IMPACT Framework

Mr Andrew Howe (Dstl)

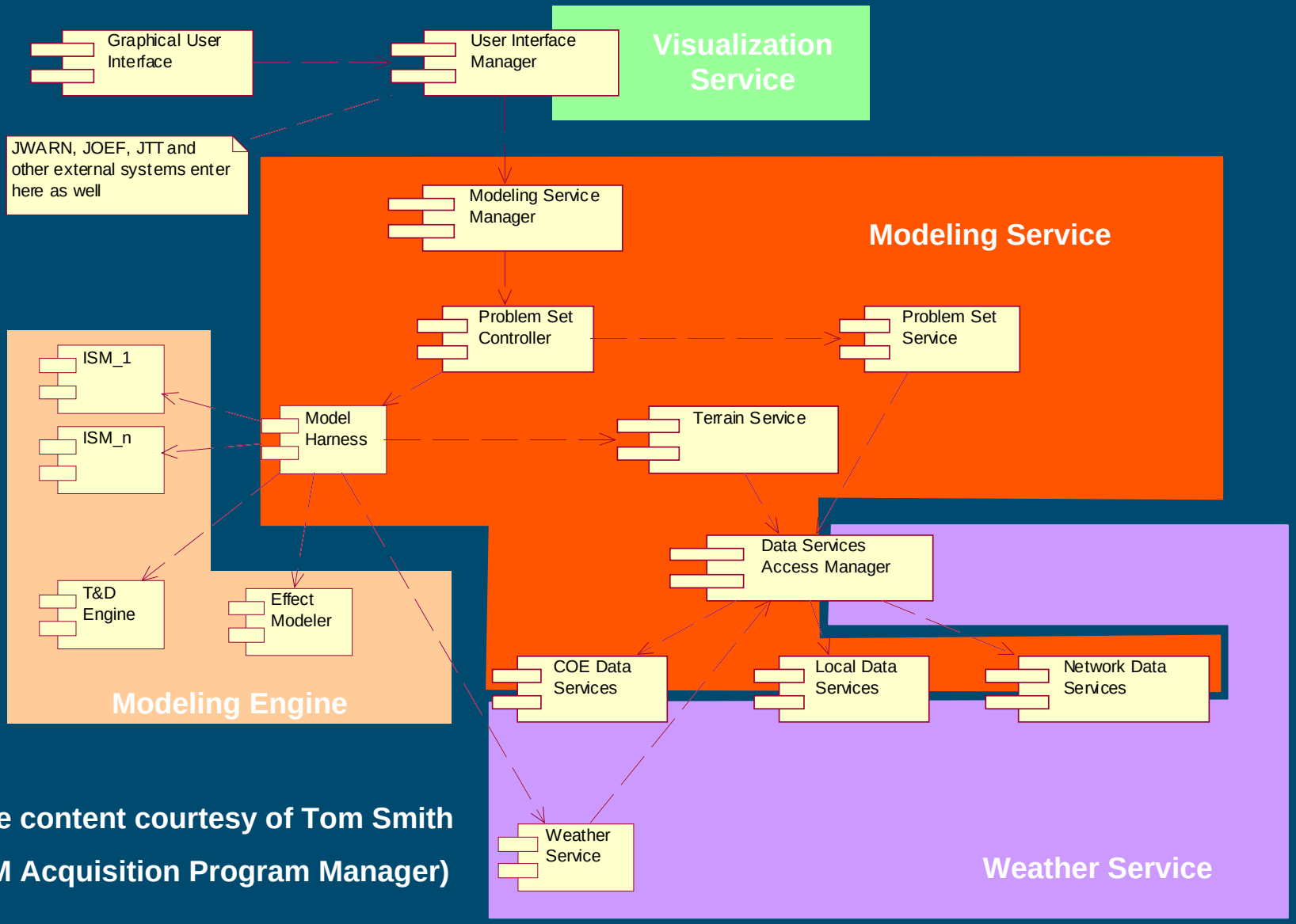
Programme Overview

Work programme initiated September 2006.

"To use the modelling capabilities developed by Dstl as a testing environment for the purposes of identifying best of breed techniques and to allow the exploration of architectural and Service Oriented Architecture (SOA) requirements and limitations that are likely to be encountered by the US Programs of Record (JEM, JOEF and JWARN), including an exploration of distributing the computational burden associated with running simulations, evaluating potential gains and proposing how they could be implemented into the software architectures of the Programs of Record."

Programme Background

- A strategy for data interoperability between the Programs of Record is already in place:
  - DoD Net-Centric Data Strategy.
- Mechanisms of achieving the goals of the strategy are under development:
  - JPM-IS CBRN Data Model.
  - JPM-IS CBRN XML Schema.
- The respective architectures of the Programs of Record are well defined.



Slide content courtesy of Tom Smith

(JEM Acquisition Program Manager)

Focus & Challenge Overview

- “Inside” the Modelling Engine (JEM)
  - Dispersion calculations are computationally demanding:
    - Meteorological conditions.
    - Complex terrain interactions.
    - Building interactions (urban environment).
    - Large domains (e.g. biological release).
  - Hardware resources are limited:
    - Limited processing power.
    - Limited communications bandwidth.

- Are there therefore any advantages / disadvantages to distributing the models that constitute a model run / simulation?

Software Challenges

- Hardware limitations:
  - Application throughput:
    - Memory
    - CPU usage
    - Communications latency
- 32 bit Operating System limitations.
- Interoperability with legacy code bases.
- Reliability / maintainability.
- Scalability.
- Security.
- Others anticipated as the programme progresses…

Dispersion Modelling Solutions

- Use of a Gaussian Puff Model:
  - Distributes particles over a 3D area, which enlarges and “thins out” as the puff moves downwind.
  - Less computationally demanding (memory and CPU) than models that process individual particles.
    - Enables calculations to run in a shorter duration over a larger domain.
- Adaptive grids:
  - Allows for greater puff fidelity in the vicinity of the source, which reduces as the puff moves downwind.

- These are “well known” solutions to challenges faced by the CBRN community (implemented e.g. in SCIPUFF).

Dispersion Modelling Solutions

- Elimination of “Opportunistic Calculations”:
  - CPU cycles are not “wasted” calculating, and then recalculating a result based on data that may have changed.
- Deletion / serialization of “non-output” puff histories.
- Modifiable “puff splitting” (higher resolution, greater computational cost) and “puff merging” (lower resolution, lower computational cost).
- Caching of coordinate conversions.

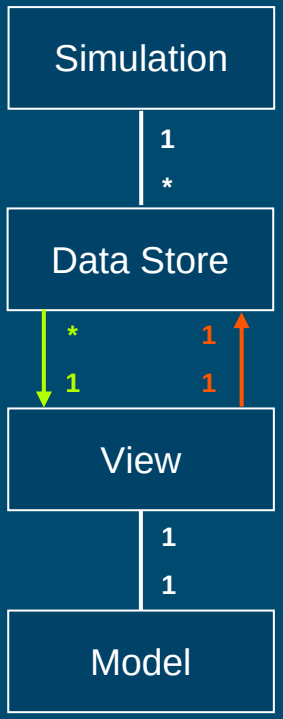
“Test Bed” Architecture

- A framework for the integration of separate models into a single coherent simulation.
  - Orchestrates the flow of data between models i.e. the system is “aware” of the dependencies between the models that constitute a simulation.
  - Decouples models by design (i.e. the models communicate with the system, not with each other, meaning that each model requires knowledge its own of data, but not how, or from where, it was created.)
  - Data streams or files can be used to replace models.

- Provides a “sand box” where concepts may be rapidly prototyped.

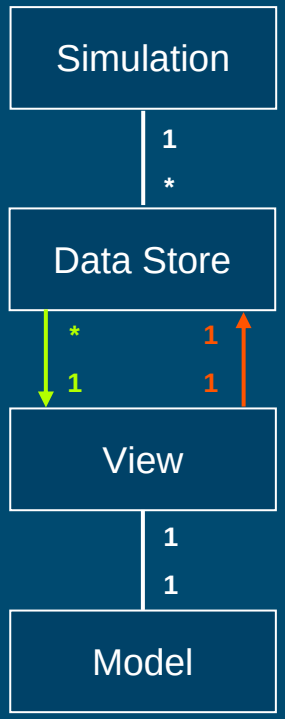
Model Concepts

- A model is essentially an “atomic” process (i.e. consumes data, processes it, and generates a result).
- Each model is controlled by the system (i.e. instructed to progress to a point in simulation time).
- Each model has a limited view of the environment in which it is executing.



View Concepts

- A view provides an interface to a shared set of data stores that contain all of the data used by a simulation:
  - get(entity, attribute, time) : value
  - set(entity, attribute, time, value) : void
- Each view has access to zero or more read only data stores.
- Each view has access to a single writable data store.

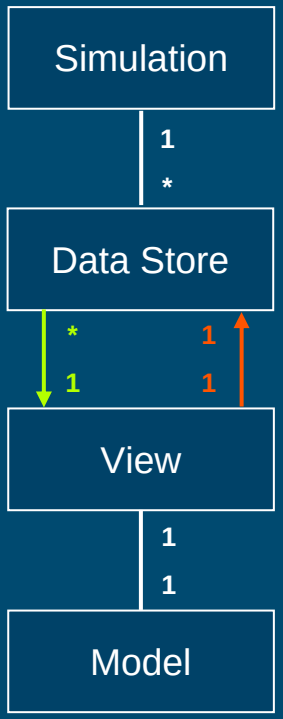


Data Store Concepts

- All data is stored in the form of discrete entities, each of which has at least one attribute.
- Each attribute has a time series of values stored in the data stores as a mapping, such that they may be referenced by the key:

  entity + attribute + time → value

- Once written to a data store, data is immutable.



Concurrency

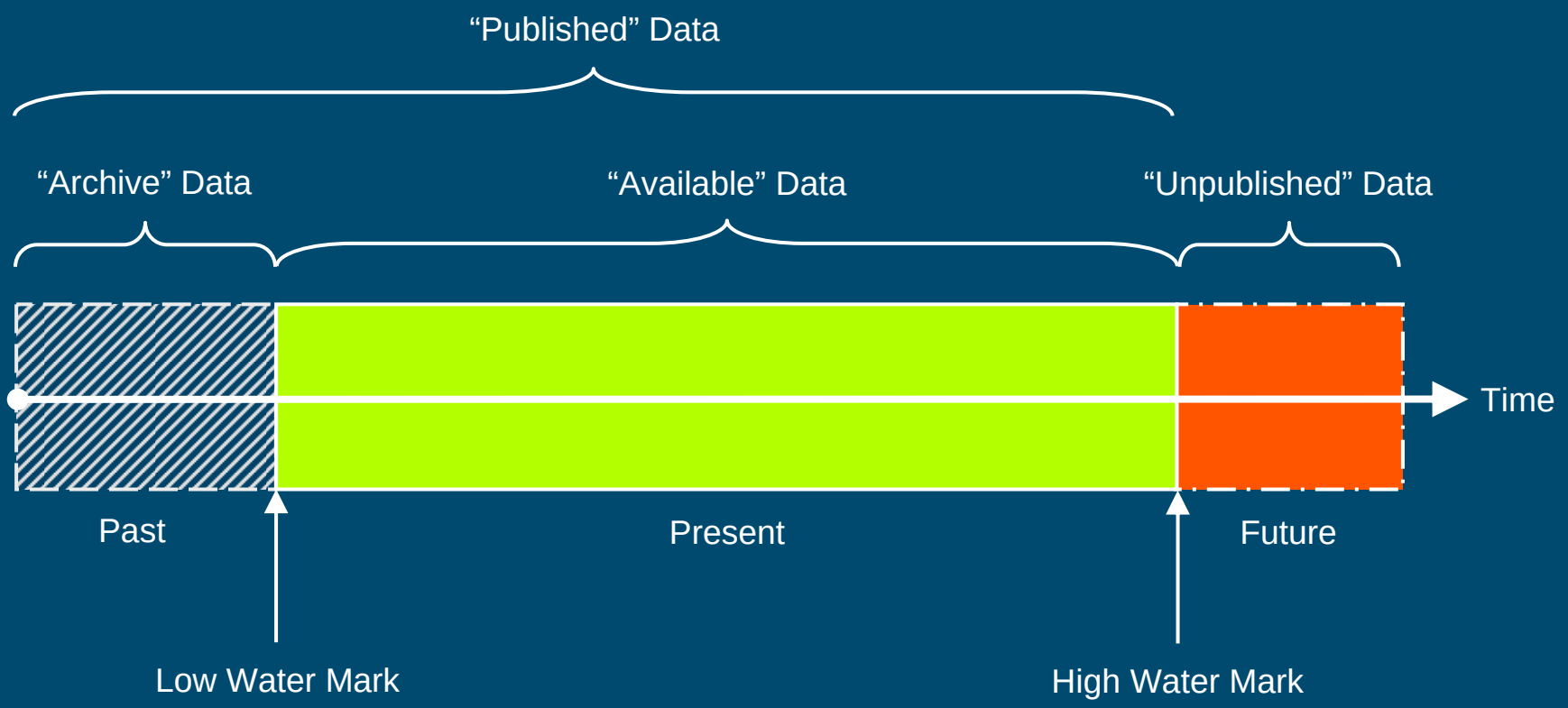
- When distributed, the system may be represented as a classical Readers & Writers problem:
  - Shared resources (Data Stores), each of which is accessed by *N* “reader” processes and a “writer” process (models (via views)).

- “Safety” Property
  - Writers *must* have exclusive access to the shared resource.
  - Many readers may access the shared resource when no writers are active.
- “Liveness” Property
  - Reader and writer processes *must* be able to gain access to the shared resource eventually.

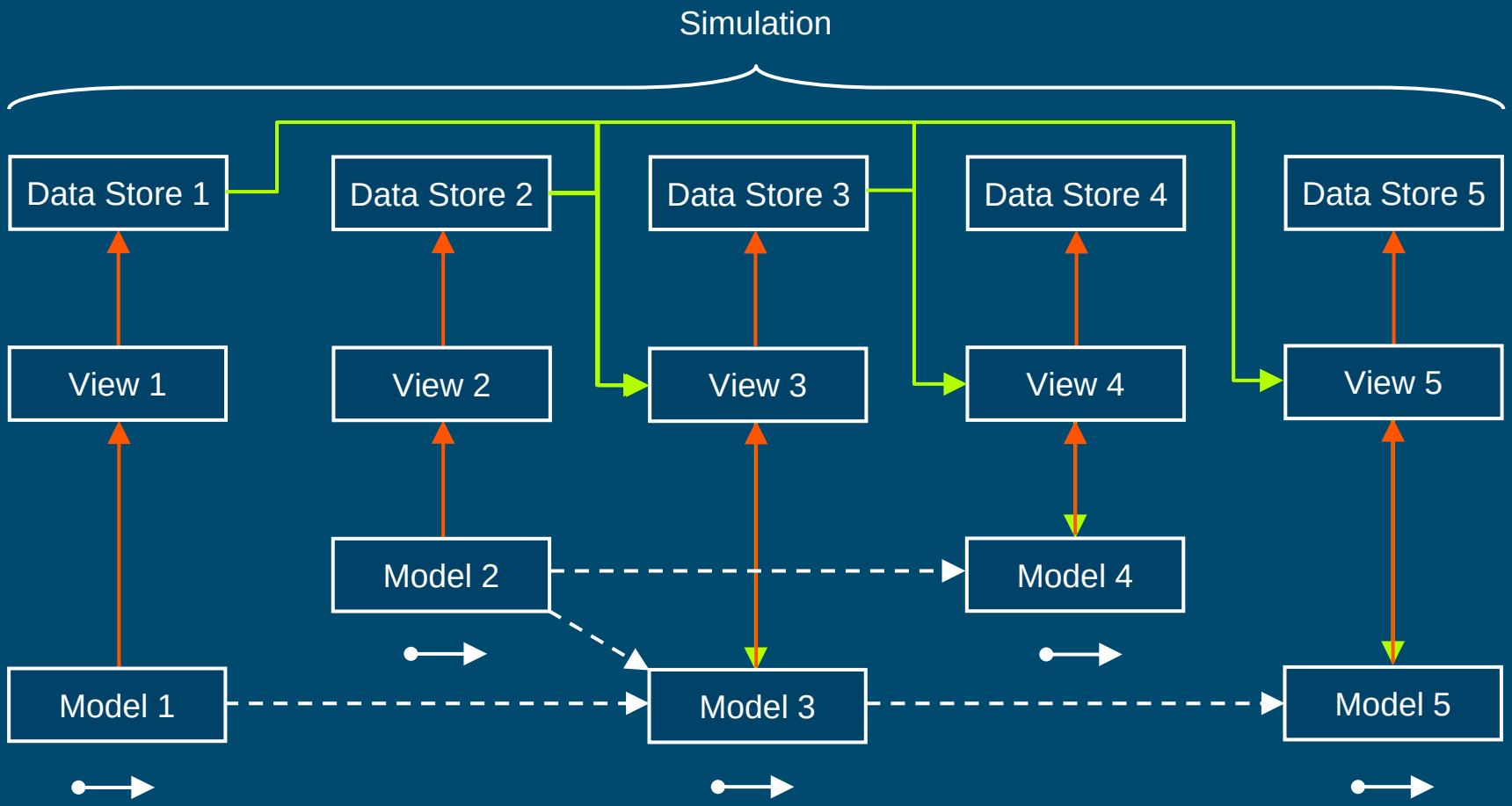
Concurrency

- Traditional solutions to the Readers & Writers problem involve implementing the Data Stores as “monitor” objects:
  - Readers Priority.
  - Writers Priority.
  - Fair.
- These result in a performance bottleneck located around the Data Stores.

Data Store Detail

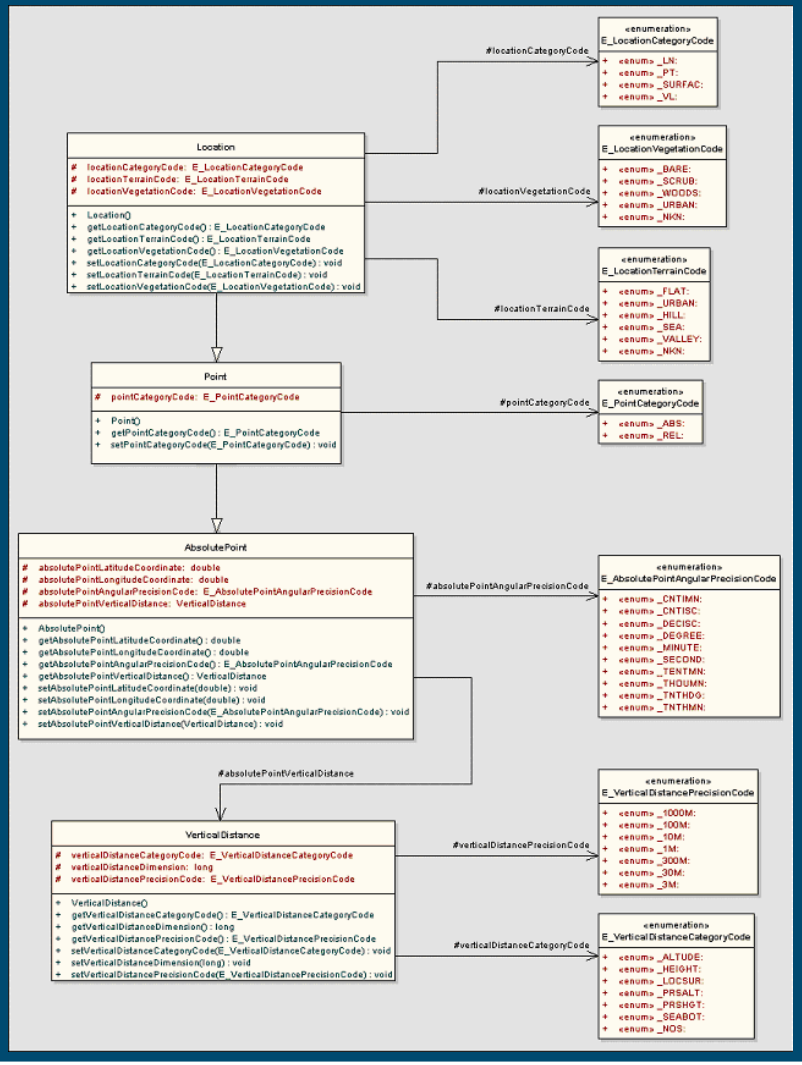


Simulation Overview



CBRN Data Model Considerations

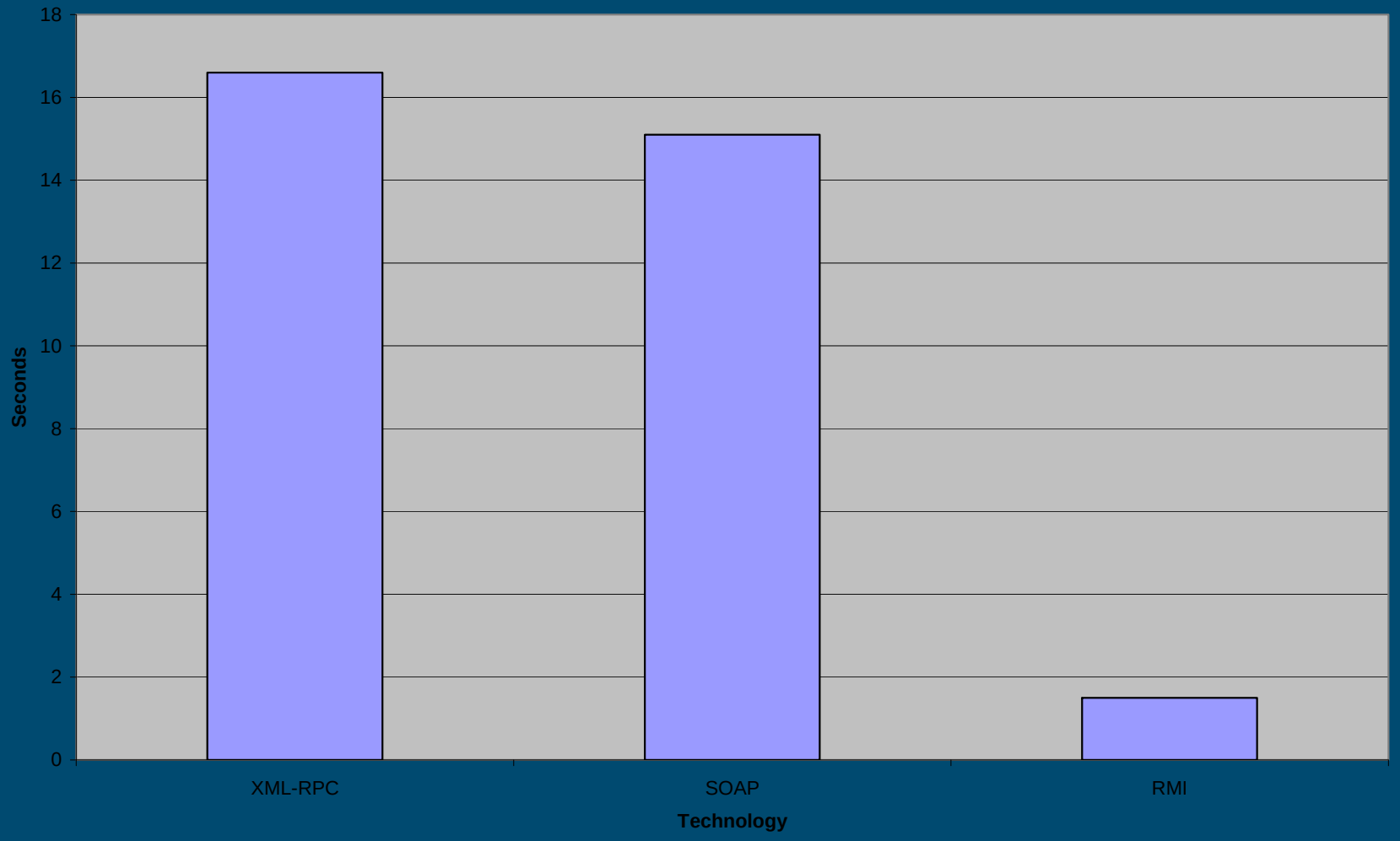
- A prototypical Java implementation of the CBRN Data Model has been developed as part of an internal Dstl effort.
- “Location” (no “MetaData” shown).



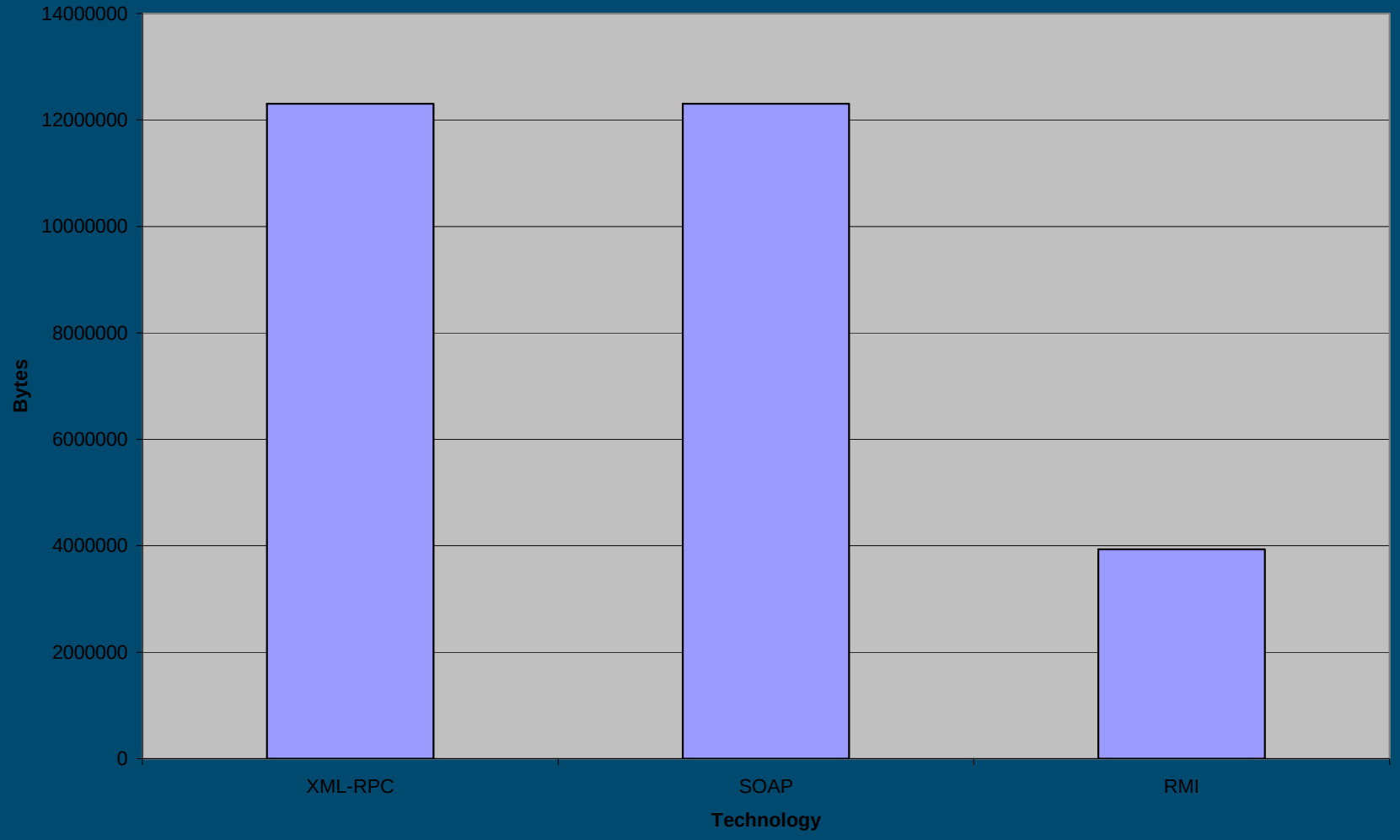
Observations

- Comparison of XML-based protocols (SOAP, XML-RPC) and Java RMI for distributed model → model interactions (NB not system → system!) performed.
- Measurements included:
  - Time (seconds) to send 100,000 “Location” objects from one process to another and back.
  - Volume of data (bytes) to send 100,000 “Location” objects from one process to another and back.
  - Percentage CPU usage (one end only) to send 100,000 “Location” objects from one process to another and back.

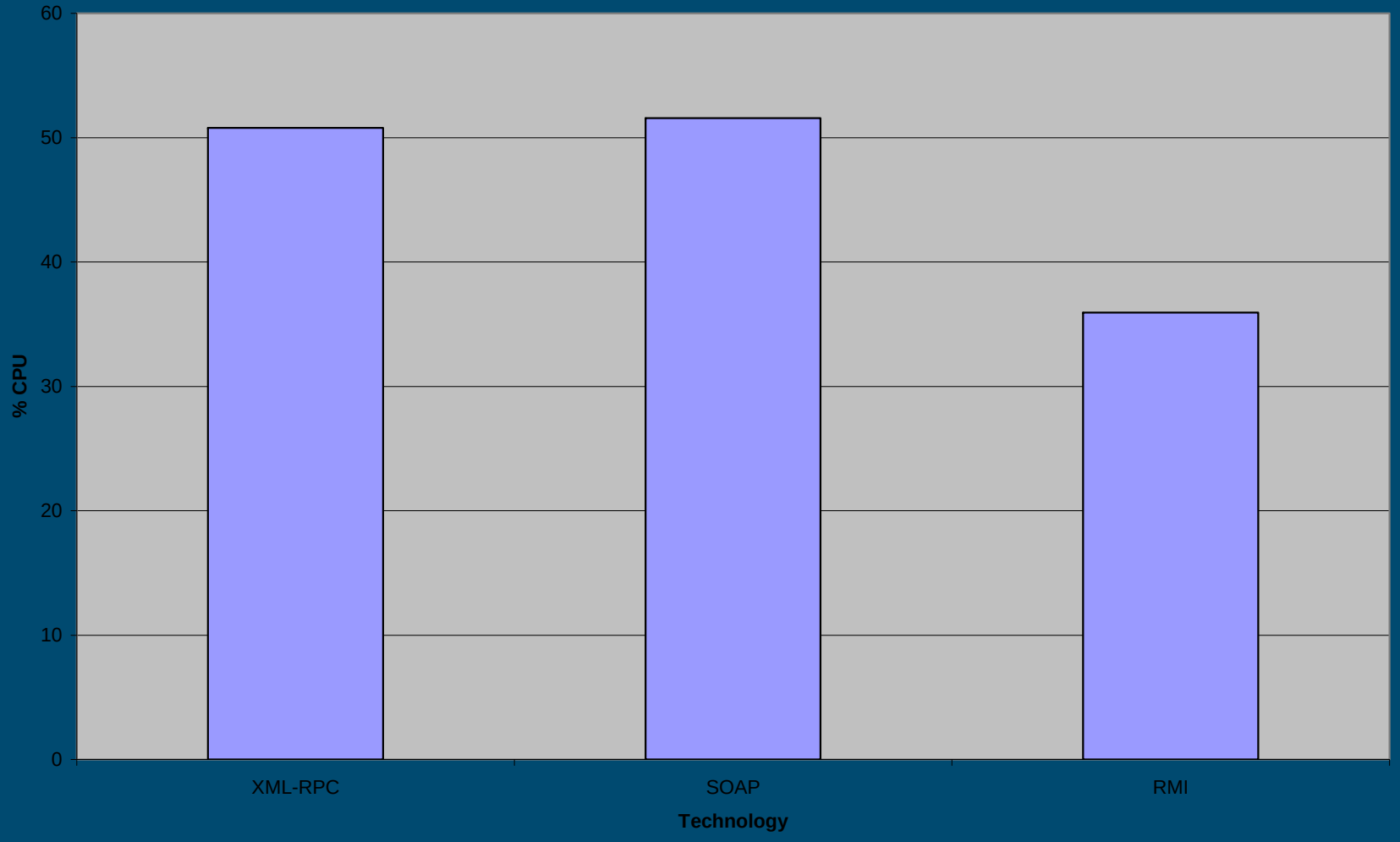
Observations (Time)



Observations (Volume of Data)



Observations (% CPU)



Future Work

- Identification and development of a necessary and sufficient set of CBRN Data Model-compliant web services that can interact with the system previously outlined.
- Examination of future modelling requirements with respect to identifying opportunities for distribution and concurrent execution of individual models:
  - JEM
  - JOEF
  - JWARN

Questions?