Klausur Summer 2012, Fragen

Einführung in die Softwaretechnik (IN0006) (Technische Universität München)

Prüfung 1 August 2012, Fragen

Einführung in die Softwaretechnik (IN0006) (Technische Universität München)

sebis

Fakultät für Informatik
Lehrstuhl für Informatik 19

Technische Universität München

Klausur zur Vorlesung „Einführung in die Softwaretechnik“

Lehrstuhl für Informatik 19 (sebis), 01. August 2012, Sommersemester 2012

Nachname:	
Vorname:	
Matrikelnummer:	
Studiengang:	

Wichtige Hinweise

- Füllen Sie auf diesem Blatt gut lesbar die obigen Felder (Nachname, Vorname etc.) aus.
- Notieren Sie auf jeder Seite des Arbeitspapiers gut lesbar Ihren Namen und die laufende Seitennummer!
- Es stehen 120 Minuten Bearbeitungszeit zur Verfügung. Maximal zu erreichen sind 130 Punkte.
- Beantworten Sie Fragen, wenn nicht anders angegeben, direkt auf dem Angabenblatt. Sollte der Platz dort nicht reichen, antworten Sie auf den karierten Bögen. Fügen Sie aber unbedingt auf dem Angabenblatt einen entsprechenden Hinweis ein!

Erlaubte Hilfsmittel

- Ein beidseitig handbeschriebenes DIN A5 Blatt.

Aufgaben

Teil 1: Wissensfragen

1. **Requirements Engineering (12 Punkte)**
   Ihr Unternehmen soll einen Online-Shop für eine Buchhandlung entwickeln. Zunächst sollen Sie deshalb die Anforderungen der verschiedenen Stakeholder erfassen.

   a) Mit welchen drei in der Vorlesung vorgestellten Arten von Anforderungen müssen Sie rechnen? (3 Punkte)

   1) _____

   2) _____

   3) _____

   b) Geben Sie jeweils ein konkretes, auf den Online-Shop bezogenes Beispiel für solch eine Anforderung. (3 Punkte)
   Hinweis: „Die Software muss hohe Qualität haben" ist zum Beispiel nicht konkret genug.

   Beispiel zu 1) _____

   _____

   Beispiel zu 2) _____

   _____

   Beispiel zu 3) _____

   _____

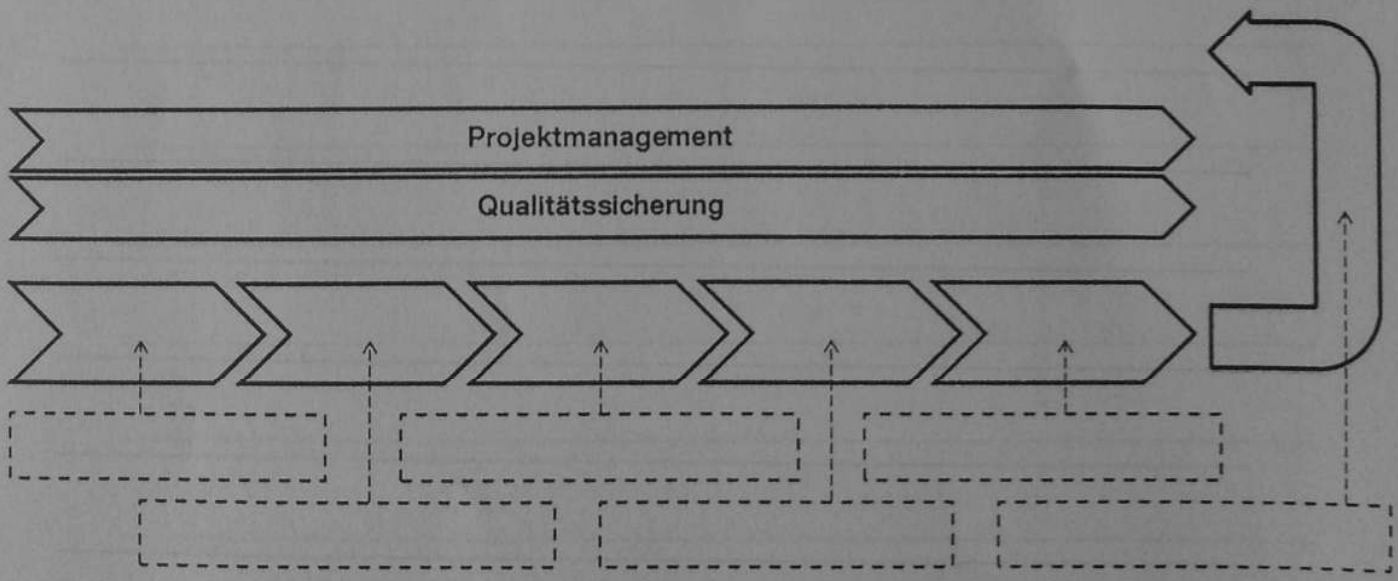
c) Grenzen Sie in jeweils drei Stichpunkten die Begriffe Pflichtenheft und Lastenheft voneinander ab. Berücksichtigen Sie dabei jeweils, wer das Dokument erstellt, wie detailliert es ist und welche Rolle es im Softwareentwicklungsprozess spielt. (6 Punkte)

Pflichtenheft:

Lastenheft:

2. **Softwareentwicklungsprozess (6 Punkte).**

Ergänzen Sie in folgender Abbildung die Namen der einzelnen Phasen des Softwareentwicklungsprozesses wie in der Vorlesung vorgestellt.



3. **Cloud computing (6 Punkte)**
Grenzen Sie die Begriffe „private cloud" und „public cloud" voneinander ab. Gehen Sie dabei auch auf die jeweils möglichen Abgrenzungen von User-Daten auf Hardware-Ebene ein.

Private Cloud:

Public Cloud:

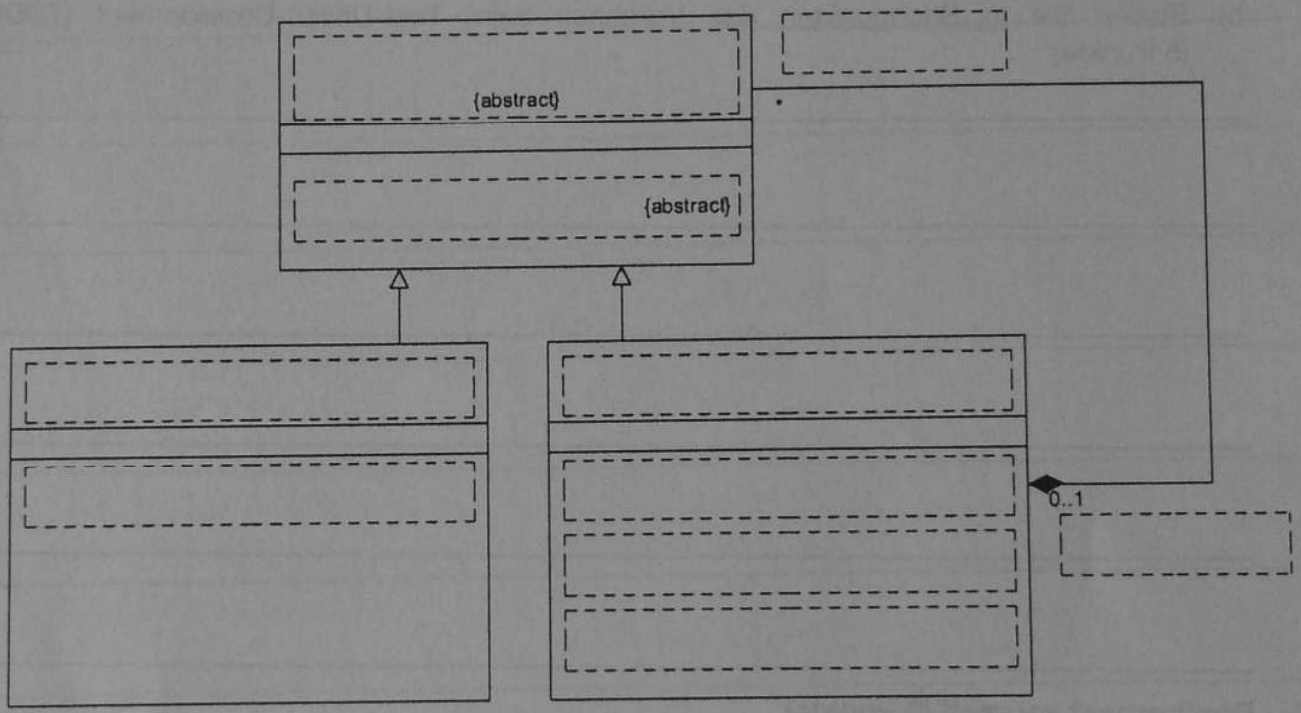
4. **Refactoring (4 Punkte)**

Was versteht man unter „Refactoring" und wofür wird es eingesetzt?

5. Entwurfsmuster (12 Punkte)

a) Welches Entwurfsmuster ist schematisch in folgendem Diagramm dargestellt? Ergänzen Sie die fehlenden Beschriftungen. (8 Punkte)

Name des Musters: _____



b) Beschreiben Sie kurz, was der Zweck des Musters aus Teilaufgabe a) ist. (4 Punkte)

6. Test-Driven-Development (7 Punkte)

a) Was ist das charakteristische Merkmal von Test-Driven-Development (TDD)? (2 Punkte)

b) Stellen Sie in Stichpunkten das Vorgehen beim Test-Driven-Development (TDD) dar. (5 Punkte)

7. Don't repeat yourself (6 Punkte)

a) Erläutern Sie das Prinzip „Don't repeat yourself" (DRY). (2 Punkte)

b) Nennen Sie in diesem Kontext die 4 Arten von Dopplung. (4 Punkte)

1.

2.

3.

4.

8. Scrum (7 Punkte)

a) Worin unterscheidet sich Scrum grundsätzlich vom Wasserfallmodell, dem klassischen Vorgehensmodell bei der Softwareentwicklung? (3 Punkte)

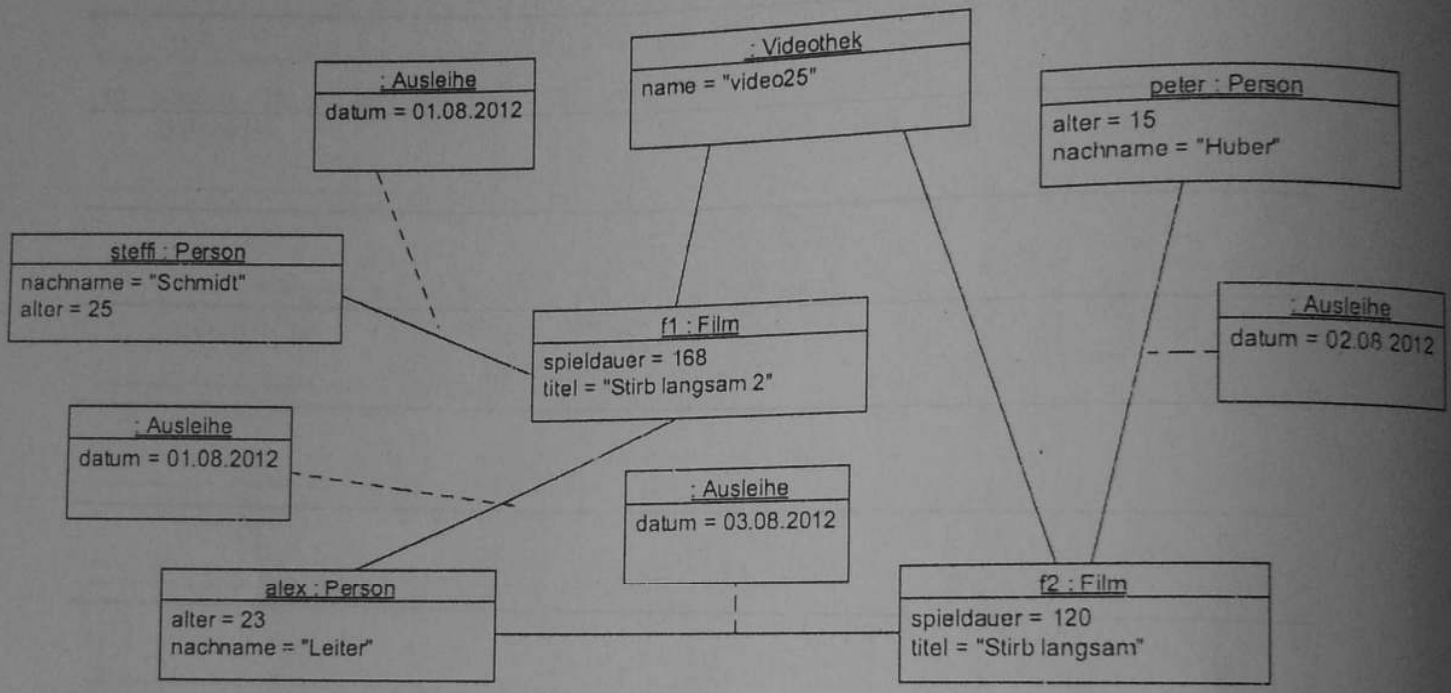
b) Welche Funktion hat der Scrum Master? (2 Punkte)

c) Nennen Sie die restlichen in der Vorlesung vorgestellten Scrum-Rollen. (2 Punkte)

Teil 2: Verständnis- und Modellierungsfragen

9. Objekt- und Klassendiagramme (12 Punkte)

Gegeben ist folgendes Objektdiagramm:



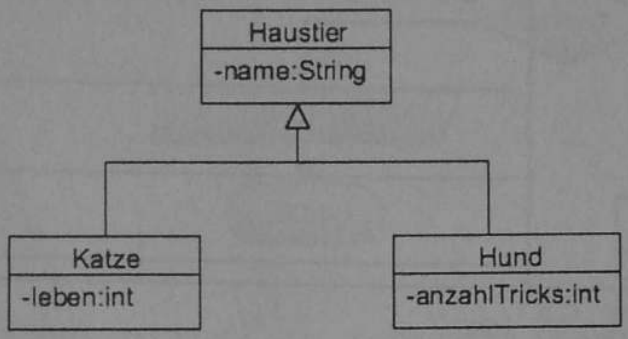
Erstellen Sie zu diesem Diagramm ein entsprechendes konzeptuelles UML 2.0 Klassendiagramm, das alle verwendeten Klassen, Beziehungen und Attribute enthält.

Hinweise:

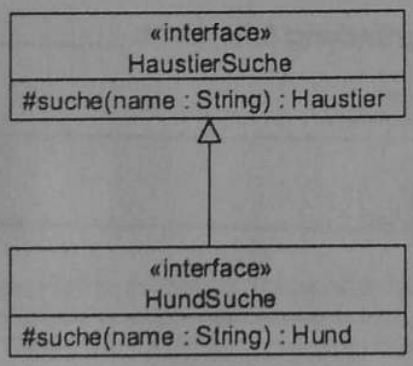
- *Die Multiplizitäten an Assoziationen können nicht immer eindeutig bestimmt werden. Treffen Sie in solchen Fällen eine sinnvolle Wahl und dokumentieren Sie diese in einem Satz.*
- *Sie können zusätzlich zu den Datentypen `int`, `double`, `boolean` und `String` auch noch die Datentypen `Currency` und `Date` verwenden.*
- *Bitte lösen Sie diese Aufgabe auf einem der ausgeteilten karierten Papierbögen.*

10. Syntaktische Kompatibilität (8 Punkte)

Gegeben seien folgende Vererbungsbeziehungen der Klassen *Haustier*, *Hund* und *Katze*. Entscheiden Sie für die vier darauffolgenden Interfacebeziehungen jeweils, ob die Operation im spezielleren Interface (Sub-Interface) syntaktisch kompatibel zur Operation im allgemeineren Interface (Super-Interface) ist. Begründen Sie ihre Antwort nur, falls die Operation nicht kompatibel ist.



a) Haustiersuche



Syntaktisch kompatibel? _____

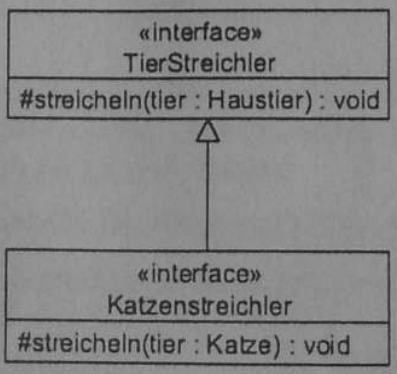
Begründung falls nicht: _____

_____

_____

_____

b) Tierstreichler



Syntaktisch kompatibel? _____

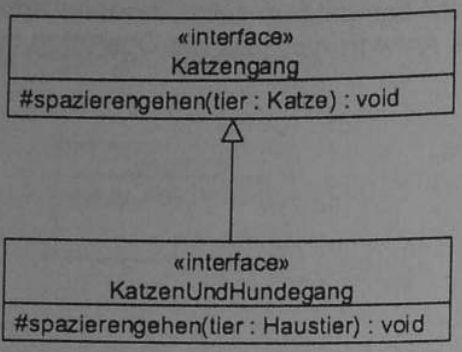
Begründung falls nicht: _____

_____

_____

_____

c) Spazieren gehen



Syntaktisch kompatibel? _____

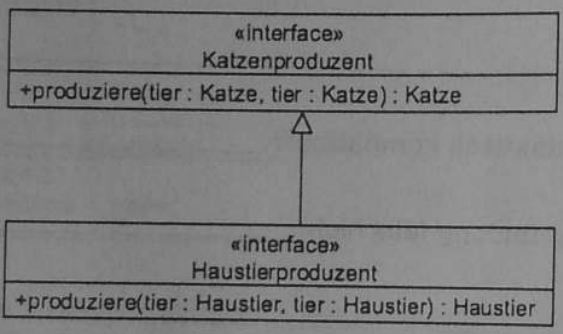
Begründung falls nicht: _____

_____

_____

_____

d) Produzenten



Syntaktisch kompatibel? _____

Begründung falls nicht: _____

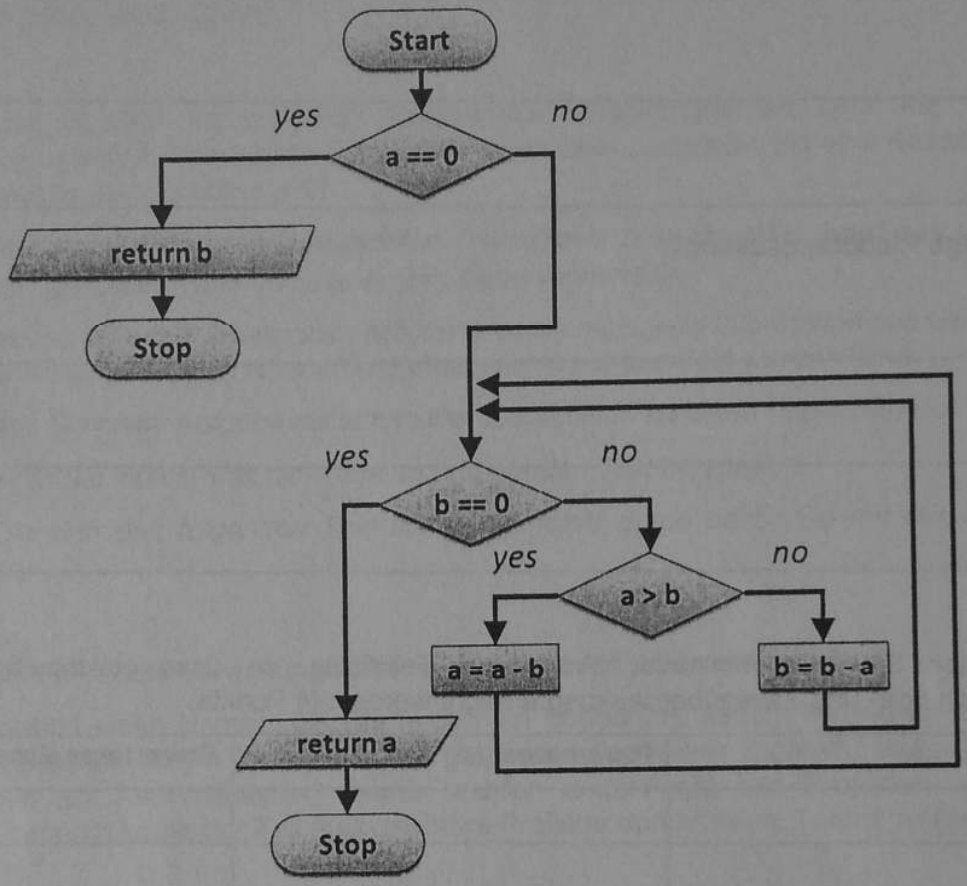
_____

_____

_____

11. Testen (14 Punkte)

Der folgende Graph beschreibt den Kontrollfluss einer Methode, die den größten gemeinsamen Teiler zweier ganzzahliger Eingaben **a** und **b** berechnet (Euklidischer Algorithmus):



Gegeben ist außerdem folgende Menge von Testfällen:

Parameter „a“	Parameter „b“	Erwartetes Ergebnis
0	7	7
0	0	0
5	10	5
1	1234	1

a) Entscheiden Sie, ob für diese Menge von Testfällen jeweils vollständige Anweisungsüberdeckung, Zweigüberdeckung und Pfadüberdeckung erreicht wird und begründen Sie Ihre Antwort. (7 Punkte)

Hinweis: Die Frage bezieht sich auf die gesamte Menge, nicht auf die einzelnen Testfälle.

Vollständige Anweisungsüberdeckung?

Vollständige Zweigüberdeckung?

____

____

____

Vollständige Pfadüberdeckung?

____

____

____

b) Erstellen Sie eine minimale Menge von Testfällen, so dass gleichzeitig vollständige Anweisungs- und Zweigüberdeckung erreicht werden: (4 Punkte)

Parameter „a“	**Parameter „b“**	**Erwartetes Ergebnis**

c) Handelt es sich bei den neu erstellten Testfällen aus Teilaufgabe b) um Blackbox oder Whitebox-Tests? Begründen Sie Ihre Antwort. (3 Punkte)

____

____

____

____

12. Konzeptuelle Modellierung (36 Punkte)

Erstellen Sie ein konzeptuelles UML 2.0 Klassendiagramm für die Projektmanagementsoftware EIST-Project. Modellieren Sie dabei die für diese Anwendung relevanten Konzepte aus folgender Beschreibung so genau wie möglich.

Hinweise:

- *Geben Sie zu allen Assoziationen ausdrücklich Multiplizitäten an, auch ein * muss also angegeben werden. Verwenden Sie Rollennamen oder benennen Sie eine Assoziation, wenn die Bedeutung nicht eindeutig ist.*
- *Sie können zusätzlich zu den bekannten Datentypen `int`, `double`, `boolean` und `String` auch noch die Datentypen `Currency` und `Date` verwenden.*
- *Beachten Sie bitte außerdem, dass möglicherweise nicht jede Information aus der Angabe auf sinnvolle Art und Weise in einem Klassendiagramm ausgedrückt werden kann.*
- *Bitte lösen Sie diese Aufgabe auf einem der ausgeteilten karierten Papierbögen.*
- *Modellieren Sie nichts, was nicht aus dem folgenden Text hervorgeht.*
- *Lesen Sie sich den folgenden Text einmal komplett durch bevor Sie mit der Modellierung beginnen.*

Das Softwaresystem soll die Verwaltung von Projekten und Aufgaben ermöglichen.

Jedes Projekt besitzt einen Namen, ein Start- und ein Enddatum. Jedem Projekt ist außerdem ein Projektteam zugeordnet. Einem Projektteam ist genau ein Teamleiter zugeteilt, sowie beliebig viele weitere Personen als Teammitglieder. Teams werden unabhängig von Projekten verwaltet und dasselbe Team kann im Laufe der Zeit auch mehrere Projekte durchführen. Teams werden über einen Namen identifiziert.

Für Personen soll jeweils Vorname und Nachname, sowie das Geburtsdatum und das Jahresgehalt erfasst werden. Außerdem hat jede Person eine Personalnummer wie zum Beispiel „007".

Im System verwaltete Aufgaben haben jeweils eine Beschreibung und ein Fälligkeitsdatum. Außerdem wird der Bearbeitungsstatus einer Aufgabe erfasst: Er ist entweder „unbearbeitet", „in Bearbeitung", „abgeschlossen" oder „abgelehnt". Eine Aufgabe kann einer Person zugeordnet werden, dies ist allerdings nicht zwingend nötig.

Zu einem Projekt kann eine Menge von Meilensteinen festgelegt werden. Jeder Meilenstein hat ein Datum und eine Nummer. Die Nummern für Meilensteine werden fortlaufend vergeben und sind wichtig, um die Reihenfolge zu jedem Zeitpunkt eindeutig festzulegen (das Datum einzelner Meilensteine kann im Laufe des Projekts verschoben werden, dies muss allerdings nicht modelliert werden). Schließlich kann ein Meilenstein von Aufgaben abhängen, die bis zum Erreichen des Meilensteins erledigt sein müssen. Eine Aufgabe kann dabei maximal einem Meilenstein zugeordnet sein.

Es gibt zwei besondere Arten von Projekten, für die spezielle Kennzahlen erhoben werden: Für Softwareentwicklungsprojekte kann die Menge an verbrauchtem Kaffee in ganzen Litern angegeben werden, für Beratungsprojekte die Zahl der erstellten Powerpoint-Folien.

Schließlich gibt es noch Beziehungen zwischen Projekten: Einem Projekt kann ein Vorgängerprojekt zugeordnet sein. Entsprechend hat ein Projekt möglicherweise auch Folgeprojekte. Dabei ist davon auszugehen, dass aus einem einzelnen Projekt zwar mehrere unabhängige Folgeprojekte hervorgehen können aber für ein Projekt höchstens ein Vorgängerprojekt angegeben werden kann.