Playing with xtUML

Alasdar Mullarney

Alasdar Mullarney Consulting

SMUG 2002

JSIMS Joint Simulation System

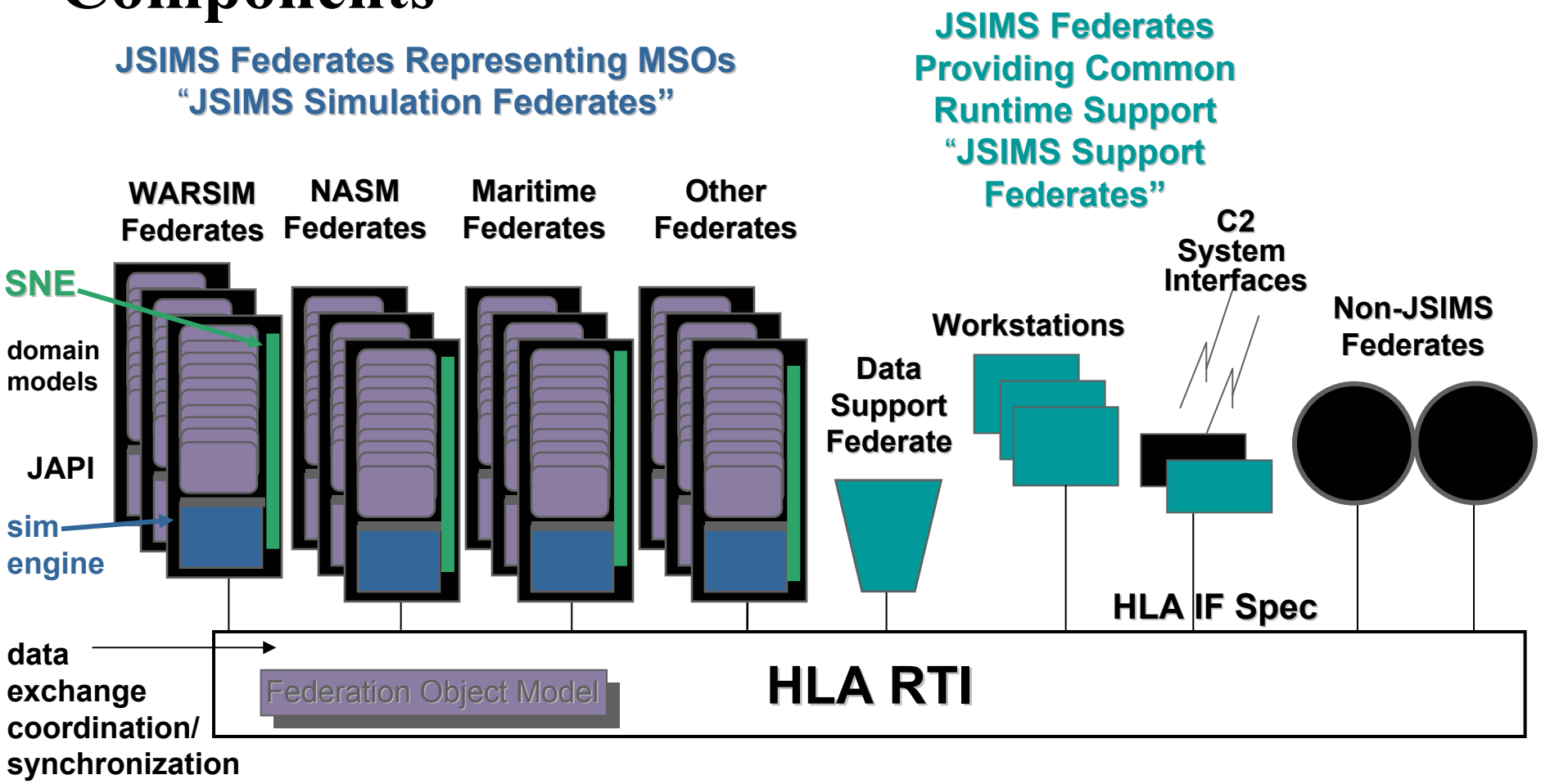
- **A training wargame**
  - **Joint = Joint Forces**
    - **Army, Airforce, Navy, USMC...**
- **1000's of simulated entities**
  - **Ships, Aircraft, Tanks, Missiles…**
    - **sensors, weapons, logistics...**
- **Multi-day/month scenarios**

JSIMS Architecture

- **Distributed HLA 'federate' simulations**
  - **HLA: a DoD standard simulation architecture**
- **Federates are independently developed**
  - **Army, Airforce, Navy, Intelligence….**
- **FOM - Federation Object Model**
  - **a data exchange model**

JSIMS Architecture

"JSIMS as an HLA Federation with Common Components"



US Navy (SPAWAR) is using xtUML

- **OOA using BridgePoint**
  - **10 analysts; varied experience**
- **MC-2020-based Recursive Design**
  - **4 architects**
- **Some consulting assistance!**

JSIMS Features

- **Size and complexity**
  - **9 modeled domains...and counting!**
- **Infrastructure interface**
  - **FOM - a very large realized domain**
    - **continual evolution!**
- **Parallel "Time Warp"**
  - **Optimistic/Rollback computation**
    - **granularity choices**

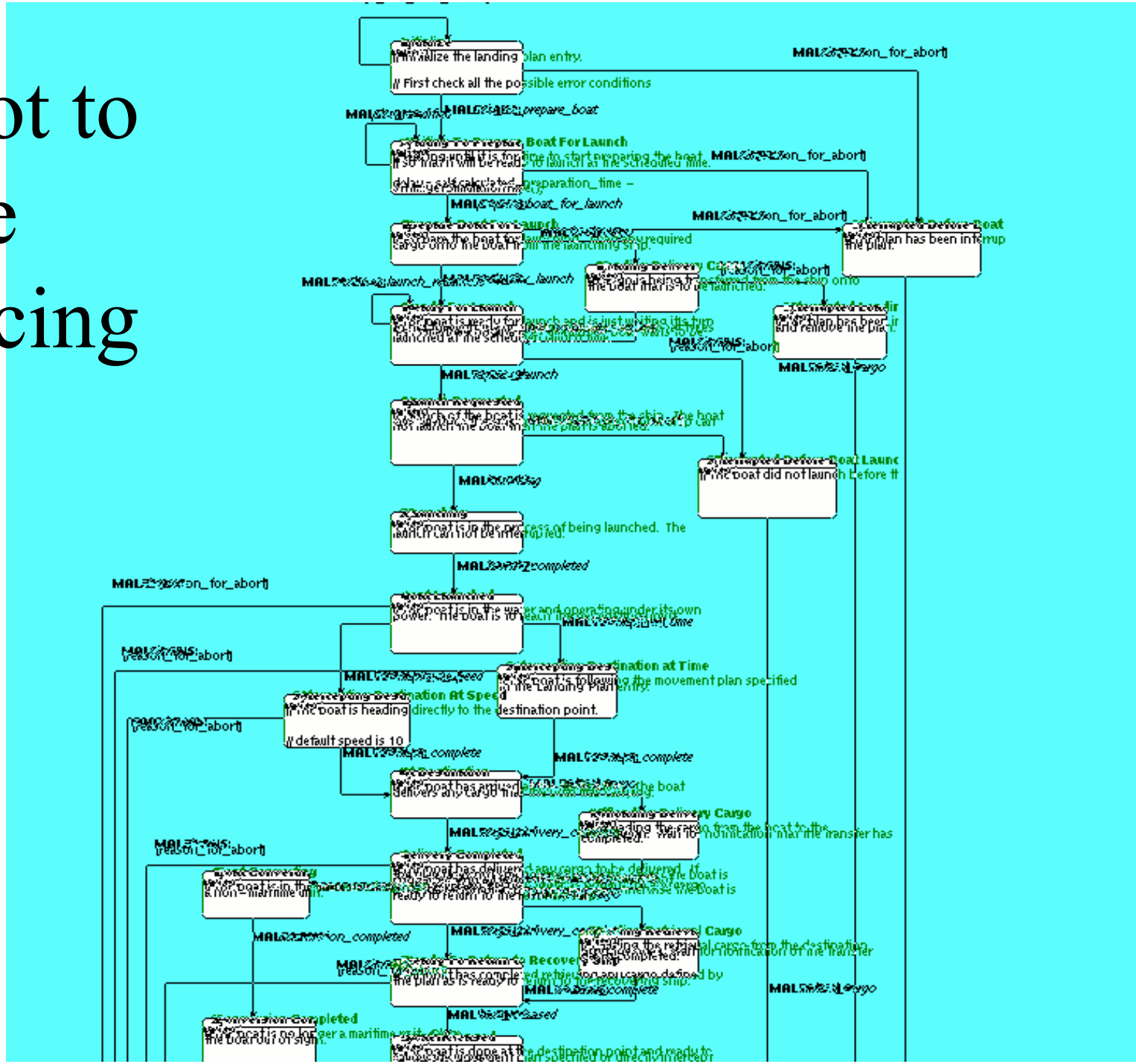
Topics

- **Domain Partitioning**
  - **evolution of the Domain Chart**
- **Modeling issues**
  - **modeling “unavailability”**
  - **synchronicity**
- **Domain bridging**
  - **Explicit / Implicit**
- **Translation effects**
  - **Diagnostics / Debugging**
  - **Deployment**

Domain Partitioning

- **Separating subject matter**
  - **almost everything moves- easy!**
  - **projectile launchers?**
- **Domains vs. subtyping?**
  - **Sensor types: a common detection cycle**
    - **detect - classify - identify - lose**
  - **weapon types target detected tracks**
- **“Discovering’ domains**
  - **Activity sequencing**
  - **Systems availability**

How not to manage sequencing

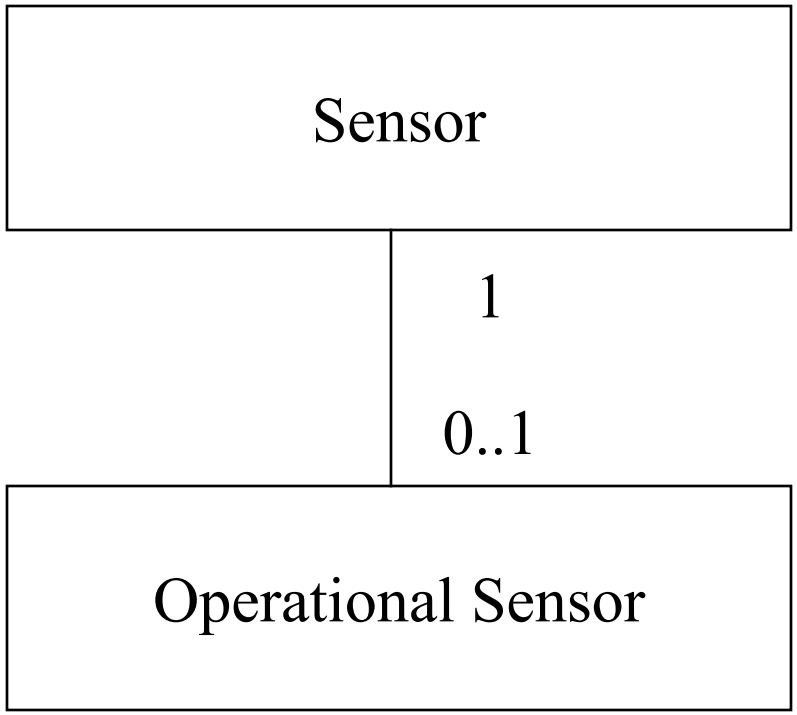


Modeling Unavailability

- **Equipment can fail - at any time**
  - **transition from every state?**
- **Damage can happen - at any time**
  - **transition from every state?**
- **Switch-off can happen - at any time**
  - **…**

Split the state model - across domains!

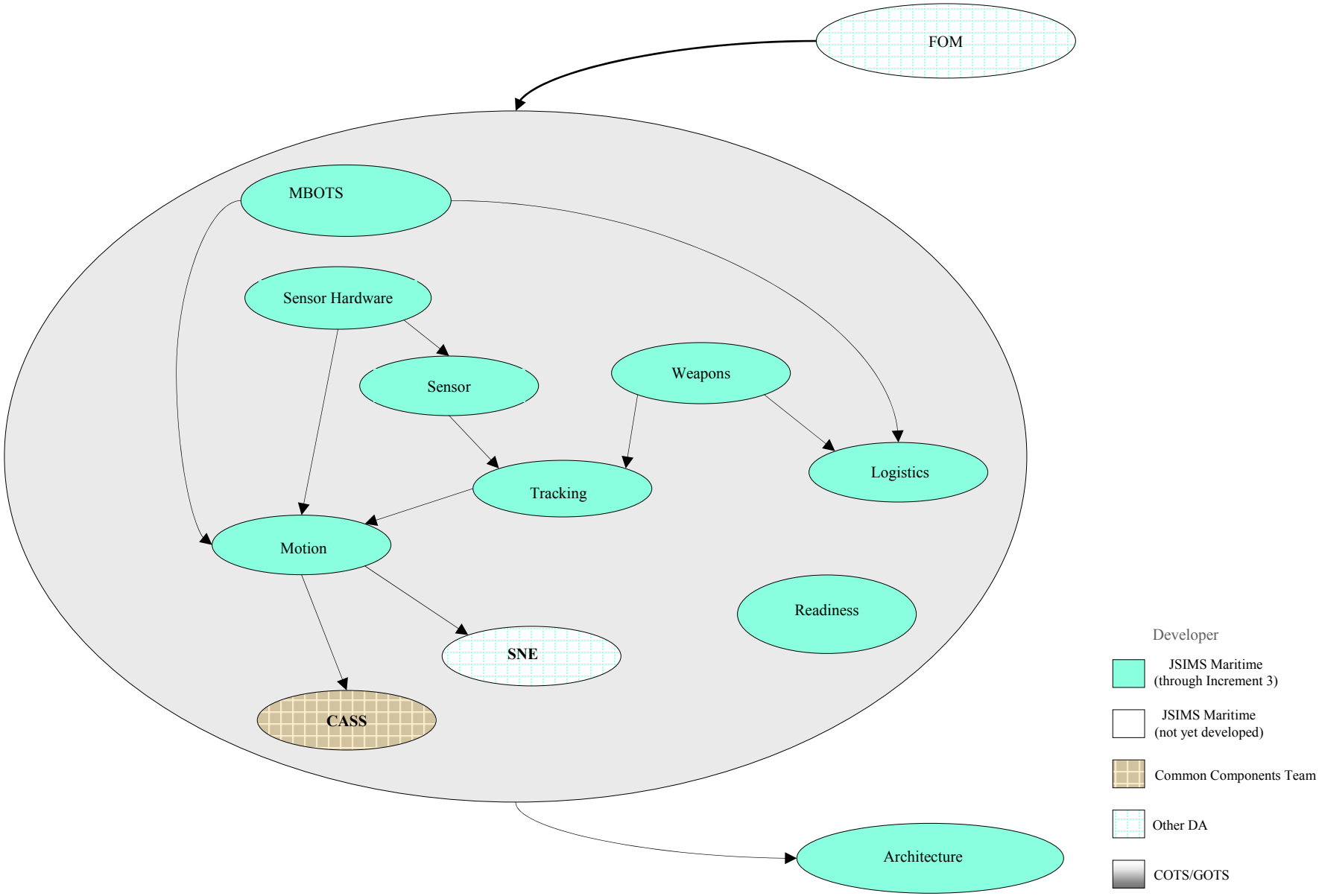
Conditional ‘operational’ object



JSIMS Maritime Domain Chart Incr 7

Draft

30 Nov 2001



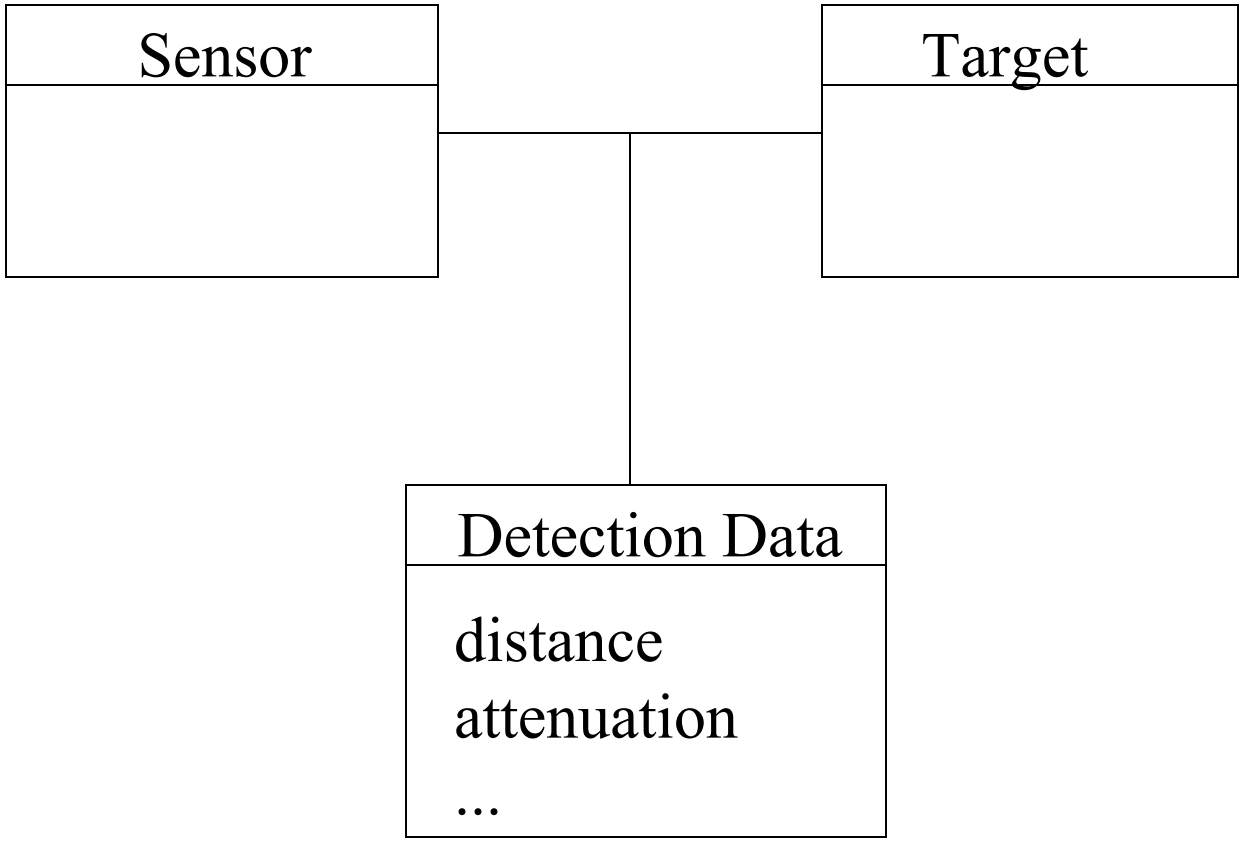
Explicit Bridging

- **External Entities model service domains**
  - **OOA ‘depends’ on service domain API**
- **Bridging invoked by Process actions**
  - **‘buried’ in the code...**
- **Asynchronous services? - ouch!**
  - **Object and domain identifiers in the models**

Implicit Bridging

- **OOA is ‘unaware’ of domain coupling**
- **cause-effect linkage separately specified**
- **bridging code added by translation**
- **Service domain requirements surface at Class Diagram level!**
- **OOA model independent of ‘system’ context**

Surfacing requirements



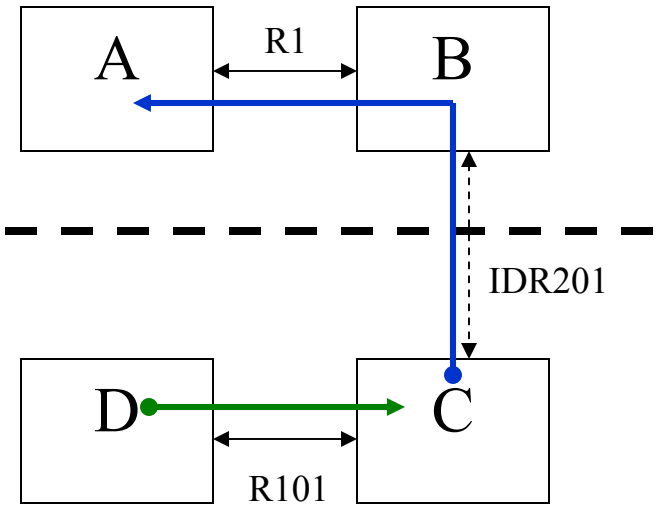
Distance between two moving objects is important!

Realizing Implicit Bridging

- **Extend OOA-of-OOA**
  - **inter-domain relationships**
  - **cause-effect mappings**
- **Populate FOM domain in repository**
  - **uniform bridging for modeled/realized domains**
- **Develop bridge specification - ‘coloring’**
  - **MC-2020 archetype language**
- **Augment translation architecture**
  - **that’s why we have architects….**

Inter-Domain Cause-Effects

Effect Side of Bridge (e.g., Do Event Generate)



Server.ifc

```
cpo->B[IDR201]->A[R1]
```

Client.brg

```
self->C[R101]
```

Causal Side of Bridge (e.g., On State Entry)

Bridge specification

.function ObjectIsLogisticsLoad

….

.function ObjectIsLogisticsTransferTask

.param TaskCounterpart

.param TaskIdr

.param LoadCounterpart

.param LoadIdr

.//

.invoke effect = Counterpart(TaskCounterpart,” TASK”, TaskIdr)

.invoke load = SelectAcrossIdr(LoadCounterpart, LoadIdr)

.invoke task = SelectAcrossIdr(TaskCounterpart, TaskIdr)

.invoke EventGenerate(task, “TRANSFER”, load)

Bridge invocation

.invoke cause = OnStateEntry(“LPLAN”, “unload”)

.invoke plan = Object(“LPLAN”, cause)

.invoke cargo = Select(plan, “CARGO”, R127)

.invoke ObjectIsLogisticsTransferTask

(plan, “ldrXfer”, cargo, “ldrCargo”)

Implicit Bridging - the early days

- **Counterpart objects - for life!**
  - **Counterpart attributes**
- **‘ad-hoc’ inter-domain navigation**
  - **two-way bridging between domains**
- **FOM - ‘leaf’ bridging only**
  - **‘cut-and-paste’ repetition**

Implicit Bridging - evolution

- **Variety of causes and effects**
  - **12 causes; 12 effects**
  - **counterpart attributes illegal!**
- **Separate Domain Interface specification**
  - **focus on service domain definition**
  - **‘subscription’ services**
- **Bridge invocation specification**
  - **hierarchy of function invocations cluster bridges into ‘packages’ of functionality**

Trouble Spots!

- **effect ordering**
- **cascading effects - synchronicity**
- **‘in-flight’ signals - synchronicity!**

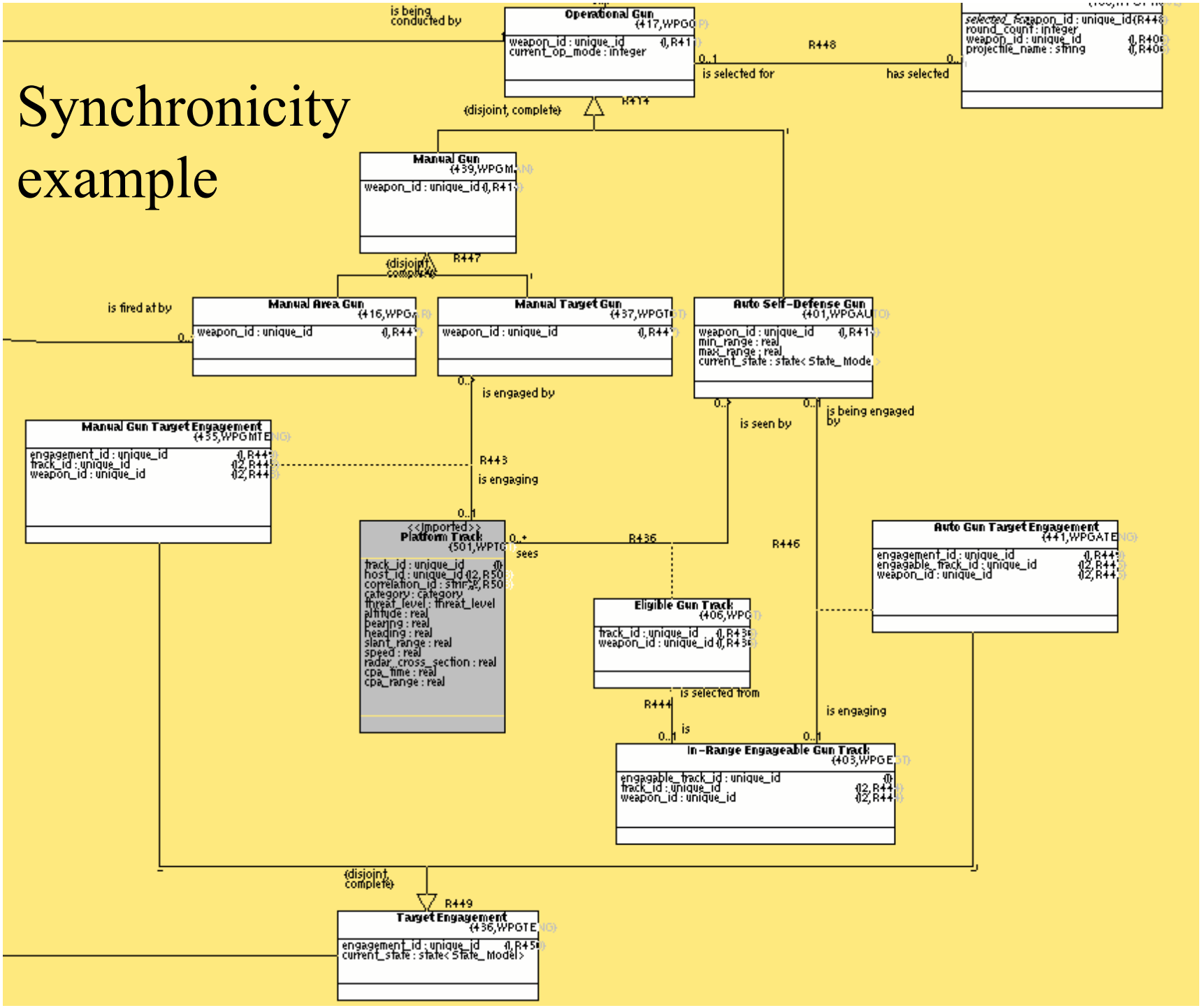
Synchronicity vs. signal exchange

- **‘External events’ may trigger one or more:**
  - **unlinking objects**
  - **object deletion**
  - **life-cycle discard**

  To maintain consistency, these effects must be done ’uninterrupted’; not safe to exchange signals!

  So, significant synchronous processing!

Synchronicity example



‘In-flight’ signal deletion:

- **When an object is deleted, any ‘in-flight’ signals are discarded**
- **The analyst is free to establish a ‘handshake’ protocol to determine what happened**

Translation gives us:

- **Embedded diagnostics**
  - **link reference counting**
  - **pending signal tracking**
- **XML log file output**
  - **selectable trace level**
- **Deployment choices**
  - **mapping to parallel computation units**
    - **coloring special datatypes**
    - **controlling granularity/allocation**

Conclusion

- **Automated Translation is essential!**
  - **independence of modeled domains**
  - **bridging added during translation**
  - **bridging to evolving FOM domain**
- **It works!**
  - **The Navy federate has consistently been a leader in demonstrated functionality at the five integration events held over the last 18 months.**