Universität Zürich[UZH]
Institut für Informatik

s.e.a.l. software evolution & architecture lab
RERG

Software Engineering Übung 5

Verträge, Aufwand- und Risikoschätzung

1 Informationen

1.1 Daten

- Ausgabe Di 23.11.2010
- Abgabe So 05.12.2010 bis 23:59 Uhr
- Besprechung am Di 14.12.2010 um 11:50 Uhr

1.2 Formales

Die Lösungen sollen als PDF Datei mit dem Namen **Ex[*n*]_[*NameA_NameB_NameC*].pdf** abgegeben werden, wobei [*n*] die Nummer der Übung ist und [*NameA_NameB_NameC*] die Nachnamen der Gruppenmitglieder sind. Die PDF Datei sollte ausserdem ebenfalls Ihre Namen und Matrikelnummern beinhalten.

Mailen sie Ihre Lösungen vor dem Abgabetermin an `charrada@ifi.uzh.ch` und `wueest@ifi.uzh.ch`. Der Betreff der E-mail sollte mit **[SE EX HS10]** beginnen. Falls Sie zusätzliche Abgabematerialien (z.B. Source Code) haben, mailen Sie bitte ein Archiv (.zip-File), welches alle Dateien, einschliesslich dem PDF, enthält. Benennen sie das Archiv anhand der oben erwähnten Konventionen.

Die Übungen sollen in 3er Gruppen gelöst werden. Jedes Gruppenmitglied muss über alle Teile der Lösungen Auskunft geben können. Verspätete Abgaben werden korrigiert, aber nicht bewertet.

2 Aufgabenstellung

Die Aufgaben beziehen sich auf die Fallstudie aus den Übungen 2 bis 4.

2.1 Vertragsformulierung (4 Punkte)

Nachfolgend finden Sie den Code eines Programmes zum Lösen quadratischer Gleichungen. Das Programm zeigt Informationen zur gewählten Gleichung an und berechnet, falls vorhanden, die Nullstellen der Gleichung. Die Anzahl Nullstellen lässt sich aus der Diskriminanten ablesen. Ist die Diskriminante negativ, so besitzt die Gleichung keine Nullstellen. Ist sie gleich 0, so gibt es genau eine Nullstelle. Ist sie grösser als 0, so hat die Gleichung genau zwei Nullstellen. Die Diskriminante berechnet sich folgendermassen, gegeben die Gleichungskoeffizienten a, b und c: $Diskriminante = b^2 - 4 * a * c$

Hinweis: Im Nachfolgenden bedeutet { ... }, dass die betreffende Methode eine Implementierung hat, diese aber hier nicht gezeigt ist.

Fügen Sie die fehlenden Verträge (jeweils Pre- und Postconditions) zur spezifizierten Schnittstelle hinzu. Für die bessere Verständlichkeit soll bei der Formulierung von Gleichheit in Ausdrücken von Verträgen das doppelte Gleichheitszeichen (==) verwendet werden. Verwenden Sie ansonsten die Notation, wie Sie diese aus der Vorlesung kennen (Folienkapitel 5.6). Sie müssen ausser den Verträgen keine weitere Implementierung vornehmen.

```
/**
 * EquationSolver is meant to solve quadratic equations.
 * It calculates the discriminant and the root(s) of the equation.
 * A quadratic equation is represented by the following expression:
 * A.x^2+B.x+C=0
 * where x represents a variable and A, B and C are
 * constants (which we are called "coefficients").
 *
 */
public class QuadraticEquationSolver {

// Coefficient A of the equation
private int coefficientA;
// Coefficient B of the equation
private int coefficientB;
// Coefficient C of the equation
private int coefficientC;


/**
 * Calculate the discriminant of the equation
 *
 * @pre
 * @post
 *
 * @return the discriminant of the equation
 */
public double discriminant(){...}

/**
 * Calculates the value of the polynomial (A.x^2+B.x+C) for a given x
 * @param x
 * @pre
```

```
 * @post
 *
 * @return the value of the polynomial in x
 */
public double evaluateP(double x){...}

/**
 * calculates the number of roots the polynomial has
 *
 * @pre
 * @post
 *
 * @return the number of roots
 */
public int rootsNumber(){...}

/**
 * Calculates the first root of the polynomial
 * it is used only when the equation has 2 roots
 *
 * @pre
 * @post
 *
 * @return the first root of the polynomial
 */
private double root1(){...}

/**
 * Calculates the second root of the polynomial
 * it is used only when the equation has 2 roots
 *
 * @pre
 * @post
 *
 * @return the second root of the polynomial
 */

private double root2(){...}

/**
 * Calculates the root of the polynomial
 * it is used only when the equation has a unique root
 *
 * @pre
 * @post
 *
 * @return the unique root of the polynomial
 */
private double uniqueRoot(){...}
}
```

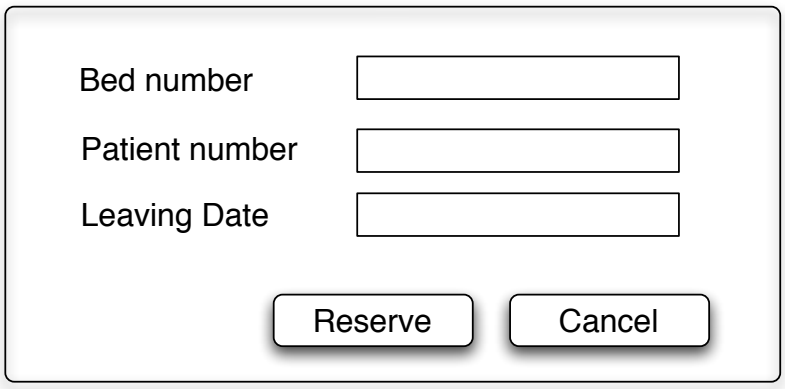

Abbildung 1: Oberflächenprototyp für die Bettenreservation

2.2 Aufwandschätzung, Function Points (6 Punkte)

Gegeben sei der Oberflächenprototyp für das Bettenreservation-System des Heilenspitals (siehe Abb. 1).

Die folgenden zusätzlichen Informationen zu dieser Eingabemaske sind gegeben:

- Alle Felder sind zu Beginn leer.
- Wenn eine Bettenreservation erfolgt, überprüft das System, ob das Bett noch verfügbar ist.

a) Ermitteln Sie die Adjusted Function Points gemäss der Vorgehensweise aus der Vorlesung. Beschreiben Sie dabei Ihre Vorgehensweise und dokumentieren Sie die Annahmen, welche Sie treffen. Die Berechnung der Function Points für den Oberflächenprototyp betrifft nur Dateneingaben und Anfragen, nicht die Datenausgaben. Nehmen Sie für die Anfragen, für die externen Schnittstellen und die internen Datenbestände jeweils, falls vorhanden, einen einfachen Schwierigkeitsgrad an. Für die Gewichtung der Eingaben können die Werte der Tabelle auf Folie 22, Kapitel 16 aus der Vorlesung analog verwendet werden. Bei der Berechnung des Gesamt-Einflussfaktors werden die Effizienz der Benutzerschnittstelle, die Wiederverwendbarkeit, die einfache Benutzbarkeit und die Erweiterbarkeit als sehr hoch bewertet. Komplexe Verarbeitungen und Installationen an mehreren Orten haben keinen Einfluss. Für die restlichen Faktoren wird ein durchschnittlicher Einfluss angenommen. Falls Sie noch weitere Annahmen treffen, dokumentieren und begründen Sie diese.

b) Was brauchen Sie neben den Adjusted Function Points noch, um eine Schätzung des Aufwandes für den Teil unter a) zu machen? Begründen Sie die Aussagen kurz.

2.3 Aufwandschätzung, COCOMO2 (6 Punkte)

Die Firma IT4U will sich für den Auftrag, das System für das *Heilenspital* zu entwickeln, bewerben. Bevor man ein Angebot macht, wird intern eine Aufwandschätzung vorgenommen. Verschiedene Möglichkeiten werden verglichen und ihre Auswirkungen auf den Aufwand werden berechnet. Als algorithmisches Schätzverfahren wird COCOMO2 verwendet.

a) Das Projektteam besteht aus Mitarbeitern, welche bereits gemeinsame Projekte bearbeitet haben. Projekte, die dem *Heilenspital* Projekt ähnlich sind, gab es allerdings noch keine. Für den Posten des Projektleiters gibt es zwei Bewerbungen. Die erste Bewerbung ist von einem Mitarbeiter, der schon lange bei IT4U arbeitet. Er macht sich grosse Hoffnungen, den Job zu erhalten, denn er möchte Erfahrungen im Leiten von Projekten sammeln. Falls er den Posten nicht bekommt, wird er als Teammitglied beim Projekt dabei sein.

Die andere Bewerbung kommt von einer Mitarbeiterin von HospitalSystemsIT, einer Firma welche sich zum Teil auf Software im Spital-Bereich spezialisiert hat. Sie hat Erfahrung in der Entwicklung von ähnlichen Systemen.

Welche Skalierungsfaktoren und Kostenfaktoren sind von der Wahl des Projektleiters betroffen? Diskutieren Sie die Auswirkungen beider Wahlmöglichkeiten auf diese Faktoren.

b) Da bisher in der Praxis schon viele Patientenverwaltungs-Systeme mit dem J2EE Framework implementiert wurden, soll, zu Zwecken besserer Wiederverwendbarkeit vorhandener Software und Kompatibilität zu vorhandenen Schnittstellen, die Software mit diesem Framework entwickelt werden. Es gibt einen Programmierer in der Firma, dem sehr gute Fähigkeiten im Programmieren mit Java und J2EE zugesprochen werden. Es müssten aber noch weitere Mitarbeiter geschult werden, um das *Heilenspital* Projekt bewältigen zu können. Alternativ könnte die gesamte Programmierung an das indische Software-Unternehmen ITIndia outgesourct werden. ITIndia stehen viele mit J2EE erfahrene Programmierer zur Verfügung.

Welche Skalierungsfaktoren und Kostenfaktoren sind von der Entscheidung, wo die Programmierung stattfinden soll, betroffen? Diskutieren Sie die Auswirkungen beider Möglichkeiten auf diese Faktoren.

c) Sie müssen den Skalierungsfaktor *Flexibilität* beurteilen. Nennen Sie zwei Eigenschaften des Projekts, welche dabei eine Rolle spielen und beurteilen Sie diese anhand des gegebenen Projektes. Falls Sie dazu Annahmen treffen müssen, begründen Sie diese.

d) Um wieviel Prozent müsste der Berechnungsfaktor KSLOC (Kilo Source Lines Of Code) verringert werden können, damit sich Zuverlässigkeit, welche den Wert des Kostenfaktors *Reliability Required* von Nominal auf Very High ändert, lohnt? Die restlichen Kostenfaktoren bleiben gleich und die Skalierungsfaktoren werden alle mit Nominal bewertet. Zur Lösung soll auch der Rechnungsweg dargestellt werden.

e) Was ist Voraussetzung, damit die Berechung mit COCOMO2 zuverlässige Werte liefert? Begründen Sie Ihre Aussage kurz.

2.4 Risikoschätzung (4 Punkte)

Im Skript Software Engineering zur Vorlesung finden Sie im Kapitel 15 die zehn häufigsten Risiken bei Software-Projekten.

a) Finden Sie 4 weitere Risiken für Ihr Softwareprojekt aus Übung 2 (keines der Risiken aus dem Skript).

b) Nehmen Sie eine Risikobewertung für die vier weiteren gefundenen Risiken vor, indem Sie das Risiko, sowie die Schadenshöhe je auf einer Skala zwischen 1 - 10 bewerten.

c) Stellen Sie passende Massnahmen zusammen, um die Risiken zu mindern oder zu eliminieren.