Homework 3: Term Project and First-Order Calculi

Due: Problems 1-3, April 6, 2000; problem 4, April 18, 2000.

In this homework, you will get a start on your term project and you will explore the first-order type systems for Abadi and Cardelli's ς-calculus.

Read Chapters 6, 7, and 8 of *A Theory of Objects*, and possibly the chapters of Castagna's book that we passed out.

1. For the first part of your term project, you will explore the dynamic semantics of some object-oriented calculus other than Abadi and Cardelli's ς-calculus. We have discussed some variations and other calculi that might be suitable in class, and I have indicated to some of you in person some other variations. These include:

   - a calculus with explicit classes, perhaps something more directly like Smalltalk,
   - a calculus that uses delegation instead of embedding,
   - calculi with generic functions (as in Castagna's $\lambda\&$-calculus or BeCecil),
   - a calculus with parallel processing or distribution features,
   - a calculus that captures the essence of some component-based framework, such as JavaBeans or Microsoft COM,
   - a calculus of events used for implicit invocation.

   You can, of course, do something else if you wish. Please talk to me about your choice of calculus well before the due date.

   What you'll do for this problem at the moment is to explore the dynamic semantics of your chosen calculus. That is, you should give a treatment similar to the one that Abadi and Cardelli give in chapter 6 of their book. However, keep in mind that eventually you'll also be doing a type system for your language. I suggest that to avoid complications, at the moment to avoid imperative features as well. That is, you can use "update", which returns a new object based on an existing one, but I recommend that you did not mutate the state of an existing object.

   Hand in your dynamic semantics, and include several examples of small programs that you can write in it. You can do most of this on paper, however I suggest that you write some version of the dynamic semantics in lambda Prolog. This will help you be sure that your example programs and semantics all work together. I suggest that you work on those before going on to other phases of dynamic semantics.

2. In system $\mathbf{Ob}_1$, prove that if E is well-formed, $E \vdash A$, $E \vdash B$, and $x \notin dom(E)$, then $E, x : B \vdash A$.

3. (Generalized subject reduction)

   (a) State carefully the variation of the operational semantics for $\mathbf{Ob}_1$, discussed in class, that uses (value) environments instead of substitution.

   (b) We will consider type environments to be the types of value environments. For example, the environment $\rho_z = \{z \mapsto []\}$ has the type of the type environment $E_z = \{z : []\}$. Give a formal definition of this notion.

   (c) Prove the appropriate generalization of the subject reduction theorem (theorem 7.5-3). You will be assuming that you are given a value environment of a given type.

4. Give a first-order type system with subtyping for your language.

(a) Describe the type checking rules for your language. (You do not have to implement this in lambda Prolog, but you may do so if you wish.)

(b) Give some examples of terms in your language and derivations of their types.

(c) Prove the subject reduction theorem for your language.