



MODULAR PROGRAMME ASSESSED COURSE-WORK SPECIFICATION

Module Details:

Module Code: uqc109s2	Module Title: Computer Networks & Operating Systems	
Module Leader: Nigel Gunton		
Module Tutors: Ian Johnson John Counsell Nigel Gunton Fred Stinchcombe Graham Darke		
Assignment CW2	Element Number: Weighting 25%	Total Assignment Time: 12 hrs + lab time

Dates:

Date assignment issued to students: Weds. 12th February	Date for return of marked work: w/b 19th May
Submission Place: post-box in N foyer, below the North stairs	Date of Submission: Thurs 10th April
	Time of Submission: 10.00am

Deliverables:

As listed on the Assignment spec sheet

Requirements :

The design and development of an RFC2324 compliant Coffee Pot Server.

Most modern applications include network or Internet connectivity. In order to gain a better understanding of the role of the operating system and networking software in supporting such applications, you are to design and implement a standards compliant server. This will support HTCP, the Hyper-Text Coffee-Pot Control Protocol. In order to achieve this you will need to :-

- i) Read and understand RFC2324. This provides an introduction to RFCs (Request For Comments). RFCs are the standards documents of the Internet, all key protocols are described and defined in them. Any networking software is required to comply with them if it is to interact correctly with other software using the same protocols. Note that, as with all protocols during their development phase, there are some ambiguities in RFC2324. It is up to you to recognize and interpret these ambiguities and omissions in such a way that they reflect the intent of the RFC. This should be documented as the first part of the deliverables. Credit will be given for the quality of decisions made at this stage.
- ii) Derive the set of messages and responses that may be generated between the client and server. These will be encapsulated within HTTP and should include any extensions to HTTP that may be required at both client and server end. These should be documented and again credit will be given for the standard of the documentation
- iii) Develop a full top-level design for both the client and the server. You will be expected to use a recognized design methodology such as UML. It is recommended that you consider designing the server as a state machine, (statechart diagrams in UML). Credit will be given for the quality of the design and can be offset against implementation, although at least a basic implementation will be expected.
- iv) Implement the design using either C or Java. There will be lab support for C only. Other languages will be considered but must be agreed upon by your lab tutor. You will be expected to use the code examples provided, it will be up to you to decide which of the examples are the most appropriate. These are all available via the links on Ian Johnsons web-pages ... <http://www.cems.uwe.ac.uk/~irjohnso> . Note that these pages are only available on-site.
- v) For the terminally curious only. You could consider modifying a web browser to support the coffee-scheme URL and application/coffee-pot-command MIME-type. This would allow browser URL requests of the type
`coffee://fullstrength.brew.org/pot-1?"part-skim"`
with an appropriate message body. The 'Dillo' web browser is highly recommended for modification as it is well written and commented and provides a good example of a threaded network application. See <http://dillo.auriga.wearlab.de/> for the source code. (Hint: It takes a trivial edit to 4 lines of code to support the 'coffee' URI).

NOTE:

Your coffee-pot server is not required to provide all additions but must recognize the additions list and respond with an appropriate message.

Deliverables:

- 1) A short (< 500 words) description in your own words that shows your understanding of RFC2324. 0% to 10%.
- 2) Documentation describing the messages and responses of your system. 0% to 10%
- 3) Your design(s). 0% to 30%
- 4) Your code for the client and the server. This must be signed and dated by your lab tutor to whom it must have been demonstrated successfully. The lab tutor must indicate the degree to which s/he considers it to have met the requirements. 0% to 30%

A further 20% is available for quality of design or quality of code or for demonstrating your client/server against a third party client/server. Factors to be considered will include clarity of the design, the degree to which the design matches the code, commenting, robustness and testing.