Scientific Theory in Informatics

IT731A / IT919F

Course Handbook

UNIVERSITY OF SKÖVDE

School of Informatics

Table of Contents

1.0 Introduction

The goal of this course is to impart an understanding and working knowledge of the theoretical foundations of a representative range of the constituent sub-disciplines of informatics. This involves the study of the scientific method, modelling methodologies, core theories, and fundamental techniques, and their application in the analysis and design of computer-based information systems. In the process, the course seeks to instil an appreciation of how theory influences practice in creating effective, efficient, and useful people-centred computer-based information systems.

Upon successfully completing the course, a student will have achieved the following learning outcomes and will be able to:

1. Explain the scientific method of problem abstraction, hypothesis formation and test, experimentation, and analysis;
2. Explain the distinction between modelling methodology and instances of specific theories and models;
3. Explain the difference between descriptive and normative models;
4. Apply a working knowledge of a representative sample of core theories and fundamental techniques in informatics, in general, and in computational, cognitive, and socio-technical systems, in particular;
5. Compare and contrast competing theories and complementary techniques in the context of typical computer-based information systems.

2.0 Course Content

The broad area of computing is traditionally broken down into different functional sub-disciplines such as computer engineering, computer science, software engineering, information systems, and information technology. Informatics takes a different approach, dealing with the design and development of systems that provide information for individuals, organizations, and society in a timely, effective, and efficient manner. It focuses on how this information is represented, processed, and communicated in natural and artificial systems. As a result, it treats the various facets of computing as a cohesive whole, addressing it from three complementary perspectives: computation, cognition, and socio-technology.

The computation perspective deals with the usual mathematical and engineering issues that are needed to design and implement a given computer application. The cognition perspective deals with how people & computer systems solve problems, make decisions, and learn. The socio-technology perspective puts both of these together in the context of organizations and focuses on the ways people can leverage benefit from the information in these systems in support of their organizational goals.

Each of these perspectives draws on a broad body of knowledge and principals that can be categorized under three headings:

1. Modelling methodologies
2. Core theories
3. Fundamental techniques

Modelling methodologies address the scientific foundations of a perspective or set of theories, including the scientific method and the manner in which scientific theories are formed, formulated, and adopted, as well as how they may be used. Modelling methodologies don't address specific scientific domains, but do influence what a theory can and cannot claim. Since a theory is effectively a model with some degree of quantitative and qualitative formalism, this level of abstraction addresses the different types of model and different approaches to modelling.

Core theories (e.g. complexity theory in computational systems or organizational theory in socio-technical systems) address particular domains but in a manner that has general applicability.

Fundamental techniques are more focused on specific methodologies, mechanisms, and algorithms for bringing about a required result within any given core theory. They provide a way of realizing a given computational theory, for example, in explicit terms that can subsequently be implemented as an operational information system or a way of producing a specific model of some product, process, or organization. Despite the fact that they are typically associated with a given theory, fundamental techniques still have general applicability across a range of domains.

3.0 Student Instruction

Students on this course are drawn from many different backgrounds in computing and have a minimal amount of shared knowledge. Given the diversity of the material to be taught, it is likely that each student will be challenged in different ways. In order to ensure that no student falls behind early on, the course will increase the rate of learning progressively. To achieve this, the course begins with an introduction to informatics as a discipline and then presents a preview of the material that comprises the remainder of the course. It then progresses to a more detailed treatment of each topic individually, and finishes with an in-depth treatment of certain topics on a student-by-student basis. The first three phases are implemented through classroom instruction (lectures and seminars); the fourth phase though individual supervision.

The nature of informatics as a team-based discipline is also reflected in the delivery, having the students work in groups in seminars and assignments so that they learn how to collaborate effectively. Since peer assessment and review is also an important and pervasive process in today's workplace, relevant exercises also included in the seminars and tutorials.

3.1 Lectures

Between ten and fifteen lectures, each lasting one hour and forty-five minutes, are typically scheduled for formal teaching. The lectures cover:

- an introduction to informatics, top-level overview of the course material, and scientific foundations of informatics
- core theories and fundamental techniques in each of the three perspectives of informatics: computation, cognition, and socio-technology.

A timetable for the delivery of the course, showing the time allocated to lectures, seminar, tutorials, and assessment, is provided on the course website.

Since this course addresses the interplay between theory and practice, the introductory lecture will address the practical essentials of informatics: its purpose and domain of operation, its component disciplines, the basic concepts of data and information, the system development lifecycle, and how theory supports practice in various aspects of this lifecycle. It will also give a top-level overview of all the course material - an explanatory preview of what material students will be meet and an opportunity to show how the material relates to the course objectives, the learning outcomes, and how the material fits together in one coherent whole.

3.2 Seminars

There are typically between four and six seminars, shared between the three perspectives: computation, cognition, and socio-technology. Seminars cover practical applications of the theories covered in the lectures.

Prior to each seminar, the students work in small groups on exercises or tasks set in the topic lectures. Typically, these assess the use of a particular theory or technique in a given application domain. Alternatively, it might be a comparative analysis of two or three complementary theories or techniques, based on a small amount of supplementary reading (typically a tutorial article, review, or survey). The goal is to provide the students an opportunity to assess different approaches based on the material covered in the lectures and to generate a synthesis of their relative strengths and weaknesses in a given context.

Groups of 3-4 students will be formed to address the exercises and tasks before the seminar and respond with a presentation. During the seminar, one or more groups will be selected to give its presentation. After each presentation, there will be a short class discussion, moderated by the instructor. This provides a way of allowing the students to validate each other's understanding and refine it as a result of the feedback from classmates and the instructor.

A complete set of presentations for the seminars from each group will form the basis of the assignment hand-in.

3.3 Tutorials

Students should expect some individual help with their case studies from a suitable member of the course team.

4.0 Student assessment

The degree to which the student achieves the required learning outcomes of the course is assessed through an assignment and a case study. Of the 7.5 hp course load, 2.5 are allocated to the seminar assignment, and 5.0 hp to the case study.

4.1 Seminar Assignment (group)

The assignment summarises the seminar work, addressing all three perspectives in the body of knowledge: computation, cognition, and socio-technology. The goal of the assignment is to show that students have understood and can apply a varied sample of modelling methodologies, core theories, or fundamental techniques. This is a horizontal theoretically-oriented study, with the goal of showing how theory can support effective practice in informatics. The assignment is conducted in seminar groups - groups will be formed with a view to ensuring that members of a group have complementary backgrounds. Groups should prepare a compilation of their seminar presentations covering all the seminar tasks and exercises. These may be amended and improved in the light of seminar feedback. Some additional written explanation is also acceptable where necessary (but not expected or obligatory). Groups should also provide a matrix showing clearly which of its members were involved in the preparation of which slides, signed by each of the group members. The seminar compilation should be organised as one single .pdf file for submission.

4.2 Case Study (individual)

The case study provides the student with an opportunity to use the material they have learned in the course in the analysis and appraisal of a computer-based information system in a more challenging context than the assignment. It complements the assignment in that it addresses a narrower spectrum of issues in just one of the perspectives of informatics, but it does so at a deeper level by assessing two or more possible modelling methodologies, core theories, or fundamental techniques, each of which is a provides a potential solution strategy for one specific problem in the development of the information system. Thus, the case study is a vertically-oriented investigation in contrast to the wider-ranging horizontal investigation associated with the assignment. The objective of the assignment is to employ an in-depth understanding of a particular area of theory in the development of a software system or digital artefact. The system or artefact to be developed will normally be defined by the student and reflect the content of their own research, though the case study must be presented the form specified for this course. The goal is to perform a critical appraisal and comparative analysis, assessing the strengths and weaknesses of two or more competing solutions strategies (modelling methodologies, core theories, or fundamental techniques) drawn from the material covered in class and deployed in either the computational, cognitive, or socio-technical aspect of that application area. It should also be clear how the chosen theories contribute to the building of the system or artefact.

The assignment and the case study are partially complementary, with the latter providing an opportunity for more detailed analysis and focussing on just one of the three complementary perspectives of informatics rather than all three. The case study is carried out individually, not than in a group. It will also require the student to read beyond the material provided in the lecture notes in order to present a detailed account of the alternative theories selected for comparison.

The case study should be submitted as a .pdf file.

The assessment of the case study will not focus on the quality or extent of the student's research but on the critical appraisal of the chosen modelling methodologies, core theories, or fundamental techniques as they relate to the informatics problem being investigated in the case study.

4.5 Assessment of course learning outcomes

As set out in the Course Plan, upon completion of the course, a student should be able to:

1. Explain the scientific method of problem abstraction, hypothesis formation and test, experimentation, and analysis;

2. Explain the distinction between modeling methodology and instances of specific theories and models;

3. Explain the difference between descriptive and normative models;

4. Apply a working knowledge of a representative sample of core theories and fundamental techniques in informatics, in general, and in computational, cognitive, and socio-technical systems, in particular;

5. Compare and contrast competing theories techniques in the context of typical computer-based information systems.

Table 1 shows the tools used to assess the degree to which these learning outcomes are achieved by the student.

Tool \ **Outcome**	1	2	3	4	5
Assignment	✓			✓	
Case study	✓	✓	✓	✓	✓

Table 1: Coverage of learning outcomes by assessment tool.

4.6 Grades

Advanced level students are awarded one of three grades: a pass with distinction (VG), pass (G) or a fail (U). Research level (PhD) students receive pass (G) or fail (U) grades

Students will be awarded marks for the two modes of assessment: assignment, and case study.

The overall mark for the course is based on the weighted sum of the individual marks awarded in each assessment. The weights are equal to the hp/ECTS loading (see Table 2). Thus, the overall mark is calculated as:

$$(2.5 \times m_1 + 5.0 \times m_2) \;/\; 7.5$$

where m_1, and m_2, are the individual marks awarded for the written assignment and case study report, respectively.

The overall mark is converted to the final grade as shown in Table 3.

Note, however, that students must pass both components of the assessment. In the event that a student fails to achieve a passing grade, they need only re-submit those components of the assessment that failed. The pass mark for advanced students is 40%, for PhD (research level) students 50%

Assessment	HP / ECTS
Assignment	2.5
Case Study	5.0

Table 2: Weighting of marks.

Overall Mark	Final Grade
≥ 70%	VG
≥ 40% and < 70%	G
< 40%	U

Table 3: Mapping from overall percentage mark to final grade (advanced level students).

To facilitate consistent marking, a standard marking scheme is used for both assignment and case study. Furthermore, the case study will be marked both by the advisor and a second-reader. The final mark is the average of the two marks, provided they do not differ by more than a given amount (10%). In that case, a third reader may be asked to mark the report.

5.0 Work profile

A balance must be struck between the time allocated to formal instruction, delivered through direct contact with members of staff in lectures and seminars, and independent study, supported by members of staff outside formal course contact hours. Furthermore, the amount of material to be delivered needs to be scoped appropriately to ensure adequate coverage at the required level of detail in the available time. As already noted, the course is delivered at a 25% pace with total of approximately 200 hours of study.

An explicit allowance of time for self-study is made in the course delivery. One hour of self-study is allocated for every lecture hour. Three hours of self-study are allocated for every seminar hour to allow adequate time for preparation.

An example work profile is shown in Table 4. As noted above, this is based on the assumption that one two-hour lecture is devoted to an introduction to the course, two lectures to a top-level overview of all the course material, one lecture to modelling methodology, and four lectures in each of the three perspectives of informatics (computation, cognition, and socio-technology). This gives a total of 15 lectures and 30 contact hours. In addition, six two-hour seminars are included: two for each perspective, giving a total of 12 contact hours. Four tutorials are included at the end of the course to address concerns students may have about the course material. Allowing one hour of independent study for each lecture and tutorial and three hours of preparatory study for each seminar hour, the total amount of time devoted to formal instruction is 124 hours. The remaining time is allocated to the seminar assignment and case study, in approximate proportion to their loading of 2.5 hp and 5.0 hp.

Week	Lectures	Seminars	Seminar preparation and assignment	Tutorials	Self-study	Case Study	Total
36	4				4		8
37	4				4		8
38	4		6		4		14
39	2	2			8		12
40	4		6		4		14
41	2	2			8		12
42	2	2			8		12
43	4		6		4		14
44	2	2			8		12
45	2	2			8		12
46	2		6		4		12
47		2	6		6		14
48						10	10
49						10	10
50				2		10	12
51				2		10	12
52				2		10	12
Total	32		12	6	70	50	200

Table 4: Example work profile (in hours).