Full Paper submission for the 3rd Annual Conference on the Teaching of Computing. Revised and extended for the CPHC-LDG / CTI workshop, July 1998, Huddersfield.

Title: Managing Coursework:
Wringing the Stone, or Cracking the Nut?
August 1995, Revised and extended June 1998

Author: Dr. John T. Latham
Department of Computer Science,
The Victoria University of Manchester,
Oxford Road, Manchester M13 9PL
England

Email: jtl@cs.man.ac.uk

Phone: +44 161 275 6250

Fax: +44 161 275 6236

Managing Coursework:
Wringing the Stone, or Cracking the Nut?

John T. Latham
Department of Computer Science,
The Victoria University of Manchester,
Oxford Road, Manchester M13 9PL
England
email: jtl@cs.man.ac.uk

August 1st, 1995
Revised and extended June 1998

1 Abstract

We briefly describe ARCADE, a system which has for several years successfully solved the problem of effectively and efficiently *managing* continuous assessment of all coursework undertaken by first and second year undergraduates in one of the UK's largest Computer Science Departments. The system is based upon a simple but effective regime of deadlines and their enforcement. It monitors and extrapolates the behaviour of all students in all of the modules taken, so that it can automatically give feedback (via email) to the students who need it. It collates individual exercise marks and computes a single mark for each student for each module. Thus it does much of the work of an overall laboratory manager, individual module managers and individual personal tutors. The system is suitable for use in any environment where the students have email access, with the coursework being of any nature and done in any medium. The paper concludes by claiming that there is much to be gained by automating coursework management.

2 Problem and solution context

In the current climate, there is much pressure on all of us to increase productivity, but naturally we worry about allowing standards to fall. There is evidence, or at least a feeling, that more students are arriving at university who need some considerable hand-holding, yet staff have less and less time to do so. This was, and is, the context for the system described in this paper. At the outset, in September 1993, the author had just been (perhaps reluctantly) placed in charge of managing the entire first year laboratories, in addition to his existing duties. This appointment was necessitated by the start of proper semesterization and a major new syllabus: the finely-grained modularity of continuous assessment in the new syllabus made it impossible to 'manage' in the ad-hoc fashion previously enjoyed. The author was keen to use computers to maximize his efficiency in this job – hoping to make the machine do all of the donkey-work, leaving only the more sensitive issues to the Human. The system was developed from scratch during its first year of use (much on a 'night-before' basis) and the following year it was used, by another person, to manage the entire second year laboratories.

It has now grown considerably, and has been used to successfully manage 5 whole first years and 4 whole second years at Manchester (corresponding to over 200,000 student contact hours).

It has also been used since September 1995, at Durham University Computer Science Department to manage all first and second year coursework, and all lecture attendance.

ARCADE is an acronym for Administering and Running Continuous Assessment with Deadlines and Extensions.

3 Problem scope and size

The problem is familiar enough:

- There are 220 to 250 students in each year.
- Most students take perhaps 12 modules, chosen from, say, 24.
- Each module typically has 10 to 16 hours of scheduled laboratory per student, with different groups of students scheduled at different times. By laboratory, we mean any form of continuous assessment work (including pen and paper based).
- The scheduled hours of laboratory for a module are divided into timetabled sessions, with (usually) compulsory attendance, and some with deadlines.
- The students are divided into groups for each module taken, and different groups have their sessions on different days at different times.
- There are around 30 lecturers and 50 postgraduate students staffing the undergraduate laboratories, each with their own view, or small set of views, of the curriculum.
- There is one year manager for each year. A year manager may thus be required to coordinate and manage work corresponding to around 30,000 hours of student contact time in a year.

In addition to being able to manage this complexity, desirable features of any solution to the problem include:

- Consistency in administration and general organization across the many different laboratory modules. This is mainly to minimize the confusion of (especially) the students.
- The ability to easily monitor the whole progress of individual students, and automatically identify people who are struggling and/or missing. This is difficult if information is not centralized quickly.
- The regular provision of feedback to the students on their overall on-going *and predicted* performance – particularly to those who are falling behind. This can be very expensive in staff time, unless it is largely automated.
- Encouragement for the students to work consistently throughout the year, in both the scheduled sessions and in their own time, and with a sensible balance of effort between their different modules. Again this can be very expensive in staff time if not automated and/or inherent.
- An even spread of assessment effort over the year for the staff, rather than a huge mountain at a few times. This clearly links with the desire for regular feedback.
- The ability for a year manager to easily monitor the complete progress of individual modules, so that problems (such as an exercise being too hard) may be corrected immediately. This can be so time-consuming that it will not happen, unless the information is readily available in a summary and consistent form.

4 A discussion on deadlines

Clearly, some of the desirable features identified above can be achieved by having many deadlines throughout the year. This helps in spreading the load of assessment and encouraging the students to work evenly. It also provides the data for regular and timely feedback to the students. One of the traditional arguments against having many deadlines is the huge cost of their administration, and this is often used as the excuse for not having them. The development of ARCADE held the automation of deadline management as the highest priority.

Another argument often used against having many deadlines is that the students are adults and should be allowed to pace their own work. The author believes that when this isn't a disguise for the reason quoted above, it is a genuine view held from a rather naive and idealistic perspective. It is true that some students can be successfully treated in this way, but the author's experience is that the vast majority simply cannot (at least, not in a city as distracting as Manchester!). If, for example, one has a 10 week period with a deadline at the end and no other monitoring of progress, then the mode pattern of work will be to start in earnest in about the 7th week, and fail to complete the work. An alternative to many deadlines would be to introduce a regular monitoring of progress by a person that the student respects (e.g. his or her tutor). However, whilst this would be certainly less patronizing, it is simply unaffordable in most universities for first and second year coursework, whereas deadlines are relatively cheap.

Another argument against regular deadlines is that they are incompatible with the students undertaking a single large exercise or project. However, experience from Manchester suggests that even in this scenario, deadlines which are just ticks and are of the type to 'convince a demonstrator that you are N% through the effort', still work well to help ensure the students have a consistent pace (but of course, they do not spread the assessment load).

More evidence that students mostly work to deadlines is obtained from observing a typical laboratory session and deadline structure, of the kind used at Manchester for many years prior to 1993. In this we have a list of sessions with associated exercises which the students should in principle do at those times. However, each of these exercises has a deadline perhaps two weeks later. So, in effect, we give the students the same list of work twice, but with a two week difference in the dates. The author's experience observed over several years is that after a very short while the students simply ignore the first list and work to the second! This begs the question: why have two such lists? One answer is that it can allow a single deadline to be set for each exercise, regardless of which laboratory group the student is in. This only works fairly if the distance from the last scheduled session to the deadline is at least a week, say, more than the spread of time over which the different groups have their scheduled session. Having a single deadline per exercise is a great saving of head-scratching in situations when there are many groups, but only in the absence of a machine managing the deadlines for us.

5 Deadlines and late flags

In ARCADE there is no distinction between the deadline and the scheduled session for an exercise. A module is taken by a number of students divided into groups. The work for a module is divided into a number of sessions. Each group is allocated a date (and time) for each of the sessions. Sessions may be given a deadline. This means that the deadline for the work associated with that session (and previous sessions since the last deadline) is the end of that session, for that particular group of students. However, an individual student has a right to obtain an extension of that deadline without any penalty. The extension is normally to the *start* of the next session, to force the student to complete the work in their own time. If the student does not complete the work by the deadline, and either fails to obtain the extension, or fails to complete by the extended deadline, then that piece of work is flagged as being late.

Late work carries a heavy penalty, but one which inherently cannot cause a student to fail a module. The work is marked in the normal way, but when the overall mark for the module is being calculated, the late and non-late portions are computed separately and only the minimum amount of the late marks, if any, is used to bring the overall mark up to the pass level; and the remaining late marks are ignored. That is, if the non-late mark is already greater than the pass level, then all the late work is discarded. Otherwise, if the sum of the two is above the pass mark, then the pass mark is awarded. If neither of the above are true, then the overall mark is the sum of the late and non-late marks.

The author's experience of using this regime for five successive sets of first year students and four sets of second year, is that it works very well in achieving the stated aims of encouraging the best work pattern from the students. They quickly realize that it is in their interests to do much of the work for an exercise *before* its scheduled session, so that they can make use of the laboratory staff when they need to.

There is, however, an axiom here that the only (normal case) reason for students to have a late flag is that they are not working hard enough. (There are mechanisms for dealing with exceptions – see the next section). This is how it should be, but sometimes those responsible for setting exercises are slow to accept that they are expecting too much! The author believes this to be a good feature of the regime: it promotes the correct pitching of the difficulty of the work, rather than allowing a poor match to be hidden in the students' bad time-management. (However, see **Future enhancements**, below.)

6 Exceptions for special circumstances

In addition to the normal case described above, there are several mechanized exception provisions. A student may be excused for not attending the session, and this automatically grants him or her the extension. A student can be excused for not getting an extension, if for example he or she forgot to obtain one in the previous session.

A student can also be excused for not completing by the extended deadline, for which there are two levels of excuse. An ordinary excuse can be awarded by a laboratory supervisor, but it only lasts until a predetermined date, usually the session following the extended deadline, after which it automatically expires. A special excuse is one which does not automatically expire, (although it can be manually expired) and can only be awarded by the year manager. Typically, there are only a handful of students needing these, and it is believed to be a good idea for one person to be aware of all such special cases.

Excuses are stored and collated by student name to help identify those who have obtained more than their fair share of, for example, being let off oversleeping by lots of different laboratory supervisors!

6.1 Students who are struggling

Excuses are normally given for special circumstances, such as illness and bereavement, or even the odd oversleeping, etc.. One of the questions which has been explored many times is what should one do when a small number of students who appear to be trying hard, and have no special circumstances, miss a deadline and ask for more time? It is obviously important to minimize the risk of demotivating these weaker students, and it can be tempting to grant their wish. However, the author's experience of trying this has lead him to conclude that it is a mistake. When a person has suffered an exceptional circumstance (e.g. an illness), they typically work harder afterwards and often manage to catch up. On the other hand, students who are working at their normal pace yet falling behind clearly need to change the way they work and/or work (even) harder; and allowing more time does not alone achieve this. The author has developed the strategy of refusing to give more time to such students, but promising that if they meet an *unextended* deadline at a future date, then their late flag on the current deadline will be retrospectively excused. This has the effect of *increasing* the pressure, but without demotivation, and is usually effective. Of course, such a policy is not announced to the whole student body, to avoid the risk of the deadlines being undermined.

7 Data collection mechanism

One might be excused at first for thinking that the data collection requirements of the regime described above make it prohibitively expensive. Certainly there is a measure of necessary bureaucracy, but the real question is whether the gain and the work saved outweighs this; and the author believes it certainly does. The gains are very high, and the level of administrative overhead is less than most people expect.

The system produces a printed paper sheet for each laboratory session, containing the module name, the session name, the group name, and the name of each student who should be in that session, along with various columns for the data collection. For a session with a deadline, these columns are attendance, completion by deadline, extension, completion by extension, date of completion (for work completed even later) and marks. An example can be found in figure 1.

The whole set of sheets for a particular module are stored in a ring binder, organized by group. These sheets are completed by hand during the laboratory session (and at other times – for example after a bout of off-line marking). This is not onerous: in the vast majority of cases the data entered for a student is just a series of ticks and a mark. For example, one of the two most common cases is a tick for attendance, a tick for completion and then a mark. The other most common case is a tick for attendance, a blank for completion (later interpreted as a cross), a tick for extension, a tick for completion by extension, and a mark.

In between laboratory sessions, the hand-written data is entered on to the computer, which then prints updated versions of the sheets to be swapped for the old ones in the ring binders. Thus at the start of a laboratory session, when one is collecting work from the previous session, the laboratory staff are adding data by hand to a sheet which already has printed data on it.

For obvious reasons, a high priority in the development of the system was to optimize the user interface to the data entry program. In the first year of operation (whilst the system was being developed

1511 A+Ext+I+L page 2

Please do not write in black ink on this sheet

Year 1, Course 1511, Group A+Ext+I+L

Lab 2D (Simple input and output) (1/11), Page 1 of 1 (24 students)

†Ordinary excuses recorded here expire after Friday 14/11.

Name	Attendance 9am Fr 1/11 (✓/×/E)	Completed by 11am 1/11 (✓/×)	Request ext. to 9am 7/11 (✓/×/-/E)	Completed by 9am Fr 7/11† (✓/×/E)	OR comp. date (-/Date)	Exercise mark (/10)
009,Antarwillas,Zaibali F	✓	×	✓	✓	-	2
023,Biset,Namanda	✓	×	✓	✓	-	6
028,Bolder,David M	✓	×	✓	✓	-	14
032,Braggs,Beth	✓	×	✓	✓	-	10
047,Chan,Phong	✓	×	✓	✓	-	10
223,Coase,Martin	✓	✓	-	✓	-	13
053,Cordell,Andrew J	✓	×	✓	✓	-	6
073,Fung,Chi Wong	✓	✓	-	✓	-	10
075,Garrison,Wilfred S	✓	✓	-	✓	-	10
078,Good,Jonathon	✓	✓	-	✓	-	13
088,Hardy,Mark I	✓	✓	-	✓	-	8
097,Husain,Bhawin	✓	✓	-	✓	-	9
127,Majil,Najma	✓	×	✓	S	21/11	7
137,McGovan,Richard V	E	×	✓	E	?▷	-LATE
153,Parsons,Kathryn J	✓	×	✓	✓	-	10
163,Rajarat,Steve H R	✓	✓	-	✓	-	13
165,Rehmurzi,Saweb	✓	×	✓	✓	-	8
182,Shah,Alkesh B	✓	×	✓	E	14/11	8
183,Shah,Nishma P	✓	×	✓	✓	-	10
184,Sharpe,Colin A	✓	✓	-	✓	-	10
188,Stone,Nicola M	✓	×	✓	✓	-	10
192,Southman,Andrew	✓	✓	-	✓	-	10
205,Waeed,Mustawa I	✓	×	✓	✓	-	9
248,Zieplowska,Paulina	✓	×	✓	E	21/11/97	10-LATE

Figure 1: Example laboratory session data-collection sheet

This is for a session occuring on the 1st November, with extensions until the 7th November and ordinary excuses expiring at the end of the 14th November. Note the non-expiring (special) excuse given to student 127, and the expired excuse for student 248. For privacy of the students, the text of excuses is not printed on the sheets, but is stored in the database.

simultaneously) the author deliberately took it upon himself to enter all of the data, with the result that the program is easy and fast to use. The user is presented with a form similar to the paper sheet, and most data entry is by single key stroke. For example, a single '/' key stroke records attendance, the cursor then moves to the next column where another '/' records completion by deadline, so the program puts a hyphen in the extension column, a tick in the completion by extension column and a hyphen in the date column, and moves the cursor to the mark column. The user could then enter the mark (if it is available) and upon pressing the enter key the cursor moves to the next student. And so on. Additionally, the space bar continually changes its function so that pressing it causes the next most obvious thing to be done – for example, to move the cursor to the next place on the sheet where data can be added.

To make data entry even faster, the computer can read out a 3 digit code uniquely identifying the student over which the cursor lies. This code also appears on the printed sheets, and thus much of the data can be reliably entered without the user having to look at the screen.

In the author's department, all the data for first and second year laboratories is entered by one member of clerical staff, who simultaneously front-ends queries to the computer systems administration staff, runs all the student printer services and the 'Student Resource Centre': handing out lecture notes, tutorial answers for photocopying, etc.. She is no more overworked than any other typical member of University Staff!

Despite the above claims, one may still wonder why such old technology as paper is used to collect the raw data. Two possible alternatives would be to allow the computer to collect (some of) the data directly from the students, or to allow the laboratory staff to enter data directly onto computer.

The author believes that collecting data directly from the students would be a false economy, probably leading to a higher academic cost than would be saved in data entry. The main concern is the danger of a student being able to escape interaction with a member of the laboratory staff. Such interaction is crucial to the well-being of, particularly weaker, students, and the collection of data is a simple mechanism to ensure that every student gets spoken to. For example, it is very common for a student to tell you he or she is struggling whilst you are taking his or her attendance.

As for laboratory staff entering data directly onto computer, this will only be practical when hand-held A4-sized note-pad machines are cheap enough to be provided to every member of staff. Meanwhile the portability and random-accessibility of a piece of paper beats asking the students to queue up at a workstation in alphabetical order!

8 Extrapolation of performance

A crucial feature of ARCADE is its ability to predict from an early stage which students are having difficulty, or are being lazy, and which are performing to a suitable standard. The approach used here is to compute an overall mark for each module for each student, right from the beginning of the semester or year. Then as time goes by, the mark gets more and more accurate and by the end it is the final mark. This approximation is based on two dimensions: which pieces of future work will the student complete, and what the marks for them will be. No attempt is made to *predict* whether future work will be flagged late or not, as the only point of the extrapolation is to determine which students are in danger of failing and late flags cannot make a student fail.

Predicting whether a student will miss a piece of work not yet handed in is done as follows. If the work is already flagged late, we cautiously assume it will be missed. Likewise if it is excused, but the extension date has passed. Otherwise, we predict that it will be missed if and only if the previous two pieces of work are either missed or predicted to be missed.

Mark prediction is even simpler: if the student has had no pieces of work marked for that module, the mark for each piece of work completed (but not yet marked) or predicted to be completed is assumed to be 70% of the maximum available. If the student has had one piece marked, the prediction is taken as the average of 70% and that mark. Otherwise, the prediction is taken as the student's average for those pieces that are marked for that module.

Although clearly very crude, this extrapolation is remarkably effective in automatically spotting those students in danger of failing. All students start off with a predicted result of 70% (before scaling). Those, for example, who miss the first two deadlines suddenly get a predicted mark of zero. Those who miss one, and do badly in the next one may fall under the pass level, and so on.

9 Student irregularities feedback

Another crucial part of the system is the provision of automatic feedback to the students through email messages. Essentially, this involves the student being informed of all the 'irregularities' in their laboratory data: absences, lack of needed extensions, missed extensions, excused missed extensions (together with expiry date if appropriate), and so on. One of the most effective pieces of feedback is the inclusion of failure predictions for those students whose predicted mark is less than the pass level in some module. The feedback can also complain about the failure of the student to archive their work (e.g. for plagiarism detection) where appropriate.

The purpose of these messages is two-fold: to check that the student agrees with the data (it may be that the student has tried to give an excuse but it has not been recorded properly, for example), and/or to give the student the necessary prod to get their act together! The email messages can be copied to the student's personal tutor to enable him or her to be made aware of the students difficulties.

An example message can be found in figure 2.

In the early days of ARCADE, these email messages were sent approximately once per week, to all students who had any irregularities. It was noticed that some students began to ignore the messages by deleting them without even reading them. So, the strategy was changed to a process which ran every day, emailing only those students whose list of irregularities was different from the day before. This was an improvement, but lead to some students getting emails several times a week! The current approach is to run the same process, but only once per week rather than daily. This seems to work the best of the approaches tried. (See **Future enhancements**, below.)

A related feature, which was added after a couple of years, is the provision to laboratory staff of a printed copy of the irregularities pertaining to the students in a particular laboratory group. This appears next to the data sheet for the current laboratory session, so that the staff can easily see what problems each student is having in their laboratory while taking attendance. This enables timely intervention and help by staff when students are struggling, or the correction of laboratory data errors at source, whichever is appropriate. It is clear that those laboratories in which the staff use this information run more smoothly than those where the staff ignore it.

10 Management aspects

The structure of the laboratory is table-driven under the control of the year manager. The number of sessions, groups, deadlines etc., are configured at the start of the year or semester. Each deadline can be given an arbitrary maximum mark, as suits the marking scheme of the person who designed the exercise, and also an arbitrary integer weighting. The advantage of the latter is that during the year, or towards its end, the year manager might decide to alter the weighting of a particular exercise – perhaps because it was too hard, or too easy, without having to alter any of the individual student's marks.

ARCADE readily provides the year manager with many results and tools. For example, he or she can see a one-page summary of what needs doing at any time, showing how much work is expected, needs marking, how much data needs entering, how many unresolved queries there are, and so on.

To assist in the monitoring of the various modules, the manager can see a report containing a one-line summary for each module. This gives information like the current percentage (predicted) failure rate, the percentage of students with irregularities, and attendance statistics for each module.

The system collates all the (predicted) marks for each module into a single table, and produces a summary for each module and a histogram with an indication of the students' pattern of working. Figures 3 and 4 show the summary and histogram respectively for a module which many students found difficult.

At the end of the year or semester, the manager can request a single file containing all the overall laboratory marks for the year: each tuple in the file relates a student, a module, and a percentage mark. (This is provided so the laboratory marks can be sucked into another system which combines them with examination marks, etc..)

The system can also provide lists of marking outstanding, or work which requires demonstrating by the student, or work expected, attendance summaries, etc.. It can provide timetables for modules, groups and even individual students. It has some provisions to assist in the allocation of laboratory demonstrators to sessions.

```
Carefully worded general introductory message goes here, including a
reminder that the data could be wrong and the student should not worry if
this is the case, but just help correct it.
....
-------------------------------------------------------------------------------
As you will see below, you have a warning that you are predicted to fail one
or more modules. This means that the prediction algorithm which tries to
extrapolate your performance up to this point in each module has projected
an overall mark below the normal honours pass level. This may be due to
missing work, or low marks. How seriously you take the prediction depends
upon how far through the laboratory schedule the prediction is made: after
only one deadline it typically contains a large proportion of guess-work,
whereas by the end it is entirely accurate (based on the data it has).
However, it is most useful early-on as an indicator of the likely outcome IF
your current performance is CONTINUED.
-------------------------------------------------------------------------------
bloggsf Warning: on current performance you are predicted to fail 1412!
bloggsf Z 1211 Lab 4D (20/11) Attended, Extension to 4/12, Excused further,
        BUT: Still to be done.
bloggsf Z 1222 Lab 2.3D (30/4) Attended, Extension to 14/5, Excused further,
        BUT: Still to be done.
bloggsf Z 1412 Lab 3D (3/3) Attendance excused, Extension to 17/3,
        Excused further, BUT: Excuse expired on 21/4, Not done, Late.
bloggsf Z 1412 Lab 4.2D (21/4) Attended, Extension to 5/5, Excused further,
        BUT: Still to be done.
bloggsf Z+B 1031 Lab 3D (13/11) Attended, Extension to 27/11,
        BUT: Done after extension, Done (on 4/12) but Late.
bloggsf Z+B 1031 Lab 4D (27/11) Attended, Extension to 11/12,
        BUT: Not done, Late.
bloggsf Z+M 1021E Ex-class 1 (6/10): Absent on 6/10.
bloggsf Z+M 1021E Ex-class 4 (27/10): Absent on 27/10.
bloggsf Z+M 1021E Ex-class 5 (3/11): Absent on 3/11.
bloggsf Z+M 1041 Lab 7D (9/12) Attended, Extension to 16/12,
        BUT: Not done, Late.
-------------------------------------------------------------------------------
Notes: Each line of summary consists of your email name, your group, the
       module name, the session/exercise name, the date of the session and
       finally your "irregularities" with respect to that session.

"Absent" means you were not there at the session, and you have no excuse
    recorded.
"Done after extension" means you completed the work after the possibly
    extended deadline.
"Not done" means you haven't completed the work at all.
"Still to be done" means you have an excuse for being late, and you are
    expected to complete it soon -- there may be a specific date mentioned.
"Excuse expired" means you did have an excuse for not doing it by the
    extension, but it no longer applies. However, please note that extra
    extensions awarded by laboratory staff are usually only valid for one
    additional session. If your problems are such that one session more was
    insufficient, please get in touch with the laboratory manager directly.
"Late" means that the marks for this work will not be added to your overall
    mark for the module once it has reached 40%.
```

Figure 2: Example student feedback email message

Course Summary for module 1001
1001 Final dynamic scaling factor (range 45%-65%) is 0.86

Deadline	1D	2D	3D	4.1D	4.3D	5.2D	Total	Marked
Weighting	10	10	10	5	20	20	75	
Maximum	10	10	10	5	20	20		
Name	1D	2D	3D	4.1D	4.3D	5.2D	Total	Marked
Done	180	180	176	175	173	112		
- by deadline	150	64	31	16	44	10		
- with extension	24	97	110	118	76	95		
- with excuse	2	12	11	7	20	2		
- late	4	7	24	34	33	5		
Still expected	0	0	0	0	0	0		
- on time	0	0	0	0	0	0		
- late	0	0	0	0	0	0		
Missed	3	3	7	8	10	71		
Marked	180	179	174	159	168	112		
- no. >= 40%	180	179	172	163	163	103		
- no. < 40%	0	1	2	0	6	9		
- average (%)	98.2	86.6	91.5	94.9	85.6	68.5		
- minimum (%)	40	30	35	40	15	20		
Assumed	0	1	0	12	1	0		
Unmarked	0	0	0	0	0	0		
Average (%)	96.6	85.2	87.9	89.8	80	41.9	60.39	64.05
No. >= 40%							172	
No. < 40%							11	
Non-zero Average (%)	97.7	86.2	88.9	90.8	80.9	42.4	61.06	64.76
Late marks (%)	2.13	3.22	10.90	16.22	12.68	1.61	6.07	
Lost late (%)							3.65	

Figure 3: Example course summary of a ‘difficult’ module

The results show that as the module progressed, the number of students completing the work reduced, the time taken to complete increased, and the average marks, even for the work completed, reduced. This evidence would suggest that the module is too hard!

```
Histogram for module 1001

1001 183 students. * = 2 students. Ave. = 60.39. Scaling (45%-65%)= 0.86.

 0- 4 |   2 | *                                    |   Work pattern
 5- 9 |   0 |                                      | 1 ################***.
10-14 |   0 |                                      | 2 #######***********=-
15-19 |   1 | *                                    | 3 ###************=---.
20-24 |   1 | *                                    | 4 ##*************=----
25-29 |   2 | *                                    | 5
30-34 |   1 | *                                    | 6 #####********==----.
35-39 |   3 | **                                   | 7
40-44 |  38 | *******************                  | 8 #**********-........
45-49 |  11 | ******                               |
50-54 |   6 | ***                                  |
55-59 |  17 | *********                            |
60-64 |  17 | *********                            |
65-69 |  10 | *****                                |
70-74 |  19 | **********                           |
75-79 |  23 | ************                         |
80-84 |  26 | *************                        |
85-89 |   6 | ***                                  |
90-94 |   0 |                                      |
95-99 |   0 |                                      |
100   |   0 |                                      |
```

Figure 4: Example histogram of a ‘difficult’ module

The bunching around the 40% mark shows the effect of the late flags in this extreme case. The work pattern clearly shows how, in this particular module, the students’ ability to cope with the deadlines dropped off as the sessions went past. Each character in the work pattern of a deadline represents 5% of the students. # means completed by deadline, * means completed with extension, = means completed late with an excuse, - means completed late without an excuse and . means missed. Sessions 5 and 7 did not have deadlines.

```
1       stud "  ABIS    " -Email | select-specific ABIS
2
3       for Module in `list-arcade-modules`
4       do
5         table -s ABIS $Module  > /tmp/jtl-$$
6         if test -s /tmp/jtl-$$
7         then
8           cat /tmp/jtl-$$
9           echo -n ^L
10          echo ABIS attendance for $Module
11          echo
12          absence-summary -s ABIS -t 100 $Module | stud - -d -Name
13          echo -n ^L
14        fi
15      done
16      rm /tmp/jtl-$$
```

Figure 5: Example user-defined shell script to sit above ARCADE

11 Architecture of ARCADE

ARCADE is implemented under UNIX, and currently only runs on SUN4 hardware, although there is a hope to port it to LINUX in the very near future. There are no plans to develop a Windows version.

It has grown considerably since 1993, and now consists of 29,000 lines of 3GL code and 7000 lines of shell script. Due to its humble and prototypical beginnings, it is not built on a commercial database package. At present, an M.Sc. project is investigating the use of ORACLE to re-implement (part of) ARCADE.

The system is designed to be used directly by a small number of trusted and trained users – these being, for example, the laboratory manager, data entry clerk, year tutor, etc.. These users can safely access the system concurrently, perhaps having to occasionally wait for ARCADE to release a resource lock. It has an open architecture based on the UNIX philosophy of powerful tools which can be used directly or composed together at the command line. Hence, with the exception of the data entry and student detail editing programs, which both have a sophisticated and efficient interface, there is no user interface for ARCADE: each job is a UNIX command.

There are two reasons for this. Firstly, the development effort has concentrated on functionality rather than posh-looking GUIs that in any case slow down the expert user. Secondly, and more importantly, a command line interface gives the more flexibility in use. As an example of this, figure 5 shows a simple shell script to produce particular results required by some other department for all students registered for the degree 'ABIS'. This was was written by the author, but acting as an ARCADE user rather than it being part of ARCADE.

Line 1 identifies the ABIS students, using the ARCADE command 'stud', and makes an ARCADE selection named 'ABIS' using the command 'select-specific'. Line 3 starts a loop, setting the value of 'Module' to each of the modules managed by ARCADE, as output by the command 'list-arcade-modules'. Line 5 uses the command 'table' to produce the final details for the particular module for all students in the selection, placing the possibly empty results in a temporary file. Line 12, which only executes if the table produced was non-empty, has the command 'absence-summary' producing the attendance data for the given module for the selected students, but using 'stud' to replace the email names with real names.

12 Success of ARCADE

It is clearly difficult to be completely objective when measuring the success of a system such as this. However, the consensus of opinion of all the staff who have been involved in its use (module managers, laboratory supervisors and post-graduate demonstrators) is that it is a great success, and this opinion remains after five years, even though memories of previous chaos are getting dimmer!

There is a strong feeling that the students actually work much harder than they did in the days before ARCADE: and that despite the extra laboratory load imposed on them by modularization, they have less problems fitting it all in.

To date, no effort has been made to actively solicit the opinions of the students on a wide-spread basis. The problem here is that the majority of students have no other experience of university education, and all of their coursework is managed under the same regime. A few students who changed from other departments have noticed the difference, but seem to accept it as fair. The only complaints brought to the attention of the author have come from a small number of students who were unhappy about the automatic dynamic scaling factor (a feature which can be enabled to even-out the overall module averages into an acceptable range). These students were unhappy because they wanted to get 100%!

One objective statement that can be made is the measure of actual laboratory session attendance. In the academic year 94/95, the second year of using ARCADE, the first year attendance was 89%, with an explained absence of 6%, and an unexplained absence of 5%. The corresponding figures for the second year students are 82%, 5% and 13%. Subsequent years have had similar levels of attendance. Figures for the previous regimes are not available, but those who were involved typically state their educated estimate of second year laboratory attendance in previous years as starting at around 75%, dropping instantly to around 40% and then moving to 100% in the last fortnight!

13 Problems with ARCADE

There were of course some problems when ARCADE was first adopted. For example, some of the staff were sceptical that the bureaucracy was necessary. However, careful diplomacy persuaded them to reserve judgement, and they eventually became convinced of both the low cost and the high gain.

On a related issue, another difficulty was persuading the staff of the need for their laboratory data to be kept accurate and up to date. This teething difficulty resulted from a lack of their appreciation of the *power* of the system, and thus the consequences on student morale if data is wrongly recorded (e.g. an excuse not written down). Nowadays, as some of them become a little complacent, the occasional reminder is needed.

Related even to this, was the need to explain to the students that mistakes in their laboratory data will happen, and to ask them to help sort them out rather than get upset about them. The students are warned right at the start of the first year that they may get email messages containing erroneous irregularities. On a similar vane, the wording which introduces the irregularities messages has evolved over the years as insight has been gained.

Perhaps the most difficult problem in the early days, was reassuring some of the post graduate demonstrators who were affected greatly by the *success* of the system. This was especially true for those who worked in the second year laboratory. The previous regime had few deadlines, and so the demonstrators were used to chatting and drinking coffee in half-full laboratories during most of a semester, but with a mad panic at the end. This was so radically changed – they were kept busy from the start – that some of them worried they would be totally unable to cope if the work pattern followed the same increase as previous years. Of course, it did not!

14 Enhancements since 1995

ARCADE has been developed considerably since it began, with some features which may have been envisaged from the start but which were not central to its success, and others for which the insight has been gained through experience.

For example there is a facility to cut down chaos at the start of a semester caused by students tending to change their options and turn up to laboratories they're not registered for. The laboratory manager can specify that certain modules have a 'Non-Registered' group for the first one or two sessions. This group contains all of the students in the database who are not registered to take that module, so there is somewhere ready printed to record their lab data if they do turn up.

Related to the above, experience has shown that students are not always quick to recognize the need to tell their department that they are intending to change modules, and this leads to them having a somewhat different view to that of the department as to which modules they are registered for. A feature was added to ARCADE that informs each student, by email, of the details which the system database has about him or her. This can be sent automatically as it changes, so the student can check the accuracy.

ARCADE even distinguishes between confirmed and unconfirmed module registrations. When a student *appears* to be taking a module which they have not officially registered for, the laboratory manager can set their registration to be unconfirmed. The student may then automatically get an email about it that night.

There are times when the laboratory manager wishes to make a note pertaining to a particular session for a particular student. Originally, the only mechanism for storing such notes was by incorporating them in the text of any excuses awarded. A feature was added which allows a query and answer to be associated with a student's record for a session. ARCADE keeps track of queries which have no answer, so this mechanism acts as both a permanent record of notes, and a reminder of things to be followed up. A good use of this, for example, is when one suspects that a piece of work was copied and the student is required to come and talk about it. At the time of suspicion being aroused, one would place the query on the work, and when the matter has been resolved, one would record the answer. Both the query and answer would appear on the student's record when it is printed or emailed to them.

The ability for ARCADE to produce timetables is an extremely useful add-on feature. This was motivated by the annoyance of having to express the laboratory timetables in ARCADE, and also write them separately in a laboratory manual to be given out to the students. Checking that each was consistent with the departmental timetable (in terms of slots in the week) and each other (in terms of actual dates in the year) was a laborious process. ARCADE can now be used to produce the timetables for publication, even writing them for immediate inclusion in a latex document! A spin off from this is the ability to produce timetables for individual students, or individual modules, etc., the latter being useful to help members of staff turn up to the right place at the right time!

Another recent enhancement, which can be taken further, is assistance in allocating laboratory staff to laboratory sessions.

A curiously successful new application of ARCADE has been in using it to email tutorial attendance prompts to tutors on the morning of their tutorials. The tutor merely has to reply to the email after the tutorial, and edit an attendance code next to each tutee's name. This has had the effect of raising the reporting of tutorial attendance to 100%! The modifications to ARCADE to add this feature took only around an hour to implement.

As insight has been gained, and confidence in ARCADE and its strategies has grown, so has a desire to make actions happen without the need for any user interaction (such as the regular emailing of student feedback). ARCADE has the ability for users to run commands automatically under any user-defined conditions at any user-determined time. The maintenance of a user's 'remote-jobs' is easy, and a log file is kept so a user can see what has been done if they wish.

15 Future enhancements

Development of ARCADE is on-going, and there is still much to do. The author currently has a list of some 150 enhancements planned for the future, varying from trivial to very complex. Some of these, in no particular order, are listed below.

There is currently no special software for editing the configuration file describing the structure of the laboratories! This is stored in a text file and carefully altered with a standard text editor.

Perhaps the most complex intended extension is a provision for running unit-based laboratory courses. The idea here is that there would be a number of equally weighted units, or 'milestones' making up the total work for one module. In order to fit in with a deadline structure, the number of units would be a whole multiple of the number of laboratory sessions. Each deadline would require a student to hand-in at least some number of units, and at most some greater number of units. For example, a module might have 10 sessions, with 20 units. The minimum hand-in for each session would be one unit, the maximum would be two. The problem with this scheme is the management of which units correspond to which deadlines: this would be different for each student depending on his or her previous pattern of work!

It is certainly intended to tie-up ARCADE more closely with plagiarism detection software (currently it can be made to complain that a student has failed to proffer a piece of work for such detection as part of his or her feedback). Future plans include managing the process of negotiation with 'accused' students, and even automatically emailing them about suspicious work before a human has looked at it!

Similarly, it would be very easy to integrate the system with automatic assessment software, and this is planned for the near future.

Much consideration has been given to the effect on the student body when the late penalty system exposes that the work is too hard, but nothing is done to reduce it. This is manifested when large numbers of students miss extended deadlines. In the discussion on deadlines and late flags above, we said that an original axiom of ARCADE is that, in general, laziness should be the only reason for a student getting a late flag. When this is not true, the effect on student morale can be very negative, even more so if they cannot skip out some deadlines in order to meet later ones. Unless one deals with this very carefully, the students tend to give up realising they have nothing to gain by trying hard, once they have enough marks to pass. Of course, the real solution is to make the work more realistic. However, when this does not seem to be possible, an alternative kind of late penalty may help: that of 'graduated lateness'. This means that the amount a piece of work is flagged late grows the later it gets, rather than all of it being late immediately. The penalty would still not be expressed by deducting marks on individual exercises, like in many schemes, as one still does not want a student to fail overall merely through being late.

Another intended enhancement is to give the students more 'ownership' of their data, by providing a mechanism to allow them look at it whenever they choose, rather than to wait until the laboratory manager emails it to them, or to have to ask for it. To this end, a recent development has been a client and server to allow a wider access to ARCADE, but the work is not yet finished. As this will be provided to non-expert users, a nice GUI is required. This client will also provide students with a structured way to respond to irregularities which they believe are erroneous. Currently, they either email the laboratory manager and typically give insufficient information, or they speak to the laboratory staff who may fail to solicit the exact information. Instead, the client program will ask them the right questions as appropriate to the irregularity concerned. A similar client-server is planned for staff use, so they can directly access information on their modules, or their tutees, etc..

As was stated earlier, the current strategy of sending irregularities once per week only to those students for which they are different from the week before, seems to be the best of the approaches tried. However, there is a feeling that even this can be improved on, and a future version will split the feedback into sections, one each for new irregularities since last sending, ones that were there last time but have changed, ones that are unchanged since last time, and those which have gone. This is based on an impression that many students do not keep a record of the feedback they are getting, so can find it hard to see in what way their irregularities have improved or got worse.

Related to the above is an idea to allow a student to confirm that he or she is aware of a particular irregularity, and thus for it no longer to be included in the automatic feedback. This is motivated by the odd student in the past who has been irritated by being told of a piece of work missed that he has known about and cannot do anything about now. However, the need for this may be somewhat undermined by the above plan to split the feedback into sections.

Much of the interaction between the laboratory manager and individual students is done through email, and ARCADE already provides several email sending tools. A future version will provide dedicated email management and storage facilities to make the process of *receiving* email less time consuming for the manager.

16 Comparison with Ceilidh

At the start of the development of ARCADE, the author was not aware of the availability of any other generally applicable coursework management software. However, perhaps one of the most obvious apparent alternative is the Ceilidh system from Nottingham University ([Benford,1993]). Whilst the author has no intimate knowledge of Ceilidh (and sincerely apologizes if this relative ignorance leads to a misrepresentation in this paper), the two systems do appear to be completely different at first glance.

The most obvious difference is the order of priority of the developments. One gets the impression that Ceilidh started as software for the automatic marking of programs, then became able to manage programming laboratories, and then was generalized to be capable of managing other types of course. If we consider that by such development a system can eventually run an entire bounded world, then certainly one could observe the two systems starting at opposite corners and progressing across the floor of such a world. A concrete example of this difference is that ARCADE runs deadlines automatically (and they are different for the different groups of students, and can be systematically extended and further extended), but has absolutely no 'knowledge' of the actual work done by the students, except for the title of each exercise. In contrast, Ceilidh is given intimate knowledge of the exercises (in the sense that it can then mark them), but the deadlines are published as text rather than held as data, are the same for all students, and are enforced by the appropriate person actually pushing the 'deadline' button at the publicized time (at least, that was so in 1993).

There must be a reason for this difference, and the author suspects it is simply a result of the project developers starting from different places. ARCADE was developed by a laboratory manager whose job it was to manage a whole year-full of different modules with different types of work associated with them. Whilst he was also involved in marking work in some particular modules, it was the management job he was trying to automate. One might expect an admission that the original motivation behind Ceilidh was the same in general – to reduce workload – but the development was started by someone trying to avoid marking programs, rather than avoid shuffling lists of names, chasing naughty students and adding up numbers!

In that sense, they are the same; and they both succeed because they take an existing manual system which works and automate a part of it in a way so that it still works, but works faster and/or with less effort.

This success is enhanced by another (subjective) factor which they both share: the 'big brother' syndrome. The theory here is that the students (by and large) work more efficiently than they might otherwise do, simply because they feel that their department is able to effectively monitor them, and they want to get good marks.

Another obvious difference between the two systems is that Ceilidh is further developed and more highly polished than ARCADE – Ceilidh started in 1988, has had many people working on it and has had some external funding. It is hence freely available and ready to use. By contrast, the developer of ARCADE is just one man, who does not even have a dog!

17 Conclusion

This paper has described a deadline regime and a supporting system for efficiently managing coursework undertaken by large numbers of students in a number of differently natured modules. It has also made the, difficult to substantiate, claim that the system is very successful. There certainly is the widespread belief among the staff of the author's department that the system not only manages the chaos, but simply smooths away nearly all of the administrative overhead of the various module managers, and at the same time, actually achieves a substantial increase in the work effort of the students.

ARCADE was designed to manage and monitor the work and marks of students independently of the kind of academic work undertaken. This common framework can be augmented with sub-systems designed specifically for the support of certain topics (e.g. plagiarism detectors for programs, machine assessments, etc.) were appropriate. The author believes the system could, perhaps with some modifications, be used in many other departments and even other disciplines. He was very encouraged at the small number of modifications that were needed to make it suitable for use at Durham University Computer Science Department and its continued success there.

However, the point of this paper is not to sell ARCADE. The primary message is that computerized work management can be very cost effective and is an attractive option for any large department to consider. The author believes that the development effort he has spent on the system is far less than the work saved by his department in general, and the work he has saved himself as a laboratory manager. With increasing pressure to gain greater efficiency, we can feel we are forever wringing blood from stones. This, however, is an example of a stone which turned out to be a tasty nut worth cracking.

References

[Benford,1993] Steve Benford, Edmund Burke, Eric Foxley, Neil Gutteridge, Abdullah Mohd Zin. *Early Experiences of Computer Aided Assessment and Administration when Teaching Computer Programming.* Department of Computer Science, University of Nottingham, U.K.. Email: ltr@cs.nott.ac.uk

Appendix: Some ARCADE commands

The following list of some of the ARCADE commands gives an indication of the scope of the system. These are all UNIX commands which can be run directly, embedded in scripts or placed in a user's 'remote-jobs' file to be run automatically, as appropriate.

```
absence-summary........................................show attendance summary
bad..................................................show 'bad' irregularities
blank................................................show where data is needed
checking..............................report on or switch data checking status
course-summary............................show summary statistics of a module
data.......................................................enter or browse data
doing....................................show numbers of students on module
due.................................................show what needs to be done
edit-privileges......................edit user privileges for current database
edit-sheet-details.................edit sheet-details describing lab structure
email............................send email to a group / selection of students
excuses.....................................................show excuse details
expected...................................................show expected work
fails.........................................show students who are failing
final-diffs...................show differences between two sets of final marks
full-story.................................show full lab details for a student
group-info.............show module group, students and staff hours information
help...................get help on ARCADE commands or UNIX commands/facilities
histogram..........................................show histogram of a module
list-all-modules..........show names of ALL modules taken by 1 or more students
list-arcade-modules..............................show names of ARCADE modules
mail-bad-summaries.......send automatic email feedback to students (and tutors)
mail-full-stories.............................email students full lab records
mail-name-list..............................email student details to students
mail-report.........email a report to the students in it, with an introduction
make-email-script...make an email script for a module (and group), or selection
make-final-marks......make final (or predicted) marks for all completed modules
marking.......................list sessions which have marking or demo needed
name-list-edit...............add new students or edit existing students details
name-list-trace...........................trace changes to a student's record
name-list-update..........create or update student details from external files
omitted.................................show where data is (probably) missed
out-of-range............................list sessions with out-of-range marks
outstanding.........................................show oustanding details
print-course-summaries.................................print course summaries
print-full-stories............................print students full lab records
print-histograms.....................................print module histograms
progress................................report simple overview of all modules
queries................................list sessions with unanswered queries
read-mail............................................read email from a student
remote-jobs-control.................set up jobs to be done remotely by crontab
restore-all-data...................................restore data from back-up
scaling....................................show scaling factors for modules
sessions-table.........................................show sessions timetable
sheets.......................................................print session sheets
stud.........................................................show student details
supervisors-report...................show 'bad' irregularities for a lab-group
table..............................show tables for certain students or modules
unchecked...................................list sessions with unchecked data
```