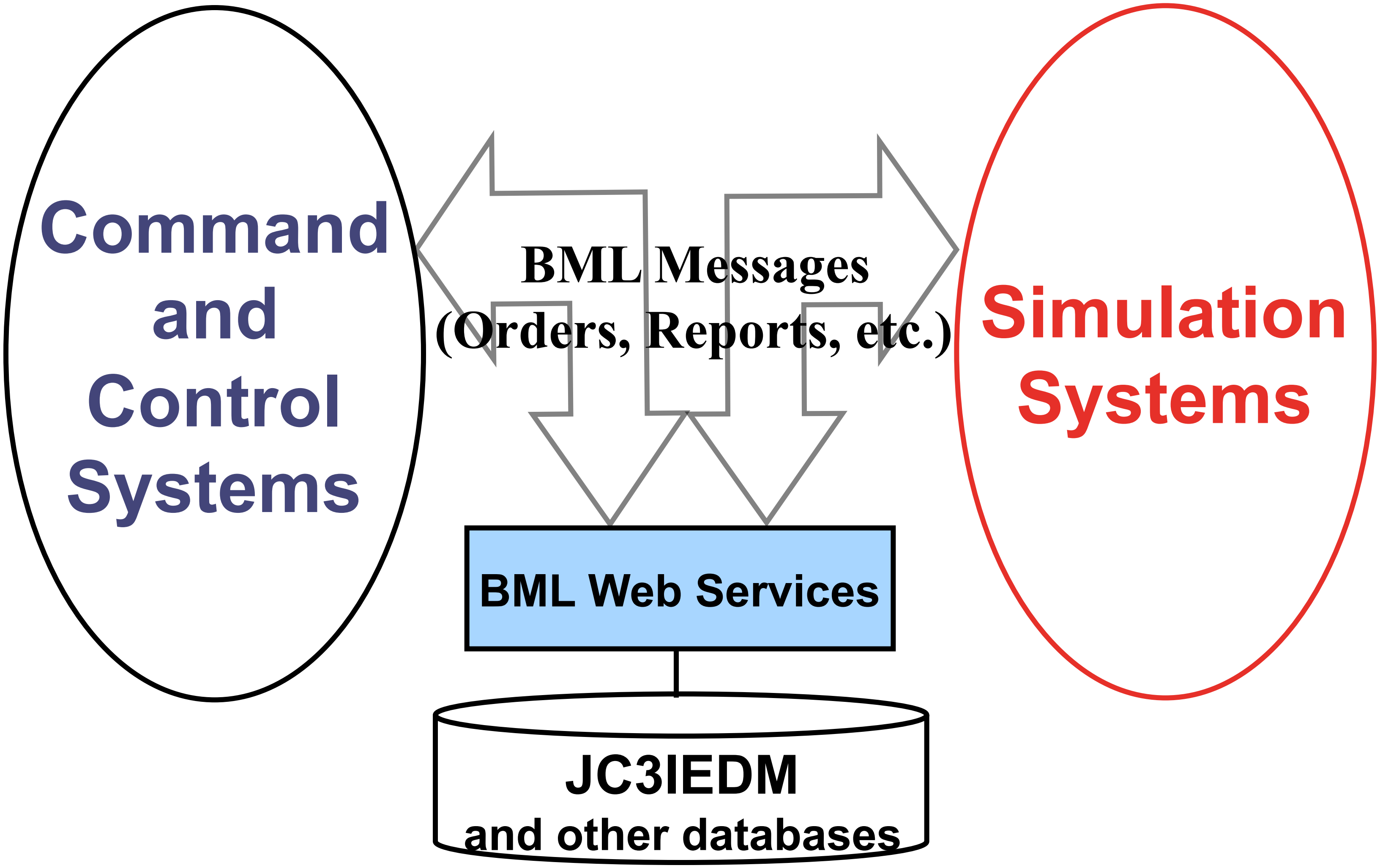
Creating a BML Software Infrastructure

Dr. J. Mark Pullen
Director, C4I Center
George Mason University
mpullen@c4i.gmu.edu

http://c4i.gmu.edu/BML

BML Architecture


Command and Control Systems
BML Messages (Orders, Reports, etc.)
Simulation Systems
BML Web Services
JC3IEDM
and other databases

Overview of Presentation

- Essential elements for BML
- The IBML Schema
- Developing the Scripted BML Server
- Implementing publish/subscribe
- Other software essential for distributed development

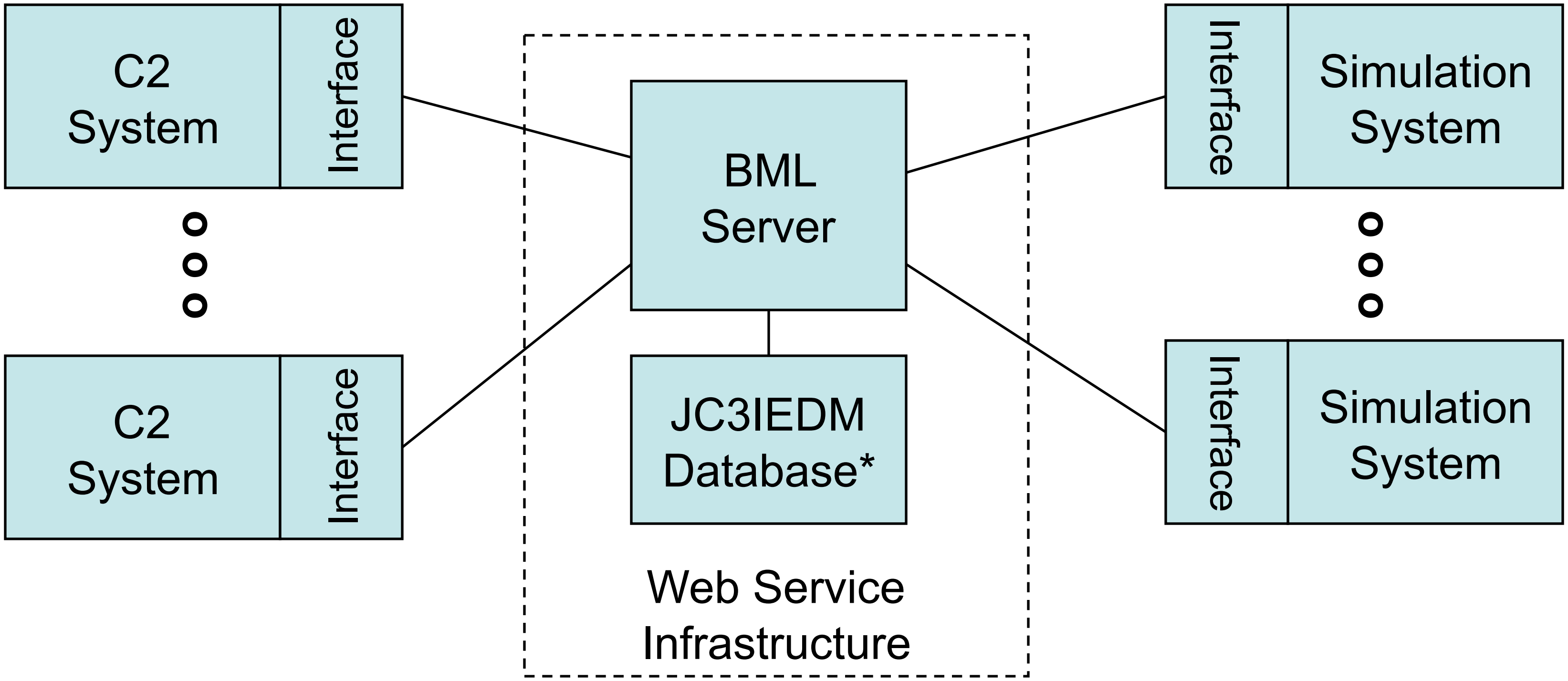
Essential Elements for BML

- C2 Systems
- Simulation systems
- BML schema
- BML server
- Network infrastructure

Terminology

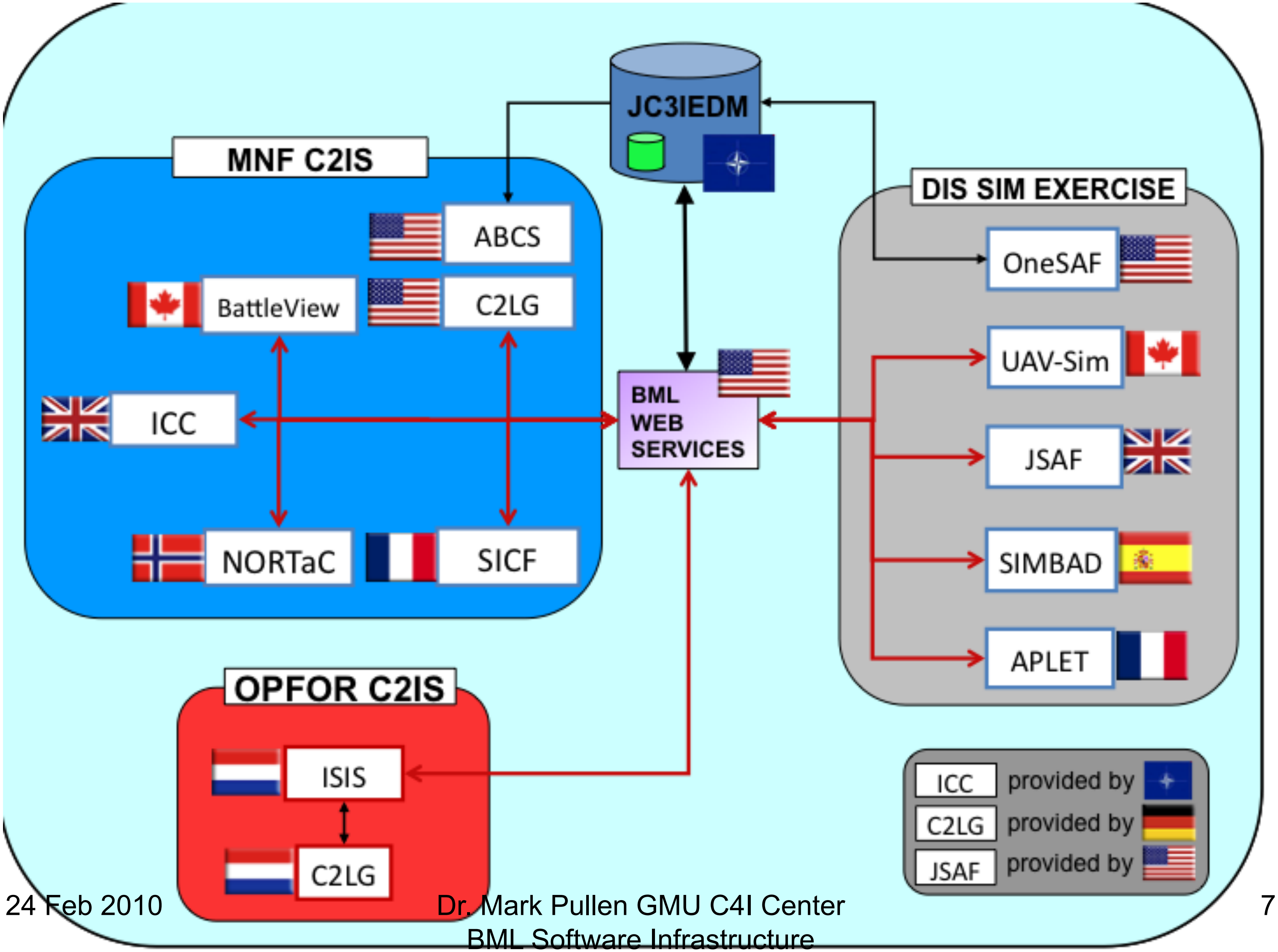
- What do you call the system of systems that interoperate via BML?
- HLA calls their assembly a “federation”
  - Implies the systems are like member states
- For BML I propose “coalition”
  - Reflects the military application
  - And the relationship among systems
  - They come together voluntarily, for a specific purpose, under mutually agreed leadership
- I will use that terminology in this talk

Basic BML Coalition



* database could be distributed via replication, allowing a coalition to employ multiple such clusters

MSG-048 Architecture



C2 and Simulation Systems

- These are the main focus of activity
  - Others will speak about them at length
- Important characteristics from infrastructure view
  - Must be possible to interface them to comply with BML schema and selected network protocols
  - Must be possible to control them in coalition context
  - C2 systems must be able to accept reports at system designated rate
  - Simulation systems but be able to throttle back to system designated reporting rate
    - A particular problem if faster than real time

BML Schema

- Essential to define information that can be exchanged
  - Expressed as XML Schema Definition (XSD)
  - Establishes structure of information
    - Basically, a tree – like any XML
  - And “namespace” of tags that identify it
- Very important to have this defined first
  - The server and all C2/simulation interfaces must implement it
  - Late binding of schema nearly derailed MSG-048

Layers in BML

BML as implemented in MSG-048 has three layers:

Top: Language layer: primary exposed service which implements tagset for operational functions (tasking, reporting, etc.)

- JBML showed how to structure this using C2 Lexical Grammar for unambiguous parsing

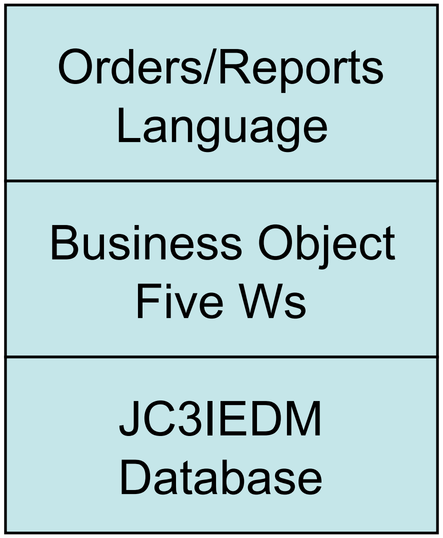
Middle: Business object layer: internal building blocks at the level of “who/what/when/where/why”

- SISO is working on a standard for this

Bottom: Database layer: JC3IEDM

- Standardized by MIP

BML Layers



Orders/Reports Language

Business Object Five Ws

JC3IEDM Database

BML Schema for *Task*

(top layer)

```
<xs:complexType name="GroundTaskType">
  <xs:sequence>
    <xs:element name="TaskeeWho“ type="WhoType”/>
    <xs:element name="What" type="WhatType“/>
    <xs:element name="Where" type="WhereType“/>
    <xs:element name="StartWhen" type="WhenType“/>
    <xs:element name="EndWhen" type="WhenType" minOccurs="0“/>
    <xs:element name="Affected" type="AffectedType" minOccurs="0“/>
    <xs:element name="Why" type="WhyType" minOccurs="0“/>
    <xs:element name="TaskControlMeasures“
      type="TaskControlMeasuresType" minOccurs="0“/>
    <xs:element name="TaskLabel" type="LabelType"/>
  </xs:sequence>
</xsd:complexType>
```

BML Schema for *WhoType*

(middle layer)

```
<xs:complexType name="WhoType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="UnitID“ type="jc3iedm:OIDType"/>
      <xs:element name="NameText“
        type="jc3iedm:Text100XmlType"/>
    </xs:choice>
    <xs:element name="Equipment“
      type="jc3iedm:Text100XmlType
      maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

The IBML Schema

- By 2008 there were multiple flavors of BML
- US Army Geospatial Center (AGC) sponsored project produce a consolidated schema
  - Combined best of JBML, geoBML, and Army OPORD
  - MSG-048 used and refined IBML
- US Army G6 sponsored an architecture effort that further refined IBML (with MSG-048 inputs)
  - Results documented in detail
  - Available on GMU C4I Center website
  - Includes a mechanism for schema maintenance
  - When SISO standardizes a Business Object layer, the Army architecture could be harmonized with it
  - SISO plans call for standardizing top layer later
  - IBML is available for use now
- IBML OPORD gives a good start on NATO OPORD schema

Role of the BML Server

- Conceptually the server plays a very simple role
  - Accept BML Orders and Reports as XML documents pushed by clients
  - Reproduce those documents on demand, based on OrderID or ReportID
  - Provide supporting services such as initialization and summarizing status
    - For example, all ReportID in a given time window
- Issues for server
  - Performance
  - Polling model versus publish/subscribe
  - Auxiliary functions such as logging

About BML Server Performance

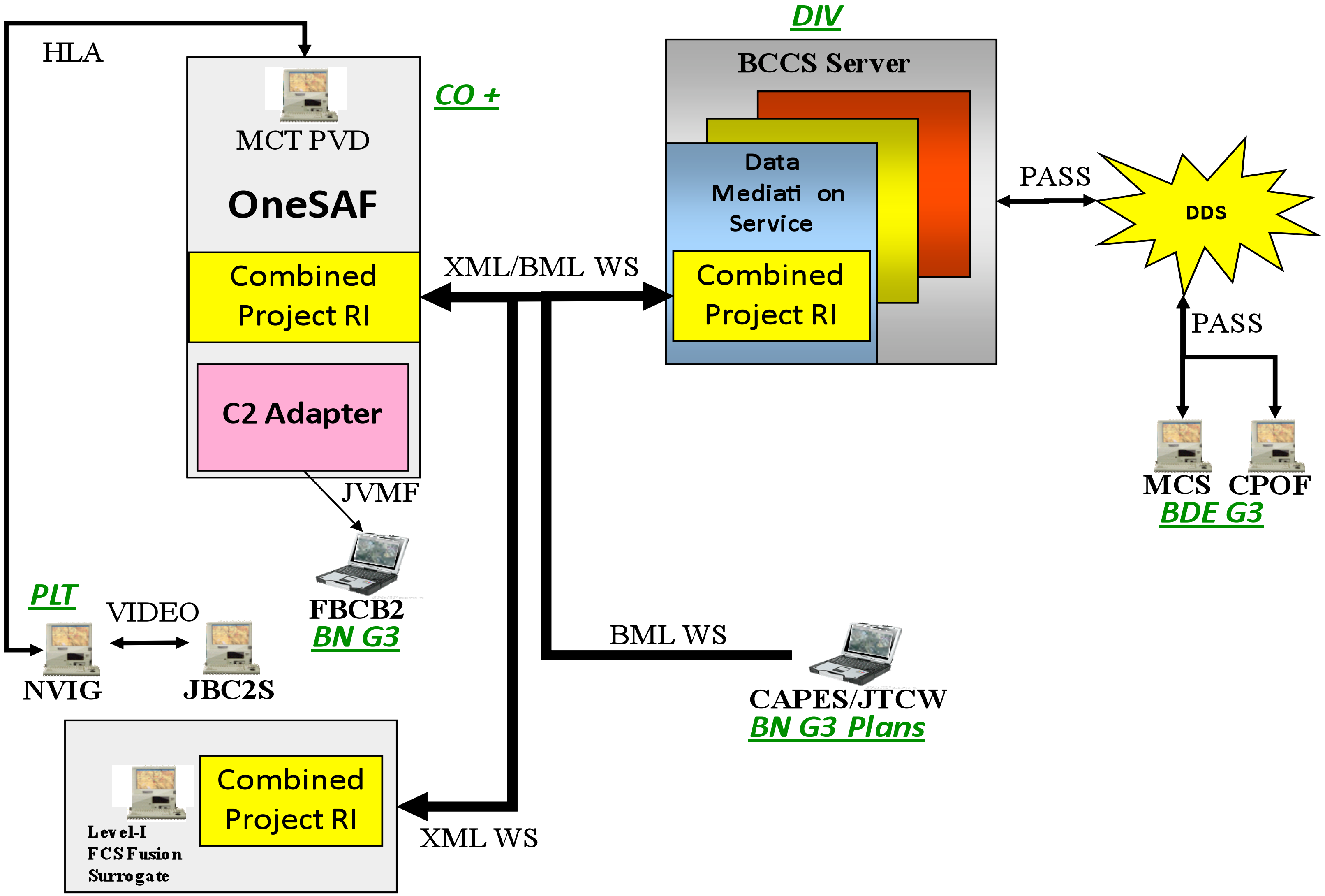
- BML Clients will always be able to overwhelm even the most powerful server
  - There are multiple clients and they can work from memory rather than database
- The appropriate question is whether the server is able to keep up with a realistic rate of orders and reports
- MSG-048 SME advised that a one minute interval would be realistic
  - After all clients implemented this worked well
  - Server used never had observable backlog
- Larger simulations will stress servers more
  - Replicating server could help but in the end the database is the bottleneck
  - Might need to invest in more powerful database
  - Some limitation on Report rate always will be necessary

Scripted BML Server

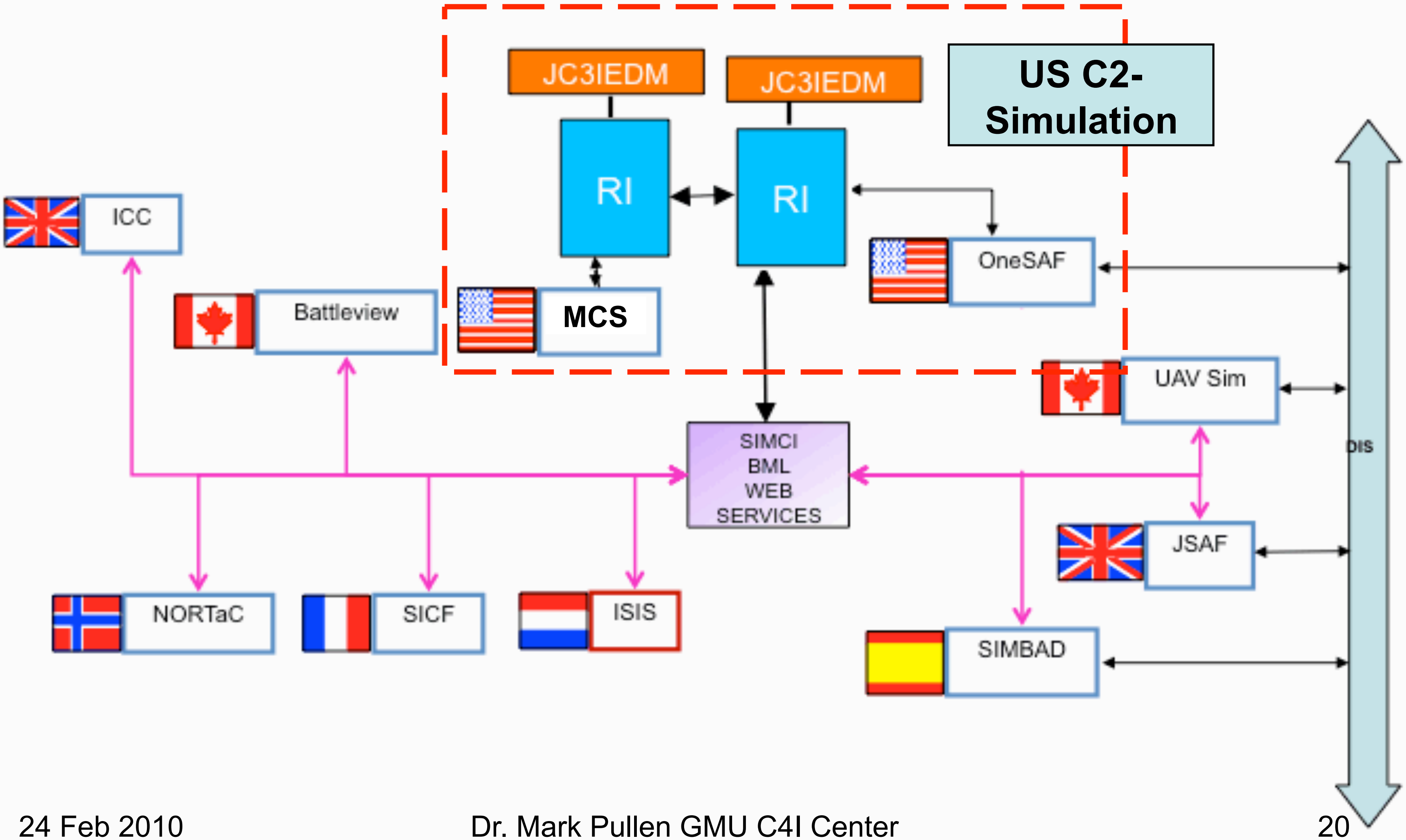
Why Scripted BML Server

- BML continues to grow and evolve
- JC3IEDM changes too, if more slowly
- GMU C4I Center provided BML server for JBML which was used by MSG-048 2007
- Hard coding the BML made server development a source of delay
    - And the server itself a source of bugs
- Needed a way to adapt to change faster
    - With fewer errors and easier to fix
    - SIMCI Combined Project 2008 supported a solution

BML in SIMCI Combined Project 2008



BML System Used By NATO MSG-048 for 2009 Experimentation



Technical Advantages of Scripted Approach

- Middleware functions don't change
  - Mapping BML to JC3IEDM and push/pull to database
  - Program these once and get them right
- Interpreted WS offers flexibility
  - Rapid implementation of new BML constructs
  - Easy to modify underlying data model
    - JC3IEDM also continues to change
  - Reduces time and cost for prototyping
  - Scripting language provides a concise definition of BML-to-data model mappings
  - Although bugs still happen, the number of possible mistakes is far smaller
- Scripted operation may, however, be slower
  - We're multithreading the server which will help this

Scripting Example: TaskeeWho

BML Input

```
<!-- Fragment of <OrderPush> -->
  <Task>
    <GroundTask>
        <TaskeeWho>
                <UnitID>UIE9 FA</UnitID>
        </TaskeeWho>
        . . .
```

Previous Approach: IDEF1x Mapping Definition

JBML mapping to JC3IEDM

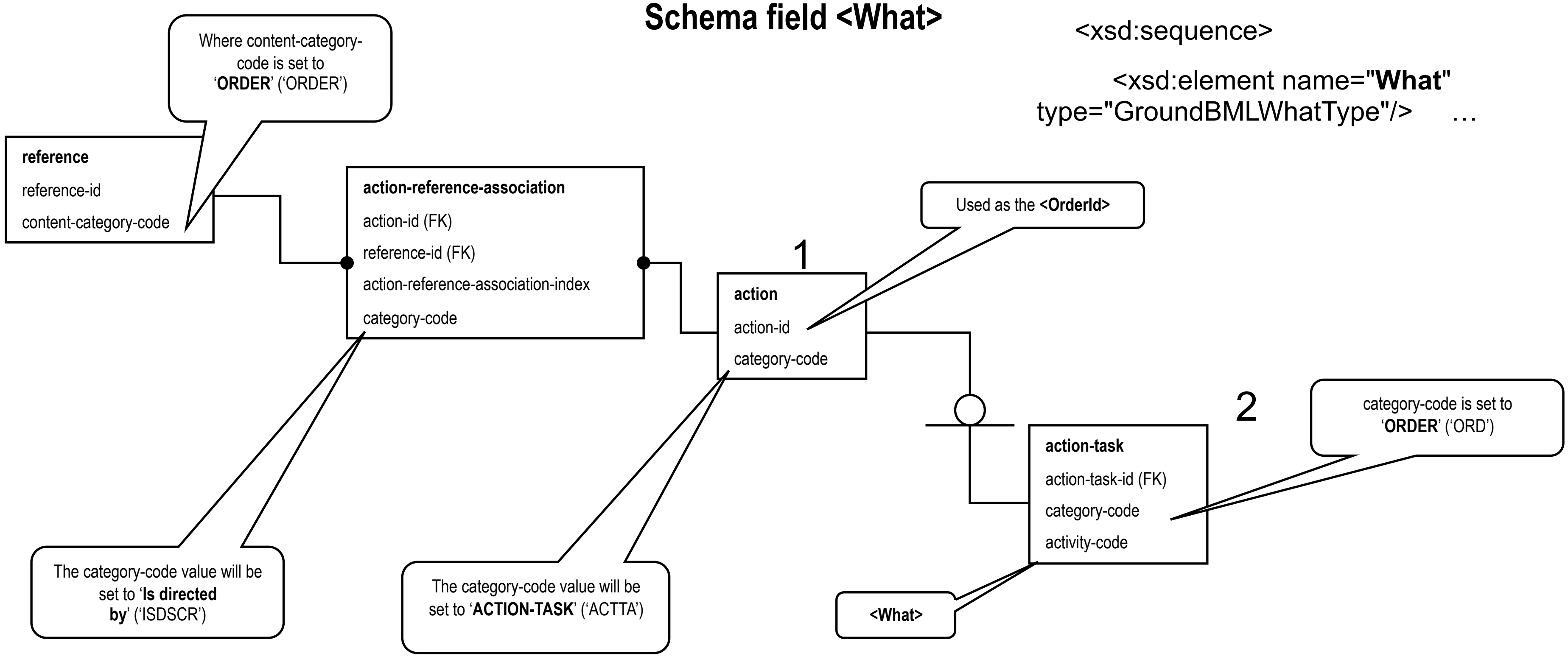
Schema field <What>

Schema Reference:

<xsd:complexType name="CommandType">

<xsd:sequence>

<xsd:element name="**What**" type="GroundBMLWhatType"/> …



Where content-category-code is set to '**ORDER**' ('ORDER')

reference
reference-id
content-category-code

action-reference-association
action-id (FK)
reference-id (FK)
action-reference-association-index
category-code

Used as the **<OrderId>**

1

action
action-id
category-code

2

action-task
action-task-id (FK)
category-code
activity-code

category-code is set to '**ORDER**' ('ORD')

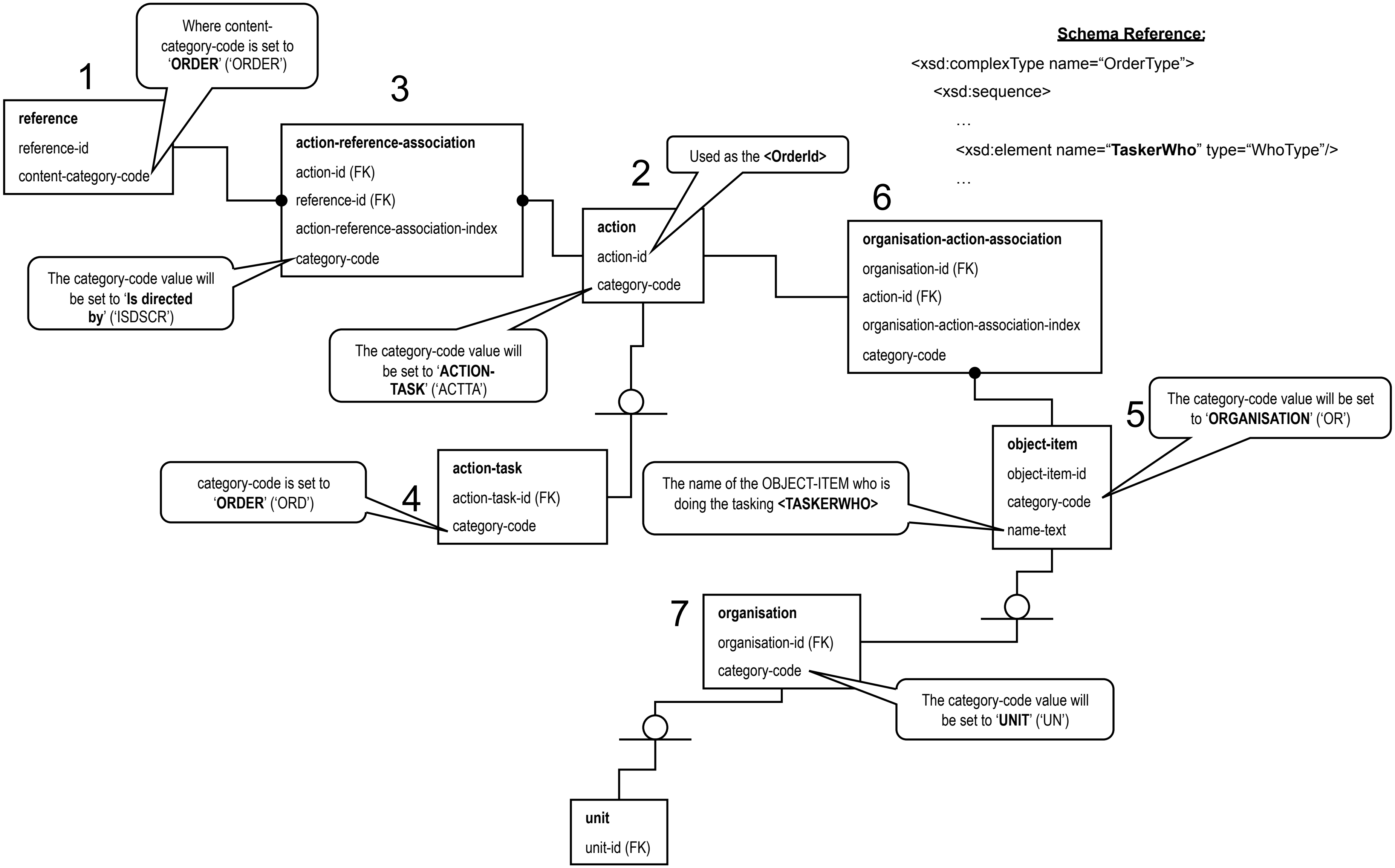
The category-code value will be set to '**Is directed by**' ('ISDSCR')

The category-code value will be set to '**ACTION-TASK**' ('ACTTA')

<What>

Not machine readable though highly structured
Script is a concise XML coding of this

IDEF1x Mapping Definition for *TaskerWho*



1
reference
reference-id
content-category-code
Where content-category-code is set to 'ORDER' ('ORDER')
3
action-reference-association
action-id (FK)
reference-id (FK)
action-reference-association-index
category-code
The category-code value will be set to 'Is directed by' ('ISDSCR')
2
Used as the <OrderId>
action
action-id
category-code
The category-code value will be set to 'ACTION-TASK' ('ACTTA')
6
organisation-action-association
organisation-id (FK)
action-id (FK)
organisation-action-association-index
category-code
5
The category-code value will be set to 'ORGANISATION' ('OR')
object-item
object-item-id
category-code
name-text
The name of the OBJECT-ITEM who is doing the tasking <TASKERWHO>
4
category-code is set to 'ORDER' ('ORD')
action-task
action-task-id (FK)
category-code
7
organisation
organisation-id (FK)
category-code
The category-code value will be set to 'UNIT' ('UN')
unit
unit-id (FK)

Schema Reference:

```
<xsd:complexType name="OrderType">
    <xsd:sequence>
      …
      <xsd:element name="TaskerWho" type="WhoType"/>
      …
```

XML Script for SBML

```
<BusinessObjectTransaction>
   <transactionName>TaskeeWhoPush</transactionName>
   <parameter>task_act_id</parameter>
   <tableQuery>
      <databaseTable>unit</databaseTable>
      <queryAction>GET</queryAction>
      <resultName>unit_id</resultName>
      <columnReference>
         <columnName>formal_abbrd_name_txt</columnName>
         <businessObjectTag>UnitID</businessObjectTag>
      </columnReference>
   </tableQuery>
   <tableQuery>
         <databaseTable>act_res</databaseTable>
         <queryAction>PUT</queryAction>
         <columnReference>
            <columnName>act_id</columnName>
```

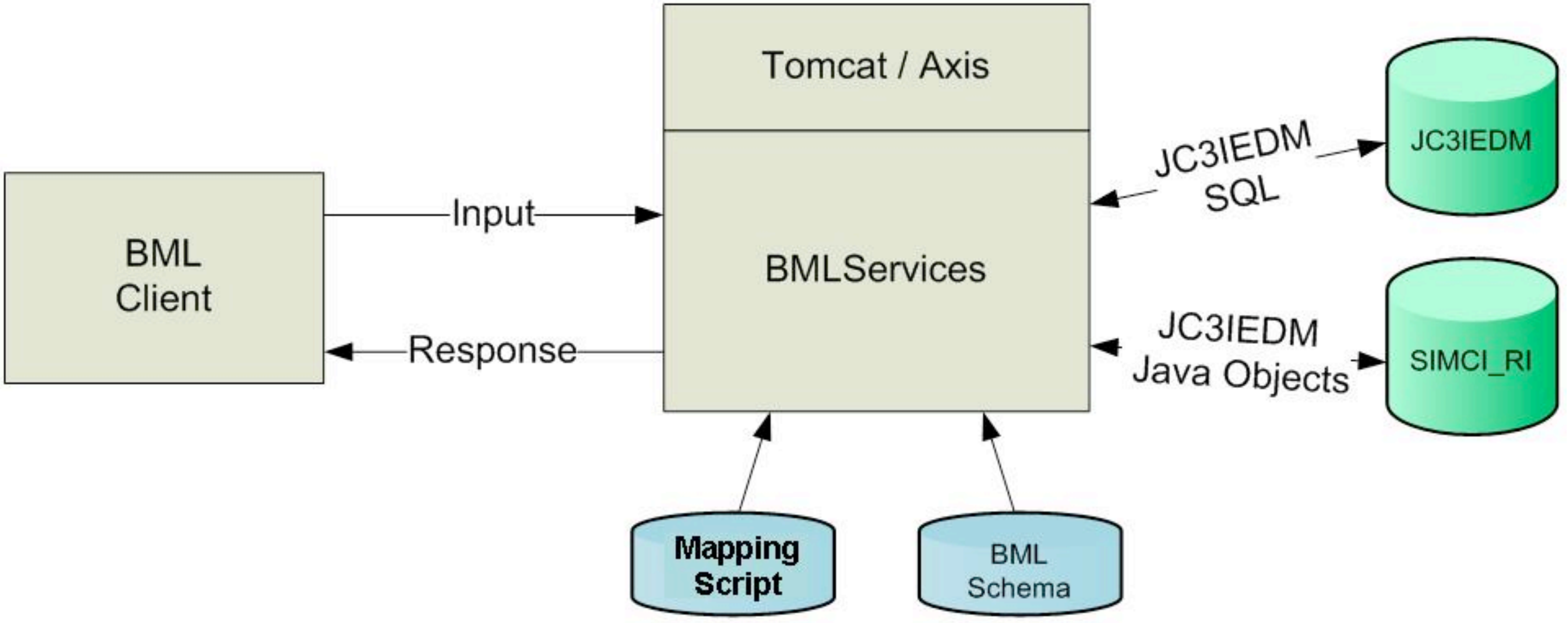
. . . (2 more pages like this)

Example Condensed Scripting Language

for complete TaskeeWhoPush

```
Transaction TaskeeWhoPush (task_act_id)
{
   Query GET table unit SET unit_id
        Column (formal_abbrd_name_txt SET UnitID);
   Query PUT table act_res
     Column (act_id FROM task_act_id NEWKEY)
     Column (act_res_ix FROM act_res_index)
     Column (cat_code FROM ”RI”)
   Column (authorising_org_id FROM unit_id);
   Query PUT table act_res_item
     Column (act_id FROM task_act_id)
     Column (act_res_ix FROM act_res_index)
     Column (obj_item_id FROM unit_id);
   Return tag Result FROM “OK”;
}
```

Scripted BML WS Configuration

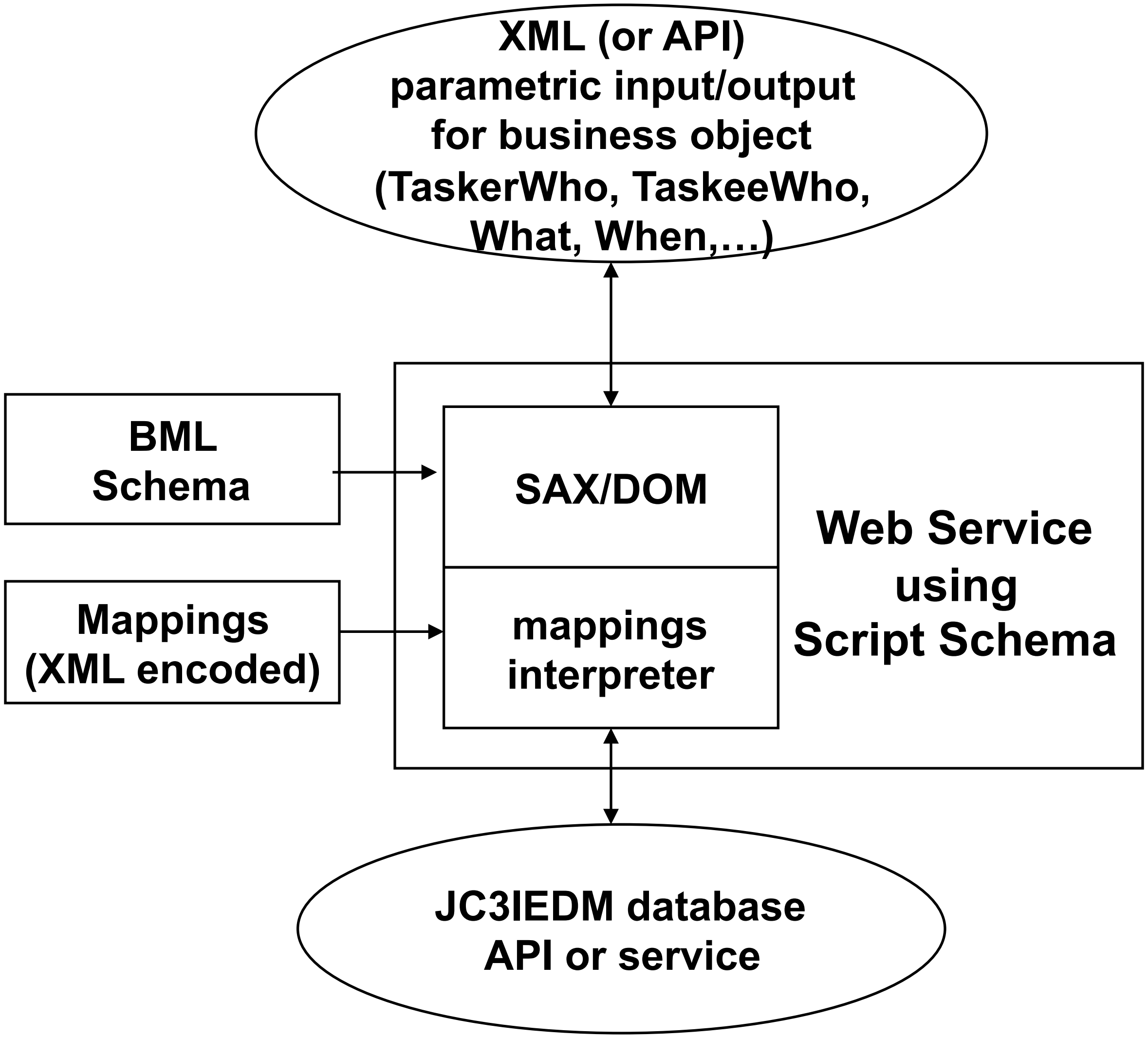


Two implementations: MySQL and SIMCI RI

Scripted BML WS Design

- Basic operations: *push* and *pull*
  - Currently, servers for SQL and RI databases
  - Scripts implement BML Orders and Reports
- Script defines implementation of Business Objects (constituents of the higher-level BML grammar) over the JC3IEDM data model
  - BO is an XML subtree rooted at a defined node in the XML file – can invoke other BO
- Interpreter uses two files plus WS input
  - Mapping file contains script
  - BML schema file provides necessary context

Scripted Interpreter

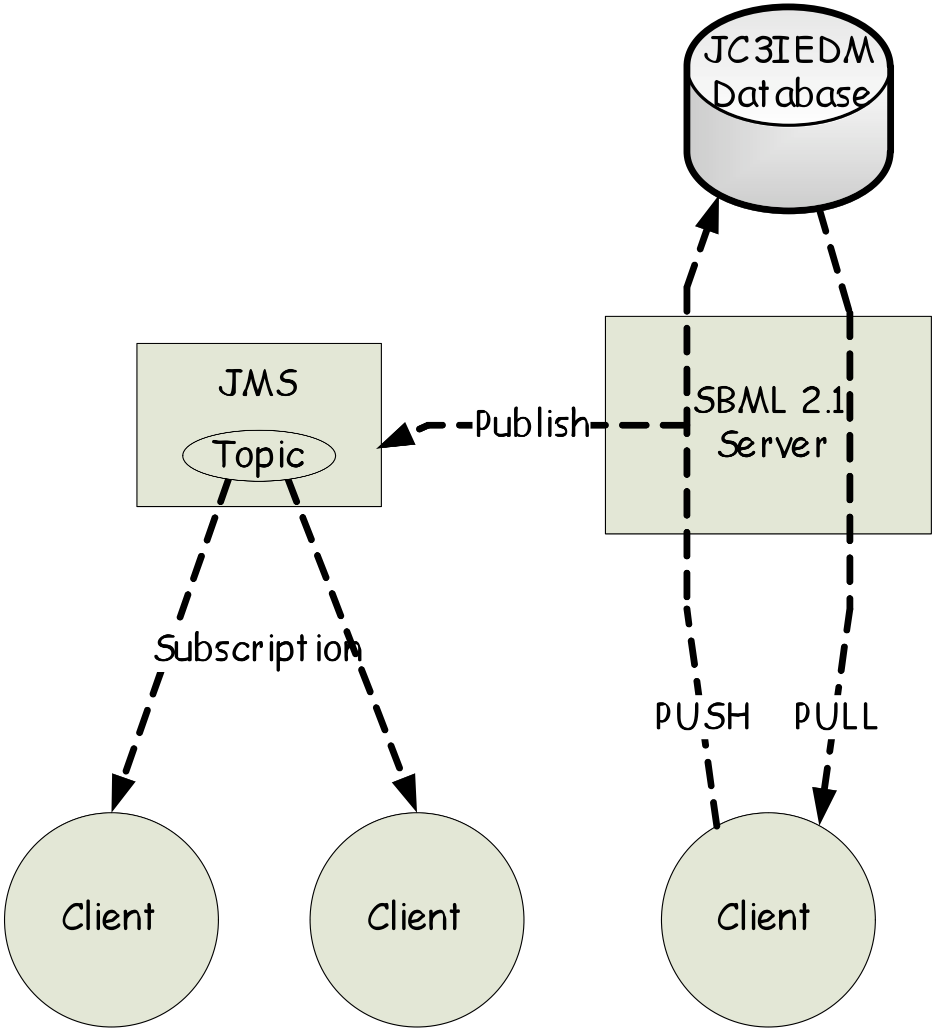


Why Publish/Subscribe

- "Pure" Web Service is always accessed by *push* or *pull* transaction from client
  - No provision for server to initiate action
- For clients to stay up to date they must pull latest status from server at rate determined by their need for up-to-date information (called *polling*)
  - Result: MSG-048 server in 2008 spent most of its time responding to status pulls
- Publish/subscribe gets around this by letting clients identify the categories of information they need – they *subscribe* to *Topics*
  - Server sends them a copy of every update associated with each subscribe Topic
  - More timely updates and a dramatic reduction in overhead

Publish/Subscribe Architecture

new in SBMLv2.1



Publish/Subscribe in SBML

- GMU team chose JBoss publish/subscribe for compatibility with SIMCI-CP architecture
  - Based on Java Messaging Service
- No native C++ interface so we provided one based on the Java Native Interface (JNI)
  - C++ interface developers got on well with this
- Established set of simple Topics based on Order and Report types
  - More sophisticated choice might have reduced message load on clients
  - However this was our first try so we opted for simplicity
  - Worked well but we might opt for more sophisticated Topics in future

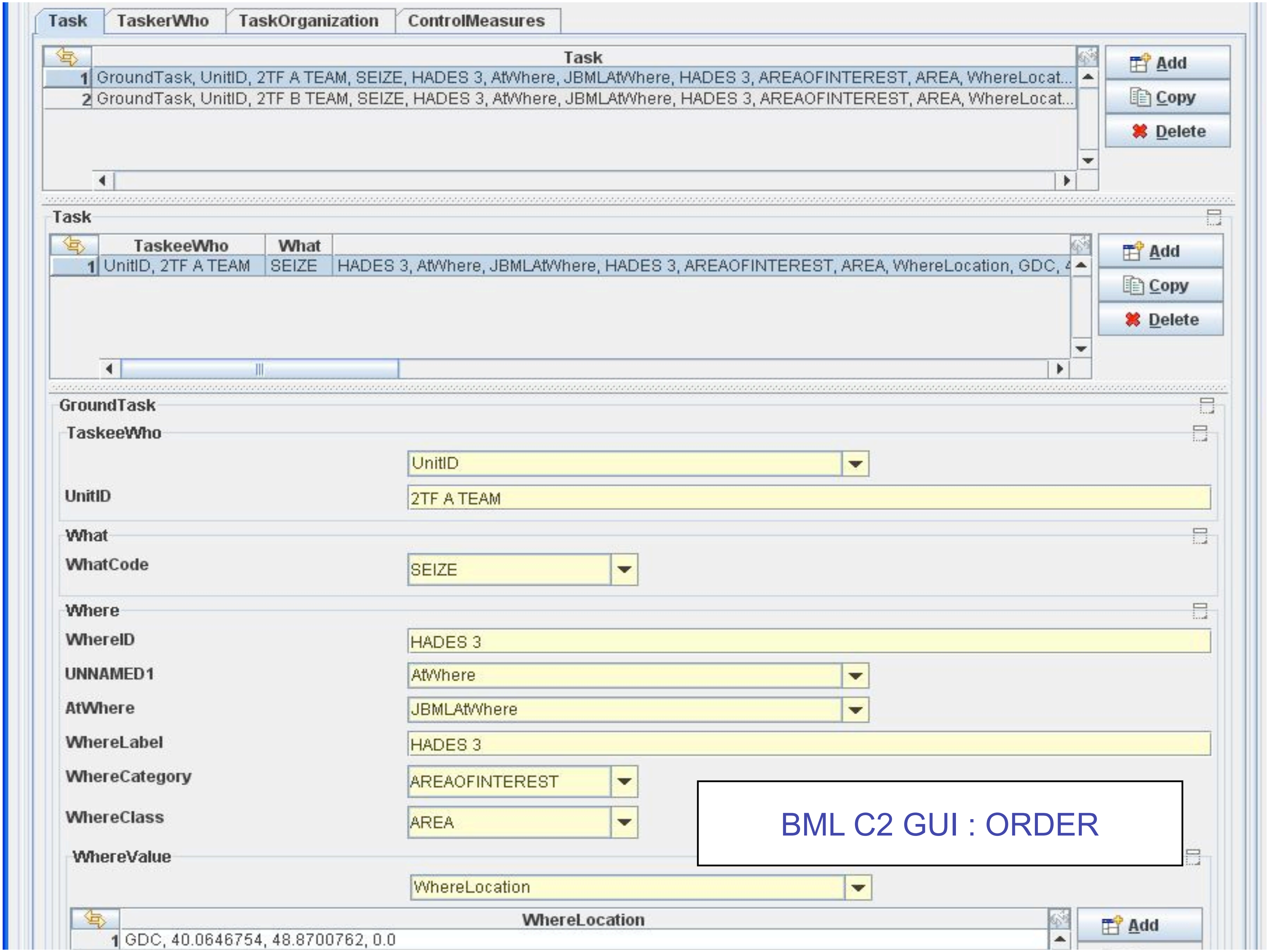
Scripted BML Summary

- Scripted BML WS has great potential as development tool
  - Enables developing services more rapidly
  - Eliminates the broad class of bugs that can’t be expressed in the script
  - Will be available as open source, offered as Reference Implementation
- Scripting language provides a highly concise definition of BML mapping
  - Paradigm for executable specification
  - Recommended that SISO C-BML use this approach
- Easy to evolve with data model
  - JC3IEDM or any other
- SBML software available open source from GMU

BML C2 GUI

- Inspired by FGAN C2 Lexical Grammar GUI
- Provides a way to inspect and modify BML orders and reports in context of schema
  - Includes ability to display BML geospatial data and control features on an open format map
  - Can be used to provide simple input/output in BML system or inspect data during development
- Open source software for public release
  - Uses “Jaxfront” commercial open source software to generate forms from schema
  - And open map graphics

BML C2 GUI : ORDER



	Task
1	GroundTask, UnitID, 2TF A TEAM, SEIZE, HADES 3, AtWhere, JBMLAtWhere, HADES 3, AREAOFINTEREST, AREA, WhereLocat...
2	GroundTask, UnitID, 2TF B TEAM, SEIZE, HADES 3, AtWhere, JBMLAtWhere, HADES 3, AREAOFINTEREST, AREA, WhereLocat...

Task

	TaskeeWho	What	
1	UnitID, 2TF A TEAM	SEIZE	HADES 3, AtWhere, JBMLAtWhere, HADES 3, AREAOFINTEREST, AREA, WhereLocation, GDC, 4

GroundTask

TaskeeWho

	UnitID
UnitID	2TF A TEAM

What

WhatCode	SEIZE

Where

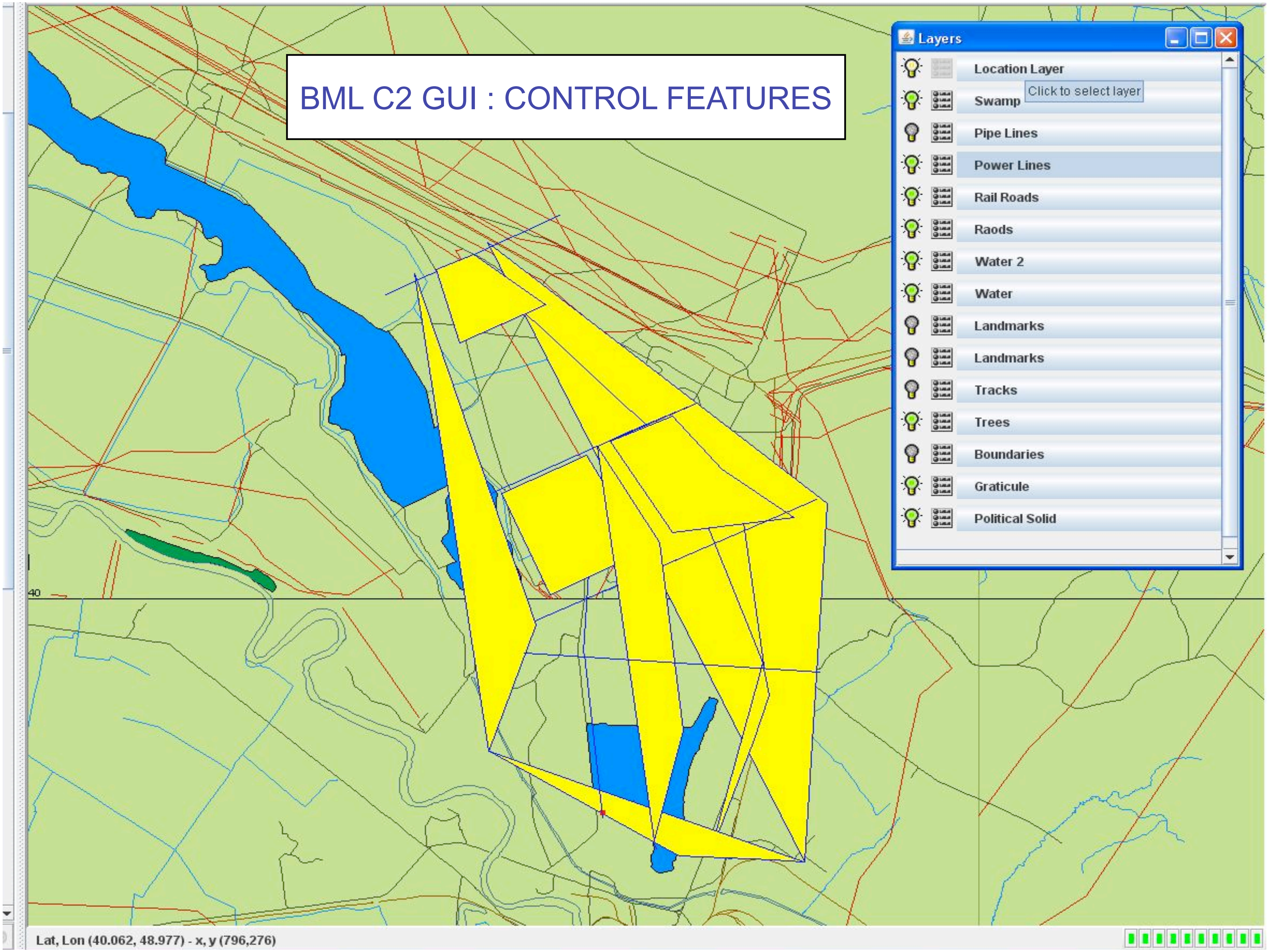
WhereID	HADES 3
UNNAMED1	AtWhere
AtWhere	JBMLAtWhere
WhereLabel	HADES 3
WhereCategory	AREAOFINTEREST
WhereClass	AREA

WhereValue

WhereLocation

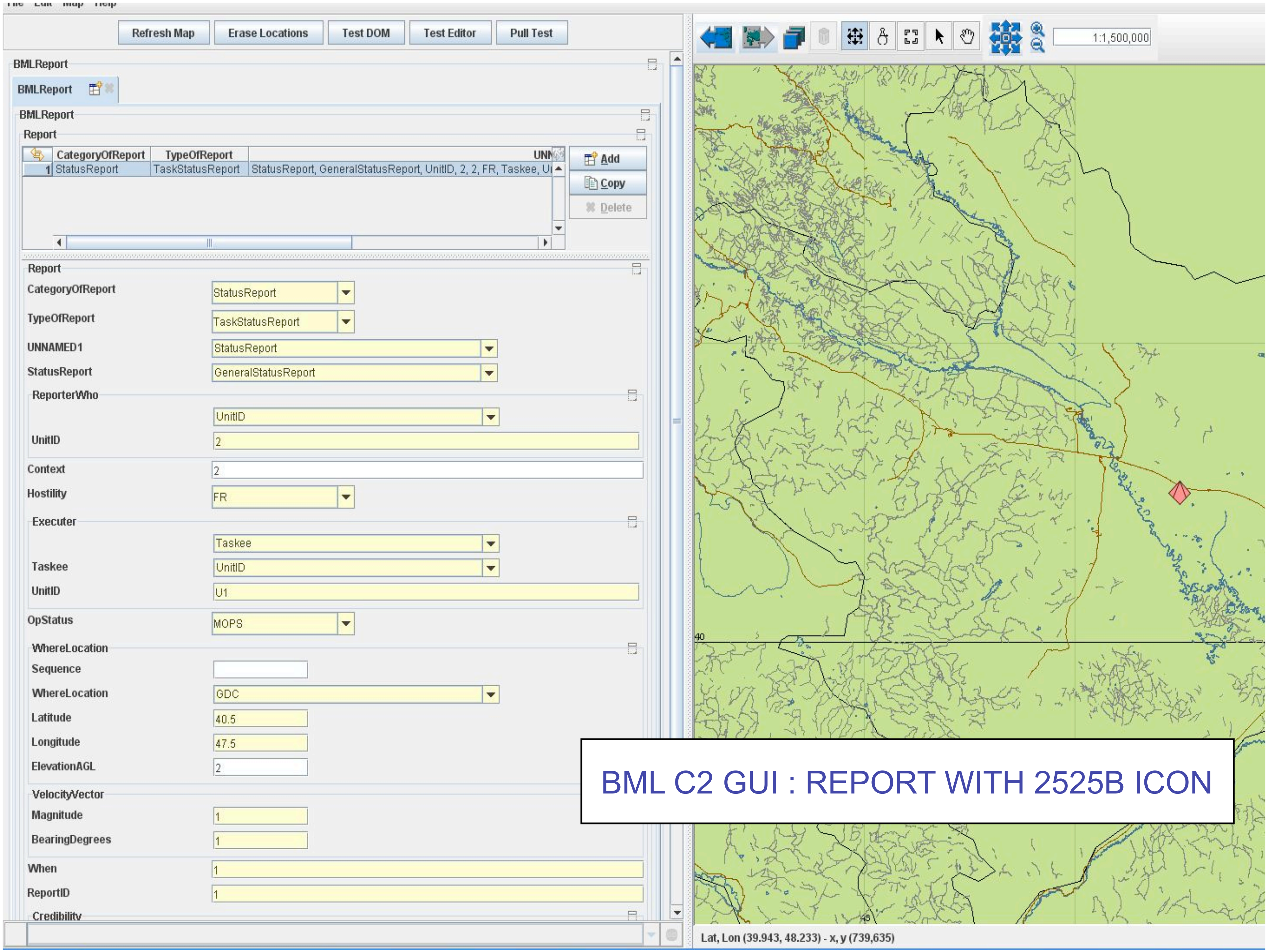
	WhereLocation
1	GDC, 40.0646754, 48.8700762, 0.0

BML C2 GUI : CONTROL FEATURES



Lat, Lon (40.062, 48.977) - x, y (796,276)

BML C2 GUI : REPORT WITH 2525B ICON



BMLReport

BMLReport

BMLReport

Report

	CategoryOfReport	TypeOfReport	UNN
1	StatusReport	TaskStatusReport	StatusReport, GeneralStatusReport, UnitID, 2, 2, FR, Taskee, U

Report

CategoryOfReport	StatusReport
TypeOfReport	TaskStatusReport
UNNAMED1	StatusReport
StatusReport	GeneralStatusReport
ReporterWho	
	UnitID
UnitID	2
Context	2
Hostility	FR
Executer	
	Taskee
Taskee	UnitID
UnitID	U1
OpStatus	MOPS
WhereLocation	
Sequence	
WhereLocation	GDC
Latitude	40.5
Longitude	47.5
ElevationAGL	2
VelocityVector	
Magnitude	1
BearingDegrees	1
When	1
ReportID	1
Credibility	

Lat, Lon (39.943, 48.233) - x, y (739,635)

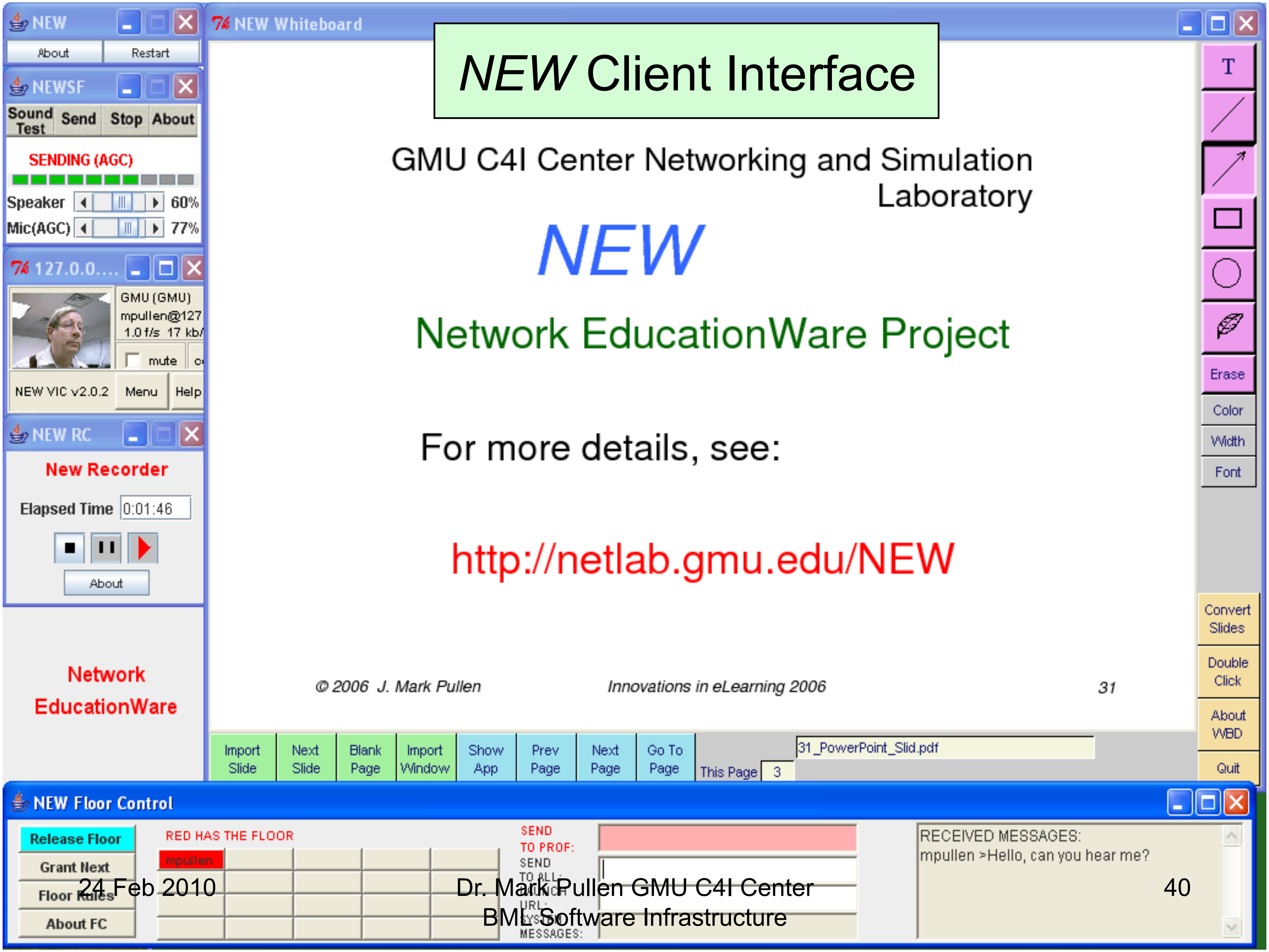
Other Software for Distributed Development

- Collaboration software
  - Internet teleconferencing
- Version Management
  - Subversion versioning system
- Trac Management Software
  - Seamless integration with Subversion
  - Wiki with collaborative editing
  - Issue tracking and resolution
- Operation software
  - Initialization of Web service

Internet Audiographic Conferencing

- Used GMU Network EducationWare (NEW)
  - Runs over open Internet and is open source/free
  - Course delivery software also for conferencing
- Important functions:
  - Half-duplex voice with quick turnover
  - Floor control establishes who is speaking
  - Whiteboard allows document to be shared and annotated by all

NEW Client Interface



GMU C4I Center Networking and Simulation Laboratory

NEW

Network EducationWare Project

For more details, see:

http://netlab.gmu.edu/NEW

© 2006 J. Mark Pullen Innovations in eLearning 2006 31

SENDING (AGC)

Speaker 60%

Mic(AGC) 77%

GMU (GMU)
mpullen@127
1.0 f/s 17 kb/

NEW VIC v2.0.2

New Recorder

Elapsed Time 0:01:46

Network EducationWare

31_PowerPoint_Slid.pdf

This Page 3

RED HAS THE FLOOR

mpullen

SEND TO PROF:

SEND TO ALL:

LAUNCH URL:

SYSTEM MESSAGES:

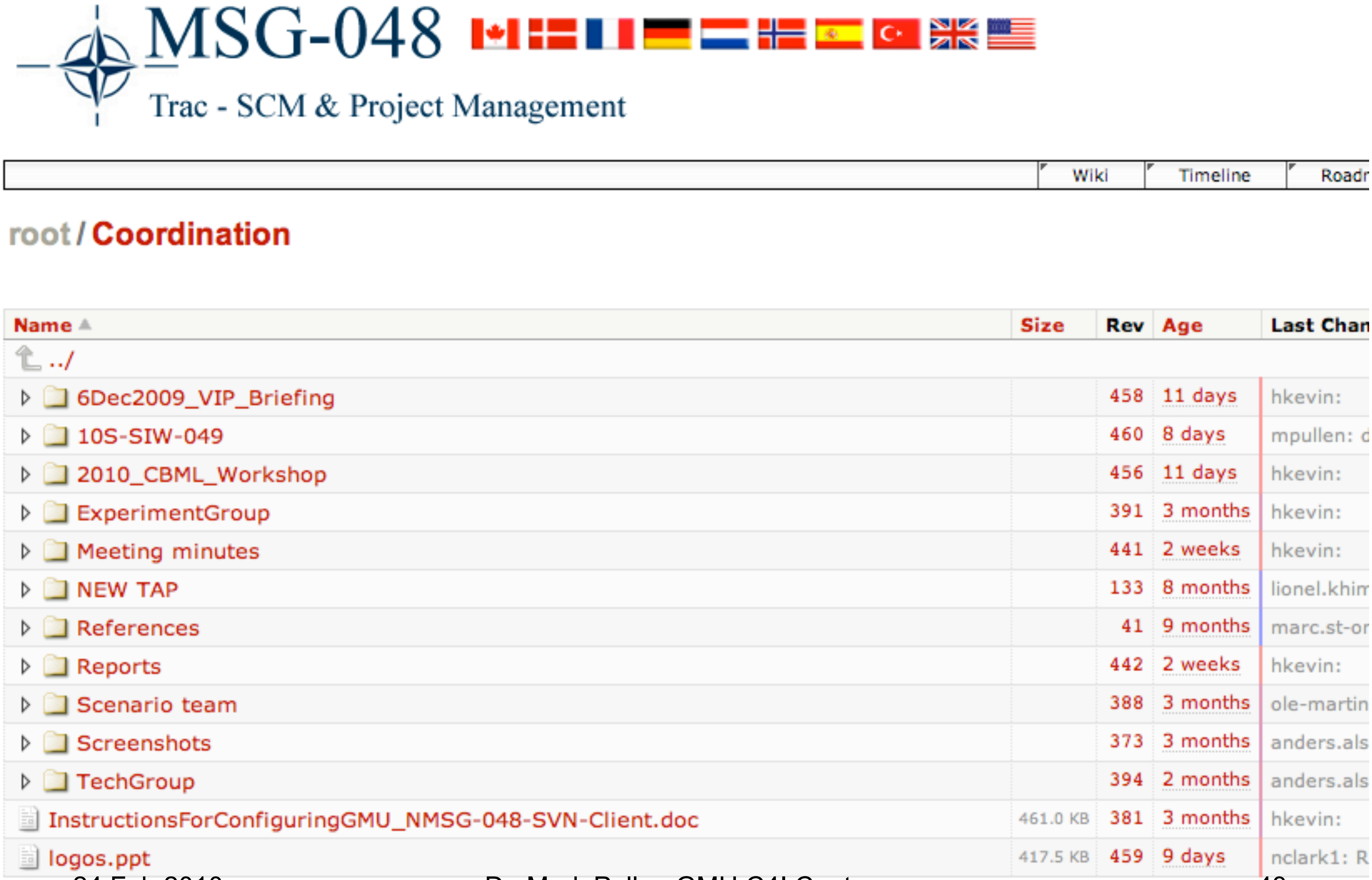
RECEIVED MESSAGES:
mpullen >Hello, can you hear me?

Subversion Versioning System

- An excellent way to maintain shared access to changing documents
  - Maintains history of all changes
  - Normally, latest update is downloaded
  - Role-based access control
- Can be accessed via Web browser
- Upload requires a special (free) client

Trac Wiki Built on Subversion

- A *Wiki* is a set of shared files that a group of users can access and update
- Using Trac, any Wiki input always can be recovered
- MSG-048 built its Wiki around the functions we were performing
  - Experiment planning and setup
  - Technical planning and execution
  - Schema maintenance



MSG-048

Trac - SCM & Project Management

root / Coordination

Name ▲	Size	Rev	Age	Last Chan
../				
6Dec2009_VIP_Briefing		458	11 days	hkevin:
10S-SIW-049		460	8 days	mpullen: d
2010_CBML_Workshop		456	11 days	hkevin:
ExperimentGroup		391	3 months	hkevin:
Meeting minutes		441	2 weeks	hkevin:
NEW TAP		133	8 months	lionel.khim
References		41	9 months	marc.st-or
Reports		442	2 weeks	hkevin:
Scenario team		388	3 months	ole-martin
Screenshots		373	3 months	anders.als
TechGroup		394	2 months	anders.als
InstructionsForConfiguringGMU_NMSG-048-SVN-Client.doc	461.0 KB	381	3 months	hkevin:
logos.ppt	417.5 KB	459	9 days	nclark1: R

Trac Bug Reporting

- With 8 project teams finding and reporting bugs, it was very difficult to manage resolution
- MSG-048 used the bug reporting and management system built into Trac
  - Collects all pertinent details of problem
  - Informs people who need to know
  - Maintains status accessible online
  - Programmer posts when problem resolved

Initialization

- Getting all connected systems to have a common state is a significant challenge
- In 2009, MSG-048 settled for providing a mechanism to initialize the BML Server
  - "NewWho" service pushed values of all needed JC3IEDM attributes needed for a unit
    - Some simulations actually did this while running
  - Much more work is needed to initialize all systems automatically
  - Start/stop and logging of operation also needed

Conclusion

- Operating a BML Coalition requires careful attention to infrastructure
- BML Schema and Server are essential factors in success
- Server performance and stability critical
  - Ease of adapting server also important while BML is experimental
  - Publish/subscribe capability essential
- Other supporting software also is essential to distributed development of and by coalition