
Modeling-Notation Source:

OPM/MAS

Version 03-03-12 17:25EST

Document authors: Marc-Philippe Huget, Iris Reinhartz-Berger, Dov Dori, Onn Shehory, Arnon Sturm

1 Description

In this document, we will consider the OPM notation and particularly, the OPM/MAS notation that is an extension of OPM for multiagent systems. OPM (Object-Process Methodology) [Dori, 2002] unifies in a single model both the static-structural and the dynamic-procedural aspects. Using a single graphical model and a corresponding textual specification, OPM unifies the system's function, structure and behaviour within one frame of reference that is expressed both diagrammatically and by a subset of English called Object-Process Language (OPL). As a consequence, OPM is distinguished from UML since it does not impose designers to represent several diagrams where one diagram can depict objects, object interaction or deployment.

OPM inherits its capabilities from both object- and process-oriented paradigms. OPM is an integrated approach to the study and development of software systems. In OPM, objects and processes have equal status and are described as things or entities. OPM handles complex systems by using recursive seamless scaling. In OPM, objects are viewed as persistent entities interacting with each other through processes—transient entities that affect objects by changing their state. Object-Process Diagrams (OPDs) enable us to describe things (objects and processes) and how they interact with each other. Things can be simple or compound. A compound thing is a thing, which is a generalization of other things, or an aggregation of other things, or is characterized by other things.

In OPM, an object class can be either systemic (internal) or environmental (external to the system). Orthogonally, it can be either physical or informatical (logical). An object class can be at one of several states, which are possible internal status values of the class objects. At any point in time, each object is at some state, and objects are transformed (generated, consumed, or change their state) through the occurrence of a process. A process can affect (change the state of) an environmental or a physical device, or the state or value of a specific attribute of an object class (rather than the state of the entire class, as in object-oriented methods).

OPM/MAS [Sturm, 2003] is an extension of OPM for multiagent systems. OPM/MAS follows principles of the Meta Object Facility (MOF) [MOF, 2002], concept which enables its flexibility. Sturm defines the OPM/MAS as a four layers architecture: (1) the meta-metamodel, which is the OPM itself; (2) the metamodel, which describes the specific building blocks within the MAS domain using OPM; (3) the model, which is based on the metamodel; and (4) the application, which is an implementation of the model.

Additionally, Sturm defines a set of building blocks that are commonly used in specifying MAS. These building blocks are split into two groups: (1) static, declarative building blocks including organization, society, platform, rule, role, user, protocol, belief, desire, fact, goal, intention, and service. These building blocks are defined as OPM objects; (2) building blocks with behavioural and dynamic nature such as agent, task and message. These building blocks are defined as process in OPM.

43 **2 Notation**

44 In this section, we elaborate the notations within OPM, which are used as the infrastructure of OPM/MAS.
 45 The notations are divided into several groups: Entities, Structural Links, Procedural Links and Event Links. In the
 46 following 4 tables these notations with their semantics and equivalent OPL sentences are depicted.
 47
 48

Table 1. Entities – Things and States

Thing: Object or Process	Systemic	Environmental	State	Symbol
Informational	Object:	Object:	Regular	
	Process:	Process:	Initial	
Physical	Object:	Object:	Final	
	Process:	Process:	Default	

49 Things in OPM may be associated with roles. The relationships between roles are specified using the metamodel
 50 technique. The role of a thing is specified in the upper left corner of the thing.
 51
 52
 53

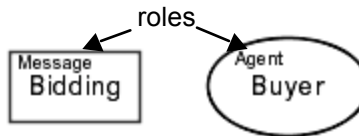


Table 2. Structural Links

Structural Relation Name	OPD Symbol	OPL Sentence
Aggregation-Participation		A consists of B .
Exhibition-Characterization		A exhibits B .
Generalization-Specialization		B is an A .
Classification-Instantiation		B is an instance of A .
Tagged Structural Link		A relates to B . A and B are equivalent.
XOR relation		

54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82

Table 3. Procedural Links

Type	Link Name	Semantics	OPD Symbol	OPL Sentence
Enabling	Instrument	The process requires the entity.		P requires A.
Transforming	Consumption	The process consumes the entity.		P consumes A.
	Result	The process generates (creates) the entity.		P yields A.
	Effect	The process changes (affects) the thing.		P affects A.
Conditional	Instrument	The process occurs if the entity exists (in some state). The process requires the entity.		P occurs if A exists. P requires A.
	Consumption	The process occurs if the entity exists (in some state). The process consumes the entity.		P occurs if A exists. P consumes A.
	Effect	The process occurs if the thing exists. The process changes (affects) the thing.		P occurs if A exists. P affects A.
	XOR relation			

