

Praxis der Software-Entwicklung (PSE)

Prof. Dr.-Ing. Gregor Snelting (Koordinator)

WS 09/10



Fakultät für **Informatik**

Praxis der Software-Entwicklung (PSE)

- ▶ Ziel: Entwicklung eines mittelgroßen Systems im Team mit moderner Softwaretechnik

Praxis der Software-Entwicklung (PSE)

- ▶ Ziel: Entwicklung eines mittelgroßen Systems im Team mit moderner Softwaretechnik
- ▶ Zielsystem: max 10kLOC
objektorientierter Entwurf (UML), Implementierung (Java/C++/C#), Qualitätssicherung (zB Jcov, Junit)

Praxis der Software-Entwicklung (PSE)

- ▶ Ziel: Entwicklung eines mittelgroßen Systems im Team mit moderner Softwaretechnik
- ▶ Zielsystem: max 10kLOC
objektorientierter Entwurf (UML), Implementierung (Java/C++/C#), Qualitätssicherung (zB Jcov, Junit)
- ▶ Teilnehmer: 3. Sem BA Informatik
Voraussetzung: Grundbegriffe der Informatik, Programmieren, Algorithmen 1, Softwaretechnik 1
- ▶ Empfehlung: PSE erst, wenn alle Module aus 1./2. Semester bestanden sind

Praxis der Software-Entwicklung (PSE)

- ▶ Ziel: Entwicklung eines mittelgroßen Systems im Team mit moderner Softwaretechnik
- ▶ Zielsystem: max 10kLOC
objektorientierter Entwurf (UML), Implementierung (Java/C++/C#), Qualitätssicherung (zB Jcov, Junit)
- ▶ Teilnehmer: 3. Sem BA Informatik
Voraussetzung: Grundbegriffe der Informatik, Programmieren, Algorithmen 1, Softwaretechnik 1
- ▶ Empfehlung: PSE erst, wenn alle Module aus 1./2. Semester bestanden sind
- ▶ Umfang: 6LP, \approx 200 Arbeitsstunden / Teilnehmer, \approx 1.5 Arbeitstage/Woche/Teilnehmer

PSE / Organisation

- ▶ Zeitplan: Mitte Oktober 2009 - Mitte März 2010;
15 Wochen Praktikumsbetrieb nach Absprache
- ▶ ca. 20 verschiedene Aufgabenstellungen von 15
Lehrstühlen
- ▶ objektorientiertes Phasenmodell verbindlich (vgl.
Modulhandbuch)
- ▶ ca. 40 Teams a 5-6 Studenten
Wünsche zu Teamzusammensetzung / Aufgabe werden
nach Möglichkeit berücksichtigt

PSE / Organisation

- ▶ Zeitplan: Mitte Oktober 2009 - Mitte März 2010;
15 Wochen Praktikumsbetrieb nach Absprache
- ▶ ca. 20 verschiedene Aufgabenstellungen von 15
Lehrstühlen
- ▶ objektorientiertes Phasenmodell verbindlich (vgl.
Modulhandbuch)
- ▶ ca. 40 Teams a 5-6 Studenten
Wünsche zu Teamzusammensetzung / Aufgabe werden
nach Möglichkeit berücksichtigt
- ▶ **Auftaktveranstaltung**, Aufgabenvorstellung, endgültige
Teamzusammenstellung: **Di 20.10.09, 15:45, HSaF**
- ▶ Bitte auch Webseite beachten:
<http://pp.info.uni-karlsruhe.de/lehre/WS200910/pse/>

- ▶ moderne Softwaretechnik ist wichtig für alle BA-Absolventen!
- ▶ vollständige Entwicklung eines größeren Systems
- ▶ Phasenmodell: Pflichtenheft, Entwurf, Spezifikation, Implementierung, Validierung
- ▶ Abschlusspräsentation
- ▶ Teamarbeit (Teams a 5-6 Teilnehmer)
- ▶ durchgehend Objektorientierung
- ▶ Toolunterstützung, zB Eclipse, Rational Architect, JUnit, Jcov, ...

Phasenziel

detaillierte Festlegung der Leistungsmerkmale eines Systems

Grundprinzipien

- ▶ Präzision
- ▶ Vollständigkeit
- ▶ Konsistenz

Vorgehen

- ▶ Systemmodell (grobe Übersicht), Systemumgebung (Hard/Software)
- ▶ vollständige funktionale Anforderungen
- ▶ GUI-Entwürfe (manuell oder programmiert)
- ▶ ausführliche Testfallszenarien

verlangt wird

Abgabe des Pflichtenheftes nach 2 Wochen;
Erläuterung im ersten Kolloquium

Entwurf & Spezifikation

objektorientiert (UML)

Phasenziel

- ▶ Festlegung der Klassenstruktur
- ▶ Schnittstellendefinition der Klassen
- ▶ Beziehungen zw. Klassen (Vererbung, Assoziationen)
- ▶ Klassendiagramm, ausgewählte Sequenzdiagramme, evtl. Zustandsdiagramm
- ▶ Einsatz von Design Patterns, MVC

Entwurf & Spezifikation

objektorientiert (UML)

Phasenziel

- ▶ Festlegung der Klassenstruktur
- ▶ Schnittstellendefinition der Klassen
- ▶ Beziehungen zw. Klassen (Vererbung, Assoziationen)
- ▶ Klassendiagramm, ausgewählte Sequenzdiagramme, evtl. Zustandsdiagramm
- ▶ Einsatz von Design Patterns, MVC

Grundprinzipien

- ▶ Geheimnisprinzip
- ▶ schwache Kopplung
- ▶ hohe Kohäsion
- ▶ Lokalitätsprinzip
- ▶ Wiederverwendbarkeit von Klassen/Subsystemen

Entwurf & Spezifikation

objektorientiert (UML)

Phasenziel

- ▶ Festlegung der Klassenstruktur
- ▶ Schnittstellendefinition der Klassen
- ▶ Beziehungen zw. Klassen (Vererbung, Assoziationen)
- ▶ Klassendiagramm, ausgewählte Sequenzdiagramme, evtl. Zustandsdiagramm
- ▶ Einsatz von Design Patterns, MVC

Grundprinzipien

- ▶ Geheimnisprinzip
- ▶ schwache Kopplung
- ▶ hohe Kohäsion
- ▶ Lokalitätsprinzip
- ▶ Wiederverwendbarkeit von Klassen/Subsystemen
- ▶ OO: Vererbung/dynamische Bindung statt Fallunterscheidung

Vorgehen

- ▶ Kombination von Top-Down und Bottom-Up Design
- ▶ Identifikation von Klassen, Vererbung, Assoziationen
- ▶ Festlegung der Schnittstellen aller Klassen
- ▶ informale Beschreibung aller Klassen
- ▶ evtl. Einsatz von Entwurfsmetriken (JMetrics)

Vorgehen

- ▶ Kombination von Top-Down und Bottom-Up Design
- ▶ Identifikation von Klassen, Vererbung, Assoziationen
- ▶ Festlegung der Schnittstellen aller Klassen
- ▶ informale Beschreibung aller Klassen
- ▶ evtl. Einsatz von Entwurfsmetriken (JMetrics)

verlangt wird

- ▶ Abgabe der UML-Diagramme nebst informeller Beschreibung nach 4 Wochen;
Verteidigung im zweiten Kolloquium
- ▶ *Nachweis der Evolutionsfähigkeit* (zB Lokalitätsprinzip)

Implementierung & Validierung

Phasenziel

Programmierung und Test des Systems

Grundprinzipien

- ▶ Programmierung in Java (evtl. C#, C++)
- ▶ Umsetzung der Architektur
- ▶ werkzeugunterstützte Qualitätssicherung

Phasenziel

Programmierung und Test des Systems

Grundprinzipien

- ▶ Programmierung in Java (evtl. C#, C++)
- ▶ Umsetzung der Architektur
- ▶ werkzeugunterstützte Qualitätssicherung

Vorgehen

- ▶ Implementierung der Methoden
- ▶ funktionaler Komponententest mit JUnit, evtl. Überdeckungstests (zB Jcov); verschränkt mit Implementierung
- ▶ Integrationstest, Robustheitstest
- ▶ Realisation der Szenarien aus Pflichtenheft

verlangt wird

- ▶ Implementierungsplan; Implementierung;
- ▶ Implementierungskolloquium (2 Wochen vor Abschluss)
- ▶ Testbericht; Systemabnahme (1 Woche vor Abschluss)
- ▶ Abschlusspräsentation (Februar/März 2010)