

# FORCE STRUCTURE - EXPAND/COLLAPSE

## *A Case Study - Battlefield Object Hierarchy & Symbology*

Version 1.2 final, 5/29/2005

### Authors

Michael Behrens, R2AD, LLC  
Steve Abrams, Cubic Defense Systems

### Abstract

Over the years, a lot of experience has been obtained on the building of an effective Command and Control (C2) system for real-world and training purposes. This document covers the basic force structure elements of C2, leaving the rest for a much larger future C2 specification document. The force structure described here is based primarily on the architecture of the Joint Readiness Training Center ([JRTC](http://www.jrtc-polk.army.mil/))<sup>1</sup> developed by Cubic Defense Systems (CDS).

### JRTC – Battlefield Replication

The JRTC system replicates the battlefield activity for training analysis. It is therefore neither a simulator nor a C2 system. The JRTC battlefield replication system includes force creation, movement, planning, engagement, supply, medical evacuation, indirect fire, weapons of mass destruction, radio communication, digital networks, GPS tracking devices, laser encoded messages, and more. Consequently, anything occurring on or about the battlefield before and during an exercise is captured digitally and available for near-realtime and post-exercise analysis. While the JRTC system is designed for replication, it also interacts with simulation systems [STOWE] and could be modified to support C2 efforts.

The JRTC system is a good example of how real C2 systems like GCCS-J can utilize GPS reporting devices. Program managers from DARPA's CPOF program have recently visited and participated in events at Ft. Polk in order to gain insights and JFCOM participates via the Combined Joint Task Force Exercise (CJTFFEX).

### *Force Model*

For each exercise, forces are generally created from scratch. Forces are given a role such as friend, enemy, neutral, etc. The role is essentially the alignment or attitude of the force from the perspective of the unit being trained. It is used to determine fratricide (damage inflicted by a force of the same role). People on the battlefield are called "players" and they can be added to units. Units can also be added to units, thereby creating a hierarchy. The system also retains the notion of an organic unit, the original parent unit. This is important, since dynamic units are created during a mission (e.g.: a recon platoon or SOF team) and afterwards are returned to their organic parent. Since the system stores this relationship, it is easy to manage force structure while at the same time providing a reporting mechanism on which units are reinforced or have a depleted force strength (which is also true when players are wounded or killed). Unit orders can also be "replicated", so transfers between units for special tasking can be managed just like they would in the field.

Since the real battlefield also has civilian and non-force entities, they can also be created and optionally tracked electronically. During the course of the exercise or any of its missions, it is

---

<sup>1</sup> <http://www.jrtc-polk.army.mil/>

not uncommon for a civilian force to switch roles and become a hostile force or for a neutral force to become a friendly force. This dynamic nature has been taken into account so that the system is able to build force structures, which are comprised of people, vehicles, etc. regardless of originally assignment. For example, a captured enemy truck might be put into service by a friendly unit. Real command and control systems also need to be able to take this into account to prevent fratricides.

The diagram below depicts in UML a force model from PC-RIS (Range Instrumentation System). PC-RIS evolved from JRTC into a PC client-server architecture, retaining the characteristics described above. The C2 Information Exchange Data Model (C2IEDM)<sup>2</sup> provides a similar structure, however the definition of a “force” is the unique concept here.

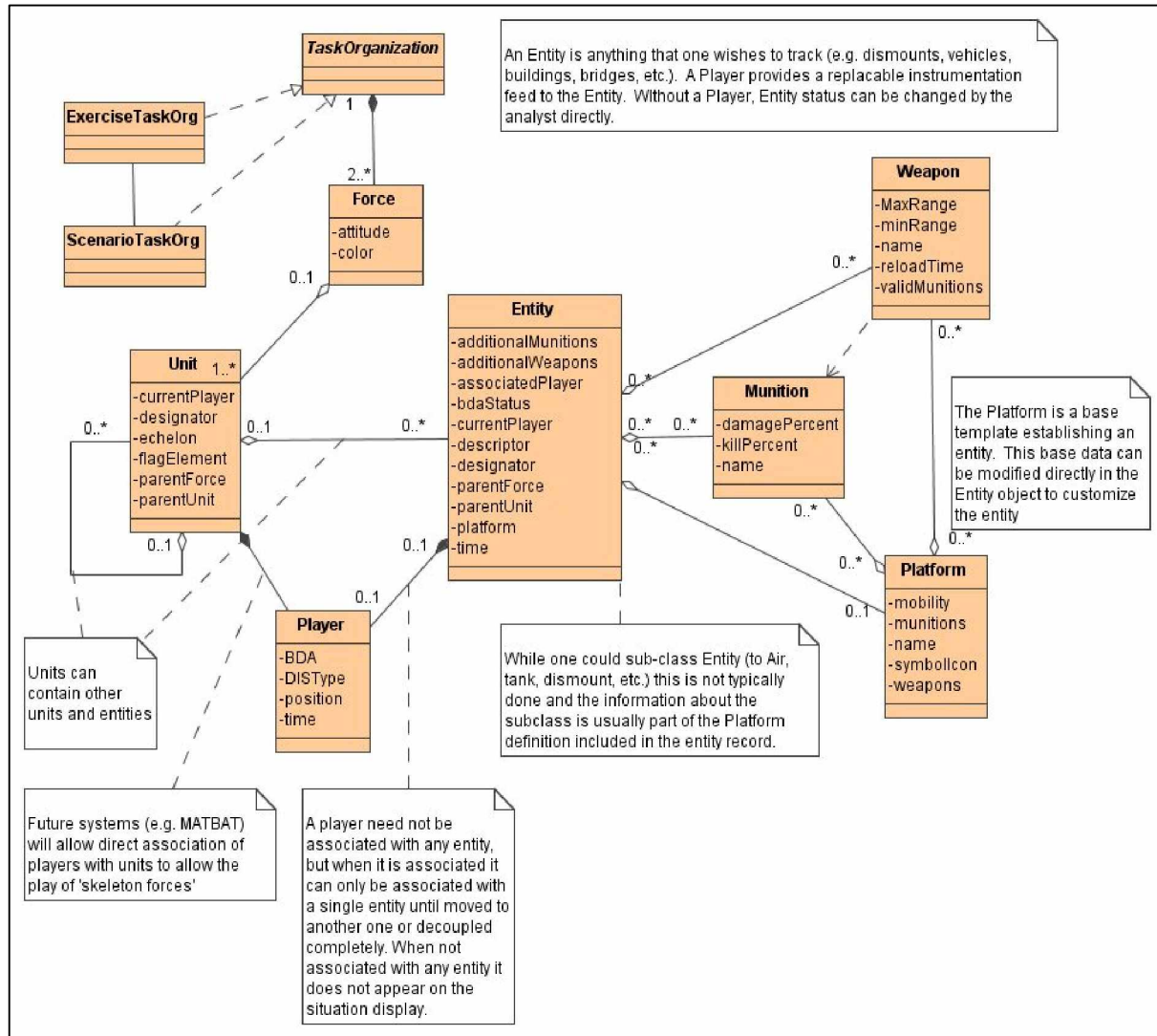


Figure 1 – PC-RIS Entity Structure (approximate). Note: The Player provides a function similar to the GPS devices in the Blue Force Tracking ACTD. A player's movement and actions could also be generated by a simulator.

### Force Location and Events

Electronic tracking and weapon pairing devices are issued to soldiers, mounted on vehicles and weapon systems. This information is mapped (barcode system) to the force structure entities (soldiers or units) so that movement and events can be tracked as the transmitters send in messages.<sup>3</sup> Some force elements can be assigned a “center of mass” so that their displayed position is calculated to be at the center of their subordinate entities. Conversely, elements can be instructed to follow or be around a flag element, which would provide an auto-linked location, which is sometimes very useful.

Entities can also have two positions replicated in the system: Where they report themselves to be and where the GPS system reports them to be. A third position might also be in order: where others think they might be (correlation engines, observers, etc.). The geo-plotted location can be toggled on the map and both locations are shown textually in the pop-up status dialog GUI. This is critical since many lessons can be learned by observing how units report information, call for fire, etc. based on where they “think” they are.

### Filter Controls

The unit’s echelon can be optionally links to map zoom levels (scale ranges, i.e.: 20-30k), causing them to collapse or expand automatically. This could be part of a Level of Detail (LOD) filter, which might also include map features and layers. Also, any unit can be expanded or collapse using either a button on their status GUI or via the right-click menu. Furthermore, individuals can be filtered away using a hierarchical unit force structure tree with a check box. Clicking on the parent unit check box toggles the selection of all subordinates. A traditional view of the order of battle could also be used to control filtering, perhaps by embedded check boxes in the organizational graph.

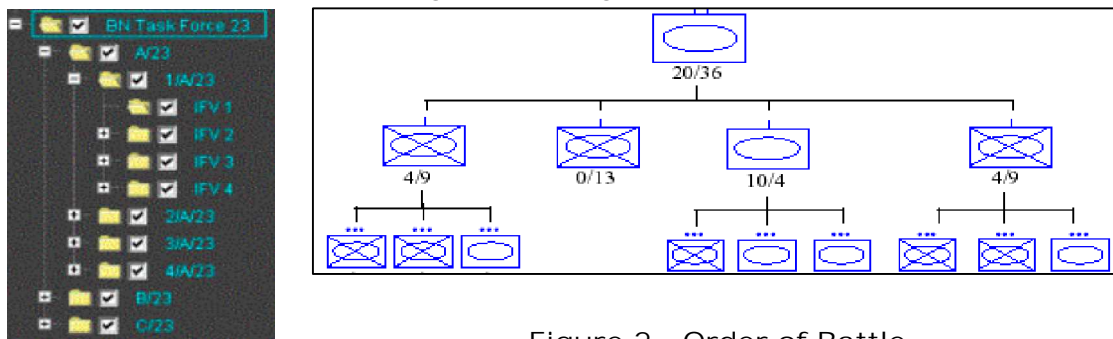


Figure 2 - Order of Battle

	<p>Visualization Filtering:</p> <ul style="list-style-type: none"> <li>• Right-click and select “Expand” or “Collapse” to control visibility filters on a single entity or selected set of entities</li> <li>• Optionally control Organizational graphical lines between parent and subordinate entities</li> <li>• Status dialogs activated via a right or double mouse click contains roll-up counts of healthy, wounded, or killed and also provides control buttons for expand/collapse and details.</li> <li>• Collapse-All and Expand-All controls are very important and are available from the tree view and from the status dialogs.</li> </ul>	
--	--	--

<sup>3</sup> Auto-assignment of transmitter IDs to entities based on prior assignment can help automate the process.







By using more than one map simultaneously or by toggling between filter sets, geo-comparative analysis of map entities can be performed to reveal differences in location and other attributes. Ensuring the availability of pedigree data is critical in order for this level of analysis to be performed.

### Symbology

JRTC allows the dynamic creation of new symbols (symbol editor) and their assignment to players. This supports internationalization among other things, since different countries use different symbols. These symbols were not transmitted, as the system was used within the LAN only. In a Service Oriented Architecture (SOA) frame of design, the symbol definitions would be available as a web service as an association to any entity. This service would provide contextual annotations which can be conveyed to interested parties as XML or rendered by provided SVG, PNG, or other formats.

### BDA

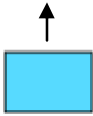


One of the more interesting aspects of JRTC (and CMTC) is how symbology is used to depict Battle Damage Assessment (BDA) and engagement vectors. As indicated in the table below, Personnel (Soldiers) could be healthy, wounded, or killed. Other entities (tanks, equipment) could be Nominal, damaged or destroyed.

Personnel	Healthy/Normal	Wounded	Killed
			
Equipment/Weapon	Healthy/Normal	Damaged radio, mobility, weapon	Destroyed
			

Generally it is better to use shapes rather than color as a method of coding BDA. Future systems could utilize dashed lines so that an accurate status can be obtained in different lighting and printing situations. JRTC/CMTC used yellow and white slashes to represent chemical and nuclear contamination respectively (not shown). Perhaps other mechanisms such as waves, dots, circles, etc. can be exploited. It would be important, in some cases, to ensure that the BDA symbology does not hinder the user's ability to read other annotations on the symbol such as the label or direction, etc.

## Engagement

When a weapon fires, that information is known electronically and disseminated either at the time of firing or afterwards as a synchronization function. Regardless of when the message is processed, replay events are always accurately rendered and can be altered based on new information. An arrow indicator above the entity depicts that the entity is currently firing. If the target is known, perhaps via a range finder or other means, such as the MILES [JRTC] system, then a vector can be rendered to the point of impact.<sup>4</sup> For indirect fire, the target location is generally known and captured via the tactical messages.

<p>A unit Firing, miss or unidentified target</p>	<p>A unit firing and hitting another entity, causing a wound (the wound strike-out would appear at the same time as the vector, then remain).</p>	<p>A unit firing and missing an entity which was previous hit and wounded. Note, just because a hit was made does mean damage has been incurred.</p>
		

Throughout the engagement period, the fire vector is made visible for a second and then made invisible. This gives the viewer a sense of activity and also provides a visual cue that something besides movement is taking place. Engagement vectors can be filtered. BDA can also be filtered.

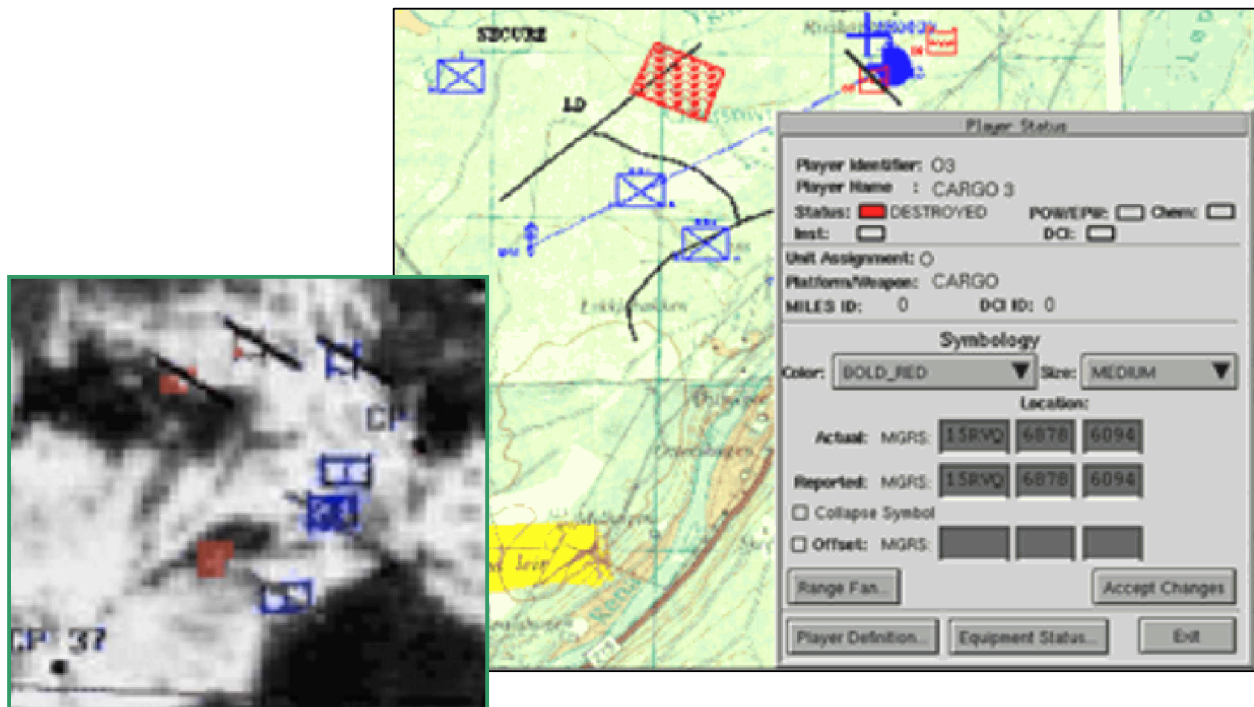


Figure 2 – CMTc engagement, BDA, and status dialog

<sup>4</sup> Note: 2525B uses an arrow to indicate direction of movement which might conflict with this method. Perhaps use of a bezier curve might be a better.

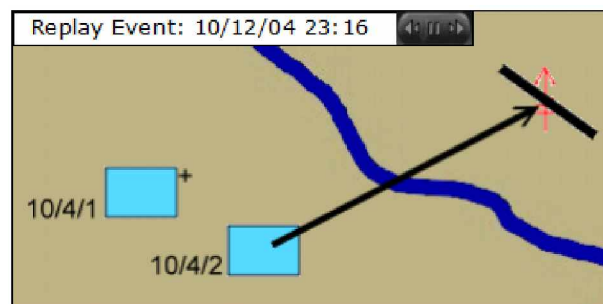


The screen shots below are from a presentation on PC-RIS developed by Cubic Defense Systems. In these actual views, note the firing vectors, BDA, and status dialog (with BDA, locations, and buttons).

## Time Mode

When looking at battle information in any particular GUI, it is important that the time mode be very clear to the operator. The time mode might be real-time, replay, or future (planning). Multiple screens or windows might be depicting different time modes at the same time, so it is important to make it clear by a standard visual theme.

Related to this, if a view is depicting real-time, there must be checks in play to detect if the real-time display becomes latent or out of sync with reality (ground truth). If this happens, the operator must be provided a means to synchronize again without losing any information that has been entered locally (transactions). The display format of the time is configurable.



## Overview First

It is a general Human Computer Interaction (HCI) principle that overview or high-level information should be presented first, followed by details upon request. The use of hierarchy and roll-up is an example of using this principle. Additionally, concepts which are important to the commander can be similarly abstracted such as readiness. The higher level graphics or symbology could indicate a summary of readiness to include force strength and status.

## Intent

The depiction of a unit or aggregate force's intent is often required in order to make decisions. Some changes in the appearance of the symbol such as a jagged edge could be used to depicting intents such as active or passive. Some examples of intent include planning to attack, reinforcing, disbanding, reconnaissance, or returning to base. Generally, the intent should help depict the answer to the question: "Why is that battle element there and what it is up to?".

## Symbology Service

Service Oriented Architecture (SOA) patterns lend itself to symbology as well. A symbology service can provide lookup and modification of symbology depicted within a Community of Interest (COI) such as a task force. Data sources and feeds should never contain a symbol ID since other data elements should always provide enough information to generate or obtain the correct symbol to represent the item(s) in question. The symbol is therefore mapped to the entity based on its attributes. Users have a requirement to augment or modify the symbol based on their knowledge and the symbology service is a way to share that augmentation with others within a controlled scope. These updates would be maintained across time by the symbology service, using the entity key and COI keys provided. Core services such as the Notification Services, ID Service, Security, and Association service would be used in the implementation of the symbology service. Some example interfaces provided by the service would include:

- SetSymbolCode(Entity\_Key, 2525 code)
- SetSymbolAnnotation(Entity\_Key, annotation code, value)
- getSymbolCode(Entity\_Key)
- getSymbolGraphic(Entity\_Key, Graphic\_Enum) (where enum is SVG, GIF, JPEG, etc)

External network applications would make calls to populate the ID to Symbol mapping maintained by the Symbology Service.

### *Generic Aggregation API*

A standard mechanism is needed to convey the hierarchical information to visualizations (i.e.: maps, graphs, org charts, tables, etc.). Creating a standard and simple way within our C2 applications for end-users to control aggregation regardless of where the data comes from is important in order to optimize the user experience.

The key features that need to be fully documented are:

- Hierarchy schema to include subordinates and Siblings (perhaps by type: equipment vs personnel)
- Universally unique ID for Entities, perhaps based on WS-Addressing or [WSRF].
- Multiple "parent of" relations (organic and current)
- Attachment types (operation control, etc).
- Messages/Eventing to control the creation and management of relations

This would then allow for interoperability between application data sources and data visualization modules.

There should be a single set of commands the developer must expose from their internal model to support aggregation visualization:

- Expand
- Collapse (de-aggregate or aggregate)
- ISHierarchical (participates in aggregation)
- GetBDABYType (counts of healthy, damaged, killed) for a status table in the GUI, perhaps by Force/Echelon like PC-RIS

This means that a consistent software model for the behavior must be present. All parties need to agree on the semantics. The XIS software suite provides some of the desired behavior, however developers are reluctant to have dependencies on a particular library. Therefore, this paper is proposes a non-library specification that relies on introspection to determine if the appropriate behavior is available in any software object.

Introspection also works on XML documents. Key patterns could be agreed upon to formulate a specification that everyone could follow. For instance, objects that are hierarchical can provide either metadata or the signature "isHierarchical" method. The specification could also state that an interface be implemented, however that is difficult to know when interrogating an XML stream or XQuery result, for instance. Also, many existing hierarchical schemas exist already, including what is present in C2IEDM. The key is that the namespace for the elements need to be standardized and registered via the DoD XML registry.

### **Drag-N-Drop and Cut-N-Paste**

Drag and Drop typically transfers full objects or information in an format which is agreed ahead of time by the involved GUI components. However if visualization components could export and import an XML document for drag and drop operations, then an API neutral mechanism can be adopted to achieve a greater degree of interoperability between and amongst software visualizations components. It is recommended therefore that during the drag protocol, the following mechanisms be recognized and supported (perhaps in this order):

- simple text (as would be the case between two different text fields)
- XML Document (schema to be agreed upon – adoption is important)
- Object (for pure native Java or .NET object support)

- File (for transferring file between the application and the OS)

### Data identification

In an ideal computing world, everything would have a unique (universal) tactical ID that could be used to trace the object to its origin if needed and would be resolvable to identify each entity across all repositories.

In the network centric world, it is recommended that an distributed ID service be utilized to perform the following functions:

- reservations of ranges of IDs for local allocation
- Lookup of ID to obtain its origin (where it was created, when, etc).
- Replication, so that the ID service could work in concert in a global network
- Alias registration so that a global ID can be resolved to a local ID
- Support permanent IDs and more temporary ones which would have a lease.

An ID can be wrapped in a reference abstraction that could be later resolved to obtain the actual ID. Such as abstraction layer is important since it supports a variety of ID representations and also since the actual ID of something might change over time, the reference is safer in that it assumes that possibility, requiring the reference to be resolved before it is used. At least then the user of the reference would know that the ID is now longer valid.

This sort of work is being standardized as part of WSRF "Renewable References" in the OASIS community and by WS-Addressing in the W3C. It is recommended that the outcome of the GGF work be adopted. There are others as well.



## *Terminology and Concepts*

The following definitions outline the terminology and usage in this paper.

**Entity:**

A unit, player, vehicle, or weapon system found on the battlefield.

**Unit:**

An aggregate container for entities, which can be players or other units.

**Organic Structure:**

The home unit of an entity. When an entity is no longer tasked, then they return to the home unit.

**Player:**

A person (soldier or non-soldier) in the exercise being tracked and belonging to a unit.

**XQuery:**

XQuery is a standard query language, published by the W3C (World Wide Web Consortium), that uses XML (Extensible Markup Language) notation to define query requests and handle query results.

## *Acknowledgements*

Special Thanks to the software managers and engineers of Cubic Defense Systems for developing state of the art training systems, which has helped the U.S Military and our allies. Thanks for 2525B and other HCI input from Dr. Kathy Fernandes, SPAWAR.

## *References*

**Patrick E. Clarke, "Replicating a Grim Reality"**

[http://www.mt2-kmi.com/archive\\_article.cfm?DocID=664](http://www.mt2-kmi.com/archive_article.cfm?DocID=664)

**William E. Bohman, "STAFFSIM, AN INTERACTIVE SIMULATION FOR RAPID, REAL TIME COURSE OF ACTION ANALYSIS BY U.S. ARMY BRIGADE STAFFS", June 1999  
NAVAL POSTGRADUATE SCHOOL**

**[JRTC]**

<http://www.jrtc-polk.army.mil/>

<http://www.peostri.army.mil/PM-TRADE/ICD/jrtc/>

**[STOWE]**

**Synthetic Synthetic Theater of War—Europe**

<http://www.wws.princeton.edu/cgi-bin/byteserv.prl/~ota/disk1/1995/9512/9512.PDF>

<http://tradoc.monroe.army.mil/historian/pubs/TRADOC25/chap3.htm>

**[C2]**

[http://www.journal.forces.gc.ca/engraph/Vol3/no1/pdf/53-64\\_e.pdf](http://www.journal.forces.gc.ca/engraph/Vol3/no1/pdf/53-64_e.pdf)

**[WSRF] WS-Resource Reference**

<http://docs.oasis-open.org/wsrp/2004/11/wsrp-WS-Resource-1.2-draft-02.pdf>

**[Naming]**

<http://lists.oasis-open.org/archives/wsrp/200412/pdf00002.pdf>

## Other Notes

- Not directly related to Force, however for a future papers/proposals:

## GPS Time

*All clients need to know the correct time. Timestamps are used to synchronize events emanating from various computers. Zulu time should generally be used internally. Servers should connect to a GPS receiver to update their systems. In some cases, updating the computers internal clock is not an option, as it affects real-time system performance. Therefore, each system can maintain an offset (positive or negative) from real-time to ensure that timestamps that are being processed are generated using a combination of GPS time and the offset. A webservice for real-time can be registered as a local service. Synchronization using NTP via GPS or other time sources is of course another option.*

## Search

*Both internal and external systems and users have a need to find data. Keeping data in a database with a propriety schemas does not help solve the information-sharing problem. Data providers could implement and expose interfaces that support external search engines. An example of this would the production of RSS files or other publicly visible and accessible documents that can be consumed by powerful enterprise level network search appliances (ala Google).*

© 2005 R2AD, LLC Corporation. All rights reserved. No other rights are granted by implication, estoppel or otherwise.

R2AD is a registered trademarks of R2AD, LLC in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Permission to copy and display this "FORCE STRUCTURE - EXPAND/COLLAPSE" Whitepaper ("this Whitepaper"), in any medium without fee or royalty is hereby granted, provided that you include the following on ALL copies of this Whitepaper, or portions thereof, that you make:

1. A link or URL to this Whitepaper at this location.
2. This Copyright Notice as shown in this Whitepaper.

The information contained in this document represents the current view of R2AD, LLC Corporation on the issues discussed as of the date of publication. Because R2AD must respond to changing market conditions, it should not be interpreted to be a commitment on the part of R2AD, and R2AD cannot guarantee the accuracy of any information presented after the date of publication.

R2AD, LLC MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AS TO THE INFORMATION IN THIS DOCUMENT.

R2AD may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from R2AD, LLC, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a). THIS WHITEPAPER IS PROVIDED "AS IS". R2AD, LLC, BiblioTronix, LLC (COLLECTIVELY, THE "COMPANIES") MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THIS WHITEPAPER ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS. THE COMPANIES WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THIS WHITEPAPER.