MAKERERE UNIVERSITY

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

DEPARTMENT OF NETWORKS

P.O.BOX 7062,

KAMPALA, UGANDA

PhD In Software Engineering

September 2009

DAY/EVENING PROGRAMME

Table of Contents

1 INTRODUCTION....3
1.1 Background to the Faculty of Computing and Information Technology....3
1.2 Objectives....4
1.3 Justification....5
1.4 Collaboration Partners on PhD in Software Engineering....5
1.4.1 University of Groningen, Radboud University Nijmegen and Eindhoven University of Technology....5
1.4.2 University of Bergen....6
1.4.3 London South Bank University....6
1.5 Computing Equipment....7
1.6 Physical Facilities....7
1.7 Financial Resources....8
2 REGULATIONS....8
2.1 Entrance Requirements....8
2.2 Duration....8
2.3 Credit Units (CU)....8
2.4 Core and Elective Courses....8
2.5 Graduation Requirements....8
2.6 Fundamentals of the revised program....9
2.7 The curriculum for Doctor of Philosophy (Software Engineering)....9
2.8 Grading of Courses....10
2.9 Minimum Pass Mark....10
2.10 Calculation of Cumulative Grade Point Average (CGPA)....10
2.11 Progression....11
2.12 Normal Progress....11
2.13 Probationary....11
2.14 Discontinuation....11
2.15 Re-taking a Course....11
2.16 PhD Dissertation....12
2.17 Passing of a Dissertation....12
2.18 Revised Dissertation....12
3 THE PROGRAMME....12
3.1 Summary of Curriculum....12
4 DETAILED CURRICULUM....13
4.1 PIS 9101: Presentations, Scientific Writing and Research Ethics (3CU)....13
4.2 PCS 9101: Philosophy of Computing (3CU)....15
4.3 PSE 9102: Science of Programming (3CU)....16
4.4 PIT 9201 Advanced Research Methods (3 CU)....18
4.5 PSE 9201: Models of Software Systems (3CU)....20
4.6 PSE 9203: Software Systems Architectures (3CU)....22
5 QUALITY ASSURANCE....24
6 LOAD DISTRIBUTION....25

1 INTRODUCTION

1.1 Background to the Faculty of Computing and Information Technology

The rate of growth of Information and Communication Technology (ICT) in Uganda in particular and the African region in general is enormous. In order to sustain the high growth useful to the economy, there is need for highly skilled and specialized ICT labor force to cater for the sophisticated ICT-jobs. Today Makerere University Faculty of Computing and Information Technology (CIT) is the main ICT training, research and consultancy centre in Uganda. CIT was established by the University Council at its 100th meeting held on 15th December 2004 by upgrading the Institute of Computer Science into a faculty with four departments of computer science, networks, and information technology and information systems. The Institute of Computer Science, which was established by the University Council in 1985, grew out of the University Computer Centre.

The Department of Networks, Faculty of Computing and Information Technology (CIT) currently supports PhD in Software Engineering by research and course work in addition to M.Sc in Data Communication and Software Engineering that has three options, namely Communication Networks, Mobile Computing and Application Software development, and Software Engineering. The department also offers Post Graduate Diploma programs on the same themes plus Bachelor of Science in software engineering

CIT's Value Statement: The Faculty of Computing and Information Technology is an innovative and industry-oriented Faculty, pursuing excellence in teaching, learning, cutting edge value-added research and consultancy, community outreach, as well as providing a vibrant student life.

Vision: To be a leader in Computing and ICT training, research and services internationally.
Mission Statement: To provide first class teaching, research and services in computing and ICT responsive to national and international needs.

The Faculty has been running a **PhD by research** since 2002 and continues to do. Specifically the Department of Networks runs the following graduate Programmes:

- PhD In Software Engineering
- M.Sc. in Data Communications and Software Engineering.
- Postgraduate Diploma in Data Communications and Software Engineering.
- Postgraduate Diploma in ICT Policy and Regulation.
- Bachelor Science in Software Engineering

The revised software engineering research will pursue the discovery of principles and the development of technologies to support the engineering of large, complex software systems. The challenging targets for this work are organizations and software systems operating in the wide-area, heterogeneous, distributed, and decentralized context of wide-area networks such as the Internet. Research in scientific computing including scientific modeling is also undertaken.

1.2 Objectives

The objectives of the PhD in Software Engineering by Coursework and Research programme are to: -

i. Build human resource capacity in the areas of software engineering in both the public and private sectors, especially in universities;
ii. Develop research capacity in the areas of software engineering;
iii. Address the increasing demand for PhD holders in the areas of computer science, information systems, information technology and software engineering;
iv. Strengthen capacity and institutional building in the area software engineering disciplines in tertiary institutions, private and public sectors.
v. Provide those masters holders with potential for PhD with opportunities to develop skills in formulating, conducting and presenting their own scholarly research through the production of a research-based dissertations and publications.
vi. Foster initiative and potential for independent self-study that will develop the students' motivation and ability to continue updating their knowledge and skills after completion of the course of study in relation to scholarship and research.
vii. Enable the students to be able to demonstrate a critical awareness and reflection on research-based information as a basis for problem solving and practice in professional contexts.
viii. Enable students to be able to demonstrate ability to interpret and report research findings in areas relevant to software engineering.

ix. Enable students to be able to demonstrate the ability to formulate research questions and problems, design and carry out their own small scale research projects and present their findings orally and in writing.
x. Equip students with research and publication skills to enable them publish research from high quality dissertations in reputable journals and/ or presentation of their research findings at academic conferences.

1.3 Justification

Software has become the driving force behind most new technologies. But the engineering of software is becoming increasingly complicated. Moreover we have seen an increased reliance on software systems for small to medium companies. This has been as a result of increased access to computer and Internet. There is also need to revise curriculum to provide students with the research skills to advance software systems. It is not to forget also that Makerere University is working hard towards establishing a research led institution that will attract learners from all corners of the world. The design of the new curriculum has given a special emphasis on this university mission.

Further, the rate of growth of Information and Communication Technology (ICT) in Uganda in particular and the African region in general is enormous. In order to sustain the high growth useful to the economy, there is need for highly skilled and specialized ICT labor force to cater for the sophisticated ICT-jobs. Today Makerere University Faculty of Computing and Information Technology (CIT) is the main ICT training, research and consultancy centre in Uganda.

1.4 Collaboration Partners on PhD in Software Engineering

1.4.1 University of Groningen, Radboud University Nijmegen and Eindhoven University of Technology

The Netherlands Government through the Netherlands Organization for International Cooperation in Higher Education (Nuffic) provided a 5.7 million euro grant for a project on 'Strengthening ICT Training and Research Capacity in the Four Public Universities in Uganda'. This project commenced on 1st June 2007 and will end on 31st May 2011. One of the objectives is to build ICT human resource capacity through staff development and implementation of graduate programmes (M.Sc. and Ph.D.) and 30 PhD students

(10 registered at the above institutions in the Netherlands and 20 at Makerere University) are supervised by PhD holders from University of Groningen, Radboud University Nijmegen, Eindhoven University of Technology and Makerere University with support from the project.

Out of the 5.7 million Euros about 2.5 million Euros is to support 10-15visits by Professors from the Institutions in Netherlands per year in a bid to support training and research in Uganda.

1.4.2 University of Bergen

On 18th November 1999 a frame agreement on research collaboration, scientific competence building, student and staff exchange, and institutional development was signed between University of Bergen and Makerere University in Kampala, Uganda. The agreement has a time frame of fifteen years.

Makerere University Faculty of Computing and Information Technology has an active student and staff exchange with the Department of Informatics and the Department of Information Science and Media at the University of Bergen (UiB) under this collaboration agreement. The staff from UiB have over the years conducted lectures in areas where the Faculty of Computing and Information Technology lacks local expertise.

1.4.3 London South Bank University

In 2005 Makerere University and London South Bank University signed a Memorandum of Understanding (MOU) in which the two universities agreed to:

(a) Develop joint degree programmes (Masters Level) in the following areas: M.Sc. Information Systems; and M.Sc. in Human Resources (International).
(b) To look at the feasibility of developing a distance learning PhD programme to include a cost model and that the programme will be designed with the view of implementation in the Faculty of Computing and Information Technology, Makerere University initially and then extended to the rest of the University in due course.
(c) Identify and seek funding for PhD studentships from the Common Wealth Scholarship Fund, British Council and other funding bodies.

(d) Explore various avenues for research funding, which particularly focus on the development needs of Uganda.
(e) Identify ways in which best practice can be shared in the areas of Teaching and Learning.
(f) Collaborate on quality assurance whereby London South Bank University will develop a proposal and costing model to help Makerere University develop mechanisms and procedures to support effective quality assurance and research monitoring at both institutional and subject levels.

A lot has been achieved under the MOU between Makerere University and London South Bank University that is still in force.

1.5 Computing Equipment

The Faculty of Computing and IT has put in place specialized research laboratories (e.g. the Multimedia Laboratory, Geographical Information Systems Laboratory, Mobile Computing Laboratory, Networking and Systems Laboratory, Software Incubation Laboratory, Computer Engineering Laboratory and E-learning Laboratory) and plans are under way to establish more laboratories using funds available under donor funded projects and internally generated funds. For example, under the project on 'Strengthening ICT Training and Research Capacity in the Four Public Universities in Uganda' there is approximately 800,000 Euros reserved for specialized equipment and software for the Faculty for Computing and Information Technology Centre of Excellence. This specialized equipment and software will be availed to the PhD students and their supervisors.

Every PhD student in the Faculty of Computing and Information Technology is given a laptop and personal computer for the whole duration of the programme. Each member of academic staff has a laptop and personal computer in the office.

1.6 Physical Facilities

The Faculty has sufficient offices for both staff and PhD students, lecture rooms, seminar rooms and computer laboratories in the faculty buildings.

1.7 Financial Resources

Tuition fee per student shall be 3,000,000 Uganda Shillings per annum for Ugandans and 3,000 US Dollars per annum for Non-Ugandans.

2 REGULATIONS

2.1 Entrance Requirements

To qualify for admission, a candidate must fulfil the general Makerere University entry requirements for a doctoral programme. In addition, to be admitted to the PhD (Software Engineering) a candidate must be a holder of a master's degree in software Engineering, Computer Science, Software Development or its equivalent.

2.2 Duration

The duration of the PhD Programme is three academic years (6 semesters).

2.3 Credit Units (CU)

The weighting unit is a credit unit. One credit unit is one contact hour per week per semester. One contact hour can be defined as equivalent to 2 tutorial hours or 2 practical hours.

2.4 Core and Elective Courses

A major is the subject/ field/ programme of specialization. A core course is compulsory course for the major and an elective course is an optional course for the major.

2.5 Graduation Requirements

To qualify for the award of the degree of Philosophy in Software Engineering a candidate is required to obtain a minimum of 18 credit units for courses passed including all the compulsory courses and the PhD Dissertation within a period stipulated by Makerere University Senate/ Council.

Let LH, CH, and CU stand for Lecture Hour, Contact Hour, and Credit Unit respectively.

2.6 Fundamentals of the revised program

This programme aims at addressing the human resource needs of the region in the areas of software engineering. By the nature of the programme, a data bank of mainly journal papers available to all researchers were and are still being created, which are updated yearly as new and continuing students review literature.

Fundamental changes in the revised program are as follows:

(a) The semester load in first year has been reduced from 12 CU to 9 CU to allow students more time to undertake individual study.
(b) The PCS 9100- Philosophy of Computing and IT has been merged with MCS 9200- Philosophy of Science and Computing Research to form PCS 9100 Philosophy of Computing.
(c) The course MCS 9105 Gender and ICT and MCS 9103 Managerial Problems in IT have been dropped. Some concepts of Managerial problems in IT have been integrated in MCS 9200 Research Project Management. Gender will be integrated in the teaching of all the courses as a cross-cutting issue.
(d) Three new courses that target advanced software engineering research and development roadmap have been introduced: Science of Programming, Models of Software Systems, Architectures of software Systems.
(e) The total duration of the course has been increased from three (3) to four (4) years.

2.7 The curriculum for Doctor of Philosophy (Software Engineering)

Code	Name	Comment
	Semester 1	
PIS 9101:	Presentations, Scientific Writing and Research Ethics	Revised
PCS9101	The Philosophy of Computing	Revised
PSE 9102	Science of Programming	New
	Semester 2	
PSE 9201	Models of Software Systems	New
PIT 9201	Advanced Research Methods	Revised
PSE 9203	Software Systems Architectures	New

2.8 Grading of Courses

a) Each Course will be graded out of a maximum of 100 marks and assigned an appropriate letter grade and a grade point as follows:

MARKS	LETTER GRADE	GRADE POINT	
90 - 100	A+	5.0	Exceptional
80----89	A	5.0	Excellent
75 – 79	B+	4.5	Very good
70 – 74	B	4.0	Good
65 – 69	C+	3.5	Fairly good
60 – 64	C	3.0	Pass
55 – 59	D+	2.5	Marginal fail
50 – 54	D	2.0	Clear fail
45 – 49	E	1.5	Bad fail
40 – 44	E-	1.0	Qualified fail
Below 40	F	0.0	Qualified Fail

b) The following additional letters will be used, where appropriate: -

W - Withdraw from Course;
I - Incomplete;
P - Pass;
F - Failure.

2.9 Minimum Pass Mark

A minimum pass grade for each course shall be 3.0 grade points.

2.10 Calculation of Cumulative Grade Point Average (CGPA)

The CGPA shall be calculated as follows: -

$$CGPA = \frac{\Sigma^{n}_{i=1}(GP_i * CU_i)}{\Sigma^{n}_{i=1} CU_i},$$

Where GP_i is the Grade Point score of a particular course i;

CU_i is the number of Credit Units of course i; and
n is the number of courses so far done.

2.11 Progression

Progression through the programme shall be assessed in three ways:

2.12 Normal Progress

This occurs when a student passes each course taken with a minimum Grade Point of 3.0.

2.13 Probationary

This is a warning stage and occurs if either the cumulative grade point average (CGPA) is less than 3.0 and/ or the student has failed a core course. Probation is waved when these conditions cease to hold.

2.14 Discontinuation

A student shall be discontinued from the program if

i. He/she fails to get a grade point of at least 3.0 from any course unit for three sittings
ii. By the end of the third semester, he/she does not have an approved research proposal
iii. Without a credible reason, he/she fails to submit the two 6 monthly consecutive progress reports
iv. The candidate shows no substantial progress for two academic years
v. Overstays on the program for more than two years
vi. Fails to pass on the third submission of the dissertation

2.15 Re-taking a Course

A Student may re-take any course when it is offered again in order to pass if the student had failed the course. A Student may take a substitute elective, where the Student does not wish to re-take a failed elective.

2.16 PhD Dissertation

Students are required to demonstrate their ability to independently formulate a detailed dissertation proposal, as well as develop and demonstrate their dissertation thoroughly.

a. A candidate shall be allowed to formally start on the dissertation after registration.
b. A candidate shall submit a dissertation proposal to the Faculty of Computing and Information Technology Higher Degrees Committee during the first semester of the first academic year.
c. The candidate shall execute the dissertation after acceptance of the dissertation proposal.
d. The candidate shall submit a dissertation report before the end of the third year (6^{th} semester).

2.17 Passing of a Dissertation

To pass the Dissertation, the candidate shall satisfy the Internal Examiner, External Examiner, and Viva Voce Committee independently.

2.18 Revised Dissertation

A candidate, who fails to satisfy the examiners, shall re-submit a Revised Dissertation in accordance with the standing University guidelines for the PhD dissertation examinations.

3 THE PROGRAMME

3.1 Summary of Curriculum

	Semester I			
Code	Name	LH	CH	CU
	3 core courses			
PIS 9101:	Presentations, Scientific Writing and Research Ethics	45	45	3
PCS9101	Philosophy of Computing	45	45	3
PSE 9102	Science of Computer Programming	45	45	3
	Semester II			
	3 Core Courses			
PIT 9201	Advanced Research Methods	45	45	3

PSE 9203	Software Systems Architectures	45	45	3
PSE 9201	Models of Software Systems	45	45	3
	Semester III ,IV, V, VI, VII and VIII			
	Independent Research, Publication and Dissertation Compilation			

4 DETAILED CURRICULUM

4.1 PIS 9101: Presentations, Scientific Writing and Research Ethics (3CU)

(a) Description: Most PhD students struggle with scientific writing and presentations in English, and normally much time in a PhD study is spent revising papers and preparing for conference talks. Given the amount of time that PhD students spend writing and preparing to present, students should invest in a systematic study of scientific writing and presentations. The course deals with the publication process from the perspectives of the author of a scientific paper and the editor of a scientific journal. It is intended for PhD students in the fields of computing and Information technology, engineering and natural sciences.

b) Aims and objectives:

The aim is to give the participants the following :

- awareness of the importance of scientific writing,
- motivation to write scientific papers, and
- prerequisites for publishing in first-class scientific journals.

(c) Learning outcomes:

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

- Make a quality conference presentation
- Write a quality journal article
- Appreciate ethics-related issues when writing a scholarly/scientific paper.
- Understand the prerequisites for choosing the market for publishing

d) **Teaching and learning patterns**:

Classes are held as a group discussion. Reading material which includes books and journal papers on scientific writing and ethics are distributed a week in

advance, and students take it in turns to research and present. The students are also given reading material on how to make excellent presentations. The lecturer addresses questions to the students to encourage them to think about and understand the material. The classes will also include viewing of recorded seminar presentations by leading academics in the field.

e) Indicative content:

- Science and writing. Reports and scientific publications. The IMRAD format. Scientific journals. Why, what, when, with whom and where publish?
- Structure of a scientific paper. The different parts of a scientific paper. Language and style. The publication process. Writing a paper. Dealing with editors, reviewers and publishers.
- Critical review of scientific papers by groups of participants.
- General principles of expository writing, pre-writing and planning. Typical formats, structure and language for scientific writing, emphasis on scientific articles as published in (primary) international scientific journals. English grammar essential to scientific papers. Designing tables, figures and graphs for scientific papers. Good style for readability. The refereeing and publishing process, what referees are looking for, how to deal with editors. Paragraphing, linking paragraphs to make the logic clear. Writing informative abstracts and crafting clear titles.
- Ethics: Honesty and credibility in scientific writing.

f) Assessment :

Progressive assessment will be based on the quality of presentations in class by each student. The final assessment will be based on a scientific paper

Review paper 60%, Presentations 40%.

g) References:

(i) How to write and publish a scientific paper, Robert A. Day and Barbara Gastel, ISBN:0-313-33027-1, 6TH Edition, 2006.

(ii) Research ethics, edited by Anna Smith IItis, 1st Edition, 2006.

(iii) The student's guide to research ethics, Oliver, 2003.

4.2 PCS 9101: Philosophy of Computing (3CU)

(a) Description:

This course explorer the philosophical foundations of the computing field. It explores the computational understanding of the major parameters that make up and support the computing field. It explores their foundations and philosophical underpinnings.

(b) Aims and Objectives

The aims of the course are:

- To give students an avenue of exploring the philosophical foundations of computing as an academic field
- To give students the historical foundation of computational thinking and interpretation
- To expose students to the philosophical thinkings of the different areas of computing

(c) Learning Outcomes

By the end of the course, the students should be able to:

e. Explain the philosophical foundations of computing

f. Explain the foundations of theoretical thinking and interpretations

g. Explain the philosophical thinkings of the different fields of computing

(d) Indicative content

- Mind and Artificial Intelligence (AI): The philosophy of artificial intelligence and its critique, computationalism, connectionism and the philosophy of mind
- Real and virtual worlds: Ontology, virtual reality, the physics of information, physics as a traditional model of the ideal science of the philosophy of science, cybernetics and artificial life
- Language and knowledge: Information and content, knowledge, the philosophy of computer languages, hypertext.

- Logic and probability: probability in artificial intelligence, game theory – Nash equilibrium

(e) Teaching and Learning patterns

Teaching will be by lectures, group work, group discussions and presentations

(f) Assessment

Assessment will be by take-home assignments and presentations. Students will be given tasks to read and write about then present in class. The lecturer will award marks for each write up and presentation.

(g) References

1. Floridi, Luciano (1999) *Philosophy and Computing: An Introduction*. Routledge: London / New York.
2. Bynum, Terrel Ward; Moor, James H. (2000) *The Digital Phoenix: How Computers are Changing Philosophy*. Blackwell Publishers: Oxford, UK.
3. Colburn, Timothy R. (2000) *Philosophy and Computer Science*. M.E. Sharpe: Armonk, NY, USA.

4.3 PSE 9102: Science of Programming (3CU)

a) Course Description

This course introduces foundational concepts and techniques of programming languages. We use typed λ-calculi and operational semantics as models of programming language concepts. These models are applicable to the design, analysis, and implementation of programming languages. We demonstrate the utility of a mathematical approach to programming languages in answering questions about program correctness, the pro's and con's of various languages, compiler correctness, and other practical issues. We focus on two of the most successful styles of semantic description: denotational and operational. We deal with small "core" languages, each chosen to illustrate a specific paradigm. We use semantics to prove properties of a language, to analyze programs, to design correct programs, to prove correctness of compiler optimizations, and to prove general laws of program equivalence.

b) Aims

The objective is to

- to study formal techniques for describing computation and compilation.
- Provide a more general understanding of programming languages, specification, logic, mathematics, and proof theory.
- apply formal reasoning to nondeterministic programs and to concurrent programs, and provide an introduction to reasoning about distributed systems (temporal logic).

c) Learning Outcome

At the end of the course students will be able to: describe and relate different programming paradigms and the mathematical models on which they build; select appropriate methodology to use in the final research work and dissertation.

d) Indicative Content

- Inductive definitions.
- Static and dynamic semantics.
- Type safety.
- Function, product, and sum types.
- Universal types and polymorphism.
- Existential types and data abstraction.
- Recursive types.
- Object types.
- Subtyping.
- Equational reasoning.
- Type inference and unification
- denotational and operational, referential transparency, criteria for choosing models
- Sequential imperative programs: state transformations, partial and total correctness, traces and runtime

- Machine language: jumps and continuations, compiling sequential pro-grams, correctness of compiler optimizations
- Parallel programming: data ow networks, shared-memory parallelism, communicating processes, safety and liveness, fair execution
- Functional programs: types and polymorphism, call-by-value, direct-and continuation-style semantics

e) Learning and Teaching

Classes are held as a group discussion. Reading material which includes journal papers is distributed a week in advance, and students take it in turns to research and present new topics. The lecturer addresses questions to the students to encourage them to think about and understand the material. The lecturer should become aware of the students' proposed topics of research so that the discussion explores how the principles in the course apply to these topics. The students make presentations of their review paper for critique from both the students and the lecturer.

f) Assessment

- Review paper 100%

g) Reading

1. John C. Reynolds. Theories of Programming Languages. Cambridge University Press, 1999.
2. Glynn Winskel. The Formal Semantics of Programming Languages. MIT Press, 1993.
3. John C. Mitchell. Foundations for Programming Languages. MIT Press, 1996.
4. Martin Abadi and Luca Cardelli. A Theory of Objects. Springer-Verlag, 1996.
5. Jean-Yves Girard. Proofs and Types. Cambridge University Press, 1989.

4.4 PIT 9201 Advanced Research Methods (3 CU)

(a) Course Objectives:

The objectives of this course are to provide:

- Philosophical underpinnings of research in computing and IT

- Practical aspects on doing research

(b) Learning outcome:

At the end of the course the students will be able to apply computing and IT research methods in their research

(c) Course Content:

The first part of the course is devoted to the philosophical underpinnings of research, which crucially influence choice of research methods and interpretations of data. The course then moves on to the more practical aspects of 'doing research' - looking at developing a research strategy as well as ways of collecting data, analysing data and communicating research findings. This course will also give guidance to students on how to identify a research problem. Students will be presented with various research paradigms and models of methodology and assisted with designing an appropriate method for their research. Students will be trained in the analysis and presentation of results, exposition of processes and methods used and conclusions drawn.

Key philosophical and epistemological bases for research are explored, and alternative methodologies are examined in relation to varied theoretical approaches. Selected sets of methods and techniques are critically appraised, while the range and scope of techniques with which students are familiar is extended. The structure of the course aims to achieve a balance between theory and practice. Considerable emphasis is therefore placed upon the logistics of setting-up, doing and disseminating research. The course not only introduces a range of research ideas and skills central to sound socio-environmental enquiry in general, but also acts as a critical and practical research forum where discussion and preparation for the PhD dissertation takes place.

(d) Teaching and Learning pattern:

Classes are held as a group discussion. Reading material which includes journal papers is distributed a week in advance, and students take it in turns to research and present new topics. The lecturer addresses questions to the students to encourage them to think about and understand the material. Each student undertakes a review of the different research methodologies and makes a presentation before the class. The students will identify

researchable problems from which they will apply the concepts taught in class with an aim of producing research proposals by the end of the semester. The students will be required to build on their proposals on a weekly basis in line with the new concepts that will be taught. The students will make presentations of their draft proposal for critique and feedback from both the students and the lecturer.

(e) Assessment method:

Evaluation shall be based on a research proposal produced by the end of the semester. Research proposal 100%.

(f) References

1. Qualitative research and evaluation methods; By Michael Quinn Patton; Edition: 3, illustrated; Published by SAGE, 2002; ISBN 0761919716, 9780761919711; 598 pages.
2. Research Design & Statistical Analysis; Third Edition; By Jerome L. Myers, Arnold D. Well, Arnold D. Well, Robert F. Lorch, Jerome L. Myers; Pages: 736; Published by: Routledge; Publication Date: 1st November 2002; ISBN: 978-0-8058-4037-7
3. *Are Your Lights On? How to Figure out what the Problem Really Is,* by Donald C Gause and Gerald M Weinberg, Dorset House, USA, 1990. A brilliant book about getting ready to make decisions.
4. Bordens, K.S. & Abbott, B.B. (1988) Research design and methods: A process approach. Mayfield.

4.5 PSE 9201: Models of Software Systems (3CU)

a) Course Description

Scientific foundations for software engineering depend on the use of precise, abstract models for characterizing and reasoning about properties of software systems. This course considers many of the standard models for representing sequential and concurrent systems, such as state machines, algebras, and traces. It shows how different logics can be used to specify properties of software systems, such as functional correctness, deadlock freedom, and internal consistency. Concepts such as composition mechanisms, abstraction

relations, invariants, non-determinism, inductive definitions and denotational descriptions are recurrent themes throughout the course.

b) Aims

By the end of the course you should be able to

- understand the strengths and weaknesses of certain models and logics, including state machines, algebraic and trace models, and temporal logics.
- to select and describe abstract formal models for certain classes of systems. to reason formally about the elementary properties of modeled systems

c) Learning Outcome

At the end of the course students will be able to: describe and relate different models of software systems; select appropriate methodology to use in the final research work and dissertation.

d) Learning and Teaching

Classes are held as a group discussion. Reading material which includes journal papers is distributed a week in advance, and students take it in turns to research and present new topics. The lecturer addresses questions to the students to encourage them to think about and understand the material. The lecturer should become aware of the students' proposed topics of research so that the discussion explores how the principles in the course apply to these topics. The students make presentations of their review paper for critique from both the students and the lecturer.

e) Indicative Content

- what is a model?
- Foundations Logic, Proof Techniques
- Sets, Relations, Functions, Proof Techniques
- State Machines ,Variations , FSP and LTSA , Reasoning about State
- Machines
- Z Techniques
- Refinement & Abstraction

- Modeling Concurrency in FSP , Modeling Techniques, Reasoning about Concurrency,
- Model Checking Linear Temporal Logic, Promela/Spin
- Petri Nets
- UML

f) Assessment

- Review paper 100%

g) Reading

1. Concepts and Notations for Concurrent Programming," Andrews and Schneider. Computing Surveys, Vol. 15, No. 1, March 1983.
2. "Formal Methods: State of the Art and Future Directions", ACM Computing Surveys, Vol. 28, No. 4, December 1996, pp. 626-643. Available as CMU-CS-96-178.
3. "Statecharts: a visual formalism for complex systems." D. Harel. Science of Computer Programming, 8:231-274, 1987.
4. "FAA En Route Resectorization - A Formal Specification." V.J. Harvey, and P.R.H Place. Unpublished manuscript, September 1999.
5. "Coloured Petri Nets: A High Level Language for System Design and Analysis." K. Jensen. In High-level Petri Nets: Theory and Application. K. Jensen and G. Rozenberg (eds.) Springer-Verlag, 1991.
6. "Temporal Logic." Draft version of chapter from book in preparation. 1996.
7. Concurrency: State Models and Java Programs. J. Magee and J. Kramer. Wiley, 1999.
8. "Petri Nets." J. L. Peterson. ACM Computing Surveys, Sept 1977.
9. Software Engineering Mathematics. J. Woodcock and M. Loomis, Addison-Wesley 1988.

4.6 PSE 9203: Software Systems Architectures (3CU)

a) description

This course covers the state-of-the-art in architectural design of complex software systems. The course considers commonly-used software system architectures, techniques for designing and implementing these architectures, models and notations for characterizing and reasoning about architectures, and case studies of actual software system architectures.

b) Aims

I. Understand the fundamental concepts of architectural software design and analysis.
II. Demonstrate the knowledge of the existing software system architectures.
III. Demonstrate the knowledge of developing software system architectures in a software design project.
IV. Demonstrate the knowledge of analyzing software system architectures.
V. Demonstrate the knowledge of software architecture patterns and their application.

c) Learning Outcome

At the end of the course students will be able to: describe and relate different software architectures; select appropriate methodology to use in the final research work and dissertation.

d) Learning and Teaching

Classes are held as a group discussion. Reading material which includes journal papers is distributed a week in advance, and students take it in turns to research and present new topics. The lecturer addresses questions to the students to encourage them to think about and understand the material. The lecturer should become aware of the students' proposed topics of research so that the discussion explores how the principles in the course apply to these topics. The students make presentations of their review paper for critique from both the students and the lecturer. .

e) Indicative Content

- Introduction to Software Architectures

- Modular design; Object-Oriented design
- Design patterns; Object-Oriented design patterns
- Pattern-oriented software architectures
- Interactive Systems architecture
- Layered architecture
- Pipes and Filters architecture
- Model Driven Architecture
- Repository architecture
- Event-based architecture
- Adaptable systems architecture
- Distributed systems architecture
- Client-server architecture
- Fault-tolerant architecture
- Process control systems architecture
- Domain-specific architectures; Reference architecture

f) Assessment

- Review paper 100%

5 QUALITY ASSURANCE

To ensure the quality and relevance of the revised program, various state holders were consulted. These include students, local private and public sector and some international institutions. Students were consulted through questionnaire during their final examination in the last semester of 2007/2008 academic year. The response collected from students indicated lack of research assessments and insufficient hands-on and transferable skills. The revised assessment methods are aimed to deliver these learning outcomes. Local institutions consulted include Uganda Communication Commission (UCC), DICTS, Ministry of ICT, and Faculty of Technology. We received constructive comments from DICTS and UCC, which we have implemented. In addition, through the first satellite workshop organized by the faculty of Computing and IT in 2008, a series of consultation with local Telecommunications providers and International companies such as Google, Nokia, Gramen foundation, etc were conducted regarding an option on Mobile Computing and Software Application

Development. It was decided on the need to have such a program and Nokia. A paper of the proposed mobile computing and software development option was presented in Brasil during W3C conference by professor Fisseha of the Faculty of Computing and IT. The initiative received a very positive feedback.

The faculty has a QA committee that monitors the teaching delivery and follows up students feedback and complaints. The faculty QA committee collects data itself from class rooms. In addition, it continuously collects some information from head of departments and students. In its monthly meetings, the committee draws action plans that are followed at different levels; at departmental and faculty management level. In most cases the follow-up of action plans was very effective.

The monitoring of course delivery has proved very efficient way in sustaining quality of teaching delivery in the faculty. It disclosed some irresponsible teaching practices that the faculty dealt with in time.

6 LOAD DISTRIBUTION

Code	**Name**	**Assessment Method**	**Staff**
PSE9102	Science of Programming	Review Paper 100%	Dr. John Quinn
PCS9101	The Philosophy of Computing	Review Paper 100%	Dr. John Ngubiri
PSE 9201	Models of Software Systems	Review Paper 100%	Dr. Benjamin Kanagwa
PSE 9203	Software Systems Architectures	Review Paper 100%	Dr. Jose Quenum
PIS 9101	Presentations, Scientific Writing and Research Ethics	Review Paper 100%	Dr Agnes Rwashana
PIT 9201	Advanced Research Methods	Research Proposal 100%	Dr Josephine Nabukenya