1. Redaktionssystem: Anforderungen

Für einen Zeitschriftenverlag soll ein neues Redaktionssystem entwickelt werden, das die Prozesse bei der Heftproduktion abbildet und durch Automatismen unterstützt. Die Anforderungsanalyse wurde im Gespräch mit Redakteuren durchgeführt und wie folgt festgehalten:

- Das System kann verschiedene **Publikationen** unterstützen, die unterschiedlich oft pro Jahr (aber regelmäßig, z. B. 12x pro Jahr, 4x pro Jahr) erscheinen. Es soll möglich sein, neue Publikationen in das Heft aufzunehmen und bestehende zu deaktivieren (wenn eine Zeitschrift eingestellt wurde, also nicht mehr erscheint).
- Für jede **Publikation** sollen einzelne **Ausgaben** verwaltet (anlegen, ändern) werden. Mit jeder **Ausgabe** (z. B.: Nummer 12/2015) sind ein Titel (Thema des Schwerpunkts der Ausgabe) und eine Reihe von Terminen (Produktionsbeginn, Versand der Druckdaten, Erstverkaufstag und ggf. weitere) verbunden. Zu jeder **Publikation** lässt sich definieren, welche ihrer **Ausgaben** gerade produziert wird; dieser Wert wird nach Abschluss jeder Produktion geändert (zur nächsten Ausgabe).
- Zu jeder **Ausgabe** wird eine Liste von **Artikeln** verwaltet (die in dieser Ausgabe erscheinen sollen). Ein Artikel hat beliebig viele **Autoren** (0 oder mehr), einen *verantwortlichen* **Redakteur**, beliebig viele weitere Redakteure (0 oder mehr), eine Anzahl Seiten und eine Startseite. Er ist außerdem einer **Rubrik** zugeordnet. Für jeden Autor eines Artikels kann ein Honorar verwaltet werden.
- **Rubriken** sind für jede **Publikation** unterschiedlich. Es ist möglich, die Rubriken einer Publikation zu ändern. Wird eine Rubrik gelöscht, darf dies aber nicht dazu führen, dass bei älteren Ausgaben, die diese Rubrik noch verwendet haben, die **Artikel**-Einträge fehlerhaft oder unvollständig werden.
- **Artikel** lassen sich verschieben – in eine andere **Ausgabe** derselben **Publikation** oder in eine frei wählbare **Ausgabe** einer anderen **Publikation**. Auch mit **Artikeln** sind Termine verbunden: Bis wann soll der Artikel vom Autor/den Autoren geliefert werden, bis wann soll er vom verantwortlichen Redakteur bearbeitet werden?
- Artikel können **Standard-Artikel** sein; das Attribut „Standard“ legt fest, dass es in jeder Ausgabe eine Version dieses Artikels geben wird. (Diese Versionen müssen nicht identisch sein.)
- Zu jedem **Autor** werden Name, Anschrift, Bankverbindung und ein Steuer-Kennzeichen (umsatzsteuerpflichtig oder nicht?) gespeichert. Autoren können auch Mitarbeiter einer **Firma** sein.
- **Redakteure** sind **Autoren**, die zusätzlich ein Redakteurskürzel (z. B. hge) haben. Es ist möglich, dass eine Person gleichzeitig als Redakteur und als Autor eines Artikels auftritt.
- Jede Ausgabe einer Publikation hat außerdem einen **Chefredakteur**. (Der kann sich von Zeit zu Zeit ändern, darum wird er mit einer Ausgabe und nicht mit der Publikation verbunden.) Zu jeder Publikation wird auch der *aktuelle* Chefredakteur vermerkt. Beim Anlegen einer neuen Ausgabe wird der *aktuelle* Chefredakteur als Chefredakteur für diese Ausgabe eingetragen, diese Einstellung ist veränderbar.
- Eine **Firma** kann in verschiedenen Rollen relevant sein: Ein **Artikel** kann über ein Produkt einer **Firma** berichten, und im Umfeld eines Artikels kann eine Anzeige einer Firma platziert sein. Der Pressekodex verbietet, dass eine Anzeige der Firma X im Umfeld eines Artikels über ein Produkt der Firma X platziert wird. Der Begriff „Umfeld“ soll über einen Mindestseitenabstand definiert sein, der global im Redaktionssystem festgelegt wird.

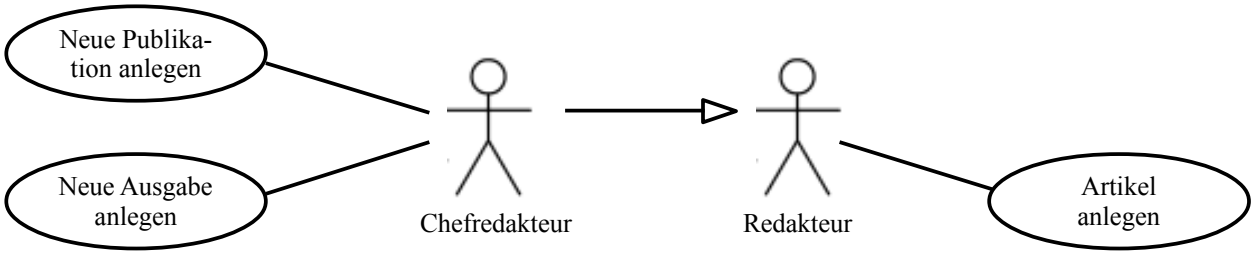
Bisher wurden im Wesentlichen die relevanten Datenstrukturen und teilweise deren Zusammenhänge sowie einzelne Aktionen beschrieben. Der Fokus lag aber auf den zu verwaltenden Daten. Ergänzend kommen nun weitere Anforderungen hinzu, die komplexe Aktionen beschreiben:

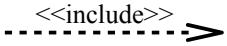
- Das System soll für einen **Autor** alle **Artikel** (mit Publikation, Ausgabe und Honorar) auflisten können, die dieser bisher geschrieben hat.

- Das System soll für einen **Redakteur** alle **Artikel** (mit Publikation und Ausgabe) auflisten können, die dieser bisher als verantwortlicher Redakteur (oder als beliebiger Redakteur; einstellbar) betreut hat.
- Chefredakteure haben vollen Zugriff auf das System: Sie können neue Publikationen anlegen und für bestehende Publikationen neue Ausgaben anlegen sowie diese auch bearbeiten. Außerdem können sie alles, was normale Redakteure können. (→ nächster Punkt)
- Redakteure haben eingeschränkten Zugriff auf das System: Sie können neue Artikel für jede Publikation und jede Ausgabe anlegen sowie bestehende Einträge verändern.
- Der Chefredakteur plant eine neue Ausgabe, indem er sie anlegt, Standard-Artikel aus einer älteren Ausgabe in die neue Ausgabe kopiert und neue Artikel anlegt.
- Autoren haben eingeschränkten Zugriff auf das System und können neue Artikel für jede Publikation und zukünftige Ausgaben anlegen – aber keine bestehenden Einträge ändern. Derart angelegte Artikel haben Vorschlagscharakter; der Chefredakteur der Ausgabe wird über den neu angelegten Eintrag informiert und kann den Artikel in einen „richtigen“ Artikel umwandeln – darüber wird der Autor dann informiert. Alternativ kann der Chefredakteur den Artikel ablehnen, damit wird der Eintrag gelöscht und ebenfalls der Autor informiert.
- Ein Autor kann einen (oder keinen) betreuenden Redakteur haben; die Betreuungsbeziehung kann sich ändern.
- Nach Abschluss der Produktion einer Ausgabe soll eine **Autorenabrechnung** für diese Ausgabe erstellt werden. Sie besteht aus separaten Anschreiben an jeden **Autor**, die alle **Artikel** des **Autors** in dieser **Ausgabe** mit den damit verbundenen Honoraren auflisten, am Ende steht außerdem die Gesamtsumme sowie ggf. Umsatzsteuer und Bruttosumme. Artikel, die mehrere Autoren haben, tauchen also mehrfach (in den Abrechnungen der beteiligten Autoren) auf. Aus buchhalterischen Gründen muss jede einzelne Abrechnung eine eindeutige, fortlaufende Abrechnungsnummer erhalten.

Zu diesem Szenario bearbeiten Sie nun *in Zweier- oder Dreiergruppen* die folgenden Aufgaben. Bei Unklarheiten können Sie diese mit dem vom Verlag als Teammitglied bereitgestellten *Experten* besprechen (das bin dann ich ;-)).

a) Erstellen Sie in UML **Use-Case-Diagramme** (Anwendungsfalldiagramme) für die sechs oben angegebenen Aktionen. Diese sind recht einfach aufgebaut, z. B.



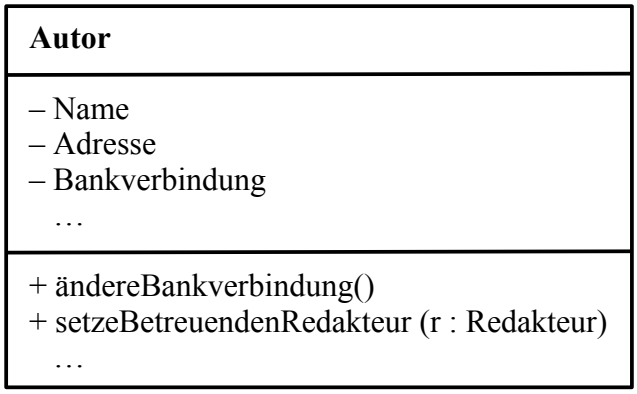
Die Use-Case-Diagramme sollen nur verdeutlichen, welche Aktoren an welchen Aktionen beteiligt sind. Eine der Aktionen (Ausgabe planen) ist zusammengesetzt; verwenden Sie für ihre Darstellung Include-Beziehungen: 

b) Die letzte Aktion (Autorenabrechnung) soll nun ausführlicher erklärt werden; erstellen Sie eine textuelle Beschreibung des Anwendungsfalls (vgl. Folien Fuhr/König, Nr. 185–187).

2. Klassen-Entwurf

a) Entwerfen Sie in UML ein Klassendiagramm für die Klassen **Publikation**, **Ausgabe**, **Artikel**, **Autor**, **Redakteur**, **Chefredakteur** und **Firma** und stellen Sie darin die Beziehungen zwischen den Klassen dar. Geben Sie außerdem für jede Klasse zwei Operationen (Methoden) an.

Versehen Sie Bezeichner mit Vorzeichen für die Sichtbarkeit, z. B.

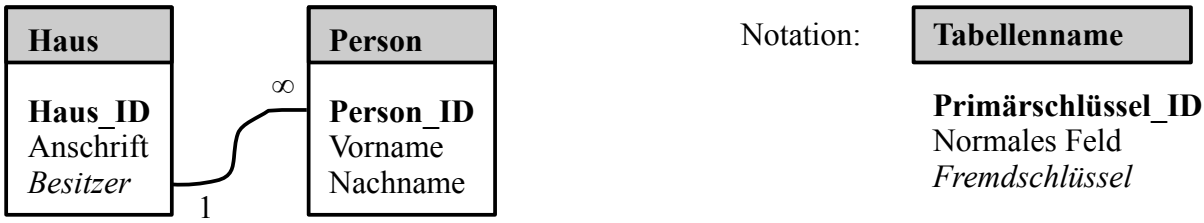


(Wir verwenden hier nur + und –, es gibt auch noch # = protected und ~ = package.)

b) Wie würden Sie hier die **Honorare** abbilden? Ist das eine eigene Klasse oder ein Attribut eines Artikels? Diskutieren Sie (in Ihrer Zweier-/Dreier-Gruppe) die verschiedenen Möglichkeiten.

3. Relationale Datenbank

Im letzten Semester haben Sie sich im Rahmen der Datenbanken-Vorlesung auch mit relationaler Datenbanken vertraut gemacht. Zeichnen Sie ein Datenbankmodell, das – auf Basis des bisherigen Entwurfs – zum Verwalten des Redaktionssystems geeignet ist. Legen Sie dabei besonderen Wert auf die Primär- und Fremdschlüssel, die in den Tabellen auftreten. Verwenden Sie die folgende Notation:



Wie setzen Sie in den Tabellen die Honorare für Artikel um?

Welche Zusammenhänge sehen Sie, wenn Sie das Klassendiagramm aus Aufgabe 2 und das Datenbankmodell aus dieser Aufgabe vergleichen?

Software-Installation („Hausaufgabe“)

Für den nächsten Veranstaltungstermin installieren Sie bitte das Programm ArgoUML, das Sie über die Projektwebseite http://argouml.tigris.org/ herunterladen können. Es sollte (als Java-Anwendung) auf allen gängigen Betriebssystemen lauffähig sein.

Wir werden beim nächsten Termin einige kleinere Übungen damit durchführen – u. a. ist ArgoUML in der Lage, aus einem Klassendiagramm Code (in Java, C++ und anderen Sprachen) zu generieren.