EasyChair Preprint

№ 4737

A Formal Method on Operator Composites on Maiar DTD Game World.

Frank Appiah

December 11, 2020

A Formal Method on Operator Composites on Maiar DTD Game World.

FRANK APPIAH.

KING'S COLLEGE LONDON, SCHOOL OF ENGINEERING, LONDON, ENGLAND, UNITED KINGDOM.

appiahnsiahfrank@gmail.com. | frank.appiah@kcl.ac.uk

Extended Abstract[+]. This thesis is based on formal theory of grammar analysis on composite operators on Maiar game world is in DTD. It describes how operator compositions are used to define the document types and element types.

Keywords. document types, formal theory, grammar, composites, operator .

Year of Study: 2017 **Year of Publication: 2020**

1 ***AFFILIATE. UNIVERSITY OF LONDON, KING'S COLLEGE LONDON, DEPARTMENT OF INFORMATICS, LONDON, UK.**

1 INTRODUCTION

In this grammar analysis, I will formalize composite operations on a game world[1] defined from a document type definitions[2]. A schema defines a document type or class of documents by imposing a set of constraints on document instances. This thesis postulates that the operator compositions in document of type, *map* introduces the whole

content of the game world. The unified formalism for both document type definition and element type definition is provided. It is achieved by new composites operation into regular expressions. This operation replaces the powerset operator and makes regular expressions equivalent to regular grammar. The basic composites of a regular expression are as follows: *Variable*: It denotes an arbitrary expression. Y is a variable. *Tagged Expression*: It is also denoted as m[Y]. *Empty Sequence*: It is denoted as (). A composition of the three composite elements need three simple operators: *Concatenation*: X, Y denotes the concatenation of the expressions X and Y. *Union*: X | Y denotes that a pattern can consist of either X or Y. and *Substitution*: is an expression enclosed by curly brackets and decorated label (l). The other operations are option, powerset, interleaving and non-empty powerset.

Option (X?) is about the denotation of X or an empty sequence:

```
X? == X | ().
```

Powers et(X*) denotation is about a set of recurring concatenation of X:

```
X* == l{(X, l)?
```

Interleaving: `X&Y == X, Y | Y, X.`

Non-empty powerset; Recurrence at least once:

```
X+== X, X*.
```

The basic type algebra is about distributive properties of concatenation and tagging in respect to the union.

2 OPERATOR COMPOSITIONS

The operator composite notations of a game world as represented as from a Document Type Definition schema is as shown below:

2.1 Regular Map Operations

The basic notations of regular map composites are:

- Variables : `exit, global, rooms, entry`.
- **`Operators:`** `exit=(). global=(). rooms=(). entry=().`

2.2 Regular Rooms Operation

The basic notations of regular rooms composition are:

- Variable: `room`
- **`Operator:`** `room*==l{(room, l)?`

2.3 Regular Room Operation

The basic notations of regular room compositions are:

- Variables: `id, name, commands, directions, items, look, intro`
- **Operators:** `commands*==l{(command, l)? id=(). name=(). directions*==l{(directions, l)?. items*==l{(items, l)? look*==l{(look, l)?. intro*==l{(intro, l)?.`

2.4 Regular Intro Operation

The basic notations of regular intro composition are:

- Variable: `message`
- **Operator**: `message=().`

2.5 Regular Look Operation

The basic notations of regular Look composition are:

- Variable: `description`
- **Operator:** `description=().`

2.6 Regular Items Operation

The basic notations of regular items composites are:

- Variable: `item`
- **Operator:** `item*=l{(item, l)?.`

2.7 Regular Item Operation

The basic notations of regular item composition are:

- Variable: `name`
- **Operator:** `name=().`

2.8 Regular Directions Operation

The basic notations of regular directions composition are:

- Variable: `north, south, east, west`

- **Operator**: `north=(). south=(). east=(). west=()`.

2.9 Regular North Operation

The basic notations of regular north composition are:

- Variable: `roomId`
- **Operator**: `roomId=().`

2.10 Regular South Operation

The basic notations of regular south composite are:

- Variable: `roomId`
- **Operator** : `roomId=().`

2.11 Regular East Operation

The basic notations of regular east composition are:

- Variable: `roomId`
- **Operator:** `roomId=().`

2.12 Regular West Operation

The basic notations of regular west composition are:

- Variable: `roomId`
- `Operator: roomId=().`

2.13 Regular Commands Operation

The basic notations of regular commands composition are:

- Variable: `command`
- **Operator**: `command*=l{(command, l)?.`

2.14 Regular Command Operation

The basic notations of regular command composition are:

- Variable: `name, action`
- **Operator**: `name=(). action=l{(action, l)?.`

2.15 Regular Action Operation

The basic notations of regular action composition are:

- Variable: `requirement, effect`
- **Operators**: `requirement=l{(requirement, l)?.`
  `effect=l{(effect, l)?`

2.16 Regular Effect Operation

The basic notations of regular effect composition are:

- Variable: `operator, value, parameter`
- **Operators**: `operator=(). value=().`
  `parameter=().`

2.17 Regular Requirement Operation

The basic notations of regular requirement composition are:

- Variables: `notSatisfied, satisfied, value,`
  `parameter`

- **Operators:**
  `value=(). parameter=().`

  `notSatisfied*=l{(notSatisfied, l)?.`

  `satisfied=l{(satisfied, l)?.`

2.18 Regular notSatisfied Operation

The basic notations of regular notSatisfied composition are:

- Variable: `action`
- **Operator**: `action*=l{(action, l)?.`

2.19 Regular satisfied Operation

The basic notations of regular satisfied composition are:

- Variable: `action`
- **Operator**: `action*=l{(action, l)?.`

2.20 Regular Global Operations

The basic notations of regular global composition are:

- Variable: `command`
- **Operator**: `command*=l{(command,l)?.`

2.21 Regular Exit Operation

The basic notations of regular exit composition are:

- Variable: `id, room`
- **Operator**: `id=().  room=().`

4 CONCLUSION

This section concludes work on grammar analysis on document type definition with operator compositions. In all, 21 operator compositions were analyzed to generate operator composite for each element of map document content element. This forms part of one of the formal thesis on world composites of Maiar game DTD based on operator composition notation.

Compliance with Ethical Standards:

(In case of funding) Funding: This is research is funded by King's Alumni Group, School of Engineering with postgraduate grant with ISAreference grant number: 204424 20821845.

Conflict of Interest:

Author, Dr. Frank Appiah declares that he has no conflict of interest .

REFERENCES

1. Appiah F. (2012/13), Design and Implementation of a Graph-based Game, KCL School of Engineering, PhD Dissertation,
2. Berthold B.(2003). Modeling Business Objects With XML Schema. Morgan Kauffman, Elsevier Publishers.