Active Server Roles for Extended Distributed Performance Complexity in Diamonds in Dystopia

Jesse Allison
School of Music and CCT
Louisiana State University
Baton Rouge, LA, USA
jtallison@lsu.edu

Frederick Ostrenko
School of Art and CCT
Louisiana State University
Baton Rouge, LA, USA
dostrenko@lsu.edu

Vincent Cellucci
College of Art & Design
Louisiana State University
Baton Rouge, LA, USA
vcellu1@lsu.edu

ABSTRACT

Distributed performance systems that utilize a centralized server for connectivity have the potential to also provide extended computational and storage resources that would not be beneficial or even possible if distributed onto mobile clients. The usage of large datasets, shared or collaborative resources, and processor intensive techniques can be performed on the server while allowing time sensitive and less complex user specific computation to occur on client devices. Many of these types of problems can be solved using tools developed for cloud computing. This approach is demonstrated in the work *Diamonds in Dystopia*, a collaborative poetry performance that incorporates audience interaction on mobile devices, generation of poetic material on server side resources, real-time synthesis distributed through mobile and venue speakers, a live poetry reading performance and the live synthesis of poetry from the collective ensemble.

1. INTRODUCTION

Current distributed performance systems tend to focus on facilitating or moderating the communication of information between mobile clients. This type of passive server role has been formed around a publication/subscription (pub/sub) model as demonstrated by the Rhizome project[7] by Sebastien Piquemal and Soundworks[6] developed at IRCAM, among others. It is an effective model for pieces where the mobile devices can be relied upon to handle the processing required and where there is not an excessive amount of network traffic. However, this does not tend to perform well if large files need to be moved between devices, if even moderately processor intensive computation is required, or if resources need to be generated collaboratively. (Figure 1)

An alternative approach is to create a more active role for the server, providing larger memory, storage, and computational resources for the parts of the work that require them. This role can offload heavy computational activities such as machine learning tasks, music information retrieval algorithms, or in our use case, natural language processing. Large data files can also be used on the server end, only sending smaller amounts of data to the client on an as needed basis. The server can also be designed to spawn child processes as needed to deal with computationally difficult tasks allowing communication to continue unimpeded while dealing with tasks.

Although this approach does not solve all distributed performance issues, it does enable some alternative types of interactive performance.

Figure 1: Participant in front of shared projection.

2. DIAMONDS IN DYSTOPIA

Diamonds in Dystopia[4] is an interactive, live-streaming poetry web app that takes the audience through the sensory decisions and experience of creating a poem collectively. It utilizes creative data mining of the transcripts of 2050 TED talks for phrases that can be organized into a poem.

A *Diamonds in Dystopia* performance has many moving parts and its design encompasses the following tasks:

- Provide a mobile friendly web portal into the current collaborative poem activity.
- Present a live poetry reading of a prepared poem - in 2016, this was *Bound by Digital Countries* created in the same TED Talk transcript data mining manner as the live performance.
- Generate music influenced by the activity of the crowd and keyed to events in the poetry reading.

Web Audio Conference WAC-2017, August 21–23, 2017, London, UK.

- Use audience mobile interactions to trigger natural language processing (markov chain permutations) of collected TED talk transcripts, creating synthesized spoken readings of these words and other sound playback on individual devices.

- Present an interface and interpret interactions by the Poet to initiate collaborative poem projection and moderately synchronized sound events across all of the audience devices and main speakers.

- Present the Poet with audience generated stanzas which can then be selected to incorporate them into the live reading at any time.

- Visualize the generated poem for the collective audience.

The computational resources, data management, and communications for this performance have stringent requirements to ensure responsive interaction, reliable connectivity, and maintain the sense of influence and interaction across all entities.

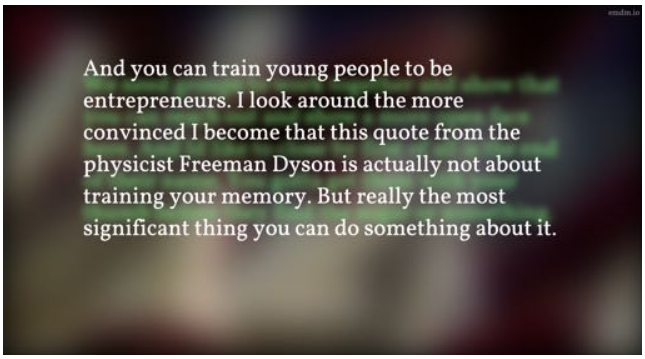

Figure 2: *Diamonds in Dystopia* Shared projection.

2.1 Artistic & Interaction Design

The audience acts as a collaborator by tapping word selections that resonate with individual users on their mobiles. These words are sent to the server to trigger Markov chain reactions, creatively data mining 2050 TED talks to send an improvisational stanza to the poet on stage. The individual taps coming from the audience also trigger synthesized speech and music at varying pitches to create an aggregate musical experience as well as contributing to the shared visual projection of the poem. Figure 2.

Instead of a cacophony of sound, we introduce a cacophony of collaboration in the form of a live-stream of generated poetry, which a performer can moderate by scrolling and selecting text generated from the audience as authors. The system generates an aggregate TED Talk inspired poem as found text to generate a collective conscious art work.

3. PERFORMANCE SYSTEM DESIGN

Diamonds is run as a clustered Node.js application that can be served from Google Cloud Compute, the HIVE HPC at Louisiana State University, or any server with sufficient resources for the number of expected participants. (Figure 3) It was built as an extension of the Nexus [1] framework for distributed performance.

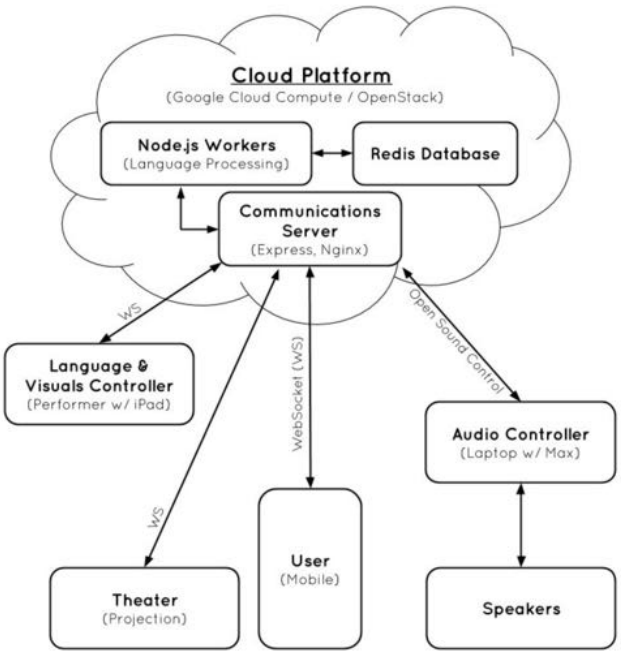

Figure 3: *Diamonds in Dystopia* Technical Diagram.

With a large collection of text and a seed poem, the system is prepped for an interactive poetry performance. The entire corpus of TED talks was loaded into a Redis server keeping the data in RAM, but allowing it to be quickly accessible to all instances of node as required.

In the premier performance, a node cluster was used to create a master node and 15 workers, each of which can handle creation of markov chains using the RiTa library for natural language processing[3]. The Redis server allowed for shared storage of the seed texts, storage of the socket.io ids, and generated poetic phrases. As the performance wore on, each node instance performed the natural language processing and responded to individual requests from the audience. During the course of the performance, the RAM requirements for these instances could grow to above 1Gb, validating the need for a centralized high performance computer with dedicated resources.

The virtualized infrastructure we used enabled us to allot high memory instances (52GB) for the performance. Nearly all of our database and processing resides in RAM via Redis to reduce latency.

Users connect to a Node based Express.js server which handles WebSocket and Open Sound Control communication in addition to serving the HTML, CSS, media files, and JavaScript for displaying the initial seed poem to users. They click on individual words that resonate with them which get passed along to a server via WebSockets. Each time a user taps a word the collection of texts, hosted on a Redis server, are searched for the top results that contain that word. Those texts are used as a source material for a markov chain algorithm to generate a new stanza of found text. These new stanzas are fed to the poet-performer who can choose, via a "controller" interface on their own mobile

device, which lines to read and display on the collective theater view. The poet's interface is shown in figure 4.

Figure 4: Performance interface for Poet.

3.1 Collective Display

This collective display within the theater comes in both sonic and visual forms: the projection of the collective poem as it is being generated, and the generation of background and event sounds piped into the venue.

The visualization is generated within a web page and simply loaded into a browser and shown full-screen. Communication with the rest of the system is through websockets. The theater view and user interface heavily rely on the CSS Animation specification for text effects. New content is displayed in front of a blurred version of the seed stanza, with its color associated with the last user to select it.

The shared music is created live within Max utilizing custom processing and Jamoma[8] modules. Communication with the server is through a specialized webpage that siphons web socket traffic through the jweb object in Max. This setup allows for the development of complex audio processes that would have been difficult if not impossible to do completely within the current implementation of web-audio. Web audio is used on mobile devices to complement user interaction and create a sea of interaction effects.

3.2 Audience

Diamonds audiences use their mobile devices connected to the internet via WiFi or a cellular network to participate interactively; user interactions contribute to the aural or musical sound installation by producing sound effects in the form of varied pitches for each user, computer generated voices of the written words that the audience chooses to tap, and audio push effects sent from controller. Projectors and speakers from the venue further enhance the audience's visual and sonic experience of the media for the theater performance.

Figure 5: Live Performance at 2016 TEDxLSU Event.

4. BOUND BY DIGITAL COUNTRIES

The premiere performance of this piece arose at a 2016 TEDxLSU event shown in Figure 5. It engaged 600 audience members of which more than 100 audience members engaged in the live interaction generating 2581 poetic phrases. A second performance occurred at the South By Southwest Interactive Conference (SXSW) in March 2017.

This event showed the drastic nature of live performance on the system, within seconds of the performance, the server is struck with massive amounts of usage from many devices making it incredibly difficult to stress test the framework adequately beforehand. In this performance, we used 8 cores on a Google Cloud Compute node with 52 Gb of RAM. Although the system performed well, the server analytics confirmed the immediate stress on the system as the pages first loaded jumping to 3.3Mb/second while downloading files and ramping to 45% CPU usage (across 8 cores) immediately as the performance began6. It is obvious that possible bottlenecks must be thoroughly tested for adequacy and possible redundancy, and a successful user experience prepared even in the case of catastrophic network failure.

Figure 6: Live Performance.

5. FUTURE DIRECTIONS

The system held up admirably under stringent performance requirements, however this was not without concern about how it would perform live with the potential of a few hundred participants. Developing automated testing tools specific to this type of performance is necessary. Not knowing exactly how a system this complex will perform when engaging a large audience during a brief, specifically allotted portion of time is not acceptable as performances become less experimental and more failure sensitive. Specific moments ripe for failure are the loading of the resources at the beginning of the work, maximum load on the active portions of the server (such as natural language processing and accessing the Redis server), and network traffic bottlenecks that may affect coordinated events across all users.

Simplifying the integration between Redis server, web socket communication with participants and performance entities, and worker tasks spawned as needed is also an important step to making this technique transferable between pieces.

Deployment is relatively simple, but exploring the usage of Dockers may make deployment to various server arrangements even more simple.

Artistically, future iteration could continue the creative data mining abilities built into Diamonds but open up the system (and subsequently the context) allowing performers to upload large collections of manuscripts and their own seed text enabling future poetry and new media art performances. This way one need not be a developer to perform heightened text collaboration–with the opportunity for generative input from an audience and a dynamic set of texts–or create unique, new media collaborative poetry from mobile interactivity. This new system would be attractive to contemporary poets, performing artists, and audiences seeking more engagement from readings as well as scholars curious about language processing to creatively and poetically datamine subtexts.

6. CONCLUSIONS

The ability to balance large amounts of data like the TED talk corpus, intensive computational processes like natural language processing, distributed communications bandwidth, and the limited resources of mobile devices is a difficult task, but can be effectively traversed by activating server resources in support of distributed computation.

7. REFERENCES

[1] J. Allison, Y. Oh, and B. Taylor. Nexus: Collaborative performance for the masses, handling instrument interface distribution through the web. In *Proceedings of the New Interfaces for Musical Expression conference*, 2013.

[2] P. L. Burk. Jammin' on the web - a new client/server architecture for multi-user musical performance. In *ICMC 2000 Conference Proceedings*. International Computer Music Conference, 2000.

[3] D. C. Howe. Rita: Creativity support for computational literature. In *Proceedings of the Seventh ACM Conference on Creativity and Cognition*, C&C '09, pages 205–210, New York, NY, USA, 2009. ACM.

[4] V. C. Jesse Allison, Derick Ostrenko. Diamonds in dystopia, 2016.

[5] R. Mills. Dislocated sound : A survey of improvisation in networked audio platforms. In *Conference on New Interfaces for Musical Expression NIME 2010*, pages 186–191. Nime, 2010.

[6] S. G. B. M. Norbert Schnell, Jean-Philippe Lambert. Soundworks, 2015.

[7] S. Piquemal. Rhizome, 2014.

[8] T. Place, T. Lossius, and N. Peters. The jamoma audio graph layer. In *Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx10)*, pages 78–86, September 2010.

[9] B. Swift, H. Gardner, and A. Riddell. Engagement networks in social music-making. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*, OZCHI '10, pages 104–111, New York, NY, USA, 2010. ACM.

[10] A. Tanaka. Mobile music making. In *Proceedings of the 2004 conference on New interfaces for musical expression*, NIME '04, pages 154–156, Singapore, Singapore, 2004. National University of Singapore.

[11] A. Tanaka. Mapping out instruments, affordances, and mobiles. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 15–18. New Interfaces for Musical Expression, 2010.

[12] A. Tanaka. Music one participates in. In *Proceedings of the 8th ACM conference on Creativity and cognition*, C&C '11, pages 105–106, New York, NY, USA, 2011. ACM.

[13] A. Tanaka, N. Tokui, and A. Momeni. Facilitating collective musical creativity. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 191–198, New York, NY, USA, 2005. ACM.

[14] F.-J. G. S. Weitzner, N. and Y. Chen. massmobile – an audience participation framework. In *Proceedings of the New Interfaces for Musical Expression Conference*, 2012.