LePUS2: Updates and Challenges

24-Feb-04

Amnon H. Eden
Department of Computer Science, University of Essex
eden@acm.org

Abstract. This document consists of a brief summary of the updates to the original version of LEPUS, presented in "Formal Specification of Object-Oriented Design" [Eden 01]. A summary of LEPUS 2, the most recent version of the language is available from the Web site [Eden 04].

1. Notation

Unless otherwise noted, I use the conventions established in FOSI [Turner 04] and the notational conventions established in [Eden 2004]. A summary of the symbols appears at the Appendix to this document.

This document contains hyperlinks and is recommended for best viewing in its digital form (`.pdf` or `.doc`).

2. Predicates

The predicates $Total$ (Definition) and $Isomorphic$ (Definition) have remained unchanged. Their definition for any binary relation $\mathcal{R}$ depends on the dimension of the arguments as follows:

$m, n : \mathbb{N}$

$X : \mathbf{P}^m(\mathbb{X})$

That is, X is a variable of type $\mathbb{X}$ and dimension m

$Y : \mathbf{P}^n(\mathbb{Y})$

$\mathcal{R} \subset \mathbb{X} \times \mathbb{Y}$

$Total(\mathcal{R}, X, Y) \triangleq$

$m = n = 0$	$\mathcal{R}(X, Y)$
$m > n = 0$	$\forall x \in X \bullet Total(\mathcal{R}, x, Y)$
$n > m = 0$	$\exists! y \in Y \bullet Total(\mathcal{R}, X, y)$
$m, n > 0$	$\forall x \in X \; \exists! y \in Y \bullet Total(\mathcal{R}, x, y)$

The writing $\mathcal{R}^{\rightarrow}(X, Y)$ as short for $Total(\mathcal{R}, X, Y)$.

Similarly:

$m, n : \mathbb{N}$

$X : \mathbf{P}^m(\mathbb{X})$

$Y : \mathbf{P}^n(\mathbb{Y})$

$\mathcal{R} \subset \mathbb{X} \times \mathbb{Y}$

$Isomorphic(\mathcal{R}, X, Y) \triangleq$

$m, n > 0$	$\forall x \in X \; \exists! y \in Y \bullet Isomorphic(\mathcal{R}, x, y) \wedge$ $\forall y \in Y \; \exists! x \in X \bullet Isomorphic(\mathcal{R}, x, y)$
Otherwise	$Total(\mathcal{R}, X, Y)$

The writing $\mathcal{R}^{\leftrightarrow}(X, Y)$ as short for $Isomorphic(\mathcal{R}, X, Y)$.

3. Hierarchies

3.1 2001 version

The first version of the language defines a hierarchy as a predicate for a set of classes h of any order. Given the predicate $Total(\mathcal{R}, S, T)$, the $Hierarchy$ predicate (Definition) can be defined as follows:

$k : \mathbb{N}$

$h : \mathbf{P}^k(\mathbb{C})$

That is, h is a class of any dimension

$Hierarchy(h) \triangleq$

$\exists r \in h \bullet Abstract(r) \wedge \forall c \in h \bullet Inherit^{*\rightarrow}(c, r)$

where $\mathcal{R}^*$ is the transitive closure of binary relation $\mathcal{R}$.

Thus, specifications in the first version used the predicate explicitly. For example, the specification of the Abstract Factory design pattern (see [Gamma et. al 95]) e.g.,

$products : \mathbf{P}(\mathbb{C})$ **AF1.1**

$Factories : \mathbf{P}^2(\mathbb{C})$

$\ldots$

$Hierarchy(factories)$

$Hierarchy(products)$

$\ldots$

The complete definition of the pattern is also available online (Abstract Factory).

3.2 2004 version

In the second version (as sketched in the Web site), we've added $\mathbf{P}^k(\mathbb{H})$ as a possible types. For example, AF1.1 can be rephrased as follows:

$products : \mathbb{H}$ **AF2.1**

$Factories : \mathbf{P}(\mathbb{H})$

$\ldots$

$\ldots$

4. Axioms

4.1 2001 version

The original set of axioms included the relation $SameSignature$, indicating that the relation was an equivalence relation.

4.2 2004 version

In the second version, signatures were added to the set of ground entities and $\mathbb{S}$ designated their collection. Thus, the axioms were modified and can be expressed formally as follows (Definitions).

The first axiom states that no two methods with the same signature are members of the same class.

$$c : \mathbb{C},\ s : \mathbb{S},\ f_1, f_2 : \mathbb{F}$$ **OOD1**

$$SignatureOf(f_1, s) \wedge SignatureOf(f_2, s) \wedge$$
$$\quad Member(f_1, c) \wedge Member(f_2, c) \Rightarrow f_1 = f_2$$

The second axiom states that the transitive closure of the binary relation *Inherit* induces a partial ordering on $\mathbb{C}$.

$$c, c_1, c_2 : \mathbb{C}$$ **OOD2**

$$\neg\ Inherit^*(c, c)$$
$$\neg\ (Inherit^*(c_1, c_2) \wedge Inherit^*(c_2 c_1)$$
$$Inherit^*(c_1, c) \wedge Inherit^*(c, c_2) \Rightarrow Inherit^*(c_1, c_2)$$

The last axiom states that the relation $SignatureOf \subset \mathbb{F} \times \mathbb{S}$ is a total and onto functional relation.

$$\forall f \in \mathbb{F}\ \exists!\, s \in \mathbb{S} \bullet SignatureOf(f, s)$$ **OOD3**
$$\forall s \in \mathbb{S}\ \exists f \in \mathbb{F} \bullet SignatureOf(f, s)$$

Let us define the relation $SameSignature$ for ground methods as follows:

$$f_1, f_2 : \mathbb{F}$$

$$SameSignature(f_1, f_2) \triangleq$$
$$\exists\ s \in \mathbb{S} \bullet SignatureOf(f_1, s) \wedge SignatureOf(f_2, s)$$

From the three OOD axioms, we should be able to prove that $SameSignature$ is an equivalence relation, and that the quotient set $\mathbb{F}/SameSignature$ partitions the methods in $\mathbb{F}$ by their signatures.

5. Clans and Tribes

5.1 2001 version

The original version did not include signatures as first-class entities and properties of sets of methods were expressed explicitly using the predicates $Clan$ and $Tribe$ (Definitions), defined for methods of any dimension.

From OOD3 we conclude that the functional notation $Sig(f)$ is defined for any ground method f. Using this notation,

$$k : \mathbb{N},\ k > 0$$
$$f : \mathbf{P}^k(\mathbb{F})$$
$$c : \mathbf{P}^k(\mathbb{C})$$

$$Clan(f, c) \triangleq$$
$$Member^{\leftrightarrow}(f, c) \wedge$$
$$\forall f_1, f_2 \in f \bullet Sig(f_1) = Sig(f_2)$$

Note that f and c are expected to be of the same dimension.

Similarly:

$k : \mathbb{N}, k > 0$
$f : \mathbf{P}^{k+1}(\mathbb{F})$
$c : \mathbf{P}^{k}(\mathbb{C})$

$Tribe(f,c) \triangleq$
$\quad \forall f' \in f \bullet Clan(f',c)$

Specifications made use of the predicates to define properties of sets of methods. In the example of the Abstract Factory pattern,

$Factories : \mathbf{P}(\mathbb{C})$ **AF1.2**
$Products : \mathbf{P}^{2}(\mathbb{C})$
$Factory\text{-}Methods : \mathbf{P}^{2}(\mathbb{F})$
$\ldots$

$Tribe(Factory\text{-}Methods, Factories)$
$Return^{\leftrightarrow}(Factory\text{-}Methods, Products)$
…

5.2 2004 version

The selection operator was introduced into the second version, defined as an operator on signatures and classes of any dimension and to return a method whose dimension is the sum of the dimension of the arguments. Thus, the function

$$\otimes : \mathbf{P}^{m}(\mathbb{C}) \times \mathbf{P}^{n}(\mathbb{F}) \rightarrow \mathbf{P}^{m+n}(\mathbb{F})$$

is defined for any natural numbers m, n as follows:

$m, n : \mathbb{N}$

$S : \mathbf{P}^m(\mathbb{S})$

$C : \mathbf{P}^n(\mathbb{C})$

$S \otimes C \triangleq$

$m = n = 0$	$f : \mathbb{F} \bullet$ $SignatureOf(f, S) \wedge Member(f, C)\}$	By OOD1 there can be only one such method
$m > n = 0$	$F : \mathbf{P}^m(\mathbb{F}) \bullet$ $\forall f \in F\ \exists!\, s \in S \bullet\ f = s \otimes C$	
$n > m = 0$	$F : \mathbf{P}^n(\mathbb{F}) \bullet$ $\forall f \in F\ \exists!\, c \in C \bullet\ f = S \otimes c$	
$m, n > 0$	$F : \mathbf{P}^{m+n}(\mathbb{F}) \bullet$ $\forall f \in F\ \exists!\, s \in S\ \exists!\, c \in C \bullet\ f = s \otimes c$	

It should be easy to show that when $m = n$, $s \otimes c$ gives a $Clan$ in c and when $m = n + 1$, $s \otimes c$ is a $Tribe$ in c.

An equivalent definition is given in the Web page in its current version (Definition).

Hence, definitions in this new version made use of the selection operator instead of the predicates $Clan$ and $Tribe$, which made specifications more elegant. For example, AF1.2 could be replaced with the following:

$$Factories : \mathbf{P}(\mathbb{C})$$
$$Products : \mathbf{P}^2(\mathbb{C})$$
$$Factory\text{-}Methods : \mathbf{P}(\mathbb{S})$$
$$\ldots$$

AF2.2

$$Return^{\leftrightarrow}(Factory\text{-}Methods \otimes Factories, Products)$$
$$\ldots$$

Note that in AF2.2, the expression $Factory\text{-}Methods \otimes Factories$ replaces the variable $Factory\text{-}Methods$ from AF1.2, and that it can be shown to be a tribe in Factories.

6. LePUS symbols

This list is not comprehensive but merely a sample of symbols used. The complete character MathType character set is available for download from http://www.mathtype.com/support/fonts/.

6.1 Symbols in the metalanguage

$\mathfrak{M} = \langle \mathbb{U}, \mathbb{R} \rangle$	Design model
$\models$	Truth-table entailment
$\mathcal{R}\ \mathcal{R}^*\ \mathcal{R}^+$	Relations, transitive closure (zero/one or more)
$f :\ S \rightarrow \mathrm{T}$	Functions
$x\ y$	Variables

$\triangleq$	Definitions
$\forall$ $\exists!$ $\wedge$ $\vee$ $\Rightarrow$ $\neg$	Quantifiers & logical connectives
$\subset$ $\in$ $\times$	Set-related operations

6.2 LePUS symbols

c_1, c_2, `Objects` : $\mathbf{P}^2(\mathbb{C})$	Variable/constant declarations, domains
$s \otimes c$	Selection operator
$\mathcal{R}^{\rightarrow}$ $\mathcal{R}^{\leftrightarrow}$	Total/bijective relations

References

E. Gamma, R. Helm, R. Johnson, J. Vlissides (1995). *Design Patterns: Elements of Reusable Object Oriented Software*. Reading, MA: Addison-Wesley.

A. H. Eden. "LePUS 2: Summary". Web page (Jan. 2004), http://www.eden-study.org/lepus/definitions.

A. H. Eden. "Formal Specification of Object-Oriented Design." *International Conference on Multidisciplinary Design in Engineering CSME-MDE* (Nov. 21—22, 2001), Montreal, Canada.

R. Turner. "The Foundations of Specifications I." Technical report CSM-396 (Feb. 2004), Department of Computer Science, University of Essex.