Software Design within Comet: Manipulating Database Tables with JavaScript and PHP

Nathan Thomas
University of Notre Dame, Department of Computer Science and Engineering, Notre Dame, Indiana

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120
Intro: Comet

- Data Sending Toolkit
  - Creates Data Channels
- Pushes Data
  - No Server Request from Client

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120
**Intro: vEOC**

- Virtual Emergency Operations Center
- Web based Emergency Training Simulation
- Research Tool
- Controller Module
- vEOC Module that Manages the Simulation
  - Needs to Send Data to Users
  - Essentially a “Dispatcher”

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120.
Design Considerations

- Flexibility
  - Easy to Port to Multiple Browsers
- Server Based Processing
- Reduce CPU Load for Users
- Synchronization
  - Everyone on Same Page = More Effective Simulation

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120
Design Challenges

- Not Sending Data
  - Controller Pulled Data from MySQL Database
    - Pulled into PHP
  - Comet was Failed to Send PHP Data
  - Converted to JavaScript
    - Sent Properly
Design Challenges

- Jumbling of Data
  - Data was Sent as a Table
    - Sent Each Cell Individually
    - Caused Cells to Overwrite Each Other
  - Converted into a String Before Sending
    - Kept Data in Proper Order
    - User Client Merely had to Split String
Design Challenges

- Sending to Individual Users
  - Data Sent to All Connected Users
    - Caused by Only One Global Channel
  - Created Unique Individual Based Channels
    - Allowed for Specific Data Sending Targets
Design Challenges

- Overwriting of Injects
  - Data Sent Over Virtual Communication Mediums
    - i.e. Cellphone, Radio, Landline, etc
  - Data from Different Mediums was Overwriting
    - Was Being Sent Over Same Channel
  - Created Sub-Levels of the Individual Channels
    - A Sub-Level Channel for Each Medium
    - Prevented Data of Different Mediums from Overwriting
Discussion

- Building a Working Controller Module
  - Design Challenges
  - Design Considerations
  - Implementation
- Other Uses of Comet & MySQL
  - Autonomously Sending Large Amounts of Data
    - Data Must be Pre-Defined
Overview

Controller Tells Server to Send Data

Server Pulls MySQL into PHP

Data sent with Comet to Client

Server Converts Data to JavaScript and places it into a String

Client Processes Data and Converts to Array

Array is interpreted and then displayed as an alert

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120
Conclusion

- Comet Creates Data Channels
  - Data is Pushed Over the Channels
- Pushed Data is Useful for Simulations
  - Can Replicate Real-Time Alerts
    - No “Requesting” From the Users
- Database Data Must be Manipulated to Push
  - Convert to JavaScript String
- This Principle Allowed for a Working vEOC Controller Module
I would like to thank Cynthia Nikolai and Dr. Gregory Mady for their mentoring in this research. I also would like to thank anyone else who helped. I also would like to thank Troy Johnson of the Miami-Dade EOC for his assistance in conducting this research.

Questions?

This research was supported in part by the National Science Foundation under grants CNS-050348 and CNS-0751120.