Can More Integrative Activity Improve the Learning Experience?

Alastair Monger, David Cox, Sheila Baron, Thomas Adam
Computing Subject Group, Technology Faculty
Southampton Solent University
al.monger@solent.ac.uk

Abstract

Both the literature, and an analysis of a typical learner experience on a unitised course, suggest that reinforcing integrative activity may improve the learning experience, and consequently retention.

There are many integrative methods and approaches, including the use of information and communication technology (ICT), which might be used to raise the level of integrative activity.

Two specific integrative activities undertaken primarily on our Computing courses (as part of a wider experiment of activities) are described and evaluated. The first activity exploits the use of a joint case study in the context of human computer interface design. The second activity shows support across units in the context of databases.

The positive outcome of this experiment leads to a recommendation to document and extend integrative activity on our courses.

1 Introduction

This paper focuses on an experiment to explore whether a more integrative approach can improve the learning experience. It is one of a number of initiatives to try to improve retention and recruitment on the Computing group's courses.

The main undergraduate computing and business information technology (BIT) programme of courses in the Computing group were first validated and delivered in a unitised structure around 1991/1992. (Note that no distinction is drawn between modularisation and unitisation in this paper. However, for consistency the term unit is used throughout). Apart from a timetabled workshop (with hours contributed by units) and some associated integrated assessments, the level of integration was not significantly greater than now. However, in the early years retention and the learning experience had not been identified as major issues.

Numbers increased rapidly through the mid-nineties stretching staff resources to the limit, with perhaps a consequent negative impact on course cohesion and integrative activity. The workshop was formally removed at revalidation in 1996, and integrated assessment activity greatly reduced. Retention issues, particularly in respect of software development, began to emerge.

Revision of the academic year structure, together with a strategic institutional objective for increased integrative activity on courses, resulted in more "long-thin" (rather than "short-fat") units in the 2001 revalidation. This offered the potential for more integrative activity.

Integration appears to be a fertile area of research in higher education (HE). According to Helen Baron [1] in a substantive study on students' perceptions of modular programmes - *"Much research on learning in HE focused on what happened within modules and units, but little existed on what happened between them"*. Furthermore, she found in her study that *"Factors which appeared to inhibit synergy development included disconnected subject areas…"*.

Computing is a subject discipline in particular which requires skills and knowledge in many subject areas such as software design, human computer interaction (HCI) and databases. However, it is essential that the learner can connect and synthesise these constituent parts. Even though the course may be well designed and coherent on paper, do we do enough to ensure that learners can make these connections on our unitised courses?

There may well be other reasons why retention has became an issue, and we have taken action in recent years to tackle it – eg doubling support for software development at level 1. Nevertheless, this background and the literature suggest that achieving a more integrated delivery might be a productive line of approach.

The experiment focuses mainly on level 2 of the Computing Programme which was considered fertile for integrative activity, particularly because of several “long-thin” application development units (in object-oriented development, databases and the internet subject areas) and an overall workload issue raised by some students.

It should be emphasised that the focus of this study is on achieving greater integration on unitised courses, not on the merits or otherwise of unitisation. We are assuming unitisation is here for the foreseeable future!

Initially, section 2 presents and analyses a student perspective. Section 3 discusses possible approaches for promoting integrative activity. Section 4 outlines the experiment. Sections 5 and 6 present in detail two specific evaluated activities from this experiment. Conclusions are drawn, and recommendations made, in the last section.

2 A Typical Learner Experience in Unitised HE? – Is There a Problem?

A primary source for this study is the following perspective, in italics, presented by the level 2 Computing Programme student representative (and paper co-author).

We rush across campus from a software engineering unit lecture to an internet development unit classroom session. Both units are perceived as well delivered, and generally this is the case across the course. There is a tendency for some students to forget that although units are taught and appear as separate entities, they are still related to one another. Overall, students' perception as to how units are delivered are well anticipated by members of staff. Of course there are discrepancies between units and varying differences of opinion from students, but most of the subjectivity is bias about how one lecturer delivers a unit, to another. These areas are not quantifiable in terms of cohesion amongst units, of course, and a lot of it is just one or two students’ attitudes towards a lecturer.

The Software Modelling Specification (SMS) unit is based on the requirements analysis, software design etc of a sales/manufacture application scenario, and is ongoing throughout the unit. The Yourdon model and C are covered. This unit is different from, say, Object-Oriented Application Development (OOAD). Indeed, this specific unit is often thought of as a "creeping" one, that is, each week requires the student to do some work to supplement understanding from the lecture. Many like this style of teaching, as it allows one to apply techniques learnt to an already known scenario. So in that way, one already has an overall view of what needs to be done, beforehand. The deliverance of this unit is unique in that it contradicts the *more formal style of teaching where there is a set lecture and associated seminars to digest the formal information.*

Overall, people liked Yourdon as a methodology even if they didn't fully understand why they were doing the diagrams, or what they represented. Although a contradiction in terms of learning outcomes versus applied theory, the distinct impression was that many of the students were drawing these diagrams because they had to, without the required underpinning for them to understand why. Of course, this comes down to the way the unit was taught -- a unique approach, and not at all like a traditional lecture of: theory -> application -> result. Since this style was new to people they perceived the unit as an individual one and couldn't see how it related to others, if only because of the way it was taught.

It was also surprising that people were unable to see the link between OOAD and SMS. Whilst the two units are near opposites of each other; what they both have in common is that they're two different methodologies. This was absolutely key to our learning to realise this -- the fact that there is more than one methodology and process one can go through to build software.

The object-oriented development unit assessment is based on a full-life cycle dating agency application development. The ICONIX methodology, UML and Java are used. This is a focal unit. The lecturer's teaching style is excellent. Everything is contextualised by examples in lectures, and more importantly each week with tasks to do which are pre-set by the lecturer. It's a big unit, and overall, the lecturer has managed to section it into deliverable chunks so as not to overload our small minds with complex points. Throughout the unit to date, there has been repeated references to other lifecycles and methodologies – in particular cross-referencing with the SMS unit.

The internet development unit covers essentially client-side web development (Java Script) and associated issues in the first part of the unit, followed by server-side (including the MySQL database and PHP) web development in the second part. The assessment reflects this split with mainly web design and implementation activities based on a flight booking application.

This is probably the most enjoyable unit, if only because it is so very different from all the others. There's no real "thinking" involved as in the other units, and it is more a light-relief. It is well structured, and everything is taught from first principles. The split between client and server-side is really good as it allows for focusing on individual aspects without getting confused with the server-side facets.

A database unit, based on using Oracle, covers essentially database design, SQL, reports/forms and multi-user issues such as database access control. The two assessments are based on a project management control application, and is on-

going throughout the unit. The first assessment is development oriented, the second assessment is more research-oriented in respect of multi-user issues.

This unit cannot be praised enough. It has been an excellent eye-opener both in terms of content, and the way in which the online resources are structured and planned. Many have commented how useful they have been. As with SMS in this sense, the idea of weekly (or in one or two cases fortnightly) tasks to submit are good, and there is plenty of opportunity to put into practise the theory learnt via the taught SQL. The familiarity of SQL from previous subjects also helps here, and could certainly be exploited further to enhance integration.

Although we generally appreciate the richness of contrasting application scenarios in the various units, this unnecessarily increases our workload. The time taken to understand these scenarios, and varying tutor expectations of what is required, often reduces the time to focus on the key subject specific learning outcomes. This apparent lack of cohesion also applies with regard to the plethora of methodologies, languages and toolsets we are required to use.

It isn't always the case that it reduces any time to focus on one aspect, what is annoying is that a lecturer may well ask whether we have done "XYZ" in one unit, and decide that they are "going to teach us properly" regardless. This is most frustrating, and does indeed appear to demonstrate a lack of cohesion between units. If this integration concept is to succeed, then lecturers are going to have to be much more aware of what is happening as a whole, rather than looking at an isolated case (such as their own unit).

Remember that the "plethora of methodologies" is there for a reason! Without them there would be no way for us to contrast other methods of working. There is already a danger of people assuming that the buck stops with Java -- an attitude reflected at the abhorrence for C in SMS. This cannot be allowed to be a dominant opinion amongst the students, and the outlet that SMS has, at least makes people aware of it, whether they agree with it or not.

Most students, particularly the high-fliers, have no difficulty in identifying and applying the links across the units. Weaker students have more difficulty, and are surprised when the database tutor, for example, is critical of a lack of evidence of testing. Students who don't seem to put much work into the course often seem able to exploit a perceived lack of cohesion and tutor communication within this (unitised) course.

One thing that the database tutor has done well is not to emphasise aspects from previous units that have already been taught. Where it is applicable for the given task to use past knowledge, then the decision to do so has always been left to the student. It is this philosophy of teaching that should help make integration amongst units successful.

So is there a problem?

On the whole this student learner perspective indicates a positive learning experience is being provided by the course team. Generally the units are well received, and the students seem to appreciate the diversity of approach, including the curriculum, and teaching and learning methods.

Nevertheless, analysis of this perspective suggests issues that may well prevail on unitised courses, but that might be resolved with a more integrative approach. These issues include:

(i) A unit focus at the expense of a course/level focus in delivery

In the Computing Group, considerable effort is expended in designing courses to be a coherent whole. Software development, internet development, databases, for example, are key interrelated themes which are developed through the levels of the course.

However, in an increasingly managed context where unit leaders are clearly held to account for the delivery of their unit, it is often more comfortable as a unit leader to be self-contained, without dependence on and reference to, the wider course context and other tutors.

Can more integrative activity during delivery better balance and cohere the unit and course level perspectives? If so, "lecturers are going to have to be much more aware of what is happening as a whole".

(ii) Workload excess and imbalance

In spite of well intentioned and much course design effort, the devil is in the detail of actual unit delivery. It seems, in spite of the best intentions of individual unit leaders, that workload seems to be a perennial student issue (and is often reported in the staff/student forums). Too many application scenarios (usually cast in an assessment context), and a "plethora of methodologies", appear to be contributory factors to this issue.

Can the workload be reduced (including the tutor's!) whilst still achieving the learning outcomes? Might shared case studies, rationalisation of the curriculum content, more integrated assessment, for example, help in this respect?

(iii) Overlapping coverage, and plurality of perspectives, of topics

Incorporating different perspectives and approaches in the learning experience is of course important. However, it is also important to clarify where the fundamental principles of a topic is taught (eg testing in a software engineering unit), and where it might be applied (and safely assumed) as a subsidiary topic in a “client” unit (eg internet development).

During this integration project, the varying approaches by tutors in the course team to topics such as analysis also emerged. Some tutors referred to and utilised use cases, others employed alternative approaches or terminology.

Can more explicit links help? Should a clearer “which unit does what topic” be established and assumed? (and so avoiding the view that “this is most frustrating, and does indeed appear to demonstrate a lack of cohesion between units”. Should consistent terminology be applied across the course?

3 Integration Approaches

Firstly, what do we really mean by integration in the context of this paper? According to the online Cambridge Advanced Learner's Dictionary [2], integrate and related words are:

integrate: "to combine two or more things in order to become more effective"
synergy: "the combined power of a group of things when they are working together which is greater than the total power achieved by each working separately"
cohesive: "united and working together effectively"

It is difficult to argue against the merits of a more integrative approach taking into account these definitions!

It is perhaps helpful to use the model elements in Figure 1 (based on previous work by Monger [3]) to help identify approaches and methods that offer the potential for integrative activity across units.

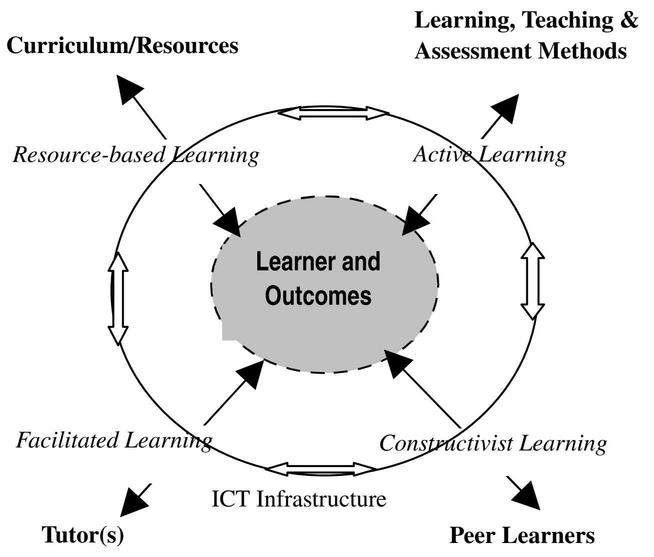


Figure 1 - Key Interrelated Learning and Teaching Elements

(i) Curriculum/Resources

Establish a clearer understanding by students and tutors of where key topics are taught and applied (including horizontally through levels, and vertically across levels). A short/sharp "live" document could, for example, achieve this. A sense of what a "client" unit requires from a "service" unit might also help.

Ensure that unit "schemes of work" (preferably in a consistent format) and learning resources are readily available online. The sharing of learning resources such as notes, presentations, activities (particularly on universally applied topics such as testing) is also desirable.

Establish an agreed application of methodologies, languages and tools throughout the course.

(ii) Teaching, Learning and Assessment (T,L&A) Methods

Shared case studies, use of shared online folders for cross-unit development activities, bi(or multi)-lateral integrative activities (including assessment).

Online discussion forums can also help integrate across units, courses and modes of study.

(iii) Tutor(s) and Peer Learners

Any methods that help integrate tutors and peer learners in a temporal, spatial or virtual manner could help. A shared lecture slot or a "cross-unit" development day in a base room, for example, provide a more integrative feel to a course.

It is also perhaps useful, at least in the context of this paper, to establish the following levels of integration as a basis for current and potential future action on integration:

High	Highly visible integrated learning and teaching experience for tutors and learners in respect of the curriculum, resources and methods. Learner has a course rather than a unit driven perspective.
Medium	Some cross-unit integrative activity. Visible course and level delivery strategies to improve cohesion of the curriculum, resources and methods. More help for learners to see links.
Low	Largely autonomous unit delivery. Limited or no course/level delivery strategy. Learner must identify implied links.

Table 1 - Levels of Integration

4 The Experiment

In the light of the analysis in section 2, the Computing Programme level 2 team designed a programme of integrative activity from September 2005 through to January 2006 (ie period/semester 1 of the current academic year). Essentially, the aim was to raise the level of integrative activity from a low to medium level as indicated in Table 1.

This integrative activity included:

- Shared delivery of topics on the Engineering Software Systems (ESS) and Object-Oriented Application Development (OOAD) units. This included shared interleaving lectures, workshops and group activities.
- Online presence of scheme of work and other information for all units.
- Assessment briefing and sign-off meeting for all units.
- Using a joint case study between Human Computer Interaction Design (HCID) and Information Systems Design (ISD).
- Support and cross-referencing across Developing for the Internet and Application Development & Databases.

Note - The student perspective in section 2 is based on the level 2 delivery in 2004/05. The SMS unit has since been replaced by ESS following revalidation of the Computing Programme in 2004. Actual unit names, however, are not considered important in the context of this study.

All these activities were implemented, although the ESS/OOAD activity was somewhat reduced due to the departure of the OOAD unit leader in December

2005. For this reason, and the sake of brevity, a detailed description and evaluation of the last two activities only are discussed separately in the following sections.

.

5 Integrative Activity 1 - Using a Joint Case Study

HCID is taught in period 1 only and covers interface design, usability engineering/testing and some software development/programming as well as establishing concepts in the area of human cognition such as perception, memory, etc. These important areas are designed to help students in other level two units which cover software and database development as well as general analysis and design.

ISD is taught throughout the year and focuses on current development methods within the context of the analysis and design of a business information system. HCID also examines the concepts indicated above in the context of a small to medium sized business enterprise. A potential unifying area in both of these units is a comprehensive case study which encompasses the areas of practice, functionality and flow of information within a typical business enterprise.

One important aspect of an integrated approach to unit delivery is the promotion of student engagement. Further to this is the idea that if a student engages with one aspect of his/her work it may help with another area of work. Students engage with a study topic for a range of reasons including career prospects and success in a piece of course-work. However, they may also engage because of the content provided in the unit syllabus or the process of curiosity and the value in learning something new [4, 5]. Indeed, Jenkins's [4] study focuses on the desire to learn and its inherent values of curiosity and enlightenment as a motivator on units which involve some aspect of programming. Curiosity and enlightenment are widely regarded as the best kind of engagement. Most level 2 students tend to see the relevance of programming but only if the end result appears to be beneficial and has some end value. They will engage well if the exercise is seen as having some curiosity value which tends to support Jenkins's view. Cox [6] emphasizes the importance of HCID human factor/design exercises with a curiosity value as a way of stimulating student interest in programming and other areas and in a later survey suggests that if a curiosity value is generated, this will provide an incentive to learn in related activities. The survey particularly emphasizes the value of human factors as a way of engaging students with other parts of their course and environment in general [7].

These studies emphasize the value of learning something new and using this newly acquired information to promote stimulation in related areas. With this in mind, the HCID and ISD units used the same case study as a basic way of promoting learning in a separate unit. The case study related to a fictional business enterprise which manufactures a range of health products and holds stocks of raw materials. The scenario was approximately two pages of A4 in length and initially distributed as part of an HCID assessment. It was then slightly modified to reflect key issues in an ISD assignment although the main elements and majority of the scenario remained exactly the same for both assessments. The ISD assessment was handed to students four weeks after they had received the HCID assessment.

A preliminary investigation of students' reaction to the joint case study was conducted in order to evaluate how effective this approach was. This was done four weeks after students had received the ISD assignment so the student body had begun to address many of the points in this assignment and had completed much of the HCID assessment. The investigation was done as part of a larger survey.

Early indications are that students do recognise the value of a joint case study. Over 82% of those who responded, agreed with the following statement: *A similar (or joint) business case study between 2 units (HCID and ISD) helps me with my studies*. This positive feedback on the value of the integrative nature of the case study was further supported by a 74% agreement with this statement: *It helps me because one unit reinforces points I have studied in the first unit.* Taken together, these two statements strengthen the concept of integration. They suggest that a recognisable link between two units helps students in the learning process and, in turn, that such a link may help support the view by Cox [7] and others that engagement via a curiosity value in one area may promote an incentive to learn in a related area.

6 Integrative Activity 2 - Support and Cross-Referencing across Units

Most level 2 students on the BIT, Computing (and Networking) programmes study both Developing for the Internet (DftI) and either Application Development & Databases (ADD), which runs throughout the academic year or Database Technology (DT) which runs in period 1 only.

Although the two database units differ significantly in the depth and breadth of study of theoretical aspects of the subject and issues relating to the implementation of large-scale databases, in period 1 they both focus on practical SQL implementation work using a text-based interface to an Oracle database.

The DftI unit focuses on developing standard-compliant client server web applications using a range of technologies. Database connectivity is an important aspect of web application development and although time prevents extensive exploration of database issues, the assessment for the unit requires students to develop an application that uses SQL embedded in PHP to connect to a simple MySQL database.

The relationship between the units has of course always been taken into account in the design of the content. For example, the SQL work completed in the database units obviates the need for this to be taught in the DftI unit. At the same time, the database units (especially ADD) expect students to draw upon their experiences using contrasting database platforms to inform their understanding of the similarities and differences between the facilities provided by different relational database implementations. However, in the past the relationship has been implicit rather than explicit.

During the 2005-06 academic year a number of measures were taken to increase the integration between the units to promote better understanding of the relationship between these subjects specifically and the overall design of their courses in general.

In the database units:

- The SQL work was restructured and rescheduled to ensure that elements of the language used in DftI were clearly indicated and had been covered before week 8 when MySQL is introduced.
- A new lecture and supporting discussion forum 'thread' were incorporated to discuss the similarities and differences between SQL as used in Oracle and MySQL.

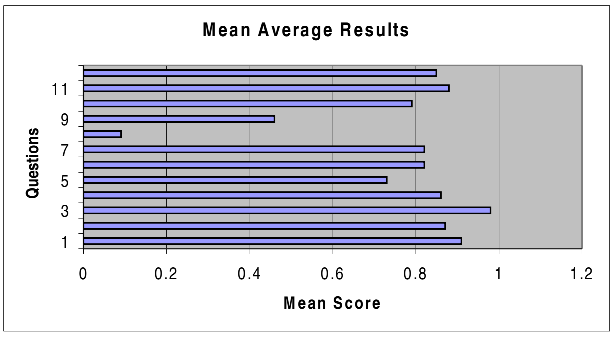
In both subjects:

- Cross-references were frequently made to work done on common issues such as client server architectures and the distinction between data storage and interface design and duplication of coverage was avoided.
- Care was taken to ensure consistent use of terminology across the units.

At the end of period 1 a survey was conducted to establish both the extent to which students were conscious of the role of the SQL work in the database units in supporting the DftI unit and which elements of the content and explicit measures taken this year they considered helpful to their work in that unit. The results of the survey are presented in Figure 2.

2005 Level 2 Integration Project databases and DftI unit questionnaire results

No		Abbreviated question text		Mean	VAR	STDEV
1a		Studying AD&D	37			
1b		Studying DT	26			
1c		Studying DftI	59			
		No studying both DftI and one of the db units	59			
2	1	Able to use SQL from db unit in DftI		0.91	0.33	0.57
3a	2	Data definition useful for DftI		0.87	0.43	0.66
3b	3	Data manipulation useful for DftI		0.98	0.2	0.45
3c	4	Data queries useful for DftI		0.86	0.49	0.7
3d	5	Strengths/limitations of SQL useful for DftI		0.73	0.76	0.87
4a	6	Timing of SQL work helpful		0.82	0.43	0.66
4b	7	Consistent terminology helpful		0.82	0.63	0.79
4c	8	Differences MySQL / Oracle helpful		0.09	1.37	1.17
4d	9	Idenfitication of common issues helpful		0.46	1.07	1.04
5a	10	Lack of duplication of content		0.79	0.8	0.89
5b	11	Lack of repetition in assessed work		0.88	0.59	0.77
5c	12	Tutor awareness of content / schedule of related units		0.85	0.62	0.79



data coded as follows:

strongly agree	= 2
agree	= 1
disagree	= -1
strongly disagree	= -2

Figure 2 - Results of Survey of Perceptions of Cross-Unit SQL Support

The positive mean response (> 0) on all points suggest that most students are aware of the relationship between the subjects and were able to apply what they had learnt in the database units to the work for DftI. They also appreciate most of the actions taken this year to ensure coherence across the units (note particularly question 12 referring to the value of tutor awareness of content and scheduling of work in the other unit).

The relatively low mean response and high variance and standard deviation for the question relating to the value of the lecture on the differences between the two SQL variants is surprising in view of the fact that this is perhaps the most obvious of the measures linking the two study areas. Discussion with students suggests that this is in part due to the timing of the survey – before the majority of students have started to tackle the assessed work for DftI. Up to that point the class exercises for that unit, whose focus is the ability to embed SQL in PHP rather than the use of SQL per se, have deliberately avoided areas of difference. In general those who have started the assessed work are more aware of the value of this information.

The other surprise was in the difference between questions 2, 3 and 4 which enquires about the applicability of different elements of the SQL language to the DftI unit. This suggests that students have found the ability to insert, update and delete data from the database more useful than either creating or querying a database. This may be due again to the nature of the exercises they have completed to date in DftI. Alternatively it may be due to the lack of variations in the syntax of the two SQL implementations for this activity.

From the point of view of the database units this project has been a success from the perspective of improving student awareness of the relationship between the subjects. It has also been helpful in increasing student motivation for database work. As one student put it on being congratulated on his mark for the assessed work "To be honest I find this subject difficult and the need to use it for internet development – which is what I want to do when I leave – was what kept me working on it".

The project has also been valuable from the tutors' perspective in improving their awareness of the revised content of related units following the recent revalidation of the programmes most of which are running the updated units for level 2 for the first time this year.

7 Conclusions and Recommendations

The evidence of this study, particularly the two detailed activities, suggests that more integrative activity can improve the learning experience. Moreover, one might argue that it can also improve the teaching experience!

Most members of the Computing group have shown an interest in the integration project, and some have contributed significantly to it. Others would have contributed given the time, but perhaps that in itself is indicating some justification for the project!

The key recommendation is for course teams to promote more integrative activity on their courses. The objective would be to move from a low to at least a medium level of integrative activity for most courses in the group (cf. Table 1).

One approach might be to produce a short, sharp document that describes (for both tutors and students) the integrative activity planned for the duration of the experiment. This should become an active, well-read ongoing document (requiring little modification from one year to the next) that could fill the gap in a meaningful way between the (often unread) validation documentation and the somewhat insular units themselves. It could cover primarily:curriculum *(what we do),* online resources *(what we do it with)* and T,L&A methods *(how we do it)*, although there is obviously overlap between these interrelated aspects.

It is recognised, however, that the potential for integrative activity is reduced when units are shared between courses/programmes with different pre-requisite/co-requisite structures. There are also potential timetabling issues.

Finally, this study is not about anything new, it is simply about re-emphasing the value of integrative activity in any educational context, unitised or not.

8 References

1 Baron H, Synergy or Segmentation? Students Perceptions of their Learning across Modular Higher Education Programmes, LTSN Generic Centre, Continuing Professional Development Series, 2003.

2 Cambridge Advanced Learner's Dictionary. http://dictionary.cambridge.org/define.asp?key=84507&dict=CALD

3 Monger A, Exploiting Effective Learning and Teaching Methods, Tools and Resources for the Distance or Out-of-Classroom Learning of Databases, LTSN-ICS Teaching, Learning and Assessment of Databases Conference (TLAD), Sunderland 2005.

4 Jenkins T, The Motivation of Students in Programming, proceedings of the 6th ITiCSE Conference (ITiCSE '01), Canterbury, UK, 2001, pp 53-56, New York ACM Press 2001.

5 Jenkins T & Davy J (2002), Diversity and Motivation in Introductory Programming, *Innovations in Teaching And Learning in Information and Computer Sciences. The e-Journal of the Higher Education Academy-ICS (formerly LTSN-ICS)*, 1, 1[on-line] (Jan 2002). URLs: http://www.ics.ltsn.ac.uk/pub/italics/issue1/tjenkins/003.PDF or http://www.ics.ltsn.ac.uk/pub/italics/issue1/tjenkins/tjenkins1.html (visited December 2005)

6 Cox D, A Pragmatic HCI Approach: Engagement by Reinforcing Perception with Functional Design and Programming, proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, Monte de Caparica, Lisbon, Portugal 2005, pp 39-43 , New York, ACM Press, 2005.

7 Cox D, Human Factors in the HCI Learning Process: A Survey, proceedings of the 11th International Conference on Human-Computer Interaction (HCI International 2005), Volume 6, Human Factors Issues in Human-Computer Interaction. Las Vegas, USA, 2005, St. Louis: Mira Digital Publishing, CD-ROM, 2005, ISBN: 0-8058-5807-5