CS 345B Homework 1: Solutions

February 10, 2007

SCORING

Total points = 10.
Exercise 1 carries 3 points (1 point for part (a) and 2 points for part (b)).
Exercise 2 carries 2 points (0.5 point for each part).
Exercise 3 carries 5 points all together (roughly 1.5 points for parts (a) and (b), and 2 points for part (c)).
Exercise 4 carries 0 points. We wanted you to play around with the XML tools.

EXERCISE 1

The DTD does not state the type for attribute ID of element author. Full points have been given for any assumed type, or disregarding the type.

Part (a) The document is not well-formed. Two changes make it well-formed: (1) Closing the element LASTNAME for "Widom"; (2) Replacing "<NUM>1<NUM>" with "<NUM>1</NUM>", in the first NUM element.

Part (b) There are multiple ways of making the Data or DTD conform to the other. And you have been given points for any reasonable solution. One possible solution is:

(i) Add FIRSTNAME as a possible element inside LASTNAME; allow multiple TEXT elements within a CHAPTER element; make TEXT element optional for INTRODUCTION; allow multiple authors for a book; making the ID attribute of AUTHOR element optional (full points have been given even if this last change wasn't suggested because the type of ID wasn't in the DTD).

(ii) Remove or move outside the LASTNAME element, the FIRSTNAME element for "Hector"; only one author per book allowed; add an ID attribute for AUTHOR element if retained; add a TEXT element for the INTRODUCTION element; and, remove one TEXT element from the last CHAPTER;

EXERCISE 2

There are various possible answers, and most of you have got this question right. Some possible answers are:

Part (a) HTML need not be well-formed; XML allows creating new elements, but HTML does not; XML separates data from presentation; XML is designed to be machine readable but HTML is not.

Part (b) elements can have multiple sub-elements within them as well as attribute within them; elements can be repeated, attributes can't; elements can have complex types;

Part (c) Namespaces allow you to define scopes for element names. Suppose you wish to use the same element name, but in different contexts to mean two different things, you can qualify them with their respective namespaces.

Part (d) DTDs are simpler and more easily understandable than XMLSchema. On the other hand, XMLSchemas are more powerful, for instance, you can have complex types.

EXERCISE 3

This was an open-ended question to get you thinking about trying to design data for real applications, and all of you have done well in this question. For pqrt (c) we expected improvements such as using complex types, date types, etc, restricting the domains, such as ages, enumeration types, etc.

All of you have come up with interesting (different) designs! So we have given full points to all of you for this question.