



TACKLING UNCERTAINTY IN CYBER-PHYSICAL SYSTEMS WITH AUTOMATED TESTING (U-TEST)

Shaukat Ali, shaukat@simula.no
Tao Yue, tao@simula.no
Man Zhang, man@simula.no
Simula Research Laboratory, Norway

DE-CPS 2016 Workshop, June 2016



www.u-test.eu



U-TEST

- Objective: Improve the dependability by Cost-Effective Uncertainty testing
- Means: Model-based and Search-based Testing
- Objective will be achieved by:
 - Uncertainty Taxonomy
 - Holistic Modeling and Testing Frameworks
 - Standards

OVERALL CONSORTIUM

Research Partners

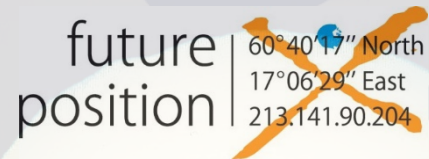
[**simula** . research laboratory]
- by thinking constantly about it



Test Bed Provider



Case Study Providers



Exploitation



Tool Vendors



Dissemination/ Administration/ Financial

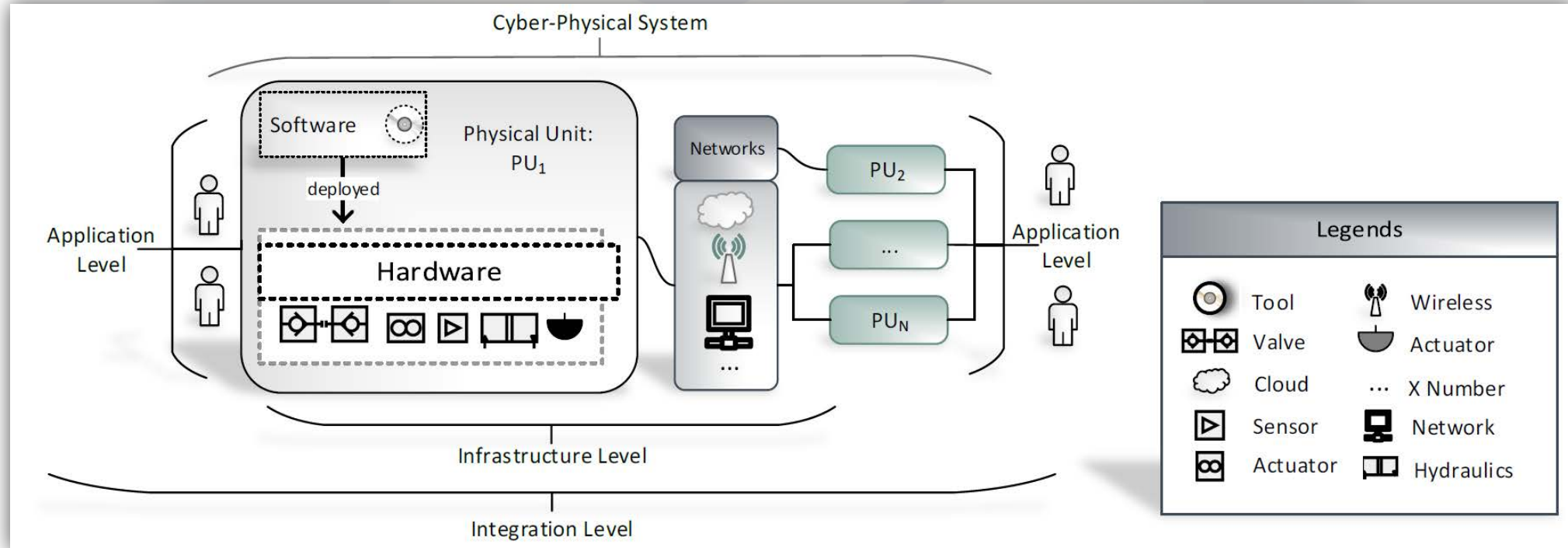


TESTING CPS UNDER UNCERTAINTY

- Motivation
 - ✓ Uncertainty is inherent in CPSs
 - ✓ Handling uncertainty in a graceful manner during the real operation of CPS is critical.
- Definition
 - ✓ The lack of certainty (i.e., knowledge) about the timing and nature of inputs, the state of a system, a future outcome, etc.
- Steps
 - ✓ Understanding Uncertainty
 - ✓ Modeling Uncertainty
 - ✓ Testing Uncertainty

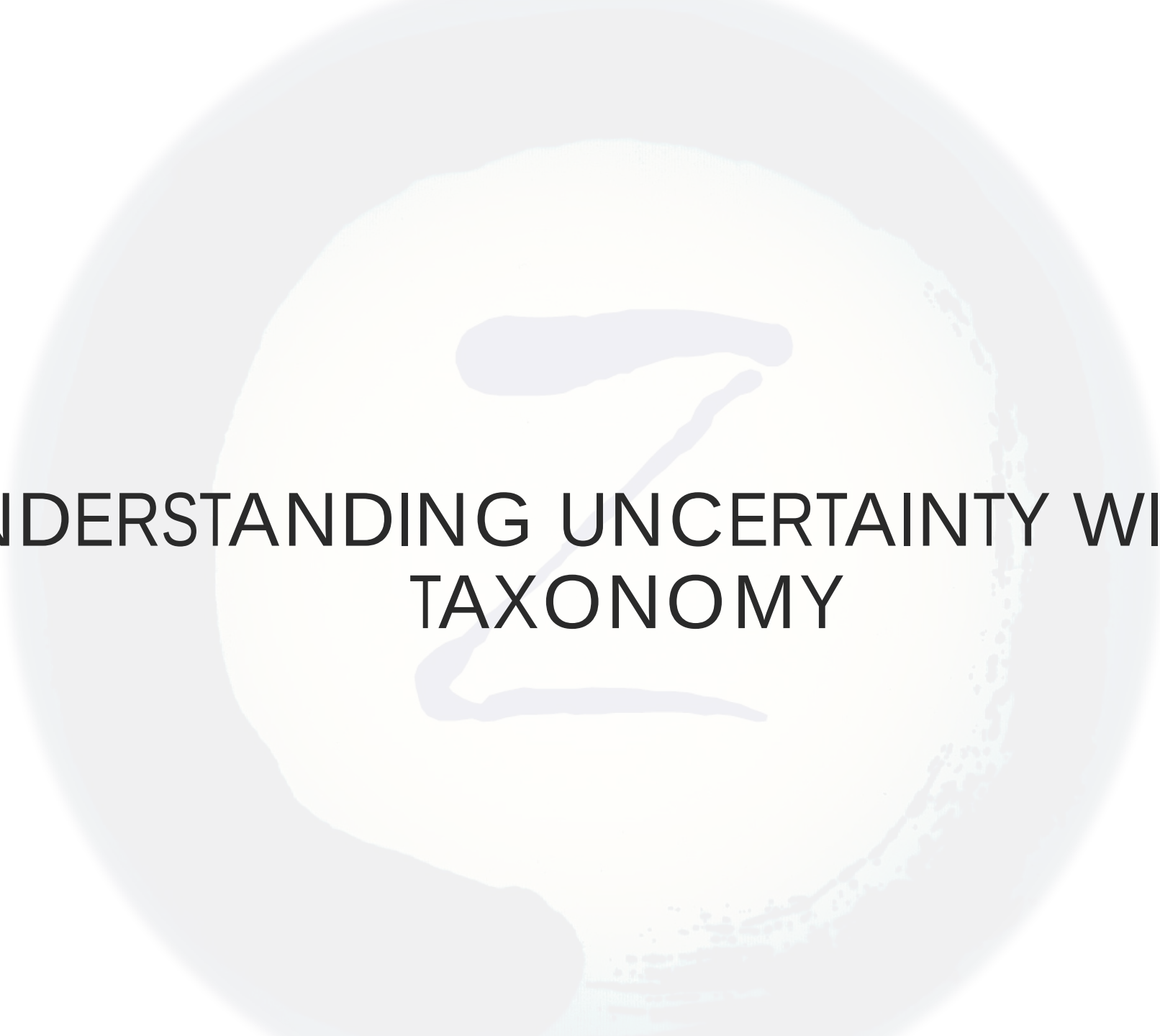
TESTING LEVELS FOR CPS

- Application Level : Events and data coming from the user space, e.g., from applications and human
- Infrastructure Level : Events and data coming from, e.g., physical units, network equipment, and cloud infrastructure
- Integration Level : Interactions between the above two levels



M. ZHANG, B. SELIC, S. ALI, T. YUE, O. OKARIZ AND R. NORGREN, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model In European Conference on Modelling Foundations and Applications (ECMFA)., 2016.

M. Zhang, B. Selic, S. Ali, T. Yue, O. Okariz and R. Norgren, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model, <https://www.simula.no/file/u-modeltrfinalpdf/download>

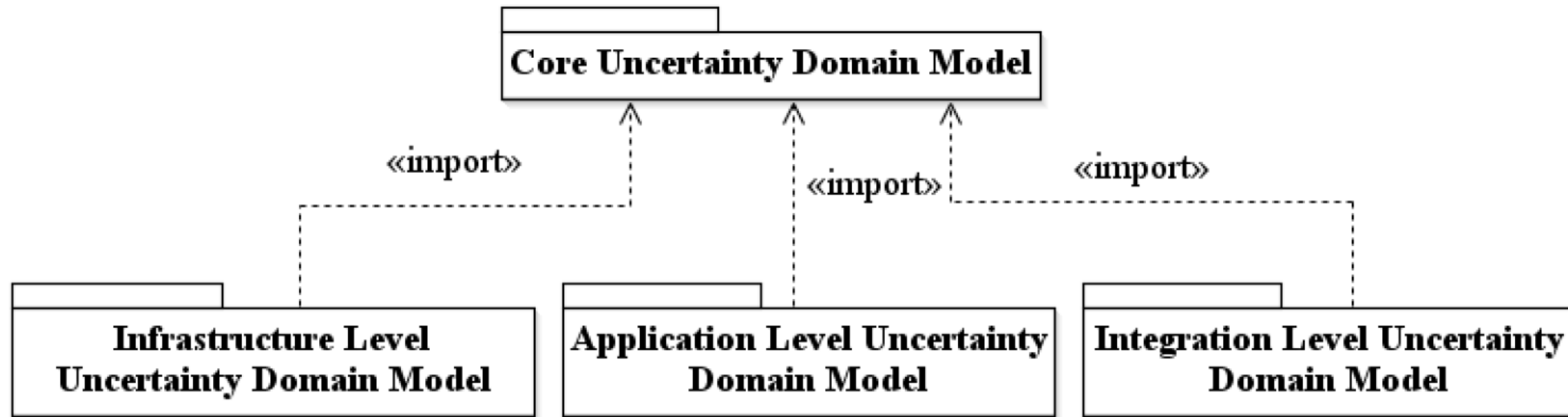


UNDERSTANDING UNCERTAINTY WITH U- TAXONOMY

U-TAXONOMY

- The *U-Taxonomy* takes a *subjective* approach to represent uncertainty.
- Provide a unified and comprehensive description of uncertainties.
- Classify uncertainties with the aim of identifying common representational patterns.
- Provide a reference model for systematically collecting uncertainty requirements.
- Serve as a methodological baseline for modeling uncertain behaviors in CPS.

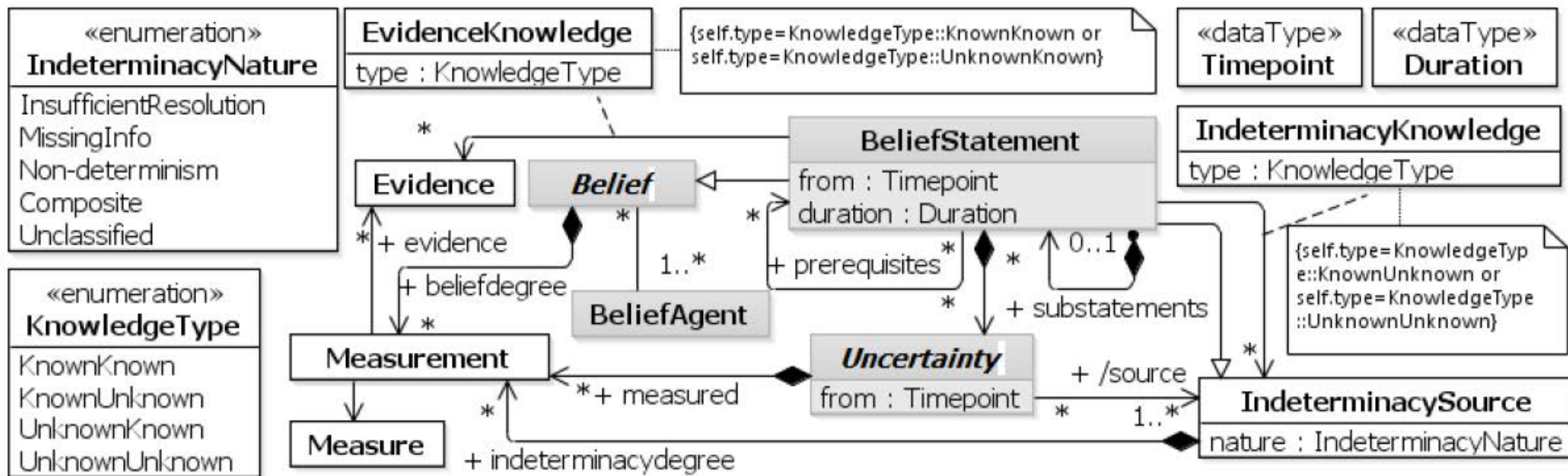
STRUCTURE OF U-TAXONOMY



M. ZHANG, B. SELIC, S. ALI, T. YUE, O. OKARIZ AND R. NORGREN, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model In European Conference on Modelling Foundations and Applications (ECMFA)., 2016.

M. Zhang, B. Selic, S. Ali, T. Yue, O. Okariz and R. Norgren, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model, <https://www.simula.no/file/u-modeltrfinalpdf/download>

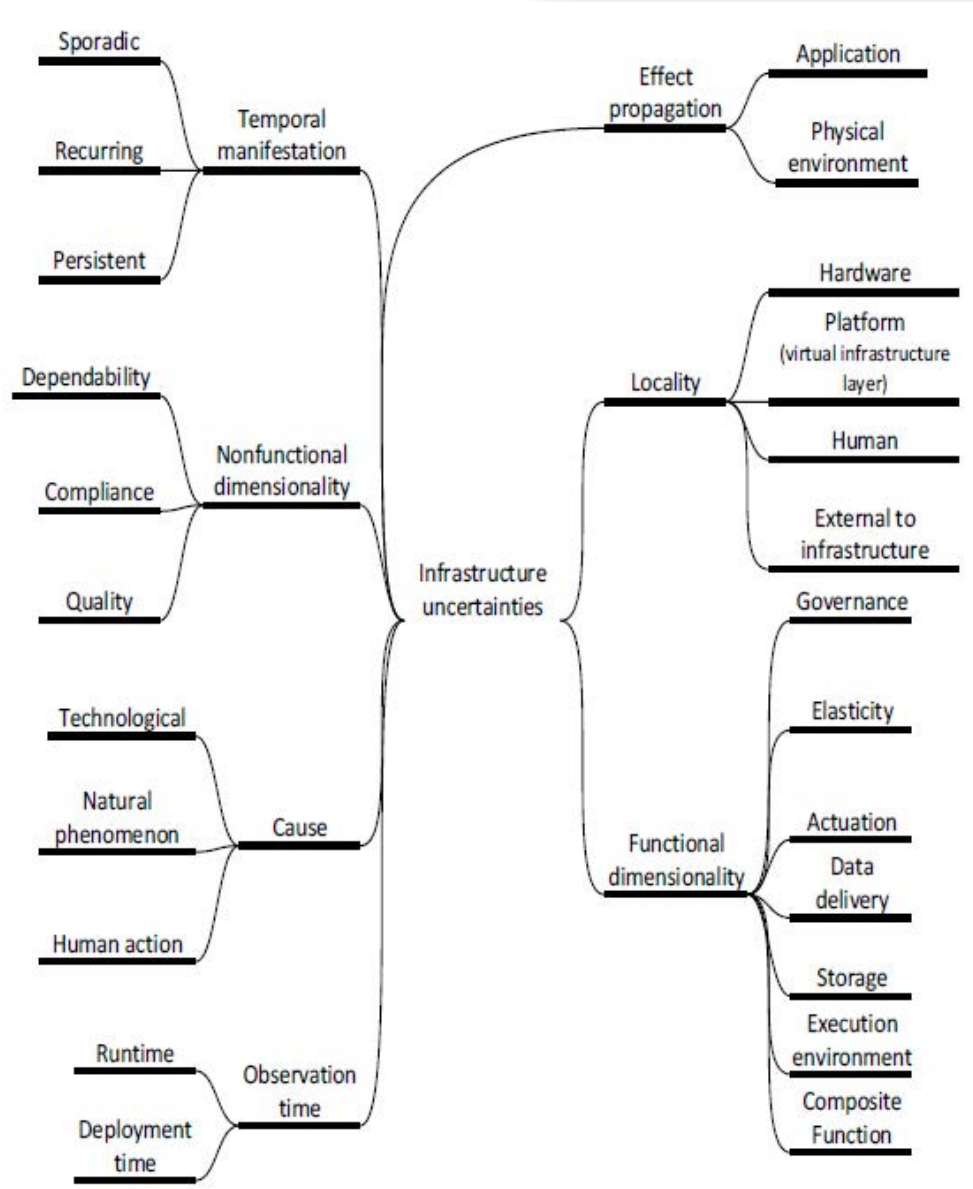
CORE UNCERTAINTY DOMAIN MODEL



M. ZHANG, B. SELIC, S. ALI, T. YUE, O. OKARIZ AND R. NORGRÉN, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model In European Conference on Modelling Foundations and Applications (ECMFA)., 2016.

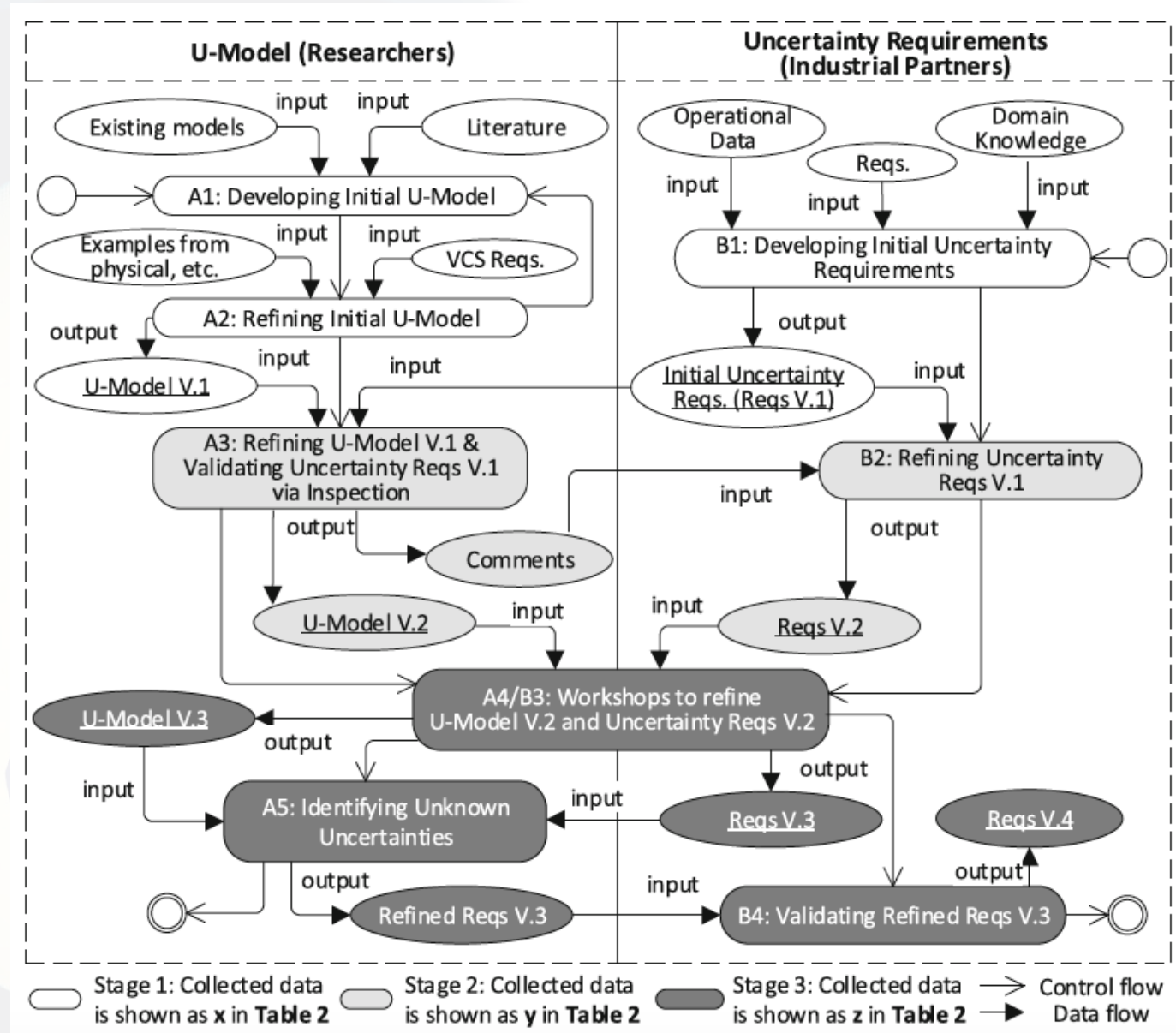
M. Zhang, B. Selic, S. Ali, T. Yue, O. Okariz and R. Norgren, Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model, <https://www.simula.no/file/u-modeltrfinalpdf/download>

INFRASTRUCTURE LEVEL TAXONOMY



Stefan Nastic and Hong-Linh Truong, Infrastructure-Level Uncertainties V2.0,
<http://dsg.tuwien.ac.at/staff/snastic/public/u-taxonomy.pdf>

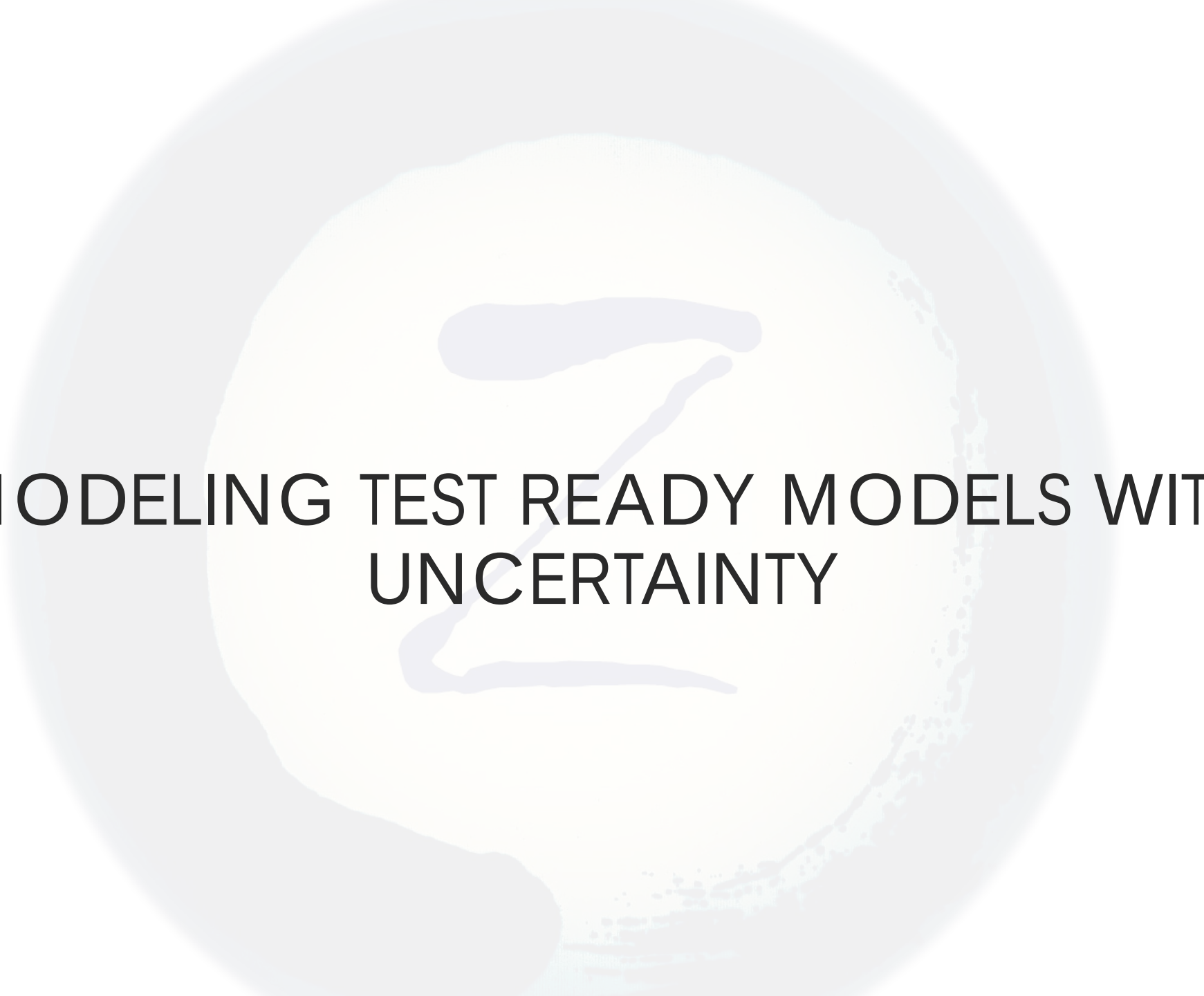
EVALUATION



RESULTS

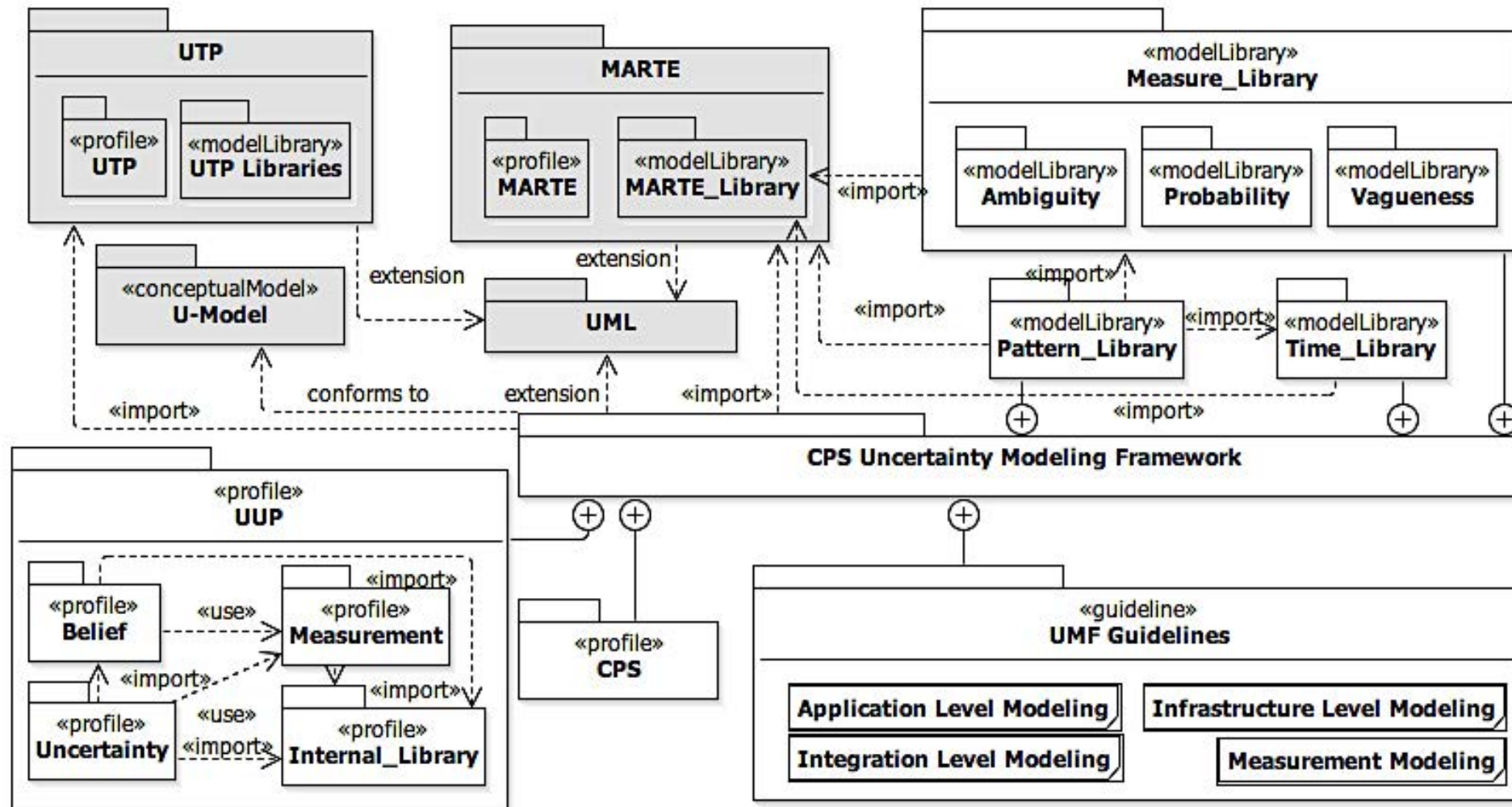
Concept		AW					GS					Freq
		$x $	y	z	R1*	R2*	x	y	z	R1	R2	Total ⁺
Uncertainty	Content	14	36	55	1.57	0.53	16	20	36	0.25	0.80	91
	Time	6	16	28	1.67	0.75	5	11	22	1.20	1.00	50
	Occurrence	27	81	126	2.00	0.56	6	50	79	7.33	0.58	205
	Environment	13	15	22	0.15	0.47	4	6	10	0.50	0.67	32
	Geographical Location	4	11	14	1.75	0.27	3	11	17	2.67	0.55	31
Sum for x, y, z/Average for R1, R2		64	159	245	1.43	0.51	34	98	164	2.39	0.72	409
Indeterminacy	Insufficient Resolution	7	18	24	1.57	0.33	11	14	18	0.27	0.29	42
	Non-determinism	7	45	52	5.43	0.16	11	20	37	0.82	0.85	89
	MissingInfo	2	19	24	8.50	0.26	0	5	7	N/A	0.40	31
Sum for x, y, z/Average for R1, R2		16	82	100	2.67	0.43	22	39	62	0.55	0.57	162
Measure	Fuzziness	6	22	51	2.67	1.32	6	15	25	1.50	0.67	76
	NonSpecificity	16	40	73	1.50	0.83	12	26	46	1.17	0.77	119
	Probability	18	56	98	2.11	0.75	4	37	50	8.25	0.35	148
Sum for x, y, z/Average for R1, R2		40	118	222	2.09	0.96	22	78	121	3.64	0.60	343

^{*}R1 = $y/x - 1$ ^{*}R2 = $z/y - 1$ ⁺Total = AW(z)+GS(z) Freq is Frequency

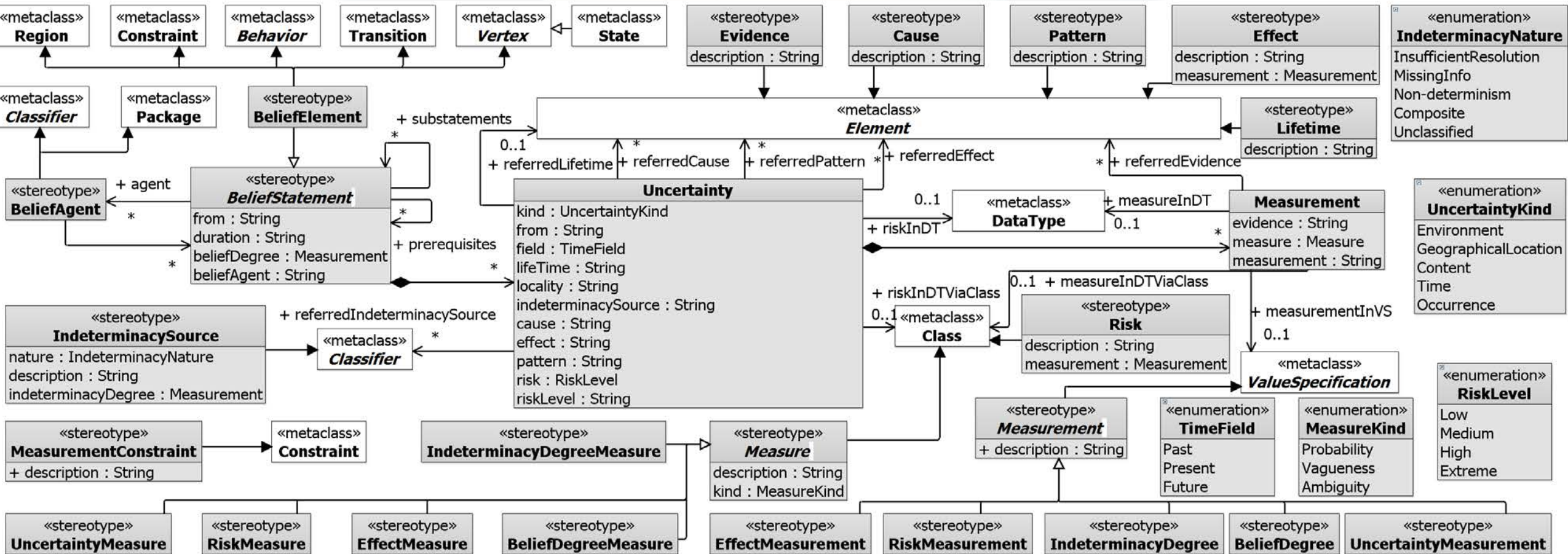


MODELING TEST READY MODELS WITH UNCERTAINTY

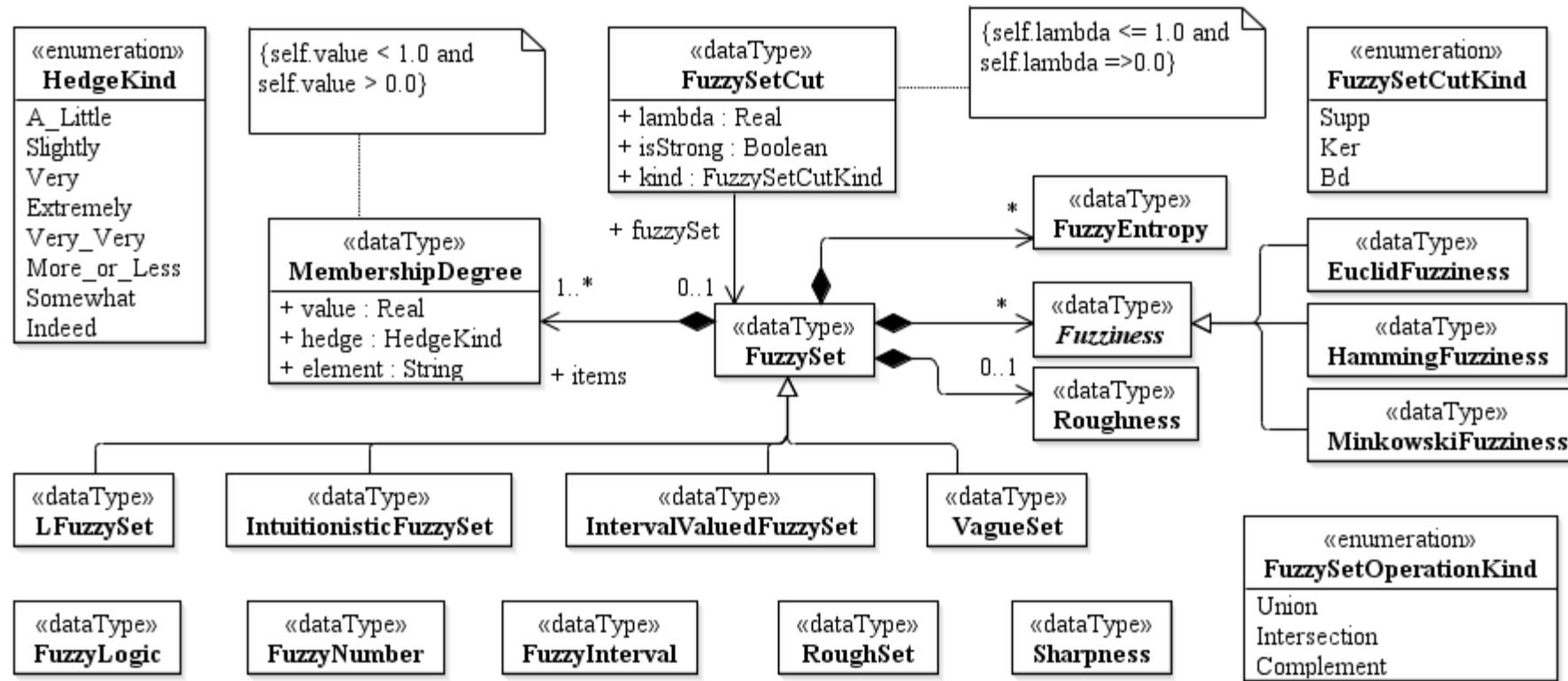
UNCERTAINTY MODELING FRAMEWORK (UMF)



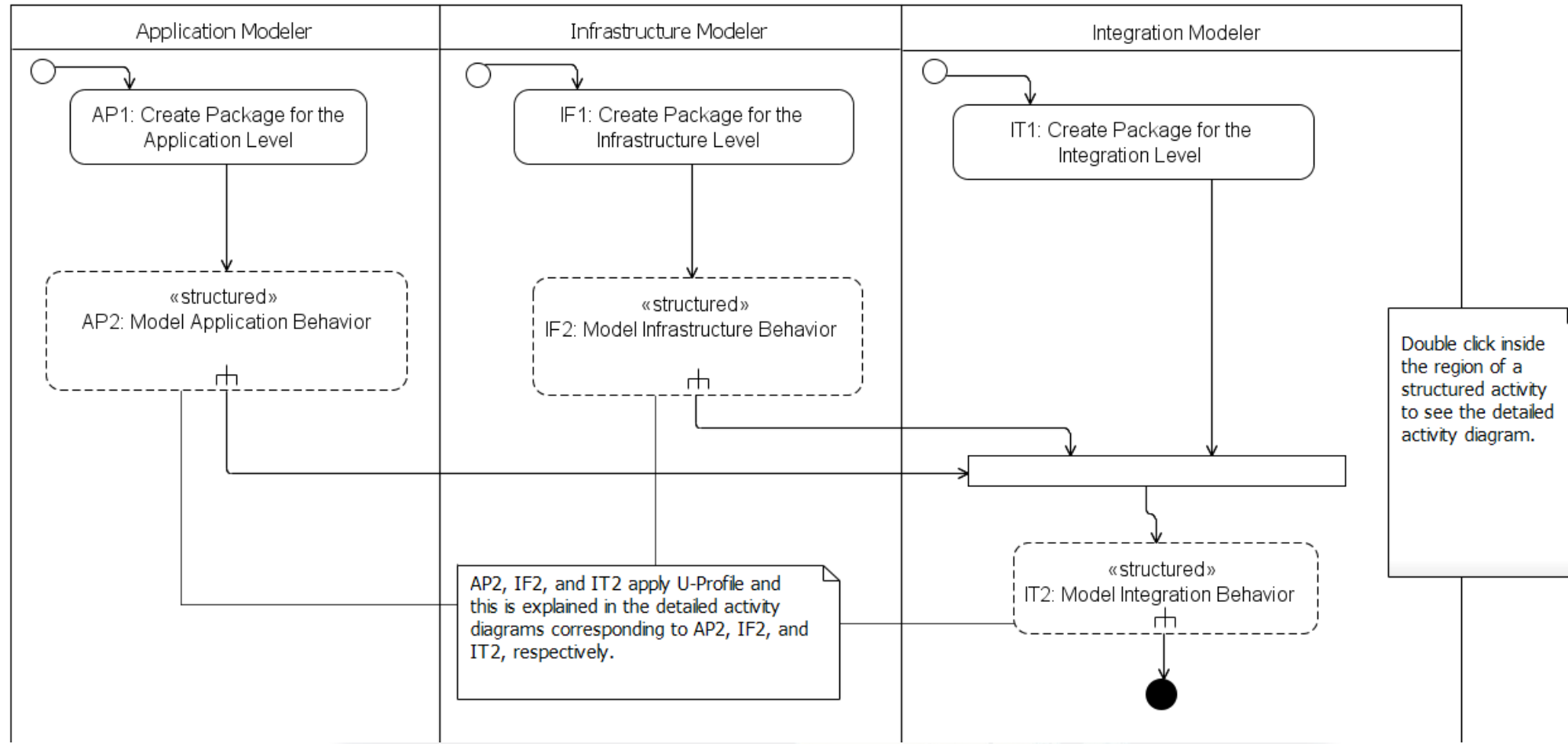
UML UNCERTAINTY PROFILE (UUP): IMPLEMENTATION OF U-TAXONOMY



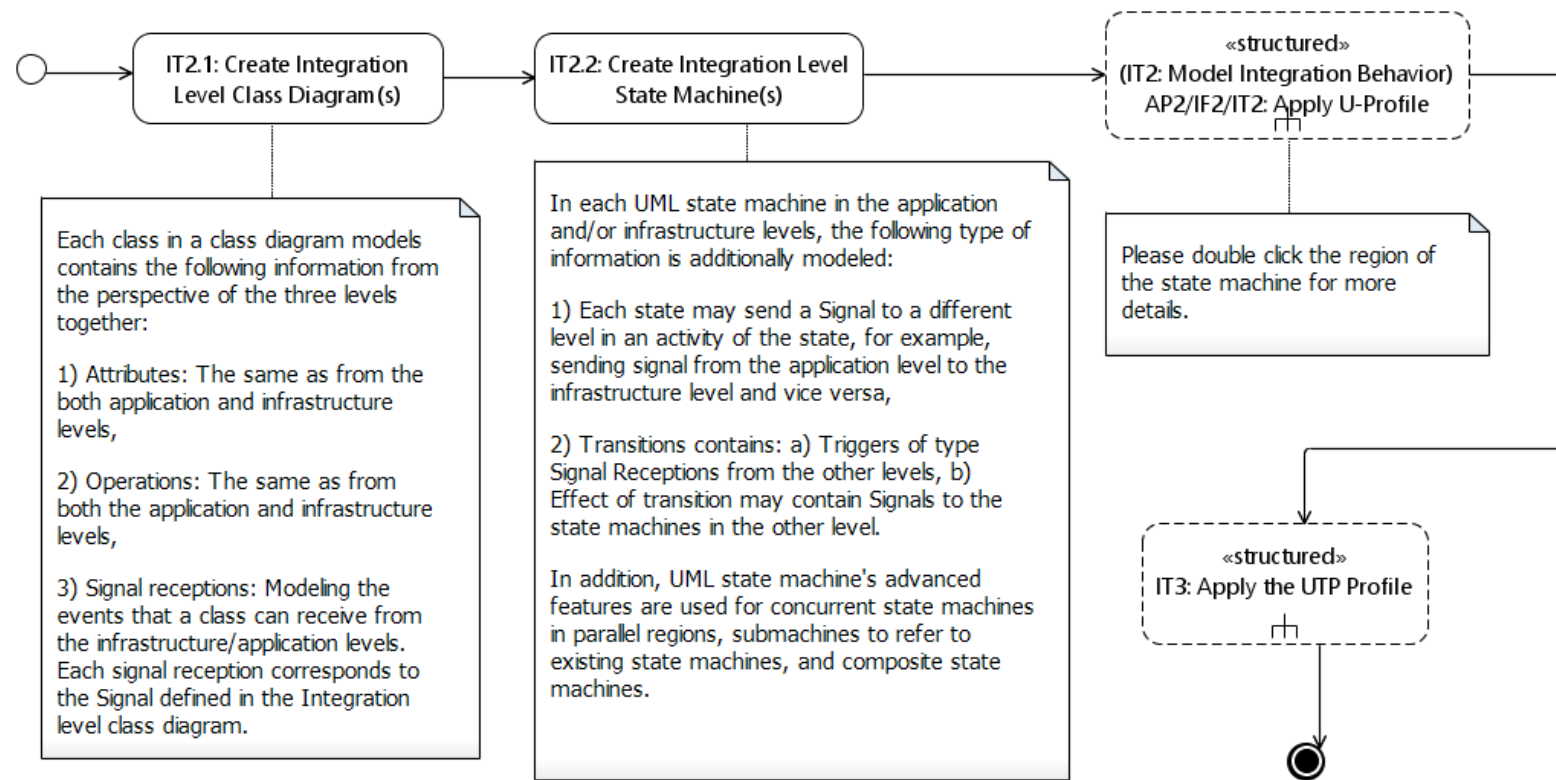
VAGUENESS LIBRARY



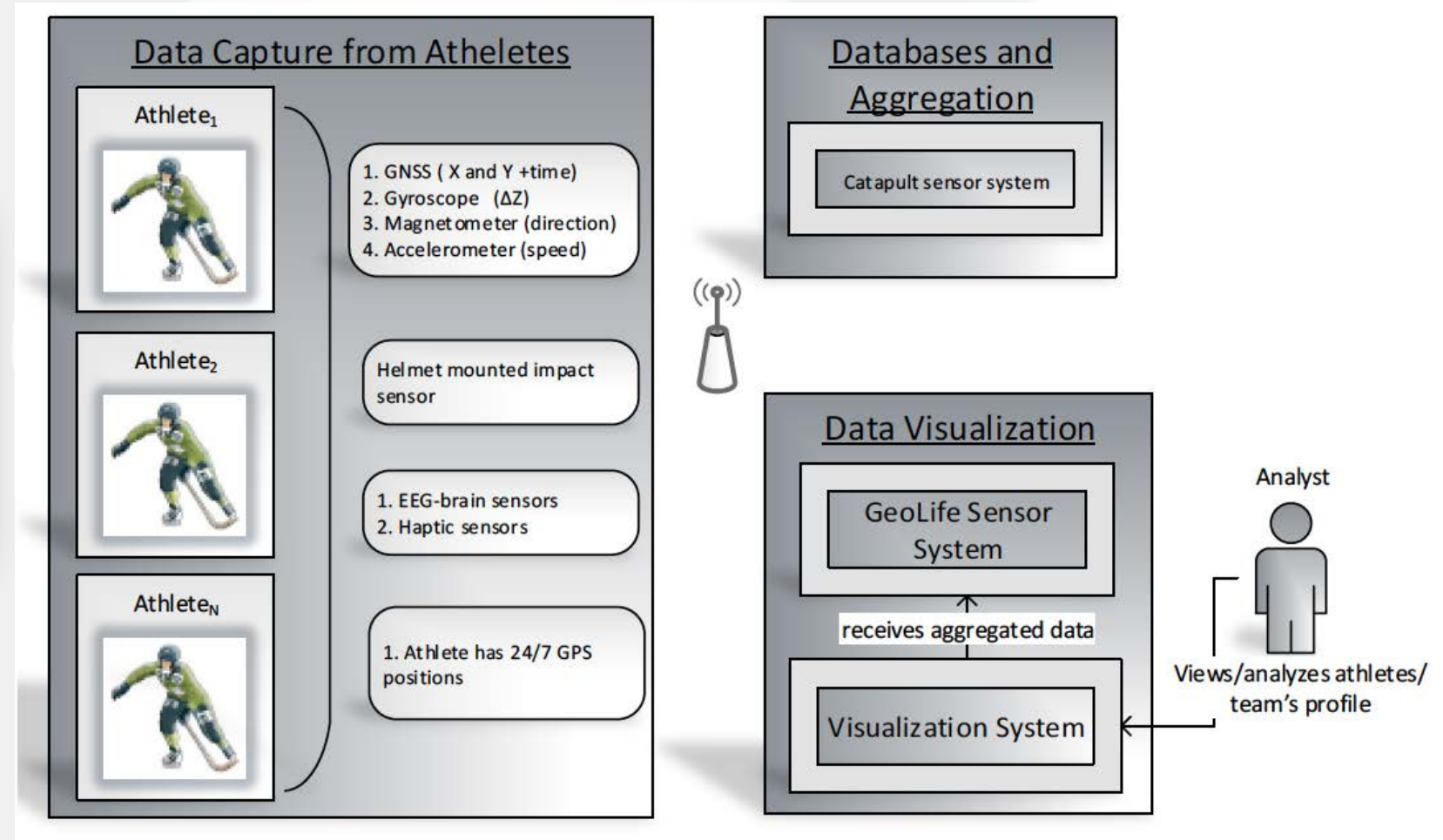
METHODOLOGY EXAMPLE (1/2)



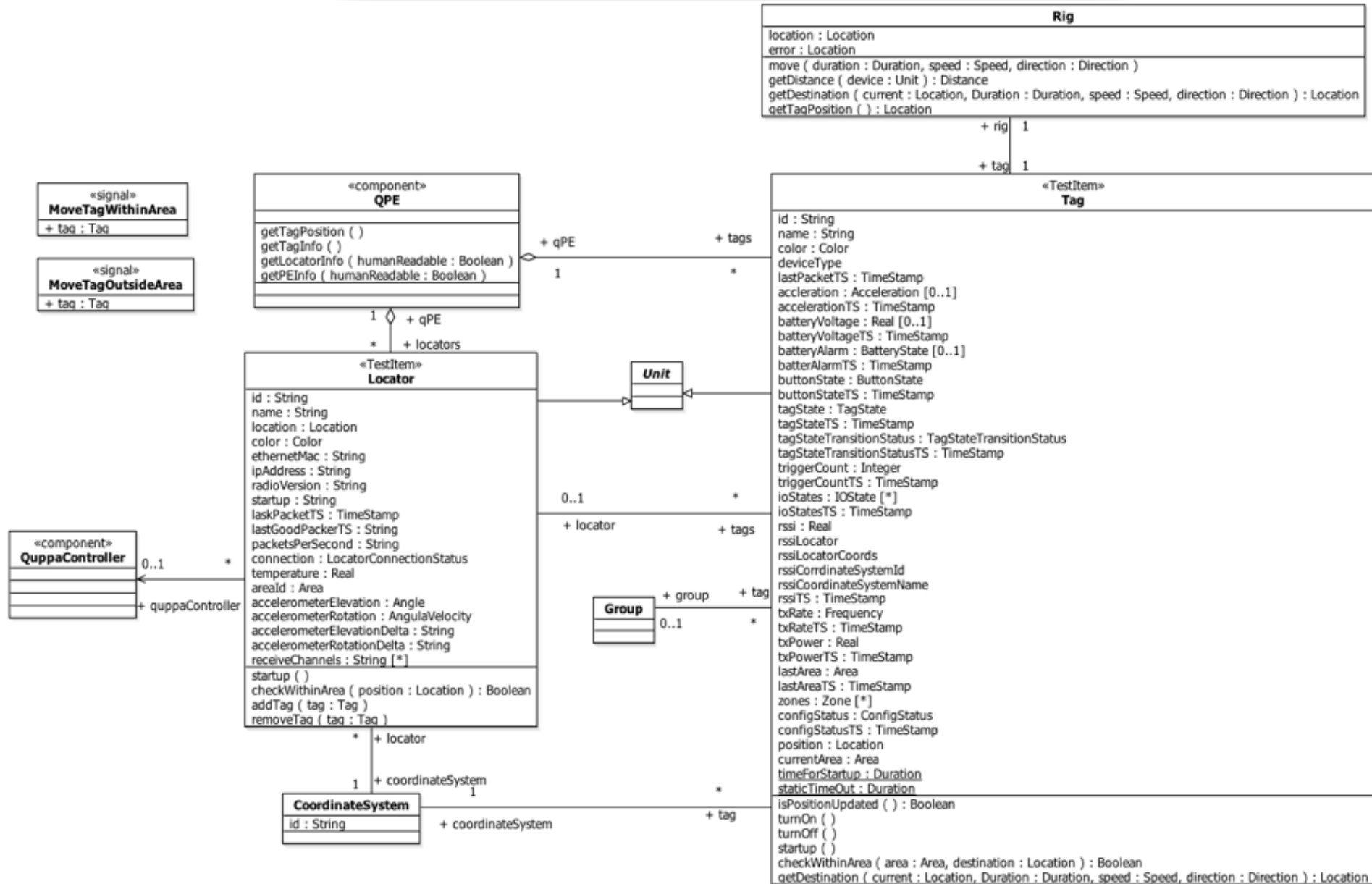
METHODOLOGY EXAMPLE (2/2)



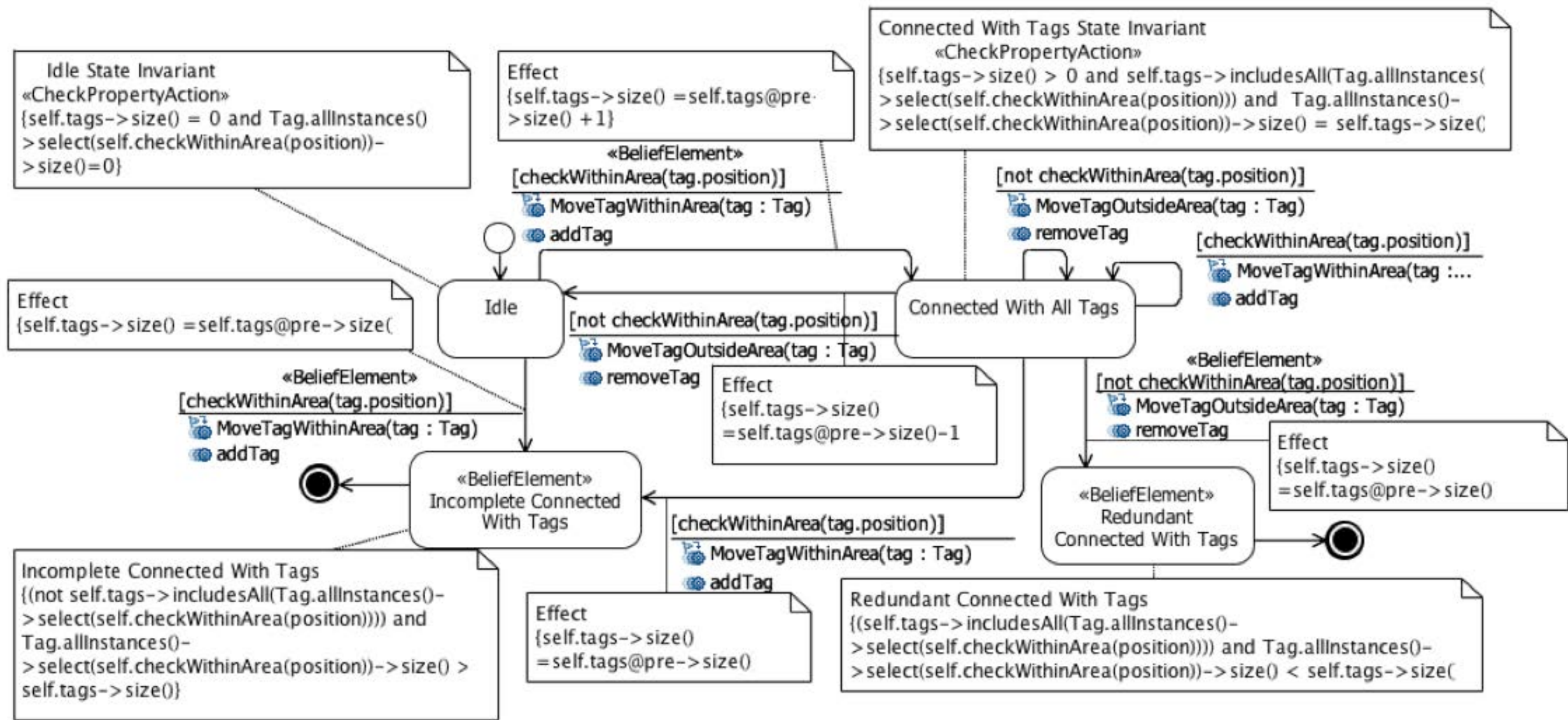
CASE STUDY PROVIDERS: GEO SPORTS



EXAMPLE MODELS: GEOSPORTS CASE STUDY



Locator: Connect with Tags



CASE STUDY PROVIDERS: HANDLING SYSTEMS



EXAMPLE MODELS: ULMA HANDLING SYSTEMS CASE STUDY

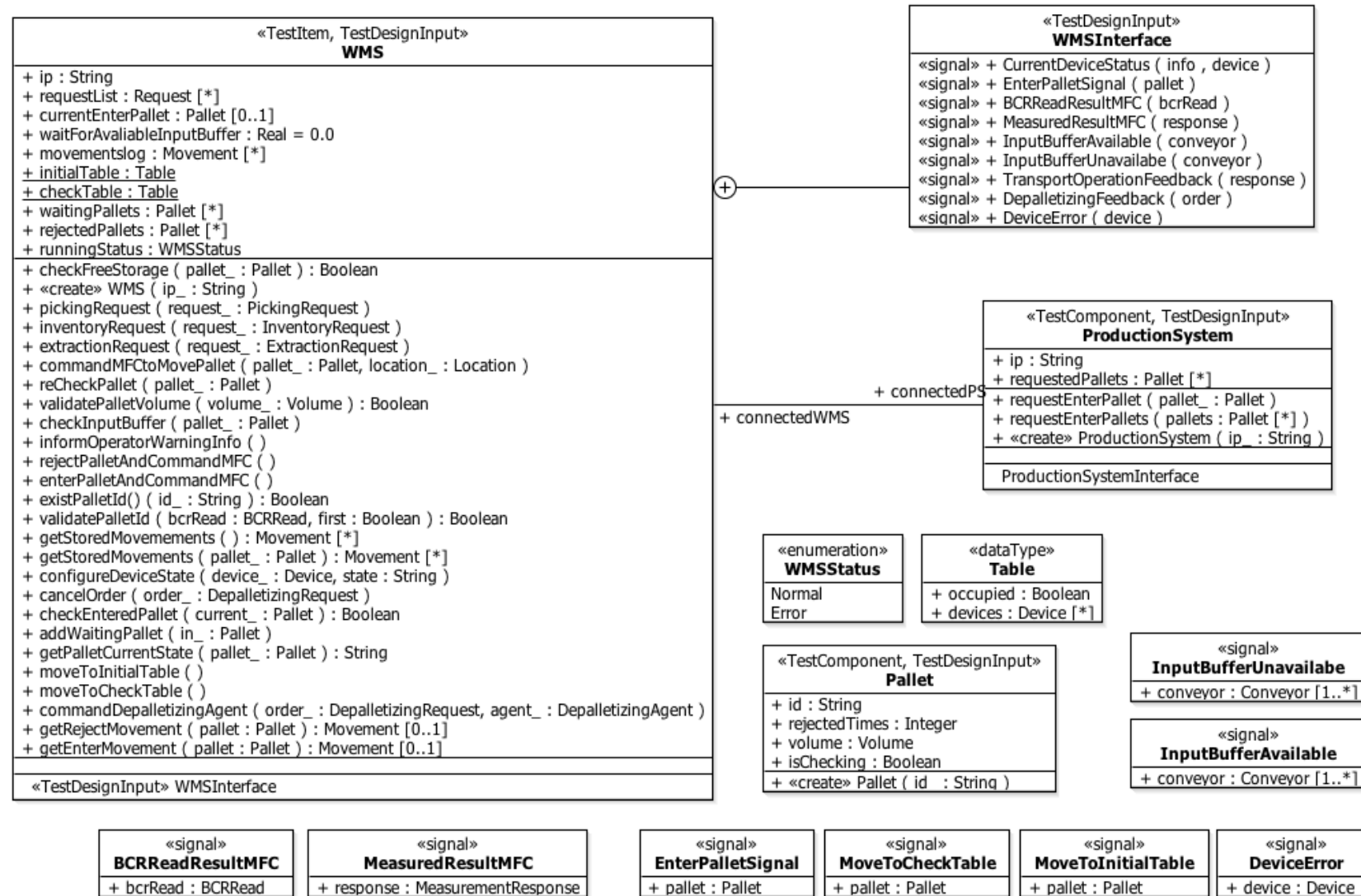
```

classDiagram
    class WMS {
        <<TestItem, TestDesignInput>>
        +ip : String
        +requestList : Request [*]
        +currentEnterPallet : Pallet [0..1]
        +waitForAvailiableInputBuffer : Real = 0.0
        +movementslog : Movement [*]
        +initialTable : Table
        +checkTable : Table
        +waitingPallets : Pallet [*]
        +rejectedPallets : Pallet [*]
        +runningStatus : WMSStatus
        +checkFreeStorage ( pallet_ : Pallet ) : Boolean
        +«create» WMS ( ip_ : String )
        +pickingRequest ( request_ : PickingRequest )
        +inventoryRequest ( request_ : InventoryRequest )
        +extractionRequest ( request_ : ExtractionRequest )
        +commandMFCtoMovePallet ( pallet_ : Pallet, location_ : Location )
        +reCheckPallet ( pallet_ : Pallet )
        +validatePalletVolume ( volume_ : Volume ) : Boolean
        +checkInputBuffer ( pallet_ : Pallet )
        +informOperatorWarningInfo ( )
        +rejectPalletAndCommandMFC ( )
        +enterPalletAndCommandMFC ( )
        +existPalletId ( id_ : String ) : Boolean
        +validatePalletId ( bcrRead : BCRRead, first : Boolean ) : Boolean
        +getStoredMovements ( ) : Movement [*]
        +getStoredMovements ( pallet_ : Pallet ) : Movement [*]
        +configureDeviceState ( device_ : Device, state : String )
        +cancelOrder ( order_ : DepalletizingRequest )
        +checkEnteredPallet ( current_ : Pallet ) : Boolean
        +addWaitingPallet ( in_ : Pallet )
        +getPalletCurrentState ( pallet_ : Pallet ) : String
        +moveToInitialTable ( )
        +moveToCheckTable ( )
        +commandDepalletizingAgent ( order_ : DepalletizingRequest, agent_ : DepalletizingAgent )
        +getRejectMovement ( pallet : Pallet ) : Movement [0..1]
        +getEnterMovement ( pallet : Pallet ) : Movement [0..1]
    }
    class WMSInterface {
        <<TestDesignInput>>
        +CurrentDeviceStatus ( info , device )
        +EnterPalletSignal ( pallet )
        +BCRReadResultMFC ( bcrRead )
        +MeasuredResultMFC ( response )
        +InputBufferAvailable ( conveyor )
        +InputBufferUnavailabe ( conveyor )
        +TransportOperationFeedback ( response )
        +DepalletizingFeedback ( order )
        +DeviceError ( device )
    }
    class ProductionSystem {
        <<TestComponent, TestDesignInput>>
        +ip : String
        +requestedPallets : Pallet [*]
        +requestEnterPallet ( pallet_ : Pallet )
        +requestEnterPallets ( pallets : Pallet [*] )
        +«create» ProductionSystem ( ip_ : String )
        +ProductionSystemInterface
    }
    class WMSStatus {
        <<enumeration>>
        +Normal
        +Error
    }
    class Table {
        <<dataType>>
        +occupied : Boolean
        +devices : Device [*]
    }
    class InputBufferUnavailabe {
        <<signal>>
        +conveyor : Conveyor [1..*]
    }
    class InputBufferAvailable {
        <<signal>>
        +conveyor : Conveyor [1..*]
    }
    class BCRReadResultMFC {
        <<signal>>
        +bcrRead : BCRRead
    }
    class MeasuredResultMFC {
        <<signal>>
        +response : MeasurementResponse
    }
    class EnterPalletSignal {
        <<signal>>
        +pallet : Pallet
    }
    class MoveToCheckTable {
        <<signal>>
        +pallet : Pallet
    }
    class MoveToInitialTable {
        <<signal>>
        +pallet : Pallet
    }
    class DeviceError {
        <<signal>>
        +device : Device
    }
    class Pallet {
        <<TestComponent, TestDesignInput>>
        +id : String
        +rejectedTimes : Integer
        +volume : Volume
        +isChecking : Boolean
        +«create» Pallet ( id : String )
    }
    WMS --> WMSInterface
    WMS --> ProductionSystem : + connectedPS
    WMS --> WMSInterface : + connectedWMS
    WMS --> WMSStatus
    WMS --> Table
    WMS --> InputBufferUnavailabe
    WMS --> InputBufferAvailable
    WMS --> BCRReadResultMFC
    WMS --> MeasuredResultMFC
    WMS --> EnterPalletSignal
    WMS --> MoveToCheckTable
    WMS --> MoveToInitialTable
    WMS --> DeviceError
    WMS --> Pallet
  
```

The diagram illustrates the architecture of the ULMA Handling Systems Case Study. It features a central **WMS** (Warehouse Management System) class, which is a **TestItem** and **TestDesignInput**. The **WMS** class has attributes such as `ip` (String), `requestList` (Request [*]), `currentEnterPallet` (Pallet [0..1]), `waitForAvailiableInputBuffer` (Real = 0.0), `movementslog` (Movement [*]), `initialTable` (Table), `checkTable` (Table), `waitingPallets` (Pallet [*]), `rejectedPallets` (Pallet [*]), and `runningStatus` (WMSStatus). It includes methods like `checkFreeStorage`, `«create» WMS`, `pickingRequest`, `inventoryRequest`, `extractionRequest`, `commandMFCtoMovePallet`, `reCheckPallet`, `validatePalletVolume`, `checkInputBuffer`, `informOperatorWarningInfo`, `rejectPalletAndCommandMFC`, `enterPalletAndCommandMFC`, `existPalletId`, `validatePalletId`, `getStoredMovements`, `configureDeviceState`, `cancelOrder`, `checkEnteredPallet`, `addWaitingPallet`, `getPalletCurrentState`, `moveToInitialTable`, `moveToCheckTable`, `commandDepalletizingAgent`, `getRejectMovement`, and `getEnterMovement`.

The **WMS** class is connected to several other components:

- WMSInterface** (TestDesignInput): A signal interface that defines signals like `CurrentDeviceStatus`, `EnterPalletSignal`, `BCRReadResultMFC`, `MeasuredResultMFC`, `InputBufferAvailable`, `InputBufferUnavailabe`, `TransportOperationFeedback`, `DepalletizingFeedback`, and `DeviceError`.
- ProductionSystem** (TestComponent, TestDesignInput): A component that manages the production system. It has attributes `ip` (String) and `requestedPallets` (Pallet [*]), and methods `requestEnterPallet`, `requestEnterPallets`, and `«create» ProductionSystem`. It also defines a `ProductionSystemInterface`.
- WMSStatus** (enumeration): An enumeration with values `Normal` and `Error`.
- Table** (dataType): A data type with attributes `occupied` (Boolean) and `devices` (Device [*]).
- InputBufferUnavailabe** (signal): A signal class with attribute `conveyor` (Conveyor [1..*]).
- InputBufferAvailable** (signal): A signal class with attribute `conveyor` (Conveyor [1..*]).
- BCRReadResultMFC** (signal): A signal class with attribute `bcrRead` (BCRRead).
- MeasuredResultMFC** (signal): A signal class with attribute `response` (MeasurementResponse).
- EnterPalletSignal** (signal): A signal class with attribute `pallet` (Pallet).
- MoveToCheckTable** (signal): A signal class with attribute `pallet` (Pallet).
- MoveToInitialTable** (signal): A signal class with attribute `pallet` (Pallet).
- DeviceError** (signal): A signal class with attribute `device` (Device).
- Pallet** (TestComponent, TestDesignInput): A component representing a pallet. It has attributes `id` (String), `rejectedTimes` (Integer), `volume` (Volume), and `isChecking` (Boolean). It includes a `«create» Pallet` method.



EXAMPLE MODELS: ULMA HANDLING CASE STUDY

