Constructing Scripts Compositionally: A Molecular Biology Example
Working Note 18

Peter Clark
Knowledge Systems
Boeing, PO Box 3707, Seattle, WA 98124
peter.e.clark@boeing.com

Bruce Porter
Department of Computer Science
University of Texas at Austin, TX 78712
porter@cs.utexas.edu

March 2000

Abstract

This Working Note follows on from the previous Working Note 17 [Clark and Porter, 2000], to present another example of constructing scripts compositionally. This example is from molecular biology, describing how a virus passes its contents (the capsid) to a cell. The reader is referred to the previous working note for a discussion of general approach. Again, the document shows two variant formulations of the representation, one using KM classes, and one using KM "prototypes", followed by a machine-generated dump of the graphlets produced from the prototype specifications. The main purpose of this note is to show KM implementations of a more detailed example of script composition.

1 Introduction

This Working Note follows on from the previous Working Note [Clark and Porter, 2000], to present another example of constructing scripts compositionally. This example is from molecular biology, describing how a virus passes its contents (the capsid) to a cell. The reader is referred to the previous working note for a discussion of general approach. Again, the document shows two variant formulations of the representation, one using KM classes, and one using KM "prototypes", followed by a machine-generated dump of the graphlets produced from the prototype specifications. In this more complex example, the class-based representation is a bit more cumbersome (but still just as valid) as the prototype equivalent.

2 The Example

2.1 Representation

A virus passes its contents (the capsid) to a cell by attaching, piercing, and then fusing with that cell[1] This is illustrated in Figure 1. We can describe this sequence of events

[1]This is a very simplistic (and possibly erroneous) description. For more detail, see http://homeport.tcs.tulane.edu/∼dmsander/WWW/224/Replication224.html

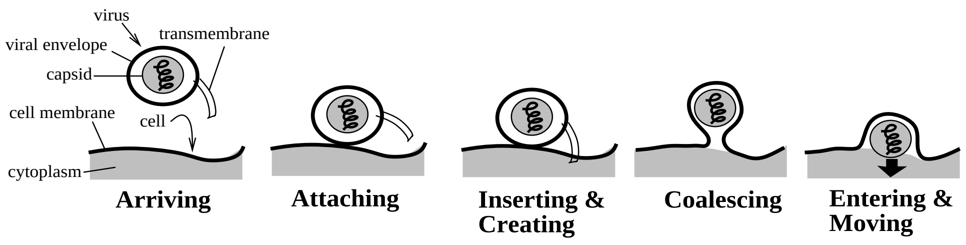

Figure 1: A highly simplified (and possibly erroneous) view of how a virus passes its contents to a cell.

compositionally using two abstractions: invading (the capsid invades the cell), and delivering (the virus delivers the capsid to the cell). Details of these abstractions then contribute to the description of the composite event:

`Virus-Visiting` = `Invading` + `Delivering`

`Invading` = `Arriving` + `Breaking` + `Entering`

`Delivering` = `Arriving` + `Moving`

The particular manner by which a virus 'breaks' into a cell is called 'fusing', where the membranes of the virus and cell merge together (a bit like two soap bubbles becoming one). Fusion is triggered by the insertion of the virus transmembrane through the cell membrane. Once the virus and cell membranes have fused, the virus capsid is inside the (new) cell boundary and can thus enter the cell. We describe this abstraction as follows:

`Fusing` = `Attaching` + `Piercing` + `Coalescing`

and Piercing itself can be described at two cotemporal events:

`Piercing` = `Inserting` (an instrument) + `Creating` (a hole)

Finally, we need to specify some additional constraints on the `Virus-Visiting`. First, for a virus, the breaking part of the invading is a special kind of breaking (namely a fusing). Second, the arriving's in the invading and delivering are coreferential (ie. the virus doesn't arrive twice). Third, the entering (at the end of the invading) and the moving (in the delivering) are really two viewpoints on the same event (the capsid entering the cell cytoplasm). We thus label them as cotemporal with each other. These constraints are part of the specification of `Virus-Visiting`, describing the special way the involved components combine.

2.2 The Performance Task

As for the restaurant visit script [Clark and Porter, 2000], the KB is designed to answer just a single question, namely "What are the subevents in <*some main event*>?". KM returns a list of events (KM instances), with their roles appropriately filled in, and with appropriate constraints on their ordering. In principle, if we added representations for these primitive events themselves (e.g., a STRIPS-like description of which facts are made true/false by the events), then KM could perform a simulation of the scenario by "executing" each action in turn. This would result in a sequence of KM situations in the knowledge-base, each describing the state of the world at a different point in the script. From here, additional questions could be answered, posed to situations in this sequence.

Both formulations (classes and prototypes) of the representation produce the following question-answering behavior by KM:

```
;;; “How does a virus pass its contents into a cell?”
KM> (the description-of-leaf-subevents of (a Virus-Visiting))
The arriving. The arriving is before the moving the fusing.
The attaching. The attaching is before the piercing the entering.
The creating. The creating is before the coalescing the entering.
              The creating is cotemporal with the inserting.
The inserting. The inserting is before the coalescing the entering.
               The inserting is cotemporal with the creating.
The coalescing. The coalescing is before the entering.
                The coalescing is after the piercing.
The moving. The moving is cotemporal with the entering.
            The moving is after the arriving.
The entering. The entering is cotemporal with the moving.
              The entering is after the fusing.
```

Note above that the creating/inserting are cotemporal, and the moving/entering are cotemporal. In addition, although not revealed by this print-out, each subevent contains information about who its players are (agent, patient, etc.). The knowledge-base could thus answer questions about these facts also.

We now show the knowledge bases themselves (using classes and prototypes), and finally the graphlets produced from the prototype specifications. This graphlet database may be useful due to its portability (e.g. easy translation to KIF), and as data for a graph manipulation interface such as WebMod.

3 KM Representation: Using Classes

```
;;; File: bioex-classes.km
;;; Date: March 2000

;;; This file requires KM 1.4.0-beta33 or later

(reset-kb)

;;; Declare some inverses...
(before has (instance-of (Slot)) (inverse (after)))
(cotemporal-with has (instance-of (Slot)) (inverse (cotemporal-with)))
(subevents has (instance-of (Slot)) (inverse (superevents)))

;;; [1] This ugly formatting simply prints out the before, cotemporal-with, and
;;; after properties for each leaf subevent of the main event.
(every Event has
  (before ((the before of (the superevents of Self))))
  (cotemporal-with ((the cotemporal-with of (the superevents of Self))))
  (all-subevents ((the subevents of Self)
                  (the all-subevents of (the subevents of Self))))
  (leaf-subevents ((allof (the all-subevents of Self)
                    where (not (the subevents of It)))))
  (subevents ((the subevents of (the component-events of Self))))
  (description-of-leaf-subevents (
      (make-sentence
             (forall (the leaf-subevents of Self)                      ; [1]
                (:seq It "."
                      (if   (has-value (the before of It))
                       then (:seq It "is before" (the before of It) "."))
                      (if   (has-value (the cotemporal-with of It))
                       then (:seq It "is cotemporal with" (the cotemporal-with of It) "."))
                      (if   (has-value (the after of It))
                       then (:seq It "is after" (the after of It) "."))
                      (format nil "~%")))))))

;;; Being lazy over the taxonomy here...
(Virus-Visiting has (superclasses (Event)))
(Invading has (superclasses (Event)))
(Delivering has (superclasses (Event)))
(Fusing has (superclasses (Breaking)))
(Coalescing has (superclasses (Event)))
(Piercing has (superclasses (Event)))
(Arriving has (superclasses (Event)))
(Breaking has (superclasses (Event)))
(Entering has (superclasses (Event)))
(Moving has (superclasses (Event)))
(Attaching has (superclasses (Event)))
(Creating has (superclasses (Event)))
(Inserting has (superclasses (Event)))

;;; ----------------------------------------
;;;       VIRUS VISITING (composition)
;;; ----------------------------------------

;;; [1] is a slightly cumbersome way of saying the two Arrivings (subevents of
;;;     the Invading and Delivering respectively) are coreferential.
;;; [2] We wish to say that the generic Breaking in the Invading is (here) a special
;;;     way of breaking into something, namely a Fusing. Here we rely on KM's
;;;     set unification mechanism to appropriately unify the Fusing (here) with the
;;;     Breaking (inherited from Invading).
(every Virus-Visiting has
  (agent ((a Virus with
            (container-wall ((a Viral-Envelope)))
            (contents ((a Capsid)))
            (attachments ((a Transmembrane))))))
  (patient ((a Cell with
             (container-wall ((a Cell-Membrane)))
             (contents ((a Cytoplasm))))))
  (component-events (
    (a Invading with
      (agent ((the Capsid contents of (the Virus agent of Self))))
      (patient ((the Cell patient of Self)))
      (barrier ((the container-wall of (the Cell patient of Self))))
      (subevents (
        (the Arriving subevents of (the Delivering component-events of Self))  ; [1]
        (a Fusing)                                                              ; [2]
        (a Entering with
          (cotemporal-with ((the Moving subevents of
                                (the Delivering component-events of Self))))))))
    (a Delivering with
      (agent ((the Virus agent of Self)))
      (package ((the Capsid contents of (the Virus agent of Self))))
      (recipient ((the Cell patient of Self)))
      (subevents (
        (the Arriving subevents of (the Invading component-events of Self))    ; [1]
        (a Moving with
          (cotemporal-with ((the Entering subevents of
                                (the Invading component-events of Self))))))))))

;;; ----------------------------------------
;;;       INVADING
;;; ----------------------------------------

(every Invading has
  (agent ((a Thing)))
  (patient ((a Thing)))
  (barrier ((a Thing with
             (surrounds ((the patient of Self))))))
  (subevents (
    (a Arriving with
      (agent ((the agent of Self)))
      (location ((the patient of Self)))
      (before ((the Breaking subevents of Self))))
    (a Breaking with
      (agent ((the agent of Self)))
      (patient ((the barrier of Self)))
      (before ((the Entering subevents of Self))))
    (a Entering with
      (agent ((the agent of Self)))
      (patient ((the patient of Self)))))))

;;; ----------------------------------------
;;;       DELIVERING
;;; ----------------------------------------

(every Delivering has
  (agent ((a Thing)))
  (package ((a Thing)))
  (recipient ((a Thing)))
  (subevents (
    (a Arriving with
      (agent ((the agent of Self)))
      (destination ((the recipient of Self)))
      (before ((the Moving subevents of Self))))
    (a Moving with
      (agent ((the agent of Self)))
      (patient ((the package of Self)))
      (destination ((the recipient of Self)))))))

;;; ----------------------------------------
;;;       FUSING
;;; ----------------------------------------

(every Fusing has
  (agent ((a Thing)))
  (patient ((a Thing)))         ; the barrier
  (subevents (
    (a Attaching with
      (agent ((the agent of Self)))
      (patient ((the patient of Self)))
      (before ((the Piercing subevents of Self))))
    (a Piercing with
      (agent ((the agent of Self)))
      (patient ((the container-wall of (the patient of Self))))
      (before ((the Coalescing subevents of Self))))
    (a Coalescing with
      (agent ((the agent of Self)))
      (patients ((the agent of Self) (the patient of Self)))))))

;;; ----------------------------------------
;;;       PIERCING
;;; ----------------------------------------

(every Piercing has
  (agent ((a Thing)))
  (patient ((a Thing)))
  (instrument ((a Thing)))                       ; a pointy thing
  (subevents (
    (a Creating with
      (agent ((the agent of Self)))
      (created ((a Portal with
                  (part-of ((the patient of Self))))))
      (instrument ((the instrument of Self)))
      (cotemporal-with ((the Inserting subevents of Self))))
    (a Inserting with
      (agent ((the agent of Self)))
      (instrument ((the instrument of Self)))
      (patient ((the patient of Self)))))))

;;; --- end ---
```

4 KM Represesentation: Using Prototypes

```
;;; File: bioex-prototypes.km
;;; Date: March 2000

;;; This file requires KM 1.4.0-beta33 or later

(reset-kb)

;;; Declare some inverses...
(before has (instance-of (Slot)) (inverse (after)))
(cotemporal-with has (instance-of (Slot)) (inverse (cotemporal-with)))
(subevents has (instance-of (Slot)) (inverse (superevents)))

;;; [1] This ugly formatting simply prints out the before, cotemporal-with, and
;;; after properties for each leaf subevent of the main event.
(every Event has
  (before ((the before of (the superevents of Self))))
  (cotemporal-with ((the cotemporal-with of (the superevents of Self))))
  (all-subevents ((the subevents of Self)
                  (the all-subevents of (the subevents of Self))))
  (leaf-subevents ((allof (the all-subevents of Self)
                    where (not (the subevents of It)))))
  (subevents ((the subevents of (the component-events of Self))))
  (description-of-leaf-subevents (
      (make-sentence
             (forall (the leaf-subevents of Self)                       ; [1]
                (:seq It "."
                      (if   (has-value (the before of It))
                       then (:seq It "is before" (the before of It) "."))
                      (if   (has-value (the cotemporal-with of It))
                       then (:seq It "is cotemporal with" (the cotemporal-with of It) "."))
                      (if   (has-value (the after of It))
                       then (:seq It "is after" (the after of It) "."))
                      (format nil "~%")))))))

;;; Being lazy over the taxonomy here...
(Virus-Visiting has (superclasses (Event)))
(Invading has (superclasses (Event)))
(Delivering has (superclasses (Event)))
(Fusing has (superclasses (Breaking)))
(Coalescing has (superclasses (Event)))
(Piercing has (superclasses (Event)))
(Arriving has (superclasses (Event)))
(Breaking has (superclasses (Event)))
(Entering has (superclasses (Event)))
(Moving has (superclasses (Event)))
(Attaching has (superclasses (Event)))
(Creating has (superclasses (Event)))
(Inserting has (superclasses (Event)))

;;; ----------------------------------------
;;;       VIRUS VISITING (composition)
;;; ----------------------------------------

(a-prototype Virus-Visiting)

;;; Introduce the objects into the prototype:
(a Virus with
  (container-wall ((a Viral-Envelope)))
  (contents ((a Capsid)))
  (attachments ((a Transmembrane))))

(a Cell with
  (container-wall ((a Cell-Membrane)))
  (contents ((a Cytoplasm))))

((the Virus-Visiting) has
  (agent ((the Virus)))
  (patient ((the Cell)))
  (component-events (
    (a Invading with
      (agent ((the Capsid)))
      (patient ((the Cell)))
      (barrier ((the Cell-Membrane))))
    (a Delivering with
      (agent ((the Virus)))
      (package ((the Capsid)))
      (recipient ((the Cell)))))))

((the Invading) has (subevents ((a Arriving) (a Breaking) (a Entering))))
((the Delivering) has (subevents ((a Arriving) (a Moving))))

;;; The Arrivings are coreferential
((the Arriving subevents of (the Invading)) ==
                (the Arriving subevents of (the Delivering)))

((the Entering) has
   (cotemporal-with ((the Moving))))

;;; In this case, the breaking occurs via fusing
((the Breaking) == (a Fusing))

(end-prototype)

;;; ----------------------------------------
;;;       INVADING
;;; ----------------------------------------

(a-prototype Invading)

((the Invading) has
  (agent ((a Thing)))
  (patient ((a Thing)))
  (barrier ((a Thing with
              (surrounds ((the patient of Self))))))
  (subevents (
    (a Arriving with
      (agent ((the agent of Self)))
      (location ((the patient of Self)))
      (before ((the Breaking subevents of Self))))
    (a Breaking with
      (agent ((the agent of Self)))
      (patient ((the barrier of Self)))
      (before ((the Entering subevents of Self))))
    (a Entering with
      (agent ((the agent of Self)))
      (patient ((the patient of Self)))))))
```

```
(end-prototype)

;;; ----------------------------------------
;;;       DELIVERING
;;; ----------------------------------------

(a-prototype Delivering)

((the Delivering) has
  (agent ((a Thing)))
  (package ((a Thing)))
  (recipient ((a Thing)))
  (subevents (
    (a Arriving with
      (agent ((the agent of Self)))
      (destination ((the recipient of Self)))
      (before ((the Moving subevents of Self))))
    (a Moving with
      (agent ((the agent of Self)))
      (patient ((the package of Self)))
      (destination ((the recipient of Self)))))))

(end-prototype)

;;; ----------------------------------------
;;;       FUSING
;;; ----------------------------------------

(a-prototype Fusing)

((the Fusing) has
  (agent ((a Thing)))
  (patient ((a Thing)))         ; the barrier
  (subevents (
    (a Attaching with
      (agent ((the agent of Self)))
      (patient ((the patient of Self)))
      (before ((the Piercing subevents of Self))))
    (a Piercing with
      (agent ((the agent of Self)))
      (patient ((the container-wall of (the patient of Self))))
      (before ((the Coalescing subevents of Self))))
    (a Coalescing with
      (agent ((the agent of Self)))
      (patients ((the agent of Self) (the patient of Self)))))))

(end-prototype)

;;; ----------------------------------------
;;;       PIERCING
;;; ----------------------------------------

(a-prototype Piercing)

((the Piercing) has
  (agent ((a Thing)))
  (patient ((a Thing)))
  (instrument ((a Thing)))                    ; a pointy thing
  (subevents (
    (a Creating with
      (agent ((the agent of Self)))
      (created ((a Portal with
                  (part-of ((the patient of Self))))))
      (instrument ((the instrument of Self)))
      (cotemporal-with ((the Inserting subevents of Self))))
    (a Inserting with
      (agent ((the agent of Self)))
      (instrument ((the instrument of Self)))
      (patient ((the patient of Self)))))))

(end-prototype)

;;; --- end ---
```

5 Graphlets for the KM Prototypes

The following shows a (automatically generated) dump of the graphlets for this example, i.e., KM's internal representation of the prototypes specified in Section 4. These graphlets are constructed by KM at load-time, when it loads the prototype descriptions in Section 4. This print-out was generated by a small Lisp procedure, separate from the KM implementation. (It is shown in Prolog-like syntax to suggest similarity in style with the WordNet database). Graphlets have well-defined semantics: If graphlet G of concept C_0 contains instances $I_0, I_1, ..., I_n$ of classes $C_0, C_1, ..., C_n$, and relations $R = \{r(I_i, I_j)\}$, then for all instances I'_0 of C_0, there exists instances $I'_1, ..., I'_n$ of classes $C_1, ..., C_n$ such that relations $R' = \{r(I'_i, I'_j)\}$ hold, where R' is a copy of R with the instances $\{I_n\}$ replaced with their corresponding instances $\{I'_n\}$.

```
graphlet(0,'Virus-Visiting').

participants(0,0).
participants(0,1).
participants(0,2).
participants(0,3).
participants(0,4).
participants(0,5).
participants(0,6).
participants(0,7).
participants(0,8).
participants(0,9).
participants(0,10).
participants(0,11).
participants(0,12).
participants(0,13).

isa(0,'Virus-Visiting').
isa(1,'Fusing').
isa(2,'Moving').
isa(3,'Arriving').
isa(4,'Entering').
isa(5,'Cytoplasm').
isa(6,'Cell-Membrane').
isa(7,'Transmembrane').
isa(8,'Capsid').
isa(9,'Viral-Envelope').
isa(10,'Delivering').
isa(11,'Invading').
isa(12,'Cell').
isa(13,'Virus').

agent(0,13).
patient(0,12).
'component-events'(0,11).
'component-events'(0,10).
superevents(1,11).
superevents(2,10).
'cotemporal-with'(2,4).
superevents(3,11).
superevents(3,10).
superevents(4,11).
'cotemporal-with'(4,2).
```

```
'contents-of'(5,12).
'container-wall-of'(6,12).
'barrier-of'(6,11).
'attachments-of'(7,13).
'contents-of'(8,13).
'agent-of'(8,11).
'package-of'(8,10).
'container-wall-of'(9,13).
agent(10,13).
package(10,8).
recipient(10,12).
'component-events-of'(10,0).
subevents(10,3).
subevents(10,2).
agent(11,8).
patient(11,12).
barrier(11,6).
'component-events-of'(11,0).
subevents(11,3).
subevents(11,1).
subevents(11,4).
'container-wall'(12,6).
contents(12,5).
'patient-of'(12,11).
'patient-of'(12,0).
'recipient-of'(12,10).
'container-wall'(13,9).
contents(13,8).
attachments(13,7).
'agent-of'(13,10).
'agent-of'(13,0).

% ----------

graphlet(14,'Delivering').

participants(14,14).
participants(14,15).
participants(14,16).
participants(14,17).
participants(14,18).
participants(14,19).

isa(14,'Delivering').
isa(15,'Moving').
isa(16,'Arriving').
isa(17,'Thing').
isa(18,'Thing').
isa(19,'Thing').

agent(14,19).
package(14,18).
recipient(14,17).
subevents(14,16).
subevents(14,15).
agent(15,19).
patient(15,18).
destination(15,17).
superevents(15,14).
```

```
after(15,16).
agent(16,19).
destination(16,17).
before(16,15).
superevents(16,14).
'recipient-of'(17,14).
'destination-of'(17,15).
'destination-of'(17,16).
'package-of'(18,14).
'patient-of'(18,15).
'agent-of'(19,15).
'agent-of'(19,16).
'agent-of'(19,14).

% ----------

graphlet(20,'Fusing').

participants(20,20).
participants(20,21).
participants(20,22).
participants(20,23).
participants(20,24).
participants(20,25).

isa(20,'Fusing').
isa(21,'Coalescing').
isa(22,'Piercing').
isa(23,'Attaching').
isa(24,'Thing').
isa(25,'Thing').

agent(20,25).
patient(20,24).
subevents(20,23).
subevents(20,22).
subevents(20,21).
agent(21,25).
patients(21,25).
patients(21,24).
superevents(21,20).
after(21,22).
agent(22,25).
before(22,21).
superevents(22,20).
after(22,23).
agent(23,25).
patient(23,24).
before(23,22).
superevents(23,20).
'patient-of'(24,23).
'patient-of'(24,20).
'patients-of'(24,21).
'agent-of'(25,21).
'agent-of'(25,22).
'agent-of'(25,23).
'agent-of'(25,20).
'patients-of'(25,21).
```

```
% ----------

graphlet(26,'Invading').

participants(26,26).
participants(26,27).
participants(26,28).
participants(26,29).
participants(26,30).
participants(26,31).
participants(26,32).

isa(26,'Invading').
isa(27,'Entering').
isa(28,'Breaking').
isa(29,'Arriving').
isa(30,'Thing').
isa(31,'Thing').
isa(32,'Thing').

agent(26,32).
patient(26,31).
barrier(26,30).
subevents(26,29).
subevents(26,28).
subevents(26,27).
agent(27,32).
patient(27,31).
superevents(27,26).
after(27,28).
agent(28,32).
patient(28,30).
before(28,27).
superevents(28,26).
after(28,29).
agent(29,32).
location(29,31).
before(29,28).
superevents(29,26).
surrounds(30,31).
'barrier-of'(30,26).
'patient-of'(30,28).
'patient-of'(31,27).
'patient-of'(31,26).
'surrounds-of'(31,30).
'location-of'(31,29).
'agent-of'(32,27).
'agent-of'(32,28).
'agent-of'(32,29).
'agent-of'(32,26).

% ----------

graphlet(33,'Piercing').

participants(33,33).
participants(33,34).
participants(33,35).
participants(33,36).
participants(33,37).
participants(33,38).
participants(33,39).

isa(33,'Piercing').
isa(34,'Portal').
isa(35,'Inserting').
isa(36,'Creating').
isa(37,'Thing').
isa(38,'Thing').
isa(39,'Thing').

agent(33,39).
patient(33,38).
instrument(33,37).
subevents(33,36).
subevents(33,35).
'part-of'(34,38).
'created-of'(34,36).
agent(35,39).
instrument(35,37).
patient(35,38).
superevents(35,33).
'cotemporal-with'(35,36).
agent(36,39).
created(36,34).
instrument(36,37).
'cotemporal-with'(36,35).
superevents(36,33).
'instrument-of'(37,35).
'instrument-of'(37,36).
'instrument-of'(37,33).
'patient-of'(38,35).
'patient-of'(38,33).
part(38,34).
'agent-of'(39,35).
'agent-of'(39,36).
'agent-of'(39,33).
```

References

[Clark and Porter, 2000] Clark, P. and Porter, B. (2000). Working note 17: $RESTAURANT re^n-visited: A KM implementation of a compositional approach. (http://www.cs.utexas.edu/users/pclark/working_notes).