



Towards Innovative and Interactive Multimedia: Models and Tools to Support Authoring

Joseph A. Konstan
Department of Computer Science and Engineering
University of Minnesota
konstan@cs.umn.edu
<http://www.cs.umn.edu/~konstan>



Characteristics of Flexible Presentations

Media Segments of Unknown, Variable, and Fixed Duration

User Interaction

- Within object (e.g., pause)
- Within presentation (e.g., skip to object)

Many Variations in Playback

- Constraints set to maintain the integrity of the message



Overview

Seven to Eight Years of Research

- Lessons learned along the way

Four Main Pieces

- FLIPS – graph-based presentation model
- Command media stream
- NSync – temporal constraint model
- DEMAIS – design tool



Variable Duration Media

Simulations

- Live animations, computer-based labs
- Computer sessions
- Streams of commands

User-Controlled Media

- Sequences of images
- Media with increasing detail or hyperlinks



FLIPS – Flexible Interactive Presentation Synchronization

with Jim Schnepf and David Du
(*IEEE JSAC 1/96; IEEE Multimedia '95*)



Coarse- vs. Fine-Grain Synchronization

Fine-Grain Synchronization

- time time tolerances; continuous enforcement
- exact mapping (even to frame level)

Coarse-Grain Synchronization

- ordering relationships
- defined synchronization points

FLIPS is concerned only with coarse-grain synchronization



Research Objective

To design and implement a model for specifying and enforcing coarse-grain synchronization for flexible presentations



Example, continued

Music plays continuously within architecture type
Slides loosely coupled with narrative
Many options when synchronization point reached

- first-to-finish, last-to-finish
- narrative-control, slide-control



Synchronization Requirements

Objects not tied to specific duration
Variety of temporal relationships

- first, last, master-slave, no sooner, no later, ...

Specify consistent and coherent presentation
Maintain globally consistent state

- with variable duration objects
- with user interaction



The FLIPS model

Objects can be synchronized at start and end
Start (end) of an object must be

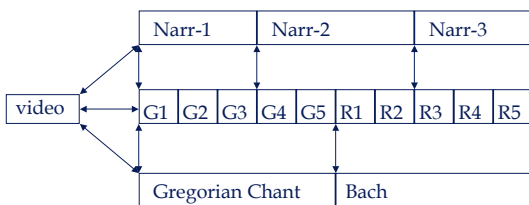
- enabled -- (and enable is satisfied)
- barrier-free -- all barriers are removed

Objects that finished but cannot end perform and alternative action
State changes propagated to achieve consistency

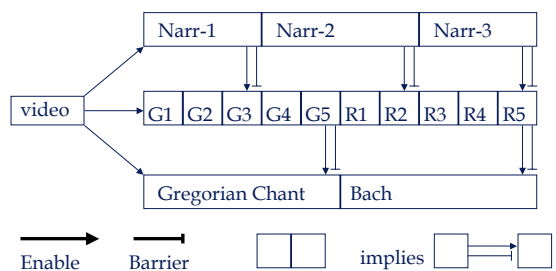


FLIPS by Example

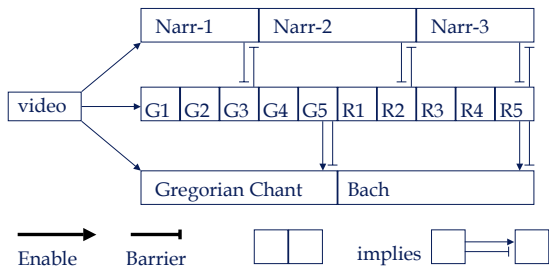
Gothic to Renaissance Architecture



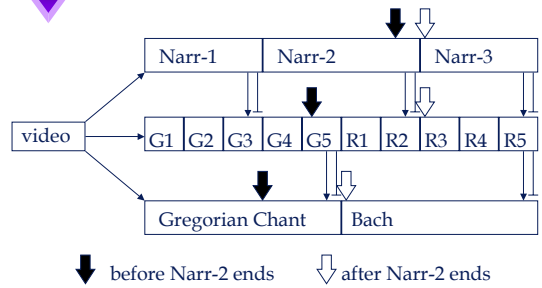
Narrative Controls Slides, Slides Control Audio



Narrative and Slides Parallel Last, Slides Control Audio



Moving Forward



Rules for Normal Play-Out

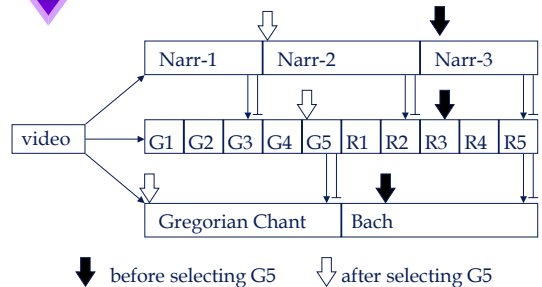
Object begins when its START is enabled and barrier-free

Object ends when its END is enabled and barrier-free

- most objects have an implicit END enable
- alternative action when END blocked by barrier

Mutual barriers handled by "sub-states"

Jumping Backward



Global Consistency: Propagating State Changes

When an object changes state

- all conditions for that state must be satisfied
- changes must be propagated across all relations

Examples

- moving forward (naturally or user jump)
- jumping backward

Implementation

Two implementations: Unix and Windows
Central control program

Implementation model

- wrappers for existing players
- native viewers when performance dictates



Related Work

Model classifications (Blakowski)

- hierarchical, timeline, *reference points*

Most require known and fixed object durations

- some allow determination before presentation
- event-based models lack barriers

Interaction

- most presume fine-grain timeline
- coarse-grain models do not define coherence



What is the Command Medium?

Arbitrary executable code

- implemented as Tcl code

Synchronized with other media

- timeline-based synchronization in CMT

Presentation support

- different speeds, directions, jumps

Coherent command streams

- not just a set of individual commands



FLIPS Conclusions

Flexible presentations

FLIPS model

- barriers and enablers
- user interaction
- not a user-level language

Goals

- integrated synchronization



Examples of Command Media

Animation

- music video, parser animation, ...

External control

- presentation support: lights, volume, ...
- other programs and/or devices

Presentation control

- select among media streams
- on-screen playout control



TclStream

with Jon Herlocker

(*Proc. ACM Multimedia '95*)



Structure of a Command (Chunk)

Time

- absolute or relative to last chunk

Primary command

Inverse command

Rush-ahead command

Rush-behind command



Examples

```

{0.1 {move_arm left_arm 100 100}, # Primary
{} # Rush-ahead
{move_arm left_arm 100 300}, # Inverse
{} # Rush-behind

{100.0a {button .b -text "Skip to Polka!"
-command {set jumped 1 ; self .lts 300;
destroy .b}}
{set jumped 0}
{catch {destroy .b} ; catch {unset jumped}}
{catch {destroy .b} ; catch {unset jumped}}}

```



Focus: Interactive Media

Assume media content is available

Express temporal behavior (layout)

- Coordinates the content through time

Express interactive behavior

- Navigation, participation, and control

Express combinations of temporal and interactive behavior

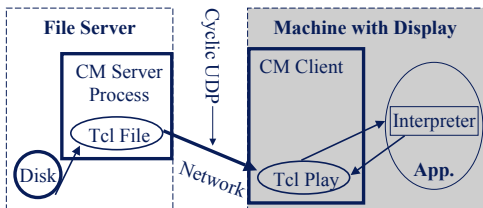
- e.g., time-dependent interaction



Implementation

Implemented as a Medium in Berkeley CMT

Commands can be any Tcl/Tk code



Authoring Language Limitations

Lack *expressive* temporal constructs

- Timers poor for expressing temporal behavior
- Polling required to enforce temporal behavior

Expressing combinations of interactive and temporal behavior not possible

- Distinct event source for each
- Burden is on author to manage the conditions



Nsync – A Language and Evaluation Technique for Synchronization and Interaction

with Brian Bailey

(*Proc. ACM Multimedia '98*)



Background / Model

Each media stream attached to a clock

Clock value is linear function of system time

$$Value = Rate * System_Time + Offset$$

Express temporal behavior as relationships among clocks

Interactive events tied to variables



Nsync Language

Associate *actions* with *expressions*

Expressions may contain multiple clocks, variables, relational operators, boolean connectives, and scalars

When the expression becomes TRUE, invoke associated action

```
When "Time > Q.end + 5 &&
    !Response" Answer=WRONG
```



Delayed Transition Example

```
When "Narration >= Overview &&
    !MoreInfo" NextSlide
```

```
When "Narration >= Overview &&
    MoreInfo" PlayDetails
```

```
When "Narration >= Overview + Details"
    NextSlide
```

- Narration: narration's logical timeline
- Overview: normal transition point
- Details: additional narrative details
- MoreInfo: records kitchen info status



Delayed Transition

Narrated slide show of home

- Split narration into overview and detail sections
- Transitions occur after overview completes

"More Info" toggle

- Pressing schedules details to be heard
- Releasing stops details and causes transition



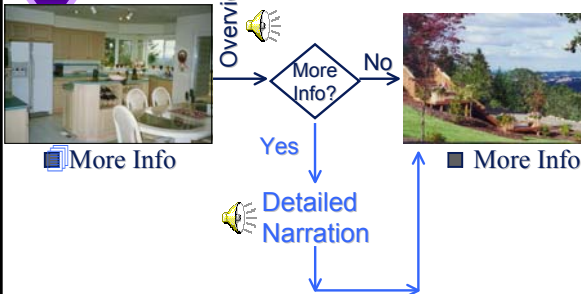
Expression Evaluation

Propositional logic breaks down

- Returns logic value only at *present* time
- Requires polling or timers to catch future logic transitions

Predictive logic

- Returns logic value at present time along with a prediction of any future transition
- Eliminates need for intermittent polling/timers
- Applied according to a few simple rules



Predictive Logic States

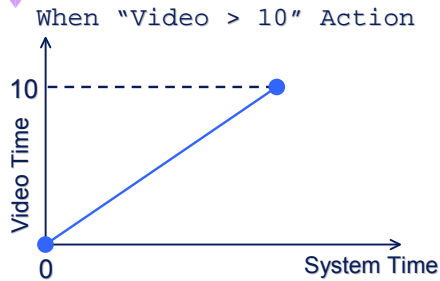
WBT(t) False now, but
Will Become True at future time t

WBF(t) True now, but
Will Become False at future time t

- FALSE = WBT(infinity)
- TRUE = WBF(infinity)



Prediction Example



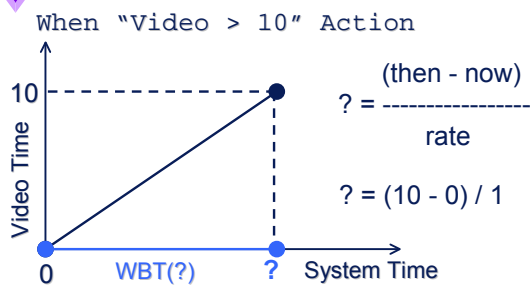
AND truth table

AND	TRUE	FALSE	WBT _{x2}	WBF _{y2}
TRUE	TRUE	FALSE	WBT _{x2}	WBF _{y2}
FALSE	FALSE	FALSE	FALSE	FALSE
WBT _{x1}	WBT _{x1}	FALSE	WBT _{max(x1,x2)}	*
WBF _{y1}	WBF _{y1}	FALSE	†	WBF _{min(y1,y2)}

* if (x1 < y2) then {WBT_{x1}, WBF_{y2}} else FALSE
 † if (x2 < y1) then {WBT_{x2}, WBF_{y1}} else FALSE



Prediction Example



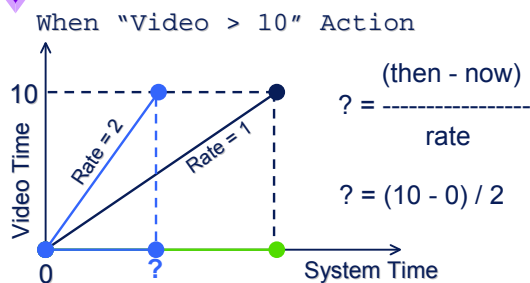
OR truth table

OR	TRUE	FALSE	WBT _{x2}	WBF _{y2}
TRUE	TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	WBT _{x2}	WBF _{y2}
WBT _{x1}	TRUE	WBT _{x1}	WBT _{min(x1,x2)}	*
WBF _{y1}	TRUE	WBF _{y1}	†	WBF _{max(y1,y2)}

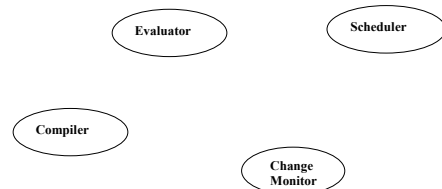
* if (x1 > y2) then {WBF_{y2}, WBT_{x1}} else TRUE
 † if (x2 > y1) then {WBF_{y1}, WBT_{x2}} else FALSE

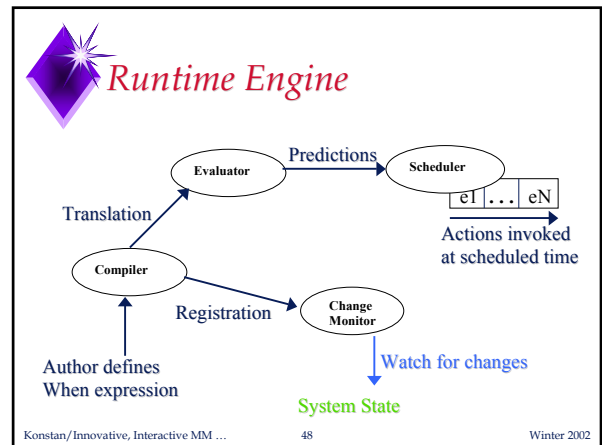
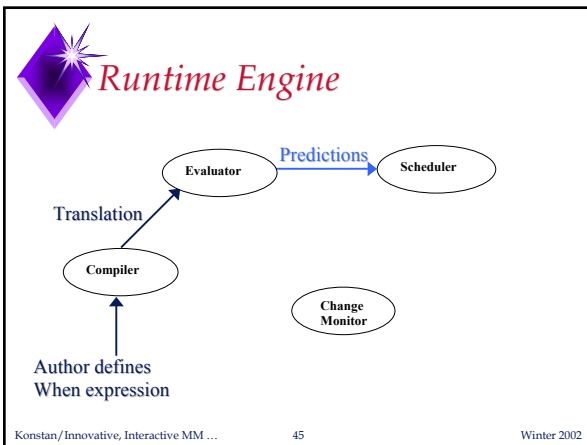
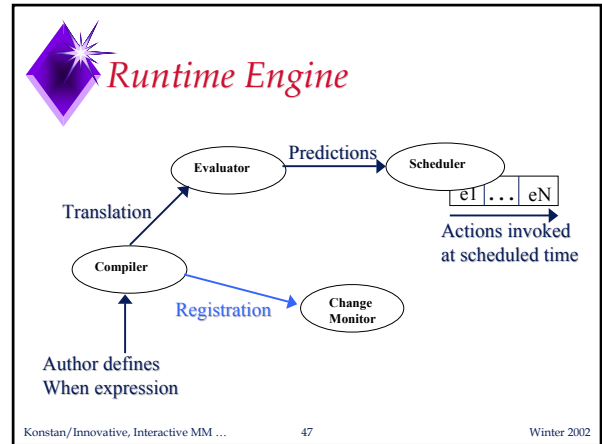
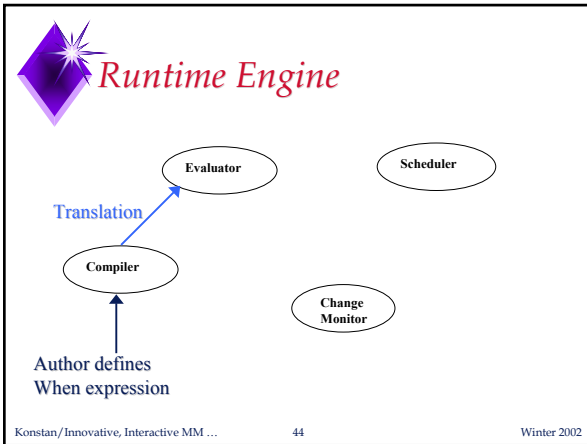
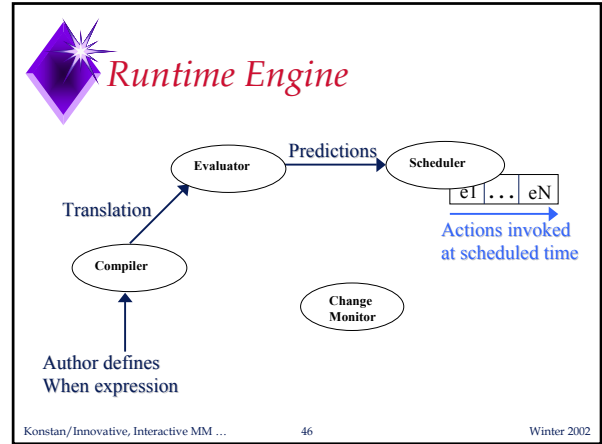
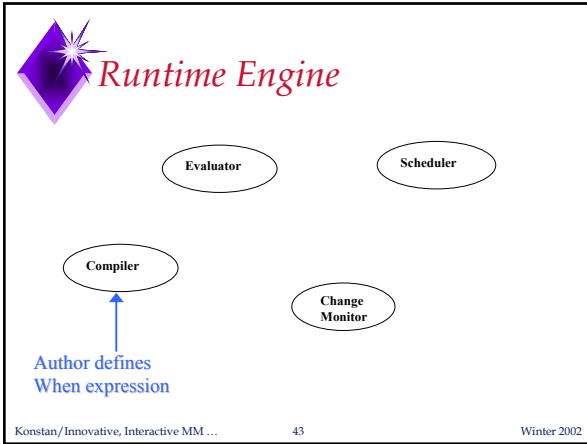


Prediction Example



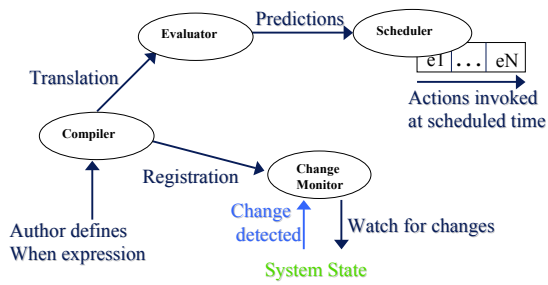
Runtime Engine







Runtime Engine



Nsync Conclusion

Complements current languages

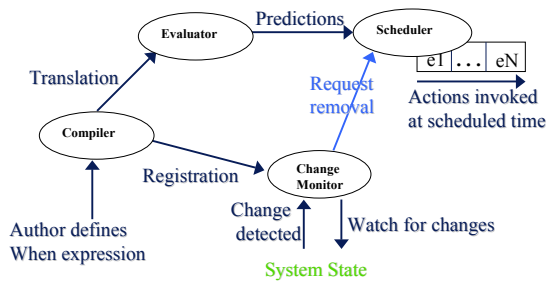
- Adds ability to express temporal behavior
- Adds ability to express combinations of interactive and temporal behavior
- Syntax can easily be translated into mark up

Predictive logic useful in run-time engines

- Eliminates need for polling/timers
- Prefetch data needed in the future



Runtime Engine

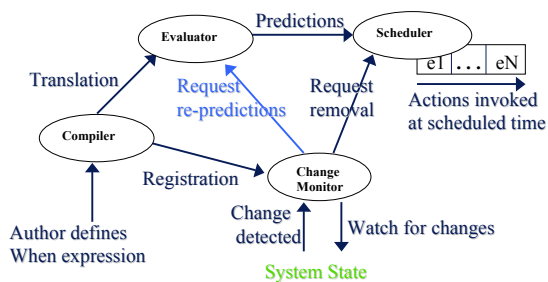


DEMAIS – A Design Tool for Interactive Multimedia

with Brian Bailey
(Proc. ACM Multimedia '01)



Runtime Engine



Design vs. Development

Designers tend not to be implementers

Need to explore and communicate ideas quickly

Need to explore interaction!

Key Ideas

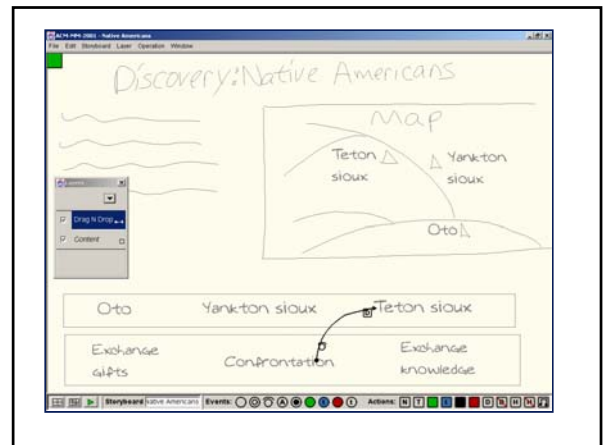
Sketching

Stroke language to add event/temporal meaning

Storyboard and multi-board views

Incorporate or record media where needed

Layers to disambiguate when needed



DEMAIS - Tool Components

Storyboard editor

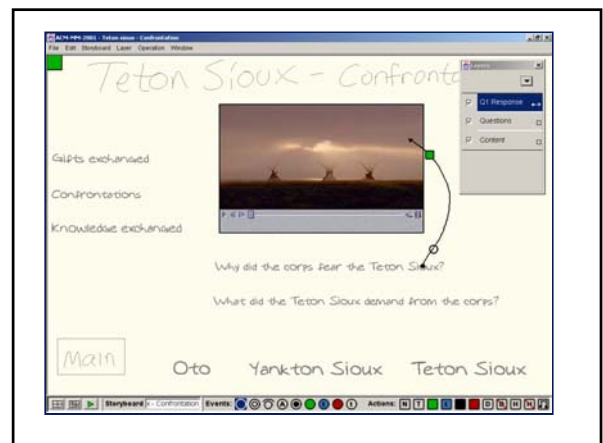
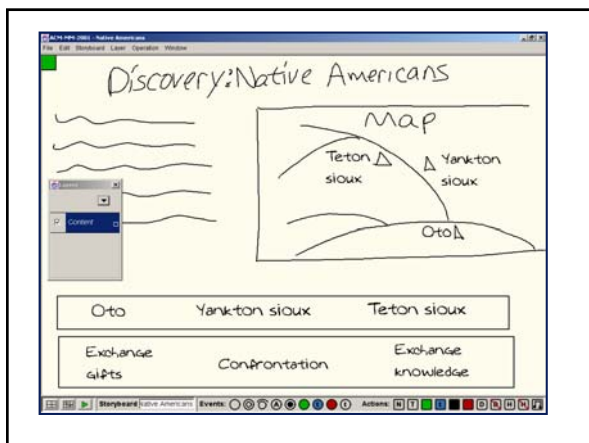
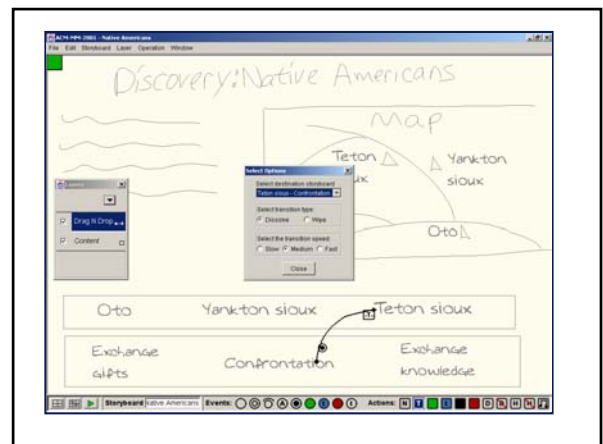
Narration editor

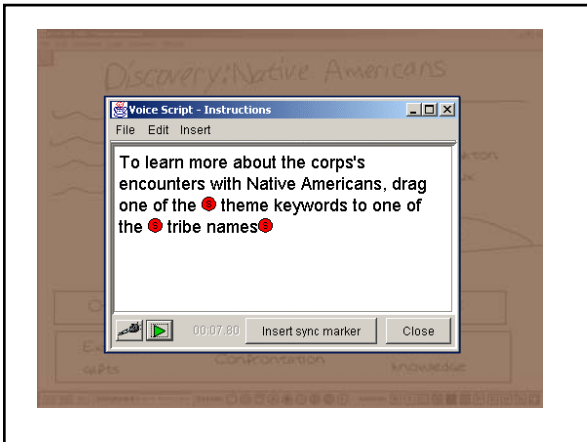
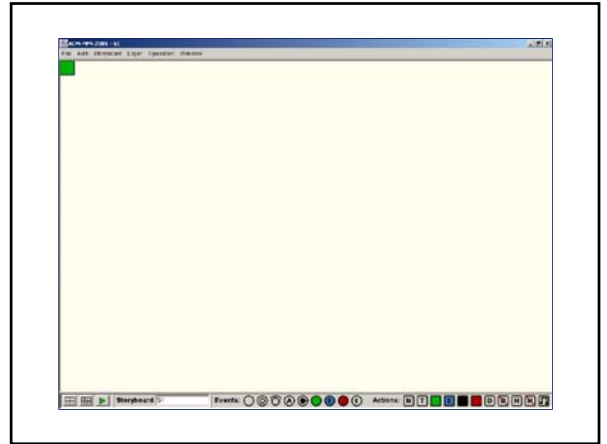
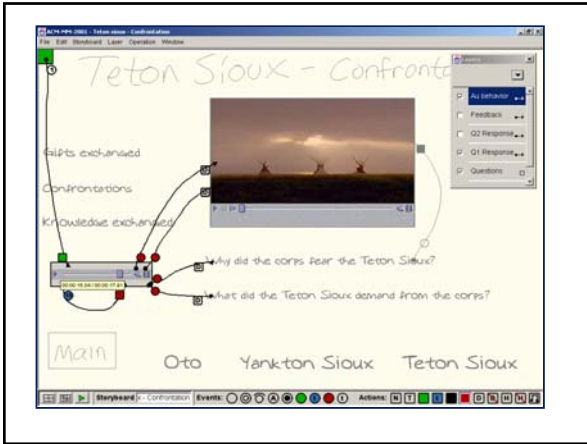
Multi-view editor

Sketch temporal and interactive behavior

Explain in context of design example

- Lewis and Clark expedition of the early 1800s





Implementation

Hardware

- Wacom display tablet with stylus

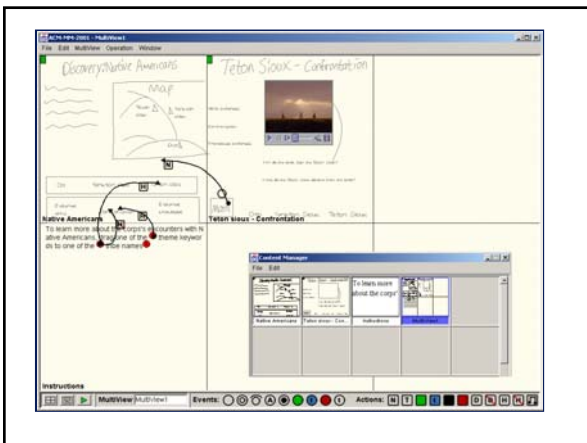
Software

Recognizing rectangle and gesture strokes

- Mahalanobis distance classifier

Runtime engine

- Adapted NSync toolkit



Status

Version available for experimentation at:
<http://www.cs.umn.edu/~bailey>

First round (formative) user testing
 completed

Second round (comparative) completed
 and being analyzed



Acknowledgements

The work reported on here has been supported by several different grants from the National Science Foundation and by grants from private companies. Many colleagues have contributed to the success of this work.



Towards Innovative and Interactive Multimedia: Models and Tools to Support Authoring

Joseph A. Konstan
Department of Computer Science and Engineering
University of Minnesota
konstan@cs.umn.edu
<http://www.cs.umn.edu/~konstan>