Major

Software Engineering for Smart Devices

(Major SES)

Person in charge :

Jean-Luc RAFFY

Objectives :

The software industry needs graduates to have both short-term, up-to-date, technical skills, and mastery of fundamental, long-term, foundational, concepts. Furthermore, as software engineering is as much an art as a science, there is great demand for creative engineers who are able to work with a wide range of people (technical and non-technical). Finally, software engineering is about working in teams to solve complex problems that may have a large impact on societies (and individuals). Industry no longer wants software engineers to develop code in isolation, it wants engineers who can play a core role in a wide range of industries, and whose expertise just happens to be in the domain of software.
The mobile device industry (phones, personal assistants, media players, satellite navigation systems, etc …) is one of the fastest growing technology markets. The devices' success is dependent on the services that they provide; and these services are engineered from software. The market for games for distributed mobile devices is growing exponentially. The techniques used for developing these (entertainment) services are state-of-the-art in terms of software engineering and information communication technologies. Furthermore, the development of such games requires understanding of foundational software engineering knowledge, combined with strong creative and social skills. In summary, students who can develop games for mobile devices will exhibit many (if not all) of the attributes that industry requires now and in the future.

The aims of this program are:

- to offer advanced education in Software Engineering to highly motivated students who wish to enhance their future career prospects in the fast growing area of mobile services, with specialisation in game development;

- to respond to the strong demand of industry, commerce, services … for qualified specialists (software engineers, technical analysts or advisors, information system designers …), and to advise on all stages of information system design and software development;

- to prepare candidates for further investigation and research in Software Engineering

Organisation :

The Software Engineering for Smart Devices program is part of the corresponding Master of Science. Its building bricks consist of 6 independent courses (each representing 45h lectures & labs and 90h homework) which define a complete and coherent graduate programme in computer sciences engineering.

During the S9 Semester, a team project (CSC5025) will allow students to gain an in-depth expertise on one of the main scientific axes presented in these UVs.

Program :

Semester 8 :

- CSC4518 : Software engineering tools
- CSC4519 : Graphic user interfaces for tools development

Semester 9 :

- CSC7321 : Software dependability
- CSC7324 : Embedded systems and real time operating systems
- CSC7013 : Agents, web services and system architectures
- CSC7323 : Software process case studies and problem-based Learning
- CSC5025 : SES major's project

CSC4518	**Software engineering tools**	
Period : S8 / P3	**ECTS :** 4	**Language :** English

Organization:

- Teaching Load / Total Load: 36/90
- Lectures/Exercises/Labs/Final Exam: 18/0/18/0

Courses provide notions and concepts about the tools that are practiced in labs.

Assessment:

One graded lab (L) and a project (P) will assess the students.

Final Grade= Average (L,P)

Objectives :

- Master software development processes,
- Master software engineering toolkits applied in industry (debugger, profiler, check of memory leak, etc.),
- Master software development industrial environments.

Keywords :

Debugger, profiler, Make, Autotools, CVS, IDE

Prerequisites :

Basic knowledge of C or C++

Course outlines:

- Basis and mechanisms to software development: MAKE and ANT
- Software debuggers: Valgrind and DDD
- GNU development tools (Automake, Autoconf, Configure, Autotools)
- Internationalization of application (gettext ...)
- Profiler (Gprof)
- Collaborative development tool. Concurrent Version Systems (CVS): main functionalities (import, checkout, commit, update, add, remove, diff)
- Integrated development platforms, environments: IDE (Eclipse, JCreator, .NET): basic usage for java development,
- Software quality and metrology.

Learning materials and literature :

- Ant User Manual, http://www.ing.iac.es/~docs/external/ant/manual/index.html
- Debugging with DDD, Byron Clark, Computer Science Department, Brigham Young University, 2005 (http://docs.cs.byu.edu/docs/ddd/)
- The Java developer's guide to Eclipse, Sherry Shavor, Jim D'anjou, Addison-Wesley, 2003
- CVS Précis et Concis, G.N. Purdy, D. Colombani, O'Reilly, 2004
- Learning the GNU development tools (Eleftherios Gkiouleka, Marcelo Roberto Jimenez): http://www.scs.stanford.edu/~reddy/links/gnu/tutorial.pdf

Person in charge:

Eric Lallet (Eric.Lallet@it-sudparis.eu)

Lecturers :

- Dr Eric Lallet, TELECOM SudParis
- Dr Jean-Luc Raffy, TELECOM SudParis

CSC4519	**Graphic user interfaces for tools development**	
Period : S8 / P4	**ECTS :** 4	**Language :** English

Organization :

- Teaching Load / Total Load: 27/90
- Lectures/Exercises/Labs/Final Exam: 12/0/15/0

Among the courses, six hours are dedicated to the presentation of industrial tools illustrated by the industrials themselves.

Assessment:

Individual labs (L) and projects (P) will assess the students.

Final Grade= Average(L,P)

Objectives :

- Master development techniques for graphic interfaces dedicated to software engineering (IHM),
- Master frameworks used in the industry,
- Design and develop user-friendly graphic interface,
- Master platform independent interface design.

Keywords :

IHM, Graphical interfaces, TCL/TK, Open-GL

Prerequisites :

Knowledge of a procedural programming language

Course Outlines :

- The Tcl language (Tool Command Language). Tcl is a script language and an interpreter for this language to be easily included in an application.
- The tool box Tk (Tool kit) that allows to create graphical user interfaces in Tcl.
- Applications development especially linked to the C, Perl, and Python language.
- Introduction to open-GL

Learning materials and literature:

- *Tcl/Tk Reference Guide*, (http://www.slac.stanford.edu/~raines/tkref.html)
- "Graphical applications with Tcl & Tk", E. Johnson M&T books, 1997

Person in Charge :

Dr Jean-Luc Raffy (jean-luc.raffy@it-sudparis.eu)

Lecturers :

Dr Jean-Luc Raffy, TELECOM SudParis
Dr Anis Laouiti, TELECOM SudParis

CSC7321 Software dependability

Period : S9 / P1 **ECTS :** 4 **Language :** English

Organization :

- Teaching Load / Total Load : 45/90
- Lectures/Exercices/Labs/Final Exam : 15/0/27/3

Evaluation :

Continual assessment (CA)

Written examination (WE)

Final Grade= Average(CA, WE)

Objectives :

- Mastering application of standard techniques and tools in 4 key areas :
  - fault prevention: how to prevent fault occurrence or introduction
  - fault tolerance: how to provide a service complying with the specification in spite of faults
  - fault removal: how to reduce the presence (number, seriousness) of faults
  - fault forecasting and modelling: how to estimate the present number, the future incidence, and the consequences of faults
- Expertise in designing and auditing of dependable systems, Critical Software Architecture, Analysis and Design of fault tolerant software and systems.

Keywords :

Critical Systems, Fault tolerance, Security, Trust, Tests.

Prerequisites :

Object Oriented Programming, Basics of Software Engineering, Mathematical Foundations

Course outlines :

- Case Studies – railways, nuclear power plants, avionics, space flights,
- Failure Modelling, Simulation and Probabilistic Analysis
- Critical Systems and system engineering
- Complexity
- Formal models and verification
- Security and Trustworthiness – the internet problem
- New areas of research – self healing systems

Learning materials and literature:

Testing Object-Oriented Systems: Models, Patterns, and Tools (The Addison-Wesley Object Technology Series) by Robert V. Binder

Automated Software Testing: Introduction, Management, and Performance by Elfriede Dustin, Jeff Rashka, John Paul, Addison-Wesley Professional

The learning material will be also based on case studies from industry with foundational information provided in the form of published research articles.

Person in Charge:

J Paul Gibson (Paul.Gibson@it-sudparis.eu)

Lecturers :

J. Paul Gibson

CSC7324	**Embedded systems and real-time operating systems**	
Period : S9 / P2	**ECTS :** 4	**Language :** English

Organization :

- Teaching Load / Total Load: 45/90
- Lectures/Exercises/Labs/Final exam: 39/0/3/3

Evaluation :

Individual labs (L) and projects (P) will assess the students. The «final exam» session is devoted to projects presentations and evaluation.
Final Grade = Average (L,P)

Objectives

- To master the architecture of embedded systems by studying the structure of a RISC microcontroller and its peripherals;

- To acquire microcontroller programming skills by manipulating and programming in assembler a RISC microcontroller (PIC family);

- To understand the principles and mechanisms of a real-time operating system: goals, architecture, fundamental concepts: tasks, scheduling, time management, shared data, concurrency, synchronization, mutual exclusion;

- To master the design and architecture of a minimal real-time kernel in theory and practice (design of a simple scheduler, semaphore management).

Keywords :

RISC Microcontroller, peripherals, real-time programming.

Prerequisites :

Computer architecture: processor, memory, peripherals etc.; microprocessor architecture; procedural programming.

Course outlines :

- Brief overview on computer architecture, CPU development history, state-of-the-art and perspectives, introduction to embedded systems in general, concept of microcontroller, main families of microcontrollers on the market;

- Detailed study of a RISC microcontroller (PIC family) and its programming; hands-on lab experiments and assembler programming in a simulated environment;

- Introduction on real-time operating systems and software;

- Real-time programming techniques;

- Design and implementation of a real-time kernel.

Learning materials and literature :

- Embedded System Design: A Unified Hardware/Software Introduction. Frank Vahid and Tony Givargis, John Wiley & Sons; ISBN: 0471386782. Copyright (c) 2002.

- Real-Time Systems, Jane W. S. Liu, University of Ilinois at Urbana-Champaign, ISBN-10: 0130996513, Prentice Hall (c) 2000.

- http://microchip.com

Person in Charge:

Daniel Ranc (Daniel.Ranc@it-sudparis.eu)

Lecturers :

Daniel Ranc

CSC7013	**Agents, web services and system architectures**	
Period : S9 / P3	**ECTS :** 4	**Language :** English

Organization :

- Teaching Load / Total Load : 33/90
- Lectures/Exercises/Labs/Final Exam : 0/23/10/0

Tutorials focus on the different concepts about agents, Web services and system architectures. Then the students will use the available tools to come from basis to labs with real case studies.

Assessment :

A common project (P) and a written work (W).

Final Grade = Moy(P+W)

Objectives :

At the end of this course, the students are supposed to be able to:

- Understand recent models of software development: MDD (Model Driven Development), Agile methodologies;
- Understand recent models of software system architecture: SOA (Service Oriented Architecture), MDA (Model Driven Architecture), MAS (Multi Agent Systems), Web Services;
- Master interoperability between: heterogeneous platforms, operating systems, communication protocols, data models and programming languages,
- Master integration of (Web) services and applications,
- Master software reuse and adaptability, etc.;
- Operate and optimize a platform for the development of MAS and Web Services

Keywords:

Software development, multi agent systems, web services.

Prerequisites :

Basics of Java, basics of XML

Course Outlines:

- General introduction to SOA, MDA and MDD. Example of SOBA (Service Oriented Business Applications) will be illustrated ;
- Web Services Description Languages (WSDL), Semantic Web and semantic interoperability;
- Introduction to multi agent systems (MAS): communication, interoperability, coordination and cooperation issues. Introduction to the JADE multi-agents platform and labs using JADE;
- Application development through Web Services and MAS

- Multi agents modeling for business processes in information systems;
- Introduction of OpenACS to deliver some package services. Labs on the OpenACS server;

Learning materials and literature:

Learning materials : slides hard copy and courses on intelligent agents (3 vol.)

Literature :

- Denis Berthier, Le savoir et l'ordinateur, chapitre 9, L'Harmattan, 2002.
- JADE documentation (http://jade.cselt.it/)
- WSDL specification (http://www.w3.org/TR/wsdl).
- http://openacs.org
- Bradshaw Jeffrey ed., Software Agents, AAAI Press / MIT Press, Cambridge, Mass., 1997.
- Klusch Matthias, ed., Intelligent Information Agents, Springer, 1999.

Software Factories, J.Greenfield and K.Short, Wiley, 2004

Person in Charge:

Denis Berthier (Denis.Berthier@it-sudparis.eu)

Lecturers :

From TELECOM SudParis :

- Denis Berthier
- Stéphane Maag

Guest lecturer :

- Cédric Besse (Univ. Paris 6)

CSC7323	**Software process case studies and problem-based Learning**	
Period : S9 / P4	**ECTS :** 4	**Language :** English

Organization :

- Teaching Load / Total Load : 45/90
- Lectures/Exercises/Labs/Final Exam : 0/20/22/3

Assessment :

Students will be required to work, in teams, on a small set of problems based on relevant, up-to-date industrial case studies. The use of software engineering processes and methods during the team development project will be evaluated – with emphasis on how the students work (together) rather than what they produce.

Final Grade = average (4 project grades and exam)

Objectives :

- Mastering Software Development Process including cover agile development, rapid-prototyping and model-based development.
- Mastering intra-team communication in software development
- Applying project management techniques

Prerequisites :

- Good knowledge : network protocols, general networking, network architectures, structured (OO) development
- Basic knowledge : underlying computer hardware and electronics.

Course Outlines:

The program will be based around an evolving set of industrial case studies and the pedagogic approach is based on Problem-based learning.

The problems will be taken from the following domains:

- Software Design for Mobile Devices - students will review a small set of real world, mobile, software and services, and will be asked to solve a small problem in this domain
- Embedded Development – students will be asked to analyse the operation of a specific embedded platform, and then to use the platform to solve a small, real-world, problem.
- Game Development for mobile devices - students will be presented with an existing game and asked to carry out maintenance and improvements.
- Robotics and Algorithm – students will be asked to design simple algorithms that allow robots to interpret and interact with a simple world around them.

Learning materials and literature:

Following the problem-based learning approach, the problems are themselves the learning material.

Person in charge:

J Paul Gibson (Paul.Gibson@it-sudparis.eu)

Lecturers :

J Paul Gibson

CSC5025	**SES major's project**	
Period: S9	**ECTS:** 8	**Language:** English

Organization :

- Scheduled Meeting / Total Load: 24h/225h

The SES major's project is done during the whole of semester 9. Each student must carry out a project in a team with at least three other students. Planning time slots are dedicated to the project. Meeting with a project manager will take place every other week, on average.

Each team will be required to choose a project within one of the domains from the *Software Process Case Studies and Problem-based Learning* module*.*

The students must demonstrate that they are able to build a high-quality solution (involving software that they have engineered) to a specific problem. We build on the experience they have gathered in the *Software Process Case Studies and Problem-based Learning* module, so that they are not required to learn new techniques or tools. Emphasis will be placed on a professional approach to system engineering (software and hardware) and the quality of the final product.

Assessment:

The validation of this project is based on the production of a real industrial strength system (evaluated under industrial criteria, G) and an oral presentation P.

Final score = 80%G + 20%P

Sample of topics :

- Software Design for Mobile Devices - students to create real world, mobile, software and/or services

- Embedded Development – students will develop their own embedded device by working in both
hardware and software to build an embedded solution on a specific embedded platform.

- Game Development for mobile devices - students will design, develop, test and market a game prototype aimed at mobile devices

- Robotics and Algorithm – students will design and implement sophisticated algorithms that allow robots to interpret and interact with a particular (industrial-oriented) environment.

Person in charge:

Jean-Luc RAFFY (jean-luc.raffy@it-sudparis.eu)

Faculty:

All lecturers of SES major