Scientific Theory in Informatics

IT706A / IT919F

Assignment

Skeleton Report and Guidelines

UNIVERSITY
OF SKÖVDE

School of Informatics

Table of Contents

Preface

This document provides some guidelines on the typical layout and content of an assignment report for Course IT706A / IT919F Scientific Theory in Informatics. In it you will find some recommendations on the length of the report and the section headings. It also provides some guidance on how to focus your report so that it addresses the assessment criteria and demonstrates that you have achieved the required learning outcomes.

As stated in the course plan and course handbook, 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.

Table 1 in the course handbook notes that the assignment is used to assess the degree to which outcomes 1 and 4 have been achieved. Therefore, it is important that the report addresses these outcomes and demonstrates that you are able to do what the outcomes say you should be able to do. In the skeleton that follows, you will find some suggestions on how to do this effectively.

Please remember that these are just suggestions. You are free to adopt or ignore them as you see fit.

1.0 Roles and contribution of each student in the group

Suggested length: 1 page.

As stated in the assignment exercise, responsibility for different aspect of the assignment is to be shared among the members of the group. You should state here what each student did and the sections of the report that were written by each student. If you can, it is a good idea to give some indication of the time spent on each component. Remember, each section in the report (and each subsection of Section 4.0) must identify the student or students responsible for writing it. Each student in the group must write at least one subsection of Section 4.0 independently.

2.0 Specification of the application scenario

Suggested length: 1-2 pages.

The application scenario and outline requirements will be provided. It is responsibility of each group to creatively augment these requirements to provide a complete specification of the problem to be addressed and the information system to be developed and deployed. Remember the goal is to showcase how theory guides practice in the design and implementation of computer-based information systems. The focus in this section is not on the solution but on the problem: the user needs (i.e. requirements) that are being addressed. Mention any special characteristics of this application and any particular problems or difficulties that it presents.

3.0 System specification

Suggested length: 3-5 pages.

Normally, after identifying the user requirements and before proceeding to the specification of the system, you would focus on modelling the problem computationally. However, the computational model and other theoretical considerations are dealt with in the next section so we skip here to the system specification. If you are unfamiliar with system development life-cycles, now would be a good time to read the short guide provided in the *Supplementary Reading* section of the course website.

The system specification states exactly what the system will do and under what circumstances it will do it. There are many aspects to the specification. For the purposes of the assignment, you should focus on just three.

1. The system functionality;

2. The non-functional characteristics;

3. The system architecture diagram.

The system functionality, sometimes referred to as the functional model, usually takes the form of a functional decomposition: a hierarchical breakdown of the major functional blocks involved in the processing, analysis, or transformation of information by the system being developed. Typically, this will be a modular decomposition of the computational model. At the very least, the functional specification should state what information or data the system takes as input, what processes that data undergoes, and what data is produced as output. Think of your system as a black box and describe what that black box does, and you won't go far wrong. Don't be tempted to say here *how* the system accomplishes all these processes; you will have a chance to address this aspect in the next section.

The non-functional characteristics refer not to what the software does but instead to the manner in which it does it: its dependability, security, composability, portability, reusability, interoperability, and re-usability, for example.[1] The non-functional characteristics very often reflects the quality of the system.

The system architecture diagram is a graphical depiction of the main functional components of the system, the flow of information between them, the flow of control between them, and the flow of information into and out of the system as a whole. In systems analysis, this architecture diagram is sometimes modelled by a level 0 data-flow diagram (also referred to as a context diagram).

In this section, you might take the opportunity to demonstrate that you can explain the scientific method of problem abstraction, hypothesis formation and test, experimentation, and analysis (Learning Outcome 1). One way to do this is to interpret the system development lifecycle as an example of the scientific method; you can do this by explaining which phase of the lifecycle corresponds to a stage in the scientific method. Another way to do it is to explain the process of computational modelling. However, it would be best to address this in the next section on the problem modelling phase of the development lifecycle.

4.0 Theory and application

Suggested length: 6-10 pages.

Drawing from the modelling methodologies, core theories, and fundamental techniques that have been covered in the course, this section should list those that relevant to the information system in particular application scenario. If you feel some are not relevant, list these too and explain why you don't think they are relevant. Be careful to justify your position well.

For each methodology, theory, and technique you select, you should identify

i. The phase of the development life-cycle to which it applies;

ii. The functional components that are most affected;

[1] In software engineering, these attributes are sometimes referred to as the '-ilities'.

iii. The reasons why each theoretical consideration is important, and the risks and potential costs associated with neglecting it.

iv. An example of how the methodology, theory, or technique could be used, addressing e.g., development strategy, design decisions, system performance, user acceptance, and organizational & social impact.

Some theories will apply to the problem modelling phase of the life-cycle, i.e. the development of the computational model which allows you to solve the particular problem(s) the system addresses (e.g. how to track people in a surveillance system, how to recognize speech in a smart phone application, how to schedule the deployment of ambulances in a hospital despatch system, how to recommend products based on user-preferences in an on-line shop). Other theories and methodologies will apply to the requirements, analysis, specification, design, implementation, test, and maintenance phases. Again, if you are not familiar with these aspects of systems development, refer to the brief guide on the website.

The assignment stipulates that each student in the group must write at least one part of this section independently. It is up to you to decide how to allocate the different modelling methodologies, theories, and techniques. One good way of doing this is to split the effort by having each student address one theory or technique from each of the computational, cognitive, and socio-technical perspectives, for example. In any case, be sure to note in Section 1 who wrote which part. Also, *group the parts written by each student together*. For example, this section might have three sub-sections, one for each student.

Each part of this section provides an opportunity to address Learning Outcome 4, i.e. the ability to 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; you don't have to do anything special here to achieve this but it may help to keep this learning outcome in mind as you write your individual part of the section.

5.0 Suggested guidelines for best practice.

Suggested length: 1-2 pages.

This section provides you with an opportunity to use your imagination and to offer some insights you might have on the best way to use a particular theory in this application domain. For example, when modelling the behaviour of a system, you might suggest ways in which a finite automaton could capture the transition of the system from state to state in response to different triggers. Remember, you need to describe five guidelines.