Recently, researchers proposed Collaboration Support Systems (CSS) as a new class of collaboration technology with which a collaboration engineer could package collaboration expertise with collaboration technology in a form that non-experts could use to successfully execute a work practice without extensive training on either the technology or the techniques.
To: technicalreports@afosr.af.mil

Subject: Final Performance Report

Contract/Grant Title: BattleSpace ActionCenters

Contract/Grant#: FA9550-08-1-0263

Reporting Period: 15 May 2008 to 14 May 2011

Project Accomplishments

Intellectual Achievements:

- Created a taxonomy of Collaboration Technology used as a foundation for process support systems (PSS) component capabilities.
- Derived a 7 layer model of collaboration which was adopted by USSTRATACOM’s GISC as the foundation for designing and measuring large-scale multi-agency collaborative intelligence analysis exercises. Derived the inductive and deductive logical foundations for the Seven Layer Model of Collaboration.
- Developed conceptual foundations for the Universal Data Model
- Created an ontology for collaboration processes and systems
- Defined generalized requirements for Collaboration Support Systems (CSS) and for the primary CSS sub-systems, Computer Assisted Collaboration Engineering (CACE) and Process Support Systems (PSS).
- Created first draft of Collaboration Engineering Modeling Language (CEML) standard

Technical Achievements:

- Conducted three international requirements workshops for architectural design of Computer Assisted Collaboration Engineering (CACE) and PSS
  - Created a server-side data architecture around a Universal Data Model
  - Created a collaborative messaging architecture.
  - Created a browser-based client architecture for CACE
  - Created conceptual class architecture for ActionCenters
  - Established developer coding standards for CACE/PSS
- Specified data requirements for ActionCenter elements
- Created conceptual UI/UX designs for Agenda Editor, Activity Editor, ActionCenter Builder, and Screen Editor.
- Implemented CACE framework, Project Explorer, Security, Administration, User Management, and Recycle Bin for the CACE tool
- Implemented three technical innovations that should reduce development cycles for collaboration software:
  - Universal Data Model (UDM). A server that accepts any data in any set of relationships in real time without having to revise server-side metadata.
- Dynamic Contribution Channels (DCC). Clients receive metadata at run time, and instruct the server what events to publish.

- Developed the ARCADE API (descended from the CSAPI) as the foundation for the ActionCenters CACE. Allows for the group-enabling of events with one or two lines of code. A pilot project based on the ARCADE API demonstrated development cycle reductions of 90% and more for collaborative components.
- Developed single, double, and triple indirector capabilities to disambiguate complex relationships among contributions to a collaborative system.
- Produced the first fully functional ActionCenter prototype for a military Course of Action Development process using the UDM, DCC, and CSAPI capabilities in order to exercise the back-end capabilities and to test UI concepts for ActionCenters.
- Implemented a Computer Assisted Collaboration Engineering (CACE) rapid development environment with an open, extensible architecture that accepts plug-in collaborative design and configuration editors, and plug-in collaborative components.
- Implemented editors and components and plugged them into the CACE framework:
  - Project Explorer
  - ActionCenter Builder
  - Element Editor
  - Palette Explorer
  - Activity, Standard Activity, Screen, Tool, Control, and Image components
  - Roster Component
  - Navigation Component
  - Outliner Component and derivatives:
    - List Component
    - Commenter Component
    - Annotation Component
  - Parent-Child Shell Component
  - File Repository Component
  - Symbol-and-arrow Diagram Component
- Implemented a Standard Activity insertion capability for runtime ActionCenters.
- Developed a collection of standard Collaborative Tools from ActionCenters components:
  - Outline Builder
  - Idea Swapper
  - Idea Organizer
  - Idea Presenter
- Produced several prototype ActionCenters applications the new CACE capability:
  - COA Development
  - Creative Problem Solving
  - Marketing Focus Groups
  - Risk Assessment
  - Commander’s Intent
Archival publications (published) during reporting period:

Journal Articles:


5. KOLFSCHOTEN, G.L. FRANTZESKAKI, N. DE HAAN, A. VERBRAECK, A. (2008), Collaborative modeling lab to increase learning engagement, Journal of the Higher Education Academy Engineering Subject Center, Volume3 issue2, pp.21


**Conference Proceedings:**


Science Research, Third Research Seminar of The Center For Collaboration Science, 27 February 2009, University of Nebraska


Book (Chapters):

Conference Abstracts:

Electronic copies of these publications are available upon request.

Changes in research objectives, if any: None

Change in AFOSR program manager, if any:

Dr. David Luginbuhl left AFOSR, and no other program manager was assigned to the project.
Extensions granted or milestones slipped, if any: None

Include any new discoveries, inventions, or patent disclosures during this reporting period (if none, report none):

None
Appendix A
Source Code Listing for the ActionCenters Collaboration Support Systems

The Source code for the ActionCenters prototype is too large to include as an e-mail attachment. It can therefore be retrieved from this URL:

https://actioncenters.svn.sourceforge.net/svnroot/actioncenters/trunk/AC/
Appendix B

Unpublished Technical Paper

Indirectors to Support Complex Relationships in the Universal Data Model for Collaboration Technology

A. Mametjanov, Douglas Kjeldgaard, Robert Briggs
Center for Collaboration Science, University of Nebraska at Omaha
{amametjanov, dkjeldgaard, rbriggs@unomaha.edu}
Indirectors to Support Complex Relationships in the Universal Data Model for Collaboration Technology

A. Mametjanov, Douglas Kjeldgaard, Robert Briggs
Center for Collaboration Science, University of Nebraska at Omaha
{amametjanov, dkjeldgaard, rbriggs@unomaha.edu}

Keywords: Computer-Assisted Collaboration Engineering, Collaboration Support Systems, Collaboration middleware, Universal Data Model.

1 Introduction
Recently, researchers proposed Collaboration Support Systems (CSS) as a new class of collaboration technology with which a collaboration engineer could package collaboration expertise with collaboration technology in a form that non-experts could use to successfully execute a work practice without extensive training on either the technology or the techniques [1]. To speed the development of CSS capabilities colleagues proposed and tested a middleware API for CSS development that encapsulates many of the difficult-to-build aspects of collaboration technology in the form of abstractions that programmers can reuse to cut development cycles for real-time collaborative components by about 90% [2]. The API sits on a universal data service that can accept any form of data in any set of relationships in real time without having to modify the metadata on the backend data store. The data store consists of a fixed set of tables organized around entities, attributes and relationships. The middleware abstractions provide a generic graph-based data manipulation of entities as nodes and relationships as edges. The conception of the universal data service assumed a hierarchical data model where any data object could be uniquely determined by a given set of relationships to superior objects.

There are, however, use cases where more-complex relationships are required. It may be useful for the same object to appear as a subordinate to several superior objects. Changes to the object in one context would be reflected in all other contexts where it appears. Consider, for instance a risk assessment scenario where a single control (a subordinate) could mitigate several risks (superiors), or a software engineering context where a single feature could fulfill several requirements. In some cases, it is valuable for an object to have the same subordinates in all contexts where it appears. In the risk assessment scenario, for example, it would be useful for the strengths and limitations (subordinates) associated with a control (superior) to appear with it under every risk that it mitigates. In other cases, it may be valuable for an object to appear with different subordinates in each context where it appears. In the software requirements scenario, for example, it would be valuable for a feature to appear with a different justification under each requirement that it fulfills.

The goal is to create among all appearances of the entity a sharing of attributes and/or superior/subordinate entities. This paper proposes a data sharing solution using indirectors within the previously proposed Universal Data Model (UDM).

2 Universal Data Model
To set the scene for the paper, we begin with a brief overview of the UDM [2]. Figure 1 summarizes the core of the model. In particular, entities define data objects, which can have zero or more attributes and can be related to other data objects with zero or more superior/subordinate relationships.
The primary benefit of the model is its generic nature. The same fixed set of tables can describe arbitrary relational data. Combined with graph-based data manipulation API, the same UDM-based data store can be used by multiple front-end client platforms to create new entities and relate them to existing data [2].

3 Indirectors in the UDM
The context of an entity in the UDM-based data store is determined by the entity’s superiors. If the same entity with its subordinates needs to be used in a different context, then a new superior could be added to the entity. However, addition of new superiors may alter the existing context producing unintended side effects. For example, in CACE-based collaborative session screen design settings [1], if a certain screen object S with its subordinates, which define the screen’s configuration, needs to be shown for a user in role R1 during collaborative activity A1, then this context can be described by a set of relationships where S is a subordinate of R1 and A1: i.e. R1 → S and A1 → S. If the same screen S needs to be displayed to users in role R2 during activity A2, then object S could be assigned two new superior entities: R2 → S and A2 → S. However, this produces an unintended side-effect of making screen S visible for users in role R2 during activity A2, because S is now their subordinate, too.

This side effect could be eliminated by cloning S as S’ [3] and making clone S’, instead of S, a subordinate of R2 and A2. Cloning, however, is inefficient because it duplicates data and creates referential integrity problems [4] when data in the original or the clone changes.

4 Proposed Solution
Whenever there is a need for an entity to appear in more than one context, we create a new induktor entity with a ‘cloneof’ relationship to the original entity. The induktor gains a) a property containing the identity of its original; b) a flag property for whether it inherits its original’s superior relationships; and c) a flag property for whether it inherits its original’s subordinate relationships. Any properties set on the induktor would override similar properties of the original. Any properties not set on the induktor would be inherited from the original.
Table 1. Exhaustive enumeration of indirector types. Each configuration describes whether the indirector has (a) the same superiors, (b) the same subordinates and (c) the same properties.

<table>
<thead>
<tr>
<th>##</th>
<th>Configuration</th>
<th>Same superiors</th>
<th>Same subordinates</th>
<th>Same attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Reflection</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Masked Reflection</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Twin</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Masked Twin</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Partner</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Masked Partner</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Understudy</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Masked Understudy</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 1 summarizes all possible configurations of shared data indirectors. Configuration 1, named “Reflection,” corresponds to full sharing of relationships and attributes, which would be useful in cases where multiple instances of a subordinate object comprise a superior object. In case 2, “Masked Reflection,” indirectors share relationships, but the attributes of the original object may be localized on the indirector without modifying the values of the original object, which can be useful, for example, for software localization/internationalization. In case 3, “Twin” an indirector shares superiors with its original, but has distinct subordinates, which can be useful in describing a distinct configuration for the same entity, or for creating different subordinate structures in different contexts. Case 4, “Masked Twin,” allows for localization of a Twin indirector. Case 5, “Partner,” defines a distinct manifestation of the same object in multiple contexts with the same subordinate structure in each context. This would be useful, for example, in defining the screen aliases, discussed in the previous section, by making the indirectors subordinate to the distinct superiors and connecting the indirectors to the original object and thereby sharing the original’s subordinates and properties. Case 6 allows localization of a Partner indirector. In Case 7, the indirectors share properties, but have their own distinct superiors and subordinates, which can be useful, for example, in defining the same object in multiple contexts and multiple configurations. Finally, Case 8, “Masked Understudy” allows for the localization of attributes for Understudy Indirectors.

References
Appendix C
Earlier Annual Progress Reports

Annual Report 2008-2009

To: technicalreports@afosr.af.mil

Dr. David Luginbuhl, david.luginbuhl@afosr.af.mil

Subject: Annual progress Statement to Dr. David Luginbuhl

Contract/Grant Title: BattleSpace ActionCenters

Contract/Grant#: FA9550-08-1-0263

Reporting Period: 15 May 2008 to 14 May 2009

Annual accomplishments:

Intellectual Achievements:

- Derived a 7 layer model of collaboration which was adopted by USSTRATACOM’s GISC as the foundation for designing and measuring large-scale multi-agency collaborative intelligence analysis exercises.
- Created a taxonomy of Collaboration Technology used as a foundation for process support systems (PSS) component capabilities.

Technical Achievements:

- Conducted three international requirements workshops for architectural design of Computer Assisted Collaboration Engineering (CACE) and PSS
  - Created a server-side data architecture around a Universal Data Model
  - Created a collaborative messaging architecture.
  - Created a browser-based client architecture for CACE
  - Created conceptual class architecture for ActionCenters
  - Established developer coding standards for CACE/PSS
- Created first draft of Collaboration Engineering Modeling Language (CEML) standard
- Specified data requirements for ActionCenter elements
- Created conceptual UI designs for Agenda Editor, Activity Editor, ActionCenter Builder, and Screen Editor.
- Implemented CACE framework, Project Explorer, Security, Administration, User Management, and Recycle Bin for the CACE tool

Operational Achievements

- Observed warfighters in mission-critical collaborative operations:
- Multi-agency crisis response exercise (Iowa & Nebraska Army & Air National Guard, and 40 civilian law enforcement, government, and NGO organizations)
- Multi-agency intelligence analysis limited objective experiment (USSTRATCOM, NORTHCOM, CENTCOM, CIA, FBI & other civilian and military agencies)

Archival publications (published) during reporting period:

**Journal Articles**


27. Kolfschoten, G.L. Frantzeskaki, N. de Haan, A. Verbraeck, A. (2008), Collaborative modeling lab to increase learning engagement, Journal of the Higher Education Academy Engineering Subject Center, Volume3 issue2, pp.21


**Conference Proceedings**


Book Chapters


**Conference Abstracts**


Electronic copies of these publications are available upon request.

**Changes in research objectives, if any:** None

**Change in AFOSR program manager, if any:** None

**Extensions granted or milestones slipped, if any:** None

**Include any new discoveries, inventions, or patent disclosures during this reporting period (if none, report none):** None
Annual Report 2009-2010

To:  technicalreports@afosr.af.mil

Dr. David Luginbuhl, david.luginbuhl@afosr.af.mil

Subject: Annual progress Statement to Dr. David Luginbuhl

Contract/Grant Title: BattleSpace ActionCenters

Contract/Grant#: FA9550-08-1-0263

Reporting Period: 15 May 2009 to 14 May 2010

Annual accomplishments:

Intellectual Achievements:

- Derived the inductive and deductive logical foundations for the Seven Layer Model of Collaboration.
- Developed conceptual foundations for the Universal Data Model

Technical Achievements:

- Implemented three technical innovations that should reduce development cycles for collaboration software:
  - Universal Data Model (UDM). A server that accepts any data in any set of relationships in real time without having to revise server-side metadata.
  - Dynamic Contribution Channels (DCC). Clients receive metadata at run time, and instruct the server what events to publish.
- Produced the first fully functional ActionCenter prototype for a military Course of Action Development process using the UDM, DCC, and CSAPI capabilities in order to exercise the back-end capabilities and to test UI concepts for ActionCenters.
- Implemented a Computer Assisted Collaboration Engineering (CACE) rapid development environment with an open, extensible architecture that accepts plug-in collaborative design and configuration editors, and plug-in collaborative components.
- Implemented the following editors and components and plugged them into the CACE framework:
  - Project Explorer
  - ActionCenter Builder
  - Element Editor
  - Palette Explorer
  - Roster Component
  - Navigation Component
- Produced the first simple prototype ActionCenter using the CACE capability.
Archival publications (published) during reporting period:

Journal Articles:


KOLFSCHOTEN, G.L., VREEDE, G.J. DE, PIETRON, L.R. (IN PRESS), A training approach for the transition of repeatable collaboration processes to practitioners, *Group Decision & Negotiation*.


Conference Proceedings:


Book (Chapters):


Conference Abstracts:
None

Electronic copies of these publications are available upon request.

Changes in research objectives, if any: None

Change in AFOSR program manager, if any: None

Extensions granted or milestones slipped, if any: None

Include any new discoveries, inventions, or patent disclosures during this reporting period (if none, report none): None