APPENDIX 2a

Entity Relationship statements

for Core Concepts in the

Software Design Minimal Methods

Appendix 2b: Core ER Diagrams

Appendix 2c:Core ER Concordance

Method has Name (Varible). [n001]
Method has FromViewPoint (Variable). [n002]
Method has ToViewPoint (Variable). [n003]
Method may have Dual (Variable). [n004]

StateMachine is_a Method. [n005]
StateMachine has CurrentState (Variable). [n006]
StateMachine has StateVectors (List). [n007]
StateMachine has Mode (Variable). [n008]
StateMachine has SMInput (Parameter). [n009]
StateMachine has SMOutput (Parameter). [n010]
StateVector has Variables (Set). [n011]
StateVector has Event (Variable). [n012]
StateVector has NowCurrentState (Variable). [n013]
StateVector has NextState (Variable). [n014]
StateVector has Action (Variable). [n015]
Action is_a FunctionName.
StateVector is_a Trigger for Function. [n017]
NextState maps_to CurrentState. [n018]
CurrentState maps_to NowCurrentState. [n019]
Event maps_to SMInput. [n020]
FunctionName maps_to SMOutput. [n021]
Mode maps_to StateVectors. [n022]
StateMachine has Operations (Function List). [n023]
GetCurrentState(OUT CurrentState) is_a Operation. [n024]
SetInitialState(IN CurrentState) is_a Operation. [n025]
SetInitialVectorList(IN StateVectors) is_a Operation. [n026]
GetAction(IN SMInput-> Event; OUT SMOutput-> Action) is_a Operation. [n027]

Petrinet is_a Method. [n028]
Petrinet has PetriMatrix. [n029]
PetriMatrix has Places (List). [n030]
PetriMatrix has Transits (List). [n031]
PetriMatrix has PetriArcs (List). [n032]
Petrinet has Markers (List). [n033]
Places has Place (Variable). [n034]
Transits has Transit (Function). [n035]
PetriArcs has PetriArc (Relation) from Place to Transit. [n036]

Markers is_a Tokens (Type). [n037]
PetriArcs maps_to Places. [n038]
PetriArcs maps_to Transits. [n039]
Markers propagate_along PetriArcs. [n040]
Markers has Color (Type). [n041]
Transit has PetriRules (List). [n042]
Transit has InputPlaces (List). [n043]
PetriRule has LeftHandSide. [n044]
PetriRule has RightHandSide. [n045]
LeftHandSide has Marker Colors in InputPlaces. [n046]
RightHandSide triggers Transit. [n047]
Petrinet has Operations (List). [n048]
FirePetriNet(null) is_a Operation. [n049]

Decomposition is_a Method. [n050]
Decomposition has Mode (Type). [n051]
Decomposition has Externals (source or sink). [n052]
Decomposition has Context (Bubble). [n053]
Context has Bubbles (Function). [n054]
Decomposition has DataArcs (Relation). [n055]
Decomposition has DataStore (Holder). [n056]
Bubbles decompose_into Bubbles (List). [n057]
Bubble has Mode. [n058]
DataItems has DataItem (Variable). [n059]
Bubble has ControlArc (Relation). [n060]
Bubble has ControlSpec (Holder). [n061]
DataArcs maps_to Bubbles. [n062]
DataArcs maps_to DataStore. [n063]
DataArcs maps_to Externals. [n064]
Bubble has InputDataArcs (Parameters). [n065]
Bubble has OutputDataArcs (Parameters). [n066]
When InputDataArcs present Bubble is Triggered. [n067]
ControlSpec may_have StateMachine. [n068]
ControlSpec may_have DecisionTable. [n069]
ControlSpec may_have ProcessActivationTable. [n070]
ControlSpec may_have PetriNet. [n071]
ControlSpec establishes Mode. [n072]
Function may_have Loop. [n073]

Function may_have Selector. [n074]
Function may_have Equation. [n075]
Function may_have Rule. [n076]
Function has atleast one Assignment.

Articulation is_a Method. [n078]
Articulation has Situation. [n079]
Situation has Entities with RelationArcs. [n080]
Articulation has DataDictionary (List). [n081]
DataDictionary has Entity. [n082]
DataDictionary has RelationArc (Relation). [n083]
RelationArcs map_to Entities from Entities. [n084]
Entity decomposes_into Entities (List). [n085]
Entity has DataStore (Holder). [n086]
Entity has Operations (Function). [n087]
DataStores has DataItems (List). [n088]
Operations modify DataItems. [n089]
DataItems has DataTypes (Type). [n090]
RelationArc has RelationAttributes (Variable). [n091]

DARTS is_a method. [n092]
DARTS has DistributedDesign. [n093]
DARTS has ConcurrentDesign. [n094]
DARTS has CommunicationsChannels (List). [n094]
DistributedDesign has ProcessingElements. [n096]
ConcurrentDesign has ProcessingElements. [n097]
ProcessorArrays is_a ProcessingElement. [n098]
ProcessorArrays decompose_into ProcessorArrays. [n099]
ProcessorArrays has Processors. [n100]
Task is_a ProcessingElement. [n101]
Processors has Tasks. [n102]
Tasks decompose_into Tasks. [n103]
CommunicationChannel has CommunicationMechanism. [n104]
CommunicationMechanism may_be Queue. [n105]
CommunicationMechanism may_be Rendezvous. [n106]
CommunicationMechanism may_be Semaphore. [n107]
CommunicationMechanism may_be Flag. [n108]
CommunicationMechanism may_be Variable. [n109]

CommunicationChannel has Protocol. [n110]
CommunicaitonChannel has DataArcs. [n111]
Protocol has Messages. [n112]
Message has DataItems. [n113]
Protocol has Sender StateMachine. [n114]
Protocol has Receiver StateMachine. [n115]
DARTS has Monitors. [n116]
Monitor has DataStore. [n117]
Monitor has Semaphore. [n118]
Task receives Message from CommunicationChannel. [n119]
Task has Function. [n120]
Function maps_to ProcessingElements. [n121]
Task has Selector of Function. [n122]
Task has ExecutiveLoop. [n123]

Allocation is_a Method. [n124]
Allocation has FunctionalMappings (List). [n125]
FunctionalMapping depends_on SystemMode (Variable). [n126]
FunctionalMapping has Functions (List). [n127]
FunctionalMapping has ProcessingElements (List). [n128]
FunctionalMapping has FunctionalArc (List of Relations). [n129]
FunctionalArc maps_to ProcessingElement from Function. [n130]

VirtualMachine is_a Method. [n131]
VirtualMachine decomposes_into VirtualMachines. [n132]
VirtualMachine has Instructions. [n133]
VirtualMachine may_have StateMachine. [n134]
Instruction is_a Function. [n135]

WorldLine is_a Method. [n136]
Worldline has Messages (List) associated with one ProcessingElement. [n137]

Scenario is_a Method. [n138]
Scenario has causally related Messages (List) between ProcessingElements. zn39]

DesignElementFlow is_a Method. [n140]
DesignElementFlow has DesignElements. [n141]
DesignElementFlow has System. [n142]

System has System. [n143]
System has StateMachine. [n144]
DesignElment has StateMachine. [n145]
System States maps_to DesignElement States. [n146]
System Actions maps_to DesignElement Actions. [n147]
DesignElement Actions maps_to System States. [n148]
System Actions maps_to DesignElement States. [n149]

InformationFlow is_a Method. [n150]
InformationFlow has Variables. [n151]
InformationFlow has Datum. [n152]
InformationFlow has SynchronicMapping. [n153]
InformationFlow has DiachronicMapping. [n154]
In Synchronic mapping the values in a set of variables relate to a single timespan. [n155]
In Diachronic mapping the datum over time moves through a set of variables. [n156]
Datum maps_to Variables. [n157]
Variables maps_to Variables via Datum. [n158]

Temporality is_a Method. [n159]
Temporality has a Sheaf (Holder). [n160]
Sheaf has Bundles (Holder). [n161]
Sheaf has SignalArc (Relation). [n162]
SignalArc maps_to Signal from Signal. [n163]
SignalArc has IntervalConstraints. [n164]
IntervalConstraint may_be Before. [n165]
IntervalConstraint may_be After. [n166]
IntervalConstraint may_be During. [n167]
IntervalConstraint may_be Startings. [n168]
IntervalConstraint may_be Finishes. [n169]
IntervalConstraint may_be Overlapping. [n170]
IntervalConstraint may_be Meets. [n171]
IntervalConstraint may_be Equals. [n172]
Bundle has Signals (List). [n173]
Signal has Interval (List). [n174]
Signal has Lacune (List). [n175]
Bunch has Intervals (List). [n176]
Interval has Duration (Variable). [n177]
Interval has Event (String). [n178]

Interval has State (String). [n179]

Interval decomposes_into Intervals (List). [n180]

Apeiron Press

PO Box 4402
Garden Grove,
California 92842-4402

714-638-7376
714-638-1210
palmer@think.net
palmer@netcom.com
palmer@exo.com
Dataline 714-638-0876

Draft #1 950710 Editorial Copy.
Not for distribution.

This book was set using Framemaker document publishing software by the author.

Electronic Version in Adobe Acrobat PDF available at http://server.snni.com:80/~palmer/homepage.html

Library of Congress
Cataloging in Publication Data

Palmer, Kent Duane

WILD SOFTWARE META-SYSTEMS

Bibliography
Includes Index

1. Philosophy-- Ontology
2. Software Engineering
3. Software Design Methods

I. Title

[XXX000.X00 199x]
9x-xxxxx
ISBN 0-xxx-xxxxx-x

Keywords:

Software, Design Methods, Ontology,
Integral Software Engineering
Methodology, Systems Theory

