Principles of Instrument and System Design for LaptOperas

Anne K. Hege
Mills College at Northeastern University
5000 MacArthur Blvd.
Oakland, CA, USA
a.hege@northeastern.edu

Curtis Ullerich
Iowa State University
2427 Union Dr.
Ames, IA
curtis@ulleri.ch

ABSTRACT
In this article, we describe the creation of an instrument ensemble control system for *The Furies: A LaptOpera*, an opera for laptop orchestra and live vocalists based on the Greek tragedy Electra.[1] We outline the practical considerations that inspired the creation of this system for the opera and the principles that guided this system's design. Through a detailed description of the development of the rope instrument and the design of the instrument ensemble control system, this paper offers tools and reflections on useful design strategies to support the rehearsal and performance of evening-length multimedia works.[2]

Author Keywords
Instrument design, Wekinator application, machine learning, instrument ensemble control system, opera, musical multimedia, performance, rehearsal, and system design.

CCS Concepts
• **Applied computing** → **Sound and music computing**; **Performing arts**; • **Computer systems organization** → **Dependable and fault-tolerant systems and networks**; **Redundancy**; **Reliability;** Maintainability and Maintenance; • **Human-centered Computing** → Human Computer Interaction (HCI); Interactive Systems and Tools; • **Computing Methodologies** → *Machine Learning;*

1. INTRODUCTION
The Furies: A LaptOpera began in the spring of 2019. Commissioned by the Stanford Laptop Orchestra, the work explored instrument building to unify the expressive and performative artistic vision of an opera for laptop orchestra and live voices (a laptOpera). Incorporating live vocals, staging, and setting into the instrument design process, composer, librettist, and creator Anne Hege, along with her collaborative team,[3] attempted to create a work that could embody the societal questions around cycles of violence posed within the Electra myth, the inspiration for the opera's narrative.

Between this project's beginning in 2019 and the full opera premiere in 2022, the instrument design team[4] faced many challenges in satisfying both the artistic vision and the practical necessities of live opera. This paper offers tools and reflections on design strategies to support the rehearsal and performance of an evening-length work. Through an in-depth discussion of one instrument within the opera, its artistic design principles, and the practical design elements, including the instrument ensemble control system, developed over time to support necessary rehearsal and performance demands, we offer basic principles for instrument construction for opera and long format multimedia works.

Figure 1: the rope instrument in "Loving It All" from *The Furies: A LaptOpera.* All photos in this article were taken by Ge Wang.

2. HISTORY OF THE FURIES: A LAPTOPERA
The Furies: A LaptOpera[5] was created in stages, beginning with Act III in the spring of 2019,[6] Act I in the fall of 2019, and from January through early March 2020, the ensemble prepared for the full premiere postponed due to COVID-19. In the summer of 2022, the project was remounted and received a full premiere on Stanford's CCRMA Stage on Nov. 11-13th, with a fourth performance at Mills College at Northeastern University Chapel on Nov. 17th, 2022. The final version consists of five laptop stations with six laptop players/chorus members and four

[1] Video documentation of the premiere of *The Furies: A LaptOpera* (2022) - https://vimeo.com/717188488.

[2] This article complements and extends Hege, A., Noufi, C., Wang, G., & Georgieva, E. (2021). "Instrument Design for *The Furies: A LaptOpera*." NIME 2021.

[3] The creative team included Curtis Ullerich, Ge Wang, Elena Georgieva, Camille Noufi, and Matt Wright.

[4] The instrument design team 2019-2020 included Anne Hege, Camille Noufi, Elena Geogieva, Ge Wang, and Matt Wright and 2020-2022 Curtis Ullerich and Anne Hege.

[5] Documentation of this project can be found on the project website - https://laptopera.org/.

[6] Link to video documentation of the premiere of Act III of *The Furies: A LaptOpera* (https://youtu.be/NScHznvVxrw).

soloists (Sidney Chen, Shauna Fallihee, Alice Del Simone, and Anne Hege). Between May 2020 and the premiere, computer programmer, instrument designer, and instrument ensemble control system designer Curtis Ullerich and composer and creator Anne Hege met regularly to develop and hone the instrument design throughout the opera. This time allowed the team to solve practical problems within the instrument design to create reliability, repeatability, and random access ability to support the rehearsal and performance of the final opera. This paper outlines critical design decisions made to fulfill this work's artistic vision and practical needs.

3. INSTRUMENT DESIGN

The Furies: A LaptOpera has fourteen unique instruments with multiple variations within a single instrument type. These variations lie within a spectrum of differences that could be compared to something as similar as a separate part (a second violin part) or as different as an alto flute or piccolo compared to a standard C flute. For this paper, we will focus on specific design successes that created clear principles of design for our project demands.

Figure 2: Performance setup for *The Furies*

3.1 The Rope Instrument in "Aegisthus' Prayer"

The rope instrument first appears in *The Furies* during "Aegisthus Prayer" in Act III as a border between the world of the furies/ghosts and the world of the living. In its first presentation, it is a length of rope that stretches across the back of the stage. The rope is attached to three GameTrak controllers, each connected to a laptop station with a co-located directional speaker. Mid-act, the furies gather behind the rope. They represent the demand for moral order and retribution that plagues Aegisthus. As Aegisthus sings his prayer, the furies respond to his questioning lament and plea for Apollo's guidance by theatrically pushing forward on the rope and collectively testing the barrier between their realm and the living world. A class in the ChucK programming language broadcasts GameTrak data to Wekinator over OSC and listens for five classification values from Wekinator.[11][2] Each classification value is mapped to the gain of a continuously looping sample. This instrument is parameterized with two arguments: Wekinator model name and a list of 5-sample banks of wav files, which performers cycle through by stepping on a button.

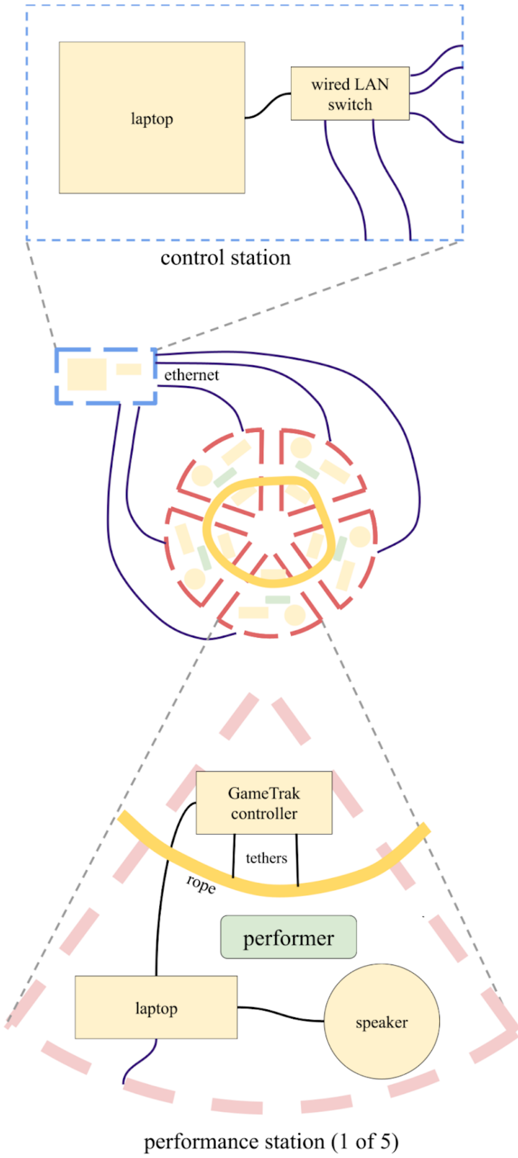

Figure 3: Physical layout of the rope instrument on stage Our setup has five performance stations with one laptop each, and a separate laptop to issue control commands to the performance stations. This is all connected into a LAN with wired ethernet switch.

Musically, the rope acts as an ambient drone using ghost-like samples filtered to the key of Bb Major, tonally supporting the vocal line. Using the Wekinator machine learning application, the instrument uses a classifier model trained on GameTrak tether input (left x, y, z and right x, y, z) to continuously mix five samples. The model outputs the mix (volume values for each sample), using the rope position as input (tether axis values, which track the points of attachment to the rope in space); the training[7] is the same for all players, but the samples are different for each of the three stations, resulting in different but complementary output even if all performers move in unison. The rope serves as a form of analog networking. Each person's movement impacts the movement of all, and the ensemble negotiates how to move together through feeling,

[7] When using the terms "model" and "training," they are interchangeable and directly reference the machine learning attributes of the Wekinator application utilized by the rope instrument.

sight, and responsive listening. Fast movement creates more texture and change in the sonic realm as the rope position moves through the spaces mapped to different mixes by the model. The instrument in this iteration allows for improvisation and responsive action. All samples support the key, and there are no wrong notes, allowing the performers to focus on textural, dynamic, and dramatic expression.

Figure 4: furies and ghosts play the rope in "Aegisthus' Prayer"

3.2 Three Instrument Design Principles

Performers in *The Furies* must think about singing, choreography, and playing an instrument simultaneously. Each performer plays several different instruments throughout the show. Limited access to equipment and rehearsal time with the ensemble required instruments that could be mastered in a limited time. With performers' attention already divided, the instruments must function in a way that does not burden performers with worries like: Will this work like it did last time? Will I trigger something wrong? Do I need to tiptoe around particular behavior because mistakes are hard to move past? Is my request for starting a piece over going to be hard to accommodate?

Consequently, when considering instrument design, one of the chief goals was to reduce performers' cognitive load while allowing a cohesive mental model of each instrument. This was accomplished through interaction design and guided by instrument design principles outlined below.

3.2.1 Principle 1: Random Access

Instruments should afford access to any of their available sounds and capabilities immediately. Consider how a piano allows random access to any note on the keyboard, as opposed to a piano roll which requires sequential access to the notes in a piece. A saxophone likewise affords random note access in a less obvious way via fingerings. Random access allows player agency in the performance of a piece. It affords mastery, which creates manageable challenges and fun. This makes playing the instrument an engaging experience for the performer. It also benefits the audience by visually communicating how the instrument works and demonstrating players' agency. Given that this work uses bespoke instruments (and many of them), this allows the audience a direct way to understand what is happening on stage. At the ensemble level, random access is likewise essential. The ensemble should be able to start at any scene, stop or start in the middle of a scene, and run the show multiple times. The rope instrument, for example, is trained to give specific gains to five samples based on the tether location. Every player can immediately access all possible sample gains as long as their movement is not physically limited by others (as they all hold the rope). This allows the ability to start at any moment within this movement for rehearsal or performance.

3.2.2 Principle 2: Repeatability

The same motion or action should produce the same sound every time. This feeds into audience understandability and allows skill and control by the performer. With repetition, the performer develops muscle memory as they master the instrument in an aural-physiological loop. In scenes where the rope is interacting with a machine learning model providing mix values for a bank of samples, the playable space remains consistent, allowing the performers to develop a memory of it over time, finely playing the textural and dynamic contrasts across the space by moving through it using practiced movements of the body. For example, in "Aegisthus' Prayer" with the Rope Instrument, each player comes to learn their sample bank and tether space so that they can build tension collectively through engagement with specific spatial locations. Over time, rope performers learn to play in response to each other, guiding individual and collective movement. Repeatability is crucial to the ensemble's ability to "play" the instrument.

3.2.3 Principle 3: Independence with Mutual Reinforcement

Each performer's instrument should be able to stand alone musically while reinforcing and complementing what other performers are doing. In "Aegisthus' Prayer," tuning the samples to reinforce the tonic key allows a mutual reinforcement of musical purpose. Textural and timbral elements support the dramatic action. Still, the rope instrument functions with one of three stations independently. This allows for the potential of human or machine error without losing the entire instrument. One station may fail to load, but this will not impact the sounding of the other two stations. Because they sonically reinforce each other, the soloist can continue with the functioning stations.

Figure 5: the furies perform "Glorious Guilt" with the Rope Instrument surrounding Electra and Orestia.

4. INSTRUMENT ENSEMBLE CONTROL SYSTEM DESIGN

To fulfill the artistic goal of creating instruments that are part of the stage setting, reinforce character relationships, and function as a part of the narrative, we were required to build new instruments and often multiple instruments for each scene. To coordinate and manage these instruments, their design updates, the transitions from scene to scene, rehearsing within a scene with the ability to stop and start at different musical moments, and create a control system where things can be restarted as needed due to human error or technical difficulties, the opera required an instrument ensemble control system. This system became the overarching architecture that housed and managed the running of each individual instrument and was crucial to the successful live performance of this ninety-minute work. Below are four principles that guided the creation of our instrument ensemble control system.

4.1 Four Instrument Ensemble Control System Design Principles

The first iteration of the instruments consisted of scripts run individually on performance station laptops by each performer as they entered their scene. By the full opera premiere, we had built a comprehensive instrument ensemble control system to manage all of this. The multi-year process of creating, rehearsing, revising, and remounting *The Furies* clarified some implementation priorities.

4.1.1 Principle 1: Robustness

Amidst the unpredictable conditions in rehearsal and performance, the ensemble should produce consistent output/results. The realities of playing a piece shift, so we accept this fact and design for it. Variables included but were not limited to the number of players available, the amount of time any component of the system has been running, whether every cable is firmly plugged in, whether the power on our network switch was turned on and off, the order in which we rehearsed the scenes, and changing which particular laptops were used for particular performance stations. We designed our code to perform consistently amidst these changing variables.

4.1.2 Principle 2: Fault Tolerance

When something inevitably goes wrong, the ensemble should keep running smoothly; we should be able to recover from it gracefully, and ideally, the audience shouldn't notice. Because instruments are designed with random access and mutual reinforcement, even restarting an instrument means we should not need to start at the top of a scene or halt the show. The absence of one station should not disrupt the musical continuity or impact the other stations' ability to keep running.

4.1.3 Principle 3: Reconfigurability

The ensemble's shape and size should be flexible. Each instrument should be usable by the entire ensemble, with or without a technician, and by a subset of the ensemble or a subset of stations.

4.1.4 Principle 4: Encapsulation

Each system component has clear boundaries and interfaces for interacting with it. This isolates components from one another so that failure/breakage is contained to one component. This design priority underpins all the others. This also provides a core point of software engineering stability, keeping our software more maintainable over time. By having clear interfaces between components, we can do automated testing of these units, enabling us to catch regressions as we modify the system.

5. INSTRUMENT ENSEMBLE CONTROL SYSTEM TO THE RESCUE

The Furies has many scenes and many instruments. Consequently, we need a mechanism to transition through them. Our instruments' hardware setup is a laptop+speaker that remains in the same position on stage throughout the whole show (see Figure 2). There is a tether controller connected that can change position and be attached to different accessories. We change from one instrument to another by changing the software running on the laptops.

Having performers individually manage the changing of instruments from scene to scene by interacting with laptops on stage causes a break in immersion and stymies narrative momentum. It also increases the surface area of mistakes made by having ten people change the state of five laptops multiple times in one show. Managing these transitions this way also requires all performers to have a more detailed knowledge of the inner workings of each instrument, the precise transition timing necessary for the show, and the technical knowledge to properly kill and launch ChucK scripts. This extra cognitive load makes it hard to focus on their primary dramatic roles.

In our instrument ensemble control system, each performance station is running a lightweight Kotlin server. The code for every instrument in the opera is wrapped in an Instrument class in Kotlin with templated start() and stop() methods which encapsulates the process management for both ChucK (our synthesis code) and Wekinator (our machine learning model). This server launches child processes for instruments and sends information and diagnostics in response to remote procedure calls (RPCs) from a control station (another laptop). This allows an on-stage or off-stage technician to interact with the ensemble of instruments as a whole without interrupting the flow of the performance.

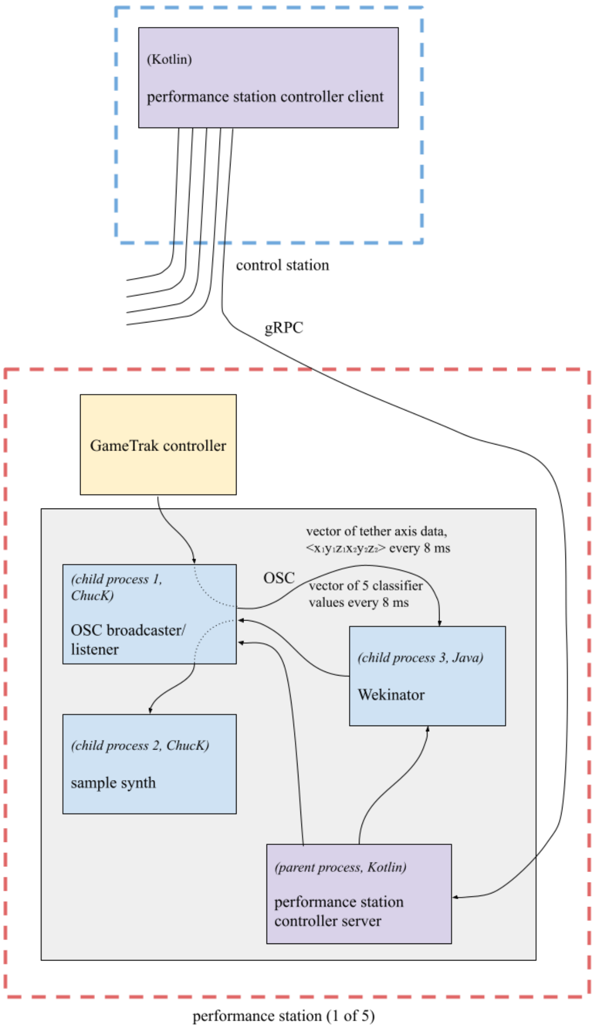

Figure 6: ensemble control system software architecture

In the event that something significant goes wrong, the technician can query the state of the whole ensemble and can control instruments with simple commands, e.g., StartInstrument(instrumentName) and StopInstrument (instrumentName).

This client-server architecture is stateless, meaning no process running on any machine directly depends on any process or data on another machine to keep its currently running processes healthy. Each component can be restarted independently without affecting the others. Components can reconnect after unexpected disconnection like a cable getting unplugged or a laptop running out of battery. This doesn't necessarily mean the system itself automatically recovers from all faults, but the system continues to run (the show goes on) and allows for the component to be corrected/restarted without stopping or restarting the whole ensemble.

With each performance station running an independent server, multiple clients can interact with it (send commands to start/stop instruments and query status information). We take advantage of this by using two client implementations: One is a run-of-show script that simply advances to the next cue in the opera each time a button is pressed. The other client is a command console that supports the gamut of available commands. We can add additional control stations to the LAN as necessary, e.g., to allow an off-stage technician to monitor what is happening and fine-tune the timing of a cue by simply opening another laptop and launching the control client which can then send RPCs to the already-running performance station servers.

As discussed above in our design principles and implementation priorities, having performance stations run independently and produce sound independently provides some inherent fault tolerance in our system. Their produced sound stands alone, but each station is mutually reinforcing. If, due to an error, a station is offline, a keen listener may notice, but it won't significantly affect the musical cohesion of the performance.

The instrument control system uses a configurable list of hosts to determine which role each laptop plays. The specific instrument and configuration loaded for a given scene depend on its location on stage. The isolation discussed above, and this configurability means we can run the rope instrument with any subset of performance stations, including just one host (which can run on the same machine as the control station), for lower-lift practice time.

Finally, if we need to make code or configuration changes on the fly, we have a simple packaging and deployment mechanism that works over the performance LAN. We can make a code or configuration change and have it running on the whole ensemble in two minutes, physically interacting only with the control station.

6. CONCLUSION

Through this article, we have explored practical and artistic considerations of instrument design and the creation of an instrument ensemble control system for *The Furies: A LaptOpera*. We have described practical design principles that create robust, reliable, and repeatable instruments that work within a system that considers the mental bandwidth of the performers. The practical principles and tools outlined above can be used to support the successful creation of instruments for a live performance production with both performance and rehearsal demands.

7. ACKNOWLEDGMENTS.

Our thanks to all performers of *The Furies: A LaptOpera,* including SLOrk 2019 (Munira Alimire, Hassan Estakhrian, Ben Gaiarin, Hillary Hermawan, Kunwoo Kim, Mark Sabini, Ryan Smith, Cara Turnbull, Jack Atherton, Trijeet Mukhopadhyay), SideLObe 2019 (Raul Altosaar, Andrea Baldioceda, Madeline Huberth, Mike Mulshine), Sidney Chen, Shauna Fallihee, Alice Del Simone, Catie Cuan, Amy Foote, Monica Covitt, Matt Wright, Nette Worthy, Cole Thomason-Redus, Brendan Larkin, Constantine Basica, David Kerr, and the CCRMA community.

8. ETHICAL STANDARDS

This project received support and funding from a CCRMA-funded Visiting Artist position (2019-2020) and a New Music USA Project Grant (2019) for the premiere of Act III. It also received funding through individual donations with fiscal sponsorship through Fractured Atlas. There were no potential conflicts of interest (financial or non-financial). We had informed consent when the research involved human participants. No animals were involved in this research. The code used in our instrument design is open source.

9. REFERENCES

[1] Cook, P.R., 2001. "Principles for Designing Computer Music Controllers," *In Proceedings of New Interfaces for Musical Expression.*

[2] Fiebrink, R. et al., (2010). "Toward Understanding Human-Computer Interaction in Composing the Instrument." *In Proceedings of the International Computer Music Conference 2010.*

[3] Hege, A. (2012). "Reclaiming What is Lost as Performance and Practice." *In Proceedings of the Symposium on Laptop Ensembles and Orchestras Conference 2012.*

[4] Hege, A., 2014. *Spirit in the Flesh: Essays on the Relationship between the Body and Meaning in Performance.* [Doctoral Dissertation, Princeton University] ProQuest Dissertations Publishing.

[5] Hege, A., Noufi, C., Wang, G., & Georgieva, E. (2021). "Instrument Design for *The Furies: A LaptOpera*." NIME 2021.

[6] Johnson, M., 2007. *The Meaning of the Body: Aesthetics of Human Understanding.* The University of Chicago Press.

[7] Smallwood, S., D. Trueman, P. R. Cook, and G. Wang. 2008. "Composing for Laptop Orchestra." *Computer Music Journal*. 32(1):9-25.

[8] Trueman, D. 2007. "Why a Laptop Orchestra?" *Organised Sound. 12(2):171-179.*

[9] Wang, G. 2018. *Artful Design: Technology in Search of the Sublime*. Stanford University Press.

[10] Wang, G., N. Bryan, J. Oh, and R. Hamilton. 2009. "Stanford Laptop Orchestra (SLOrk)." *In Proceedings of the International Computer Music Conference.* Montreal.

[11] Wang, G., P. R. Cook, S. Salazar. 2015. "ChucK: A Strongly-timed Computer Music Language." Computer Music Journal. 39(4):10-29.