



Collaborative Editing Systems and their Application to Distributed Software Engineering

Jon A. Preston

PhD Student

Department of Computer Science

Georgia State University

Interim Department Head

Department of Information Technology

Clayton College and State University



Thinking Collaboratively

“Great things can happen, when you don’t care who gets credit.”

- Mark Twain



Agenda

- CSCW
 - Synchronous & Asynchronous
 - Virtual Presence and HCI Issues
 - Transparent & Aware Models
- Distributed Software Engineering
 - Configuration Management
 - Distributed CMS
- Well-known Problems
 - Cache Coherency
 - Mutex
 - Merging
- Recent Work
- Open Research



Computer Supported Collaborative Work (CSCW)



CSCW Defined

- Computer Supported Collaborative/Cooperative Work
- Focus on collaboration (HCI)
- Focus on cooperation (IS)
- Goal-driven (work)
- Computer systems support



Synchronous and Asynchronous

- Users can edit at different times
 - Asynchronous
- Users can edit at the same time
 - Synchronous
- Patterns in collaboration
 - Begins asynchronous (email sufficient)
 - Migrates to synchronous (deadline)



Virtual Presence

- Provide information about other users in the system
 - Name, picture, email, “state,” chat, VoIP
- Display what sections of the document others are editing
 - Show “ownership”
 - Show historical information



Other HCI/Social Issues

- Allow users to not be bothered
 - Allow “deep” work
- Integrate communication tools into IDE
 - “Eclipse” by Cheng et al. (2004)
- Shared and personal views of collaborative space



Transparencies

- CSCW system knows nothing about the software “underneath”
- Useful to support legacy systems
- Significant overhead
 - CSCW system is not coupled to the user software
- Capture and broadcast UI events



Aware CSCW

- Tightly-coupled CSCW systems to shared program
- Can be optimized
 - CSCW system knows about collaborative software
- Cache events and broadcast what is important

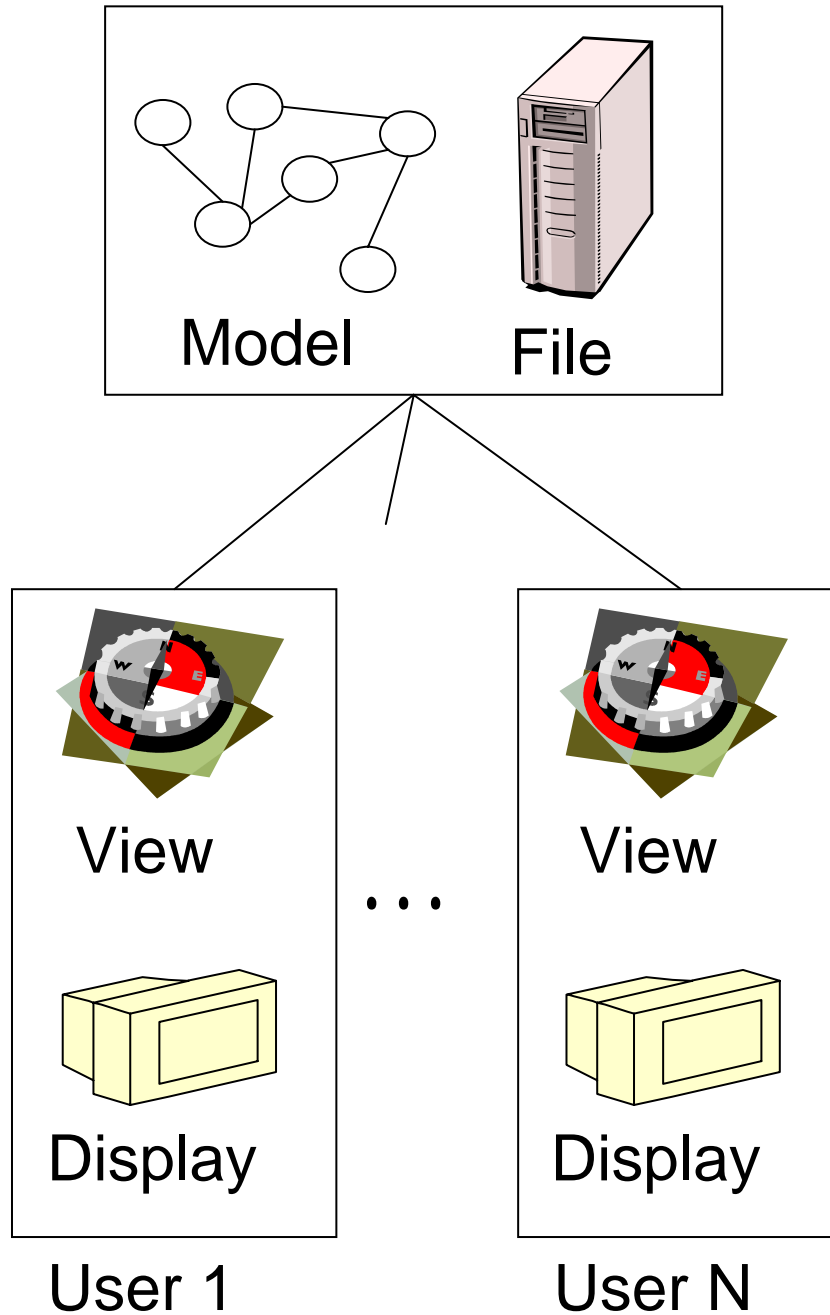


Levels within Groupware

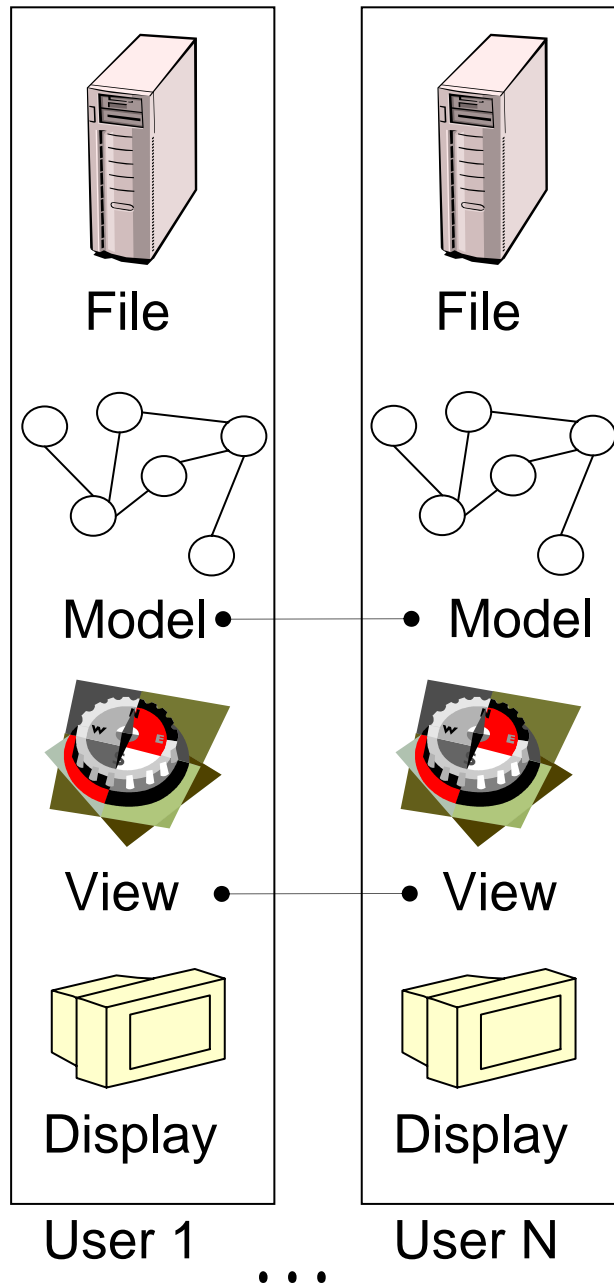
- Display
 - Renders the application to the user
- View
 - Contains the application's logical presentation
- Model
 - The application's state and internal information
- File
 - The persistent information of the application

Roth and Unger(2000)

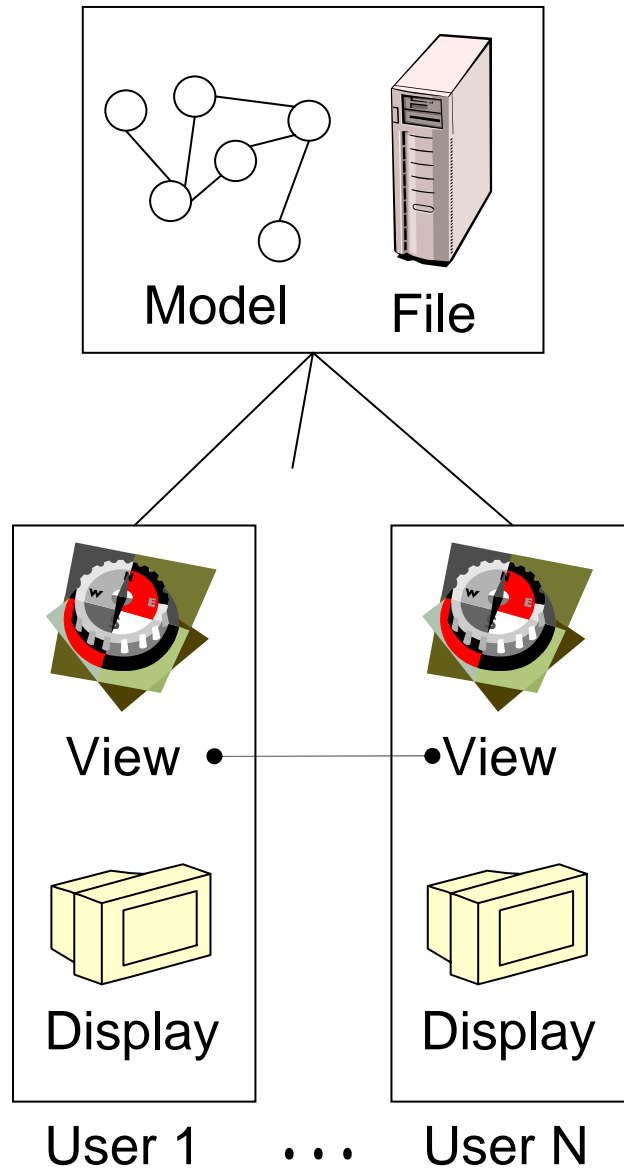
Shared Model



Shared View



Hybrid Model





Distributed Software Engineering



Applicability of CSCW

- Evolution
 - Changes must be tracked among many different versions of the software
- Scale
 - Increasingly large systems involve more interactions among methods and developers
- Distribution of knowledge
 - Many people all working on the system
- Clustering of edits (12.5% within 24h)
 - Perry et al. (2001)



Configuration Management

- Mutual Exclusion
 - Check in and check out
 - EREW, CREW, & CRCW
- Versioning
 - Rollbacks, branches, and joins
- Permission/access



Distributed CMS

- Configuration Management Systems
 - CVS, RCS, VSS, etc.
- Can distribute repository on multiple machines
 - Ubiquity of networked machines
 - Improve network access (distributed cache)
- Currently distribute at file-level only



Mappings to Known Problems

Cache Coherency

- Assume a CRCW policy
- Changes must be propagated
 - Various update protocols
- Approaches
 - Snoopy cache
 - Write invalidate
 - Directory-based protocol (shared, exclusive, unused)

Mutex

- Must ensure that only one user can edit any given area
- Helps mitigate merge problem
- Fine granularity
 - Improves parallelism, complicates modification effects
- Course granularity
 - Eases modification effects, reduces parallelism

Merging

- User 1 has artifact A
- User 2 has artifact A
- Both users edit (asynchronously)
 - User 1 creates A_1
 - User 2 creates A_2
- Check-in both modified versions
- Automate the merge process
 - Detect collisions in documents

Handling Synchronicity

- Conflict Resolution Matrix
 - $O(n^2)$ – can be parallelized
 - Shen & Sun (2002)
- Dependency Trees
 - Firewalls and cursors
 - Change propagation into subtrees
 - Kaiser & Kaplan (1993)



Recent Work



Other in the Field

- Grouplab
 - University of Calgary
- Cooperative Systems Engineering Group
 - Computing Department, Lancaster University
- Graphics, Visualization & Usability Center
 - Keith Edwards
 - Georgia Institute of Technology
- Xerox PARC
- SigGROUP of the ACM
- Many Others!

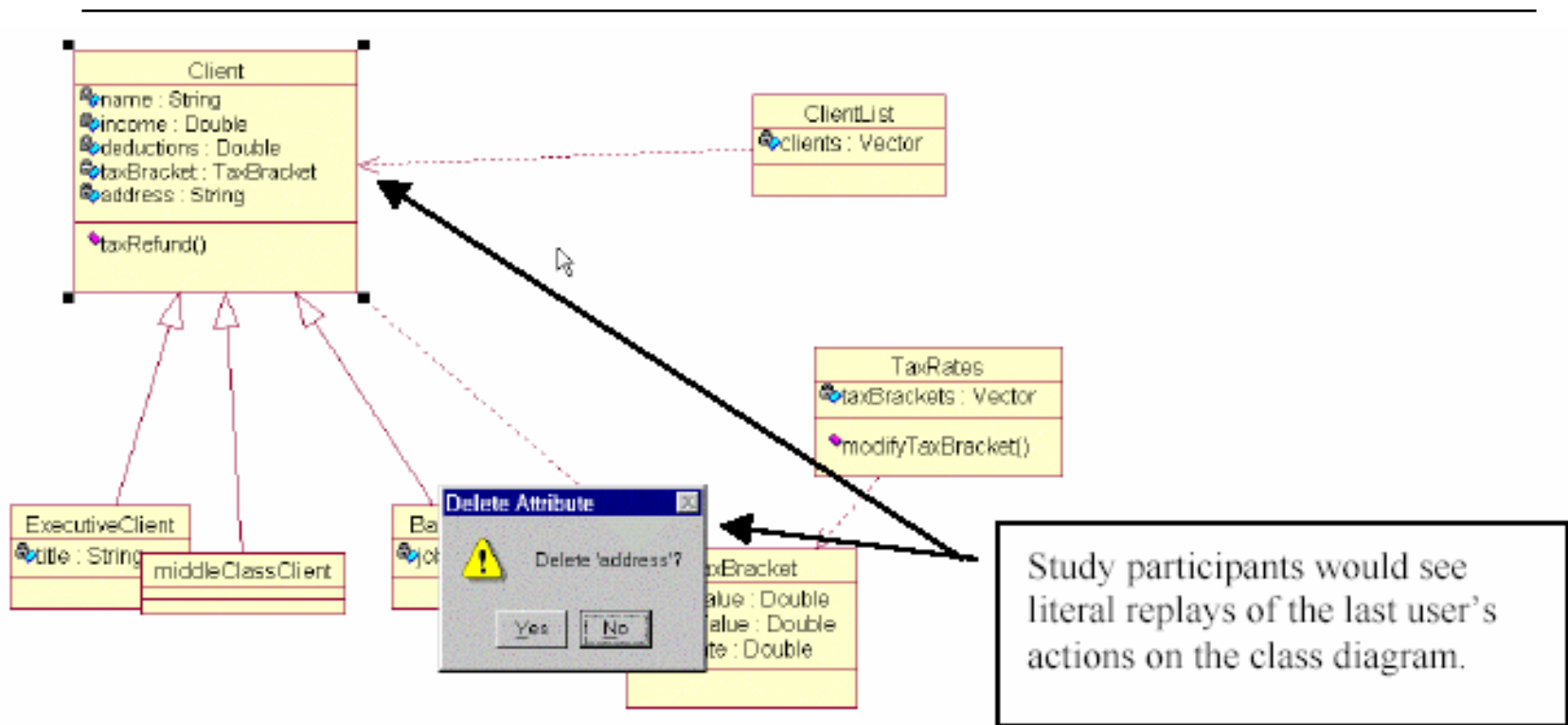


A Sample in CSCW Software Engineering Research

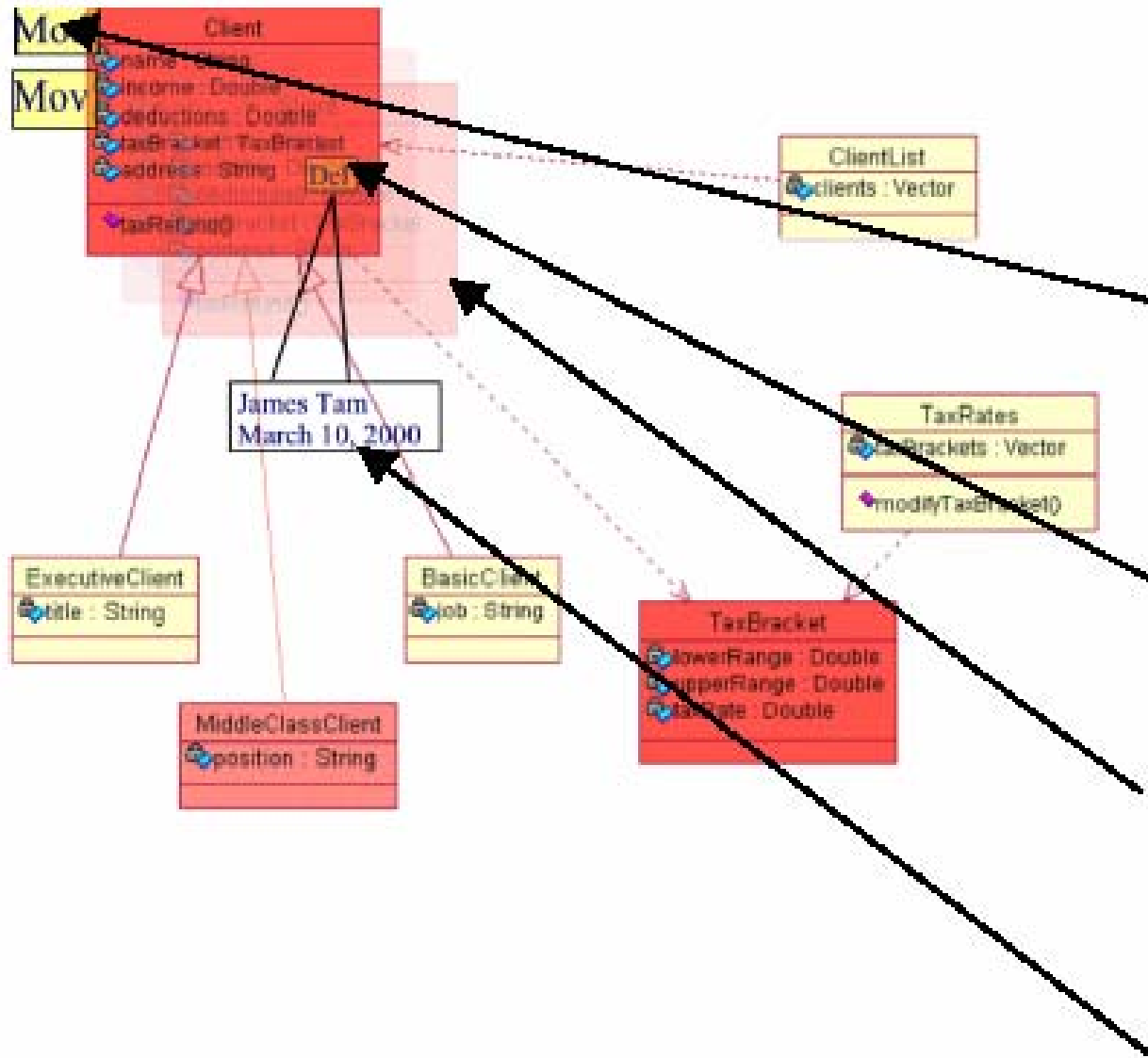
Tam, J., McCaffrey, L, Maurer, F. and Greenberg, S. (2000) **Change Awareness in Software Engineering Using Two Dimensional Graphical Design and Development Tools**. Report 2000-670-22, Department of Computer Science, University of Calgary, Alberta, Canada, October.

The Study

- Asynchronous
- Prototyped a software modeling tool using UML
- Placement: situated vs. separate
- Presentation: symbolic vs. literal
- Four change visualizations
 - Animated replays (situated & literal)
 - Storyboards (separate & literal)
 - Iconic representations (situated & symbolic)
 - Documentation (separate & symbolic)



Situated & Literal: Animated Replays



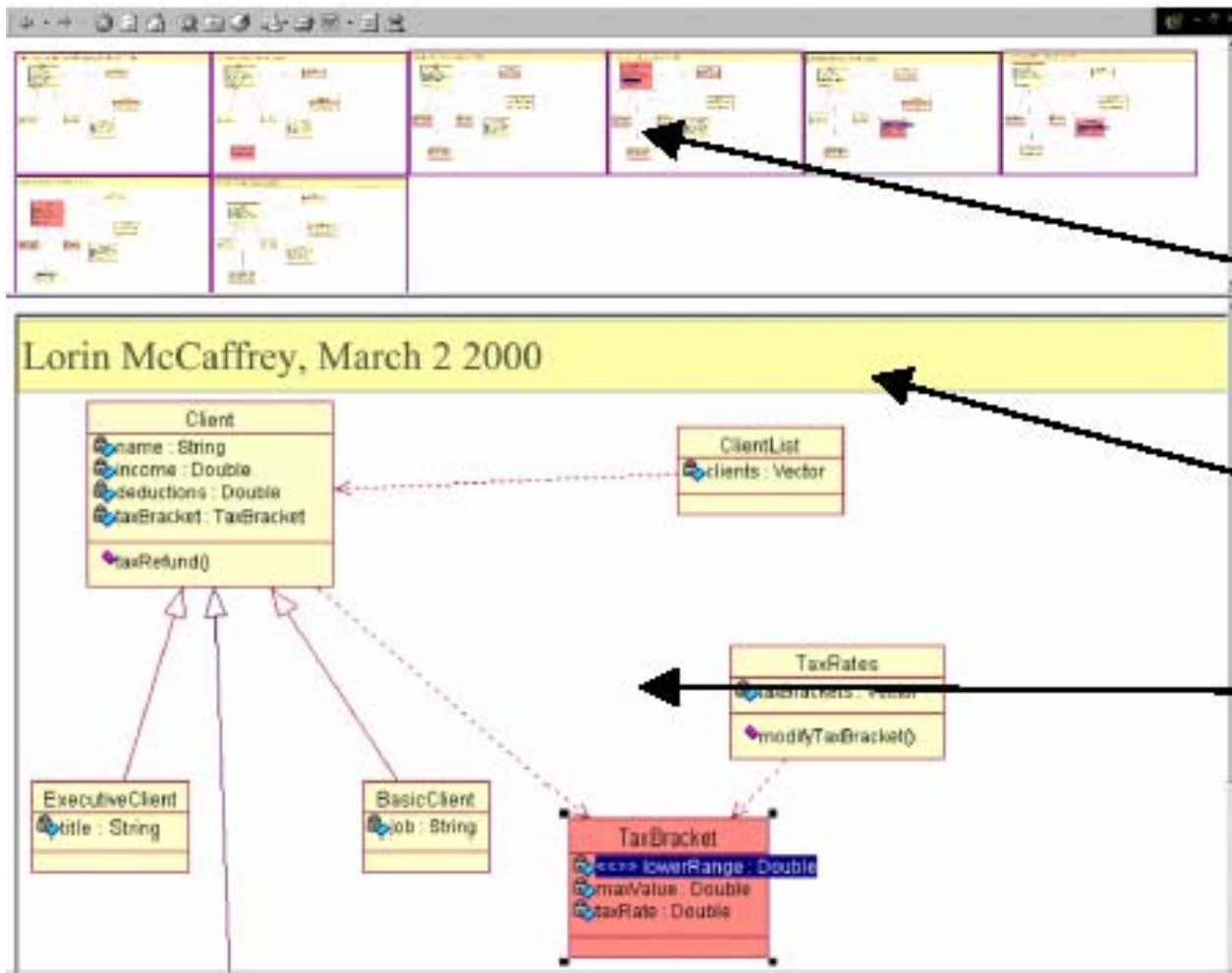
Class level change Indicator.

Field level change indicator.

Ghosted image indicating class movement.

Extra change information

Situated & Symbolic: Iconic

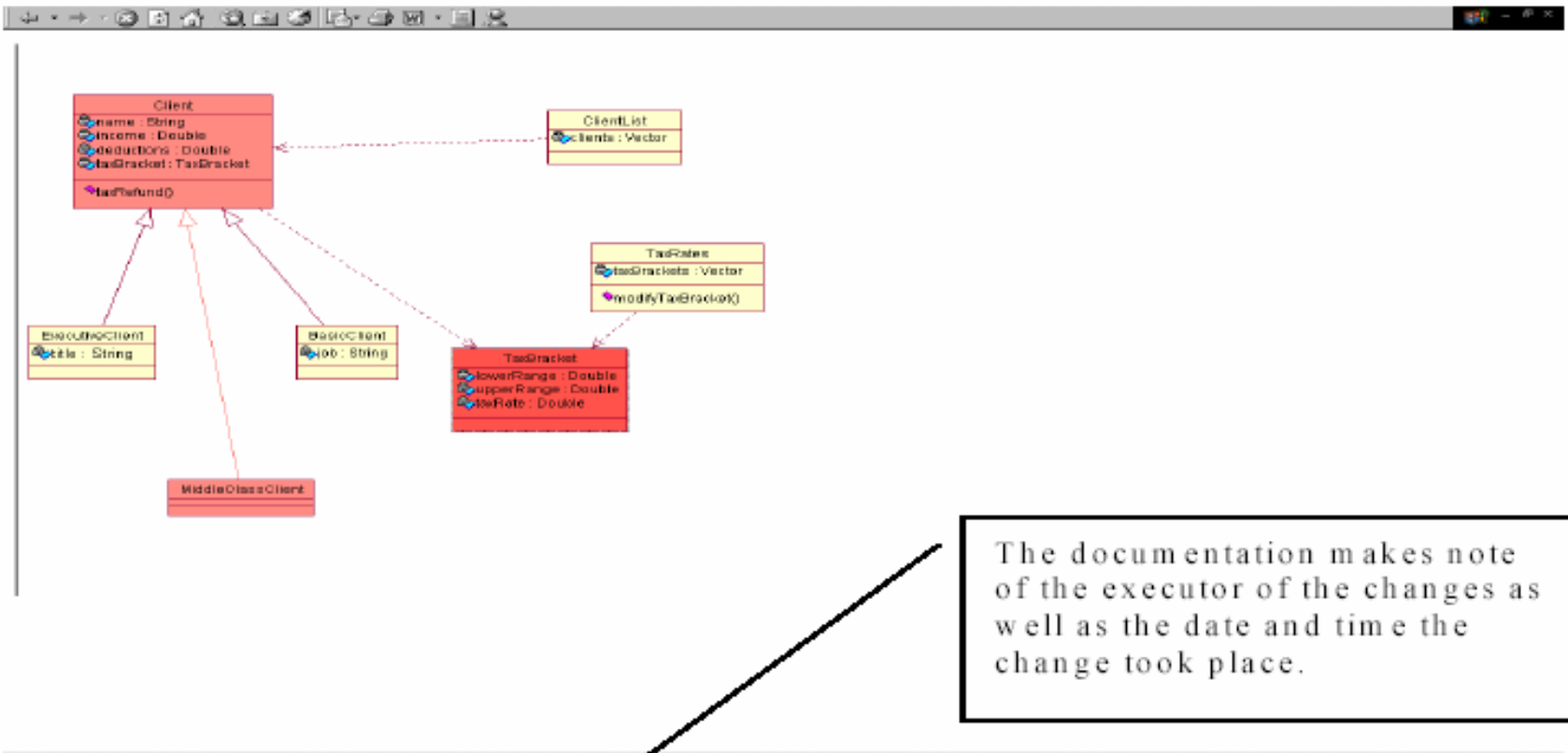


Storyboard pane

Extra change information

Detail pane

Separate & Literal: Storyboards



The documentation makes note of the executor of the changes as well as the date and time the change took place.

Changes for class TaxBracket:

- lowerRange renamed from minValue, modified by James Tam on March 13, 2000.
- UpperRange renamed from maxValue, modified by James Tam on March 15, 2000.

Separate & Symbolic: Documentation



Procedures

- Four changes
 - Addition of information
 - Deletion from diagram
 - Modification
 - Movement
- Pre test
 - Knowledge of UML and CS
- Post test
 - How the mechanism worked
 - Likert 1 (useless) - 5 (very effective) scale

Results

- Symbolic well accepted (abstraction)
- Gaze shift of separate a negative
- Before-after comparisons of storyboarding time consuming
- Difficulty in visualizing movement
- Users wanted to see current version of document
- “Who” and “why” of change critical



Potential Open Research



Granularity

- Examine fine-level of granularity in configuration management systems
- Lock at a class level
- Lock at a method level
- Lock at a block level



Implications of Fine-level Locking

- Modifications within a class (private)
 - Could impact other elements of a class
- Modifications to a class (public)
 - Could impact API (clients) of class
- Modifications at the block level
 - Could impact other block elements
- Track/notify user of implications



Models/Patterns of Software Editing/Development Behavior

- How do developers build/edit software?
- PSP/TSP and others address productivity
- Other models attempt to capture process at a higher-level
- Extreme Programming
- But are there patterns to how code is built and/or edited?



Non-Code-Based Synchronous Editing Systems

- Integrate into existing, widely-adopted document editing systems
- Peer-to-peer communication
- Improve Versioning (CM) into document editing

References

- KAISER G. E. and KAPLAN S. M. *Parallel and Distributed Incremental Attribute Evaluation Algorithms for Multiuser Software Development Environments*. ACM Transactions on Software Engineering and Methodology, vol. 2, no. 1, pp. 47-92, January 1993.
- PERRY D. E., SIY H. P., and VOTTA L. G. *Parallel Changes in Large Scale Software Development: An Observational Case Study*. ACM Transactions on Software Engineering and Methodology, vol. 10, no. 3, pp. 308-337, July 2001.
- ROTH, J. and UNGER C. *Developing synchronous collaborative applications with TeamComponents*. 4th International Conference on Cooperative Systems. 2000.
- SHEN H. and SUN C. *Flexible Notification for Collaborative Systems*. In Proceedings of CSCW'02, New Orleans Louisiana, pp. 77-86, November 2002.
- TAM, J., MCCAFFREY, L, MAURER, F. and GREENBERG, S. *Change Awareness in Software Engineering Using Two Dimensional Graphical Design and Development Tools*. Report 2000-670-22, Department of Computer Science, University of Calgary, Alberta, Canada, October 2000.



Questions
