AUTHOR(S): JONSSON, B.T., LARUSDOTTIR, M.K., DANIELS, M., CLEAR, A., CLEAR, T. and MCDERMOTT, R.

TITLE: Aligning quality assurance at the course unit and educational program levels.

YEAR: 2016

Publisher citation: JONSSON, B.T., LAUSDOTTIR, M.K., DANIELS, M., CLEAR, A. and MCDERMOTT, R. 2016. Aligning quality assurance at the course unit and educational program levels. In Proceedings of the IEEE frontiers in education 2016 conference (FIE 2016): the crossroads of engineering and business, 12-15 October 2016, Erie, USA. Piscataway: IEEE [online], pages 1-9. Available from: https://doi.org/10.1109/FIE.2016.7757419.

OpenAIR citation: JONSSON, B.T., LAUSDOTTIR, M.K., DANIELS, M., CLEAR, A. and MCDERMOTT, R. 2016. Aligning quality assurance at the course unit and educational program levels. In Proceedings of the IEEE frontiers in education 2016 conference (FIE 2016): the crossroads of engineering and business, 12-15 October 2016, Erie, USA. Piscataway: IEEE [online], pages 1-9. Available from: https://doi.org/10.1109/FIE.2016.7757419.

Aligning Quality Assurance at the Course Unit and Educational Program Levels

Björn Þór Jónsson,[1] Marta Kristín Lárusdóttir,[2] Mats Daniels,[3] Alison Clear,[4] Tony Clear,[5] Roger McDermott[6]
[1]Reykjavik University, Reykjavik, Iceland, Email: bjorn@ru.is
[2]Reykjavik University, Reykjavik, Iceland, Email: marta@ru.is
[3]Uppsala University, Uppsala, Sweden, Email: mats.daniels@it.uu.se
[4]EIT, Auckland, New Zealand, Email: aclear@eit.ac.nz
[5]Auckland University of Technology, Auckland, New Zealand, Email: tony.clear@aut.ac.nz
[6]RGU, Aberdeen, Scotland, Email: roger.mcdermott@rgu.ac.uk

***Abstract*—Quality assurance is a subject that has grown dramatically in importance in recent times. In previous work, we have described how the ACM Curricula can be used to support the Quality Assurance process of educational programs, using the Computer Science program at Reykjavik University as an example. Faculty members and employers of graduates participated in the process, that resulted in providing both detailed quantitative data and qualitative information. The assessment also raised awareness of how abstract topics and learning outcomes from an international standard can be used when revising the curricula of a particular course in a CS program. Quality assurance is indeed a continuous process, where the results of evaluations should be used to drive improvements. In this paper we focus on how a Database course was re-structured based on a recent quality assurance process.**

***Index Terms*—Educational Quality Assurance, Curriculum Design, ACM Curriculum, Database Course.**

I. Introduction

Quality assurance is a subject that has grown dramatically in importance in recent times [1], [2], [3]. The last fifty years have seen the higher education sector undergo a period of accelerated change in response to a range of diverse societal pressures. Political demands to widen democratic access to higher education, increased insistence on universities to act as catalysts for national economic growth, and the desire to change the perceived elitist nature of some institutions, have resulted in massive changes to the education system on a global scale [4], which have become "one of the defining features of the late 20th and early 21st centuries" [5]. Western Europe, USA and Australasian countries have seen a significant expansion in the number of students entering university, accompanied by an increase in the social, cultural and academic diversity of the entry cohort. In almost every developed country, the response of the university sector to these changes has been to increase substantially the number and type of higher education institution. This has led to a transformation of the perceived role of the university within society and a marked change in social expectations for the sector with, e.g., greater public scrutiny of funding arrangements [6].

In the context of national university systems funded primarily through semi-independent government agencies (such as within the Nordic countries, the UK and Western Europe) this accountability for allocated resources takes a variety of forms but inevitably includes regular institutional-level and discipline-level audits of performance. Review of quality within a discipline such as Computer Science (CS) is very much dependent on the specifics of national quality assurance processes [7] but similar approaches have become more common due to the importance of factors such as globalisation and international harmonisation frameworks such as the Bologna process. One such approach is the use of international subject benchmarks which has the attractive feature that it is based on a transnational definition of quality. For CS programs, the most mature of these is the ACM/IEEE computer science curricula (henceforth referred to as the "ACM Curricula") [8].

In previous work, we have described how the ACM Curricula can be used to support the quality assurance process of educational programs, using the CS program in Reykjavik University as an example [9]. Faculty members and employers of graduates participated in the process, that resulted in providing both detailed quantitative data and qualitative information. The assessment also raised awareness of how abstract topics and learning outcomes from an international standard can be used when revising the curricula of a particular course in the CS program. Quality assurance is indeed a continuous process, where the results of evaluations should be used to drive improvements. The main focus of this paper is on how a specific course unit can be re-designed using the outcome of such a quality assurance process, thus aligning quality assurance at the course unit and educational program levels.

The specific course in this case is the Databases course unit and the quality assurance process is the one that took place at Reykjavik University in 2013-2014. The choice of this course unit is based on identifying deficiencies in the coverage of the Information Management (IM) knowledge area in the quality assurance process. In particular, the coverage of "core" IM topics and learning outcomes was found lacking; To quote the report, the course covered "none of the four Tier 1 topics and 69% of the Tier 2 topics" [10]. (For a description of Tier 1 and Tier 2, please refer to Section III.) This deficit in the CS program was addressed by adding topics to the Databases course unit, which is a core course unit given in the third semester in the CS program at Reykjavik University.

In designing or re-designing a course following a quality assurance recommendation, there are a number of options. The first option is to base the design on the knowledge and experience of the course unit teacher, or a small group of faculty members. This may lead to a narrow course which omits topics that are generally considered important. The second option is to study and imitate similar course descriptions of other universities. There are two obstacles to this method: first, there is limited knowledge of the quality of the course design at those other universities, and second, not many detailed course descriptions are available online. A third option is to use the benchmark reference as a guideline for the course design, thus building on the accumulated work of the benchmark designers. That is the method presented here.

The process used to develop the Databases course unit was as follows:

1) Identify the suitable knowledge units from the Information Management knowledge area;
2) Turn the ACM requirements for those knowledge units into a manageable set of learning outcomes; and
3) Ensure that the teaching plan covered these learning outcomes.

This was implemented by the teacher of the 2014 instance of the Databases course unit, who revised the course unit to adhere more strongly to the ACM Curricula. The missing learning outcomes generally referred to either "social" topics, which relate to data in a social context, or recent developments in the database area, towards semi-structured and big data. The old course unit, on the other hand, focused very much on technical issues relating to software development. The goal was to increase coverage of the core topics and learning outcomes of the ACM Curricula, while retaining the strong technical focus of the course unit.

We will expand on this by first describing the course unit context within the CS program. Then, the rationale for the choice of IM topics to cover in the course unit is presented. We describe how the final learning outcomes were chosen, followed by an evaluation of the coverage. More details can be found in [11].

This paper adopts the perspective of educational quality as "production" of the curriculum [12], while acknowledging with Pears that the "service" and "student development" dimensions are also valid and important aspects of educational quality. From the production perspective, Pears has observed: "Education provision can be viewed as the process of equipping students with a set of skills and competencies. Assessment of educational structure, delivery process and content are clearly important aspects of quality assurance activity" [12]. We therefore outline how the course was implemented and describe an evaluation of how the implemented course unit conforms to the learning outcomes.

We conclude with a summary of how the development of an individual course unit was influenced by the previous quality assurance process of the education program as a whole, how additional aspects of educational quality were incorporated, and some general lessons learned that could benefit others.

II. Context of the Databases Course Unit

Reykjavik University was founded in 1998, so it is a young institution. The School of Computer Science (SCS), which was based on an existing teaching college, was one of the founding departments. Despite its young age, the SCS quickly became considered to be the strongest Computer Science (CS) department in Iceland, and has graduated the majority of all computer scientists in Iceland in the last decade.

As the largest CS program in the country, the SCS has a responsibility to prepare students both for industry and continued study. The CS program therefore has a strong emphasis on balancing theory and practice, including a strong emphasis on applied work and project work. Since the vast majority of all software projects these days build on database technology, the database course (T-202-GAG1 Databases) is given relatively early in the program.

Currently, Reykjavik University offers the following undergraduate programs in CS and related fields: BSc in Computer Science - General (180 ECTS), BSc in Computer Science - Research Based (180 ECTS), BSc in Software Engineering (180 ECTS), BSc in Discrete Mathematics and Computer Science (180 ECTS), and a Diploma in Applied Computing (120 ECTS). Furthermore, many Engineering students take some CS courses and Business and Psychology students have an option of a minor in CS.

A. Course Context and Student Preparation

The Databases course unit is 6 ECTS, corresponding to 150-180 hours of student work.[1] Teaching takes place in a twelve week long semester, where four 6 ECTS courses are taught concurrently, followed by a two week examination period. The course is taught in the third semester.

Most students of the database course unit have taken a course on "Introduction to CS" which briefly introduces relational systems, tables and SQL. Most students have also taken the course "Web Programming" where students learn to access relational data from a high-level programming language. Some of the students of the course, however, in particular Software Engineering students, have not taken Web Programming, and other engineering students and students taking a Business degree with CS as a minor have taken neither. All topics from the Introduction to CS course are repeated in the Databases course, but are more difficult for students lacking the background. Software Engineering students subsequently take Web Programming, but they do lack some prior knowledge when taking the Databases course that other students have obtained.

The Quality Assurance report proposed adding a new course for engineering students that would cover the topics at a lower level [10]. This course has already been developed and is called "Data Processing", but unfortunately it is given in a 3-week semester, so not all these students can take it;

[1] ECTS is short for the European Credit Transfer and Accumulation System, which is intended as a standard for comparing higher education studies across EU and collaborating European countries. One academic year corresponds to 60 ECTS credits, which in turn should be equivalent to 1500–1800 hours of study. For more details, refer to http://ec.europa.eu/education/ects/ects_en.htm.

in particular the Business-CS students cannot. Nevertheless, the focus of the Databases course unit—and of the revision described here—is on SCS students.

It should be noted that an elective database course has been given every other year within the SCS, called "Performance of Database Systems". The elective course covers the main algorithms of database management systems in detail, including algorithms for transaction management, and for query processing and optimization. Some of the elective topics of the IM knowledge area are covered in that course.

B. Goals of the Course Revision

The goal of the current course revision was two-fold. First, the goal was to address the lack of coverage of the ACM Curricula, as described in the introduction. A second goal was to retain the strong emphasis on practical aspects of software development that are needed to prepare students for industry.

As mentioned in the introduction, the missing learning outcomes generally referred to "social" topics, relating to databases in social context, and more recent developments. The old course unit, on the other hand, focused very much on technical issues relating to software development. The goal was to increase coverage of the ACM core topics and learning outcomes, while retaining the strong technical focus.

Through various discussions with industry, from large software development companies to small start-up companies and at various levels of formality, it has become clear that the local industry considers the most important aspects of a database course to be proficiency with SQL query development (including views, stored procedures and triggers), effective database design, and familiarity with database administration tasks. The last topic is rarely covered in traditional first databases courses, as it is an extensive topic and the details of database administration are very dependent on the system in question. Some institutions, such as Auckland University of Technology, offer advanced undergraduate courses that include extensive coverage of database administration topics, and indeed many such topics are covered in the course on "Performance of Database Systems" at Reykjavik University. It was nevertheless decided to include some insight into this topic in this first undergraduate course.

C. The Revision Process

The revision process consisted of four steps, which are described in more detail in the following chapters:

1) Choose Information Management knowledge units from the guidelines in the ACM Curricula (Chapter IV).
2) Develop manageable learning outcomes from the numerous topics and learning outcomes of the chosen knowledge units (Chapter V).
3) Review the learning outcomes to ensure adequate coverage of the ACM Curricula (Chapter VI).
4) Devise a complete teaching plan for the course to ensure conformance with the developed learning outcomes (Chapter VII).

III. Review of the ACM Curricula

In this section, we briefly review the structure and requirements of the ACM Curricula, as well as the Information Management (IM) knowledge area. Knowledgeable readers can safely skip this section.

A. Core and Electives

The ACM Curricula defines topics and corresponding learning outcomes, which should be covered in undergraduate programs. The topics are divided into core topics and elective topics; the core topics are further divided into Tier 1 and Tier 2 topics; the same division is applied to the learning outcomes. The requirements for computer science programs are then described as follows [8, p. 27]:

- A curriculum should include all topics in the Tier 1 core and ensure that all students cover this material.
- A curriculum should include all or almost all topics in the Tier 2 core and ensure that all students encounter the vast majority of this material.
- A curriculum should include significant elective material: Covering only "core" topics is insufficient for a complete curriculum.

The requirement for Tier 2 topics is subsequently elaborated as follows [8, p. 30]: "A computer-science curriculum should aim to cover 90-100% of the Core Tier 2 topics, with 80% considered a minimum." While coverage requirements are not explicitly defined for learning outcomes, we assume the same requirements for learning outcomes.

For each learning outcome, the level of mastery is also given. Three levels are defined [8, p. 34] (similar levels have been adopted in the Icelandic educational system):

Familiarity: The student understands what a concept is or what it means. This level of mastery concerns a basic awareness of a concept as opposed to expecting real facility with its application. It provides an answer to the question "What do you know about this?"

Usage: The student is able to use or apply a concept in a concrete way. Using a concept may include, for example, appropriately using a specific concept in a program, using a particular proof technique, or performing a particular analysis. It provides an answer to the question "What do you know how to do?"

Assessment: The student is able to consider a concept from multiple viewpoints and/or justify the selection of a particular approach to solve a problem. This level of mastery implies more than using a concept; it involves the ability to select an appropriate approach from understood alternatives. It provides an answer to the question "Why would you do that?"

B. Information Management Knowledge Area

The IM knowledge area is composed of twelve knowledge units [8, p. 112]. Figure 1 shows these areas, as well as the number of topics and learning outcomes in each of the three categories: Tier 1, Tier 2 and electives. The most important

Knowledge Unit	Topics / Learning Outcomes		
	Tier 1	Tier 2	Elective
IM/Information Management Concepts	4 / 6	4 / 7	
IM/Database Systems		6 / 8	1 / 1
IM/Data Modeling		6 / 8	
IM/Indexing			6 / 6
IM/Relational Databases			11 / 13
IM/Query Languages			11 / 6
IM/Transaction Processing			4 / 7
IM/Distributed Databases			8 / 5
IM/Physical Database Design			8 / 9
IM/Data Mining			7 / 8
IM/Information Storage And Retrieval			21 / 5
IM/MultiMedia Systems			7 / 6
Total	4 / 6	16 / 23	84 / 66

Fig. 1. Overview of the Information Management (IM) knowledge area.

observation is that the 20 core topics and 29 core learning outcomes are all from the first three knowledge units, which must therefore be covered extensively. On the other hand, most of the 84 elective topics and 66 elective learning outcomes are found in the remaining nine knowledge units, which can thus be covered selectively in the course unit.

IV. Choice of Knowledge Units

Based on the output of the quality assurance process [10], two main goals were set for the choice of knowledge units. The first goal was to cover the core topics and learning outcomes from the ACM Curricula at (or very near) 100%. The second goal was to select a good set of knowledge units to support the emphasis on software development, and to cover as many topics and learning outcomes of the chosen knowledge units as possible. Finally, the coverage of the chosen knowledge units must of course fit within the 6 ECTS scope of the course unit.

The IM knowledge area is composed of twelve knowledge units [8, p. 112]. Three of these units contain all the mandatory topics and learning outcomes, and must therefore be covered fully in this course, namely: Information Management Concepts; Database Systems; and Data Modeling. The remaining nine units contain only optional topics.

The decision of which of these units to cover in this course was based on their perceived relevance to industry, focusing on topics deemed most likely to make students better software engineers. The following five units were chosen: Indexing, Relational Databases, Query Languages, Transaction Processing, and Physical Database Design. In total, the chosen knowledge units contain 57 topics (4 Tier 1; 16 Tier 2; and 37 elective) and 67 learning outcomes (6 Tier 1; 23 Tier 2; and 38 elective). In Chapter V, we evaluate the coverage of these topics and learning outcomes in the final course design.

The remaining units, while interesting in their own right, were largely omitted from the course, although a few topics and learning outcomes happened to be covered. These units were: Distributed Databases, Data Mining, Information Storage and Retrieval, and Multimedia Systems. Some of the topics of Indexing and Transaction Processing are covered in detail in the Performance of Database Systems elective course and most of the Data Mining unit is covered in an elective course on Introduction to Machine Learning. The remaining topics are not covered in any course within the CS program, but since they are elective topics, this is acceptable.

V. Reduction to Learning Outcomes

The main focus was placed on writing a good set of learning outcomes, as they best describe what students should be capable of doing following the course. Designing a course with 67 learning outcomes, however, is not feasible as neither students nor teachers will be able to determine what the key aspects of the course are. In this section, we therefore describe the process used to produce the final course description and learning outcomes, and compare the result to the previous learning outcomes.

A. Reduction Process

The course unit teacher read through all the 57 topics and 67 learning outcomes from the chosen knowledge units in the ACM Curricula, noting topics and goals of the learning outcomes, and aggregated the course goals into a smaller set of learning outcomes. After coming up with a set of 15 preliminary course learning outcomes, he then went through the entire list of learning outcomes and topics from the ACM Curricula and marked topics and learning outcomes that he believed were covered by the 15 identified learning outcomes to check the coverage of the learning outcomes suggested in the ACM Curricula.

After this process, a small set of topics and learning outcomes were identified as missing from the chosen knowledge units. In addition, as mentioned above, discussions with industry representatives had indicated that students were not sufficiently aware of the requirements for database administration, so a learning outcome was added to that effect. After this revision, 18 learning outcomes had been identified; after yet one more round of refinement, the final set of learning outcomes were ready. The learning outcomes were then further aggregated into an overall description of the course. The complete course description is shown in Figure 2, where learning outcomes are numbered for ease of reference.

The learning outcomes were divided up to three levels: knowledge, skills and competences, in accordance with the national guidelines [13]; these levels correspond roughly with the familiarity, usage and assessment levels used in the ACM Curricula [8]. This resulted in a clear and concise grouping:

- Learning outcomes at the knowledge level mostly relate to the context of databases on the one hand and theory of database management on the other.
- Learning outcomes at the usage level relate to database queries and programming, mostly in SQL, but also in the theoretical languages.
- Learning outcomes at the assessment level related to database design, both logical and physical.

Not all learning outcomes are created equal, however; the knowledge level learning outcomes are judged to correspond to about 10% of the learning effort, the usage level learning

Description

The course is a hands-on introduction to information management in general and relational database management in particular, covering the following topics: the role and function of database management systems; the relational database model, including relational concepts and relational query languages; data modeling using the ER model and its conversion into a relational database schema; all major aspects of the SQL language, covered in detail, including DDL, DML, complex queries, views, procedures, triggers and transactions; transaction and administration concepts; and, finally, a brief discussion of alternative data models and approaches, such as unstructured databases, information retrieval and "big data".

Learning Outcomes

At the end of the course, the student should be able to:

Knowledge

[1] Discuss structured and unstructured databases in social and organizational context.
[2] Describe concepts and measures related to reliability, scalability, efficiency and effectiveness.
[3] Describe major components and functions of database management systems.
[4] Describe and compare common data models.
[5] Describe fundamental principles of the relational model.
[6] Describe fundamental transaction concepts.
[7] Describe basic database administration functions.
[8] Discuss concepts and techniques for unstructured data and information retrieval.
[9] Discuss major approaches to storing and processing large volumes of data.

Skills

[10] Write SQL commands to create a complete relational database.
[11] Write SQL commands to insert, delete, and modify data.
[12] Write simple and complex SQL queries to retrieve data, including joins, aggregates, sub-queries, nested sub-queries, and division.
[13] Write simple database views, stored procedures, triggers and transactions.
[14] Write queries in relational algebra and tuple relational calculus.

Competences

[15] Model data requirements and constraints using the ER-model
[16] Convert an ER-model into a corresponding relational schema.
[17] Normalize a relational schema.
[18] Select and create the appropriate indices for simple database queries and constraints.

Fig. 2. Course description and learning outcomes.

outcomes are considered about 60% as they include all the database coding outcomes, and the evaluation and design level learning outcomes represent about 30% of the final grade.

B. Comparison with Previous Learning Outcomes

In the pre-existing course description, there were six learning outcomes, which were not classified into coverage levels. These were the following:

- Design and build a simple interactive web-based application and demonstrate how to implement a database-driven web site.
- Cite the basic goals, functions, models, components, applications, and social impact of database systems.
- Demonstrate use of the relational algebra operations from mathematical set theory.
- Create a relational database schema in SQL that incorporates key, entity integrity, and referential integrity constraints.
- Demonstrate data definition in SQL and retrieving information from a database using the SQL language.
- Cite the main normal forms defined in relational database theory and how they affect database design.

As discussed in Section II, the first learning outcome is currently covered in the course "Web Programming" and thus no longer belongs in the Databases course. The remaining learning outcomes have all been expanded and strengthened in the new course description. In particular, normalization has moved from the Familiarity level (indicated by the verb "cite") to the Usage level. Furthermore, the single learning outcome relating to data modification and retrieval in SQL has been expanded to three more detailed learning outcomes. Finally, some topics have been added, such as database administration and approaches to storing and processing large volumes of data. Overall, it is our opinion that the new course description is much more comprehensive and up to date, and gives much more detailed guidance to future course unit teachers.

VI. ACM Conformance Evaluation

To evaluate the conformance to ACM learning outcomes, we considered all the ACM learning outcomes, both from the eight chosen IM knowledge units to be integrated in the course and from the remaining IM knowledge units, and estimated the level at which they would be covered while satisfying the 18 course learning outcomes. Following this evaluation, we could then aggregate the outcomes and obtain coverage ratios.

A. Core Topics and Learning Outcomes

The three main knowledge units had 4 Tier 1 topics, 16 Tier 2 topics and 1 elective topic. All these 21 topics were covered. There were 6 Tier 1 learning outcomes, 23 Tier 2 learning outcomes and 1 elective learning outcome. These were covered with a few exceptions: one was covered in a different course; one was omitted altogether; two were covered at a lower level; and one was covered at a higher level. Figure 3 gives details for the core learning outcomes where the coverage differs from the requirements of the ACM Curricula, while the left hand side of Figure 4 gives a detailed overview of the coverage of core topics and learning outcomes.

To transfer the coverage of learning outcomes into a single numerical value, we have three choices for the two partially

Learning Outcome	Describe the main concepts of the OO model such as object identity, type constructors, encapsulation, inheritance, polymorphism, and versioning [8, p. 114].
Coverage	[Familiarity] → Omitted.
Motivation	Fully covered in the Web Programming course.
Learning Outcome	Give a semi-structured equivalent (e.g., in DTD or XML Schema) for a given relational schema [8, p. 115].
Coverage	[Usage] → Omitted.
Motivation	Too time-consuming.
Note	While this learning outcome is not covered specifically in any course, we believe CS graduates could at least do this to the level of familiarity.
Learning Outcome	Describe the differences between relational and semi-structured data models [8, p. 115].
Coverage	[Assessment] → [Familiarity]
Motivation	More detailed coverage would be too time-consuming.
Note	The verb "describe" in a learning outcome at the level of assessment is surprising, as this verb is typically associated with the familiarity level.
Learning Outcome	Identify vulnerabilities and failure scenarios in common forms of information systems [8, p. 113].
Coverage	[Usage] → [Knowledge]
Motivation	The course unit teacher considered it important that students were well aware of (types of) failures, but that more detailed coverage could be achieved in a security course.
Note	We believe that it would be useful to integrate this learning outcome to the usage level in the future.
Learning Outcome	Describe concepts in modeling notation (e.g., Entity-Relation Diagrams or UML) and how they would be used [8, p. 114].
Coverage	[Familiarity] → [Usage]
Motivation	Modeling is key to software development, so students must be adept at modeling; hence the more detailed coverage.

Fig. 3. Changes in core learning outcome coverage in the Databases course unit.

	Tier 1	Tier 2					Elective				
Knowledge Unit	Covered	Covered	Other Courses	Lower Level	Not Covered	Total	Covered	Other Courses	Lower Level	Not Covered	Total
IM/Information Management Concepts	4 / 6	4 / 6		/ 1		4 / 7					
IM/Database Systems		6 / 8				6 / 8	1 / 1				1 / 1
IM/Data Modeling		6 / 5	/ 1	/ 1	/ 1	6 / 8					
IM/Indexing							6 / 6				6 / 6
IM/Relational Databases							10 / 12			1 / 1	11 / 13
IM/Query Languages							9 / 3	1 / 1		1 / 2	11 / 6
IM/Transaction Processing							4 / 3		/ 2	/ 2	4 / 7
IM/Distributed Databases										8 / 5	8 / 5
IM/Physical Database Design							6 / 2			2 / 7	8 / 9
IM/Data Mining										7 / 8	7 / 8
IM/Information Storage And Retrieval							/ 3			21 / 2	21 / 5
IM/MultiMedia Systems										7 / 6	7 / 6
Total	4 / 6	16 / 19	/ 1	/ 2	/ 1	16 / 23	36 / 30	1 / 1	/ 2	47 / 33	84 / 66
Percentage	100% / 100%	100% / 83%	/ 4%	/ 9%	/ 4%	100% / 100%	43% / 45%	1% / 2%	/ 3%	56% / 50%	100% / 100%

Fig. 4. Overview of the coverage of core and elective topics / learning outcomes. Shaded knowledge units are those not chosen for the course.

covered learning outcomes: count them as covered; count them as not covered; and count them as "half" covered. The coverage is 96% with the first choice and 87% with the second choice. Using the third and perhaps most logical choice the coverage is 93%; well within the requirements of the ACM Curricula.

B. Elective Topics and Learning Outcomes

The 8 selected core and elective knowledge units contained 41 topics. Of these, a total of 37 topics were covered in the course, or 90%. Counting all elective topics from the entire IM knowledge area, however, the coverage was 44%. The right hand side of Figure 4 gives a detailed overview of the coverage of elective topics and learning outcomes.

Considering first the eight chosen knowledge units, 28 learning outcomes are fully covered, 2 partially and 8 are not covered, for a coverage of 76%. Considering the entire IM knowledge area, 31 learning outcomes were fully covered, 2 partially and 33 not covered, for a coverage of about 48% of all elective learning outcomes. We believe that this satisfies the requirements of the ACM Curricula that the "curriculum should include significant elective material" [8, p. 27].

C. Summary of Conformance

Figure 5 summarizes the conformance to the ACM standard. More detailed analysis of the coverage of topics and learning outcomes can be found in [11], including the motivations for exclusion of particular elective topics and learning outcomes.

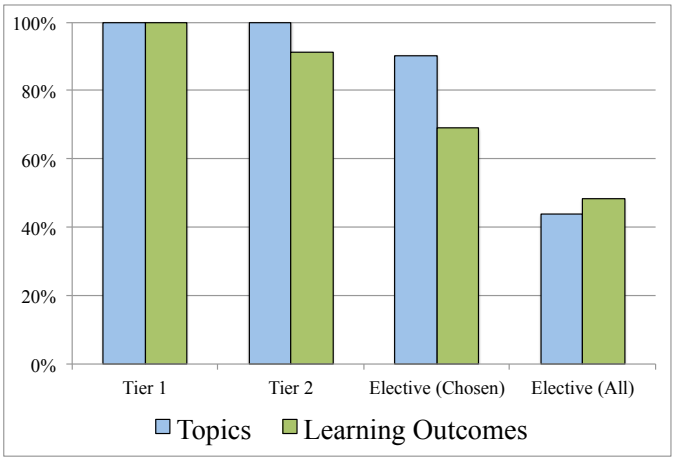

Fig. 5. Summary of the conformance of the Databases course to the ACM Curricula. Coverage of electives is given both with respect to the eight "chosen" knowledge units and "all" twelve knowledge units. The figure shows that Tier 1 topics are fully covered, Tier 2 topics are fully covered but Tier 2 learning outcomes covered to 90%, and about half of the elective topics and learning outcomes from the IM knowledge area are covered, thus satisfying the requirements of the ACM Curricula.

VII. Course Execution

The final step of the course design was to ensure that the teaching plan fulfilled the learning outcomes. In terms of student effort and course grading, about 10% of the effort was devoted to the theory and context topics, 60% to the query language and programming topics, and 30% to the design topics. These numbers were given to students at the start of the course and reflected in projects and final examination.

A. Teaching-to-Learning-Outcome Matrix

After creating the teaching plan, the course unit teacher created a learning outcome support matrix, shown in Figure 6, with the 12 weeks on one axis and the 18 learning outcomes on the other. The course unit teacher then filled in an "X" where the week's teaching would support the learning outcome. This way the teacher could ensure that each learning outcome was covered somewhere in the course.

As Figure 6 shows, some weeks contribute to many learning outcomes, while others contribute only to a single learning outcome. For example, week 4 on "Advanced SQL" contributes only to learning outcome 12 (Write simple and complex SQL queries to retrieve data, including joins, aggregates, sub-queries, nested sub-queries, and division) while week 11, which covers "Non-relational data (unstructured, IR, streams, big data)" contributes to five different learning outcomes. In the former case, the single learning outcome is at the core of the technical content of the course, while in the latter case, the multiple learning outcomes relate to the social context and are all at the knowledge level.

Note that the learning outcome "Describe basic database administration functions"—which was added following a request from industry—was satisfied by having an experienced database administrator from industry give a guest lecture; this was very well received by students.

B. Course Unit Implementation

During the course, the course unit teacher emphasized creating a learning environment rather than a teaching environment. This was difficult given that there were 250 students registered, including several distance-learning students. Furthermore, since lectures were recorded and many students preferred recordings to attending lectures, the attendance ratio was low.

During lectures the course teacher used many examples, and often took the time to have students work out queries or solutions to problems within the lecture (in the recording the teacher encouraged listeners to do the same). The course teacher explicitly forbade using computers in the classroom, in order to remove one source of distraction; this was generally met with a positive reaction once students got over the shock, but some students were quite dissatisfied.

Weekly assignments were given, that should be completed within the 90-minute practical session, as well as five larger projects. Overall the emphasis was to have students learn with their "brains and fingers"—to both have a solid understanding of the theory and a good handle on its usage.

C. Student Evaluation of Teaching

Students evaluated the course twice, first in weeks 4-5 and then at the end of the course, but before the exams. The overall course grade, on a scale of 1 to 5, was around 3.5. This is close to average for a core course of this size.

Specific comments related mostly to specific aspects of the performance of the course, such as project workload and lectures. Among these comments, some will be quite useful for the upcoming versions of the course.

Only a few comments were made on the contents of the course. A few students complained about the amount of material covered and contended that it did not fit within 6 ECTS. Finally, some complained about the coverage of relational algebra and calculus, as those topics would be of limited use in industry. A couple of students mentioned that they liked the guest lecture from the database administrator. No comments were made, however, that show any reason to change the course description or learning outcomes.

D. Broader Considerations

Viewing the course quality from the "education as production" perspective [12], it is clear that we have satisfied the goals that we set ourselves, in ensuring coverage of appropriate material, informed by the ACM Curricula guidelines. This paper presents a detailed description of the process and the mechanics involved in the course redesign. We may also claim that we have addressed another of Pears' notions of educational quality through the "quality as service" dimension [12]. In this course redesign, a driving goal was the wish to produce graduates suitably skilled to meet industry needs in the local setting. Thus a primary goal of the course design was also to provide "service" to a key set of stakeholders, namely the employers of the resulting graduates. While previous industry input was used when designing the course, an evaluation

Week	Lecture Topics	Learning Outcomes																	
		[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]
1	Intro, relational model basics, SQL DDL	X		X		X									X				
2	Basic SQL queries (joins, aggregates)										X	X	X						
3	Relational algebra, calculus and SQL					X							X		X				
4	Advanced SQL												X						
5	Semantic database design (ER modeling, schema conversion)				X											X	X		
6	Conceptual database design (normalization)					X					X							X	
7	Files, indices, introduction to physical database design		X	X															X
8	Coding with SQL (views, procedures, triggers, transactions)													X					
9	Transaction concepts		X	X			X												
10	Introduction to security and administration		X	X				X											
11	Non-relational data (unstructured, IR, streams, big data)	X	X		X				X	X									
12	Database history and summary																		

Fig. 6. Evaluation of teaching coverage of course unit learning outcomes; the column headings refer to learning outcomes from Figure 2.

of industry satisfaction with the course and the graduate capabilities remains for later work.

We also acknowledge that there are many perspectives on curriculum paradigms and educational quality that could have been adopted in this evaluation (e.g., educational quality as "transformation of the student" [14]; functional, transactional and critical curriculum paradigms [15]. This review could be considered as situated within a "functional curriculum paradigm", where the curriculum:

> "Is set in the present. Fits what the industry or society needs now for that person to take up that job. Reproductive. Technical. Task and skills-based for a specific occupation. Content of subject area is very important. Has objectives that are often set by an external body or an industry group with some input from teachers. Sometimes referred to as practical. Methodology often involves set lectures and teacher-directed demonstrations, workshops or laboratories" [15].

But the goal here was a very pragmatic one for the course lecturer of producing a high quality revision of a database course informed by local needs and sound and current international guidance.

VIII. Conclusions

The major focus of quality assurance work is often on the evaluation process. In this paper, however, we have presented an example of the re-design of a particular course unit using the ACM Curricula, in order to address concerns raised by a quality assurance evaluation made using the same standard. The focus has thus been on the influence from the quality assurance process at the education (degree) program level down to a specific course instance. This re-design and subsequent evaluation will be used as input to future evaluations of the education program.

We have first presented the process from synthesizing the ACM topics and learning outcomes into a course description and associated learning outcomes. We have then presented an evaluation of the ACM conformance, which shows that the objectives of the course re-design were met. Finally, we have described the execution of the course, and in particular how the course unit teacher ensured coverage of the learning outcomes.

We believe that the experience of using the ACM Curricula for the course re-design was very positive. An informal comparison with courses at other universities revealed that the course is very similar in content to those courses; it is essentially a traditional first course on databases. But using the ACM Curricula gives confidence that the topics and the learning outcomes of the course are appropriate and sufficient. Essentially, it allowed the course designer to "rest on the shoulders of giants."

Using the learning outcomes of the ACM Curricula directly, however, would result in an excessive number of learning outcomes, which in turn would be confusing for both the students and the teacher. We have therefore described the process of aggregating the ACM learning outcomes to the course learning outcomes, as well as presented the final outcome. We have evaluated the coverage of topics and learning outcomes, and found both to be well within the requirements of the ACM Curricula. We have described in some detail the choices that deviated from the ACM Curricula, and the motivation in each case. We also described some topics that could be added or covered in more depth, without changing the overall description of the course. We can conclude, however, that the course design has met its goals of increasing ACM coverage, while retaining a strong focus on software development.

The final contribution is that of using the week-to-learning-outcome matrix, to ensure that the course execution satisfies the course description. Needless to say, this is a very subjective evaluation, but it has the advantage that it could be relatively easily checked by an outside evaluator. Overall, based on the student evaluation results and the analysis of the course unit teacher, we believe that the course design and execution were successful, resulting in a course that satisfied ACM requirements as intended, as well as the university's goals of preparing students both for industry jobs and further study.

References

[1] L. Harvey and J. Williams, "Fifteen years of quality in higher education," *Quality in Higher Education*, vol. 16, no. 1, pp. 3–36, 2010.

[2] G. Neave, "Quality enhancement: A new step in a risky business? a few adumbrations on its prospect for higher education in europe," in *Quality assurance in higher education: contemporary debates*. Palgrave MacMillan, 2014, pp. 32–50.

[3] E. Hazelkorn, *Rankings and the reshaping of higher education: The battle for world-class excellence*. Palgrave Macmillan, 2015.

[4] B. Salter and T. Tapper, *The State and Higher Education*. Routledge, 2013.

[5] S. Guri-Rosenblit, H. Šebková, and U. Teichler, "Massification and diversity of higher education systems: Interplay of complex dimensions," *Higher Education Policy*, vol. 20, no. 4, pp. 373–389, 2007.

[6] J. Browne, *Securing a sustainable future for higher education: An independent review of higher education funding and student finance*. UK Department for Business, Innovation & Skills, 2010.

[7] R. McDermott, M. Daniels, and M. K. Lárusdóttir, "Subject-level quality assurance in computing: Experiences from three national perspectives," in *Frontiers in Education Conference (FIE)*, Oct. 2014, pp. 1–8.

[8] The Joint Task Force on Computing Curricula, *Computer Science Curricula 2013*. ACM/IEEE, 2013.

[9] M. K. Lárusdóttir, M. Daniels, and R. McDermott, "Quality assurance using international curricula and employer feedback," *Australian Computer Science Communications*, vol. 160, pp. 19–27, 2015.

[10] A. Ingólfsdóttir, M. K. Lárusdóttir, and H. A. Úlfarsson, "Comparing the computer science curricula at reykjavik university to the suggestions given in the ACM/IEEE standard," Reykjavik University, Tech. Rep. RUTR-CS14001, 2014.

[11] B. Jónsson, M. K. Lárusdóttir, M. Daniels, T. Clear, A. Young, and R. McDermott, "Re-design of a database course unit using the ACM Computer Science Curricula 2013," Reykjavik University, Tech. Rep. RUTR-CS16003, 2016.

[12] A. Pears, "Does quality assurance enhance the quality of computing education?" in *Proceedings of the Australasian Conference on Computing Education*, T. Clear and J. Hamer, Eds., vol. 103, 2010, pp. 9–14.

[13] "National qualification framework for higher education," Icelandic Ministry of Education, Science and Culture, 2011, https://www.stjornartidindi.is/Advert.aspx?ID=7fa0729e-dacc-47e3-b626-96efb036ef68, accessed April 15, 2015.

[14] M. Corder, M. Horsburgh, and M. Melrose, "Quality monitoring, innovation and transformative learning," *Journal of Further and Higher Education*, vol. 23, no. 1, pp. 101–108, 1999.

[15] T. Clear and A. Young, "Met a researcher? Research paradigms among those new to research," in *Proc. 14th Annual Conference of the NACCQ*, 2001, pp. 23–31.