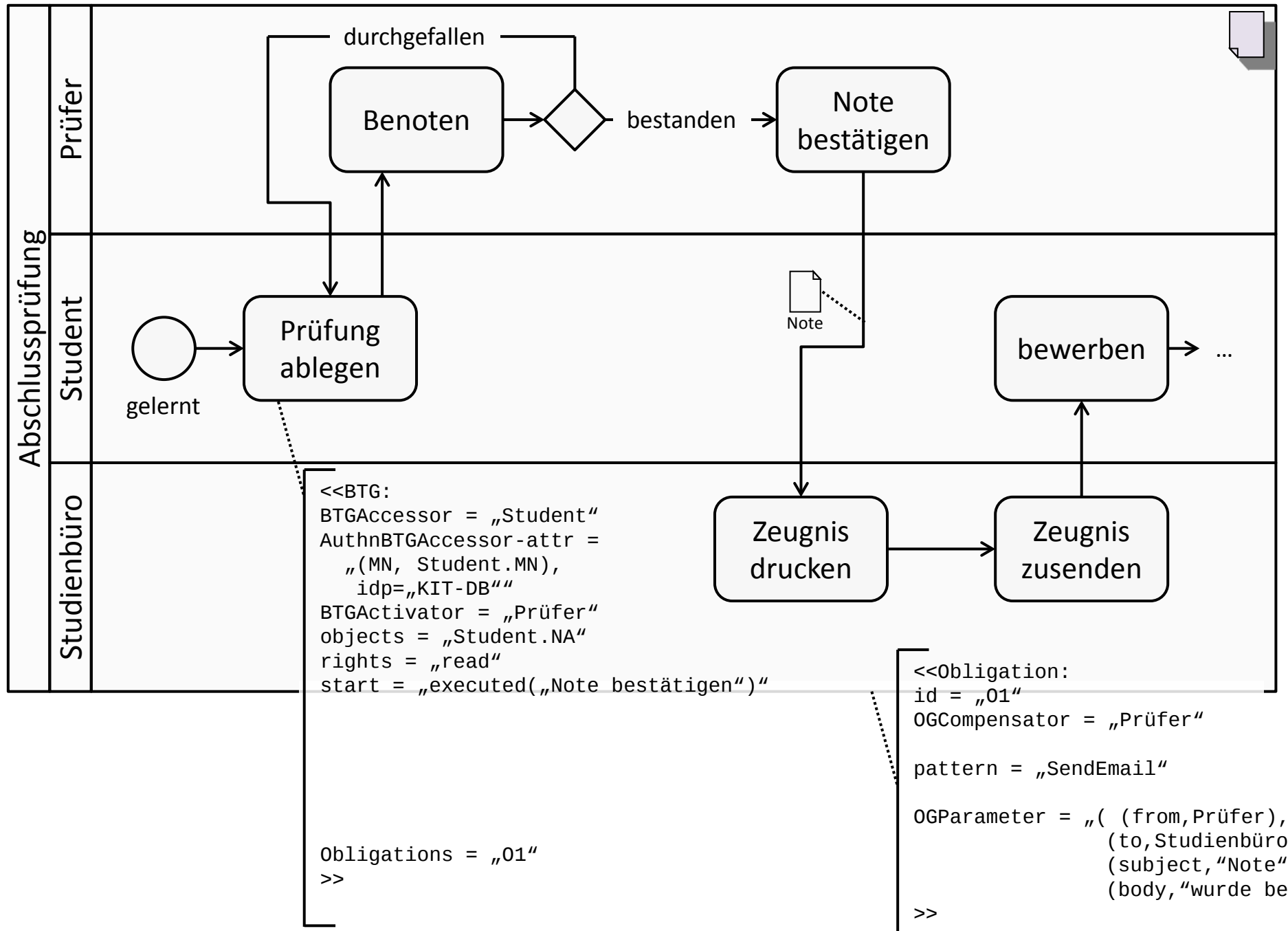


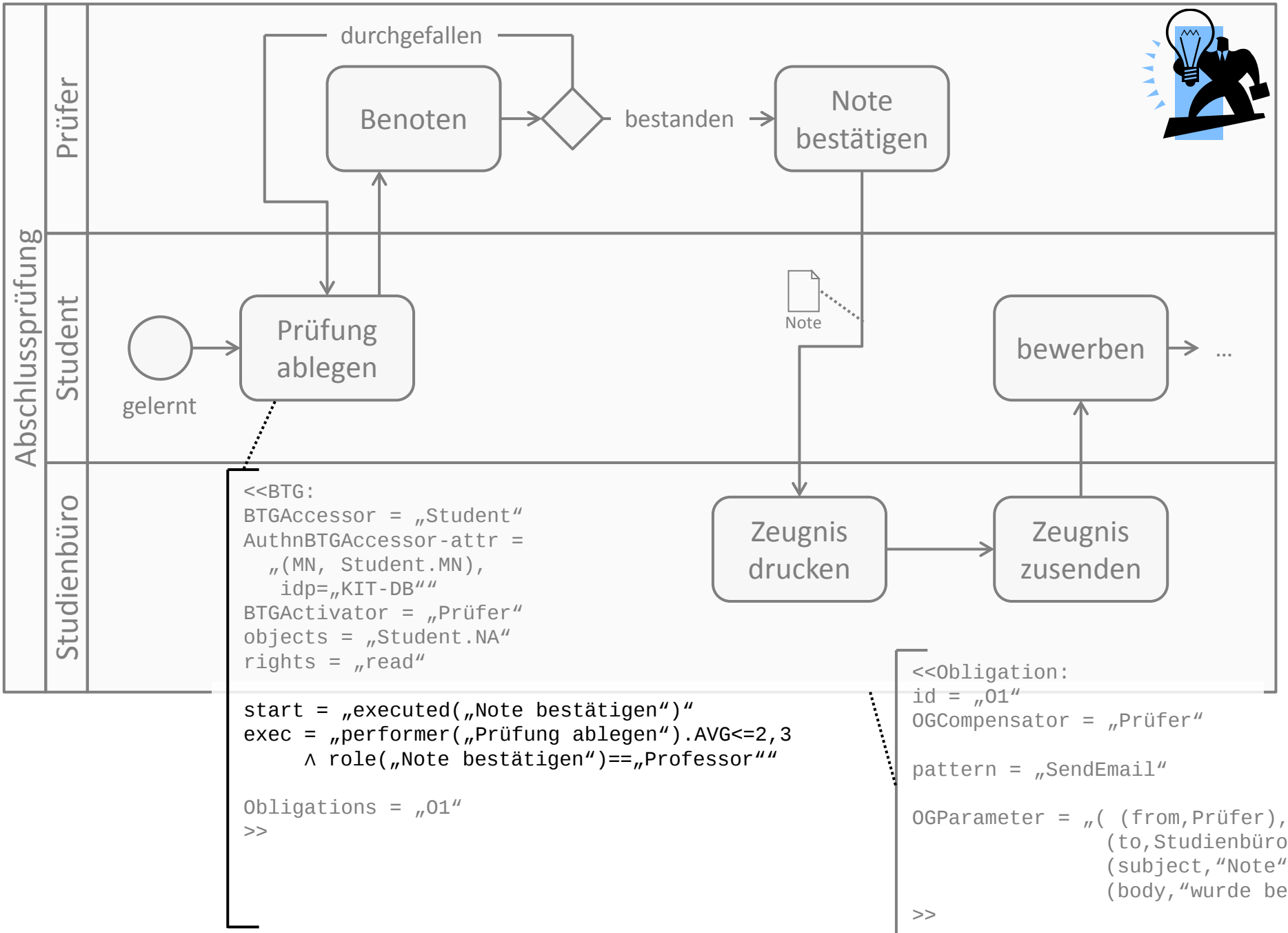
Aufgabe 6.2

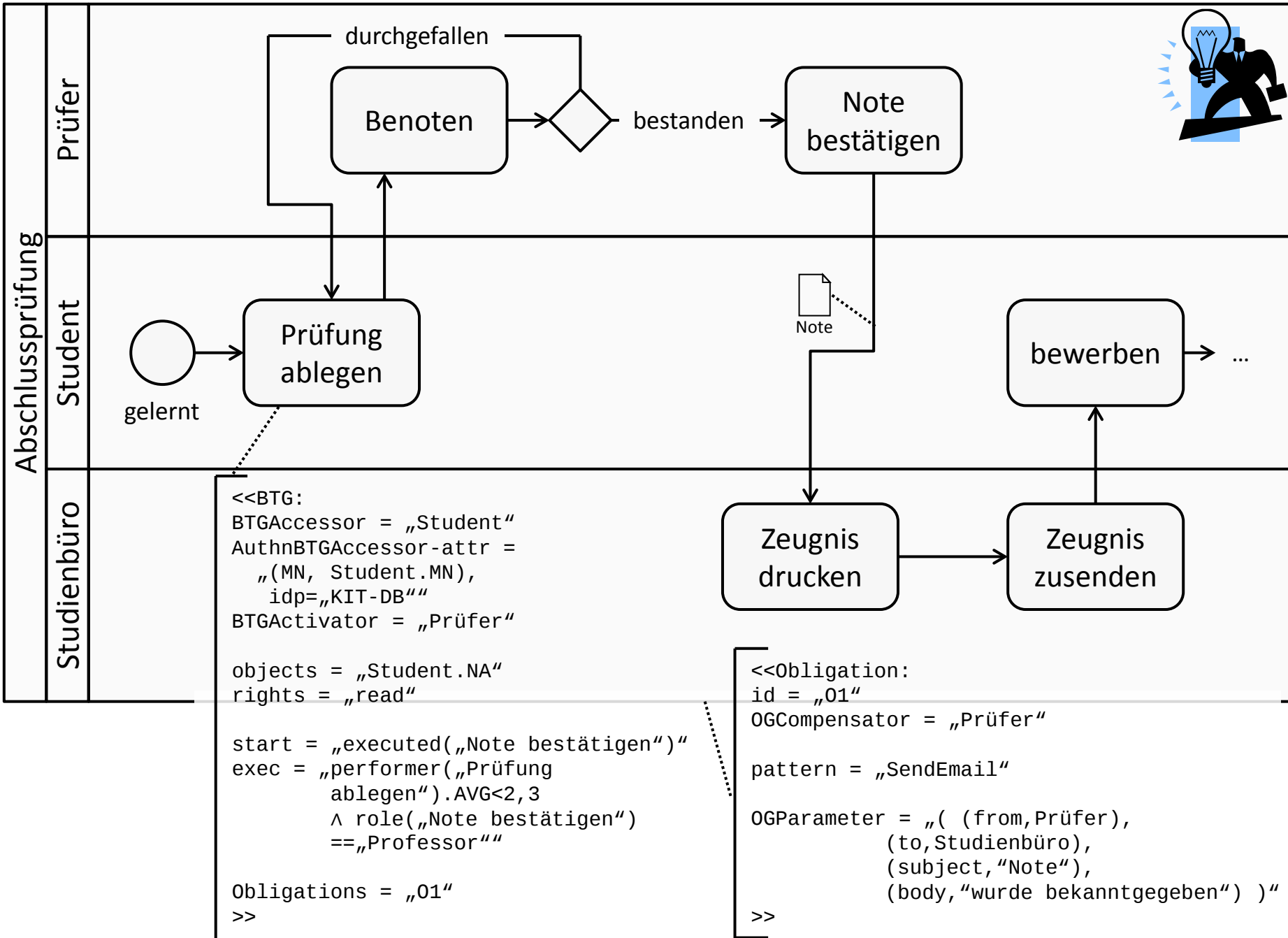
Fortgeschrittener BPCC



- ❖ **Neuen Regelungen des Prüfungsausschusses zufolge darf ein Vorab-Zeugnis nur dann ausgestellt werden, wenn**
 - ✓ Der Student mindestens einen Notenschnitt (AVG) von 2,3 hat
 - ✓ Der Prüfer Professor ist
- ❖ **Erweitern Sie den BPCC aus Aufgabe 6.1 entsprechend**







AGENDA

1. Einführung und Motivation
2. BTG-Konzept
3. Rollenverteilung
4. BTG-Annotationen
5. Obligations
6. BPCC
- 7. Weitere Aufgaben**
8. Fragebogen

7. Weitere Aufgaben

Einschub: Nötige Zusätze

- Auf einzelne Attribute eines Zeitwerts kann mittels „years“, „months“, „days“, „hours“, „minutes“, „seconds“ zugegriffen werden
 - `duration(„AufgabeX“).hours`

Aufgabe 7.1

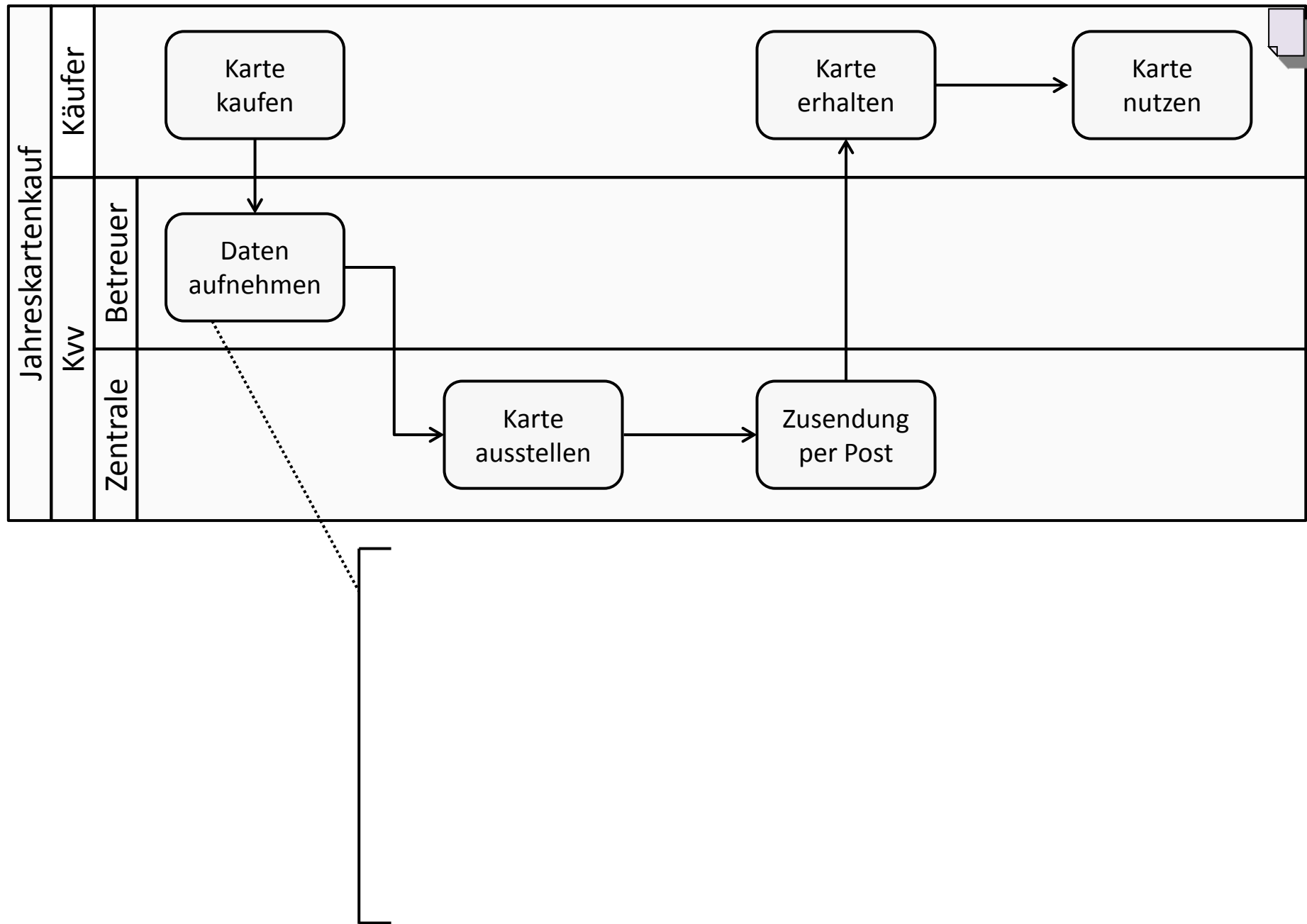
KVV-Jahreskarte

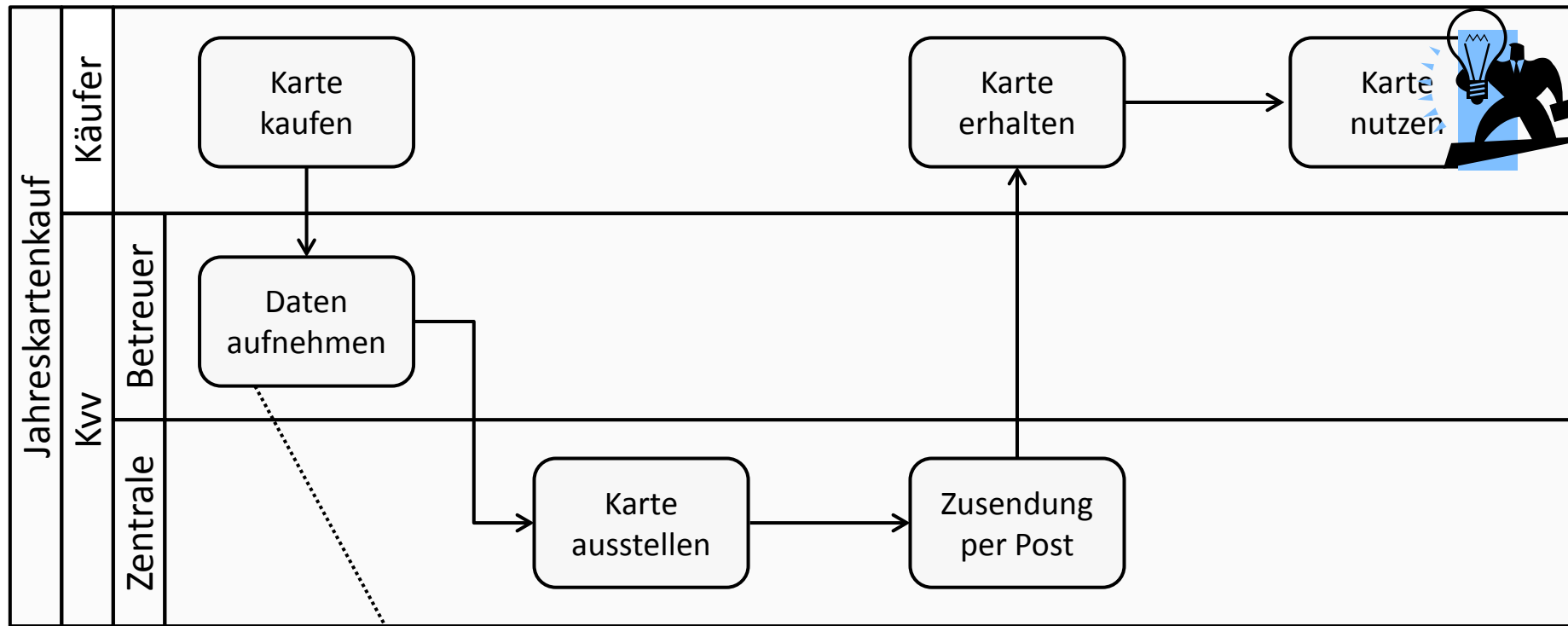


- Ein Kunde kauft eine KVV-Jahreskarte am Schalter beim KVV-Kundenbetreuer
- Im Normalfall wird diese nach einigen Tagen von der KVV-Zentrale ausgestellt und dem Kunden per Post zugesandt
- Unter gewissen Umständen besteht allerdings die Möglichkeit, eine Vorab-Karte direkt am Schalter zu erhalten
 - Wenn aktuell einer der ersten 5 Tage des Monats ist
 - Der Kunde erwachsen ist
- Dazu muss der Betreuer des KVV auf die Karten-DB zugreifen, auf die ihm sonst der Zugriff verwehrt bliebe (im Normalfall hat nur die Zentrale die entsprechenden Zugriffsrechte)

❖ Modellieren Sie diese Ausnahme mittels BTG

- ✓ Betreuer ist Accessor
- ✓ Er identifiziert sich mittels eines Passworts (PW), das mit seinem eigenen übereinstimmen muss
- ✓ Da der Computer die Bedingungen prüft ist kein menschlicher Activator nötig
- ✓ Der Zugriff auf die „\$Karten-DB“ erfolgt lesend und schreibend
- ✓ Temporale Startbedingung: Einer der ersten 5 Tage des Monats
- ✓ Konditionale Bedingung: Der Käufer ist erwachsen
- ✓ Verpflichtungen (Obligations) entstehen nicht





```

<<BTG:
BTGAccessor = „Betreuer“
AuthnBTGAccessor-attr = „(PW,Betreuer.PW)“

objects = „$Karten-DB“
rights = „update“

start = „start-time(„Daten aufnehmen“).day
      <= 5“
exec = „performer(„Karte kaufen“).age
      >= 18“

>>
  
```

Aufgabe 7.2

Bibliotheksszenario: Verliehenes Buch

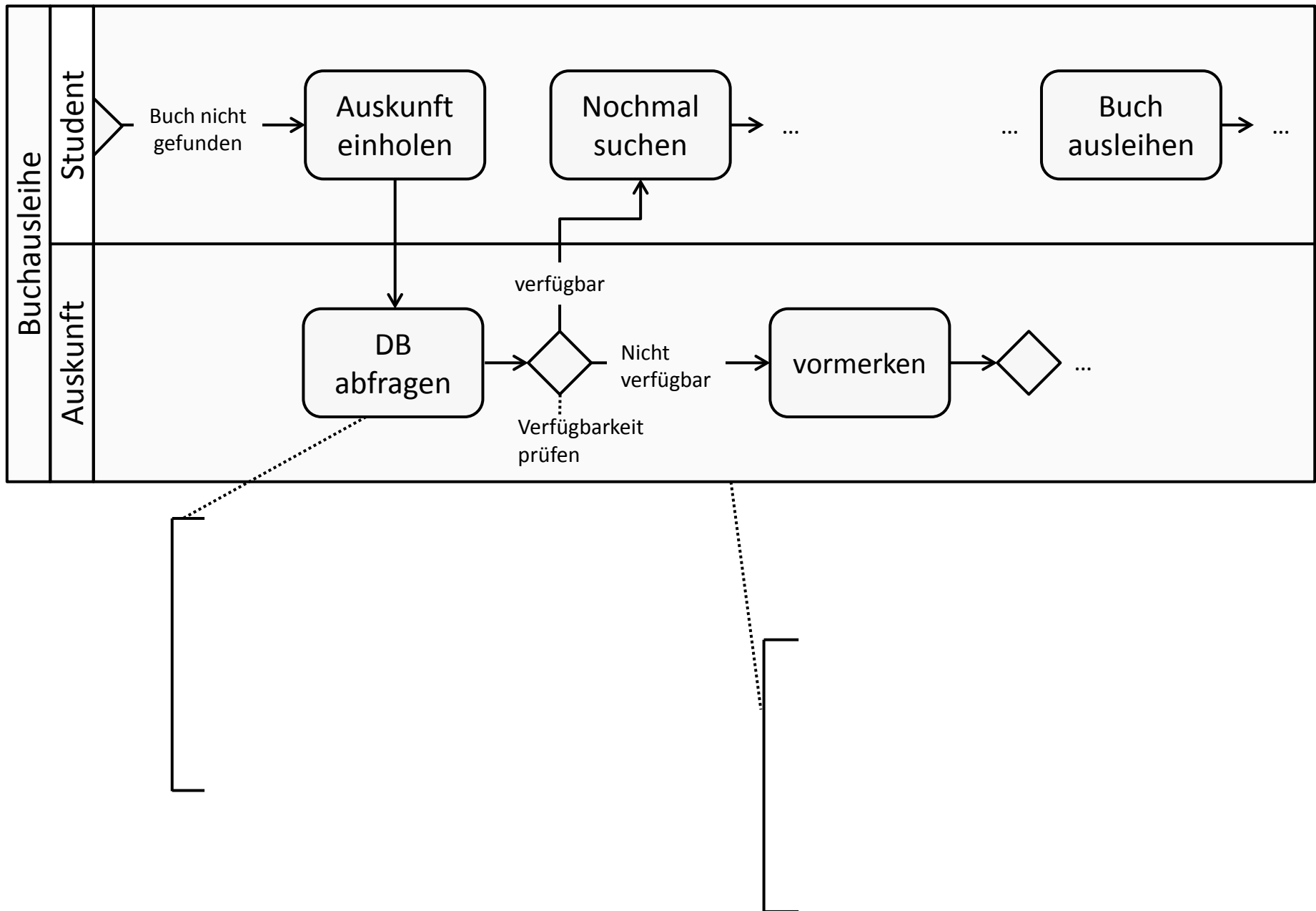
- Student: Suche nach „Buch“
 - Buch ist bereits verliehen (an irgendwen; für Student unbekannt)
- Student möchte Kontaktdaten des Ausleihers wissen, um diesen zu kontaktieren
 - Bücherei-Auskunft kennt diese Kontaktdaten
 - Der Ausleiher muss der Herausgabe seiner Daten zustimmen
- Im Nachgang muss die Auskunft eine E-Mail an den „Leiter“ der Bücherei schicken
 - Dies ist nur zu tun, falls der Ausleiher die Rolle KITStudent innehat

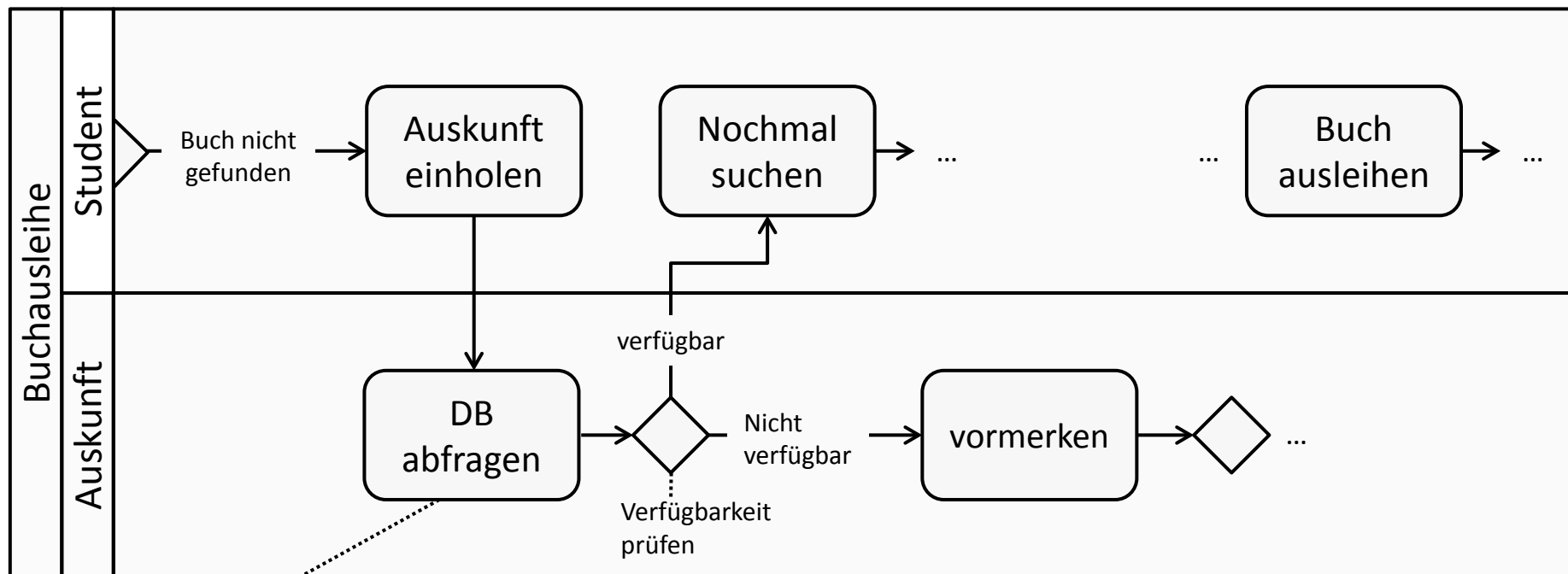
❖ Modellieren Sie diese Situation mittels BTG

- ✓ Accessor: Student (Auszuweisen mittels Matrikelnummer MN; Überprüfung durch die \$KIT-DB)
- ✓ Activator: Ausleihender (Person, die „Buch“ ausgeliehen hat; der „Ausleiher“)
- ✓ Leserechte für die Kontaktdaten des Ausleihers
- ✓ BTG sollte natürlich nur dann Anwendung finden, wenn wirklich kein Buch mehr in der Bücherei vorrätig ist

❖ Modellieren Sie auch eine entsprechende Obligation

- ✓ Compensator: Die Auskunft
- ✓ Vergessen Sie die zusätzliche Bedingung nicht





```

<<BTG:
  BTGAccessor = „Student“
  AuthnBTGAccessor-attr = „ (MN,Student.MN),
  idp=„$KIT-DB“ “
  BTGActivator = „Ausleiher“
  objects = „$Ausleiher.Kontaktdaten“
  rights = „read“
  exec = „fulfilled(„Verfügbarkeit
  prüfen“,„Nicht verfügbar“) == true“
  obligations = „01“
>>

```

```

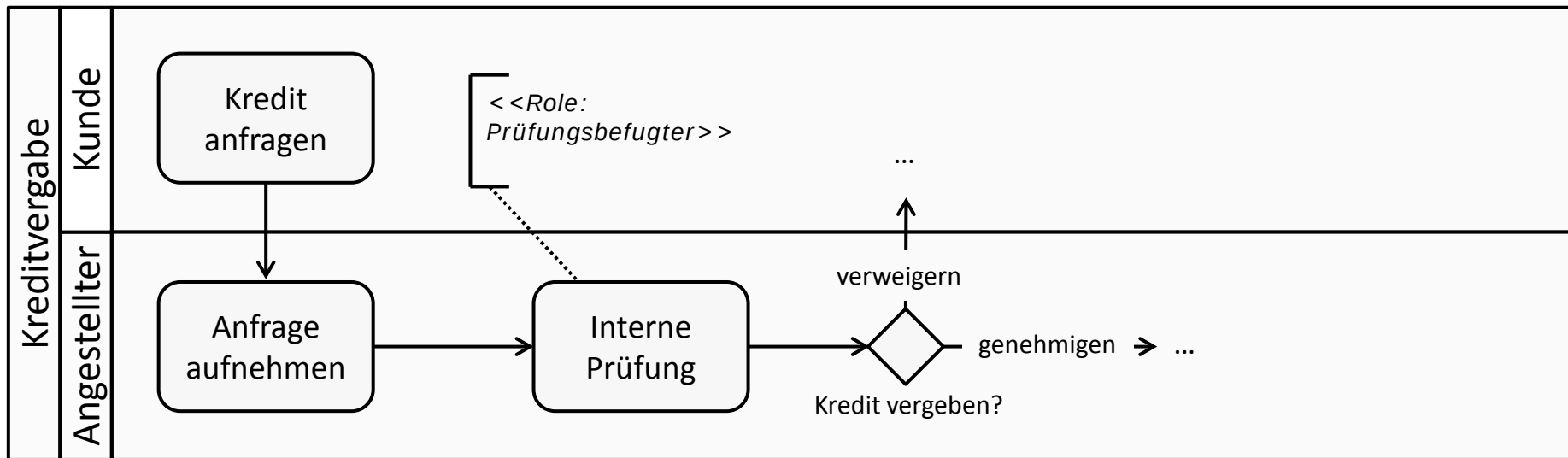
<<Obligation:
  id = „01“
  OGCompensator = „Auskunft“
  pattern = „SendEmail“
  OGParameter = „(from,Auskunft),
  (to,Leiter),(subject,„Daten“),
  (body,„wurden herausgegeben“)“
  exec = „role(owner($Kontaktdaten)) ==
  KITStudent“
>>

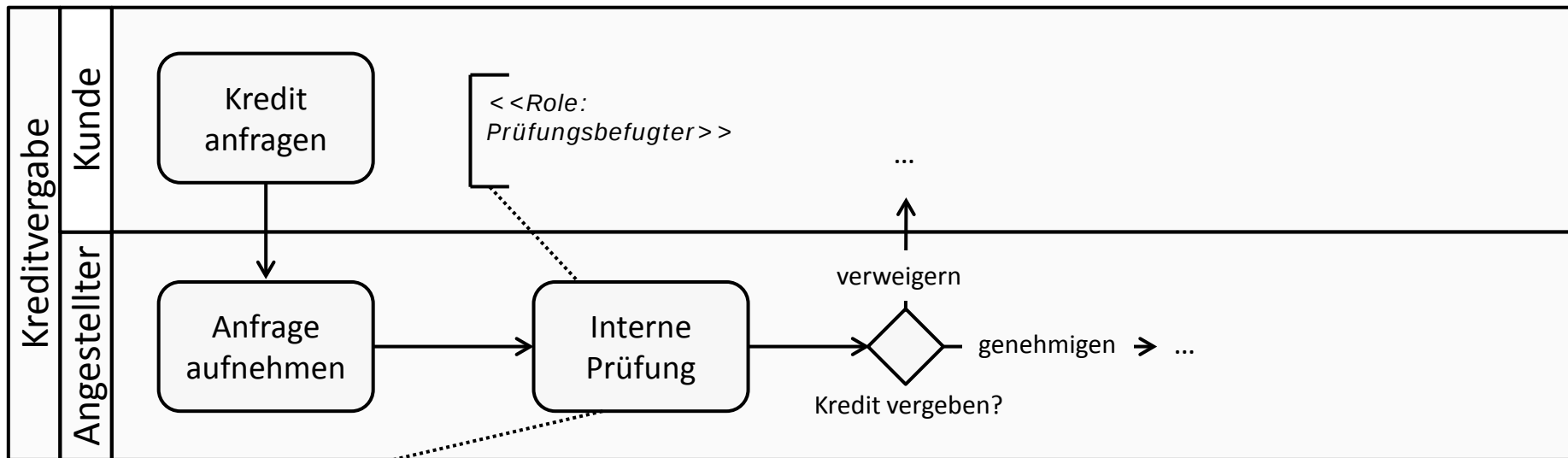
```

Aufgabe 7.3

Kreditvergabe einer Bank

- Ein Schritt im Prozessablauf ist die „Interne Prüfung“
 - Diese darf normalerweise nur von Angestellten mit der Rolle „Prüfungsbefugter“ durchgeführt werden, da nur diese auf die Kundendaten zugreifen dürfen
 - Ist der Kreditantrag allerdings mindestens 7 Tage lang in „Interne Prüfung“ (sprich es passiert nichts), so dürfen auch normale Bankangestellte die „Interne Prüfung“ durchführen und dabei vor allem auf die Kundendaten zugreifen
 - Als Konsequenz muss eine E-Mail an den Abteilungsleiter (Rolle: „Abteilungsleiter“) geschickt werden; dies macht das System automatisch
 - Im Text muss der Mitarbeiter stehen, der die Prüfung durchgeführt hat
 - Außerdem muss der Abteilungsleiter die Entscheidung zur Kreditvergabe, die vom normalen Angestellten durchgeführt wurde, im Nachgang überprüfen
-
- ❖ **Modellieren Sie eine entsprechende BTG-Annotation für das vorliegende Beispiel**
 - ❖ **Modellieren Sie das Versenden der E-Mail an den Abteilungsleiter ebenfalls aus**
 - ❖ **Die Überprüfung durch den Abteilungsleiter lässt sich mit dem bisher Gelernten nicht direkt modellieren**
 - ✓ Beschreiben Sie, wie Sie vorgehen würden
 - ✓ Notieren Sie die entsprechende Obligation gegebenenfalls





```
<<BTG:
BTGAccessor = „Angestellter“
objects = „$Kundendaten“
rights = „update“
start = „duration(„Interne Prüfung“).days
      >= 7“
obligations = „01,02“
>>
```

```
<<Obligation:
id = „01“
pattern = „SendEmail“
OGParameter = „(to,Abteilungsleiter),
(subject,$Kreditantrag), (body,„geprüft von
performer(„Interne Prüfung“)“)“
>>
```

```
<<Obligation:
id = „02“
OGCompensator = „Abteilungsleiter“
pattern = „EntscheidungPrüfen“
OGParameter = „(what,„Kredit vergeben“), (when,
14 days)“
>>
```

Für die Prüfung muss ein neues Obligation-Pattern erstellt werden. Dieses wird vom Prozessmodellierer als Prozessfragment in BPMN ausmodelliert und für die weitere Verwendung als Obligation definiert.
Beispielhaft hier: Fragment zur Überprüfung der Entscheidung an einem per Parameter „what“ spezifizierten Gateway binnen des per Parameter „when“ definierten Zeitraums.

AGENDA

1. Einführung und Motivation
2. BTG-Konzept
3. Rollenverteilung
4. BTG-Annotationen
5. Obligations
6. BPCC
7. Weitere Aufgaben
- 8. Fragebogen**

8. Fragebogen

- Danke für die Teilnahme an unserem Nutzer-Test!
- Damit wir die BTG-Annotationssprache für Business Processes noch weiter verbessern können möchten wir Sie bitten, zum Abschluss noch einen kurzen Fragebogen auszufüllen
 - Was hat Ihnen gut gefallen?
 - Was hat Sie gestört?
 - Auf welche Herausforderungen sind Sie gestoßen?
 - Haben Sie konkrete Verbesserungsvorschläge?

8. Fragebogen

- Bitte antworten Sie ehrlich und Ihrem Gefühl folgend!
- Beschönigungen helfen uns nicht bei der Fehlerfindung
- ...
- Die krampfhafteste Suche nach Fehlern allerdings ebenso wenig
- Teilen Sie uns einfach mit, was Sie denken!

8. Fragebogen

- Bitte am Computer ausfüllen
- Lieber zu viel als zu wenig in die Freitextanmerkungen schreiben 😊

<http://btg.grabatin.com/index.php/826811/lang-de>
bzw.

[http: / / goo.gl / 2qy65](http://goo.gl/2qy65)