



acm

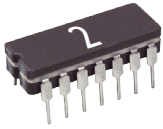
ACM

communications

Volume I, Edition 6

Carnegie Mellon

Image Courtesy: <http://www.nis>



power of the cloud

Shashank Pradhan (CS '11)

What is cloud computing? Is it something that happens when clouds interact in various ways? Or is it working on your computer while flying in an airplane? Neither are correct.

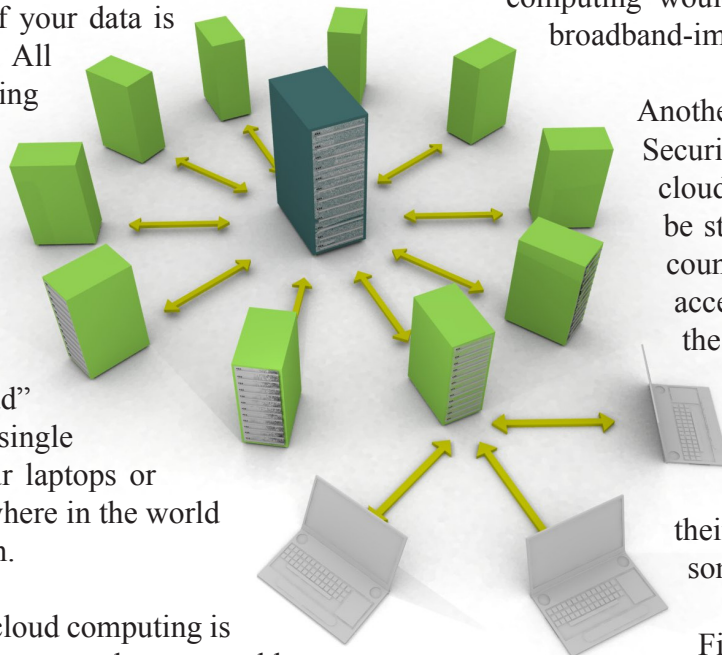
The “cloud” is a metaphor for the internet. The premise of cloud computing is that all you need to access all of your data is an internet connection. All the processing and saving of data takes place on a server that could be located anywhere in the world, provided you pay for the amount of data and processing time you use up. When we rent or buy a “cloud” we can access every single bit of information on our laptops or personal computers anywhere in the world via an internet connection.

One major advantage of cloud computing is the portability of data. For example, you could save 30 GB of our work on a cloud in New York, then fly to Tokyo on the same day and access all 30 GB of data on any computer with an internet connection. This would make USB drives and external hard disks completely redundant. You may not know it but we have already been using a version of cloud computing since quite some time. This cloud computing is called E-mail. Our emails are all read and saved online. Gmail, Hotmail, Yahoo mail, etc. There are a lot of companies that provide us with these “free” services.

Another advantage of cloud computing includes lower computer costs. Since all the applications run on the cloud, your PC/laptop doesn't need fast processing speeds or the hard disk space which we so desperately want and need right now. One could buy a netbook for

200\$-300\$ and run all their web apps online. This in turn leads to improved performance from your operating system. Also, we get virtually limitless storage space in clouds online.

However, cloud computing does have its set of drawbacks. One is the problem is the internet connection requirement. When internet connection fails on your laptop offline-work, such as working on a word document, can still be done. But with cloud computing, since all work is done on the cloud access is impossible without an internet connection. This also leads to the problem of slow connections. A low speed internet connection, such as dial-up services, makes cloud computing very slow and frustrating. Cloud computing would not be very useful for the broadband-impaired.



Another source of concern is security. Security is a major concern for all cloud providers. Your data could be stored on a server in a different country. Hackers could obtain access to all your data by infiltrating the server your data is kept in. In today's digital age nothing is secure, regardless of the amount of encryption and firewalls one puts to protect their data. There will always be someone smarter than you are.

Finally, you don't own your data anymore. The cloud hosting company does. They could start demanding higher prices from their consumers to make their data available. Unless the government presiding over the company or company's server restricts them from practicing these kinds of activities, you will be helpless against their demands.

Cloud computing is one of the fastest growing segments in the IT industry and it has a good reason for doing so. Many Silicon Valley companies like Google, Microsoft and IBM have already started development on their cloud computing clients. Cloud computing is spreading like wildfire as a novel way of using the internet and many are already using it on a full scale. However, before taking the plunge into cloud computing, be sure to research the pitfalls it has.

charlie garrod



Interview with Charlie Garrod, Post Doctorate Fellow

How many years have you been here at CMU?

I've been here since 2001 as a graduate student and graduated in 2008.

What courses have you taught?

I've taught web apps (15-437), fundamental data structures and algorithms (15-211) and co-taught distributed systems (15-440) with Dave Anderson.

Which research areas and technologies are you interested in?

My original research was in databases, distributed systems and scalability, but its shifting towards general distributed systems research. For example, I'm interested in power consumption, mobile systems and transportation infrastructure.

Could you talk about a project you're currently working on?

I'm currently working on a project to use data from cellphones to improve bus schedules. I.e., if you're waiting for a bus you might be interested in where the bus is at the moment. If there are people on the bus with cell phones, you could possibly find out if there was some app on their cell phones that sends their coordinates via GPS to some server, and your phone could access that data. Unfortunately, that way requires too much power from the phones and relies on people running an app when they get on the bus and turning the app off when they get off the bus. So instead of relying on data from a few people, you could determine the positions of everyone with a cellphone using triangulation off the cell phone towers, and using machine learning algorithms to determine if they are on the bus or not.

Could undergraduates assist in your research?

It may be too late. The position closes by the end of the spring semester, and normally the process of approving independent study credits takes a while. But they are welcome to try, if any are interested.

What would you say about going to graduate school over say, entering the workforce?

If you use the metric of percentage of people who graduate with a phd, then going straight to graduate school after college is better than entering the workforce with the intention of eventually going back and getting a graduate degree. On the other hand, if I had entered the workforce instead I might have gained maturity and expanded my interests.

What advice would you give to students in terms of expanding learning beyond coursework?

I'd tell them to get involved in programming team type activities, be a teaching assistant as much as you can for as many different courses as possible, and do non-technical things.

What do you think students get out of being a teaching assistant?

Teaching the material gives you a deeper understanding of the material. You'll also learn communication skills, and get paid for doing so. Not that getting paid should be the motivation there.

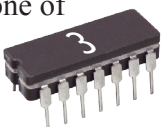
Wait, did you say to get involved in non-technical things? Why would we do that?

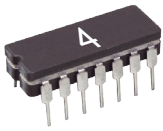
To be well rounded. I think its important to realize the context of computing technologies, something that gives you context for your work. For example, with my current project, its important to realize that, due to various political processes, transportation will essentially always remain underfunded. As a result, we have to become more efficient with what we already have. I came from a liberal arts college, in which one third of the courses were in my major, one third were in general sciences and the rest were in humanities and arts. So you can see where my bias comes from.

Is there anything else that you would like to communicate to our readers?

No. There are so many weird things, so I'll communicate none of them. You'd be surprised at the number of faculty here who can unicycle or juggle.* Also, I bought the domains luisvonahn.com and evilvonahn.com, and am open to ideas for their purpose. In fact I encourage people send me suggestions for their use, since otherwise the domain is just sitting there and I may default to giving it to Von Ahn.

* anonymous sources indicate Manuel Blum is one of them.





project olympus

A LOOK INSIDE

Nolan Hergert (ECE '12)

Do you have any ideas for projects that you want to sell someday? Would you like the opportunity to develop your ideas further into a possible startup? Project Olympus (a grant-funded entrepreneurial arm of the School of Computer Science) is an active and growing hub for students that want to investigate the business potential of their ideas and learn how to see them to fruition, a process that I started back in my freshman year at Carnegie Mellon.

My first startup was largely inspired by my frustrated attempts to find useful scholarships in my high school's counseling department. I wanted to alleviate that pain for other high school and college students, so I worked to create a service that would deliver accurate and relevant local scholarship information by aggregating and indexing local scholarship listings from existing counselor databases, other high school websites, and well-selected Google searches.

After visiting high schools back at home and getting some initial code working, I was stuck on one detail—how to make it scale up! I couldn't do all of the leg work on my own, and just plastering the site with Google Adwords wasn't going to pay enough to have people work for me. Luckily, I had heard about Project Olympus and thought it would be a good time to mention my startup idea.

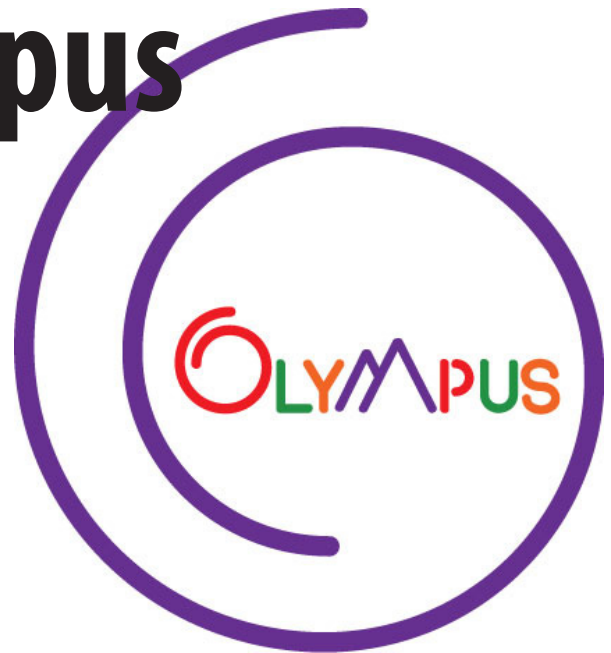
Over the next four months Kit Needham, Olympus' student advisor, helped me put my idea to the test as we investigated my market size, analyzed and learned from my competitors, thought of at least 5 half-decent ways how I could generate revenue without annoying my visitors, and even developed some value-added services that I hadn't thought of before. It was so much more fun than reading about business plans in a textbook and the best part was that I didn't have to sign my startup idea away or pay anything for

their help (something that you will have to do if you approach CMU with the idea). Although in the end we discovered that a well-established college-prep website was taking the same route, it was an awesome experience for me to play with my own idea and analyze its business potential.

If you have any ideas that you think could turn into potential startups or want to find out what other startups exist at CMU, I would definitely encourage you to meet Project Olympus' staff. They have a strong track record with their student-run startups, yielding three that have graduated to local incubator AlphaLab, one that has received major VC investment, and one that was bought out by Google! They offer additional services including free office space, micro-grants, and an extensive student and faculty contact list. Their frequent Show and Tell and Student CONNECTS events are also a great way to meet other students, hear about potential and successful startups, and perhaps even pitch your own idea.

Visit <http://olympus.cs.cmu.edu> to look at their current startups and to find out more.

If you would like to write for ACM Communications, subscribe, or get involved with ACM@CMU, e-mail us at cmu.acmc@gmail.com.



AuraFX

A SIMPLE AND FLEXIBLE APPROACH TO INTERACTIVE AUDIO EFFECT-BASED COMPOSITION AND PERFORMANCE

Roger B. Dannenberg
Robert Kotcher

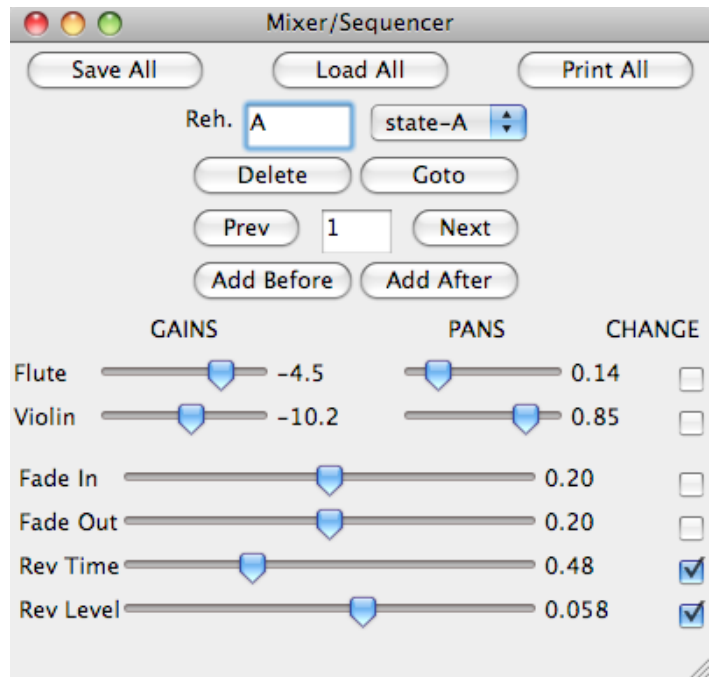
An interactive sound processor is an important tool for just about any modern composer. These days, performers and composers use interactive computer systems to process sound from live instruments. In many cases, audio processing could be handled using off-the-shelf signal processors. However, most composers favor a system that is more open-ended and extensible. Programmable systems are open-ended, but they leave many details to the composer, including graphical control interfaces, mixing and cross-fade automation, saving and restoring parameter settings, and sequencing through configurations of effects. Our work attempts to establish an architecture that provides these facilities without programming. It factors the problem into a framework, providing common elements for all compositions, and custom modules, extending the framework with unique effects and signal processing capabilities. Although we believe the architecture could be supported by many audio programming systems, we have created a particular instantiation (AuraFX) of the architecture using the Aura system.

Interactive music compositions span a wide range of organizations, intentions, and implementations. Most music in this category are labeled “experimental,” and receives very open-ended and general software for development. One of the reasons for generality is to avoid falling into the trap of the “paint-by-number” approach, where only a fixed set of effects, timbres, or patches are available to choose from. This makes the music recognizable and unoriginal, something composers try to avoid.

As computer music has matured and more systems have become available, the “paint-by-number” approach has become more acceptable for several reasons: (1) more effects and sound processing systems are available, so the composer is not limited to a small set of choices; (2) modern computing power has enabled more flexibility, parameter choices, and other ways to customize off-the-shelf effects; (3) computer music systems are widely available and attractive to non-expert programmers. When computer music required considerable engineering

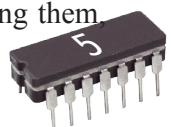
skill, it was normal for musicians to do a considerable amount of programming, but today, many musicians find it acceptable to use relatively pre-packaged engineering solutions in order to instead focus their creativity on music composition and performance.

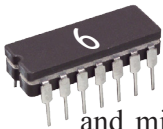
Our implementation of the AuraFX system follows a



tradition of trying to understand the general nature of some broad class of computer music systems and make it easy to create new systems in that class. Ideally, it should allow the composer or sound engineer to accomplish tasks with a minimum of effort and a maximum of flexibility. The architecture that underlies AuraFX is simple: a sequence of sets of effects with adjustable levels, fade-in and fade-out times, and channel assignments. The effects have a well-defined interface with the system so that new effects can be constructed and “plugged in” to the architecture. Transitions from one set of effects to another can be sequential (e.g. advanced by a simple pedal interface or timer) or controlled by an external program.

An important issue in our design is the user’s technical competence. Many composers say they use a visual programming system such as MAX MSP [1] or Pd [2] because they “don’t know how to program,” but even these visual programming systems require programming. Developing patches, controlling them, sequencing them

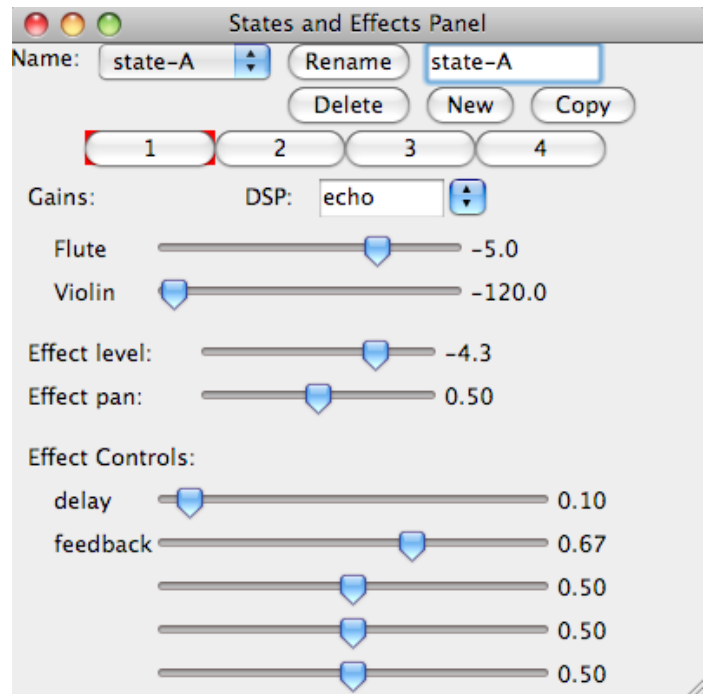




and mixing their outputs all requires programming, yet these languages have at best a limited way to organize and accomplish these programming tasks. AuraFX imposes much more structure and in return requires much less programming than even visual programming systems. In fact, the only programming available in AuraFX is the writing of effects using either a scripting language or a dataflow-like visual programming language. AuraFX has built-in mechanisms for allocating, mixing, and sequencing effects. On the other hand, AuraFX is more open-ended than an automated mixer because it is not restricted to sequential or time-based changes and it is particularly adept at dynamically scheduling and managing multiple effects which might overwhelm the processor(s) if all of the effects were set up at once on different effect-send busses.

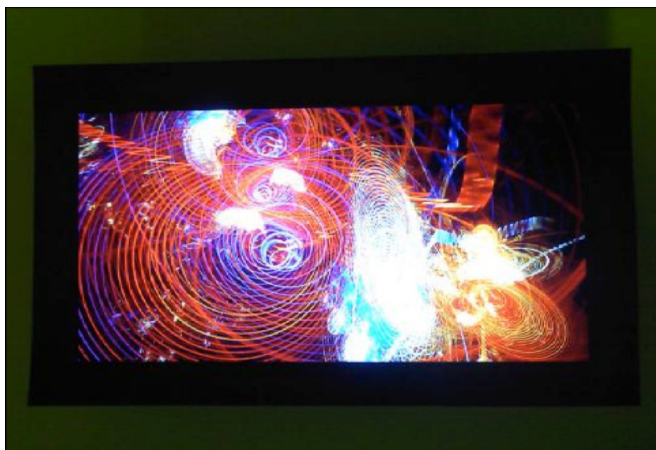
AuraFX has several motivations. It was originally created for a performance by the Pittsburgh New Music Ensemble (PNME) at the Edinburgh Festival Fringe. The PNME planned to use extensive audio processing but needed a system that could be rapidly reconfigured after arriving at the performance space. Although the ensemble abandoned their plans for electronics as too ambitious, the discussion forced us to think about how to create a flexible audio processing system for use by non-programmers. The second motivation is the first author's own work with Aura, an open-ended system for interactive multimedia. One of the things we noticed in using Aura is that

much of the most difficult programming effort in actual compositions had to do with managing transitions: making sure effects are running and connected when needed, making soft switches rather than abrupt connections, and building interfaces not only for normal operation, but for rehearsals and debugging. Once AuraFX was started, we realized it could also be used in improvisational settings where effects can be called up at will (for example using foot pedals or algorithmic selection), and the system can be extended over time in a modular way.



tech pics

Nolan Hergert (ECE '12)



artfulelectricsheep The colorful and artsy morphing animation that is prominently displayed on the 5th floor of Gates is actually the work of thousands of computers and their human controllers working together to create beautiful fractal imagery. Using genetic cross-breeding algorithms and a user-controlled voting system, each fractal (called a "sheep") adapts and changes to make amazing new works of computer-generated art.



binary=>ascii Do the binary codes written on the seats in Rashid Auditorium mean anything? After further investigation, we have concluded that while the binary patterns seem not to repeat, they do not map to any meaningful ASCII words. Better luck next time!