

Studienführer Master of Advanced Studies in Software Engineering

www.hsr.ch/weiterbildung

Berufsbegleitendes Nachdiplomstudium MAS FHO in Software Engineering



**HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL**



WEITERBILDUNG

Summary

Zielgruppe	Im Umfeld der Softwareentwicklung tätig
Ziel	Professionelle Anwendung von aktuellem Software Engineering
Umfang	4 Semester berufsbegleitend
Aufbau	3 Semester Zertifikatskurse mit Abendunterricht am Dienstag und Donnerstag sowie ca. 10 Stunden pro Woche Selbststudium Masterarbeit von ca. 400 Stunden im 4. Semester
Durchführungsort	HSR Hochschule für Technik Rapperswil
Aufnahmebedingungen	Hochschulabschluss oder entsprechende Praxisnachweise, Programmiererfahrung
Kosten	CHF 23 000.–
Anmeldung	Mit Anmeldedossier (beiliegend oder siehe Seite 13) bis zum 31. Januar 2012 Berücksichtigung in der Reihenfolge des Eingangs
Start	10. April 2012
Abschluss	Eidgenössisch anerkannter «Master of Advanced Studies» der Fachhochschule Ostschweiz
Titel	MAS FHO in Software Engineering
Warum an der HSR?	– Langjährige Erfahrung mit Weiterbildung in Software Engineering – Dozenten mit praxisbezogenem Know-how – Gute Akzeptanz in Wirtschaft

Zielgruppe

Der Master of Advanced Studies richtet sich an Hochschulabsolventinnen und -absolventen, die in der Softwareentwicklung tätig sind. Zugelassen werden auch ausgewiesene Praktiker mit mehrjähriger Berufserfahrung.

Viele Teilnehmerinnen und Teilnehmer dieser Ausbildung haben ursprünglich nicht Informatik studiert, sind aber in der Softwareentwicklung tätig und möchten ihr Know-how professionalisieren oder planen den nächsten Karriereschritt.

Zielsetzung

Der Master of Advanced Studies in Software Engineering ist in der Lage, Softwareprojekte von der Inception- bis in die Transition-Phase kompetent zu bearbeiten.

Das Studium gibt den Teilnehmerinnen und Teilnehmern die modernsten Methoden, Prozesse und Technologien des Software Engineerings in die Hand. Ziel ist das Anwendenkönnen.

Studienkonzept und Aufbau des Studiums

Das Masterstudium gliedert sich in drei Zertifikatskurse von je einem Semester sowie eine Masterarbeit von einem Semester.

1. Semester Certificate SE foundation Grundlagen, Methoden Unterricht, Selbststudium	2. Semester Certificate SE intermediate Methoden, Technologien Unterricht, Selbststudium	3. Semester Certificate SE advanced Methoden, Technologien, Management Unterricht, Selbststudium	4. Semester MAS-Arbeit Selbstständige Teamarbeiten mit Coach
--	--	---	--

Für jeden Certificate-Kurs gilt:

- Ca. 200 Lektionen Unterricht und ca. 200 Stunden Selbststudium
- Aufbau aus Kursmodulen und Kursblöcken
- Hoher Übungsanteil (in den meisten Kursblöcken 50 %)
- Unterrichtszeiten Dienstag und Donnerstag je 5 Lektionen: 17.15–21.50 Uhr
- An den Prüfungsterminen werden jeweils mehrere Kursblöcke geprüft

Die Certificate-Kurse bauen inhaltlich aufeinander auf. Wie im modernen Software Engineering wird zum Teil auf ein iteratives Vorgehen gesetzt: Zentrale Themen werden zuerst eingeführt, später vertieft.

In der MAS-Arbeit führen Studierende in Teams ein Software Engineering Projekt durch. Die Themen werden von den Studierenden eingebracht. Jedem Team steht ein Coach aus dem Kreis der Dozierenden zur Seite.

Studieninhalte Certificate of Software Engineering – foundation

Einführung Objektorientierte Softwareentwicklung

- Grundbegriffe: Klassen, Vererbung, Polymorphismus
- Einführung in UML
- Objektorientierte Analyse und Objektorientiertes Design mit UML
- Umsetzung des Objektorientierten Designs in Programmiersprache Java

Arbeiten in Teams

- Selbstmanagement
- Kommunikation und Kommunikationsmodelle
- Teambildung und Teamdesign
- Teamdiagnose und Teamentwicklung

Grundlagen Betriebssysteme

- Programmausführung und Hardware
- Systemprogrammierung
- Prozesse, Threads und Interprozesskommunikation
- Ein-/Ausgabe

Einführung in Requirements Engineering

- Übersicht Requirements Engineering Methoden
- Use Cases für Funktionale Anforderungen
- Qualitätsmodell für Nichtfunktionale Anforderungen
- Requirements Management: Verwalten, Priorisieren, Changemanagement
- Quellen und Erhebungstechniken für Requirements

Projektarbeit Objektorientierte Softwareentwicklung

- Objektorientierte Analyse, Objektorientiertes Design und Objektorientierte Programmierung eines Warenautomats in Java
- Arbeit wird in kleinen Teams ausgeführt

Objektorientierte Modellierungspraxis

- Grundlagen der Domainmodellierung
- UML für Domainmodellierung
- Modellierungstraining an mehreren Fallstudien

Programmieren mit Java

- Datentypen, Operatoren, Anweisungen und Kontrollstrukturen
- Programmstruktur (Blöcke, Methoden, Klassen, innere Klassen, Anwendung von Generics, Packages)
- Schnittstellen, Vererbung und Polymorphismus, Exception Handling
- Ein-/Ausgabe und Streams
- Architektur der Java Virtual Machine

Projektmanagement

- Projektmanagement und -organisation
- Projektziele, Aktoren und Rollen
- Phasen- und Entwicklungsmodelle, Rational Unified Process (RUP)
- Dokumentation, Fortschrittskontrolle, Berichtswesen,

Studieninhalte Certificate of Software Engineering – foundation

- Projektabschluss, Debriefing
- Change Management, Software-Konfigurationsmanagement

Windows-Betriebssysteme

- Einführung: Entstehung, Versionen und Architektur
- Windows Registry und Services
- Zugriffsrechte
- Prozesse, Threads und Scheduling, Interprozesskommunikation
- Windows Graphical User Interface
- Dateisysteme: FAT, NTFS und Festplattenpartitionierung

Software-Prozesse

- Phasen, Meilensteine, Deliverables
- Wasserfallmodell und iterative Prozesse
- Rational Unified Process: mit Artifacts und Activities, Tailoring
- Kurzüberblick Agile Prozesse: Extreme Programming (XP), Crystal
- Bewertung des Reifegrades von Software-Entwicklungsorganisationen (SEI-CMM)

Programmieren in C++

- Datentypen, Operatoren, Anweisungen und Kontrollstrukturen

- Blöcke, Funktionen, Klassen, friend Klassen, Templates, Namespaces
- Vererbung, Virtuelle Funktionen, Mehrfachvererbung
- Dynamische Speicherverwaltung
- Schnittstelle zur Programmiersprache C, Runtime-Type-Information RTTI
- Exception Handling, Input- und Output in C++
- ANSI/ISO C++ und Klassenbibliotheken

Unix-Betriebssysteme

- Einführung: Entstehung, Derivate, Normierung
- UNIX-Philosophie und Architektur (Kern und GUI-Aufsätze)
- Unix Shell und Shell Scripts, Unix-Programmierung
- Parallelverarbeitung und Interprozesskommunikation, Prozesszustände, Scheduling
- POSIX Threads, Interprozesskommunikation (IPC): Unix Signale, Unix Pipes

Algorithmen und Datenstrukturen

- Rekursion
- Analyse von Algorithmen: O-Notation, Arithmetische Progression
- Sortierung: Selection-, Insertion-, Merge- und Quick-Sort
- Collections: Vector, ArrayList, Stack, Queue, Linked List, Map und Dictionary, Bäume

Studieninhalte Certificate of Software Engineering – intermediate

Programmieren mit Java advanced

- Programmierung von Generics
- Annotations
- Java-Reflection API
- Garbage-Collection (strong-, Weak-, Soft- und Phantom-Referenzen)
- Java Native Interface (JNI)
- Aspekt-Orientierte Programmierung AOP mit AspectJ
- Design-by-Contract mit Assertions und der Java Modeling Language (JML)
- Java-Internationalization und Logging-Framework

Software-Architekturen

- Grundprinzipien guter Applikationsarchitekturen
- Rolle des Architekten
- Architekturtypen: Schichten, Pipes und Filters, Interaktive und Verteilte Systeme
- Sichten auf Architekturen
- Architektur-Patterns

XML-Technologien

- XML-Grundlagen: Anwendung, Syntax, Gültigkeit (DTD, Schemas), Stylesheets
- XML-Parser: Übersicht, SAX- und DOM-Parser
- XML-Protokolle: XML-RPC, WSDL, UDDI

Objektorientiertes Design (OOD)

- Grundprinzipien Objektorientiertes Design
- Responsibility Driven Design
- UML als Designnotation
- Design-Verifikation
- Design Pattern
- Control Style und Software-Architektur

Grundlagen Datenbanken

- Datenmodellierung, Relationales Datenmodell, Normalisierung
- SQL als DDL und DML
- DB-Programmierung (JDBC)
- Stored Procedures und Triggers

Parallel- und Netzwerkprogrammierung

- Multi-Threading mit Java, Java Memory Modell
- Synchronisation (Kritische Abschnitte, Monitor Prinzip)
- Zustandssynchronisation und gegenseitiger Ausschluss
- Semaphoren und Locks (Condition Variable)
- Deadlocks, Klassische Synchronisationsprobleme
- Socketprogrammierung in Java und C++
- Architektur und Patterns für verteilte Prozesssysteme mit Sockets

Studieninhalte Certificate of Software Engineering – intermediate

Einführung Internettechnologien

- Architektur von Internetapplikationen, Übersicht Internettechnologien
- Java-Programmierung auf dem Webserver: Servlets, JSP, Tag Libraries
- MVC für Web-Applikationen, Framework, Struts

Distributed Computing, EJB

- Grundlagen: Basisarchitekturen, Interprozesskommunikation, Service-Architekturen
- Middleware: Synchrone Kommunikation (RMI, CORBA), Asynchrone Kommunikation (JMS)
- Design, Implementation und Integration: Modularisierung, Schnittstellen-Design, Design eines Services
- Enterprise JavaBeans

Security

- Netzwerk- und Internetsicherheit
- Risiken der Internetnutzung
- Kryptologie, Public und Private Key Systems
- Massnahmen zur Verbesserung der Sicherheit (Passworte, SSL/TLS, VPN, Firewalls)
- Rechtslage

C# und .net

- Grundlagen des .NET Frameworks
- Grundkonzepte von C#, LINQ
- Datenbankzugriff mit ADO.NET
- Windows Presentation Framework

Software Testing

- Überblick Teststufen (Komponententest, Integrationstest, Systemtest, Akzeptanztest)
- Test-driven development (TDD)
- Unit Testing mit JUnit, Mock Objekte, Design for Test, Coverage
- Testautomatisierung und Projektautomatisierung
- Testtools für System- und Akzeptanztests (Fit/Fitnesse, Webtest)
- Testprozess und Testmanagement
- Testfalladministration und Auswertung der Testresultate

Studieninhalte Certificate of Software Engineering – advanced

Requirements Engineering advanced

- Requirements Engineering Prozess
- Qualitätsmodell für Anwendungssysteme
- Essenzielle Objektorientierte Analyse der Funktionalen Anforderungen
- Systemdesign der Funktionalen und Nicht-funktionalen Anforderungen

Requirements Engineering Projektarbeit

- Anwendung der Methode aus Requirements Engineering advanced an umfangreicher Fallstudie

Datenbanken advanced

- Objektrelationale DBMS, OR-Mapper und OODBMS
- XML-Datenbanken
- Dataware House, OLAP-Cubes, ETL-Prozess

Rechtliche Aspekte

- Überblick über das Immaterialgüterrecht
- Software: Schutz und Überlassung (Übertragung und Lizenzierung)
- Domain-Streitigkeiten
- E-Commerce: Rechtliche, insbesondere Datenschutz- und persönlichkeitsrechtliche Aspekte im Zusammenhang mit einer Internetplattform

Service Oriented Architecture

- Service Oriented Architecture (SOA): SOA-Grundlagen, Service-Provider-Layer, Intermediary Layer, Business Processing Layer

Qualitätsmanagement

- Grundlagen der Software-Prüfung
- Messbare Merkmale von Software
- Reviews
- Übersicht Qualitätsmanagement
- Capability Maturity Integration Model (CMMI)
- Risikomanagement

Agile Software Development

- Grundkonzepte, Agile Manifesto
- Agile Prozesse: XP, Crystal Clear, Adaptive Software Development (ASD), Feature Driven Development (FDD), Scrum
- Agile Unified Process
- Vergleich und Einordnung Agiler Prozesse

Project Automation

- Build Automation: Einführung, Ant
- Scheduled Builds mit Cruise Control
- Testing, Packaging und Deployment mit Ant
- Release Automation mit Ant und Maven

Studieninhalte Certificate of Software Engineering – advanced

Performante Teams, Process Communication Model

- Moderationsformen von Teams
- Konstruktive Kommunikation als Schlüssel zur Teamarbeit
- Gruppenregeln zur Effizienzsteigerung in Gruppenarbeiten
- Kommunikationsmodell der Process Communication: grundlegende Persönlichkeitsprofile, Dynamik der zwischenmenschlichen Kommunikation

Model Driven Software Development

- MDSM Ideen und Prinzipien (Modelle, Transformationen, Plattformen)
- Implementation von MDSM (UML profiles for Domain Specific Languages)
- Alternative Ansätze, Erfahrungen, Nutzen

IT Architecture

- Architectural Patterns und Reference Architectures
- Enterprise Architecture (Position, Methods, Capabilities & Principles)
- Enterprise Architecture Framework
- Business Architecture through Component Business Modeling
- Governance & SOA Governance
- Quality Constraints in IT Architecture

Webframeworks und -technologien

- JavaScript: Einführung in Sprache, Objektmodell, Closures
- DOM-Manipulationen, Event-Handler in JavaScript
- AJAX: Konzept und Anwendung, XMLHttpRequest-Object, Übermittlungsformate, JSON
- JavaServer Faces: Einführung, JSF Life Cycle, UI Component Model, Event Listener, Converter und Validator, JSF Frameworks
- Ruby on Rails: Einführung in Ruby, Architektur von Webapplikationen mit Ruby on Rails

Human Computer Interaction Design, GUIs in Java

- Grundlagen Userinterfaces in Java: Model View Controller (MVC) Architektur in Java, Composite Pattern in GUIs, Ereignisverarbeitung, Übersicht der Java-Bibliotheken für Graphical User Interfaces: awt, Swing, SWT
- Usage Centered Design of User Interfaces (Constantine u.a.)
- Techniken des HCI-Designs
- Experimentelle Userinterfaces

Dozierende

Die Dozierenden des MAS Software Engineering zeichnen sich durch langjährige Erfahrung und einen hohen Praxisbezug aus. In den praktischen Übungen werden die Dozierenden von zusätzlichen Übungs-

betreuern unterstützt, um eine gute Betreuung der Studierenden zu gewährleisten. Die nachfolgende Liste der Dozierenden ist eine Momentaufnahme und Änderungen unterworfen.

Baeriswyl Bruno, Dr. iur.

Bianchi Bernhard, Dipl. Ing.

Bögli Alex, Dipl. Inform. UNIZ

Briner Thomas, Dipl. Inf-Ing. ETH

Forster Heinrich, Prof.

Frühauf Karol, Dipl. Ing.

Glatz Eduard, Prof.

Grau Rainer, Dipl. Ing.

Hauri Christian, lic. phil. Psychologe

Heinzmann Peter, Prof. Dr.

Hoidn Hans-Peter, Dr.

Huser Hansjörg, Prof.

Joller Josef, Prof. Dr.

Datenschutzbeauftragter des Kantons Zürich

Bianchi & Partner GmbH

Zühlke Engineering AG

Abraxas Informatik AG

HSR Hochschule für Technik Rapperswil

INFOGEM AG

HSR Hochschule für Technik Rapperswil

Zühlke Engineering AG

Hauri Ergonomie & Coaching

HSR Hochschule für Technik Rapperswil, cnlab AG

Senior IT Architect, IBM Software Group

HSR Hochschule für Technik Rapperswil

HSR Hochschule für Technik Rapperswil

Dozierende

Jöhr Thomas, Dipl. Ing.	Bianchi & Partner GmbH
Jucker Jürg, Dipl. Inf.-Ing. FH	HSR Hochschule für Technik Rapperswil
Kessler Markus	Consulting IT Specialist, IBM Schweiz
Koch Andres, Dipl. Ing., M. Math	Object Engineering GmbH
Kolb Robert, Dipl. Ing.	FROX communication AG
Letsch Thomas, Dipl. Ing.	Letsch Informatik
Mattmann Rudolf, Dr.	Mettler-Toledo
Stolze Markus, Prof. Dr.	HSR Hochschule für Technik Rapperswil
Qvortrup Michael, Dipl. Ing. ETH	Zühlke Engineering AG
Rudin Hans, Prof.	HSR Hochschule für Technik Rapperswil
Ruggli Sandro, lic. iur., Rechtsanwalt, LL. M.	GRP GLOOR RUGGLI PARTNER
Schlatter Marcel, Dr. sc. techn.	Distinguished Engineer, IBM Schweiz
Tobler Daniel, Dipl. Ing.	Zühlke Engineering AG
Weber Roland, lic. phil. I	Zühlke Engineering AG
Zimmermann Martin, Prof. Dr.	Hochschule Offenburg, Deutschland

Studienleitung und Studienberatung

Aktuelle Daten und Kosten entnehmen Sie bitte dem Anmeldedossier:

> <http://www.hsr.ch/Software-Engineering.2087.0.html>

Download der Broschüre und des Anmeldedossiers MAS Software Engineering unter:

> <http://www.hsr.ch/Software-Engineering.2087.0.html>

Bitte verlangen Sie per E-Mail oder Telefon das Anmeldedossier für den MAS Software Engineering

E-Mail weiterbildung@hsr.ch

Telefon +41 (0)55 222 49 22/21

Allgemeine Auskünfte und Anmeldung

HSR Hochschule für Technik Rapperswil

Weiterbildung

Susanne Rigling

Oberseestrasse 10

CH-8640 Rapperswil

Telefon +41 (0)55 222 49 22

Fax +41 (0)55 222 44 00

E-Mail susanne.rigling@hsr.ch

Studienleitung und Studienberatung

Für eine persönliche Studienberatung stehen wir Ihnen gerne zur Verfügung.

Akademische Studienleitung
HSR Hochschule für Technik Rapperswil
IFS Institut für Software
Prof. Hans Rudin, Dipl. El. Ing. ETH
Oberseestrasse 10
CH-8640 Rapperswil
Telefon +41 (0)55 222 49 36
E-Mail hans.rudin@hsr.ch
> www.ifs.hsr.ch

Administrative Studienleitung
HSR Hochschule für Technik Rapperswil
Weiterbildung
Peter Nedic, MBA
Oberseestrasse 10
CH-8640 Rapperswil
Telefon +41 (0)55 222 49 21
E-Mail peter.nedic@hsr.ch
> www.hsr.ch/weiterbildung