

1. Problembeschreibung:

Aufgabenstellung dieses Projekts ist es, ein Planungstool für die Koordinierung aller laufenden Kundenprojekte eines Unternehmens zu erstellen. Als realer Hintergrund wird ein Unternehmen, das Softwareprojekte, z.B. die Erstellung von Internetseiten für Kunden, abwickelt, angenommen. Bei diesen und ähnlichen Projekten kommt es oftmals zu Zeitverzögerungen einerseits mangels realistischer Abschätzung des Umfangs andererseits aufgrund schlechter Planung und Koordination der Ressourcen. Die Folge sind häufig Zahlungsforderungen einer Pönale seitens des Kunden.

Mit Hilfe des zu entwickelnden Tools sollen die einzelnen Projekte je nach Ressourcen (interne und externe Mitarbeiter) optimal eingeplant werden können. Die Arbeitszeit der einzelnen Mitarbeiter, die zur Planung zur Verfügung steht, muss zu Beginn unter Berücksichtigung des Wochenarbeitspensums (Voll- bzw. Teilzeit), Urlauben u.ä. eingegeben werden. Diese Daten müssen von den einzelnen Mitarbeitern laufend aktualisiert werden. D.h. jeder Mitarbeiter gibt seine Arbeitsstunden je Projekt, an dem er gerade mitarbeitet, regelmäßig, am besten täglich, ein. Weiters wird festgelegt welche Tätigkeiten, z.B. Grafik, Texten, etc. wie viel Zeit (Stundenumfang im Durchschnitt) in Anspruch nehmen.

Liegt nun ein Projekt vor, werden zuerst der genaue Umfang (Anzahl der Projektphasen) und gewünschte Zeitpunkt der Fertigstellung eingegeben. Dann wählt der Projektmanager die Mitarbeiter (Grafiker, Texter, etc.) aus, die an dem Projekt mitarbeiten sollen. Das Programm sollte dann einen Zeitplan erstellen, die notwendigen Ressourcen dazu errechnen und die Verfügbarkeit der einzelnen Mitarbeiter überprüfen. Kommen weitere Projekte hinzu, sollte eine Optimierung im Ablauf der einzelnen Projekte, d.h. der erforderlichen Tätigkeiten durchgeführt werden. Kommt es zu Engpässen können externe Mitarbeiter – sofern verfügbar - herangezogen werden oder es werden Minuspunkte für Zeitüberschreitung vergeben.

Am Ende sollte unter Berücksichtigung aller zur Verfügung stehenden Ressourcen und Möglichkeiten und unter Vermeidung von Leerläufen ein optimaler Projektplan entstehen.

2. Analyse und Modellierung

Use-Case Diagramm

Use-Case-Name: Projektplanungstool

Akteure: PM, Mitarbeiter

Vorbedingung: Auftrag für ein neues Projekt vom Kunden liegt vor

Nachbedingung: Projekt im Zeitrahmen abgeschlossen

Auslöser: PM legt neues Projekt an

Ablaufbeschreibung:

- PM gibt Umfang (Grafikelemente, Textseiten, Bilder) und gewünschtes Datum der Fertigstellung ein.
- PM wählt Mitarbeiter (Grafiker, Texter, ...), die benötigt werden, aus.

- Verfügbarkeit der Mitarbeiter wird überprüft (Urlaub, engagiert für anderes Projekt)
- Überschneidungen mit anderen Projekten werden überprüft
- Zeitplanerstellung unter Berücksichtigung des Zieldatums

Fehlsituationen:

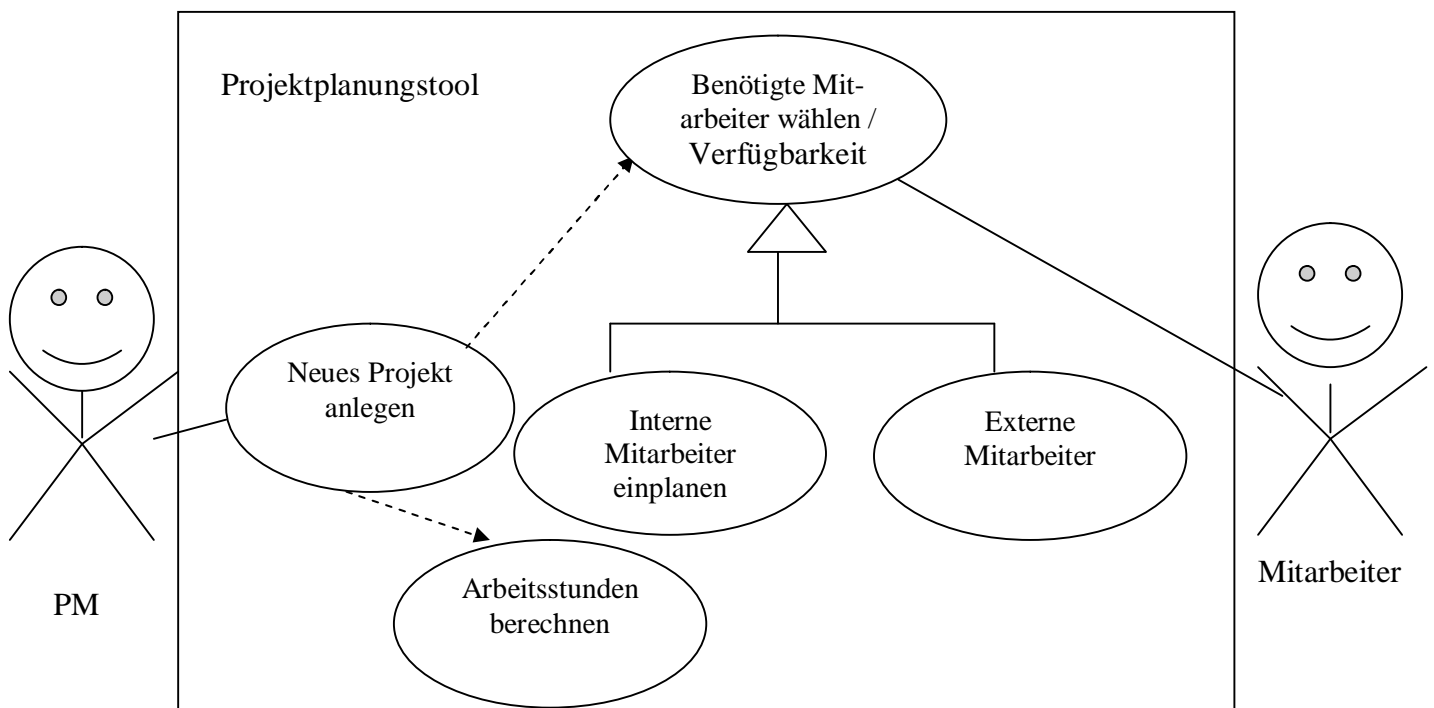
- Zeitrahmen wird überschritten (Pönaleforderung seitens des Kunden)
- Kollision mit anderen Projekten
- Nachträgliche Änderung des Projektumfangs
- Zeitliche Verschiebung seitens des Kunden

Variationen:

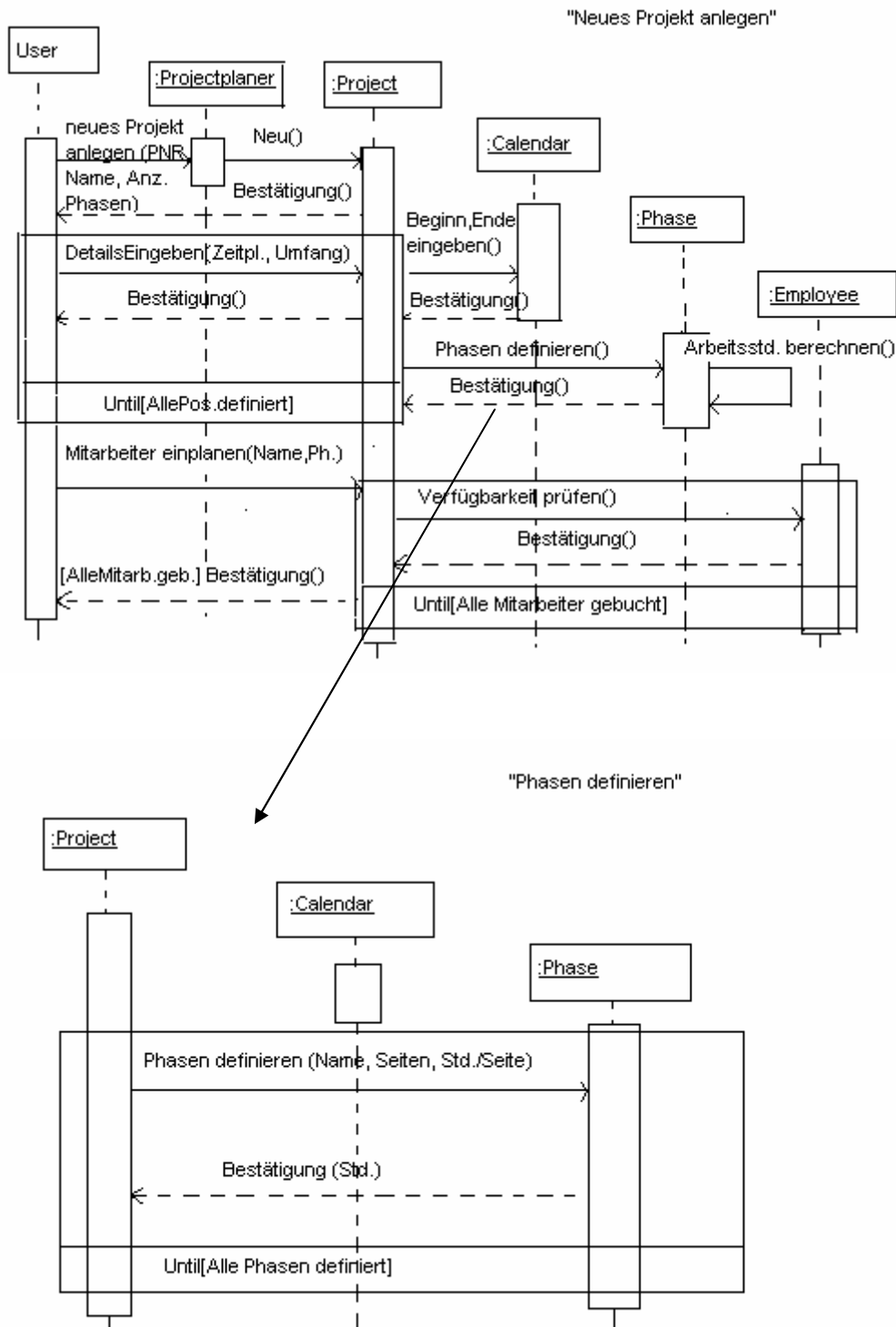
- Externe Mitarbeiter heranziehen
- Projekt früher als geplant fertig

Ergebnisse: Projekt erfolgreich und rechtzeitig abgeschlossen

Nicht-funktionale Anforderungen: Kunde zufrieden, keine Leerläufe, keine Überstunden, keine Pönalezahlungen



Sequenzdiagramm



3. Klassendefinition

Klasse „Projectplaner“

Diese Klasse - in dieser Version noch nicht vollständig programmiert – ist dazu gedacht, die einzelnen, laufenden Projekte untereinander nach Zieldatum und Mitarbeiterverfügbarkeit zu koordinieren. Neue Projekte sollten über diese Klasse angelegt werden (Vector projects) und es wäre dann möglich einen Projektübersichtsplan sowie eine Projektliste aller Projekte zu erstellen. Durch eine Verknüpfung mit dem Kalender könnten Überschneidungen mit anderen Projekten geprüft werden, um so Umschichtungen oder zeitlichen Verschiebungen vorzunehmen.

Klasse „Project“

Variablen

- Int pNR: Projektnummer
- String name: Projektbezeichnung
⇒ Eingabe erfolgt über Konstruktor
- Phase [] phases / int number: Das Projekt besteht aus mehreren Phasen, wie Design, Texteingabe, Bildeinbau, usw., die durch ein Array der Anzahl „number“ von der Klasse „Phase“ definiert sind.
- Calendar calendarBegin / Calendar calendarEnd: Verknüpfung zur vordefinierten Klasse „GregorianCalendar“ (Java API) mit aktuellem Datum

Methoden

- public final void setBegin: hier kann das Beginndatum des Projektes - wenn abweichend von aktuellem Tagesdatum – eingegeben werden
- public final void setEnd: mit dieser Methode wird das Zieldatum für das Projektende festgesetzt

Über die beiden folgenden Methoden können die einzelnen Mitarbeiter zum Projekt ausgewählt und eingeplant werden:

- public void addEmployee: diese Methode ruft “public void addEmp” in der Klasse “Phase” auf und fügt den ausgewählten Mitarbeiter zur jeweiligen Phase hinzu
- public void addEmployeeExtern: ruft „public void addEmpX“ in der Klasse „Phase“ auf und plant bei Bedarf und Verfügbarkeit einen externen Mitarbeiter für die Projektbearbeitung ein
- public double getWorkDays: Ausgabe der benötigten Arbeitstage pro Projekt, indem die Methode „calculateDays“ , definiert in der Klasse „Phase“, über alle Phasen abgerufen wird

Klasse “Phase”

Variablen

- String name
- int page: bezeichnet die Anzahl der Internet- oder Textseiten bzw. der Bilder in den einzelnen Phasen
- double hours: benötigte Arbeitsstunden pro Seite
- double totHours: Gesamtstunden (Seite x Stunden) pro Phase
- Vector employeeInt / Vector employeeExt: Mitarbeiterlisten, intern und extern

Methoden

- public void addEmp: Mitarbeiter werden je Phase gebucht, wird über Klasse „Project“ aufgerufen
- public void addEmpX: Externer Mitarbeiter wird herangezogen, Aufruf ebenso über Klasse „Project“
- public double calculateHours: berechnet die Gesamtarbeitszeit pro Phase in Std. (= totHours)
- public double addTime: mit Hilfe dieser Methode kann zur Gesamtarbeitszeit pro Phase ein Zeitpuffer addiert werden
- public double calculateDays: rechnet die Gesamtarbeitszeit in Std. in Arbeitstage um

Klasse „Employee“

In dieser Klasse werden die Daten der Mitarbeiter hinterlegt. In dieser Version noch nicht programmiert aber vorgesehen wäre ein Kalender pro Mitarbeiter mit dem ein Wochen- bzw. Monatsarbeitsplan erstellt werden kann. Hier würden dann Arbeitsstunden je Projekt eingetragen werden ebenso wie Urlaubs- und Krankentage. Über die Klasse „Projectplaner“ könnten dann die einzelnen Mitarbeiter je nach Verfügbarkeit für die laufenden Projekte eingeplant werden. Bei Engpässen können externe Mitarbeiter herangezogen werden bzw. Projekte je nach Dringlichkeit zeitlich nach hinten verschoben werden.

Variablen

- String name
- String sVNR: Sozialversicherungsnummer
- double salary
- double workHours: Wochenarbeitszeit in Stunden
- double freeTime: nicht verplante Arbeitszeit
⇒ die Arbeitsstunden sollten von jedem Mitarbeiter täglich eingetragen und aktualisiert werden

Die **Klassen „Assistant“, „Copywriter“ und „Designer“** stellen eine Generalisierung zur **Klasse „Employee“** und werden mit „extends“ umgesetzt.

Klasse „EmployeeExt“

Variablen

double hours: gibt den Stundenumfang an, den ein externen Mitarbeiter zur Verfügung steht
double costs/h: Kosten pro Stunde

4. Installation und Wartung

Um das Programm für ein Projekt laufen lassen zu können, müssen die folgenden Klassen installiert und kompiliert werden: Klasse „Project, Phase, Employee, EmployeeExt, Assistant, Copywriter, Designer“ und das File „UseProject“. Letzteres startet das Programm.

Für einen einwandfreien Betrieb des Projektplanungstools ist es unerlässlich, dass alle Beteiligten (Projektmanager, Mitarbeiter) die erforderlichen Daten, v.a. Arbeitszeiten und –stunden täglich eintragen und laufend aktualisieren.

Klassendiagramm

Anmerkung:
grau markierte Klassen
Variablen und Methoden
sind nicht programmiert

