Best Practices and Design Patterns for Defense Developers

Kyle Krattiger
Christopher Moore

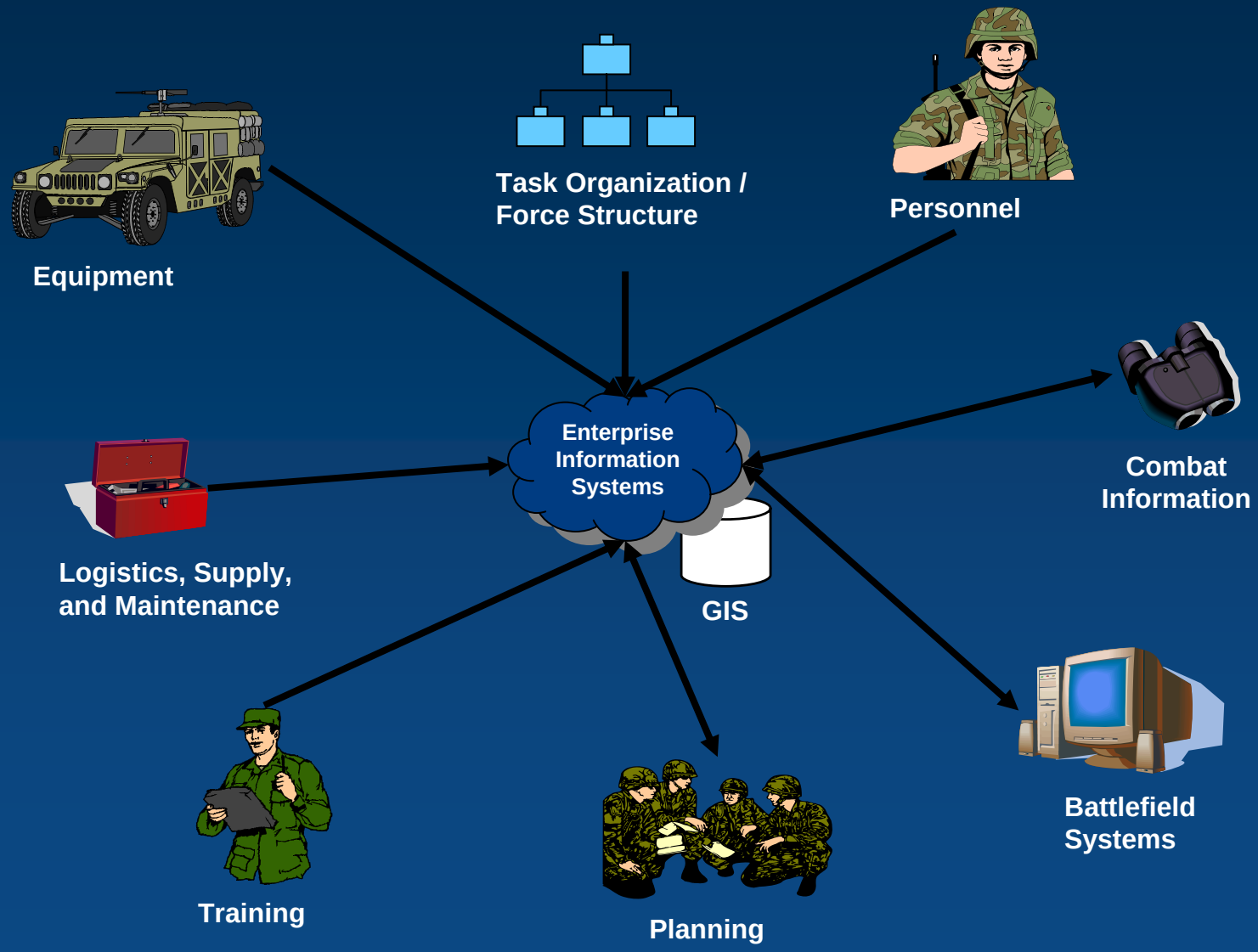
ESRI Defense Solutions

Agenda

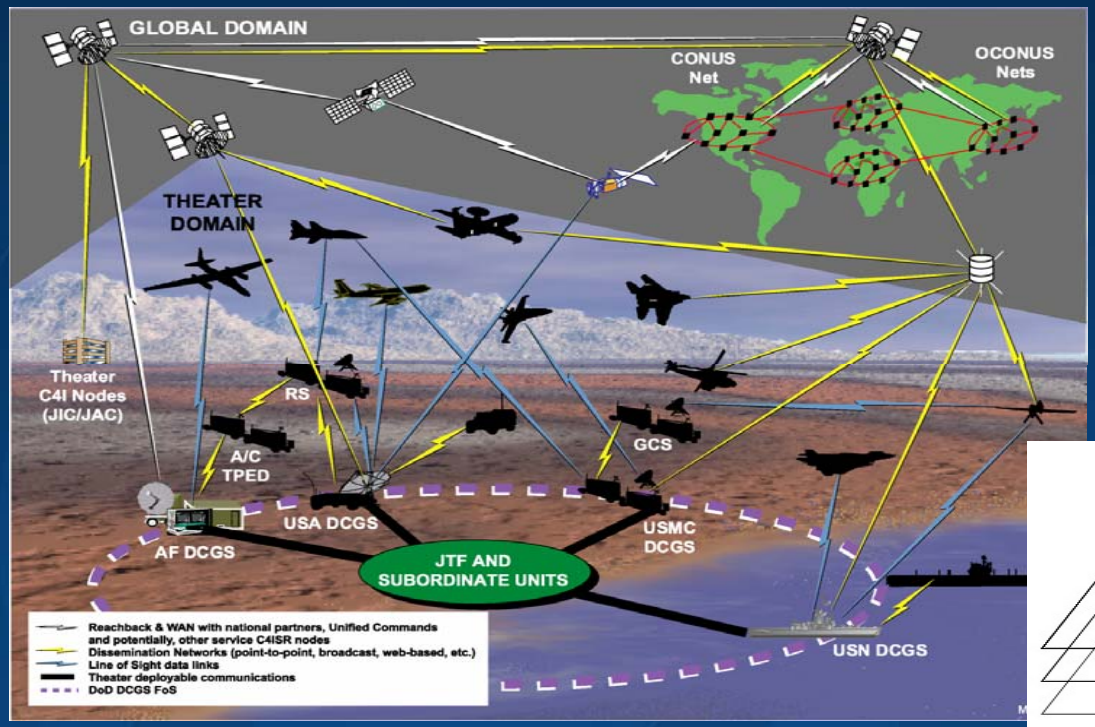
- Typical Defense Application Requirements
- Defense Development and the ArcGIS Platform
- About Patterns and Practices
- Sample Best Practices
- Military Analyst
- Military Symbology with the Military Overlay Editor (MOLE)
- Sample Reference Application
- Conclusion

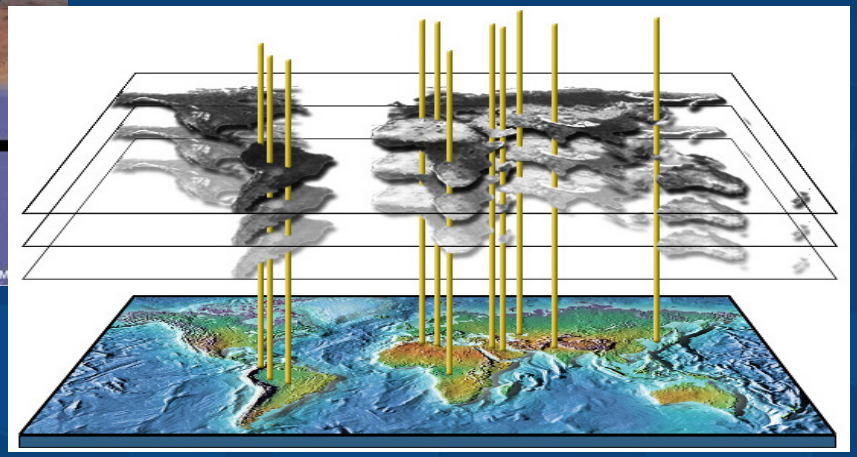
Defense Application Development and the ArcGIS Platform

Elements of Defense Information Systems



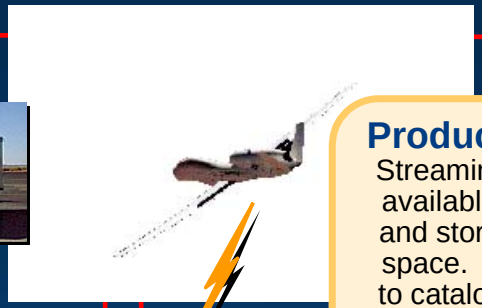
Providing Geospatial Intelligence …A Common Operational Perspective





Geospatial Visualization and Analysis…
a Key Component of the Integrated Battlespace

Making Data Available



Consumer
Automated search of data based on core metadata standard. Pulls data of interest. Based on producer registered format and definitions, translates into needed structure.

Producer
Streaming video available for use, tagged and stored in shared space. Metadata added to catalog based on registered format.

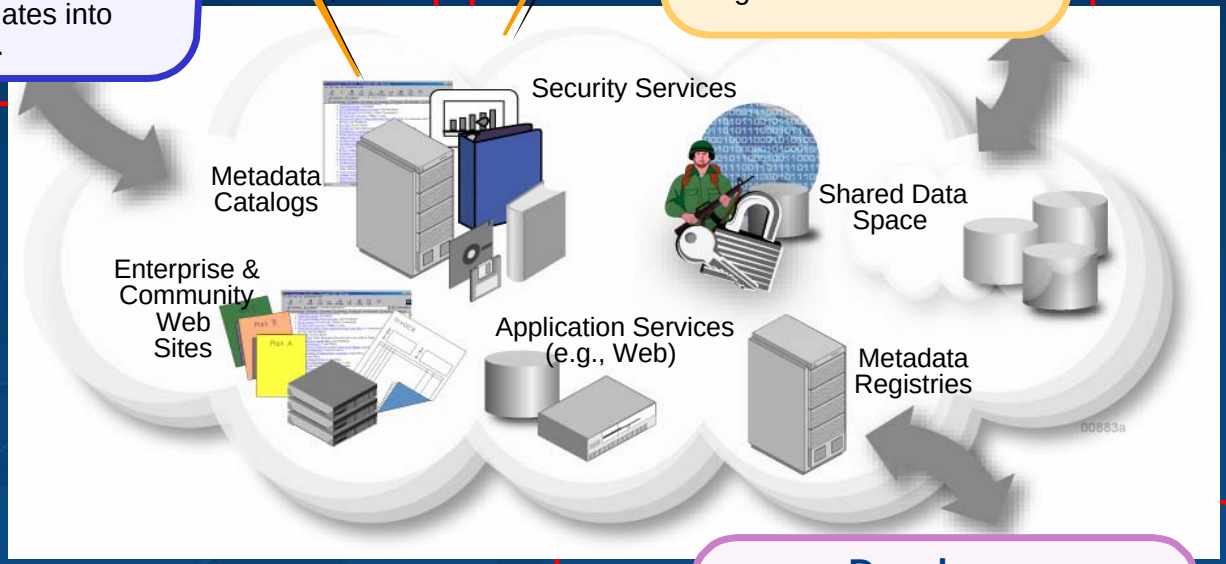
Describes content using metadata

Posts metadata in catalogs and data in shared space

Searches metadata catalogs to find data (e.g., community and enterprise-wide search services)

Analyzes metadata to determine context of data found

Pulls selected data based on understanding of metadata

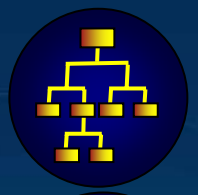


Ubiquitous Global Network

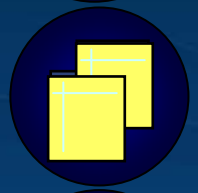
Posts to and uses metadata registries to structure data and document formats for reuse and interoperability

Developer
Understand the data format to build applications that post, process, exchange, and display target information.

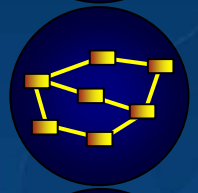
“Abstracts” Intelligence (Knowledge): Five Basic Elements



Data Models



Geodata Sets



Work Flow Models

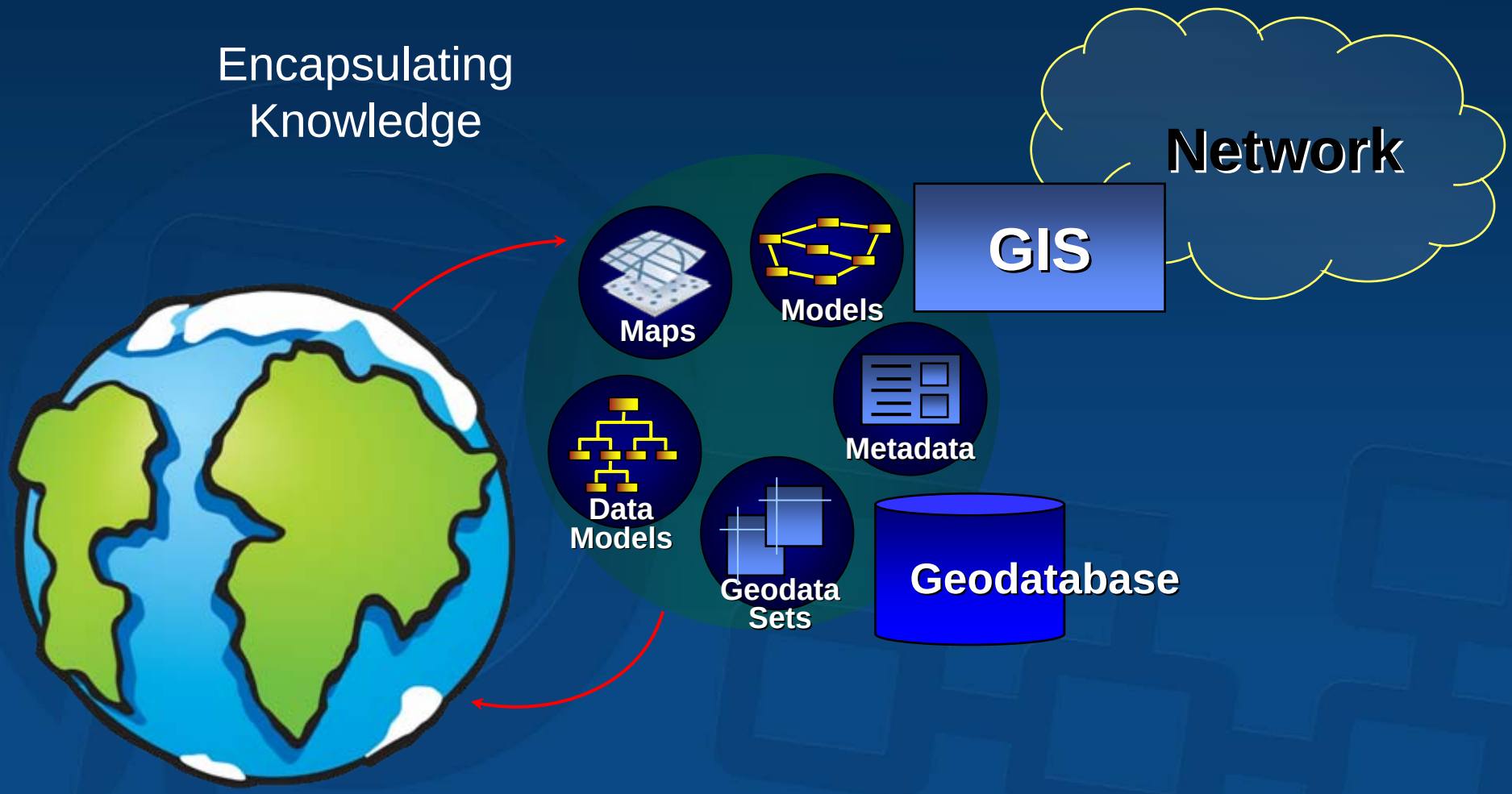
Maps and Globes

Metadata

Together they represent the building blocks of geographic knowledge

GIS Manages These Elements in a Geodatabase

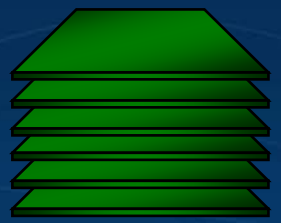
Encapsulating Knowledge



. . . and Makes Them Directly Usable & Accessible

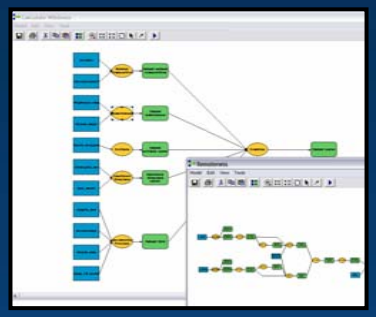
Analysis and Decision Support …with GIS

Fuse Data/Knowledge



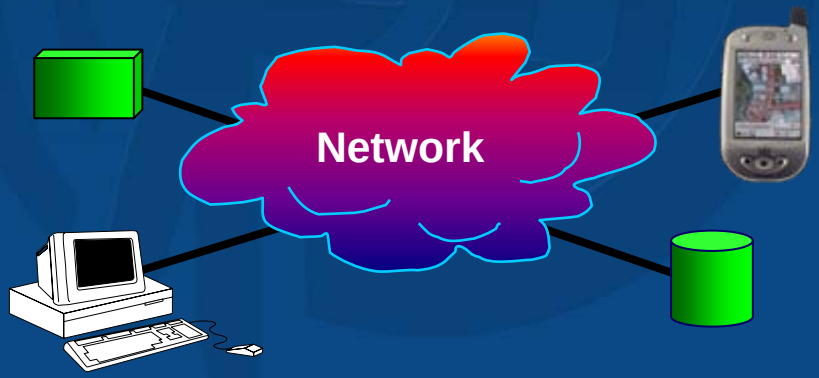
Space and Time

Systematize Analysis/ Processing/Workflow



Enable Intuitive Analysis and Decisions

Share the Data, Knowledge, Workflow, and Decisions



Better, Faster, and Cheaper

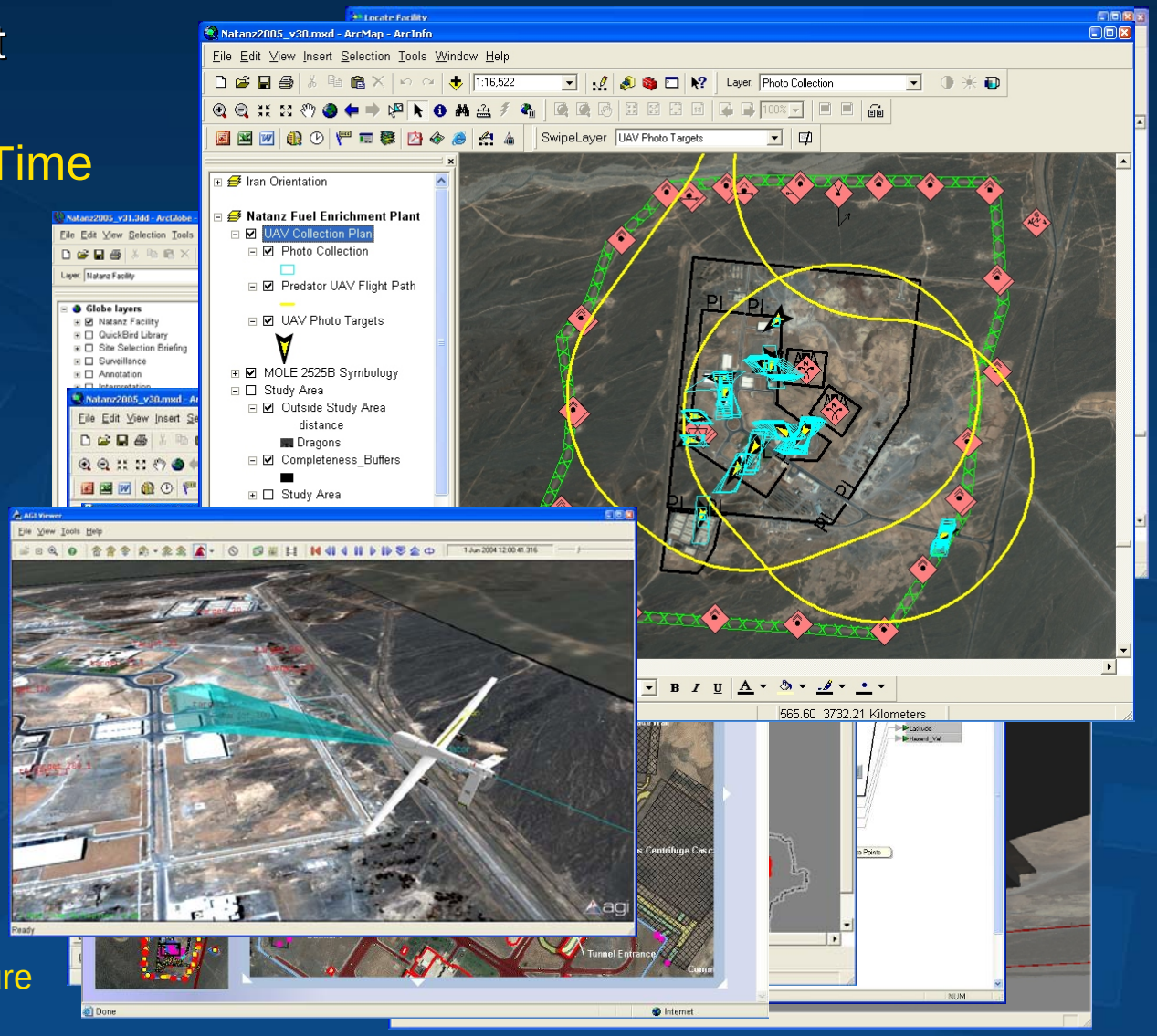
Typical Defense Application Requirements

1. Map Display
2. Map Navigation
3. Military Symbology Display
4. Symbology – Static & Dynamic
5. Add and remove layers
6. Change layer properties
   - Layer visibility, order, transparency, etc.
7. Intervisibility Analysis
   - Line of Sight, Terrain Profile, Viewsheds
8. Creation and Update Performance

Functional Components of a GIS

Geospatial Tradecraft

- Mapping Space & Time
- Visualization
- Interoperability
- Metadata, Quality, Geodatabase
- Geoprocessing, predictability & Uncertainty
- Collaboration



…In a Services Oriented Architecture

Factors that Differentiate Defense Development

- Defense usage differs due to the importance of
  - Having numerous Client Application Viewers available (Thin & Thick, Web, Smart Client, 2D & 3D)
  - Highly collaborative
  - Data modeling (sharing, customizing, and extending)
  - Data interoperability with a wide range of legacy and emerging data types (VPF, RFP, WMS/WFS, etc.)
  - Performance – operating in a real time environment
  - Security
  - Scalability
  - Terrain and Terrain Modeling/Analysis

Typical Defense Applications based on the ArcGIS Platform

- Command and Control
- Battlefield Planning and Management
  - Mission Planning
  - Terrain analysis
- Intelligence Gathering
- Defense and Intelligence Geospatial Analysis
- Data Product Preparation (Raster and Vector)
- Incident Planning and Response
- Search and Rescue
- Facility, environmental, and infrastructure management

ArcGIS Integration Strategies and Deployment Platforms – Desktop Clients

- Desktop Clients
  - ArcGIS Desktop
    - Desktop Extensions
    - Leverage the full power of ArcGIS Desktop and develop plug-in extensions for any additional needed functionality
  - ArcGIS Engine
    - Standalone Applications based on ArcGIS Engine Controls and the standard ArcObjects GIS framework
    - Cost-effective deployment with a smaller footprint requiring only an ArcGIS Engine Runtime
  - Development in standard developer languages including COM, .NET, Java, and C++

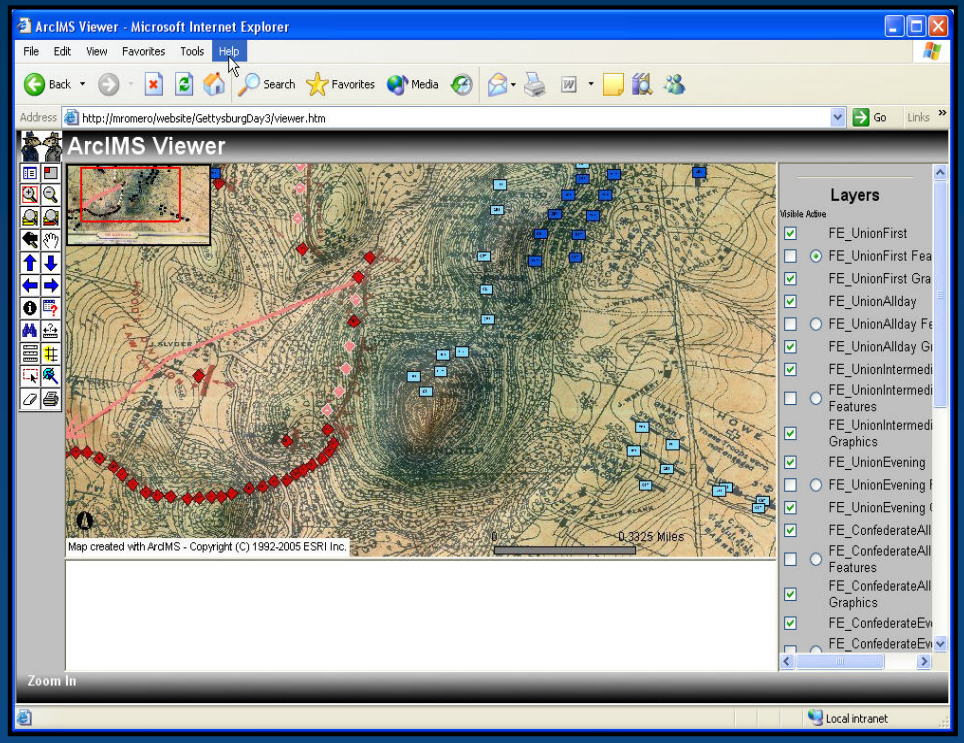
ArcGIS Integration Strategies and Deployment Platforms – Web Clients and Smart Clients

- Web Clients
  - Embedded Applications (Applets/ActiveX Controls)
  - Pure HTML/XML and JavaScript over HTTP(S)
    - Can use standard Web Browsers on all computers
    - Requires no additional software
    - Useful for users that do not have advanced requirements
  - ArcIMS and ArcGIS Server
- Smart Clients – ADF Framework
  - Extends ArcGIS Server with an application web service that supports:
    - Raster and vector extraction/caching
    - Transaction Management
  - New easy-to-use Smart Client Developer Kit
    - Cross platform (mobile and desktop)
    - Templates, integrated documentation

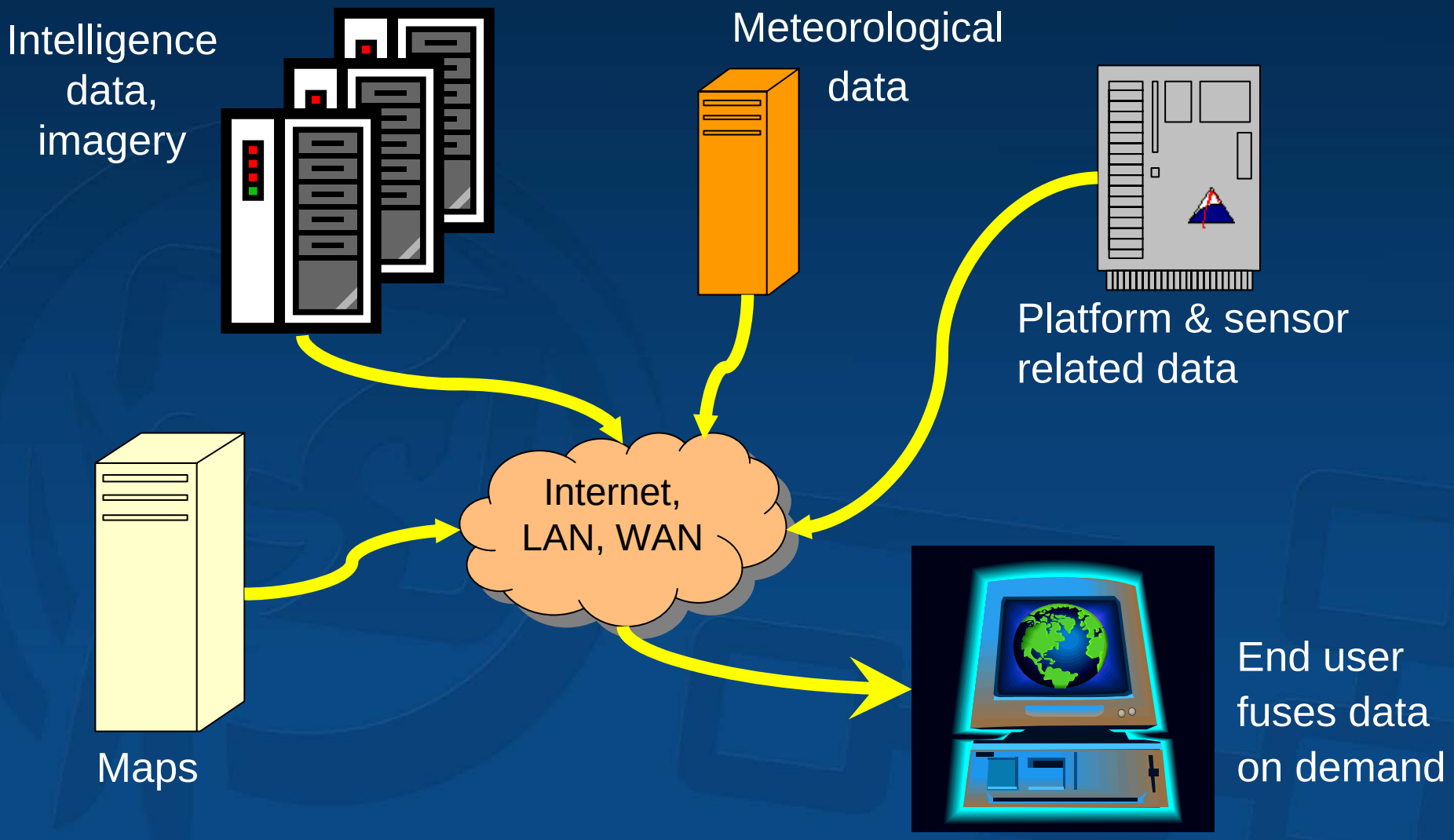
Deployment Platforms
Web Clients

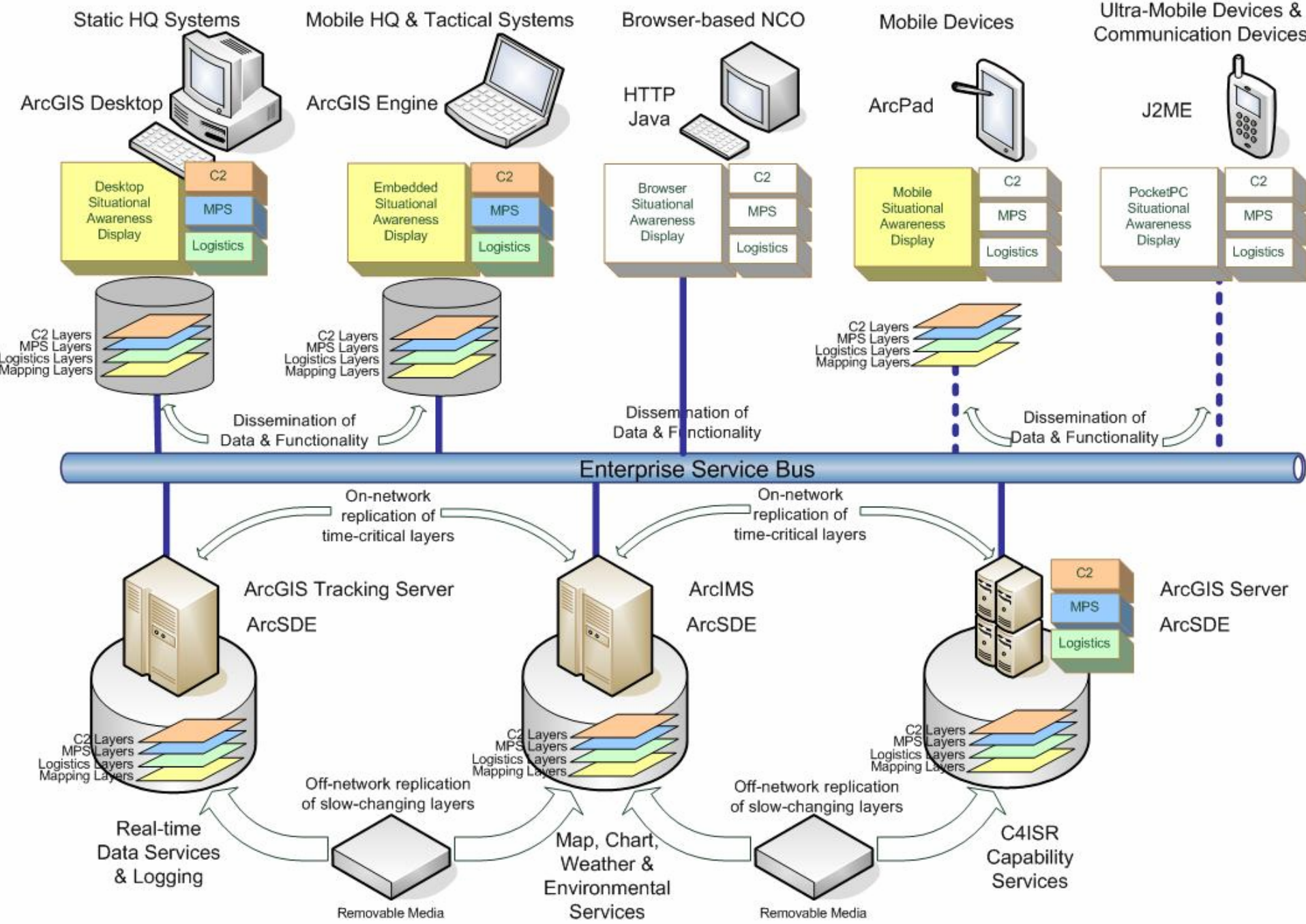
ArcGIS Server Client

ArcIMS Client



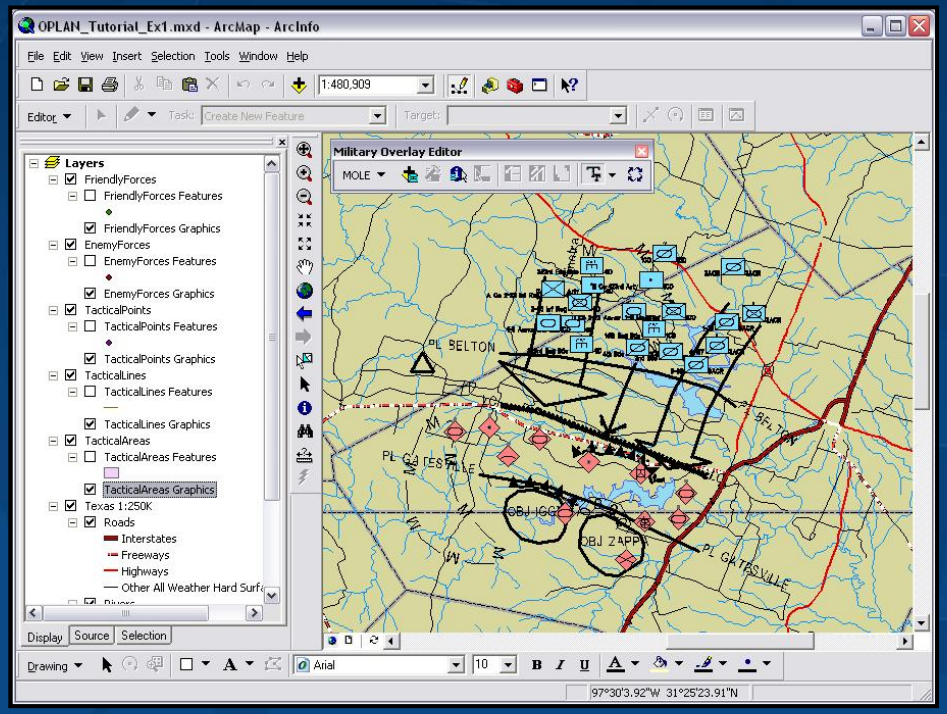
Net-Centric via Web Services



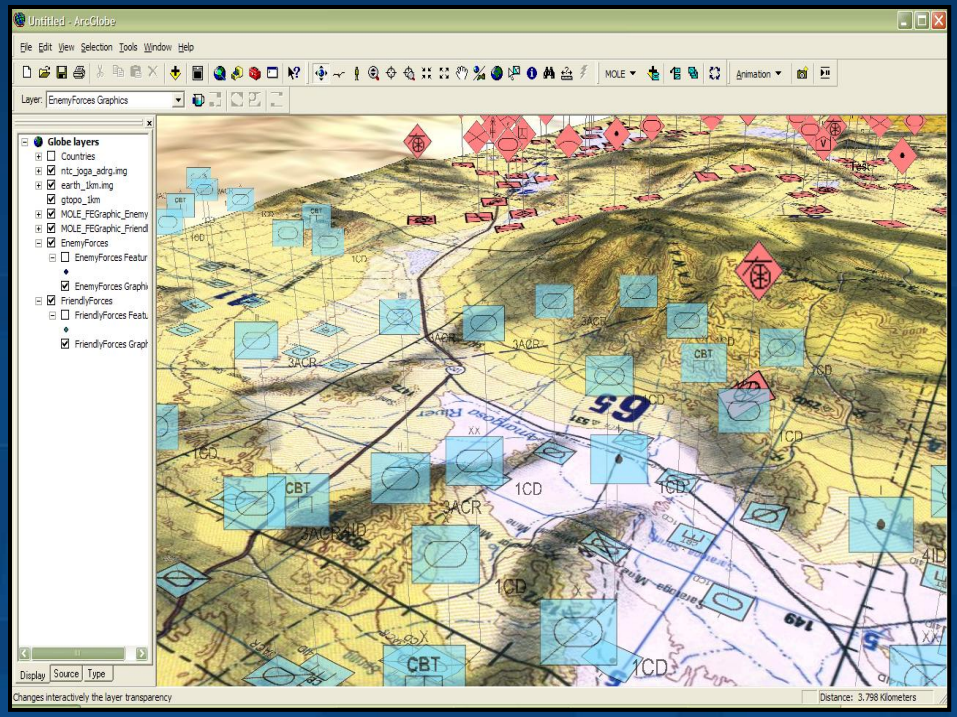

Static HQ Systems
ArcGIS Desktop
Desktop Situational Awareness Display
C2
MPS
Logistics
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
Mobile HQ & Tactical Systems
ArcGIS Engine
Embedded Situational Awareness Display
C2
MPS
Logistics
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
Browser-based NCO
HTTP
Java
Browser Situational Awareness Display
C2
MPS
Logistics
Mobile Devices
ArcPad
Mobile Situational Awareness Display
C2
MPS
Logistics
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
Ultra-Mobile Devices & Communication Devices
J2ME
PocketPC Situational Awareness Display
C2
MPS
Logistics
Dissemination of Data & Functionality
Dissemination of Data & Functionality
Dissemination of Data & Functionality
Enterprise Service Bus
On-network replication of time-critical layers
On-network replication of time-critical layers
ArcGIS Tracking Server
ArcSDE
ArcIMS
ArcSDE
ArcGIS Server
ArcSDE
C2
MPS
Logistics
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
C2 Layers
MPS Layers
Logistics Layers
Mapping Layers
Off-network replication of slow-changing layers
Off-network replication of slow-changing layers
Real-time Data Services & Logging
Removable Media
Map, Chart, Weather & Environmental Services
Removable Media
C4ISR Capability Services

Deployment Platforms
Desktop Platforms

2D

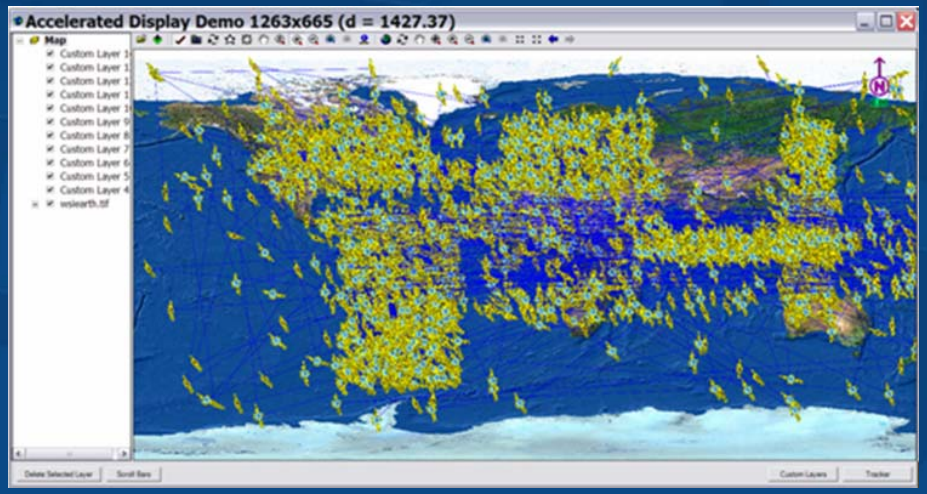


3D



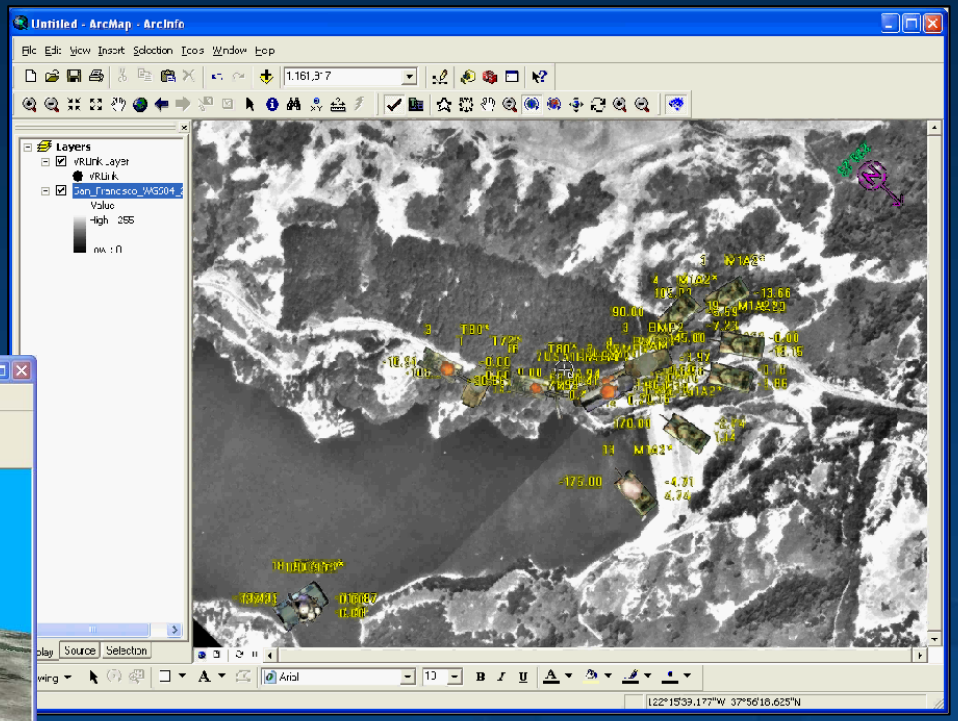
Dynamic GIS

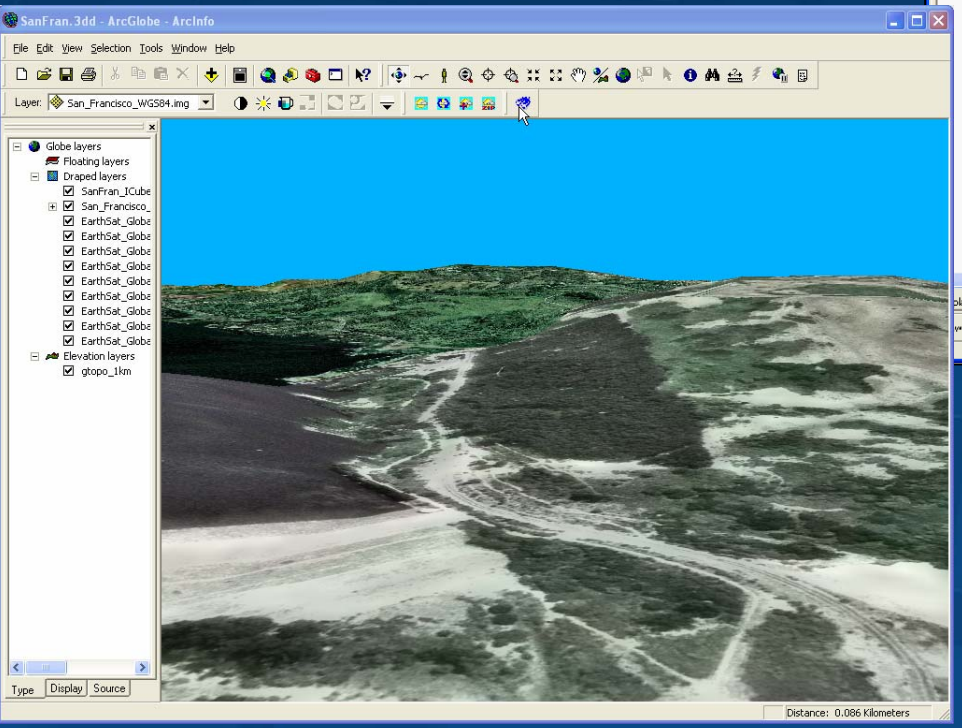
- Dynamic Display
  - of spatially related information
  - Visualizing changes in real-time in a continuous manner
    - Animated moving objects
    - Animated object representations
- Dynamic Content
  - Frequent update of the content
    - Location
    - Orientation
    - Attributes
- Display Filtering



Defense Modeling and Simulation HLA/DIS Integration Demonstration

- Embedded training
- Planning and rehearsal





1 - ArcMapHPDVRLink2D_1.mov.lnk

- Live operation
- After Action Review (AAR)

About Patterns and Practices

Patterns and Practices Motivation

- Patterns facilitate reuse of successful software architectures, designs, and *proven* solutions
- Capture accumulated expert knowledge and design tradeoffs
- Pass on this expert 'wisdom' on how to architect & implement software
  - Example: of a master craftsman teaching an apprentice a trade.
- Sample ArcGIS Design patterns:
  - Connecting to different data sources
  - Moving objects on the display
  - Adding Vector data

Patterns and Practices: Types of Guidance

- Guides
  - Written guidance covering topics such as
  - Patterns, Architecture, Integration, Performance, etc.
- Reference Implementations
  - Compilable and Executable sample applications that demonstrate patterns, practices and guidance in action
- Application Blocks
  - Reusable source code components that provide proven solutions to common development tasks

Patterns and Practices

ESRI Developer Network - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://edndoc.esri.com/arcobjects/9.1/ Go

Links MOLE_SERVER_DEMO Demo ArcObjects Online ArcZone CJMTK CQ CQ-Star Team

ESRI.com | Support | More ESRI Sites...

EDN ESRI Developer Network Login ESRI Developer Network

Home | Documentation Library | Discussion Forums | Downloads | Code Exchange | Media Center | Developer Summit

Hide Locate Home Get Page URL

- Appendices
- Technical Documents
  - Technical Documents Start Pag
  - ArcGIS Bitmaps
  - Shape Comparision Language
  - Singleton Objects
  - Using Modeless Visual Basic Dia
  - 3D Analyst
  - Cross-Platform development
  - Geodatabase
  - Geoprocessing
    - Building Geoprocessing Func
  - Migrating
  - Names and IDs
  - Networks
  - Rasters
    - **Custom Raster Formats**
  - Server Development
  - Spatial Analyst

ArcGIS SDK

Technical Documents

Building a Customized Raster Format DLL

Raster support in ArcGIS is built on an open structure based upon the DLL technology. With this open structure, users can add their own raster format support into ArcGIS Desktop by creating a customized DLL and plugging it into the system. The customized raster format, created by implementing some predefined interfaces, will be treated the same as other built-in raster formats. The customized DLL created in ArcGIS Desktop can be plugged into other products such as ArcIMS (4.0.1 or later).

The zip file included here contains a technical paper, a reference book of the interface function definitions, C code samples and sample data, a configuration tool, and other supporting files.

Click the link below to download the zip file.

FormatDLLDistribution_v90.zip (Updated 08/06/04, 2730 KB Zip file)

Send feedback on this page

Internet

http://edn.esri.com/index.cfm?fa=home.welcome

Sample Best Practices

Best Practice General: Start with the Requirements

- Display Types and Content
  - 2D, 3D, Web Client
  - Standard, Military, and Custom Symbology
  - Performance and Update Rates
  - Number of Symbols and Labels
- Data Sources and Storage
  - Source
    - Hardware Devices, RDBMS's, Networked and Streaming Data Sources
  - Data Type
    - EGDB, Shapefile, PGDB, etc.
  - Update Frequency
  - Data Volume

Best Practice General:
Start with the Requirements - Other Considerations

- Display
  - Real-time
  - Refresh
  - Animation and moving objects
  - Caches
- Editing and User Interaction with Display and Data
- Tracking
- Layers
  - Custom layers
  - Normal layers

Best Practice: Data Management

- Understand Implications of I/O on performance
  - With layers tied to a disk-based data source, refresh commands will encounter I/O slowdown
  - Optimize performance by batching/grouping I/O operations where possible
- Understand Queries and Cursors
  - IFeatureCursor's obtained from calls to IFeatureClass's Insert, Search, or Update methods
  - These differing origins will have different performance implications - see ArcGIS Documentation for recommended usage
  - Query only the data you need to - known tiles if possible, not entire dataset
- Understand the performance of your Data Source
  - Performance for data sources varies
  - ArcSDE, Shapefile, PGDB, Networked Data Sources

Best Practice: Understand the Display Pipeline

- Normal Pipeline
  - Map::Refresh
    - Layer::Draw
      - Render::Draw
        - » Symbol::Draw
- Use IActiveView.PartialRefresh rather than the full IActiveView.Refresh
- Only Refresh the desired view phase, e.g. ActiveView.PartialRefresh(esriViewForeground, layer, desiredEnvelope)
- If moving graphics/symbols, refresh only those graphics that have changed
- If performing custom drawing, only draw those graphics which are visible and in the current extent
- *Dynamic Display* capabilities if applicable

Best Practice: Practices for 3D

- When Using Elevation Data, ensure that data set is projected, and that X, Y & Z values are the same
- Determining how to handle data that lacks an altitude component
  - Extruding, draping, ground clamping, terrain exaggeration
- Multiple level of detail models
- Using realistic 3D Models useful for some types of data (individual platforms) but not for all data
  - E.g. military tactical symbols that represent 1000s of entities

Best Practice:
Map Configuration Considerations

- Number of Layers
- Scales dependencies for
  - Layer
  - Labels
  - E.g. Not showing street level maps at a global scale
- Layer Order
- Static versus Dynamic Layers

Best Practice: Labeling

- Labeling is performance-expensive
- Drop altogether, if possible for moving/dynamic data
- Minimize the number of labels visible
  - Turn off labels when animating/moving (important)
  - Turn off when zoomed out beyond a meaningful extent
  - Consider using alternate methods to present detailed symbol attributes
    - Pop-up balloons, separate attribute window for selected graphic, or turn-on labels only for graphics near map center

Best Practice: Methods of Using MOLE

- Cached Graphics
  - Lowest level of using MOLE
  - Use when the most important concern is dynamic, high-speed throughput and visual updates
- Display Lists
  - Use when low-level control is needed but also need decluttering, leadering/stacking, and callbacks
- Force Element / Tactical Graphic Layers
  - Connect directly to back-end GIS
  - Performance determined by feature class

Performance & Complexity

Simplicity & Overhead

Best Practice:
Data Format Considerations

Vector Formats

- Extended Formats
  - OLEDB provider
  - Plug-in data source
- FME
  - Data sources
  - Transformers
- ETL
  - Formats
  - Workflow
  - Performance

Raster Formats

- Extend Raster Formats (Format DLL)
- Pyramids
- Loading rasters

General

- Spatial indexing

Military Analyst

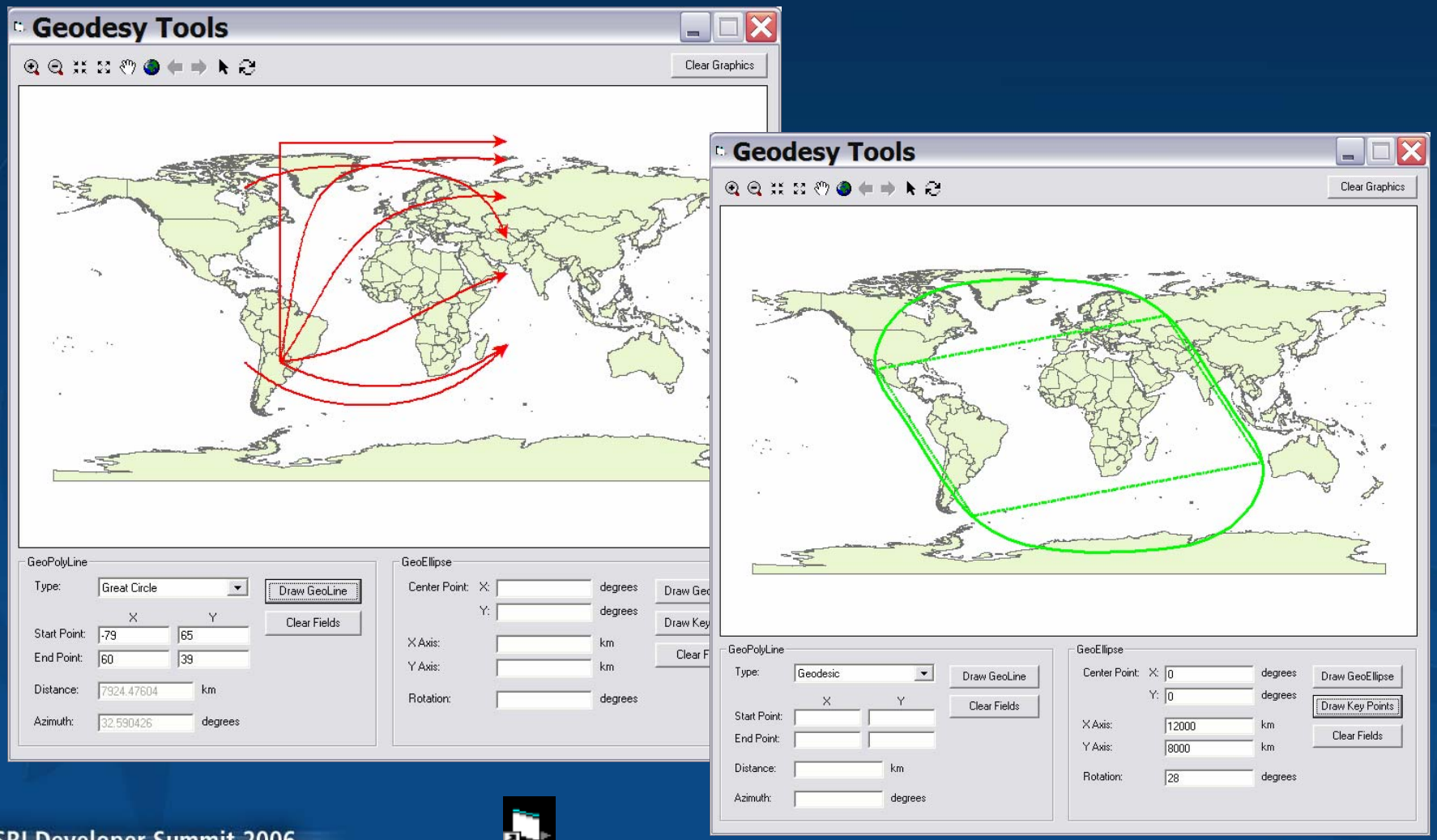
Military Analyst Desktop Capabilities

- Extension to ArcGIS
- Collection of defense-related functionality
- Manage and exploit Military data formats in ArcGIS
- Convert coordinates between DD, DMS, UTM, & MGRS
- Measure geodetically accurate distances
- Analyze terrain data and visualize in 3D
- Builds on core ArcGIS capabilities
- Functionality always increasing according to the needs of the defense/intel user community

Military Analyst ArcEngine Capabilities

- Military Analyst Layer Management
  - VPF
  - RPF
  - DTED
- Coordinate Conversion
- Geodesics
  - Rhumb Line Distances
  - GeoEllipse
  - GeoPolygon
  - GeoPolyline

Military Analyst Application Demo



GeodesyMapControl.exe.lnk

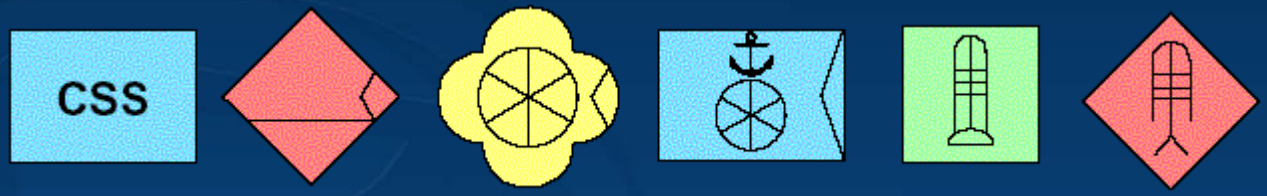
Military Overlay Editor (MOLE)

About MOLE

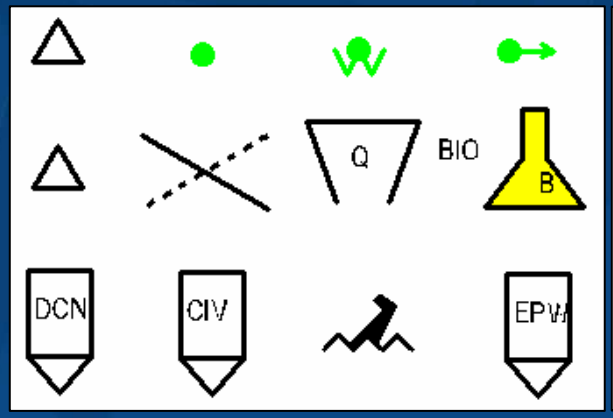
- MIL-STD-2525B war fighting symbology (all Appendices) in the ArcGIS framework
- Designed to easily create, display, edit and manage MIL-STD-2525B war fighting symbology and datasets
- Designed to work with
  - ArcGIS Desktop and Extensions
    - ArcMap, ArcGlobe, ArcReader, ArcIMS, Tracking Analyst,, etc.
  - ArcGIS Engine
  - 2D (ArcMap/MapControl)
  - 3D (ArcGlobe/GlobeControl)

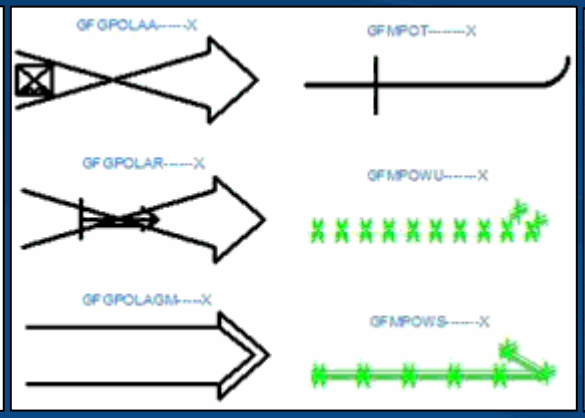
MOLE Terminology

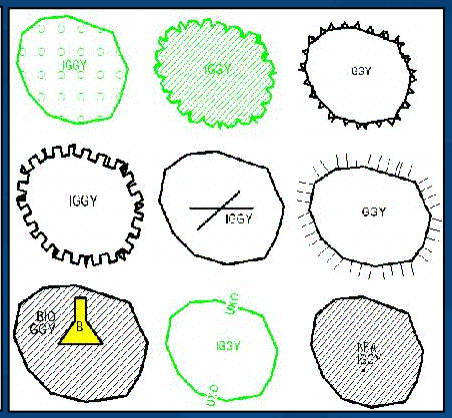
- Force Elements – C2 Symbology Units, Equipment, and Installations



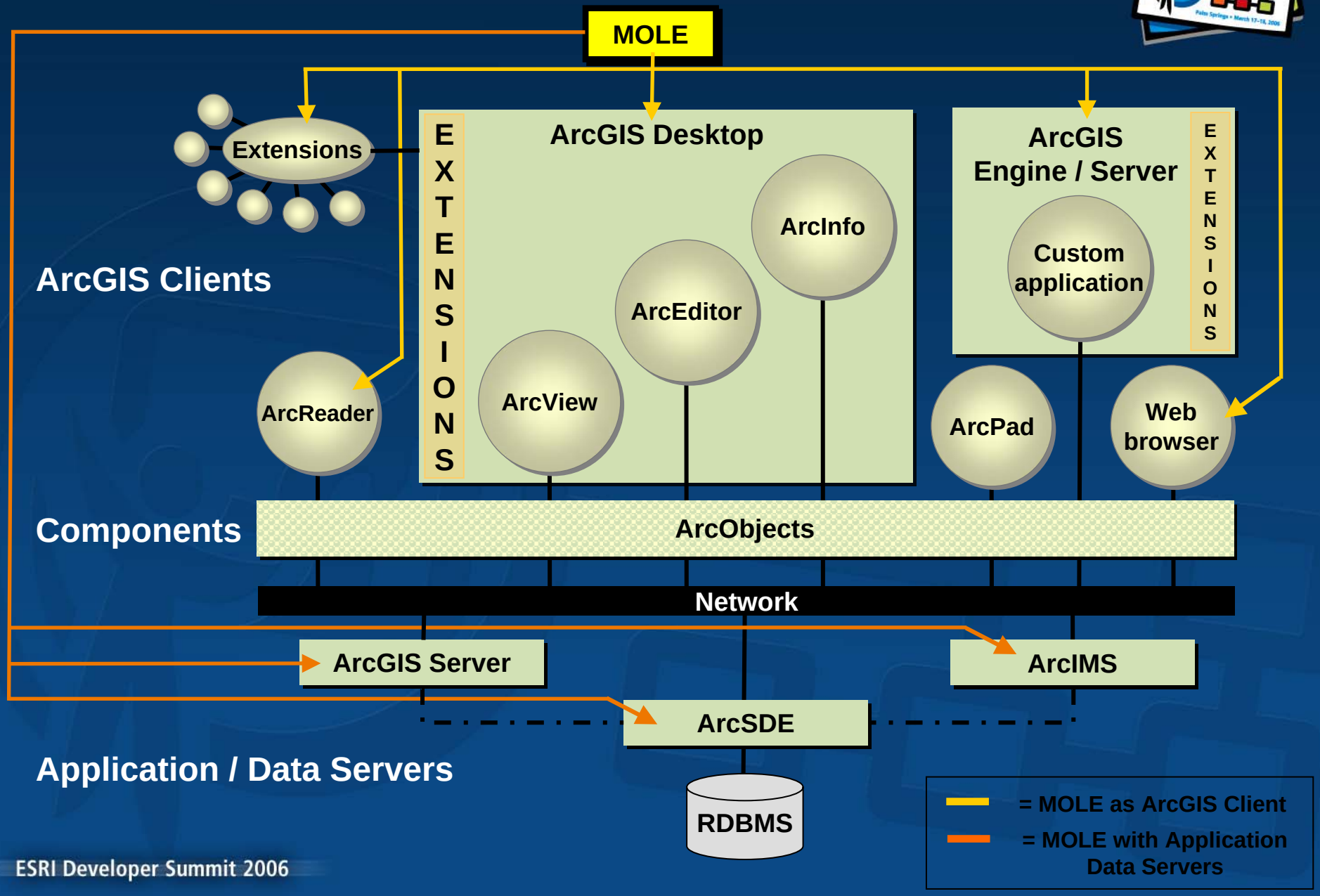
- Tactical Graphics – C2 Symbology Military Operations



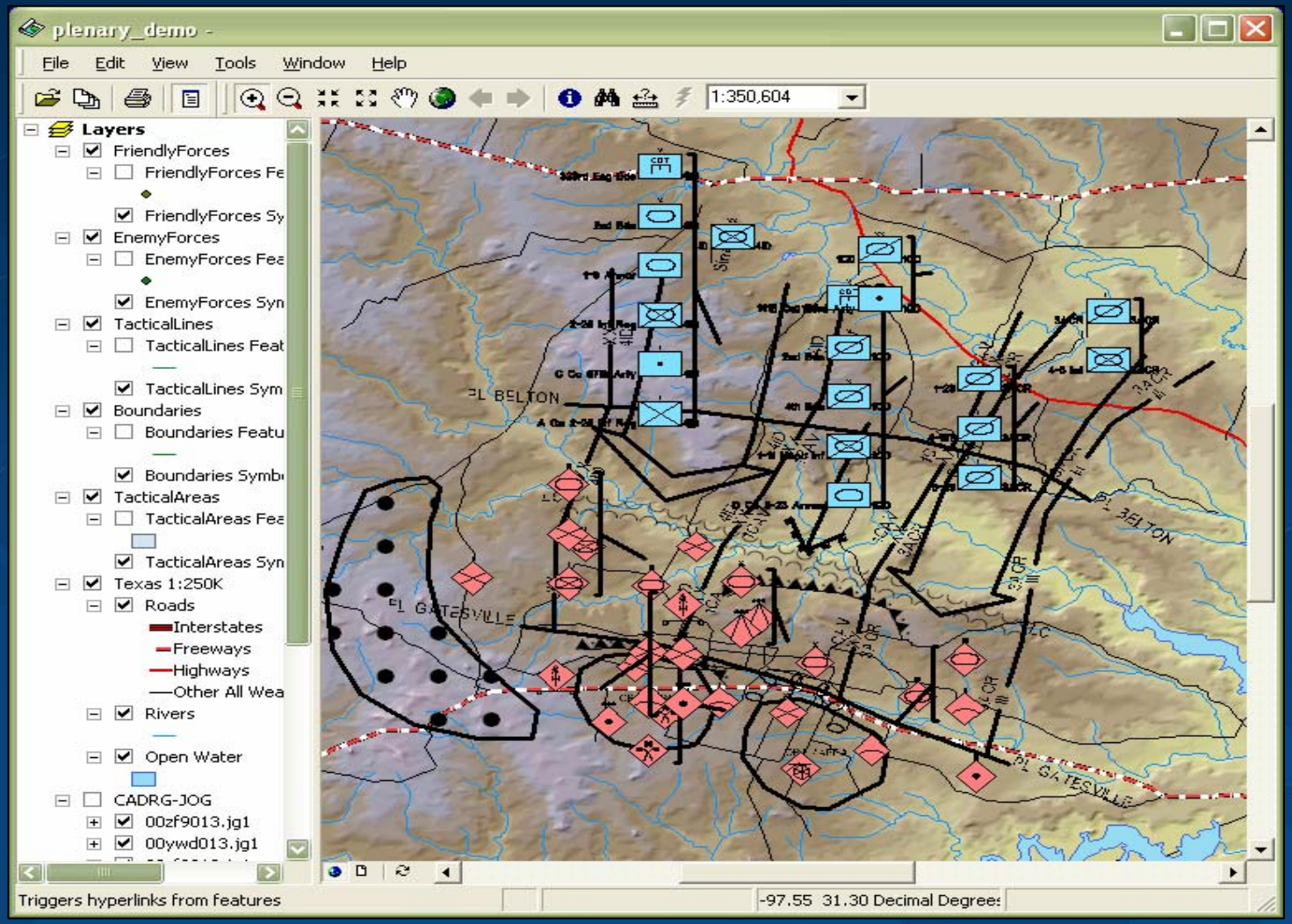




MOLE in the ArcGIS 9 Framework

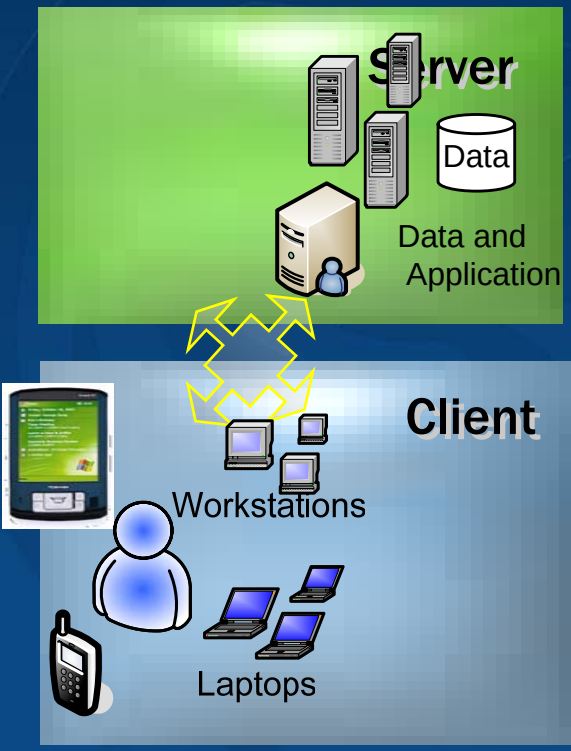


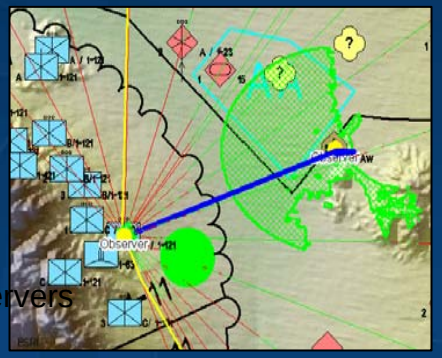
MOLE Application Demo

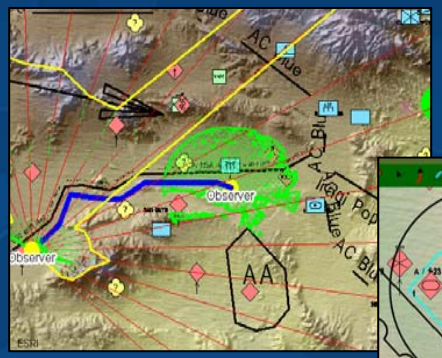


MOLE Demo
ArcGIS Server Application

- Client defines geometry and attributes, sends information to server
- Symbology generation, analysis performed on server, sent to clients
- Client displays common operational picture (COP) and makes and see changes







http://scenario.esri.com/moleeditor/

Sample Reference Application

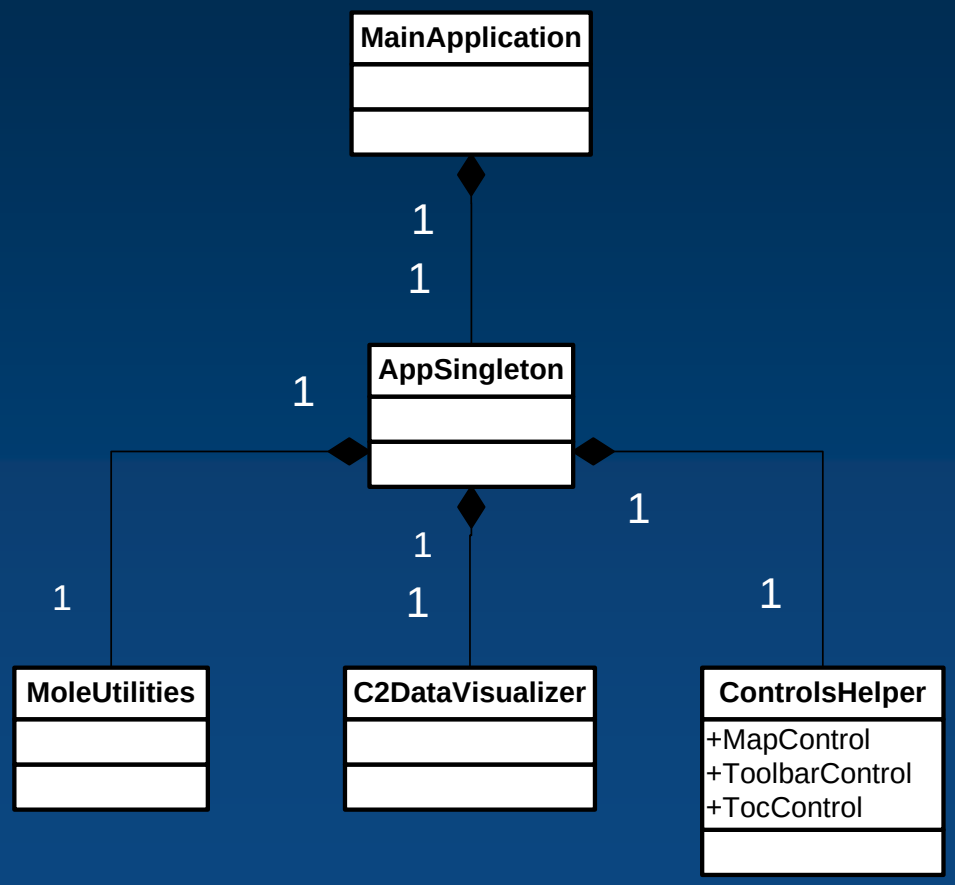
Sample Reference Application Purpose

- Demonstrates
  - Using the ArcGIS Engine Controls in conjunction with MOLE Military Symbology
  - An ArcEngine solution to a fully-functional GIS application for creating, viewing, and exporting GIS data and Military Symbology
  - A solution implemented using several development languages/environments
  - Typical requirements and design alternatives
- Important Caveat: only represents one possible solution for the stated requirements - not the *only* solution

Case Study:
ArcGIS Military Symbology Application

- General ArcGIS Engine Usage
  - Using the ArcGIS Engine Controls.
  - Programatically adding commands and tools to the ToolbarControl and TOCControl .
  - Adding layers to the map and displaying and modifying their properties.
  - Subscribing to ArcGIS Events.
- MOLE Usage
  - Using MOLE to generate a list of valid Symbol ID Codes (SICs).
  - Creating, displaying, and modifying any Force Element and Tactical Graphic symbol.
  - Using MOLE symbols as ICachedGraphics to obtain maximum performance.
  - Exporting MOLE symbols to image files.
  - Dynamically moving Force Element symbols on the map

ArcGIS Military Symbology Application: Architecture



Commands/Tools

AddLayerCommand
C2AddUnitTool
C2MultiSelectTool
DeleteLayerCommand
LayerPropertiesCommand
ZoomToLayerCommand

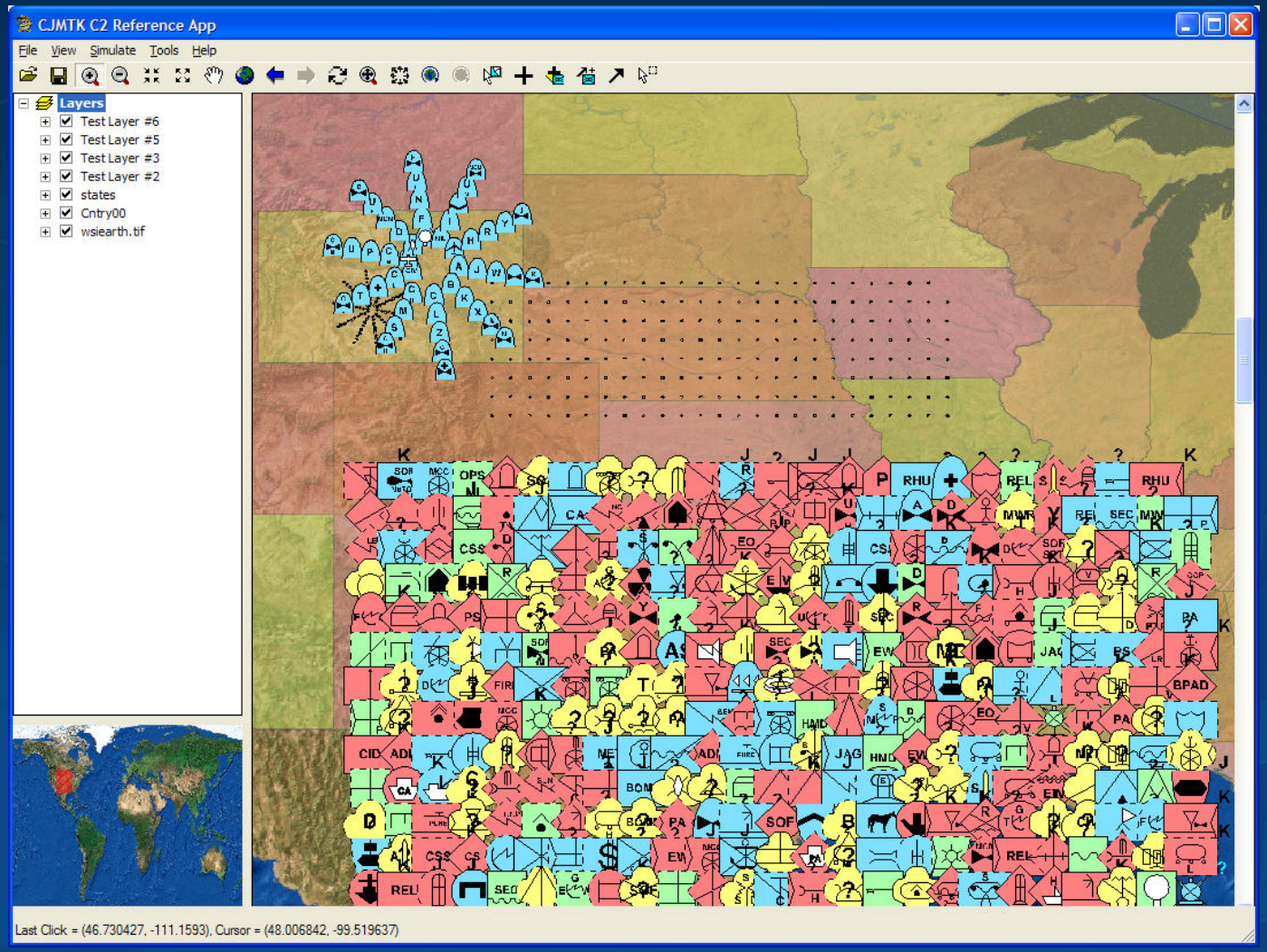
Forms

AddUnitsForm
MultiGraphicSelectForm
LayerPropertiesForm
MoleUnitSettingsForm
MoveUnitsForm
TacticalLayerPropertiesForm

Utility Classes

DrawUtilities
GeneralUtilities
HighResTimer
Logger
MapUtilities
SicHelper
Symbols
TacticalPointsCount

Sample Reference Application Demo



Conclusion

Conclusion

- Many different target and integration options available to defense developers and system integrators
- Some methods are better than others
- Patterns and Practices provide a road map to
  - Aid in climbing the learning curve
  - Exploring existing solutions
  - Navigating options and alternatives
  - Choosing proven best path(s)

Additional Resources

- ESRI Defense Solutions
  - http://www.esri.com/defense
- ESRI Developer Network
  - http://edn.esri.com/
- EDN Code Exchange (Samples and Scripts)
  - http://edn.esri.com/index.cfm?fa=codeExch.gateway
- MOLE and Military Analyst Forums
  - http://forums.esri.com/forums.asp?c=5

????

Questions???

????

Session Evaluations Reminder

Session Attendees:

Please turn in your session evaluations.

. . . Thank you

Backup Slides

MOLE Approaches

- There are essentially three levels at which developers can incorporate MOLE into applications:

- Cached Graphics
- Display Lists
- Force Element Layers

Performance & Complexity

Simplicity & Overhead

Cached Graphics

- Approach
  - Use a MOLE FEGraphicFactory's 'Make' method to create force elements (FEGraphic's / ICachedGraphic's).
  - Hook into the IActiveViewEvents AfterDraw event and manually render the created ICachedGraphic's (ICachedGraphic.Draw)
- Benefits
  - Optimum speed
- Drawbacks
  - Requires most low-level ArcObjects programming of all approaches
  - Only works in 2D (ArcMap)
- When To Use
  - When the most important concern is dynamic, high-speed throughput and visual updates

Display Lists

- Approach
  - Create force elements (FEGraphic's / ICachedGraphic's)
  - Add them to a MOLE ForceElementDisplayList,
  - Hook into the IActiveViewEvents AfterDraw event and render the created ForceElementDisplayList with a single call
- Benefits
  - Allows for some advanced MOLE features such as decluttering, leadering, stacking, and Before/AfterDraw Callbacks
- Drawbacks
  - Requires some low-level ArcObjects
  - Only works in 2D (ArcMap)
- When To Use
  - When some performance can be sacrificed for more features

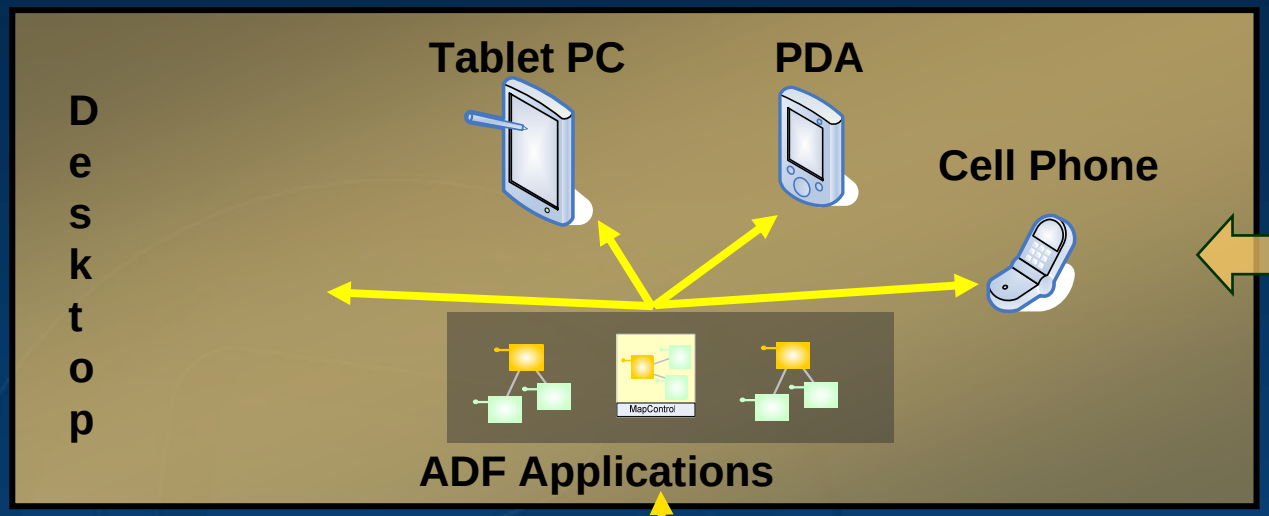
Force Element Layers

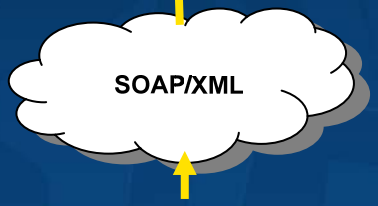
- Approach
  - Load an IFeatureLayer containing MOLE data
  - Attach it to a MOLE ForceElementLayer (and MOLE renderer)
  - Add the ForceElementLayer to the map/globe
- Benefits
  - Requires least developer effort
  - Currently the only approach that works in 3D
  - Only practical approach to use MOLE if data is provided by/connected to a back-end GIS solution (ArcSDE, shapefiles, PGDB, etc.)
- Drawbacks
  - Less atomistic control over the appearance of individual FEGraphics
  - Performance constrained by performance of feature class provider (ArcSDE, shapefiles, PGDB, etc.)
- When To Use
  - When low-level control over individual graphic appearance is not critical, or when in 3D

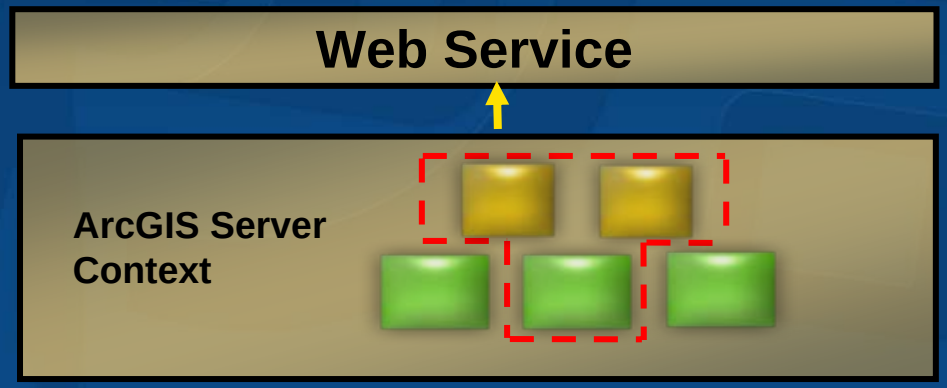
Code Sample
Create a MOLE Force Element

```
//Create a MOLE Symbol as a Cached Graphic
public ICachedGraphic CreateMoleUnit(string SymbolIdCode, IPropertySet pSet, IPoint pPoint,
double Size)
{
   pCachedGraphic   = null;
   IFEGraphic pForceElementGraphic = UnitGraphicFactory.Make(SymbolIdCode);

   pForceElementGraphic.Style = StyleForCreate;
   IForceElement pForceElement = new ForceElementClass();

   pForceElement.PropertySet = pSet;
   pForceElement.Shape = pPoint;
   pForceElement.MessageString = SymbolIdCode;

   pForceElementGraphic.ForceElement = pForceElement;

   ICachedGraphic pCachedGraphic = pForceElementGraphic as ICachedGraphic;
   IGeometry pGeo = pPoint as IGeometry;
   pCachedGraphic.Size = Size;
   pCachedGraphic.Geometry = pGeo;
   pCachedGraphic.IsLocked = false;

   return pCachedGraphic;
}
```

Overview of ADF Architecture







Thick Clients

ArcGIS Engine-based deployment

Smart Clients

Rich-client applications
Leverage web services
Easy to build and deploy

Thin Clients

Web Applications
Browser Applets